USB Simply Buffered (USB)

Mass Storage Class – Bulk-Only Transport

Environment

- Debian Etch 4.0.
- x86 system.
- Sandisk Cruzer Mini 256 MB USB thumb drive.

The mass storage devices that comply with USB Mass Storage Class Specifications are supported by the USB Mass Storage Class Working Group. The various specifications are:

- USB Mass Storage Class Control/Bulk/Interrupt (CBI) Transport
- USB Mass Storage Class Bulk-Only Transport
- USB Mass Storage Class UFI Command Specification
- USB Mass Storage Class Bootability Specification
- USB Mass Storage Class Compliance Test Specification

Most mass-storage devices follow the Bulk-Only Transport specification. This specification is available in usbmassbulk_10.pdf. Reading "USB Simply Buffered – Device Enumeration" is a pre-requisite for this documentation.

There is a Command/Data/Status protocol that is followed:

- Command Block Wrapper (CBW)
- Data Transfer (IN/OUT)
- Command Status Wrapper (CSW)

SCSI Primary Commands -2 (SPC-2) specification, and SCSI Block Commands – 2 (SBC-2) commands are used by USB, and follow SCSI emulation for communication with the device for mass storage devices. The data transfer phase is optional if the host and device decide not to have one.

So, the commonly used commands are:

- GetMaxLun
- INQUIRY
- TEST UNIT READY
- READ CAPACITY
- MODE SENSE
- REQUEST SENSE
- READ
- WRITE

The Sandisk USB thumb drive was connected to a GNU/Linux host PC, and it got automounted. A small file was copied to the disk, and finally unmounted.

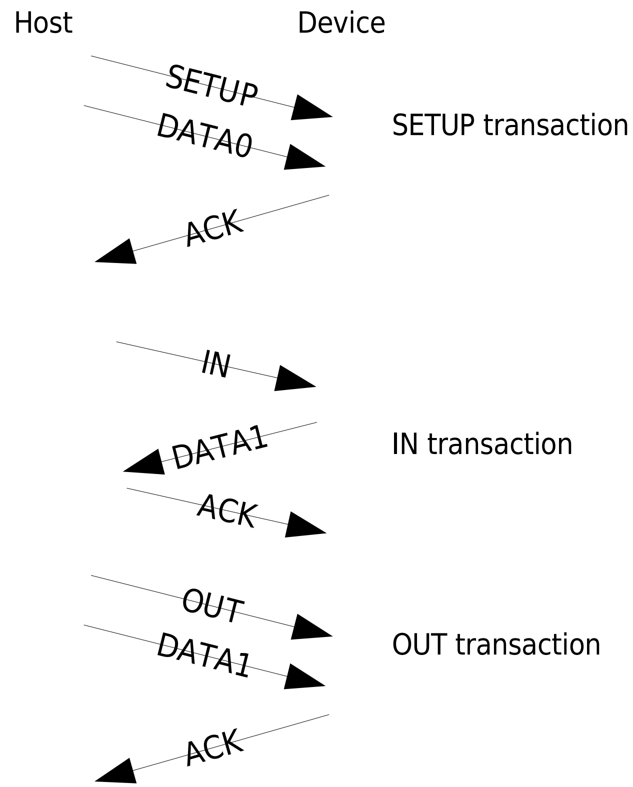
The commands can be issued anytime when the device is in configured state.

The packet details on the transactions for each of the requests/commands are explained in detail:

1. GetMaxLun

GetMaxLun request consists of the following transactions:

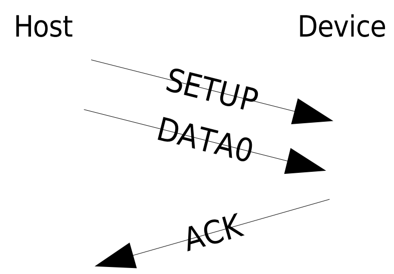
- SETUP transaction (->)
- IN transaction (<-)
- OUT transaction (->)



1.1 SETUP transaction

The SETUP transaction has the following three packets:

- SETUP packet (->)
- DATA0 packet (->)
- ACK packet (<-)



1.1.1 SETUP packet

A SETUP packet consists of:

	L	M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L	M	L M	L M	L M
	1011	0100	1100 000	0000	01010
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
01010 0000 000 0011 0010 1101

Regrouping in groups of 4-bits:

M L
0101 0000 0000 0011 0010 1101

Hex values:

M L
5 0 0 3 2 D

Regrouping as a byte:

M L
50 03 2D

Summary (SETUP packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	00101101	00000011	2D	03
02	01010000		50	

1.1.2 DATA0 packet

A DATA0 packet consists of:

	L	M	L M	L M
	PID	!PID	SETUP Data	CRC
(bits)	4	4	64	16

	L	M	L M	L M
	1100	0011	SETUP Data	CRC
(bits)	4	4	64	16

The PID arranged in MSB to LSB order:

	M	L
	1100	0011
(bits)	4	4

Hex values (PID):

M L
C 3

Since this is for a SETUP packet, the data consists of 8 bytes. The format is:

	L M	L M	L M	L M	L M
	bmRequestType	bRequest	wValue	wIndex	wLength
(bytes)	1	1	2	2	2

The 8 bytes for SETUP details are as follows (in decimal):

	L M	L M	L M	L M	L M
	10000 10 1	0111 1111	0000 0000 0000 0000	0000 0000 0000 0000	1000 0000 0000 0000
(bytes)	1	1	2	2	2

bmRequestType:

D7: Data transfer direction
1 = Device-to-host

D6...D5: Type
1 = Class

D4...D0: Recipient
1 = Interface

bRequest:

GET_MAX_LUN request code is 0xFE.

wValue:

wValue is 0.

wIndex:

wIndex is interface number, which is interface zero.

wLength:

wLength is set to one.

Putting it in MSB to LSB order:

	M				L
	0000 0000 0000 0001	0000 0000 0000 0000	0000 0000 0000 0000	1111 1110	1010 0001
(bytes)	2	2	2	1	1

In Hex:

M				L
00 01	00 00	00 00	FE	A1

The CRC observed in the sample capture is:

M L
00011111 01101010
16

(bits)

CRC (in Hex):

M L
1F 6A
16

(bits)

Putting it all (PID + SETUP DATA + CRC) together,

Summary (DATA0 packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11000011	10100001	C3	A1
02	11111110	00000000	FE	00
04	00000000	00000000	00	00
06	00000000	00000001	00	01
08	00000000	01101010	00	6A
10	00011111		1F	

1.1.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

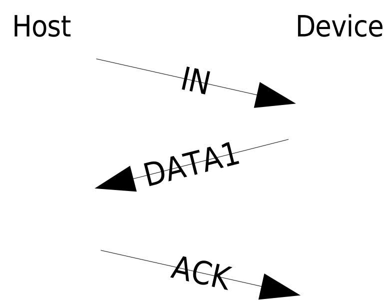
Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

1.2 IN transaction

The IN transaction has the following three packets:

- IN packet (->)
- DATA1 packet (<-)
- ACK packet (->)



1.2.1 IN packet

A IN packet consists of:

	L PID	!PID M	L Function Address M	L Endpoint M	L CRC M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L	M L	M L	M L	M
	1001	0110	1100 000	0000	01010
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
01010 0000 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L
0101 0000 0000 0011 0110 1001

Hex values (as a byte):

M L
50 03 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	00000011	69	03
02	01010000		50	

1.2.2 DATA1 packet

A DATA1 packet for GetMaxLun returns the number of LUNs (Logical Unit Numbers) supported by the device. If the device has 0 to 3 LUNs, it returns 3. If there is no LUN associated with, as in our case, it simply returns zero:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	8	16

Values:

	L	M	L	M
	1101	0010	Data	CRC
(bits)	4	4	8	16

============================ *BEGIN* ==============================
Get Max LUN

Table 3-2: Get Max LUN

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10100001	11111110	0000	Interface	0001	1 byte

Reference: USB Mass Storage Class – Bulk Only Transport: 3.2 Get Max LUN, page 7
============================= *END* ===============================

So, the the device returns 00 in the data field.

The CRC observed in the sample capture is:

M	L
10111111 01000000	
(bits) 16	

CRC (in Hex):

M	L
BF 40	
(bits) 16	

Putting it together (PID + 1 byte of data + CRC):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01001011	00000000	4B	00
02	01000000	10111111	40	BF

1.2.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

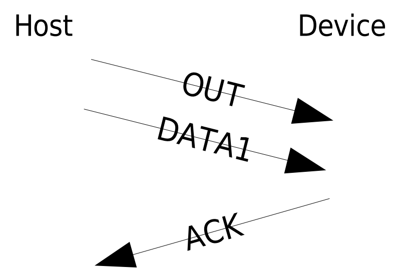
Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

1.3 OUT transaction

The OUT transaction has the following three packets:

- OUT packet (->)
- DATA1 packet (->)
- ACK packet (<-)



1.3.1 OUT packet

An OUT packet consists of:

	L M	L M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L M	L M	L M	L M	L M
	1000	0111	1100 000	0000	01010
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
01010 0000 000 0011 1110 0001

Regrouping in groups of 4-bits:

M L
0101 0000 0000 0011 1110 0001

Hex values (as a byte):

M L
50 03 E1

Summary (OUT packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11100001	00000011	E1	03
02	01010000		50	

1.3.2 DATA1 packet

The DATA1 packet has no data. It consists of:

	L	M	L M
	PID	!PID	CRC
(bits)	4	4	16

Values:

	L	M	L M
	1101	0010	CRC
(bits)	4	4	16

The CRC observed in the sample capture is:

	M L
	00000000 00000000
(bits)	16

CRC (in Hex):

	M L
	00 00
(bits)	16

The packet arranged in MSB to LSB order:

	M		L
	00000000 00000000	0100	1011
(bits)	16	4	4

In groups of 4 bits:

M L
0000 0000 0000 0000 0100 1011

In Hex,

M	L
00 00 4B	

Summary (DATA1 packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01001011	00000000	4B	00
02	00000000		00	

1.3.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M	L
1101	0010

Hex values:

M	L
D	2

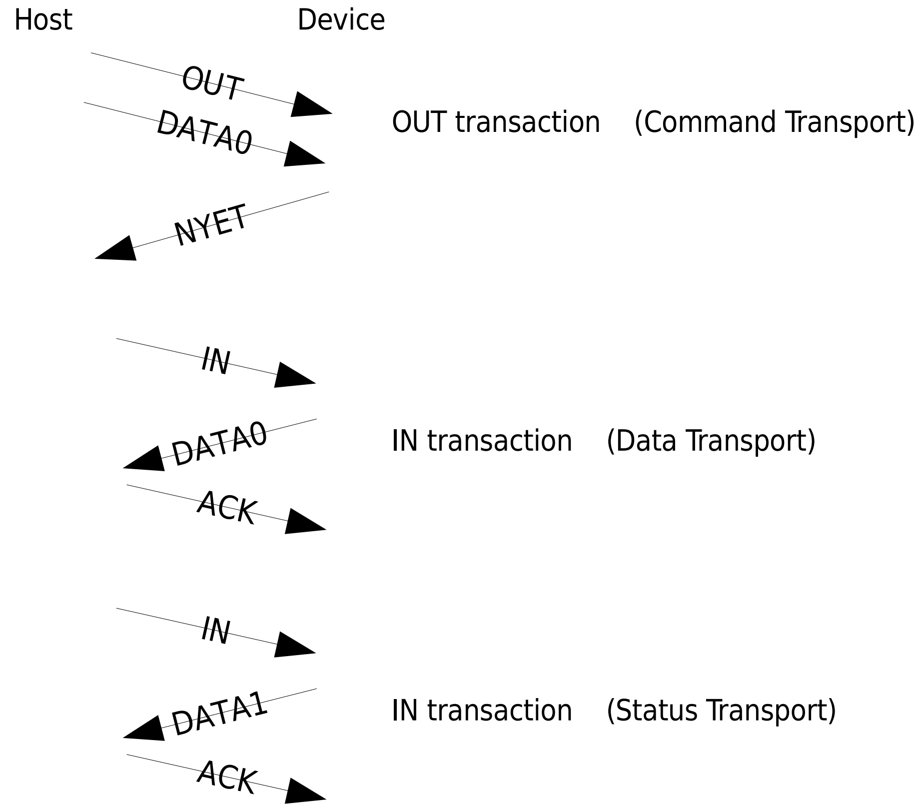
Summary (ACK packet):

Offset	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
	0	0
00	11010010	D2

2. INQUIRY

INQUIRY command consists of the following transport phases and transactions:

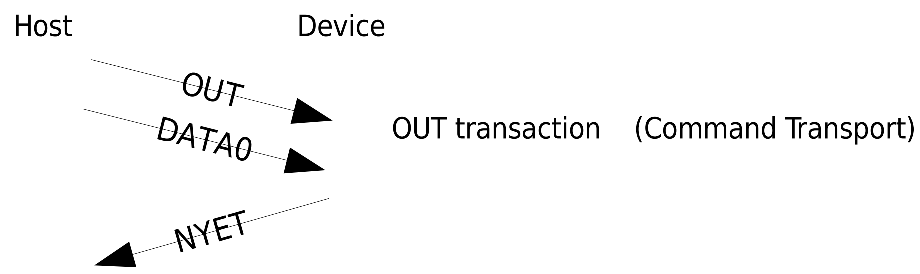
- Command Transport
    - OUT transaction (->)
- Data Transport
    - IN transaction (<-)
- Status Transport
    - IN transaction (<-)



2.1 OUT transaction (Command Transport)

The OUT transaction has the following three packets:

- OUT packet (->)
- DATA0 packet (->)
- NYET packet (<-)



2.1.1 OUT packet

An OUT packet consists of:

	L M	L M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L M	L M	L M	L M	L M
	1000	0111	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L

11100 0001 000 0011 1110 0001

Regrouping in groups of 4-bits:

M L

1110 0000 1000 0011 1110 0001

Hex values (as a byte):

M L

E0 83 E1

Summary (OUT packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11100001	10000011	E1	83
02	11100000		E0	

2.1.2 DATA0 packet

A DATA0 packet for Command Transport consists of 31 bytes of Command Block Wrapper data. It consists of:

	L M	L		M
	PID	!PID	Data	CRC
(bits)	4	4	248	16

Values:

	L M	L		M
	1100	0011	Data	CRC
(bits)	4	4	248	16

============================ BEGIN =================================
Command Block Wrapper (CBW)

Table 5-1: Command Block Wrapper

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCBWSignature							
7-4	dCBWTag							
11-8	dCBWDataTransferLength							
12	bmCBWFlags							
13	Reserved (0)				bCBWLUN			
14	Reserved (0)			bCBWBLength				
30-15	CBWCB							

dCBWSignature:
    The value for CBW is 0x43425355.

dCBWTag:
    A tag sent by host, which will be echoed back by device in dCSWTag.

dCBWDataTransferLength:
    The number of bytes of data transfer during data transport phase. If zero, there will be no data transfer.

bmCBWFlags:
    Bit 7: Direction
        0 = Data Out, from host to device
        1 = Data In, from device to host
    Bit 6: Obsolete. Set to zero.
    Bit 5...0: Reserved. Set to zero.

bCBWLUN:

The device Logical Unit Number (LUN) to which command block is sent.

bCBWCBLength:

Valid length of CBWCB in bytes.

CBWCB:

The command block to be executed by the device.

Reference: USB Mass Storage Class, Bulk-Only Transport, page 13

============================ *END* ===============================

So, the data for our capture was:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCBWSignature (0x43425355)							
7-4	0x00000001							
11-8	dCBWDataTransferLength 36 bytes, (0x00000024)							
12	Device to host (0x80)							
13	Reserved (0x0)				bCBWLUN (0x0)			
14	Reserved (0x0)			bCBWBLength 6 bytes (0x6)				
30-15	CBWCB							

The command block CBWCB has the INQUIRY command details.

============================ *BEGIN* ================================
INQUIRY

Table 45: INQUIRY command

Byte Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (0x12)							
1	Reserved						CMDDT	EVPD
2	PAGE OR OPERATION CODE							
3	Reserved							
4	ALLOCATION LENGTH							
5	CONTROL							

OPERATION CODE:
Value is 0x12.

CMDDT:
This bit specifies if device shall return the optional command support data or not.

EVPD:
Enable Vital Product Data to one specifies that device shall return vital produce data specified by PAGE OR OPERATION CODE field.

If both EVPD and CMDDT are zero, then standard INQUIRY data is returned.

Reference: SCSI Primary Commands -2 (SPC-2), Page 80
============================= *END* ==================================

So, the data for our CBWB capture was:

Byte Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (0x12)							
1	Reserved (0x0)						0	0
2	0x0							
3	Reserved (0x0)							
4	ALLOCATION LENGTH (0x24)							
5	CONTROL (0x00)							

There are ten more bytes to filled with 0x00 to fill the size of 30-15 bytes of the CBW.

The CRC observed in the sample capture is:

	M	L
	01011001 11010110	
(bits)	16	

CRC (in Hex):

	M	L
	59 D6	
(bits)	16	

Putting it together (PID + 31 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	11000011	01010101	01010011	01000010	C3	55	53	42
04	01000011	00000001	00000000	00000000	43	01	00	00
08	00000000	00100100	00000000	00000000	00	24	00	00
12	00000000	10000000	00000000	00000110	00	80	00	06
16	00010010	00000000	00000000	00000000	12	00	00	00
20	00100100	00000000	00000000	00000000	24	00	00	00
24	00000000	00000000	00000000	00000000	00	00	00	00
28	00000000	00000000	00000000	00000000	00	00	00	00
32	11010110	01011001			D6	59		

2.1.3 NYET packet

A NYET packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0110	1001
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1001 0110

Hex values:

M L
9 6

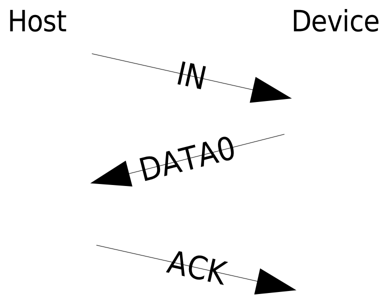
Summary (NYET packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	10010110	96

2.2 IN transaction (Data Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA0 packet (<-)
- ACK packet (->)



2.2.1 IN packet

A IN packet consists of:

	L	M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L	M	L M	L M	L M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L
1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L
E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

2.2.2 DATA0 packet

A DATA0 packet for INQUIRY command provides the 36 bytes of data. It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	288	16

Values:

	L	M	L	M
	1100	0011	Data	CRC
(bits)	4	4	288	16

============================ *BEGIN* ==============================
INQUIRY data

Table 46: Standard INQUIRY data format

Bit Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	RMB	Reserved						
2	VERSION							
3	AERC	Obsolete	NORMACA	HISUP	RESPONSE DATA FORMAT			
4	ADDITIONAL LENGTH (n-4)							
5	SCCS	Reserved						
6	BQUE	ENCSERV	VS	MULTIP	MCHNGR	Obsolete	Obsolete	ADDR16
7	RELADR	Obsolete	WBUS16	SYNC	LINKED	Obsolete	CMDQUE	VS
8 15	(MSB) VENDOR IDENTIFICATION (LSB)							
16 31	(MSB) PRODUCT IDENTIFICATION (LSB)							
32 35	(MSB) PRODUCT REVISION LEVEL (LSB)							
36 55	VENDOR SPECIFIC							
56	Reserved				CLOCKING		QAS	IUS
57	Reserved							
58 59	(MSB) VERSION DESCRIPTOR 1 (LSB)							
	...							

72	(MSB) VERSION DESCRIPTOR 8
73	(LSB)
74	Reserved
95	
	Vendor specific parameters
96	Vendor specific
n	

Reference: SCSI Primary Commands -2 (SPC-2), Page 82

============================ *END* ================================

Since, there are lot of fields, we will only discuss the relevant values to the fields used in our example. The first 36 bytes are returned to the host.

PERIPHERAL QUALIFIER:
  Whether the device is capable of supporting on this unit. 000b means connected to this logical unit (Table 47).

PERIPHERAL DEVICE TYPE:
  Direct-access device, 0x00 (Table 48)

RMB:
  Removable Medium Bit set to one indicates device is removable.

VERSION:
  Implementation version of standard used, ANSI X3.131: 1994, 0x02 (Table 49).

AERC:
  Asynchronous Event Reporting Capability (AERC). Not supported.

NORMACA:
  Normal ACA Supported bit. Not supported.

HISUP:
  Hierarchical support for hierarchical addressing mode of LUNs. Not supported.

RESPONSE DATA FORMAT:
  Follows this standard format, 0x2.

ADDITIONAL LENGTH:
  Length of the parameters passed, 0x1F (31 bytes).

SCCS:
  SCC Supported bit if device has embedded storage array controller. Not supported.

BQUE:
  Basic Queueing. Not supported.

ENCSERV:
  Enclosure Services. Not supported.

VS:
Vendor Specific.

MULTIP:
Multi Port indicates that the device contains an enbedded enclosure services component. No component.

MCHNGR:
Medium Changer indicates that device is attached to a medium transport element. No component.

ADDR16:
Specific to SPI-3.

RELADR:
Relative Addressing support. Not supported.

WBUS16:
Specific to SPI-3.

SYNC:
Specific to SPI-3.

LINKED:
Device supports linked commands. Not supported.

CMDQUE:
Device supports tagged tasks (command queuing). Not supported.

VENDOR IDENTIFICATION:
8 bytes. “SanDisk ”

PRODUCT IDENTIFICATION:
16 bytes. “Cruzer Mini “

PRODUCT REVISION LEVEL:
4 bytes. “0.1 “

The CRC observed in the sample capture is:

M	L
10011110 11001010	

(bits) 16

CRC (in Hex):

M	L
9E CA	

(bits) 16

Putting it together (PID + 36 bytes of data + CRC):

	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
Offset	0	1	2	3	0	1	2	3
00	11000011	00000000	10000000	00000010	C3	00	80	02
04	00000010	00011111	00000000	00000000	02	1F	00	00
08	00000000	01010011	01100001	01101110	00	53	61	6E
12	01000100	01101001	01110011	01101011	44	69	73	6B
16	00100000	01000011	01110010	01110101	20	43	72	75
20	01111010	01100101	01110010	00100000	7A	65	72	20
24	01001101	01101001	01101110	01101001	4D	69	6E	69
28	00100000	00100000	00100000	00100000	20	20	20	20
32	00100000	00110000	00101110	00110001	20	30	2E	31
	00100000	11001010	10011110		20	CA	9E	

2.2.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

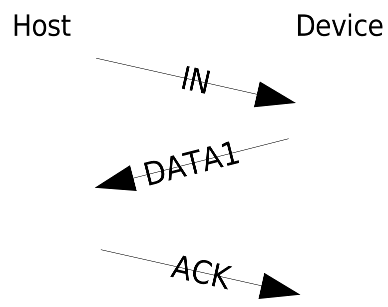
Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

2.3 IN transaction (Status Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA1 packet (<-)
- ACK packet (->)



2.3.1 IN packet

A IN packet consists of:

	L ... M	L ... M	L ... M	L ... M	L ... M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L ... M	L ... M	L ... M	L ... M	L ... M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M ... L
11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M ... L
1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M ... L
E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

2.3.2 DATA1 packet

A DATA1 packet for INQUIRY command (Status Transport) provides the 13 bytes of Command Status Wrapper (CSW). It consists of:

	L	M L		M
	PID	!PID	Data	CRC
(bits)	4	4	104	16

Values:

	L	M L		M
	1101	0010	Data	CRC
(bits)	4	4	104	16

============================ *BEGIN* ==============================
Command Status Wrapper (CSW)

Table 5.2: Command Status Wrapper

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCSWSignature							
7-4	dCSWTag							
11-8	dCSWDataResidue							
12	bCSWStatus							

dCSWSignature:
The value for CSW is 0x53425355.

dCSWTag:
A tag sent by the host as dCBWTag will be echoed back here as dCSWTag.

dCSWDataResidue:
Difference between data expected and data processed by device.

bCSWStatus:
Success or failure of the command.
00 = Command passed.
01 = Command failed.
02 = Phase error.
03, 04 = Reserved (obsolete).
05 to FF = Reserved.

Reference: USB Mass Storage Class, Bulk-Only Transport, page 13
============================ *END* ================================

So, the 13 bytes of data in our example is as follows:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCSWSignature (0x53425355)							
7-4	dCSWTag (0x00000001)							
11-8	dCSWDataResidue (0x00000000)							
12	bCSWStatus, Command Passed (0x00)							

The CRC observed in the sample capture is:

M	L
00100010 01010100	

(bits) 16

CRC (in Hex):

M	L
22 54	

(bits) 16

Putting it together (PID + 13 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	01001011	01010101	01010011	01000010	4B	55	53	42
04	01010011	00000001	00000000	00000000	53	01	00	00
08	00000000	00000000	00000000	00000000	00	00	00	00
12	00000000	00000000	01010100	00100010	00	00	54	22

2.3.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

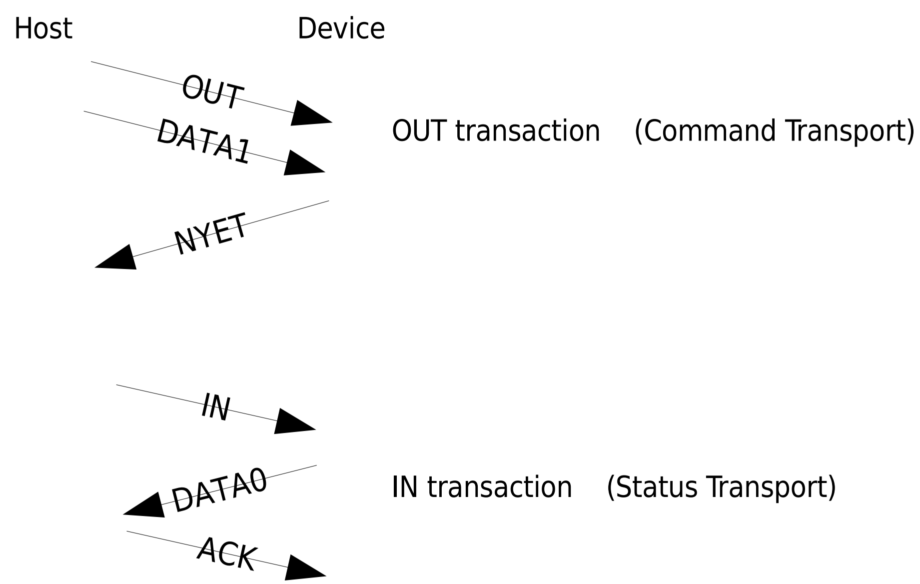
Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

3. TEST UNIT READY

TEST UNIT READY command consists of the following transport phases and transactions:

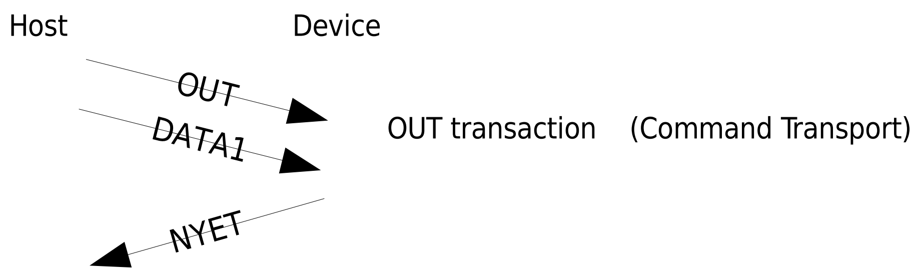
- Command Transport
    - OUT transaction (->)
- Status Transport
    - IN transaction (<-)



3.1 OUT transaction (Command Transport)

The OUT transaction has the following three packets:

- OUT packet (->)
- DATA1 packet (->)
- NYET packet (<-)



3.1.1 OUT packet

An OUT packet consists of:

	L M	L M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L M	L M	L M	L M	L M
	1000	0111	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
11100 0001 000 0011 1110 0001

Regrouping in groups of 4-bits:

M L
1110 0000 1000 0011 1110 0001

Hex values (as a byte):

M L
E0 83 E1

Summary (OUT packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11100001	10000011	E1	83
02	11100000		E0	

3.1.2 DATA1 packet

A DATA1 packet for Command Transport consists of 31 bytes of Command Block Wrapper data. It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	248	16

Values:

	L	M	L	M
	1101	0010	Data	CRC
(bits)	4	4	248	16

So, the data for our capture was:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCBWSignature (0x43425355)							
7-4	0x00000002							
11-8	dCBWDataTransferLength 0 bytes, (0x00000000)							
12	Host to device (0x00)							
13	Reserved (0x0)				bCBWLUN (0x0)			
14	Reserved (0x0)			bCBWBLength 6 bytes (0x6)				
30-15	CBWCB							

The command block CBWCB has the TEST UNIT READY command details.

============================ *BEGIN* ================================
TEST UNIT READY

Table 116: TEST UNIT READY command

Byte Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (0x00)							
1	Reserved							
2	Reserved							
3	Reserved							
4	Reserved							
5	CONTROL							

OPERATION CODE:
Value is 0x00.

Reference: SCSI Primary Commands -2 (SPC-2), Page 163
============================== *END* ================================

The CRC observed in the sample capture is:

M	L
01110111 10101010	
(bits) 16	

CRC (in Hex):

M	L
77 AA	
(bits) 16	

Putting it together (PID + 31 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	01001011	01010101	01010011	01000010	4B	55	53	42
04	01000011	00000010	00000000	00000000	43	02	00	00
08	00000000	00100100	00000000	00000000	00	00	00	00
12	00000000	00000000	00000000	00000110	00	00	00	06
16	00000000	00000000	00000000	00000000	00	00	00	00
20	00000000	00000000	00000000	00000000	00	00	00	00
24	00000000	00000000	00000000	00000000	00	00	00	00
28	00000000	00000000	00000000	00000000	00	00	00	00
32	10101010	01110111			AA	77		

3.1.3 NYET packet

A NYET packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0110	1001
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1001 0110

Hex values:

M L
9 6

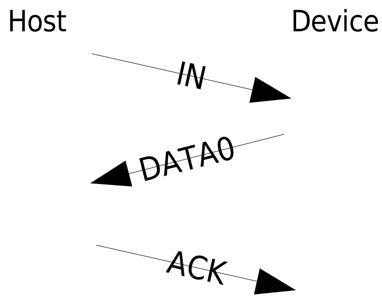
Summary (NYET packet):

Offset	Binary (M...L) Offset 0	Hexadecimal (M...L) Offset 0
00	10010110	96

3.2 IN transaction (Status Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA0 packet (<-)
- ACK packet (->)



3.2.1 IN packet

A IN packet consists of:

	L PID	!PID M	L Function Address M	L Endpoint M	L CRC M
(bits)	4	4	7	4	5

Values:

	L 1001	0110 M	L 1100 000 M	L 1000 M	L 00111 M
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M ... L
11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M ... L
1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M ... L
E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

3.2.2 DATA0 packet

A DATA0 packet for TEST UNIT READY command (Status Transport) provides the 13 bytes of Command Status Wrapper (CSW). It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	104	16

Values:

	L	M	L	M
	1100	0011	Data	CRC
(bits)	4	4	104	16

So, the 13 bytes of data in our example is as follows:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCSWSignature (0x53425355)							
7-4	dCSWTag (0x00000002)							
11-8	dCSWDataResidue (0x00000000)							
12	bCSWStatus, Command Passed (0x00)							

The CRC observed in the sample capture is:

	M L
	11010010 01000000
(bits)	16

CRC (in Hex):

	M L
	D2 40
(bits)	16

Putting it together (PID + 13 bytes of data + CRC):

	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
Offset	0	1	2	3	0	1	2	3
00	11000011	01010101	01010011	01000010	C3	55	53	42
04	01010011	00000010	00000000	00000000	53	02	00	00
08	00000000	00000000	00000000	00000000	00	00	00	00
12	00000000	00000000	01000000	11010010	00	00	40	D2

3.2.3 ACK packet

An ACK packet consists of:

	L PID	!PID M
(bits)	4	4

Values:

	L 0100	1011 M
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L

1101 0010

Hex values:

M L

D 2

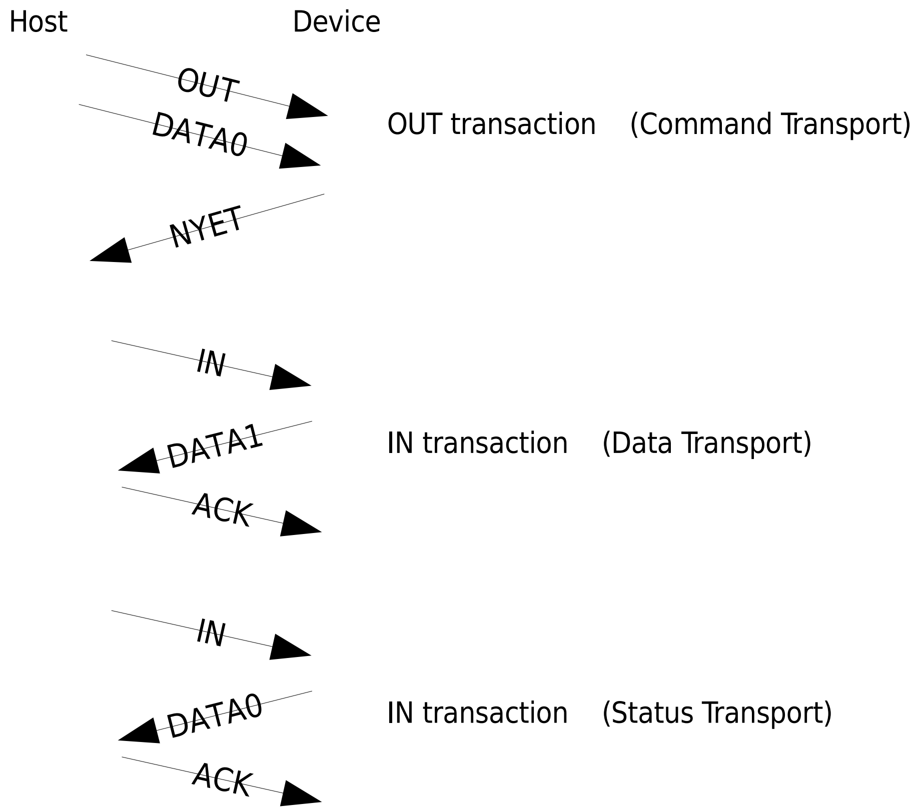
Summary (ACK packet):

Offset	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
	0	0
00	11010010	D2

4. READ CAPACITY

READ CAPACITY command consists of the following transport phases and transactions:

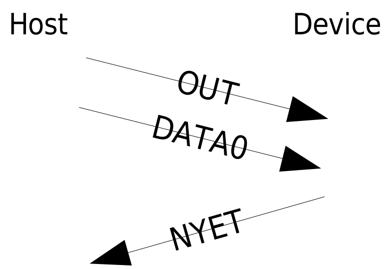
- Command Transport
    - OUT transaction (->)
- Data Transport
    - IN transaction (<-)
- Status Transport
    - IN transaction (<-)



4.1 OUT transaction (Command Transport)

The OUT transaction has the following three packets:

- OUT packet (->)
- DATA0 packet (->)
- NYET packet (<-)



OUT transaction (Command Transport)

4.1.1 OUT packet

An OUT packet consists of:

	L M	L M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L M	L M	L M	L M	L M
	1000	0111	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L

11100 0001 000 0011 1110 0001

Regrouping in groups of 4-bits:

M L

1110 0000 1000 0011 1110 0001

Hex values (as a byte):

M L

E0 83 E1

Summary (OUT packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11100001	10000011	E1	83
02	11100000		E0	

4.1.2 DATA0 packet

A DATA0 packet for Command Transport consists of 31 bytes of Command Block Wrapper data. It consists of:

	L PID	!PID M	L Data	CRC M
(bits)	4	4	248	16

Values:

	L 1100	0011 M	L Data	CRC M
(bits)	4	4	248	16

So, the data for our capture was:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCBWSignature (0x43425355)							
7-4	0x00000003							
11-8	dCBWDataTransferLength 8 bytes, (0x00000008)							
12	Device to host (0x80)							
13	Reserved (0x0)				bCBWLUN (0x0)			
14	Reserved (0x0)			bCBWBLength 10 bytes (0xA)				
30-15	CBWCB							

The command block CBWCB has the READ CAPACITY command details.

============================ *BEGIN* ================================
READ CAPACITY

Table 34: READ CAPACITY command

Byte Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (0x25)							
1	Reserved							Obsolete
2	(MSB) LOGICAL BLOCK ADDRESS							
5								(LSB)
6	Reserved							
7								
8	Reserved							PMI
9	CONTROL							

OPERATION CODE:
Value is 0x25.

LOGICAL BLOCK ADDRESS:
It specifies the first logical block accessed by this command.

PMI:
Partial Medium Indicator set to one indicates that device returns information on last logical block.

Reference: SCSI Block Commands -2 (SBC-2), Page 54
============================ *END* ================================

So, the data for our READ CAPACITY is as follows:

Byte Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (0x25)							
1	Reserved							Obsolete
2	(MSB) LOGICAL BLOCK ADDRESS (0x00000000)							
5								(LSB)
6	Reserved							
7								
8	Reserved							PMI (0)
9	CONTROL							

The CRC observed in the sample capture is:

M	L
00000101 00000101	

(bits) 16

CRC (in Hex):

M	L
05 05	

(bits) 16

Putting it together (PID + 31 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	11000011	01010101	01010011	01000010	C3	55	53	42
04	01000011	00000011	00000000	00000000	43	03	00	00
08	00000000	00001000	00000000	00000000	00	08	00	00
12	00000000	10000000	00000000	00001010	00	80	00	0A
16	00100101	00000000	00000000	00000000	25	00	00	00
20	00000000	00000000	00000000	00000000	00	00	00	00
24	00000000	00000000	00000000	00000000	00	00	00	00
28	00000000	00000000	00000000	00000000	00	00	00	00
32	00000101	00000101			05	05		

4.1.3 NYET packet

A NYET packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0110	1001
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1001 0110

Hex values:

M L
9 6

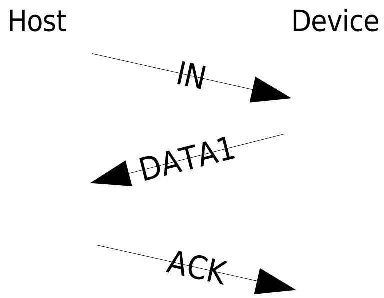
Summary (NYET packet):

Offset	Binary (M...L) Offset 0	Hexadecimal (M...L) Offset 0
00	10010110	96

4.2 IN transaction (Data Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA1 packet (<-)
- ACK packet (->)



4.2.1 IN packet

A IN packet consists of:

	L ... M	L ... M	L ... M	L ... M	L ... M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L ... M	L ... M	L ... M	L ... M	L ... M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L

11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L

1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L

E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

4.2.2 DATA1 packet

A DATA1 packet for READ CAPACITY command provides 8 bytes of data. It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	64	16

Values:

	L	M	L	M
	1101	0010	Data	CRC
(bits)	4	4	64	16

============================ *BEGIN* ===============================
READ CAPACITY parameter data

Table 35: READ CAPACITY parameter data

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)			RETURNED LOGICAL BLOCK ADDRESS				
3								(LSB)
4	(MSB)			BLOCK LENGTH IN BYTES				
7								(LSB)

Reference: SCSI Block Commands -2 (SBC-2), Page 54
============================= *END* ===============================

So, the observed value in our capture is:

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB) Last LOGICAL BLOCK ADDRESS, 501 758 (0x0007A7FE)							
3								(LSB)
4	(MSB) BLOCK LENGTH, 512 BYTES (0x00000200)							
7								(LSB)

Note: The (MSB) and (LSB) are how you write the values inside the box. In memory, everything is in little-endian:

Byte	Byte
0	0x00
1	0x07
2	0xA7
3	0xFE
4	0x00
5	0x00
6	0x02
7	0x00

The CRC observed in the sample capture is:

M	L
10010111 11111000	

(bits) 16

CRC (in Hex):

M	L
97 F8	

(bits) 16

Putting it together (PID + 8 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	01001011	00000000	00000111	10100111	4B	00	07	A7
04	11111110	00000000	00000000	00000010	FE	00	00	02
08	00000000	11111000	10010111		00	F8	97	

4.2.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

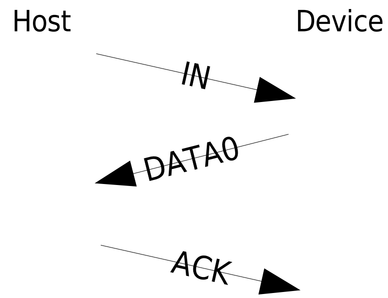
Summary (ACK packet):

Offset	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
	0	0
00	11010010	D2

4.3 IN transaction (Status Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA0 packet (<-)
- ACK packet (->)



4.3.1 IN packet

A IN packet consists of:

	L PID	!PID M	L Function Address M	L Endpoint M	L CRC M
(bits)	4	4	7	4	5

Values:

	L 1001	0110 M	L 1100 000 M	L 1000 M	L 00111 M
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L

11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L

1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L

E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

4.3.2 DATA0 packet

A DATA0 packet for READ CAPACITY command (Status Transport) provides the 13 bytes of Command Status Wrapper (CSW). It consists of:

	L PID	!PID M	L Data	CRC M
(bits)	4	4	104	16

Values:

	L 1100	0011 M	L Data	CRC M
(bits)	4	4	104	16

So, the 13 bytes of data in our example is as follows:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCSWSignature (0x53425355)							
7-4	dCSWTag (0x00000003)							
11-8	dCSWDataResidue (0x00000000)							
12	bCSWStatus, Command Passed (0x00)							

The CRC observed in the sample capture is:

	M 01000010 01001101 L
(bits)	16

CRC (in Hex):

	M 42 4D L
(bits)	16

Putting it together (PID + 13 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	11000011	01010101	01010011	01000010	C3	55	53	42
04	01010011	00000011	00000000	00000000	53	03	00	00
08	00000000	00000000	00000000	00000000	00	00	00	00
12	00000000	00000000	01001101	01000010	00	00	4D	42

4.3.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

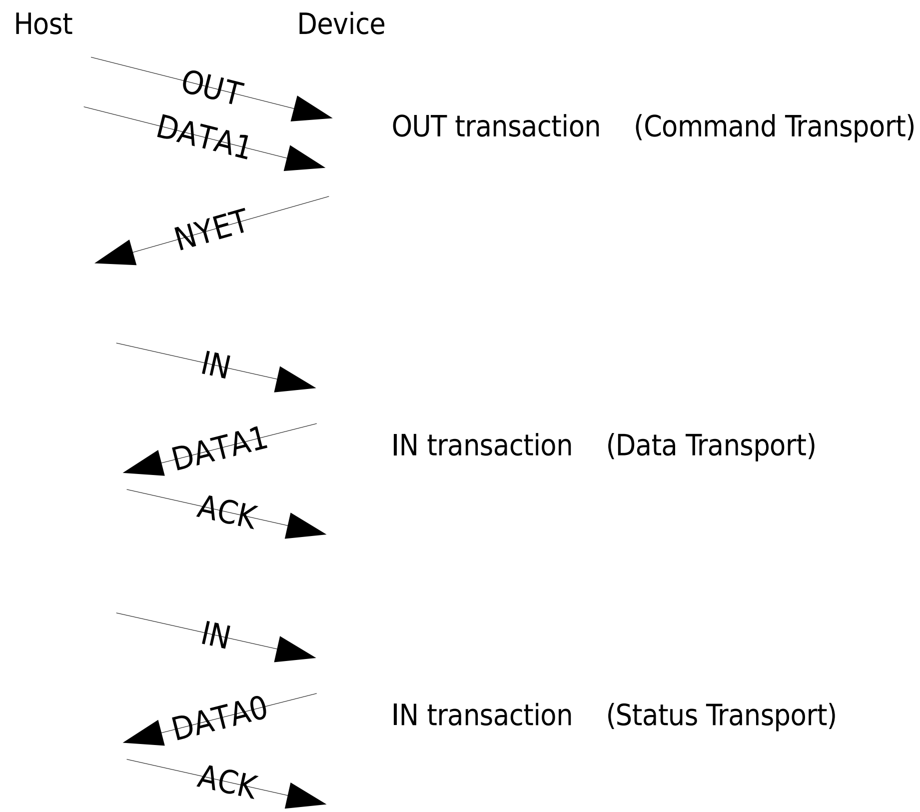
Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

5. MODE SENSE

MODE SENSE command consists of the following transport phases and transactions:

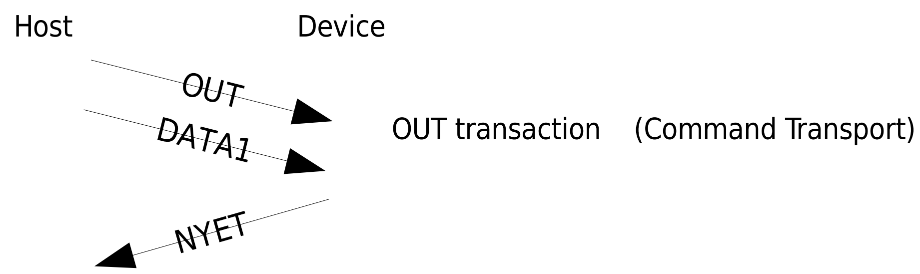
- Command Transport
    - OUT transaction (->)
- Data Transport
    - IN transaction (<-)
- Status Transport
    - IN transaction (<-)



5.1 OUT transaction (Command Transport)

The OUT transaction has the following three packets:

- OUT packet (->)
- DATA1 packet (->)
- NYET packet (<-)



5.1.1 OUT packet

An OUT packet consists of:

	L	M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L	M	L M	L M	L M
	1000	0111	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
11100 0001 000 0011 1110 0001

Regrouping in groups of 4-bits:

M L
1110 0000 1000 0011 1110 0001

Hex values (as a byte):

M L
E0 83 E1

Summary (OUT packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11100001	10000011	E1	83
02	11100000		E0	

5.1.2 DATA1 packet

A DATA1 packet for Command Transport consists of 31 bytes of Command Block Wrapper data. It consists of:

	L	M L		M
	PID	!PID	Data	CRC
(bits)	4	4	248	16

Values:

	L	M L		M
	1101	0010	Data	CRC
(bits)	4	4	248	16

So, the data for our capture was:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCBWSignature (0x43425355)							
7-4	0x00000004							
11-8	dCBWDataTransferLength 192 bytes, (0x000000C0)							
12	Device to host (0x80)							
13	Reserved (0x0)				bCBWLUN (0x0)			
14	Reserved (0x0)			bCBWBLength 6 bytes (0x6)				
30-15	CBWCB							

The command block CBWCB has the MODE SENSE command details.

============================ *BEGIN* ================================
MODE SENSE

Table 62: MODE SENSE command

Byte Bit	7	6	5	4	3	2	1	0
0	Operation Code (0x1A)							
1	Reserved				DBD	Reserved		
2	PC		PAGE CODE					
3	Reserved							
4	ALLOCATION LENGTH							
5	CONTROL							

OPERATION CODE:
Value is 0x1A.

DISABLE BLOCK DESCRIPTORS:
0 = Device may return zero or more block descriptors in MODE SENSE data.
1 = Device will not return any block descriptors.

PAGE CONTROL:
00 = Curent values.
01 = Changeable values.
10 = Default values.
11 = Saved values.

Defines the type of mode parameter values to be returned in the mode pages.

PAGE CODE:
00 = Vendor specific (does not require page format).
01-1F = Specific device-types.
20-3E = Vendor specific (page format required).
3F = Return all mode pages.

Reference: SCSI Primary Commands -2 (SPC-2), Page 100
============================ *END* ================================

So, the data for our MODE SENSE is as follows:

Byte Bit	7	6	5	4	3	2	1	0
0	Operation Code (0x1A)							
1	Reserved				DBD (0)	Reserved		
2	PC (00)		PAGE CODE (0x3F)					
3	Reserved							
4	ALLOCATION LENGTH (0xC0)							
5	CONTROL (0x00)							

The CRC observed in the sample capture is:

	M	L
	00100111 01101001	
(bits)	16	

CRC (in Hex):

	M	L
	27 69	
(bits)	16	

Putting it together (PID + 31 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	01001011	01010101	01010011	01000010	4B	55	53	42
04	01000011	00000100	00000000	00000000	43	04	00	00
08	00000000	00000000	00000000	00000000	00	00	00	00
12	00000000	10000000	00000000	00000110	00	80	00	06
16	00011010	00000000	00111111	00000000	1A	00	3F	00
20	11000000	00000000	00000000	00000000	C0	00	00	00
24	00000000	00000000	00000000	00000000	00	00	00	00
28	00000000	00000000	00000000	00000000	00	00	00	00
32	01101001	00100111			69	27		

5.1.3 NYET packet

A NYET packet consists of:

L	M
PID	!PID
4	4

(bits)

Values:

L	M
0110	1001
4	4

(bits)

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1001 0110

Hex values:

M L
9 6

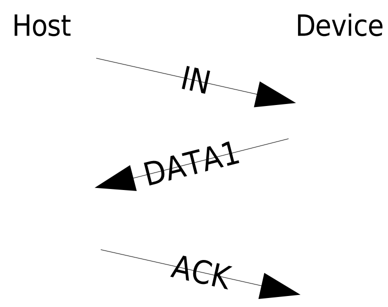
Summary (NYET packet):

Offset	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
	0	0
00	10010110	96

5.2 IN transaction (Data Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA1 packet (<-)
- ACK packet (->)



5.2.1 IN packet

A IN packet consists of:

	L M	L M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L M	L M	L M	L M	L M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L
1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L
E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

5.2.2 DATA1 packet

A DATA1 packet for MODE SENSE command provides 4 bytes of data. It consists of:

	L … M	L … M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	32	16

Values:

	L … M	L … M	L	M
	1101	0010	Data	CRC
(bits)	4	4	32	16

============================ *BEGIN* ================================

MODE SENSE parameter header

Table 147: MODE SENSE parameter header

Bit Byte	7	6	5	4	3	2	1	0
0	MODE DATA LENGTH							
1	MEDIUM TYPE							
2	DEVICE-SPECIFIC PARAMETER							
3	BLOCK DESCRIPTOR LENGTH							

MODE DATA LENGTH:
The length in bytes of the following data.

MEDIUM TYPE:
Unique to each device type.

DEVICE-SPECIFIC PARAMETER:
Unique to each device type.

BLOCK DESCRIPTOR LENGTH:
Specifies the length in bytes of all the block descriptors.

Reference: SCSI Primary Commands -2 (SPC-2), Page 189

============================ *END* ================================

So, the observed value in our capture is:

Bit Byte	7	6	5	4	3	2	1	0
0	MODE DATA LENGTH (0x03)							
1	MEDIUM TYPE, (default, 0x00)							
2	DEVICE-SPECIFIC PARAMETER (0x00)							
3	BLOCK DESCRIPTOR LENGTH (0x00)							

The CRC observed in the sample capture is:

M	L
10011111 11111000	

(bits) 16

CRC (in Hex):

M	L
9F FF	

(bits) 16

Putting it together (PID + 4 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	01001011	00000011	00000000	00000000	4B	03	00	00
04	00000000	11111111	10011111		00	FF	9F	

5.2.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M	L
1101	0010

Hex values:

M	L
D	2

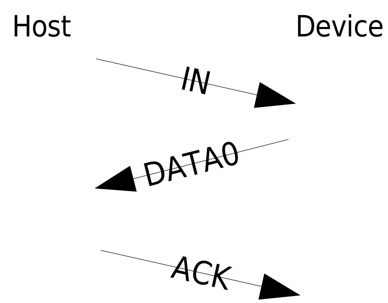
Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

5.3 IN transaction (Status Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA0 packet (<-)
- ACK packet (->)



5.3.1 IN packet

A IN packet consists of:

	L M	L M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L M	L M	L M	L M	L M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L
1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L
E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

5.3.2 DATA0 packet

A DATA0 packet for MODE SENSE command (Status Transport) provides the 13 bytes of Command Status Wrapper (CSW). It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	104	16

Values:

	L	M	L	M
	1100	0011	Data	CRC
(bits)	4	4	104	16

So, the 13 bytes of data in our example is as follows:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCSWSignature (0x53425355)							
7-4	dCSWTag (0x00000004)							
11-8	dCSWDataResidue (0x000000BC)							
12	bCSWStatus, Command Passed (0x00)							

The CRC observed in the sample capture is:

	M L
	10101001 00111010
(bits)	16

CRC (in Hex):

	M L
	A9 3A
(bits)	16

Putting it together (PID + 13 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	11000011	01010101	01010011	01000010	C3	55	53	42
04	01010011	00000100	00000000	00000000	53	04	00	00
08	00000000	10111100	00000000	00000000	00	BC	00	00
12	00000000	00000000	00111010	10101001	00	00	3A	A9

5.3.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

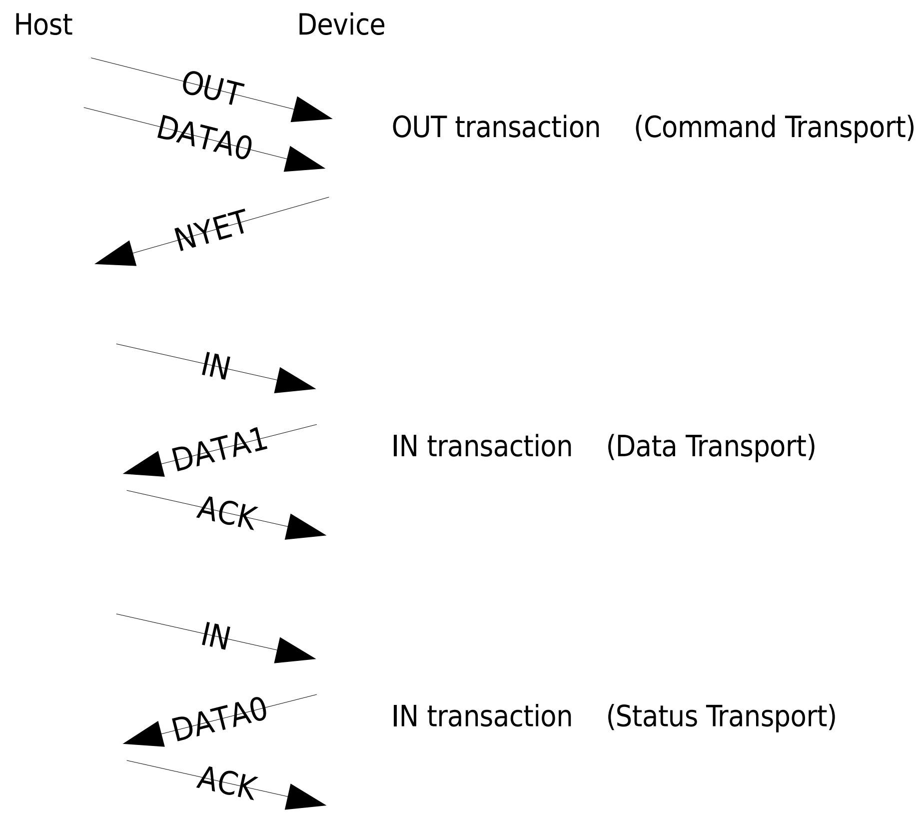
Summary (ACK packet):

<table>
<tr><td rowspan="3">Offset</td><td>Binary (M...L)</td><td>Hexadecimal (M...L)</td></tr>
<tr><td>Offset</td><td>Offset</td></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td>00</td><td>11010010</td><td>D2</td></tr>
</table>

6. REQUEST SENSE

REQUEST SENSE command consists of the following transport phases and transactions:

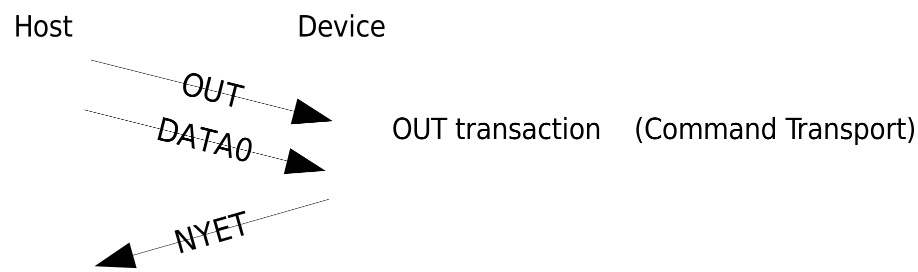
- Command Transport
    - OUT transaction (->)
- Data Transport
    - IN transaction (<-)
- Status Transport
    - IN transaction (<-)



6.1 OUT transaction (Command Transport)

The OUT transaction has the following three packets:

- OUT packet (->)
- DATA0 packet (->)
- NYET packet (<-)



6.1.1 OUT packet

An OUT packet consists of:

	L	M	L … M	L … M	L … M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L	M	L … M	L … M	L … M
	1000	0111	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M … L
11100 0001 000 0011 1110 0001

Regrouping in groups of 4-bits:

M … L
1110 0000 1000 0011 1110 0001

Hex values (as a byte):

M … L
E0 83 E1

Summary (OUT packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11100001	10000011	E1	83
02	11100000		E0	

6.1.2 DATA0 packet

A DATA0 packet for Command Transport consists of 31 bytes of Command Block Wrapper data. It consists of:

	L PID	!PID M	L Data	CRC M
(bits)	4	4	248	16

Values:

	L 1100	0011 M	L Data	CRC M
(bits)	4	4	248	16

So, the data for our capture was:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCBWSignature (0x43425355)							
7-4	0x00000007							
11-8	dCBWDataTransferLength 18 bytes, (0x00000012)							
12	Device to host (0x80)							
13	Reserved (0x0)				bCBWLUN (0x0)			
14	Reserved (0x0)			bCBWBLength 6 bytes (0x6)				
30-15	CBWCB							

The command block CBWCB has the REQUEST SENSE command details.

============================ *BEGIN* ==============================
REQUEST SENSE

Table 101: REQUEST SENSE command

Byte Bit	7	6	5	4	3	2	1	0
0	Operation Code (0x03)							
1	Reserved							
2	Reserved							
3	Reserved							
4	ALLOCATION LENGTH							
5	CONTROL							

OPERATION CODE:
Value is 0x03.

Reference: SCSI Primary Commands -2 (SPC-2), Page 135
============================ *END* ================================

So, the data for our REQUEST SENSE is as follows:

Byte Bit	7	6	5	4	3	2	1	0
0	Operation Code (0x03)							
1	Reserved (0x00)							
2	Reserved (0x00)							
3	Reserved (0x00)							
4	ALLOCATION LENGTH (0x12)							
5	CONTROL							

The CRC observed in the sample capture is:

	M	L
	00011100 10101111	
(bits)	16	

CRC (in Hex):

	M	L
	1C AF	
(bits)	16	

Putting it together (PID + 31 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	11000011	01010101	01010011	01000010	C3	55	53	42
04	01000011	00000011	00000000	00000000	43	07	00	00
08	00000000	00010010	00000000	00000000	00	12	00	00
12	00000000	10000000	00000000	00000110	00	80	00	06
16	00000011	00000000	00000000	00000000	03	00	00	00
20	00010010	00000000	00000000	00000000	12	00	00	00
24	00000000	00000000	00000000	00000000	00	00	00	00
28	00000000	00000000	00000000	00000000	00	00	00	00
32	10101111	00011100			AF	1C		

6.1.3 NYET packet

A NYET packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0110	1001
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1001 0110

Hex values:

M L
9 6

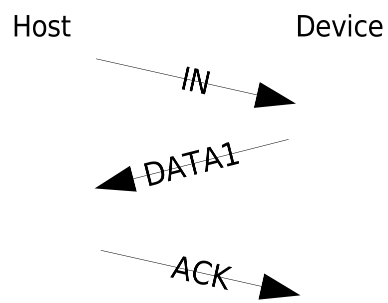
Summary (NYET packet):

Offset	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
	0	0
00	10010110	96

6.2 IN transaction (Data Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA1 packet (<-)
- ACK packet (->)



6.2.1 IN packet

A IN packet consists of:

	L PID	!PID M	L Function Address M	L Endpoint M	L CRC M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L	M	L M	L M	L M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L

11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L

1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L

E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

6.2.2 DATA1 packet

A DATA1 packet for REQUEST SENSE command provides the 18 bytes of data. It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	144	16

Values:

	L	M	L	M
	1101	0010	Data	CRC
(bits)	4	4	144	16

============================= *BEGIN* ===============================
SENSE data format

Table 102: SENSE data format

Bit Byte	7	6	5	4	3	2	1	0
0	VALID	RESPONSE CODE (0x70 or 0x71)						
1	Obsolete							
2	FILEMARK	EOM	ILI	Reserved	SENSE KEY			
3	(MSB) INFORMATION							
6								(LSB)
7	ADDITIONAL SENSE LENGTH (n-7)							
8	(MSB) COMMAND-SPECIFIC INFORMATION							
11								(LSB)
12	ADDITIONAL SENSE CODE							
13	ADDITIONAL SENSE CODE QUALIFIER							
14	FIELD REPLACEABLE UNIT CODE							
15	SKSV	SENSE-KEY SPECIFIC						
17	SENSE-KEY SPECIFIC							
18	ADDITIONAL SENSE BYTES							
n								

VALID:

0 = INFORMATION field is not as defined in this standard
1 = INFORMATION field defined as in this standard

RESPONSE CODE:

Value is 0x70 or 0x71.

FILEMARK:
Mandatory for sequential-access devices, reserved for others.
1 = Current command has read a filemark or setmark.

END OF MEDIUM (EOM):
Mandatory for sequential-access devices, reserved for others.
1 = End-of-medium condition.

INCORRECT LENGTH INDICATOR (ILI):
Indicates that request logical block length did not match the logical block length of data on the medium.

SENSE KEY, ADDITIONAL SENSE CODE, ADDITIONAL SENSE CODE QUALIFIER:
Provide hierarchy of information relating to error and exception conditions.
Table 107 – Sense key descriptions. Page 141. SCSI Primary Commands -2 (SPC-2).

INFORMATION:
Device-type or command-specific.

ADDITIONAL SENSE LENGTH:
Additional sense bytes that follow.

COMMAND-SPECIFIC INFORMATION:
Command-specific.

FIELD REPLACEABLE UNIT CODE:
Defines a device-specific mechanism or unit that has failed.

SENSE KEY SPECIFIC VALID (SKSV):
1 = SENSE KEY SPECIFIC field contains valid informatino as per this standard.

SENSE-KEY SPECIFIC:
SENSE-KEY specific.

Reference: SCSI Primary Commands -2 (SPC-2), Page 136
============================ *END* ================================

So, the observed value in our capture is:

Bit Byte	7	6	5	4	3	2	1	0
0	VALID	RESPONSE CODE (0x70)						
1	Obsolete							
2	FILEMARK	EOM	ILI	Reserved	SENSE KEY (0x5)			
3	(MSB) INFORMATION (0x00000000)							
6								(LSB)
7	ADDITIONAL SENSE LENGTH (0x0A)							
8	(MSB) COMMAND-SPECIFIC INFORMATION (0x00000000)							
11								(LSB)
12	ADDITIONAL SENSE CODE (0x24)							
13	ADDITIONAL SENSE CODE QUALIFIER (0x00)							
14	FIELD REPLACEABLE UNIT CODE (0x00)							
15	SKSV (0)	SENSE-KEY SPECIFIC (0x00)						
17	SENSE-KEY SPECIFIC (0x00)							
18	ADDITIONAL SENSE BYTES (0x00)							

SENSE KEY 0x5 represents ILLEGAL REQUEST. This REQUEST SENSE was called after a MEDIUM REMOVAL transfer failed.

ASC 0x24 and ASCQ 0x00 correspond to INVALID FIELD IN CDB (Table 108 – ASC and ASCQ assignments, part 6 of 13, SCSI Primary Commands – 2, SPC-2, page 148).

The CRC observed in the sample capture is:

	M ... L
	10101111 01110000
(bits)	16

CRC (in Hex):

	M ... L
	AF 70
(bits)	16

Putting it together (PID + 18 bytes of data + CRC):

	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
Offset	0	1	2	3	0	1	2	3
00	01001011	01110000	00000000	00000101	4B	70	00	05
04	00000000	00000000	00000000	00000000	00	00	00	00
08	00000000	00000000	00000000	00000000	0A	00	00	00
12	00000000	00100100	00000000	00000000	00	24	00	00
16	00000000	00000000	00000000	01110000	00	00	00	70
20	10101111				AF			

6.2.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

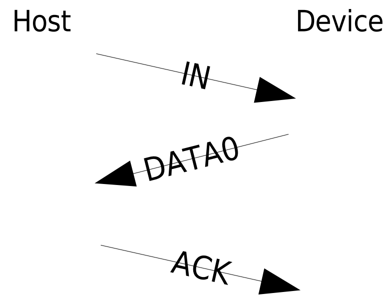
Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

6.3 IN transaction (Status Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA0 packet (<-)
- ACK packet (->)



6.3.1 IN packet

A IN packet consists of:

	L	M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L	M	L M	L M	L M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L
1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L
E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

6.3.2 DATA0 packet

A DATA0 packet for REQUEST SENSE command (Status Transport) provides the 13 bytes of Command Status Wrapper (CSW). It consists of:

	L PID	!PID M	L Data	CRC M
(bits)	4	4	104	16

Values:

	L 1100	0011 M	L Data	CRC M
(bits)	4	4	104	16

So, the 13 bytes of data in our example is as follows:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCSWSignature (0x53425355)							
7-4	dCSWTag (0x00000007)							
11-8	dCSWDataResidue (0x00000000)							
12	bCSWStatus, Command Passed (0x00)							

The CRC observed in the sample capture is:

	M 01111111 10000010 L
(bits)	16

CRC (in Hex):

	M 7F 82 L
(bits)	16

Putting it together (PID + 13 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	11000011	01010101	01010011	01000010	C3	55	53	42
04	01010011	00000111	00000000	00000000	53	07	00	00
08	00000000	00000000	00000000	00000000	00	00	00	00
12	00000000	00000000	01111111	10000010	00	00	7F	82

6.3.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

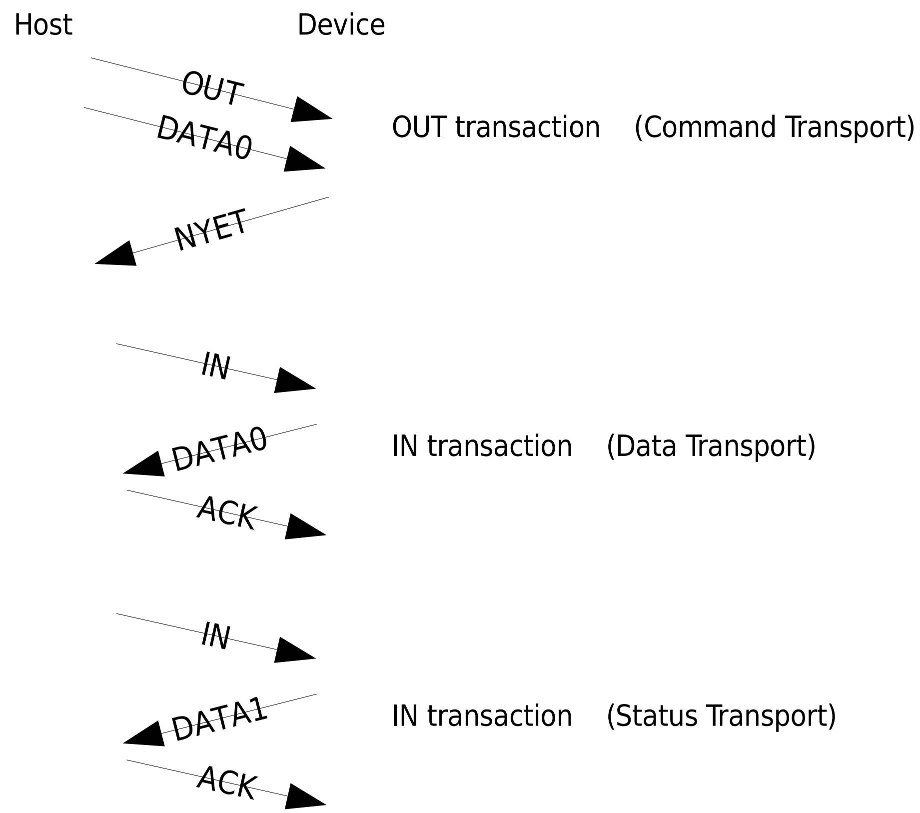
Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

7. READ

READ command consists of the following transport phases and transactions:

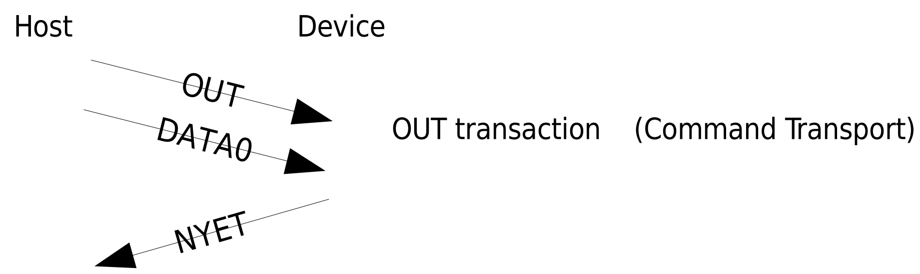
- Command Transport
    - OUT transaction (->)
- Data Transport
    - IN transaction (<-)
- Status Transport
    - IN transaction (<-)



7.1 OUT transaction (Command Transport)

The OUT transaction has the following three packets:

- OUT packet (->)
- DATA0 packet (->)
- NYET packet (<-)



7.1.1 OUT packet

An OUT packet consists of:

	L ... M	L ... M	L ... M	L ... M	L ... M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L ... M	L ... M	L ... M	L ... M	L ... M
	1000	0111	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M ... L
11100 0001 000 0011 1110 0001

Regrouping in groups of 4-bits:

M ... L
1110 0000 1000 0011 1110 0001

Hex values (as a byte):

M ... L
E0 83 E1

Summary (OUT packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11100001	10000011	E1	83
02	11100000		E0	

7.1.2 DATA0 packet

A DATA0 packet for Command Transport consists of 31 bytes of Command Block Wrapper data. It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	248	16

Values:

	L	M	L	M
	1100	0011	Data	CRC
(bits)	4	4	248	16

So, the data for our capture was:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCBWSignature (0x43425355)							
7-4	0x00000045							
11-8	dCBWDataTransferLength 512 bytes, (0x00000200)							
12	Device to host (0x80)							
13	Reserved (0x0)				bCBWLUN (0x0)			
14	Reserved (0x0)			bCBWBLength 10 bytes (0xA)				
30-15	CBWCB							

The command block CBWCB has the READ command details.

============================ *BEGIN* ==============================
READ
Table 28: READ(10) command

Byte Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (0x28)							
1	RDPROTECT			DPO	FUA	Reserved	FUA_NV	Obsolete
2	(MSB) LOGICAL BLOCK ADDRESS							
5								(LSB)
6	Reserved			GROUP NUMBER				
7	(MSB) TRANSFER LENGTH							
8								(LSB)
9	CONTROL							

OPERATION CODE:
Value is 0x28.

RDPROTECT:
See Table 29, page 48 (SBC-2).

DISABLE PAGE OUT (DPO):
0 = Retention priority determined by RETENTION PRIORITY fields.

FORCE UNIT ACCESS (FUA), FORCE UNIT ACCESS NON-VOLATILE CACHE (FUA_NV):
See Table 30, page 51 (SBC-2).

LOGICAL BLOCK ADDRESS:
First logical block accessed by this command.

GROUP NUMBER:
Specifies the group into which attributes associated with the command should be collected.

TRANSFER LENGTH:
Number of continguous logical blocks of data that shall be read.

Reference: SCSI Block Commands -2 (SBC-2), Page 48
============================ *END* ==============================

So, the data for our CBWB capture was:

Byte Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (0x28)							
1	RDPROTECT			DPO	FUA	Reserved	FUA_NV	Obsolete
2	(MSB) LOGICAL BLOCK ADDRESS, 567th, (0x00000237)							
5								(LSB)
6	Reserved			GROUP NUMBER (0x00)				
7	(MSB) TRANSFER LENGTH (0x0001)							
8								(LSB)
9	CONTROL (0x00)							

There are six more bytes to filled with 0x00 to fill the size of 30-15 bytes of the CBW.

The CRC observed in the sample capture is:

M	L
01100001 00100111	

(bits) 16

CRC (in Hex):

M	L
61 27	

(bits) 16

Putting it together (PID + 31 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	11000011	01010101	01010011	01000010	C3	55	53	42
04	01000011	01000101	00000000	00000000	43	45	00	00
08	00000000	00000000	00000010	00000000	00	00	02	00
12	00000000	10000000	00000000	00001010	00	80	00	0A
16	00101000	00000000	00000000	00000000	28	00	00	00
20	00000010	00110111	00000000	00000000	02	37	00	00
24	00000001	00000000	00000000	00000000	01	00	00	00
28	00000000	00000000	00000000	00000000	00	00	00	00
32	00100111	01100001			27	61		

7.1.3 NYET packet

A NYET packet consists of:

	L PID	!PID M
(bits)	4	4

Values:

	L 0110	1001 M
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1001 0110

Hex values:

M L
9 6

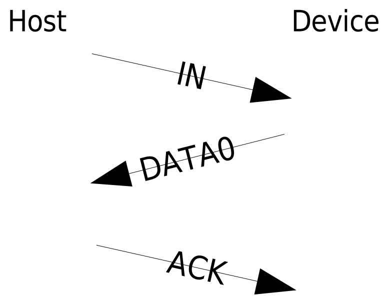
Summary (NYET packet):

Offset	Binary (M...L) Offset 0	Hexadecimal (M...L) Offset 0
00	10010110	96

7.2 IN transaction (Data Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA0 packet (<-)
- ACK packet (->)



7.2.1 IN packet

A IN packet consists of:

	L M	L M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L M	L M	L M	L M	L M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L
1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L
E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

7.2.2 DATA0 packet

A DATA0 packet for READ command provides the 512 bytes of data. It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	512*8	16

Values:

	L	M	L	M
	1100	0011	Data	CRC
(bits)	4	4	512*8	16

For this LBA 567 on the SanDisk, the 512 bytes of data contained 0x00.

The CRC observed in the sample capture is:

	M L
	01000100 10111110
(bits)	16

CRC (in Hex):

	M L
	44 BE
(bits)	16

Putting it together (PID + 512 bytes of data + CRC):

	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
Offset	0	1	2	3	0	1	2	3
00	11000011	00000000	10000000	00000010	C3	00	00	00
04	00000000	0000000	00000000	00000000	00	00	00	00
08	...	...	...	...	00	00	00	00
512	00000000	10111110	01000100		00	BE	44	

7.2.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

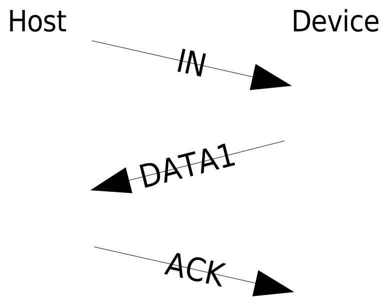
Summary (ACK packet):

<table>
<tr><td rowspan="3">Offset</td><td>Binary (M...L)</td><td>Hexadecimal (M...L)</td></tr>
<tr><td>Offset</td><td>Offset</td></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td>00</td><td>11010010</td><td>D2</td></tr>
</table>

7.3 IN transaction (Status Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA1 packet (<-)
- ACK packet (->)



7.3.1 IN packet

A IN packet consists of:

	L PID	!PID M	L Function Address M	L Endpoint M	L CRC M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L	M L	M L	M L	M L M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L

11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L

1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L

E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

7.3.2 DATA1 packet

A DATA1 packet for the READ command (Status Transport) provides the 13 bytes of Command Status Wrapper (CSW). It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	104	16

Values:

	L	M	L	M
	1101	0010	Data	CRC
(bits)	4	4	104	16

So, the 13 bytes of data in our example is as follows:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCSWSignature (0x53425355)							
7-4	dCSWTag (0x00000045)							
11-8	dCSWDataResidue (0x00000000)							
12	bCSWStatus, Command Passed (0x00)							

The CRC observed in the sample capture is:

	M L
	00100001 01010111
(bits)	16

CRC (in Hex):

	M L
	21 57
(bits)	16

Putting it together (PID + 13 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	01001011	01010101	01010011	01000010	4B	55	53	42
04	01010011	01000101	00000000	00000000	53	45	00	00
08	00000000	00000000	00000000	00000000	00	00	00	00
12	00000000	00000000	01010111	00100001	00	00	57	21

7.3.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

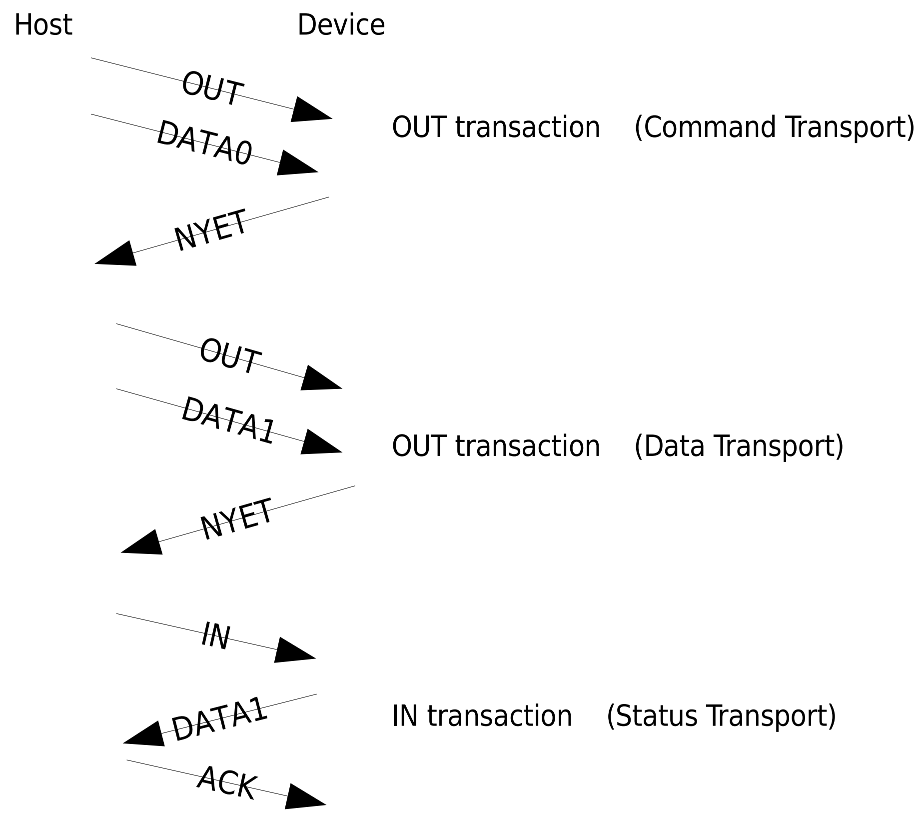
Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

8. WRITE

WRITE command consists of the following transport phases and transactions:

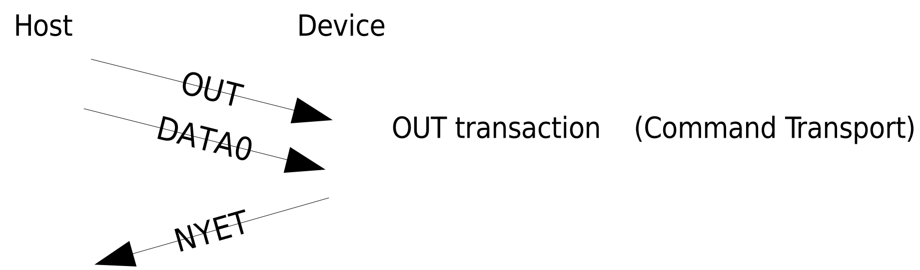
- Command Transport
    - OUT transaction (->)
- Data Transport
    - OUT transaction (<-)
- Status Transport
    - IN transaction (<-)



8.1 OUT transaction (Command Transport)

The OUT transaction has the following three packets:

- OUT packet (->)
- DATA0 packet (->)
- NYET packet (<-)



8.1.1 OUT packet

An OUT packet consists of:

	L M	L M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L M	L M	L M	L M	L M
	1000	0111	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L

11100 0001 000 0011 1110 0001

Regrouping in groups of 4-bits:

M L

1110 0000 1000 0011 1110 0001

Hex values (as a byte):

M L

E0 83 E1

Summary (OUT packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11100001	10000011	E1	83
02	11100000		E0	

8.1.2 DATA0 packet

A DATA0 packet for Command Transport consists of 31 bytes of Command Block Wrapper data. It consists of:

	L PID	!PID M	L Data	CRC M
(bits)	4	4	248	16

Values:

	L 1100	0011 M	L Data	CRC M
(bits)	4	4	248	16

So, the data for our capture was:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCBWSignature (0x43425355)							
7-4	0x00000155							
11-8	dCBWDataTransferLength 512 bytes, (0x00000200)							
12	Host to device (0x00)							
13	Reserved (0x0)				bCBWLUN (0x0)			
14	Reserved (0x0)			bCBWBLength 10 bytes (0xA)				
30-15	CBWCB							

The command block CBWCB has the WRITE command details.

============================ *BEGIN* ================================
WRITE
Table 62: WRITE (10) command

Byte Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (0x2A)							
1	WRPROTECT			DPO	FUA	Reserved	FUA_NV	Obsolete
2	(MSB) LOGICAL BLOCK ADDRESS							
5								(LSB)
6	Reserved			GROUP NUMBER				
7	(MSB) TRANSFER LENGTH							
8								(LSB)
9	CONTROL							

OPERATION CODE:
Value is 0x2A.

WRPROTECT:
See Table 63, page 79 (SBC-2).

DISABLE PAGE OUT (DPO):
0 = Retention priority determined by RETENTION PRIORITY fields.

FORCE UNIT ACCESS (FUA), FORCE UNIT ACCESS NON-VOLATILE CACHE (FUA_NV):
See Table 30, page 51 (SBC-2).

LOGICAL BLOCK ADDRESS:
First logical block accessed by this command.

GROUP NUMBER:
Specifies the group into which attributes associated with the command should be collected.

TRANSFER LENGTH:
Number of continguous logical blocks of data that shall be read.

Reference: SCSI Block Commands -2 (SBC-2), Page 79
============================ *END* ================================

So, the data for our CBWB capture was:

Byte Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (0x2A)							
1	RDPROTECT			DPO	FUA	Reserved	FUA_NV	Obsolete
2	(MSB) LOGICAL BLOCK ADDRESS, 567th, (0x00000043)							
5	(LSB)							
6	Reserved			GROUP NUMBER (0x00)				
7	(MSB) TRANSFER LENGTH (0x0001)							
8	(LSB)							
9	CONTROL (0x00)							

There are six more bytes to filled with 0x00 to fill the size of 30-15 bytes of the CBW.

The CRC observed in the sample capture is:

M	L
10111000 11001111	

(bits) 16

CRC (in Hex):

M	L
B8 CF	

(bits) 16

Putting it together (PID + 31 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	11000011	01010101	01010011	01000010	C3	55	53	42
04	01000011	01010101	00000001	00000000	43	55	01	00
08	00000000	00000000	00000010	00000000	00	00	02	00
12	00000000	00000000	00000000	00001010	00	00	00	0A
16	00101010	00000000	00000000	00000000	2A	00	00	00
20	00000000	01000011	00000000	00000000	00	43	00	00
24	00000001	00000000	00000000	00000000	01	00	00	00
28	00000000	00000000	00000000	00000000	00	00	00	00
32	11001111	10111000			CF	B8		

8.1.3 NYET packet

A NYET packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0110	1001
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1001 0110

Hex values:

M L
9 6

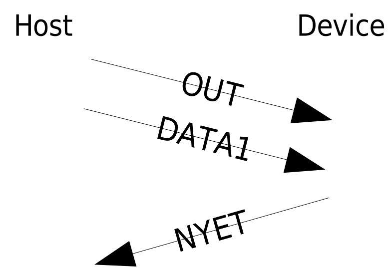
Summary (NYET packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	10010110	96

8.2 OUT transaction (Data Transport)

The OUT transaction has the following three packets:

- OUT packet (->)
- DATA1 packet (->)
- NYET packet (<-)



OUT transaction (Data Transport)

8.2.1 OUT packet

An OUT packet consists of:

	L M	L M	L M	L M
PID	!PID	Function Address	Endpoint	CRC
4	4	7	4	5

(bits)

Values:

	L M	L M	L M	L M
1000	0111	1100 000	1000	00111
4	4	7	4	5

(bits)

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L

11100 0001 000 0011 1110 0001

Regrouping in groups of 4-bits:

M L

1110 0000 1000 0011 1110 0001

Hex values (as a byte):

M L

E0 83 E1

Summary (OUT packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	11100001	10000011	E1	83
02	11100000		E0	

8.2.2 DATA1 packet

A DATA1 packet for Data Transport consists of 512 bytes of data. It consists of:

	L	M	L	M
	PID	!PID	Data	CRC
(bits)	4	4	512*8	16

Values:

	L	M	L	M
	1101	0010	Data	CRC
(bits)	4	4	512*8	16

For this LBA 67 WRITE on the SanDisk, 512 bytes of data were written.

The CRC observed in the sample capture is:

	M … L
	01100100 01000001
(bits)	16

CRC (in Hex):

	M … L
	64 41
(bits)	16

Putting it together (PID + 512 bytes of data + CRC):

	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
Offset	0	1	2	3	0	1	2	3
00	11000011	...	...	...	4B	...	...	...
04	...	...	...	...	...	...	...	...
08	...	...	...	...	...	...	...	...
512	...	01000001	01100100		...	41	64	

8.2.3 NYET packet

A NYET packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0110	1001
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1001 0110

Hex values:

M L
9 6

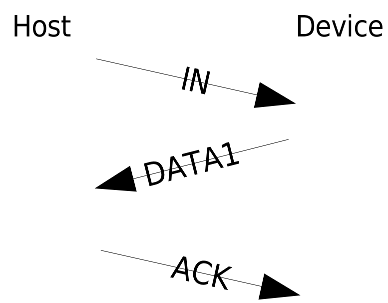
Summary (NYET packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	10010110	96

8.3 IN transaction (Status Transport)

The IN transaction has the following three packets:

- IN packet (->)
- DATA1 packet (<-)
- ACK packet (->)



8.3.1 IN packet

A IN packet consists of:

	L M	L M	L M	L M	L M
	PID	!PID	Function Address	Endpoint	CRC
(bits)	4	4	7	4	5

Values:

	L M	L M	L M	L M	L M
	1001	0110	1100 000	1000	00111
(bits)	4	4	7	4	5

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
11100 0001 000 0011 0110 1001

Regrouping in groups of 4-bits:

M L
1110 0000 1000 0011 0110 1001

Hex values (as a byte):

M L
E0 83 69

Summary (IN packet):

	Binary (M...L)		Hexadecimal (M...L)	
	Offset		Offset	
Offset	0	1	0	1
00	01101001	10000011	69	83
02	11100000		E0	

8.3.2 DATA1 packet

A DATA1 packet for the WRITE command (Status Transport) provides the 13 bytes of Command Status Wrapper (CSW). It consists of:

	L PID	!PID M	L Data	CRC M
(bits)	4	4	104	16

Values:

	L 1101	0010 M	L Data	CRC M
(bits)	4	4	104	16

So, the 13 bytes of data in our example is as follows:

Byte Bit	7	6	5	4	3	2	1	0
3-0	dCSWSignature (0x53425355)							
7-4	dCSWTag (0x00000155)							
11-8	dCSWDataResidue (0x00000000)							
12	bCSWStatus, Command Passed (0x00)							

The CRC observed in the sample capture is:

	M 00101101 01011011 L
(bits)	16

CRC (in Hex):

	M 2D 5B L
(bits)	16

Putting it together (PID + 13 bytes of data + CRC):

Offset	Binary (M...L)				Hexadecimal (M...L)			
	Offset				Offset			
	0	1	2	3	0	1	2	3
00	01001011	01010101	01010011	01000010	4B	55	53	42
04	01010011	01010101	00000001	00000000	53	55	01	00
08	00000000	00000000	00000000	00000000	00	00	00	00
12	00000000	00000000	01011110	00101101	00	00	5B	2D

8.3.3 ACK packet

An ACK packet consists of:

	L	M
	PID	!PID
(bits)	4	4

Values:

	L	M
	0100	1011
(bits)	4	4

USB is little-endian. LSB goes out of the wire, first.

Displaying from MSB to LSB:

M L
1101 0010

Hex values:

M L
D 2

Summary (ACK packet):

	Binary (M...L)	Hexadecimal (M...L)
	Offset	Offset
Offset	0	0
00	11010010	D2

References

[1] Universal Serial Bus Specification. Revision 2.0. April 27, 2000. http://www.usb.org

[2] SCSI Primary Commands - 2 (SPC-2). Project T10/1236-D.

[3] SCSI Block Commands – 2 (SBC-2). Project T10/1417-D.

[4] USB Mass Storage Class – Bulk-Only Transport.

[5] USB Mass Storage Class – Specification Overview.

GNU Free Documentation License

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document
straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections
and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add
to it an item stating at least the title, year, new authors, and
publisher of the Modified Version as given on the Title Page. If
there is no section Entitled "History" in the Document, create one
stating the title, year, authors, and publisher of the Document as
given on its Title Page, then add an item describing the Modified
Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for
public access to a Transparent copy of the Document, and likewise
the network locations given in the Document for previous versions
it was based on. These may be placed in the "History" section.
You may omit a network location for a work that was published at
least four years before the Document itself, or if the original
publisher of the version it refers to gives permission.
K. For any section Entitled "Acknowledgements" or "Dedications",
Preserve the Title of the section, and preserve in the section all
the substance and tone of each of the contributor acknowledgements
and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document,
unaltered in their text and in their titles. Section numbers
or the equivalent are not considered part of the section titles.
M. Delete any section Entitled "Endorsements". Such a section
may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements"
or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections
Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright
resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on
covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole
aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special
permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

    Copyright (c) YEAR YOUR NAME.
    Permission is granted to copy, distribute and/or modify this document
    under the terms of the GNU Free Documentation License, Version 1.2
    or any later version published by the Free Software Foundation;
    with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
    A copy of the license is included in the section entitled "GNU
    Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

    with the Invariant Sections being LIST THEIR TITLES, with the
    Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.