Developing a driver for a film scanner by means of USB sniffing and reverse engineering

Hugo Platzer

University of Salzburg

November 16, 2017

1 The USB standard

1.1 Motivation

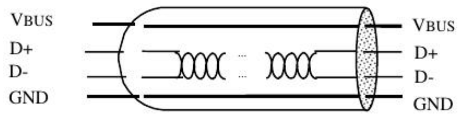
USB is an interface intended to connect various peripherals to PCs. These include: Human interface devices like keyboards and mice; storage devices like card readers, external hard disks, memory sticks and smartphones; multimedia devices like microphones, speakers, cameras and scanners. Some highlights leading to its wide adoption:

- Unified interface for all kinds of peripherals
- Plug and play: The user plugs in the device, the configuration (e.g. loading the appropriate drivers) is done automatically by the operating system
- Number of ports can be increased using hubs. Multiple hubs can be chained allowing for up to 127 devices on a single root port.
- High data rate of 480 Mbit / s (USB 2.0 high-speed mode). Also offers low-latency transfers for real-time audio/video applications
- Backwards compatibility: Older USB 1.1 devices can be used at 2.0 hosts. High-speed USB 2.0 devices can also be used on older machines supporting only USB 1.1 (albeit at lower speed).

1.2 Electrical side

A USB cable has four wires: One as ground, VBUS for a 5 V power supply and two for data transmission. The power line allows it to draw up to 100 mA without any configuration. This is useful for simple devices that are not using the data lanes, just the power, like USB lights. Also it allows for devices not taking much power to be self-powered which eliminates the need for an extra

Figure 1: USB cable cross-section



power supply and connector. Devices can ask the host for more power (up to 500 mA), those that need even more (like a lot of scanners) need an external supply.

1.3 Signaling

1.3.1 Low-level states

USB is a serial bus which means there is only a single path for data transmission. Differential signalling across the D- and D+ wires is used, which means the difference in voltage across the two wires (rather than some absolute) determines the state. This is beneficial because noise during transmission should affect both lines equally, not changing the difference. Higher frequencies and thus data rates become possible.

Table 1: USB speed modes

Mode	Speed	Bit time
Low Speed	1.5 Mbit / s	667 ns
Full Speed	12 Mbit / s	83 ns
High Speed	480 Mbit / s	2 ns

Table 2: Low-level data line states (only applies to Full Speed)

Levels	State
Differential '0'	D- high, D+ low
Differential '1'	D- low, D+ high
Single Ended Zero (SE0)	both low
Single Ended One (SE1)	both high (illegal state, should never happen)
Data 'J' state	Differential '1'
Data 'K' state	Differential '0'
Idle state	Data 'J' state
Start of Packet (SOP)	Switch from idle to 'K'
End of Packet (EOP)	SE0 for 2 bit times followed by 'J' for 1 bit time
Disconnect	SE0 for $\geq$ 2 us
Connect	Idle for 2.5 us
Reset	SE0 for $\geq$ 2.5 us

Low Speed is used for devices where speed is not important (mice, keyboards). It allows for cheaper cables and electronics. High Speed is only available in USB 2.0. For reasons of simplicity, only full speed signaling will be covered here.

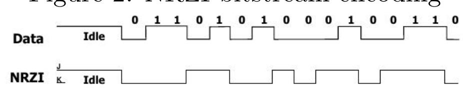
SE0 is the state of the data lines if no device is connected. The host recognizes a device being plugged in by the D+ line being "pulled up" to high. It then most likely initiates a reset so the device is in a known state for communication to begin. Similarly, a disconnect is sensed by a SE0 for some time.

It is important to note that **all communication on the USB bus is initiated by the host**. Devices on the bus can not directly talk to each other and can only talk to the host as a direct response to a request made by it before.

1.3.2 Bitstream encoding

USB uses NRZI encoding for the transmitted data: A zero is represented by a change to the opposite state while a one is represented by staying in the same state.

Figure 2: NRZI bitstream encoding



For keeping the receiver clock in sync with the data it is not ideal if the signal stays at J or K for too long. To prevent this, a technique called "bit stuffing" is used: Before doing NRZI encoding, a zero is inserted after every six consecutive ones in the data. The receiver recognizes the stuffed bits during decoding and discards them.

1.4 Packets

Packets are the atomic unit of data transmission in USB. In between packet transmission, the bus remains in an idle state. Every packet starts with a sync pattern to synchronize the clocks between sender and receiver. Next are the actual data bits. The packet is terminated by an EOP state. Fields in a packet are transmitted least-significant bit first.

The first 8 bits of every packet contain the packet ID (PID) which identifies its type and thus how the rest of the packet data should be interpreted. The PID is 4 bits long, they are transmitted a second time in reverse bit order to allow the receiver to quickly discard a faulty packet. There are 17 different packet types (PRE and ERR have the same ID, some are only relevant for High Speed):

Table 3: USB packet types; notice how the least-significant two bits identify the packet category

PID type	PID name	PID bits ($< 3 : 0 >$)	Description
Token	OUT	0001	Address + endpoint number for host-to-device transaction
	IN	1001	Address + endpoint number for device-to-host transaction
	SOF	0101	Start-of-frame marker, frame number
	SETUP	1101	Special host-to-device transaction for device configuration
Data	DATA0	0011	Data packet
	DATA1	1011	Data packet
	DATA2	0111	Data packet (only High Speed)
	MDATA	1111	Data packet (only High Speed)
Handshake	ACK	0010	Receiver accepts error-free data packet
	NAK	1010	Receiver cannot accept data or transmitter cannot send data
	STALL	1110	Endpoint halted or control pipe request not supported
	NYET	0110	Data packet (only High Speed)
Special	PRE	1100	Preamble to enable downstream traffic to low-speed devices
	ERR	1100	Split Transaction error handshake
	SPLIT	1000	High speed Split Transaction token (only High Speed)
	PING	0100	High speed control flow probe (only High Speed)
	Reserved	0000	Reserved PID

1.4.1 Token packet

Figure 3: Token packet composition

Field	PID	ADDR	ENDP	CRC5
Bits	8	7	4	5

Token packets are used at the start of so-called transactions to specify the target of the transaction on the bus, namely a certain device and endpoint. There are 127 possible devices on a bus (address 0 is reserved for a device that has not been configured yet).

Endpoints are logical entities on a device that are used as sources and sinks of data in so-called pipes. A pipe is either in OUT (to device) or IN (to host) direction. Endpoint 0 is a special bidirectional pipe that must be available on every device right after the reset. It is used mainly for identifying and configuring the device.

1.4.2 Data packet

Figure 4: Data packet composition

Field	PID	DATA	CRC16
Bits	8	0-8192	16

Used to transmit the actual data in a transaction.

1.4.3 Handshake packet

Figure 5: Handshake packet composition

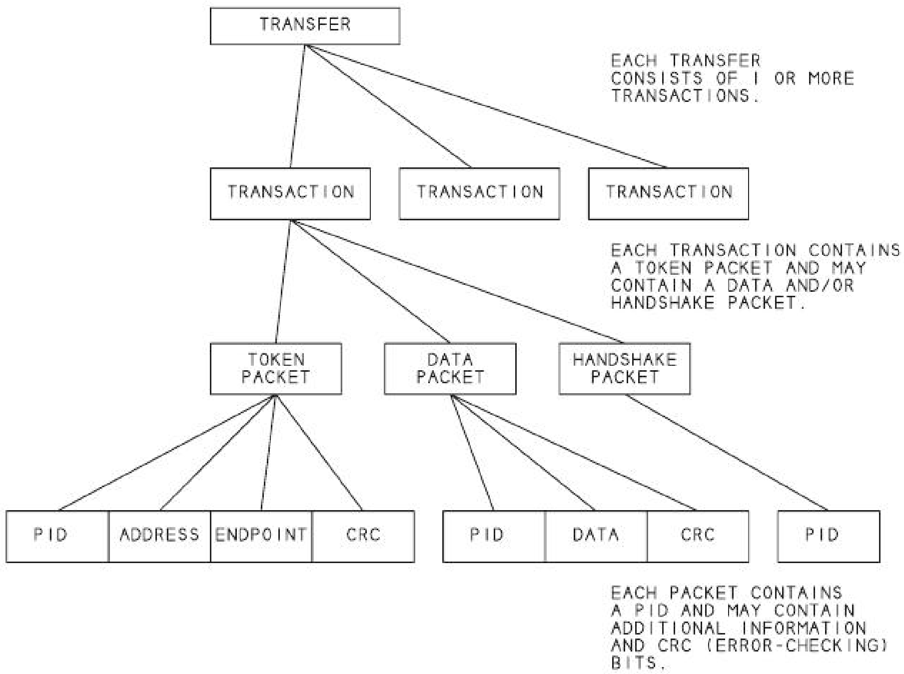
Field	PID
Bits	8

Used to report the status of a transaction.

1.5 Transactions

1.6 Transfers

Figure 6: Composition of a USB transfer



1.6.1 Control transfer

1.6.2 Interrupt transfer

1.6.3 Bulk transfer

1.6.4 Isochronous transfer

1.7 Enumeration, Configuration

References

[1] Universal Serial Bus Specification, Revision 2.0. April 27, 2000. Available at `http://www.usb.org/developers/docs/usb20_docs/`

[2] USB Complete: Everything You Need to Develop Custom USB Peripherals (3rd edition). Jan Axelson, Lakeview Research LLC 2012. ISBN: `978-1-931448-03-1`