USB in a NutShell

Making sense of the USB standard

Starting out new with USB can be quite daunting. With the USB 2.0 specification at 650 pages one could easily be put off just by the sheer size of the standard. This is only the beginning of a long list of associated standards for USB. There are USB Class Standards such as the HID Class Specification which details the common operation of devices (keyboards, mice etc) falling under the HID (Human Interface Devices) Class - only another 97 pages. If you are designing a USB Host, then you have three Host Controller Interface Standards to choose from. None of these are detailed in the USB 2.0 Spec.

The good news is you don't even need to bother reading the entire USB standard. Some chapters were churned out by marketing, others aimed at the lower link layer normally taken care off by your USB controller IC and a couple aimed at host and hub developers. Lets take a little journey through the various chapters of the USB 2.0 specification and briefly introduce the key points.

Chapter	Name	Description	Pages
1	Introduction	Includes the motivation and scope for USB. The most important piece of information in this chapter is to make reference to the Universal Serial Bus Device Class Specifications. No need reading this chapter.	2
2	Terms and Abbreviations	This chapter is self-explanatory and a necessary evil to any standard.	8
3	Background	Specifies the goals of USB which are Plug'n'Play and simplicity to the end user (*not developer*). Introduces Low, Full and High Speed ranges with a feature list straight from marketing. No need reading this chapter either.	4
4	Architectural Overview	This is where you can start reading. This chapter provides a basic overview of a USB system including topology, data rates, data flow types, basic electrical specs etc.	10
5	USB Data Flow Model	This chapter starts to talk about how data flows on a Universal Serial Bus. It introduces terms such as endpoints and pipes then spends most of the chapter on each of the data flow types (Control, Interrupt, Isochronous and Bulk). While it's important to know each transfer type and its properties it is a little heavy on for a first reader.	60
6	Mechanical	This chapter details the USB's two standard connectors. The important information here is that a type A connector is oriented facing downstream and a type B connector upstream. Therefore it should be impossible to plug a cable into two upstream ports. All detachable cables must be full/high speed, while any low speed cable must be hardwired to the appliance. Other than a quick look at the connectors, you can skip this chapter unless you intend to manufacture USB connectors and/or cables. PCB designers can find standard footprints in this chapter.	33
7	Electrical	This chapter looks at low level electrical signalling including line impedance, rise/fall times, driver/receiver specifications and bit level encoding, bit stuffing etc. The more important parts of this chapter are the device speed identification by using a resistor to bias either data line and bus powered devices vs self powered devices. Unless you are designing USB transceivers at a silicon level you can flip through this chapter. Good USB device datasheets will detail what value bus termination resistors you will need for bus impedance matching.	75
8	Protocol Layer	Now we start to get into the protocol layers. This chapter describes the USB packets at a byte level including the sync, pid, address, endpoint, CRC fields. Once this has been grasped it moves on to the next protocol layer, USB packets. Most developers still don't see these lower protocol layers as their USB device IC's take care of this. However a understanding of the status reporting and handshaking is worthwhile.	45
9	USB Device Frame Work	This is the most frequently used chapter in the entire specification and the only one I ever bothered printing and binding. This details the bus enumeration and request codes (set address, get descriptor etc) which make up the most common protocol layer USB programmers and designers will ever see. This chapter is a must read in detail.	36
10	USB Host Hardware and Software	This chapter covers issues relating to the host. This includes frame and microframe generation, host controller requirements, software mechanisms and the universal serial bus driver model. Unless you are designing Hosts, you can skip this chapter.	23
11	Hub Specification	Details the workings of USB hubs including hub configuration, split transactions, standard descriptors for hub class etc. Unless you are designing Hubs, you can skip this chapter.	143

So now we can begin to read the parts of the standard relevant to our needs. If you develop drivers (Software) for USB peripherals then you may only need to read chapters,

- 4 - Architectural Overview
- 5 - USB Data Flow Model
- 9 - USB Device Frame Work, and
- 10 - USB Host Hardware and Software.

Peripheral hardware (Electronics) designers on the other hand may only need to read chapters,

- 4 - Architectural Overview
- 5 - USB Data Flow Model
- 6 - Mechanical, and
- 7 - Electrical.

USB in a NutShell for Peripheral Designers

Now lets face it, (1) most of us are here to develop USB peripherals and (2) it's common to read a standard and still have no idea how to implement a device. So in the next 7 chapters we focus on the relevant parts needed to develop a USB device. This allows you to grab a grasp of USB and its issues allowing you to further research the issues specific to your application.

The USB 1.1 standard was complex enough before High Speed was thrown into USB 2.0. In order to help understand the fundamental principals behind USB, we omit many areas specific to High Speed devices.

Introducing the Universal Serial Bus

USB version 1.1 supported two speeds, a full speed mode of 12Mbits/s and a low speed mode of 1.5Mbits/s. The 1.5Mbits/s mode is slower and less susceptible to EMI, thus reducing the cost of ferrite beads and quality components. For example, crystals can be replaced by cheaper resonators. USB 2.0 which is still yet to see day light on mainstream desktop computers has upped the stakes to 480Mbits/s. The 480Mbits/s is known as High Speed mode and was a tack on to compete with the Firewire Serial Bus.

USB Speeds

- High Speed - 480Mbits/s
- Full Speed - 12Mbits/s
- Low Speed - 1.5Mbits/s

The Universal Serial Bus is host controlled. There can only be one host per bus. The specification in itself, does not support any form of multimaster arrangement. However the On-The-Go specification which is a tack on standard to USB 2.0 has introduced a Host Negotiation Protocol which allows two devices negotiate for the role of host. This is aimed at and limited to single point to point connections such as a mobile phone and personal organiser and not multiple hub, multiple device desktop configurations. The USB host is responsible for undertaking all transactions and scheduling bandwidth. Data can be sent by various transaction methods using a token-based protocol.

In my view the bus topology of USB is somewhat limiting. One of the original intentions of USB was to reduce the amount of cabling at the back of your PC. Apple people will say the idea came from the Apple Desktop Bus, where both the keyboard, mouse and some other peripherals could be connected together (daisy chained) using the one cable.

However USB uses a tiered star topology, simular to that of 10BaseT Ethernet. This imposes the use of a hub somewhere, which adds to greater expense, more boxes on your desktop and more cables. However it is not as bad as it may seem. Many devices have USB hubs integrated into them. For example, your keyboard may contain a hub which is connected to your computer. Your mouse and other devices such as your digital camera can be plugged easily into the back of your keyboard. Monitors are just another peripheral on a long list which commonly have in-built hubs.

This tiered star topology, rather than simply daisy chaining devices together has some benefits. Firstly power to each device can be monitored and even switched off if an overcurrent condition occurs without disrupting other USB devices. Both high, full and low speed devices can be supported, with the hub filtering out high speed and full speed transactions so lower speed devices do not receive them.

Up to 127 devices can be connected to any one USB bus at any one given time. Need more devices? - simply add another port/host. While most earlier USB hosts had two ports, most manufacturers have seen this as limiting and are starting to introduce 4 and 5 port host cards with an internal port for hard disks etc. The early hosts had one USB controller and thus both ports shared the same available USB bandwidth. As bandwidth requirements grew, we are starting to see multi-port cards with two or more controllers allowing individual channels.

The USB host controllers have their own specifications. With USB 1.1, there were two Host Controller Interface Specifications, UHCI (Universal Host Controller Interface) developed by Intel which puts more of the burden on software (Microsoft) and allowing for cheaper hardware and the OHCI (Open Host Controller Interface) developed by Compaq, Microsoft and National Semiconductor which places more of the burden on hardware(Intel) and makes for simpler software. Typical hardware / software engineer relationship. . .

With the introduction of USB 2.0 a new Host Controller Interface Specification was needed to describe the register level details specific to USB 2.0. The EHCI (Enhanced Host Controller Interface) was born. Significant Contributors include Intel, Compaq, NEC, Lucent and Microsoft so it would hopefully seem they have pooled together to provide us one interface standard and thus only one new driver to implement in our operating systems. Its about time.

USB as its name would suggest is a serial bus. It uses 4 shielded wires of which two are power (+5v & GND). The remaining two are twisted pair differential data signals. It uses a NRZI (Non Return to Zero Invert) encoding scheme to send data with a sync field to synchronise the host and receiver clocks.

USB supports plug'n'plug with dynamically loadable and unloadable drivers. The user simply plugs the device into the bus. The host will detect this addition, interrogate the newly inserted device and load the appropriate driver all in the time it takes the hourglass to blink on your screen provided a driver is installed for your device. The end user needs not worry about terminations, terms such as IRQs and port addresses, or rebooting the computer. Once the user is finished, they can simply lug the cable out, the host will detect its absence and automatically unload the driver.

The loading of the appropriate driver is done using a PID/VID (Product ID/Vendor ID) combination. The VID is supplied by the USB Implementor's forum at a cost and this is seen as another sticking point for USB. The latest info on fees can be found on the USB Implementor's Website

Other standards organisations provide a extra VID for non-commercial activities such as teaching, research or fiddling (The Hobbyist). The USB Implementors forum has yet to provide this service. In these cases you may wish to use one assigned to your development system's manufacturer. For example most chip manufacturers will have a VID/PID combination you can use for your chips which is known not to exist as a commercial device. Other chip manufacturers can even sell you a PID to use with their VID for your commercial device.

Another more notable feature of USB, is its transfer modes. USB supports Control, Interrupt, Bulk and Isochronous transfers. While we will look at the other transfer modes later, Isochronous allows a device to reserve a defined amount of bandwidth with guaranteed latency. This is ideal in Audio or Video applications where congestion may cause loss of data or frames to drop. Each transfer mode provides the designer trade-offs in areas such as error detection and recovery, guaranteed latency and bandwidth.

Chapter 2 : Hardware