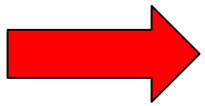
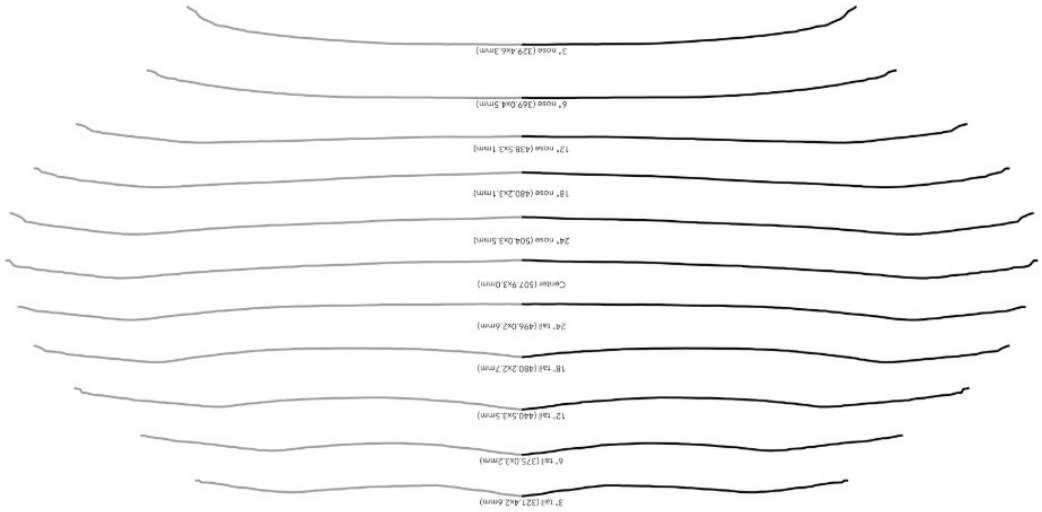
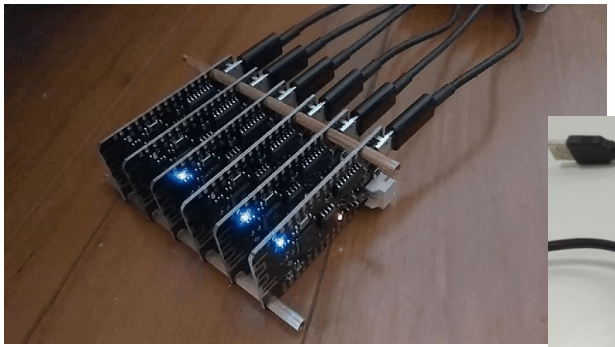
How to profile a surfboard?

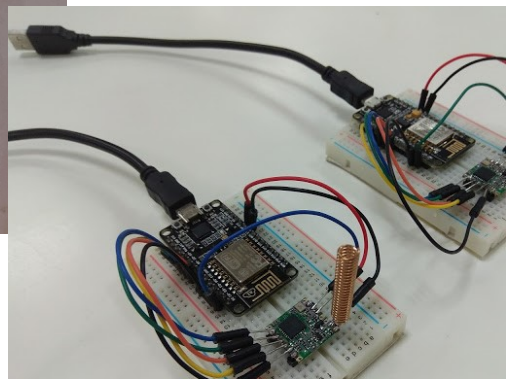




About Me

- **Wei Lin**
  - fb : https://www.facebook.com/**wei.lin.921025**
- GitHub:
  - https://github.com/**Wei1234c**
- PyCon TW:
  - Building Distributed System with Celery on Docker Swarm
  - Elastic Network of Things with MQTT and MicroPython
- Projects:
  - ESP32 cluster,
  - LoRa transceiver driver
  - LoRa cellular networks
  - ...





USB (2.0) Introduction

https://github.com/Wei1234c/Universal_Serial_Bus/blob/master/notebooks/USB%202.0%20Introduction.pdf

Wei Lin
20191221

Motivation of USB

Goals of USB

- Comprehension of various PC configurations and form factors
- Low-cost solution that supports transfer rates up to 480 Mb/s
- Integration in commodity device technology
- **Full support for real-time data for voice, audio, and video**
- Full backward compatibility of USB 2.0 for devices built to previous versions of the specification
- Enabling new classes of devices that augment the PC’ s capability
- **Provision of a standard interface capable of quick diffusion into product**
- **Ease-of-use for PC peripheral expansion**
- Protocol flexibility for mixed-mode isochronous data transfers and asynchronous messaging

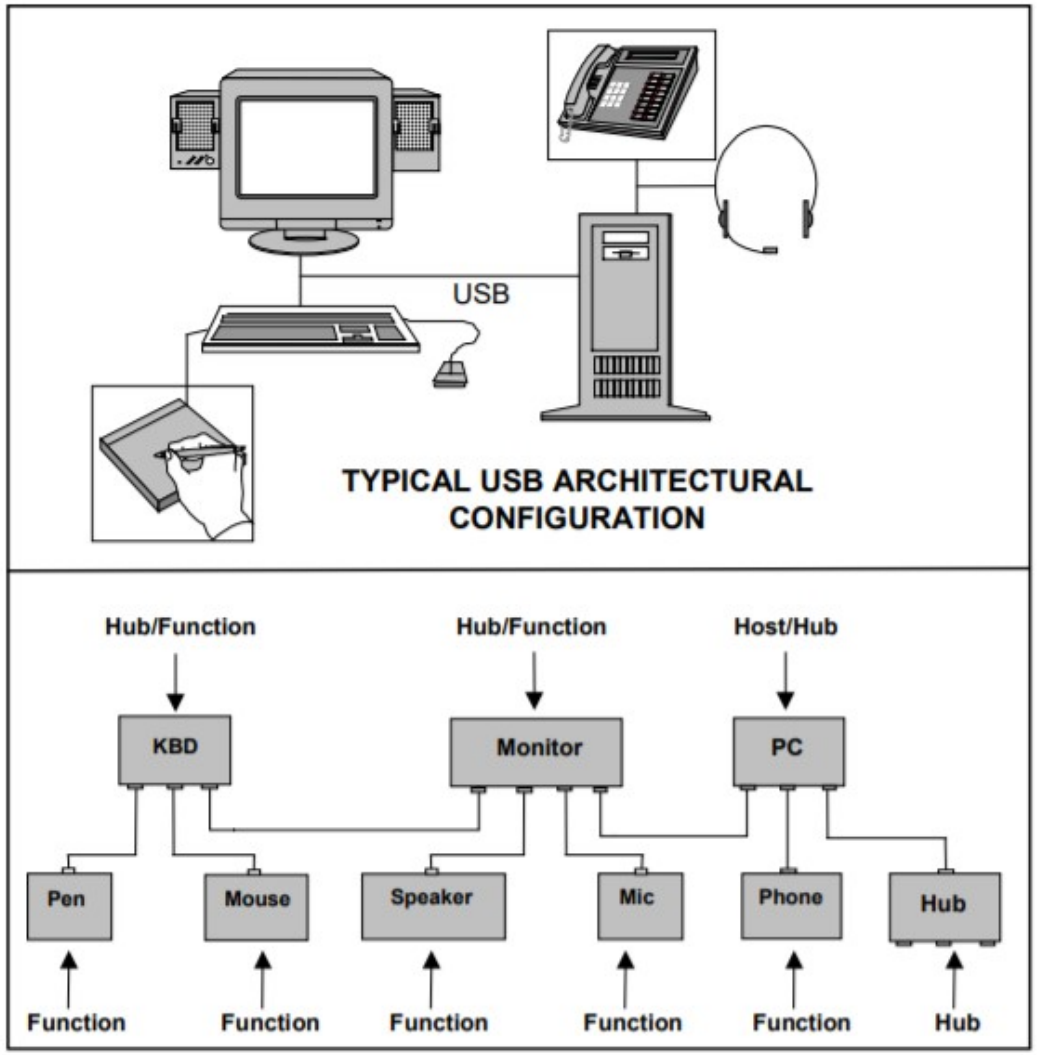
USB documents

- USB 2.0:
  - https://zh.wikipedia.org/wiki/USB
  - https://www.usb.org/document-library/usb-20-specification
- USB HID:
  - https://www.usb.org/documents?search=hid&items_per_page=50
- USB Audio:
  - https://www.usb.org/document-library/audio-devices-rev-20-and-adopters-agreement

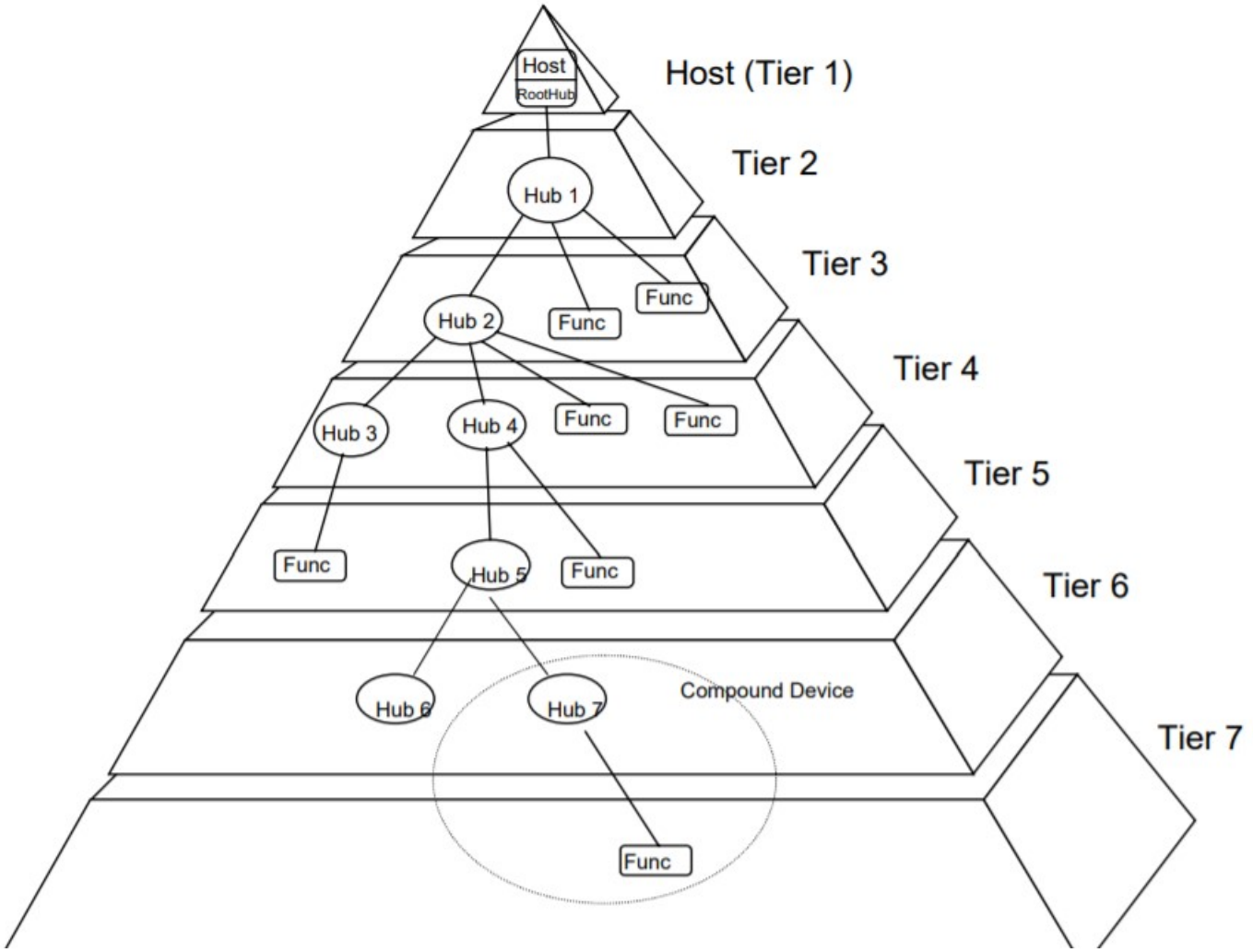
Other documents

- USB in a NutShell
  - https://www.beyondlogic.org/usbnutshell/usb1.shtml
- USB made simple:
  - http://www.usbmadesimple.co.uk
- 成大資工 on USB:
  - http://wiki.csie.ncku.edu.tw/embedded/USB
- USB 實驗室
  - http://www.usblab.idv.tw/
- USB 101: An Introduction to Universal Serial Bus 2.0
  - https://www.cypress.com/file/134171/download

An USB environment



Bus Topology

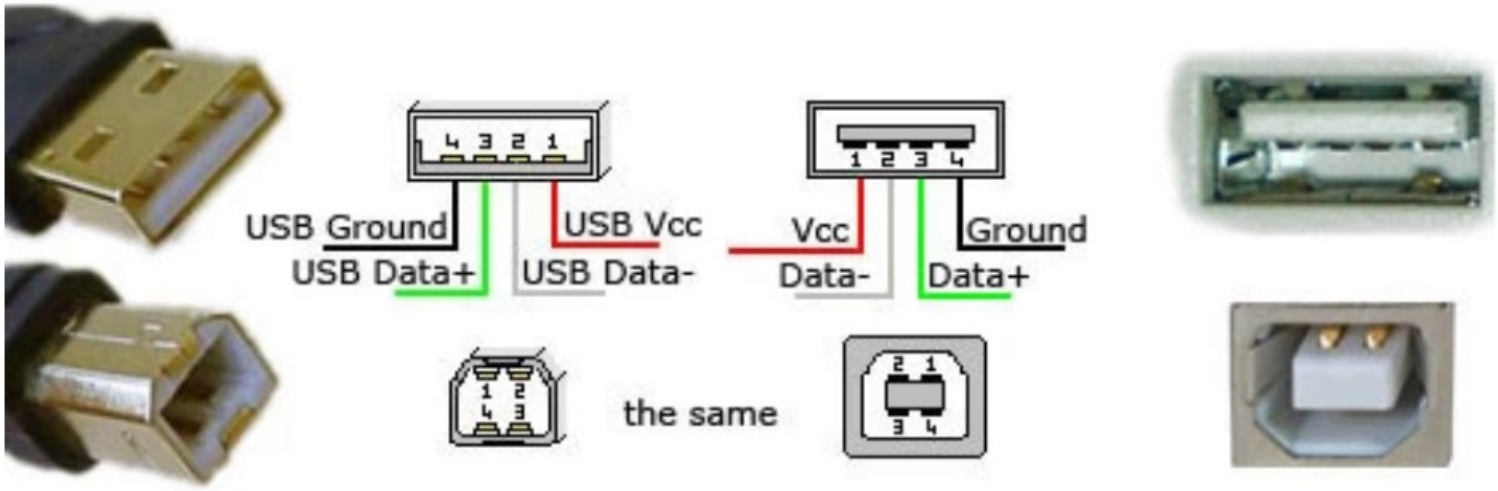


Layers – from ground up

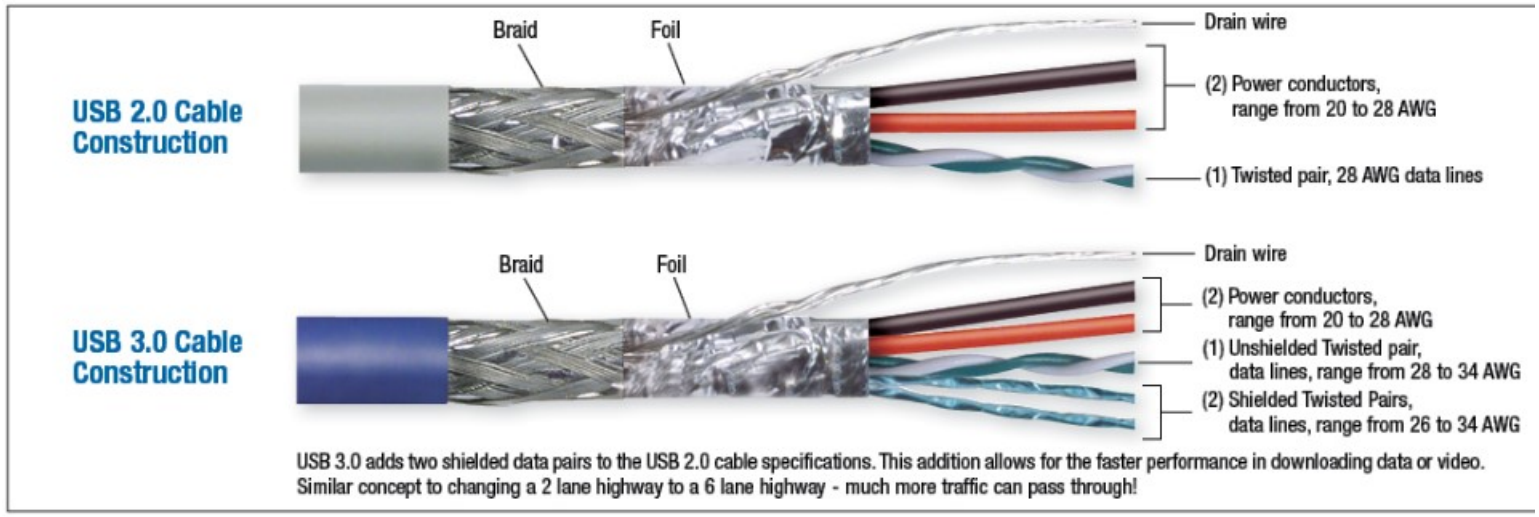
Connectors


USB 2.0 Type A Plug
USB 2.0 Type A Jack
USB 3.0 Type A Plug
USB 3.0 Type A Jack
USB 2.0 Type B Plug
USB 2.0 Type B Jack
USB 3.0 Type B Plug
USB 3.0 Type B Jack
USB 2.0 Mini Type B Plug (4 Position)
USB 2.0 Type B Jack (4 Position)
USB 2.0 Micro Type B Plug
USB 2.0 Micro Type B Jack
USB 2.0 Mini Type B Plug (5 Position)
USB 2.0 Type B Jack (5 Position)
USB 3.0 Micro Type B Plug
USB 3.0 Micro Type B Jack

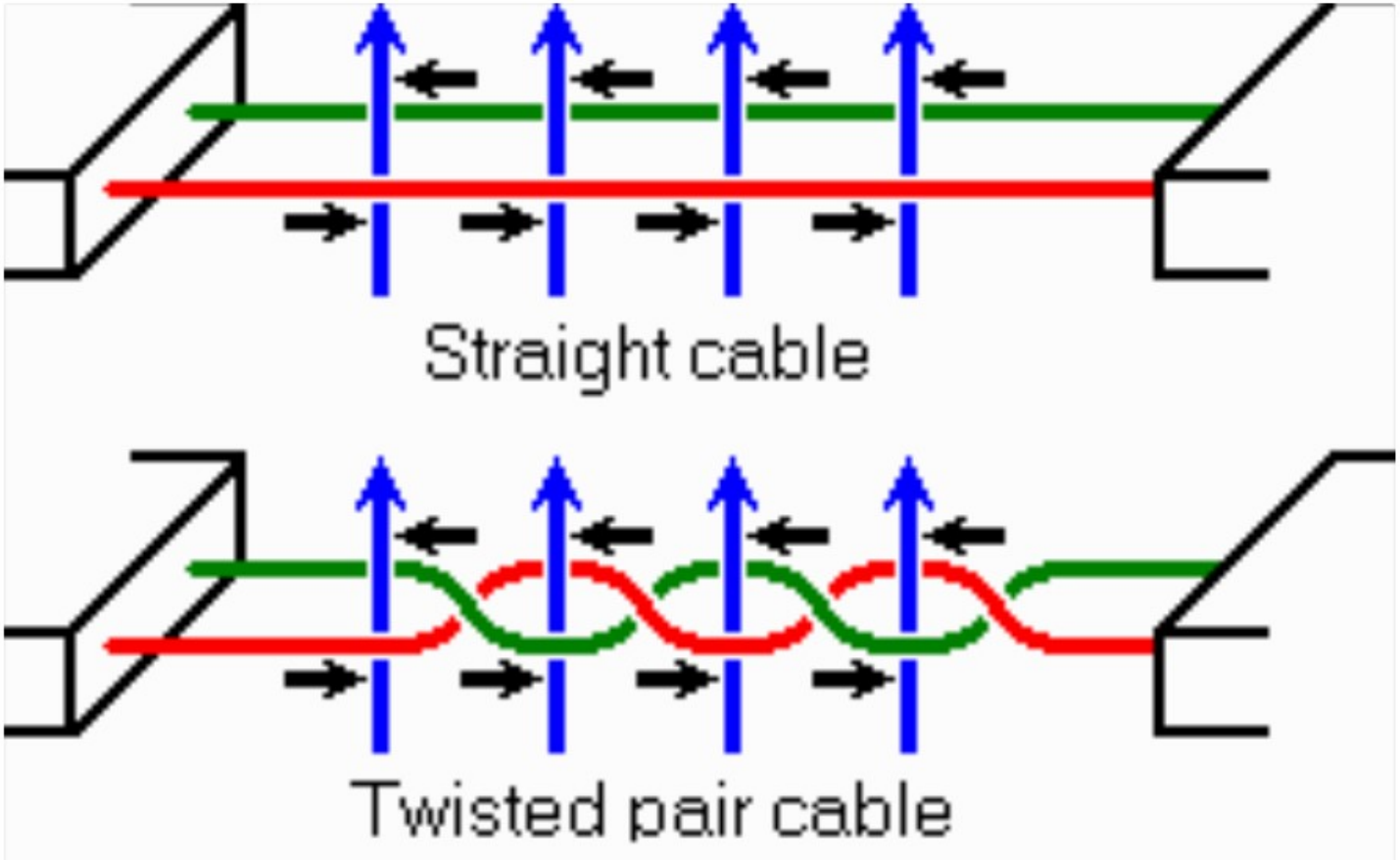
Pin outs


4 3 2 1
USB Ground
USB Data+
USB Data-
USB Vcc
1 2 3 4
Vcc
Data-
Data+
Ground
1 2
4 3
the same
2 1
3 4

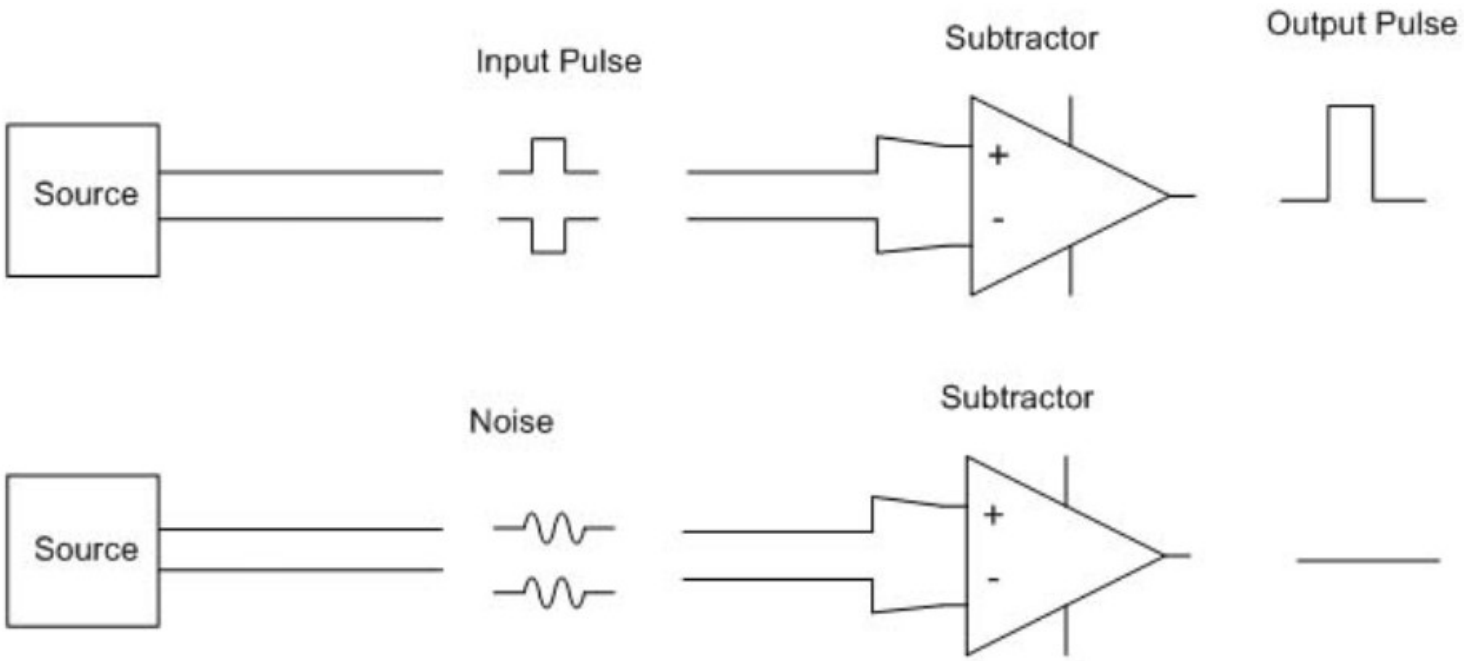
Inside the cable



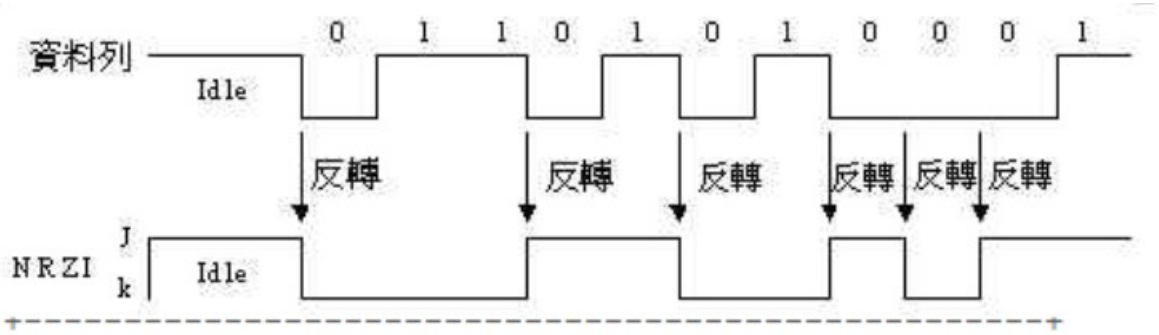
Why twisted?

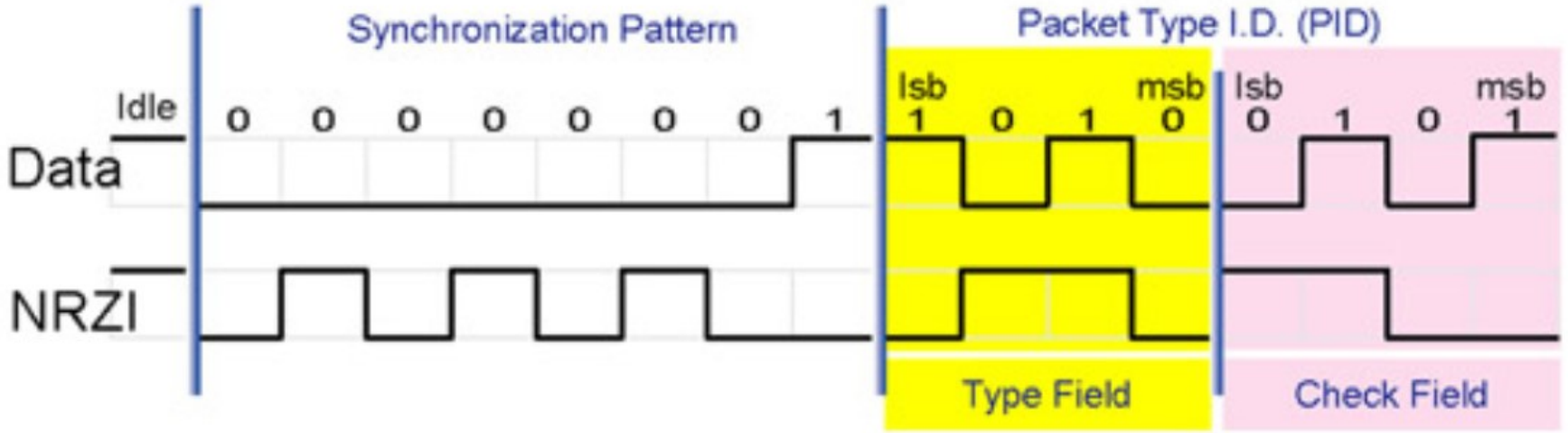


Why differential?



NRZI(Non Return to Zero Invert)





Something special about USB

- Asynchronous
- Half-duplex
- Centralized
  - Only host can start a transaction
  - In, Out @ host’ s viewpoint
- Token is Addressed

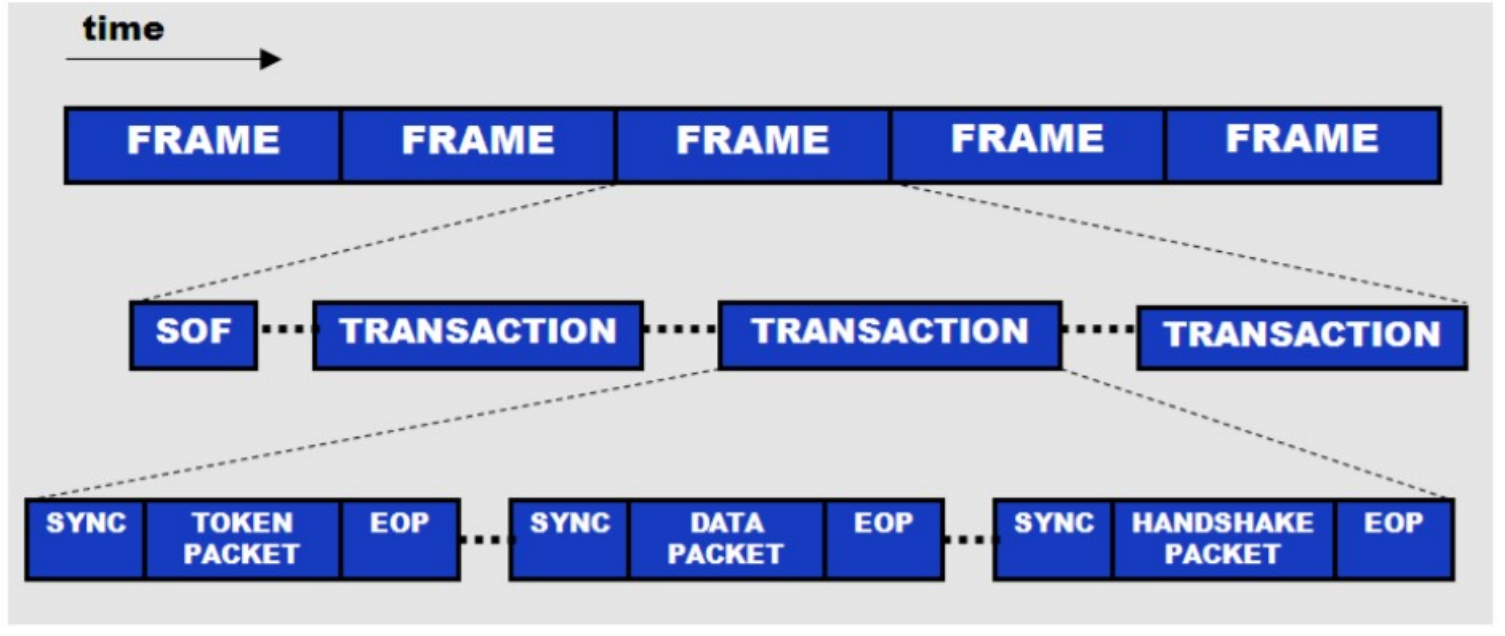
(lsb) (msb)

PID	ADDR	ENDP	CRC5
8	7	4	5

bits

- Bits/Bytes order
  - Little endian
    - 0x09 => 1001 0000
    - 0x05 => 1010 0000
    - 0x0905 => 0x05, 0x09 => 1010 0000 1001 0000

USB Communication



Packet types

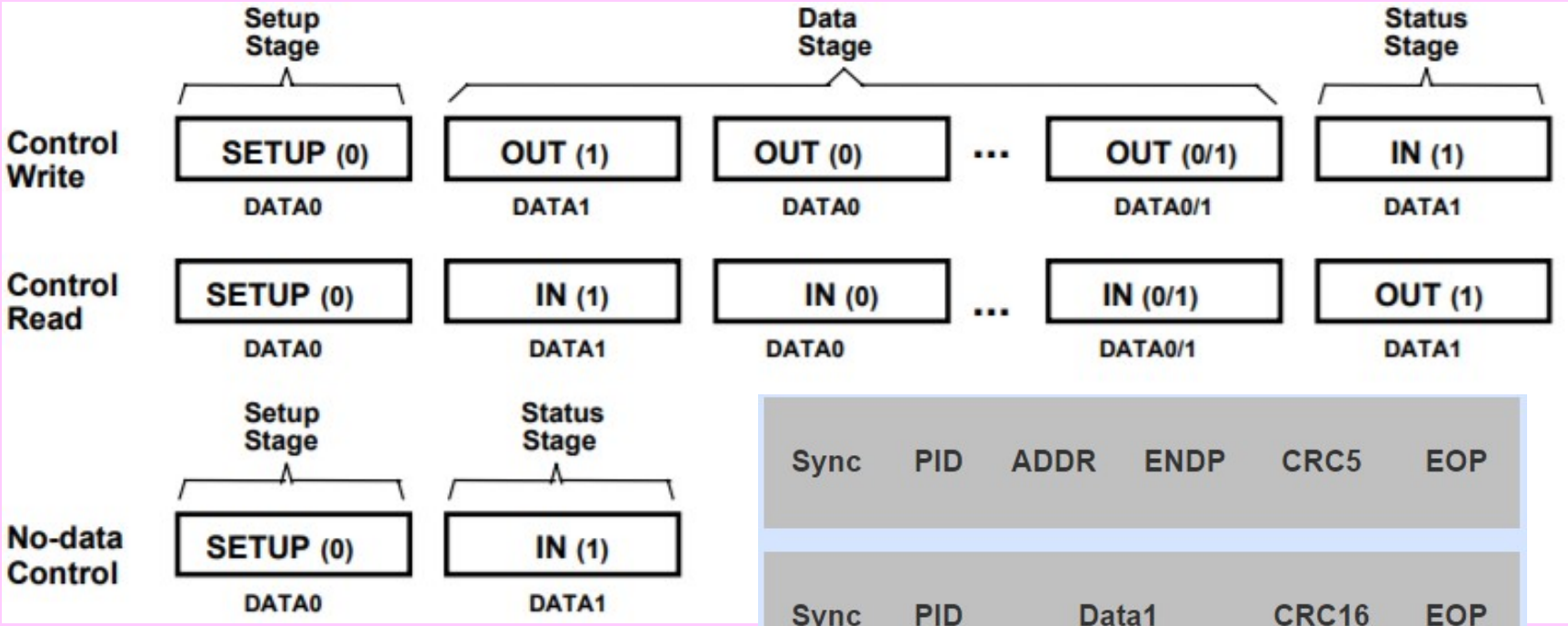
https://www.beyondlogic.org/usbnutshell/usb3.shtml#USBProtocols

Group	PID Value	Packet Identifier
Token	0001	OUT Token
	1001	IN Token
	0101	SOF Token
	1101	SETUP Token
Data	0011	DATA0
	1011	DATA1
	0111	DATA2
	1111	MDATA
Handshake	0010	ACK Handshake
	1010	NAK Handshake
	1110	STALL Handshake
	0110	NYET (No Response Yet)
Special	1100	PREamble
	1100	ERR
	1000	Split
	0100	Ping

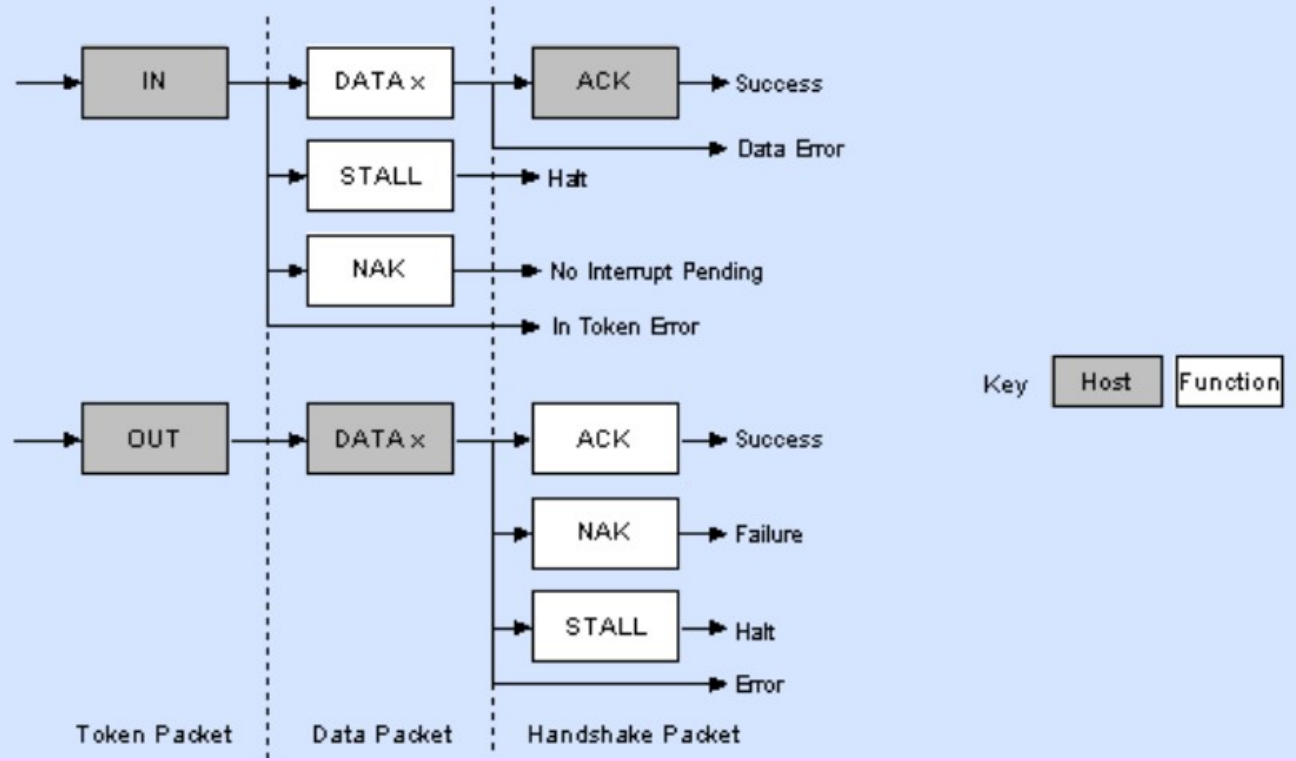
Transfer (Transaction) types

- Control Transfers
- Isochronous Transfers
- Interrupt Transfers
- Bulk Transfers

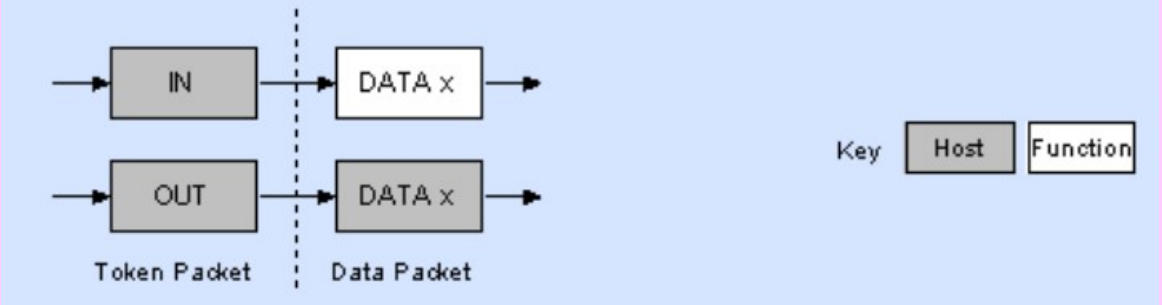
Control Transfers
(Figure 8-37)



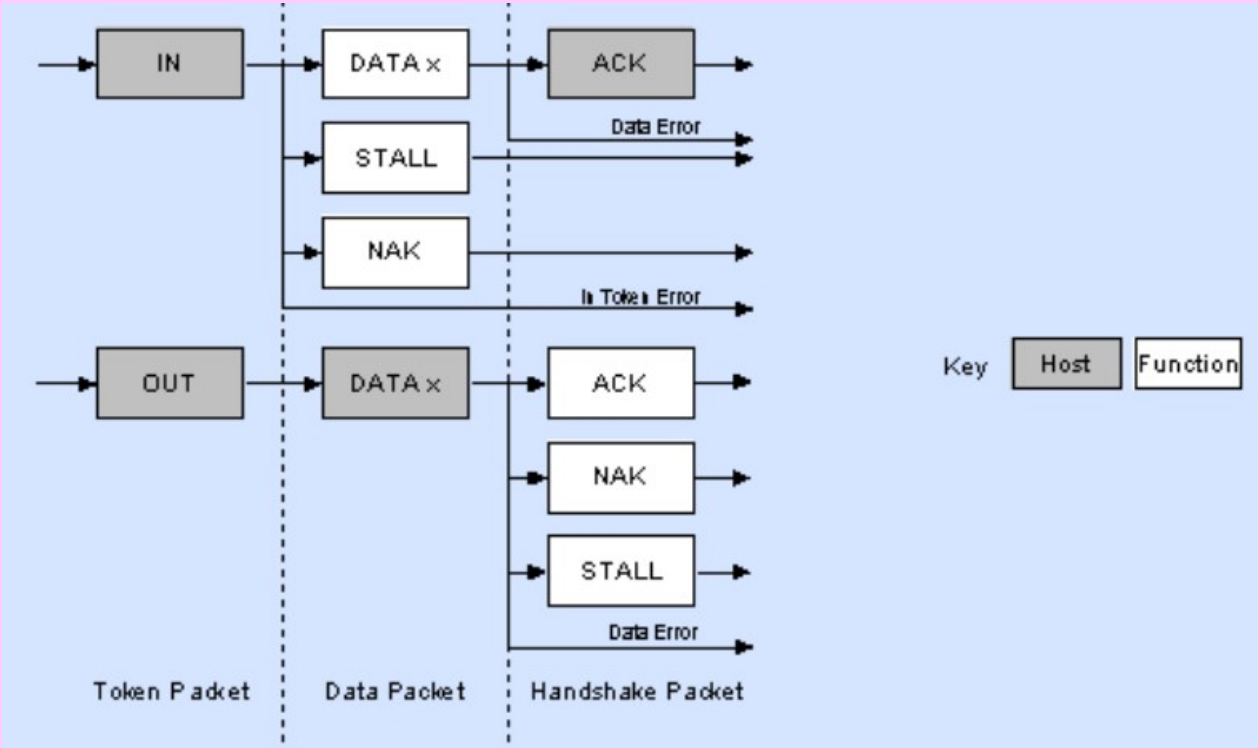
Interrupt Transfers



Isochronous Transfers



Bulk Transfers

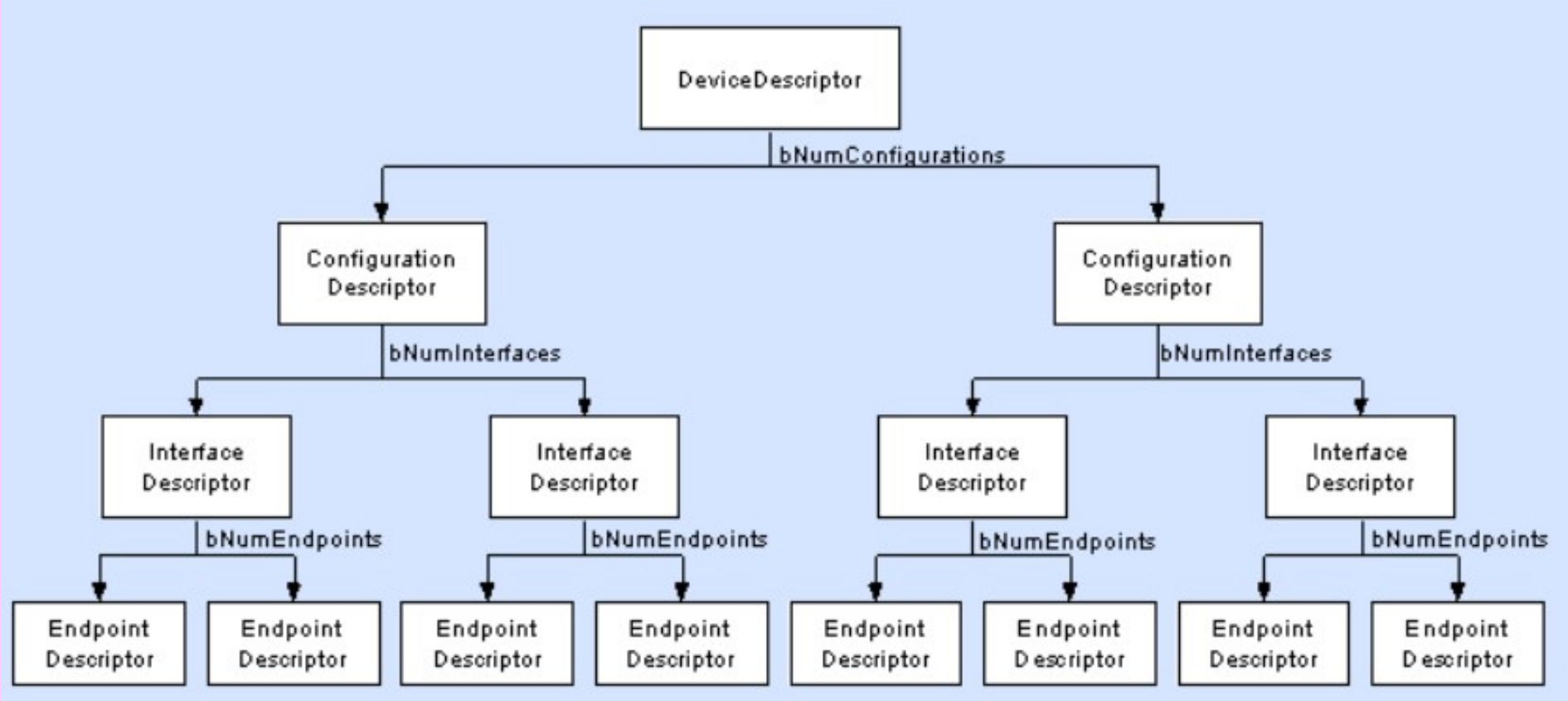


Standard Requests

Standard Request Codes (Table 9-4)

bRequest	Value
GET_STATUS	0
CLEAR_FEATURE	1
Reserved for future use	2
SET_FEATURE	3
Reserved for future use	4
SET_ADDRESS	5
GET_DESCRIPTOR	6
SET_DESCRIPTOR	7
GET_CONFIGURATION	8
SET_CONFIGURATION	9
GET_INTERFACE	10
SET_INTERFACE	11
SYNCH_FRAME	12

USB Descriptors



Descriptor Types (Table 9-5)

Descriptor Types	Value
DEVICE	1
CONFIGURATION	2
STRING	3
INTERFACE	4
ENDPOINT	5
DEVICE_QUALIFIER	6
OTHER_SPEED_CONFIGURATION	7
INTERFACE_POWER[1]	8

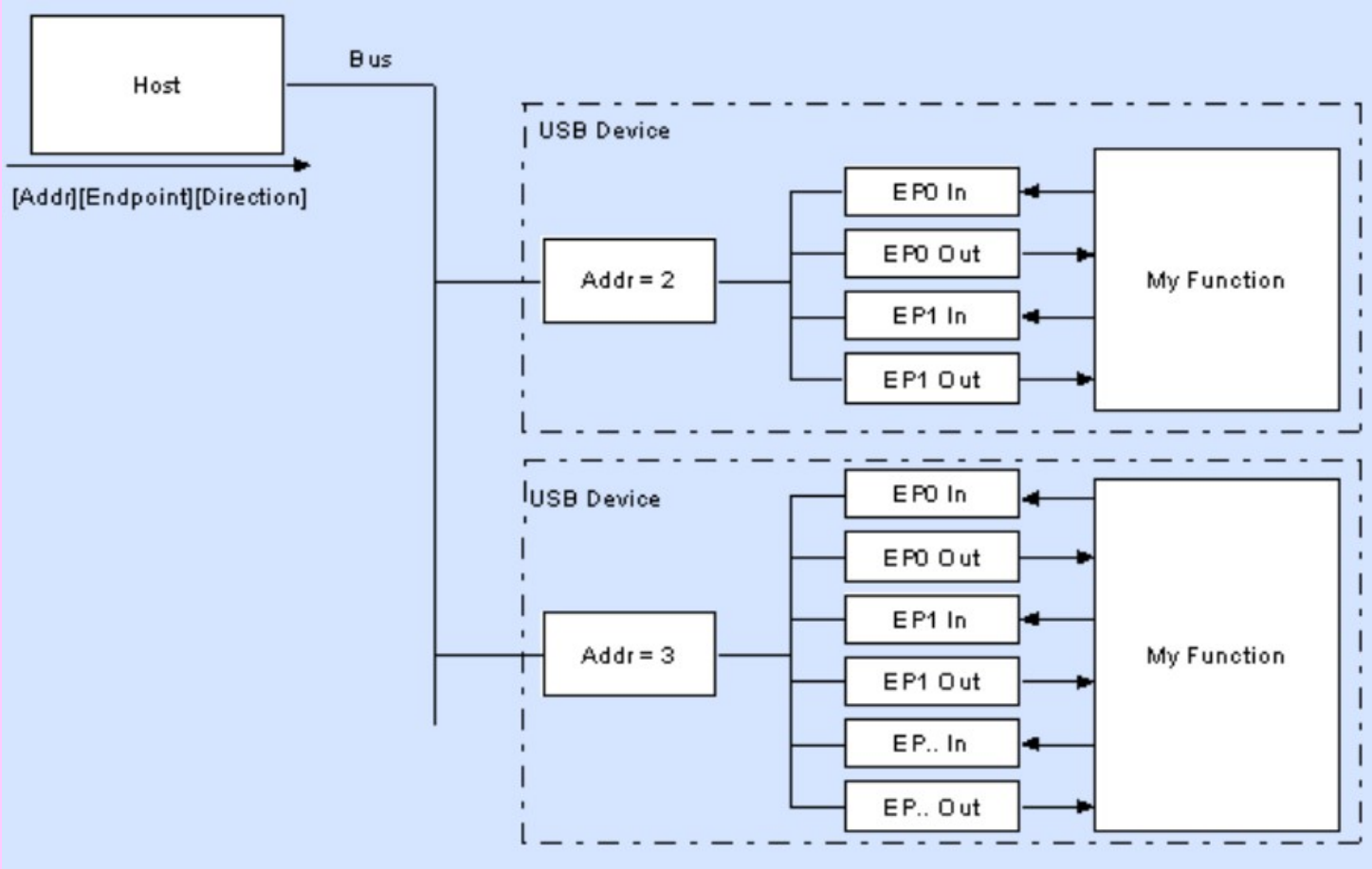
Format of Setup Data (Table 9-2)

Offset	Field	Size	Value	Description
0	*bmRequestType*	1	Bitmap	Characteristics of request: D7: Data transfer direction 0 = Host-to-device 1 = Device-to-host D6...5: Type 0 = Standard 1 = Class 2 = Vendor 3 = Reserved D4...0: Recipient 0 = Device 1 = Interface 2 = Endpoint 3 = Other 4...31 = Reserved
1	*bRequest*	1	Value	Specific request (refer to Table 9-3)
2	*wValue*	2	Value	Word-sized field that varies according to request
4	*wIndex*	2	Index or Offset	Word-sized field that varies according to request; typically used to pass an index or offset
6	*wLength*	2	Count	Number of bytes to transfer if there is a Data stage

Standard Device Requests (Table 9-3)

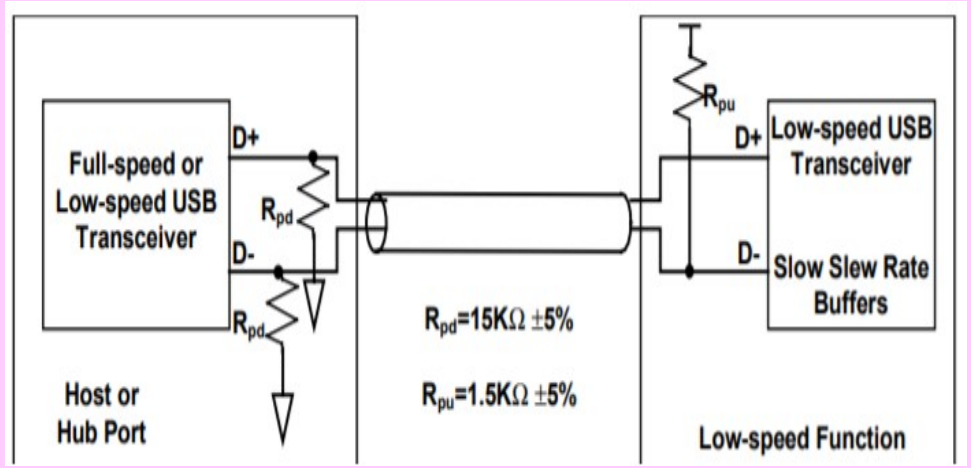
bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B 00000001B 00000010B	CLEAR_FEATURE	Feature Selector	Zero Interface Endpoint	Zero	None
10000000B	GET_CONFIGURATION	Zero	Zero	One	Configuration Value
10000000B	GET_DESCRIPTOR	Descriptor Type and Descriptor Index	Zero or Language ID	Descriptor Length	Descriptor
10000001B	GET_INTERFACE	Zero	Interface	One	Alternate Interface
10000000B 10000001B 10000010B	GET_STATUS	Zero	Zero Interface Endpoint	Two	Device, Interface, or Endpoint Status
00000000B	SET_ADDRESS	Device Address	Zero	Zero	None
00000000B	SET_CONFIGURATION	Configuration Value	Zero	Zero	None
00000000B	SET_DESCRIPTOR	Descriptor Type and Descriptor Index	Zero or Language ID	Descriptor Length	Descriptor
00000000B 00000001B 00000010B	SET_FEATURE	Feature Selector	Zero Interface Endpoint	Zero	None
00000001B	SET_INTERFACE	Alternate Setting	Interface	Zero	None
10000010B	SYNCH_FRAME	Zero	Endpoint	Two	Frame Number

USB Functions

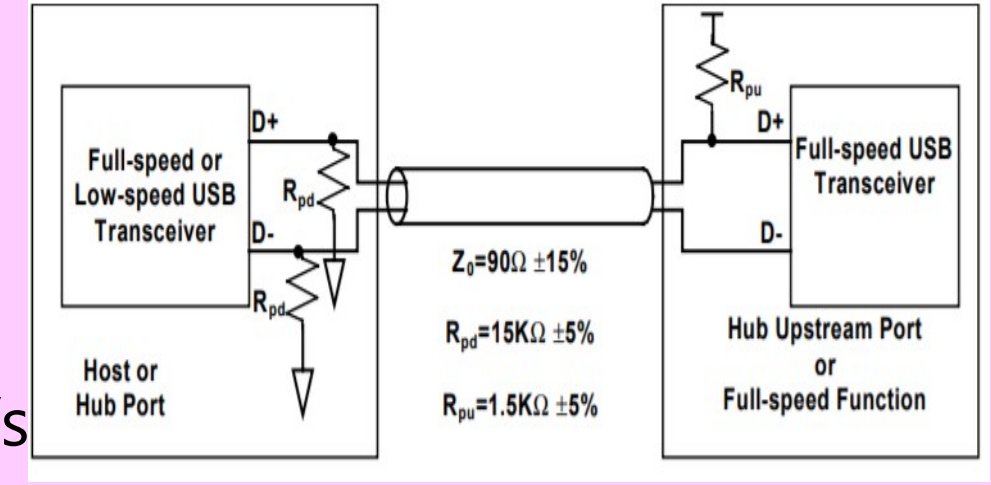


When a device is plugged in

- Low Speed (1.5Mbits/s)



- Full Speed (12Mbits/s)



- High Speed (480Mbits/s

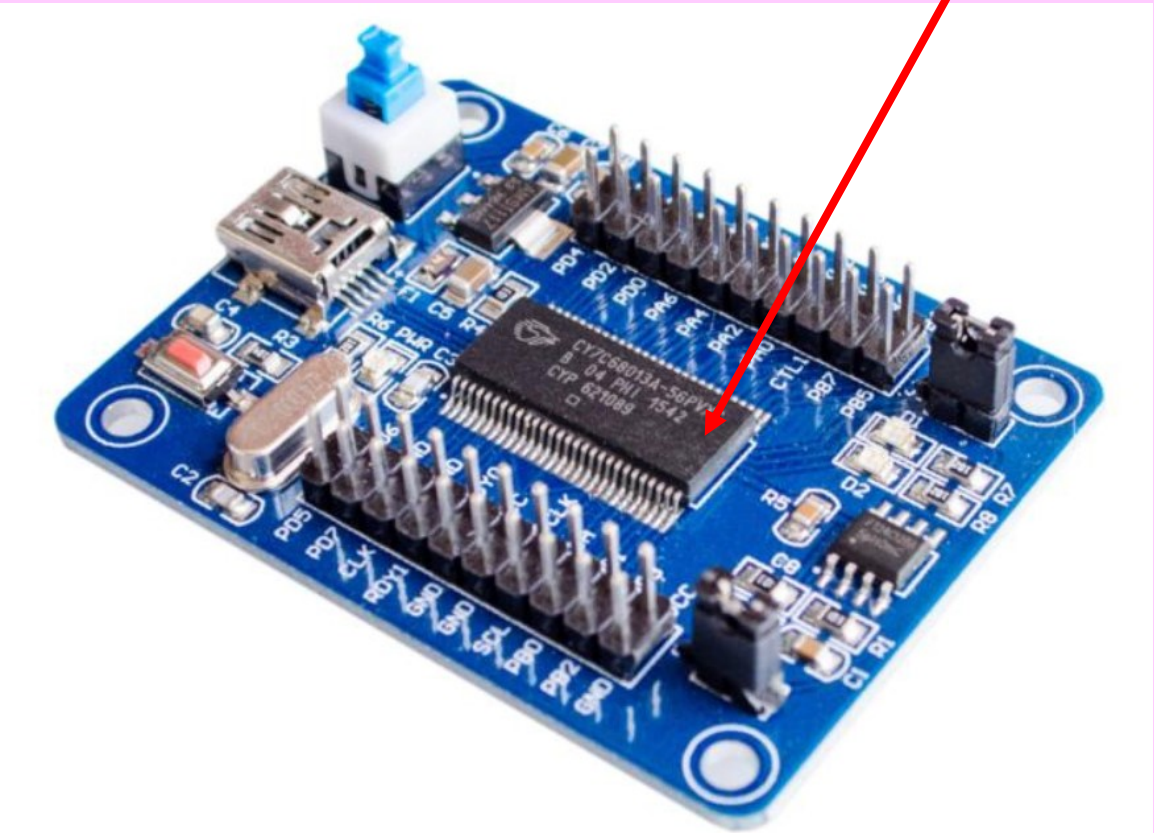
Bus Enumeration

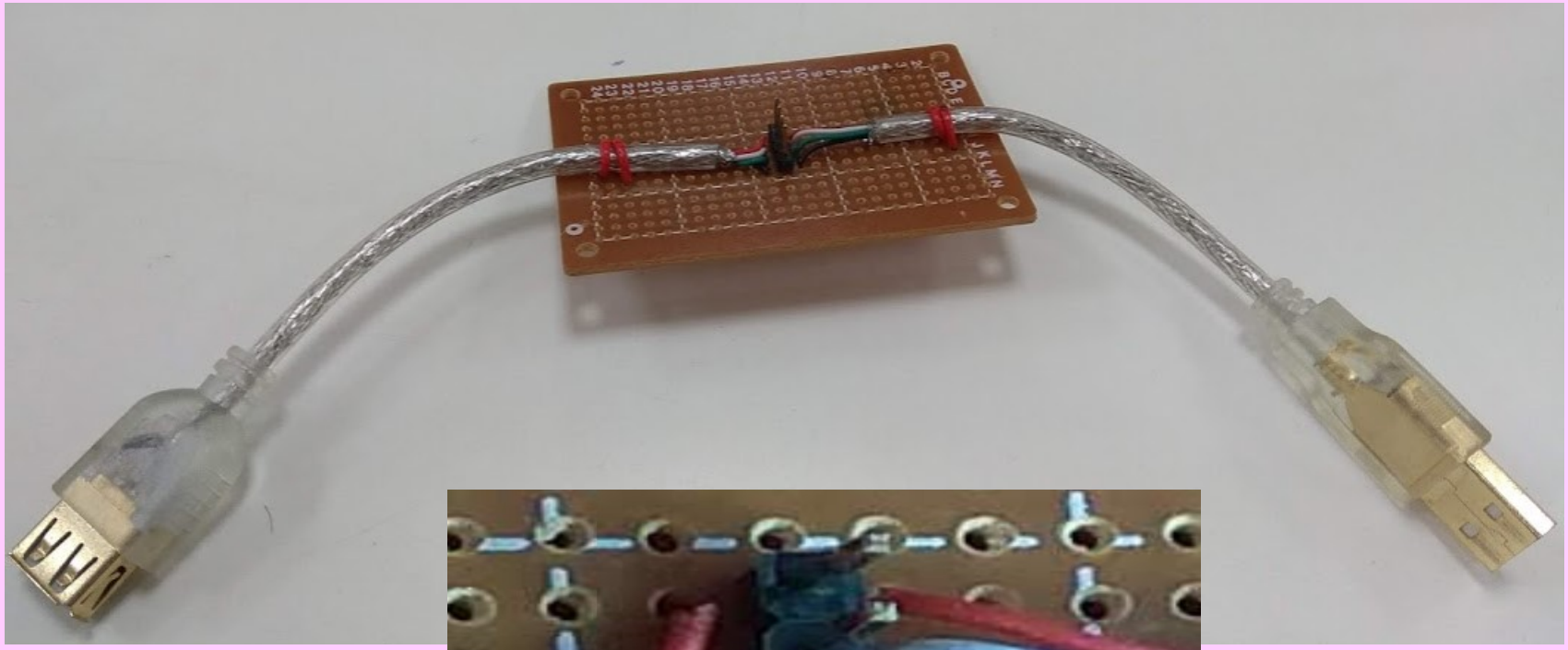
- Get device descriptor
- Set Address
- Get device descriptor
- Get configuration descriptor
- Set configuration
- Set interface

Tools

- Logic Analyzer
  - Software:
    - Sigrok **PulseView**: https://sigrok.org/wiki/PulseView
  - Hardware:
    - Cypress EZ-USB **FX2LP**: https://www.cypress.com/products/ez-usb-fx2lp

Cypress EZ-USB FX2LP (CY7C68013A)





Tracing USB packets

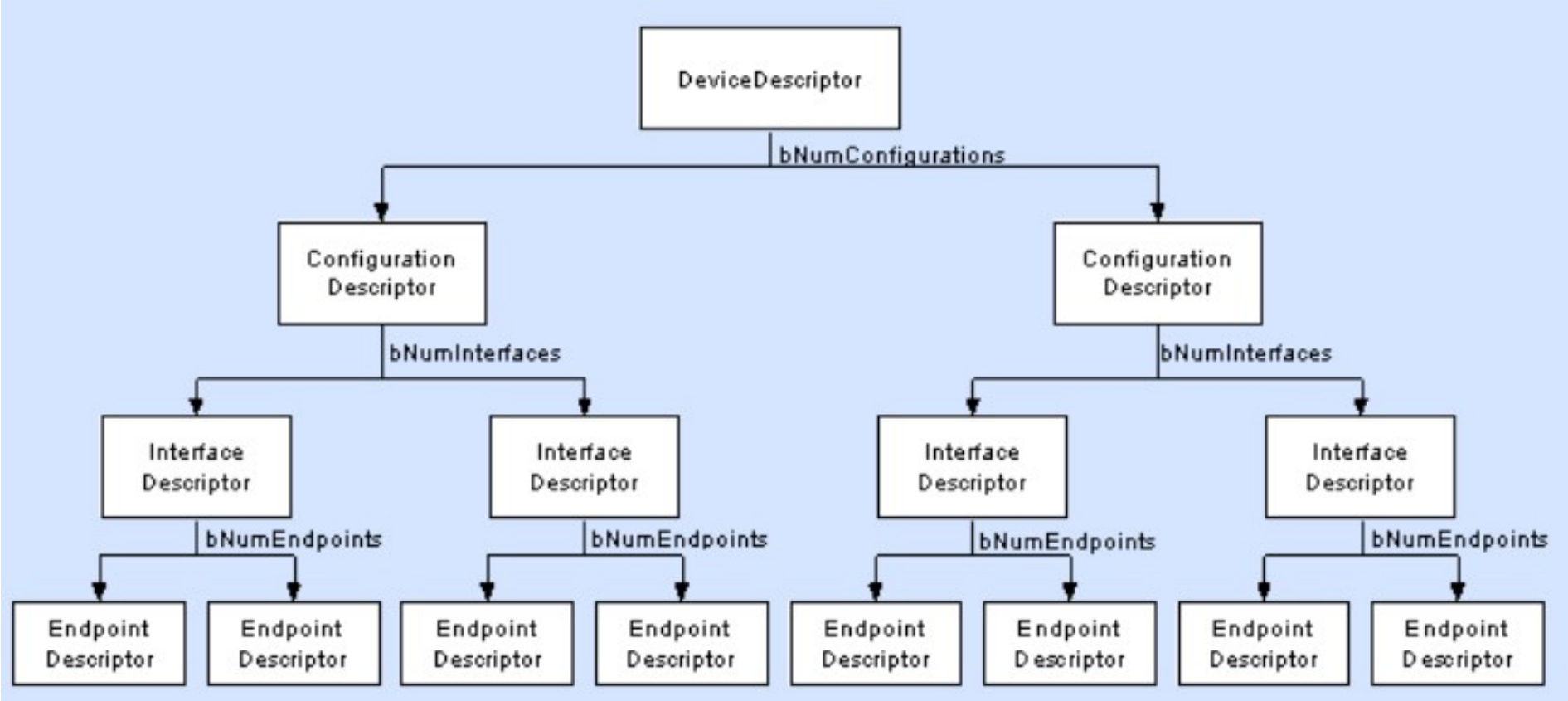
Tools

- Software
  - Packets tracking:
    - **Wireshark:** https://www.wireshark.org/
    - Bus Hound: http://www.perisoft.net/bushound/
    - Device Monitoring Studio: https://www.hhdsoftware.com/Downloads/device-monitoring-studio

Tools

- Python package:
  - PyUSB:
    - https://github.com/pyusb/pyusb
    - https://github.com/pyusb/pyusb/blob/master/docs/tutorial.rst
- My packages:
    - https://github.com/Wei1234c/Universal_Serial_Bus
    - https://github.com/Wei1234c/USB_HID
    - https://github.com/Wei1234c/USB_Audio

USB Descriptors

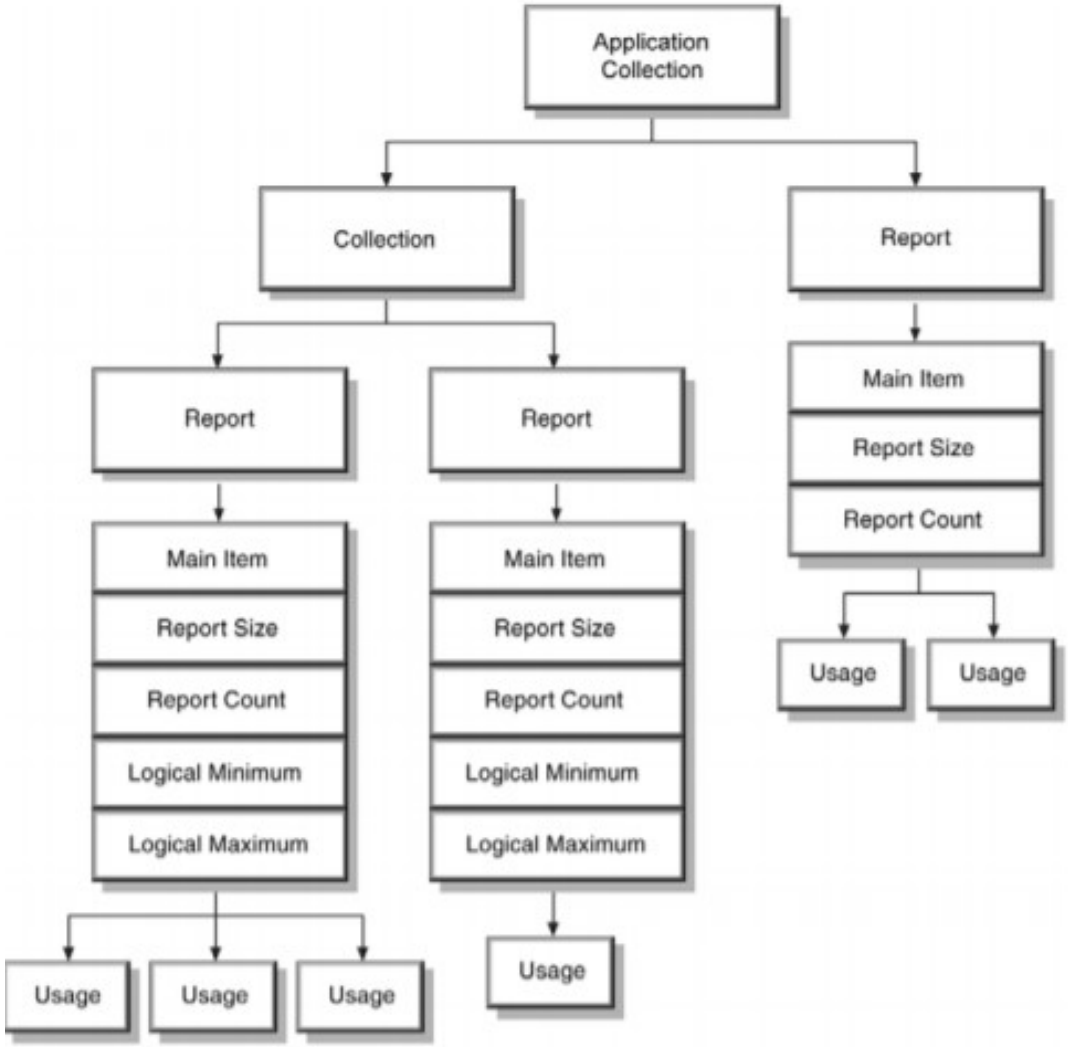


Defined Class Codes

https://www.usb.org/defined-class-codes

- Audio: 1
  - https://www.usb.org/document-library/audio-devices-rev-20-and-adopters-agreement
- HID (Human Interface Device) : 3
  - https://www.usb.org/document-library/device-class-definition-hid-111

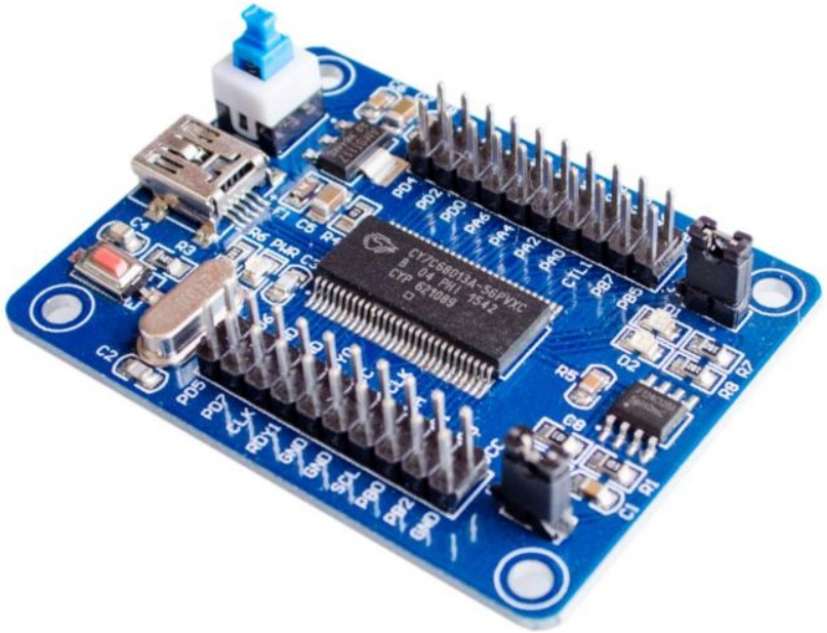
Report descriptor structure



Tracking Mouse with Python

Inside a USB device

- Hardware:
  - Cypress EZ-USB FX2LP: https://www.cypress.com/products/ez-usb-fx2lp
- SDK:
  - https://www.cypress.com/products/ez-usb-fx2lp#tabs-0-bottom_side-2
- Firmware examples:
  - https://www.cypress.com/documentation/application-notes/an65209-getting-started-fx2lp#res574
- Books:
  - https://www.books.com.tw/products/0010505575

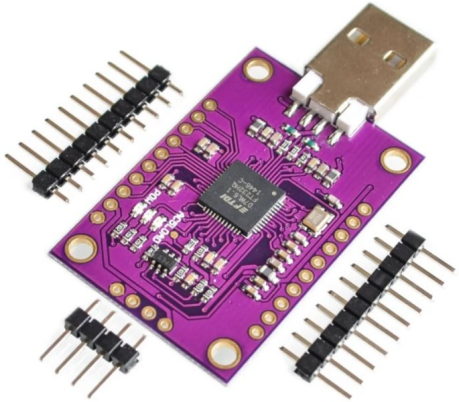



USB2.0
高速週邊裝置設計
之實務應用

Vendor defined device

https://www.cypress.com/documentation/application-notes/an45471-create-your-own-usb-vendor-commands-using-fx2lp

Tools



- Hardware:
  - FTDI FT232H: https://www.ftdichip.com/Products/ICs/FT232H.htm
- Python modules:
  - PyFtdi: https://github.com/eblot/pyftdi
- My packages:
  - Bridges: https://github.com/Wei1234c/Bridges

Bridges as I2C

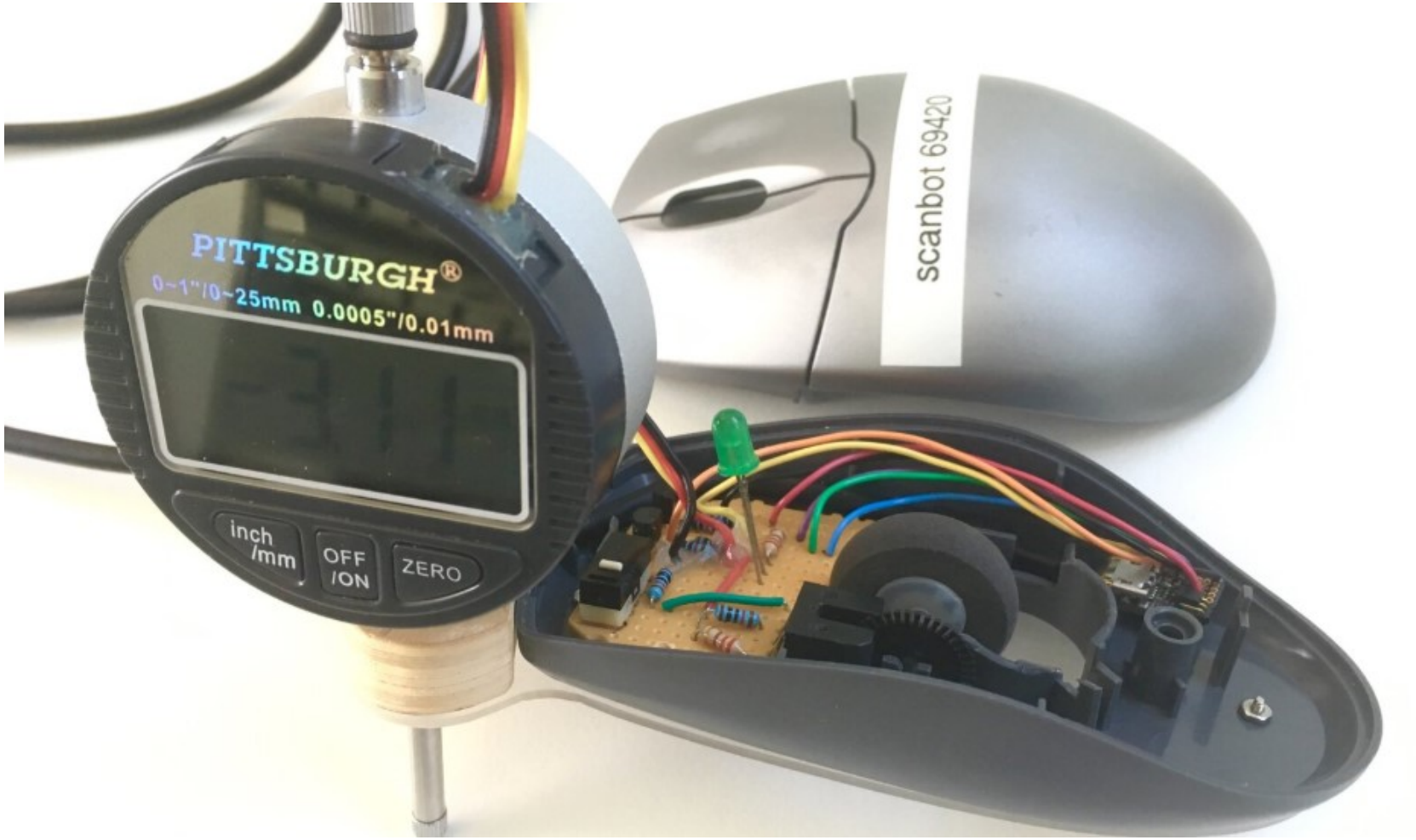
https://github.com/Wei1234c/Bridges

```
# On real Raspberry Pi
from smbus2 import SMBus

bus = SMBus(1)
b = bus.read_byte_data(80, 0)
print(b)
```

```
# On PC with FTDI FT232H
from bridges.ftdi.controllers.i2c import I2cController
SMBus = I2cController().SMBus

bus = SMBus(1)
b = bus.read_byte_data(80, 0)
print(b)
```

Profile a surfboard

http://ryanschenk.com/code/scanbot.html



Profile a surfboard

http://ryanschenk.com/code/scanbot.html

Q & A