ECE 485/585
Microprocessor System Design

Prof. Mark G. Faust

Maseeh College of Engineering
and Computer Science

PORTLAND STATE UNIVERSITY

Serial Buses

ECE 485/585

Mark G. Faust

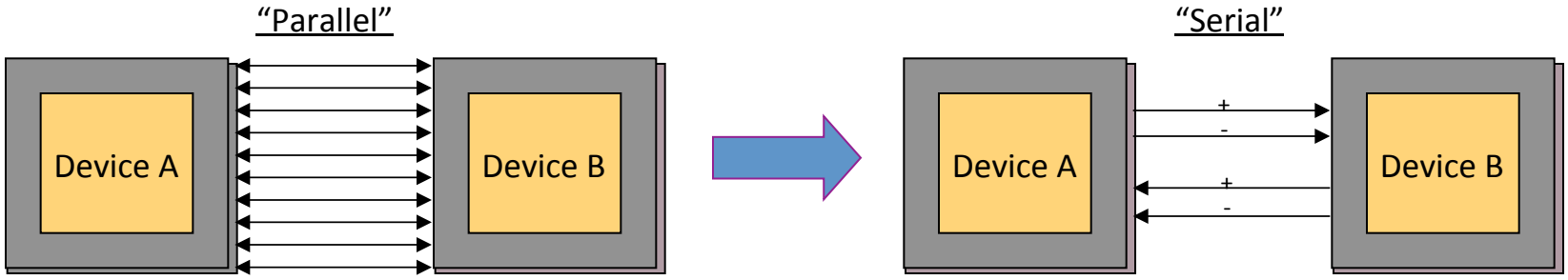
Serial Buses

- Bus Trends
- Clocking
- Line Codes
- Differential Signaling
- USB 2.0, USB 3.0

Bus Trends

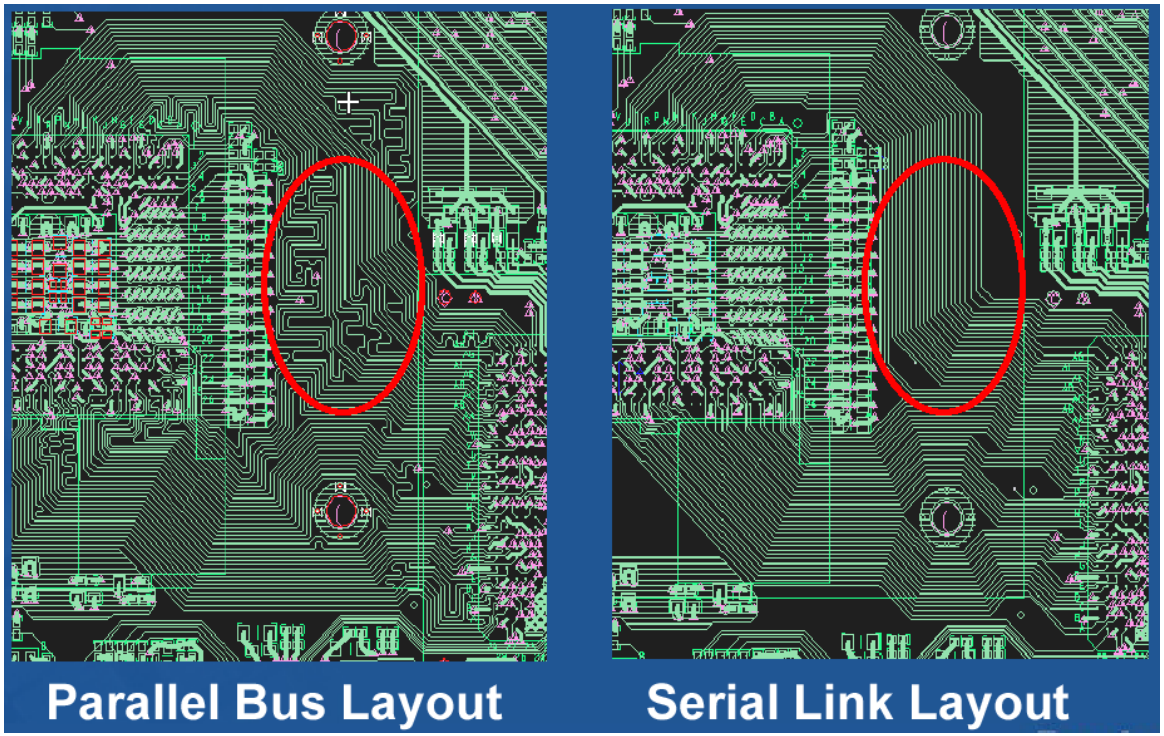
- Parallel to Serial
- Lower Voltage
  - Can't accomplish large voltage swings at high speed
  - Power consumption
- Differential Signaling
- Clock Forwarding and Clock Embedding
- Encoding
  - Line codes: NRZI
  - 8b/10b, "bit-stuffing"

Why Serial?

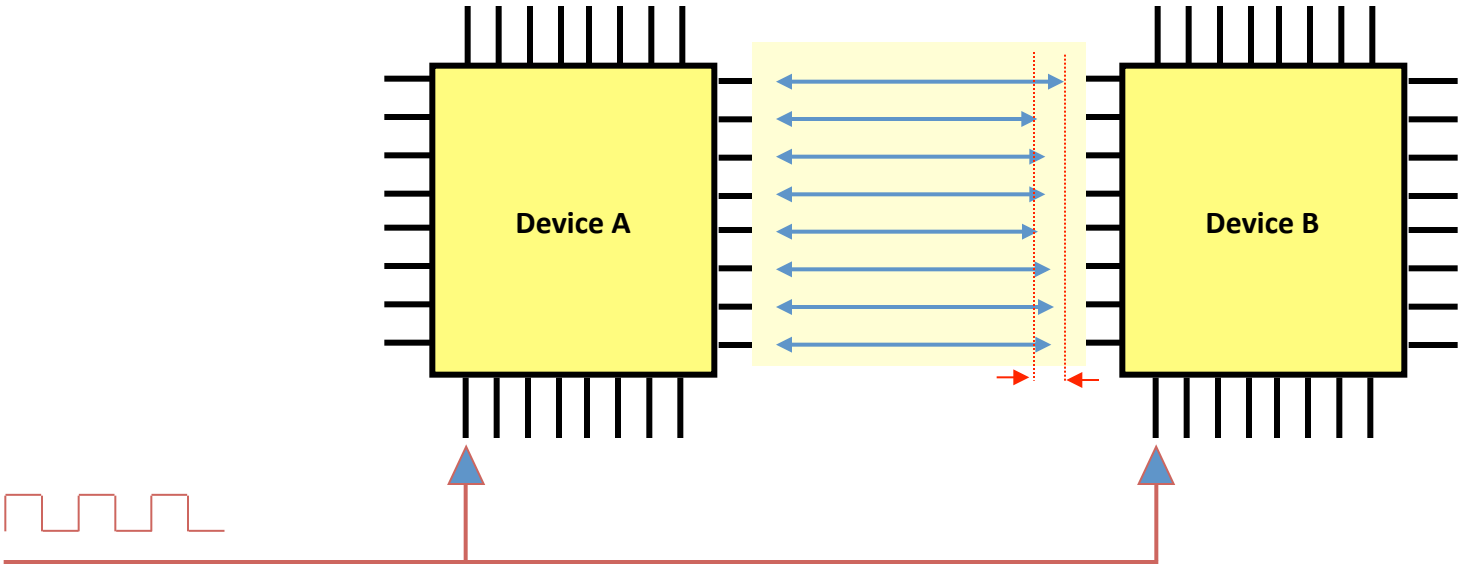


10 bidirectional wires at 250Mbps

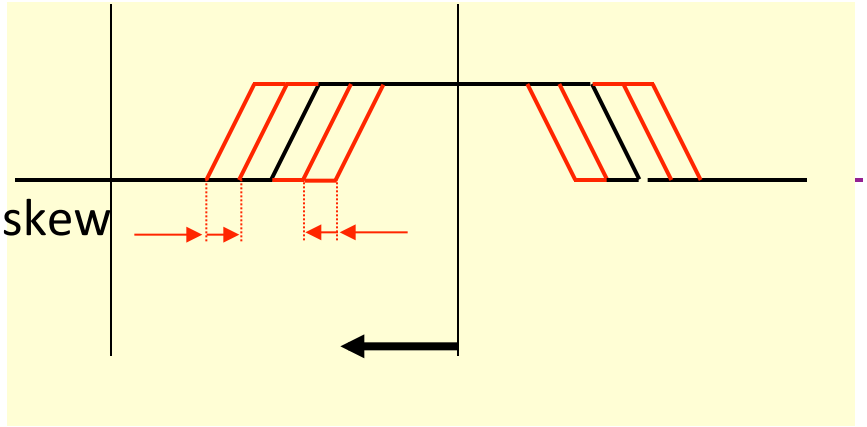
pair unidirectional wires at 2500Mbps (2.5Gbps)



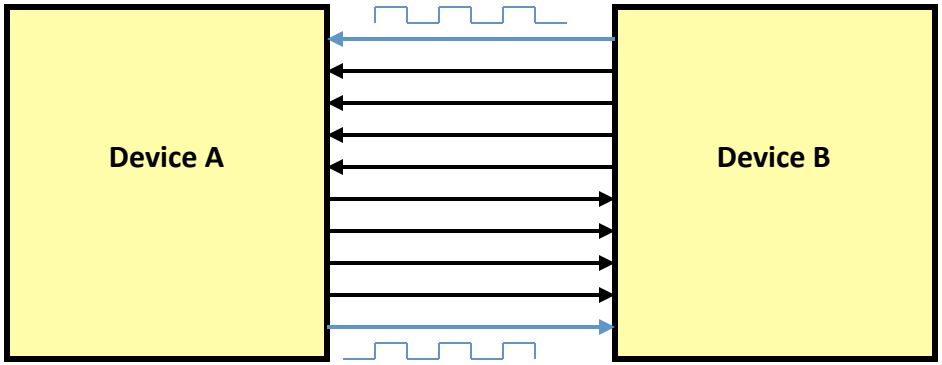
Traditional Parallel Bus

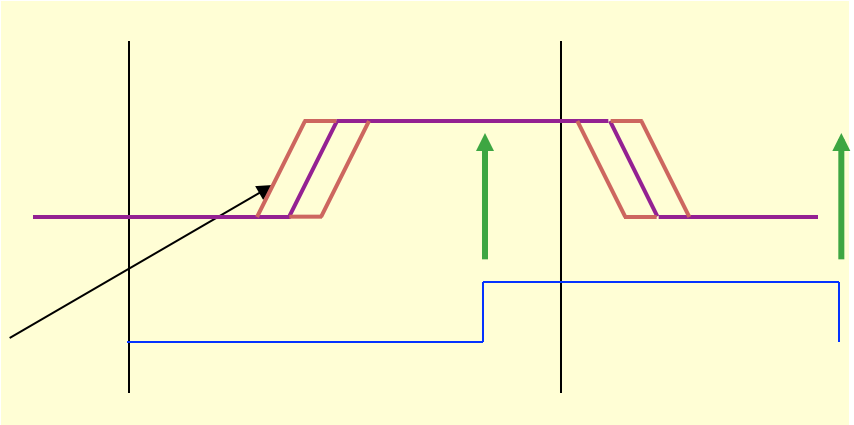


- Used in low to medium 100 MHz range
- Issues
  - Board trace length mismatch effect on skew
  - Clock skew across devices
  - Faster data rate squeezes “eye”



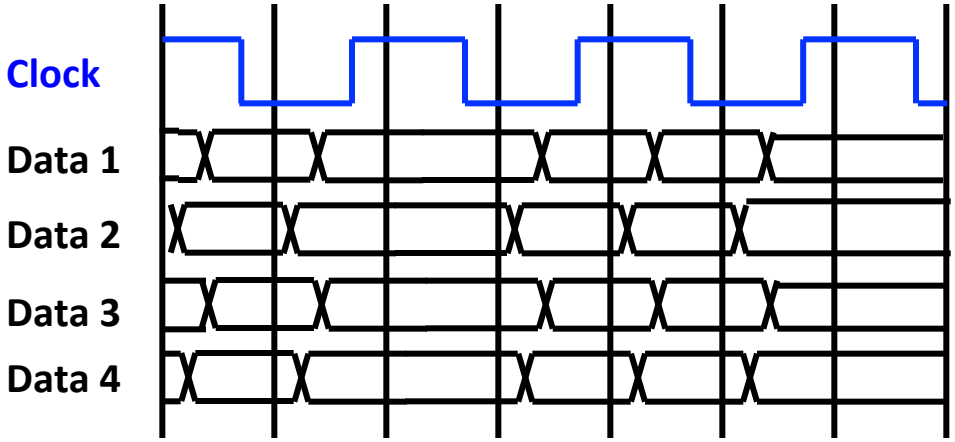
Source Synchronous Bus





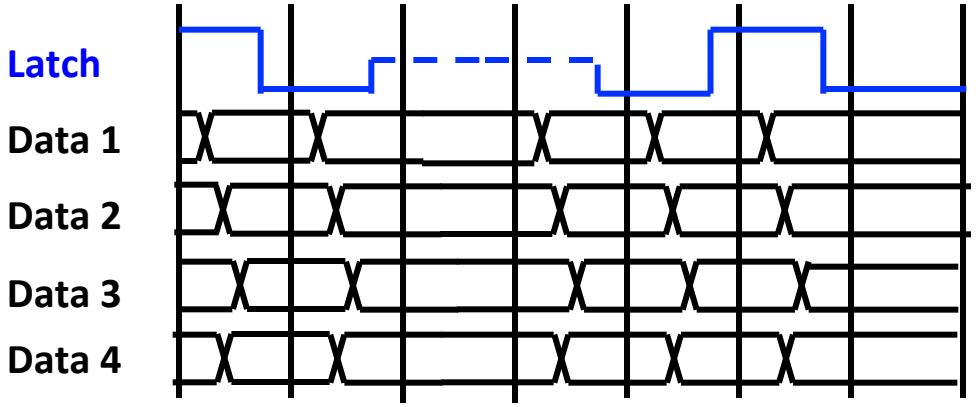
- Used in 200MHz to 1.6GHz range
- Clock signal is “forwarded” with data
- Design impact:
  - Board layout track length mismatch still adds to skew
  - Eliminates skew error term caused by clock domain skew
  - Allows faster cycle times than parallel

Source Synchronous - Clock Forward



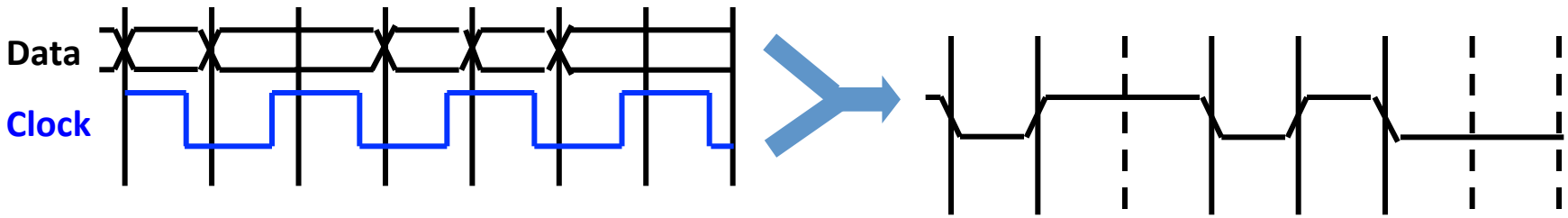
- Clock is transmitted continuously from Tx to Rx
- Removes clock distribution skew
- Examples
  - HyperTransport
  - Parallel RapidIO

Source Synchronous - Latch Forward



- Latch assertion is transmitted when valid data is present
- Examples
  - Bi-directional buses
  - Memory Interfacing (e.g. DDR)

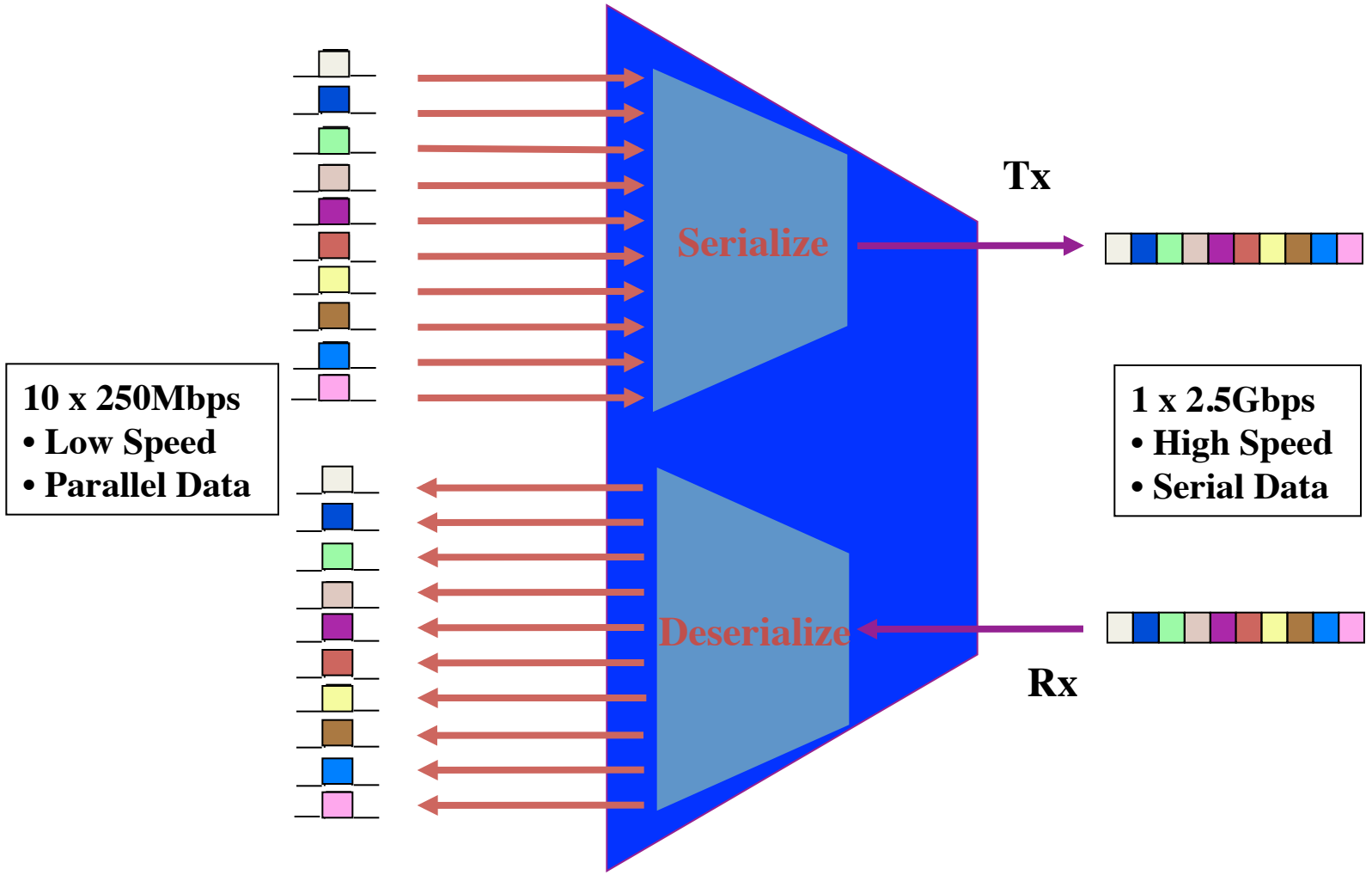
Embedded Clock



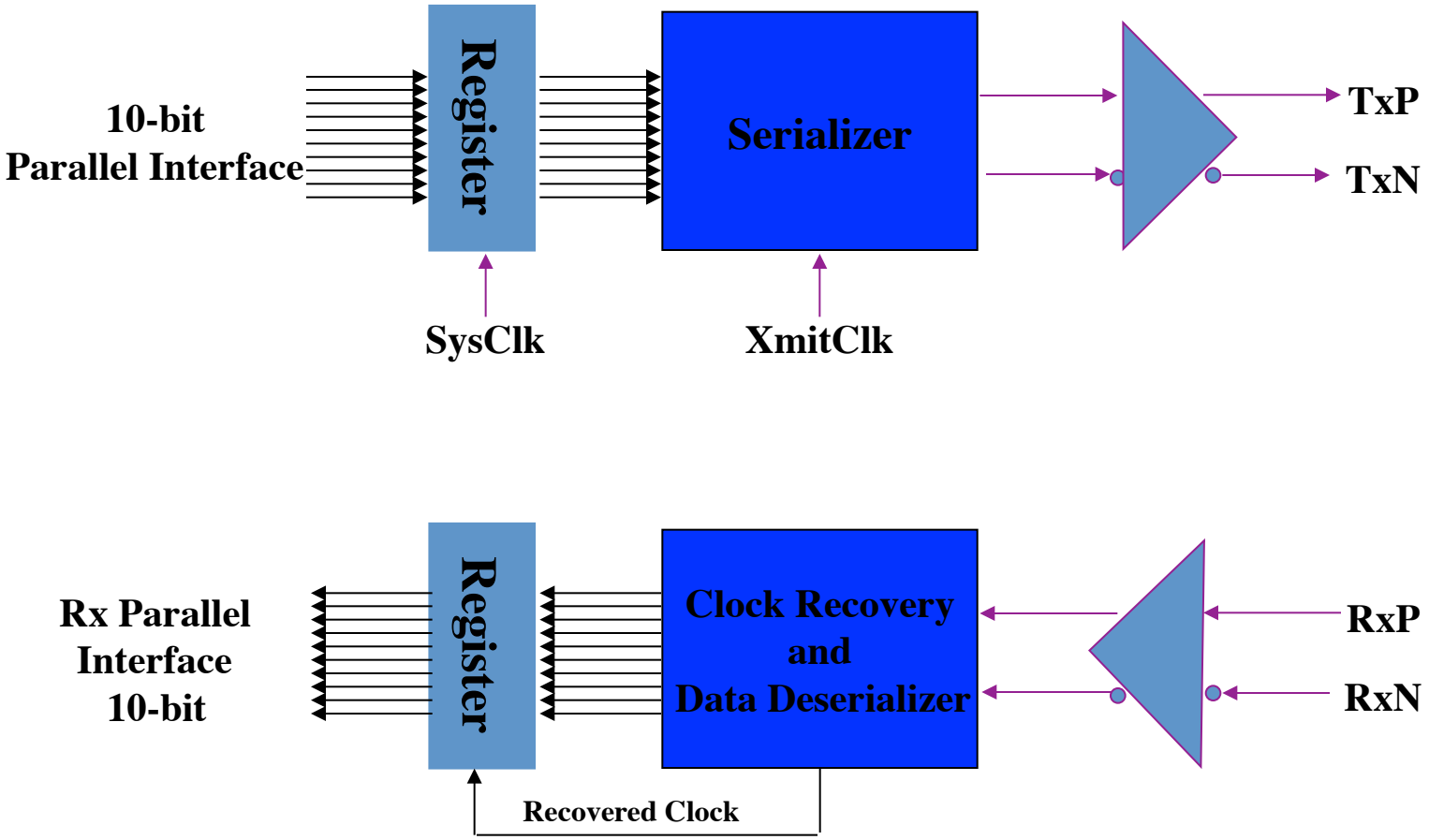
Clock signal embedded with data

- Clock signal is embedded with data
  - Edge density guaranteed by *encoding* scheme
- Examples
  - PCI Express
  - USB
  - Serial RapidIO
  - Infiniband

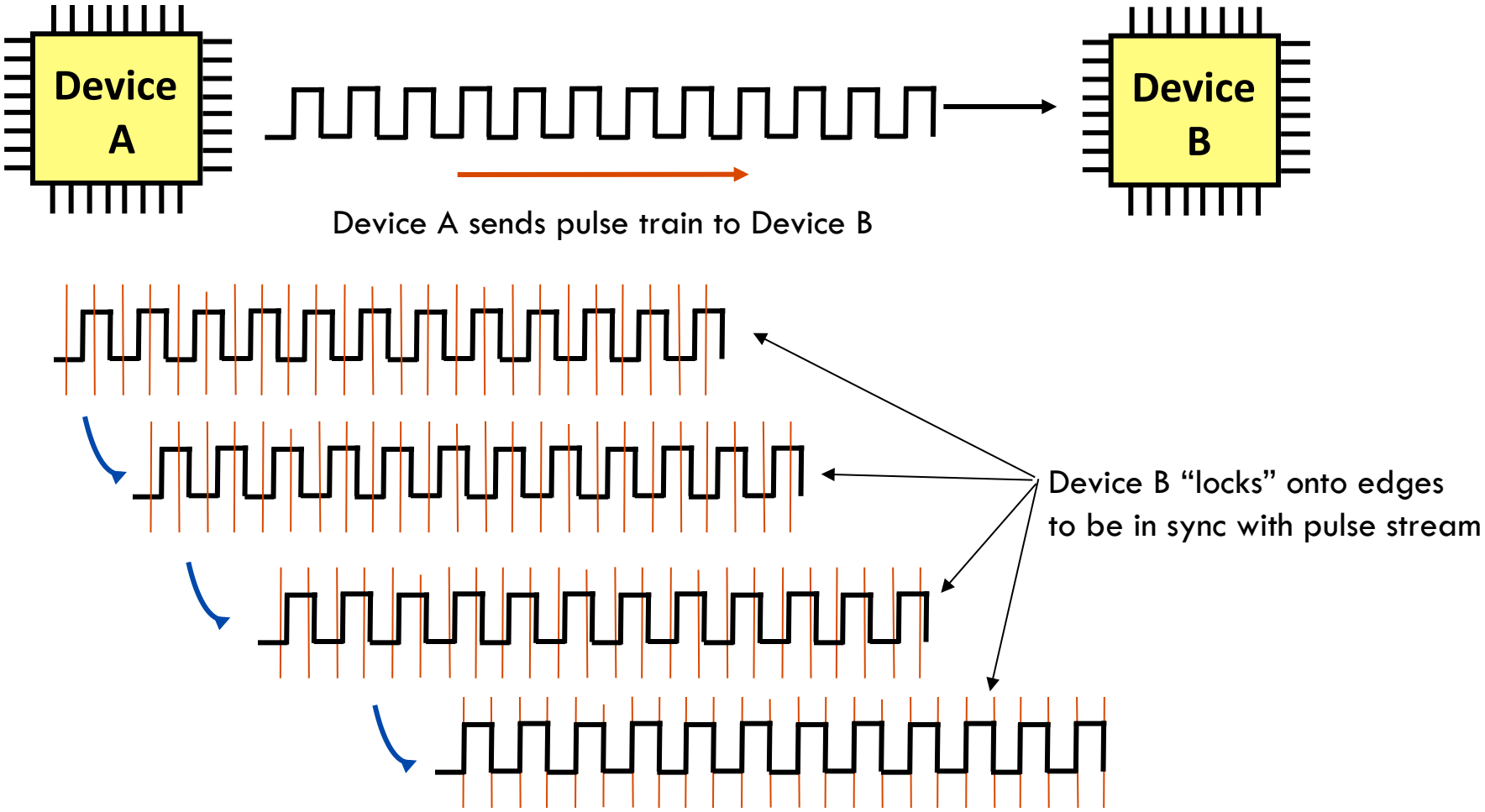
SerDes (Serial/Deserializer)



SerDes - Parallel and Serial Conversion



Edge Lock Technique (Tracking Receiver)



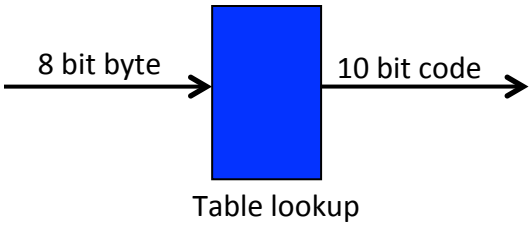
Ensuring Edge Density: m-of-n codes

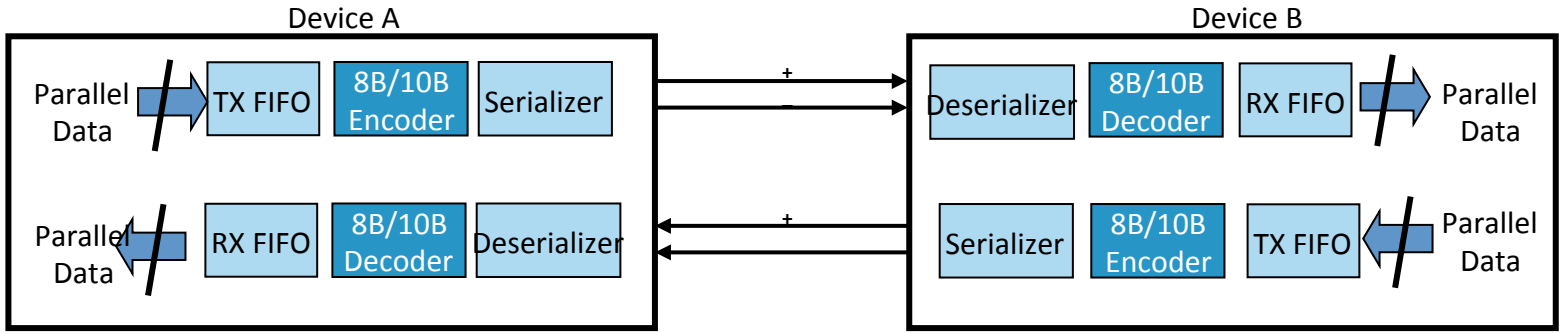
- Some 8-bit code words have too few 1s (or 0s) to ensure edge density sufficient to recover clock
- 8b10 encoding (developed by IBM in 1983)
  - Use subset of 2^{10} 10-bit code words having a "balanced" number of 0s and 1s
  - Benefits
    - Ensure edge density
    - Avoid DC bias at receiver from imbalance
      - Running disparity for unbalanced codewords
  - 256 data characters
    - All 8-bit bytes
    - 12 control characters (INIT, etc)

$$\binom{n}{m} = \frac{n!}{m! \times (n-m)!}$$

$$\binom{10}{5} = \frac{10!}{5! \times 5!} = 252$$

$$\binom{10}{4} = \binom{10}{6} = 210$$





8b/10b Transmission Code

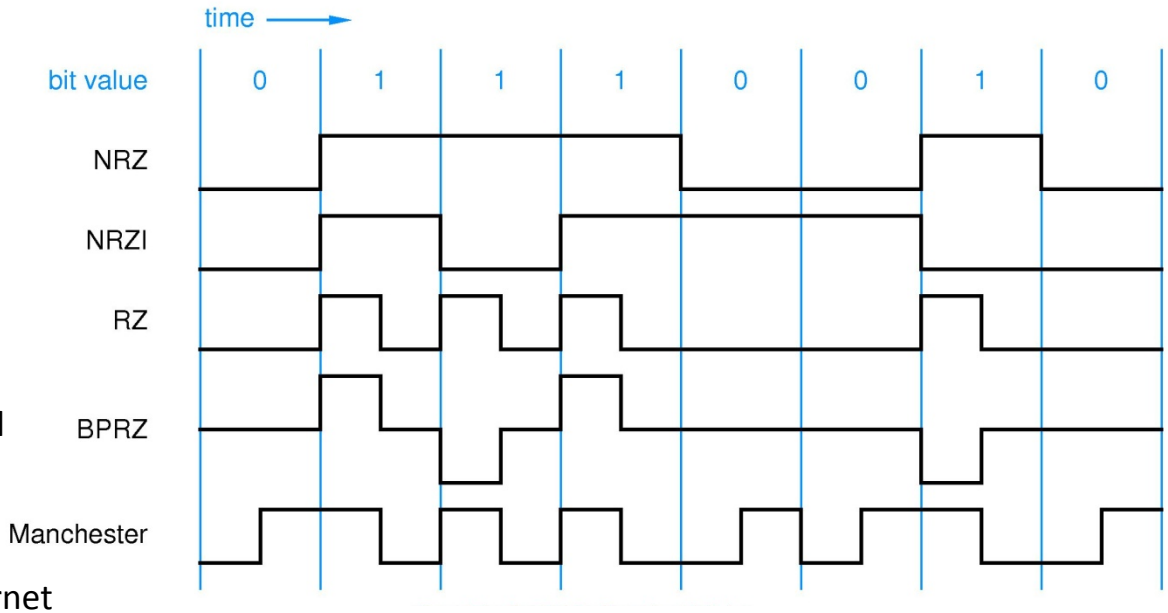
Name	Hex Value	8-bit binary	10-bit Encode
D0.0	00	00000000	1001110100
D1.0	01	00000001	0111010100
D2.0	10	00000010	1011010100
D3.0	11	00000011	1100011011

- Reasons for using 8B/10B encoding/decoding
  - Guarantees transition density to ensure correct PLL operation
  - Error correction to detect signaling errors
  - Ensures signal is DC balanced – no DC offset develops over time
  - Support of special characters that can be used as delimiters for control, such as sync or framing, or other generalized commands

Codes for Serial Data Transmission

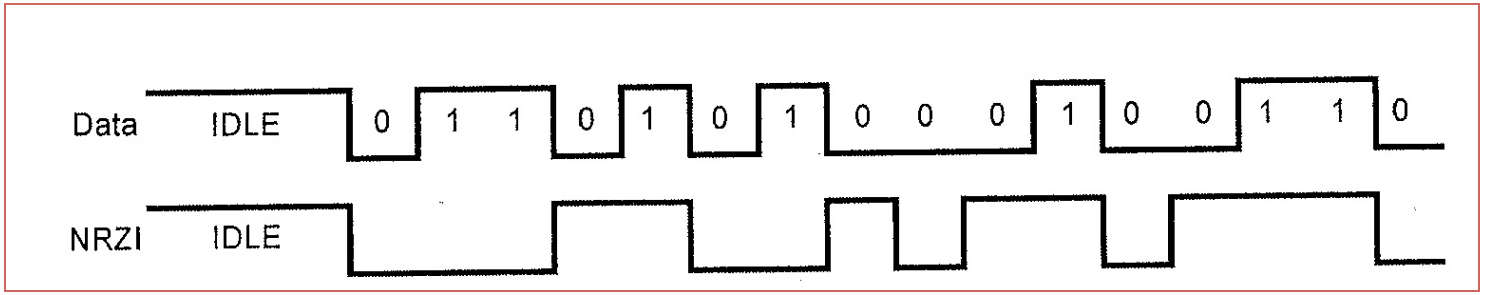
- Objectives/Properties of Line Codes
  - Facilitate clock recovery by providing edge density
  - Avoid DC bias by being DC balanced
  - Noise immunity (if used with differential pairs)

- NRZ – Non Return to Zero
- NRZI – Non Return to Zero Invert (on ones)
  - Transition based
  - Differential signaling: USB
- RZ – Return to Zero
- BPRZ – Bipolar Return to Zero
  - "DC balanced"
  - MLT-3 (100 Base T Ethernet)
- Manchester encoding
  - Guarantees transition every bit cell
  - Facilitates clock recovery
  - Requires higher bandwidth
  - Original coax-based 10 Mbps Ethernet
- Other techniques for DC balancing (and edge density)
  - m-out-of-n-codes (e.g. 8B10B): Gigbabit Ethernet



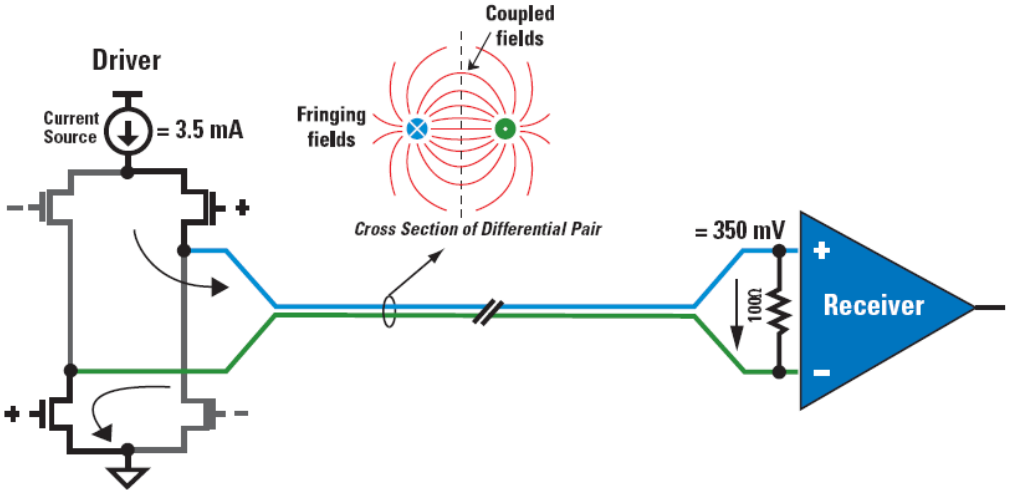
NRZI Encoding

- Ensures data integrity
  - Better than NRZ in transition-based communication
    - Edge-based rather than level-based (LVDS)
  - Edge density for PLL (phase locked loop)
    - Clock extraction
- 0 toggles output level
- 1 holds output level
- "Bit stuffing" (alternative to 8b/10b encoding)
  - after six consecutive 1s a 0 is inserted in the data stream
  - discarded by receiver

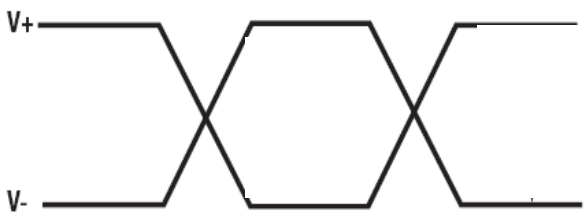


Differential Signaling

- Differential, point to point
  - Complementary signals transmitted
  - Receiver detects voltage difference between lines
  - Low amplitudes (200mV - 400mV typical), high speeds
  - Good noise immunity
    - Pair routed together – noise cancels out
- LVDS – Low Voltage Differential Signaling
  - ANSI/TIA/EIA 644-1995 standard (signaling only, not protocol or connectors)
  - 3.125 Gbps, +/- 350mV
  - "Gbps at mWs" -- High speed & low power consumption
  - FibreChannel, Gigabit Ethernet, HDMI, DVI



May be edge-based!



USB

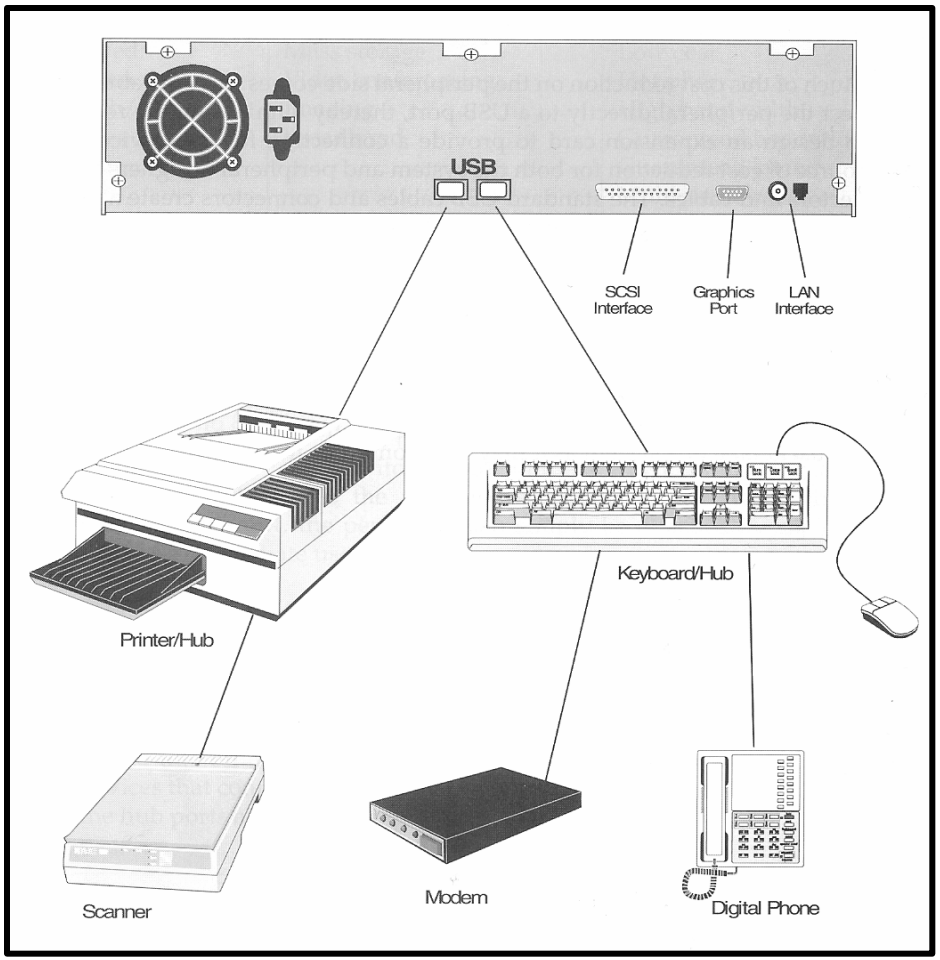
- Universal Serial Bus
  - Need for new peripheral bus
    - Inexpensive
    - Capable of supporting wide range of devices
    - Simple (no IRQ contention, etc)
    - "Hot-pluggable" (plug and play)
  - USB introduced 1995
    - Compaq, Digital, IBM, Intel, Microsoft, NEC, Northern Telecom, later 25 companies
  - USB 2.0 Finalized April 2000
    - Fully backward compatible → implications!

USB 1.X	LS	Low Speed	1.5 Mbps
	FS	Full Speed	12 Mbps
USB 2.0	HS	High Speed	480 Mbps

Low speed devices: keyboard, mice

Higher speed devices: scanners disks, CD/RW, flash storage

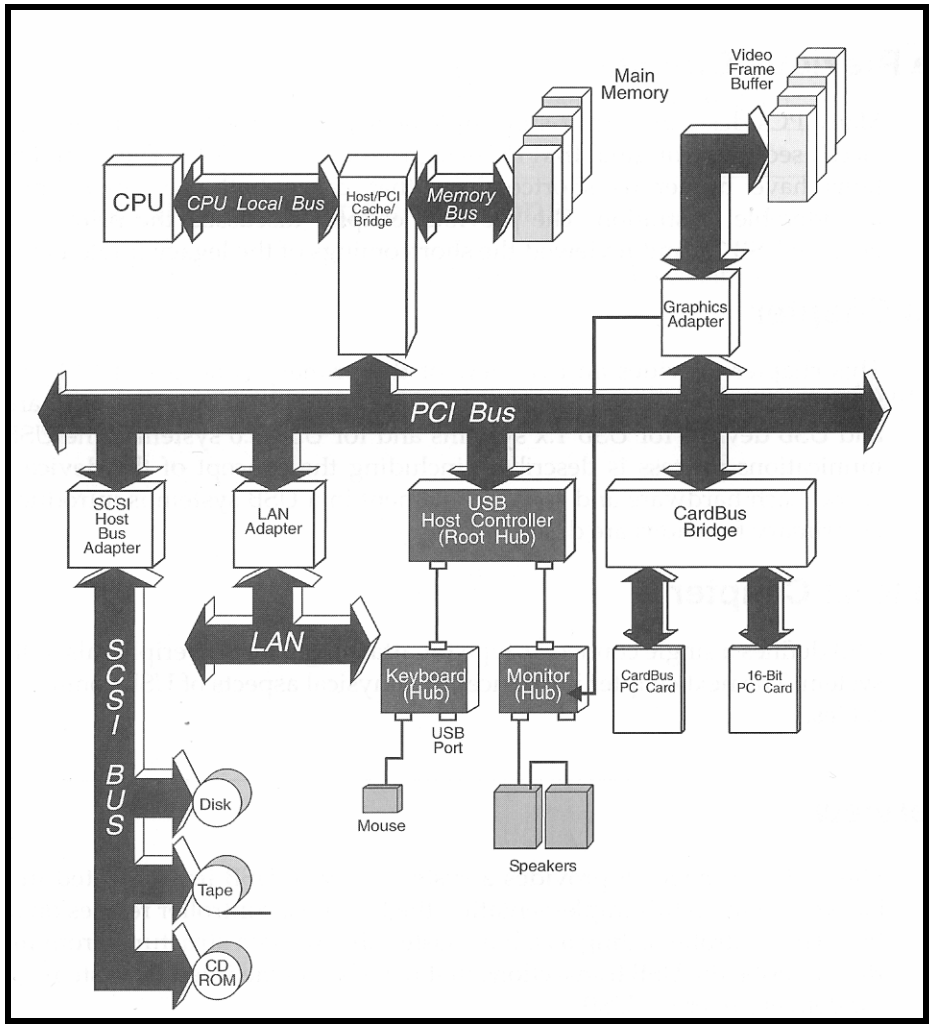
USB – Flexible Peripheral Interface



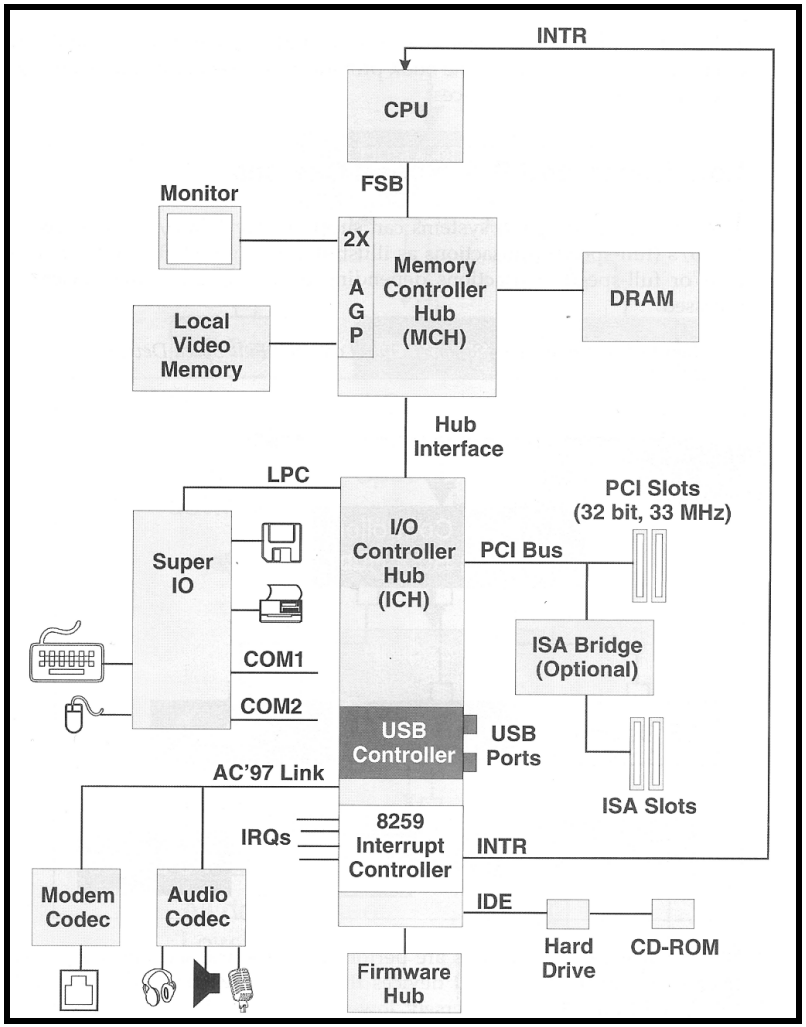
- Tree Topology
  - Point-to-point links
  - Hubs (7 peripherals/hub)
  - 127 devices maximum
  - 5m maximum cable length
- Serial Interface
  - LSB first
- Low power devices can be powered through cable
  - 100mA
  - 500mA if permitted by host
- Isochronous
  - Device can request guaranteed bandwidth
    - "Mission Critical" applications
    - Audio, streaming video

USB Interfaces

Through PCI Bus

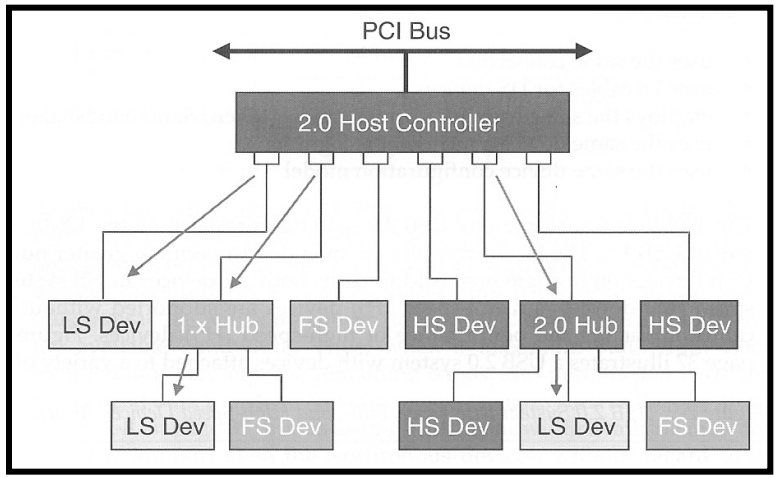


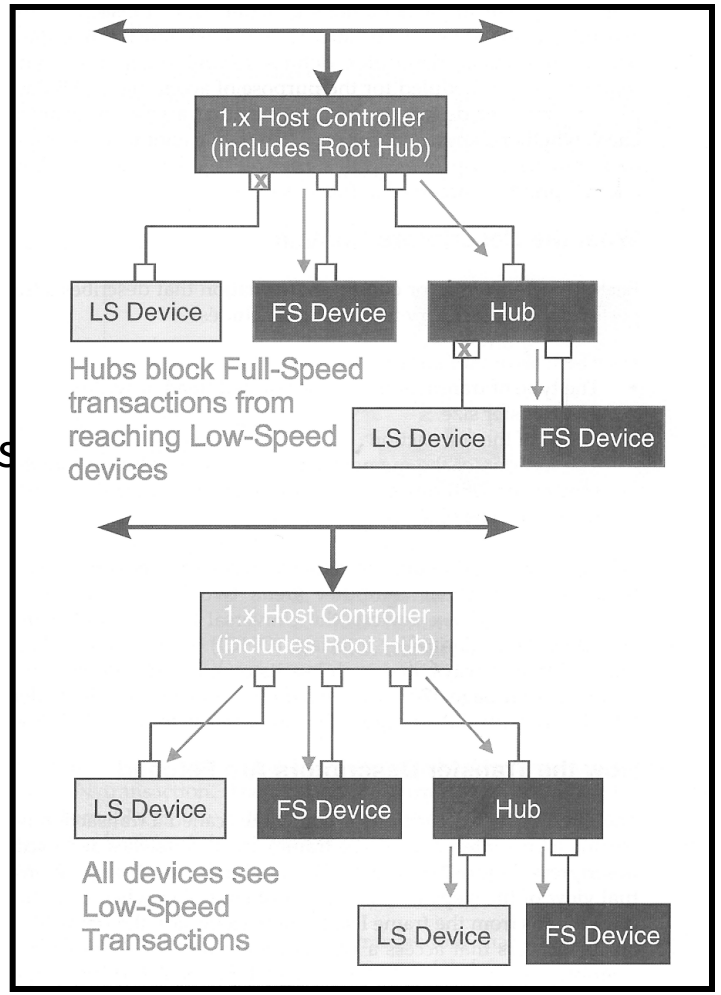
Integrated into I/O Controller Hub (SouthBridge)



Tree Topology

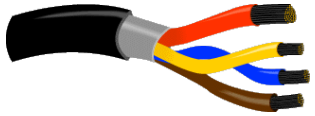
- Host Controller
  - Generates transactions
- Hubs
  - Controls power to its ports
  - Enables/disables ports
  - Recognizes devices attached ports
  - Reports status events associated with ports (when requested by host)
- Devices





USB 2.0 Connectors

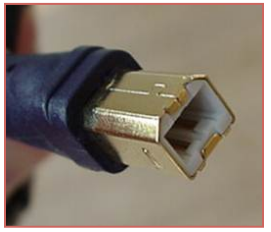
- Standard A connectors
  - USB cable to USB port
  - Most common
  - Cable attached to peripheral at other end



Contact Number	Signal Name	Cable Color
1	Vcc/VBus	Red (Orange)
2	-Data	White (Yellow)
3	+Data	Green (Grey)
4	Ground	Black (Blue, Brown)

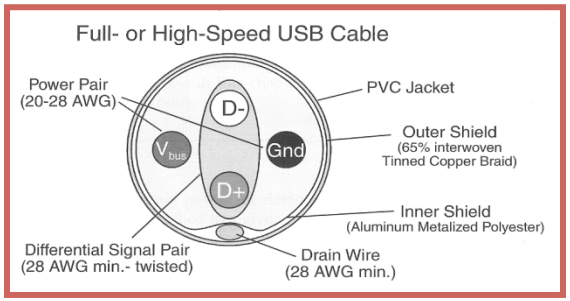
- Standard B connectors
  - USB cable to peripheral
  - Used with detachable cables
- Mini connectors (A and B)
  - 5th wire: ID signal (Gnd/NC for A/B)
  - No longer being developed
- Micro connectors (A and B)
  - Replace mini connectors
  - OTG (On The Go) devices
    - Micro A-B receptacle accepts micro-A or micro-B plugs
    - Devices senses ID, behaves as host or peripheral
- Power contacts (1mm) longer than differential data contacts
  - Ensure power prior to data contacts mating

"A" connection upstream to PC



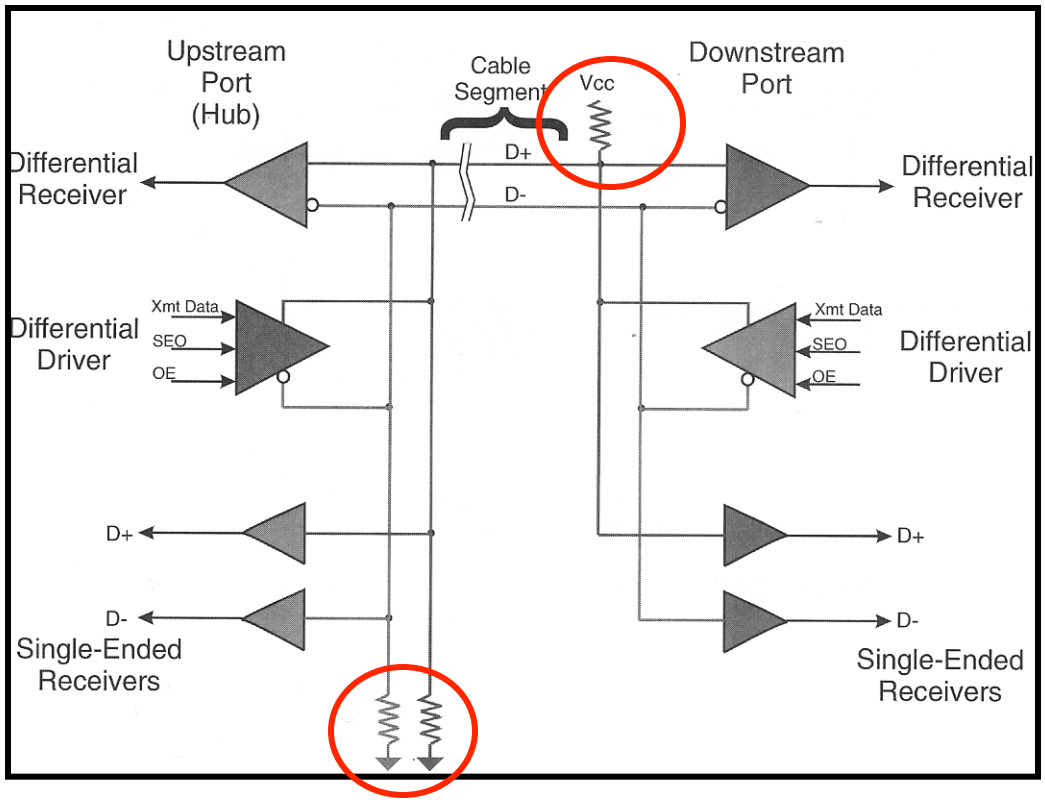
"B" connection downstream to peripheral

"Mini B" connection downstream to peripheral



USB Signaling

- Differential and Single-Ended Receivers

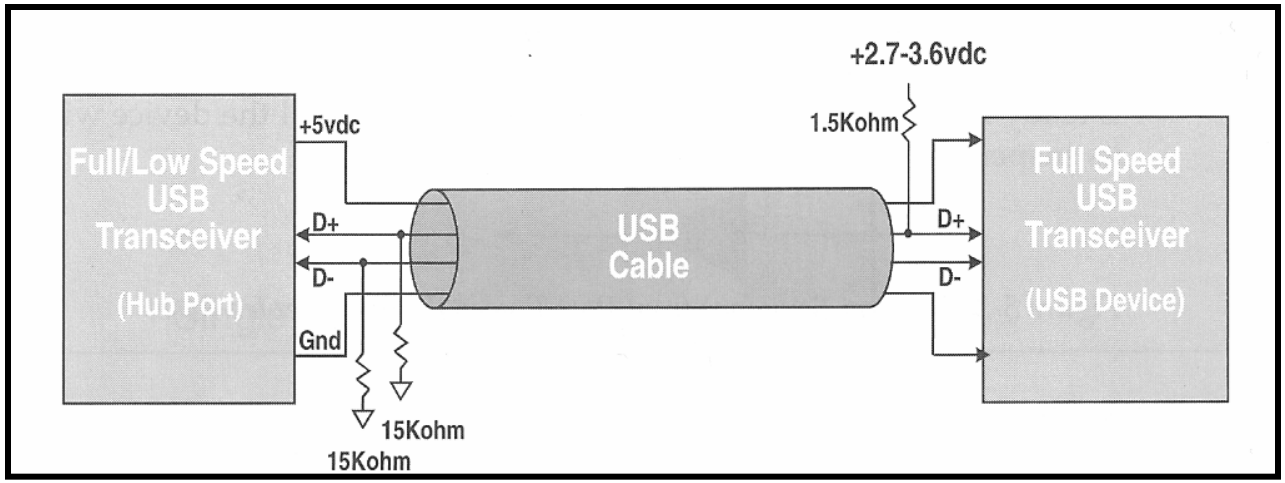


When no device connected,
D+ and D- both low

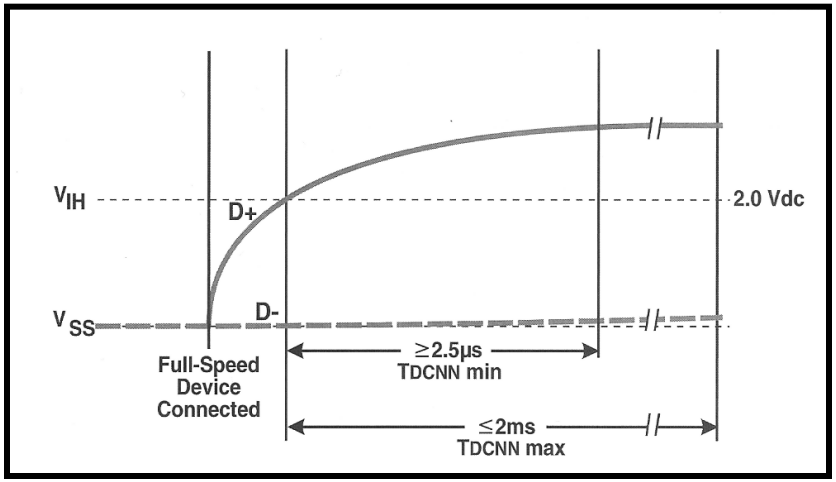
When device attached, pull-up
D+ high (FS device)

D- high (LS device)

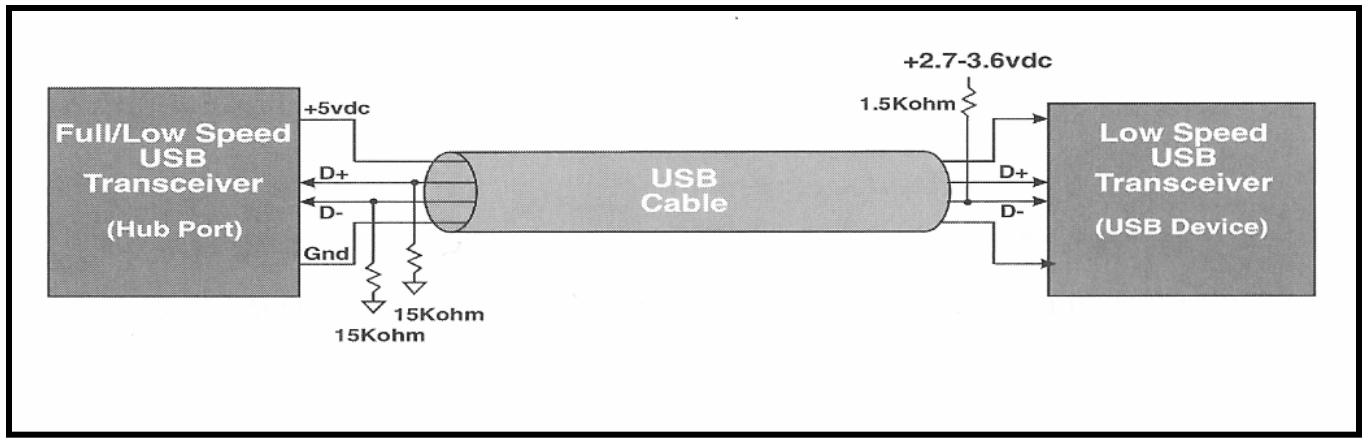
FS Device Connection/Detection



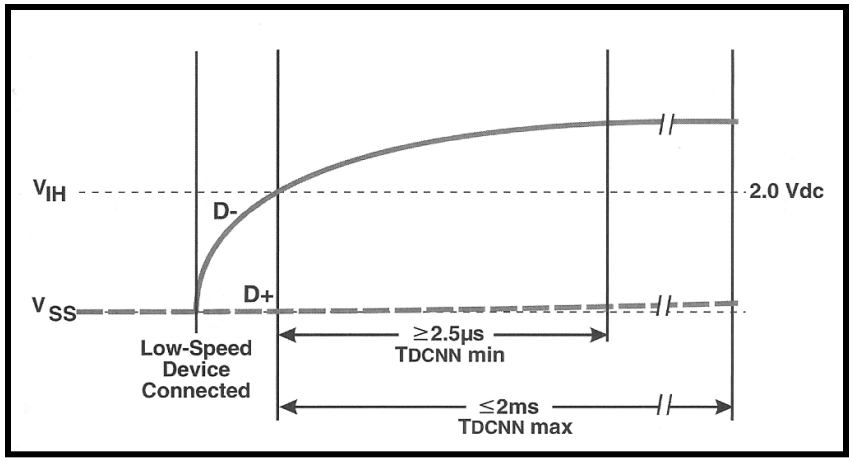
- For FS devices, D+ is pulled up
  - R for pulldown is 15KΩ
  - R for pullup is 1.5KΩ
  - V will be 90% of Vcc



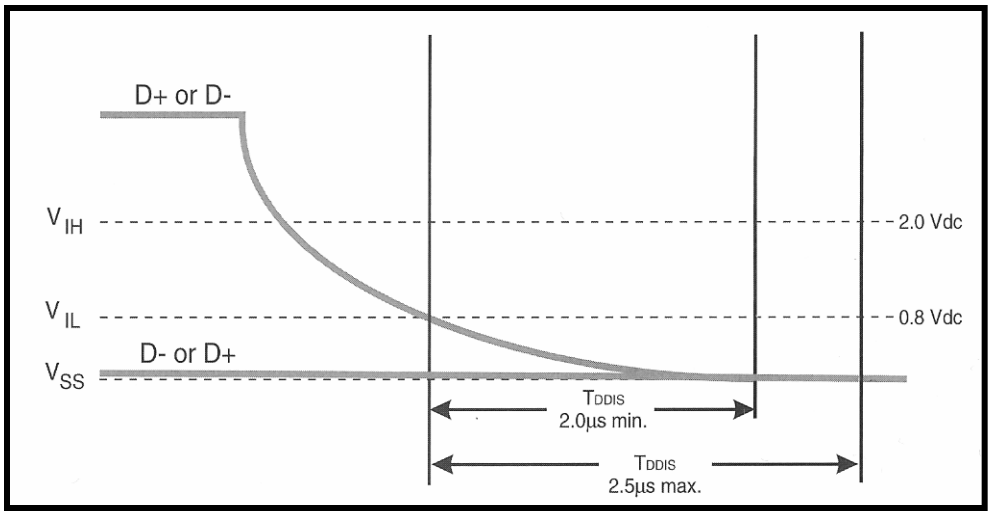
LS Device Connection/Detection



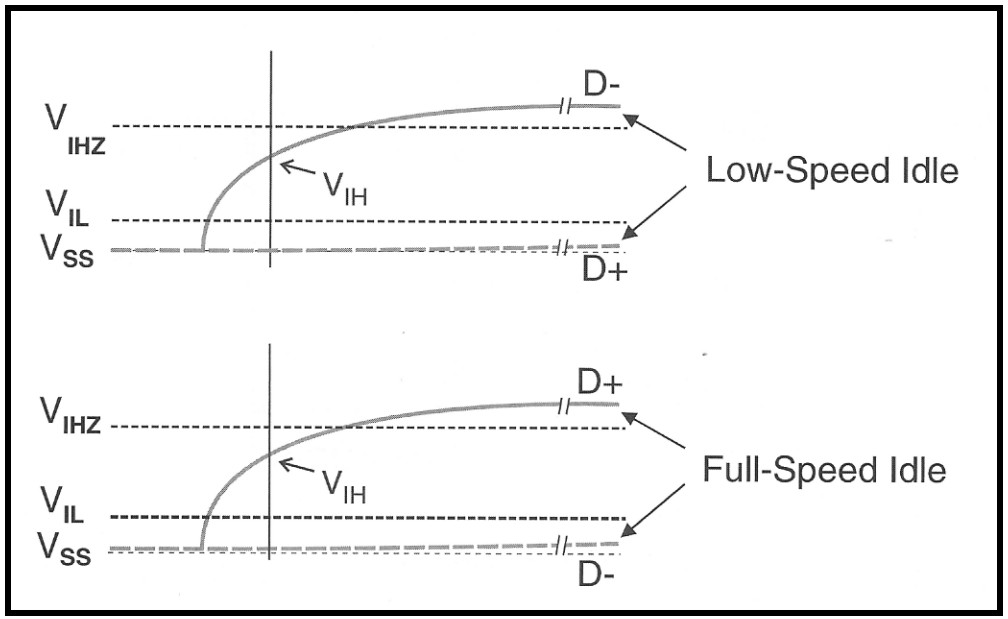
- For LS devices, D- is pulled up
  - R for pulldown is 15KΩ
  - R for pullup is 1.5KΩ
  - V will be 90% of Vcc



Device Disconnect

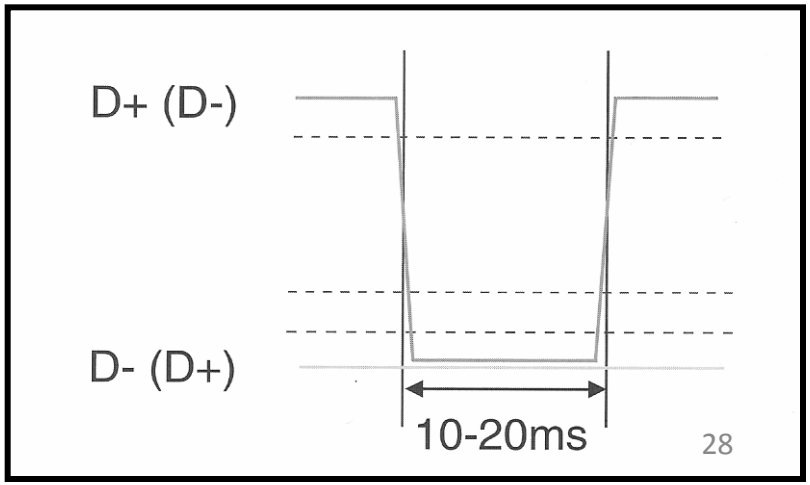


Bus Idle State (J state)

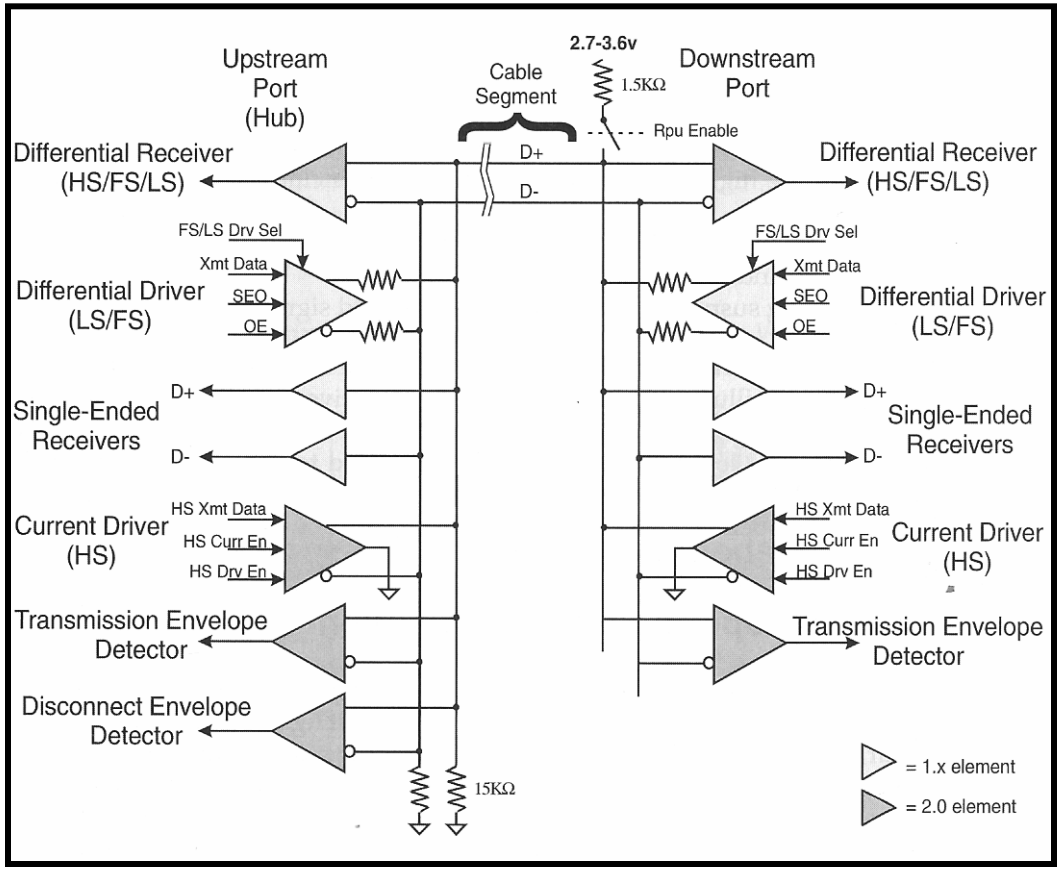


High Speed: Both D- and D+ low

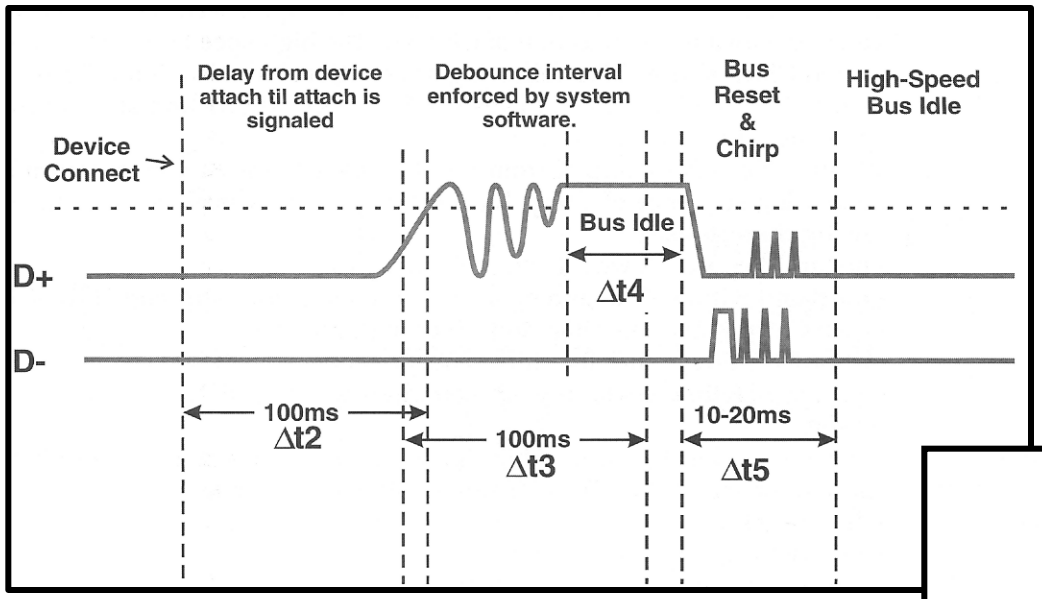
Device Reset



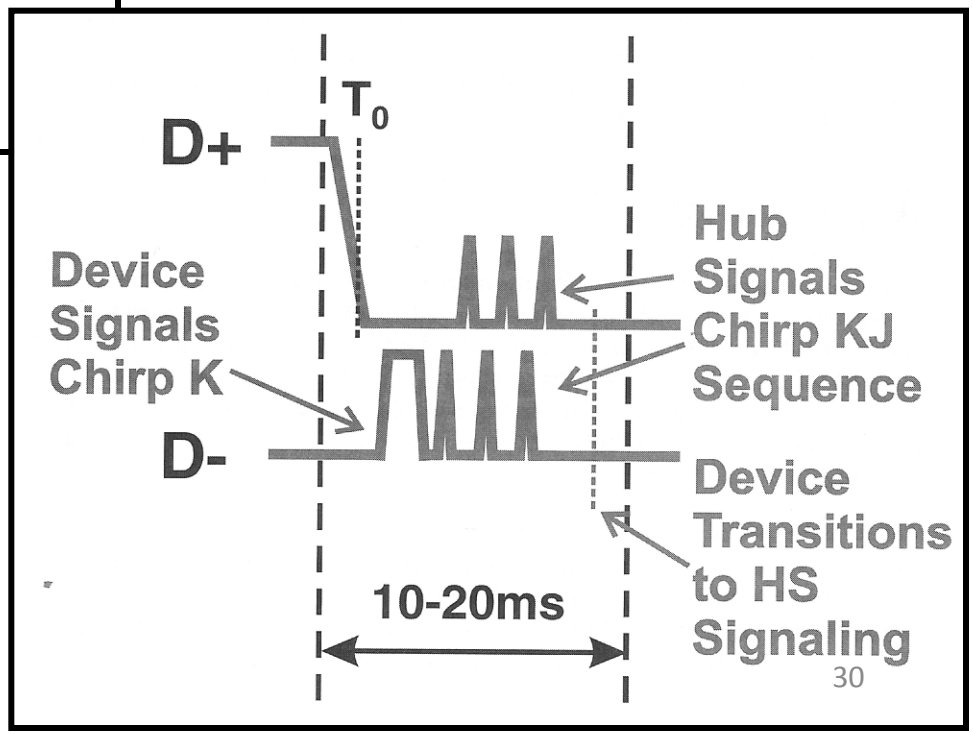
High Speed Devices



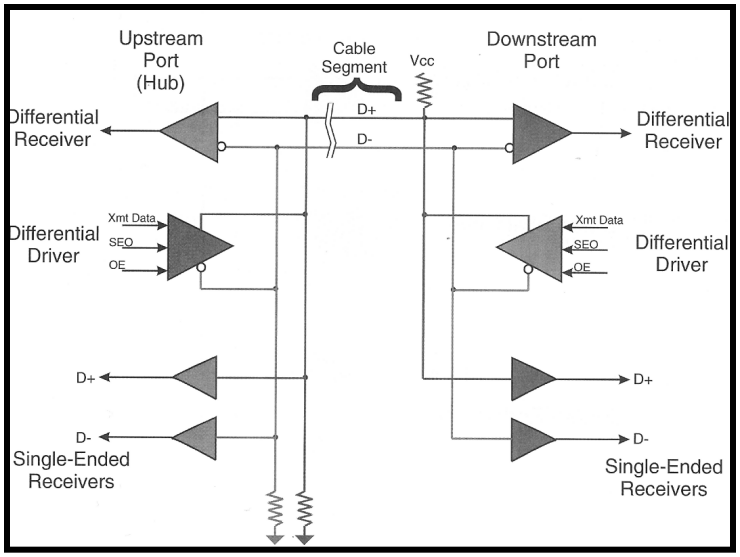
High Speed Device Connect

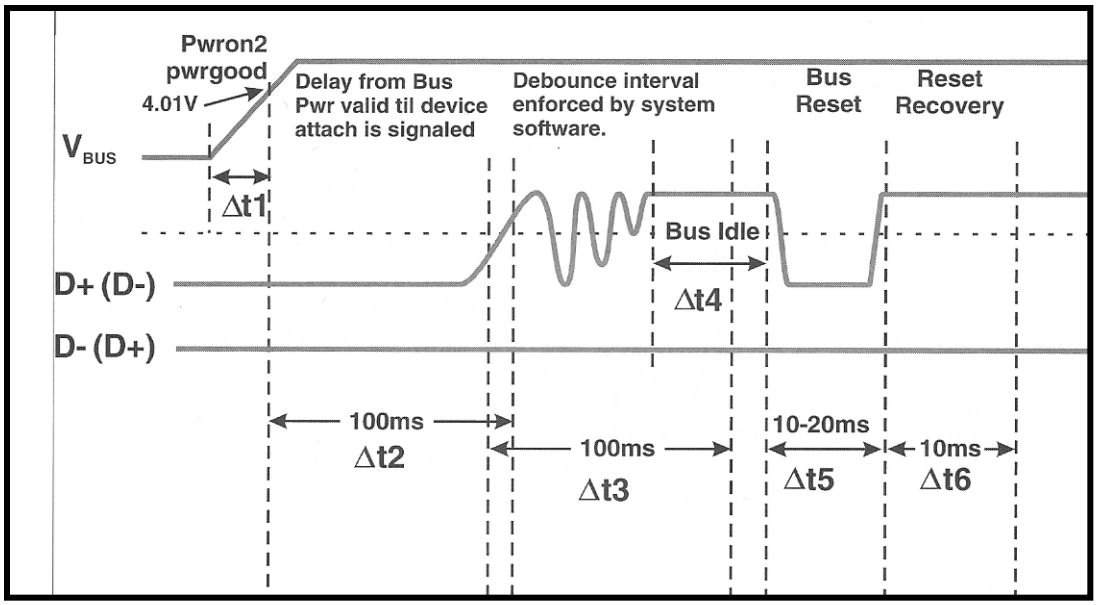


Initially connect as Full Speed devices
Then "chirp" K state awaiting KJ chirp response from hub if hub is HS capable

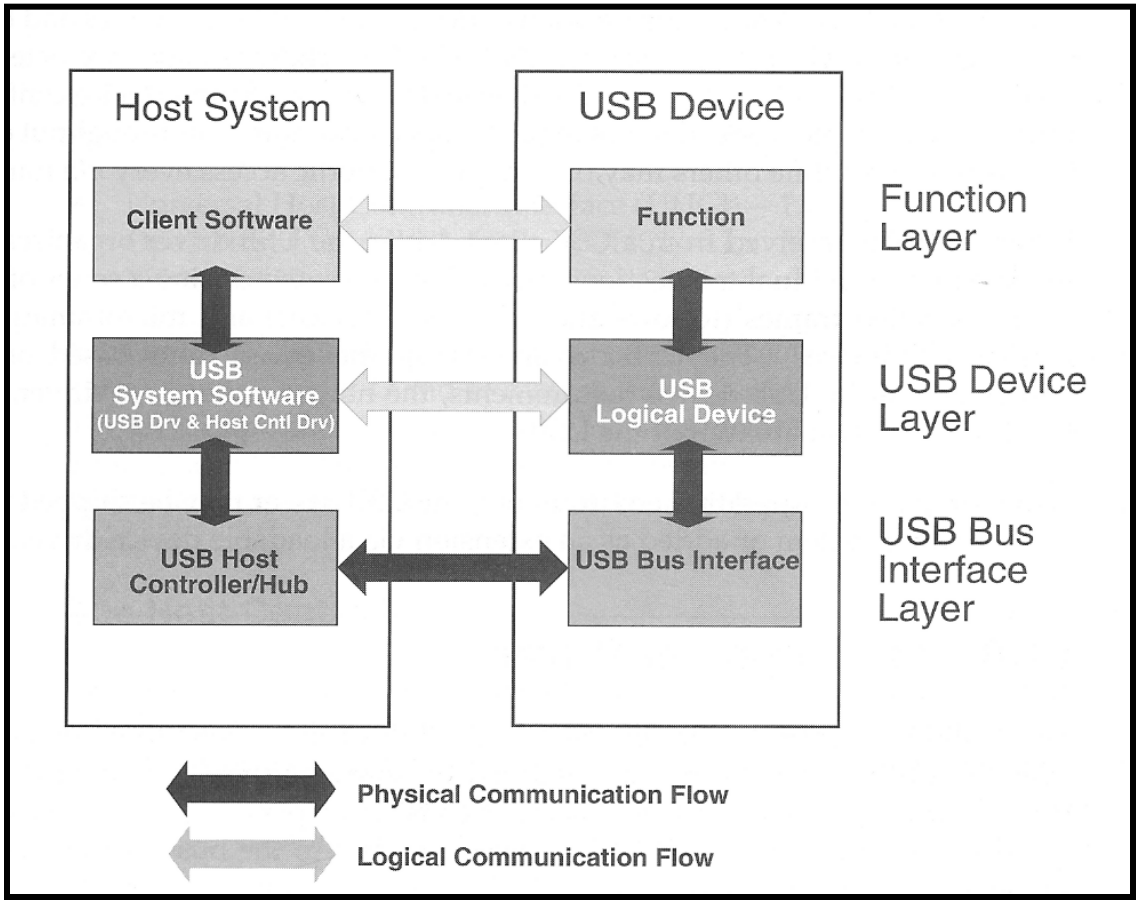


Power Up (Software Controlled)



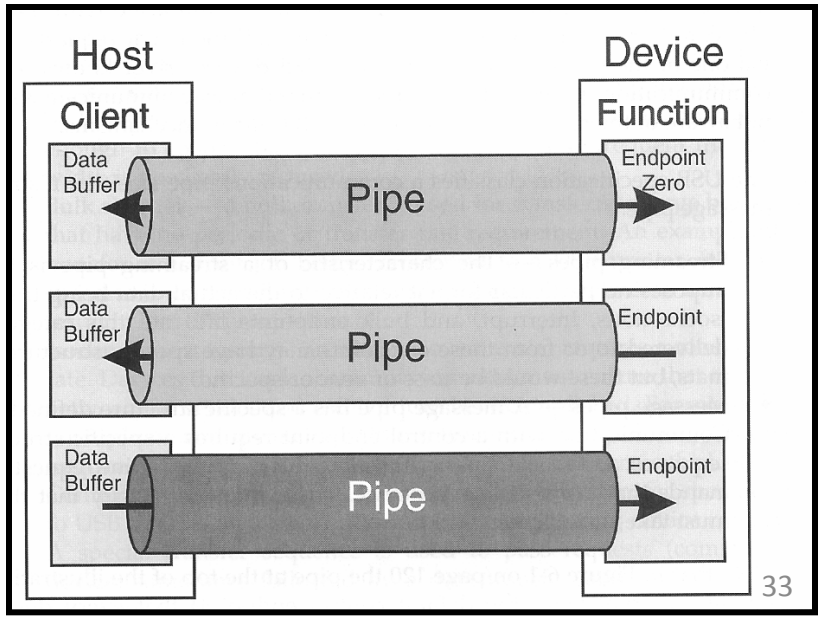


Layers and Logical Communication

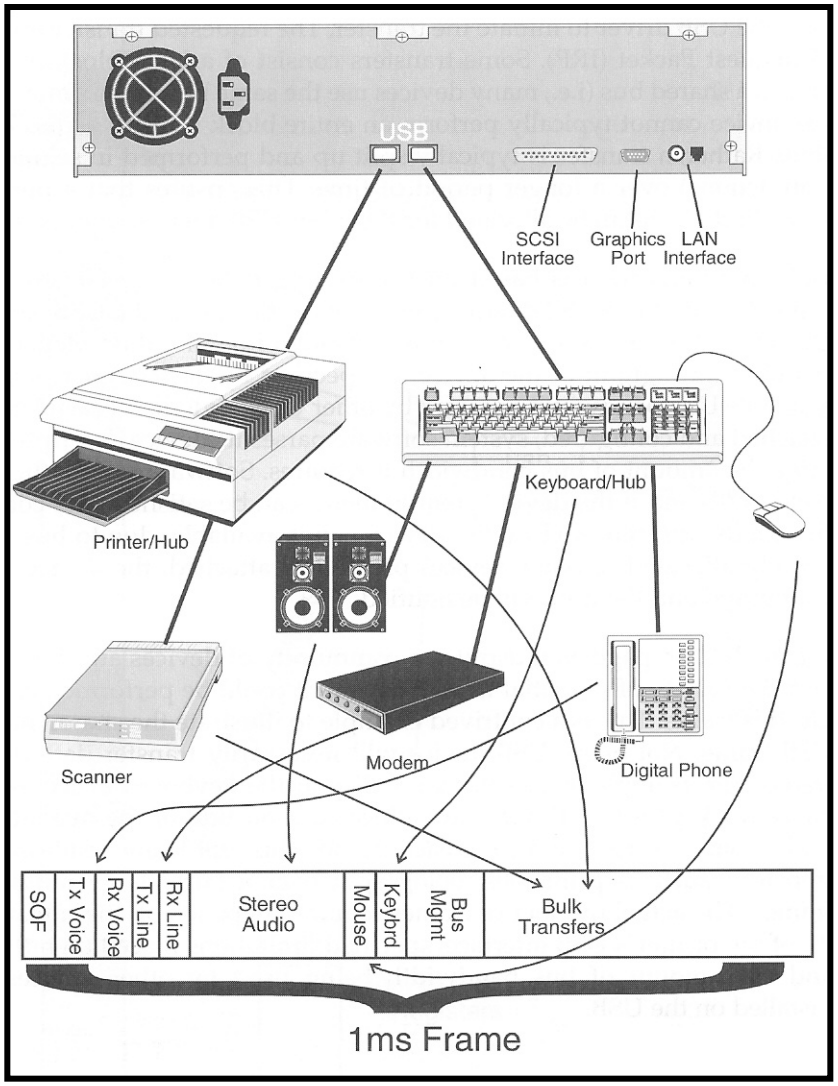


Endpoints

- Drivers communicate with device endpoints
- All devices must support endpoint 0 (control)
- Except for endpoint 0 all endpoints unidirectional
- Devices can implement multiple endpoints
- CD/ROM
  - Endpoint for control
  - Endpoint for writing data
  - Endpoint for reading data
  - Endpoint for reading audio
    - Isochronous



USB Devices Share “Frames”

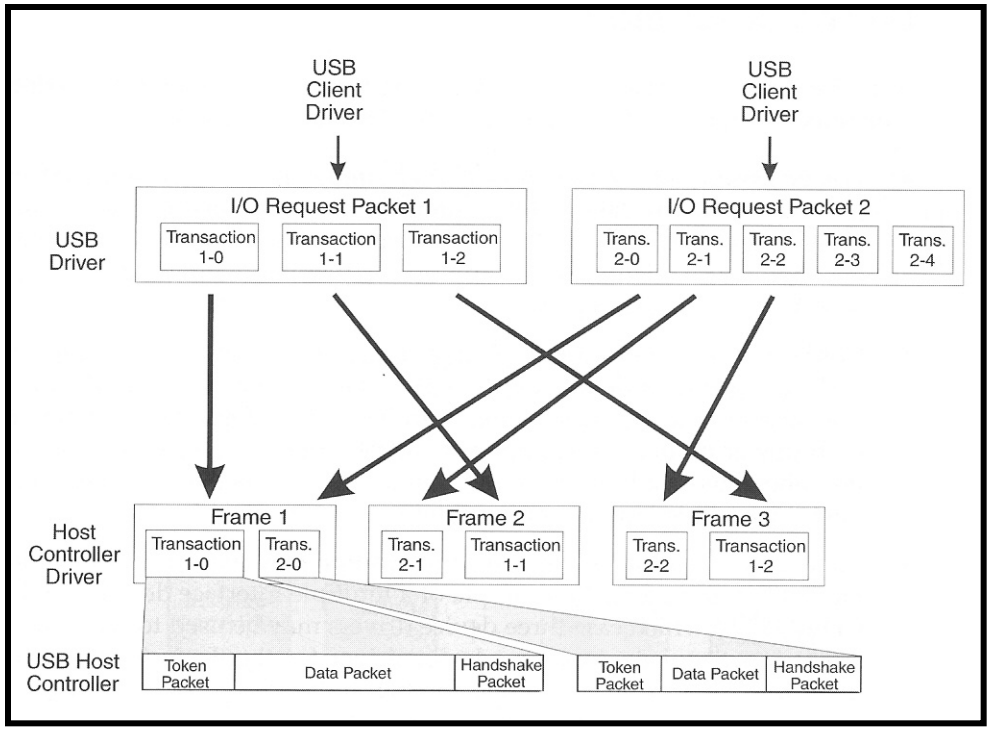


Transfer Types

- Control
  - Used to transfer requests to devices
  - Particular use for configuration
- Interrupt
  - For traditionally interrupt-driven devices (e.g. keyboard)
  - Actually implemented with polling
- Bulk
  - Large blocks of data without transfer rate requirements (e.g. printer)
- Isochronous
  - Data requiring constant delivery rate (e.g. audio, video-conferencing)
  - No error checking (occasional bad data better than no or late data)
  - During configuration devices indicate amount of guaranteed bandwidth needed

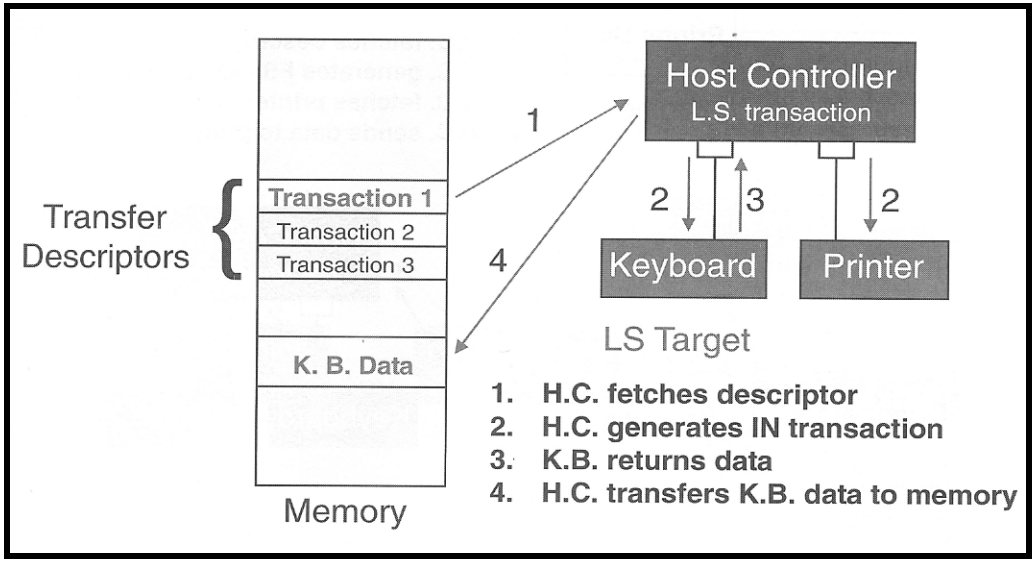
Transactions

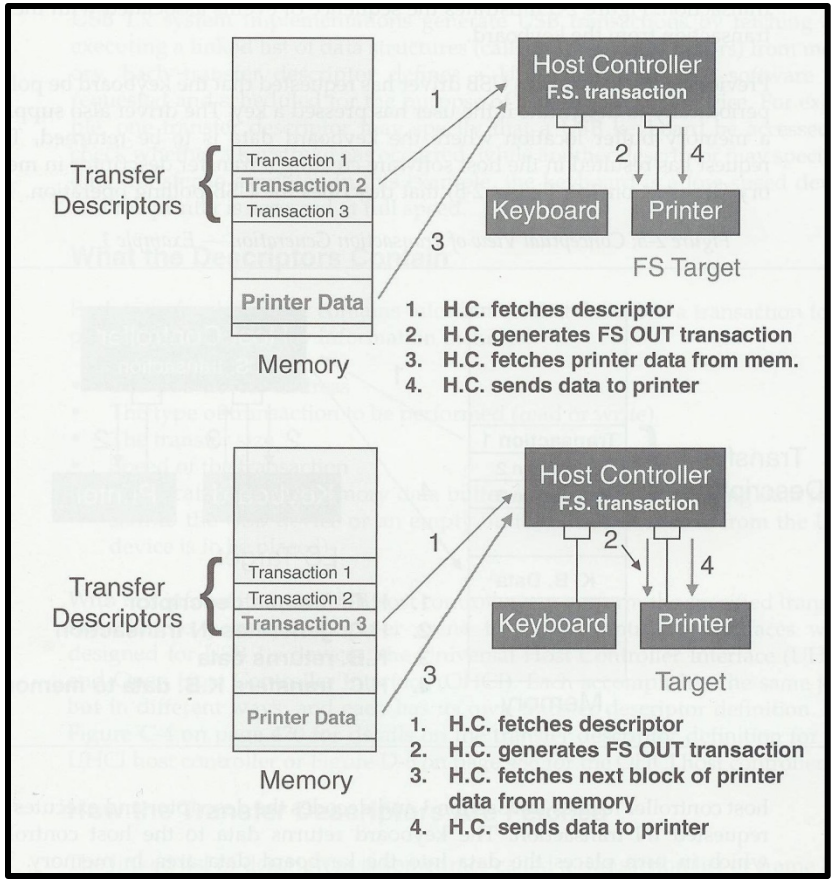
- Transaction consists of multiple packets
  - Token/Data/ACK
- Must complete in (micro) frame
- USB Client Driver
  - Initiates transfers
  - Using IRP (I/O Request Packet)
- USB Driver
  - Breaks IRPs into transactions
  - Achievable in single frame
  - Transaction descriptors created
- Host Controller Driver
  - Schedules transaction descriptors
  - Using knowledge of devices
  - Allocates to frames
- USB Host Controller
  - Traverses transaction descriptors
  - Serializes data
  - Creates USB packets



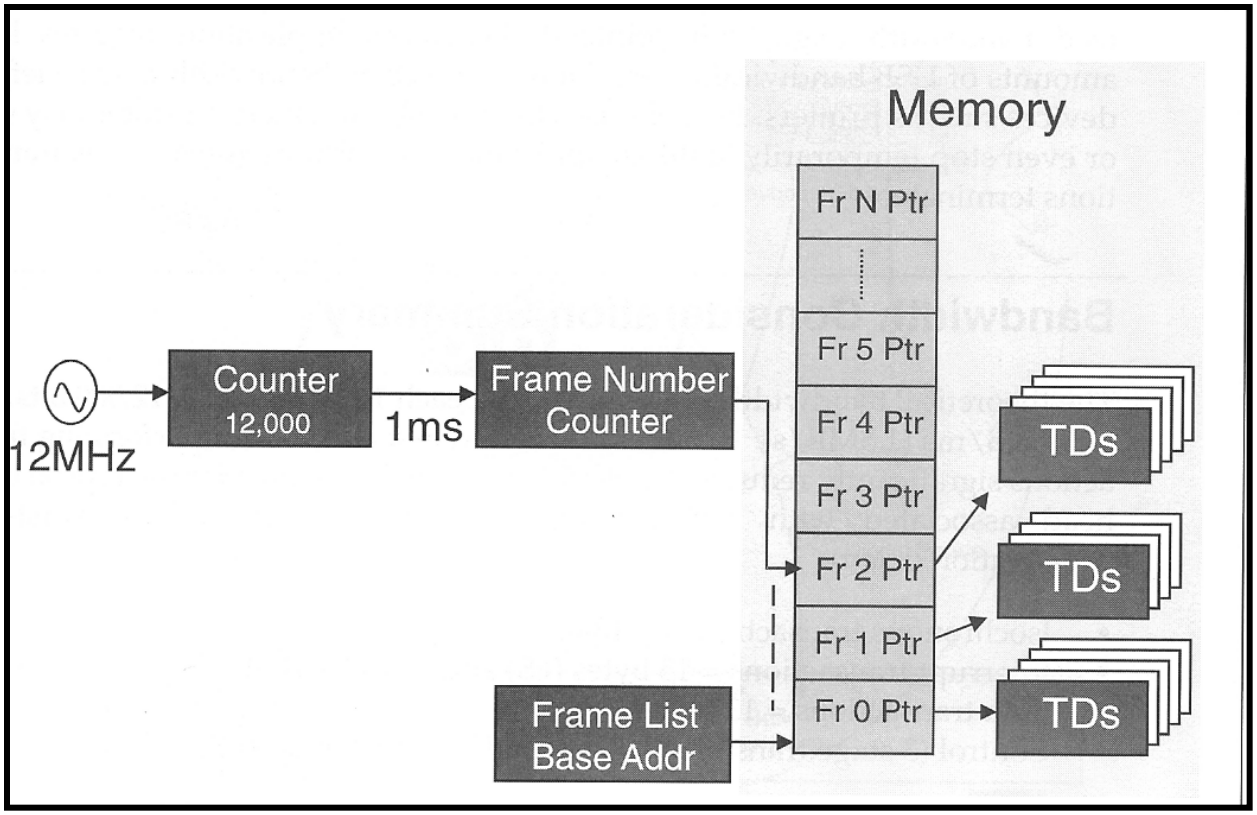
Transfer Descriptors

- USB Device Address
- Type of Transfer
- Direction of Transfer
- Speed of Device
- Address of Device Driver's Buffer

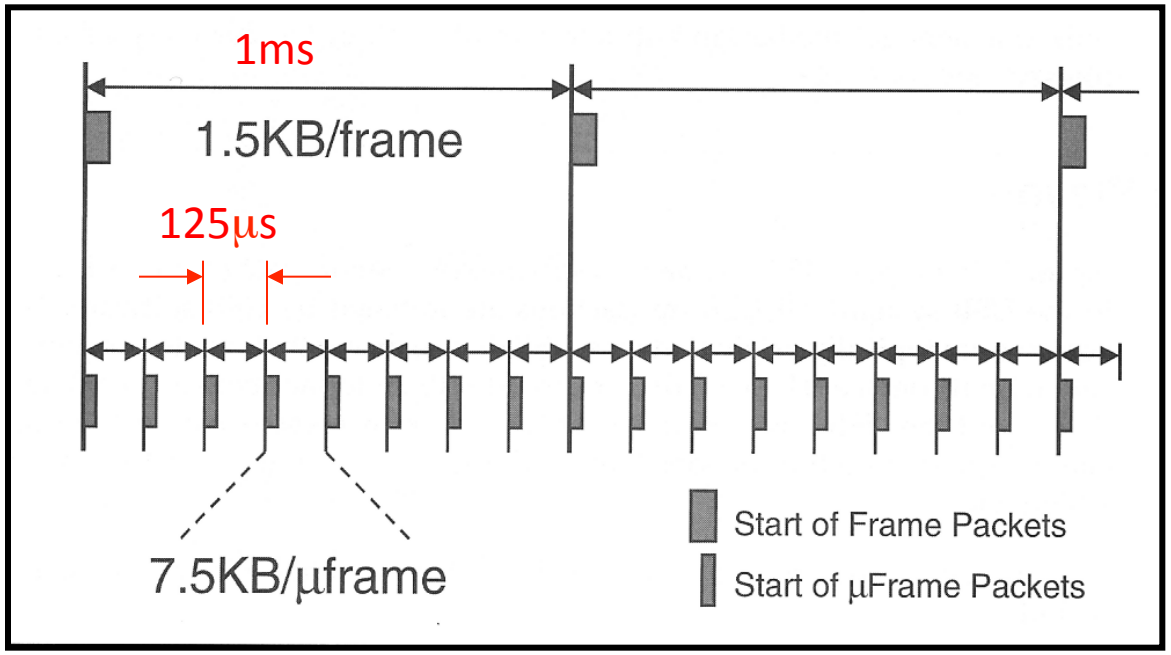




Frame Generation



Frames and μFrames

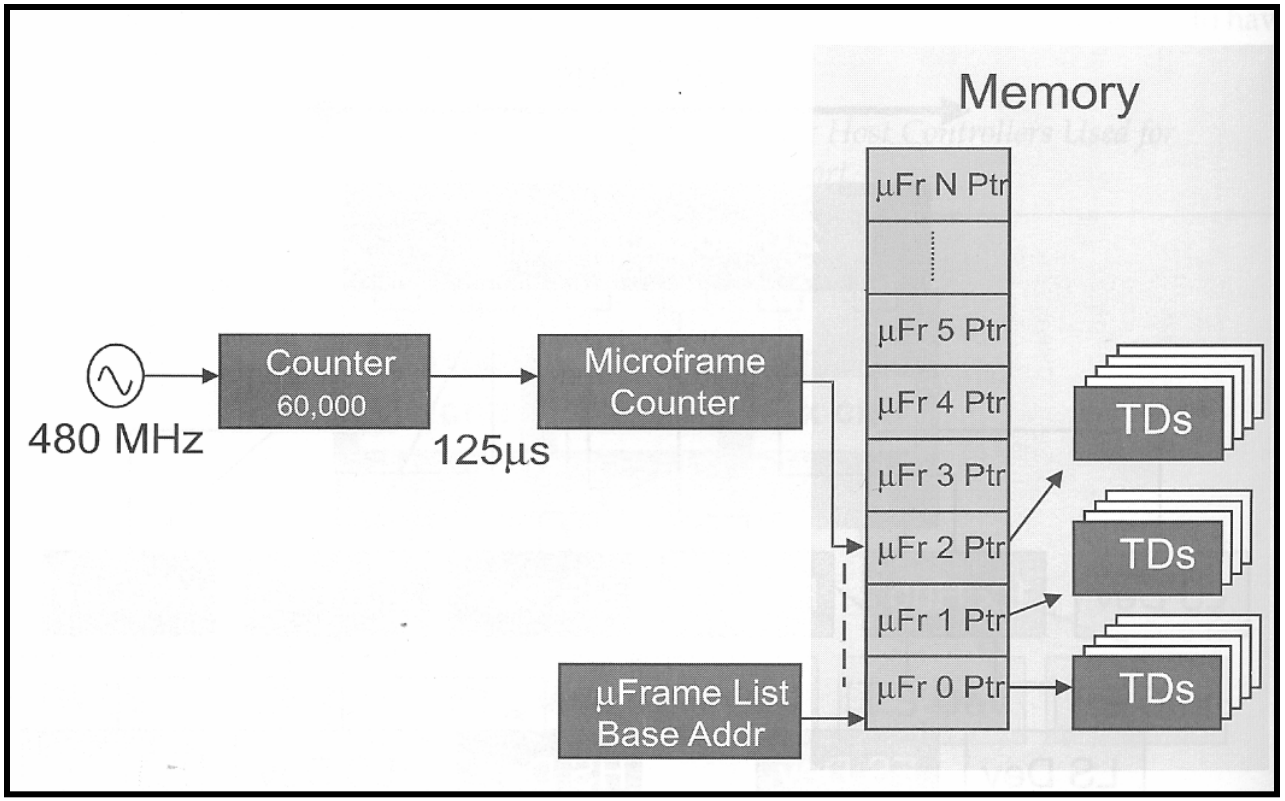


Low Speed Mode
- 1ms frames
- 1,500 bits per frame
- 187.5 KB/s

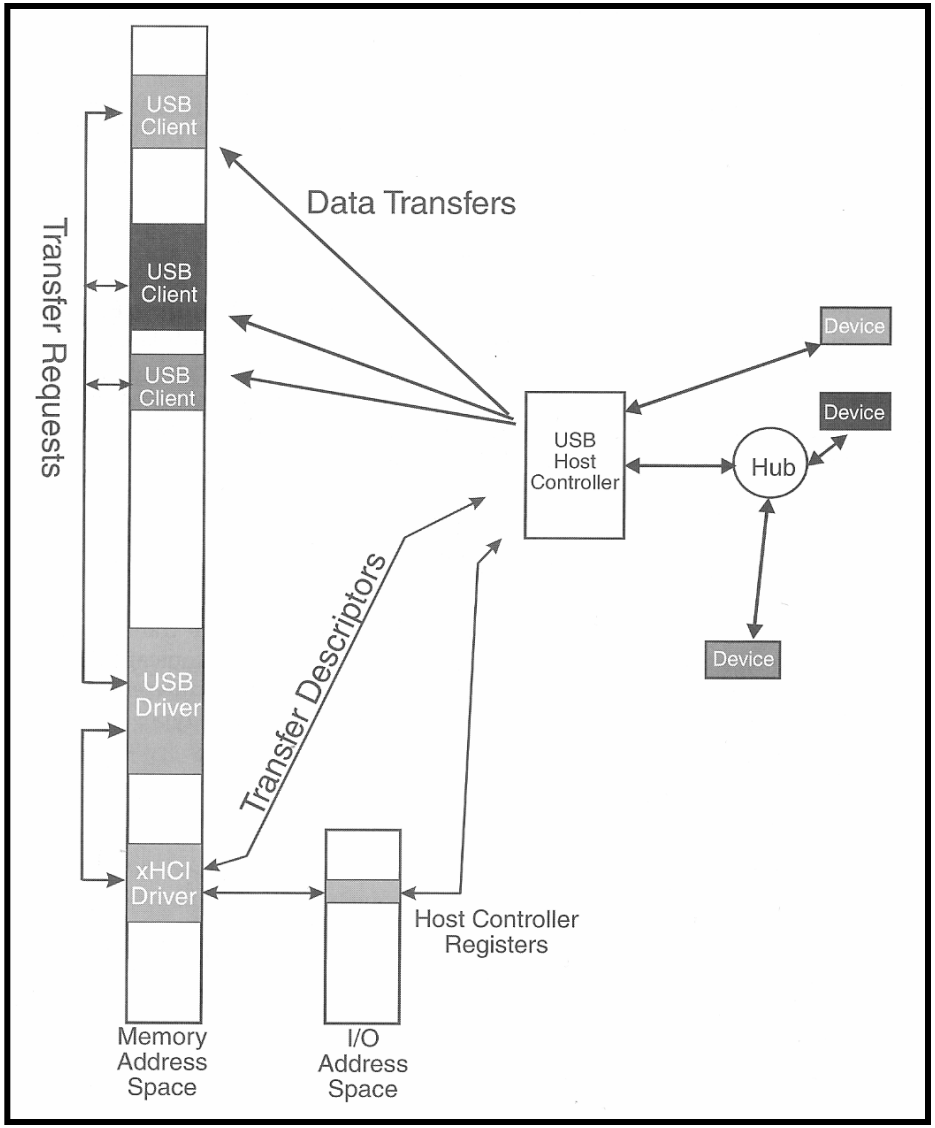
Full Speed Mode
- 1ms frames
- 12,000 bits per frame
- 1.5 MB/s

High Speed Mode
- 125μs frames
- 60,000 bits per μframe
- 60 MB/s

Microframes



Host Controller Handles USB



Packet Details

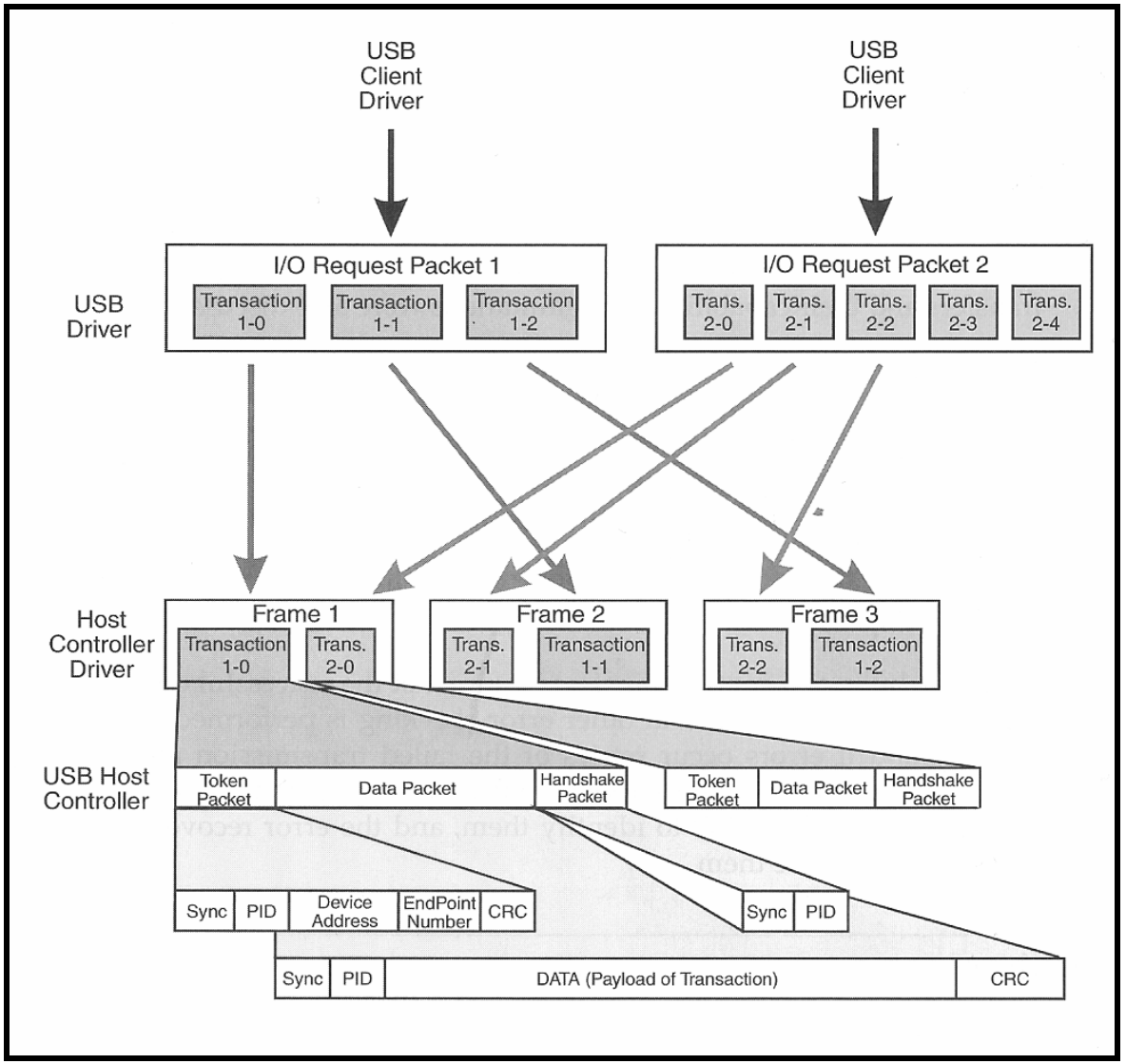
"Payload" Efficiency
How effectively is the available bandwidth used to deliver data?

High Speed

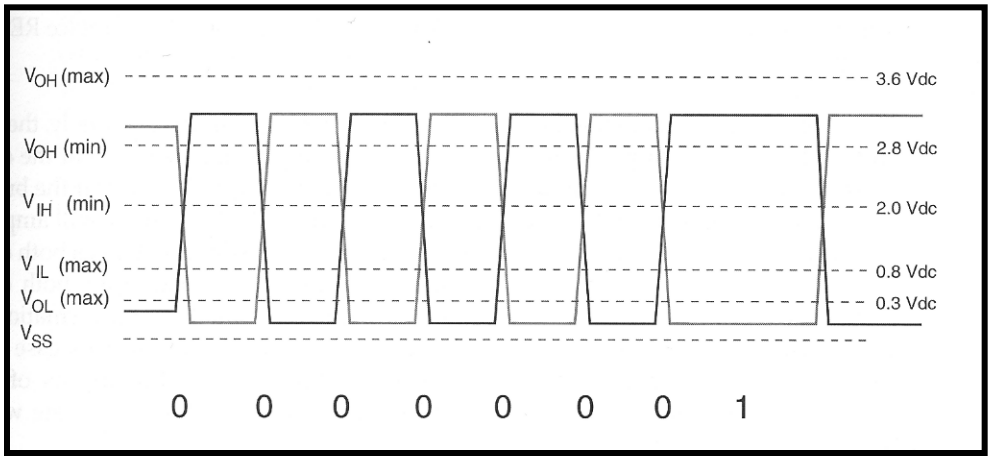
Transfer Type	Max Packet Size	Efficiency
Isochronous	1024 bytes	~96%
Interrupt	1024 bytes	~96%
Bulk	512 bytes	~90%
Control	64 bytes	~27%

Full Speed

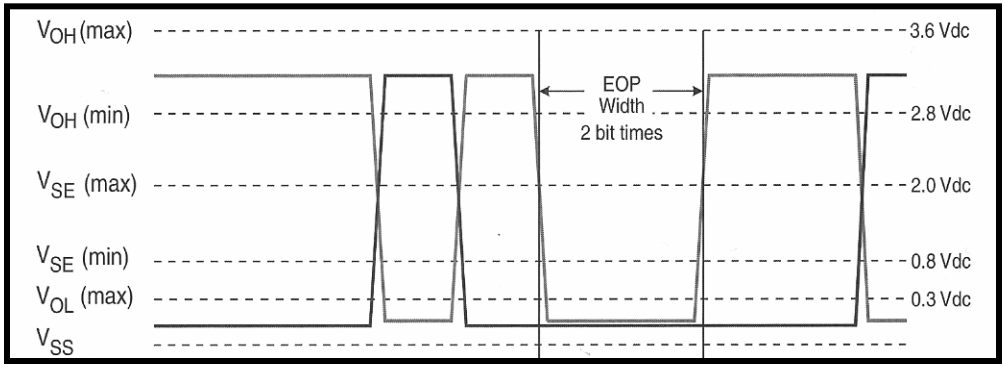
Transfer Type	Max Packet Size	Efficiency
Isochronous	1023 bytes	~99%
Isochronous	512 bytes	~98%
Isochronous	64 bytes	~86%
Other	64 bytes	~82%



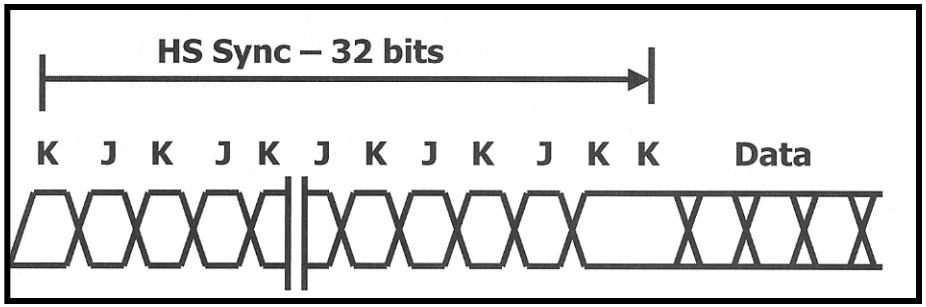
Start of Packet (SOP)
Sync



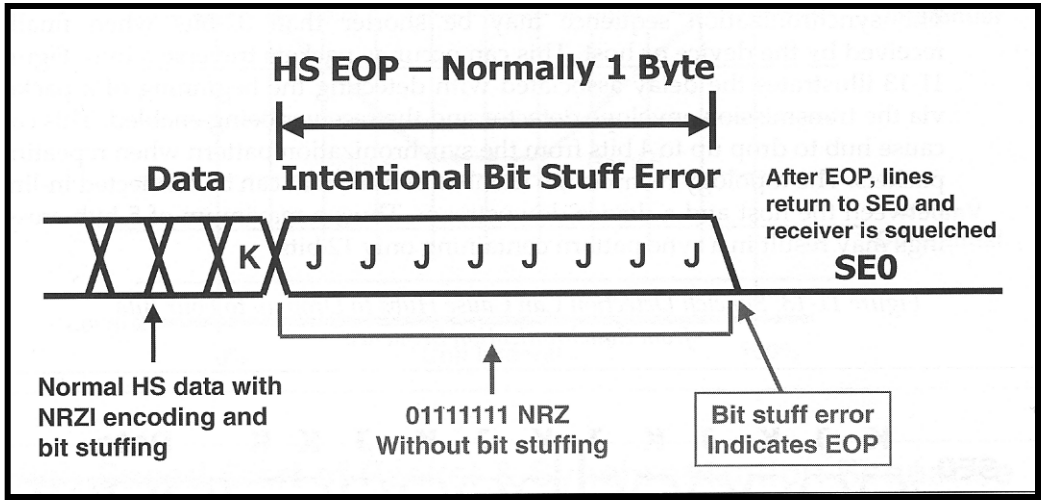
End of Packet (EOP)



High Speed Start Of Packet

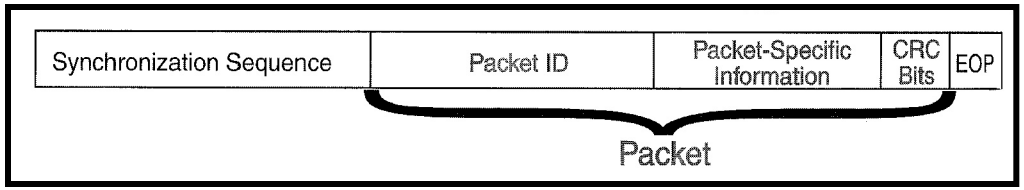


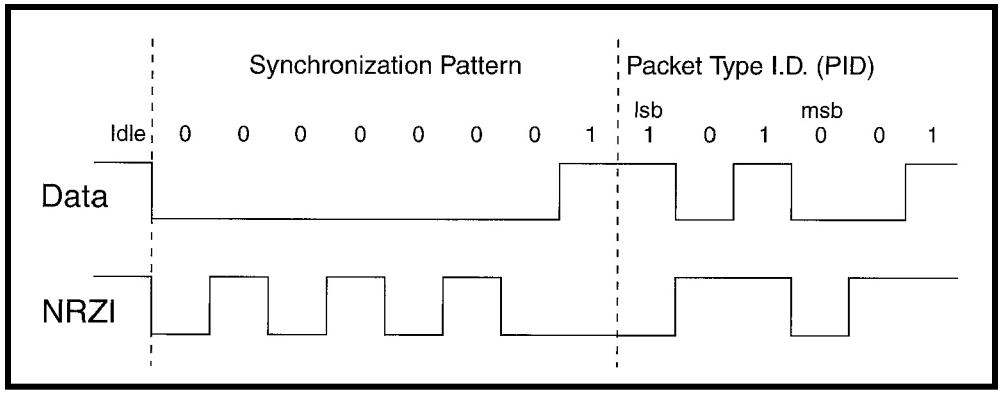
High Speed End Of Packet



Packet Format

Packet Type ID (PID) determines packet type. Actually four bits. Second 4 bits are first 4 bits complemented (as error check)

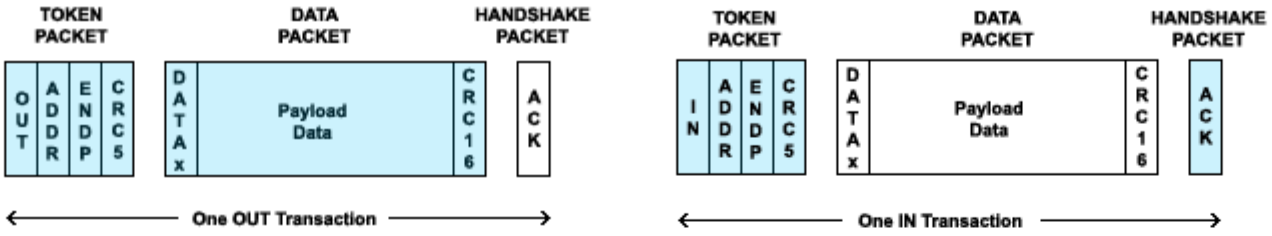




Packets

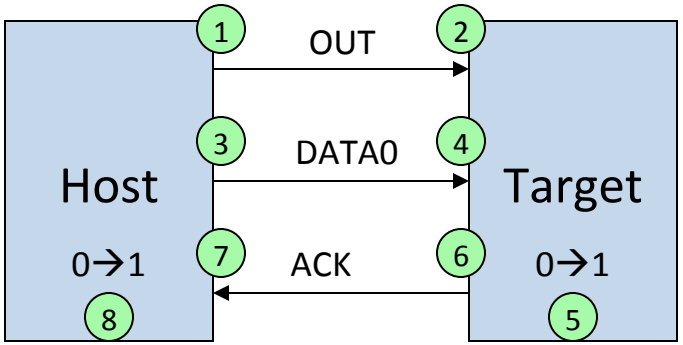
- Token packets
  - Sent at beginning of transaction to specify target endpoint address
- Define type of transaction
  - SOF (Start of Frame)
  - IN (transfer from target USB device to system)
  - OUT (transfer from system to target USB device)
  - SETUP (start of control transfer)
- Data packets
  - For data payload (follow token packets)
- Handshake packets
  - Sent by device to acknowledge receipt of data
    - ACK/NAK/NYET/STALL
- Special packets
  - Pre-amble packets to enable low-speed ports prior to sending low-speed packets

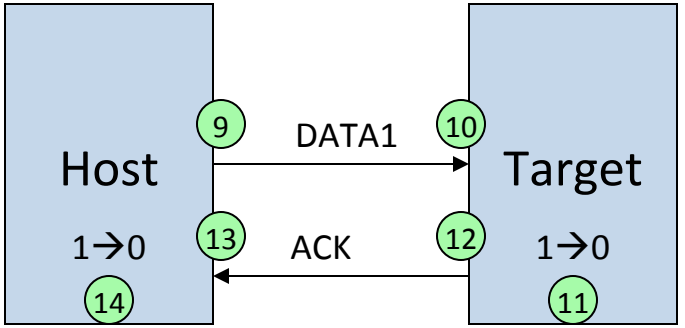
Transactions



Isochronous transactions do not use handshake packets

Data Toggle: Data sent in alternating DATA0 and DATA1 packets to facilitate error recovery.

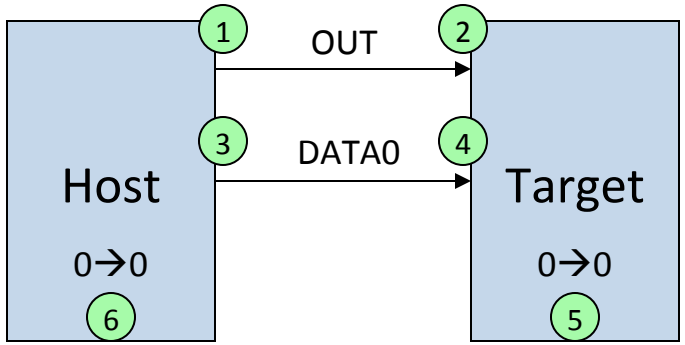


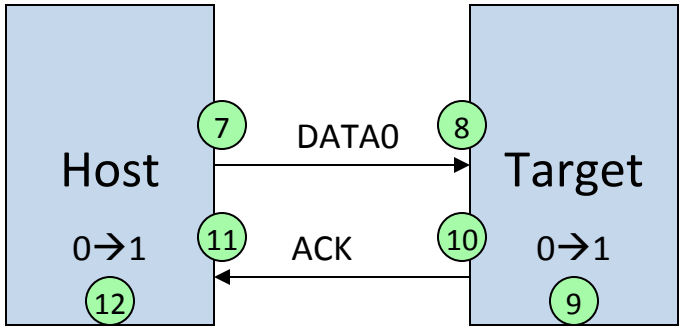


Host begins with toggle = 0, sends DATA0 packet
Target (also toggle=0) is expecting DATA0 packet
Target, receiving correct packet type, toggles & sends ACK
Host, receiving ACK, toggles

Host with toggle = 1, sends DATA1 packet
Target (also toggle=1) is expecting DATA1 packet
Target, receiving correct packet type, toggles & sends ACK
Host, receiving ACK, toggles

Transactions – Data Packet Errors



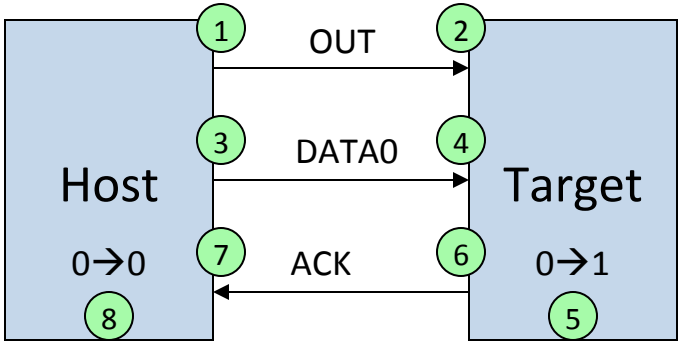


Host begins with toggle = 0, sends DATA0 packet
Target (also toggle=0) is expecting DATA0 packet
Target detects errors in packet, ignores packet, discards data, maintains toggle = 0
Host, not receiving ACK, maintains toggle and retries…

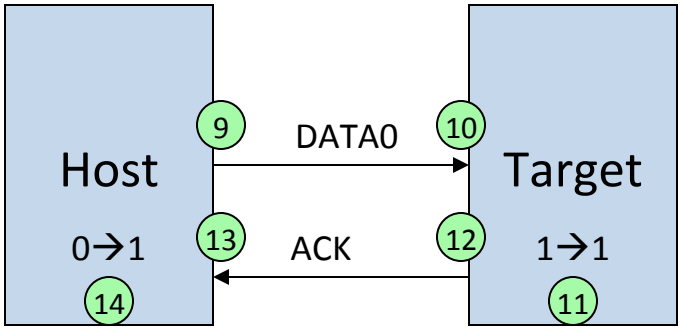
Host retries with DATA0 packet
Host and target consistent

Errors can also occur because ACK wasn't received in specified number of bit times

OUT transaction, bad ACK

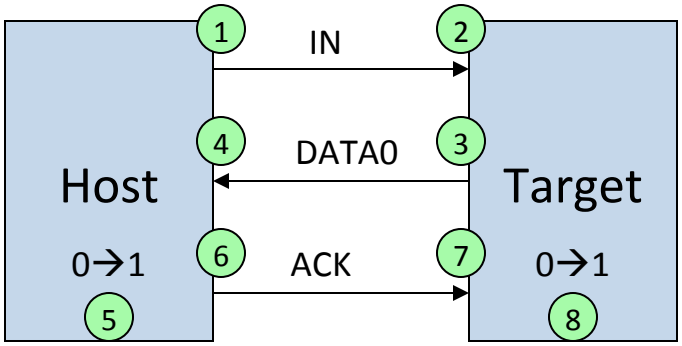


Host begins with toggle = 0, sends DATA0 packet
Target (also toggle=0) is expecting DATA0 packet
Target, receiving correct packet type, toggles & sends ACK
Host fails to receive correct ACK handshake, doesn't toggle

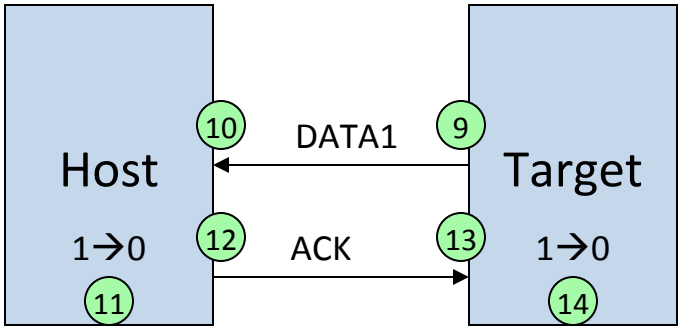


Host with toggle = 0, re-sends DATA0 packet
Target (toggle=1) expecting DATA1 packet, discards DATA0 packet, maintains toggle, sends ACK
Host, receiving ACK, toggles, next transaction will use DATA1 packet

IN transaction, good data

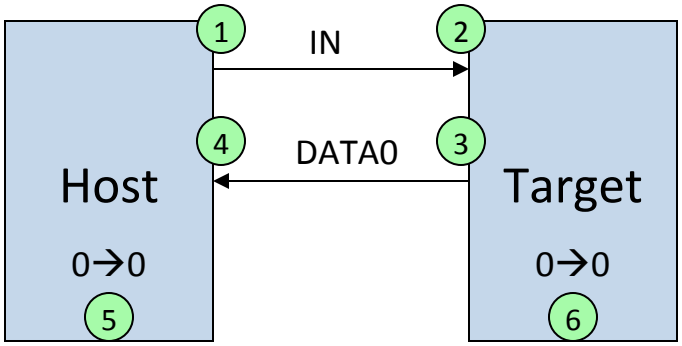


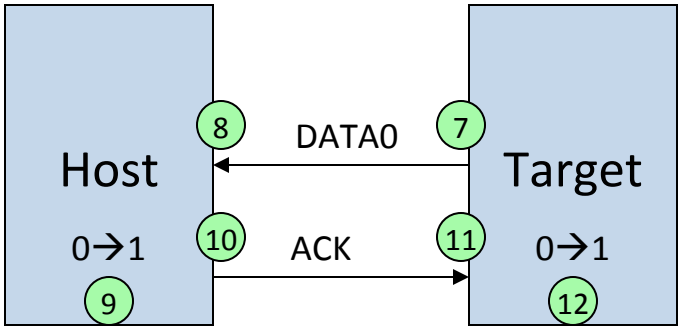
Target begins with toggle = 0, sends DATA0 packet
Host, expecting DATA0 packet, toggles, sends ACK
Target, receiving ACK, toggles



Target with toggle = 1, sends DATA1 packet
Host (also toggle=1) is expecting DATA1 packet
Host, receiving correct packet, toggles, sends ACK
Target, receiving ACK, toggles

IN transaction, bad data

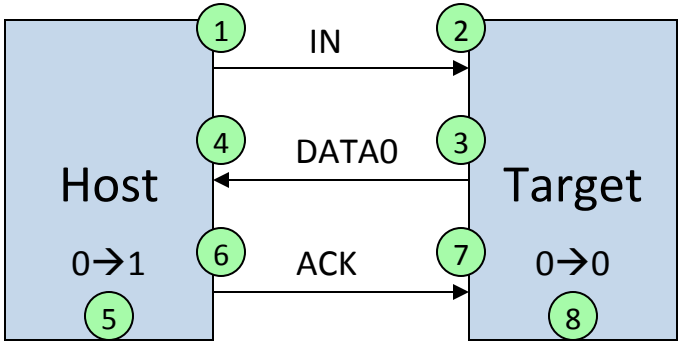




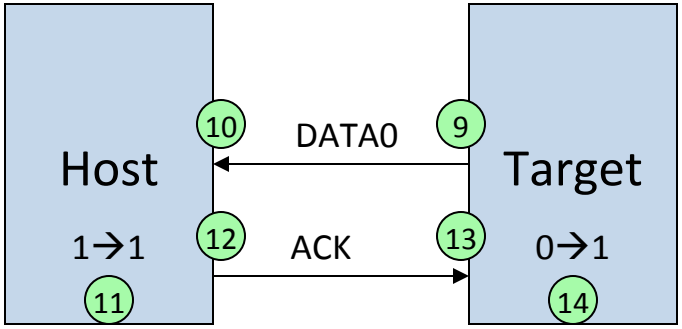
Target begins with toggle = 0, sends DATA0 packet
Host, not receiving good data packet, maintains toggle, doesn't ACK
Target, failing to receive ACK, retries…

Target with toggle = 0, re-sends DATA0 packet
Host (also toggle=0) is expecting DATA0 packet
Host, receiving correct packet, toggles, sends ACK
Target, receiving ACK, toggles

IN transaction, bad handshake



Target begins with toggle = 0, sends DATA0 packet
Host, expecting DATA0 packet, toggles, sends ACK
Target, not receiving good ACK, maintains toggle



Target with toggle = 0, re-sends DATA0 packet
Host (toggle=1) expecting DATA1 packet, discards data, sends ACK, maintains toggle
Target, receiving ACK, toggles

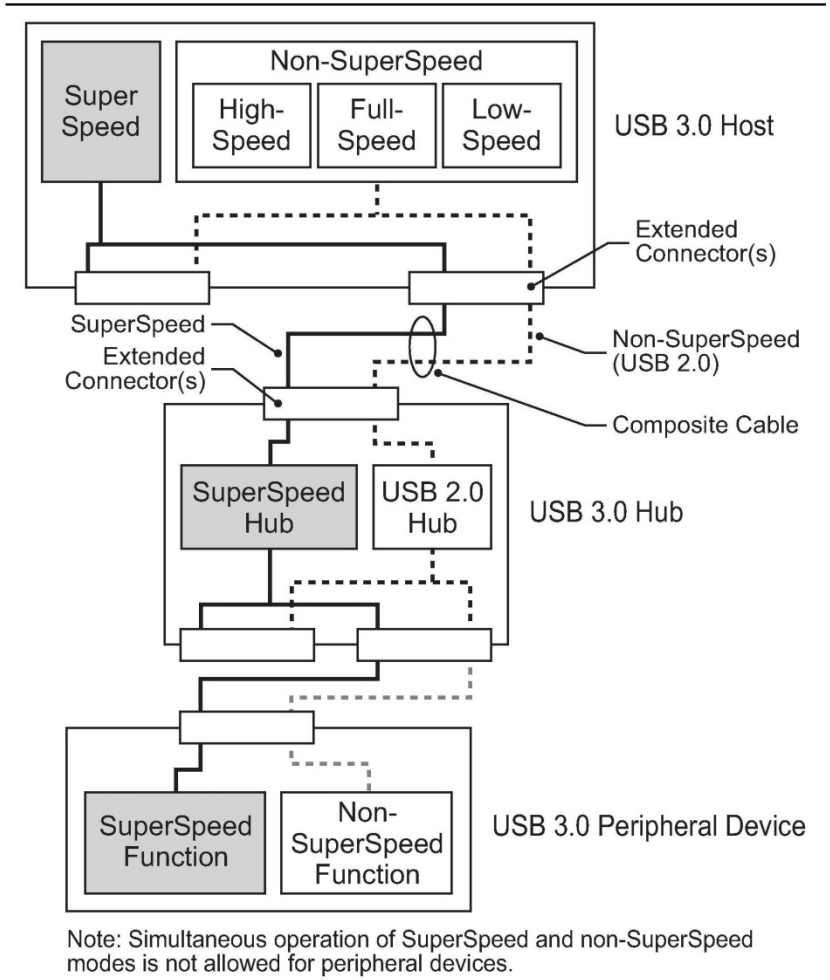
USB Features

- No arbitration!
  - Everything controlled by host (master)
  - Point-to-point with host as one point
  - No device-to-device communication
- USB devices don't use I/O or memory address space
  - Memory used for buffers, descriptors
  - Devices have "addresses" but not from I/O or memory space
- Error detection and correction
  - CRC (16-bit for data packets, 5-bit for others)
  - ACK, DATA0/DATA1
  - Resend bad packets (unless isochronous)

USB 3.0

- Status
  - Introduced September 2007 at Intel Developer's Forum
  - Version 1.0 specification complete November 2008
  - First consumer products certified in January 2010
- Introduces SuperSpeed Mode
  - Preserves backwards compatibility
    - USB 2.0 devices will work in USB 3.0 systems
  - Forward compatible
    - USB 3.0 devices will work in USB 2.0 systems
  - Signaling similar to PCI-e 2.0
  - Essentially adds second bus alongside USB 2.0
    - 4.8 Gbps
    - Two additional differential signal pairs (separate transmit/receive)
    - Eliminate polling
    - Additional power and ground
    - Additional power available to devices
    - Low power modes

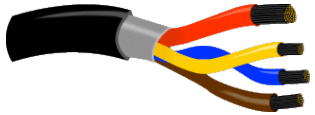
USB 3.0



Note: Simultaneous operation of SuperSpeed and non-SuperSpeed modes is not allowed for peripheral devices.

USB 2.0 Connectors

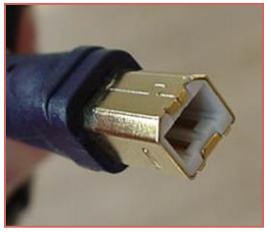
- Standard A connectors
  - USB cable to USB port
  - Most common
  - Cable attached to peripheral at other end



Contact Number	Signal Name	Cable Color
1	Vcc/VBus	Red (Orange)
2	-Data	White (Yellow)
3	+Data	Green (Grey)
4	Ground	Black (Blue, Brown)

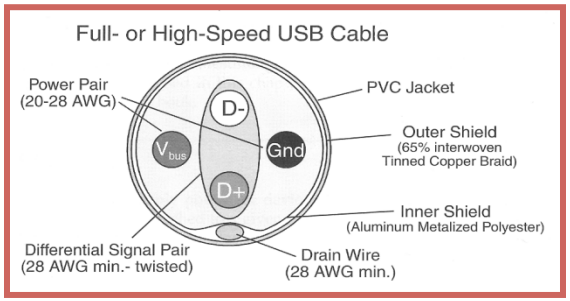
- Standard B connectors
  - USB cable to peripheral
  - Used with detachable cables
- Mini connectors (A and B)
  - 5th wire: ID signal (Gnd/NC for A/B)
  - No longer being developed
- Micro connectors (A and B)
  - Replace mini connectors
  - OTG (On The Go) devices
    - Micro A-B receptacle accepts micro-A or micro-B plugs
    - Devices senses ID, behaves as host or peripheral
- Power contacts (1mm) longer than differential data contacts
  - Ensure power prior to data contacts mating

"A" connection upstream to PC

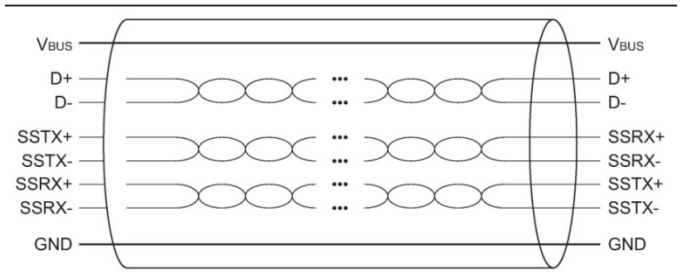


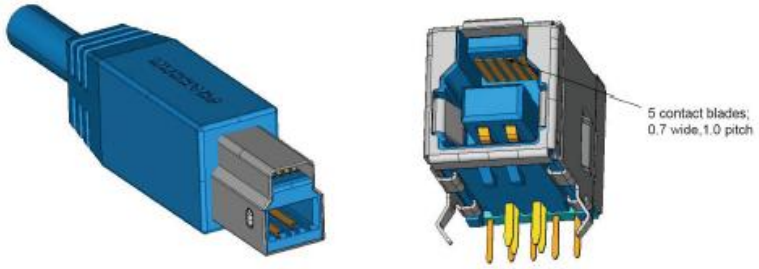
"B" connection downstream to peripheral

"Mini B" connection downstream to peripheral

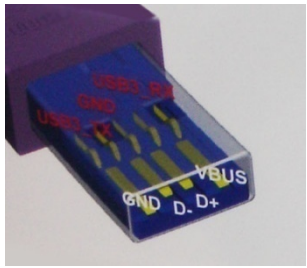


USB 3.0 Connectors





USB 3.0
Standard-B plug and receptacle
Receptacle can accept USB 2.0 plug

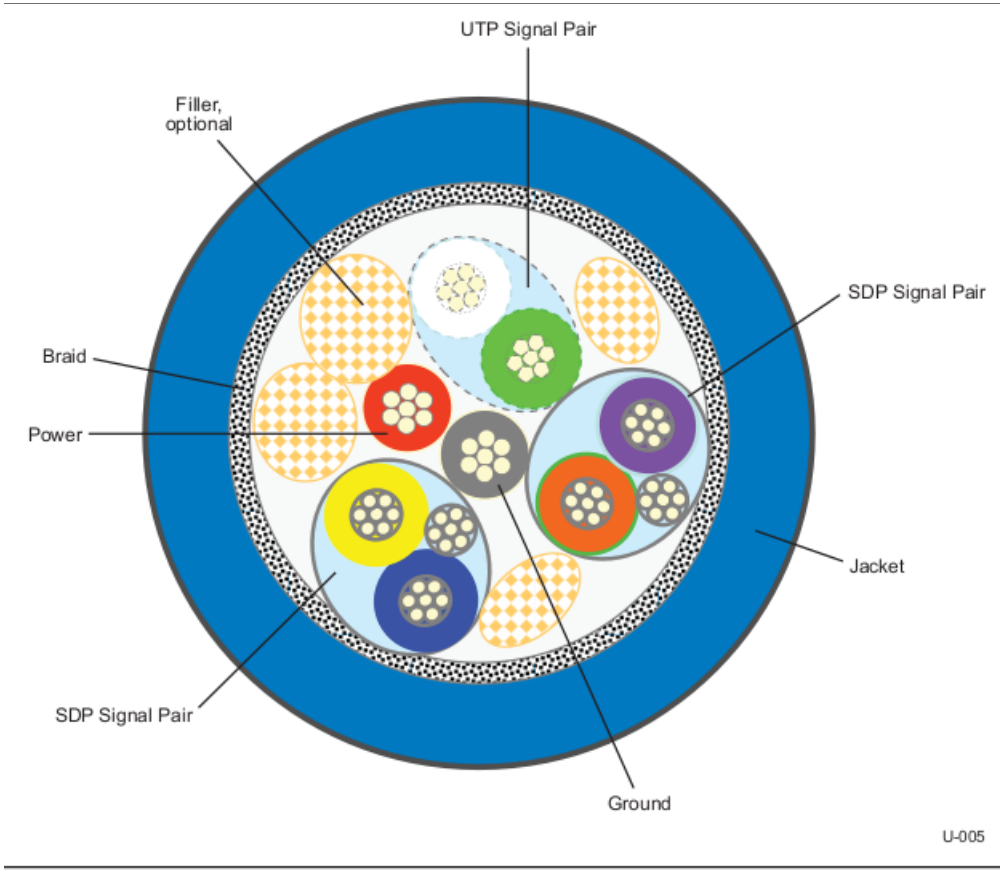


USB 3.0
Standard-A plug
Works with USB 2.0 Standard-A receptacle

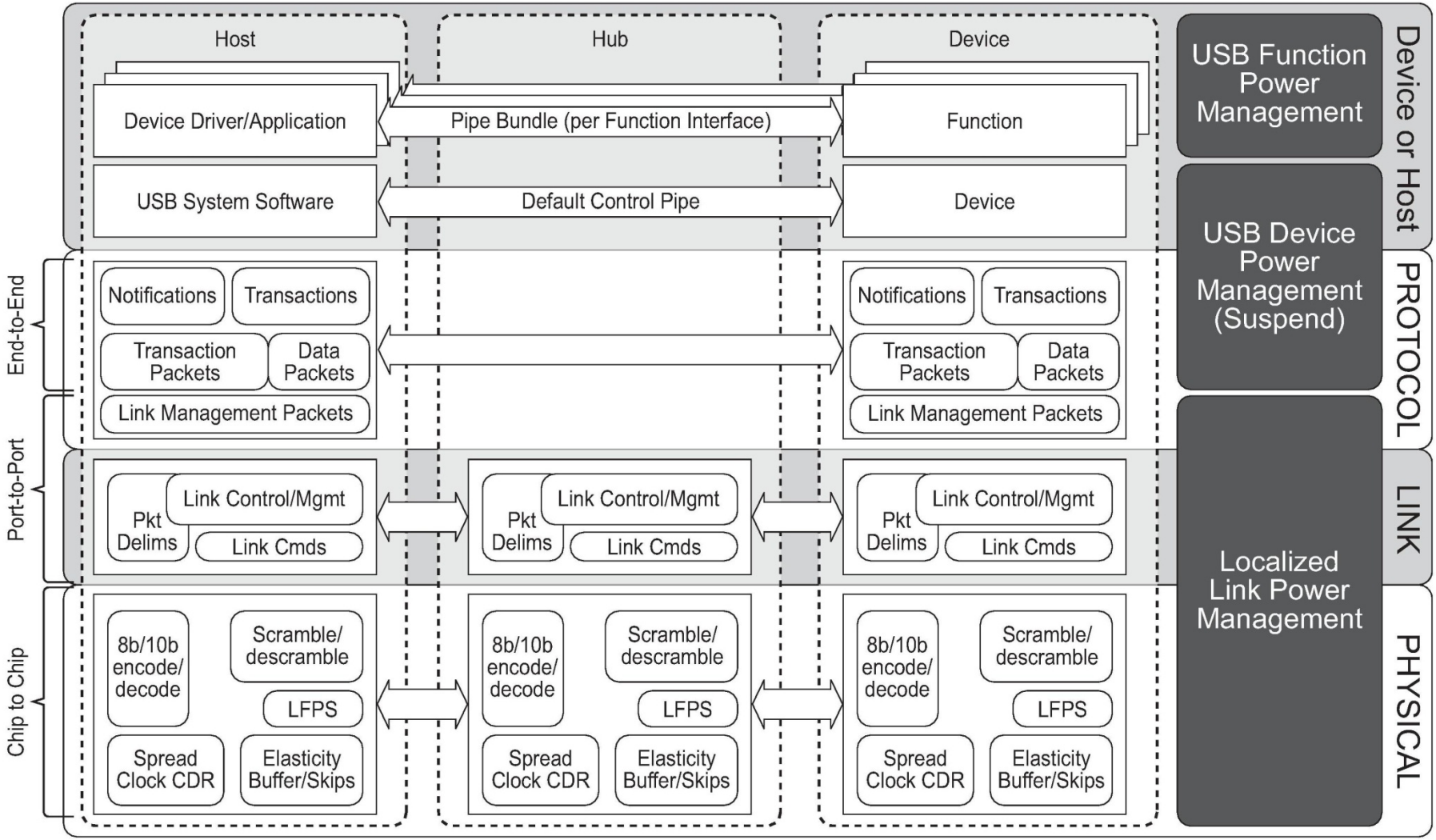
Receptacle	Plugs Accepted
USB 2.0 Standard-A	USB 2.0 Standard-A or USB 3.0 Standard-A
USB 3.0 Standard-A	USB 3.0 Standard-A or USB 2.0 Standard-A
USB 2.0 Standard-B	USB 2.0 Standard-B
USB 3.0 Standard-B	USB 3.0 Standard-B or USB 2.0 Standard-B
USB 3.0 Powered-B	USB 3.0 Powered-B, USB 3.0 Standard-B, or USB 2.0 Standard-B
USB 2.0 Micro-B	USB 2.0 Micro-B
USB 3.0 Micro-B	USB 3.0 Micro-B or USB 2.0 Micro-B
USB 2.0 Micro-AB	USB 2.0 Micro-B or USB 2.0 Micro-A
USB 3.0 Micro-AB	USB 3.0 Micro-B, USB 3.0 Micro-A, USB 2.0 Micro-B, or USB 2.0 Micro-A

USB 3.0 Signals

- 6 wires added for USB 3.0
  - one TX pair
  - one RX pair
  - two ground wires
- USB 2.0 devices use 4 separate wires
  - one TX/RX pair
  - two ground wires



USB 3.0


Host
Hub
Device
Device Driver/Application
Pipe Bundle (per Function Interface)
Function
USB System Software
Default Control Pipe
Device
USB Function Power Management
USB Device Power Management (Suspend)
Localized Link Power Management
Device or Host
PROTOCOL
LINK
PHYSICAL
End-to-End
Port-to-Port
Chip to Chip
Notifications
Transactions
Transaction Packets
Data Packets
Link Management Packets
Pkt Delims
Link Control/Mgmt
Link Cmds
8b/10b encode/ decode
Scramble/ descramble
LFPS
Spread Clock CDR
Elasticity Buffer/Skips

USB 3.0

- Details
  - 8b/10b encoding
  - Spread spectrum clocking

IEEE 1394

- Introduced 1995
- Apple FireWire, Sony iLink
- 400 Mbps
- Supports peer-to-peer communications
  - Doesn't require PC
- Consumer electronics
  - e.g. Video camera to PC

Comparison

Interface	Introduced	Speed
Serial Port	1960s	20 Kbps
Parallel Port	1981	1.1 Mbps
USB	1995	12 Mbps
FireWire (1394)	1995	400 Mbps
USB 2.0	2000	480 Mbps
FireWire 800	2001	850 Mbps
USB 3.0	2010	4.8 Gbps (bidir)