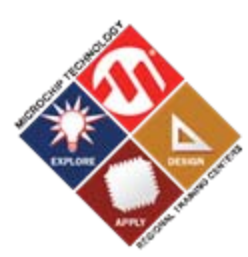


HANDS-ON

Training

USB for Embedded Applications Microchip Technology



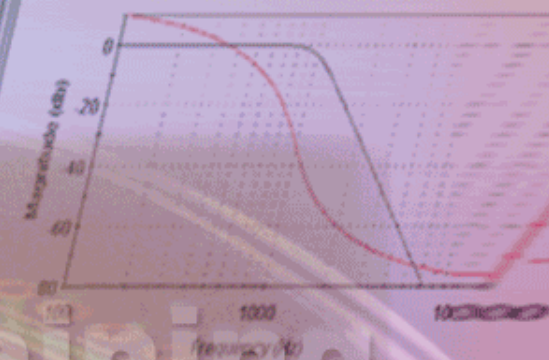


Objective

- **Understand the basics of USB, and how it can be used in an embedded application**
- **Be familiar with Microchip's MCUs, development boards, and USB software framework.**
- **Be able to create a simple PC application that exchanges data with a USB device**

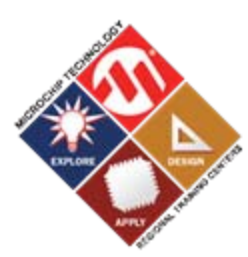
HANDS-ON

Training



Part 1. Introduction to Full-Speed USB

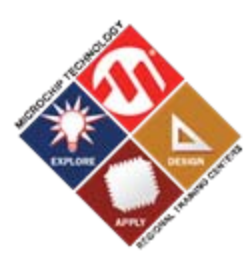




Objectives

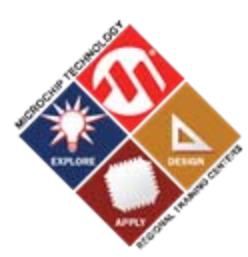
- **Understand how USB can be useful in an embedded system**
- **Learn about fundamental USB architecture, protocol and programmer's model**
- **Be aware of the factors important in designing a USB application**
- **Identify key USB capabilities in PIC18 USB MCUs**





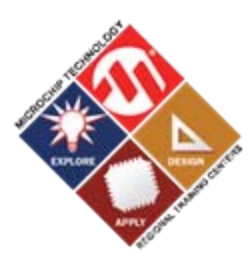
Agenda

- **Brief history of USB & USB-IF**
- **USB Fundamentals - The serious & important stuff**
 - ➔ **Basics/Speeds**
 - **Architecture/Programmer's Model**
 - **Physical Connection**
 - USB Transactions
 - USB Transfers
 - Device Classes
 - Enumeration
 - Descriptors
 - Power Planning
 - VID/PID & USB Compliance
- **PIC18F USB Microcontrollers**
- **Microchip Demo/Development Solutions**

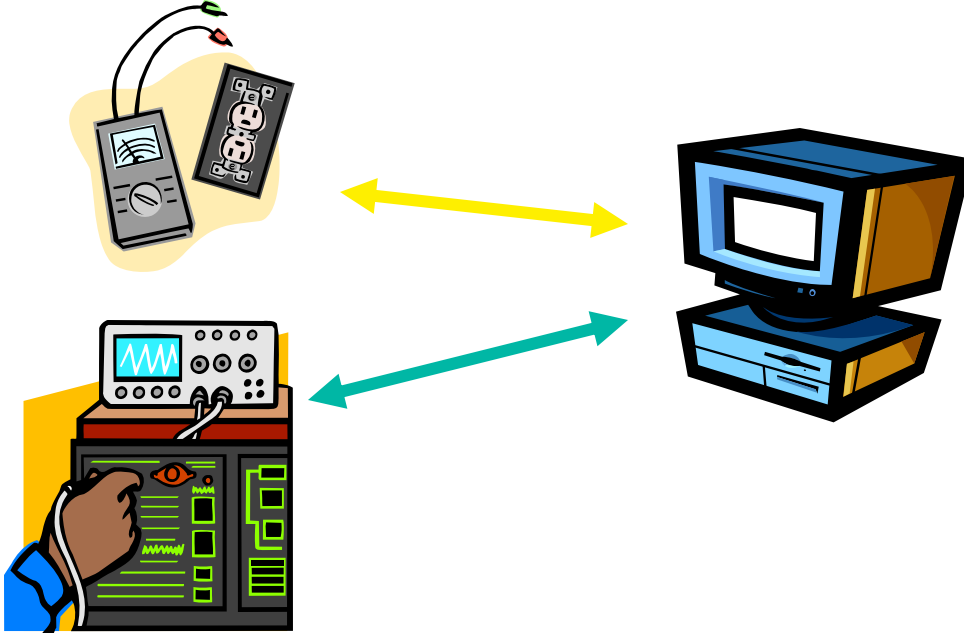


A little history...

- **USB was co-developed by a group of companies....**
 - Compaq
 - Intel
 - Microsoft
 - NEC
 - ...who wanted to make it much easier to add/remove peripheral devices from PCs
- **1998 - USB 1.1**
- **2000 - USB 2.0**



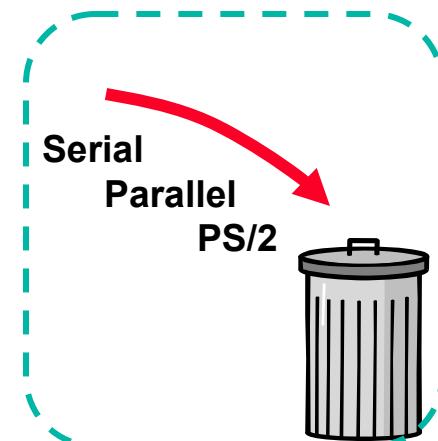
Universal Serial Bus



Extend the functionality of your computer!

Data Analysis,
Data Logging,
Firmware Updates,
Diagnostics,
Embedded Applications!

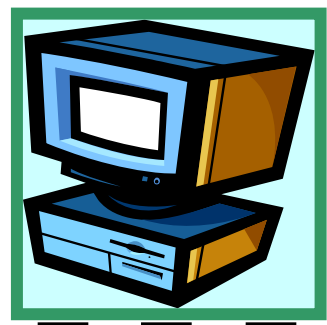
- Auto detection & configuration (Plug&Play)
- Easy expansion using hubs
- Bus power
- Data CRC protected, bad packets resent.
- Three speeds:
Low- 1.5, Full- 12, High- 480 Megabits / second



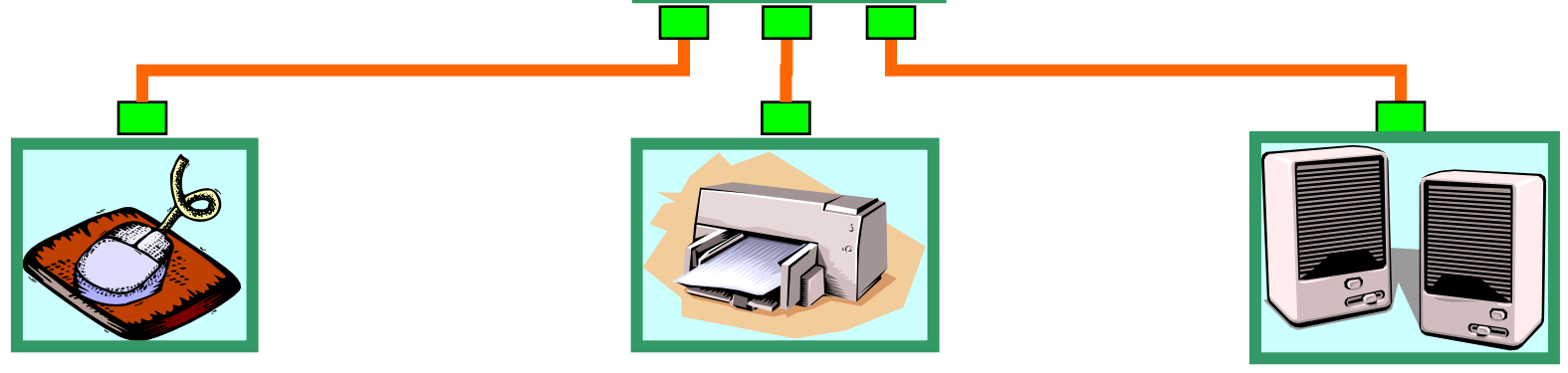


USB Basics

- USB is a “Single Master + Multiple Slaves” polled bus



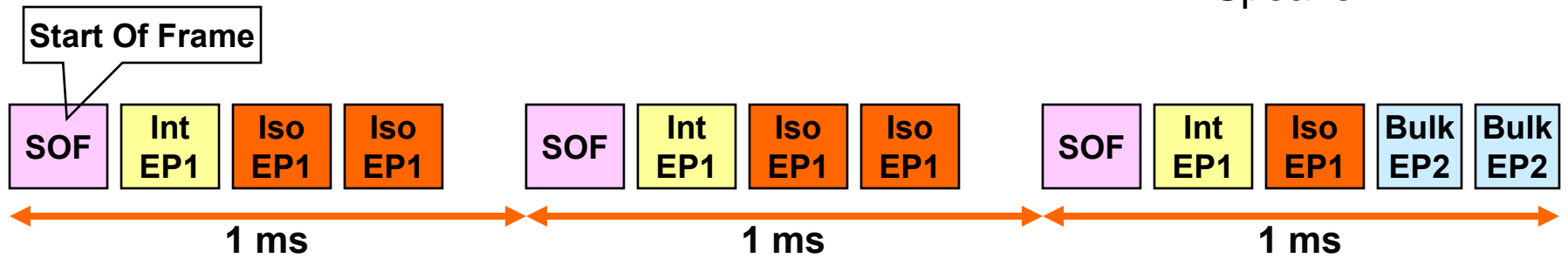
USB Host Controller (Master) and Root Hub

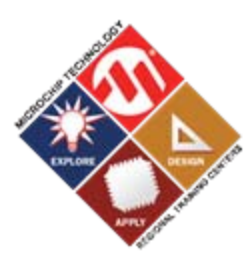


Mouse

Printer

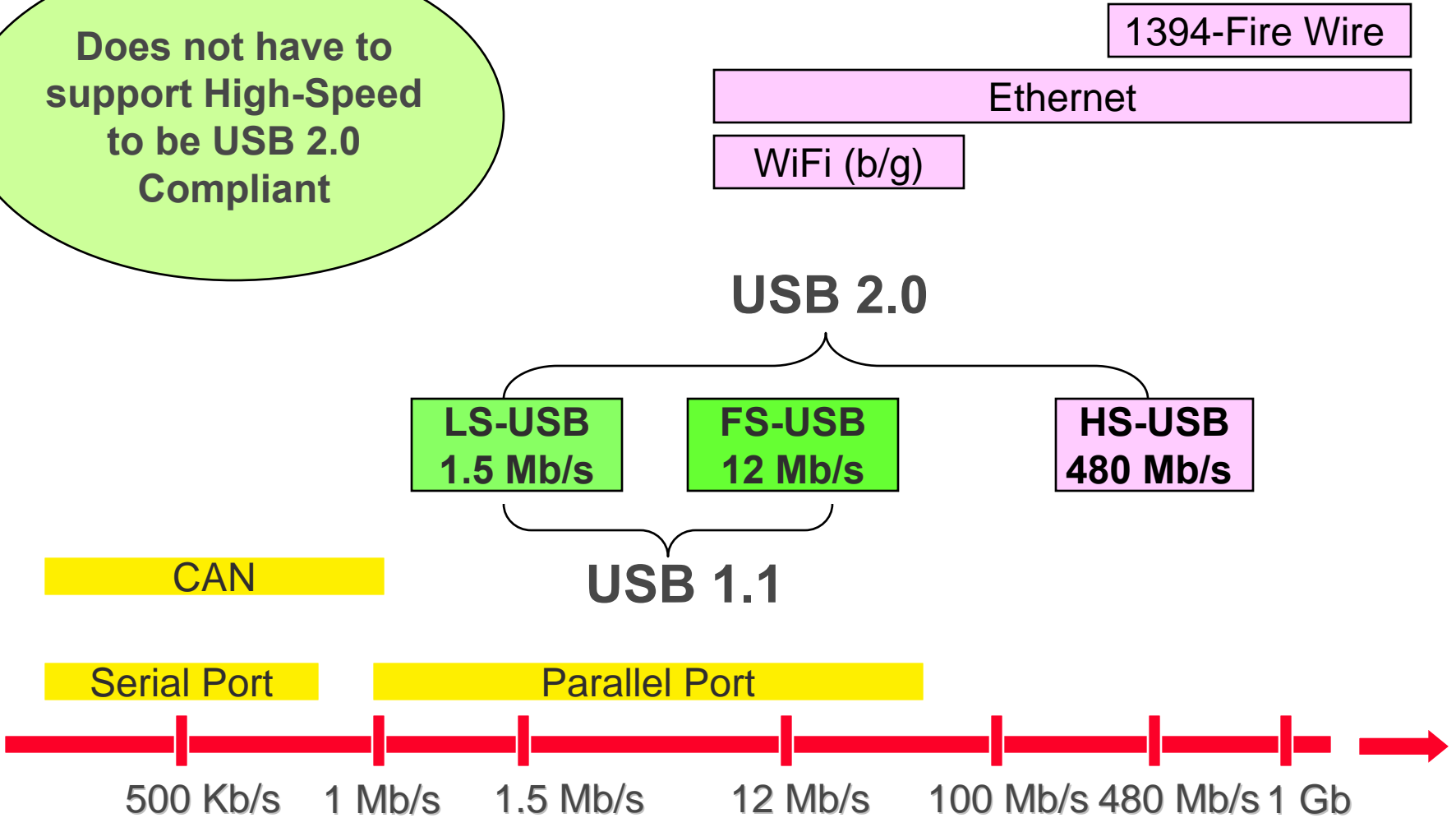
Speaker

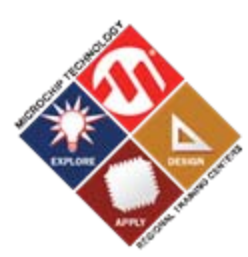




Buses & Speeds Comparison

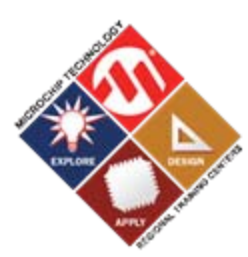
Does not have to support High-Speed to be USB 2.0 Compliant





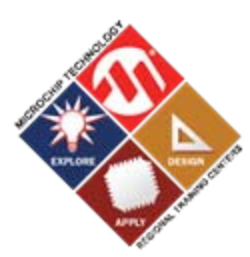
Biggest Myth

- **MYTH:** A Low-Speed USB peripheral can transfer application data up to 187.5 KB/s (1.5 Mbps)
- **FACT:** Impossible, because of a USB specification restriction:
 - 8 byte data transfer every 10 ms
 - = 800 Bytes/second only



Next Biggest Myth

- **MYTH:** A Full-Speed USB peripheral can transfer data up to 1.5 MB/s (12 Mb/s)
- **FACT:** Impossible, 1.5 MB/s is the total bus bandwidth
 - **Must be shared among peripherals**
 - **Protocol overhead**
 - **Protocol restrictions**
 - **Realistic raw data throughput to a single peripheral is ~1.0 MB/s**
 - **Only 64KB/s in some cases**



Physical Bus Topology

Host (Tier 1)



USB Host Controller & Root Hub

Tier 2

Keyboard

Speaker

Hub

Tier 3

Logic Analyzer

Hub

Tier 4

Hub

Tier 5

Hub

Tier 6

Hub

Tier 7

Data Logger

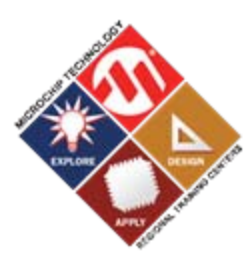
Up to 126 peripherals...



Hub: Max Chaining = 5

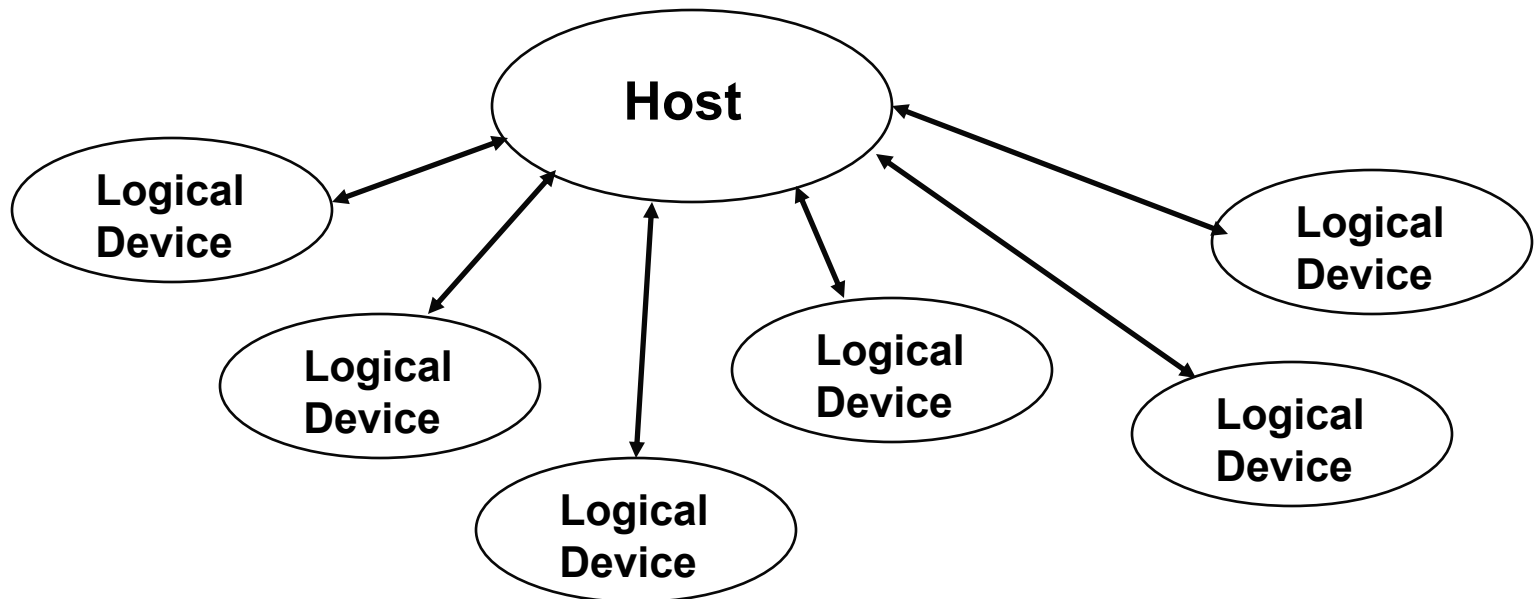
PIC18 USB devices are designed to be peripherals!

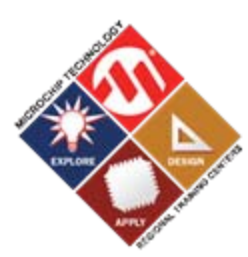




Logical Bus Topology

- **Not a tiered-star!**
- **Host software communicates to each “logical” device as if it were directly connected to the root hub**





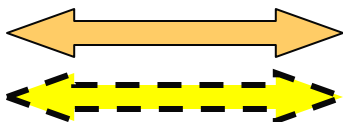
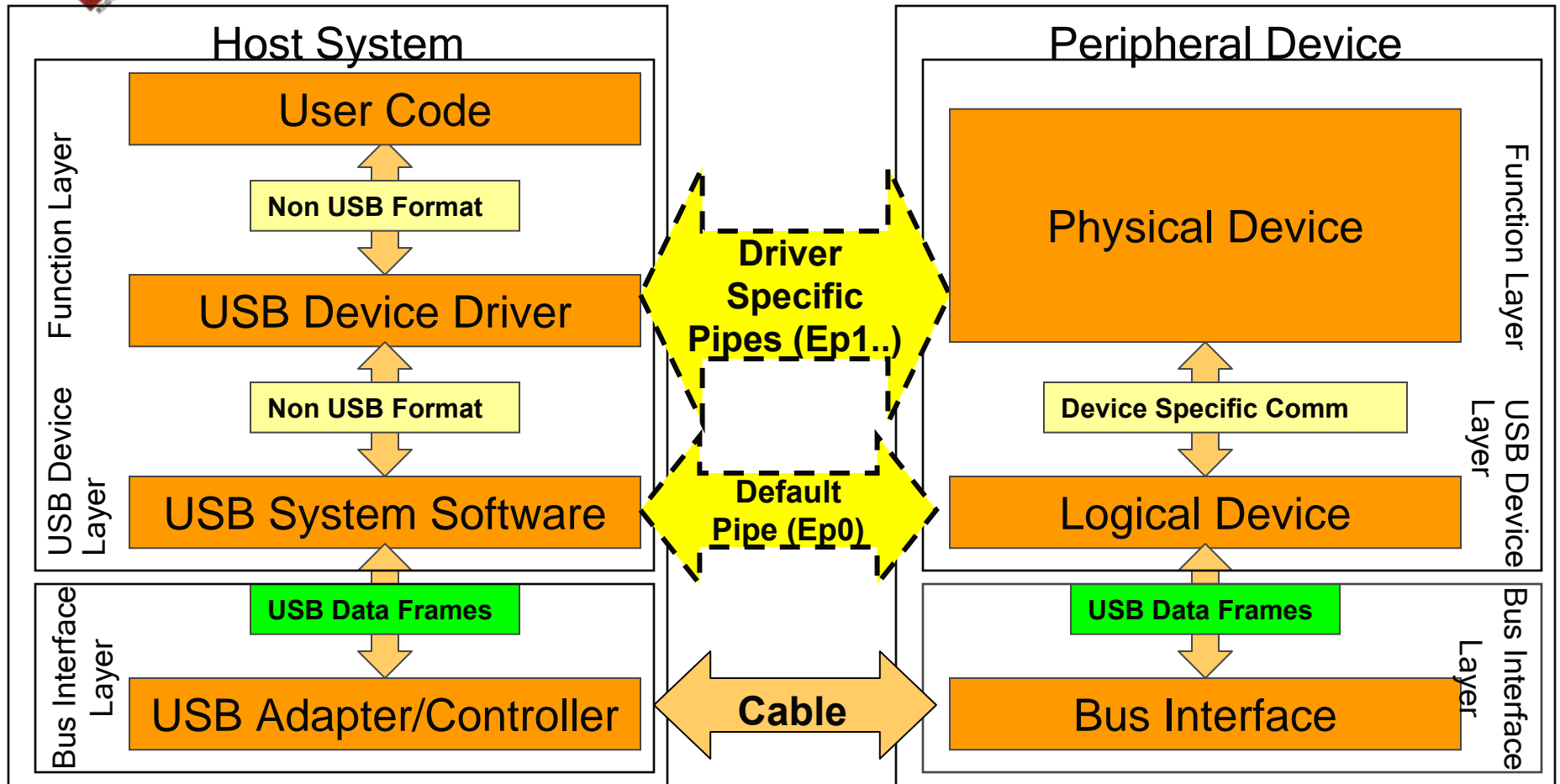
Accessing PC Peripherals

- **Old Way**
- **PC Peripherals:**
 - Memory-mapped into the x86 I/O address space
 - Assigned a specific IRQ line
 - Assigned a specific DMA channel
- **Accessed directly (ISA, PCI, PCMCIA)**
- **USB Way**
- **PC Peripherals:**
 - Mapped into a virtual 127 device address space
 - Does not use any PC I/O, IRQ or DMA resources
- **Accessed indirectly using the programming interface provided by the device driver**



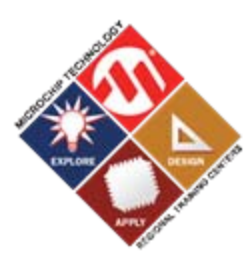
USB Device Framework

- Software View of Hardware -

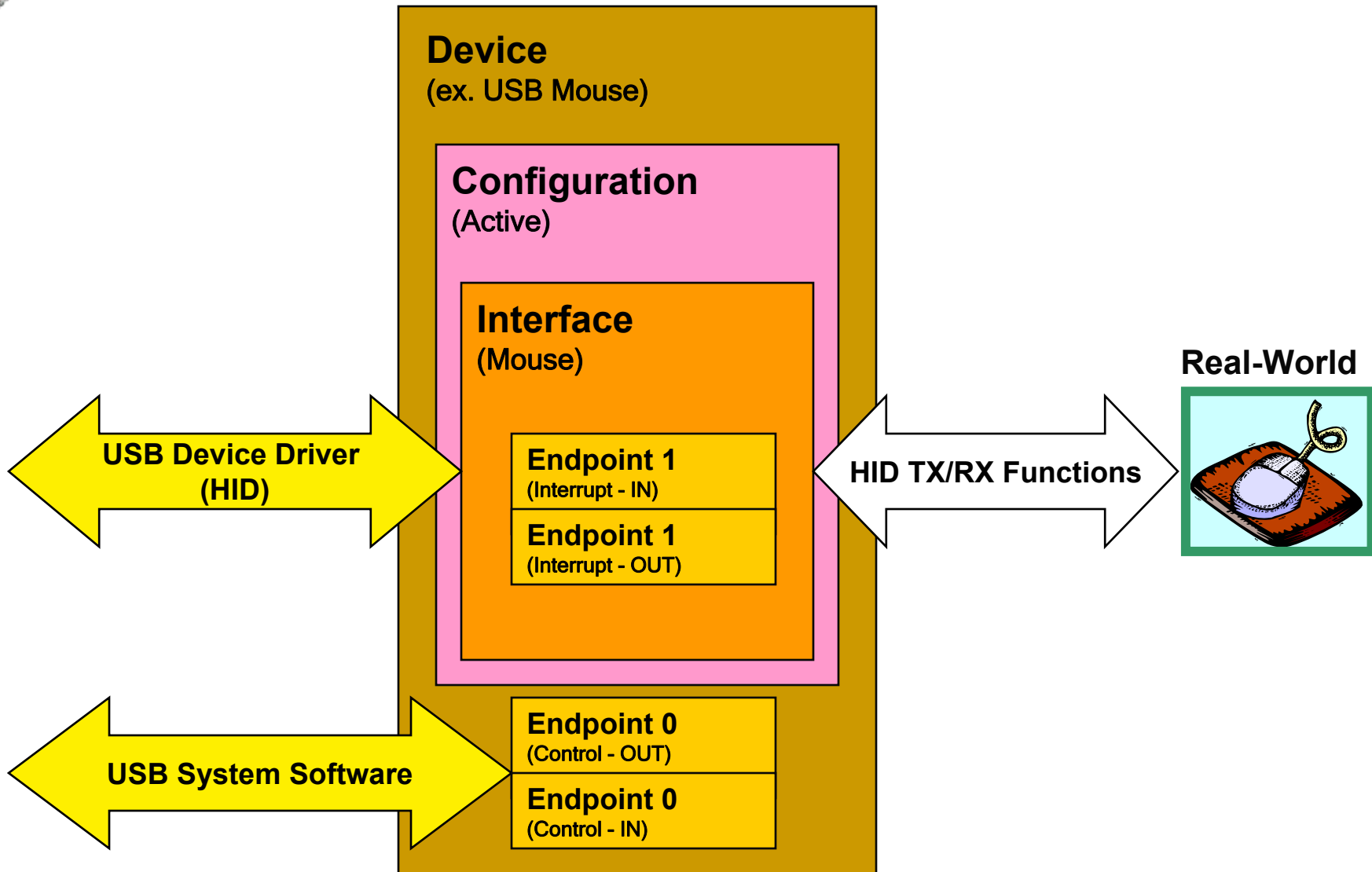


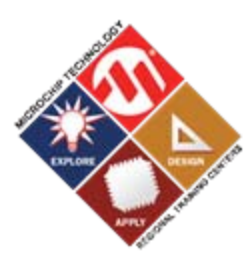
Physical Communication Path

Logical Communication Path ("Pipe")

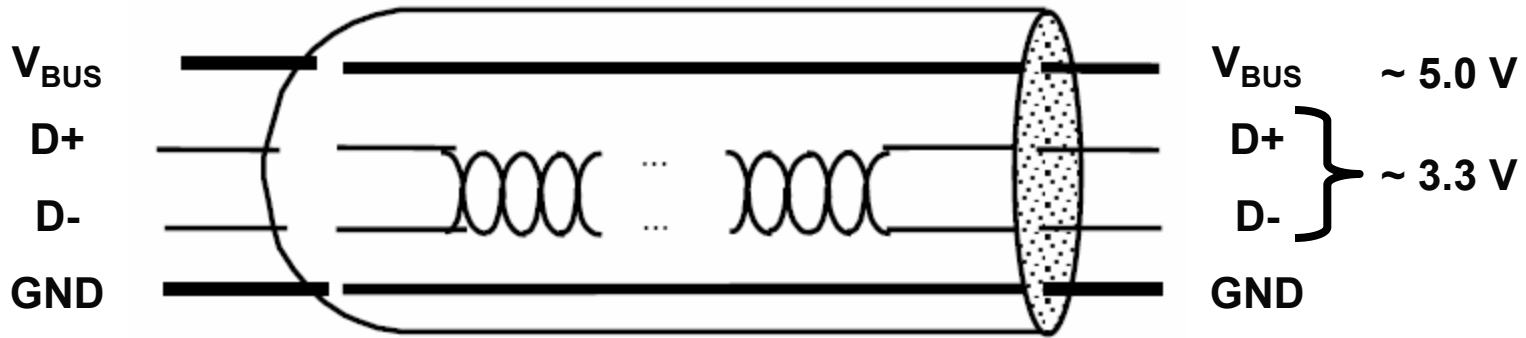


The “Logical” USB Device



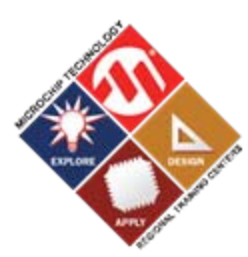


Physical Interface

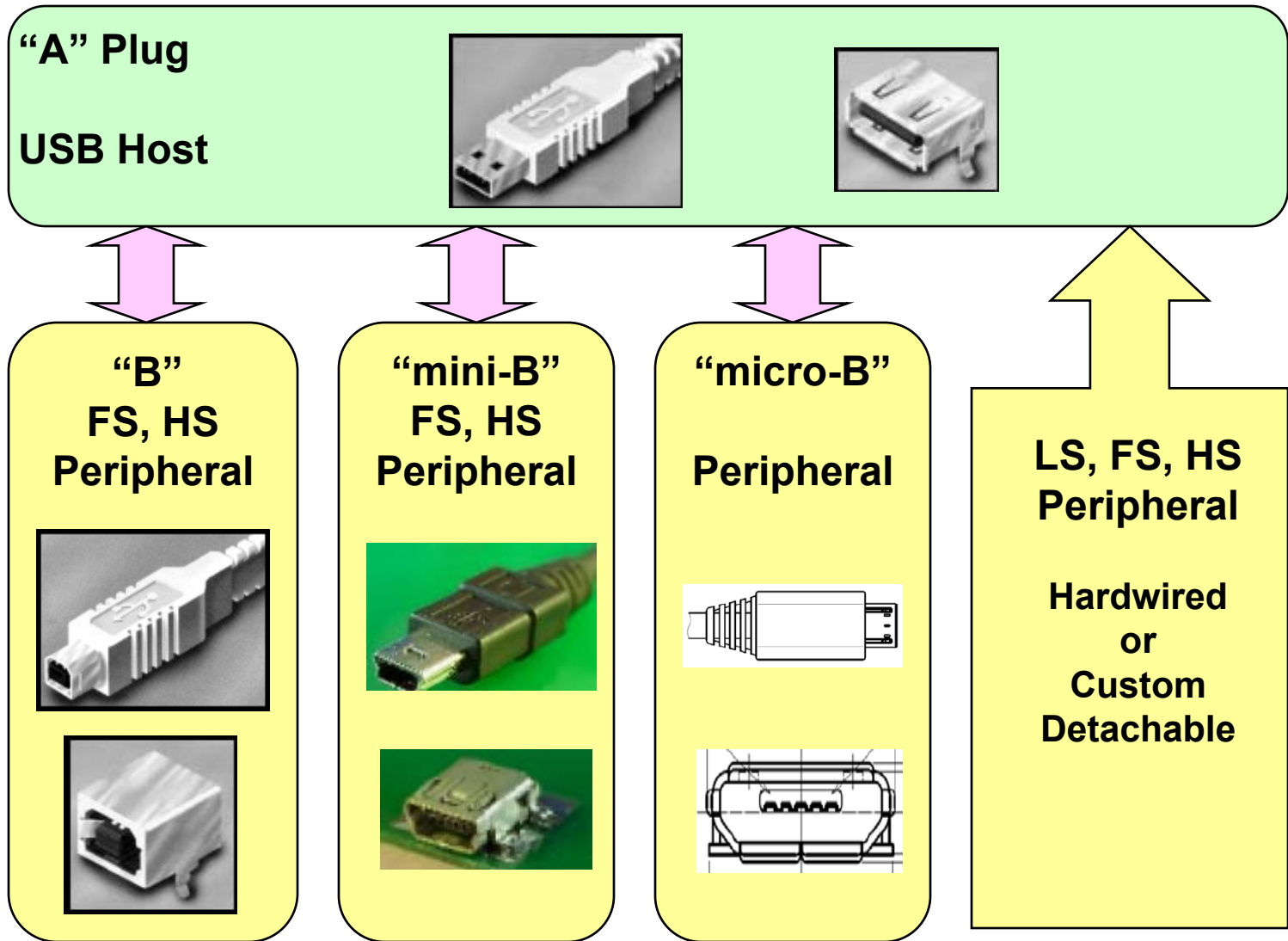


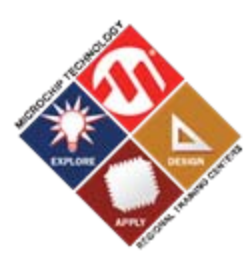
- **Half Duplex with NRZI Data Encoding**
- **Bus Power to each device:**
 - 4.40 - 5.25 V
 - Guaranteed 100 mA
 - 500 mA maximum through negotiation

Must use external power if more is required



USB Connectors





Agenda

- **Brief history of USB & USB-IF**
- **USB Fundamentals - The serious & important stuff**
 - Basics/Speeds
 - Architecture/Programmer's Model
 - Physical Connection
 - **– USB Transactions**
 - **USB Transfers**
 - **Device Classes**
 - Enumeration
 - Descriptors
 - Power Planning
 - VID/PID & USB Compliance
- **PIC18F USB Microcontrollers**
- **Microchip Demo/Development Solutions**

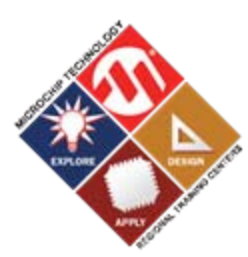
HANDS-ON

Training

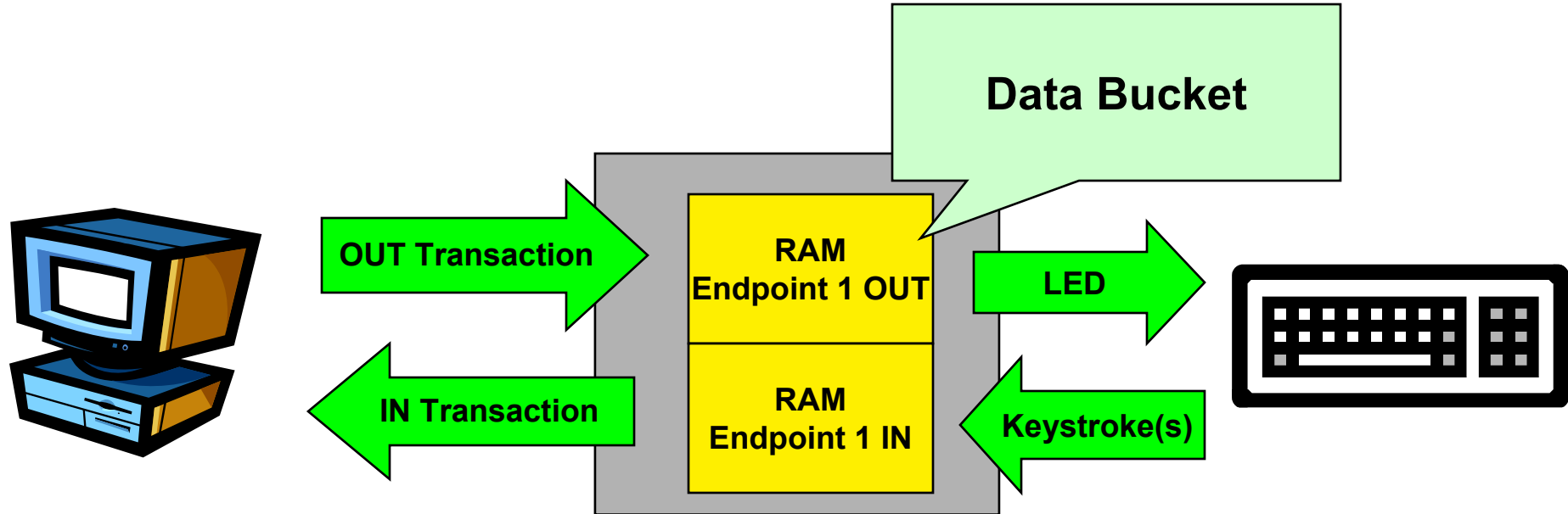
How do the host and the device communicate?

Transactions...

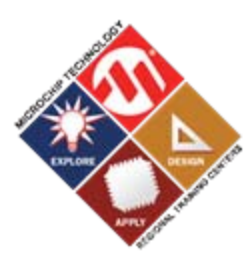




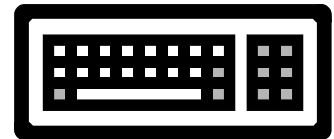
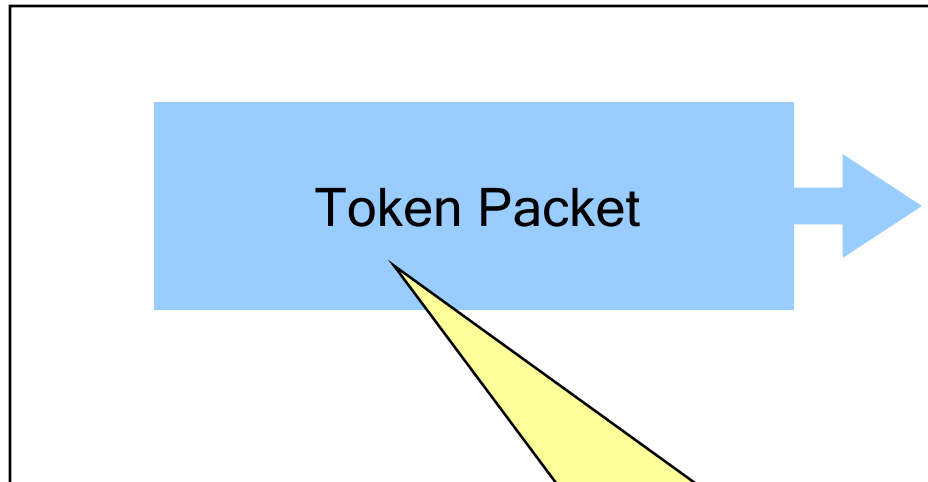
An Endpoint is a Buffer



- **Maximum number of endpoints per device specified by USB specification:**
 - 16 OUT endpoints + 16 IN endpoints = 32 endpoints
 - PIC18F87J50, PIC18F4550 supports up to 32 endpoints
- **EP0 = Default Communication Pipe**



USB Transaction

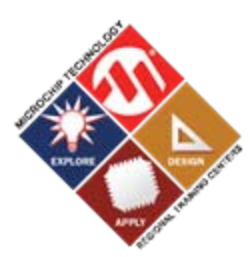


SETUP and OUT token types inform the target device that the host wants to send data.

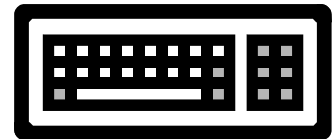
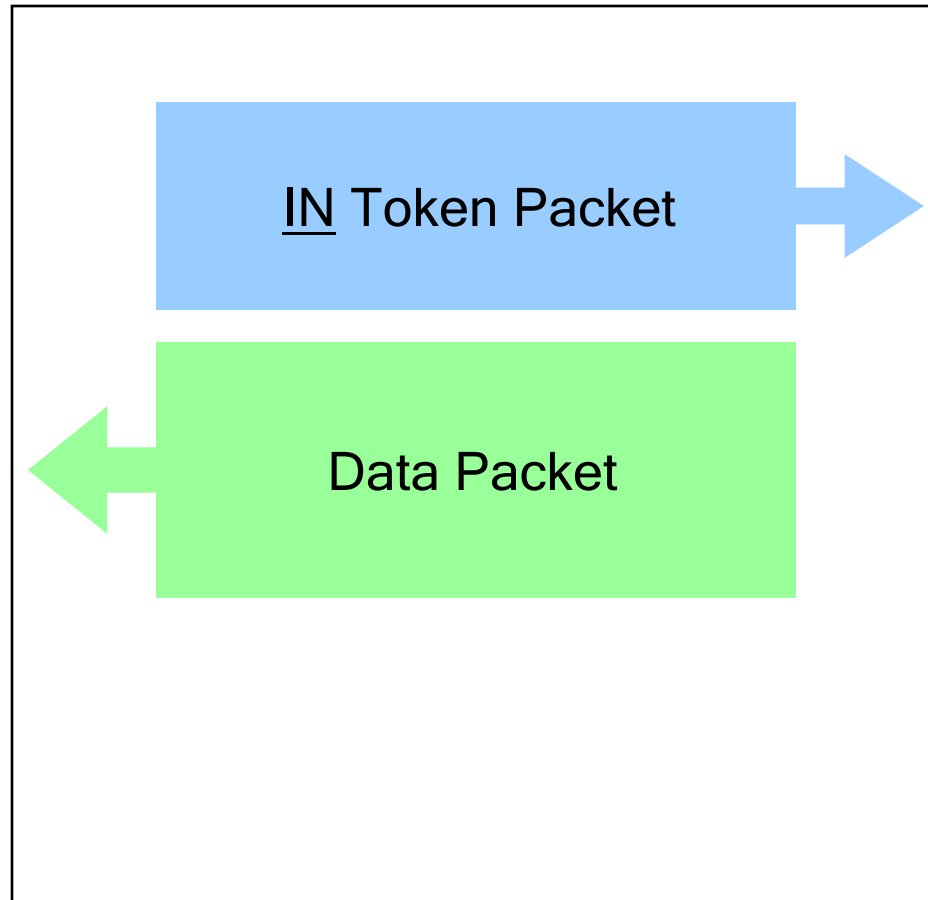
IN token type informs the target device that the host wants to fetch data.

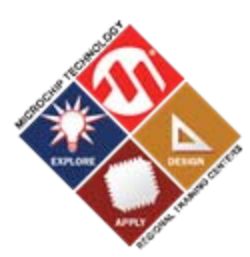
Specifies:

- Target device address
- Endpoint number
- Direction of the data transfer

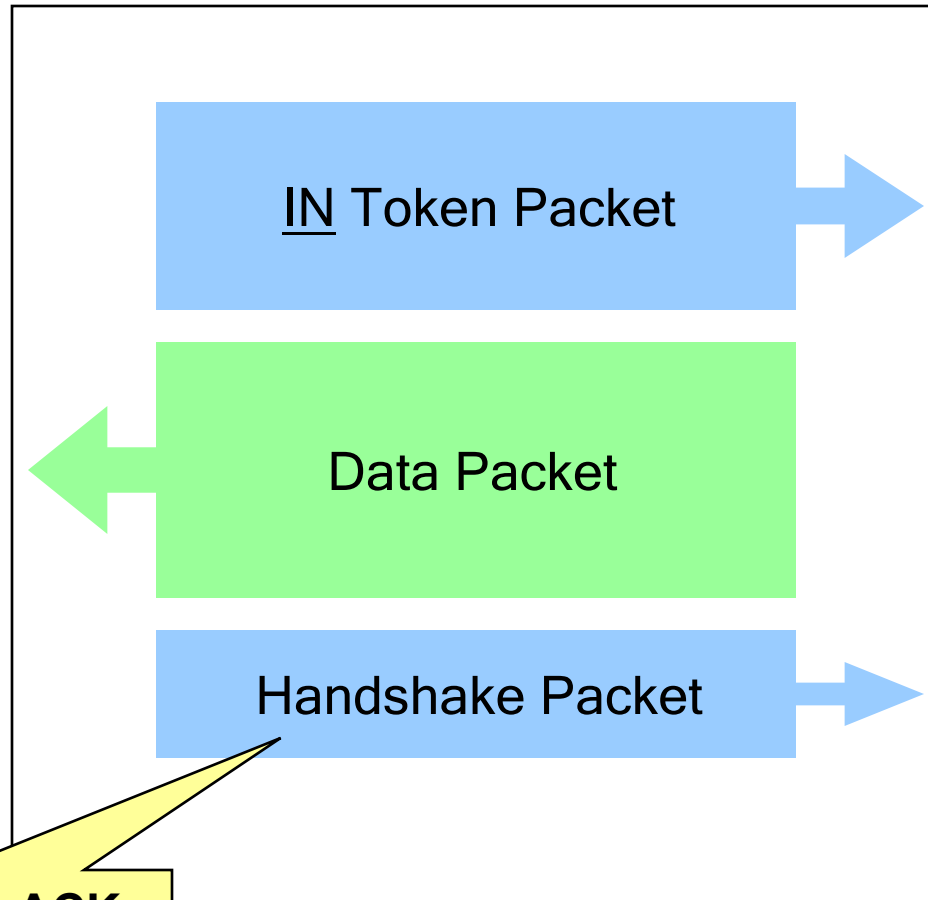


USB Transaction - IN

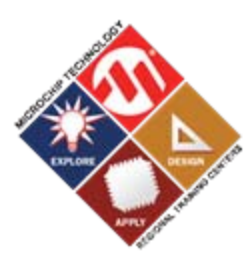




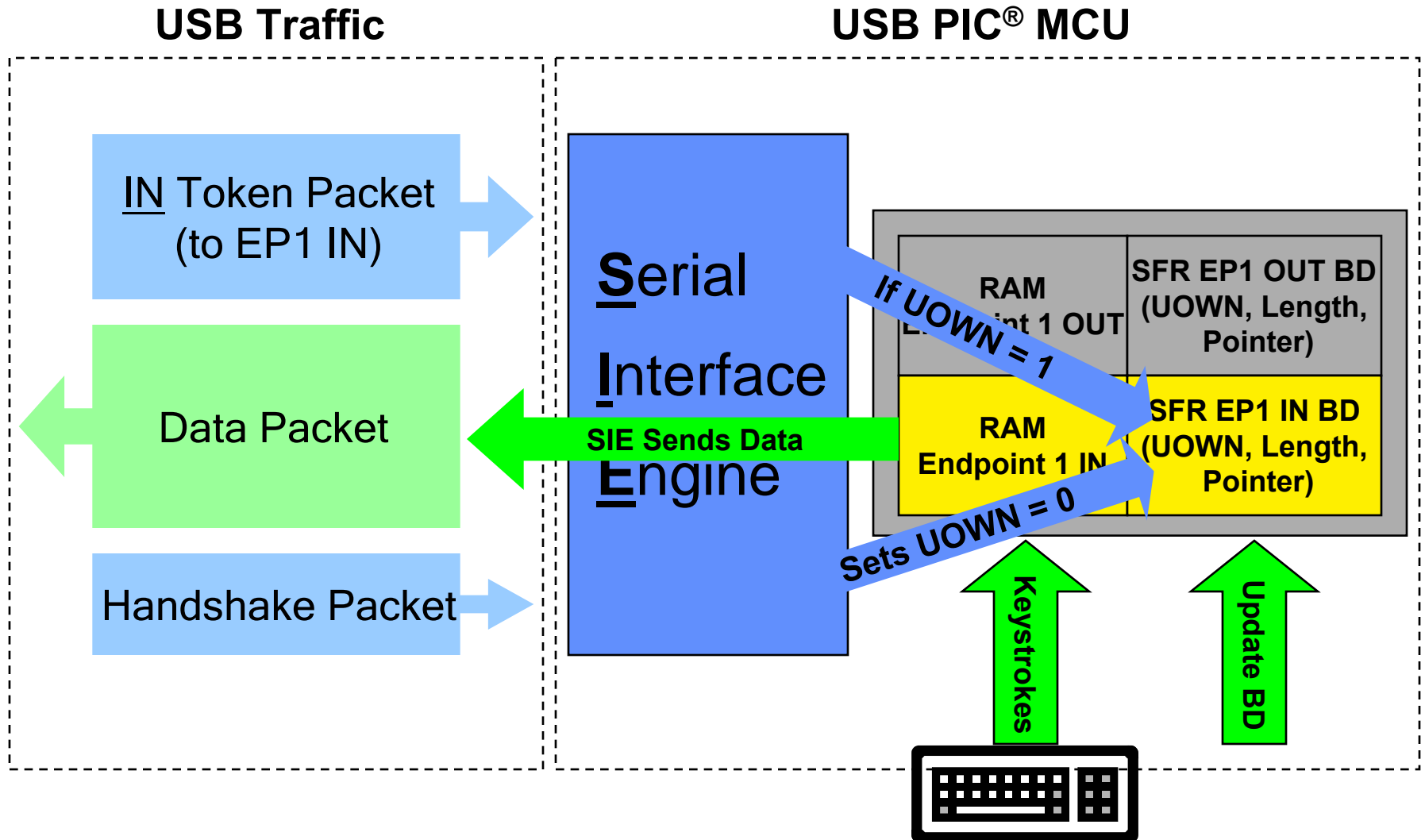
USB Transaction - IN

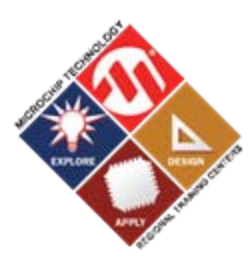


Acknowledge - ACK



IN Transaction ACK

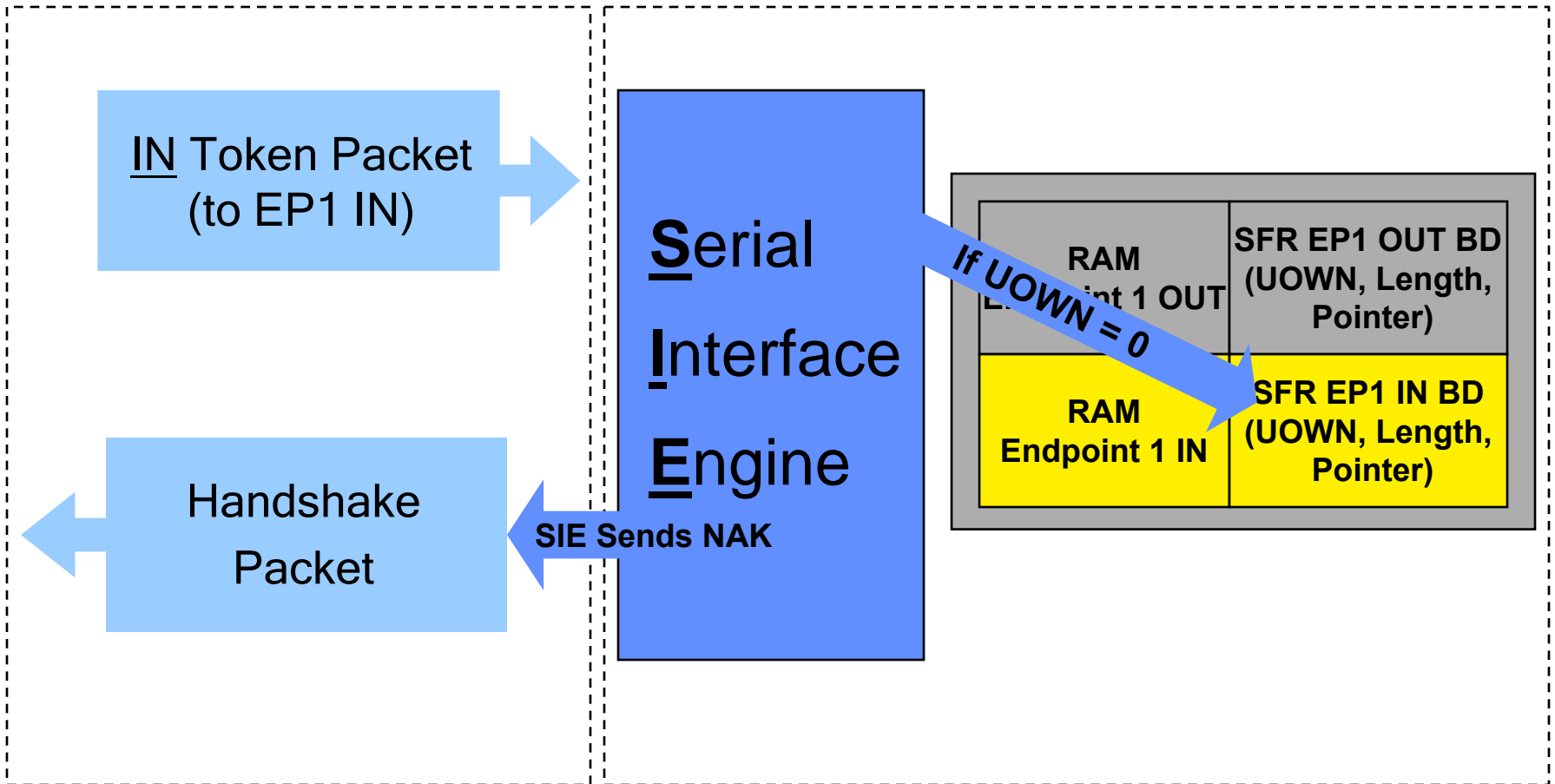


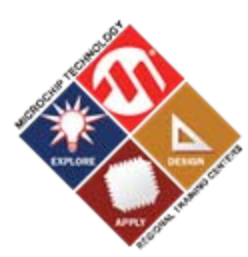


IN Transaction NAK

USB Traffic

USB PIC[®] MCU

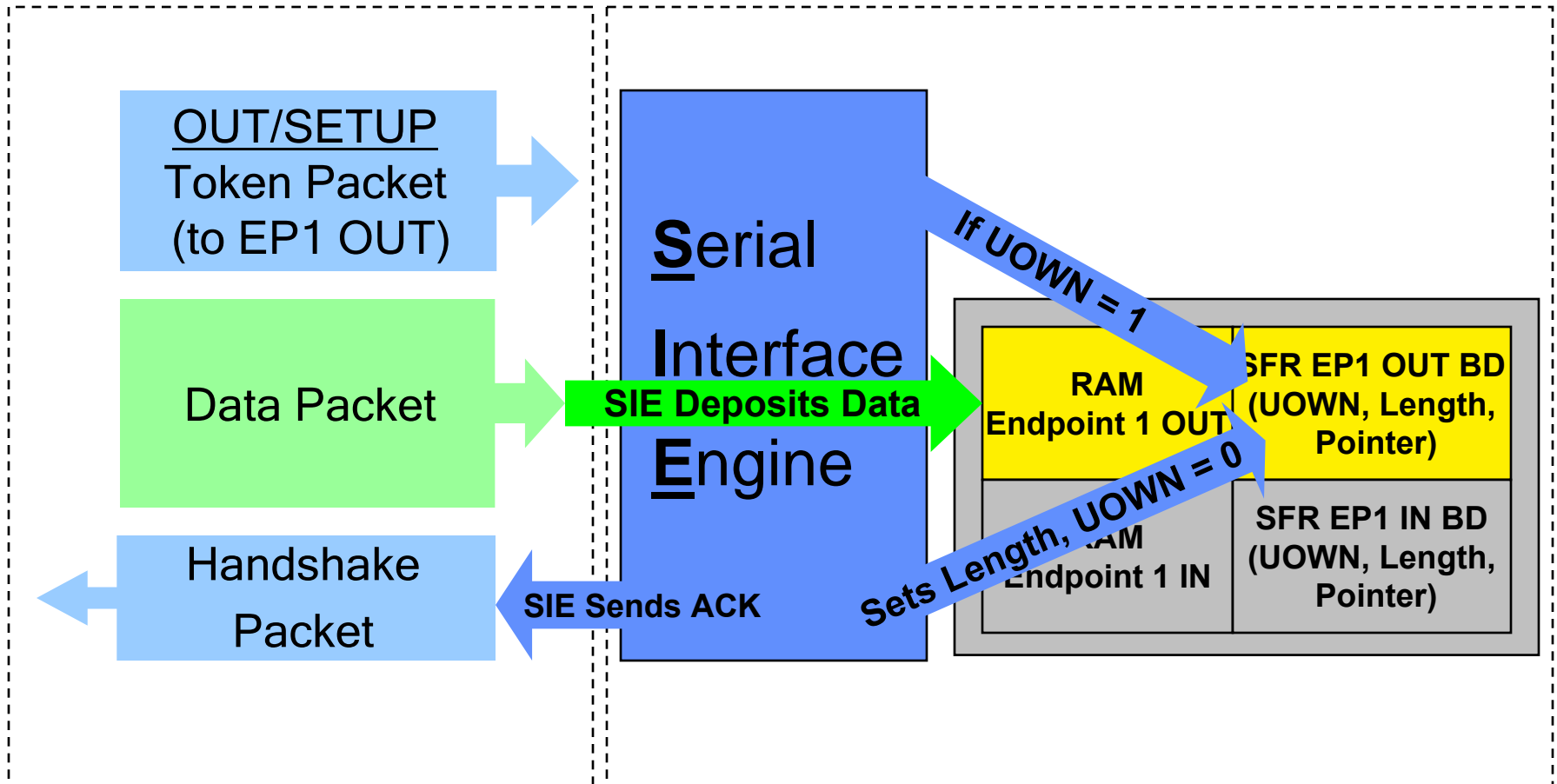


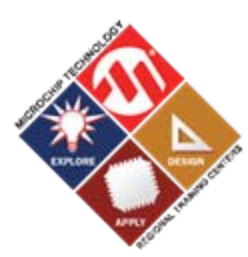


OUT/SETUP Transaction ACK

USB Traffic

USB PIC[®] MCU

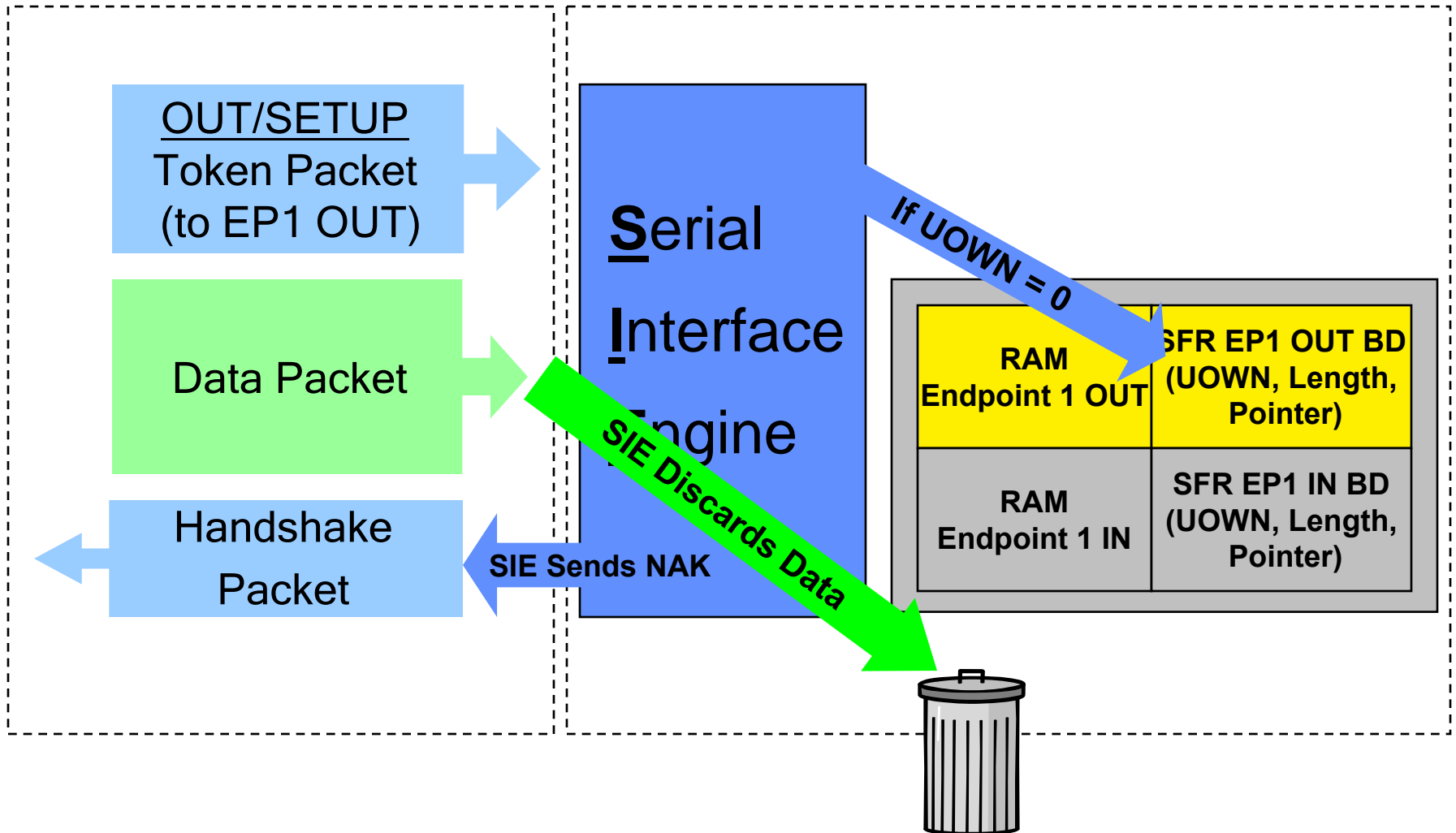


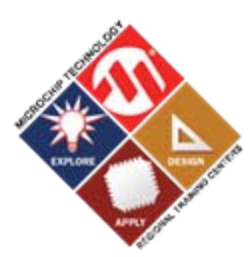


OUT/SETUP Transaction NAK

USB Traffic

USB PIC[®] MCU





Key: Token Types



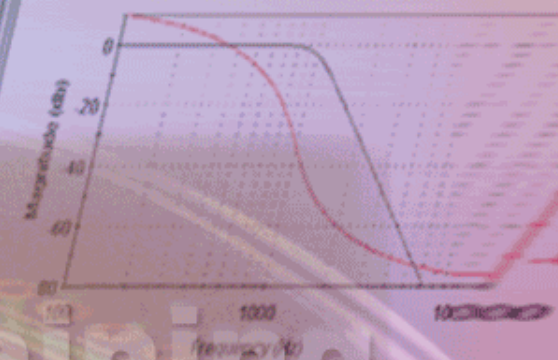
SETUP

OUT

IN

HANDS-ON

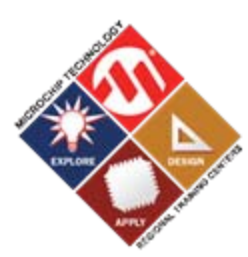
Training



How do the host and device communicate?

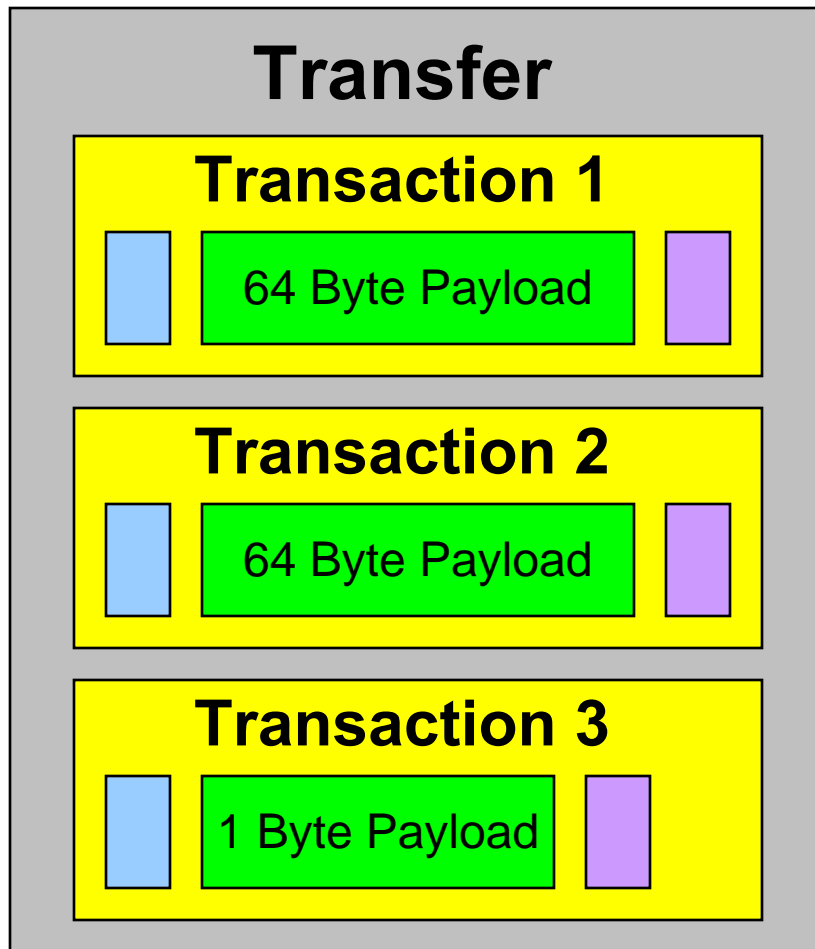
Transfers...



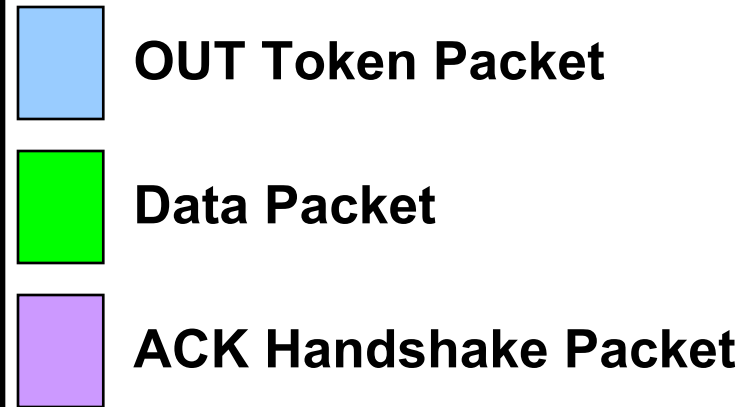


Transfers Vs. Transactions

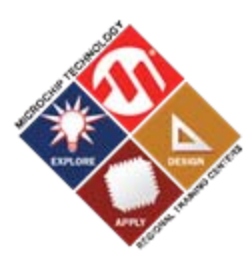
MPUSBWrite(EP7, Pointer, Size = 129, Timeout)



Key:

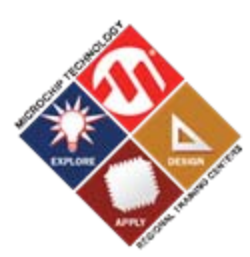


Transfer: Group of related transactions.



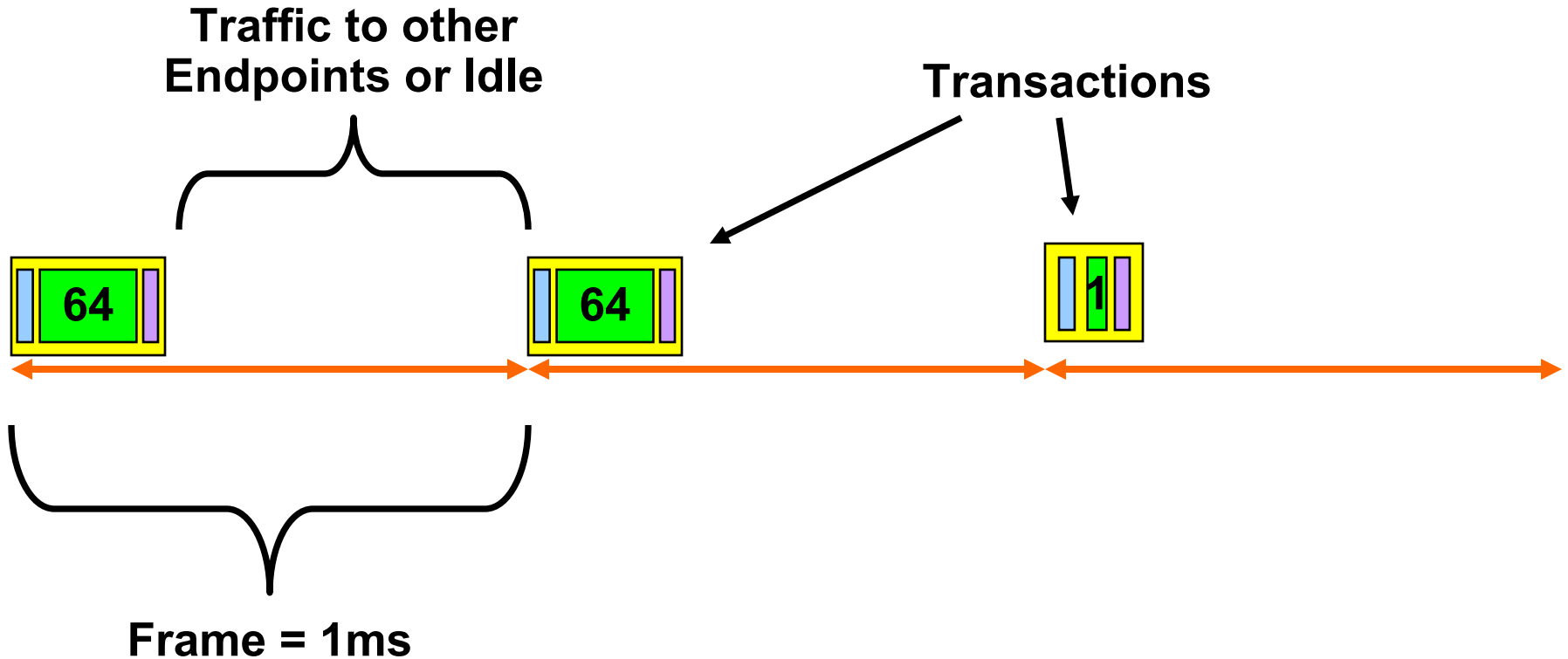
Data Transfer Types

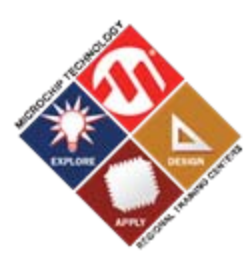
<u>Transfer/ Endpoint Type</u>	<u>Polling Interval</u>	<u>Priority</u>	<u>Guarantees</u>
Interrupt	Fixed, Periodic	High	64 Bytes/Period, Data Integrity
Isochronous	Fixed, Periodic	High	1023 Bytes/Period
Bulk	Variable, Uses Free Bandwidth	Low	Data Integrity
Control	Variable	Medium	Some Bandwidth, Data Integrity



Interrupt Transfer Example

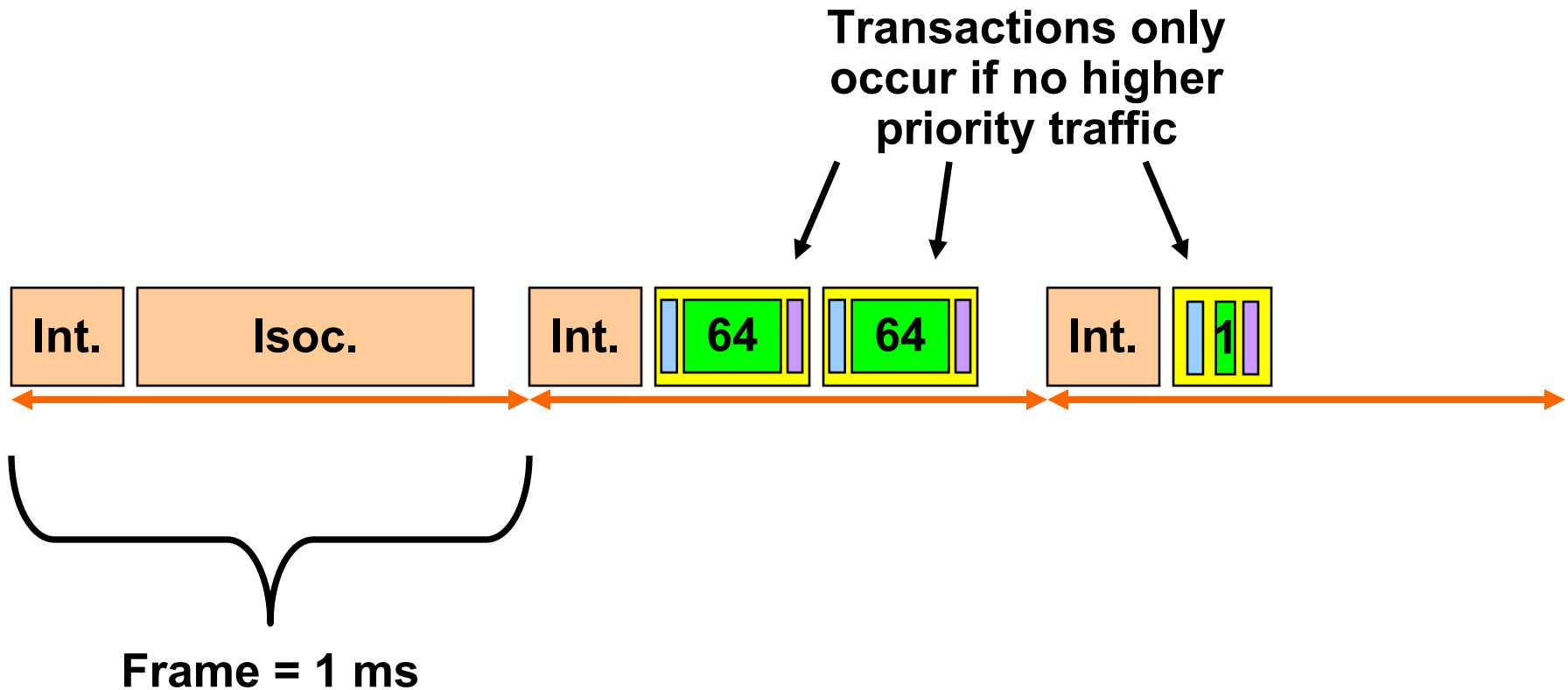
MPUSBWrite(EP7, Pointer, Size = 129, Timeout)

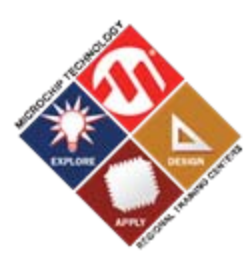




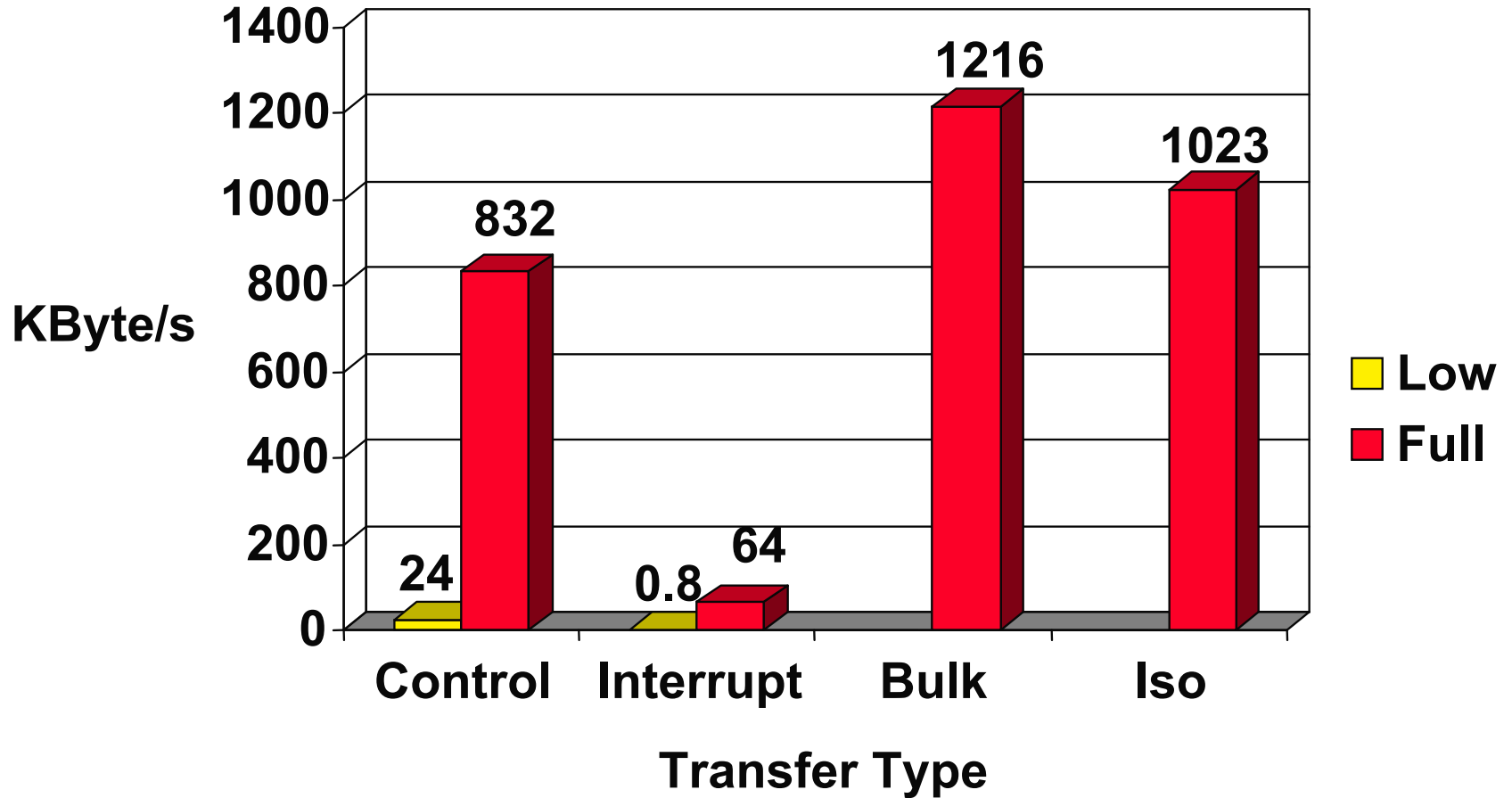
Bulk Transfer Example

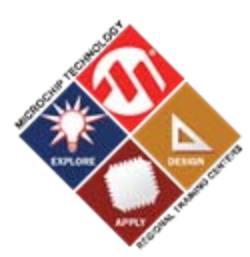
MPUSBWrite(EP7, Pointer, Size = 129, Timeout)



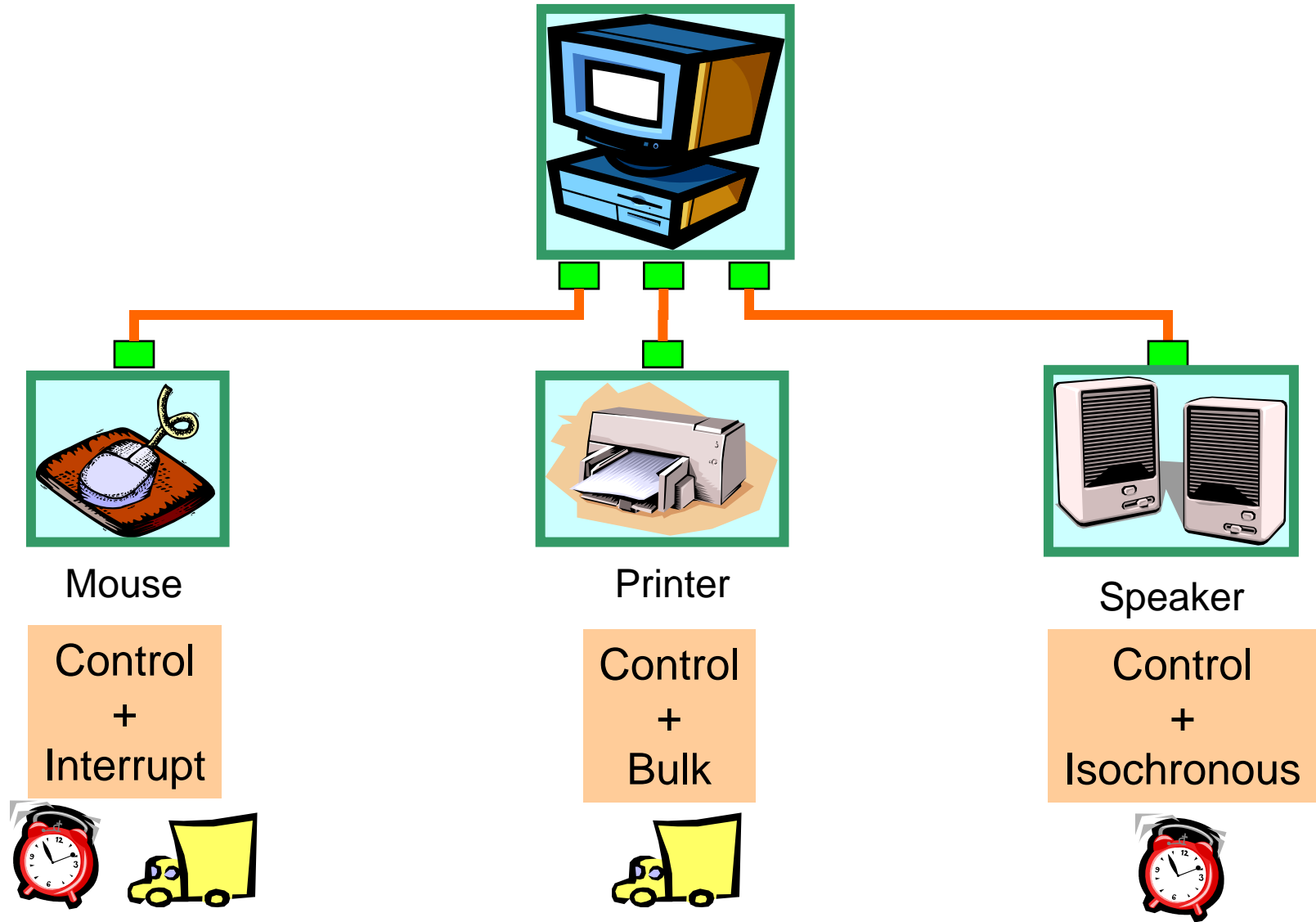


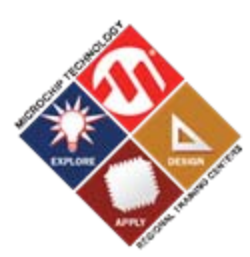
Theoretical Maximum Transfer Rate Per Endpoint



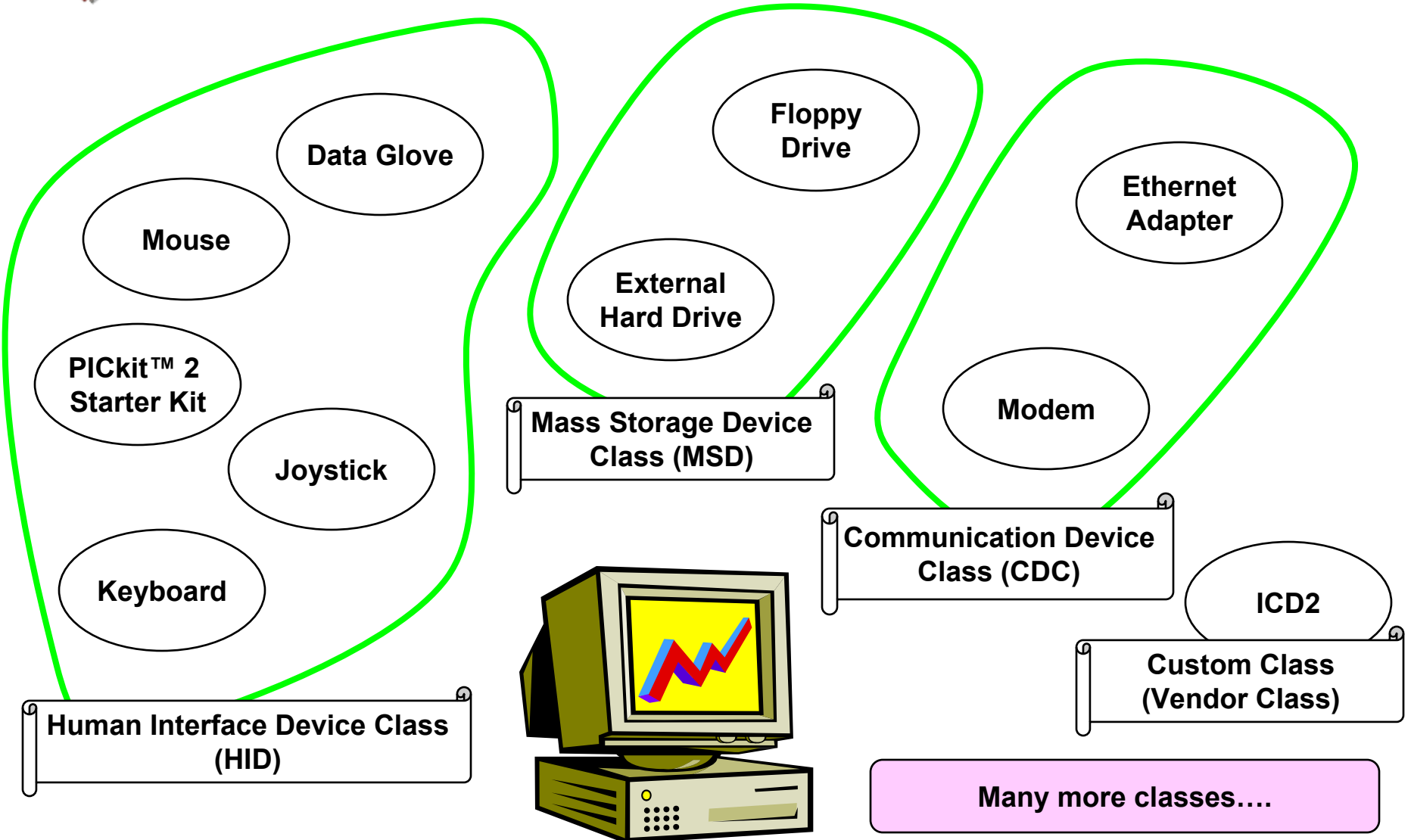


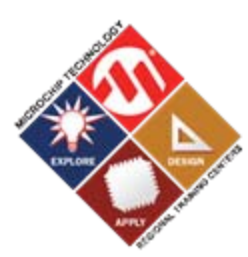
Transfer Types - Examples





USB Device Classes





Agenda

- **Brief history of USB & USB-IF**
- **USB Fundamentals - The serious & important stuff**
 - Basics/Speeds
 - Architecture/Programmer's Model
 - Physical Connection
 - USB Transactions
 - USB Transfers
 - Device Classes
 - – **Enumeration**
 - **Descriptors**
 - **Power Planning**
 - **VID/PID & USB Compliance**
- **PIC18F USB Microcontrollers**
- **Microchip Demo/Development Solutions**

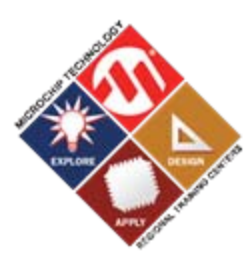
HANDS-ON

Training

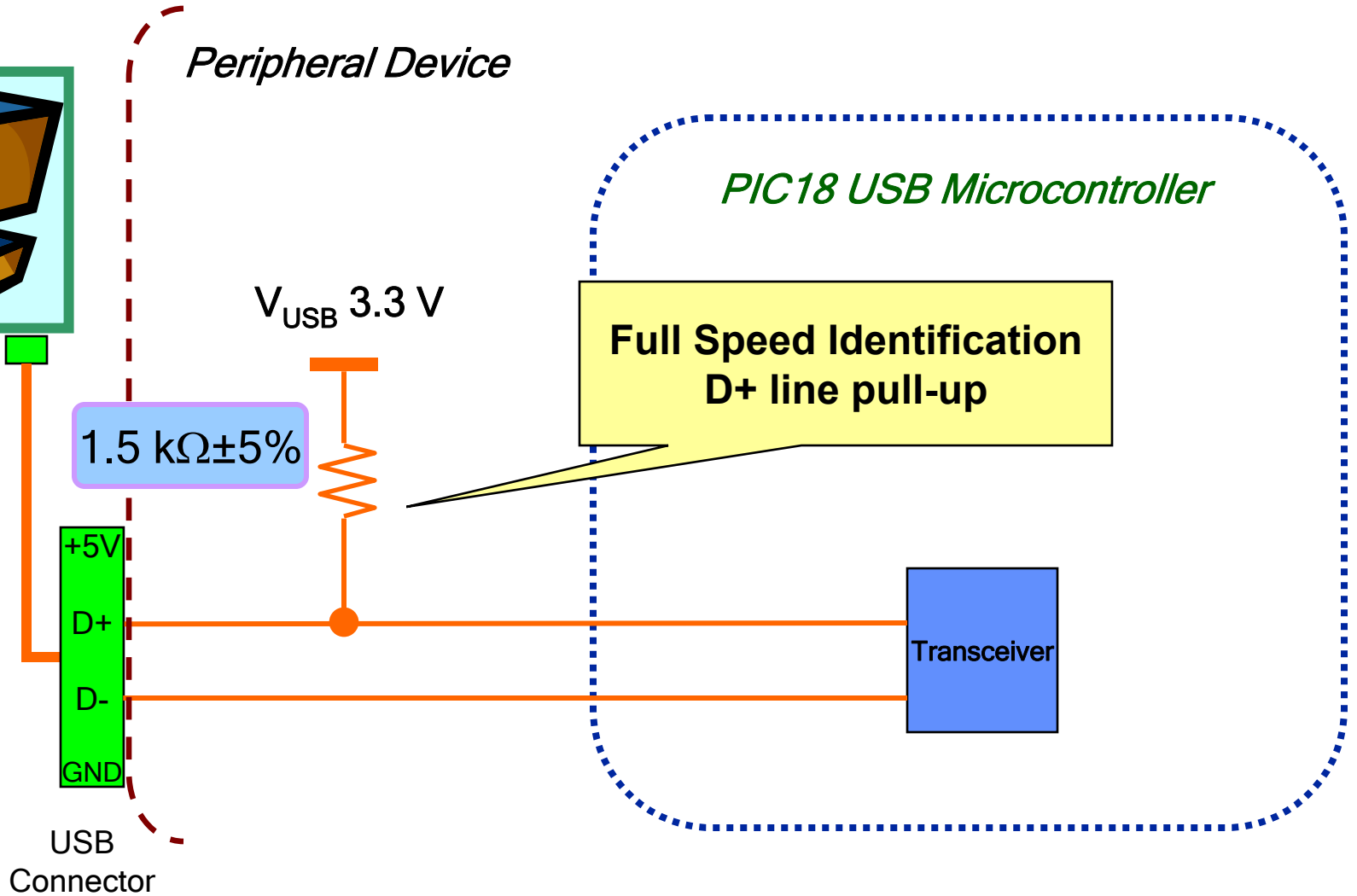
Enumeration and the magic behind “Plug&Play”

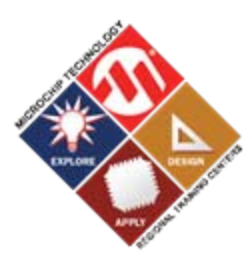
Auto-Detection & Auto-Configuration





Auto-Detection: Full-Speed



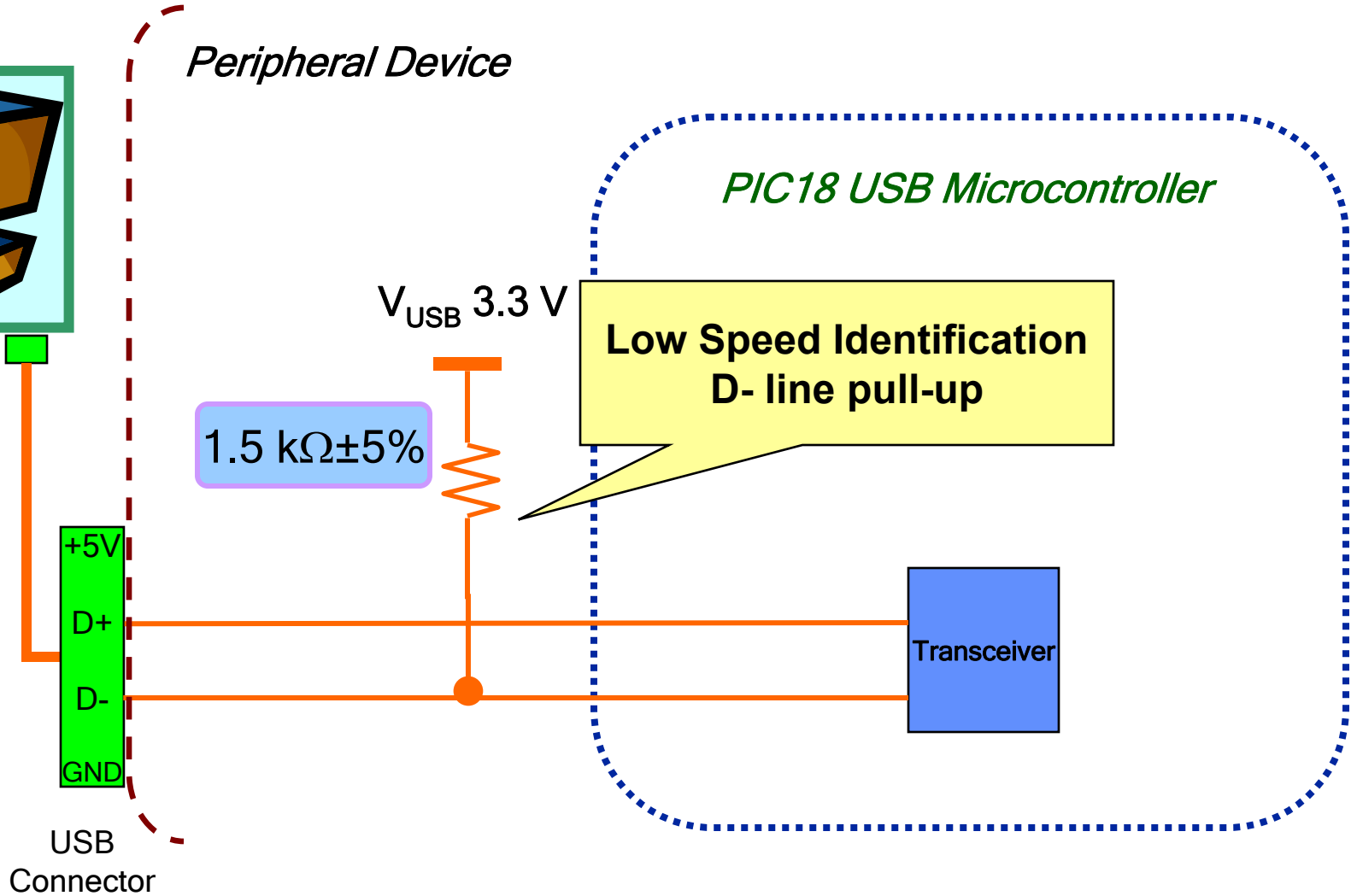


Auto-Detection: Low-Speed



Peripheral Device

PIC18 USB Microcontroller



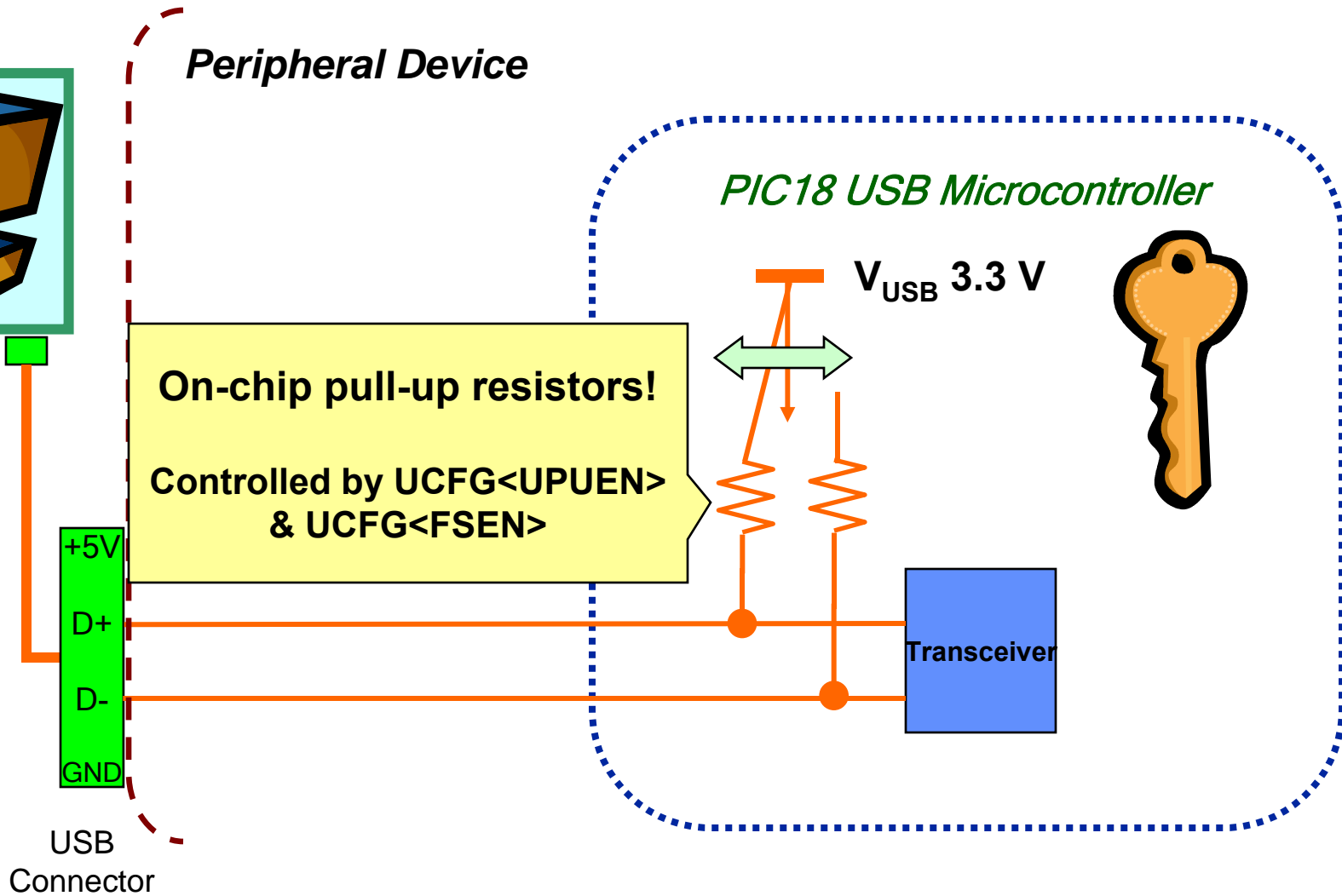


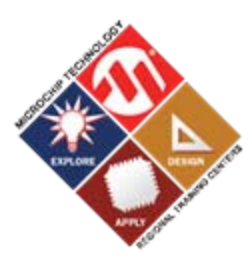
On-chip Pull-up Resistors



Peripheral Device

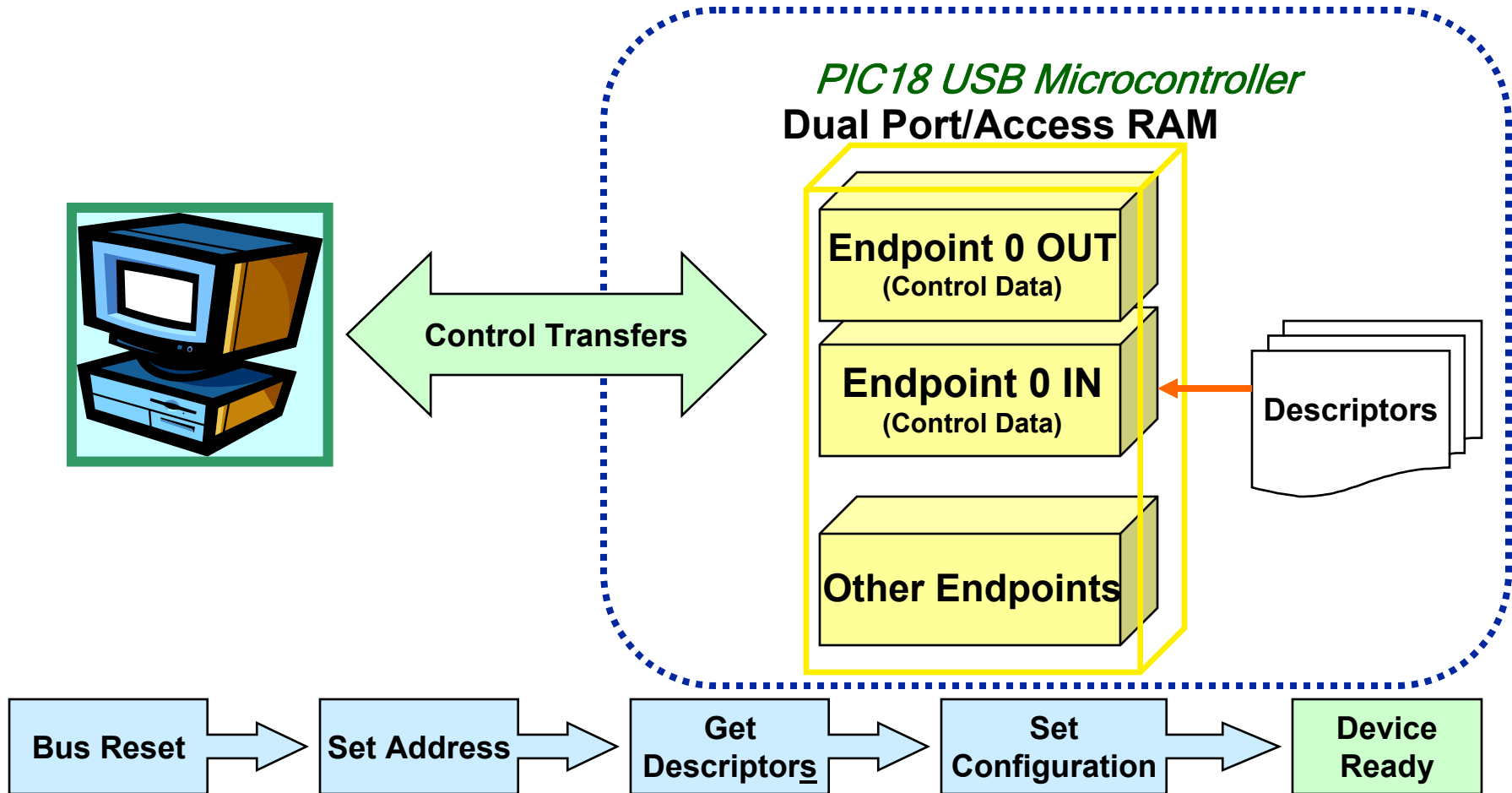
PIC18 USB Microcontroller

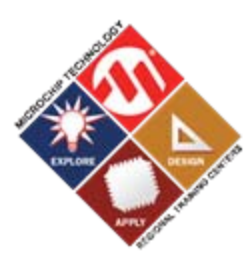




Endpoint 0 and Enumeration

- See Chapter 9 in USB 2.0 Spec for more info.



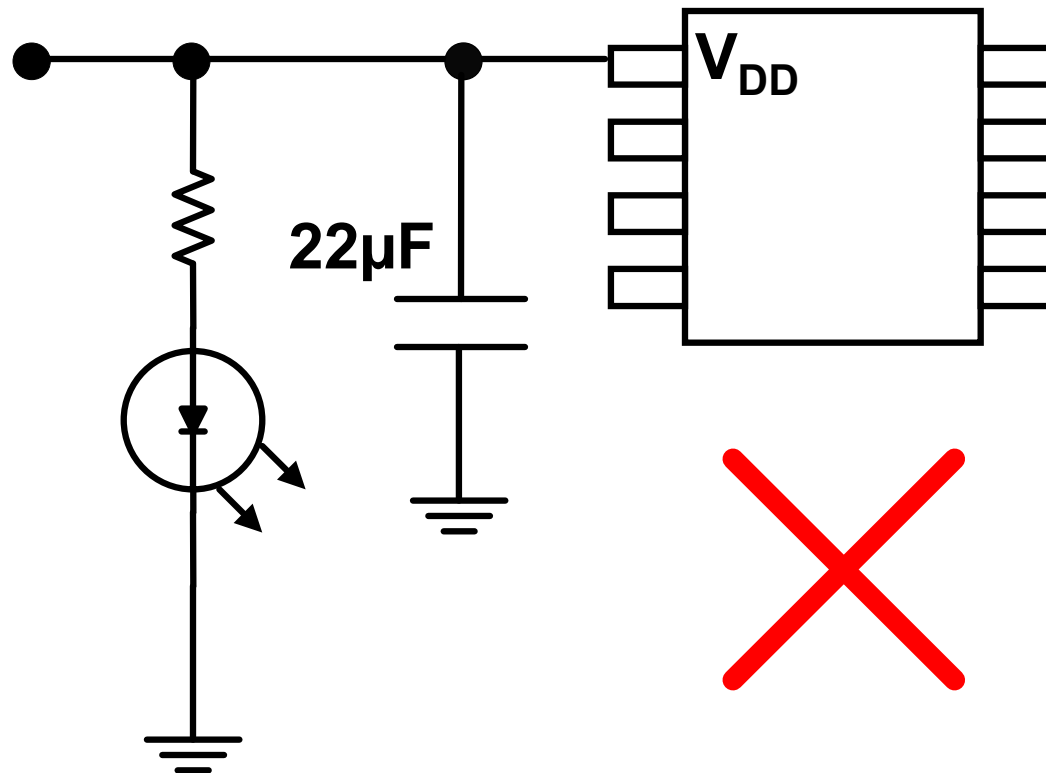


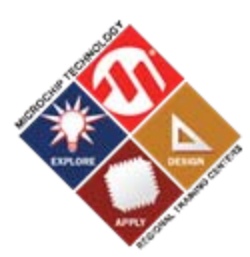
Power Planning

- Max USB suspend current is 0.5/2.5mA

- Don't:

Power from
USB Cable



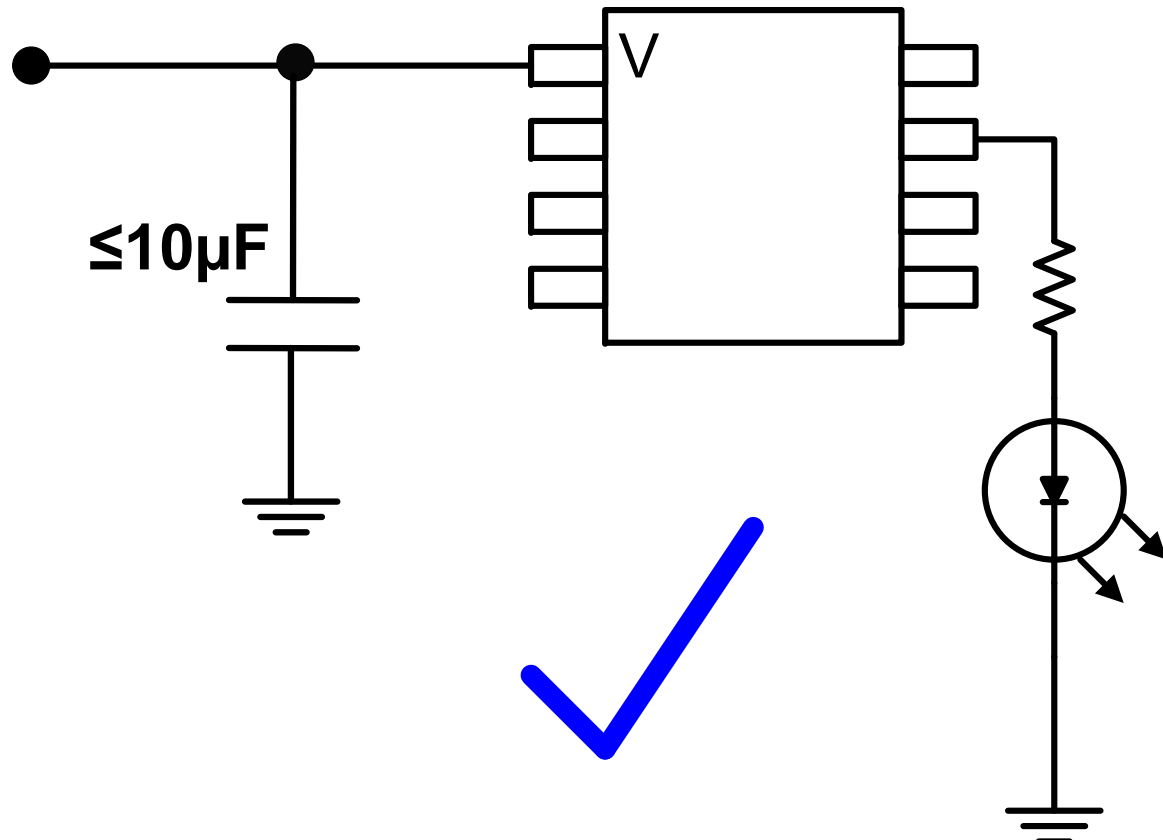


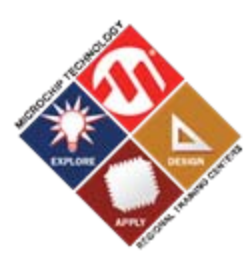
Power Planning

- **Max USB suspend current is 0.5/2.5mA**

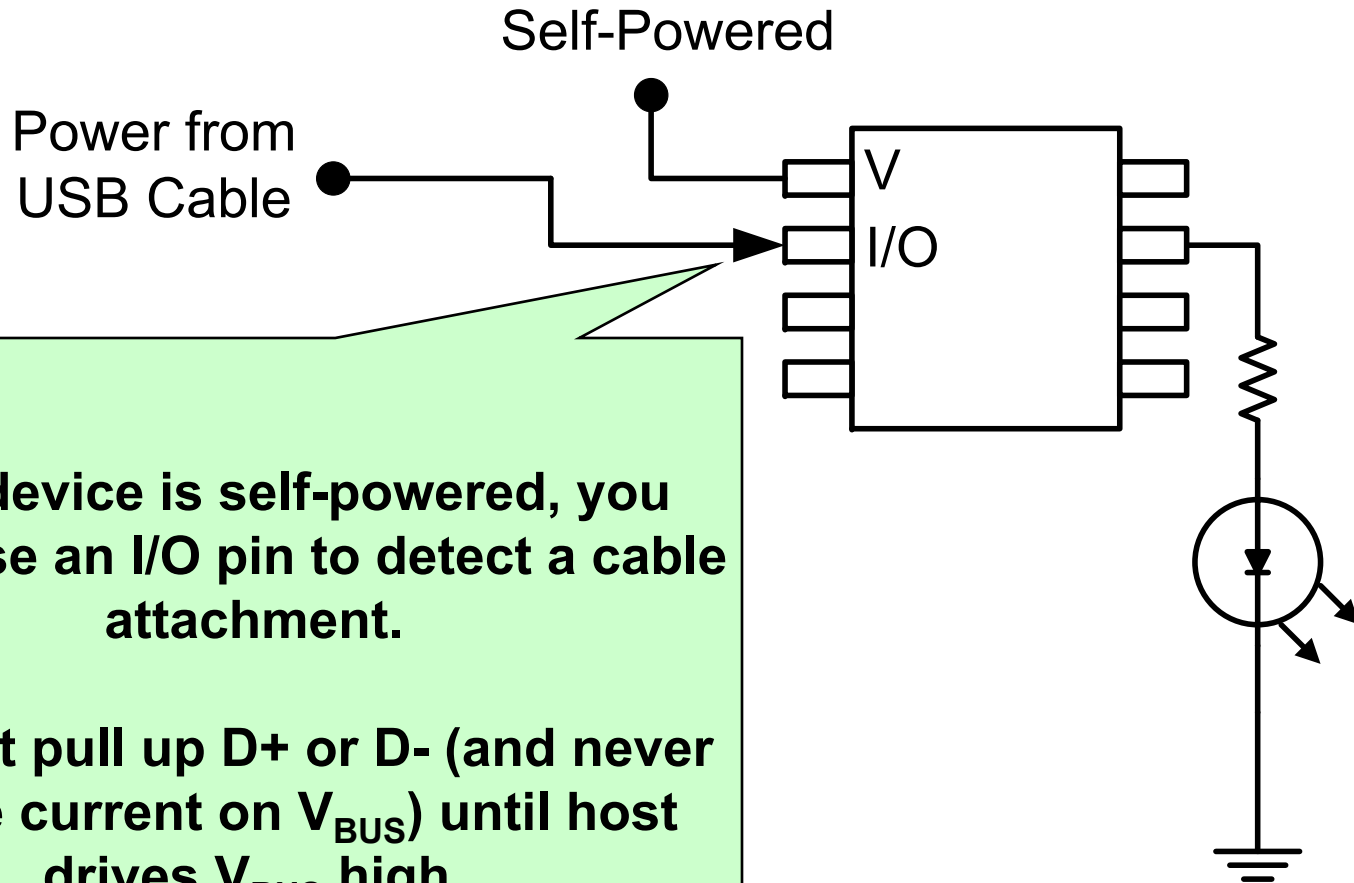
- **Do:**

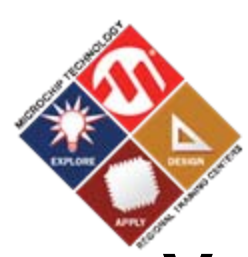
Power from
USB Cable





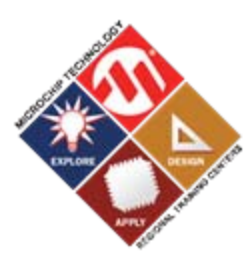
PIC[®] MCU: Detecting a USB Attachment





VID & PID

- **Vendor ID (VID)** *16-bit number*
 - Required to market your product
 - <http://www.usb.org/developers/vendor/>
 - USD \$2,000
 - Technical & Legal trouble if not using your own VID
- **Product ID (PID)** *16-bit number*
 - Microchip's Sub-licensing Program
- **Every product line is required to have a unique combination of VID and PID**



USB Compliance

- **Compliance Testing**

- Must pass to use USB logo
- USD ~\$1,500

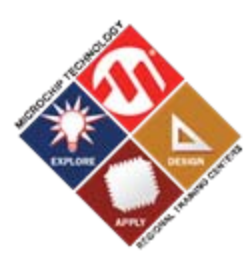


- **Ch9 and other USB Firmware**

- USB Protocol Analyzer
- “USBCV” www.usb.org/developers/tools/

- **Electrical Signal Quality**

- **Power Management**



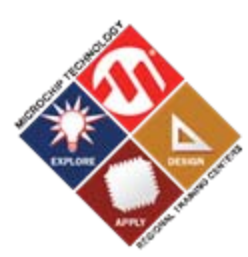
Compliance Testing

- **For USB Compliance: Independent Test Labs**
- **For Device ‘Sanity Check’: USB “Plugfest”**

- **For USB Compliance Testing:**
 - Must submit a compliance checklist
 - www.usb.org/developers/compliance/peripheral_low/
 - Download “Peripheral Checklist”

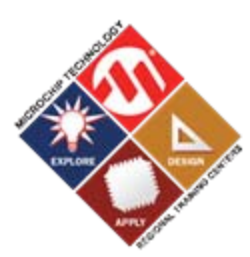
 - TID: Test ID
 - Use certified USB receptacle and cable for testing
 - Know the TID of your components
 - All USB PIC[®] MCUs have a TID number. Get it at www.microchip.com/usb

- **Probably a good idea to take a look at the checklist even before starting your design!**

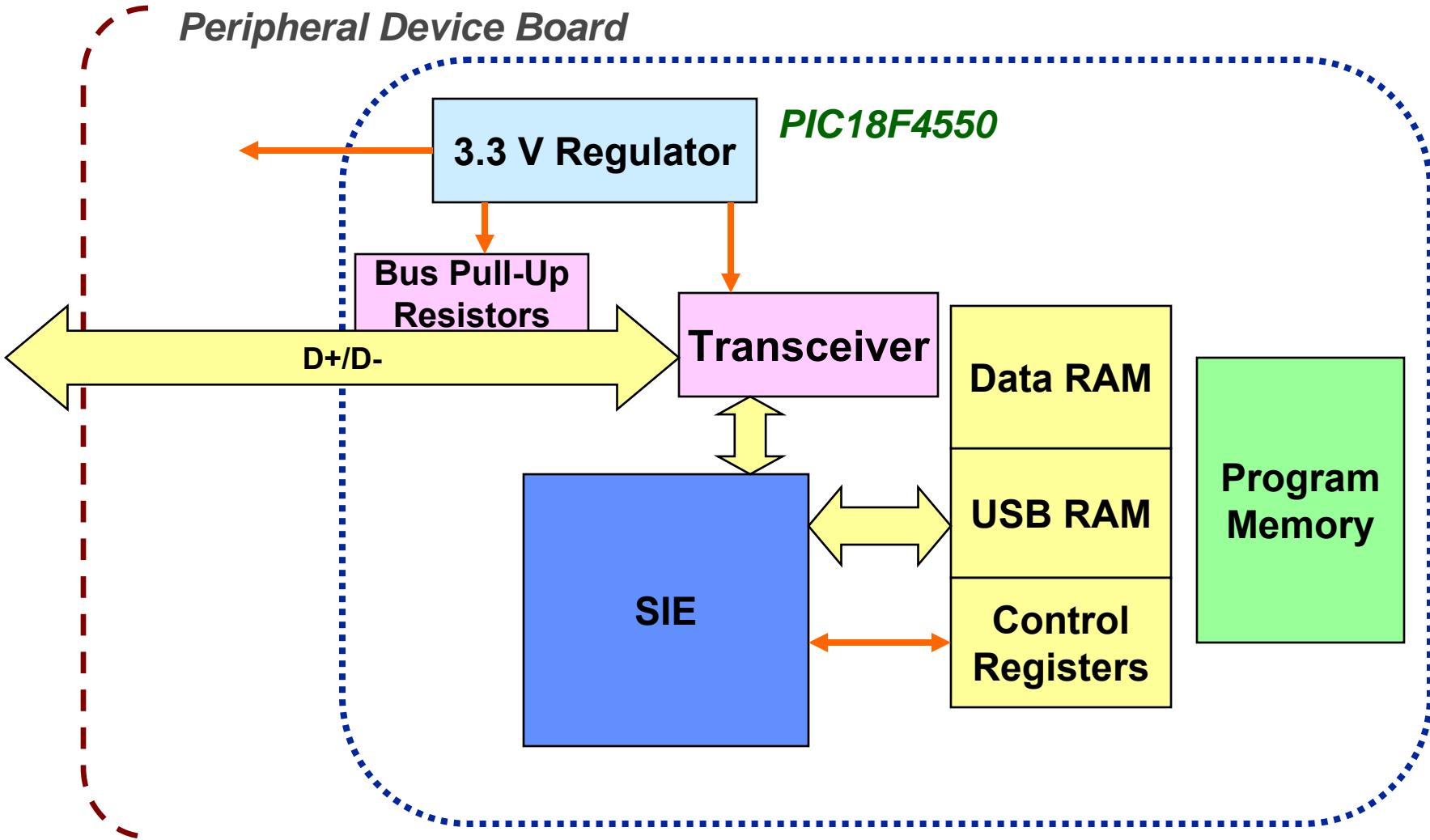


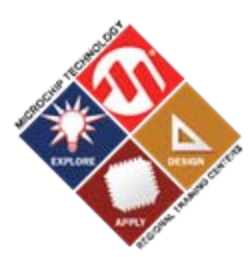
Agenda

- **Brief history of USB & USB-IF**
- **USB Fundamentals - The serious & important stuff**
 - Basics/Speeds
 - Architecture/Programmer's Model
 - Physical Connection
 - USB Transactions
 - USB Transfers
 - Device Classes
 - Enumeration
 - Descriptors
 - Power Planning
 - VID/PID & USB Compliance
- **PIC18F USB Microcontrollers**
- **Microchip Demo/Development Solutions**



USB Module



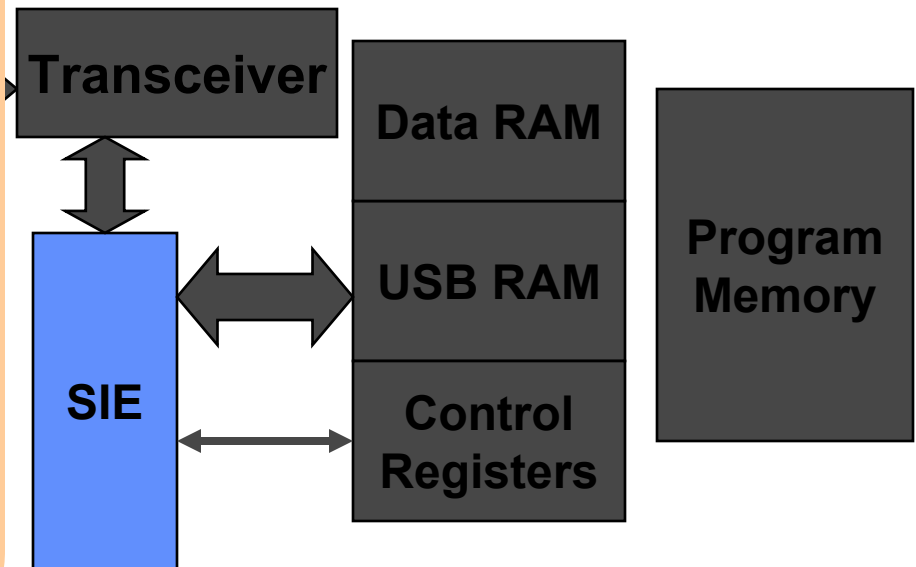


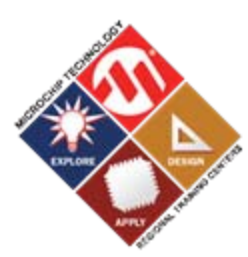
Serial Interface Engine

SIE ...

- Serializes and deserializes USB data
- Encodes and decodes NRZI data
- Handles bit stuffing
- Checks CRC to validate data packet
- Detects bus signaling events and notifies the CPU through interrupts
- Handles USB transactions
- Handles handshaking protocol

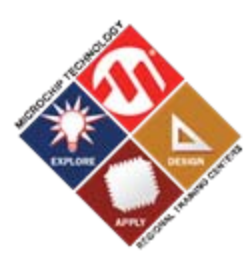
PIC18 USB MCU





Agenda

- **Brief history of USB & USB-IF**
- **USB Fundamentals - The serious & important stuff**
 - Basics/Speeds
 - Architecture/Programmer's Model
 - Physical Connection
 - USB Transactions
 - USB Transfers
 - Device Classes
 - Enumeration
 - Descriptors
 - Power Planning
 - VID/PID & USB Compliance
- **PIC18F USB Microcontrollers**
- **Microchip Demo/Development Solutions**

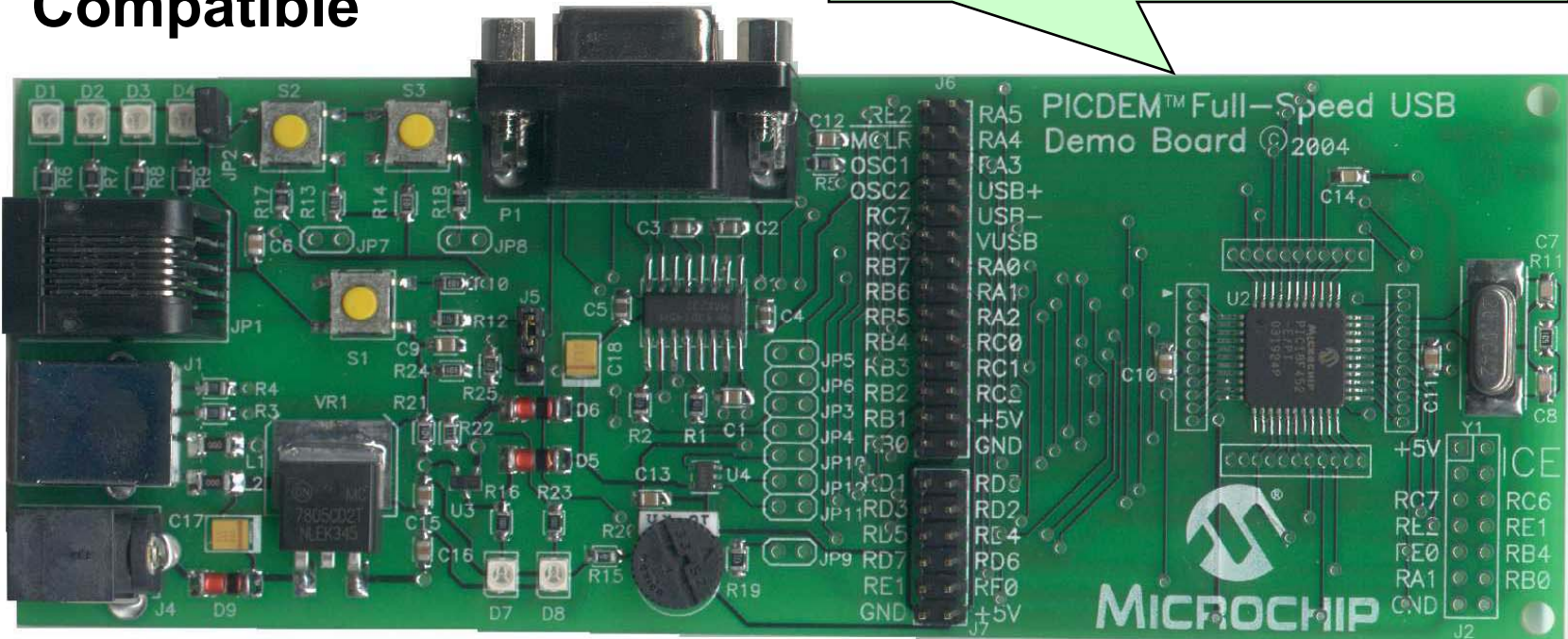


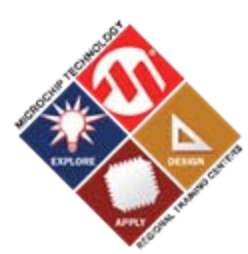
PICDEM™ Full-Speed USB

- PIC18F4550, 20 MHz xtal
- USB Port
- Serial Port
- PICtail™ Daughter Board Compatible
- Power LED Indicators
- Potentiometer
- Temperature Sensor



Part Number - DM163025



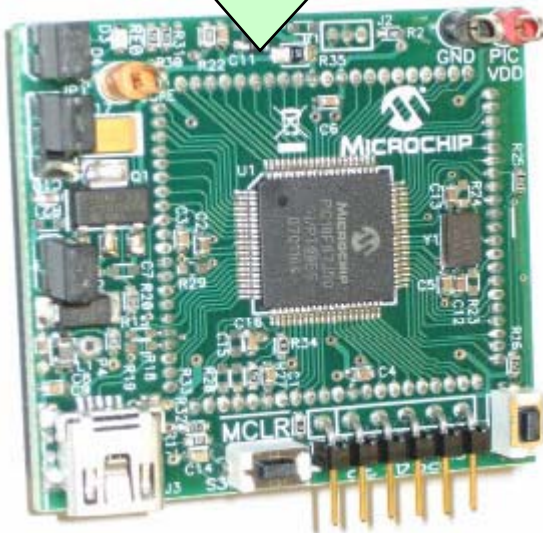


PIC18F87J50 FS USB Plug-In Module (PIM)

- PIC18F87J50, 12 MHz xtal
- USB Port, mini-B
- Linear Regulator
- 2 LEDs, 2 Pushbuttons
- Stand alone operation
- Operation with: HPC Explorer Demo Board
- ICSP™ Technology: 6-pin header

Part Number – MA180021

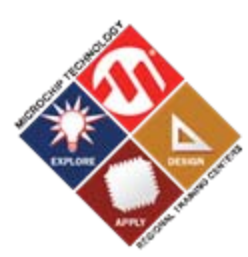
HPC Explorer – DM183022



Part Number – AC164110



PICDEM™ HPC Explorer Board

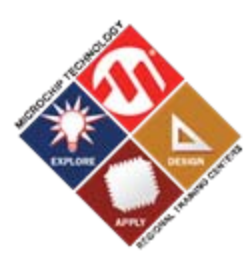


PICtail™ Daughter Board for SD and MMC

- **Secure Digital Card Interface - using SPI**
- **AN1003: USB Mass Storage**

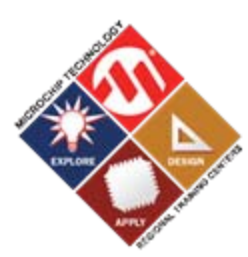
Part Number – AC164122





Microchip USB Firmware Characteristics

- **C18 Compatible**
- **MPLAB[®] IDE Project Centric**
- **Polling Scheme**
- **Cooperative Multi-Tasking (No Blocking Routines)**
- **Program Memory Usage**
 - USB Enumeration (Chapter 9) - 3 KB
 - HID - 1 KB
 - CDC (RS-232 Emulation) - 1 KB
 - USB Mass Storage - 4 KB



Summary

- **Basics/Architecture**
 - Up to 126 devices sharing bandwidth
 - Host is master
- **Host/Device Communication**
 - Transactions
 - Transfers
- **Enumeration/Chapter 9**
 - Descriptors
- **Microchip Offerings: MCUs, Demo Boards, Firmware, Custom Driver**

HANDS-ON

Training

Windows[®] Programming



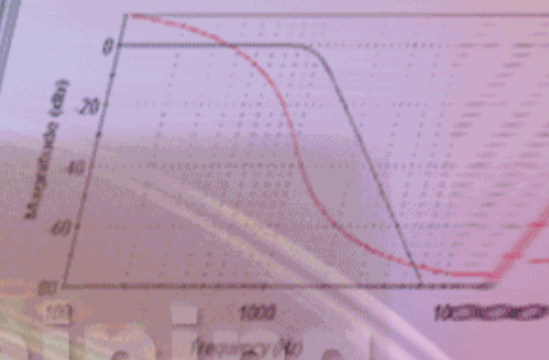


Windows[®] Programming

- **With Visual C++ 2005 you have three basic ways of creating a interactive, graphical application:**
 - Using the **Windows API (Win32)**
 - Most programming intensive
 - Using the **Microsoft Foundation Classes (MFC)**,
 - Encapsulate the Win32 API - easier
 - Using **Windows Forms (.NET Framework)**
 - Least programming intensive
- **Visual C++ 2005 also allows you to create two types of console application (Win32 and CLR (.NET))**
 - We will be developing a “Win32 Console” C application
 - Why? Simplicity & Desire to focus on the DLL functions

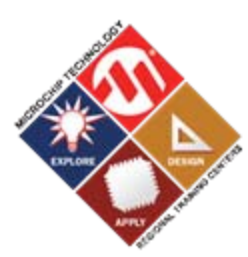
HANDS-ON

Training



USB OTG “On The Go”





Types of USB

Supported

Not Supported

PIC24FJ256GB1

**On the Go
(Master & Slave)**

IN DEVELOPMENT

**Mini-Host
(Thumb Drives)**

IN DEVELOPMENT



NOW

**Low Speed
1.2 Mbps**

Device/Slave/Node

NOW

**Full Speed
12 Mbps**

**Ideal for Embedded Market –
Serial Port Replacement**

Other USB PIC microcontrollers

Hub

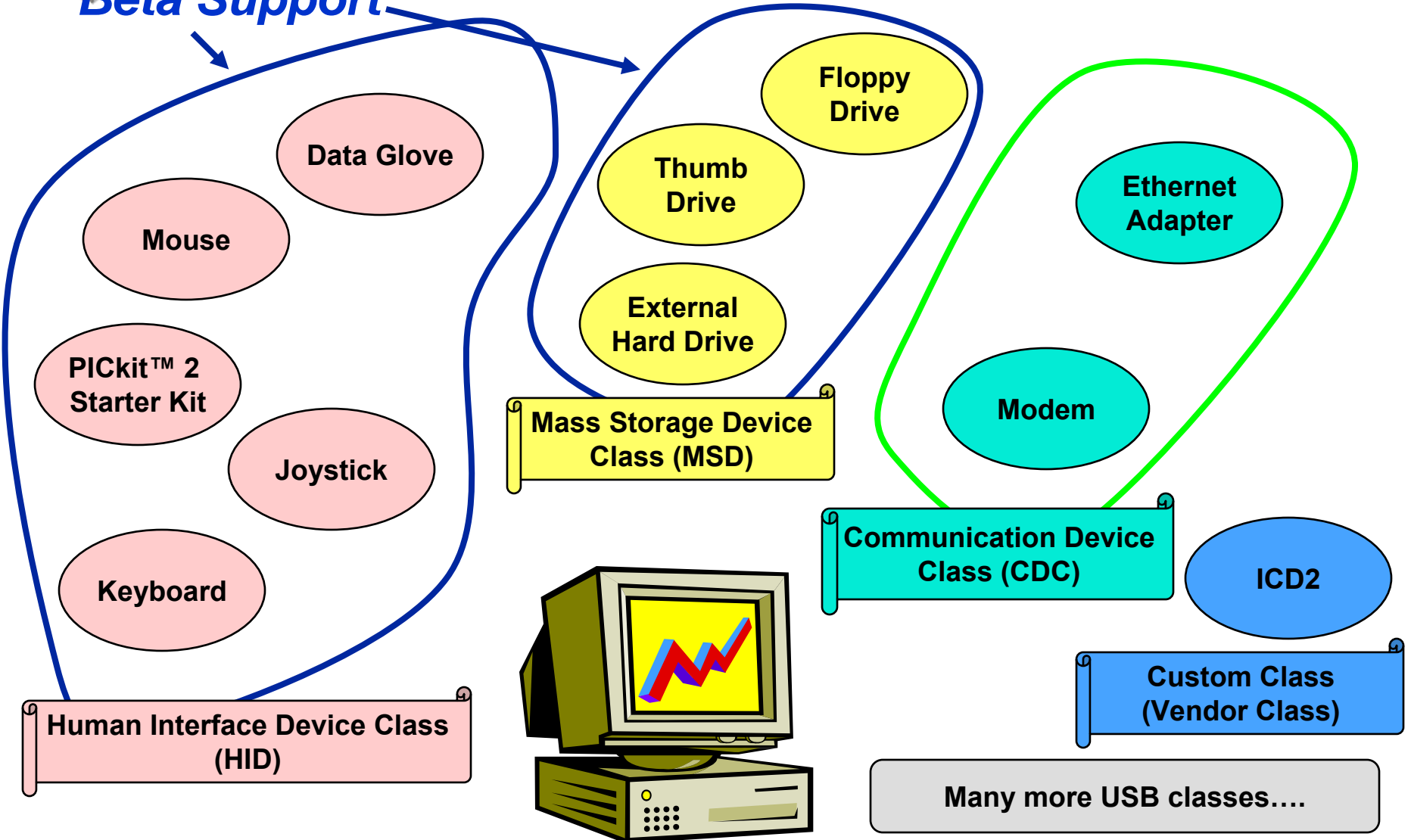
**Not Targets for
Embedded
Market**

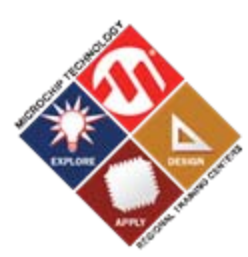
**High Speed
480 Mbps**



USB Device Classes

Beta Support





Mass Storage Application Software Architecture for Thumb Drive Demo

Application (“PIC-DOS”)

FAT16

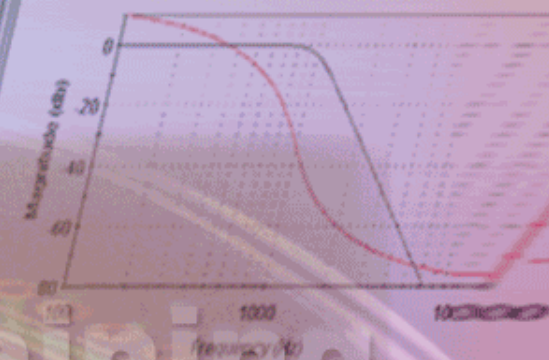
SCSI Interface

USB Mass Storage Class (Host)

USB Host

HANDS-ON

Training



Thank you !

