More on IO:
The Universal Serial Bus (USB)
Adminstrivia

- Project 5 is: USB Programming (read from a mouse)
- Optional (helps you to catch up on points)

- Slide credit: Omid Fatemi, NUI Galway
Today’s Lecture

• Generic Interfaces beyond GPIO
• USB concepts
Data Communication Channels

• **Simplex**
  - Simplex channel is always one way. Simplex channels are not often used because it is not possible to send back error or control signals to the transmit end. Example: Keyboard, individual GPIO pins.

• **Half Duplex**
  - Half duplex channel is two way, but not at the same time. Example: Old sound cards, old serial busses.

• **Full Duplex**
  - Data can travel in both directions simultaneously. Example: USB
Parallel and Serial Data

- Parallel transmission (e.g. 8 bits)
  - Each bit uses a separate wire
  - To transfer data on a parallel link, a separate line is used as a clock signal. This serves to inform the receiver when data is available. In addition, another line may be used by the receiver to inform the sender that the data has been used, and its ready for the next data.
Parallel and Serial Data

• Serial (e.g. 8 bits)
  – Each bit is sent over a single wire, one after the other
  – Usually no signal lines are used to convey clock (timing information)
  – There are two types of serial transmission, essentially having to do with how the clock is embedded into the serial data
    • Asynchronous serial transmission
    • Synchronous serial transmission

• Parallel transmission is obviously faster, in that all bits are sent at the same time, whereas serial transmission is slower, because only one bit can be sent at a time. Parallel transmission is very costly for anything except short links.
Asynchronous Serial Transmission: RS232

- Most widely used standard for decades!
- Software-configurable: UART (Universal asynchronous receiver/transmitter)
- Groups data together into a sequence of bits (five to eight), then prefixes them with a start bit and a stop bit.

Electrical interface is held at **high** between the 8-bit characters. The start of transmission of a character is signaled by a drop in signal level to **low**. The receiver starts its clock. After one bit time (the start bit) come 8 bits of true data followed by one or more stop bits at 1 level.

- Transmitter and receiver only have to have approximately the same clock rate (5% difference OK!).
- High **high overhead**, in that each byte carries at least two extra bits: a 20% loss of line bandwidth.
Synchronous Serial Transmission: PS/2

- The PS/2 mouse and keyboard implement a bidirectional synchronous serial protocol.
- The bus is "idle" when both lines are high. This is the only state where the keyboard/mouse is allowed begin transmitting data. The host has ultimate control over the bus and may inhibit communication at any time by pulling the Clock line low.
- The device (slave) always generates the clock signal. If the host wants to send data, it must first inhibit communication from the device by pulling Clock low. The host then pulls Data low and releases Clock. This is the "Request-to-Send" state and signals the device to start generating clock pulses.

<table>
<thead>
<tr>
<th>Summary: Bus States</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data = high, Clock = high: <strong>Idle state.</strong></td>
</tr>
<tr>
<td>Data = high, Clock = low: <strong>Communication Inhibited.</strong></td>
</tr>
<tr>
<td>Data = low, Clock = high: <strong>Host Request-to-Send</strong></td>
</tr>
</tbody>
</table>

Data is transmitted 1 byte at a time:
- 1 start bit. This is always 0.
- 8 data bits, least significant bit first.
- 1 parity bit (odd parity - The number of 1's in the data bits plus the parity bit always add up to an odd number. This is used for error detection.).
- 1 stop bit. This is always 1.
- 1 acknowledge bit (host-to-device communication only)
## Serial Communication: Some Examples

<table>
<thead>
<tr>
<th>Name</th>
<th>Sync/Async</th>
<th>Type</th>
<th>Duplex</th>
<th>Max devices</th>
<th>Max speed (Kbps)</th>
<th>Max distance (feet)</th>
<th>Pin count (not including ground)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS-232</td>
<td>async</td>
<td>peer</td>
<td>full</td>
<td>2</td>
<td>115.2</td>
<td>30</td>
<td>2 (or 4 with HW handshake)</td>
</tr>
<tr>
<td>RS-422</td>
<td>async</td>
<td>multi-drop</td>
<td>half</td>
<td>10</td>
<td>10000</td>
<td>4,000</td>
<td>1 (unidirectional only, additional pins for each bidirectional comm.)</td>
</tr>
<tr>
<td>RS-485</td>
<td>async</td>
<td>multi-point</td>
<td>half</td>
<td>32</td>
<td>10000</td>
<td>4,000</td>
<td>2</td>
</tr>
<tr>
<td>I²C</td>
<td>sync</td>
<td>multi-master</td>
<td>half</td>
<td>Limitation based on bus capacitance and bit rate</td>
<td>3400</td>
<td>&lt;10</td>
<td>2</td>
</tr>
<tr>
<td>SPI</td>
<td>sync</td>
<td>multi-master</td>
<td>full</td>
<td>Limitation based on bus capacitance and bit rate</td>
<td>&gt;1000</td>
<td>&lt;10</td>
<td>3+1 (Additional pins needed for every slave if slave count is more than one)</td>
</tr>
<tr>
<td>Microwire</td>
<td>sync</td>
<td>master/slave</td>
<td>full</td>
<td>Limitation based on bus capacitance and bit rate</td>
<td>&gt;625</td>
<td>&lt;10</td>
<td>3+1 (Additional pins needed for every slave if slave count is more than one)</td>
</tr>
<tr>
<td>1-Wire</td>
<td>async</td>
<td>master/slave</td>
<td>half</td>
<td>Limitation based on bus capacitance and bit rate</td>
<td>16</td>
<td>1,000</td>
<td>1</td>
</tr>
</tbody>
</table>
Side note: Data Multiplexing

- A multiplexer is a device which shares a communication link between a number of devices (users).
- Rather than provide a separate circuit for each device, the multiplexer combines each low speed circuit onto a single high speed link. The cost of the single high speed link is less than the required number of low speed links.
- It does this by time or frequency division.
Getting rid of the chaos: Universal Serial Bus

- Ease-of-use for PC peripheral expansion
- Low-cost solution that supports transfer rates up to 12Mb/s
- Full support for real-time data for voice, audio, and compressed video
- Protocol flexibility for mixed-mode isochronous data transfers and asynchronous messaging
- Integration in commodity device technology
- Comprehension of various PC configurations and form factors
Taxonomy of USB Application Space

<table>
<thead>
<tr>
<th>PERFORMANCE</th>
<th>APPLICATIONS</th>
<th>ATTRIBUTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOW-SPEED</td>
<td>Keyboard, Mouse</td>
<td>Lower Cost</td>
</tr>
<tr>
<td></td>
<td>Stylus</td>
<td>Hot Plug-unplug</td>
</tr>
<tr>
<td></td>
<td>Game Peripherals</td>
<td>Ease-of-use</td>
</tr>
<tr>
<td></td>
<td>Virtual Reality Peripherals</td>
<td>Multiple Peripherals</td>
</tr>
<tr>
<td></td>
<td>Monitor Configuration</td>
<td></td>
</tr>
<tr>
<td>MEDIUM-SPEED</td>
<td>ISDN</td>
<td>Low Cost</td>
</tr>
<tr>
<td></td>
<td>PBX</td>
<td>Ease-of-use</td>
</tr>
<tr>
<td></td>
<td>POTS</td>
<td>Guaranteed Latency</td>
</tr>
<tr>
<td></td>
<td>Audio</td>
<td>Guaranteed Bandwidth</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dynamic Attach-Detach</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Multiple devices</td>
</tr>
<tr>
<td>HIGH-SPEED</td>
<td>Video</td>
<td>High Bandwidth</td>
</tr>
<tr>
<td></td>
<td>Disk</td>
<td>Guaranteed Latency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ease-of-use</td>
</tr>
</tbody>
</table>

Low Speed: Interactive Devices (10 – 100kb/s)
Medium Speed: Phone, Audio, Compressed Video (500Kb/S - 10Mb/s)
High Speed: Video, Disk (25 - 500Mb/s)
USB’s Feature list

• Easy to use for end user
  – Single model for cabling and connectors
  – Electrical details isolated from end user (e.g., bus terminations)
  – Self-identifying peripherals, automatic mapping of function to driver, and configuration
  – Dynamically attachable and re-configurable peripherals

• Wide range of workloads and applications
  – Suitable for device bandwidths ranging from a few kb/s to several Mb/s
  – Supports isochronous as well as asynchronous transfer types over the same set of wires
  – Supports concurrent operation of many devices (multiple connections)
  – Supports up to 127 physical devices
  – Supports transfer of multiple data and message streams between the host and devices
  – Allows compound devices (i.e., peripherals composed of many functions)
  – Lower protocol overhead, resulting in high bus utilization
USB Feature list (cont’d)

- **Isochronous bandwidth**
  - Guaranteed bandwidth and low latencies appropriate for telephony, audio, etc.
  - Isochronous workload may use entire bus bandwidth

- **Flexibility**
  - Supports a wide range of packet sizes, which allows a range of device buffering options
  - Allows a wide range of device data rates by accommodating packet buffer size and latencies
  - Flow control for buffer handling is built into the protocol

- **Robustness**
  - Error handling/fault recovery mechanism is built into the protocol
  - Dynamic insertion and removal of devices is identified in user-perceived real-time
  - Supports identification of faulty devices
USB Feature list (cont’d II)

- Relation with PC industry
  - Protocol is simple to implement and integrate
  - Consistent with the PC plug-and-play architecture
  - Leverages existing operating system interfaces
- Low-cost implementation
  - Low-cost sub-channel at 1.5Mb/s
  - Optimized for integration in peripheral and host hardware
  - Suitable for development of low-cost peripherals
  - Low-cost cables and connectors
  - Uses commodity technologies
- Upgrade path
  - Architecture upgradeable to support multiple USB Host Controllers in a system
Universal Serial Bus System Components

- Interconnect
- Devices
- Host
USB Interconnect: Bus Topology

- **Bus Topology**: Connection model between USB devices and the host.
- **Inter-layer Relationships**: The USB tasks that are performed at each layer in the system.
- **Data Flow Models**: The manner in which data moves in the system over the USB between producers and consumers.
- **USB Schedule**: The USB provides a shared interconnect. Access to the interconnect is scheduled in order to support isochronous data transfers and to eliminate arbitration overhead.
# Physical Interface

<table>
<thead>
<tr>
<th>pin</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Vcc</td>
<td>+5 Vdc</td>
</tr>
<tr>
<td>2</td>
<td>D-</td>
<td>Data-</td>
</tr>
<tr>
<td>3</td>
<td>D+</td>
<td>Data+</td>
</tr>
<tr>
<td>4</td>
<td>GND</td>
<td>Ground</td>
</tr>
</tbody>
</table>
Throughput

- **Low speed**
  - 1.5 Mbit/s

- **Full speed**
  - 12 Mbit/s

- **High Speed**
  - 480 Mbit/s (USB 2.0)

- **SuperSpeed**
  - 5 Gbit/s (USB 3.0)

- **SuperSpeed+**
  - 10 Gbit/s (USB 3.1 Gen 2)

- **Dynamic mode switching**
Power Distribution

• Devices:
  – Bus-powered devices
  – Self-powered devices

• Power management
  – Host based
  – Power events:
    » Suspend
    » Resume
USB Protocol

• Polled bus
• Data transfers initiated only by host controller
• Three packets:
  – Token packet
    » Type
    » Direction
    » Address
    » End point number
  – Data packet
  – Handshake packet
    » ACK
    » NAK

More on this in Project 5!
Transfer Model

- **Pipe**
  - Stream pipes
    - No USB standard format
  - Message pipes
    - USB format
      - Request
      - Data
      - Status

- **End point zero**
  - Default control pipe
  - Always exists
Data Flow Types

• Control Transfers:
  – Used to configure a device at attach time and can be used for other device-specific purposes, including control of other pipes on the device.

• Bulk Data Transfers:
  – Generated or consumed in relatively large and bursty quantities.

• Interrupt Data Transfers:
  – Used for characters or coordinates with human-perceptible echo or feedback response characteristics.

• Isochronous Data Transfers:
  – Occupy a pre-negotiated amount of USB bandwidth with a pre-negotiated delivery latency.
USB Devices

- All have endpoint zero

- Hub
- Function
- Compound
Typical Configuration

TYPICAL USB ARCHITECTURAL CONFIGURATION

- Hub/Function
- KBD
- Mouse
- Speaker
- Mic
- Phone
- Hub

- Host/Hub
- PC

- USB

Function

Function

Function

Function

Function

Hub
USB Host Responsibility

- Detecting the attachment and removal of USB devices
- Managing control flow between the host and USB devices
- Managing data flow between the host and USB devices
- Collecting status and activity statistics
- Providing power to attached USB devices.
Physical / Logical Bus Topology

**Figure 5-5. USB Physical Bus Topology**

**Figure 5-6. USB Logical Bus Topology**
Summary

- I/O Communication Parameters:
  - Speed
  - Length of cable
  - Complexity of implementation

- USB
  - Complex network for „smart“ devices