6.111 Serial Data Demo

Miren Bamforth - Fall 2015
Serial and Parallel Links

Currently pushing 10-20Gb/s...
Serial Communications

- Sending information one bit at a time vs. many bits in parallel
  - Serial: good for long distance (save on cable, pin and connector cost, easy synchronization). Requires “serializer” at sender, “deserializer” at receiver
  - Parallel: issues with clock skew, crosstalk, interconnect density, pin count. Used to dominate for short-distances (eg, between chips).
  - BUT modern preference is for parallel, but independent serial links (eg, PCI-Express x1, x2, x4, x8, x16) as a hedge against link failures.

- A zillion standards
  - Asynchronous (no explicit clock) vs. Synchronous (CLK line in addition to DATA line).
  - Recent trend to reduce signaling voltages: save power, reduce transition times
  - Control/low-bandwidth Interfaces: SPI, I²C, 1-Wire, PS/2, AC97
  - Networking: RS232, Ethernet, T1, Sonet
  - Computer Peripherals: USB, FireWire, Fiber Channel, Infiniband, SATA, Serial Attached SCSI
Intro to Serial Data

How standards proliferate:
(See: A/C chargers, character encodings, instant messaging, etc.)

Situation: There are 14 competing standards.

14?! Ridiculous! We need to develop one universal standard that covers everyone's use cases. Yeah!

Soon:

Situation: There are 15 competing standards.
RS232 (aka “serial port”)

- **Labkit:** simple bidirectional data connection with computer.
- **Characteristics**
  - Large voltages => special interface chips
    - (1/mark: -12V to -3V, 0/space: 3V to 12V)
  - Separate xmit and rcv wires: full duplex
  - Slow transmission rates (1 bit time = 1 baud): most interfaces
    support standardized baud rates: 1200, 2400, 4800, 9600, 19.2K,
    38.4K, 57.6K, 115.2K
  - **Format**
    - Wire is held at 1/mark when idle
    - Start bit (1 bit of “0” at start of transmission)
    - Data bits (LSB first, can be 5 to 8 bits of data)
    - Parity bit (none, even, odd)
    - Stop bits (1, 1.5 or 2 bits of 1/mark at end of symbol)
    - Most common 8-N-1: eight data bits, no parity, one stop bit
RS232 interface

- Transmit: easy, just build FSM to generate desired waveform with correct bit timing
- Receive:
  - Want to sample value in middle of each bit time
  - Oversample, eg, at 16x baud rate
  - Look for 1->0 transition at beginning of start bit
  - Count to 8 to sample start bit, then repeatedly count to 16 to sample other bits
  - Check format (start, data, parity, stop) before accepting data.

Figure from http://www.arcelect.com/rs232.htm
**SPI (Serial Peripheral Interface)**

- Simple, 3-wire interface + devices selects
  - SCLK generated by master (1-70MHz). Assert data on one edge, sample data on the other. Default state of SCLK and assignment of edges is often programmable.
  - Master Out Slave In (MOSI) data shifted out of master register into slave register
  - Master In Slave Out (MISO) data shifted out of slave register and into master register
  - Selects (usually active low) determine which device is active. Assertion often triggers an action in the slave, so master waits some predetermined time then shifts data.

*Figures from Wikipedia*
I²C (Inter-Integrated Circuit)

- 2 open-drain wires (SCL = clock, SDA = data)
- Multiple-master, each transmission addresses a particular device, many devices have many different sub-addresses (internal registers)
- Format (all addresses/data send MSB first):
  - Sender: Start [S] bit (SDA↓ while SCL high)
  - Sender: One or more 8-bit data packets, each followed by 1-bit ACK
    - Data changed when SCL low, sampled at SCL↑
    - Receiver: Active-low ACK generated after each data packet
  - Sender: Stop [P] bit (SDA↑ while SCL high)
- SCL and SDA have pullup resistors, senders only drive low, go high-impedance to let pullups make line high (so multiple drivers okay!)
  - Receiver can hold SCL low to stretch clock timing, sender must wait until SCL goes high before moving to next bit.
  - Multiple senders can contend using SDA for arbitration

Intro to Serial Data
USB (Universal Serial Bus)

- 2-wire (D+, D-) for high-speed, bidirectional polled transmission between master and addressable endpoints in multiple devices. Full speed (12Mbps) and High speed (480Mbps) data rates.
- Multi-level tiered-star topology (127 devices, including hubs)
- FTDI UM245R USB-to-FIFO module for bidirectional data transfer using a handshake protocol, also asynchronous “bit-bang” mode with selectable baud rates.
  - 24-pin DIP module, wire to user pins
  - Drivers for Windows workstations in lab

Figures from ftdi.com
Serial data in summary:

- Serial allows communication with few wires between devices
- Common protocols: UART, SPI, I2C, etc
- Asynchronous vs Synchronous
1) Synchronous Transmission:

- Transmitter sends bits on falling edge of the clock
- Receiver reads bits on rising edge of the clock

Note: Many synchronous protocols send MSB first
Asynchronous means no clock...so how do we know when to look for data bits?

Asynchronous Serial Data
Predetermined timing specifications are the key to asynchronous serial data!

Lab 5b:
- 2.4ms start bit
- 1.2ms 1 bit
- 0.6ms 0 bit
STARTING: begin

   // sample whenever expired is true to read the start bit
   if (expired) begin
      if (ir_clean) begin
         if (start_counter < 5'b11111) begin
            start_counter <= start_counter + 1;
         end
      end
      else if (!ir_clean && ir_prev) begin
         if (start_counter > 5'b11100) begin
            state <= READING;
            bit_counter <= 4'd0;
            start <= 1'b1;
         end
      end
      else begin
         start_counter <= 5'b00000;
         state <= IDLE;
      end
   end
   ir_prev <= ir_clean;
end
The DMX512 protocol is characterized as an asynchronous serial data stream that runs at 250 kHz. As a result, each “bit” will be 4 μs long. DMX512 has one start bit (low), eight bits of data, 2 stop bits and no parity.

Figure 4. DMX512 Timing Diagram

<table>
<thead>
<tr>
<th>Table 1. DMX512 Timing Chart</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Break</td>
</tr>
<tr>
<td>Mark After Break (MAB)</td>
</tr>
<tr>
<td>Frame Width</td>
</tr>
<tr>
<td>Start/Data/Stop bits</td>
</tr>
<tr>
<td>Mark Time Between Frames (MTBF)</td>
</tr>
<tr>
<td>Mark Time Between Packets (MTBP)</td>
</tr>
</tbody>
</table>
Asynchronous Serial Data: DMX512

Some useful links:


UART, I2C, SPI guide (ignore the device-specific info): [https://tessel.io/blog/108840925797/a-web-developers-guide-to-communication-protocols](https://tessel.io/blog/108840925797/a-web-developers-guide-to-communication-protocols)

USB: [http://www.beyondlogic.org/usbnutshell/usb1.shtml](http://www.beyondlogic.org/usbnutshell/usb1.shtml)