Serial Communications

Prof. Stephen A. Edwards
sedwards@cs.columbia.edu

Columbia University
Spring 2008
Early Serial Communication

Morse code key

<table>
<thead>
<tr>
<th>Letters</th>
<th>Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
</tr>
<tr>
<td>D</td>
<td>4</td>
</tr>
<tr>
<td>E</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>6</td>
</tr>
<tr>
<td>G</td>
<td>7</td>
</tr>
<tr>
<td>H</td>
<td>8</td>
</tr>
<tr>
<td>I</td>
<td>9</td>
</tr>
<tr>
<td>J</td>
<td>0</td>
</tr>
<tr>
<td>K</td>
<td>-</td>
</tr>
<tr>
<td>L</td>
<td>-</td>
</tr>
<tr>
<td>M</td>
<td>-</td>
</tr>
<tr>
<td>N</td>
<td>-</td>
</tr>
<tr>
<td>O</td>
<td>-</td>
</tr>
<tr>
<td>P</td>
<td>-</td>
</tr>
<tr>
<td>Q</td>
<td>-</td>
</tr>
<tr>
<td>R</td>
<td>-</td>
</tr>
<tr>
<td>S</td>
<td>-</td>
</tr>
<tr>
<td>T</td>
<td>-</td>
</tr>
<tr>
<td>U</td>
<td>-</td>
</tr>
<tr>
<td>V</td>
<td>-</td>
</tr>
<tr>
<td>W</td>
<td>-</td>
</tr>
<tr>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>Y</td>
<td>-</td>
</tr>
<tr>
<td>Z</td>
<td>-</td>
</tr>
</tbody>
</table>

Serial Communications – p.
Later Serial Communication

Data Terminal Equipment

Data Communications Equipment
RS-232

Defined in early 1960s
Serial, Asynchronous, Full-duplex,
Voltage-based, point-to-point, 100 ft+ cables

\[
\begin{align*}
+12V & \quad \text{SPACE} = 0 \\
+3V & \\
-3V & \quad \text{MARK} = 1 \\
-12V & 
\end{align*}
\]

Idle Start LSB B1 B2 B3 B4 B5 B6 MSB Stop

Tx
## RS-232 Signals

<table>
<thead>
<tr>
<th>Signal</th>
<th>Pin</th>
<th>DCE</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>RxD</td>
<td>2</td>
<td>←</td>
<td>Data received by DTE</td>
</tr>
<tr>
<td>TxD</td>
<td>3</td>
<td>→</td>
<td>Data sent by DTE</td>
</tr>
<tr>
<td>SG</td>
<td>5</td>
<td>—</td>
<td>Ground</td>
</tr>
<tr>
<td>DSR</td>
<td>6</td>
<td>←</td>
<td>Data Set Ready (I’m alive)</td>
</tr>
<tr>
<td>DTR</td>
<td>4</td>
<td>→</td>
<td>Data Terminal Ready (me, too)</td>
</tr>
<tr>
<td>DCD</td>
<td>1</td>
<td>←</td>
<td>Carrier Detect (hear a carrier)</td>
</tr>
<tr>
<td>RTS</td>
<td>7</td>
<td>→</td>
<td>Request To Send (Yo?)</td>
</tr>
<tr>
<td>CTS</td>
<td>8</td>
<td>←</td>
<td>Clear To Send (Yo!)</td>
</tr>
<tr>
<td>RI</td>
<td>9</td>
<td>←</td>
<td>Ring Indicator</td>
</tr>
</tbody>
</table>
Most UARTs actually use 16\times clocks
Parity bit: (Even = true when even number of 1s)

Two stop bits:
### Baud Rate

Baud: bits per second

<table>
<thead>
<tr>
<th>Baud</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>ASR-33 Teletype</td>
</tr>
<tr>
<td>300</td>
<td>Early acoustic modems</td>
</tr>
<tr>
<td>1200</td>
<td>Direct-coupled modems c. 1980</td>
</tr>
<tr>
<td>2400</td>
<td>Modems c. 1990</td>
</tr>
<tr>
<td>9600</td>
<td>Serial terminals</td>
</tr>
<tr>
<td>19200</td>
<td></td>
</tr>
<tr>
<td>38400</td>
<td>Typical maximum</td>
</tr>
</tbody>
</table>
Physical Variants

Connectors: DB-25, DB-9, Mini DIN-8
RS-422: Differential signaling  RS-485: Bus-like

![Diagram showing RS-422 and RS-485 configurations](image-url)
Philips invented the Inter-IC bus c. 1980 as a very cheap way to communicate slowly among chips. E.g., good for setting control registers. 100, 400, and 3400 kHz bitrates.

SCL: Clock, generated by a single master
SDA: Data, controlled by either master or slaves
I²C Bus Transaction

Idle | Start | “0” | “1” | Ack | Stop

SCL

SDA

Write data

<table>
<thead>
<tr>
<th>S</th>
<th>slave address</th>
<th>W</th>
<th>data</th>
<th>A</th>
<th>data</th>
<th>A</th>
<th>P</th>
</tr>
</thead>
</table>

< n data bytes >

Read data

<table>
<thead>
<tr>
<th>S</th>
<th>slave address</th>
<th>R</th>
<th>data</th>
<th>A</th>
<th>data</th>
<th>A</th>
<th>P</th>
</tr>
</thead>
</table>

< n data bytes > last data byte

Master

Slave

transmitter

receiver

transmitter

receiver

S = Start condition
A = Acknowledge
P = Stop condition

R/W = read / write not
A = Not Acknowledge
USB: Universal Serial Bus

1.5 Mbps, 12 Mbps, and 480 Mbps (USB 2.0)
Point-to-point, differential, twisted pair
3–5m maximum cable length
# USB Connectors

<table>
<thead>
<tr>
<th>Series &quot;A&quot; Connectors</th>
<th>Series &quot;B&quot; Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Series &quot;A&quot; plugs are always oriented <strong>upstream</strong> towards the <em>Host System</em></td>
<td>• Series &quot;B&quot; plugs are always oriented <strong>downstream</strong> towards the <em>USB Device</em></td>
</tr>
</tbody>
</table>

**"A" Plugs**
*From the USB Device*

**"A" Receptacles**
*Downstream Output from the USB Host or Hub*

**"B" Plugs**
*From the Host System*

**"B" Receptacles**
*Upstream Input to the USB Device or Hub*
NRZI: 0 = toggle, 1 = no change

Bit stuffing: 0 automatically inserted after six consecutive 1s

Each packet prefixed by a SYNC field: 3 0s followed by two 1s

Low- vs. full-speed devices identified by different pull-ups on D+/D- lines
USB Packets

Always start with SYNC
Then 4-bit type, 4-bit type complemented
2 bits distinguish Token, Data, Handshake, and Special, other two bits select sub-types
Then data, depending on packet type
Data checked using a CRC
Addresses (1-128) assigned by bus master, each with 16 possible endpoints
Polled bus: host initiates all transfers.
Most transactions involve three packets:
- “Token” packet from host requesting data
- Data packet from target
- Acknowledge from host

Supports both streams of bytes and structured messages (e.g., control changes).
USB Data Flow Types

- Control
  For configuration, etc.

- Bulk Data
  Arbitrary data stream: bursty

- Interrupt Data
  Timely, reliable delivery of data. Usually events.

- Isochronous Data
  For streaming real-time transfer: prenegotiated bandwidth and latency
**USB: Flash Card Device**

Bus 001 Device 002: ID 05e3:0760 Genesys Logic, Inc.
- `bcdUSB`: 2.00
- `bMaxPacketSize0`: 64
- `idVendor`: 0x05e3 Genesys Logic, Inc.
- `idProduct`: 0x0760
- `bcdDevice`: 1.14
- `iManufacturer`: 2 Genesys
- `iProduct`: 3 Flash Reader
- `iSerial`: 4 002364

**Configuration Descriptor:**
- `bNumInterfaces`: 1
- `MaxPower`: 300mA

**Interface Descriptor:**
- `bNumEndpoints`: 2
- `bInterfaceClass`: 8 Mass Storage
- `bInterfaceSubClass`: 6 SCSI
- `bInterfaceProtocol`: 80 Bulk (Zip)

**Endpoint Descriptor:**
- `bEndpointAddress`: 0x81 EP 1 IN
  - `bmAttributes`: 2
    - `Transfer Type`: Bulk
    - `Synch Type`: none
    - `wMaxPacketSize`: 64
- `Endpoint Descriptor:`
  - `bLength`: 7
  - `bDescriptorType`: 5
  - `bEndpointAddress`: 0x02 EP 2 OUT
  - `bmAttributes`: 2
    - `Transfer Type`: Bulk
    - `Synch Type`: none
    - `wMaxPacketSize`: 64

**Language IDs:** (length=4)
- 0409 English(US)
Bus 002 Device 002: ID 04b4:0001 Cypress Semiconductor Mouse

Device Descriptor:
  bcdUSB 1.00
  idVendor 0x04b4 Cypress Semiconductor
  idProduct 0x0001 Mouse
  bcdDevice 4.90
  iManufacturer 1 Adomax Sem.
  iProduct 2 USB Mouse
  iSerial 0

Configuration Descriptor:
  bNumInterfaces 1
  bmAttributes 0xa0
    Remote Wakeup
  MaxPower 100mA

Interface Descriptor:
  bNumEndpoints 1
  bInterfaceClass 3 Human Interface Devices
  bInterfaceSubClass 1 Boot Interface Subclass
  bInterfaceProtocol 2 Mouse
  iInterface 5
    EndPoint1 Interrupt Pipe

HID Device Descriptor:
  bDescriptorType 34 Report
  wDescriptorLength 52

Endpoint Descriptor:
  bEndpointAddress 0x81 EP 1 IN
  bmAttributes 3
    Transfer Type Interrupt
    Synch Type none
  wMaxPacketSize 4
  bInterval 10

Language IDs: (length=4)
  0409 English(US)
On the DE2, one downstream port, one host
Operates at 12 or 480 Mbps speeds
Two control endpoints + 14 user endpoints
4096 (host) + 2462 (device) bytes buffer memory
Supports DMA data transfers
Many configuration and status registers
150-page data “sheet” + 99-page embedded programming guide