Programming for GCSE
Topic 7.1: Principles of Communication
Outline

• Activity 1
  • Encoding and modulation
  • Error correction
  • Multiplexing
  • Communicate in both directions

• Activity 2
  • Low-level communication
  • Clock synchronisation
  • Framing
Teaching Issues
Teaching Issue

- GCSE material on networks and communication lack concepts
  - It is also quite out of date

- Principles
- Real-world examples
From the specification

- **OCR GCSE Computing.** Candidates should be able to:
  - (a) explain the advantages of networking stand-alone computers into a local area network
  - (b) describe the hardware needed to connect stand-alone computers into a local area network, including hub/switches, wireless access points
  - (c) explain the different roles of computers in a client-server and a peer-to-peer network
  - (d) describe, using diagrams or otherwise, the ring, bus and star network topologies
From the specification

- **OCR GCSE Computing.** Candidates should be able to:
- (e) describe the differences between a local area network and a wide area network such as the internet.
- (f) explain the terms IP addressing, MAC addressing, packet and protocols.
- (g) explain the need for security measures in networks, such as user access levels, suitable passwords and encryption techniques.
- (h) describe and justify network policies such as acceptable use, disaster recovery, failover, back up, archiving.
Activity: Transmitting Data

• Simple activity
  • Exchange data
  • Look at principles

• Elaborate activity
  • Notice problem
  • New principle

What's a protocol
**Basic Activity**

- **Sending station**
  - Application: create a message
  - Code: text $\rightarrow$ binary
  - Modulate: binary $\rightarrow$ noise

- **Receiving station**
  - Demodulate: noise $\rightarrow$ binary
  - Decode: binary $\rightarrow$ text
  - Application: enjoy message
## Basic Activity – Code

<table>
<thead>
<tr>
<th>Space</th>
<th>00000</th>
<th>N</th>
<th>01110</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>00001</td>
<td>O</td>
<td>01111</td>
</tr>
<tr>
<td>B</td>
<td>00010</td>
<td>P</td>
<td>10000</td>
</tr>
<tr>
<td>C</td>
<td>00011</td>
<td>Q</td>
<td>10001</td>
</tr>
<tr>
<td>D</td>
<td>00100</td>
<td>R</td>
<td>10010</td>
</tr>
<tr>
<td>E</td>
<td>00101</td>
<td>S</td>
<td>10011</td>
</tr>
<tr>
<td>F</td>
<td>00110</td>
<td>T</td>
<td>10100</td>
</tr>
<tr>
<td>G</td>
<td>00111</td>
<td>U</td>
<td>10101</td>
</tr>
<tr>
<td>H</td>
<td>01000</td>
<td>V</td>
<td>10110</td>
</tr>
<tr>
<td>I</td>
<td>01001</td>
<td>W</td>
<td>10111</td>
</tr>
<tr>
<td>J</td>
<td>01010</td>
<td>X</td>
<td>11000</td>
</tr>
<tr>
<td>K</td>
<td>01011</td>
<td>Y</td>
<td>11001</td>
</tr>
<tr>
<td>L</td>
<td>01100</td>
<td>Z</td>
<td>11010</td>
</tr>
<tr>
<td>M</td>
<td>01101</td>
<td>?</td>
<td>11011</td>
</tr>
</tbody>
</table>
Transmitting

- Message format
  - Read & transmit binary **left to right**

```
E 00101
```

Two noises to signal

```
1 0 0 0
```
Layout – Seating

Application
Encoder
Transmitter

Receiver
Decoder
Application
Transmission in Practice

- Data can be sent using 2 signals
  - Two noises resembles 'frequency' modulation (FM)

- Real modulation schemes very complex
  - Achieve very high data rates over twisted pair
  - E.g. see Wikipedia article on OFDM
Elaborations

1. Error detect using Parity
   • Add a parity bit
   • Acknowledge safe receipt

2. Send two messages at once
   • Two applications on each computer
   • Share link – **multiplex**

3. Reply to a message
   • Communication in both directions
   • Channel is **multiple access**
Elaboration 1: Parity

Transmitter: 5 bits → 6 bits

Receiver shouts 'OK'

A  0 0 0 0 1 1
Parity: Roles

• Transmitter adds parity bit
  • All code words have an **even** number of '1' digits
  • Parity bit at right hand side: transmitted last
• Receiver's decoder counts '1'
  • ACK (acknowledge) correct code word
  • Delete the parity bit
• Retransmission
  • If no acknowledgement then RETRANSMIT the code word
In Practice: Parity and ACK

- Error detection VITAL
  - Parity cannot detect two errors
  - CRC – more complex than parity

- By shouting 'ok', we have cheated
  - It's a third symbol!

- ACK is a separate message (a reply)
  - Latency: if you have to wait for a ACK then the messages over long distances are slow
Elaboration 2: Multiplex

- How can we share link?

```
P   L   E   H
10000 01100 00101 01000
```

Diagram:
- Green squares for P, L, E, H
- Orange squares for 10000, 01100, 00101, 01000
Elaboration 2: Multiplex

- Sending station
  - Application: create a message
  - **Multiplex: 0 or 1**
  - Code: text $\rightarrow$ binary
  - Modulate: binary $\rightarrow$ noise

Each character: now 7 bits

- Receiving station
  - Demodulate: noise $\rightarrow$ binary
  - Decode: binary $\rightarrow$ text
  - **De-multiplex: 0 or 1**
  - Application: enjoy message
Message Format

• Application
  • 1 bit address of destination application

• Encoder

• Transmitter
  • Add parity
  • Transmitted left to right
Take a character from one application: add 0 or 1
Multiplex in Practice

- Message of multiple code words
  - Message length
  - Error check for whole message
- Source and destination address
Elaboration 3: Multiple Access and Reply

• Shared channel
  • Wi-fi
  • Bus-topology (old Ethernet)

• *How to avoid confusion?*
Elaboration 3: Multiple Access

- Shared channel
  - Wi-fi
  - Bus-topology (old Ethernet)
- How to avoid confusion?
  - Token exchange
  - **Random turn taking** (wifi)
Elaboration 3: Multiple Access

- 2 stations at either end
  - Sender
  - Receiver
- Rules for transmitting
  - Listen for silence before starting
  - ... transmit when you have data
  - If two stations transmit at once
    - LONG BLAST then STOP
  - Wait a bit; try again

Note: with multiple access we could have more than two tables but more addresses needed
Elaboration 3: Reply

- Applications are either
  - Server – answers a question
  - Clients – asks a question
- Place one server and one client on each table
  - Client asks
  - Server responds
- Application protocol
  - Question ends with a '?'

Note: HTTP is an application protocol to display web pages. It sends messages like 'get index.html'
Layout - Seating

App2 Client → Decoder → Receiver → Transmitter → Encoder → App1 Client

App1 Server → Encoder → Transmitter → Receiver → Decoder → App2 Server
In Practice: Multiple Access

- Multiple access is used in
  - Original Ethernet
  - Wifi
- Switches now avoid multiple access in Ethernet
- We have ignored
  - Station address: MAC address is used in Ethernet, Wifi and Bluetooth
In Practice: Reply

• Client / server is the basis of the Internet
  • E.g. web, email
  • Conservations are between applications on hosts
  • Ok to browse same web site twice
  • Our address (1 bit) is a combination of an IP address (for a host) and TCP 'port' (for an application)

• We have assumed sender address is same as destination address
  • In practice, need both
COMMUNICATION PRINCIPLES

Summary
Concepts

- **Signal**: transmit binary
- **Modulate**: encode the binary to transmit
- **Parity**: detect errors
- **Multiplex**: share a link
- **Protocol**: agree on rules of communication