Contents

1 Data Representation

1.1 Number systems

1.1.1 Computers and binary

The number system consisting of digits 0 through 9 is called the denary or decimal system (10 digits). Computers use another number system called binary, consisting of digits 0 through 1. This is done such that computers are able to pass the data through logic gates and can be stored in registers.

1.1.2 The binary, denary and hexadecimal systems

The number of digits in a number system is the base of that system; denary is base 10; binary is base 2; hexadecimal is base 16.

Conversions

From positive denary to positive binary:

1. Perform short division on given denary number, taking note of the remainders.
2. Write the remainders from bottom to top, resulting in the binary number.

From positive binary to positive denary:

1. Write the binary number with their *place powers*.
2. Sum the products of each binary digit with the place power, resulting in the converted denary number.
3. Write the remainders from bottom to top, resulting in the binary number.

From positive denary to positive hexadecimal:

1. Convert given number to binary
2. Split resulting binary number to four-bit parts.
3. Convert the four-bit binaries to denary.
4. 1 = 1; 2 = 2; ... 10 = A; 11 = B; 12 = C; 13 = D; 14 = E; 15 = F.
5. Arrange resulting digits side-by-side.

From positive hexadecimal to positive denary:

1. Convert each given hex number to denary using the index in step 4 of denary-hex.
2. Convert each denary number to binary.

3. Arrange the resulting four-bit binary pieces, producing binary result.

From positive hexadecimal to positive binary:

1. Convert to denary.
2. Convert to binary.

From positive binary to positive hexadecimal:

1. Convert to binary.
2. Convert to denary.

1.1.3 Uses of the hexadecimal system

The hexadecimal number system is used to make life easier for humans dealing with bare low-level computer code. Hexadecimal requires less digits, and are easier to compare with the naked eye. Data errors can be easier to find when looking at this shortened form of binary, hexadecimal.

1.1.4 Binary addition

To add two binary numbers, refer to the following:

- $0 + 0 = 0$.
- $1 + 0 = 1$.
- $1 + 1 = 10$. (the one is carried on)
- $10 + 1 = 11$. (the one is carried on)

Sometimes, the addition results in an extra bit, which **overflows** off. This is because computers have predifined limits to which it can store its numbers (16, 32 bits) and when a value outside this limit is returned, it is not stored and an overflow error occurs.

1.1.5 Logical binary shifts

When performing logical shifts, we simply move the bits of a binary number to the right or left depending on what is required. We "delete" and hence lose the leftmost (most significant) or rightmost (least significant) bit depending on the direction of the shift performed.

Shifting right means dividing by two.

Shifting left means multiplying by two.

1.1.6 Two's complement

Two's complement is a method used to represent negative binary numbers. We simply convert given denary number to binary (if need be), invert all the bits and add $(1)_2$ to the result.

1.2 Text, sound and images

1.2.1 Text

Text is converted to binary so that a computer can process it. It does so by converting each character into an integer, as defined in the **ASCII** standard (American Standard Code For Information Interchange) and subsequently into a binary number.

Unicode is another such standard, which allows a greater range of characters, in various languages, as a result it also requires more bits per character.

1.2.2 Sound

Sounds are composed of waves. When we record values of the sound, we do so at set intervals, this process is called **sampling**. The more samples taken per unit time, the more accurate the sound recorded will be, i.e., higher the **sample rate** the greater the sound quality.

The sound values, which are usually denary numbers, can be converted to binary and stored into a computer.

The **sample resolution** is the number of bits allocated per sample value. So, the larger the sample resolution, the more the amount of digits that can be stored into a file. Thus, the higher the sample resolution, the higher the sound quality.

The file size of a sound file increases, with increased sample rate and sample resolution, that means storing high quality sound requires more space that low quality sound.

1.2.3 Images

An image is composed of **pixels**. The computer stores these pixels by processing them to binary, by assigning a binary number to a certain colour.

The **resolution** of an image is the number of pixels stored in it. Usually in the format: width $\times$ height.

The **colour depth** of an image is the number of bits allocated for each pixel of the image. Higher the colour depth, the more the number of colours that can be displayed.

Higher quality images result in larger file sizes as resolution and colour depth are large.

1.3 Data storage and compression

1.3.1 Measurement of data storage

- Bit: 1 or 0. Smallest possible data measurement.
- Nibble: 4 bits.
- Byte: 8 bits.

- Kibibyte (KiB): 1024 Bytes.
- Mebibyte (MiB): 1024 Kibibytes.
- Gibibyte (GiB): 1024 Mebibytes.
- Tebibyte (TiB): 1024 Gibibytes.
- Pebibyte (PiB): 1024 Tebibytes.
- Exbibyte: (EiB) 1024 Pebibytes.

1.3.2 File size calculation

To calculate the file size of an image, find the product of the image's width, height, colour depth

For the sound's file size, multiply the sample rate, sample resolution and soundtrack length.

1.3.3 Compression

Files can be compressed to:

- Reduce storage space needed.
- Less transmission times among devices.
- Quicker upload and download times.
- Requires less bandwidth for file transmission.

Compression is of two types:

1. Lossy: Reduces file size by permanently reducing colour depth, resolution, or sample rate.
2. Lossless: Reduces file size without permanent loss of data, using Run Length Encoding (RLE). RLE groups similar data together and hence some file space can be saved.

2 Data transmission

2.1 Types and methods of data transmission

2.1.1 Packets and transmission

When data is transmitted from one device to another, it is organised into packets. A packet consists of three parts:

- Header: Consisting of three parts again: destination address, packet number and originators address.

  The destination address is an Internet Protocol (IP) address which is a unique identifier address for every computer connected to the internet.

  The packet number assigned to a packet helps the recieving computer reorder and organise the data sent, because often data may be sent out of order.

  The originators address too is an IP address. It is that of the device from which data has been sent. It helps to trace origin of transmitted data.

- Payload: Consists of the actual data being sent.

- Trailer (Footer): Consists of data for any error detection system, and the data to signal to the reciever that this is the end of the packet.

Data is transmitted across a network (the internet for the majority of cases). Networks consist of routers. From one device to another, there exist multiple connected routers amongst whom there exists multiple paths whicch data packets can take. The routes are decided by the routers themselves. Packets may arrive out of order as a result but the packer number stored in the header helps the reciever reorder the recieved data. This is the process of packet switching.

2.1.2 Methods of dataa transmission

There are two methods of data transmission regarding the volume of data transferred:

- Serial: Data is transmitted along a single wire a bit at a time. Sequencec is maintained, very little interference is likely and cheaper for manufacturer and consumer because only requires one wire.

  However, data transmission is very slow and start and stop bits must be sent additionally.

- Parallel: Data is transmitted along multiple wires, multiple bits at a time. Data transmission is quicker, because of the sending of multiple bits at one time. No need for conversion for transmission across networks as computers use parallel transmissions internally.

  However, the bits may arrive out of order and skewing is a risk. Interference is likely and errors may arise. Expensive due to more wires required.

There are three methods of data transmission regarding the direction of data transferred:

- Simplex: Data is transferred in one direction only.

- Half-duplex: Data is transferred in both directions, not at the same time.

- (Full-)Duplex: Data is transferred in both directions at the same time.

2.1.3 The universal serial bus (USB) interface

The USB interfaec inclcudes the port, cable, connection and device. Data transmission, here is a special type of serial which allows high speed transmissions.

The USB interface is simple and connections can only be made in one way, less errors in onnecting devices are likely. The speed of data transfer is quite high. It is the industry standard so almost all devices are equipped with a USB port. When USB devices are connected, required drivers for the devices are automatically detected and downloaded. It does not need its own power source and can be used to charge devices.

The length of a USB cable is limited to a maximum of five metres. Transmission is quick yet not as quick as ethernet.

2.2 Methods of error detection

2.2.1 Necessity of error checking

Errors may occur as a result of interference during transmission, consisting of loss, gain and change of data being transmitted.

2.2.2 Processes to detect errors

These errors can be resolved in the following ways:

- Parity check: Given data is said to have even or odd parity. Bytes of data are sent with a parity bit, which is determined by the data itself. If the number of ones in the binary data is even and the parity is even, the parity bit will be one, otherwise zero. Same stands for odd parity.
- Checksum: A calculated value is transmitted with transmitted using a certain method. The reciever then uses the same method to calculate the value itself. If the transmitted and calculated values match, the data is error-free, otherwise it is corrupt.
- Echo check: Recieved data is sent back to the sender, who compares it with original data. If datas match, no error. Otherwise data is sent again.

2.2.3 Check digits

Check digits are identical to checksums but the the result of the generating algorithm is in a single digit. ISBN (International Standard Book Numbers) use check digits and so do barcodes.

2.2.4 Automated Repeat Queries (ARQ)

ARQs work in the following sequence:

1. Data is sent to reciever.
2. Reciever checks errors.

3. If data free, send positive acknowledgement to sender.
4. Otherwise send negative acknowledgement and sender re-sends data.
5. If no acknowledgement is sent beyond the timeout limit, sender sends data again.
6. Without acknowledgement, data is sent for a set number of times.

2.3 Encryption

2.3.1 The need for data transmission

Data needs to be encrypted so as not to lose sensitive data to potential hackers.

The original data is called the plaintext, having used an encryption algorithm, usually with an encryption key, a ciphertext is formed. This ciphertext is then sent across the network and some method of data decryption is used by reciever.

2.3.2 Methods of data encryption

Data can be encrypted in two ways:

1. Symmetric: Data is encrypted and decrypted using the same encryption key which is send along with the data itself. Vulnerable method as encryption key can also be compromised.
2. Asymmetric: Data is encrypted with the senders public key, decrypted with public key. This is the safer method as compromise is unlikely with only public key.

3 Hardware

3.1 Computer architecture

3.1.1 The central processing unit (CPU)

It is responsible for the process of data inputted into the computer to turn it into an output. A microprocessor is present in embedded systems and is an integrated circuit which is able to perform many of the functions of a CPU.

3.1.2 The Von Neumann architecture

The Von Neumann architecture consists of three stages in a cycle: fetch, decode and execute.

- Fetch:
    1. Inputted data, instructions and data from hard drive are initially stored and put into the RAM (Random Access Memory).

2. A register called the PC (Program Counter) stores the address of the next instruction to be processed. The address is the next location of RAM.
3. When an instruction is to be processed, the address from the PC is brought into the MAR (Memory Address Register). The address bus is used for the movement of registers.
4. Using the address stored in the MAR, the address bus retrieves the data at the address in the RAM, bringing it back into the MDR (Memory Data Register) using the data bus.
5. Once the MDR recieves the data, which is the next instruction to be processed, this data is sent to the CIR (Current Instruction Register). The transfer is done by the data bus.
6. This is the end of the fetch stage, the CIR is part of the CU (Control Unit) which is responsible for the second stage: decode.

- Decode:
  1. Using an instruction set, the CU decodes the instruction stored in the CIR.
  2. This is the end of the decode stage, now the execute stage can begin.
- Execute:
  1. Here actions required for carrying-out of instructions are done.
  2. Calculations are done by the ALU (Arithmetic Logic Unit). The ALU has a register called the ACC (ACCumulator) where any temporary values needing to be stored are stored.

The stages in the fetch-code-decode cycle are coordinated by signals transmitted through the control bus.

3.1.3 CPU Performance

CPU Performance is controlled by three main factors: number of cores, clock speed and cache.

- Cores: The more the number of cores the better the performance as more fetch- decode-execute cycles can run simultaneously.
- Clock speed: A CPU contains an internal clock which controls speed of processing of instructions. Using overclocking, CPU can process more instructions quicker but it can cause overheating.
- Cache size: Cache is a type of storage inside and hence near the CPU. The more instructions stored in cache the better as it takes less time than fetching instructions all the way from RAM.

3.1.4 Embedded systems

Embedded systems are essentially small computers that are built to do a very specific task. Examples include the systems embedded into domestic appliances (microwaves, fridges, etcetera), vending machines, security systems or lighting systems. They use microcontrollers in place of a CPU and usually lack some parts of the Von Neumann architecture.

3.2 Input and output devices

An input device is that which allows any entering of data into a computer system, including text, image and sound.

Required input devices:

- Barcode scanner: Used to scan data encoded into a barcode (a linear image consisting if dark and light lines).
- QR (Quick Response) code scanner: Used to scan data incoded into a QR code (an image consisting of dark and light squares arranged in a matrix pattern).
- Digital camera: Used to take digital images of surroundings.
- Keyboard: Used to type in text into a computer system.
- Microphone: Used to input sound data into a computer system.
- Optical mouse: A pointing device which uses a CMOS (Complementary Metal Oxide Semiconductor) to detect movement and maps that movement into the pointer on the screen.
- Touchscreen: Can be of three types: capacitive, resistive and infrared:
  - Capacitive: Voltages are produced at all four corners of the screen, any contact by finger or stylus results in change in electric field produced, position of contact can then be calculated.
  - Resistive: Consists of two layers, when pressure is applied, the layers come into contact, completing a circuit and position of contact is calculated.
  - Infrared: Infrared light beams are shot across thes screen in an X-Y pattern. When a finger or stylus contacts the screen, the light rays are blocked and the position of contact can easily be calculated.
- 2D and 3D scanners: 2D scanners are used to scan what is printed onto a piece of paper, a document. This is done to digitise the document.

  3D Scanners scan and produce a 3D image of a given object. This can be used in CAD (Computer Aided Design) circumstances.