# AQA Computer Science A-Level

# 4.5.5 Information Coding Systems

## Concise Notes

## Specification:

### 4.5.5.1 Character form of a decimal digit:

Differentiate between the character code representation of a decimal digit and its pure binary representation.

### 4.5.5.2 ASCII and Unicode:

Describe ASCII and Unicode coding systems for coding character data and explain why Unicode was introduced.

### 4.5.5.3 Error checking and correction:

Describe and explain the use of:

- parity bits
- majority voting
- checksums
- check digits

## Character form of a decimal digit

- When computers represent characters, an information coding system is used
- A character code is a decimal digit used to represent a character

## ASCII and Unicode

- ASCII stands for American Standard Code for Information Interchange
- ASCII and Unicode are two widely used information coding systems
- ASCII uses 7 bits and can represent 128 (= $2^7$) different characters
- With the advent of the Internet, there was a requirement for an information coding system that could represent character sets other than the Latin alphabet
- Unicode allows the representation of a wide variety of alphabets by computers
- Unicode uses anywhere from 8 to 48 bits (1 to 6 bytes) per character
- Unicode can represent a much wider range of different characters than ASCII

## Error checking and correction

Parity bits

- A single bit added to a transmission
- This bit can be used to check for errors in the transmitted data
- The bit's value is calculated based on the transmitted data itself
- There are two types: even parity and odd parity
- In even parity, the parity bit makes the total number of 1s in the transmitted data even
- In odd parity, the parity bit makes the total number of 1s in the transmitted data odd
- When data is received, a parity check is carried out
- If an error is detected, the computer asks the sender to retransmit the data
- If an even number of bits are changed during transmission, the error is not detected

Majority voting

- Each bit of the data is transmitted multiple times
- When received, the most commonly occurring value is taken to be correct
- Majority voting doesn't just detect the error but also corrects the error
- Therefore there's no need for retransmission like when using a parity bit
- Majority voting can correct errors when multiple bits change
- The volume of data being transmitted is increased, increasing the time taken to transmit data

Checksums

- A value is appended to the transmitted data
- This value is determined by the data itself
- Once received, the recipient removes the checksum
- A check is carried out to ensure that the checksum matches the transmitted data
- If the two do not match, the recipient cannot correct the error itself
- In this situation, the recipient asks the sender to retransmit the data

Check digits

- A check digit is a type of checksum
- A single digit is added to the transmitted data
- This reduces the number of different algorithms that could be used to calculate the value of the check digit
- Hence the variety of errors that the method can detect is limited

| | Can *detect* errors in transmission | Can *correct* errors in transmission | Efficiency |
|---|---|---|---|
| **Parity bit** | Yes<br>- but only if an odd number of bits are changed | No | Very efficient |
| **Majority vote** | Yes | Yes<br>- as long as the majority of bits remain unchanged | Inefficient<br>- each bit is sent multiple times |
| **Checksum** | Yes | No | Mostly efficient<br>- a complex algorithm could make the process less efficient |
| **Check digit** | Yes | No | Efficient<br>- the algorithms used to calculate the check digit are limited in complexity |