Unit 1 – Computer Systems

Data Representation

Text

ASCII

Each character in the character set is represented as a unique 8 bit code.
ASCII Code uses 1 bit as a parity bit to provide an error check

character	0	1	2	A	B	C	a	b	c
ASCII value	48	49	50	65	66	67	97	98	99

Advantages: Universal code promotes compatibility between software packages and allows data to be shared and exchanged.

Disadvantages: Limited character set. No styles or fonts.

RTF

Rich Text Format is an alternative representation that allows italics, bold and some additional fonts.

EBDIC

Extended Binary Coded Decimal Interchange Code.
An IBM designed representation for text.

Numbers

Binary

Represents Positive Integers only.

128	64	32	16	8	4	2	1
1	0	1	1	0	1	1	1

= 128 + 32 + 16 + 4 + 2 + 1 = 183

Min value		**Max value**
8 bits	0	256
16 bits	0	65535
n bits	0	$2^n - 1$

Sign And Magnitude

Simple scheme to represent positive and negative integers.
Top bit is used to indicate the sign (+ or -). The rest is the magnitude

- ✓ Represents Positive and Negative Integers
- × Arithmetic does not always work
- × Has two codes for zero…… +0 and -0
- × Does not use all codes efficiently so range is not optimal.

+/-	64	32	16	8	4	2	1
1	0	1	1	0	1	1	1

\- 32 + 16 + 4 + 2 + 1 = -55

Min value		**Max value**
8 bits	-127	127
16 bits	-32767	32767
n bits	$-(2^{n-1}-1)$	$2^{n-1} - 1$

2s Complement Represents positive and negative integers

- ✓ Represents Positive and Negative Integers
- ✓ Arithmetic works
- ✓ Uses all codes efficiently so range is optimal

This table illustrates simplest way to work with values.

-128	64	32	16	8	4	2	1
1	0	1	1	0	1	1	1

-128 + 32 + 16 + 4 + 2 + 1 = - 73

This table describes how 2s complement works.

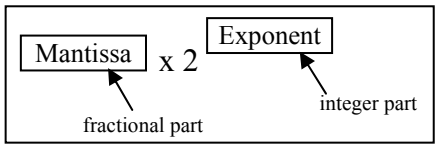
Example Change +55 into -55

'Flip the bits and add 1'

128	64	32	16	8	4	2	1	Binary value
0	0	1	1	0	1	1	1	55
1	1	0	0	1	0	0	0	Flip bits
						+	1	Add 1
-128	64	32	16	8	4	2	1	2s complement value
1	1	0	0	1	0	0	1	-55

Check: = -128 +64 + 8 +1 = - 55

Floating Point Represents Real numbers
Similar to Standard form



Example 8 bit value with 5 bit mantissa and 3 bit exponent

1	0	1	1	1	0	1	1

Interpreted as

Mantissa						Exponent		
1/2	1/4	1/8	1/16	1/32		4	2	1
1	0	1	1	1		0	1	1

$= {}^{1}/_{2} + {}^{1}/_{8} + {}^{1}/_{16} + {}^{1}/_{32}$ = 2 + 1

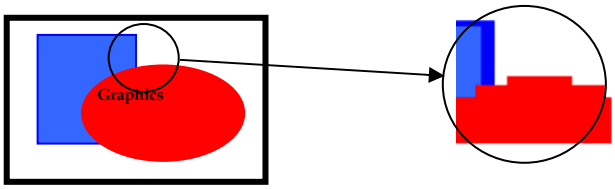
$= {}^{23}/_{32}$ = 3

$= 0.71875 \times 2^{3}$

= **5.75**

Graphics

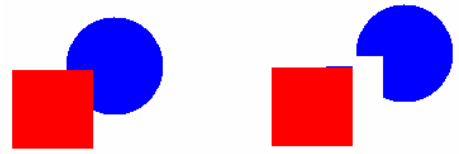
Bitmap



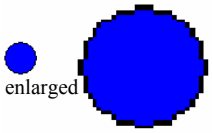
Bit depth	colours
1	b + w
2	4
3	8
8	256
10	1024
16	65536

- Image is divided into a 2D array of pixels
- Each pixel is stored as a binary code representing its colour
- Number of colours is determined by the bit depth
- Quality is determined by resolution dots per inch (dpi)

× Large file sizes as every pixel in image must be stored.
× Parts of image cannot be separated



× Image is resolution dependant.
  i Quality is fixed when image is created.
  ii Output quality not determined by output device but by image resolution
  iii If image enlarged then quality is reduced.



✓ Editing can be done at pixel level allowing good detail

1.5”

2.5”

250 dpi
1024 colours

Storage Calculation 1:

= 1.5 x 250 x 2.5 x 250 = 234375 dots

= 2343475 x 10 = 2343750 bits

Bit depth

= 234750 / 8 = 292968.75 bytes = **286.1 Kbytes**

2.0”

3..5”

300 dpi
256 colours

Storage Calculation 2:

= 1.5 x 250 x 2.5 x 250 = 630000 dots

= 630000 x 1 = 630000 bytes

Bit depth = 1 byte

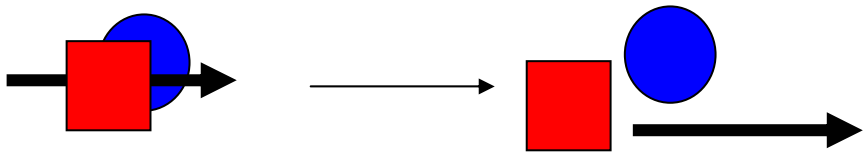
= 630000 / 1024 = **615.2 Kbytes**

Note: USUALLY exam questions use bit depth of 8bits (or 1byte)

Vector Graphics

Image stored as its objects and vectors
Example: Circle: centre_x, centre_y, fill_red, Line_blk, order_0

- ✓ File size smaller as only 'used' sections of image saved
- ✓ Editing can be done at attribute level
- ✓ Overlapping objects can be separated again (see diagram)



- ✓ Resolution independent so:
  - i no loss in quality when image enlarged
  - ii image displayed at resolution of output device
- × Not suitable for editing fine detail such as photographs

Standard File Formats

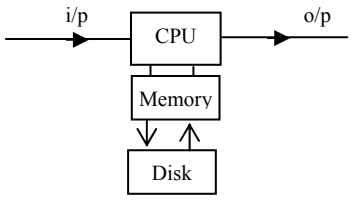
Standard representations exist for all types of data. These are used to promote compatibility between different software packages, allowing data to be shared.

Data Type	File Formats
Text	ASCII, RTF
Graphics	JPEG, GIF, TIFF
Audio	MP3, WAV, MIDI
Video	MPEG, REAL, WMF
Raw data	CSV, TSV

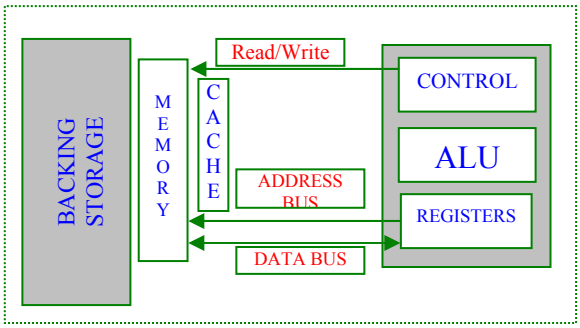
Low Level Machine

CPU

Simple View of CPU



Detailed View of CPU



Reasons for using a 2 state machine is a common exam question

Two State Machine

- ✓ **Only 4 rules for addition therefore simple circuitry**
- ✓ **Good tolerance as degraded 1 still recognised as a 1**
- ✓ **Simple to store data on magnetic media**
- ✓ **Simplicity - Only two voltage levels to generate and detect**

Control Unit Controls the fetching, decoding and executing instructions within the CPU. Controls the other parts of the CPU.

ALU Arithmetic and Logic Unit performs arithmetic operations and does logical comparisons.

Registers Individual storage locations on the processor used to hold instructions and data that are either waiting to be processed or that are the result of an operation.

Cache An area of fast access storage used to store frequently/recently used data

Data Bus Carries data and instructions between memory and CPU. 2-way bus.
The size of the data bus is usually called the word size and may indicate how much data is processed in a single operation.

Address Bus Used to pinpoint a location in memory. 1-way bus.
The width of the bus determines how many unique addresses may be created. This is called addressability.

Control Bus Not really a bus at all but a number of control lines, each with a different function.

Line	Function
Reset	Clears registers and starts loading instructions from preset location.
Clock	Regular pulse used to synchronise events in the CPU.
Read	Indicates data to be read from memory to CPU.
Write	Indicates data to be written from CPU to memory location.
Interrupt	Indicates some external event has occurred.
NMI	Non maskable interrupt. Interrupt that cannot be ignored.

Main Memory

RAM Used to hold programs and data currently being used
2 forms: **Static** and **Dynamic**
Dynamic is most common form because:
i Continually refreshed ii Simple Circuitry and Uses Less Power

ROM Used to hold Bootstrap Loader – a small program that loads up the remainder of operating system from backing store.

Addressability Calculation
Calculate the maximum memory size for a computer with a 24 bit Data Bus and 32 bit Address Bus.

Calculation: = 2^{32} x 24
= 103079215104 bits
= 12884901888 bytes
= 12582912 Mbytes
= 12.3 GBytes

System Performance

Measurement

Clock Speed Measured in Gigahertz. This tells you how fast the internal clock ticks. It does not give any indication of how much is being done on each clock tick so it is a poor measurement. It is only useful for comparing similar machines. A bit like the RPMs of a car engine, it tells you how fast the engine is running but not how fast the car is going.

MIPS Millions of Instructions per second. This is a measure of throughput. It indicates how many instructions the processor can handle per second. It does not indicate the complexity of instructions so is only useful for comparing similar systems.

FLOPS Floating Point Operations per Second. Better than MIPS as FP operations are more complex and require same 'effort' from all machines.

Benchmarks Small programs designed to test processor. Not always good test as manufacturers can design processor to perform well in recognised benchmark tests.

Real Programs Using databases and other real applications to see how fast processor can process complex tasks

Factors

Clock Speed — Increasing clock spooed will shorten time between events, increasing throughput.

Memory — Increasing main memory means more instructions can be held in fast access storage so less are loaded from backing store so faster processing.

Cache — This is memory with very short access time, between memory and CPU. Allows frequently used data to be accessed more quickly than reading from RAM. Needs to be kept quite small to keep seek time low.

Data Bus — Increasing width means more data read in each fetch so fewer fetches required to process each instruction.

Address Bus — Increasing width has no immediate effect. It does increase addressability meaning more main memory can be addressed. Indirect increase in performance

Peripherals — Slow peripherals may process input/output data slowly. Increasing the power of the interface may speed up performance. Two solutions are Buffering and Spooling

- **Buffering** — Data is sent in blocks to a reserved area of Memory. This storage may be on the device interface itself. Data is processed by the peripheral until the buffer is empty. The device interface can request more data when the buffer is empty.
- **Spooling** — tasks (e.g. print 'jobs') are sent to an area on Disk and put in a queue. The device then processes each job in the queue. This is often used in a network environment where many devices share the same peripheral. The queue is held on a print server so even when clients log out the task remains in the queue.

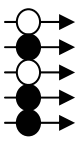
Interfaces

An interface is the hardware and software required to allow a device, such as a printer, to be connected and used with a computer.

Software component is called the **Driver Program**

The hardware part has several functions:

- Provides a physical connection
- Provides Status information…e.g. "Printer out of Paper" or "Device Off-Line"
- Provides Data Storage
- To carry out signal conversion…e.g. Serial to Parallel
- To perform device selection where 2 devices share the same bus

- Serial Line 
- Parallel Line 

Networks

There are 4 main topologies covered in this unit. Their relative advantages/disadvantages are described in the table.

Type of Network	Advantage	Disadvantage	Affect on Performance
Mesh	Fault in 1 cable doesn't affect network Messages are re-routed	Lots of wiring Expensive	Excellent Performance
Star	Short path between any two nodes Easy Expansion	Fault in central node means network unusable Congestion at central node	Control Computer gives more robust network but slows down comms between nodes
Bus	Fault in one station has no effect on rest of network Very easy to expand	All stations use the same line hence contention occurs	Instant access but high rate of data crash
Ring	Control system in charge of transmissions. Stations guaranteed access to transmission	Additional expense for control software and system May have to wait turn to transmit	Network down to add new station but thereafter few data crashes

Network Modes

Client Server

Server: Device that provides a network resource. Usually a more powerful computer with large storage and processing capabilities

Client: The combination of user and workstation. The client makes requests of the server. Client usually has a predefined level of access within a hierarchy.

Management of resources, backups, security and access is managed by a super user or administrator.

Peer to Peer

A small, TRUSTED network of users (max 9). All users of the network have equal status with no administrator. Everyone has access to resources on other stations but individuals are responsible for managing resources such as printers and making their own backups.

Data Integrity

Ensuring the accuracy of data when it is sent across a network. It involves error checking mechanisms to identify and rectify errors.

Parity Bit: An additional bit set to force an even number of 1s (Even Parity) or an odd number of 1s (Odd Parity). A simple scheme but only useful for spotting single bit errors.

Odd Parity better than even parity as it can spot a total channel failure.

Checksum: Numerical value of the data is added up to find a total and this is appended to the data. When the block of data is received it is recalculated to ensure both totals match. Uses up more data.

File Access Rights: Each user in a network has access rights allocated to them so that they can be restricted from reading or editing certain files. In addition, files themselves can have read/write permissions set to restrict certain users from changing or deleting sensitive files.

See also **File Locking** in outcome 2 notes

Software

Applications Software

This includes the general purpose packages used in most schools, businesses and industries such as word processing and database software.

Systems Software

Operating System

Software that manages the interaction between user and computer.
Software that hides underlying complexity from user.

Single User OS **Has 5 main components**

1. File Management
Manages creation deletion and updating of files within the hierarchy of a filing system.

2. Memory Management
Manages where programs and data are placed in memory so that programs and associated data don't overwrite one another.

3. Process Management
Manages which of the processes running gets scheduled processing time. It may involve prioritising processes so that critical processes are handled first.

4. Command Language Interpreter
Interprets commands made within the interface by the user and passes them onto the appropriate part(s) of the OS to act upon.

5. Input Output System
Handles the actual data transfers between memory and peripheral devices such as monitor and printer.

Network OS **Has 3 Additional components**

Multi User Access
Logging in users - verifying name and password. Managing users' access to files within the hierarchy of the network. E.g. an administrator should have complete access to files whilst a low priority user (like a pupil) should only have access to their own files.

Data Integrity
Where 2 or more users try to access the same shared file on the network, the file must be '**locked**' for editing so that only one user at a time can change the contents of the file.
also
Employing error checking mechanisms so that data sent across the network is accurate and error free.

Encryption
Encoding sensitive data using complex techniques so that it can pass through network safely and only be read by intended recipient.

Utility Software Small programs used to simplify tasks and aid in the maintenance of the computer system.

- Disk defragmenter
- Virus Checker
- File Registry Clearup
- Compiler and Interpreters

Programming Languages

Procedural — The program has a set start and end point and follows a prescribed algorithm. All possibilities must be catered for. Has logical and arithmetical operators included. Often used for general problem solving.
Examples: BASIC, PASCAL, COMAL

Declarative — Programs encode facts and relationships between entities within the program in order to build up rules and a knowledge base. A query can be written that can interrogate the knowledge base in order to draw conclusions and answer questions. Used for artificial intelligence. Uses recursion as a form of looping where a procedure calls itself.
Examples: LISP and PROLOG

Script — A language often built in as part of an applications package. The language allows low level access to the applications existing functions so that programs can be written to customise the application and add new features. MACROs can be written using a scripting language. Operating systems often allow commands to be combined in a script program.
Example: VBA (BEST EXAMPLE), DOS Script, UNIX script

Event Driven — Code is evoked when events within the program interface occur. Program code is usually procedural but may be executed with no specific start/end point. The program can respond to different events in different ways. An example of an event is a command button being clicked.
Example: Visual Basic

Factors Affecting Selection of Programming Language:

- Algorithmic Complexity
- General Application Area
- Developers Expertise
- Time and Financial Constraints
- Objects and Operations Involved

Translators

A translator is a program that translates high level languages into the many millions of binary machine code instructions that the CPU processes.

Interpreter

- Instructions are translated then executed line by line.
- Programs will run until an error is found
- Errors are reported as they are trapped by the translator
- Loops are translated each time around which is inefficient
- To Run program – Source code, and translator must be in memory so program takes up more storage and has slow execution time.
- Used program development

Compiler

- Whole program is translated into an object code program
- Any errors found will be reported at compilation time
- Program will not run until all syntax errors fixed
- To Run program – object code only is required so less memory required and execution speed is faster
- Used for final version of program but not for development