ZNOTES // A-LEVEL SERIES

visit www.znotes.org

Updated to 2017-19 Syllabus

CIE AS-LEVEL COMPUTER SCIENCE 9608

SUMMARIZED NOTES ON THE THEORY SECTION

TABLE OF CONTENTS

1. Information Representation

1.1 Number Representation

- **Denary:** normal integer numbers
- **Binary**
  - Values stored as a series of ones and zeroes
  - A single 0 or 1 is called a binary digit or bit.
  - Any group of 0s & 1s can represent a character

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

  - E.g. 65 in binary is 1000001
- **BCD** (Binary Coded Decimal):
  - Each digit in the number is written separately in a series of 4 bits
  - E.g. 398 in BCD
  - 3 = 0011 9 = 1001 8 = 1000
  - 398 = 001110011000
  - **Use of BCD:** used in business applications (arithmetic) where every digit has to be retained in a result
- **Hexadecimal Numbers:**
  - The base-16 number system (counting in 16)
  - After 9, numbers represented by letters from A to F

256	16	1
0	0	0

  - E.g. A5 in Denary = $(16\times10)+(1\times5)=165$
  - E.g. 65 in Hexadecimal = $65 \div 16 = 4\ Remainder\ 1$
    $\therefore = 41$ H
  - Quick way of translating hexadecimal to binary is by converting the hexadecimal to BCD
  - E.g. A5 in Binary: A = 1010 5 = 0101 A5 = 10100101
- **Two's Complement:**
  - We can represent a negative number in binary by making the most significant bit (MSB) a sign bit, which will tell us whether the number is positive or negative.

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

- Converting a negative denary number into binary Two's Complement:
  - Find the binary equivalent of the denary number
  - Add an extra bit before the MSB and turn that into 0
  - Flip all the bits: 0 to 1; 1 to 0
  - Add 1
- Converting Two's Complement to denary:
  - Flip all the bits: 0 to 1; 1 to 0
  - Add 1
  - Simple conversion from binary to denary (put a –ve)

1.2 Image Representation

- A **bitmapped image** is encoded by assigning a solid color to each pixel.
- **Pixels** are small blocks of addressable areas and the color they have is represented by binary & stored as bits
- A bitmapped image also contains a **file header** which are a few bytes of binary and represents basic information about the graphic, such as image resolution, size and number of colors.
- **Image Resolution:** the amount of pixels an image contains per inch or per centimeter.
- **Screen resolution:** the number of pixels per row by the number of pixels per column, e.g. HD is 720 by 1280 px

$$Total\ Number\ of\ Pixels = Width \times Height$$

- **Color depth:** number of bits used to represent the color of a single pixel
  - E.g. a 1 bit image can only store 2^1 pixels which equals to 2. Therefore, a 1 bit image is monochromatic
  - An image with *n* bits has 2^n colors per pixel
  - The higher the color depth, the better color quality
  - The higher the color depth, the larger the file size.

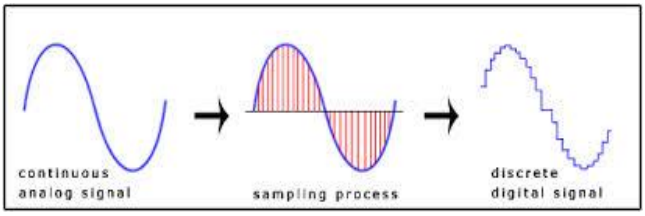
$$File\ Size = Number\ of\ Pixels \times Colour\ Depth$$

- **Vector graphics**: images defined using mathematics & geometry i.e. points, lines, curves & shapes/polygon(s)
- Objects and properties are stored mathematically.
- **Drawing list:** set of commands used to define a vector
- Vectors are scalable and do not pixelate like a bitmap image therefore are used by corporations to create logos as they can be resized without losing quality
- Bitmapped images are used by general computer users as they are not big in file size and can be manipulated.

1.3 Sound

- Sound are vibrations that travel through a medium
- Sound waves are continuous in nature, which means there is infinite amount of detail for a sound.
- **Analogue to Digital Converter (ADC):** converts analogue sound into digital signals which can be digitally stored
- **Digital to Analogue Converter (DAC)**: converts digital signals into analogue sound that can be output

- An analogue sound wave is picked up by a microphone and sent to an ADC in the form of analogue electrical signals. Once the sound wave is converted into a digital form it can be stored and manipulated
- Sound in analogue form can be represented by wave forms; the height of these waves can be **sampled** regularly with the height being represented by a bit-code



- **Sampling Rate:** number of samples taken per second
- A higher sampling rate means the waveform will be converted from analog to digital form more accurately.
- **Sampling Resolution:** no. of bits assigned to each sample
- The sampling resolution allows you to set the range of volumes storable for each sample
- The quality of a sound produced using a sound sampler depends on the sampling rate and sampling resolution
- Higher sampling rate/resolution means larger file size
- **Bit Rate:** number of bits required to store 1sec of sound

$$Bit\ Rate = Sampling\ Rate \times Sampling\ Resolution$$
$$File\ Size = Bit\ Rate \times Length\ of\ Sound$$

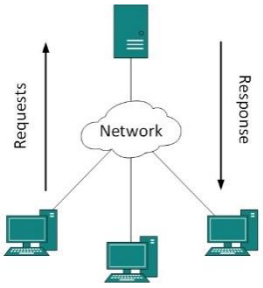
- **Lossless Compression:** type of compression that allows original data to be perfectly reconstructed from compression
  - **Run-length encoding:** compression in which sequences with same data value in many consecutive values are stored as a single data value and count
  - E.g. 00001234111111 can be written as (0-4)1234(1-6)
- **Lossy Compression:** type of compression in which file accuracy is low, but file size is smaller than lossless
  - **Perceptual coding:** works by reducing certain parts of a sound which are less audible to human hearing

1.4 Video

- **Frame Rate (FPS):** frequency at which frames in a video sequence are displayed
  - Higher frame rate, better quality video = greater size
- **Interlaced (e.g. 1080i):**
  - Old Technology (70 years ago)
  - Each field of a video image displays every other horizontal line of a complete image ∴ doubles FPS
  - Horizontal lines often visible to eye due to distortion
  - **Pros:** Refreshes faster, better visual smoothness and saves bandwidth
  - **Cons:** Becoming outdated and interlaced screen will not show fast moving objects clearly
- **Progressive (e.g. 720p):**
  - Excite every horizontal line simultaneously
  - Thus frame rate is the same as the number of individual pictures in a video sequence
  - This gives more detail in each frame
  - **Pros:** Crisp, detailed frames and is new and popular
  - **Con:** Rough frame transition
- **Inter-frame compression:**
  - Type of video compression and decreases file size
  - It removes neighboring frames which are similar
  - Some change in image data is redundant
  - How redundant the change in image between frames is determines the amount of compression possible
- **Temporal redundancy**: redundancy **between** frames
  - This is the correlation between adjacent data points
  - It is based upon abrupt transitions between frames
- **Spatial redundancy:** redundancy **within** a frame
  - An intercoded frame that is divided into macro blocks
  - Blocks are not directly encoded with raw pixel values
  - Encoder finds a block similar to the one in last frame
  - This frame is the reference frame
  - This process is done using algorithms
- **Multimedia Container Formats:**
  - Contains different types of data
  - Can be audio or video or both codecs
  - This interleaves the different type of data
  - The video is compressed into codecs
  - E.g. .avi, .mov, .mp4, .ogg, .rm
- **Lossy Data Compression:**
  - These programs eliminate unnecessary bits of data
  - Reduces data file size more than lossless
  - Unable to get back original version
- **Lossless Data Compression:**
  - Breaks data into smaller form of transmission
  - Allows recreation of original

2 COMMUNICATION & INTERNET TECHNOLOGIES

2.1 Client/Server Model



- Type of computing system in which one workstation serves requests of other systems
- Server is generally most powerful computer in network
- Clients are the individual components which are connected in a network.
- Clients rely on servers for resources, such as files, devices, and even processing power
- **Examples of networks:** file, application, printer, proxy
- **Internet:** global system of interconnected networks that uses standard Internet protocol suite (TCP/IP)
- **Transmission Control Protocol/Internet Protocol (TCP/IP):** communication language of the Internet;
- **World Wide Web:** system of interlinked hypertext documents accessed via the Internet
- **HTTP:** defines how messages are formatted and transmitted, and what actions web servers and browsers should take in response to various commands.

2.2 Hardware

Network:

- **Local Area Network (LAN):** not over large geo area
- **Wide Area Network (WAN):** formed by a number of LANs connected together; large geo area
- The Internet is a WAN therefore as more LANs are set up, it allows the Internet to expand and if they are maintained, they are in turn supporting the Internet

Routers:

- A device that forwards packets of data between networks using IP addresses
- Can be used to connect multiple network segments
- Can route packets of the same protocol over networks with dissimilar architecture over most efficient route

Gateways:

- A device used to connect networks with different architecture and protocols.
- When packets arrive at a gateway, software strips all networking information leaving only raw data
- Gateway translates data into new format and sends it on using networking protocols of destination system

Server

- Computer that runs the server program
- Servers run to serve requests of clients.
- Clients typically connect to server through the network

2.3 Communication Systems

The Public Service Telephone Network (PSTN):

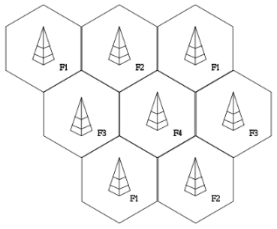
- Refers to all telephone networks in the world
- All networks connected together by switching centers
- Allows any telephone to communicate with another
- Internet connection using PSTN is known as dialup
- Data is transferred using existing telephone lines
- When data is being transmitted, the computer dials the network to set up a connection

Dedicated Line:

- A telecommunications path between two points
- Not shared in common among multiple users
- Allows you to maintain a continuous, uninterrupted presence on the Web
- Able to host websites as well browse

Cell phone network:

- Wireless network spread over land areas called cells
- Each cell is served by at least one fixed-location transceiver known as a base stations
- Each cell uses a different set of frequencies to avoid interference
- When joined together, cells provide radio coverage over a wide geographic area.
- Portable transceivers (e.g. mobile phones) are able to communicate with each other and also access the internet via base stations



2.4 Communication Methods

	Benefits	Drawbacks
Copper Cable	• Best conductor • Flexible • Safe - high melting point	• Doesn't perform well with small charges. • Affected by electromagnetism • Expensive
Fiber-Optic Cabling	• Thinner • Less signal degradation • Lightweight	• Needs expensive optical transmitters and receivers.

Radio waves	• Wireless • Can travel over large distances • Not expensive.	• Low frequency so transmit less data at one time. • Affected by radio stations with similar frequency
Micro-waves	• Wireless • Larger bandwidth	• Emitting towers expensive to build • Physical obstacles can interfere
Satellites	• Wireless • Cheap with long distance	• Easy to interfere • Expensive to set up

2.5 Bit Streaming

- **Bit stream:** contiguous sequence of bits, representing a stream of data, transmitted continuously over a communications path, serially (one at a time).
- **Real-time streaming:**
  - An event is captured live
  - The video signal is encoded to streaming media files
  - The encoded feed is then uploaded to a file server
  - Streaming servers duplicate the feed and send it to all clients requesting it in real time
- **On-demand streaming:**
  - Video is stored on a server as streaming media files
  - If client requests to watch a specific video, a bit stream is set up which transmits the saved video
- It is important for the client to have a fast **broadband speed** when **bit streaming** because the client has to download the bits and display it at the same time.
- Broadband speed required depends on type of stream user requires; if user requires better quality then higher speed needed because each frame larger in size
- Users who stream real-time need faster internet speeds in comparison to on-demand because there are greater number of users requesting same data simultaneously

2.6 IP Addressing

- **IP address:** numerical label assigned to each device (e.g. computer) participating in a computer network that uses the Internet Protocol
- The most common form of IP Address is "IPv4" which requires 32 bits (4 bytes) to store an address.

$$\underbrace{192 \cdot 168}_{\text{network}} \cdot \underbrace{12 \cdot 162}_{\text{host}}$$

- An IP address serves two principal functions:
  - host or network interface identification
  - location addressing
- When communicating, a device can send another a message by stating their IP address and requesting a message to be sent through the router.

Public IP Address	Private IP Address
• Address provided to home network by ISP • Address is unique throughout Internet • Allows two computers to identify each other	• Address issued by router to each device in home network • Address only unique in the network and cannot be accessed through the internet

- **Uniform Resource Locator (URL):** A character string referring to the location of an internet resource
- URLs allow us to specify the domain name and exact location of a resource on the internet.
- **For example:**

$$\underbrace{http://}_{\text{protocol}}\underbrace{commons.wikimedia.org}_{\text{hostname}}\underbrace{/wiki/File:George\_Clausen\_WWI\_poster.jpg}_{\text{location on server}}$$

- A URL can ∴ be summarized as:

$$protocol://hostname/location\_of\_file$$

- **Domain Name:** humanly-memorable names for Internet participants such as computers and networks. One domain name can be connected to multiple IP addresses
- **Domain Name Service (DNS):** naming system used for computers or resources having internet connection

$$\underbrace{windows.}_{\text{host}}\underbrace{microsoft.}_{\text{domain name}}\underbrace{com}_{\text{top level domain}}$$

2.7 Client and Server Side Scripting

- **Sequence of Events when Viewing a Website:**
  1. User specifies a URL in their client
  2. Client sends DNS lookup request to convert URL to an IP address and initiates a TCP connection to server
  3. Server acknowledges TCP connection, client sends HTTP requests to retrieve content for the URL.
  4. Server replies with content for web page and browser retrieves content from the HTTP packets and renders
- **Client-side script:** code that runs on client written using language supported by browser e.g. Javascript. Enables web pages to be scripted; to have different and changing content depending on user input, or other variables.
- **Server-side script:** code that runs on server written using language supported by server e.g. PHP. Used to provide interface for client & to limit client access to databases

3 HARDWARE

3.1 Input Devices

- **Keyboard:** device used to input text into a computer system. When a key is pressed, an electrical circuit is completed. Circuit transmits a binary signal to computer, using ASCII code which represent the key pressed.
- **Trackerball mouse:** pointing device consisting of a ball held by a socket containing sensors to detect a rotation of the ball about two axes.
- **Laser mouse:** pointing device which uses an infrared laser diode to illuminate the surface and a light sensor to detect movement relative to a surface
- **Flatbed scanner:** optically scans documents, & converts it to a digital image. Composed of a glass pane, bright light which illuminates pane and a moving optical array.
- **Sensor:** device that detects events or changes in physical quantities and provides a corresponding output, generally as an electrical or optical signal

3.2 Output Devices

- **Actuator:** type of motor that is responsible for moving or controlling a mechanism or system through which a control system acts upon an environment.
- **Printer:** output device that makes a persistent readable representation of graphics/text on physical media.
  - **Inkjet printer:** non-impact printers which work by propelling variably-sized droplets of liquid or molten material (ink) onto physical media
  - **Laser printer:** non-impact printers which work by repeatedly passing a laser beam back & forth over an electron-charged, cylindrical drum, to define a differentially-charged image. Drum selectively collects charged, powdered ink, and transfers image to loaded paper, which is then heated to fuse.
- **Speakers:** device which outputs sound by converting digital signals of sound from the computer to analogue

3.3 Secondary Storage Device

- **Optical disks:** flat, usually circular disc which encodes binary data in form of pits and lands. Pits = 0, due to lack of reflection and lands = 1, due to reflection when read
- **Hard disk:** data storage device consisting of rotating disks, platters, and magnetic heads arranged on an actuator arm to read and write data to the surfaces.
- **Flash memory:** solid state devices which have no moving parts and data is programmed onto them

3.4 Need for Primary and Secondary Storage

- Primary storage: computer's main memory; RAM & ROM
- Necessary for executing a program (RAM) or storing permanent system data (ROM)
- However RAM is volatile and ROM is non-editable ∴ important to have a more permanent, non-volatile form of storage to store e.g. documents and images = secondary storage device, generally a hard disk

3.5 RAM and ROM

RAM	ROM
• Random access memory • Used to store data during runtime of the computer • Can read & write data • **Volatile:** data gone when computer switched off	• Read only memory • Used to store system information • Can only read data • **Non volatile:** data is never deleted

3.6 Static and Dynamic RAM

Static RAM (SRAM)	Dynamic RAM
• Needs 6 to 8 transistors • More space in a chip • 4× more expensive • Consumes more power • Faster data access time • Lower storage capacity • Not possible to refresh programs	• Needs 3 to 4 transistors • Less space in a chip • Cheaper • Consumes less power • Slower data access time • Higher storage capacity • Memory can be deleted & refreshed while running a program

3.7 Logic Gates

- **Logic Gates:** use one or more inputs and produces a single logical output
- **AND gate:** If both inputs high, output is high

$A.B$

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

- **OR gate:** If either inputs high, output is high

$A + B$

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

- **NOT gate:** an inverter

$\bar{A}$

A	Output
1	0
0	1

- **NAND gate:**

$\overline{A.B}$

A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0

- **NOR gate:**

$\overline{A+B}$

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

- **XOR gate:**

$\overline{A \oplus B}$

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

- **Integrated circuit:** set of electronic circuits on one small plate ("chip") of semiconductor material (e.g. silicon)

4. Processor Fundamentals

4.1 Von Neumann Machine

- Von Neumann realized data and programs are indistinguishable & can therefore use the same memory.
- Von Neumann architecture uses a single processor.
- It follows a linear sequence of fetch–decode–execute operations for the set of instructions i.e. the program.
- In order to do this, the processor has to use registers.

4.2 Registers

- **Registers:** an extremely fast piece of on-chip memory, usually 32 or 64 bits in size for temporary storage
- Registers are outside the immediate access store and consequently allow faster access to the data they store.

Special registers have special purposes:

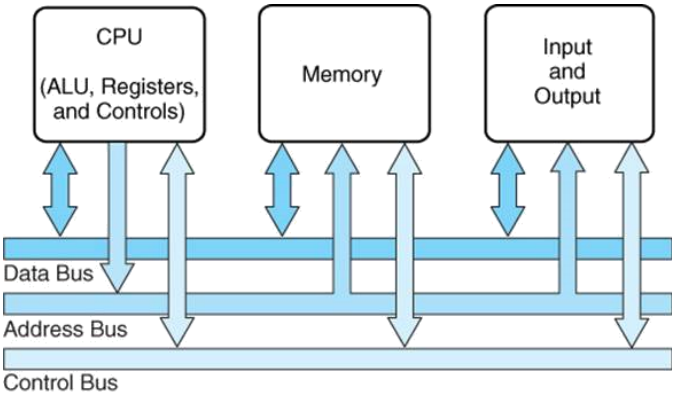
- **Program counter (PC):** keeps track of where to find next instruction so that a copy of the instruction can be placed in the current instruction register.
- **Memory data register (MDR):** acts like a buffer & holds anything copied from memory ready for processor to use
- **Memory address register (MAR):** used to hold memory address that contains either the next piece of data or an instruction that is to be used.
- **Index register (IR):** a microprocessor register used for modifying operand addresses during run of a program. Used if address indirect; a constant from the instruction added to contents of IR to form address to operand/data
- **Current instruction register (CIR):** holds the instruction that is to be executed.
- **Status register (SR):** holds results of comparisons to decide later for action, intermediate results of arithmetic performed and any errors occurred during arithmetic
- **General-purpose register:** One or more registers in the CPU that temporarily store data
- **Accumulator:** single general-purpose register inside ALU. It is a single general-purpose register where all values held when processed by arithmetic & logical operations.

4.3 The Processor

- **Arithmetic and logic unit (ALU):** part of processor that performs arithmetic calculations & logical decisions
  - Arithmetic operations: add, subtract, multiply etc.
  - Logical operations: comparing binary patterns and making decisions
- **Control unit:** part of processor that fetches instructions from memory, decodes them & synchronizes operations before sending signals to other parts of the computer.
- **System clock:** timing device connected to the processor that synchronizes when fetch-execute cycle runs

4.4 System Buses

- **Bus:** set of parallel wires that connect various components & provides communication between them



- **Data bus:** bi-directional, used to carry data and instructions between system components. The memory data register (MDR) is at one end of the data bus.
- **Address bus:** unidirectional, carries address of main memory location or input/output device about to be used from processor to memory address register (MAR)
- **Control bus:** bi-directional, used to send control signals from control unit to ensure access/use of data & address buses by components of system does not lead to conflict

4.5 Performance of a Computer System

- **Clock speed:** the number of cycles that are performed by the CPU per second. Faster clock speed means processes of fetch, decode and execute occur faster however faster clock speed causes processor to heat up.
- **Bus width:** determines maximum possible memory capacity of the system. Wider bus width means more bits can be transferred simultaneously.

4.6 Peripheral Devices

- **Peripheral devices:** hardware device outside the CPU
- Cannot be connected directly to processor ∴ processor controls and communicates through an I/O controller
- **I/O controller:** electronic circuit board consisting of:
  - Interface allowing connection of controller to system
  - Set of data, command and status registers
  - **I/O Port:** Interface that enables the connection of the controller to the device
- I/O ports allow I/O devices to be connected to the CPU without having to have specialized hardware for each one i.e. a USB port can connect many different I/Os

4.7 Fetch-Execute Cycle

- **The fetch stage:**
  - Copy the address that is in the PC into the MAR
  - Increment the PC (ready for the next fetch)
  - Load the instruction in address of MAR into the MDR
  - Load the instruction in MDR into the CIR
- **The decode stage:**
  - Identify type of addressing used by instruction
  - *Direct address:* load copy into MAR & retrieve data
  - *Indirect address:* add address to IR, copy result to MAR
  - Retrieve contents of new address
  - Decode the instruction
- **The execute stage:**
  - If instruction is a jump instruction, load address operand into PC then go to start
  - If not, execute instruction then go to start

4.8 Register Transfer Notation (RTN)

- Here the cycle is depicted by RTN:
  - MAR ← [PC]
  - PC ← [PC] + 1
  - MDR ← [[MAR]]
  - CIR ← [MDR]
  - Decode
  - Execute
  - Return to start
- Square bracket denote value currently in that register
- Double square bracket means the CPU must do a logical process and then copy this value

4.9 Interrupts

- A signal from some device or source seeking the attention of the processor
- Interrupt register: a special register in the CPU
- Before each cycle, the **interrupt register** is checked.
- Depending on the priority of the interrupt, the running process is saved and the control of the processor is passed to the interrupt handler

Interrupt handler a.k.a. Interrupt Service Routine (ISR):

- Different ISRs are used for different source of interrupt
- Typical sequence of actions when interrupt occurs:
  - Current fetch-execute cycle completed
  - Contents of all registers, especially PC, stored away
  - Source of interrupt identified
  - If low priority: then disabled
  - If high priority: PC loaded with address of relevant ISR
  - ISR executed
  - All registers except for PC are restored to original
  - Interrupts re-enabled
  - PC is then restored
  - Return to start of cycle

4.10 Processor's Instruction Set

Op Code	Operand	Explanation
		Addressing
LDM	#n	**Immediate:** Load n into ACC
LDD	<address>	**Direct:** load contents at address into the ACC
LDI	<address>	**Indirect:** Load contents of address at given address into ACC
LDX	<address>	**Indexed:** Load contents of given address + IR into ACC
		Data Movement
STO	<address>	Store contents of ACC into address
		Arithmetic Operations
ADD	<address>	Add contents of register to ACC
INC	<register>	Add 1 to contents of the register
		Comparing
CMP	<address>	Compare contents of ACC with that of given address
CMP	#n	Compare contents of ACC with n
		Conditional Jumps
JPE	<address>	Jump to address if compare TRUE
JPN	<address>	Jump to address if compare FALSE
		Unconditional Jumps
JMP	<address>	Jump to given address
		I/O Data
IN		Input any character and store ASCII value in ACC
OUT		Output character whose ASCII value is stored in ACC
		Ending
END		Return Control to operating system

 # denotes immediate addressing
B denotes a binary number, e.g. B01001010
& denotes a hexadecimal number, e.g. &4A

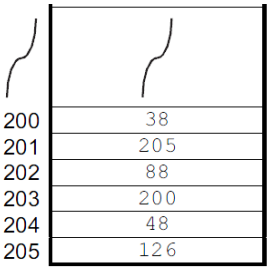
4.11 Addressing

Indirect Addressing: The address to be used is at the given address. Load contents of this second address to the ACC

- Example:

```
LDI 203
```

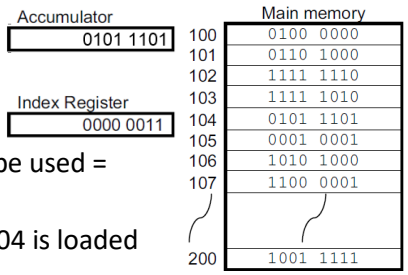
  - Go to location 203
  - Contents of location 203 is 200 so go to location 200
  - From location 200, loads contents into accumulator



Indexed addressing: form the address to be used as <address> + the contents of the IR (Index Register)

- **Example:**

```
LDX 101
```



  - Index Register = 3
  - Therefore address to be used = 101 + 3 = 104
  - Contents of address 104 is loaded into the Accumulator

Relative addressing: next instruction to be carried out is an offset number of locations away, relative to address of current instruction held in PC; allows for relocatable code

- Example of jumps in pseudocode:

```
if accumulator == 2 jmp + 3
else jmp + 5
carryon
```

  - 1st line shows a conditional jump, next instruction 3 lines away if accumulator contains 2
  - 2nd line shows an unconditional jump where 5 lines are skipped and the code carries on, avoiding some lines
- **Conditional jump:** has a condition that will be checked (like using an IF statements)
- **Unconditional jump:** no condition to be followed, simply jump to the next instruction as specified

4.12 Assembly Language and Machine Code

- **Assembly language:** low-level programming language with instructions made up of an op code and an operand
- **Machine code:** code written in binary that uses the processor's basic machine operations
- **Relationship:** every assembly language instruction translates into one machine code instruction (1 to 1)
- **Symbolic addressing:**
  - Mnemonics used to represent operation codes e.g. IN
  - Labels can be used for addresses e.g. PRICE
- **Absolute addressing:** a fixed address in memory
- **Assembly directives** are used to specify:
  - Starting addresses for programs
  - Starting values for memory locations
  - Specify the end of program text
- **Macros:** sequence of instructions, assigned by a name and could be used anywhere in the program. Macro instructions are expanded into a group of instructions.

- **Assembler:**
  - Software that changes assembly language into machine code for the processor to understand
  - The assembler replaces all mnemonics and labels with their respective binary values
  - These binary values are predefined before by the assembler software

Assembly processes:

- **One pass assembler:**
  - Assembler converts mnemonic source code into machine code in one sweep of program
  - Cannot handle code that involves forward referencing
- **Two pass assembler:** software makes 2 passes thru code

  On the first pass:
  - Symbol table created to enter symbolic addresses and labels into specific addresses
  - All errors are suppressed

  On the second pass:
  - Jump instructions can access memory addresses from the table
  - Whole source code translate into machine code
  - Error reported if they exist

5 System Software

5.1 Operating System

- A set of programs designed to run in the background on a computer system which
  - Controls operation of computer system
  - Provides a user interface
  - Controls how computer responds to user's requests
  - Controls how hardware communicate with each other
  - Provides an environment in which application software can be executed

The operating system (OS) must provide:

- Management of resources of the computer system:
  - Processor management: for multiprogramming decide which job will get the next use of the processor
  - File management: maintaining a list of files, directories and which fi le allocation units belong to which fi les
  - Input/output management: control of all input and output devices attached to the computer
- An interface between the user and the machine
- An interface between applications software & machine
- Security for the data on the system
- Utility software to allow maintenance to be done.

5.2 Utility Programs

- Type of system software designed to help analyze, configure, optimize and maintain computer.

Disk formatter:

- Carries out the process of preparing a data storage device such as a hard disk drive, for initial use
- Formatting, process where the computer 'draws lines' on the surface of disk surface to split it into small areas.

Virus checker:

- Finds and removes computer viruses from a computer
- When files sent from one computer to another, possible it may contain a virus that can infect receiving computer
- Virus checker keeps a constant check on files, searching for viruses and deletes it if found.

Defragmenter software:

- Reorganizes files & unused space on hard disk so operating system accesses data quicker
- Files can be big so have to be stored in multiple sectors
- Fragmentation, contents of a file are scattered across two or more noncontiguous sectors
- Fragmentation slows down disk access and thus the performance of the entire computer.
- Defragmenting disk: reorganizing files so they are stored in contiguous sectors

Disk content analysis software:

- Software utility for visualization of disk space usage
- Gets size for each folder and files, and generates a graphical chart showing disk usage distribution according to folders or other user defined criteria.

File compression:

- Reduces the size of a file by cutting out much of the duplication of data in the file.
- Causes improvements in the computer's performance by reducing the data that needs to be stored

Automatic Backup:

- Duplicate important data in the event of a hard drive failure, user error, disaster or accident.
- Can be used for individual computers or for an enterprise's computers and servers.
- Software can be set to backup selected files and folders on a scheduled basis or as required.

5.3 Library Programs

- Collections of resources used to develop software
- Made up of pre-written code & functions, subroutines
- Provide services to other more complex programs

- Example: the design of a program running on Windows 7
  - All programs run on Windows use Windows GUI library
  - Produces same "feel" when running any program

Advantages	Disadvantages
• Less time consuming • Reduces need for testing • Better/advanced engines	• Prone to viruses • May require manual tweak - tedious

Dynamic Link Library Files (DLL):

- A file acting as a library that contains code and data
- Can be used by different programs running simultaneously ∴ reduces strain on memory
- Each DLL file can be seen as modules in a more complex program, making it easier to install and run updates

5.4 Language Translators

- **Assembler:**
  - Translates assembly language into machine code
  - The mnemonics used translates into machine opcodes
  - Process simple because assembly language has a one-to-one relationship with machine code.
  - Assembler basically translate mnemonics into binary
- **Compiler:** translation of a high-level language program
  - Translates high level code into low level code
  - Reason for translating: create an executable program

Advantages	Disadvantages
• .exe easily distributed • With .exe, don't need compiler software • No source code – users not able to edit/plagiarize	• Only be produced when all errors are fixed • Development process long – locating errors • Uses a lot of resources

- **Interpreter:** execution of a high-level language program
  - Program translated line-by-line

Advantages	Disadvantages
• Can run program any time, even before code finished • Debugging easier/faster	• Execution very slow – translated each time run • Interpreter needed • No .exe produced

- High level language programs may be partially compiled and partially interpreted, such as Java

6 Security, Privacy and Data Integrity

- **Data security:** making sure that data is protected against loss and unauthorized access.
- **Data integrity:** making sure that the data is valid.
- **Privacy of Data:** deals with the ability to determine what data is shared with the third party

6.1 Security of Data and Computer System

Security of Data	Security of System
• Protection of **data** on a computer system • To prevent corruption of data and prevent hackers from using data • E.g. encryption	• Protection of the **computer system** • To prevent access of viruses to the system and prevent hackers to enter your computer system • E.g. ID & Password

6.2 Security Measures for Computer Systems

- **User Accounts:**
  - Each user has an ID and password which are the log in details for a his user account
  - User assigned privilege which gives him access to only his workspace, preventing him to have admin rights.
  - Can assign privilege to files so users with low privileges do not have access.
- **Firewalls:**
  - Set of programs, located at network gateway, that protects the resources of a private network from users from other networks
  - Filters information incoming to computer system
  - Information packets can be flagged by firewall and thus not allowed through
- **Authentication techniques:**
  - Digital signatures: a unique personal trait required by a computer to access certain data
  - Password: case-sensitive, unique string of letters, numbers and special characters.
  - Biometric scans e.g. retina, fingerprint, palm

6.3 Security Measures for Data

- **Data backup:** an exact copy of an original piece of data in case the original is lost or corrupted
  - Can be made on any type of storage device
  - May be within the same computer system or at a different site

- **Disk-mirroring strategy:** real-time strategy that writes data to two or more disks at the same time.
  - If one fails the other is still there to be read off of
  - It can be done using more than one extra hard disk
- **Encryption:** conversion of data to code by encoding it
  - Data stored in an incomprehensive state
  - Doesn't prevent illegal access but appears meaningless
  - Necessary to use decryption software to decode data
  - Encryption keys complex algorithms; cannot crack
- **Access rights to data (authorization):** different users are assigned different authorization levels which prevent them from accessing all the data so increase security

6.4 Errors that can Occur

Data held in a computer system may become corrupted in different ways and at many stages during data processing:

- **Errors on input:** data keyed in wrongly, a batch of data could be lost or accidently entered twice
- **Errors in operating procedures:** update master file twice
- **Program errors:** could lead to a corruption of files which may only surface later on
- **Viruses:** files can be corrupted or deleted when a disk is infected with a virus
- **Transmission errors:** interference in communications link may cause bits to be wrongly received

6.5 Data Validation

Data validation: checks data entered is sensible

- **Range check:** data must be between a set of values
- **Character check:** data must have correct character type
- **Format check:** data must follow correct pattern/order
- **Length check:** data must have an exact no. of characters
- **Existence check:** data entered must exist (e.g. barcode must correlate to an item)
- **Check digit:** 1 digit is used to be answer to an arithmetic operation of other digits in data. If not matched then data entered incorrectly

Data can be valid but this doesn't mean data is accurate

6.6 Data Verification for Data Entry

Data verification: checks data entered is accurate

- Person manually compares original data with that entered to check if correct
- Enter data into computer twice and compares.
- If differences found, go back to raw data to fix error

6.7 Data Verification during Data Transfer

- Errors may occur when data moved from one point to another point in system. Following find errors:

Parity Check:

- All data transmitted as bits
- Number of 1s in a byte must always be either an odd number or an even number
- Parity can be set either as even or odd
- Example: two communicating devices decide there will always be an odd number of 1s. A byte is received that has an even number of 1s so error must have occurred and receiving device would ask for it to be sent again
- Used also when data sent between parts of the CPU
- Not foolproof: if 2 errors made, data accepted

Checksum Check:

- Data sent from one place to another as block of bytes rather than individual bytes
- Computer adds together all bytes being sent
- Any bits lost at most-significant end as carry ignored so answer is an 8 bit number
- Checksum calculated before and after data sent
- If two bytes different, error occurred therefore block of bytes must be sent again

7 Ethics and Ownership

- **Computer Ethics**: how computing professionals should make decisions regarding professional & social conduct.

7.1 ACM/IEEE Software Engineering Code of Ethics

Eight ethics principles of software engineers (SEs):

- **Public:** act consistently in the public interest
- **Client:** act in the best interests of client
- **Employer:** act in the best interests of employer
- **Product:** software and related modifications meet highest possible standards
- **Judgment:** maintain integrity and independence in their professional judgment
- **Management:** team leaders should subscribe to and promote an ethical approach to the management of software development and maintenance.
- **Profession:** advance integrity & reputation of profession
- **Colleague:** be fair to & supportive of colleagues
- **Self:** participate in lifelong learning regarding practice of profession

7.2 Ownership

- **Data ownership:** having legal rights and complete control over a single piece or set of data elements.
- Copyright gives the creators of some types of media rights to control how they're used and distributed.
- Programming ideas and methods can be stolen by competitors, software can easily be copied and bootlegged (sold illegally) hence legislation is needed to protect the ownership, usage and copy right of data.

7.3 Software Licensing

Free Software Foundation:

- License gives user freedom to run, copy, distribute, study, change and improve software.
- Condition: any redistributed version of software must be distributed with original terms of free use, modification, and distribution (aka copyleft)

The Open Source Initiative:

- Source code of an open-source software is readily available to users under a copyright; does not enable user to re-distribute the software
- Concept of open-source program relies on fact that a user can review a source-code for eliminating bugs in it

Shareware:

- Demonstration software that is distributed for free but for a specific evaluation period only
- Distributed on trial basis and with an understanding that sometime later a user may be interested in paying for it
- Used for marketing purposes

Commercial: Requires payment before it can be used, but includes all program's features, with no restrictions

8 Database and Data Modelling

8.1 File Based System

- Data stored in discrete files, stored on computer, and can be accessed, altered or removed by the user

Disadvantages of File Based System:

- No enforcing control on organization/structure of files
- Data repeated in different files; manually change each
- Sorting must be done manually or must write a program
- Data may be in different format; difficult to find and use
- Impossible for it to be multi-user; chaotic
- Security not sophisticated; users can access everything

8.2 Database Management Systems (DBMS)

- **Database:** collection of non-redundant interrelated data
- **DBMS:** Software programs that allow databases to be defined, constructed and manipulated

Features of a DBMS:

- **Data management:** data stored in relational databases - tables stored in secondary storage
- **Data dictionary** contains:
  - List of all files in database
  - No. of records in each file
  - Names & types of each field
- **Data modeling:** analysis of data objects used in database, identifying relationships among them
- **Logical schema:** overall view of entire database, includes: entities, attributes and relationships
- **Data integrity:** entire block copied to user's area when being changed, saved back when done
- **Data security:** handles password allocation and verification, backups database automatically, controls what certain user's view by access rights of individuals or groups of users

Data change clash solutions:

- Open entire database in **exclusive mode** – impractical with several users
- **Lock all records** in the table being modified – one user changing a table, others can only read table
- **Lock record** currently being edited – as someone changes something, others can only read record
- User specifies **no locks** – software warns user of simultaneous change, resolve manually
- Deadlock: 2 locks at the same time, DBMS must recognize, 1 user must abort task

Tools in a DBMS:

- **Developer interface:** allows creating and manipulating database in SQL rather than graphically
- **Query processor:** handles high-level queries. It parses, validates, optimizes, and compiles or interprets a query which results in the query plan.

8.3 Relational Database Modelling

- **Entity:** object/event which can be distinctly identified
- **Table:** contains a group of related entities in rows and columns called an entity set
- **Tuple:** a row or a record in a relation
- **Attribute:** a field or column in a relation

- **Primary key:** attribute or combination of them that uniquely define each tuple in relation
- **Candidate key**: attribute that can potentially be a primary key
- **Foreign key:** attribute or combination of them that relates 2 different tables
- **Referential integrity:** prevents users or applications from entering inconsistent data
- **Secondary key:** candidate keys not chosen as the primary key
- **Indexing:** creating a secondary key on an attribute to provide fast access when searching on that attribute; indexing data must be updated when table data changes

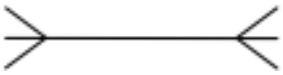
8.4 Relational Design of a System

ONE-TO-ONE

ONE-TO-MANY

MANY-TO-ONE

MANY-TO-MANY



8.5 Normalization

1st Normal Form (1NF): contains no repeating attribute or groups of attributes. Intersection of each tuple and attribute contains only 1 value. Example:

DELNOTE

Num	CustName	City	Country	ProdID	Description
005	Bill Jones	London	England	1	Table
005	Bill Jones	London	England	2	Desk
005	Bill Jones	London	England	3	Chair
008	Mary Hill	Paris	France	2	Desk
008	Mary Hill	Paris	France	7	Cupboard
014	Anne Smith	New York	USA	5	Cabinet
002	Tom Allen	London	England	7	Cupboard
002	Tom Allen	London	England	1	Table
002	Tom Allen	London	England	2	Desk

2nd Normal Form (2NF): it is in 1NF and every non-primary key attribute is fully dependent on the primary; all the incomplete dependencies have been removed. Example:

DELNOTE

Num	CustName	City	Country
005	Bill Jones	London	England
008	Mary Hill	Paris	France
014	Anne Smith	New York	USA
002	Tom Allen	London	England

DEL_PROD

Num	ProdID
005	1
005	2
005	3
008	2
008	7
014	5
002	7
002	1
002	2

PRODUCT

ProdID	Description
1	Table
2	Desk
3	Chair
7	Cupboard
5	Cabinet

3rd Normal Form (3NF): it is in 1NF and 2NF and all non-key elements are fully dependent on the primary key. No inter-dependencies between attributes.

- MANY-TO-MANY functions cannot be directly normalized to 3NF, must use a 2 step process e.g.

becomes:

Example:

DELNOTE

Num	CustName	City
005	Bill Jones	London
008	Mary Hill	Paris
014	Anne Smith	New York
002	Tom Allen	London

CITY_COUNTRY

City	Country
London	England
Paris	France
New York	USA

DEL_PROD

Num	ProdID
005	1
005	2
005	3
008	2
008	7
014	5
002	7
002	1
002	2

PRODUCT

ProdID	Description
1	Table
2	Desk
3	Chair
7	Cupboard
5	Cabinet

8.6 Data Definition Language (DDL)

- Creation/modification of the database structure using this language - written in SQL
- **Creating a database:**

```
CREATE DATBASE <database-name>
```

- **Creating a table:**

```
CREATE TABLE <table-name> (…)
```

- **Changing a table:**

```
ALTER TABLE <table-name>
ADD <field-name><data-type>
```

- **Adding a primary key:**

```
PRIMARY KEY (field)
```

- **Adding a foreign key:**

```
FOREIGN KEY (field) REFERENCES <table>(field)
```

- **Example:**

```
CREATE DATABASE 'Personnel.gdb'
CREATE TABLE Training
(EmpID      INT NOT NULL,
CourseTitle VARCHAR(30) NOT NULL,
CourseDate  Date NOT NULL,
PRIMARY KEY (EmpID, CourseDate),
FOREIGN KEY (EmpID) REFERENCES Employee(EmpID))
```

8.7 Data Manipulation Language (DML)

- Query and maintenance of data done using this language – written in SQL

Queries:

- **Creating a query:**

```
SELECT <field-name>
FROM <table-name>
WHERE <search-condition>
```

- **SQL Operators:**

=	Equals to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
<>	Not equal to
IS NULL	Check for null values

- **Sort into ascending order:** `ORDER BY <field-name>`
- **Arrange identical data into groups:**

```
GROUP BY <field-name>
```

- **Joining together fields of different tables:**

```
INNER JOIN
```

Data Maintenance:

- **Adding data to table:**

```
INSERT INTO <table-name>(field1, field2, field3)
VALUES (value1, value2, value3)
```

- **Deleting a record:**

```
DELETE FROM <table-name>
WHERE <condition>
```

- **Updating a field in a table:**

```
UPDATE <table-name>
SET <field-name> = <value>
WHERE <condition>
```

CIE AS-LEVEL COMPUTER SCIENCE//9608