0478 Syllabus Paper 1 Review Notes

created by **Hardy Wen**

- 前段时间在看大佬写的0610和0620的Syllabus Change，很认同他强调的Syllabus的重要性

- 这个是我在对着syllabus过点的时候记录的，里边儿的定义能syllabus上面取就会从上面取，自己加了一些 syllabus上面提到但是没有给到解释的概念。在答题的时候请根据题目给的背景进行答题并且给到相关例子就 再好不过了。

    - syllabus: [595424-2023-2025-syllabus.pdf (cambridgeinternational.org)](#)

1. Data Representation

Number Systems

- *Why do computers use* ==binary== *data*?

    - computers **can only** process binary

    - data is processed using **logic gates** and stored in **registers**

- *Why* ==hexadecimal== *is used to represent data*?

    - used in **HTML Color code, error code, IPv6, MAC address**

    - **Easier** to **read, write & understand** for humans

        - as it is a **shorter** representation of the binary

- *What is an* ==overflow==?

    - a computer or a device has a predefined limit that it can represent or store

    - an overflow occurs when a value **outside this limit** should be returned

Text, Sound, Image

- *Why is a* ==character set== *used?*

    - text is converted to binary, a character set is a **dictionary** of characters and their corresponding representations in binary form

    - **Unicode** allows for a **greater range** of characters and symbols than **ASCII**

        - it can represent different languages and emojis

        - it requires **more bits** per character

- *How is* ==sound== *represented?*

    - a sound wave is converted to **binary**

    - using

        - **sample rate**: the number of samples taken in a second

        - **sample resolution**: the number of bits per sample

    - **accuracy** depends on sample rate and resolution

- *How is the* ==image== *represented?*

- an image is **a series of pixels** that are converted to **binary** to be processed by the computer
  - using
    - **resolution**: the number of pixels in the image
    - **colour depth**: the number of bits used to represent each color (usually a power of 2)
  - **file size and quality** depend on resolution and colour depth

Data Compression

- bit < nibble (4 bits) < byte (8 bit) < KiB (1024 byte) < MiB < GiB < TiB < PiB < EiB

- *Why do we use <mark>data compression</mark>?*
  - **less bandwidth** required
  - **less storage space** required
  - **less transmission time** required

2. Data Transmission

Data transmission methods

- *What are <mark>data packets</mark>?*
  - Data is broken down into **packets** to be transmitted
  - a packet includes
    - a **packet header**, including
      - destination address
      - packet number
      - originator's address
    - a **packet number**, the number of the packet in a series of packets to help reconstruct the data
    - a **trailer**, including the sign marked that it is the end of the packet and error-checking information
- *What is the process of <mark>packet switching</mark>?*
  - data is broken down into packets
  - each packet should take a different route
  - a router controls the route a packet takes
  - packets may arrive out of order
  - once the last packet has arrived, packets are reordered
- *What are the methods for data transmission?*
  - **serial**: **one bit** over a **single wire** at a time
    - less chance of skewed, long distance
    - slow

- **parallel**: **multiple bits** over **multiple wires** at a time
    - fast
    - higher chance of skewed, data-interference (not used for long distances)
  - **simplex**: data transmitted in **one direction only**
  - **duplex**: data transmitted in **both directions**, but **one at a time**
  - **full-duplex**: data transmitted in **both directions** at the **same time**
- *How to evaluate the* <mark>*Universal Serial Bus (USB)*</mark>*?*
  - a type of **serial** transmission
    - A **USB** interface includes items such as a **USB port** (USB 接口), **USB cable** (USB 线), **USB device** (USB 设备)
  - advantages
    - **impossible** to be connected in a wrong way
    - **industry-standar**d so can be widely used
    - **fast**
  - disadvantages
    - **limited length** to **5 meters**
    - limited lifespan

Error Checking

- *Why do we check errors in data transmission?*
  - errors can occur during data transmission due to **interference**
    - e.g., data loss, data gain, and data change
- *What are the methods for* <mark>*error checking*</mark>
  - **Parity check (odd and even)**
    - parity bit -> 剩下的数据加起来看奇偶和parity bit是否一致
  - **Checksum**
    - checksum value -> 用特殊算法算这个checksum看和传来的是否一致
  - **Echo check**
    - 把数据传回发送者让发送者比较
- *What is* <mark>*check digit*</mark>*?*
  - 预先算好check digit值和数据存储，当数据再一次被输入时再算一遍check digit并和之前的比较
  - examples: **ISBN** and **Barcodes**
- *What is* <mark>*ARQ*</mark>*?*
  - **Positive ARQ**

- 发包 -> 接受者检查无误发positive acknowledgement -> 发下一个包
发包 -> 有误 -> 发送者在timeout内没收到positive acknowledgement -> 一直重新发同样的包直到它收到了acknowledgement或者超过了重新发送设定次数
    - **Negative ARQ**
        - 发包 -> 接收者检查无误则不动 -> 发送者超过timeout没收到回复就发下一个包
        - 发包 -> 有误 -> 接收者发送negative acknowledgement -> 发送者重发

Encryption

- *Why do we need <mark>encryption</mark>?*
    - data that is transmitted often needs to be protected during transmission because data is valuable (imagine your bank account and password)
- *What are the types of <mark>encryption</mark>?*
    - **Symmetric**
        - public key
    - **Asymmetric**
        - both public key and private key
        - encrypted by the public key, decrypted by the private key

3. Hardware

Computer Architecture

- *What is the <mark>role of a CPU</mark>?*
    - it **processes instructions and data** that are <u>input into the computer</u> so that *the result can be output*
- *What is a <mark>microprocessor</mark>?*
    - a type of **integrated circuit** on a **single chip**
- *What are the <mark>components of CPU</mark>*
    - **Arithematic Logic Unit (ALU)**: performs all the mathematical and logical operations
    - **Control Unit (CU)**: controls all the operations of the CPU
    - **Registers**
        - **Program Counter (PC)**: stores **address** of <u>the next instruction</u> to be located in RAM
        - **Memory Address Register (MAR)**: holds the **address** of the data or instructions to be located in RAM
        - **Memory Data Register (MDR)**: holds the **data or instructions** that are fetched from the RAM
        - **Current Instructions Register (CIR)**: built in **CU**, holds the **current instruction** being processed
        - **Accumulator (ACC)**: build in **ALU**, stores all the interim calculation results

- Buses
    - **Data Bus**: used for the transmission of **data and components**
    - **Control Bus**: used for the transmission of **control signals** -> what components should do
    - **Address Bus**: used for the transmission of **addresses**
- *What are the process of <mark>fetch-decode-execute cycle</mark>?*
    - the instruction is fetched **from the RAM into the CPU**
    - the instruction is decoded within the CPU
        - the **CU** uses an **instruction set** to decode the instructions into readable commands that are in machine code
        - **instruction set**: a list of all the commands that can be processed by a CPU and the commands are **machine code**
    - actions that are required for the instruction are carried out
    - Overall,
        1. storing data and addresses into **specific registers**
        2. using **busses** to transmit data, addresses and signals
        3. using **units** to fetch, decode and execute data and instructions
- *What will influence the <mark>performance of the CPU</mark>?*
    1. **Number of cores**
        - **Core**: the part of the CPU that contains all the components for the fetch-decode-execute cycle
        - $Core\ Number \propto Performance$
            - but **dual core** (2 cores) $\neq$ double performance
    2. **Size of the cache**
        - **Cache**: stores the most frequently used data and instructions, which will be fetched by the CPU instead of RAM
        - $Cache\ Size \propto Performance$
            - but **too large** cache would make the processing time **too long**
    3. **Speed of the clock**
        - **Clock speed**: the number of **FDE** cycles that can be performed **per second**
        - $Clock\ Speed \propto Performance$
- *What is an <mark>embedded system</mark>?*
    - an embedded system is used **to perform a dedicated function**, such as **domestic appliances**, **cars**, **security systems**, **lighting systems** or **vending machines**
    - it is different to a **general purpose computer** that is used to perform different functions

Input and Output

- *What are the ==input devices==?*
  - Barcode scanner, Digital camera, Keyboard, Microphone, Optical mouse, QR code scanner, Touch screen (resistive, capacitive, and infra-red), 2D and 3D scanners
- *What are the ==output devices==?*
  - **Actuator**, DLP and LCD projectors, inkjet and laser printers, LED and LCD screens, Speaker, 3D printer
- *What are the ==sensors==?*
  - a type of **input device**, captures **analogue data**
  - Acoustic, Accelerometer, Flow, Gas, Humidity, Infra-red, Level, Light, Magnetic field, Moisutre, pH, Pressure, Proximity, Temperatur

Data Storage

- *What is meant by ==primary storage==?*
  - **directly accessed by the CPU**
  - **Random Accessed Memory** -> stores the applications and files currently being used
  - **Read Only Memory** -> stores the instructions and data used for boot the computer
- *What is meant by ==secondary storage==?*
  - **not directly accessed by the CPU**
  - **necessary for more permanent storage**
- *What are the operations of secondary storage?*
  - **Magnetic** (HDD): uses **platters** which are divided into **tracks** and **sectors**; data read using **electromagnets**
  - **Optical** (CD): uses **pits** and **lands** which are created and read by **lasers**
  - **Solid-state** (SSD): uses **NAND** or **NOR** technology; **transistors** are used as **control gates** and **floating gates**
- *What is meant by ==virtual memory==?*
  - When the RAM gets full, the computer system can transfer pages of data to the virtual memory until it is needed
  - **Necessary:** frees up RAM by swapping data that has not been used recently over to a storage device
    - important for improving system performance, multitasking and using large programs.
- *What is meant by ==cloud storage==?*
  - cloud storage can be accessed **remotely** in comparison to storing data locally
  - **physical servers** and **storage** are needed to store data in cloud storage
  - **Adv**: don't pay for the hardware, can be accessed anywhere using any device, can increase storage space easily

- **Disadv**: pay for the provider, need an internet, pay you at risk if the provider is risked

Network Hardware

- *What is an <mark>Network interface card</mark>?*
  - a computer needs a **NIC** to access a network
  - it is given a **MAC address** at the point of **manufacture**
    - **MAC**: written in hexadecimal, splitted into **manufacture code** and the **serial code**
- *What is meant by an <mark>internet protocol (IP)</mark>?*
  - an IP address is **allocated by network**, can be **static or dynamic**
  - there is **IPv4** and **IPv6**
    - **IPv4**: 32-bit, denary
    - **IPv6**: 128-bit, hexadecimal
- *What is the role of a <mark>router</mark>?*
  1. **sends data to a specific destination** in the network
  2. **assigns IP addresses**
  3. **connects a local network to the internet**

4. Software

Types of Software and Interrupts

- *What are the <mark>types of software</mark>?*
  - **System software**: provides the **services that the computer requires**, including **operating system** and **utility software**
  - **Application software**: provides the **services that the user requires**
- *What are the <mark>functions of an OS</mark>?*
  - Managing files, Managing memory, Managing multitasking, Managing user accounts, Managing peripherals and drivers, Providing an interface, Providing a platform for running applications, Providing system security, Handling interrupts
- *What are required to run applications software?*
  - Applications are run on the operating system
  - The operating system is run on the firmware
  - The firmware is run on the hardware
  - Hardware -> Firmware -> OS -> Applications
- *What are the roles and operation of <mark>interrupts</mark>?*
  - **Software interrupts**: division by zero, two processes trying to access the same memory location
  - **Hardware interrupts**: pressing a key on the keyboard, moving the mouse
  - **Results of interrupts**:

- if there is an interrupt, the current process in the CPU is stored and the interrupt is loaded
- the CPU checks the source of the interrupt and calls the Interrupt Service Routine
  - **Interrupt Service Routine**:
    - ISR examines an interrupt and determines how to handle it, executes the handling, and then returns a logical interrupt value

Types of Programming Language, Translators and IDEs

- *What are the types of programming languages?*
  - **High-level language**: ease of reading and writing, ease of debugging, **machine independence**
  - **Low-level language**: direct manipulation of hardware, fast to run

- *What is assembly language?*
  - a form of **low-level language** that uses **mnemonics**
  - an **assembler** is needed to translate assembly language into machine code

- *What are the translators?*
  - **Compiler**: translates the whole code at once, producing an **executable file**; provides **an error report** for the whole code and every error at once
    - used in **production** to translate the whole program
  - **Interpreter**: translates and executes the code **line-by-line**; **stops execution** when an error is found
    - used in **development**

- *What are the common features of an Integrated Development Environment?*
  - **code editors**: auto-completion, auto-correction, prettyprint, error diagnostic
  - **run-time environment**
  - **translators**

5. The internet and its uses

The Internet and WWW

- *What are the differences between the internet and the WWW?*
  - **Internet**: the **infrastructure**
  - **World Wide Web**: the **collection** of **website and webpages** accessed using the internet

- *What is meant by a uniform resource locator (URL)?*
  - a **text-based** address for a web page
  - contains the **protocol, the domain name, and the file name**

- *What are HTTP and HTTPS?*
  - **Hypertext Transfer Protocol**:
    - a **protocol** that is used for the transmission of web pages and related data across the internet

- **Hypertext Transer Protocol Secure**:
  - Uses **Secure Socket Layer** to make the data transmission secured
    - 有一个证书机构(certificate authority)给网站发证书，浏览器在和网站的服务器建立联系之前会先要求服务器发送网站的证书，浏览器检查证书是正确的才会建立连接传输数据
  - Data is **encrypted** in the transmission
- *What are <mark>web browsers</mark>?*
  - **Purpose**: to **render HTML** and **display web pages**
  - **Functions**: storing bookmarks and favourites, recording user history, allowing use of multiple tabes, storing cookies, providing navigation tools, providing an address bar
- *How webpages are retrieved?*
  1. User types in the **URL**
  2. Browser sends the URL to the **Domain name server** (DNS)
  3. DNS looks into its database to find the corresponding **IP address** of the webserver that stores the website and sends it back to the **Browser**
  4. Browser sends a request to the **web server** through the IP address to ask for the web page (using HTTP or HTTPS)
  5. Web server sends the **HTML** content of the webpage to the **Browser**
  6. Browser renders the HTML and displays it to the user
- *What is meant by cookies?*
  - **small text file** that is used to store **personal data**, by the **web browser**
    - **Senssion Cookies**: **deleted** when the browser is closed
    - **Persistent Cookies**: stored by the web browser **until it is deleted by the user** or because **it is expired**
  - **Functions**: saving personal details, tracking user preferences, holding items in an online shopping cart, storing login details

Digital Currency

- *What is a <mark>digital currency</mark>?*
  - one that **only exists electronically**
  - **used** in credits cards and digital payments (AliPay)
- *What is a <mark>blockchain</mark>?*
  - a **digital ledger**, a **time-stamped series of records** that **cannot be altered**
  - **process**:
    1. a transaction is initiated
    2. this action creates a **block** that represents that specific transaction or data
    3. the **block** is sent to **every computer node** in the network
    4. **authorized nodes** verify the transaction and add the block to the existing **blockchain**

5. the verified block is **time-stamped** with **cryptographic hash** and has a reference to the previous block's hash

6. the unit of value moves from the sender's account to the receiver's account. The update is distributed across the network, which finalizes the transaction
   - once a transaction is completed, it <mark>cannot</mark> be **deleted** or **altered**

Cybersecurity

- *What are the <mark>cybersecurity threats</mark>?*
  - 要知道每个threat的过程和目的，感觉CIE很喜欢靠这玩意儿
  - **Brute-force attack**: 暴力试密码
  - **Data interception**: 截取传输在网络中的数据包 (packet sniffing)
  - **Distributed denial of service (DDoS)**: 抓一个肉鸡网络(botnet)不停的访问服务器让其崩溃
  - **Hacking**: 攻击系统漏洞获得控制权限
  - **Malware**: 各种奇奇怪怪的有害软件
    - virus, worm, Trojan horse, spyware, adware, ransomware
  - **Pharming**: 给你装一个malware让你访问正常网站的时候redirect到钓鱼网站
  - **Phishing**: 钓鱼邮件让你打开钓鱼网站
  - **Social enginerring**: 通过非技术手段获取你的信息 譬如电信诈骗钓鱼邮件
- *What are the solutions?*
  - **Access levels**: 不同的用户不同权限 被黑的要是权限低黑客也拿不到信息
  - **Anti-malware (anti-virus and anti-spyware)**: 检测你的系统有没有malware
  - **Authentication**: 让你没法轻易猜到密码
    - username and password, biometrics, two-step verification
  - **Automating software updates**: 升级让你的系统没有漏洞可黑
  - **Checking the spelling and tone of communications**: 看看钓鱼邮件/钓鱼网站的语气对不对
  - **Checking the URL attached to a link**: 看看是不是钓鱼网站 网址是不是官网的
  - **Firewalls**: 设定防火墙规则组织那些不友好的IP或者软件
  - **Privacy Settings**: 避免信息公开容易被获取
  - **Proxy-servers**: 控制对Webserver的请求数量 防止DDoS
  - **SSL**: 看看你访问的网站是否有证书 对传输的数据进行加密免得给人截了

6. Automated and emerging technologies

Automated systems

- *How to use <mark>sensors, microprocessors and actuators</mark> to create automated systems?*
  1. the sensor **constantly** reads the data from the environment
  2. the **analogue data from the sensor** is sent to the **ADC** (describe ADC's function here as well)

3. the **digital data** is sent to the **microprocessor**
4. the microprocessor **compares** the incoming data with the stored or **pre-set value**
5. if the data is within the range or matches the stored value then, no action is taken, and the process continues
6. if the data is outside the range, then the microprocessor will send a signal to the **actuator** via a **DAC**
7. the whole process **continues** and **loops**
   - 答题时候要contexualize

- *How to evaluate automated systems*?
  - **Adv**: less running cost (not to pay wages), continues working 24/7, precision
  - **Disad**: replace jobs (Economics^_^), high initial costs

Robotics

- *What is meant by <mark>robotics</mark>?*
  - a **branch of computer science** that incorporates the **design, construction and operation** of robots
  - Examples include **factory equipment, domestic robots and drones**
- *What are the <mark>characteristics of robots</mark>*
  - a **mechanical structure or framework**
  - **electrical components**: sensors, microprocessors and actuators
  - **programmable**
- *What are the roles of robots*?
  - in industry, transport, agriculture, medicine, domestic, and entertainment

Artificial Intelligence

- *What is meant by <mark>AI</mark>?*
  - a **branch of computer science** dealing with **the simulation of intelligent behaviours by computers**
- *What are the main characteristics of AI?*
  1. **the collection of data**
  2. **a set of programmed rules**
  3. **the ability to reason**
  4. **the ability to learn and adapt** (for machine learning only)
- *What are the types of AI*?
  - **Expert systems**: have a knowledge base, a rule base, an inference engine and an interface
  - **Machine learning**: a program has the ability to automatically adapt its own process and/or data

- **supervised learning**: when the program's dataset has both **input** and **target**, and the program is just finding relationships between the inputs and their related targets

- **unsupervised learning**: when the program's dataset has only **inputs**, and the program finds pattern in these inputs