Queen Mary
University of London

ECS404U: COMPUTER SYSTEMS & NETWORKS

2020/2021– Semester 1

Prof. Edmund Robinson, Dr. Arman Khouzani

Lab 2: Introduction to Computers, Basics of Binary Representation

September 28/30, 2020

Student Name:

Yujeong JUNG

Student ID:

200699145

Brief Feedback to student (commendations, areas for improvement):

Instructions: This lab has two sections: (1) “Introduction to Computer Architecture”, and (2) “Introduction to Digital Representation”.
For your convenience, the pdf file of the lab manual now contains fields into which you can directly enter your answers. Once finished, you need to save your filled out worksheet as a pdf file, and then upload it through the link on the QM+ module page (available under Week 2). Recall that even though the deadline wil be 10:00 AM of the following Thursday, but you are expected to finish the tasks within the 2 hours of your lab session.

Note: Right after filling out your name and ID fields above, experiment with saving your pdf file (one of the methods of “save” or “save as” a .pdf file, “export” as a pdf file, “print” to a pdf file, etc.) Then close this document and open the saved pdf file to asertain that your inputs are indeed saved. If there is any problem in saving your answers, it is better to catch and resolve it now!

At submission, make sure you haven’t just saved it as a draft, but finalised it all the ay through (you should receive an automatic email confirmation). Please do ask for help throughout the lab whenever you need.

1 Introduction to Computer Architecture

Learning Objectives:

- In the first lecture, we discussed that computers contain certain standard components, notably **CPU (Central Processing Unit)**, **memory**, and **long-term storage**. The main objective of this task is to reinforce this idea by identifying these components in a number of devices;

Also:

- Demonstrate that the components (and detailed technologies used) are often similar but not identical across a range of device types;
- Become familiar with the appropriate ways that the performance of each of these components are measured and compared;
- Gain a rough idea about the typical performance metrics of these components across a range of some commonly used devices, and be able to roughly justify their choice.

Reading material Reading to support Section 1 of the lab: Course notes chapters 1 and 2.

1.1 Investigate the architecture of various computers

Outline: Find out which components are used in the device in question. Preferably, try to get the information directly from the device itself (if you have access to it). Otherwise, search for its specification on its web pages. Some example links are provided (but we trust you are good with online search engines!).

1.1.1 Standard computers (desktops, laptops, and servers)

We start by investigating the most obvious computers around us: personal computers (desktops, laptops) and servers. Here are some hints on where to get the information about the processor, memory and storage on different operating systems:

Windows: right click Start → Run → `msinfo32` → enter (more detail from this link)

MacOs: Apple → About This Mac → System Report

Linux: some specific system commands, but also given in files mainly in `/proc` (details to follow)

For Linux, we can use the student login server named grover, with the full name `login.student.eecs.qmul.ac.uk` or `grover.student.eecs.qmul.ac.uk`. Recall that you can (SSH) log in to grover server from anywhere. For instance, using a Linux terminal, you can do that by typing in:

```
ssh aa123@grover.student.eecs.qmul.ac.uk
```

Replace `aa123` with your eecs login, which has that format. To log out from a ssh remote session, you can use the commands `logout` or `exit`.
The following system commands, executed by entering them in a "terminal", can help you gather the required information in Linux:

`lscpu` : "Display information about the CPU architecture". The most relevant output lines for us are:

`Socket(s)`: gives the number of physical chips

`Core(s) per socket`: gives the number of cores on each physical chip

`Model name`: gives the actual name of the microprocessor and its clock speed

Note that the output line for `CPU(s)` takes into account the number of threads that can be simultaneously run on each core as well. So it may not be the total number of physical cores (but a multiple of it).

`free -h` : "Display amount of free and used memory in the system". The `-h` flag makes it give sizes in a more human readable form.

`lsblk` : lists block devices. You may see a tree-like hierarchy, with the main storage disk at the top of the tree (look for the one with the largest size!).

More information is available through commands that are only runnable as `root`. For example, `lshw`, extracts and lists information about all the hardware on the computer.
Finally, Linux keeps a lot of useful system information in specific files in the `/proc` filesystem. Some of the relevant ones are listed below (to see the content of a (text) file in a linux terminal, you can type in `more <filename>`)

- CPU information from `/proc/cpuinfo` (type `more /proc/cpuinfo`)
- memory from `/proc/meminfo` (type `more /proc/meminfo`)

Now, your task is to fill out the requested information in the following tables.

`grover.eecs.qmul.ac.uk`		
CPU	name	Intel (R) Xeon (R) CPU E5-2680 v4
	clock speed	2397.224 MHz
	# of cores	2
Main Memory (RAM)	size (GB)	1.8
Disk (long term storage)	size (GB)	2.0

Your laptop or PC:	MacBook Pro 2018	
CPU	name	Quad-Core Intel Core i5
	clock speed	2.3 GHz
	# of cores	4
Main Memory (RAM)	size (GB)	8
	type (DDR3, DDR4, ...)	LPDDR3
Disk (long term storage)	size (GB)	499.96
	type (HDD, SSD, ...)	Macintosh HD

1.1.2 Mobile phones and Tablets:

Less immediately obvious examples of computers are smart-phones and tablets. Here, we see how they share the same components with a general computer, though with different performance metrics.
Note that some iOS and Android systems may not make detailed information of their hardware easily accessible. If you find yourself in such a situation, look up the specification of your phone/tablet on the web. Search

for the combination of *make* and *model* of your device of choice plus the keyword "specifications" (or "specs", for short). Start with the manufacturer's own website as the most authoritative source. Manufacturers may not list specs of chips, but you may be able to find these via tear-downs or sites like ifixit.com.
Example android smartphone: `Samsung Galaxy Note10`, `HUAWEI Mate 20` or 30, ...
Example Apple smartphone: `iPhone X` or `11`

Smart phone/tablet (model):		iPhone Xs
CPU	name	Apple A12 Bionic
	clock speed	2490
	# of cores	6
Main Memory	size (GB)	64
Long term memory	size (GB)	4
	type	LPDDR4X

1.1.3 Consumer devices: Game Consoles, Home Routers/Hubs, ...

Another example among consumer devices that you can find a computer is a game console (`PlayStation`, `Xbox`, `Nintendo`, etc.). But as we mentioned in the class, home routers are among examples of computers too, as well as "smart" speakers that come with a virtual assistant (`Google Home`, `Amazon Echo`, `Apple HomePod`, etc.).
Technical information of the computing components of these devices is often hard to find on their manufacturers sites. Here are some links to help (don't worry if you can't find all the requested specifications – get as many as you can!)

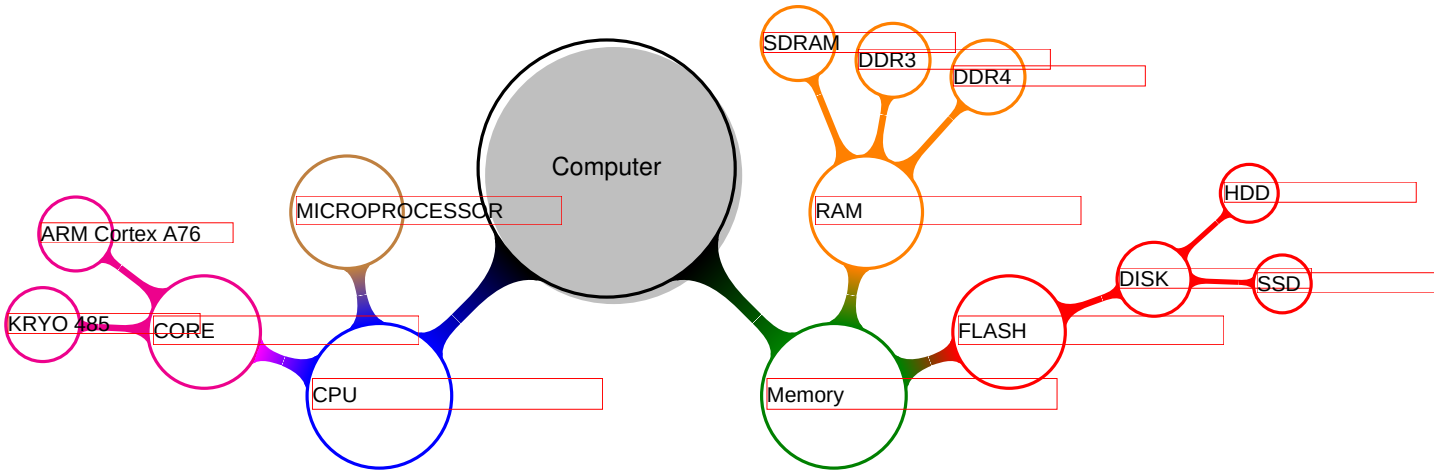
Playstation 4	`https://www.playstation.com/en-gb/explore/ps4/tech-specs/` `https://en.wikipedia.org/wiki/Jaguar_(microarchitecture)`
BT Smart Hub	`https://wikidevi.com/wiki/BT_Smart_Hub_Type_A` `https://en.wikipedia.org/wiki/BT_Smart_Hub`

`Playstation 4`		
CPU	name	x86-64 AMD "Jaguar"
	clock speed	1.6GHz
	# of cores	8
Main Memory	size (GB)	500GB, 1TB
Long term memory	size (GB)	8
	type	GDDR5

`BT Home Hub`		
CPU	name	Broadcom BCM63137
	clock speed	1GHz
	# of cores	2
Main Memory	size (GB)	256MB
Long term memory	size (GB)	256MB
	type	DDR3

1.2 Basic computer architecture: relationship diagram

In the previous exercise, we saw a lot of terms and abbreviations. Let's organise them in relation to each other to maintain a big-picture view. In particular, let's create a "mind-map".
A mind-map is a graph (a diagram) that visualises information by losely showing the hirearchy and relationship between different concepts, terms, objects, etc. The idea is simple: we arrange conceptually closer items next to each other, and add links based on relationships such as "is an example of" or "is a type of" or "contains", etc.[1] So let's draw a mind-map for the following concepts (use an online search engine if you are not sure of an item):
`CPU, MICROPROCESSOR, Computer, CORE, RAM, SDRAM, DDR3, DDR4, ARM Cortex A76, HDD, SSD, DISK, Memory, FLASH, KRYO 485`
The skeleton of the mindmap is provided for you. The nodes are clickable (a drop-down menu should apear to select the item for that node).



1.3 Some technical questions

1.3.1 CPU Clock Speed (Clock Rate)

As we noticed in the first exercise, one of the attributes of a CPU microprocessor is its clock speed.

1. What is its unit of measurement? Hertz (Hz)
2. What does the clock rate of a CPU signify? (What does it really mean?) Briefly describe.

   The clock speed means how fast CPU can run. The speed has limits but it is possible to control in the computer BIOS. A faster CPU uses more energy and creates more heat.

[1] The relationships in a mindmap is implicitly inferred, and is not as rigorous. A more rigorous visualisation approach is using "concept maps" where the type of the relation should also be specified (e.g., "is an example of", "contains", "contributes to", "causes", "requires", and so on. You will learn more about these diagrams in your "Information Systems" module in Sem 2, and later on during your "Software Engineering", "Graphical User Interfaces", and "Database Systems" modules in year 2.

Optional: To get an idea of how amazingly fast modern processors has become, compute the distance that light travels (in vacuum) during one clock cycle of the CPU processor of `grover`'s. You can take the speed of light as roughly $3 \times 10^8 m/s$ (meters per second).

Solution: Light travels [ ] meters in one of `grover`'s clock cycles.

Also Optional: The duration of an eye "blink" in humans is about 0.1 second. How many clock cycles does the grover's CPU go through during a blink of an eye?

Solution: `grover`'s CPU goes through [ ] many clock cycles during an eye-blink.

1.3.2 CPU or Microprocessor? Technically speaking.

There is a distinction between the concepts of *function* and *implementation* (as you will revisit in your other CS modules): A functional description is an (abstract) description of what something does, that is, the specific behaviour of the component in terms of its "outputs" based on "inputs". An implementational description, on the other hand, is about "how" something is actually built (in order to carry out that functional description). Given this high-level idea, make a delineation between a "CPU" and a "microprocessor" (determine which one of the terms 'function' vs 'implementation' applies more closely to which, thereby making clear both the connection and the distinction between a CPU and a microprocessor.)

The CPU is equivalent to the brain of the computer, which interprets and computes commands entered by the user and outputs the results.
The most basic role of computer CPU is computation. For example, if a user issues the command "1+3", the CPU accepts it and calculates it, and shows the result of "4" as an image output device (monitor, etc).
"Microporcessor" refers to a single silicon chip containing computer math logic calculator, register, program counter, command decoder, control circuit, etc. In other word, it functions to interpret and execute commands stored in the main memory.
Microprocessor is widely used in various fields, including military weapons and home

Given the above discussion, it is technically inaccurate to say, e.g., my laptop's CPU is `Intel(R) Core(TM) i7-6600U`. Suggest what should be said instead to be more technically accurate.

The CPU of my laptop is Intel Core i5. Simply put, there are five slaves who will do the work when you input the command on your laptop.

1.3.3 "Cores"

Modern CPU's feature several cores. What is a core?

"Core" is a heart operating device inside a processor that corresponds to the brain of a smartphone or computer.
The core of the network is the central backbone, and the core of the microprocessor chip is the main processing circuit.

1.3.4 Same device, different types of cores? Why?

Looking at some of the chips in smartphones, you will notice that their central processor often has **two** or even three different *types* of cores (and potentially, multiple cores from each type). For instance, Qualcomm's `Snapdragon 835` system-on-ship, used e.g. in *Samsung Galaxy S8*, has 8 cores, 4 of which are `Kryo 280 Gold` (based on `ARM Cortex-A73`) with clock rate of about 2.45 GHz each, and the other 4 are `Kryo 280 Silver` (based on `ARM Cortex-A53`) with clock rate of about 1.9 GHz each.
Do some research and find out why this is the case. In particular, explain why we don't just use the type with a higher clock rate for all of the 8 cores?
You may find the following web pages useful, but you should express your explanation in your own words:

- `https://developer.arm.com/technologies/big-little`
- `https://en.wikipedia.org/wiki/ARM_big.LITTLE`
- `https://en.wikipedia.org/wiki/Comparison_of_ARMv8-A_cores`

If the clock speed is too high, the heat and power consumption will increase. This is especially risk for portable devices.
Multi-core CPUs have similar performance to having a single-core CPU installed at the same time as the number of embedded cores.
However, multi-core CPUs can have a great effect only when they perform various tasks at the same time or run software optimized for multi-core operations.

2 Digital Representation: Basic Binary Numbers

This is a first exercise in the use of binary representation, concentrating on translating between binary and decimal. **Learning Objectives:**

- Ensure that you are familiar with the basic use of binary numbers;
- Ensure you can convert between unsigned binary (base 2) to and from decimal (base 10).

If you are not familiar with these, see:

- Watch the pre-recorded video of week 1 titled "Digital Representation 1.1: Binary" (about 13 minutes).
- Course-notes, chapters 9 and 10 (if you are in a hurry, at least read section 10.3 of the Course Notes, which is just 2 pages long!);
- Watch the video clip "Representing Numbers and Letters with Binary: Crash Course Computer Science #4" on YouTube (about 10 minutes).
- Ask from any of the moderators (but please after going through the worked-out example first)!

Acceptable answers must include the detailed working.

2.1 Powers of 2

Write out (and *memorise!*) the powers of 2 from 2^0 to 2^{10}:

n	0	1	2	3	4	5	6	7	8	9	10
2^n	1	2	4	8	16	32	64	128	256	512	1024

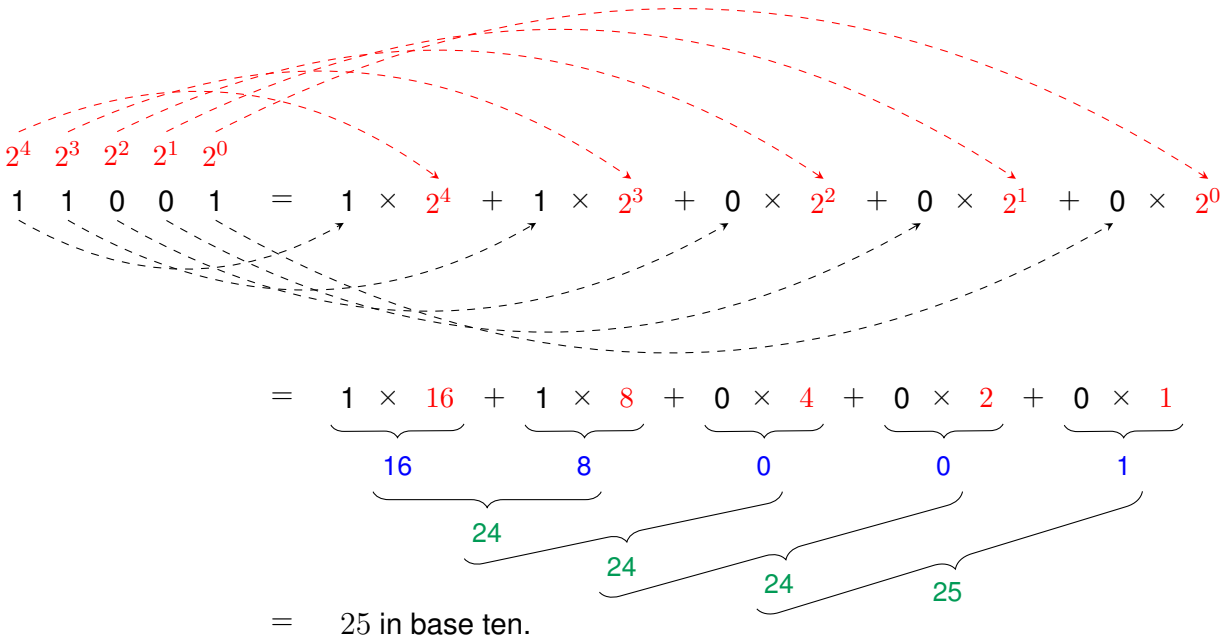
2.2 Convert from binary to decimal

Example: 11001

Using $**$ for the exponential operator as in Python (Java uses `math.pow`):

$$\begin{aligned} 11001_2 &= 1*2**4+1*2**3+0*2**2+0*2**1+1*2**0 \\ &= 16+8+1 \\ &= 25_{10} \end{aligned}$$

Note that we can show the base either as a superscript, like 11001_2, or explicitly say it, like 11001 in base 2. In more detail (note that the picture is just for your learning purposes, you are not expected to provide it. Only the three lines as above suffices. However, this describes the steps you would be taking in your mind to produce those three lines, so make sure you understand it!):



Repeat the working above for the following numbers. For the first line express as a sum of powers of 2 (e.g. $1*2**4+1*2**3+0*2**2+0*2**1+1*2**0$ for our worked out example), for the second as the sum of the components (as e.g. $16+8+1$), and for the third the result (as e.g. 25).

(a) 101

101_2

$=$ 1*2**2+0*2**1+1*2**0

$=$ 1*4+0*2+1*1

$=$ 5 in base ten

(b) 1011

1011_2

$=$ 1*2**3+0*2**2+1*2**1+1*2**0

$=$ 1*8+0*4+1*2+1*1

$=$ 11 in base ten

(c) $1010\ 0111$

$1010\ 0111_2$
= 1*2**7+0*2**6+1*2**5+0*2**4+0*2**3+1*2**2+1*2**1+1*
= 1*128+0*64+1*32+0*16+0*8+1*4+1*2+1*1
= 167 in base ten

(d) $1001\ 1010\ 0010$

$1001\ 1010\ 0010_2$
= 1*2**11+0*2**10+0*2**9+1*2**8+1*2**7+0*2**6+1*2**5+
= 1*2048+0*1024+0*512+1*256+1*128+0*64+1*32+0*16+
= 2466 in base ten

2.3 Convert from decimal to binary

Example: 106

$$\begin{aligned} 106_{10} &= 64 + 42 \\ &= 64 + 32 + 10 \\ &= 64 + 32 + 8 + 2 \\ &= 2**6 + 2**5 + 2**3 + 2**1 \\ &= 110\ 1010_2 \end{aligned}$$

106 = **0**×128 + **1**×64 + **1**×32 + **0**×16 + **1**×8 + **0**×4 + **1**×2 + **0**×1
Here is a more detailed explanation (again, this picture is only for your learning, and just describes how you should think and go about this problem in your mind, so make sure you understand it!):

$$\begin{aligned} 106 &= ?\times 2^7 + ?\times 2^6 + ?\times 2^5 + ?\times 2^4 + ?\times 2^3 + ?\times 2^2 + ?\times 2^1 + ?\times 2^0 \\ &= ?\times 128 + ?\times 64 + ?\times 32 + ?\times 16 + ?\times 8 + ?\times 4 + ?\times 2 + ?\times 1 \\ &= 0\times 128 + 1\times 64 + \underbrace{?\times 32 + ?\times 16 + ?\times 8 + ?\times 4 + ?\times 2 + ?\times 1}_{42} \\ &= 0\times 128 + 1\times 64 + 1\times 32 + \underbrace{?\times 16 + ?\times 8 + ?\times 4 + ?\times 2 + ?\times 1}_{10} \\ &= 0\times 128 + 1\times 64 + 1\times 32 + 0\times 16 + 1\times 8 + \underbrace{?\times 4 + ?\times 2 + ?\times 1}_{2} \\ &= 0\times 128 + 1\times 64 + 1\times 32 + 0\times 16 + 1\times 8 + 0\times 4 + 1\times 2 + \underbrace{?\times 1}_{0} \\ &= 0\times 128 + 1\times 64 + 1\times 32 + 0\times 16 + 1\times 8 + 0\times 4 + 1\times 2 + 0\times 1 \\ &= (01101010)_2 \end{aligned}$$

Repeat this for other numbers. First line should be the analogue of $64 + 32 + 8 + 2$, second the analogue of $2**6 + 2**5 + 2**3 + 2**1$ and the final the binary.

(a) 87

87_{10}
$=$?*2**7+?*2**6+?*2**5+?*2**4+?*2**3+?*2**2+?*2**1+?*
$=$ 0*128+1*64+0*32+1*16+0*8+1*4+1*2+1*1
$=$ 0101 0111 in binary

(b) 153

153_{10}
$=$?*2**7+?*2**6+?*2**5+?*2**4+?*2**3+?*2**2+?*2**1+?*
$=$ 1*128+0*64+0*32+1*16+1*8+0*4+0*2+1*1
$=$ 1001 1001 in binary

(c) (**optional**) 1250

1250_{10}
$=$?*2**11+?*2**10+?*2*9+?*2**8+?*2**7+?*2**6+?*2**5+?
$=$ 0*2048+1*1024+0*512+0*256+1*128+1*64+1*32+0*16+
$=$ 010011100010 in binary

(d) 3274

3274_{10}
$=$?*2**11+?*2**10+?*2*9+?*2**8+?*2**7+?*2**6+?*2**5+?
$=$ 1*2048+1*1024+0*512+0*256+1*128+1*64+0*32+0*16+
$=$ 110011001010 in binary

2.3.1 An example: IPv4 addresses

A machine's IPv4 address consists of 4 numbers, each between 0 and 255 (by convention, this is in decimal). Each number represents an 8 bit sequence. An IPv4 address is hence 32 bits long. Give the bit sequences of the following IP addresses:

(a) 40.85.117.194 (qmplus.qmul.ac.uk)

101000 . 1010101 . 1110101 . 11000010

(b) 138.37.37.237 (grover.eecs.qmul.ac.uk)

10001010 . 100101 . 100101 . 11101101

End of questions