Transistor & CPU

Date: 15 – 19 February 2021 **Week:** 2

Task I: Tutorial questions

1. Explain Moore's law. Is Moore's law a Physics law?
2. Modern CPUs are measured in nano-metre scale (e.g. 14 nm CPU). What does this number mean? Is this the size of each individual transistor?
3. Describe the quantum tunneling problem in the transistor design.

Task II: Practical exercises

Performing the following Linux commands, using one of the environments we set up in Week 1.

```
$ date
```

Will tell the current date. The output may be captured to a file using re-direction. For example

```
$ date > today
```

will redirect the result from running the program 'date' to the file called today. You can check on this by typing the contents of today with the 'cat' (short for concatenate) program.

```
$ cat today
```

This feature is particularly useful as programs do not need to be written in a special way to take advantage of this. In fact

```
$ date >> today
```

will append the output from 'date' onto the end of the file today. Check that this has happened with the 'cat' program.

```
$ cat today
```

Commands may be given arguments as with the *cal* program which prints a calendar for a particular year or month of year as with

```
$ cal 1 2010
```

To print the calendar for January 2010. Try printing the calendar for September 1752. Notice anything odd. Can you explain why this is?

The program 'who' will list who is on the system. Again, this may be captured on to a file using re-direction. On a PC, you could be the only user. However, you may have several logins on this machine. As machines are networked you may find other people have logged in to the machine you are on. You may, at this point, think about security.

To find out what files you have in the current directory you can use the command ls.

```
$ ls
```

or to produce a longer list with the creation dates etc. of these files use the flag '-l'

```
$ ls -l
```

The output may be re-directed into a file

```
$ ls -l > files
```

Try this out to verify that the program works in the way that you expect it to do. Do this again and replace ‘files’ with a file name you like.

One of the more powerful features of Unix/Linux is pattern matching in file names. This pattern matching or use of wild card can simplify many tasks.

```
$ cat *
```

will list the contents of all your files/directories

```
$ cat p*
```

would list only those files/directories which start with p

A subdirectory is created with the mkdir command. For example, to create the subdirectory mysub the following command is used:

```
$ mkdir mysub
```

To see that the directory has been created, just type

```
$ ls
```

To go into this subdirectory the command `cd mysub` is used. Without any parameter, cd will change to your root home directory - a useful facility if you get lost navigating through many directories.

To change to the directory immediately above the current directory the following form of the cd command is used:

```
$ cd ..
```

The command

```
$ pwd
```

prints out the position of the current directory. The current directory is represented with . (dot). If you execute

```
$ cd .
```

you should stay in the same directory. Check this with the pwd command

The symbol ~ (tilde) also represent your home directory. Executing

```
$ cd ~
```

will take you directly to your home directory.

```
$ ls ~/mysub
```

will list the content of your sub directory from anywhere where you are in the directory structure.

What do you think this the following commands will produce?

```
$ ls ~
```

```
$ ls ~/..
```

Find out how many hidden files are in your home directory.

The following command will copy file ‘today’ into file ‘tomorrow’ in the same directory.

```
$ cp today tomorrow
```

You will copy this file to your mysub directory. First change to my sub directory:

```
$ cd ~/mysub
```

Now copy files to your current directory:

```
$ cp ~/tomorrow .
```

Don’t forget . (dot) as it represents the current directory.

List the content of the directory to check that the file has been copied into.

```
$ ls
```

Now create a new directory called *“backups”* inside the current directory, and copy your ‘tomorrow’ file to that directory. List the backups directory.

To rename or move files you use command mv. To rename ‘tomorrow’ into ‘files.txt’, type the following command, inside the “backups” directory created earlier.

```
$ mv tomorrow files.txt
```

Using the *rm* command, delete the file ‘*files.txt’* from the backups directory.

```
$ rm files.txt
```

Document your answers in the learning journal, including screenshots of the Linux commands