Exercise Sheet 3

Exercise 1 (Computer Architecture)

1. Which are the two esential components of any CPU?

2. Name two other (optional) components a CPU may possess.

3. Which three digital bus systems contains each computer system according to the Von Neumann architecture?

4. Which tasks are carried out by the three digital bus systems of subtask 3?

5. What is the Front Side Bus (FSB)?

6. Which two components contains the chipset?

7. Name the tasks of the components of the chipset.

Exercise 2 (Input/Output Devices)

1. What is the fundamental distinction between character and block I/O devices?

2. Name two examples for character and block devices.

3. Name three possible ways for processes to read data from I/O devices.
   -
   -
   -

4. Name a benefit and a drawback for each possible way from subtask 3.
   -
     –
     –

-
  -
  -
-
  -
  -

Exercise 3 (Digital Data Storage)

1. Name one mechanic digital data storage.

2. Name two rotating magnetic digital data storages.

3. Name four benefits of data storage without moving parts compared with data storage with moving parts.

4. What is random access?

5. Name one non-persistent data storage.

6. The storage of computer systems is distinguished into the categories primary storage, secondary storage and tertiary storage. Which category or categories can the CPU access directly?

7. Which category or categories of subtask 6 can the CPU only access via a controller?

Exercise 4 (Write policies)

1. Name the two basic cache write policies.

2. With which cache write policy of subtask 1 may inconsistencies occur?

3. With which cache write policy of subtask 1 is the system performance lower?

4. With which cache write policy of subtask 1 are so called dirty bits used?

5. For what reason are dirty bits used?

Exercise 5 (Process States)

Implement a program that create new processes and turn them into zombies.

Check the information about the processes in the *proc* filesystem ($\longrightarrow$ `man 5 proc`).

Exercise 6 (Forking Processes)

In this programming exercise you have to work with processes including forking, executing other programs, and waiting for child processes.

In this assignment, you will write a little application to launch another program and measure its CPU runtime.

Basic functionality when user executes your program **mytime**:

1. If no command line arguments are given, the program should print information on how to call the program correctly.
2. When executed with at least one command line argument the first argument should be interpreted as a program name. Your program should then execute this program – as a separate process – and pass all remaining command line arguments as arguments to the executed program.
3. For any executed program its return value and the CPU time of the process in milliseconds should be printed.
4. You must create a *Makefile* such that when someone types `make` in your working directory it will compile the program with an output of `mytime`.

You will need the system commands `fork`, a version of `exec`, `waitpid`, and `clock_gettime` to complete this task. For details on how to use these, you can use UNIX's man pages. There is also an online version at `https://www.kernel.org/doc/man-pages/`.