Exercise sheet 5 – Processor architecture

Goals:

- Synchronisation commands
- Endianness

Exercise 5.1: Endianness (theoretical)

(a) The given struct `meier` is transmitted serially (byte-by-byte) from a *little-endian* to a *big-endian* architecture. Assume both are 32-bit architectures.

```
1  struct employee {
2      char name[12];      //ASCII/UTF-8 (1 byte)
3      char birthday[12]; //ASCII/UTF-8 (1 byte)
4      int32_t id;
5  };
6
7  struct employee meier = {
8      .name     = "Josef Meier",
9      .birthday = "01.02.1957",
10     .id       = 0x123456
11 };
```

Provide a solution similar to the scheme in the lecture. Which corrections may be necessary?

Exercise 5.2: Endianness with integer (coding)

Given is a *big-endian* system program—the Java runtime environment—that transfers data via a file to a *little-endian* system C program. Investigate the behaviour: You may have to fix something for a correct transfer.

(a) Update the `CA_exercises` repository with `git pull`.

(b) Change into the directory `CA_exercises/sheet_05_endianness/Endianness/Java_BE`

(c) Inspect, build, and run the given Java program.

(d) Inspect and interpret the result file `output.txt`. *Hint: Use a HEX viewer, for example:* `xxd`.

```
1  xxd -c 1 output.txt #to show the byte per byte view (memory/file view) \\
2  xxd -c 4 output.txt #to show the word view (register view in 32 bit/4 byte mode)
```

(e) Change into the directory `CA_exercises/sheet_05_endianness/Endianness/C_LE`

(f) Inspect, build, and run the given C program.

(g) Analyse the output of the C program. What happened? What could be the cause of this?

(h) Fix the problem in the C program, following the *TODOs*.

(i) Build and run the C program again. Is the problem now solved?

Exercise 5.3: Endianness with an UTF16 character (coding)

Given is a *big-endian* UTF16 encoded character saved in a file. A C/C++ program (*little-endian*) wants to read and print the encoded character to the terminal.

(a) Change into the directory
`CA_exercises/sheet_05_endianness/Endianness_UTF16/read_UTF16_character`

(b) Inspect the UTF16 character within the given file
`CA_exercises/sheet_05_endianness/Endianness_UTF16/utf16_character_be.txt`.
Hint: Use a HEX viewer, for example: `xxd`.

```
xxd -c 1 utf16_character_be.txt #to show the byte per byte view (memory/file view) \\
xxd -c 4 utf16_character_be.txt #to show the word view (register view in 32 bit/4 byte
```

(c) Inspect, build, and run the given C/C++ program.

(d) Analyse the output of the C/C++ program. What happened? What could be the cause of this?

(e) Fix the problem in the C/C++ program, following the *TODOs*.

(f) Build and run the C program again. Is the problem now solved?