ECE xxxx: Applied Awesomeness

Lab 1: Some stuff

May 4, 2025

Jonathan Anderson
200125805
jonathan.anderson@mun.ca

Andy Anderson
201012345
andy.anderson@mun.ca

1. Pre-lab preparation

1.1. Meaning of `xxd -r`

> **Info**
>
> In your pre-lab preparation, I will expect you to answer questions, with reference to appropriate sources, etc. You will include these things here.

According to the `xxd(1)` manual page [1], the `-r` flag is used to reverse the hex dump operation. For example, given a file `hello.dump` with the contents:

```
00000000: 4865 6c6c 6f20 776f 726c 640a            Hello world.
```

the command `xxd -r hello.dump` will output:

```
Hello world
```

1.2. Simple I/O program

The `main` function for a program that will output our names is:

> **Code**
>
> Here's an example of how you can include code with nice syntax highlighting and **no screenshots**. Please avoid screenshots unless they're absolutely necessary.

```
1  #include <stdio.h>
2
3  int main(int argc, char *argv[])
4  {
5          printf("Jon and Andy\n");
6          return 0;
7  }
```

Listing 1: An example of a source code snippet

> **Tip**
>
> You can also include portions of source code files like this. Please don't include enormous files of source code inline within a report: include snippets here, and if you need the whole thing somewhere, you can include it as an appendix at the end of the report (and reference it from here).

The source code we wrote for this part of the lab is shown in Listing 2. The full source code listing is shown in Section 3.

```
6  double subtract(double x, double y)
7  {
8    return x - y;
9  }
```

Listing 2: Another source code snippet, showing just lines 6–9 of a file

1.3. Circuit design

The required digital circuit is shown in Figure 1.

> **Idea**
>
> Most figures that you produce will be in the form of images (PNG, JPEG, etc.). The example below uses a package for drawing circuits right in Typst, which is cool and fun, but not required.

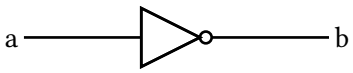

Figure 1: A digital circuit

2. Setup

2.1. Compilation

We assembled our code using the command:

```
riscv32-esp-elf-gcc -c app_main.s
```

This created the file `app_main.o`.

2.2. Disassembly

We disassembled the `app_main.o` object file using the command:

```
riscv32-esp-elf-objdump -d app_main.o
```

which output the following:

> **Tip**
>
> The example below is a **summarized** version of a tool's output. We don't want to see pages and pages of extraneous outputs: you are responsible for presenting the evidence that is relevant to the claims you're making.

```
app_main.o:     file format elf32-littleriscv


Disassembly of section .text:

[... some functions omitted for clarity ...]

000003fc <delay>:
 3fc:   1afd                    addi    s5,s5,-1
 3fe:   fe0a9fe3                bnez    s5,3fc <delay>
 402:   6ac1                    lui     s5,0x10
 404:   8082                    ret
```

Listing 3: Output of `objdump`

3. Appendix: full source code listing

This is the complete C source file that was referenced in Section 1.2.

```c
double add(double x, double y)
{
  return x + y;
}

double subtract(double x, double y)
{
  return x - y;
}
```

Bibliography

[1] T. Nugent, B. Boolenaar, and J. Weigert, “xxd(1) General Commands Manual.” May 2024.