Computer Systems and Networks UE WS 19/20

Report Assignment 1

Group Member

Platzer Andreas	11771913
Reindl Hannes	01532129
Manuel Sammer	11903022

Toolchain Setup

Setting up your Programming & Compilation Environment

Here we encountered the first pitfall with setting up virtual machines on our system. On some systems virtualization was deactivated in the BIOS which made it impossible to run Oracle Virtual Box at first, but we could easily fix this via adjusting settings inside the BIOS.

Getting the Raspberry Pi ready for Bare-Metal Programming & Testing your Toolchain

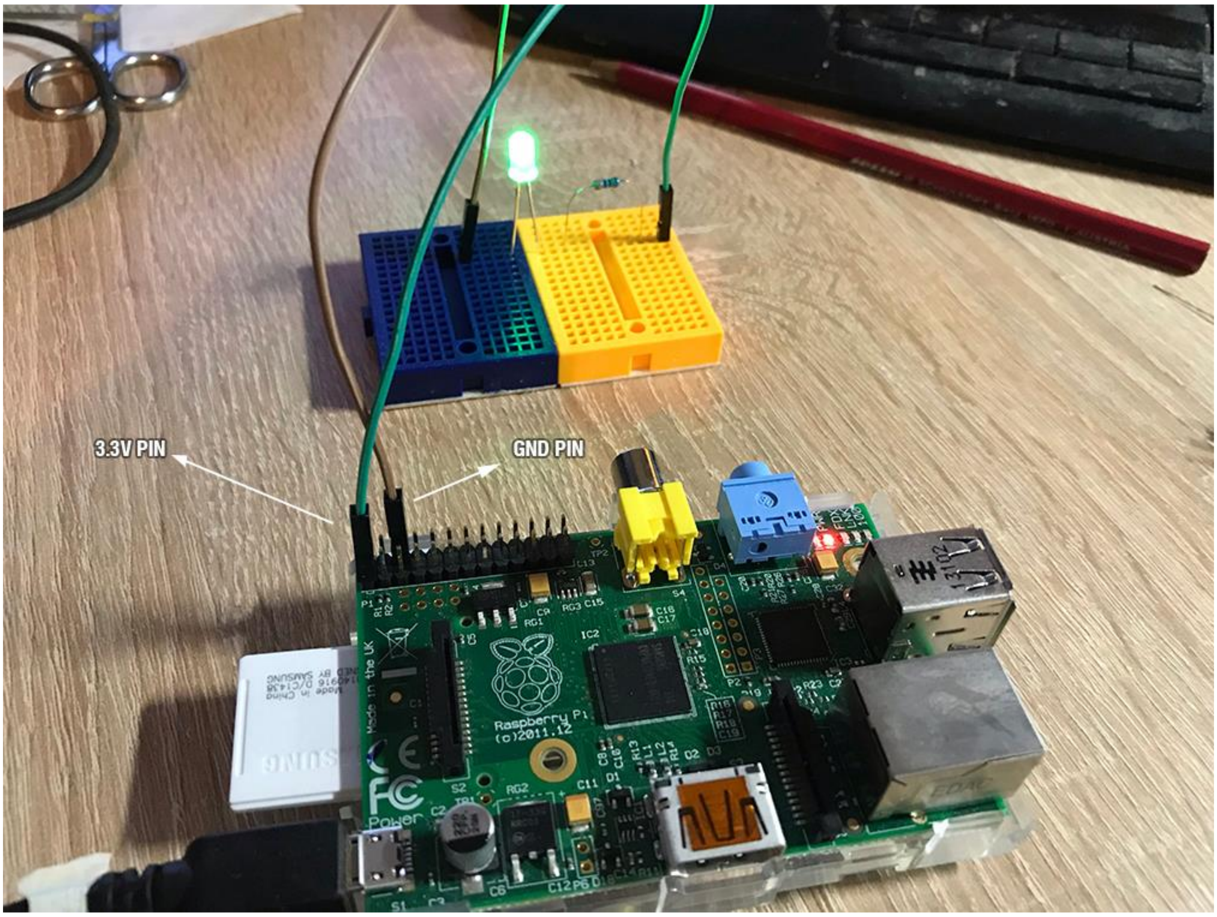
We encountered problems with our compilation environment on Ubuntu 18 LTS.

It was impossible for us to compile the test program properly which caused a lot of frustration at the beginning. After some hours of tinkering one of our members decided to check out the windows compiler environment and it worked instantly.

Bare Metal "Hello World"

Connecting all Components

This was an easy and fast task to accomplish. We only had to look up which side of the LEDs are the negative and positive pins and then it was easy with the provided manual.



Bare Metal ACT LED Blinking (b)

We read the provided tutorial as well as the manuals and based on this information we created an easy system to control the GPIO pins as input and output.

The way we realized this that we wrote three functions where one (setPortFunction) is responsible for declaring the type of the pin meanwhile the other two functions (turnOn, turnOff)are there to control if the pin is set to high or low in a very logical sense.

We still encountered some pitfalls. For example, we used Array arithmetic's but still used hex decimal values to get to the next element. Instead of array[1] we accessed array[4] because we used array[0x4] and not simply array[1] which would have been the second element in the array.

We also used a wrong start address (GPIO_START) where the GPIO settings can be edited.

Code inside: ***ass1_2_b.c***

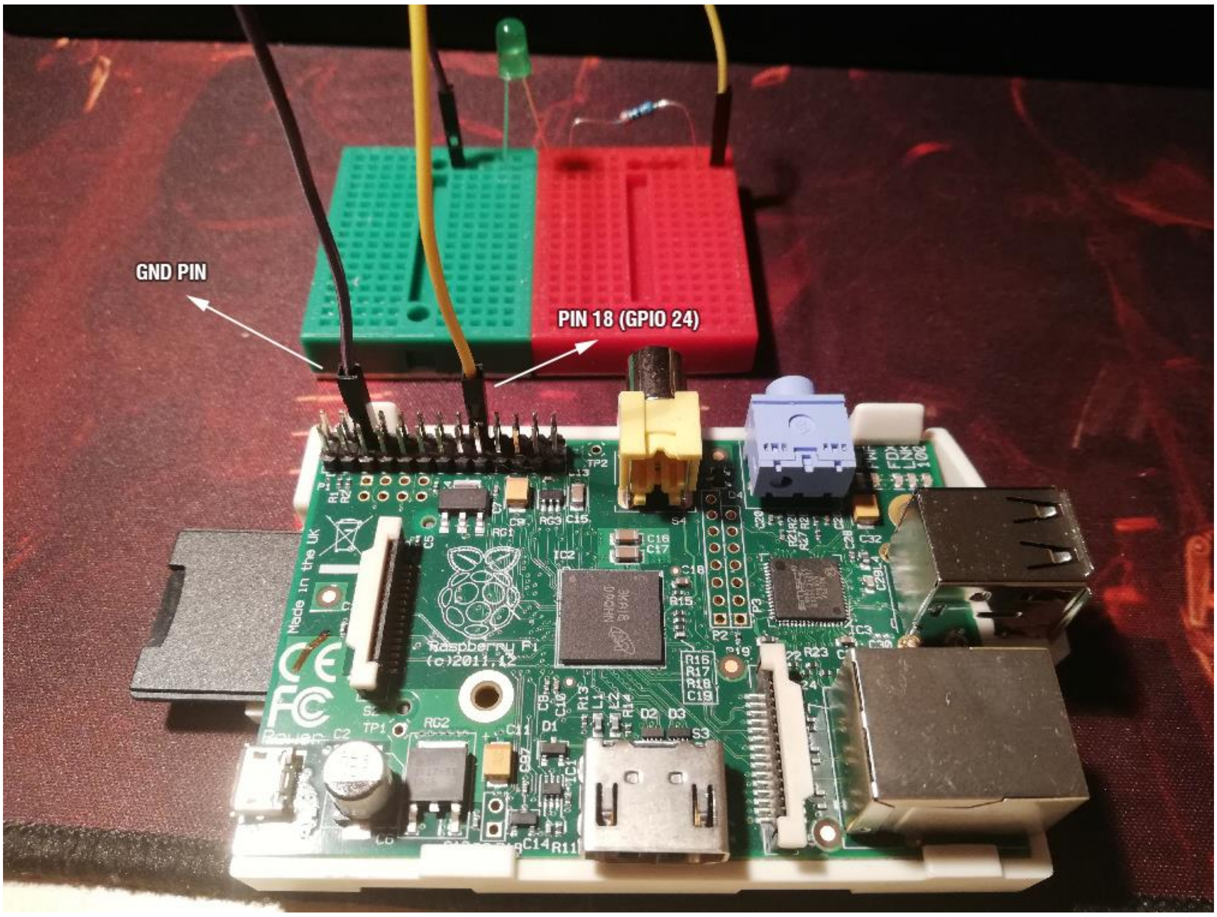
Compile with commands from the tutorial

Bare Metal ACT LED Blinking (c)

Based on the work in 2b we could simply adjust the code by editing 3 lines and it instantly worked for us without any problems by connecting the LED cables to the proper pins as seen in the manuals.

Code inside: ***ass1_2_c.c***

Compile with commands from the tutorial



Understanding the BCM2835

1. What is gpio[LED_GPFSEL] (in the code that blinks the ACT LED)?

   The array is an unsigned integer array which means we always access 32bits. At this position of the array we can adjust the bits of all ports in the range of port 10-19. These bits (3 each per port) decide how a pin works (Output, Input, AltFunctions). If we want to edit pin 15 we have to shift for example a 001 bit pattern 5 times 3 positions to the left.

2. How would you modify this line in order to turn all GPIO pins that are covered by the GP Function Select Register 2 into outputs?

   Note: We assume that the register is set to 0x0

   ```
   for(int counter = 0; counter < 10; counter++)
   {
     gpio[LED_GPFSEL] |= 1<<(counter * 3);
   }
   ```