Raspberry Pi Introduction

Adrian Wälchli

April 2, 2019

1 Overview

The Raspberry Pi[1] is a pocket-sized computer with minimal equipment and low power consumption (see fig. 1). It includes all essential parts on one board, that is, a 1.2 GHz Quad-Core CPU, 1 GB RAM, Ethernet, Wireless LAN, four USB ports, one HDMI port and a Micro-SD card with pre-installed *Raspbian* OS. As part of the computer architecture course you will solve two programming assignments using the Raspberry Pi as an interface to interact with LED's, buttons and other electronic components. For that purpose, you receive from us a package containing the following parts.

- Raspberry Pi
- Raspberry Pi enclosure
- power adapter
- HDMI cable
- SD card (already plugged in)
- GPIO extension board with connector cable
- breadboard
- and more (see table 1)

We already attached all necessary components to the breadboard so you use it right away. In order to work with the Raspberry Pi and start programming, you additionally need a keyboard, a mouse and a monitor with HDMI. You can use your own peripherals at home or the ones in the ExWi computer pool[2].

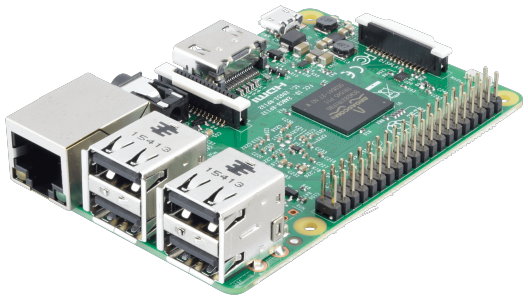
2 Start and Setup

If you are looking for a power button on the Raspberry Pi, you will not find one. The device automatically turns on as soon as it is supplied with power. You should reach the login screen within 30 seconds, otherwise, check the connection to the monitor and make sure you select HDMI as input. After successful login, it will show the Raspbian Desktop (fig. 1) where you can access programs and settings.

To solve the exercises, you will essentially only need the command line (or terminal), which you can also start from the Raspbian menu. You are of course allowed to install and use other programs, if you wish to do so. We assume that you have basic knowledge of the Linux command line. If this is not the case, please inform yourself about the basic commands to navigate folders, create, edit, and delete files, etc. As a text editor, we recommend `nano` oder `vim`.

[1] `https://en.wikipedia.org/wiki/Raspberry_Pi`

[2] Sidlerstrasse 5, rooms A93 and A94



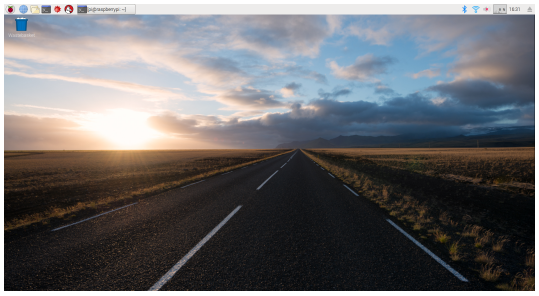


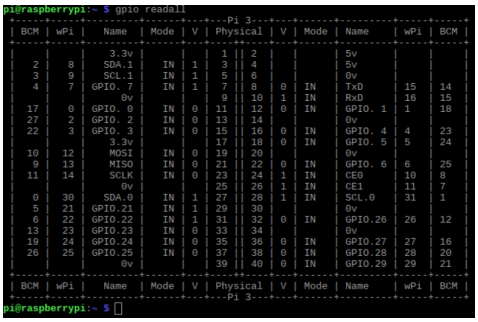
Figure 1: Raspberry Pi 3 Model B and Raspbian Desktop.

3 Using GPIO and Breadboard

GPIO Interface As you can see in figure 1, the Raspberry Pi has an array of 40 pins. This interface is called the *GPIO* (general purpose input/output). Every pin can be individually controlled and used for different purposes. With the command

```
$ gpio readall
```

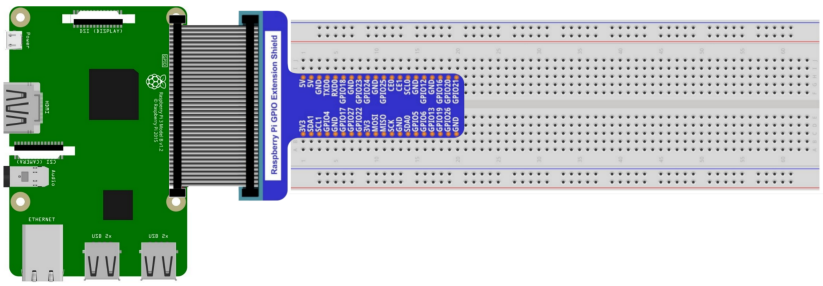
you can open an overview of all pins and their state.



```
pi@raspberrypi:~ $ gpio readall
 +-----+-----+---------+------+---+---Pi 3---+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 |     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
 |   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5v      |     |     |
 |   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
 |   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 0 | IN   | TxD     |  15 |  14 |
 |     |     |      0v |      |   |  9 || 10 | 1 | IN   | RxD     |  16 |  15 |
 |  17 |   0 | GPIO. 0 |   IN | 0 | 11 || 12 | 0 | IN   | GPIO. 1 |   1 |  18 |
 |  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
 |  22 |   3 | GPIO. 3 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4 |   4 |  23 |
 |     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 |   5 |  24 |
 |  10 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |     |
 |   9 |  13 |    MISO |   IN | 0 | 21 || 22 | 0 | IN   | GPIO. 6 |   6 |  25 |
 |  11 |  14 |    SCLK |   IN | 0 | 23 || 24 | 1 | IN   | CE0     |  10 |   8 |
 |     |     |      0v |      |   | 25 || 26 | 1 | IN   | CE1     |  11 |   7 |
 |   0 |  30 |   SDA.0 |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0   |  31 |   1 |
 |   5 |  21 | GPIO.21 |   IN | 1 | 29 || 30 |   |      | 0v      |     |     |
 |   6 |  22 | GPIO.22 |   IN | 1 | 31 || 32 | 0 | IN   | GPIO.26 |  26 |  12 |
 |  13 |  23 | GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v      |     |     |
 |  19 |  24 | GPIO.24 |   IN | 0 | 35 || 36 | 0 | IN   | GPIO.27 |  27 |  16 |
 |  26 |  25 | GPIO.25 |   IN | 0 | 37 || 38 | 0 | IN   | GPIO.28 |  28 |  20 |
 |     |     |      0v |      |   | 39 || 40 | 0 | IN   | GPIO.29 |  29 |  21 |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+---Pi 3---+---+------+---------+-----+-----+
pi@raspberrypi:~ $
```

The two columns in the middle represent the physical GPIO interface, where the pins are numbered from 1-40. The columns on the left and right describe the settings of each pin. The column `Mode` indicates the mode of a pin. For our exercises, the relevant modes are `IN` (input) and `OUT` (output). The associated input- or output value of a pin is shown in the column named `V` (value). You can ignore the columns `Name` and `wPi`. These are essentially different naming schemes for the same pins. In our exercises, we will use the "BCM" naming scheme in order to identify each pin (see first and last column respectively).

Breadboard To make things easier and organized, we use a breadboard to connect multiple electronic components. The breadboard is essentially a grid that lets us install and interconnect components without the need for soldering, and it keeps things organized and easy to debug. Connect the breadboard to the Raspberry Pi as shown below. Note that the cable only fits one way.



As you can see on the blue header, the pins are labeled with their names according to the "BCM" convention. There are also special pins that are used for constant power delivery (3.3V, 5V, GND).

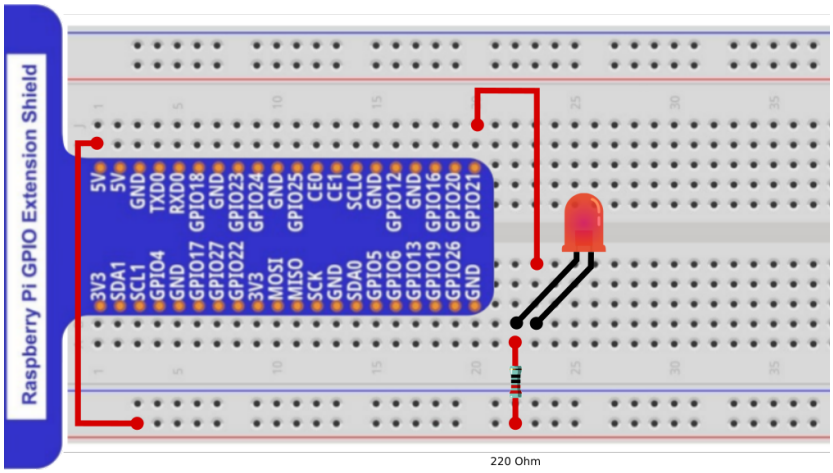


Figure 2: LED circuit diagram.

At this point, you don't need to be concerned about what the pre-installed components do and how they work. All necessary information for that will follow in the corresponding exercises in series 4 and 5.

LED Test For testing purposes, the command line tool `gpio` allows you to manually access each pin. As shown in figure 2, the LED on your breadboard is connected to pin 21. To turn the LED on or off, you first have to set the mode of the pin to "output". This can simply be done with

```
$ gpio -g mode 21 out
```

Now you can use the commands

```
$ gpio -g write 21 1
```

and

```
$ gpio -g write 21 0
```

to turn the LED on or off. Test it yourself! You can now see the effect of these commands on pin 21 in the overview with `gpio readall`. Furthermore, with `man gpio` you have an overview of everything else you can do with the `gpio` tool.
Exercise: Find out how to read an input signal on pin 18 and test it by pressing the button on the breadboard.

4 Assembly Tutorial: Blinking LED

As an introduction to the programming exercises, we will discuss a small program that makes the LED blink repeatedly. The procedure is very similar to the manual tests described above. We have put all necessary files in your home folder so you can directly run the code we show in this tutorial.[3] First, navigate to the source code folder and compile the program with `make`.

```
$ cd ~/Documents/blink
$ make
```

Now you can simply run the program.

```
$ ./blink
```

You should see the LED blink repeatedly. The program can be stopped with `Ctrl+C`. In the case that your LED does not work when running the program, please make sure that the breadboard is properly attached to the Raspberry Pi and cables are connected as shown in figure 2.

Next, open the source file `blink.s` with an editor of your choice. For example, use

[3] You can also find the code on ILIAS.

```
$ nano blink.s
```

directly in the terminal. You now see the code shown in Listing 1.

The source code is separated into different blocks: `main`, `configurePin`, `blinkLoop` and `exit`. The entry point is at `main`, where the external subroutine `wiringPiSetupGpio` is invoked. In particular, it is responsible for initializing the access to the GPIO address range.

In the next section, `configurePin`, the LED is set to output mode similarly to how you have done this with the `gpio` command. In detail, please note:

- `pinMode(pin, mode)` is an externally accessible function from the pre-installed `wiringPi` library.
- The two arguments to this function are stored in registers `R0` and `R1` before invocation.
- The constants `.LED_PIN` and `.OUTPUT` are defined at the end of the file and are used as placeholders to make the code more readable.

In the following section `blinkLoop`, the calls to `digitalWrite` and `delay` are handled in the same fashion. *Question:* What does `digitalWrite` do? *Exercise:* Change the source code such that the LED blinks faster.

Listing 1: blink.s

```
.global main
.func main

main:
        // This will setup the wiringPi library.
        // In case something goes wrong, we exit the program.
        BL      wiringPiSetupGpio
        CMP     R0, #-1
        BEQ     exit

configurePin:
        // Here we configure the pin.
        // We use the pin number 21 as defined at the bottom of
        // this file and set the pin to output mode.
        LDR     R0, .LED_PIN
        LDR     R1, .OUTPUT
        BL      pinMode

blinkLoop:
        // This is where the loop for blinking the LED starts
        // Turn the LED on
        LDR     R0, .LED_PIN
        LDR     R1, .HIGH
        BL      digitalWrite

        // Wait 500 milliseconds
        MOV     R0, #500
        BL      delay

        // Turn the LED off
        LDR     R0, .LED_PIN
        LDR     R1, .LOW
        BL      digitalWrite

        // Wait 500 milliseconds
        MOV     R0, #500
        BL      delay

        // Repeat
        B       blinkLoop

exit:
        MOV     R7, #1
        SWI     0


// We use GPIO pin 21 (BCM-style) to connect the LED.
.LED_PIN:               .word   21

// Define constants for high- and low signals on the pins
.HIGH:                  .word   1
.LOW:                   .word   0

// The mode of the pin can be set to input or output.
.OUTPUT:                .word   1
.INPUT:                 .word   0
```

Name	Quantity	Name	Quantity
Raspberry Pi	1	Enclosure for Raspberry Pi	1
Power adapter	1	HDMI cable	1
SD card	1	Breadboard	1
GPIO extension shield	1	40-pin GPIO connector cable	1
LED red/green/blue	1	Push button	2
LED bar	1	NPN transistor	1
220 Ω resistor	9	74HC595 shift register	1
1 kΩ resistor	1	Active buzzer	1
10 kΩ resistor	4	Jumper cable M/M	26

Table 1: Raspberry Pi Kit contents.