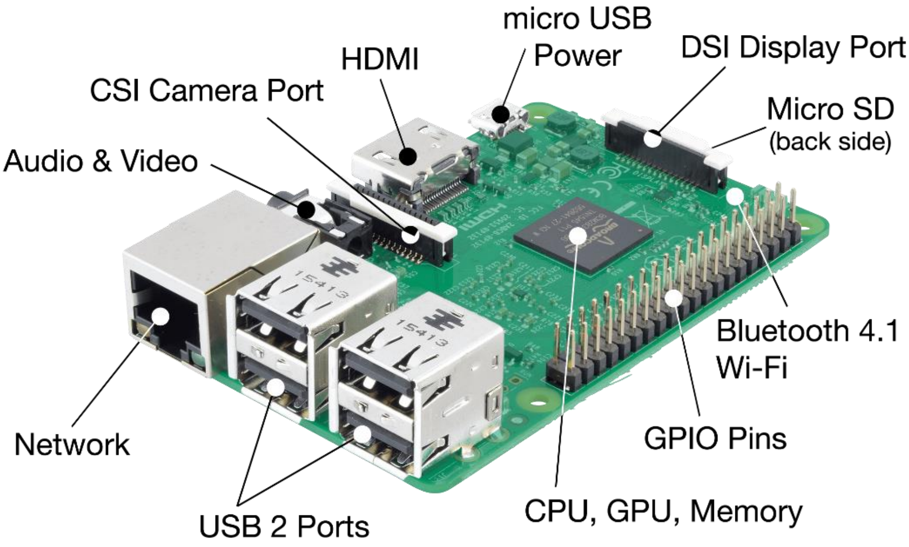
Introduction to the Raspberry Pi

This document contains a series of exercises to perform using the Raspberry Pi (https://www.raspberrypi.org/products/). The following figure shows the device's main ports.



Links to Raspberry Pi tutorials:

- https://pihw.wordpress.com/guides/direct-network-connection/
- http://www.circuitbasics.com/how-to-connect-to-a-raspberry-pi-directly-with-an-ethernet-cable/
- https://projects.raspberrypi.org/en/
- https://elinux.org/RPi_Projects
- https://www.element14.com/community/welcome
- https://ronnyvdbr.github.io/

Food:

- Raspberry Pi requires 5V power (usually via micro USB)
- Power can also be supplied via the GPIO pins (from another board)
    - Be careful when supplying power from the arduino, as the Raspberry Pi does not protect itself against overvoltage. A 5V to 3.3V converter is required (otherwise the Raspberry Pi may be damaged)
- GPIO pins can also be used to supply power to other boards (as long as the voltage is 5V) generally, an output of 3.3V is sufficient. A converter may be needed

Initial setup

1. **Memory card setup:** The Raspberry Pi needs a properly formatted memory card (bootloader + operating system) to function. The two main operating systems used with the Raspberry Pi are *Raspbian* (based on Debian) and *Pidora* (based on Fedora).
   a. Download the Raspberry Pi Imager[1]
   b. Insert a memory card of at least 4GB into a computer with a memory card reader
   c. Format memory card using Raspberry Pi Imager

2. **Links** :
   a. Insert formatted memory card into Raspberry Pi
   b. Connect the HDMI cable between the Raspberry Pi and the monitor, the network cable between the Raspberry Pi and the router (or connect via Wi-Fi) and the mouse and keyboard to the USB ports
   c. Connecting power to the micro-USB

3. **First *boot*** :
   a. Install Raspbian Full and make the required settings: Country / Language / Timezone; Password; Screen; wifi
   b. Let the configurator update the software and start the Raspberry Pi

4. **Configure remote connections** : In the console, enter the command `sudo raspi-config`
   a. Enable SSH : option 3 – `Interface Options` | option `I2 - SSH` | Activate. Test: on the computer, install Putty [2]. Create a binding to the [3]Raspberry Pi IP [4](Hostname)
   b. Enable VNC : option `5 – Interface Options` | option `I3 – VNC` | Activate. Test: on the computer, install the VNC Viewer [5]. Create a binding for Raspberry Pi [4]IP [3]

[1] https://www.raspberrypi.com/software/

[2] **Download Putty** : https://www.putty.org

[3] To find out which is the IP assigned to the Raspberry Pi, in the console, run the command `ifconfig`

[4] Username: " `pi` ", password: the defined during configuration

[5] **Download VNC viewer** : https://www.realvnc.com/pt/connect/download/viewer/

c. Configure the Hostname : option `1 - System options` | option `S4 - Hostname` | Define a hostname for the Raspberry Pi.

5. **Configure NetBeans IDE to deploy via network** :

    a. Install NetBeans IDE (download: https://netbeans.apache.org/download/index.html)

    b. In NetBeans go to menu “Tools” > “Java Platforms” click on “Add Platform” and choose “Remote Java Standard Edition”

    c. To fill in:

Platform Name	Raspberry Pi
host	Raspberry Pi IP
username	Raspberry Pi username (pi)
password	Raspberry Pi Password (idc2022)
Remote JRE Path	in principle it will be /usr/lib/jvm/java-11-openjdk-armhf

    d. Test the connection:

        i. In NetBeans, create a project (ex: RaspberryPi_teste0) that just prints “I did it!”.

        ii. Configure NetBeans for this project to run on the Raspberry Pi

            1. Click on “File” and then on “Project Properties (RaspberryPi_teste0)”
            2. Click on “Run” on the left and then on “Runtime Platform”,
            3. Choose the one defined in the previous step (Raspberry Pi) and enter a name for the configuration (eg Raspberry Pi)
            4. Click OK on all windows

        iii. run the project

        iv. Check, on the Raspberry Pi, if a folder appears in the /home/pi/NetBeansProjects/ folder with the name of the project created in NetBeans

        v. Go to /home/pi/NetBeansProjects/ RaspberryPi_test0/dist and run the jar (java -jar RaspberryPi_test0.jar)

        vi. “I got it” should be printed

6. **Configure NetBeans IDE to use pi4j** :

    a. Download and extract the pi4j ZIP to your computer: https://pi4j.com/download/pi4j-1.2.zip

    b. Create a project in Netbeans (ex: RaspberryPi_teste1). Ensure that “Create Main Class” is selected and that the class is called ControlGpioExample

    c. In the project tree, right-click on “Libraries” and then on “Add Library”

    d. Create a library using the name “Pi4J-Core”

    e. On each Tab, use “Add JAR/folder” to add the extracted jars:

Tab	**jar**
Classpath	pi4j-core.jar
Sources	pi4j-core-sources.jar
javadoc	pi4j-core-javadoc.jar

    f. In the ControlGpioExample class insert the desired code (referred to in the “First Raspberry pi/gpio/java project)

    g. Configure to run on raspberry (as in the previous configuration)

    h. To execute

7. **Install pi4j on Raspberry Pi** . In the terminal, run the command:

```
curl -sSL https://pi4j.com/install | sudo bash
```

8. **Install WiringPi on Raspberry Pi** . In the terminal, run the commands:

   - `sudo apt-get update`
   - `sudo apt-get dist-upgrade`
   - `git clone https://github.com/WiringPi/WiringPi.git`
   - `cd WiringPi`
   - `./build`

9. **Install MySQL (MariaDB) on Raspberry Pi** :

   a. run the commands

   - `sudo apt install mariadb-server`
   - `sudo mysql_secure_installation`

   b. In the configuration, perform the steps:

   - enter
   - n + enter
   - y + enter
   - enter password + enter
   - re-enter the password + enter
   - n + enter
   - n + enter
   - n + enter
   - n + enter

   c. Open mysql- Run the command `mysql -u root -p` (enter the password defined in the previous step)

   d. Create a database with just one table (to be used in exercise 6). In mysql, run the code:

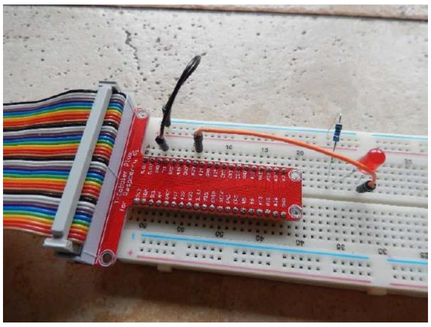
```
create database rpi_teste;
use rpi_teste;
CREATE TABLE record(
   datetime VARCHAR(50),
   pin VARCHAR(50),
   state VARCHAR(50));
```

Exercises

1. First raspberry pi/gpio/python project

Material needed:

- 1 Raspberry Pi
- 1 Pi Port extender
- 1 breadboard
- 2 connectors (wires)
- 1 LED
- 1 resistor (eg: 10Ω)



Procedure:

1. Connecting the port extender between the breadboard and the Raspberry Pi
2. Connect the port extender GND to the negative of the board (black wire in the figure)
3. Connect the negative resistance to another part of the board (see figure)
4. Connect the negative of the LED (shorter leg) to the resistor (see figure)
5. Connect the positive of the LED (longer leg) to port 23 of the port extender (orange wire in the figure)
6. On the Raspberry Pi, create the led.py file with the following content:

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(23,GPIO.OUT)
print("LED on")
GPIO.output(23,GPIO.HIGH)
time.sleep(1)
print("LED off")
GPIO.output(23,GPIO.LOW)
```

7. Run the code with the command sudo python led.py

2. Blink 3 times for a second with python

Create led3.py file which, instead of blinking the LED just once per second, blinks 3 times per second (the time for the led to be on/off is 1 second).

3. Blink X times for Y seconds with python

Create the file led_arg.py which, instead of flashing the LED just once for a second, or flashing 3 times for a second, takes two arguments on the command line. The first argument defines how many times to blink. The second argument defines the time it should be on/off. For example, `sudo python led_arg 5 1` should flash the LED 5 times for 1 second each time.

4. First Raspberry pi/gpio/java project

With the same hardware configuration used for Exercise 1 with Python.

Procedure:

1. Following point 6.f) of "6. Configure NetBeans IDE to use pi4j", insert existing code in the file codigo1.txt (attachment) in the ControlGpioExample class (in this case the LED is connected to pin 29 because, using WiringPi/Pi4j, the numbering is different, according to the existing image on the last page of the document.
2. To execute. Check that the LED flashes as indicated in the code.

5. Show the status of a button in the console

Connect a button with one of the pins connected to 3.3 volts (Pi extender 1) and the other connected to GPIO 2 (Pi extender 13).

Procedure:

1. In a new project similar to the previous one, insert the code from the codigo2.txt file (attachment) into the class where the main is located.
2. To execute. When you press the button, the status change should appear on the console.

6. Save the state of a button in a database

With the same hardware configuration used for Exercise 1 with Python.

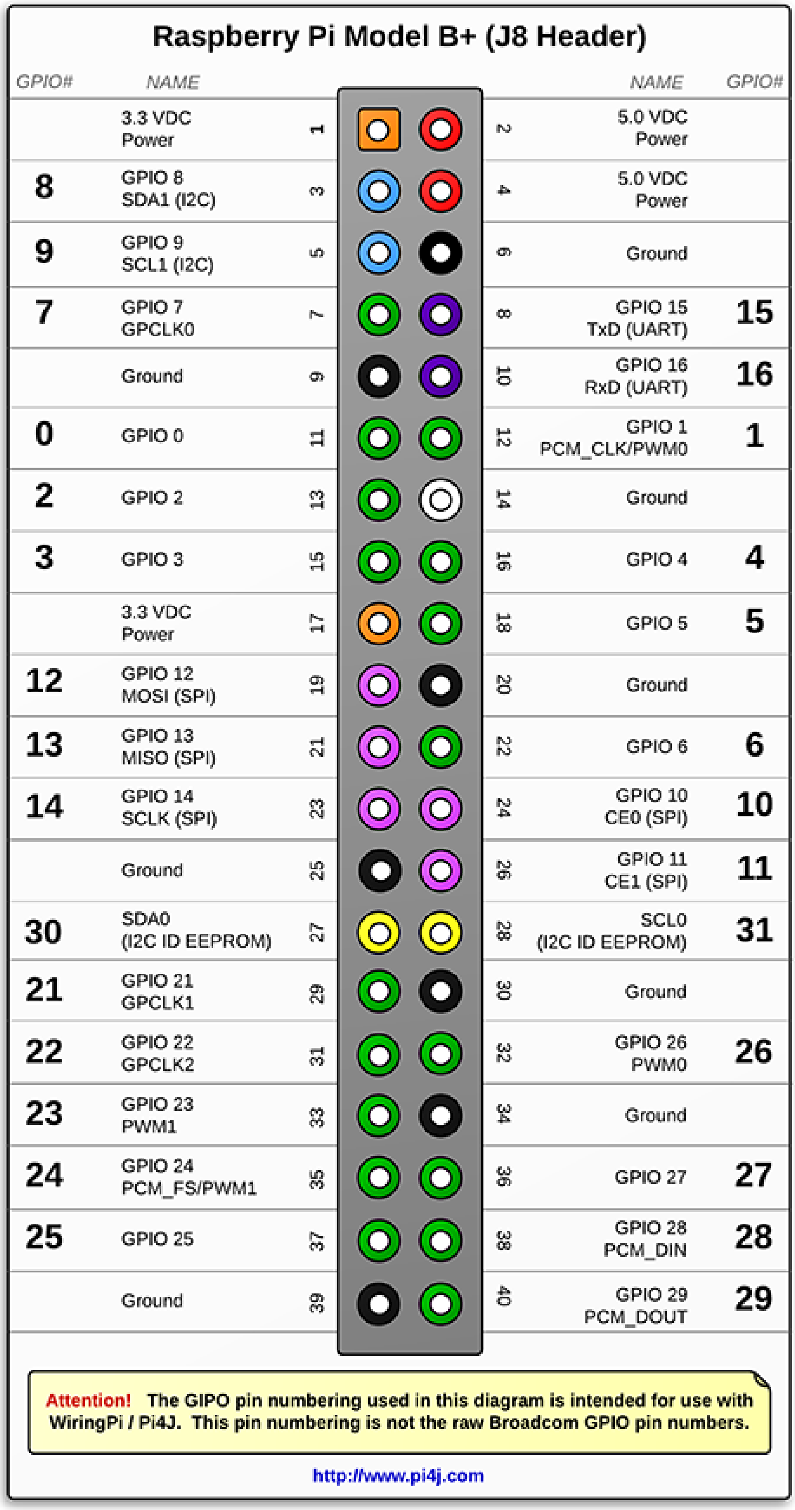
Procedure:

In the previous project, adapt the code so that, instead of the information being shown to the console, it is stored in a database

1. Download the mysql/java connector jar file[6]
2. Add the connector to the project:
    a. Right-click on the project name > Properties
    b. On the left, select Libraries
    c. Click on Add JAR/Folder and select the jar file obtained
3. Replace the main class code with the existing one in the codigo3.txt file (attached)
4. To execute. Check that, when pressing the button, the records are inserted in the database table (use the command `SELECT * FROM record;` in mysql)

[6] https://static.javatpoint.com/src/jdbc/mysql-connector.jar

Pin numbering in WiringPi



Raspberry Pi Model B+ (J8 Header)

GPIO#	NAME			NAME	GPIO#
	3.3 VDC Power	1	2	5.0 VDC Power	
8	GPIO 8 SDA1 (I2C)	3	4	5.0 VDC Power	
9	GPIO 9 SCL1 (I2C)	5	6	Ground	
7	GPIO 7 GPCLK0	7	8	GPIO 15 TxD (UART)	15
	Ground	9	10	GPIO 16 RxD (UART)	16
0	GPIO 0	11	12	GPIO 1 PCM_CLK/PWM0	1
2	GPIO 2	13	14	Ground	
3	GPIO 3	15	16	GPIO 4	4
	3.3 VDC Power	17	18	GPIO 5	5
12	GPIO 12 MOSI (SPI)	19	20	Ground	
13	GPIO 13 MISO (SPI)	21	22	GPIO 6	6
14	GPIO 14 SCLK (SPI)	23	24	GPIO 10 CE0 (SPI)	10
	Ground	25	26	GPIO 11 CE1 (SPI)	11
30	SDA0 (I2C ID EEPROM)	27	28	SCL0 (I2C ID EEPROM)	31
21	GPIO 21 GPCLK1	29	30	Ground	
22	GPIO 22 GPCLK2	31	32	GPIO 26 PWM0	26
23	GPIO 23 PWM1	33	34	Ground	
24	GPIO 24 PCM_FS/PWM1	35	36	GPIO 27	27
25	GPIO 25	37	38	GPIO 28 PCM_DIN	28
	Ground	39	40	GPIO 29 PCM_DOUT	29

Attention! The GIPO pin numbering used in this diagram is intended for use with WiringPi / Pi4J. This pin numbering is not the raw Broadcom GPIO pin numbers.

http://www.pi4j.com