C. RODRIGUES	TP Internet of Things

TP 2 : Questions

Goals:

1. **Learn to manipulate a Raspberry Pi**
2. **Learn to assemble and use different devices with a Raspberry Pi**
3. **Implement simple applications in Python, using sensors and actuators connected to the Raspberry**

II – Exercises:

1. **Before proceeding to the exercises, you MUST complete the tutorial examples to understand how to assemble and manipulate the devices we are going to use during this course.**
2. **You will deliver a report that will contain the python code and a picture or sketch (preferable) of the model you assembled for each question.**

Exercise 01: Create an application that prompts the user to input an angle (between 0º and 180º degrees) and rotates the shaft to that angle.

Exercise 02: Turn on a red LED when no movement is detected by the PIR motion sensor. Activate a green LED when movement is detected.

Exercise 03: Retrieve and display the current ambient temperature in Celsius and Fahrenheit every 3 seconds.

Exercise 04: Develop a program that requests a temperature in Celsius from the user. Turn on an LED only when the sensor-detected temperature exceeds the specified value. Turn off the light if the temperature decreases.

Exercise 05: Extend the previous exercise by adding another LED. This time, the user inputs values for both temperature and humidity. The new LED turns on if humidity exceeds the user-defined threshold.

Exercise 06: Create a program that reads the temperature and humidity of the environment and writes them to a file.

Exercise 07: Utilize the HC-SR04 ultrasonic sensor to calculate the distance between the sensor and an object. Allow the user to choose between meters and feet.

Exercise 08: Construct a circuit with an ultrasonic sensor and an LED. The LED blinks faster as an object approaches. When the object is less than 5 cm away, the LED blinks for 50 ms. At 1 m, the blinking period is 1s. For distances between 5 cm and 1 m, the blinking period is proportional. If the distance exceeds 1 m, the LED remains off.

Exercise 09: Simulate a protected area using a sonar to detect when someone surpasses an imaginary fence. Incorporate a red and green LED and a buzzer. When an object is within 1 meter, activate the buzzer for 5 seconds, turn on the red LED, and turn off the green LED. Otherwise, turn off the buzzer, turn on the green LED, and turn off the red LED. Allow the user to modify the distance to activate the alarm, respecting detection limits.

Exercise 10: In this simulation, we aim to replicate an automatic door commonly found in malls. The project comprises a servo and a motion sensor. Initially, the system detects a person in front of the door (near the motion sensor). If a person is detected, the system activates the servo to move the door to 90º (indicating an open position). Subsequently, if no one else is detected, the servo returns to 0º (indicating the door closing).

Additionally, two LEDs (red and green) are integrated. The system allows the manager to definitively close the door via software. Using a Python script, the manager can set the door system to either 'enabled' or 'disabled'. When in the 'enabled' state, the system operates as described in the initial phase of this exercise. Conversely, when set to 'disabled', the door remains closed even if someone is detected. The Python script prompts the manager to choose between two options: 1 = Enable Door or 2 = Disable Door.

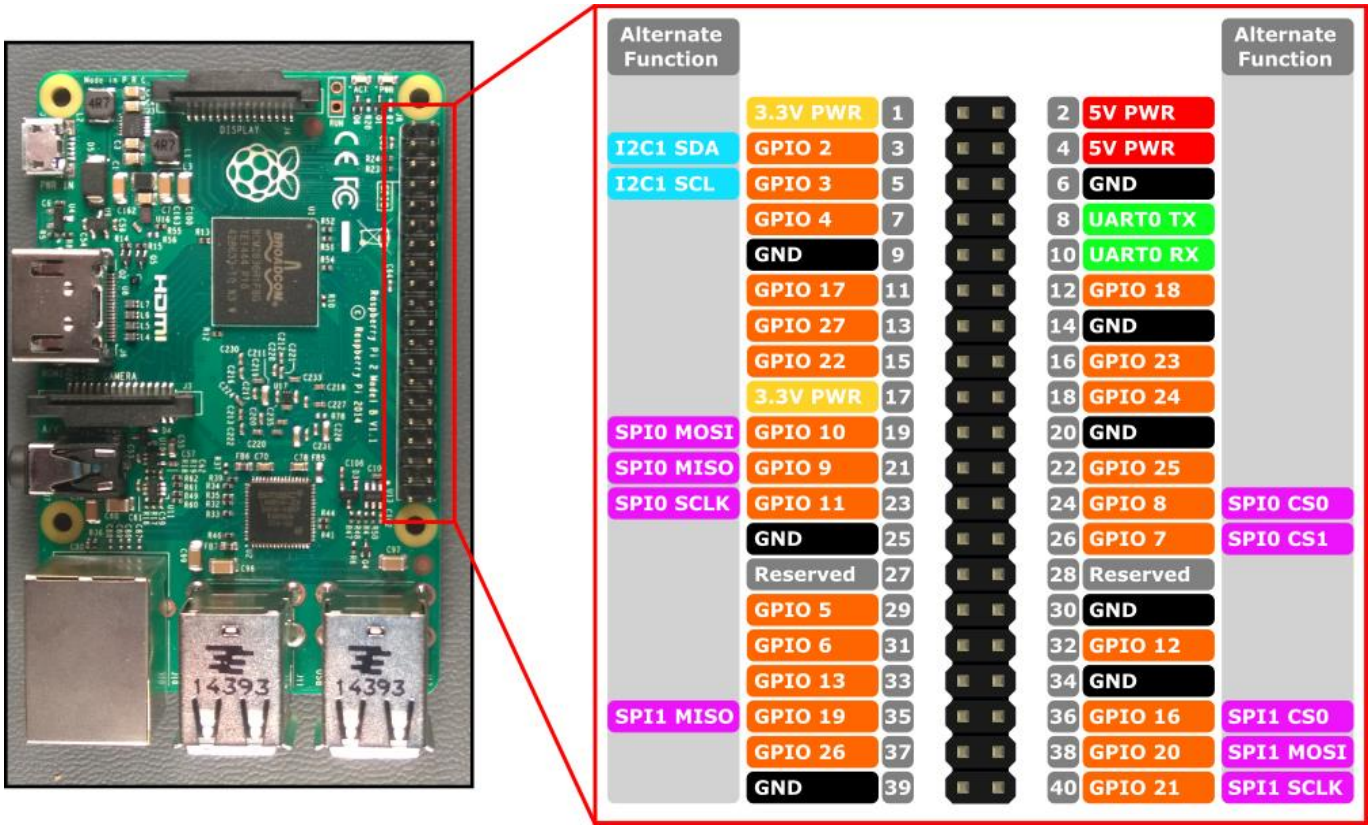
Exercise 11:

This exercise introduces the concept of a basic smart home, featuring sensors and actuators controlled by both a user and automated services. In Table 1, the first column lists various available services, including the alarm system (formed by multiple motion sensors strategically placed around the house for enhanced vigilance, though only one will be utilized), Lights (represented in different colors to denote different rooms), and Temperature control (deployed through multiple sensors spread inside the house to detect fires). All these services are depicted in the project via components in the third column of Table 1. The system must efficiently manage all services through a menu, incorporating the options detailed in the second column of Table 1.

To implement this simple smart home, consider utilizing timeouts or threads for enhanced functionality.

Services	Actions	Component
Alarm system	1 - Enable 2 - Disable	Buzzer and Motion
Lights	1 - Enable [Room] 2 - Disable [Room] 3 - Turn all on 4 - Turn all off	LED Red - Living Room LED Green - Bedroom Led Yellow - Kitchen
Temperature	1 - Set value to trigger 2 - View temperature	Temperature and Buzzer

Table 1: Smart Home



For more information about Raspberry Pi GPIO ports, check:

https://pinout.xyz