Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Automation and Applied Informatics

SMART HOME IOT PROJECT

using ESP8266 microcontroller with WiFi module

Contents

Abstract

The development of Smart Homes is necessary in the accelerated advancement of technology and busy lifestyle. The goal, among others, is ensuring comfort in the daily life, the assistance of elderly, handicapped people and saving up money and energy.

This project should investigate the design and development of such Smart Home system with the usage of the ESP8266 WiFi module. The development and design process should contain the following points:

1. Do research on the possible IOT applications of ESP8266
2. Study the documentation of ESP8266 and set up its required environment (bread board with 3.3V power supply, Serial Programmer, Arduino (optional), wires, resistors, capacitors etc.)
3. Study and evaluate Smart Home compatible devices
4. Design and set up the Smart Home System
5. Develop a smartphone software that visualizes the data of the sensors and enables automatic and manual control of a household device

1 Introduction

In the rapidly evolving technological world, increasing amount of tools are available, that can gather information or intervene in a process. Connecting these devices, new systems can be created, such that they are able to make decisions independently. The Internet of Things (IoT) based on this idea creates a network of interconnected devices. It connects embedded systems to the internet that can gather data or intervene, so they can communicate, interact, collaborate and help the operation of each other.

IoT has countless applications such as Home Automation (also called Smart Homes), Industry 4.0, Energy Management, etc.

Home Automation involves the control and automation of lighting, heating, ventilation, air conditioning, security and home appliances.

Industry 4.0 connects the supply chain and the ERP system directly to the production line to form an integrated, automated and, potentially, autonomous manufacturing processes that make better use of capital, raw materials, and human resources. [1]

Energy Management is an application form, where the system measures the consumption through connected sensors and reduces the consumption while keeping the desired output, e.g.: controlling a heating system.

The goal of this project is to design and develop an event based Smart Home System using an ESP8266 WiFi module that can be controlled from an Android compatible phone. To create a system where any actuators can be inserted and controlled with events such as certain sensor values. To illustrate the usage and operation of such system, a specific event based application will be implemented.

A coffee machine will be event controlled. It will be switched on and off based on time, and temperature sensor values, or manually with a phone.

2 State of the Art

The ESP8266 WiFi module has been used in several smart home projects in the last 2-3 years, because of its small size and cost efficiency. There have been multiple configurations for different programs such as controlling lights, home appliances, and reading sensors.

Numerous projects were created that enables the user to create their smart home application easily. Without the need for completeness, here are a few from them.

2.1 WiFi-IoT Beta

It is a firmware builder, that lets the user build a firmware for the ESP-01 by just selecting supported features. These features include sensors, services, hardware, systems and displays. [2]

2.2 Controllium

It is an Android application that enables the user to control ESP8266 projects. The user can send data to the ESP or receive from it using WiFi. The application can control multiply ESPs. [3]

2.3 Mechanical Wall Plug

It is a wall plug (**Figure 1**), that can be controlled with a mechanically programmable wheel. It can control the current that goes through it. Its time resolution is 30 minutes, so it can decide if current should flow or not in every half an hour.

Figure 1 Mechanical wall plug

3 Theoretical Background

The coffee machine controlling application consists of an Android phone, a temperature sensor unit and a relay unit. The phone is the central unit, that receives the data, sends out the commands to the relay unit. The temperature sensor unit is reading the temperature and sends it to the phone. The relay unit receives the commands and sets the relay accordingly. The generation, evaluation and handling of the events happen on the phone. The units only read and send the data, so later new features can be easily introduced, like machine learning based decision making, etc.

The sensor and relay units consist of an ESP8266 and a temperature sensor or a relay respectively. They contain other parts that are required for proper operation e.g.: resistors, capacitors, power supply parts etc.

3.1 ESP8266 module

The ESP8266 is a low-cost WiFi chip with full TCP/IP stack and MCU (Microcontroller Unit) capabilities produced by a Shanghai-based Chinese manufacturer, *Espressif Systems*. The first module came to attention in 2014 was the ESP-01 from a third-party manufacturer, Ai-Thinker. [4] This is the module that is used during this project.

There are three ways of using this module, in order of increasing complexity [5]:

1. Sending it *AT* commands from a computer via an USB to serial adapter. This is mostly useful for testing and setup.
2. Interfacing with an Arduino or any other microcontroller and using this board as a peripheral.
3. Programming the module directly and use its GPIO pins to talk to sensors, eliminating the need for a second controller.

In this project the 3rd option of programming is used.

3.2 Temperature Sensor

The ESP-01 does not contain output connections for the Analog-Digital-Converter, therefore, analog sensors can’t be used with the module without making

modifications to the board, so the DS18B20 one-wire temperature sensor is chosen. It can measure temperature between -55°C - 125°C. The accuracy of the sensor is 0.5°C between – 10°C – 85°C. The accuracy is enough for the current application, because the only thing that is important, if the warm coffee reached a certain level in the carafe or not.

To use the temperature sensor, its library can be downloaded in the Arduino Framework from the Library manager under 'Sketch/Include Library/Manage Libraries…'.

3.3 Relay

As a coffee machine should be controlled, the easiest way is to control an extension cord where the machine is plugged in. To control that, a 5V-230V relay is chosen, because the mains electricity should be controlled with the low power (3.3V) ESP-01.

To switch a relay's state only a certain voltage should be provided between the input pins of the relay.

It is worth mentioning, that an ESP WiFi relay module [6] is commercially available, that is almost what this project needs. The only problem with the above module configuration is that it can't be powered from a power bank or the mains electricity without external parts.

3.4 Arduino

As stated in Chapter 3.1, the ESP modules will be programmed directly. The modules can be programmed with Arduino, an open-source electronics platform based on easy-to-use hardware and software [7] through an FTDI Programmer.

The module has to be connected to the FTDI programmer as **Figure 2** shows.

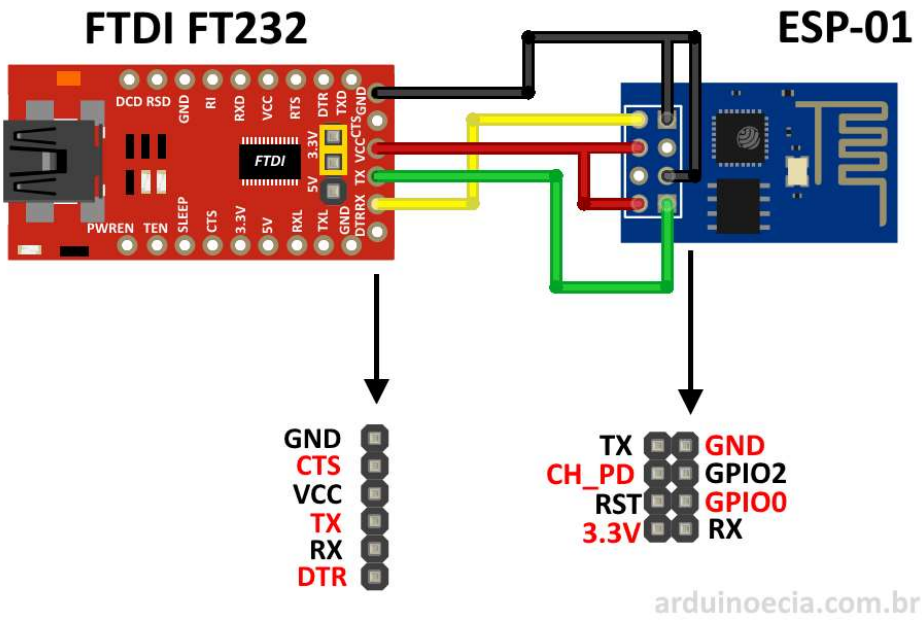

Figure 2 FTDI Programmer and ESP wiring diagram [8]

Then to use it with Arduino a special board manager should be installed to the Arduino platform as shown in [9] article. Then it can be programmed as the article shows.

3.5 Python

To program the Android compatible phone python is used, as it is a platform independent language, so the software can be used on computers too without the limitation of the operating system. [10]

To create graphical interface for the software Kivy is used. [11] It is an open-source platform independent python library for rapid development of applications that make use of innovative user interfaces, such as multi-touch apps.

4 System design

The coffee machine controlling system mainly comprises 3 parts as a sensor unit, a relay unit, and a phone. The sensor unit is only responsible for reading the sensor, and sending its data to the phone application. The relay unit is responsible for controlling the relay by commands it gets. The phone application is responsible for reading the data from the sensor, sending the commands to the relay unit and making decisions based on events.

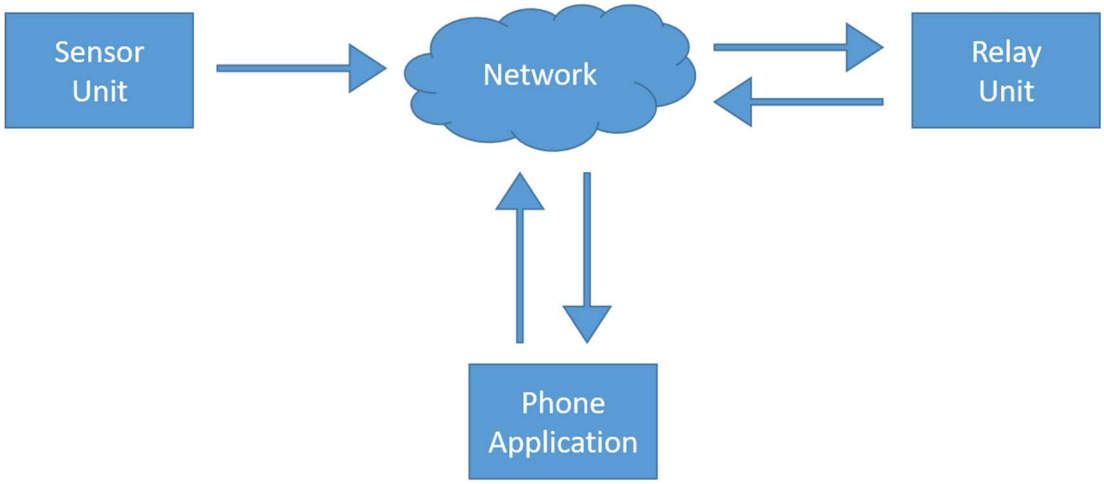

Figure 3 Block Diagram of System

The block diagram of the system can be seen on **Figure 3**.

4.1 Operation concept

The concept of operation is that the phone turns the coffee machine on if a certain time is reached or manually by a button. Then it turns the machine off when the temperature of the top of the coffee carafe get above a certain value. The temperature of the carafe's top will get high when warm coffee reaches that level.

4.2 Hardware Design

The design of the hardware can be separated into the design of the relay unit, the sensor unit and the power supply, as it will be the same in both units.

4.2.1 Power Supply

The ESP operates on 3.3V, but the relay works with 5V, so the perfect solution is a 5V power supply and a 3.3V regulator. 5V can be provided from USB. It is a good

solution, because it can operate from the mains electricity or from a power bank, making it portable.

As the voltage can get high or drop down some capacitors should be inserted in the circuit to flat the voltage out. The schematic of the power supply can be seen on **Figure 4**.

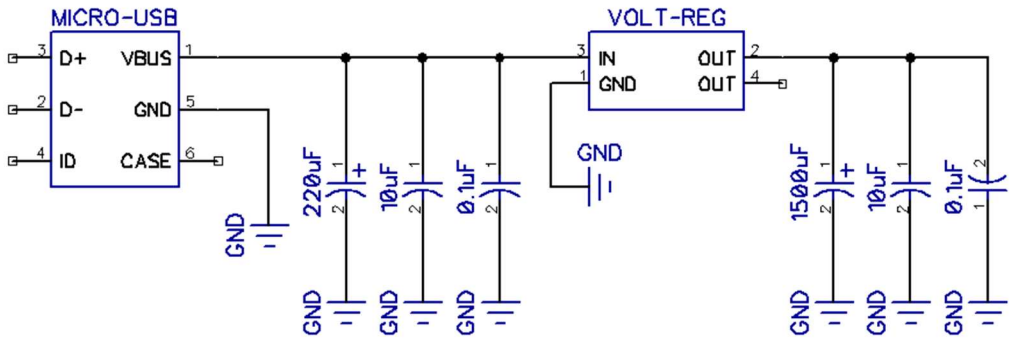

Figure 4 Schematic of Power Supply

Before radio devices (such as ESP-01) higher capacitance should be used, therefore, there is the 1500uF. The 220uF value comes from the 5V to 3.3V regulation. Multiple capacitances should be used in one application to filter out the voltage changes in the different frequencies. That is why the 10uF and 0.1uF capacitors are used.

4.2.2 Sensor Unit

The sensor unit consists of the elements of the power supply, the ESP-01, DS18B20 and a 47k resistor. The wiring of the temperature sensor can be seen on **Figure 5**.

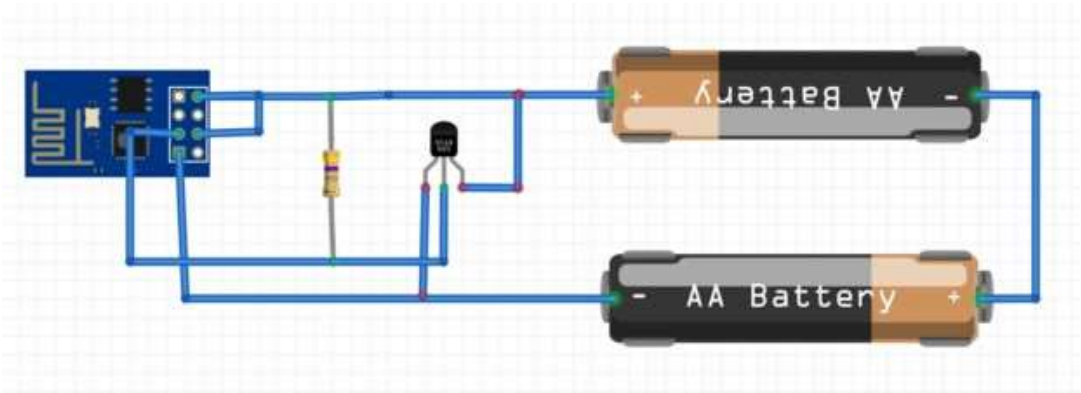

Figure 5 Wiring of DS18B20 sensor to ESP-01 25 [12]

As the ESP-01 generates heat, the temperature sensor should be placed further from the module. On the other hand, the sensor should be portable and easy to place. Therefore, the temperature sensor is connected to the board with an RJ11 connector, so it can be positioned anywhere without moving the board.

Keeping these in mind, the schematic of the sensor unit can be seen on **Figure 6**.

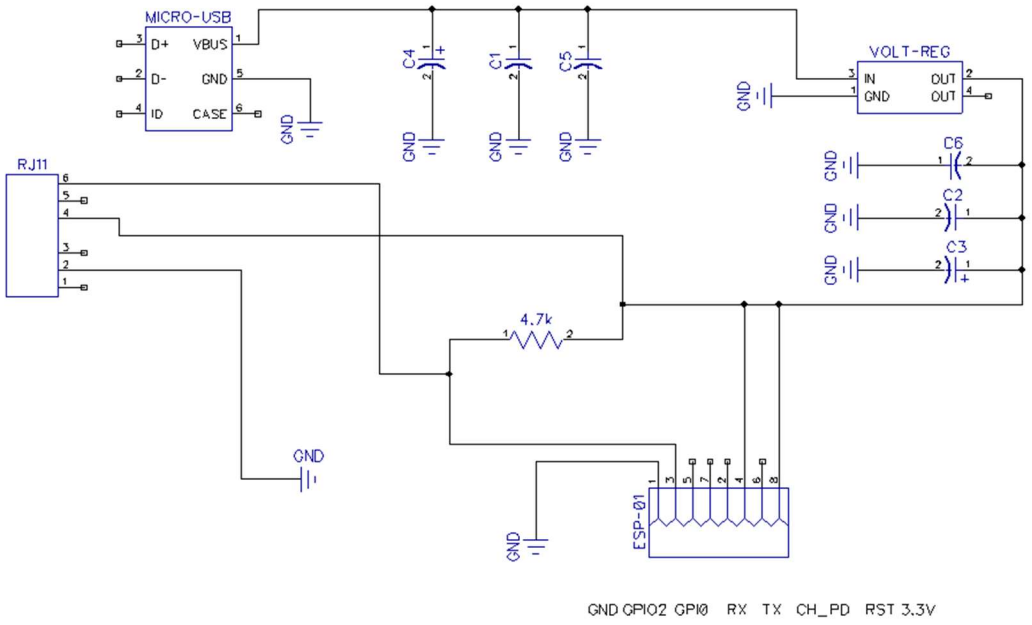

Figure 6 Schematic of Sensor Unit

The printed circuit board (PCB) should be the smallest while still having all the components, so the unit stays portable and easy to place. On the other hand, the PCB will be printed with ironing method, so a trace width of 0.7 mm is chosen. [13] Keeping these in mind, the PCB can be seen on **Figure 7**.

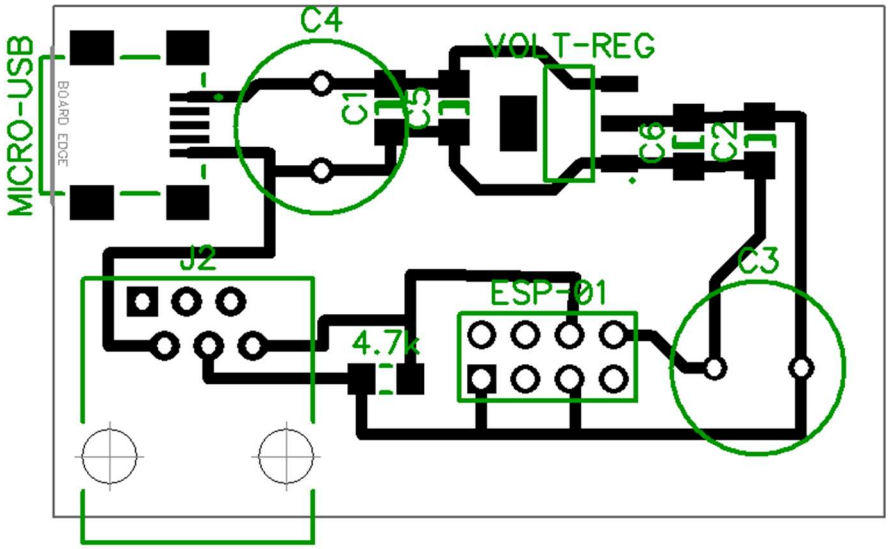

Figure 7 The PCB of sensor unit

The printed PCB with the ironed parts can be seen on the next figures:

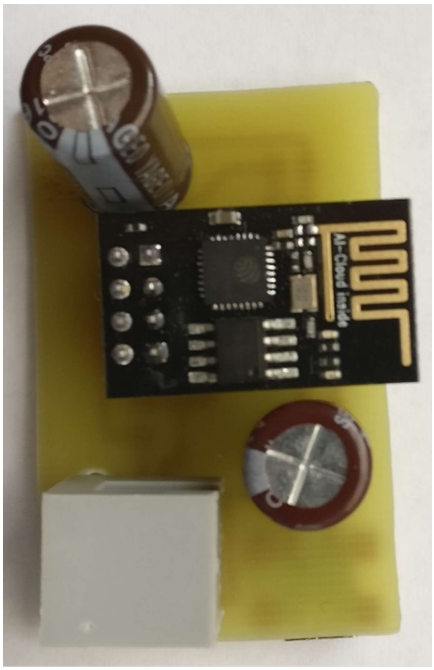

Figure 8 The top of the PCB of the sensor

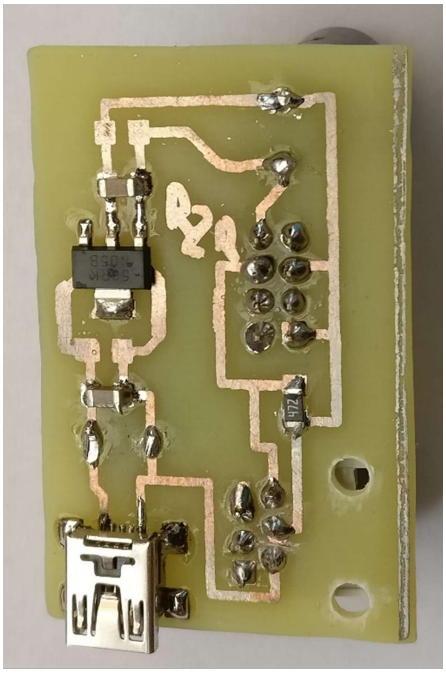

Figure 9 The bottom of the PCB of the sensor

4.2.3 Relay unit

The relay unit consists of the power supply, the ESP-01, a relay module, a transistor and 2 resistors. The configuration can be seen on **Figure 10**.

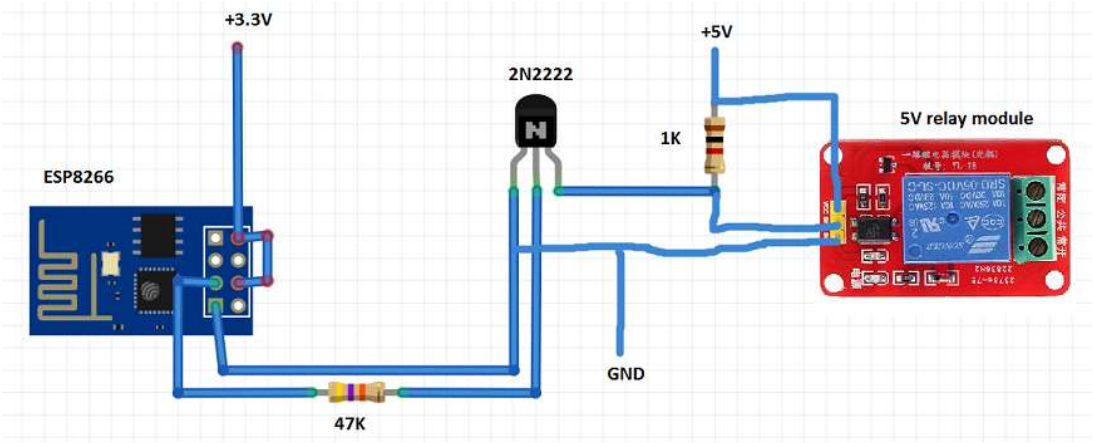

Figure 10 Relay unit wiring [14]

The relay module is built on the unit from parts, not using a commercially available module. The schematic and the PCB layout of the relay unit can be found on **Figure 11** and **Figure 12.**

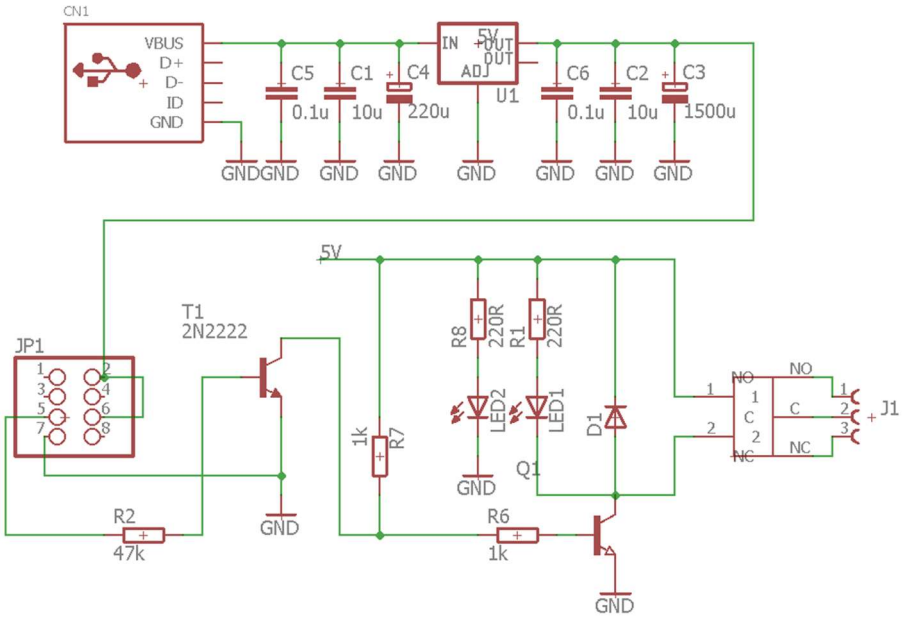

Figure 11 Schematic of the relay unit

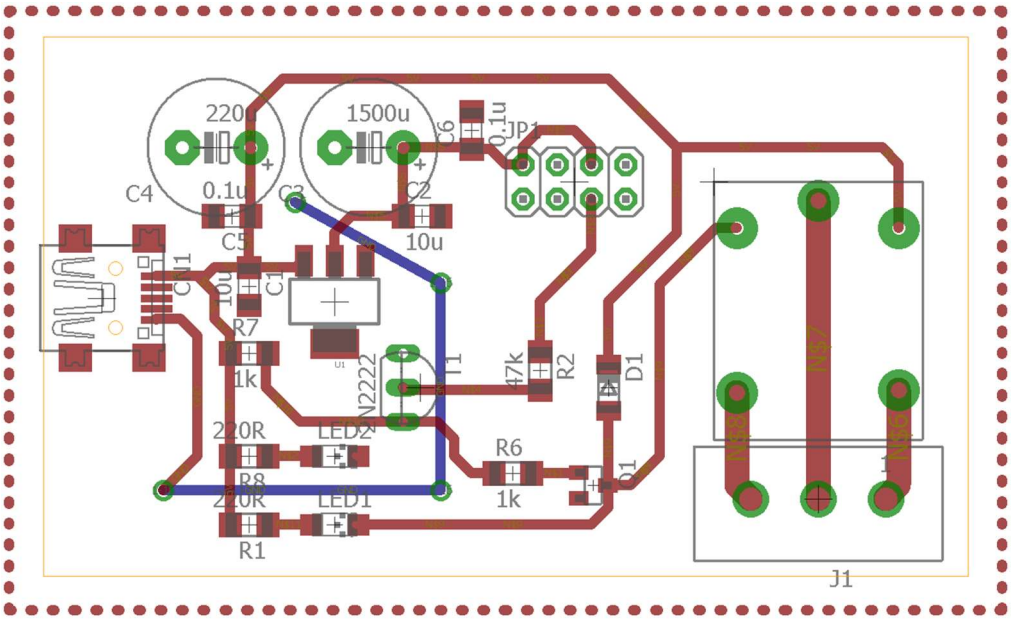

Figure 12 PCB of the relay unit

The printed PCB with the ironed parts can be seen on the next figures:

Figure 13 The top of the PCB of the relay unit

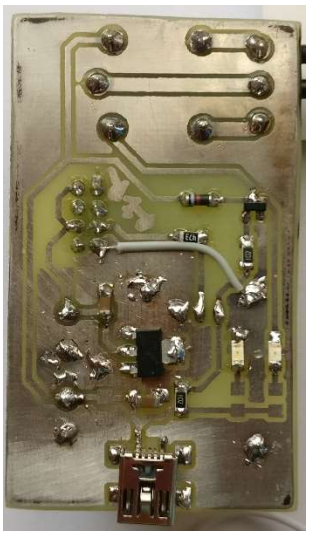

Figure 14 The bottom of the PCB of the relay unit

The relay itself is a SRD-05VDC-SL-C device. This device can switch a maximum of 10A current with 5V input voltage. The switching logic is realized with a 2N2222 (left on **Figure 11**) low power bipolar npn transistor and a BC817 (right on **Figure 11**) general purpose bipolar npn transistor. When the output of the ESP-01 is low, the 2N2222 is off, the current from the 5V supply flows to BC817, opening it and thus switching the relay on. When the output of the ESP-01 is high, the 2N2222 is on, so the current from the supply does not flow to the BC817, and thus the relay is switched off. To ensure that these two transistors are suitable for this application, calculations were made.

The DC current gain of the 2N2222 can be anywhere between 35 and 70. The calculation are made for the worst case scenario:

$$\frac{3.3[V]}{47[kOhm]} = 0.0702[mA] \quad (1.)$$

$$35 \cdot 0.0702[mA] = 2.46[mA] \quad (2.)$$

$$\frac{5[V]}{2 \cdot 1[kOhm]} = 2.5[mA] \quad (3.)$$

If the 2N2222 is switched on by the 3.3V output of the ESP-01, a minimum of 98% of the current from the supply can flow through the 2N2222 (this is the worst case scenario), so practically no current can flow to the BC817.

The BC817 has a minimum DC current gain of 100, and has 2.5mA base current if the 2N2222 is off, so

$$100 \cdot 2.5[mA] = 250[mA] \quad (4.)$$

current can flow through the relay, which is enough, because the relay requires a maximum of 100mA current.

4.3 Software Design

The software design can be separated into two parts as the communication level and the function level. As the sensor and relay units are only sending data and receiving commands, the real function level is only in the phone application.

4.3.1 Communication design

The communication between the devices is through TCP/IP, because it ensures no packet loss, therefore, more precise operation.

As the Android program should work with any devices, the phone should make the connection requests. So the steps of the basic communication can be seen on **Figure 15**.

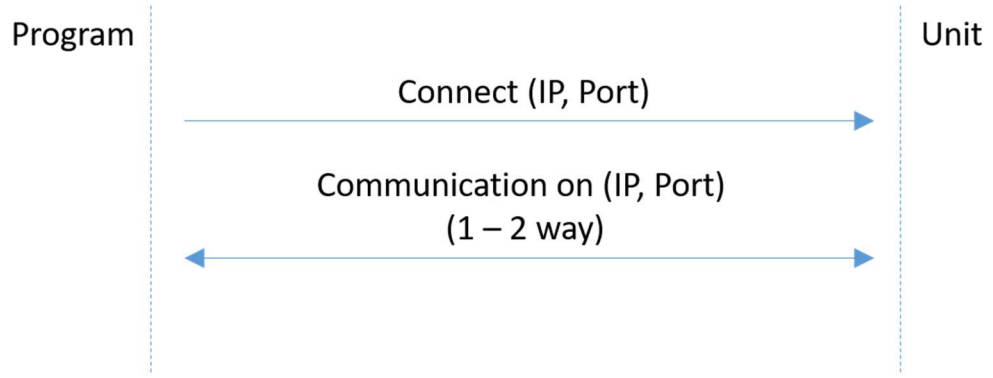

Figure 15 Communication between program and units

It can be seen, that first the phone connects to the unit, then the whole communication can start. As stated before, the sensor only sends data, therefore, their communication is as follows:

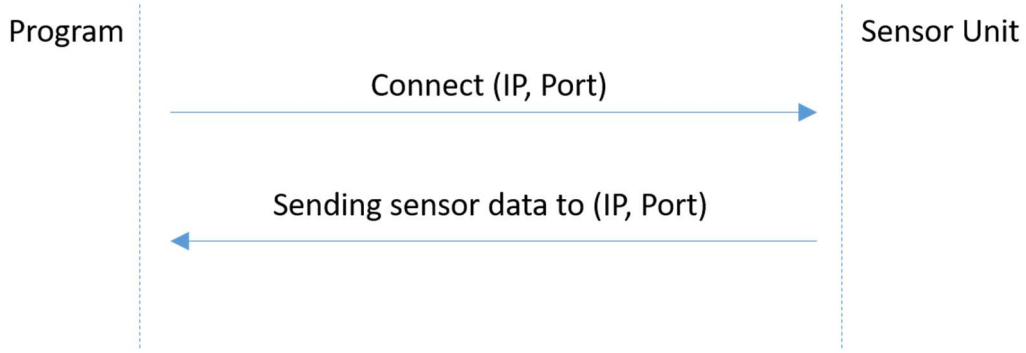

Figure 16 Sensor unit communication

The relay module communication is a slightly more complicated, as it should receive a command, and then reply to it. Its communication can be seen on **Figure 17**.

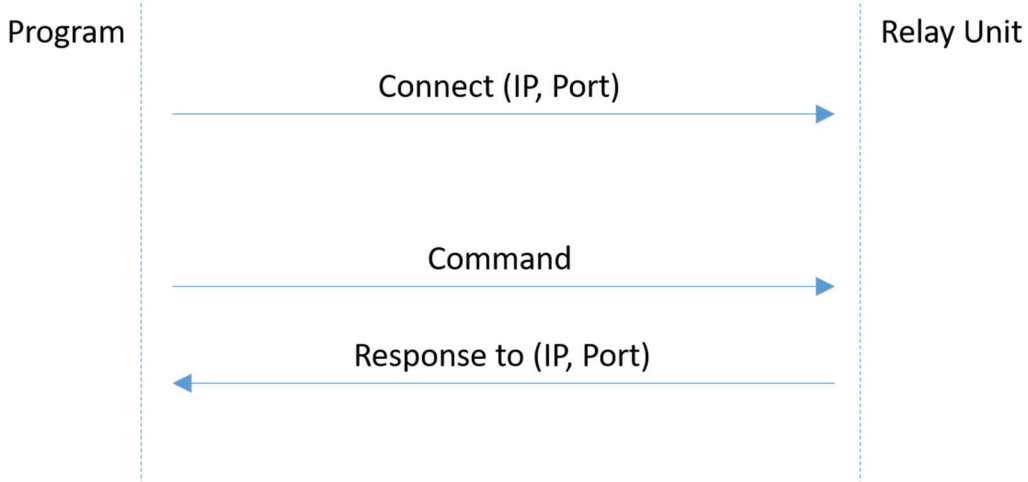

Figure 17 Relay unit communication

4.3.2 Function design

The main idea is to make an event based smart home system. It is implemented with programs that has an actuator and conditions, as shown on **Figure 18**. The condition can be the pushing of a button, reaching a certain time or a value of a sensor. If one condition is met, a certain command is sent to the actuator to set its state to the desired one.

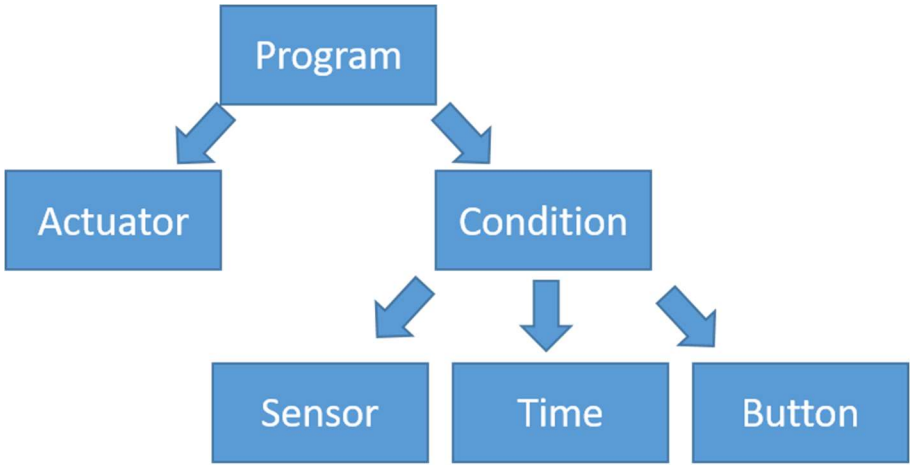

Figure 18 Software functionality architecture

More programs can be created, and multiple conditions can be added. Each condition has a desired state for the actuator, so if it is met, the actuator is triggered.

To make the application more dynamic, the states of the actuators are stored on the unit, so in case of introducing new actuators, only the code of the actuators should be written.

4.3.3 User interface

The application has an actuator screen (**Figure 19**), where the actuators can be switch on or off manually. It has a program screen (**Figure 20**), where programs can be added, deleted, edited or started.

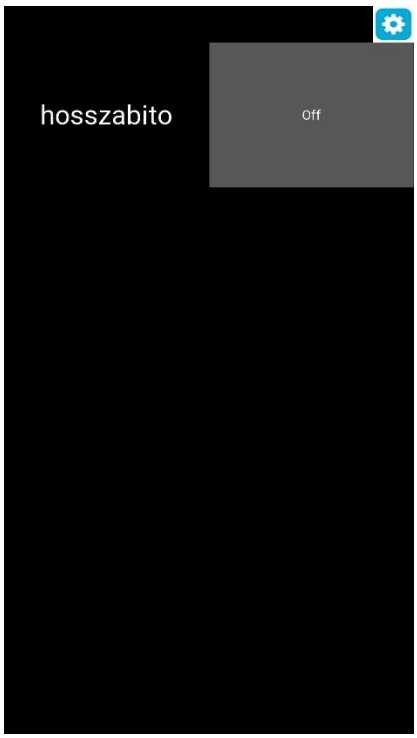

Figure 19 Actuator screen

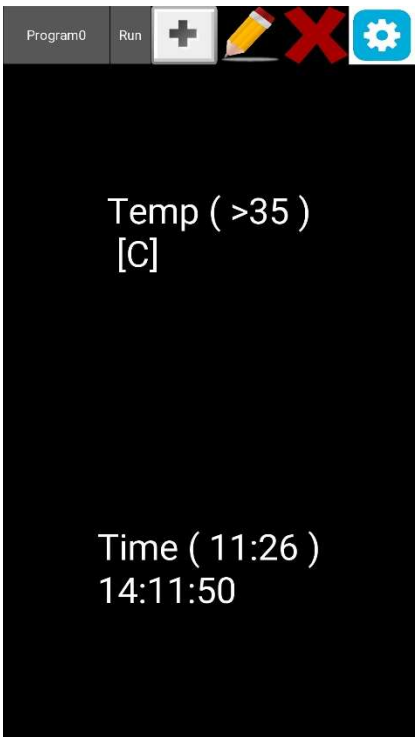

Figure 20 Program screen

Clicking on the edit button (Pencil icon), the condition screen (**Figure 21**) appears, where conditions can be added, deleted or edited. From the condition screen, by clicking on the "Add" button, the condition adding screen appears (**Figure 22** and **Figure 23**), where conditions based on sensor data or time can be added.

Figure 21 Condition screen

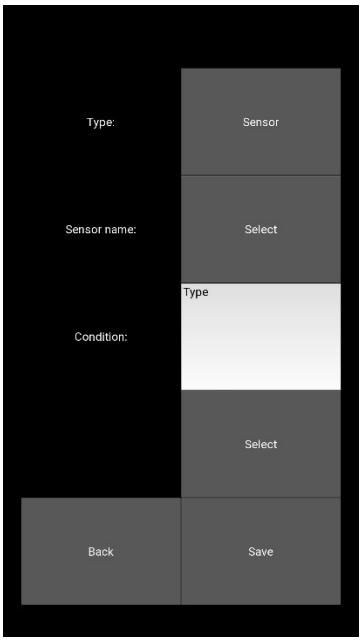

Figure 22 Add condition based on sensor data

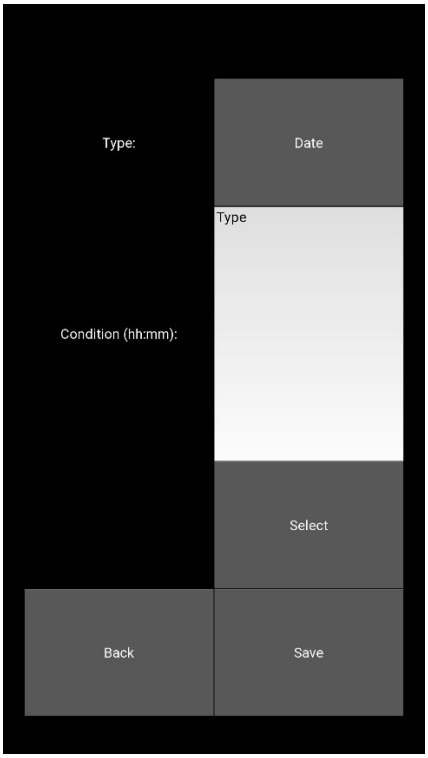

Figure 23 Add condition based on time

5 Testing

The testing configuration consists of the sensor unit (**Figure 24**), the relay unit with the cost efficient security housing (**Figure 25**), the coffee machine and the android phone.

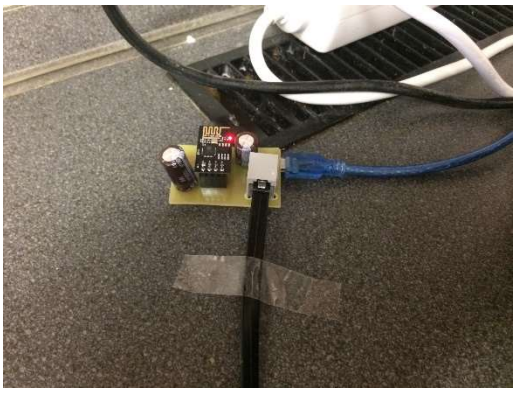

Figure 24 Sensor unit

Figure 25 Relay unit

The temperature sensor is secured to coffee carafe as can be seen on **Figure 26**.

Figure 26 The temperature sensor attached to the coffee carafe

The whole configuration can be seen on the next figure:

Figure 27 Event based coffee machine controlling system configuration

As the temperature sensor is attached to the coffee carafe that holds the warm coffee, the heat transfer needs some time to happen. Its characteristic can be measured. As the current usage doesn't require high accuracy, it can be measure easily, by recording a video, where the coffee level, and the temperature can be seen at the same time. During the test, for cost efficiency reasons, water has been used instead of coffee. The characteristic of the heat transfer can be seen on the following diagram:

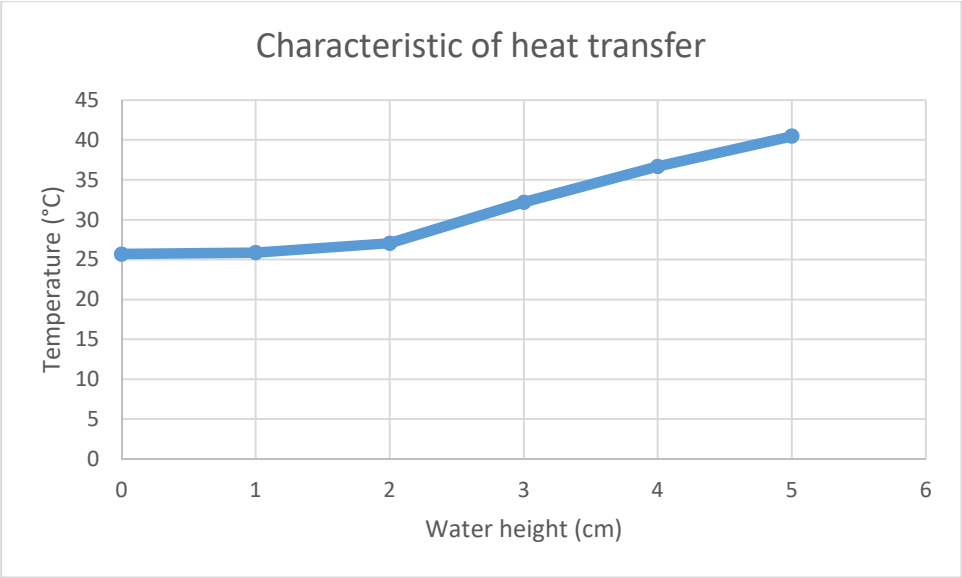

Diagram 1 The characteristic of heat transfer diagram

Therefore, setting the condition to 35°C is appropriate to turn off the coffee machine, as the rest of the coffee will come out anyway because the entire water is already in steam state.

6 Summary

The created system is working as expected and could be extended for further use. The current configuration is capable of switching the relay based on time, button interaction or sensor values. It means that e.g. a thermostat can be created with a program in the Android software that works with temperature hysteresis.

The coffee machine can be turned on with a button, or turn on at a certain time and it turns of when the coffee is done.

It worth mentioning, that the warming speed of the DS18B20 sensor is slow. It can be measured by attaching the sensor to a warm surface and waiting it to warm up.

6.1 Further development possibilities

The following developments could be made in the future:

- Adding more sensors and actuators.
- Handle complicated expressions of conditions in the application.
- Change the DS18B20 to another faster temperature sensor.
- Create protective housing to the PCBs and redesign them according to it.

Bibliography

[1] https://www.networkworld.com/article/3199671/internet-of-things/what-is-industry-4-0.html

[2] WiFi-IoT Beta, https://wifi-iot.com/

[3] Controllium, https://play.google.com/store/apps/details?id=com.daedalusstone.sigma.controllium

[4] https://en.wikipedia.org/wiki/ESP8266

[5] http://electronut.in/an-iot-project-with-esp8266/

[6] ESP Wifi relay module, https://www.hestore.hu/prod_10037959.html

[7] Arduino open-source electronics platform, https://www.arduino.cc/en/Guide/Introduction

[8] http://1.bp.blogspot.com/-2zMMrsmzNVk/VzkpezD6j2I/AAAAAAAAFNE/VU5DisFVRMMaz1S0S7R-j8CTe0j_ot8sACK4B/s1600/Circuito-ESP8266-ESP-01-FTDI-FT232.png

[9] Adding new board manager, https://create.arduino.cc/projecthub/ROBINTHOMAS/programming-esp8266-esp-01-with-arduino-011389

[10] Python, https://docs.python.org/3/faq/general.html#what-is-python

[11] Kivy, https://kivy.org/#home

[12] https://iot-playground.com/blog/2-uncategorised/41-esp8266-ds18b20-temperature-sensor-arduino-ide

[13] PCB Ironing, http://www.instructables.com/id/Making-A-Customized-Circuit-Board-Made-Easy/

[14] ESP01 wifi relay, https://iot-playground.com/blog/2-uncategorised/40-esp8266-wifi-relay-switch-arduino-ide