Automated Batch Reactor with a Raspberry PI

Owner:

Köllmann, Matthias

Essinger, Timothy

Table of Contents

Table of Figures

Introduction

Today the topic of automation is quite common in the process industry. Due to the fact that with automation the process can be better controlled, downtimes reduced and the product quality can be increased. This task can be done by expensive equipment from different providers. In order to reduce the cost of this equipment the basic automation task could be done by a microcomputer. In this case a Raspberry Pi is used. This microcomputer provides a high flexibility for different tasks. Therefore, it can be used for batch or for continuous processes in different industries. Of course, the Raspberry Pi is not able to meet the safety standards of the core process in the process industry. But for pilot-plant scale this could be a sufficient solution. The task was to build a small process plant with one sensor and one actuator.

Basic concept

The main idea is to automate a batch reactor. In the reactor should be a certain pH-value and a specified temperature. The thresholds for both values have to be put in by the operator. To achieve the specified temperature a heater will be used. For the pH-value will be a vinegar solution used. This will be given into the system from a different vessel with a small pump. To have a better understanding of the concept the following process diagram is introduced.

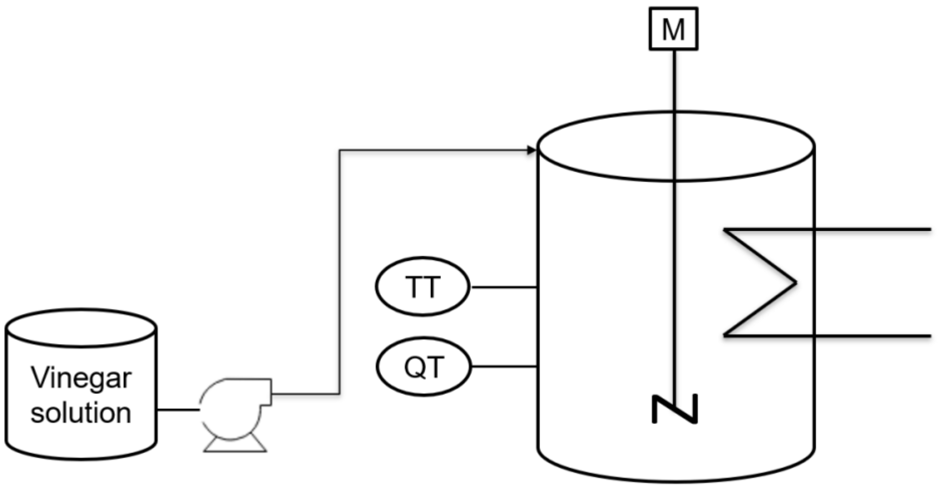


Figure 1: Basic process design

Wiring

Before the Raspberry Pi can be deployed it has to be set up. This can be achieved by installing a operating system on the SD card. After this the Raspberry Pi able to boot and provide a user interface. Subsequently, wiring must be performed. The components to connect are the remote module, the temperature sensor, the pH-sensor and the analogue to digital converter (ADC). The importance of this will be discussed in the outlined section. The different components have also different voltage needs. The temperature sensor needs 3 V power supply. All the others need 5 V. Though the Raspberry Pi has limited pins for the power supply. In order to solve this problem, the usage of an external distributor is performed. By now one 5 V pin from the Pi is able to supply more than one electrical load. The wiring of the remote can be taken from the following figure.

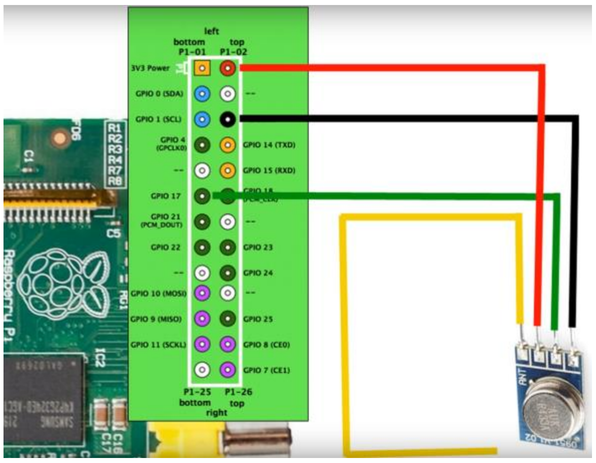


Figure 2: Wiring scheme for the remote module [1]

The wiring of the temperature sensor can be taken from the following figure. It is important to use the right resistance which can be selected by the colour rings around it.

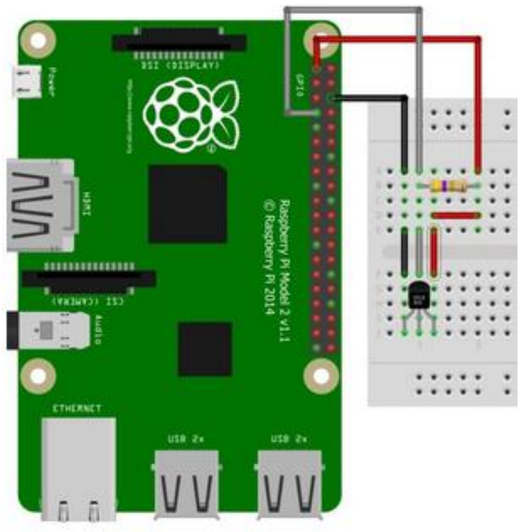


Figure 3: Wiring Temperature sensor [2]

The connection of the pH-senor is a more difficult. The pH-sensor has to be connected to the ADC which is connected to the Pi. To have a sufficient connection at the Pi the contacts have to be soldered.

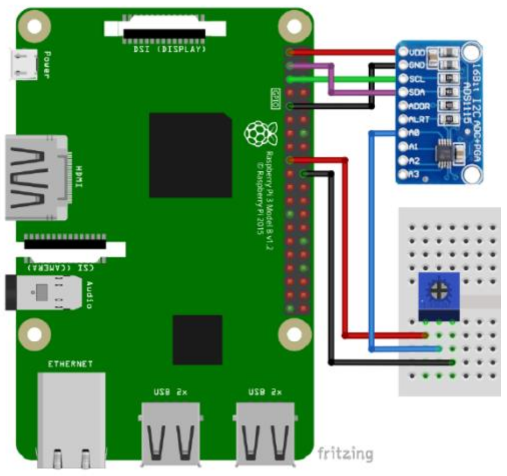


Figure 4: Wiring scheme of the ADC [3]

In the schematic above, the wiring of the ADC to the Raspberry Pi is shown, instead of the used pH sensor, there is an exemplary potentiometer in place instead. In order for the Raspberry Pi to recognize the signals on Pin 3 & 5 as a serial bus instead of just a Boolean state, a bus protocol has to be used. In this case, the Inter-Integrated Circuit (I²C) communication bus standard is used. First of all, only selected pins of the Raspberry Pi GPIOs do support the usage of this bus protocol and thus have to be wired correctly. Afterwards the communication protocol I²C has to be activated in the Raspberry Pi's OS.

Challenges

During the project, many challenges arose which had to be addressed accordingly. In this subsection the problem and the proper solution will be presented. At the first beginning the remote control of the plugs was a big problem. It was quite easy to find some lines of code that turn the plugs on form the command window of the Pi. The challenge was to achieve the same result with a python code. With the following code the problem could be solved. It can be seen that the python code just opens the command window and gives the instruction there.

```
#plug for the heater
os.system("cd /home/pi/raspberry-remote; ./send 10101 1 0")
heater_on=False
#plug for the stirrer
os.system("cd /home/pi/raspberry-remote; ./send 10101 2 0")
stirrer_on=False
#plug for the pump
os.system("cd /home/pi/raspberry-remote; ./send 10101 3 0 ")
pump_on=False
```

Figure 5: Remote control

Furthermore, the plugs are secured with a 5-bit encryption. This type of basic encryption denies unwanted access from unauthorized parties. The level of protection is sufficient for use at home or for a low-risk laboratory experiment. For the usage in industry, this kind of encryption would not be sufficient as the number of possible codes is very limited and thus easy to break. Even basic coordinated attacks, such as brute force attacks, have a high chance to break through this kind of encryption. The hardware switches for the 5-bit encryption can be seen on the figure below.

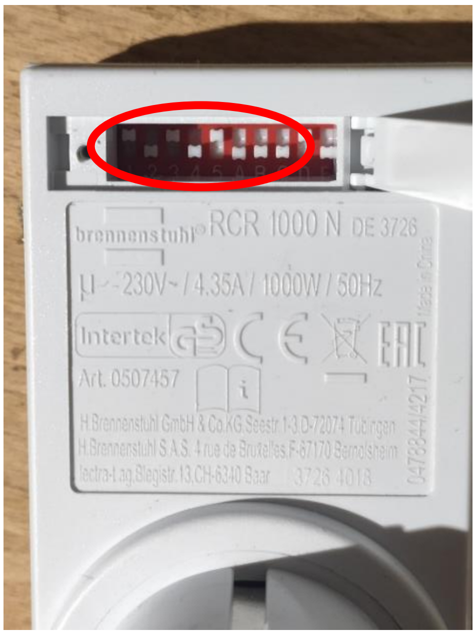

brennenstuhl RCR 1000 N DE 3726
230V~ / 4.35A / 1000W / 50Hz
Intertek
Art. 0507457
H.Brennenstuhl GmbH & Co.KG Seestr.1-3 D-72074 Tübingen
H.Brennenstuhl S.A.S. 4 rue de Bruxelles F-67170 Bernolsheim
lectra-Lag Blegistr.13 CH-6340 Baar
3726 1018
Made in China
04738444217

Figure 6: 5-bit encryption

Another challenge to face was the pH-sensor. The pH-sensor is limited to analogue signal transmission. Therefore, an Analogue to Digital Converter (ADC) has to be used. As the name indicates, it is used to convert the analogue signal to a digital one that can be understood by the Raspberry Pi, as the Raspberry Pi, opposed to its competitor Arduino, does not have an integrated ADC and is therefore not compatible with any analogue sensor without an external ADC. The signal that the Pi obtains are just numbers that has to be brought into the right context. For example, a value from the pH-sensor could be 10,000. For the translation of the arbitrary number into a real pH-value a mathematical function is needed. Normally, a pH-value should between 1 up to 14. It becomes clear that some reverence measurements have to be performed. With calibrate substance some values from the sensor could be allocated to a pH-value. From this a regression line can be drawn.

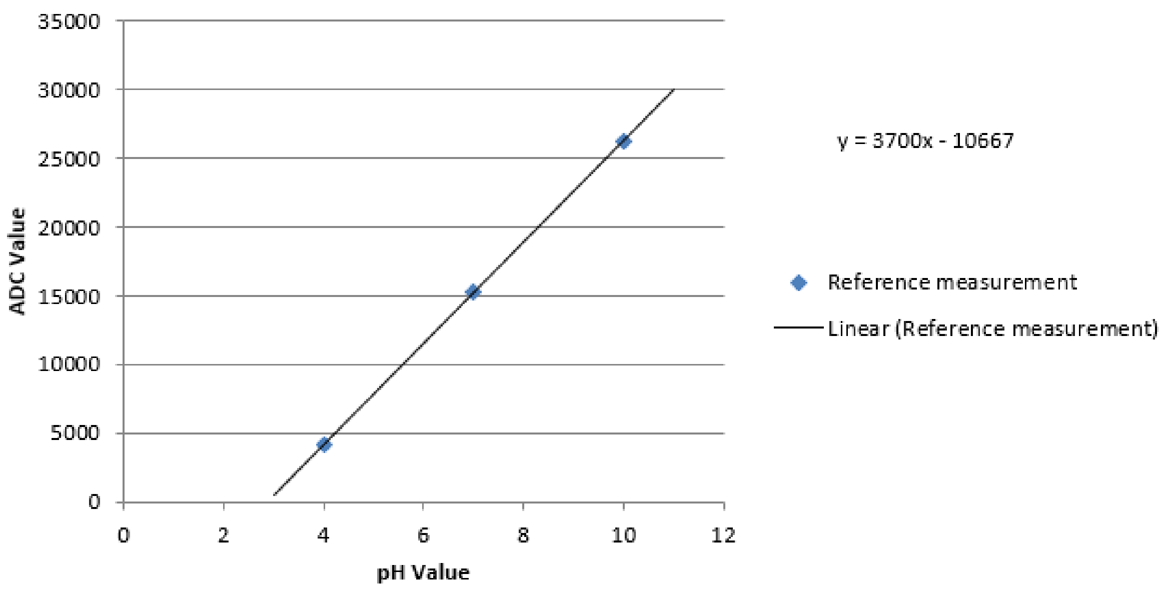


Figure 7: pH-Value Reference Measurement

The biggest problem was the fluctuating values from the pH-sensor. The shown values variates from the upper to the lower limit in random orders. The first idea was that the pH-sensor is broken. But the sensor worked well in all other vessels beside the one it will be used. Therefore, the problem could be restricted to this vessel. The second idea was that there is something wrong with the power supply of the ADC and the pH-sensor. Additionally, there could be something gone wrong by setting up the gain of the ADC. This has to be defined by the operator. After all values were checked in a proper way the sensor behaved exactly the same. The last idea was that it could have something to do with inferences between the measurement and other electrical components. Basically, a creeping current may have occurred between the electrical sensors and the ground wire of the plugs. The problem occurs because there is no insulation between the sensors, the medium and the metal electrical devices connected to the plug. Therefore, an additional ground pole is provided. This hypothesis was developed due to the fact that in the other vessels without the stirrer the pH-sensor worked fine. To decouple the system a plastic stirrer was printed out with a 3D printer. This is shown in the figure below.

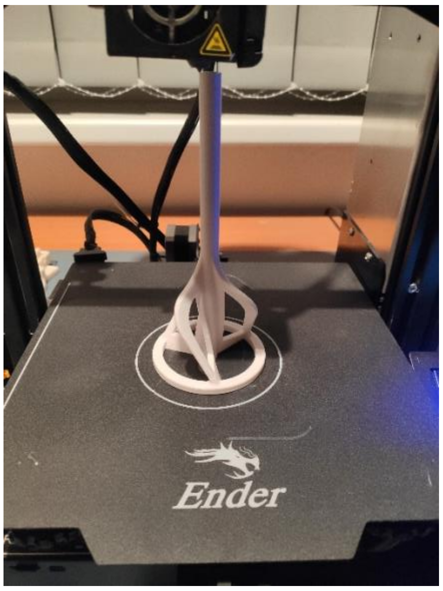


Figure 8: 3D printed stirrer

Furthermore, the heater was initially planned to be an immersion heater. This would lead to the same problem and the concept had to be changed. Consequently, the system is now heated up by a stove. To prevent conductive behaviour clay is put between the stove and the vessel. This will of course lower the velocity of the heat transfer, but it is a safe method to avoid interferences and therefore sufficient.

The temperature sensor disturbs the signal and has to be isolated. This is done with a test tube that is filled with water, acting as an improvised measuring sleeve. Because of the small amount of water and the small wall thickness the thermal inertia is neglectable. It is shown in the following figure.

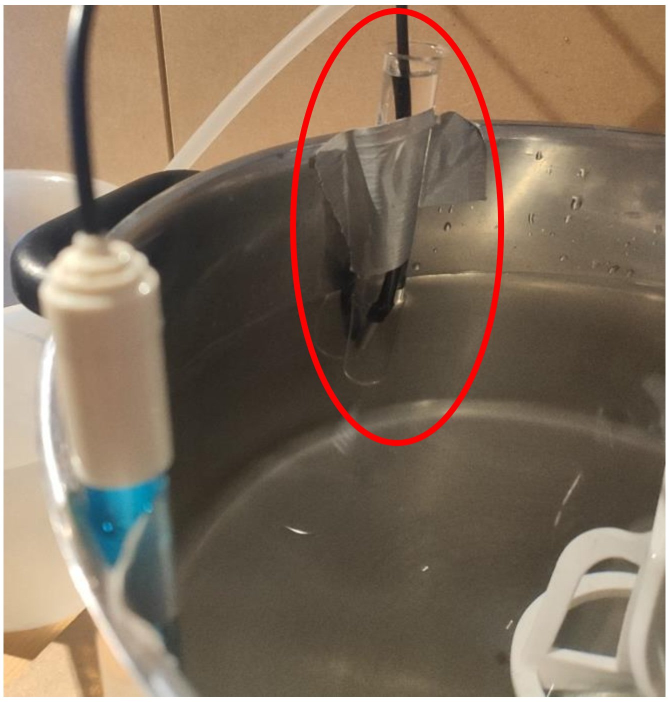


Figure 9: Improvised measuring sleeve

Final process design

The final process design can be taken from the following figure.

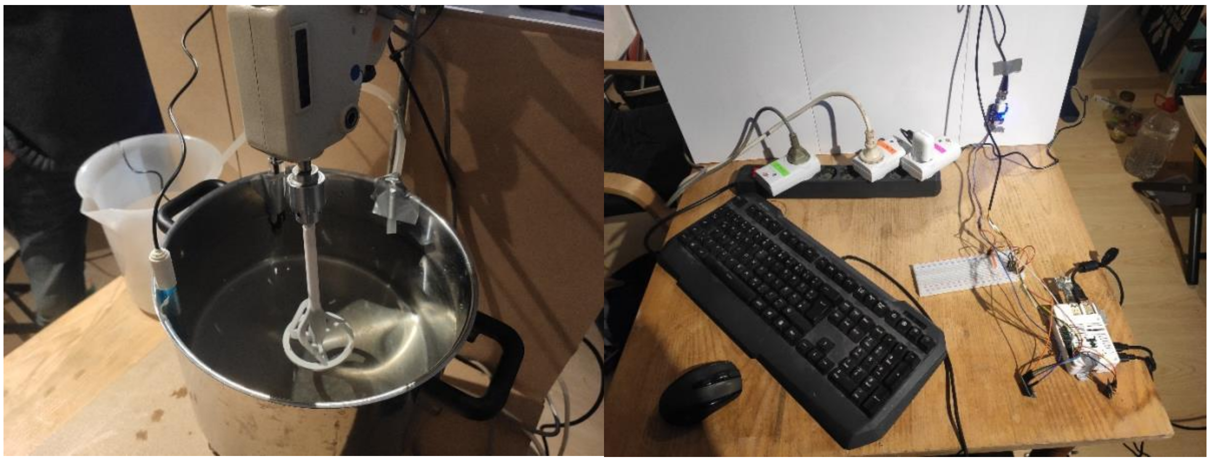
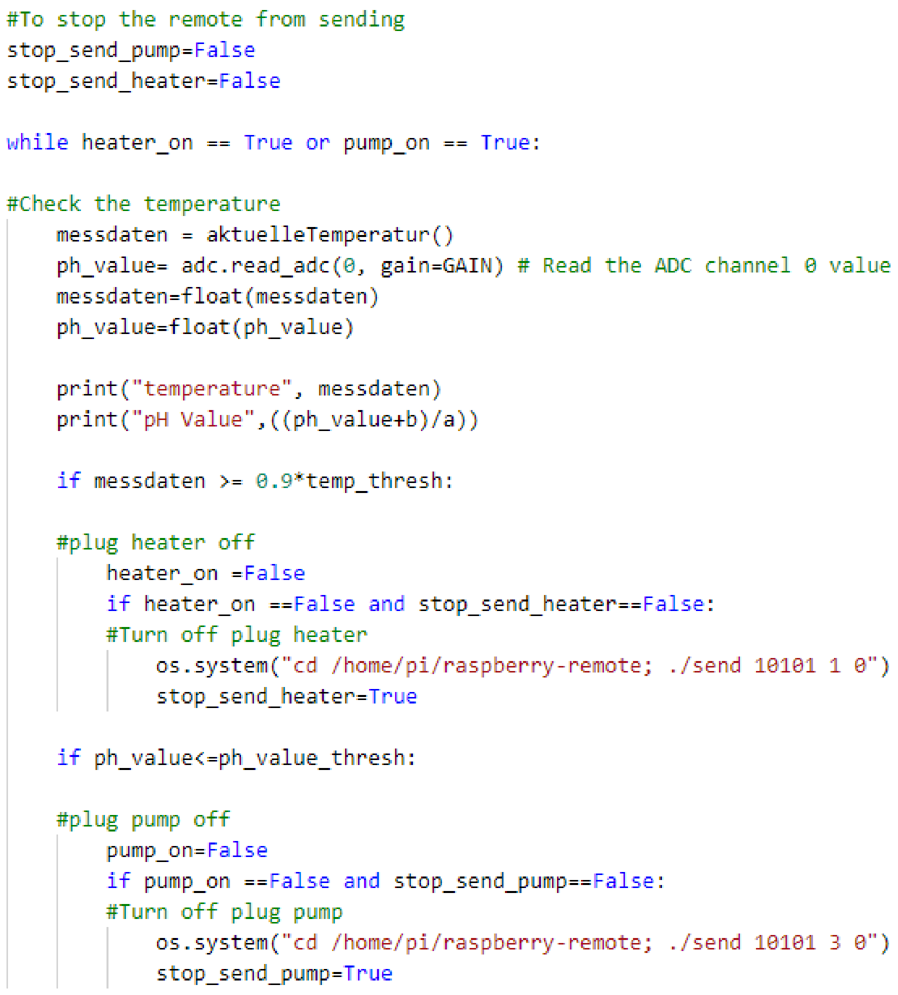


Figure 10:Final Process Design

There were also some lines of codes developed to control the system. These are shown in the following figure. The comments can be used to understand the code. At this moment, some code lines should be familiar. For example, the lines that turns the plugs on and off.


```
#To stop the remote from sending
stop_send_pump=False
stop_send_heater=False

while heater_on == True or pump_on == True:

#Check the temperature
    messdaten = aktuelleTemperatur()
    ph_value= adc.read_adc(0, gain=GAIN) # Read the ADC channel 0 value
    messdaten=float(messdaten)
    ph_value=float(ph_value)

    print("temperature", messdaten)
    print("pH Value",((ph_value+b)/a))

    if messdaten >= 0.9*temp_thresh:

    #plug heater off
        heater_on =False
        if heater_on ==False and stop_send_heater==False:
        #Turn off plug heater
            os.system("cd /home/pi/raspberry-remote; ./send 10101 1 0")
            stop_send_heater=True

    if ph_value<=ph_value_thresh:

    #plug pump off
        pump_on=False
        if pump_on ==False and stop_send_pump==False:
        #Turn off plug pump
            os.system("cd /home/pi/raspberry-remote; ./send 10101 3 0")
            stop_send_pump=True
```
Figure 11: Main Loop

Basically, there is one big loop in which every iteration steps the temperature, and the pH-value is checked. If they are beyond the threshold the plug for this component obtains the order to turn off. At the beginning of the code there are just some lines so the operator can set the thresholds. The program then performs a quick plausibility check on the input values, so no unsafe temperatures or unreachable pH values can be set as thresholds. This prevents both damage to the measuring equipment as well as helping to prevent danger due to an infinite amount of heating required which in turn could lead to the evaporation of the water or even fire in case the heater would never turn off. Also, the pH-value has to be transformed into a value that the sensor can be compared with. This will be converted back at the end of the process.

The second loop is necessary because of the thermal inertia. The system will set off the plug for the heater after it reaches 90 % of the threshold.

After both the pH and the temperature thresholds have been reached, the program sends one last shut down command to each and every remote plug, in order to return the system to its safe state even if something went wrong during the iterations before.

Furthermore, a physical separation of the electric part and the water-bearing part was performed. A barrier was put in between the different areas in order to provide additional protection in case of water spillages and thus help prevent short circuits in case of any unforeseen incidents. To further protect the electronics, a protective case for the Raspberry Pi was 3D-printed in order to protect it from mechanical hazards as well as possible water spillages.

Results

In the experiment it was shown that a microcomputer like a Raspberry Pi can be utilised to realise a flexible, low-cost automation for a simple batch reactor. Obviously, the kind of measurement and the kind of actuators may vary depending on the needs of the process, but special attention has to be paid at the selection of sensors in order to minimise the negative influence of one another. This problem is not specific to the used Raspberry Pi setup though and is a point to be considered anyway. Furthermore, it was shown that for simple, small scale and low risk experiments, the usage of 433 MHz remote plugs is a quick and easy way to implement an automation of different actuators which are also protected against unauthorized access on a basic level.

References

[1] SemperVideo, "*Raspberry Pi: Steckdosen fernsteuern"* https://www.youtube.com/watch?v=ZOgJtNaJZ_4 (uploaded at 06.06.2014, last visited at 15.01.2021).

[2] http://webnist.de/temperatur-sensor-ds1820-am-raspberry-pi-mit-python/ (written at 07.10.2015, last visited at 28.02.2021).

[3] AZ-Delivery, *ADS1115 Manual*