HSB
Hochschule Bremen
City University of Applied Sciences

DOCUMENTATION

TeaCooker

MODUL:

Microcontroller and Applications

SUPERVISOR:

Christoph Flores
Volker Schmitt

STUDENT:

Moumita Ahmad, *5027729*

Table of Content

1 Introduction **3**

2 Hardware **5**
- 2.1 Sequence diagram 5
- 2.2 Block diagram 5
- 2.3 Circuit diagram 6

3 Software **8**
- 3.1 Difficulties 8
  - 3.1.1 inaccurate color sensor 8
  - 3.1.2 LCD and interrupts 8
  - 3.1.3 buzzer 8

4 Reflection and Summary **9**

5 Attachment **10**

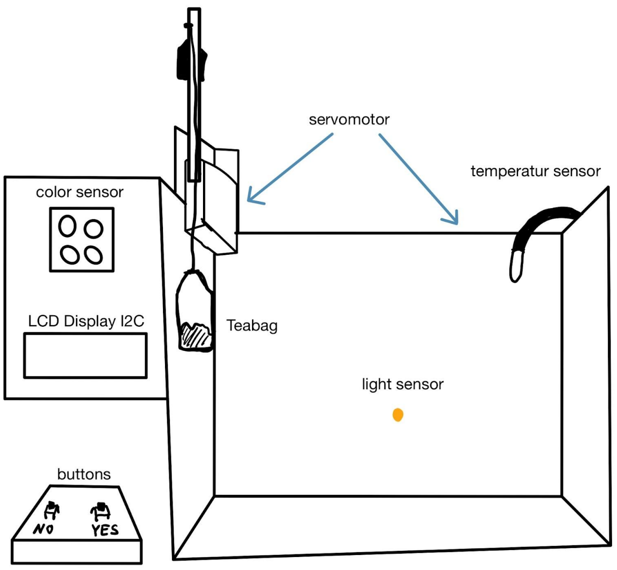
1 Introduction

This part gives a brief overview of all the changes made from the project's specifications, submitted at the beginning of the semester.
The following list shows all requirements as defined in the project's specifications:

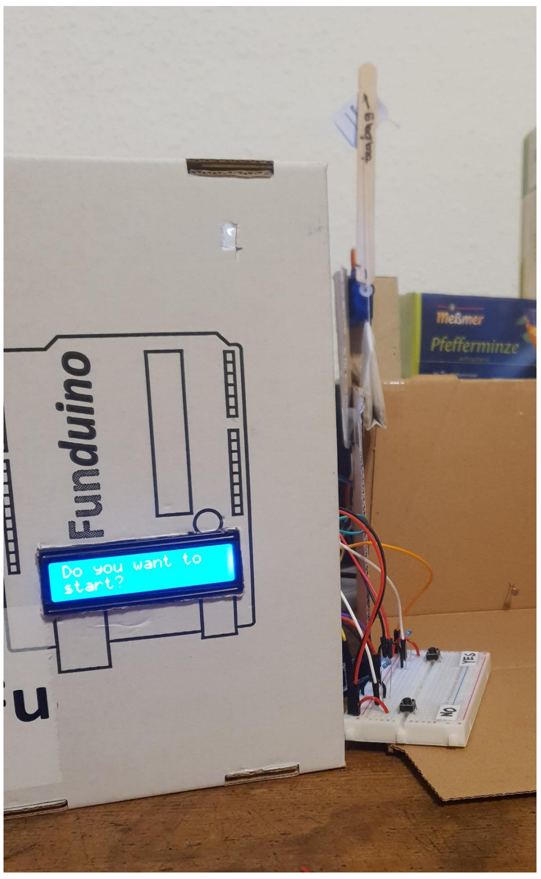
- choose a type of tea (high)
- detect the type of tea (medium)
- detect if a cup is placed (medium)
- read out the tea specifications (high)
- move the teabag (high)
- measure the water temperature (high)
- display the temperature (high)
- block and unblock Waterflow (high)
- notify the user about the finished tea (high)

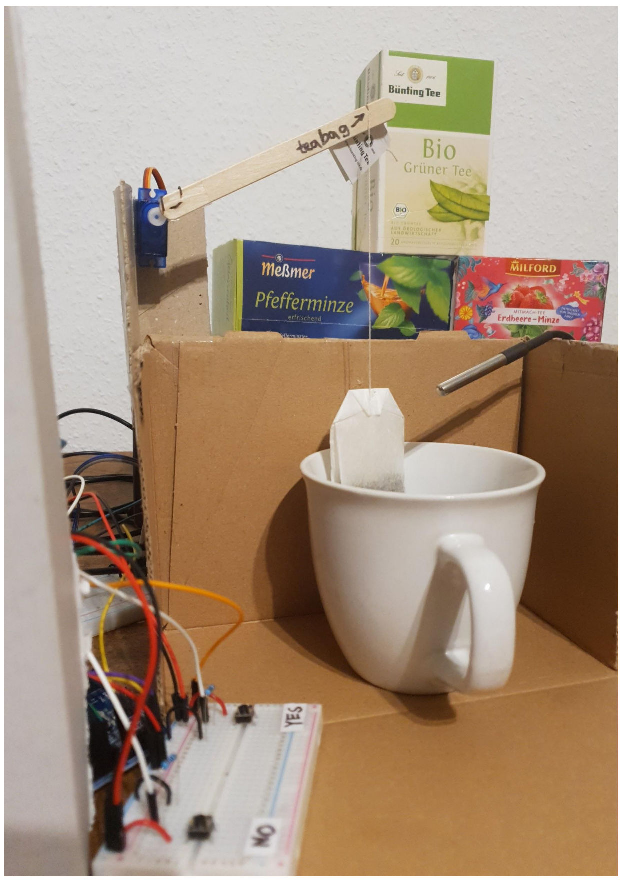
While most of the requirements and core functionalities stayed the same during the project and are now represented in the end-product, some changes needed to be made during the development process.
One obstacle was the transportation of hot water from an attached tank to the cup. To solve the problem it was decided to remove this part of the project and have the user fill up the cup by themself.
Another change was made in the transportation of the teabag.
Originally, the idea was to use a slider and a winch to transport the teabag in and from the cup. But after creating a prototype that used a stepper motor, some cord, and a rubber band this method did not seem to be practical for the desired purpose. The transportation was quite slow and while using a different motor would have helped, with the hardware available it seemed to be more practical to implement a fan-like motion instead. Therefore the alternative for the slider described in *chapter 4.1 risks and alternatives* was used. The same construction was also later used to transport the temperature sensor in and out of the cup.
And lastly, also the cardboard construction changed a bit, but all the functionalities were kept.

Therefore in the end-product had the following concept:



Which turned into this:

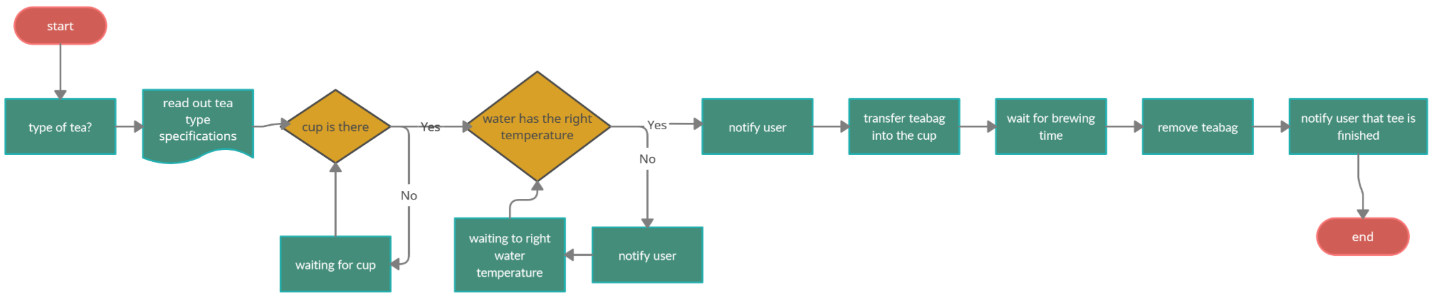




2 Hardware

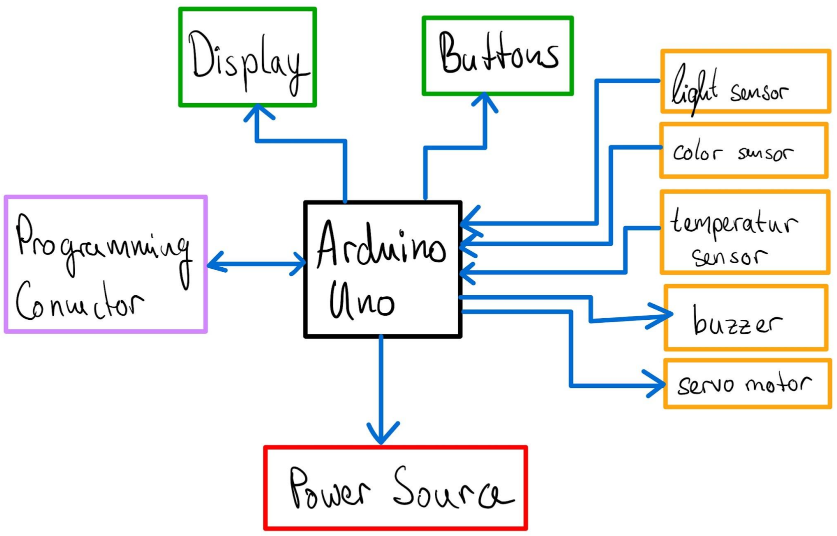
2.1 Sequence diagram

The Sequence diagram changed a bit from the one in the project's specification, therefore the updated version is displayed below:


start
type of tea?
read out tea type specifications
cup is there
Yes
No
waiting for cup
water has the right temperature
Yes
No
waiting to right water temperature
notify user
notify user
transfer teabag into the cup
wait for brewing time
remove teabag
notify user that tee is finished
end

2.2 Block diagram

The following block diagram shows an overview of all the hardware components used for the project:

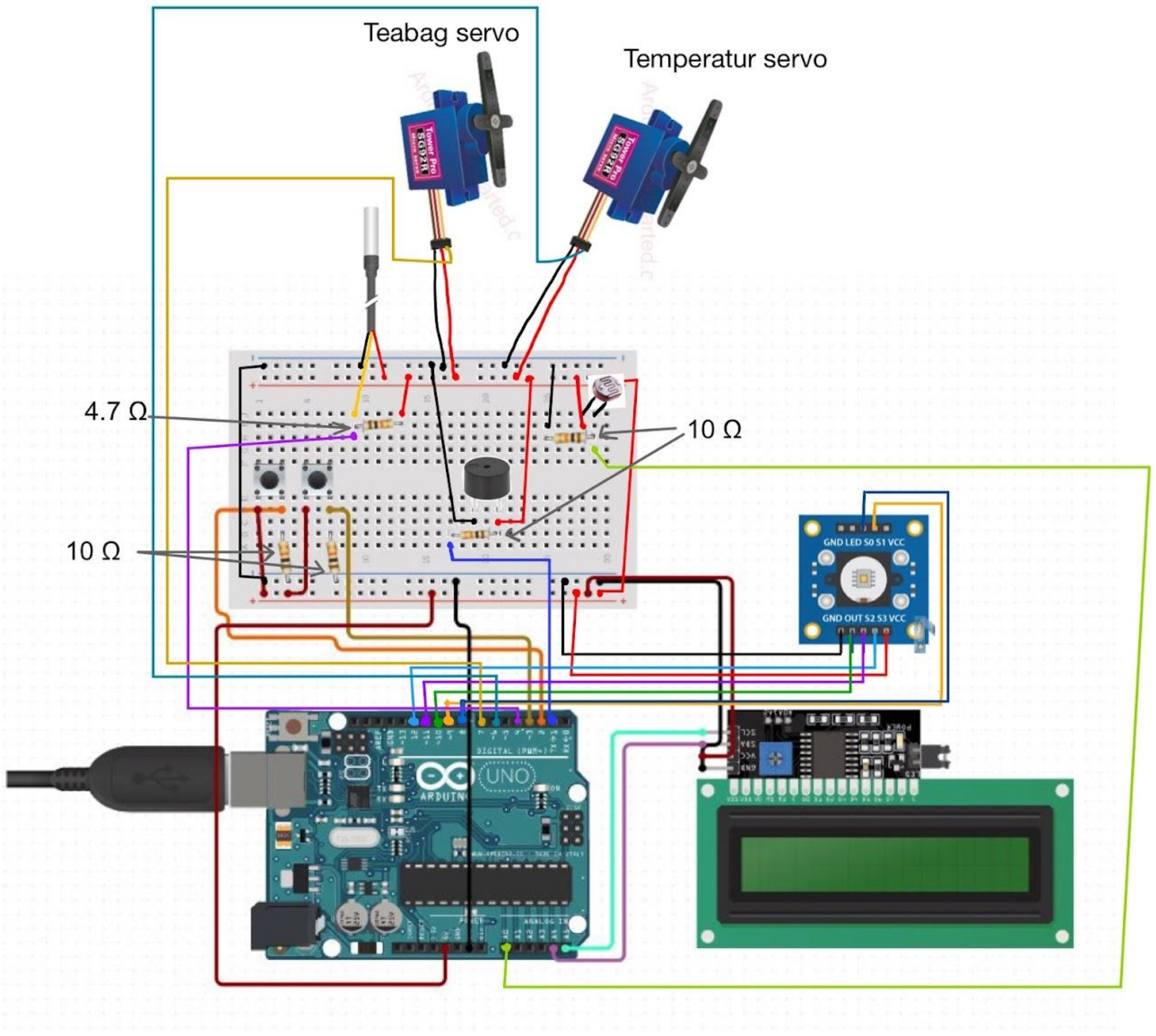


To implement the application the following components:
(the abbreviation are later used in the pin table)

- Arduino Uno
- LCD Display 16x2 I2C (LCD)
- two Mini Pushbuttons (RB, LB))
- TCS230 Color RGB Sensor (CS)
- Photoresistor LDR Light Sensor (LS)
- DS18B20 1-Wire Temperature Sensor - Waterproof (TS)
- active buzzer (BZ)
- two 9g Micro Servo motors (TMS, BMS)

2.3 Circuit diagram

The next diagram shows the hardware setup and all the pin connections. The table afterward displays the pin connections again more clearly.

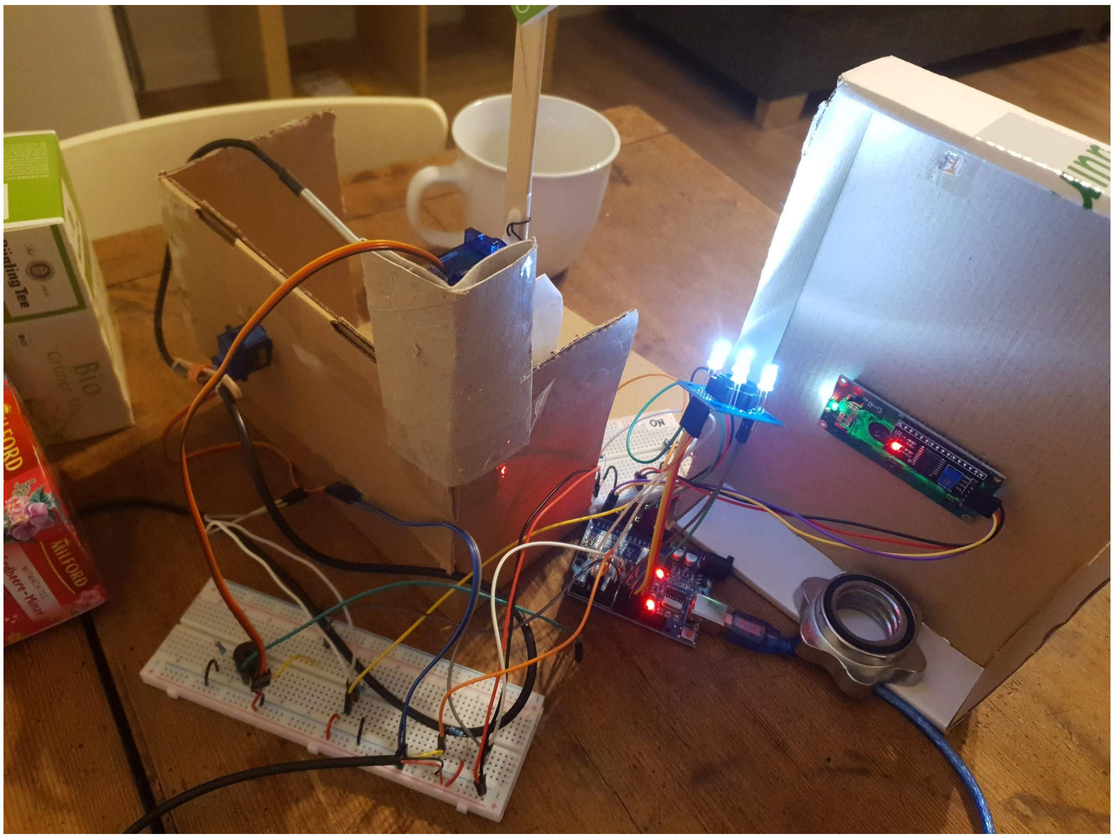


The pin connections can also be seen in the following table:

Pins on the Adruino Uno	**Connected Pin**
D1	BZ
D2	LB
D3	RB
D4	TS
D6	TMS
D7	BMS
D8	CS - S0
D9	CS - S1
D10	CS - OUT
D11	CS - S2
D12	CS - S3
A0	LS - Data
A4	LCD - SCL
A5	LCD - SDA

* *(the abbreviation are described on page 5)*

After everything was connected the following picture shows what the behind the scenes looked like for me:



3 Software

To realize all functionalities a sketch was written for the Arduino Uno.
The code can be viewed here: *https://github.com/moumitahmad/TeaCooker*.

3.1 Difficulties

3.1.1 inaccurate color sensor

The color sensor is detecting the color from a large area which sometimes leads to inaccurate color detections. So to still get the right tea type the user now needs to confirm the tea detected by the application. Also reducing the range where a certain color is identified helped to make more accurate detection. The only disadvantage is that the scan time is increased which could be frustrating for the user.

3.1.2 LCD and interrupts

One focus in the development was to make the project understandable for first-time users, therefore the class *userFeedback* was written to give the user constant feedback about the current step. A requirement for this class was to display any text to the user and also be able to handle a ‘yes’ or ‘no’ answer from the user. So the first idea was to scroll the displayed text to adjust to any length and have an interrupt to react to a button click. But the *New-LiquidCrystal* or the *Wire* library, used for the display, seemed to be in conflict with the build-in interrupt function so that the entire program stops when the interrupt was triggered. This was solved by implementing a polling approach instead.

3.1.3 buzzer

Another idea was to include sound in the project by using an active or passive buzzer.
But while implementing a simple melody worked fine in a separate sketch (page 10), including it into the main project turned out to be difficult and the buzzer either made a sound constantly or not at all.

4 Reflection and Summary

Although the project works and the whole process of brewing tea can be completed, if the application would be used long term, a few improvements could help.
First, the project could be designed a bit more sturdy and permanently.
Due to trying to keep the cost of this project low cardboard was the main material for the frame and also most components were not soldered together to make them easier reusable for later projects. But when using the project actively something more durable would be needed. Also, some waterproof material for the frame would be recommended.
Second, the previously planed water tank could be another improvement. So exploring more ideas to transport the hot water could be effective.
Another possible improvement would be heating or cooling down the water inside the cup. For example, using an AC water heater like in this project *(https://create.arduino.cc/projecthub/mohannad-rawashdeh/arduino-control-ac-water-heater-temperature-e0712e*) could be an interesting option and would also improve the user experiments considerably. At the moment the user needs to know the correct temperature and the application acts more like an inspector and does not do a lot of the work.

In conclusion can be said, that most of the original requirements are implemented and even though the water tank is not implemented, the application still fulfills its core purpose: to assist in brewing the optimal tea.
So, all in all, I am quite happy with how the project turned out and the only thing I would need to change to use it regularly would be to make it more durable so that the cables are not loosen as easily.

5 Attachment

Separate buzzer script:

```
#define SOUND_PIN 1

void setup() {
  pinMode(SOUND_PIN, OUTPUT);
}

void loop() {
   // inspired by https://forum.arduino.cc/t/piezo-buzzer-win-and-fail-sound/133792
   int melody[] = {
      262, 196, 196, 220, 196, 0, 247, 262
   };
   int noteDurations[] = {
      4, 8, 8, 4, 4, 4, 4, 4
   };
   for (int thisNote = 0; thisNote < 8; thisNote++) {
      int noteDuration = 1000 / noteDurations[thisNote];
      tone(SOUND_PIN, melody[thisNote] * 8, noteDuration);
      int pauseBetweenNotes = noteDuration * 1.50;
      delay(pauseBetweenNotes);
      noTone(SOUND_PIN);
  }
}
```

*(**note:** this problem could be fixed later on by changing the Arduino pin to pin 5; pin 0 and 1 are digital pin and are used differently)*