HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Herr
Béla Schroth

Konzeption und Entwicklung eines Küchengeräts zur Kühlung eines trinkfertigen Getränks

Mittweida, 2023

Fakultät Ingenieurwissenschaften

BACHELORARBEIT

Konzeption und Entwicklung eines Küchengeräts zur Kühlung eines trinkfertigen Getränks

Autor:
Herr

Béla Schroth

Studiengang:
Mechatronik

Seminargruppe:
Me18wA-B

Erstprüfer:
Prof. Dr.-Ing. Michael Kuhl

Zweitprüfer:
M.Sc. Kevin Blümel

Einreichung:
Leipzig, 30.05.2023

Verteidigung/Bewertung:
Mittweida, 2023

Faculty Engineering

BACHELOR‘S THESIS

Design and development of a device for cooling drinks in a drinking-vessel of any kind on demand

author:
Herr

Béla Schroth

course of studies:
Mechatronics

seminar group:
Me18wA-B

first examiner:
Prof. Dr.-Eng. Michael Kuhl

second examiner:
M.Sc. Kevin Blümel

submission:
Leipzig, 30.05.2023

defence/ evaluation:
Mittweida, 2023

Bibliografische Beschreibung:

Schroth, Béla:
Konzeption und Entwicklung eines Getränkekühlers,
95 Seiten, 27 Abbildungen, 7 Tabellen, 8 Anlagen,
Hochschule Mittweida (FH),
Fakultät Ingenieurwissenschaften

Bachelorarbeit, 2023

Referat:

In dieser Arbeit geht es um die Entwicklung und Testung eines neuartigen Küchengerätes zur Abkühlung von trinkfertigen Getränken. So wird ermöglicht, Tassen oder Trinkgläser mit sowohl Heiß- als auch Kaltgetränken in das Gerät zu stellen, und diese schnell auf eine zuvor eingegebene Zieltemperatur abzukühlen. Der Nutzer tätigt seine Eingaben dabei über ein Touch-Display. Dieses Konzept wird in dieser Arbeit auf seine Praktikabilität überprüft.

Inhalt

Inhalt I

Abbildungsverzeichnis V

Tabellenverzeichnis VII

Abkürzungsverzeichnis VIII

1 Einleitung 1

1.1 Aufgabenstellung 2

2 Stand der Technik 3

2.1 Kühl-Technologie 3
2.1.1 Kühlung bis 0 °C 3
2.1.2 Abwärme-Abtransport 4

2.2 Elektrische Bauteile 5
2.2.1 Schaltbauteile 5
2.2.2 Sensorik 6
2.2.3 Mikrocontroller 7

2.3 Mechanische Bauteile 8
2.3.1 Pumpen 8
2.3.2 Ventile 8

2.4 Gehäusewerkstoffe & Fertigungsverfahren 9

2.5 Touch-Displays 10

2.6 Konkurrenz 10

3 Präzisierung der Aufgabenstellung 11

4 Konzeption 12

4.1 Kühlsystem 13

4.2 Prozesssteuerung 15

4.3 UI und Software 16

5 Auswahl der Hardware 18

5.1 Elektrik 18

5.1.1 Peltiers ... 18
5.1.2 Stromversorgung ... 19
5.1.3 Lüfter ... 20
5.1.4 Sensorik ... 21
5.1.5 Display und µC ... 25

5.2 Leitungssystem ... 27
5.2.1 Kühlkörper und Schlauch ... 27
5.2.2 Pumpe und Ventil ... 29

6 Aufbau der Hardware ... 30

6.1 Elektrischer Aufbau ... 30
6.1.1 Schaltplan ... 30
6.1.2 Leiterplattenlayout ... 32

6.2 Gehäuse ... 33
6.2.1 CAD ... 34
6.2.2 Fertigung ... 36

6.3 Montage ... 37

7 Software ... 41

7.1 Konzeption ... 41

7.2 Programmierung – µC ... 42
7.2.1 FrigidusPotus.h ... 43
7.2.2 Programmcode ... 44

7.3 Programmierung – Display ... 49

8 Leistungsversuch ... 52

9 Konklusion und Ausblick ... 56

Literatur ... 59

Anlagen ... 61

Anlagen, Teil 1 ... 63

Anlagen, Teil 2 ... 65

Anlagen, Teil 3 ... 66

Anlagen, Teil 4 ... 70

Anlagen, Teil 5 ... 72

Anlagen, Teil 6 76

Anlagen, Teil 7 86

Anlagen, Teil 8 91

Selbstständigkeitserklärung 95

Abbildungsverzeichnis

Abbildung 1: Spannungsteiler 7

Abbildung 2: Systemfunktion 12

Abbildung 3: Kühlsystem 13

Abbildung 4: Prozesssteuerung 15

Abbildung 5: UI - Aufbau 16

Abbildung 6: Peltiers 18

Abbildung 7: Netzteil 20

Abbildung 8: Lüfter 20

Abbildung 9: NTCs 21

Abbildung 10: IR-Sensor 23

Abbildung 11: Versuchsaufbau IR-Sensor 24

Abbildung 12: Display 27

Abbildung 13: Schlauch 28

Abbildung 14: Kühlkörper 28

Abbildung 15: Pumpe 29

Abbildung 16: Ventil 29

Abbildung 17: Fuß & Klappe 36

Abbildung 18: Gehäuse, unlackiert 36

Abbildung 19: Gehäuse, lackiert 37

Abbildung 20: Spannungswandler 37

Abbildung 21: Netzteil Anschlüsse 38

Abbildung 22: Netzteil, installiert 38

Abbildung 23: Leiterplatte, installiert 38

Abbildung 24: Foto- & Touchsensor 38

Abbildung 25: Display, installiert 39

Abbildung 26: Pumpe, Ventil installiert 39

Abbildung 27: Pumpe, modifiziert 39

Tabellenverzeichnis

Tabelle 1: Peltiers 18

Tabelle 2: Abwärme-Versuch 21

Tabelle 3: NTC-Versuch 22

Tabelle 4: IR-Sensoren 22

Tabelle 5: IR-Sensor-Versuch 24

Tabelle 6: Nextion Displays 26

Tabelle 7: Leistungsdiagramm 53

Abkürzungsverzeichnis

UI	**U**ser **I**nterface
GUI	**G**raphical **U**ser **I**nterface
µC	**M**ikro**c**ontroller
RGB	**R**ed **G**reen **B**lue
MOSFET	**M**etal-**O**xide-**S**emiconductor **F**ield-**E**ffect **T**ransistor
A/D	**A**nalog/ **D**igital
PWM	**P**ulse **W**idth **M**odulation
CAD	**C**omputer **A**ided **D**esign
IR	**I**nfra-**R**ed
FOV	**F**ield **O**f **V**iew
RTP	**R**esistive **T**ouch **P**anel
CPU	**C**entral **P**rocessing **U**nit
PC	**P**ersonal **C**omputer
IARC	**I**nternational **A**gency for **R**esearch on **C**ancer
ADU	**A**nalog-**D**igital-**U**msetzer
NTC	**N**egative **T**emperature **C**oefficient
PTC	**P**ositive **T**emperature **C**oefficient
TDP	**T**hermal **D**esign **P**ower
RTOS	**R**eal **T**ime **O**perating **S**ystem
MSB	**M**ost **S**ignificant **B**it
FDM	**F**used **D**eposition **M**odeling
DC	**D**irect **C**urrent
UART	**U**niversal **A**synchronous **R**eceiver **T**ransmitter
DTR	**D**ata **T**erminal **R**eady
PAP	**P**rogramm**a**blauf**p**lan

1 Einleitung

Bei dem Konsum von Heißgetränken kann das Problem bestehen, dass ein Getränk zu heiß ist, um es ohne gesundheitliche Risiken zu verzehren. Unter Zeitdruck kommt es vor, dass die verfügbare Zeit nicht ausreicht, um auf eine natürliche Abkühlung des Getränks zu warten.

Bei dem Verzehr von Kaltgetränken tritt möglicherweise das Gegenteil ein: ein Getränk im Kühlschrank auf eine niedrige Temperatur zu kühlen braucht Zeit, da der Wärmeaustausch zwischen dem Gefäß und der kalten Luft nicht besonders schnell voran geht.

Beide Fälle scheitern an der geringen Wärmeleitfähigkeit von Luft, welche 0,02 $\frac{W}{m*K}$ beträgt (Duden, 2012, S. 15). Im Vergleich dazu beträgt die Wärmeleitfähigkeit von Wasser 0,6 $\frac{W}{m*K}$ und die von Aluminium 234 $\frac{W}{m*K}$ (ebd.).

Ein Problem beim kühlen von Kaltgetränken in der Gastronomie ist, dass meist viele verschiedene Kaltgetränke in einem Kühlschrank gelagert werden. Immer wenn ein Gast ein bestimmtes Getränk bestellt, muss der Kühlschrank geöffnet werden, was die kalte Luft in ihm entweichen lässt. Dies erzeugt erhöhte Energiekosten, was es gerade in Zeiten einer Energiewende zu vermeiden gilt.

Diese drei Tatsachen legen ein Überdenken der Kühlmethoden von Getränken nahe. Dies führte zu der Idee eines Gerätes, welches imstande ist, eine möglichst große Bandbreite an verzehrfertigen Getränken möglichst effektiv in ihrem jeweiligen Gefäß zu kühlen.

Zusätzlich könnte ein solches Küchengerät unter anderem die Kreation neuartiger, z. Bsp. alkoholischer Mixgetränke ermöglichen, welche dann aufgrund ihres Alkoholgehaltes nahe oder sogar unter 0°C abgekühlt werden könnten, ohne zu gefrieren.

1.1 Aufgabenstellung

Das Ziel dieser Arbeit ist die Konzeption und Entwicklung eines universellen Getränkekühlers für in Gläsern oder Tassen befindlichen Getränken. Es soll der gesamte Weg von der Idee bis hin zu einem Prototyp erarbeitet werden.

Folgende Unterpunkte sollen dabei abgearbeitet werden:

- theoretische Vorüberlegungen
- Erarbeitung eines Konzepts
- Entwurf der Hardware
- begründete Auswahl von Komponenten
- Planung und Modellierung eines Gehäuses im CAD
- Fertigung des Gehäuses
- Entwicklung und Programmierung einer Software
- empirische Ermittlung der Kühlleistung
- Abschätzung der Praktikabilität

Anforderungen

Die Anforderungen an das Gerät sind vor allem, den Getränkekühler im Küchenumfeld oder in einer Bar auf Abruf einsetzbar zu machen. Dabei soll es möglich sein, ein möglichst beliebiges Trinkgefäß mit möglichst beliebigem Füllstand auf einen dafür vorgesehenen Platz zu stellen, und die darin befindliche Flüssigkeit in möglichst kurzer Zeit auf eine beliebig niedrigere Temperatur als die Ausgangstemperatur abzukühlen. Das Gerät soll imstande sein, eine eingegebene Zieltemperatur auf 1°C genau zu treffen.

2 Stand der Technik

2.1 Kühl-Technologie

Dieses Kapitel beschäftigt sich mit den Kühltechnologien, die für den Getränkekühler relevant sind oder in Zukunft sein könnten.

2.1.1 Kühlung bis 0 °C

Um eine Temperatur niedriger als die Umgebungstemperatur erzeugen zu können, muss nach dem 2. Hauptsatz der Thermodynamik Arbeit aufgewendet werden. Betrachtet werden zu diesem Zweck Kompressionskälteanlagen und Peltiers.

Das Funktionsprinzip einer Kompressionskälteanlage ähnelt einer umgekehrten Dampfkraftanlage. In einem Kühlkreislauf wird ein Stoff verwendet, welcher bei Temperaturen über 0°C gasförmig, und darunter flüssig ist. So können Ammoniak (NH_3), Kohlenstoffdioxyd (CO_2) oder auch Chlor-Fluor-Derivate verwendet werden, wobei letztere aus Umweltschutz- und gesundheitlichen Gründen heutzutage nicht mehr verwendet werden (Hoffmann, 1956, S. 474).

Das Kühlmittel wird mittels eines Kompressors verdichtet, bis es flüssig wird. Aufgrund der Gesetzmäßigkeiten der Thermodynamik, muss die innere Energie dabei gleichbleiben. Aufgrund der Volumenarbeit des Kühlmittels steigt also dessen Temperatur. Diese kann mittels eines Leitungssystems an die Umgebung abgegeben werden. Das Kühlmittel bewegt sich durch ein Drosselorgan, an dessen Ende es aufgrund des Druckabfalls wieder gasförmig wird. Dadurch, dass es nun einen Teil seiner inneren Energie an die Umgebung abgegeben hat, ist es im gasförmigen Zustand, also unter normalem Druck, kälter als vorher. „Mit fallender Kühltemperatur vergrößert sich demnach die Kompressionsarbeit“ (Hoffmann, 1956, S. 476).

Peltiers stellen eine weitere Möglichkeit der Kühlung dar. Der Peltier-Effekt beschreibt einen Temperaturunterschied, welcher durch einen elektrischen Strom erzeugt wird (Drebushachak, 2008, S. 311).

Peltier-Elemente werden aus Legierungen aus Halbleitern und anderen Metallen bzw. Halbmetallen hergestellt. Abhängig vom elektrischen Widerstand und der thermischen Leitfähigkeit sowie ihrem Seebeck-Koeffizienten, sind Peltiers unterschiedlich Leistungsstark. Der erzeugbare Temperaturunterschied kann bis zu 78K betragen. Die Kühlleistung ist linear abhängig vom Temperaturunterschied zwischen der warmen und kalten Seite des Peltiers. Außerdem ist für eine optimale Leistung eine Regelung der Betriebsspannung des Peltier-Elements nötig, da bei höherer Temperatur der elektrische Widerstand

im Bauteil steigt, weshalb die Spannung erhöht werden muss, um den Strom konstant zu halten. Dies ist wichtig, da der Peltier-Effekt, wie eingangs erwähnt, von der Stromstärke abhängt (W. M. Yim, 1972, S. 1121 ff).

Peltiers haben im Vergleich zu Kompressionskälteanlagen den Vorteil, dass sie auf Abruf kalt werden können. Das ist ein entscheidender Faktor für die Funktion des Getränkekühlers. Weitere Auswahlkriterien sind in Kapitel 5.1.1 zu finden.

2.1.2 Abwärme-Abtransport

Dieses Kapitel handelt von Möglichkeiten der Kühlung der warmen Seite der Peltiers.

Kühlkörper sind eine Möglichkeit. Sie bestehen aus wärmeleitendem Material, meist Kupfer oder Aluminium und gehören in die Kategorie der Kontaktkühlung. Ihre Funktionsweise basiert auf der Vergrößerung der Oberfläche, die die Wärme an die Umgebung abgeben kann. Man spricht von passiver Kühlung. Zusätzlich kann ein Lüfter verwendet werden, um die Kühlleistung zu erhöhen. Dann spricht man von aktiver Kühlung (Grabavac, 2011, S. 1).

Wärmerohre (engl. Heatpipes) „sind spezielle Rohre, die Wärme effizient transportieren können“ (Grabavac, 2011, S. 2). Dies erreichen sie durch die gute Wärmeleitfähigkeit des Rohrs und durch ein in ihm eingeschlossenen Kühlmittels. Dieses verdampft an der Wärmequelle, und Kondensiert an der so genannten Wärmesenke. Diese Wärmesenke kann beispielsweise ein Kühlkörper sein (ebd.).

Eine dritte Methode ist die Wasserkühlung. Sie besteht aus einer Pumpe, welche das im Kühlsystem befindliche Wasser zirkulieren lässt, einem Kühlkörper aus wärmeleitendem Material, welcher auf der Wärmequelle platziert und der von Wasser durchströmt wird, und Radiatoren, welche aus vielen dünnen Lamellen oder Kupferrohren bestehen, durch welche das warme Wasser fließt und es somit seine Wärme an die Umgebung abgeben kann (ebd.). „Aktive Radiatoren besitzen zusätzliche Lüfter“ (Grabavac, 2011, S. 2), welche die Kühlleistung nochmals steigern. Zusätzlich existiert ein Ausgleichsbehälter für das Kühlwasser.

Laut den Ergebnissen eines Experiments scheinen Heatpipes die Wärme besser abzutransportieren (Grabavac, 2011, S. 6). Ein hybrides System zwischen Heatpipes und Kühlkörpern mit Lüftern scheint optimal. Weitere Kriterien der Auswahl finden sich in Kapitel 5.1.3.

2.2 Elektrische Bauteile

2.2.1 Schaltbauteile

In diesem Kapitel geht es um die Theorie von Schaltbauteilen zur Schließung von Stromkreisen höherer Spannung und Leistung mittels niedriger Steuerspannung und Steuerleistung.

Es existieren vier Arten von Bauteilen, die diesem Zweck dienen könnten, nämlich Bipolar-Transistoren, Thyristoren, Feldeffekt-Transistoren und Relais. Thyristoren sind nur zum Schalten sehr geringer Leistungen verwendbar und werden deshalb im Folgenden nicht weiter betrachtet.

Bipolar-Transistoren sind ebenfalls für ehr geringere Leistungen geeignet. Zur Verstärkung einer Steuerleistung sind sie jedoch sehr gut geeignet. Das liegt daran, dass Bipolar-Transistoren stromgesteuerte Bauelemente sind. Der Basisstrom I_B steuert den Kollektorstrom I_C. Der Kollektorstrom passiert dabei zusammen mit dem Basisstrom den Emitter (NPN-Transistor in Emitterschaltung). Dieser besteht aus einer dünnen n-dotierten Halbleiterschicht. Aufgrund ihres ohmschen Widerstandes entsteht hier eine Verlustleistung, welche in Form von Wärme abgegeben wird. Bei hohen Strömen kann die entstehende Wärme zur Zerstörung des Bauteils führen. Deshalb ist ein Bipolar-Transistor nicht für hohe Ströme geeignet (Schnabel, 2017, S. 157 ff).

Feldeffekt-Transistoren sind spannungsgesteuerte Transistoren. Dadurch können sie wesentlich verlustärmer und mit höheren Schaltfrequenzen betrieben werden als Bipolar-Transistoren. Zusätzlich ermöglichen sie das Arbeiten mit wesentlich höheren Strömen. Besonders MOSFETs sind für solche Anwendungen gut geeignet. Je nach Bauart darf der Drain-Source-Strom I_{DS} höher oder niedriger sein. MOSFETs werden in n-Kanal- und p-Kanal-Bauart unterteilt, bei denen die Steuerspannung U_{GS} einmal positiv und einmal negativ ist (Schnabel, 2017, S. 170 ff). Im Prototyp werden deshalb n-Kanal-MOSFETs zur Schaltung der Lüfter, der Pumpe und des Ventils verwendet.

Zur Schaltung sehr hoher Leistungen können außerdem Relais verwendet werden. Sie sind meist elektromagnetische Schalter, welche eine vollständige Trennung zweier Stromkreise ermöglichen. Zusätzlich sind sie gegenüber Strom- und Spannungsspitzen sehr tolerant (Schnabel, 2017, S. 135). Nicht zuletzt kann ihr Innenwiderstand sehr gering sein, wodurch sie kaum Wärme emittieren und keine Kühlung benötigen. Relais eignen sich deshalb im Prototyp gut zum Schalten der Peltiers. Aufgrund dessen, dass Relais aus Elektromagneten mit einer gewissen Verlustleistung bestehen, eignen sich, zur Vorverstärkung der Steuerungssignale des µC, Bipolar-Transistoren, um den Strom, der durch den µC fließt, zu minimieren und somit dessen Lebensdauer zu verlängern.

2.2.2 Sensorik

„Ein Sensor ist das primäre Element in einer Meßkette, das eine variable Eingangsgröße in ein geeignetes Meßsignal umsetzt“ (Schaumburg, 1992, S. 1). Für diesen Prototyp wird Sensorik zur Messung verschiedener Temperaturwerte benötigt.

Grundsätzlich existieren sechs Methoden zur Messung von Temperaturen. In Form von Sensoren werden diese Methoden als Ausdehnungs-, Schwingquarz- und Widerstandsthermometer, als Thermoelemente, Pyrometer und in Form der Faseroptischen Temperaturmessung angewandt (Bonfig, 1995, S. 5, 7, 10, 16, 18).

Für den Prototyp sind Widerstandsthermometer und Pyrometer von Interesse.

Widerstandsthermometer funktionieren, indem der elektrische Widerstand eines Halbleiters oder einer Keramik gemessen wird. Der Widerstand ist anhand der Temperaturkennlinie des Sensors berechenbar. Die Steigung der Kennlinie ist abhängig von den verwendeten Materialien des Sensors, und kann grob in NTCs und PTCs eingeteilt werden. NTCs besitzen einen negativen Temperaturkoeffizienten, d. h. ihr Widerstand sinkt mit zunehmender Temperatur, PTCs besitzen einen positiven Koeffizienten (Bonfig, 1995, S. 7).

Die Vorteile von Widerstandsthermometern sind die einfache Auswertbarkeit mittels eines µC, die Einfachheit in der Herstellung und die weite Verbreitung, da damit ein geringer Preis verbunden ist. Zur Auswertung kann man eine Analog-Digital-Umsetzung, oder auch Wandlung, nutzen, bei welcher die Spannung, welche über den NTC abfällt, gemessen wird.

Diese ADUs arbeiten mittels einer Referenzspannung, einem binären Eingangswort und einem Operationsverstärker. Ist die Spannung des NTCs gleich der Referenzspannung, nimmt das binäre Eingangswort seinen maximalen Wert an. Der Wert ist abhängig von der Länge des Eingangsworts, also aus wie vielen Bits es besteht. Der A/D-Wandler ermittelt den Wert der Eingangsspannung indem er im binären Eingangswort jedes einzelne Bit ermittelt, angefangen beim MSB. Die Eingangsspannung wird jeweils mit dem Wert der Referenzspannung, multipliziert mit dem Wert des aktuell untersuchten Bits und der zuvor ermittelten Bits, durch den Operationsverstärker verglichen. Ist die Eingangsspannung größer oder gleich, so wird dem untersuchten Bit eine 1 zugewiesen, andernfalls wird es 0 gesetzt (Huag, 1991, S. 66 ff).

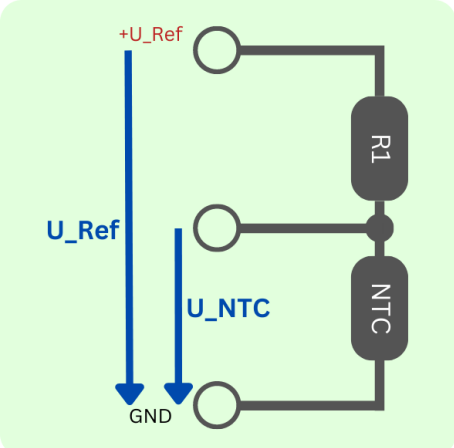


Abbildung 1: Spannungsteiler

So ergibt sich eine näherungsweise Ermittlung der Spannung, welche über den NTC abfällt, und aus welcher man seinen Widerstand ermitteln kann. Dies wird über einen Spannungsteiler, wie in Abbildung 1 skizziert, realisiert.

Ist der NTC sehr heiß, sein Widerstand also sehr gering, fällt über ihn kaum Spannung ab, das heißt die Eingangsspannung U_{NTC} ist sehr niedrig. Hat der NTC einen sehr hohen Widerstand, ist also seine Temperatur sehr niedrig, fällt über ihn fast die gesamte Referenzspannung U_{Ref} ab, was sich in einem hohen oder sogar maximalen Wert des binären Eingangswortes äußert.

Pyrometer sind Strahlungs-Temperaturfühler und messen die vom Messobjekt ausgehende Strahlungsenergie eines kleinen oder großen Teils des elektromagnetischen Spektrums. So messen die meisten Sensoren im Bereich der infraroten und sichtbaren Strahlung. Im Allgemeinen besteht ein Pyrometer aus einem Objektiv, einer Blende, einem Filter, einem Strahlungsempfänger und einer elektronischen Auswertung (Bonfig, 1995, S. 103 f). Die dabei messbare Strahlungsenergie ist abhängig vom Gesamtemissionsvermögens des gemessenen Materials (Bonfig, 1995, S. 18 ff). Ein Versuch dazu wird in Kapitel 5.1.4 durchgeführt. Diese Art von Sensoren ist zwar sehr präzise, muss aber für den jeweiligen Anwendungsfall speziell kalibriert bzw. kompensiert werden (Bonfig, 1995, S. 135 ff).

Die Auswertung eines solchen Sensors ist ebenfalls recht einfach, da die Temperaturdaten dank der elektronischen Auswertung des Sensors meist schon in digitaler Form vorliegen, und beispielsweise über eine serielle Schnittstelle oder einen Bus dem µC übermittelt werden können.

2.2.3 Mikrocontroller

Bei Mikrocontrollern handelt es sich „um Mikroprozessoren mit speziellen Schnittstellen und Hardwarekomponenten sowie festen Programmen - im auf dem Chip integrierten oder externen Speicher - zur Steuerung und Regelung bestimmter Vorgänge“ (Bähring, 2002, S. 349).

Seine Vorteile sind ein geringer Platzbedarf, geringer Stromverbrauch und geringe Kosten. Bei ihrer Auswahl zu beachten sind seine Taktfrequenz, von welcher die Anzahl an Berechnungen pro Zeiteinheit abhängt, sein Programmspeicher, welcher meist nur extern beschrieben und gelöscht werden kann, sein Arbeitsspeicher zur Berechnung von Variablen, die vorhandene Hardware, wie Zeitgeber (Timer) und die Anzahl und Funktion von

digitalen und analogen Eingangs- bzw. Ausgangspins (Bähring, 2002, S. 352 ff). Die konkrete Auswahl ist in Kapitel 5.1.5 beschrieben.

2.3 Mechanische Bauteile

2.3.1 Pumpen

Die Grundfunktion von Pumpen ist das Fördern und verdichten von Fluiden (Müller, 2007, S. 40).

Pumpen können auf fünf Weisen funktionieren. Einmal „[d]urch Einwirkung einer Druckkraft auf die Förderflüssigkeit mittels eines hin- und hergehenden oder rotierenden Kolbens“ (Schulz, 1955, S. 1) und zweitens „durch Übertragung mechanischer Arbeit auf das Fördermittel mit Hilfe eines Schaufelrades“ (ebd.). Die dritte Methode ist durch Impulsaustausch, z. Bsp. mittels eines schneller strömenden Treibmittels, welche seine kinetische Energie teilweise an die langsamer strömende Flüssigkeit abgibt. Viertens und fünftens sind das Zumischen von Druckluft und plötzliche Stoßwirkungen auf die Flüssigkeit weitere Methoden (Schulz, 1955).

Eine Pumpe, welche auf Druckkräften basiert, ist die einzige in Frage kommende Methode, da die zweite Methode einen Kontakt mit der zu transportierenden Flüssigkeit voraussetzt, und alle weiteren unpraktikabel wären.

Kolbenpumpen haben einen hohen Wirkungsgrad und sind aufgrund der Massenwirkung, welche durch die intermittierende Arbeitsweise verursacht wird, an kleinere Drehzahlen gebunden. Dieser Effekt tritt aber nur bei größeren Bauarten auf, und wird deshalb in diesem Fall lediglich moderate Vibrationen verursachen. Weiteres zur Auswahl der Pumpe ist in Kapitel 5.2.2 zu lesen.

2.3.2 Ventile

Ein Ventil ist eine „Vorrichtung, mit der das Ein-, Aus-, Durchlassen von Flüssigkeiten oder Gasen gesteuert wird“ (Cornelsen Verlag GmbH, 2023).

Ventile werden nach ihrer Funktion in zwei Kategorien eingeteilt. Diese Funktionen sind einmal einen Kreislauf eines Fluids zu unterbrechen und einmal den Flüssigkeitsdurchsatz zu kontrollieren. Unter den unterbrechenden Ventilen gibt es u. A. Mehrwegeventile. Diese besitzen mehrere Ein- und Ausgänge (VIRTUALEXPO GROUP, 2023).

Für den Getränkekühler wird ein 3-Wege-Ventil benötigt, da der Prototyp zwischen dem Ansaugen von Getränk und dem von Luft wechseln können muss.

Weiterhin unterscheiden sich Ventile nach der Art ihrer Ansteuerung. Da das Ventil im Prototyp elektrisch gesteuert werden muss, ist hier ein Magnetventil sinnvoll. Weiteres zur Auswahl steht im Kapitel 5.2.2.

2.4 Gehäusewerkstoffe & Fertigungsverfahren

Dieses Kapitel befasst sich mit möglichen Fertigungsverfahren des Gehäuses des Prototyps und dessen Werkstoffen.

„Rapid Prototyping bezeichnet die Anwendung der Technologie der Additiven Fertigungsverfahren zur Herstellung von Modellen und Prototypen, also von physischen Bauteilen ohne Produktcharakter“ (Gebhardt, 2016, S. 6). Im 3D-Druck existieren unterschiedliche Verfahren unterschiedlicher Preisklassen mit unterschiedlichen Vorteilen und Eigenschaften.

Grundsätzlich wird in der additiven Fertigung schichtweise gearbeitet. Die festen Schichten können aus einem Material in flüssiger Form generiert werden, wie z. Bsp. bei der Stereolithographie. Feste Ausgangsmaterialien können einerseits an- oder aufgeschmolzen werden, wie bei Extrusions-, Sinter- und Schmelzverfahren. Sie können auch ausgeschnitten werden wie bei Schicht-Laminat-Verfahren oder aber ein pulver- oder granulatförmiges Material, wie z. Bsp. Keramik, wird verklebt (Gebhardt, 2016, S. 47).

FDM findet Anwendung in Konzeptmodellen, Funktionsprototypen und sogar Endprodukten. Gegenüber allen anderen Verfahren ist es das günstigste, sowohl in der Anschaffung der Maschinen, als auch in der Anwendung (Gebhardt, 2016, S. 260). Mittels FDM können viele Thermoplaste verarbeitet werden, wie z. Bsp. Polyethylen (PE), Acrylnitril-Butadien-Styrol-Polymerisate (ABS), Polyalkylenterephthalate (PET) und Polylactide (PLA) (Erwin Baur, 2019, S. 93 ff). Aufgrund des geringen Preises, der einfachen Handhabung und der Auswahl an verarbeitbaren Kunststoffen ist FDM für den Prototyp bestens geeignet.

Der Kunststoff PET verfügt dabei über gute mechanische, elektrische und thermische Eigenschaften. Es besitzt im amorphen Zustand, welcher durch die Fertigung mittels FDM vorliegt, eine mittlere Härte bei hoher Zähigkeit (Erwin Baur, 2019, S. 157). Das ermöglicht eine einfache Nachbearbeitung und eine gute Handhabung bei der Montage, da die Zähigkeit eventuelle mechanische Belastungen beim Zusammen- oder Umbau kompensieren kann. Die Temperaturbeständigkeit von -30 °C bis 110 °C (ebd.) liegt im optimalen Bereich, da das Gehäuse mit keiner Temperatur von mehr als 100 °C oder weniger als 0 °C in Berührung kommt. Elektrisch wirkt es isolierend (ebd.), was dem Nutzer Schutz vor elektrischen Potenzialen und der Elektrik selbst Schutz vor Kurzschlüssen bietet.

Darum eignet sich der FDM-Druck mittels PET für den Prototyp am besten.

2.5 Touch-Displays

Ein Touchscreen ist ein „berührungssensitiver Bildschirm […] der die Auswahl von Kommandos aus einem Menü (Menütechnik) oder anderen angezeigten Daten durch Berührung der entsprechenden Bildschirmposition mit dem Finger oder einem spitzen Gegenstand erlaubt“ (Sieperman, 2018).

Die zwei am weitesten verbreiteten Touch-Technologien sind resistive und kapazitive Touchscreens. Neben diesen existieren Displays auf Basis von Infrarot-Licht, Induktion, Schall oder dem Piezoeffekt (Litzius, 2017, S. 10).

Kapazitive Touchscreens sind am weitesten verbreitet, da sie in Smartphones, Tablets und ähnlichem verbaut sind. Die Eingabe basiert auf zwei um 90° zueinander verdrehten Streifengittern aus extrem feinen Drähten. Jeder Kreuzpunkt wirkt wie ein Kondensator, dessen Kapazität sich durch Annäherung eines dielektrischen oder leitfähigen Gegenstandes, wie einem Finger, ändert. Diese Änderung der Kapazität zwischen genau zwei Drähten wird gemessen, und ermöglicht eine präzise Positionsbestimmung (Litzius, 2017, S. 10 f).

Resistive Touchscreens (RTP) bestehen aus zwei elektrisch leitfähigen Schichten, welche mit Abstandshaltern übereinander installiert sind. Da auf die untere Schicht abwechselnd in x- und y-Richtung eine geringe Spannung angelegt wird, und bei Berührung des Displays die obere und die untere Schicht sich berühren, kann somit ein Stromkreis in Form eines Spannungsteilers geschlossen werden. Je nachdem wo sich die Schichten berühren, ist der Weg des Stroms durch die Schichten länger, wodurch der Widerstand steigt und ein geringerer Strom fließt. Der Strom ist somit Maß für jeweils die x- bzw. y-Position der Berührung (Litzius, 2017, S. 10).

Es eignen sich also beide der am weitesten verbreiteten Technologien als Display des Prototyps. Die endgültige Wahl wird in Kapitel 5.1.5 begründet.

2.6 Konkurrenz

Juno ist ein Crowdfunding Projekt mit dem Ziel, eine „Mikrowelle für Kälte“ zu erfinden. Die Idee ist ein Gerät zu entwickeln, welches Getränke in Flaschen und Dosen kühlen kann. Das Gerät basiert auf einem thermisch leitfähigen Zylinder, welcher durch einen Motor rotiert wird. Dadurch kann der Zylinder, mit einer Flasche bestückt, von jeder Seite von der Kühleinheit gekühlt werden. Das fertige Produkt soll etwa 400 $ kosten. Die Kampagne wurde bisher mit 268.669 € unterstützt, scheint aber seit 2021 nicht weiter zu laufen, da das letzte Update zu diesem Projekt am 03. März 2021 erschienen ist (Juno, 2020).

Weiterhin existieren Kühlgeräte z. Bsp. in Form eines Becherhalters, welche Leistungswerte von unter 36 W Kühlleistung aufweisen (Reichelt Elektronik, 2023).

3 Präzisierung der Aufgabenstellung

Grundsätzlich soll das Gerät nach zwei Schwerpunkten aufgebaut werden, nämlich der Hardware, d.h. der Auswahl der Komponenten für das Kühlsystem und nach der Software inkl. der entsprechenden Auswahl eines Mikrocontrollers und einer UI.

Im Fokus soll die Auswahl der Peltiers, der passenden Kühlkörper und Lüfter, sowie die Auswahl des Leitungssystems mit all seinen Komponenten stehen. Dabei sind ein Sensor zur Messung der Temperatur des Getränks, sowie zwei Temperatursensoren für einmal die kalte Seite der Peltiers, und einmal die warme Seite, auszuwählen und zu implementieren.

Zusätzlich soll ein zuvor gewählter µC mitsamt seinen Steuerungsbauteilen auf einer selbst designten Leiterplatte installiert werden. Weiterhin soll ein passendes Gehäuse erdacht und gefertigt, der Prototyp aufgebaut und anschließend getestet werden. Die Tests sollen Aufschluss über die Praktikabilität eines solchen Gerätes anhand der Kühlleistung und ggf. der Nutzererfahrung geben.

Ziel ist insgesamt die Überprüfung der Sinnhaftigkeit der Idee eines solchen Getränkekühlers und die Ermittlung seiner möglichen Einsatzzwecke.

4 Konzeption

Die Konzeption beginnt mit dem Gesamtkonzept. Dieses ergibt sich direkt aus der präzisierten Aufgabenstellung.

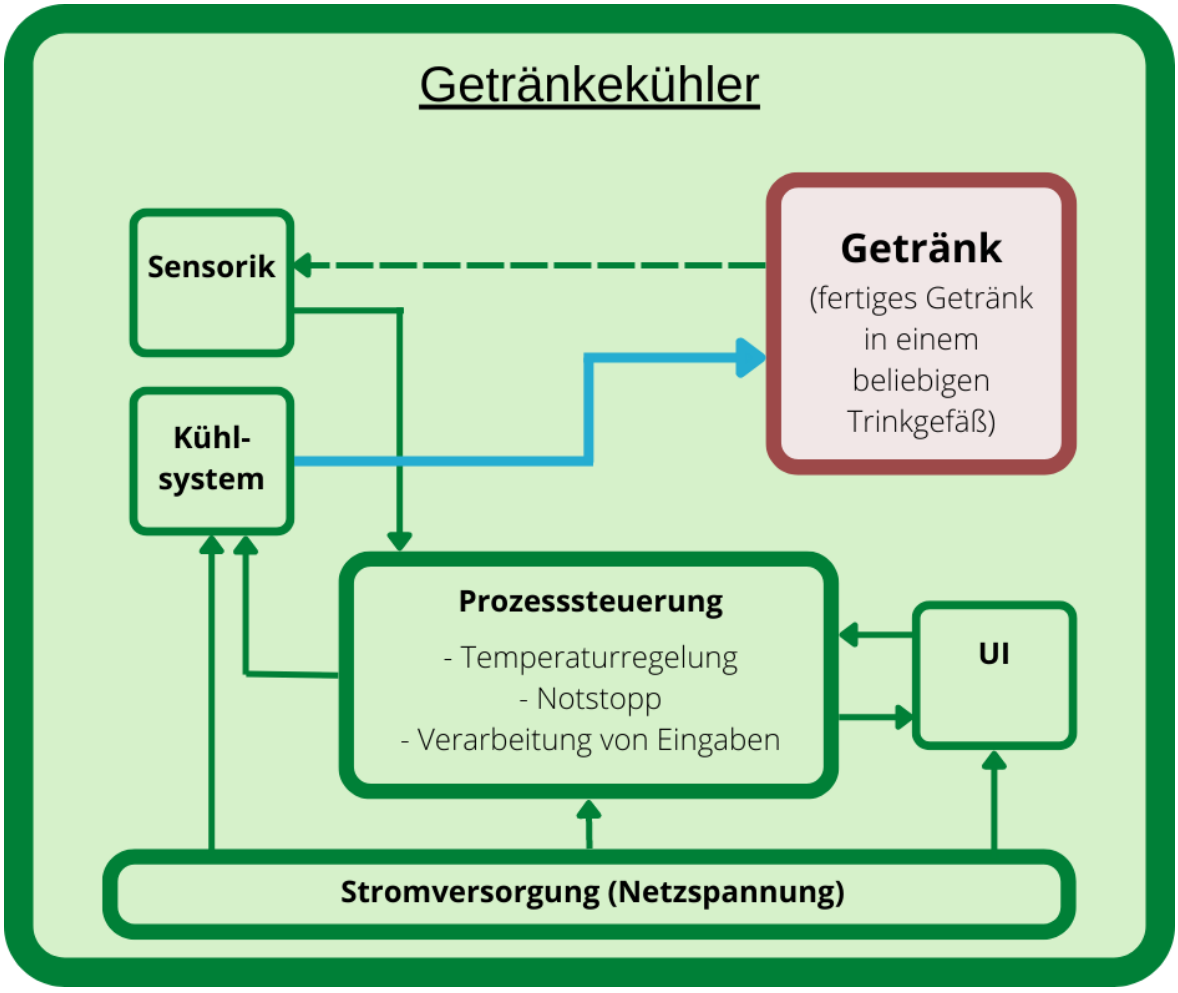


Abbildung 2: Systemfunktion

Die Abbildung 2 soll die Funktionsweise des Getränkekühlers verdeutlichen. Er besteht aus einem Kühlsystem, einer Prozesssteuerung, Sensorik, einer UI und der Stromversorgung.

Das Kühlsystem hat die Anforderung, ein beliebig warmes Getränk auf Wasserbasis im flüssigen Zustand abzukühlen. Bei Wasser, also H2O, liegt dieser Bereich bei einem Luftdruck von 1013 mbar bei 0°C - 100°C (Duden, 2012, S. 16). Dabei soll es eine beliebig niedrigere Temperatur im Bereich des flüssigen Aggregatzustandes erreichen können, und dies auch möglichst schnell bei moderater Effizienz.

Die Prozesssteuerung hat die Aufgabe diesen Vorgang zu überwachen und zu steuern. Dazu benötigt sie die aktuelle Temperatur des Getränks, welche mithilfe der Sensorik erschlossen wird. Des Weiteren muss die Prozesssteuerung auf Eingaben des Nutzers reagieren, welche aus Sicherheitsgründen auch eine Notstopp-Funktion beinhaltet.

Die Eingaben werden über das UI in das System aufgenommen. Gleichzeitig hat dieses den Zweck, den Nutzer über den aktuellen Status des Gerätes, eines Kühlvorgangs oder eines Reinigungsprogramms zu informieren. Vor dem Betrieb ist es dem Nutzer möglich, verschiedene Betriebsmodi auszuwählen oder eine spezielle Zieltemperatur einzustellen.

Das gesamte System abgesehen vom µC wird schlussendlich von einer zentralen Spannungsversorgung gespeist, die ihrerseits durch die Netzspannung gespeist wird. Der µC besitzt sein eigenes Netzteil, welches die durch das Kühlsystem erzeugten Spannungsschwankungen am Hauptnetzteil vom µC fernhalten soll.

4.1 Kühlsystem

Im Folgenden geht es um die Konzeption des Kühlsystems im Detail.

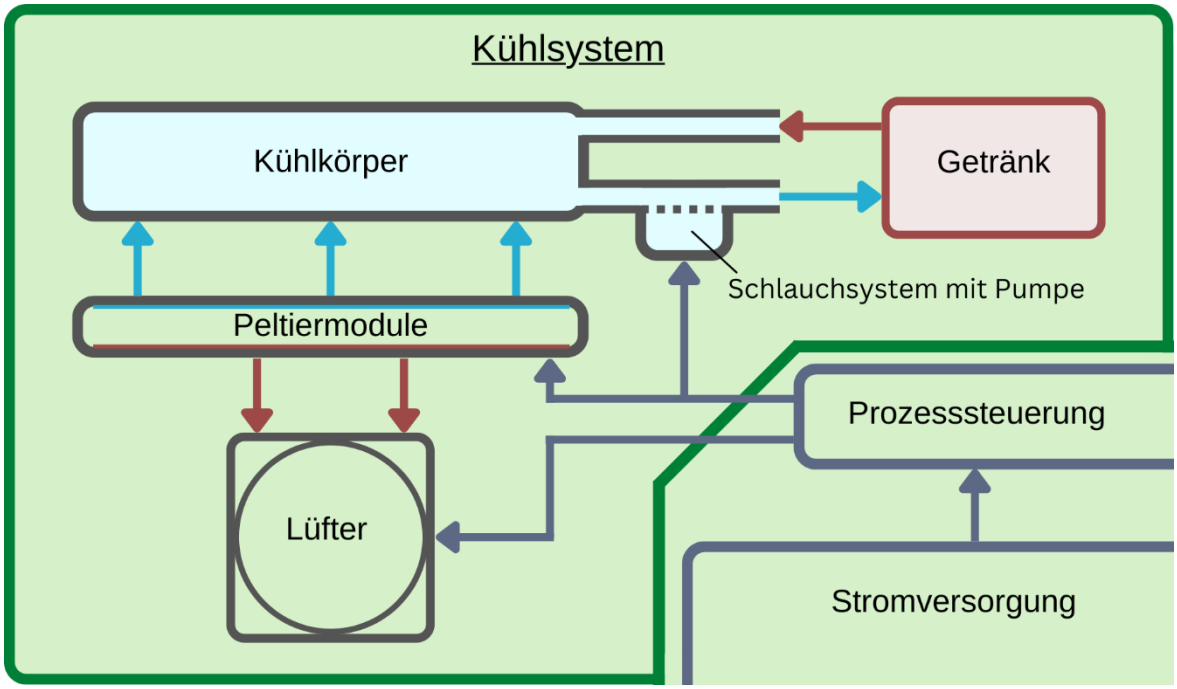


Abbildung 3: Kühlsystem

In der Abbildung 3 ist die Funktionsweise des Kühlsystems dargestellt. Es besteht im Wesentlichen aus einem Kühlkörper, an dessen Enden jeweils ein Schlauch befestigt ist. Einer dieser Schläuche wird durch eine Saugpumpe mit Ventil unterbrochen. Es muss eine Saug- und keine Tauchpumpe sein, um das Getränk aus einem Gefäß in den Kopf des Gerätes saugen zu können, in welchem sich der Kühlkreislauf befindet. Auf der Oberfläche des Kühlkörpers sind Peltiers angebracht. Auf der anderen Seite der Peltiermodule

sind Heatpipes mit Kühlrippen und einem Lüfter befestigt, um die Abwärme der Peltiers abzutransportieren und somit den Temperaturunterschied zwischen warmer und kalter Seite gering zu halten (siehe Kapitel 2.1.1). Die Alternative zu solchen Kühlrippen mit Lüfter wäre eine Wasserkühlung mit Radiatoren, welche aufgrund ihres erhöhten Preises für diesen Prototyp nicht in Frage kommt. Vor der Pumpe sitzt außerdem das Ventil, um ein Ansaugen von Luft statt der im Gefäß befindlichen Flüssigkeit zu ermöglichen und um somit nach Abschluss eines Kühlvorgangs oder eines anderen Programms das Leitungssystem leer zu pumpen.

Das Schlauchsystem inklusive der Pumpe befördert die Flüssigkeit zunächst aus ihrem Gefäß, hinein in den Kühlkörper. Das Volumen des Schlauchsystems und des Kühlkörpers muss möglichst gering sein, um Rückstände von Getränken im Gerät zu minimieren und die erforderliche Mindestmenge an Flüssigkeit niedrig zu halten. Der Kühlkörper wird an der Oberfläche von den Peltiers abgekühlt. Dies wird durch eine Wärmeleitpaste zwischen dem Kühlkörper und den Peltiers begünstigt, welche außerdem eine erhöhte Kühlleistung ermöglicht.

Für die Steuerung dieses Vorgangs ist die Prozesssteuerung zuständig, welche unter anderem die Pumpe für den Flüssigkeitstransport steuert. Das Getränk wird währenddessen im Kühlkörper abgekühlt, und über den zweiten Schlauch zurück in sein Gefäß geleitet.

Die Abwärme der Peltiers überträgt sich zunächst über Wärmeleitpaste in eine Schicht Kupfer, welche sie, durch seine hohe Wärmeleitfähigkeit von 398 $\frac{W}{m*K}$ (Duden, 2012, S. 15) auf mehrere Heatpipes, und damit Kühlrippen, gleichverteilt. Diese transportieren die Abwärme dann mithilfe eines durch einen Lüfter erzeugten Luftstroms in die Umgebung.

Die Stromversorgung aller Einzelsysteme wird teils über Relais, teils über MOSFETs verwirklicht. Die Auswahl ergibt sich aus der Theorie aus dem Kapitel 2.2.1.

4.2 Prozesssteuerung

Hierbei geht es um einen Einblick in das Konzept der Prozesssteuerung.

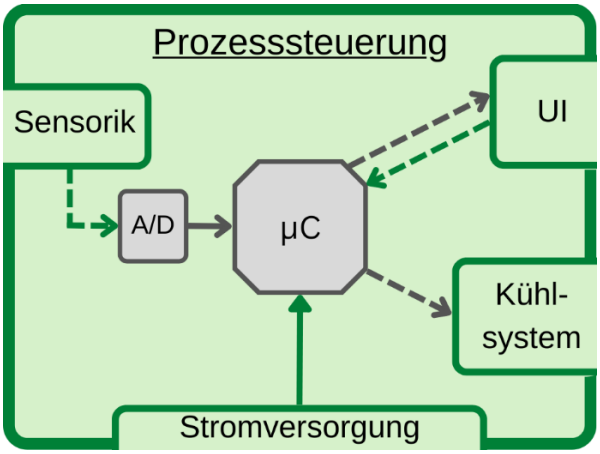


Abbildung 4: Prozesssteuerung

In der Abbildung 4 kann man erkennen, dass die Prozesssteuerung im Wesentlichen aus einem µC besteht. Dieser steuert mithilfe seiner Software alle einzelnen Bestandteile des Getränkekühlers an, so wie in den Abbildungen 2 und 3 skizziert und im Vorhergehenden schon beschrieben.

Dies wird beispielsweise bei den Peltiers aufgrund des maximalen Stroms von ca. 13A bei 24V über Relais, und bei den Lüftern, der Pumpe sowie dem Ventil aufgrund der niedrigeren Leistung und des niedrigeren Preises über MOSFETs realisiert. Relais werden verwendet, um die Wärmeentwicklung an der Leiterplatte zu minimieren, und somit auf eine aktive Kühlung verzichten zu können. Dies ist nötig, da die Leiterplatte im Fuß des Gerätes verortet ist (Kapitel 6.2.1) und dort aufgrund des Risikos eines Wasserschadens durch ein umkippendes Getränk auf Lüftungsschlitze verzichtet wurde.

Zudem benötigt er einen A/D-Wandler um Daten zweier Temperatursensoren zur Messung der Temperaturen der Kühlkörper auswerten zu können. Die Theorie hierzu ist im Kapitel 2.2.2 zu finden. Dies ist hilfreich, um bei niedriger Ausgangstemperatur des Getränks ein Zufrieren der Leitungen zu verhindern und die Lüfter so lange anzusteuern, bis die Kühlrippen abgekühlt sind. Die Temperatur des Getränks wird mittels eines weiteren NTCs ermittelt. Der Grund hierfür wurde ebenfalls in Kapitel 2.2.2 beschrieben.

Alle Daten, wie die aktuelle Temperatur des Getränks, die verstrichene Zeit, der gewählte Betriebsmodus usw. kommuniziert die Prozesssteuerung dem Nutzer über die UI. Dies wird über eine Serielle Schnittstelle ausgeführt, was an der Auswahl des Displays in Kapitel 5.1.5 liegt.

Der µC hat vor allem die Anforderung genügend I/O-Pins zu besitzen, um jede Komponente anzusteuern. Die Anzahl der Timer spielt auch eine Rolle, da hiervon mindestens zwei benötigt werden. Außerdem muss er genügend RAM und Programmspeicher beinhalten. Bluetooth, W-Lan und andere drahtlose Kommunikationsmöglichkeiten benötigt dieser nicht, da dies dem Getränkekühler keinen wirklichen Mehrwert schaffen würde, weil eine Tasse oder ein Glas mit dem zu kühlenden Getränk in jedem Fall manuell in das Gerät gestellt werden muss.

4.3 UI und Software

In diesem Kapitel geht es um das Konzept des User Interfaces und die wichtigsten Funktionen der Software.

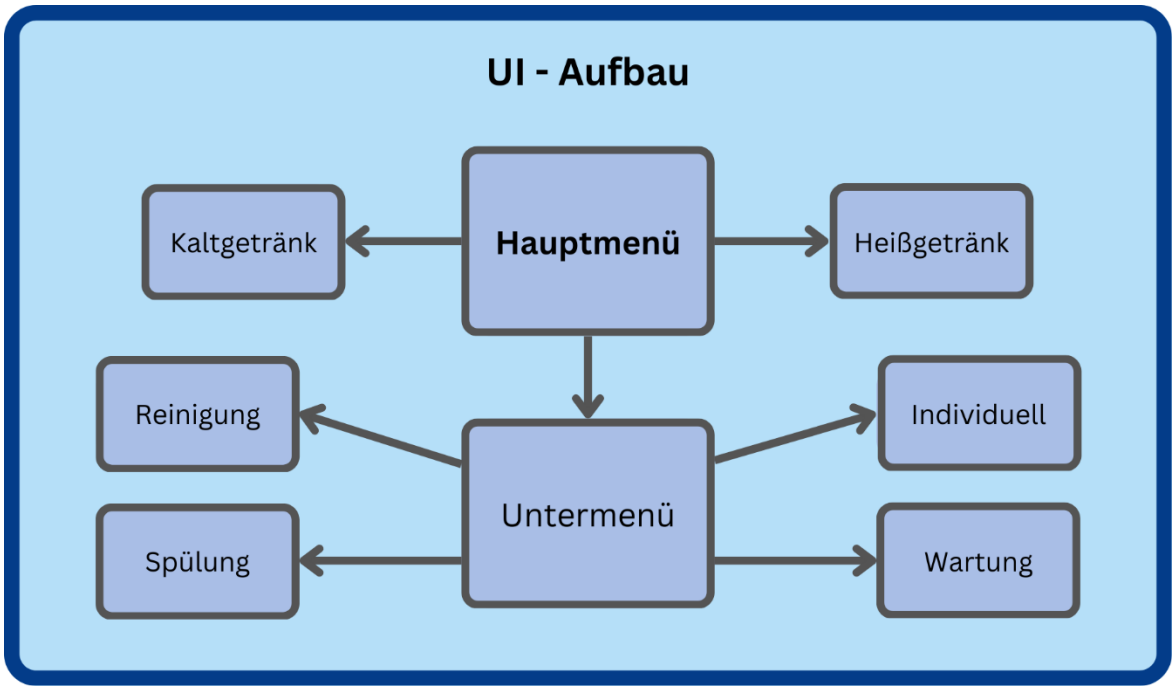


Abbildung 5: UI - Aufbau

Die Abbildung 5 zeigt das Konzept der Funktionsweise des UI. Das Hauptmenü dient der Übersichtlichkeit und Benutzerfreundlichkeit. Das Programm für die Kühlung eines Kaltgetränks, sowie das für ein Heißgetränk, sind vom Hauptmenü aus leicht zugänglich, da man diese Programme am häufigsten braucht.

Alle weiteren Programme sind von einem Untermenü aus zu erreichen, um eine Reizüberflutung und mögliche Verwirrung des Nutzers zu vermeiden. Die Anzeige des Hauptmenüs sowie die des Untermenüs haben eine Schaltfläche, bei dessen Betätigung der Nutzer über alle verfügbaren Funktionen des jeweiligen Menüs aufgeklärt wird.

Das Programm „Kaltgetränk“ startet einen Kühlvorgang mit einer Zieltemperatur von 7 °C. Diese Temperatur ist ein grobes Mittel von optimalen Trinktemperaturen verschiedener Biere (Cyberstore GmbH, 2012) und dem Softdrink Coca-Cola (Brusch, 2016).

Der Taster „Heißgetränk“ startet ein gleichnamiges Programm mit einer Zieltemperatur von 60°C. Diese Temperatur begründet sich dadurch, dass die IARC Getränke mit einer Temperatur über 65°C als „sehr heiß“ eingestuft hat, welche laut verschiedener Untersuchungen auf Dauer das Risiko auf Speiseröhrenkrebs deutlich erhöhen (Farhad Islami, 2009).

Das Programm „Individuell“ steht für ein Kühlprogramm mit manueller Eingabe der Zieltemperatur. Diese kann zwischen 3°C und 100°C liegen. 3°C als Untergrenze wurden deshalb gewählt, um auch bei geringen Mengen an Getränk in den Kühlkörpern nicht zu nahe an den Gefrierpunkt zu kommen. Die 100°C sollen dem Nutzer möglichst große Entscheidungsgewalt über die Zieltemperatur bieten.

Grundsätzlich gleicht der µC nach Start eines Kühlprogramms die Temperatur des Getränks mit der Zieltemperatur ab, und ermittelt, ob ein Kühlvorgang überhaupt nötig ist. Ist die Temperatur des Getränks im Vergleich zur Zieltemperatur zu niedrig, wird der Kühlvorgang abgebrochen.

Andernfalls wird ein Kühlvorgang gestartet, welcher durch eine Steuerung die jeweilige Zieltemperatur auf ca. 1°C genau erreicht. Die Wahl einer Steuerung statt einer Regelung hat mit dem elektrischen Aufbau zu tun, mehr dazu im Kapitel 6.1. Während einer Kühlung wird dem Nutzer der Fortschritt dieser Kühlung in Prozent, sowie sekundengenau die verstrichene Zeit und die aktuelle Temperatur des Getränks angezeigt.

Das Programm Spülung startet ein zeitterminiertes Programm zur Spülung des Kühlkreislaufs. Hierbei wird der Nutzer angewiesen, ein Glas Leitungswasser im Gerät zu platzieren und dies zu bestätigen. Eine normale Spülung besteht aus 20 Sekunden Spülen und Zehn Sekunden Leitungen leeren.

Die Reinigung besteht aus drei Schritten, welche ebenfalls zeitterminiert sind. Der erste Schritt ist die eigentliche Reinigung, welche aus 40 Sekunden Spülung mit einem warmen Spülmittel-Wasser-Gemisch und einer 20 Sekunden langen Leerung der Leitungen besteht. Danach folgen zwei Spülungen, welche ähnlich zur normalen Spülung verlaufen, jedoch statt insgesamt 30 Sekunden nun 40 Sekunden lang dauern.

Das Wartungsprogramm ist für Funktionstests gedacht. Hier kann man alle Komponenten einzeln ansteuern und zeitgleich alle Sensoren auslesen. Dieses Programm dient der Fehlersuche und der allgemeinen Funktionsüberprüfung.

Nach Beendigung eines Programms wird dies auf dem Bildschirm Kund getan, und nach Drei Sekunden kehrt das Gerät in das Hauptmenü zurück.

Jedes Programm kann jederzeit vom Nutzer durch Drücken der physisch am Gerät vorhandenen Stopp-Taste abgebrochen werden. Hierbei wird die Kühlungshardware sofort abgeschaltet und das entsprechende Programm beendet. Danach wird der Nutzer gefragt, ob die Leitungen geleert werden sollen. Je nach Entscheidung des Nutzers wird dies getan, oder direkt zum Hauptmenü zurückgekehrt.

5 Auswahl der Hardware

5.1 Elektrik

5.1.1 Peltiers

Abbildung 6: Peltiers

Die Auswahl der Bauelemente beginnt bei den Peltier-Elementen. Wichtig hierbei ist, dass eine der Vorgaben des Getränkekühlers eine schnelle Kühlung ist. Das setzt eine hohe Kühlleistung bei den Peltiers voraus. Gleichzeitig darf der Preis nicht zu hoch sein.

Die Peltiers sind benannt nach der maximalen Spannung und Stromaufnahme, z. Bsp. bei QC-127-1.4-8.5AS sind es 12,7 V bei maximal 8,5 A. Dies ist direkt abhängig von der Anzahl, und somit bei gleichbleibender Oberfläche der Bauteile, der Dichte der verbauten Halbleiterelemente. Verglichen wurden im Folgenden Peltiers eines spezialisierten Herstellers. Alle Werte stammen vom Arbeitspunkt (ΔT = 40 K, $\vartheta_{heiß}$ = 50 °C, U = 12 V), welcher sich real im Gerät nach einiger Zeit einstellt und bei der Temperatur von 10 °C des Getränks eintritt. 10 °C wurden gewählt, da diese Temperatur im weniger optimalen Arbeitsbereich der Peltiers liegt, und weil es eine sinnvolle Temperatur ist, die beim Kühlen von Kaltgetränken auftritt und die Peltiers vergleichbar macht. Bei Heißgetränken und somit niedrigerem ΔT steigen Kühlleistung und Wirkungsgrad erheblich, wie in Kapitel 8 ermittelt wird.

Tabelle 1: Peltiers

Kriterium	QC-127-1.4-3.7AS	QC-127-1.4-8.5AS	QC-127-2.0-15.0AX
Q	16 W	33 W	60 W
$P_{elektrisch}$	35,2 W	74,8 W	136,4 W
η	0,45	0,44	0,44
Preis	39,83 €	34,76 €	47,03 €

In der Tabelle 1 beschreibt Q die Kühlleistung. $P_{elektrisch}$ zeigt die elektrische Leistungsaufnahme, während η den sich aus P und Q ergebenden Wirkungsgrad darstellt.

Wie man sieht, steigt mit der Stromaufnahme, bzw. der Dichte der Halbleiterelemente, die maximale Kühlleistung. Gleichzeitig steigt ebenfalls die maximal aufgenommene elektrische Leistung, wodurch bei Modulen mit höherer Stromaufnahme der Wirkungsgrad minimal sinkt.

Gleichzeitig steigt der Preis tendenziell, was sich vor allem bei den unterschiedlichen Preisen ab jeweils einer Bestellmenge von über 100 Stück manifestiert. In der Tabelle stehen die Preise für einzelne Peltiers.

Aufgrund der deutlich unterschiedlichen Kühlleistungen und des doch verhältnismäßig geringeren Preisunterschiedes, wurde sich für die QC-127-2.0-15.0AX entschieden. Ihre Kühlleistung ist bei Verwendung für Heißgetränke umso besser (bis zu 110 W bei $\Delta T = 10$ K), und selbst das Kühlen nahe 0 °C ($\Delta T = 50$ K) ist mit immer noch 40 W pro Peltier in Ordnung.

Um eine möglichst hohe Kühlleistung, bei immer noch ausreichendem Wärmeabtransport durch die Lüfter, zu erhalten, sind vier Peltiers verbaut worden.

Sechs von Ihnen haben in den ersten Tests keine guten Ergebnisse gebracht. Nach einigen Temperaturmessungen an verschiedenen Stellen, konnte man feststellen, dass die Kühlung der heißen Seite der Peltiers zu schwach war, und es zu einer Wärmerückkopplung auf die kalte Seite der Peltiers kam. Dies verschlechterte den Wirkungsgrad erheblich, wobei zwei Peltiers weniger deutlich bessere Ergebnisse brachten.

Diese erreichten ohne ein Getränk und mit entsprechender thermischer Isolierung -16°C auf der kalten Seite, bei 52°C auf der heißen. Zu erwarten wären laut Datenblatt ca. -20°C gewesen, was bedeutet, dass dieser Aufbau nahe am Optimum arbeitet. Der leichte Leistungsverlust lässt sich teilweise durch eine leicht verringerte Gleichspannung des Netzteils von gemessenen 22,8V statt 24V begründen. Der Rest könnte an der verlustbehafteten thermischen Isolierung liegen.

5.1.2 Stromversorgung

Die Anforderungen an die Stromversorgung ergeben sich vor allem aus den zuvor ausgewählten Peltiers, von welchen jeweils zwei in Reihe geschaltet sind. Diese benötigen also 24 V und maximal einen Strom von 28 A. Zusätzlich benötigen andere Komponenten eine gewisse Leistung, so braucht die Pumpe 30 W, das Ventil 20 W und die Lüfter ca. 11 W. Gleichzeitig ist der Preis auch bei Netzteilen wichtig, da Netzteile niedriger Spannung und hoher Leistung teilweise sehr teuer sind.

Abbildung 7: Netzteil

Eine weitere Spannungsebene von 12 V lässt sich aus 24 V mit geringem Aufwand erzeugen, was ein weiterer Vorteil ist.

Weil es hier um einen Prototyp geht, bei dem zu Testzwecken ein gewisser Puffer Sinn ergibt, und weil die Kühlung des Netzteils möglicherweise nicht optimal ist, wird hier also ein Netzteil mit einem Strom von maximal 50 A bei 24 V genutzt.

Zusätzlich wird ein kleineres Netzteil mit einer maximalen Leistung von 10 W und einer Spannung von 5 V für den µC verwendet.

Dies dient der Stabilisierung der Spannung, vor allem für die Messung der Temperaturen über die NTCs, welche als Temperatursensoren dienen. Das Thema Analog-Digital-Wandlung und einige seiner Probleme wurde im Kapitel 2.2.2 erläutert.

5.1.3 Lüfter

Eine relativ effiziente und vor allem kostengünstige Variante der aktiven Kühlung sind CPU-Kühler. Ihre Heatpipe-Kontaktflächen besitzen zudem grob die Maße der Peltiers. Eine Alternative wäre eine Wasserkühlung mit Radiatoren gewesen, welche allerdings für diesen Prototyp als zu kostenintensiv erschien.

Deshalb gibt es folgende Kriterien für die Auswahl:

Abbildung 8: Lüfter

- möglichst gute Wärmeleitung
- möglichst niedriger Preis
- „stehende“ Bauart, d. h. der Lüfter lüftet tangential an den Peltiers vorbei

Die letztendliche Auswahl stellt einen Kompromiss zwischen guter Kühlleistung mit hohem Wärmeabtransport, und dem Preis dar. Verbaut wurde ein Lüfter mit jeweils vier Wärmeleitungsrohren aus Kupfer auf jeder Seite aus. Die Kühlleistung eines solchen Kühlers ist abhängig vom Temperaturunterschied zwischen zu kühlender Fläche und der Umgebungstemperatur. Ausgelegt ist der Lüfter für eine CPU mit 180W TDP. Die genauen Anforderungen lassen sich nummerisch nur schwer ermitteln.

Deshalb wurde ein Versuch mittels des Prototyps durchgeführt. Gemessen wurde die Temperatur der heißen Seite der Peltiers nachdem diese und die Lüfter fünf Minuten lang aktiviert waren.

Tabelle 2: Abwärme-Versuch

	Messung 1	Messung 2	Messung 3	Messung 4
Temperatur in °C	51	52	52	52

Der Versuch zeigt, dass die heiße Seite der Peltiers sich bei rund 52 °C stabilisiert. Beim Kühlen von Getränken bis minimal 3 °C läge die maximale Temperaturdifferenz bei unter 50 K. Dies garantiert eine Mindestkühlleistung pro Peltier von rund 40 W. Somit reichen diese Lüfter aus. Der Preis von 20,90 € pro Kühler addiert sich unter Verwendung von vier Kühlern (einer pro Peltier) auf 83,6 € auf. Das ergibt ein adäquates Preis-Leistungs-Verhältnis.

5.1.4 Sensorik

Die Kühlkörper benötigen eine Messung der Temperatur um ein Einfrieren der Leitungen zu verhindern und die Lüfter aktiviert zu lassen, bis die Kühlrippen der CPU-Lüfter abgekühlt sind. Heißleiter, genannt NTC-Widerstände, engl. für Negative Temperature Coefficient (Rosenkranz, 2020), sind in solchen Anwendungen weit verbreitet. Auch in 3D-Druckern kommen diese häufig zum Einsatz, wie z. Bsp. der NTC 3950 100k. Diese sind standardisiert und deshalb sehr günstig.

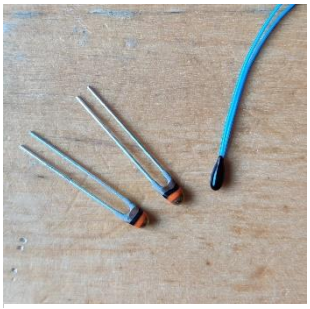


Abbildung 9: NTCs

Allerdings weist der folgende Versuch auf eine geringe Messgenauigkeit hin. Gemessen wurde Wasser mit einem abgedichteten NTC 3950 100K und einem abgedichteten Messfühler meines Multimeters, um die Werte zu vergleichen.

Tabelle 3: NTC-Versuch

Temperaturen	Messung 1	Messung 2	Messung 3
NTC in °C	22	32	54
Multimeter in °C	19	28	46

Möglicherweise liegen diese Abweichungen an einem ungenauen Temperaturkoeffizienten des NTC 3950 100K. Aufgrund höherer Genauigkeit, welche mit +-1% im Datenblatt angegeben ist, und aufgrund seiner guten Dokumentation wurde sich für zwei der NTC-0,2 10Ks von der Firma VISHAY entschieden, welche beide in der Abbildung 9 zu sehen sind.

Des Weiteren wird ein Sensor zur Messung der Temperatur des Getränks in seinem Gefäß benötigt. Einen direkten Kontakt mit dem Getränk zu vermeiden wäre hygienischer als mit einem Temperatur-Messwiderstand zu messen. Hierfür eignet sich ein Infrarot-Thermosensor. Um dieses Konzept zu überprüfen, wurde im Folgenden eine Messreihe, zur Überprüfung der Messgenauigkeiten beim Messen von Oberflächen verschiedener Materialien, durchgeführt.

Verwendet wurde ein Sensor der Firma Adafruit, welcher auf einem Breakout Board installiert ist. Im speziellen wurde der MLX90614 DCI verwendet, da dieser die höchste Genauigkeit aufweist, um wiederum andere Störfaktoren im Versuch ausschließen zu können. Hierfür im Folgenden eine Gegenüberstellung der verschiedenen Infrarot-Sensoren.

Tabelle 4: IR-Sensoren

Kriterium	BAA	BCC	DCI
Anz. Messpunkte	1	1, kompensiert*	1, kompensiert*
Spannung	3 V	3 V	3 V, medizinische Genauigkeit
FOV	Ca. 70°	35°	5°
Preis ca.	15 €	20 €	40 €

*Die Kompensation kompensiert den Wärmegradienten durch die Umgebungstemperatur.

Abbildung 10: IR-Sensor

Dies Erhöht die Genauigkeit. Mehr Informationen im Datenblatt, zu finden im Anhang Teil 1. Die Tabelle impliziert die hohe Messgenauigkeit des Sensors und sein geringes FOV. Dies ist nützlich, um genau sagen zu können, welche Oberfläche von dem Sensor gemessen wird.

Der Versuch der Überprüfung der Anwendbarkeit eines solchen Sensors wurde an den in der folgenden Tabelle aufgelisteten Oberflächen vorgenommen. Alle diese Oberflächen haben die gleiche Temperatur wie der Raum, da sie seit Tagen bei Raumtemperatur gelagert wurden. Die Raumtemperatur wurde mittels drei Messungen auf 26,2 °C gemittelt.

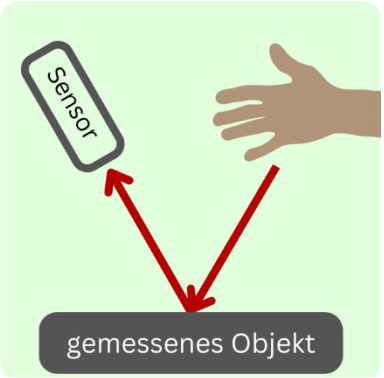


Abbildung 11: Versuchsaufbau IR-Sensor

Der Versuchsaufbau ist in Abbildung 11 dargestellt. In einem Winkel von ungefähr 55° wurde die Oberfläche gemessen. Von einer Oberfläche wurde jeweils eine Messreihe mit, und eine ohne die eigene Hand als Wärmequelle aufgenommen. Somit wurde untersucht, ob der Sensor jeweils die Temperatur der untersuchten Oberfläche, oder anteilig die Reflektion der Infrarotstrahlung der Hand misst.

Zusätzlich wurde warmes Seifenwasser mit einer Temperatur von 47 °C vorsätzlich aufgeschäumt, um zu untersuchen, ob der Sensor trotz des Schaums die Temperatur des Seifenwassers korrekt messen kann. Alle Werte sind in °C angegeben.

Tabelle 5: IR-Sensor-Versuch

Oberfläche	Messung 1	Messung 2	Messung 3	Gerundetes Arithmetisches Mittel
schwarzer Stoff ohne Hand	25,99	25,87	26,13	26,0
schwarzer Stoff mit Hand	26,13	26,17	26,09	26,1
weißes Papier ohne Hand	26,21	26,17	26,23	26,2
weißes Papier mit Hand	26,73	26,71	26,81	26,8
Alufolie ohne Hand	24,99	24,91	24,97	25,0
Alufolie mit Hand	29,79	29,71	29,81	29,8
Seifenwasser-Schaum	29,69	29,35	30,45	29,8

Grundsätzlich könnte es zu geringen Abweichungen der Werte von bis zu 0,3°C durch die Dauer des Versuchs gekommen sein, da sich in dieser Zeit die Raumtemperatur etwas geändert haben könnte.

Zu erkennen ist, dass der schwarze Stoff jeweils korrekt gemessen wurde, da es kaum eine Abweichung zwischen den Messungen mit und ohne Hand gibt.

Die Messungen des weißen Papiers weisen eine deutliche, aber geringe Abweichung auf. Dies lässt auf eine geringe Reflektion von Infrarotstrahlung der Hand schließen.

Die Temperatur der Alufolie ohne Hand ist deutlich zu gering. Eine Abweichung von über 1°C könnte bedeuten, dass die Raumdecke statt der Alufolie gemessen wurde. Vor allem die Temperatur mit Hand zeigt deutlich, dass die Hand statt der Alufolie gemessen wurde. Dies muss aufgrund der Reflektivität des metallischen Materials passiert sein.

Die Messung des Schaums des Seifenwassers zeigt, dass der Schaum die Wärmestrahlung der Flüssigkeit teilweise blockiert. Dies tritt wahrscheinlich aufgrund der thermischen Isolationswirkung des Schaums und der darin eingeschlossenen Luft auf.

Fazit:

Der Sensor weist eine sehr hohe Genauigkeit auf. Das ist an den geringen Abweichungen der einzelnen Messwerte einer Messreihe zu erkennen. Trotz dessen ist er leider völlig ungeeignet. Verschiedenfarbige Getränke, unterschiedliche Transparenzen und im schlimmsten Falle Schaum, welcher bei Tees und z. Bsp. Bier auftreten kann, würden die Messwerte der Temperatur des jeweiligen Getränks teilweise stark verfälschen. Das würde ein genaues Erreichen der Zieltemperatur beim Kühlvorgang unmöglich machen.

Aus diesem Grund wird für die Messung des Getränks ebenfalls ein NTC verwendet. Speziell wird ein EPC B57861-S103 der Firma EPCOS verwendet, welcher eine Genauigkeit von +-1% aufweist, PTFE-isoliert ist und durch seinen geringen Durchmesser von 2,41mm sehr schnell auf Temperaturänderungen reagiert. Dieser ist in Abbildung 9 zu sehen. Zusätzlich befindet sich der Sensor in einem versiegelten PTFE-Schlauch, da dieser wasserdicht und nahrungsmittelverträglich ist.

Außerdem wird ein Fotowiderstand verwendet, um die Helligkeit der Umgebung zu messen. Ein Fotowiderstand ist preisgünstiger als eine Fotodiode und einfach auszuwerten. Mittels dieser Messungen wird die Displayhelligkeit angepasst, um die Lesbarkeit des Displays zu garantieren und in dunklen Räumen den Nutzer nicht zu blenden.

5.1.5 Display und µC

Aufgrund dessen, dass der Prototyp unter anderem die Nutzererfahrung untersuchen soll, und aufgrund der Menge an verschiedenen Anzeigen und Funktionen, fällt die Wahl auf ein Display mit Touchscreen.

Für eine reibungslose Entwicklung und Iterierung des UI und des Prototyps allgemein, muss die Entwicklung und Programmierung der GUI, sowie die Anbindung an einen µC möglichst einfach realisierbar sein.

Hierfür bieten sich die Entwickler-Displays der Firma NEXTION an. Diese sind für eine einfache Implementierung und Anbindung an Entwicklerboards der Firma Arduino konzipiert worden. Hierfür und für die Programmierung des Displays selbst, stellt NEXTION einen passenden GUI-Editor.

Die Anforderungen an das Display sind eine passende Bildschirmdiagonale, adäquate Funktionalität durch vorhandene Hardware und der richtige Preis. Zur Ermittlung der richtigen Bildschirmdiagonalen wurden einige Rechtecke im Maßstab 1:1 mit den entsprechenden Maßen der Displays auf ein Blatt Papier gezeichnet. Somit konnte abgeschätzt werden, dass Diagonalen von ca. 3,5 bis 5 Zoll angemessen sind. Diese These wird im Kapitel 9 ausgewertet.

Die Displays von NEXTION sind unterteilt in verschiedene Produktreihen, die jeweils auf verschiedene Anwendungsgebiete abzielen. Zur konkreten Auswahl wurden Displays mit möglichst ähnlichen Bildschirmdiagonalen gegenübergestellt.

Tabelle 6: Nextion Displays

Kriterium	Basic	Enhanced	Discovery	Intelligent
Bildschirm-diagonale	3,5 ″	3,5 ″	3,5 ″	4,3 ″
Flash	16 MB	32 MB	16 MB	128 MB
RAM	3584 B	8192 B	3584 B	512 kB
MCU	48 MHz	108 MHz	64 MHz	200 MHz
Preis ca.	40 $	40 $	30 $	47 $

Die Auflösung, die Farbdarstellung sowie das Touch-Panel sind bei allen Displays gleich, d. h. es ist jeweils ein RTP.

Auf der Website sind zusätzliche Informationen zu finden. So ist die Discovery-Serie der Firma neuste Produktreihe, welche laut eigener Aussagen mit dem Fokus auf Low-Energie und im Hinblick auf den aktuellen Lithiummangel entwickelt wurde.

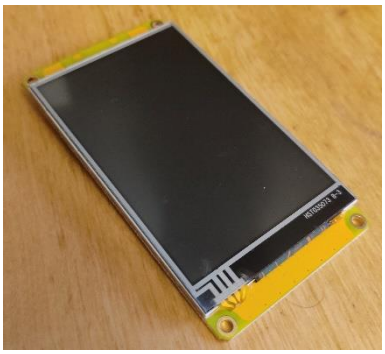


Abbildung 12: Display

So wurde bei diesen Displays bewusst an Ressourcen und Hardware gespart, dafür aber die Software, insbesondere der Bildkomprimierungsalgorithmus, verbessert. Das soll in der Praxis zu ähnlichen Leistungswerten wie bei der Enhanced-Serie führen.

Zur weiteren Beurteilung wurden Beispiele von Projekten mit solchen Displays analysiert. Es ergibt sich, dass das Display der Discovery-Reihe für mein Projekt geeignet ist, nicht zuletzt aufgrund des geringeren Preises.

Als µC wurde der ATMega 328P genutzt. Dieser ist weit verbreitet, deshalb kostengünstig und gut dokumentiert. Für einen Prototyp ergibt es Sinn, einen weit verbreiteten µC zu nutzen, um bei möglichen Problemen in Internetforen o. Ä. schnell eine Lösung zu finden. Das kann den Forschungsprozess beschleunigen.

Zusätzlich erfüllt der ATMega 328P alle Anforderungen an seine Hardware, und ist gleichzeitig nicht überdimensioniert. Dazu gehören u. A. die drei Timer, von denen der Prototyp zwei verwendet und der Programmspeicher, welcher durch die Software zu 37% ausgelastet ist, was Spielraum für weitere Funktionen und Optimierungen lässt. Der RAM ist durch die verwendeten globalen Variablen schon zu 39% ausgelastet, weshalb der nächst kleinere µC der ATMega Serie, der 168PA, nicht in Frage kommt. Weiterhin bieten die 10-Bit-A/D-Wandler eine ausreichende Auflösung zur Ermittlung der verschiedenen Temperaturen im Bereich von -20 °C bis +100 °C. Zuletzt ist auch die übliche Programmierumgebung für Arduinos, welche u. A. den 328P nutzen, aufgrund ihrer mittlerweile guten Debugging-Werkzeuge und der zahlreichen C++-Bibliotheken für das Entwickeln eines Prototyps ohne RTOS gut geeignet.

5.2 Leitungssystem

5.2.1 Kühlkörper und Schlauch

Polytetrafluorethylen (PTFE) ist sehr hitzebeständig und reaktionsträge, und deshalb auch ungefährlich für den menschlichen Organismus (HANSER Verlag, 2023). Aufgrund seiner geringen Haftreibung ist es hygienisch, dieses Material für den Schlauch zu verwenden, da daran nur geringe Mengen an Flüssigkeit nach einer Kühlung haften bleiben.

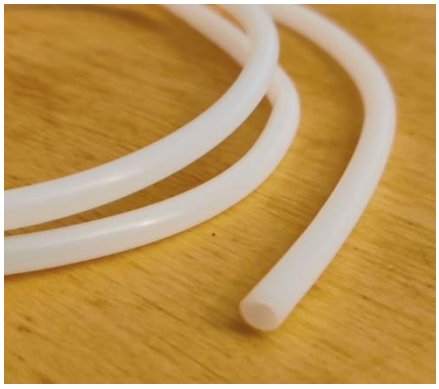


Abbildung 13: Schlauch

Die Berechnung des nötigen Innendurchmessers wurde anhand des sich ergebenden Volumens des Leitungssystems getätigt.

Hierfür wurde die Formel (inneres Schlauchvolumen) + 2 * (inneres Kühlkörpervolumen) genutzt.

Das Schlauchvolumen ist das eines Zylinders, welches sich mit $\frac{\pi}{4} \cdot d^2 \cdot h$ berechnet, während d der Innendurchmesser und h in diesem Fall die Länge des Schlauchs ist.

Das innere Volumen eines Kühlkörpers ergibt sich durch $l \cdot b \cdot h$, wobei l die Länge, b die Breite und h die Höhe eines Kühlkörpers ist. Um das Ergebnis von Litern in Milliliter umzurechnen, wird es mit 1000 multipliziert.

Der Innendurchmesser des Schlauchs wird mit 5 mm und die Länge mit 1 m angenommen. Eingesetzt ergibt sich die Formel $\left(\frac{\pi}{4} \cdot 0{,}025\mathrm{dm}^2 \cdot 10dm + (0{,}4dm \cdot 1{,}2dm \cdot 0{,}1dm) \cdot 2\right) \cdot 1000$, was im Ergebnis ein Volumen von 116 ml ergibt. Dies beschreibt gleichzeitig die minimal nötige Menge an Getränk, um dieses effektiv kühlen zu können. Um einen guten Durchfluss zu ermöglichen, wurde deswegen ein Schlauch mit einem Innendurchmesser von 5 mm und einem Außendurchmesser von 6 mm gewählt.

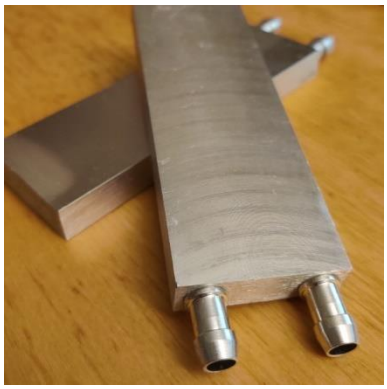


Abbildung 14: Kühlkörper

Die Kühlkörper sind 120 cm lang und 40 cm hoch, da so zwei Peltiers nebeneinander darauf Platz finden und somit mit Zweien gleicher Bauart insgesamt vier Peltiers verbaut werden können. Diese Maße ergeben sich aus den Maßen von CPUs, für welche diese Kühlkörper ausgelegt sind. Eine perfekt passende Alternative konnte nicht zu einem angemessenen Preis gefunden werden. Deshalb wird Kupfer als Verteiler der Kälte, bzw. der Ableitung der Wärme genutzt, um die 5 mm Überstand der Peltiers zu kompensieren. Diese haben eine Fläche von 50 mm x 50 mm.

Zusätzlich muss das Material nahrungsmittelverträglich sein, da durch dessen Leitungsstruktur das Getränk geleitet werden wird. Das dritte Auswahlkriterium ist die thermische Leitfähigkeit. Hierbei liegt Kupfer mit einer Leitfähigkeit von 398 $\frac{W}{m \cdot K}$ vorne, gefolgt von Aluminium mit λ = 234 $\frac{W}{m \cdot K}$ (Duden, 2012, S. 15). Da Leitungen aus Kupfer auf Dauer nicht lebensmittelverträglich (Laboranalyse24, 2022), und teurer sind, wurden Kühlkörper aus Aluminium gewählt.

5.2.2 Pumpe und Ventil

Pumpe und Ventil müssen mit dem gewählten Schlauch kompatibel und lebensmittelverträglich sein.

Die Pumpe muss eine Saugpumpe sein, um die Flüssigkeit in den Kopf des Gerätes saugen zu können. Weiterhin muss sie mindestens genügend Saug-Leistung aufbringen können, um eine Wassersäule mit dem Durchmesser von 5 mm und einer Höhe von mindestens 230 mm vertikal ansaugen zu können. Diese Maße ergeben sich aus dem Durchmesser des Schlauchs und der Anordnung im CAD. Mehr dazu im Kapitel 6.2.

Abbildung 15: Pumpe

Die nötige Saugleistung, als Unterdruck in mmHg (Torr) angegeben, wurde mithilfe einer ein-Meter-Wassersäule abgeschätzt. Zur Umrechnung wurde die Definition $1\ mWs = 9{,}81\ \cdot\ 10^3\ Pa$ und $1\ mmHg = 133{,}32\ Pa$ genutzt. Somit ergab sich ein maximal nötiger Unterdruck von $\frac{9{,}81 \cdot 10^3}{133{,}32} \approx 74\ \mathrm{mmHg}$. Darum wurde eine Pumpe verwendet, welche mit 12 V betrieben werden und einen Unterdruck von - 500 mmHg leisten kann. Sie hält außerdem laut ihres Datenblatts bis zu 110°C warme Flüssigkeiten aus. Das ist gerade ausreichend für den Prototyp, welcher u. A. Heißgetränke kühlen können soll.

Abbildung 16: Ventil

Als Ventil wurde ein Magnetventil zur Umschaltung zwischen dem Ansaugen von Getränk, und dem Ansaugen von Luft zur Leerung des Leitungssystems gewählt. Es ist speziell ausgelegt für den Betrieb mit Flüssigkeiten und benötigt eine Betriebsspannung von rund 24 V.

Die Anschlüsse des Magnetventils sind für Schläuche mit einem Außendurchmesser von 6 mm ausgelegt, und passen somit zum Schlauch und zur Pumpe. Der Formfaktor ist optimal, da das Ventil etwas kleiner ist als die Pumpe, und ihr gegenüber einfach platziert werden kann. Der Preis dieses Ventils ist vergleichsweise hoch, allerdings fehlen alternative Angebote.

6 Aufbau der Hardware

Dieses Kapitel beschäftigt sich mit der Konzeptionierung und dem Design des elektrischen Aufbaus, der Anordnung aller Komponenten im dreidimensionalen Raum, der Fertigung des Gehäuses und der anschließenden Montage.

6.1 Elektrischer Aufbau

Dieses Kapitel beschäftigt sich mit dem Konzept, dem Design und Fertigung der Leiterplatte.

Um dem µC die Kommunikation mit allen weiteren Komponenten zu ermöglichen, ist dieser auf eine Leiterplatte gelötet. Diese Leiterplatte beinhaltet zudem alle nötigen passiven elektrischen Bauteile für den Betrieb der Hardware.

6.1.1 Schaltplan

Zunächst geht es um das Design des Schaltungsaufbaus, welches anhand des in den Anlagen, Teil 1 zu findenden Schaltplans erläutert wird. Die schwarz markierten Schaltungsabschnitte werden im Folgenden als Bereiche bezeichnet, und sind anhand ihrer Nummerierung zu identifizieren.

Um den µC überhaupt betreiben zu können, wird für den ATMega 328P eine Spannung von 5 V DC benötigt. Diese werden extern von einem Netzteil bereitgestellt. In Bereich 1 ist eine Ausgleichschaltung dargestellt. Um niederfrequente, stärkere Schwankungen auszugleichen, wurde zwischen 5 V und GND ein Elektrolytkondensator mit einer Kapazität von 100 µF platziert. Ebenso wurde zum Ausgleichen von hochfrequenten Spannungsschwankungen ein Kondensator mit 100 nF parallelgeschaltet. Zusätzlich existiert eine rote LED zur Anzeige, ob die Spannungsversorgung gegeben ist.

Bereich 2 zeigt Anschlüsse für die Spannungsversorgung der Leiterplatte. Grundsätzlich ist die Leiterplatte in drei Stecker für direkten Datenaustausch verschiedener Sensoren und Komponenten mit dem µC, sowie dessen Programmierung konzipiert. So gehören die Anschlüsse der Gruppen Flash-X, Head-0X und Head-1X jeweils zu einem Stecker. In Bereich 2 sind also die Spannungsanschlüsse für die UART-Schnittstelle (Flash-X), welche zur Programmierung des µC dient, den Stecker 0 (Head-0X) und den Stecker 1 (Head-1X) abgebildet.

Bereich 3 dient des Zurücksetzens des µC. So liegen am Reset-Pin des µC über einen Pull-Up-Widerstand konstant 5 V an, da dieser LOW-aktiv ist. Ein Reset kann einerseits

über einen Reset-Taster ausgelöst werden. Andererseits benötigt die UART-Schnittstelle einen so genannten DTR-Pin, welcher den µC für eine Programmierung in Bereitschaft versetzt. Für dessen Funktion wird der Kondensator 3 (C3) benötigt.

Bereich 4 zeigt eine blaue LED, welche an den Pin 17 des µC angeschlossen ist. Diese ist programmierbar und dient der Fehleranalyse. Im Kapitel 7.2 wird erwähnt, dass eine Animation programmiert wurde, anhand derer die korrekte Arbeitsweise des µC ablesbar ist.

Bereich 5 zeigt Anschlüsse, um wenn nötig noch weitere Pull-Up-Widerstände für die Nutzung des I2C-Buses nachzurüsten.

Um alle analogen Sensoren auslesen zu können, wurden in den Schaltplan fünf Spannungsteiler integriert, welche von den Widerstandswerten auf die verwendeten Sensoren ausgelegt sind. In Bereich 6 ist zu erkennen, dass vier von ihnen auf einen Messwiderstand von 10 kΩ ausgelegt sind. Drei von ihnen werden für die Temperaturmessungen, wie schon in Kapitel 5.1.4 beschrieben, verwendet. Der vierte Anschluss existiert für eine mögliche Nachrüstung eines vierten Messwiderstandes. Darüber hinaus gibt es einen Anschluss für den Fotowiderstand zur Ermittlung der nötigen Displayhelligkeit. Verwendet werden dafür die Pins 22 bis 26 des µC, welche alle A/D-Wandler mit einer Auflösung von 10 Bit aufweisen.

Zur Ansteuerung der Peltiers ist Bereich 7 konzipiert. Die Pins 9 und 10 sind jeweils für die Ansteuerung eines Peltier-Paares verantwortlich. An Pin 11 kann ein weiteres Peltier-Paar hinzugefügt werden. Es folgt eine Transistor-Verstärkerschaltung. Da die Bipolar-Transistoren nur als Schalter dienen, benötigen diese nur einen Vorwiderstand zur Begrenzung des Steuerstroms. An ihrem Kollektor sind jeweils die Spulen der Relais angeschlossen. Somit schalten die Transistoren ihrerseits die Relais. Wie schon in Kapitel 2.2.1 angedeutet, dienen die Bipolar-Transistoren der Vorverstärkung des digitalen Steuersignals, um den Strom an den Pins des µC zu minimieren. Die Relais selbst schließen den Stromkreis zwischen 24 V, einem Peltier-Paar (also zwei Peltiers in Reihe) und GND. Um die Relais auf maximal 16 A auszulegen, sind die beiden Schalter der Relais parallelgeschaltet. Die Relais besitzen jeweils eine so genannte Freilaufdiode (engl. flyback diode), welche die negativen Spannungsspitzen, welche bei Deaktivierung der Spulen der Relais auftreten, ableiten sollen.

Die Bereiche 8.1 und 8.2 erfüllen ähnliche Aufgaben. Sie dienen der Ansteuerung der Pumpe (PUM), der Lüfter (LUE) und des Ventils (VEN). Die Komponenten werden mittels einer MOSFET-Schaltung angesteuert. Diese besteht aus einem MOSFET und einem Pull-Down-Widerstand, welcher zur Entladung des MOSFETs und damit zur Öffnung des durch den MOSFET geschlossenen Schaltkreises verhilft. An den jeweiligen Bauteilen selbst wurde, ähnlich zu den Relais, jeweils eine Freilaufdiode angeschlossen. Hierzu wurden beispielhafte Versuche durchgeführt. Es wurde die Spannung über der Pumpe und dem Ventil bei Abschaltung des jeweiligen Bauteils einmal mit und einmal ohne Freilaufdiode gemessen.

In den Anlagen, Teil 1 unter „Freilaufdioden“ sind die Ergebnisse der Versuche zu finden. Bei der Pumpe kann man klare Spannungsspitzen von unter -500 V bei Abschaltung ohne Freilaufdiode feststellen, wohingegen die negative Spannungsspitze von -4,6 V mit Diode sehr gering erscheint. Beim Betrieb des Magnetventils ist es ähnlich. Eine Spannungsspitze von weniger als -600 V wurde ohne Diode gemessen. Mit installierter Freilaufdiode waren es nur noch -8 V. Außerdem verkürzt die Freilaufdiode in beiden Fällen die Einschwingdauer der Spannung am jeweiligen Bauteil erheblich, wahrscheinlich aufgrund der wegfallenden Selbstinduktion der Spulen.

Bereich 9 des Schaltplans zeigt die Anschlüsse des Stopp-Knopfes und des Displays. Der Stopp-Knopf ist an Pin 32 angeschlossen, welcher einer von zwei unterbrechungsfähigen Pins (externe Interrupts) ist. Somit kann bei entsprechender Programmierung der Stopp-Knopf sowohl digital abgefragt werden, als auch bei Tastendruck ein Interrupt im µC auslösen. Das Display kommuniziert über die integrierte serielle Schnittstelle mit dem µC, welche an den Pins 31 und 30 liegt.

An den Pins 7 und 8 des µC ist ein Oszillator, welcher aus einem Quarzkristall und zwei integrierten Kondensatoren besteht, angeschlossen. Dieser gibt den Prozessortakt von 16 MHz dem µC vor.

Weiterhin sind die Pins 13 bis 16, 17 und 19 des µC offen, welche für weitere Modifikationen verwendet werden können.

6.1.2 Leiterplattenlayout

In den Anlagen, Teil 2 ist das Leiterplattenlayout, einmal von oben und einmal von unten dargestellt. Die Farbe der Leiterplatte wurde hier zur besseren Sichtbarkeit der Leiterbahnen angepasst.

Die Leiterplatte besteht aus zwei Ebenen. Die Hauptebene ist jene, welche man in der Draufsicht erkennen kann. Die gelben Kennzeichnungen sind auf die Leiterplatte aufgedruckt und markieren die Positionen aller Bauteile und benennen diese. Der verwendete Schaltplan ist jener aus Kapitel 6.1.1.

In der Mitte ist der µC aufgelötet. Seine zentrale Platzierung erleichterte die Planung der Leiterbahnen (Routing). Zusätzlich ist die Platzierung aller Objekte in einer Weise gestaltet, dass alle Anschlüsse für Sensoren und Kommunikationen des µC nach außerhalb der Platine leicht zugänglich sind. Dafür sind beispielsweise die UART-Schnittstelle (ganz links), der Stecker 0 (oben Mitte) und der Stecker 1 (oben rechts) ganz am Rand der Leiterplatte platziert. Die UART-Schnittstelle dient der Programmierung des µC. Stecker 0 ist für die Kommunikation mit dem Display, eventuelle Kommunikation über I2C und für den Stopp-Knopf zuständig. Stecker 1 beherbergt die Anschlüsse aller analoger Sensoren. Beide Stecker besitzen, wie im Kapitel 6.1.1 gezeigt, eine eigene Spannungsversorgung.

Die Stecker für die physisch arbeitenden Komponenten, also Lüfter, Pumpe, Ventil und Peltiers, sitzen jeweils sehr nah an ihrem jeweiligen Schaltungsbauteil. So sind die Anschlüsse der Peltiers direkt neben ihren Relais, und die der Lüfter, Pumpe und des Ventils in der Nähe ihrer MOSFETs. Das verringert die Wärmeentwicklung und induktive Störungen in der Leiterplatte. Die vier Pads in den Ecken dienen der Befestigung der Platine durch Schrauben und gleichzeitig als GND-Potential und -Anschluss.

In der Nähe der UART-Schnittstelle befinden sich außerdem die Power- und die programmierbare LED, sowie der Reset-Knopf. Der Oszillator sowie beide Ausgleichskondensatoren befinden sich nahe des µC, um induktive Einflüsse zu vermeiden. Oben links auf der Leiterplatte sind die Anschlüsse der verschiedenen Spannungsversorgungen verortet, wobei jedes Peltier seinen eigenen Anschluss hat. Dies ergibt deswegen Sinn, da dadurch der Strom, welcher durch die Leiterplatte fließt, stark minimiert und somit die Wärmeentwicklung geringgehalten wird. Zusätzlich ermöglicht es schmalere und dünnere Leiterbahnen.

Die Leiterplatte ist 1,6 mm dick, 74 mm hoch und 114 mm breit. Leiterbahnen, welche nur für den Datentransport ausgelegt sind, haben eine Breite von 0,3 mm. Die Leiterbahnen der 5V-Spannungsversorgung sind 0,8 mm, die zwischen MOSFETs und ihren Anschlüssen 2 mm und die zwischen Relais und Stecker jeweils 5 mm breit.

Die erforderlichen Breiten der Leiterbahnen wurde mit folgendem Tool berechnet:
https://www.4pcb.com/trace-width-calculator.html
Dabei musste jeweils mit einer Dicke von 1 oz/ft² gerechnet werden, da dies der Dicke der Kupferschichten in der Leiterplatte entspricht. Die maximal fließenden Ströme sind hierfür aufgerundet worden. Für einen niedrigen Kontaktwiderstand werden für die Peltiers, die Lüfter, die Pumpe und das Ventil T-Stecker mit vergoldeten Kontakten verwendet.

Hergestellt wurde die Leiterplatte von der Firma JLCPCB. Gelötet wurde alles mittels einer Heißluft-Lötstation, einer Kontaktlötstation und eines digitalen Lötmikroskops. Zuerst wurden die Bereiche 1, 2, und 3, welche auf dem Schaltplan in den Anlagen, Teil 1 zu sehen sind, inklusive des µC und Oszillators, fertiggestellt und getestet. Hierbei wurde ebenfalls ein Bootloader auf den µC gebrannt. Folgend wurden die Bereiche 4, 5 und 6 gelötet und getestet. Als letztes wurden alle verbleibenden Komponenten installiert und ebenfalls erfolgreich getestet.

6.2 Gehäuse

Dieses Kapitel beschäftigt sich mit dem Konzept, dem Design und der Fertigung des Gehäuses.

6.2.1 CAD

Details zur Anordnung der Bauteile sind in den Anlagen, Teil 3 zu erkennen. Die Anordnung ergibt sich aus den folgenden Vorgaben:

- kurze erforderliche Schläuche
- Display nah am Nutzer
- kompakte Bauweise
- genügend Platz für große Gläser/ Tassen
- möglichst keine beweglichen Gehäuseteile (mechanisch einfacher Aufbau)

Auf dem Bild „Komponentenanordnung“ in den Anlagen, Teil 3 ist zu erkennen, dass der Kühlapparat zentral platziert ist. Er besteht aus den Ausgewählten Kühlkörpern in der Mitte, auf derer nach außen zeigender Seite die ausgewählten Peltiers angebracht sind. Im Modell sind drei schmale Peltiers in Weiß zu erkennen, da das Modell für bis zu drei Peltiers pro Seite konzipiert ist. An der anderen Seite der Peltiers sind die Kühler bestehend aus Heatpipes, Kühlkörper und Lüfter angebracht.

Vor diesem Kühlapparat sind Pumpe, Ventil und Display platziert. Der Eingang des Ventils ist nach unten, in das modellhafte Glas gerichtet. Der Ausgang liegt exakt gegenüber dem Eingang der Pumpe. Der Ausgang der Pumpe ist auf gleicher Höhe des Eingangs des ersten Kühlkörpers des Kühlapparats. Das Display liegt zentriert vor der Pumpe und dem Ventil.

Aufgrund der Größe des Netzteils, gibt es für seine Platzierung nur zwei Optionen: hinten oder unter dem Glas. Da das Gerät bei einer Platzierung unter dem Glas noch höher wäre, ist es in der Rückwand platziert. Gleichzeitig bietet es dort dem gesamten Aufbau zusätzliche Stabilität. Die Platzierung der Leiterplatte ist abhängig von der Verortung der Anschlüsse des Netzteils. Darum befindet sie sich unter dem Glas und dem Netzteil, im Fuß des Aufbaus.

Zudem ermöglicht dieser Aufbau eine optimale Kühlung des Netzteils. Dieses besitzt einen integrierten Lüfter, der in diesem Aufbau nach oben zeigt. Durch den im Kopf des Gerätes vorliegenden Luftstrom, welcher vom Kühlapparat und dessen Lüftern erzeugt wird, kann dieser die warme Abluft des Netzteils mit nach hinten, aus dem Gerät transportieren.

Das Gehäuse ist in seiner Gesamtheit ebenfalls in den Anlagen, Teil 3 dargestellt. Es wurde im Wesentlichen für die Montage, aber ebenfalls für die Fertigung, in grob zwei Einzelteil-Gruppen unterteilt: Den unteren Teil und den Kopf. Der untere Teil besteht aus dem Fuß und dem Mittelteil. Der Kopf besteht aus dem Hauptteil des Kopfes und dem vorderen Teil.

Der Prototyp besitzt für eine gute Montage und Demontage, sowie ggf. für das Installieren weiterer Modifikationen drei Klappen. Eine Klappe nimmt die halbe Unterseite des Fußes

ein. Diese wurde für den Einbau der Spannungswandler, die Leiterplatte, das Netzteil, den Netzanschluss mit Hauptschalter, für das aufstecken aller Stecker auf die Leiterplatte und für die Programmierung des µC benötigt. In den Anlagen, Teil 3 ist der Fuß ohne Klappe zu erkennen.

Die zweitwichtigste Klappe befindet sich unten am vorderen Teil des Kopfes. Der Kopf ist in den Anlagen, Teil 3 ohne Klappe dargestellt. Diese benötigte man zur Installation aller im vorderen Teil des Kopfes befindlicher Komponenten und für das Aktualisieren der Software des Displays. Zusätzlich musste von hier aus der Kühlkreislauf geschlossen werden, indem die Pumpe mit dem Kühlmodul verbunden wurde.

Die Klappe, welche sich über den gesamten oberen Teil des Hauptteils des Kopfes erstreckt, wurde zur Installation des Netzteils, der Pumpe, des Ventils und zum Einsetzen des gesamten Kühlmoduls, welches in dieser Klappe befestigt ist, benötigt.

Auf dem Bild des Fußes sind die Montageplätze der Spannungswandler (vorne im Fuß) und der Leiterplatte zu sehen. Im hinteren Teil sieht man zwei Aussparungen, wobei die Rechte für den Hauptschalter, und die Linke für eine Kaltgerätebuchse designt wurde. Zusätzlich ist eines von insgesamt zwei im Fuß befindlichen Montagelöchern für Schrauben des Netzteils zu erkennen.

Der Fuß wurde mit dem Mittelteil verklebt. Auf dem Bild in den Anlagen ist der mittlere Teil des Gehäuses von hinten abgebildet. Dort sind Lüftungsschlitze zu erkennen, durch welche das Netzteil seine kühlende Luft bezieht. Im oberen Teil dieses Gehäuseteils sind zwei weitere Löcher für Schrauben des Netzteils erkennbar. Details zu Verschraubungen und wie das Gerät montiert wurde, sind in Kapitel 6.3 zu finden.

Auf das mittlere Teil wurde der Hauptteil des Kopfes geklebt. Zu erkennen sind auf dem entsprechenden Bild in den Anlagen, Teil 3 die Lüftungsschlitze hinten und die Lufteinlässe vorne. Des Weiteren sind Kanten für die Verklebung des vorderen Teils des Kopfes mit dem Hauptteil zu sehen. An den Seiten sieht man Löcher, mittels welchen der Deckel des Hauptteils durch Schrauben fixiert wurde.

Das vordere Teil des Kopfes ist von hinten unten dargestellt. In dessen vorderen Teil ist eine Aussparung für das Display samt Montagemöglichkeit abgebildet. Darüber befindet sich eine Absenkung für den kapazitiven Stopp-Knopf. Die Schaltfläche des Stopp-Knopfes ist das Sechseck, welches in der Gesamtansicht des Gehäuses, neben dem Schriftzug „Stopp“ zu erkennen ist. Dahinter, also aus der Perspektive des Bildes davor, ist die Halterung der Pumpe platziert, an welche die Pumpe angeschraubt wurde. Im unteren Teil des Gehäuseteils befindet sich die Halterung des Ventils, durch welche dieses der Pumpe exakt gegenüber liegt. Nach unten gerichtet sind Schraubenlöcher, welche zum Anschrauben der entsprechenden Klappe benötigt wurden.

An Schrauben wurden entweder M4 oder M3 Schrauben verwendet. Die Löcher für die M4 Schrauben sind auf einschmelzbare Messinggewinde ausgelegt, welche durch das

Erwärmen mittels eines Lötkolbens in den Kunststoff eingelassen werden. Dadurch sind diese Schraubverbindungen stabiler und langlebiger, was vor allem bei der vorderen und der unteren Klappe Sinn ergibt, wenn man diese häufiger abnehmen und dranbauen möchte, um ggf. die Software zu iterieren. Diese Gewinde befinden sich, in den Gegenstücken der erwähnten Klappen, sowie in allen vier Schraubenlöchern des Netzteils und in den Löchern für die Leiterplatte im Fuß.

6.2.2 Fertigung

Wie schon in Kapitel 2.4 gezeigt, ist der 3D-Druck, genauer das FDM-Verfahren eines der besten Fertigungsverfahren für einen Prototyp dieser Art. Als Material wurde PETG statt PET verwendet, welches durch seinen Zusatz von Glykol niederviskoser ist, und sich somit leichter drucken ließ (Filamentworld, 2023).

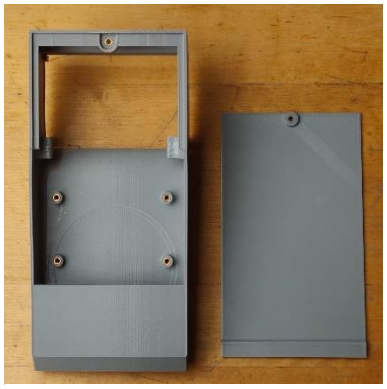


Abbildung 17: Fuß & Klappe

Die Einzelteile des Gehäuses mussten aufgrund des Bauraums von 220 mm x 220 mm x 250 mm (Länge, Breite, Höhe) des verwendeten Druckers teils in mehrere Teile geteilt werden. Hierfür wurden im CAD die Bauteile jeweils an einer sinnvollen Stelle geteilt, sodass sich beide Einzelteile möglichst einfach drucken ließen. Zusätzlich wurden Klebekanten installiert, für eine einfachere Verklebung während der Montage.

So wurden der Fuß, das Mittelteil, der Hauptteil des Kopfes und dessen Klappe jeweils in zwei Teile geteilt. Der Fuß besteht zusätzlich aus dem Fuß an sich, und den vier seitlich angebrachten Stützfüßen, welche alle separat gedruckt wurden. Insgesamt ergab sich eine Anzahl von 15 Einzelteilen, welche insgesamt innerhalb einer Druckzeit von 208,6 Stunden gedruckt wurden. Die Sichtdicke betrug dabei 0,2 mm, die Wanddicke überall 2,5 mm.

Rückblickend betrachtet wären für eine ausreichende Auflösung Schichtdicken von 0,3 mm bis 0,4 mm völlig ausreichend gewesen, welche die Druckdauer erheblich reduziert hätten. Insgesamt wurden 1928 g Material verarbeitet, wovon grob 400 g Stützmaterial waren.

In Abbildung 17 ist der Fuß und daneben dessen Klappe zu erkennen. Zusätzlich kann man bei genauerem Hinsehen die Messinggewinde für die Montage der Leiterplatte erkennen. Diese

Abbildung 18: Gehäuse, unlackiert

wurden allen Bauteilen nach erfolgreichem Druck an den entsprechenden Stellen eingesetzt.

Die gedruckten Teile wurden alle mittels eines speziellen Kunststoffklebers verklebt, abgesehen von den drei Klappen. zusätzlich wurden die durch das Kleben entstandenen „Nähte“ mittels eines 3D-Stifts verschweißt. Alle außen liegenden Oberflächen wurden folgend mittels einer Feile und Sandpapier geebnet und rau angeschliffen. Dies diente der Vorbehandlung vor dem Lackieren.

Abbildung 19: Gehäuse, lackiert

Zusätzlich wurden alle Lücken im Gehäuse abgeklebt. Das Gehäuse vor dem Lackieren, mit seinen abgeklebten Lüftungsschlitzen und der Aussparung des Displays, ist in Abbildung 18 zu erkennen.

Im Folgenden wurde das gesamte Gehäuse einer Vorbehandlung mittels einer Kunststoff-Grundierung unterzogen, welche für bestmögliche Haftung zwischen dem Kunststoff und dem Lack sorgt. Nach einer Trocknung von 20 Minuten wurde das Gehäuse mit einem vor UV-Strahlung und Wasser schützenden, blauen Lack überzogen. Hierfür wurde es an den in den Abbildungen 18 und 19 erkennbaren Stricken aufgehängt, um von allen Seiten lackiert werden zu können.

Das Gehäuse in lackierter Form ist in Abbildung 19 zu erkennen. Danach wurden alle Lücken von ihren Abklebungen befreit. Im letzten Schritt wurden Filzgleiter mit einer Gummischicht unter dem Gerät platziert, um einen Abstand zum Boden und gute Standfestigkeit zu garantieren.

6.3 Montage

Die Montage folgte einem strengen Ablaufplan, da einige Komponenten nur in richtiger Reihenfolge installierbar waren. Zu Anfang wurden alle nötigen Kabel vorbereitet und angelötet. So wurden die Versorgungsleitungen der Spannungswandler und der Leiterplatte angelötet, und mit Aderendhülsen für die Verschraubung im Netzteil versehen. Weiterhin wurden die Sensoren, Peltiers, Lüfter, die Pumpe und das Ventil mit Kabeln und den entsprechenden Steckern versehen. Außerdem wurde die Kaltgerätebuchse mit dem Hauptschalter verbunden, während alle offenen, für die Verschraubung im Netzteil bestimmten Drahtenden mit Aderendhülsen versehen wurden.

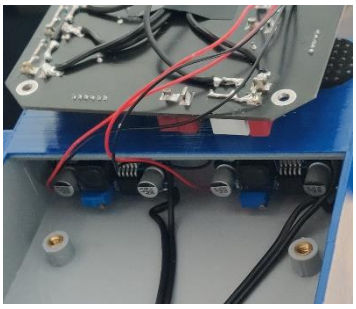


Abbildung 20: Spannungswandler

Eingebaut wurden im Folgenden die Spannungswandler, welche zuvor mit der Leiterplatte verlötet wurden, wie in Abbildung 20 zu sehen.

Abbildung 23: Leiterplatte, installiert

Abbildung 22: Netzteil, installiert

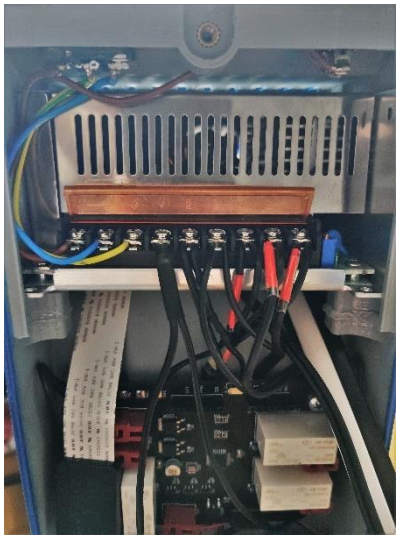


Abbildung 21: Netzteil Anschlüsse

Darauffolgend wurde die Leiterplatte eingebaut. Hierbei wurden an zwei der Schrauben der Leiterplatte Kabel für das GND-Potential, mittels Kabelschuhen, angebracht. Zudem wurden die Kaltgerätebuchse und der Hauptschalter in das Gehäuse eingesetzt. Dies ist in Abbildung 23 zu sehen.

Im Folgenden wurde das Netzteil eingebaut und angeschlossen, wie auf den Bildern 22 und 21 dargestellt. Die oberen Schrauben des Netzteils wurden mit Hilfe eines langen Schraubenziehers, welcher durch zwei dafür vorgesehene Löcher in der Rückwand des Gehäuses geführt worden war, angezogen.

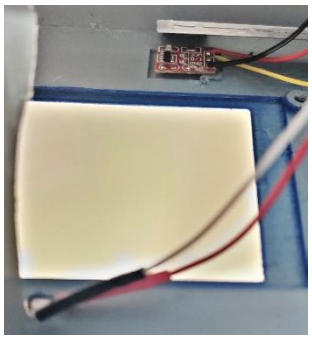


Abbildung 24: Foto- & Touchsensor

Im nächsten Schritt wurden in den vorderen Teil des Kopfes des Gehäuses der kapazitive Touchsensor und der Fotosensor installiert, welche beide in Abbildung 24 sichtbar sind.

Der Touchsensor wurde mittels eines Sekundenklebers an seiner Stelle hinter dem Stopp-Symbol auf dem Gehäuse, fixiert. Hier half der Sekundenkleber die genaue Positionierung des Sensors zu erreichen.

Der Fotosensor wurde unterhalb des für das Display vorgesehenen Platzes eingeklebt. Nachträglich wurde des Sensors Rückseite abgedunkelt, um Rückkopplungen der Helligkeitsänderungen des Displays auf den Sensor zu verringern.

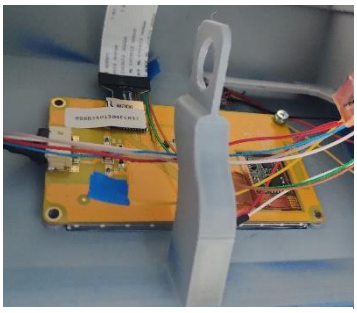


Abbildung 25: Display, installiert

Im nächsten Schritt wurde das Display eingebaut, wie in Abbildung 25 zu sehen. Dieses wurde ebenfalls angeschlossen und es ein erster Test aller bisher verbauten Bauteile wurde durchgeführt. Das Display besitzt außerdem einen Mikro-SD-Karten-Adapter, welcher den im Gehäuse nach oben zeigenden Mikro-SD-Karten-Slot für Updates der Software des Displays einfacher zugänglich macht.

Im Folgenden wurden die Pumpe und das Ventil installiert. Hierbei wurde der Pumpe vorher der Verbindungsschlauch zwischen Pumpe und Ventil hinzugefügt. Eingebaut werden musste zuerst die Pumpe, welche mittels drei Schrauben an ihrem Platz befestigt wurde. Dies ist in Abbildung 27 zu sehen.

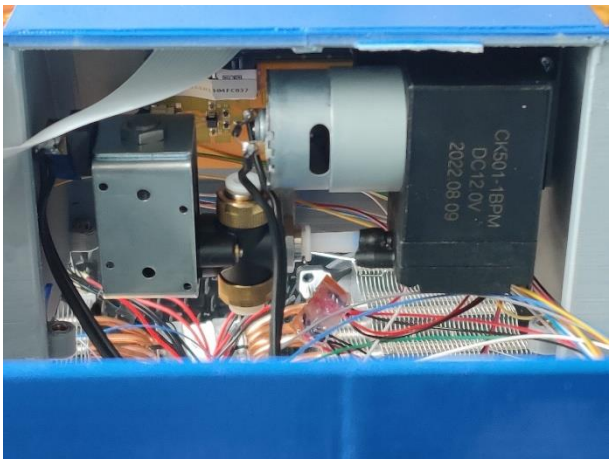


Abbildung 26: Pumpe, Ventil installiert

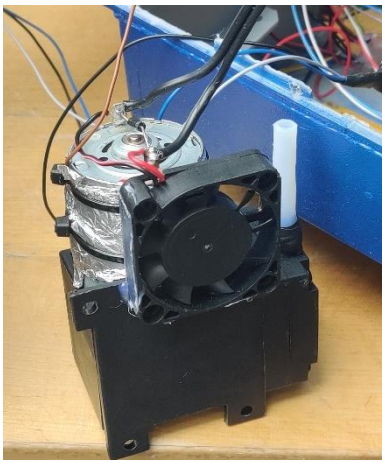


Abbildung 27: Pumpe, modifiziert

Aufgrund dessen, dass die Pumpe durch das Magnetfeld ihres Motors das Interrupt des Stopp-Knopfes auslöste, wurde diese, wie in Abbildung 27 dargestellt, modifiziert. Hierbei wurde ihr Motor in Alufolie gewickelt, welche mit dem Gehäuse des Netzteils verbunden wurde, um sie mit dem Massepotential zu verbinden.

Da die Folie thermisch isolierend wirkt, wurde der Pumpe zusätzlich ein Lüfter parallelgeschaltet, welcher somit zeitgleich mit der Pumpe aktiv ist. Der Lüfter wurde, wie in Abbildung zu erkennen, installiert.

Zuletzt wurde das Ventil, wie in Abbildung 26 dargestellt, eingebaut und mit der Pumpe verbunden. Die Kabel aller im Kopf verbauter Komponenten laufen vor dem Netzteil nach unten zur Leiterplatte.

Im weiteren Verlauf wurde dann das Kühlmodul vormontiert. Im Folgenden wird auf die Bilder und deren Nummerierung in den Anlagen, Teil 4 Bezug genommen.

Als erstes wurde, wie in Bild 01 zu sehen, das Leitungssystem vorbereitet. Hierbei wurden beide Kühlkörper mit einem Schlauch verbunden. zusätzlich wurden die Schläuche für

den Ein- und Ausgang des Leitungssystems angebracht. Bei der Montage der Schläuche wurde ein sich selbst verschweißendes Gummi verwendet, um den Halt und die Dichtheit abzusichern.

In Bild 02 ist die Anordnung der Lüfter einer Seite zu erkennen. Ein solcher Kühlblock besteht aus zwei Kühlkörpern und zwei Lüftern. Beide Lüfter zeigen in dieselbe Richtung.

Auf diesen Aufbau wurde mit Wärmeleitpaste ein Kupferblech aufgebracht, wie in Bild 03 gezeigt. Dieses soll die Abwärme besser auf die Heatpipes der CPU-Lüfter übertragen.

Darauf wurden die Peltiers angebracht, wie auf Bild 04 zu sehen. In der aktuellen Version sind es zwei statt drei der im Kapitel 5.1.1 ausgewählten Peltiers. Diese wurden ebenfalls mittels Wärmeleitpaste thermisch verbunden. Die warme Seite der Peltiers zeigt dabei nach unten, und liegt am Kupfer an.

Auf den Peltiers ist, wie auf Bild 05 dargestellt, ein Kühlkörper angebracht und ebenfalls mit Wärmeleitpaste verbunden. Dieser Aufbau wird von Kabelbindern zusammengedrückt. Kabelbinder eignen sich hier hervorragend, da sie gut festgezogen werden können, und somit nicht nur den physischen Halt des Aufbaus, sondern vor allem einen geringen Abstand der Einzelteile und somit eine gute thermische Verbindung ermöglichen.

Dieser gesamte Vorgang wurde im Folgenden für die zweite Seite mit dem zweiten Kühlkörper wiederholt. Somit ergab sich der Aufbau aus Bild 06.

Dieses gesamte Kühlmodul wurde dann mittels einer Verklebung im Deckel des Hauptteils des Kopfes des Gehäuses angebracht (Bild 07) und anschließend mit Styropor thermisch isoliert (Bild 08). In der aktuellen Version fällt die thermische Isolierung etwas umfassender aus. So wurden auch die Ober- und Unterseite der Kühlkörper des Leitungssystems abgedeckt, um diese vor allem von der warmen Abluft des Netzteils, welches von unten mittig gegen das Kühlmodul lüftet, zu isolieren.

Zum Schluss wurden die Peltiers in Paaren in Reihe geschaltet und die Lüfter wurden in zwei Paaren, welche jeweils in Reihe geschaltet wurden, parallelgeschaltet. Zusätzlich wurden die Kabel mit T-Steckern versehen.

Der letzte Schritt der Montage war das Einsetzen des Kühlmoduls in das Gehäuse. Nachdem das Kühlmodul mit der Pumpe verbunden wurde, wurden die Stecker aller Komponenten auf der Leiterplatte eingesteckt und ein erfolgreicher Test der gesamten Hardware durchgeführt. Das Endresultat kann im Versuchsaufbau in den Anlagen, Teil 8 besichtigt werden.

7 Software

Dieses Kapitel beschäftigt sich mit dem Konzept und der Umsetzung der Software des Prototyps.

7.1 Konzeption

Dieses Kapitel nimmt Bezug auf die Anlagen, Teil 5.

Grundsätzlich wurde die Kommunikation zwischen dem µC und dem Display minimal gehalten, indem das Display fast jede Anzeige selbst steuert. So wechselt das Display in den meisten Fällen selbstständig die Seite, und auch einige Animationen sind vom Display selbst gesteuert. Somit bleibt dem µC mehr Rechenkapazität zur Auswertung von Sensordaten, Berechnung von Variablen und Steuerung von Hardware. Mehr dazu im Kapitel 7.3.

Im PAP des Hauptprogramms ist dessen Arbeitsweise dargestellt. Wird der µC zum ersten Mal eingeschaltet, durchläuft er ein Setup, in welchem die Serielle Kommunikation mit dem Display, die Timer-Interrupts und das externe Interrupt vorbereitet werden. Zusätzlich werden die Aus- und Eingabepins als solche definiert.

Danach beginnt die Endlosschleife, auf der alles Weitere basiert. Innerhalb dieser Schleife wird mit dem Display kommuniziert, sodass ermittelt werden kann, ob ein Bedienfeld auf dem Display betätigt wurde. Die grafische Navigation durch das Menü verarbeitet das Display intern. Mehr dazu in Kapitel 7.3. Zusätzlich wird innerhalb dieser Endlosschleife überprüft, ob die CPU-Kühler, also die Kühler der Peltiers, zu warm sind. Entsprechend werden diese dann durch die Aktivierung der Lüfter gekühlt.

Es gibt insgesamt drei große Unterprogramme, welche die Aufgaben des Getränkekühlers umsetzen. Vor Beginn dieser Unterprogramme wird die jeweils verwendete Hardware, wie z. Bsp. die Zeitgeber (Timer), in einem Setup vorbereitet. Erstens existiert ein temperaturdeterminiertes Kühlprogramm. Im Kern ist dieses immer gleich, denn es endet bei Erreichen der Zieltemperatur. Dieses wird, je nach Wahl des Kühlprogramms, mit unterschiedlichen Werten der Zieltemperatur initiiert, wie im PAP des Hauptprogramms ersichtlich.

Im PAP des Kühlprogramms ist sichtbar, dass zu Beginn dieser Wert übernommen wird. Anschließend wird die nötige Hardware für den Kühlvorgang aktiviert und das Programm betritt eine Schleife. Diese Schleife überwacht nun den Kühlvorgang, indem sie u. A. die Temperatur misst und berechnet. Sie bricht normalerweise ab, wenn die Zieltemperatur erreicht wurde, kann aber auch durch den Stopp-Knopf oder eine zu niedrige Temperatur

der Kühlkörper abgebrochen werden. In dieser Schleife werden konstant alle drei Temperaturen gemessen und der Fortschritt der Kühlung, die verstrichene Zeit und die Temperatur des Getränks auf dem Display dargestellt. Nach Erreichen der Zieltemperatur werden die Peltiers deaktiviert, und das Programm wechselt in der Unterprogramm „Leeren“.

Das Leerungsprogramm aktiviert die Pumpe und das Ventil, um statt weiterer Flüssigkeit nun Luft anzusaugen, und somit die Leitungen zu leeren. Das Programm ist zeitdeterminiert, das bedeutet es endet nach einer im Programm festgelegten Zeit und schaltet danach jegliche Hardware ab. Während der Leerung wird der Nutzer mittels eines Fortschrittsbalkens über den Fortschritt des Programms informiert.

Ein weiteres Basisprogramm ist das Spülprogramm. Es ist dem Leerungsprogramm ähnlich, da es ebenfalls zeitdeterminiert ist. Im entsprechenden PAP ist dessen Initiierung nachvollziehbar. Es wartet auf eine Bestätigung des Nutzers, dass dieser entsprechend der Anweisung gehandelt hat. Nachfolgend aktiviert es die Pumpe, informiert über den zeitlichen Fortschritt und startet eine Leerung nach Ablauf der im Programmcode festgelegten Zeit.

Auf dessen Basis funktioniert ebenfalls das Reinigungsprogramm. Bei ihm ist lediglich die Anweisung an den Nutzer eine andere, und zusätzlich wird über den Fortschritt des gesamten Reinigungsprozesses aufgeklärt. Dieser besteht aus drei Schritten: der Reinigung mittels mit Spülmittel versetzten Wassers und zwei Nachspülungen mit klarem Wasser. Die Nachspülung ist eine Spülung mit dem Zusatz, dass über den Fortschritt der gesamten Reinigung informiert wird. Die PAPs zu beiden Programmen sind ebenfalls in den Anlagen, Teil 5 zu finden.

Das Programm der Wartung existiert für die manuelle Kontrolle und Überwachung aller Komponenten. Wie in seinem PAP zu sehen, prüft es, ob der Nutzer ein Bauteil aktiviert haben möchte, und tut dies dementsprechend. Zusätzlich zeigt es alle gemessenen Sensorwerte an. Nach Beendigung eines jeden Programms kehrt die Software in die Endlosschleife des Hauptprogramms zurück.

7.2 Programmierung – µC

Dieses Kapitel bezieht sich auf die Anlagen, Teil 6. Programmiert wurde alles in der Arduino IDE Version 2.1.0.

Für die Software des µC wurden drei Bibliotheken verwendet. Erstens wurde eine Bibliothek für das Empfangen von Eingaben über das Display inkludiert. Diese ermöglicht eine einfachere Implementierung der Kommunikation mit dem Display. Zweitens wurde die standartmäßig implementierte Bibliothek „Arduino.h“ verwendet, welche das Nutzen von für Arduinos typische Funktionen ermöglicht, wie z. Bsp. „analogRead“ oder „Serial.print“.

Drittens wurde eine Bibliothek mit häufig genutzten Funktionen erstellt, welche in den Anlagen unter dem Namen FrigidusPotus.h zu finden ist. Zudem erfüllt die Bibliothek den Zweck, den Programmcode etwas übersichtlicher zu machen. Die Art der Initialisierung der jeweiligen Funktionen sind der Header-Datei zu entnehmen.

7.2.1 FrigidusPotus.h

FrigidusPotus.h beinhaltet die in der Header-Datei stehenden Funktionen. Dazu gehört die Animation der Status-LED auf der Leiterplatte. Diese ist eine Case-Verzweigung, welche bei jedem Aufruf weiter voranschreitet. Ihre Geschwindigkeit ist somit abhängig von der Frequenz ihrer Aufrufe, welche im Hauptprogramm (Anlagen - Programmcode, ganz unten) festgelegt ist. Die LED wird von der Funktion „StatusLED“ an- und ausgeschaltet, je nachdem welchen Wert der Zähler „LEDcount“ besitzt. Die Funktion ist zyklisch aufgebaut, das heißt der Wert des Zählers wird in der letzten Case-Anweisung zurückgesetzt.

Die Regulierung der Displayhelligkeit ist eine weitere Funktion der Bibliothek. Die Funktion „Helligkeit“ ist für die Messung und Berechnung der durch den Fotosensor wahrgenommenen Helligkeit, und die Berechnung der somit nötigen Displayhelligkeit in Prozent zuständig. Die Berechnung folgt einer linearen Funktion und wird mathematisch korrekt auf Ganzzahlen gerundet. Der Wert dieser Helligkeit wird über die Funktion „HellAnzeige“ an das Display übergeben.

Die beiden Funktionen für die Temperaturmessung sind jeweils auf die entsprechenden Sensoren angepasst. Die Widerstandswerte der NTCs der Kühlkörper werden anhand einer in ihrem Datenblatt angegebenen Funktion berechnet. Der NTC für die Temperaturbestimmung des Getränks wird über eine allgemeingültige Funktion berechnet. Dies dient der Auswertung ihrer elektrischen Widerstände und somit der Berechnung der Temperatur. Details zur Berechnung sind den Kommentaren zu entnehmen.

„TempAnzeige“ ist eine Funktion zur Übermittlung der Temperaturwerte an das Display, genauer an spezielle UI-Komponenten auf dem Display. Die Werte werden über die serielle Schnittstelle übermittelt, indem die Werte der auf dem Display befindlichen UI-Komponenten direkt geändert werden.

Die Funktion „KuehlAnzeige“ ist speziell für die Anzeige der Temperatur des Getränks, der verstrichenen Zeit in Minuten und Sekunden sowie des prozentualen Fortschritts während einer Kühlung konzipiert. Diese Werte werden ebenfalls seriell übermittelt.

„Timer1Toggle” ist eine Funktion zur Aktivierung und Deaktivierung des Timer 1. Er zählt ganze Sekunden und wird sowohl als „Stoppuhr“ als auch als „Eieruhr“ genutzt. „Timer2Toggle“ existiert nicht mehr, da Timer 2 immer aktiv bleibt, welcher eine Periodendauer von einer Millisekunde hat. Timer 0 bleibt ungenutzt.

„EndTrans“ ist eine Funktion, welche sowohl im Code, als auch im Programmspeicher Platz spart. Sie enthält die Sendung der drei Datenpakete, welche immer am Ende einer Übertragung an das Display gesendet werden müssen. Sie markieren dem Display also das Ende einer Übertragung.

7.2.2 Programmcode

Der Programmcode ist ebenfalls in den Anlagen, Teil 6 zu finden.

Variablen

Der Code startet mit der Einbindung der genutzten Bibliotheken. Anschließend findet die globale Variablendeklaration statt.

Zuerst werden alle Displayobjekte, welche bei Berührung ein individuelles Datenpaket zum µC senden, definiert. Die Art und Weise dieser Definition ist den Kommentaren zu entnehmen. Diese Objekte werden im Folgenden einer Liste zugeordnet, welche später im Hauptprogramm wiederholt abgefragt wird.

Es folgen die Pin-Zuweisungen der Ein- und Ausgabehardware des Prototyps. Die Zuweisung ist in einer Weise gestaltet, sodass die entsprechende Hardware über eine Bitweise-Logikoperation aktiviert oder deaktiviert werden kann.

Danach werden so genannte Flags, also Marker definiert. Diese speichern den Status einer bestimmten Funktion oder Komponente des Gerätes. Beispielsweise speichert die Variable „Nachluefften“, ob die Lüfter aufgrund einer zu hohen Temperatur der Peltier-Kühler aktiv sind. Des Weiteren finden sich hier ebenfalls Variablen zur Messung von Zeit, bzw. zur Vorwärts- oder Rückwärtszählung der Zeitvariable des Timer 1. Weitere Details stehen in den Kommentaren.

Unter der Überschrift „Variablendeklaration Global“ stehen im Grunde weitere Marker oder Flags, mit ähnlichen Funktionen.

Setup

Wie in Kapitel 7.1 besprochen, wird beim Start des µC eine Setup-Funktion einmalig durchlaufen. Im Programmcode ist diese von Zeile 423 bis Zeile 470 zu finden. Sie beginnt mit der Initialisierung der seriellen Kommunikation. Danach folgt das Beschreiben der für die Timer und das externe Interrupt nötigen Register im µC. Der Timer 1 ist so eingestellt, dass er ganze Sekunden zählt. Er wird für die Messung verstrichener Zeit im Kühlprogramm und für die zeitliche Determinierung der Spülfunktion. Für die zeitliche Determinierung wirkt der Timer 1 als „Eieruhr“, indem er rückwärtsläuft und die verbleibende Zeit herunterzählt. Der Timer 2 misst exakt eine Millisekunde. Mit seiner Hilfe wird die Frequenz von Messungen und die Aktualisierungsrate von Werten auf dem Display ge-

steuert. Dies ergibt Sinn, da so der serielle Puffer des µC nicht überlastet, und der µC genügend Zeit hat, Eingaben vom Display oder dem Stopp-Knopf auszuwerten.

Die Bedeutungen der einzelnen Werte der Register der Timer und Interrupts sind den Kommentaren zu entnehmen. Die Funktion „sei()“ aktiviert globale Interrupts. Dies wäre hier nicht zwingend nötig, da dies vom Kompilierer (engl. Compiler) der Arduino IDE automatisch implementiert wird. Es ist aber wichtig, möchte man einen anderen Compiler verwenden.

Anschließend findet die Initialisierung der UI-Komponenten statt. Hier wird jedem Bedienfeld, welches eine Funktion im µC auslösen soll, ein Ereignis zugewiesen. Ein Push-Ereignis tritt auf, wenn die Schaltfläche berührt und ein Pop-Ereignis, wenn sie losgelassen wird. Diese Ereignisse werden hier mit den zugehörigen Bedienfeldvariablen verknüpft, während einem Ereignis jeweils eine spezielle Funktion zugewiesen wird.

Zuletzt werden die Pin-Modi der Pins als Ein- oder Ausgang definiert. Dies ist für die korrekte Ansteuerung der Peripheriegeräte nötig.

Hauptprogramm

Wie in Kapitel 7.1 erläutert, ist die Endlosschleife, das Hauptprogramm, das Herz des Systems. Zu finden ist der Code im Programmcode ab Zeile 476. Zuerst wird hier das Stopp-Flag auf „falsch“ gesetzt. Dies dient des Zurücksetzens der Stopp-Knopf-Abfrage. Danach folgt das Überprüfen aller UI-Objekte.

Im Hintergrund zählt immer der Timer 2 aufwärts. Überschreitet er 100, sind also 100 ms vergangen, so wird eine Routine begonnen. In dieser Routine wird zunächst die Displayhelligkeit justiert. Weiterhin schreitet die Animation der Status-LED voran.

Zusätzlich wird hier die Temperatur eines repräsentativen Kühlkörpers der Peltier-Kühler gemessen. Ist dieser Wert über 33 °C, werden die Lüfter aktiviert. Diese bleiben aktiv, bis der Sensor kurz 28 °C misst. Zusätzlich wird überprüft, ob sich das Programm im Wartungsmodus befindet. Dieser wird ebenfalls über das Hauptprogramm umgesetzt. Ist dem so, werden die Lüfter deaktiviert, da das Wartungsprogramm für einen vollständig manuellen Betrieb ausgelegt ist. Wird das Wartungsprogramm verlassen und beträgt die Temperatur über 33 °C, werden die Lüfter reaktiviert. Das An- und Ausschalten wird jeweils über eine Bitweise-Logikoperation realisiert.

Zuletzt befindet sich in der Routine die Abfrage, ob sich das Gerät im Wartungsbetrieb befindet. Ist dem so, werden die beiden übrigen Temperaturen gemessen, und alle drei werden angezeigt. Am Ende wird der Timer 2 auf 0 gesetzt, und der Zyklus beginnt von vorne.

Wartung

Der Wartungsbetrieb findet an sich im Hauptprogramm statt, denn hier werden konstant alle Bedienflächen überprüft. Die Bedienflächen umfassen alle Peripheriekomponenten. Die dazugehörigen Funktionen sind im Programmcode von Zeile 163 bis Zeile 207 zu sehen.

Wird das Bedienfeld der Funktion „Wartung_start“ betätigt, wird das Wartungs-Flag auf „wahr“ gesetzt. Dadurch erkennt das Hauptprogramm, dass sich das Gerät nun im Wartungsmodus befindet.

Wird das Bedienfeld der Funktion „Schliessen“ betätigt, so werden sämtliche Peripheriekomponenten deaktiviert, und die Flags dieser Komponenten zurückgesetzt.

Die Flags sind wichtig damit das Programm weiß, welche Komponenten im Moment aktiv sind. Alle anderen Bedienflächen gehören zu der jeweiligen Komponente. Wird davon eine berührt, wird das Flag dieser Komponente auf „wahr“ gesetzt. Danach folgt ein so genanntes „Hardware-Update“, bei dem die Komponenten mit einem „wahren“ Flag aktiviert werden, und die anderen deaktiviert bleiben. Somit ändert eine Schaltfläche jeweils nur den Zustand einer Komponente, alle anderen Zustände bleiben erhalten.

Spülprogramm

Wird einer der Bedienfelder, die mit den Funktionen „Spuelung“, „Reinigung“ oder „Nachspuel“ verknüpft sind (Programmcode Zeilen 209 - 228), wird der jeweilige Wert in das Zählregister des Timer 1 geschrieben. Zusätzlich existiert eine zweite Zeitvariable, nach dessen Erreichen ein Ereignis ausgelöst wird. Dieses ist im Falle des Spülprogramms die Aktivierung des Ventils, um die Leitungen zu leeren. Folgend wird der Timer 1 aktiviert und das Spülprogramm aufgerufen.

Dieses (Zeile 392 - 417) startet mit dem Setzen der Vergleichsvariable des Timer 1. Sie existiert für die Anzeige des prozentualen Fortschritts der Spülung, Reinigung oder Nachspülung. Der Timer wird auf „rückwärts zählen“ gesetzt und die Pumpe aktiviert.

Während dieses Vorgangs ermittelt der µC alle 50 ms den Fortschritt in Prozent und die Displayhelligkeit, und übergibt beide Werte an das Display. Zusätzlich wird der Stopp-Knopf abgefragt.

Wird das Interrupt des Stopp-Knopfes ausgelöst, wird ein Flag gesetzt. Wenn dem so ist, wird dies von der Funktion „StoppCheck“ erkannt. Dabei wartet die Funktion mittels des Flags „StoppZeit“ eine weitere Periode ab, bis tatsächlich die Stopp-Routine ausgelöst wird. Dies dient dem Abfangen ungewollter Interrupts, welche durch Spannungsspitzen des Netzteils schon ab 0,3 V ausgelöst werden können. Dieser Wert wurde mittels eines Oszilloskops ermittelt. Die Stopp-Routine nutzt dann die Funktion „AllesAus“, um die gesamte Peripherie-Hardware zu deaktivieren. Zusätzlich wird eine bestimmte Seite (Anla-

gen, Teil 7, Bild 16) auf dem Display angezeigt. Mehr dazu in Kapitel 7.3. Zuletzt wird das Flag „Stopp“ auf „wahr“ gesetzt, um die Schleife des aktuell bearbeiteten Programms zu beenden, und ein ordnungsgemäßes Zurücksetzen aller Register und Variablen zu ermöglichen.

Wird kein Stopp ausgelöst, pumpt das Gerät so lange, bis die Zeit des Event 1 erreicht ist. Dann wird das Ventil angeschaltet, wodurch die Leitungen geleert werden. Nach Ablauf der Zeit wird der Timer 1 abgeschaltet. Der Fortschritt von 100% wird sicherheitshalber noch einmal an das Display übergeben, da dieses auf Basis dessen die nächste Seite anzeigt.

Folgend wird die gesamte Peripherie-Hardware deaktiviert und die Stoppzeit sicherheitshalber auf 0 gesetzt.

Leerungsprogramm

Wenn ein Programm durch den Stopp-Knopf unterbrochen wurde, wird der Nutzer gefragt, ob die Leitungen geleert werden sollen. Bestätigt er dies, wird das Programm „Leeren“ aufgerufen. Dieses funktioniert wie eine Spülung, nur ohne Zwischenereignis, und ist 15 Sekunden lang. Für die Spülung wird dieses Programm nicht verwendet, da die Zeiten für die Leerung je nach gewähltem Programm unterschiedlich sein können. Zusätzlich wird das Programm am Ende einer Kühlung gestartet.

Kühlprogramm

Werden die Bedienfelder der Funktionen „Kaltgetr“ oder „Heissgetr“, welche für die Programme „Kaltgetränk“ und „Heißgetränk“ stehen, ausgelöst, so werden die entsprechend vordefinierten Temperaturwerte dem µC übergeben, und das Kühlprogramm wird aufgerufen.

Wird die Schaltfläche der Funktion „Manuell“, also des Programms, in welchem man seine Zieltemperatur selbst wählt, betätigt, so muss die entsprechende Zieltemperatur erst aus dem seriellen Puffer ausgelesen werden, da diese vom Display, nach dem Befehl des Knopfdrucks, gesendet wurde.

Dies wird erreicht, indem die Daten des seriellen Puffers in einer Zeichenkette gespeichert werden. Diese wird dann in einzelne Zeichen, in Form eines Zeichen-Feldes (Array) zerlegt. Im Folgenden wird dieses Array in eine lange Ganzzahlvariable konvertiert. Die Zieltemperatur wird dann gleich dieser Variable gesetzt, wonach ebenfalls das Kühlprogramm aufgerufen wird.

Dieses sendet die ermittelte Zieltemperatur zum Display zurück. Das dient der Fehlersuche, denn ist die Zieltemperatur falsch, kann der Nutzer den Kühlvorgang abbrechen. Im Folgenden wird die Temperatur des Getränks erstmalig ermittelt. Erstens wird dieser Wert einer Vergleichsvariable übergeben, um den prozentualen Fortschritt der Temperatur zu

berechnen. Zweitens wird darauffolgend überprüft, ob die Zieltemperatur kleiner ist als die aktuelle Temperatur, also ob eine Kühlung überhaupt möglich ist. Wenn nicht, wird die entsprechende Seite auf dem Display angezeigt und es findet kein Kühlvorgang statt.

Ist die Zieltemperatur niedriger als die aktuelle Temperatur, so werden zuerst einige Variablen zur Messung der verstrichenen Zeit, des Fortschritts und der Temperatur der warmen Seite der Peltiers angelegt. Zusätzlich wird die Zieltemperatur modifiziert, wenn sie kleiner als 20 °C ist. Dies ermöglicht, laut Tests, ein genaueres Treffen der gewünschten Temperatur bei niedrigen Zieltemperaturen. Zusätzlich wird der Timer 1 vorbereitet, und jegliche nötige Hardware für den Kühlvorgang aktiviert.

Die folgende Schleife wird so lange durchlaufen, bis entweder die Zieltemperatur erreicht ist, der Stopp-Knopf lange genug gedrückt wurde oder die Temperatur der Kühlkörper unter den Gefrierpunkt fällt. Dabei wird die Schleife mindestens zwei Sekunden lang durchlaufen. Dies dient der Stabilität am Anfang eines Kühlvorgangs, bei dem durch die plötzliche Belastung des Netzteils höhere Spannungseinbrüche und -spitzen entstehen können. Somit fängt es mögliche Falschmessungen der Sensoren in dieser Zeit ab.

Anders als in den Kommentaren erläutert, wird die leere Serial.print-Funktion nicht benötigt, damit sich der µC nicht aufhängt. Vielmehr benötigt der Compiler eine Funktion, um den Inhalt der Schleife nicht einfach zu löschen. Scheinbar „weiß“ der Compiler nicht, dass im Hintergrund ein Timer hochzählt, und da alle Werte zu Beginn der Schleife so sind, dass keine der Bedingungen der folgenden Verzweigungen erfüllt sind, geht er von einer Endlosschleife aus.

Nun wird alle 10 ms eine Temperaturmessung am Getränk durchgeführt, um die Temperatur zu mitteln und weitere Schwankungen der 5V-Gleichspannung auszugleichen.

Nach 100 ms wird eine Routine gestartet, welche den Kühlvorgang überwacht und steuert. Dabei werden alle nötigen Temperaturen ermittelt. Zudem werden aus den Werten des Timer 1 die vergangenen Minuten und Sekunden berechnet. Zuletzt wird der Fortschritt in Prozent ermittelt, bevor dann alle Variablen an das Display gesendet werden. Weiterhin wird der Stopp-Knopf überprüft.

Nach Erreichen der Zieltemperatur oder eines anderen Abbruchkriteriums, wird der Fortschritt, wie im Spülprogramm, vorsichtshalber auf 100 % gesetzt. Zudem werden die Peltiers abgeschaltet. Aus Sicherheitsgründen wird maximal 250 ms gewartet, bis das Display die 100 % verarbeitet hat. Mehr dazu in Kapitel 7.3.

Wurde die Schleife aufgrund einer zu niedrigen Temperatur am Kühlkörper abgebrochen, so wird eine Fehlermeldung angezeigt. Als letztes wird das Leerungs-Programm aufgerufen, um die Leitungen zu leeren.

7.3 Programmierung – Display

Dieses Kapitel bezieht sich auf die Anlagen, Teil 7, wobei die in Klammern angegebenen Bilder jene in den Anlagen sind. Der Editor, welcher genutzt wurde um das UI zu programmieren, ist zusammen mit den Display-Seiten in den Anlagen abgebildet.

Beim Start des Displays zeigt es für 1,5 Sekunden seine erste Seite, Seite 0 (Bild 01), an. Dies dient dem Gerät als Vorlauf. So benötigt vor allem das Netzteil im Schnitt zwischen 1 und 1,5 Sekunden für seine Aktivierung. Der zeitliche Fortschritt wird über den Ladebalken am unteren Rand des Bildes angezeigt.

Nach Ablauf der Zeit wird automatisch das Hauptmenü (Bild 02) angezeigt. Auf diesem Bildschirm befinden sich vier funktionale Schaltflächen. Das Zahnrad, welches zu den Systemeinstellungen führen soll, ist noch nicht implementiert und kann zu einem späteren Zeitpunkt hinzugefügt werden. Die Funktionen der Bedienflächen werden dem Nutzer erklärt, indem dieser das Fragezeichen betätigt.

Dieses öffnet die Seite 2 des Displays (Bild 03). Um zum Hauptmenü zurück zu kehren, muss der Nutzer das weiße Kreuz auf rotem Hintergrund betätigen. Dieses wird mehrfach in der Software verwendet.

Die Hilfe-Seite zeigt, dass bei Berührung der Limonade ein Kühlprogramm mit der Zieltemperatur von 7 °C, und bei Berührung der Tee-Tasse eines mit einer Zieltemperatur von 60 °C gestartet wird. Bei Betätigung der Bedienfläche, der mit den drei Punkten, wird das Untermenü „Weiteres“ (Bild 04) aufgerufen. Das Hauptmenü dient sozusagen dem normalen Gebrauch, alles Weitere findet sich im Untermenü.

Dieses enthält sechs Schaltflächen, von denen eines der Schließen-Knopf zur Rückkehr in das Hauptmenü und eines ein weiteres Fragezeichen ist. Dieses (Bild 05) klärt auch hier über die Funktionen des Menüs auf.

So sind hier die Programme für Reinigungen, kurze Spülungen und das Kühlprogramm, bei welchem der Nutzer die Zieltemperatur manuell eingibt, verortet. Zusätzlich ist in der rechten unteren Ecke des Untermenüs das Wartungsprogramm zu finden, welches genutzt werden kann, um Wartungen durchzuführen. Wie in Kapitel 7.2 erklärt, versetzt das Berühren dieser Schaltfläche das Gerät in den Wartungsmodus. Gleichzeitig wird die Seite 6 (Bild 15) angezeigt. Diese Informiert über die Temperaturen, welche durch die Sensoren gemessen werden. Die Bezeichnungen der Werte ist ähnlich denen im Programmcode des µC.

Die einzelnen „Schalter“ aktivieren bzw. deaktivieren die jeweilige Hardware bei Berührung, so wie in Kapitel 7.2 beschrieben. Jede dieser Schaltflächen entspricht im Programmcode des µC einem Objekt, welches jeweils eine Funktion zugewiesen hat. Beispielsweise entspricht der Schalter „Lüfter“ dem Objekt „qLue“, welches an die Funktion „Lue“ (Zeile 175 im Programmcode) geknüpft ist. Diese Funktion setzt in diesem Fall, wie

vorher erklärt, im µC das Flag „LueAN“ auf „wahr“ und führt ein Hardwareupdate durch, sodass die Lüfter eingeschaltet werden. Nachzuvollziehen ist das Ganze im Programmcode in den Anlagen, Teil 6 und im Kapitel 7.2. Das Display übernimmt dabei die Animation des Schalters, indem es das Bild des Schalters auf den eingeschalteten Zustand ändert. Hierbei bewegt sich der „Schiebeschalter“ von links nach rechts und nimmt statt der roten Farbe eine grüne an.

Beim Verlassen des Wartungsprogramms durch das rote Symbol, in der linken oberen Ecke, wird die Funktion „Schließen“ im µC gestartet (Kapitel 7.2). Das Display kehrt in das Untermenü „Weiteres“ zurück.

Bei Betätigung der Schaltfläche „Reinigung“ im Untermenü, wird Seite 11 (Bild 06) angezeigt. Analog wird durch das Bedienfeld „Spülung“ die Seite 12 (Bild 07) angezeigt. Bei jeweils der Betätigung der Schaltfläche in der Mitte auf den Seiten 11 und 12, wird die Seite 13 (Bild 08) gezeigt. Wurde die Reinigung ausgewählt, wird über den Fortschritt durch den Fortschrittsbalken und den Gesamtfortschritt, in dem Moment also „Schritt 1/3“, informiert. Bei einer Spülung existiert die Anzeige des Gesamtfortschrittes nicht. Bei der Reinigung wird die Funktion „Reinigung“ vom µC gestartet, bei der Spülung die Funktion „Spuelung“. Mehr dazu im Kapitel 7.2.

Da die Reinigung, wie schon erwähnt, aus drei Schritten besteht, wird nach Endung des ersten und zweiten Schrittes die Seite 14 (Bild 07) dargestellt. Hierbei besteht der Unterschied lediglich darin, welche Funktion durch die mittlere Schaltfläche ausgelöst wird. Bei einer Nachspülung (Schritt 2 oder 3 der Reinigung) wird die Funktion „Nachspuel“ aktiv. Zusätzlich existiert bei Nachspülungen der Schließen-Knopf in der rechten oberen Ecke nicht.

Wird statt der Spülungen und des Wartungsprogramms im Unterprogramm „Weiteres“ die Kühlung mit selbstdefinierter Zieltemperatur gewählt, so wird der Nutzer zu einem Tastenfeld (Bild 09) weitergeleitet. Hierbei kann er Zahlen von 0 bis 100 eingeben, welche den Zieltemperaturen in °C entsprechen. Betätigt er dann den blauen Pfeil, wird überprüft, ob die Zieltemperatur zwischen 3 °C und 100 °C liegt. Sie darf ebenfalls diese beiden Werte annehmen. Ist dem nicht so, wird die entsprechende Fehlermeldung (Bild 10) gezeigt, wonach der Nutzer die Möglichkeit hat, eine neue Temperatur einzugeben.

Liegt die Zieltemperatur im geforderten Bereich, so wird der Nutzer gefragt, ob alles für die Kühlung bereit ist (Bild 11). Dies kann er bestätigen oder Verneinen. Ebenso tritt diese Abfrage bei Auswahl eines Kühlprogramms im Hauptmenü auf. Je nach zuvor gewähltem Programm, wird eine der drei entsprechenden Funktionen ausgeführt. Mehr dazu in Kapitel 7.2. Von da an funktionieren alle Kühlprogramme fast gleich, wie im vorherigen Kapitel beschrieben. Hat der Nutzer zuvor eine Temperatur selbst bestimmt, so wird diese über die serielle Schnittstelle an den µC gesendet, der diese, wie zuvor erklärt, auswertet und ggf. den Kühlvorgang startet. Seite 17 (Bild 12) ist jene Seite, welche für fünf Sekunden angezeigt wird, sollte der µC feststellen, dass keine Kühlung möglich ist.

Während einer Kühlung wird Seite 15 (Bild 13) vom Display dargestellt. Hier werden, wie auf dem Bild zu sehen und in Kapitel 7.2 erklärt, die eingegebene Zieltemperatur, die aktuelle Temperatur des Getränks, die verstrichene Zeit in Minuten und Sekunden und der prozentuale Fortschritt im Fortschrittsbalken angezeigt.

Wird ein beliebiges Spül- oder Kühlprogramm durch den Stopp-Knopf unterbrochen, so wird, nach Abschaltung der Hardware, Seite 18 (Bild 16) dargestellt. Entscheidet sich der Nutzer für eine Leerung, wird diese entsprechend des Programmcodes des µC durchgeführt, während dem Nutzer der zeitliche Fortschritt durch Anzeige der Seite 19 (Bild 17) im Fortschrittsbalken Kund getan wird. Danach kehrt das Gerät selbstständig in das Hauptmenü zurück.

Nach ordnungsgemäßer Beendigung eines Spül- oder Kühlprogramms und seiner Leitungsleerung, wird der Nutzer darüber, durch Anzeige der Seite 16 (Bild 14), welche drei Sekunden lang angezeigt wird, informiert. Folgend kehrt das Gerät ebenfalls selbstständig in das Hauptmenü zurück.

8 Leistungsversuch

In diesem Kapitel geht es um eine Versuchsreihe zur Ermittlung der effektiven Kühlleistung des Prototyps.

Im Folgenden wurde ein Versuch zur Ermittlung der temperaturabhängigen Leistungskurve des Getränkekühlers durchgeführt. Somit lassen sich Schlussfolgerungen darüber ziehen, ob und wie gut der Prototyp seinen Zweck erfüllt, von welchen Faktoren seine Leistung abhängig ist und welche Verbesserungen seine Leistungsfähigkeit steigern könnten. Es wurden drei Messreihen aufgenommen, bei denen jeweils die Temperatur im Abstand von 5 °C gemessen und die entsprechende Zeit aufgenommen wurde.

Hierfür wurde warmes Leitungswasser, dessen Temperatur zu Anfang bestimmt wurde, von einem Volumen von rund 500 ml genutzt. Diese Menge an Flüssigkeit reduziert den Einfluss verschiedener Messfehler und der Umgebungstemperatur. Das Wasser wurde in einem thermisch schlecht leitfähigen Kunststoff-Behältnis gekühlt. Seine Temperatur wurde durch den Getränkekühler sowie zur Kontrolle durch das Multimeter gemessen. Der Sensor des Multimeters wurde hierfür abgedichtet. Beide Sensoren lagen sehr dicht zusammen, wodurch garantiert wurde, dass sie mit Wasser der gleichen Temperatur in Berührung kamen. Die Zeit wurde mittels der in der Software integrierten Stoppuhr gemessen. Diese wurde zuvor eine Stunde lang mittels einer anderen Stoppuhr verglichen. Dabei wies die Uhr des µC nach einer Stunde keine messbare Abweichung auf. Der Versuchsaufbau ist in den Anlagen, Teil 8 sichtbar.

Im Folgenden wird sich auf die in den Anlagen, Teil 8 befindlichen Messungen bezogen. Wie dort eingangs zu lesen ist, werden die dort angegebenen Einheiten verwendet.

Die Ausgangstemperatur des Wassers wurde im Bereich 50 °C bis 60 °C festgelegt. Es wurden keine höheren Temperaturen verwendet, da die Pumpe des Prototyps bei Temperaturen ab ca. 70 °C denkwürdige Geräusche von sich gab. Wie in Kapitel 5.2.2 beschrieben, sollte die Pumpe laut ihres Datenblatts für Temperaturen bis 110 °C ausgelegt sein, jedoch wurde aus Vorsicht auf Temperaturen über 60 °C verzichtet.

Zu Anfang einer jeden Messung wurden die Masse (m) des Wassers mittels einer Feinwaage und die Temperatur (ϑ_{Start}) mittels des Multimeters bestimmt. Zusätzlich wurde die Temperatur der kalten Kühlkörper (ϑ_{KKk}), also jenen, durch welche das Versuchsmedium geleitet wurde, bestimmt. Dies dient der differenzierteren Auswertung der Ergebnisse. Als Zieltemperatur (ϑ_{Ziel}) wurden 5 °C durch Eingabe dieses Wertes im manuellen Kühlprogramm bestimmt.

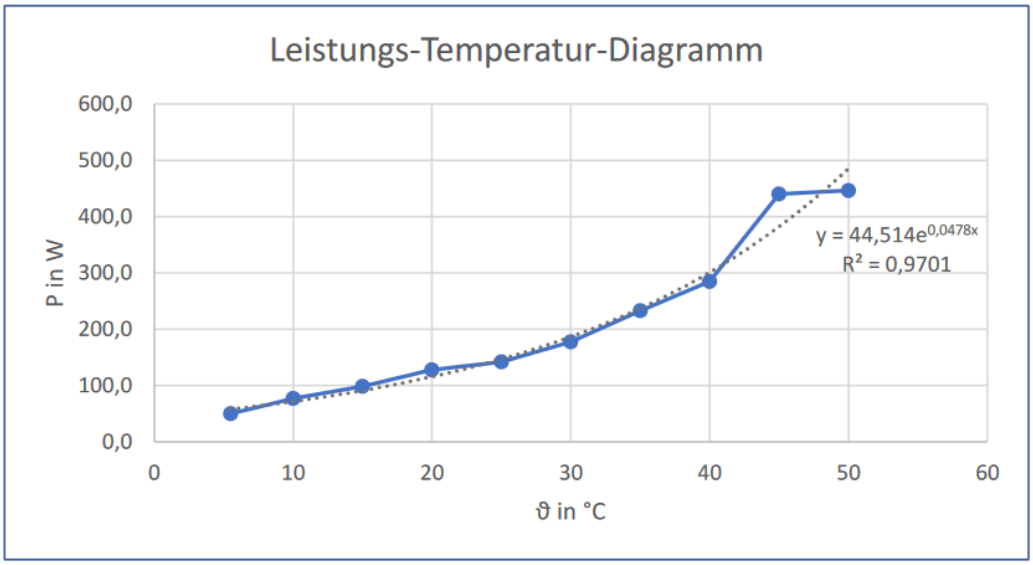
Durch die Messungen 1, 2 und 3 ergibt sich jeweils eine exponentielle Kurve im Temperatur-Zeit-Diagramm. Die Dauer einer Kühlung von 500 ml Wasser um ca. 50 K bis 55 K lag bei ca. 800 s, was rund 13 Minuten und 20 Sekunden entspricht. Es ist zu beobachten, dass die Temperatur am Anfang einer Kühlung stark fällt und diese Fallgeschwindigkeit mit fortschreitender Zeit abnimmt. Die Trendlinie folgt dem Verlauf der Messungen sehr exakt, was neben der optischen Auswertung der Wert der Varianz (R^2) zeigt. Dieser liegt bei allen Messungen bei mindestens rund 99,8 %. Das deutet auf genaue Messwerte und somit auf ein klar umgekehrt exponentielles Verhalten hin.

Die Kühlleistung des Gerätes (P) wurde mit der Formel $P = \frac{c \cdot m \cdot \Delta\vartheta \cdot 1000}{\Delta t}$ ermittelt. Dabei entspricht c der spezifischen Wärmekapazität von Wasser, also $4{,}19\,\frac{kJ}{kg \cdot K}$ (Duden, 2012, S. 14), während die 1000 die kJ in J umrechnen. „m“ ist gleich der Masse des Wassers in kg. $\Delta\vartheta$ in K ergibt sich aus dem vorherigen, höheren Wert der Temperatur minus der zum Messzeitpunkt aktuellen. 1 °C ist bei der nachträglichen Umrechnung gleich 1 K. Δt in Sekunden ergibt sich aus der zum Messzeitpunkt aktuellen Zeit minus der vorherigen.

Tabelle 7: Leistungsdiagramm

Durchschnittliche Leistungen

ϑ_{Mittel}	P_1	P_2	P_3	P_{Mittel}
50	-	427,4	465,6	446,5
45	473,8	427,4	419,0	440,1
40	326,3	267,1	261,9	285,1
35	229,1	237,4	232,8	233,1
30	179,5	178,1	174,6	177,4
25	143,6	142,5	139,7	141,9
20	126,7	125,7	130,9	127,8
15	107,7	97,1	91,1	98,6
10	76,9	76,3	77,6	76,9
6	46,6	48,1	55,5	50,0



Die Kühlleistung weist entsprechend der exponentiell fallenden Kurve im ϑ-t-Diagramm ebenso unterschiedliche Werte im Verlauf einer Kühlung auf. Die Werte liegen zwischen ca. 470 W und 47 W. Da die Kühlleistung der Peltiers laut Datenblatt in Abhängigkeit des Temperaturunterschiedes zwischen kalter und warmer Seite steht, und mit einem größeren Unterschied geringere Leistungen erzeugt, liegt der Schluss nahe, dass die Leistung hier abhängig von der Temperatur des Getränks ist.

Die Abbildung 28 zeigt ein Leistungs-Temperatur-Diagramm und seine dazu gehörige Wertetabelle.

Die Nummerierung der Leistungen in der Tabelle in der Abbildung entspricht der der Messreihen. Im Diagramm ist eine Abhängigkeit der Leistung von der Temperatur abzulesen. Laut Datenblatt sollte dieses Verhältnis annähernde Linearität besitzen, allerdings scheint in der Praxis ein gewisser exponentieller Anteil enthalten zu sein.

Dieser könnte von verschiedenen Einflussfaktoren abhängen, wie der Umgebungstemperatur, einer steigenden Temperatur der Peltier-Kühler und damit einem steigenden Temperaturunterschied zwischen warmer und kalter Seite der Peltiers, durch einen temperaturabhängig steigenden ohmschen Widerstand der Peltiers oder einer verminderten Gleichspannung des Netzteils durch dessen Temperaturanstieg. Diese Effekte könnten sich zum Verhalten der Peltiers teilweise addieren und teilweise multiplizieren.

Der etwas zu hoch scheinende Leistungswert bei 45 °C kommt möglicherweise teilweise von der Wärmekapazität der Kühlkörper. Wie vorher erwähnt, wurde deren Anfangstemperatur (ϑ_{KKk}) aufgezeichnet, welche in den Anlagen, Teil 8 über den Tabellen der Messreihen zu sehen ist. Dadurch, dass das Wasser zu Anfang eine wesentlich höhere Temperatur aufweist als die Kühlkörper, nehmen diese zu Beginn einer Kühlung einen Teil der Wärme in sich auf. Dies kann zum Nachteil werden, soll eine Flüssigkeit gekühlt werden, die vor Beginn der Kühlung kälter ist als die Kühlkörper.

Zusätzlich ist der Temperaturunterschied beider Seiten der Peltiers zu Beginn gleich null. Daher ist ihre Leistung entsprechend der Spannung maximal. Diese könnte in dem Moment maximal 130 W pro Peltier, also rund 500 W insgesamt betragen.

Letztendlich wird der Sprung der Leistung zu Beginn einer Kühlung allerdings hauptsächlich am ersten Wert der gemessenen Leistung der ersten Messreihe liegen, welcher unverhältnismäßig hoch erscheint.

Die durchschnittliche Minimalleistung von 50 W trägt weiterhin zu einer voranschreitenden Kühlung bei. Jedoch kann ein Kühlprozess aufgrund des exponentiellen Verhaltens durch die Wahl einer etwas höheren Zieltemperatur deutlich beschleunigt werden.

Zusatz zur Leistung des Gerätes:

Vor dem Umbau des Kühlmoduls auf die nun ausgewählten Peltiers, wurden sechs Peltiers anderer Bauart verwendet. Diese brauchten in einem Test über 16 Minuten um 228 ml Wasser von 43 °C auf 5 °C zu kühlen. Dies entsprach einer durchschnittlichen Gesamtleistung von ca. 30 W. Die durchschnittliche Gesamtleistung des Prototyps jetzt, mit seinen vier Peltiers, beträgt beim Kühlen von 500 ml Wasser von 45 °C auf 5 °C rund 116 W. Die Werte dazu sind der Messreihe 3 entnommen.

Ein Getränk mit einer Ausgangstemperatur von 25 °C (standardisierte Raumtemperatur) und einem Volumen von 300 ml (angenommene Größe eines handelsüblichen Trinkglases) auf 7 °C zu kühlen (Kaltgetränke-Programm des Gerätes) sollte laut der Messergebnisse etwas über vier Minuten lang dauern.

An den Messungen kann man zusätzlich beobachten, dass der Kühlvorgang scheinbar immer kurz vor Erreichen der eigentlichen Zieltemperatur beendet wird. Das kommt wahrscheinlich daher, dass der µC den Kühlvorgang beendet, sobald er einmal die Zieltemperatur im Getränk misst. Vermutlich liegt dies an der Hysterese des Gerätes, d. h. an der Trägheit der Temperaturen des Gefäßes, der Kühlkörper und der Peltiers, welche jeweils eine höhere Temperatur zum Abschaltungszeitpunkt besitzen, und diese Wärme dann an das Getränk zurückgeben. Schwankungen der 5V-Spannungsebene konnten als Ursache ausgeschlossen werden. Dieser Fehler wäre ohne die Korrektur der Zieltemperatur für Werte ab und unter 20 °C, um 1 °C nach unten, welche im Programmcode in den Anlagen, Teil 6 ab Zeile 280 zu finden ist, noch um 1 °C größer. Aufgrund der höheren Kühlleistung bei wärmeren Getränken wird dabei diese Korrektur nicht benötigt, da die Trägheit des Systems den Fehler teilweise ausgleicht.

9 Konklusion und Ausblick

Grundsätzlich wurden alle Ziele dieser Arbeit erreicht. Der Prototyp kann Getränke in beliebigen Gefäßen, je nach Volumen, schnell abkühlen. Zudem trifft er die Zieltemperatur auf ca. ein halbes Grad genau.

Wie in Kapitel 8 gezeigt, ist die Kühlleistung des Prototyps deutlich messbar. Der Prototyp ist aufgrund seines exponentiellen Leistungsverhaltens für Heißgetränke am besten geeignet. Dabei wird das Gerät bei der Kühlung von Heißgetränken im Vergleich zur Kühlung von Kaltgetränken voraussichtlich die fünf- bis achtfache durchschnittliche Leistung aufbringen als beim Kühlen von Kaltgetränken. Um auch die Kühlung von Kaltgetränken effektiver und effizienter zu machen, lohnt es sich, die Kühlung der warmen Seite der Peltiers bestmöglich zu verbessern. Hierfür könnten eine Wasser- oder andere Flüssigkeitskühlung mit einer gut wärmeleitenden Flüssigkeit und aktiven Radiatoren, oder mehr und größere Heatpipes, die Lösung sein.

Nichtsdestotrotz funktioniert auch das Kühlen bis hin zu niedrigeren Temperaturen gut. In der Praxis lohnt es sich bei diesem Gerät mehr, mehrfach kleinere Volumen eines Getränks zu kühlen, statt eines großen Volumens, da diese dann jeweils sofort verzehrt werden können, wobei große Volumen ihrerseits über die Zeit aus der Umgebung wieder Wärme aufnehmen, werden sie nicht zügig konsumiert.

Die Bedienung des Getränkekühlers ist, subjektiv gesehen, recht simpel und intuitiv. Das Display könnte etwas größer sein, denn gerade für Menschen mit breiteren Fingern ist es etwas zu klein. Hierzu würde sich in Zukunft eine größere Umfrage oder weitere Recherche lohnen. Allgemein funktioniert der Prototyp technisch einwandfrei, denn bis zur Abgabe dieser Arbeit traten keine weiteren Probleme auf.

Zusätzlich könnten Systemeinstellungen implementiert werden. Hier könnte man die Zieltemperaturen des Kalt- und Heißgetränkeprogramms modifizieren, sowie ggf. den Hintergrund. Zusätzlich könnte hier eine Statistik über die Nutzung des Gerätes geführt werden. In den Einstellungen könnte man dann eine Funktion zur Wiederherstellung des Werkszustandes des Gerätes implementieren. Sowohl der µC, als auch das Display, haben für solche Erweiterungen noch genügend Kapazitäten.

In einer zweiten Version könnte man außerdem das Gehäuse aus einem Blech aus Edelstahl fertigen, was der Langlebigkeit und Optik des finalen Produkts zu Gute kommen würde. Zudem könnte man bei zuvor stark verbesserter Kühlung der Peltiers, noch leis-

tungsstärkere Varianten einbauen. Unter Verwendung einer Wasserkühlung könnte das Gerät asymmetrisch aufgebaut werden.

So könnte z. Bsp. auf der linken Seite ein Kühlkörper für die Getränke liegen, an welchem auf beiden Seiten Peltiers angebracht sind, an deren warmen Seiten jeweils Anschlüsse für die Wasserkühlung angebracht wären. Nur einen Kühlkörper zu verwenden würde die Flüssigkeits-Rückstände im Leitungssystem drastisch verringern, da die Flüssigkeit im aktuellen Aufbau einen Höhenunterschied überwinden muss, um von einem in den anderen Kühlkörper zu gelangen. Dadurch sammelt sich etwas Flüssigkeit vor diesem Übergang. Weiterhin könnte man in der rechten Seite des Gerätes die aktiven Radiatoren platzieren. Das würde das Gerät deutlich kompakter machen.

Unter Verwendung einer Leistungselektronik, welche höhere Spannungen (z. Bsp. über 100 V) reguliert, könnte man alle vier, oder mehr, Peltiers in Reihe schalten. Der µC müsste dann den Strom, welcher durch die Peltiers fließt, messen und entsprechend durch Regulierung der Spannung über die Leistungselektronik konstant halten. Das würde eine sehr viel höhere und konstantere Leistung ermöglichen und das Potential der Peltiers voll ausschöpfen. Eine solche Leistungselektronik wäre zudem wahrscheinlich etwas kompakter als das 24V-Netzteil, wodurch das Gerät etwas flacher konstruiert werden könnte.

Zudem würde dieser Aufbau eine Regelung der Leistung, z. Bsp. mittels eines PID-Reglers ermöglichen, da der µC die Leistung der Peltiers über die angelegte Spannung stufenlos und ganzheitlich regulieren könnte. Somit könnte das Gerät insgesamt schneller und präziser die Zieltemperaturen erreichen. Momentan wäre eine Regelung aufgrund des Aufbaus ungenauer als die implementierte Steuerung, und somit auch unnötig aufwändig.

Über die Messung der Temperaturveränderung des Getränks über die Zeit, und die Messung von Stromstärke und Spannung, könnte während eines Kühlvorgangs über die berechnete, aktuelle Kühlleistung das Volumen des Getränks approximiert, und somit der Regelungsprozess optimiert werden.

Darüber hinaus könnten die Schläuche flexibler und einfahrbar sein. Das würde dem Nutzer einen maximalen Nutzungskomfort bieten, da er nicht manuell die Schläuche in das Gefäß mit dem Getränk führen, sondern sein Gefäß nur platzieren und sein gewünschtes Programm starten müsste.

Ein Problem der Arbeitsweise des Gerätes ist, dass der Großteil an vorhandener Kohlensäure eines Getränks während des Prozesses verloren geht. Aus diesem Grund wurde bisher bewusst keine Kühlung von Bieren untersucht, da wahrscheinlich das Gefäß des Bieres während einer Kühlung vor Schaum sofort überlaufen würde. Hierfür wurde noch kein Lösungsansatz gefunden.

Optimal eignet sich der Prototyp momentan zur schnellen Konvertierung von frisch aufgebrühten Tees in Eistee. Ich selbst war davon überrascht, wie neu und einzigartig sich diese Erfahrung anfühlt. Ob das als Verkaufsargument reicht, ist allerdings zweifelhaft. Außerdem muss für diesen Prozess der Tee meistens zuvor gefiltert werden, vor allem ist für sein Aufgießen ein Tee-Ei genutzt worden. Darum könnte eine zweite Version ein, möglicherweise selbstreinigendes, Filtersystem besitzen, um gröbere Partikel aus dem Getränk zu filtern.

Zusätzlich könnte die Fördergeschwindigkeit der Pumpe regulierbar gemacht und erhöht werden. Somit könnte ein Modus realisiert werden, bei welchem der Getränkekühler vor dem Start eines Kühlvorgangs den Kühlkörper unter 0 °C kühlt, um den Start der Kühlung zu beschleunigen. Diese Vorkühlung hätte allerdings eine etwas verminderte Effizienz zufolge.

Literatur

Bähring, H. (2002). *Mikrorechner-Technik.* Berlin: Springer Verlag.

Beev, N. (02. September 2022). *Wikipedia*. Von https://de.wikipedia.org/wiki/Peltier-Element abgerufen

Bonfig, K. W. (1995). *Temperatursensoren.* Renningen-Malmsheim: Expert Verlag.

Brusch, M. (20. 07 2016). Abgerufen am 26. 09 22 von Berliner Morgenpost: https://www.morgenpost.de/vermischtes/article207916213/Cola-Etiketten-melden-richtige-Temperatur-mit-Farbwechsel.html

Cornelsen Verlag GmbH. (10. Mai 2023). *DUDEN*. Von https://www.duden.de/rechtschreibung/Ventil abgerufen

Cyberstore GmbH. (28. 09 2012). Abgerufen am 26. 09 22 von Craftbeer.de: https://www.craftbeer.de/wissen/bei-welcher-temperatur-sollte-ein-bier-serviert-werden/

Drebushachak, V. A. (2008). The Peltier Effect. *Journal of Thermal Analysis und Calorimetry, Vol. 91*, 311-315.

Duden. (2012). *Formeln und Tabellen.* Mannheim: Bibliographisches Institut GmbH.

Erwin Baur, G. H. (2019). *Werkstoff-Führer Kunststoffe.* München: Karl-Hanser-Verlag.

Farhad Islami, P. B.-S. (2009). High-temperature beverages and foods and esophageal cancer risk—A systematic review. *IJC*, 524.

Filamentworld. (21. Mai 2023). *filamentworld*. Von https://www.filamentworld.de/3d-druck-wissen/was-ist-petg/ abgerufen

Gebhardt, A. (2016). *Additive Fertigungsverfahren.* München: Hanser Verlag.

Grabavac, D. (2011). Kühlung bei LED-Projektoren: Ein Vergleich von zwei Kühlsystemen. *Informatics Inside*, 6.

HANSER Verlag. (20. Mai 2023). *Kunststoffe.de*. Von https://www.kunststoffe.de/a/grundlagenartikel/polytetrafluorethylen-ptfe-285527 abgerufen

Hoffmann, C. (1956). *Lehrbuch der Bergwerksmaschinen.* Berlin Heidelberg: Springer Verlag.

Huag, A. &. (1991). *Angewandte Elektrische Messtechnik.* Braunschweig: Friedr. Vieweg & Sohn Verlagsgesellschaft mbH.

Juno, T. (06. März 2020). *Indiegogo*. Von https://www.indiegogo.com/projects/juno-like-a-microwave-for-cooling#/ abgerufen

Laboranalyse24. (20. April 2022). Von https://www.laboranalyse24.de/info/kupfer-wasserleitung/ abgerufen

Litzius, K. (Februar 2017). Wie funktioniert eigentlich ein Touchscreen? *Chemie in unserer Zeit, Folge 51*, S. 71.

Müller, J. (2007). Eine gemeinsame Basis für Pumpen. *Asset Management*. Herzogenrath, Deutschland: LeiKon GmbH.

Reichelt Elektronik. (20. Mai 2023). *Reichelt Elektronik*. Von https://www.reichelt.de/kfz-thermoelektrischer-getraenkehalter-12-v--kfz-20188-p323129.html?PROVID=2788&gclid=Cj0KCQjwyLGjBhDKARIsAFRNgW_Ex6ySgF6dVe5AOY6c9H6XJsJEJ_69lcL7a8UJ7yssy9S_xl08wZcaAj5EEALw_wcB abgerufen

Rosenkranz, A. (02. Februar 2020). *heizung.de*. Von https://www.heizung.de/ratgeber/diverses/ntc-widerstand-eigenschaften-und-anwendung.html abgerufen

Schaumburg, H. (1992). *Sensoren.* Stuttgart: B. G. Teubner.

Schnabel, P. (2017). *Elektronik-Fibel.* Ludwigsburg: Elektronik-Kompendium.

Schulz, H. (1955). *Die Pumpen.* Berlin Heidelberg: Springer-Verlag.

Sieperman, D. M. (19.. Februar 2018). *Gabler Wirtschaftslexikon*. Von https://wirtschaftslexikon.gabler.de/definition/touch-screen-49915/version-273141 abgerufen

VIRTUALEXPO GROUP. (12. Mai 2023). *Direct Industry*. Von https://guide.directindustry.com/de/__trashed-6/ abgerufen

W. M. Yim, F. D. (1972). Compund tellurides and their alloys for Peltier cooling - a review. *Solid-State Electronics, Vol. 15*, 1121 - 1140.

Anlagen

Anlagen, Teil 1 63

Freilaufdioden 63

Schaltplan 64

Anlagen, Teil 2 65

Leiterplattenlayout 65

Anlagen, Teil 3 66

Komponentenanordnung 66

Gehäuse 67

Fuß 68

Mittelteil 68

Kopf Hauptteil 69

Kopf Vorderteil 69

Anlagen, Teil 4 70

Montage des Kühlmoduls 70

Anlagen, Teil 5 72

Programmablaufplan – Kühlprogramm 72

Programmablaufplan – Hauptprogramm 73

Programmablaufplan – Spülung 74

Programmablaufplan – Reinigung 74

Programmablaufplan – Nachspülung 74

Programmablaufplan – Leeren 75

Programmablaufplan – Wartung 75

Programmablaufplan – Temperatur 75

Anlagen, Teil 6 76

FrigidusPotus.h – Header 76

FrigidusPotus.h *77*

Programmcode *79*

Anlagen, Teil 7 ..... **86**

Display-Seiten *86*

Editor *90*

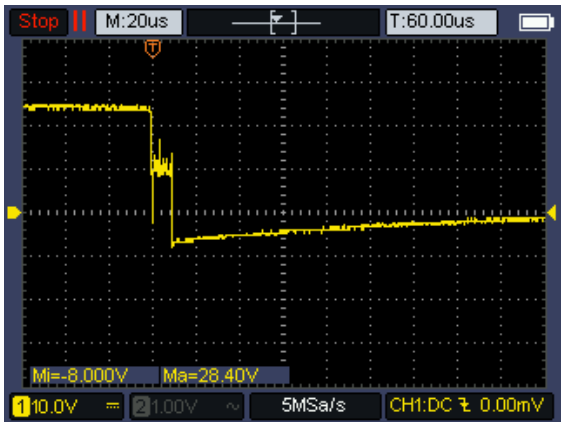
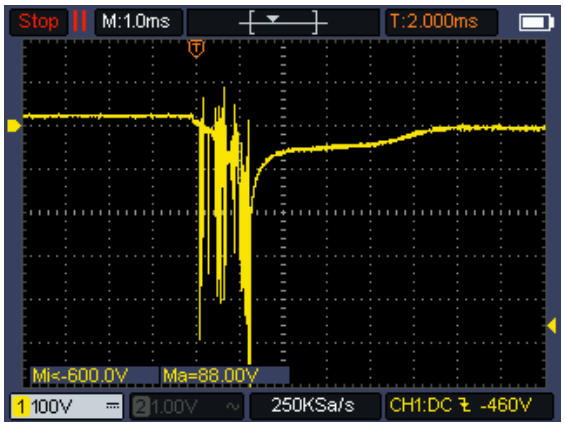
Anlagen, Teil 8 ..... **91**

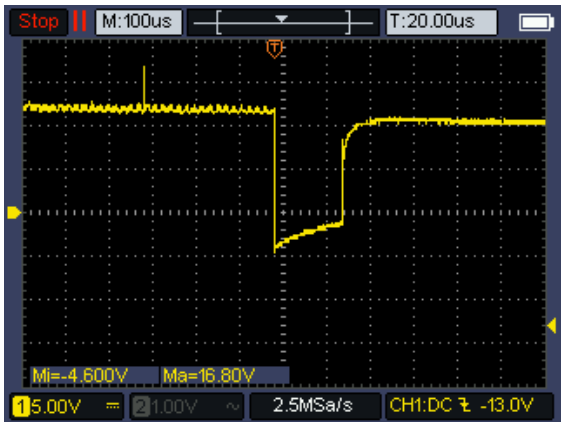
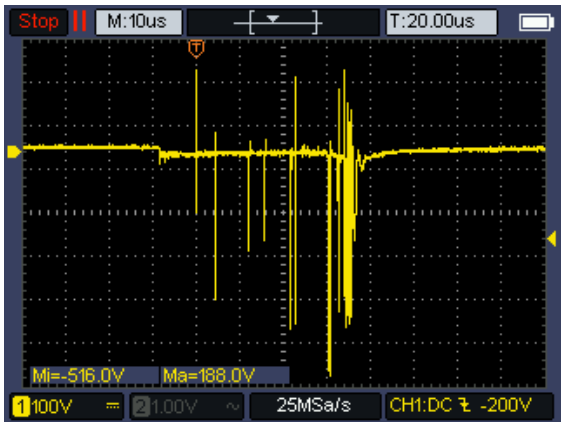
Versuchsaufbau *91*

Messungen *92*

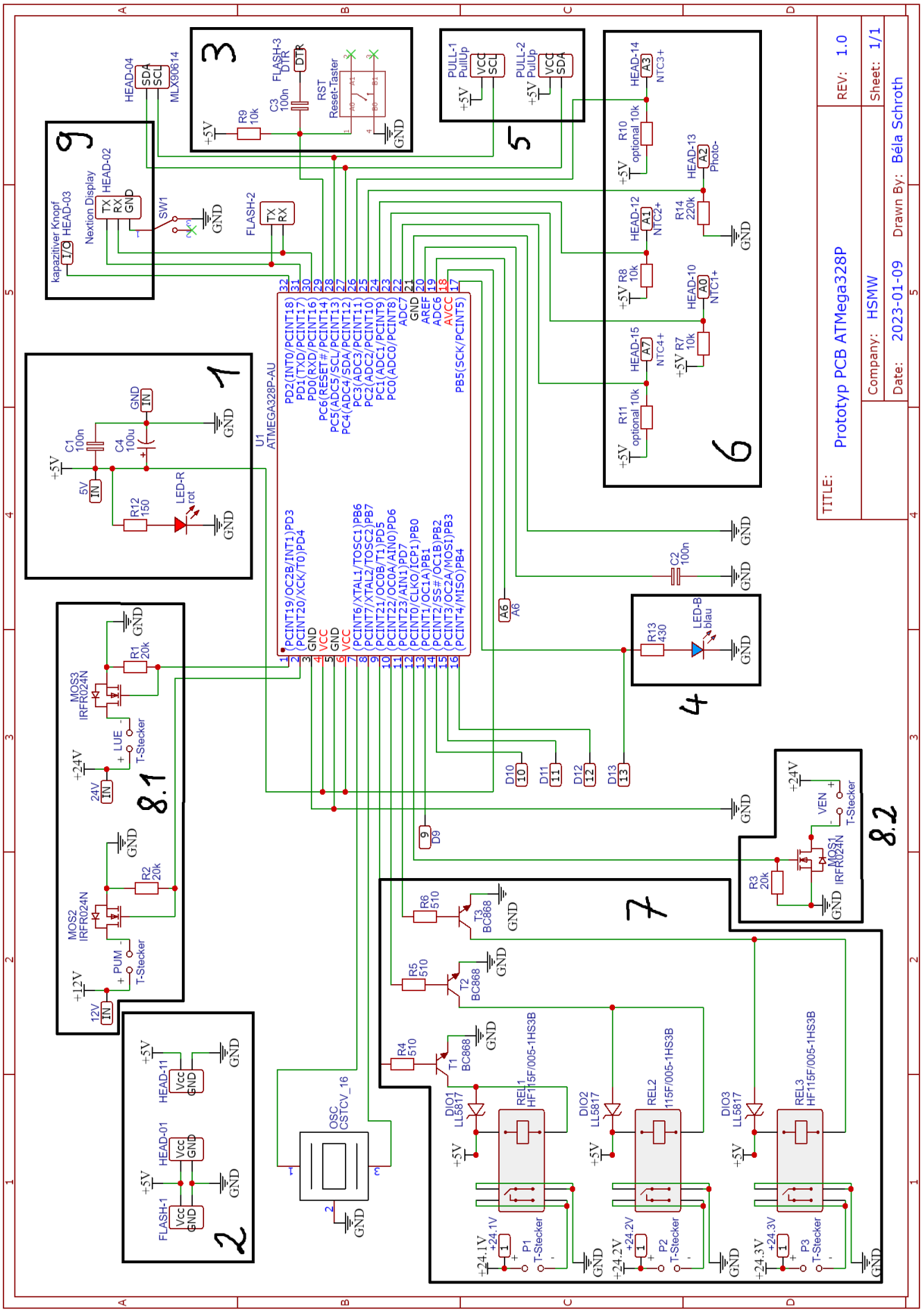
Anlagen, Teil 1

Freilaufdioden

	
Ventil mit Freilaufdiode	Ventil ohne Freilaufdiode

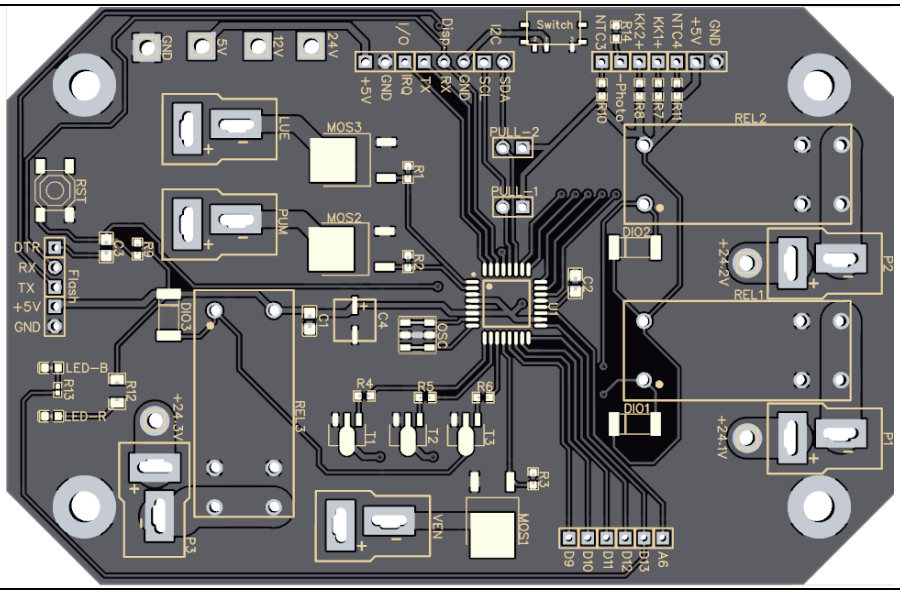
	
Pumpe mit Freilaufdiode	Pumpe ohne Freilaufdiode

Schaltplan

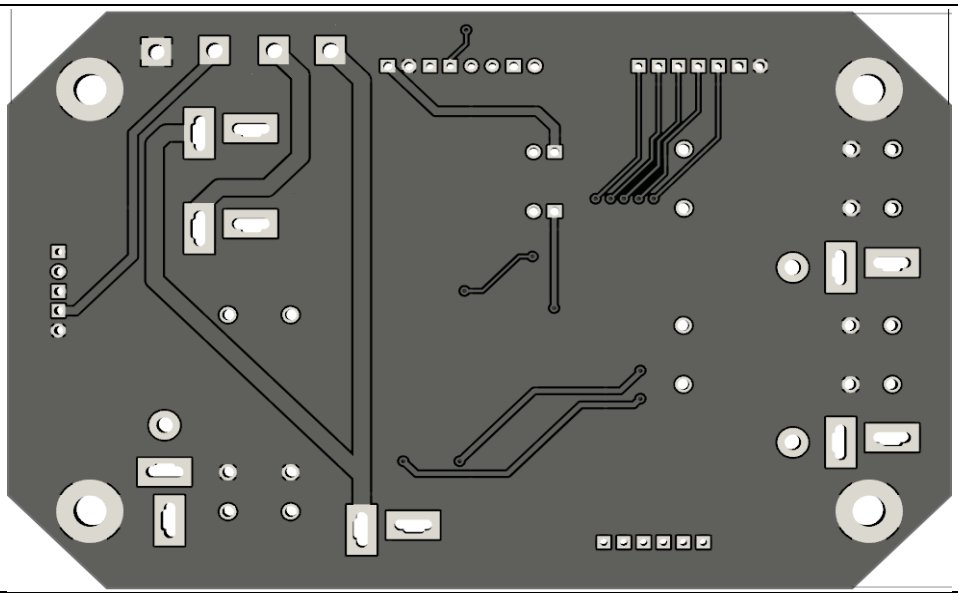

TITLE: Prototyp PCB ATMega328P
REV: 1.0
Company: HSMW
Sheet: 1/1
Date: 2023-01-09
Drawn By: Béla Schroth
U1 ATMEGA328P-AU
OSC CSTCV_16
HEAD-04 MLX90614
HEAD-03 Nextion Display
HEAD-02
HEAD-11
HEAD-01
HEAD-14 NTC3+
HEAD-13 Photo-
HEAD-12 NTC2+
HEAD-10 NTC1+
HEAD-15 NTC4+
FLASH-1
FLASH-2
FLASH-3
PULL-1 PullUp
PULL-2 PullUp
SW1
RST Reset-Taster
LED-R rot
LED-B blau
MOS1 IRFR024N
MOS2 IRFR024N
MOS3 IRFR024N
P1 T-Stecker
P2 T-Stecker
P3 T-Stecker
LUE T-Stecker
PUM T-Stecker
VEN T-Stecker
REL1 HF115F/005-1HS3B
REL2 115F/005-1HS3B
REL3 HF115F/005-1HS3B
DIO1 LL5817
DIO2 LL5817
DIO3 LL5817
T1 BC868
T2 BC868
T3 BC868
kapazitiver Knopf
8.1
8.2

Anlagen, Teil 2

Leiterplattenlayout



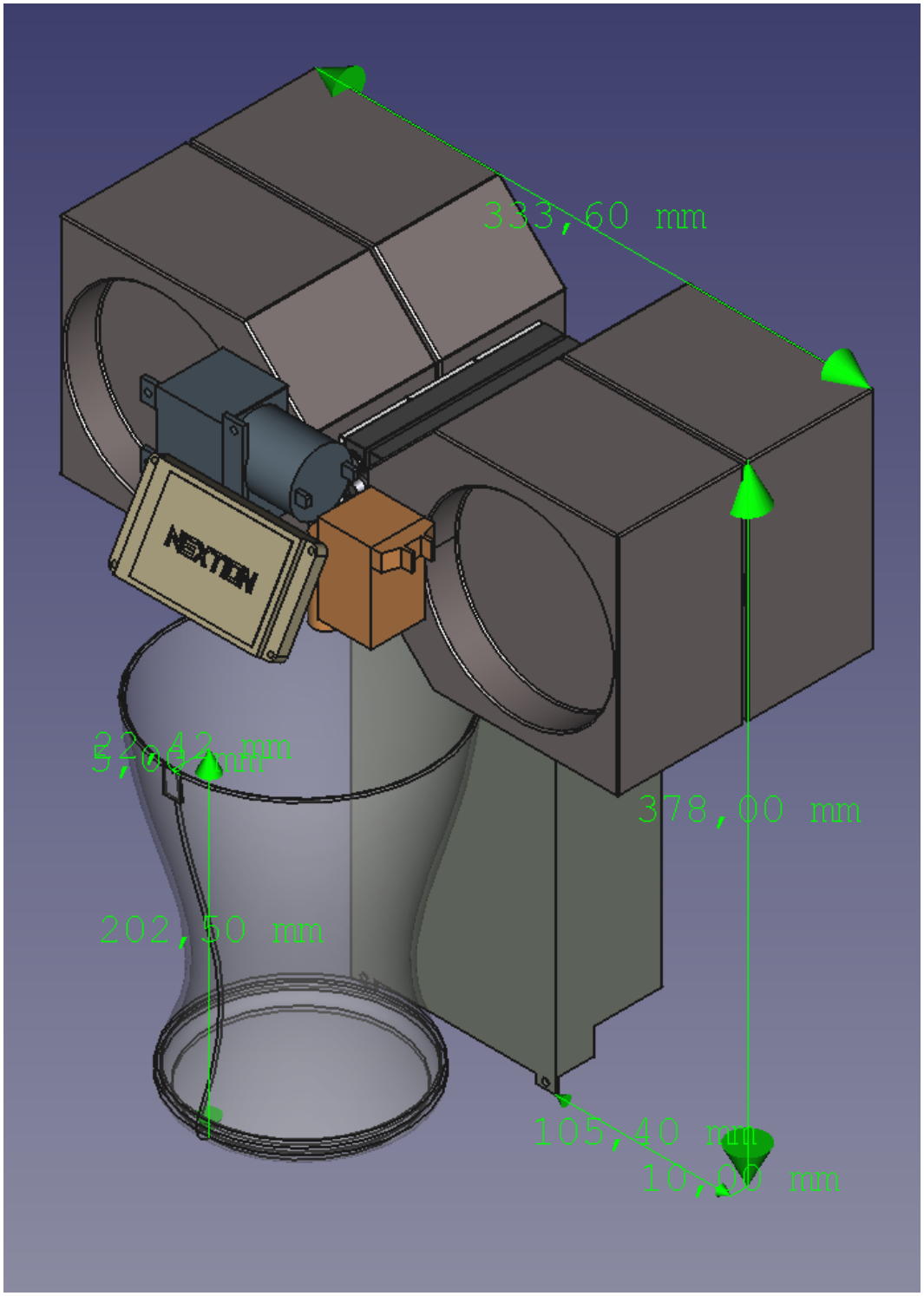
Leiterplatte Oberseite



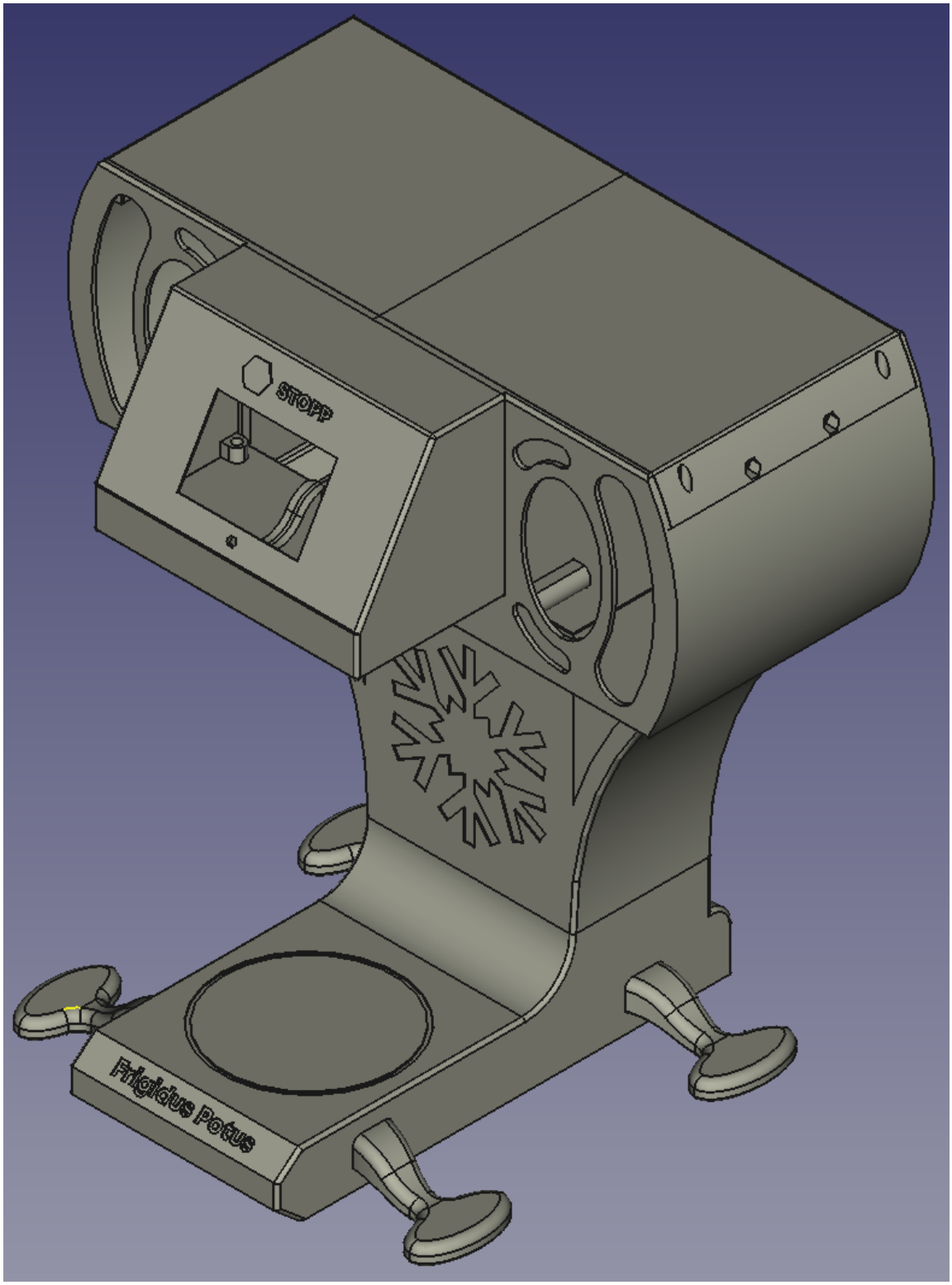
Leiterplatte Unterseite

Anlagen, Teil 3

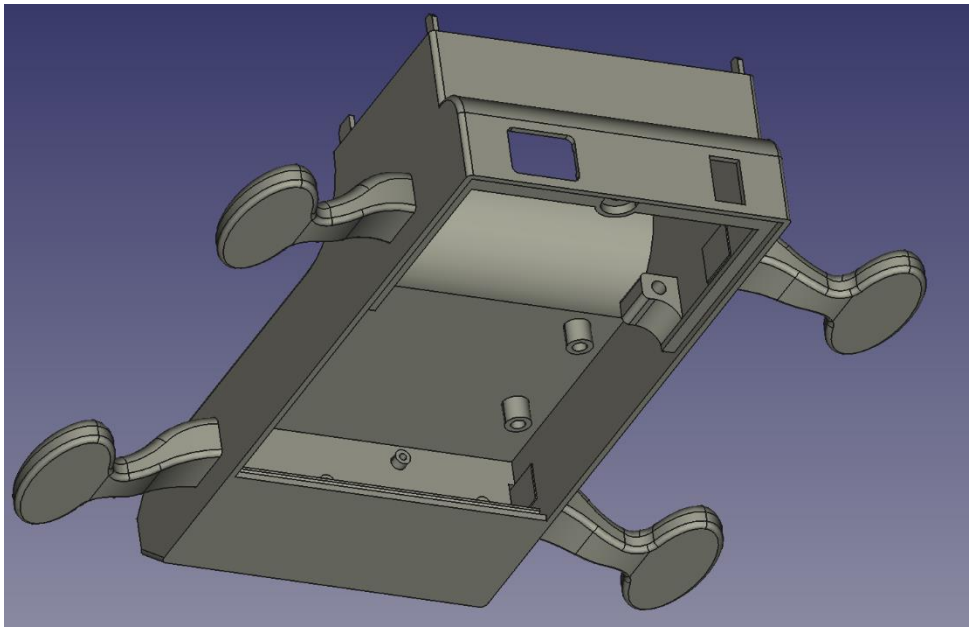
Komponentenanordnung



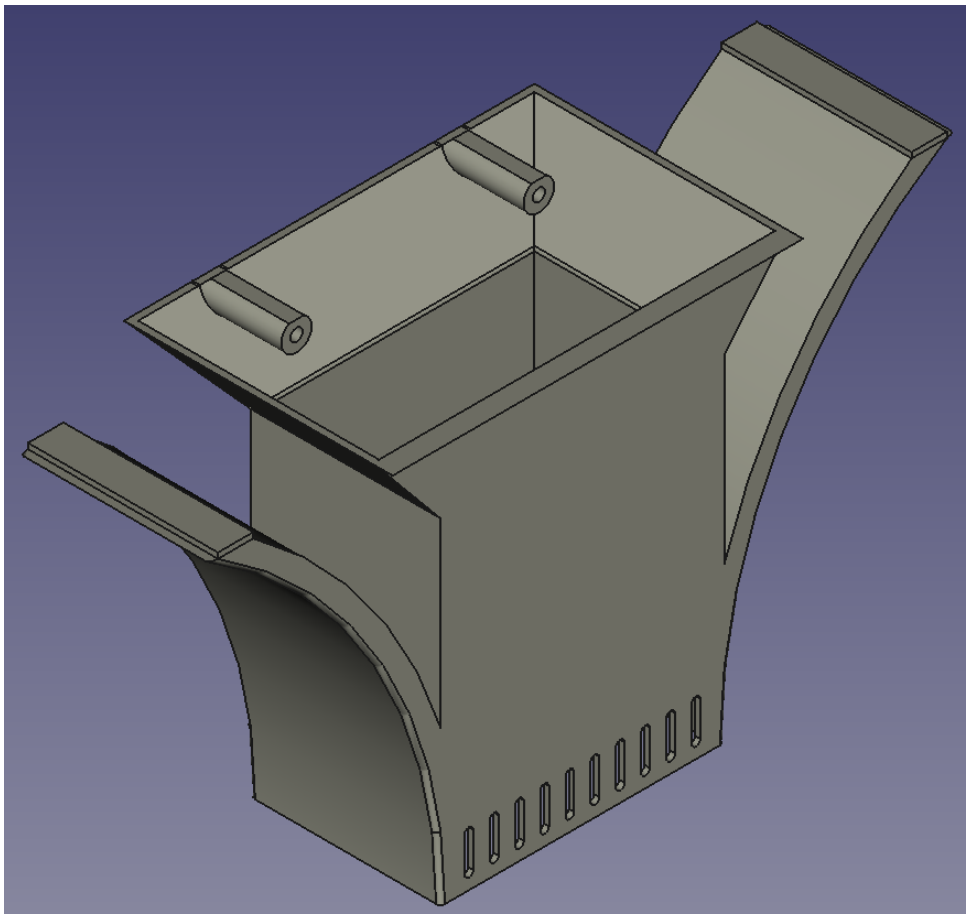
Gehäuse



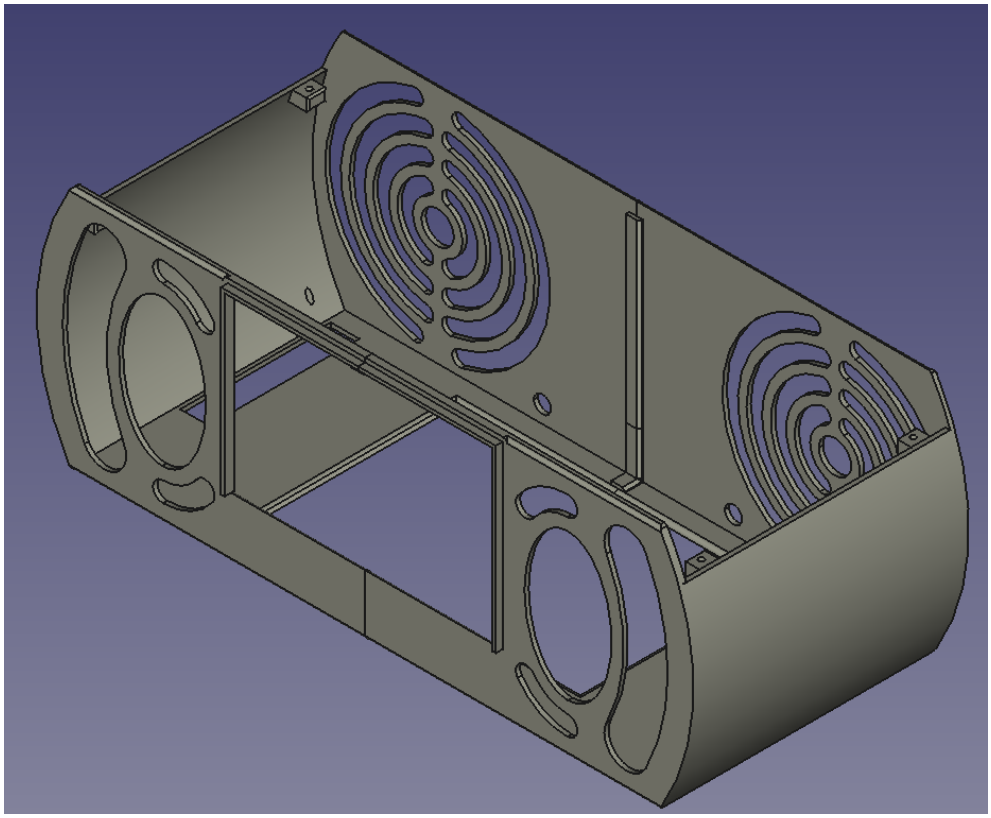
Fuß



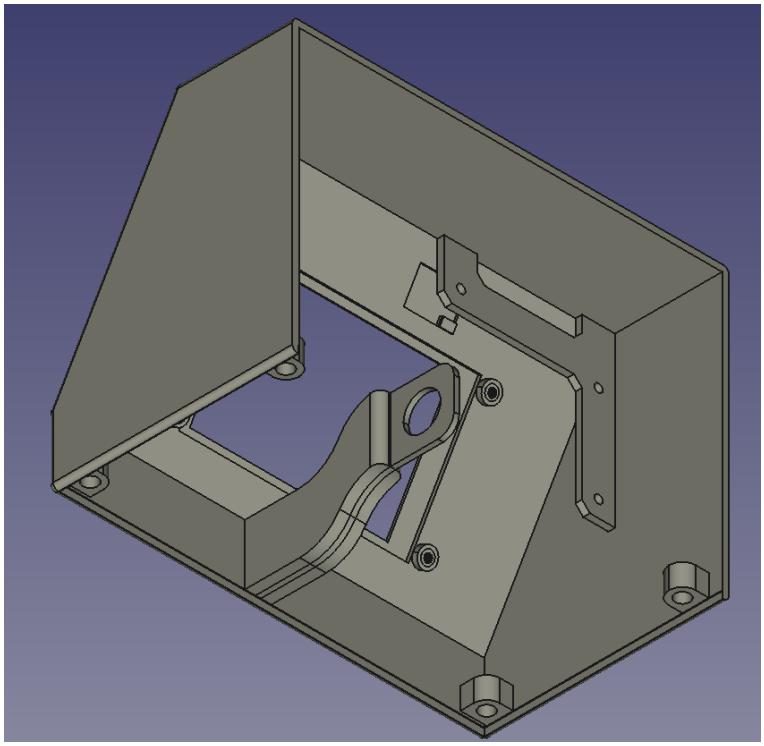
Mittelteil



Kopf Hauptteil

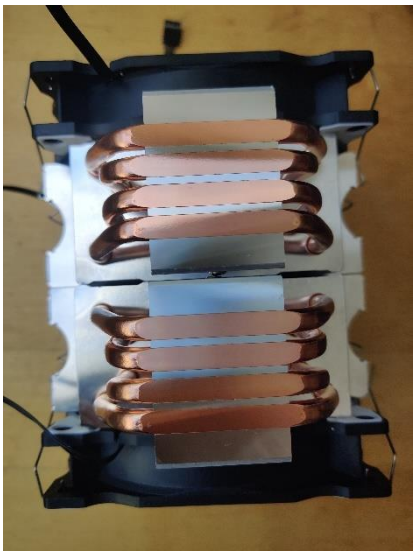


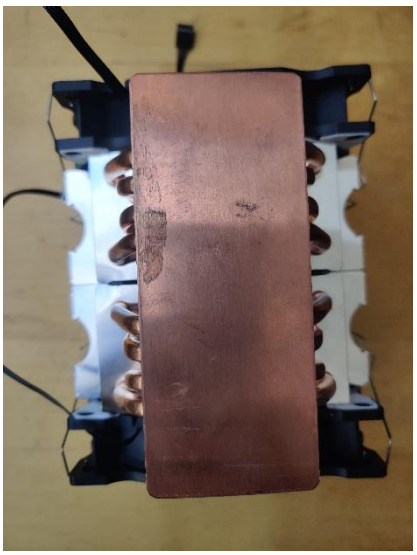
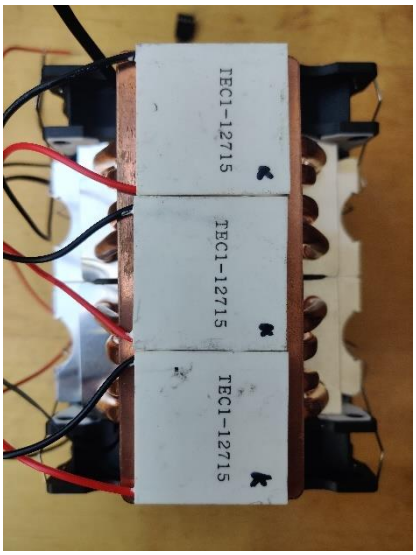
Kopf Vorderteil

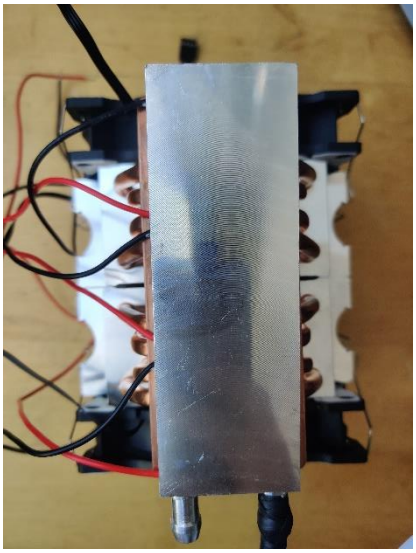
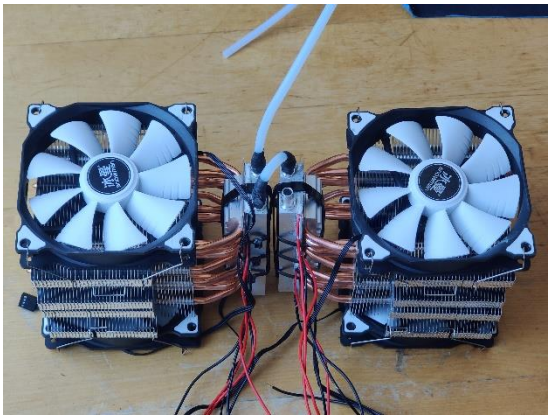


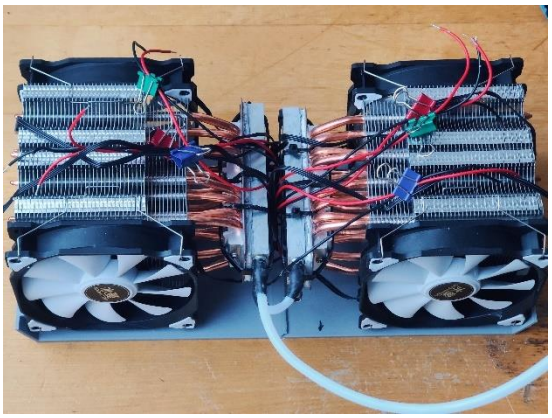
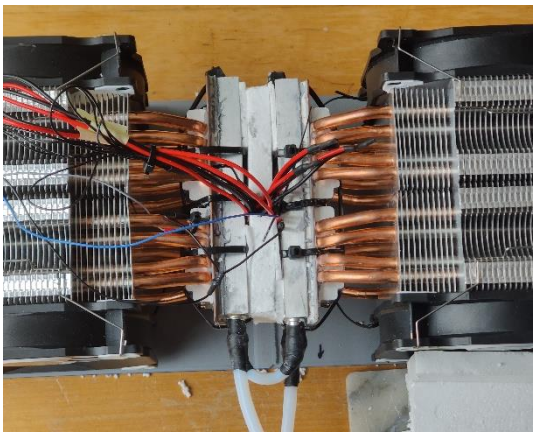
Anlagen, Teil 4

Montage des Kühlmoduls

	
01: Leitungen	02: Lüfter

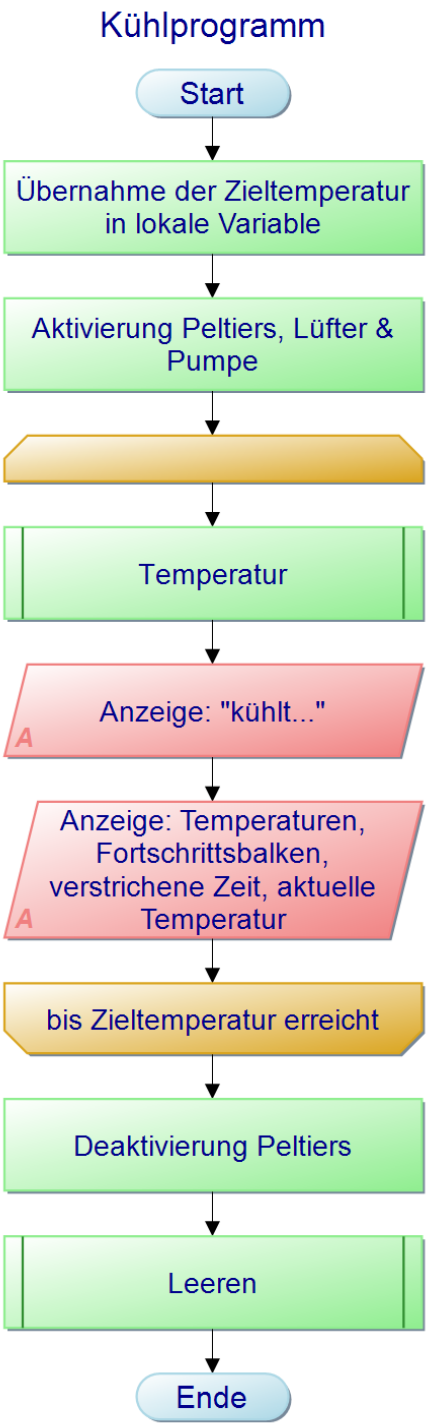
	
03: Kupferblech	04: Peltiers

	
05: Kühlkörper	06: Kühlmodul

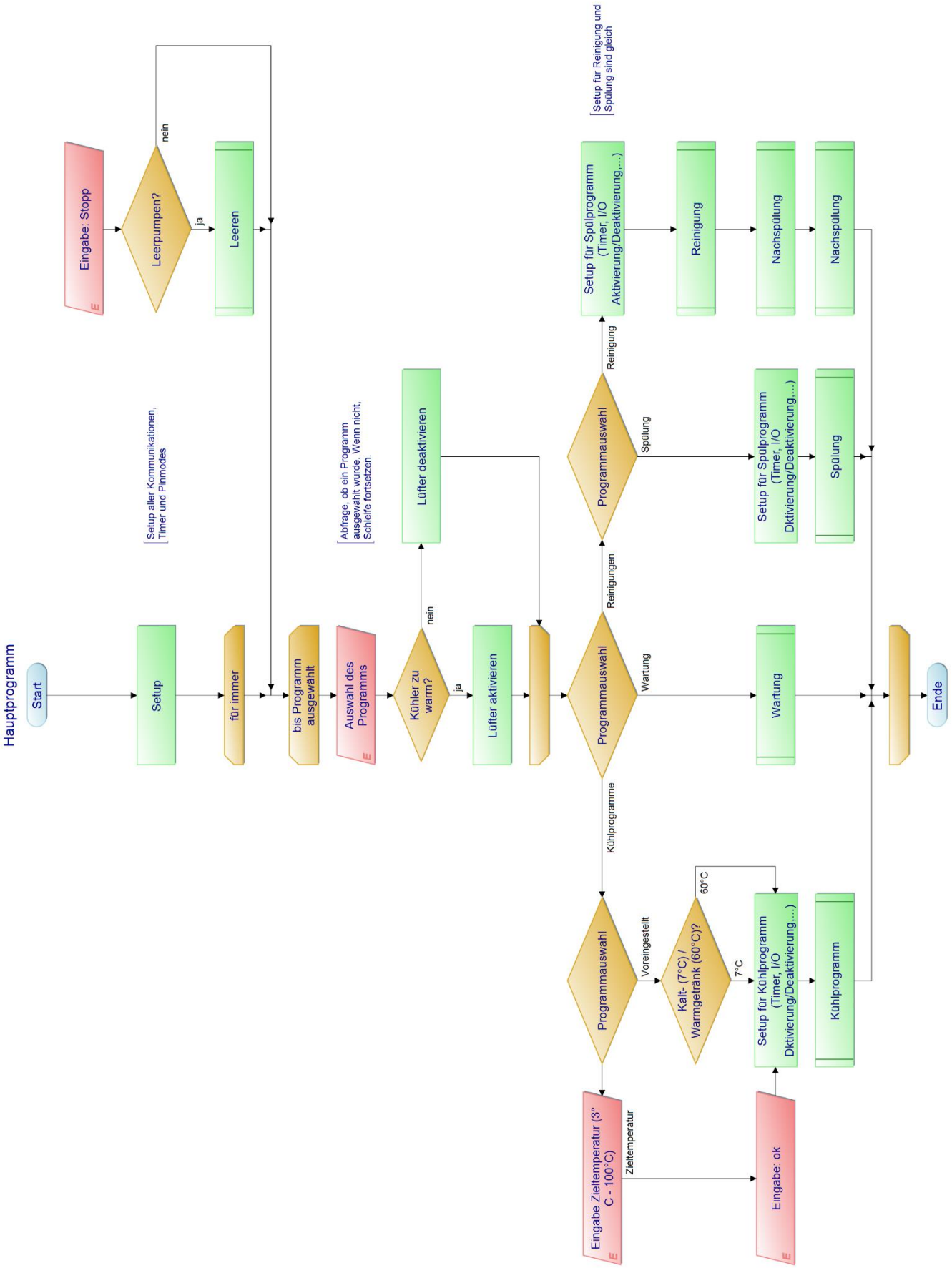
	
07: Klappe	08: Isolierung

Anlagen, Teil 5

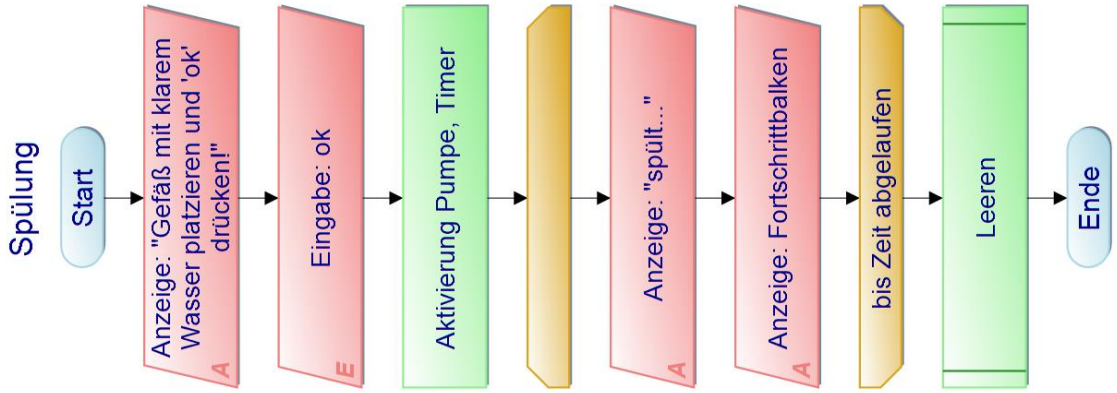
Programmablaufplan – Kühlprogramm



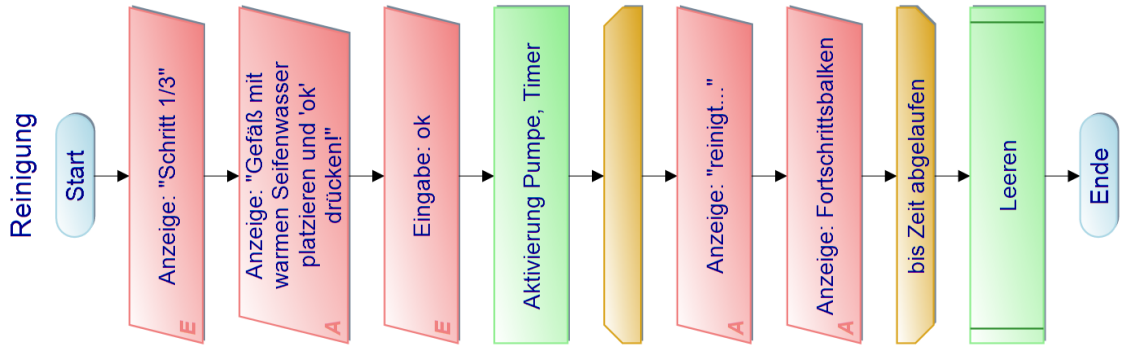
Programmablaufplan – Hauptprogramm



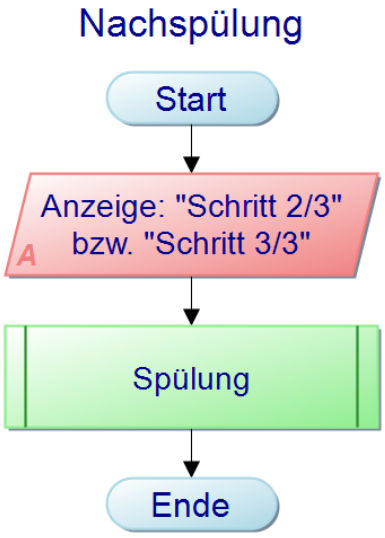
Programmablaufplan – Spülung



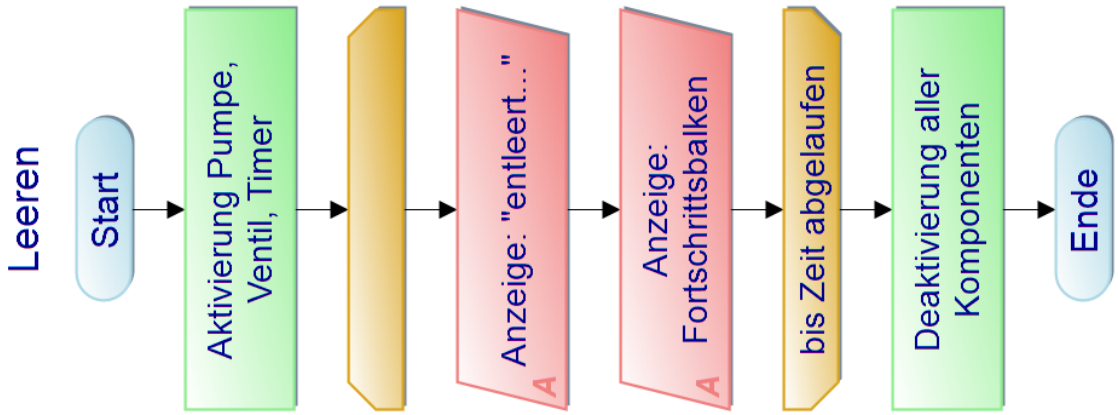
Programmablaufplan – Reinigung



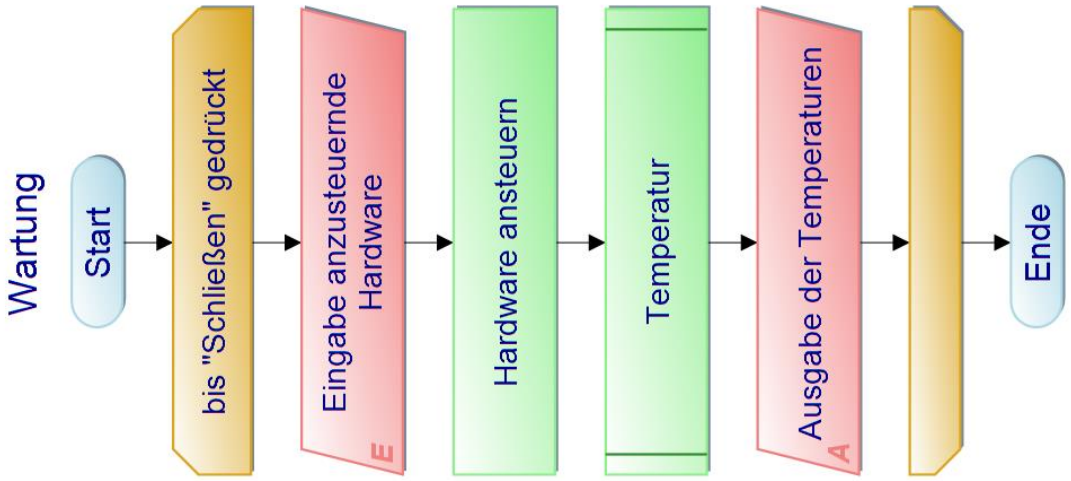
Programmablaufplan – Nachspülung



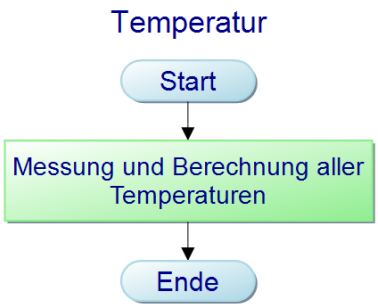
Programmablaufplan – Leeren



Programmablaufplan – Wartung



Programmablaufplan – Temperatur



Anlagen, Teil 6

FrigidusPotus.h – Header

```
#ifndef FrigidusPotus_H
#define FrigidusPotus_H

#include <Arduino.h>

uint8_t StatusLED(uint8_t LEDcount, uint8_t StatLED);
uint8_t Helligkeit(uint8_t PhotoR, uint8_t Hell);
int TempMessungKK(uint8_t Pin);
int TempMessungGetr(uint8_t Pin);
void TempAnzeige(int Temp_KKh, int Temp_KKk, int Temp_Getr);
void KuehlAnzeige(int Temp_Getr, int Minuten, int Sekunden, int Fortschritt);
void HellAnzeige(int Hell);
void Timer1Toggle();
void Timer2Toggle();
void EndTrans();

//Funktionsaufrufe (Beispiele)
  /*
  LEDcount = StatusLED(LEDcount, StatLED);
  Hell = Helligkeit(PhotoR, Hell);
  int Temp_KKh = TempMessungKK(NTC_KKh);
  int Temp_Getr = TempMessungGetr(NTC_Getr);
  TempAnzeige(Temp_KKh, Temp_KKk, Temp_Getr);
  HellAnzeige(Hell);
  Timer1Toggle();
  */

#endif
```

FrigidusPotus.h

```
#include "Arduino.h"
#include "FrigidusPotus.h"

//Ende einer jeden Datenübertragung zum Display
void EndTrans(){
  Serial.write(0xFF);
  Serial.write(0xFF);
  Serial.write(0xFF);
}

//An- / Ausschalten des Timer1
void Timer1Toggle(){
  TIMSK1 ^= (1 << 1);                       //Bit 1 im TIMSK1 wird geflippt (Interrupt an/ aus)
  TCNT1 = 0;                                //Timer auf 0 setzen
}

//Messung Temperatur Kühlkörper
int TempMessungKK(uint8_t Pin){
  int val = analogRead(Pin);                                        //Einlesen des Analogwertes 0 - 1023
  float VerhR = val / (1023.0 - val);                               //Berechnung des Verhältnis R / Rref
  float Temp_KKh = pow((3354.016 + 256.985 * log(VerhR) + 2.620131 *  //Berechnung von T in °C durch Formel
  log(VerhR) * log(VerhR) + 0.06383091 * log(VerhR) * log(VerhR)      //  aus Datenblatt und Erweiterung mit
  * log(VerhR)) / 1000000, -1) - 273.15;                            //  1Mio für Variablenbereich von float
  if((Temp_KKh-(int)Temp_KKh)*10 >= 5){                             //runden
    Temp_KKh++;
  }
  return (int)Temp_KKh;
}

//Messung Temperatur Getränk
int TempMessungGetr(uint8_t Pin){
  int RN = 10000;                                                   //2. Widerstand im Spannungsteiler
  float val = analogRead(Pin);
  float RT = RN * (val / 1023) / (1 - (val / 1023));                //Berechnung Widerstand
  float Temp_Getr = pow(log(RT / RN) / 3988.0 + (1.0 / 299.15), -1) - 274.15;   //Berechnung mit Konstante B25/100
  if((Temp_Getr-(int)Temp_Getr)*10 >= 5){                           //runden
    Temp_Getr++;
  }
  return (int)Temp_Getr;
}

//Anzeige der Temperaturen
void TempAnzeige(int Temp_KKh, int Temp_KKk, int Temp_Getr){
  Serial.print("nKKh.val=");
  Serial.print(Temp_KKh);
  EndTrans();
  Serial.print("nKKk.val=");
  Serial.print(Temp_KKk);
  EndTrans();
  Serial.print("nGetr.val=");
  Serial.print(Temp_Getr);
  EndTrans();
}

//Anzeige aller Werte im Kühlprogramm
void KuehlAnzeige(int Temp_Getr, int Minuten, int Sekunden, int Fortschritt){
  EndTrans();         //Workaround, ohne scheint das Display den Wert nicht anzunehmen
  Serial.print("nTempGetr.val=");   //Anzeige aller Werte auf Display
  Serial.print(Temp_Getr);
  EndTrans();
  Serial.print("nMinuten.val=");
  Serial.print(Minuten);
  EndTrans();
  Serial.print("nSekunden.val=");
  Serial.print(Sekunden);
  EndTrans();
  Serial.print("jFortschritt.val=");
  Serial.print(Fortschritt);
  EndTrans();
}
```

```
//Ermittlung der benötigten Displayhelligkeit
uint8_t Helligkeit(uint8_t PhotoR, uint8_t Hell){
  int val = analogRead(PhotoR);                    //Einlesen des Photowiderstandes (invertiert)
  float Hell_neu = 0.096 * val + 3.0;              //Berechnung der Helligkeit nach linearer Fkt
  if((Hell_neu - (int)Hell_neu) * 10 >= 5)         //mathematisch korrektes Runden
    Hell_neu += 1;
  if(abs(Hell - Hell_neu) >= 5){            //Änderungshysterese -> nur bei Änderung der Helligkeit
    return Hell_neu;                        //  ab 5% wird diese tatsächlich geändert
  }else{
    return Hell;
  }
}

//Übermittlung der Helligkeit an Display
void HellAnzeige(int Hell){
  uint8_t hell_min = 5;                                  //Einhaltung der Grenzen des
  if(Hell > 100)                                         //  Helligkeitswerts
    Hell = 100;
  if(Hell < hell_min)
    Hell = hell_min;
  Serial.print("dim=");                                   //Hintergrundbeleuchtung
  Serial.print(Hell);                                    //  (Bildschirmhelligkeit)
  EndTrans();
}
```

```
//Blinken der Status LED (Wird genutzt um die Hardware zu debuggen und die Funktion des uC zu prüfen.)
uint8_t StatusLED(uint8_t LEDcount, uint8_t StatLED){
    switch (LEDcount){
      case 0:
          digitalWrite(StatLED, LOW);
          LEDcount++;
       break;

      case 1:
          digitalWrite(StatLED, HIGH);
          LEDcount++;
       break;

      case 2:
          digitalWrite(StatLED, LOW);
          LEDcount++;
       break;

      case 3:
        LEDcount++;
        break;
      case 4:
        LEDcount++;
        break;
      case 5:
        LEDcount++;
        break;
      case 6:
        LEDcount++;
        break;
      case 7:
        LEDcount++;
        break;
      case 8:
        LEDcount++;
        break;
      case 9:
        LEDcount++;
        break;
      case 10:
        LEDcount++;
        break;
      case 11:
        LEDcount++;
        break;
      case 12:
        LEDcount++;
        break;

      case 13:
          digitalWrite(StatLED, HIGH);
          LEDcount = 0;
       break;
    }
    return LEDcount;
}
```

Programmcode

```
#include <Nextion.h>                          //nötige Bibliotheken inkludieren
#include "FrigidusPotus.h"

//Nextion Objekte global                      (Seite, ID, Name)
NexButton bSchliessen (6, 1, "bSchliessen"); //"Schließen"-Knopf deaktiviert alle Komponenten
NexCrop qLue (6, 2, "qLue");                 //"Bildausschnitte" (GUI-Komponenten)
NexCrop qPum (6, 3, "qPum");                 //  entsprechen Schaltern und sind im
NexCrop qVen (6, 4, "qVen");                 //  Displaycode als solche programmiert
NexCrop qPel1 (6, 5, "qPel1");
NexCrop qPel2 (6, 6, "qPel2");
NexCrop qPel3 (6, 7, "qPel3");
NexButton bWartung (3, 4, "bWartung");
NexButton bReinigung (11, 2, "bReinigung");
NexButton bSpuelung (12, 2, "bSpuelung");
NexButton bNachspuel (14, 2, "bNachspuel");
NexButton bKaltgetr (4, 1, "bKaltgetr"); //
NexButton bHeissgetr (5, 1, "Heissgetr");
NexButton bManuell (8, 1, "bManuell");
NexButton bEntleeren (18, 1, "bEntleeren");

//Setup Brührungs-Ereignisse                  (mit Komma getrennt)
NexTouch *nex_listen_list[] = {
  &bSchliessen,
  &qLue,
  &qPum,
  &qVen,
  &qPel1,
  &qPel2,
  &qPel3,
  &bWartung,
  &bReinigung,
  &bSpuelung,
  &bNachspuel,
  &bKaltgetr,
  &bHeissgetr,
  &bManuell,
  &bEntleeren,
  NULL
};

//Pinzuweisung
const uint8_t StoppKnopf = 2;                //Stoppknopf (normale digitale Auslesung)
const uint8_t Pumpe = 4;                     //Pumpe (Pin 5 für PWM mit Timer OC0B)
const uint8_t Luefter = 3;                   //Lüfter
const uint8_t Ventil = PB0;                  //Ventil (Pin 8)
const uint8_t Peltier1 = 5;                  //Relais 1
const uint8_t Peltier2 = 6;                  //Relais 2
const uint8_t Peltier3 = 7;                  //Relais 3
const uint8_t NTC_KKh = A0;                  //Temperatursensor Kühlkörper heiße Seite
const uint8_t NTC_KKk = A1;                  //Temperatursensor Kühlkörper kalte Seite
const uint8_t PhotoR = A2;                   //Photosensor
const uint8_t NTC_Getr = A3;                 //Temperatursensor Getränk
const uint8_t StatLED = 13;                  //Status LED an Pin 13

//Flags
bool Nachlueften = false;                    //Flag für Aktivierung der Lüfter außerhalb eines Programms
bool Int0 = false;                           //Flag für ISR
bool Stopp = false;                          //wurde gestoppt?
bool Wartung = false;                        //wird Wartungsinterface angezeigt?
float Timer1 = 0.0;                          //Zähler für Timer 1
float Timer1vgl = 0.0;                       //um prozentualen Fortschritt bei Timer 1 zu berechnen
int Timer1Event1 = 0;                        //Um weitere Ereignisse im Zeitverlauf zu ermöglichen
int Timer2 = 0;                              //Zähler für Timer 2
float ZielTemp = 0.0;                        //Zieltemperatur
float DeltaTemp = 0.0;                       //um prozentualen Fortschritt bei Kühlung zu berechnen
float DeltaTempVgl = 0.0;
uint8_t StoppZeit = 0;                       //misst die Zeit des Stopp-Tasten-Druck
uint8_t LEDcount = 0;                        //Zähler für Status LED

//Variablendeklaration Global
uint8_t Hell = 90;                           //zur Änderungshysterese der Helligkeit
bool LueAN = false;                          //Variablen zur Aktivierung/ Deaktivierung
bool PumAN = false;                          //einzelner Komponenten
bool VenAN = false;
```

```
81  bool Pel1AN = false;
82  bool Pel2AN = false;
83  bool Pel3AN = false;
84  int8_t T1 = 1;                              //Zeitkonstanten für Timer, entweder +1 oder -1
85
86
87
88  //      +++       Interrupt Service Routinen        +++
89
90  ISR(INT0_vect){
91    Int0 = true;
92  }
93
94  ISR(TIMER1_COMPA_vect){
95    Timer1 += T1;
96  }
97
98  ISR(TIMER2_COMPA_vect){
99    Timer2++;
100 }
101
102
103
104 //          +++         Funktionen          +++
105
106 void AllesAus(){
107   PORTD &= ~(1 << Luefter);                 //Setzen der entsprechenden Bits auf 0 im PortD-Register
108   PORTD &= ~(1 << Pumpe);
109   PORTB &= ~(1 << Ventil);
110   PORTD &= ~(1 << Peltier1);
111   PORTD &= ~(1 << Peltier2);
112   PORTD &= ~(1 << Peltier3);
113 }
114
115 void HardwareUpdate(){
116   if(LueAN)                                 //Lüfter
117     PORTD |= (1 << Luefter);
118   else PORTD &= ~(1 << Luefter);
119   if(PumAN)                                 //Pumpe
120     PORTD |= (1 << Pumpe);
121   else PORTD &= ~(1 << Pumpe);
122   if(VenAN)                                 //Ventil
123     PORTB |= (1 << Ventil);
124   else PORTB &= ~(1 << Ventil);
125   if(Pel1AN)                                //Peltier 1 - 3
126     PORTD |= (1 << Peltier1);
127   else PORTD &= ~(1 << Peltier1);
128   if(Pel2AN)
129     PORTD |= (1 << Peltier2);
130   else PORTD &= ~(1 << Peltier2);
131   if(Pel3AN)
132     PORTD |= (1 << Peltier3);
133   else PORTD &= ~(1 << Peltier3);
134 }
135
136 //Abfrage des Stoppknopfes
137 void StoppCheck(){
138   if(Int0){
139     if(StoppZeit < 1)
140     {
141       if(digitalRead(StoppKnopf) == HIGH){
142         StoppZeit++;                                    //Erhöhung der Zeit des Tastendrucks
143       }else{
144         Int0 = false;                                   //Reset der Stoppknopf-Flags
145         StoppZeit = 0;
146       }
147     }else{
148       //Abschaltung von allem und Statusänderung
149       AllesAus();
150       Serial.print("page 18");
151       EndTrans();                                       //siehe FrigidusPotus.cpp
152       Int0 = false;                                     //Reset der Stoppknopf-Flags
153       StoppZeit = 0;
154       Stopp = true;
```

```
    }
  }
}



//        +++       Nextion-Events        +++

void Schliessen(void *ptr){
  //Deaktivierung aller Komponenten
  AllesAus();
  LueAN = false;
  PumAN = false;
  VenAN = false;
  Pel1AN = false;
  Pel2AN = false;
  Pel3AN = false;
  Wartung = false;
}

void Lue(void *ptr){                          //Änderungen der Stati entsprechend des Berührungs-Events,
  LueAN = !LueAN;                             //  *sprich der Stati der Checkboxen
  HardwareUpdate();
}

void Pum(void *ptr){
  PumAN = !PumAN;
  HardwareUpdate();
}

void Ven(void *ptr){
  VenAN = !VenAN;
  HardwareUpdate();
}

void Pel1(void *ptr){
  Pel1AN = !Pel1AN;
  HardwareUpdate();
}

void Pel2(void *ptr){
  Pel2AN = !Pel2AN;
  HardwareUpdate();
}

void Pel3(void *ptr){
  Pel3AN = !Pel3AN;
  HardwareUpdate();
}

void Wartung_start(void *ptr){
  Wartung = true;
}

void Spuelung(void *ptr){
  Timer1 = 30.0;                          //Zeit = 30s
  Timer1Event1 = 10.0;                    //Ereignis bei 10 verbleibenden Sekunden
  Timer1Toggle();                         //Aktivierung des Timers 1
  Spuelprogramm();                        //Spulprogramm aufrufen
}

void Reinigung(void *ptr){
  Timer1 = 60.0;                            //siehe Spuelung()
  Timer1Event1 = 20.0;
  Timer1Toggle();
  Spuelprogramm();
}

void Nachspuel(void *ptr){
  Timer1 = 40.0;                            //siehe Spuelung()
  Timer1Event1 = 15.0;
  Timer1Toggle();
  Spuelprogramm();
}
```

```
229
230 void Kaltgetr(void *ptr){
231   ZielTemp = 7.0;                          //Zieltemperatur auf 7°C festgelegt
232   Kuehlprogramm();                         //Aufruf des Kühlprogramms
233 }
234
235 void Heissgetr(void *ptr){
236   ZielTemp = 60.0;                         //Zieltemperatur auf 60 °C festgelegt
237   Kuehlprogramm();                         //Aufruf des Kühlprogramms
238 }
239
240 union{                                     //Umwandlung Daten in Integer
241   char DataChar[4];
242   long valueLong;
243 }value;
244 void Manuell(void *ptr){
245   //Zieltemperatur auslesen
246   String DataString = "";                  //Rohdaten
247   while(!Serial.available()){              //Warten bis Zieltemperatur übermittelt wurde
248   }
249   DataString += char(Serial.read());       //Lesen der Zieltemperatur in Rohdaten-String
250   for(int i = 0; i <= 4; i++){
251     value.DataChar[i] = char(DataString[i]);  //Einzelne Zahlen aus Zeichen-Array extrahieren
252   }
253   ZielTemp = value.valueLong;              //Speicherung der sich ergebenden Zieltemperatur
254   Kuehlprogramm();                         //Aufruf des Kühlprogramms
255 }
256
257 void Entleeren(void *ptr){                 //Entleerung der Leitungen nach Abbruch durch Stoppknopf
258   Leeren();
259 }
260
261
262
263 //        +++       Unterprogramme        +++
264
265 void Kuehlprogramm(){
266   int Temp_KKk = 20;                         //Initialisierung mit 20°C für Schleife
267   Serial.print("nZielTemp.val=");             //Sendung der Zieltemperatur zum Display
268   Serial.print((int)ZielTemp);               //  um Richtigkeit manuell überprüfen zu können
269   EndTrans();
270   int Temp_Getr = TempMessungGetr(NTC_Getr);
271   DeltaTempVgl = Temp_Getr - ZielTemp;       //zur Berechnung des Fortschritts
272   if(Temp_Getr <= ZielTemp){                 //Abfrage, ob überhaupt gekühlt werden muss
273     Serial.print("page 17");                 //ggf. auf Display benachrichtigen
274     EndTrans();                              //  und abbrechen
275   }else{
276     int Timer1Minuten = 0;
277     int Timer1Sekunden = 0;
278     int Temp_KKh = 0;
279     int Fortschritt = 0;
280     if(ZielTemp <= 20.0){                    //für präzieseres Erreichen der Zieltemperatur
281       ZielTemp--;                            // (bei Regelung wäre das nicht nötig)
282     }
283     T1 = 1;                                  //Timer 1 auf aufwärts zählen
284     Timer1 = 0;                              //Timer 1 auf 0 setzen
285     Timer1Toggle();                          //Timer 1 aktivieren
286     PORTD |= (1 << Luefter);                 //Aktivierung der vollen Kühlleistung und der Pumpe
287     PORTD |= (1 << Pumpe);
288     PORTD |= (1 << Peltier1);
289     PORTD |= (1 << Peltier2);
290     PORTD |= (1 << Peltier3);
291     int Temp_Getr_Puff = 0;
292     uint8_t Timer2Event1 = 1;
293     //Schleife, bis Zieltemperatur erreicht ist, Stopp gedrückt wurde oder der Kühlkörper zu kalt ist
294     while(Temp_Getr > ZielTemp && !Stopp && Temp_KKk > -1 || Timer1 < 2){ //Abbruch nicht vor T=2 Sekunden
295       Serial.print("");                      //Workaround, ohne scheint sich der uC jedes mal aufzuhängen
296       if(Timer2 >= (Timer2Event1 * 10)){
297         Timer2Event1++;                      //Werte für Temp_Getr Mittlung
298         Temp_Getr_Puff += TempMessungGetr(NTC_Getr);
299       }
300       if(Timer2 >= 100){                                    //Update alle
301         Timer2Event1 = 1;
302       //Messung aller Temperaturen (ggf. für Regelungszwecke später)
```

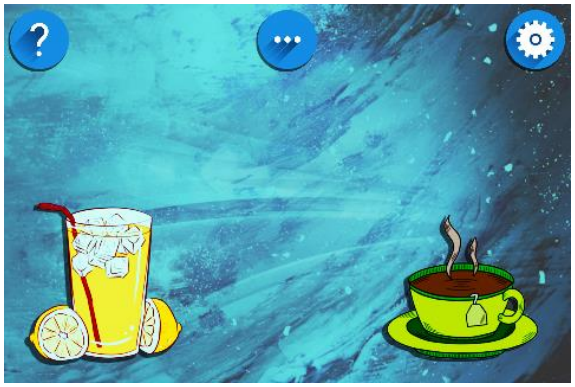
```
303           Temp_KKh = TempMessungKK(NTC_KKh);
304           Temp_KKk = TempMessungKK(NTC_KKk);
305           Temp_Getr = Temp_Getr_Puff / 10;      //Temp_Getr Mittlung
306           Temp_Getr_Puff = 0;                   //Reset für Temp_Getr Mittlung
307           //Umrechnung der Sekunden aus Timer1 in Minuten und Sekunden
308           Timer1Minuten = Timer1 / 60;
309           Timer1Sekunden = Timer1 - (Timer1Minuten * 60);
310           //Berechnung des Fortschritts
311           DeltaTemp = Temp_Getr - ZielTemp;
312           Fortschritt = 100.0 - (100.0 * (DeltaTemp / DeltaTempVgl));
313           if(Fortschritt == 100){
314             Fortschritt -= 1;
315           }
316           if(Fortschritt < 0){
317             Fortschritt = 0;
318           }
319           //Anzeige aller relevanter Werte
320           KuehlAnzeige(Temp_Getr, Timer1Minuten, Timer1Sekunden, Fortschritt);
321           StoppCheck();
322           Hell = Helligkeit(PhotoR, Hell);  //Helligkeit des Displays regulieren
323           HellAnzeige(Hell);
324
325           //Modifizierung d. Hardware (genauere Steuerung statt Regelung)
326
327           Timer2 = 0;
328         }
329       }
330       //Anzeige aller relevanter Werte
331       if(ZielTemp <= 19){
332         Temp_Getr = ZielTemp + 1;
333       }
334       KuehlAnzeige(Temp_Getr, Timer1Minuten, Timer1Sekunden, Fortschritt);
335       Timer1Toggle();                         //Timer 1 deaktiviert
336       if(!Stopp){
337         Serial.print("jFortschritt.val=100");   //Triggert nächstes Ereignis auf Display
338         EndTrans();
339         PORTD &= ~(1 << Peltier1);              //Kühlleistung = 0, Lüfter und Pumpe
340         PORTD &= ~(1 << Peltier2);              //  bleiben an
341         PORTD &= ~(1 << Peltier3);
342         int Warten = 0;                         //Wartet, dass das Display mit Sicherheit
343         while(Warten < 5){                      // den Fortschritt = 100 verarbeitet hat
344           Serial.print("");                     //Workaround, ohne scheint sich der uC jedes mal aufzuhängen
345           if(Timer2 >= 50){
346             Warten++;
347             Timer2 = 0;
348           }
349         }
350         if(Temp_KKk <= -1){
351           String cmd = "\"";
352           Serial.print("tLeitungen.txt=" + cmd + "" + cmd);       //Fehlermeldung die Temperatur
353           EndTrans();                                             //  der Kühlkörper betreffend
354           Serial.print("twerden.txt=" + cmd + "Fehler!" + cmd);
355           EndTrans();
356           Serial.print("tgeleert.txt=" + cmd + "" + cmd);
357           EndTrans();
358         }
359         Leeren();
360       }
361     }
362     StoppZeit = 0;
363   }
364
365   //Entleerung des Leitungssystems
366   void Leeren(){
367     Timer1 = 15.0;                          //Setup ähnlich zu Spuelung() mit T = 10s
368     T1 = -1;
369     Timer1Toggle();
370     Timer1vg1 = Timer1;
371     PORTD |= (1 << Pumpe);                  //Pumpe und Ventil aktivieren
372     PORTB |= (1 << Ventil);
373     while(Timer1 > 0){
374       Serial.print("");                     //Workaround (vorher beschrieben)
375       if(Timer2 >= 50){
376         //Fortschrittsberechnung
```

```
      float Fortschritt = 100.0-(100.0*(Timer1/Timer1vgl));
      Serial.print("jFortschritt.val=");  //Fortschrittsanzeige in % auf Display
      Serial.print((int)Fortschritt);
      EndTrans();
      Hell = Helligkeit(PhotoR, Hell);
      HellAnzeige(Hell);
      Timer2 = 0;
    }
  }
  Serial.print("jFortschritt.val=100");   //Sicherstellen, dass der Wert 100 am Display ankommt
  EndTrans();
  AllesAus();
  Timer1Toggle();
}

void Spuelprogramm(){
  Timer1vgl = Timer1;                    //beide gleich setzen -> Fortschritt = 0%
  T1 = -1;                               //Timer 1 auf runterzählen
  PORTD |= (1 << Pumpe);                 //Aktivierung der Pumpe
  while(Timer1 > 0.0 && !Stopp){
    if(Timer2 >= 50){                    //Aktualisierung aller 50ms
      //Fortschrittsberechnung
      float Fortschritt = 100.0-(100.0*(Timer1/Timer1vgl));
      Serial.print("jFortschritt.val=");  //Fortschrittsanzeige in % auf Display
      Serial.print((int)Fortschritt);
      EndTrans();
      Hell = Helligkeit(PhotoR, Hell);
      HellAnzeige(Hell);
      StoppCheck();
      Timer2 = 0;
    }
    if(Timer1 <= Timer1Event1){
      PORTB |= (1 << Ventil);            //Aktivierung des Ventils zum leeren
    }
  }
  Timer1Toggle();
  Serial.print("jFortschritt.val=100");   //Sicherstellen, dass der Wert 100 am Display ankommt
  EndTrans();
  AllesAus();
  StoppZeit = 0;
}



//          +++       Setup-Code        +++

void setup() {
  //Setup Kommunikation
  Serial.begin(9600);                          //Serielle Kommunikation mit Display initialisieren
  Serial.setTimeout(20);                       //Wartezeit der Seriellen Kommunikation setzen

  //Setup Timer Interrupt
  TCCR1A = 0x00;                               //Deaktivierung analogWrite bei Pin 9 & 10
  TCCR1B = 0x0C;                               //Prescaler auf 256 setzen & CTC aktivieren (Auto-Reset)
  OCR1A = 62500;                               //Vergleichwert für T = 1s
  TIMSK1 = 0x00;                               //Timer1 deaktiviert
  TCNT1 = 0;                                   //Timer auf 0 setzen

  TCCR2A = 0x02;                               //Deaktivierung analogWrite bei Pin 3 & 11 & CTC aktivieren
  TCCR2B = 0x05;                               //Pre-Skaler auf 128 setzen
  OCR2A = 125;                                 //Wert für T = 1ms
  TIMSK2 = 0x02;                               //Aktivierung CompA Interrupt
  TCNT2 = 0;                                   //Timer auf 0 setzen

  //Setup externes Interrupt
  EICRA = 0x03;                                //beide ISC0 auf 1 -> Interrupt bei steigender Flanke auf INT0
  EIMSK |= (1 << INT0);                        //Aktivierung des Interrupts für INT0

  sei();

  //Setup Nextion
  nexInit();
  bSchliessen.attachPop(Schliessen, &bSchliessen);
  qLue.attachPush(Lue, &qLue);                 //Knopfname.attachPush(Funktion, &Knopfname);
```

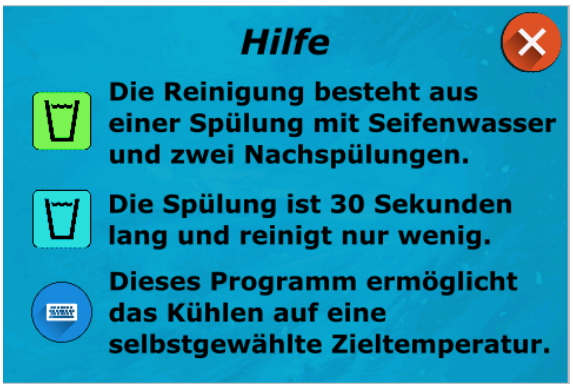
```
  qPum.attachPush(Pum, &qPum);
  qVen.attachPush(Ven, &qVen);
  qPel1.attachPush(Pel1, &qPel1);
  qPel2.attachPush(Pel2, &qPel2);
  qPel3.attachPush(Pel3, &qPel3);
  bWartung.attachPop(Wartung_start, &bWartung);
  bReinigung.attachPop(Reinigung, &bReinigung);
  bSpuelung.attachPop(Spuelung, &bSpuelung);
  bNachspuel.attachPop(Nachspuel, &bNachspuel);
  bKaltgetr.attachPop(Kaltgetr, &bKaltgetr);
  bHeissgetr.attachPop(Heissgetr, &bHeissgetr);
  bManuell.attachPop(Manuell, &bManuell);
  bEntleeren.attachPop(Entleeren, &bEntleeren);

  //Setup der Pinmodi
  DDRD = 0xF8;                                  //Pins 0 - 2 auf Eingang, Pins 3 - 7 auf Ausgang (B 1111 1000)
  DDRB |= (1 << Ventil);                        //Pin 8 auf Ausgang
  DDRB |= (1 << StatLED);                       //Status-LED auf Ausgang setzen & aktivieren
  pinMode(StatLED,OUTPUT);
}



//          +++       Hauptcode         +++

void loop() {
  Stopp = false;
  nexLoop(nex_listen_list);

  if(Timer2 >= 100){                             //je 50ms
    Hell = Helligkeit(PhotoR, Hell);             //Helligkeit des Displays anpassen
    HellAnzeige(Hell);
    LEDcount = StatusLED(LEDcount, StatLED);     //Status-LED blinken lassen

    int Temp_KKh = TempMessungKK(NTC_KKh);

    if(Nachlueften && Wartung){
      PORTD &= ~(1 << Luefter);
      Nachlueften = false;
    }

    if(Temp_KKh >= 33 && !Wartung){              //Nachlüften wenn Restwärme zu hoch
      Nachlueften = true;                        //  (hilft auch dem Luftstrom des Netzteils)
    }
    if(Nachlueften){
      if(Temp_KKh >= 28){
        PORTD |= (1 << Luefter);
      }else{
        PORTD &= ~(1 << Luefter);
        Nachlueften = false;
      }
    }

    if(Wartung){                                 //Wartungsprogramm
      int Temp_KKk = TempMessungKK(NTC_KKk);
      int Temp_Getr = TempMessungGetr(NTC_Getr);
      TempAnzeige(Temp_KKh, Temp_KKk, Temp_Getr);
    }

    Timer2 = 0;
  }
}

```

Anlagen, Teil 7

Display-Seiten

	
01: Startseite	02: Hauptmenü

	
03: Hilfe – Hauptmenü	04: Weiteres



05: Hilfe – Weiteres

06: Reinigung

07: Spülung

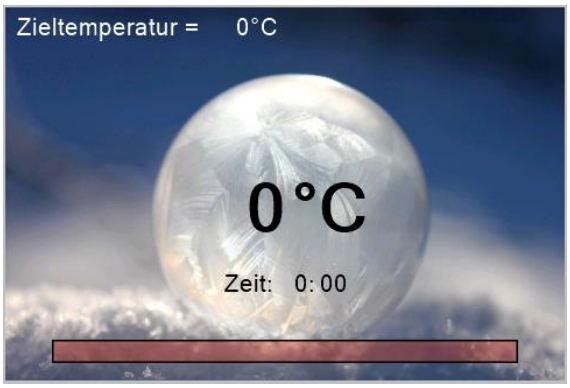
08: Spülprogramm

09: Tastenfeld

10: Eingabefehler

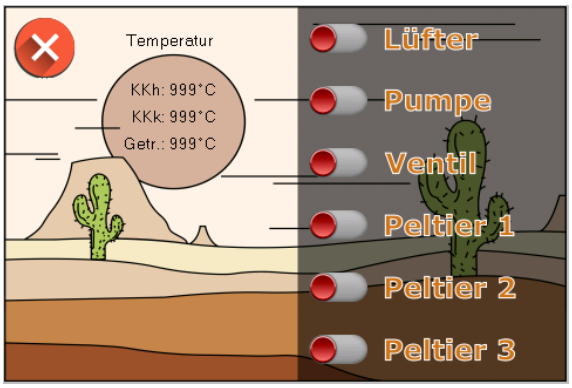
11: Bereit?

12: Abbruch

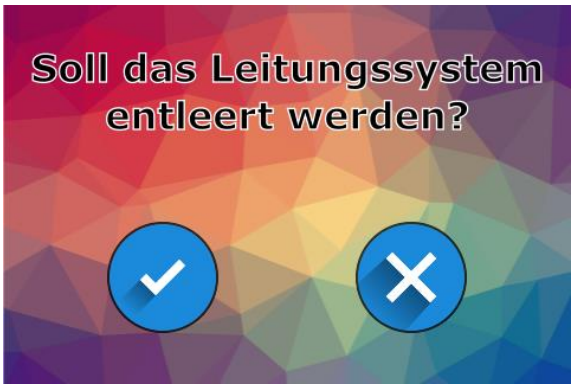


13: Kühlung

14: Fertig



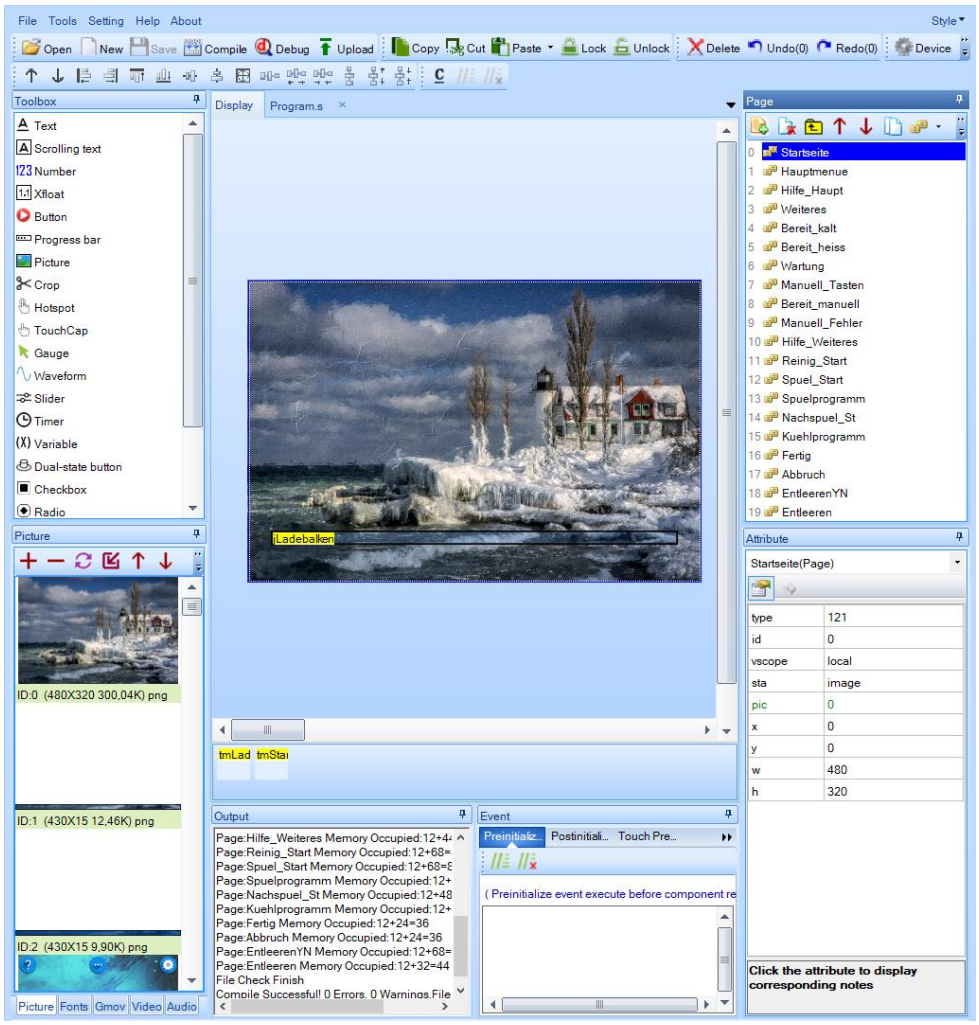
15: Wartung



16: Leeren?

17: Leeren

Editor



Anlagen, Teil 8

Versuchsaufbau

Messungen

Leistungsversuch

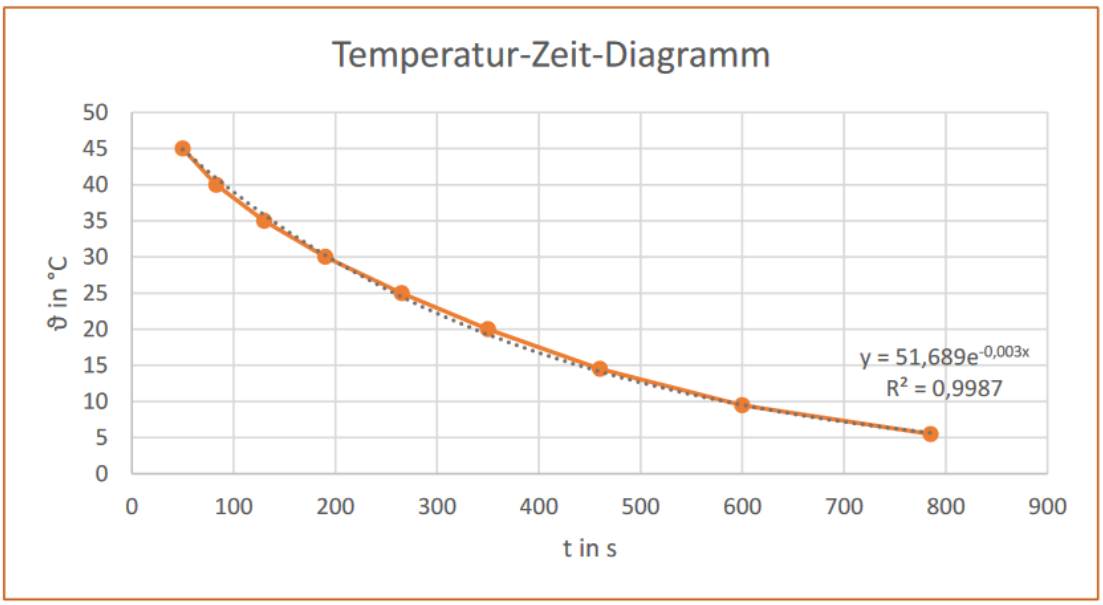
Alle Temperaturwerte sind in °C, alle Massen in kg,
alle Zeiten in s und alle Leistungen in W angegeben.

Messung 1

m = 0,514 ϑ_{Start} = 56

ϑ_{KKk} = 25 ϑ_{Ziel} = 5

$\vartheta_{Multimeter}$	$\vartheta_{Prototyp}$	ϑ_{Mittel}	t	P
45	45	45	50	473,8
40	40	40	83	326,3
35	35	35	130	229,1
30	30	30	190	179,5
25	25	25	265	143,6
20	20	20	350	126,7
15	14	15	460	107,7
10	9	10	600	76,9
6	5	6	785	46,6

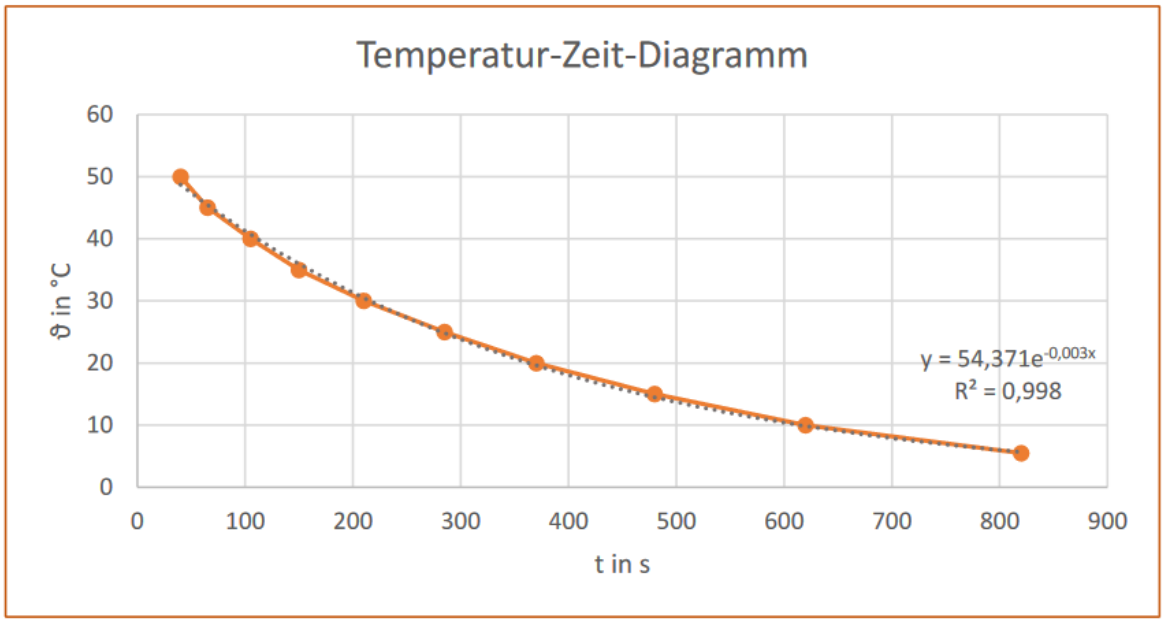


Messung 2

m = 0,510 ϑ_{Start} = 58

ϑ_{KKk} = 28 ϑ_{Ziel} = 5

$\vartheta_{Multimeter}$	$\vartheta_{Prototyp}$	ϑ_{Mittel}	t	P
50	50	50	40	427,4
45	45	45	65	427,4
40	40	40	105	267,1
35	35	35	150	237,4
30	30	30	210	178,1
25	25	25	285	142,5
20	20	20	370	125,7
15	15	15	480	97,1
10	10	10	620	76,3
6	5	6	820	48,1

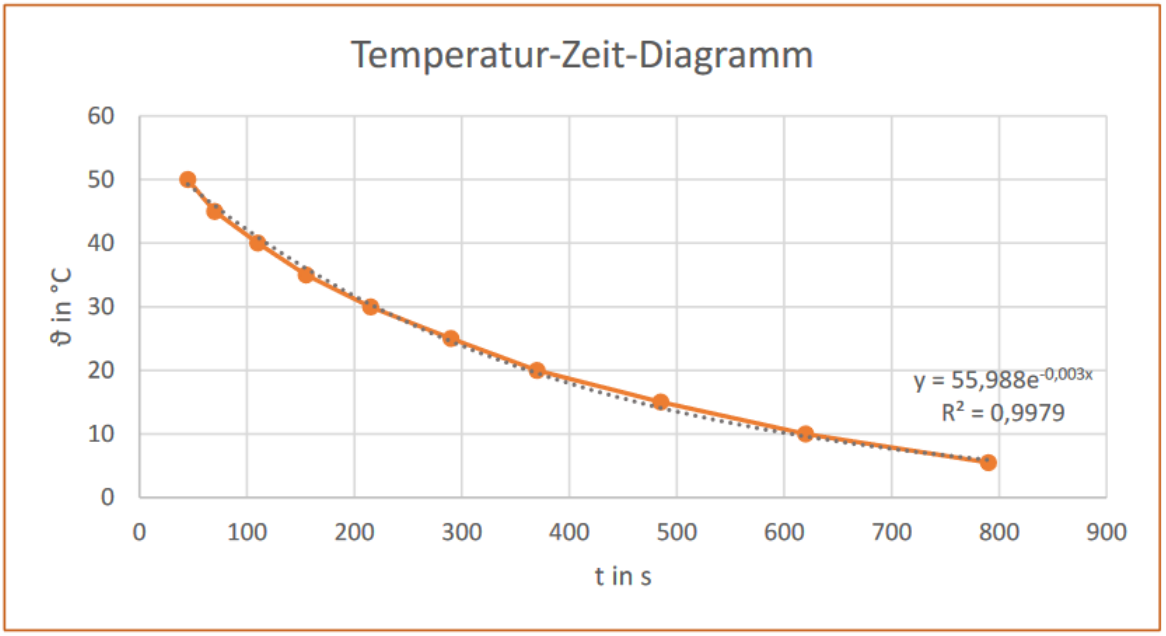


Messung 3

m = 0,500 ϑ_{Start} = 60

ϑ_{KKk} = 27 ϑ_{Ziel} = 5

$\vartheta_{Multimeter}$	$\vartheta_{Prototyp}$	ϑ_{Mittel}	t	P
50	50	50	45	465,6
45	45	45	70	419,0
40	40	40	110	261,9
35	35	35	155	232,8
30	30	30	215	174,6
25	25	25	290	139,7
20	20	20	370	130,9
15	15	15	485	91,1
10	10	10	620	77,6
6	5	6	790	55,5



Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Leipzig, den 30.05.2023

Béla Schroth