

# Resistive und induktive Heizung in einer Kristallzüchtungsanlage: Automatisierung mit Python und Vermessung der elektromagnetischen Parameter

---

Bachelorarbeit

Name des Studiengangs  
Elektrotechnik

Fachbereich 1 - Ingenieurwissenschaften – Energie  
und Information

vorgelegt von  
Vincent Funke

Datum:  
Berlin, 31.03.2022

Erstgutachterin: Prof. Dr. Anett Bailleu  
Zweitgutachter: Dr. Kaspars Dadzis

# Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>IV</b>
<b>Tabellenverzeichnis</b>	<b>VIII</b>
<b>Abkürzungs- und Formelzeichenverzeichnis</b>	<b>IX</b>
<b>1 Einleitung</b>	<b>1</b>
<b>2 Grundlagen</b>	<b>3</b>
2.1 Widerstandsheizung (Resistiv) . . . . .	3
2.2 Induktionsheizung (Induktiv) . . . . .	4
2.3 PID-Regelung . . . . .	5
2.4 Sensoren . . . . .	6
2.4.1 Temperatursensoren . . . . .	6
2.4.2 Magnetfeldsensoren . . . . .	8
2.4.3 Elektrische Messungen (P, I, U, B, Phi) . . . . .	8
2.5 Emissivitäten . . . . .	13
<b>3 Temperatur-Regelung mit PID</b>	<b>14</b>
3.1 Eurotherm Regler (Variante 1) . . . . .	14
3.2 Digitale Schnittstelle am Eurotherm Regler . . . . .	14
3.3 Eurotherm Befehle . . . . .	16
3.4 Eurotherm Emulation mit Arduino (Variante 2) . . . . .	19
3.4.1 Software der Emulation . . . . .	19
3.4.2 Hardware der Emulation . . . . .	23
3.5 Arduino Leistungssteuerung und Eurotherm PID (Variante 3) . . . . .	27
3.6 PID-Regler - Arduino vs. Eurotherm . . . . .	28
3.7 Testaufbau mit Kochplatte . . . . .	32
<b>4 Messung des Emissionsgrades</b>	<b>34</b>
4.1 Messaufbau . . . . .	34
4.2 Erklärung des Programms . . . . .	45
4.2.1 Hauptprogramm . . . . .	45
4.2.2 Python Bibliotheken für Einzelgeräte . . . . .	48
4.2.3 Rezept und Parameterliste . . . . .	48
4.3 Messreihen und Ergebnis . . . . .	51
<b>5 Vermessung der elektromagnetischen Parameter</b>	<b>55</b>
5.1 Kalibrierung mit Helmholtzspule und Hall-Sensor . . . . .	55
5.1.1 Messaufbau . . . . .	55
5.1.2 Erklärung des Messprogramms . . . . .	57
5.1.3 Ergebnis und Auswertung . . . . .	60
5.2 Magnetische Profilbestimmung der Spule in der Test-CZ-Anlage . . . . .	65
5.2.1 Messaufbau . . . . .	65
5.2.2 Erklärung des Messprogramms . . . . .	68
5.2.3 Ergebnis und Auswertung . . . . .	70
5.3 Leistungsmessungen an beiden Heizern . . . . .	78
5.3.1 Messaufbau . . . . .	78
5.3.2 Erklärung des Messprogramms . . . . .	80
5.3.3 Ergebnis und Auswertung . . . . .	81

<b>6 Zusammenfassung</b>	<b>96</b>
<b>Literaturverzeichnis</b>	<b>97</b>
<b>7 Anhang</b>	<b>XI</b>
7.1 Geräteliste . . . . .	XI
7.2 Programme . . . . .	XII
7.3 Weitere Texte . . . . .	XV
7.4 Bilder . . . . .	XVI

## Abbildungsverzeichnis

1	Test-CZ Anlage mit Spule (Quelle: eigene Darstellung) . . . . .	1
2	Grafit-Heizer (mäanderförmig) (Quelle: eigene Darstellung) . . . . .	3
3	Anlage mit Grafit-Tiegel (Suszeptor) (Quelle: eigene Darstellung) . . . . .	5
4	Oszilloskop-Kurven (Quelle: eigene Darstellung) . . . . .	9
5	Helmholtzspule ( $R = \text{Radius}$ ) (Quelle: [Wik]) . . . . .	12
6	Klemmenanschluss - Eurotherm 905S (900 EPC) (Quelle: [Eura] S. 2-13) . . . . .	15
7	Programmcode - Schnittstelle (Quelle: eigene Darstellung) . . . . .	15
8	Ablaufschema des Arduino Programms (Quelle: eigene Darstellung) . . . . .	22
9	Ausgabe zweite Schnittstelle mit Putty (Quelle: eigene Darstellung) . . . . .	23
10	Hardwaresicherheit - Spannungsteiler und Diode (Quelle: eigene Darstellung) . . . . .	24
11	Extra Elemente für Kontrolle und Sicherheit . . . . .	26
	a    Arduino Adafruit-Modul Pt100 und LED mit $330\ \Omega$ Widerstand (Quelle: eigene Darstellung) . . . . .	26
	b    Schnittstelle 2 und Kondensatoren (Quelle: eigene Darstellung) . . . . .	26
	c    DAC-Chip und Spannungsteiler (Quelle: eigene Darstellung) . . . . .	26
	d    Kondensatoren am Pt100-Anschluss zum Adafruit-Modul (Quelle: eigene Darstellung) . . . . .	26
12	Gesamt Aufbau an der Test-CZ Anlage . . . . .	27
	a    Alles auf einem Steckbrett (Quelle: eigene Darstellung) . . . . .	27
	b    Aufbau Anlage - Eurotherm, Arduino, Versorgung, Multimeter (Quelle: eigene Darstellung) . . . . .	27
13	AutoTune Eurotherm an Testaufbau (Quelle: eigene Darstellung) . . . . .	29
14	AutoTune Eurotherm an Test-CZ (Quelle: eigene Darstellung) . . . . .	29
15	Zeichnerische Lösung Ziegler-Nichols-Methode (Quelle: eigene Darstellung) . . . . .	30
16	Eurotherm Regelung (Quelle: eigene Darstellung) . . . . .	31
17	Arduino Regelung (Quelle: eigene Darstellung) . . . . .	32
18	Programm Test-Aufbau mit Kochplatte (Quelle: eigene Darstellung) . . . . .	33
19	Solid-State-Relais für Kochplatte . . . . .	33
	a    Solid-State-Relais innen (Quelle: eigene Darstellung) . . . . .	33
	b    Anschluss des Solid-State-Relais (Quelle: eigene Darstellung) . . . . .	33
20	Skizze des Grafit-Heizers (Quelle: eigene Darstellung) . . . . .	34
21	Aluminium-Körper mit Grafit-Probe (Quelle: [Fun21a] S. 3) . . . . .	35
22	Grafit-Heizer und stehender Aluminium-Körper (Quelle: eigene Darstellung) . . . . .	36
23	Praktikums- und Neuer Aufbau . . . . .	37
	a    ehemaliger Aufbau (Quelle: [Fun21a] S. 3) . . . . .	37
	b    verbesserter Aufbau (Quelle: eigene Darstellung) . . . . .	37
24	Genutzte Messgeräte (Quelle: eigene Darstellung) . . . . .	38
	a    Eurotherm 905S . . . . .	38
	b    Adafruit-Modul MAX31865 . . . . .	38
	c    Pyrometer KW . . . . .	38
	d    zwei Pyrometer LW . . . . .	38
25	Anschluss Geräte (Quelle: eigene Darstellung) . . . . .	39
	a    Raspberry Pi 400 . . . . .	39
	b    USB-Hub von Transcend . . . . .	39
26	Genutzte Sensoren im/am Aluminium-Körper (Quelle: eigene Darstellung) . . . . .	40
27	Genutzter Pt100 (Quelle: eigene Darstellung) . . . . .	40
28	Oberflächensor ehemaliger Aufbau (Quelle: eigene Darstellung) . . . . .	41
	a    Pt100 angepresst . . . . .	41
	b    Grafit-Filz für extra Halt . . . . .	41
29	Oberflächensor verbesserter Aufbau (Quelle: eigene Darstellung) . . . . .	41

a	Pt100 angepresst . . . . .	41
b	Aussehen des Pt100-Sensors . . . . .	41
30	Ausrichtung des Kurzweligen Pyrometers (IGA 6/23) (Quelle: eigene Darstellung)	42
31	Ausrichtung des Langwelligen Pyrometers (Impac Series 600 - AV 20:1) (Quelle: eigene Darstellung) . . . . .	42
32	Reale Ausrichtung der Pyrometer (Quelle: eigene Darstellung) . . . . .	43
a	Draufsicht . . . . .	43
b	Laser der Pyrometer . . . . .	43
c	Seitenansicht . . . . .	43
33	Versorgung Grafit-Heizer durch Leistungseinheit (Quelle: eigene Darstellung) . . . . .	44
a	Leistungseinheit (Quelle: eigene Darstellung) . . . . .	44
b	Leistungseinheit innen (Quelle: eigene Darstellung) . . . . .	44
34	Kompletter Aufbau des Experimentes - von links nach rechts: Raspberry Pi, Test-CZ-Anlage mit Pyrometern und Wärmebildkamera, Computer zur Aufnahme des Stromes und der Wärmebildkamerabilder, Schaltschrank der Anlage (Quelle: eigene Darstellung) . . . . .	45
35	GUI des neuen Programmes (Quelle: eigene Darstellung) . . . . .	46
36	Messreihe mit Eurotherm PI-Regelung (Quelle: eigene Darstellung) . . . . .	51
37	Messreihe mit Eurotherm PI-Regelung (Quelle: eigene Darstellung) . . . . .	52
38	Zusammenfassung der Emissiongrade (Quelle: eigene Darstellung) . . . . .	53
39	Ausschnitt der Abbildungen 36 und 37 (Quelle: eigene Darstellung) . . . . .	53
40	Schaltplan Kalibrierung (Quelle: eigene Darstellung) . . . . .	55
41	Genutzte Helmholtzspule (Quelle: eigene Darstellung) . . . . .	56
42	Genutzter Hall-Sensor (Quelle: eigene Darstellung) . . . . .	56
43	Aufbau des Experimentes zur Kalibrierung (Quelle: eigene Darstellung) . . . . .	57
44	Ergebnis der Kalibrierung (Quelle: eigene Darstellung) . . . . .	61
45	Kontrolle des Magnetfeldes der Helmholtzspule (Quelle: eigene Darstellung) . . . . .	63
46	Korrekturfunktion aus Excel (Quelle: eigene Darstellung) . . . . .	64
47	Korrekturfunktion mit Python (Quelle: eigene Darstellung) . . . . .	64
48	Test-CZ-Anlage mit Spule und Hall-Sensor (Quelle: eigene Darstellung) . . . . .	65
49	Versorgung der Spule (Quelle: eigene Darstellung) . . . . .	66
a	Generator TruHeat MF 5030 von TRUMPF . . . . .	66
b	Schwingkreis und Wasseranschluss . . . . .	66
50	Kompletter Aufbau für die Profilbestimmung (Quelle: eigene Darstellung) . . . . .	66
51	Antrieb der Spindel (Welle) (Quelle: eigene Darstellung) . . . . .	67
52	Ausgebauter Hall-Sensor (Quelle: eigene Darstellung) . . . . .	67
53	Ausgebauter Hall-Sensor aus verschiedenen Perspektiven (Quelle: eigene Darstellung) . . . . .	68
a	Sensor von der Seite . . . . .	68
b	Sensor von oben . . . . .	68
54	Bildschirm des Schaltschrankes der Test-CZ Anlage (Quelle: eigene Darstellung) . . . . .	69
55	Position des Hall-Sensors an der Spindel (Quelle: eigene Darstellung) . . . . .	70
a	in der Mitte der Anlage . . . . .	70
b	5 cm von Mitte der Anlage entfernt . . . . .	70
56	Messreihen-Skizze der Hall-Sensor-Position (geradliniges Profil) (Quelle: eigene Darstellung) . . . . .	71
57	Kurvenschärfe des geradlinigen Profils (Quelle: eigene Darstellung) . . . . .	71
58	Messreihen-Skizze der Hall-Sensor-Position (rotierendes Profil) (Quelle: eigene Darstellung) . . . . .	72
59	Kurvenschärfe des rotierenden Profils (Quelle: eigene Darstellung) . . . . .	73

60	Kurvenschär der geradlinigen Profile umgerechnet zu $B_z(l)$ (Quelle: eigene Darstellung) . . . . .	75
61	Kurvenschär der rotierenden Profile umgerechnet zu $B_z(\alpha)$ (Quelle: eigene Darstellung) . . . . .	76
62	Koordinatensystem für die gefahrenen Wege (bzw. Höhe) (Quelle: eigene Darstellung) . . . . .	77
63	Messgeräte für Strom und Spannung am Oszilloskop (Quelle: eigene Darstellung) . . . . .	78
a	Spannungsmessung mit Differentialtastkopf - DP10013 von Micsig . . . . .	78
b	Strommessung mit einer Rogowskispule von Typ MA 200 . . . . .	78
64	Leitungsanschluss der Rogowskispule (Quelle: eigene Darstellung) . . . . .	79
65	Anschluss der Messgeräte am Grafit-Heizer (markiert durch weißen Kreis) (Quelle: eigene Darstellung) . . . . .	80
a	Anschluss Differentialtastkopf Minus . . . . .	80
b	Anschluss Differentialtastkopf Plus und Rogowskispule . . . . .	80
66	Plot des Programmes <i>hauptprogramm_Leistung.py</i> (Quelle: eigene Darstellung) . . . . .	81
67	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 2 (Quelle: eigene Darstellung) . . . . .	82
68	Darstellung von Phi in einem rechtwinkligen Dreieck (Quelle: eigene Darstellung) . . . . .	86
69	Test Grafit-Block in Test-CZ Spule (Quelle: eigene Darstellung) . . . . .	87
70	Lissajous-Figur zum Versuch Nr. 1 (Quelle: eigene Darstellung) . . . . .	88
71	Lissajous-Figur zum Versuch Nr. 10 (Quelle: eigene Darstellung) . . . . .	88
72	Werte-Anzeige des Oszilloskops (Quelle: eigene Darstellung) . . . . .	90
a	Phasenverschiebung und Phasenverschiebungszeit . . . . .	90
b	Spitzen-Spitzen-Wert und Periodendauer . . . . .	90
73	Generator-Bildschirm (Quelle: eigene Darstellung) . . . . .	90
74	Oszilloskop-Plot der ersten Messung mit OP = 10 % (Quelle: eigene Darstellung) . . . . .	92
75	Oszilloskop-Plot der zweiten Messung mit OP = 12 % (Quelle: eigene Darstellung) . . . . .	92
76	Korrigierte Kurve mit OP = 10 % mit p(t) (Quelle: eigene Darstellung) . . . . .	93
77	Korrigierte Kurve mit OP = 12 % mit p(t) (Quelle: eigene Darstellung) . . . . .	93
78	Ausschnitt von p(t) aus Abbildung 76 (Quelle: eigene Darstellung) . . . . .	94
79	Ausschnitt von p(t) aus Abbildung 77 (Quelle: eigene Darstellung) . . . . .	94
80	Lissajous-Figur mit OP = 10 % (Quelle: eigene Darstellung) . . . . .	95
a	nicht angepasste Kurve . . . . .	95
b	angepasste Kurve . . . . .	95
81	Lissajous-Figur mit OP = 12 % (Quelle: eigene Darstellung) . . . . .	95
a	nicht angepasste Kurve . . . . .	95
b	angepasste Kurve . . . . .	95
82	Neuer Eurotherm (Quelle: eigene Darstellung) . . . . .	XV
83	Raspberry Pi und Adafruit-Module mit Pt100 - Schaltplan (Quelle: eigene Darstellung) . . . . .	XVII
84	Arduino Mega, LED, Adafruit-Modul mit Pt100 - Schaltplan (Quelle: eigene Darstellung) . . . . .	XVIII
85	DAC-Chip und Hardware-Sicherheit - Schaltplan (Quelle: eigene Darstellung) . . . . .	XIX
86	Schnittstelle 2 - Schaltplan (Quelle: eigene Darstellung) . . . . .	XX
87	USB-Hub mit Pyrometern und Eurotherm - Schaltplan (Quelle: eigene Darstellung) . . . . .	XXI
88	Versorgung und Heizer - Schaltplan (Quelle: eigene Darstellung) . . . . .	XXII
89	Leistungsmessung Grafit-Heizer - Schaltplan (Quelle: eigene Darstellung) . . . . .	XXIII
90	Magnetfeld-Profil Messung - Schaltplan (Quelle: eigene Darstellung) . . . . .	XXIV
91	Leistungsmessung Spule - Schaltplan (Quelle: eigene Darstellung) . . . . .	XXV
92	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 1 (Quelle: eigene Darstellung) . . . . .	XXVI

93	Lissajous-Figur zum Versuch Nr. 1 (Quelle: eigene Darstellung) . . . . .	XXVI
94	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 2 (Quelle: eigene Darstellung) . . . . .	XXVII
95	Lissajous-Figur zum Versuch Nr. 2 (Quelle: eigene Darstellung) . . . . .	XXVII
96	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 3 (Quelle: eigene Darstellung) . . . . .	XXVIII
97	Lissajous-Figur zum Versuch Nr. 3 (Quelle: eigene Darstellung) . . . . .	XXVIII
98	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 4 (Quelle: eigene Darstellung) . . . . .	XXIX
99	Lissajous-Figur zum Versuch Nr. 4 (Quelle: eigene Darstellung) . . . . .	XXIX
100	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 5 (Quelle: eigene Darstellung) . . . . .	XXX
101	Lissajous-Figur zum Versuch Nr. 5 (Quelle: eigene Darstellung) . . . . .	XXX
102	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 6 (Quelle: eigene Darstellung) . . . . .	XXXI
103	Lissajous-Figur zum Versuch Nr. 6 (Quelle: eigene Darstellung) . . . . .	XXXI
104	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 7 (Quelle: eigene Darstellung) . . . . .	XXXII
105	Lissajous-Figur zum Versuch Nr. 7 (Quelle: eigene Darstellung) . . . . .	XXXII
106	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 8 (Quelle: eigene Darstellung) . . . . .	XXXIII
107	Lissajous-Figur zum Versuch Nr. 8 (Quelle: eigene Darstellung) . . . . .	XXXIII
108	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 9 (Quelle: eigene Darstellung) . . . . .	XXXIV
109	Lissajous-Figur zum Versuch Nr. 9 (Quelle: eigene Darstellung) . . . . .	XXXIV
110	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 10 (Quelle: eigene Darstellung) . . . . .	XXXV
111	Lissajous-Figur zum Versuch Nr. 10 (Quelle: eigene Darstellung) . . . . .	XXXV
112	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 11 (Quelle: eigene Darstellung) . . . . .	XXXVI
113	Lissajous-Figur zum Versuch Nr. 11 (Quelle: eigene Darstellung) . . . . .	XXXVI
114	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 12 (Quelle: eigene Darstellung) . . . . .	XXXVII
115	Lissajous-Figur zum Versuch Nr. 12 (Quelle: eigene Darstellung) . . . . .	XXXVII
116	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 13 (Quelle: eigene Darstellung) . . . . .	XXXVIII
117	Lissajous-Figur zum Versuch Nr. 13 (Quelle: eigene Darstellung) . . . . .	XXXVIII
118	Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 14 (Quelle: eigene Darstellung) . . . . .	XXXIX
119	Lissajous-Figur zum Versuch Nr. 14 (Quelle: eigene Darstellung) . . . . .	XXXIX

## Tabellenverzeichnis

1	Berechnung der Reglerparameter nach Ziegler und Nichols (Quelle: [Lun10] S. 463)	6
2	Steuerzeichen - Eurotherm Befehl (Quelle: [Eurb] S. 4-2)	16
3	Wahrheitstabelle XOR (Quelle: eigene Darstellung)	18
4	Benutzte Mnemonik Befehle (Quelle: eigene Darstellung)	19
5	Befehle des DAQ6510 Multimeters von Keithley (Quelle: [Mul])	58
6	Befehle des DSOX1204G Oszilloskops von Keysight (Quelle: [Osz])	59
7	Daten der Messungen (Quelle: eigene Darstellung)	74
8	Bezeichnung und Nummer des zugehörigen Plots (Quelle: eigene Darstellung)	83
9	Berechnete und abgelesene Frequenzen, Zeiten und Winkel (Quelle: eigene Darstellung)	83
10	Ablese Werte für Strom und Spannung (Quelle: eigene Darstellung)	84
11	Berechnung von Strom, Spannung und Leistung (Quelle: eigene Darstellung)	85
12	Werte von Oszilloskop abgelesen (Quelle: eigene Darstellung)	89
13	Leistungswerte aus Mittelwertbildung verglichen mit der zeichnerischen Lösung (Quelle: eigene Darstellung)	91
14	Aufzählung der genutzten Geräte (Quelle: eigene Darstellung)	XI
15	Referenzen zu den genutzten Programm Bibliotheken für Ordner <b>Emissivität</b> (Quelle: eigene Darstellung)	XII
16	Referenzen zu den genutzten Programm Bibliotheken für Ordner <b>Emulation Eurotherm</b> (Quelle: eigene Darstellung)	XIII
17	Referenzen zu den genutzten Programm Bibliotheken für Ordner <b>Magnetfeld-messung</b> (Quelle: eigene Darstellung)	XIV

## Abkürzungs- und Formelzeichenverzeichnis

Formelzeichen:

A	$m^2$	Querschnittsfläche
$A_L$	$mm^2$	Leiterquerschnitt
B	T	Magnetische Flussdichte
c	mm	Hypothenuse (Länge)
e	%	Emissiongrad (auch $\epsilon$ )
f	Hz	Frequenz
I	A	Stromstärke
$I_{eff}$	A	Effektiver Strom
$\hat{i}$	A	Spitzen-Strom
$i_{SS}$	A	Spitzen-Spitzen Strom
$k_D$	/	Differenzierfaktor
$k_I$	/	Integriertfaktor
$k_P$	/	Verstärkungsfaktor
$k_S$	/	statische Verstärkung
L	H	Induktivität
l	m	Länge
$l_L$	m	Leiterlänge
N	Einheiten los	Windungen
n	Einheiten los	Anzahl von Datenpunkten bzw. Anzahl
P	W	Wirkleistung
$p(t)$	W	Momentanleistung
Q	var	Blindleistung
R	$\Omega$	elektrischer Widerstand (Resistanz)
$R_L$	$\Omega$	Leiterwiderstand
r	$m$	Radius
S	VA	Scheinleistung
T	s	Periodendauer
$T_1, T_2$	$^{\circ}\text{C}$	Temperaturen
$T_D$	s	Vorhaltezeit
$T_I$	s	Nachstellzeit
t	s	Zeit
U	V	Spannung
$U_{eff}$	V	Effektive Spannung
$U_{Hall}$	V	Hall-Spannung
$\hat{u}$	V	Spitzen-Spannung
$u_{SS}$	V	Spitzen-Spitzen Spannung
v	mm/s	Geschwindigkeit
X	$\Omega$	Blindwiderstand (Raktanz)
$X_L$	$\Omega$	induktiver Blindwiderstand
Z	$\Omega$	Impedanz (Scheinwiderstand)

I -> der Unterstrich deutet auf komplexe Zahl hin

$\alpha$	$\circ$	Winkel
$\Delta t$	s	Phasenverschiebungszeit
$\Delta \varphi$	Einheitenlos (Bogenmaß)	Phasenwinkel in Bogenmaß
$\epsilon$	%	Emissiongrad
$\kappa$	$m/(\Omega mm^2)$	elektrische Leitfähigkeit

$\mu_0$	$N/A^2$	Magnetische Feldkonstante
$\mu_r$	Einheiten los	Magnetische Permeabilität
$\pi$	Einheiten los	Kreiskonstante
$\rho$	$(\Omega mm^2)/m$	spezifischer elektrischer Widerstand
$\sigma$	$W/(m^2 K^4)$	Stefan-Boltzman-Konstante
$\varphi$	$\circ$	Phasenwinkel / Phasenverschiebungswinkel
$\omega$	1/s	Kreisfrequenz

Abkürzungen:

AV	Aspekt-Verhältnis
CCW	gegen Uhrzeigersinn (engl. Counter Clock Wise)
CW	im Uhrzeigersinn (engl. Clock Wise)
CZ	Czochralski-Verfahren
GUI	graphische Umgebungen (engl. Graphical User Interface)
IKZ	Institut für Kristallzüchtung
KW	Kurzwellig
LW	Langwellig

# 1 Einleitung

Diese Arbeit wurde als Anschluss an das Fachpraktikum der Hochschule für Technik und Wirtschaft am Leibniz Institut für Kristallzüchtung (kurz IKZ) absolviert. Am IKZ wurde die Arbeit in der Gruppe für Modellexperimente geschrieben. Die Gruppe erstellt verschiedene Experimente für verschiedene Kristallzüchtungsverfahren. Diese Verfahren heißen Czochralski-Verfahren (kurz CZ) und Float-Zone-Verfahren. Die Experimente werden in der Test-CZ Anlage durchgeführt. In dieser wird das gerade genannte Czochralski-Verfahren durchgeführt. In Abbildung 1 kann man die Test-CZ Anlage sehen.



Abbildung 1: Test-CZ Anlage mit Spule (Quelle: eigene Darstellung)

In dieser Arbeit werden unterschiedliche Experimente zur Messung von verschiedenen Größen wie der magnetischen Flussdichte und des Emissionsgrades vorgestellt. Sowohl die Steuerung der Experimente, also der genutzten Geräte sowie die Auswertung bzw. Verarbeitung werden von Python Programmen durchgeführt.

In der Abbildung kann man sehen das eine Spule eingebaut ist. Die Spule ist der induktive Heizer und einer der Heizer der Anlage mit dem in dieser Arbeit gearbeitet wird. Der andere Heizer ist ein Widerstandsheizer bzw. resistiver Heizer, der in den folgenden Kapiteln noch gezeigt wird. Die Spule wird in der Anlage als Heizer für die Kristallzüchtung verwendet. Das Ziel der Arbeit ist nun die Untersuchung der beiden Heizer. Eine wichtige Eigenschaft ist die Emissivität von Grafit, da dieses Material sowohl beim Widerstandsheizer als auch als Suszeptor für den induktiven Heizer genutzt wird. Diese Eigenschaft wurde dann in einem speziellen Experiment gemessen.

Das Ziel dieser Arbeit ist die Automatisierung von Experimenten und Messungen an der Test-

CZ Anlage. So sollen Automatisierungen mit Python und Vermessungen der elektromagnetischen Parameter an den beiden genannten Heizern vorgenommen werden. Dafür werden folgende Experimente durchgeführt:

- Emissionsgradmessung mit vorherigen Heiztesten am Widerstandsheizer,
- Kalibrierung eines Hall-Sensors mithilfe einer Helmholtzspule,
- Bestimmung von Magnetfeldprofilen mit den Daten der Kalibrierung,
- Leistungsmessungen an den beiden Heizern durchgeführt werden.

Die Automatisierung soll folgende Themen umfassen:

- Temperatur der Heiztests,
- Emissionsgradanpassung und -bestimmung,
- Kalibrierung des Hall-Sensors und,
- Aufnahme der Magnetfeldprofile.

Mit dem Experimenten und der Automatisierung sollen folgende Größen bestimmt werden:

- Magnetische Flussdichte,
- Emissionsgrad und,
- Leistung.

## 2 Grundlagen

Ab Kapitel 3 werden verschiedene Experimente dargestellt. Mit diesen Experimenten sollen die verschiedensten Größen gemessen werden mit vielen verschiedenen Messgeräten. Um dazu das nötige Vorwissen zu haben, werden in den folgenden Unterkapiteln diese Grundlagen erläutert. So werden die beiden Heizer der Test-CZ Anlage sowie die genutzten Messgeräte und Regler erläutert.

### 2.1 Widerstandsheizung (Resistiv)

Ein Widerstandsheizer ist ein elektrischer Leiter, der von einem Strom durchflossen wird. Durch diesen Stromfluss erwärmt sich der Leiter und strahlt diese Wärme an die Umgebung ab. In dem Experiment zur Emissionsgradbestimmung (Kapitel 4) wird ein Grafit-Heizer genutzt. Wie der Name es schon sagt ist das Material, das für den Widerstandsheizer genommen wurde, Grafit (spezifisch: CZ3/R6300). In Abbildung 2 ist der genutzte Heizer zu sehen.

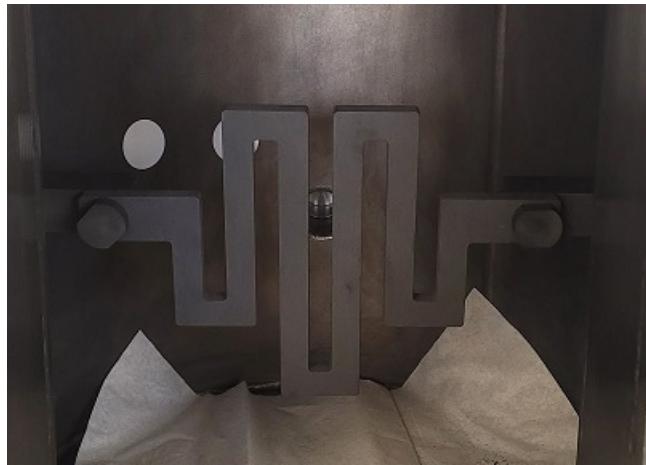


Abbildung 2: Grafit-Heizer (mäanderförmig) (Quelle: eigene Darstellung)

Auch dieser Heizer ist nur ein elektrischer Leiter, wodurch man den elektrischen Widerstand wie bei einem elektrischen Leiter berechnen kann. Die Gleichung 1 zeigt dies.

$$R_L = \frac{\rho * l_L}{A_L} \quad (1)$$

Alle Größen in der Gleichung 1 sind Material Daten des spezifischen genutzten Leiters. Dabei braucht man die Länge des Leiters ( $l_L$ ), die Querschnittsfläche des Leiters ( $A_L$ ) und den spezifischen Widerstand ( $\rho$ ) des Materials. Wenn man diesen Wert in den Datenblättern nicht findet, so kann man auch den elektrischen Leitwert ( $\kappa$ ) nutzen. Die Umrechnung wird in Gleichung 2 gezeigt.

$$\rho = \frac{1}{\kappa} \quad (2)$$

Wenn man den Leiterwiderstand hat kann man mit der Gleichung 3 die Leistung (P) bzw. die abgegebene thermische Leistung bestimmen.

$$P = I^2 * R_L \quad (3)$$

Die Gleichung 3 ergibt sich durch das Einsetzen des Ohmschen Gesetzes (Gleichung 14 auf S. 9) in die Leistungs-Formel:

$$P = U * I \quad (4)$$

## 2.2 Induktionsheizung (Induktiv)

Bei einer Induktionsheizung wird eine Spule als Heizer genutzt. Spulen sind induktive Blindwiderstände in einem Wechselstromkreis. Durch eine ideale Spule wird in solchen Stromkreisen der Strom um 90° der Spannung nacheilen. Die bedeutende Größe einer Spule ist die Induktivität. Diese lässt sich bei Zylinderspulen wie folgt berechnen. Die Gleichung ist in Quelle [Erb+12] auf Seite 87 zu finden.

$$L = \frac{\mu_0 * \mu_r * N^2 * A}{l} \quad (5)$$

Die Induktivität in Gleichung 5 lässt sich also aus den Materialeigenschaften der Spule berechnen. Von der Spule muss man die Länge (l), die Querschnittsfläche (A), die Windungszahl (N), die Permeabilitätszahl ( $\mu_r$ ) und die magnetische Feldkonstante ( $\mu_0$ ) kennen.

Für die Kalibrierung eines Hall-Sensors (Kapitel 5.1 S. 55) wird eine besondere Zylinderspule genutzt, die Helmholtzspule. Zu dieser Spule erfährt man in Kapitel 2.4.3 noch etwas mehr. Aus der Quelle [Wik] stammt die Gleichung 6 für die Berechnung der Induktivität (L) dieser Formel.

$$L = 2 * N^2 * r * \mu_0 * \left( \frac{\pi r}{l + \frac{2r}{2,2}} + \frac{4,941}{4\pi} \right) \quad (6)$$

In Abbildung 3 kann man die Test-CZ Anlage während einer Kristallzüchtung sehen. In der Spule steht ein Grafit-Tiegel mit Zinn (Sn) darin. Die Erwärmung des Tiegels erfolgt über die Spule. Diese Erwärmung erfolgt durch das Magnetfeld der Spule. Dieses entsteht, sobald die Spule von Strom durchflossen wird.



Abbildung 3: Anlage mit Grafit-Tiegel (Suszeptor) (Quelle: eigene Darstellung)

Durch Magnetfelder werden Ströme in elektrischen Leitern induziert. Wie schon erwähnt steht in der Spule ein Grafit-Tiegel. Dieser Grafit-Tiegel ist ein Suszeptor. Suszeptoren wandeln elektromagnetische Energie in thermische Energie um und geben diese wieder ab. Somit kann man Wärme an andere Objekte (metallisch oder nicht leitend) übergeben. In dem Fall wird somit das Metall für die Kristallzüchtung aufgeschmolzen. Der Vorteil dieses Verfahrens ist, dass man mit einem Induktionsheizer (Spule) auch Material erwärmen kann, das nicht elektrisch leitend ist. [Vgl. [Sus]]

Spulen sind induktive Blindwiderstände, die von der Induktivität der Spule und der Frequenz ( $f$ ) bzw. Kreisfrequenz ( $\omega$ ) abhängig sind. In Kapitel 2.4.3 in Gleichung 35 und 36 wird die Berechnungsformel dazu gezeigt.

### 2.3 PID-Regelung

In der Regelungstechnik gibt es verschiedene Regler. In dieser Arbeit wird mit einem PID-Regler gearbeitet. Der PID-Regler wird genutzt, weil dieser eine Sollwertabweichung ausgleicht und schneller als andere Regler geregelt.

Der PID-Regler besteht aus den Anteilen proportional (P), integrieren (I) und differenzieren (D). Jedes dieser Anteile hat eine Auswirkung auf die Regelung der zu regelnden Größe. Der P-Anteil sorgt für eine schnelle Regelung, aber kann die Sollwertabweichung nicht beheben bzw. ausgleichen. Der I-Anteil sorgt für den Ausgleich der Sollwertabweichung. Der D-Anteil reagiert auf die Abweichung der Regelung. Somit wird die Stellgröße durch den Regler bei Regelabweichungen stark geändert. [Vgl. [Lun10] S. 395]

Im Regelkreis wird der Regler zwischen Stelleinrichtung (Teil der Regelstrecke) und Vergleichsstelle eingeordnet. Das Eingangssignal des Reglers ist die Regelabweichung  $e(t)$  die von der

Vergleichstelle kommt. In dieser wird der Sollwert und der Istwert der zu regelnden Größe verglichen, die Differenz dieses Vergleichs wird an dem Regler übergeben. Dieser Regler berechnet anhand dieser Abweichung die Stellgröße  $u(t)$ .

$$u(t) = k_P e(t) + \frac{k_p}{T_I} \int_0^t e(\tau) d\tau + k_p T_D \frac{de(t)}{dt} \quad (7)$$

Die Gleichung 7 (Quelle [Lun10] S. 395) zeigt den Grundsatz des PID-Reglers. Diese Gleichung kann über die Laplace-Transformation in die Übertragungsfunktion des Reglers umgewandelt werden. Die Übertragungsfunktion stellt den Eingang zum Ausgang des Reglers da. Die Gleichung 8 (Quelle [Lun10] S. 394) zeigt die Übertragungsfunktion des PID-Reglers. Mit den Gleichungen 9 und 10 werden die Nachstellzeit ( $T_I$ ) und die Vorhaltezeit ( $T_D$ ) berechnen. Auch diese Formeln stammen aus der Quelle [Lun10] auf Seite 394.

$$K_{PID}(s) = k_P + \frac{k_I}{s} + k_D s = kp(1 + \frac{1}{T_I s} + T_D s) \quad (8)$$

$$T_I = \frac{k_P}{k_I} \quad (9)$$

$$T_D = \frac{k_D}{k_P} \quad (10)$$

Durch die Struktur der Gleichung 7 und 8 kann man durch einfaches weglassen der Komponenten einen anderen Regler erstellen. Lässt man z.B. den D-Teil weg, bekommt man einen PI-Regler. [Vgl. [Lun10] S. 396]

Um die Parameter des PID-Reglers näherungsweise zu bestimmen gibt es verschiedene Verfahren. Eines davon ist die Methode nach Ziegler und Nichols. Damit man diese Methode anwenden kann benötigt man die Sprungantwort des Reglers.

Tabelle 1: Berechnung der Reglerparameter nach Ziegler und Nichols (Quelle: [Lun10] S. 463)

Voraussetzung	Regler	Regelparameter
Approximation der Regelstrecke durch ein $PT_1T_t$ -Glied	P	$k_P = \frac{1}{k_S \frac{T}{T_t}}$
	PI	$k_P = \frac{0,9}{k_S} \frac{T}{T_t}, T_I = 3,33T_t$
	PID	$k_P = \frac{1,2}{k_S} \frac{T}{T_t}, T_I = 2T_t, T_D = 0,5T_t$

Bei der Tabelle 1 wurde nur der Teil aus der Quelle [Lun10] S. 463 entnommen der für diese Arbeit von Relevanz ist.

## 2.4 Sensoren

In dem Kapitel wird auf die Grundlagen der genutzten Sensoren eingegangen und in Kapitel 2.4.3 dann auf die gemessenen elektrischen Größen. Die Sensoren für die Temperatur finden ihren Einsatz in dem Experiment in Kapitel 4 (Emissionsgradbestimmung). In Kapitel 5 (Magnetfeld Profile und Kalibrierung eines Hall-Sensors) finden die Magnetfeldsensoren ihre Anwendung.

### 2.4.1 Temperatursensoren

Für die Messung der Temperatur werden in den Experimenten Pyrometer und Widerstandsthermometer genutzt. Im folgenden werden diese beschrieben.

## Pyrometer:

In dem Experiment zur Bestimmung des Emissionsgrades (Kapitel 4) werden langwellige (kurz LW) und kurzwellige (Kurz KW) Pyrometer genutzt. Mit LW und KW sind die Wellenlängenbereiche der Wärmestrahlung gemeint, in denen die Geräte sensitiv sind. Das LW Pyrometer ist ein Impac® Series 600 von Advanced Energy und das KW Pyrometer ist ein IGA 6/23 – Laser von der Firma LumaSense.

Nach der Quelle [Pyra] (S. 9)<sup>1</sup> hat das KW Pyrometer einen Spektralbereich (Wellenlänge) von 2 bis 2,6  $\mu\text{m}$ . Auch die LW Pyrometer haben einen Spektralbereich, der nach Quelle [Pyrb] (S. 3-1) bei 8 bis 14  $\mu\text{m}$  liegt.

Das Pyrometer ist ein Messgerät zur Bestimmung der Temperatur. Die Temperatur wird vom Pyrometer Berührungslos gemessen. Die Grundlage hierfür ist das jeder Körper Wärmestrahlung abgibt. Über das Stefan-Boltzmann-Gesetz kann man die Strahlungsleistung des Körpers berechnen. Dieses Gesetz wird in Gleichung 11 gezeigt. [Vgl. [Her18] S. 402]

$$P_S = \epsilon * \sigma * A * (T_1^4 - T_2^4) \quad (11)$$

$$\text{mit } \sigma = 5,671 * 10^{-8} \frac{W}{m^2 K^4} \quad (12)$$

Die Gleichung 11 und die Stefan-Boltzman-Konstante aus Gleichung 12 sind in der Quelle [Her18] auf der Seite 402 zu finden. Die dargestellte Gleichung zeigt die Form der realen Strahler. Dies wird durch den Emissionsgrad  $\epsilon$  gezeigt. Die restlichen Größen sind die Fläche des Strahlers ( $A$ ), die Temperatur des Strahlers selbst ( $T_1$ ) und die Temperatur um den Strahlers bzw. des Sensors selbst ( $T_2$ ). [Vgl. [Her18] S. 402]

Wie schon erwähnt gibt jeder Körper eine Wärmestrahlung ab, diese Strahlung wird von dem Pyrometer aufgenommen und in ein elektrisches Signal umgewandelt. Anhand dieses Signales kann man die Oberflächentemperatur des Messobjektes bestimmen bzw. das Pyrometer zeigt eine Temperatur anhand dieses Signales an. Der Temperaturmessbereich umfasst bei Pyrometern -50°C bis 4000 °C. Der große Vorteil der Pyrometer ist die Berührungslose Messung der Temperatur, da somit der Verschleiß des Pyrometers minimiert wird und Messungen an Orten möglich ist an die man nicht so einfach kommt. Der Nachteil der Pyrometer ist der Emissionsgrad von Körpern. Für eine genaue Messung muss man diesen kennen. In Kapitel 2.5 wird auf den Emissionsgrad eingegangen. Zudem ist es mit Pyrometern nur möglich die Temperatur der Oberfläche zu bestimmen. [Vgl. [Her18] S. 403]

Ein weiterer wichtiger Punkt der Pyrometer ist der Messfleck. Der Messfleck eines Pyrometers ist der Bereich der vom Pyrometer erfasst ist und aus dem die Temperatur bestimmt wird. Ist der Messfleck zu groß können Materialien, die sich in der Umgebung der Probe befinden, die gemessene Temperatur beeinflussen, da so auch deren Wärmestrahlung Einfluss auf die gemessene Temperatur nimmt. Der Messfleck des Pyrometers wird mit größeren Abstand auch größer.

---

<sup>1</sup>Die Quelle zeigt das Manual des Pyrometers. Genutzt wurde das Pyrometer von Typ MB 13.

## Widerstandsthermometer

Anders als die Pyrometer sind die Widerstandsthermometer berührende Temperatursensoren. In dieser Arbeit werden nur Pt100 Widerstandsthermometer genutzt. Diese Sensoren bestehen aus Platin und haben bei 0 °C einen Widerstand von 100 Ω. [Vgl. [Her18] S. 61]

Der elektrische Widerstand begrenzt den Strom in Materialien. Diese Begrenzung des Stromes wird durch die Temperatur größer, was einen höheren Widerstand darstellt. Somit gibt es eine Abhängigkeit zwischen elektrischen Widerstand und Temperatur. Diese Abhängigkeit wird mit der Gleichung 13 (Quelle [Her18] S. 60) gezeigt. Mit der Gleichung lässt sich ein Widerstand ( $R_0$ ) einer Bezugstemperatur ( $T_0$ ), die in der Regel 20 °C oder 0 °C ist, mit Hilfe der Temperaturkoeffizienten (a, b) des Materials in einen Widerstand ( $R_T$ ) bei einer anderen Temperatur (T) umrechnen lassen. Die gezeigte Gleichung findet bei Metallen und Metall-Legierungen Anwendung. Durch diesen Zusammenhang kann man über den gemessenen Widerstand einen Rückschluss auf die Temperatur ziehen. [Vgl. [Her18] S. 60]

$$R(T) = R_0(1 + a(T - T_0) + b(T - T_0)^2) \quad (13)$$

### 2.4.2 Magnetfeldsensoren

Für die Messung der Magnetischen Flussdichte wird ein Hall-Sensor genutzt. Der Hall-Sensor misst eine Hall-Spannung, die im späteren dann in die Magnetische Flussdichte umgerechnet wird. In Kapitel 5.1.3 werden dann die Berechnungen rund um den Hall-Sensor gezeigt. Hier soll die Funktion beschrieben werden.

Der Hall-Effekt entsteht dadurch, dass der Hall-Sensor in ein magnetisches Feld eintaucht. Um den Effekt zu erhalten muss der Sensor zum einen von Strom durchflossen sein und zum anderen senkrecht zum Magnetfeld stehen. [Vgl. [Her18] S. 47]

Durch die Nutzung einer Helmholtzspule kann man diese senkrechte Ausrichtung leicht erreichen und diese eignet sich gut zur Kalibrierung. Weitere Erklärungen zur Helmholtzspule stehen im Kapitel 2.4.3. Bei der Profilbestimmung wird eine Zylinderspule genutzt. Um dort die senkrechte Ausrichtung zu gewährleisten wurde der Hall-Sensor an der Spitze geknickt (siehe Kapitel 5.2.1 - Abbildung 53 S. 68).

Die Hall-Spannung entsteht auf folgende Weise. Dadurch dass der Strom senkrecht zu dem Magnetfeld fließt, entsteht eine Lorentz-Kraft und somit ein Elektronenmangel und ein Elektronenüberschuss auf jeweils einer Seite des Sensors. Durch die Polung der Seiten des Sensors entsteht ein elektrisches Gegenfeld, das der Lorentz-Kraft entgegen wirkt. Die beiden Kräfte wirken nun gegeneinander. Sobald sich ein Gleichgewicht der Kräfte ausgebildet hat, entsteht die Hall-Spannung die proportional zur magnetischen Flussdichte ist. [Vgl. [Her18] S. 47]

### 2.4.3 Elektrische Messungen (P, I, U, B, Phi)

In den späteren Kapiteln werden verschiedene Experimente vorgestellt und ausgewertet. In den Experimenten werden verschiedene Größen wie Spannung und Strom gemessen. In den Kapiteln werden dann auch verschiedene Messgeräte vorgestellt. Mit diesen Messgeräten werden die elektrischen Größen gemessen. So wird z.B. in Kapitel 5.3 der Strom und die Spannung mit einem Oszilloskop gemessen. In diesem Kapitel sollen die Grundlagen und Formeln für diese Messung gezeigt werden.

### **Strom (I), Spannung (U) und Leistung (P):**

Zunächst werden die Größen Strom, Spannung und Leistung beschrieben. Das grundlegende Gesetz für Strom, Spannung und den elektrischen Widerstand ( $R$ ) ist das Ohmsche Gesetz (Gleichung 14).

$$U = R * I \quad (14)$$

Das Ohmsche Gesetz kann auch auf komplexe Widerstände angewandt werden, hierbei wird das  $R$  (Wirkwiderstand) zu einem  $Z$  (Scheinwiderstand oder Impedanz).

Mit einem Oszilloskop kann man Wechselspannungen und Wechselströme messen und darstellen. So eine Darstellung kann man in Abbildung 4 sehen<sup>2</sup>.

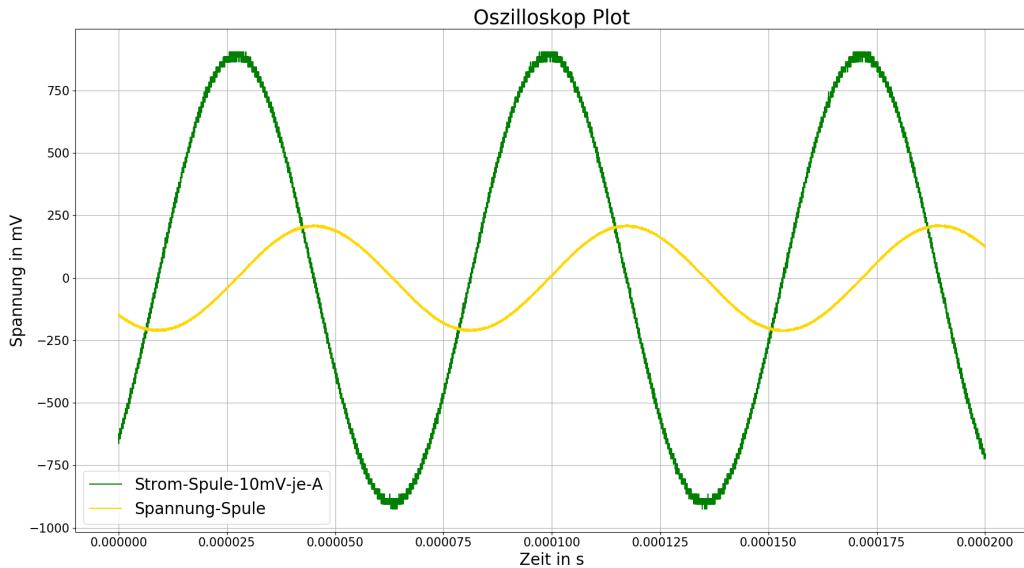


Abbildung 4: Oszilloskop-Kurven (Quelle: eigene Darstellung)

Aus einem solchen Diagramm kann man nun drei verschiedene Spannungen und Ströme ablesen. Zum einen kann man die Spitzen-Spitzen-Werte ( $u_{SS}$ ), den Spitzen-Wert ( $\hat{u}$ ) und den Effektiv- oder RMS-Wert ( $U_{eff}$ ) ablesen. In den folgenden Gleichungen kann man die Berechnung dieser sehen. Bei den Gleichungen ist es für Strom und Spannung identisch, daher wird in den folgenden Gleichungen nur die Spannung dargestellt.

$$\hat{u} = \frac{u_{SS}}{2} \quad (15)$$

$$U_{eff} = \frac{\hat{u}}{\sqrt{2}} \quad (16)$$

Aus Gleichung 15 kann man aus dem Spitzen-Spitzen-Wert den Spitzen-Wert der Größe machen und in Gleichung 16 wird dann aus dem Spitzen-Wert der Effektivwert gemacht. Bei dem Spitzen-Spitzen-Wert handelt es sich um den Betrag vom höchsten Wert der Welle bis zum niedrigsten Wert der Welle. Der Spitzen-Wert ist demnach die Hälfte dieses Wertes.

---

<sup>2</sup>Bei der Abbildung wird der Strom über ein Messgerät gemessen, das anstatt eines Stromes eine Spannung misst, dieser Umstand wird in Kapitel 5.3.1 erläutert.

Wenn man die Phasenverschiebung (kommt etwas später) und die Effektivwerte von Strom und Spannung hat, kann man Scheinleistung (S), Blindleistung (Q) und Wirkleistung (P) berechnen. Wenn man die beiden Kurven (Wellen) von Strom und Spannung multipliziert, bekommt man die Momentanleistung  $p(t)$  heraus. Aus dieser Kurve kann man durch die Bildung des arithmetischen Mittels einer bestimmten Periodenzahl (nur volle Perioden dürfen dafür genutzt werden) die Wirkleistung berechnen (Gleichung 17). Dabei ist die Kurvenart nicht von Bedeutung.

$$P = \frac{1}{n} * \sum_{i=0}^n p_i(t) \quad (17)$$

Sowohl die drei Widerstände (Wirk- (R), Blind- (X) und Scheinwiderstand (Z)) als auch die Leistungen stehen sich in einem rechtwinkligen Dreieck gegenüber. Somit kann man die drei Leistungen und Widerstände über den Satz des Pythagoras berechnen. Dieser Umstand wird für die Leistung in Gleichung 18 gezeigt und ist für die Widerstände identisch (nur Formelzeichen ändern).

$$S = \sqrt{P^2 + Q^2} \quad (18)$$

Des Weiteren kann man die Leistungen auch aus Strom, Spannung und Phasenwinkel bestimmen. Diese Formeln sehen wie folgt aus und richten sich auf volle Sinuswellen und Cosinuswellen.

$$S = U_{eff} * I_{eff} \quad (19)$$

$$P = U_{eff} * I_{eff} * \cos(\varphi) \quad (20)$$

$$Q = U_{eff} * I_{eff} * \sin(\varphi) \quad (21)$$

Wenn man die Scheinleistung in Gleichung 19 berechnet hat, kann man diese auch in die Gleichungen 20 und 21 einsetzen oder eine der Leistungen mit dem Satz des Pythagoras aus Gleichung 18 berechnen.

### Phasenverschiebungswinkel ( $\varphi$ ):

Als nächstes wird erläutert, wie man die Phasenverschiebung bestimmt. Wenn man zwei der oben genannten Leistungen hat, kann man die Gleichungen 20 und 21 nach Phi umstellen oder mit Blind- und Wirkleistung den Tangens bilden und auch nach Phi umstellen (Gleichung 22). Die andere Lösung wäre es, den Phasenwinkel zeichnerisch aus dem Oszilloskop-Bild der Strom- und Spannungswelle zu bestimmen oder die Cursor-Funktion und Messfunktionen des Oszilloskops zu nutzen.

$$\varphi = \arctan\left(\frac{Q}{P}\right) \quad (22)$$

Die Phasenverschiebung zwischen Strom und Spannung wird durch den genutzten komplexen Widerstand verursacht. Wenn man einen einfachen Ohmschen Widerstand (Wirkwiderstand oder Resistanz genannt) an Wechselspannung anschließt, so wird die Phasenverschiebung idealerweise Null sein. Strom und Spannung haben denselben Nulldurchgang und liegen vom Aussehen der Kurven übereinander. Hat man aber einen Blindwiderstand (auch Reaktanz genannt), so wird die Phasenverschiebung im idealen Fall  $90^\circ$  betragen. Bei einer Spule (induktiver Widerstand) eilt der Strom der Spannung um  $90^\circ$  hinterher. Bei einem kapazitiven Blindwiderstand wie einem Kondensator ist dieser Umstand umgedreht und somit eilt der Strom um  $90^\circ$  der Spannung voraus. In den Gleichungen für die Berechnung der Leistungen wird der Betrag der Phasenverschiebung genutzt.

Bei der Bestimmung der Phasenverschiebung muss man darauf achten, dass die beiden Wellen dieselbe Frequenz haben und dass die Punkte zur Bestimmung auf derselben Höhe liegen. Somit kann man z.B. die Nulldurchgänge oder Spitzenwerte der Kurven nehmen. Des Weiteren muss man darauf achten, dass die beiden Punkte im selben Bereich liegen, also dass die Kurve in beiden Fällen z.B. abfällt.

Als nächstes werden die Berechnungs-Möglichkeiten für den Phasenverschiebungswinkel gezeigt. Aus der Zeichnung (aus dem Diagramm) kann man den Winkel Phi auf zwei Arten bestimmen. Zum einen kann man die Phasenverschiebungszeit  $\Delta t$  aus der Zeichnung ablesen, wenn die x-Achse über der Zeit abgebildet wurde. Die Zeit ist zwischen den ausgewählten Punkten zu finden. Mit Gleichung 23 wird diese Zeit dann in das Bogenmaß des Winkels umgerechnet.

$$\Delta\varphi = 2\pi f * \Delta t \quad (23)$$

Wenn die x-Achse im Bogenmaß angegeben ist, kann man das Bogenmaß direkt ablesen und mit Gleichung 24 in einen Winkel umrechnen. Die Gleichung zeigt eine Verhältnis-Gleichung (Dreisatz). Diese Art Gleichung wird in den Berechnungen in dieser Arbeit oft verwendet. So wurde auch die Phasenverschiebungszeit von einer Länge über den Dreisatz in eine Zeit umgerechnet (siehe z.B. Abbildung 92 auf S. XXVI).

$$\varphi = \frac{\Delta\varphi * 180^\circ}{\pi} \quad (24)$$

Die andere Lösung der Berechnung des Phasenwinkels ist eine solche Verhältnis-Gleichung. Wenn man die Länge der Phasenverschiebungszeit und die Periodendauer ( $T$ ) hat, kann man den Phasenverschiebungswinkel daraus berechnen.

$$\varphi = \frac{360^\circ * l_{\Delta t}}{l_T} \quad (25)$$

$$\varphi = \frac{360^\circ * \Delta t}{T} \quad (26)$$

Die beiden Gleichungen 25 und 26 zeigen dasselbe. In Gleichung 25 werden die ausgemessenen Längen der Zeichnung verwendet und in Gleichung 26 werden die über den Dreisatz umgerechneten Zeiten genutzt.

Des Weiteren kann man die Gleichungen 26 und 23 ineinander umrechnen. Dafür muss man folgendes in Gleichung 23 einsetzen.

$$T = \frac{1}{f} \quad (27)$$

$$360^\circ = 2\pi \quad (28)$$

Wenn man die Gleichungen 27 und 28 nun in die Gleichung 26 einsetzt und die Gleichung ausrechnet, kommt Gleichung 23 heraus. Diese Umrechnung soll mit Gleichung 29 gezeigt werden. Durch das  $2\pi f$  bekommt man anstelle eines Winkels das Bogenmaß zurück, da dieser Teil der Winkelgeschwindigkeit  $\omega$  entspricht.

$$\varphi = \frac{360^\circ * \Delta t}{T} = \frac{2\pi * \Delta t}{\frac{1}{f}} = 2\pi f * \Delta t = \Delta\varphi \quad (29)$$

Mit Gleichung 27 wird auch der Zusammenhang zwischen Frequenz und Periodendauer gezeigt. Zudem entspricht die Periodendauer T einem Vollwinkel von  $360^\circ$  und somit  $2\pi$ . Mit den Gleichungen 23 bis 35 wurden die beiden Berechnungsansätze für die Zeichnerische Lösung dargestellt.

Eine andere Methode der Abschätzung des Phasenwinkels gibt die Lissajous-Figure. Durch den Plot der y-Daten der Sinus-Kurven des Oszilloskops, also zweier Kanäle, bekommt man diese Kurve heraus. Dafür müssen die y-Daten des einen Kanals als x-Daten genutzt werden.<sup>3</sup> Um diese Kurve darstellen zu können, müssen beide Sinus-Kurven (bzw. angezeigte Kurven) dieselbe Frequenz haben. Durch den Phasenwinkel wird die Form dieser Kurve verändert. Bei einem Winkel von  $0^\circ$  wird die Kurve eine Gerade sein und bei  $90^\circ$  wird die Kurve ein Kreis sein. In Kapitel 5.3.3 werden diese Kurven für die Leistungsmessung gezeigt. Durch das bloße Ansehen der Kurve kann man den Phasenwinkel nur grob abschätzen. [Vgl. [Ber20] S. 67 und 69]

### **Magnetische Flussdichte (B):**

In Kapitel 2.4.2 wurde der Sensor für die Magnetfeldmessung beschrieben bzw. der Effekt dahinter. Die Messgröße ist die magnetische Flussdichte, die über die Hall-Spannung bestimmt wird.

Um den Hall-Sensor kalibrieren zu können wird eine Helmholtzspule genutzt. Diese Art Spule könnte man als eine Zylinderspule mit fehlenden Windungen beschreiben, bzw. man könnte es auch als zwei Spulen aus demselben Draht beschreiben. Durch die Lücke kann man besser in die Spule eingreifen.

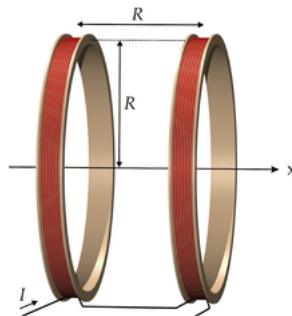


Abbildung 5: Helmholtzspule ( $R = \text{Radius}$ ) (Quelle: [Wik])

In Abbildung 5 kann man die Besonderheiten der Spule sehr gut sehen. Zum einen muss der Abstand der beiden Spulen dem Radius der Spule entsprechen und zum anderen müssen beiden Spulen dieselbe Wickelrichtung besitzen. Des Weiteren müssen sie parallel nebeneinander stehen und vom Strom in derselben Richtung durchflossen werden. Die Abbildung 5 zeigt eine Helmholtz-Spule. Die Lücke in der Spule ist die große Besonderheit einer Helmholtz-Spule. Durch diesen kann man leicht im homogenen Bereich der Spule arbeiten, da man nicht an die Windungen stößt. Ein weiterer Vorteil ist das der homogene Bereich so leichter untersuchbar ist. Die Spule wurde von Hermann von Helmholtz, einem Physiker aus Deutschland, entwickelt. [Vgl. [Hel]]

Des Weiteren findet man in der Quelle [Hel] die Berechnungsformel für die Helmholtzspule. Die genutzte Helmholtzspule wird in Gleichung 65 (S. 62) berechnet. Um die Magnetische Flussdichte zu berechnen benötigt man die Windungszahl (N) einer Spulen-Hälfte, den Radius ( $r$  - in Abbildung 5 mit  $R$  dargestellt und in der Gleichung 30 angepasst), den Strom ( $I$ ) durch die Spule und die magnetische Feldkonstante ( $\mu_0$ ).

<sup>3</sup>Das in Kapitel 5.3.2 beschriebene Programm liest die y-Daten der Kanäle aus und speichert diese in einer Text-Datei. Diese Dateien können ausgelesen werden und gegeneinander geplottet werden.

$$B = \mu_0 * \frac{8 * N}{\sqrt{125 * r}} * I \quad (30)$$

$$\text{mit } \mu_0 = 1,2566 * 10^{-6} \frac{N}{A^2} \quad (31)$$

$$[B] = 1 \frac{\frac{N}{A^2} * A}{m} = 1 \frac{\frac{kg * m}{s^2}}{m} = 1 \frac{kg * \eta h * A}{A^2 * s^2 * \eta h} = 1 \frac{kg}{A * s^2} = 1 T \quad (32)$$

Mit Gleichung 32 wird die Einheitenbetrachtung der Gleichung 30 gezeigt. Für die Magnetische Flussdichte kommt Tesla heraus.

Aus der Gleichung 30 kann man die Proportionalität zum Strom sehen. Da der Radius, die Windungen und die magnetische Feldkonstante konstante Werte sind, ist die magnetische Flussdichte direkt proportional zum Strom. Aus dem Ohmschen Gesetz, bei komplexen Widerständen, weiß man, dass der Strom durch den induktiven Widerstand frequenzabhängig wird, was in den Gleichungen 33 bis 36 zu sehen ist.

$$\underline{I} = \underline{U} * \underline{Z} \quad (33)$$

$$Z = \sqrt{R^2 + X^2} \quad (34)$$

$$X_L = \omega L \quad (35)$$

$$\omega = 2\pi f \quad (36)$$

Der induktive Bindwiderstand (Gleichung 35) und auch der kapazitive Blindwiderstand enthalten die Frequenz durch die Kreisfrequenz  $\omega$ . Wenn man nun aber die Spannung verringert oder vergrößert, kann man somit den Strom konstant halten, wodurch die magnetische Flussdichte wieder frequenzunabhängig wird. Dies wird auch in Kapitel 5.1 bei der Kalibrierung getan.

## 2.5 Emissivitäten

Wird ein Körper von Wärmestrahlung getroffen, so wird diese Strahlung absorbiert, reflektiert und/oder transmittiert. Die absorbierte Strahlung in einem Körper wird aber auch wieder emittiert. Die Wärmestrahlung ist proportional zur Emissivität bzw. Emissionsgrad. Der Emissionsgrad wird entweder als einheitenlose Zahl oder als Prozentzahl angegeben. Des Weiteren ist der Emissionsgrad eine Materialeigenschaft. Ein Emissionsgrad von 1 (100 %) wird dem sogenannten Schwarzen Strahler oder idealen Strahler zugeordnet. Jegliche Strahlung wird von einem solchen Strahler absorbiert und auch wieder emittiert. Der Emissionsgrad für reale Strahler (z.B. Grafit) liegt somit zwischen 0 und 1 bzw. 0 % und 100 %. Dieser Umstand ist der oben genannten Reflexion und Transmission der Strahlung zuzuschreiben. Weiterhin hängt der Emissionsgrad von der Wellenlänge der Strahlung ab. [Vgl. [Pas]]

Der Emissionsgrad ist für die Pyrometer, die in Kapitel 2.4.1 erwähnt werden, von großer Wichtigkeit. Um mit den Pyrometern richtig messen zu können muss der Emissionsgrad des Messkörpers bekannt sein. In Kapitel 4 wird ein Experiment zur Bestimmung der Emissionsgrade vorgestellt. Dafür wird die gemessene Temperatur der Pyrometer bei  $e = 100\%$  aufgenommen und mit der Temperatur eines Pt100 auf der Oberfläche der Messprobe verglichen. Anhand dieses Vergleiches wird der Emissionsgrad solange angepasst bis die Temperaturen nahezu identisch sind.

### 3 Temperatur-Regelung mit PID

Dieses Kapitel und das folgende Kapitel 4 gehören zusammen. In diesem Kapitel wird die Temperaturregelung mit dem PID-Regler erläutert. In Kapitel 4 wird dann das Experiment erläutert wozu man die Temperaturregelung benötigt. Auch das Programm zur Regelung der Temperatur bzw. der Sollwertregelung werden an der Stelle erläutert.

In diesen Kapitel werden die Regler-Geräte vorgestellt, die die Überwachung von Sollwert und Istwert übernehmen sollen. Um genau zu sein wird hier der Regler von Eurotherm vorgestellt, sowie der Ersatz durch Arduino. In den folgenden Unterkapiteln wird dann der Grund für den Ersatz des Eurotherms näher erläutert. Zunächst werden die Grundlagen (Kapitel 3.1 - 3.3) zum Eurotherm gezeigt.

Ein Eurotherm-Regler ist ein Regler, der aus verschiedenen Steckkarten besteht. Über die Steckkarten erhält der Regler seinen Mikroprozessor, die Module und Funktionen sowie die Versorgung vom Netz. [Vgl. [Eura] S. 2-4]

Über die Anschlüsse des Eurotherm-Reglers können dann verschiedene Temperatursensoren, wie Thermoelement und Widerstandsthermometer angeschlossen werden. [Vgl. [Eura] S. 2-7]

Ein Arduino ist ein Mikrocontroller, mit dem man die verschiedensten Sachen machen kann. Über die Anschlüsse kann man z.B. LEDs blinken lassen oder als Emulation für den Eurotherm dienen. Über das zugehörige Arduino Programm kann man den Arduino mit Code beschreiben. Der Arduino kann immer nur ein Programm in sich haben. Nach dem das Programm auf dem Arduino ist läuft dieser mit diesem Programm, bis es überschrieben wird.

#### 3.1 Eurotherm Regler (Variante 1)

Für die Temperaturregelung soll ein Regler von der Firma Eurotherm genutzt werden. Der derzeitige Regler von Eurotherm an der Test-CZ Anlage ist das Modell 902P. Zurzeit wurde dieser mit Hand bedient. Dieser Umstand soll mit der Programmierung des Eurotherms geändert werden.

Während der Arbeiten mit dem Eurotherm sind verschiedene Probleme aufgetreten. Aufgrund der Fehler wurden drei Eurotherm Geräte verwendet: 1x 902P und 2x 905S.

Folgende Fehler sind bei den Eurotherm-Reglern aufgetreten:

1. Über Schnittstelle keine Kommunikation (902P),
2. Temperaturanzeige fehlerhaft und stark schwankend (905S),
3. 10 V Spannungs-Ausgang konnte nicht angesprochen werden (905S).

Aufgrund der Fehler wurden eine Emulation (Kapitel 3.4 - Seite 19) als Variante 2 entwickelt und eine Kombination aus dem dritten Eurotherm (dritter Fehler) und der Emulation (Kapitel 3.5 - Seite 27) als Variante 3 erarbeitet.

In den folgenden Kapiteln werden unterschiedliche Teile für die Temperatur-Regelung und den Eurotherm-Regler erläutert. Es werden die anderen beiden Varianten der Regelung und die Spezifikationen des Eurotherms wie Schnittstelle und Kommunikationsbefehle erläutert.

#### 3.2 Digitale Schnittstelle am Eurotherm Regler

Um ein Gerät mit einem Computer oder ähnlichem anzusprechen muss man die richtige Schnittstelle nutzen. Bei dem Eurotherm-Regler kann man entweder die Geräteschnittstelle RS232 oder RS485 nutzen. Der Eurotherm-Regler besitzt auf der Rückseite Klemmenanschlüsse. An diese

Anschlüsse kommen neben der Spannungsversorgung der Temperaturregler (Widerstandsthermometer Pt100) und der Anschluss der Schnittstelle. Je nachdem was man hat wird dies anders verdrahtet bzw. angeklemmt.

X1	TX1 (+)
X2	TX1 (-)
X3	RX1 (+)
X4	RX1 (-)
X5	COM 1

Abbildung 6: Klemmenanschluss - Eurotherm 905S (900 EPC) (Quelle: [Eura] S. 2-13)

In Abbildung 6 kann man die Klemmen sehen, an denen die Schnittstelle befestigt werden muss. Für RS485 müssen alle Anschlüsse verwendet werden. Für die Temperaturregelung mit Eurotherm wird aber RS232 als Schnittstelle genutzt. Für die Nutzung von RS232 müssen die Klemmen X1, X3 und X5 verwendet werden. [Vgl. [Eura], Seite 2-13.]<sup>4</sup>

Für die Nutzung der Schnittstelle muss der RS232 Stecker auf USB Kabel geändert werden, heißt ein USB-RS232 Kabel muss genutzt werden. Wenn diese Verbindung hergestellt ist, kann man mit dem Python-Programm die Schnittstelle initialisieren und nutzen. Um eine Schnittstelle nutzen zu können, muss man die *Python Bibliothek - serial* (Quelle [Pys]) nutzen.

```

1 import serial
2
3 ser_py = serial.Serial(
4     port = "COM11",
5     baudrate = int(9600),
6     parity = "E",
7     stopbits = int(1),
8     bytesize = int(7),
9     timeout = 2.0)
10 print('Eurotherm Initialisiert')
11

```

Abbildung 7: Programmcode - Schnittstelle (Quelle: eigene Darstellung)

Die Abbildung 7 zeigt den allgemeinen Programmcode für das Arbeiten mit der seriellen Schnittstelle. Fast alle Geräte (Pyrometer und Heizer), die hier in der Temperaturregelung genutzt werden, verwenden diese Programmzeilen. Diese Zeilen werden dann z.B. in den Geräte-Bibliotheken *heizer.py* und *pyrometer.py* genutzt. Über die Parameterliste werden dann die Variablen *port*, *baudrate*, *parity*, *stopbits* und *bytesize* übergeben. Die Schnittstelle wird dann in der Variable *ser\_py* gespeichert. Die Variable wird hier im Beispiel genommen, aber auch für die Geräte. Von der Python Bibliothek *serial* werden die Funktionen *write* und *readline* genutzt. Um die Befehle richtig zu benutzen und zu empfangen muss man die Funktionen *encode* und *decode* nutzen. Wie im Kapitel 3.1 schon erwähnt gab es Probleme mit der Schnittstelle beim Eurotherm 902P. Die Ursache für das Problem muss nicht das Gerät sein, es könnten auch die Schnittstelle am

<sup>4</sup>Die Seitenzahl bei den Quellen für Manuals, heißen nicht von Seite bis Seite, sondern geben Kapitelnummer-Seite im Kapitel an!

Computer oder das Schnittstellenkabel sein. Um dies zu überprüfen wurde ein Echotest durchgeführt.

### Echotest:

Mit einem Echotest werden Schnittstellen und die Kommunikation mit dem Computer getestet. Für dieses Unterfangen werden Befehle vom Computer über die Schnittstelle an ein Gerät gesendet und wieder vom Gerät zurückgegeben bzw. sofort vom Programm/Computer ausgelesen. Dieser Echotest wurde mit einem Arduino Mega durchgeführt, ein Beispiel kann man in der Quelle [Ard] finden. Als Konsequenz aus den Tests kam hervor, dass die Schnittstelle funktioniert. Zum anderen wurde auch mit dieser Methode geschaut, wie die Befehle (siehe Kapitel 3.3) encodiert und decodiert werden. Nachdem die Funktion der Schnittstelle und die Gesamtheit der Adressen am Eurotherm 902P geprüft wurden, wurde es ausgetauscht. Mit dem neuen Gerät (905S) funktionierten die Befehle des Eurotherms und die Kommunikation über die Schnittstelle.

## 3.3 Euotherm Befehle

Um über die Schnittstelle mit dem Eurotherm zu kommunizieren benötigt man bestimmte Befehle. Bei den verschiedenen Reglern von Eurotherm werden verschiedene Protokolle zur Kommunikation genutzt. Das Protokoll, das hier genutzt wurde, ist das EI-Bisynch Protokoll, welches auf ANSI X3.28-2.5 A basiert. [Vgl. [Eurb], Seite 4-1.]

Die Quelle [Eurb] bezieht sich auf die 2000er Serie von Eurotherm-Reglern. Doch auf den Seiten 4-1 bis 4-6 werden die Lese- und Schreibbefehle für das oben genannte Protokoll genannt. Nach einer Anfrage zu den Befehlen beim Eurotherm Kundenservice wurde ein Dokument übermittelt, was die Befehle für die Eurotherm-Regler der 900er Serie zeigt. Diese Befehle sind bei den beiden Serien identisch.

Im folgenden werden die Befehle kurz erläutert. Die Befehle bestehen aus Steuerzeichen, einer Nummer zur Identifikation des Gerätes und dem Befehl selbst. Die Befehle für die Geräte werden Mnemonic genannt. Eine Mnemonic-Tabelle kann man in der Quelle [Eura] auf Seite 7-6 bis 7-11 sehen. Wie schon erwähnt werden Steuerzeichen in den Befehlen genutzt, diese sind in Tabelle 2 zu sehen.

Tabelle 2: Steuerzeichen - Eurotherm Befehl (Quelle: [Eurb] S. 4-2)

Hex Value	Name	Usage
02	STX	Start of data in a message
03	ETX	End of message
04	EOT	End of transmission sequence
05	ENQ	Enquiry for a value
06	ACK	Positive Acknowledge
15	NAK	Negative Acknowledge

### Lesebefehl:

Programm: [EOT](GID)(GID)(UID)(UID)(C1)(C2)[ENQ]  
Antwort Eurotherm: [STX](C1)(C2)<DATA>[ETX](BCC)

Die Kanal-Nummer wird hier weggelassen, da diese bei der Kommunikation mit dem Eurotherm bisher keine Rolle spielte, da keine Kanäle genutzt werden. Beim Lesebefehl werden die Steuerzeichen EOT und ENQ an das Gerät gesendet. Zwischen diesen steht die Adresse (GID - Gruppennummer und UID - Einheitennummer), jeweils zweimal zur Validierung, und der Befehl

(C1 und C2) aus der Mnemonic-Liste. Manche der Mnemonic-Befehle können auch drei Zeichen beinhalten, z.B. wenn einer der beiden Regelkreise genutzt wird. Das Gerät antwortet dann mit den Steuerzeichen STX und ETX. Die Daten beginnen immer mit dem gesendeten Befehl und dann dem gewollten Wert. Das BCC ist eine Blockprüfnummer, diese wird bei den Schreibbefehlen näher beschrieben. [Vgl. [Eurb], Seite 4-3.]

Bei Schreib- und Lesebefehl sendet das Python-Programm die Steuerzeichen als Hexadezimal-Zahl (siehe Tabelle 2) und die restlichen Zeichen als ASCII-Zeichen.

### Beispiel

Programm: \x040033PV\x05  
Antwort Eurotherm: \x02PV100.5\x03\*

### Schreibbefehl:

Programm: [EOT](GID)(GID)(UID)(UID)[STX](C1)(C2)<DATA>[ETX](BCC)  
Antwort Eurotherm: [NAK] oder [ACK]

Bei den Schreibbefehlen wird nun vom Python-Programm bzw. dem Benutzer ein Befehl mit Daten an das Gerät gesendet. Wie beim Lesebefehl wird hier erst die Adresse genannt und dann der Befehl und der Wert. Im Unterschied zu dem Lesebefehl wird der BCC Wert nun vom Benutzer gefordert und das Gerät gibt lediglich ACK (Befehl angenommen) oder NAK (es gibt ein Problem mit dem Befehl) zurück. Die Steuerzeichen sind hier nun EOT, STX und ETX für den Schreibbefehl. Der BCC kann jedes ASCII Zeichen sein. Berechnet wird es über die logische Funktion XOR (Exklusiv Oder). Hierfür werden alle Zeichen nach dem STX bis zum ETX einschließlich dem ETX verrechnet. [Vgl. [Eurb], Seite 4-5.]

### Beispiel

Programm: \x040033\x02SL120.0\x031  
! Antwort Eurotherm: \x06

Die hintere 1 ist die Blockprüfsumme (BCC). Ob man nun die Zeichen als Hexadezimal-, Binär- oder als Dezimalzahl nimmt ist egal, man muss die ASCII Zeichen nur umwandeln und alles einheitlich mit XOR verknüpfen. Folgend wird diese Berechnung am Beispiel gezeigt:

S XOR L XOR 1 XOR 2 XOR 0 XOR . XOR 0 XOR ETX = BCC

```
01010011 XOR
01001100 =
00011111 XOR
00110001 =
00101110 XOR
00110010 =
00011100 XOR
00110000 =
00101100 XOR
00101110 =
00000010 XOR
00110000 =
00110010 XOR
00000011 =
00110001 = 31h = 1 (ASCII)
```

Für das Beispiel wurde die Binäre Schreibweise gewählt, da man daran das XOR besser verfolgen kann. Für die Berechnung auf diese Weise wird das Ergebnis von zwei Zeichen mit dem folgenden Zeichen verrechnet. Die Wahrheitstabelle von XOR wird in Tabelle 3 gezeigt.

Tabelle 3: Wahrheitstabelle XOR (Quelle: eigene Darstellung)

Eingang 1	Eingang 2	Ausgang
0	0	1
0	1	0
1	0	0
1	1	1

Dieser Wert wird im Python-Programm berechnet. Beim Lesebefehl wird er zum Prüfen des Ergebnisses berechnet. Die Code-Zeilen stammen aus **heizer.py** und sehen für diese Berechnung wie folgt aus:

#### Code-Zeilen:

```
def bcc(self, string):
    bcc_list = []
    for c in string:
        dec = ord(c)
        bcc_list.append(dec)
    bcc_list.append(3)
    bcc = 0
    for item in bcc_list:
        bcc = (bcc^item)
    if log == True:    logging.info(f'BCC (DEC) = "{bcc}"')
    return chr(bcc)
```

Diese Code-Zeilen stehen in dem Bibliotheks-Programm *heizer.py* (im Kapitel 7.2 - Anhang zu finden) in der *HeizerEurotherm* Klasse. Die Hauptfunktionen der Klasse sind *read* und *send*, welche die Lese- und Schreibbefehle darstellen sollen. Bei beiden Befehlsarten wird nach einem Befehl immer der Mnemonic EE abgefragt, welcher eine Fehlermeldung beinhaltet. Die *bcc* Funktion nimmt den zu sendenden String oder beim Lesen den erhaltenen String und zerlegt diesen in seine Einzelzeichen, um jedes Zeichen in seine Dezimalschreibweise zu ändern. Danach wird in der Funktion dann der BCC mit XOR (Rechenzeichen in Python ist das `^` und anschließend wieder in ein ASCII-Zeichen umgewandelt. Zum Ende der Funktion hat man dann nur noch ein Zeichen, welches dann an den Schreibbefehl gehängt wird.

Zum Schluss dieses Kapitels werden die genutzten Mnemonic-Zeichen in der Tabelle 4 aufgezählt. In der Spalte *Verwendung* steht dann immer, wie der Befehl in dem Programm genutzt werden kann.

Tabelle 4: Benutzte Mnemonik Befehle (Quelle: eigene Darstellung)

Nr.	Mnemonik Befehl	Erklärung	Verwendung
1.	II	Identifikation/Name des Gerätes	Lesen
2.	EE	Fehlermeldung	Lesen
3.	PV	Istwert	Lesen
4.	SL	Sollwert	Lesen, Schreiben
5.	V0	Software Version	Lesen
6.	HS	Sollwertgrenze Max	Lesen
7.	LS	Sollwertgrenze Min	Lesen
8.	11H	Istwertgrenze Max	Lesen
9.	11L	Istwertgrenze Min	Lesen
10.	OP	Ausgangsleistung in %	Lesen
11.	HO	Max. Ausgangsleistung in %	Lesen, Schreiben
12.	XP	Proportional Band (PID-Parameter - P-Glied)	Lesen, Schreiben
13.	TI	Integral Zeit (PID-Parameter - I-Glied)	Lesen, Schreiben
14.	TD	Ableitungszeit (PID-Parameter - D-Glied)	Lesen, Schreiben

Wie man in der Tabelle in der Quelle [Eura] auf Seite 7-6 bis 7-11 sehen kann gibt es noch viele weitere Befehle. Für den Zweck der Temperaturmessung und der Regelung dieser Größe sind nicht mehr Befehle notwendig.

### 3.4 Eurotherm Emulation mit Arduino (Variante 2)

Wie schon in der vorherigen Kapiteln erwähnt gab es Probleme mit den Eurotherm-Reglern. Aus diesem Grund wird in dem Kapitel die erste Ersatz-Variante erläutert und auch das dazugehörige Emulations-Programm.

Die Programme die hier beschrieben werden bzw. genannt werden sind in Kapitel 7.2 ab Seite XII im Anhang zu finden.

#### 3.4.1 Software der Emulation

In der Variante wird der Eurotherm vollständig ersetzt mit einem Arduino Mega. Dieser Arduino Mega läuft dann mit den Programmen hauptprogramm.py und heizer.py (Beschreibung in Kapitel 4.2). Aus den Programmen wird der Arduino über die Klasse *HeizerEurotherm* gesteuert. Das Arduino-Programm wurde zum großen Teil vom IKZ gestellt. Für diese Arbeit und diese Variante wurde dann die Nachbildung der Eurotherm-Befehle erstellt. Der Arduino bekommt dieselben Befehle wie der Eurotherm-Regler 905S. Diese Befehle wurden ausgiebig im Kapitel 3.3 erläutert.

Das Arduino Programm *arduino\_test\_eurotherm\_testpid.ino.ino* beginnt die Bearbeitung der Eurotherm-Befehle in der Loop-Funktion<sup>5</sup>. In dem Loop wird jedes Zeichen einzeln aus der

---

<sup>5</sup>Eine Referenz zu den Sprachbefehlen von Arduino kann man in der Quelle [Ref] finden

Schnittstelle des Computers (hier Raspberry Pi 400) gezogen und bearbeitet. Da die Eurotherm-Befehle sehr lang sind, sollte man zwischen Senden und Auslesen ein kleines Delay einbauen. Dieses Delay ist hier auf 0,5 s gesetzt worden.

Die Bearbeitung der Befehle funktioniert über Boolesche Variablen und über eine Programmierung mit switch-case. In der switch-case-Methode werden an bestimmten Steuerzeichen Boolesche Variablen gesetzt und sonst ignoriert. Die anderen Zeichen werden ausgelesen und zusammengehängen, bei bestimmten Strings wie dem Mnemonik-Befehl und der Adresse wird der String (Variable *eingabe*) gelöscht, wodurch am Ende nur ein Zahlenwert in dem String steht. Die Adresse wird gelöscht, da es sich hier nur um eine Emulation handelt, das Programm in Arduino ist also nur auf die Adresse 03 bzw. wegen der Validierung auf 0033 ausgelegt. Die Mnemonik-Befehle werden vor dem Löschen des Strings in einer separaten Variable gespeichert. Die switch-case-Zeilen prüfen die ankommenden ASCII Zeichen und setzen in den String (Variable *eingabe*) dann die Zahlen und Buchstaben ein, alle nicht einprogrammierten ASCII Zeichen werden vom Programm ignoriert. Derzeitig werden nur die Zeichen der Befehle in Kapitel 3.3 im Arduino-Programm erkannt sowie alle Zahlen und die Steuerzeichen.

### **Wichtige boolesche Variablen sind:**

1. einlesen,

- wird bei ETX (Schreibbefehl) und ENQ (Lesebefehl) auf FALSE gesetzt (sagt Befehl fertig eingelesen)
- löst die Bearbeitung des Befehls aus (Wert schreiben oder Wert auslesen) und das Zurücksenden an das Python Programm
- beim Schreibbefehl wird somit der BCC Wert ignoriert
- wird nach Abarbeitung des Befehls wieder auf TRUE gesetzt

2. schreiben,

- wird durch STX auf TRUE gesetzt
- sagt dem Programm, dass ein Schreibbefehl vorliegt
- wird nach Abarbeitung des Befehls wieder auf FALSE gesetzt

3. lesen,

- wird durch ENQ auf TRUE gesetzt
- sagt dem Programm, dass ein Lesebefehl vorliegt
- wird nach Abarbeitung des Befehls wieder auf FALSE gesetzt

4. befehl\_fertig.

- wird durch ENQ auf TRUE gesetzt
- der Befehl ist notwendig, da das Programm nach dem Einlesen des Lesebefehls noch einmal in die if-Anweisung springen soll, damit die Bearbeitung der Ausgabe durchgeführt wird
- der Lesebefehl endet mit ENQ, wodurch die serielle Schnittstelle nicht mehr erreichbar ist (im Programm wird dies Zeile nicht ausgeführt: `Serial.available() > 0`), weshalb man in der Programmierung noch einen weiteren Durchlauf der if-Abfrage braucht
- wird nach Abarbeitung des Befehls wieder auf FALSE gesetzt

Die Problematik des 4. Punktes in der Liste kann beim Schreibbefehl ignoriert werden. Jeder Schreibbefehl endet mit dem BCC Wert, der hier vom Programm zwar eingelesen wird, aber sonst ignoriert wird. Durch das Umstellen von *einlesen* kommt der BCC-Wert nicht im switch-case an. Nach dem Senden der Antwort an das Programm werden die String-Variablen geleert und die Booleschen Variablen wie in der Liste genannt zurückgesetzt.

Mit der Funktion *void Eurotherm(String befehl, String value, bool solllesen, bool sollschreiben)* im Arduino Code wird die Ausgabe bearbeitet. Hierbei wird der Mnemonik-Befehl (in Variable *befehl* bzw. Variable *code*) anhand der Booleschen Variable *lesen* oder *schreiben* bearbeitet. Liegt ein Lesebefehl vor, so wird der Wert in *value* bzw. *eingabe* unbeachtet gelassen und der Arduino sucht den gesuchten Wert und gibt ihn wie in Kapitel 3.3 zurück. Um genau zu sein wird beim Lesebefehl nun der BCC berechnet, damit das Python-Programm ihn dann vergleichen kann. Würde Arduino diesen nicht zurückgeben, würde das Python-Programm an der Stelle Fehler ausgeben.

Beim Schreiben eines Befehls wird der Wert beachtet und an der richtigen Stelle im Arduino-Code eingeschrieben. Das Programm gibt darauf immer das ACK zurück. Neben diesem Umstand gibt es noch weitere Vereinfachungen gegenüber dem richtigen Eurotherm. Diese Einsparungen sind:

1. Die Antwort für das Schreiben ist immer ACK,
2. Der BCC vom eingehenden Befehl wird ignoriert,
3. II, EE und V0 geben feste Werte zurück,
4. EE sagt immer, dass es keine Fehler gibt (Rückgabe = 0000).

Mit der Abbildung 8 soll das zuvor beschriebene Programm noch einmal veranschaulicht werden.

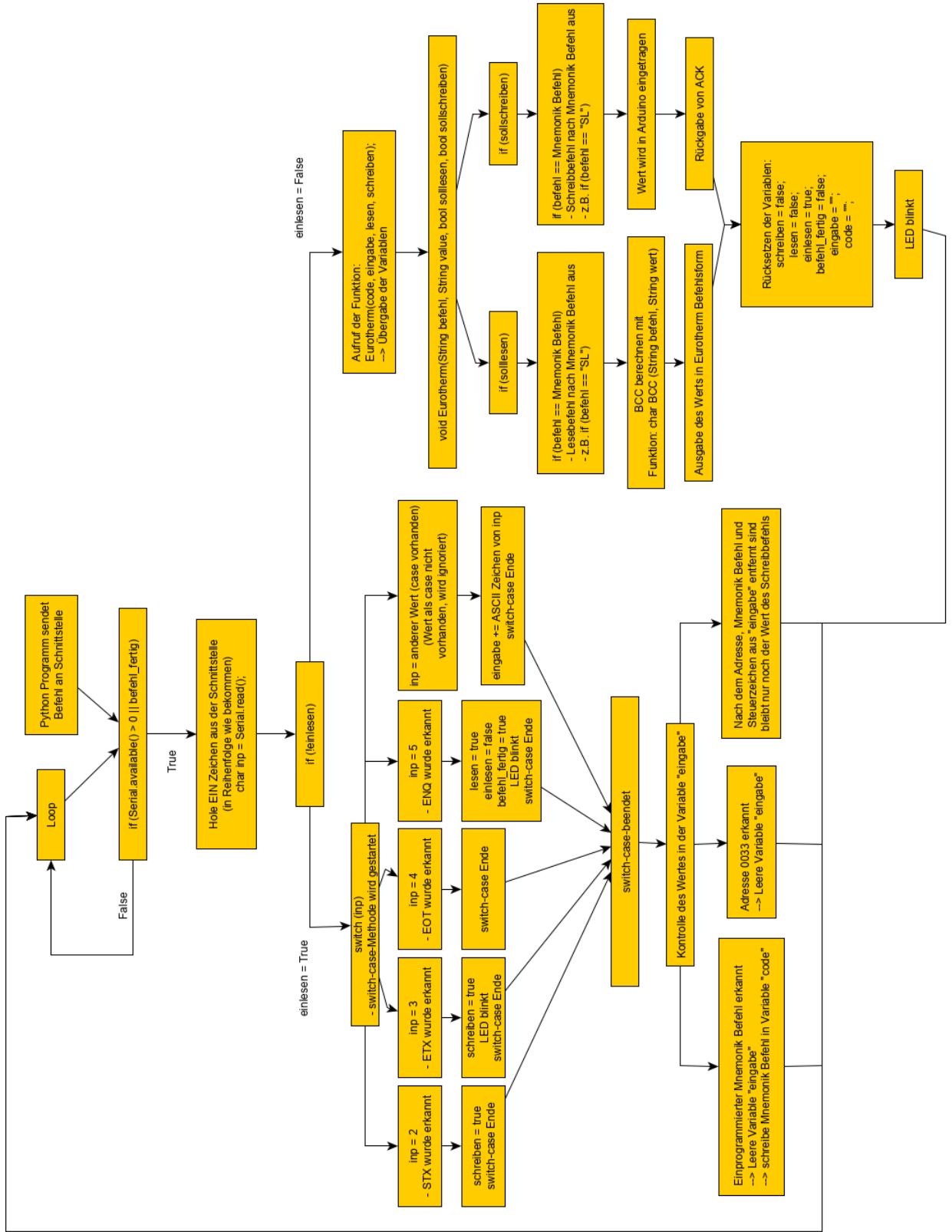


Abbildung 8: Ablaufschema des Arduino Programms (Quelle: eigene Darstellung)

Ein weiterer Unterschied zwischen Arduino und Eurotherm ist der PID-Regler. Bei Arduino stammt der PID-Regler aus einer eingebundenen Bibliothek und bei Eurotherm ist ein solcher Regler eingebaut. Der Unterschied der beiden Regler ist die Verarbeitung der Parameter. Eurotherm nutzt die Parameter TI, TD und kP (Mnemonik XP), während der PID-Regler auf kI, kD und kP zurückgreift. In Kapitel 2.3 kann man dazu mehr lesen. In Kapitel 2.3 wird der PID-Regler näher erläutert.

In der Parameterliste zu dem Programm *hautprogramm.py* muss man für die Umstellung von Eurotherm zu Arduino nur die Schnittstelle, das Delay und die PID-Parameter anpassen. Zudem muss man darauf achten, dass die im folgenden Kapitel genannte Variante in der Parameterliste nicht mehr aktiviert ist, weil sonst die Schnittstelle abstürzt durch Doppel-Nutzung. Was wie umgestellt werden muss wird im Kapitel 3.5 gezeigt. Für die Emulation wurden auch Hardware Erweiterungen vorgenommen, diese werden im folgenden Kapitel 3.4.2 näher erläutert.

### 3.4.2 Hardware der Emulation

Für die Variante 2 wurden noch zwei weitere Dinge mit eingebaut, welche aber auch in Variante 3 Anwendung finden. Diese Dinge sind eine LED und eine zweite RS232 Schnittstelle. Die LED blinkt je Befehl zweimal, einmal beim Finden des Abschlusszeichens des Befehls und das zweite Mal, nachdem die Ausgabe bearbeitet wurde. Die LED ist auch ein weiterer Grund für das Delay von 0,5 s, da diese 0,4 s je Befehl einnimmt. Das Delay von 0,5 s wurde ausgetestet und auch verringert, die Verringerung brachte aber wieder Probleme. Das Problem war eine falsche Rückgabe des Arduinos, in der Form einer vorherigen Antwort. Durch das Delay hat das Programm genug Zeit alles zu bearbeiten. Mit der LED wird zudem eine visuelle Ausgabe gegeben, die die Funktion der Emulation zeigt.

Die zweite Schnittstelle ist eine weitere Sicherheit für das Programm. In dieser Variante werden alle ankommenden Befehle und Zeichen in der zweiten Schnittstelle ausgegeben, sowie bestimmte Werte zum Leistungsausgang oder zu Sollwert-Änderungen. Diese Schnittstelle kann z.B. mit dem Programm Putty (Quelle [Put]) beobachtet werden. Eine solche Ausgabe kann man in Abbildung 9 sehen.

```

3
3
2
5
L
3
0
0
3
/
Befehl = SL

Vorher: 20
Nachher: 300
PIDOutputDACvolts      = 0,00
PIDOutputDACbits        = 0
PIDOutputDACvoltsCheck = 0,00
DDPIDLimitMaxDAC       = 1,80
4
0
0
3
3
E
E
5
-1
Befehl = EE

```

Abbildung 9: Ausgabe zweite Schnittstelle mit Putty (Quelle: eigene Darstellung)

Während der Bearbeitung der Emulation sind verschiedene Probleme aufgetreten. Bisher wurden bereits zwei Kontrollen kurz erwähnt. Mit der LED und der zweiten Schnittstelle wurden Fehler im Programm gesucht und behoben. Diese Lösungen sind in den Abbildungen 84 auf Seite XVIII

(Kondensatoren am Regel Pt100 am Arduino) und 86 auf Seite XX (Hardwaresicherheit für die Steuerspannung) als Schaltplan zu sehen.

Des Weiteren wird die Ausgangsleistung des Eurotherm-Reglers bzw. des Arduinos begrenzt. Diese Steuerspannung darf nur zwischen 0 und 10 V sein. Mit einem DAC-Chip (DAC = Digital Analog Umsetzer) wird die Steuerspannung erzeugt und mit einer zusätzlichen Hardwaresicherheit wird die Spannung begrenzt. Die Hardwaresicherheit wird durch einen Spannungsteiler und eine Diode realisiert. Sowohl der DAC-Chip als auch die Sicherheit sind in Abbildung 85 (S. XIX) zu sehen. In der Abbildung kann man auch sehen, dass ein Voltmeter genutzt wurde um die Steuerspannung zu beobachten. Ein Multimeter wurde für viele Zwischenmessungen genutzt, zum Beispiel zum Überprüfen der Widerstandsthermometer als Ohmmeter oder um auch die Spannung am Vorwiderstand der Kalibrierung (Kapitel 5.1) zu überwachen und zu kontrollieren.

Die Diode hat eine Schleusenspannung von 0,7 V, wodurch die Spannungsdifferenz zwischen der Eingangsspannung und der Ausgangsspannung 0,7 V betragen muss. In Abbildung 10 kann man die Schaltung des Spannungsteilers sehen.

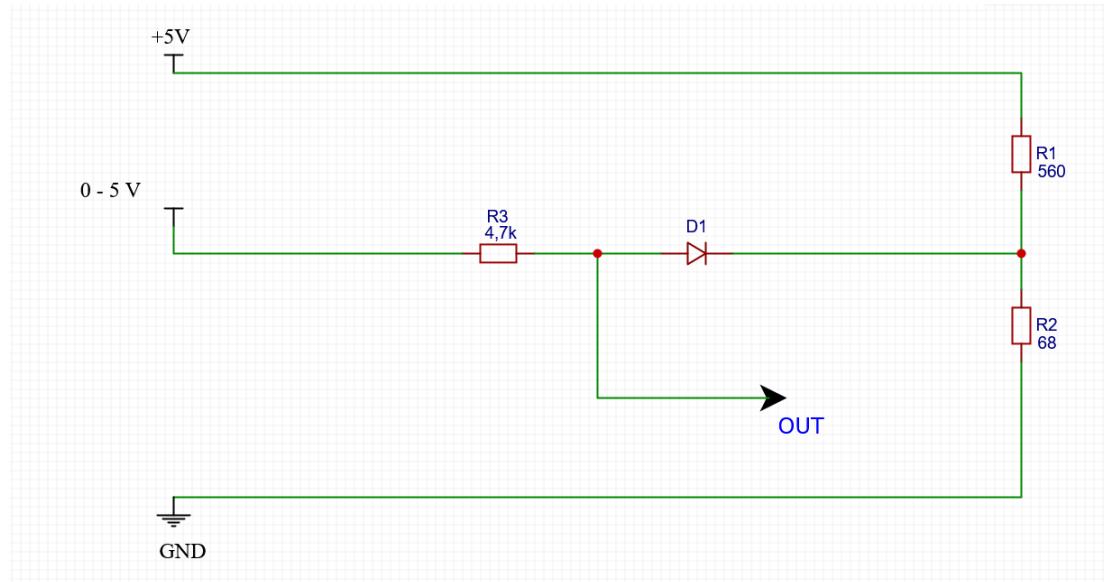


Abbildung 10: Hardwaresicherheit - Spannungsteiler und Diode (Quelle: eigene Darstellung)

Zunächst wurde die Spannung über dem Widerstand  $R_2$  für eine Ausgangsspannung von ca. 1 V bestimmt.

$$U_2 = U_{ges} * \frac{R_2}{R_2 * R_1} \quad (37)$$

$$U_2 = 5V * \frac{68\Omega}{68\Omega + 560\Omega} \quad (38)$$

$$U_2 = \underline{\underline{0,54V}} \quad (39)$$

Um am Ausgang 1 V maximal zuzulassen, braucht man 0,54 V am Teiler (Gleichungen 37 bis 39). Die Diode wird erst ab 0,7 V durchlässig, wenn also auf der Eingangsseite der Diode ca. 1,2 V liegen, wird die Diode wegen der Spannungsdifferenz ( $1,24\text{ V} - 0,54\text{ V} = 0,7\text{ V}$ ) durchlässig.

Für die Leistung, die von den Geräten ausgegeben wird, heißt dies, dass nicht mehr als 10 % möglich sind, weil die maximale Leistung 10 V Steuerspannung entspricht. Mit den 10 % konnten

Temperaturen von 300 bis 400 °C nicht schnell und völlig erreicht werden, deshalb muss man die Schaltung, also  $R_2$ , ändern um eine höhere Steuerspannung erhalten zu können. Am Ausgang sollen 2 V ankommen, was ca. 20 % an Leistung entspricht. Diese Maßnahme kommt daher, dass die Temperaturen mit einer Steuerspannung von 1 V nicht schnell erreichbar waren bzw. gar nicht erreichbar waren.

$$U_{Grenz} = 2V \quad (40)$$

$$\text{Mit Toleranz: } U_{Grenz} = 2,2V \quad (41)$$

$$U_2 = U_{Grenz} - U_S = 2,2V - 0,7V = \underline{\underline{1,5V}} \quad (42)$$

Die Toleranz aus Gleichung 41 kommt durch die Leistungseinheit, den Versorger des Grafit-Heizers. Sobald dieser mit der Gesamtschaltung verbunden war, ist die Steuerspannung um ca. 0,2 V abgesunken. Mit den 1,5 V aus Gleichung 42 kann man nun die Gleichung 37 nach  $R_2$  umstellen.

$$\frac{U_2}{U_{ges}} = \frac{R_2}{R_1 + R_2} \quad | * (R_1 + R_2) | * U_{ges} \quad (43)$$

$$U_2 * R_1 + U_2 * R_2 = U_{ges} * R_2 \quad | - (U_2 * R_2) \quad (44)$$

$$U_2 * R_1 = U_{ges} * R_2 - U_2 * R_2 \quad (45)$$

$$U_2 * R_1 = R_2 * (U_{ges} - U_2) \quad | : (U_{ges} - U_2) \quad (46)$$

$$R_2 = \frac{U_2 * R_1}{U_{ges} - U_2} \quad (47)$$

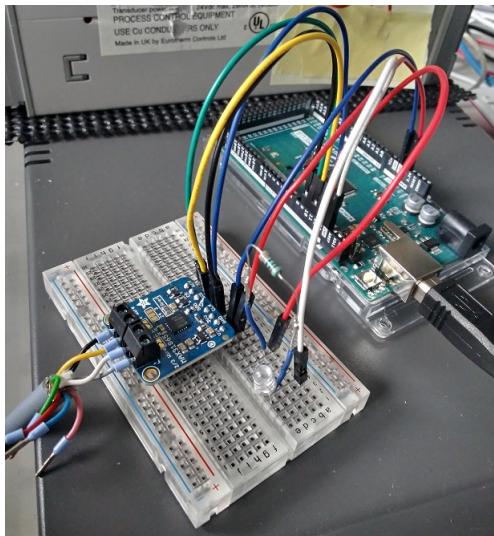
$$R_2 = \frac{1,5V * 560\Omega}{5V - 1,5V} = \underline{\underline{240\Omega}} \quad (48)$$

Um eine Steuerspannung von 2 V zu bekommen, muss man den  $68\Omega$  Widerstand durch einen  $240\Omega$  Widerstand ersetzen. Da kein  $240\Omega$  Widerstand zu finden war, wurde ein  $220\Omega$  Widerstand eingesetzt. Maximal können mit diesem Widerstand 2,1 V als Steuerspannung ausgegeben werden.

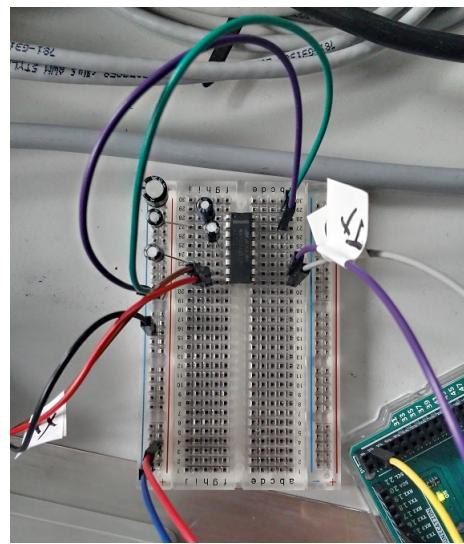
Die letzten Verbesserungen bzw. Fehlerbehebungen sind die gezeigten Kondensatoren in den Abbildungen 86 (S. XX) und 84 (S. XVIII). Diese Kondensatoren sind einmal Elektrolytkondensatoren und zum anderen Keramikkondensatoren. Sie wurden bei der zweiten Schnittstelle und bei dem Adafruit-Modul, das mit dem Arduino verbunden ist, eingebaut. Während der Versuche hat die Temperaturmessung Störungen gehabt, diese Störungen wurden mit den Kondensatoren ausgefiltert. Die Kondensatoren sollen Wechselspannungsanteile weg filtern.

In dem Kapitel 4 werden Adafruit-Module genutzt um Pt100 Widerstandsthermometer auslesen zu können. Für die Variante 2 wird der Pt100 der am Eurotherm angeschlossen ist nun mit dem Arduino verbunden. Dies kann man in Abbildung 11a sehen. Über einen Stecker ist das umstecken zwischen den beiden Varianten sehr leicht. Die Anschlüsse sind bei allen Adafruit-Modulen gleich.

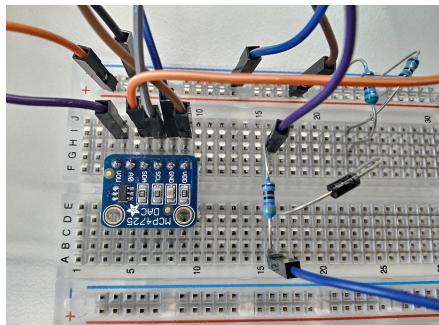
Als letztes werden in der Abbildung 11 die gerade erläuterten Schaltungselemente im Realen gezeigt.



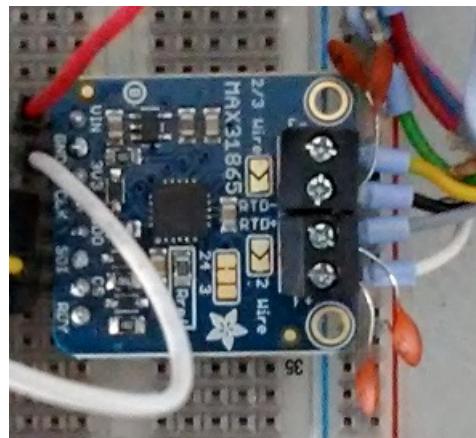
**a)** Arduino Adafruit-Modul Pt100 und LED mit  $330 \Omega$  Widerstand  
(Quelle: eigene Darstellung)



**b)** Schnittstelle 2 und Kondensatoren (Quelle: eigene Darstellung)



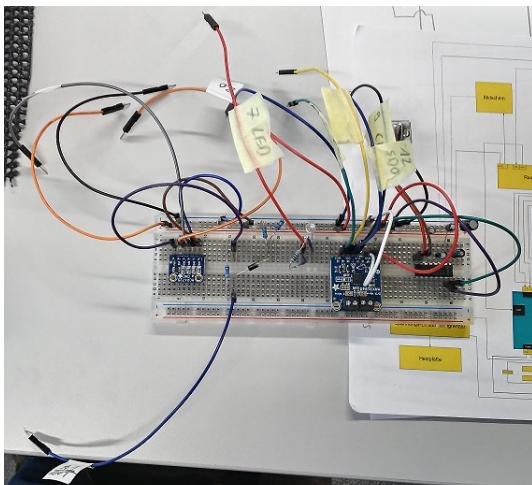
**c)** DAC-Chip und Spannungsteiler  
(Quelle: eigene Darstellung)



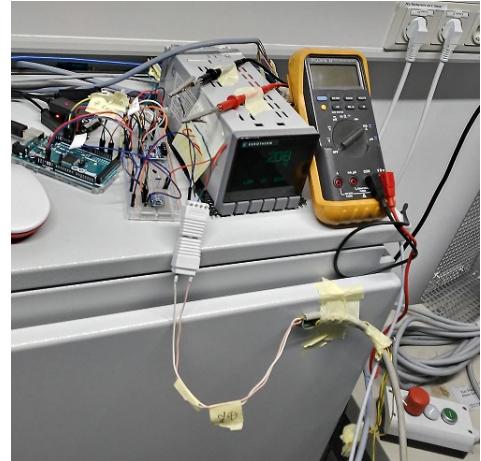
**d)** Kondensatoren am Pt100-Anschluss zum Adafruit-Modul  
(Quelle: eigene Darstellung)

Abbildung 11: Extra Elemente für Kontrolle und Sicherheit

In der Abbildung 12 kann man zum einem die Teile aus Abbildung 11 auf einem Breadboard sehen (Steckbrett) (Abbildung 12a) und zum anderen den Aufbau an der Test-CZ-Anlage (Abbildung 12b) sehen. Im der letzteren Abbildung kann man auch sehen, wie die Steuerspannung zum Versorger geht.



a) Alles auf einem Steckbrett (Quelle: eigene Darstellung)



b) Aufbau Anlage - Eurotherm, Arduino, Versorgung, Multimeter (Quelle: eigene Darstellung)

Abbildung 12: Gesamt Aufbau an der Test-CZ Anlage

Da diese Erweiterungen zur Fehlerbehandlung mit dem Arduino zu tun haben, wurde der Teil des Aufbaus in diesem Kapitel beschrieben und nicht in Kapitel 4.1. In dem Kapitel erfährt man dann mehr zum Gesamtaufbau und zum Heizer. Aufgrund der Probleme aus Kapitel 3.1 wurden nur die Varianten 2 und 3 für die Regelung verwendet.

Zum Schluss muss man sagen das die Emulation schlussendlich nur ein Ersatz für den Eurotherm-Regler ist und eine weitere Möglichkeit der Nutzung des Temperatur/Emissivitäts-Programmes ermöglicht. Im folgenden Kapitel 3.5 wird dann die dritte Variante der Regelung besprochen.

### 3.5 Arduino Leistungssteuerung und Eurotherm PID (Variante 3)

In dieser Variante wird der Eurotherm für die Regelung genutzt. Das Gerät nimmt die Temperaturdaten auf und bestimmt die Leistungsregelung. Diese Leistungswerte werden von dem Python-Programm dann an die Emulation des Eurotherm-Reglers mit Arduino gesendet. Durch den Arduino wird dann die Steuerspannung erzeugt und über die Hardwaresicherung wird eine Überschreitung der maximal möglichen Spannung verhindert. Die Berechnung so einer Spannung wird mit den Gleichungen 43 bis 48 gezeigt. Somit arbeiten beide Geräte zusammen und das Eurotherm Gerät kann genutzt werden.

Für diese Variante wurden weitere Zusätze in das Python-Programm *heizer.py* und das Arduino Programm *arduino\_test\_eurotherm\_testpid.ino.ino* eingebaut. In *heizer.py* wird über das Setzen der Variable *arduino\_in* auf TRUE unter Eurotherm in der Parameterliste die Variante freigeschaltet. Das Programm belegt nur bei TRUE die Schnittstelle des Arduinos und sendet die vom Eurotherm erhaltenen OP-Werte (Leistung in %) an den Arduino. Das Arduino-Programm hat dadurch einen extra Schreibbefehl erhalten, nämlich OP. Bei Eurotherm ist dieser Wert nur unter bestimmten Voraussetzungen schreibbar, dafür muss der Regler auf Manual stehen. Auch im Arduino-Programm muss man im Setup eine Variable ändern und zwar in der Zeile *myPID.SetMode(AUTOMATIC);<sup>6</sup>*. Wenn in dieser Zeile AUTOMATIC steht, wird der Pt100 über ein Adafruit-Modul vom Arduino ausgelesen und der PID-Regler aus der Arduino-Bibliothek regelt die Temperatur. Wenn man aber die Variante mit dem Eurotherm nutzen will, muss man

<sup>6</sup>Gehört zu dem Programm, das vom IKZ gestellt wurde.

in der Zeile das *AUTOMATIC* zu *MANUAL* ändern. Zusätzlich wird nicht nur dem Eurotherm die maximale Ausgangsleistung mit HO gesendet, sondern dem Arduino auch gleich.

```
def get_power_OUT(self):          # Leistung abfragen
    if test_on == False:
        befehl = 'OP'
        euroPow = self.read(befehl)
        if emu_on == True:           # Regelung PID, Leistungsregelung Arduino
            self.emu_write(befehl, euroPow)
        if log == True: logging.info('Leistungswert ausgelesen! - get_power_OUT()')
    else:
        euroPow = random.uniform(0,100)
    return float(euroPow)
```

Mit den Code-Zeilen aus *heizer.py* wird die Übertragung der Leistung an den Arduino gewährleistet. Für den Zweck hat die Klasse *HeizerEurotherm* noch die Funktion *emu\_write* bekommen. Da bei der Variante nur Werte in das Arduino-Programm eingesetzt werden, wurde die *send* Funktion etwas gekürzt und somit *emu\_write* erstellt.

### 3.6 PID-Regler - Arduino vs. Eurotherm

Beide Geräte (Eurotherm und Arduino) haben die Möglichkeit eines PID Reglers. Bei Eurotherm ist dieser im Inneren bereits programmiert und bei der Emulation mit Arduino gibt es eine spezielle PID-Bibliothek. Diese wird im Anhang (Kapitel 7.2 referenziert. Auch die hier erwähnten Programme und ihre Variablen sind dort zu finden.

Um die besten PID Parameter herauszufinden wurden diese mithilfe von Programmen bestimmt. Beide Programme ähneln sich sehr und wurden auf das benutzte Gerät angepasst. Die Programme heißen *Autotune\_Testprogram.py* (Eurotherm) und *Autotune\_Testprogram\_Arduino.py* (Arduino). Beim Eurotherm gibt es eine interne Funktion, die AutoTune oder Selbstoptimierung genannt wird. In der Quelle [Eura] kann man mehr zum AutoTune auf den Seiten 3-21 bis 3-32 lesen. Das Programm *Autotune\_Testprogram.py* nimmt die Werte auf und sendet die Leistungswerte des Eurotherms an den Arduino. Am Ende des Programmes werden die vom Eurotherm geänderten PID-Parameter ausgegeben.

Der Eurotherm stellt die Leistung auf Maximum und nach einer kleinen Zeit wieder auf das Minimum. Danach wird die Temperatur gemessen und nach einer Weile beendet sich der AutoTune von selbst. Dies sieht man an dem verschwindenden „AT“, auf dem Bildschirm.

Dieses Verhalten wurde mit dem zweiten Programm (*Autotune\_Testprogram\_Arduino.py*) für die Emulation nachgebildet. In dem Programm wird nach einer bestimmten Zeit der Leistungswert auf Maximum gesetzt und dann nach einer weiteren Zeit wieder auf Minimum. Danach wartet man, bis die Temperatur ein Plateau erreicht und wertet die Temperaturkurve aus. Diese Auswertung wird mit der Methode von Ziegler und Nichols durchgeführt, die in Kapitel 2.3 erläutert wird.

Wie bereits erwähnt kann der Eurotherm-Regler selbst eine Optimierung der Parameter durchführen. Die Ergebnisse zweier dieser AutoTunes werden hier gezeigt. Die Abbildung 13 zeigt den ersten AutoTune Versuch mit dem Eurotherm Regler. Diese Messung wurde an dem Test-Messaufbau mit der Kochplatte durchgeführt. Das Ergebnis des AutoTunes war  $k_p$  ist 5,8 groß,  $T_I$  ist 594,9 s groß und  $T_D$  ist 99,1 s groß. Während der arbeiten mit den Arduino wurde aufgrund der Temperaturschwankungen beschlossen den D-Anteil auf Null zusetzen, da die Schwankungen die Regelung durch den Anteil erschweren. Der AutoTune wurde im folgenden an der Test-CZ

Anlage durchgeführt. Das Ergebnis zeigt die Abbildung 14. Das Ergebnis des AutoTunes war  $k_p$  ist 3,4 groß und  $T_I$  ist 165,4 s groß.

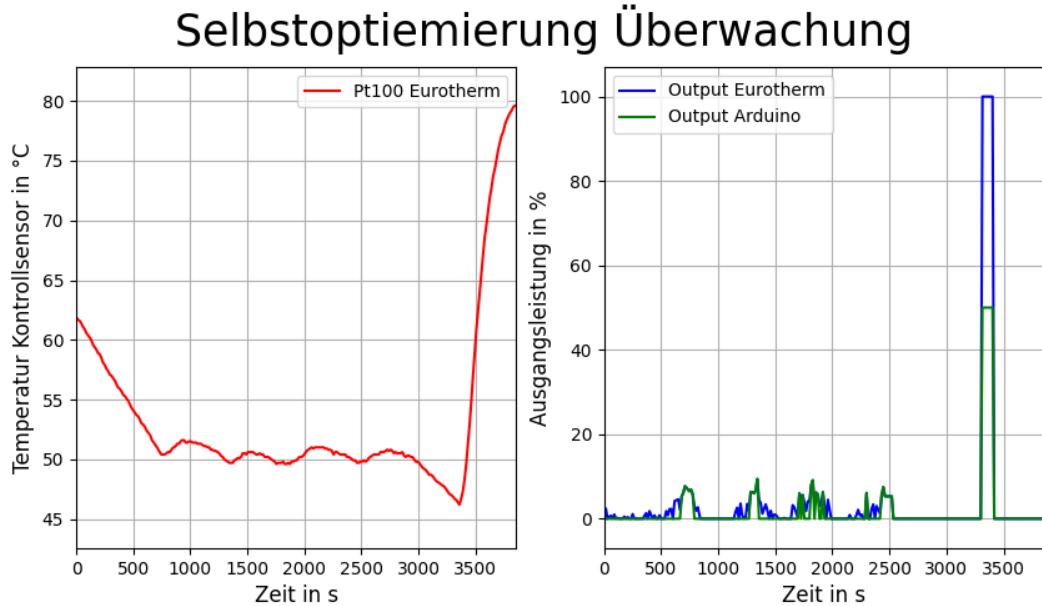


Abbildung 13: AutoTune Eurotherm an Testaufbau (Quelle: eigene Darstellung)

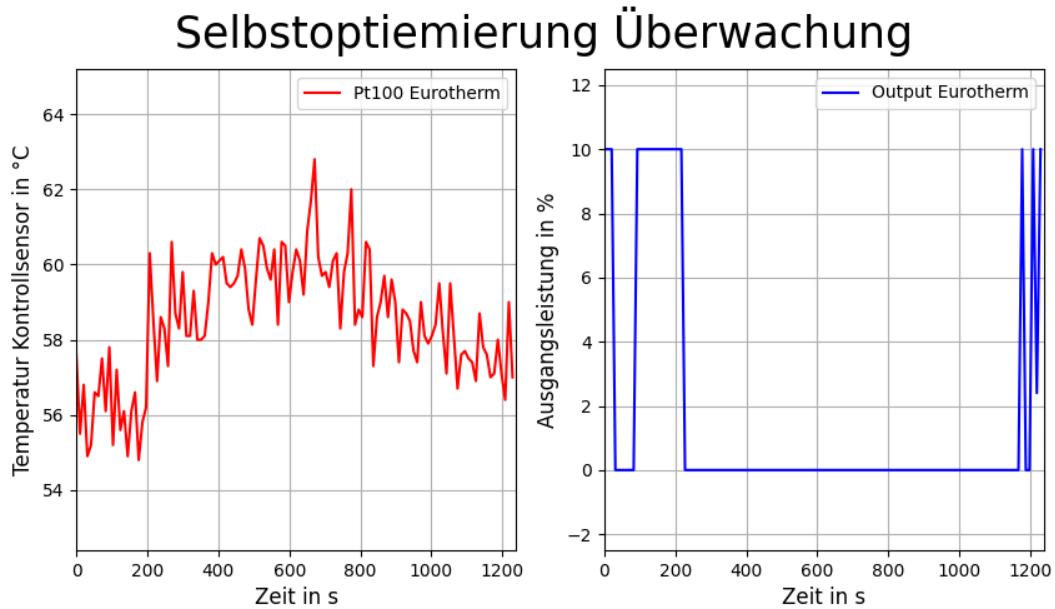


Abbildung 14: AutoTune Eurotherm an Test-CZ (Quelle: eigene Darstellung)

In den beiden Abbildungen kann man gut erkennen, das auch der letzte Eurotherm-Regler langsam defekt wird. In Abbildung 13 sind die Temperaturschwankungen noch nicht so stark wie in Abbildung 14. Der Eurotherm-Regler schwangte sich zu Beginn der Messungen noch auf eine Temperatur ein, doch mit der Zeit wurde dies immer schlimmer. Des weiteren wurde die Ausgabe des Leistungswertes des Arduinos in Abbildung 14 aufgrund von Fehlern unterdrückt. Zwischen der Aufnahme der beiden Diagramme wurden die Programme für die Emissionsgradmessung und die des Arduinos weiterentwickelt, wodurch die Diagramme leicht anders sind.

Für die Arduino Regelung muss die Optimierung der PID Parameter per Hand durchgeführt werden. Die Grundlagen zu der Methode von Ziegler und Nichols stehen in Kapitel 2.3 (S. 5). Im folgenden wird das Ergebnis gezeigt. Die Abbildung 15 zeigt zum einen die Messung des Programmes für die PI-Parameter Bestimmung und zum anderen den zeichnerischen Teil der Lösung.

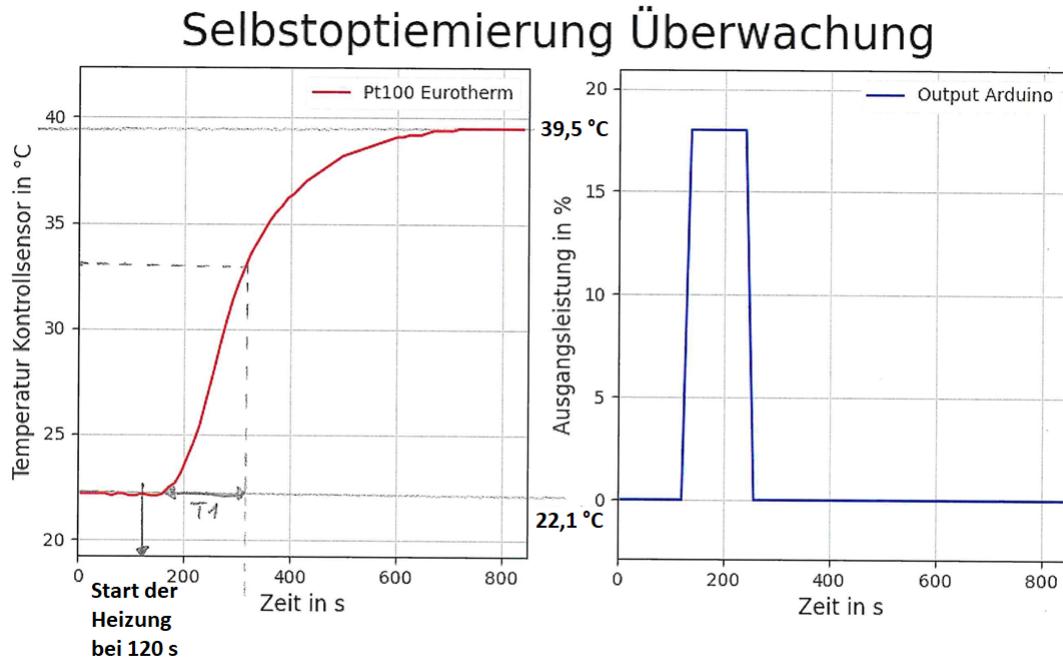


Abbildung 15: Zeichnerische Lösung Ziegler-Nichols-Methode (Quelle: eigene Darstellung)

Die Totzeit ( $T_t$ ) ist in diesem Fall 40 s groß. Diese Größe ist die Zeit nach dem Start der Messung, also wenn die Leistung auf 18 % gesetzt wurde, bis zur Reaktion der Kurve. Den Endwert und Startwert der Messung (Temperatur) kann man in der Abbildung 15 sehen.

Nebenrechnung:

$$0,63 * (\vartheta_{Endwert} - \vartheta_{Startwert}) + \vartheta_{Startwert} = 0,63 * (39,5^\circ C - 22,1^\circ C) + 22,1^\circ C = 33,062^\circ C \quad (49)$$

$$x = \frac{2,2cm * 3,062C}{5C} = 1,347 cm \quad (50)$$

$$T = \frac{1,7 cm * 400 s}{4,4 cm} = 154,6 s \quad (51)$$

Über die Gleichungen 49 und 50 wird die Zeit T zeichnerisch bestimmt. Für die Bestimmung dieser Zeit muss man 63 % des Endwertes berechnen. Die Verschiebung von 22,1 °C muss somit auch beachtet werden. Bei Gleichung 50 und 51 handelt sich über Verhältnisgleichungen. Mit der ersten Verhältnisgleichung wird der Abstand in dem Kästchen 30 bis 35 °C für die 63 % in der Abbildung berechnet. Mit der zweiten Verhältnisgleichung wird dann die aus der Abbildung gemessen Länge für die Zeit in eine Zeit umgerechnet.

$$k_S = \frac{\text{Endwert}}{u} = \frac{39,5}{18} = 2,194 \quad (52)$$

$$k_P = 0,9 * \frac{154,6s}{2,194 * 40s} = \underline{\underline{1,59}} \quad (53)$$

$$T_I = 3,33 * T_t = 3,33 * 40s = \underline{\underline{133,2s}} \quad (54)$$

$$k_I = \frac{1,59}{133,2} \approx \underline{\underline{0,012}} \quad (55)$$

Die Formeln stehen in der Tabelle 1 auf Seite 6. Die 18 in Gleichung 52 ist das Ausgangssignal des Reglers, was mit u gekennzeichnet wird. Aus dem Endwert und dem Reglerausgang wurde dann die statische Verstärkung ( $k_S$ ) berechnet. Der Endwert ist der Wert wo das Plateau der Temperatur erreicht ist.

In Kapitel 2.3 werden mit Gleichung 8 zwei verschiedene Übertragungsgleichungen für den PID-Regler gezeigt. Zudem wird in dem Kapitel gezeigt wie man die Verstärkungsparameter in die Zeiten umrechnet. Bei dem Eurotherm-Regler werden für D- und I-Anteil die Zeiten übergeben und bei dem Arduino werden die Verstärkungsparameter für diese Anteile übergeben.

In dem Kapitel 4.3 werden die Ergebnisse der PI-Regler gezeigt. Um hier die Regler von Arduino und Eurotherm zu vergleichen werden hier die Ergebnisse aus den Heiztesten gezeigt.

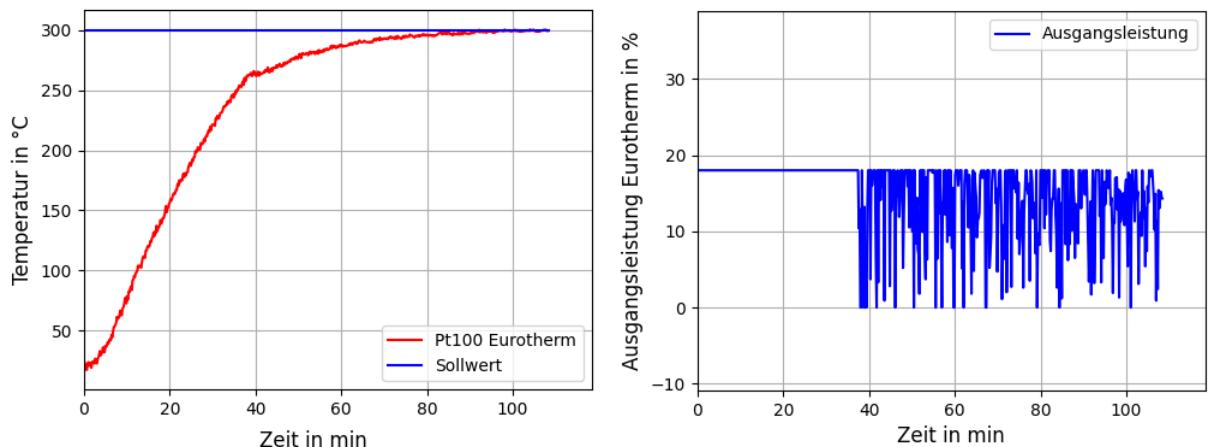


Abbildung 16: Eurotherm Regelung (Quelle: eigene Darstellung)

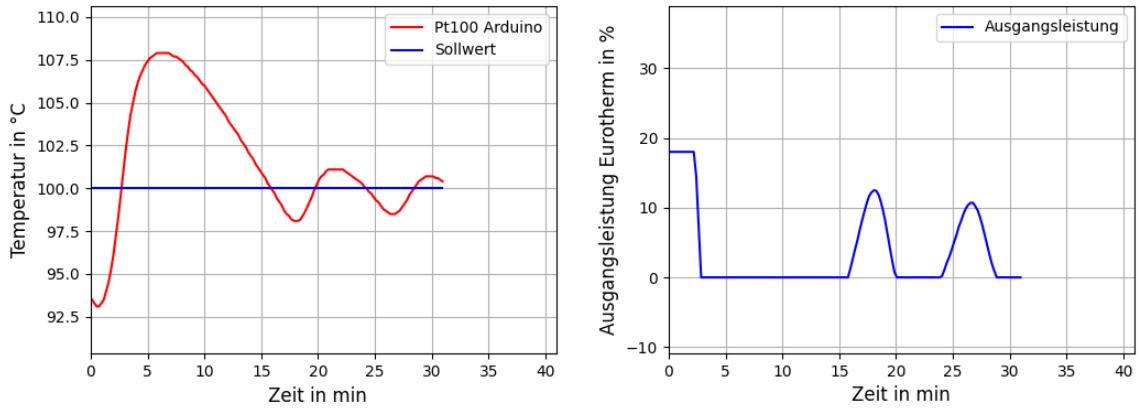


Abbildung 17: Arduino Regelung (Quelle: eigene Darstellung)

Bei Abbildung 16 wurden noch die alten PID-Parameter genutzt, die mit Abbildung 13 bestimmt wurden. Die PI-Parameter für die Abbildung 17 waren für  $k_P = 5,8$  groß und für  $k_I = 0,01$  groß. Die Parameter für Abbildung 17 sind aus den Parametern von Abbildung 16 bestimmt worden. Die Formel für die Umrechnung kann man in Gleichung 9 (S. 6) sehen.

$$k_I = \frac{k_P}{T_I} = \frac{5,8}{594,9} \approx 0,01 \quad (56)$$

Auch wenn die Leistungsregelung des Eurotherms sehr schlecht ist, da diese so verrauscht ist, passt sich der Temperaturwert sehr gut an den Sollwert an. In der Kurve kann man zudem sehen das die Regelung bereits vor dem Sollwert anfängt zu regeln, wodurch die Kurve so abklappt. Bei dem Arduino ist die Leistungsregelung nicht so verrauscht, dafür schwingt der Istwert der Temperatur deutlich mehr um den Temperatursollwert. Der Eurotherm zeigt in Abbildung 16 schon eine starke Schwankung der Temperaturkurve. Die Leistungsregelung schwankt wahrscheinlich durch den D-Anteil so stark, deswegen wurde dieses auch später auf Null gesetzt. Bei dem Arduino müsste man die Parameter noch weiter verbessern um eine bessere Kurve zu bekommen. Zusammenfassend kann man sagen das beide Regler mit den bisherigen Einstellungen Vorteile und Nachteile hatten. Wenn man die Parameter noch weiter anpasst kann man diese Regelung noch verbessern.

### 3.7 Testaufbau mit Kochplatte

In Kapitel 4 wird der Aufbau des Experimentes für die in diesen Kapiteln beschriebene Temperaturregelung erläutert. Bevor die Regelung mit den aus Kapitel 3.4.2 beschriebenen Hardwarezusätzen an die Test-CZ Anlage umzog wurde alles mit einer Kochplatte ausgetestet. Die Kochplatte ist in der Abbildung 18 zusehen.

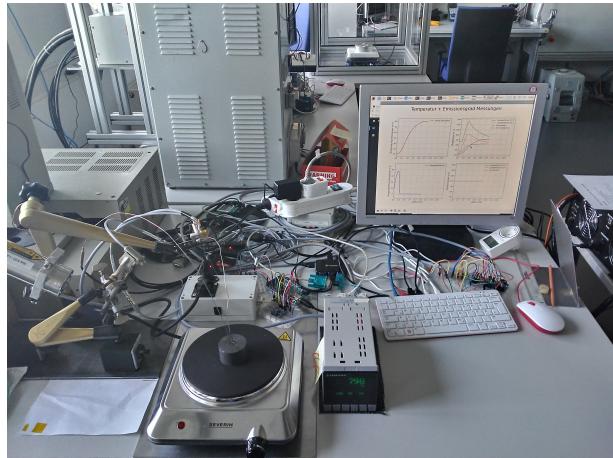


Abbildung 18: Programm Test-Aufbau mit Kochplatte (Quelle: eigene Darstellung)

Um diese Kochplatte richtig zu steuern, wurde ein Solid-State-Relais genutzt. Die Kochplatte ist in Abbildung 19 zu sehen. Des Weiteren wird in Abbildung 84 (S. XVIII) die Kochplatte und das Solid-State-Relais kurz dargestellt.

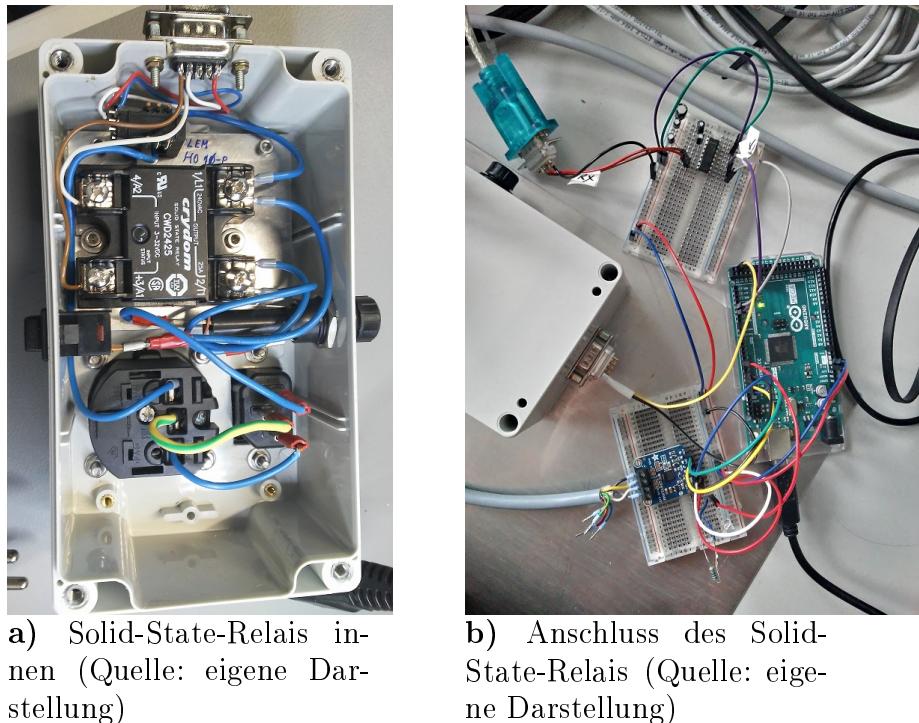


Abbildung 19: Solid-State-Relais für Kochplatte

Auch der erste AutoTune des Eurotherm-Reglers aus Abbildung 13 wurde an diesem Aufbau durchgeführt.

## 4 Messung des Emissionsgrades

In diesem Kapitel wird nun der Aufbau für die Emissionsgrad-Messung und der im Kapitel 3 gezeigten Temperaturregelung erläutert. Des Weiteren wird auch das für die Messung genutzte Python-Programm dargestellt.

### 4.1 Messaufbau

Für die Messung des Emissionsgrades wird ein Grafit-Heizer genutzt. Die Grundlagen dieses Heizers wurden in Kapitel 2.1 erklärt. So wurden in dem Kapitel die Berechnungsformeln für den Widerstand dieses Heizers erläutert. Zunächst wird nun die Berechnung des in der Test-CZ-Anlage genutzten Grafit-Heizers gezeigt. Mit der Abbildung 20 ist eine Skizze mit Maßen für den Grafit zu sehen.<sup>7</sup>

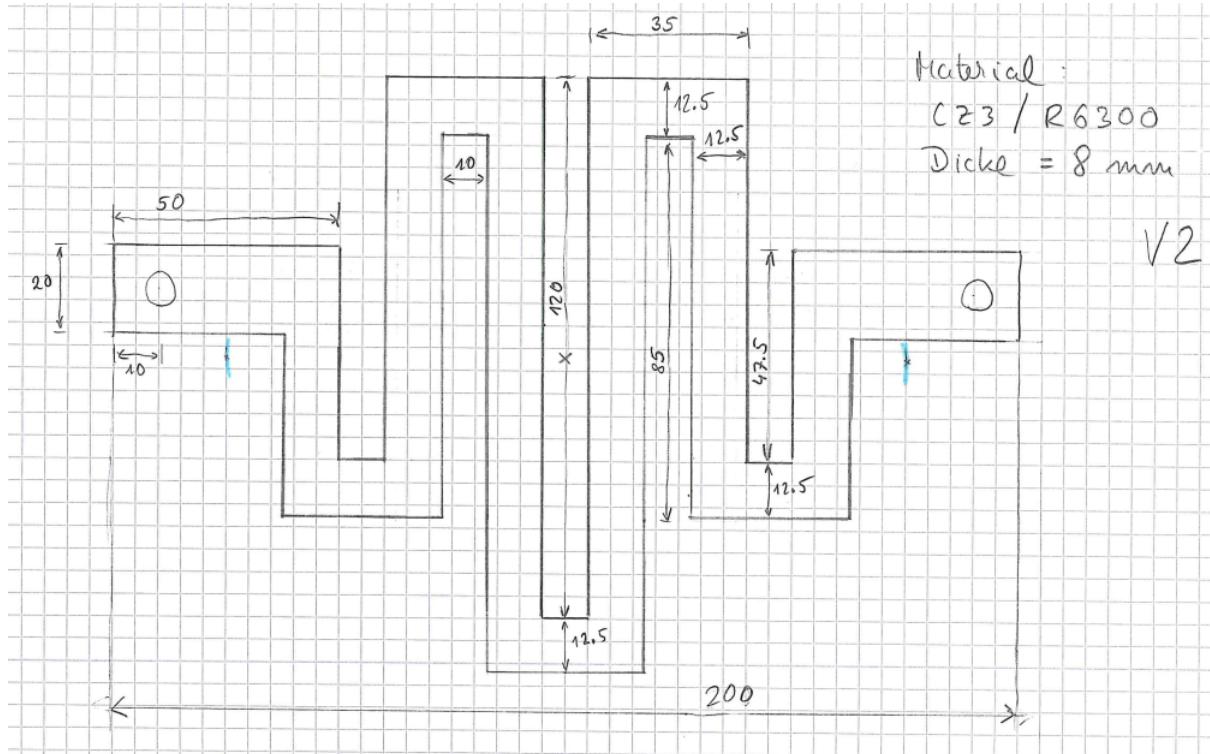


Abbildung 20: Skizze des Grafit-Heizers (Quelle: eigene Darstellung)

Für die Länge des Leiters wurde die Mittellinie von Anschluss rechts bis Anschluss links (beide Male Mitte Kreis in Skizze) genommen. Da der Heizer symmetrisch ist, sieht die Berechnung der Länge wie folgt aus. Die Berechnung der Länge läuft von links nach rechts.

$$l_L = ((50 - 10 - \frac{12,5}{2})mm + 47,5mm + (35 - 12,5)mm + 85mm + (35 - 12,5)mm + 120mm + (\frac{35}{2} - \frac{12,5}{2})mm) * 2 = \underline{\underline{685mm}} \quad (57)$$

$$A_L = 8mm * 12,5mm = \underline{\underline{100mm^2}} \quad (58)$$

<sup>7</sup>Das V2 in der Abbildung 20 steht für Version 2. Während der Heiztests ist der erste Grafit-Heizer zerstört worden.

Mit den Werten aus den Gleichungen 57 und 58 hat man die ersten Werte für die Gleichung 1 (S. 1) berechnet. Nun fehlt nur noch der spezifische Widerstand, der sich aus der elektrischen Leitfähigkeit berechnen lässt. Aus dem Datenblatt (Quelle [Gra]) für das Material CZ3/R6300 (Grafit) kann man den spezifischen Widerstand  $\rho$  direkt ablesen.

$$\rho = 17 \mu\Omega m = 17 \frac{\Omega mm^2}{m} \quad (59)$$

Nun kann man mit den Werten den Widerstand des Heizers über die Gleichung 1 berechnen und erhält somit einen elektrischen Widerstand des Heizers von  $116 \text{ m}\Omega$ .

$$R_L = \frac{17 \frac{\Omega mm^2}{m} * 0,685m}{100mm^2} \approx \underline{\underline{0,116\Omega}} \quad (60)$$

Für die Messung des Emissionsgrades wird ein Material, z.B. Grafit, in einen Behälter aus Aluminium (siehe Abbildung 21) gesetzt. Dieser Aluminium-Körper dient dem gleichmäßigen Erwärmen der Probe im Inneren. Über einen Aluminium-Deckel wird die Probe festgehalten und durch ein Loch können die Pyrometer weiterhin auf die Probe sehen. Der Aluminium-Körper wird durch einen Grafit-Heizer erwärmt. Dieser Heizer, in Abbildung 22 zu sehen, ist mäanderförmig und auf die Größe des Aluminium-Körpers angepasst. Über Halterungen aus Aluminium-Oxid-Keramik, welche auch in Abbildung 22 zu sehen sind, wird der Kontakt zwischen Heizer und Aluminiumkörper verhindert. Die Daten zu dem genutzten Grafit-Heizer und Grundlagen sind in Kapitel 2.1 (S. 3) beschrieben. Des Weiteren stehen der Aluminium-Körper und der Heizer in einer Grafit-Filz-Isolierung (siehe Abbildung 23b).



Abbildung 21: Aluminium-Körper mit Grafit-Probe (Quelle: [Fun21a] S. 3)

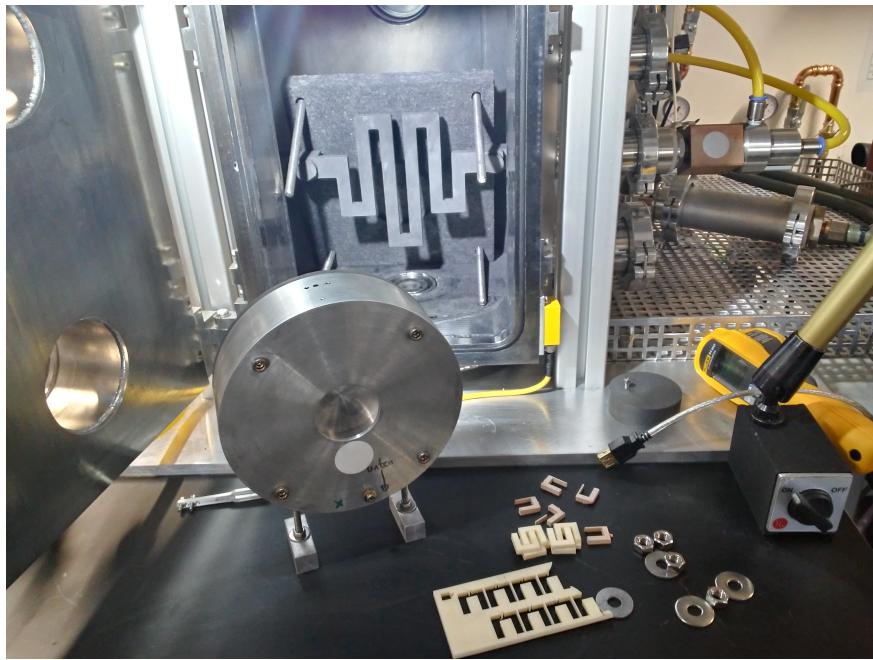


Abbildung 22: Grafit-Heizer und stehender Aluminium-Körper (Quelle: eigene Darstellung)

Der Aufbau ist eine Verbesserung des Aufbaus des abgelegten Fachpraktikums am IKZ. Im Fachpraktikum wurde der Aluminium-Körper mit dem Loch nach oben auf einer Heizplatte des Herstellers IKA verwendet. Bei dem Aufbau gab es Probleme mit den Pyrometern. Durch die aufsteigende Wärme wurden die Messungen verfälscht und die Pyrometer mussten in einem größeren Abstand platziert werden, wodurch die Messflecke sich vergrößerten.

Im Aufbau in der Test-CZ-Anlage steht der Aluminium-Körper aufrecht, also mit dem Loch nach vorne. Durch diese Änderung können die Pyrometer näher an die Probe. Dieser Umstand ist besonders für die langwelligen Pyrometer wichtig. Diese haben einen festen Fokus, wodurch sich mit größerem Abstand der Messfleck auch vergrößert. Bei dem genutzten kurzweligen Pyrometer kann man den Fokus und somit den Messfleck einstellen. Zudem kann man durch den Laser dieses Pyrometers die Größe des Messfleckes sehen. Beide Aufbauten sind in der Abbildung 23 zu sehen. In Abbildung 23a wird der alte Aufbau gezeigt und in Abbildung 23b der neue Aufbau.

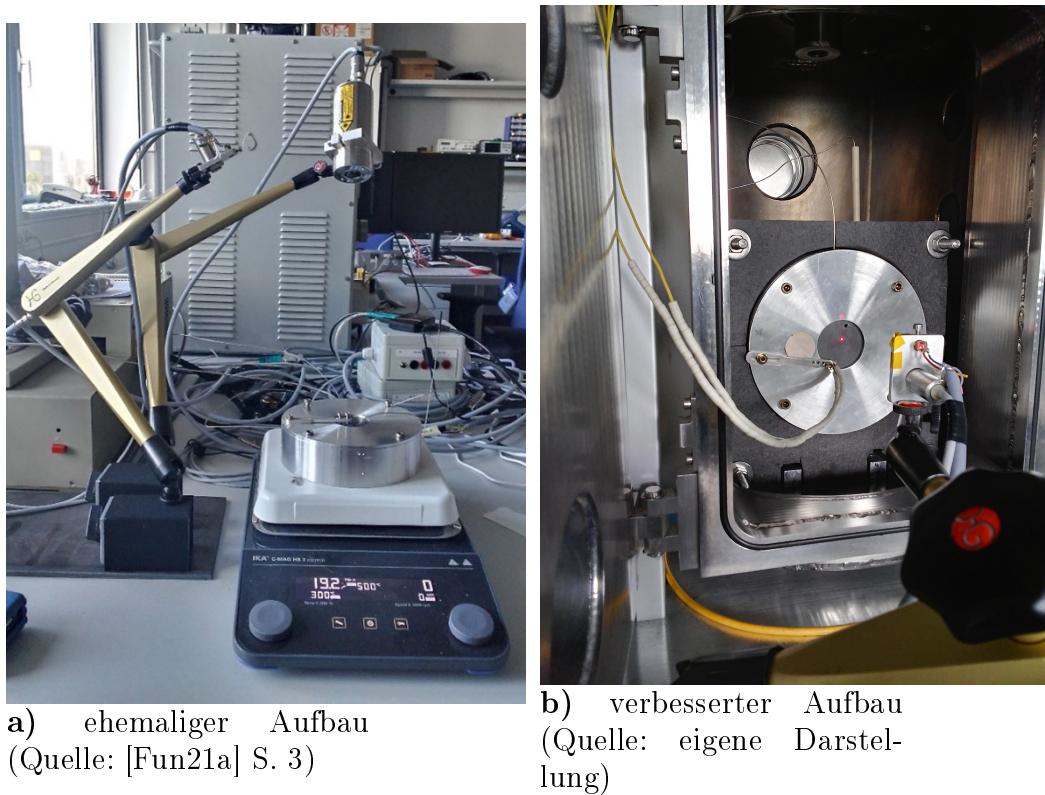


Abbildung 23: Praktikums- und Neuer Aufbau

**Folgende Messgeräte sind im Einsatz in dem Aufbau der Test-CZ-Anlage:**

1. Temperatur-Regler-Möglichkeiten (Kapitel 3): Eurotherm 905S (siehe Abbildung 24a), Arduino Mega - Eurotherm-Emulation (Kapitel 3.4) oder Eurotherm und Arduino zusammen (Kapitel 3.5)
2. Pyrometer
  - Kurzwelliges Pyrometer - IGA 6/23 – Laser von der Firma LumaSense (siehe Abbildung 24c)
  - Langwelliges Pyrometer - Impac® Series 600 von Advanced Energy (siehe Abbildung 24d)
    - Aspektverhältnis von 20:1 - im Einsatz in dem Aufbau
    - Aspektverhältnis von 10:1 - nur in den Testläufen - Kochplatte genutzt
3. Widerstandsthermometer Pt100 - alle Pt100 (außer am Eurotherm) werden über ein Adafruit-Modul (siehe Abbildung 24b) ausgelesen
  - Regelsensor am Eurotherm/Arduino - im Aluminium-Körper
  - Vergleichssensor für die Emissionsgradbestimmung - auf der Oberfläche der Messprobe
4. Thermoelement Typ-K - Temperaturkontrolle Grafit-Heizer - keine Programm-Überwachung



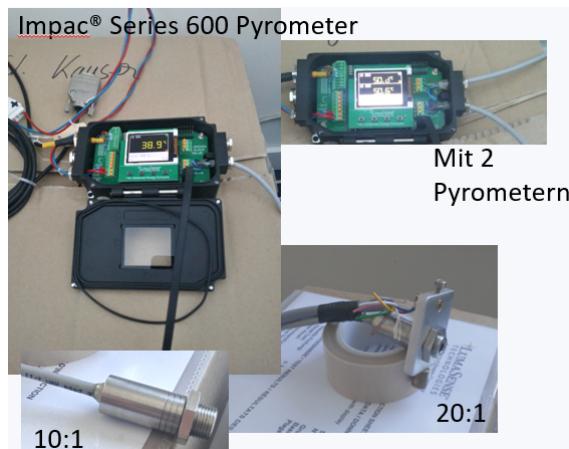
a) Eurotherm 905S



b) Adafruit-Modul MAX31865



c) Pyrometer KW



d) zwei Pyrometer LW

Abbildung 24: Genutzte Messgeräte (Quelle: eigene Darstellung)

Der Anschluss der Geräte am Computer erfolgt über einen Raspberry Pi 400 und einem USB-Hub. Diese beiden Geräte sind in Abbildung 25 zu sehen.

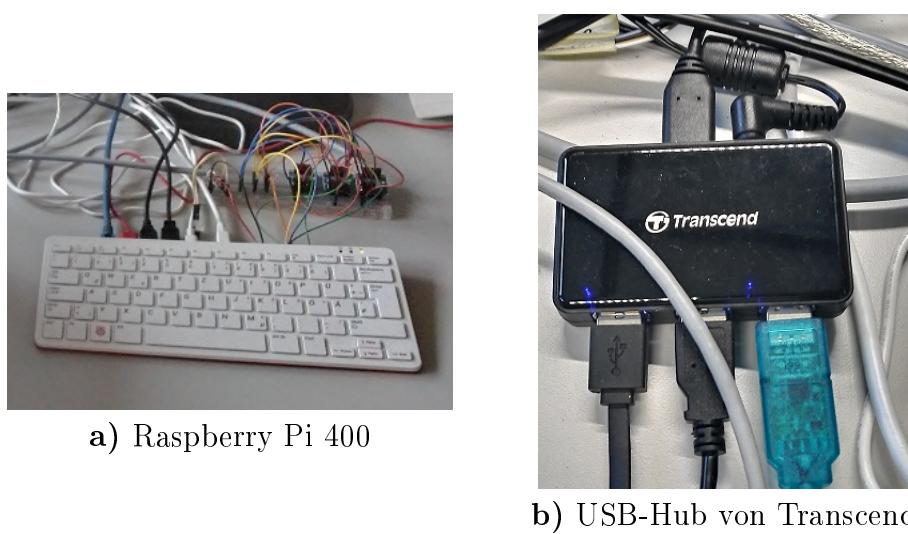


Abbildung 25: Anschluss Geräte (Quelle: eigene Darstellung)

Nachdem man die Hauptmessgeräte kennengelernt hat, werden nun die Widerstandsthermometer etwas näher erläutert. In Abbildung 26 (S. 40) kann man diese beiden genutzten Pt100-Sensoren sehen. Der eine wird auf die Grafit-Oberfläche gepresst und der andere steckt im Aluminium-Körper drinnen und reicht bis zum Grafit-Körper. Der Aluminium-Körper verfügt über mehr als eine Bohrung. Bei der Konstruktion des Aluminium-Körpers wurden 4 Bohrungen in den Körper gebohrt, zwei durchgehende und zwei nicht durchgehende Bohrungen. Die Grafit-Probe verfügt auch über eine Bohrung in der Höhe der durchgehenden Bohrungen des Aluminium-Körpers. Durch diesen Umstand kommt man mit dem Regelsensor bis in die Mitte des runden Körpers. Wie schon erwähnt ist bei der Messung der Regelsensor nicht im Grafit. Der wichtigere der beiden Pt100-Sensoren ist der Oberflächensor. Des Weiteren kann man in Abbildung 27 einen Pt100-Sensor sehen, der dann über einen Stecker an ein Adafruit-Modul angeschlossen werden kann. So ein Sensor wird als Regelsensor genutzt.



Abbildung 26: Genutzte Sensoren im/am Aluminium-Körper (Quelle: eigene Darstellung)

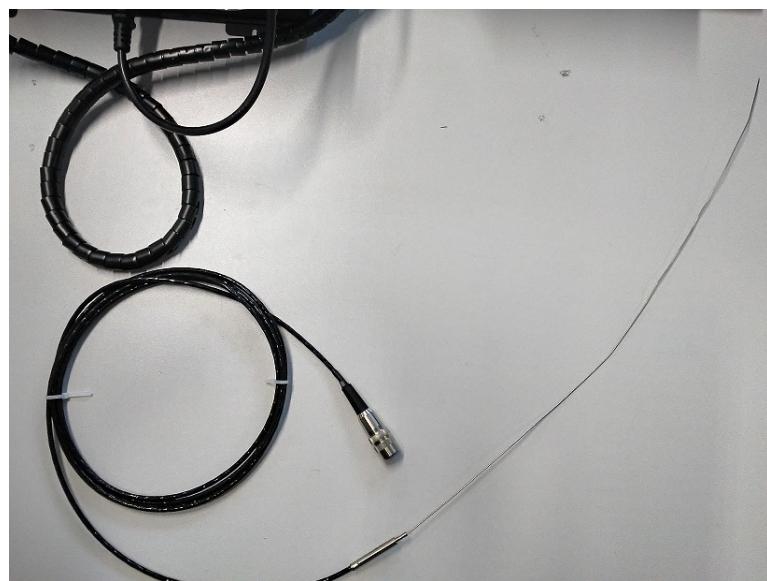


Abbildung 27: Genutzter Pt100 (Quelle: eigene Darstellung)

In den Abbildungen 28 (S. 41) und 29 (S. 41) kann man den ehemaligen Aufbau und den verbesserten Aufbau für die Oberflächensensoren sehen. Im alten Aufbau wurde der runde Sensor einfach unter die Halterung (Abbildung 28a) geklemmt. Da dies aber beim Anpressen immer verrutscht ist, wurde zum besseren Anpressen Grafit-Filz (Abbildung 28b) genutzt. Diese Konstruktion hatte auch durch den Platzverlust Probleme mit dem Messfleck des langwelligen Pyrometers verursacht.

In dem neuen, verbesserten Aufbau werden die runden Pt100 nur noch in die Löcher des Aluminium-Körpers eingesteckt. In Abbildung 29b kann man den neuen Pt100-Sensor sehen. Der Sensor ist sehr klein und flach anstatt rund. Der Sensor besitzt eigentlich einen Zweileiter-Anschluss, welcher dann für die genutzten Adafruit-Module zu einem Vierleiter-Anschluss umgewandelt wurde. Wie im alten Aufbau wird der neue Oberflächensensor auch an die Oberfläche angepresst. Diesen Umstand kann man in Abbildung 29b sehen.

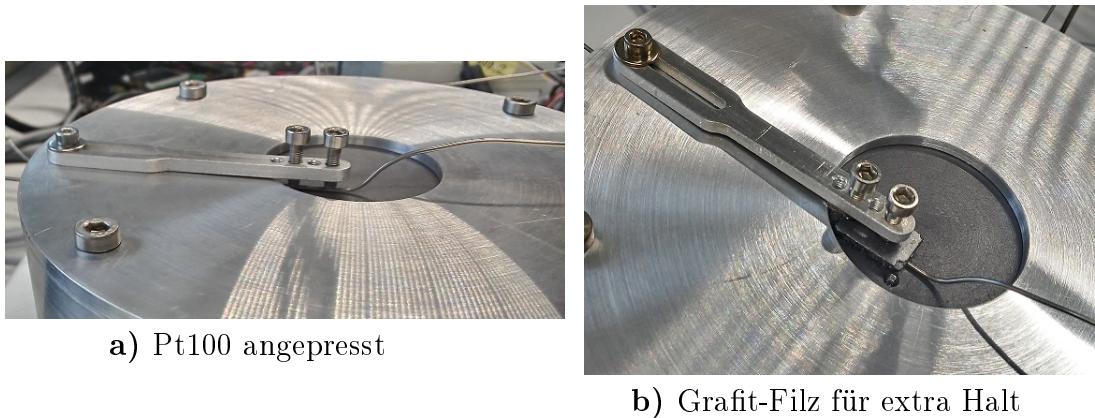


Abbildung 28: Oberflächensensor ehemaliger Aufbau (Quelle: eigene Darstellung)

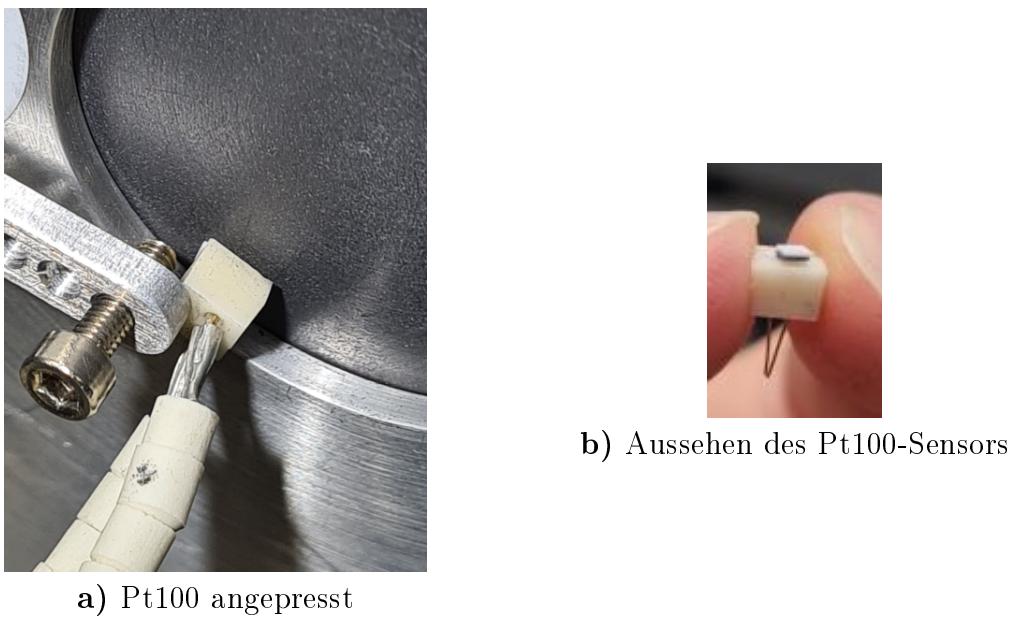


Abbildung 29: Oberflächensensor verbesserter Aufbau (Quelle: eigene Darstellung)

Ein weiterer wichtiger Fakt für das Experiment ist die Ausrichtung der Pyrometer, die im Folgenden gezeigt wird. Diese Darstellung folgt aus Skizzen und kann in den Abbildungen 30 und 31 angesehen werden. Wie der Messfleck der beiden Pyrometer berechnet wird bzw. wie er bei welchem Abstand aussieht kann für das KW Pyrometer in der Quelle [Pyra] auf Seite 21 (MB

13) nachgelesen werden und für das LW Pyrometer in der Quelle [Pyrb] auf den Seiten 5-3 bis 5-5 (genutztes Pyrometer 20:1) gefunden werden.

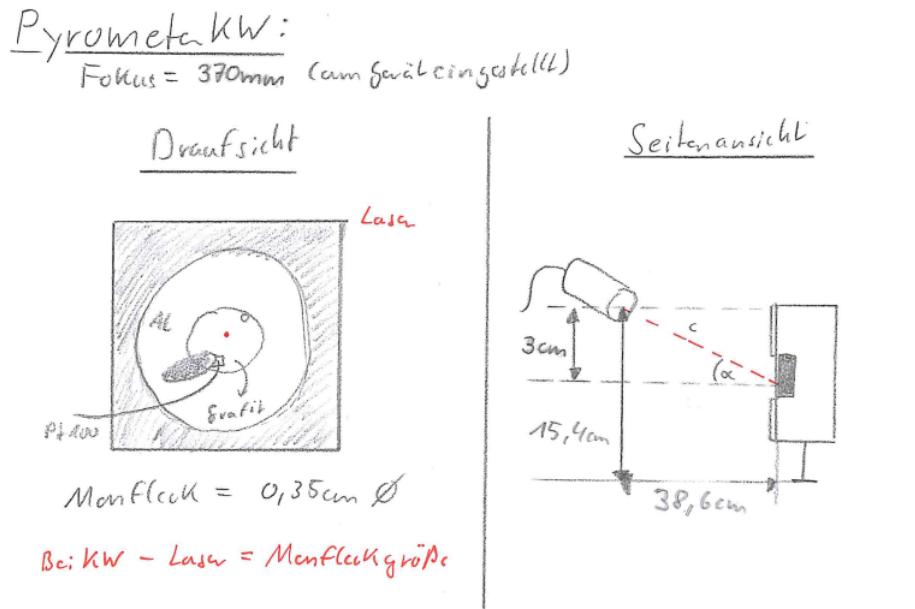


Abbildung 30: Ausrichtung des Kurzwelligen Pyrometers (IGA 6/23) (Quelle: eigene Darstellung)

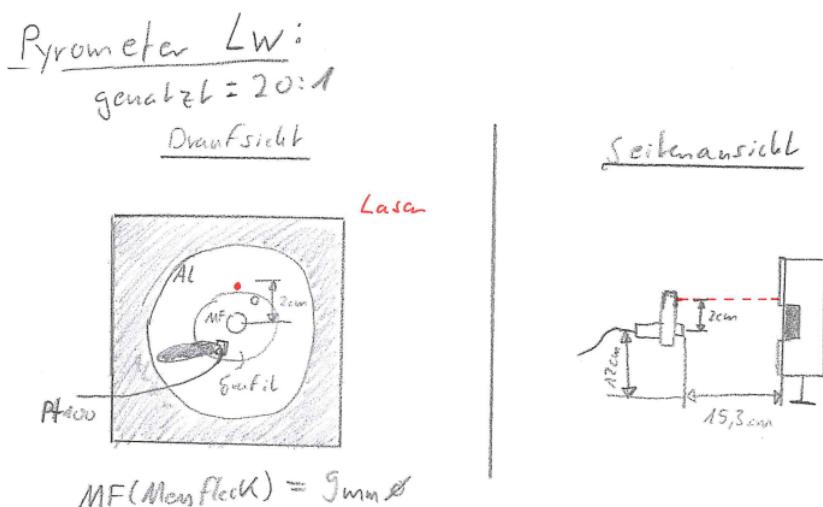


Abbildung 31: Ausrichtung des Langwelligen Pyrometers (Impac Series 600 - AV 20:1) (Quelle: eigene Darstellung)

Aus den Maßen von Abbildung 30 kann man für das KW Pyrometer den Ausrichtungswinkel bestimmen. Das LW Pyrometer wurde senkrecht, auf die Probe blickend, ausgerichtet. Der eingestellte Fokus des KW Pyrometers ist 370 mm. Dies passt ungefähr mit dem berechneten Abstand aus der Abbildung 30, Gleichung 62, überein (Hypotenuse (c) des rechtwinkligen Dreiecks). Der Winkel wird mit Gleichung 61 berechnet.

$$\alpha = \arctan\left(\frac{3\text{cm}}{38,6\text{cm}}\right) = 4,44^\circ \quad (61)$$

$$c = \sqrt{(3\text{cm})^2 + (38,6\text{cm})^2} = 38,72\text{cm} \quad (62)$$

In der Abbildung 32 kann man die reale Anlage mit den ausgerichteten Pyrometern sehen.

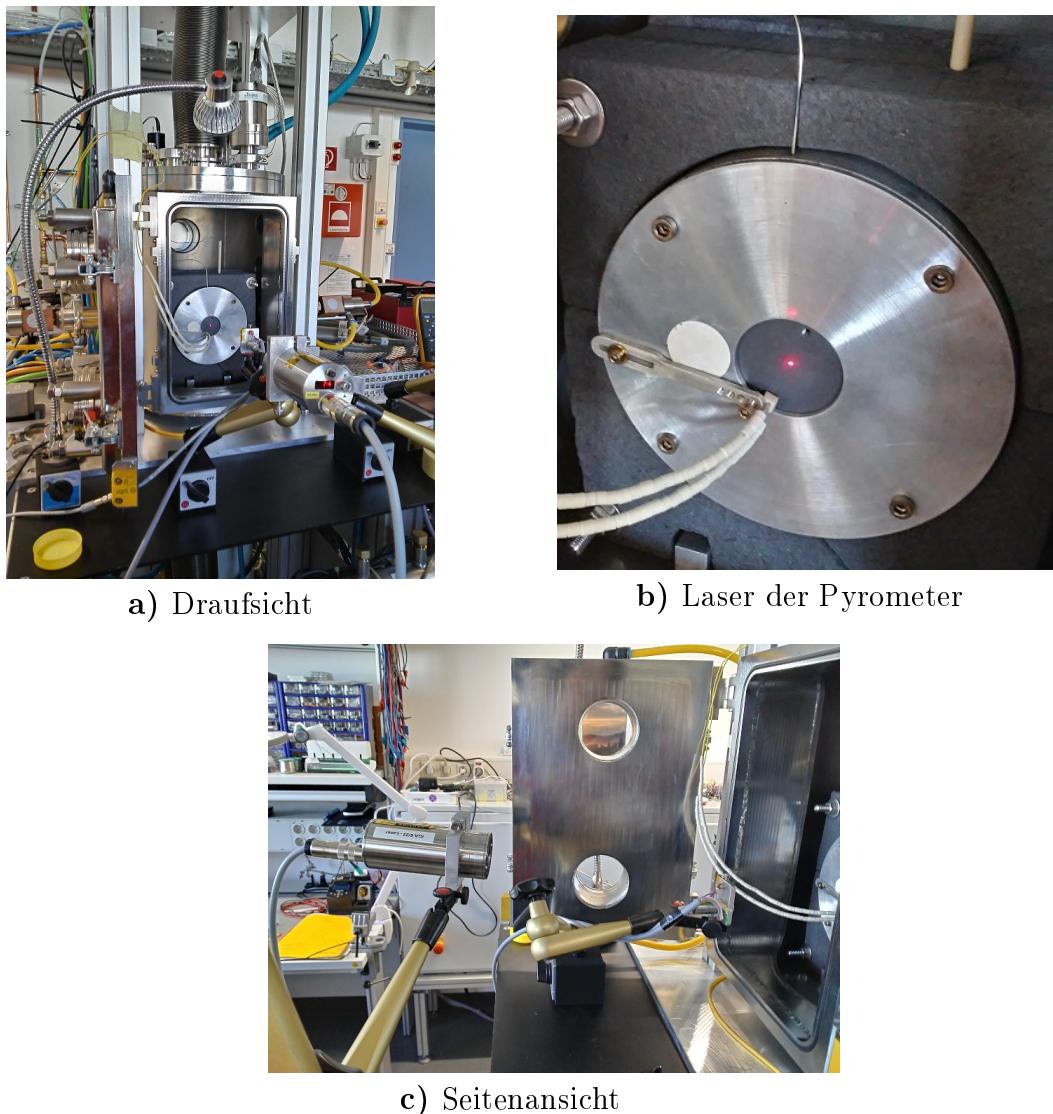


Abbildung 32: Reale Ausrichtung der Pyrometer (Quelle: eigene Darstellung)

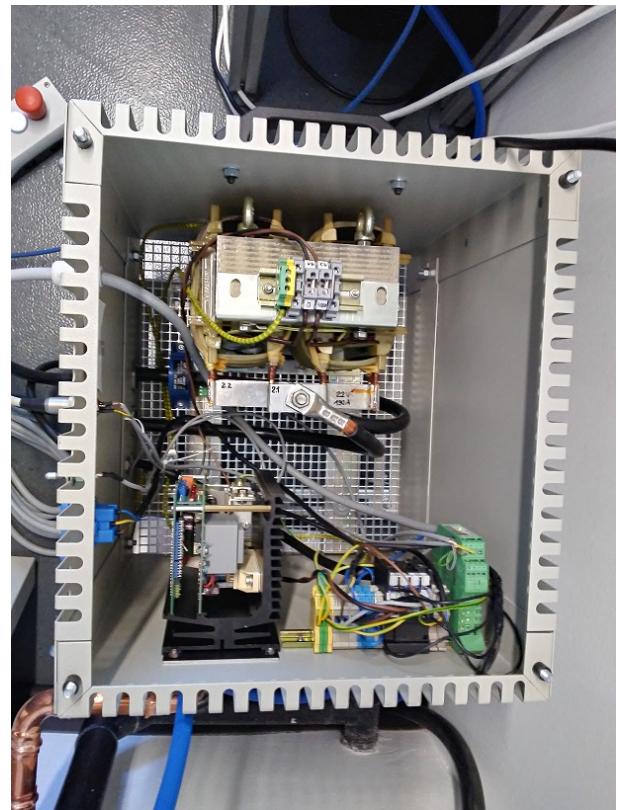
Die Test-CZ-Anlage wird in dem Aufbau mit dem Grafit-Heizer von einer Leistungseinheit<sup>8</sup> gespeist. Dieser bekommt sein Steuersignal von dem Eurotherm bzw. vom Arduino. In Abbildung 33a kann man das Gerät sehen.

---

<sup>8</sup> Besteht aus Thyristorsteller, Stromsensor und Transformator



a) Leistungseinheit (Quelle: eigene Darstellung)



b) Leistungseinheit innen (Quelle: eigene Darstellung)

Abbildung 33: Versorgung Grafit-Heizer durch Leistungseinheit (Quelle: eigene Darstellung)

Das letzte zum Aufbau des Experiment werden die zusätzlichen genutzten Geräte sein. Die gesamte Steuerung des Experiments läuft über einen Raspberry Pi 400 mit Linux-Betriebssystem. Die Geräte sind alle über USB-Hubs (USB zu RS232 Kabel) angeschlossen und die Adafruit-Module hängen direkt an dem GPIO-Anschluss des Raspberry Pi dran. Des Weiteren laufen neben dem Temperaturregel- und Emissivitätsprogramm noch Programme vom IKZ. Mit diesen Programmen werden der Strom durch den Heizer geprüft und eine Wärmebildkamera ausgelesen. Zum Abschluss kann man die gesamte Anlage und den Aufbau in der Abbildung 34 (S. 45) sehen.

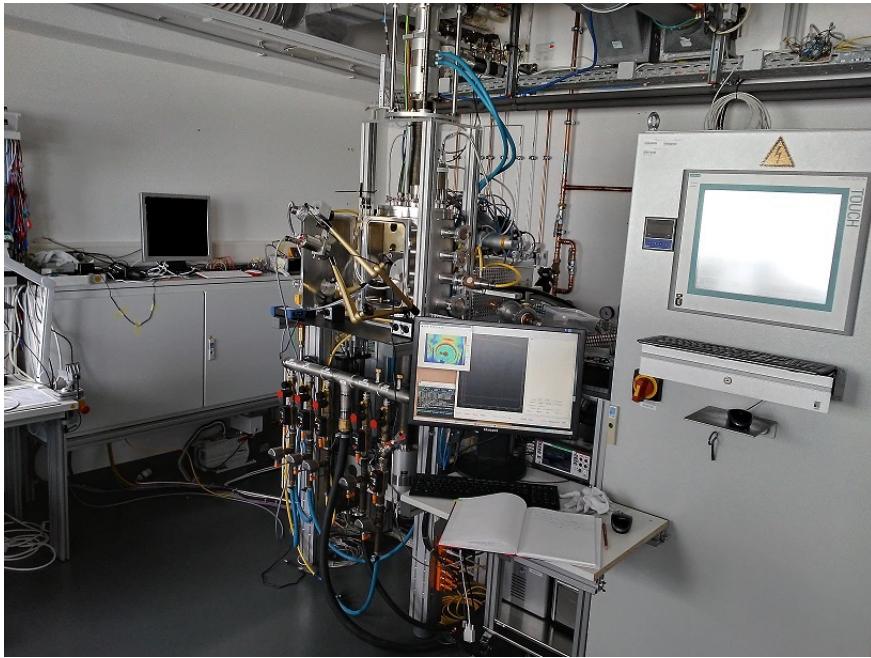


Abbildung 34: Kompletter Aufbau des Experimentes - von links nach rechts: Raspberry Pi, Test-CZ-Anlage mit Pyrometern und Wärmebildkamera, Computer zur Aufnahme des Stromes und der Wärmebildkamerabilder, Schaltschrank der Anlage (Quelle: eigene Darstellung)

In den Abbildungen 83 bis 89 im Anhang kann man den gesamten Aufbau des Experimentes als Schaltplan sehen.

## 4.2 Erklärung des Programms

Nachdem nun der Aufbau des Experimentes bekannt ist, wird der Programm-Aufbau erklärt. Diese Programme sind unter Kapitel 7.2 im Anhang zu finden. Alle im folgenden genannten Programme, sowie Beispiele zu Rezept und Parameterliste sind dort zu finden. Das Programm arbeitet mit dem in Kapitel 4.1 genannten Geräten und liest diese aus oder sendet ihnen Befehle und Werte. Wie man aus dem Messaufbau schon erfahren hat gab es ein Experiment zum Bestimmen von Emissionsgraden im Praktikum. Das im Folgenden erläuterte Programm baut auf dem im Praktikum erstellten Programm auf. Das Programm wurde ebenfalls am und für das IKZ geschrieben. Aus dem Grund sind Teile des alten Programms übernommen worden. Das Programm kann auf der Internetseite [Fun21b] angesehen werden. Für den neuen Aufbau wurde das Programm angepasst, erweitert und verbessert. Mehr zu diesem Thema folgt dann in den Unterkapiteln.

### 4.2.1 Hauptprogramm

Zunächst wird das Hauptprogramm erläutert. In dem Programm finden die gesamten Messungen, Grafik-Erzeugungen, Text-Datei-Arbeiten (Erstellung und Erweiterung) und die Emissionsgrad-Anpassung statt, welche im späteren Verlauf des Kapitels erläutert wird. Die große Änderung zu dem alten Programm ist das Arbeiten mit einer Parameterliste, einem neuen Heizer und der objektorientierten Programmierung. Des Weiteren wurde die manuelle Bedienung in dem neuen Programm entfernt. Das Programm läuft nur noch mit einem Heizrezept. Das bedeutet, dass große Teile vollständig aus der Quelle [Fun21b] entnommen wurden. In der Quelle wird das alte Programm erläutert, weshalb die folgenden Erklärungen für dieses und Kapitel 4.2.2 zu der Quelle ähnlich sind.

Das Hauptprogramm *hauptprogramm.py* arbeitet über die Python-Bibliothek tkinter. Mit dieser Bibliothek kann man graphische Umgebungen (kurz GUI) erzeugen. In der Abbildung 35 kann man diese GUI sehen. Mit der GUI kann man die Messreihe starten und zu jeder Zeit beenden. Des Weiteren kann man das Auto-Scaling der Temperaturmessungen und den Live-Plot per Knopfdruck pausieren. Eine weitere Funktion der GUI ermöglicht es, den Live-Plot zu jeder Zeit zwischenspeichern.

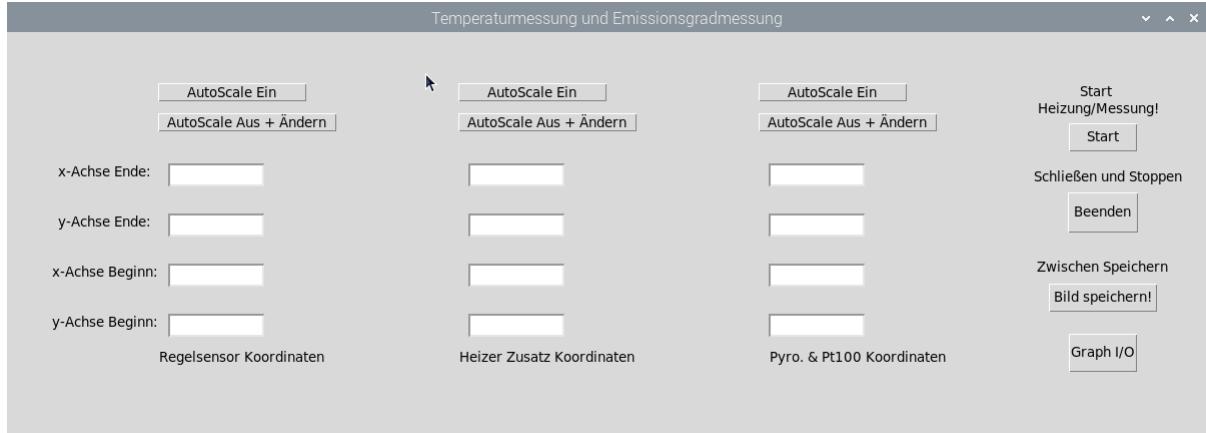


Abbildung 35: GUI des neuen Programmes (Quelle: eigene Darstellung)

Nachdem das Fenster, also die GUI, durch tkinter erstellt wurde wird ein Loop ausgelöst. Solange der Startknopf unbetätigt ist, wird der Loop alle 10 ms wiederholt. Mit der Betätigung des Knopfes wird die Messung und die Abarbeitung des Rezeptes gestartet. Danach probiert das Programm die Funktion *get\_Measurement()* jede Sekunde aufzurufen, was aber durch die langsamten Geräte wie Eurotherm und Arduino nicht geschafft wird. Besser gesagt dauert die Funktion durch die Menge der Abfragen und Wertänderungen an den Geräten länger als eine Sekunde. In jedem Loop werden die Messwerte geholt und in Text-Dateien notiert, entschieden ob die Emissionsgradanpassung anfangen darf und der Live-Plot aktualisiert. Mit dem Knopf Beenden wird der Graph gespeichert, was auch bei Rezept Ende automatisch geschieht. Wie das Rezept bearbeitet wird und dieses aussieht wird in Kapitel 4.2.3 erläutert.

Das Programm erstellt drei Text-Dokumente und ein Diagramm. Zu jedem Zeitpunkt können aber Zwischenergebnisse als PNG Datei gespeichert werden. Die drei Text-Dateien beinhalten folgende Daten:

- Die erste Datei enthält Temperaturdaten von Pyrometern, Regelsensor und Oberflächensensor, sowie die Zeit (absolut und relativ) und je nach Heizerwahl die Ausgangsleistung (Eurotherm) oder die Heizplattentemperatur (IKA-Heizplatte),
- Die zweite Datei beinhaltet eine Aufzählung der Emissionsgradanpassungsschritte, also alle 16 Schritte, die bei einer Anpassung auftreten können, sowie zeitliche Unterbrechungen und die Endzeit der Anpassung,
- Die dritte Datei beinhaltet nur noch die Solltemperatur, sowie die Vergleichstemperatur und die gemessenen bzw. angepassten Emissionsgrade aus dem 16. und letzten Schritt der Anpassung.

In dem Diagramm, siehe Abbildung 36 (S. 51) für ein Beispiel, werden je nach Auswahl von Messgeräten und Heizer bestimmte Diagramme angezeigt. Das Diagramm zeigt immer vier Diagramme an - Temperatur des Reglers mit Sollwert, Temperaturen von Adafruit Pt100 und Pyrometern, Ausgangsleistung in Prozent oder Heizplattentemperatur und zuletzt die Emissionsgrade.

Des Weiteren kann man, wenn es gewollt ist, eine Logging-Datei erstellen. Dafür sind über die ganzen Programmzeilen Logging-Aufforderungen programmiert. In der Logging-Datei kann man dann sehen, ob die gewünschten Zeilen oder Messwerte (sowie Listengrößen) auftraten.

Durch die objektorientierte Programmierung können über die Parameterliste viele Pyrometer schnell einprogrammiert werden. Die Grundlagen zur Emissivität kann in Kapitel 2.5 und die der Pyrometer kann in Kapitel 2.4.1 nach gelesen werden. Im Folgenden wird die Funktion für die Emissionsgradanpassung gezeigt. Wie schon erwähnt wird diese in *get\_Measurment()* aufgerufen und befindet sich in *hauptprogramm.py*.

```
#####
def Emissions_Anpassung(Temp_Pyro, Temp_Oberf, e_Alt, e_Drauf, o_Grenze, u_Grenze):
#####
    if Temp_Pyro != Temp_Oberf:
        e_Drauf = e_Drauf/2
    if Temp_Oberf > Temp_Pyro:
        e_Alt = round(e_Alt - e_Drauf,1)
        if e_Alt < u_Grenze:
            e_Alt = u_Grenze
            e_Drauf = e_Drauf * 2
    if Temp_Oberf < Temp_Pyro:
        e_Alt = round(e_Alt + e_Drauf,1)
        if e_Alt > o_Grenze:
            e_Alt = o_Grenze
            e_Drauf = e_Drauf * 2
    e_Neu = e_Alt
    return e_Neu, e_Drauf
```

In diesen Zeilen werden alle Pyrometer bearbeitet, aus Darstellungsgründen wurden die Kommentare aus dem Programm hier entfernt. Für die Bestimmung des Emissionsgrades braucht man die Temperatur des Pyrometers und die Oberflächentemperatur des Materials. Zudem muss die Pyrometertemperatur bei  $e = 100\%$  kleiner sein als die der Oberfläche. Dieser Umstand hat folgenden Grund: sobald der Emissionsgrad sinkt, steigt die gemessene Temperatur der Pyrometer an. Wenn der Emissiongrad steigt, sinkt die gemessene Temperatur. Das Programm begrenzt den Emissionsgrad auf 100 % und verändert diesen dann nicht, deshalb muss die Pyrometertemperatur am Anfang kleiner sein, sonst würde es heißen, dass das Material einen Emissionsgrad von 100 % hat.

Während des Programmes wird der Emissionsgrad immer bei 100 % sein. Sobald der durch das Rezept einprogrammierte Bereich erreicht ist, beginnt ein Vergleich der beiden Temperaturen (Oberfläche und Pyrometer). Der Emissionsgrad wird bei jedem Durchgang geändert. Die große Rolle dabei spielt die Pyrometertemperatur, die nämlich an die Oberflächentemperatur angepasst wird. In den oben gezeigten Programmzeilen werden in jedem Durchgang für jedes Pyrometer die Temperaturen verglichen. Die Variable *e\_Drauf* spielt dabei die größte Rolle. Zu Beginn wird diese auf 100 gesetzt und dann in jedem Schritt halbiert. Je nach Vergleichsausgang wird dieser Wert von dem Emissionsgrad abgezogen oder auf den Emissionsgrad addiert. Durch die Grenze wird der erste Wert von 50 immer von den anfänglichen 100 % abgezogen. Im nächsten Durchgang wird dann der Wert von 25 abgezogen oder addiert. Diese Prozedur wird im Programm 16 Mal durchgeführt. Nach 10 Läufen ist der Wert in *e\_Drauf* bereits so klein, dass der Emissionsgrad sich nicht mehr ändern wird. Die übrigen Durchgänge sind zur Kontrolle da und werden in einer Text-Datei gespeichert. Die Pyrometer können nicht mehr als eine

Nachkommastelle genaue Werte bearbeiten und ausgeben.<sup>9</sup> Nach diesem Prozedere sollten die Temperaturkurven übereinander liegen. Dies zeigt dann, dass der Emissionsgrad bestimmt wurde.

Nach dem nun die Funktion des Hauptprogrammes erläutert wurde folgen in Kapitel 4.2.2 die Geräte Befehle in separaten Bibliotheken und in Kapitel 4.2.3 die weiteren Dateien zur Steuerung der Temperatur und Daten.

### 4.2.2 Python Bibliotheken für Einzelgeräte

Jedes der Geräte aus Abbildung 24 hat seine eigene Bibliothek, somit gibt es drei Bibliotheken - *heizer.py*, *pyrometer.py* und *adafruit.py*. Diese drei Bibliotheken arbeiten mit dem in Kapitel 4.2.1 genannten Programm zusammen. Der große Unterschied zu dem Praktikums-Programm aus der Quelle [Fun21b] ist, dass die Geräte nun objektorientiert programmiert wurden. Durch diesen Umstand ist das Programm deutlich flexibler für eine Vermehrung der Pyrometer und Pt100-Widerstandsthermometer, die über die Adafruit-Module gesteuert werden.

Die Bibliothek *heizer.py* besteht derzeitig aus den Klassen für die IKA-Heizplatte und dem Eurotherm-Regler. Die IKA-Heizplatte war der Heizer für die im Praktikum durchgeführten Messreihen. Für die neuen Messreihen wird aber der Eurotherm bzw. auch die Emulation mit Arduino die Messreihen bestimmen. Die Eurotherm-Programmierung war daher der Kern des neuen Programms. Die Befehle wurden wie in Kapitel 3.3 gezeigt programmiert. Die Befehle der IKA-Heizplatte sind in dem Programm geblieben, damit man den Heizer auch wechseln kann. Damit dieser Wechsel vonstatten gehen kann, wurden bestimmte Funktionen in den Klassen der Heizer identisch genannt. Die Schnittstelle beider Geräte wird identisch programmiert und wird von der Klasse *Heizer* initialisiert. Eine weitere Funktion der Eurotherm-Klasse ist die Arduino-Nutzung. Über die Parameterliste kann diese für die in Kapitel 3.4 gezeigte Variante in Betrieb genommen werden.

Bei der Bibliothek *pyrometer.py* handelt es sich um die Befehle für die lang- und kurzweligen Pyrometer. Die Befehle und das Ursprungs-Programm wurde vom IKZ gestellt. Für die Verbesserung des Praktikumsprogramms wurden auch diese zu Klassen umgearbeitet. Zudem waren die Pyrometer-Bibliotheken vorher getrennt, was sich nun geändert hat. Bei der Programmierung der langwelligen Pyrometer muss man bei der Programmierung der Schnittstelle etwas achtgeben, da diese über ein Array angeschlossen sind (siehe Abbildung 24d S. 38). Über die Extra-Klassen *Array* und *Pyrometer* wird die Schnittstelle geregelt. Die Programmierung der Schnittstelle für Heizer und Pyrometer sehen so aus wie in Abbildung 7 (S. 15).

Die Bibliothek *adafruit.py* funktioniert anders als die anderen beiden Bibliotheken. Für die Funktion der Adafruit-Module müssen die Pins des Moduls mit dem Raspberry Pi verbunden werden. Das Modul arbeitet dann erst mit den Bibliotheken *board*, *digitalio* und *adafruit\_max31865* zusammen. Die Bibliotheken können einfach über das Konsolen-Fenster installiert werden. Im Anhang (Kapitel 7.2) findet man die Referenzen für die gerade genannten Bibliotheken.

### 4.2.3 Rezept und Parameterliste

Den letzten Punkt der Programmierung bilden die Rezepte für die Temperaturregelung und die Parameterliste. Die Parameterliste ist eine YAML-Datei und wird auch über die gleichnamige Python-Bibliothek eingefügt. Im Folgenden wird der Code für die Parameterliste kurz gezeigt.

<sup>9</sup>Die Änderung der Werte sieht wie folgt aus: 100 -> 50 -> 25 -> 12,5 -> 6,25 -> 3,125 -> 1,5625 -> 0,78125 -> 0,390625 -> 0,1953125 -> 0,09765625. Ab der letzten Zahl wird der Emissionsgrad durch die Limitierung der ersten Nachkommastelle nur noch minimal angepasst.

Der Code in dem Kapitel stammt aus dem Programm *hauptprogramm.py* und der Parameterliste selbst.

```
config_file = 'config_Parameter.yml'  
with open(config_file) as fi:  
    config = yaml.safe_load(fi)
```

Nachdem die Parameterliste initialisiert ist und dem Programm zur Verfügung steht, kann man die Daten aus der Liste ziehen. Die Parameterliste besteht aus Dictionaries. Durch diesen Umstand kann man dann die Geräteanzahl sehr einfach erweitern. Im Großen und Ganzen muss man nur einen alten Eintag kopieren und einfügen und dann die Parameter ändern. Das Auslesen kann auf verschiedene Arten erfolgen. Einige werden im Folgenden gezeigt.

```
pyroKW = {}  
if 'Pyrometer_KW' in config:  
    for name, data in config['Pyrometer_KW'].items():  
        pyKW = pyrometer.PyrometerKW(name, **data)  
        pyroKW.update({name: pyKW})  
        pyKW.anpassung(100,100)
```

In dem Beispiel werden die Daten unter dem Punkt *Pyrometer\_KW*, wenn vorhanden, ausgelesen und ein Objekt für die Daten erstellt. Dieses Objekt wird in einem neuen Dictionary gespeichert und bei Bearbeitung wieder aufgerufen.

```
delay_heiz = config['Delay']['Heizer']  
heizer.serial_delay(delay_heiz)
```

Des Weiteren kann man auch ganz bestimmte Variable auslesen lassen. Dies kann man im oben stehenden Beispiel sehen. In dem Beispiel wird der Wert für den Delay des Heizers abgefragt.<sup>10</sup> Die gesuchte Variable liegt unter *Delay* und dann unter *Heizer*. Um diesen Umstand zu sehen wird die Passage aus der Parameterliste im Folgenden gezeigt.

```
Delay:  
Heizer: 0.5
```

Zu guter Letzt wird das Rezept, der Träger des Programms, erläutert. Im Gegenzug zur Programmierung aus Quelle [Fun21b] ist für die neue Programmierung nur der manuelle Teil verschwunden. Das Aussehen und die Bearbeitung des Rezeptes haben sich nicht geändert.

---

<sup>10</sup>Delay zwischen Senden und Lesen eines Befehls - Arduino 0,5 s und für Eurotherm 0,1 s - Erfahrungswerte

Das Rezept wird in einem Text-Dokument erstellt, welches im Programm ausgelesen wird. Diese Auslesung sieht wie folgt aus.

```
StartConfig = False
loop = 0
nextTempIn = ''
TempTrep = []
TempArea = []
TempTime = []
Config_File = config['Strings']['Rezept']
cp = configparser.ConfigParser()
cp.read(Config_File)
for section_name in cp.sections():
    if section_name == 'Heating':
        for name, zeile in cp.items(section_name):
            TempTrep.append(zeile.split(',') [0])
            TempArea.append(zeile.split(',') [1])
            TempTime.append(zeile.split(',') [2])
print('Start des Rezeptes')
print(f'Sollwerte      = {TempTrep}')
print(f'Sollwertbereich = {TempArea}')
print(f'Zeiten im Bereich = {TempTime}')
print()
if args.log == True:    logging.info('Rezept Eingelesen')
```

Über die Zeilen werden die Rezept-Variablen initialisiert und ausgelesen. Das Rezept dient der Temperaturregelung (Sollwertvorgabe) und löst die Emissionsgradanpassung aus. In dem Rezept gibt es drei Werte - Sollwert, Toleranz und Zeit, in der der Sollwert mit Beachtung der Toleranz bleiben soll. Im Programm *hauptprogramm.py* wird in der Funktion *get\_Measurement()* die Bearbeitung des Rezeptes durchgeführt. Ein Rezept-Zyklus ist erst dann abgeschlossen, wenn die angegebene Zeit geschafft wurde. Unterbrechungen einer Emissionsgradanpassung werden in der zweiten Text-Datei gespeichert (siehe S. 46). Sollte der Sollwert-Bereich in der Zeit verlassen werden, so beginnt die Anpassung von vorne. Diese Funktion ist gewollt, da es zu Beginn oder auch im weiteren Verlauf zu Überschwingungen kommen kann.

Das Programm zur Anpassung prüft zunächst, ob der Istwert im Sollwert-Bereich liegt. Solange dies nicht der Fall ist, werden die Pyrometer einmalig auf 100 % Emissionsgrad gesetzt. Zur gleichen Zeit werden die Variablen für die Anpassung auch auf 100 gesetzt. Die Variable *StartConfig* verhindert das Verändern der Variablen und der Pyrometer, wenn diese auf False steht (Initial-Wert siehe Code S. 50). Sollte der Sollwert-Bereich erreicht worden sein, so wird die Variable auf True gesetzt. Die Werte werden bei Verlassen des Bereiches und des Umstandes True zurückgesetzt, wie oben beschrieben. Bei False wird die Variable auf False gesetzt und der aktuelle Emissionsgrad ausgelesen.

Wenn der Bereich erreicht ist und *StartConfig* auf False steht, wird das Ende des Zyklus berechnet. Diese Zeit wird im Konsolenfenster angezeigt und in der zweiten Text-Datei notiert. In den folgenden Programm-Loops wird die Anpassung durchgeführt. Diese Anpassung wurde im Kapitel 4.2.1 auf Seite 47 erläutert. Die 16 Schritte sind der Programmierung von  $<=15$  geschuldet. Mit dem Erreichen der zu Beginn bestimmten Zeit wird der Zyklus beendet und der nächste Sollwert an den Heizer übergeben. Die Emissionsgrade werden wie schon gesagt auf 100 % zurückgesetzt.

Zum Schluss des Kapitels wird ein Beispiel für das Rezept gezeigt.

[Heating]

```
zy1: 50,0.5,20
zy2: 100,0.5,20
zy3: 150,0.5,20
zy4: 200,0.5,20
```

Der erste Zyklus würde eine Solltemperatur von 50 °C mit einer Toleranz von  $\pm 0,5$  °C verursachen. Der Heizer probiert die 50 °C zu erreichen, doch das Programm würde bei 49,5 °C und 50,5 °C mit der Anpassung beginnen. Dieser Bereich soll dann für 20 min gehalten werden, damit der Zyklus abgeschlossen ist.

### 4.3 Messreihen und Ergebnis

In dem Kapitel werden die durchgeführten Messreihen für die Emissionsgrade gezeigt. Die Messreihen wurden einmal durch den Eurotherm mit Arduino geregelt und zum anderen nur mit dem Arduino. Im Folgenden wird zunächst das letzte Ergebnis dieser Messreihen gezeigt. Neben diesen Messungen wurden zudem die Leistungsmessungen, die in Kapitel 5.3.3 zum Grafit-Heizer erläutert werden, durchgeführt.

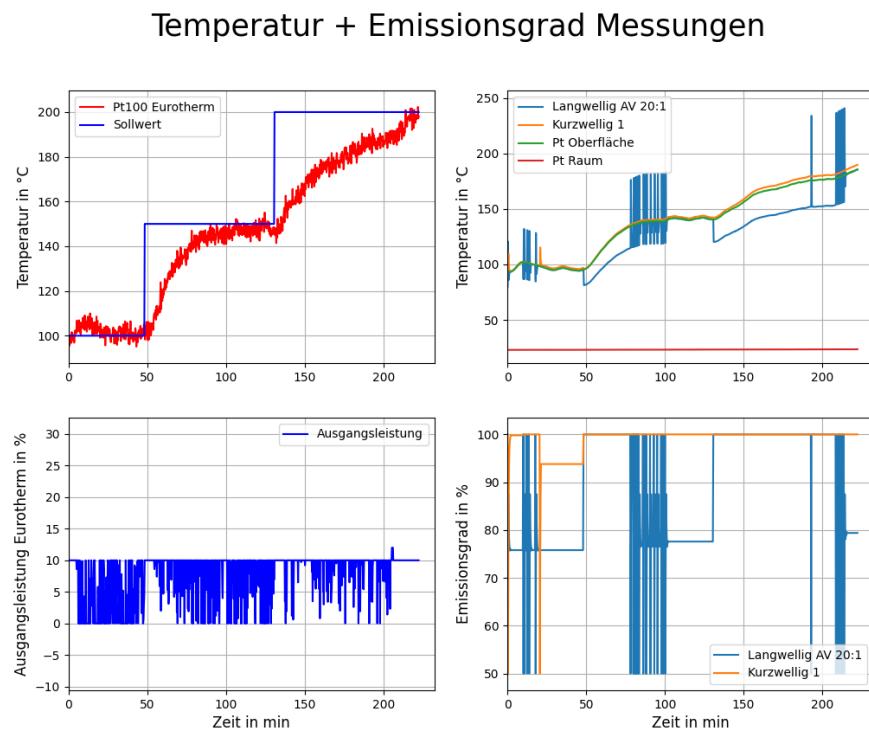


Abbildung 36: Messreihe mit Eurotherm PI-Regelung (Quelle: eigene Darstellung)

## Temperatur + Emissionsgrad Messungen

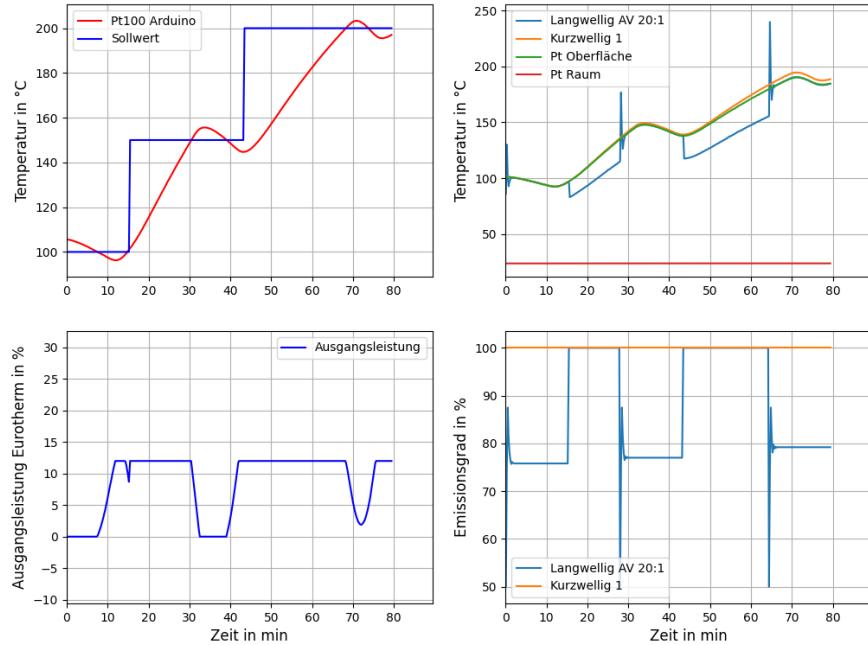


Abbildung 37: Messreihe mit Eurotherm PI-Regelung (Quelle: eigene Darstellung)

Eine Erkenntnis, die man in den Abbildungen 36 und 37 sehen kann ist, dass das KW Pyrometer hier nicht so wie bei den Heiztesten gemessen hat. Dadurch, dass die Temperatur die meiste Zeit über der der Oberfläche lag, wird hier der Emissionsgrad mit 100 % bestimmt. Durch dieses Verhalten ist die Anpassung des KW Pyrometers nicht mehr völlig richtig, was auch mit der Programmierung zu tun hat. Damit ist der Umstand gemeint, wenn der Temperaturwert im Späteren unter den Wert der Oberfläche fällt. Die Anpassung des Emissionsgrades, in Kapitel 4.2.1 erläutert, wird nur 16 Mal vom Programm durchgeführt und wenn die Temperatur erst beim 13. Mal fällt, beginnt die Anpassung für dieses Pyrometer erst da. Da Probleme bei dem kurzwelligen Pyrometer auftraten, werden auch Messdaten aus den Heiztesten herangezogen. Die Erkenntnis aus Kapitel 3.6 ist auch hier zu sehen. Das Problem bei dem Eurotherm ist nun, dass die Temperaturmessung sehr stark schwankend ist und somit das Rezept stark angepasst wurde. Zusätzlich musste die Leistung von 10 % am Ende der Messung in Abbildung 36 gehalten werden, damit die 200 °C erreicht werden konnten. Darauf wurde der Leistungswert für die Arduino-Messung auf 12 % erhöht. Auch bei der Messung musste das Rezept verändert werden. Bei beiden Messungen brauchte man einen Sollwertbereich von mehr als  $\pm 5$  °C. Die Grundlage zu den Rezepten wird in Kapitel 4.2.3 erläutert.

Eurotherm Rezept:

```
Solltemperaturen:      ['100', '150', '200']
Regelbereich:          ['8', '8', '8']
Zeit im Regelbereich: ['30', '30', '30']
```

Arduino Rezept:

```
Solltemperaturen:      ['100', '150', '200']
Regelbereich:          ['8', '8', '8']
Zeit im Regelbereich: ['15', '15', '15']
```

## Emissionsgrad über Temperatur

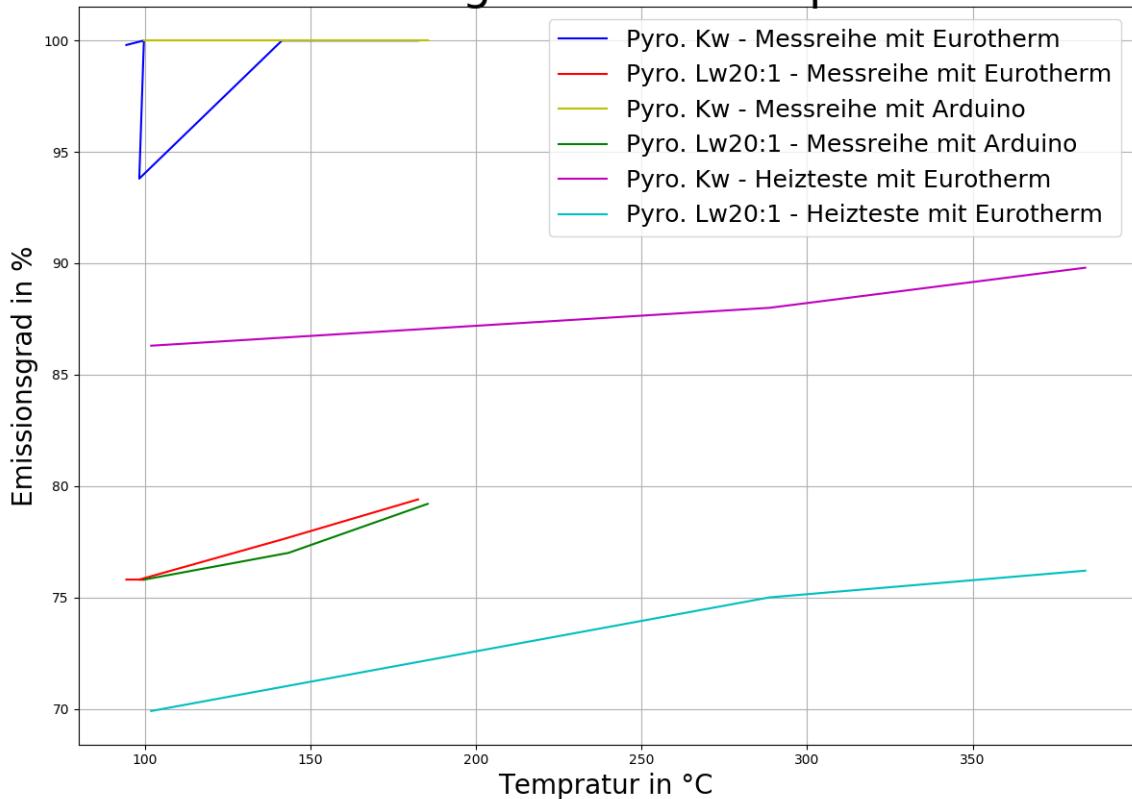


Abbildung 38: Zusammenfassung der Emissiongrade (Quelle: eigene Darstellung)

In der Abbildung 38 können die Ergebnisse der Messungen angesehen werden. Die Messreihen in den Abbildungen 36 und 37 haben für das LW Pyrometer sehr ähnliche Ergebnisse. Des Weiteren kann man einen Anstieg des Emissiongrades bei steigenden Temperaturen erkennen. In dem Diagramm werden die Emissionsgrade zu der Vergleichstemperatur geplottet und nicht zur Solltemperatur. Die x-Achse zeigt daher den Messwert des Oberflächensensors. Die gute Anpassung kann man in den Abbildungen 36 und 37 daran sehen, dass die Kurven übereinander liegen. Wenn die Kurven übereinander liegen, heißt das, dass die Temperaturen nahezu identisch sind und somit der Emissiosngrad richtig ermittelt wurde. Um dies zu zeigen, werden in Abbildung 39 die oberen rechten Diagramme aus den Abbildungen 36 und 37 dargestellt. Die grüne und die blaue Kurve liegen während der Anpassung übereinander.

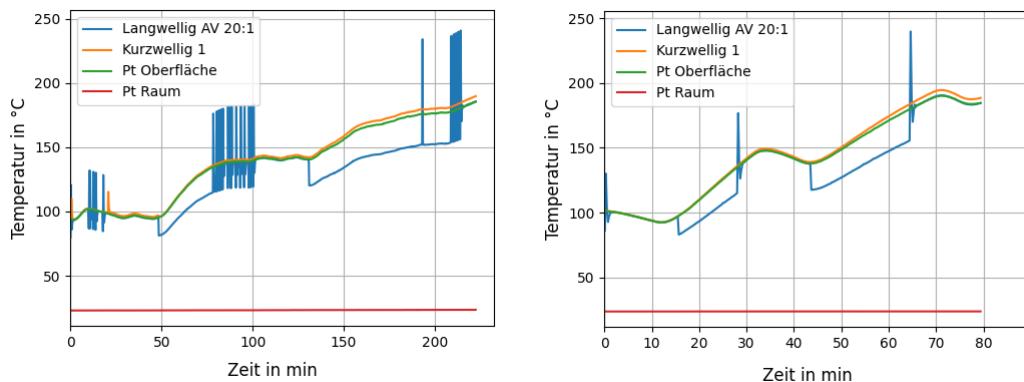


Abbildung 39: Ausschnitt der Abbildungen 36 und 37 (Quelle: eigene Darstellung)

Die Kurven in lila und cyan in Abbildung 38 sind eine Zusammenstellung von Ergebnissen, die während der Heizteste bestimmt wurden. Die in Kapitel 4.1 gezeigte Ausrichtung war bei den Heiztesten anders. Dies liegt daran, dass der erste Grafit-Heizer durch die hohen Temperaturen zerstört wurde und die Anlage deswegen umgebaut wurde bzw. der Heizer ausgetauscht wurde. Aus diesem Grund wurde bei den oben gezeigten Messreihen auch nur bis 200 °C und max. 12 % gegangen. Man wollte den Heizer nur bis 400 °C betreiben, damit das Grafit nicht wieder beschädigt wird.

Die Kurve in cyan, aus Abbildung 38, ist in der Nähe des Messreihen-Ergebnisses, und der Emissionsgrad wurde auch mit der Temperatur größer. Die Kurve in lila, aus Abbildung 38, zeigt mehrere Ergebnisse<sup>11</sup> für das KW Pyrometer. Auch hier kann man ein Anwachsen des Emissionsgrades mit der Temperatur sehen.

In der Quelle [Fun21a] auf Seite 11 findet man das Ergebnis des alten Aufbaus. Für Grafit war der Emissionsgrad bei dem KW Pyrometer zwischen 94 und 97 % groß und bei dem LW Pyrometer war der Emissionsgrad bei ca. 57 %. Des Weiteren kann man bei dem Ergebnis sehen, dass der Emissionsgrad weitestgehend gleich blieb über die Temperatur. Mit dem neuen Aufbau kann man in dem Ergebnis bereits eine Besserung sehen. Der Emissionsgrad für das LW Pyrometer liegt zwischen 70 und 80 %. Bei dem KW Pyrometer wurde bereits gesagt, dass bei der letzten Messung (blaue und gelbe Kurve in Abbildung 38) Probleme entstanden, die so nicht erwartet wurden. Aus der lila Kurve kann man sehen, dass der Emissionsgrad in einen Bereich von 85 bis 90 % liegt. Eventuell ist die veränderte Ausrichtung des Pyrometers an der Änderung schuld.

In den Manuals für die Pyrometer werden auch Vergleichswerte für die Emissionsgrade bei Grafit genannt. In der Quelle [Pyra] auf Seite 24 findet man für das KW Pyrometer einen Emissionsgrad (bei einer Wellenlänge von 2,3  $\mu\text{m}$ ) von 80 bis 90 %. Die lila Kurve würde dem vom Emissionsgrad her entsprechen. Bei dem LW Pyrometer findet man den Emissionsgrad in der Quelle [Pyrb] auf Seite 5-19. Dort wird der Emissionsgrad für einen Wellenlängenbereich von 8  $\mu\text{m}$  bis 14  $\mu\text{m}$  mit 75 bis 92 % angegeben. Das Ergebnis der Messreihen (grüne und rote Kurve) von Arduino und Eurotherm liegen in dem Bereich.

Zusammenfassend kann man für das Experiment sagen, dass der neue Aufbau ein besseres Ergebnis erzielt hat als der alte Aufbau. Besonders die Messung mit den LW Pyrometern war durch den senkrechten Aufbau der Probe besser, da die aufsteigende Wärme somit den Aufbau nicht störte. Das LW Pyrometer konnte somit deutlich näher an die Probe heran, wodurch der Messfleck sehr klein gehalten werden konnte. Des Weiteren kann man in den meisten Kurven die Abhängigkeit zur Temperatur erkennen. Die Emissionsgrade steigen hier mit wachsender Temperatur.

---

<sup>11</sup>Es sind nur drei Ergebnisse bei den Solltemperaturen 100, 300 und 400 °C. Das selbe gilt für die Kurve in cyan!

## 5 Vermessung der elektromagnetischen Parameter

In dem Kapitel werden die Programme, Experiment-Aufbauten und Ergebnisse für die Messung von elektromagnetischen Parametern dargestellt. Zunächst wird sich das Kapitel mit der Magnetischen Flussdichte befassen. Dafür wird ein Hall-Sensor in Unterkapitel 5.1 (S. 55) kalibriert und mit der Kalibrierung wird dann in Unterkapitel 5.2 (S. 65) das magnetische Profil der Spule in der Test-CZ-Anlage bestimmt. Zu guter Letzt wird dann in Unterkapitel 5.3 (S. 78) die Messung der Heizer-Leistung für Grafit-Heizer und Spule gezeigt.

### 5.1 Kalibrierung mit Helmholtzspule und Hall-Sensor

Für die Bestimmung des Magnetfeldprofils (Kapitel 5.2) wird ein Hall-Sensor genutzt. Um bestimmte Zusammenhänge an dem Hall-Sensor zu erkennen, wird der Hall-Sensor kalibriert. Auf den folgenden Seiten wird diese Kalibrierung erläutert.

#### 5.1.1 Messaufbau

Zunächst wird der Aufbau des Experiments erläutert. Die gesamte Messung wird automatisch von dem Programm *hauptprogramm\_Kalibrierung.py* (Kapitel 5.1.2) gesteuert. Der Aufbau besteht aus einer Helmholtzspule, deren Grundlage in Kapitel 2.4.3 erläutert wird. Die Sonde (Hall-Sensor) wird in die Lücke der Helmholtzspule geschoben. Der Hall-Sensor wird am besten mittig platziert.

Das Schema des Aufbaus wird in Abbildung 40 gezeigt. Die Skizze wurde mit dem Zeichenprogramm *EasyEda* erstellt. Über das Programm kann man mehr in der Quelle [Eas] erfahren.

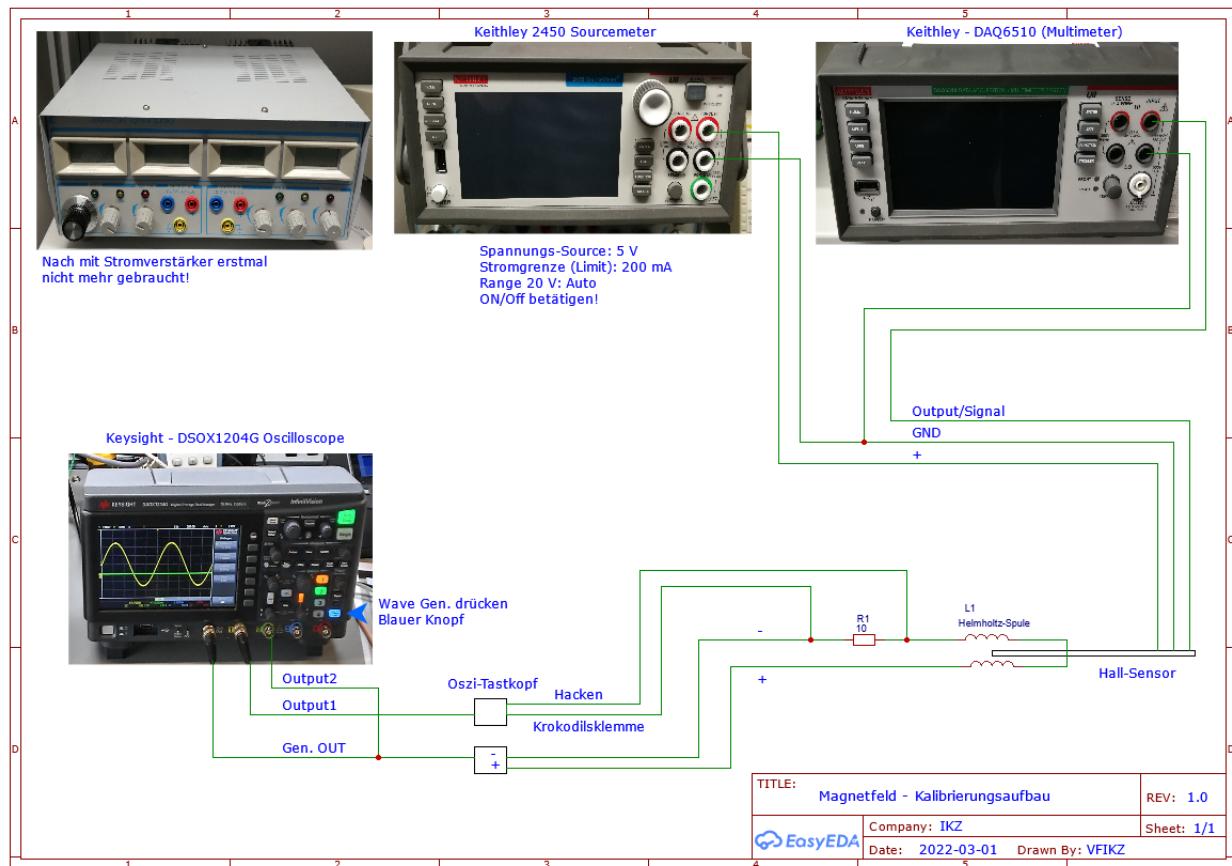


Abbildung 40: Schaltplan Kalibrierung (Quelle: eigene Darstellung)

In dem Schaltplan wird die Helmholtzspule als zwei Spulen dargestellt. Die richtige Spule kann man in Abbildung 41 sehen. Die Spule befindet sich in einer im IKZ gefertigten Halterung und der Kupferdraht wurde dann in deren Einkerbungen gewickelt. Die Spule hat einen Durchmesser von 20 mm und 60 Windungen je Spulen-Hälfte. Des Weiteren hat die Spule einen Vorwiderstand von  $10\Omega$ .



Abbildung 41: Genutzte Helmholtzspule (Quelle: eigene Darstellung)

Für die Kalibrierung werden drei Geräte genutzt. Diese Geräte sind in Abbildung 40 gezeigt. Das Gerät ohne Anschluss (rechts oben) war zu Beginn für die Arbeit mit Stromverstärker gebraucht. Die Arbeit mit dem Stromverstärker wurde wegen zu stark schwankender Werte wieder eingestellt. Als Alternative wurde die in Abbildung 41 gezeigte Helmholtzspule genutzt. Zuvor wurde eine größere Helmholtzspule mit  $50\Omega$  Vorwiderstand verwendet.

Von den drei genutzten Geräten wurden zwei automatisiert. Das Keithley 2450 Sourcemeter ist das Gerät, das nicht programmiert ist (zu sehen Mitte oben in Abbildung 40). Das Gerät liefert die Versorgungsspannung für den Hall-Sensor. Die Geräte DAQ6510 (Multimeter) von Keithley (in Abbildung 40 oben links) und DSOX1204G Oszilloskop von Keysight (in Abbildung 40 unten rechts) wurden automatisiert.

Mit dem Oszilloskop wurde die Spule vom eingebauten Wellengenerator gespeist und die Spannung am Vorwiderstand gemessen. Mit dieser Messung kann man auf den Strom rückschließen. Das Multimeter misst die Hall-Spannung des Hall-Sensors. Mit dieser Spannung kann auf die Magnetische Flussdichte  $B$  zurückgerechnet werden. Diese Umrechnung ist dann ein Punkt in der Auswertung in Kapitel 5.1.3. Der Hall-Sensor hat drei Anschlüsse - Plus, Minus und Output. Der Hall-Sensor ist in Abbildung 42 zu sehen.



Abbildung 42: Genutzer Hall-Sensor (Quelle: eigene Darstellung)

In Abbildung 40 wird das Schema des Aufbaus gezeigt. Um auch den tatsächlichen Aufbau einmal zu zeigen wurde die Abbildung 43 eingefügt. In der Abbildung kann man die Halterung der Helmholtzspule und des Hall-Sensors sehen. Der Sensor wurde von oben in die Helmholtzspule geschoben, und zwar 1,5 cm.<sup>12</sup> Ein weiterer Umstand, der in der Abbildung zu sehen ist, ist dass die Vorwiderstandsspannung von einem weiteren Multimeter überwacht wird.

---

<sup>12</sup>Die Halterung der Spule hat einen Durchmesser von 3 cm, der Spulendurchmesser beträgt 2 cm. Die Mitte der Spule liegt also bei 1,5 cm von außen gesehen.

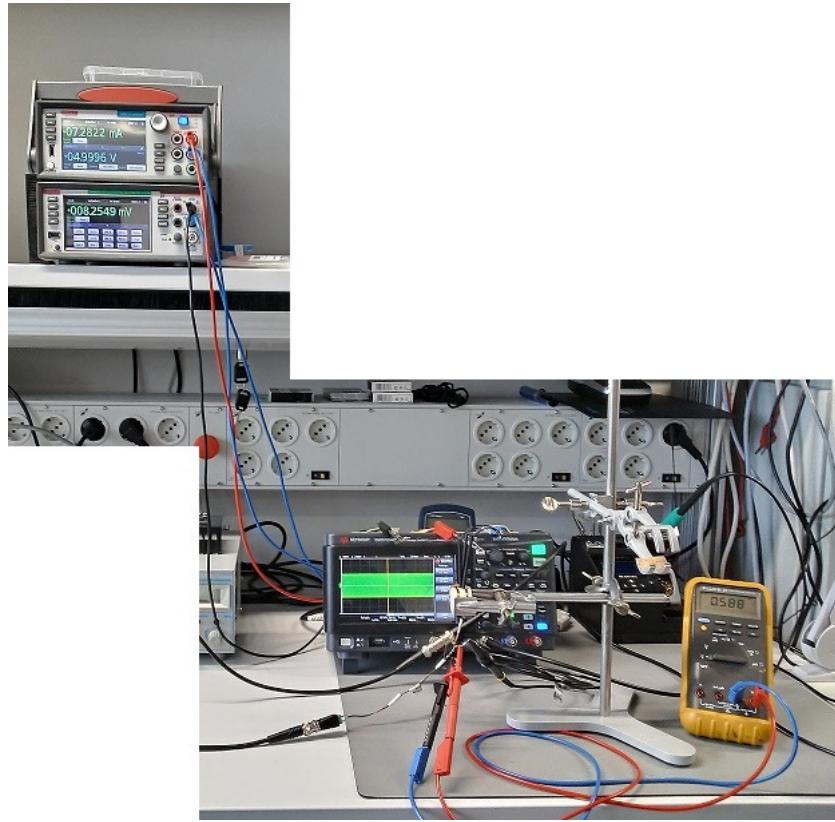


Abbildung 43: Aufbau des Experimentes zur Kalibrierung (Quelle: eigene Darstellung)

### 5.1.2 Erklärung des Messprogramms

Bevor das Programm (*hauptprogramm\_Kalibrierung.py*) zur Bestimmung der Kalibrierdaten erläutert wird, werden erst einmal die Befehle der beiden Geräte (Multimeter und Oszilloskop) erklärt. Dies ist wichtig, da die folgenden Experimente in den Kapiteln 5.2 und 5.3 auch auf diese Geräte zugreifen. Die erklärten Programme sind in Kapitel 7.2 im Anhang zu finden.

Zunächst wird die Programmierung der Schnittstelle erläutert. Bei dem DAQ6510 Multimeter wird die Schnittstelle über die Code-Zeilen aus Abbildung 7 (S. 15) initialisiert. Diese Code-Zeilen sind bei den ganzen hier genutzten Programmen Standard. Das Gerät hat einen Anschluss für RS232.

Für das DSOX1204G Oszilloskop sieht dies etwas anders aus. Um das Gerät initialisieren zu können, braucht man die Bibliothek *usbtmc* und die beiden Nummern *idVendor* und *idProduct*. Die beiden Nummern kann man über den Konsolen-Befehl *lsusb* finden.<sup>13</sup> [Vgl. [Git]] Wie man die Schnittstelle für das Oszilloskop genau einstellt, kann man in der Quelle [Git] nachlesen. Der Grund für die andere Schnittstelle ist, dass das Oszilloskop keine RS232-Schnittstelle besitzt, sondern nur einen USB-Anschluss.

Sobald die Schnittstellen laufen, kann man sich mit den Befehlen der Geräte befassen. Die Befehle richten sich nach dem *Standard Commands for Programmable Instruments* (kurz SCPI) (Quelle [Mul] S. 11-1). Die Befehle unterteilen sich in Schreibbefehle und Lesebefehle. Ein Lesebefehl wird durch ein Fragezeichen gekennzeichnet und jeder der Befehle beginnt mit einem Doppelpunkt. Alle Befehle besitzen eine Kurzschreibweise, dafür werden in den Manuals die

<sup>13</sup> Als Konsole kann man z.B. PowerShell nutzen. Bei der Suche der Werte lohnt es sich, den Befehl ohne Gerät auszuführen und dann ein zweites Mal mit. Somit findet man schnell die Nummern heraus!

Befehle teilweise groß und klein geschrieben. Die Großbuchstaben markieren die Zeichen, die im Befehl mindestens stehen müssen. Bei den Befehlen kommen bestimmte Parameter wie z.B. Kanalnummer nach dem Hauptbefehl (bei Lesebefehlen nach dem Fragezeichen). Die Trennung von Parametern und Befehlen wird durch ein Leerzeichen erzielt. Befehle wie z.B. Reset oder Identifikation des Gerätes beginnen mit einem \*. [Vgl. [Mul] S. 11-1 bis 11-6]

In der Quelle [Mul] auf den Seiten 11-1 bis 11-6<sup>14</sup> werden die Befehle noch ausführlicher erläutert. Der obere Absatz sollte eine kleine Zusammenfassung der für die genutzten Befehle wichtigen Formatierungen sein.

In den Tabellen 5 und 6 werden die genutzten Befehle der Geräte gezeigt. Des Weiteren wird auch immer die Seite und Quelle der Befehle genannt, also wo man diese im Manual findet. Die Erklärungen der Befehle beziehen sich auf die genutzten Befehle (Code-Zeilen aus dem Programm *hauptprogramm\_Kalibrierung.py*), also auf die Form des Befehls im Programm. Die Tabelle 5 befindet sich nicht in der Quelle [Mul], sondern soll zeigen, auf welcher Seite in dieser Quelle die entsprechenden Befehle gefunden wurden.

Tabelle 5: Befehle des DAQ6510 Multimeters von Keithley (Quelle: [Mul])

<b>Befehle und Seiten</b>	<b>Nutzung in Programm und Erklärung</b>
:MEASure? S. 12-7 bis 12-9	:MEAS:VOLT:AC? "{buffer_U}", FORM  Mit dem Befehl wird aus dem Buffer (Übergabe durch Parameterliste des Python Programmes) der Messwert für die AC-Spannung in der Form wie der Wert auf dem Bildschirm des Gerätes erscheint (hier mit Einheit).
:TRACe:MAKE S. 12-192 bis 12-193	:TRACe:MAKE "{buffer_U}", 10000  Mit dem Befehl werden der Wertmessbuffer erstellt und die Größe des Buffers festgelegt.
*RST S. 17-7	*RST  Durch den Befehl führt das Gerät einen Reset aus. In dem Programm wird dieser Befehl genutzt, damit der Wertmessbuffer von Messung zu Messung gelöscht wird und keine Fehlermeldung bei erneutem Erstellen auftritt.
*IDN? S. 17-5	*IDN?  Mit dem Befehl wird der Gerätename (Identifikation) erfragt.

Bei den in Tabelle 5 gezeigten Befehlen gibt es im Python-Programm noch die Besonderheit, dass sie mit einem \n an das Gerät gesendet werden. Das Schreiben eines Befehls wird von der Funktion *write* aus der Python-Bibliothek *serial* übernommen.

Auch die folgende Tabelle 6 ist so nicht in der Quelle [Osz] vorhanden. Es soll auch hier nur wieder zu den Seitenzahlen Auskunft geben.

<sup>14</sup>11-1 = Kapitelnummer-Seite im Kapitel

Tabelle 6: Befehle des DSOX1204G Oszilloskops von Keysight (Quelle: [Osz])

Befehle und Seiten	Nutzung in Programm und Erklärung
:WGEN:FREQuency <frequency> S. 107 und S. 692	:WGEN:FREQ {n}  Über den Befehl werden die Frequenzen an den Wellengenerator gesendet. Das n steht für die Frequenzen.
:WGEN:FUNCtion <signal> S. 107 und S. 693 bis 695	:WGEN:FUNC {funktion}  Der Befehl setzt die Funktionsart des Befehls.
:WGEN:VOLTage <amplitude> S. 109 und S. 716	:WGEN:VOLT {amp}  Der Befehl übergibt die Amplitude der Welle.
:MEAS:VRMS? [<interval>][,] [<type>][,] [<source>] S. 83 und S. 406	:MEAS:VRMS? AC, CHAN  Mit dem Befehl wird der RMS-Wert der Spannung an dem angegebenen Kanal ausgelesen (Kanal-Nummer wird vom Programm noch angehängt).
*IDN? S. 57 und S. 124	*IDN?  Mit den Befehl wird der Gerätename (Identifikation) erfragt.

Bei den Befehlen in der Spalte **Nutzung in Programm und Erklärung** kommen die geschweiften Klammern durch den formatierten String. In Python wird ein solcher String durch ein `f` gestartet. Der String steht dann zwischen den Anführungsstrichen. In den geschweiften Klammern werden dann Variablen in den String übergeben. Diese Variablen bekommen durch die Parameterliste ihren Wert. Die Befehle werden dann als String über die Bibliotheksfunktionen `ask` und `write` von `usbtmc` an das Gerät übergeben.

Das Python-Programm für den Messaufbau der Kalibrierung heißt `hauptprogramm_Kalibrierung.py`. Auch dieses Programm besitzt eine Parameterliste mit Yaml (siehe 4.2.3 S. 48).

Im Folgenden wird der Programmablauf beschrieben. Zu Beginn werden neben der Parameterliste auch die beiden Geräte DAQ6510 (Multimeter) von Keithley (in Abbildung 40 oben links) und DSOX1204G Oszilloskop von Keysight (in Abbildung 40 unten rechts) initialisiert. Nach der Initialisierung werden die Parameterlisten-Werte ausgelesen. Darunter wird der Wertmessbuffer für das Multimeter erstellt, der Frequenzbereich und die Bearbeitungsrichtung festgelegt und der Funktions-Typ sowie die Startamplitude des Wellengenerators übermittelt.

Sind die File-Köpfe und die GitHub-Versionsnummer<sup>15</sup> ausgelesen, beginnt der Hauptteil. Die gesamte Messung wird in einer for-Schleife abgearbeitet. Der besondere Teil ist, dass der Strom am Vorwiderstand bzw. die gemessene Spannung gleichbleibend sein soll. Die for-Schleife arbeitet nach der Frequenz. Sagt man dem Programm, dass man von 20 Hz bis 20000 Hz in Schritten von 20 Hz messen will, so gibt die Frequenz die Bearbeitung der Schleife vor. Über die Parameterliste kann man mit *reverse* (auf True setzen) die Bearbeitung der Frequenz umkehren, dabei läuft die Frequenz dann von 20000 Hz bis 20 Hz in Schritten von -20 Hz. Die erste Frequenz wird an das Oszilloskop übergeben und dann je Schleifen-Loop um die beispielhaften 20 Hz reduziert oder erhöht.

Nach dem Einstellen der Frequenz, wird der erste Spannungswert (vom Vorwiderstand) als Effektivwert (kurz RMS) ausgelesen. Dieser Wert wird nun in einer while-Schleife überprüft. Über die Parameterliste kann man einen Sollwertbereich für diese Spannung angeben und den Berichtigungswert für die Generatorenspannung (Amplitude Welle - Spitze-Spitze-Wert) angeben. Ist der aktuelle Wert außerhalb des Bereiches, wird die Amplitude angepasst. Liegt der Istwert über dem Sollwert-Bereich, wird die Amplitude um den Berichtigungswert verringert. Ist der Istwert kleiner, wird die Amplitude bis max. 12 V erhöht. Die 12 V sind die Grenze der Amplitude.

Sobald der Messwert der Vorwiderstandsspannung im Sollwert-Bereich liegt, so werden die Hall-Spannung und die Amplitude des Wellengenerators ausgelesen. Zum Schluss werden die beiden Werte und die Vorwiderstandsspannung in einem Text-Dokument gespeichert und die nächste Frequenz an das Oszilloskop übergeben. Danach wird das Prozedere wieder von vorn beginnen, bis die letzte Frequenz erreicht ist.

Zum Schluss werden noch Besonderheiten der Programmierung genannt. Bei dem Multimeter will man die Hall-Spannung des Hall-Sensors messen. Diese Spannung ist eine AC-Spannung, weshalb man bei dem Auslesen der Spannung dies expliziert in dem Befehl nennen muss. Bei dem Befehl ist aber die DC-Spannung der Default-Wert. Bei den anfänglichen Testen gab es Ausgabe-Schwierigkeiten bei dem Umstellen von DC auf AC. Das Gerät braucht Zeit, um dies zu schaffen. Aus dem Grund bekommt das Gerät durch das Programm beim ersten Nachfrage-Befehl ein 5 s Delay, bevor der Wert ausgelesen wird. Des Weiteren wird der Spannungswert durch den Wertmessbuffer so ausgegeben wie angezeigt. Aus dem Grund wurde die Funktion *um()* geschrieben, die die ausgegebene Einheit mit dem Wert in Volt umrechnet. Damit stehen in dem File immer nur Spannungen in der Grundeinheit.

### 5.1.3 Ergebnis und Auswertung

In den beiden vorherigen Kapiteln wurden der Messaufbau und das Messprogramm erläutert und dargestellt. Mit dem Programm wurde nun eine Kalibrierung erstellt. Die Einstellung für diese Kalibrierung sind:

- Helmholtzspule mit Durchmesser 20 mm und 60 Windungen je Spulen-Hälfte,
- Vorwiderstand von  $50\Omega$ ,
- Start-Amplitude des Wellengenerators bei 12 V und Sinus-Kurve,
- Vorwiderstandsspannung gemessen an Kanal 1,
- Hall-Spannung bei keinem angelegten Magnetfeld Betrag 2,3114 mV,
- Sollwert der Vorwiderstandsspannung Betrag  $550.0\text{ mV} \pm 1.0\text{ mV}$  bei einer Amplituden-Korrektur (Spitze-Spitze-Spannung) von 0,01 V,

---

<sup>15</sup>Während der Arbeit am IKZ wurde das Programm mit GitHub gepflegt. Durch die Arbeit mit GitHub konnte man auf ältere Versionen des Programms zugreifen bzw. die Änderungen sehen. Aus dem Grund wird in nahezu allen Hauptprogrammen bei den Messreihen auch immer die Versionsnummer des Programmes notiert.

- Frequenzverlauf von 20000 Hz bis 20 Hz in -20 Hz Schritten,
- Versorgung Hall-Sensor von Keithley - 2450 Sourcingemeter mit 5 V Versorgung und 200 mA Stromgrenze (am Gerät).

In Abbildung 44 kann man das Ergebnis der Kalibriermessung sehen. In der Darstellung sind derzeitig nur der Strom und die Magnetische Flussdichte berechnet worden. Weitere Auswertungen folgen im Anschluss. Das Diagramm wurde mit dem Auswertungsprogramm *Auswertung\_Text-Datei.py* (im Anhang unter Kapitel 7.2 zu finden) erstellt. Das Programm berechnet die beiden Werte und stellt alle gemessenen Größen und berechneten Größen in je einem Subdiagramm dar und das über die Frequenz.

Der Strom wurde über das Ohmsche Gesetz (Gleichung 14 S. 9) berechnet und die Umrechnungsformel von Hall-Spannung zur Magnetischen Flussdichte wird im Folgenden gezeigt. Die Werte die man mit den folgenden Gleichungen berechnet sind unkalibrierte Werte.

$$B_{Gaus} = \frac{U_{Hall} - U_{Hall0}}{3,125 \frac{mV}{G}} \quad (63)$$

$$B_{Tesla} = \frac{B_{Gaus}}{10000} \quad (64)$$

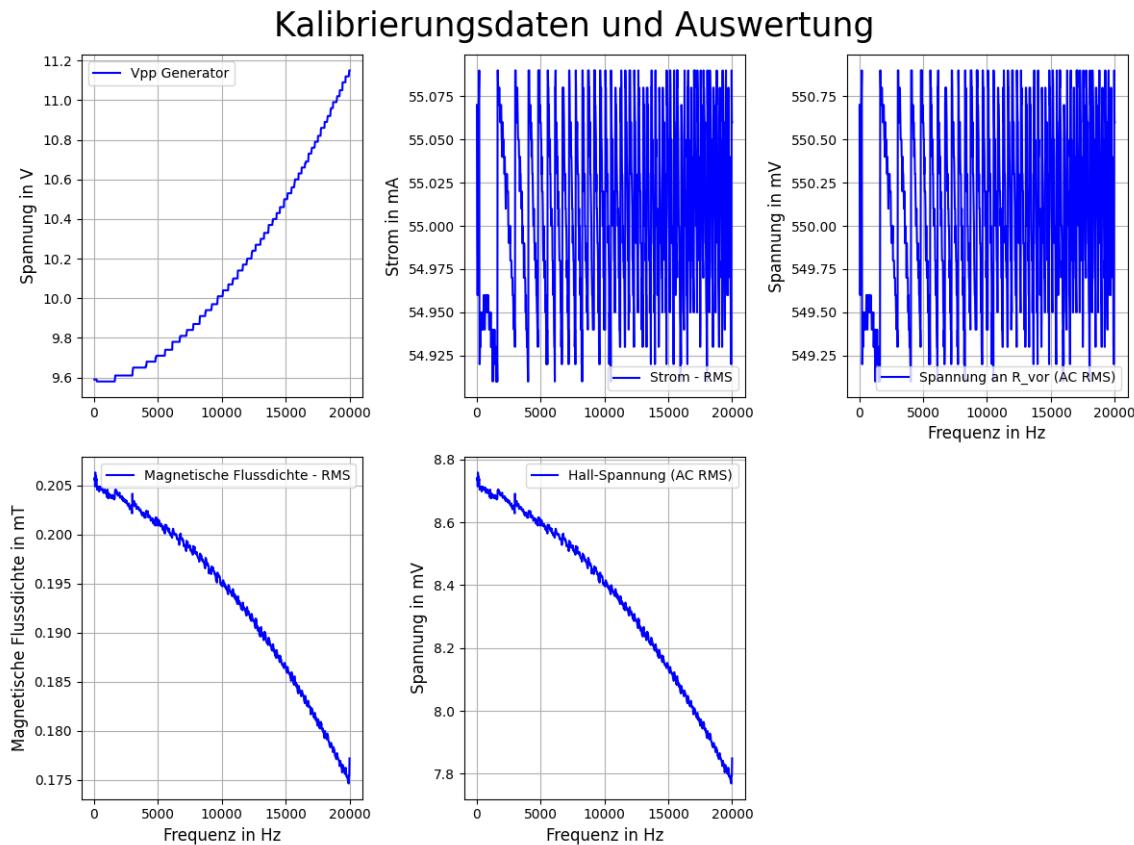


Abbildung 44: Ergebnis der Kalibrierung (Quelle: eigene Darstellung)

Im Folgenden werden die Diagramme aus der Abbildung 44 vorgestellt. Oben links kann man die Berichtigung der Wellen-Amplitude sehen. Da der Sollwert bei 550 mV lag, ist der erste Wert bei

20 kHz nicht 12 V, sondern 11,15 V (Wert aus Text-Datei bzw. Messdaten), da dieser angepasst wurde, so wie es in Kapitel 5.1.2 beschrieben wurde. In dem oberen mittleren Bild wird der Strom über der Frequenz dargestellt<sup>16</sup>. Der Strom wird hier über das Ohmsche Gesetz (Gleichung 14) berechnet. In dem oberen rechten Bild kann man die Spannung des Vorwiderstandes sehen. Auch wenn die beiden Bilder sehr verrauscht aussehen, wurde der Strom sehr konstant gehalten, eben in dem angegebenen Sollwertbereich. Es wurde versucht, mit niedrigerem Sollwertbereich und Berichtigungswert zu arbeiten, jedoch wurde der Wert mit der Zeit immer sprunghafter, wodurch die Messung mehrere Stunden dauern würde, bis hin zu gar nicht endend. In dem unteren linken Bild kann man die Magnetische Flussdichte sehen und rechts daneben die Hall-Spannung. Man kann bei den beiden Diagrammen (Hall-Spannung und Magnetische Flussdichte) sowie bei den Vorwiderstands-Größen (Strom und Spannung) die Proportionalität der Größen gut erkennen.

Die Berechnungsgleichung der Magnetischen Flussdichte kann in Gleichung 63 und 64 nachvollzogen werden. Diese Formeln werden in dem oben genannten Auswertungsprogramm ausgeführt. In der ersten Gleichung wird das Magnetfeld in Gaus berechnet und mit der zweiten dann in Tesla umgerechnet. Der Faktor  $3,125 \frac{mV}{G}$  stammt aus der Quelle [Hal] von Seite 2.<sup>17</sup> Der Hall-Sensor der in Abbildung 42 (S. 56) gezeigt wird, stammt von der Firma Honeywell und gehört der Serie SS490 an. [Vgl. [Hal] S. 2]

In der Gleichung 63 wird von der gemessenen Hall-Spannung ein Wert abgezogen. Dieser Wert ist die Hall-Spannung, die auftritt, wenn kein Magnetfeld anliegt bzw. wird hier ein natürliches (störendes) Magnetfeld gemessen. Da dieses Magnetfeld den Wert verfälscht, wird dies vom gemessenen Magnetfeld (Hall-Spannung) abgezogen. Dieser Wert muss vor jeder Messung bestimmt werden. Diese anfängliche Hall-Spannung wird über die Parameterliste an das Programm *hauptprogramm\_Kalibrierung.py* (Kapitel 5.1.2) gegeben und in der Text-Messdatei notiert. Der Wert wird händisch in die Parameterliste eingetragen und auch händisch abgelesen.

In dem unteren Diagrammen (Abbildung 44) kann man eine Frequenzabhängigkeit der Hall-Spannung und somit auch der Magnetischen Flussdichte sehen. Diese Abhängigkeit kommt durch einen Messfehler, da bei konstanten Strom die magnetische Flussdichte auch konstant ist. Dies liegt an der Proportionalität zwischen Strom und magnetischer Flussdichte, welche in Kapitel 2.4.3 erläutert wurde. Nach den in Kapitel 2.4.3 (S. 8) genannten Grundlagen zur Helmholtzspule dürfte dies bei konstantem Strom nicht der Fall sein. Im Folgenden wurden die Magnetfeld-Daten und die Frequenz genommen und weiter bearbeitet. Mit den Daten und dem Theorie-Wert der Magnetischen Flussdichte für die Helmholtzspule wird eine Funktion  $k(f)$  entwickelt. Mit dieser Funktion kann man dann die in Kapitel 5.2 bestimmten Magnetfeld-Profile für die Frequenz korrigieren. In der Theorie zur Helmholtzspule (Kapitel 2.4.3) werden die Berechnungsformeln (Gleichung 30 S. 13) näher erläutert. Im Folgenden wird der Theorie-Wert der genutzten Helmholtzspule ( $d = 20$  mm und  $N = 60$ ) berechnet.

$$B_{Theorie} = \frac{8 * 1,2566 * 10^{-6} \frac{N}{A^2} * 60 * 0,055 A}{\sqrt{125 * \frac{0,02 m}{2}}} * 1000 \frac{mT}{T} = 0,29672 mT \quad (65)$$

Der in Gleichung 65 berechnete Wert zeigt den Theorie-Wert der Helmholtzspule. In Abbildung 45 kann man die Magnetische Flussdichte der Helmholtzspule sehen. In dem Fall wurde die Magnetische Flussdichte mit einem Teslameter von der Firma LakeShore Cryotronics über eine LakeShore-Sonde (auch Hall-Sensor) gemessen. Der Wert wurde bei den Einstellungen Sinus,

<sup>16</sup>Da die x-Achse immer identisch ist, steht die x-Achsen-Bezeichnung in Abbildung 44 immer unter dem unteren Diagramm (wenn vorhanden).

<sup>17</sup>Die in der Quelle gezeigten Toleranzen wurden in der Rechnung nicht beachtet!

50 Hz, 9,6 Vpp am Oszilloskop-Generator-Ausgang (Wellengenerator) gemessen. Aus Abbildung 44 (S. 61) kann man diese Einstellung für 50 Hz erschließen. Bei den kleineren Frequenzen ist die Spitzen-Spitzen-Spannung der Amplitude ca. 9,6 V groß. Die Art der Welle kann man aus der Mess-Text-Datei auslesen. Der Kontrollmesswert liegt bei 0,2704 mT. Man kann also einen kleinen Unterschied zum theoretischen Messwert sehen. Diese könnten durch:

- Umgebungseinflüsse (Störmagnetfelder von anderen elektrischen Geräten),
- Oder der Lage des Sensors entstehen.

verursacht werden. Wenn man sich aber den Wert bei 50 Hz vom Hall-Sensor ansieht (Wert  $\approx 0,205$  mT), kann man sehen, dass die LackShore-Sonde ein höherwertiges Messgerät ist, da die Differenz zum Theorie-Wert nicht so hoch ist. Mit der gemachten Messung sollte der Hall-Sensor kalibriert werden und mit dem Theorie Wert und den Kalibrierdaten wird nun wie bereits erwähnt die Funktion  $k(f)$  entwickelt, um die Frequenzabhängigkeit zu korrigieren.



Abbildung 45: Kontrolle des Magnetfeldes der Helmholtzspule (Quelle: eigene Darstellung)

Um die Funktion  $k(f)$  zu erzeugen wird folgende Rechnung durchgeführt. Durch die Rechnung bekommt man den Abweichungswert zum Theorie-Wert bzw. den späteren Korrekturwert für die Profil-Bestimmung bei bestimmten Frequenzen.

$$k(f) = \frac{B_{Theorie}}{B_{Messdaten}(f)} \quad (66)$$

Die Bestimmung von  $k(f)$  wurde mit dem Windows-Tool Excel durchgeführt. Dafür wurden die Frequenz- und Magnetfeld-Daten in eine Textdatei gelegt und von Excel ausgelesen. Über die Option *Daten* und *Aus Text/CSV* kann man die Daten aus einer Text-Datei in Excel importieren. Mit den Daten erstellt man ein Diagramm. Über die Funktionen *Diagrammentwurf*, *Diagrammelemente hinzufügen*, *Trendlinie* und *weitere Trendlinienoptionen...* kann man nun

die Kurve erstellen. Unter den Optionen der Trendlinie muss man nur noch *Polynomisch* auswählen und man bekommt eine Polynomfunktion zweiten Grades. Die Berechnung von  $k(f)$  folgt nach Gleichung 66.

In den folgenden Abbildungen 46 und 47 wird die Trendlinie bzw. Funktion  $k(f)$  gezeigt. Die Abbildung 46 zeigt das Ergebnis aus Excel.<sup>18</sup> In der anderen Abbildung wurde die Kennlinie in Python programmiert und mit den Daten ausgegeben.

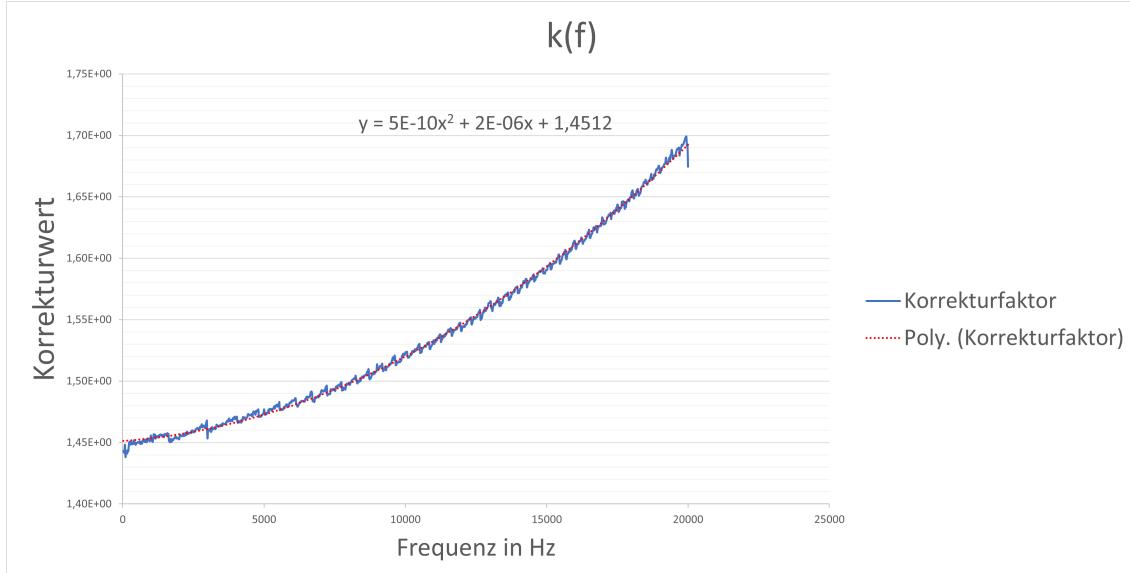


Abbildung 46: Korrekturfunktion aus Excel (Quelle: eigene Darstellung)

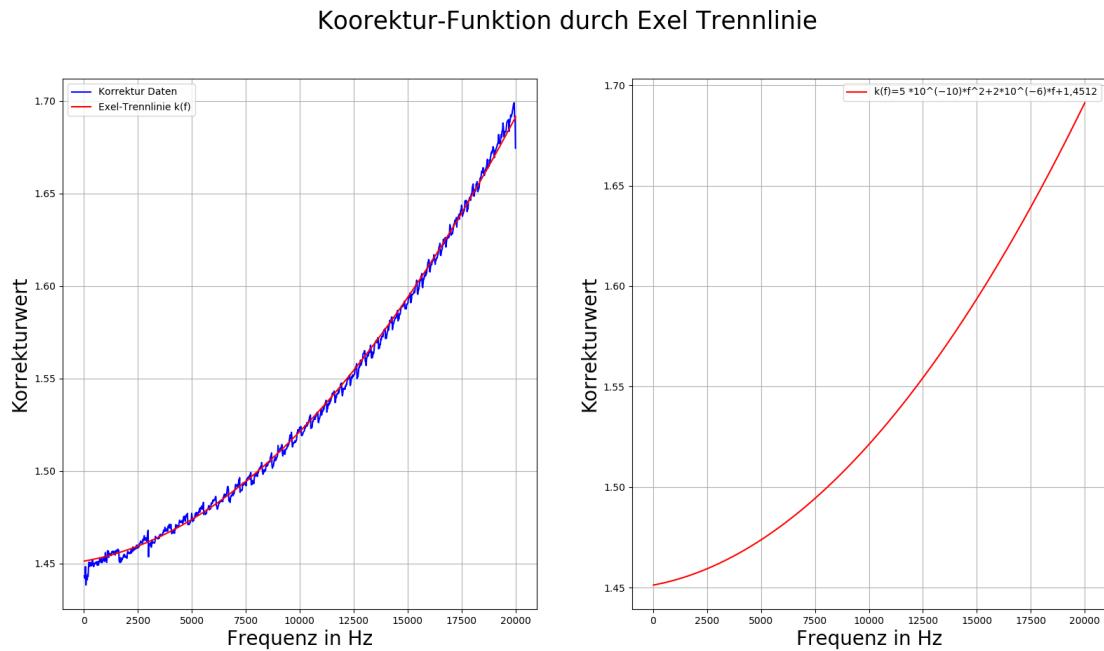


Abbildung 47: Korrekturfunktion mit Python (Quelle: eigene Darstellung)

<sup>18</sup>Um Rundungsfehler zu vermeiden, wurde der Theorie-Wert auch in Excel berechnet.

Die Funktion  $k(f)$  lautet wie folgt:

$$k(f) = 5 * 10^{-10} \frac{1}{Hz^2} * f^2 + 2 * 10^{-6} \frac{1}{Hz} * f + 1,4512 \quad (67)$$

Mit der Funktion (Gleichung 67) kann man die in Kapitel 5.2 gezeigten Magnetfeld-Profile der Test-CZ-Spule dann korrigieren. Das Experiment für die Profil-Bestimmung folgt im anschließenden Kapitel.

## 5.2 Magnetische Profilbestimmung der Spule in der Test-CZ-Anlage

In dem Kapitel werden die Messreihen für die Magnetische Flussdichte (Magnetfeld) bei bewegtem Hall-Sensor vorgestellt. Die Grundlage bzw. eine Voraussetzung für das Experiment ist die Kalibrierung des Hall-Sensors und die daraus folgende Korrekturfunktion  $k(f)$  (siehe Gleichung 67 S. 65). Wie bei den Kapiteln zu Experimenten wird zunächst der Aufbau erklärt.

### 5.2.1 Messaufbau

Der Aufbau dieses Experimentes sieht wie in der Abbildung 34 (S. 45) aus. Die einzigen Unterschiede sind die genutzten Messgeräte und der Umstand, dass eine Spule an der Stelle des Grafit-Heizers ist (siehe Abbildung 48). Auch die Versorgung wird nun anders geregelt. Die Versorgung der Spule läuft über einen Generator und einen Schwingkreis (beides in Abbildung 49 zu sehen). Des Weiteren ist die Spule wassergekühlt, heißt durch die Spule fließt Wasser.



Abbildung 48: Test-CZ-Anlage mit Spule und Hall-Sensor (Quelle: eigene Darstellung)

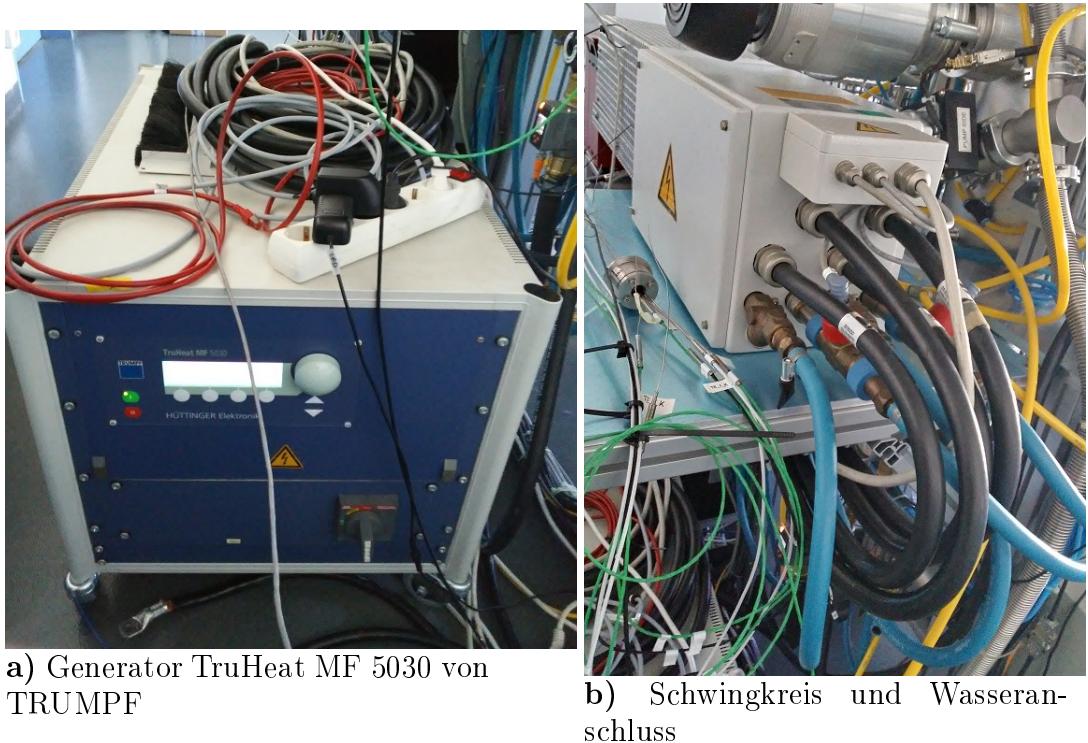


Abbildung 49: Versorgung der Spule (Quelle: eigene Darstellung)

Die komplette Anlage mit Messgeräten ist auf Abbildung 50 gezeigt.

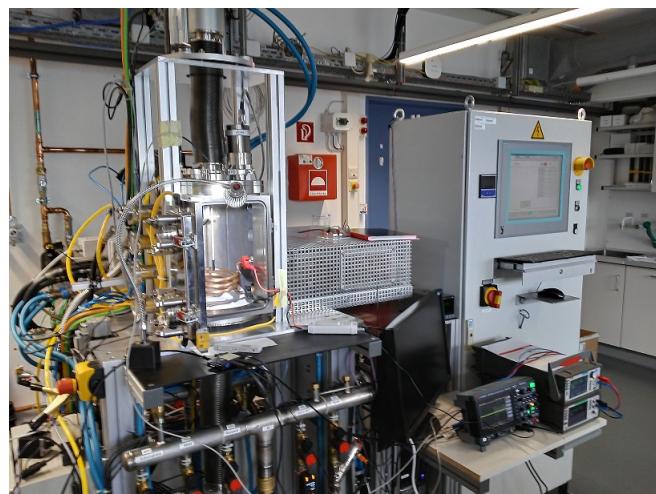


Abbildung 50: Kompletter Aufbau für die Profilbestimmung (Quelle: eigene Darstellung)

Ein Teil der Programmierung wird auch durch den Schaltschrank, der in Abbildung 50 zu sehen ist, ausgeführt. Dazu wird im nächsten Kapitel mehr gesagt. Mit diesem Schaltschrank wird die Welle bzw. der Antrieb der Spindel gesteuert. Dieser Teil der Test-CZ-Anlage wird in Abbildung 51 gezeigt.



Abbildung 51: Antrieb der Spindel (Welle) (Quelle: eigene Darstellung)

Auf den folgenden Abbildungen (52 und 53) kann man den ausgebauten Hall-Sensor der Test-CZ-Anlage sehen. In den Abbildungen wird dieser aus verschiedenen Perspektiven gezeigt.

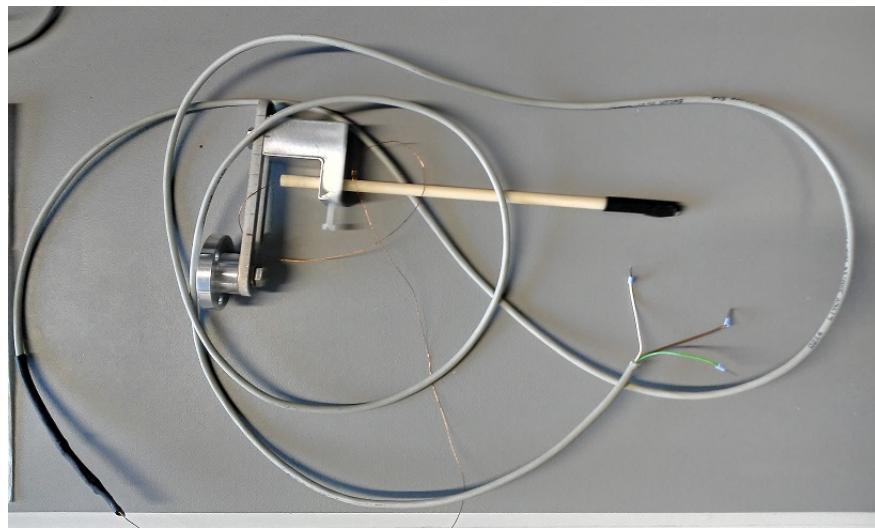


Abbildung 52: Ausgebauter Hall-Sensor (Quelle: eigene Darstellung)



a) Sensor von der Seite



b) Sensor von oben

Abbildung 53: Ausgebauter Hall-Sensor aus verschiedenen Perspektiven (Quelle: eigene Darstellung)

Des Weiteren findet man im Anhang eine schematische Zeichnung des Aufbaus. Diese wird in Abbildung 90 auf S. XXIV gezeigt. In dem Schaltplan werden der Generator, die Messgeräte, der Eurotherm-Regler sowie Spule und Hall-Sensor gezeigt.

### 5.2.2 Erklärung des Messprogramms

Das Programm für die Messungen heißt *hauptprogramm\_Profil.py* und basiert von der Struktur auf dem Programm *hauptprogramm.py* aus Kapitel 4.2.1. Das Programm arbeitet nur mit dem Multimeter DAQ6510 von Keithley. Die Besonderheiten zu dem Gerät, die in Kapitel 5.1.2 erläutert wurden, sind auch hier wieder zu finden und auch die Programmierung dieser Kommunikation ist in das Programm übernommen worden. In dem Programm werden die Befehle aus Tabelle 5 (S. 58) genutzt. Die erklärten Programme sind in Kapitel 7.2 im Anhang zu finden.

Wie schon erwähnt wurden Teile aus dem Programm *hauptprogramm.py* aus Kapitel 4.2.1 übernommen.<sup>19</sup> Im großen und ganzen wurden nur die Funktionsnamen übernommen und der Inhalt an die Magnetfeldmessung angepasst. Viele der Funktionen wie Grafik erzeugen und updaten, File erzeugen und updaten, Zeiten verarbeiten, Schnittstellen initialisieren, Parameterlisten bearbeiten und Versionen (von GitHub z.B.) auslesen sind immer identisch und werden daher von Programm zu Programm immer wieder gebraucht und daher kopiert. Die Übernahme der Struktur wurde getan da eine GUI hier angebracht war. Durch die GUI beginnt die Messung nicht sofort mit dem Start des Programmes. Die GUI besteht nur noch aus Start- und Beenden-Knopf.

<sup>19</sup>Übernommen wurden die Funktionen AutoScroll und Update\_Graph. Zudem wurde die GUI Programmierung übernommen bzw. nur die Variablen (Code-Zeilen) für Start und Beenden Knopf.

Wie auch schon die anderen Programme besitzt dieses Hauptprogramm auch eine Parameterliste. Über diese werden die Schnittstelle und der Wertmessbuffer des Multimeters übergeben und Daten für die Text-Datei.

Das Programm *hauptprogramm\_Profil.py* erstellt ein Diagramm und eine Text-Datei. In die Text-Datei kommen die erwähnten Daten aus der Parameterliste und die Messdaten. Das Diagramm dient als Live-Überwachung und muss am Ende der Messung selbst beendet werden. Nach dem Start des Programmes (Start-Knopf drücken) beginnt die Messung. Während das Programm startet muss man die Bewegung des Hall-Sensors starten. Durch dieses nebeneinander starten gibt es eine kleine Ungenauigkeit in der Messung.

Der andere Teil der Programmierung läuft über den Schaltschrank der Anlage. Diese Programmierung wurde vom IKZ gestellt und ist nicht teil der Arbeit. Für das Experiment müssen aber bestimmte Einstellungen hier getroffen werden, damit der Hall-Sensor bewegt werden kann.

Der Schaltschrank ist auf den Abbildungen 50 (S. 66) und 34 (S. 45) zu sehen. In beiden Abbildungen ist dieser auf der rechten Seite des Bildes zu sehen. Der Schaltschrank verfügt über einen Bildschirm über dem die Wellen der Anlage angesteuert werden können. In dem Experiment werden eine geradlinige und eine rotatorische Bewegung verwendet.

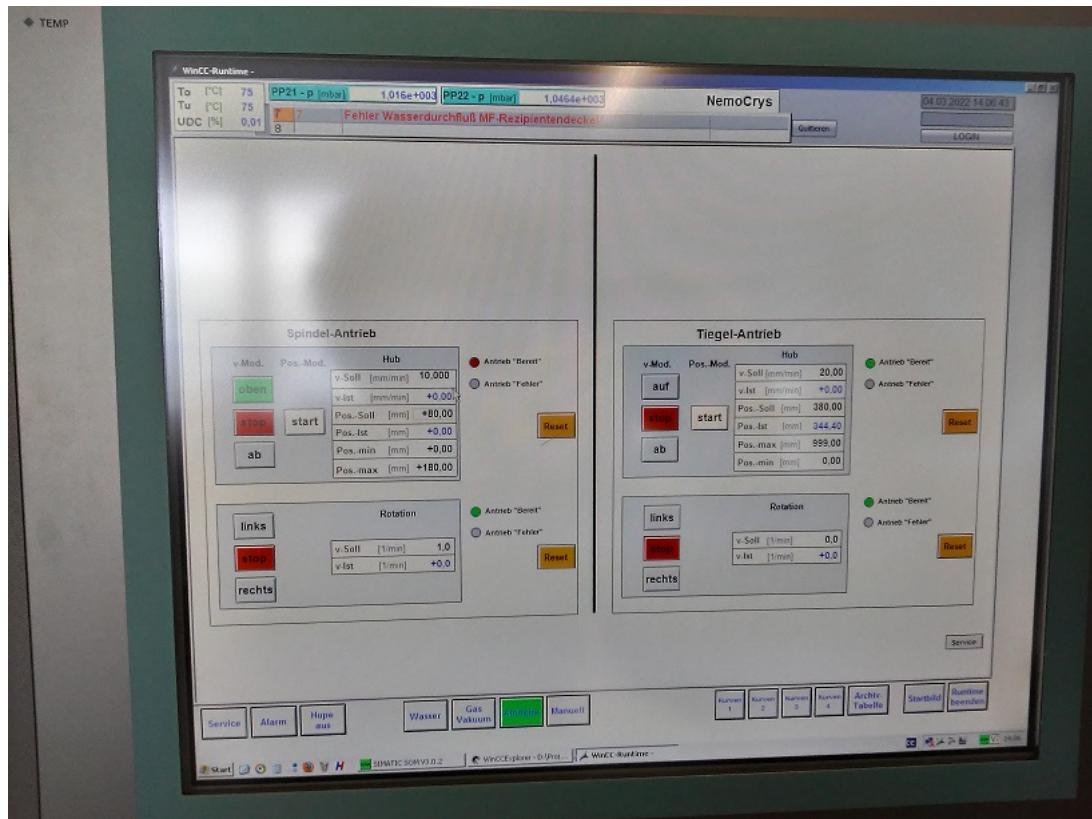


Abbildung 54: Bildschirm des Schaltschrances der Test-CZ Anlage (Quelle: eigene Darstellung)

In der Abbildung 54 kann man diesen Bildschirm sehen. Auf der linken Seite hat man die gewollten Bewegungen. Die oberen Werte sind für die Bewegung auf und ab und die unteren Werte sind für die Rotation zuständig. In Abbildung 48 (S. 65) kann man den Hall-Sensor in der Mitte der Spule sehen. Die Steuerung der geradlinigen Bewegung kann man über einen Sollwert-Position regeln. Der Sensor befindet sich bei der höchsten Position (Sollwert-Position 0 mm) 11,5 cm über dem Boden (wo in der Abbildung das Tuch liegt). Wenn man diesen Sollwert auf z.B. 80 mm stellt und Start betätigt, fährt der Sensor von 11,5 cm bis 3,5 cm über den Boden und hält

bei 3,5 cm an. Dadurch ist die Synchronisation der beiden Programme beim Beenden leichter zu erreichen, da man akustisch hört, wenn die Welle anhält. Bei der Rotation kann man nur die Drehrichtung und die Geschwindigkeit der Drehung einstellen. Die Programmierung hinter diesem Schaltschrank ist ein Teil der Test-CZ Anlage, die es schon vor dieser Arbeit gab. Um die Profile zu verstehen wurde der Bildschirm gezeigt und das Vorgehen der Steuerung erläutert.

Mit dem Schaltschrank kann man neben dem Antrieb der Spindel (im Experiment genutzt) auch den Tiegel anheben und absenken. Des Weiteren kann man den Generator auch mit dem Schaltschrank sofort ausschalten.

### 5.2.3 Ergebnis und Auswertung

In dem Experiment werden zwei verschiedene Profile bestimmt. Aus den beiden vorherigen Kapiteln 5.2.1 und 5.2.2 konnte man herauslesen, dass es einmal geradlinige Profile und zum anderen Rotationsprofile geben wird. Am Ende von Kapitel 5.2.2 wurden die Arbeit mit dem Schaltschrank und auch die Steuerung der Höhe des Sensors erläutert. Insgesamt wird der Sensor in zwei verschiedenen Positionen an der Spindel angebracht. Diese Positionen werden in der Abbildung 55 gezeigt.

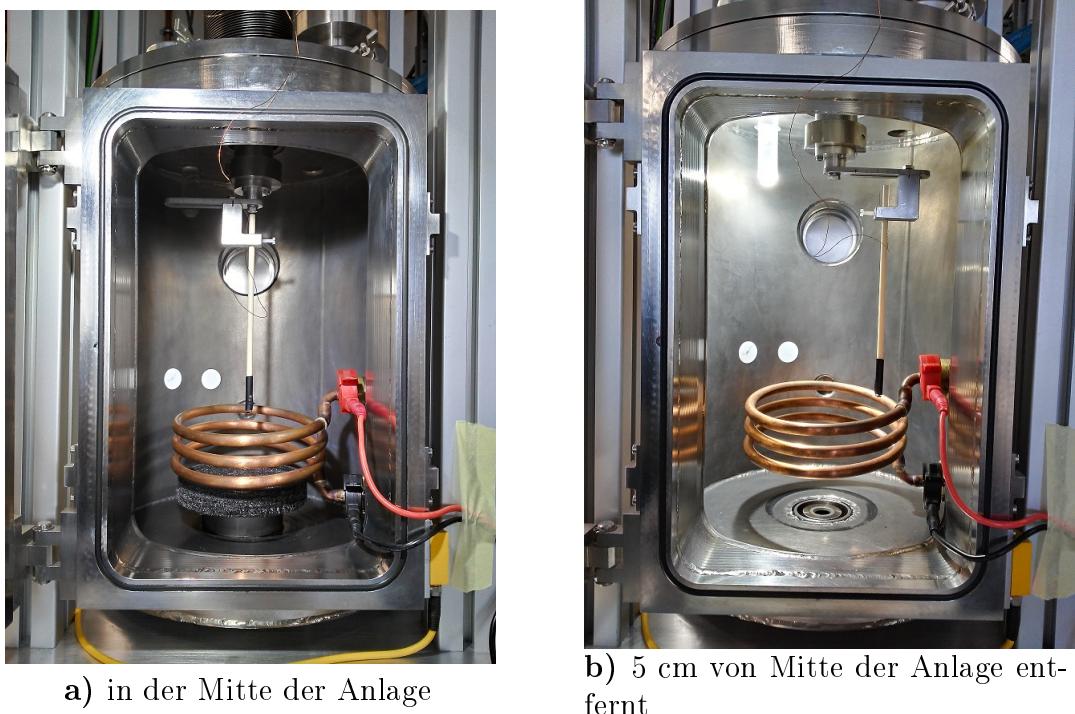


Abbildung 55: Position des Hall-Sensors an der Spindel (Quelle: eigene Darstellung)

Die Rotationsbewegungen werden alle mit der Position in Abbildung 55b durchgeführt. Im Folgenden werden die Messreihen erläutert und dann eine Zusammenstellung der Kurven aus dem Programm aus Kapitel 5.2.2 gezeigt.

Insgesamt wurden 6 geradlinige Messungen und 4 rotierende Messungen durchgeführt. Bei den geradlinigen Bewegungen werden 5 Positionen genutzt. In der Regel werden die Messungen von den in Kapitel 5.2.2 (Absatz auf S. 70) genannten 11,5 cm gestartet. Das bedeutet, dass die Messungen immer an der höchstmöglichen Position beginnen. Eine der Messungen startete aber bei den 3,5 cm.

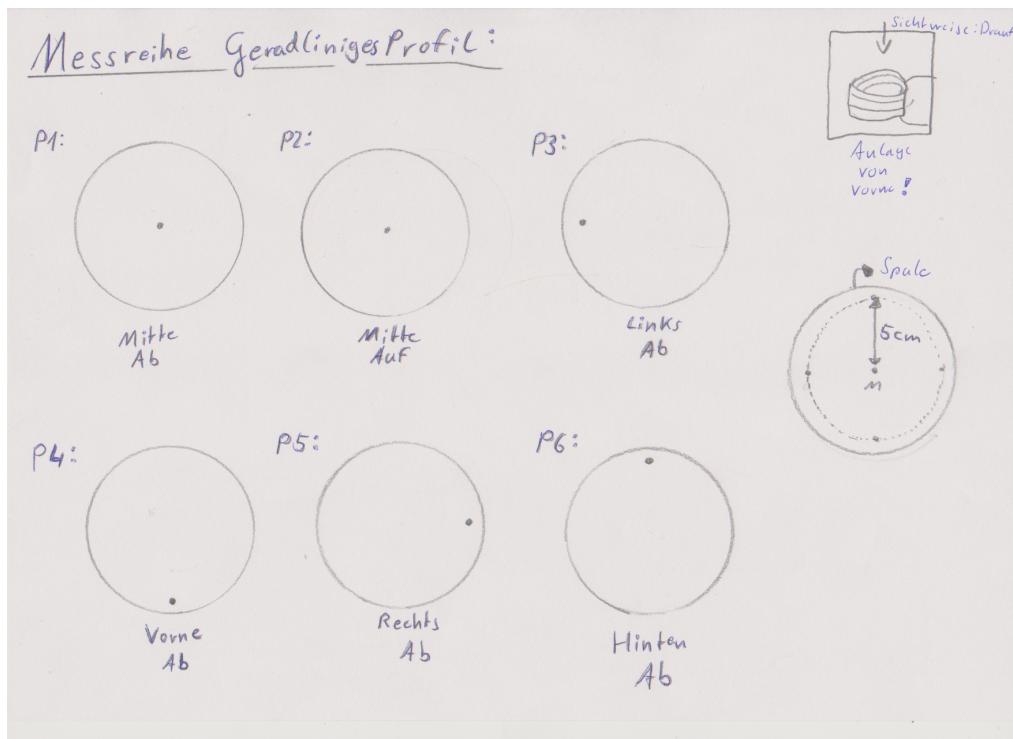


Abbildung 56: Messreihen-Skizze der Hall-Sensor-Position (geradliniges Profil) (Quelle: eigene Darstellung)

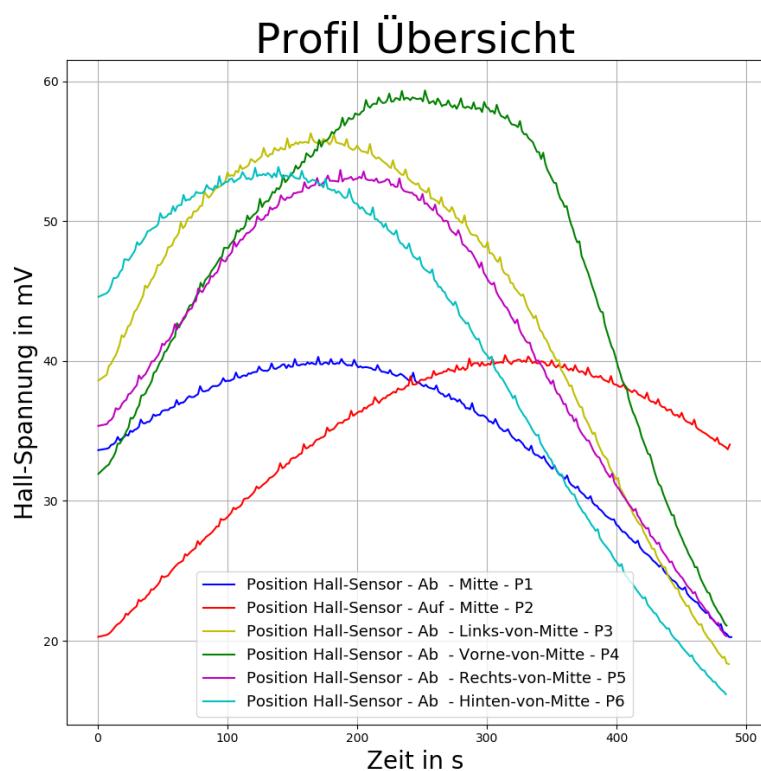


Abbildung 57: Kurvenschar des geradlinigen Profils (Quelle: eigene Darstellung)

In der Abbildung 56 wird eine Skizze der Profil-Messreihen gezeigt. In der darauffolgenden Abbildung 57 wird dann das Messergebnis des Programmes *hauptprogramm\_Profil.py* aus Kapitel 5.2.2 gezeigt.

Wenn in der Kurvenbezeichnung (Legende) **Ab** steht, ist der Hall-Sensor von 11,5 cm auf 3,5 cm gefahren und wenn die Kurvenbezeichnung **Auf** beinhaltet, ist der Hall-Sensor von 3,5 cm auf 11,5 cm gefahren. Aus dem Diagramm (Abbildung 57) kann man schon sehen, dass das größte Magnetfeld vorne bei der Spule ist. Das niedrigste Magnetfeld ist in der Mitte der Spule zu finden. Die Umrechnung folgt später in dem Kapitel. Zunächst werden die Profile für die Rotation gezeigt.

Wie schon erwähnt wurden 4 Messungen mit einer rotierenden Bewegung vorgenommen. Wie bei der geradlinigen Bewegung wurde das erste Profil auch einmal entgegengesetzt bewegt. Im Regel-Fall bewegte sich der Hall-Sensor gegen den Uhrzeigersinn (kurz CCW). Die eine Kurve dreht sich dann im Uhrzeigersinn (kurz CW). Zunächst wird die Messreihen-Skizze in Abbildung 58 gezeigt und dann wieder die Kurvenschar (Messungen mit dem Programm *hauptprogramm\_Profil.py*) in Abbildung 59.

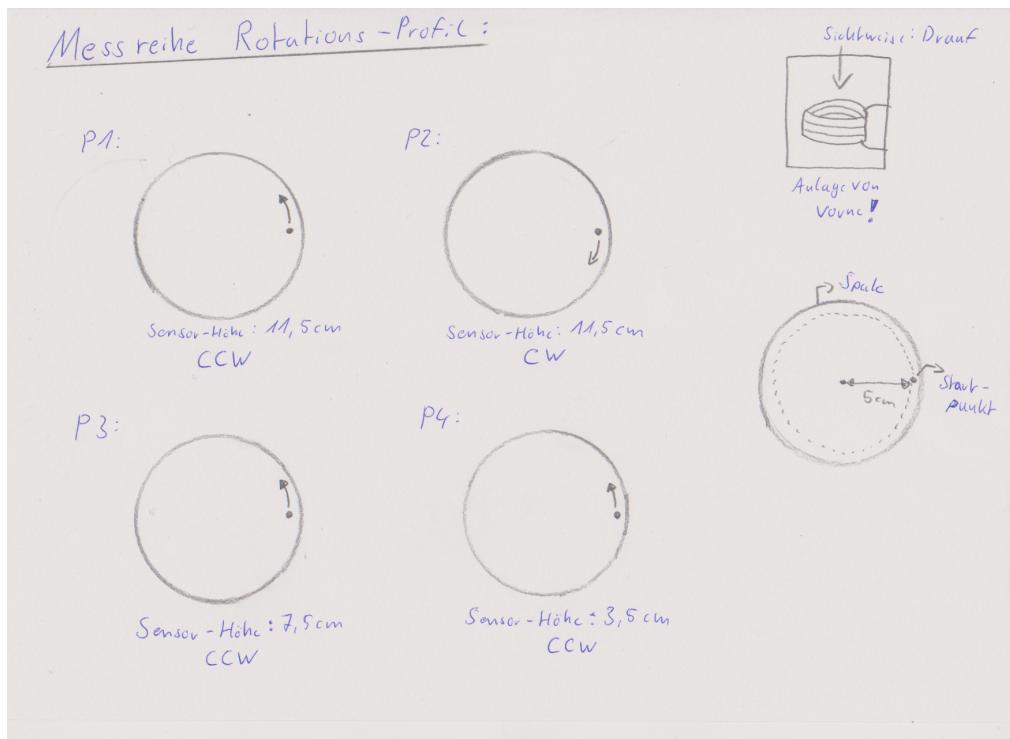


Abbildung 58: Messreihen-Skizze der Hall-Sensor-Position (rotierendes Profil) (Quelle: eigene Darstellung)

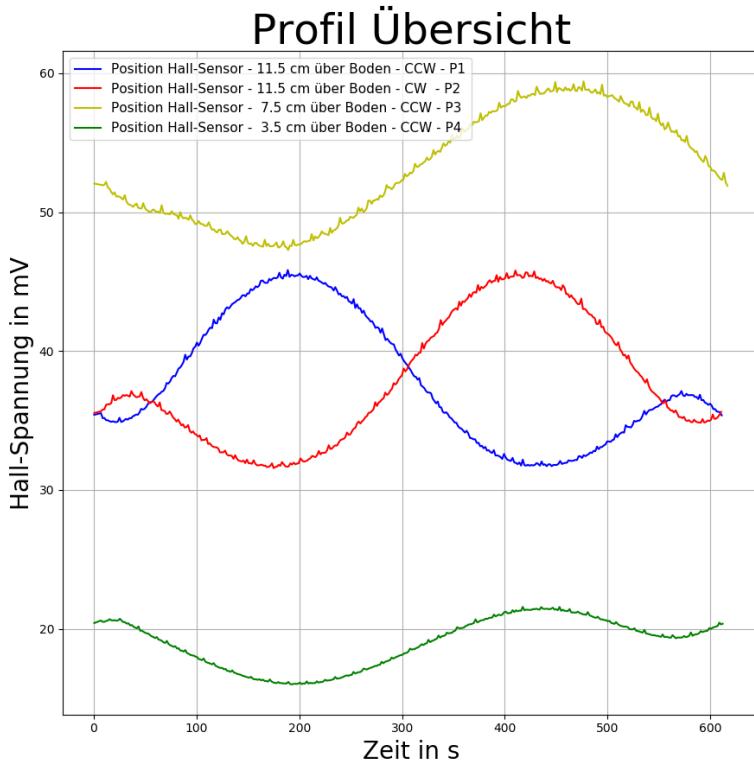


Abbildung 59: Kurvenschar des rotierenden Profils (Quelle: eigene Darstellung)

Bei den rotierenden Profilen wurde, wie man in Abbildung 58 sehen kann, immer auf der rechten Seite der Spule gestartet (von vorne drauf schauend). Danach hat man den Hall-Sensor  $360^\circ$  lang laufen lassen und das Programm danach gestoppt. Wenn sich der Sensor zu lange gedreht hat, sieht man dies in den späteren Diagrammen durch einen Winkel größer  $360^\circ$ .

Derzeitig sind die Profil-Daten nur als Funktion  $U_{Hall}(t)$  bekannt. Die Hall-Spannung wird also über die Zeit dargestellt. In den folgenden Abschnitten wird die Umrechnung zu den Funktionen  $B_z(l)$  und  $B_z(\alpha)$  erläutert.

Mit dem Programm *Profil\_Umrechnung.py* (in Kapitel 7.2 (Anhang) zu finden) werden die Profile umgerechnet. Beachten muss man, dass sowohl der Datei-Pfad als auch die Werte für Bewegungsrichtung, Frequenz, Start- und Endwerte in dem Programm per Hand eingetragen werden müssen. Im Folgenden wird der Korrekturwert für die angegebene Frequenz (vom Generator abgelesen) berechnet. Diese Korrekturkurve wird in Gleichung 67 (S. 65) im Kapitel 5.1.3 gezeigt. Danach wird die Magnetische Flussdichte berechnet und gleich mit dem Korrekturwert berichtet. Im Anschluss wird nach Auswahl ein Weg oder ein Winkel berechnet. Zur gleichen Zeit werden anhand der Entscheidung zur Umwandlungsgröße das Diagramm und die neue Text-Datei erstellt, die auch vom Programm automatisch gespeichert werden. Der Code für die Entscheidung der Umrechnung sieht wie folgt aus. Das Programm *hauptprogramm\_Profil.py* sowie ein Beispiel zur Parameterliste sind in Anhang 7.2 zu finden. Die folgenden Zeilen sind aus dem Programm *Profil\_Umrechnung.py*.

```

fre = 13900          # in Hz
bewegung = 'Rotation' # Hub oder Rotation
# Hub Einstellungen:
richtung_hub = 'ab'   # auf oder ab für die Bewegung
weg_begin = 115        # in mm
weg_Ende = 35          # in mm
# Rotationseinstellungen:
rotation_Begin = 0     # in °
richtung_rot = 'CCW'    # CW oder CCW

```

Über die im Code gezeigten Variablen kann man die Umrechnung der Zeitdaten einstellen. In den Kommentaren stehen die möglichen Strings für die Bewegung. Wenn man z.B. bei *richtung\_hub* auf auswählt, sollte man bei *weg\_beginn* auch die kleine Zahl eintragen z.B. die 35 und bei *weg\_ende* dann die 115. Bei CW sollte man als Beginn-Wert 360 als Zahl wählen. Über die Variable *bewegung* lässt sich der Bewegungstyp einstellen. Neben der Datei-Pfad-Angabe sind diese Code-Zeilen die wesentlichen Zeilen. Das Programm kann nur eine Kurve je Start berechnen. In der folgenden Tabelle werden die Ausgangsdaten für die beiden Messungen gezeigt.

Tabelle 7: Daten der Messungen (Quelle: eigene Darstellung)

	<b>Profil Geraadlinig</b>	<b>Profil Rotierend</b>
Frequenz in kHz	14	13,9
$U_{Hall0}$ in mV	2,0043	2,0301
Geschwindigkeit	$10 \frac{mm}{min}$	$\frac{1}{10min} = \frac{0,1}{1min}$
k(f)	1,5772	1,575605

Die Geschwindigkeit und die Anfangs-Hall-Spannung, die in Tabelle 7 genannt werden, werden aus dem Text-Datei-Kopf ausgelesen. Die anfängliche Hall-Spannung  $U_{Hall0}$  wird in Kapitel 5.1.3 (S. 62) erläutert. Dadurch dass die beiden Messungen an verschiedenen Tagen gemacht wurden, wurde der Wert immer neu aufgenommen.

Das Magnetische Feld wird nach Gleichung 63 und 64 (S. 61) berechnet. Des Weiteren wird jeder der Messwerte bzw. die berechneten Magnetischen Flussdichten mit dem Korrekturwert bei der Frequenz berichtet. Diese Rechnung ist in Gleichung 7 zu sehen.

$$B_{korrigiert} = B_{Tesla} * k(f) \quad (68)$$

Nachdem die Geschwindigkeit ausgelesen ist und die Art der Bewegung ausgewählt wurde, werden die Werte wie folgt berechnet.

Bei der geradlinigen Bewegung wird zunächst mit Gleichung 69 die Geschwindigkeit von mm/min in mm/s umgerechnet, da die Zeit in Sekunde angegeben ist. Danach wird je nach Bewegung (auf oder ab) entschieden, ob der Wert aus Gleichung 70 vom Startwert abgezogen oder addiert werden soll.

$$v_{Sekunde} = \frac{v_{Minute}}{60 \frac{s}{min}} \quad (69)$$

$$l_{Abzug} = v_{Sekunde} * t \quad (70)$$

Bei der rotierenden Bewegung wird die Winkelgeschwindigkeit von Umdrehung/min zunächst in °/s umgerechnet. Diese Umrechnung ist in Gleichung 71 zu sehen. Im Folgenden wird der Wert aus Gleichung 72 von dem Startwinkel abgezogen oder dazu addiert, je nach Drehrichtung. Das

t in den Gleichungen 72 und 70 soll den Messwert der Zeit darstellen. In jedem Loop (bei beiden Bewegungsarten) wird der Wert in den Gleichungen 72 und 70 neu berechnet und immer vom Startwert abgezogen!

$$\omega_{Sekunde} = \omega_{Minute} * \frac{360}{60 \frac{s}{min}} \quad (71)$$

$$\alpha_{Abzug} = \omega_{Sekunde} * t \quad (72)$$

Für die rotierenden Profile ist die eingestellte Geschwindigkeit 0,1 Umdrehungen je min. Mit der Gleichung 71 wird diese Geschwindigkeit zu 0,6° je Sekunde.

Da man nun weiß, wie die Umrechnung der Werte aus den Abbildungen 59 (S. 73) und 57 (S. 71) funktioniert, folgen die Kurvenscharen der umgerechneten Profile.

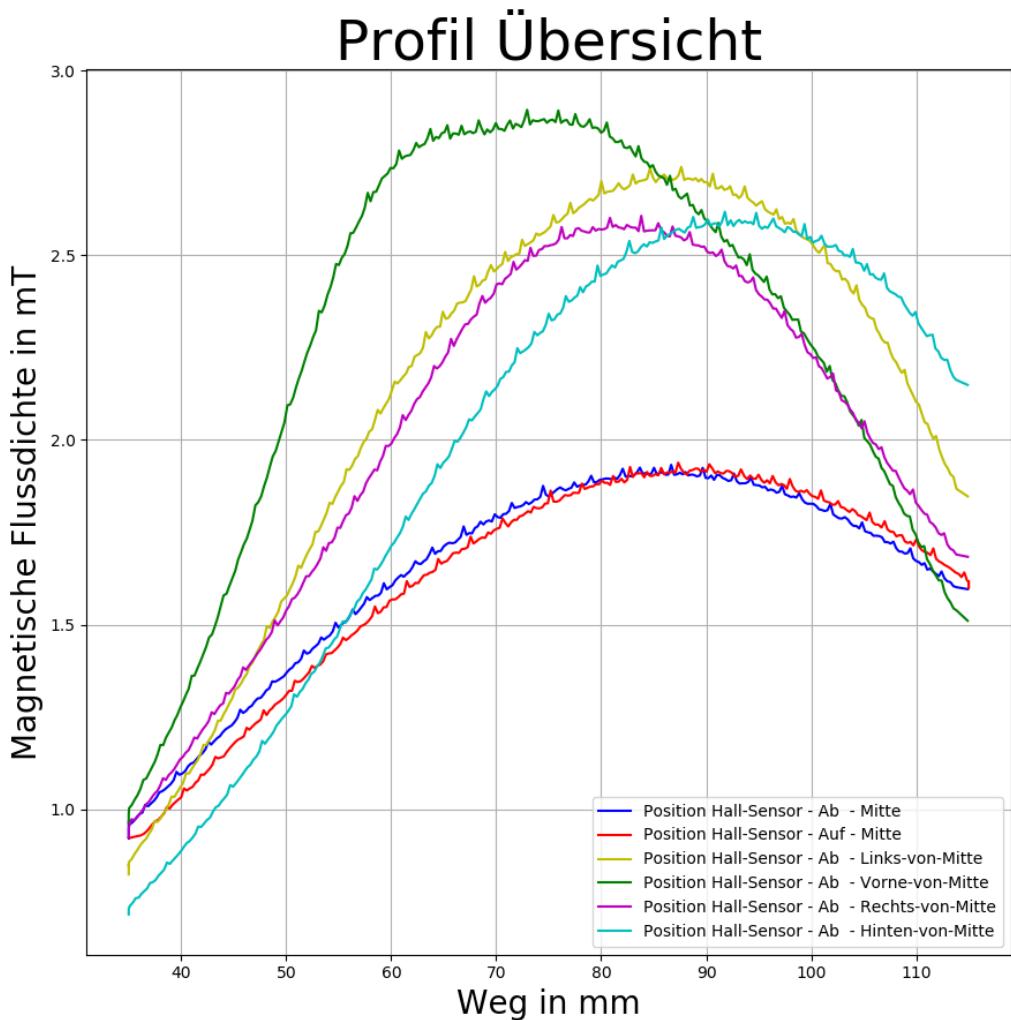


Abbildung 60: Kurvenschar der geradlinigen Profile umgerechnet zu  $B_z(l)$  (Quelle: eigene Darstellung)

## Profil Übersicht

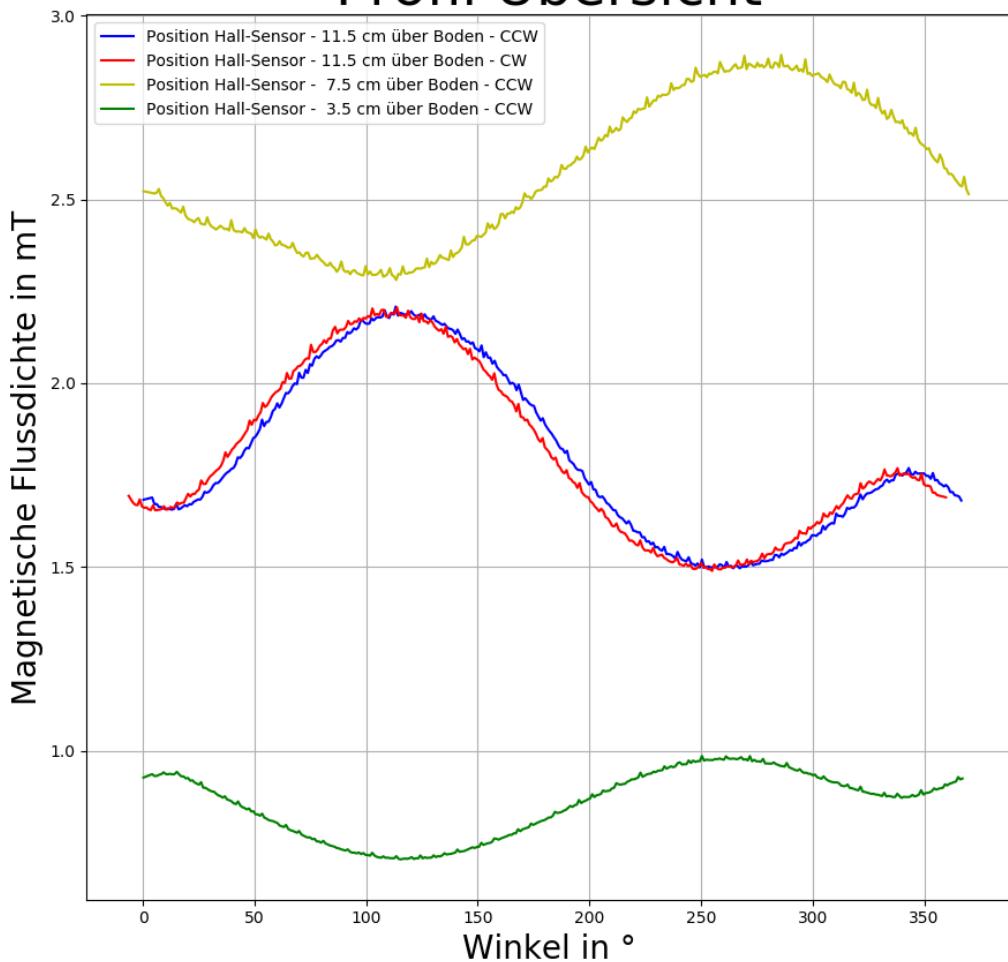


Abbildung 61: Kurvenschar der rotierenden Profile umgerechnet zu  $B_z(\alpha)$  (Quelle: eigene Darstellung)

Die bis hier beschriebenen Höhen sollen mit der Abbildung 62 verdeutlicht werden.

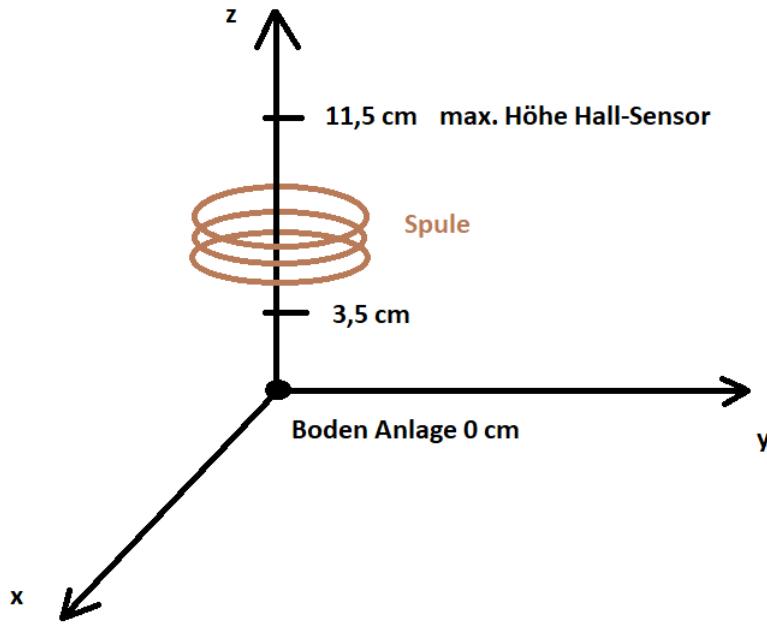


Abbildung 62: Koordinatensystem für die gefahrenen Wege (bzw. Höhe) (Quelle: eigene Darstellung)

In den beiden neuen Diagrammen (Abbildung 60 und 61) kann man zunächst erkennen, dass die Richtung der Messung keine Rolle mehr spielt. Diesen Umstand kann man an jeweils der roten und blauen Kurve sehen. Diese Kurven sind, bei denselben Einstellungen, immer das Gegenteil von der Bewegung zueinander. Bei den geradlinigen Bewegungen gibt es noch eine Besonderheit in dem Programm zur Umrechnung der Profile. Wenn der berechnete Wert unter dem Endwert landet, wird der Endwert eingesetzt, was man in der Abbildung 60 bei 35 mm an den parallelen Linien zur Ordinaten-Achse sehen kann. Diese Abflachung kommt durch den Synchronisationsversuch des Python-Programms (*hauptprogramm\_Profil.py*) und dem Start am Schaltschrank (Abbildung 54 S. 69). Zu Beginn der Messung gehen 1-2 s verloren, die dies dann im Diagramm verursachen.

Bei der rotierenden Bewegung geschieht dies durch den Synchronisationsversuch auch. Bei der Kurve gibt es aber nicht so eine Abflachung. Die Winkel werden dann größer  $360^\circ$  oder kleiner  $0^\circ$  sein. Bei beiden Werten bedeutet das, dass eine Umdrehung zu Ende war und die nächste Umdrehung anfing.

Bei beiden Kurven wurde dieser Fehler so klein wie möglich gehalten.

Aus den umgerechneten Profilen kann man sehen, dass die Spule kein gleichmäßiges Magnetfeld aufbaut. Je nach Position (z.B. vorne oder hinten) oder Höhe des Hall-Sensors ist das gemessene Magnetfeld unterschiedlich groß. Bei den geradlinigen Profilen kann man sehen, dass man bei der Einstellung des Sensors in der Mitte die kleinste Spitze für die Magnetische Flussdichte erhält. Die höchste Spitze der Magnetischen Flussdichte ist bei der Einstellung zu sehen, wo sich der Sensor vorne befindet. Zudem sind die Maximalwerte (Spitzen) immer bei anderen Wegen erreicht. Die Form der Kurve verläuft immer nach demselben Prinzip. Von 11,5 cm steigt die Magnetische Flussdichte bis zu einem Maximum, bis die Kurve dann bis 3,5 cm absinkt. Das Maximum sowie die Steigungen sind von Kurve zu Kurve anders.

Bei den rotierenden Profilen kann man dies noch besser sehen. In 11,5 cm Höhe ist die Kurve ge-

nau gespiegelt (vom Aufbau der Kurve) zu den Kurven bei den Höhen 7,5 und 3,5 cm. Das höchste Magnetfeld findet man bei 7,5 cm Höhe des Hall-Sensors. Am schwächsten ist das Magnetfeld bei 3,5 cm Höhe des Hall-Sensors. Bei der Kurve in 7,5 cm Höhe ist der Maximalwert bei ca. 270° erreicht. Da sich die Kurve CCW dreht, befand sich der Hall-Sensor bei 270° vorne.<sup>20</sup> Aus der geradlinigen Kurve (Abbildung 60 - grüne Kurve) kann man erkennen, dass im Bereich 8 bis 6 cm der maximale Wert der Kurve erreicht ist. Somit befindet sich auch der Wert von 7,5 cm in dem Bereich. Die größte Magnetische Flussdichte ist daher bei ca. 7,5 cm Höhe und der Postion vorne.

Mit den aufgenommenen Profilen wird im IKZ dann ein 3D Modell des Magnetfeldes der Spule erstellt. Mit diesem Modell will man sich das Magnetfeld verdeutlichen.

### 5.3 Leistungsmessungen an beiden Heizern

Während der Experimente aus den Kapiteln 5.2 (S. 65) und 4 (S. 34) wurde zur selben Zeit die Leistung der beiden Heizer gemessen. In den folgenden Kapiteln werden die Messungen, der Aufbau dieser und das Messprogramm für die Leistungsmessung der beiden Heizer (Grafit-Heizer und Spule) erwähnt.

#### 5.3.1 Messaufbau

In dem Kapitel werden die Messaufbauten der beiden Leistungsmessungen für die Heizer gezeigt. In beiden Fällen wird das DSOX1204G Oszilloskop von Keysight zur Messung genutzt.

##### Leistungsmessung an dem induktiven Heizer:

Die schematische Darstellung der Leistungsmessung an der Spule kann man in den Abbildungen 90 (S. XXIV) und 91 (S. XXV) sehen, die sich im Anhang befinden. Die Abbildung 90 zeigt die Messgeräte der Messung und die Abbildung 91 den Anschluss. In der folgenden Abbildung 63 kann man die beiden Messgeräte für Strom und Spannung sehen.



a) Spannungsmessung mit Differenzialtastkopf - DP10013 von *Micsig*



b) Strommessung mit einer Rogowski-Spule von Typ MA 200

Abbildung 63: Messgeräte für Strom und Spannung am Oszilloskop (Quelle: eigene Darstellung)

<sup>20</sup>Für die Positionen kann man sich die Abbildungen 56 (S. 71) und 58 (S. 72) ansehen.

Die Anschlüsse für die Spannungsmessung mit dem Gerät aus Abbildung 63a kann man in Abbildung 55 (S. 70) sehen. Bei der Rogowskispule (Abbildung 63b) wird der Anschluss an der Leitung in Abbildung 64 gezeigt. Der Käfig um die Leitung der Spule dient der Abschirmung des Magnetfeldes und dem Berührungsschutz.



Abbildung 64: Leitungsanschluss der Rogowskispule (Quelle: eigene Darstellung)

Auf dem Oszilloskop werden durch die beiden Messgeräte zwei Spannungskurven erzeugt. Die Spannungskurve wird durch das Messgerät aus Abbildung 63a um den Faktor 50 reduziert. Dieser Umstand wird durch das x50 gezeigt und auf dem Messgerät leuchtet eine Lampe dann grün, was in der Abbildung des Messgerätes zu sehen ist.

Das Messgerät zur Strommessung, siehe Abbildung 63b, misst auch eine Spannung. Die Umrechnung zu einem Strom wird über den Wahlschalter festgelegt. In den Messreihen wurde dieser auf 300 A gestellt, was einer Umrechnung von 10 mV je 1 A bedeutet.<sup>21</sup>

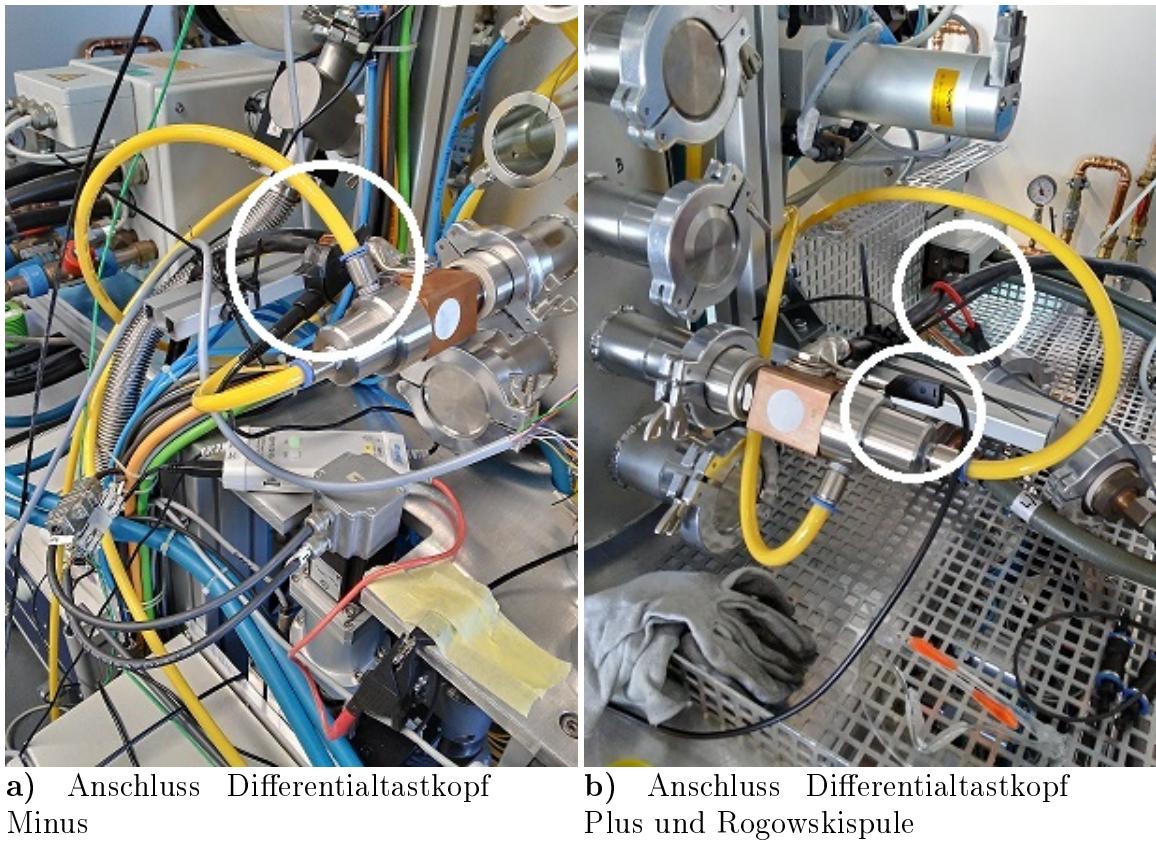
Während der Messungen wurde auch ein Grafit-Block in die Spule gestellt, um die Auswirkung des magnetischen Feldes zu sehen. Des Weiteren wird hier die Ausgangsleistung des Generators über einen Eurotherm-Regler gesteuert. Diese Regler wurden in Kapitel 3 (S. 14) erläutert. Um genau zu sein wird in dem Fall der Eurotherm-Regler 902P verwendet. Am Ende von Kapitel 3.3 werden die genutzten Eurotherm-Befehle aufgezählt. In dieser Aufzählung gibt es den Befehl *OP*. Zu diesem Befehl wird gesagt, dass er nur lesbar ist, was so nicht ganz stimmt. Wenn man den Eurotherm in den manuellen Modus<sup>22</sup> versetzt, kann man den OP Wert über Programme und auch per Hand einstellen. In dem Fall wird dies per Hand eingestellt.

#### **Leistungsmessung am Grafit-Heizer:**

Der Aufbau für die Leistungsmessung des Grafit-Heizers ist nahezu identisch mit dem der Spule. Es werden dieselben Messgeräte mit denselben Umrechnungsfaktoren sowie dasselbe Oszilloskop genutzt. Die Geräte aus Abbildung 63 wurden wie vorher an das Oszilloskop gesteckt, Spannung Kanal 1 und Strom Kanal 2. In der Abbildung 65 werden die Anschlüsse des Differentialtastkopf (Spannungsmessung) und der Rogowskispule (Strommessung) gezeigt. Eine Schaltungsskizze ist in den Abbildungen 88 und 89 zu sehen.

<sup>21</sup>Der Umrechnungsfaktor steht auf dem Messgerät!

<sup>22</sup>Gibt einen Knopf unter dem Bildschirm des Gerätes.



a) Anschluss Differentialtastkopf  
Minus

b) Anschluss Differentialtastkopf  
Plus und Rogowskispule

Abbildung 65: Anschluss der Messgeräte am Grafit-Heizer (markiert durch weißen Kreis) (Quelle: eigene Darstellung)

Wie bei den vorherigen Experimenten wird im folgenden Kapitel das Programm für dieses Experiment erläutert.

### 5.3.2 Erklärung des Messprogramms

Das Programm zur Aufnahme der Oszilloskop-Kurven heißt *hauptprogramm\_Leistung.py*. Dieses Programm ist mit einem Beispiel zur Parameterliste in Kapitel 7.2 (Anhang) zu finden. Die Kommunikation über die Schnittstelle und die Art der Befehle wurden in Kapitel 5.1.2 (S. 57) bereits erläutert. In dem Programm werden aber andere Befehle aus der Quelle [Osz] genutzt. Diese Befehle lauten (Seitenzahl aus Quelle [Osz]):

- :WAVEform:FORMAT <value> (zu finden auf S. 653 und 664),
- :WAVEform:SOURce <source> (zu finden auf S. 654 und auf den S. 674 bis 677),
- :WAVEform:DATA? (zu finden auf S. 653 und auf den S. 662 bis 663),
- :WAVEform:XINCrement? (zu finden auf S. 655 und 682).

Des Weiteren wird der Befehl \*IDN? aus der Tabelle 6 (S. 59) genutzt. Über die gerade genannten Befehle werden die Kurven auf dem Oszilloskop-Bildschirm ausgelesen. Die Werte werden in dem Programm als ASCII Zeichen ausgelesen und über die Parameterliste werden die genutzten Kanäle angegeben. Die Kanalnummer wird über den Source-Befehl übergeben und sagt dem Oszilloskop somit, welche Kurve ausgelesen werden soll. Mit dem Data-Befehl werden die y-Werte ausgelesen aus dem ausgewählten Kanal in der ausgewählten Form. Der letzte Befehl

XINCrement liest die Sampling-Rate aus bzw. den Abstand zweier y-Werte auf der x-Achse.

In dem Python-Programm werden diese Werte dann in der Klasse *Kurve* gespeichert. In dieser Klasse landen dann alle Parameter aus der Parameterliste des Programmes. Zudem wird der Name der Text-Datei hier erstellt.

Nach dem Initialisieren des Oszilloskops und der Parameterliste werden die Objekte für die Kurven der Kanäle erstellt und belegt. Das Oszilloskop hat 4 Kanäle, weshalb auch die 4 entsprechenden Farben<sup>23</sup> für die Kanäle in dem Programm vorbereitet wurden. Im Folgenden werden die Objekte in einer for-Schleife bearbeitet. Für jeden Kanal werden die ausgelesenen y-Werte in eine Text-Datei gesetzt und die Kurve erstellt. In der Text-Datei werden zudem die Sampling-Rate und eine Notiz zum Zeitpunkt der Messung gemacht. Sobald alle Kurven erstellt wurden, wird ein Diagramm mit allen Kurven erstellt.

Das Programm erstellt zum Schluss also eine Kopie des Oszilloskop-Bildschirms und speichert die Messdaten und das Bild ab. Des Weiteren kann das Programm für beide Heizer genutzt werden.

### 5.3.3 Ergebnis und Auswertung

Nachdem nun Programm und Aufbau erläutert wurden, folgen die Ergebnisse und ihre Auswertung. Die Grundlagen zu der Leistungsberechnung sind in dem Kapitel 2.4.3 (S. 8) beschrieben.

#### Leistungsmessung an dem induktiven Heizer:

Mit dem in Kapitel 5.3.2 beschriebenen Programm bekommt man die unkorrigierten Werte des Stromes und der Spannung von dem Oszilloskop. Die Umrechnungsfaktoren für die beiden Größen sind in Kapitel 5.3.1 zu finden. In der Abbildung 66 kann man einen Plot sehen, der von dem Programm erstellt wurde.

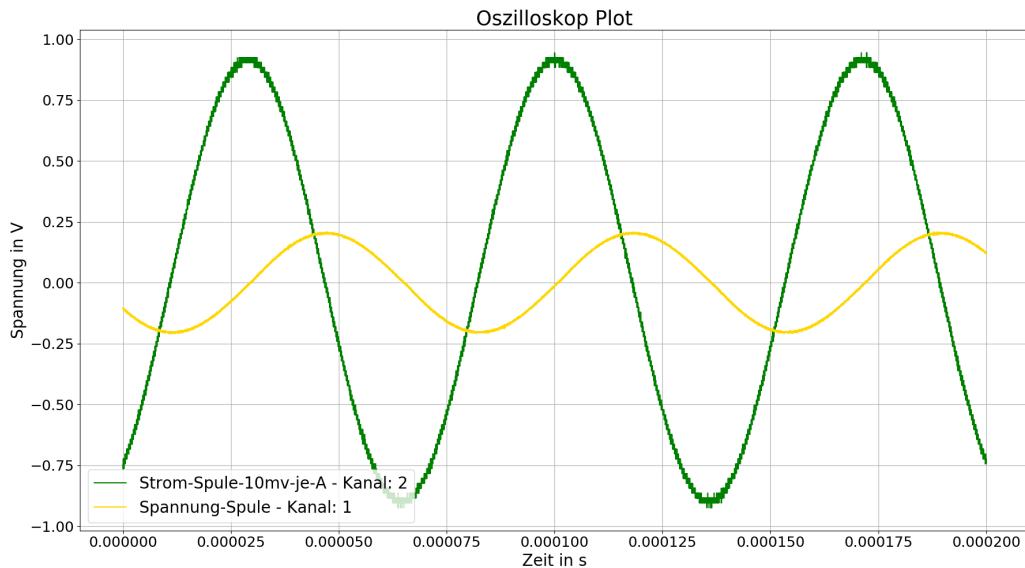


Abbildung 66: Plot des Programmes *hauptprogramm\_Leistung.py* (Quelle: eigene Darstellung)

Mit den Diagrammen aus dem Python-Programm soll nun der Phasenwinkel zwischen Strom und Spannung bestimmt werden sowie die Wirkleistung. Um dies zu erreichen wurden verschiedene Versuche unternommen. Durchgeführt wurden folgende Auswertungen:

<sup>23</sup>Kanal 1 = Gelb (im Programm Gold gewählt), Kanal 2 = Grün, Kanal 3 = Blau und Kanal 4 = Rot

- Zeichnerisches (Plot ausdrucken und mit Bleistift und Lineal bearbeiten) Bestimmen der Periodendauer und des Phasenwinkels aus den Diagrammen des Programmes *hauptprogramm\_Leistung.py*,
- Das Erstellen von Lissajous-Figur zur ungefähren Einschätzung des Phasenwinkels,
- Und das direkte Ablesen am Oszilloskops.

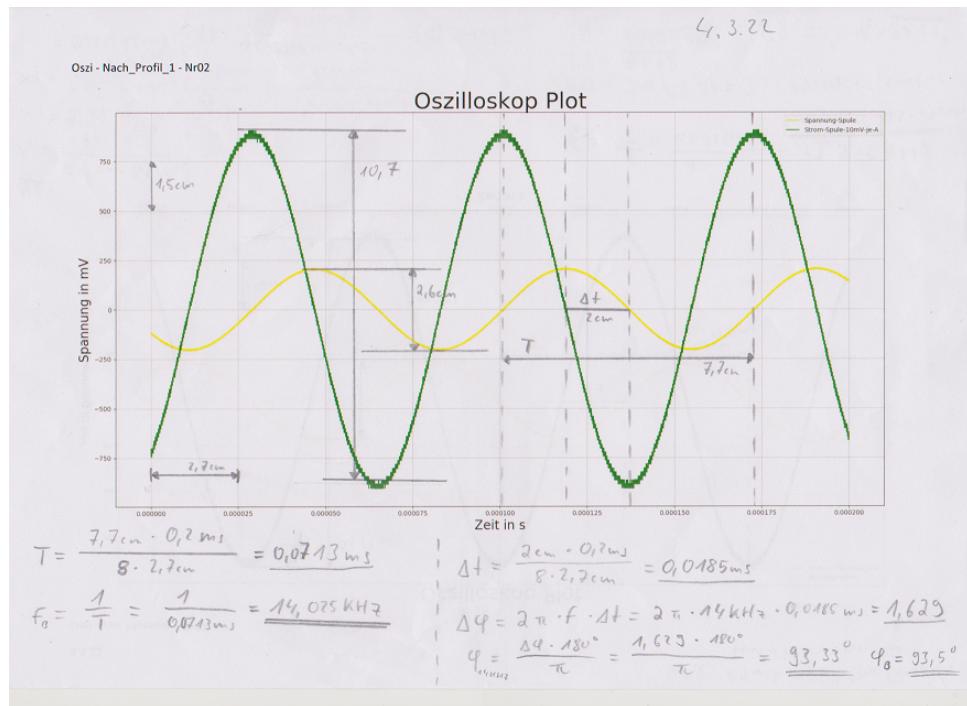


Abbildung 67: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 2 (Quelle: eigene Darstellung)

Im Folgenden werden die Ergebnisse aus den gerade genannten Auswertungs-Möglichkeiten gezeigt. Begonnen wird mit der zeichnerischen Bestimmung. Die grundlegenden Gleichungen kann man in Kapitel 2.4.3 (ab S. 8) finden. Die Abbildung 67 zeigt eine der zeichnerischen Lösungen<sup>24</sup>. In den folgenden Tabellen werden die Ergebnisse dieser zeichnerischen Lösung dargestellt. Bei der Berechnung des Phasenwinkels wurden beide Wege der Berechnung genutzt, die in den Grundlagen gezeigt werden (siehe Gleichung 23 auf S. 11 und Gleichung 26 auf S. 11). Auf den Abbildungen im Anhang befinden sich Rechnungen. In der folgenden Ergebnis-Darstellung werden nur die Verhältnisgleichungen gezeigt, die für die Berechnung genutzt wurden.

In der Abbildung 67 kann man sehen, dass die Längen von Periodendauer, Phasenverschiebungszeit und die Spitzen-Spitzen-Werte von Strom und Spannung aufgenommen bzw. gemessen wurden. Die Werte werden aus den abgelesenen Längen bestimmt. Auch die Länge und Höhe eines Kästchens ist angegeben.

In der Tabelle 8 werden die Namen der Kurven im Anhang genannt und jeder Kurve eine Nummer zugeordnet. Anhand dieser Nummer kann man in den folgenden Tabellen den Kurven ihre Werte zuordnen. Die Werte in den folgenden Tabellen beziehen sich auf die gemessenen Längen in den Abbildungen, die in der Tabelle 8 genannt werden. Die berechneten Phasenwinkel in Tabelle 9 beziehen sich auf die berechnete Frequenz. Zum Vergleich wird dort auch die am Generator eingestellte Frequenz gezeigt.

<sup>24</sup>Die anderen Lösungen sind im Anhang mit ihren Lissajous-Figur zu finden in Abbildung 92 bis 119.

Tabelle 8: Bezeichnung und Nummer des zugehörigen Plots (Quelle: eigene Darstellung)

<b>Versuchs-Nr.</b>	<b>Datum</b>	<b>Beschreibung des Versuchs</b>	<b>OP Wert [%]</b>	<b>Abbildung/Seite</b>
1.	4.3.22	Vor_Messung	5,5	92 / XXVI
2.	4.3.22	Nach_Profil_1	5,5	67 / 82
3.	4.3.22	Wieder_an_nach_Umstellung	5,5	96 / XXVIII
4.	4.3.22	Nach_Profil_6	5,5	98 / XXIX
5.	9.3.22	Vor_Messung_Rot_1	5,5	100 / XXX
6.	9.3.22	Nach_Messung_4-Ende_Messung	5,5	102 / XXXI
7.	10.3.22	Extra-Test-Mit-Grafitblock	5,5	104 / XXXII
8.	10.3.22	Extra-Test-Mit-Grafitblock	15	106 / XXXIII
9.	16.3.22	Extra-Test-Luft	5,5	108 / XXXIV
10.	16.3.22	Extra-Test-Luft	5,5	110 / XXXV
11.	16.3.22	Extra-Test-Grafit	5,5	112 / XXXVI
12.	16.3.22	Extra-Test-Grafit	5,5	114 / XXXVII
13.	16.3.22	Extra-Test-Grafit	15	116 / XXXVIII
14.	16.3.22	Extra-Test-Grafit	15	118 / XXXIX

Die Versuche Nr. 1 bis 6 wurden während der Magnetfeldprofilbestimmung (Kapitel 5.2) durchgeführt. Ihre Bezeichnungen beziehen sich darauf. Des Weiteren wird Versuchs-Nr. im Folgenden mit Nr. abgekürzt.

Tabelle 9: Berechnete und abgelesene Frequenzen, Zeiten und Winkel (Quelle: eigene Darstellung)

<b>Nr.</b>	<b><math>f_{eingestellt}</math> [kHz]</b>	<b><math>f_{berechnet}</math> [kHz]</b>	<b>T [ms]</b>	<b><math>\varphi</math> [°]</b>	<b><math>t_{ges}</math> [ms]</b>
1	14	13,845	0,072226	87,69	0,2
2	14	14,025	0,0713	93,5	0,2
3	14	14,117	0,0708	89,408	0,2
4	14	14,025	0,0713	88,825	0,2
5	13,9	13,783	0,0725	86,8959	0,2
6	13,9	13,77	0,0726	88,52	0,2
7	14	14,237	0,0702	91,53	0,2
8	14	14	0,0714	90	0,2
9	13,9	13,82	0,0724	91,64	0,2
10	13,9	13,75	0,0727	87,19	0,2
11	14	13,75	0,0727	84,375	0,2
12	14	14,286	0,07	94,29	0,1
13	14,2	14,19	0,0705	92,9	0,2
14	14,2	14,4	0,0694	92,16	0,1

Die Berechnung für Frequenz, Periodendauer und Phasenwinkel ist auf den Abbildungen (siehe Anhang und Tabelle 8) zu finden. Die letzte Spalte der Tabelle 9 zeigt die Gesamtzeit der Plots, also die abgebildete Zeit auf der x-Achse.

Tabelle 10: Ablese Werte für Strom und Spannung (Quelle: eigene Darstellung)

Nr.	Bereich Spannung [cm]	Bereich Spannung [mV]	$l_{u_{SS}}$ [cm]	Bereich Strom [cm]	Bereich Strom [mV]	$l_{i_{SS}}$ [cm]
1	2 * 1,4	500	2,4	8 * 1,4	2000	10,1
2	2 * 1,5	500	2,6	8 * 1,5	2000	10,7
3	2 * 1,45	500	2,5	8 * 1,45	2000	10,1
4	2 * 1,5	500	2,55	8 * 1,5	2000	10,1
5	2 * 1,6	500	2,75	8 * 1,6	2000	11,1
6	2 * 1,65	500	2,8	8 * 1,65	2000	11,7
7	2 * 1,6	500	2,6	8 * 1,6	2000	11,2
8	2 * 2,1	2000	2,5	6 * 2,1	6000	11,5
9	2 * 1,4	500	2,55	8 * 1,4	2000	10,5
10	2 * 1,6	500	2,8	8 * 1,6	2000	12,3
11	2 * 1,7	500	2,7	8 * 1,7	2000	12,2
12	2 * 1,7	500	2,9	8 * 1,7	2000	12,3
13	2 * 2,2	2000	2,9	8 * 2,2	6000	12,1
14	2 * 2,2	2000	2,85	6 * 2,2	6000	11,8

In der Tabelle 10 gibt es die Spalten **Bereich Spannung** und **Bereich Strom** zweimal, da damit der Bereich gemeint ist, in der die Kurve in der jeweiligen Abbildung liegt. Bei der Spalte, die die Länge dieses Bereiches angibt, ist die erste Zahl der Multiplikation die Anzahl der Kästchen, in der die Kurve liegt. In dem Fall werden hier die Längen und die Vergleichswerte für die Spitzen-Spitzen-Werte der Kurven gesucht. Durch das sichtbare Rauschen wurde bei den Kurven versucht, immer die obere Spitze und den unteren Wert der unteren Spitze (also nicht direkt die Spitze) zu nehmen. Damit sollte das Rauschen etwas ausgeglichen werden. Dieses Rauschen ist bei den Strom-Kurven deutlicher zu sehen.

Aus den Werten der Tabelle 10 kann man nun über den Dreisatz (Verhältnisgleichung) die Spitzen-Spitzen-Werte von Strom und Spannung berechnen. In Kapitel 2.4.3 (S. 8) wird die Umrechnung von Scheitel-Scheitel-Wert zu Scheitel-Wert (Gleichung 15) und Effektivwert (Gleichung 16) gezeigt.

$$i_{SSmV} = \frac{l_{i_{SS}} * I_{Bereich}}{l_{Bereich-I}} \quad (73)$$

$$i_{SS} = \frac{i_{SSmV}}{10^{\frac{mV}{A}}} \quad (74)$$

$$u_{SS} = \frac{l_{u_{SS}} * U_{Bereich}}{l_{Bereich-U}} \quad (75)$$

$$u_{SSk} = u_{SS} * k \quad (76)$$

Die Gleichungen 73 und 75 beziehen sich auf die Tabelle 10. Im Kapitel 5.3.1 (S. 78) wird erläutert, dass der Strom und die Spannung noch umgerechnet werden müssen. In Gleichung 74 steht der Umrechnungsfaktor bereits drin. Bei Gleichung 76 ist der Korrekturfaktor 50 groß.

Bevor es weitergeht, wird im Folgenden die Rechnung für die bisherigen Tabellen für die Abbildung 67 dargestellt.

$$i_{SSmV} = \frac{10,7cm * 2000mV}{8 * 1,5cm} = 1783,3mV \quad (77)$$

$$i_{SS} = \frac{1783,3mV}{10 \frac{mV}{A}} = 178,33mV \quad (78)$$

$$u_{SS} = \frac{2,6cm * 500mV}{2 * 1,5cm} = 433,33mV \quad (79)$$

$$u_{SSk} = \frac{433,33mV * 50}{1000 \frac{mV}{V}} = 21,67V \quad (80)$$

Die folgenden Gleichungen sind in Kapitel 2.4.3 beschrieben. Wie schon erwähnt wurden die Werte auf zwei verschiedene Weisen bestimmt. Am Ende wurde sich dann entschieden, in der Tabelle 11 nur die Phasenwinkel anzugeben, die auf der berechneten Frequenz beruhen.

$$T = \frac{7,65cm * 0,2ms}{8 * 2,7cm} = 0,0708ms \quad (81)$$

$$f_B = \frac{1}{0,0708ms} = 14,117kHz \quad (82)$$

$$\Delta t = \frac{1,9cm * 0,2ms}{8 * 2,7cm} = 0,0176ms \quad (83)$$

$$\Delta\varphi = 2\pi * 14,117kHz * 0,0176ms = 1,5605 \quad (84)$$

$$\varphi_B = \frac{1,5678 * 180^\circ}{\pi} = 89,407^\circ \quad (85)$$

Mit den oben genannten Gleichungen und den Leistungsgleichungen aus Kapitel 2.4.3 werden die in der folgenden Tabelle gezeigten Ströme, Spannungen und Leistungen berechnet.

Tabelle 11: Berechnung von Strom, Spannung und Leistung (Quelle: eigene Darstellung)

Nr.	$i_{SS}$ [A]	$u_{SS}$ [V]	$I_{eff}$ [A]	$U_{eff}$ [V]	S [VA]	P [W]	Q [var]
1	180,36	21,43	63,77	7,58	483,099	19,47	482,71
2	178,33	21,67	63,05	7,66	482,99	-29,49	482,09
3	174,14	21,55	61,57	7,62	469,12	4,85	469,097
4	168,33	21,25	59,51	7,51	447,14	9,17	447,04
5	173,44	21,48	61,32	7,596	465,77	25,22	465,09
6	177,27	21,21	62,68	7,5	470,04	12,14	469,88
7	175	20,31	61,87	7,18	444,34	-11,86	444,18
8	547,62	59,52	193,61	21,04	4074,55	0	4074,55
9	187,5	22,77	66,29	8,05	533,62	-15,27	533,403
10	192,19	21,875	67,95	7,73	525,51	25,76	524,88
11	179,41	19,85	63,43	7,02	445,23	43,64	443,09
12	180,88	21,32	63,95	7,54	482,13	-36,07	480,78
13	412,5	65,90	145,84	23,302	3398,44	-171,94	3394,09
14	536,36	64,77	189,63	22,9	4342,72	-219,71	4337,16

Mit der Tabelle 11 wird das Endergebnis der zeichnerischen Lösung gezeigt. Ein Problem das aufgetreten ist, ist dass einige der Winkel in Tabelle 9 über die theoretischen  $90^\circ$  gehen. Dieser Umstand ist unplausibel, da das Verhältnis zwischen den Leistungen einem rechtwinkligen Dreieck entspricht. Durch den Innenwinkelsatz und der Definition eines rechtwinkligen Dreiecks kann

kein anderer Winkel außer dem Winkel gegenüber der Hypotenuse größer gleich  $90^\circ$  sein. Diesen Umstand kann man in Abbildung 68 sehen. Wie schon erwähnt kann die Summe der drei Winkel in einem Dreieck nicht mehr als  $180^\circ$  (Innenwinkelsatz) betragen. Durch den festen Winkel von  $90^\circ$  bei rechtwinkligen Dreiecken kann die Summe der restlichen beiden Winkel nur noch  $90^\circ$  ergeben. Wenn Phi in die Nähe der  $90^\circ$  kommt, wird die Wirkleistung immer kleiner und die Blindleistung größer. Wenn man die ideale Verschiebung von  $90^\circ$  hat, hat man auch kein Dreieck mehr und auch keine Wirkleistung. In dem Fall ist die Scheinleistung gleich der Blindleistung.

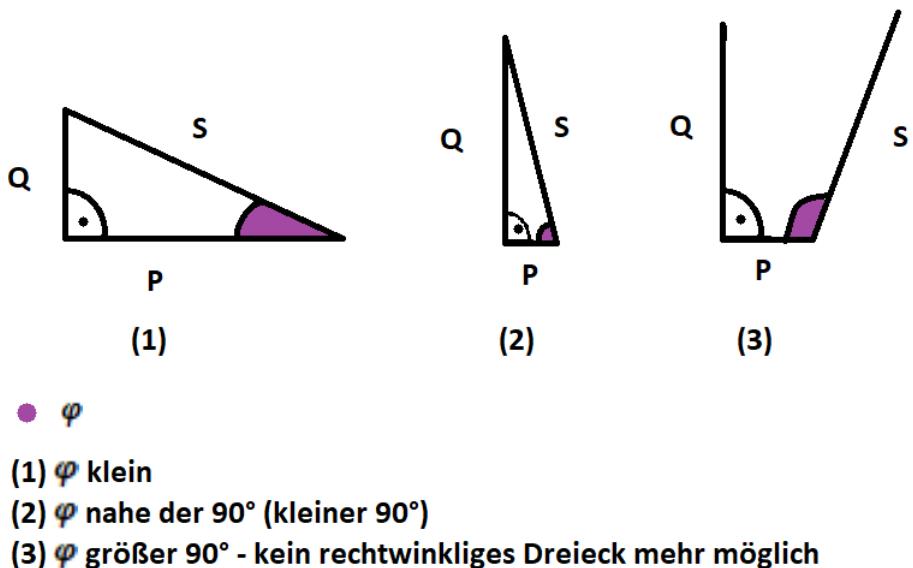


Abbildung 68: Darstellung von Phi in einem rechtwinkligen Dreieck (Quelle: eigene Darstellung)

Während der Experimente wurde die Leistung mit dem Eurotherm-Regler erhöht und ein Grafit-Block in die Spule gestellt (Versuche Nr. 7 bis 8 und 11 bis 14). Dieser Grafit-Block ist ein Suszeptor. Was das ist, wird in Kapitel 2.2 näher erläutert. Durch das Magnetfeld wurde der Grafit-Block heiß, was beim Herausnehmen zu fühlen war und bei den höheren Temperaturen mit einem Hand-Pyrometer messbar war. Dieser Aufbau ist in Abbildung 69 zu sehen. Für die Messung mit dem Pyrometer ist auf dem Grafit-Block ein Emissionsgradaufkleber<sup>25</sup>.

---

<sup>25</sup>Diese Aufkleber haben einen Emissionsgrad, den man kennt.



Abbildung 69: Test Grafit-Block in Test-CZ Spule (Quelle: eigene Darstellung)

Das Problem einer zeichnerischen Lösung ist die Ungenauigkeit beim Zeichnen und Ablesen. Des Weiteren kann man anhand der Daten aus den Tabellen sehen, dass die Messungen schon nicht genau waren. Diese Ungenauigkeit kann man an den Messungen 9 und 10, 11 und 12 und 13 und 14 sehen, da diese kurz nacheinander aufgenommen wurden<sup>26</sup>. Weiterhin wurden insgesamt drei verschiedene Einstellungen vorgenommen. Diese waren

- Messung bei Luft mit  $OP = 5,5\%$  (Versuch Nr. 1 bis 6 und 9 bis 10),
- Messung mit Grafit in Spule mit  $OP = 5,5\%$  (Versuch Nr. 7, 11 und 12),
- Messung mit Grafit in Spule mit  $OP = 15\%$  (Versuch Nr. 8, 13 und 14).

Die Schwankungen der Strom- und Spannungskurven kann man an den verschiedenen Phasenwinkeln sehen, die zu unterschiedlichen Zeiten gemessen wurden. Der Phasenwinkel schwankte während der Messungen immer um die  $90^\circ$ . Als die Leistung durch den Eurotherm-Regler erhöht wurde, wurde der Betrag der Phasenverschiebung auch größer.

Bei all den Werten in den Tabellen (9 bis 11) kann man aber sehen, dass die Werte ähnlich sind. Bei der höheren eingestellten Leistung am Eurotherm kamen für Strom, Spannung und Leistung auch größere Werte heraus. Die negativen Wirkleistungen kommen durch einen Messfehler beim Phasenwinkel.

Um den ungefähren Bereich der Phasenverschiebung zu erlangen, wurde für die Kurven die sogenannte Lissajous-Figur erstellt. Was das für Kurven sind und wie sie erstellt werden, kann man in Kapitel 2.4.3 nachlesen. Hier wird die Erstellung aber noch einmal kurz erläutert. In der Abbildung 66 kann man den Plot zweier Kanäle sehen. Um die Lissajous-Figur zu erstellen, werden die y-Daten beider Kurven genommen und in ein Diagramm gesetzt. Der eine Kanal wird zur x-Achse der Lissajous-Figur und der andere bleibt als y-Achse. Im Folgenden werden zwei dieser Lissajous-Figuren dargestellt.

---

<sup>26</sup>Bei Nr. 9 und 10 wurde dieselbe Kurve hintereinander zweimal aufgenommen und bei den anderen beiden Paaren wurde in die Kurve hineingezoomt, also eine Einstellung auf dem Oszilloskop geändert.

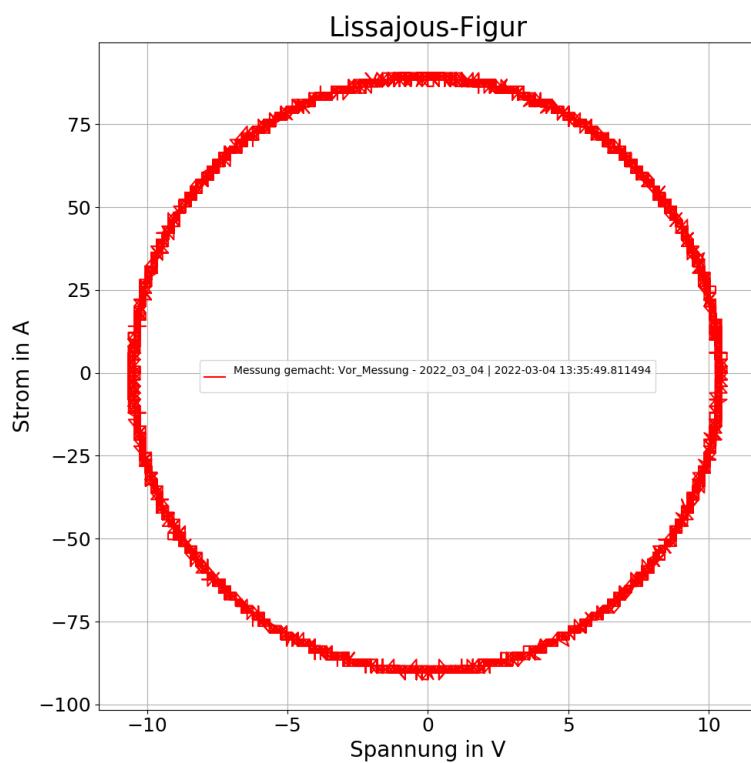


Abbildung 70: Lissajous-Figur zum Versuch Nr. 1 (Quelle: eigene Darstellung)

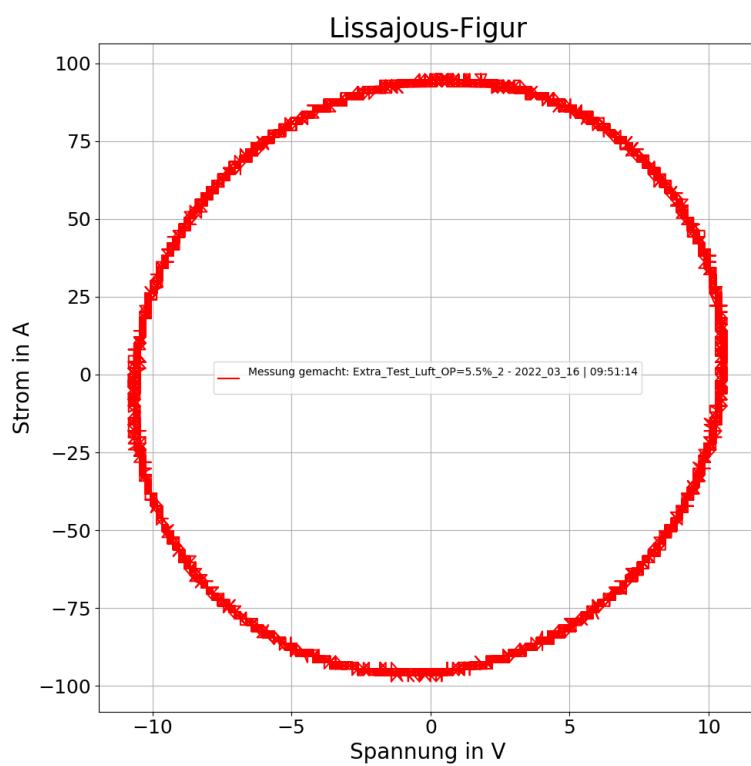


Abbildung 71: Lissajous-Figur zum Versuch Nr. 10 (Quelle: eigene Darstellung)

Beide Lissajous-Figuren (Abbildung 70 und 71) wurden bei Luft und 5,5 % Leistung am Eurotherm eingestellt ausgeführt. Die Lissajous-Figur in Abbildung 70 weist eher einen stehenden Kreis auf als die Abbildung 71. Aus der Tabelle 9 (S. 83) kann man die Phasenwinkel für die beiden Kurven ablesen. Bei Abbildung 70 (in Tabellen Nr. 1) wäre Phi gleich  $87,69^\circ$  groß und bei Abbildung 71 (in Tabellen Nr. 10) wäre Phi gleich  $87,19^\circ$  groß. Die Lissajous-Figur hilft also in den Messungen als ungefähre Einschätzung des Phasenwinkels. Alle Lissajous-Figuren sind in der Nähe eines Kreises. Nach der Quelle [Ber20] (S. 69) sind die Lissajous-Figuren bei Winkeln um  $90^\circ$  nicht mehr so aussagekräftig. Des Weiteren kann man alle Lissajous-Figuren und die dazugehörigen Kurven im Anhang finden (Abbildungen 92 bis 119 - S. XXVI bis XXXIX).

Um von den zeichnerischen Lösungen wegzukommen, wurden die Werte auch direkt vom Oszilloskop abgelesen. So wurden die Werte für Phasenverschiebungszeit, Phasenverschiebungswinkel, Amplitude (Spitze-Spitze) und Periodendauer eingestellt und abgelesen. Diese Werte wurden neben der Aufnahme der Plots vom 16.3.22 (in Tabelle 8 Nr. 9 bis 14) aufgenommen. Bei der manuellen Aufnahme der Werte konnte man das Schwanken der Werte gut sehen. Vom Oszilloskop wurden die Phasenverschiebung, Spitze-Spitze-Werte beider Kanäle, die Verzögerungszeit und die Periodendauer beider Kanäle aufgenommen. Wie bei den vorigen Messungen war Kanal 1 die Spannung und Kanal zwei der Strom als Spannung (Messgerät verschuldet). Des Weiteren wurde auch hier die Frequenz vom Generator direkt abgelesen.

Tabelle 12: Werte von Oszilloskop abgelesen (Quelle: eigene Darstellung)

Messinfo	$T_{K1}$ [ $\mu\text{s}$ ]	$T_{K2}$ [ $\mu\text{s}$ ]	$u_{SS_{K1}}$ [V]	$u_{SS_{K2}}$ [V]	f [kHz]	$\varphi$ [ $^\circ$ ]	$\Delta t$ [ $\mu\text{s}$ ]
Luft OP = 5,5% Tabelle 8 Nr. 9	71,844	71,840	0,43	1,8	13,9	-91 bis -89	-18,082 bis -17,776
Grafit OP = 5,5 % Tabelle 8 Nr. 11	71,248	71,26	0,426	1,81	14	-94 bis -91	-18,36
Grafit OP = 15 % Tabelle 8 Nr. 13	70,47	70,46	1,31	5,6	14,2	-92 bis -91	-18,1 bis -17,9

Die Messung bzw. das Ablesen dieser Werte ist auch nicht so genau gewesen, da wie schon erwähnt die Kurven immer schwankten. Wenn man die Frequenz aus den Periodendauern bestimmt, kommt man auch ungefähr auf die abgelesene Frequenz. Dies wird in Gleichung 86 mit den Werten aus der Zeile 2 aus Tabelle 12 gezeigt. Die Periodendauer der beiden Kanäle waren ungefähr identisch.

$$f = \frac{1}{T} = \frac{1}{71,844 * 10^{-6} \text{s}} \approx 13919 \text{Hz} = \underline{\underline{13,919 \text{kHz}}} \quad (86)$$

Der Phasenwinkel und die Phasenverschiebungszeit sind negativ, da der Strom der Spannung nacheilt. In Abbildung 72a kann man sehen, wie die beiden Werte am Oszilloskop bestimmt wurden. In den zeichnerischen Bestimmungen wurde dies auch so gemacht (nur ein anderer Nulldurchgang gewählt).

$$\varphi = \frac{\Delta t * 360^\circ}{T} = \frac{-18,082\mu s * 360^\circ}{71,844\mu s} \approx -90,6^\circ \quad (87)$$

In der Gleichung 87 wird die Berechnung des Phasenwinkels aus den in Tabelle 12 in Zeile 2 gezeigten Werten berechnet. Mit  $-90,6^\circ$  ist man schon nahe an den abgelesenen  $-91^\circ$ .

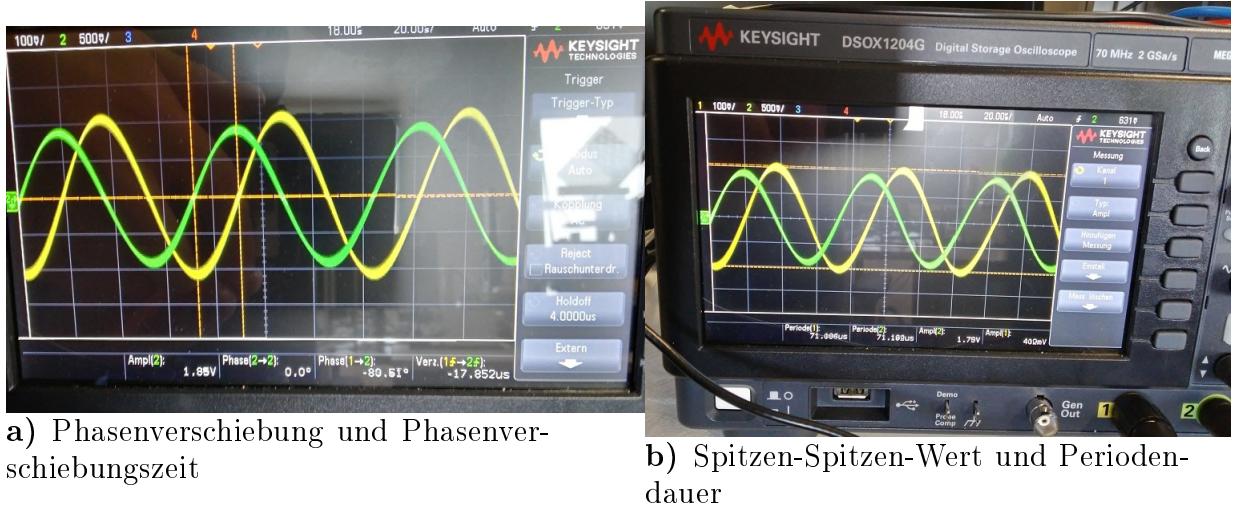


Abbildung 72: Werte-Anzeige des Oszilloskops (Quelle: eigene Darstellung)



Abbildung 73: Generator-Bildschirm (Quelle: eigene Darstellung)

In der Abbildung 72 kann man die Messung von Phasenwinkel und Phasenverschiebungszeit (Abbildung 72a) und die Messung der Amplitude (Spitze-Spitze) (Abbildung 72b) in Form der Cursor sehen. Die nicht ausgewählten Kanäle und Größen kann man im unteren Bereich des Bildschirmes anzeigen lassen. Das Oszilloskop verfügt über eine Messfunktion, über die man verschiedene Größen zu den Kanalkurven angeben kann. Die Abbildung 73 zeigt den Anzeige-Bildschirm des Generators aus Abbildung 49a.

Auch das Oszilloskop zeigt einen Phasenwinkel über  $90^\circ$  an, was nicht plausibel ist für die Messung an einer Spule. Wie in Abbildung 68 (S. 86) gezeigt kann ein weiterer Winkel von  $90^\circ$  und größer eigentlich nicht bei dem Aufbau vorzufinden sein. Das Maximale, was bei der Spule auftreten sollte, sind die  $90^\circ$  Phasenverschiebung der idealen Spule, wobei dann Schein- gleich Blindleistung ist und Wirkleistung nicht mehr vorhanden ist. Somit kann man die Leistungsmessung an der Spule für nicht plausibel einstufen. An irgendeiner Stelle scheint es ein Problem oder eine Einwirkung gegeben haben, die die Messwerte verfälscht.

Mit der Gleichung 17 (S. 10) kann man die Wirkleistung aus der Momentanleistung berechnen. Um dies zu tun wurden die Punkte einer Periode aus den Leistungskurven genommen und mit der Gleichung berechnet.

Tabelle 13: Leistungswerte aus Mittelwertbildung verglichen mit der zeichnerischen Lösung (Quelle: eigene Darstellung)

Nr.	Material in Spule	OP Wert [%]	P [W] Mittelwert	P [W] Zeichnung	$\varphi [^\circ]$ Zeichnung
1	Luft	5,5	-8,689	19,47	87,69
2	Luft	5,5	-4,008	-29,49	93,5
3	Luft	5,5	-4,013	4,85	89,408
4	Luft	5,5	-0,069	9,17	88,825
5	Luft	5,5	-4,908	25,22	86,8959
6	Luft	5,5	-5,244	12,14	88,52
7	Grafit	5,5	-19,787	-11,86	91,53
8	Grafit	15	-171,654	0	90
9	Luft	5,5	9,443	-15,27	91,64
10	Luft	5,5	29,846	25,76	87,19
11	Grafit	5,5	25,51	43,64	84,375
12	Grafit	5,5	-25,58	-36,07	94,29
13	Grafit	15	-171,757	-171,94	92,9
14	Grafit	15	-193,456	-219,71	92,16

Das Ergebnis der Berechnung der Wirkleistung über den Mittelwert der Momentanleistung  $p(t)$  wird in Tabelle 13 gezeigt. Die berechneten Werte weichen stark von den Wirkleistungswerten aus Tabelle 11 ab. Nur wenige der Werte sind nahezu identisch. So sind die Wirkleistungswerte von Nummer 10 und 13 (die Nummer in den Tabellen ist Zuordnung der Werte aus Tabelle 8) sehr ähnlich. Bei der zeichnerischen Lösung (Tabelle 11) können immer Ungenauigkeiten durch das Zeichnen entstehen. Beim Mittelwert werden die ausgewählten Punkte einfach zusammenge rechnet. Dadurch ist die Bestimmung des Ergebnisses genauer als das der zeichnerischen Lösung. Am Ende sind aber alle Leistungsbestimmungen wegen des Messfehlers beim Phasenwinkel zu ungenau gewesen.

### **Leistungsmessung am Grafit-Heizer:**

In die beiden Abbildungen 74 kann man den Oszilloskop-Plot sehen, der mit dem Programm aus Kapitel 5.3.2 aufgenommen wurde. Der seltsame Verlauf kommt durch die Phasenanschnittssteuerung der Leistungseinheit (Versorger des Grafit-Heizers - zu sehen in Abbildung 33 S. 44). Eine weitere Auffälligkeit bildet die spiegelverkehrte Anordnung von Strom- und Spannungskurven. Dieser Umstand könnte durch einen falschen Aufbau der Messung verursacht worden sein, z.B. durch das verkehrte Anschließen des Differentialtastkopfes der Spannungsmessung. Diese Anordnung der Kurven lässt einen Phasenwinkel von  $180^\circ$  vermuten.

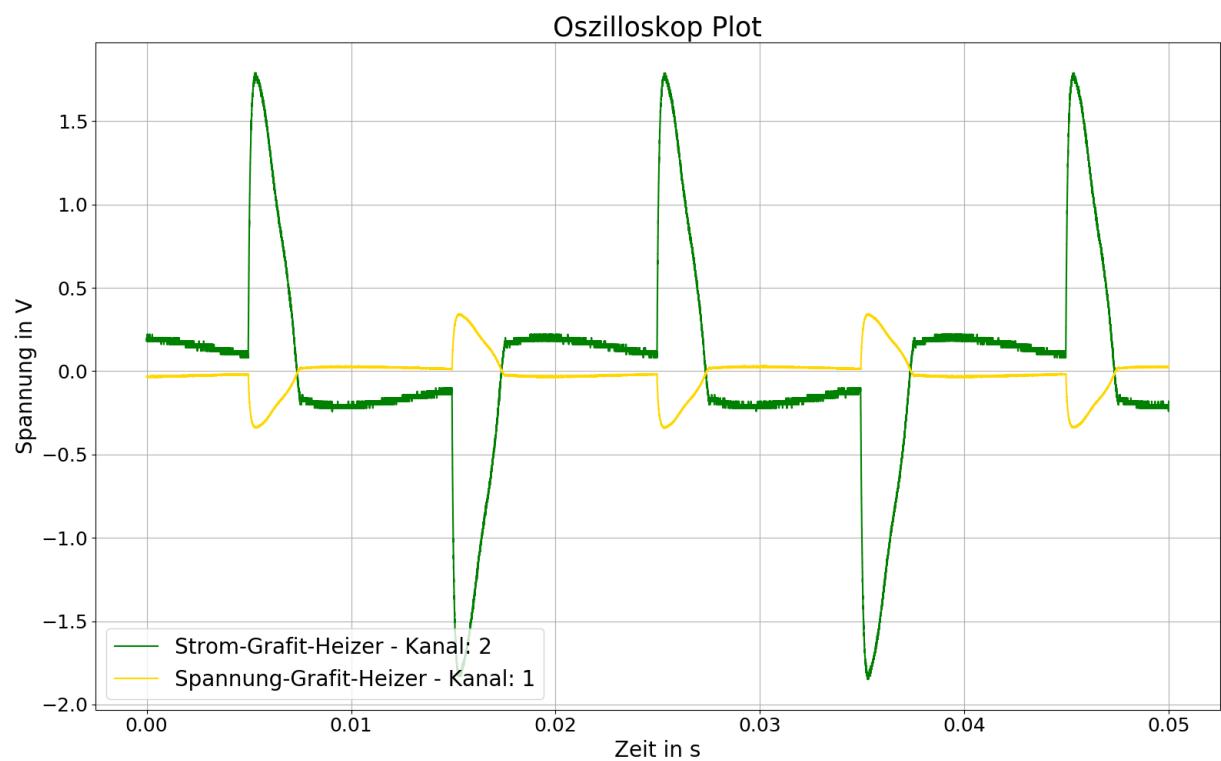


Abbildung 74: Oszilloskop-Plot der ersten Messung mit OP = 10 % (Quelle: eigene Darstellung)

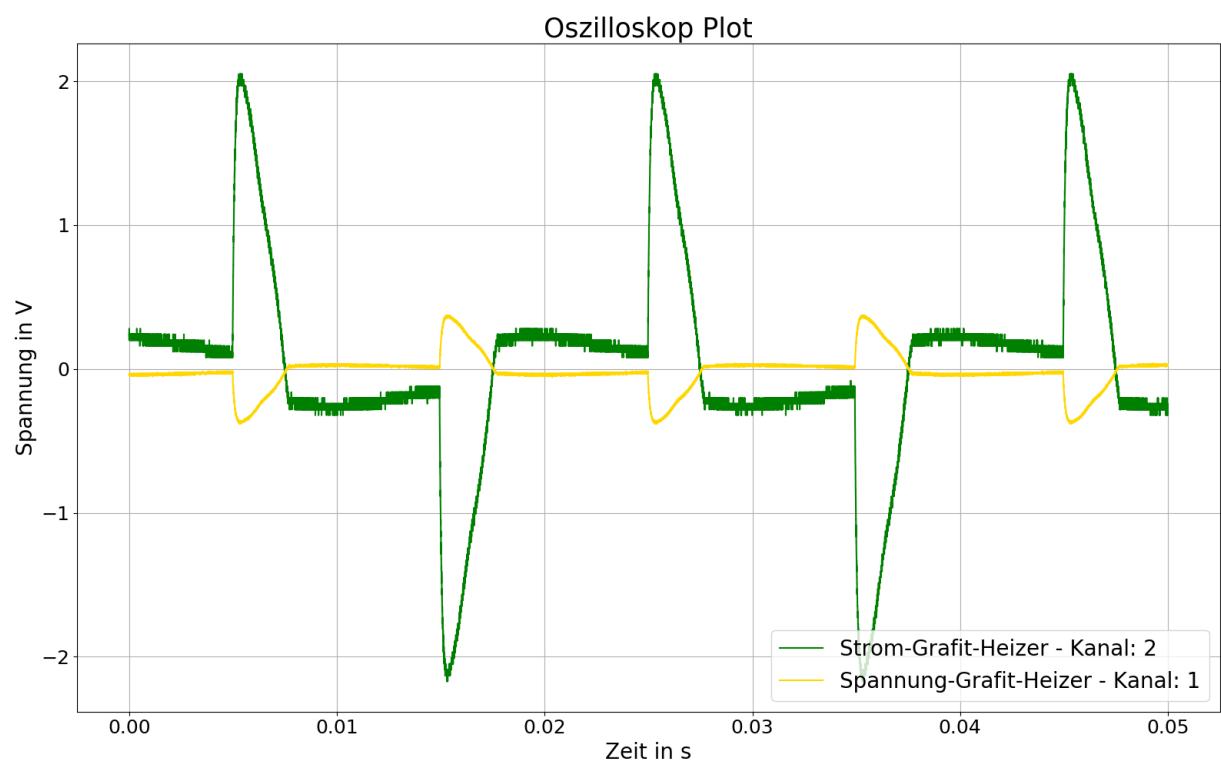


Abbildung 75: Oszilloskop-Plot der zweiten Messung mit OP = 12 % (Quelle: eigene Darstellung)

Um dies zu auszugleichen, wurde die Spannungskurve mit dem Korrekturfaktor -50 korrigiert, das Minuszeichen dient dazu, die Kurve zu spiegeln und die 50 kommt von dem Messgerät. In den Abbildungen 76 und 77 kann man die verbesserte Kurve mit der zugehörigen Momentanleistung sehen.

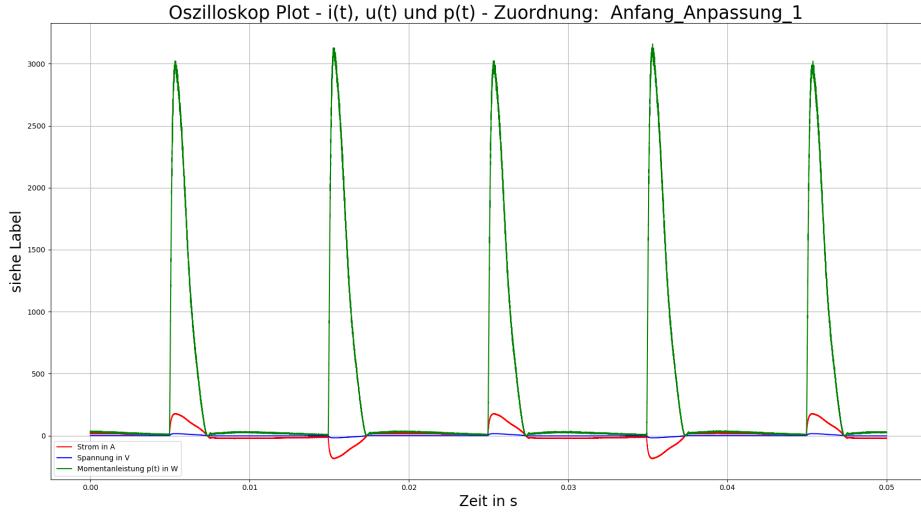


Abbildung 76: Korrigierte Kurve mit OP = 10 % mit p(t) (Quelle: eigene Darstellung)

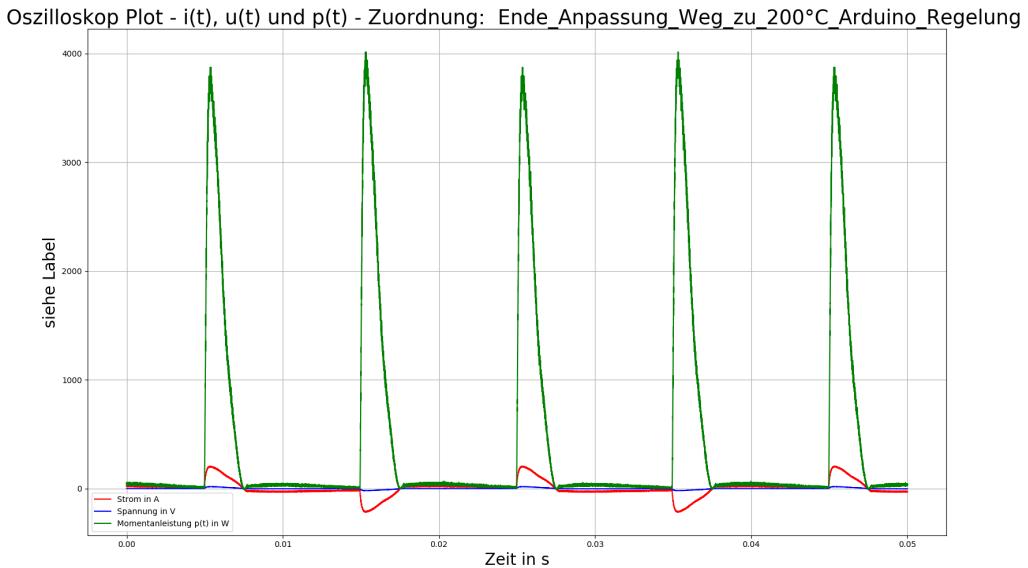


Abbildung 77: Korrigierte Kurve mit OP = 12 % mit p(t) (Quelle: eigene Darstellung)

Wie man es aus den gezeigten Diagramm erahnen kann wurden zwei Messungen neben dem Experiment aus Kapitel 4 vorgenommen. Die Messungen wurden einmal zu Beginn des Experiments und einmal gegen Ende des Experimentes durchgeführt. Durch einige Probleme während der Messung (siehe Kapitel 4.3 S. 51) wurden die Kurven bei verschiedenen Steuerspannungen aufgenommen. Die Berechnung der Leistung wurde für diese Kurven nur über die Gleichung 17 durchgeführt. Die Leistungseinheit hat eine Frequenz von 50 Hz. Mit dieser Frequenz wurden 0,02 s (Periodendauer mit Gleichung 27 S. 11 berechnet) der Leistungskurve über die Daten der Textdatei ausgelesen und in die Gleichung eingesetzt. Durch das Messprogramm bekommt man die Samplingrate der Daten und mit dieser kann man die Zeit bestimmen. Beide Messungen haben eine Samplingrate von +8.0000000E-007 s. Im Folgenden werden in den Abbildungen 78 und 79 die Ausschnitte der oberen Kurven gezeigt, um die Wirkleistung zu bestimmen. Die berechnete

Wirkleistung wird zudem in den Abbildungen gezeigt. Die Grundlage für die Berechnung wird in Kapitel 2.4.3 erläutert.

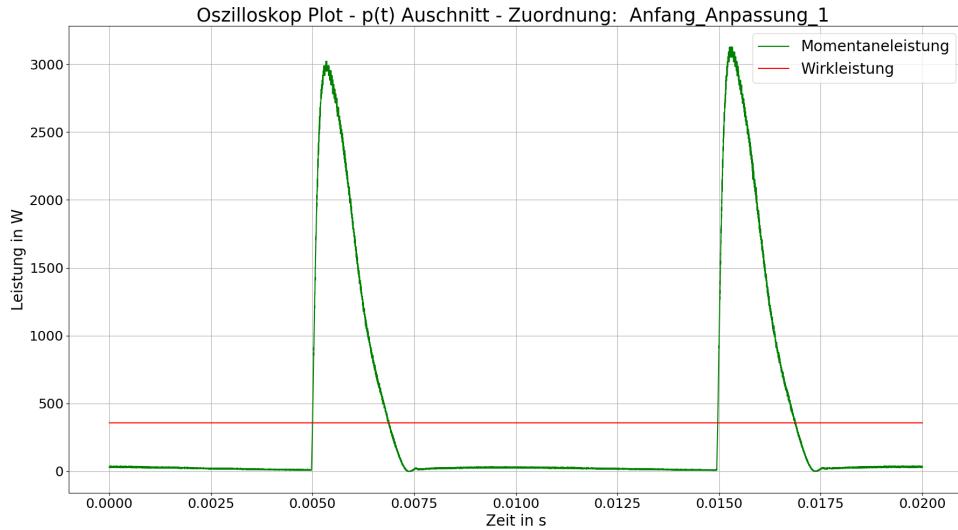


Abbildung 78: Ausschnitt von  $p(t)$  aus Abbildung 76 (Quelle: eigene Darstellung)

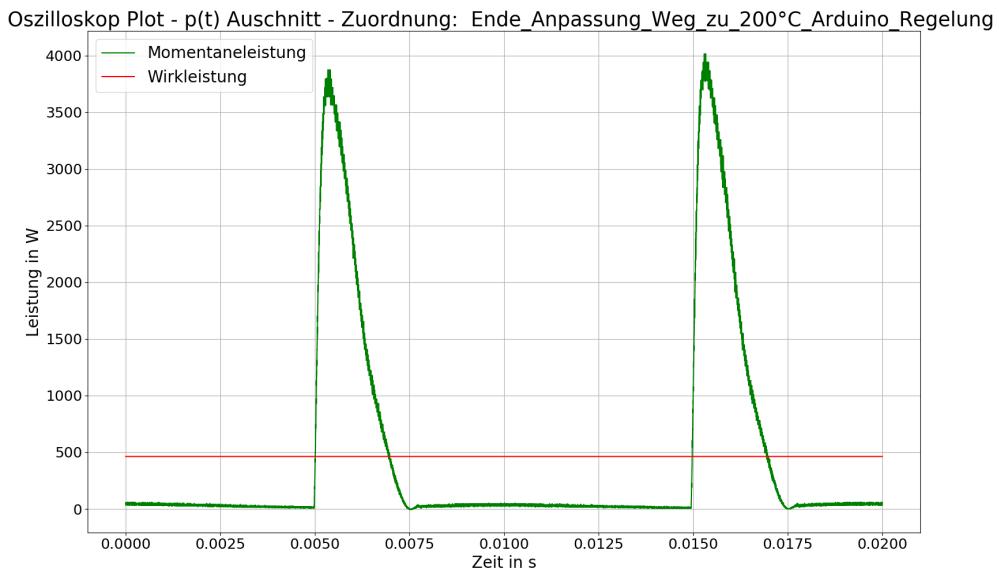


Abbildung 79: Ausschnitt von  $p(t)$  aus Abbildung 77 (Quelle: eigene Darstellung)

Die Wirkleistungen für die Messungen sind bei:

- Messung 1 (Abbildung 74) mit OP gleich 10 % 356,56 W,
- Messung 2 (Abbildung 75) mit OP gleich 12 % 462,44 W.

Ein weiterer Umstand, den man bei der Leistung beachten muss ist, dass die Messungen bei verschiedenen Temperaturen des Grafit-Heizers durchgeführt wurden. Die erste Messung wurde bei ca. 60 °C gemacht und die zweite Messung bei ca. 160 °C<sup>27</sup>. Die Messwerte sind plausibel im Vergleich zu 10 % von der 4 kW maximalen Leistung der Leistungseinheit.

<sup>27</sup> Durch die Programme aus Kapitel 4.2.1 und 5.3.2 werden Text-Dateien erstellt, die auch immer den Zeitpunkt der Messung beinhalten. Somit ist die ungefähre Angabe dieser Werte möglich.

Um die Phasenverschiebung von  $180^\circ$  noch einmal näher zu erläutern, werden hier die Lissajous-Figuren noch einmal angesprochen. In den Grundlagen 2.4.3 zur Phasenwinkelbestimmung wurde gesagt, dass bei Ohmschen Widerständen der Phasenwinkel  $0^\circ$  groß ist und dies eine Gerade in dieser Figur entstehen lässt. In der Quelle [Ber20] kann man auf S. 67 Beispiele für diese Figuren sehen. Der Phasenwinkel von  $0^\circ$  bekommt dort eine Gerade mit positiver Steigung und bei  $180^\circ$  ist diese Kurve gespiegelt, wird also eine Gerade mit negativer Steigung. Im Folgenden werden die Lissajous-Figuren der beiden Messungen in korrigierter und unkorrigierter Weise gezeigt.

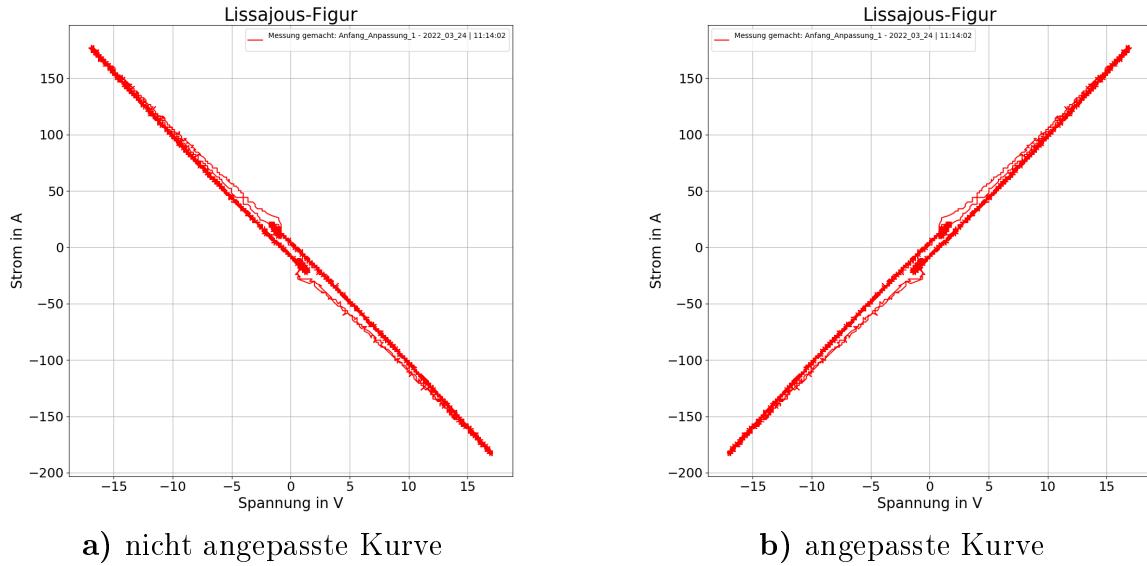


Abbildung 80: Lissajous-Figur mit  $OP = 10\%$  (Quelle: eigene Darstellung)

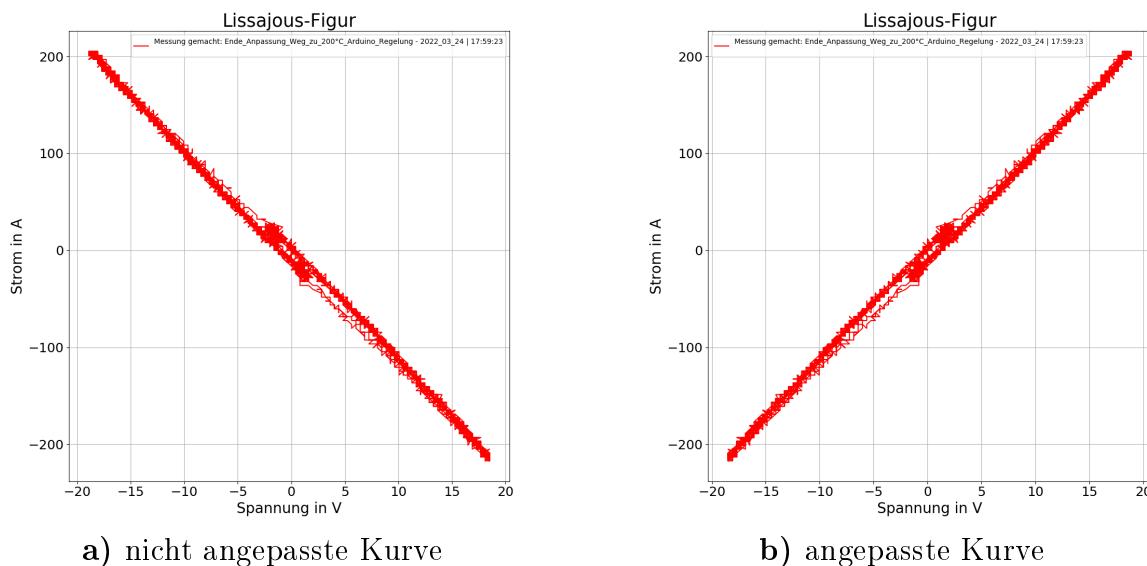


Abbildung 81: Lissajous-Figur mit  $OP = 12\%$  (Quelle: eigene Darstellung)

In beiden Fällen kann man die Geraden sehen. Da es sich hier nicht um einen idealen ohmschen Widerstand handelt, weicht der Phasenwinkel von  $0^\circ$  und  $180^\circ$  ab. Des Weiteren ist das gemessene Signal kein glattes, sondern ein verrausches. Wie man aus den umkorrigierten Kurven sehen kann, zeigt die Kurve einen ungefähren Winkel von  $180^\circ$  an. Dies belegt die Vermutung eines Messfehlers bzw. eines fehlerhaften Aufbaus.

## 6 Zusammenfassung

Die Experimente, die in der Einleitung (Kapitel 1 S. 1) genannt wurden, wurden alle ausgeführt, aber mit unterschiedlichen Erfolgen.

Das erste Experiment war die Bestimmung des Emissionsgrades und die Temperaturregelung. Bei der Temperaturregelung gab es viele Probleme, da die genutzten Eurotherm-Regler Fehler aufwiesen. Durch diese Fehler wurden zwei neue Varianten erstellt. Zum einen wurde der Eurotherm durch eine Emulation der Eurotherm-Befehle mit dem Arduino Mega Mikrocontroller völlig ersetzt und zum anderen wurde eine Kombination beider Geräte erstellt, damit man den PID-Regler des Eurotherm-Reglers nutzen konnte.

Mit dem Python-Programm *hautprogramm.py* konnte bei dem Experiment für Emissionsgradbestimmung die Temperatur nach einem Rezept gesteuert werden. Des Weiteren war die Messung des Emissionsgrades der Grafit-Probe erfolgreicher als die Messung aus dem Praktikum. Der neue Aufbau hat sich somit bewährt. Das einzige Problem machte das KW Pyrometer bei der letzten Messung. Die sonstigen Ergebnisse konnten durch die Herstellerangaben der Pyrometer als plausibel eingestuft werden.

Das Experiment zur Kalibrierung des Hall-Sensors war erfolgreich. Auch wenn die Vorbereitungen, z.B. mit dem Stromverstärker, nicht so verliefen wie gehofft, war das Experiment mit der kleinen selbst gewickelten Helmholtzspule eine gute Alternative. Aus den gemessenen Daten hat man somit eine Kalibrierungskurve für den Hall-Sensor gewonnen. Der Messfehler durch die Frequenz kann somit ausgeglichen werden.

Mit den Kalibrierungsdaten wurden im Folgenden dann die Profile der magnetischen Flussdichte des induktiven Heizers (Spule) bestimmt. Die Profile zeigten durch ihr unterschiedliches Aussehen, dass die Spule kein gleichmäßiges Magnetfeld erzeugt.

Das letzte Experiment zur Messung der Leistung der beiden Heizer war nicht so erfolgreich. Durch Ungenauigkeiten bei der Vermessung des Phasenwinkels der Spule kamen negative Wirkleistungen heraus. Dies kann nicht sein, da bei Versuchen mit einem Grafit-Block (Suszeptor) dieser auch wärmer wurde. Dies zeigt, dass eine Leistung übertragen wurde. Um die Leistung zu bestimmen, wurden hier die verschiedensten Auswertungsversuche durchgeführt, doch alle ohne Erfolg.

Bei dem Widerstandsheizer war die Messung besser, jedoch entstand durch ein falsches Anschließen der Spannungsmessung eine Verschiebung von  $180^\circ$  zwischen den Kurven. Nach der Behebung dieses Fehlers wurden plausible Leistungsmesswerte erzielt.

## Literaturverzeichnis

- [Adaa] *Arduino Bibliothek Adafruit MAX31865*. URL: [https://github.com/adafruit/Adafruit\\_MAX31865](https://github.com/adafruit/Adafruit_MAX31865) (besucht am 31.03.2022).
- [Adab] *Python Bibliothek Adafruit Blinka*. URL: [https://github.com/adafruit/Adafruit\\_Blinka](https://github.com/adafruit/Adafruit_Blinka) (besucht am 31.03.2022).
- [Adac] *Python Bibliothek Adafruit CircuitPython MAX31865*. URL: [https://github.com/adafruit/Adafruit\\_CircuitPython\\_MAX31865](https://github.com/adafruit/Adafruit_CircuitPython_MAX31865) (besucht am 31.03.2022).
- [Ard] *Arduino serial echo*. URL: <https://lucidar.me/en/arduino/arduino-serial-echo/> (besucht am 30.03.2022).
- [Ber20] Bernstein, Herbert. *Messen mit dem Oszilloskop*. 3. Auflage. Springer Vieweg, 2020. ISBN: 978-3-658-31092-9.
- [Eas] *Software EasyEDA*. URL: <https://easyeda.com/de> (besucht am 30.03.2022).
- [Erb+12] Erbrecht, Rüdiger, König, Hubert, Martin, Karlheinz, Pfeil, Wolfgang und Wörstenfeld, Willi. *Das große Tafelwerk. Sekundarstufen I und II. Neubearbeitung*. 1. Auflage. Cornelsen Verlag GmbH, 2012. ISBN: 978-3-06-020760-2.
- [Eura] *Industrie- und Prozeßregler 900 EPC Benutzer-Handbuch*. Druck Nr. HA 150 789. Fa. Eurotherm, März 1996. URL: <https://www.eurotherm.com/?wpdmld=27254> (besucht am 15.03.2022).
- [Eurb] *Series 2000 Modbus and EI-Bisynch Digital Communications Handbook*. Fa. Eurotherm, Okt. 1998. URL: <https://www.esrf.fr/computing/bliss/guides/detection/eurotherm/1pdfs/eurothermCOM.pdf> (besucht am 07.03.2022).
- [Fun21a] Funke, Vincent. *Bericht Fachpraktikum*. Hochschule für Technik und Wirtschaft (HTW), Berlin, Dez. 2021.
- [Fun21b] Funke, Vincent. *exp-T-control*. Dez. 2021. URL: <https://github.com/nemocrys/exp-T-control> (besucht am 15.03.2022).
- [Git] *Python Bibliothek USBTMC*. URL: <https://github.com/python-ivi/python-usbtmc> (besucht am 18.03.2022).
- [Gra] *Physikalische Eigenschaften/ Reinheitsgrade*. Fa. SGL Carbon Group, 2006.
- [Hal] *Solid State Hall Effect Sensor SS490 Series*. Fa. Honeywell, 2012.
- [Hel] *Magnetfeld einer HELMHOLTZ-Spule*. URL: <https://www.leifiphysik.de/elektrizitaetslehre/stroeme-magnetisches-feld/grundwissen/magnetfeld-einer-helmholtz-spule> (besucht am 17.03.2022).
- [Her18] Hering Ekbert / Schönenfelder, Gert, Herausgeber. *Sensoren in Wissenschaft und Technik*. 2. Auflage. Springer Vieweg, 2018. ISBN: 978-3-658-12562-2.
- [Log] *Python Dokumentation logging*. URL: <https://docs.python.org/3/library/logging.html> (besucht am 31.03.2022).
- [Lun10] Lunze, Jan. *Regelungstechnik 1*. 8. Auflage. Springer Vieweg, 2010. ISBN: 978-3-642-13808-9.
- [Mcp] *Arduino Bibliothek MCP4725*. URL: <https://github.com/RobTillaart/MCP4725> (besucht am 31.03.2022).
- [Mul] *Model DAQ6510 Data Acquisition and Multimeter System*. Fa. KEITHLEY, Sep. 2019. URL: [https://download.tek.com/manual/DAQ6510-901-01B\\_Sept\\_2019\\_Ref.pdf](https://download.tek.com/manual/DAQ6510-901-01B_Sept_2019_Ref.pdf) (besucht am 18.03.2022).

- [Osz] *Keysight InfiniiVision 1200 X-Series and EDUX1052A/G Oscilloscopes.* Version 02.10.0001. Programmer's Guide. Fa. KEYSIGHT, Apr. 2020. URL: <https://www.keysight.com/de/de/assets/9018-07747/programming-guides/9018-07747.pdf> (besucht am 18.03.2022).
- [Pas] Paschotta, Rüdiger. *Emissionsgrad.* URL: <https://www.energie-lexikon.info/emissionsgrad.html> (besucht am 24.03.2022).
- [Pid] *Arduino Bibliothek PID.* URL: <https://github.com/br3ttb/Arduino-PID-Library/> (besucht am 31.03.2022).
- [Put] *Software Putty.* URL: <https://www.putty.org/> (besucht am 31.03.2022).
- [Pyra] *Betriebsanleitung IMPAC Pyrometer IGA 6/23 Advanced.* Fa. LumaSense, März 2018.
- [Pyrb] *Impac Series 600 Pyrometer User Manual.* Fa. Advanced Energy, Juli 2020.
- [Pys] *Python Bibliothek pySerial.* URL: <https://pythonhosted.org/pyserial/> (besucht am 30.03.2022).
- [Ref] *Arduino Dokumentation.* URL: <https://www.arduino.cc/reference/de/> (besucht am 30.03.2022).
- [Sus] *Suszeptorwärmung.* URL: <https://www.ghinduction.com/process/suszeptorwärmung/?lang=de> (besucht am 25.03.2022).
- [Tim] *Arduino Bibliothek TimerOne.* URL: <https://github.com/PaulStoffregen/TimerOne> (besucht am 31.03.2022).
- [Tki] *Python Dokumentation tkinter.* URL: <https://docs.python.org/3/library/tk.html> (besucht am 31.03.2022).
- [Wik] *Helmholtz-Spule.* URL: <https://de.wikipedia.org/wiki/Helmholtz-Spule> (besucht am 25.03.2022).
- [Yml] *Python Dokumentation PyYAML.* URL: <https://pyyaml.org/wiki/PyYAMLDocumentation> (besucht am 31.03.2022).

## **Eidesstattliche Erklärung**

Hiermit erkläre ich durch meine Unterschrift, dass ich die vorliegende Arbeit eigenständig und ohne fremde Hilfe erstellt und andere als die angegebenen Quellen und Hilfsmittel nicht verwendet habe.

Alle Texte, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Publikationen entnommen wurden, sind als solche kenntlich gemacht.

Die Arbeit hat in gleicher oder ähnlicher Form - auch auszugsweise - noch keiner anderen Prüfungsbehörde vorgelegen und wurde bisher nicht veröffentlicht.

Berlin, den 31.03.2022

Vincent Funke

## **Danksagungen**

Ich bedanke mich bei meinen Betreuern: Prof. Dr. Anett Bailleu und Dr. Kaspars Dadzis für hilfreiche Diskussionen und wertvolle Hinweise zur Arbeit.

Einen besonderen Dank möchte ich an meine Kollegen Dr. Josef Pal, Dr. Kaspars Dadzis und Arved Enders-Seidlitz für die Unterstützung mit den Anlagen, die Bereitstellung der Geräte und die Hilfe mit Programmierung richten.

Des Weiteren möchte ich auch dem restlichen Personal des IKZ für die freundliche Behandlung und die tatkräftige Unterstützung bei Anfragen danken.

Ich möchte weiterhin Karin Funke danken für das Kontrolllesen in Hinsicht auf Rechtschreibung und Grammatik.

## 7 Anhang

### 7.1 Geräteliste

Tabelle 14: Aufzählung der genutzten Geräte (Quelle: eigene Darstellung)

Gerät	Typ	Hersteller
Raspberry Pi 400	Mikrocomputer	Raspberry Pi
Arduino Mega	Mikrocontroller	Arduino
Impac® Series 600	Pyrometer LW	Advanced Energy
IGA 6/23 - Laser	Pyrometer KW	LumaSense
Eurotherm 905S	Temperaturregler	Eurotherm
MCP4725 DAC	Digital Analog Umwandler Chip	Mikrochip
Modul MAX31865	Temperaturwandler	Adafruit
MAX3232EEPE	Schnittstellen Chip	Maxim
Multimeter	Messgerät	Fluke
TS-HUB3K	USB-Hub	Transcend
Kochplatte	-	Serverin
2450 Sourcemeter	Spannungsversorgung	Keithley
DAQ6510	Multimeter	Keithley
DSOX1204G	Oszilloskop	Keysight
F41	Teslameter	LackShore
TruHeat MF 5039	Generator	Trumpf
DP10013	Differentialtastkopf	Micsig
Typ MA 200	Rogowskispule	Chauvin Arnoux
Hall-Sensor	Magnetfeldsensor	Honeywell

## 7.2 Programme

Aufgrund der Größe und Anzahl der Programme, die für die einzelnen Kapitel erstellt wurden, wird eine Zip-Datei mit bereitgestellt.

In dieser Datei mit dem Namen **Programme\_Bachelorarbeit.zip** liegen alle Programm drinnen. In den folgenden Zeilen werden die Programme anhand ihres dazugehörigen Kapitels aufgezählt.

Programme für Kapitel 3 und 4:

Ordner 1: Emissivität\_Kapitel\_4

Inhalt:

- Beispieldateien
  - Beispiel\_config\_Parameter.yml (Beinhaltet alle Parameter)
  - Beispiel\_Messreihe\_Rezept.txt (Temperaturrezept)
- **hauptprogramm.py** (Emissivitätsmessprogramm)
- **heizer.py** (Steuerung des Reglers für den Heizer)
- **pyrometer.py** (Steuerung der Pyrometer)
- **adafruit.py** (Steuerung der Widerstandsthermometer)

Das Programm *hauptprogramm.py* sowie auch *heizer.py*, *pyrometer.py* und *adafruit.py* basieren auf der Vorarbeit von Quelle [Fun21b], Fachpraktikum am IKZ für die HTW.

Tabelle 15: Referenzen zu den genutzten Programm Bibliotheken für Ordner **Emissivität** (Quelle: eigene Darstellung)

Bibliothek	Referenz	Für Programm genutzt
serial	[Pys]	heizer.py pyrometer.py adafruit.py
tkinter	[Tki]	hauptprogramm.py
logging	[Log]	hauptprogramm.py heizer.py pyrometer.py adafruit.py
yaml	[Yml]	hauptprogramm.py
adafruit_max31865	[Adac]	adafruit.py
board, digitalio	[Adab]	adafruit.py

## Ordner 2: Emulation\_Eurotherm\_Kapitel\_3\_und\_4

Inhalt:

- arduino\_test\_eurotherm\_testpid.ino
  - **arduino\_test\_eurotherm\_testpid.ino.ino** (Eurotherm Emulation)
- Test-Programm-Autotune
  - Autotune\_Testprogram.py (Selbstoptimierung für Eurotherm)
  - Autotune\_Testprogram\_Arduino.py (Selbstoptimierung für Arduino)

In dem Programm *arduino\_test\_eurotherm\_testpid.ino.ino* wurde die eigene Arbeit mit // - **Vincent Funke - 1.2.22 - Beginn** und // - **Vincent Funke - 1.2.22 - Ende** markiert. Der Rest ist vom IKZ gestellt worden. Das Datum in der Markierung könnte auch anders sein!

Tabelle 16: Referenzen zu den genutzten Programm Bibliotheken für Ordner **Emulation Eurotherm** (Quelle: eigene Darstellung)

Bibliothek	Referenz	Für Programm genutzt
Adafruit_MAX31865	[Adaa]	arduino_test_eurotherm_testpid.ino.ino
MCP4725	[Mcp]	arduino_test_eurotherm_testpid.ino.ino
PID_v2	[Pid]	arduino_test_eurotherm_testpid.ino.ino
TimerOne	[Tim]	arduino_test_eurotherm_testpid.ino.ino

Programme für Kapitel 5:

Ordner 3: Magnetfeldmessung\_Kapitel\_5

Inhalt:

- Auswertung
  - Profil\_Umrechnung.py (Programm um die gemessenen Profile in  $B(l)$  und  $B(\alpha)$  umzuwandeln)
  - Auswertung\_Text-Datei.py (Erstellt mehrere Diagramme mit den Kalibrierungsdaten)
- Beispiel Dateien
  - Beispiel\_parameter.yml (Parameterliste für hauptprogramm\_Kalibrierung.py)
  - Beispiel\_parameter\_Leistung.yml (Parameterliste für hauptprogramm\_Leistung.py)
  - Beispiel\_parameter\_Profil.yml (Parameterliste für hauptprogramm\_Profil.py)
- **hauptprogramm\_Kalibrierung.py** (Messprogramm für die Hall-Sensor Kalibrierung)
- **hauptprogramm\_Leistung.py** (Messprogramm für die Leistung (bzw. des Oszilloskop Plots))
- **hauptprogramm\_Profil.py** (Messprogramm für die Profile der Magnetischen Flussdichte)

Tabelle 17: Referenzen zu den genutzten Programm Bibliotheken für Ordner **Magnetfeldmessung** (Quelle: eigene Darstellung)

Bibliothek	Referenz	Für Programm genutzt
serial	[Pys]	hauptprogramm_Profil.py hauptprogramm_Kalibrierung.py
tkinter	[Tki]	hauptprogramm_Profil.py
yaml	[Yml]	hauptprogramm_Profil.py hauptprogramm_Kalibrierung.py hauptprogramm_Leistung.py
usbtmc	[Git]	hauptprogramm_Kalibrierung.py hauptprogramm_Leistung.py

### 7.3 Weitere Texte

#### Auswahlkriterium für neuen Eurotherm-Regler:

In Kapitel 3.1 wurde auf ein Problem mit den genutzten Eurotherm-Reglern aufmerksam gemacht. Die Eurotherm-Regler, die bisher genutzt werden, sind schon sehr alte Modelle und sind daher bereits fehleranfälliger. Mit dem zweiten Eurotherm 905S waren die Aussichten besser, jedoch konnte der 10 V Ausgang für die geplante Leistungsregelung nicht eingestellt werden. An dem Ausgang wurden immer 24 V gemessen. Zum anderen braucht das Gerät lange zum Einpendeln der Temperatur, die gemessen werden soll.

Mit den ständig auftretenden Fehlern wurde beschlossen, ein neueres Modell zu kaufen. Eine Rolle spielen die Auswahlkriterien

1. Kommunikations-Protokoll - EI-Bisynch und Mnemonic Befehle,
2. Verfügbare Schnittstelle - RS232 bzw. RS422/485,
3. PID-Regler - einstellbar (Programm und Hand) + AutoTune (Selbstoptierung),
4. 10 V/20 mA Ausgang muss vorhanden sein oder Relais-Steuerung,
5. Rampensteuerung,
6. Anschluss für Thermoelemente und Widerstandsthermometer,
7. Sensor in Konfiguration einstellbar,
8. Spannungsversorgung - 24 V DC/AC, 85-264 V AC,
9. ON/OFF-Steuerung.



Abbildung 82: Neuer Eurotherm (Quelle: eigene Darstellung)

Am Ende wurde vom IKZ das Modell 3504 ausgewählt und bestellt. Dieses ist in Abbildung 82 zu sehen. Für diese Arbeit wird weiterhin der Eurotherm 905S genommen, weil das neue Gerät innerhalb der Arbeit nicht mehr in Betrieb genommen werden konnte.

## 7.4 Bilder

In dem Kapitel zum Anhang werden spezielle Bilder gezeigt, die für die Darstellung der Experimente noch wichtig sind.

In den Abbildungen 83 bis 89 (S. XVII - XXIII) werden Schaltpläne, die mit EasyEda angefertigt wurden, zu den Kapiteln 3 und 4 gezeigt. Die Bilder zeigen die einzelnen Abteilungen des gesamten Messaufbaus für die Emissionsgradmessung. Des weiteren sind dort alle drei Varianten des Heizers, die in Kapitel 3 genannt wurden, gezeigt. Auf den Schaltplänen sind Übergänge zu anderen Sheets gezeigt. Die Nummer eines Sheets steht in der unteren rechten Ecke der Abbildung. Bei Sheet 1 steht dort z.B. **Sheet: 1/7**.

Die Abbildungen 90 (S. XXIV) und 91 (S. XXV) zeigen den Aufbau für Kapitel 5.2 (S. 65) und 5.3 (S. 78) (Leistungsmessung Spule).

Die Abbildungen 92 bis 119 zeigen die zeichnerischen Lösungen für die Leistungsbestimmung (Spitzen-Spitzen Werte von Strom und Spannung, Phasenverschiebung und Periodendauer) der Spule für Kapitel 5.3.3 und die dazugehörige Lissajous-Figur. Des weiteren sind auf den Zeichnungen die Rechnungen einiger Werte. Nach einer zeichnerischen Lösung folgt immer die Lissajous-Figur zur Kurve.

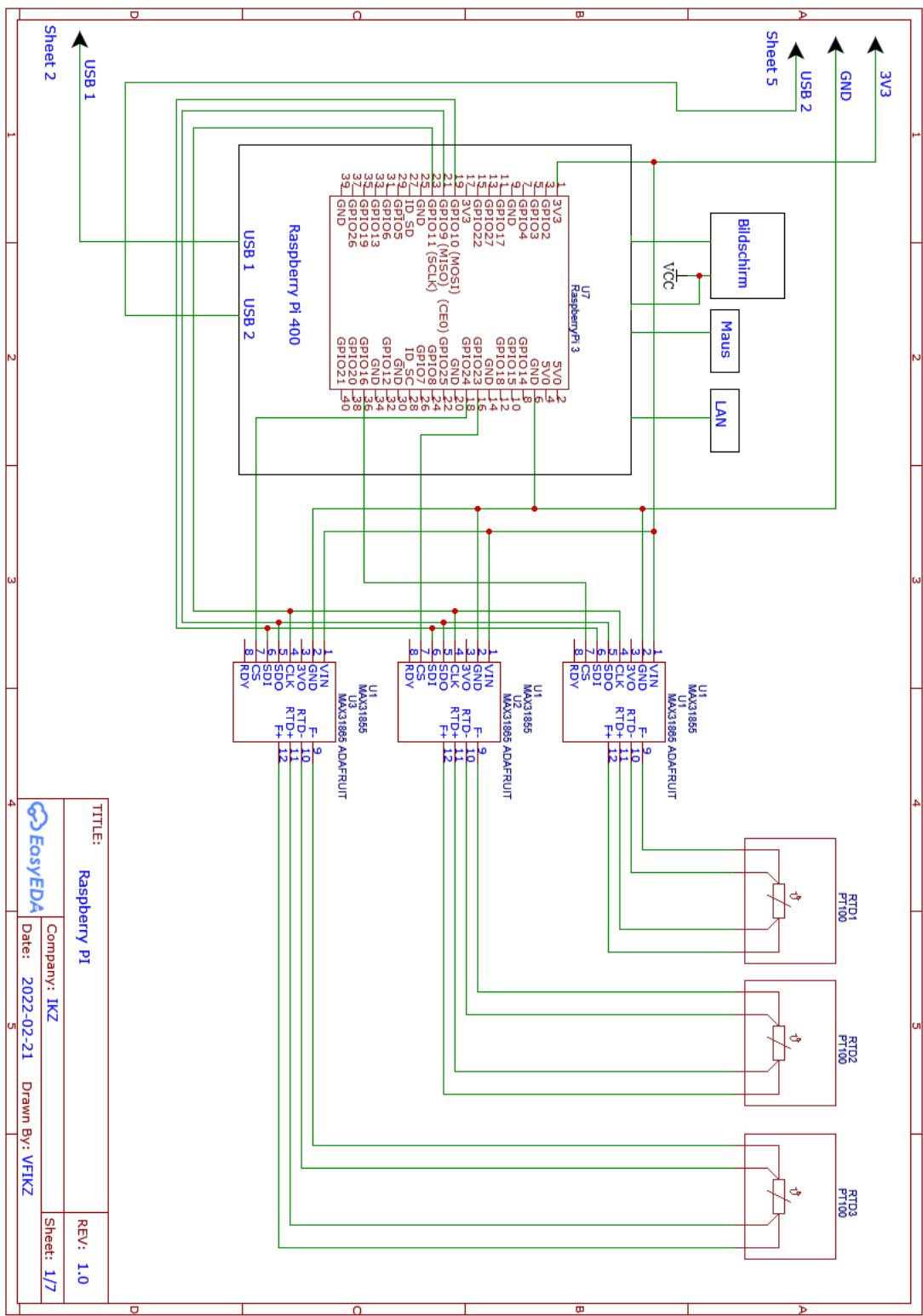
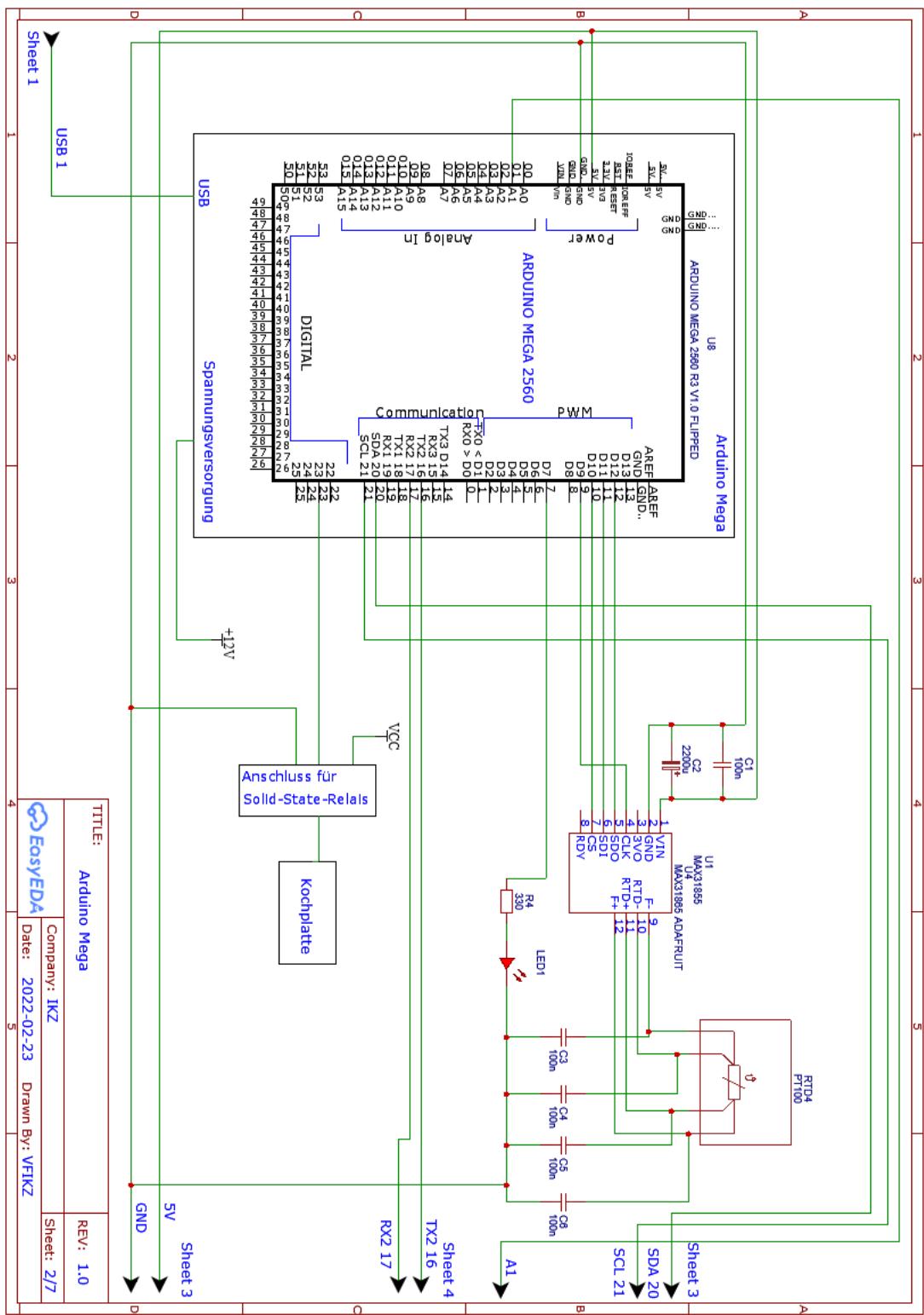


Abbildung 83: Raspberry Pi und Adafruit-Module mit Pt100 - Schaltplan (Quelle: eigene Darstellung)



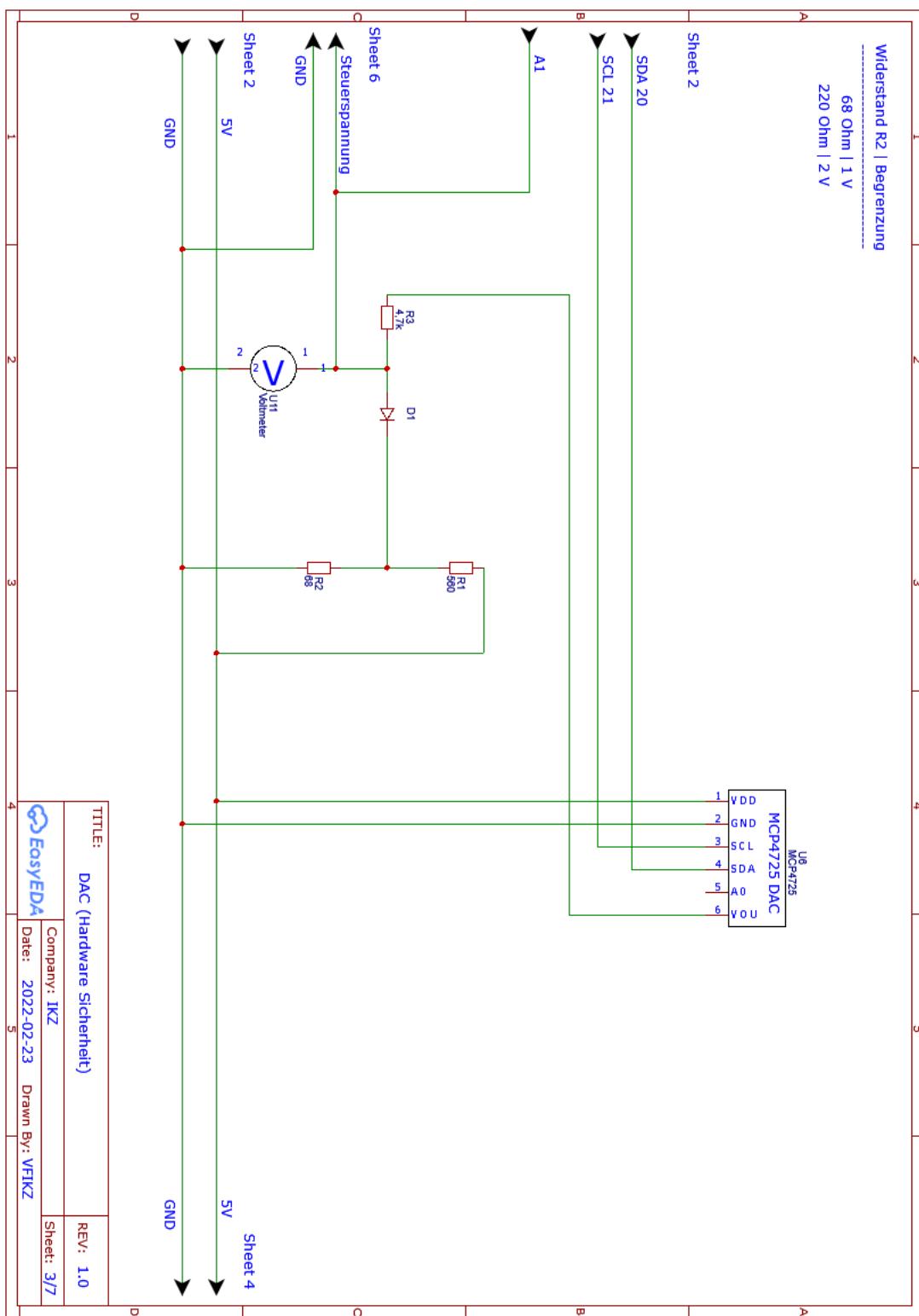


Abbildung 85: DAC-Chip und Hardware-Sicherheit - Schaltplan (Quelle: eigene Darstellung)

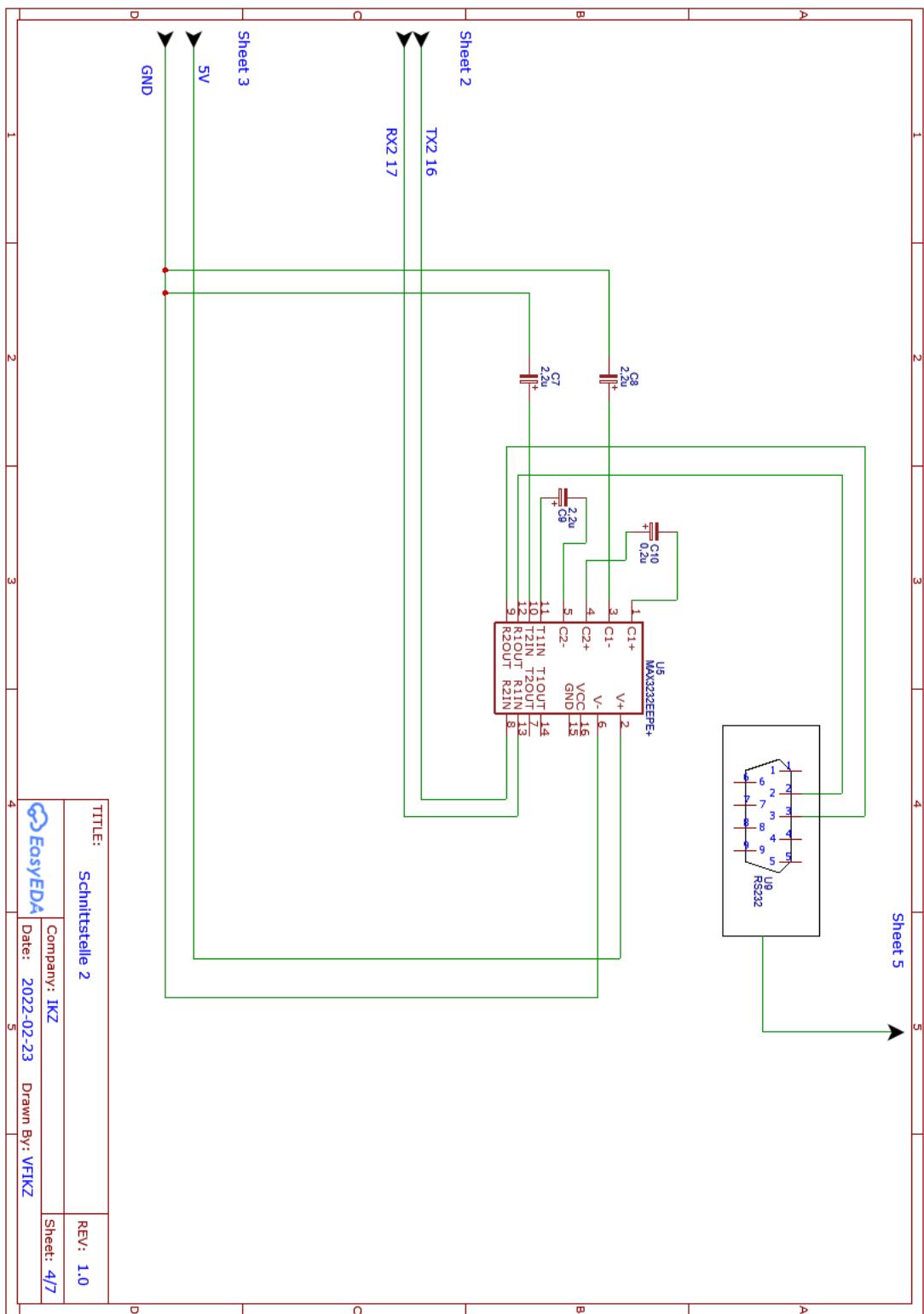


Abbildung 86: Schnittstelle 2 - Schaltplan (Quelle: eigene Darstellung)

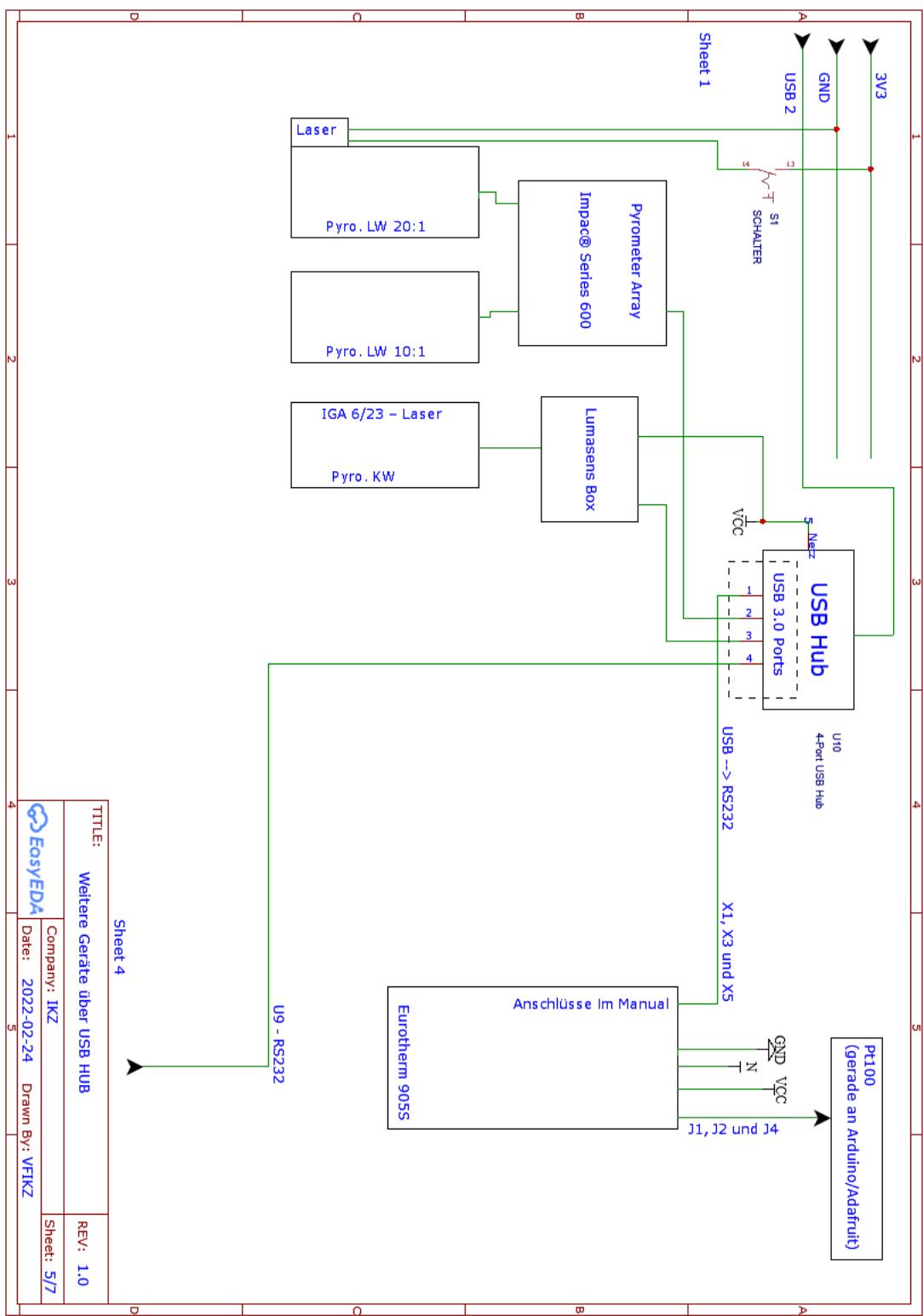


Abbildung 87: USB-Hub mit Pyrometern und Eurotherm - Schaltplan (Quelle: eigene Darstellung)

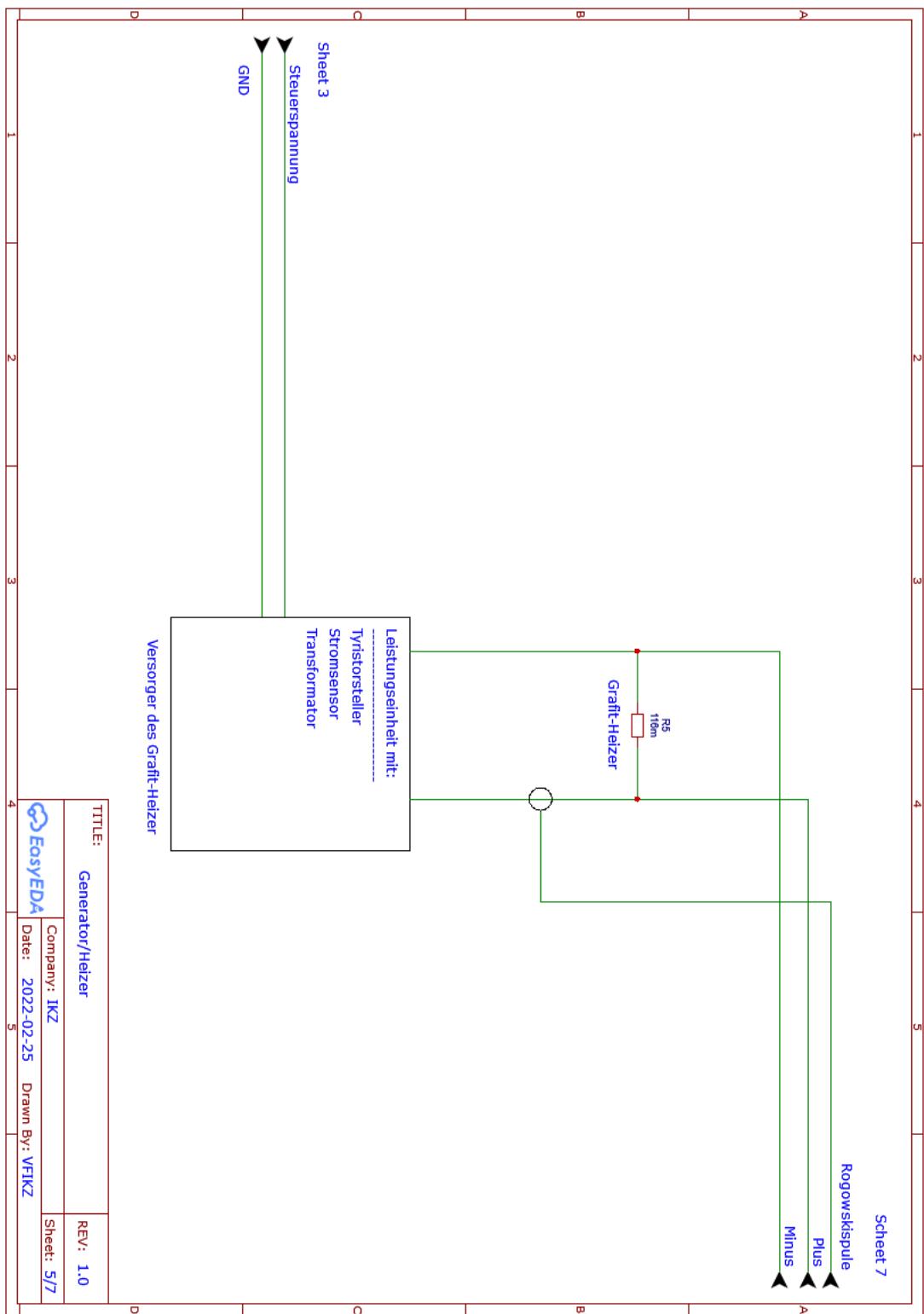


Abbildung 88: Versorgung und Heizer - Schaltplan (Quelle: eigene Darstellung)

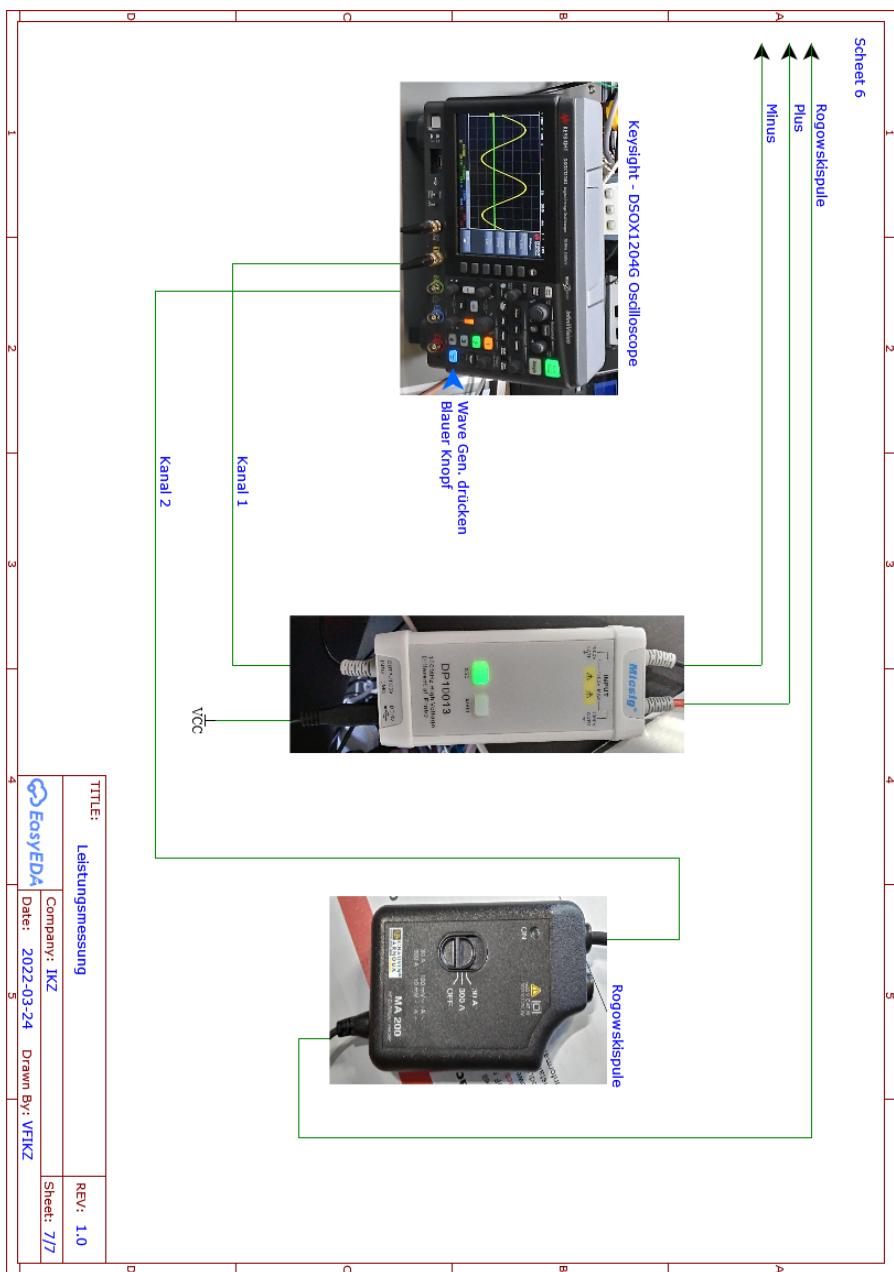


Abbildung 89: Leistungsmessung Grafit-Heizer - Schaltplan (Quelle: eigene Darstellung)

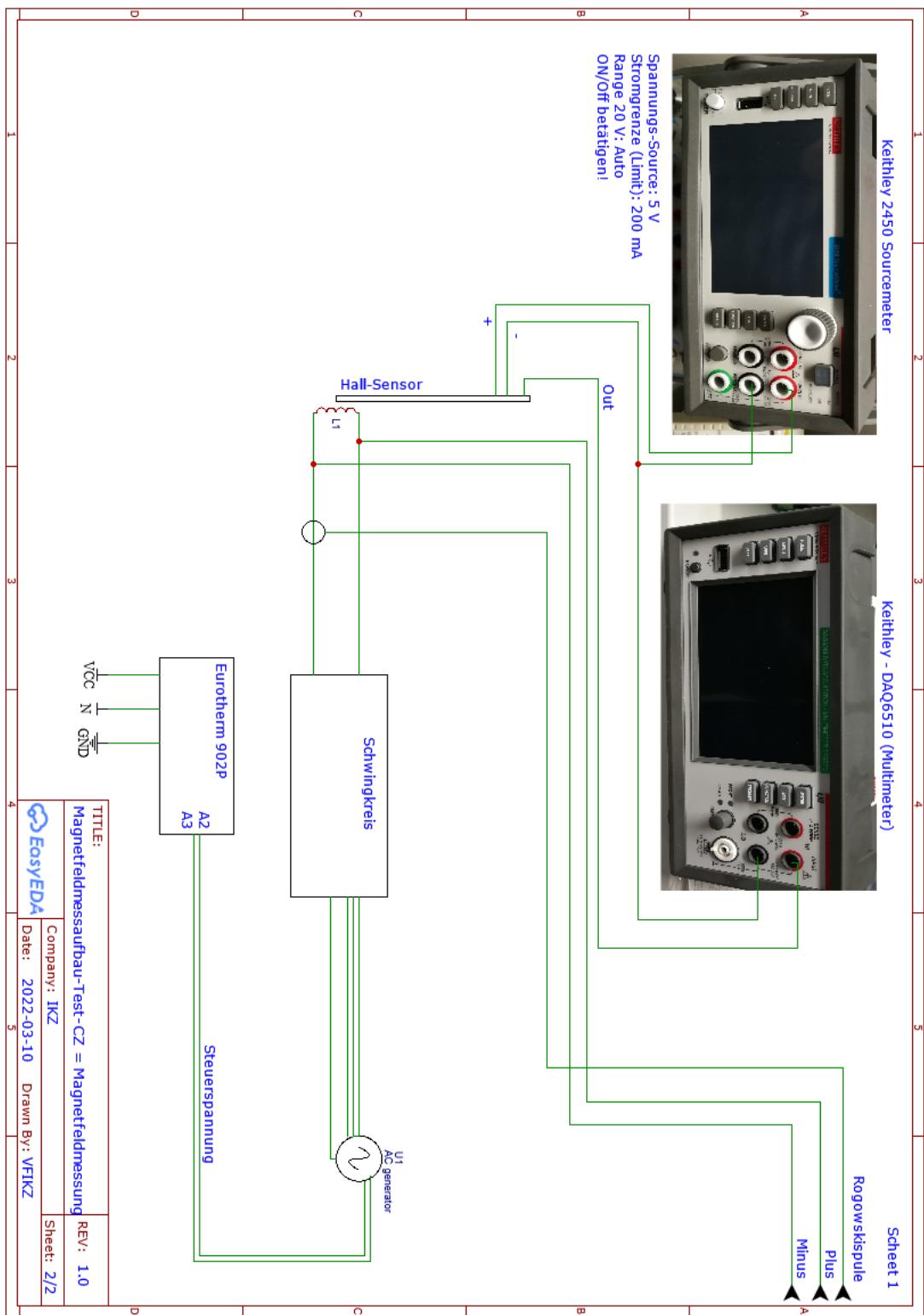


Abbildung 90: Magnetfeld-Profil Messung - Schaltplan (Quelle: eigene Darstellung)

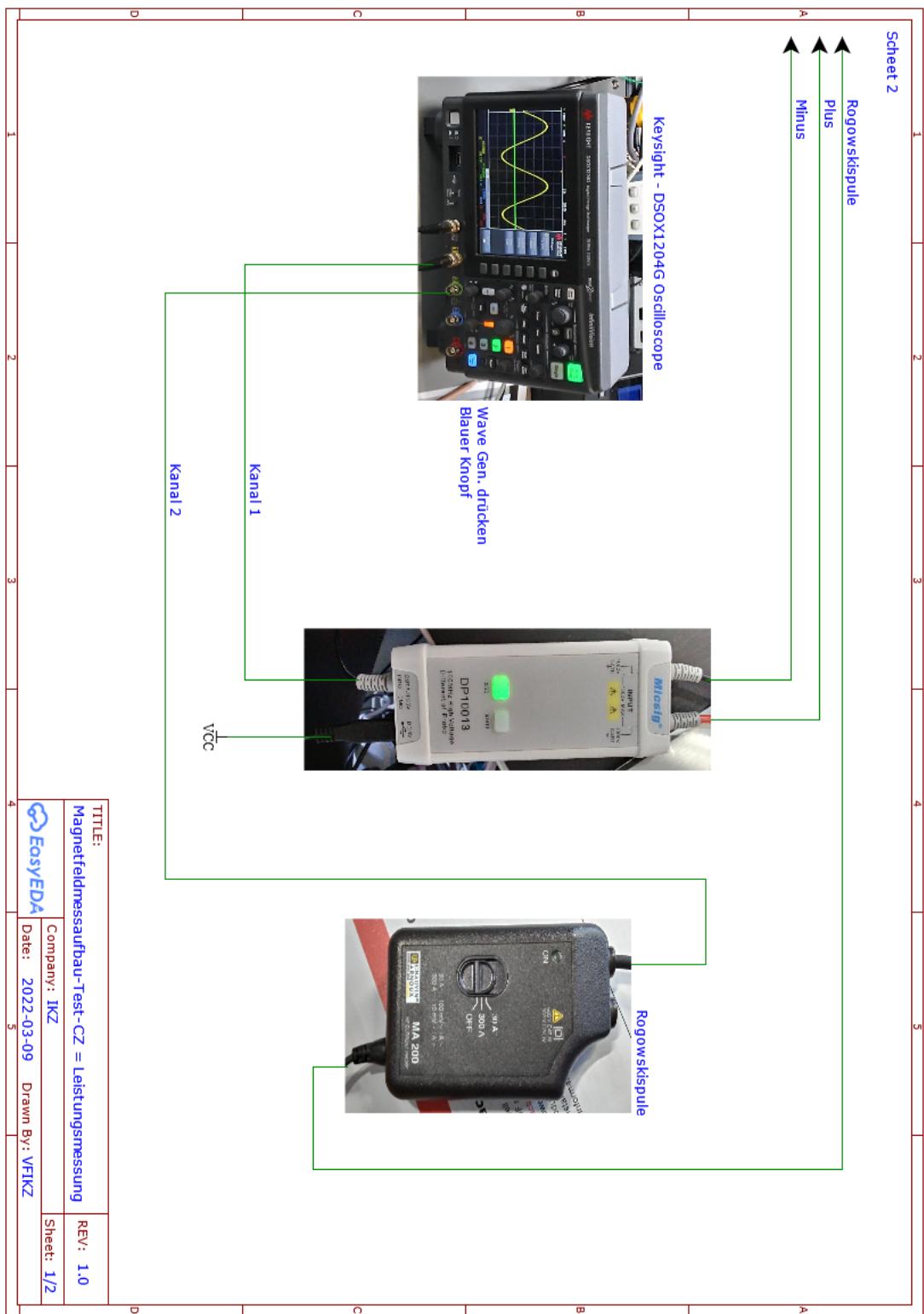


Abbildung 91: Leistungsmessung Spule - Schaltplan (Quelle: eigene Darstellung)

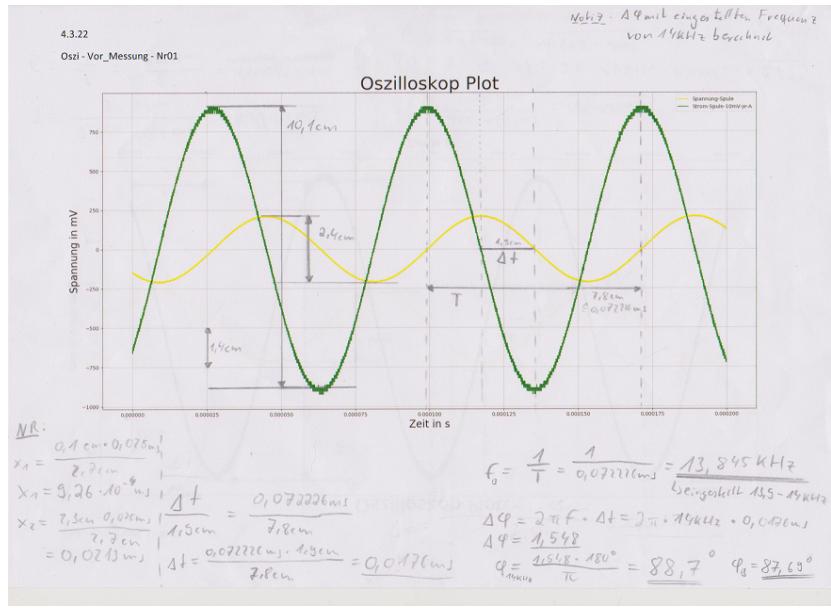


Abbildung 92: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 1 (Quelle: eigene Darstellung)

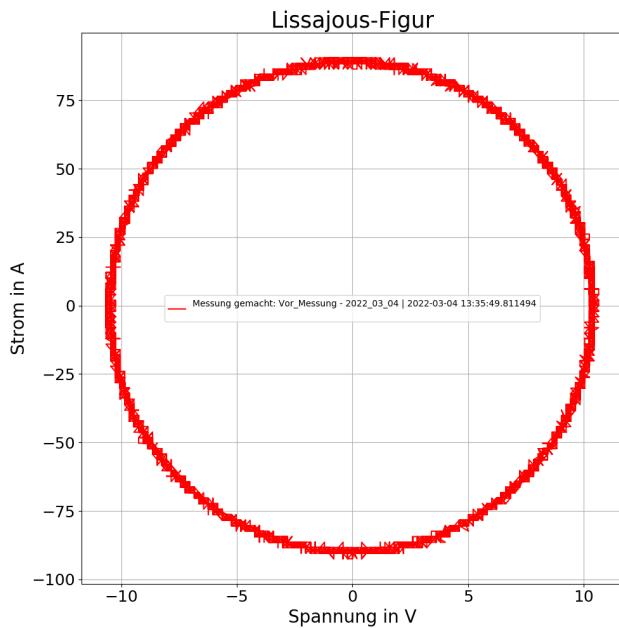


Abbildung 93: Lissajous-Figur zum Versuch Nr. 1 (Quelle: eigene Darstellung)

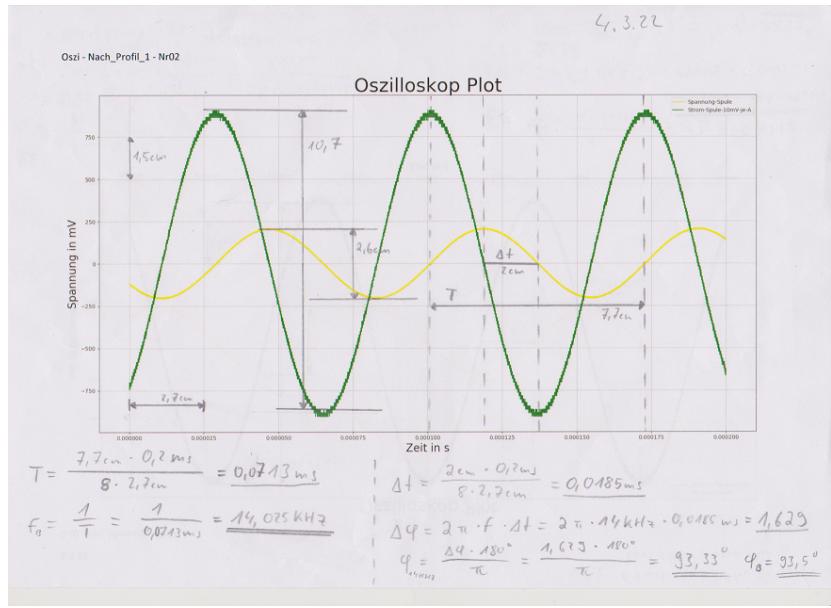


Abbildung 94: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 2 (Quelle: eigene Darstellung)

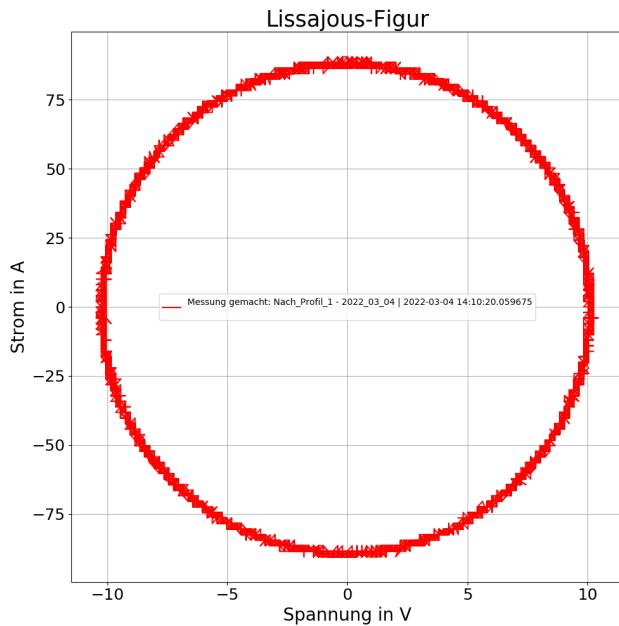


Abbildung 95: Lissajous-Figur zum Versuch Nr. 2 (Quelle: eigene Darstellung)

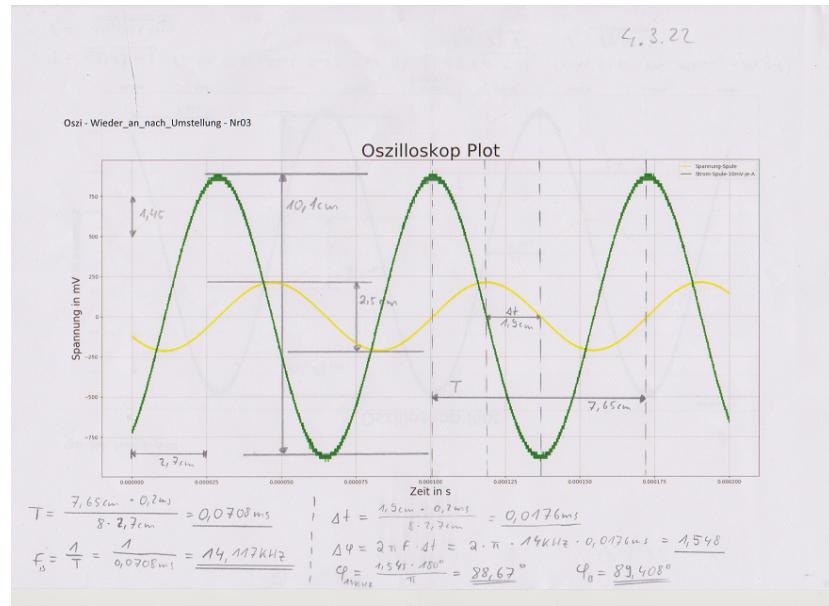


Abbildung 96: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 3 (Quelle: eigene Darstellung)

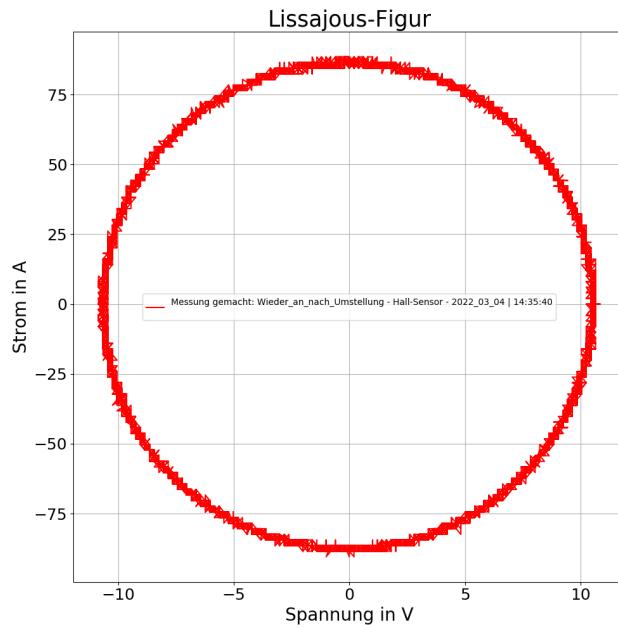


Abbildung 97: Lissajous-Figur zum Versuch Nr. 3 (Quelle: eigene Darstellung)

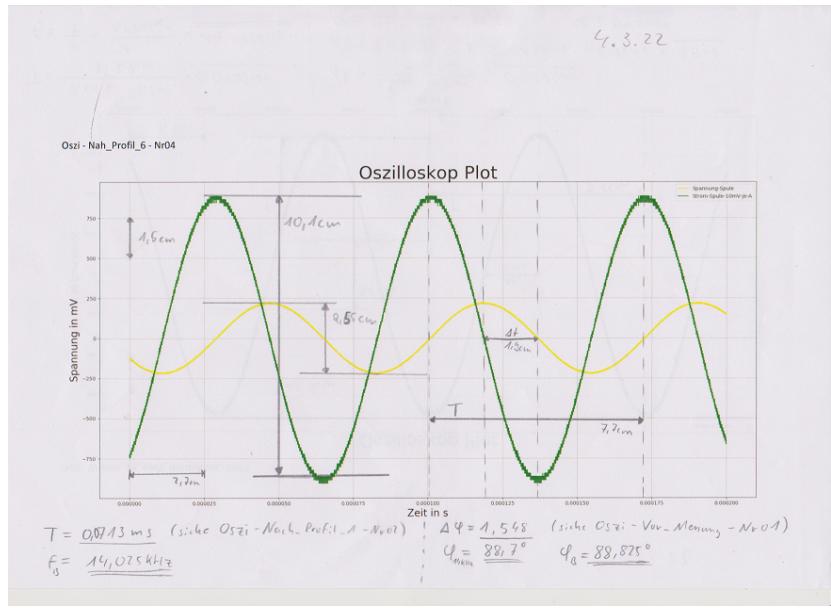


Abbildung 98: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 4 (Quelle: eigene Darstellung)

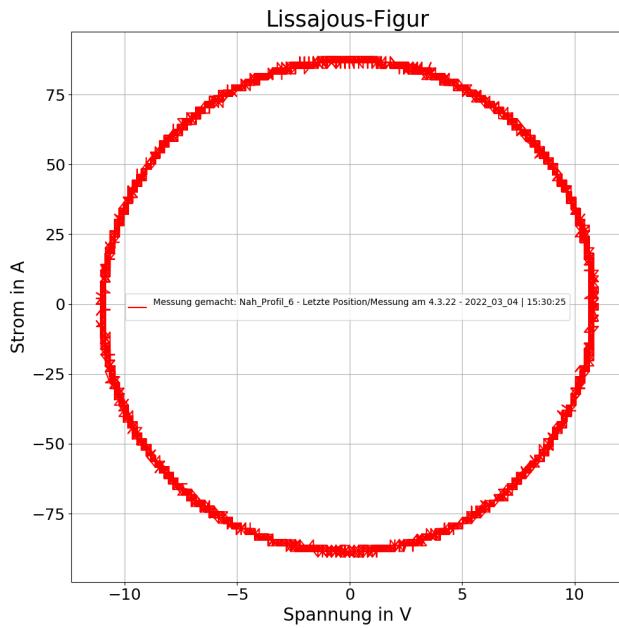


Abbildung 99: Lissajous-Figur zum Versuch Nr. 4 (Quelle: eigene Darstellung)

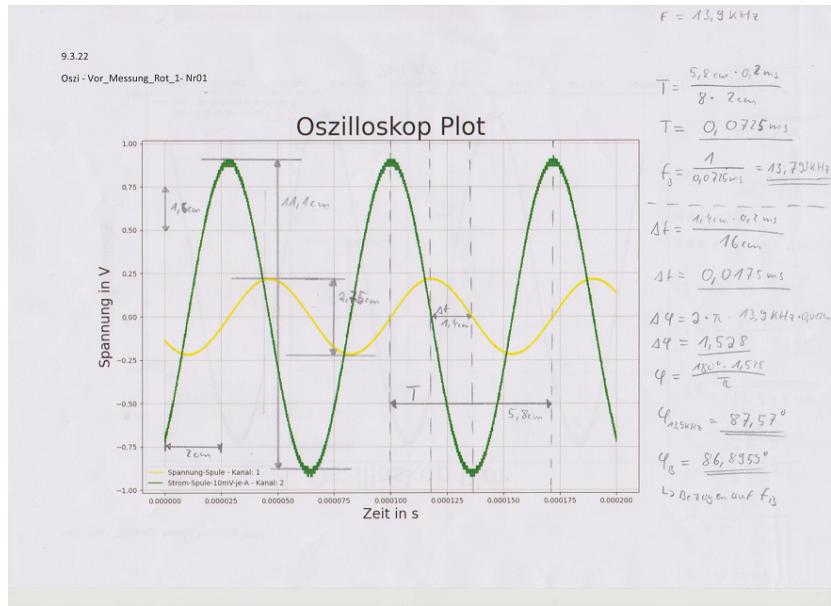


Abbildung 100: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 5 (Quelle: eigene Darstellung)

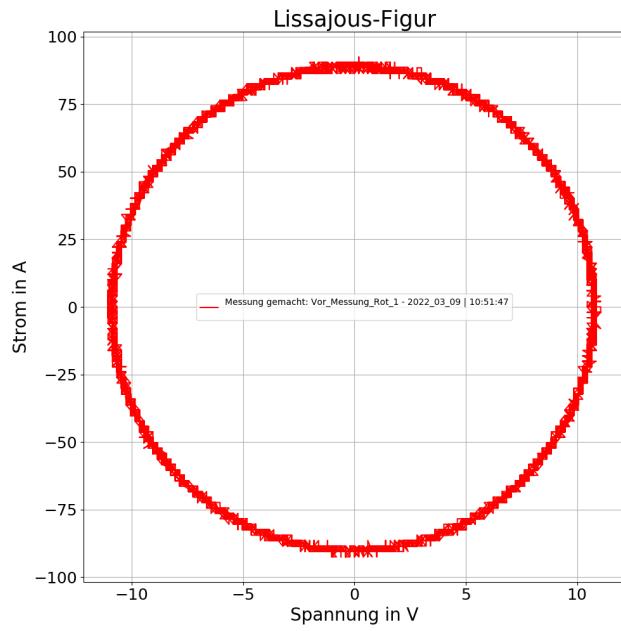


Abbildung 101: Lissajous-Figur zum Versuch Nr. 5 (Quelle: eigene Darstellung)

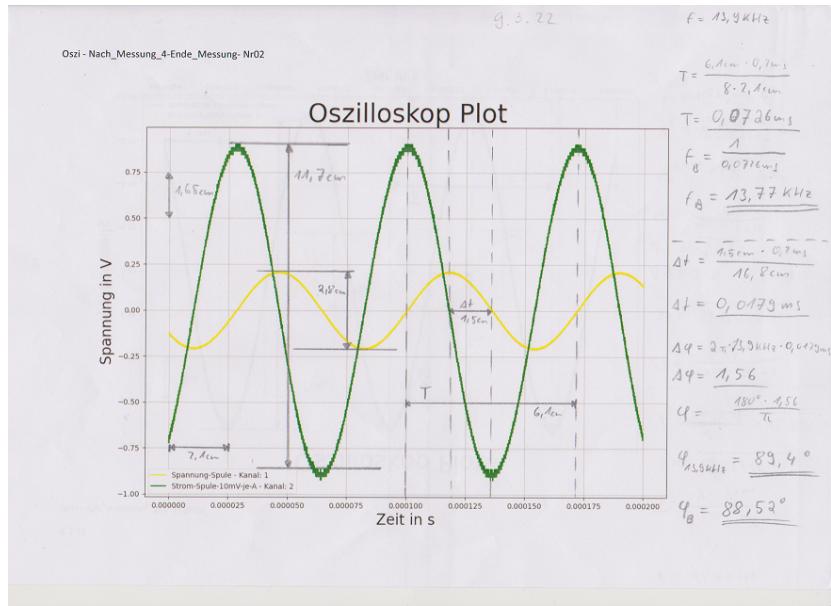


Abbildung 102: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 6 (Quelle: eigene Darstellung)

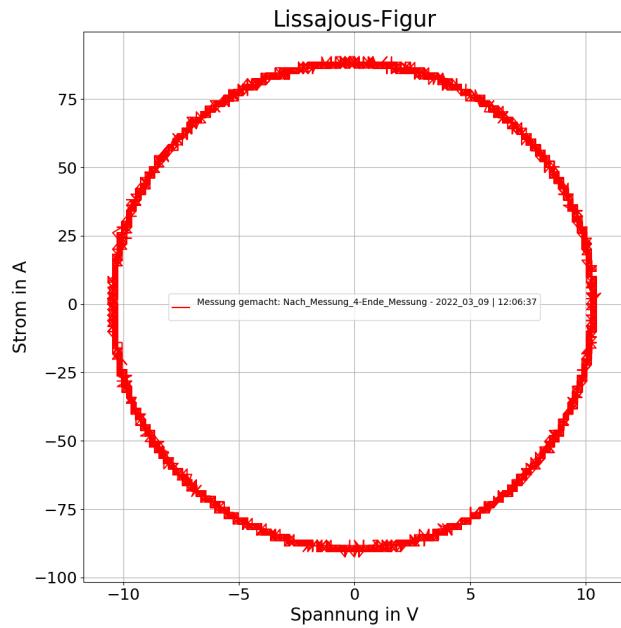


Abbildung 103: Lissajous-Figur zum Versuch Nr. 6 (Quelle: eigene Darstellung)

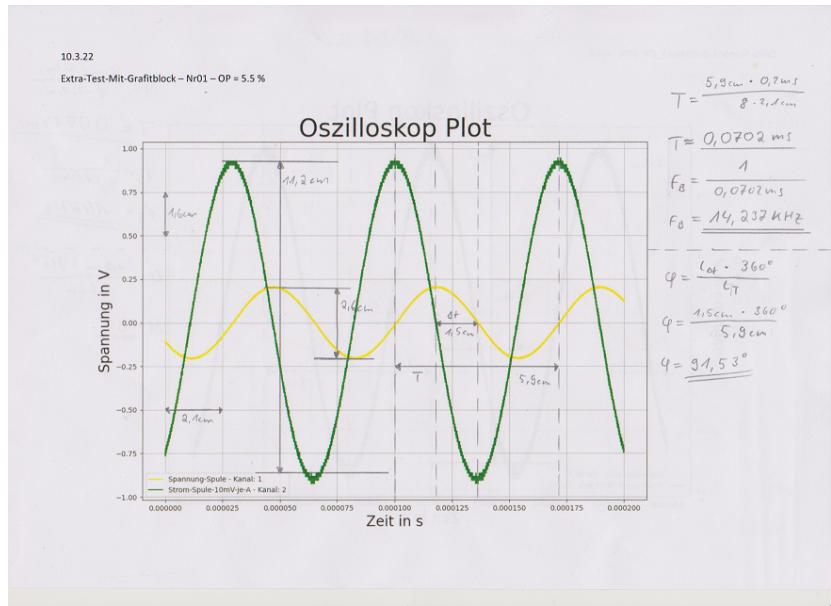


Abbildung 104: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 7 (Quelle: eigene Darstellung)

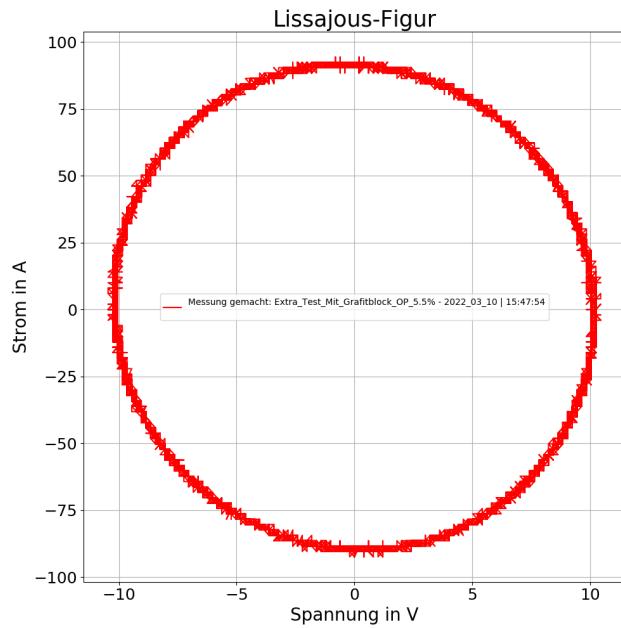


Abbildung 105: Lissajous-Figur zum Versuch Nr. 7 (Quelle: eigene Darstellung)

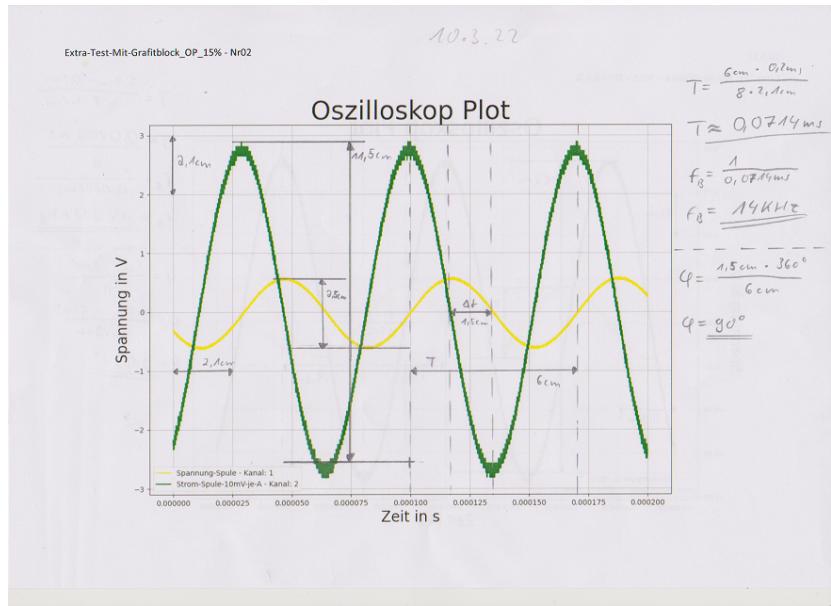


Abbildung 106: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 8 (Quelle: eigene Darstellung)

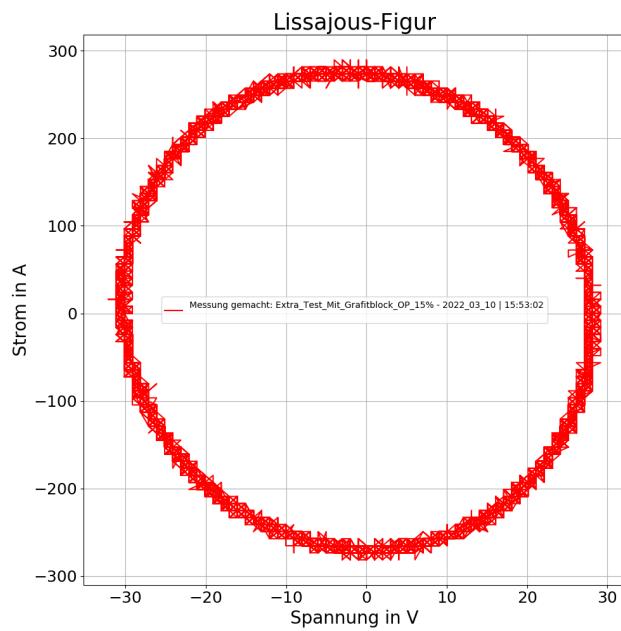


Abbildung 107: Lissajous-Figur zum Versuch Nr. 8 (Quelle: eigene Darstellung)

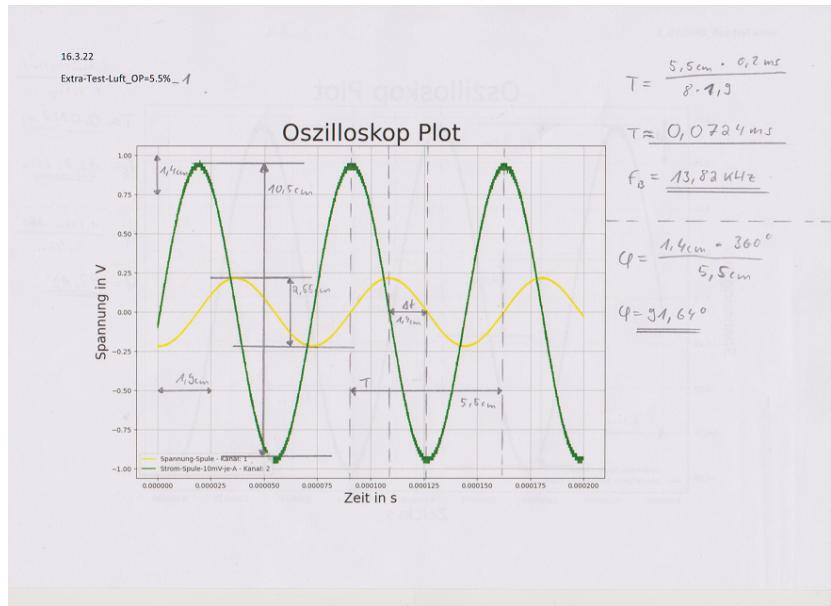


Abbildung 108: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 9 (Quelle: eigene Darstellung)

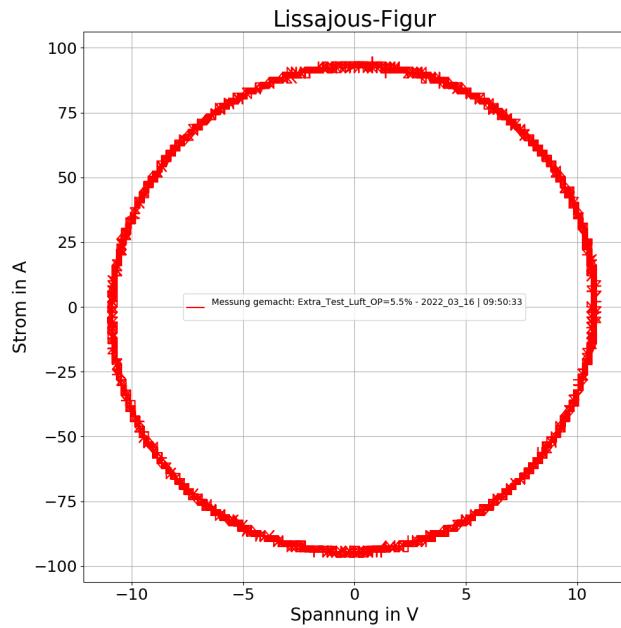


Abbildung 109: Lissajous-Figur zum Versuch Nr. 9 (Quelle: eigene Darstellung)

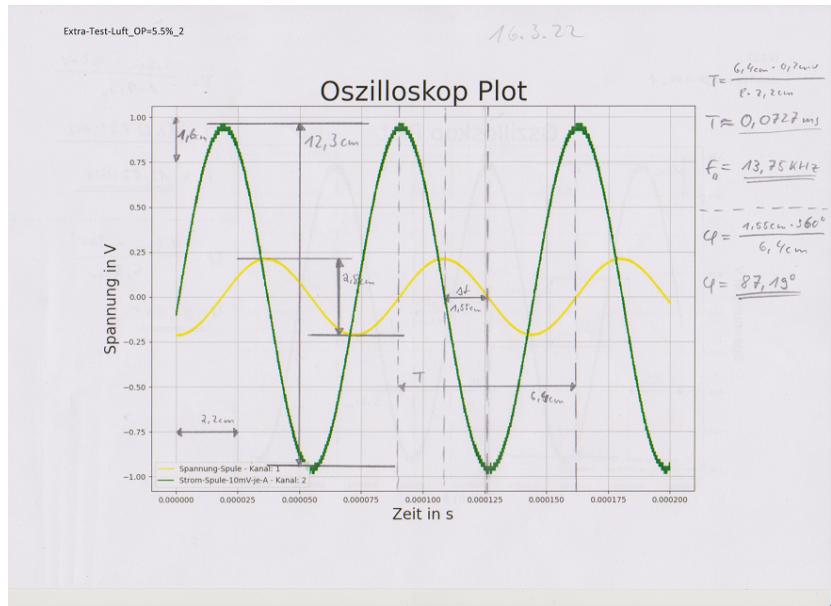


Abbildung 110: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 10 (Quelle: eigene Darstellung)

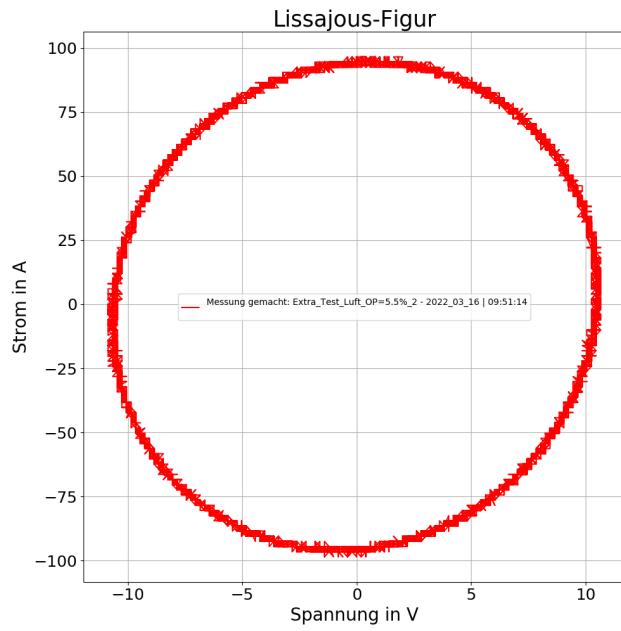


Abbildung 111: Lissajous-Figur zum Versuch Nr. 10 (Quelle: eigene Darstellung)

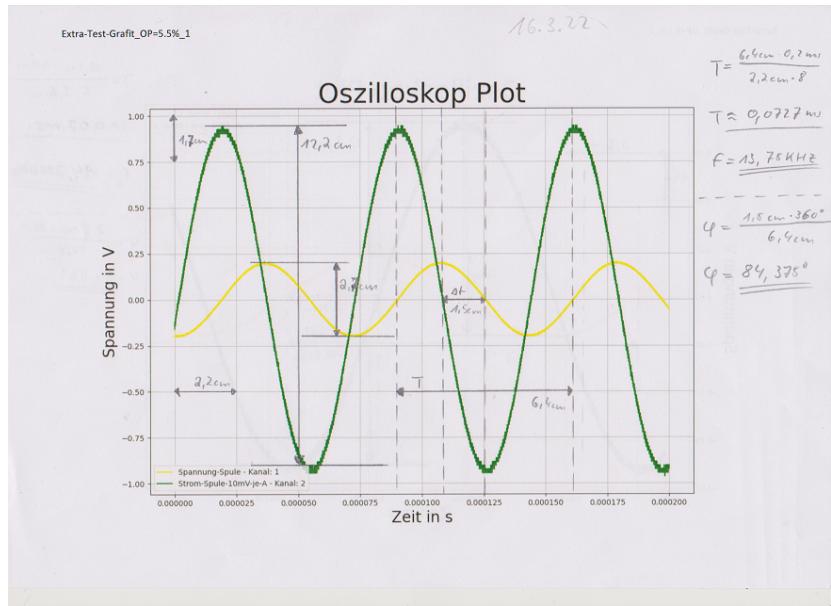


Abbildung 112: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 11 (Quelle: eigene Darstellung)

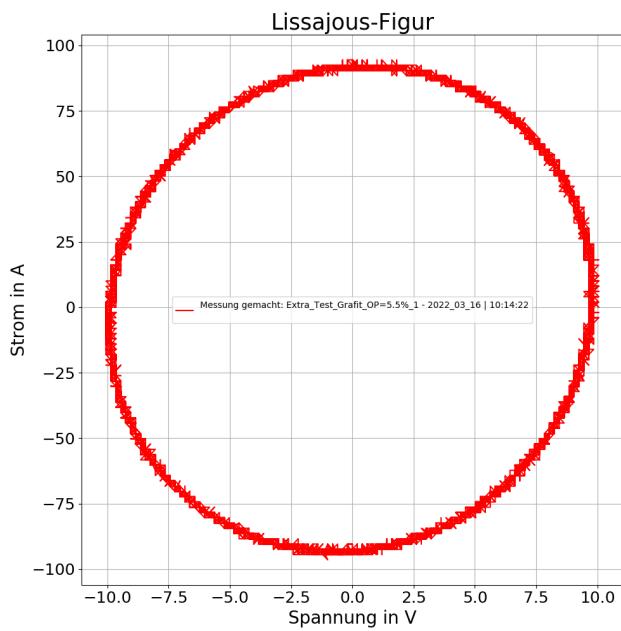


Abbildung 113: Lissajous-Figur zum Versuch Nr. 11 (Quelle: eigene Darstellung)

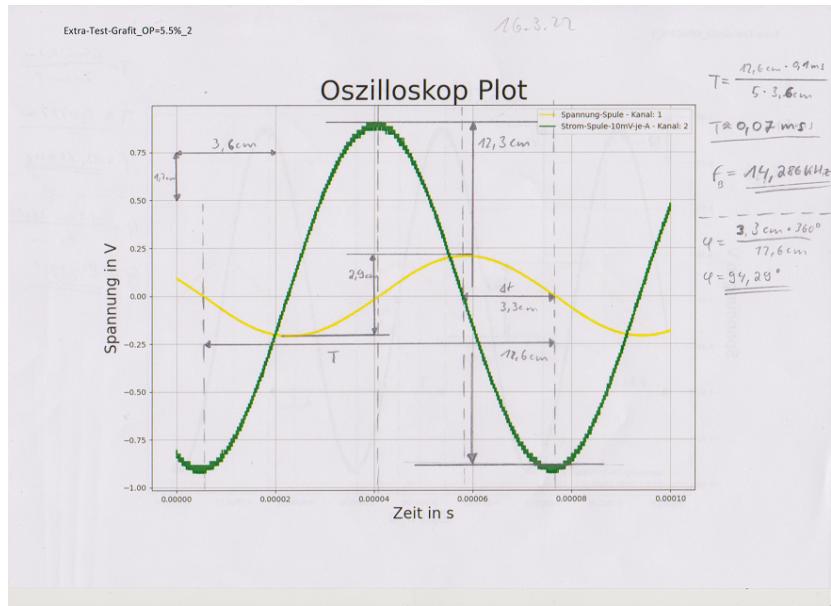


Abbildung 114: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 12 (Quelle: eigene Darstellung)

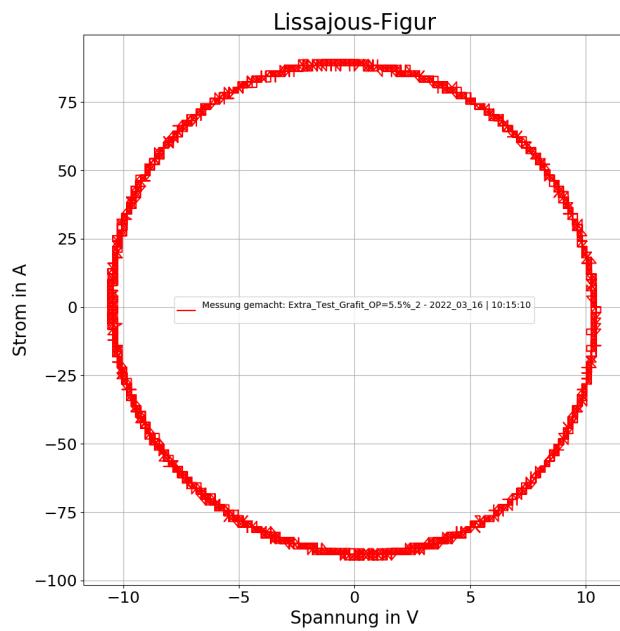


Abbildung 115: Lissajous-Figur zum Versuch Nr. 12 (Quelle: eigene Darstellung)

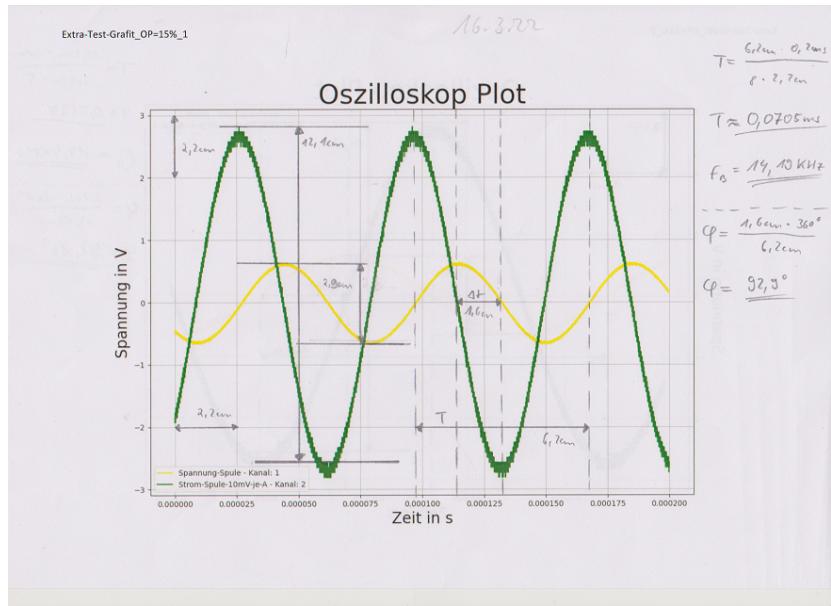


Abbildung 116: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 13 (Quelle: eigene Darstellung)

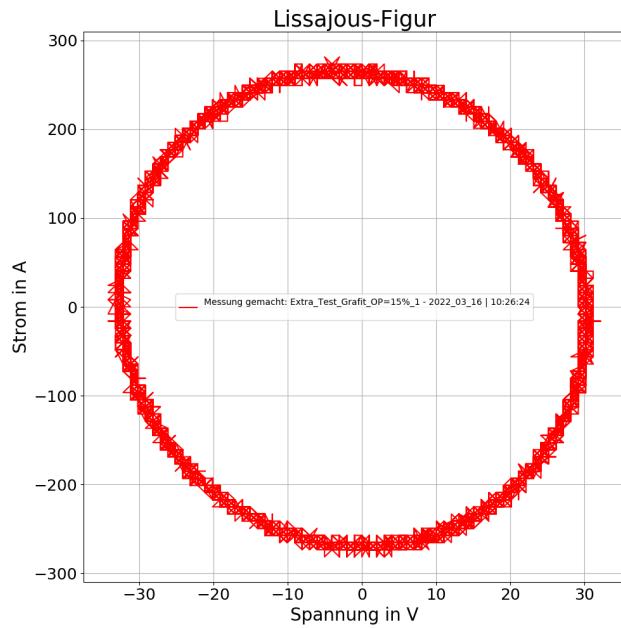


Abbildung 117: Lissajous-Figur zum Versuch Nr. 13 (Quelle: eigene Darstellung)

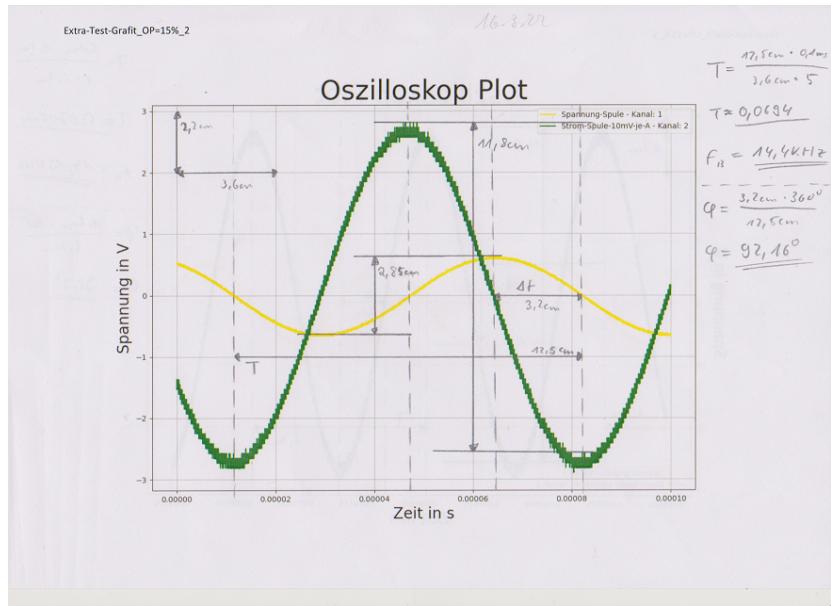


Abbildung 118: Zeichnerische Lösung - Phasenverschiebung - Versuchs-Nr. 14 (Quelle: eigene Darstellung)

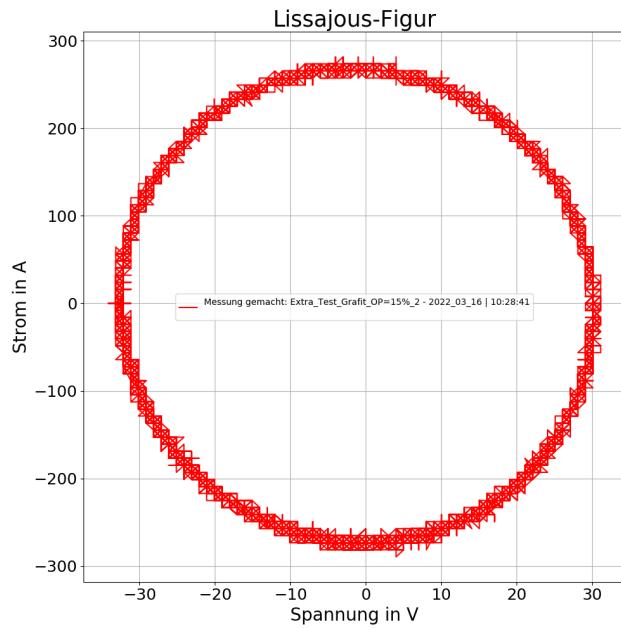


Abbildung 119: Lissajous-Figur zum Versuch Nr. 14 (Quelle: eigene Darstellung)



**European Research Council**  
Established by the European Commission

This research has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (project NEMOCRYS, grant agreement No 851768).