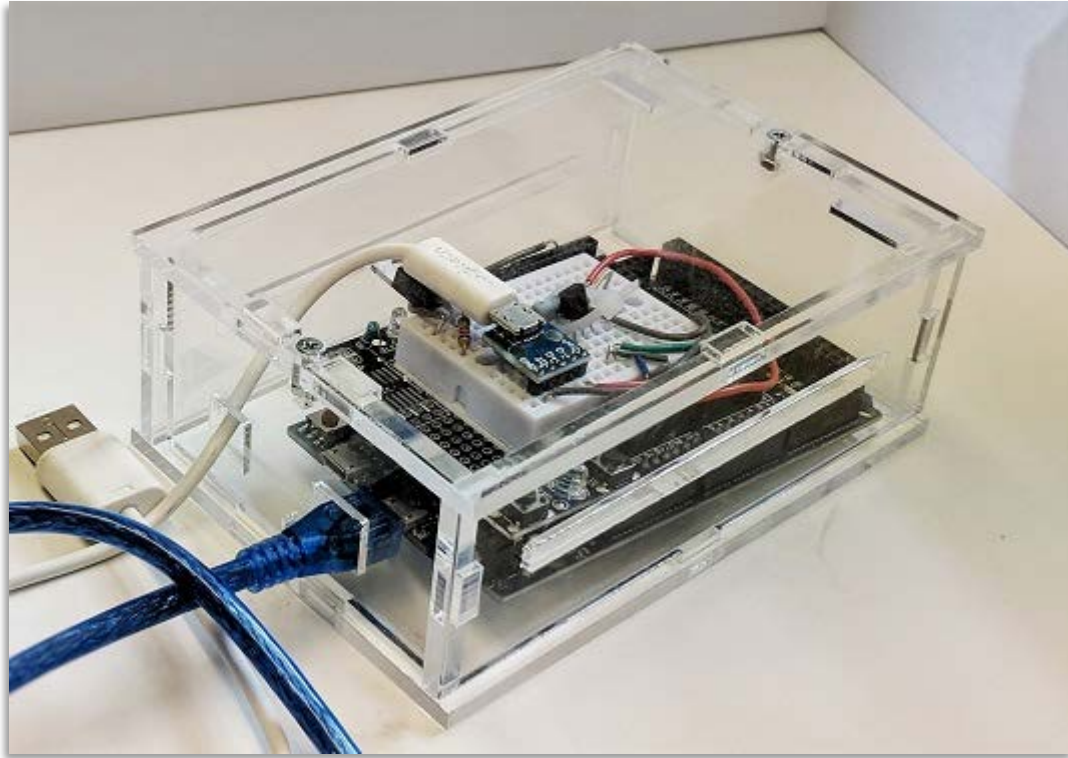


Labor für Mess- und Regelungstechnik

Laborleiter: Prof. Dr.-Ing. Peter Zentgraf



Mobile Temperaturregelung Teil 2

Projektarbeit II

Im Master Studiengang Angewandte Forschung und Entwicklung
in den Ingenieurwissenschaften

Vorgelegt am 04.03.2019

von

Sebastian Fischer

Inhaltsverzeichnis

Abbildungsverzeichnis	iii
1 Einleitung	5
2 Geräteplanung.....	6
2.1 Anforderungen an das Projekt.....	6
2.2 Komponenten	6
2.2.1 Mikrocontroller	6
2.2.2 Leistungsteil	8
2.2.3 Leistungswiderstand.....	10
2.2.4 Temperatursensor	11
3 Hard- und Softwaretechnische Umsetzung.....	13
3.1 Stückliste für den Hardware Aufbau.....	13
3.2 Ansteuerung des MOSFET	13
3.3 Temperaturmessung.....	19
3.4 Experimentelle Bildung der Übertragungsfunktion mit pzMove.....	20
3.5 Messung und Berechnung des Frequenzgangs	25
3.6 Aufbau der Simulink Programme	28
3.6.1 Aufbau der Steuerung	29
3.6.1 Aufbau der Steuerung mit Wunschübertragungsfunktion.....	31
3.6.2 Aufbau der Regelung.....	33
3.6.3 Störgrößenkompensation	36
4 Zusammenfassung	42
5 Literatur	45

Abbildungsverzeichnis

Abbildung 2-1: Arduino DUE Draufsicht [1].....	7
Abbildung 2-2: Funktionsweise eines n-Kanal MOSFET.....	8
Abbildung 2-3: Aufbau eines n-Kanal-Anreicherungstyp MOSFET	8
Abbildung 2-4: MOSFET n-Kanal, Anreicherungstyp	8
Abbildung 2-5: Verbindungs-Schaubild Schaltung	9
Abbildung 2-6: Verbindungs-Schaubild Protoshield mit Arduino Due [7].....	10
Abbildung 2-7: Platinschaltplan	10
Abbildung 2-8: Verifikation des LM35cz Temperatursensors mittels eines hochpräzisen Temperatursensors	11
Abbildung 2-9: Temperatursensor LM35cz	11
Abbildung 3-1: Matlab Add-Ons.....	13
Abbildung 3-2: SIMULINK Library Browser.....	14
Abbildung 3-3: Pulsweitenmodulation [9].....	14
Abbildung 3-4: PWM-Signal.....	15
Abbildung 3-5: Vergleich von gemessener mit berechneter Leistung und Spannung.....	16
Abbildung 3-6: Pcalc über Pmeas.....	17
Abbildung 3-7: Ansteuerung des PWM Blocks in SIMULINK	18
Abbildung 3-8: Temperaturmessung in Simulink.....	19
Abbildung 3-9: Vergleich von vier Sprungantworten bei gleichem Input und unterschiedlichen Anfangs- und Raumtemperaturen.....	20
Abbildung 3-10: Experimentelle Bildung der Übertragungsfunktion mit pzMove	22
Abbildung 3-11: stationäre Temperaturänderung bei verschiedenen effektiver Eingangsleistung (Systemruhelage)	23
Abbildung 3-12: Stationäre Verstärkung bei steigendem Leistungsinput	23
Abbildung 3-13: dT_w über dT_m	24
Abbildung 3-14: Vergleich der stationären Verstärkungen bei unterschiedlichen U_{max}	25
Abbildung 3-15: Frequenzgangmessung bei $\omega = 0.05$ rad/s	26
Abbildung 3-16: Frequenzgangmessung bei $\omega = 0.4$ rad/s	27
Abbildung 3-17: Bodeplot - Vergleich von experimentell gebildeter Übertragungsfunktion mit gemessenem Frequenzgang	28
Abbildung 3-18: Simulink-Modell für Steuerung.....	29
Abbildung 3-19: Vergleich der Ein- und Ausgangsgrößen der Simulation mit den realen Werten der Steuerung	30
Abbildung 3-20: Simulink Modell der Steuerung mit Wunschübertragungsfunktion.....	32
Abbildung 3-21: Steuerung mit GW, 1500 Sekunden	32
Abbildung 3-22: Vergleich der Ein- und Ausgangsgrößen der Simulation mit dem realen Werten der Steuerung mit GW inkl. Ventilatorstörung, 500 Sekunden.....	33
Abbildung 3-23: Aufbau eines Regelkreises	33
Abbildung 3-24: PID Einstellregeln nach Chien, Hrones und Reswick	34
Abbildung 3-25: Simulink Modell der Regelung	35
Abbildung 3-26: Regelung, 1500 Sekunden	35
Abbildung 3-27: Vergleich der Ein- und Ausgangsgrößen der Simulation mit dem realen Werten der Regelung, 500 Sekunden.....	36
Abbildung 3-28: Schätzen einer Eingangsstörung	37
Abbildung 3-29: Gleichzeitiges Schätzen von Ein- und Ausgangsstörung D_{ur} und D_{uy} , sowie N_r	38

2.2.1 Mikrocontroller

Abbildung 3-30: Kompensation der Störung im geschlossenen Regelkreis	38
Abbildung 3-31: Vergleich der Störgrößenkompensation bei verschiedenen Zeitkonstanten T	41
Abbildung 4-1: Simulink Modell Steuerung	42
Abbildung 4-2: Simulink Modell Steuerung Mit GW	42
Abbildung 4-3: Simulink Modell Regelung	42
Abbildung 4-4: Simulink Modell Kompensation	42
Abbildung 4-5: Steuerung	42
Abbildung 4-6: Steuerung mit GW	42
Abbildung 4-7: Regelung	42
Abbildung 4-8: Kompensation	42
Abbildung 4-9: Vergleich Regelung, Steuerung mit GW	43

1 Einleitung

Temperaturregelungen sind heutzutage nicht mehr aus Alltag und Industrie wegzudenken. Sei es eine Heizungsregelung für Smart-Home Produkte, oder die exakte Temperaturregelung in industriellen Anlagen, bei welchen kleine Abweichungen schon einen großen Einfluss haben können. Sogar im eigenen Körper findet eine Regelung der Körpertemperatur statt, welche entscheidend dafür ist, dass wir leben können. Der Begriff „Regelung“ wird im allgemeinen Sprachgebrauch oft verwechselt mit „Steuerung“. Eine Steuerung ist im Gegensatz zu einer Regelung ein rein vorwärts gerichteter Prozess ohne Rückkopplung. Die Ausgangsgröße wird dabei nicht überwacht und kann sich durch Störungen von außen verändern. Eine Steuerung ist daher auch sehr schnell, was ein entscheidender Vorteil gegenüber einer Regelung ist. Das Regeln ist ein Vorgang, bei dem die Ausgangsgröße fortlaufend überwacht wird und bei Abweichung über die Stellgröße korrigiert wird. Der sich ergebende Wirkungsablauf findet in einem geschlossenen Kreis, dem Regelkreis statt. Da der Sollwert ständig mit einem Istwert verglichen wird, ist die Regelung bedeutend langsamer als die Steuerung. Diese Projektarbeit dient einerseits dem Zweck, Studenten der Regelungstechnik den generellen Unterschied zwischen Steuerung und Regelung auf anschauliche und verständliche Art und Weise zu erklären. Andererseits hat diese Arbeit das Ziel, die theoretischen Grundlagen der Regelungstechnik anhand eines praktischen Beispiels besser verstehen zu können, indem die Regelstrecke vom Hardware Aufbau bis hin zur softwaretechnischen Umsetzung aufgebaut wird.

2 Geräteplanung

2.1 Anforderungen an das Projekt

Zu aller erst werden zur Planung des Temperatur Demonstrationsversuchs die Anforderungen, die er erfüllen soll, definiert. Dazu gehören folgende Punkte:

- Der Demonstrationsversuch soll leicht und mobil sein, sodass er in die Vorlesung mitgebracht werden kann
- Mittels eines Gehäuses soll die Robustheit des Versuchsaufbaus im Vergleich zur Projektarbeit I gesteigert werden
- Die Versuchsbestandteile sollen mittels einer Stückliste zusammengefasst sein und der Aufbau von Studenten mit möglichst kleinem Aufwand nachgebildet werden können
- Die Komponenten sollen möglichst günstig sein
- Die Temperatur eines geeigneten Leistungswiderstandes soll um ca. 10 Kelvin erhöht werden können (eine Reduzierung der Temperatur muss nicht realisiert werden, es reicht das bloße Abkühlen durch Warten aus)
- Die Leistungsversorgung soll mittels USB 2.0 Port eines Laptops geschehen (5V, 0,5 A)
- Der Heizvorgang soll in möglichst kurzer Zeit geschehen, damit eine kurze Vorführung in der Vorlesung möglich ist
- Die Hardware soll per Laptop mit Simulink, einem Zusatzprodukt von Matlab geregelt und gesteuert werden
- Mittels unterschiedlicher Simulink Modelle soll eine Steuerung, eine Regelung und eine Regelung mit Störgrößenkompensation realisiert und miteinander verglichen werden

2.2 Komponenten

2.2.1 Mikrocontroller

Ein Mikrocontroller ist prinzipiell dasselbe wie ein Mikrorechner. Ein Mikrorechner wiederum ist ein Computer/Rechner, welcher aus einem oder mehreren Mikroprozessoren besteht. Der Mikroprozessor ist die Zentraleinheit (CPU, Central Processing Unit) eines Datenverarbeitungssystems, welche heutzutage in der Regel mit weiteren Komponenten auf einem einzigen Chip untergebracht werden. Neben dem oder den Mikroprozessor(en) sind auf einem Mikrorechner weitere Komponenten wie Speicher, Ein-/Ausgabeschnittstellen sowie ein Verbindungssystem enthalten [2]. Ziel eines Mikrocontrollers ist es, Steuerungs- oder Kommunikationsaufgaben mit möglichst wenig Bausteinen zu lösen. Dabei sind die Komponenten Prozessorkern, Speicher und Ein-/Ausgabeschnittstellen eines Mikrocontrollers auf die Lösung solcher Aufgaben ausgelegt [2].

Ein Beispiel für einen Mikrocontroller stellt der im Projekt verwendete Arduino Due der Firma Arduino dar. Für die Verwendung dieses Mikrocontrollers spricht die Kompatibilität zu Simulink, einem Zusatzprodukt von MATLAB, welches in den Projektanforderungen als Programmiersoftware vorgeschrieben ist. Der Arduino Due ist ein Mikrocontroller Board, basierend auf einem Atmel SAM3X8E ARM Cortex-M3 CPU. Es ist das erste Arduino Board, welches auf einem 32-bit ARM core basiert und folgende technische Daten aufweist:

- 54 digitale Input/Output Pins (von denen 12 als PWM Outputs genutzt werden können)
- 12 analog Inputs
- 4 UARTs
- 84 MHz clock
- 512 KB Speicher
- USB bzw. OTG fähige Verbindung
- 2 Digital to Analog Converter (12-Bit Auflösung)
- I2C und SPI Schnittstelle

Der Arduino Due wird im Gegensatz zum Arduino Uno mit 3,3 V betrieben. Hinzugefügte Spannungen zu jedem Input/Output welche höher als die 3,3V sind, können das Board beschädigen [1]. Im Gegensatz dazu basiert der am häufigsten verwendete Arduino Uno auf einem ATmega328P, hat lediglich 14 digitale Input/Output Pins (von denen nur 6 als PWM Outputs genutzt werden können), 6 analoge Inputs, 32 KB Speicher und USB Verbindung [3]. Im Verlaufe des Projekts hat sich gezeigt, dass die deutlich höhere Leistung des Arduino Due auch gebraucht wird, sobald die Simulink Modelle umfangreicher werden. Der Arduino Due ist mit ca. 37 \$ allerdings wesentlich teurer als die Uno Variante (ca. 20 \$). Jedoch kann auf eine kostengünstigere Variante des Due aus China zurückgegriffen werden, welche einwandfrei funktioniert, genauso rechenstark ist aber lediglich ca. 14 € kostet [4].

Folgende Abbildung zeigt einen Arduino Due. Zur Spannungsversorgung kann entweder ein Hohlbuchsenstecker oder der USB Anschluss verwendet werden.

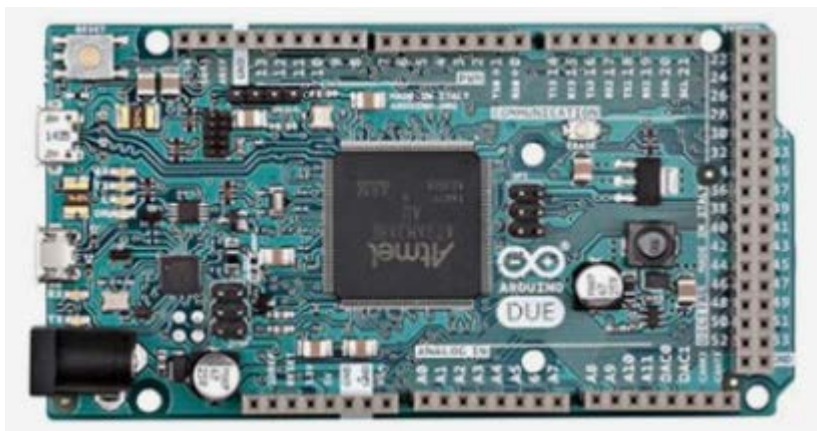


Abbildung 2-1: Arduino DUE Draufsicht [1]

Der USB Anschluss für die Programmierung ist mit einem Atmega 16U verbunden, der als USB-Host (UART) fungiert. Dieser Chip kommuniziert wiederum mit einem seriellen Bus

2.2.2 Leistungsteil

(Serial 0) mit dem Hauptprozessor. Der zur Programmierung vorgesehene USB Anschluss ist der untere, direkt neben dem Hohlbuchsenstecker [1].

2.2.2 Leistungsteil

Zur Reduzierung der Projektkosten wird als Leistungsteil nicht das in Projektarbeit I genutzte VMA03 Motorshield von Velleman verwendet, welches mit ca. 20,00 € verhältnismäßig teuer ist, sondern stattdessen eine eigene Schaltung entwickelt. Noch dazu, ist somit keine weitere Spannungsquelle wie in Projektarbeit I nötig. Es kann stattdessen der USB 2.0 Port eines Laptops als Leistungsquelle verwendet werden. Durch einen IRF540N MOSFET, welcher durch einen PWM Pin des Arduino Due gesteuert wird, kann später die konstante Gleichspannung von 5V des USB Ports zwischen 0 bis 5 V variiert werden. Der genannte MOSFET ist bestens für den Arduino Due mit 3,3V Betriebsspannung geeignet und fällt mit 0,70 € kaum ins Gewicht.

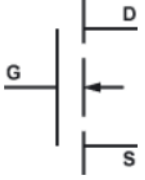
	n-Kanal
MOS-FET Typ	Anreicherungstyp (selbstsperrend)
I_D bei U_{DS}	positiv
U_{GS} (Steuerspannung)	positiv
Schaltzeichen	
Anwendung	Leistungsverstärker

Abbildung 2-4: MOSFET n-Kanal, Anreicherungstyp

MOSFETs (Metal Oxide Semiconductor Field Effect Transistor) sind Basisbausteine der Mikroelektronik, welche ein praktisch verlustfreies und schnelles elektronisches Schalten bei hoher Integration ermöglichen [5]. Insgesamt gibt es vier verschiedene MOS-FET-Typen, wobei man zwischen n-Kanal- und p-Kanal- sowie

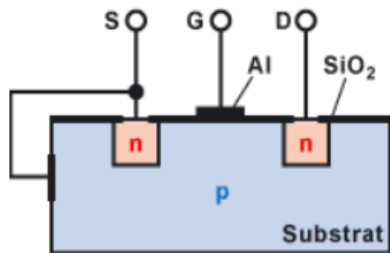


Abbildung 2-3: Aufbau eines n-Kanal-Anreicherungstyp MOSFET

Abbildung 2-3 zeigt den Aufbau eines n-Kanal-Anreicherungstyp MOSFET [6]. Sofern keine positive Spannung zwischen Gate (G) und Source (S) anliegt, befindet sich der MOSFET stets im Sperr-Zustand. Legt man nun zwischen G und S eine positive Spannung U_{GS} an, entsteht ein elektrisches Feld. Die Elektronen werden vom G-Anschluss aus dem p-leitenden Substrat (wenig Elektronen, viele Löcher) angezogen und wandern dadurch unter das Siliziumdioxid. Die Löcher wiederum wandern entgegengesetzt. Somit enthält die Zone zwischen den n-leitenden Inseln überwiegend Elektronen als freie Ladungsträger. Es befindet sich also ein n-leitender Kanal zwischen S und D. Durch die Gatespannung U_{GS} kann die

Anreicherungstyp und Verarmungstyp unterscheidet [6]. Da für die vorliegende Schaltung ein n-Kanal Anreicherungstyp (selbstsperrend) verwendet wird, soll auch nur dieser in Aufbau und Funktionsweise kurz erklärt werden. Abbildung 2-4 zeigt das Schaltzeichen eines n-Kanal MOSFET. Ein n-Kanal-Anreicherungstyp - Feldeffekttransistor besteht aus einem Substrat (p-leitender Kristall) und zwei eindotierten n-leitenden Inseln. Der Kristall ist mit einer Isolierschicht, Siliziumdioxid abgedeckt.

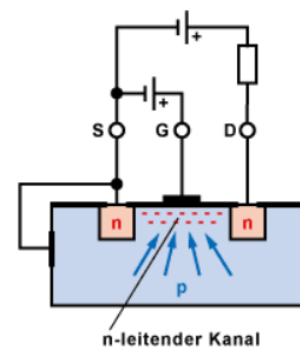


Abbildung 2-2: Funktionsweise eines n-Kanal MOSFET

Leitfähigkeit des Kanals gesteuert werden. Dabei führt eine Vergrößerung der Gatespannung zu einer Steigerung der Leitfähigkeit und umgekehrt [6]. Diese Funktionsweise ist in Abbildung 2-2 dargestellt. In diesem Projekt kann die Gatespannung mittels PWM Pin des Arduino Due zwischen 0 und 3,3 V verändert werden und somit die Spannung einer externen Quelle (USB 2.0 Port) von 0 bis 5 V eingestellt werden.

Die elektrische Schaltung können die Studenten mittels untenstehenden Schaltplans, ganz leicht auf einem kostengünstigen Protoshield inkl. Steckbrett, welches zum Arduino Due kompatibel ist, erstellen. Abbildung 2-5 zeigt ein einfaches Verbindungs-Schaubild zwischen Steckbrett und Protoshield welches mit der Software Fritzing erstellt wurde. Das fertige Protoshield inkl. aufgebauter Schaltung, muss dann nur noch auf den Arduino Due aufgesteckt werden (siehe Abbildung 2-6). Dabei sei darauf zu achten, es auch richtig herum drauf zu stecken, sodass die diversen Pin Beschriftungen zueinander passen. Die Anschlüsse des realen Micro- USB Port können eventuell von den Anschlüssen auf dem Schaubild abweichen. Dabei sollte darauf geachtet werden, die deutlich gekennzeichneten Ground (GND) - und Spannungsversorgungspunkte (5V) korrekt zu verbinden. In der untenstehenden Grafik verdeutlichen rote Linien immer 5V+ und graue Linien immer GND-.

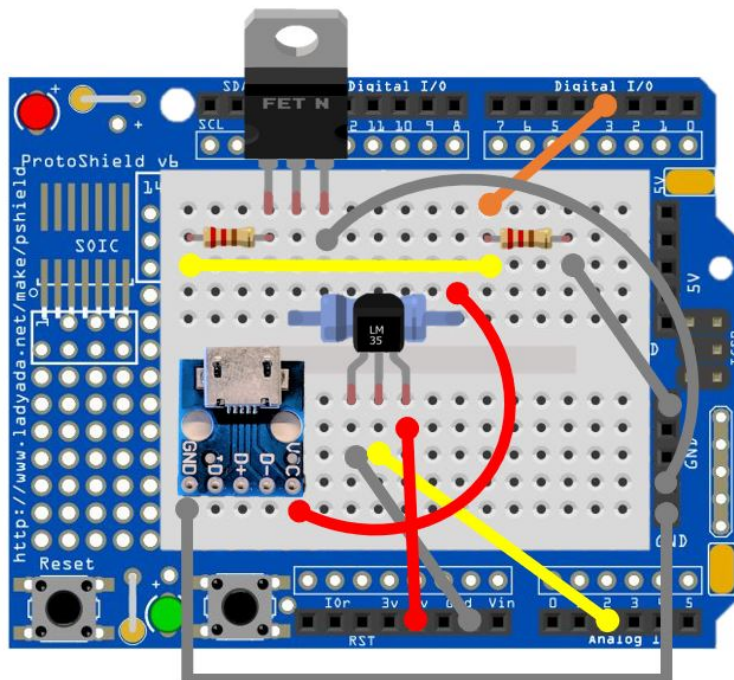


Abbildung 2-5: Verbindungs-Schaubild Schaltung

Folgende Abbildung zeigt das Schaubild mit aufgestecktem Protoshield auf dem Arduino Due [7].

2.2.3 Leistungswiderstand

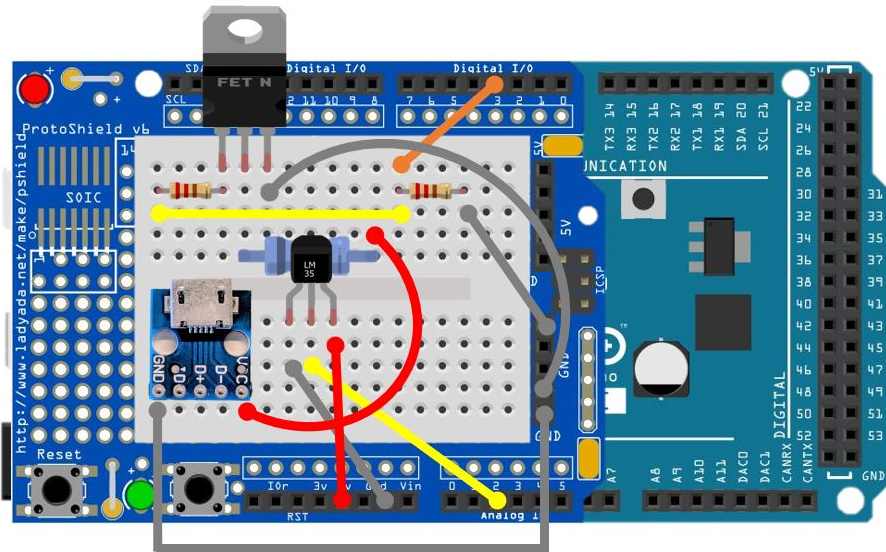


Abbildung 2-6: Verbindungs-Schaubild Protoshield mit Arduino Due [7]

Untenstehend ist noch ein Schaltplan der elektrischen Schaltung dargestellt.

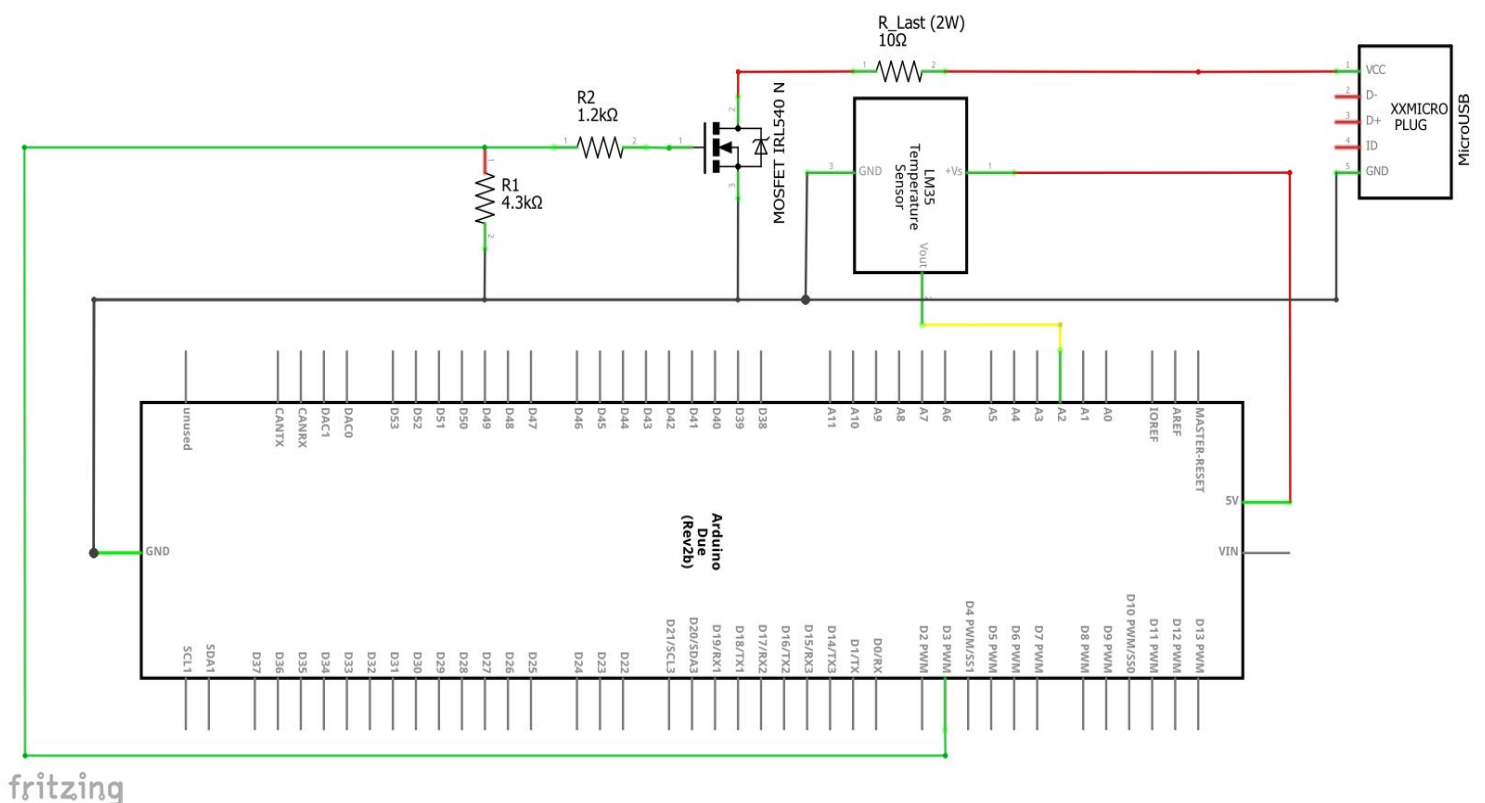


Abbildung 2-7: Platinschaltplan

2.2.3 Leistungswiderstand

Der Leistungswiderstand ist das zentrale Bauteil dieses Projektes. Denn dieser soll erwärmt und auf einer bestimmten Temperatur gehalten werden. Dabei ist dieser in Reihe im Stromkreis eingebaut und dient als Verbrauchswiderstand. Übersetzt heißt das, dass die Leistung, am Widerstand direkt in Wärme umgesetzt wird. Resultierend aus einer

deutlich kleineren extern verfügbaren Leistung (2W, USB Port) als in Projektarbeit 1 (15W, extra Netzteil), muss der Widerstand neu ausgelegt bzw. verkleinert werden, da die Zeitdauer den Widerstand um 10 K zu erwärmen, bei gleich großem Widerstand, viel zu lange dauern würde. Da aus dem USB Port nicht genau 5V und 0,5 A ausgegeben werden, sondern lediglich ca. 4,42 V und 0,44 A (mittels True RMS Messgerät nachgemessen), liegt die maximale Leistung bei 1,95 Watt. Damit der Widerstand möglichst schnell erhitzt werden kann, sollte er so ausgelegt sein, dass er bei maximal verfügbarer Leistung an das eigene Leistungsverträglichkeits-Limit stößt und gleichzeitig einen kleinen Widerstandswert aufweisen. D.h. es sollte ein Widerstand mit einer Leistungsverträglichkeit von nicht weniger oder nur geringfügig weniger als die 2 W gewählt werden. Somit wird der Widerstand auf 10 Ω und 2 W festgelegt. Zu prüfen sei nun noch ob ein Vorwiderstand als Strombegrenzer nötig ist. Gemäß der Formel $P = \frac{U^2}{R} = \frac{(4,42 V)^2}{10 \Omega} = 1,95 W$ ist diese Leistung zwar gerade so an der maximalen Leistungsverträglichkeit des Widerstandes von 2W, da aber bei späterer Nutzung nie die maximale Leistung über einen langen Zeitraum benötigt wird, ist kein Vorwiderstand notwendig. Zur Verdeutlichung hätte der Widerstand lediglich 5 Ω und 2 W wäre gemäß der Formel $P = \frac{U^2}{R} = \frac{(4,9 V)^2}{5 \Omega} = 4,8 W$ ein Vorwiderstand notwendig, da die 4,8 W deutlich höher als die 2 W des Widerstandes wären.

2.2.4 Temperatursensor



Abbildung 2-9: Temperatursensor LM35cz

Zur Messung der Widerstandstemperatur dient ein analoger Temperatursensor LM35cz, welcher auch schon in Projektarbeit 1 verwendet wurde. Der Temperaturbereich des LM35CZ geht von +0°C bis 110°C mit einer Auflösung von 10 mV/°C und einer typischen Genauigkeit von $\pm 0,4^\circ\text{C}$ bei Raumtemperatur bzw. $\pm 0,8^\circ\text{C}$ bei T_{max} . was absolut ausreichend für das Projekt ist. Das getestete Genauigkeitslimit kann bei erhöhter Temperatur bis zu $\pm 1,0^\circ\text{C}$ betragen [8]. Die Genauigkeit des Temperatursensors wird, um sicher zu gehen, noch mit einem sehr präzisen Temperatursensor verglichen und somit verifiziert. Dabei konnte festgestellt werden, dass der Sensor ab einer Temperatur von ca. 60°C um etwa 3 Grad abweicht und ab einer Temperatur von 100°C um etwa 6°C.

Damit ist der Sensor zwar nicht so genau wie es das Datenblatt beschreibt, für dieses Projekt aber immer noch im Rahmen. Abbildung 2-8 zeigt die Abweichungen bei Temperaturen von 22°C – 100°C. Die blaue Linie ist die Referenzlinie. Die rote Linie zeigt die gemessenen Temperaturwerte des LM35cz Sensors über die tatsächlichen Temperaturwerte (x-Achse) verifiziert durch den Thermonics

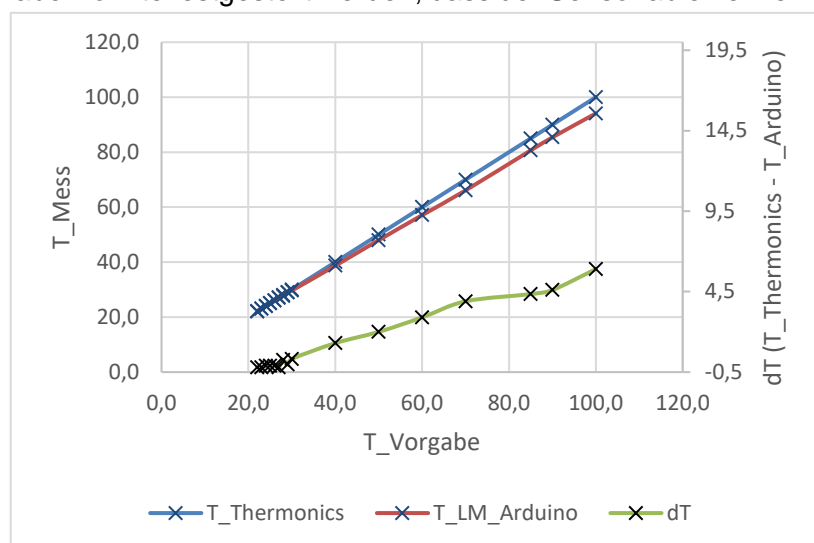


Abbildung 2-8: Verifikation des LM35cz Temperatursensors mittels eines hochpräzisen Temperatursensors

2.2.4 Temperatursensor

Temperatursensor. Zusätzlich ist noch eine grüne Linie miteingefügt, welche die Abweichung von Thermionics Sensor zu LM35cz zeigt. Mit steigender Temperatur nimmt die Abweichung zu.

3 Hard- und Softwaretechnische Umsetzung

3.1 Stückliste für den Hardware Aufbau

Nr	Bezeichnung	Anzahl	Preis	Link
1	Arduino Due R3 SAM3X8E 32-bit ARM Cortex-M Control Board Module +Cable New	1	13,42 €	Arduino Due
2	Protoshield	1	6,29 €	Shield
3	LM35cz Temperatur Sensor	1	3,35 €	LM35cz
4	USB auf Micro USB Kabel	1	1,15 €	USB / Micro USB
5	Micro USB port	1	3,97 €	Micro USB Breakboard
6	Leistungswiderstand Panasonic ERG-2SJ100 10R2W	1	0,35 €	Widerstand
7	4,3 k Widerstand	1	0,10 €	
8	1,2 K Widerstand	1	0,10 €	
9	Gehäuse	1	21 €	Gehäuse
10	Verbindungskabel	1	3,89 €	Kabel
11	MOSFET IRL 540N	1	0,70 €	Mosfet irl 540N
12	Ventilator	1	14,03€	Ventilator
13	Matlab 2018a Lizenz	1	-	
14	SIMULINK Support Package for Arduino Hardware	1	-	
15	SIMULINK Modell	1	-	
Summe			68,3 €	

3.2 Ansteuerung des MOSFET

Die Ansteuerung des in Kapitel 2.2.2 verwendeten n-Kanal MOSFETs ist von entscheidender Bedeutung, da die Leistung bzw. das PWM Signal später das Stellsignal für Steuerung und Regelung ist. Damit das PWM Signal allerdings angesteuert werden kann,

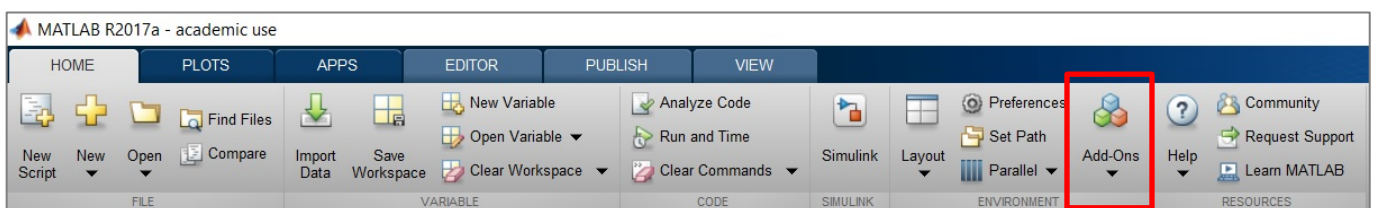


Abbildung 3-1: Matlab Add-Ons

muss zunächst das Simulink Support Package for Arduino Hardware in Matlab installiert werden. Dazu öffnet man Matlab und wählt im Reiter „Home“ → „Environment“ die Schaltfläche Add-Ons aus (siehe Abbildung 3-1). Anschließend nur noch nach „Simulink Support Package for Arduino Hardware“ suchen und installieren. Somit erhält man Zugriff auf diverse Simulink Blöcke. Siehe Abbildung 3-2.

3.2 Ansteuerung des MOSFET

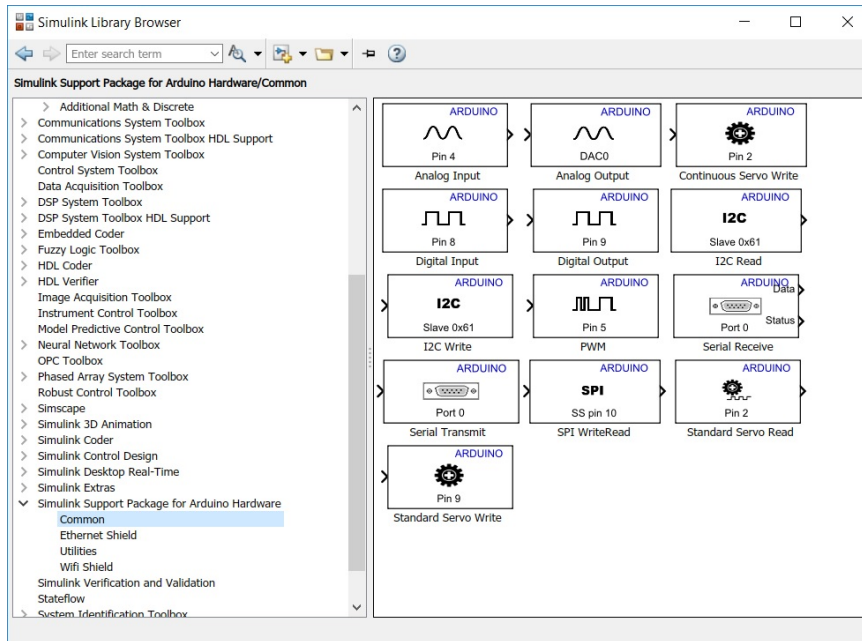


Abbildung 3-2: SIMULINK Library Browser

Unter anderem erscheint nun im Simulink Library Browser der Baustein PWM. Unter PWM (Pulsweitenmodulation) versteht man ein Rechtecksignal mit konstanter Periodendauer, welches zwischen zwei Spannungspegeln oszilliert. Im Falle des Arduino Due ist der untere Spannungspegel 0 V und der obere Spannungspegel 3,3 V. Grundsätzlich wird das Signal in schneller Folge ein- und ausgeschaltet. Das Verhältnis von Einschaltzeit zu Ausschaltzeit kann dabei variiert werden und bildet den Duty-Cycle (Tastverhältnis), siehe Abbildung 3-3. Durch Ein- und Ausschaltverhältnis bildet sich ein mittlerer Spannungswert, welcher direkt vom Duty-Cycle DC mit der Formel

$$U_m = V_{cc} * \frac{t_{ein}}{t_{ein} + t_{aus}} = V_{cc} * DC \quad 1$$

abhängig [9].

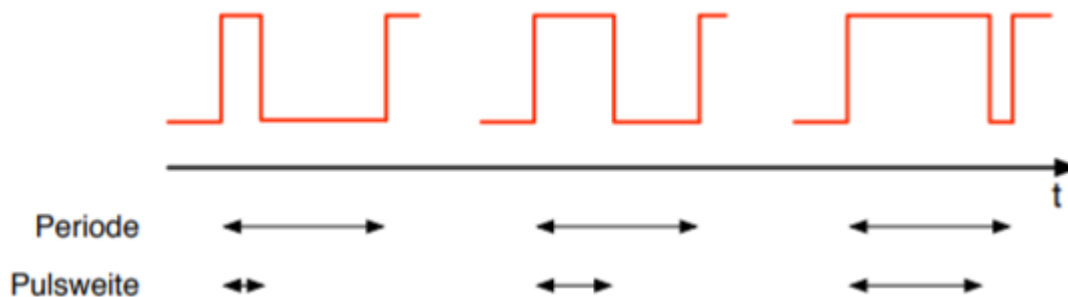


Abbildung 3-3: Pulsweitenmodulation [9]

Ist das Tastverhältnis also 0,5 (= 50%), so stellt sich ein linearer Mittelwert von $0,5 * V_{cc}$ ein. Für den Simulink PWM Block sind Werte von 0 bis 255 zulässig. Dabei bedeutet der Wert 255, einen DC von 1, also die gesamte Periodendauer high. Der Wert 0 bedeutet dagegen, dass das PWM Signal über die gesamte Periodendauer 0 ist und somit keinerlei Leistung ausgegeben wird.

Da bedingt durch das PWM Stellsignal eine Mischspannung, also eine Überlagerung aus Gleich- und Wechselspannung, vorliegt, ist im Folgenden eine Herleitung der effektiven Spannung und Leistung, welche zur Widerstandserwärmung führt gezeigt.

Für periodische Spannungen mit der Periode T gelten folgende Definitionen:

Linearer Mittelwert:
$$\bar{u} = U_{DC} = \frac{1}{T} \int_0^T u(t) dt \quad 2$$

Gleichrichtwert:
$$|\bar{u}| = \frac{1}{T} \int_0^T |u(t)| dt \quad 3$$

Effektivwert:
$$u_{eff} = \sqrt{\frac{1}{T} \int_0^T u^2(t) dt} \quad 4$$

Effektivwert des Wechselanteils:
$$U_{AC} = \sqrt{\frac{1}{T} \int_0^T [u(t) - U_{DC}]^2 dt} \quad 5$$

Effektivwert bei Überlagerung von Gleich- und (reiner) Wechselspannung:

$$u_{eff} = \sqrt{U_{AC}^2 + U_{DC}^2} \quad 6$$

Wendet man die oben aufgeführten Gleichungen auf unseren Fall eines PWM Signals an, so leitet sich die Leistung, welche den Widerstand erwärmt wie folgt her:

$$U_{eff} = \sqrt{\frac{1}{T} \int_0^T u^2(t) dt} \quad 7$$

$$P = U_{eff} * I_{eff} = \frac{(U_{eff})^2}{R} \quad 8$$

Nachstehende Grafik zur Verdeutlichung des PWM Signals:

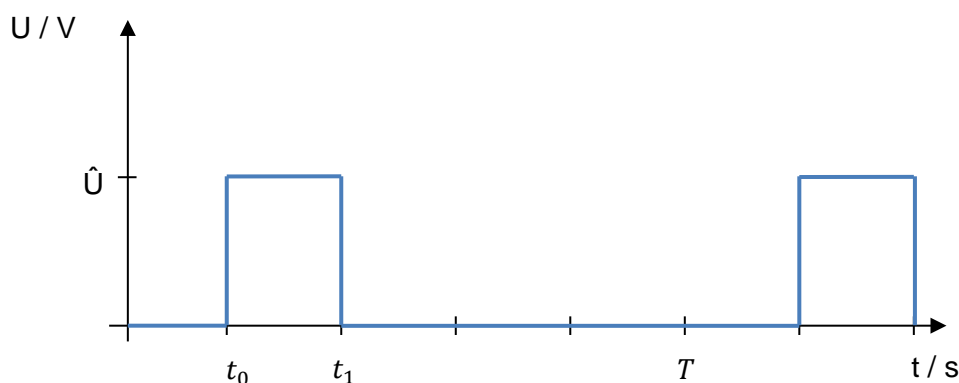


Abbildung 3-4: PWM-Signal

$$\begin{aligned} U_{eff} &= \sqrt{\frac{1}{T} \left(\int_{t_0}^{t_1} u^2(t) dt + \int_{t_1}^T u^2(t) dt \right)} \\ &= \sqrt{\frac{1}{T} \left(\int_{t_0}^{t_1} \hat{u}^2(t) dt + \int_{t_1}^T 0 dt \right)} \end{aligned}$$

3.2 Ansteuerung des MOSFET

$$= \sqrt{\frac{1}{T} * \hat{u}^2 * (t_1 - t_0)} = \sqrt{\frac{t_1 - t_0}{T} * \hat{u}^2}$$

Damit ist die effektive Leistung, welche den Widerstand erwärmt:

$$P_{eff} = \frac{U_{eff}^2}{R} = \frac{\frac{t_1 - t_0}{T} * \hat{u}^2}{R}$$

9

Das T steht zwar für die Periodendauer, da im Falle des PWM Bausteins in Simulink allerdings der Wert 255 simultan für die gesamte Periodendauer auf „high“ steht, kann T durch den Wert 255 ersetzt werden. Dementsprechend ist $t_1 - t_0$ die Zeit in der das PWM-Signal auf „high“ steht und somit nichts anderes als ein Wert zwischen 0 und 255. Ein PWM Wert von 127,5 bedeutet somit ein Tastverhältnis von $\frac{127,5-0}{255} = 0,5$ und somit eine effektive Leistung von $P = \frac{U_{eff}^2}{R} = \frac{0,5 * (U_{max=4,42V})^2}{10\Omega} = 0,97 W$. Abbildung 3-5 vergleicht die berechnete Leistung der aus Gleichung 9 mit der gemessenen Leistung sowie die verschiedenen Gleich- und Wechselspannungen. Zur Messung der Leistung wurden mit einem True RMS Messgerät zu verschiedenen PWM Tastverhältnissen die Gleichspannung U_{DC} sowie die zugehörige Wechselspannung U_{AC} gemessen und durch Gleichung 6 und Gleichung 9 die effektive Leistung berechnet. Ein Blick auf Abbildung 3-5 verdeutlicht zwar den linearen Zusammenhang zwischen effektiver Leistung und PWM, es ist aber dennoch eine kleine Abweichung von berechneter zu gemessener Leistung ersichtlich. Somit muss die berechnete Leistung noch an die gemessene Leistung angepasst werden.

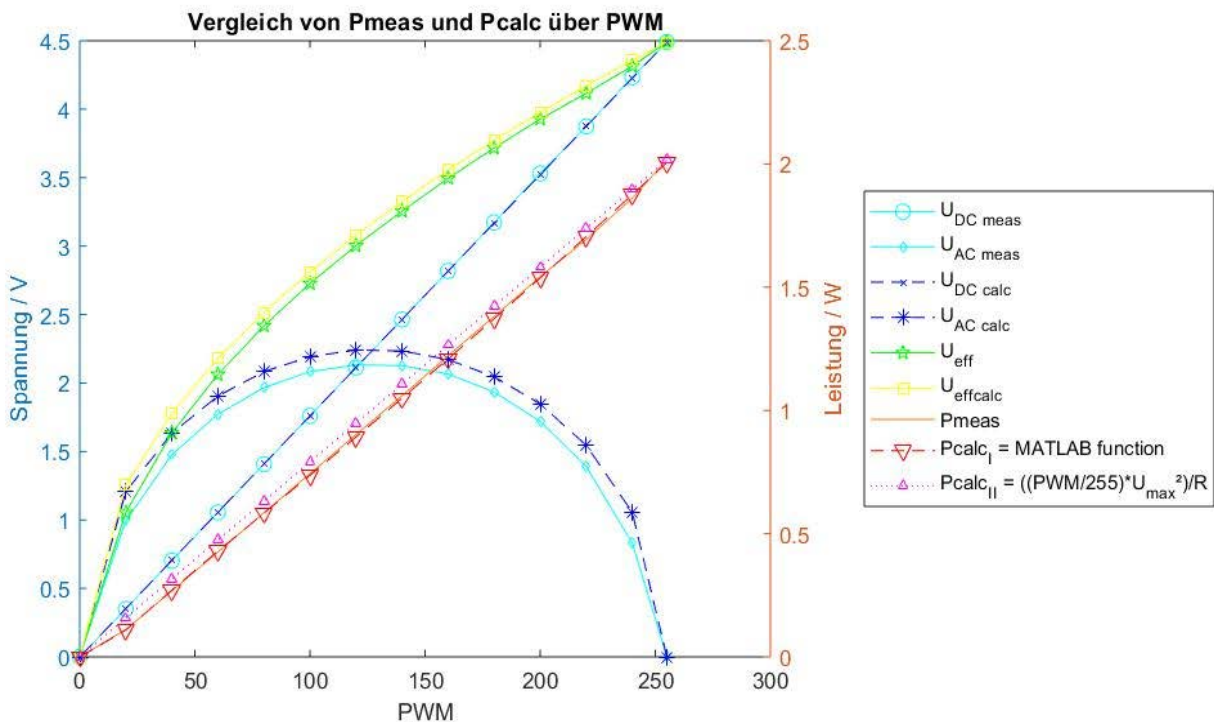


Abbildung 3-5: Vergleich von gemessener mit berechneter Leistung und Spannung

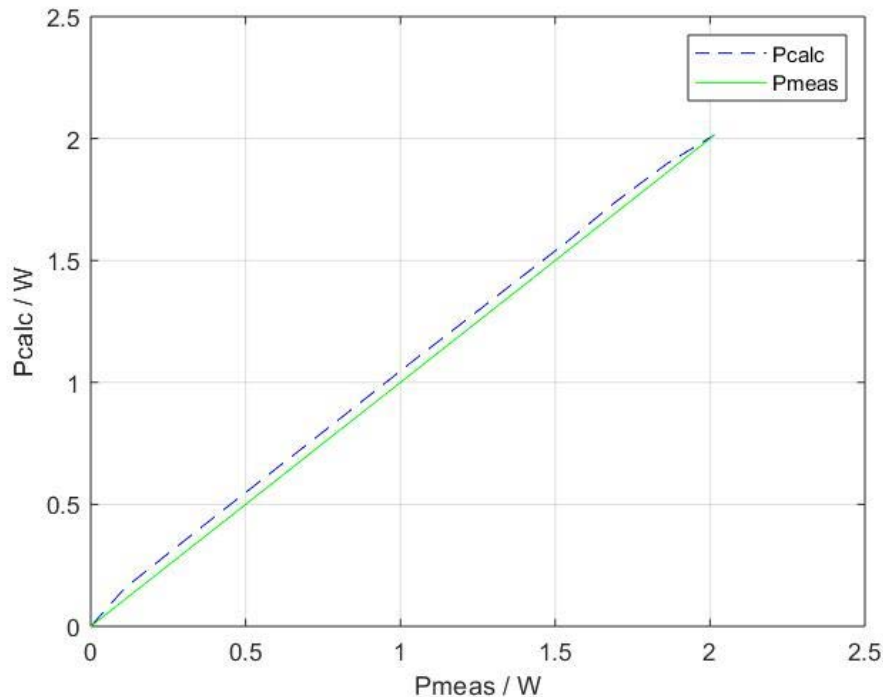


Abbildung 3-6: Pcalc über Pmeas

In Abbildung 3-6 sind die berechneten Leistungswerte P_{calc} über die tatsächlich gemessenen Werte P_{meas} dargestellt (blaue Linie). Mittels einer Lookup Table in Simulink werden die berechneten Leistungswerte P_{calc} in die tatsächlich gemessenen Leistungswerte P_{meas} umgewandelt. Wichtig zu erwähnen ist, dass diese Umrechnung nur für eine konstante maximale Spannung von 4,4V gültig ist. Da die maximale Spannung allerdings abhängig von verschiedensten Faktoren wie z.B. Akkubetrieb des Laptops, Netzbetrieb des Laptops, etc. nicht immer exakt gleich bzw. sogar zwischen ca. 4,6 und 4,0 Volt variieren kann, ist es möglich, dass trotz Anpassung der Leistungswerte mittels Lookup Table Abweichungen in der Stellgröße vorkommen können. Dies hat zur Folge, dass ein PWM Signal von 255, in einem Zustand, einer Leistung von z.B. 2,1 Watt entspricht und im nächsten Zustand könnte es sein, dass ein PWM von 255 einer Leistung von lediglich 1,7 Watt entspricht. Somit führen gleiche PWM Werte zu unterschiedlichen Stellgrößen und folglich zu unterschiedlichen Temperaturen. Man stelle sich vor, die experimentelle Bildung der Übertragungsfunktion wird bei einer maximalen Leistung von 2 Watt durchgeführt und am nächsten Tag liefert der Port aus diversen Gründen lediglich eine maximale Leistung von 1,8 Watt. Ist dies der Fall, beschreibt die Übertragungsfunktion natürlich nicht mehr exakt das Übertragungsverhalten von Eingang zu Ausgang, da die Stellgröße unterschiedlich groß ist. Dies gilt es zu berücksichtigen.

Abbildung 3-7 zeigt die Ansteuerung des ausgewählten Arduino PWM Pin 3 in Simulink. Da der PWM Pin nur Werte zwischen 0 und 255 verwendet, muss das Stellsignal welches durch den Regler berechnet wird (Leistung in Watt) wieder in einen Wert zwischen 0 und 255 umgewandelt werden. Durch umstellen von Gleichung 9 kann dies ermöglicht werden.

3.2 Ansteuerung des MOSFET

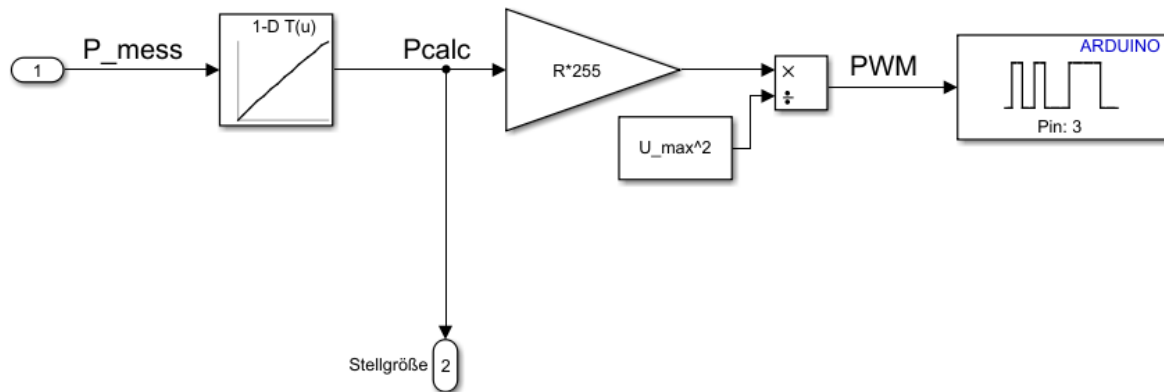


Abbildung 3-7: Ansteuerung des PWM Blocks in SIMULINK

Die Default Einstellungen für den PWM Pin führen dazu, dass der Pin seinen Wert nach stoppen des Simulink Modells beibehält und das System somit weiter geheizt wird. Auf Grund dessen muss noch ein bestimmtes MATLAB File ausgetauscht werden. An dieser Stelle gilt ein großer Dank an den MATLAB Support Service, welcher mir bei diesem Problem helfen konnte. Dabei ist folgendermaßen vorzugehen:

Das File "arduino_PWMOutput.p" muss das gleichnamige File in folgendem Ordner ersetzen: C:\ProgramData\MATLAB\SupportPackages\R2018a\toolbox\target\supportpackages\arduino\base\+codertarget\+arduino\base\+internal\

Zu Sicherheit sollte davor das bereits existierende File in einem beliebigen anderen Ordner, z.B. Desktop, gesichert werden.

Nach erfolgreichem Ersetzen des "arduino_PWMOutput.p" File, müssen folgende Befehle im Matlab command window ausgeführt werden.

1. clear pcode
2. clear classes
3. rehash toolboxcache
4. Run the model

Nun wird der Widerstand nach beenden des Simulink Modells nicht weiter geheizt. Es müssen somit keine Kabel abgesteckt werden, sondern können durchgehen angesteckt bleiben.

Zusätzlich zum Simulink Support Package for Arduino Hardware müssen noch auf gleiche Art und Weise wie oben beschrieben folgende Add-Ons installiert werden, damit die späteren Simulink Modelle reibungslos funktionieren:

- Curve Fitting Toolbox (Version 3.5.7)
- System Identification Toolbox (Version 9.8)
- MATLAB Support Package for Arduino Hardware (Version 18.1.1)
- MATLAB Support for MinGW-w64 C/C++ Compiler (Version 18.1.0)
- Embedded Coder (Version 7.0)

3.3 Temperaturmessung

Im Gegensatz zur Leistung, welche die Eingangsgröße ist, dient der Temperatursensor dem Messen der Ausgangsgröße. Wie in Kapitel 2.2.3 2.2.4 dargestellt, wird für das Projekt ein LM35cz analog Temperatursensor verwendet. Der Vorteil liegt darin, dass es vorgefertigte Simulink Blöcke im Support Package for Arduino Hardware gibt, mit denen man analoge Eingangssignale messen kann. Der digitalisierte Spannungswert muss anhand der DAC-Auflösung und der Angaben des Temperatursensordatenblatts umgerechnet werden, um die resultierende Temperatur zu erhalten. In diesem Fall wird der gemessene Wert mit dem A/D Wandler Wert 1023 (da 10-Bit Auflösung) dividiert, mit der Arduino Due Betriebsspannung von 3,3 V multipliziert und durch den Auflösungsfaktor von 10mV/°C ($\cong 0,01 \text{ V}/^\circ\text{C}$) dividiert. Abbildung 3-8 zeigt das Simulink Model zur Temperaturmessung.

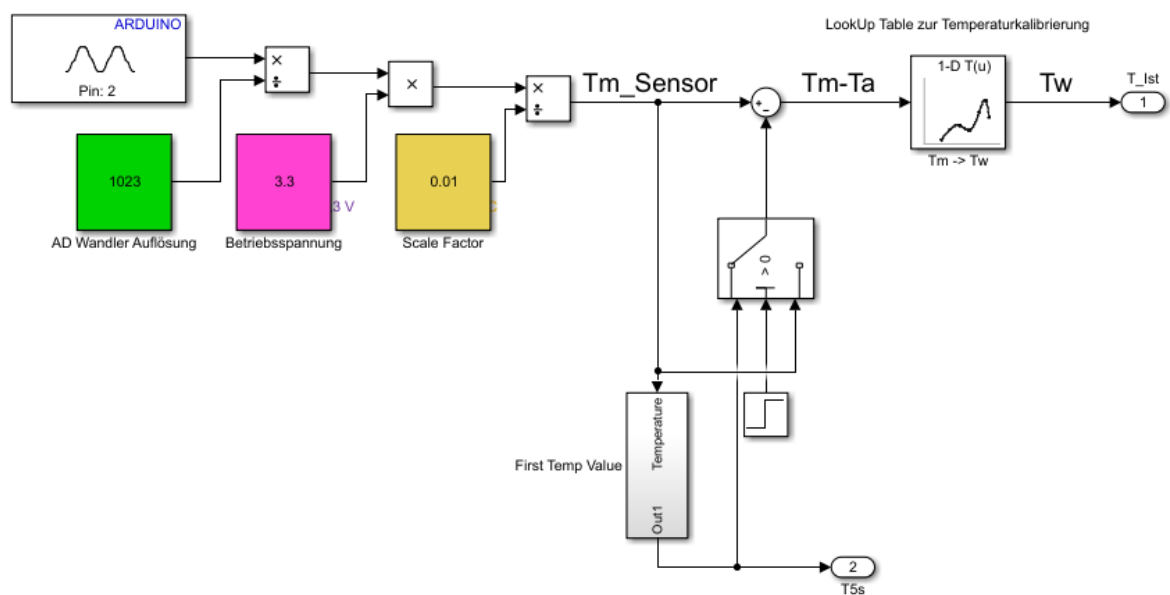


Abbildung 3-8: Temperaturmessung in Simulink

Der gemessene Anfangstemperaturwert wird nach erfolgreicher Umrechnung wieder vom Messwert abgezogen damit man bei $dT = 0$ startet. Mittels lookup Table wird der gemessene Temperaturwert $T_m - T_a$ noch in einen Kalibrierten Wert T_w umgewandelt. Der Grund dafür ist in Kapitel 3.4 erläutert.

3.4 Experimentelle Bildung der Übertragungsfunktion mit pzMove

In diesem Kapitel wird die experimentelle Bildung der Übertragungsfunktion von Eingang $u(t)$ Leistung zu Ausgang $y(t)$ Temperatur beschrieben. Dabei wird der zu erwärmende Widerstand mit einer konstanten Leistung erwärmt und der entsprechende Temperaturverlauf aufgezeichnet. Die Übertragungsfunktion wird anschließend mittels pzMove [10] erstellt, wobei lediglich eine Matrix (Zeit, Leistung, Temperatur) der Messdaten in das Programm eingefügt wird und mittels diverser Parametereinstellungen automatisch eine Übertragungsfunktion berechnet wird (siehe Abbildung 3-10). Wichtig dabei ist, dass von den gemessenen Temperaturwerten noch die aktuelle Raumtemperatur, welche der Anfangstemperatur entspricht abgezogen werden muss, damit sich das System in Ruhelage befindet und bei null Input auch null Output resultiert [$U(0) = Y(0)$].

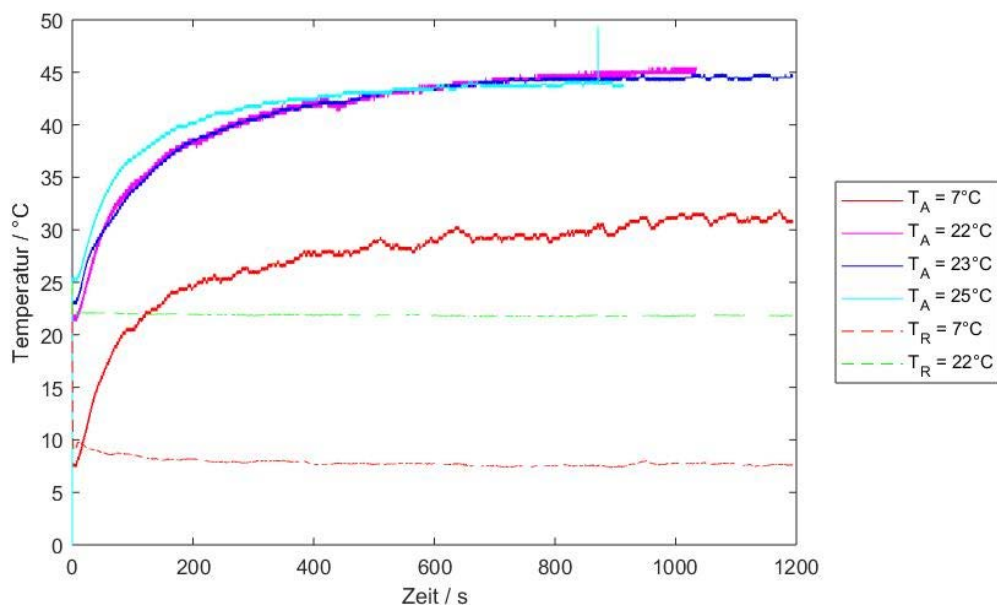


Abbildung 3-9: Vergleich von vier Sprungantworten bei gleichem Input und unterschiedlichen Anfangs- und Raumtemperaturen

Abbildung 3-9 zeigt einen Vergleich von vier verschiedenen Sprungantworten. Die Rote Temperaturkurve wurde draußen bei einer „Raumtemperatur“ von ca. 7°C in einer Windgeschützten Umgebung aufgenommen. Alle anderen Linien wurden bei 22°C Raumtemperatur aufgenommen. Die Magenta farbene Linie wurde aus dem thermodynamischen Gleichgewicht heraus gestartet (Anfangstemperatur = Raumtemperatur). Die blaue Linie wurde bei einer um 1°C erhöhten Anfangstemperatur aufgenommen und die Cyan farbene Linie bei einer um 3°C erhöhten Anfangstemperatur. Zudem wurde mit einem extra Temperatursensor die tatsächliche Raumtemperatur gemessen. Mit diesem Versuch wird untersucht, inwiefern sich eine Abweichung der Anfangstemperatur von der Raumtemperatur auf die stationäre Verstärkung der Übertragungsfunktion auswirkt.

Nach Aufnahme der Temperaturverläufe bei exakt gleicher Leistung werden aus den Daten mittels pzMove die resultierenden Übertragungsfunktionen von Input (Leistung) zu Output (Temperatur) berechnet und miteinander verglichen.

Im Folgenden sind die dazu gehörigen Übertragungsfunktionen dargestellt:

3.4 Experimentelle Bildung der Übertragungsfunktion mit pzMove

- TA = 7°C: $GM = \frac{38.652 \cdot (1 + 198.5s)}{(1 + 351.6s)(1 + 1.852 \cdot (19.71s) + (19.71s)^2)}$
- TA = 22°C: $GM = \frac{40.973 \cdot (1 + 330.6s)}{(1 + 9.655s)(1 + 59.66s)(1 + 510.8s)}$
- TA = 23°C: $GM = \frac{35.46 \cdot (1 + 176.7s)}{(1 + 7.875s)(1 + 59.17s)(1 + 293.4s)}$
- TA = 25°C: $GM = \frac{31.36 \cdot (1 + 221.4s)}{(1 + 6.181s)(1 + 52.78s)(1 + 302.6s)}$

Betrachtet man die unterschiedlichen stationären Verstärkungen (sV) der jeweiligen Übertragungsfunktionen, so wird ersichtlich, wie signifikant unterschiedlich die Verstärkungen schon bei geringfügig verschiedenen Anfangstemperaturen sind. Weicht die Anfangstemperatur nämlich schon um 1°C von der Raumtemperatur ab, so ergibt sich anstatt einer sV von 40,9 eine sV von 35,5. Ist die Anfangstemperatur um 3°C höher als die Raumtemperatur, so ergibt sich eine sV von 31,4, was beinahe 25% weniger Verstärkung entspricht. Selbst im Thermodynamischen Gleichgewicht ergibt sich ein kleiner Unterschied der sV zwischen der Übertragungsfunktion bei 7°C und der bei 22°C.

Es ist also fundamental, dass sowohl die Übertragungsfunktion als auch die Modelle im Späteren Verlauf für Steuerung, Steuerung mit GW und Regelung im thermodynamischen Gleichgewicht gestartet werden. Ich empfehle deshalb, den Widerstand vor der Messung ca. 5 Minuten mit einem Ventilator zu belüften, damit er möglichst schnell Raumtemperatur annimmt. Anschließend sollte mindestens 10 Minuten gewartet werden, bis ein Modell durchgeführt wird.

Abbildung 3-10 zeigt einen Ausschnitt aus pzMove. Dabei sollten folgende Einstellungen bei jeder Berechnung vorgenommen werden.

- Selected end time limit (te): 1500 Sekunden
- Numerator: 1
- Denominator: 3
- Assumed Initial Conditions: Zero
- Filter Solutions: Only Stable
- Gewichtung: 0-300 Sekunden auf 200, dann 1

3.4 Experimentelle Bildung der Übertragungsfunktion mit pzMove

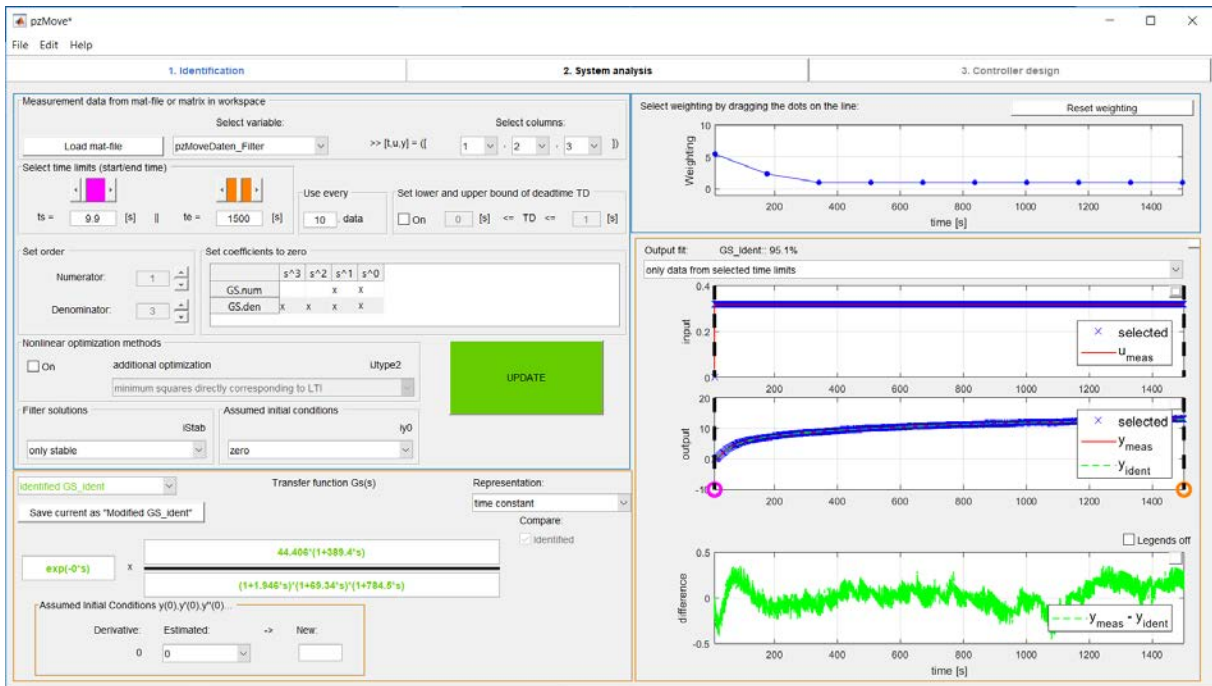


Abbildung 3-10: Experimentelle Bildung der Übertragungsfunktion mit pzMove

Da in der vorliegenden Arbeit ein lineares zeitinvariantes System angenommen wird, müsste die Übertragungsfunktion bei egal welchem Input-Wert, also bei egal welcher Leistungseingabe, immer dieselbe sein bzw. dürften Unterschiede lediglich auf diverse Ein- und Ausgangsstörungen sowie auf Messgenauigkeiten der Sensoren resultieren.

Abbildung 3-11, zeigt eine Übersicht der stationären Temperaturänderung dT bei unterschiedlichen Leistungsinputs. Dabei wurde der Widerstand jeweils solange mit einer konstanten Leistung erwärmt, bis sich eine stationäre Temperatur am Widerstand eingestellt hat. Jede Messung wurde im Thermodynamischen Gleichgewicht gestartet.

Auf den ersten Blick sieht der Plot für dT_m (Messtemperatur) wie eine gerade aus. Dies bedeutet, dass das Verhältnis von Temperaturänderung zu Eingangsleistung über den gesamten Inputbereich annähernd konstant ist und somit ein linearer Zusammenhang zwischen Temperaturänderung und Leistung besteht.

Vergleicht man das gemessene Verhalten mit den Gesetzen der Thermodynamik bzw. mit dem Stromwärme Gesetz, wonach sich die Temperatur eines Körpers mit einer bestimmten Fläche A und dessen Wärmeübergangskoeffizient α proportional zur aufgenommenen elektrischen Leistung ändert,

$$\Delta\vartheta = \frac{P}{\alpha \cdot A} \quad [11]$$

so scheinen die Ergebnisse aus Messung und Theorie in Bezug auf Linearität übereinzustimmen. Betrachtet man die Kurve allerdings etwas genauer so ist ersichtlich, dass diese Kurve nicht exakt linear ist. Sie scheint mehr einer quadratischen Funktion zu ähneln.

3.4 Experimentelle Bildung der Übertragungsfunktion mit pzMove

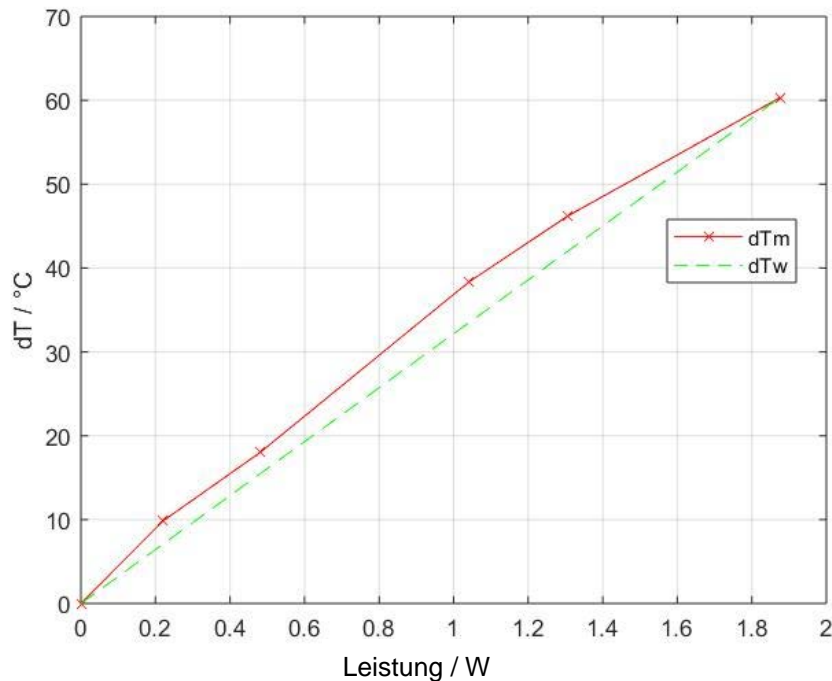


Abbildung 3-11: stationäre Temperaturänderung bei verschiedenen effektiver Eingangsleistung (Systemruhelage)

Die Abweichung der roten Linie zu einer exakten Nullpunkt-Halbgerade führt zu den in Abbildung 3-12 dargestellten unterschiedlichen stationären Verstärkungen (rote Linie).

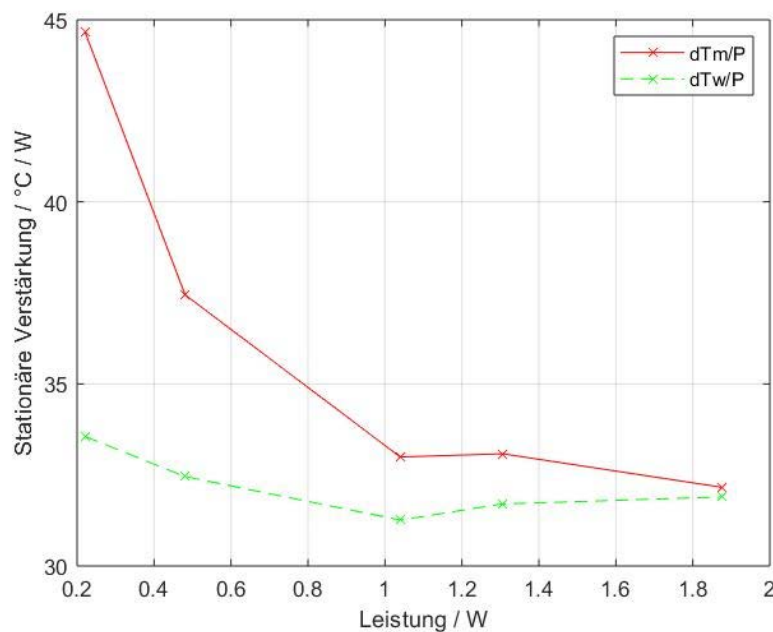


Abbildung 3-12: Stationäre Verstärkung bei steigendem Leistungsinput

Damit über den gesamten Input-Bereich eine einzige Übertragungsfunktion gültig ist, wird im Folgenden die Ausgangsgröße, sprich Temperaturänderung, so kalibriert, dass Messwerte die der roten Linie in Abbildung 3-11 entsprechen, zu Werten umgewandelt werden, welche auf der grünen Linie liegen. Dazu wird die rote Linie (dTm) über die grüne Linie (dT_w) geplottet und eine Lookup Table im Modell verwendet, welche die Sensor-Messwerte (rote Linie) in die grüne Linie umwandelt.

3.4 Experimentelle Bildung der Übertragungsfunktion mit pzMove

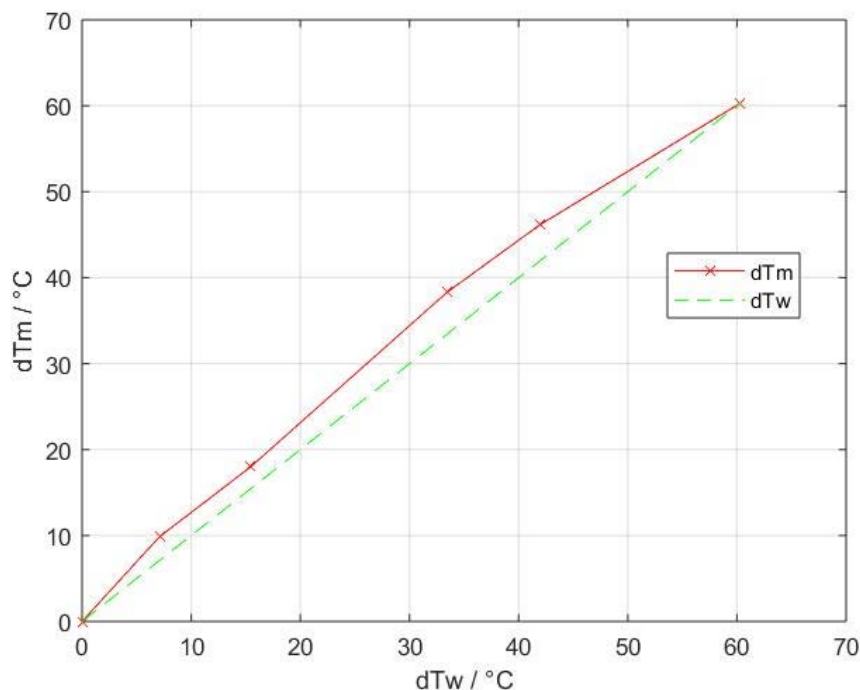


Abbildung 3-13: dTw über dTm

Da der rote Plot in Abbildung 3-13 einer quadratischen Funktion ähnelt, werden die Zwischenpunkte aus den Messwerten nicht linear interpoliert, sondern kubisch. Wird nun die Übertragungsfunktion erneut für verschiedene Leistungswerte aufgenommen ergibt sich die grüne Linie aus Abbildung 3-12. Die stationäre Verstärkung ist nun für alle Inputgrößen annähernd gleich groß. Unterschiede resultieren aus Sensorungenauigkeiten bzw. Stellgrößenungenauigkeiten.

Einzige Ungenauigkeit ist nun noch die maximale Spannung U_{max} . Da wir ausgehend von dieser Größe die Leistung berechnen, sei zu untersuchen inwiefern sich eine Abweichung von der tatsächlich anliegenden maximalen Spannung auf die Übertragungsfunktion auswirkt. Dazu werden die Variablen $U_{max1} = 4V$ und $U_{max2} = 4,3V$ verwendet. Trägt man die sich daraus resultierenden unterschiedlichen stationären Verstärkungen in die Grafik aus Abbildung 3-12, so wird ersichtlich, dass ein U_{max} -Unterschied von 0,3 V zwar durchaus eine Auswirkung auf die stationäre Verstärkung des Systems hat. Mit dem Hintergrund, dass kleinste Abweichungen der Widerstandes-Anfangstemperatur von der Raumtemperatur (Abbildung 3-9) allerdings bereits ähnliche Unterschiede hervorrufen, wird dieser Einfluss bis auf Weiteres vernachlässigt. Die eingefügte Temperaturkalibrierung reicht somit aus, um eine ausreichende Genauigkeit zu erzielen. Es ist allerdings trotzdem wichtig zu erwähnen, dass Unterschiede in der stationären Verstärkung oder in den späteren Steuerungs-Simulink Modellen nicht nur auf Sensorungenauigkeiten oder Störungen beruhen, sondern auch aufgrund der nicht konstanten maximalen Spannung resultieren können.

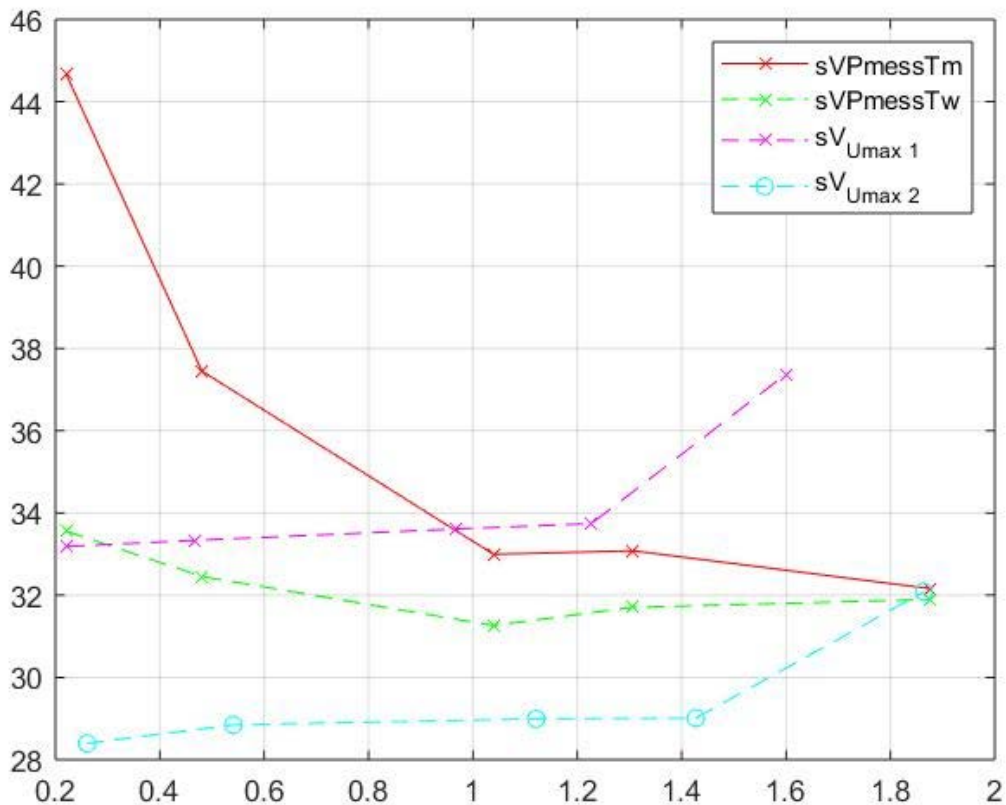


Abbildung 3-14: Vergleich der stationären Verstärkungen bei unterschiedlichen Umax

Startet man nun die Messung für die Übertragungsfunktion aus dem thermodynamischen Gleichgewicht heraus und verwendet eine konstante Leistung von ca. 0,58 Watt, ergibt sich mit Einbeziehung der beschriebenen Temperaturkalibrierung, eine für alle Leistungsinputs gültige Übertragungsfunktion mit drei Polen und einer Nullstelle. Die stationäre Verstärkung beträgt 36,7. Unten dargestellt ist die Übertragungsfunktion in Zeitkonstanten Darstellung.

$$GM = \frac{36.752 \cdot (1 + 176.6s)}{(1 + 7.621s)(1 + 50.07s)(1 + 405.9s)} \quad \text{Zeitkonstanten Darstellung}$$

Alle weiteren Modelle werden mit exakt dieser Übertragungsfunktion durchgeführt.

Zum Ausführen der experimentellen Bildung der Übertragungsfunktion muss lediglich der Hardwareaufbau mit den USB Ports eines Laptops verbunden werden und das MATLAB-Skript „Execute_Transferfunction.m“ ausgeführt werden.

3.5 Messung und Berechnung des Frequenzgangs

Der Frequenzgang beschreibt die Übertragungseigenschaft eines dynamischen Systems für sinusförmige Eingangsgrößen, wobei nur das stationäre Verhalten berücksichtigt wird [12]. Für das stationäre Verhalten y_s erhält man mit der Zerlegung des komplexen Frequenzganges in der Form

$$G(j\omega) = |G(j\omega)| \cdot e^{j\varphi(j\omega)}, [12]$$

3.5 Messung und Berechnung des Frequenzgangs

wobei $|G(j\omega)|$ die Amplitude und $\varphi(j\omega)$ die Phase des Frequenzganges bezeichnet, die Beziehung

$$y_s = |G(j\omega)| \cdot u \cdot \sin(\omega t + \varphi_u + \varphi(j\omega)), \quad [12]$$

Dabei sind u und φ_u Parameter der Eingangsgröße und $G(j\omega)$ sowie $\varphi(j\omega)$ Parameter des Systems, die von der Frequenz ω abhängig sind [12].

Der Frequenzgang kann nicht nur durch Aufnahme einer Übertragungsfunktion, wie im vorherigen Kapitel beschrieben, berechnet werden, sondern auch für ein gegebenes unbekanntes System experimentell bestimmt werden, ohne die Übertragungsfunktion zu kennen. Dabei legt man für eine bestimmte Kreisfrequenz ω die sinusförmige Eingangsgröße an das System an und wartet, bis das Übergangsverhalten abgeklungen ist, sodass sich die Amplitude der Ausgangsgröße von Periode zu Periode nicht mehr ändert. Wird nun das Eingangs- und Ausgangssignal in dasselbe Diagramm geplottet, kann man den Maximalwert von $y(t)$ ablesen und berechnet daraus den Betrag des Frequenzganges (Amplitudengang) entsprechend der Gleichung $|G(j\omega)| = \frac{|Y|}{|U|}$. Zur Bestimmung der Phase φ_y vergleicht man die Zeitpunkte gleichartiger Nulldurchgänge der Eingangs- und Ausgangsgrößen. Hat u einen Nulldurchgang bei T und y seinen nächsten Nulldurchgang bei T' , so gilt $\varphi_y = \omega(T - T')$ (Vorsicht bei positiver Phasenverschiebung bzw. bei $\varphi_y > 360^\circ$!) [12].

Als Beispiel ist in Abbildung 3-15 die Frequenzgangmessung für $\omega = 0.05 \text{ rad/s}$ dargestellt.

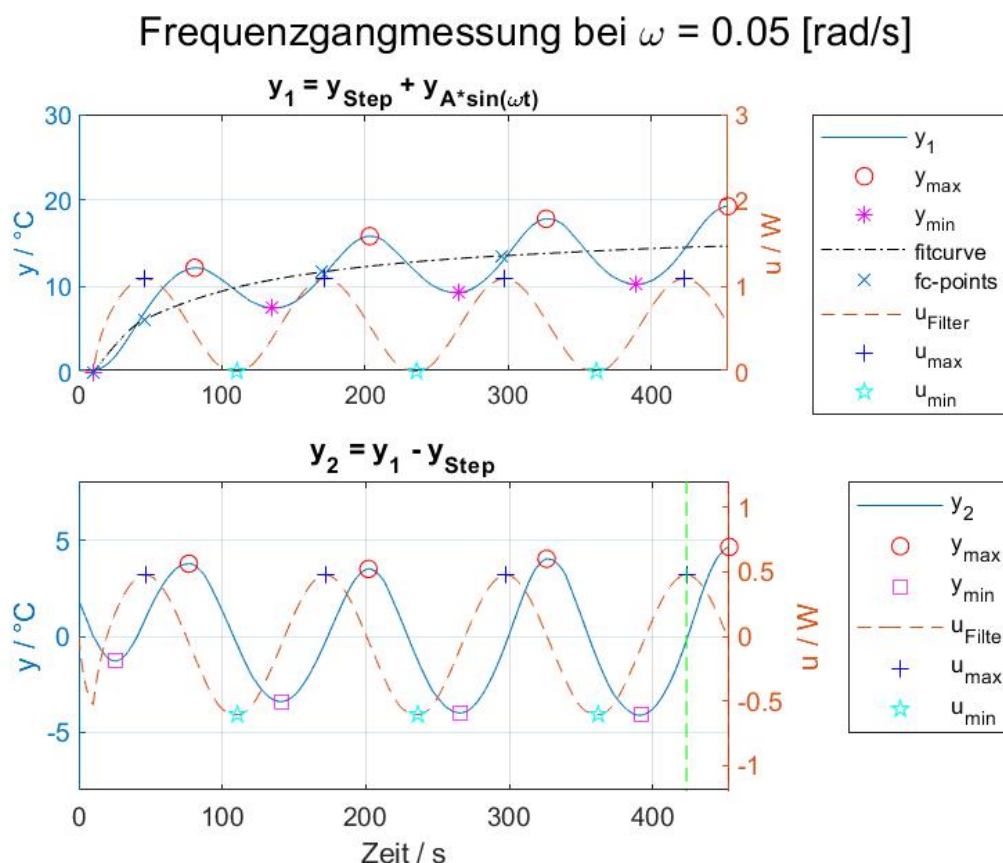
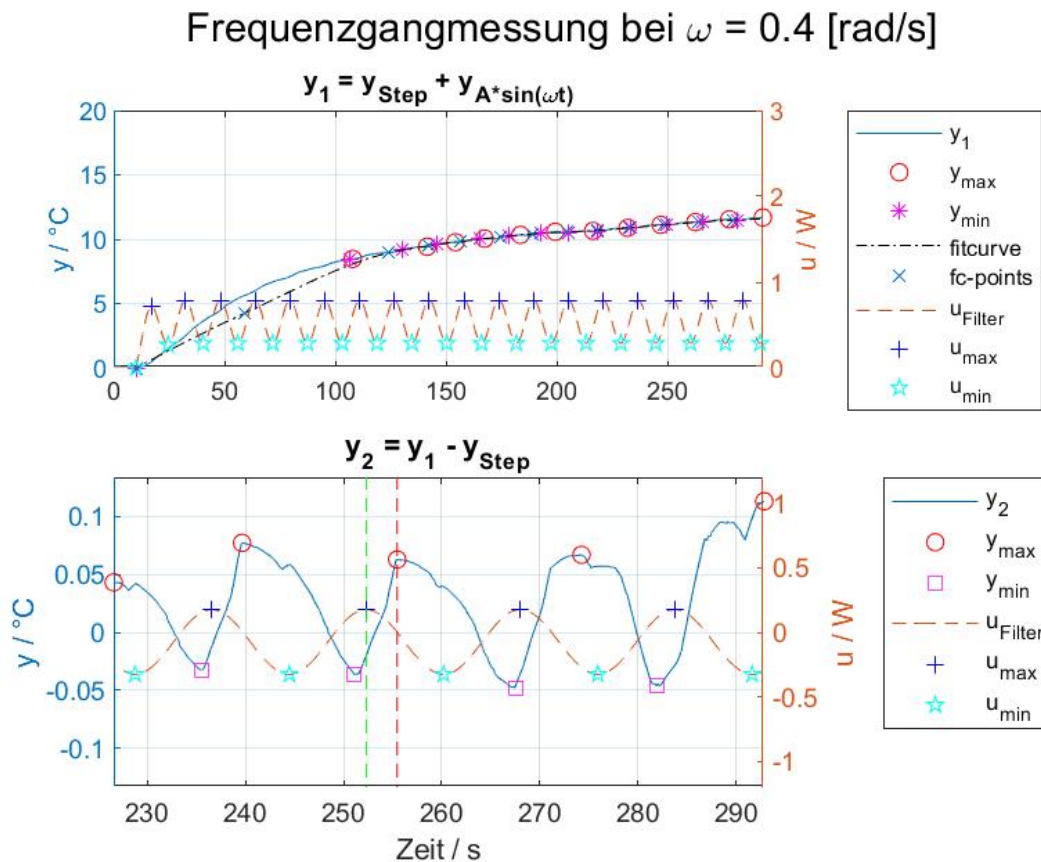


Abbildung 3-15: Frequenzgangmessung bei $\omega = 0.05 \text{ rad/s}$

Da in diesem Projekt der Widerstand lediglich erwärmt werden kann, ist eine Oszillation um den Input-Nullpunkt mit einer bestimmten Amplitude nicht möglich, da wir keine negative

Leistung einstellen können. Wie oben beschrieben wird Phase und Betrag des Frequenzganges allerdings durch ein sinusförmiges Eingangssignal (Leistung) und der daraus resultierenden stationären Antwort des Ausgangs (Temperatur) bestimmt. Deswegen nutzen wir einen kleinen Trick, indem wir den Eingang um einen definierten Step sinusförmig oszillieren lassen und anschließend die Erwärmung aufgrund des Sprungs wieder von der resultierenden Systemantwort abziehen. Übrig bleibt ein reines Sinussignal, welches um den Nullpunkt schwingt und zur Berechnung des Frequenzgangs verwendet werden kann (siehe Abbildung 3-15 unterer Plot). Y1 Des oberen Plots aus Abbildung 3-15 zeigt die Überlagerung von Sprung und Sinus. Da der Temperatursensor ein gewisses Rauschen aufweist und dies bei höheren Frequenzen dazu führt, dass die Temperatur nicht mehr genau genug gemessen werden kann, wird der Ausgang durch ein PT1-Glied gefiltert (Beispiel siehe Abbildung 3-16). Dies hat zur Folge, dass auch der Eingang durch das identische PT1-Glied gefiltert werden muss, damit Ein- als auch Ausgang gleich verzögert sind und eine genaue Phasenverschiebung berechnet werden kann. Allerdings kann die Phasenverschiebung ab einer Frequenz größer ca. 0,1 rad/s trotz Filter nicht mehr ganz exakt gemessen werden. Dafür hat der Temperatursensor eine zu geringe Messgenauigkeit (siehe Abbildung 3-16).

Die schwarze Strichpunkt-Linie „fitcurve“ ist die reine Sprungantwort des Ausgangs. Diese Linie wird durch geschicktes curve fitting erstellt und dient dazu, den unteren Plot, bei welchem der Sprung abgezogen ist, zu erstellen. Der untere Plot aus Abbildung 3-15, zeigt Ein- und Ausgang mit abgezogenem Sprung. Also die reine Sinusantwort.

Abbildung 3-16: Frequenzgangmessung bei $\omega = 0.4$ rad/s

3.6 Aufbau der Simulink Programme

Nachdem der Frequenzgang für ausreichend viele Frequenzen gemessen wurde, kann der Bodeplot erstellt werden. Jedes Kreuz in Abbildung 3-17 ist das Ergebnis einer Frequenzgangmessung, welche ca. 20 Minuten dauert.

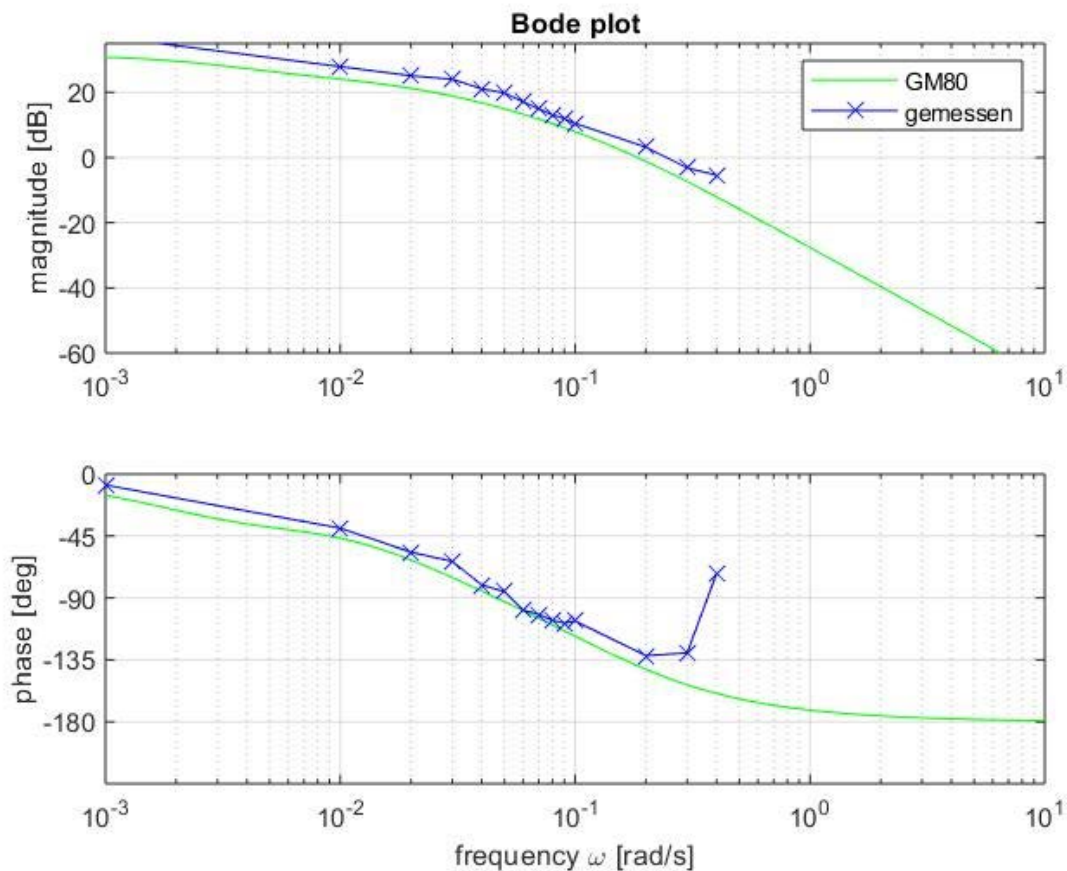


Abbildung 3-17: Bodeplot - Vergleich von experimentell gebildeter Übertragungsfunktion mit gemessenem Frequenzgang

Generell passen beide Bodeplots der experimentell bestimmten Übertragungsfunktion (grüne Linie) mit der Frequenzgangmessung (blaue Linie) bis zu einer Frequenz von 0,1 rad/s sehr gut überein. Somit ist auch die Frequenzgangmessung eine Möglichkeit zur Bestimmung des Übertragungsverhaltens. Allerdings ist die zeitliche Dauer dafür erheblich größer als bei der experimentellen Bildung der Übertragungsfunktion und ab einer bestimmten Frequenz nicht mehr genau bestimmbar.

Zum Ausführen der Frequenzgangmessung wird der Hardwareaufbau mit dem Computer über die USB Ports verbunden und anschließend das MATLAB-Skript „Execute_FreqRespMeas.m“ ausgeführt. Um alle Frequenzen zu messen, benötigt das Skript ca. 8 Stunden.

3.6 Aufbau der Simulink Programme

Nachdem nun das Stellglied angesteuert, die Ausgangsgröße gemessen und außerdem die Übertragungsfunktion bekannt ist, werden die 3 Simulink Programme für einfache Steuerung, Steuerung mit Wunsch-Übertragungsfunktion sowie Regelung erstellt.

3.6.1 Aufbau der Steuerung

Beim Steuerungsprogramm wird die Soll-Temperaturerhöhung als Step Funktion (Erhöhung um 10°C ab 10 Sekunden) einerseits in die Simulation und andererseits in das Reale Model gegeben, sodass ein Vergleich zwischen Simulation und Realität möglich ist. Zudem wird der Step noch mit dem Kehrwert der stationären Verstärkung der Übertragungsfunktion multipliziert, damit man auch auf die gewünschte Temperaturerhöhung von lediglich 10K kommt. Andernfalls würde sich der Widerstand bei einem Step von eins um die stationäre Verstärkung von 27 Kelvin erhöhen. Das Stellsignal ist für Simulation und reales System das Gleiche. Dieses Programm dient dem Zweck zu sehen, dass man auch ohne eine Regelung, sprich open loop, die gewünschte Temperaturerhöhung erreicht. Dies dauert allerdings sehr lange, weshalb man sich die Frage stellen könnte ob das nicht auch schneller geht. Genügend Leistung ist ja vorhanden.

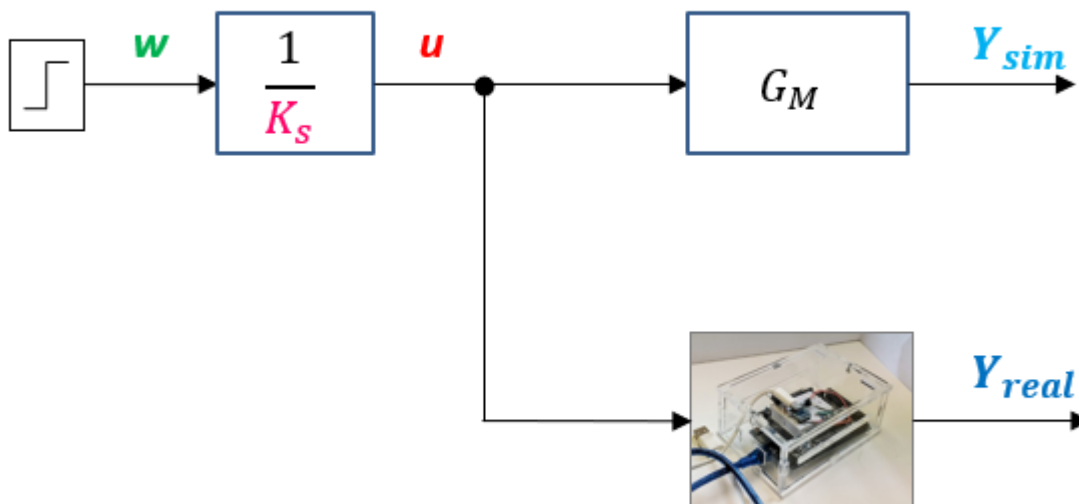


Abbildung 3-18: Model für Steuerung

In Abbildung 3-19 ist der Verlauf von Ein- und Ausgangsgröße sowie der Temperaturdifferenz zwischen Simulation und Realität dargestellt. Der untere Plot zeigt die Stellgröße u in Watt, der obere Plot die Temperatur in °C. Die Anordnung der Bilder ist für die nachfolgenden Plots dieselbe. Dabei dauert es ca. 1500 Sekunden bis die zugeführte konstante Leistung zu einer Sättigung des Temperaturverlaufes führt. Die Linien für Simulation (grün) und reale Messwerte (rot) stimmen dabei sehr gut überein, weshalb man von einer guten Modellgüte sprechen kann. Die stationäre Verstärkung von 27,074 scheint die Realität sehr gut widerzuspiegeln. Allerdings ist die Dauer, die zum Erreichen der Soll-Temperatur benötigt wird, mit 1500 Sekunden sehr lang, was uns zum nächsten Kapitel führt. Steuerung mit Wunsch-Übertragungsfunktion.

3.6 Aufbau der Simulink Programme

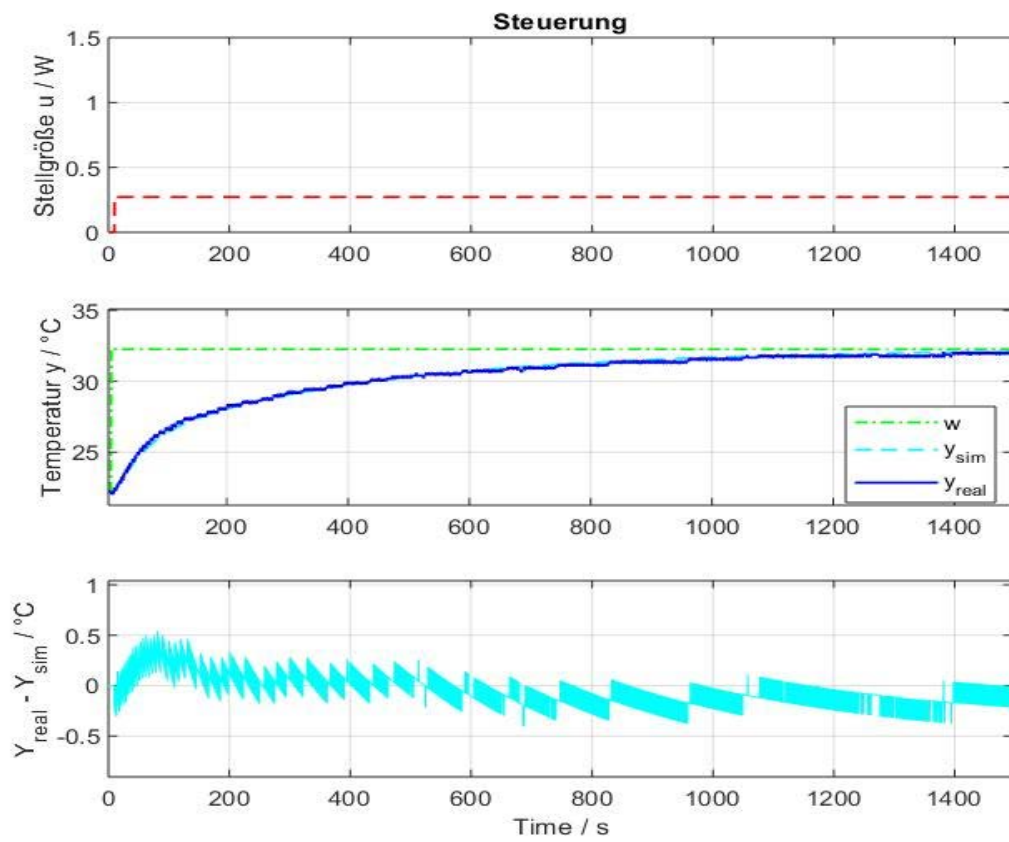
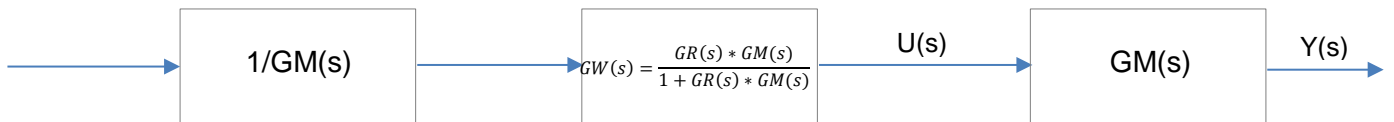


Abbildung 3-19: Vergleich der Ein- und Ausgangsgrößen der Simulation mit den realen Werten der Steuerung

3.6.1 Aufbau der Steuerung mit Wunschübertragungsfunktion

Im Folgenden wird die Übertragungsfunktion invertiert und mit einer Wunsch-Übertragungsfunktion multipliziert, welche gleich der Regler-Übertragungsfunktion des Closed-Loops im Folgekapitel ist. Also $GW(s) = \frac{GR(s)*G(s)}{1+GR(s)*G(s)}$, wobei $GR(s) = Kp + \frac{Ki}{s}$ ist. Anschließend wird die Stellgröße einerseits durch die Übertragungsfunktion $GM(s)$ geschickt (Simulation) und andererseits in das reale System gegeben. Das Stellsignal ist nun das Ergebnis aus invertierter Übertragungsfunktion multipliziert mit der Wunsch-Übertragungsfunktion.



Das Produkt der drei Übertragungsfunktionen ergibt letztlich $GW(s)$. Vorausgesetzt, die identifizierte Übertragungsfunktion entspricht genau der Realität. In der Praxis gibt es allerdings immer kleine Abweichungen, sodass zusätzlich zu $GW(s)$ noch ein Fehler multipliziert wird, welcher allerdings sehr klein sein sollte. Einzige Einschränkung ist, dass das Stellsignal natürlich nicht größer als die 1,95 Watt sein darf, da das der Grenze unseres Systems entspricht (Deshalb wird ein Saturation Block eingefügt). Zusätzlich darf die Nenner-Ordnung der Übertragungsfunktion $GM(s)$ nicht größer als die Nenner-Ordnung unserer Wunschübertragungsfunktion sein. Da wir $GM(s)$ invertieren steht nun der vorherige Nenner ja im Zähler und die Zähler-Ordnung darf nicht größer als die Nenner-Ordnung sein, da wir sonst ein nicht-kausales System hätten, welches physikalisch nicht umsetzbar ist. Die Stellgröße sollte nun also genau der Stellgröße aus dem Folgekapitel entsprechen. Abbildung 3-20 zeigt das Simulink Modell, Abbildung 3-22 und Abbildung 3-21 zeigen den Vergleich von Simulation zu Realität.

3.6 Aufbau der Simulink Programme

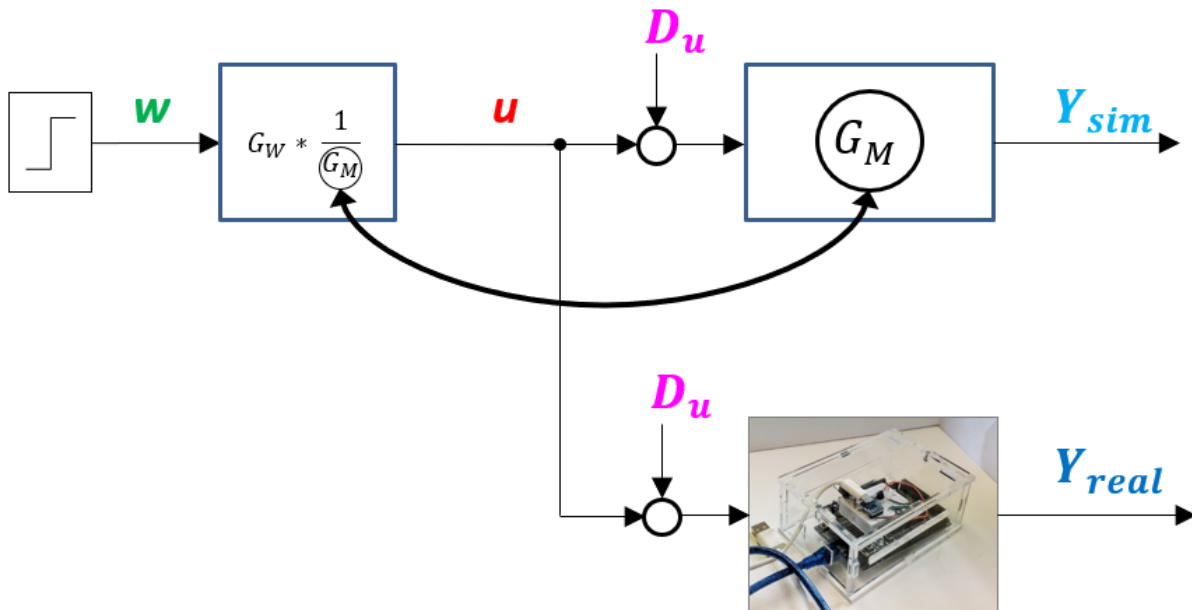


Abbildung 3-20: Modell der Steuerung mit Wunschübertragungsfunktion

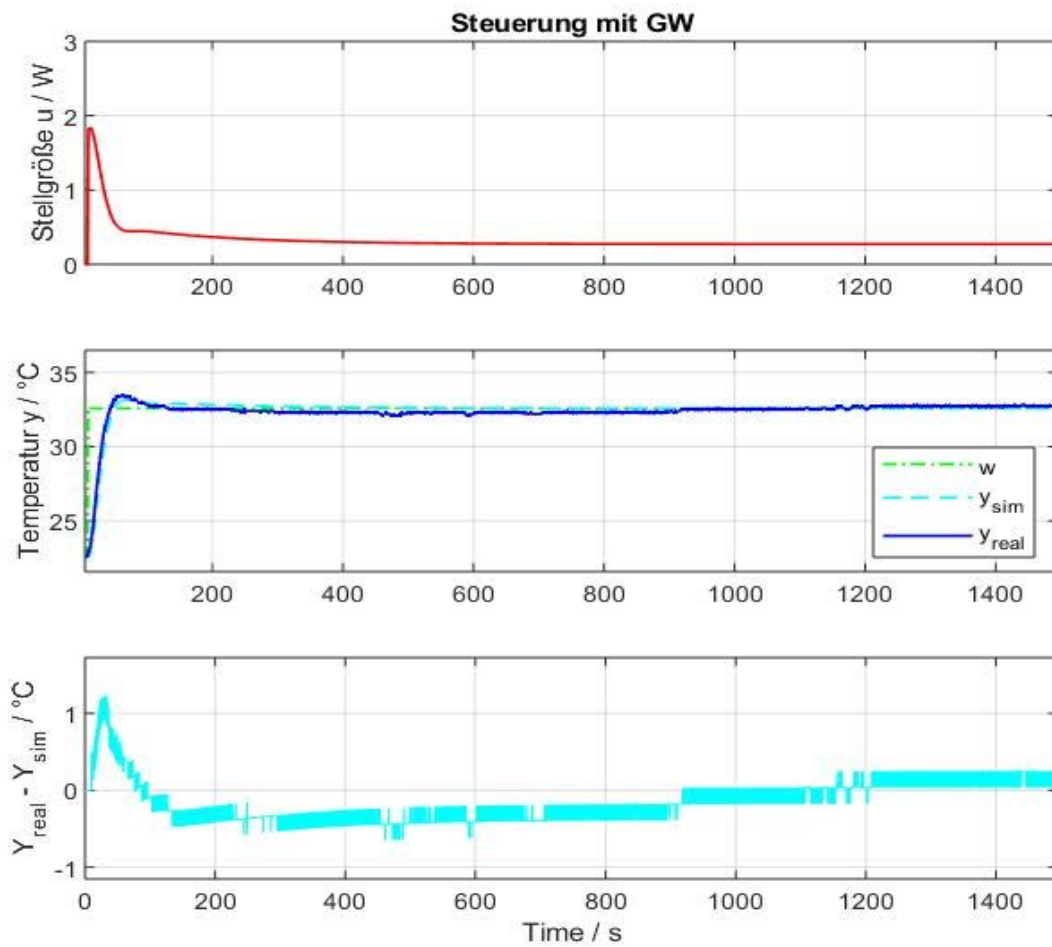


Abbildung 3-21: Steuerung mit GW, 1500 Sekunden

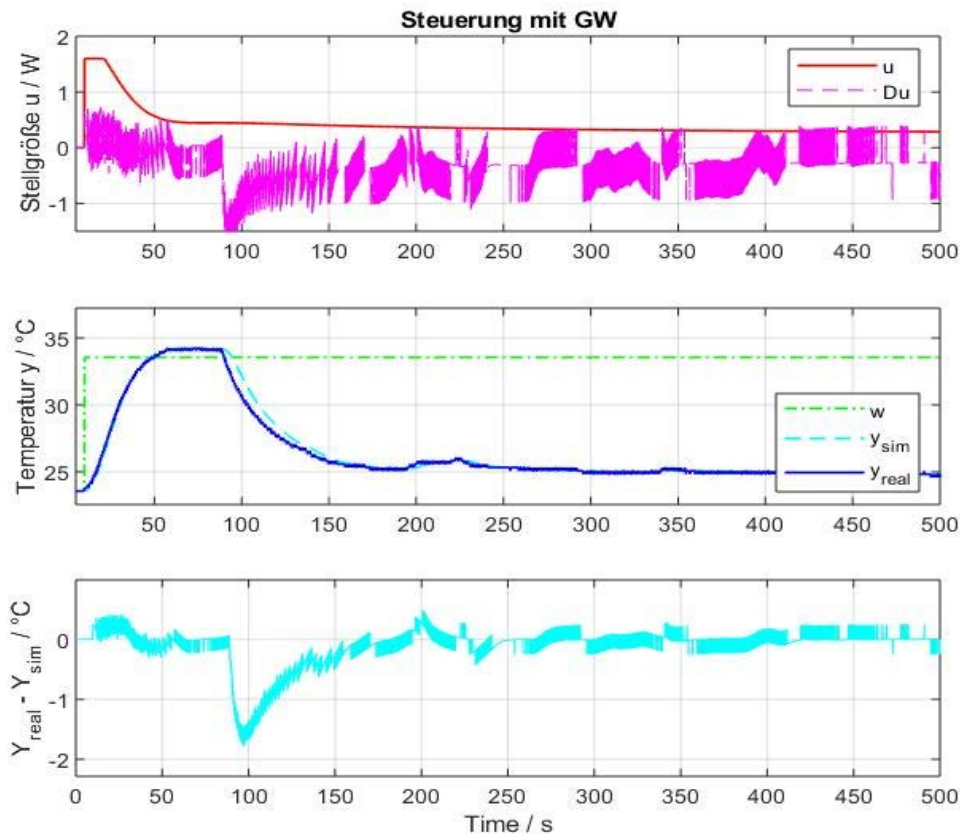


Abbildung 3-22: Vergleich der Ein- und Ausgangsgrößen der Simulation mit dem realen Wert der Steuerung mit GW inkl. Ventilatorstörung, 500 Sekunden

Der Vergleich zwischen Abbildung 3-21 und Abbildung 3-19 zeigt deutlich, dass das System nun wesentlich schneller mit Wunschübertragungsfunktion ist als ohne. Der negative Temperatursturz in Abbildung 3-22 bei ca. 80 Sekunden resultiert aus einer mit Absicht eingefügten konstanten Störung in Form eines Ventilators. Die Stellgröße reagiert dabei wie zu erwarten nicht auf die Störung, da es keine Rückführung bei einer Steuerung gibt. Die Temperaturabweichung bleibt also bestehen.

3.6.2 Aufbau der Regelung

Nachdem in den beiden letzten Kapiteln die Steuerungen genauer betrachtet wurden, beschreibt dieses Kapitel die Temperaturregelung. Abbildung 3-11 zeigt den grundsätzlichen Aufbau eines Regelkreises.

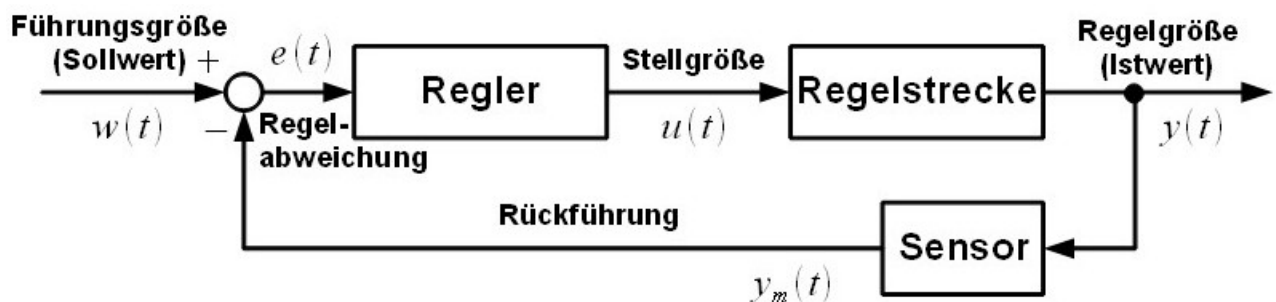


Abbildung 3-23: Aufbau eines Regelkreises

3.6 Aufbau der Simulink Programme

Als Sollwert wird wiederum der Temperatursprung von 10°C vorgegeben den wir schon in der Steuerung verwendet haben. Abbildung 3-25 zeigt das Simulink Modell der Temperaturregelung. Die Regelparameter werden nach dem Einstellverfahren der Herren Chien, Hrones und Reswick (no overshoot) vorgenommen [13]. Abbildung 3-24 zeigt dieses Verfahren.

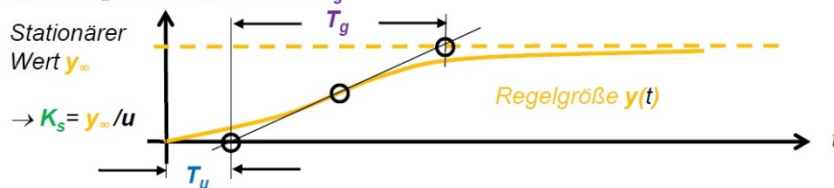
Einstellregeln (2)

2. Einstellregeln nach Chien, Hrones und Reswick

2.1 Regelstrecke mit Ausgleich (PT_n -Glieder)

Verfahrensschritte, entweder an der Strecke selber oder ihrem Modell durchzuführen:

a) Beaufschlagen der Regelstrecke mit Ausgleich mit einem sprungförmigen Sollwert der Höhe u und Analyse der Sprungantwort – wie beim Wendetangentenverfahren - nach stationärer Verstärkung K_s , Verzugszeit T_u und Ausgleichzeit T_g .



b) Die optimalen Reglerparameter sind dann wie folgt:

Regler	Aperiodisch (0% Überschwingen)	20% Überschwingen
P-Regler	$1/K_p = 3.3 K_s T_u / T_g$	$1/K_p = 1.4 K_s T_u / T_g$
PI-Regler	$1/K_p = 2.9 K_s T_u / T_g$, $T_i = 1.2 T_g$	$1/K_p = 1.6 K_s T_u / T_g$, $T_i = 1.0 T_g$
PID-Regler	$1/K_p = 2.9 K_s T_u / T_g$, $T_i = 1.0 T_g$; $T_d = 0.5 T_u$	$1/K_p = 1.05 K_s T_u / T_g$, $T_i = 1.35 T_g$; $T_d = 0.47 T_u$
PD-Regler	$1/K_p = 0.56 K_s T_u / T_g$, $T_d = 0.5 T_u$	

Abbildung 3-24: PID Einstellregeln nach Chien, Hrones und Reswick

Als Regler wird ein PI-Regler mit folgenden Werten für K_p und K_i gewählt:

- $K_p = 0,18$
- $K_i = 0,0032$

Nachstehend ist in Abbildung 3-25 das Simulink Modell der Regelung dargestellt und in Abbildung 3-26 und Abbildung 3-27 der Vergleich von Simulation mit realem System.

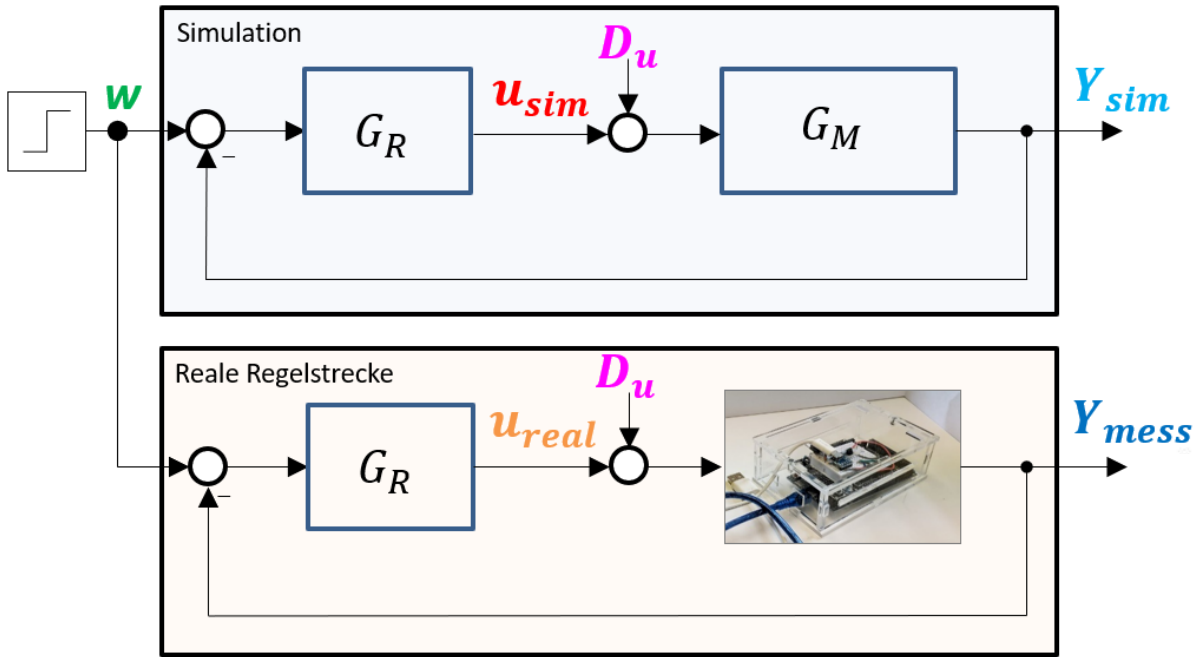


Abbildung 3-25: Modell der Regelung

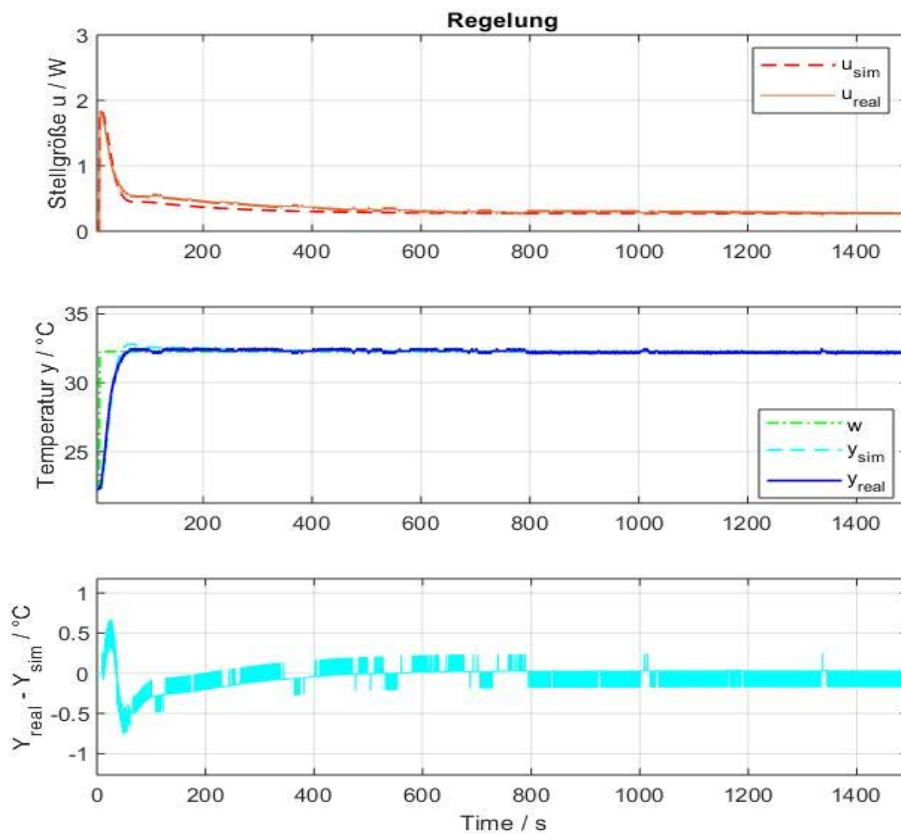


Abbildung 3-26: Regelung, 1500 Sekunden

3.6 Aufbau der Simulink Programme

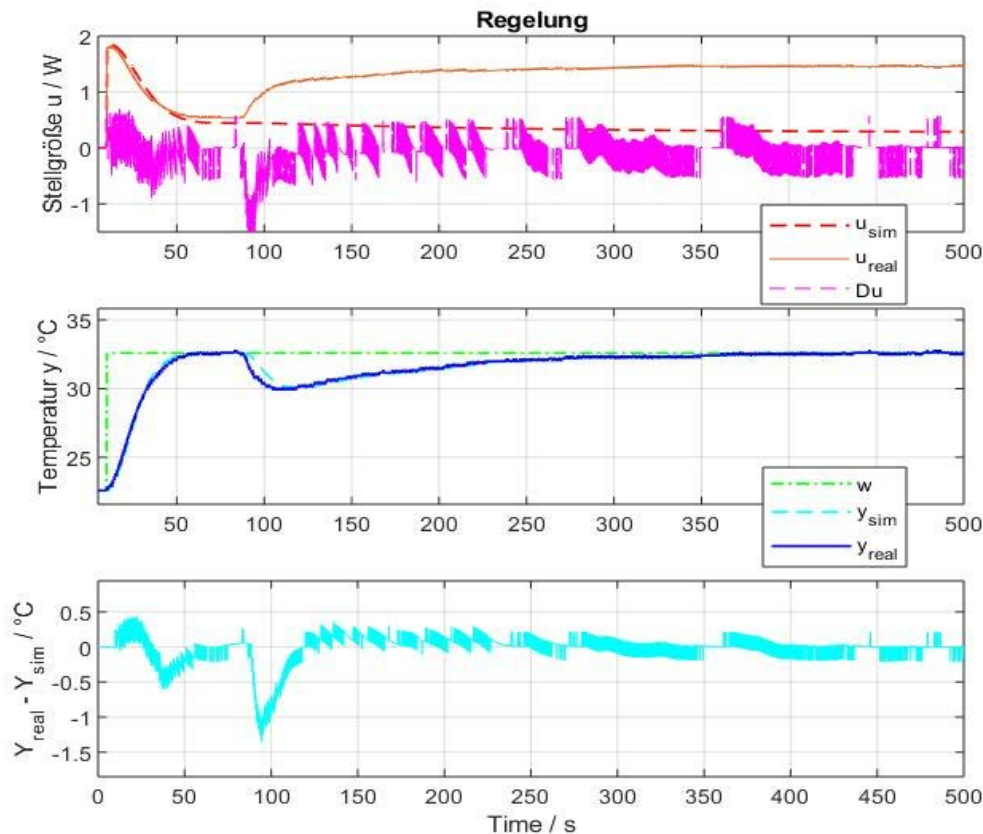


Abbildung 3-27: Vergleich der Ein- und Ausgangsgrößen der Simulation mit dem realen Werten der Regelung, 500 Sekunden

Bis zur Störung kann man keinen wesentlichen Unterschied zwischen Steuerung (mit GW) und Regelung feststellen. Erst wenn eine Störung auf das System gebracht wird, wird der Unterschied zwischen Steuerung und Regelung ersichtlich. Während die Stellgröße der Steuerung keinerlei Reaktion auf die Störung zeigt, reagiert die Stellgröße der Regelung sofort und erwärmt den Leistungswiderstand entsprechend der Regelparameter wieder auf die vorgegebene Temperatur.

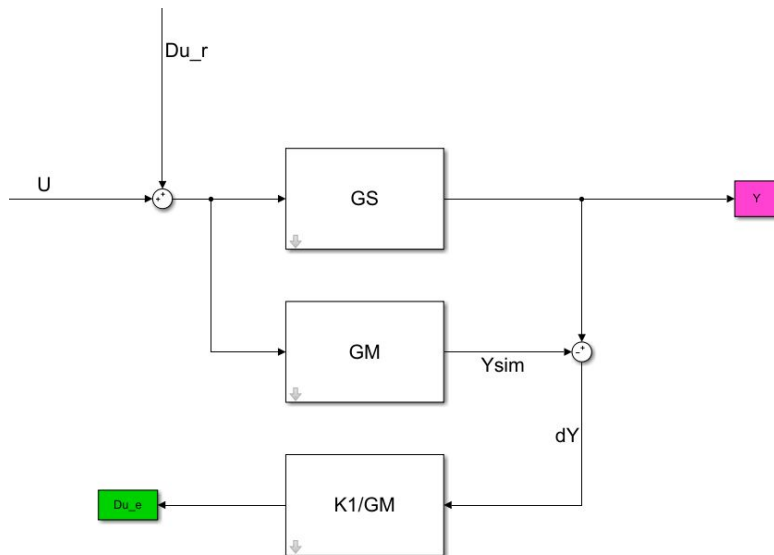
3.6.3 Störgrößenkompensation

Kapitel 3.6.2 zeigt, dass mit hinzufügen einer Feedback-Schleife und dem Vergleich von Soll- und Ist- Temperatur, das System auf eine Störung reagieren kann. Wie groß diese Störung ist, weiß man allerdings nicht. Man weiß lediglich die Reaktion der Störung am Ausgang (Temperaturabfall). Würde man allerdings beim Auftreten der Störung wissen, welcher negativen Stellgröße dies entspricht, könnte man direkt beim Auftreten der Störung in Form einer entsprechend größeren Stellgröße darauf reagieren. Das System würde also wesentlich schneller wieder die Solltemperatur erreichen. Im Folgenden ist gezeigt wie man diese Störung mittels eines geeigneten Modells schätzen und folglich kompensieren kann.

- Schätzen einer Eingangsstörung:

Um die Eingangsstörung zu schätzen, verwendet man die reale Stellgröße U und berechnet sich durch die Übertragungsfunktion des Modells GM die simulierte Temperatur. Zieht man diese von der realen Temperatur ab erhält man einen Temperaturdifferenz zwischen Simulation und Realität. Man kann die Störung also in Form eines Temperaturunterschiedes am

Ausgang ausdrücken. Multipliziert man dieses dY mit der Modell-Übertragungsfunktion GM^{-1} , so berechnet man eine Stellgröße, welche genau zu diesem Temperaturunterschied führt. Da GM^{-1} allerdings nicht mehr Pole als Nullstellen haben darf und unsere Übertragungsfunktion 3 Pole und eine Nullstelle hat, fügt man eine zusätzliche Übertragungsfunktion $K1$ ein, welche diesen Polüberschuss ausgleicht damit ein kausales System vorliegt.



Mit:

U: Stellgröße

Du_r: Störung

GS: Reale Strecke

GM: Modell von GS

K1: Kompensationsglied zum Generieren eines Polüberschusses

Abbildung 3-28: Schätzen einer Eingangsstörung

Bedingung an K1:

1. $K1/GM$ muss realisierbar sein
2. $K1$ hat eine stationäre Verstärkung von 1

Ansatz: $K1 = \frac{1}{(1+Ts)^{n_p}}$, wobei $n_p \geq \text{Polüberschuss von GM}$

$$\rightarrow (1 + Ts)^{n_p} \rightarrow s \geq \frac{1}{T}$$

- **Gleichzeitiges Schätzen von Ein- und Ausgangsstörung Du_r und Dy_r , sowie N_r**

3.6 Aufbau der Simulink Programme

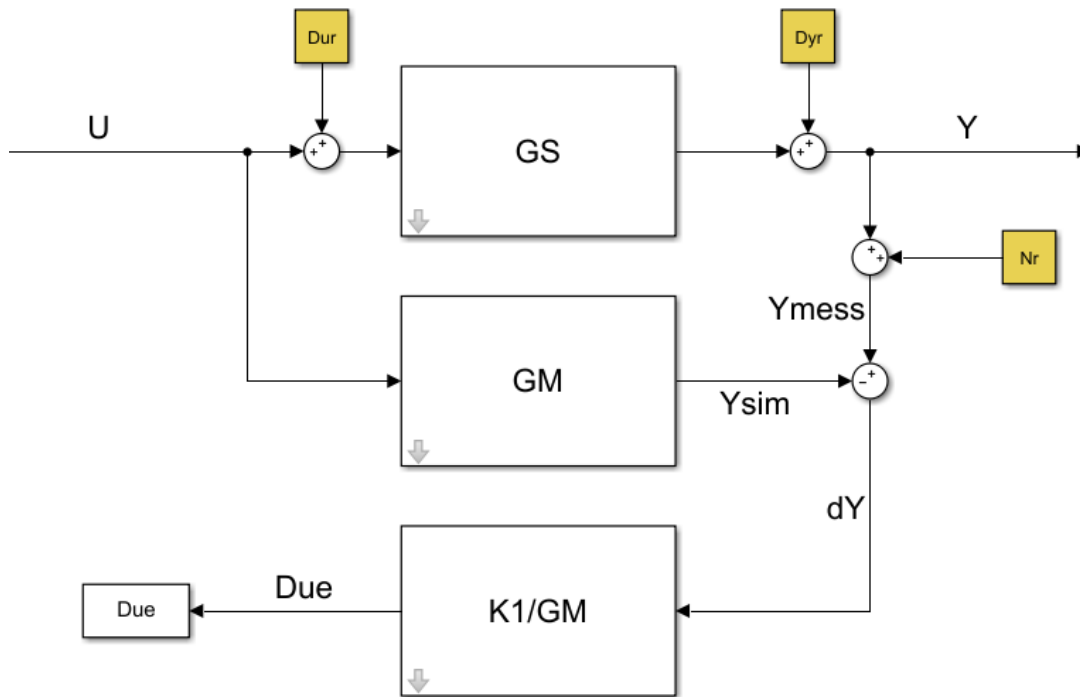


Abbildung 3-29: Gleichzeitiges Schätzen von Ein- und Ausgangsstörung D_{ur} und D_{yr} , sowie N_r

Nun kann nicht mehr auf die 3 unabhängigen Störgrößen zurückgerechnet werden. Ihre Gesamtwirkung auf dY kann aber in einer einzigen Ersatzstörung D_{ue} abgeschätzt werden, welche den gleichen Einfluss auf dY hat.

Die Frage ist nun, wie wirkt sich die Kompensation von D_{ur} , D_{yr} , N_r und D_{ue} auf den Regelkreis aus.

- Kompensation der Störgrößen im geschlossenen Regelkreis

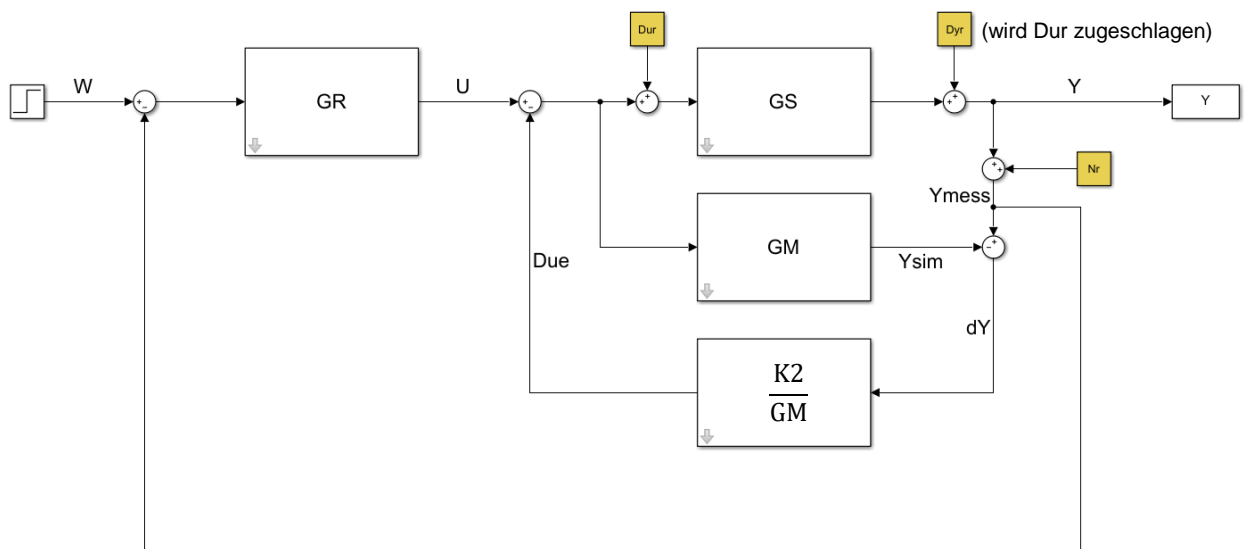
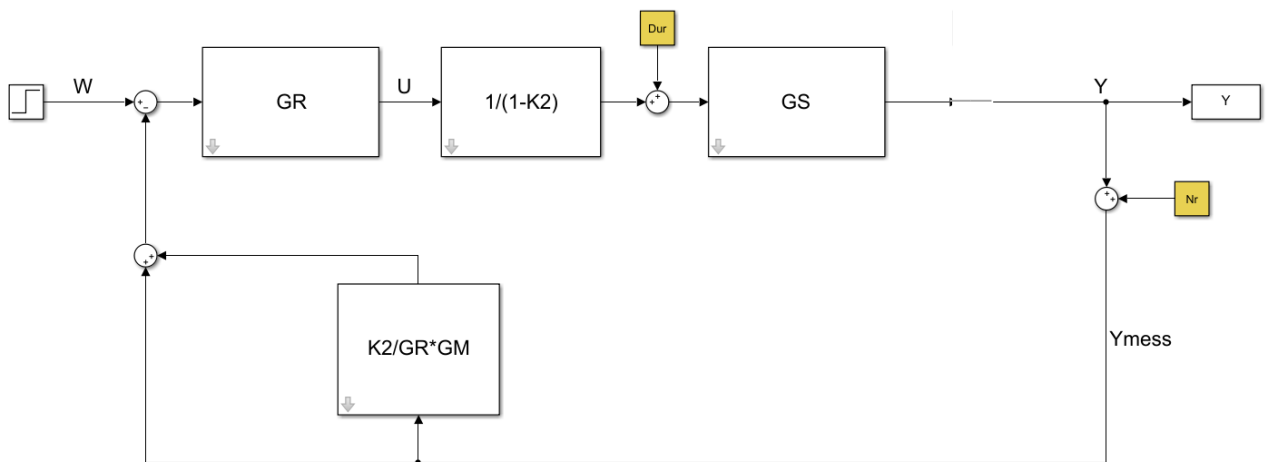
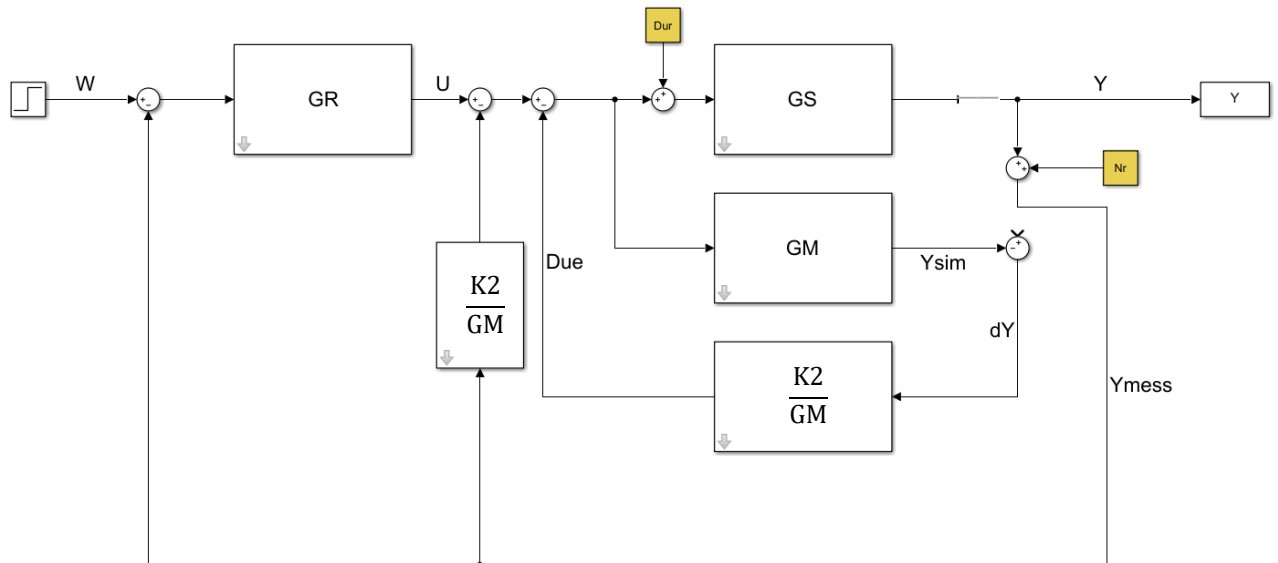


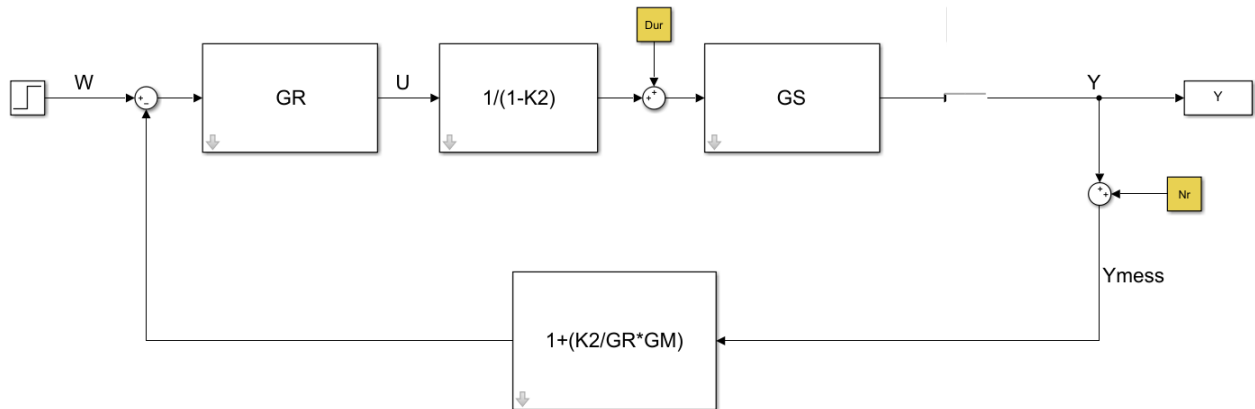
Abbildung 3-30: Kompensation der Störung im geschlossenen Regelkreis

3.6 Aufbau der Simulink Programme

Der Regelkreis in Abbildung 3-30 wird nun so umgestellt, dass keine feedback-loops innerhalb der Regelschleife vorhanden sind. Das ist notwendig, um einen theoretischen Zusammenhang aller Größen auf Y abzubilden.



3.6 Aufbau der Simulink Programme



Als Folge der Umformungen, entsteht ein Regelkreis mit nur einem einzigen feedback-loop. Nun lässt sich Y theoretisch beschreiben. Die oben gezeigten Umformungen entstammen allesamt aus [14].

Als Annahme für die nun folgende theoretische Darstellung von Y gilt, dass die reale Strecke GS gleich der experimentell gebildeten Übertragungsfunktion GM ist: $GS = GM$

$$Y = \frac{G_R G_M * \frac{1}{1-K2}}{1 + G_R G_M * \frac{1}{1-K2} * \frac{G_R G_M + K2}{G_R G_M}} * W + \frac{G_M}{1 + G_R G_M * \frac{1}{1-K2} * \frac{G_R G_M + K2}{G_R G_M}} * Dur$$

$$+ \frac{\frac{G_R G_M + K2}{G_R G_M} * G_R G_M * \frac{1}{1-K2}}{1 + \frac{G_R G_M + K2}{1-K2}} * Nr$$

$$Y = \frac{G_R G_M}{1 - K2 + G_R G_M + K2} * W + \frac{G_M * (1 - K2)}{1 - K2 + G_R G_M + K2} * Dur + \frac{G_R G_M + K2}{1 - K2 + G_R G_M + K2} * Nr$$

$$Y = \frac{G_R G_M}{1 + G_R G_M} * W + \frac{G_M}{1 + G_R G_M} (1 - K2) * Dur + \frac{G_R G_M + K2}{1 + G_R G_M} * Nr$$

Aus obiger Gleichung ist nun deutlich, dass die Störung Dur genau dann kompensiert wird, wenn $K2 = 1$ ist. Da $K2 = \frac{1}{(1+Ts)^n}$ ist, muss die Zeitkonstante T, welche frei wählbar ist, möglichst klein sein, damit $K2 = 1$ ist. Gleichzeitig bedeutet $K2 = 1$, dass das Sensorrauschen Nr nicht gefiltert wird. Es ist also ein T zu finden, für das die Störung sowie das Sensorrauschen Nr einen möglichst kleinen Einfluss auf Y hat.

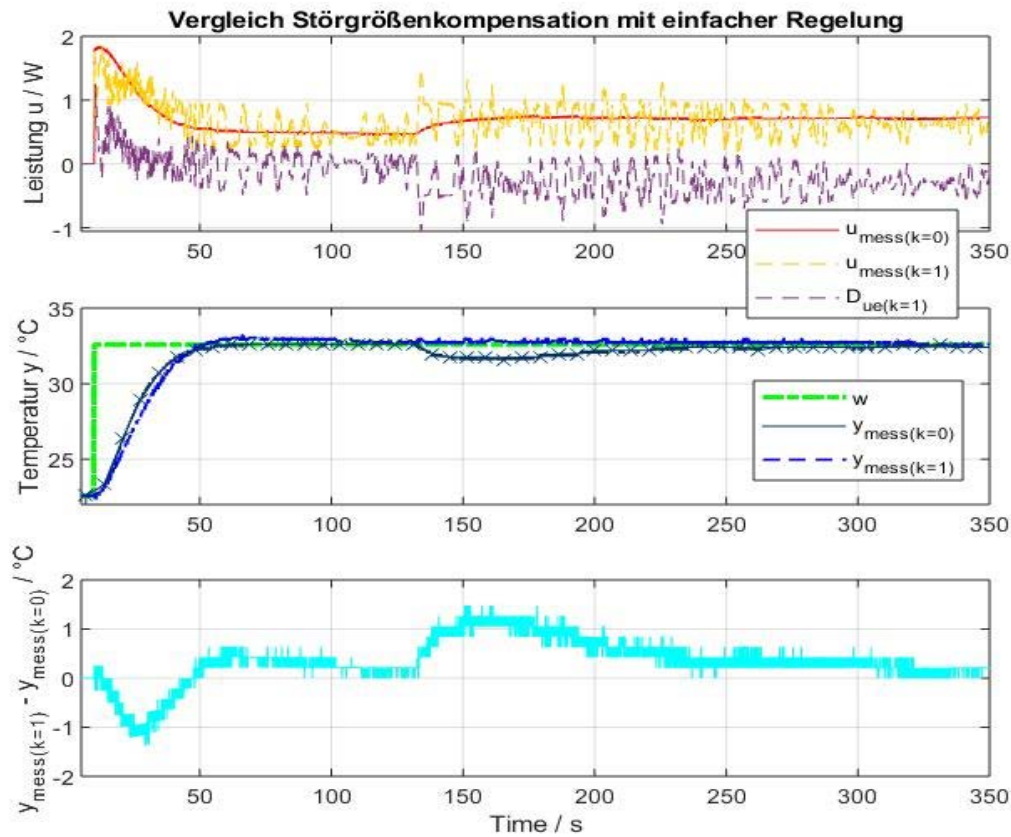


Abbildung 3-31: Vergleich der Störgrößenkompensation bei verschiedenen Zeitkonstanten T

Der mittlere Plot aus Abbildung 3-31 zeigt den Ausgang Temperatur der Regelung mit Störgrößenkompensation (blaue Linie) verglichen mit der einfachen Regelung (dunkelblaue Linie mit x) aus Kapitel 3.6.2. Dabei ist zu erkennen, dass durch eine Störgrößenkompensation die Störung auf die Temperatur keinerlei Einfluss mehr hat. Die Störung wird dabei in Form eines Ventilators erzeugt und ist bei der einfachen Regelung sowie der Regelung mit Störgrößenkompensation gleich groß. Der obere Plot zeigt wiederum die Stellgröße sowie die berechnete Störung. Dabei ist zu erkennen, dass die gelbe Linie (mit Kompensation) wesentlich schneller auf die Störung reagiert als die rote Linie (ohne Kompensation). Das ist auch die Ursache warum die Temperatur im mittleren Plot nicht abfällt. Der untere Plot zeigt zusätzlich noch die Differenz aus Temperatur mit Kompensation – Temperatur ohne Kompensation.

4 Zusammenfassung

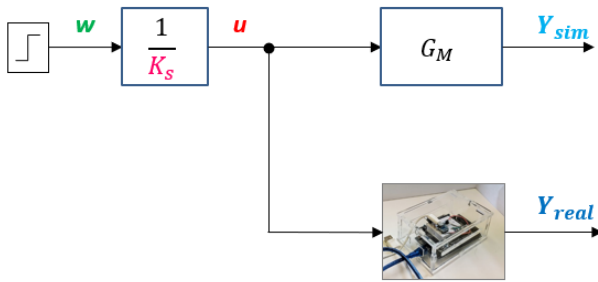


Abbildung 4-1: Modell Steuerung

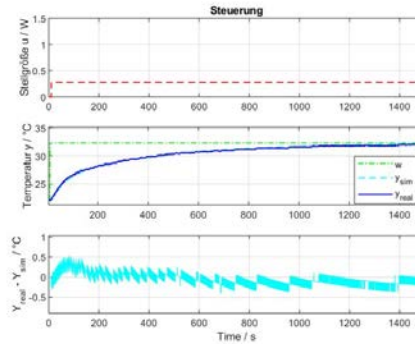


Abbildung 4-5: Steuerung

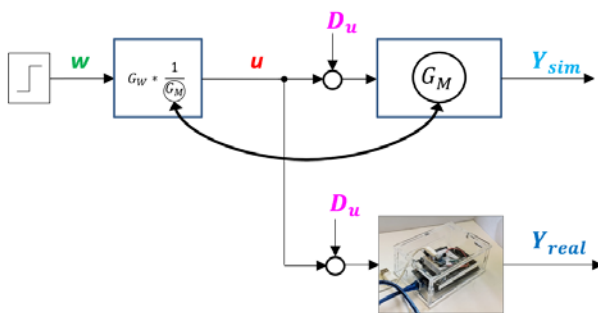


Abbildung 4-2: Modell Steuerung mit GW

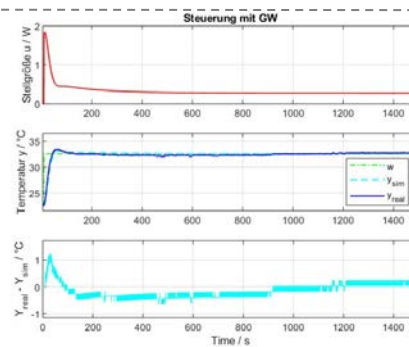


Abbildung 4-6: Steuerung mit GW

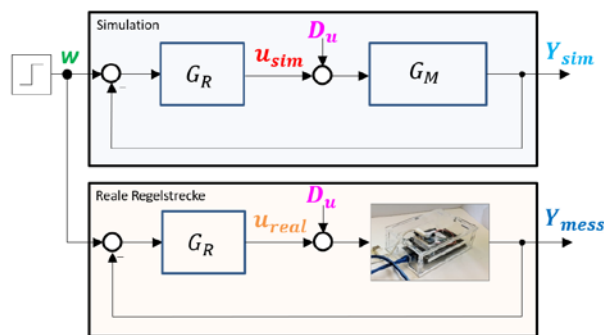


Abbildung 4-3: Modell Regelung

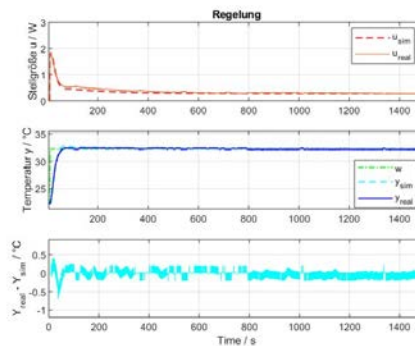


Abbildung 4-7: Regelung

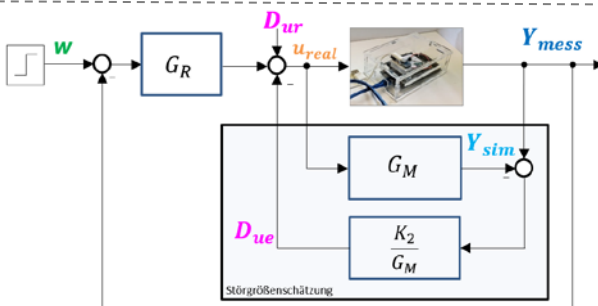


Abbildung 4-4: Modell Kompensation

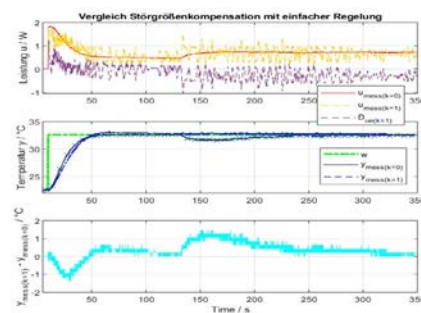


Abbildung 4-8: Kompensation

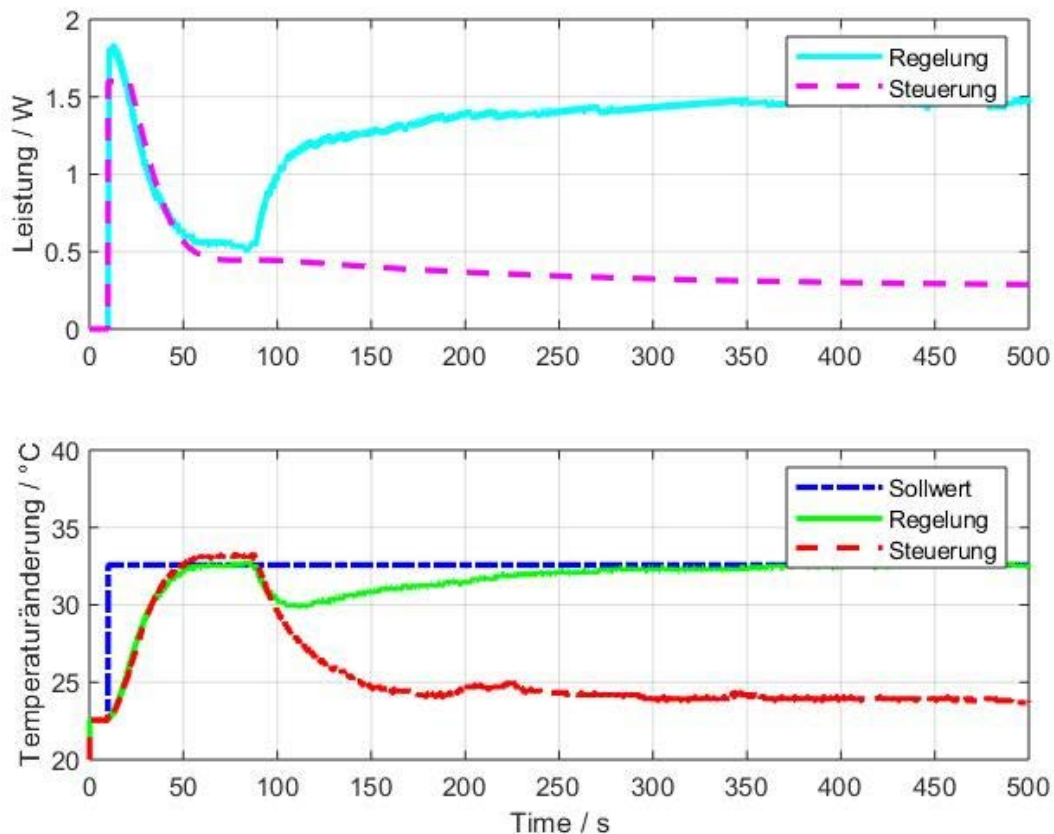


Abbildung 4-9: Vergleich Regelung, Steuerung mit GW

Die Abbildung 4-1 bis Abbildung 4-9 zeigen zusammenfassend die Simulink Modelle sowie die jeweiligen Vergleiche von Simulation zu realem System.

Abbildung 4-1 zeigt das Simulink Model für die Steuerung. Dabei wird die Sprungfunktion durch die stationäre Verstärkung der Übertragungsfunktion geteilt, um die gewünschte Soll-Temperatur zu erreichen. Die stationäre Verstärkung ist 36,7. Betrachtet man Abbildung 4-5, so nähert sich die Temperatur mit konstanter Stellgröße über einen Zeitraum von ca. 1500 Sekunden an die Soll-Temperatur.

Abbildung 4-2 zeigt das Simulink Model für die Steuerung mit einer Wunschübertragungsfunktion. Die Wunschübertragungsfunktion ist dabei $GW(s) = \frac{GR(s) \cdot G(s)}{1 + GR(s) \cdot G(s)}$. Durch Invertierung von $G(s)$ und Multiplikation mit $G(W)$, ergibt sich die gleiche Übertragungsfunktion wie die Regler-Übertragungsfunktion. Betrachtet man den zugehörigen Plot für die Steuerung mit GW (Abbildung 4-6), so ist deutlich zu sehen, dass das System nun wesentlich schneller die Soll-Temperatur erreicht. Wird eine anhaltende Störung auf das System in Form eines Ventilators gebracht, so reagiert die Stellgröße nicht darauf. Die Temperaturabweichung bleibt bestehen.

In Abbildung 4-3 ist das Simulink Model für die Regelung dargestellt. Während Abbildung 4-7 wiederum die zugehörigen Simulations-Daten mit den Messdaten vergleicht. Für die Regelung wird ein PI Regler verwendet mit $K_p = 0,18$ und $K_i = 0,0032$. Betrachtet man den Plot für die Regelung (mit Rückführung) so fällt bis zur Störung kein wesentlicher Unterschied zur Steuerung mit GW auf. Die Dauer bis zum Erreichen der Soll-Temperatur ist dieselbe wie bei

3.6 Aufbau der Simulink Programme

der Steuerung mit Wunschübertragungsfunktion. Man kann also bis zu diesem Zeitpunkt nicht sagen ob es eine Steuerung oder eine Regelung ist. Erst mit hinzufügen der Störung wird ersichtlich, dass die Regelung im Vergleich zur Steuerung auf die Störung in Form einer Stellgrößen-Veränderung reagiert (siehe Abbildung 4-9).

Abbildung 4-8 zeigt noch einmal das Potenzial einer Störgrößenkompensation mit Online Störschätzung. Dabei wird der Sollwert schon nach wenigen Sekunden nach Auftreten der Störung wieder erreicht. Bei der Regelung ohne Kompensation dauert dies mehrere Minuten.

Die Ziele der Projektarbeit, einen leichten, mobilen, robusten und kostengünstigen Hardwareaufbau inkl. Aufbauanleitung zu entwickeln um einen Widerstand in kurzer Zeit mittels Steuerung und Regelung um 10°C zu erwärmen wurden erfüllt.

Zusätzlich wird der Unterschied zwischen Steuerung und Regelung anhand der Praxis anschaulich erklärt. Mit Hilfe des Kapitels 3.4 wird außerdem die Bedeutung des Frequenzgangs durch ein Praxisbeispiel erklärt und eine Alternative zum Messen einer Übertragungsfunktion gezeigt.

5 Literatur

- [1] Arduino: ARDUINO DUE. URL - <https://store.arduino.cc/arduino-due>, Zugriff am: 04.09.2018.
- [2] Uwe Brinkschulte, T. U.: Mikrocontroller und Mikroprozessoren: Springer 2007.
- [3] Arduino: ARDUINO UNO. URL - <https://store.arduino.cc/arduino-uno-rev3>, Zugriff am: 04.09.2018.
- [4] modul_technik: ebay. URL - <https://www.ebay.de/itm/272436392748>.
- [5] A. Laucht: Fortgeschrittenen - Praktikum Versuch 24 MOSFET. URL - <https://www.ph.tum.de/academics/org/labs/fopra/docs/userguide-24.de.pdf>.
- [6] Elektronik Kompendium: MOS-Feldeffekttransistor (MOS-FET). URL - <https://www.elektronik-kompendium.de/sites/bau/0510161.htm>.
- [7] N.N.: Fritzing: FH Potsdam.
- [8] Texas Instruments: LM35 Precision Centigrade Temperature Sensors. URL - http://www.redrok.com/TemperatureSensor_LM35_10mVperC.pdf.
- [9] Universität Ulm: Pulsweitenmodulation. URL - https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.050/vorlesungen/wise1011/les/Uebung_2.pdf.
- [10] Prof. Dr.-Ing. Peter Zentgraf, Johannes Fischer (B.Eng.), Daniel Weindler (B.Eng), Christian Ostermaier, Konstantin Shekhovstov: pzMove 2018.
- [11] Cosmos indirekt: Stromwärmegesetz. URL - <https://physik.cosmos-indirekt.de/Physik-Schule/Stromw%C3%A4rmegesetz>.
- [12] Jan Lunze: Regelungstechnik 1 – Systemtheoretische Grundlagen, Analyse und Entwurf einschleifiger Regelungen. 11. Auflage: Springer.
- [13] Prof. Dr.-Ing. Peter Zentgraf: Regelungstechnik I | Kap. 7; Synthese – SS 2015 2015.
- [14] Prof. Dr.-Ing. Peter Zentgraf: Regelungstechnik I | Kap. 4; Beschreibung der Übertragungssysteme im Bild- und Frequenzbereich 2015.