

DIPLOMARBEIT

Dreiphasige Synchronsteuerung für hybride Hochstromeinschaltsysteme

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines
Diplom-Ingenieurs unter der Leitung von

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Johann Ertl (TU Wien)
E370 Institut für Energiesysteme
und
elektrische Antriebe

eingereicht an der Technischen Universität Wien Fakultät für Elektrotechnik
und Informationstechnik

von

Peter Jonke
e0625410, Knz: 435
A-1130 Wien, Wattmannngasse 24/13

Wien, im September 2011

Vorwort

Diese Arbeit ist eine Fortsetzung der Entwicklung eines hybriden Hochstromschalt-systems des AIT (Austrian Institute of Technology). In Vorarbeiten wurde bereits ein Hybridschaltersystem entwickelt und gebaut. Der letzte Schritt soll nun durch diese Arbeit erfolgen, in dem ein Ansteuerungskonzept für das Hybridschaltersystem entwickelt wird.

Ich möchte mich an dieser Stelle bei all jenen bedanken, die mich bei der Anfertigung meiner Diplomarbeit so kräftig unterstützt haben. Ganz besonders möchte ich mich bei meinem Professor, Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Johann Ertl und Dipl.-Ing. Georg Brauner bedanken, die mich während meiner Diplomarbeit betreut und mit ihrem umfassenden Fachwissen unterstützt haben. Außerdem möchte ich mich bei Dipl.-Ing. Dr. Wolfgang Hribernik bedanken, der diese Arbeit ermöglicht hat und darüberhinaus gefördert hat.

Danken möchte ich auch Herrn Dipl.-Ing. Bernhard Kubicek, der mich bei der Fertigung der Printplatten tatkräftig unterstützt hat, sowie Herrn Hanna Raheb, MSc der mir während der gesamten Arbeit mit zahlreichen Anregungen geholfen hat und auch Filip Andren, MSc der mit dem von ihm erstellten Matlab/Simulink Modell des Hybridschalters zum Erfolg der Arbeit maßgeblich beigetragen hat.

Ein ganz besonderer Dank gilt auch meiner Freundin sowie meiner Familie, ohne Sie wäre dieses Studium nicht möglich gewesen.

Inhaltsverzeichnis

Vorwort	i
Zusammenfassung	v
Abstract	vii
1 Aufgabenstellung	1
2 Stand der Technik	3
2.1 Leistungsversuchsfeld AIT	3
2.2 Hybrides synchrones Hochstromeinschaltssystem am AIT	8
2.3 Zusatzbeschaltung	10
3 Technische Spezifikationen und Methoden	13
3.1 Anforderungen an die Steuerungshardware	13
3.2 Anforderungen an die Steuerungssoftware	16
3.3 Methoden - Entwicklung	17
3.4 Methoden - Simulationen	17
3.5 Methoden - Validierung	18
4 Grundlagen	19
4.1 Einschaltverhalten von L-R Wechselstromkreisen	19
4.2 Einschalten des Hybridschalters unter Nebenbedingungen	22
5 Einschaltvorgang des Dreiphasensystems	25
5.1 Allgemeiner Aufbau des Dreiphasensystems	25
5.2 Einschaltverhalten L1 - N	26
5.3 Einschaltverhalten L1 - L2	27
5.4 Einschaltverhalten L1 - L2 - L3	28
6 Entwicklung der Hardware	33
6.1 Konzept	33
6.2 Controller	34
6.3 Ansteuerschaltung eines Hybridschalters	39
6.4 Entwicklung der Zusatzbeschaltung	44
6.5 Ansteuerschaltung der freien Kanäle	46
7 Simulationen	49
7.1 Allgemeines	49
7.2 Synchronisation	49
7.3 Ansteuerung mechanischer Schalter	53
7.4 Ansteuerung der optisch-triggerbaren Thyristoren	54

7.5	Status Meldekontakte	56
7.6	Dreiphasiges Einschalten	58
8	Entwicklung der Software	61
8.1	Konzept	61
8.2	Mikrocontroller - Synchronsteuerung	64
8.3	Mikrocontroller - Hybridschalter	68
8.4	Entwicklung in LabView	68
8.5	Implementierung der Signalverlaufstabellen	74
9	Messungen	79
9.1	Allgemeines	79
9.2	Verzögerungsmessung der Solid-State Relais	79
9.3	Verzögerungszeiten eines mechanischen Schalters	80
9.4	Einschaltvorgänge Hybridschalter	81
10	Validierung mit Hardware-in-the-loop	83
10.1	Simulation mit Hardware-in-the-loop	83
10.2	Offline Simulation	84
10.3	Hardware-in-the-loop - Simulationen	86
11	Schlussfolgerungen	89
	Quellcode - Mikrocontroller	91
1	Header und Funktionen	91
2	Synchronsteuerung - Hauptprogramm	98
3	Ansteuerung der Hybridschalter - Hauptprogramm	103
	Schematische Pläne	107
4	Übersicht	107
	Layout und Fertigung	111
5	Übersicht	111

Zusammenfassung

In dieser Arbeit werden die theoretischen Grundlagen und die Entwicklung einer Steuerung für eine dreiphasige Synchronsteuerung für hybride Hochstromeinschaltsysteme beschrieben. Dabei wird auf eine bestehende Entwicklung von Hybridschaltern aufgebaut, welche für Prüfungen im Leistungsversuchsfeld des AIT (Austrian Institute of Technology) eingesetzt werden sollen.

Dazu ist zunächst eine Analyse der Einsatzumgebung (Leistungsversuchsfeld) und der bereits entwickelten Hybridschalter notwendig, um die Anforderungen bzw. Spezifikationen bestimmen zu können. Als Grundlage für die Entwicklung der Steuerung, wird sodann das Einschaltverhalten eines einphasigen Hybridschalters genau analysiert. Basierend auf diesen Grundlagen kann ein dreiphasiges Modell für den Einschaltvorgang entwickelt werden. Für das Konzept der Hardware wird auf die Anforderungen der bereits vorhandenen Hybridschalter Rücksicht genommen. Auf die Entwicklung der notwendigen Software-Bedienoberfläche bzw. Steuerungssoftware für die eingesetzten Mikrocontroller wird ebenfalls eingegangen. Das entwickelte Gesamtkonzept sieht vor, dass die Software durch die Eingabe aller für den Einschaltvorgang wichtigen Parameter, die notwendigen Ansteuerungsmaßnahmen berechnet und anschließend ausführt. Ein weiterer Aspekt ist die Validierung bzw. Simulation der Hardware/Software. Zunächst wird die Hardware und Teile der erarbeiteten Grundlagen mit PSpice simuliert und die Ergebnisse zur Überprüfung herangezogen. Für die Validierung des Gesamtkonzepts kommt im Rahmen von „Hardware-in-the-loop“ ein Echtzeitsimulator zum Einsatz, welcher es ermöglicht, die entwickelte Steuerung zu testen. Dies geschieht durch eine Simulation des Hybridschalters auf dem zuvor genannten Echtzeitsystem. Damit ist es möglich, sowohl die entwickelte Hardware als auch die entwickelte Software zu validieren. Zuletzt wird noch ein Ausblick auf die weitere Entwicklung und Ausbaufähigkeit des Projekts gegeben.

Abstract

This master thesis presents a simulation-based development of a controller for a three-phase hybrid switchgear system.

Conventional mechanical switches show a considerably high dispersion of the mechanical inherent time. This makes it impossible to switch on the current at a defined time. By using a combination of a mechanical switch and thyristor devices, which have a high switching accuracy, into a single switch, a new type of component is created called hybrid switch. This system is able to switch high currents with a switching accuracy of a few degrees. The concept of the hybrid switchgear and its control strategies are described in detail in this master thesis. The main development task is to control the hybrid switch synchronously with the three-phase system, allowing precise switching accuracy and switching times in the vicinity of the zero crossing of the supply voltage. By using a simulation-based development process the hybrid switchgear was first tested using simulations. A first prototype of the controller hardware was developed and validated using Hardware in the Loop (HIL) simulation. In this paper a HIL simulation for the validation of the switching accuracy of the hybrid switchgear is presented. The results show that the controller fulfils the required switching accuracy and is able to handle the dispersion of the mechanical inherent time.

1 Aufgabenstellung

Für ein vorhandenes hybrides Hochstromschaltssystem soll eine dreiphasige Synchronsteuerung entworfen werden, welche bei definierten Zeitpunkten bis zu 1° elektrisch genau (dies entspricht ca. $55 \mu s$ bei 50 Hz) den Stromkreis schließen soll. Aufbauend auf einer genauen Analyse des Einschaltvorgangs muss zunächst eine Steuerungshardware für die Hybridschalter entwickelt werden, welche zu den vorhandenen Hybridschaltern kompatibel sein muss. Dabei muss der Hybridschalter (bestehend aus einem mechanischen Schalter und Thyristoren) angesteuert werden und auch Parameter, wie die Verzögerungszeit des mechanischen Schalters, gemessen werden. Die Synchronisation mit der Netzspannung muss für das synchrone Einschalten mit einer großen Genauigkeit erfolgen. Hierfür ist ebenfalls eine geeignete Schaltung zu entwerfen. Für die Bedienung des Hybridschalters ist eine grafische Benutzeroberfläche in LabView zu entwickeln, welche es ermöglichen soll, sowohl den Hybridschalter zu steuern, wie auch noch zusätzlich freie Kanäle um einen gesamten Prüfablauf steuern zu können. Ein weiterer Aspekt ist die Validierung des Hybridschalters in einer Hardware-in-the-loop Umgebung, um die Hybridschalter und damit die realen Einsatzkomponenten der Steuerung zu simulieren. Mit der Hardware-in-the-loop Umgebung soll die entwickelte Steuerung auf ihre Funktion überprüft werden.

2 Stand der Technik

2.1 Leistungsversuchsfeld AIT

Anforderungen

Das Leistungsversuchsfeld des AIT ist nach [1] für die experimentelle Entwicklung, Untersuchung und Prüfung von elektrotechnischen Systemen, Produkten und Komponenten der Energie-, Verkehrs-, Bau- und Informationstechnik im Hinblick auf Sicherheit, Zuverlässigkeit, Funktionalität und Effizienz unter Betriebs- und Störbedingungen konzipiert worden.

Ein Bestandteil des Leistungsversuchsfeldes ist die Niederspannungs- Hochstromanlage NH150, welche für Prüfungen im Niederspannungsbereich von 0,1 kV bis 2 kV mit Prüfleistungen bis 150 MVA ausgelegt ist. Typische Untersuchungsgegenstände für die Prüfung mit der Niederspannungs- Hochstromanlage NH150 sind:

- Schaltgeräte (Leistungsschalter, Last-Trennschalter, Trennschalter, Schutzschalter)
- Sicherungen
- Schaltanlagen
- Transformatoren und Wandler
- Drosselpulen, Kapazitäten, Widerstände
- Kabel und Leitungen

Diese Niederspannungs-Betriebsmittel können nun verschiedenen Prüfungen unterzogen werden. Dabei kann man sie auf das Schaltvermögen unter Betriebs- und Kurzschlussbedingungen, die Störlichtbogenfestigkeit, thermische und elektrodynamische Kurzzeitstromtragfähigkeit, thermische Langzeitstromtragfähigkeit (Erwärmung) und auch das Betriebsverhalten untersuchen. Die genannten Untersuchungen bzw. Prüfungen können als Typ-, Stück- oder Sonderprüfungen nach Richtlinien und Bestimmungen durchgeführt werden. Ein Auszug der europäischen Bestimmungen für Niederspannungs-Systeme, Produkte und Komponenten sei im Folgenden gegeben:

- EN 60947 ... Niederspannungs-Schaltgeräte
- EN 60269 ... Niederspannungs-Sicherungen
- EN 60898 ... Leitungsschutzschalter
- EN 61008 ... Fehlerstromschutzeinrichtungen
- EN 61439 ... Niederspannungs-Schaltgerätekombinationen
- EN 60099 ... Überspannungsableiter

Für den Einschaltvorgang während Kurzschlussprüfungen ist der Einsatz von Hybrid-schaltern angedacht, die ein synchrones Einschalten bei beliebigen Momentanwerten der Spannung ermöglichen sollen.



Abbildung 2.4: 110 kV Anspeisung und Mittelspannungstransformator M850

Einspeisetransformator M850

Die 110 kV Einspeisung mit Erdungstrenner, Leistungsschalter und nachfolgendem Einspeisetransformator M850 ist in Abbildung 2.4 dargestellt. Der M850 besitzt auf der Primärseite einen Trafostufenschalter mit insgesamt 25 Stellungen um das Übersetzungsverhältnis von 110 kV/30 kV zur Spannungsanpassung ändern zu können. Mit der Versorgung aus dem 110 kV-Netz sind folgende (aus Sicht der Anlagenauslegung) Prüfleistungen zulässig:

- $S(\leq 1\text{ s})=150\text{ MVA}$ (Stoßleistung)
- $S(\leq 3\text{ s})=75\text{ MVA}$ (Stoßleistung)
- $S(\leq 10\text{ s})=40\text{ MVA}$ (Stoßleistung)
- $S(\infty)=5\text{ MVA}$ (Dauerleistung)

Hochstromtransformator NH150

Der Hochstromtransformator NH150 kann oberspannungsseitig als Dreieck- (Dy5) oder Sternschaltung (Yy0) geschaltet werden. Der NH150 besitzt oberspannungsseitig neun Stufenschalterstellungen und niederspannungsseitig 12 galvanisch getrennte Wicklungen pro Phase. Diese können je nach Bedarf in Serie oder parallel geschaltet werden. Damit ergibt sich unterspannungsseitig ein Spannungsbereich von 105,7 V-1268 V für eine Yy0 Verschaltung bzw. 183 V-2196 V für eine Dy5 Verschaltung.



Abbildung 2.5: Hochstromtransformator NH150

Vorimpedanzen

Die für die Nachbildung der Netzseite eingesetzten ohmschen Widerstände und Luftdrosseln, sind in in Abbildung 2.6 dargestellt. Mit Hilfe dieser Impedanzen ist es möglich, den prospektiven Kurzschlussstrom sowie den Leistungsfaktor $\cos(\varphi)$ zu bestimmen. Die Vorimpedanzen sind überspannungsseitig in Serie zu den Spulen des Hochstromtransformators NH150 geschaltet und werden somit in der Mittelspannungsebene eingesetzt.



(a) Ohmsche Widerstände



(b) Luftdrosseln

Abbildung 2.6: Vorimpedanzen des Leistungsversuchsfeldes

2.2 Hybrides synchrones Hochstromeinschaltssystem am AIT

Konzept

Für Kurzschlussprüfungen von elektrischen Betriebsmitteln ist es oftmals erforderlich, den Stromkreis bei genau definierten Momentanwerten der Spannung des 50 Hz Netzes zu schließen. In jedem Fall empfiehlt sich hierfür der Einsatz von Halbleiterschaltern, um ein präzises und entprelltes Einschalten gewährleisten zu können. Es wurden hierzu nach [2] einige Recherchen zu Hochstromeinschaltssystemen angestellt. Aus wirtschaftlichen und technischen Gründen hat man sich für die Realisierung von Hybridschaltern entschieden.

Die Strombelastbarkeit ist bei Halbleiterschaltern im Allgemeinen nicht so hoch wie bei mechanischen Schaltern (Leistungsschaltern). Um die Strombelastbarkeit zu erhöhen, kann der Einsatz von mehreren parallelen Halbleiterschaltern angedacht werden. Der Nachteil bei dieser Variante besteht jedoch darin, dass Halbleiterschalter sehr kostenintensiv sind und auch im Betrieb höhere Verluste als mechanische Schalter aufweisen. Eine hohe Strombelastbarkeit ist aber kurzfristig auch mit entsprechend weniger Halbleiterschaltern möglich, sofern man sicherstellen kann, dass der Strom nach kurzer Zeit (1 ms - 2 ms) auf einen mechanischen Schalter kommutiert. Dies ist das Prinzip des Hybridschalters, also eine Kombination aus Halbleiterschalter, meistens Thyristoren bzw. Triacs, und einem mechanischen Schalter.

In [3] wurden die Hybridschalter für ein dreiphasiges hybrides Hochstromeinschaltssystem entworfen und ausgelegt. Die Spezifikationen für dieses Schaltsystem sind im Folgenden gegeben:

- max. Kurzschlussstrom I_K : 100 kA (je Phase)
- max. Außenleiterspannung U_A : 1000 V

Anforderungen an den mechanischen Schalter

Nach [3] sind die wichtigsten Kenndaten für die Auswahl des mechanischen Schalters das Bemessungskurzschlusseinschaltvermögen I_{cm} bzw. der Bemessungskurzzeitstrom I_{cw} . Für den Einsatz im Hybridschalter wurde der Leistungsschalter NW20HF der Firma Schneider-Electric gewählt. Die Kenndaten sind nach [4] in Tabelle 2.1 angegeben.

Kenndaten Leistungsschalter NW20HF	
Bemessungsstrom I_n ($T = 40^\circ$)	2 kA
Bemessungskurzschlusseinschaltvermögen \hat{I}_{cm} (440 V)	187 kA
Bemessungskurzschlusseinschaltvermögen \hat{I}_{cm} (690 V)	187 kA
Bemessungskurzzeitstrom 3 s	75 kA
Ausschaltvermögen I_{cu}	85 kA
Lebensdauer mechanisch (ohne Wartung)	10.000
Lebensdauer elektrisch (ohne Wartung, 690 V)	6.000

Tabelle 2.1: Kenndaten Leistungsschalter NW20HF [4]

Im Weiteren wurden Messungen angestellt um die Einschaltverzögerung des mechanischen Schalters festzustellen. Dabei wurde festgestellt, dass der Einschaltverzug bei dem Prototypen zwischen 48 ms und 49 ms liegt. Die Streuung liegt also im Bereich von 1 ms. Mindestens diese Zeit muss der Thyristor also den Strom führen können, bis der Strom auf den mechanischen Schalter kommutiert.

Anforderungen an den Thyristor

Wichtig für die Auslegung des Thyristors ist der Grenzstrom sowie das Grenzlastintegral. Der Grenzstrom ist der maximale nicht betriebsmäßige Spitzenstrom für die Zeitdauer einer Halbwelle. Das Grenzlastintegral ist nach [5] $\int I^2 dt$ (auch $I^2 t$ -Wert) ein Kriterium für die Kurzzeitüberlastbarkeit eines Halbleiterbauelements. Es gibt den zulässigen Höchstwert an, der innerhalb einer bestimmten Zeitdauer t (meist 10 ms, also eine Halbwelle im 50 Hz Spannungsnetz) anstehen darf. Halbleiterbauelemente sind sehr empfindlich auf Überlast. Schon eine Überlast von wenigen ms kann zur Überschreitung der zulässigen Sperrschicht-Temperatur und somit zur Zerstörung des Bauteiles führen. Sie werden daher durch schnell reagierende Sicherungen abgesichert, die über einen geringeren $I^2 t$ -Wert verfügen, als das abzusichernde Bauelement. Das Grenzlastintegral des Thyristors sollte mit Sicherheitsfaktor maximal $\int i^2 dt = 40 \text{ MA}^2 \text{s}$ aufweisen. Die Anforderungen an den Thyristor sind im Folgenden angegeben:

- Grenzstrom: $I_{TSM} = 90 \text{ kA}$
- Grenzlastintegral: $\int i^2 dt = 40 \text{ MA}^2 \text{s}$
- Blockier- und Sperrspannung: $V_{BO} = 3800 \text{ V}$

Der gewählte Thyristor T4003N von Eupec erfüllt diese Spezifikationen und wurde für den Einsatz im Hybridschalter ausgewählt. Der gewählte Thyristor ist optisch-triggerbar und bringt damit eine galvanische Trennung zwischen Steuerstromkreis und Leistungsstromkreis mit. Die Kenndaten des Thyristors sind gegeben durch [6]:

Kenndaten Thyristor T4003N	
Spitzensperrspannung U_{RRM}	5400 V
Stoßstrom-Grenzwert I_{TSM}	105 kA
Grenzlastintegral $I^2 t$	$55 \cdot 10^6 \text{ A}^2 \text{s}$
Bemessungskurzzeitstrom 3 s	75 kA
erforderliche Zündleistung P_{LM}	40 mW
Haltestrom	100 mA

Tabelle 2.2: Kenndaten Thyristor T4003N [6]

Topologie des Hybridschalters

Die Topologie des Hybridschalters ist in Abbildung 2.7 dargestellt. Da der Leistungsschalter NW20HF als dreiphasiger Schalter ausgeführt ist, sind hier drei Pole vorhanden die bei der Verwendung als einphasiger Schalter parallel geschaltet werden. Es bietet sich hier an, die gleiche Anzahl an Thyristoren zu wählen. Es gilt nur zu beach-

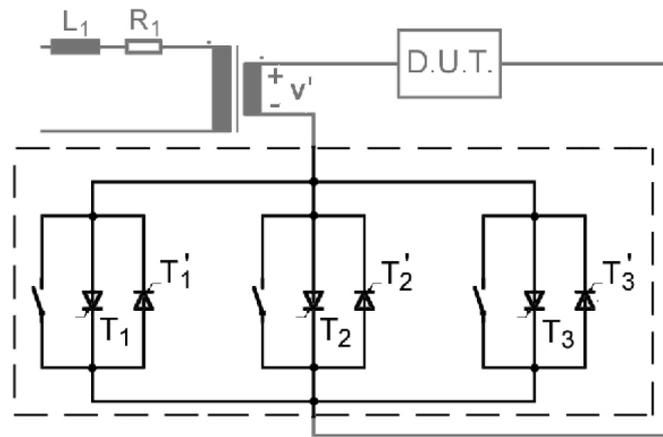


Abbildung 2.7: Topologie des Hybridschalters nach [7]

ten, dass für einen Pol zwei anti-parallel geschaltete Thyristoren zu verwenden sind. Der Aufbau eines Hybridschalters ist in Abbildung 2.8 dargestellt.

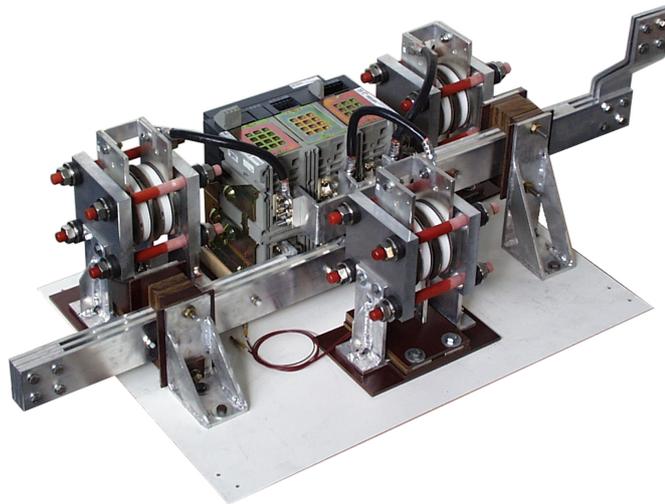


Abbildung 2.8: Aufbau des Hybridschalters nach [7]

2.3 Zusatzbeschaltung

Als Halbleiterschalter wurde ein optisch-triggerbarer Thyristor gewählt, der über Lichtwellenleiter mit der Steuerung verbunden wird. Um den Thyristor richtig zu zünden, ist es notwendig, dass der Thyristor den Lichtimpuls intern in einen elektrischen Impuls umwandelt. Dies kann aber nur dann passieren, wenn eine ausreichend hohe Anoden-Kathodenspannung anliegt. Ab einer Spannung von ca. 50 V kann der Thyristor schalten, die Verzögerungszeit beträgt dabei $100 \mu s$ (ist aber für den Anwendungsfall zu lange). Bei einer Spannung von 80 V beträgt die Verzögerungszeit $10 \mu s$. Es war nun

das Ziel, eine Schaltung zu entwickeln, welche mind. 80 V bereitstellt um auch nahe des Spannungsnulldurchgangs den Thyristor präzise schalten zu können.

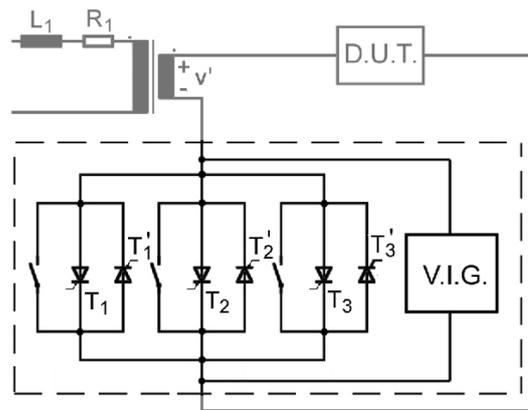


Abbildung 2.9: Topologie des Hybridschalters mit Zusatzbeschaltung nach [7]

Um dies zu realisieren wurde nach [7] die Schaltung in Abbildung 2.10 entwickelt, im Weiteren Zusatzschaltung genannt. Für den Zeitpunkt, bei dem der Zusatzspannungsimpuls angelegt wird, kann die Wechselspannungsquelle als Gleichspannungsquelle modelliert werden. T_1 ist der optisch-triggerbare Thyristor der das Einschalten des Hochstroms an dem Prüfobjekt vornimmt. R_L begrenzt den Strom der über C_1 fließt. Der Thyristor T schaltet den Kondensator C_1 mit dem Thyristor T_1 parallel. Wenn die Spannung an der Wechselspannungsquelle gerade im Nulldurchgang ist, dann sind die Vorimpedanz und der Thyristor parallel geschaltet. Der Kondensator muss im Fall einer Zündung genügend Strom liefern um beide Komponenten zu versorgen. Um ein verlässliches schalten zu gewährleisten, wird der Kondensator auf eine Spannung von 140 V dimensioniert und mit einer Kapazität von $100 \mu F$ ausgelegt. Der größte Fehlerfall wäre, wenn die Zusatzbeschaltung bei einer negativen Halbwelle zündet. Dadurch kann ein Strom von 2 kA über C_1 auftreten. Realisiert wurde der Kondensator C_1 durch drei seriell geschaltete $400 \mu F$ Kapazitäten, welche in der Lage sind, auch die Stromspitzen im Fehlerfall abzudecken.

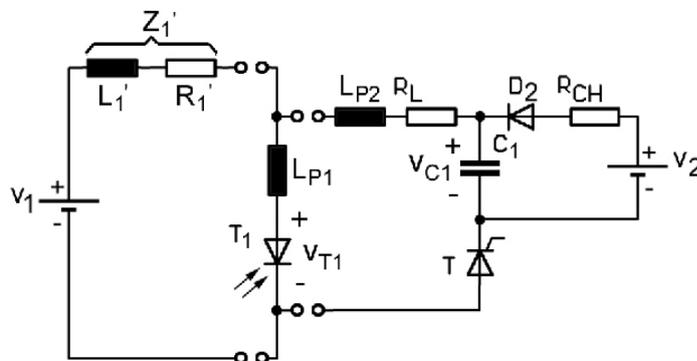


Abbildung 2.10: Zusatzbeschaltung des Hybridschalters nach [7]

3 Technische Spezifikationen und Methoden

3.1 Anforderungen an die Steuerungshardware

Konzept

Das Konzept der zu entwickelnden Steuerung ist in Abbildung 3.1 dargestellt. Dabei ist das Gesamtsystem auf zwei Komponenten aufgeteilt worden. Zunächst soll in der Messkabine die Steuerung von zusätzlichen freien Kanälen möglich sein und die Parametrierung für die Hybridschalter erfolgen (im Folgenden zusammenfassend Synchronsteuerung genannt). Die Ansteuerung der Hybridschalter muss unmittelbar in der Nähe der installierten Hybridschalter erfolgen, also direkt im Prüffeld. Aus sicherheitstechnischen Gründen muss die Kommunikation galvanisch getrennt vom Prüffeld erfolgen (Einsatz von Lichtwellenleiter). Im Folgenden werden die einzelnen Kompo-

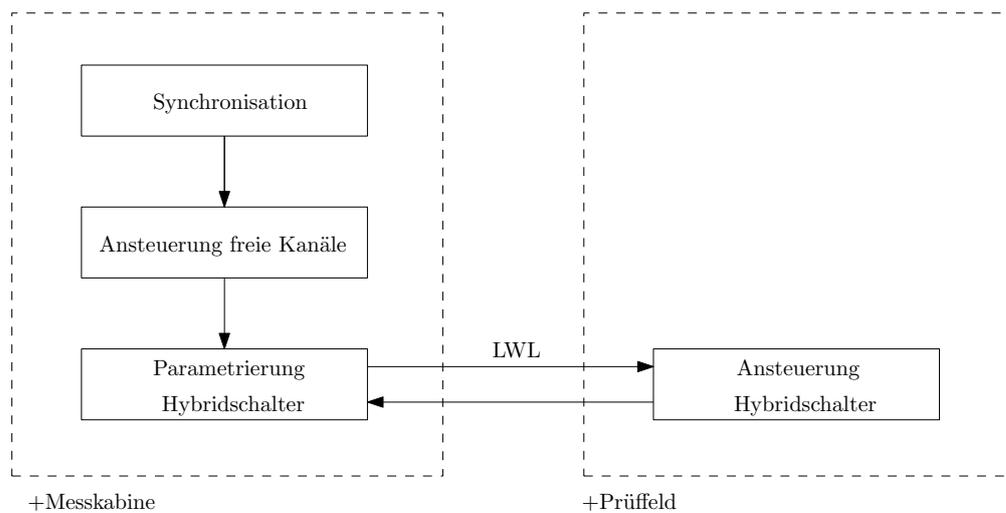


Abbildung 3.1: Blockdiagramm Spezifikationen

nenten genau spezifiziert.

Synchronisation

Die Synchronisation mit der Netzspannung ist für das exakte Einschalten essentiell. Hierfür ist eine geeignete Schaltung zu evaluieren und zu entwerfen. Für die Steuerung kommt ein Mikrocontroller zum Einsatz. Es ist eine Genauigkeit auf 1° gefordert, dies entspricht einer Zeit von $55,5 \mu s$ bei einer 50 Hz Netzspannung. Um diese Genauigkeit sicher zu gewährleisten, wird die Steuerung darauf ausgelegt auf $10 \mu s$ genau zu

arbeiten. Als Eingangssignal für die Synchronisation dient ein Differentialverstärker, welcher die eingestellte Netzspannung entweder im Verhältnis 1:10 oder 1:100 teilt. Dabei ist die maximale Amplitude der Ausgangsspannung des Differentialverstärkers 7 V.

- *max. Eingangsspannung* ± 7 V
- *Ausgangsspannung*: 0 V – 5 V (TTL)
- *Reaktionszeit des Systems*: 10 μ s (= 100 kHz)
- *gewählte Frequenz*: 16 MHz

Datenübertragung

Für die Übertragung der Parameter zwischen grafischer Benutzeroberfläche und Synchronsteuerung soll eine USB Schnittstelle verwendet werden, welche eine serielle Schnittstelle (RS232) emuliert. Für die Übertragung zwischen der Synchronsteuerung und der Ansteuerung der Hybridschalter müssen Lichtwellenleiter verwendet werden.

- *PC* \leftrightarrow *Synchronsteuerung*: USB-Verbindung
- *Synchronsteuerung* \leftrightarrow *Ansteuerung Hybridschalter*: Lichtwellenleiter

Steuerung - freie Kanäle

Die Anzahl der zu steuernden Kanäle soll insgesamt 16 betragen, wobei ein Kanal für die Ansteuerung des Hybridschalters vorzusehen ist. Damit verbleiben zur freien Verfügung 15 weitere Kanäle, die für die Steuerung eines Prüfablaufs verwendet werden können. Jeder Kanal ist extra abzusichern und soll ebenfalls mit dem Spannungsnetz synchronisiert steuerbar sein.

- *Anzahl freier Kanäle*: 15
- *Steuerspannung*: 24 V
- *max. Ausgangsstrom / Kanal*: 500 mA

Ansteuerung - Hybridschalter

Für die Ansteuerung der Hybridschalter müssen verschiedene Teilschaltungen spezifiziert werden.

Ansteuerung optisch-triggerbarer Thyristoren: Da es sich bei den Thyristoren um optisch triggerbare Bauelemente handelt, muss die Übertragung der Zündimpulse über Lichtwellenleiter erfolgen. Hierfür ist ebenfalls eine geeignete Beschaltung zu entwerfen und zu definieren. Für die Erzeugung der Lichtimpulse sind die Laserdioden PL90 von Osram zu verwenden. Der Thyristor wird daher über Lichtwellenleiter mit der Steuerschaltung verbunden.

- *Lichtleistung*: 40 mW
- *Haltestrom*: 100 mA
- *max. Impulsdauer*¹: 100 μ s

¹Wird die Impulsdauer überschritten kann es nach [8] zu einer Zerstörung der Laserdioden kommen.

Ansteuerung mechanischer Schalter: Für die Ansteuerung des mechanischen Schalters (Leistungsschalter MasterNW20) muss eine Einschaltspule angesteuert werden. Dabei ist auch eine parallel geschaltete Freilaufdiode vorzusehen.

- *Steuerspannung:* 220 VDC
- *max. Ausgangsstrom:* 2 A

Zusatzbeschaltung: Das Einschalten des Hybridschalters in der Nähe des Spannungsnulldurchgangs ist problematisch, da hier eine zu kleine Anoden- Kathodenspannung am Thyristor anliegt. Der eingesetzte Thyristor benötigt allerdings Spannung um einen optischen Zündimpuls zu detektieren, weswegen man im Bereich des Spannungsnulldurchganges nicht ohne zusätzliche Maßnahmen schalten kann. Dafür wird eine eigens entwickelte Beschaltung nach [7] eingesetzt, welche es ermöglicht bei kleinen Winkeln zu schalten. Diese Schaltung muss noch geeignet dimensioniert werden und entsprechende Messungen sind ebenfalls vorzunehmen.

Mechanische Schaltverzögerungsmessung: Um ein hybrides Schalten des Systems zu ermöglichen, sollte die Schaltverzögerung mit der Synchronsteuerung messbar sein und dementsprechend auch kalibrierbar, da die mechanischen Schalter eine Streuung der Schaltverzögerung aufweisen. Um die Schaltverzögerung des mechanischen Schalters messen zu können, wird ein Meldekontakt verwendet, welcher eine konstante Verzögerung zum Hauptkontakt besitzt.

3.2 Anforderungen an die Steuerungssoftware

Bedienungsfeld

Die Bedienung in der Messkabine soll sowohl hardwaremäßig (nur Auslösung des Prüfvorgangs) als auch softwaremäßig realisiert werden. Die Auslösung für die Prüfung ist hardwaremäßig mit einem Taster auszuführen. Einzustellen sind die Einschaltzeiten der freien Kanäle und der Hybridschalter in μs (nicht in Grad) mit einer Auflösung von $10 \mu s$ und die Ausschaltzeiten für die freien Kanäle. Für die Entwicklung bietet sich hier LabView an.

Steuerung

Um eine stabile Ausführung des Programms zu gewährleisten, soll auf den Einsatz von Interrupts weitgehend verzichtet werden. Da die Steuerung durch einen Mikrocontroller erfolgt und der zu erwartende Programmablauf komplex ist, wird die Programmiersprache C bevorzugt. Nach erfolgter Synchronisation soll der Ablauf der Schaltvorgänge chronologisch abgearbeitet werden. Dafür ist Abbildung 3.2 eine Vorgabe.

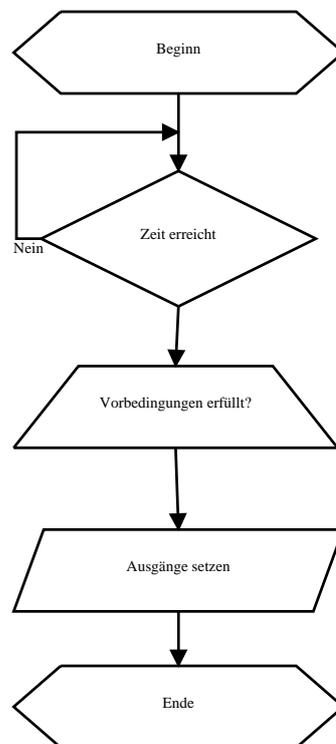


Abbildung 3.2: Spezifikation - Programmablaufdiagramm

3.3 Methoden - Entwicklung

Da auf bestehende Entwicklungen aufgebaut wird, müssen alle anzusteuern und verwendeten Komponenten genau analysiert werden. Sind die erforderlichen Parameter nicht bekannt, so werden diese durch Messungen bestimmt. Dadurch ist es möglich, die Spezifikationen und Anforderungen für die Entwicklung der Hardware/Software zu bestimmen. Die Vorgehensweise bei der Entwicklung ist in 3.3 dargestellt.

Bei der Entwicklung wird zusätzlich auch modular vorgegangen, um später Änderungen einzelner Komponenten leichter vornehmen zu können. Im Speziellen werden Ansteuerungen und Controller hardwaremäßig getrennt ausgeführt.

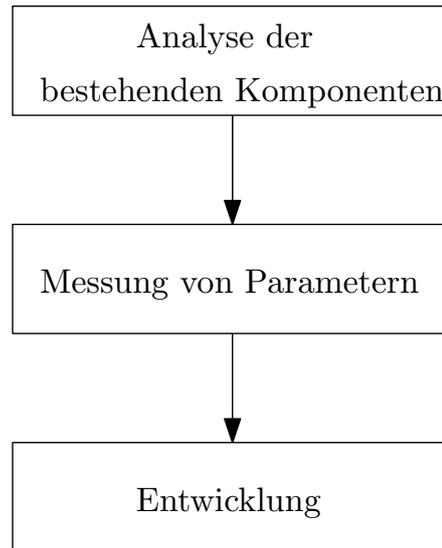


Abbildung 3.3: Methode - Entwicklung

3.4 Methoden - Simulationen

Um die entworfene Hardware zu verifizieren, wird die Funktionsweise durch eine reine softwaremäßige Simulation getestet werden. Dabei wird nach Abbildung 3.4 zunächst das Verhalten unter normalen Betriebsbedingungen analysiert. Darüber hinaus werden bei sensiblen Komponenten auch erweiterte Simulationen ausgeführt, um das Verhalten bei nicht idealen Betriebsbedingungen festzustellen. Die eingesetzte Simulationssoftware ist PSpice, wobei mit den jeweilig passenden Modellen gearbeitet wird (nicht ideale Modelle der Bauelemente).

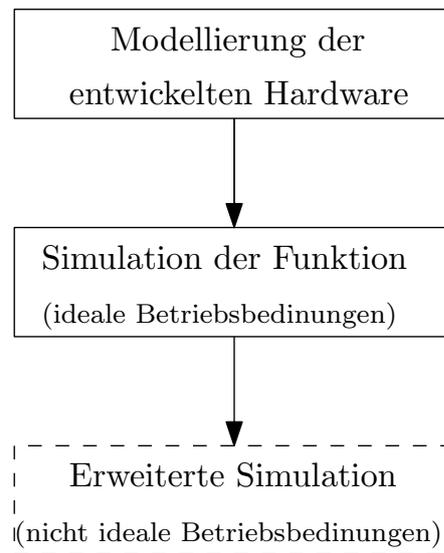


Abbildung 3.4: Methode - Simulation

3.5 Methoden - Validierung

Für die Validierung der Hardware und der Software steht Hardware-in-the-loop zu Verfügung. Mit Hardware-in-the-loop ist es möglich elektrische Netzwerke auf einem Echtzeitsystem zu simulieren und auf das Verhalten einer angeschlossenen Hardware zu testen.

Die Methode der Hardware-in-the-loop Validierung wird hier konkret so verwendet, dass auf dem Echtzeitsystem das Spannungsnetz sowie die Hybridschalter simuliert werden. Das entworfene Steuerungssystem wird in die Hardware-in-the-loop Umgebung eingebunden und die Funktion wie in Abbildung 3.5 validiert.

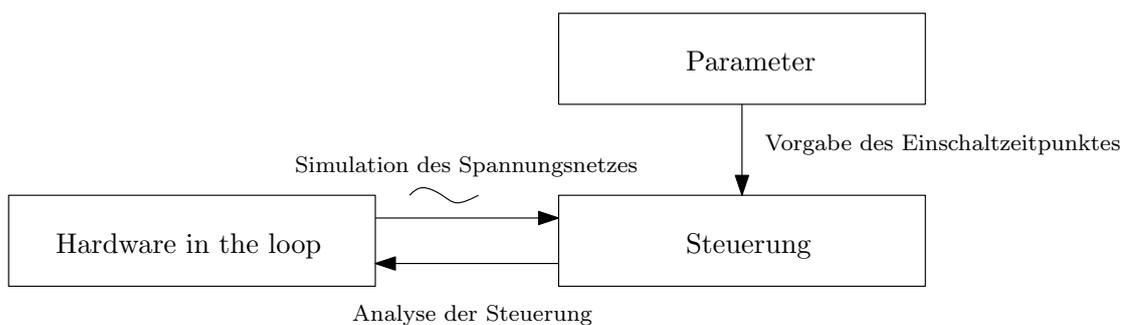


Abbildung 3.5: Methode - Hardware in the loop

4 Grundlagen

4.1 Einschaltverhalten von L-R Wechselstromkreisen

Um das Einschaltverhalten eines Hybridschalters zu beschreiben, werden zunächst die stationären Zustände vor bzw. nach dem jeweiligen Einschaltzeitpunkt analysiert und infolge das Einschwingverhalten berechnet. Es ist für das Einschalten des Hybridschalters von großer Bedeutung, dass diejenigen Thyristoren (positive oder negative Stromrichtung) gezündet werden, welche einen entsprechenden Haltestrom aufbauen können und dass nach dem erfolgten Zünden der Haltestrom nicht unterschritten wird, solange der Strom nicht auf den mechanischen Schalter kommutiert ist.

In der Versuchsanordnung sind vor dem Hybridschalter ein Widerstand und eine Spule geschaltet, damit man einen beliebigen Phasenwinkel zwischen Strom und Spannung einstellen kann. Um sich zu nächst ein Bild von den stationären Spannungen und Strömen machen zu können werden nun der Zeitpunkt (t_{0-}) vor dem Einschalten und der Zeitpunkt (t_{∞}) lange nach dem Einschalten von S_1 analysiert. Abbildung 4.1 zeigt den prinzipiellen Aufbau der Versuchsanordnung und soll im Folgenden als Beispiel zur Analyse dienen. Um die weitere Rechnung zu vereinfachen wird die Vorimpedanz zur Gesamtimpedanz \underline{Z}_1 zusammengefasst.

$$\underline{Z}_1 = R_1 + j\omega L_1 \quad (4.1)$$

$$(4.2)$$

Zum Zeitpunkt t_{0-} wird kein Strom fließen können und es wird die Strangspannung am Schalter auftreten. Zeitpunkt (t_{0-}):

$$\underline{I}_{S_1}(t_{0-}) = 0 \quad (4.3)$$

$$\underline{U}_{S_1}(t_{0-}) = \underline{U}_1 \quad (4.4)$$

Für dem Zeitpunkt t_{∞} wird Strom (der durch die Strangspannung und der Vorimpedanz \underline{Z}_1 gebildet wird) über den Schalter fließen, allerdings wird keine Spannung am Schalter abfallen.

Zeitpunkt (t_{∞}):

$$\underline{I}_{S_1}(t_{\infty}) = \frac{\underline{U}_1}{\underline{Z}_1} \quad (4.5)$$

$$\underline{U}_{S_1}(t_{\infty}) = 0 \quad (4.6)$$

Da der Thyristor den Strom nur kurzzeitig führen wird, sind die stationären Werte nicht von so großem Interesse wie der Einschwingvorgang, bei dem sich der Strom aufbauen wird. Es ist nun möglich, den zeitlichen Verlauf der Stromes $I_{S_1}(\omega t)$ zu bestimmen, der über den Schalter fließen wird. Für den Zeitpunkt t_{0+} kann nun mit den vorher ermittelten Anfangsbedingungen die entsprechende Differentialgleichung gelöst werden (\underline{Z}_1 ist wie oben erwähnt sowohl ohmsch als auch induktiv).

$$u_1(t) = L \frac{di(t)}{dt} + Ri(t) \quad (4.7)$$

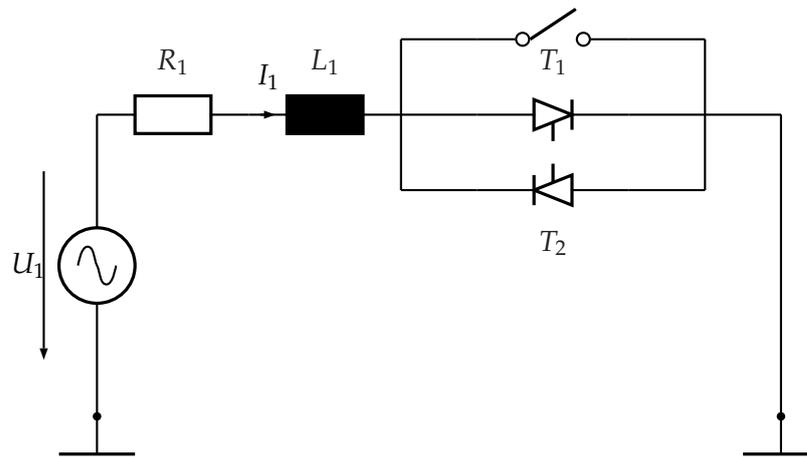


Abbildung 4.1: Ersatzschaltbild Hybridschalter - Einschaltverhalten

Die homogene Lösung der Differentialgleichung lässt sich mit dem Ansatz $i(t) = ke^{\lambda t}$ bzw. $\frac{di(t)}{dt} = k\lambda e^{\lambda t}$ lösen:

$$\begin{aligned} 0 &= Lk\lambda e^{\lambda t} + Rke^{\lambda t} \\ 0 &= L\lambda + R \\ \implies \lambda &= -\frac{R}{L} \end{aligned} \quad (4.8)$$

Die homogene Lösung ergibt sich dadurch zu

$$i_h(t) = ke^{-\frac{R}{L}t} \quad (4.9)$$

Für die partikuläre Lösung muss man die vollständige inhomogene Differentialgleichung mit der Störfunktion aufstellen

$$u_1(t) = \hat{u} \sin(\omega t) = L \frac{di(t)}{dt} + Ri(t) \quad (4.10)$$

Da es sich hier um eine sinusförmige Anregung handelt, kann für die partikuläre Lösung die komplexe Lösung herangezogen werden.

$$\begin{aligned} i_p(t) &= \frac{\hat{u}_1 \sin(\omega t)}{R + j\omega L} = \frac{\hat{u}_1 \sin(\omega t) \underline{Z}^*}{\underline{Z} \underline{Z}^*} = \frac{\hat{u}_1 \sin(\omega t) \underline{Z}^*}{|\underline{Z}|^2} = \frac{\hat{u}_1 \sin(\omega t) |Z| e^{-j\varphi}}{|Z|^2} = \frac{\hat{u}_1 \sin(\omega t) e^{-j\varphi}}{|Z|} \\ &\quad \text{mit } \varphi = \arctan\left(\frac{\omega L}{R}\right) \text{ und } |Z| = \sqrt{R^2 + (\omega L)^2} \end{aligned} \quad (4.11)$$

Die gesamte Lösung setzt sich zusammen aus $i(t) = i_h(t) + i_p(t)$

$$i(t) = \frac{\hat{u}_1}{\sqrt{R^2 + (\omega L)^2}} \left[\sin(\omega t - \varphi) + ke^{-\frac{R}{L}t} \right] \quad (4.12)$$

Zur Bestimmung von k muss man noch die Anfangsbedingungen einsetzen $i(0) = 0$ und $t = t_0$ und erhält:

$$k = -\frac{\frac{\hat{u}_1}{\sqrt{R^2 + (\omega L)^2}} \sin(\omega t_0 - \varphi)}{e^{-\frac{R}{L}t_0}} \quad (4.13)$$

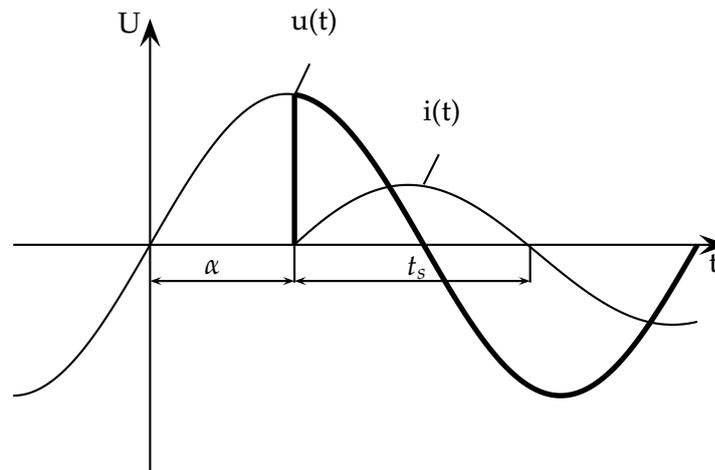


Abbildung 4.2: Einschwingverhalten der ohmsch induktiven Last - Stromverlauf Spannungsverlauf

Damit ergibt sich der Strom zu

$$i(t) = \frac{\hat{u}_1}{\sqrt{R^2 + (\omega L)^2}} \left[\sin(\omega t - \varphi) - \sin(\omega t_0 - \varphi) e^{-\frac{R}{L}(t-t_0)} \right] \quad (4.14)$$

In Abbildung 4.2 ist der Verlauf von Strom und der angelegten Spannung dargestellt. Für den Einschaltzeitpunkt ist es nun wichtig zu analysieren, wie sich der Strom in Abhängigkeit der angelegten Spannung verhält und wie lange dieser Strom in Abhängigkeit vom Einschaltzeitpunkt fließen kann (bis der Haltestrom unterschritten wird). Die Stromflussdauer wird mit t_s bezeichnet und der Steuerwinkel mit α . Dazu betrachten wir zwei Extremfälle, eine rein ohmsche Last sowie auch eine rein induktive Last¹.

Rein ohmsche Last: $\varphi = 0, \omega L = 0, \omega t_s = \pi - \alpha$

$$\begin{aligned} 0 \leq \omega t \leq \alpha : \quad & i(t) = 0 \\ \alpha \leq \omega t \leq \pi : \quad & i(t) = \frac{\hat{u}_1}{\sqrt{R^2 + (\omega L)^2}} [\sin(\omega t)] \end{aligned} \quad (4.15)$$

Rein induktive Last: $\varphi = \frac{\pi}{2}, \omega R = 0, \omega t_s = 2\pi - 2\alpha$

$$\begin{aligned} 0 \leq \omega t \leq \alpha : \quad & i(t) = 0 \\ \alpha \leq \omega t \leq 2\pi - \alpha : \quad & i(t) = \frac{\hat{u}_1}{\sqrt{R^2 + (\omega L)^2}} [-\cos(\omega t) + \cos(\alpha)] \\ 2\pi - \alpha \leq \omega t \leq 2\pi : \quad & i(t) = 0 \end{aligned} \quad (4.16)$$

Wenn man nun die erste Periode während des Einschaltvorganges betrachtet, ist ersichtlich, dass je nachdem zu welchem Zeitpunkt ωt_0 der Zündvorgang stattfindet, sich der Strom in Richtung der angelegten Spannung ändert. Mit anderen Worten bedeutet dies, wird während der positiven Halbwelle gezündet, so bildet sich ein positiver Strom aus bzw. wenn während der negativen Halbwelle gezündet wird, bildet sich

¹Vgl.[9, Kapitel 8]

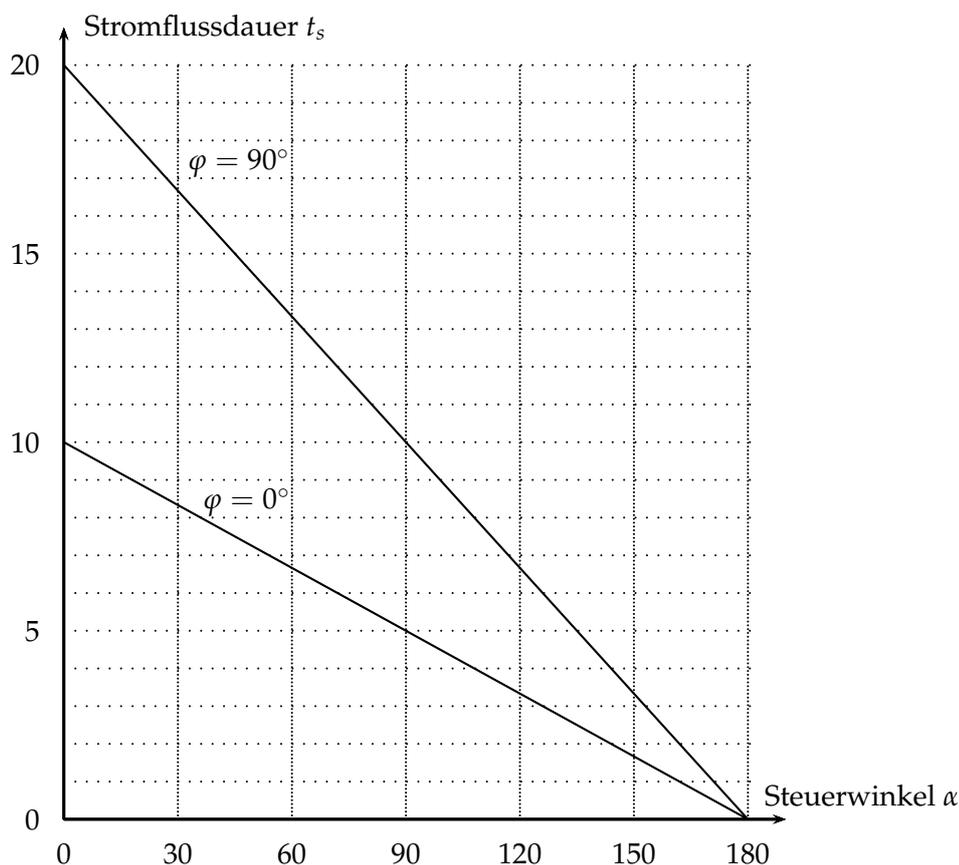


Abbildung 4.3: Steuerkennlinie des Hybridschalters

ein negativer Strom aus. Die Zeitdauer t_s sowie auch die Amplitude von $i(t)$ ist von der Impedanz abhängig. In Abbildung 4.3 ist die Steuerkennlinie für den Einschaltvorgang des Hybridschalters zu sehen. Es sei angemerkt, dass der Steuerwinkel unabhängig vom Phasenwinkel φ gewählt werden kann, da es sich bei dem Einschaltvorgang um kein periodisches Ansteuern handelt, wie dies z.B. beim Wechselstromsteller der Fall ist. Zusammenfassend lässt sich also sagen, dass alleine durch Kenntnis des Spannungsverlaufs eine Aussage über den qualitativen Stromverlauf möglich ist. Will man zusätzlich noch die Stromleitdauer t_s und den Momentanwert $i(t)$ bestimmen, muss man auch die Impedanz \underline{Z} ermitteln.

4.2 Einschalten des Hybridschalters unter Nebenbedingungen

Es stellt sich nun die Frage, welche Nebenbedingungen beim Einschaltvorgang existieren, um die Thyristoren zünden zu können. Zunächst einmal muss jenen Thyristoren (positive oder negative Stromrichtung) ein Zündimpuls gegeben werden, die auch einen entsprechenden Haltestrom aufbauen können. Dies lässt sich wie vorher erläutert, alleine durch den Momentanwert der Spannung feststellen. Liegt zum Einschaltzeitpunkt beispielsweise eine positive Spannung an, so müssen auch die positiven Thyristoren mit einem Zündimpuls getriggert werden. Dasselbe gilt für die negati-

Spannung	Stromleitdauer	Zündvorgang Thyristoren	Zusatzbeschaltung
$u(t_0) > U_{z,min} $	$t_s > t_\sigma$	T1	nein
$ U_{h,min} < u(t_0) < U_{z,min} $	$t_s > t_\sigma$	T1	ja
$ U_{h,min} < u(t_0) < U_{z,min} $	$t_s < t_\sigma$	T1 & T2	ja
$u(t_0) < U_{h,min} $	$t_s < t_\sigma$	T2	ja

Tabelle 4.1: Beschreibung der Bedingungen zum Einschaltzeitpunkt für eine Halbwelle

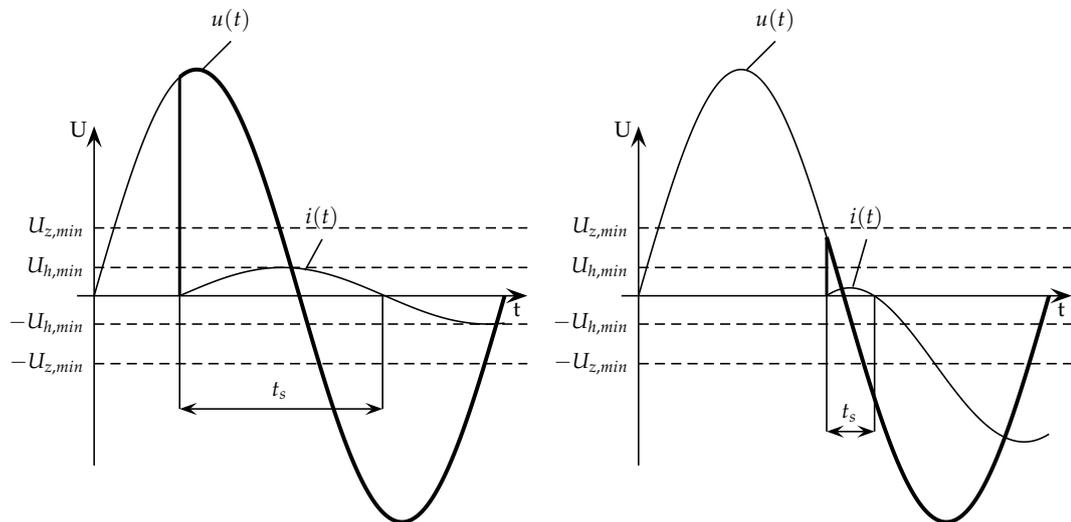
ven Momentanwerte und die negativen Thyristoren (siehe Abbildung 4.4a). Als weitere Nebenbedingungen sind die Eigenschaften des Hybridschalters bzw. im speziellen der Thyristoren, ausschlaggebend. Um einen Zündimpuls überhaupt zu detektieren ist eine minimale Anoden- Kathodenspannung von ca. $|U_{z,min}| = 80 \text{ V}$ notwendig. Sollte der Momentanwert der Spannung zum Einschaltzeitpunkt darunter liegen, so muss mit der Zusatzbeschaltung eine entsprechende Spannung zwischen Anode und Kathode bereit gestellt werden (siehe Abbildung 4.4b).

Ebenfalls muss überprüft werden, ab welcher Spannung es keinen Sinn mehr macht den Thyristor zu zünden, weil sich, abhängig von der Impedanz, der minimale Haltestrom² von 100 mA nicht mehr aufbauen kann. Diese Spannungsschwelle wird mit $|U_{h,min}|$ bezeichnet und tritt im Bereich des Nulldurchgangs auf. Sollte nun ein Einschaltzeitpunkt gewählt werden, der knapp vor dem Nulldurchgang liegt und die Spannungsschwelle wird zu diesem Zeitpunkt unterschritten, muss gewartet werden, bis die Spannung bei der entgegengesetzten Richtung der Thyristoren die Spannungsschwelle $|U_{h,min}|$ überschritten hat. Erst dann wird gezündet werden (siehe Abbildung 4.4c).

Zuletzt muss noch die Stromleitdauer t_s berücksichtigt werden, da der gezündete Thyristor solange den Strom führen muss bis dieser auf den mechanischen Schalter kommutiert ist. Da die Streuung des Einschaltverzuges des mechanischen Schalters $t_\sigma = 1 \text{ ms}$ beträgt, muss gewährleistet sein, dass der Thyristor für diesen Zeitraum den Haltestrom nicht unterschreitet. Dies kann aber passieren, wenn ebenfalls kurz vor dem Nulldurchgang gezündet wird, und der Momentanwert der Spannung noch größer ist als $|U_{h,min}|$. In diesem Fall kann es sein, dass der Haltestrom unterschritten wird und noch nicht auf den mechanischen Schalter kommutiert ist. Es muss dann die andere Thyristorengruppe zusätzlich gezündet werden, sobald die Spannung in der nachfolgenden Halbwelle $|U_{h,min}|$ wieder überschritten ist (siehe Abbildung 4.4d).

Daraus lässt sich nun folgende Tabelle (siehe Tabelle 4.1) für eine Halbwelle ableiten. Diese Tabelle kann nun als Grundlage für eine mögliche Ansteuerung dienen.

²[6, Datenblatt Thyristor]

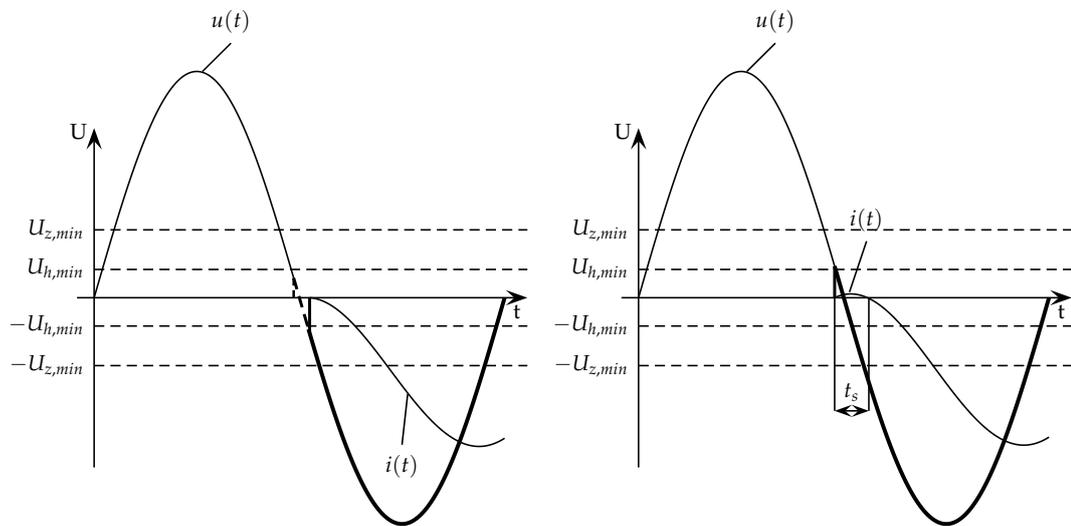


(a) Einschaltvorgang bei $u(t) > U_{z,min}$

(b) Einschaltvorgang bei $u(t) < U_{z,min}$,

$$u(t) > U_{h,min},$$

$$t_s \geq t_\sigma$$



(c) Einschaltvorgang bei $u(t) < U_{z,min}$,

$$u(t) < U_{h,min},$$

$$t_s \leq t_\sigma$$

(d) Einschaltvorgang bei $u(t) < U_{z,min}$,

$$u(t) > U_{h,min},$$

$$t_s \leq t_\sigma$$

Abbildung 4.4: Einschaltvorgang zu verschiedenen Zeitpunkten mit $\varphi = 75^\circ$

5 Einschaltvorgang des Dreiphasensystems

5.1 Allgemeiner Aufbau des Dreiphasensystems

Für das Dreiphasensystem ist zu untersuchen, ob und wie sich die Bedingungen für das Einschalten des Hybridschalters ändern. Dies bedeutet, dass die Rückwirkung beim Schalten der einzelnen Phasen aufeinander untersucht werden müssen.

Um das Einschaltverhalten des dreiphasigen Hybridschalters zu analysieren, müssen die möglichen Prüfanordnungen einzeln betrachtet werden. Ein vereinfachtes allgemeines Ersatzschaltbild ist in Abbildung 5.1 zu sehen und wird im Folgenden für die Analyse verwendet. Die durchgeführten Prüfungen mit dem Hybridschalter sind im Allgemeinen Kurzschlussprüfungen. Die dargestellten Impedanzen sind die eingestellten Vorimpedanzen und das zu prüfende Betriebsmittel bildet den jeweiligen Kurzschluss.

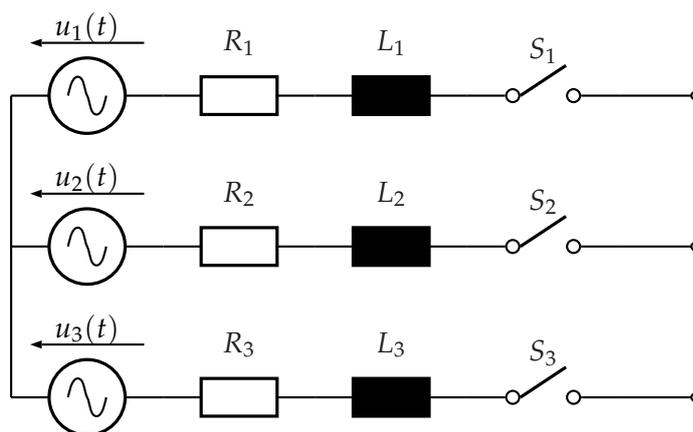


Abbildung 5.1: Dreiphasiges System

Es kann dabei davon ausgegangen werden, dass sowohl das Drehstromsystem als auch die Impedanzen vollkommen symmetrisch aufgebaut sind. Für das Drehstromsystem gilt allgemein:

$$u_1(t) = \hat{u}_1 \sin(\omega t) \tag{5.1}$$

$$u_2(t) = \hat{u}_2 \sin\left(\omega t - \frac{2\pi}{3}\right) \tag{5.2}$$

$$u_3(t) = \hat{u}_3 \sin\left(\omega t - \frac{4\pi}{3}\right) \tag{5.3}$$

$$\text{mit } \hat{u}_1 = \hat{u}_2 = \hat{u}_3 \tag{5.4}$$

Des Weiteren sind die Impedanzen aus einem ohmschen und einem induktiven Anteil

gebildet.

$$R_1 = R_2 = R_3 \quad (5.5)$$

$$L_1 = L_2 = L_3 \quad (5.6)$$

Sofern komplex gerechnet werden darf, gilt für die Impedanz \underline{Z}_n

$$\underline{Z}_n = R_n + j\omega L_n \quad (5.7)$$

Als Prüfschaltungen werden die angeführten Verschaltungen verwendet und müssen daher für den Einsatz des Hybridschalters betrachtet werden (es werden hier die miteinander kurzgeschlossenen Phasen angeführt):

- L1 - N
- L2 - N
- L3 - N
- L1 - L2
- L2 - L3
- L3 - L1
- L1 - L2 - L3
- L1 - L2 - L3 - N

Im Unterschied zum einphasigen System, ändern sich beim zweiphasigen bzw. dreiphasigen System nach jeder Schalthandlung die Spannungen an den noch offenen Schaltern. Es müssen nun die Spannungen an den Hybridschaltern vor dem Einschalten ermittelt werden ($u_s(t_{0-})$) sowie die Ströme lange nach dem Einschalten ($i_s(t_\infty)$), wobei in diesem Fall davon ausgegangen werden kann, dass das System eingeschwungen ist und mit \underline{I}_s komplex gerechnet werden kann. Das ist insofern von Interesse, da man wie vorher bereits erläutert, entscheiden muss, welche Thyristoren (positive oder negative Polarität) eingeschaltet werden müssen bzw. ob man sich zum Einschaltzeitpunkt in der Nähe des Nulldurchgangs befindet und daher eine Verschiebung des Einschaltzeitpunktes stattfinden muss. Die Ströme zum Zeitpunkt t_∞ sind deswegen von Interesse, um einen Ansatz für das Einschwingverhalten des Stromes zu erhalten. Da einige Fälle äquivalent sind, werden zur Illustration nur die vom Prinzip her unterschiedlichen Prüfschaltungen analysiert.

5.2 Einschaltverhalten L1 - N

Bei einem einphasigen Kurzschluss mit dem N-Leiter wird immer nur die betreffende Phase eingeschaltet (hier L1). Um sich zunächst ein Bild von den Spannungen und Strömen machen zu können, werden nun der Zeitpunkt (t_{0-}) vor dem Einschalten und der Zeitpunkt (t_∞) lange nach dem Einschalten von S_1 analysiert. Abbildung 5.2 zeigt den prinzipiellen Aufbau der Versuchsanordnung und soll im Folgenden als Beispiel zur Analyse dienen. Es liegt hier der exakt selbe Fall wie schon im Abschnitt 4.1 behandelt worden ist. Der Vollständigkeit halber sind hier in Tabelle 5.1 aber die Spannungen und Ströme eingetragen. Die Bedingungen für das Einschalten lassen sich dadurch direkt aus Tabelle 4.1 übernehmen und sind in Tabelle 5.2 dargestellt.

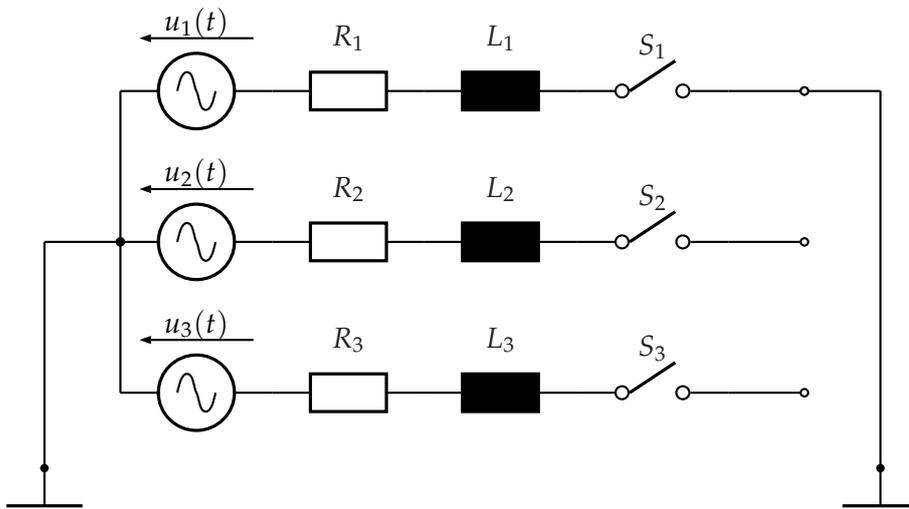


Abbildung 5.2: Dreiphasiges System mit Kurzschluss von L1 - N

Schalter 1		
	$t_{1,0^-}$	$t_{1,\infty}$
$u_{S1}(t)$	$u_1(t)$	0
I_{S1}	0	$\frac{U_1}{Z_1}$

Tabelle 5.1: Spannungen und Ströme für L1-N

5.3 Einschaltverhalten L1 - L2

Eine weitere Prüfanordnung ist die in Abbildung 5.3 dargestellte, mit einem zweiphasigen Kurzschluss. Hierbei werden die Schalter S_1 und S_2 allerdings nicht gleichzeitig geschlossen. Um zum jeweiligen Einschaltzeitpunkt die Spannungen an den einzelnen Schaltern im Vorhinein bestimmen zu können, werden die Schalter mit kleinen Verzögerungen ($10 \mu s - 20 \mu s$) nacheinander geschaltet. Wie in Abbildung 5.3 schon dargestellt, wird zuerst der Schalter S_1 geschlossen. Hier ist kein hybrides Schalten erforderlich, da nach dem Einschalten kein Strom fließen kann.

$$u_{S1}(t_{1,0^-}) = 0 \text{ und } I_{S1}(t_{1,\infty}) = 0 \quad (5.8)$$

Der nächste Schritt ist nun das Einschalten des Schalters S_2 . Hierfür muss nun die Spannung bestimmt werden, die beim Einschalten anliegen wird. Legt man in Abbildung 5.3 eine Masche, so ergibt sich für

$$u_{S2}(t_{2,0^-}) = u_2(t_{2,0^-}) - u_1(t_{2,0^-}) = -u_{1,2}(t_{2,0^-}). \quad (5.9)$$

Der Strom in L_2 ergibt sich nun durch die verkettete Spannung sowie die zwei Impedanzen Z_1 und Z_2 .

$$I_2 = \frac{-U_{1,2}}{Z_1 + Z_2} \quad (5.10)$$

Zusammengefasst können die Spannungen und Ströme aus Tabelle 5.3 entnommen werden. Die Bedingungen für das Einschalten lassen sich auch hier wieder direkt aus

Spannung	Stromleitdauer	Zündvorgang Thyristoren	Zusatzbeschaltung
$u_1(t_0) > U_{z,min} $	$t_s > t_\sigma$	T1	nein
$ U_{h,min} < u_1(t_0) < U_{z,min} $	$t_s > t_\sigma$	T1	ja
$ U_{h,min} < u_1(t_0) < U_{z,min} $	$t_s < t_\sigma$	T1 & T2	ja
$u_1(t_0) < U_{h,min} $	$t_s < t_\sigma$	T2	ja

Tabelle 5.2: Beschreibung der Bedingungen zum Einschaltzeitpunkt für eine Halbwelle
- Prüfschaltung L1-N

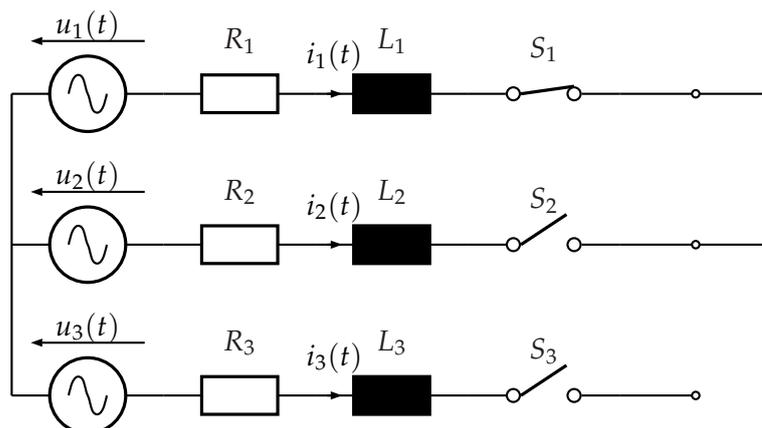


Abbildung 5.3: Dreiphasiges System - Kurzschluss auf L1-L2 mit offenem Schalter in Phase L2

Tabelle 4.1 übernehmen. Dabei gilt es zu beachten, dass die Spannung mit der hier gerechnet werden muss, immer $-u_{1,2}$ ist.

5.4 Einschaltverhalten L1 - L2 - L3

Bei einem dreipoligen Kurzschluss wird ebenfalls davon ausgegangen, dass die Schalter gestaffelt eingeschaltet werden, um definierte Momentanwerte der Spannungen (und somit auch der Ströme) zum jeweiligen Einschaltzeitpunkt zu haben. Der Grund für dieses Vorgehen liegt darin, dass man kaum ein exakt gleichzeitiges Einschalten der Thyristoren gewährleisten kann. Daher schaltet man die drei Phasen bewusst nicht zum gleichen Zeitpunkt, um so eine zufällige Schaltreihenfolge der drei Phasen zu vermeiden.

Der Einschaltvorgang beim dreipoligen Kurzschluss gestaltet sich derart, dass bevor der eigentliche Prüfzyklus beginnt, der Schalter S_1 geschlossen wird (mechanisch). Es fließt hier noch kein Strom, da der Stromkreis noch nicht geschlossen ist. Während des Prüfzyklus wird dann S_2 geschlossen und die Spannung $u_{S_2}(t_{2,0-})$ bzw. der Strom $I_{S_2}(t_{2,\infty})$ verhalten sich analog zum Prüfaufbau L1-L2. Nun müssen noch die Spannung $u_{S_3}(t_{3,0-})$ bzw. der Strom $I_{S_3}(t_{3,\infty})$ für S_3 ermittelt werden. Für den Zeitpunkt $(t_{3,0-})$ vor dem Einschalten von S_3 wird Abbildung 5.4 analysiert und in Folge über die Kirchhoff'schen Maschengleichungen die Spannung ermittelt. Es wird dabei wieder davon

	Schalter 1		Schalter 2		Schalter 3	
	$t_{1,0^-}$	$t_{1,\infty}$	$t_{2,0^-}$	$t_{2,\infty}$	$t_{3,0^-}$	$t_{3,\infty}$
$u_{S1}(t)$	0	0	0	0	0	0
I_1	0	0	0	$\frac{-U_{1,2}}{Z_1+Z_2}$	0	0
$u_{S2}(t)$	0	$-u_{1,2}$	$-u_{1,2}$	0	0	0
I_2	0	0	0	$\frac{-U_{1,2}}{Z_1+Z_2}$	0	0
$u_{S3}(t)$	0	0	0	0	0	0
I_3	0	0	0	0	0	0

Tabelle 5.3: Spannungen und Ströme für L1-L2

Spannung	Stromleitdauer	Zündvorgang Thyristoren	Zusatzbeschaltung
$-u_{12}(t_0) > U_{z,min} $	$t_s > t_\sigma$	T1	nein
$ U_{h,min} < -u_{12}(t_0) < U_{z,min} $	$t_s > t_\sigma$	T1	ja
$ U_{h,min} < -u_{12}(t_0) < U_{z,min} $	$t_s < t_\sigma$	T1 & T2	ja
$-u_{12}(t_0) < U_{h,min} $	$t_s < t_\sigma$	T2	ja

Tabelle 5.4: Beschreibung der Bedingungen zum Einschaltzeitpunkt für eine Halbwelle
- Prüfschaltung L1 L2

5 Einschaltvorgang des Dreiphasensystems

ausgegangen, dass hier der größte Spannungsabfall am Schalter selbst stattfindet.

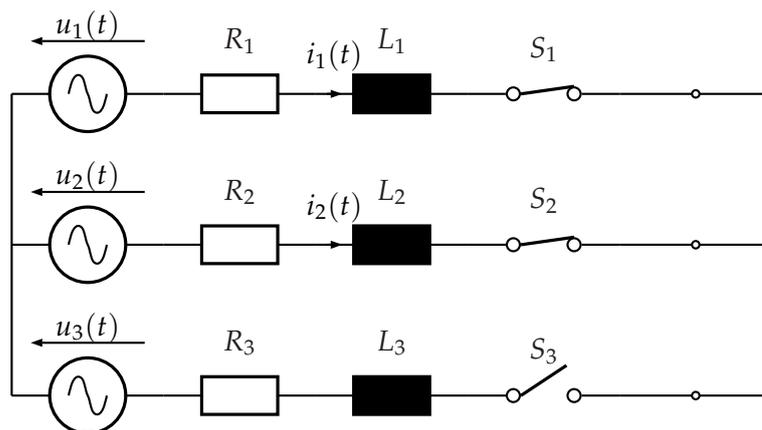


Abbildung 5.4: Dreiphasiges System - Kurzschluss auf L1-L2-L3 mit offenem Schalter in Phase L3

Es wird nun die Spannung $u_{S_3}(t_{0-})$ ermittelt:

$$\text{Masche I: } -u_1(t) + R_1 i_1(t) + L_1 \frac{di_1(t)}{dt} - R_2 i_2(t) - L_2 \frac{di_2(t)}{dt} + u_2(t) = 0 \quad (5.11)$$

$$\text{Masche II: } -u_2(t) + R_2 i_2(t) + L_2 \frac{di_2(t)}{dt} - u_{S_3}(t) + u_3(t) - R_3 i_3(t) - L_3 \frac{di_3(t)}{dt} = 0 \quad (5.12)$$

$$\text{mit: } i_1(t) = -i_2(t) \text{ und } i_3(t) = 0$$

ergibt sich die Spannung am Schalter S_3 vor dem Einschaltzeitpunkt t_{0-} .

$$u_{S_3}(t_{0-}) = \frac{2 u_3(t_{0-}) - (u_1(t_{0-}) + u_2(t_{0-}))}{2} \quad (5.13)$$

Um die Phasenlage der berechneten Spannung zu bestimmen wird nun mit 5.4 die Gleichung 5.13 zu

$$u_{S_3}(t_{0-}) = \frac{\hat{u}_1 \left[2 \sin(\omega t - \frac{4\pi}{3}) - (\sin(\omega t) + \sin(\omega t - \frac{2\pi}{3})) \right]}{2} \quad (5.14)$$

Mit Hilfe des Additionstheorems $\sin x + \sin y = 2 \sin\left(\frac{x+y}{2}\right) \cos\left(\frac{x-y}{2}\right)$ kann man 5.14 zu

$$u_{S_3}(t_{0-}) = \frac{\hat{u}_1 \left[2 \sin(\omega t - \frac{4\pi}{3}) - \left(2 \sin\left(\frac{2\omega t - \frac{2\pi}{3}}{2}\right) \cos\left(\frac{-\pi}{3}\right) \right) \right]}{2} \quad (5.15)$$

umformen. Rechnet man nun $\cos\left(\frac{-\pi}{3}\right) = \frac{1}{2}$ und $\sin(\omega t - \frac{\pi}{3}) = \sin(\pi - \omega t + \frac{\pi}{3}) = \sin(\frac{4\pi}{3} - \omega t) = -\sin(\omega t - \frac{4\pi}{3})$ aus, so ergibt sich

$$u_{S_3}(t_{0-}) = \frac{3u_3(t_{0-})}{2} \quad (5.16)$$

Nun muss noch der Strom I_3 lange nach dem Einschaltzeitpunkt ermittelt werden. Da sich nach dem Einschalten von S_3 ein symmetrisches Drehstromsystem ausbilden kann,

	Schalter 1		Schalter 2		Schalter 3	
	$t_{1,0^-}$	$t_{1,\infty}$	$t_{2,0^-}$	$t_{2,\infty}$	$t_{3,0^-}$	$t_{3,\infty}$
$u_{S1}(t)$	0	0	0	0	0	0
\underline{I}_1	0	0	0	$\frac{\underline{U}_{1,2}}{\underline{Z}_1 + \underline{Z}_2}$	$\frac{\underline{U}_{1,2}}{\underline{Z}_1 + \underline{Z}_2}$	$\frac{\underline{U}_1}{\underline{Z}_1}$
$u_{S2}(t)$	0	$-u_{1,2}$	$-u_{1,2}$	0	0	0
\underline{I}_2	0	0	0	$\frac{-\underline{U}_{1,2}}{\underline{Z}_1 + \underline{Z}_2}$	$\frac{-\underline{U}_{1,2}}{\underline{Z}_1 + \underline{Z}_2}$	$\frac{\underline{U}_2}{\underline{Z}_2}$
u_{S3}	0	$-u_{1,3}$	$-u_{1,3}$	$\frac{3}{2} u_3$	$\frac{3}{2} u_3$	0
\underline{I}_3	0	0	0	0	0	$\frac{\underline{U}_3}{\underline{Z}_3}$

Tabelle 5.5: Spannungen und Ströme für L1-L2-L3

Spannung	Stromleitdauer	Zündvorgang Thyristoren	Zusatzbeschaltung
$\frac{3u(t_0)}{2} > U_{z,min} $	$t_s > t_\sigma$	T1	nein
$ U_{h,min} < \frac{3u(t_0)}{2} < U_{z,min} $	$t_s > t_\sigma$	T1	ja
$ U_{h,min} < \frac{3u(t_0)}{2} < U_{z,min} $	$t_s < t_\sigma$	T1 & T2	ja
$\frac{3u(t_0)}{2} < U_{h,min} $	$t_s < t_\sigma$	T2	ja

Tabelle 5.6: Beschreibung der Bedingungen zum Einschaltzeitpunkt für eine Halbwelle
- Prüfschaltung L1L2L3

sind dadurch die Strangspannungen gleich den Phasenspannungen. Die Spannung, welche den Strom \underline{I}_3 hervorruft ergibt sich hier zu \underline{U}_3 .

$$\underline{I}_3(t_{3,\infty}) = \frac{\underline{U}_3}{\underline{Z}_3} \quad (5.17)$$

Mit diesen Ergebnissen ist es nun möglich zu bestimmen, welche Thyristoren zu welchen Zeitpunkten gezündet werden sollen. Man kann nun wieder die Einschaltbedingungen der dritten Phase aus Tabelle 4.1 ableiten und erhält Tabelle 5.6.

6 Entwicklung der Hardware

6.1 Konzept

In Abbildung 6.1 ist der gesamte Aufbau schematisch dargestellt, wobei die zu entwickelnde Hardware örtlich getrennt aufgebaut ist. In der Messkabine befindet sich die sogenannte Synchronsteuerung, welche über einen PC parametrieren werden soll. Die Synchronsteuerung soll freie anschließbare Kanäle synchron mit dem Spannungsnetz schalten, die Ansteuerung der Hybridschalter parametrieren und ggf. die Ansteuerung der Hybridschalter mit dem Spannungsnetz synchronisieren können. Die Ansteuerung der Hybridschalter befindet sich jedoch außerhalb der Messkabine im Prüffeld, daher muss aus Sicherheitsgründen die Kommunikation mittels Lichtwellenleiter erfolgen.

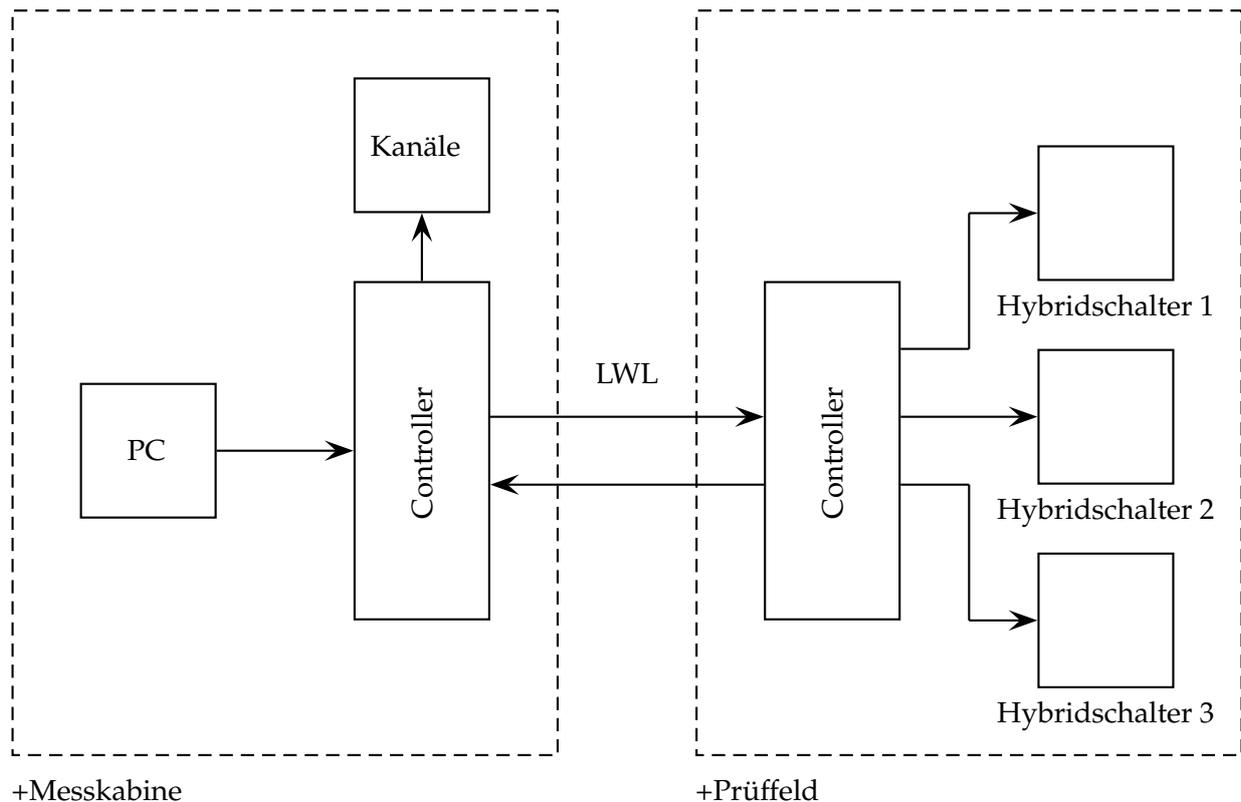


Abbildung 6.1: Blockschaltbild - Konzept der Ansteuerung

6.2 Controller

Anforderungen

Ein Controller stellt die Basis für die Synchronsteuerung sowie auch für die Ansteuerung der Hybridschalter dar. Die Controller sind für beide Schaltungen gleich aufgebaut. Dies hat den Vorteil, dass zunächst nur eine Controllerplatine dimensioniert werden muss und ein derartiger Aufbau auch im „Stand-Alone“ Betrieb arbeiten kann. Somit sind die einzelnen Steuerungen prinzipiell voneinander unabhängig betreibbar. Die Aufgaben des Controllers sind:

- Synchronisation mit dem Netz
- Parametrierung über USB-Schnittstelle sowie über Lichtwellenleiter
- Steuerung der Schaltungen für die freien Kanäle ¹
- Steuerung der Schaltungen für die Hybridschalter ²

In Abbildung 6.2 ist der Controller schematisch dargestellt.

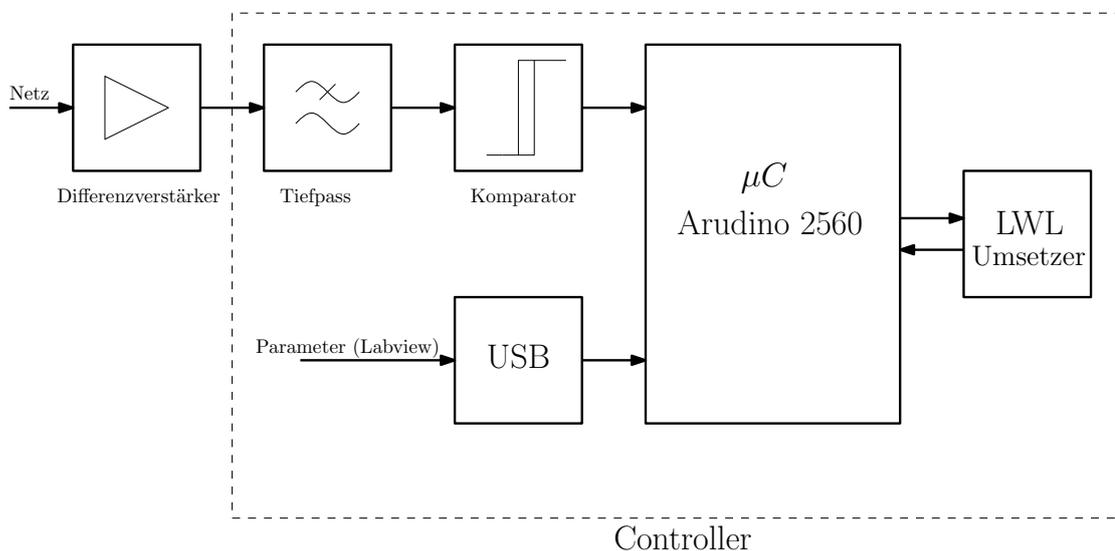


Abbildung 6.2: Blockschaltbild - Controller

Microcontroller

Um die Anforderungen die in Kapitel 3 an den Mikrocontroller gestellt wurden zu erfüllen, muss im Speziellen auf die Anzahl der I/O's und der Kommunikationsschnittstellen geachtet werden.

Der gewählte Typ ist hier der ATmega2560 der Firma Atmel, da diese Type die Anforderungen an die I/O's mit elf Ports (entspricht ca. 85 I/O's) und vier Kommunikationsschnittstellen mehr als ausreichend erfüllt. Ein weiteres Kriterium ist, dass die Open-Source Plattform Arduino ein bereits voll bestücktes Board mit dem ATmega2560 entwickelt hat und dieses sofort einsetzbar ist. Die wichtigsten Spezifikationen nach [10] sind in Tabelle 6.1 angegeben.

¹Wenn der Controller für die Synchronsteuerung eingesetzt wird.

²Wenn der Controller für die Hybridschalter eingesetzt wird.

Mikrocontroller ATmega2560	
Betriebsspannung	5V
Eingangsspannung (empfohlen)	7-12V
Eingangsspannung (Grenzen)	6-20V
Digital I/O's	54
Analog I/O's	16
DC Strom per I/O Pin	40 mA
DC Strom für 3.3V Pin	50 mA
Flash Memory	256
SRAM	8 KB
EEPROM	4 KB
Taktfrequenz	16 MHz

Tabelle 6.1: Spezifikationen Mikrocontroller ATmega2560

In Abbildung 6.3 ist der Mikrocontroller Arduino - ATmega 2560 dargestellt.

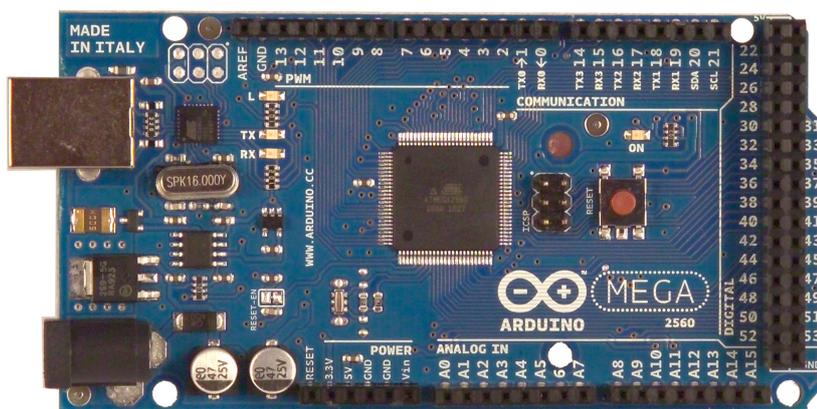


Abbildung 6.3: Mikrocontroller Arduino - ATmega2560 [10]

Tiefpass

Um eine präzise Synchronisation zu gewährleisten wird die Ausgangsspannung des Differenzverstärkers³ zunächst über einen Tiefpass geführt. Die Dimensionierung erfolgt nach [11, S.835-837].

³Der verwendete Differenzverstärker sollte aufgrund der gewählten Versorgungsspannung der Operationsverstärker von +/-15 V nicht über +/-13 V Ausgangsspannung besitzen. Für die Versuche wurde ein Differentialastkopf Testec TT-SI 9001x10/x100 verwendet, welcher eine maximale Ausgangsspannung von +/-7 V besitzt.

Die Übertragungsfunktion eines Tiefpasses 1. Ordnung lautet allgemein

$$A(s_n) = \frac{A_0}{1 + a_1 s_n} \quad (6.1)$$

wobei s_n die komplexe normierte Frequenz mit $s_n = \frac{j\omega}{\omega_g}$ und a_1 einen reellen Koeffizienten darstellen. Für Filter 1. Ordnung gilt unabhängig von der Filtertype $a_1 = 1$. Bei Filter höherer Ordnung gilt das im Allgemeinen nicht mehr.

Einfache Tiefpässe wie ein RC-Glied, haben den Nachteil, dass sich ihre Filtercharakteristik mit der Belastung ändert. Um dies zu verhindern, werden bevorzugt aktive Tiefpässe verwendet. Abbildung 6.4 zeigt den für den Controller eingesetzten Tiefpass 1. Ordnung. Die Übertragungsfunktion dieses Tiefpasses ergibt sich zu

$$A(s_n) = -\frac{R_{T2}/R_{T1}}{1 + \omega_g R_{T2} C_{T1} s_n} \quad (6.2)$$

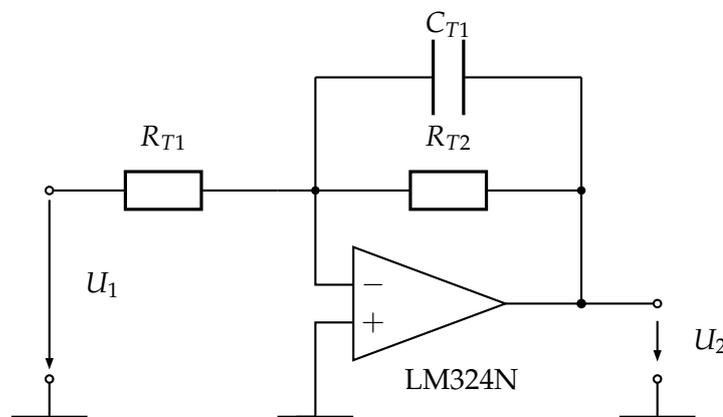


Abbildung 6.4: Tiefpass

Um den Tiefpass dimensionieren zu können, muss die Verstärkung A_0 , die Kapazität C_{T1} und die Grenzfrequenz f_g vorgegeben werden. Für eine korrekte Synchronisation mit dem Spannungsnetz ist es notwendig, dass das Synchronisationssignal eine konstante Phasenverschiebung gegenüber der Netzspannung hat. Dies ist nur dann möglich, wenn die Grenzfrequenz des Filters hinreichend groß ist. Es wird also für die Grenzfrequenz $f_g = 5 \text{ kHz}$, für $A_0 = 1$ und die Kapazität $C_{T1} = 30 \text{ nF}$ gewählt. Die Widerstände ergeben sich durch Koeffizientenvergleich zu

$$R_{T2} = \frac{1}{2\pi f_g C_{T1}} \text{ und } R_{T1} = -\frac{R_{T2}}{A_0} \quad (6.3)$$

und berechnen sich zu $R_{T2} = 1,061 \text{ k}\Omega$ sowie $R_{T1} = 1,061 \text{ k}\Omega$, wobei hier für $R_{T1} = R_{T2} = 1,1 \text{ k}\Omega$ gewählt wurde.

Komparator und Halbwellengleichrichter

Der Komparator ist die für die Synchronisation wichtigste Komponente, er vergleicht die Eingangsspannung U_1 (welche am positiven Eingang anliegt) mit der Spannung am negativen Eingang. Prinzipiell schaltet der Komparator die positive Versorgungsspannung an den Ausgang, vorausgesetzt die Eingangsspannung am positiven Eingang ist höher als die anliegende Eingangsspannung am negativen Eingang. Für den

kleine Hysterese angestrebt, daher wurde ein Verhältnis $R_{K1}/R_{K2} = 1/50$ gewählt. Hier wurde $R_{K1} = 1 \text{ k}\Omega$ und $R_{K2} = 50 \text{ k}\Omega$ gewählt.

Da der Komparator die Eigenschaft hat, den Ausgang jeweils nahe der Versorgungsspannung auszusteuern, muss, bevor das Signal an einen Mikrocontroller angeschlossen wird, eine Pegelumwandlung stattfinden. Dies erfolgt über einen sogenannten Halbwellengleichrichter, welcher die negative Ausgangsspannung des Komparators unterdrückt und durch eine Abschwächung das Signal auf TTL-Pegel umsetzt.

Die Verstärkung des Halbwellengleichrichters ergibt sich wie bei einer invertierenden Verstärkerschaltung nach [12, S.160] zu

$$U_3 = -\frac{R_{K3}}{R_{K4}}U_2. \quad (6.6)$$

Da die positive Ausgangsspannung des Komparators in jedem Fall $+15 \text{ V}$ beträgt, muss die Verstärkung des Halbwellengleichrichters $1/3$ betragen um ein TTL-Signal zu erhalten. Daher wurden die Widerstände $R_{K3} = 1 \text{ k}\Omega$ und $R_{K4} = 3 \text{ k}\Omega$ gewählt. Die verwendeten Dioden D_1, D_2 sind schnelle Kleinsignaldioden der Type 1N4148.

Serielle Übertragung mittels Lichtwellenleiter

Wie Eingangs schon erwähnt, ist aus Sicherheitsgründen die Kommunikation zwischen der Synchronsteuerung und der Ansteuerung für den Hybridschalter über Lichtwellenleiter auszuführen. Dabei wurde auf die HFBR-Serie der Firma Agilent zurückgegriffen. Diese Komponenten machen es möglich, die serielle UART-Schnittstelle eines Mikrocontrollers auf Lichtwellenleiter umzusetzen. Für diese Umsetzung ist auch keine weitere Pegelumwandlung notwendig und kann direkt nach [14] erfolgen. Die maximal übertragbare Distanz beträgt 2.7 km mit einer Datenrate von 160 Mbd . Diese Parameter sind bei weitem ausreichend, da es sich bei der Distanz zwischen Synchronsteuerung und der Ansteuerung der Hybridschalter maximal um $20\text{-}25 \text{ m}$ handelt. In Abbildung 6.6 ist die Schaltung für die Umsetzung von einem TXD-Signal (Senden der UART Schnittstelle) auf Lichtwellenleiter gezeigt.

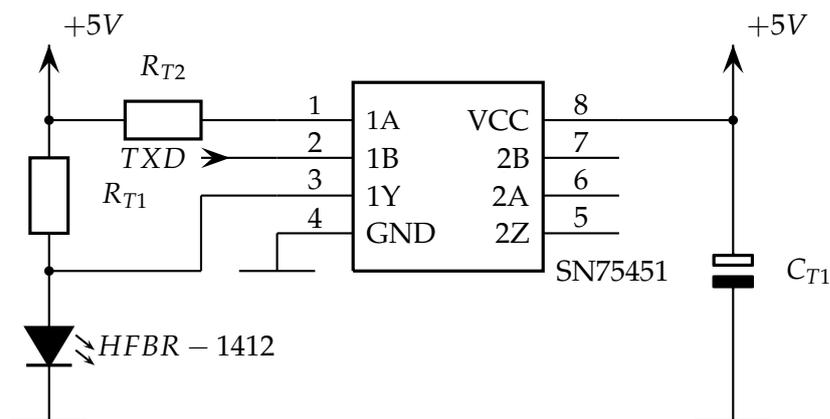


Abbildung 6.6: Serielle Übertragung LWL-Transmitter

Für das Senden wird die Komponente HFBR-1412 benötigt, welche im Grunde eine Diode ist und mit einem ST Gehäuse ausgeführt wird um den Lichtwellenleiter aufzustecken. Der Widerstand R_{T1} wird durch den erforderlichen Strom bestimmt der durch HFBR-1412 fließen muss. Nach [14] muss dieser mindestens 48 mA betragen um eine

Strecke von 1750 m sicher übertragen zu können. Dadurch ergibt sich

$$R_{T1} = \frac{V_{CC} - V_F}{I_F} = \frac{5 \text{ V} - 1,5 \text{ V}}{48 \text{ mA}} = 70,4 \, \Omega \quad (6.7)$$

und da R_{T2} lediglich als Pullup-Widerstand gedacht ist, kann man diesen mit $R_{T2} \geq 1 \text{ k}\Omega$ annehmen. Das Bauelement SN75451 hat hier lediglich die Funktion eines schnellen Treibers, damit der Ausgang des Mikrocontrollers nicht belastet wird. Da die beiden Eingänge miteinander auf ein logisches UND geführt werden, muss einer der beiden Eingänge (in diesem Fall 1A) mit dem zuvor dimensioniert Pullup-Widerstand auf „High“ gesetzt werden.

Für das Empfangen wird die Komponente HFBR-2412 verwendet, welche aus einem Phototransistor und einem Vorwiderstand besteht. Um das empfangene Signal dem Mikrocontroller richtig zu übermitteln, muss es über $R_{R1} \geq 1 \text{ k}\Omega$ geführt werden und anschließend mit einem Logikbauelement 74LS04N invertiert werden, damit das ursprünglich übertragene Signal wiedergestellt wird. Abbildung 6.7 zeigt die Schaltung die für die Umwandlung von Lichtwellen auf ein RXD-Signal (Empfang UART-Schnittstelle) notwendig ist.

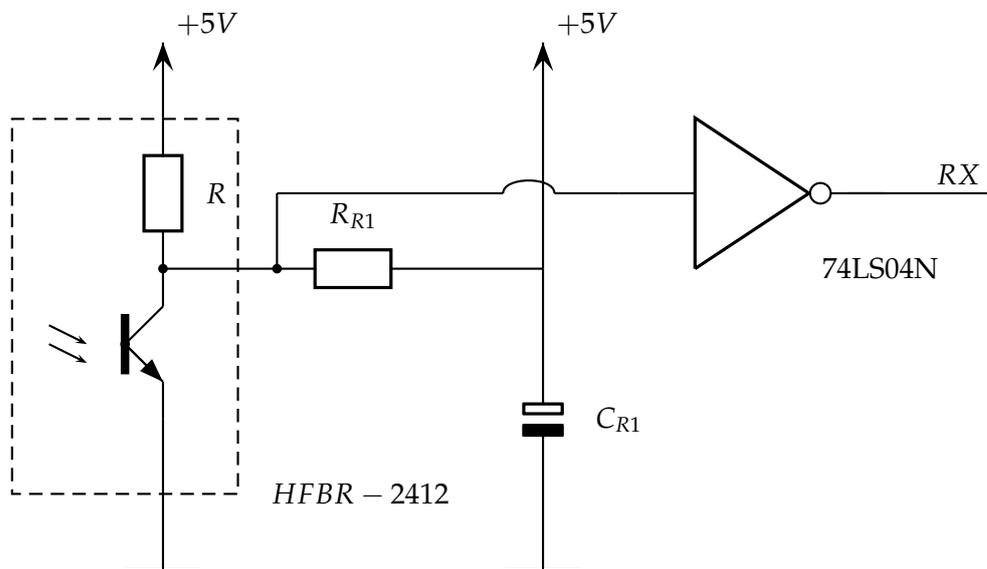


Abbildung 6.7: Serielle Übertragung LWL-Reciever

6.3 Ansteuerschaltung eines Hybridschalters

Anforderungen

Die Ansteuerung eines Hybridschalters erfordert prinzipiell mehrere getrennte Ansteuerkreise. Für den mechanischen Schalter ist eine Einschaltspule für den Einschaltvorgang anzusteuern. Die Thyristoren sind optisch-triggerbar und daher werden sie mit Lichtimpulsen eingeschaltet. Darüber hinaus ist noch eine zusätzlich Zündschaltung für Thyristoren notwendig. Diese muss (wie im Kapitel 6.4) im Falle eines Einschaltvorganges in der Nähe des Spannungsnulldurchgangs eine ausreichend hohe Anoden-Kathodenspannung zur Verfügung stellen. Ein weiterer wichtiger Punkt ist die Messung der mechanischen Verzugszeiten. Diese können mit Hilfe des Status eines Melde-

kontaktes ermittelt werden, da der Meldekontakt eine dem Hauptkontakt proportionale Verzugszeit besitzt. In Abbildung 6.8 ist die Ansteuerschaltung für einen Hybridschalter schematisch dargestellt. Die Aufgaben der Ansteuerung eines Hybridschalters sind kurz zusammengefasst:

- Ansteuerung des mechanischen Schalters (Leistungsschalter)
- Ansteuerung der optisch-triggerbaren Thyristoren (positive und negative Spannungsrichtung)
- Ansteuerung der Zusatzbeschaltung über Thyristoren (positive und negative Spannungsrichtung)
- Abfrage des Status des Meldekontakts

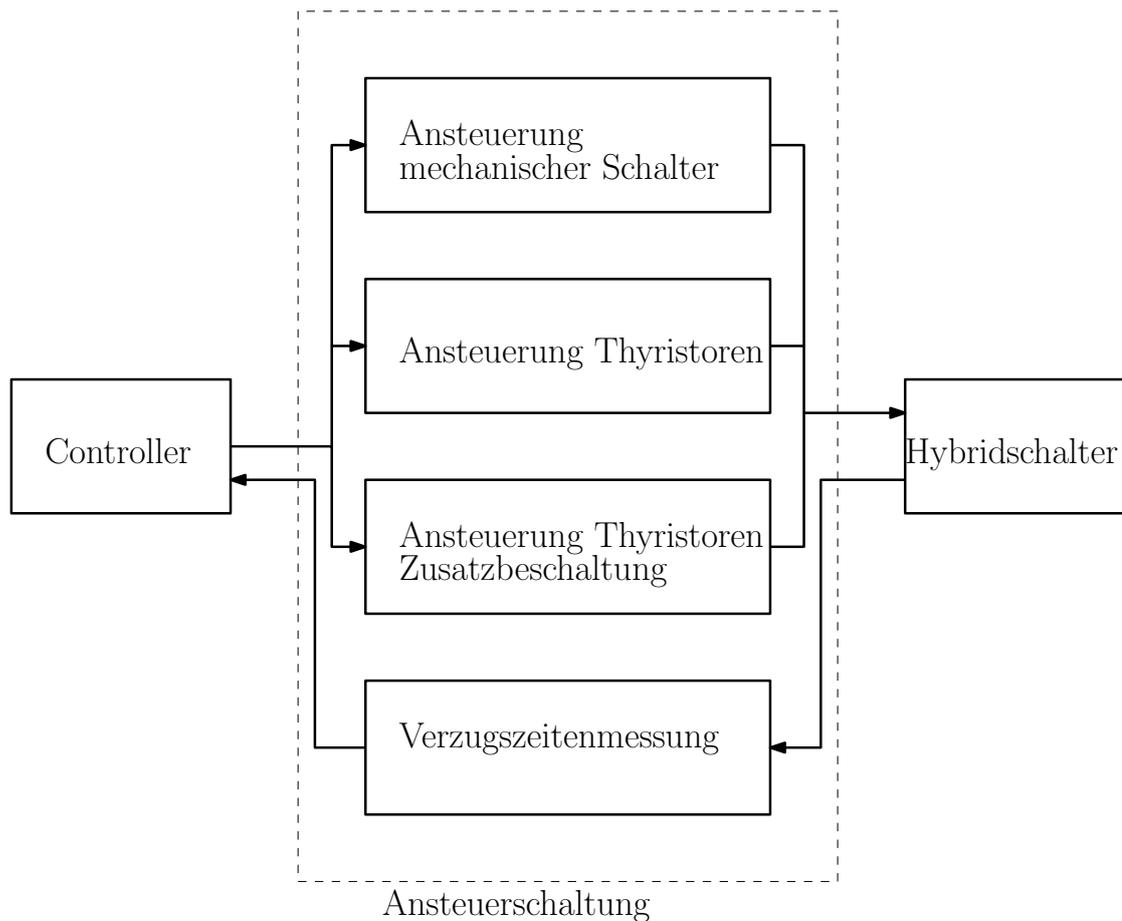


Abbildung 6.8: Blockschaltbild - Ansteuerung

Ansteuerung mechanischer Schalter

Der eingesetzte mechanische Schalter ist ein Leistungsschalter der Firma Schneider-Electric von der Type Masterpact NW20 HF. Dieser Leistungsschalter ist mit einem Motorantrieb ausgestattet, welcher einen Kraftspeicher spannt. Dieser Kraftspeicher wird dazu verwendet, den Leistungsschalter zu betätigen (Ein- bzw. Ausschalten). Aktiviert wird er durch eine Einschaltspule, die standardmäßig mitgeliefert wird. Diese benötigt nach [4] eine Steuerspannung $U_{Steuer} = 220 \text{ VDC}$, damit der Leistungsschalter eingeschaltet werden kann.

Um den Leistungsschalter nun über den Controller bedienen zu können, muss das TTL-Signal des Controllers die Steuerspannung von 220 VDC schalten. Für diese Aufgabe bietet sich der Einsatz eines Solid-State Relais an. Solid-State Relais sind Halbleiterrelais, die auch eine galvanische Trennung zwischen Steuer- und Hauptstromkreis (erreicht durch Optokoppler) besitzen. Das eingesetzte Solid-State Relais ist von der Firma Crydom und der Type D4D07 mit MOSFET Ausgang. Die Spezifikationen sind nach [15]: Um die Ausgänge des Mikrocontrollers so wenig wie möglich zu belasten, wird

Spezifikationen Solid-State Relais	
Steuerspannung	3.5 – 32 VDC
Maximaler Eingangsstrom	28 mA
Maximale Ausgangsspannung	400 VDC
Maximaler Ausgangsstrom	7 A

Tabelle 6.2: Spezifikationen Solid-State Relais

das Solid-State Relais mit einem Treiberbaustein (MC34151) angesteuert. Es ist wichtig, den Treiberbaustein mit einem Pull-up-Widerstand $R_{P1} \geq 1 \text{ k}\Omega$ zu versehen, da der Treiberbaustein den Ausgang invertiert. Abbildung 6.9 zeigt den Aufbau der Ansteuerung des mechanischen Schalters.

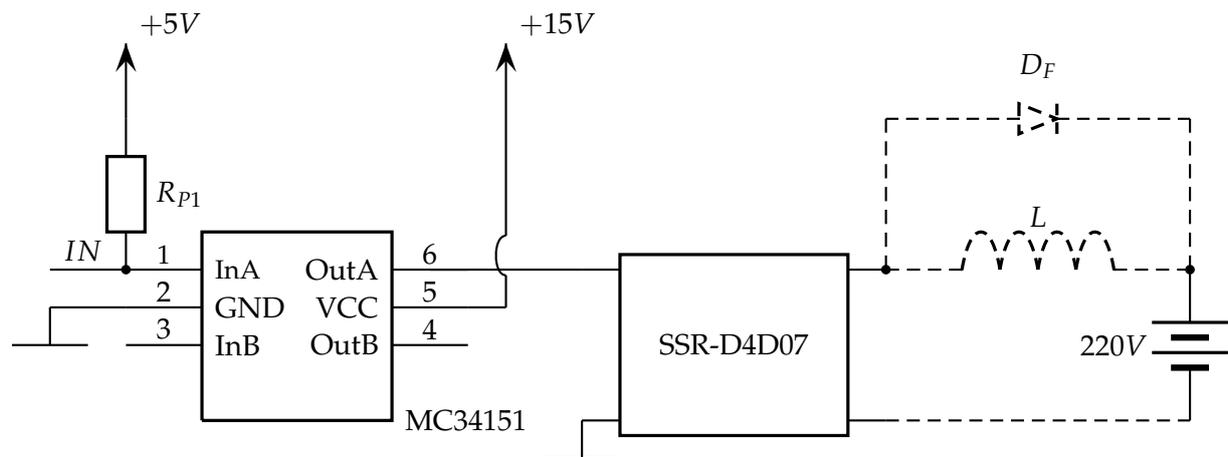


Abbildung 6.9: Ansteuerung mechanischer Schalter

Die strichlierten Elemente sowie die Steuerspannungsquelle sind nicht in die Ansteuerung inkludiert und sind daher extern anzuschließen. Des Weiteren ist eine Freilaufdiode D_F der Einschaltspule parallel zu schalten, um das Solid-State Relais vor Überspannungen zu schützen. Die Einschaltverzögerung, bestimmt in Abschnitt 9.2, beträgt $t_{\text{VerzugSSR}} = 20 \mu\text{s}$ und ist gegen die Verzugszeit der mechanischen Schalter zu vernachlässigen.

Ansteuerung optische Thyristoren

Die im Hybridschalter eingesetzten Halbleiter sind optisch-triggerbare Thyristoren der Firma Eupec vom Typ T4003-N und benötigen daher zum Zünden einen Lichtimpuls.

Nach [6] ist die minimal benötigte Lichtleistung $P_{LM} = 40 \text{ mW}$. Die Thyristoren wurden auch mit passenden Laserdioden der Firma Osram vom Typ PL90 ausgeliefert. Diese Laserdioden sind nach [8] nicht für einen Dauerbetrieb ausgelegt. Daher muss verhindert werden, dass selbst bei einer Fehlfunktion der Logik bzw. des Mikrocontrollers die Dioden nicht überlastet werden. Um zu gewährleisten, dass der Strom nur eine kurze Zeit fließt, muss dieser über einen Kondensator C_{O1} bereitgestellt werden, da nach der Entladung kein Strom mehr fließen kann. Abbildung 6.10 zeigt die prinzipielle Ausführung der Ansteuerung für die optischen Thyristoren nach dem Konzept von [3, S. 60]. Da ein Hybridschalter drei parallel geschaltete Thyristoren pro Polarität besitzt, sind auch drei Laserdioden in eine Ansteuerung für eine Richtung zu integrieren. Für die gesamte Ansteuerung eines Hybridschalters ist die gezeigte Schaltung für beide Spannungsrichtungen auszuführen. Die Dioden $D_{O1} \dots D_{O3}$ sind die oben genannten

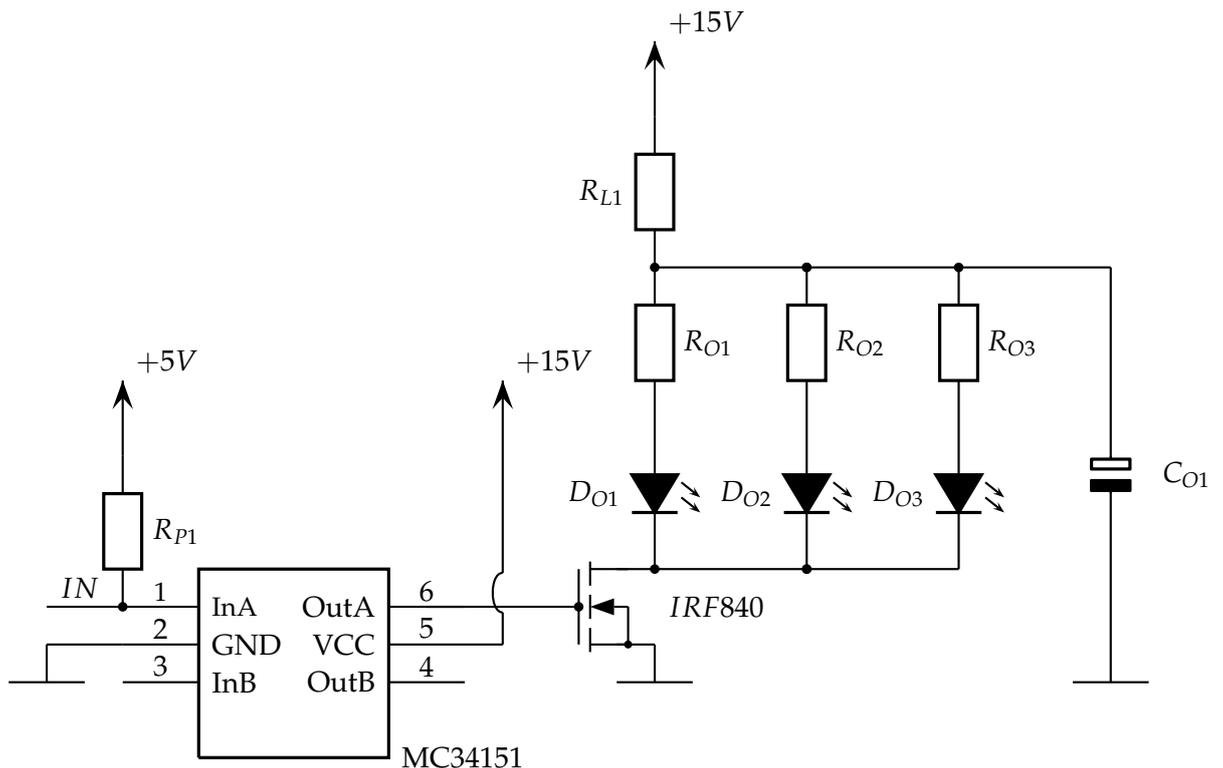


Abbildung 6.10: Zündschaltung für optische Thyristoren

Laserdioden und werden über den MOSFET IRF840 eingeschaltet. Nach der Kennlinie von Vorwärtsstrom über Lichtleistung von [8] ergibt ein Strom von $I_F = 1 \text{ A}$ eine Lichtleistung von $P_{Licht} = 1 \text{ W}$. Damit ergeben sich die Vorwiderstände zu

$$R_{O1...3} = \frac{V_{CC}}{I_F} = \frac{15\text{V}}{1\text{A}} = 15 \Omega. \quad (6.8)$$

Der Kondensator C_{O1} ist nach der Zeitkonstanten zu dimensionieren. Der Zündverzögerung der Thyristoren beträgt nach [6] ca. $5 \mu\text{s}$, das bedeutet, dass der Zündimpuls mindestens für diese Zeit anstehen muss. Ein Kondensator ist ungefähr nach 5τ vollständig entladen und daher ergibt sich die Dimensionierung des Kondensators zu

$$C_{O1} = \frac{5\tau}{R} = \frac{5 \cdot 5 \mu\text{s}}{5\Omega} = 5 \mu\text{F}. \quad (6.9)$$

Der Widerstand R_{L1} ist ein hochohmiger Ladewiderstand für den Kondensator und wird mit $R_{L1} = 10 \text{ k}\Omega$ dimensioniert um zu gewährleisten, dass der Strom im Falle einer zu späten Abschaltung des Transistors, nicht zu einer Zerstörung der Dioden

führt. Der Widerstand R_{p1} ist wieder ein Pull-up Widerstand der mit $R_{p1} \geq 1 \text{ k}\Omega$ zu dimensionieren ist, um den invertierenden Treiberbaustein und damit den Transistor im ausgeschalteten Zustand zu halten.

Bestimmung des Status der Meldekontakte

Um den Hybridschalter ordnungsgemäß betreiben zu können, d.h. um die Thyristoren nicht zu überlasten, ist die Kenntnis der Verzugszeit des mechanischen Schalters von aller größter Bedeutung. Bei jedem Schaltvorgang wird daher die aktuelle Verzugszeit der Hauptkontakte des mechanischen Schalters mit Hilfe von Meldekontakten mit gemessen. Diese Meldekontakte haben selbst wieder Verzugszeiten gegenüber dem mechanischen Schalter die sich nach den Messungen in Abschnitt 9.3 zu konstant $\Delta t_{\text{Meldekontakt}} = 2,4 \text{ ms}$ (nacheilend) ergeben. Bei Änderung des Status eines Meldekontaktes, soll der Mikrocontroller in der Lage sein sofort einen Interrupt auszulösen und die aktuelle Zeit abzuspeichern. Die Abbildung 6.11 zeigt wie der Zustand eines Meldekontaktes abgefragt werden kann. Der verwendete Optokoppler (Typ CNY17) ermöglicht eine Potentialtrennung zwischen dem Mikrocontroller und einem Meldekontakt. Wichtig für die Potentialtrennung ist auch eine separate Spannungsversorgung.

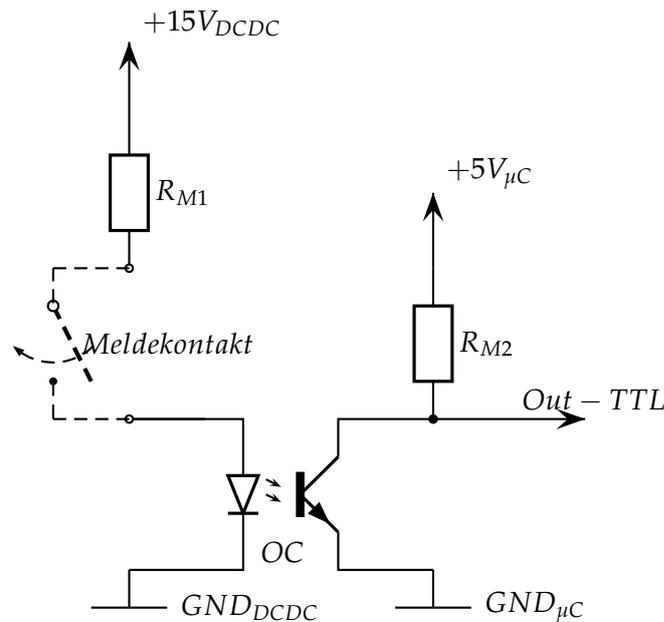


Abbildung 6.11: Meldekontakt Status

Dies wird mittels eines DC/DC Konverters (nicht eingezeichnet - Typ NMS0515) realisiert, der $+15 \text{ V}$ für die Versorgung der Diode bzw. des Meldekontaktes bereitstellt. Der Vorwärtsstrom der LED für den Optokoppler sollte $I_F = 15 \text{ mA}$ betragen um den Einschaltimpuls durch den Phototransistor detektieren zu können. Damit ergibt sich

$$R_{M1} = \frac{15V_{DCDC}}{15 \text{ mA}} = 1 \text{ k}\Omega. \quad (6.10)$$

Der Widerstand R_{M2} ist wieder ein Pull-up Widerstand der mit $R_{M2} \geq 1 \text{ k}\Omega$ zu dimensionieren ist.

6.4 Entwicklung der Zusatzbeschaltung

Anforderungen

Für Einschaltzeitpunkte in der Nähe des Spannungsnulldurchgangs, bei der die Anoden-Kathodenspannung unter 80 V fällt, ergibt sich die Problematik, dass die Zündimpulse der Thyristoren des Hybridschalters nicht mehr zuverlässig detektiert werden können (siehe Abschnitt 2.3). Dafür ist in [7] eine Zusatzbeschaltung entwickelt worden, die in diesem Fall die notwendige Anoden-Kathodenspannung zur Verfügung stellen soll. Die Abbildung 6.12 zeigt diese Zusatzbeschaltung in erweiterter, modifizierter Weise und wird im Folgenden erklärt. Der Transformator stellt eine galvanische Trennung

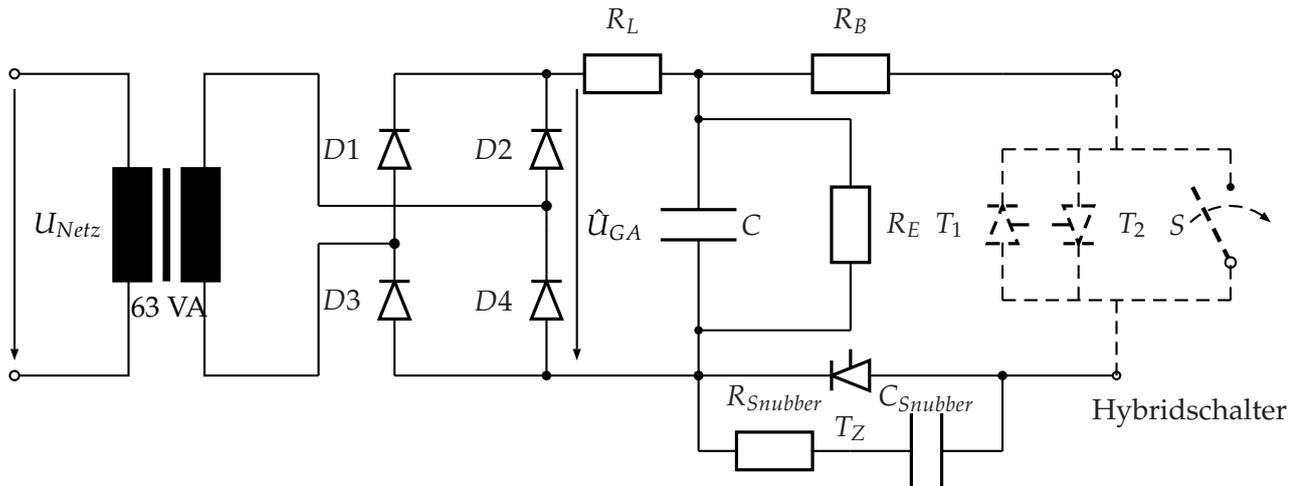


Abbildung 6.12: Zusatzbeschaltung

zwischen dem Spannungsnetz und der Schaltungsseite dar und transformiert die Netzspannung von $U_{Netz} = 230\text{ V}$ auf $U_{Sek} = 115\text{ V}$. Die Dioden $D_1 \dots D_4$ bilden einen Vollweggleichrichter (Typ GBPC12 von Vishay mit einem Maximal-Strom von 5 A) und laden über R_L den Kondensator C auf. Die verwendeten Leistungskondensatoren haben eine Kapazität von $C = 133\text{ }\mu\text{F}$. Bei der Dimensionierung von R_L muss auf die Ausgangsspannung des Gleichrichters Rücksicht genommen werden, die sich mit

$$\hat{U}_{GA} = 115\text{ V} \cdot \sqrt{2} - 1,4\text{ V} = 161,2\text{ V} \quad (6.11)$$

ergibt. Mit Rücksicht auf die Leistungsgrenzen des Ladewiderstandes R_L sollte er, sofern ein 50 W Widerstand zum Einsatz kommt, wie folgt dimensioniert werden

$$R_L = \frac{\hat{U}_{GA}^2}{50\text{ W}} = \frac{161,2^2}{50\text{ W}} = 518\text{ }\Omega. \quad (6.12)$$

Damit ergibt sich für die Ladeschaltung eine Zeitkonstante von

$$\tau = R \cdot C = 518\text{ }\Omega \cdot 133\text{ }\mu\text{F} = 68,9\text{ ms} \quad (6.13)$$

Um sicher zu stellen, dass der Kondensator C der Schaltung im nicht eingeschalteten Zustand entladen ist, wird ein Entladewiderstand, der im Bereich von $5 \cdot R_L$ liegen sollte, parallel geschaltet. Damit ergibt sich der Widerstand zu $R_E \geq 2600\text{ }\Omega$.

Für den Entladevorgang wird der Stromkreis über den Thyristor T_Z geschlossen. Da

nach den Spezifikationen in Abschnitt 3 mit Effektivspannungen bis zu 1200 V am Hybridschalter zu rechnen sind, muss der Thyristor T_Z auf diese Sperrspannung ausgelegt sein. Mit $U_{DRM} = 1800 \text{ V}$ erfüllt der Thyristor N1718NS180 von Westcode diese Forderung. Wichtig für den Einsatz dieses Thyristors ist die Verwendung eines Snubbers, der sich aus einem niederohmigen $R_{Snubber} = 15 \Omega$ ergibt und einer Kapazität die ungefähr das dreifache der Sperrschichtkapazität des Thyristors ausmachen sollte. Die Sperrschichtkapazität ist nach [16] $C_S = 330 \text{ nF}$. Damit ergibt sich

$$C_{Snubber} = 3 \cdot C_S = 3 \cdot 330 \text{ nF} = 990 \text{ nF} \approx 1 \mu\text{F}. \quad (6.14)$$

Der Widerstand R_B ist als Begrenzung des Stromes für den Entladevorgang auszulegen. Hier ist zu beachten, dass der Haltestrom der Thyristoren in jedem Fall aufgebaut werden kann. Der Haltestrom nach [16] beträgt $I_h = 1.5 \text{ A}$ damit ergibt sich

$$R_B \leq \frac{161,2 \text{ V}}{1,5 \text{ A}} = 107,5 \Omega \quad (6.15)$$

Dabei tritt an dem Widerstand für max. 10 ms eine Leistung von $P_{RB} = 241,8 \text{ W}$ auf. Es kann daher trotzdem ein 50 W-Widerstand verwendet werden, da dieser für eine Sekunde mit bis zur zehnfachen Nennleistung betrieben werden kann.

Ansteuerung der Thyristoren für die Zusatzbeschaltung

Für die zuvor dimensionierte Zusatzbeschaltung muss die Zündschaltung des Thyristors ebenfalls in die Ansteuerschaltung für den Hybridschalter integriert werden. Zur Erzeugung eines Zündimpulses wird die in Abbildung 6.13 verwendete Zündschaltung nach [17] verwendet.

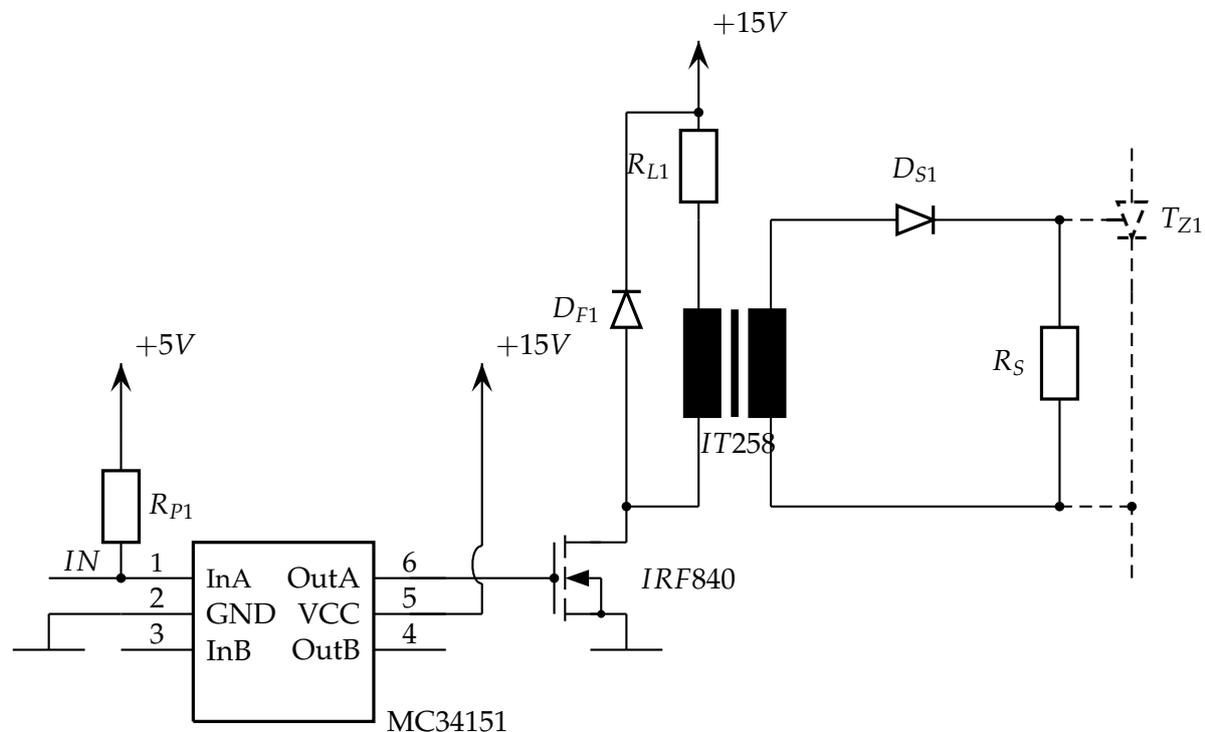


Abbildung 6.13: Zündschaltung für Thyristoren der Zusatzbeschaltung

Der Thyristor N1718NS180 benötigt zum sicheren Zünden einen Gatestrom von 300 mA sowie eine Triggerspannung von 3 V. Dabei sollte der Triggerimpuls für 50 μs anliegen.

Damit ergibt sich eine Spannungszeitfläche von

$$\int u dt = (3 \text{ V} + 0.7 \text{ V}) \cdot 50 \mu\text{s} = 185 \text{ V}\mu\text{s} \quad (6.16)$$

Dadurch konnte der Impulsübertrager IT258 der Firma Schaffner ausgewählt werden, der eine Spannungszeitfläche von $\int u dt = 2580 \text{ V}\mu\text{s}$ und eine Induktivität von $L_p = 2,5 \text{ mH}$ besitzt. Der Widerstand R_{L1} ergibt sich mit $U_{DF1} + U_{GK,TZ1} \approx 2 \text{ V}$ zu

$$R_{L1} = \frac{15 \text{ V} - 2 \text{ V}}{300 \text{ mA}} = 43 \Omega \quad (6.17)$$

Aus der Differentialgleichung

$$u_M = L_M \cdot \frac{di_m}{dt} \approx L_M \cdot \frac{\Delta \hat{i}_m}{\Delta t} \quad (6.18)$$

folgt

$$i_m = \frac{3 \text{ V}}{2,5 \text{ mH}} \cdot 50 \mu\text{s} = 60 \text{ mA} \text{ und } \hat{u}_m = 60 \text{ mA} \cdot 50 \Omega = 6 \text{ V} \quad (6.19)$$

Der Widerstand R_{p1} ist wieder ein Pull-up Widerstand der mit $R_{M1} \geq 1 \text{ k}\Omega$ zu dimensionieren ist. i_m ist der Magnetisierungsstrom und \hat{u}_m der Spitzenwert der übertragenen Spannung.

6.5 Ansteuerschaltung der freien Kanäle

Anforderungen

Zusätzlich zur Ansteuerung des Hybridschalters müssen noch freie Kanäle für die gesamte Steuerung eines Messablaufs vorhanden sein. Diese Kanäle sollen eine Steuerungsspannung von 24 V schalten können um z.B. für das Triggern der Messeinrichtungen, das Ausschalten des Hauptstromkreises oder das Zurücksetzen von Messeinrichtungen verwendet werden können.

Ansteuerung der Kanäle

Abbildung 6.14 zeigt die Ansteuerung von acht Kanälen schematisch. Dabei werden wieder Solid-State Relais verwendet, welche für 24 VDC Ausgangsspannung ausgelegt sind. Die Solid-State Relais können direkt ohne Vorwiderstand angesteuert werden, da sie einen internen Vorwiderstand von 270Ω besitzen. Um die Ausgänge des Mikrocontrollers trotzdem nicht zu überlasten, kommt der Treiberbaustein ULN2803⁵ zum Einsatz. Die Solid-State Relais besitzen nach [18] einen Schaltverzug von max. $t_{on} = 50 \mu\text{s}$, welcher noch innerhalb der geforderten Spezifikationen liegt.

⁵Der ULN2803 besteht aus acht separat schaltbaren Darlington-Schaltungen und kann somit an den Ausgängen bis 500 mA Strom liefern

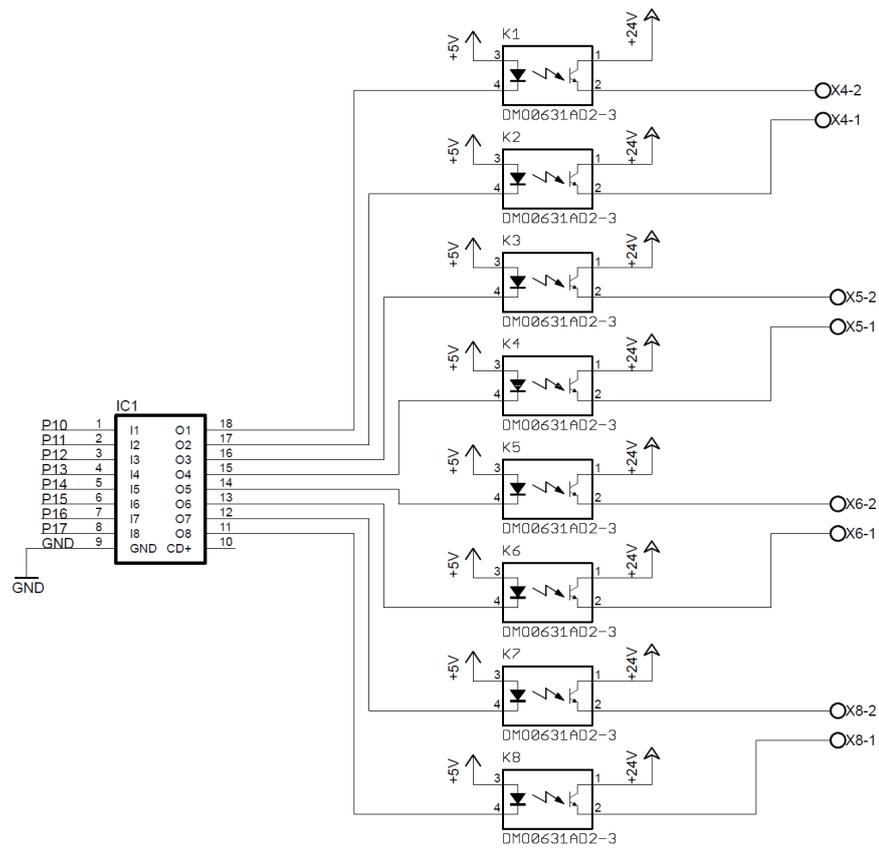


Abbildung 6.14: Ansteuerung von 8 Kanälen

7 Simulationen

7.1 Allgemeines

Um die zuvor dimensionierten und berechneten Schaltungen einer ersten Funktionskontrolle zu unterziehen, werden diese mit Hilfe von OrCAD-PSpice simuliert und ausgewertet. Es eignen sich allerdings nur Teile der ausgelegten Schaltungen für die Simulationen, da z.B. für die Lichtwellenleiterübertragung keine geeigneten Modelle existieren. Mit Hilfe von PSpice können auch nicht ideale Betriebsbedingungen, wie Spannungseinbrüche beim Netz oder Oberschwingungen, simuliert und anschließend das Verhalten der Schaltungen analysiert werden. Zur Simulation werden folgende Schaltungen herangezogen:

- Schaltung zur Synchronisation mit dem Spannungsnetz
- Ansteuerung des mechanischen Schalters
- Ansteuerung der optischen Thyristoren
- Schaltung zur Bestimmung des Status der Meldekontakte
- Dreiphasiges Einschaltverhalten

7.2 Synchronisation

Schematische Schaltung

Die Synchronisation mit dem Spannungsnetz ist ein Bestandteil des Controllers und muss mit einer hohen Genauigkeit arbeiten, da Einschaltwinkel bis auf ein Grad elektrisch genau erreichbar sein sollen. Die Synchronisation umfasst mehrere Teilschaltungen, die in Abbildung 7.1 zu einer Einheit zusammengefasst sind. Dabei ist der erste Teil der Tiefpass, gefolgt vom Komparator und einem Halbwellgleichrichter. Als Eingangsspannung wurde die Ausgangsspannung des Differenzverstärkers nachgebildet, welcher die Netzspannung unverfälscht auf ein Sinussignal mit einer Amplitude von $\hat{U} = 7 \text{ V}$ umwandeln soll. Für die Ausgangsspannung der Synchronisationseinrichtung ist ein TTL-Signal zu erwarten, wobei die positive Flanke gleichzeitig mit dem Spannungsnulldurchgang (während des positiven Anstieges) erfolgen soll.

Simulation bei idealen Betriebsbedingungen

Um die Funktion der Synchronisationseinrichtung zu überprüfen, wurde die Schaltung mit idealen Betriebsbedingungen simuliert. In Abbildung 7.2 ist das Simulationsergebnis zu sehen, wobei der durchgezogene, sinusförmige Verlauf, die Eingangsspannung darstellt und der strichlierte, rechteckförmige Verlauf, die Ausgangsspannung. Man kann deutlich erkennen, dass die positive Flanke der Ausgangsspannung mit dem

7 Simulationen

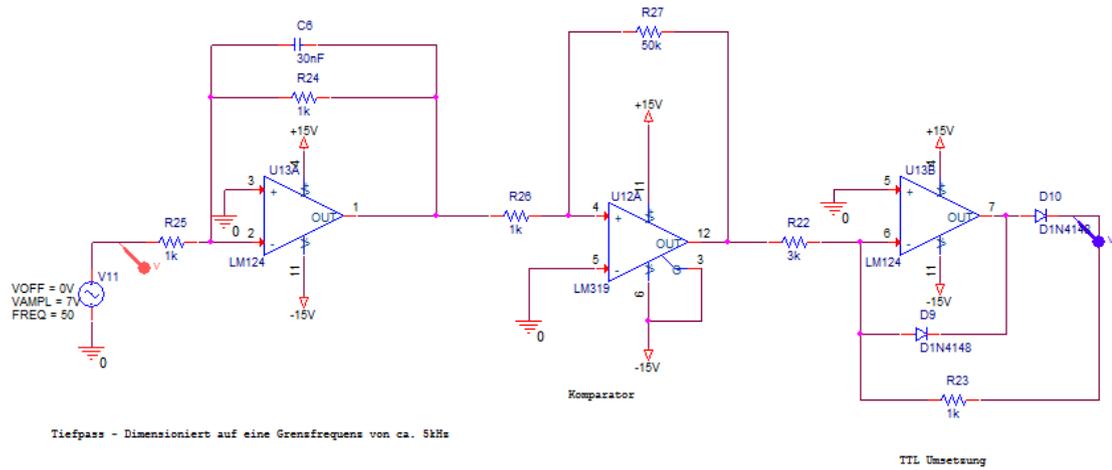


Abbildung 7.1: Schematische Schaltung - Synchronisation

Nulldurchgang der Eingangsspannung zusammenfällt und die Schaltung daher korrekt arbeitet. Auch die Amplitude der rechteckförmigen Ausgangsspannung beträgt exakt 5 V, was für die Verarbeitung im Mikrocontroller wichtig ist.

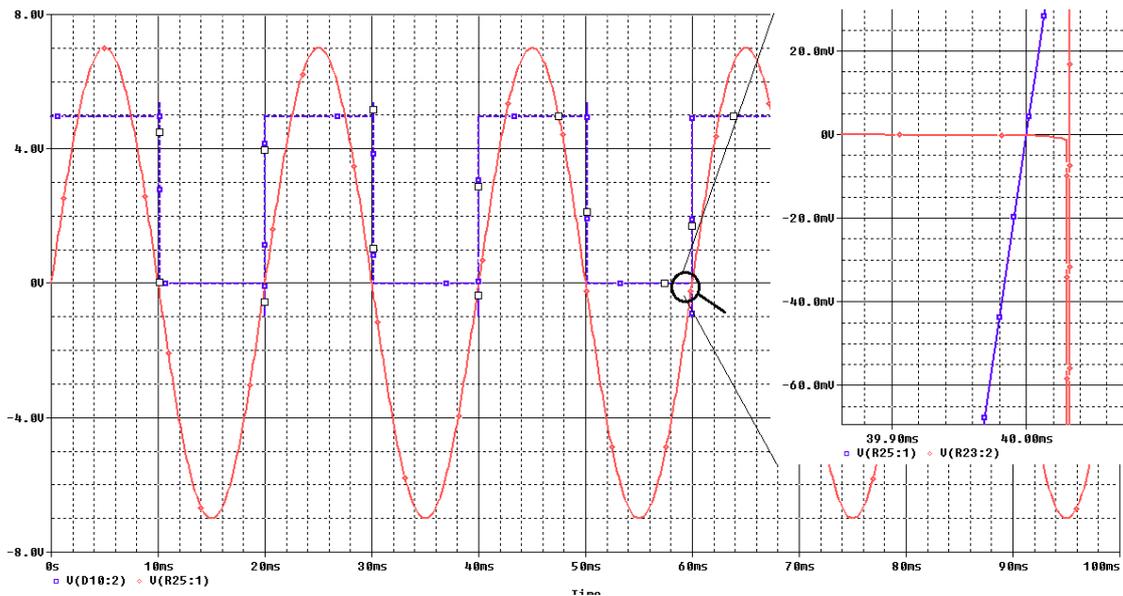


Abbildung 7.2: Simulation - Synchronisation bei idealen Betriebsbedingungen

Bodediagramm Tiefpass

Das Verhalten des Tiefpasses bei der Synchronisation ist bezüglich des Amplituden- und Phasenganges von großem Interesse. Der Amplitudengang bestimmt, ab welcher Frequenz Oberschwingungen bzw. Rauschen unterdrückt werden können. Der Phasengang muss deshalb untersucht werden, da es im Bereich der Netzfrequenz ($f_N = 50 \text{ Hz}$) zu keiner Phasenverschiebung zwischen Eingangsspannung und Ausgangsspannung kommen darf, damit eine korrekte Synchronisation gewährleistet wird. Um die Phasen-

verschiebung in diesem Bereich klein zu halten, ist die Grenzfrequenz auf $f_g = 5 \text{ kHz}$ dimensioniert worden. Abbildung 7.3 zeigt den Amplituden- bzw. Phasengang des Tiefpasses. Im Amplitudengang ist die Grenzfrequenz von $f_g = 5 \text{ kHz}$ gut erkenn-

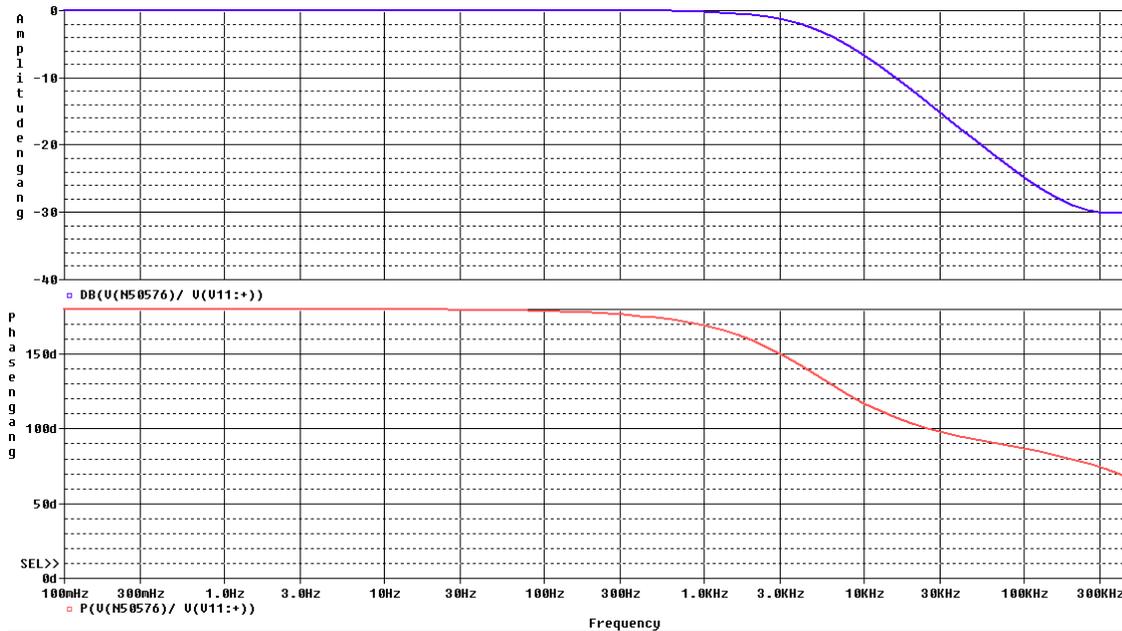


Abbildung 7.3: Simulation - Bodediagramm Tiefpass

bar, da hier die Dämpfung bei ca. 3 dB liegt. Beim Phasengang ist ersichtlich, dass es im Bereich der Netzfrequenz zu keiner Phasenverschiebung kommt. Die angezeigte Phasenverschiebung von 180° rührt vom verwendeten negativen Eingang des Operationsverstärkers her und wird durch den Einsatz des Halbwellengleichrichters wieder kompensiert.

Simulation mit Spannungseinbrüchen

Da die Netzspannung im Bereich von $\pm 10\%$ schwanken kann, kann auch die Ausgangsspannung des Differenzverstärkers in diesem Bereich variieren. Die Synchronisation muss auch in diesen Betriebsfällen zuverlässig funktionieren und daher wurden derartige Spannungsschwankungen im Eingangssignal simuliert. Abbildung 7.4 zeigt das Simulationsergebnis mit einer schwankenden Eingangsspannung im Bereich von $6,3 \text{ V} - 7,7 \text{ V}$. Man kann hier deutlich erkennen, dass das Ausgangssignal trotz der variablen Amplitude der Eingangsspannung, den Nulldurchgang exakt detektiert. Die Amplitude des rechteckförmigen Verlaufs bleibt ebenfalls konstant, dadurch ist die Funktion auch bei Netzspannungseinbrüchen gegeben.

7 Simulationen

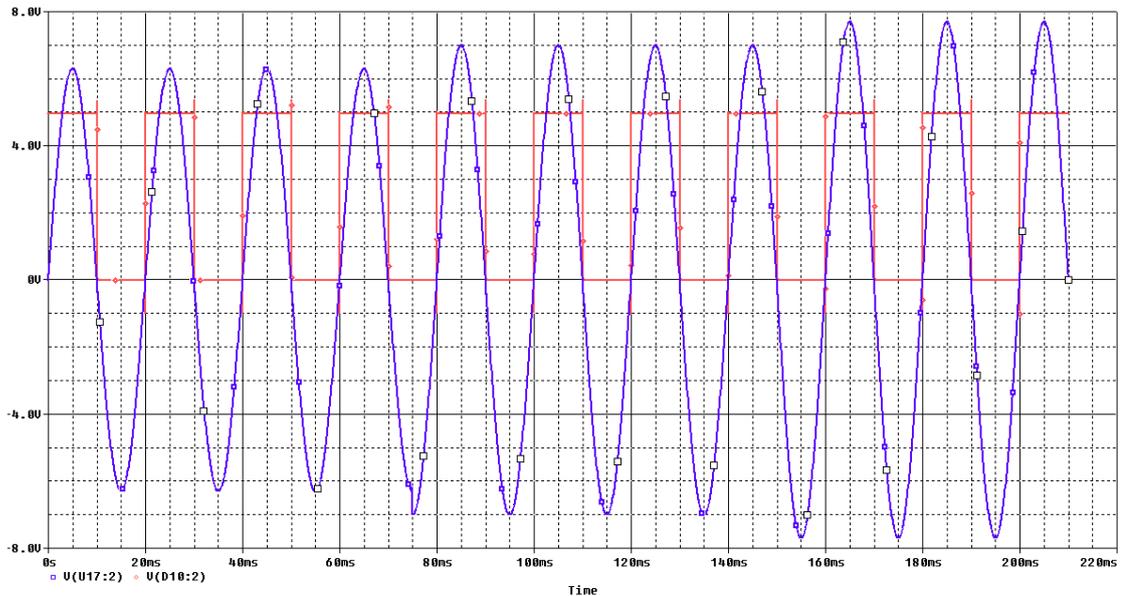


Abbildung 7.4: Simulation - Synchronisation mit Spannungseinbrüchen

Simulation mit Oberschwingungen

Ein weiteres Problem bei der Synchronisation sind Störspannungen im Netz. Diese Störspannungen können z.B. von Schaltvorgängen (hohe Frequenzen) oder von sogenannten Rundsteuersignalen (niedrige Frequenzen) herrühren. Abbildung 7.5 zeigt die Simulation mit dem Eingangssignal und einer 150 Hz Oberschwingung (die 10 % der Grundschwingungsamplitude besitzt), welche eine Phasenverschiebung von 5° besitzt. Man kann deutlich erkennen, dass die Schaltung wieder auf die Grundschwingung

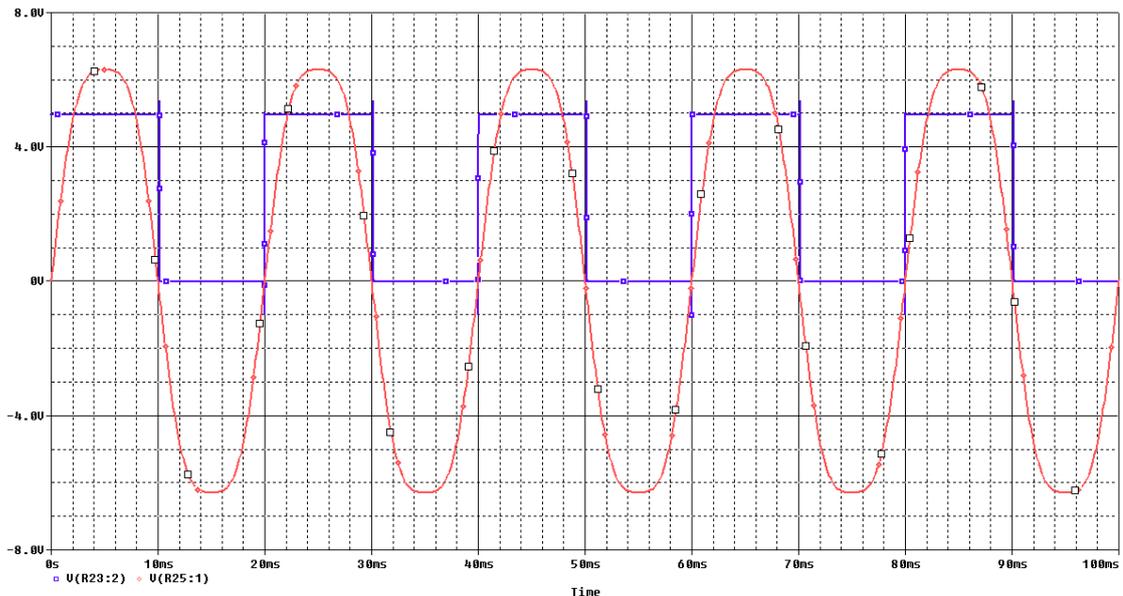


Abbildung 7.5: Simulation - Synchronisation mit einer 150 Hz Oberschwingung

synchronisiert.

7.3 Ansteuerung mechanischer Schalter

Schematische Schaltung

Für den mechanischen Schalter (Leistungsschalter Masterpact NW-20 von Schneider-Electric) ist eine Ansteuerung mittels Solid-State Relais vorgesehen. Das Solid-State Relais steuert die Einschaltspule des mechanischen Schalters, welche die gemessenen Parameter $R_S = 270 \Omega$ und $L_S = 100 \text{ mH}$ besitzt. Abbildung 7.6 zeigt den schematischen Aufbau der Simulation. Das Solid-State Relais wurde durch einen spannungsgesteuerten Schalter ersetzt und für den Einschaltimpuls wurde eine Einschaltzeit von 100 ms gewählt.

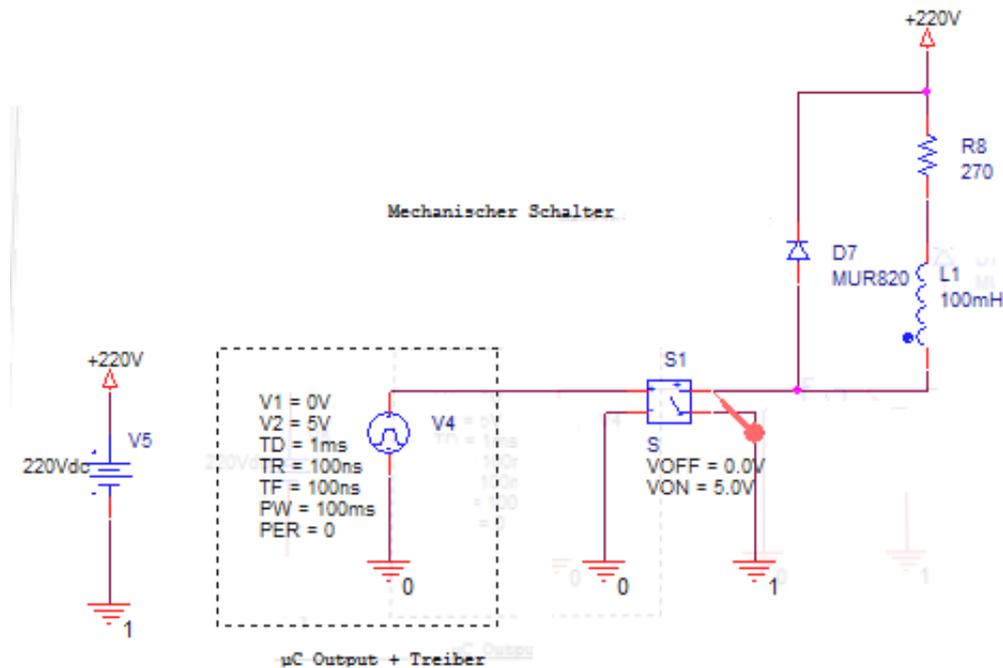


Abbildung 7.6: Schematische Schaltung - Ansteuerung mechanischer Schalter

Simulation bei idealen Betriebsbedingungen

Als Ergebnis der Simulation erhält man den den Stromverlauf an der Einschaltspule. In Abbildung 7.7 ist der Stromverlauf dargestellt. Es ist ersichtlich, dass die Einschaltspule den gemäß $\frac{di}{dt} = \frac{220 \text{ V}}{100 \text{ mH}} = 2,2 \frac{\text{A}}{\text{ms}}$ begrenzt, sich dies aber nicht wesentlich auf die Funktion des Einschaltvorgangs auswirkt.

7 Simulationen

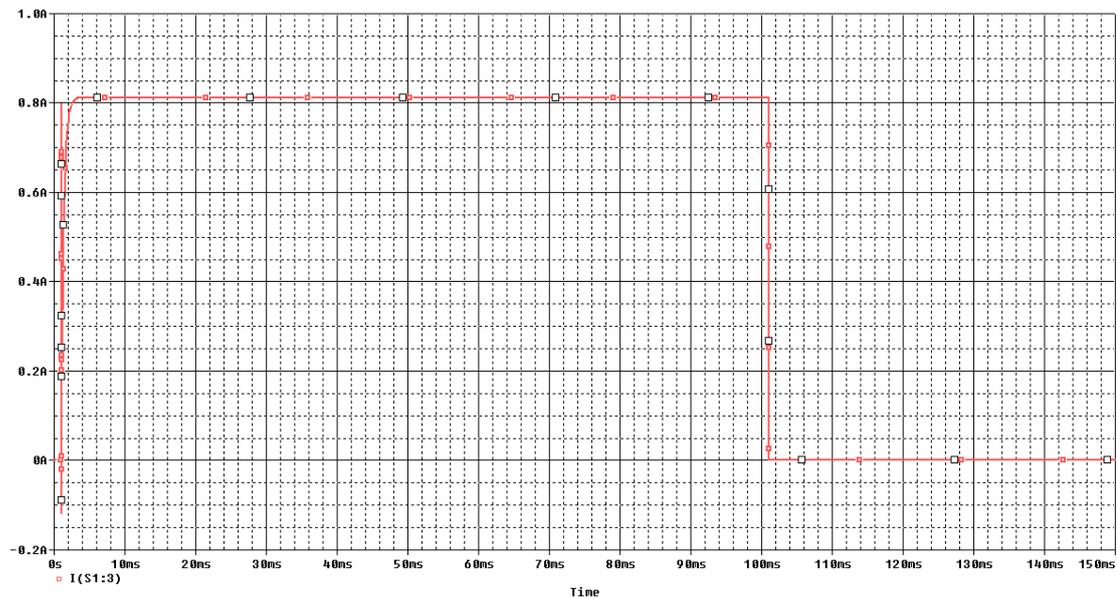


Abbildung 7.7: Simulation - Ansteuerung mechanischer Schalter

7.4 Ansteuerung der optisch-triggerbaren Thyristoren

Schematische Schaltung

Für die optisch-triggerbaren Thyristoren müssen Laserdioden kurzzeitig im Bereich von $5 \mu\text{s} - 10 \mu\text{s}$ angesteuert werden um die Thyristoren zu zünden und gleichzeitig die Laserdioden nicht zu überlasten. Deshalb wird der Strom durch die Laserdioden über die Entladung eines Kondensators erzeugt, wodurch die Stromflussdauer natürlich begrenzt ist. Abbildung 7.8 zeigt den schematischen Aufbau der Schaltung.

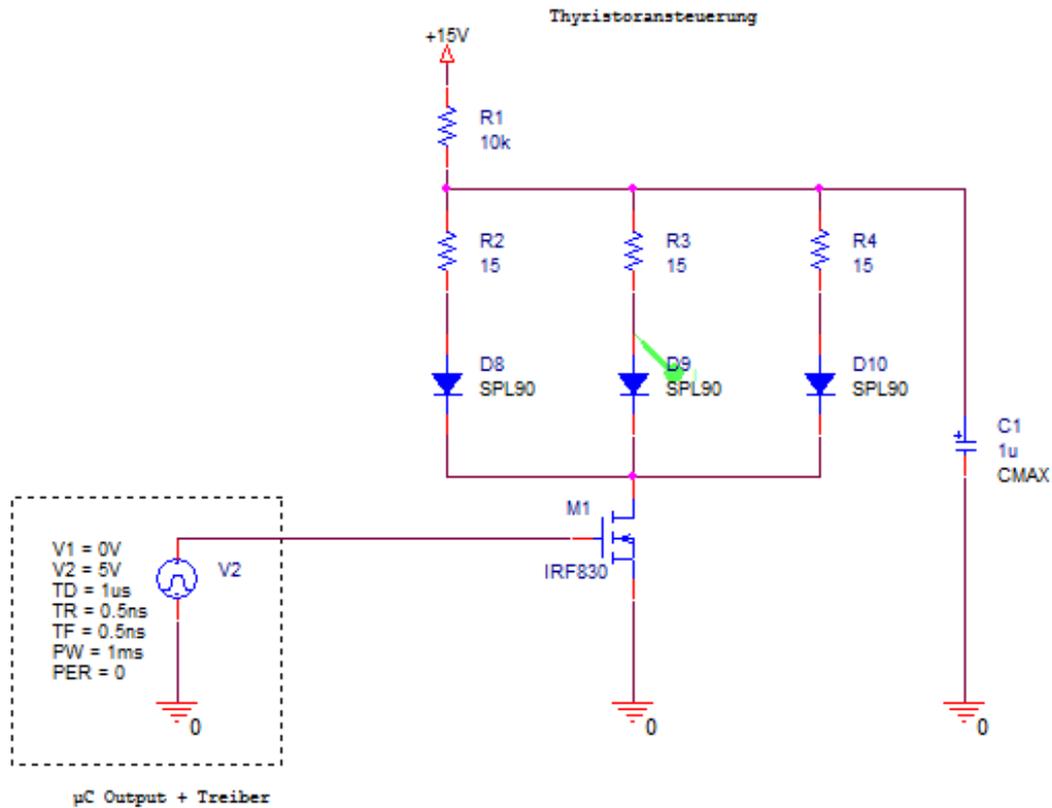


Abbildung 7.8: Schematische Schaltung - Ansteuerung optische Thyristoren

Simulation bei idealen Betriebsbedingungen

Die Simulationsergebnisse der Schaltung sind in Abbildung 7.9 dargestellt, wobei der obere Verlauf die Spannung am Kondensator C_1 darstellt und der untere Verlauf den Strom in einem Zweig für die Ansteuerung der Laserdioden. Die Spannung wie auch der Strom nehmen hier exponentiell ab. Der Strom beträgt nach $5 \mu\text{s}$ noch ca. 300 mA (entspricht nach [8] ca. 300 mW Lichtleistung), d.h. noch genügend Lichtleistung, ausreichend für eine sichere Zündung der Thyristoren. Als Modell für die SPL90 wurde eine Bibliothek des Herstellers (Osram) verwendet.

7 Simulationen

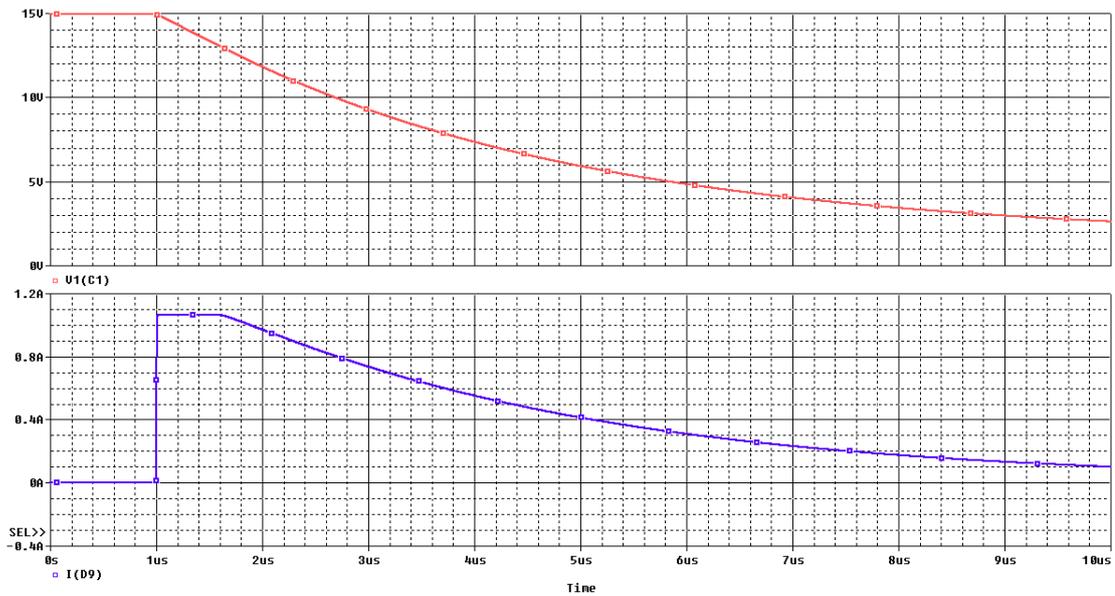


Abbildung 7.9: Simulation - Ansteuerung optische Thyristoren

7.5 Status Meldekontakte

Schematische Schaltung

Bei einer Stellungsänderung der Meldekontakte soll ein Interrupt ausgelöst werden können, um auf die aktuelle Verzögerungszeit des mechanischen Schalters schließen zu können. Dabei soll ein Optokoppler der Type CNY17 zum Einsatz kommen und dessen grundlegende Funktion simuliert werden. Abbildung 7.10 zeigt die schematische Schaltung zur Abfrage des Status der Meldekontakte.

Simulation bei idealen Betriebsbedingungen

Für eine korrekte Messung ist eine hohe Flankensteilheit, sowie eine unverzögerte Übertragung der Spannungsänderung notwendig. Abbildung 7.11 zeigt im oberen Verlauf die Spannung am Meldekontakt (inkl. Vorwärtsspannung der Diode des Optokopplers) und im unteren Verlauf die Spannung auf der Seite des Mikrocontrollers. Es ist zu sehen, dass die Steilheit der Flanken durch den Optokoppler nicht beeinträchtigt wird und auch die Verzögerungszeit (kleiner $40 \mu\text{s}$) des Optokopplers gegen die zu messende Zeit (im Bereich von 50ms) gering ist.

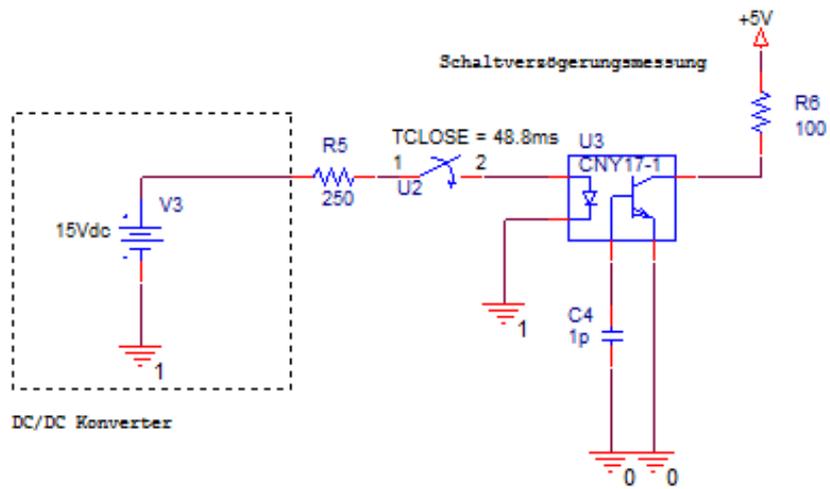


Abbildung 7.10: Schematische Schaltung - Status Meldekontakte

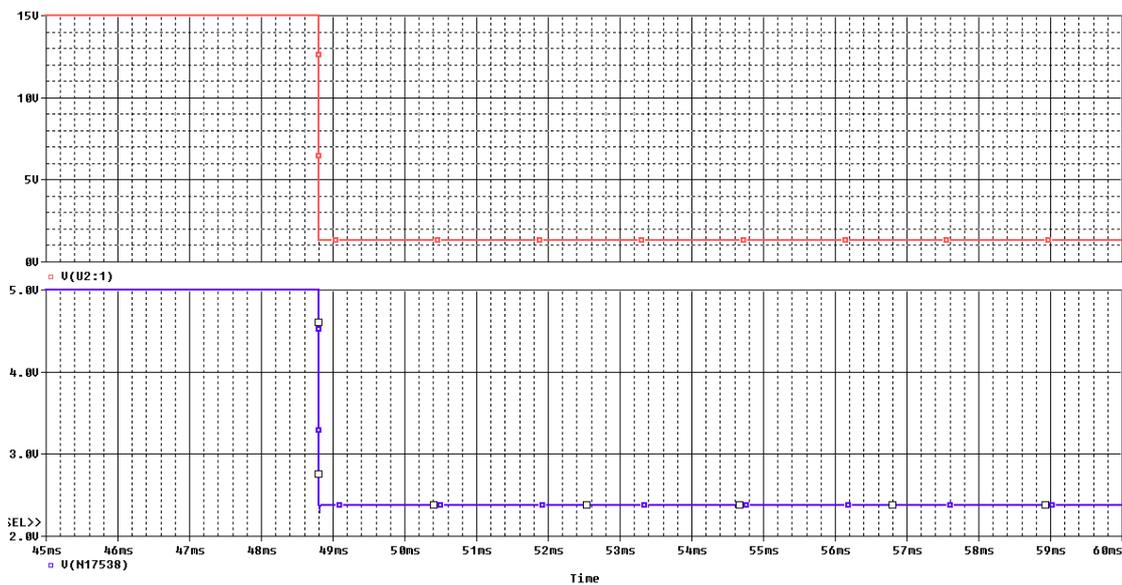


Abbildung 7.11: Simulation Status Meldekontakte

7.6 Dreiphasiges Einschalten

Schematische Schaltung

Für die Validierung der in Kapitel 5 beschriebenen dreiphasigen Einschaltvorgänge eignet sich PSpice hervorragend. Das dreiphasige Modell ist in Abbildung 7.12 schematisch dargestellt. Dafür wird wieder von einem symmetrisch aufgebauten dreiphasigen Netz ausgegangen, wobei die Spannungsquelle V2 um 120° und V3 um 240° phasenverschoben ist, unter der Voraussetzung, dass es sich hier um Kurzschlussversuche handelt, d. h. die Ströme nur durch eine Vorimpedanz begrenzt sind und die Phasen durch das zu untersuchende Betriebsmittel kurzgeschlossen sind. Es gilt nun, die berechneten Modelle mittels Simulation zu validieren.

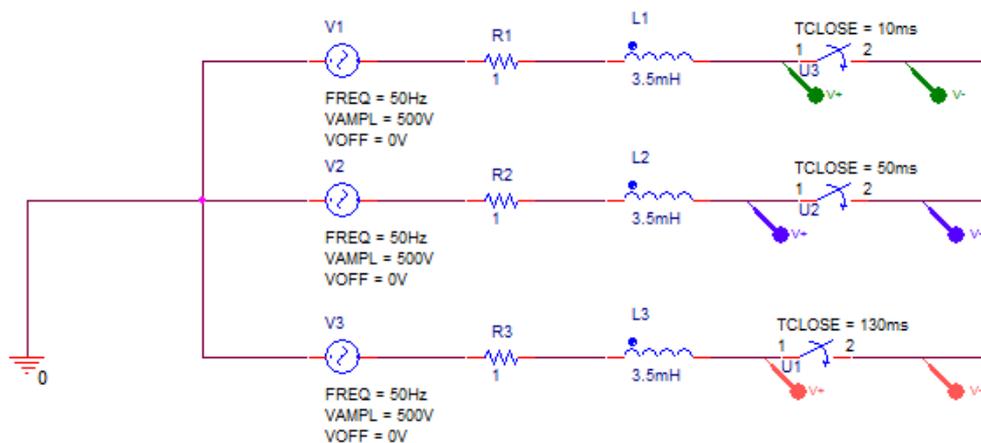


Abbildung 7.12: Schematische Schaltung - Dreiphasiges System

Ohne geerdeten Sternpunkt

Der Schaltvorgang sollte derart ablaufen, dass zuerst der Schalter in Phase L1 geschlossen wird und anschließend der Schalter in Phase L2 synchron geschlossen wird. Dabei liegt am Schalter in Phase L2 die negative verkettete Spannung zwischen L1 und L2 an. Nachdem der Schalter in L2 ebenfalls geschlossen ist, muss noch der Schalter in L3 synchron geschlossen werden. Dabei liegt vor dem Schaltvorgang an dem Schalter in L3 die Spannung $u_{S3}(t) = \frac{3u_3(t)}{2}$ an. Abbildung 7.13 zeigt, wie sich die einzelnen Schalterspannungen nach dem Einschalten der einzelnen Phasen ändern. Zu beachten ist hier, dass die Abstände der einzelnen Schaltzeitpunkte nur zu Simulationszwecken so groß gewählt wurden, um die Berechnungen validieren zu können. Nach dem Schließen des Schalters in L1 liegt an den beiden anderen Schaltern die verkettete Spannung zwischen L1-L2 bzw. L1-L3 mit $\hat{u}_{S2} = \hat{u}_{S3} = \hat{u}_{L12} = \hat{u}_{L13} = \sqrt{3}500 \text{ V} = 866 \text{ V}$ an. Schließt man nun den Schalter in L2, so liegt an dem Schalter in L3 die Spannung $u_{S3}(t) = \frac{3u_3(t)}{2}$ mit $\hat{u}_{S3} = \frac{3 \cdot 500}{2} = 750 \text{ V}$

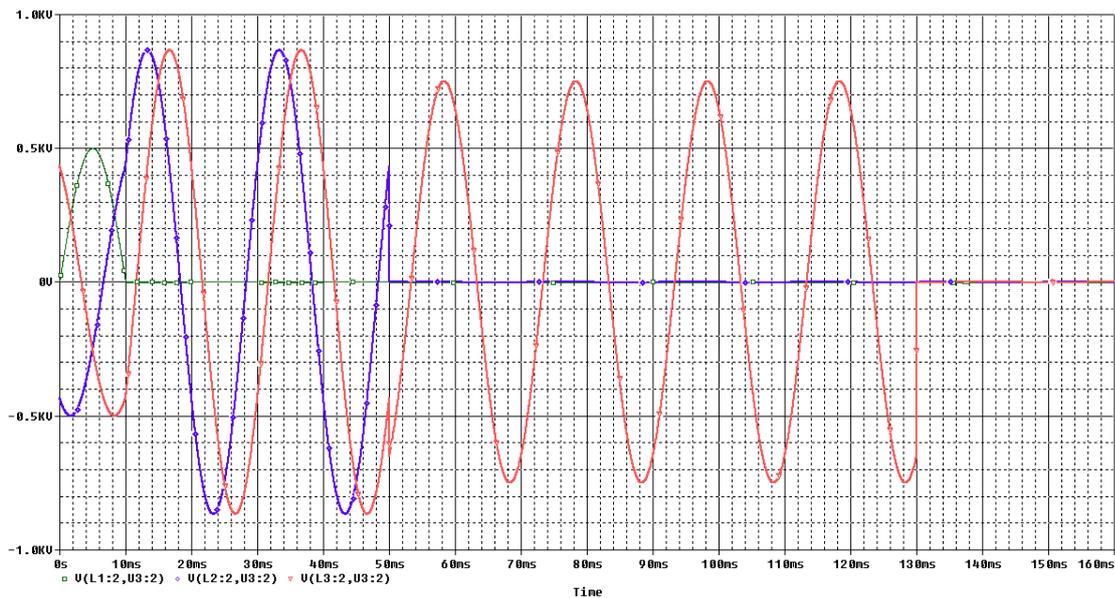


Abbildung 7.13: Simulation dreiphasiges Einschalten ohne geerdeten Sternpunkt

Mit geerdetem Sternpunkt

In Abbildung 7.14 ist dieselbe Simulation wie oben durchgeführt, nur mit geerdetem Sternpunkt. Dabei werden die Schalterspannungen durch die Schalthandlungen in den anderen Phasen nicht beeinflusst.

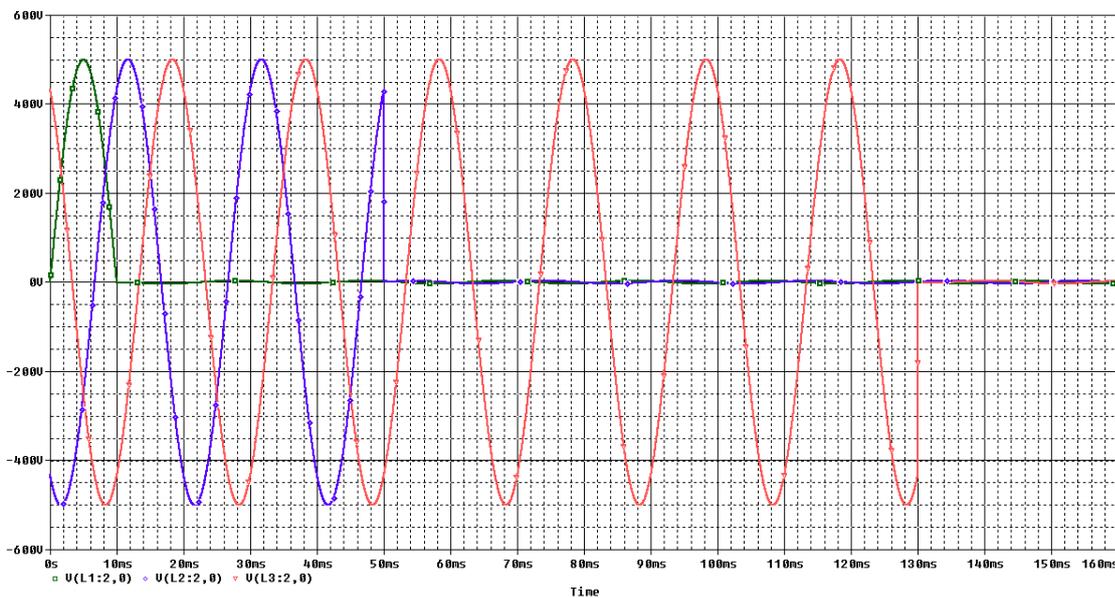


Abbildung 7.14: Simulation dreiphasiges Einschalten mit geerdetem Sternpunkt

8 Entwicklung der Software

8.1 Konzept

Anforderungen

Die entwickelte Software soll es ermöglichen, sowohl den Hybridschalter, als auch die freien Kanäle über eine grafische Benutzeroberfläche steuern zu können. Für die Programmierung der Benutzeroberfläche kommt LabView zum Einsatz und für die Programmierung der Mikrocontroller wird ein maschinennahes C-Derivat von ATMEL verwendet.

Bei der eingesetzten Steuerungssoftware gibt es nun zwei Ebenen. Auf der unteren Ebene steht die Software des Mikrocontrollers. Da ein Mikrocontroller für die Synchronsteuerung eingesetzt wird und ein Mikrocontroller für den Hybridschalter, sind hier zwei Programme zu erstellen. Abbildung 8.1 zeigt schematisch das Konzept der Software. Die grafische Benutzeroberfläche hat die Aufgabe, die Eingabe der Parameter¹ zu

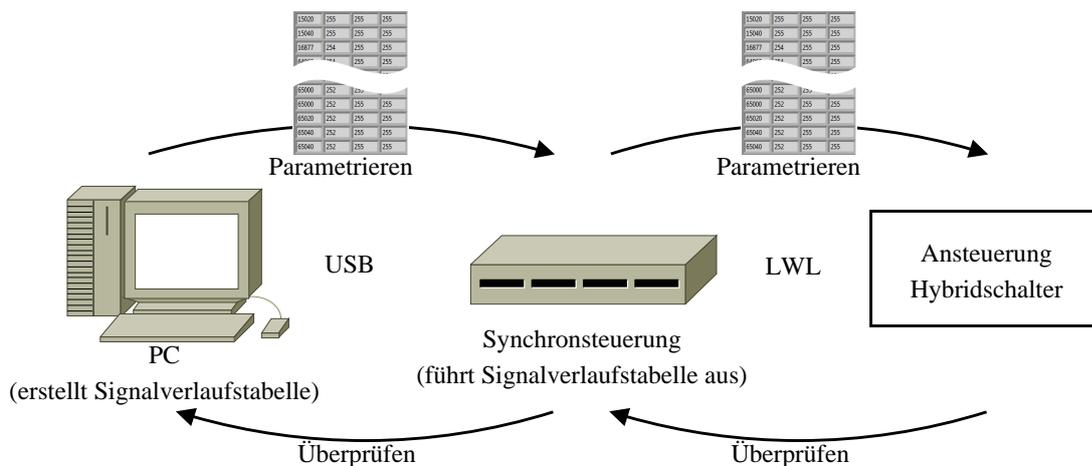


Abbildung 8.1: Software Konzept

ermöglichen. Aus diesen Parametern werden dann die Einschaltzeitpunkte für den mechanischen Schalter, die optisch-triggerbaren Thyristoren sowie für die Zusatzbeschaltung errechnet und Signalverlaufstabellen erstellt. Für die Steuerung der freien Kanäle wird nach dem selben Prinzip verfahren und ebenfalls Signalverlaufstabellen erstellt. Diese Tabellen enthalten die Einschalt- bzw. Ausschaltzeitpunkte und für jeden Zeitpunkt ein entsprechendes Byte, das an einem Ausgangsport die einzelnen Ausgänge setzt. Die Übertragung der gesamten Signalverlaufstabelle vom PC auf die Synchronsteuerung erfolgt via USB. Die Synchronsteuerung übergibt den für die Ansteuerung des Hybridschalter relevante Signalverlaufstabelle via Lichtwellenleiter.

¹Parameter sind z.B.: Einschaltzeitpunkt des Hybridschalters bezogen auf einen Spannungsnulldurchgang, Effektivwert der Spannung

Zeitliche Abfolge

Für die prinzipielle zeitliche Abfolge ist es wichtig, alle Eigenzeiten des anzusteuernenden Hybridschalters zu kennen, insbesondere ist die Schaltverzögerung der mechanischen Schalter ausschlaggebend. Es muss hier mit einer Schaltverzögerung t_{Verzug} um ca. 50 ms gerechnet werden.

Der Einschaltzeitpunkt für den Hybridschalter bzw. für die freien Kanäle wird immer bezogen auf einen Spannungsnulldurchgang angegeben auf den sich die Ansteuerung synchronisiert. Hat man nun einen Einschaltzeitpunkt vorgegeben, so synchronisiert sich die Schaltung mit dem Spannungsnetz und führt die erforderlichen Schaltvorgänge aus. Bei einer 50 Hz Wechselspannung (ergibt eine Periodendauer von $T_{Netz} = 20\text{ ms}$) muss man bei der oben angeführten Schaltverzögerung des mechanischen Schalters mindestens drei Vollwellen (entspricht 60 ms) vorübergehen lassen, um einen hybriden Einschaltvorgang zu ermöglichen. Damit ergibt sich der Einschaltbefehl für den mechanischen Schalter, für einen Einschaltzeitpunkt Δt nach $t_{ein} = 3 \cdot T_{50\text{Hz}} - t_{Verzug} + \Delta t$. In Abbildung 8.2 ist die zeitliche Abfolge für einen einfachen hybriden Einschaltvorgang dargestellt. Dabei bedeutet t_{ein} den Zeitpunkt für den Einschaltbefehl des mechani-

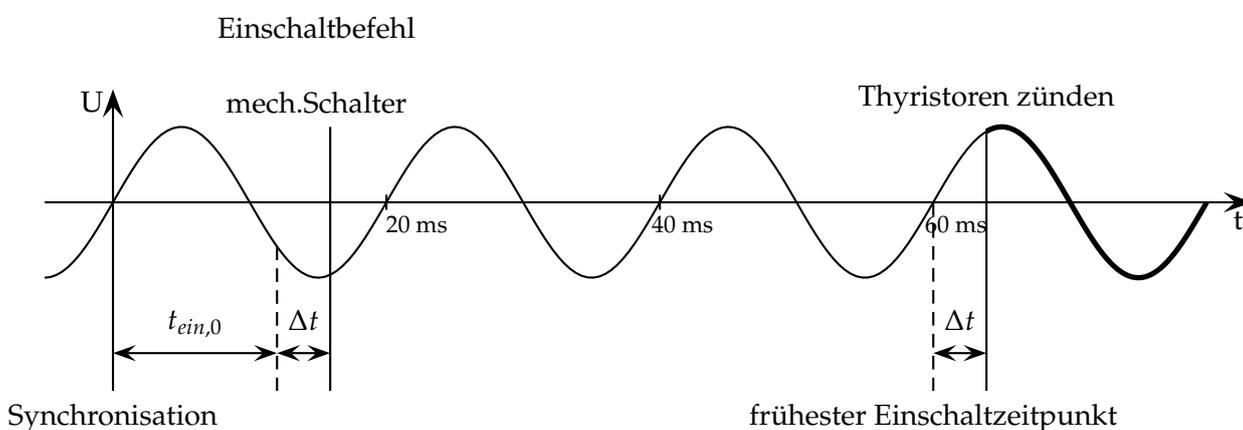


Abbildung 8.2: Zeitliche Abfolge der Steuerung

schens Schalters und die $t_{ein,0}$ den ehest möglichen Einschaltzeitpunkt für den mechanischen Schalter, wenn man im Spannungsnulldurchgang einschalten möchte. Der tatsächliche Einschaltzeitpunkt des Hybridschalters ist durch das Zünden der Thyristoren gegeben. Abhängig von der Steuerung der Verzögerungszeit des mechanischen Schalters schließt dann kurz darauf der mechanische Schalter und der Strom kann von den Thyristoren auf den mechanischen Schalter kommutieren. Danach ist der Einschaltvorgang abgeschlossen.

Kommunikationsprotokoll

Für die Parametrierung der Mikrocontroller ist es notwendig, ein Kommunikationsprotokoll zwischen LabView und Mikrocontroller bzw. zwischen Mikrocontroller und Mikrocontroller zu entwickeln. Wie in Abbildung 8.1 angedeutet, werden die von LabView übertragenen Tabellen² wieder zurück an LabView übergeben, um eine sichere Datenübertragung zu gewährleisten und den Kommunikationsstatus zu überprüfen. Das Konzept baut dabei auf eine hierarchische Kommunikationsstruktur auf. Labview

²tatsächlich werden die Spalten der Tabellen einzeln, also lediglich Vektoren übertragen

übernimmt dabei die Rolle eines „Master“ und die Mikrocontroller sind die dazugehörigen „Slave“. Für die Kommunikation zwischen PC und der Synchronsteuerung via USB wird die Schnittstelle UART0 und für die Kommunikation zwischen der Synchronsteuerung und Ansteuerung des Hybridschalters wird die Schnittstelle UART2 des ATmega2560 verwendet. Diese Schnittstellen sind ebenfalls für die Programmierung des Mikrocontrollers zu verwenden.

Für die Parametrierung müssen nun die Tabellen bzw. die einzelnen Spalten nach einem bestimmten Protokoll übertragen werden. Die nun erläuterten Protokolle sind sowohl für die Kommunikation zwischen LabView und Mikrocontroller als auch für die Kommunikation zwischen Mikrocontroller und Mikrocontroller³ implementiert worden. In Abbildung 8.3 ist der Ablauf für das Senden eines einzelnen Vektors dargestellt. Zu Beginn muss mit dem Senden eines einzigen Steuerzeichens die Funktion auf dem

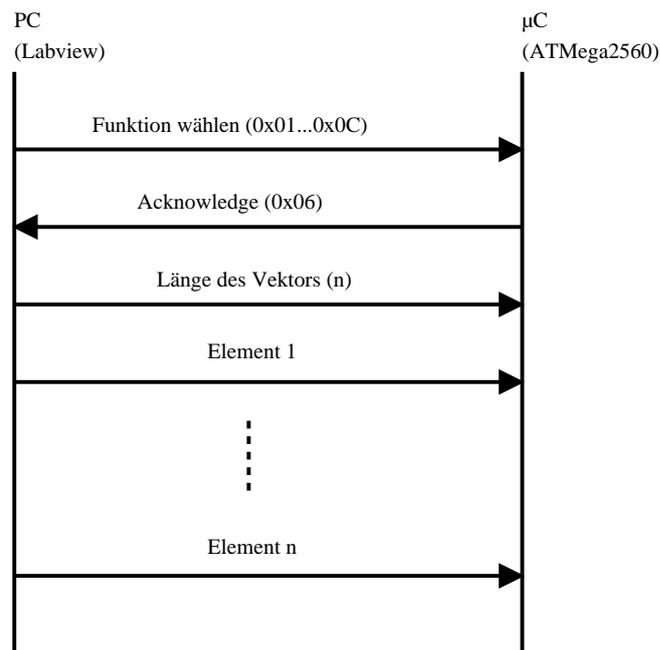


Abbildung 8.3: Kommunikationsprotokoll - Vektor an Steuerung senden

Slave gewählt werden. Welche Funktionen prinzipiell zu Verfügung stehen zeigt später Abbildung 8.5. Wird die, wie in Abbildung 8.3 dargestellte Funktion für das Senden eines Vektors gewählt, so folgt das Zeichen für „Acknowledge“. Damit signalisiert der Slave, dass er sendebereit ist und die Übertragung beginnt mit der Übergabe der Länge des Vektors. Durch die Länge des Vektors ist dem Slave bekannt, wieviele Elemente der Master senden wird und empfängt daher auch nur genau diese Anzahl an Elementen. Für die Überprüfung, ob der Slave alle Daten richtig erhalten hat, bzw. für die Übertragung von Messwerten, ist es auch wichtig, dass der Slave Vektoren senden kann. Der prinzipielle Ablauf ist in Abbildung 8.4 dargestellt. Die Abfrage dieser Vektoren startet wieder der Master mit der Auswahl der Funktion und ein darauf folgendes „Acknowledge“ des Slaves, danach überträgt der Slave alle Elemente nacheinander. Die Anzahl der übertragenen Elemente muss in diesem Fall nicht übermittelt werden, da dem Master die Anzahl der übermittelten Elemente ohnehin bekannt ist.

³In diesem Fall übernimmt der Mikrocontroller der Synchronsteuerung die Funktion des Masters

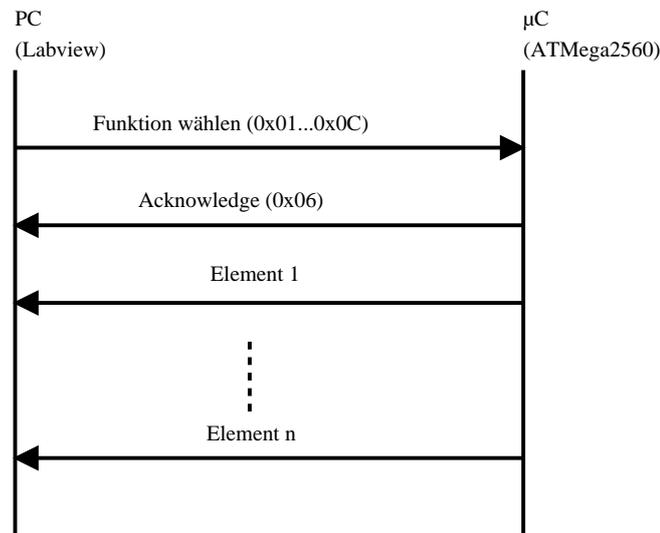


Abbildung 8.4: Kommunikationsprotokoll - Vektor von Steuerung lesen

8.2 Mikrocontroller - Synchronsteuerung

Anforderungen und verwendete Peripherie

Die Programmierung der Synchronsteuerung muss sowohl die Kommunikation herstellen, als auch die Ausführung der Signalverlaufstabellen ausführen können. Dafür benötigt das Programm auch einiges an Peripherie nach Tabelle 8.1, welche der ATMEga 2560 bereitstellt. Für die Übertragung sind die schon zuvor genannten Schnittstellen

Verwendete Peripherie des Mikrocontrollers	
USB - Schnittstelle	UART0
LWL - Schnittstelle	UART2
Synchronisation	Ext. Interrupt 3
Auslösung des Schaltvorganges	Ext. Interrupt 4
16-bit Timer	Timer1

Tabelle 8.1: Verwendete Peripherie des Mikrocontrollers für die Synchronsteuerung

zu konfigurieren. Für die Synchronisation wird der externe Interrupt 3 auf eine positive Flanke konfiguriert um den Spannungsnulldurchgang zu detektieren. Die Auslösung des synchronen Schaltvorganges soll manuell erfolgen, dafür wird der externe Interrupt 4 verwendet, welcher den Schaltvorgang startet (erst nach der Synchronisation starten die Schaltvorgänge wirklich). Nach der Synchronisation beginnt sofort ein Timer, der es ermöglicht die übergebene Signalverlaufstabelle zu den exakten Zeitpunkten auszuführen, zu zählen.

Funktionen der Synchronsteuerung

Das Programm der Synchronsteuerung ist in Abbildung 8.5 dargestellt. Es ist hier das Hauptprogramm (linke Seite), die Initialisierung, sowie auch die Interrupts abgebildet (rechte Seite). Das Hauptprogramm ermöglicht es, eine Auswahl der in Frage kommenden Funktionen (wie z.B. Vektoren übertragen, Schaltvorgänge ausführen) zu treffen. Das Hauptprogramm wird permanent ausgeführt und ist nach der Abarbeitung einer Funktion wieder bereit für die Auswahl einer neuen Funktion. Die Initialisierung konfiguriert die Interrupts, den Timer und die UART-Schnittstellen. Der Synchronisationsinterrupt startet nach erfolgter Synchronisation den Timer und somit die Abarbeitung der Signalverlaufstabelle. Der Timer für die Abarbeitung wird als 32-bit Timer verwendet und muss daher während eines Zählerüberlaufs die Anzahl der Überläufe (Overflows) mitzählen.

Ablauf der Schaltvorgänge

Die wichtigste Funktion im Programm der Synchronsteuerung ist die Abarbeitung der Signalverlaufstabelle nach erfolgter Synchronisation.

Dazu sei noch eine Anmerkung zum implementierten Timer gemacht. Die Geschwindigkeit des Timers ist von besonderer Bedeutung und ist mit der Oszillatorfrequenz und dem sog. Prescaler durch folgenden Zusammenhang gegeben

$$f_{OC} = \frac{f_{CPU}}{Prescaler}. \quad (8.1)$$

Für eine Oszillatorfrequenz von $f_{CPU} = 16 \text{ MHz}$ und einem Prescaler von 8 ergibt sich $f_{OC} = 2 \text{ MHz}$ was einem Zeitabstand von $t_{OC} = 0,5 \mu\text{s}$ entspricht. Das bedeutet, dass die übergebenen Zeit-Vektoren ⁴ noch mit einem Faktor 2 multipliziert werden müssen bevor sie verglichen werden können. Dadurch kann man sich auch die Zeitdauer berechnen, die ein 16-bit Register abarbeiten kann, bis es überläuft. Die maximale Zähldauer ergibt sich zu

$$t_{max} = 2^{16} \cdot 0,5 \mu\text{s} = 32,768 \text{ ms}. \quad (8.2)$$

Da diese Zeitdauer aber nicht ausreichend ist um den gesamten Schaltvorgang abzudecken, aber die Auflösung des Timers auch nicht verändert werden sollte, wird ein 32-bit Timer implementiert, welcher durch einen 16-bit Timer mit Überlaufzähler realisiert ist. Damit ist eine maximale Zähldauer von

$$t_{max,32} = 2^{32} \cdot 0,5 \mu\text{s} = 2147,48 \text{ s} \quad (8.3)$$

möglich.

Abbildung 8.6 zeigt nun den Ablauf der Schaltvorgänge. Die Abläufe sind bei der Synchronsteuerung und der Ansteuerung der Hybridschalter prinzipiell gleich und wird deshalb auch nur hier gezeigt. Die Unterschiede beim Ablauf bestehen nur darin, dass unterschiedliche Ports (Ausgänge) gesetzt werden und dass sich die Synchronsteuerung mit dem Spannungsnetz synchronisiert, die dann die Ansteuerung des Hybridschalters startet.

Die Überprüfung, ob der mechanische Schalter ausgelöst wurde, erfolgt nur im Rahmen der Ansteuerung für den Hybridschalter und nicht bei der Synchronsteuerung.

⁴Die Zeiten werden in μs übergeben

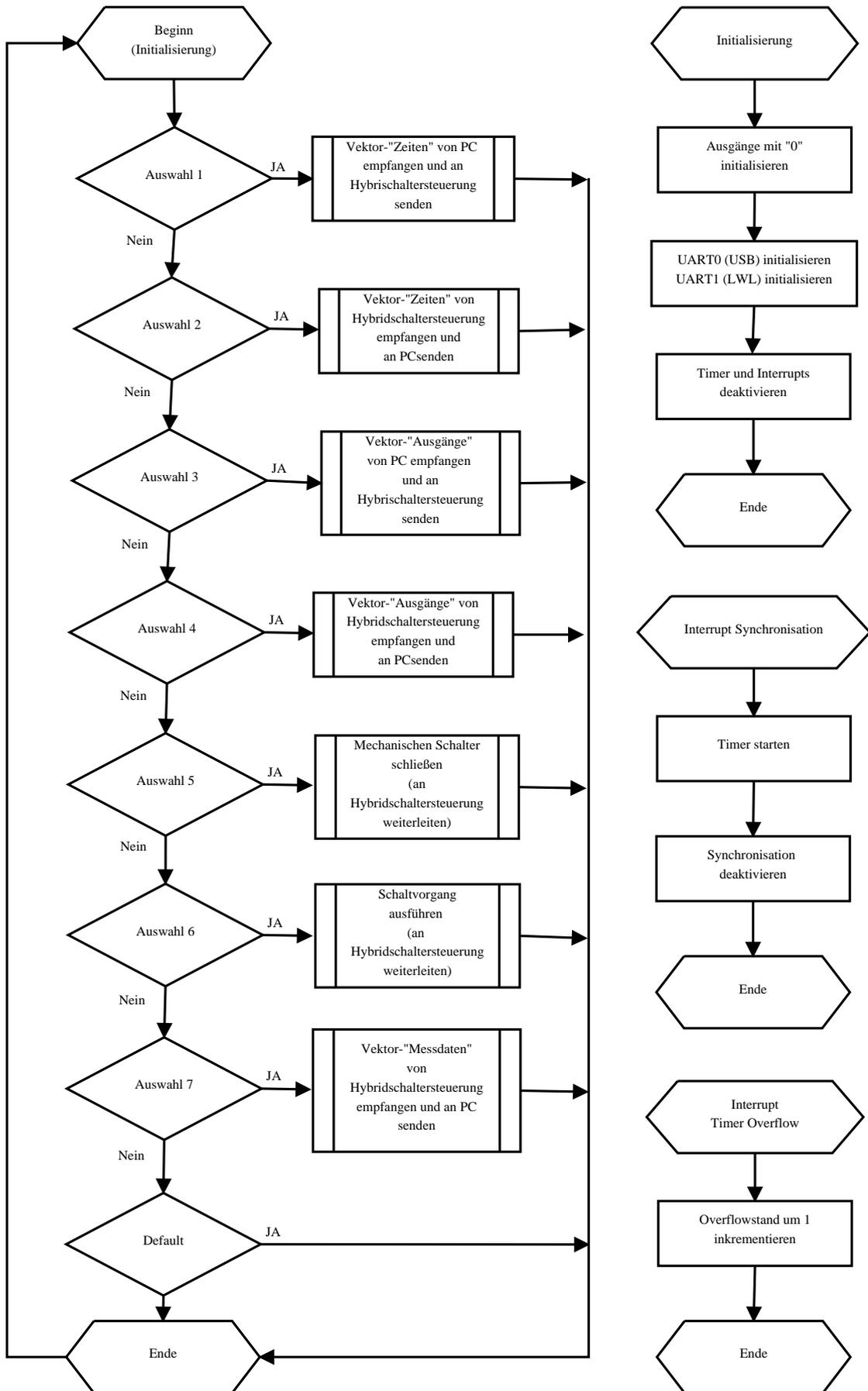


Abbildung 8.5: Ablaufdiagramm - Microcontroller Synchronsteuerung

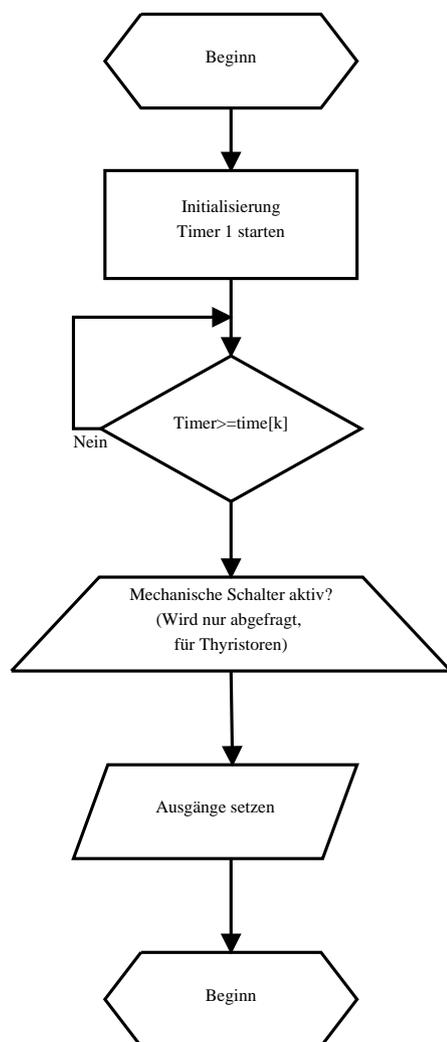


Abbildung 8.6: Ablaufdiagramm - Ausführung der Schaltvorgänge

8.3 Mikrocontroller - Hybridschalter

Anforderungen und verwendete Peripherie

Das Programm für den Mikrocontroller der Hybridschalter muss sowohl die Kommunikation mit der Synchronsteuerung herstellen können als auch die Signalverlaufstabellen abarbeiten können. Dafür wird die Peripherie nach Tabelle 8.2 des ATmega2560 benützt. Für die Kommunikation ist hier nur die Übertragung der Lichtwellenleiter zu

Verwendete Peripherie des Mikrocontrollers	
LWL - Schnittstelle	UART2
Status mechanischer Schalter 1	Ext. Interrupt 0
Status mechanischer Schalter 2	Ext. Interrupt 1
Status mechanischer Schalter 3	Ext. Interrupt 2
16-bit Timer	Timer1

Tabelle 8.2: Verwendete Peripherie des Mikrocontrollers für die Hybridsteuerung

konfigurieren. Da die Ansteuerung für die Hybridschalter auch die Verzögerungszeiten der mechanischen Schalter mitmessen soll, muss hier eine Abarbeitung der Eingangssignale mit Interrupts erfolgen. Für jeden einzelnen mechanischen Schalter ist daher ein entsprechender externer Interrupt zu konfigurieren. Um den aktuellen Zähler- und Überlaufzählerstand abzuspeichern werden Interrupt 0 - Interrupt 2 verwendet. Die Ausführung der Signalverlaufstabelle erfolgt hier wieder mit Hilfe eines 32-bit Timers.

Funktionen des Hybridschalters

In Abbildung 8.7 ist der Ablauf des Programmes für den Mikrocontroller der Hybridschalter abgebildet. Die Auswahl der Funktionen ist aus Kompatibilitäts- und Wartungsgründen ähnlich aufgebaut wie die der Synchronsteuerung. Das Hauptprogramm (rechte Seite) bietet wieder eine Auswahl der Funktionen. Die Initialisierung setzt die Ausgänge zurück, konfiguriert die Interrupts sowie auch die Kommunikation. Wird eine Interruptroutine durch das Schließen eines mechanischen Schalters ausgelöst, so speichert diese sowohl den aktuellen Zählerstand als auch den Überlaufzählerstand ab. Diese Zählerstände werden dann nach der erfolgten Abarbeitung an LabView übergeben. Es ist dann in LabView möglich, die aktuellen mechanischen Verzugszeiten auszurechnen.

8.4 Entwicklung in LabView

Anforderungen

Die Programmierung in LabView erfolgt grundsätzlich in zwei Ebenen. Zur Gestaltung der Benutzeroberfläche steht das sogenannte Frontpanel zur Verfügung, welches Elemente für die grafische Aufbereitung bereitstellt. Für die eigentliche Programmierung

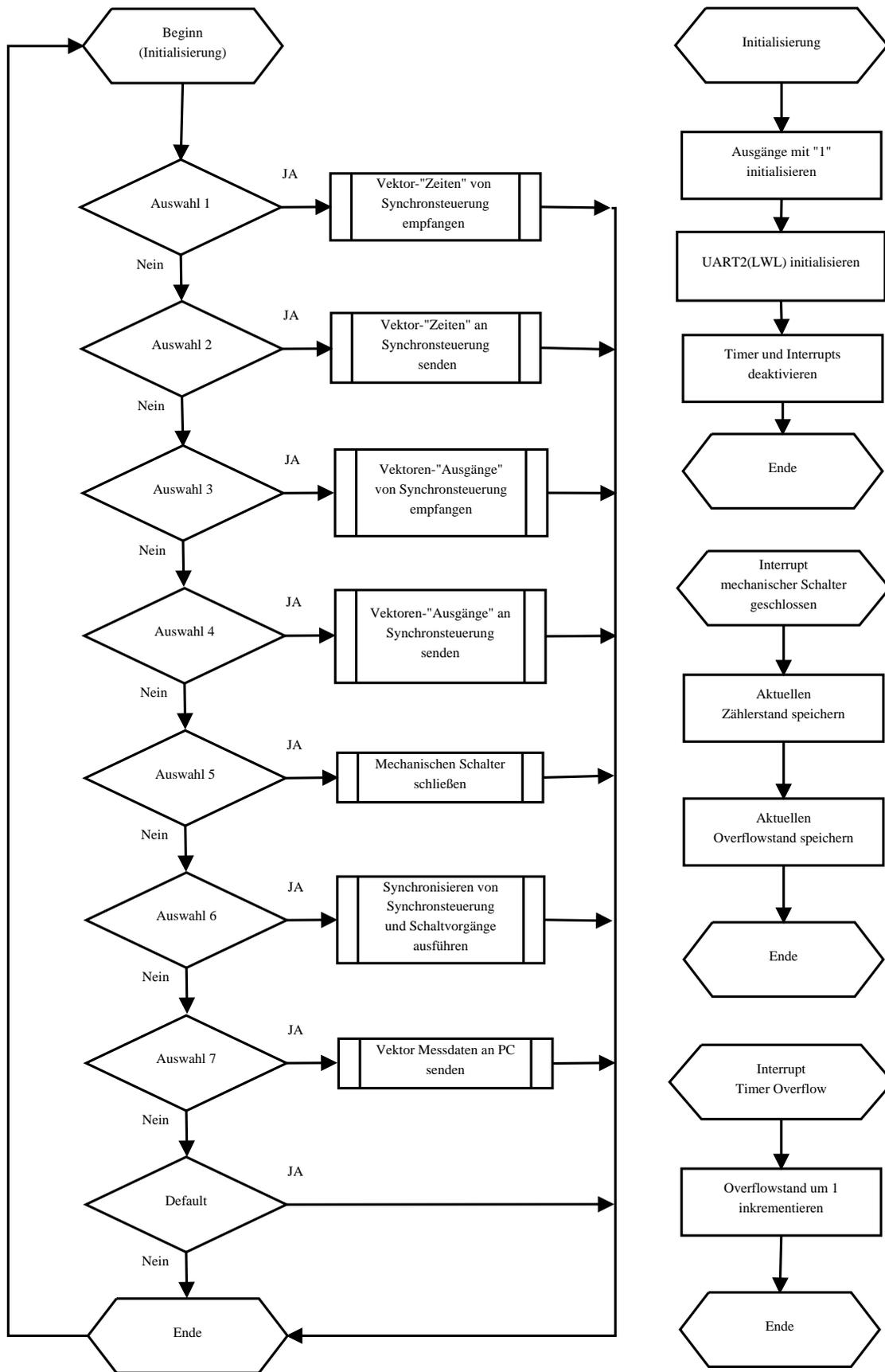


Abbildung 8.7: Ablaufdiagramm - Microcontroller Hybridschalter

werden sogenannte Blockdiagramme verwendet, welche für den Benutzer nicht sichtbar sind bzw. im Hintergrund ausgeführt werden.

In LabView erfolgt nun die Eingabe der ein zu stellenden Parameter und in weiterer Folge die Erstellung der Signalverlaufstabellen, welche dann an die Mikrocontroller übertragen werden. Die Auswertung der Verzugszeiten des mechanischen Schalters, sowie weitere Einstellungen erfolgen ebenfalls in LabView.

Frontpanel

Wie oben bereits erwähnt, stellt das Frontpanel die Benutzeroberfläche für die Hauptanwendung dar. In Abbildung 8.8 ist das Frontpanel des implementierten Programmes zu sehen. Für die Ansteuerung des Hybridschalters ist der Einschaltzeitpunkt in μs , der Effektivwert der Spannung in *Volt*, sowie die Art der Prüfung als Parameter einzugeben⁵. Die Arten der Prüfung sind wie in Kapitel 5 gegeben durch:

- L1 - N
- L2 - N
- L3 - N
- L1 - L2
- L2 - L3
- L3 - L1
- L1 - L2 - L3
- L1 - L2 - L3 - N

Es wird durch diese Eingabe eine Signalverlaufstabelle für die Ansteuerung des Hybridschalters generiert und der Verlauf der Spannungen wird ebenfalls in einem Diagramm dargestellt.

Für Prüfungen, die es erfordern einen einzelnen mechanischen Schalter zu schließen, kann dies ebenfalls über das Frontpanel gesteuert werden. Eine zusätzliche Anzeige ermöglicht es, den Status der einzelnen Schalter zu überprüfen. Die Synchronsteuerung ist ebenfalls über das Frontpanel konfigurierbar. Dabei ist der Einschaltzeitpunkt sowie der Ausschaltzeitpunkt eines Kanals anzugeben und zusätzlich besteht die Möglichkeit einer Aktivierung/Deaktivierung der Kanäle.

Einstellungen - Ansteuerung des Hybridschalters

Die Ansteuerung des Hybridschalters erfordert mehr als nur die Eingabe der Parameter im Frontpanel. Es ist im Dialog - Einstellungen Hybridschalter (Menü Hybridschalter → Einstellungen Hybridschalter) in Abbildung 8.9 möglich, sowohl die Dauer des Einschaltimpulses für die Zusatzbeschaltung, die Spannungsschwellen, als auch die Einschaltverzögerung zwischen L1 - L2 - L3 einzustellen. Die Dauer des Einschaltimpulses für die Zusatzbeschaltung ist durch einen Einschaltzeitpunkt und einen Ausschaltzeitpunkt gegeben und beträgt max. 100 μs . Diese Zeitpunkte sind allerdings relativ zum Einschaltzeitpunkt der optisch-triggerbaren Thyristoren einzugeben. Die

⁵Auf die Eingabe des $\cos(\varphi)$ wird zunächst verzichtet und ein $\cos(\varphi) = 1$, also rein ohmsche Impedanzen unterstellt. Dadurch kann es passieren, dass in Abhängigkeit vom Einschaltzeitpunkt, ein lückender Betrieb entsteht.

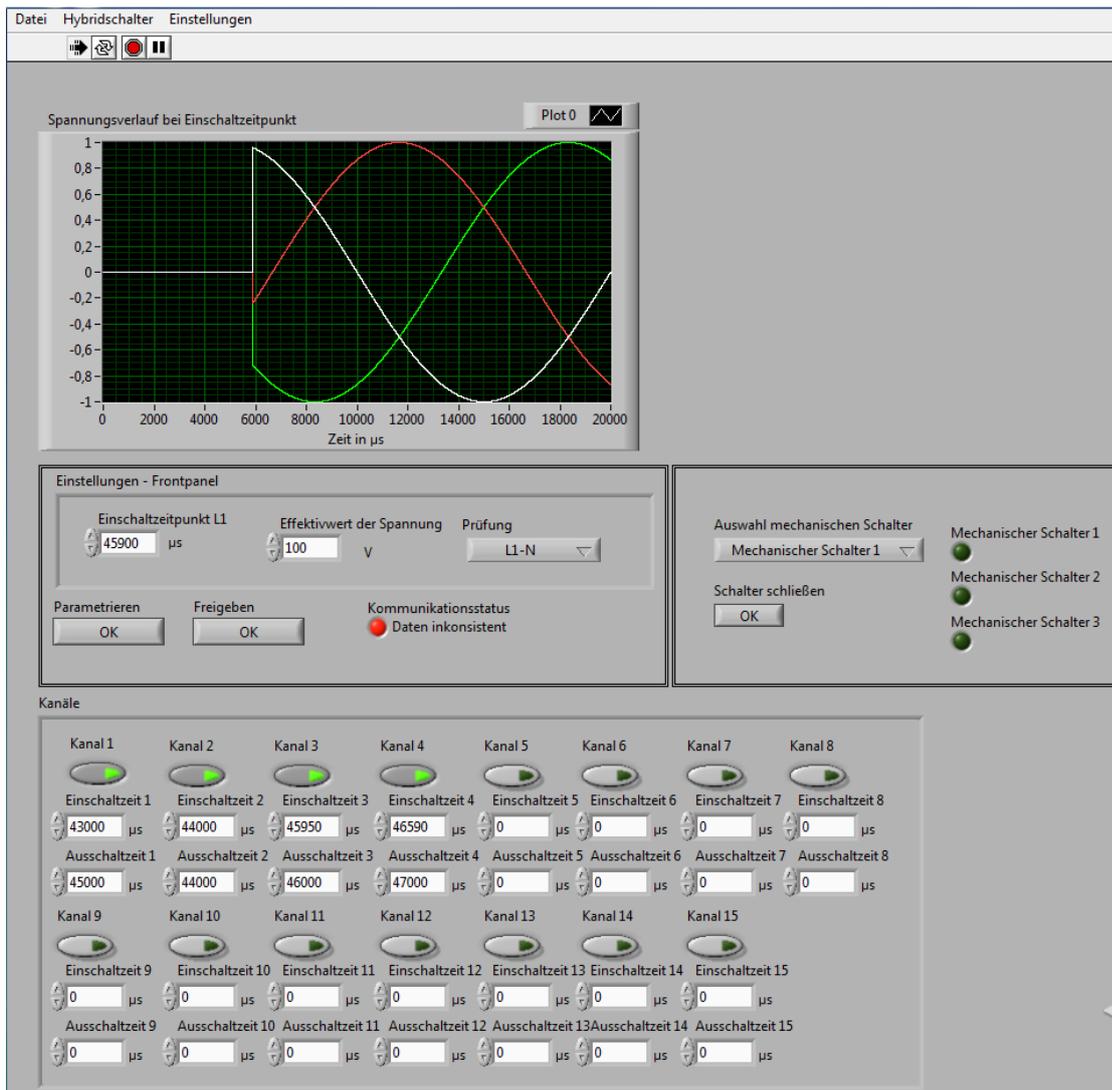


Abbildung 8.8: Frontpanel - Hauptprogramm (Labview)



Abbildung 8.9: Dialog - Einstellungen Hybridschalter

Einstellungen in „ZB Ein 1 – 40 μs “ und „ZB Aus 1 – 40 μs “ bedeuten, dass der Einschaltimpuls für den Thyristor der Zusatzbeschaltung 40 μs vor dem Einschaltimpuls der optisch-triggerbaren Thyristoren beginnt und 40 μs nach dem Einschaltimpuls der optisch-triggerbaren Thyristoren endet.

Die Spannungsgrenzen für den Einsatz der Zusatzbeschaltung bzw. für die untere Spannungsschwelle sind ebenfalls hier einzustellen. Die Standardwerte sind für $U_{Z,min} = 80\text{ V}$ und $U_{h,min} = 20\text{ V}$. Die Einschaltverzögerungen zwischen L1-L2-L3 sind ebenfalls noch einzustellen. Sie stellen den Abstand zwischen den einzelnen Schaltvorgängen in Phasen dar (nur im Falle einer dreiphasigen Prüfung). Die Standardwerte hierfür liegen im Bereich von 20 μs für L2 und 40 μs für L2.

Erfassung der mechanischen Verzögerungszeiten

Wie schon vorher erläutert, stellt die Kenntnis der mechanischen Verzögerungszeiten neben den eingegebenen Parametern die Grundlage für die Berechnung der Signalverlaufstabelle dar. Mit Hilfe des Dialogs - Erfassung Verzögerungszeiten (Menü Hybrid-schalter \rightarrow Verzögerungszeiten Leistungsschalter) in Abbildung 8.10, wird die Visualisierung und die Verwaltung der Verzögerungszeiten ermöglicht. Die Visualisierung erfolgt für jeden der drei mechanischen Schalter separat mit Hilfe einer Häufigkeitsverteilung. Die gemessenen Verzögerungszeiten werden automatisch in tabellarischer Form gespeichert und können ggf. editiert werden. Für die Berechnung der Signalverlaufstabelle wird das Minimum der bisher gemessenen Verzögerungszeiten herangezogen. Wichtig für die Benutzung ist auch die Eingabe der Verzögerungszeit zwischen dem Hauptkontakt und dem Meldekontakt. Diese sind nach den Messungen in Abschnitt 9.3 mit 2400 μs zu bemessen.

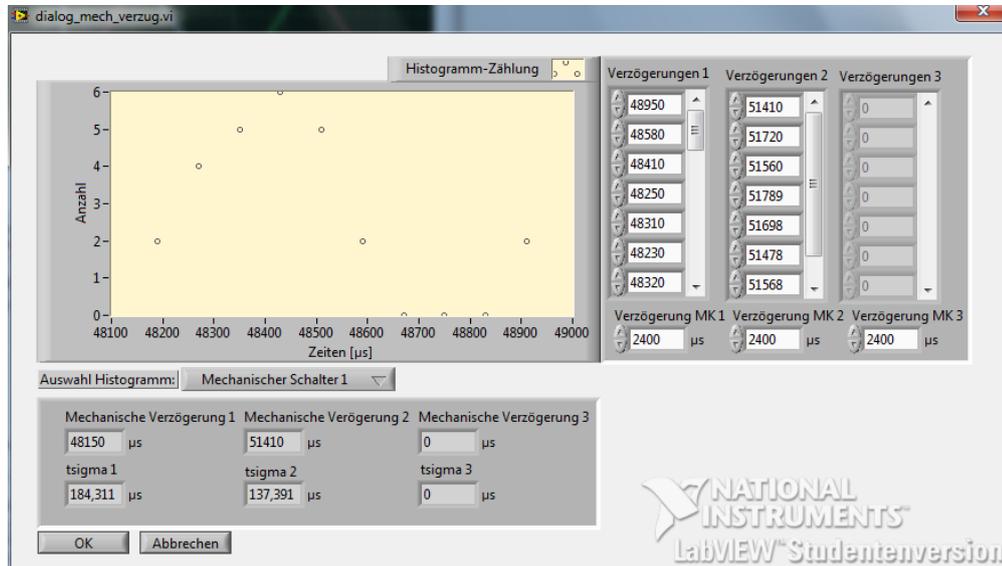


Abbildung 8.10: Dialog - Erfassung Verzögerungszeiten

Einstellungen Serielle-Schnittstelle

Die Einstellungen für die serielle Schnittstelle (Menü Einstellungen → Serielle-Schnittstelle) betreffen die Kommunikation zwischen Synchronsteuerung und dem PC auf dem LabView eingesetzt wird. Die eingesetzt USB-Schnittstelle emuliert eine RS232-Schnittstelle auf dem PC und muss daher auch entsprechend eingesetzt werden. Die Einstellungen der Synchronsteuerung sind in Tabelle 8.3 gegeben.

Einstellungen RS232-Schnittstelle	
Baudrate	250000
Flusskontrolle	kein
Stoppbit	1
COM-PORT	„aktueller PORT“
Time-out	10000 ms

Tabelle 8.3: Einstellungen RS232

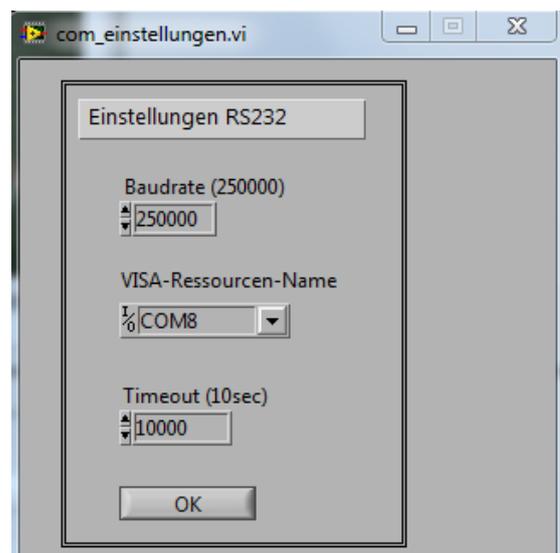


Abbildung 8.11: Dialog - Serielle Schnittstelle Einstellungen

8.5 Implementierung der Signalverlaufstabellen

Anforderungen

Die Signalverlaufstabellen stellen den Ablauf der Steuerung in tabellarischer Form dar. Sie werden von der Synchronsteuerung für die Steuerung der freien Kanäle verwendet und von der Ansteuerung des Hybridschalters für die Steuerung des mechanischen Schalters, der optisch-triggerbaren Thyristoren und der Zusatzbeschaltung.

Die Tabelle 8.4 zeigt exemplarisch das Format einer Signalverlaufstabelle. Dabei stellen die Elemente in der Spalte „Zeit“ einen Zeitpunkt⁶ dar, bei dem sich das Byte eines Ausgangsport (hier Ansteuerung eines Hybridschalters) ändert. Bei der Ausführung einer solchen Tabelle werden die entsprechenden Bytes zu den jeweiligen Zeitpunkten dann neu gesetzt. Die Erstellung dieser Tabelle gestaltet sich für die Synchronsteue-

Zeit [μ s]	Hybridschalter L1	Hybridschalter L2	Hybridschalter L3
13610	255	254	255
16850	255	255	254
64960	255	250	254
64980	255	250	250
.	.	.	.
.	.	.	.

Tabelle 8.4: Beispiel einer Signalverlaufstabelle für den Hybridschalter

rung relativ einfach, da hier der Anwender direkt die Zeitpunkte vorgibt und auch die Kanäle entsprechend aktiviert. Für die Ansteuerung des Hybridschalters gestaltet sich die Erstellung etwas komplizierter, da hier erst aus den Parametern die Tabelle erstellt werden muss.

Signalverlaufstabelle - Synchronsteuerung

Für die Erstellung der Signalverlaufstabelle der Synchronsteuerung gibt der Anwender für den jeweiligen Kanal den Einschaltzeitpunkt sowie den Ausschaltzeitpunkt vor. Darüber hinaus muss der Anwender die betreffenden Kanäle aktivieren/deaktivieren. In Abbildung 8.12 ist der prinzipielle Ablauf der Erstellung einer Signalverlaufstabelle für die Synchronsteuerung abgebildet. Nachdem die Eingaben erfolgt sind, werden die aktivierten Kanäle zu Bytes zusammengefasst, sodass der Mikrocontroller bei der Abarbeitung nur noch die entsprechenden Ausgangsports zuweisen muss. Die Tabelle wird in LabView auch noch chronologisch sortiert, dadurch ist auch eine chronologische Abarbeitung möglich. Auf der Ebene des Mikrocontrollers muss die Tabelle nur mehr an den Zähler angepasst werden, wenn dieser die Ausführung nicht in 1μ s Schritten abarbeitet.

⁶Der Zeitpunkt wird in μ s angegeben und bezieht sich auf den synchronisierten Spannungsnulldurchgang

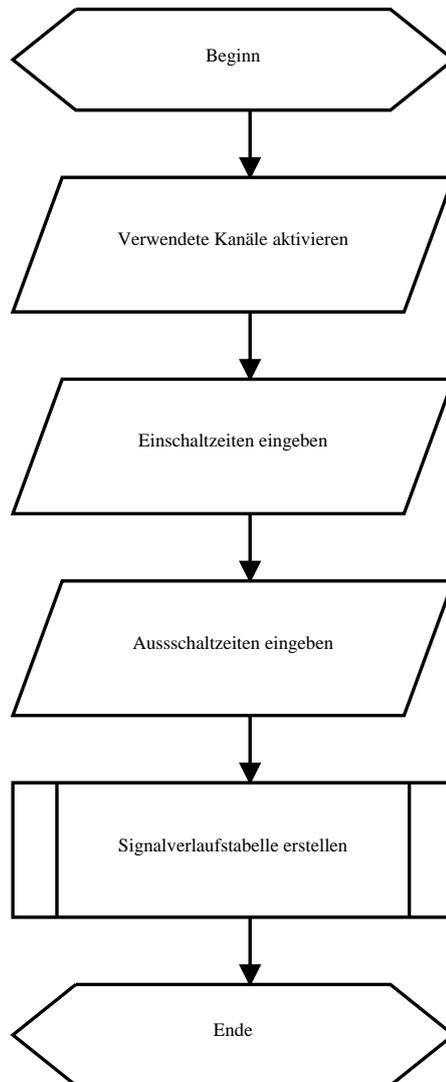


Abbildung 8.12: Ablaufdiagramm - Signalverlaufstabelle Synchronsteuerung

Signalverlaufstabelle - Hybridschalter

Die Signalverlaufstabelle des Hybridschalters wird aus einzugebenden Parametern und gemessenen Parametern erstellt bzw. errechnet. Der Benutzer muss den Einschaltzeitpunkt und den Effektivwert der Spannung eingeben. Die gemessenen Parameter der Verzögerungszeiten des mechanischen Schalters und die Einstellungen für die Verwendung der Zusatzbeschaltung müssen ebenfalls herangezogen werden.

In Abbildung 8.13 sind die notwendigen Schritte für die Erstellung einer Signalverlaufstabelle ersichtlich. Im ersten Schritt werden die einzelnen Schaltzeitpunkte (wie z.B.: Einschaltzeitpunkte der mechanischen Schalter, Einschaltimpulse Zusatzbeschaltung etc.) nach Abschnitt 8.1 berechnet und daraus die Basis der Signalverlaufstabelle erstellt.

Der Einschaltzeitpunkt muss für jede einzelne Phase (L1,L2,L3) noch genau analysiert werden, was sowohl die Stromleitdauer (Abstand bis zum nächsten Nulldurchgang des Stromes und damit sperren des Thyristors) wie auch den Momentanwert der Anoden-Kathodenspannung betrifft. Dafür werden die Abstände bis zum nächsten Nulldurchgang berechnet und auch die Momentanwerte der einzelnen Spannungen berechnet. In Abhängigkeit der berechneten Werte werden dann nach einem Entscheidungsbaum in Abbildung 8.14 die Ausgangsports für die einzelnen Ansteuerungen des Hybridschalters gesetzt. Hier muss entschieden werden, welche der Thyristoren, abhängig von der Stromrichtung gezündet werden müssen, ob die Zusatzbeschaltung benötigt wird und ob die Stromleitdauer für die Thyristoren ausreichend ist.

Die Tabelle wird in LabView auch wieder chronologisch sortiert, sodass auch eine chronologische Abarbeitung möglich ist. Auf der Ebene des Mikrocontrollers muss die Tabelle an den Zähler angepasst werden, wenn dieser die Ausführung nicht in $1 \mu s$ Schritten abarbeitet.

8.5 Implementierung der Signalverlaufstabellen

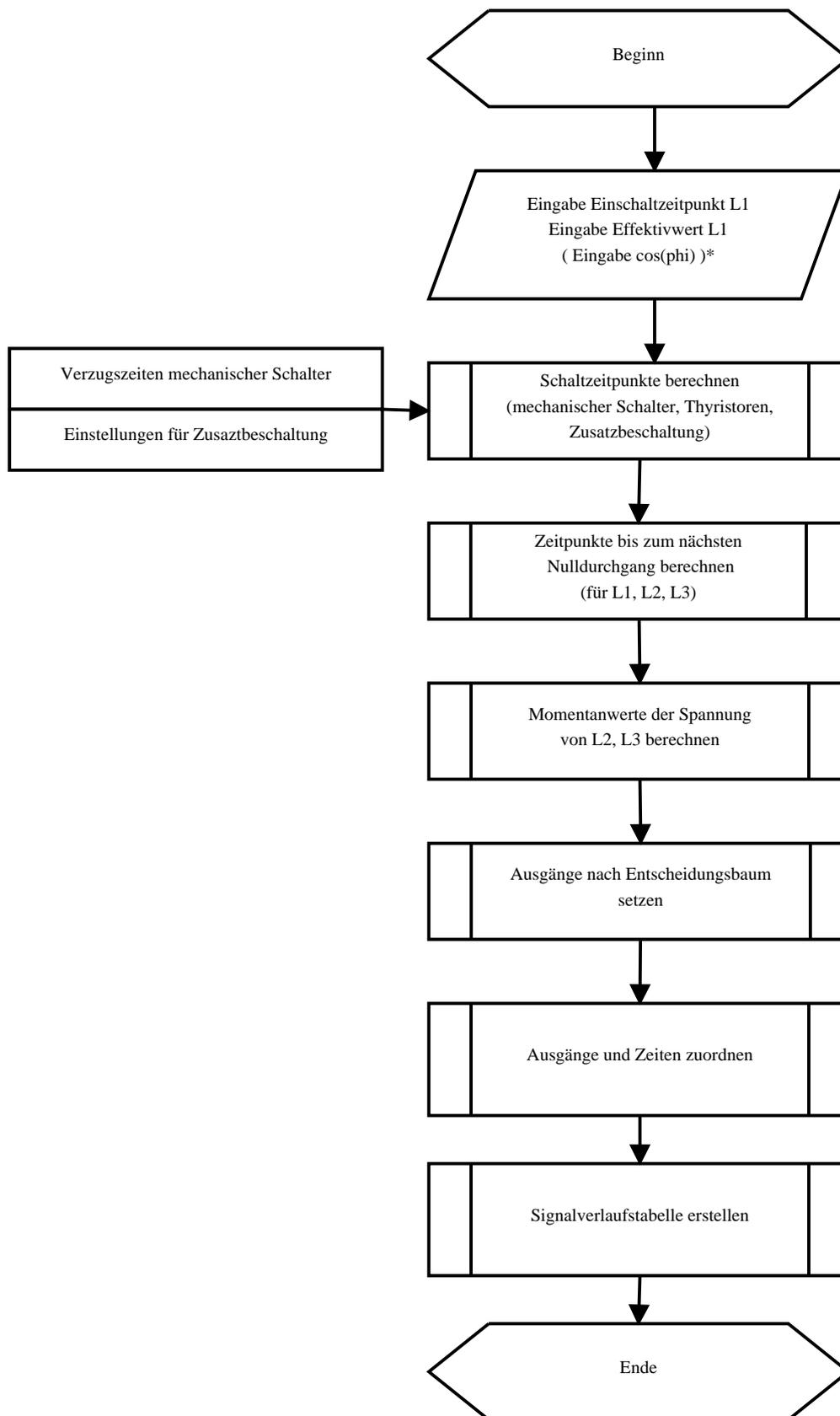


Abbildung 8.13: Ablaufdiagramm - Signalverlaufstabelle Hybridschalter

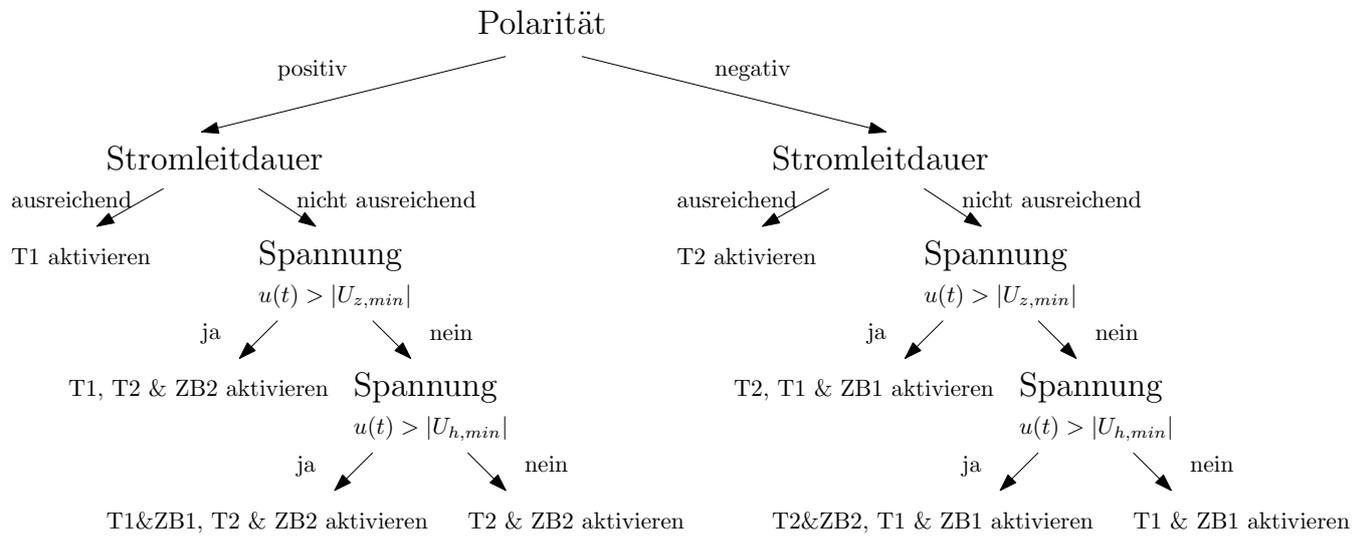


Abbildung 8.14: Entscheidungsbaum - Hybridschalter

9 Messungen

9.1 Allgemeines

Für die Bestimmung der Verzögerungszeiten der Solid-State Relais, der Verzögerungszeiten der mechanischen Schalter und der Funktion der Hybridschalter wurden entsprechende Messungen durchgeführt. Da die Einschaltvorgänge transient und nicht periodisch sind, wird für die Messungen ein Transientrekorder verwendet. Die eingesetzten Messgeräte sind im Folgenden angegeben:

- Transientenrekorder Nicolet BE256-XE (mit der Software Team256 Pro)
- Differentialastkopf Testec TT-SI 9001x10/x100

Die angeführten Messungen wurden lediglich für einen einphasigen Testaufbau durchgeführt.

9.2 Verzögerungsmessung der Solid-State Relais

Für die Entwicklung der Ansteuerung ist es von großer Bedeutung, die Verzögerung der einzelnen Bauelemente zu kennen bzw. zu bestimmen. Da der Hersteller keine Angaben über die Verzögerungszeit der Solid-State Relais macht, muss diese gemessen werden. Bei dem verwendeten Solid-State Relais handelt es sich um ein Crydom D4D07 mit MOSFET Ausgang. Zu diesem Zweck wird der Einschaltimpuls des Mikrocontrollers (Eingang des Solid-State Relais), sowie die geschaltete Steuerspannung (Ausgang des Solid-State Relais) mit dem Transientenrekorder gemessen. Tabelle 9.1 zeigt die gemessenen Werte, welche sich relativ zu dem Einschaltzeitpunkt der Thyristoren beziehen. Diese Tabelle zeigt, dass die Verzögerung der Solid-State Relais lediglich 20 μs be-

Einschaltimpuls (Eingang SSR)	220 VDC Ausgang (Ausgang SSR)
-48310 μs	-48330 μs
-48320 μs	-48340 μs
-48290 μs	-48310 μs

Tabelle 9.1: Verzögerungsmessung des Solid-State Relais (Crydom D4D07)

trägt. Dieser Wert ist im Vergleich zur Verzögerung der mechanischen Schalter, welche im Bereich von ca. 50 ms liegt, marginal. In Abbildung 9.1 ist eine Verzögerungsmessung mit dem Transientenrekorder und der entsprechenden Auswertung dargestellt. Dabei ist der Eingangsimpuls („Is out“) und der Ausgangsimpuls („SSRout“) dargestellt und die gemessene Verzögerung beträgt 19,86 μs .

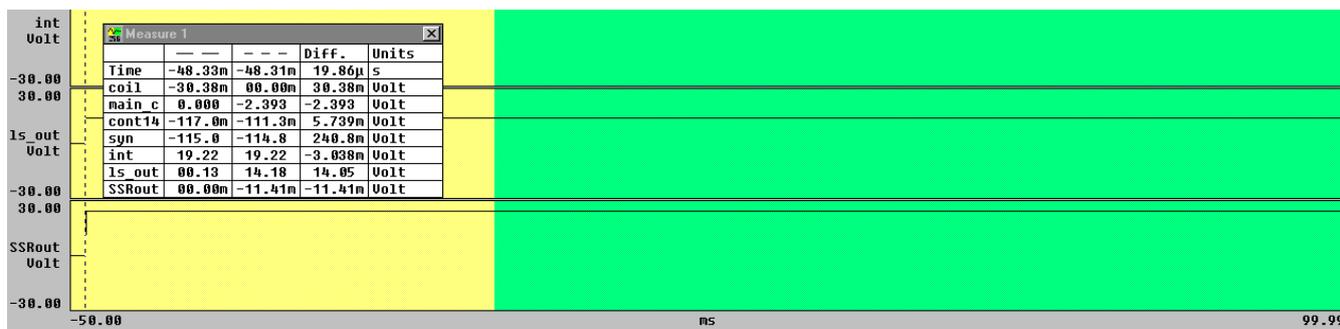


Abbildung 9.1: Messergebnis - Verzögerung der Solid-State Relais

9.3 Verzögerungszeiten eines mechanischen Schalters

Um die Verzögerungszeiten eines mechanischen Schalters bestimmen zu können, wird der Einschaltimpuls an der Einschaltpule und der Einschaltvorgang im Hauptstromkreis (Hauptkontakte) aufgezeichnet. Zusätzlich wird auch noch die Einschaltverzögerung zwischen den Hauptkontakten und der Meldekontakte aufgezeichnet. Tabelle 9.2 zeigt die Messergebnisse der Verzögerungszeiten. Aus den Messergebnissen geht

Verzögerung Hauptkontakt	Verzögerung Meldekontakt / Hauptkontakt
51410 μ s	2410 μ s
51720 μ s	2396 μ s
51436 μ s	2411 μ s
51587 μ s	2397 μ s
52018 μ s	2410 μ s
51892 μ s	2398 μ s
51256 μ s	2401 μ s

Tabelle 9.2: Verzögerungsmessungen mechanischer Schalter (Masterpact NW20)

hervor dass die Verzögerung dieses mechanischen Schalters (Masterpact NW20) im Bereich von 51 ms-52 ms liegt. Messungen nach [3] haben Verzögerungen im Bereich von 48 ms-49 ms festgestellt. Dabei ist aber anzumerken, dass die absolute Verzögerung zwar bekannt sein muss aber keine Auswirkungen auf die Steuerung hat. Die Streuung allerdings muss für das verwendete Konzept für alle eingesetzten mechanischen Schalter im gleichen Bereich liegen, da ansonsten die Stromführung der Thyristoren möglicherweise zu lange dauert und eine thermische Überlastung auftreten könnte.

Für die Messung der Verzögerungszeiten der mechanischen Schalter im Hybridschalterbetrieb, wird ein Meldekontakt verwendet. Die Messungen haben hier ergeben, dass zwischen dem Schalten des Hauptkontaktes und des Meldekontaktes eine Verzögerung von rund 2,4 ms besteht. Diese Zeit muss bei den Messungen während des Betriebs berücksichtigt werden um die korrekten Verzögerungszeiten der mechanischen Schalter bestimmen zu können.

In Abbildung 9.3 ist eine Messung der Verzögerungszeit eines mechanischen Schalters

Vorgabe Einschaltzeitpunkt	Tatsächlicher Einschaltzeitpunkt	Verzögerung Meldekontakt (gemessen)	Verzögerung Meldekontakt Labview
5000 μs	4989 μs	54210 μs	54340 μs
7000 μs	7011 μs	53920 μs	53888 μs
8000 μs	8004 μs	53560 μs	53565 μs
15000 μs	15060 μs	53980 μs	53960 μs
17000 μs	17050 μs	53790 μs	53815 μs

Tabelle 9.3: Einschaltvorgänge Hybridschalter

dargestellt. Dabei ist der Einschaltimpuls an der Einschaltpule („coil“) und der Einschaltvorgang am Hauptkontakt (main c) zu sehen.

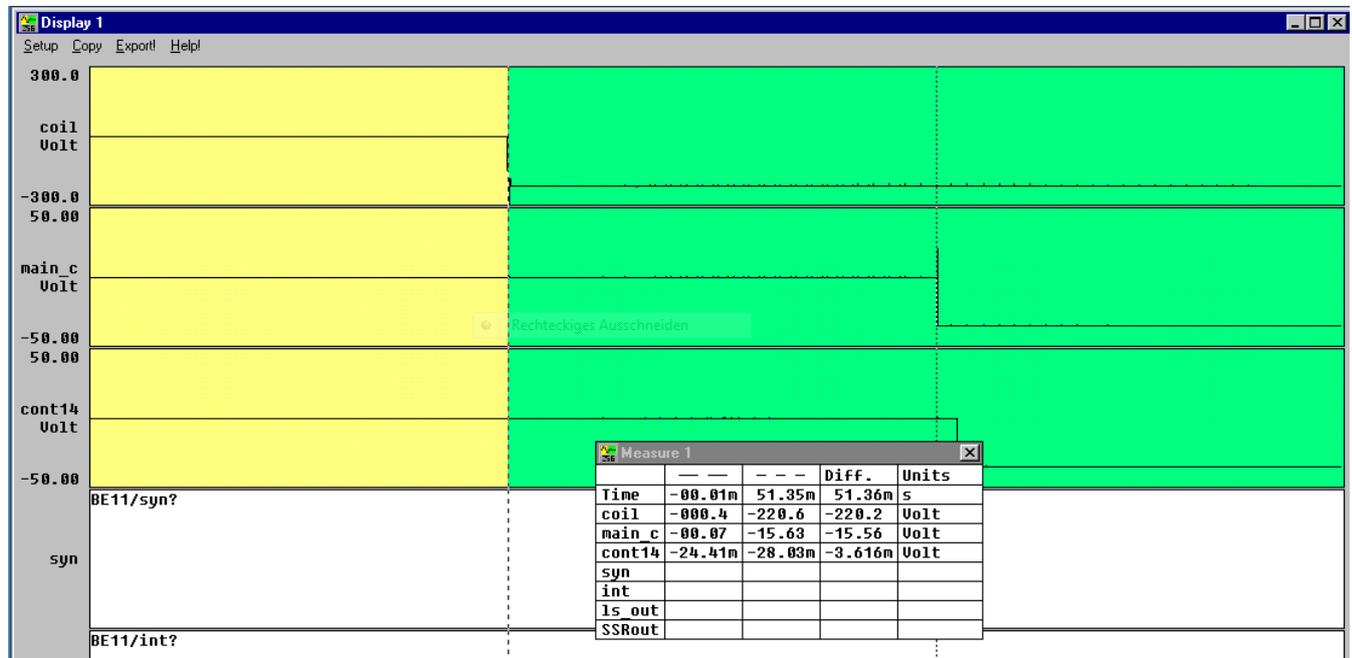


Abbildung 9.2: Messergebnis - Verzögerungszeit mechanischer Schalter

9.4 Einschaltvorgänge Hybridschalter

Um die Funktion der Ansteuerung des Hybridschalters zu validieren, wurden mehrere Einschaltvorgänge zu verschiedenen Einschaltzeitpunkten¹ aufgezeichnet. Dabei wurden die Einschaltzeitpunkte über die grafische Benutzeroberfläche vorgegeben. Der tatsächliche Einschaltzeitpunkt ist durch den Einschaltzeitpunkt der Thyristoren gegeben und weist in Tabelle 9.3 lediglich kleine Abweichungen auf.

¹Die Einschaltzeitpunkte sind relativ auf einen Spannungsnulldurchgang bezogen

9 Messungen

Um die Genauigkeit des integrierten Messsystems für die Verzögerungszeiten der mechanischen Schalter zu überprüfen, werden die Verzögerungszeiten der Meldekontakte ebenfalls mit Transientenrekorder gemessen. Dabei ergibt sich zwischen den gemessenen und dem in der Ansteuerung integrierten Messsystem eine Abweichung von $10\ \mu\text{s} - 20\ \mu\text{s}$. In Abbildung 9.3 ist eine Messung eines hybriden Einschaltvorganges mit $5000\ \mu\text{s}$ nach dem Spannungsnulldurchgang zu sehen. Dabei ist wieder der Einschaltimpuls an der Einschaltspule („coil“) und der Einschaltvorgang am Hauptkontakt (main c) zu sehen. Zusätzlich ist auch der Verlauf der Spannung des Meldekontaktes („cont14“) und der Netzspannung („syn“) zu sehen. Der Einschaltzeitpunkt ist hier wieder genau im Spannungsmaximum (entspricht den eingestellten $5000\ \mu\text{s}$ nach dem Spannungsnulldurchgang). Der gemessene Einschaltzeitpunkt nach dem Spannungsnulldurchgang beträgt $4989\ \mu\text{s}$.

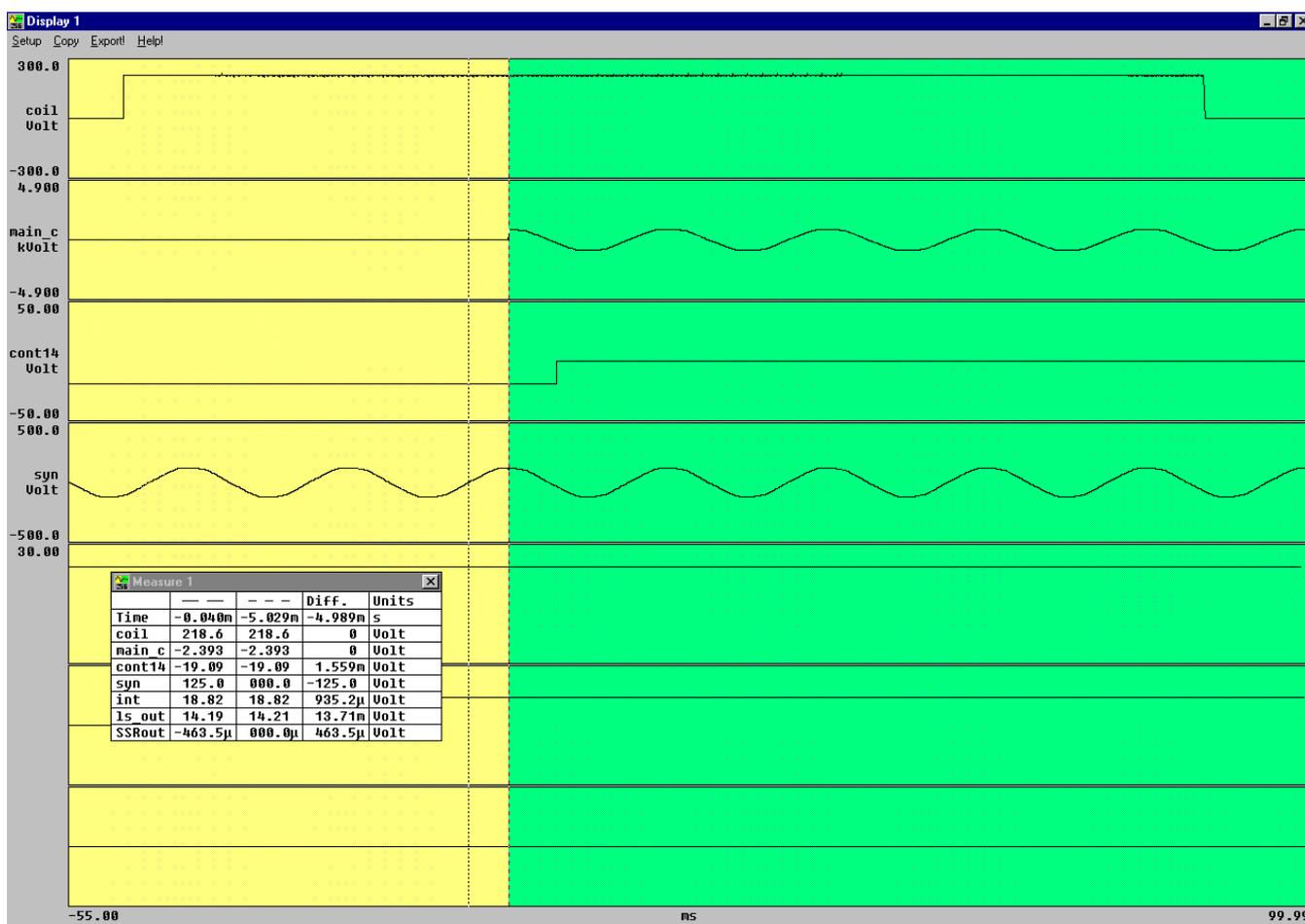


Abbildung 9.3: Messergebnis - Einschaltvorgang Hybridschalter nach $5000\ \mu\text{s}$

10 Validierung mit Hardware-in-the-loop

10.1 Simulation mit Hardware-in-the-loop

Konzept

Um die Funktionen der entwickelten Hardware zu testen, gibt es zunächst einmal die Möglichkeit einen reinen softwarebasierten Ansatz zu verfolgen. Dabei wird die entwickelte Hardware modelliert und dann simuliert. Der Nachteil bei dieser Art der Validierung ist, dass die reale Umgebung der entwickelten Hardware nie zur Gänze nachgebildet werden kann.

Echtzeit¹ Hardware-in-the-loop (HIL) Systeme bieten die Möglichkeit, die entwickelte Hardware in einem simulierten elektrischen Netzwerk zu testen. Dabei werden im Allgemeinen die Ein-/Ausgänge der entwickelten Hardware mit einem Echtzeitsimulationssystem verbunden, welche die elektrische Umgebung des realen Systems nachbildet. Bei HIL-Simulationen geht es dabei in erster Linie um die Validierung von Steuerungen bzw. Regelungen. Die entwickelte Hardware wird dann in eine simulierte Regelschleife integriert. Das Echtzeitsystem gibt dann die Führungsgröße vor, verarbeitet als virtuelle Regelstrecke dann die Stellgröße und beeinflusst damit in der geschlossenen Schleife die Regelungsgröße bzw. die Rückführung. Einsatzgebiete von HIL sind:

- Anlagenbau (z.B.: SPS, Steuerungen für Verfahren, einzelne Controller etc.)
- Mobilitätsbereich (Automobilbereich, elektrische Antriebe etc.[20])
- Luft -und Raumfahrttechnik (Regelungen, Avionik)
- Leistungselektronik (Steuerungen von DC/DC Konvertern, Steuerungen von H-Bridges etc. [21])

Der Vorteil von HIL ist, dass die Umgebung bzw. die zu steuernden Komponenten der entwickelten Hardware physikalisch nicht vorhanden sein müssen um die Funktion validieren zu können. Dies impliziert, dass während des Entwicklungsprozesses möglicherweise große und kostenintensive Komponenten (z.B.: Motoren und Antriebe, el. Anlagen, Traktionssysteme etc.) nicht vorhanden sein müssen und somit im Falle eines Fehlers nicht beschädigt werden können.

Die Simulationsfähigkeit eines Systems ist abhängig von dessen Größe und besonders von der Frage, wie genau es modelliert werden muss um alle signifikanten Effekte zu erfassen. HIL muss prinzipiell in Echtzeit ablaufen. Wird das eingesetzte Modell zu groß bzw. ist der eingesetzte Simulator zu langsam um die Berechnungen des Modells in der vordefinierten Zeit abzuarbeiten, so kommt es zu sogenannten „Overruns“. Es können also keine beliebig großen Modelle erstellt und ausgeführt werden.

Für die Erstellung der Modelle bietet sich Matlab/Simulink an, da hier beliebig detail-

¹Nach [19] bedeutet Echtzeit lediglich, dass ein System auf ein Ereignis innerhalb eines vorgegebenen Zeitrahmens reagieren muss. Der Begriff sagt nichts über die Geschwindigkeit oder Verarbeitungsleistung eines Systems aus. Es muss vorhersehbar sein wann das System auf eine Eingangsgröße reagiert.

lierte Modelle erstellt werden können. In Abbildung 10.1 ist der prinzipielle Ablauf einer HIL-Simulation dargestellt. Zuerst wird, ein Modell des gesamten Systems in Mat-

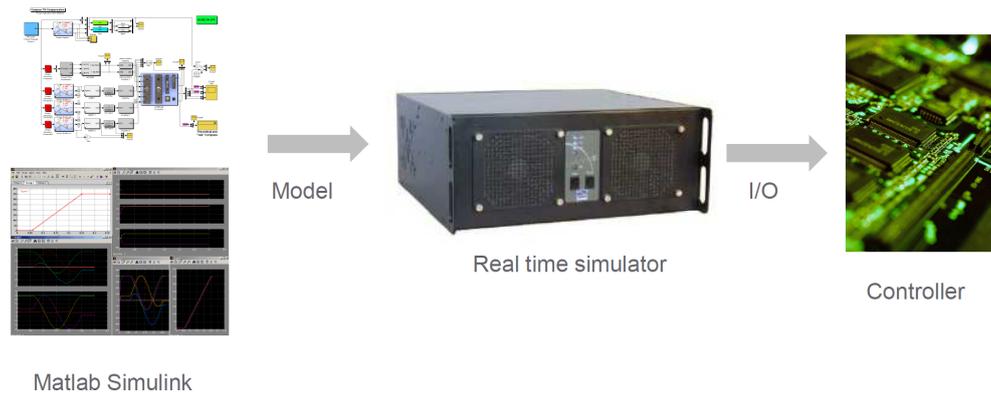


Abbildung 10.1: Konzept von Hardware-in-the-loop

lab/Simulink erstellt. Dann wird das Modell C-Code übersetzt und anschließend kompiliert, also in eine für das Echtzeitsystem ausführbare Form gebracht. Anschließend wird die entwickelte Hardware mit dem Echtzeitsystem verbunden und ausgeführt.

Opal RT

Das eingesetzte Echtzeitsystem ist von der Firma Opal RT mit einem Echtzeit-Linux Derivat von Redhat. Die Spezifikationen des Echtzeitsystems sind nach [22] in Tabelle 10.1 dargestellt.

Spezifikationen Opal-RT	
CPU	2× Quad Core Intel i7 CPU 3 GHz
Speicher	2 GB
Analog-Input	16 Kanäle, 16-bit, $\pm 1 V - \pm 16 V$
Analog-Output	16 Kanäle, 16-bit $\pm 5 V - \pm 100 V$
Digital-Input	32 Kanäle, 5 V - 24 V
Digital-Output	32 Kanäle, 5 V - 24 V
Minimal Step-time	10 μs

Tabelle 10.1: Spezifikationen Opal-RT [22]

Abbildung 10.1 zeigt das Echtzeitsystem Wanda 4 von Opal RT.

10.2 Offline Simulation

Für die Modellbildung und um anschließend das Modell testen zu können, ist die offline Simulation von Vorteil. Die Steuerung wird bei dieser Art der Simulation eben-



Abbildung 10.2: Opal RT Echtzeitsimulator Wanda 4 [22]

falls modelliert. Es handelt sich hierbei um eine reine Simulation, die kein Echtzeitsystem benötigt. Darüber hinaus ist hier möglich verschiedene Konzepte für die Steuerung zu evaluieren, ohne dabei die entwickelte Hardware zu testen. Für die Simulation

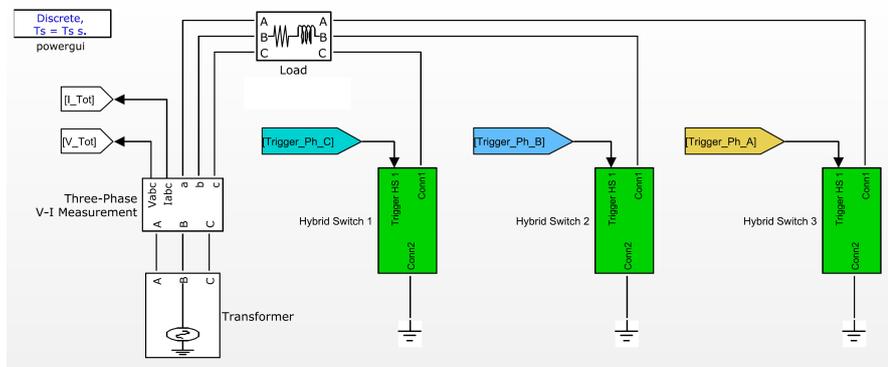


Abbildung 10.3: Erstelltes Matlabmodell nach [23]

wurde auf ein vorhandenes Matlab/Simulink Modell des Hybridschalters zurückgegriffen, dass in [23] erstellt wurde. Für die Programmierung wurde die SimPowerSystems Toolbox von Simulink verwendet. In Abbildung 10.3 wurde ein dreiphasiges Modell des Hybridschaltersystems implementiert. Ausgehend von einem Transformator der als dreiphasige Spannungsquelle modelliert ist, sind die Hybridschalter über eine ohmsch-induktive Last verbunden. Für das offline Modell ist die Steuerung ebenfalls modelliert und befindet sich im unteren Teil der Abbildung. Für die Synchronisation mit der Spannungsquelle wird der oberste Eingang des Steuerungssystems verwendet, die Verzögerungszeiten der mechanischen Schalter können ebenfalls in dem Modell als Parameter berücksichtigt werden. Die unteren zwei Eingänge der Steuerung werden für die Anzeige von Messergebnissen verwendet. Die Ausgänge der Steuerung geben die Triggerimpulse für die Hybridschalter aus, wobei hier sowohl die Triggerimpulse für die mechanischen Schalter als auch für die Thyristoren übertragen werden. Das Modell für die offline Simulation kann für erste Analysen des gesamten Systems verwendet werden. Es können hier allerdings EMV-Einflüsse, sonstige Störungen und das exakte Verhalten der Steuerung nicht nachgebildet werden. Um das Verhalten der entwickelten Steuerung zu validieren, wird im Folgenden die simulierte Steuerung durch den hardwaremäßigen Aufbau in einer Hardware-in-the-loop Umgebung ersetzt.

10.3 Hardware-in-the-loop - Simulationen

Für den Einsatz des Modells in einer Hardware-in-the-loop Umgebung muss es noch entsprechend adaptiert werden. Die Ein- bzw. Ausgänge des modellierten Controllers werden für den Einsatz der hardwaremäßigen Steuerung adaptiert. Die entwickelte Hardware wird dann mit dem vorher beschriebenen Echtzeitsystem von Opal RT verbunden. Das angepasste Modell wird kompiliert und am Echtzeitsystem ausgeführt. Das verwendete Simulationskonzept ist in Abbildung 10.4 dargestellt. Dabei bleibt das Modell des Hybridschalters unverändert und wird, getrennt von der übrigen Simulation, von einer eigenen CPU ausgeführt. Die Ein- bzw. Ausgänge der modellierten Steuerung sind über Schnittstellen des Opal RT Systems mit der Hardware der entwickelten Steuerung verbunden und werden ebenfalls auf einer eigenen CPU verarbeitet.

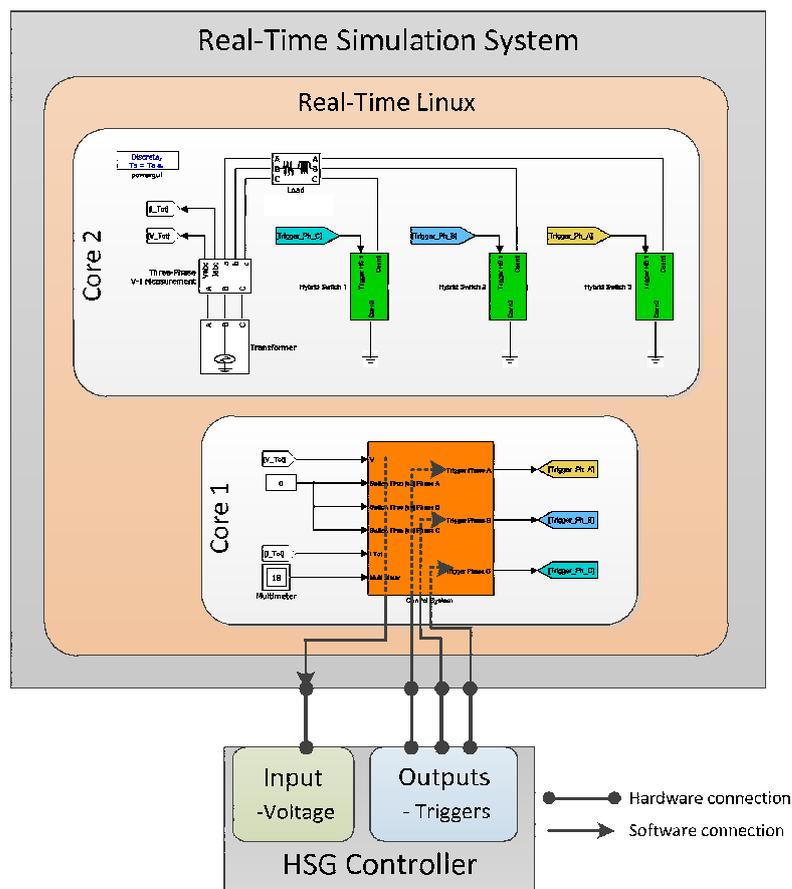


Abbildung 10.4: Hardware-in-the-Loop System nach [23]

Für die Entwicklung der Hardware für die Steuerung des Hybridschalters ist es wichtig, die Funktionen validieren zu können. In diesem Fall ist es ausreichend, lediglich eine einphasige Simulation durchzuführen, da die Hardware für das dreiphasige System entsprechend repliziert wurde.

Für die Validierung des entwickelten Konzepts wird im Folgenden der einphasige hybride Schaltvorgang genauer analysiert. Das Ergebnis der ersten Simulation ist in Abbildung 10.5 dargestellt. Dabei wird die Netzspannung über das Echtzeitsystem an der Steuerung simuliert (mittleres Diagramm - Grid Voltage). Die Steuerung synchronisiert auf die Netzspannung und löst entsprechende Triggerimpulse für den mechanischen Schalter und die Thyristoren aus (unteres Diagramm - Trigger). Für den Einschaltzeit-

punkt wurden $1000 \mu\text{s}$ nach einem Spannungsnulldurchgang auf der Steuerung eingestellt. Dies ist der Zeitpunkt, bei dem die Thyristoren einschalten müssen und kurz darauf der mechanische Schalter schließen muss. Daher muss der Triggerimpuls für den mechanischen Schalter entsprechend der Verzögerungszeit vorher erfolgen². Im obersten Diagramm ist der Stromverlauf über den Hybridschalter dargestellt (Current). Dabei zeigt der vergrößerte Bereich den Einschaltvorgang mit den Thyristoren und eine anschließende Kommutierung auf den mechanischen Schalter. Die verschiedenen qualitativen Stromverläufe ergeben sich auf Grund der unterschiedlichen Impedanzen der Thyristoren und der mechanischen Schalter. Es ist in Abbildung 10.5 ersichtlich, dass

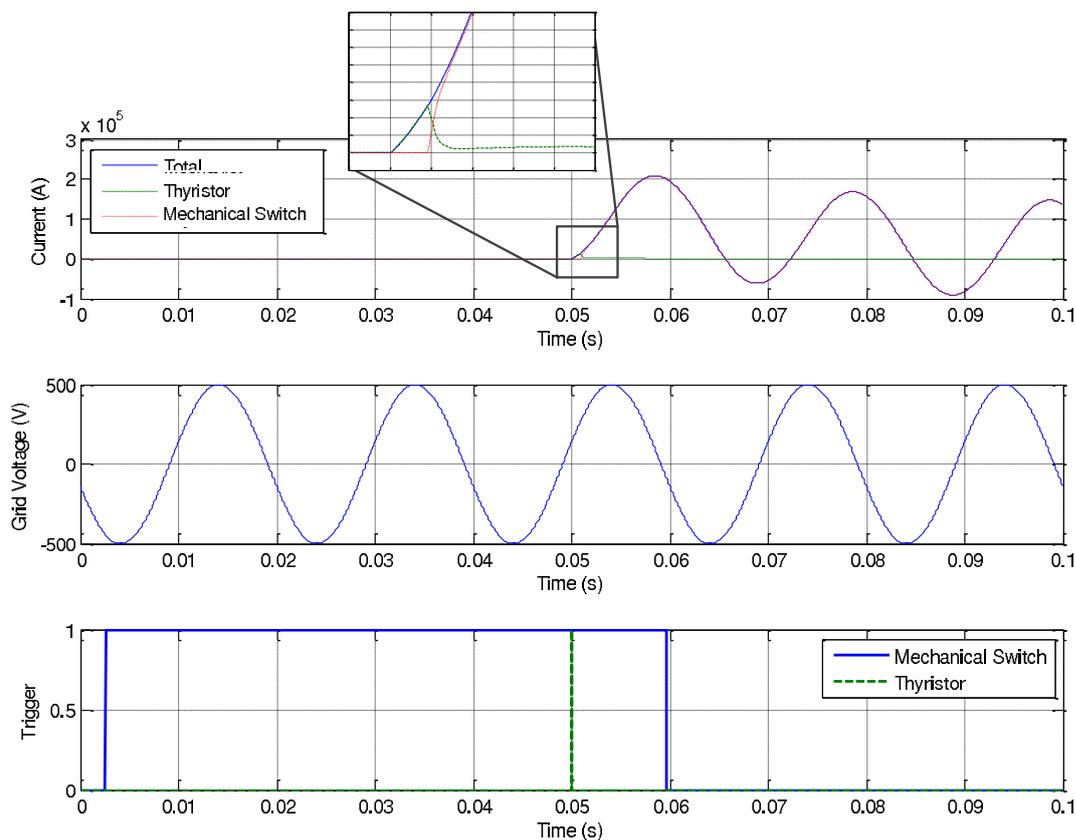


Abbildung 10.5: Simulationsergebnis - Einschaltzeitpunkt $1000 \mu\text{s}$ [23]

der Einschaltvorgang des Stromes $1000 \mu\text{s}$ nach dem Spannungsnulldurchgang erfolgt und auch die Verzögerung des mechanischen Schalters richtig berücksichtigt wurde und somit eine erste Validierung des Konzepts erfolgt ist.

Im Folgenden werden nun zwei weitere Simulationsergebnisse gezeigt, um die prinzipielle Funktion der Ansteuerung eines Hybridschalters zu beweisen. In Abbildung 10.6 ist das Ergebnis der Hardware-in-the-loop Simulation mit einem eingestellten Einschaltzeitpunkt von $2000 \mu\text{s}$, nach einem Spannungsnulldurchgang, zu sehen.

In Abbildung 10.7 ist das Ergebnis der Hardware-in-the-loop Simulation mit einem eingestellten Einschaltzeitpunkt von $5000 \mu\text{s}$, nach einem Spannungsnulldurchgang, zu sehen.

²Es wird angemerkt, dass die Zeitbasis in den Diagrammen lediglich relativ zu interpretieren ist. Bei den Simulationen wird immer auf den Einschaltvorgang des Stromes getriggert. Um aber die Triggerimpulse der mechanischen Schalter ebenfalls erfassen zu können, müssen die Diagramme bis $0,05 \text{ s}$ vor dem Einschaltvorgang ebenfalls aufgezeichnet werden.

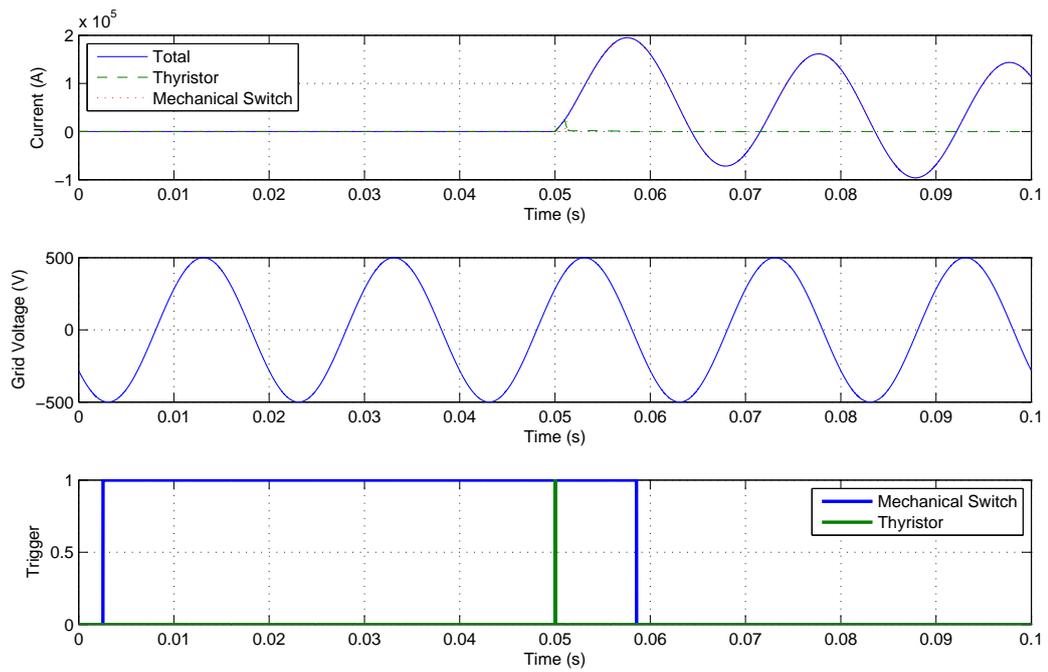


Abbildung 10.6: Simulationsergebnis - Einschaltzeitpunkt $2000 \mu s$

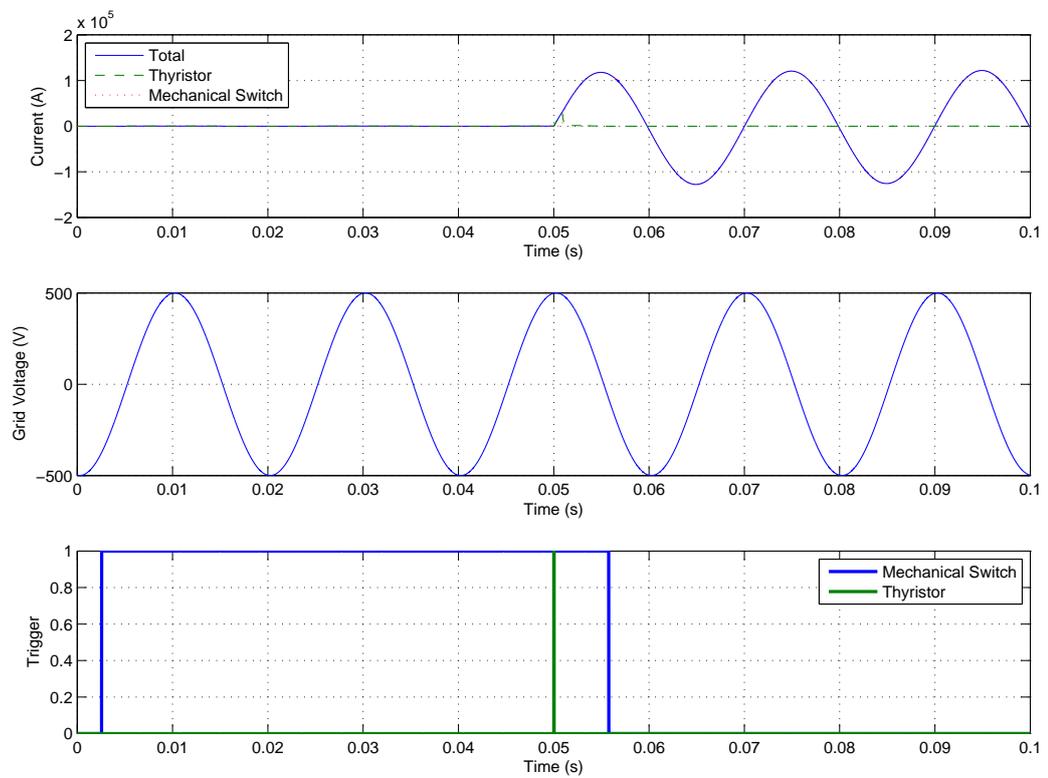


Abbildung 10.7: Simulationsergebnis - Einschaltzeitpunkt $5000 \mu s$

11 Schlussfolgerungen

Im Rahmen dieser Diplomarbeit konnte das entwickelte Konzept der Ansteuerung bereits durch einen einphasigen Aufbau erfolgreich getestet werden. Die Funktionsweise konnte sowohl durch die Hardware-in-the-loop Methode als auch am Hybridschalter selbst, nachgewiesen werden. Für eine dreiphasige Ansteuerung der Hybridschalter ist die entwickelte Hardware bereits ausgelegt und aufgebaut. Eine Validierung, die nur an der Hardware vorgenommen wird, ist für die dreiphasige Ansteuerung nicht zwingend notwendig, da es sich lediglich um eine Erweiterung des einphasigen Aufbaus (mit bereits validierten Komponenten) handelt. Dadurch ist es bereits jetzt schon möglich, das entwickelte System im Leistungsversuchsfeld des AIT zu installieren. Es muss hier allerdings darauf geachtet werden, dass die bereits aufgebaute Ansteuerung mit Rücksicht auf die elektromagnetische Verträglichkeit installiert wird, da die Ansteuerung der Hybridschalter direkt im Prüffeld aufgebaut werden muss. Um diesem Umstand Rechnung zu tragen, muss die Ansteuerung in einem passenden elektromagnetisch geschirmten Gehäuse untergebracht werden. Diese Maßnahme kann allerdings erst nach dem Aufbau des dreiphasigen Hybridschaltersystems erfolgen, um Rücksicht auf die örtlichen Gegebenheiten nehmen zu können.

Bei dem implementierten Konzept handelt sich um eine modellbasierte Steuerung. Für die Validierung des gesamten dreiphasigen Systems ist also noch die bereits entwickelte Software zu testen. Diese Tests können prinzipiell direkt an dem dreiphasigen Hybridschaltersystem gemacht werden, da die entwickelte Ansteuerung (wie oben schon erwähnt) bereits installiert werden kann.

Der letzte Schritt für das gesamte System ist der Aufbau und Anschluss der entwickelten Zusatzbeschaltung für Einschaltvorgänge im Bereich des Spannungsnulldurchgangs. In der Ansteuerung für die Hybridschalter ist eine entsprechende Zündschaltung bereits vorhanden und ebenfalls getestet.

Mit den hier beschriebenen Maßnahmen und aufbauend auf dieser Diplomarbeit wird es möglich sein, die dreiphasige Ansteuerung des vorhanden Hybridschaltersystems in Betrieb zu nehmen.

Quellcode - Mikrocontroller

1 Header und Funktionen

Der folgende Header und die folgenden Funktionen werden für die Hauptprogramme der Synchronsteuerung und der Ansteuerung der Hybridschalter verwendet.

Header

```
1 /*
2   UART-Init:
3   Berechnung des Wertes für das Baudratenregister
4   aus Taktrate und gewünschter Baudrate
5 */
6
7 #ifndef F_CPU
8 /* In neueren Version der WinAVR/Mfile Makefile-Vorlage kann
9   F_CPU im Makefile definiert werden, eine nochmalige Definition
10  hier wuerde zu einer Compilerwarnung fuehren. Daher "Schutz" durch
11  #ifndef/#endif
12
13  Dieser "Schutz" kann zu Debugsessions führen, wenn AVRStudio
14  verwendet wird und dort eine andere, nicht zur Hardware passende
15  Taktrate eingestellt ist: Dann wird die folgende Definition
16  nicht verwendet, sondern stattdessen der Defaultwert (8 MHz?)
17  von AVRStudio – daher Ausgabe einer Warnung falls F_CPU
18  noch nicht definiert: */
19 #warning "F_CPU war noch nicht definiert, wird nun nachgeholt mit 16000000"
20 #define F_CPU 16000000UL // Systemtakt in Hz – Definition als unsigned long beachten
21                          // Ohne ergeben sich unten Fehler in der Berechnung
22 #endif
23
24 #define BAUD 250000UL    // Baudrate
25
26 // Berechnungen
27 #define UBRR_VAL ((F_CPU+BAUD*8)/(BAUD*16)-1)
28 #define BAUD_REAL (F_CPU/(16*(UBRR_VAL+1))) // Reale Baudrate
29 #define BAUD_ERROR ((BAUD_REAL*1000)/BAUD) // Fehler in Promille, 1000 = kein Fehler.
30
31 #if ((BAUD_ERROR<990) || (BAUD_ERROR>1010))
32   #error Systematischer Fehler der Baudrate grösser 1% und damit zu hoch!
33 #endif
34 /*---Konstanten---*/
35 #define LEN 21          // Konstante für die Anzahl der Zeilen in der Wahrheitstabelle
36 #define TIMETO 2       // Konstante als Rechenfaktor zwischen Zeiten und Zählerstand
37 /*---INCLUDE---*/
38 #include <avr/io.h>
39 #include <stdlib.h>
```

```

40 #include <util/delay.h>
41 #include <avr/interrupt.h>
42 #include <stdint.h>
43
44 /*---Globale Variable---*/
45 //volatile uint16_t overflow1;           //Overflow Timer 1
46 volatile uint16_t overflow1;           //Overflow Timer 1
47 volatile uint16_t sw_delay_low[3];     //Messdaten für Verzögerungsmessung
48 volatile uint16_t sw_delay_high[3];   //Messdaten für Verzögerungsmessung
49 volatile uint8_t enable;               //Aktion ermöglichen
50
51 /*---Funktionen UART 0---*/
52 void uart_init(void);
53 int uart_putc(unsigned char);
54 char uart_getc(void);
55 void uart_puts (char*);
56 int uart_gets (char*);
57 void uart_array_send (uint32_t[], uint16_t);
58 void uart_array_get(uint32_t[], uint16_t);
59 void uart_array_send16(uint16_t[], uint16_t);           //Only for debugging
60
61 /*---Funktionen UART 1---*/
62 void uart_init_2(void);
63 void uart_putc_2(unsigned char);
64 unsigned char uart_getc_2(void);
65 void uart_puts_2 (char*);
66 int uart_gets_2 (char*);
67 void uart_array_send_2 (uint32_t[], uint16_t);
68 void uart_array_get_2(uint32_t[], uint16_t);
69
70 /*---Timer---*/
71 void timer_start();
72 void timer_stop();
73 uint32_t get_time(void);
74 /*---Interrupt---*/
75 void interrupt_init(void);

```

Interrupt

```

1 #include "header.h"
2
3 /*Initialisierung des Interrupt*/
4 void interrupt_init(void)
5 {
6     /*Auf Eingang schalten*/
7     DDRD &= ~ (1<<DDD0)|(1<<DDD1)|(1<<DDD2)|(1<<DDD3);
8     DDRE &= ~ (1<<DDE4);
9     /* ACHTUNG! Reihenfolge der Initialisierung unbedingt einhalten ,
10    da es ansonsten zu Fehlern kommen kann*/
11    /*reagiert auf negative Flanke, außer Synchronisation*/
12    EICRA |= (1<<ISC01)|(1<<ISC11)|(1<<ISC21)|(1<<ISC31)|(1<<ISC30);
13    EICRB |= (1<<ISC41);
14    /*INT0 Enable, INT1 Enable, INT2 Enable, INT3 Enable*/
15    EIMSK |= (1<<INT0)|(1<<INT1)|(1<<INT2)|(1<<INT3)|(1<<INT4);
16    /*Flags löschen*/

```

```

17     EIFR |= (1<<INTF0)|(1<<INTF1)|(1<<INTF2)|(1<<INTF3)|(1<<INTF4);
18 }
19
20 ISR(INT0_vect)
21 {
22     sw_delay_low[0] = TCNT1;
23     sw_delay_high[0] = overflow1;
24 }
25
26 ISR(INT1_vect)
27 {
28     sw_delay_low[1] = TCNT1;
29     sw_delay_high[1] = overflow1;
30 }
31
32 ISR(INT2_vect)
33 {
34     sw_delay_low[2] = TCNT1;
35     sw_delay_high[2] = overflow1;
36 }
37
38 ISR(INT3_vect)
39 {
40     if(enable==0)
41     {
42         timer_start();
43         enable=2;
44     }
45 }
46
47 ISR(INT3_vect)
48 {
49     if(enable==0)
50     {
51         timer_start();
52         enable=2;
53     }
54 }
55
56 /*Interrupt für Startknopf*/
57 ISR(INT4_vect)
58 {
59
60 }
61
62 ISR(TIMER1_OVF_vect)
63 {
64     overflow1++;
65 }

```

Timer

```

1 #include "header.h"
2
3 void timer_start()
4 {

```

```

5 TCNT1=0;
6 TCCR1B|= (1<<CS11); //Prescaler 8
7 TIMSK1 |= (1<<TOIE1); //Interrupt Enable für Overflow
8 }
9
10 void timer_stop()
11 {
12 TCNT1=0;
13 TCCR1B&= ~(1<<CS11); //Prescaler 8
14 TIMSK1 &= ~(1<<TOIE1); //Interrupt Enable für Overflow
15 }
16
17 uint32_t get_time(void) // read T as 32 bit timer
18 {
19 uint32_t val;
20 uint32_t tifr1;
21
22 cli();
23 val = overflow1 + TCNT1;
24 tifr1 = TIFR1; // read interrupt flags
25 sei();
26 if((tifr1 & 1<<TOV1) && !(val & 0x8000)) // overflow prior reading TCNT1 ?
27 val += 65536; // then add overflow
28
29 return val;
30 }

```

UART0

```

1 #include "header.h"
2
3 /* Initialisierung UART0 für ATMEGA 2560*/
4 void uart_init(void)
5 {
6 UCSR0B |= (1<<TXEN0); // UART Transmitter einschalten
7 UCSR0B |= (1<<RXEN0); // UART Receiver einschalten
8 UCSR0C |= (1<<UCSZ01)|(1<<UCSZ00); // 8bit 1 Stoppbit
9
10 UBRR0H = UBRR_VAL >> 8; //Oberen 8bit des UART Baud Rate Register
11 UBRR0L = UBRR_VAL & 0xFF; //Unteren 8bit des UART Baud Rate Register
12 }
13
14 /* Senden über UART0*/
15 int uart_putc(unsigned char c)
16 {
17 //UDRE warten bis Senden moeglich (wenn UDRE 1 ist wird die While Schleife verlassen)
18 while (!(UCSR0A & (1<<UDRE0))) /
19 {
20 }
21 UDR0 = c; // sende Zeichen über UART Data Register
22 return 0;
23 }
24
25 /*Empfangen über UART0*/
26 char uart_getc(void)
27 {

```

```

28 // warten bis Zeichen verfuegbar (wenn RXC 1 ist wird die While Schleife verlassen)
29   while (!(UCSR0A & (1<<RXC0)))
30     ;
31   return UDR0; // Zeichen aus UDR an Aufrufer zurueckgeben
32 }
33
34 /*UART0 Zeichenkette schreiben*/
35 void uart_puts (char *s)
36 {
37     int z=0;
38     uart_putc(0x06);
39     /* so lange *s != '\0' also ungleich dem "String-Endezeichen" */
40     do
41     {
42         uart_putc(s[z]);
43         z++;
44     } while(s[z-1]!=0x00);
45 }
46 /*UART0 Zahlen Array schreiben*/
47 void uart_array_send (uint32_t array[], uint16_t len)
48 {
49     uint16_t z=0, i=0;
50     char out[15];
51
52     uart_putc(0x06);
53
54     for(i=0;i<len;i++)
55     {
56         ltoa(array[i],out,10); //ltoa(int, char[], int(base));
57
58         /* so lange *s != '\0' also ungleich dem "String-Endezeichen" */
59         do
60         {
61             uart_putc(out[z]);
62             z++;
63         } while(out[z-1]!=0x00);
64         z=0;
65     }
66     uart_putc(0xA);
67 }
68
69 /*UART0 Zahlen Array schreiben*/
70 void uart_array_send16(uint16_t array[], uint16_t len)
71 {
72     uint16_t z=0, i=0;
73     char out[15];
74
75     uart_putc(0x06);
76
77     for(i=0;i<len;i++)
78     {
79         ltoa(array[i],out,10); //ltoa(int, char[], int(base));
80
81         /* so lange *s != '\0' also ungleich dem "String-Endezeichen" */
82         do
83         {
84             uart_putc(out[z]);
85             z++;
86         } while(out[z-1]!=0x00);

```

```

87         z=0;
88     }
89     uart_putc(0xA);
90 }
91
92 /*UART0 Zeichenkette lesen mit Bestätigung ob Zeichen angekommen ist (0x06)*/
93 int uart_gets (char *r)
94 {
95     int z=0;
96
97     uart_putc(0x06);
98     do
99     {
100         r[z]=uart_getc();
101         z++;
102     } while(r[z-1]!=0x00);
103 return z;
104 }
105
106 /*UART0 Zahlen Array empfangen*/
107 void uart_array_get(uint32_t array[], uint16_t len)
108 {
109     uint16_t z=0,i=0;
110     char in[15];
111
112     uart_putc(0x06);
113     uart_gets(&in[0]);
114     //len=atol(in); //Achtung hier wird len überschrieben!!!
115
116     for(i=0;i<len;i++)
117     {
118         do{
119             in[z]=uart_getc();
120             z++;
121         } while(in[z-1]!=0x00);
122         z=0;
123         array[i]=atol(in);
124     }
125 }

```

UART2

```

1 #include "header.h"
2
3 /* Initialisierung UART2 für ATMEGA 2560*/
4 void uart_init_2(void)
5 {
6     UCSR2B |= (1<<TXEN2); // UART Transmitter einschalten
7     UCSR2B |= (1<<RXEN2); // UART Receiver einschalten
8     UCSR2C |= (1<<UCSZ21)|(1<<UCSZ20); // 8bit 1 Stoppbit
9
10    UBRR1H = UBRR_VAL >> 8; //Oberen 8bit des UART Baud Rate Register
11    UBRR1L = UBRR_VAL & 0xFF; //Unteren 8bit des UART Baud Rate Register
12 }
13
14
15

```

```

16
17 /* Senden über UART2*/
18 void uart_putc_2(unsigned char c)
19 {
20     //UDRE warten bis Senden moeglich (wenn UDRE 1 ist wird die While Schleife verlassen)
21     while (!(UCSR2A & (1<<UDRE2)))
22     {
23     }
24     UDR2 = c; // sende Zeichen über UART Data Register
25     // return 0;
26 }
27
28
29 /*Empfangen über UART2*/
30 unsigned char uart_getc_2(void)
31 {
32     // warten bis Zeichen verfuegbar (wenn RXC 1 ist wird die While Schleife verlassen)
33     while (!(UCSR2A & (1<<RXC2)))
34     ;
35     return UDR2; // Zeichen aus UDR an Aufrufer zurueckgeben
36 }
37
38 /*UART2 Zeichenkette schreiben*/
39 void uart_puts_2 (char *s)
40 {
41     int z=0;
42
43     uart_putc_2(0x06);
44     /* so lange *s != '\0' also ungleich dem "String-Endezeichen" */
45     do
46     {
47         uart_putc_2(s[z]);
48         z++;
49     } while(s[z-1]!=0x00);
50 }
51
52 /*UART2 Zeichenkette lesen mit Bestätigung ob Zeichen angekommen ist (0x06)*/
53 int uart_gets_2 (char *r)
54 {
55     int z=0;
56
57     do
58     {
59         r[z]=uart_getc_2();
60         z++;
61     } while(r[z-1]!=0x00);
62 return z;
63 }
64
65 /*UART2 Zahlen Array schreiben*/
66 void uart_array_send_2(uint32_t array[], uint16_t len)
67 {
68     uint16_t z=0, i=0;
69     char out[15];
70
71     _delay_ms(10);
72     for(i=0;i<len;i++)
73     {
74         ltoa(array[i],out,10); //ltoa(int, char[], int(base));

```

```
75     /* so lange *s != '\0' also ungleich dem "String-Endezeichen" */
76     do
77         {
78             uart_putc_2(out[z]);
79             z++;
80             } while(out[z-1]!=0x00);
81             z=0;
82         }
83     }
84
85     /*UART2 Zahlen Array empfangen*/
86     void uart_array_get_2(uint32_t array[], uint16_t len)
87     {
88         uint16_t z=0,i=0;
89         char in[15];
90
91         // uart_putc_2(0x06);
92         // uart_gets_2(&in[0]);
93         //len=atol(in);           //Achtung hier wird len überschrieben!!!
94
95         for(i=0;i<len;i++)
96             {
97                 do{
98                     in[z]=uart_getc_2();
99                     z++;
100                    } while(in[z-1]!=0x00);
101                    z=0;
102                    array[i]=atol(in);
103                }
104            }
```

2 Synchronsteuerung - Hauptprogramm

```
1 #include "header.h"
2
3 void init()
4 {
5     overflow1=0;
6     enable=0;
7     PORTA|=(1<<PA0)|(1<<PA1)|(1<<PA2)|(1<<PA3)|(1<<PA4)|(1<<PA5);
8     PORTC|=(1<<PC0)|(1<<PC1)|(1<<PC2)|(1<<PC3)|(1<<PC4)|(1<<PC5);
9     PORTL|=(1<<PL0)|(1<<PL1)|(1<<PL2)|(1<<PL3)|(1<<PL4)|(1<<PL5);
10    PORTF=0x00;
11    PORTK=0x00;
12 }
13
14
15 void convert_array(uint32_t a_32[32], volatile uint16_t a_high[32],
16 volatile uint16_t a_low[32], uint16_t length)
17 {
18     uint16_t i=0;
19     uint32_t tmp_32[32]={0};
20
21
22     for(i=0;i<length;i++)
```

```

23     {
24         tmp_32[i]=a_32[i];
25         tmp_32[i]=tmp_32[i]*TIMETO;
26         a_low[i]=(uint16_t)(tmp_32[i]&0xFFFF);
27         a_high[i]=(uint16_t)(tmp_32[i]>>16);
28     }
29 }
30 }
31
32 void convert_a16toa32(uint32_t a_32[LEN],
33 volatile uint16_t a_high[LEN], volatile uint16_t a_low[LEN])
34 {
35     uint16_t i=0;
36     uint32_t tmp_32=0;
37
38
39     for(i=0;i<LEN;i++)
40     {
41         tmp_32=a_high[i];
42         tmp_32=tmp_32<<16;
43         tmp_32+=a_low[i];
44         tmp_32=tmp_32/TIMETO;
45         a_32[i]=tmp_32;
46     }
47 }
48
49 void convert_a32toa8(uint32_t a_32[32], volatile uint8_t a_high[32],
50 volatile uint8_t a_low[32], uint16_t length)
51 {
52     uint16_t i=0;
53     uint32_t tmp_32=0;
54
55     for(i=0;i<length;i++)
56     {
57         tmp_32=a_32[i];
58         a_high[i]=(uint8_t)(tmp_32>>8);
59         a_low[i]=(uint8_t)(tmp_32&0xFF);
60     }
61 }
62
63 /*---MAIN---*/
64 int main (void)
65 {
66
67     char start=0, in[10]={0}, out[10]={0};
68     uint32_t time[LEN]={0}, time_sync[32]={0}, out1[LEN]={0}, out2[LEN]={0},
69     out3[LEN]={0}, out_sync[32]={0}, sw_delay[3]={0};
70     uint16_t schalter=0, time_low[LEN]={0}, time_high[LEN]={0}, i=0, tmp[LEN]={0},
71     tcnt=0,over=0, switches=0, len_syn_array=0;
72     uint8_t out_high[32], out_low[32];
73     DDRB |= (1<<DDB7); //Auf Ausgang schalten
74     DDRA |= (1<<DDA0)|(1<<DDA1)|(1<<DDA2)|(1<<DDA3)|(1<<DDA4)|(1<<DDA5);
75     DDRC |= (1<<DDC0)|(1<<DDC1)|(1<<DDC2)|(1<<DDC3)|(1<<DDC4)|(1<<DDC5);
76     DDRL |= (1<<DDL0)|(1<<DDL1)|(1<<DDL2)|(1<<DDL3)|(1<<DDL4)|(1<<DDL5);
77     DDRA &=~(1<<DDA6);
78     DDRC &=~(1<<DDC6);
79     DDRL &=~(1<<DDL6);
80     DDRF =0xFF;
81     DDRK= 0xFF;

```

```

82
83  /*Initialisierung UART*/
84  uart_init ();
85  uart_init_2 ();
86  interrupt_init ();
87  init ();
88
89  //sei ();
90  /*PROGRAMM*/
91  while (1) {
92
93
94      start=uart_getc ();
95
96      /*PC sendet Time-Array an uC und uC weiter an HS*/
97      if (start==0x01)
98      {
99          uart_array_get(&time[0],LEN);
100         uart_putc_2(0x01);
101         uart_array_send_2(&time[0],LEN);
102     }
103
104     /*uC holt Daten von HS sendet Time-Array an PC*/
105     else if (start==0x02)
106     {
107         uart_putc_2(0x02);
108         uart_array_get_2(&time[0], LEN);
109         uart_array_send(&time[0],LEN);
110     }
111     /*PC sendet Out-Array 1 an uC und uC weiter an HS*/
112     else if (start==0x03)
113     {
114         uart_array_get(&out1[0],LEN);
115         uart_putc_2(0x03);
116         uart_array_send_2(&out1[0],LEN);
117     }
118
119     /*uC sendet Out-Array 1 an PC*/
120     else if (start==0x04)
121     {
122         uart_putc_2(0x04);
123         uart_array_get_2(&out1[0], LEN);
124         uart_array_send(&out1[0],LEN);
125     }
126     /*PC sendet Out-Array 2 und uC und uC weiter an HS */
127     else if (start==0x05)
128     {
129         uart_array_get(&out2[0],LEN);
130         uart_putc_2(0x05);
131         uart_array_send_2(&out2[0],LEN);
132     }
133
134     /*uC sendet Out-Array 2 an PC*/
135     else if (start==0x06)
136     {
137         uart_putc_2(0x06);
138         uart_array_get_2(&out2[0], LEN);
139         uart_array_send(&out2[0],LEN);
140     }
141     /*PC sendet Out-Array 3 an uC und uC weiter an HS*/

```

```

141     else if (start==0x07)
142     {
143         uart_array_get(&out3[0],LEN);
144         uart_putc_2(0x07);
145         uart_array_send_2(&out3[0],LEN);
146     }
147
148     /*uC sendet Out-Array 3 an PC*/
149     else if (start==0x08)
150     {
151         uart_putc_2(0x08);
152         uart_array_get_2(&out3[0], LEN);
153         uart_array_send(&out3[0],LEN);
154     }
155     /*PC sendet Länge von Sync Array an uC*/
156     else if (start==0x12)
157     {
158         uart_gets(&in[0]);
159         len_syn_array=atol(in);
160     }
161     /*PC sendet Time-Array Sync an uC*/
162     else if (start==0x09)
163     {
164         uart_array_get(&time_sync[0],len_syn_array);
165     }
166
167     /*uC sendet Time-Array Sync an PC*/
168     else if (start==0x0A)
169     {
170         uart_array_send(&time_sync[0],len_syn_array);
171     }
172     /*PC sendet Out-Array Sync an uC*/
173     else if (start==0x0B)
174     {
175         uart_array_get(&out_sync[0],len_syn_array);
176     }
177
178     /*uC sendet Out-Array Sync an PC*/
179     else if (start==0x0C)
180     {
181         uart_array_send(&out_sync[0],len_syn_array);
182     }
183
184     /*Mechanischen Schalter schließen*/
185     else if (start==0x0D)
186     {
187         uart_gets(&in[0]);
188         uart_putc_2(0x0D);
189         uart_puts_2(&in[0]);
190     }
191
192
193     /*Mechanischen Schalter Status?*/
194     else if (start==0x0E)
195     {
196         uart_putc_2(0x0E);
197         uart_gets_2(out);
198         uart_puts(&out[0]);
199     }

```

```

200
201
202     /*Aktion ausführen*/
203     else if (start==0x0F)
204     {
205
206         /*Vorbereiten*/
207         convert_array(&time_sync[0], &time_high[0], &time_low[0], len_syn_array);
208         uart_array_send16(&time_low[0], len_syn_array);
209         convert_a32toa8(&out_sync[0], &out_high[0], &out_low[0], len_syn_array);
210         timer_stop(); //Timer stoppen falls aktiv durch mechanischen Schalter
211         init();
212         sei();
213         i=0;
214         /*Ausführen*/
215         while(i<len_syn_array)
216         {
217
218             if(enable==2) //Hybridschalter starten
219             {
220                 uart_putc_2(0x0F);
221                 enable=3;
222             }
223             //Wichtig Kompilierung auf O3 stellen,
224             //dann ist es auf Geschwindigkeit optimiert
225             tcnt=TCNT1;
226             over=overflow1;
227
228             if (tcnt>0xFFFFE)
229             {
230                 tcnt=TCNT1;
231                 over=overflow1;
232             }
233
234             if ((( tcnt>=time_low[i])&&(over>=time_high[i])) || (over>time_high[i]))
235             {
236                 PORTK=out_high[i]; //1..8 Kanal
237                 PORTF=out_low[i]; //9..16 Kanal
238                 i++;
239             }
240
241             }
242             timer_stop();
243             init();
244         }
245         else if (start==0x11)
246         {
247             uart_putc_2(0x11);
248             uart_array_get_2(&sw_delay[0], 3);
249             uart_array_send(&sw_delay[0], 3);
250         }
251         else
252         {
253             uart_putc('X');
254         }
255     }
256 }

```

3 Ansteuerung der Hybridschalter - Hauptprogramm

```

1 #include "header.h"
2
3 void init()
4 {
5     overflow1=0;
6     enable=0;
7     PORTA|=(1<<PA0)|(1<<PA1)|(1<<PA2)|(1<<PA3)|(1<<PA4)|(1<<PA5);
8     PORTC|=(1<<PC0)|(1<<PC1)|(1<<PC2)|(1<<PC3)|(1<<PC4)|(1<<PC5);
9     PORTL|=(1<<PL0)|(1<<PL1)|(1<<PL2)|(1<<PL3)|(1<<PL4)|(1<<PL5);
10 }
11
12
13 void convert_array(uint32_t a_32[LEN],
14 uint16_t a_high[LEN], uint16_t a_low[LEN])
15 {
16     uint16_t i=0;
17     uint32_t tmp_32[LEN]={0};
18
19
20     for(i=0;i<LEN;i++)
21     {
22         tmp_32[i]=a_32[i];
23         tmp_32[i]=tmp_32[i]*TIMETO;
24         a_high[i]=(uint16_t)(tmp_32[i]>>16);
25         a_low[i]=(uint16_t)(tmp_32[i]&0xFFFF);
26     }
27 }
28
29 void convert_a16toa32(uint32_t a_32[LEN],
30 volatile uint16_t a_high[LEN], volatile uint16_t a_low[LEN])
31 {
32     uint16_t i=0;
33     uint32_t tmp_32=0;
34
35
36     for(i=0;i<LEN;i++)
37     {
38         tmp_32=a_high[i];
39         tmp_32=tmp_32<<16;
40         tmp_32+=a_low[i];
41         tmp_32=tmp_32/TIMETO;
42         a_32[i]=tmp_32;
43     }
44 }
45
46 /*---MAIN---*/
47 int main (void)
48 {
49
50     char start=0, in[10]={0}, out[10]={0};
51     uint32_t time[LEN]={0}, out1[LEN]={0},
52         out2[LEN]={0}, out3[LEN]={0}, sw_delay[3]={0};
53     uint16_t schalter=0, time_low[LEN]={0},
54         time_high[LEN]={0}, i=0, tmp[LEN]={0}, tcnt=0,over=0, switches=0;
55     DDRB |= (1<<DDB7); //Auf Ausgang schalten

```

```

56 DDRA |= (1<<DDA0)|(1<<DDA1)|(1<<DDA2)|(1<<DDA3)|(1<<DDA4)|(1<<DDA5);
57 DDRC |= (1<<DDC0)|(1<<DDC1)|(1<<DDC2)|(1<<DDC3)|(1<<DDC4)|(1<<DDC5);
58 DDRL |= (1<<DDL0)|(1<<DDL1)|(1<<DDL2)|(1<<DDL3)|(1<<DDL4)|(1<<DDL5);
59 DDRA &=~(1<<DDA6);
60 DDRC &=~(1<<DDC6);
61 DDRL &=~(1<<DDL6);
62
63 /*Initialisierung UART*/
64 uart_init();
65 uart_init_2();
66 interrupt_init();
67 init();
68 //sei();
69 /*PROGRAMM*/
70 while (1) {
71
72
73     start=uart_getc_2();
74
75     /*uC sendet Time-Array an HS*/
76     if (start==0x01)
77     {
78         //PORTB^=(1<<PB7);
79         uart_array_get_2(&time[0],LEN);
80         convert_array(&time[0], &time_high[0], &time_low[0]);
81     }
82
83     /*HS sendet Time-Array an uC*/
84     else if (start==0x02)
85     {
86         uart_array_send_2(&time[0],LEN);
87     }
88
89     /*uC sendet Out-Array 1 an HS*/
90     else if (start==0x03)
91     {
92         uart_array_get_2(&out1[0],LEN);
93     }
94
95     /*HS sendet Out-Array 1 an uC*/
96     else if (start==0x04)
97     {
98         uart_array_send_2(&out1[0],LEN);
99     }
100    /*uC sendet Out-Array 2 an HS*/
101    else if (start==0x05)
102    {
103        uart_array_get_2(&out2[0],LEN);
104    }
105
106    /*HS sendet Out-Array 2 an uC*/
107    else if (start==0x06)
108    {
109        uart_array_send_2(&out2[0],LEN);
110    }
111    /*uC sendet Out-Array 3 an HS*/
112    else if (start==0x07)
113    {
114        uart_array_get_2(&out3[0],LEN);

```

3 Ansteuerung der Hybridschalter - Hauptprogramm

```
115     }
116
117     /*HS sendet Out-Array 3 an uC*/
118     else if (start==0x08)
119     {
120         uart_array_send_2(&out3[0],LEN);
121     }
122
123     /*Mechanischen Schalter schließen*/
124     else if (start==0x0D)
125     {
126         //timer_start();
127
128         uart_gets_2(&in[0]);
129         schalter=atol(in);
130         if (schalter==0)
131             PORTA=0xFE;
132         else if (schalter==1)
133             PORTC=0xFE;
134         else if (schalter==2)
135             PORTL=0xFE;
136     }
137
138     /*Mechanischen Schalter Status?*/
139     else if (start==0x0E)
140     {
141         if (!(PIND & (1<<PIND0)))
142             switches |=1<<0;
143
144         if (!(PIND & (1<<PIND1)))
145             switches |=1<<1;
146
147         if (!(PIND & (1<<PIND2)))
148             switches |=1<<2;
149
150         ltoa (switches ,out ,10);
151         uart_puts_2(out);
152     }
153
154     /*Aktion ausführen*/
155     else if (start==0x0F)
156     {
157
158         /*Vorbereiten*/
159         timer_start();
160         //uart_array_send16(&time_low[0],LEN);
161         //Timer stoppen falls aktiv durch mechanischen Schalter
162         timer_stop();
163         init();
164         //sei();
165         i=0;
166         /*Ausführen*/
167         while (i<LEN)
168         {
169 //Wichtig Kompilierung auf O3 stellen, dann ist es auf Geschwindigkeit optimiert
170             tcnt=TCNT1;
171             over=overflow1;
172
173             if (tcnt>0xFFFFE)
```

```

174         {
175             tcnt=TCNT1;
176             over=overflow1;
177         }
178
179 if (((tcnt>=time_low[i])&&(over>=time_high[i])) || (over>time_high[i]))
180     {
181         //Abfrage ob mech. Schalter ausgelöst hat
182 if (((i<3) || (PINA & (1<<PINA6)) || (PINC & (1<<PINC6)) || (PINL & (1<<PINL6))))
183     {
184         PORTA=out1[i];
185         PORTC=out2[i];
186         PORTL=out3[i];
187     }
188     i++;
189
190     }
191 }
192 timer_stop();
193 init();
194 }
195 else if (start==0x11)
196 {
197 convert_a16toa32(&sw_delay[0], &sw_delay_high[0], &sw_delay_low[0]);
198 uart_array_send_2(&sw_delay[0],3);
199 }
200
201 else
202 {
203 uart_putc('X');
204 }
205 }
206 }

```

Schematische Pläne

4 Übersicht

- Schematischer Plan - Controller
- Schematischer Plan - Ansteuerung Hybridschalter
- Schematischer Plan - Ansteuerung freie Kanäle

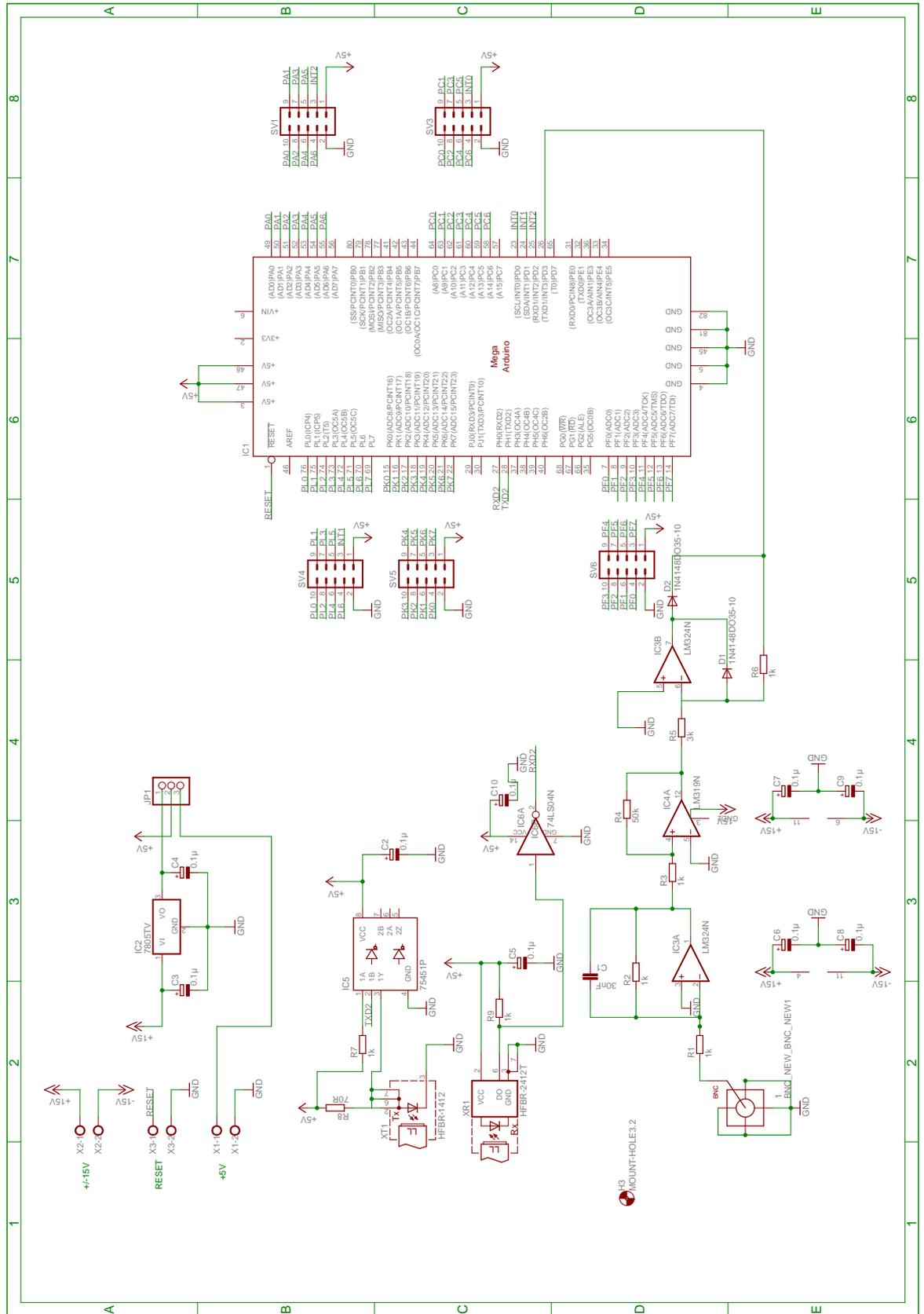


Abbildung 1: Schematischer Plan - Controller

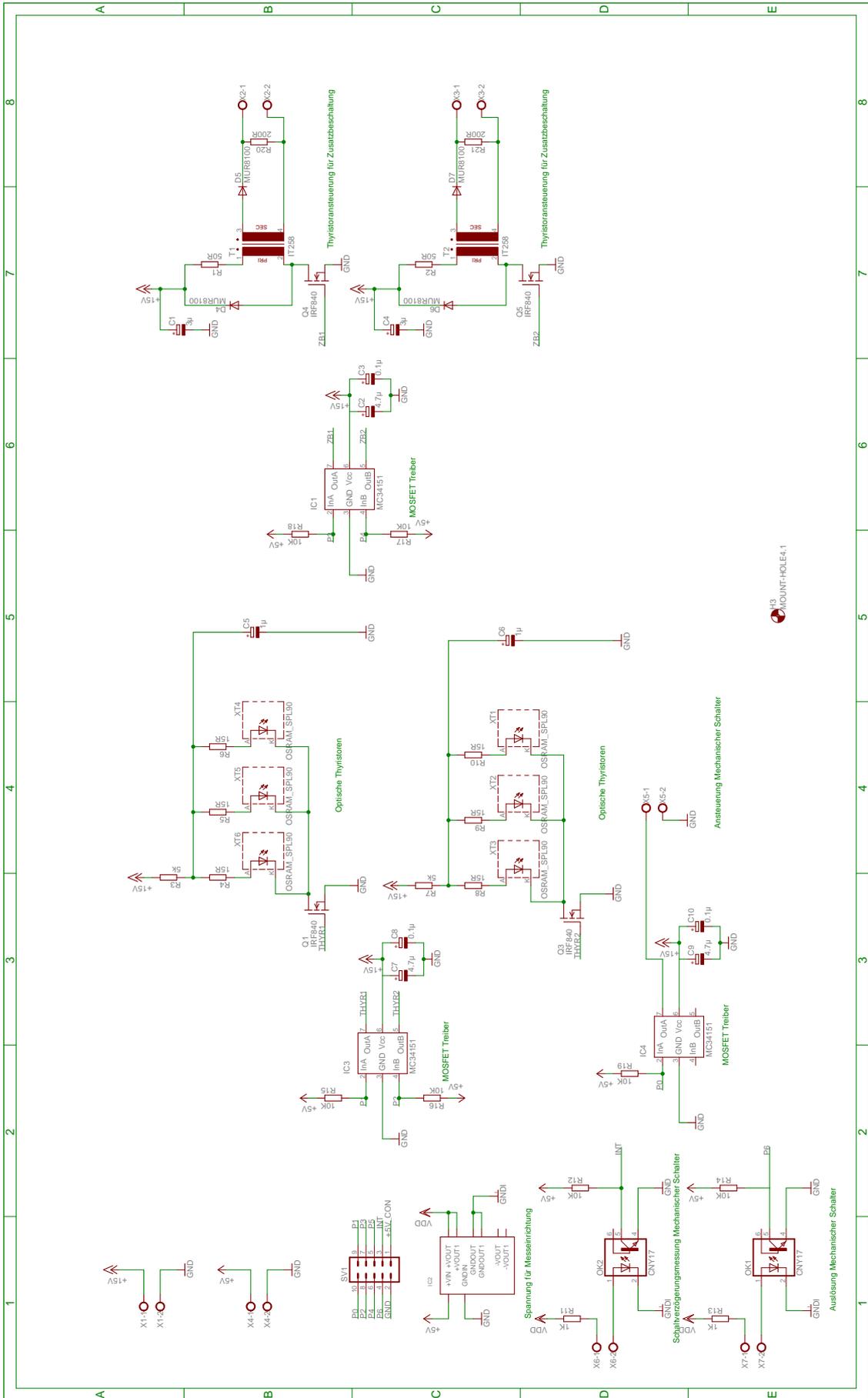


Abbildung 2: Schematischer Plan - Ansteuerung

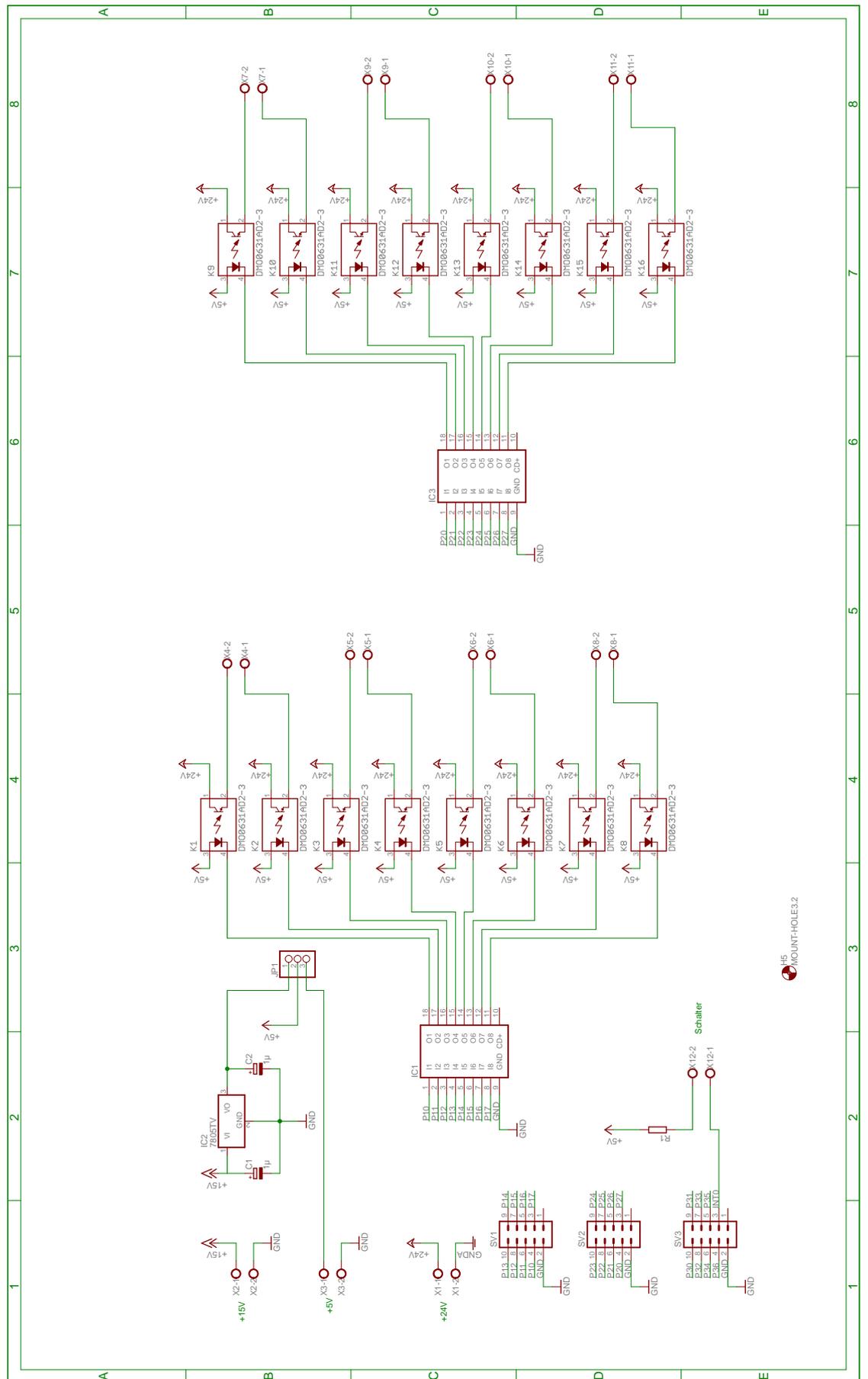


Abbildung 3: Schematischer Plan - Synchronsteuerung

Layout und Fertigung

5 Übersicht

- Layout - Controller
- Layout - Ansteuerung Hybridschalter
- Layout - Ansteuerung freie Kanäle
- Bestückungsplan - Controller
- Bestückungsplan - Ansteuerung Hybridschalter
- Bestückungsplan - Ansteuerung freie Kanäle
- Platine - Controller
- Platine - Ansteuerung Hybridschalter
- Platine - Ansteuerung freie Kanäle
- Synchronsteuerung
- Ansteuerung der Hybridschalter

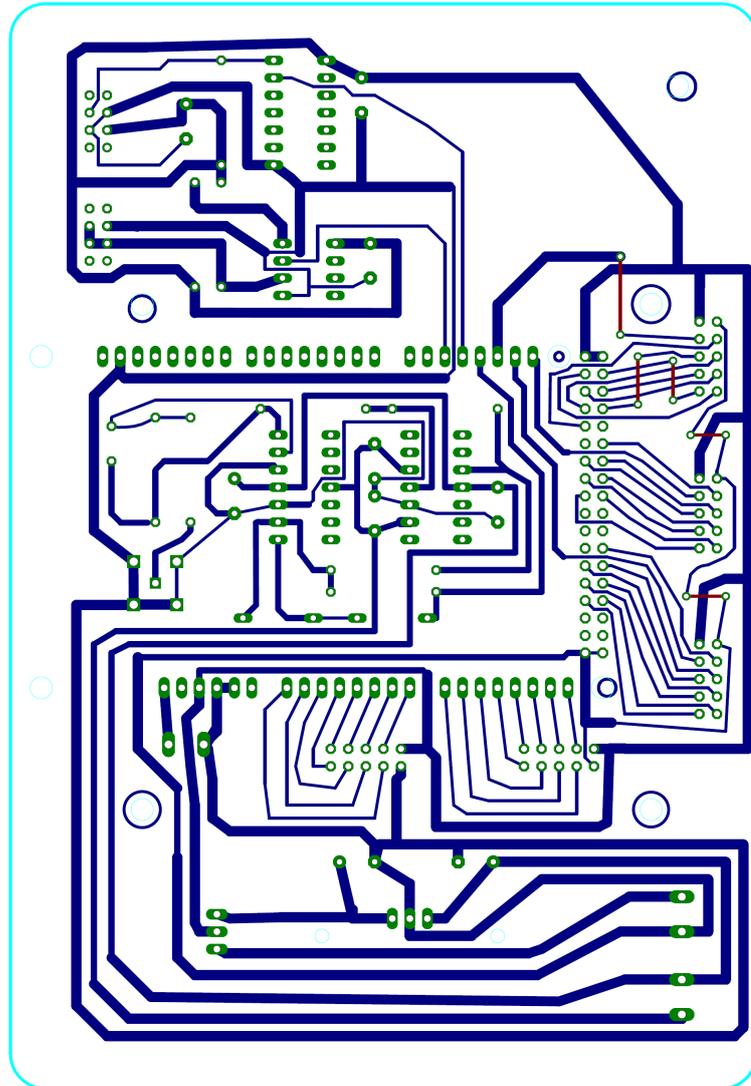


Abbildung 4: Layout - Controller

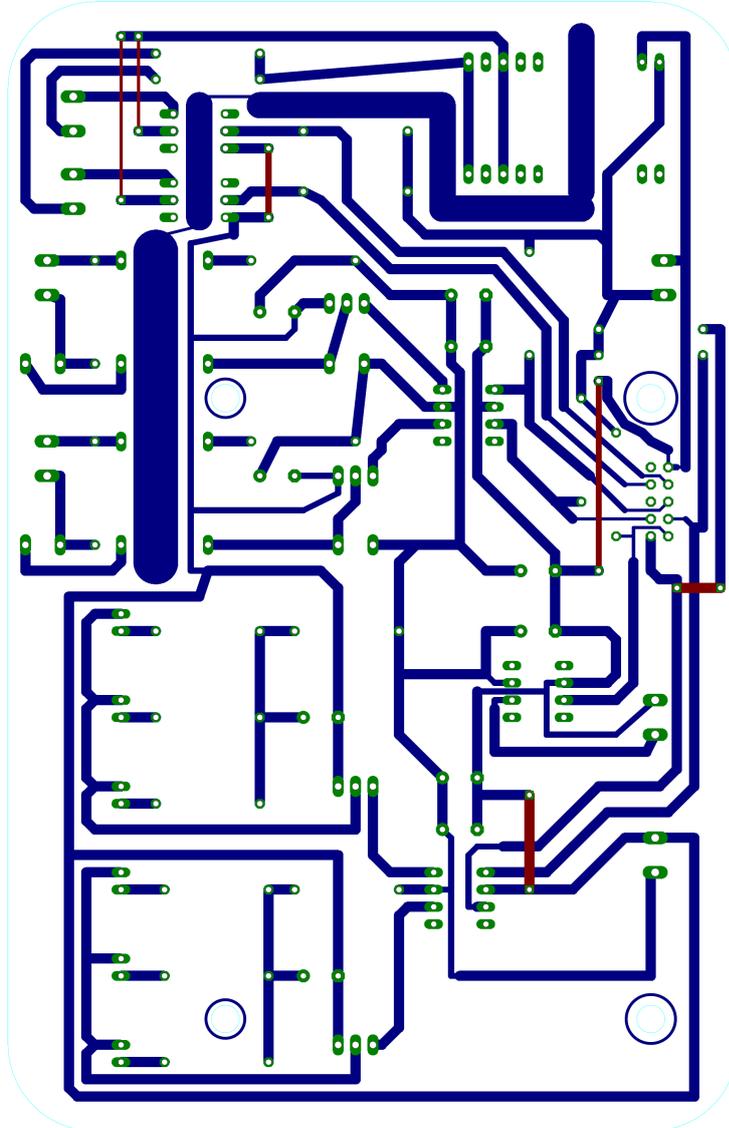


Abbildung 5: Layout - Ansteuerung für einen Hybridschalter

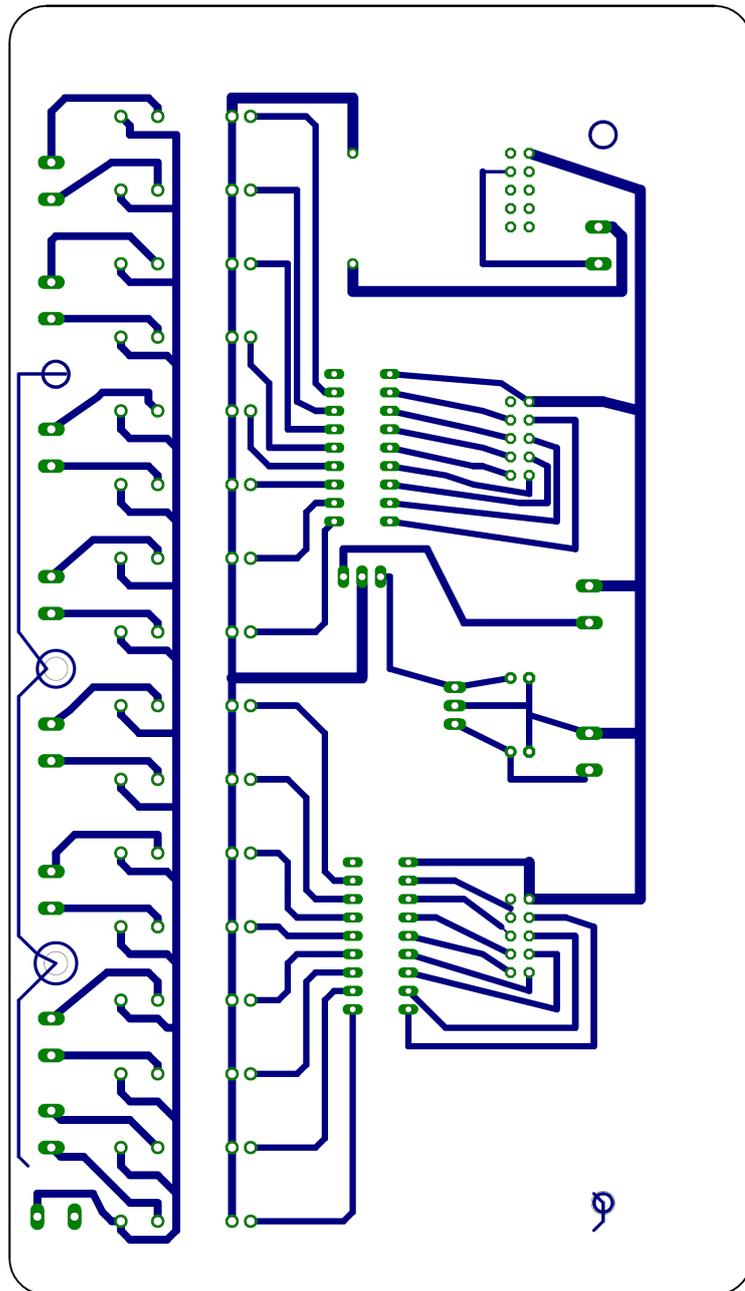


Abbildung 6: Layout - Ansteuerung der freien Kanäle

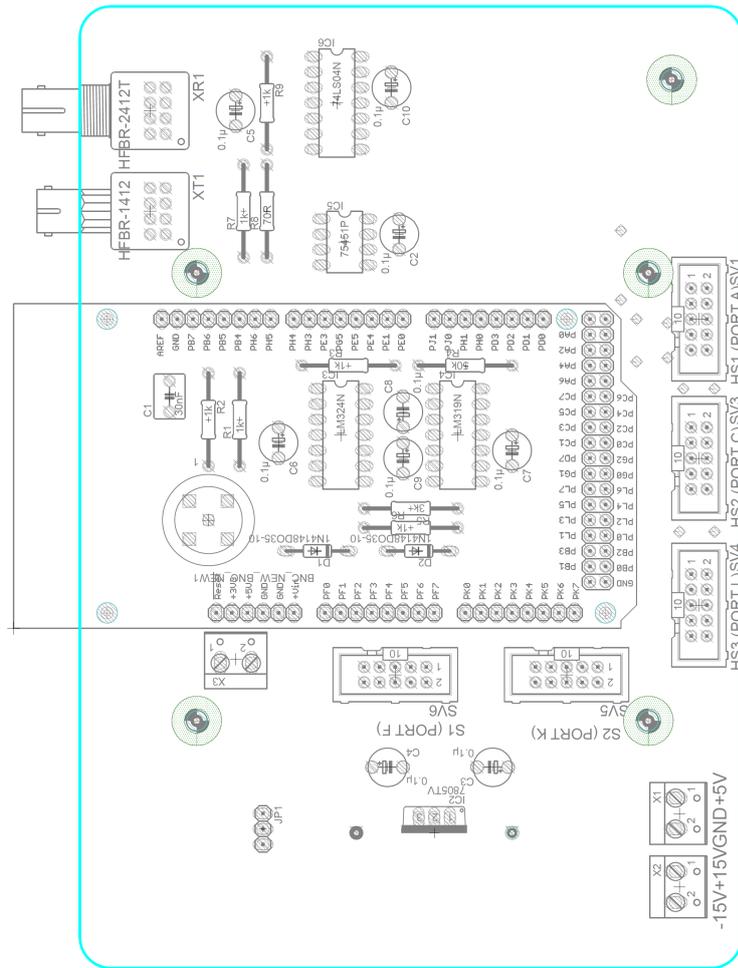


Abbildung 7: Bestückungsplan - Controller

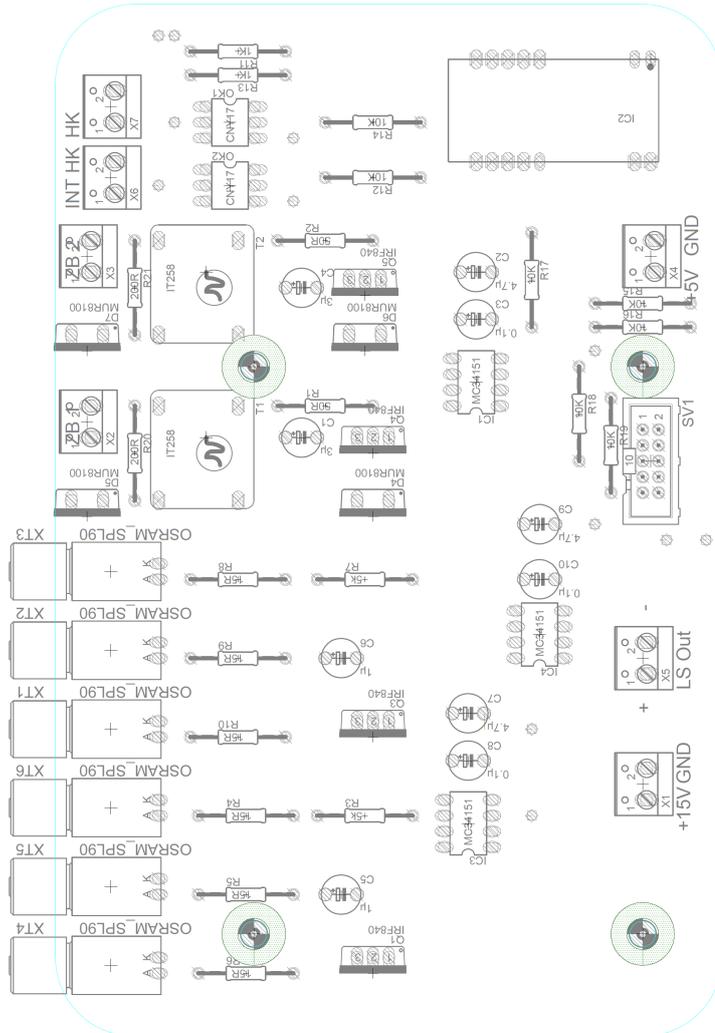


Abbildung 8: Bestückungsplan - Ansteuerung für einen Hybridschalter

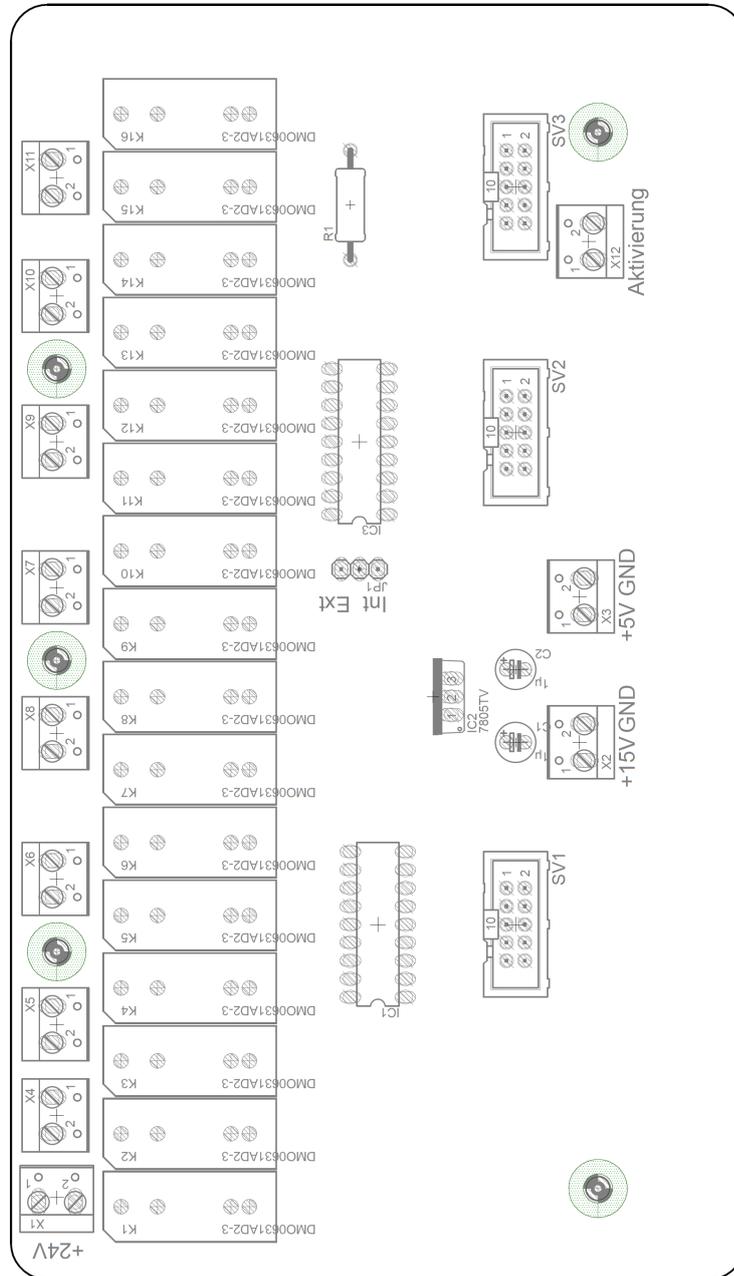


Abbildung 9: Bestückungsplan - Ansteuerung der freien Kanäle

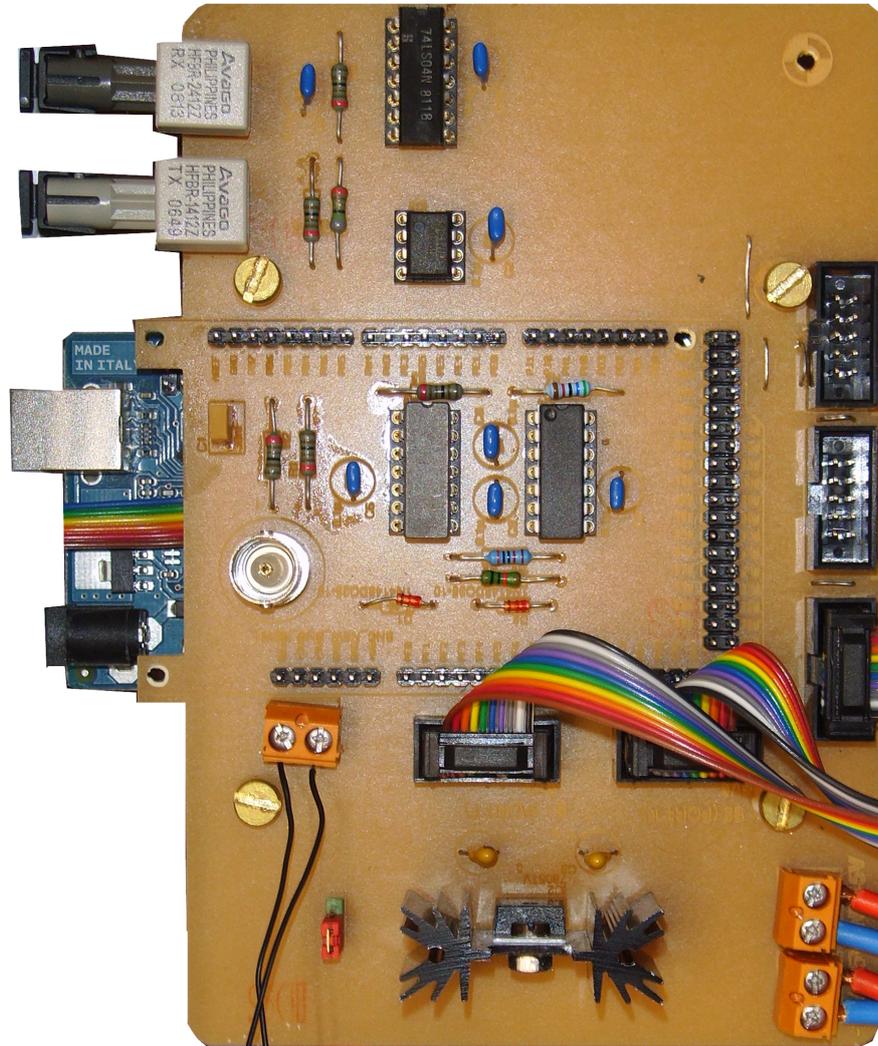


Abbildung 10: Platine - Controller

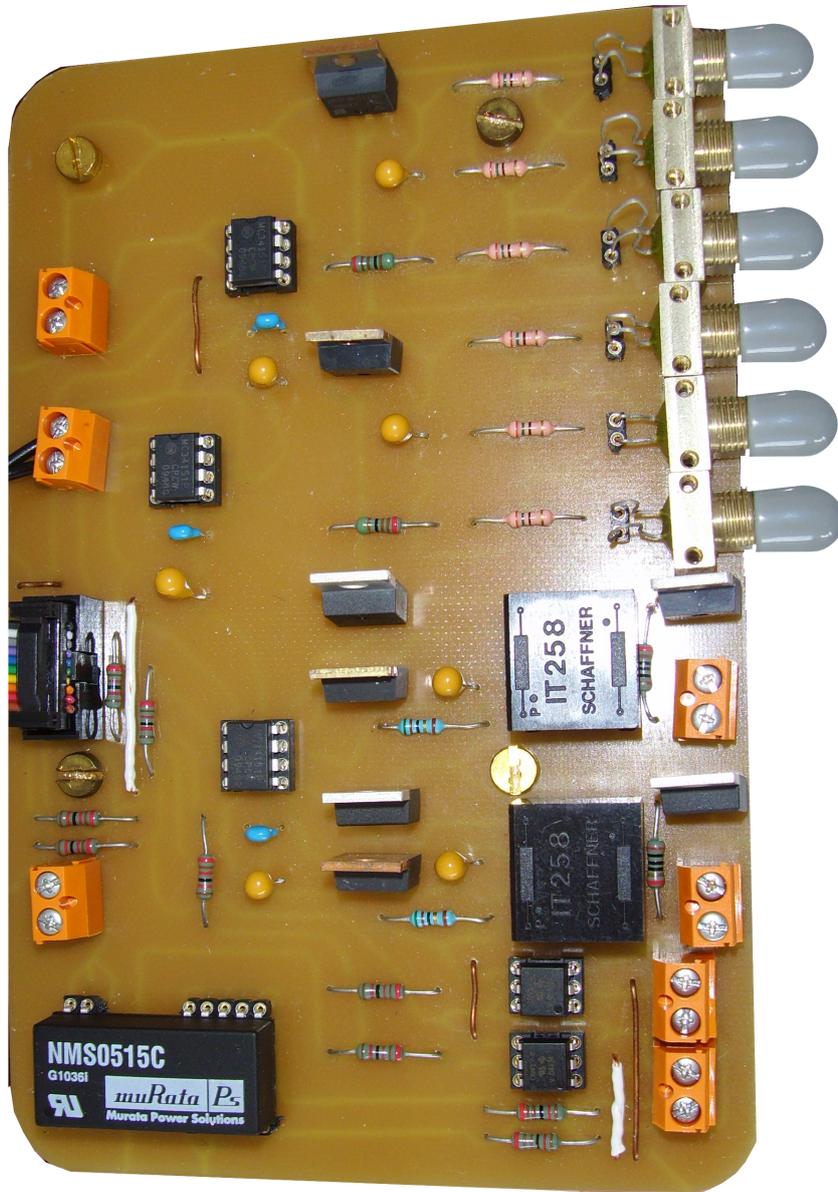


Abbildung 11: Platine - Ansteuerung für einen Hybridschalter

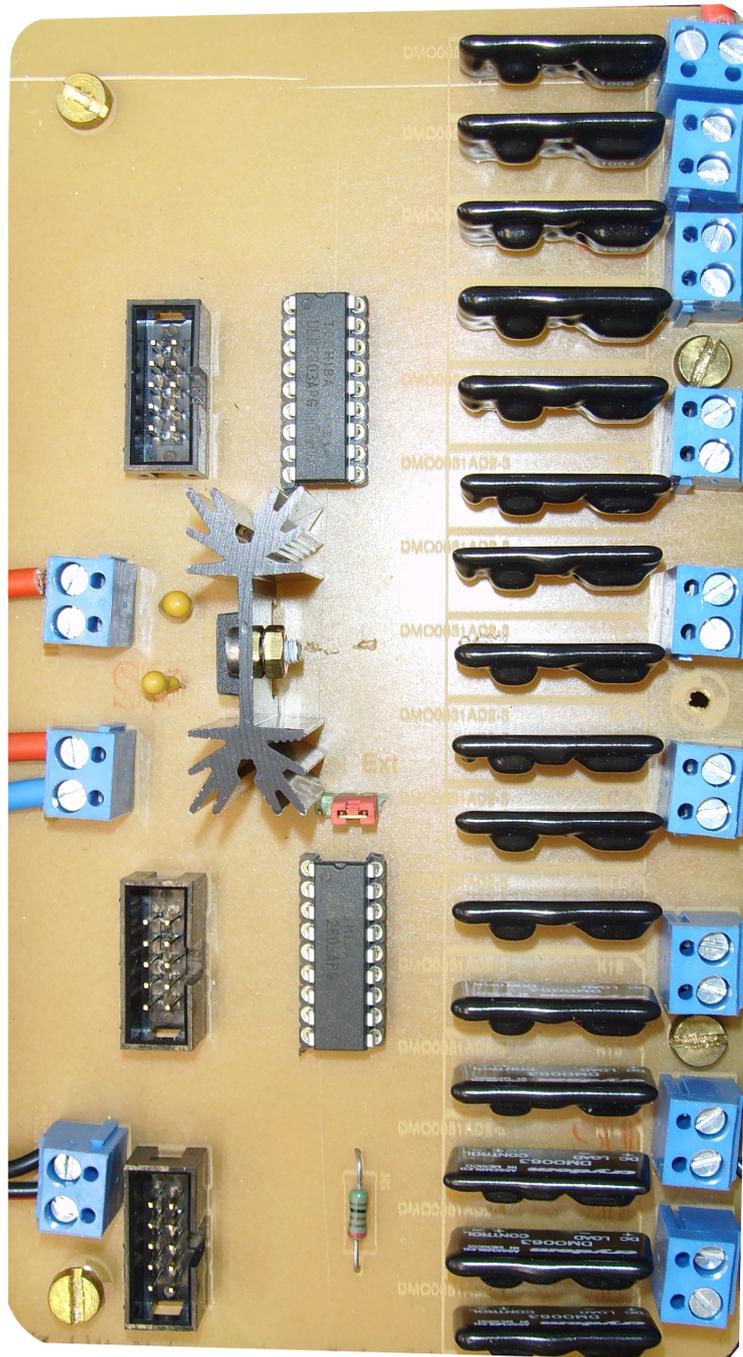


Abbildung 12: Platine - Ansteuerung der freien Kanäle

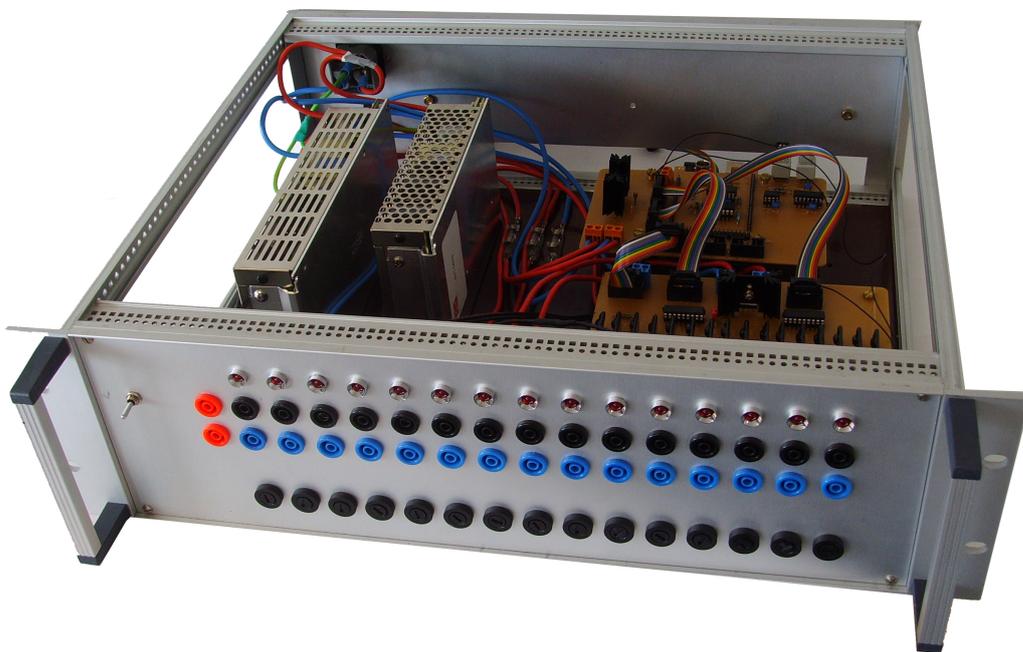


Abbildung 13: Aufbau Synchronsteuerung

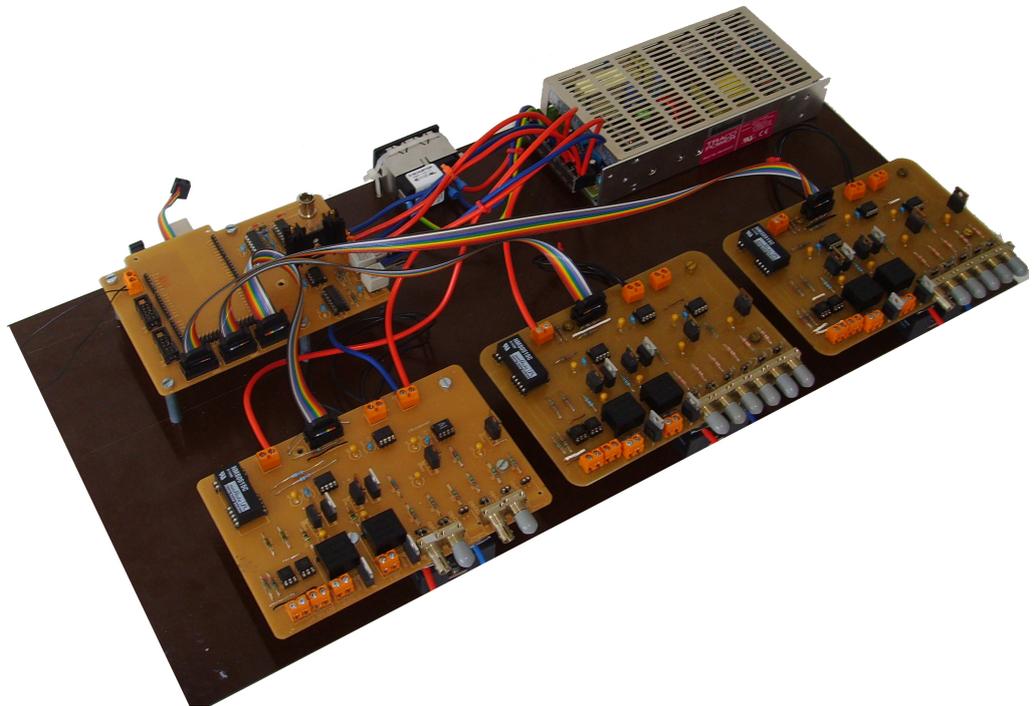


Abbildung 14: Aufbau Ansteuerung dreiphasiger Hybridschalter

Literaturverzeichnis

- [1] VASKO+PARTNER INGENIEURE ; ARSENAL RESEARCH: *Funktionale Beschreibung des Leistungsversuchsfeldes*. Wien, 2004
- [2] LECHNER, T. ; BRAUNER, G. ; HIMMELSTOSS, F.A.: Hybrid Switching System for Test Applications. In: *OPTIM 2002 1* (2002), Nr. T22
- [3] FENZ, Claus J.: *Diplomarbeit - Hybrides synchrones Hochstromeinschaltssystem*. Wien : Technikum Wien, 2002
- [4] SCHNEIDER-ELECTRIC: *Masterpact NW20 HF*. Datasheet, 2009. – Masterpact NF, NW Datasheet
- [5] WIKIPEDIA: *Grenzlastintegral*. Website, 2011. – Online verfügbar unter der Adresse: <http://de.wikipedia.org/wiki/Grenzlastintegral>, besucht am 02.Februar 2011
- [6] EUPEC: *EUPEC-T4003N Datasheet*. Datasheet, 2003. – EUPEC-T4003N Datasheet
- [7] FENZ, Claus J. ; BRAUNER, Georg ; KRAL, Christian ; HIMMELSTOSS, Felix: Hybrid Switching of High Currents in the Vicinity of the Zero Crossing of the Supply Voltage. In: *Proceedings of the 9th international conference on optimization of electrical an electronic equipments 1* (2004), S. 199–204
- [8] OSRAM: *Impuls-Laserdiode SPL PL90*. Datasheet, 2009. – Osram Impuls-Laserdiode Datasheet
- [9] MICHEL, Manfred: *Leistungselektronik: Einführung in Schaltungen und deren Verhalten*. Berlin : Springer Verlag, Januar 2011
- [10] ARDUINO: *Arduino Mega2560*. Website, 2011. – Online verfügbar <http://arduino.cc/en/Main/ArduinoBoardMega2560>; besucht am 06.Juli 2011
- [11] TIETZE, Ulrich ; SCHENK, Christoph: *Halbleiter-Schaltungstechnik*. Berlin : Springer Verlag, 2002
- [12] BÖHMER, Erwin ; EHRHARDT, Dietmar ; OBERSCHELP, Wolfgang: *Elemente der angewandten Elektronik*. Siegen : Vieweg, 2010
- [13] SEMICONDUCTOR, National: *LM319 High Speed Dual Comparator*. Datasheet, 2000. – LM319 High Speed Dual Comparator Datasheet
- [14] AGILENT: *Agilent HFBR-0400, HFBR-14xx and HFBR-24xx Series Low Cost, Miniature Fiber Optic Components with ST, SMA, SC and FC Ports*. Datasheet, 2000. – HFBR-1412 and HFBR-2412 Datasheet
- [15] CRYDOM: *400V Series 1- DC - Panel Mount SSRs*. Datasheet, 2010. – Crydom SSR D4D07 Datasheet
- [16] WESTCODE: *N1718NS120 Phase Control Thyristor*. Datasheet, 2001. – Westcode Phase Control Thyristor
- [17] ERTL, Johann: *Leistungselektronik*. Wien : Skriptum-TU Wien, Oktober 2009
- [18] CRYDOM: *24V Series DO SSRs*. Datasheet, 2010. – Crydom SSR DMO063 Datasheet

- [19] WIKIPEDIA: *Definition - Echtzeit*. Website, 2011. – Online verfügbar unter der Adresse: <http://de.wikipedia.org/wiki/Echtzeit>, besucht am 21.Juni 2011
- [20] HAO, H. ; GUOQING, X ; YANG, Z.: Hardware-in-the-loop simulation of electric vehicle powertrain system. In: *APPEEC 1* (2009), Nr. 1, S. 1–5
- [21] LU, Bin ; WU, Xin ; FIGUEROA, H. ; MONTI, A.: A Low-Cost Real-Time Hardware-in-the-Loop Testing. In: *Industrial Electronics, IEEE Transactions* 54 (2007), Nr. 2, S. 919–931
- [22] RT, Opal: *Opal RT Technologies*. Website, 2011. – Online verfügbar <http://www.opal-rt.com/>, besucht im Juni 2011
- [23] JONKE, P. ; ANDREN, F. ; BRAUNER, G. ; HRIBERNIK, W. ; ERTL, H.: Simulation-based development of a three-phase hybrid switchgear for high-current testing laboratories. In: *Cigre SC Ar Technical Colloquium* (2011)