Masterarbeit

Segmentierte Impedanz Messung an einer PEM Brennstoffzelle

Reinhard Klambauer

19.09.2014

Institut für Elektronik
Technische Universität Graz
A-8010 Graz

Betreuer: Ass. Prof. Dipl.-Ing. Dr. Bernd Eichberger

EIDESSTATTLICHE ERKLÄRUNG

AFFIDAVIT

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis.

Datum/Date	Unterschrift/Signeture

Zusammenfassung

Diese Masterarbeit befasst sich mit der Entwicklung eines Messsystems, welches in der Lage ist eine temperaturkorrigierte Impedanz Messung parallel an einer Vielzahl von Kanälen in einer PEM Brennstoffzelle durchzuführen. Diese Arbeit ist Teil des Projektes A3 Falcon das von der österreichischen Forschungsförderungsgesellschaft unterstützt wird.

Für die Messungen wird mit Hilfe eines Leistungsverstärkers ein Wechselstrom in die Brennstoffzelle eingeprägt. Zur Ansteuerung des Verstärkers ist die Implementierung eines Generators für sinusförmige Signale notwendig.

Mit diesem System soll die Impedanz beziehungsweise die Stromdichte zweidimensional im Querschnitt einer Brennstoffzelle bestimmt werden. Eine solche Messung kann zur Verifizierung von Simulationsmodellen der Zelle herangezogen werden. Im Betrieb der Brennstoffzelle kann eine genaue Aussage über ihren aktuellen Zustand („State of Health“) getroffen werden.

Die Arbeit umfasst die Definition eines Systemkonzeptes, die Entwicklung eines Schaltplanes, das Erstellen eines Platinenlayoutes, die Implementierung eines FPGA-Designs sowie die Programmierung der Software für ein embedded Linux System. Abschließend wurde ein funktionsfähiger Prototyp aufgebaut und getestet.

Abstract

This master thesis deals with the development of a measurement system capable of performing a temperature corrected impedance measurement simultaneously on multiple channels in a fuel cell. It is part of the project A3 Falcon which is supported by the Austrian “Forschungsförderungsgesellschaft (FFG)”.

For these measurements it is necessary to excite the fuell cell with an AC current produced by a power amplifier. This amplifier is controlled by a generator for sinusoidal signals.

This system is able to measure the impedance and the current density in the two dimensional cross section of a fuel cell. This measurement can be used to verify the correctness of simulation models for the cell. During the operation of the fuel cell it is possible to accurately determine its “state of health”.

This work involved the definition of a system concept, the development of schematics, the design of a PCB layout and the implementation of an FPGA design as well as the programming of an application software for an embedded Linux module. Finally a prototype of this system was assembled and tested.

Danksagung

Ich möchte mich bei allen Projektpartnern für die Möglichkeit bedanken meine Masterarbeit im Zuge dieses A3 Falcon Projektes durchführen zu können.

Des Weiteren möchte ich mich bei jedem bedanken der mich moralisch wie auch fachlich bei dieser Arbeit unterstützt hat. Mein besonderer Dank gilt dabei meinem Betreuer Dr. Bernd Eichberger für seinen Rat und für seine Unterstützung.

Ebenfalls möchte ich mich bei meinen Kollegen am Institut für Elektronik und allen Freunden bedanken, unter ihnen vor allem bei Dipl.-Ing. Christoph Adelmann, Dipl.-Ing. Harald Axmann und David Gmeindl, BSc, die bei Problemen immer mit einem offenen Ohr zur Verfügung standen.

Meiner Familie gebührt ein ganz besonderer Dank, meiner Mutter Maria Klambauer, meinem Vater Vinzenz Klambauer† sowie meiner Schwester Martina Jakubetz, dafür dass ich immer auf ihre Unterstützung zählen konnte und weiterhin kann.

Zu guter Letzt geht ein Dankeschön an all meine Verwandten, unter ihnen auch mein Onkel Josef Winkler. Sie haben es mir, vor allem in letzter Zeit, ermöglicht mich ganz auf mein Studium und diese Arbeit zu konzentrieren.

Vorwort

Der kommerzielle Einsatz von PEM Brennstoffzellen in Serienfahrzeugen ist an zwei Bedingungen geknüpft. Zum einen der Senkung der Herstellungskosten und zum anderen der kontinuierlichen Steigerung von Effizienz und Lebensdauer.

Zur Beeinflussung dieser Faktoren ist das grundlegende Verständnis sowie die Vorhersagbarkeit des Verhaltens einer PEM Brennstoffzelle unter kritischen Betriebszuständen unabdingbar.

Das von der Forschungsförderungsgesellschaft unterstützte Projekt A3 Falcon beschäftigt sich mit dieser Thematik. Die internationalen Projektpartner sind in alphabetischer Reihenfolge die Firmen AVL, Intelligent Energy, S++ sowie die Institute für Chemische Verfahrenstechnik und Umwelttechnik, für Elektronik, und für Physikalische und Theoretische Chemie der Technischen Universität Graz.

Ziel des A3 Falcon Projektes ist neben der Entwicklung eines zweidimensionalen Degradationsmodels für kritische Zustände die experimentelle Erfassung der flächigen Verteilung der Impedanz in Brennstoffzellen. Mit Hilfe der gewonnen experimentellen Daten werden die Vorhersagen der Simulationsmodelle überprüft.

Damit es möglich ist die Impedanz experimentell zu erfassen sind eine spezialisierte Messtechnik und die entsprechende Weiterverarbeitung der gewonnen Messwerte notwendig. Diese Messtechnik muss im Gegensatz zu den gegenwärtig am Markt verfügbaren Lösungen in der Lage sein eine Impedanz Messung an einer Vielzahl von Kanälen gleichzeitig durchzuführen.

Die Realisierung dieser Messtechnik und die Lösung der dafür notwendigen Problemstellungen im Zuge des A3 Falcon Projektes ist das Ziel dieser Masterarbeit.

Inhalt

1 Problemstellung

Ziel dieser Masterarbeit ist die Entwicklung einer Messelektronik zur Verifizierung von Simulationsmodellen von Brennstoffzellen im Zuge des von der Forschungsförderungsgeselschaft (FFG) unterstützten Projektes A3 Falcon.

Das Hauptaugenmerk liegt auf einem „Proton Exchange Membrane“ Brennstoffzellenstapel. Erfasst werden soll die Stromdichte und Impedanz im Querschnitt einer Zelle eines Brennstoffzellenstapels der Firma Intelligent Energy mit extern eingeprägtem Strom bei verschiedenen Frequenzen.

Zur Durchführung solcher Messungen muss der Stapel bereits mit zwei segmentierten Sensorplatten, welche von der Firma S++ bereitgestellt werden, gefertigt worden sein. Die zu untersuchende Zelle liegt zwischen zwei Sensorplatten. Diese Platten werden auf die jeweiligen Abmessungen der zu untersuchenden Zellen angepasst. Mit dem Grad der Segmentierung und somit der Auflösung dieses flächigen Messverfahrens steigt der Aufwand quadratisch an. Für jedes Segment müssen Strom und Spannung erfasst werden. Die Strommessung wird nach dem Prinzip einer Spannungsmessung an einem Shunt Widerstand realisiert. Als Shunt Widerstände fungieren die Segmente einer der Sensorplatten. Diese bestehen aus Kupfer und sind somit nicht sehr temperaturstabil. Aus diesem Grund muss auch die Temperatur jedes einzelnen Segmentes zu Korrektur der Strommessung bekannt sein. Damit steigt die Anzahl der zu erfassenden Werte auf drei je Kanal. Das bedeutet, dass bereits bei einer moderaten Auflösung von acht mal acht „Bildpunkten“ 192 synchrone Messungen durchgeführt werden müssen. Bei der Erreichung dieser Auflösung soll es einen zwischen Schritt von vier mal vier als „Proof of Concept“ geben.

Die Erregung der zu untersuchenden Zelle soll im Kilohertz Bereich stattfinden. Daher ist eine erfassbare Bandbreite des Strom- bzw. des Spannungssignals im Bereich von zehn Kilohertz erstrebenswert. Es soll möglich sein harmonische Frequenzkomponenten, die durch nicht lineare Verzerrungen der Zelle entstehen, zu detektieren um damit zusätzlich eine THD Analyse betreiben zu können, welche Aufschlüsse über interne Vorgänge bzw. Betriebszustände der Zelle geben kann.

Da bei der Temperatur keine entsprechend schnellen Änderungen zu erwarten sind ist hier eine Bandbreite im Bereich von zehn Hertz ausreichend. Gemessen werden soll die Temperatur über die Widerstandsänderung von Kupferleitungen in der Sensorplatte.

Neben der Messung muss das System in der Lage sein einen Strom mit Hilfe eines externen Leistungsverstärkers, der als gesteuerte Stromquelle arbeitet, in die Brennstoffzelle einzuprägen. Dieser Verstärker wird im Zuge des Projektes von der Firma AVL entwickelt. Zur Ansteuerung dieses Verstärkers ist die Erzeugung eines sinusförmigen Signals mit variabler Amplitude und Frequenz notwendig.

Die Analysen sollen auch in der Kleinsignal Domäne stattfinden. Aus diesem Grund muss die Messelektronik in der Lage sein entsprechend kleine Amplituden der Messsignale erfassen zu können. Besonderes Augenmerk ist dabei auf den Dynamik Bereich der Spannungsmessung zu richten da dort ein „kleiner“ Wechselspannungsanteil (AC) gemeinsam mit einer „großen“ Gleichspannungskomponente (DC) gemessen werden muss.

Die Auswertung der Daten wird von einer Analyse Software der Firma AVL auf einem handelsüblichen PC durchgeführt werden. Diese Software soll mit Hilfe der Messelektronik in der Lage sein eine elektrochemische Impedanz Spektrographie oder auch eine „total harmonic distortion“ Analyse durchzuführen. Hierfür müssen die Daten, mit entsprechender Datenrate, an diesen „Client“ Computer übertragen werden. Ebenso muss dieser Computer in der Lage sein das Messsystem zu steuern.

Wenn es sich bei dem zu untersuchenden Brennstoffzellenstapel um einen entsprechend „großen“ Typ handelt kann die Ausgangspannung im Bereich mehrerer 100 V liegen. Aus diesem Grund soll die gesamte Messung potentialgetrennt aufgebaut werden.

Das System soll über genügend Rechenleistung verfügen um in zukünftigen Anwendungen selbst Analysen der Messwerte durchführen zu können.

2 Grundlagen

Im Folgenden sollen die für diese Arbeit nötigen theoretischen Grundlagen zusammengefasst werden.

2.1 Brennstoffzellen

Eine Brennstoffzelle ist ein elektrochemischer Energie Konverter welcher die in einem Brennstoff, meist Wasserstoff, enthaltene chemische Energie direkt in elektrische Energie umwandelt. Sie umgeht dabei den sonst meist verwendeten Umweg über thermische und mechanische Energie. Das Umgehen eines jeden Umwandlungsprozesses wirkt sich positiv auf die Effizienz aus. [1]

Es gibt verschiedenste Typen von Brennstoffzellen. Im weiteren Text handelt es sich, wenn nicht anders beschrieben, um PEM Brennstoffzellen.

2.1.1 Einsatzgebiete

Brennstoffzellen eignen sich nicht zuletzt wegen ihrer guten Skalierbarkeit für viele verschiedene Einsatzgebiete.

- Für mobile Anwendungen:
  - Kraftfahrzeuge
  - Boote
  - U-Boote
  - Raumfahrzeuge
  - Bord Netz von Flugzeugen
  - …
- Für stationäre Anwendungen:
  - Versorgung von Insel Systemen
  - Notfallversorgungen
  - …

2.1.2 Funktionsweise

Eine Brennstoffzelle ähnelt in ihrer Funktionsweise einer Batterie. Es gibt eine Anode und eine Kathode die durch einen Elektrolyt voneinander getrennt sind, im Falle der PEM Brennstoffzelle durch eine protonenleitfähige Membran. Im Gegensatz zu einer Batterie muss eine Brennstoffzelle kontinuierlich mit neuem Brennstoff und Oxidationsmittel versorgt werden. Die Elektroden der Brennstoffzelle sollen sich bei der chemischen Reaktion nicht verändern oder verbrauchen. [1]

Die grundlegenden Reaktionen für den Betrieb einer Brennstoffzelle werden wie folgt beschrieben.

- An der Anode:

$$H_2 \rightarrow 2H^+ + 2e^-$$

- An der Kathode:

$$\frac{1}{2}O_2 + 2H^+ + 2e^- \rightarrow H_2O$$

- Gesamt:

$$H_2 + \frac{1}{2}O_2 \rightarrow H_2O$$

Der Anodenseite wird Wasserstoff zugeführt und über eine Gasdiffusionsschicht flächig verteilt. An einer katalytischen Schicht aus Platin werden die Wasserstoffmoleküle in Wasserstoff Ionen und Elektronen getrennt. Die Wasserstoff Ionen können die elektrisch isolierende aber protonenleitfähige Membran durchqueren. Die Elektronen müssen den Umweg über eine externe Leitung, welche die Anodenseite mit der Kathodenseite verbindet, nehmen. Damit kann der Elektronenfluss zum Verrichten von elektrischer Arbeit verwendet werden.

Der Kathodenseite wird Sauerstoff über die Umgebungsluft zugeführt und ebenfalls über eine Gasdiffusionsschicht verteilt. An einer weiteren katalytischen Schicht oxidiert der Wasserstoff zu Wasser und muss abgeführt werden.

2.2 Impedanz Messung

Im folgenden Kapitel sollen die Grundlagen für eine Impedanz Messung zusammengefasst werden.

2.2.1 Impedanz

Die Impedanz eines elektrischen Zweipols beschreibt den komplexen Zusammenhang zwischen dem Strom und der Spannung dieses Systems. Die Impedanz kann so wie jede andere komplexe Zahl als Real und Imaginär Teil oder als Betrag und Phase angeschrieben werden.

Im Kartesischen Koordinaten System besteht die Impedanz aus elektrischer Reaktanz der Projektion auf die imaginäre Achse und dem Wirkwiderstand der Projektion auf die reale Achse.

$$\underline{Z} = Re(\underline{Z}) + jIm(\underline{Z})$$

Im Polaren Koordinaten System wird die Impedanz durch den Betrag des Verhältnisses von Spannung zu Strom und dem durch die Phasenverschiebung zwischen Spannung und Strom bestimmten Winkel dargestellt.

$$\underline{Z} = |\underline{Z}|e^{j\varphi}$$

Weist die Impedanz einen negativen Imaginärteil auf verhält sich der Zweipol kapazitiv, bei einem positiven Imaginärteil induktiv.

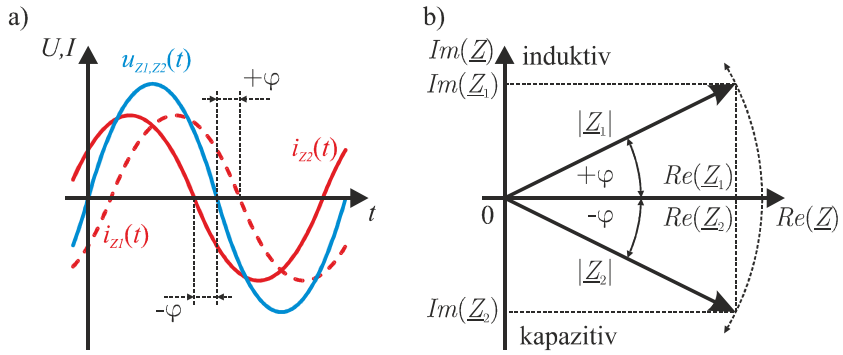

Abbildung 1: Darstellung induktiver und kapazitiver Impedanzen [2]

a) Zeitlicher Verlauf von Strom und Spannung mit positiver und negativer Phasenverschiebung.

b) Darstellung der beiden zugehörigen Impedanzen aus a) mit Betrag und Phase.

2.2.2 Messmethoden

Zur Bestimmung der Impedanz müssen die Spannung am und der Strom durch das System bekannt sein. Neben der Amplitude muss auch die Phasenverschiebung zwischen Strom und Spannung festgestellt werden.

Für die Messung kann das System entweder mit einer bekannten Wechselspannung oder einem bekannten Wechselstrom beaufschlagt werden. Die jeweils andere Größe muss in diesem Fall gemessen werden. In der Praxis werden jedoch beide Größen gemessen um Unsicherheiten bei der Erzeugung der beaufschlagenden Größe zu umgehen. Nach diesem Prinzip wird die Impedanz durch Betrag und Phase bestimmt. [3]

In jedem Fall ist es notwendig den zeitlichen Verlauf von Strom und/oder Spannung zu erfassen. Aus deren Verlauf können Frequenz, Amplitude und Phasenverschiebung bestimmt werden.

Um direkt den Real- und Imaginärteil zu messen kann eine Synchrondemodulation angewendet werden. Wird ein konstanter Wechselstrom in das System eingeprägt kann die Spannung über dem System zum einen in Phase und zum anderen in Quadratur (um 90° verschoben) gleichgerichtet werden. Das Mittel der in Phase gleichgerichteten Spannung ist proportional dem Realteil der Impedanz. Das Mittel der um 90° verschoben gleichgerichteten Spannung ist proportional dem Imaginärteil der Impedanz.

2.2.3 Impedanz Spektrographie

Bei einer Impedanz Spektrographie wird die Impedanz des Zweipols bei verschiedenen Frequenzen bestimmt. Betrachtet man die Änderung der Impedanz über die Frequenz kann der Einfluss verschiedener Zeitkonstanten im System bestimmt werden. Dadurch ist ein Rückschluss auf die internen Parameter des untersuchten Systems möglich. [3]

Ein Beispiel soll die Möglichkeiten einer Impedanz Spektrographie veranschaulichen. Das zu untersuchende System wird mit einem Model, bestehend aus drei in Serie geschalteten Widerständen mit jeweils einem Kondensator parallel, nachgebildet, siehe Abbildung 2. Die Werte der einzelnen Komponenten des Models seien noch unbekannt.

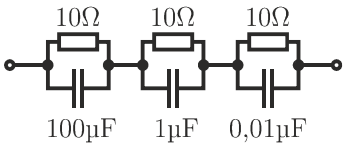

Abbildung 2: RC Modell einer Impedanz mit drei Zeitkonstanten [4]

Wird eine Impedanz Spektrographie durch Impedanz Messungen wie in 2.2.2 beschrieben durchgeführt kann das Ergebnis grafisch in einem Nyquist Diagramm dargestellt werden, siehe Abbildung 3. Der eingezeichnete Verlauf der Impedanz beginnt rechts oben mit der Frequenz Null Hertz, diese steigt nach links bis theoretisch unendlich.

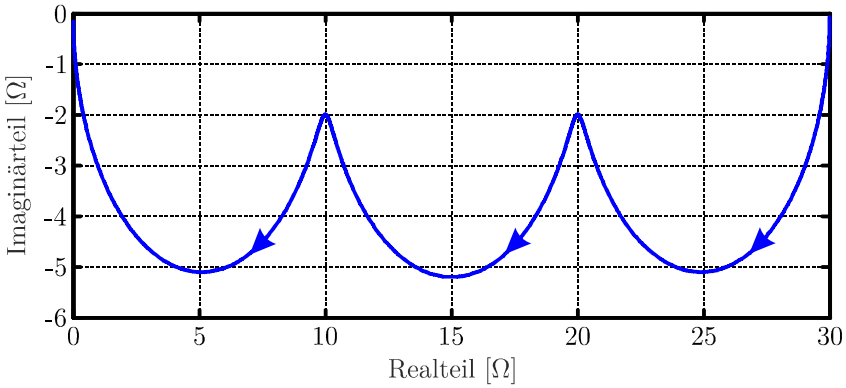

Abbildung 3: Nyquist Diagramm von einer Impedanz mit drei Zeitkonstanten [4]

Bei der Frequenz Null Hertz weist die Impedanz einen reinen Realteil von 30 Ω auf. Die Impedanz wird nur durch die ohmschen Widerstände bestimmt, Abbildung 2. Mit steigender Frequenz nimmt der Einfluss der Kondensatoren auf die Gesamtimpedanz zu. Es ist deutlich zuerkennen wie mit steigender Frequenz der Betrag der Blindwiderstände der Kapazitäten abnehmen. Bei sehr hohen Frequenzen schließen die Kondensatoren ihre Widerstände kurz und der Betrag der Impedanz strebt dem Wert Null entgegen.

2.3 Signalverarbeitung

Dieses Kapitel bietet einen Überblick über die für dieses Projekt notwendige Signalverarbeitungstheorie.

2.3.1 Abtastung

Für die folgenden Erklärungen wird ein idealer Analog zu Digital Konverter angenommen.

Die beim periodischen Abtasten entstehende zeitdiskrete Folge entspricht dem zeitkontinuierlichen Signal (x_c) zu äquidistanten Zeitpunkten. [5]

$$x[n] = x_c(nT)$$

Der Ausgang (x_s) eines idealen Analog zu Digital Konverters berechnet sich aus der Modulation des Eingangssignals mit einer Diracimpulsfolge. [6]

$$x_s(t) = x_c \sum_{n=-\infty}^{\infty} \delta(t - nT)$$

Eine Multiplikation im Zeitbereich entspricht einer Faltung im Frequenzbereich. Die Fouriertransformierte der Diracimpulsfolge (S) entspricht wiederum einer Diracimpulsfolge. [6]

$$S = \frac{2\pi}{T} \sum_{n=-\infty}^{\infty} \delta(\Omega - n\Omega_s)$$

Das Spektrum des Ausganges (X_s) berechnet sich damit aus der Faltung des Spektrums des Einganges mit der Fouriertransformierten der Diracimpulsfolge [6]

$$X_s(j\Omega) = \frac{1}{2\pi} X_c(j\Omega) * \frac{2\pi}{T} \sum_{n=-\infty}^{\infty} \delta(\Omega - n\Omega_s)$$

Abbildung 4 zeigt die graphische Interpretation der Faltung im Frequenzbereich des Einganges mit der Diracimpulsfolge.

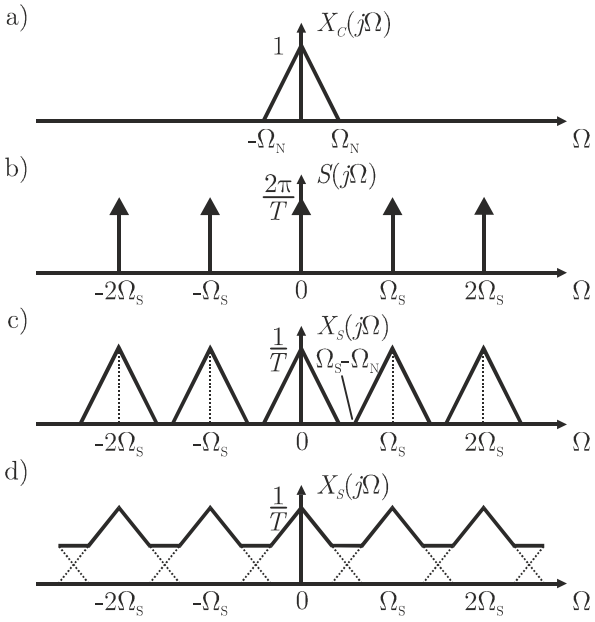

Abbildung 4: Faltung im Frequenzbereich [6]

a) Spektrum des Eingangssignals
b) Spektrum der Diracimpulsfolge
c) Ergebnis der Faltung aus a) mit b)
d) Ergebnis der Faltung mit einem breitbandigeren Eingangssignal

Besitzt das Spektrum des Eingangssignal Anteile bei Frequenzen größer gleich der halben Abtastfrequenz (Ω_s) kommt es bei der Faltung zur Überlagerung dieser Komponenten, siehe Abbildung 4 d).

Bei einer solchen Überlagerung spricht man von „Aliasing“ und eine korrekte Rekonstruktion des Eingangssignals ist nicht mehr möglich. Dies kann durch eine Bandbreitenbegrenzung des Eingangssignals durch ein analoges Tiefpass Filter auf die halbe Abtastfrequenz verhindert werden. Dieser Filter wird als Anti Aliasing Filter bezeichnet.

Die halbe Abtastfrequenz muss in jedem Fall höher sein als die höchste Frequenz Komponente des abzutastenden Signals.

$$f_N > f_{C\,max}$$

Dieses Kriterium wird Nyquist Kriterium genannt. Die halbe Abtastfrequenz wird als Nyquist Frequenz (f_N) bezeichnet.

Damit die theoretische Nutzbandbreite bei gegebener Abtastrate möglichst gut ausgenützt wird ist ein steiles Filter notwendig. Dieses stellt eine große Anforderung an die analoge Filtertechnik.

2.3.2 Überabtastung

Durch die Ausnützung von Überabtastung können die Anforderungen an das analoge AAF heruntergesetzt werden. Soll die weitere Verarbeitung des abgetasteten Signals trotzdem nicht mit der höheren Rate durchgeführt werden kann ein digitales Filter anwendet werden welches die Bandbreite beschränkt und durch Dezimation die Abtastrate herab setzt, siehe Abbildung 5.

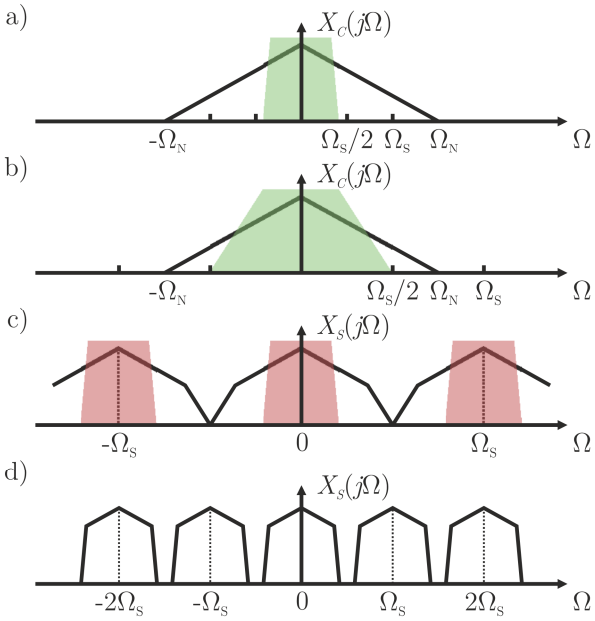

Abbildung 5: Ausnützung von Überabtastung

a) Um bei gegebener Abtastfrequenz möglichst viel der Bandbreite des Eingangssignals nützen zu können wird ein steiles AAF, in Grün dargestellt, benötigt.

b) Wird die Abtastfrequenz verdoppelt kann ein einfacheres AAF verwendet werden um die gleiche Bandbreite wie in a) verwenden zu können.

c) Wird ein digitales Filter, in Rot dargestellt, im zeitdiskreten Bereich, Abtastung von b), gerechnet kann durch Dezimation die Abtastrate wieder gesenkt werden.

d) Spektrum des abgetasteten Signals wie in a) dargestellt bzw. nach Filterung und Dezimation um den Faktor zwei von c).

Durch Überabtastung kann das Quantisierungsrauschen von realen Analog zu Digital Wandlern, die die Amplitude und Zeit eines Signals diskretisieren, beeinflusst werden.

Das Quantisierungsrauschen ist gleichverteilt über den Bereich von Null bis zur Nyquist Frequenz. Wird die Bandbreite durch ein digitales Tiefpass Filter begrenzt werden auch die Rauschanteile mit höherer Frequenz entfernt. Damit ist die Energie des Rauschens innerhalb der Nutzbandbreite geringer als wenn mit einer niedrigeren Frequenz abgetastet worden wäre.

2.3.3 „Total Harmonic Distortion“

„Total Harmonic Distortion“ (THD) wird das Verhältnis der Summe der Leistungen der Oberwellen zu der Leistung ihrer Grundwelle bezeichnet. Mit diesem Wert kann eine Aussage über das Vorhandensein von nichtlinearen Verzerrungen in einem System getroffen werden.

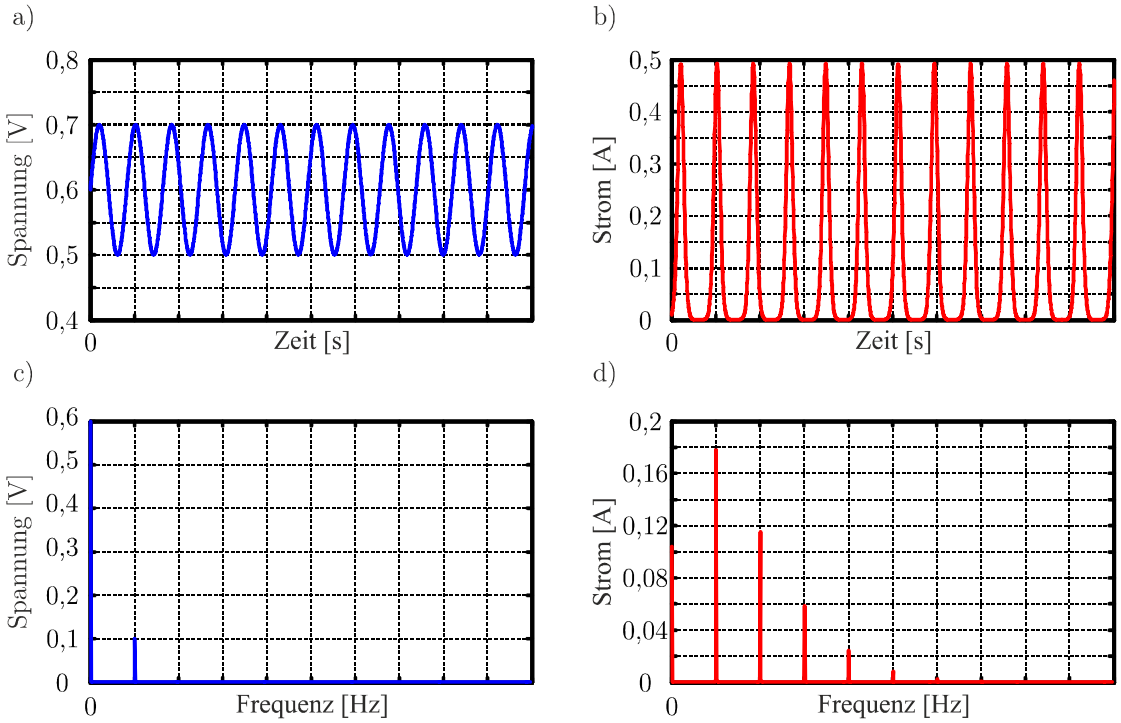

Abbildung 6: Nichtlineare Verzerrung durch eine Diode

a) Wechselspannung mit Gleichanteil über einer Diode.
b) Stromfluss durch die Diode zufolge der Spannung aus a)
c) Positiver Teil des Spektrums der Spannung aus a)
d) Positiver Teil des Spektrums des Stromes aus b)

Abbildung 6 zeigt ein qualitatives Beispiel eines Systems in Form einer Diode mit Spannung als Eingang und Strom als Ausgang. Das Ausgangssignal weist eine deutliche nichtlineare Verzerrung des Eingangssignals auf.

Im Spektrum der Spannung ist neben dem Gleichanteil nur eine Frequenzkomponente vorhanden. Das Spektrum des Stromes weist dagegen neben der Grundfrequenz zusätzliche Harmonische auf.

Der prozentuelle THD Wert berechnet sich wie folgt.

$$THD_{Prozent} = \frac{\sum P_{harmonische}}{P_{Grundfrequenz}} * 100$$

Es ist für diese Berechnung nicht notwendig das gesamte Spektrum des untersuchten Signals zu kennen. Alternativ kann die Spektrale Leistung einzelner Frequenzen mit Hilfe des Görtzelalgorithmus bestimmt werden.

Eine THD Analyse kann ähnlich der Impedanz Spektroskopie, siehe 2.2.3, verwendet werden um Aussagen über innere Vorgänge eines Systems zu treffen.

Anwendung findet die THD Analyse zum Beispiel bei der Beurteilung von Audioverstärkern.

2.4 Hardware

In den folgenden Abschnitten werden die grundlegenden Funktionsweisen einiger der für dieses System benötigter Komponenten erläutert.

2.4.1 „Field Programmable Gate Array“

Bei einem FPGA handelt es sich um eine programmierbare Logik ähnlich einem „Complex Programmable Logic Device“ (CPLD). Diese ICs können so konfiguriert werden, dass sie eine beliebige logische Funktion realisieren. Während ein CPLD für simple Aufgaben gedacht ist können mit einem FPGA komplexe Funktionen oder Algorithmen bis hin zu kompletten Prozessoren implementiert werden. [7]

FPGAs werden häufig in Applikation mit harter Echtzeit Anforderung eingesetzt. Anwendungen mit einem hohen Datendurchsatz können, sofern die Aufgabe parallelisiert werden kann, von dem Einsatz eines FPGAs profitieren. Ein weiteres wichtiges Einsatzgebiet ist bei der Entwicklung von ASICs als Prototyp.

Ein FPGA besteht aus einer Vielzahl simpler Logik Elemente. Ein solches Element beinhaltet eine konfigurierbare „look up“ Tabelle (LUT) mit vier, bei neueren FPGAs auch bis zu sechs, Eingängen und einem Ausgang. Durch die Konfiguration der „look up“ Tabelle kann jede kombinatorische Funktion mit vier beziehungsweise sechs Eingängen abgebildet werden. Der Ausgang kann entweder direkt weiterverwendet werden oder erst über ein D-Flip-Flop, zur Synchronisation mit einem Takt, geleitet werden. Die Verbindung zwischen den einzelnen Logik Elementen wird durch ein engmaschiges Leiternetz, welches an verschiedenen Stellen konfigurierbare Schaltmatrizen besitzt, hergestellt. Neben diesem allgemeinen Netzwerk existiert meist noch ein zweites, welches zur dedizierten Verteilung der Taktsignale genutzt wird. [7]

Sämtliche Speicher für die konfigurierbaren Komponenten werden in SRAM Technologie realisiert und können ihren Wert somit nicht permanent halten. Aus diesem Grund muss ein FPGA seine Konfiguration nach jedem Einschalten von einem zusätzlichen Speicher, welcher sich extern oder auch im selben Gehäuse befinden kann, laden. Das Laden kann über verschiedene herstellspezifische Schnittstellen geschehen. Alternativ wird meistens eine JTAG Schnittstelle angeboten über die das FPGA neben anderen Funktionen auch geladen werden kann. Der Hersteller eines FPGAs bietet in den meisten Fällen passende Lade ROM Bausteine an. Alternativ kann ein FPGA, wenn in der Applikation ein zusätzlicher Prozessor vorhanden ist, von diesem konfiguriert werden.

Zusätzlich zu den konfigurierbaren Logik Elementen bieten FPGAs festverdrahtete Komponenten wie beispielsweise Multiplizierer, PLLs, RAM Blöcke oder Prozessoren und Schnittstellen. Diese Komponenten können schneller arbeiten und verbrauchen weniger Platz in dem FPGA als wenn sie durch konfigurierbare Logik realisiert wären.

Die Konfiguration kann auf verschiedenste Weisen festgelegt werden. Eine Möglichkeit ist es direkt den Inhalt der LUTs sowie wie deren Verbindungen festzulegen. Dieser Ansatz führt, wenn optimiert, zu den effizientesten Designs im Hinblick auf Laufzeit und Platz. Bei komplexeren Systemen empfiehlt es sich die Funktion mit einer Hardware Beschreibungssprache wie VHDL oder Verilog zu beschreiben und einen Synthesizer die FPGA Konfiguration aus dieser Beschreibung erstellen zu lassen.

2.4.2 „Direct Digital Synthesizer“

Ein DDS ist ein System um die Werte eines sinusförmigen Signals mit variabler Frequenz und Phase zu berechnen. Dieses System besteht aus zwei primären Komponenten: einem Phasen Akkumulator und einer „look up“ Tabelle. [8]

Der Phasen Akkumulator ist im Prinzip nichts anderes als ein Zähler der mit einem Takt um einen bestimmten Wert ($\Delta\theta$) inkrementiert wird. Je höher dieses Inkrement ist umso schneller kommt es zum Überlauf des Zählers. Die Frequenz (f_{out}) des erzeugten Signals hängt von diesem Wert, dem Takt (f_{clk}) mit dem inkrementiert wird sowie der Bitbreite des Akkumulators ($B_{\theta(n)}$) ab. [8]

$$f_{out} = \frac{f_{clk} \cdot \Delta\theta}{2^{B_{\theta(n)}}}$$

Die Phase des erzeugten Signals kann durch einen Initialwert des Zählers angepasst werden.

Die Werte im Zählerregister entsprechen einem Signal mit Sägezahnform. Diese Werte werden als Index für die „look up“ Tabelle verwendet um von diesem Sägezahn einen Sinus abzuleiten. [8]

Der Aufbau der „look up“ Tabelle kann je nach Implementierung unterschiedlich sein. Ist die Ausführungszeit das primäre Design Ziel so wird eine komplette Periode des Sinus hinterlegt. Wenn der verfügbare Speicher begrenzt ist reicht es das erste Viertel einer Periode abzuspeichern und diese Werte abschnittsweise zu negieren beziehungsweise den Index umzudrehen. Bester Kompromiss zwischen diesen beiden Zielen ist das Abspeichern einer Halbwelle. So können die Werte der zweiten Halbwelle durch einfaches Negieren bestimmt werden. Die Größe der „look up“ Tabelle hängt von der

gewünschten Bitbreite des erzeugten Signals und der Bitbreite des als Index verwendeten Phasenakkumulators ab.

2.4.3 Aktive Filter

Mit aktiven Filtern können analoge Filter mit einer Ordnung höher zwei realisiert werden. Bei den aktiven Bauelementen handelt es sich um Transistoren oder Operationsverstärker.

Auch für Filter zweiter Ordnung ist der Einsatz von aktiven Filtern vorteilhaft gegenüber einer RLC Kombination. Die Induktivität des RLC Filters muss für tiefe Grenzfrequenzen entsprechend groß dimensioniert werden. Damit ist der Einsatz von großen Bauformen von Induktivitäten notwendig. Beim Einsatz eines aktiven Filters zweiter Ordnung, wie z.B. einem Sallen Key Filter, sind zusätzlich zu einem Operationsverstärker Widerstände und Kapazitäten notwendig. Diese nehmen in Summe weniger Platz ein als die RLC Variante. [9]

Wird in einer Anwendung ein analoges Signal abgetastet und steht eine Recheneinheit zur Verfügung muss entschieden werden ob der Einsatz aufwändiger analoger aktiver Filter zum Beispiel durch Überabtastung umgangen werden kann. Nachteile analoger Filter im Vergleich zu digitalen Filtern sind:

- Anzahl der benötigten Bauelemente
- Genauigkeit der gewünschten Grenzfrequenz hängt von der Genauigkeit der Werte der verwendeten Kapazitäten ab
- Langzeit Stabilität der Filter Eigenschaften ist auf Grund der Alterung der Bauteile problematisch

Vorteile der Verwendung von aktiven Filter in der Signalkette von abtasteten Systemen sind der geringere Entwicklungsaufwand sowie die geringere benötigte Rechenzeit für die digitale Signalverarbeitung.

Für Serienprodukte ist in jedem Fall die Anzahl der benötigten Bauteile das ausschlaggebende Argument gegen aufwendige analoge Signalverarbeitung.

2.4.4 Sukzessive Approximation ADC

Ein „Analog to Digital Converter“ (ADC) der nach dem Prinzip der Sukzessiven Approximation (SAR, Succesive Approximation Register) arbeitet wird grundsätzlich aus einem Komparator, einem „Digital to Analog Converter“ (DAC), einem Register für das Ergebnis und einer Steuerlogik aufgebaut.

Bei der Konversion wird, beginnend mit dem höchstwertigsten Bit, jedes Bit im Ergebnis Register nacheinander auf eins gesetzt. Der Wert wird nach dem Setzen jedes Bits durch den DAC in eine analoge Spannung gewandelt welche durch den Komparator mit der Eingangsspannung verglichen wird. Ist die gewandelte Spannung größer wird dieses Bit wieder gelöscht, wenn nicht bleibt es gesetzt. Mit jedem Bit nähert sich der Wert im Ergebnis Register dem Eingangssignal an. [10]

Die Auflösung des Ergebnisses dieser Art von ADCs kann unter gewissen Voraussetzungen durch die Mittelung mehrerer aufeinander folgender Abtastwerte erhöht werden. Dabei muss bedacht werden das dies nur möglich ist wenn das zu messende Signal einen Rauschanteil besitzt der sich zumindest im Bereich eines LSBs des Ergebnisses bewegt. In der Praxis trifft dies meist zu. Würde das Eingangssignal theoretisch nicht rauschen würde jede Konversion nach dem Prinzip dieses ADCs dasselbe Ergebnis liefern und eine Mittelung würde zu keiner besseren Näherung des tatsächlichen Wertes führen.

Ein Sigma-Delta Wandler würde sich in diesem Fall anders verhalten. Sein integrierendes Prinzip sorgt dafür dass je länger gemessen wird der tatsächliche Wert umso genauer angenähert wird.

Rauscht das Eingangssignal mit zumindest einem LSB steigt die mögliche Näherung mit der Anzahl der Abtastwerte des SAR ADC. Würde sich zum Beispiel der tatsächliche Wert genau zwischen den beiden Werten, die durch das LSB dargestellt werden können, befinden, würde im Ergebnis das LSB einen zufälligen Wert annehmen. Bei einer genügend großen Anzahl an Abtastwerten wird sich eine Verteilung des Wertes des LSBs von 50 % „high“ und 50 % „low“ einstellen. Durch die Mittelung der Werte mit dieser Verteilung kann der tatsächliche am Eingang anliegende Wert recht genau bestimmt werden. Dasselbe Prinzip funktioniert auch mit Eingangswerten welche nicht genau zwischen den beiden kleinsten erfassbaren Werten liegen. In diesem Fall stellt sich bei statistischer Gleichverteilung der Amplituden des Rauschens ein entsprechendes Verhältnis ein.

3 Konzept

In diesem Kapitel sollen sämtliche Überlegungen diskutiert werden, die zu dem realisierten System geführt haben. Das System muss in der Lage sein die geforderten Spezifikationen zu erfüllen, trotz der Komplexität soll es mit begrenzten Ressourcen machbar bleiben.

3.1 Hardware System Architektur

Zur Implementierung der Hardware bedarf es zuerst eines sinnvollen Konzeptes. Dieses muss der Software ermöglichen die gewünschte Funktionalität realisieren zu können. Es erweist sich als sinnvoll einen Entwicklungsschritt weiter zu denken. Es sollten Möglichkeiten zur leichten Änderung oder Erweiterung, wenn dies mit vertretbarem Aufwand möglich ist, vorgesehen werden.

3.1.1 Die Sensorplatten

Alle Größen werden mit Hilfe von zwei fest in der zu untersuchenden Brennstoffzelle verbauten Sensorplatten gemessen. Eine dieser speziell gefertigten Leiterplatten befindet sich auf der Kathodenseite, die andere auf der Anodenseite einer Zelle. Der Anschluss der Sensorplatten erfolgt über mehrere hochpolige Steckverbinder.

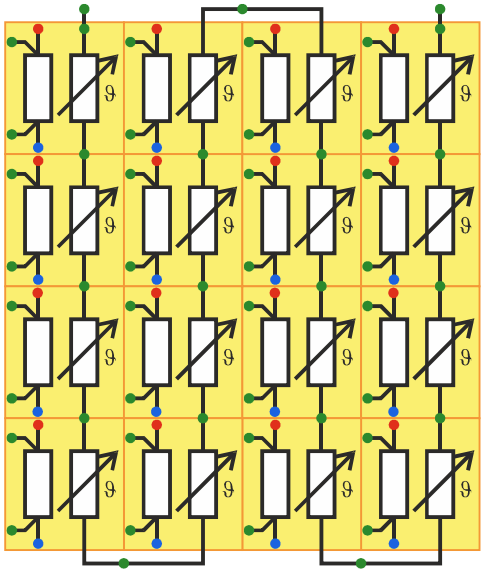

Abbildung 7: Schematischer Aufbau der Sensorplatten mit 16 Segmenten

Die Kontakte in grün sind zum Steckverbinder geführt. Die roten Kontakte befinden sich auf der Oberseite der Sensorplatte, die Blauen auf der Unterseite.

Die Platten sind segmentiert um die flächige Verteilung der gemessenen Größen feststellen zu können. Jedes Segment enthält einen Shunt Widerstand mit Anschlüssen für eine Vierleiter Messung und einen Widerstand zur Messung der Temperatur, siehe Abbildung 7.

Die Kontakte an der Ober- und Unterseite der Leiterplatten stellen die „Drive“ Anschlüsse des Shunt Widerstandes dar. Die Kontakte zum Messen sind über die Steckverbinder erreichbar.

Die Temperaturfühler der Segmente sind in Serie geschaltet. Abgriffe der Verbindungen zwischen den Fühlern sowie zwei Leitungen zum Einprägen eines Stromes sind zu den Steckverbindern geführt

Die Spannungsmessung erfolgt zwischen den inneren Kontakten des jeweiligen Segmentes der beiden Sensorplatten. Es wird die Überlagerung der Zellspannung mit einem Erregersignal gemessen.

Die Strommessung erfolgt über die Shunt Widerstände einer der Sensorplatten. Dabei wird der Laststrom inklusive eines eingeprägten Stromes gemessen.

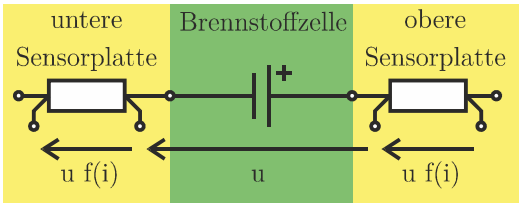

Abbildung 8: Strom- und Spannungsmessung an den Sensorplatten

Die Temperatur wird über den Widerstand der Temperaturfühler bestimmt. Dafür wird die Spannung über jedem Fühler gemessen und in Relation zum fließenden Strom gesetzt.

3.1.2 Messkanäle

Der erste Schritt bei der Definition des Messkanals ist die Festlegung der Bandbreite und die damit verbundene Abtastrate.

Für die Strom- und Spannungsmessung ist eine Bandbreite von zehn bis zwanzig Kilohertz gefordert. Eine gängige maximale Abtastrate von ADCs liegt bei 32 kHz, was theoretisch eine maximale Frequenz des Eingangssignals von 16 kHz erlaubt. Ein derartiger ADC wäre für dieses System gut geeignet. Allerdings ist ein sehr steiles analoges Anti Aliasing Filter (AAF) notwendig um die Bandbreite von 16 kHz annähernd voll ausnutzen zu können. Ein solches analoges Filter ist zum einen schaltungstechnisch aufwändig und zum anderen weist es einige negative Eigenschaften auf, siehe 2.4.3. Ein

Problem beim Einsatz in diesem System ist der wegen der hohen Anzahl an Kanälen benötigte Platzbedarf der Bauteile auf der Platine.

Eine Alternative ist die Ausnutzung von Überabtastung um die Anforderung des analogen AAF herunter zu setzen und stattdessen ein digitales Dezimationsfilter zu rechnen. Die nötige Rechenleistung um dieses Filter an einer zentralen Stelle zu implementieren wäre wegen der Kanalanzahl und der hohen Abtastrate nicht zu unterschätzen. Dieses Filter muss nicht an die Problemstellung angepasst sein. Es kann ein ADC verwendet werden der Überabtastung betreibt und die Datenrate durch ein in Hardware implementiertes Dezimationsfilter auf 32 kHz reduziert.

Die Auflösung der ADCs spielt vor allem bei der Spannungsmessung eine große Rolle, da das eingeprägte Signal inklusive der Zellenspannung gemessen wird. Als guter Kompromiss zwischen anfallender Datenmenge und gewünschter Genauigkeit sollen die Messungen mit einer Auflösung von 16 Bit durchgeführt werden.

Um den Messbereich der beiden ADCs optimal auszunützen soll die Amplitude der Eingangssignale durch einen Vorverstärker entsprechend angepasst werden. Im Fall der Strommessung muss die Messung als differentieller Eingang ausgeführt werden.

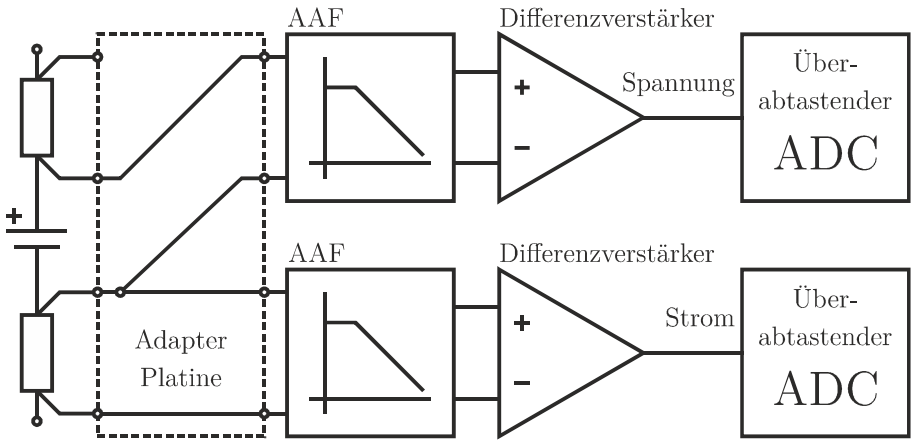

Abbildung 9: Aufbau eines Messkanals für Strom und Spannung [11]

Durch die Adapterplatine wird das System mit den Sensorplatten verbunden, siehe 3.1.9.

Aus Gründen der Symmetrie wird auch die Spannungsmessung differentiell ausgeführt. Dadurch kann das Layout des Strom- und Spannungskanals identisch sein. Für die Impedanz Messung ist die Phaseninformation zwischen dem Strom und der Spannung essentiell, deswegen soll der Phasengang der beiden Vorverstärker bzw. AAF möglichst identisch über den auszuwertenden Frequenzbereich verlaufen.

Der ADC für die Temperaturmessung kann wesentlich langsamer abtasten. Er muss allerdings über eine entsprechend hohe Auflösung verfügen um die geringe Widerstandsänderung der Temperatursensoren zu erfassen. Wegen der niedrigen Abtastrate kann hier ein ADC verwendet werden der mehrere analog gemultiplexte Eingänge besitzt. Dies bringt einen Vorteil in Bezug auf den notwendigen Platz auf der Platine. Die Wahl der Datenrate fällt in diesem Fall auf 25 Hertz was für die Temperaturmessung ausreichend ist. Dies ermöglicht den Einsatz eines ADCs mit eingebautem digitalem Filter, welches längere Einschwingzeiten aufweisen kann.

In diesem Fall kann ein für Temperatur bzw. Druckmessung spezialisierter ADC verwendet werden. Ein solcher ADC hat beispielsweise sechs differentielle Kanäle und wendet selbst einen digitalen Tiefpass Filter auf diese Kanäle an. Zur Realisierung einer ratiometrischen Messung muss der ADC die Verwendung einer externen Referenz zulassen.

Ein Messkanal besteht somit aus zwei überabtastenden ADCs mit einer Datenrate von 32 kHz mit Vorverstärker und AAF sowie einem ADC Kanal eines langsamen ADCs mit AAF.

Die ADCs sollen über eine SPI Schnittstelle ausgelesen werden. Bei den 64 Strom- und Spannungskanälen wären das bereits 128 Schnittstellen. Daher sollten ADCs verwendet werden welche in einer „daisy chain“ Konfiguration betrieben werden können. Die Anzahl von ADCs in einer Kette hängt davon ab wie schnell es möglich ist sie auszulesen und ihre Messwerte weiter zu verarbeiten. Ein guter Kompromiss sind in diesem Fall acht Stück, bei einer SPI Frequenz von etwa 10 MHz. Damit werden 16 Schnittstellen benötigt.

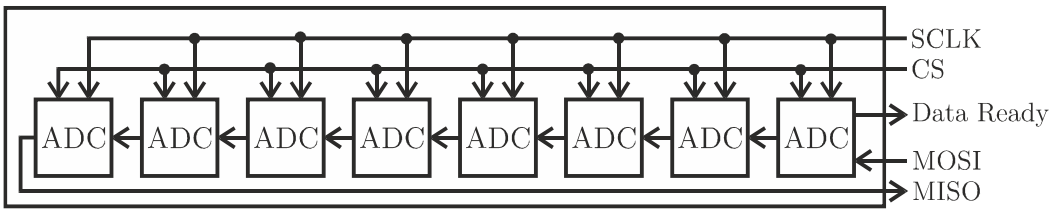

Abbildung 10: Aufbau einer SPI „daisy chain“

Für das Auslesen der langsamen ADCs werden zusätzliche SPI Schnittstellen benötigt. Je „daisy chain“ ist ein ADC, wenn dieser vier Eingänge besitzt, für die Temperaturmessung notwendig. Die SPI dieser ADCs müssen nicht in einer „daisy chain“ zusammengefasst werden, können aber mit Hilfe dezidierter „chip select“ Leitungen parallel geschaltet werden.

3.1.3 Datensammlung

Die gemessenen Werte müssen von den ADCs entsprechend schnell ausgelesen werden. Hierfür könnte eine Gruppe von „micro“ Prozessoren eingesetzt werden. Vorteil dieser Lösung wäre die einfache Programmierung sowie meistens ein bis zwei verfügbare „Hardware SPI master“ Schnittstellen. Ein solcher „controller“ müsste in der Lage sein, wenn er zum Beispiel zwei „SPI Slave“ Ketten bedient, mehr als 1024000 Messwerte pro Sekunde, im einfachsten Fall, zwischenspeichern und weiterleiten zu können. Ist zusätzlich eine Verstärkung beziehungsweise Nullpunktkorrektur notwendig. Ergibt sich ein nicht unerheblicher Rechenaufwand der bewältigt werden muss. Bei zwei SPI Schnittstellen pro Einheit wären acht „controller“ notwendig und ein Bussystem mit entsprechender Nutzdatenrate welches die Daten weiterleiten kann.

Eine alternative Lösung ist die Verwendung eines FPGAs. Der größte Nachteil bei dessen Einsatz liegt im wesentlich höheren Entwicklungsaufwand. Die parallele Implementierung von mehreren Schnittstellen ist hier kein Problem. Bei der Wahl eines entsprechend „großen“ Typs, in Bezug auf die verwendbaren GPIO Pins und Logikeinheiten, könnten alle 16 SPI Schnittstellen von einer Einheit bedient werden. Dies würde auch die Notwendigkeit eines gemeinsamen externen Bus Systems eliminieren.

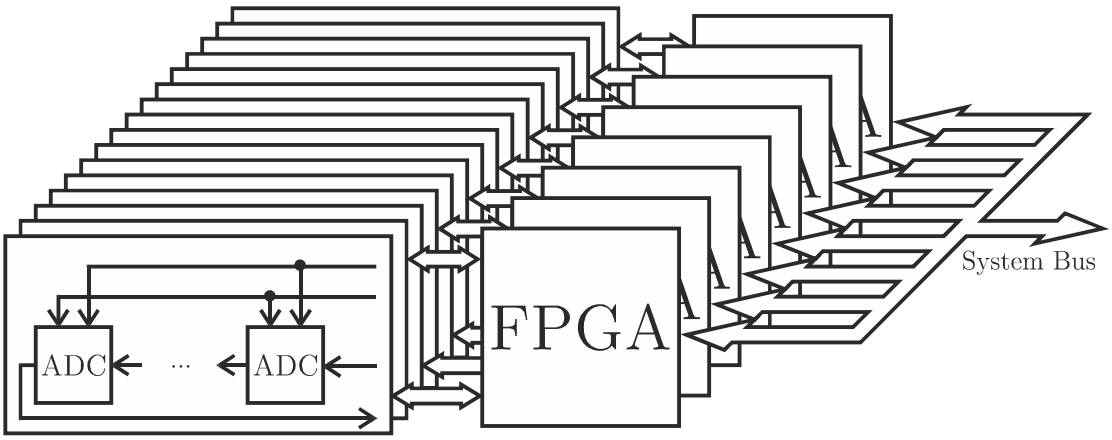

Abbildung 11: Verteilung der SPI „daisy chains“ auf FPGAs für 32 Kanäle

Problematisch bei der Benützung eines ausreichend großen Typs ist dessen Gehäuse, welches meistens als „ball grid array“ ausgeführt und händisch nicht lötbar ist. Aus diesem Grund muss ein kleineres FPGA gewählt werden welches über ein, mit einfachen Mitteln, lötbares Gehäuse verfügt. Ein solches FPGA hat weder genug Pins noch ausreichend Logikeinheiten um sämtliche SPI Schnittstellen zu implementieren. Deshalb muss auch bei der FPGA Variante das Auslesen auf mehrere Einheiten verteilt

werden. Allerdings kann hier ein einfacher Adress-/Datenbus aufgebaut werden um die Messwerte von den einzelnen Datensammlern abzuholen, siehe Abbildung 11.

Dieses Bus System wird in den folgenden Abschnitten als System Bus bezeichnet. Ein solcher Bus erlaubt es den Zugriff auf die I/O Schnittstellen der einzelnen Teilnehmer in den Speicherbereich eines Prozessors zu legen, was ein sehr elegantes Auslesen über einen einfachen Speicherzugriff ermöglicht. Die Messwerte besitzen eine Auflösung von 16 Bit. Es bietet sich daher eine Bus Breite von 16 Bit an.

Würde jeder Teilnehmer, bei 64 Kanälen, über eine eigene „chip select“ Leitung angesprochen, wären dafür 16 Leitungen nötig. Kodiert man diese Information in die Adresse können 12 Leitungen eingespart werden. Um für künftige Anwendungen mehr Teilnehmer erlauben zu können werden 5 Adressleitungen zur „chip select“ Generierung reserviert.

Die benötigte Adressbreite ergibt sich aus der Anzahl der zu adressierenden Daten. Es müssen 48 Messwerte ausgelesen werden, dafür sind zumindest sechs Leitungen notwendig. Eine zusätzliche Leitung wird für zukünftige Anwendungen reserviert. Zur Kalibrierung müssten 48 Offset und 48 Verstärkungskorrekturwerte abgespeichert werden. Es wäre aber auf Grund der zusätzlichen Leitungen nicht sinnvoll diese direkt zu adressieren. Dies kann durch „paging“, Umschalten des Speicherbereiches, erfolgen.

Für den System Bus sind somit 28 Leitungen notwendig, 12 Adressleitungen und 16 Datenleitungen.

3.1.4 Datenübermittelung

Die von den einzelnen Datensammlern ausgelesenen Messwerte müssen an einer Stelle zusammen geführt und an den „Client“ Computer weitergeleitet werden. Bei der zu erwartenden Datenrate von etwas mehr als acht Mega Byte pro Sekunde, bei 64 Kanälen und 32 kHz Abtastrate, kommen nur entsprechend schnelle Schnittstellen wie SATA, USB oder Ethernet in Betracht. Der Einsatz des Messsystems in einer Prüfstandumgebung erfordert die entfernte Steuerung und Auslesbarkeit. Diese beiden Anforderungen können mit der Ethernet Schnittstelle erfüllt werden.

Für den Einsatz von Ethernet ist ein „Ethernetstack“ notwendig. Diese Softwarekomponente kann zugekauft oder durch Verwendung einer „Open Source“ Variante realisiert werden. Ein embedded Linux System implementiert einen solchen „Stack“ und bringt zusätzlich den Komfort eines Betriebssystems, dessen Hardware Abstraktion und High Level Programmierung, mit sich.

Um die anfallende Datenmenge zu verarbeiten und nebenbei ein Betriebssystem auszuführen muss die Wahl der Plattform auf einen ausreichend performanten Rechner fallen.

Der hardwaretechnische Aufwand mit Prozessor, Arbeitsspeicher, Festwertspeicher und dem Platinen Layout für ein solches System wäre zu groß für diese Masterarbeit und nur bei einem Serienprodukt gerechtfertigt. Diverse Firmen bieten für solche Fälle Module in Form einer Aufsteckplatine an, welche über verschiedenste hochpolige Stecker kontaktiert werden.

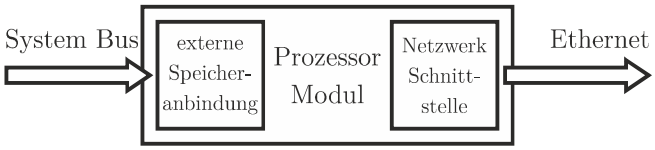

Abbildung 12: Datensammlung und Weiterleitung über Ethernet

Um die Verbindung über den System Bus mit den Datensammlern realisieren zu können muss ein solches Modul bzw. der darauf verbaute Prozessor eine externe Speicherschnittstelle bieten, siehe Abbildung 12.

3.1.5 Datenkonzentration

Das Zusammenführen der Messwerte würde den Prozessor einiges an Ausführungszeit kosten da das Abholen der Daten in Echtzeit erfolgen muss, also im Prinzip nur Interrupt gesteuert möglich ist. Das Bedienen eines Interrupts mit einer Rate von 32 kHz stellt einen Applikationsprozessor noch vor keine unlösbare Aufgabe. Sollte es aber notwendig werden Rechenoperationen auf die Messdaten anzuwenden könnte es bereits schwierig werden. Um hier mögliche Engpässe zu vermeiden wird diese Aufgabe in ein weiteres FPGA ausgelagert. Dieses „Master“ FPGA liest die Daten vom System Bus und führt die gemessenen Werte zusammen. Es können bereits Pakete vorbereitet werden, welche vom DMA des Prozessors ausgelesen werden.

Abbildung 13: Prozessor Modul mit FPGA als Gateway für den System Bus

Damit dient dieses FPGA als „Gateway“ zwischen dem System Bus und dem Adress-/Datenbus zum Prozessor, welcher im weiteren als Modul Bus bezeichnet wird.

3.1.6 Signalerzeugung

Zur Einprägung von Strömen in die Brennstoffzelle muss eine externe Stromquelle angesteuert werden. Diese Ansteuerung soll durch einen Signalgenerator erfolgen. Dieser soll ein Sinussignal mit variabler Amplitude und Frequenz erzeugen. Die Frequenz soll in einem Bereich von 0.01 bis 16 kHz eingestellt werden können. Diese Aufgabe kann von einem DDS IC mit eingebautem DAC erledigt werden. Zur Anpassung der Amplitude kann ein multiplizierender DAC nachgeschaltet werden. Dieses Signal muss noch gefiltert werden um Frequenzanteile der Ausgaberate zu unterdrücken. Die Anforderungen an dieses analoge Filter können durch Einsatz von Überabtastung herabgesetzt werden. Die Ausgaberate sollte möglichst hoch sein. Dadurch ist ein Tiefpassfilter zweiter Ordnung (Sallen Key) in der Lage diese Anforderungen zu erfüllen.

Der Pegel des erzeugten Signals muss an den Eingangspegel der externen Stromquelle angepasst werden. Dies kann durch den Verstärkungsfaktor des zur Filterung nötigen aktiven Filters geschehen.

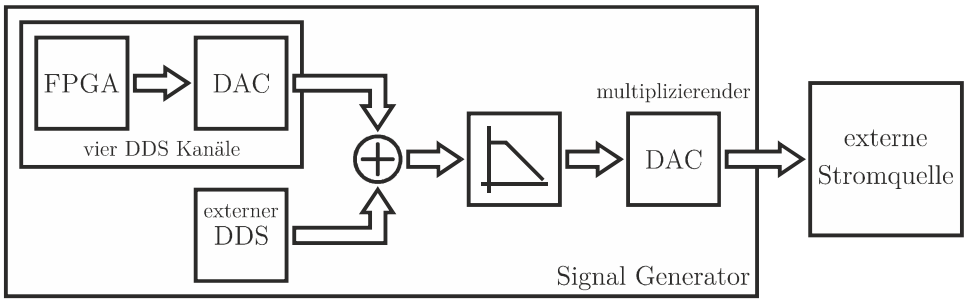

Abbildung 14: Schematische Darstellung der Signalerzeugung

Für THD Analysen wäre es vorteilhaft ein Signal mit mehr als einer Frequenzkomponente ausgeben zu können. Dies kann durch mehrere Kanäle des bereits beschriebenen Generators und eines analogen Summierers realisiert werden. Ein solcher diskreter Aufbau würde eine große Fläche auf der Platine einnehmen sowie eine hohe Anzahl an eng tolerierten Bauteilen benötigen. Deshalb ist es sinnvoller die DDS Kanäle in einem FPGA zu implementieren und das Ergebnis mit Hilfe eines schnellen DAC auszugeben. Besitzt der DAC eine entsprechend hohe Auflösung kann der multiplizierende DAC entfallen. Eine Auflösung von 16 Bit ist für diese Anforderung angemessen. Das Ziel für den gerechneten DDS ist dieselbe Abtastrate wie jene des externen DDS.

Nachteil dieses Konzeptes ist der höhere Entwicklungsaufwand. Das Gesamtsystem sollte möglichst schnell Einsatzbereit sein. Aus diesem Grund wurde entschieden, den im FPGA gerechneten DDS Kanälen einen diskreten Kanal hinzuzufügen.

Die Ansteuerung der Komponenten kann über einen gemeinsamen SPI Bus mit mehreren Auswahl (chip select) Leitungen erfolgen. Dieser Bus muss mit einer hohen Bitrate betrieben werden. Innerhalb einer Periode muss der nächste DDS Wert berechnet und an den DAC gesendet werden können.

3.1.7 IO Interface

Um die externe Stromquelle steuern zu können wird jeweils ein digitaler Ein- und Ausgang benötigt. Der Ausgang dient dem Ein- und Ausschalten und der Eingang um ein Überstromsignal abfragen zu können. Im Falle eines Überstromes soll die Signalerzeugung möglichst schnell abgeschaltet werden.

Um die schnellste Reaktion auf einen Überstrom zu erhalten bietet es sich an diese Überwachung im „Master“ FPGA zu realisieren.

Um Zusätzliche Geräte oder Schalter bedienen zu können sollen jeweils vier digitale Ein- und Ausgänge vorgesehen werden.

3.1.8 Modularer Aufbau

Die bisher beschriebenen Komponenten benötigen eine große Platinenfläche. Eine dementsprechend große Platine ist schwierig zu handhaben, zum Beispiel beim Einbau in ein Gehäuse. Ein solches Design bietet auch keine Möglichkeit Meilensteine im Entwicklungsprozess zu setzen. Das System müsste als ganzes fertiggestellt und in Betrieb genommen werden.

Aus diesen Gründen wird ein System aus mehreren Platinen im Standard Europakartenformat, welche übereinander gestapelt werden können, angestrebt. Für die Verteilung der Komponenten beziehungsweise der Aufgaben bietet es sich an eine Basisplatine und eine Reihe von identischen „Slave“ Karten zu definieren.

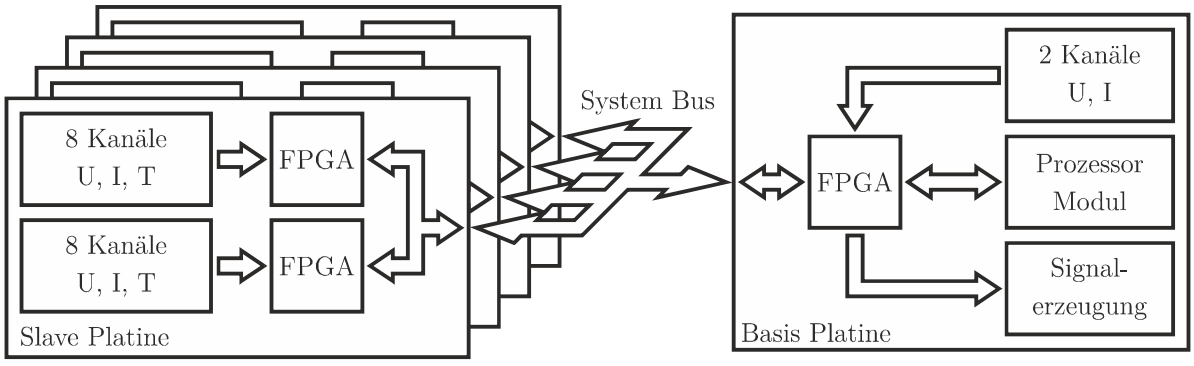

Abbildung 15: Modularer Aufbau für 64 Kanäle mit „Slave“ und Basis Platinen

Eine „Slave“ Karte verfügt über eine Anzahl von Messkanälen und eine Schnittstelle zum gemeinsamen System Bus. Ein in der Aufgabenstellung definiertes Zwischenziel ist ein System mit 16 Kanälen. Es bietet sich an die Anzahl der Kanäle pro „Slave“ Einheit auf 16 festzulegen. Damit werden auf jeder „Slave“ Platine zwei FPGAs, 32 schnelle ADCs und vier langsame ADCs verbaut.

Die Basisplatine kann Träger des Prozessormodules sein und das Zusammenführen der Messdaten sowie die Kommunikation mit dem „Client“ Computer übernehmen. Dabei ist das im Abschnitt 3.1.5 definierte zusätzliche FPGA von Vorteil weil die Basisplatine durch Hinzufügen von zusätzlichen Messkanälen auch ohne eine „Slave“ Karte genutzt werden kann. Dieses FPGA wird auch zur Signalerzeugung, wie in 3.1.6 beschrieben, verwendet.

3.1.9 Verbindungen

Der System Bus muss neben dem eigentlichen Adress-/Datenbus noch weitere Informationen im System verteilen. Dazu gehören ein gemeinsamer Abtasttakt um die Synchronität zwischen dem „Master“ FPGA und den „Slave“ FPGAs zu gewährleisten sowie eine Leitung zur Signalisierung des Beginns des Messvorganges.

Den Datensammlern müssen Adressen zugeordnet werden um sie über den Bus ansprechen zu können. Dies könnte zum Beispiel durch einen Kodierschalter realisiert werden. Ein solcher Ansatz ist einfach aber nicht sehr praktisch da händisch die korrekte Adresse eingestellt werden muss.

Mittels einer Kette, in der jeder Teilnehmer seine Adresse um eins inkrementiert weiter gibt, können die Adressen automatisch nach dem Systemstart zugeordnet werden.

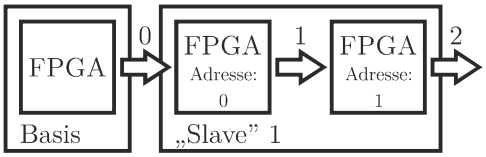

Abbildung 16: Ablauf der Verteilung der Adressen der FPGAs nach Systemstart

Für diese Zuordnung wären fünf Leitungen notwendig. Eine Leitung kann eingespart werden wenn nicht die Adresse der Teilnehmer, sondern die Adresse der „Slave“ Platine verteilt wird und das niederwertigste Bit der Adresse den beiden Teilnehmern über einen nach Masse beziehungsweise Versorgung gezogenen Pin kodiert wird. Es muss jedoch eine Leitung von Teilnehmer zu Teilnehmer führen um anzeigen zu können

wann die Adresse übernommen werden darf und eine gemeinsame Leitung die dem Master anzeigt ab wann alle Teilnehmer bereit sind. Diese gemeinsame Leitung kann durch eine „wired and“ Logik realisiert werden.

Damit sind bereits 37 Leitungen notwendig womit die Wahl auf einen hochpoligen Steckverbinder fällt. Um Kabel im System zu vermeiden sollen die Platinen einfach übereinander gesteckt werden. Der gewählte Stecker muss an derselben Stelle auf beiden Seiten der Platine positioniert werden.

Neben der Information soll auch die Spannungsversorgung über denselben Stecker wie der System Bus verteilt werden. Die einzelnen Verbindungen eines solchen Steckers sind nicht für allzu hohe Ströme ausgelegt, deshalb müssen mehrere Kontakte parallel geschaltet werden. Es muss eine ausreichend große Anzahl an Kontakten sein um sicher zu gehen dass auch im Fall eines Ausfalles oder schlechten Kontaktierung einzelner Verbindungen die Restlichen nicht überlastet werden.

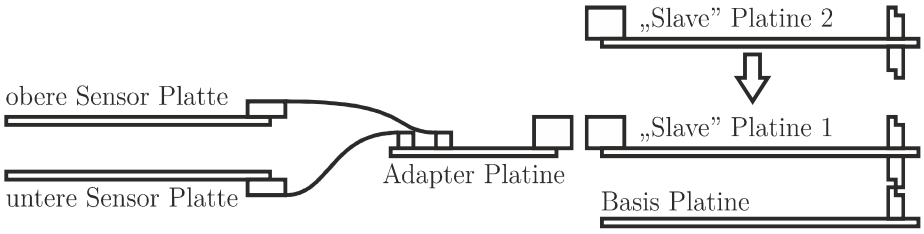

Abbildung 17: Darstellung der Verbindung der Systemkomponenten

Neben der Verbindung zwischen den Modulen muss die Verbindung zu der Sensorplatte definiert werden. Die Platte ist mit einem hochpoligen Steckersystem bestückt. Weil die Pinbelegung je nach Anzahl der Segmente variiert ist es nicht sinnvoll die Module direkt mit den „Slave“ Karten zu verbinden. Eine bessere Lösung ist hier eine zusätzliche Adapterplatine welche die Pinbelegung der Sensorplatte an die des Messsystems anpasst. Mit einem solchen Adapter kann schnell auf Änderungen dieser Sensorplatten reagiert werden. Damit kann ein leichter handhabbares Stecker System wie Messer/Feder Leisten (DIN 41.612) zwischen Slave und Adapterplatine zum Einsatz kommen.

3.1.10 Galvanische Trennung

Eine galvanische Trennung kann in diesem System an verschiedenen Stellen durchgeführt werden. Es ist zu bedenken, dass neben den Informationsleitungen auch die Versorgung getrennt werden muss.

Die Stelle sollte nach den Kriterien der wenigsten Leitungen, der niedrigsten Datenrate und der geringsten zu übertragenden Leistung ausgewählt werden. Die Stellen die sich zur Trennung eignen sind:

- Die Busverbindung zwischen dem Datenkonzentrator und den Datensammlern auf der „Master“ Platine
- Die Busverbindung zwischen dem Datenkonzentrator und den Datensammlern auf jeder „Slave“ Platine
- Die SPI Schnittstellen zwischen den Datensammlern und den ADCs

Bei der Trennung zwischen dem Datenkonzentrator und den Datensammlern auf der „Master“ Platine müsste die gesamte Leistung zur Versorgung der „Slave“ Platinen übertragen werden. Führt man stattdessen die Trennung auf den Slave Platinen durch steigt die Anzahl der benötigten Trennelemente mit der Anzahl der „Slaves“. Es muss nur die Leistung zur Versorgung eines Slave Systems übertragen werden.

Am wenigsten Leistung muss bei der Trennung der SPI Schnittstellen übertragen werden, da ab dort nur noch die ADCs und die Eingangsverstärker zu versorgen sind. Die Anzahl der benötigten Trennelemente ist höher als bei der Trennung des System Busses auf der Master Platine. Sie ist jedoch geringer als bei Trennung des Busses auf den „Slave“ Platinen, da die SPI Busse wesentlich weniger Leitungen benötigen.

Die Trennung der SPI Schnittstellen stellt einen guten Kompromiss dar und wird aus diesem Grund realisiert. Zusätzliche Messkanäle sowie der Signalgenerator auf dem Master können nach dem gleichen Prinzip getrennt werden.

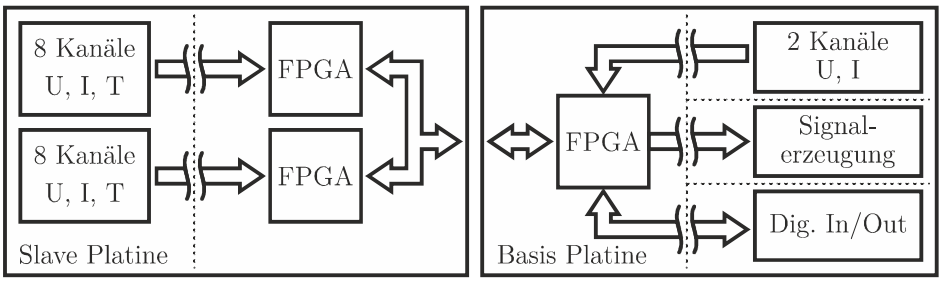

Abbildung 18: Galvanisch voneinander getrennte Bereiche auf beiden Platinen

Datenleitungen werden über Magnetkoppler getrennt und für die Versorgung werden DC/DC Wandler Module verwendet. Die digitalen Ein- und Ausgänge werden über Optokoppler getrennt. Dies hat den Vorteil, dass die getrennte Seite keine gemeinsame Versorgung eines Magnetkopplers benötigt und damit jeder voneinander getrennt ist.

3.1.11 Versorgung

Zur Versorgung des Systems muss ein Netz mit verschiedenen Spannungsebenen aufgebaut werden. Die vom System benötigten Spannungen könnten an einer zentralen Stelle des Systems erzeugt werden und über die System Busverbindung verteilt werden. Dieser Ansatz hätte den Vorteil, dass jede Spannung nur einmal erzeugt werden müsste. Dafür wären insgesamt weniger Bausteine für die Versorgung notwendig allerdings müssten diese umso mehr Strom liefern. Nachteil der zentralen Versorgung ist auch die potentielle Gefahr von Störungen die auf diesem Weg im System verteilt werden. Spannungsabfälle auf den langen Zuleitungen würden nicht ausgeregelt werden.

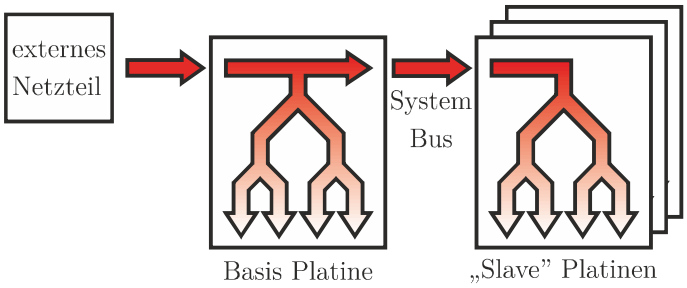

Abbildung 19: Schema der Verteilung der Versorgung auf den Platinen

Aus diesem Grund wird jede Platine die von ihr benötigten Spannungen selbst erzeugen. Nur die Versorgung vom externen Netzteil wird über den System Bus verteilt, siehe Abbildung 19.

Um bei der Wahl des externen Netzteils nicht eingeschränkt zu sein soll es möglich sein das System innerhalb eines großen Spannungsbereichs zu versorgen. Der Bereich soll von 12 bis 28 V reichen.

3.2 Software System Architektur

Das im Kapitel 3.1 definierte System verlangt die Erstellung von drei verschiedenen Softwarelösungen.

- Ein Design für die „Slave“ FPGAs
- Ein Design für das „Master“ FPGA
- Ein Software Packet für das Prozessormodul

3.2.1 „Slave“ FPGA Design

Im folgenden Kapitel wird ein Konzept eines FPGA Designs beschrieben welches in der Lage sein soll die Aufgaben der Problemstellung der „Slave“ Platinen zu lösen. Die Aufgaben werden an Module verteilt um sie einzeln behandeln zu können und sie ebenso zu implementieren.

Die Hauptaufgabe dieser FPGAs ist es die ADCs auszulesen und die gesammelten Daten für das „Master“ FPGA bereitzustellen.

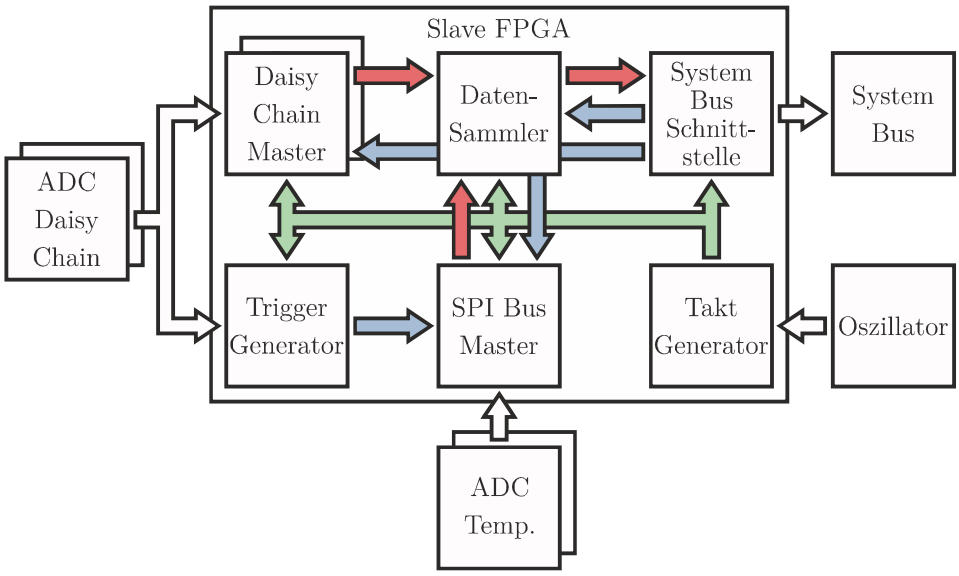

Abbildung 20: Schematische Darstellung der Module im „Slave“ FPGA

In Rot ist der Datenfluss der Messwerte dargestellt. Steuerleitungen sind Blau markiert, Taktleitungen Grün.

3.2.1.1 Takt Generator

Im System werden verschiedene Taktsignale benötigt. Ein Systemtakt, zwei verschiedene SPI Takte und zwei verschiedene Abtasttakte für die beiden ADC Systeme. Diese werden von einem externen Oszillator abgeleitet. Die Generierung der Takte erfolgt in diesem Modul.

3.2.1.2 „Daisy Chain Master"

Jede SPI Kette wird von einem solchen Modul gesteuert. Es sorgt dafür, dass die ADCs zum richtigen Zeitpunkt ausgelesen werden. Dafür muss die Logik für eine SPI Schnittstelle realisiert werden.

Neben dem Auslesen der Werte wird auch dafür gesorgt, dass diese weiterverarbeitet werden. Eine abgeschlossene Konversion soll intern durch ein Signal „Wert bereit" angezeigt werden um andere Module darüber zu informieren dass neue Daten bereitstehen.

3.2.1.3 SPI Bus „Master"

Dieses Modul dient zu Steuerung der langsamen ADCs. Die zur Realisierung dieses Moduls benötigte „state machine" ist wesentlich aufwendiger als die zur Steuerung der ADCs in der „daisy chain". Die Werte müssen nicht einfach nur periodisch nach jeder Konversion ausgelesen werden. Jeder ADC auf dem Bus muss initialisiert und nach jeder abgeschlossenen Konversion der Multiplexer des jeweiligen ADCs auf den nächsten Kanal eingestellt werden. Eine neue Konversion wird abhängig von einem Triggersignal gestartet werden.

3.2.1.4 Trigger Generator

Um die Messung der beiden Abtastsysteme zu synchronisieren wird in diesem Modul der Trigger für den SPI Bus „Master" vom Zeitpunkt der abgeschlossenen Konversion der „daisy chain" ADCs abgeleitet.

3.2.1.5 Datensammler

Da die Daten von außen asynchron ausgelesen werden können bietet es sich an, die Messwerte in einem „dual ported" RAM zu sammeln. Von der einen Seite kann über den System Bus direkt darauf zugegriffen werden. Die andere Seite wird von einer „state machine" befüllt, welche die Werte von den „daisy chain Master" Modulen und dem SPI Bus „Master" Modul holt. Dies wird durch die entsprechenden „Wert Bereit" Signale gestartet.

Die Werte müssen vom „Master" FPGA abgeholt werden nachdem der letzte aktuelle Wert in das RAM geschrieben worden ist und bevor der erste neue Wert geschrieben wird. Um dem „Master" dafür zumindest eine Periode des 32 kHz Taktes Zeit zu geben werden zwei Bereiche im RAM verwendet die als Puffer fungieren. Immer wenn ein Datensatz komplett ist kann die Adresse umgeschaltet werden und der eine Bereich

zum Auslesen freigegeben werden, währenddessen der andere bereits wieder mit den nächsten Werten befüllt wird.

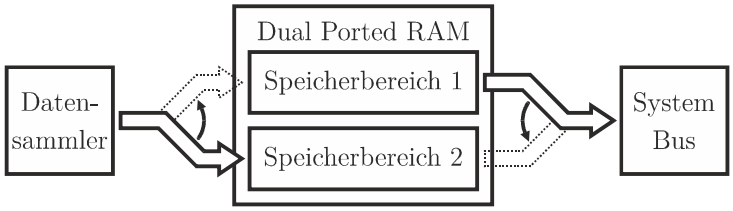

Abbildung 21: Schematische Darstellung des Puffers im „dual ported" RAM

Diese Methode erhöht die Komplexität des Schreibens der langsamen ADC Daten, da immer bedacht werden muss in welchen Speicherbereich geschrieben werden darf. Aus diesem Grund ist es sinnvoller zwei getrennte Datensammler zu realisieren. Die Schnittstelle des „dualported" RAMs die zu dem System Bus führt muss in diesem Fall über einen internen Bus zusammengeschaltet werden. Je nach angeforderter Adresse werden „chip select" Signale für die beiden Busteilnehmer generiert.

3.2.1.6 System Bus Schnittstelle

Diese Einheit wählt während des Systemstarts eine Adresse auf dem System Bus und ermöglicht danach dem „Master" FPGA den Zugriff auf das interne Bus System. Sie stellt die Verbindung zwischen dem „Slave" FPGA und dem System Bus dar.

Das Modul leitet die Steuerleitungen und den Abtasttakt für die Messung vom „Master" FPGA an den Datensammler sowie den SPI und den „daisy chain Master" weiter.

3.2.2 „Master“ FPGA Design

Die Hauptaufgabe dieses FPGAs ist das Abholen der Messwerte von den einzelnen „Slave“ FPGAs, diese mit Zeitstempel zu versehen und für das Auslesen durch das Prozessormodul vorzubereiten. Zusätzlich sollen sämtliche Hardwarekomponenten, welche eine Ansteuerung in Echtzeit benötigen, auf der Basis Platine gesteuert werden.

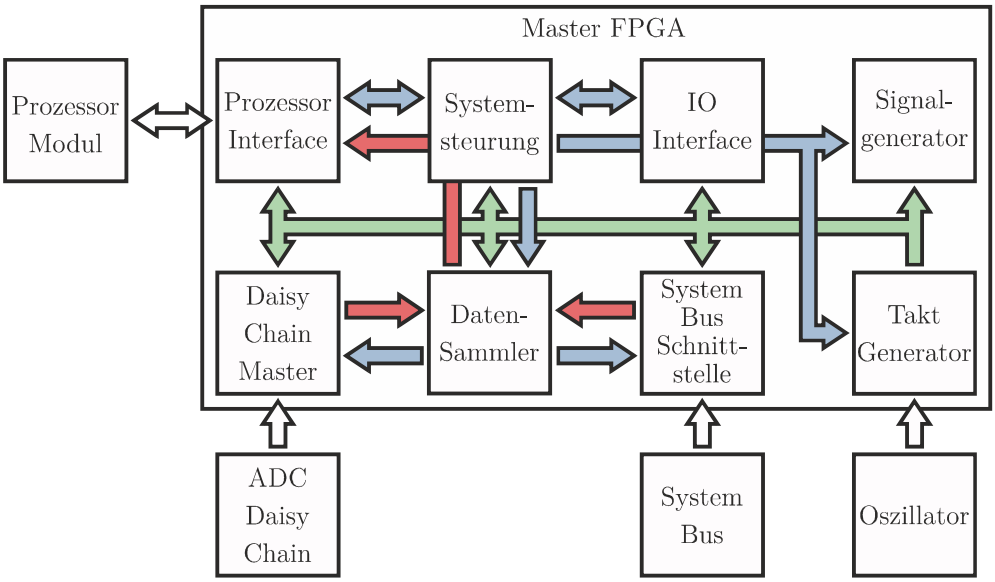

Abbildung 22: Schematische Darstellung der Module im „Master“ FPGA

In Rot ist der Datenfluss der Messwerte dargestellt. Steuerleitungen sind Blau markiert, Taktleitungen Grün.

3.2.2.1 Takt Generator

Ähnlich dem „Slave“ Design werden auch hier verschiedene Taktsignale benötigt, ein Systemtakt, ein SPI Takt und der Abtasttakt.

Als Systemtakt wird direkt der externe Oszillatortakt verwendet und der SPI Takt davon abgeleitet. Das „Master“ FPGA muss den Takt für das Abtasten erzeugen. Damit dieser genau ist muss ein entsprechender Oszillator, dessen Frequenz ein ganzzahliges Vielfaches der gewünschten Frequenz ist, eingesetzt werden.

3.2.2.2 Prozessormodul Interface

Dieses Modul ermöglicht den Zugriff auf den internen Bus für das Prozessormodul. Es ermöglicht dem Prozessor einen Zugriff auf einen externen Speicher.

Um das Design des internen Busses zu erleichtern soll dieser explizite Adress- und Datenleitungen besitzen. Mit Hilfe dieses Moduls soll die Architektur des externen Modul Busses in die des internen Busses übersetzt werden.

Der adressierbare Speicherbereich wird im FPGA auf verschiedene Module aufgeteilt. Aus der Adresse auf dem internen Bus kann ein „chip select“ Signal für den Zugriff auf die einzelnen Module abgeleitet werden.

3.2.2.3 Systemsteuerung und Statusregister

Damit das gesamte FPGA System gesteuert werden kann sollen dem Prozessor ähnlich dessen internen Peripherie Einheiten Steuerregister zur Verfügung stehen.

Der Prozessor beschreibt diese Register mit seinem Bustakt. Damit verhalten sie sich asynchron zu den restlichen Signalen. Darauf muss bei der Verwendung von Signalen die mit den Registern verknüpft sind geachtet werden.

Diese Register sind je nach Anwendung entweder nur lesbar oder les- und schreibbar implementiert.

3.2.2.4 “Daisy Chain Master”

Zur Verwendung der Master Platine als „stand alone“ System wird das Master FPGA ebenfalls Messkanäle auslesen können. Dabei wird es sich um wenige Strom- und Spannungskanäle handeln. Hierfür kann grundsätzlich dasselbe „Daisy Chain Master“ Modul wie in den Datensammlern verwendet werden.

3.2.2.5 Datensammler

Der Datensammler des „Master“ FPGAs hat die Aufgabe nach jeder Abtastung die Werte von allen Kanälen aller „Slave“ FPGAs in einem „dualported“ RAM abzulegen. Dies soll bereits in einem zur Sendung über die Ethernet Schnittstelle des Prozessormodules vorbereiteten Datenrahmen geschehen. Die Werte der Temperaturmessung werden gesondert in einem eigenen Datenrahmen gesammelt.

Beim Zwischenspeichern der Daten kommt es zu demselben Problem wie zuvor bei den „Slave“ FPGAs, siehe 3.2.1.5. Im Fall des „Master“ FPGAs ist die gleiche Lösung nicht sinnvoll da das Prozessormodul mit entsprechend hoher Geschwindigkeit die Daten auslesen können muss. Der zusätzliche Bus zwischen den RAMs würde zusätzliche Gatterlaufzeiten mit sich bringen. Aus diesem Grund werden hier sämtliche Daten in einem RAM gespeichert. Dieses RAM wird in vier Bereiche aufgeteilt, je zwei Bereiche für die Strom- und Spannungsmessung und zwei für die Temperaturmessung. Eine entsprechend aufwendige Logik für die Adressierung ist notwendig um die Pufferung für beide Arten der Datenrahmen zu realisieren.

3.2.2.6 Signalgenerator

Die Berechnung der DDS Kanäle mit den in Registern hinterlegten Parametern wird von diesem Modul durchgeführt. Als Grundlage dieses Moduls wird ein DDS „IP Core“ dienen.

Die berechneten Werte werden über einen SPI Bus in den DAC geschrieben. Über denselben SPI Bus werden zu Beginn der Signalerzeugung der externe DDS sowie der multiplizierende DAC konfiguriert.

Die Signalgenerierung soll im Falle eines Überstromes schnell beendet werden. Die externe gesteuerte Stromquelle ist in der Lage eine solche Situation zu erkennen und daraufhin den Pegel eines ihrer digitalen Ausgänge zu wechseln. Dies muss vom FPGA erkannt und entsprechend behandelt werden.

3.2.2.7 IO Interface

Das System soll in der Lage sein eine Reihe von digitalen Ein- und Ausgängen bedienen zu können. Einer der Eingänge ist für die Erkennung eines Überstromes der externen Stromquelle vorgesehen. Der Rest ist für zukünftige Anwendungen vorgesehen um damit andere externe Komponenten bedienen zu können.

Das Lesen der Eingänge sowie das Schreiben der Ausgänge soll durch dieses Modul ermöglicht werden. Es soll durch eigene Register gesteuert werden.

3.2.2.8 System Bus Schnittstelle

Dieses Modul ist das Gegenstück zu jenem der „Slave“ FPGAs. Es übersetzt den internen Zugriff des Datensammlers auf den System Bus.

Der Adressenverteilungsvorgang wird von diesem Modul aus gestartet indem die Adresse für den ersten Teilnehmer der Adressierungskette ausgegeben wird und der „valid“ Pin nach einer kurzen Wartezeit gesetzt wird.

3.2.3 Software Prozessormodul

Die Software des Prozessormoduls übernimmt die Aufgabe der Kommunikation mit dem „Client“ Rechner, die Einstellung der Parameter am „Master“ FPGA sowie das Auslesen und Verschicken der Messdaten.

Dem PC müssen zwei Schnittstellen geboten werden. Eine Schnittstelle zur Konfiguration und eine zur Übermittlung der Messwerte. Die Anforderungen an diese beiden Kommunikationskanäle sind verschieden.

Die Konfiguration stellt keine Ansprüche an die Geschwindigkeit. Sie soll einfach zu implementieren und zu erweitern sein. Es bietet sich in diesem Fall TCP an mit dem Vorteil einer gesicherten Verbindung. Aufsetzend auf dieser Verbindung muss ein Protokoll implementiert werden. Dieses Protokoll kann auf Grund der mäßigen Geschwindigkeitsanforderung und der hohen Bandbreite der Ethernet Schnittstelle als Klartextnachrichten implementiert werden. Die einfache Erweiterbarkeit kann durch den Einsatz einer „Markup“ Sprache erreicht werden. XML eignet sich dafür ausgezeichnet. Damit kann auf bereits verfügbare Bibliotheken aufgesetzt werden. Die Implementierung eines eigenen „Parser“ kann somit entfallen.

Bei der Datenverbindung für die Messwerte kommt es vor allem auf die Geschwindigkeit an. Da die anfallende Datenmenge von etwas mehr als acht Mega Byte pro Sekunde bei 64 Kanälen bringt die 100 Mbit Ethernet Schnittstelle bereits nahe an ihre praktische nutzbare Bandbreite. Aus diesem Grund soll während der Messung die Konfigurationsverbindung getrennt werden. Um keine Bandbreite an andere Netzwerkteilnehmer zu verlieren ist es vorgesehen das Gerät immer direkt mit dem „Client“ Rechner zu verbinden. Eine gesicherte Übertragung spielt in diesem Fall keine Rolle da einiges an Zeit nötig wäre um ein verloren gegangenes Paket neu anzufordern. Darum kann das verbindungslose Protokoll UDP verwendet werden. Der „Overhead“ zu den Nutzdaten soll so gering wie möglich sein, es empfiehlt sich einen UDP Datenrahmen möglichst gut auszunützen. Das bereits im FPGA zusammengestellte Paket soll 1000 Bytes nicht überschreiten. Dies soll sicherstellen, dass jedes UDP Paket, nicht fragmentiert, mit einem einzigen IP Paket übertragen wird. Die maximale „Payload“ Größe von einem IP Rahmen beträgt 1500. [12] Sollten Bytes frei bleiben hat das den Vorteil, dass noch Platz für zukünftige Erweiterungen vorhanden ist.

Die Trennung der beiden Kommunikationskanäle bringt zusätzlich den Vorteil dass die Konfiguration und das Empfangen der Messdaten an verschiedene Teilnehmer im Netzwerk vergeben werden kann.

Die Schnittstelle zum „Master“ FPGA wird über externe Speicherzugriffe realisiert. Es ist notwendig diesen Speicherbereich im Linux Kernel entsprechend zu registrieren. Dafür muss ein „Platform Device Driver“ geschrieben werden. Dieser Treiber ist dafür zuständig den externen Speicher „controller“ zu konfigurieren, den externen Speicherbereich einem physikalischen Speicherbereich zuzuweisen und diesen wiederum in den virtuellen Adressraum abzubilden. Der auf dem Modul verfügbare Linux Kernel 3.2 bietet noch nicht die Möglichkeit der Plattformkonfiguration über einen sogenannten „Device Tree“. [13] Es muss ein zur Plattform gehörendes „Board File“ existieren welches die Registrierung der „Platform Device Driver“ vornimmt.

Die Aufgaben der Software können in drei Bereiche gegliedert werden.

- Eine Hardware Abstraktionsebene für die Initialisierung und den direkten Hardware Zugriff.
- Eine darauf aufbauende Mittelschicht, welche die hardwarenahen Funktionen weiter abstrahiert, damit eine Applikation die Hardware steuern kann.
- Eine Applikationssoftware welche die Netzwerkkommunikation und die Ablaufsteuerung übernimmt.

Die ersten beiden Bereiche können von einem Kernel Treiber abgedeckt werden. Auf diesen Treiber kann dann eine „user space“ Software zugreifen. Damit ist das eigentliche Messsystem von der Kommunikation mit dem „Client“ Computer getrennt. Wird der Treiber in Form eines „Character Device“ Treibers implementiert steht eine klare Schnittstelle über Datei Operationen zur Verfügung. Diese Trennung der Software in ein „Back End“ für die eigentliche Messung und in ein „Front End“ zur Steuerung des Systems ermöglicht es die beiden Module getrennt voneinander zu entwickeln und zu testen. Das „Back End“ kann bei Bedarf ohne „Front End“ über die Kommando Zeile bedient werden. Sollte es für zukünftige Anwendungen nötig sein eine andere Art der Kommunikation zu verwenden könnten verschiedene „Front Ends“ je Bedarf gestartet werden.

Das Versenden der Daten mit acht Mega Byte pro Sekunde bringt die 100 Mbit Ethernet Schnittstelle nahe an ihre praktische Grenze. Es ist darum wichtig den Ethernet Kontroller so schnell wie möglich mit Daten zum Senden zu versorgen. Das Transferieren der Daten aus dem Kernel Modul zum „user space“ Programm kann im Prinzip ohne Zeitverlust erledigt werden indem das Programm dem Treiber eine Speicheradresse mitteilt und die Messwerte direkt an diese geschrieben werden. Allerdings findet die Netzwerkkommunikation auf einer höheren Abstraktionsebene als im „Kernel Space“

statt. Darum ist es sinnvoller den UDP Client in den Kernel Treiber zu verschieben. Dem Treiber muss nur die IP Adresse des UDP Servers bekannt gegeben werden und dieser kann dann die Daten so direkt wie möglich versenden, siehe Abbildung 23.

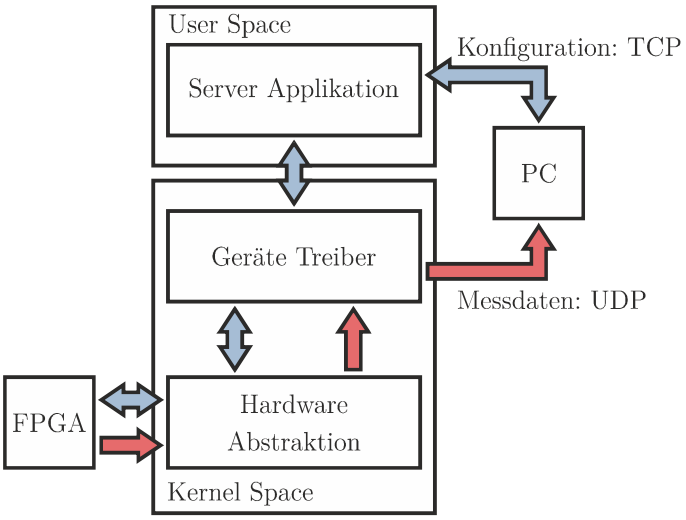

Abbildung 23: Schema des Software Konzeptes für das Prozessor Modul

Blau zeigt den Pfad der Konfigurationsinformation und Rot den der Messdaten.

4 Hardware

Dieses Kapitel beschreibt die Entwicklung und Implementierung der einzelnen Hardware Komponenten. Ziel ist die Erstellung von Schaltplänen und Layouts für zwei Platinen im Format 160 mm mal 100 mm, welche einseitig bestückt und maximal vierlagig sind.

4.1 CAD Tool

Für die Entwicklung der Hardware wird die CAD Software EAGLE V5 der Firma CadSoft verwendet. Diese Software bietet einen Schaltplan Editor sowie einen Layout Editor. Layout und Schaltplan werden konsistent gehalten.

Neben den Editoren wird die Möglichkeit geboten Bauteilbibliotheken mit Schaltsymbolen und den zu gehörigen Bauformen anzulegen.

Zur Simulation einzelner Schaltungsteile wird der LTSPICE Simulator verwendet. Dieser Simulator wird von der Firma Linear Technology gratis angeboten.

4.2 Implementierung

In den folgenden Abschnitten wird die Realisierung einzelner Hardwareeinheiten auf der „Slave“ sowie der Basis Platine beschrieben.

4.2.1 „Slave“ Platine

Eine „Slave“ Platine realisiert im Prinzip zwei getrennte Systeme, eines je FPGA. Um das gleiche Design in beiden FPGAs verwenden zu können sind beide Systeme absolut symmetrisch aufgebaut. Einzige Ausnahme ist ein „pull up“ beziehungsweise „pull down“ Widerstand zum Festlegen der System Bus Adresse.

4.2.1.1 Strom- und Spannungsmessung

Die Spannung wird zwischen den inneren Kontakten der beiden Sensorplatten in der Brennstoffzelle gemessen. Der Strom wird über die Shunt Widerstände einer Sensorplatte gemessen, siehe 3.1.1.

Als ADC wird der AD7767-2 verwendet. Dieser besitzt einen differentiellen Eingang, ist ein SAR ADC der mit bis zu 1,024 MHz abtastet, rechnet intern einen FIR Filter zur Dezimation um den Faktor 32 und kann über SPI in einer „daisy chain“ Konfiguration ausgelesen werden. Der ADC zeigt über einen zusätzlichen Pin an dass eine Konversion abgeschlossen ist. Da alle ADCs der Kette mit dem gleichen Abtasttakt verbunden sind werden alle zum selben Zeitpunkt fertig. Es ist daher nur bei einem ADC nötig diese Leitung zum FPGA zu legen.

Jeder Eingang wird mit einer ESD Schutzdiode und einem Serienwiderstand beschaltet. Bevor das Signal verstärkt wird passiert es eine analoge Filterstufe. Der darauf folgende Verstärker hat drei Aufgaben, die Anpassung der Amplitude, Filterung und „single ended to differential“ Konversion.

Als Operationsverstärker wird der Baustein ADA4941 verwendet der zur Konversion von „single ended“ Signalen einen zweiten fix als Inverter beschalteten Operationsverstärker enthält.

Der erste Operationsverstärker wird als Differenzverstärker beschaltet, siehe Abbildung 25. Die Referenz Spannung der ADCs wird auf 2,5 V gelegt. Die zu erwartende Eingangsspannung bei der Spannungsmessung liegt bei ungefähr einem 1 V Gleichanteil, dies entspricht der maximalen Spannung einer Zelle, hinzu kommt der eingeprägte Wechselanteil im Bereich von 1 – 100 mV. Der Eingangsbereich des ADCs wird so betrachtet mit einem Verstärkungsfaktor von zwei gut ausgenützt.

Durch die Konversion des Signals mit Hilfe des Inverters, welcher direkt das Ausgangssignal des Differenzverstärkers invertiert, ist das differentielle Ausgangssignal um den Faktor zwei größer als das Eingangssignal. Der Differenzverstärker selbst darf daher das Signal nicht verstärken, siehe Abbildung 24.

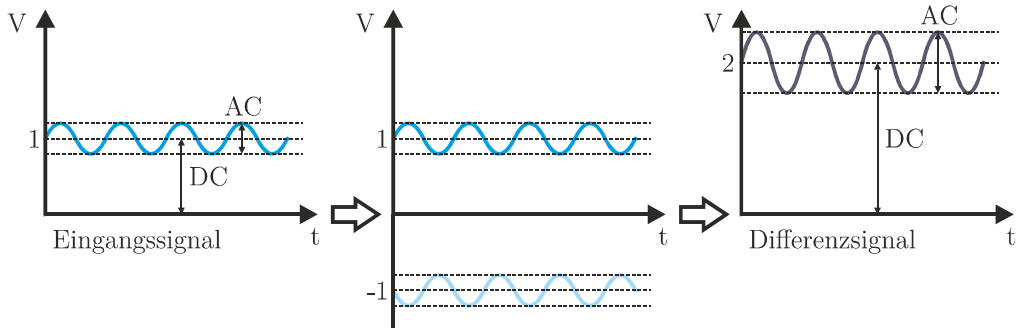

Abbildung 24: Konversion des Eingangssignals in ein Differenzsignal

Der Verstärkungsfaktor der Stromkanäle kann wesentlich höher gewählt werden, da der überlagerte DC-Anteil des konstant fließenden Stromes geringer ausfällt. Die zu messende Spannung wird wegen des kleinen Shunt Widerstandes (im Milliohmbereich) im Bereich einiger zehn Millivolt liegen. Gewählt wurde eine Verstärkung von 100, dies entspricht einem Verstärkungsfaktor von 50 für den Differenzverstärker.

Die Einstellung der Verstärkungsfaktoren erfolgt mit 0.1 % Widerständen. Der Widerstandswert (R_2, R_4) an beiden Eingängen des Differenzverstärkers wird auf mehrere Widerstände verteilt. Damit wird vor allem bei der Strommessung der Einfluss der Leitungswiderstände zu den Shunt Widerständen für die Verstärkung berücksichtigt.

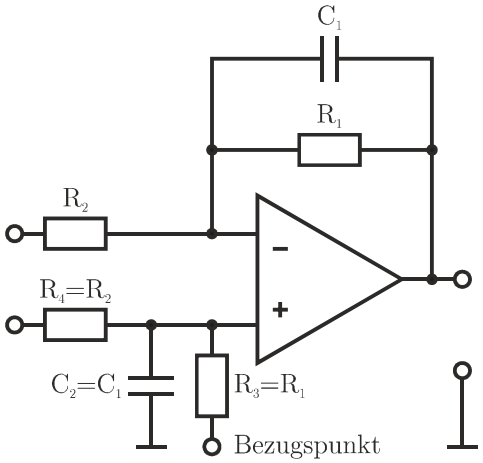

Abbildung 25: Schaltbild des Differenzverstärkers

Die Widerstände R_1 und R_3 der Differenzverstärker werden für die Strom- und Spannungsmessung gleich groß gewählt. Diese bestimmen neben der Verstärkung zusammen mit den ihnen parallelgeschalteten Kondensatoren C_1 und C_2 den Frequenz- und Phasengang. Mit $R_1 = R_3$ und $C_1 = C_2$ kann die Grenzfrequenz (f_G) des Filters mittels der Formel für einen Tiefpass Filter 1. Ordnung berechnet werden. [9]

$$f_G = \frac{1}{2\pi R_1 C_1}$$

Damit der Frequenz- und Phasengang für die Strom- und Spannungsmessung gleich verläuft wird C_1 und C_2 bei allen Kanälen gleich gewählt.

Das Ausgangssignal des Differenzverstärkers und des Inverters kann prinzipiell negativ werden. Aus diesem Grund müssten die Operationsverstärker bipolar versorgt werden. Die negative Versorgungsspannung kann eingespart werden in dem der Nullpunkt des Ausgangssignals um den halben Aussteuerbereich, in diesem Fall 1,25 V, angehoben wird. Dies wird durch eine Anhebung des Bezugspunktes des Ausgangsignals um 1,25 V erreicht.

Bei den verwendeten Operationsverstärkern handelt es sich grundsätzlich um „rail to rail“ Typen. Um sicher zu gehen, dass es ganz in der Nähe der Aussteuergrenzen zu keinen Problemen, wie nicht linearen Verzerrungen, kommt wird der ADA4941 mit einer etwas höheren positiven Versorgung betrieben. Für die negative Versorgung wird eine kleine negative Hilfsspannung verwendet. Der Baustein wird anstelle der notwendigen 0 und 2,5 V mit -0,3 und +2,8 V versorgt. Damit ist sichergestellt dass dieser auch wirklich den vollen Bereich von 0 bis 2,5 V aussteuern kann.

Die Versorgung jeder ADC und Verstärker Einheit wird lokal von 100 nF Kondensatoren gestützt. Dies geschieht um eine gegenseitige Störung der Einheiten, über die Versorgungsleitungen, möglichst gut zu unterdrücken.

4.2.1.2 Temperaturmessung

In den Sensorplatten befindet sich neben den Shunt Widerständen jeweils ein Temperaturfühler aus Kupfer. Die Temperatur wird durch die Widerstandsänderung des Kupfers bestimmt, siehe 3.1.1.

Um diese relative kleine Widerstandsänderung im Milliohm Bereich erfassen zu können findet die Messung in Vierleitertechnik statt. Wird ein Strom in die Serienschaltung eingespeist kann der Spannungsabfall der einzelnen Fühler an den Abgriffen gemessen werden. Ist der eingeprägte Strom bekannt kann über die Spannung auf den Widerstandswert zurückgerechnet werden.

Um die Spannung über den einzelnen Fühlern messen zu können wird ein ADC mit differentiellen Eingängen benötigt. Verwendet wird der AD7794 mit sechs differentiellen Eingängen. Für die 16 Kanäle wären drei Einheiten ausreichend. Um das System symmetrisch für beide FPGAs aufbauen zu können werden vier ADCs verwendet, zwei pro FPGA.

Als Referenz für die ADCs wird der Spannungsabfall über einem Referenzwiderstand, der vom selben Strom wie die Fühlerkette durchflossen wird, verwendet. Damit ergibt sich eine ratiometrische Messung. Der Widerstandswert der Fühler kann in diesem Fall direkt vom ADC Ergebnis abgeleitet werden. Das Ergebnis der Konversation entspricht dem Verhältnis des Fühler Widerstandes zum Referenzwiderstand.

Jeder Eingang ist mit einer ESD Schutzdiode und einem RC Tiefpass beschaltet.

4.2.1.3 Potential Trennung

Wie in 3.1.10 beschrieben sollen die SPI Leitungen zu den ADCs potentialgetrennt werden. Für die SPI Leitungen zu den „daisy chain“ ADCs müssen Signale mit Bitraten um 10 Mbits pro Sekunde übertragen werden. Für die Übertragung von digitalen Signalen in diesen Frequenzbereichen gibt es Magentkoppler Bausteine. Die Wahl fiel auf ICs des Typs ADUM14xx. Die schnellste Variante dieser Magnetkoppler ist in der Lage ein Signal mit 90 Mbits pro Sekunde zu übertragen. Allerdings verzögern diese Bausteine das Signal um 18 ns.

Bei einem SPI Takt von 10 MHz beträgt die Periodendauer 100 ns. Die Verzögerung durch den Magnetkoppler wirkt sich bei der SPI Kommunikation doppelt aus. Einmal bei der Übertragung des Taktes hin zur potentialgetrennten Seite und ein zweites Mal

bei der Rückübertragung des MISO Signals. Das bedeutet dass in Summe um 36 ns verzögert wird. Dies stellt bereits mehr als ein Drittel der Periode dar und darf beim Abtasten der MISO Leitung nicht ignoriert werden.

Zusätzlich zu den SPI Leitungen werde der Abtasttakt für die ADC Systeme sowie die Leitungen zum Anzeigen einer abgeschlossenen Konversion übertragen.

Die Versorgung der potential getrennten Seite wird über DC/DC Wandler realisiert, siehe 4.2.1.4.

4.2.1.4 Versorgung

Die Platine wird über den System Bus Stecker über die Basis Platine von einem externen Netzteil versorgt.

Die größten nicht potential getrennten Verbraucher sind die beiden FPGAs. Diese benötigen drei verschiedene Versorgungsspannungen: 3,3 V, 2,5 V und 1,2 V. Die 3,3 V werden mit Hilfe eines „step down“ Konverters aus der Versorgung erzeugt. Verwendet wird ein LM25576 Schaltregler. Dieser besitzt einen großen Eingangsspannungsbereich und kann einen Strom von bis zu 3 A liefern. Der Eingangsspannungsbereich wird auf 8 bis 32 V festgelegt. Die Kondensatoren am Eingang des Schaltreglers sind mit einer entsprechend hohen Spannungsfestigkeit von 50 V gewählt. Dieser Spannungsregler bietet einen „Power Down“ Pin. Dieser Pin wird zum System Bus Stecker geführt damit die Basis Platine die „Slave“ Platinen nach Bedarf ein- und ausschalten kann.

Die anderen beiden FPGA Spannungen werden durch zwei hintereinander geschaltete Längsregler mit fixer Ausgangsspannung erzeugt.

Zur Versorgung der Messkanäle kommen zwei parallel geschaltete DC/DC Wandler, die als „step up“ Konverter fungieren, zum Einsatz. Diese erzeugen aus den 3,3 V potential getrennt 5 V. Damit stehen 5 V, zur Erzeugung der Versorgungsspannungen der Messkanäle, zur Verfügung. Diese benötigen Versorgungen mit 3,3 V, 2,8 V, 2,5 V sowie ein kleine negative Hilfsspannung. Neben der Versorgung müssen auch die Referenzspannungen 2,5 V und 1,25 V erzeugt werden.

Zur Generierung der negativen Hilfsspannung wird der negative Ausgang des DC/DC Wandlers nicht direkt als Masse verwendet sondern sein Potential um den Spannungsabfall einer Shottky Diode angehoben. Dadurch stehen gegenüber Masse in etwa +4,7 V und -0,3 V bereit.

Aus den 4,7 V wird durch einen Längsregler 3,3 V erzeugt. Zwei weitere hintereinander geschaltete Längsregler werden für 2,8 V und 2,5 V verwendet. Beide werden von den 4,7 V abgeleitet. Es werden allerdings keine Längsreger mit fixer Ausgangspannung

verwendet sondern der Typ LT3080 mit einstellbarer Ausgangsspannung. Dieser Regler benötigt ein externes „Feedback“ von seinem Ausgang. Mit Hilfe dieses Feedbacks ist eine „Tracking“ Funktion realisiert. Jede der beiden Versorgungen fährt gemeinsam mit der Ausgangsspannung des Längsreglers für die nächst höhere Spannung hoch. Dadurch ist gewährleistet, dass die drei Versorgungen in etwa zu gleichen Zeit bereit sind.

Als 2,5 V Referenz wird der Baustein AD421 verwendet. Von ihm werden durch einen Spannungsteiler mit 0,1 % genauen Widerständen 1,25 V abgeleitet und mit einem Operationsverstärker als Spannungsfolger gepuffert.

4.2.1.5 Anschlüsse

Um die Platine mit der Sensorplatte zu verbinden wird wie in 3.1.9 bereits beschrieben eine zusätzliche Adapterplatine verwendet. Zu dieser Platine müssen je Messkanal vier Leitungen geführt werden. Bei 16 Kanälen sind das bereits 64 Leitungen plus 19 für die Temperaturmessung. Als einfach handhabbarer Verbinder werden Messer/Feder Leisten (DIN 41.612) verwendet.

Der System Bus benötigt 12 Adressleitungen, fünf davon alleine zum Auswählen des gewünschten Teilnehmers, 16 Daten Leitungen sowie 2 Steuerleitungen. Zusätzlich soll zwischen den Leitungen immer eine Masseleitung geführt werden. Damit benötigt der System Bus alleine bereits 60 Leitungen. Weitere 10 Leitungen werden unter anderem für die Zuordnung der Adresse zu Systemstart, der Kontrolle des Messvorganges und als Reserve vorgesehen. Die Versorgung wird auf 8 Leitungen übertragen. In Summe werden mindesten 78 Leitungen benötigt.

Die Platinen sollen als Stapel zusammengesteckt werden können, darum müssen der Stecker und sein Gegenstück übereinander auf beiden Seiten einer Platine bestückt werden können. Für solche Anwendungen gibt es hermaphrodite SMD Steckverbinder. Verwendet wird ein Verbinder der Firma SamTec vom Typ LSS mit 100 Kontakten. Diese Stecker sind für verschiedene Abstände zwischen den Platinen im zusammengesteckten Zustand verfügbar. Damit die Messer/Feder Leisten Platz haben wird die Variante mit dem größten Abstand verwendet, dieser beträgt 12 mm. Bei der Auswahl der Bauteile für die „Slave“ Platine musste dieses Limit beachtet werden.

4.2.1.6 Layout

Das Layout der „Slave“ Platinen war aufgrund der hohen Dichte von Bauteilen bei der Realisierung zeitaufwändig.

Für die 16 Messkanäle sind für die Strom- und Spannungsmessung 32 ADCs inklusive Vorverstärker auf der Platine unter zu bringen. Dafür wurde ein Konzept ähnlich

dem Layout von digitalen Schaltungsteilen in ICs verfolgt. Jeder ADC mit seinem Verstärker stellt ein Modul dar. Die Bauteile eines solchen Moduls werden so platziert dass sie aneinander gereiht werden können. Dabei wird darauf geachtet dass die Versorgungen immer auf der gleichen Höhe kontaktiert werden können. Auf diese Art kann man die Module mit Versorgungsschienen, welche auf unteren Lagen verlaufen, verbinden, siehe Abbildung 26.

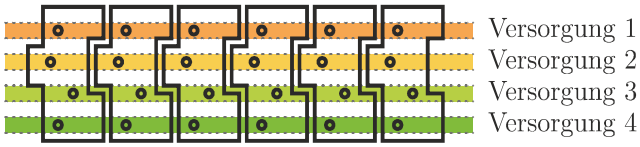

Abbildung 26: Beispiel eines modularen Layouts mit Versorgungsschienen

Das gleiche gilt für die SPI Leitungen. Die digitalen Leitungen wurden dabei möglichst weit entfernt von den Versorgungen auf einer Seite platziert um Störungen gering zu halten. Auf einer Platine sitzen vier Stränge von jeweils acht dieser ADC Module. Zwei dieser Stränge können mit der Seite ihrer digitalen Anschlüsse zusammenstehen. Dadurch ist der größte mögliche Abstand zwischen den SPI Leitungen und den analogen Versorgungs- und Signalleitungen gegeben, siehe Abbildung 28.

Die Platzierung der FPGAs ist so gewählt, dass alle Leitungen, die Signale mit hoher Frequenz führen, möglichst kurz sind. Die Leitungen für den System Bus sind für beide FPGAs genau gleich lang um Laufzeitunterschiede zu vermeiden.

Für die Potentialtrennung ist die Platine in zwei Bereiche aufgeteilt. Zwischen den Bereichen wurde versucht einen Abstand von 2 mm nicht zu unterschreiten. Aufgrund der hohen Bauteildichte gibt es einen Abschnitt in dem sich beide Bereiche überlappen. Die Trennung in diesem Bereich wird erreicht in dem die Signale des jeweiligen Bereichs entweder nur auf den beiden Lagen der Oberseite der Platine oder der Unterseite geführt werden.

Für die Längsregler und den Schaltregler sind große Kühlflächen vorgesehen um deren Verlustleistung in Form von Wärme abführen zu können.

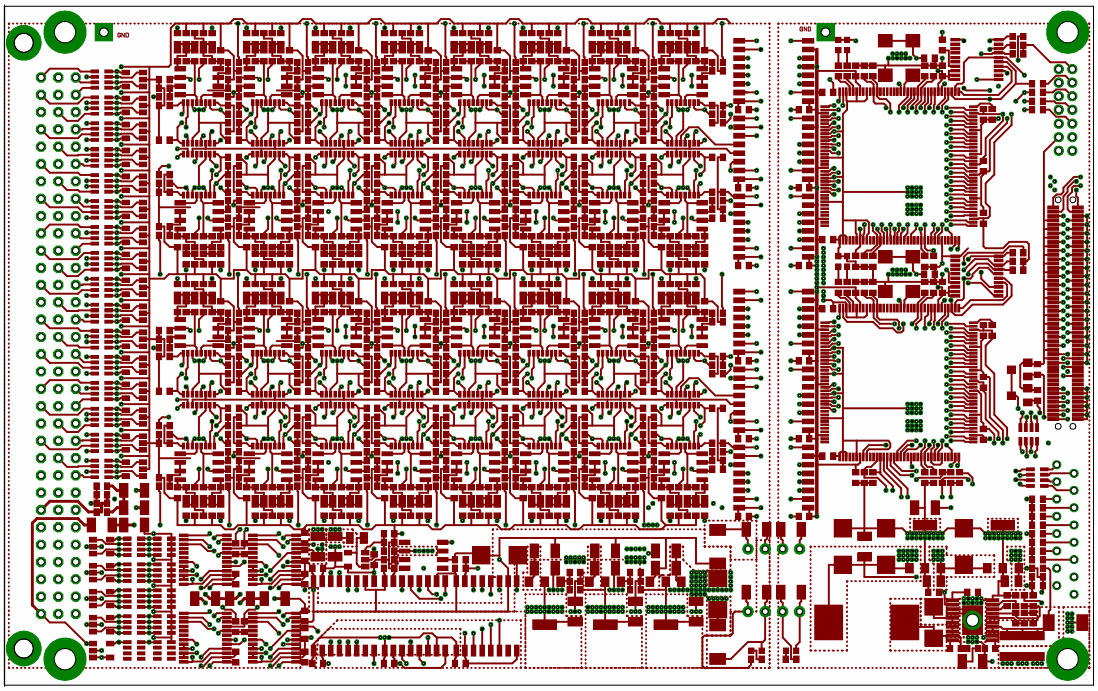

Abbildung 27: Layout der Oberseite der „Slave“ Platine

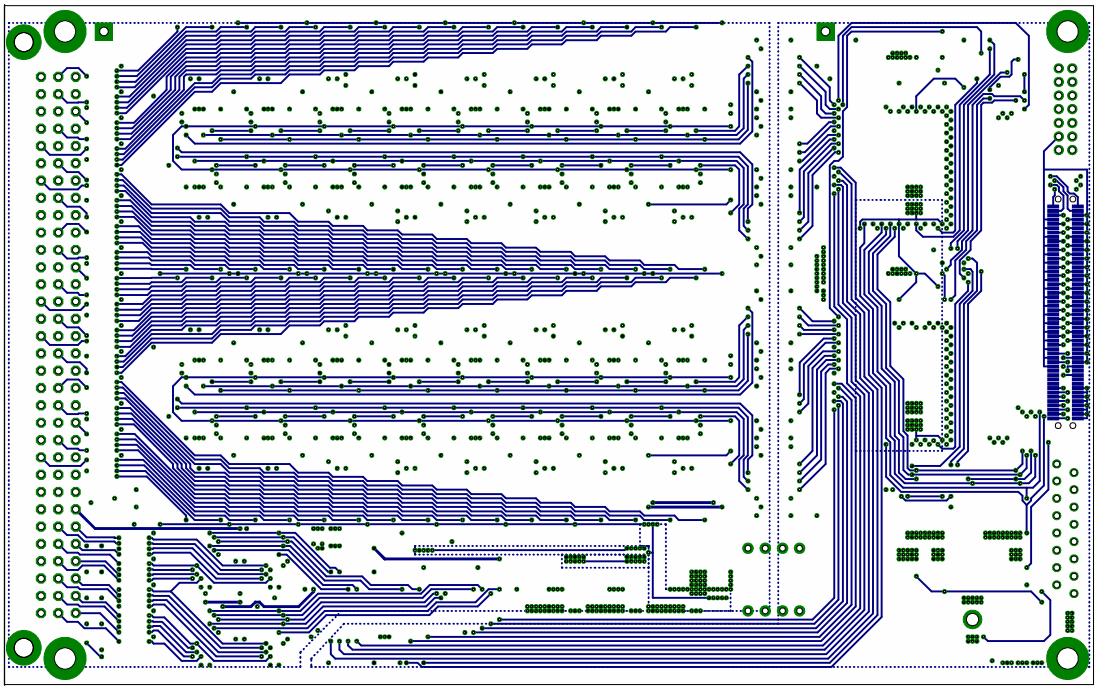

Abbildung 28: Layout der Unterseite der „Slave“ Platine

Es wurde versucht die inneren beiden Lagen ausschließlich für die Versorgung der Schaltung zu reservieren. Eine Lage dient als fast durchgängige Massefläche und die andere verteilt die verschiedenen Spannungen der Versorgung auf der Platine.

4.2.2 Basis Platine

Die Basis Platine realisiert für sich ein System das in der Lage ist ein Signal mit bis zu vier Frequenzkomponenten zu erzeugen und auf zwei Kanälen Strom und Spannung zu messen. Dieses System baut auf einem embedded Linux Modul mit Ethernet Anbindung auf und kann durch „Slave“ Platinen um zusätzliche Messkanäle erweitert werden.

4.2.2.1 Prozessor Modul

Auf Grund der verfügbaren Rechenleistung, der Peripherie, zum Beispiel einem externen Speicher Kontroller, und der benötigten Schnittstellen Ethernet und RS232 fiel die Wahl des Prozessor Moduls auf das phyCore-AM355x. Basis dieses Moduls ist der ARM Cortex A8 Prozessor AM3559 Sitara, der mit einem Takt von 800 MHz arbeitet.

Dem Prozessor stehen 512 MB DDR3 RAM Arbeitsspeicher sowie 512 MB NAND Flashspeicher zur Verfügung.

Das Modul wird über zwei hochpolige Steckverbinder mit der Trägerplatine verbunden. Diese ermöglichen den Zugriff auf fast jeden Pin des Prozessors.

4.2.2.2 Messkanäle

Zur Verwendung der Basis ohne „Slave“ Platinen sind zwei Kanäle zur Impedanz Messung, wie in 4.2.1.1 beschrieben, auf dieser Platine verbaut.

Die ADCs der beiden Kanäle werden jeweils zu einer eigenen „daisy chain“ zusammen geschaltet. Die beiden Kanäle sind potential getrennt voneinander aufgebaut.

Im Gegensatz zu beiden SPI Ketten auf den „Slave“ Platinen, welche von einem „Slave“ FPGA bedient werden, sind nicht alle Leitungen getrennt zum „Master“ FPGA geführt. Um Pins des „Master“ FPGAs zu sparen werden alle Leitungen bis auf die beiden MISO Ausgänge zusammengefasst.

4.2.2.3 Signalerzeugung

Wie in 3.1.6 beschrieben kommt ein DDS IC für die schnelle Inbetriebnahme zum Einsatz. Die Wahl fiel auf den AD9833. Dieser kann mit einer Ausgaberate von bis zu 25 MHz betrieben werden. Da bereits ein genauer Takt von 1,024 MHz für die ADCs erzeugt werden muss wird dieser auch für den DDS verwendet werden. Der DDS Baustein kann mit seinem 28 Bit Phasenakkumulator den gewünschten Frequenzbereich abdecken und wird über eine SPI Schnittstelle konfiguriert. Der AD9833 liefert ein Sinussignal mit Gleichspannungsoffset. Der DC Anteil des Ausgangssignals wird mit Hilfe eines Kondensators im Signalpfad geblockt. Der AC Anteil wird nach dem Kondensator

mit einem Spannungsfolger gepuffert. Dieses gepufferte Signal stellt eines der beiden DDS Signale dar.

Das zweite DDS Signal kommt von dem durch das FPGA gesteuerten DAC. Dafür wird der AD5542, ein 16 Bit ADC, verwendet. Neben der SPI Schnittstelle bietet dieser IC einen zusätzlichen Eingang mit dem das Aktualisieren des ausgegebenen Wertes unabhängig von der SPI Kommunikation gesteuert werden kann. Dieser ADC benötigt einen externen Operationsverstärker um ein Signal ausgeben zu können. Dieser Operationsverstärker wird so beschalten dass ein bipolares Signal erzeugt werden kann. Null entspricht in diesem Fall dem halben Aussteuerbereich des DACs.

Beide DDS Signale werden zu einem „Sallen Key“ Filter geführt. Dieser Filter dient gleichzeitig als Summierer für die beiden Signale.

Zur Einstellung der Amplitude des erzeugten Signals wird dem Filter ein multiplizierender DAC nachgeschaltet. Verwendet wird hierfür der AD5452 welcher einen vier Quadranten Betrieb zulässt. Die eigentliche Ausgangsgröße dieses DACs ist Strom. Erst mit Hilfe eines externen Operationsverstärkers, der als I zu U Konverter arbeitet, wird ein Spannungssignal erzeugt. Dieser DAC bietet eine Auflösung von 16 Bit, was eine sehr feine Einstellung der Amplitude ermöglicht.

Das erzeugte Signal soll in differenzieller Form ausgegeben werden. Darum invertiert ein zusätzlicher Operationsverstärker das Ausgangssignal.

Alle einstellbaren Komponenten für die Signal Erzeugung werden über einen SPI Bus mit verschiedenen „chip select“ Leitungen angesprochen. Um den DAC durch das FPGA für die Ausgabe von Signalen im Bereich von zehn Kilohertz entsprechend schnell beschreiben zu können muss dieser SPI Bus mit einer hohen Taktrate betrieben werden.

4.2.2.4 Digitale Ein- und Ausgänge

Die digitalen Ein- und Ausgänge sollten durch Pins des FPGAs realisiert werden. Wie sich herausgestellt hat werden die meisten Pins bereits für den Modul Bus und den System Bus benötigt. Um Pins einzusparen wird ein SPI IO Expander der Firma Microchip verwendet. Damit werden Anstelle von acht Pins nur vier für die Kommunikation mit diesem Baustein benötigt.

4.2.2.5 Potential Trennung

Die Potentialtrennung findet ähnlich der „Slave“ Platine, wie bereits in 4.2.1.3 beschrieben, statt.

Der wesentlich höhere SPI Takt für die Komponenten der Signalerzeugung führt zu keinem zusätzlichen Problem mit der Verzögerung der Magnetkoppler da vom FPGA nur Daten geschrieben aber nicht gelesen werden.

Jeder Ein- und Ausgang des IO Expanders wird über einen Optokoppler getrennt. Zum Schutz der Optokoppler Seite, die bei den Eingängen von außen erreichbar ist, sind antiparallel zu den Leuchtdioden normale Siliziumdioden geschaltet um die Spannung bei Verpolung zu begrenzen. Zusätzlich befindet sich auch ein Widerstand in Serie um den Strom durch die Dioden zu begrenzen.

Der Transistor des jeweiligen Optokopplers, der als Ausgang beschaltet ist, wird durch Zenerdioden, parallel zu dem Kollector und Emitter Anschluss, gegen Überspannungen abgesichert.

Damit jeder Ein- und Ausgang potential mäßig unabhängig voneinander verwendet werden kann gibt es keinen gemeinsame Masse.

4.2.2.6 Versorgung

Die Versorgung der Basis Platine ist nach einem ähnlichen Prinzip wie in 4.2.1.3 beschrieben aufgebaut.

Im Gegensatz zur „Slave“ Platine ist das phyCore Modul der größte Verbraucher auf der nicht potentialgetrennten Seite und benötigt 5 V. Aus diesem Grund werden mit dem Schaltregler auf dieser Platine 5 V erzeugt. Für die 3,3 V wird ein zusätzlicher Längsregler mit fixer Ausgangsspannung verwendet.

Die beiden Messkanäle sind nicht nur von der restlichen Elektronik potentialgetrennt sondern auch untereinander. Das bedeutet, dass die Versorgung für diese Kanäle zweimal ausgeführt ist. Anstelle von „step up“ DC/DC Wandlern zur Potentialtrennung kommen hier DC/DC Wandler die 5 V übertragen zum Einsatz.

Die ebenfalls potentialgetrennt ausgeführte Signalerzeugung wird über einen weiteren DC/DC Wandler versorgt. Dieser erzeugt +5 V und -5 V die gefiltert direkt als Versorgung der Komponenten verwendet werden.

4.2.2.7 Anschlüsse

Die beiden Messkanäle sollen gemeinsam mit ihrer Versorgung ausgeführt werden. Das bedeutet je Kanal 6 Leitungen. Verwendet wird hierfür ein Wannenstecker.

Die Signalerzeugung deren Ausgang differentiell ist soll ebenfalls mit einer Versorgung ausgeführt werden. Auch dafür wird auch ein Wannenstecker mit 6 Kontakten verwendet.

Die digitalen Ein- und Ausgänge werden auf einen eigenen Wannenstecker gelegt. Es werden 16 Kontakte benötigt denn wie in 4.2.2.5 beschrieben gibt es keine gemeinsame Masse.

Das Prozessor Modul benötigt zwei verschiedene Stecker der Firma SamTec der Serie BTH mit 50 und 60 Kontakten.

Für die Ethernet Schnittstelle wird eine RJ-45 Buchse mit integriertem Übertrager eingesetzt. Zur Konfiguration des „Bootloaders“ und als Terminal wird eine RS232 Schnittstelle inklusive Pegelumsetzer auf eine Stiftleiste geführt.

Versorgt wird das gesamte System über eine Stiftleiste im 3,5 mm Raster mit drei Kontakten. Zwei dienen der Versorgung und einer ist mit dem Schirm der Ethernet Buchse verbunden.

Die Basis Platine benötigt zur Verbindung mit dem System Bus einen Stecker wie er bereits in 4.2.1.5 beschrieben ist.

4.2.2.8 Layout

Im Vergleich zum Layout der „Slave“ Platinen, siehe 4.2.1.5, ist dieses wegen der geringeren Bauteildichte weniger aufwändig.

Das phyTec Modul und das FPGA sind so platziert dass die Leitungen der Speicheranbindung so kurz wie möglich sind.

Die Platine ist in fünf potential getrennte Bereiche aufgeteilt. Ein Bereich für je einen Messkanal, das Layout dieser Bereiche ist identisch. Einen für die Signalerzeugung, einen für die digitalen Ein- und Ausgänge und der Hauptbereich auf dem sich das phyTec Modul sowie das FPGA befinden, siehe Abbildung 29.

Alle Anschlüsse sind am Rand der Platine verteilt. Der System Bus Verbinder ist auf der Unterseite der Platine an der zu den „Slave“ Platinen passenden Stelle angebracht. Die Platinen werden mit ihrer jeweiligen Unterseite zueinander zusammen gesteckt. Auf diese Weise ist es während der Entwicklung der Software möglich Messungen an beiden Systemen im Betrieb einfach durchzuführen.

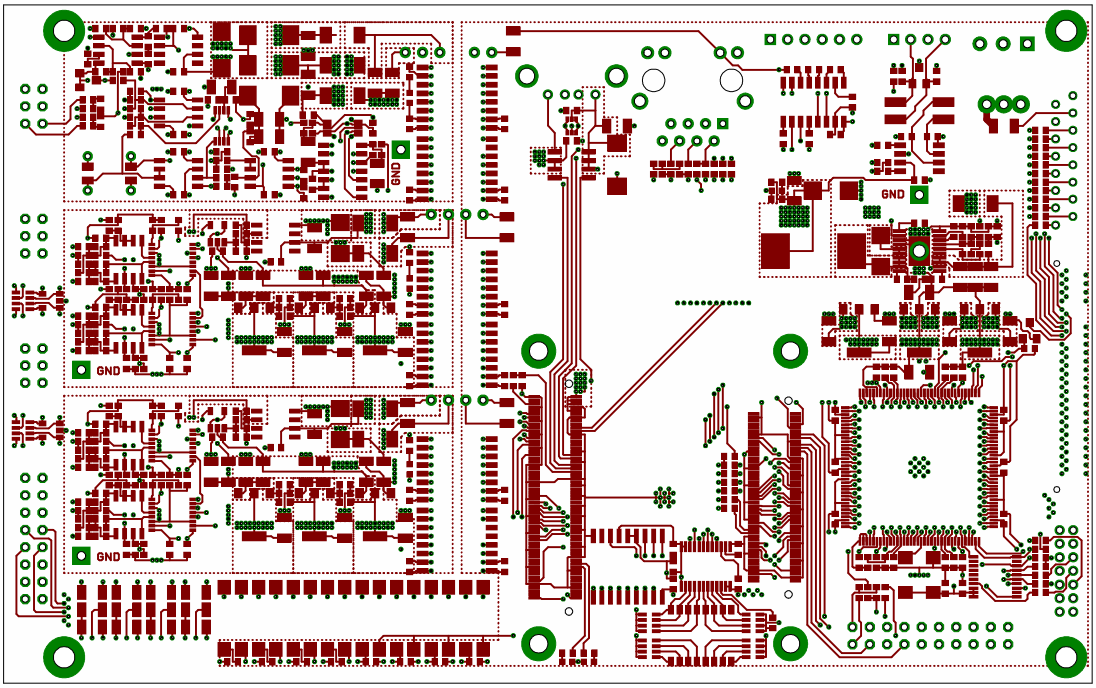

Abbildung 29: Layout der Oberseite der Basis Platine

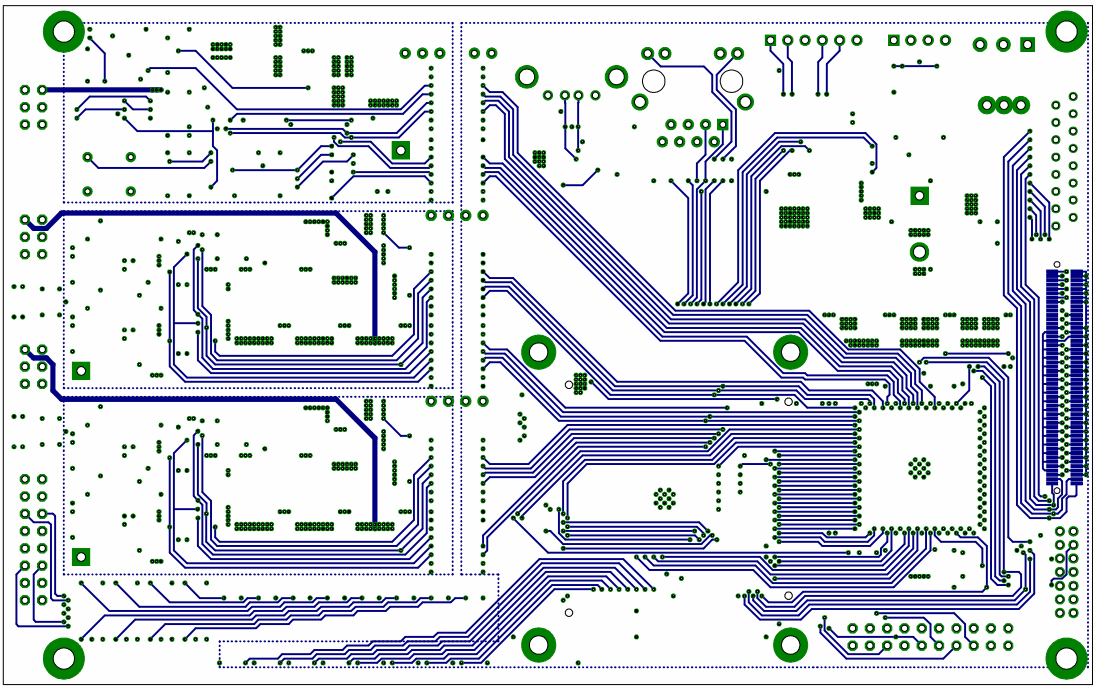

Abbildung 30: Layout der Unterseite der Basis Platine

Die beiden inneren Lagen wurden für Versorgungen und eine möglichst durchgängige Massefläche freigehalten.

4.3 Ergebnisse

Das Ergebnis der Hardware Entwicklung sind der Schaltplan und das Layout, siehe 4.2.1.6 und 4.2.2.8, für beide Platinen sowie je ein funktionsfähiger Prototyp.

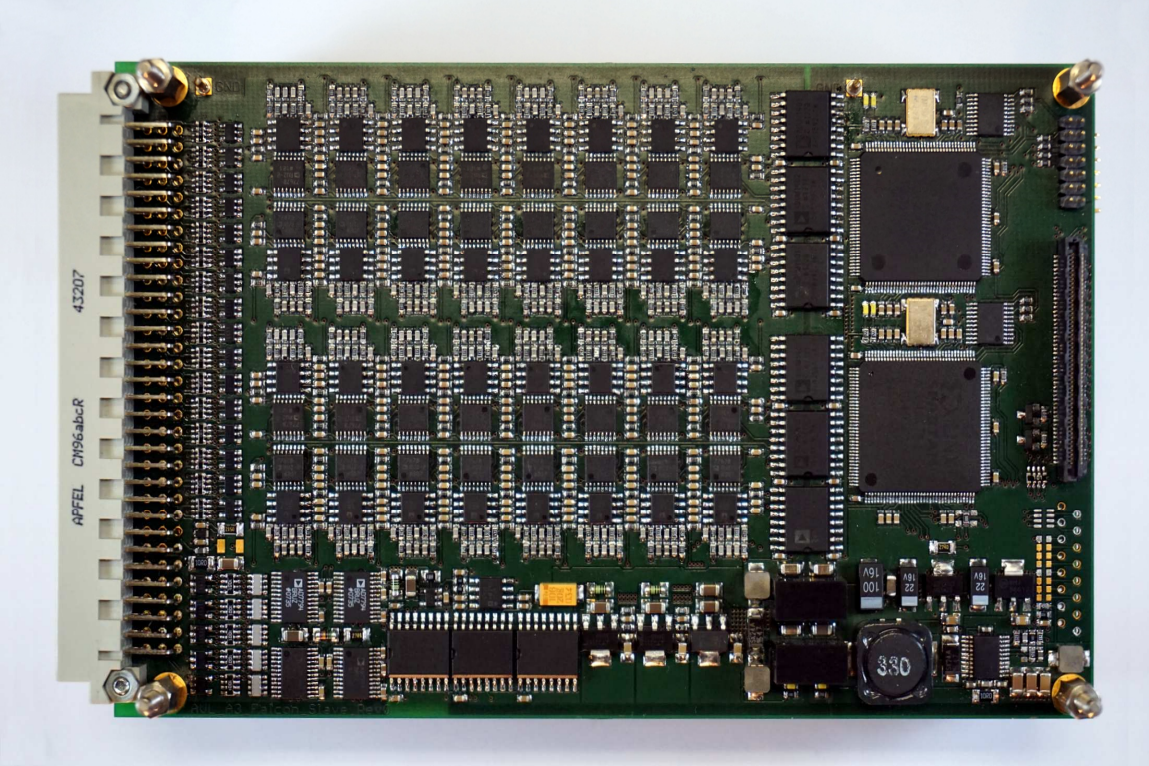

Abbildung 31: Voll bestückte „Slave“ Platine

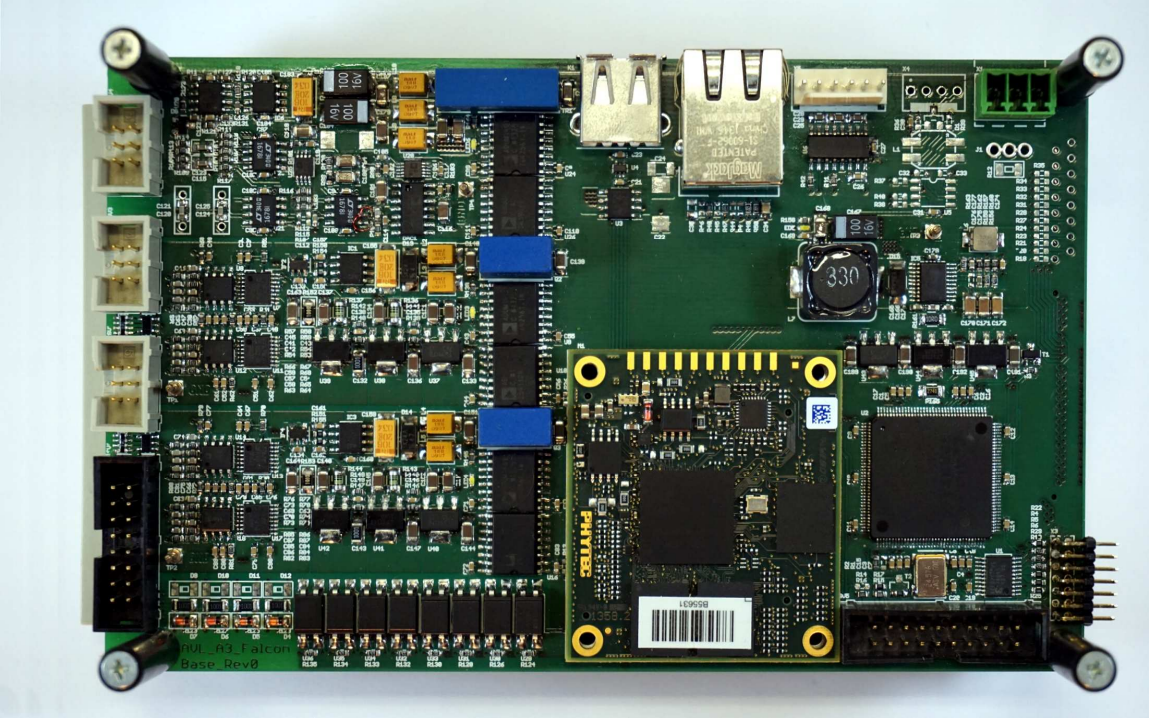

Abbildung 32: Voll bestückte Basis Platine

Die Platinen wurden händisch bestückt. Dadurch war es möglich einen Abschnitt nach dem anderen aufzubauen und in Betrieb zu nehmen. Auf diese Weise konnten Probleme schnell isoliert und behoben werden.

5 FPGA Design

Dieses Kapitel behandelt den Aufbau und die Implementierung der beiden FPGA Designs.

5.1 Entwicklungsumgebung

Die Entwicklung der FPGA Designs wurde mit dem frei verfügbaren Software Paket „Web Pack 13“ der Firma Xilinx durchgeführt. Dieses Paket bietet einen Synthesizer sowie einen Simulator für die Hardwarebeschreibungssprachen VHDL und Verilog.

In diesem Projekt wird VHDL für die Beschreibung der Funktionalität verwendet. Das Top Level Design wird textuell beschrieben. Das „Web Pack“ bietet daneben auch die Möglichkeit einer graphischen Beschreibung. Das textuelle Top Level Design hat den Vorteil, bei höherer Komplexität des Designs übersichtlicher und leichter wartbar zu sein. Daneben eignet sich die textuelle Beschreibung besser zur Verwendung mit einer Versionskontrolle.

5.2 Simulation

Bei der Entwicklung von FPGA Designs spielt die Simulation eine wichtige Rolle. Es gibt keine Möglichkeit das Design, ohne dieses zu verändern, in einem FPGA zu debuggen. Jede Debug Information muss durch zusätzliche Logik, zum Beispiel auf Pins, ausgegeben werden.

Die Simulation bietet den Vorteil den Verlauf von internen Signalen schnell und ohne zusätzlichen Aufwand nachverfolgen zu können. Dabei stellt der Simulator Werkzeuge, wie sie auch in der konventionellen Software Entwicklung Anwendung finden, zur Verfügung. Beispiele dafür sind „Breakpoints“ und „Watch Windows“. Daneben sind spezielle Werkzeuge, wie die grafische Darstellung von Signalverläufen, enthalten. Auf diesem Weg können logische Fehler in der Beschreibung schnell gefunden und ausgebessert werden.

Mit Hilfe einer Timing Simulation können Probleme, die erst in der realen Hardware auftreten würden, entdeckt werden. Das Modell für diese Simulation wird aus dem bereits synthetisierten, platzierten und gerouteten Design extrahiert. Der Rechenaufwand für diese Art von Simulation ist um ein vielfaches höher als bei der einfachen logischen Simulation.

Um ein Design zu simulieren muss eine Test Umgebung beschrieben werden. In dieser Umgebung wird das gesamte Design als Modul instanziiert und die Eingänge müssen entsprechen der realen Umgebung stimuliert werden.

5.3 Implementierung

Im folgenden Kapitel wird beschrieben wie die im Konzept definierten Module der beiden FPGA Systeme implementiert wurden.

5.3.1 „Slave" FPGA

Das „Slave" FPGA implementiert die Steuerung der ADCs für Strom-, Spannungs- und Temperaturmessung sowie die Kommunikation mit dem „Master" FPGA über den System Bus.

5.3.1.1 Takt Generator

Zur Generierung des Systemtaktes wird ein im FPGA enthaltenes PLL Modul verwendet um den Takt des externen 24 MHz Oszillators zu verdoppeln. Damit steht ein 48 MHz Takt für die diversen zeitkritischen Aufgaben zu Verfügung. Die für die Kommunikation mit den beiden ADC Systemen notwendigen Takte werden ebenfalls vom 24 MHz Takt abgeleitet. Das verwendete PLL Modul bietet die Ausgabe von heruntergeteilten Takt Signalen an. Mögliche Teiler sind 1,5 bis 7,5 in Schritten von 0,5 und 8 bis 16 in Schritten von 1.

Die Wahl des Taktes für das Auslesen der ADCs in den beiden „daisy chains" fällt auf den niedrigsten Möglichen, da so der Einfluss der Verzögerungen durch die Magnetkoppler gering gehalten wird. Das Auslesen eines Wertes von jedem Teilnehmer der Kette darf in Summe eine Periode des 32 kHz Abtasttaktes dauern. Dabei müssen $24 * 8 = 192$ Bit übertragen werden. Daraus folgt dass theoretisch ein Takt von mindesten $32000 * 192 = 6144000$ Hz benötigt wird. 24 MHz geteilt durch den Faktor 3,5 ergibt einen Takt von ungefähr 6,86 MHz. Dieser bietet noch genügend Spielraum für das Setzen und Löschen der „Chip Select" Leitungen.

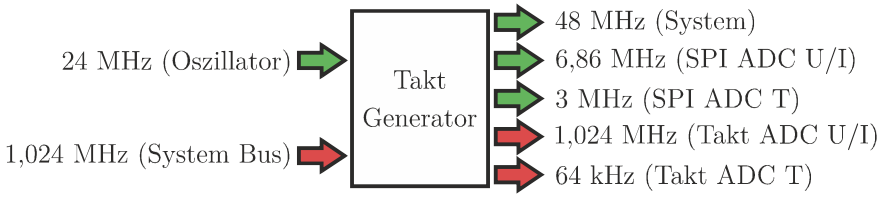

Abbildung 33: Überblick der im „Slave" FPGA erzeugten Taktsignale

In Rot sind die Taktsignale dargestellt, welche eine hohe Anforderung an die Genauigkeit haben.

Der Takt für die Kommunikation mit den langsamen ADCs wurde auf 3 MHz, einem Teiler Faktor von acht, festgelegt.

Der 1,024 MHz Abtasttakt für die ADCs der Strom- und Spannungsmessung wird nicht im „Slave“ FPGA erzeugt sondern vom „Master“ FPGA im System verteilt damit die Abtastung absolut synchron erfolgt. Die ADCs der Temperaturmessung benötigen einen 64 kHz Abtasttakt. Dieser wird lokal von dem 1,024 MHz Takt durch einen Frequenz Teiler abgeleitet.

5.3.1.2 „Daisy Chain Master”

Dieses Modul wird von dem „Master“ FPGA über ein Signal zum Starten der Messung und den Abtasttakt gesteuert.

Der Abtasttakt wird direkt zu den beiden ADC Ketten weitergeleitet. Das Messung Aktiv Signal dient als asynchroner Reset für die Ablaufsteuerung. Nach Verlassen des Reset Zustandes wird das „Power Down“ Signal für die ADCs „high“ gesetzt damit diese aus ihrem Ruhezustand geholt werden.

Ist dies geschehen wartet die Steuerung auf die steigende Flanke des „data ready“ Signals der ADCs. Mit der fallenden Flanke beginnt die Steuerung mit dem Auslesen der Messwerte. Für das Auslesen wird ein 24 Bit Schieberegister verwendet.

Die Magnetkoppler haben eine Verzögerung im Bereich von 40 ns. Diese Verzögerung wirkt sich trotz des relativ niedrigen SPI Taktes aus. Aus diesem Grund wird das Abtasten der MISO Leitung um einen halben SPI Takt verzögert.

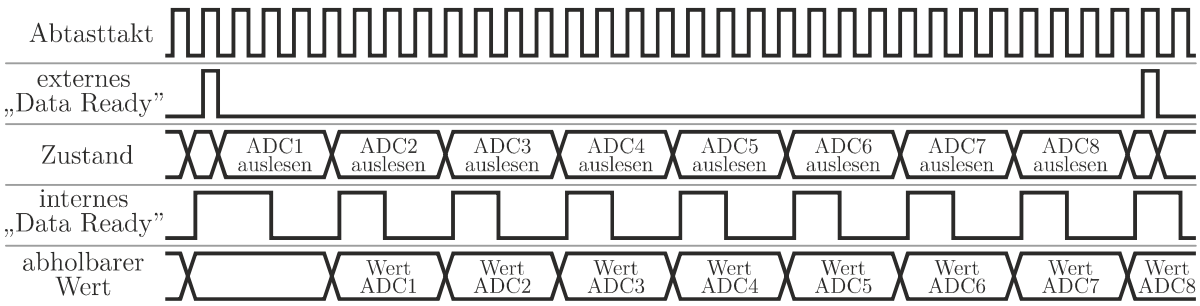

Abbildung 34: Auslesen eines Abtastwertes aller ADCs in einer Kette

Bei jedem 24. Bit wird der Inhalt des Schieberegisters in ein Ergebnisregister kopiert und ein internes „data ready“ Signal gesetzt. Zusätzlich wird die Nummer des ADCs zu dem dieser Wert gehört bereitgestellt. Der Wert kann dann von einem anderen Modul gelesen werden. Das interne „data ready“ Signal bleibt gesetzt bis das 12. Bit des nächsten Wertes in das Schieberegister geschoben wurde.

Dieser Vorgang wiederholt sich für jeden ADC in der Kette. Danach wartet die Steuerung wieder auf die steigende Flanke des „data ready“ Signals der ADCs, siehe Abbildung 34.

Werden die ADCs direkt nach ihrem Aufwachen mit dem Abtasttakt versorgt ist nicht sichergestellt wie viele Taktschläge jeder einzelne ADC bis zum ersten Ergebnis braucht. Zwischen den Ergebnissen sind es immer 32. Um sicherzustellen dass alle ADCs synchron mit dem Abtasten beginnen müssen sie zurückgesetzt werden. Dies kann das „Master“ FPGA erreichen indem der Abtasttakt 32 Taktschläge macht bevor das Messung Aktiv Signal gesetzt wird und die ADCs aufgeweckt werden.

Ein „Slave“ FPGA muss zwei SPI Ketten bedienen. Daher wird dieses Modul zweimal instanziiert. Zum Sparen von Ressourcen ist es sinnvoll die Wertausgabe als gemeinsames Bus System zu realisieren. Dies ermöglicht dem auslesenden Modul über „Chip Select“ Signale den neuesten Wert abzuholen.

Beide SPI Ketten teilen sich ein „data ready“ Signal der ADCs. Dieses kann von beiden Modul Instanzen parallel verwendet werden.

5.3.1.3 SPI Bus Master

Die Steuerung der ADCs für die Temperaturmessung wird von diesem Modul übernommen. Als Taktsignal werden der langsame SPI Takt und der 64 kHz Abtasttakt benötigt.

Diese ADCs sind in Registern organisiert. Es muss vor jedem Lese- oder Schreibzugriff die Adresse des gewünschten Registers übertragen werden.

Die Ablaufsteuerung dieses Moduls wird über das Messung Aktiv Signal des „Master“ FPGAs asynchron zurückgesetzt.

Wird der Resetzustand verlassen, werden die beiden ADCs einer nach dem anderen konfiguriert. Eingestellt werden dabei die Verwendung der externen Referenz, die Verwendung eines externen Abtasttaktes, die Filter Update Rate und der analoge Multiplexer wird auf den ersten Kanal eingestellt.

Sind die ADCs konfiguriert wartet die Steuerung auf ein Trigger Signal, siehe 3.2.1.4. Dieses sorgt für die Synchronität zwischen den beiden ADC Systemen.

Bei der steigenden Flanke des Trigger Signals werden beide ADCs in einen „Single Conversion“ Modus gebracht. Erst danach wird der Abtasttakt zu ihnen weitergeleitet. Diese ADCs haben keine dezidierte „data ready“ Leitung. Stattdessen wird der Abschluss der Konversion durch eine fallende Flanke auf der MISO Leitung angezeigt. Wurde diese Flanke detektiert wird der Abtasttakt deaktiviert und die Ergebnisse beider ADCs werden nacheinander ausgelesen und in separaten Registern zwischen gespeichert. Nach dem Auslesen werden die analogen Multiplexer auf den jeweils nächsten

Kanal geschaltet und es wird auf das nächste Trigger Signal gewartet, siehe Abbildung 35.

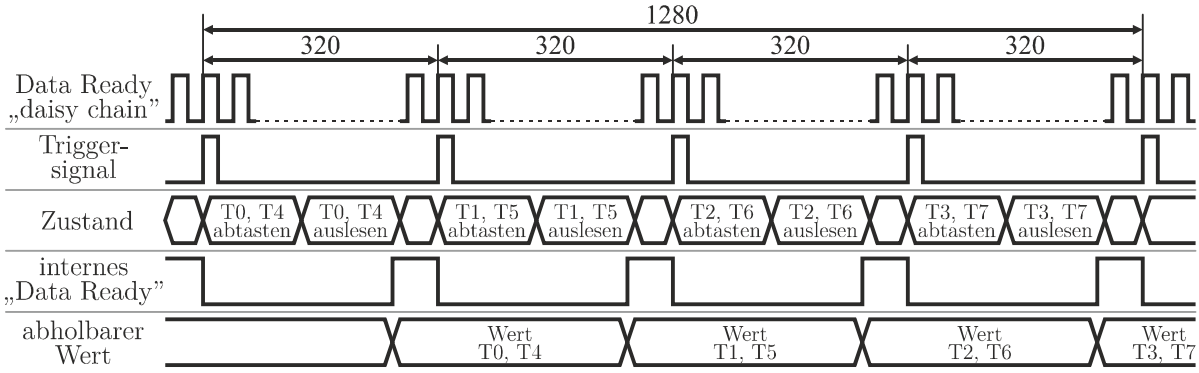

Abbildung 35: Auslesen eines Abtastwertes aller ADC Kanäle (T0-T7)

Das Auslesen der Daten für andere Module erfolgt ähnlich dem „Daisy Chain Master", allerdings ist keine Bus Topologie notwendig da im System nur eine Instanz existiert. Die beiden Werte können gleichzeitig aus ihren Zwischenspeichern gelesen werden nach dem über ein eigenes internes „data ready" Signal angezeigt wurde dass neue Werte vorhanden sind.

5.3.1.4 Trigger Generator

Dieser Generator erzeugt das Trigger Signal für den SPI Bus „Master". Es ist für die Synchronität der beiden Abtastsysteme verantwortlich und bestimmt die Abtastrate der Temperaturmessung.

Zur Festlegung der Abtastrate muss die Dauer einer Konversion pro Kanal inklusive dem Umschalten des Kanals in Betracht gezogen werden. Der verwendete ADC benötigt mindestens 4 ms zum Einschwingen des Filters nach dem Wechsel des verwendeten Kanals. Die Abtastrate wird auf 25 Hz pro Kanal festgelegt. Das bedeutet, dass jeder Kanal alle 40 ms abgetastet werden muss. Bei vier Kanälen folgt daraus dass für jeden einzelnen Kanal die Abtastung maximal 10 ms dauern darf.

Als Zeitbasis dient das „data ready" Signal der „daisy chain" ADCs. Die Frequenz des „data ready" Signals liegt bei 32 kHz. Das SPI Bus Master Modul muss, damit alle 10 ms eine Konversion gestartet wird, nach jeder 320. „data ready" Flanke getriggert werden, siehe Abbildung 35.

Die Ablaufsteuerung dieses Moduls wird über das Messung Aktiv Signal des „Master" FPGAs asynchron zurückgesetzt. Sobald die Steuerung den Resetzustand verlässt wartet sie auf die erste Flanke des „data ready" Signals. Wurde diese erkannt wird das

erste Triggersignal ausgegeben. Die weiteren Triggerzeitpunkte werden durch Zählen der „data ready“ Flanke bestimmt.

5.3.1.5 Datensammler

Der Datensammler holt die Daten von dem „daisy chain Master“ bzw. dem SPI Bus „Master“ ab und speichert sie temporär für das „Master“ FPGA.

Als Zwischenspeicher wird ein „dual ported“ RAM verwendet. Um die Realisierung einer Pufferfunktion einfach zu halten werden zwei getrennte Datensammler mit eigenen „dual ported“ RAMs implementiert, siehe 3.2.1.5. Einer für die Strom- und Spannungsmessung und einer für die Temperaturwerte.

Die Ablaufsteuerung der Datensammler wird durch das Messung Aktiv Signal des „Master“ FPGAs gestartet. Danach warten sie auf die internen „data ready“ Signale von dem „daisy chain Master“ bzw. dem SPI Bus „Master“. Wird eine positive Flanke bei diesen Signalen erkannt, liest der Datensammler die neuen Werte von dem Modul und speichert sie an den entsprechenden Stellen im RAM.

Wurden alle Werte eines Abtastzeitpunktes aller Kanäle im RAM gespeichert wechseln die Datensammler den verwendeten Speicherbereich und ermöglichen dem internen Bus und somit dem „Master“ FPGA die Werte auszulesen während die nächsten Werte bereits gesammelt werden.

5.3.1.6 System Bus Schnittstelle

Dieses Modul realisiert die Schnittstelle zum System Bus und stellt die Verbindung zwischen dem „Slave“ und dem „Master“ FPGA dar.

Die Wahl der Adresse wird durch ein „valid“ Signal gestartet, wie in 3.1.9 beschrieben. Zeigt dieses an dass die an den Eingängen anliegende Adresse gültig ist wird diese übernommen. Beide FPGAs auf einer Slave Platine lesen dieselbe Adresse. Die Unterscheidung zwischen den beiden FPGAs wird durch einen weiteren Eingang festgelegt, dessen Pegel die niederwertigste Stelle in der Adresse bildet.

Ist dieser Vorgang abgeschlossen wird die Adresse ohne die niederwertigste Stelle um eins erhöht an die FPGAs der nächsten „Slave“ Karte weitergegeben. Diese Verbindung auf der Platine besteht immer nur bei einem FPGA. Trotzdem geben beide FPGAs die Adresse aus. Abschließend wird das „valid“ Signal für die nächsten FPGAs gesetzt sowie die gemeinsame „ready“ Leitung für das „Master“ FPGA.

Während des normalen Betriebes regelt dieses Modul den Zugriff auf den System Bus. Dafür werden die obersten fünf Adressbits mit der gewählten Adresse verglichen und daraus ein „Chip Select“ Signal abgeleitet. Abhängig von der Adresse können da-

mit die Ausgangstreiber der Daten Leitungen in den hochohmigen Zustand versetzt werden.

Ob es sich um einen Lese- oder Schreibzugriff handelt wird durch eine zusätzliche „read/write“ Leitung bestimmt.

5.3.2 „Master“ FPGA

Das „Master“ FPGA implementiert das Sammeln der Messwerte von den „Slave“ FPGAs, die Steuerung der externen Stromquelle mit der dafür nötigen Signalerzeugung sowie die Kommunikation mit dem Prozessor Modul.

5.3.2.1 Takt Generator

Im Master FPGA werden wie im „Slave“ FPGA mit Hilfe eines PLL Moduls die verschiedenen Takt Signale erzeugt.

Als System Takt wird direkt der 24,576 MHz Takt des externen Oszillators verwendet. Für die Kommunikation mit den ADCs der beiden zusätzlichen Strom- und Spannungskanäle wird der gleiche Teiler Faktor von 3,5, wie in 5.3.1.1, verwendet. Dies ergibt eine Frequenz von ungefähr 7 MHz. Es könnte auch ein niedrigerer Takt verwendet werden da sich nur zwei Teilnehmer in der „daisy chain“ befinden. Der schnellere Takt bietet die Möglichkeit, sollte es notwendig sein, ADCs mit höherer Abtastrate zu verwenden.

Der 1,024 MHz Abtasttakt für die ADCs auf der Basis Platine und den „Slave“ Platinen wird durch einen Zähler direkt vom externen Oszillator abgeleitet. Mittels dieses Zählers wird auch eine „prescaler“ Funktion implementiert. Je nach Einstellung in dem Systemkontroll Register kann der Systemtakt durch 24, 48, 96 oder 192 geteilt werden. Damit kann eine Abtastrate von 4, 8, 16 oder 32 kHz eingestellt werden.

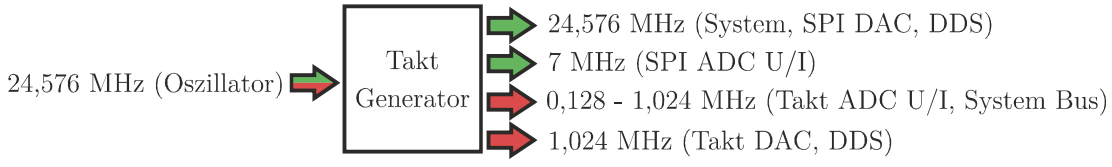

Abbildung 36: Überblick der im „Master“ FPGA erzeugten Taktsignale

In Rot sind die Taktsignale dargestellt, welche eine hohe Anforderung an die Genauigkeit haben.

Der externe DDS des Signal Generators wird mit einem von dem „prescaler“ unabhängigen 1,024 MHz Takt versorgt. Damit die internen DDS Kanäle den gleichen Frequenzbereich wie der externe DDS erzeugen können muss der für die Ausgabe

zuständige DAC mit dieser Frequenz neu berechnete Werte ausgeben. In diesem Fall muss ein 16 Bit Wert in weniger als 1 µs an den DAC geschrieben werden. Damit muss der SPI Takt für die Signalerzeugungskomponenten entsprechend hoch sein. Das erfordert theoretisch eine Bitrate von mindestens 16,384 Mbit pro Sekunde. Für die Kommunikation mit dem DAC wird deshalb der 24,576 MHz Systemtakt verwendet.

5.3.2.2 Prozessor Modul Schnittstelle

Damit der Prozessor Daten auf das FPGA schreiben beziehungsweise vom FPGA lesen kann muss dieses die Funktionen eines SRAMs abbilden. Der Zugriff erfolgt über einen synchronen, gemultiplexten 16 Bit Adress-/Datenbus.

Bis das FPGA durch die entsprechende „Chip Select“ Leitung vom Prozessor Modul selektiert wird sind alle bidirektionalen Pins im Zustand „High Impedance“. Greift der Prozessor auf das FPGA zu, legt dieser, nachdem die „Chip Select“ Leitung auf Null gesetzt wurde, die gewünschte Adresse auf den Bus. Die Adresse wird, sobald die „address valid“ Leitung des Prozessors auf Null gegangen ist, bei der nächsten fallenden Flanke des Bus Taktes übernommen und auf die internen von den Datenleitungen getrennten Adressleitungen gelegt.

Durch die Leitungen „Write Enable“ und „Output Enable“ signalisiert der Prozessor um welche Art von Zugriff es sich handelt. Abhängig davon werden die Daten auf den internen Bus gelegt oder von diesem gelesen und auf den externen Bus gelegt. Ein internes „read/write“ Signal wird dem entsprechend gesetzt um den Bus Teilnehmern im FPGA die Art Zugriffes mitzuteilen. Der Bustakt vom Prozessor wird direkt als interner Bustakt verwendet.

Intern werden die Busteilnehmer durch „chip select“ Leitungen selektiert. Diese Aufgabe übernimmt ein zusätzliches Modul. Das Modul vergleicht die auf dem internen Bus anliegende Adresse mit den Adressräumen der verschiedenen Teilnehmer und setzt dem entsprechend die „chip select“ Leitungen.

5.3.2.3 Systemsteuer- und Statusregister

Zur Steuerung und Überwachung aller Module werden Register implementiert die vom Prozessor gelesen und teilweise geschrieben werden können. Dieses Modul ist an den internen Adress-/Datenbus angebunden. Die Steuerung der anderen Module erfolgt über dezidierte Signale. Die Register werden als ein „Array“ von „Standard Logic“ Vektoren mit einer Breite von jeweils 16 Bit angelegt.

Greift der Prozessor auf den Adressbereich dieses Moduls zu wird die entsprechende interne „chip select“ Leitung gesetzt. Ist das Register Modul selektiert wird durch eine

Schleife in der VHDL Beschreibung die Adresse auf dem internen Bus mit dem „Array“ Index der Register verglichen.

Bei einem Lesezugriff wird der Inhalt des Registers, welches der anliegenden Adresse zugeordnet ist, bei der nächsten fallenden Flanke des Bustaktes auf den Bus gelegt. Bei einem Schreibzugriff werden die auf dem Bus liegenden Daten bei der nächsten fallenden Flanke in das Register kopiert.

Folgende Register mit den beschriebenen Funktionen sind implementiert:

- Status
  Dieses Register kann nur gelesen werden. Es enthält zwei Flags, eines zum Anzeigen ob gerade eine Messung durchgeführt wird und eines ob die Signalerzeugung aktiv ist.
- Datensammlersteuerung
  Mit diesem Register wird festgelegt wie viele Slave Karten an der Messung beteiligt sind sowie der „prescaler“ Faktor für den Abtasttakt. Es enthält ein Flag zum Starten und Stoppen eines Messvorganges.
- Ein- und Ausgangssteuerung
  Die Kommunikation mit dem IO Expander sowie die Überstromüberwachung wird von diesem Register aus aktiviert. Zusätzlich kann der logische Pegel auf den die Überstromüberwachung reagiert eingestellt werden.
- Zustände der Ausgänge
  Die Zustände der Ausgänge können über dieses Register festgelegt werden.
- Zustände der Eingänge
  Die Zustände der Eingänge können über dieses Register ausgelesen werden.
- Signalerzeugungssteuerung
  Mit Hilfe eines Flags in diesem Register wird die Signalerzeugung gestartet.
- Gesamt Verstärkung
  Einzustellender Wert für den multiplizierenden DAC am Ende der Signalerzeugungskette.
- Frequenz des externen DDS
  Einzustellende Frequenz für den externen DDS. Der 28 Bit Wert ist auf zwei Register aufgeteilt.
- Frequenz, Phase und Verstärkung der internen DDS 0 – 3.
  Einzustellende Werte für die Frequenz, Phase und Verstärkung der vier internen

DDS. Die Werte für Frequenz und Phase sind jeweils auf zwei Register aufgeteilt.

- DC Offset
  Offset Wert der intern berechneten DDS Werte.
- Version
  Dieses Register kann nur gelesen werden und dient zur Identifikation der Version des „Master“ FPGA Designs.

Die Überstromüberwachung ist in diesem Modul implementiert da es Zugriff auf das „Enable“ Flag der Signalerzeugung und den Zustand der Eingänge besitzt. Ist die Überwachung aktiv wird die Erzeugung nur freigeschaltet wenn der erste Eingang nicht dem eingestellten Pegel entspricht. Kommt es während der aktiven Signalerzeugung zu einem Überstromzustand wird diese deaktiviert. Aktiviert kann sie nur wieder durch das Löschen und erneute Setzen des „Enable“ Flags der Signalerzeugung durch den Prozessor werden.

5.3.2.4 „Daisy Chain Master”

Zum Auslesen der beiden „on board“ Kanäle benötigt auch das „Master“ FPGA ein solches Modul. Die Funktionsweise entspricht dem des „Slave“ FPGA Moduls wie in 5.3.1.2 beschrieben.

Im Gegensatz zu den beiden absolut getrennten SPI Ketten auf der „Slave“ Platine teilen sich die beiden Ketten hier alle Leitungen bis auf die MISO Leitung, siehe 4.2.2.2.

In diesem Modul werden zwei Schieberegister parallel implementiert um die Werte von beiden MISO Leitungen gleichzeitig einlesen zu können. Die insgesamt vier Ergebnisse werden parallel ausgegeben. Jedes Modul das diese Werte benötigt kann sie parallel lesen.

5.3.2.5 Datensammler

Die in diesem Modul implementierte Ablaufsteuerung kontrolliert den gesamten Messvorgang und ermöglicht dem Prozessor die Messwerte in Paketen auszulesen. Die dafür notwendigen Aufgaben sind auf drei „state machines“ verteilt: die grundsätzliche Ablaufsteuerung, die Messsteuerung die das Freischalten des Abtasttaktes kontrolliert und die Triggersteuerung zur Signalisierung fertiger Pakete für den Prozessor.

Zum Daten Austausch zwischen dem FPGA und dem Prozessor wird ein „dual ported“ RAM verwendet. Port A des RAM Modules ist mit dem internen Bus verbun-

den. Greift der Prozessor auf den entsprechenden Adressbereich zu kann er Daten aus dem RAM lesen. Über Port B erhält die Ablaufsteuerung Zugriff auf das RAM.

Die Pakete werden durch die Ablaufsteuerung vorbereitet. Jedes Paket enthält einen 64 Bit Zeitstempel sowie ein 32 Bit Feld mit den Informationen wie viele Slave Karten an der Messung beteiligt sind, den Zustand der Eingänge und ob es sich bei diesem Paket um Strom- und Spannungswerte oder Temperaturwerte handelt. Die maximal 1036 Bytes eines Paketes werden für den Prozessor transparent gepuffert. Sobald ein Paket fertig gestellt ist erhält der Prozessor Zugriff auf dessen RAM Bereich. Im Hintergrund wird dann bereits das nächste Paket zusammengestellt. Für den Prozessor sieht es aus als würde er jedes Paket von der gleichen Stelle im RAM lesen.

Wird die Messung durch das „Enable“ Signal des Datensammlersteuerungsregisters gestartet, beginnt die Ablaufsteuerung damit die Anzahl der verwendeten „Slave“ Karten zwischenzuspeichern und die Messsteuerung zu aktivieren. Ist die Anzahl Null werden die beiden Kanäle auf der Basis Platine verwendet. Die Anzahl der „Slaves“ dient als Index für eine „look up“ Tabelle um die Anzahl der Messwerte, die in ein Paket passen, zu bestimmen. Es werden sämtliche Register und Zähler die für den Ablauf notwendig sind zurückgesetzt.

Zu Beginn werden für beide Pakete, Strom und Spannung beziehungsweise Temperatur, die Zeitstempel und die Zusatzinformationen geschrieben. Danach wartet die Ablaufsteuerung auf ein Signal der Messsteuerung, welches anzeigt dass neue Werte von den „Slave“ Platinen oder den „on board“ Kanälen gelesen werden können.

Währenddessen beginnt die Messsteuerung die ADCs der Messkanäle, durch ausgeben von 32 Perioden des Abtasttaktes ohne dem „Enable“ Signal zur Synchronisation, wie in 5.3.1.2 beschrieben, zurück zu setzen. Ist dies geschehen benötigt der AD7767-2 2370 Taktschläge bis der Filter eingeschwungen und der erste Wert bereit ist. [14] Nach diesen Taktschlägen wartet die Messsteuerung zwei weitere Taktschläge um sicherzustellen, dass die Datensammler der „Slave“ FPGAs bereits die Puffer gewechselt haben und die Werte ausgelesen werden können. Während der restlichen Messung wird der Ablaufsteuerung alle 32 Taktschläge signalisiert dass neue Werte bereit sind. Die Messsteuerung zählt die Anzahl der bereits fertigen Werte. Dies stellt die Zeitbasis für die Temperaturmessung dar. Sind 960 Messwerte fertig wurde gerade die Messung des letzten Temperatur Kanals auf den „Slave“ Karten gestartet. Um auch hier sicherzustellen dass der letzte Wert fertig und die Puffer getauscht sind wird beim ersten Mal auf 1600 Messwerte gewartet und dann immer 1280.

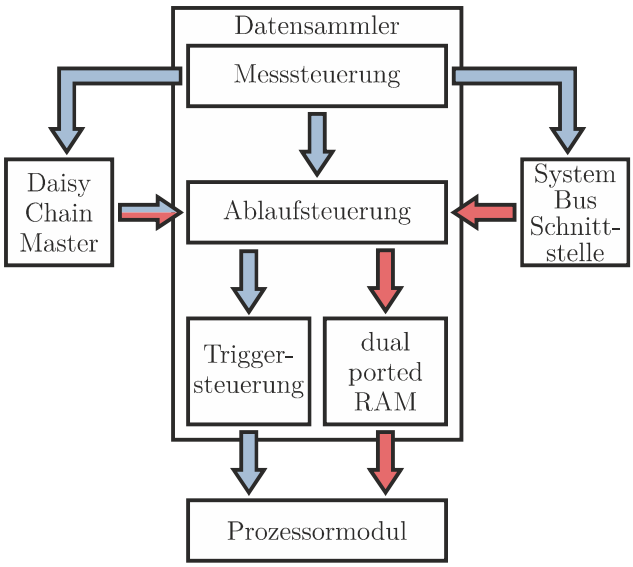

Abbildung 37: Aufbau des Datensammlers im „Master“ FPGA

In Rot ist der Datenfluss der Messwerte dargestellt. Mit Blau sind die Steuerleitungen markiert

Wenn die Ablaufsteuerung das Signal für neue Messwerte erhält beginnt sie entweder damit die „on board“ Kanäle auszulesen oder die „Slave“ FPGAs. Das Auslesen der „Slave“ FPGAs erfolgt in einer zweistufigen Pipeline. Mit der ersten negativen Flanke auf der Systembustaktleitung wird die gewünschte Adresse ausgegeben. Bei der darauffolgenden positiven Flanke übernehmen die Busteilnehmer diese Adresse. Mit der nächsten negativen Flanke wird die nächste gewünschte Adresse ausgegeben. Bei der nächsten positiven Flanke übernehmen die Busteilnehmer wiederum die neue Adresse und gleichzeitig können die Daten der letzten Adresse gelesen werden. Dieser Vorgang wiederholt sich für jeden Messwert der von den einzelnen FPGAs ausgelesen wird. Beim Wechsel des Zugriffs von einem FPGA zum nächsten muss diese Pipeline geleert und wieder neu gefüllt werden. Dies ist notwendig da das „chip select“ von jedem Teilnehmer asynchron aus der am Bus liegenden Adresse abgeleitet wird. Sobald die Adresse in den Bereich eines anderen Teilnehmers kommt ändern sich die Daten auf dem Bus sofort da nun dieser Teilnehmer auf den Bus zugreifen darf. Anstelle der Daten der letzten Adresse würden in diesem Fall die Daten der aktuellen Adresse gelesen werden.

Hat die Messsteuerung angezeigt dass Temperaturwerte bereit sind werden diese direkt nach den Strom- und Spannungswerten auf dieselbe Art und Weise ausgelesen. Beim Wechsel des Zugriffs von einem RAM auf das nächste in jedem „Slave“ FPGA tritt dasselbe Problem wie beim Wechsel zwischen den FPGAs auf. Die Lesepipeline muss geleert und neu befüllt werden.

Werden die „on board“ Kanäle ausgelesen wird auf das „data ready“ Signals des „daisy chain Masters“ anstelle des Signals der Messsteuerung gewartet.

Beim Lesen der Daten werden diese nacheinander in das „dual ported“ RAM an die entsprechende Stelle des gerade in Bearbeitung stehenden Paktes geschrieben. Wurden alle Werte des aktuellen Abtastzeitpunktes ausgelesen wird überprüft ob das Paket fertig ist. Ist dies bei einem oder beiden Paketen der Fall wird der Puffer getauscht und die Triggersteuerung damit beauftragt dem Prozessor ein oder zwei neue Pakete zum Auslesen zu signalisieren. Danach werden die entsprechenden Metadaten der nächsten Pakete geschrieben und es wird wieder auf das nächste Signal der Messsteuerung oder der „daisy chain“ gewartet.

Die Triggersteuerung erzeugt einen Puls auf der „Packet bereit“ Leitung die zum Prozessor führt und löst dort einen Interrupt aus. Ist auch ein Temperatur Paket fertig wird dies durch eine zusätzliche „Temperatur Paket bereit“ Leitung angezeigt. Beide Pakete befinden sich im selben RAM. Das Umschalten des Puffers auf der Seite des Prozessors geschieht durch ein höherwertiges Bit in der Adresse. Damit werden vom Prozessor aus gesehen immer beide Puffer gleichzeitig getauscht. Die Triggersteuerung darf nur dann Anzeigen dass ein Temperatur Paket bereit ist wenn der richtige Speicherbereich vom Prozessor gelesen werden kann.

Wird die Messung durch das „Enable“ Signal beendet kehren die Ablaufsteuerung und die Triggersteuerung in ihre Ruhezustände zurück. Die Messsteuerung gibt den Abtasttakt noch solange aus bis der nächste Wert fertig wäre und kehrt dann in ihren Ruhezustand zurück. Damit ist gewährleistet, dass die „Slave“ FPGAs die Messung ebenso geordnet beenden können.

5.3.2.6 Signal Generator

Dieses Modul wird mit zwei verschiedenen Taktsignalen versorgt. Zum einen mit dem Systemtakt und zum anderen mit dem für die Signalerzeugung gewählten 1,024 MHz Ausgabetakt.

Der verwendete DDS „IP Core“ bietet die Möglichkeit mehrere DDS Kanäle im Zeitmultiplex zu berechnen. Damit kann die Anforderung der Generierung eines Signals mit 4 diskreten Frequenzen gelöst werden. Die Bitbreite des Phasen Akkumulators sowie die Auflösung der Ausgangsgröße können konfiguriert werden. Als Auflösung der Ausgangsgröße wird 16 Bit entsprechend dem verwendeten DAC gewählt. Um den gewünschten Frequenzbereich abdecken zu können muss der Phasen Akkumulator min-

destens 21 Bit breit sein. Damit der Frequenzbereich identisch mit dem des externen DDS ist wird eine Breite von 28 Bit verwendet.

Die Signalerzeugung wird mit einer „state machine“, welche den generellen Ablauf und die SPI Kommunikation mit den externen Komponenten steuert, und einer Zweiten, welche den DDS „IP Core“ kontrolliert, realisiert. Diese werden nachfolgend mit Ablaufsteuerung und DDS Steuerung bezeichnet.

Nach dem Systemstart wird der externe DDS deaktiviert und der multiplizierende DAC sowie der für die Signalerzeugung verwendete DAC auf Null gesetzt.

Danach wartet die Ablaufsteuerung auf das „Enable“ Flag im Kontrollregister der Signalerzeugung. Wird dieses Flag durch den Prozessor gesetzt werden alle Register im Zusammenhang mit der Signalerzeugung zwischengespeichert und die DDS Steuerung aktiviert. Der externe DDS wird für die gewünschte Frequenz konfiguriert und mit dem Abtasttakt versorgt. Der multiplizierende DAC wird auf die gewünschte Abschwächung eingestellt.

Währenddessen lädt die DDS Steuerung den „IP Core“ mit den Inkrementen für die vier gewünschten Frequenzen und den Phasenverschiebungen. Nach dieser Initialisierung wird auf die nächste steigende Flanke des Ausgabetaktes gewartet. Diese dient als Trigger um den „Core“ mit dem Systemtakt zu versorgen. Ist ein berechneter Wert fertig wird dieser ausgelesen und mit seinem Skalierungsfaktor gewichtet.

Der 16 Bit Skalierungsfaktor wird als „Fixed Point“ Zahl interpretiert bei der das MSB mit 2^{-1} gewertet wird. Durch diese Darstellung sind Faktoren im Bereich von 0 bis 0.999984741 möglich. Der Faktor eins ist nicht darstellbar, daher kann nicht der ganze mögliche Wertebereich des Ergebnisses verwendet werden. Diese Einschränkung liegt im Bereich eines LSBs und ist vernachlässigbar.

Die Skalierung wird durch eine Multiplikation von zwei 16 Bit Zahlen realisiert, das Ergebnis ist eine 32 Bit Zahl, deren MSB mit 2^{15} gewertet wird. Für die weitere Berechnung werden die oberen 16 Bit verwendet. Diese Art der Skalierung funktioniert nur mit vorzeichenlosen Werten. Der DDS „Core“ liefert vorzeichenbehaftete Werte in Zweierkomplement Darstellung. Negative Werte in dieser Darstellung können nicht direkt wie beschrieben skaliert werden. Um dieses Problem zu umgehen wird zuerst der Absolut Wert gebildet, dieser skaliert und im Falle eines negativen Wertes negiert. Die Negierung wird durch bitweise Inversion und Addition von eins realisiert.

Die Werte der einzelnen DDS Kanäle können nacheinander abgeholt werden. Daher ist es möglich dieselbe Multiplizierereinheit für alle vier Werte zu verwenden.

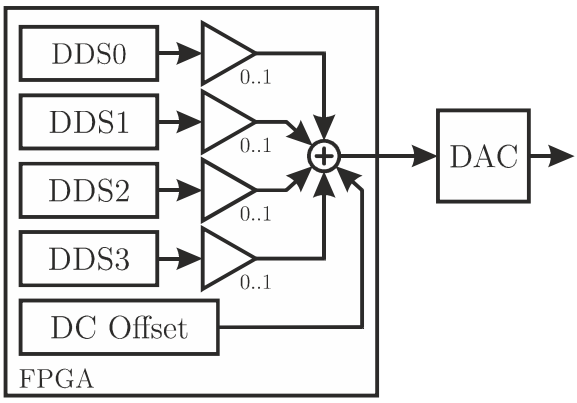

Abbildung 38: Schematische Darstellung des vierkanaligen DDS

Nach der Skalierung werden die Werte aufsummiert und der DC Offset dazugezählt. Diese Summe wird auf den mit einer vorzeichenbehafteten 16 Bit Zahl darstellbaren Bereich beschränkt. Wurde der letzte Kanal ausgelesen wird der Takt des „IP Cores“ bis zum nächsten Trigger durch den Ausgabetakt deaktiviert. Für den DAC muss der berechnete Wert von einer Zweierkomplement Darstellung in einen vorzeichenlosen Wert mit dem halben Bereich als Offset umgerechnet werden.

Ist ein solcher Zyklus abgeschlossen teilt die DDS Steuerung der inzwischen wartenden Ablaufsteuerung mit dass der DAC aktualisiert werden kann. Die Ablaufsteuerung schreibt den neuen Wert in den DAC und wartet mit dem Signal zum Übernehmen auf die steigende Flanke des Abtasttaktes. Mit der steigenden Flanke wird der DAC über die „load“ Leitung aktualisiert. Danach wird wieder auf die DDS Steuerung gewartet.

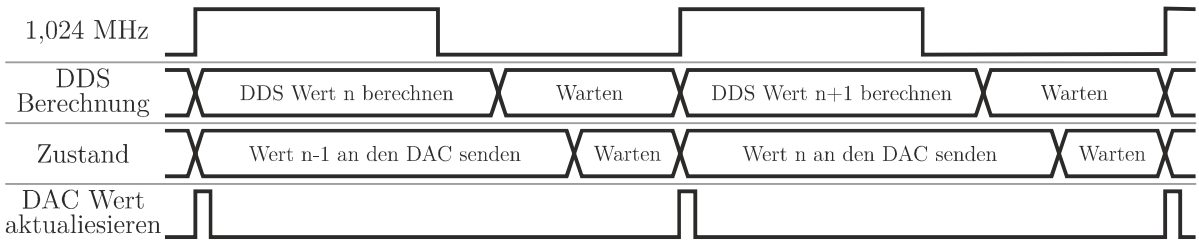

Abbildung 39: Berechnung der DDS Werte und Aktualisierung des DAC

Ist das „Enable“ Flag im Kontrollregister der Signal Erzeugung nicht mehr gesetzt werden der externe DDS deaktiviert und die beiden DACs auf Null gesetzt.

Die Abbildung 39 zeigt den Ablauf der Zustände der beiden „state machines“ während der Signalerzeugung.

5.3.2.7 IO Interface

Wie in 4.2.2.4 erklärt wird für die digitalen Ein- und Ausgänge ein SPI IO Expander verwendet. Dieses Modul muss mit diesem IC zyklisch kommunizieren um die Eingänge auszulesen sowie die Ausgänge auf die aktuell geforderten Pegel zu setzen.

Die Kommunikation mit dem IO Expander erfolgt Register basiert. Bei jedem Zugriff muss zuerst die Adresse des gewünschten Registers inklusive der Information ob es sich um einen Lese oder Schreibzugriff handelt übermittelt werden.

Die Ablaufsteuerung initialisiert nach dem Reset den IO Expander. Eingestellt wird welche der acht GPIOs als Ein- bzw. Ausgänge fungieren und ob die Pegel der Eingänge invertiert werden sollen. Danach wird zyklisch das Register mit den Pegeln der Eingänge abgefragt und das Register für das Setzen der Ausgänge beschrieben.

Die Pegel Information der Eingänge wird über vier Signale im restlichen System bereitgestellt. Die gewünschten Pegel der Ausgänge werden dem Modul ebenso über vier Signale mitgeteilt.

5.3.2.8 System Bus Schnittstelle

Dieses Modul ermöglicht dem Datensammler den Zugriff auf den System Bus und somit auf die gesammelten Daten der „Slave“ FPGAs.

Zu Systemstart wird die Konfiguration der Adressen der Busteilnehmer durch ein „valid“ Signal initiiert. Als Adresse für die erste „Slave“ Platine wird null ausgegeben. Sobald auf der gemeinsamen Rückleitung die Bereitschaft aller „Slave“ FPGAs durch das Erkennen des rezessiven Zustandes der „wired and“ Logik festgestellt wird, wird dies dem Datensammler signalisiert.

Als Takt für den System Bus wird der 24,576 MHz Systemtakt verwendet. Damit ist es möglich Daten bei einem inkrementellen Zugriff unter Ausnützung von „pipelining“, siehe 5.3.2.5, mit einer Rate von etwas weniger als 50 MB/s zu transferieren.

5.3.3 Ergebnisse

Ergebnis der FPGA Entwicklung ist die synthetisierbare Beschreibung zweier Systeme welche in der Lage sind die geforderten Messungen in harter Echtzeit durchzuführen. Zusätzlich implementieren diese Systeme die Steuerung für die restlichen Komponenten auf der Basis sowie der „Slave“ Platine.

Die Synthese eines FPGA Designs erzeugt eine Konfigurationsdatei welche mit Hilfe eines JTAG Adapters direkt in das Ziel FPGA geladen werden kann. Damit die Konfiguration automatisch beim Systemstart vom FPGA aus einem Lade ROM gelesen wird muss diese Datei in ein entsprechendes Abbild umgewandelt werden. Dieses Abbild kann ebenfalls mit Hilfe ein JTAG Adapters im Lade ROM gespeichert werden.

6 Software

Dieses Kapitel beschreibt die Entwicklung und Implementierung der verschiedenen Software Komponenten welche im Prozessor Modul ausgeführt werden.

6.1 Entwicklungsumgebung

Die gesamte Software wird für einen Linux Kernel 3.2 entwickelt welcher auf einem ARM Cortex A8 System ausgeführt wird. Das verwendete Entwicklungssystem ist eine „Virtual Machine“ auf der Ubuntu 12.04 als Betriebssystem installiert ist. Die verwendete „Tool Chain“ ist das „Cross Compiler“ System PTXDIST der Firma „OSELAS“ mit integrierter Patch und Packet Verwaltung. Die Grundlage der Patch Verwaltung ist das Programm „quilt“ welches zur Erstellung und Anwendung von Patch Serien verwendet wird. PTXDIST stellt alle Werkzeuge zur Erzeugung eines Kernel Images und des „root“ Dateisystems bereit. Als Editor wird in der Kommandozeile „VIM“ verwendet und in der graphischen Benutzeroberfläche die integrierte Entwicklungsumgebung „Eclipse“.

Grundlage der gesamten Entwicklung ist das für dieses Prozessor Modul verfügbare „Board Support Package“. In diesem Paket sind alle notwendigen „Patches“ enthalten um den Linux Kernel an die, auf dieser Plattform verfügbare, Hardware anzupassen.

Für den Kernel betreffende Module wird die Programmiersprache C verwendet. Das „user space“ Programm ist in C++ geschrieben.

Änderungen am Kernel werden in dessen „Build“ System als zusätzlicher „Patch“ für die „Sourcen“ des Kernels integriert.

Der Kernel Treiber wird über ein selbst erstelltes „Makefile“ kompiliert. Für die „user space“ Software wird das von „Eclipse“ generierte „Makefile“ verwendet. Der Kernel selbst wird mit Hilfe von PTXDIST kompiliert.

Die Entwicklung der Software wurde bereits begonnen bevor eine Hardware zur Verfügung stand. Um dies zu ermöglichen wurde ein Entwicklungs-Kit der Firma phyTec für deren Cortex A8 Modul verwendet.

6.2 Implementierung

Ziel der Implementierung ist wie bei jeder Software Entwicklung ein gut strukturiertes, einfach zu erweiterndes Programm.

6.2.1.1 Das „Board File“

Das für dieses Modul angepasste „Board File“ im „Board Support Package“ geht von der Hardware, welche auf dem Entwicklungs-Kit verfügbar ist, aus.

Um das System an die neue Hardware anzupassen werden alle Konfigurationen betreffend den Pins des Prozessors sowie die Initialisierung von Plattform Geräten, wie zum Beispiel dem Display Kontroller, welche nicht mehr verfügbar sind entfernt.

An den externen Bus ist auf einer „chip select“ Leitung der „Flash“ Speicher für das Kernel Image und das „root“ Dateisystem angeschlossen. Auf einer weiteren „chip select“ Leitung liegt jetzt das „Master“ FPGA. Der Flashspeicher benützt ein 8 Bit Nand Interface. Das FPGA wird über eine synchrone 16 Bit SRAM Schnittstelle angesteuert. Aus diesem Grund müssen die noch nicht verwendeten Leitungen des externen Speicher Kontrollers auf die entsprechenden Pins gelegt werden. Ein wichtiges Detail ist hier dass die Takt Leitung für den synchronen Zugriff, obwohl sie ein Ausgang ist, als Eingang konfiguriert werden muss. Der Kontroller nutzt das Eingangssignal dieses Pins zur Resynchronisierung um damit ohne Verzögerungen durch Gatter Laufzeiten der Ausgangstufe den gleichen Takt wie der externe Speicher zu bekommen.

Der „Platform Device“ Treiber für das „Master“ FPGA ist zur Initialisierung bei Systemstart hinzugefügt.

6.2.1.2 Der „Platform Device“ Treiber

Dieser Treiber wird dem Kernel Quellcode als Patch hinzugefügt und mit ihm kompiliert. Der Treiber initialisiert die entsprechende „chip select“ Leitung des externen Speicher Kontrollers für das „Master“ FPGA. Dieser Kontroller bietet auf dem verwendeten Prozessor die Möglichkeit bis zu vier verschiedene Speicher anzusteuern. Dafür gibt es zu jeder „chip select“ Leitung einen eigenen Register Satz.

Eingestellt werden muss die Art des Speicherzugriffes. In diesem Fall wird synchron mit 16 Bit und gemultiplexten Adress-/Datenleitungen verwendet. Das bedeutet das es eine zusätzliche „address valid“ Leitung und „clock“ Leitung gibt.

Neben der Art muss auch das Timing für den Zugriff parametrisiert werden. Ausgegangen wird dabei von dem 100 MHz Haupttakt der Speicher Kontrollers. Dieser kann um den Faktor 1, 2, 3 oder 4 geteilt werden um den Takt für den externen Speicher zu

bestimmen. Der Takt für das FPGA wird auf 50 MHz eingestellt. Die Zeitpunkte für die diversen Aktionen während eines Zugriffes werden ab Beginn des Zugriffes in Schlägen des Haupttaktes bestimmt. Die zu bestimmenden Zeitpunkte sind der Wechsel von Adresse zu Daten, das Abtasten der Datenleitungen beim Lesezugriff und das Abtasten der Datenleitungen durch den externen Speicher beim Schreibzugriff. Bei den beiden Abtastzeitpunkten muss darauf geachtet werden dass sie mit einer steigenden Taktflanke des Speichertaktes zusammen fallen. Die Dauer des „adress valid“ Signals, die Ausgabe des „read/write“ Signals und das Setzen der „chip select“ Leitung vor und nach dem Zugriff werden ebenfalls konfiguriert.

Durch die getroffenen Einstellungen ist eine Datentransferrate mit dem FPGA in der Größenordnung von 20 MB/s möglich.

6.2.1.3 Der Kernel Treiber

Der Treiber stellt die Schnittstelle zwischen dem „user space“ Programm und der Hardware dar. Um aus dem Treiber Modul auf das FPGA zugreifen zu können muss ein „Platform Device“ angelegt werden welches bei seiner Initialisierung einen Speicherbereich reserviert, diesen im verwendbaren Adressraum abbildet und dem entsprechenden „chip select“ des externen Speicher Kontrollers zuordnet. Ist dies erledigt kann mit den IO „read/write“ Funktionen direkt auf das FPGA zugegriffen werden.

Das Transferieren von großen Datenmengen beim Messen ist auf diese Weise nicht sinnvoll da der Prozessor dabei blockiert werden würde. Dies wird durch die Verwendung des DMAs des Prozessors umgangen. Dafür muss das „Platform Device“ auch einen DMA Kanal reservieren. Der auf dem AM3559 Prozessor verfügbare DMA wird vom Hersteller als EDMA bezeichnet.

Zum Starten eines DMA Transfers müssen die physikalischen Adressen von Quelle und Ziel bekannt sein. Wird mit virtuellen Adressen gearbeitet muss darauf geachtet werden dass der DMA Transfer nicht über eine „Page“ hinausschreibt da im Allgemeinen nicht gewährleistet ist dass ein angeforderter Speicherbereich zusammenhängend im Speicher liegt. Die Größe einer normalen „Page“ liegt bei 4096 Bytes. Es empfiehlt sich, wenn ein Speicherbereich für einen DMA Transfer angefordert wird, eine spezielle Art des Allokierens zu verwenden, bei der sichergestellt wird, dass der Speicher linear adressierbar ist. Dabei sollte bedacht werden dass, wenn ein größerer Speicherbereich benötigt wird dieser möglichst nach dem Systemstart reserviert wird, da die Fragmentierung des Arbeitsspeichers mit der Laufzeit zunimmt.

Der Treiber wird als „Character Device“ implementiert. Dafür müssen Funktionen zur Verfügung gestellt werden welche einen Dateizugriff abbilden. Bei der Initialisierung wird eine Datei mit dem Namen „falcon“ im „/dev“ Dateisystem erzeugt. Auf diese kann lesend und schreibend zugegriffen werden. Die Datei ermöglicht einen Low-Level Zugriff auf den FPGA Speicher. Beim Lesen von der Datei wird ein String mit der aktuellen Adresse und deren Speicherinhalt zurückgegeben. Beim Schreiben auf die Datei kann entweder nur eine Adresse geschrieben werden um damit die aktuelle Adresse zu setzen oder eine Adresse und ein Wert um diesen in den Speicher zu schreiben. Um zusätzliche Informationen während der Messung auslesen zu können wird ebenso ein Eintrag im „/proc“ Dateisystem erstellt.

Neben der Schnittstelle über die Datei Operationen ist auch ein „ioctl“ Befehlssatz implementiert um die Messung zu Starten beziehungsweise zu Stoppen und dem Treiber weitere Parameter mitgeben zu können. Implementiert sind folgende Befehle.

- Messung starten
- Messung stoppen
- IP Adresse und Port Nummer des UDP Servers setzen
- Anzahl der verwendeten Slave Karten setzen
- Signalgenerator konfigurieren
- Signalgenerator aktivieren
- Signalgenerator deaktivieren
- FPGA Versionsnummer auslesen
- Treiber Versionsnummer auslesen
- Digitale Eingänge auslesen
- Digitale Ausgänge setzen
- Überstromüberwachung aktivieren

Vor dem Start der Messung muss in dem Treiber die IP Adresse sowie der Ziel Port des UDP Servers mit Hilfe des „ioctl“ Befehlssatzes gesetzt werden.

Beim Start der Messung wird ein externer Interrupt freigeschaltet der auf die „Packet bereit“ Leitung reagiert. Die entsprechenden Register im FPGA werden zum Starten beschrieben. Löst der Interrupt aus wird ein DMA Transfer für ein Paket von Strom- und Spannungsmesswerten gestartet. Zusätzlich überprüft die Interrupt Service Routine die Leitung „Temperatur Paket bereit“. Ist diese gesetzt wird ein Flag zum Auslesen des Temperatur Paketes gesetzt.

Nach Beendigung des DMA Transfers wird eine „Call Back“ Routine aufgerufen. Ist das Temperatur Flag gesetzt wird bei den nächsten 518 Aufrufen der Routine jeweils ein 16 Bit Wort des fertigen Temperatur Paketes gelesen.

Das Temperatur Paket wird nicht als Ganzes mit einem zweiten DMA Transfer ausgelesen da sich dies im Falle von 64 Kanälen zwischen den zyklischen Transfers der Strom- und Spannungswerte nicht ausgehen würde.

Prinzipiell wäre es möglich in dieser „Call Back“ Routine die Daten sofort zu versenden. Allerdings wird das Versenden der Daten von dem in Linux implementierten Ethernet Stack erledigt und es ist nicht garantiert, dass dies in Echtzeit passiert. Würde das Versenden einmal nicht schnell genug erfolgen, würden Messdaten verloren gehen.

Sinnvoller ist es daher die Messdaten in einen FIFO Puffer zu kopieren der von einem Kernel „Threat“ im Hintergrund ausgelesen wird welcher für das Versenden zuständig ist. Dieser „Threat“ wird mit der Messung gestartet und überprüft zyklisch ob Daten im Puffer bereit zum Senden sind. Dadurch wird die Echtzeit Anforderung an das Versenden der Daten eliminiert.

Der FIFO wird als Ringpuffer implementiert. An den Puffer wird die Anforderung gestellt „Threat Safe“ zu sein da jede Leseoperation zu jedem Moment von einer Schreiboperation des Interrupts unterbrochen werden kann. Dies könnte durch die Verwendung eines „Locking Mechanismus“ der vom Kernel zur Verfügung gestellt wird realisiert werden. Da dies zum Teil Performance Einbußen mit sich bringen würde wird stattdessen ein Puffer implementiert dessen Logik darauf beruht, dass der Schreib- und Lesezeiger immer nur an einer Stelle in den Zugriffsfunktionen verändert wird. Das Verändern passiert durch die Zuweisung eines 32 Bit Wertes was eine atomare Operation auf dem verwendeten Prozessor darstellt. Damit ist es möglich den Ringpuffer im Interrupt ohne Risiko verwenden zu können.

Zum Stoppen des Messvorganges werden die entsprechenden Register im FPGA gelöscht und der Interrupt deaktiviert. Sollten sich zu diesem Zeitpunkt noch Daten im Puffer befinden werden diese von dem Hintergrund „Threat“ versendet bevor sich dieser selbst deaktiviert.

6.2.1.4 Die Kommunikationsprotokolle

Wie in 3.2.3 beschrieben werden zwei verschiedene Protokolle benötigt. Das auf XML basierende Protokoll soll eine möglichst einfache Implementierung der „Client“ sowie der „Server“ Seite ermöglichen.

Das XML Protokoll basiert auf Rahmen die übertragen werden. Jeder Rahmen kann eine beliebige Anzahl an Kommandos beinhalten. Nach der Abarbeitung der Kommandos antwortet das System mit einem Rahmen welcher die Ergebnisse der einzelnen Kommandos enthält.

Der Rahmen sowie jedes Kommando entspricht einem XML „Tag“. Damit ein Antwort Rahmen eindeutig dem auslösenden Kommando Rahmen zugeordnet werden kann wird dem „XML Tag“ des Rahmens ein Metadaten Feld mit einer Identifikationsnummer hinzugefügt.

Der Typ eines Kommandos wird ebenso durch ein Metadaten Feld bestimmt. Jeder Kommando Typ kann eine Anzahl von Parametern besitzen. Die Parameter werden dem Kommando „Tag“ als eigene Parameter „Tags“ hinzugefügt.

Der Antwort Rahmen enthält einen Status „Tag“ welcher angibt ob es möglich war den kompletten Kommando Rahmen auszuführen. Dem Status „Tag“ folgen die Ergebnisse beziehungsweise Fehlercodes der einzelnen Kommandos.

Die Messdaten werden in binären Paketen vom System versendet. Jedes Paket hat einen „Header“ bestehend aus einem 64 Bit Zeitstempel und einem 32 Bit Metadaten Feld.

Es gibt zwei verschiedene Arten von Paketen. Zum einen die Messwerte der Spannungs- und Stromkanäle und zum anderen die Ergebnisse der Temperaturmessung. Unterschieden werden die beiden Arten durch ein Bit im Metadaten Feld.

Nach den Metadaten folgen die Messwerte. Die Anzahl der Daten hängt von der Anzahl der verwendeten Kanäle ab. Die Messwerte aller Kanäle werden nacheinander angeordnet. In einem Paket sind die Werte nach ihrem Abtastzeitpunkt geordnet. Die Basis der Abtastzeitpunkte ist Strom- und Spannungsmessung. Die Nummer des Abtastzeitpunktes im Paket für Strom und Spannung wird aus dem Zeitstempel und der Anzahl der Kanäle berechnet. Im Paket für die Temperatur entspricht der Zeitstempel dem Abtastzeitpunkt der ersten Werte der Temperaturkanäle 0, 4, 8 und 12. Die nächsten Kanäle 1, 5, 9 und 13 haben ein Offset von 320. Die Nummerierung ergibt sich aus der Funktionsweise des SPI Bus „Master“ Moduls in den „Slave“ FPGAs, siehe 3.2.1.3

Mit Hilfe der Nummer des Abtastzeitpunktes können die Werte der Temperaturmessung dem Abtastzeitpunkten der Strom- und Spannungsmessung zugeordnet und somit synchronisiert werden.

6.2.1.5 Die „User Space“ Software

Für diese Software wurde ein Grundgerüst geschrieben, welches Klassen für das Loggen von Information im Linux System Log, erweiterte Standard String Funktionalität und Netzwerk „Sockets“ sowie Basisklassen für Kommandos und Parameter bietet.

Die Software implementiert grundsätzlich einen TCP Server. Dafür wird ein „Socket“ vom Betriebssystem angefordert und in den Zustand „Listening“ versetzt. Verbindet sich ein „Client“ mit diesem Socket wird dies von der Software detektiert. Ab diesem Moment ist der Server bereit Befehle zu empfangen.

Wird eine Nachricht empfangen muss diese geparst werden. Dafür wird die freie XML Bibliothek „tinyXml2“ verwendet. Konnte die Nachricht erfolgreich geparst werden steht ein „Tree“ aus den enthaltenen Elementen zu Verfügung.

Für den Befehlssatz werden die einzelnen Befehle von einer Befehls Basisklasse abgeleitet. Diese Basisklasse implementiert eine Liste in der Objekte für die einzelnen Parameter registriert werden können. Für jeden Typ Parameter gibt es wiederum eine von einer gemeinsamen Basisklasse abgeleitete Klasse. Diese Klassen stellen ein Interface zum Auslesen ihres Wertes aus dem XML „Tree“ bereit. Implementiert sind folgende Parameter Typen.

- unsigned integer 16 Bit
- unsigned integer 32 Bit
- float
- strings

Eine Befehlsklasse registriert in ihrem Konstruktor in ihrer Parameter Liste die zu erwartenden Parameter.

Die Befehls Basisklasse beinhaltet auch eine virtuelle Methode, die in den abgeleiteten Klassen die eigentliche Ausführung des Befehls implementiert, und eine Methode zum Einlesen der Werte ihrer Parameter. Beim Einlesen der Werte wird über alle Einträge in der Parameter Liste iteriert und deren Einlese Funktion ausführt. Zusätzlich wird eine Methode bereitgestellt mit der die Werte des letzten empfangenen Kommandos des jeweiligen Typs ausgelesen werden können. Diese Methode dient zur Implementierung eines Kommandos zum Auslesen der aktuellen Einstellungen.

Folgende Befehle werden von der Basisklasse abgeleitet:

- Aktivieren der Messung
- Konfigurieren der Messung

- Abfragen der Version
- Konfigurieren des Signalgenerators
- Aktivieren des Signalgenerators
- Setzen der Ausgänge
- Lesen der Eingänge
- Konfigurieren der Überstromüberwachung
- Auslesen der letzten korrekten Parameter eines jeden Kommandos

In der virtuellen Methode zum Ausführen des Kommandos greift die Software über das „ioctl“ Interface auf den Treiber zu um das FPGA entsprechend steuern zu können.

Alle Befehlsklassen sind durch einen privaten Konstruktor als „Singleton“ ausgelegt. Die Befehlsklassen implementieren eine Methode mit der die Referenz auf ein statisches Objekt dieser Klasse angefordert wird. Die Referenzen der Befehlsklassen werden in einer gemeinsamen „Map“ mit ihrer Kommando ID als Schlüssel abgelegt.

Konnte der XML „Tree“ erstellt werden kann aus ihm die Kommando ID ausgelesen werden und damit das entsprechende Element in der „Map“ gefunden werden. Dieses ist in der Lage die Werte der Parameter aus dem restlichen „Tree“ auszulesen. Nach erfolgreichem Abschluss dieses Vorganges kann der Befehl ausgeführt werden.

Beim Aufbau einer Verbindung durch einen „Client“ wird dem Kernel Treiber automatisch dessen IP Adresse als UDP Server Adresse mitgeteilt. Wird von dem „Client“ der Befehl zum Starten der Messung gesendet wartet die Software bis der „Client“ die TCP Verbindung trennt um dem Treiber den Start der Messung mitzuteilen. Die Messung wird gestoppt wenn eine neue „Client“ TCP Verbindung aufgebaut wird. In der neuen Verbindung hat der „Client“ die Möglichkeit den automatischen Start der Messung wieder zu deaktivieren.

Neben der Ausführung unterschiedlichster Befehle kann die Software auch verschiedene Parameter abspeichern und laden. Dies geschieht im vom Linux angebotenen Dateisystem. Da die Software ohnehin die XML Bibliothek benötigt werden die Systemparameter ebenfalls im XML Format abgespeichert. Für jeden Typ von Systemparametern ist wiederum eine eigene Klasse implementiert. Zusätzlich zu diesen Klassen gibt es eine Containerklasse, welche eine Gruppe von Parameter Objekten und Objekte der Containerklasse selbst beinhalten kann. Damit ist eine Gruppierung der Parameter möglich was zu einer besseren Lesbarkeit der erzeugten XML Datei führt. Zum Laden beziehungsweise Speichern der Parameter wurde ein „Visitor“ Pattern ver-

wendet. Dadurch kann das Laden und das Speichern in zwei getrennten „Visitor“ Klassen implementiert werden.

6.3 Ergebnisse

Ergebnis der Software Entwicklung sind ein auf die Hardware angepasster Linux Kernel, eine Server Applikation sowie ein Treiber für das an das Prozessor Modul angeschlossene „Master“ FPGA.

Durch ein „Script“ im „rc.d“ System des Linux Systems wird bei Systemstart der Kernel Treiber geladen und anschließend der Server gestartet. Sollte sich der Server unvorhergesehen schließen startet das „Script“ diesen umgehend neu.

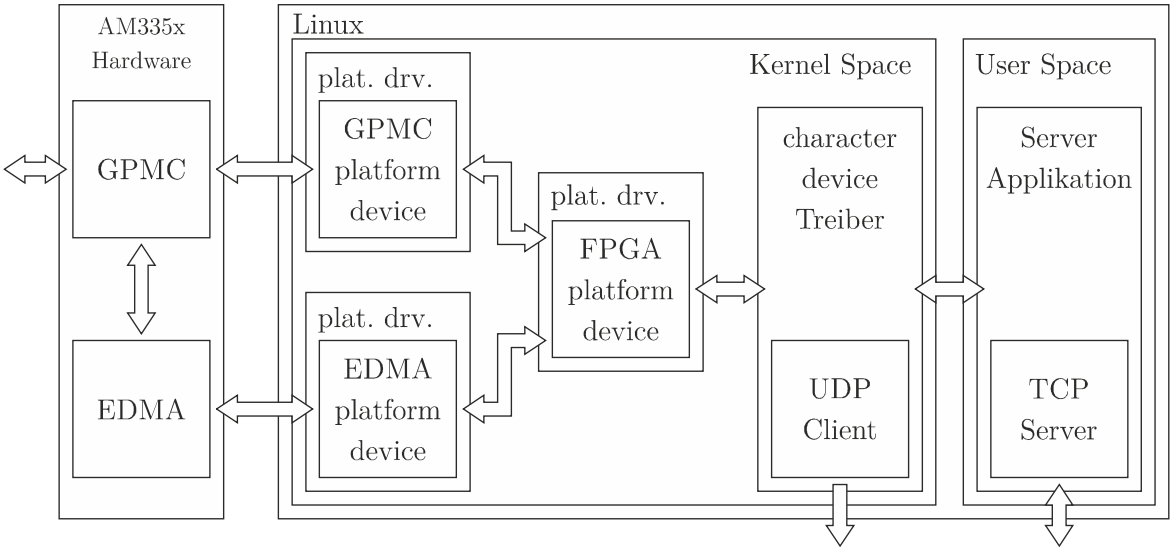

Abbildung 40: Interaktion der verschiedenen Software Komponenten

Abbildung 40 zeigt die Interaktion der einzelnen Software Komponenten untereinander sowie den Zugriff auf die Hardware des Prozessors. Die Abstraktionsebene steigt dabei von links nach rechts.

7 Ergebnis

Ergebnis der Masterarbeit ist ein funktionsfähiger Prototyp, inklusive der zum Betrieb notwendigen Software, mit dem Impedanzmessungen an einer Brennstoffzelle durchgeführt wurden.

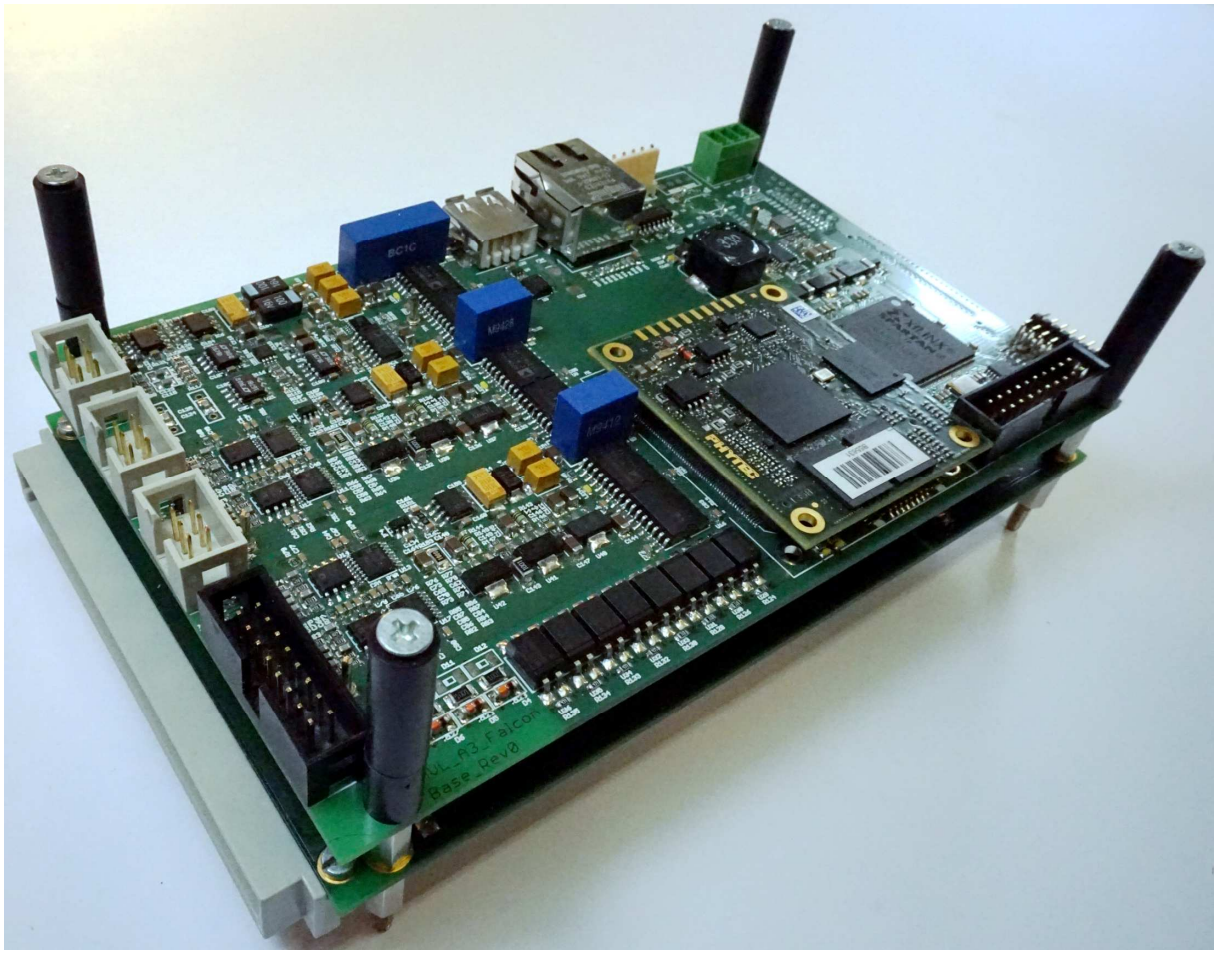

Abbildung 41: Aufbau des Gesamtsystems mit Basis und „Slave“ Platine

Abbildung 41 zeigt den ersten Prototypen von insgesamt drei aufgebauten Systemen. Links vorne ist der Anschluss für die Adapterplatine auf der „Slave“ Platine zu sehen. Darüber befinden sich auf der Basis Platine die Anschlüsse für die digitalen Ein- und Ausgänge, die beiden zusätzlichen Strom- und Spannungskanäle und der Ausgang des Signalgenerators.

7.1 Messung an einer PEM Brennstoffzelle

Um die Funktion des Gesamtsystems zu testen wurden Messungen an einer Brennstoffzelle auf einem Prüfstand durchgeführt.

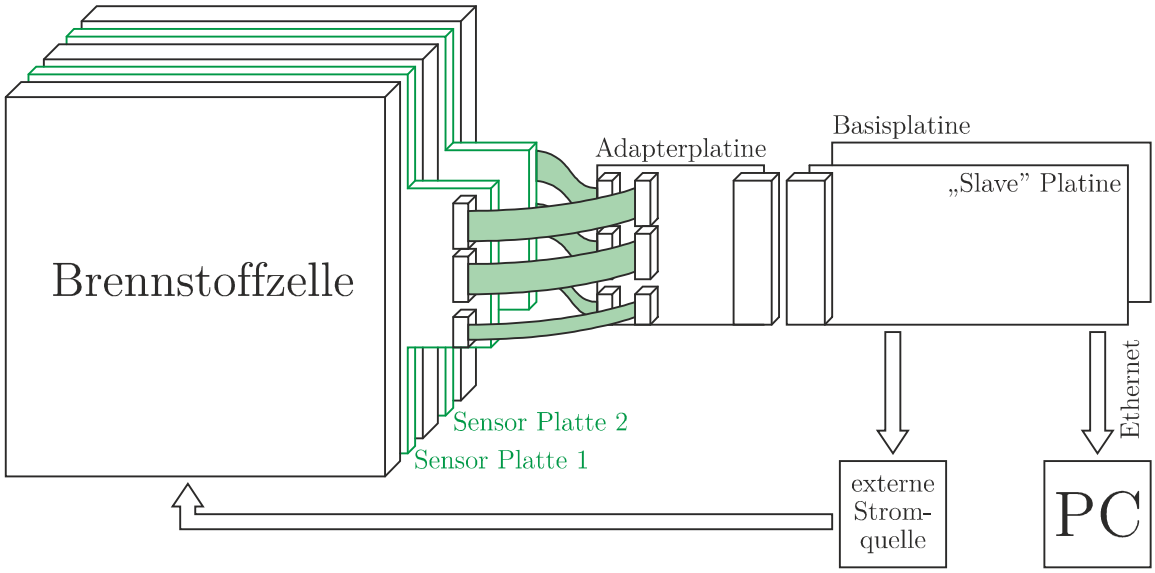

Abbildung 42: Aufbau der Messung an einer Brennstoffzelle [11]

Der Aufbau der Messung wird schematisch in Abbildung 42 dargestellt. Das System steuert die externe Stromquelle an und sendet die Daten an den „Client" Computer.

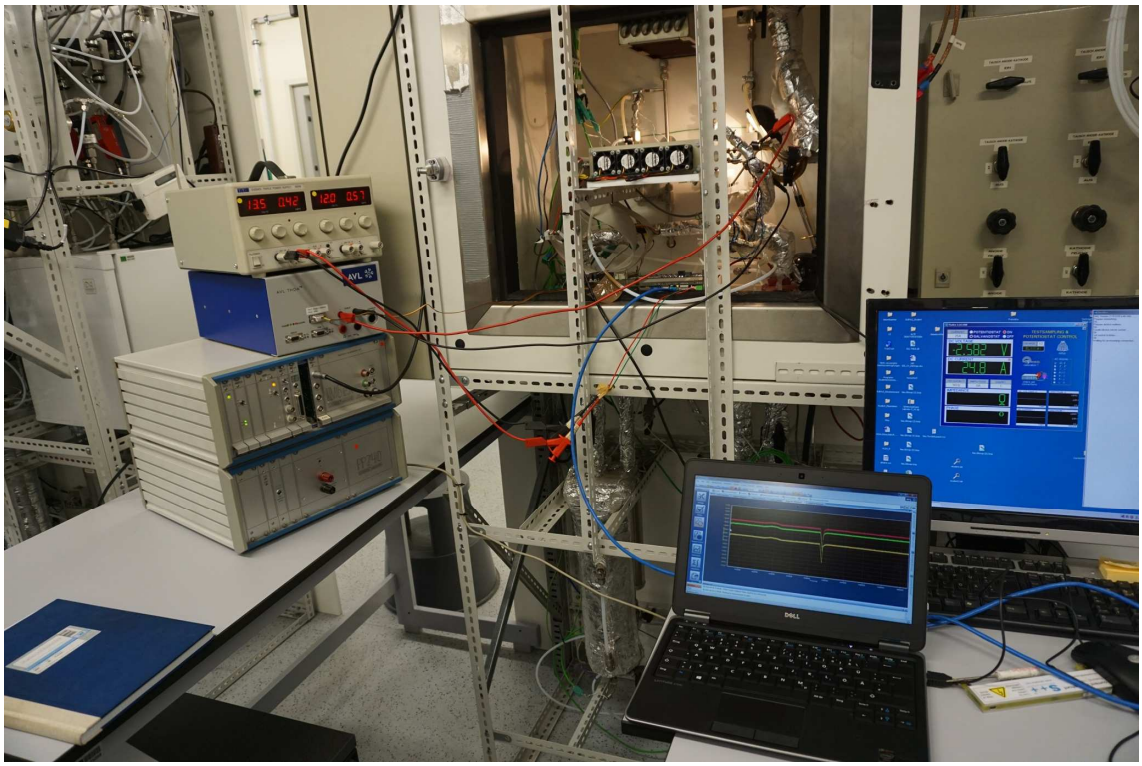

Abbildung 43: Messung an einer Brennstoffzelle auf einem Prüfstand

Der reale Aufbau auf einem Prüfstand ist in Abbildung 43 im Labor des Institutes für Chemische Verfahrenstechnik und Umwelttechnik zu sehen. Es wurde an einem PEM Brennstoffzellenstapel mit fünf Zellen und Sensorplatten mit 16 Segmenten gemessen.

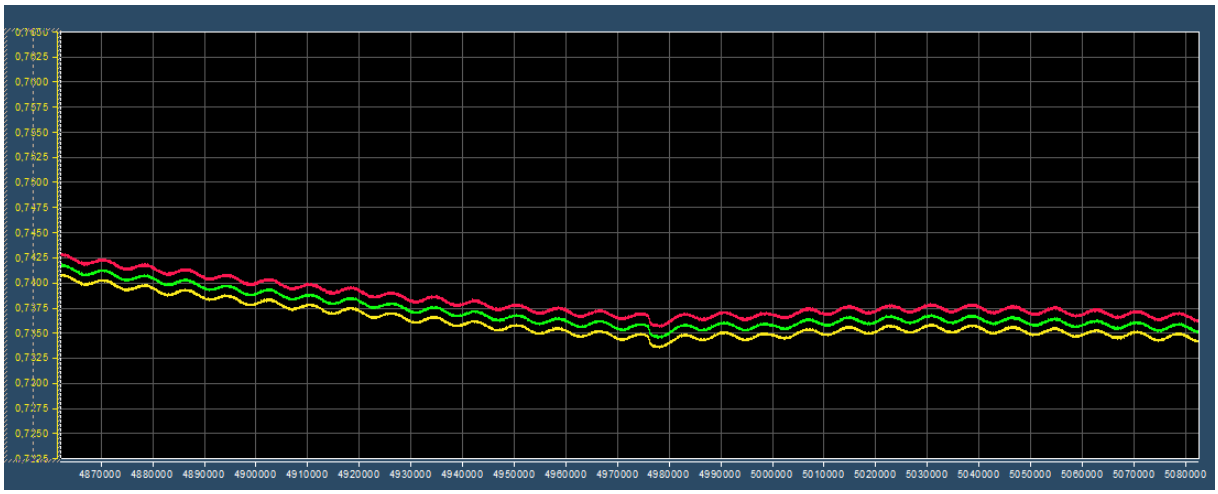

Abbildung 44: Sinkende Segmentspannungen vor einem „Purge“ Vorgang

Y Achse entspricht Spannung in Volt, X Achse entspricht der Zeit in 62,5 µs. Gezeigt wird die Spannung von drei Segmenten.

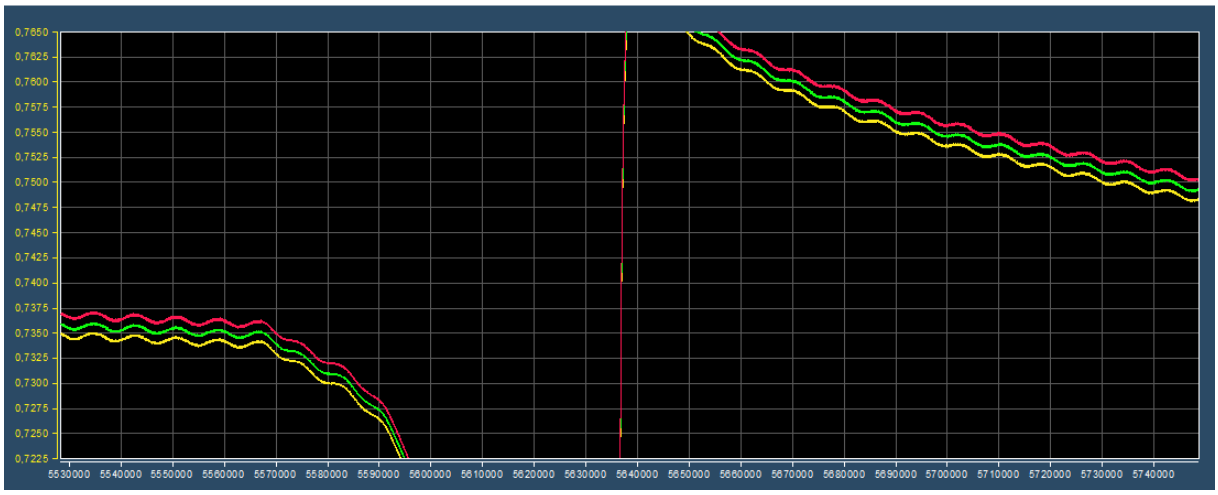

Abbildung 45: Segmentspannungen während eines „Purge“ Vorganges

Y Achse entspricht Spannung in Volt, X Achse entspricht der Zeit in 62,5 µs. Gezeigt wird die Spannung von drei Segmenten.

Auf dem „Client“ Computer im Vordergrund läuft die Analyse Software, welche die empfangenen Messwerte graphisch darstellt. Die Messwerte von Strom und Spannung

können entweder „online“ betrachtet werden oder für weitere rechenintensivere Analysen, wie Impedanz Spektrographie oder THD Bestimmung, abgespeichert werden.

Abbildung 44 und Abbildung 45 zeigen die Ergebnisse erster Messungen an einer Brennstoffzelle. Die Abtastrate wurde durch den „prescaler“ auf 16 kHz reduziert, siehe 5.3.2.1. Mit Hilfe des externen Leistungsverstärkers wurde ein Wechselstrom mit zwei Hertz zusätzlich zum konstanten Laststrom in den Brennstoffzellenstapel eingeprägt. Der „große“ Offset zwischen den drei Spannungen ist auf die fehlende Offset Kalibrierung zurück zuführen.

Der Brennstoffzellenstapel wird im „dead end“ Modus betrieben. Das bedeutet, dass es keinen kontinuierlichen Durchfluss von Wasserstoff gibt. Ein Stapel besitzt einen Einlass sowie einen Auslass für die Wasserstoffzufuhr. Im „dead end“ Betrieb wird der Auslass durch ein Ventil verschlossen. Dadurch wird der gesamte Wasserstoff verbraucht und nur die verbrauchte Menge muss nachgeliefert werden. Allerdings sammelt sich Wasser auf der Anodenseite wodurch die Leistung der Zelle abnimmt. Um dieses Wasser aus der Zelle zu spülen wird das Ventil am Auslass in festgelegten Intervallen geöffnet, dieser Vorgang wird „Purge“ genannt. Das Sinken der Zellspannung durch die Abnahme der Leistung vor jedem „Purge“ ist in Abbildung 44 und Abbildung 45 zu sehen.

Bei dem eingeprägten Wechselstrom mit einer Frequenz von zwei Hertz konnten nichtlineare Verzerrungen der Spannungssignale beobachtet werden, siehe Abbildung 44. Dieser Effekt nimmt kurz vor einem „Purge“ Vorgang zu.

Abbildung 45 zeigt den Spannungsverlauf während eines „Purge“ Vorganges. Es ist zu erkennen wie sich die Zellspannung nach dem „Purge“ wieder erholt.

8 Abkürzungen

AAF	Anti-Aliasing Filter
ADC	Analogue to Digital Converter
ASIC	Application-Specific Integrated Circuit
CAD	Computer Aided Design
CPLD	Complex Programmable Logic Device
DAC	Digital to Analogue Converter
DDS	Direct Digital Sythesizer
DMA	Direct Memory Access
EAGLE	Einfach Anzuwendender Grafischer Layout-Editor
EDMA	Enhanced Direct Memory Access
ESD	Electro Static Discharge
FIFO	First In First Out
FPGA	Field Programmable Gate Array
GPIO	General Purpose In/Output
IC	Integrated Circuit
JTAG	Joint Test Action Group
LSB	Least Significant Bit
LUT	Look Up Table
MISO	Master In Slave Out (SPI Signal)
MOSI	Master Out Slave In (SPI Signal)
MSB	Most Significant Bit
PCB	Printed Circuit Board
PEM	Proton Exchange Membrane
PLL	Phase Locked Loop
RAM	Random Access Memory
SAR	Successive Approximation Register
SMD	Surface Mounted Device
SPI	Serial Peripheral Interface
TCP	Transmission Control Protocol
THD	Total Harmonic Distortion
UDP	User Datagram Protocol
VHDL	Very High Speed Integrated Circuit Hardware Description Language
XML	Extensible Markup Language

9 Abbildungsverzeichnis

10 Literaturverzeichnis

[1] F. Barbir, PEM Fuel Cells: Theory and Practice, San Diego: Academic Press, 2005.

[2] R. Klambauer, „Impedance Measurement," in *Poster at Advanced Studies of Polymer Electrolyte Fuel Cells 6th International Summerschool*, Yokohama, 2013.

[3] J. R. Macdonald, Impedance Spectroscopy Theory, Experiment, and Applications, New Jersey: John Wiley & Sons, Inc., 2005.

[4] B. Eichberger, „Impedance Spectroscopy Measurement Techniques for Fuel Cell Analysis," in *Advanced Studies of Polymer Electrolyte Fuel Cells 6th International Summerschool*, Yokohama, 2013.

[5] A. V. Oppenheim und R. W. Schafer, „Periodic Sampling," in *Discrete-Time Signal Processing*, New Jersey, Prentice-Hall, Inc., 1999, p. 140.

[6] A. V. Oppenheim and R. W. Schafer, "Frequency-Domain Representation of Sampling," in *Discrete-Time Signal Processing Second Edition*, New Jersey, Prentice-Hall, Inc., 1999, pp. 142-144.

[7] U. Tietze und C. Schenk, „Programmierbare logische Bauelemente," in *Halbleiter-Schaltungstechnik*, Berlin, Springer, 1993, pp. 753-765.

[8] J. Vankka und K. A. Halonen, Direct Digital Synthesizers, Dordrecht: Kluwer Academic Publisher, 2001.

[9] U. Tietze und C. Schenk, „Aktive Filter," in *Halbleiter-Schaltungstechnik*, Berlin, Springer, 1993, pp. 839-863.

[10] R. J. Baker, „The Successive Approximation ADC," in *CMOS Circuit Design Layout and Simulation*, New Jersey, John Wiley & Sons, Inc., 2010, pp. 1003-1005.

[11] S. Weinberger, Gebetsroither, R. Klambauer, Hacker, Eichberger und Karpenko-Jereb, „Real- Time Monitoring of Fuel Cell Stacks," in *Poster at A3PS - Eco-Mobility 2013 - International Conference*, Wien, 2013.

[12] C. M. Kozierok, "IP Datagram Size and the Underlying Network Frame Size," in *The TCP/IP Guide: A Comprehensive, Illustrated Internet Protocols Reference*, San Francisco, No Starch Press, 2005, p. 340.

[13] G. Likely, “www.kernel.org,” [Online]. Available: www.kernel.org/doc/Documentation/devicetree/usage-model.txt. [Accessed 1 9 2014].

[14] Analog Devices, *Datenblatt AD7767.*