



Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Hauke Schröder

Ein SoC-System zur Indoor-Positionsermittlung mittels
Beschleunigungs-, Gyro- und Kompassensordaten

Hauke Schröder

Ein SoC-System zur Indoor-Positionsermittlung mittels
Beschleunigungs-, Gyro- und Kompassensordaten

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. rer. nat. Wolfgang Fohl
Zweitgutachter: Prof. Dr.-Ing. Bernd Schwarz

Abgegeben am 01. August 2011

Hauke Schröder

Thema der Bachelorarbeit

Ein SoC-System zur Indoor-Positionsermittlung mittels Beschleunigungs-, Gyro- und Kompassensordaten

Stichworte

SoC, System-on-Chip, Virtex-5, ML507, Digitalkompass, Gyroskop, Drehratensensor, Beschleunigungssensor, MicroBlaze, HMC5843, ITG-3200, ADXL345

Kurzzusammenfassung

Diese Bachelorarbeit befasst sich mit der Aufgabe mit Hilfe von Inertial- und Magnetfeldsensorik translatorische und rotatorische Beschleunigungskräfte und magnetische Flussdichten zu messen und diese Daten mit Hilfe eines Embedded Systems in Echtzeit auszuwerten.

Die Entwicklung des Embedded-Systems gliedert sich in drei Teilabschnitte, wobei sich der erste Teil des Entwicklungsprozesses mit der Konfiguration des MicroBlaze Softcore RISC-Prozessors beschäftigt. Der zweite Entwicklungsschritt ist die Implementierung der C-Routinen zur Sammlung und Auswertung der Daten, welche nach Fertigstellung des Systems im dritten Schritt auf Genauigkeit und Fehler hin untersucht werden. Als Plattform für das Embedded-System wird ein Xilinx Virtex-5 FPGA verwendet.

Das entwickelte System unterstützt die Messung von Rotationen und Translationen in allen drei räumlichen Ebenen, die Auswertung von Roll und Nick Winkeln, eine filterlose Approximation der Geschwindigkeit, sowie die daraus resultierende zurückgelegte Distanz in der X-Y-Ebene und die Berechnung des Azimiuth-Winkels (magnetische Ausrichtung des Sensors in Bezug zum Erdmagnetfeld) in Echtzeit.

Title of the paper

An SoC-Indoor-Positioning-System using digital acceleration, gyroscope and compass sensors

Keywords

SoC, System-on-Chip, Virtex-5, ML507, digital compass, gyroscope, accelerometer, MicroBlaze, HMC5843, ITG-3200, ADXL345

Abstract

This thesis describes the development of an embedded-system based on a Xilinx Virtex-5 FPGA, which evaluates rotational and translational acceleration, in real-time, and magnetic flux density, which is measured by inertial sensors and a magnetic field sensor.

The development of this embedded-system is structured in three steps, detailed as follows. The first step deals with the configuration of the MicroBlaze Softcore RISC-Processor, the second step handles the implementation of the C-routines gathering and processing the sensor data. Once the embedded-system has been accomplished the data is analyzed considering accuracy and quality.

The system is able to measure rotational and translational acceleration in all three planes, to calculate roll and pitch-angles, an approximation of the unfiltered velocity as well as the resulting distance in the X-Y-plane and the azimuth angle (magnetic orientation of the sensor relating to the earth's magnetic field) in real time.

Danksagungen

Ich möchte mich bei allen Menschen, Verwandten, Freunden und Kommilitonen bedanken, die mich während des Bachelorstudiums unterstützt, mein schwaches Nervenkostüm erduldet und meine schlechte Laune ertragen haben.

Mein besonderer Dank gilt Prof. Dr. Wolfgang Fohl und Prof. Dr.-Ing. Bernd Schwarz.

Für die moralische und finanzielle Unterstützung danke ich meinen Eltern Ortrud und Erwin Schröder und meiner Freundin Sandra Krafft, ohne die ich diesen Schritt in meinem Leben nicht hätte machen können!

Hauke Schröder, Juli 2011

“Die dünnen Falten, die sich in die Stirn der Denker fressen, bekamen nicht selten ihre Nahrung durch den Zweifel.“

Christa Schybol

Inhaltsverzeichnis

1	Einführung	1
2	Funktions- und Anforderungsdefinition	4
2.1	Funktionale Anforderungen	4
2.2	Hard- und Softwareanforderungen	4
2.2.1	Minimierung von Messabweichungen	5
2.2.2	Portabilität	5
2.2.3	Userinterface	6
2.2.4	Dateninterface	6
3	Entwicklungsumgebungen zur Erstellung des Embedded Systems	7
3.1	Evaluation Board - Virtex-5 ML507	7
3.2	Xilinx Platform Studio	9
3.3	Xilinx Software Development Kit	9
4	Magnetfeld- und Inertialsensorik	11
4.1	Einführung	11
4.2	Digitalkompass - HMC5843	12
4.2.1	HMC5843 - Funktionsweise	12
4.2.2	HMC5843 - Datenerfassung, Messabweichungen und Kompensation .	14
4.3	Accelerometer - ADXL345	16
4.3.1	ADXL345 - Funktionsweise	17
4.3.2	ADXL345 - Datenerfassung, Messabweichungen und Kompensation .	18
4.4	Gyroskop - ITG-3200	20
4.4.1	ITG3200 - Funktionsweise	20
4.4.2	ITG3200 - Datenerfassung, Messabweichungen und Kompensation .	23
4.5	Datenauswertung	24
4.5.1	Neigung	24
4.5.2	Ausrichtung	25
5	Embedded System zur Indoor-Positionsermittlung	27

5.1	Einführung	27
5.2	Hardwareentwurf	28
5.2.1	Konfiguration des MicroBlaze	29
5.2.1.1	LCD-Display und Oszilloskop	34
5.2.1.2	I ² C EEPROM	34
5.2.2	Synthese	35
5.3	Software	37
5.3.1	Treiber für Hardware-Module	38
5.3.2	Software-Routinen zur Verarbeitung der Sensordaten	39
5.3.3	Datencontainer	41
6	Test	42
6.1	Gyroskop	42
6.1.1	Testaufbau	42
6.1.2	Testergebnis	43
6.2	Accelerometer	46
6.2.1	Testaufbau	46
6.2.2	Testergebnis	47
6.3	Kompass	50
6.3.1	Testaufbau	50
6.3.2	Testergebnis	51
7	Fazit	53
	Abbildungsverzeichnis	57
	Literaturverzeichnis	61
A	Inhalt der CD-Rom	62
B	Xilinx Platform Studio Schemata	63
C	9 Degrees of Freedom Sensor Stick	69
D	Embedded System	74
E	Display- und Terminalausgaben des Systems	78

Kapitel 1

Einführung

Diese Bachelorarbeit befasst sich mit der Umsetzung des Konzepts mit Hilfe von Inertial- und Magnetfeldsensorik in einem SoC-System Angaben zur aktuellen Position und Ausrichtung machen zu können. Diese Angaben sollen für lineare Bewegungen und Rotationen in der horizontalen Ebene gemacht werden können.

Zur Umsetzung dieses Konzepts werden drei unterschiedliche Sensoren eingesetzt, welche durch einen I²C-Bus¹ an ein SoC-System² angeschlossen sind und von diesem ausgewertet werden. Für die Messung der Beschleunigung wird ein Beschleunigungssensor eingesetzt. Der Sensor misst die Beschleunigungen mit Hilfe von zwei veränderlichen Kondensatoren und wandelt das Ergebnis mittels Analog/Digital Konvertern in digitale Signale um. Rotationen und daraus resultierende Trägheitskräfte werden von einem Gyroskop gemessen und digitalisiert. Der Azimuthwinkel, also die Ausrichtung des Sensors in Bezug zum Erdmagnetfeld, wird mit Hilfe eines digitalen Kompasses bestimmt. Zur Messung bedient sich der Sensor dem Prinzip des anisotropen magnetoresistiven Effekts. Alle drei Sensoren legen die digitalisierten Daten im 2er-Komplement Format in jeweils zwei Registern pro Achse ab. Dieses Sensorarray wird häufig als 9 Degrees-of-Freedom- oder MARG³-Sensor Array bezeichnet. Sie sind die Erweiterung der IMUs⁴, welche lediglich Inertialsensorik nutzen, also Sensorik, die translatorische und rotatorische Beschleunigungen messen kann. Wird diesen drei Sensoren ein Auswertungssystem für die gelieferten Daten hinzugefügt wird von einem AHRS⁵ gesprochen.

¹I²C: Inter-Integrated Circuit - Ein von Philips Semiconductors entwickelter Datenbus

²SoC: System-on-Chip - Kombination unterschiedlicher Bauelemente und Integration mehrerer Funktionen auf einem Chip

³MARG: Magnetic, Angular Rate and Gravity

⁴IMU: Inertial Measurement Unit

⁵AHRS: Attitude and Heading Reference System - Ein Sensorarray mit Accelerometer, Gyroskop und Kompass und einem System zur Auswertung dieser Sensoren

Ein Gyroskop misst wie zuvor erwähnt Rotationen, mit deren Hilfe bei bekannten Startparametern auf die Ausrichtung in Bezug zum Erdmagnetfeld, also den Azimuthwinkel geschlossen werden kann. Gyroskope neigen zum Driften, was bedeutet, dass Messabweichungen akkumuliert werden und das Integrationsergebnis mit jedem Sample ungenauer wird. [2]

Ein ähnliches Verhalten weisen Beschleunigungssensoren auf, sodass eine filterlose Integration der gemessenen Werte zur Geschwindigkeitsmessung zwar möglich ist, allerdings kaum annehmbare Ergebnisse liefern kann und somit nicht sinnvoll nutzbar ist. Sowohl der Drehratensensor, als auch der Beschleunigungssensor werden durch die Beschleunigungen des anderen Sensors beeinflusst, was die Messwerte zusätzlich verfälscht. Die einzige, nicht driftende und nicht durch Beschleunigungen beeinflusste Komponente des Sensor Arrays ist das Magnetometer, welches wiederum leicht durch äußere Einflüsse, wie nahe gelegene ferromagnetische Objekte oder elektrische Felder gestört werden kann.

Um die Schwächen jedes einzelnen Sensors möglichst gut ausgleichen zu können, werden Orientierungfilter eingesetzt, mit denen die Messwerte der drei Sensoren zusammengefügt und so Fehler minimiert werden. Für solche Aufgaben wird häufig das Kalman Filter [47] verwendet, welches versucht mit Hilfe anspruchsvoller Mathematik und bekannten Messabweichungen der Sensorik, Rückschlüsse auf die Messabweichungen zu machen und so die Messergebnisse nachträglich zu korrigieren. Ein neuer und ressourcensparender Ansatz für ein Orientierungfilter wurde von Madgwick *et al* [43] veröffentlicht, das einen Kalman Filter mit einer Jacobi Matrix zur Reduktion des Rechenaufwandes nutzt. Zur Berechnung werden Quaternionen genutzt, welche einen Zahlenraum bilden, der häufig zur Berechnung von Rotationen im dreidimensionalen Raum genutzt wird. Mit Hilfe der Eulerschen Winkel kann die Orientierung des Sensors in Grad umgerechnet werden.

Die beiden zuvor genannten Filter wurden aus einer Vielzahl von Filtern ausgewählt, da diese die Kernproblematiken bei der Verarbeitung von digitalen Sensordaten eines MARG Sensor Arrays versuchen zu lösen, jedoch ist die Implementierung dieser Filter auf der gegebenen Plattform nicht ohne weiteres möglich. Beide Filter nutzen Variablen vom Datentyp *float* und einige Funktionen die auf Fließkomma-Arithmetik zurückgreifen, was nötig ist, um möglichst präzise Berechnungen sowohl bei sehr kleinen, als auch bei sehr großen Werten gewährleisten zu können. Bei der in dieser Bachelorarbeit verwendeten Plattform handelt es sich um ein Xilinx Virtex-5 FPGA, in das ein MicroBlaze Softcore RISC-Prozessor programmiert wird. Dieser Mikrocontroller ist zwar in der Lage Fließkomma-Arithmetik auszuführen, allerdings dauern diese Operationen, verglichen mit entsprechenden Festkomma-Operationen, lange. Aus Gründen mangelnder Genauigkeit der Festkomma-Zahlen und fehlender schneller Gleitkomma-Arithmetik wird auf die Implementierung eines der zuvor genannten Filter verzichtet.

Diese Arbeit gliedert sich in fünf Abschnitte, wobei sich der erste Abschnitt der Anforderungsanalyse (Kapitel 2), der zweite der Entwicklungsumgebung (Kapitel 3), der dritte der Sensorik (Kapitel 4), der vierte der Entwicklung von Hard- und Software (Kapitel 5) und der fünfte dem Test des Systems (Kapitel 6) widmet.

Kapitel 2

Funktions- und Anforderungsdefinition

Das folgende Kapitel beschäftigt sich mit der Konkretisierung der Anforderungen an das System. Nach Definition der funktionalen Anforderungen können die Anforderungen an das Hard- und Softwaresystem bestimmt werden.

2.1 Funktionale Anforderungen

Das Ziel des Projektes ist die Entwicklung eines SoC-Systems, das in Verbindung mit entsprechender Sensorik eine Standortänderung feststellen und auswerten kann. Diese Änderungen können Translationen und Rotationen sein, wobei Translationen als Beschleunigungen vom Sensor gemessen werden und in $\frac{m}{s^2}$ und Rotationen in $\frac{Grad}{s}$ gemessen werden.

Durch Integration der gemessenen Beschleunigung wird zunächst auf die Geschwindigkeit und anschließend durch erneute Integration auf die zurückgelegte Strecke geschlossen. Die gemessenen Rotationen werden von einem Gyroskop und einem Digitalkompass gemessen. Mit Hilfe der Daten des Gyroskops können die gemessenen Kompassdaten stabilisiert werden.

Die errechneten Daten werden dem Benutzer zur Verfügung gestellt, sodass zu jeder Zeit bekannt ist, wo er sich relativ zu seinem Ausgangsstandort gerade befindet. Es werden nur Translationen und Rotationen in horizontaler (X-Y-) Ebene betrachtet.

2.2 Hard- und Softwareanforderungen

In den nächsten Abschnitten werden die Anforderungen an die Hard- und Software in aus Sicht des Entwicklers sinnvolle Abschnitte unterteilt und erläutert.

2.2.1 Minimierung von Messabweichungen

Um die Qualität der Sensordaten zu erhöhen werden einige Schritte zur Minimierung der Messabweichungen unternommen. Um geeignete Schritte zur Fehlerminimierung einleiten zu können, müssen zunächst mögliche Quellen für Fehler identifiziert werden.

Um die Fehlerquellen einzugrenzen, wurde zunächst überprüft, dass die Datenübertragung via I²C und das System zur Verarbeitung der Daten fehlerfrei funktionieren, sodass Fehler lediglich bei der Datenerfassung auftreten können.

Bei der Messung von Beschleunigungen mit dem Beschleunigungssensor wirken sich Messabweichungen anders aus, als bei Messabweichungen des Kompasses und des Drehratensensors. Die gemessenen Daten des erstgenannten Sensors werden zur Ermittlung von Geschwindigkeiten einfach und Distanzen doppelt integriert, woraus resultiert, dass auch die auftretenden Messabweichungen einfach beziehungsweise doppelt integriert werden. Gleiches gilt für die einfach integrierten Daten des Gyroskops. Die Daten des Digitalkompasses hingegen werden nicht integriert, da lediglich die aktuelle Ausrichtung des Sensors zur Zeit des Samples relevant ist.

Alle drei Sensoren müssen zu Beginn kalibriert werden, um Messabweichungen zu reduzieren. Durch Kalibrierung der Sensoren können Messabweichungen ausgeglichen werden. Beispielsweise können durch Kalibrierung des Digitalkompasses Störungen des Erdmagnetfeldes kompensiert werden (siehe Kapitel 4.2.2). Messabweichungen können entweder systematisch oder zufällig sein. Handelt es sich um systematische Messabweichungen, haben diese einen Betrag und ein Vorzeichen und können, sofern sie bekannt sind, berichtigt werden. Sind die Messabweichungen systematisch, aber unbekannt, werden diese häufig in einer Messunsicherheitskomponente u_s zusammengefasst. Ist eine Messabweichung zufällig, schwanken die Messabweichungen in ihrem Betrag und Vorzeichen, was bedeutet, dass Versuche unter annähernd gleichen Bedingungen nicht zwingend gleiche Messwerte liefern.

2.2.2 Portabilität

Die Funktion des Systems ist das Erkennen und Bestimmen von Bewegungen. Um diese sowohl für nicht ortsfeste sich bewegende Objekte, als auch für Menschen erfüllen zu können, muss bei der Konzeption der Hardware auf einige Dinge geachtet werden:

1. Stromverbrauch: Das System muss über einen begrenzten Zeitraum mit einem Akku betrieben werden können.
2. Größe und Gewicht: Das System sollte kompakt und leicht sein, um die Bewegungen des Objekts beziehungsweise des Menschen an dem es befestigt ist, nicht einzuschränken.

2.2.3 Userinterface

Zur Steuerung des Systems werden die von dem Entwicklungsboard ML507 zur Verfügung gestellten Dip-Switches und Push-Buttons genutzt. Es wird eine Resetfunktion, mit der das ganze System neu gestartet werden kann und ein Funktion zum Herunterfahren des Systems angeboten. Die Daten können entweder auf dem Display des Virtex-5 Evaluationsboard dargestellt oder über die RS-232 Schnittstelle an ein anderes System übertragen werden. Bei der Ausgabe auf das Display des Evaluationsboards ist zu beachten, dass das Beschreiben des Displays sehr lange dauert.

Mit Hilfe der Dip-Switches können die auszugebenden Daten konfiguriert werden (siehe Tabelle D.2).

2.2.4 Dateninterface

Die Daten, die die Software dieses Projekts sammelt, sollen anderen Projekten als Grundlage dienen und zur Verfügung gestellt werden. Aus diesem Grund werden sowohl die zu jedem Zeitpunkt t (t = letztes Sample) diskret vorhandenen Messdaten der aktuellen Messfahrt und die daraus errechneten Bewegungsdaten zur Verfügung gestellt (siehe Kapitel 5.3.3).

Kapitel 3

Entwicklungsumgebungen zur Erstellung des Embedded Systems

Im Rahmen dieses Projektes soll ein echtzeitfähiges, mikroprozessorgesteuertes Embedded-System erstellt werden. Für die Erstellung dieses Systems werden verschiedene Entwicklungsumgebungen benötigt.

Eine Entwicklungsumgebung ist ein System, welches dem Entwickler bei der Erstellung von Hardware oder Software hilft. Bei der Entwicklung von Embedded-System kommt ein zusätzlicher PC zum Einsatz, mit dessen Hilfe zunächst das Embedded-System synthetisiert und anschließend die darauf ausgeführte Software entwickelt wird. Das Embedded-System wurde auf einem externen PC mit einem Windows 7 64-Bit Betriebssystem entwickelt. Ein RAID5 auf einem externen System und eine automatisierte Datensicherung vor jedem Ausschalten des Entwicklungsrechners sorgt für ausreichende Datensicherheit.

3.1 Evaluation Board - Virtex-5 ML507

Zur Durchführung des Projekts wird ein Xilinx Virtex-5 ML507 Evaluationsboard der HAW Hamburg genutzt. Der Vorteil dieser Plattform besteht darin, dass sowohl SoC (System-on-Chip) Co-Designs, als auch auf einem integrierbaren MicroBlaze Softcore Mikroprozessor ausgeführter C- und Assembler-Code erstellt werden kann.

Der MicroBlaze ist ein 32-Bit Big-Endian RISC⁶-Prozessor, der in diesem Projekt mit 125 MHz getaktet wird.

⁶RISC: Reduced Instruction Set Computer - Designphilosophie für Computerprozessoren, die einfache Maschinenbefehle bevorzugt

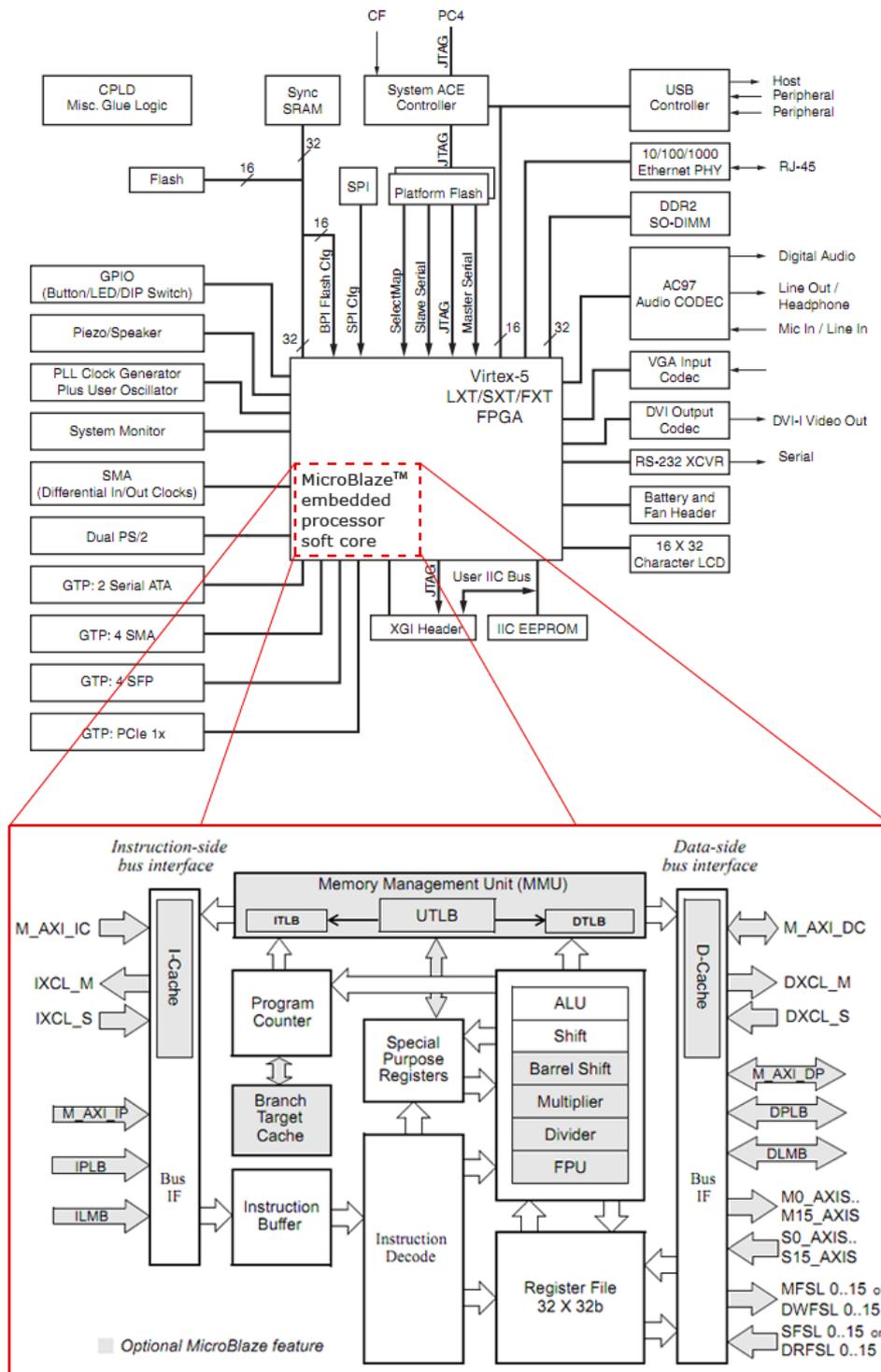


Abbildung 3.1: Oben: Blockdiagramm des ML507 Evaluationsboards [15]
 Unten: Blockdiagramm eines MicroBlaze Softcore Prozessors [10]

Das Evaluationsboard stellt diverse Schnittstellen zum Anschluss zusätzlicher Hardware zur Verfügung. Für dieses Projekt werden allerdings nur vier Pins für das I²C Interface und zwei Pins für den optionalen Anschluss eines Oszilloskops zur Messung des Zeitverhaltens benötigt:

- V_{CC} (Versorgungsspannung - 3,3 Volt)
- GND (Masse)
- S_{CL} (Signal - Clock)
- S_{DA} (Signal - Daten)
- Oszilloskop Pin 1
- Oszilloskop Pin 2

3.2 Xilinx Platform Studio

Um das auf dem Virtex-5 ML507 befindliche FPGA⁷ zu programmieren wird das von Xilinx zur Verfügung gestellte Platform Studio und ein USB-JTAG Adapter verwendet.

Neben dem MicroBlaze Mikroprozessor und den grundlegenden Hardwaremodulen (DDR-2 RAM, Busleitungen etc.) werden dem Systemdesign zusätzliche Komponenten hinzugefügt. Hierzu zählen die RS-232 Schnittstelle für die Kommunikation, General Purpose IO für Dip-Switches, Push-Buttons, LEDs, das Display des Virtex-5 und die Pins zum Anschluss des Oszilloskops, Timer, Interruptcontroller, sowie das I²C EEPROM. Das Xilinx Platform Studio weist durch entsprechende Anweisungen jeder eingebundenen Komponente Adressen aus dem verfügbaren Speicherbereich zu, sodass dieser als Interface für das Gerät fungiert. Die Verwendung von Memory-Mapped-IO hat den Vorteil, dass die Treiberentwicklung einfacher und einheitlicher ist, als bei IO-Mapped-IO.

Nach der Synthese steht eine *.Bit*-Datei zur Programmierung des FPGAs zur Verfügung.

3.3 Xilinx Software Development Kit

Für die Programmierung und Ausführung von nativem C Code bietet Xilinx das Software Development Kit an, eine auf Eclipse basierende Entwicklungsumgebung.

⁷FPGA: Field Programmable Gate Array - Integrierter mit logischen Schaltungen programmierbarer Schaltkreis

Mit dem Export eines Projektes aus dem Xilinx Platform Studio in das Xilinx Software Development Kit, werden Treiber für synthetisierte Module und Bibliotheken zur Verfügung gestellt. Wie in den meisten Fällen, werden auch für die in diesem Projekt genutzte I²C Schnittstelle Funktionen zum Lesen aus und Schreiben in das I²C EEPROM bereitgestellt.

Debugging- oder Systemausgaben der Software können entweder auf dem Display des ML507 oder durch Verwendung der RS-232 Schnittstelle auf dem angeschlossenen Entwicklungssystem in ein Terminalprogramm gemacht werden. Bei der Ausgabe ist zu beachten, dass die korrekte Übertragungsrate (115200 Baud) der RS-232 Schnittstelle ausgewählt ist.

Kapitel 4

Magnetfeld- und Inertialsensorik

4.1 Einführung

Der 9 Degrees of Freedom Sensor Stick von Sparkfun stellt drei verschiedene Sensoren zur Verfügung.

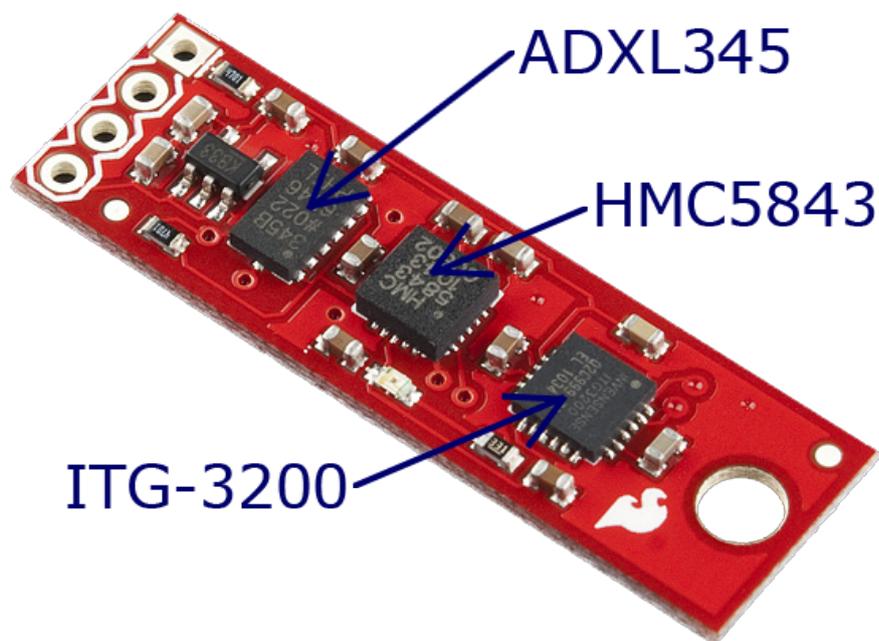


Abbildung 4.1: 9 Degrees of Freedom Sensor Stick von Sparkfun (ADXL345 Accelerometer, HMC5843 Digitalkompass und ITG-3200 Drehratensensor) [32]

Bei dem verwendeten Gyroskop handelt es sich um den ITG-3200 von InvenSense, das

digitale Accelerometer ADXL345 von Analog Devices und der Digitalkompass HMC5843 von Honeywell. Als Schnittstelle zu dem Virtex-5 ML507 Evaluationsboard wird der 2001 von Philipps entwickelte I²C Bus verwendet. Alle drei Sensoren unterstützen den I²C Fast Mode, der laut Spezifikation aus dem Jahr 2003 einen maximalen Takt von 400kHz unterstützt. Der langsamste auf dem Sensor Stick verbaute Sensor ist der Digitalkompass, der eine maximale Aktualisierungsrate von 50Hz unterstützt.

In diesem Kapitel werden die Sensoren selbst, die Theorie, die zur korrekten Nutzung dieser notwendig ist, sowie Informationen zu Messabweichungen und deren Kompensation dargestellt.

4.2 Digitalkompass - HMC5843

Der HMC5843 von Honeywell gehört zu der Klasse der anisotropen magnetoresistiven (AMR) 3-Achsen-Digitalkompass.

Durch drei orthogonal angeordnete Sensoren wird das zum Zeitpunkt der Messung umliegende Magnetfeld gemessen. Dieses Magnetfeld kann durch verschiedene externe Einflüsse gestört werden, was zu Messabweichungen führen kann. Die Messwerte werden von Analog/Digital Konvertern umgewandelt, in zwei Registern je Achse bis zum nächsten Sample gespeichert und können durch Anschluss an einen I²C Bus ausgelesen werden.

Die maximale Abtastfrequenz des Sensors liegt bei 50Hz und einer Genauigkeit von 7 Milli-Gauss und einer maximalen Feldstärke von +/-4 Gauss. Als Vergleichsgröße könnte das Erdmagnetfeld in Mitteleuropa mit einer Feldstärke von 0,48 Gauss genannt werden. Zu dieser Angabe ist hinzuzufügen, dass sich das Erdmagnetfeld in ständigem Wandel befindet und sehr empfindlich gegenüber Störungen von äußeren Einflüssen, wie elektrischen Geräten oder ferromagnetischen Objekten, ist.

4.2.1 HMC5843 - Funktionsweise

Die Messung des anliegenden Magnetfeldes erfolgt mit Hilfe von drei orthogonal angeordneten Sensoren, welche durch Anwendung des 1857 von William Thomson entdeckten anisotropen magnetoresistiven Effekts [49] die Flussrichtung des Magnetfeldes bestimmen können.

Bei diesem Verfahren wird durch Bestimmung des Widerstands eines Permalloy Elements die Ausrichtung des Sensors in Bezug zum umliegende Magnetfeld gesetzt. Bei Permalloy handelt es sich um eine Legierung aus 81% Nickel und 19% Eisen. Die spezielle

Eigenschaft dieser ferromagnetischen⁸ Legierung besteht darin, dass diese aufgrund eigener Magnetisierung abhängig von dem umgebenden Magnetfeld einen bestimmten Widerstand hat. Ist das umliegende Magnetfeld stark genug, so wird die interne Orientierung des Magnetfeldes so gedreht, dass sich das Magnetfeld der Legierung und das umliegende Magnetfeld in gleicher Richtung angeordnet haben. Der Widerstand ist maximal, wenn der Stromdichtevektor des durch das Metall fließenden Stroms und der Magnetfeldvektor des Bauelements parallel zueinander sind und minimal, wenn diese senkrecht zueinander stehen. Da der Widerstand des Leiters ohne umliegendes und mit umliegendem orthogonalen Magnetfeld bekannt ist, kann die Ausrichtung des Leiters in Bezug zum umliegenden Magnetfeld bestimmt werden.

Durch orthogonale Anordnung mehrerer Permalloy Leiter kann eine Abbildung der gemessenen Daten in einem entsprechenden mehrdimensionalen kartesischen Koordinatensystem erstellt werden.

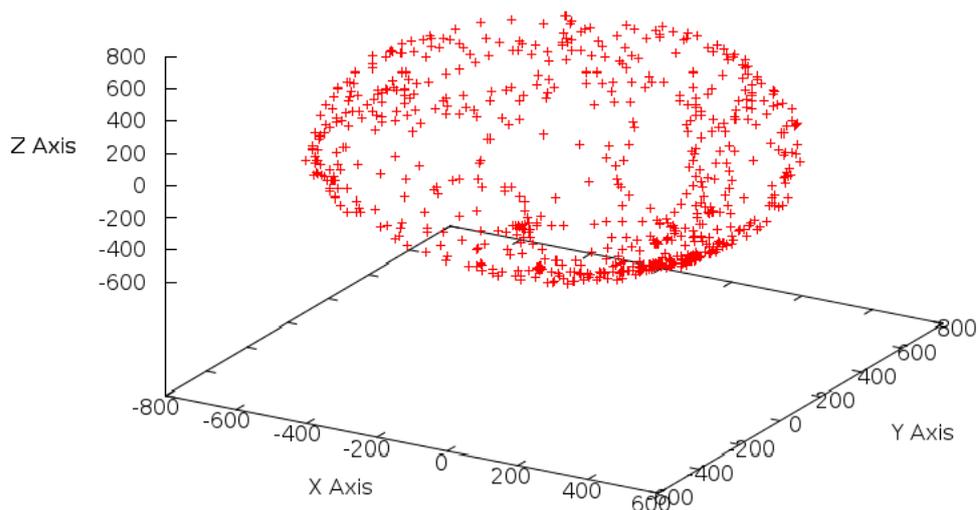


Abbildung 4.2: Graph mit den Messwerten des HMC5843 nach 3D Rotation - je kugelförmiger der Ellipsoid ist, desto besser ist die Kalibrierung des Digitalkompasses [33]

⁸ferromagnetisch: Ein ferromagnetischer Stoff (Eisen, Kobalt, Nickel...) wird von einem Magneten angezogen und lässt sich durch Aussetzung eines externen magnetischen Feldes magnetisieren.

4.2.2 HMC5843 - Datenerfassung, Messabweichungen und Kompensation

Nach der Messung des Widerstandes werden die Daten von Analog-Digital-Konvertern in digitale Werte umgewandelt und in jeweils zwei Registern abgelegt. Werden diese nun vom I²C Master gelesen, müssen sie zuerst zusammengefügt werden. Dabei muss beachtet werden, dass die Daten in den Registern im 2er-Komplement Format vorliegen.

```
1 sensor_data[0] = (((int16_t)compass_buffer[0] << 8) | compass_buffer[1]);  
2 sensor_data[1] = (((int16_t)compass_buffer[2] << 8) | compass_buffer[3]);  
3 sensor_data[2] = (((int16_t)compass_buffer[4] << 8) | compass_buffer[5]);
```

Um eine Abschätzung über die Qualität der Daten machen zu können, werden diese in Hinblick auf die Wertebereiche jeder Achse untersucht. Durch Betrachtung dieser können verschiedene Störungen ermittelt werden. Werden Messabweichungen entdeckt, können diese teilweise oder komplett mit Hilfe einer Kalibrierungsroutine getilgt werden.

Die folgende Grafik stellt den Einfluss dreier verschiedener Störungen des Erdmagnetfeldes dar. Bei der ersten Störung handelt es sich um eine Verschiebung des Ursprungs des Magnetfeldes, welche zumeist *Hard Iron Distortion* oder *Hard Iron offset* genannt wird. Diese Störung zeichnet sich dadurch aus, dass sie relativ zum Sensor gesehen, immer die gleich Position hat und orientierungsunabhängig ist. Die zweite Störung resultiert aus einer falsch skalierten Achse und wird *gain mismatch* genannt. Die Korrektur dieser Störung erfolgt durch Multiplikation mit einem Skalierungsfaktor, der wie nachfolgend beschrieben errechnet wird. Bei der dritten Störung handelt es sich um eine aus schadhaftem Magnetfeld resultierende Störung, die häufig *Soft iron distortion* genannt wird und durch ferromagnetische Materialien wie Eisen und Nickel erzeugt wird (siehe Abbildung 4.6). Ob eine der genannten Störungen vorliegt, kann nur schwer anhand der Messwerte erkannt werden. Um die Störungen zu visualisieren werden die Messwerte in Koordinatensysteme eingetragen, wodurch zwei- oder dreidimensionale Graphen entstehen (siehe Abbildungen 4.2, 4.4 und 4.5), anhand deren Form und Lage die zuvor genannten Störungen ermittelt werden können. Algorithmen zur Kalibrierung werden nachfolgend beschrieben.

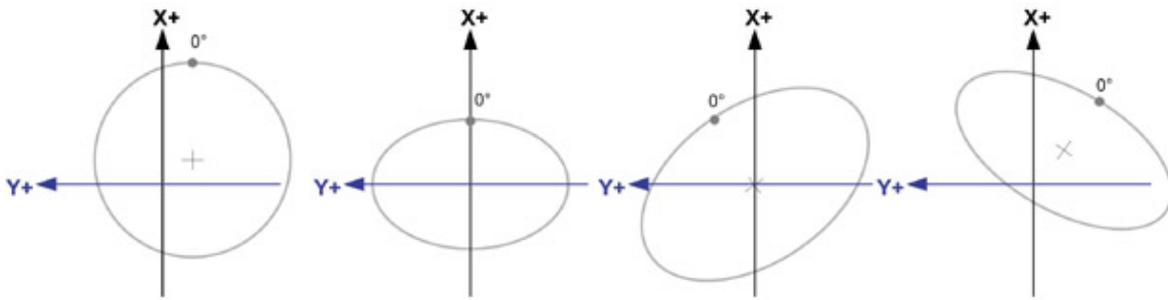


Abbildung 4.3: X-Y-Graphen unter Einfluss verschiedener magnetischer Störungen [29]

Um diese Abschätzung machen zu können, werden zunächst die Daten einer langsamen stationären 360 Grad Rotation gesammelt. Durch eine exakte stationäre Rotation, also eine Rotation um die Z-Achse (um das Zentrum des Sensors) wird sichergestellt, dass sich die umliegenden Magnetfelder während der gesamten Messfahrt nicht ändern.

Zur Visualisierung können die aggregierten Daten in einem kartesischen Koordinatensystem abgebildet werden:

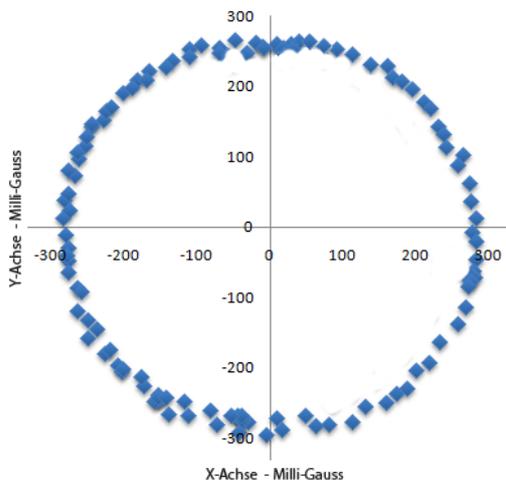


Abbildung 4.4: Aggregierte Daten in der X-Y-Ebene einer stationären Rotation

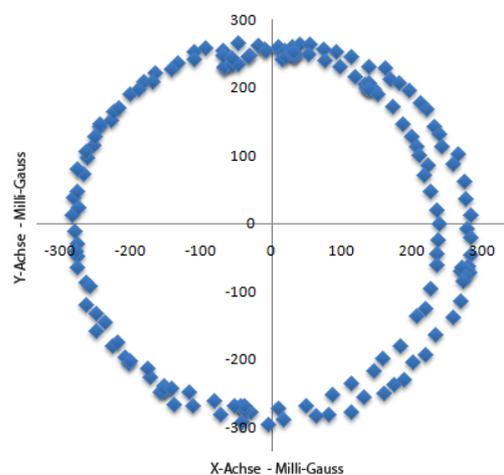


Abbildung 4.5: Aggregierte Daten in der X-Y-Ebene einer nicht stationären Rotation

Die linke Grafik zeigt eine korrekt ausgeführte Rotation. Die rechte zeigt eine nicht korrekt ausgeführte Rotation, wobei der Sensor während der Rotation in die Nähe eines ferromagnetischen Gegenstandes verschoben wurde.

Die aggregierten Daten werden auf Maxima und Minima untersucht. Sind diese für alle Achsen ermittelt, werden die Offsets für die drei Achsen berechnet:

$$\begin{aligned} X_{Offset} &= \frac{X_{Max} - X_{Min}}{2} \\ Y_{Offset} &= \frac{Y_{Max} - Y_{Min}}{2} \\ Z_{Offset} &= \frac{Z_{Max} - Z_{Min}}{2} \end{aligned}$$

Durch Berechnung der Offsets ist es möglich die Messwerte auf den Koordinatenursprung zu verschieben. Mit Hilfe der gemessenen Maxima und Minima kann neben der Verschiebung auch der Skalierungsfaktor bestimmt werden, mit dem die Ellipse entzerrt werden kann. Dazu werden die gemessenen Werte mit den zuvor errechneten Offsets korrigiert und auf das betragsmäßige Maximum aller Werte hin untersucht.

$$\begin{aligned} X_{Scale} &= \frac{Max}{X_{Max}} \\ Y_{Scale} &= \frac{Max}{Y_{Max}} \\ Z_{Scale} &= \frac{Max}{Z_{Max}} \end{aligned}$$

Unter Verwendung der berechneten Skalierungsfaktoren und der Offsets können die Messwerte korrigiert werden um einen möglichst zentrierten und unverzerrten Ellipsoid zu erhalten:

$$\begin{aligned} X_{Corrected} &= X * X_{Scale} - X_{Offset} \\ Y_{Corrected} &= Y * Y_{Scale} - Y_{Offset} \\ Z_{Corrected} &= Z * Z_{Scale} - Z_{Offset} \end{aligned}$$

Es ist zu beachten, dass die vorgenommene Kalibrierung zur Beseitigung der aus der zuvor beschriebenen Soft-Iron-Distortion resultierenden Verzerrung stark fehleranfällig ist und lediglich für exakt den Standort funktioniert, an dem die Kalibrierung vorgenommen worden ist, was darauf zurückzuführen ist, dass das Erdmagnetfeld durch verschiedene Störquellen negativ beeinflusst wird. Störquellen können sowohl Geräte sein, welche elektromagnetische Felder emittieren, als auch Gegenstände, die ferromagnetische Elemente enthalten.

4.3 Accelerometer - ADXL345

Das ADXL345 Accelerometer gehört zur Klasse der MEMS⁹-Sensoren. Bei diesen Sensoren handelt es sich um miniaturisierte elektro-mechanische Systeme, die nur wenige Mikrometer groß sind.

⁹MEMS: Micro-Electro-Mechanical System - Ein miniaturisiertes Gerät, dessen Größe im Mikrometerbereich liegt

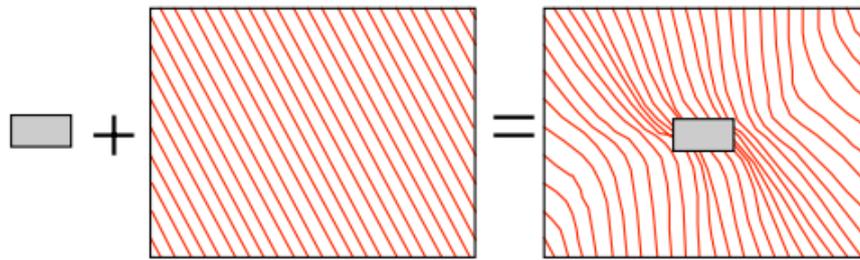


Abbildung 4.6: Einfluss eines ferromagnetischen Objekts auf ein Magnetfeld [1]

Durch Kapazitätsmessung von variablen Kondensatoren (siehe Abbildung 4.7) wird die auf den Sensor wirkende Beschleunigung gemessen. Die gemessenen Beschleunigungen werden in g ($1g = 9,81 \frac{m}{s^2}$) gemessen, von Analog/Digital Konvertern digitalisiert und in zwei Registern pro Achse abgespeichert. Als Interface dient dem Sensor der I²C Bus. Die maximale Abtastfrequenz des Sensors liegt bei 3200Hz und einer Genauigkeit von 4,3 Milli- g und einem maximalen Messbereich von $\pm 16 g$.

4.3.1 ADXL345 - Funktionsweise

Zur Herstellung von Mikro-Elektrisch-Mechanischen-Systemen werden lithographische Verfahren (Fotolithographie, Elektronenstrahlstrukturierung. . .) und herkömmliche miniaturisierte Herstellungsverfahren (Fräse, Laser. . .) eingesetzt. Diverse unterstützende Verfahren (Evaporation, Doping, CVD. . .) kommen zum Einsatz, um die nur wenige Mikrometer großen Systeme herzustellen. [22]

Um die auf Beschleunigungssensoren wirkenden Beschleunigungskräfte zu messen, werden drei gestapelte Platten über Spiralfedern miteinander verbunden. Die beiden äußeren Platten werden fixiert, die mittlere wird von den Spiralfedern gehalten und bildet mit jeder der äußeren Platten einen variablen Kondensator.

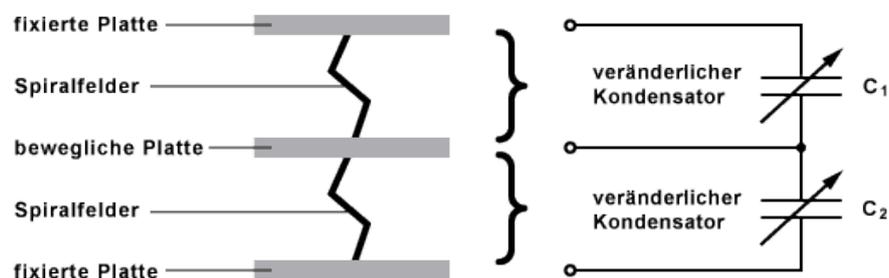


Abbildung 4.7: Aufbau eines MEMS-Beschleunigungssensors mit zwei Kondensatoren [27]

Die Kapazität eines Kondensators ist abhängig von der Oberfläche der Elektroden, der Dielektrizitätszahl des Dielektrikums und dem Kehrwert des Abstandes der Elektroden, wobei in diesem Fall lediglich der Abstand der beiden Elektroden beschleunigungsabhängig und damit von Relevanz ist.

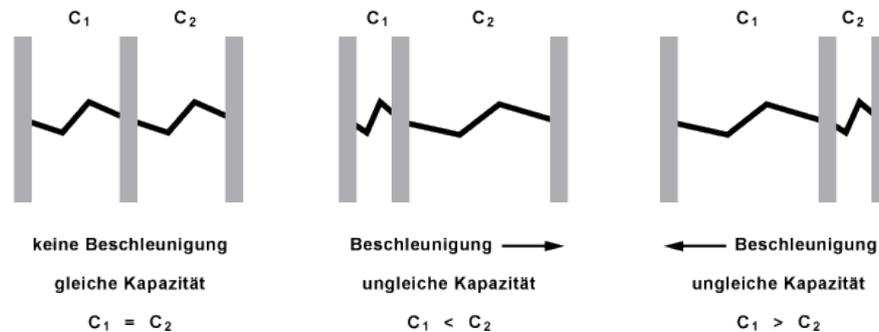


Abbildung 4.8: Funktionsweise eines MEMS-Beschleunigungssensors [27]

Liegt keine auf das System wirkende Geschwindigkeitsänderung vor, so befinden sich die beiden Kondensatoren im ausgeglichenen Zustand und der Unterschied der beiden Kapazitäten ist null. Bei Beschleunigung des Systems wird die mittlere Platte in die der Beschleunigung entgegengesetzten Richtung verschoben.

Durch Verschiebung der mittleren Platte ändern sich die Kapazitäten der beiden in Reihe geschalteten Kondensatoren. Die gemessene Kapazitätsänderung ist proportional zur Beschleunigung. [21]

4.3.2 ADXL345 - Datenerfassung, Messabweichungen und Kompensation

Nach dem Schluss von der Kapazitätsänderung auf die Beschleunigung werden die Werte im 2er-Komplement Format in zwei Register pro Achse geschrieben und können vom I²C Master gelesen werden. In Kapitel 4.2.2 wird beschrieben, wie die beiden Register korrekt zusammengefügt werden, wobei bei dem Beschleunigungssensor zu beachten ist, dass die beiden Register vertauscht sind und dieses entsprechend in der Routine angepasst werden muss. Nach dem korrekten Zusammenfügen der Daten können die Daten von g (Erdbeschleunigung) in $\frac{m}{s^2}$ umgewandelt werden.

Um möglichst geringe Messabweichungen zu erhalten wird der Beschleunigungssensor kalibriert. Dazu wird der Durchschnitt einiger Messwerte im stationären Zustand (X-Y-Ebene und in Ruhe befindlich) gebildet. Die Abweichungen vom Nullwert ($0g$) werden ermittelt und zur Korrektur der Messwerte vor weiterer Verarbeitung verwendet.

```

1     for(i = 0; i < sampleCount; i++) {
2         iic_get_data_accelerometer();
3         Accel_X_Offset += sensor_data[6];
4         Accel_Y_Offset += sensor_data[7];
5         Accel_Z_Offset += sensor_data[8];
6     }
7     Accel_X_Offset = -Accel_X_Offset / sampleCount + 0.5;
8     Accel_Y_Offset = -Accel_Y_Offset / sampleCount + 0.5;
9     Accel_Z_Offset = (-Accel_Z_Offset / sampleCount + 0.5) + 250;

```

In dem Datenblatt des ADXL345 [31] wird als *0g Bias Level* für die Z-Achse der Wert 128 (+/- 4g) angegeben. Dieser Wert stellt die Erdbeschleunigung mit 1g ($9,81 \frac{m}{s^2}$) dar. Nach der Kalibrierung werden die Messwerte in $\frac{m}{s^2}$ umgerechnet:

$$\begin{aligned}
 Accel_{X-Tmp} &= \frac{Accel_{X-Tmp}}{9,81} \\
 Accel_{Y-Tmp} &= \frac{Accel_{Y-Tmp}}{9,81} \\
 Accel_{Z-Tmp} &= \frac{Accel_{Z-Tmp}}{9,81}
 \end{aligned}$$

Die vom Beschleunigungssensor gemessenen Geschwindigkeitsänderungen, werden zu den drei Beschleunigungsvariablen hinzu addiert, wobei die Beschleunigungsvariablen die zur Zeit des Samples t aktuellen Beschleunigungen enthalten:

$$\begin{aligned}
 Accel_X &= Accel_X + Accel_{X-Tmp} \\
 Accel_Y &= Accel_Y + Accel_{Y-Tmp} \\
 Accel_Z &= Accel_Z + Accel_{Z-Tmp}
 \end{aligned}$$

Die aus der Beschleunigung resultierende Geschwindigkeit wird durch Anwendung des Euler-Verfahrens auf die drei Beschleunigungsvariablen errechnet:

$$\begin{aligned}
 Velocity_X &= Velocity_X + Accel_X * f_s \\
 Velocity_Y &= Velocity_Y + Accel_Y * f_s \\
 Velocity_Z &= Velocity_Z + Accel_Z * f_s
 \end{aligned}$$

Mit Hilfe des Euler-Verfahrens wird aus der aktuellen Geschwindigkeit die zurückgelegte Strecke errechnet:

$$\begin{aligned}
 Distance_X &= Distance_X + Velocity_X * f_s \\
 Distance_Y &= Distance_Y + Velocity_Y * f_s \\
 Distance_Z &= Distance_Z + Velocity_Z * f_s
 \end{aligned}$$

Das hier verwendete Euler-Verfahren ist das simpelste Verfahren ein Integral näherungsweise zu bestimmen. Hier könnten bessere numerische Verfahren, wie das Runge-Kutta Verfahren eingesetzt werden.

Der Sensor misst sowohl dynamische Beschleunigungen, welche aus Bewegungen oder Erschütterungen resultieren, als auch statische Beschleunigung, wie die Erdanziehungskraft. Da diese beiden Beschleunigungen jedoch prinzipiell nicht unterscheidbar sind, muss dieses bei der Auswertung der Daten beachtet werden.

4.4 Gyroskop - ITG-3200

Das Gyroskop, auch Drehratensensor genannt, misst Rotationen pro Zeiteinheit. Der auf dem 9 Degrees-of-Freedom Sensor Stick verbaute ITG-3200 ist ein digitales MEMS-Gyroskop, welches zur Messung der Rotation die auf den Sensor wirkende Coriolis-Kraft mit Hilfe von stimmungsbelförmig angeordneten Testmassen bestimmt. Nach der Digitalisierung der Messwerte mit Analog/Digital-Konvertern, werden die Daten in zwei Registern pro Achse abgelegt und können vom I²C Master gelesen werden. Der Sensor unterstützt Drehraten von bis zu $\pm 2000 \frac{\text{Grad}}{\text{s}}$, bei einer Auflösung von $0,07 \frac{\text{Grad}}{\text{s}}$.

Zur Einordnung von Rotationsgeschwindigkeiten einige typische Richtwerte bestimmter Anwendungen:

Anwendung	Drehrate
Fahrzeugnavigation	10 $\frac{\text{Grad}}{\text{s}}$
Fahrzeug Kontrollsysteme	30 $\frac{\text{Grad}}{\text{s}}$
Bildstabilisation in Kameras	100 $\frac{\text{Grad}}{\text{s}}$
Game Controller	300 $\frac{\text{Grad}}{\text{s}}$
Erfassung der Schwungbewegung eines Profigolfers	3000 $\frac{\text{Grad}}{\text{s}}$

Tabelle 4.1: Übersicht typischer Drehraten [39]

Ausgehend von den Werten dieser Tabelle, ist der ITG-3200 von InvenSense für das Auswerten von menschlichen Bewegungen bestens geeignet.

4.4.1 ITG3200 - Funktionsweise

Im folgenden Kapitel wird die Funktionsweise des ITG-3200 Drehratensensors erläutert, dazu wird näher auf die gemessene Kraft, das Herstellungsverfahren und das Messprinzip des Sensors eingegangen.

Der ITG-3200 Drehratensensor bedient sich zur Messung der Rotation dem Coriolis-Kraft-Effekt. Dieser Effekt beschreibt das Phänomen, dass ein Gegenstand der sich auf einer drehenden Scheibe befindet und von außen betrachtet eine lineare Bewegung, aber in Bezug auf die sich drehende Scheibe bei konstanter Geschwindigkeit eine gekrümmte Bahn vollführt, was darauf zurückzuführen ist, dass die sich mit konstanter Geschwindigkeit drehende Scheibe in den äußeren Radien eine höhere Geschwindigkeit besitzt als in den kleineren inneren Radien.

Bei der Coriolis-Kraft handelt es sich um eine Schein- oder Trägheitskraft, die auf jeden in Bewegung befindlichen massehaltigen Körper in einem rotierenden Bezugssystem wirkt und senkrecht sowohl zur Rotationsachse des Bezugssystems, als auch zur Bewegungsrichtung des Gegenstandes ist.

Der MEMS-Drehratensensor ITG-3200 von InvenSense wird unter Verwendung des vom Firmengründer entwickelten *Nasiri*-Herstellungsverfahren hergestellt. [38] Durch dieses Verfahren können Komponenten hergestellt werden, bei denen besonders Kosteneffizienz, Größe und Leistung eine Rolle spielen. Erreicht wird dieses durch Verwendung von einfach-kristallinem-Silizium und dem *AlGe Eutectic Bonding*¹⁰, einem Verfahren, bei dem jeder der mehreren Tausend auf einem Wafer¹¹ befindlichen CMOS Bausteine zeitgleich mit einem mikro-elektro-mechanischen-System verbunden wird. Zwischen den beiden verbundenen Bauteilen befindet sich ein hermetisch versiegeltes Vakuum, das zum einen präzises Messen ermöglicht und zum anderen die Struktur des mikromechanischen Systems schützt. [35]

Bei dem ITG-3200 handelt es sich um ein Resonanzgyroskop, dessen Testmassen ähnlich zwei Stimmgabelzinken angeordnet sind.

¹⁰AlGe Eutectic Bonding - Verbindung von Al auf dem CMOS Wafer und Ge auf dem MEMS Wafer

¹¹Wafer - Silizium Scheibe, die als Substrat (Grundplatte) für elektronische Bauelemente dient.

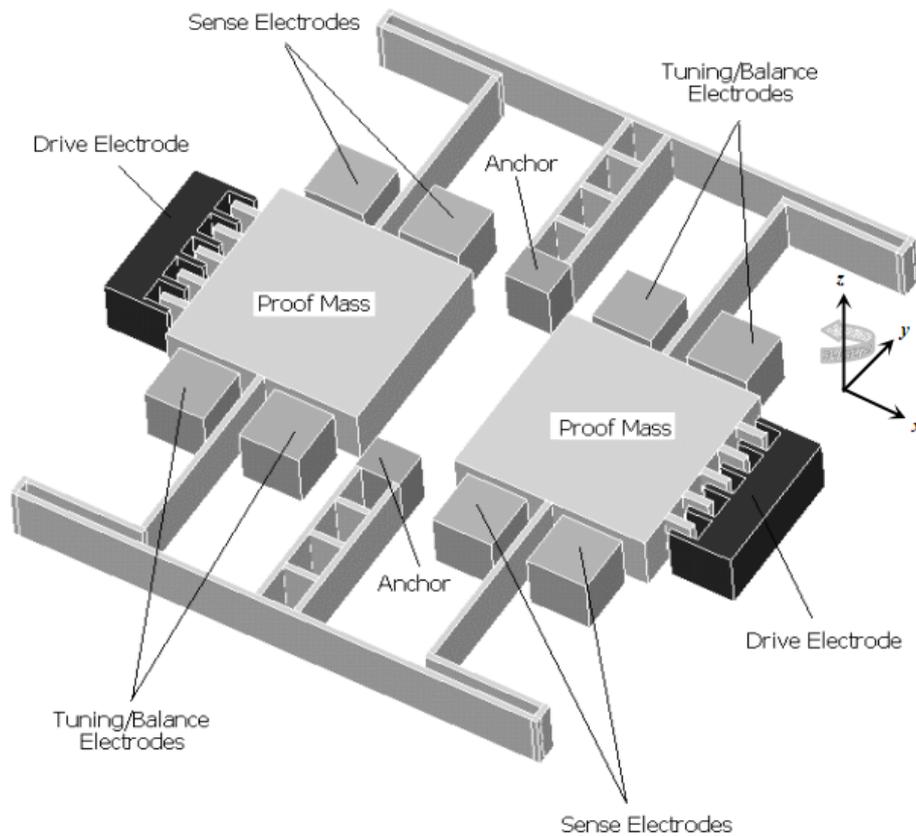


Abbildung 4.9: Schematischer Aufbau eines Resonanzgyroskops (Stimmgabelprinzip) [38]

Zwei Haltepunkte (Anchors) befestigen den äußeren Rahmen mit dem Substrat (Grundplatte). Dieser Rahmen wird auch *Drive Frame* genannt. An dem Drive Frame sind zwei Testmassen (Proof Masses) befestigt, die mit Hilfe der von zwei Elektroden (Drive Electrodes) erzeugten elektrostatischen Felder in Schwingungen versetzt werden. Dabei ist die Schwingung der beiden Testmassen genau gegensätzlich gerichtet. Erfährt das System eine Rotation, wirkt die Corioliskraft auf die beiden Testmassen und verschiebt diese senkrecht zur Schwingungsrichtung. Diese Verschiebung wird kapazitiv mit jeweils zwei Elektroden (Sense electrodes) je Testmasse ermittelt.

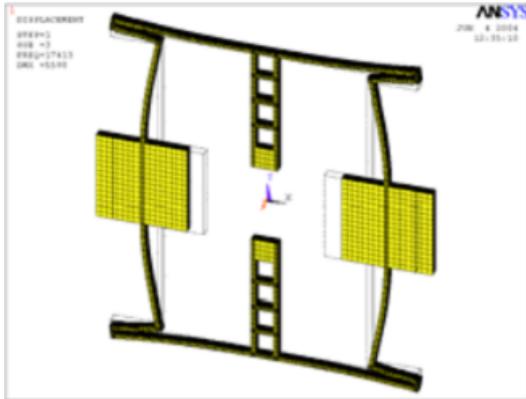


Abbildung 4.10: In Schwingung versetzte Testmassen [35]

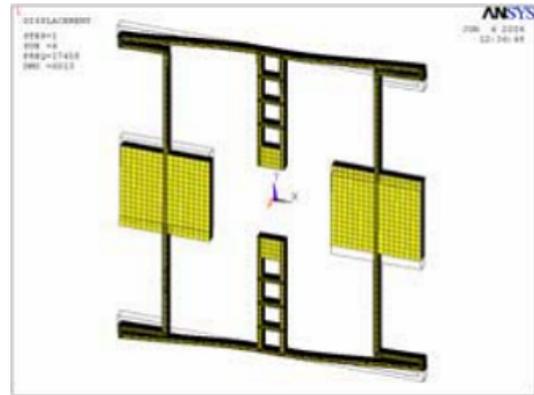


Abbildung 4.11: Aus Rotation und Schwingung resultierender Coriolis-Kraft-Effekt [35]

Der für diesen Drehratensensor verwendete Aufbau hat den Vorteil, dass auf die Testmassen wirkende lineare Beschleunigungen keinen Einfluss auf das Ergebnis der kapazitiven Auswertung haben, was damit zu begründen ist, dass lineare Beschleunigungen auf beide Testmassen exakt gleich wirken (Verschiebung in gleiche Richtung), sich aber Kräfte, die aus dem Coriolis-Kraft-Effekt resultieren, unterschiedlich auf die beiden Testmassen auswirken. Deswegen werden in der Regel symmetrisch aufgebaute MEMS zur Messung von radialen Bewegungen verwendet.

Sofern empfindlichere Sensorik zur Messung von sehr langsamen Rotationen benötigt wird, werden in der Regel optische Gyroskope (Ring-Laser-Gyroskop) anstatt der hier verwendeten Resonanz Gyroskope verwendet, die sich dem Sagnac-Effekt [48] bedienen und Genauigkeiten bis zu $0,001 \frac{\text{Grad}}{h}$ erreichen können.

4.4.2 ITG3200 - Datenerfassung, Messabweichungen und Kompensation

Nach der Digitalisierung der Messwerte von Analog/Digital Konvertern werden die Daten in zwei Registern abgelegt und können entsprechend dem zuvor beschriebenen Verfahren zusammengefügt werden (siehe Kapitel 4.2.2). Der Sensor muss vor Benutzung kalibriert werden, dazu wird die in Kapitel 4.3.2 beschriebene Kalibrieroutine für das hier verwendete Gyroskop angepasst. Es ist darauf zu achten, dass in diesem Fall alle drei Achsen gegen den Nullwert kalibriert werden.

Bei der Datenverarbeitung muss darauf geachtet werden, dass eine korrekte Konvertierung

der Daten von $\frac{\text{Grad}}{s}$ wie der Sensor sie liefert, in die der Samplerate entsprechende umgewandelt wird:

$$\text{Rotation}_X = \text{Rotation}_X * f_S$$

$$\text{Rotation}_Y = \text{Rotation}_Y * f_S$$

$$\text{Rotation}_Z = \text{Rotation}_Z * f_S$$

Durch Addition aller erfassten Rotationen, kann die Lageänderung in Bezug zur Ausgangslage bestimmt werden.

Eine Rotation ändert die Lage des Sensorkoordinatensystems in Bezug zum Koordinatensystem der Umgebung, was zur Folge hat, dass sich die Richtung der Beschleunigung bei einer Rotation in der X-Y-Ebene ändert. Dieses muss später bei der Ermittlung der Position beachtet werden.

4.5 Datenauswertung

Nachfolgend wird beschrieben, wie die Neigung und die Ausrichtung des Sensors bestimmt wird. Hier wird davon ausgegangen, dass die Sensoren optimal kalibriert und die gelieferten Daten korrekt sind.

4.5.1 Neigung

Um die Neigung bestimmen zu können, wird auf die Werte des Accelerometers zurückgegriffen. Da dieses neben dynamischen Beschleunigungen auch statische Beschleunigungen wie die Erdanziehungskraft misst, können diese Daten zur Bestimmung der Lage des Sensors im Raum genutzt werden. Dieser Ansatz ist zwar nur korrekt, solange sich das System in Ruhe befindet (es also nicht bewegt wird), wovon zu diesem Zeitpunkt ausgegangen wird.

Ein Objekt kann im dreidimensionalen Raum in drei verschiedene Richtungen geneigt sein. Man spricht von Neigung, da sich bei einer Rotation um eine Achse, die Ausrichtung der anderen beiden Achsen ändert, sich das Objekt also neigt:

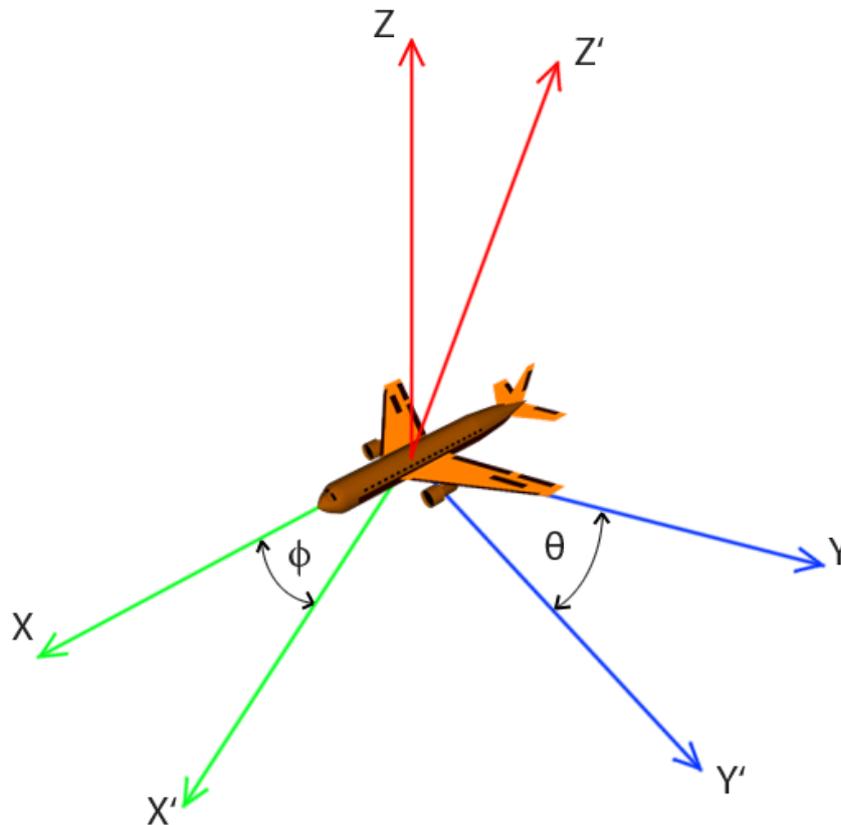


Abbildung 4.12: Roll- und Nickwinkel eines Objektes im dreidimensionalen Raum

Durch trigonometrische Analyse des Diagramms ergeben sich folgende Formeln zur Bestimmung der eingezeichneten Winkel, wobei θ der Winkel zwischen der X-Achse und der horizontalen Ebene und ϕ den Winkel zwischen der Y-Achse und der horizontalen Ebene darstellt:

$$\phi = \arctan\left(\frac{Y}{\sqrt{X^2+Z^2}}\right)$$

$$\theta = \arctan\left(\frac{X}{\sqrt{Y^2+Z^2}}\right)$$

4.5.2 Ausrichtung

Zur Bestimmung der Ausrichtung des Sensors ψ werden die vom Magnetometer gemessenen Werte der X- und Y-Achse verwendet. Es reicht die Betrachtung dieser beiden Achsen, da das Erdmagnetfeld im Idealfall lediglich diese beiden Achsen beeinflusst:

$$\psi = \text{atan2}(Y_h, X_h)$$

Diese Gleichung nutzt die Werte der X- (X_h) und Y-Achse (Y_h) zur Bestimmung der Ausrichtung in Bezug auf das Erdmagnetfeld und funktioniert allerdings nur dann, wenn Roll- und Nickwinkel Null sind, sich der Sensor also auf einer Horizontalebene befindet. Ist dieses nicht der Fall, so kann mit Hilfe der im vorherigen Kapitel berechneten Winkel für Roll (ϕ) und Nick (θ) eine Neigungskorrektur für den digitalen Kompass durchgeführt werden:

$$X_h = X * \cos(\theta) + Y * \sin(\phi) * \sin(\theta) - Z * \cos(\phi) * \sin(\theta)$$

$$Y_h = Y * \cos(\phi) + Z * \sin(\phi)$$

$$\psi = \text{atan2}(Y_h, X_h)$$

Es ist anzumerken, dass diese Neigungskorrektur von *Michael J. Caruso* [1] auf den vom Beschleunigungssensor gelieferten Messwerten und den daraus resultierenden Roll- und Nickwinkeln beruht und somit den gleichen Einschränkungen unterliegt, wie die Berechnung der Neigungswinkel.

Kapitel 5

Embedded System zur Indoor-Positionsermittlung

Im folgenden Kapitel wird die Erstellung der Hard- und Software des Embedded Systems beschrieben.

5.1 Einführung

Die Entwicklung des Embedded Systems zur Sammlung und Auswertung der von den Sensoren gelieferten Daten gliedert sich mehrere Abschnitte. Zu Beginn wird unter Einsatz des Xilinx Platform Studios ein MikroBlaze Mikrocontroller konfiguriert, der als zentrale Schnittstelle für alle weiteren Hardwaremodule fungiert und die Steuerung des Systems, sowie die Ausführung der entwickelten C-Routinen übernimmt. Nachdem dem System alle Komponenten hinzugefügt worden sind, müssen diese mit dem Mikrocontroller und den externen Interfaces verbunden werden. Ist dieser Schritt abgeschlossen, wird der Hardwareentwurf synthetisiert und zur Programmierung des FPGAs verwendet.

Der nächste Schritt ist die Überprüfung der Hardware, bei der die Funktionalität der eingebundenen Hardwaremodule grundlegend getestet wird. Einige Hardwaremodule müssen softwareseitig initialisiert werden, um deren Funktionalität überprüfen zu können. Wenn die Funktionalität der Hardware überprüft und verifiziert worden ist, kann mit der Entwicklung der Software begonnen werden. Die verwendete Time-Triggered-Architektur stellt sicher, dass die Daten immer rechtzeitig von den Sensoren gelesen und ausgewertet werden. Bei dem Entwurf des System wurde darauf geachtet, keine *Delay*-Funktionen zu verwenden, da sich bereits zu Beginn der Softwareentwicklung gezeigt hat, dass diese nicht immer korrekt funktionieren. Es hat sich ergeben, dass die für das *Busy-Waiting* verwendeten Zählvariablen, trotz aktivierter Daten- und Instruktionscaches, nicht immer im

Cache liegen, was zur Folge hatte, dass die Ausführungszeit der *Delay*-Funktion um den Faktor 23 länger war als geplant.

5.2 Hardwareentwurf

Das Embedded System wird mit Hilfe des *Base-System-Builders* des Xilinx Platform Studios erzeugt. Dieses Tool ermöglicht es, mit nur wenigen Parametern einen MicroBlaze Mikrocontroller zu erzeugen, an den bereits einige der benötigten Hardwaremodule angeschlossen sind. Alle nicht im Assistenten abgefragten Parameter und nicht zur Auswahl stehende Hardwaremodule (IPs) können im Nachhinein angepasst und hinzugefügt werden. Die Anpassung der UCF¹², MSS¹³ und MHS¹⁴ Dateien ist direkt im Xilinx Platform Studio möglich, was bedeutet, dass Änderungen an diesen Dateien direkt übernommen werden. Nachfolgend ist ein Blockdiagramm dargestellt, welches alle externen Schnittstellen des Hardwareentwurfs für dieses Projekt darstellt:

¹²UCF: User Constraint File - In dieser Datei werden die Verbindungen zwischen logischen Netzen des FPGAs und physikalischen Pins des Boards definiert.

¹³MSS: Microprocessor Software Specification - In dieser Datei werden Informationen für die Erzeugung von Bibliotheken und Treibern hinterlegt.

¹⁴MHS: Microprocessor Hardware Specification - In dieser Datei steht die Konfigurationen der implementierten Hardwaremodule.

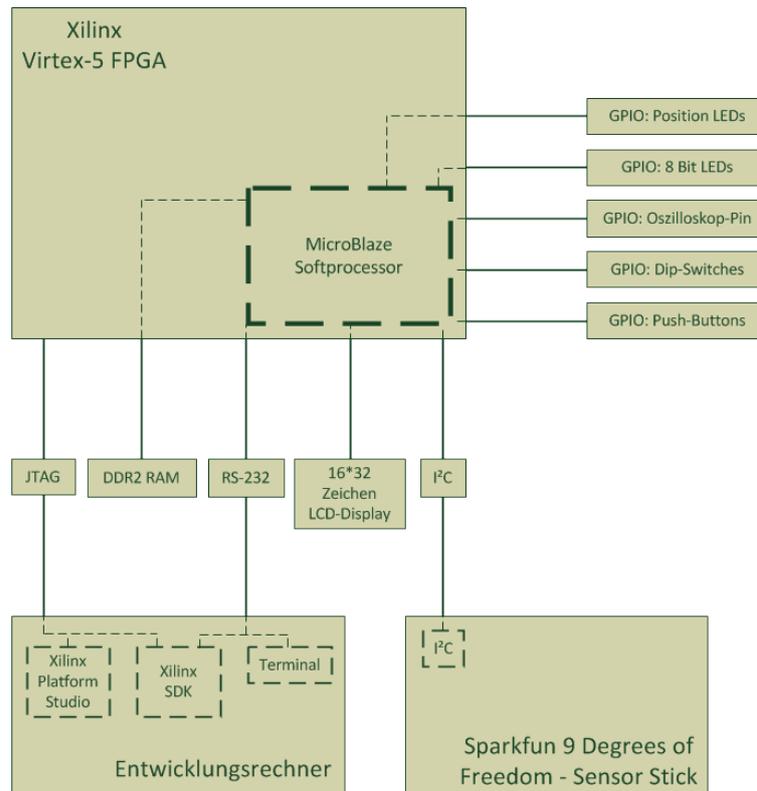


Abbildung 5.1: Blockdiagramm der externen Schnittstellen des Embedded Systems

5.2.1 Konfiguration des MicroBlaze

Mit dem zuvor erwähnten *Base-System-Builder* wird nach Auswahl des Xilinx Virtex-5 ML507 und des Speicherortes für das Projekt eine Auswahl mit verfügbaren Prozessoren und Prozessorkonfigurationen dargestellt. Auf dem Virtex-5 kann ein MicroBlaze Mikroprozessor mit maximal 125 MHz getaktet und 64 Kilobyte BRAM ausgestattet werden. Da der Ressourcenbedarf zum Zeitpunkt der Hardwareerstellung nicht bekannt ist, werden für beide Parameter die Maximalwerte ausgewählt. Die Größe des BRAM wurde so groß gewählt, damit möglichst die gesamte Software darin abgelegt werden kann und nicht mittels Linker Script in das DDR2-SDRAM ausgelagert werden muss. Der Vorteil einer komplett im BRAM liegenden Implementierung besteht darin, dass die Zugriffszeiten um ein Vielfaches kürzer sind, als wenn Daten aus dem DDR2-SDRAM geladen werden müssen. Sollte die Softwareimplementierung doch größer als 64 Kilobyte werden, so können bestimmte Teile der Software mit Hilfe des Linker Scripts in den DDR2-SDRAM ausgelagert werden. Um die Zugriffe auf die ausgelagerten Softwareabschnitte zu beschleunigen, können bei der Erstellung des Systems durch den *Base-System-Builder* die Instruktionen-

und Datencaches direkt aktiviert werden. Die Größe der Caches ist auf 32 Kilobyte je Cache limitiert, wobei auch hier wieder das Maximum ausgewählt wird.

Anschließend können alle später benötigten und an dieser Stelle verfügbaren Hardwaremodule ausgewählt und dem System hinzugefügt werden:

- DDR2-SDRAM
- Interrupt Controller
- Timer
- RS-232 Schnittstelle
- GPIO: Dip-Switches
- GPIO: Push-Buttons
- GPIO: 8 Bit LEDs
- GPIO: Position LEDs

DDR2-SD RAM

Da der kompilierte C-Code aufgrund eingebundener Bibliotheken größer als 64 Kilobyte sein wird und der maximal verfügbare Blockram des MicroBlaze auf 64 Kilobyte begrenzt ist, wird dem Hardwaredesign eine Anbindung des MicroBlaze an den dediziert vorhandenen DDR2-SDRAM über den MPMC¹⁵ hinzugefügt.

Um zu zeigen, dass sogar einzelne Funktionen den vorhandenen Speicher des BRAMs fast vollständig belegen können, ist in der folgenden Tabelle der Speicherplatzbedarf verschiedener Ausgabefunktionen in C dargestellt:

Funktion	Größe in Byte
print()	185 Byte
putnum()	284 Byte
xil_printf()	2.953 Byte
iprintf()	18.294 Byte
printf()	51.788 Byte

Tabelle 5.1: Speicherplatzbedarf verschiedener Ausgabefunktionen in C [44]

¹⁵MPMC: Multiport Memory Controller - Ein Memory Controller, der von Xilinx zur Verfügung gestellt wird und bis zu acht unterschiedliche Speicher verwalten kann.

Interrupt Controller

Das entworfene System hat zwei Interruptquellen. Da der MicroBlaze aber lediglich einen Interrupteingang hat, ist es notwendig einen Interruptcontroller zu verwenden, der zwischen dem MicroBlaze und den interruptfähigen Geräten verbaut ist und die Verwaltung der anfallenden Interrupts übernimmt. Der von Xilinx angebotene Interruptcontroller unterstützt Edge- und Level-sensitive Interrupts und kann bis zu 32 Geräte verwalten. Die Priorität der mit dem Interruptcontroller verbundenen Geräte wird durch die Reihenfolge ihrer Verbindung mit dem Interruptcontroller festgelegt. In diesem Projekt hat das Sampling der Sensoren die höchste Priorität, gefolgt von den Interrupts der Push-Buttons. Die für das System genutzten Dip-Switches werden periodisch mit den Sensoren abgefragt.

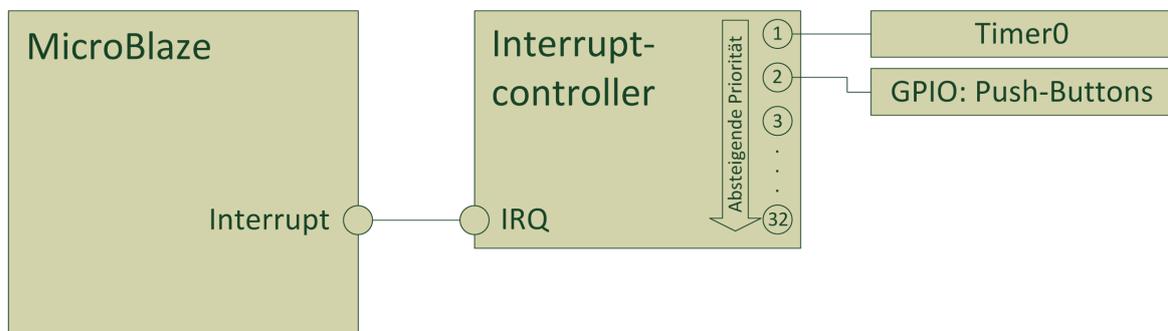


Abbildung 5.2: Interrupthierarchie und Anschluss der interruptfähigen Komponenten

Timer

Um regelmäßiges Sampling der Sensorik gewährleisten zu können, wird vom System ein Timer zur Erzeugung einer Periode von 10 Millisekunden genutzt. Der MicroBlaze Mikroprozessor wird mit 125 MHz getaktet. Um einen dem Mikroprozessor entsprechend getakteten Timer mit einer Periode von 10 Millisekunden, also mit 100 Hz zu erhalten, muss das Load-Register¹⁶ des Timers auf den Wert 1.250.000 eingestellt werden.

$$\text{Wert}_{\text{Load-Register}} = \frac{\text{Takt des Timers}}{\text{Durchläufe pro Sekunde}} = \frac{125.000.000}{100} = 1.250.000$$

Serielle Schnittstelle

Die serielle Schnittstelle wird für die Kommunikation zwischen dem Entwicklungsrechner und dem Embedded-System genutzt. Die Übertragungsrate wird auf 115200 Baud¹⁷ eingestellt. Während der Entwicklung des Systems wird die RS-232 Schnittstelle für das

¹⁶Load-Register: Dieses Register enthält den Wert, bis zu dem der Timer hochzählt. Bei Erreichen des eingestellten Wertes wird ein Interrupt erzeugt und von vorne begonnen zu zählen.

¹⁷Baud: Einheit für die Übertragungsrate einer Verbindung, wobei eine Übertragungsrate von 1 Baud bedeutet, dass 1 Symbol pro Sekunde übertragen werden kann.

Debugging genutzt. Im Normalbetrieb wird die Schnittstelle zur Übertragung der Daten an ein Terminalprogramm auf dem Entwicklungsrechner eingesetzt.

Dip-Switches, Push-Buttons, LEDs

Bei den Dip-Switches, Push-Buttons und LEDs handelt es sich um GPIO-Komponenten, die durch das Beschreiben und Lesen von Speicheradressen ausgewertet und gesteuert werden können. Dieses Vorgehen nennt sich Memory-Mapped-IO und ermöglicht es dem Entwickler angeschlossene Peripherie durch Speicherzugriffe zu steuern. Dazu werden dem Gerät Adressen aus dem zur Verfügung stehenden Adressraum des Mikrocontrollers (siehe Grafik B.6) zugewiesen und die External Ports des FPGAs (siehe Grafik B.5) mit den Pins der Geräte [15] in der .UCF-Datei verbunden:

```
#### Modul LEDs 8Bit
Net LEDs_8Bit_GPIO_IO_O_pin<0> LOC=AE24 | IOSTANDARD=LVCMOS18 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_8Bit_GPIO_IO_O_pin<1> LOC=AD24 | IOSTANDARD=LVCMOS18 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_8Bit_GPIO_IO_O_pin<2> LOC=AD25 | IOSTANDARD=LVCMOS18 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_8Bit_GPIO_IO_O_pin<3> LOC=G16 | IOSTANDARD=LVCMOS25 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_8Bit_GPIO_IO_O_pin<4> LOC=AD26 | IOSTANDARD=LVCMOS18 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_8Bit_GPIO_IO_O_pin<5> LOC=G15 | IOSTANDARD=LVCMOS25 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_8Bit_GPIO_IO_O_pin<6> LOC=L18 | IOSTANDARD=LVCMOS25 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_8Bit_GPIO_IO_O_pin<7> LOC=H18 | IOSTANDARD=LVCMOS25 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;

#### Modul LEDs 5Bit
Net LEDs_Positions_GPIO_IO_O_pin<0> LOC=E8 | IOSTANDARD=LVCMOS33 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_Positions_GPIO_IO_O_pin<1> LOC=AF23 | IOSTANDARD=LVCMOS33 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_Positions_GPIO_IO_O_pin<2> LOC=AG12 | IOSTANDARD=LVCMOS33 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net LEDs_Positions_GPIO_IO_O_pin<3> LOC=AG23 | IOSTANDARD=LVCMOS33 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net LEDs_Positions_GPIO_IO_O_pin<4> LOC=AF13 | IOSTANDARD=LVCMOS33 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

Modul Push Buttons

```
Net Push_Buttons_5Bit_GPIO_IO_I_pin<0> LOC=AJ6 | IOSTANDARD=LVCMOS33 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net Push_Buttons_5Bit_GPIO_IO_I_pin<1> LOC=AJ7 | IOSTANDARD=LVCMOS33 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net Push_Buttons_5Bit_GPIO_IO_I_pin<2> LOC=V8 | IOSTANDARD=LVCMOS33 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net Push_Buttons_5Bit_GPIO_IO_I_pin<3> LOC=AK7 | IOSTANDARD=LVCMOS33 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net Push_Buttons_5Bit_GPIO_IO_I_pin<4> LOC=U8 | IOSTANDARD=LVCMOS33 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

Modul Dip Switches

```
Net DIP_Switches_8Bit_GPIO_IO_I_pin<0> LOC=U25 | IOSTANDARD=LVCMOS18 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net DIP_Switches_8Bit_GPIO_IO_I_pin<1> LOC=AG27 | IOSTANDARD=LVCMOS18 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net DIP_Switches_8Bit_GPIO_IO_I_pin<2> LOC=AF25 | IOSTANDARD=LVCMOS18 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net DIP_Switches_8Bit_GPIO_IO_I_pin<3> LOC=AF26 | IOSTANDARD=LVCMOS18 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net DIP_Switches_8Bit_GPIO_IO_I_pin<4> LOC=AE27 | IOSTANDARD=LVCMOS18 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net DIP_Switches_8Bit_GPIO_IO_I_pin<5> LOC=AE26 | IOSTANDARD=LVCMOS18 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net DIP_Switches_8Bit_GPIO_IO_I_pin<6> LOC=AC25 | IOSTANDARD=LVCMOS18 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

```
Net DIP_Switches_8Bit_GPIO_IO_I_pin<7> LOC=AC24 | IOSTANDARD=LVCMOS18 |  
  PULLDOWN | SLEW=SLOW | DRIVE=2;
```

Nach Abschluss des Assistenten werden dem Hardwareentwurf noch zwei GPIO-Komponenten (LCD-Display und Oszilloskop) und das I²C EEPROM zur Kommunikation mit dem Sensor hinzugefügt.

5.2.1.1 LCD-Display und Oszilloskop

Wird das System ohne Entwicklungsrechner und RS-232 Schnittstelle verwendet, können die gesammelten Daten auf dem LCD-Display ausgegeben werden. Hierbei ist zu beachten, dass das LCD-Display für den Aufbau der Zeichen einige Zeit benötigt und weniger Platz für Informationen bietet als das Terminal. Aus diesen zwei Gründen werden die auszugebenden Daten auf die erste mit den Dip-Switches ausgewählte Information (siehe Tabelle 5.3) begrenzt.

Zur Messung des Zeitverhaltens des Systems werden zwei Pins zum Anschluss eines Oszilloskops herausgeführt.

Die Pins beider Komponenten werden durch entsprechende Einträge in der *.UCF*-Datei mit dem FPGA verbunden:

```
#### Modul LCD
Net lcd_ip_0_lcd_pin<0> LOC=AC9 | IOSTANDARD=LVCMOS33 | TIG | PULLDOWN;
Net lcd_ip_0_lcd_pin<1> LOC=J17 | IOSTANDARD=LVCMOS25 | TIG | PULLDOWN;
Net lcd_ip_0_lcd_pin<2> LOC=AC10 | IOSTANDARD=LVCMOS33 | TIG | PULLDOWN;
Net lcd_ip_0_lcd_pin<3> LOC=T11 | IOSTANDARD=LVCMOS33 | TIG | PULLDOWN;
Net lcd_ip_0_lcd_pin<4> LOC=G6 | IOSTANDARD=LVCMOS33 | TIG | PULLDOWN;
Net lcd_ip_0_lcd_pin<5> LOC=G7 | IOSTANDARD=LVCMOS33 | TIG | PULLDOWN;
Net lcd_ip_0_lcd_pin<6> LOC=T9 | IOSTANDARD=LVCMOS33 | TIG | PULLDOWN;

#### Modul Oszilloskop
Net Oszilloskop_2Bit_GPIO_IO_O_pin<0> LOC=G32 | IOSTANDARD=LVCMOS18 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
Net Oszilloskop_2Bit_GPIO_IO_O_pin<1> LOC=J32 | IOSTANDARD=LVCMOS18 |
    PULLDOWN | SLEW=SLOW | DRIVE=2;
```

5.2.1.2 I²C EEPROM

Das I²C EEPROM ist eine von Xilinx zur Verfügung gestellte Komponente, die die Kommunikation über den I²C Bus ermöglicht. In diesem Projekt fungiert der Mikroprozessor als I²C-Master und der Sensor als I²C-Slave.

Xilinx stellt für den Betrieb dieses Hardwaremoduls Treiber zur Verfügung, mit denen direkt auf Geräte geschrieben und von Geräten gelesen werden kann. Die Adressierung der an den I²C Bus angeschlossenen Geräte erfolgt durch Adressierung der in den Datenblättern angegebenen Adressen:

Sensor	Adresse
Accelerometer	0x53
Gyroskop	0x68
Kompass	0x1E

Tabelle 5.2: I²C Adressen der Sensoren auf dem Sensor Stick

Beim Hinzufügen des I²C EEPROMs zum Hardwareentwurf werden automatisch entsprechende Einträge in der *.UCF*-Datei vorgenommen. Jedoch sind die vom Xilinx Platform Studio automatisch eingerichteten Verbindungen bei dem in diesem Projekt genutzt Virtex-5 ML507 nicht nutzbar, da während eines vorangegangenen Projekt ein Extension-Board auf den Pins untrennbar montiert worden ist und die zwei für I²C vorgesehenen Pins des *J5-Pin-Headers* verdeckt. Stattdessen werden die Pins *H33* und *H34* genutzt (siehe Grafik D.1).

```
#### Modul IIC EEPROM
Net fpga_0_IIC_EEPROM_Sda_pin LOC=H34 | SLEW=SLOW | DRIVE=6 |
  IOSTANDARD=LVCMOS33;
Net fpga_0_IIC_EEPROM_Scl_pin LOC=H33 | SLEW=SLOW | DRIVE=6 |
  IOSTANDARD=LVCMOS33;
```

5.2.2 Synthese

Der Abschluss des Hardwareentwurfs ist die Synthese des Hardwareentwurfs, dessen Ziel eine *.BIT*-Datei ist, mit der das FPGA über die JTAG Schnittstelle programmiert werden kann.

Dieser Prozess gliedert sich grob in zwei Schritte:

1. Erzeugung der Hardwarenetzliste

In diesem Schritt werden die Netzlisten synthetisiert. Dazu analysiert der *Platform-Generator* zunächst die *.MHS*- und *.MPD*-Dateien und erstellt Wrapper-Dateien für jede instantiierte Komponente, welche alle notwendigen VHDL Schnittstelleninformationen für die Synthese der *.NGC*-Dateien enthält.

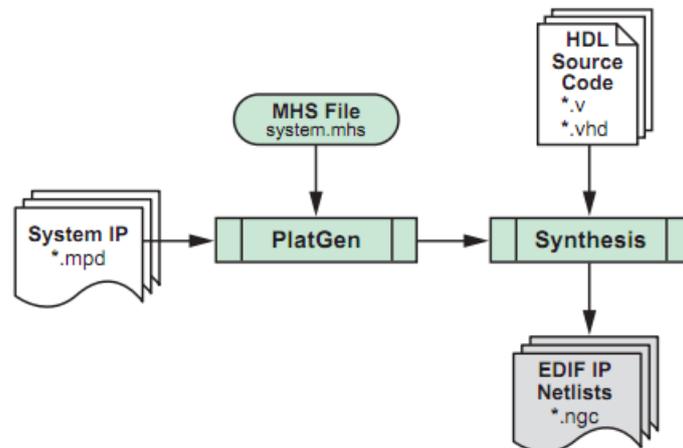


Abbildung 5.3: Erzeugung der Hardwarenetzlisten [45]

2. Erzeugung des Hardwarebitstreams

Während der Erzeugung des Hardwarebitstreams (.BIT-Datei) werden die zuvor erzeugten .NGC-Dateien durch verschiedene Programme (*NGDBuild*, *MAP*, *PAR*, *TRACE* und *BitGen*) und unter Berücksichtigung der Board-Parameter (Virtex-5 ML507), sowie der .UCF-Datei zu einem Hardwarebitstream zusammengefügt und im Projektverzeichnis gespeichert.

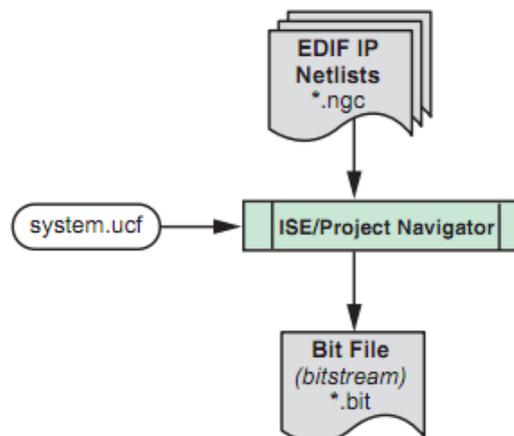


Abbildung 5.4: Erzeugung des Hardwarebitstreams [45]

Die in diesen beiden Schritten entstandene Datei wird zur Programmierung des FPGAs genutzt und ist das Ergebnis des Hardwareentwurfsprozesses.

5.3 Software

Bei der Software für das im Rahmen dieser Bachelorarbeit entwickelte Embedded-System wird alle 10 Millisekunden durch einen Timer ein Interrupt erzeugt. Dieser periodisch auftretende Interrupt sorgt dafür, dass die Daten der Sensoren regelmäßig gelesen werden und so nicht verloren gehen.

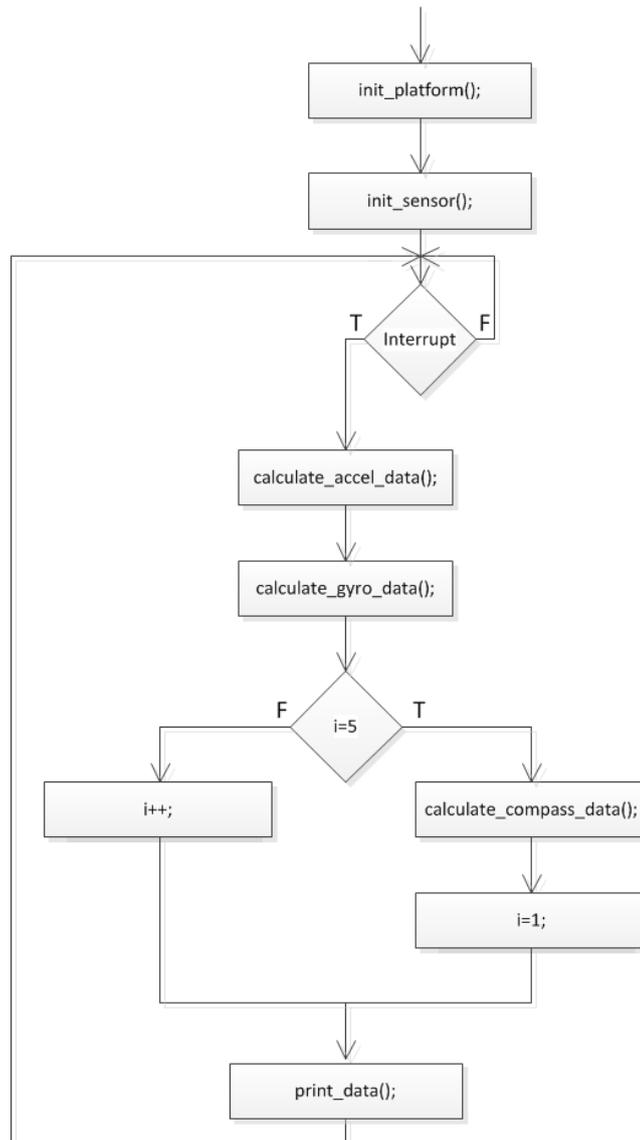


Abbildung 5.5: Struktogramm des Softwareprojekts

Bei der Implementierung der Software wird auf Sparsamkeit geachtet, damit möglichst das ganze Programm im Blockram des Mikroprozessors Platz findet. Aus diesem Grund wird

auf aufwändige Funktionen zur Ausgabe der Debuginformationen verzichtet und stattdessen hexadezimal mit der von Xilinx zur Verfügung gestellten Funktion `xil_printf()`; ausgegeben, da diese wie in Tabelle 5.1 beschrieben, sehr ressourcenschonend arbeitet. Da neben Debuginformationen während der Entwicklung auch Statusinformationen während des Normalbetriebs über das Terminal oder das LCD-Display ausgegeben werden sollen, werden die Messwerte mittels Shift-Operationen auf Integerwerte gerundet und ausgegeben.

Das Linker Script wird so eingestellt, dass das ganze Programm im Blockram des Mikroprozessors abgelegt wird:

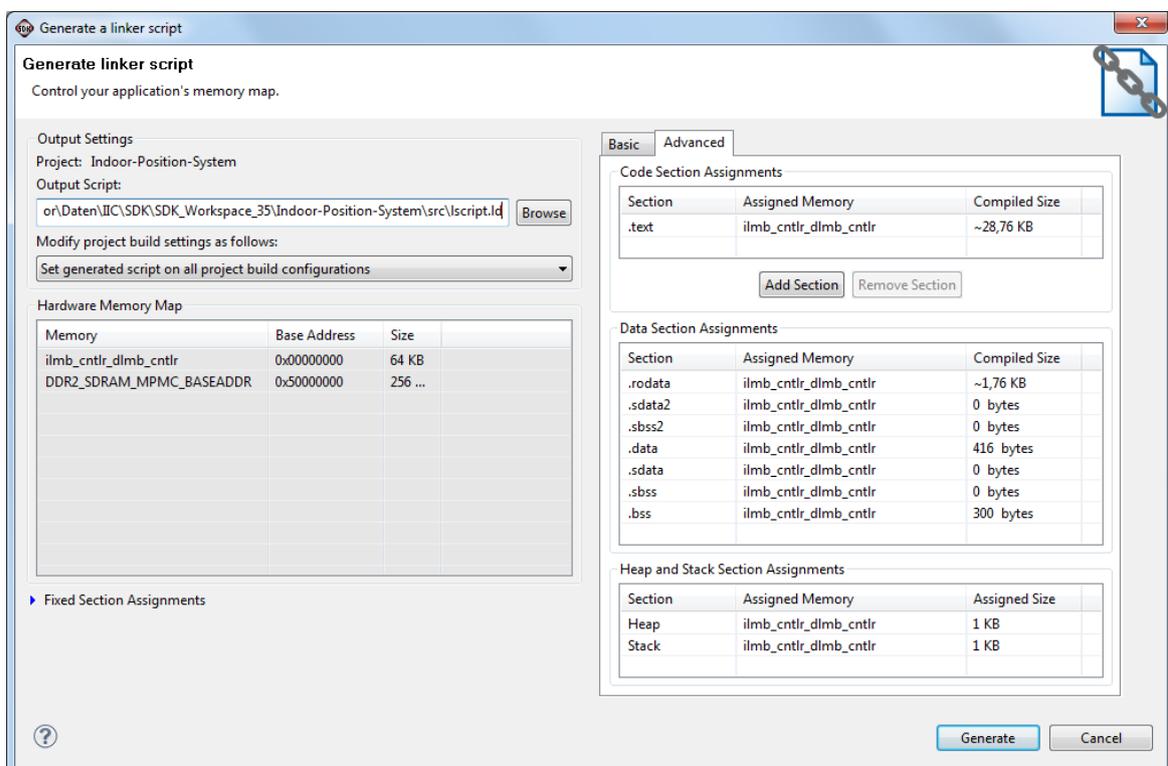


Abbildung 5.6: Linker Script des Softwareprojekts im Xilinx SDK

5.3.1 Treiber für Hardware-Module

Xilinx stellt für alle Hardwaremodule, die im IP Katalog von Xilinx angeboten werden, Treiber zur Verfügung. Diese Treiber unterstützen mindestens rudimentäre Funktionen, um die Funktionsfähigkeit eines angeschlossenen Gerätes testen zu können.

5.3.2 Software-Routinen zur Verarbeitung der Sensordaten

Die Verarbeitung der vom Sensor gemessenen Daten erfolgt in mehreren Schritten. Vor Beginn der regulären Messungen werden die Sensoren initialisiert und wie in den Kapiteln 4.2.2, 4.3.2 und 4.4.2 beschrieben, kalibriert.

Nachdem die Kalibrierung abgeschlossen ist, werden die Daten der Sensoren in regelmäßigen Zeitintervallen von 10 Millisekunden vom Mikroprozessor gelesen und ausgewertet. Anschließend werden die Messwerte mit Hilfe der bei der Kalibrierung errechneten Skalierungsfaktoren und Offsets korrigiert und sensorspezifisch ausgewertet. Die so entstehenden Werte haben verschiedene Einheiten und fordern somit unterschiedliche Verfahren zur Weiterverarbeitung ein.

Die Daten des Gyroskops werden akkumuliert, da es sich bei den gemessenen Daten um $\frac{\text{Grad}}{\text{s}}$ handelt und in diesem Projekt nicht die Drehrate, sondern die absolute Drehung ermittelt werden soll. Auf ähnliche Weise wird mit den Messwerten des Beschleunigungssensors verfahren. Diese werden zunächst von g in $\frac{m}{s^2}$ umgerechnet und anschließend akkumuliert, um die absolute Beschleunigung in $\frac{m}{s^2}$ zu bestimmen. Aus dieser kann durch Integration die Geschwindigkeit in $\frac{m}{s}$ und durch erneute Integration die Distanz in m ermittelt werden.

Die bei der Inertialsensorik eingesetzten Verfahren sind stark fehleranfällig, was darauf zurückzuführen ist, dass neben dem korrekten Messwert auch die bei jedem Sample auftretenden Messabweichungen akkumuliert werden.

Anders ist es bei dem Kompass, dessen Messwerte zu jeder Zeit die gleiche Güte aufweisen und deswegen häufig als *driftlos* bezeichnet werden.

In der folgenden Abbildung ist für jeden Sensor ein Auswertungszyklus dargestellt:

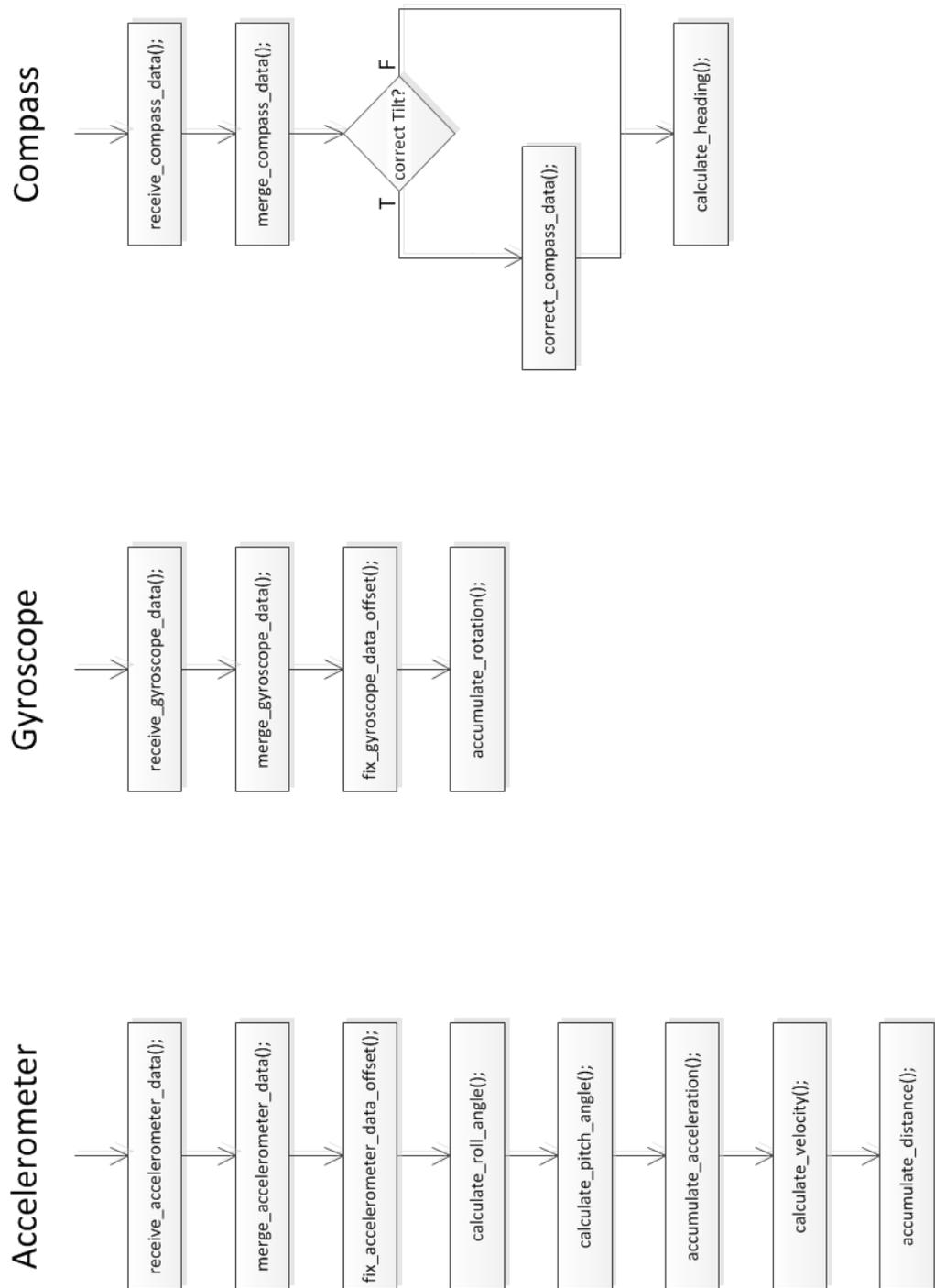


Abbildung 5.7: Auswertungszyklen der Sensoren

5.3.3 Datencontainer

Die ausgewerteten Daten werden in einem Array mit Platz für 25 Elemente vom Typ `int16_t` zur Verfügung gestellt. Die Bedeutung der Daten ist in der folgenden Tabellen beschrieben:

Array[Position]	Interpretation
<code>sensor_data[0]</code>	Kompass Rohdaten X-Achse
<code>sensor_data[1]</code>	Kompass Rohdaten Y-Achse
<code>sensor_data[2]</code>	Kompass Rohdaten Z-Achse
<code>sensor_data[3]</code>	Gyroskop Rohdaten X-Achse
<code>sensor_data[4]</code>	Gyroskop Rohdaten Y-Achse
<code>sensor_data[5]</code>	Gyroskop Rohdaten Z-Achse
<code>sensor_data[6]</code>	Accelerometer Rohdaten X-Achse
<code>sensor_data[7]</code>	Accelerometer Rohdaten Y-Achse
<code>sensor_data[8]</code>	Accelerometer Rohdaten Z-Achse
<code>sensor_data[9]</code>	Temperatur
<code>sensor_data[10]</code>	Rotation in Grad um die X-Achse (Roll)
<code>sensor_data[11]</code>	Rotation in Grad um die Y-Achse (Pitch)
<code>sensor_data[12]</code>	Betriebszeit in Sekunden
<code>sensor_data[13]</code>	Ausrichtung des Sensors (Nordpol = 0Grad)
<code>sensor_data[14]</code>	Magnitude des Kompass (Magnetische Feldstärke aller drei Achsen)
<code>sensor_data[15]</code>	<i>ungenutzt</i>
<code>sensor_data[16]</code>	Gyroskop Integration (Rotation in Grad) X-Achse
<code>sensor_data[17]</code>	Gyroskop Integration (Rotation in Grad) Y-Achse
<code>sensor_data[18]</code>	Gyroskop Integration (Rotation in Grad) Z-Achse
<code>sensor_data[19]</code>	Geschwindigkeit (in $\frac{m}{s}$) X-Achse
<code>sensor_data[20]</code>	Geschwindigkeit (in $\frac{m}{s}$) Y-Achse
<code>sensor_data[21]</code>	Geschwindigkeit (in $\frac{m}{s}$) Z-Achse
<code>sensor_data[22]</code>	Zurückgelegte Entfernung (in m) X-Achse
<code>sensor_data[23]</code>	Zurückgelegte Entfernung (in m) Y-Achse
<code>sensor_data[24]</code>	Zurückgelegte Entfernung (in m) Z-Achse

Tabelle 5.3: Bedeutung der im Datencontainer zur Verfügung gestellten Werte

Kapitel 6

Test

Um die Güte der vom System zur Verfügung gestellten Messwerte testen zu können, wird für jeden der drei auf dem Sensor Stick verbauten Sensoren ein Test entwickelt, mit dem die zu Beginn des Projekts festgelegten Funktionalitäten getestet werden. Die Bewertung der Tests erfolgt für jeden Sensor am Ende des jeweiligen Testergebnisses (siehe Kapitel 6.1.2, 6.2.2 und 6.3.2).

6.1 Gyroskop

Die vom ITG-3200 aufgezeichneten Messwerte werden verwendet, um beliebige Rotationen des Sensors beziehungsweise des Objekts an dem dieser befestigt ist um die drei Koordinatenachsen auswerten zu können. Es werden Tests bezüglich Rotationen um einzelne Achsen, um alle drei Achsen und des Driftverhaltens in Ruhe durchgeführt.

6.1.1 Testaufbau

Zur Verifizierung der Messwerte des Drehratensensors wurde eine runde Scheibe aus Acrylglas mittig an einer Rotationsachse befestigt. Der Drehratensensor wurde möglichst zentriert auf der Acrylglasscheibe befestigt, damit die Auswirkungen der Rotation auf die anderen beiden Achsen möglichst gering ist.

Um mit diesem Aufbau alle drei Achsen des Gyroskops testen zu können, wird der Sensor im Versuchsaufbau in die entsprechende Richtung gedreht, wodurch gewährleistet werden kann, dass die Tests unter konsistenten Bedingungen durchgeführt werden. Nachdem alle drei Achsen einzeln in 90 Grad Schritten getestet worden sind, wird der Sensor 360 Grad um jede der drei Koordinatenachsen gedreht. Als letztes wird das Driftverhalten aller drei Koordinatenachsen des Sensors in Ruhe getestet.

In allen Tests wird darauf geachtet, dass die Zeit zwischen dem Reset des Systems und

dem Abschluss des Tests möglichst kurz gehalten wird, damit der Einfluss des Drifts den dieser Sensor bedingt durch seine Bauart hat, möglichst gering ist.

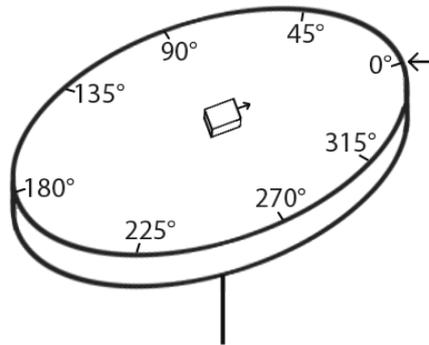


Abbildung 6.1: Schematischer Testaufbau Gyroskop

6.1.2 Testergebnis

Im folgenden werden die Ergebnisse der zuvor beschriebenen Tests des Drehratensensors detailliert dargestellt:

Testcase	Soll	Ist	Abweichung in Grad
Rotation X-Achse	90	91	1
	-90	-89	1
	90	90	0
	180	178	2
	-180	-181	1
	180	177	3
	270	276	6
	-270	-269	1
	270	273	3
	360	362	2
	-360	-354	6
	360	355	5

Fortsetzung der Tabelle auf der nachfolgenden Seite

Fortsetzung der Tabelle von der vorhergehenden Seite

Testcase	Soll	Ist	Abweichung in Grad
Rotation Y-Achse	90	90	0
	-90	-88	2
	90	91	1
	180	178	2
	-180	-178	-2
	180	183	3
	270	274	4
	-270	-272	2
	270	267	3
	360	354	6
	-360	-356	4
	360	356	4
Rotation Z-Achse	90	89	1
	-90	-88	2
	90	90	0
	180	182	2
	-180	-179	1
	180	183	3
	270	267	3
	-270	-273	3
	270	272	2
	360	356	4
	-360	-364	4
	360	357	3

Tabelle 6.1: Test - Rotation um 1 Achse - Gyroskop

Für den nächsten Test wird der Sensor in möglichst kurzer Zeit 360 Grad um jede seiner drei Achsen gedreht:

Testcase	Soll	Ist	Abweichung in Grad
X-Achse	360	375	15
Y-Achse	-360	-372	12
Z-Achse	360	366	6

Tabelle 6.2: Test - 360 Grad Rotation - Gyroskop

Die nachfolgend dargestellten Drifts wurden gemessen, während sich der Sensor 60 Sekunden lang in Ruhe befand:

Testcase	Soll	Ist	Abweichung in Grad
Drift X-Achse	0	-48	-48
	0	-48	-48
	0	-46	-46
Drift Y-Achse	0	-59	-59
	0	-60	-60
	0	-58	-58
Drift Z-Achse	0	49	49
	0	52	52
	0	51	51

Tabelle 6.3: Test - Drift - Gyroskop

Die durchgeführten Tests werden zur Verifizierung der Korrektheit der Daten des Gyroskops genutzt. Die Ergebnisse des ersten Tests (siehe Tabelle 6.1.2) zeigen, dass der Sensor bei den durchgeführten Rotationen einen Fehler von etwa 1 Grad pro 90 Grad Rotation, also etwa 1,1% aufweist. Die Rahmenbedingungen dieses Tests wurden so gewählt, dass zum einen die Rotationsgeschwindigkeit bei allen Messungen möglichst konstant und zum anderen die Zeit zwischen Reset des Sensors und Abschluss der Messung möglichst kurz war. Bei Betrachtung der Ergebnisse des zweiten Tests (siehe Tabelle 6.2) ist zu erkennen, dass die Messabweichung größer ist, als bei den Einzelmessungen des Versuchs zuvor. Dieses kann entweder auf den Drift des Sensors, die achsübergreifende Sensitivität (die fälschlicherweise gemessene Rotation, um nicht rotierte Achsen) oder auf Fehler, die bei der Durchführung dieses Versuchs aufgetreten sein können, zurückgeführt werden. Die Ergebnisse des dritten Tests (siehe Tabelle 6.3) zeigen, dass der Sensor wie erwartet bereits nach kurzer Zeit stark driftet. Allerdings scheint der Drift des Sensors näherungsweise konstant zu sein, was bedeutet, dass dieser durch entsprechende arithmetische Operationen gemindert werden kann.

6.2 Accelerometer

Da der Beschleunigungssensor sowohl statische Beschleunigungen, wie die Erdanziehungskraft, als auch dynamische Beschleunigungskräfte, wie lineare Translationen erfasst, kann einerseits wie in Kapitel 4.5.1 beschrieben, die Neigung des Sensors beziehungsweise die Neigung des Objekts an dem der Sensor befestigt ist, bestimmt und andererseits die zurückgelegte Strecke aus den Beschleunigungen approximiert werden. Nachfolgend wird zuerst die Approximation der zurückgelegten Distanz und anschließend die Neigungsermittlung getestet.

6.2.1 Testaufbau

Um die zurückgelegte Distanz testen zu können, wurde ein 3,5 Meter langes gerades Kantholz am Arbeitsplatz befestigt, das die Linearität der Translationen gewährleistet wird. Bei der Versuchsdurchführung wurde darauf geachtet, dass die Mehrheit der auftretenden Beschleunigungen in dem für den Sensor messbaren Bereich lagen.

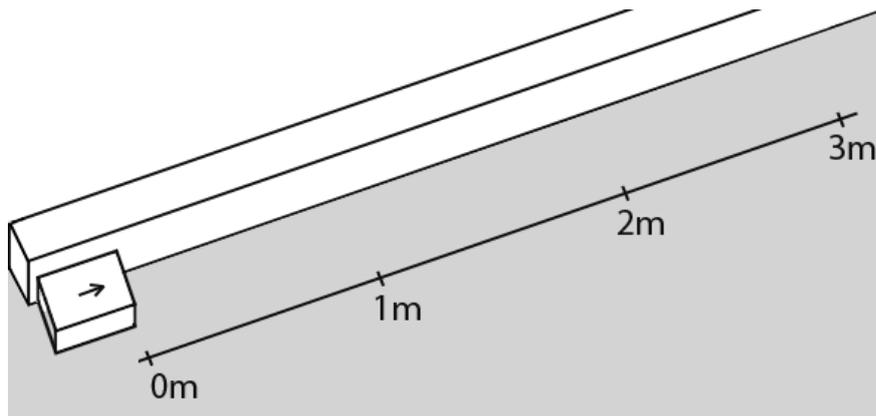


Abbildung 6.2: Schematischer Testaufbau zur Distanzmessung mit dem Accelerometer

Der zweite Versuchsaufbau dient der Untersuchung der vom Beschleunigungsmesser gemessenen statischen Beschleunigung, also der Erdanziehungskraft, mit deren Hilfe die Neigung des Sensors bestimmt werden kann. Hierzu sind zwei Acrylglasplatten über ein Scharnier verbunden worden, wobei die untere dieser Platten zusätzlich mit dem Arbeitsplatz verbunden worden ist. An der zweiten Acrylglasplatte wurde ein Band befestigt, mit dessen Hilfe über einen Haken an der Wand der Anstellwinkel dieser verändert werden kann.

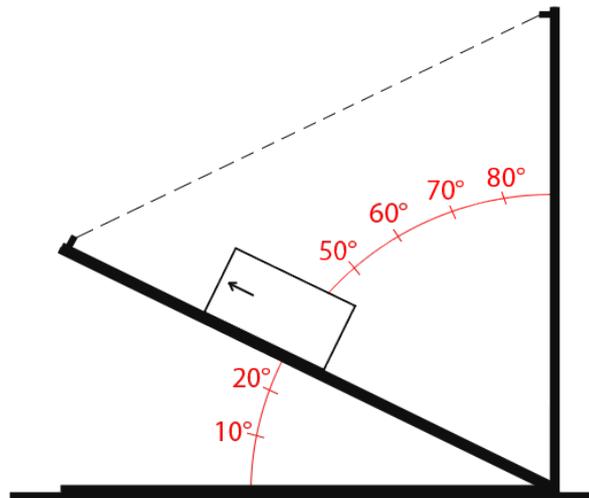


Abbildung 6.3: Schematischer Testaufbau zur Messung von Roll- und Nickwinkeln mit dem Beschleunigungssensor

6.2.2 Testergebnis

Distanzdrift über 60 Sekunden:

Testcase	Soll	Ist	Abweichung in m
Drift X-Achse	0	76	76
Drift Y-Achse	0	-127	127
Drift Z-Achse	0	-111	111

Tabelle 6.4: Test - Distanzdrift - Accelerometer

Zur Ermittlung der zurückgelegten Distanz werden die vom Accelerometer gemessenen Beschleunigungen doppelt integriert:

Testcase	Soll	Ist	Abweichung in m
Distanz X-Achse	0,25	0	0,25
	0,5	1	0,5
	0,75	1	0,25
	1,0	2	1,0
	1,5	2	0,5
	2,0	4	2,0
	2,5	5	2,5
	3,0	6	3,0
Distanz Y-Achse	0,25	0	0,25
	0,5	1	0,5
	0,75	1	0,25
	1,0	1	0
	1,5	2	0,5
	2,0	3	1,0
	2,5	3	0,5
	3,0	5	2,0
Distanz Z-Achse	0,25	0	0,25
	0,5	0	0,5
	0,75	1	0,25
	1,0	1	0
	1,5	1	0,5
	2,0	1	1,0
	2,5	1	1,5
	3,0	2	1,0

Tabelle 6.5: Test - Distanz - Accelerometer

Zur Ermittlung der Roll- und Nickwinkel wird der in Kapitel 4.5.1 beschriebene Algorithmus verwendet:

Testcase	Soll	Ist	Abweichung in Grad
Roll-Winkel	-90	-90	0
	-80	-78	2
	-60	-58	2
	-45	-45	0
	-20	-21	1
	-10	-9	1
	0	0	0
	10	9	1
	20	20	0
	45	45	0
	60	61	1
	80	78	2
	90	89	1
Nick-Winkel	-90	-88	2
	-80	-81	1
	-60	-61	1
	-45	-45	0
	-20	-20	0
	-10	-9	1
	0	0	0
	10	10	0
	20	20	0
	45	45	0
	60	59	1
	80	81	1
	90	89	1

Tabelle 6.6: Test - Roll- und Nickwinkel - Accelerometer

Die vom System ausgegebenen Accelerometerdaten sind auf Meter gerundet (siehe Kapitel 5.3), was zur Folge hat, dass sehr kleine Distanzen gar nicht und größere Distanzen nur unter dem Einfluss von starkem Drift erfasst werden können. In Tabelle 6.4 ist zu sehen, dass der Sensor in 60 Sekunden trotz vollkommener Ruhe eine Distanz von durchschnittlich 105 Meter je Achse erfasst hat. Die Quelle dieses Fehlers beruht auf den Messwerten des Sensors, diese schwanken in Ruhe zwischen +4 LSB¹⁸ und -4 LSB, wobei

¹⁸LSB: Least Significant Bit

1 LSB bei einem Messbereich von $\pm 4 g$ einer Genauigkeit von 7,8 Milli- g entspricht. Das bedeutet, dass der Sensor fälschlicherweise Beschleunigungen von bis zu 31,2 Milli- g wahrnimmt. Dieser Test erklärt das schlechte Ergebnis des zweiten Tests (siehe Tabelle 6.5), bei dem die Distanzmessung des Systems getestet worden ist. Es zeigte sich, dass die zurückgelegte Distanz bereits nach kurzer Strecke so ungenau ist, dass eine Abschätzung über zurückgelegte Distanzen auf diese Weise nicht gemacht werden kann. Hingegen zeigte der Test der vom Accelerometer gelieferten Daten zur Bestimmung der Neigung des Sensors (siehe Tabelle 6.6), dass das System die Neigung eines Objekts an dem der Sensor befestigt ist, ausreichend genau bestimmen kann, um mit diesen Messwerten eine Korrektur der Kompassdaten zur Bestimmung des Azimuthwinkels durchführen zu können. Die durchschnittliche Abweichung der Ist-Werte von den Soll-Werten beträgt weniger als 1 Grad pro Messwert und ist offensichtlich nicht von der Lage des Sensors im Raum abhängig.

6.3 Kompass

Der digitale Kompass misst das Magnetfeld der Erde (siehe Kapitel 4.2.1). Durch Auswertung dieser Daten (siehe Kapitel 4.5.2) kann der Azimuthwinkel, also die Ausrichtung des Sensors in Bezug zum Magnetfeld der Erde, bestimmt werden. Da der digitale Kompass keinem Drift unterliegt, wie es bei den beiden Inertialsensoren der Fall ist, wird für diesen Sensor lediglich die Bestimmung des Azimuthwinkels getestet.

6.3.1 Testaufbau

Zur Verifizierung der Messwerte des Kompass, wird der Sensor auf einer drehbaren runden Scheibe aus Acrylglas gefestigt. Eine Linie entlang der X-Achse durch das Zentrum des Sensors stellt die Richtung, in die der Sensor zeigt. Eine Schablone, die außerhalb von der Acrylglasscheibe befestigt ist, zeigt die Himmelsrichtung und den Azimuthwinkel zur Bestimmung der Soll-Werte an.

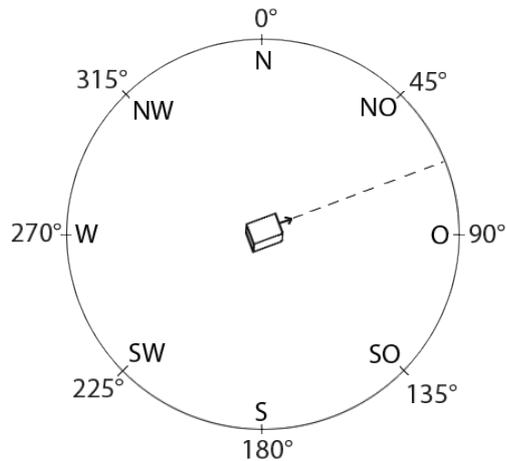


Abbildung 6.4: Schematischer Testaufbau zur Messung des Azimuthwinkels

6.3.2 Testergebnis

Testcase	Soll	Ist	Abweichung in Grad
Azimuthwinkel	0	2	2
	30	31	1
	60	58	2
	90	92	2
	120	115	5
	150	154	4
	180	181	1
	210	216	6
	240	245	5
	270	263	7
	300	303	3
	330	325	5

Tabelle 6.7: Test - Azimuth - Kompass

Die in Tabelle 6.7 angegebenen Messwerte sind Durchschnittswerte, die sich daraus ergeben haben, dass der Sensor einige Sekunden nicht bewegt wurde und ein Mittelwert über die teilweise stark schwankenden Messwerte errechnet worden ist. Die Werte des digitalen Kompasses schwanken teilweise sehr stark, sodass ein Filter zur Tilgung dieser Ausreißer notwendig ist, um gute Werte für den Azimuthwinkel zu erhalten. Auch wenn die

Messwerte des Kompasses nicht genau sind, ist dieser für die meisten Anwendungen ausreichend genau.

Kapitel 7

Fazit

Das Ziel dieses Projekts war die Entwicklung eines Systems, das mittels Inertial- und Magnetfeldsensorik Rotationen und Translationen auswerten und Angaben zur Position machen kann.

Bei der ursprünglich für diesen Zweck angeschafften Sensorik handelt es sich ebenfalls um ein MARG-Sensor Array, welches allerdings ein USB 2.0 Interface zur Kommunikation mit dem Xilinx Virtex-5 ML507 zur Verfügung hat. Der Vorteil dieses Sensor Arrays besteht darin, dass es aufgrund des USB Interfaces zur Kommunikation mit vielen verschiedenen Systemen geeignet ist. Allerdings ist diese Schnittstelle auch das Problem dieser Sensorik, denn anders als erwartet ist die Implementierung einer USB Schnittstelle deutlich aufwändiger als erwartet. Deswegen wurde der 9 Degrees of Freedom - Sensor Stick von Sparkfun angeschafft, der ein I²C Interface zur Verfügung stellt, das zur Kommunikation mit dem FPGA eingesetzt werden kann und mit dem dieses Projekt durchgeführt worden ist. Der Sensor Stick von Sparkfun liegt in der zweiten Version (SEN-10321) vor, welche Anfang Juli durch die dritte Version des Sensor Sticks (SEN-10724) ersetzt worden ist. Bei diesem wird ein neuer digitaler Kompass eingesetzt, der die Kalibrierung selbstständig durchführt und keine Probleme bezüglich stark schwankender Werte (siehe Kapitel 6.3.2) aufweisen soll, wie es die vorliegende Sensorik hat.

Die in Kapitel 6 durchgeführten Tests haben gezeigt, dass das entstandene System bedingt zur Erfassung von Rotationen und Translationen geeignet ist. Wie in Kapitel 6.2.2 gezeigt, können Neigungen (Roll- und Nickwinkel) sehr genau erfasst und zur Korrektur der Kompassdaten genutzt werden. Da sowohl das Gyroskop als auch das Accelerometer zum Driften neigen, ist die korrekte Auswertung von Translationen und Rotationen nicht ohne weiteres möglich. Wie in der Einleitung dieser Ausarbeitung (siehe Kapitel 1) erwähnt, werden zur Lösung dieser Probleme häufig verschiedene Filter eingesetzt. Auf die Implementierung solcher Filter wurde, wie in Kapitel 1 beschrieben, in diesem Projekt verzichtet.

Während der Entwicklung des Embedded Systems und der anschließenden Dokumentationsphase, wurden einige Ansätze für Verbesserungen identifiziert. Da das FPGA ausreichend Ressourcen zur Verfügung stellt, könnte ein Filter entwickelt werden, das einen ähnlich Ansatz, wie das von Madgwick *et al* [2] entworfene Filter verfolgt. Weiter könnten die verwendeten trigonometrischen Funktionen durch Lookup-Tabellen ersetzt oder mit angepassten Algorithmen in Hardware [46] errechnet werden.

Tabellenverzeichnis

4.1	Übersicht typischer Drehraten	20
5.1	Speicherplatzbedarf verschiedener Ausgabefunktionen in C	30
5.2	I ² C Adressen der Sensoren auf dem Sensor Stick	35
5.3	Bedeutung der im Datencontainer zur Verfügung gestellten Werte	41
6.1	Test - Rotation um 1 Achse - Gyroskop	44
6.2	Test - 360 Grad Rotation - Gyroskop	45
6.3	Test - Drift - Gyroskop	45
6.4	Test - Distanzdrift - Accelerometer	47
6.5	Test - Distanz - Accelerometer	48
6.6	Test - Roll- und Nickwinkel - Accelerometer	49
6.7	Test - Azimuth - Kompass	51
D.1	Push-Button Funktionsübersicht	74
D.2	Dip-Switch Funktionsübersicht	74
D.3	FPGA Ressourcenverbrauch	75

Abbildungsverzeichnis

3.1	Blockdiagramme (ML507 Evaluationsboard und MicroBlaze Softcore Prozessors)	8
4.1	9 Degrees of Freedom Sensor Stick von Sparkfun	11
4.2	Graph mit den Messwerten des HMC5843 nach 3D Rotation	13
4.3	X-Y-Graphen unter Einfluss verschiedener magnetischer Störungen	15
4.4	Aggregierte Daten in der X-Y-Ebene einer stationären Rotation	15
4.5	Aggregierte Daten in der X-Y-Ebene einer nicht stationären Rotation	15
4.6	Einfluss eines ferromagnetischen Objekts auf ein Magnetfeld	17
4.7	Aufbau eines MEMS-Beschleunigungssensors mit zwei Kondensatoren	17
4.8	Funktionsweise eines MEMS-Beschleunigungssensors	18
4.9	Schematischer Aufbau eines Resonanzgyroskops (Stimmgabelprinzip)	22
4.10	In Schwingung versetzte Testmassen	23
4.11	Aus Rotation und Schwingung resultierender Coriolis-Kraft-Effekt	23
4.12	Roll- und Nickwinkel eines Objektes im dreidimensionalen Raum	25
5.1	Blockdiagramm der externen Schnittstellen des Embedded Systems	29
5.2	Interrupthierarchie und Anschluss der interruptfähigen Komponenten	31
5.3	Erzeugung der Hardwarenetzlisten	36
5.4	Erzeugung des Hardwarebitstreams	36
5.5	Struktogramm des Softwareprojekts	37
5.6	Linker Script des Softwareprojekts im Xilinx SDK	38
5.7	Auswertungszyklen der Sensoren	40
6.1	Schematischer Testaufbau Gyroskop	43
6.2	Schematischer Testaufbau zur Distanzmessung mit dem Accelerometer	46
6.3	Schematischer Testaufbau zur Messung von Roll- und Nickwinkeln mit dem Beschleunigungssensor	47
6.4	Schematischer Testaufbau zur Messung des Azimuthwinkels	51
B.1	Bus Interfaces des Embedded-Systems	63

B.2	Ports des Embedded-Systems	64
B.3	Ports des Embedded-Systems erweiterte Ansicht (Schwerpunkt: Interrupt) . .	65
B.4	Ports des Embedded-Systems erweiterte Ansicht (Schwerpunkt: I ² C)	66
B.5	Externe Ports des Embedded-Systems	67
B.6	Adressen des Embedded-Systems	68
C.1	9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun (1)	69
C.2	9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun (2)	69
C.3	9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun (3)	69
C.4	9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun (4)	69
C.5	Schematic der 9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun	70
C.6	Blockdiagramm HMC5843	71
C.7	Blockdiagramm ADXL345	72
C.8	Blockdiagramm ITG-3200	73
D.1	Interfaces des Embedded-Systems	76
D.2	Sicht von oben auf das ML507 Evaluationsboard	77
D.3	Sicht von unten auf das ML507 Evaluationsboard	77
E.1	Terminalausgabe des Embedded Systems	78
E.2	Ausgabemodi auf das Virtex-5 LCD Displays	79

Literaturverzeichnis

- [1] CARUSO MICHAEL J. (HONEYWELL): *Applications of Magnetic Sensors for Low Cost Compass Systems.* , (Abgerufen am 28.05.2011).
- [2] SEBASTIAN O.H. MADGWICK: *An efficient orientation filter for inertial and inertial/magnetic sensor arrays.* , (Abgerufen am 16.05.2011), 2010.
- [3] SEBASTIAN O.H. MADGWICK: *Automated calibration of an accelerometers, magnetometers and gyroscopes - A feasibility study.* , (Abgerufen am 17.05.2011), 2010.
- [4] STMICROELECTRONICS: *Using LSM303DLH for a tilt compensated electronic compass.* , (Abgerufen am 13.05.2011), 2010.
- [5] HINES, LINDSEY: *Digital Compass Accuracy.*, University of St. Thomas , (Abgerufen am 17.05.2011), 2007.
- [6] VASCONCELOS, J.F.; ELKAIM, G.; SILVESTRE, C.; OLIVEIRA, P.; CARDEIRA, B.: *A Geometric Approach to Strapdown Magnetometer Calibration in Sensor Frame.* , (Abgerufen am 26.05.2011), 2008.
- [7] MICROCHIP TECHNOLOGY INC.: *Designing a Digital Compass Using the PIC18F2520.* , (Abgerufen am 23.05.2011), 2005.
- [8] WELLENDORPH EJSING, LOUISE: *Planar Hall sensor for influenza immunoassay.* , (Abgerufen am 19.05.2011), 2006.
- [9] N. SLOSS, ANDREW; SYMES, DOMINIC; WRIGHT, CHRIS: *ARM System Developer's Guide - Designing and Optimizing System Software.* , (Abgerufen am 20.05.2011), 2006.
- [10] XILINX INC.: *MicroBlaze Processor Reference Guide.* (Version 9.0), 2008.
- [11] XILINX INC.: *EDK 12.3 Concepts, Tools, and Techniques.* , 2010.
- [12] XILINX INC.: *Embedded system tools reference manual EDK 12.4.* , 2010.

- [13] XILINX INC.: *MicroBlaze RISC 32-Bit Soft Processor*. , 2002.
- [14] XILINX INC.: *System Generator for DSP User Guide*. , 2008.
- [15] XILINX INC.: *ML505/ML506/ML507 Evaluation Platform User Guide*. (Version 3.1.2), 2011.
- [16] XILINX INC.: *Virtex-5 FPGA User Guide* , 2010.
- [17] XILINX INC.: *Virtex-5 Family Overview* (Version 5.0), 2009.
- [18] XILINX INC.: *Virtex-5 FPGA Configuration User Guide* (Version 3.9.1), 2010.
- [19] XILINX INC.: *ML505/ML506/ML507 Getting Started Tutorial* (Version 3.0.3), 2009.
- [20] XILINX INC.: *ML505/ML506/ML507 Reference Design* (Version 3.1), 2009.
- [21] XILINX INC.: *OPB Universal Serial Bus 2.0 Device (v1.00a)* , 2007.
- [22] SUNDARARAJAN ANANTHAKRISHNAN (XILINX INC.): *Reference System: Using the OPB EPC with the Cypress CY7C67300 USB Controller* (Version 1.3), 2007.
- [23] XILINX INC.: *XPS USB Host Controller (v1.01a)* , 2009.
- [24] REICHARDT, JÜRGEN; SCHWARZ, BERND: *VHDL-Synthese Entwurf digitaler Schaltungen und Systeme*. , Oldenbourg Wissenschaftsverlag, München, 5. Auflage, 2009, ISBN 978-3-486-58987-0.
- [25] RECHENBERG, PETER: *Technisches Schreiben: (nicht nur) für Informatiker* , Carl Hanser Verlag GmbH & CO. KG; 3. Auflage, München, 2006, ISBN 978-3-446-40695-7.
- [26] KUHR, JAN: *Hardware-Software-Codesign eines Echtzeit-Audiosignalverarbeitungssystems zur räumlichen Positionierung virtueller Schallquellen über Stereokopfhörer*. , 2010.
- [27] SCHNABEL, PATRICK (ELEKTRONIK KOMPENDIUM): (<http://www.elektronik-kompodium.de/sites/bau/1503041.htm>) *MEMS - Micro-Electro-Mechanical Systems*. , (Abgerufen am 02.06.2011).
- [28] DRESCHER, ROLF (UNIVERSITÄT HEIDELBERG): *MEMS? Herstellung, Bauformen, Realisierungen*. , 2003.
- [29] MILLER, JOSEPH:: (<http://www.circuitcellar.com/library/print/0404/Miller165/f3.htm>) *First steps with the HMC5843 with Arduino: verify the accuracy of its results*. , (Abgerufen am 28.05.2011).

- [30] XILINX INC.:
(<http://www.xilinx.com/products/boards/ml507/docs.htm>)
ML507 Evaluation Platform Documentation , (Abgerufen am 02.06.2011).
- [31] ANALOG DEVICES INC.: *ADXL345 Data Sheet Rev. C* , 2011.
- [32] SPARKFUN ELECTRONICS:
(<http://www.sparkfun.com/products/10321>) *9 Degrees of Freedom - Sensor Stick* , (Abgerufen am 21.05.2011).
- [33] VARESANO, FABIO:: (<http://www.varesano.net/blog/fabio/first-steps-hmc5843-arduino-verify-accuracy-its-results>)
First steps with the HMC5843 with Arduino: verify the accuracy of its results. ,
(Abgerufen am 21.05.2011).
- [34] AYAZI, FARROKH::
(<http://www.freepatentsonline.com/7043985.html>)
High-resolution in-plane tuning fork gyroscope and methods of fabrication. , 2006
(Abgerufen am 08.06.2011).
- [35] SHARMA, AJIT; ZAMAN, FAISAL M.; AMINI, BABAK V.; AYAZI, FARROKH (GEORGIA INSTITUTE OF TECHNOLOGY): *A High-Q In-Plane SOI Tuning Fork Gyroscope* , 2004.
- [36] APOSTOLYUK, VLADISLAV (ASTRISE CORPORATION): *Theory and Design of Micromechanical Vibratory Gyroscopes* , 2006.
- [37] AUTOMOBIL ELEKTRONIK: *MEMS-Sensoren im Überblick* , 2007.
- [38] SEEGER, JOE; LIM, MARTIN; NASIRI, STEVE (INVENSENSE): *Development of high-performance, high-volume consumer mems gyroscopes* , 2010.
- [39] EPSON TOYOCOM CORPORATION: (<http://www.epsontoyocom.co.jp/english/gyroportal/about.html>)
Gyro sensors - How they work and that's ahead , (Abgerufen am 06.06.2011).
- [40] ZHAO, NEIL: *Full-Featured Pedometer Design Realized with 3-Axis Digital Accelerometer* , Analog Dialogue 44, 06.06.2010.
- [41] HONEYWELL INC.: *3-Axis Digital Compass IC HMC5843* , 2010.
- [42] INVENSENSE INC.: *ITG-3200 Product Specification Rev. 1.4* , Document Number: PS-ITG-3200A-00-01.4, 30.03.2010.

- [43] S.O.H. MADGWICK, R. VAIDYANATHAN, A.J.L. HARRISON: *An Efficient Orientation Filter for IMU and MARG Sensor Arrays* , Department of Mechanical Engineering, University of Bristol, April 2010.
- [44] XILINX INC.:
(<http://www.xilinx.com/support/answers/19592.htm>) *11.1 EDK - Why does the standard C printf() function use so much memory?* , April 2009.
- [45] XILINX INC.: *EDK 10.1 Concepts, Tools, and Techniques.* , 2008.
- [46] WIKIPEDIA: (<http://de.wikipedia.org/wiki/CORDIC>) *CORDIC.* , (Abgerufen am 15.07.2011).
- [47] JAN WEDEL: *Integrierte Navigationssysteme - Sensordatenfusion, GPS und Inertiale Navigation.* , Oldenbourg Wissenschaftsverlag, München, 1. Auflage, 2007, ISBN 978-3-486-58160-7.
- [48] WIKIPEDIA: (http://en.wikipedia.org/wiki/Sagnac_effect) *Sagnac effect.* , (Abgerufen am 26.07.2011).
- [49] THOMSON, WILLIAM, SIR; KELVIN, BARON: *Mathematical and Physical Papers.* , Cambridge University Press, Cambridge, Volume V, 1911, (Abgerufen am 26.07.2011).

Anhang A

Inhalt der CD-Rom

- Bachelorarbeit "Ein SoC-System zur Indoor-Positionsermittlung mittels Beschleunigungs-, Gyro- und Kompassensordaten" (PDF)
- Quellcode für das Hard- und Software-Design
- Excel-Sheet zur Analyse der Kompassstörungen
- Diverse Dokumente (Artikel, Manuals, Application Notes, White Papers, . . .) zum behandelten Themengebiet
- Alle in dieser Bachelorarbeit verwendeten Grafikdateien in Originalqualität.

Anhang B

Xilinx Platform Studio Schemata

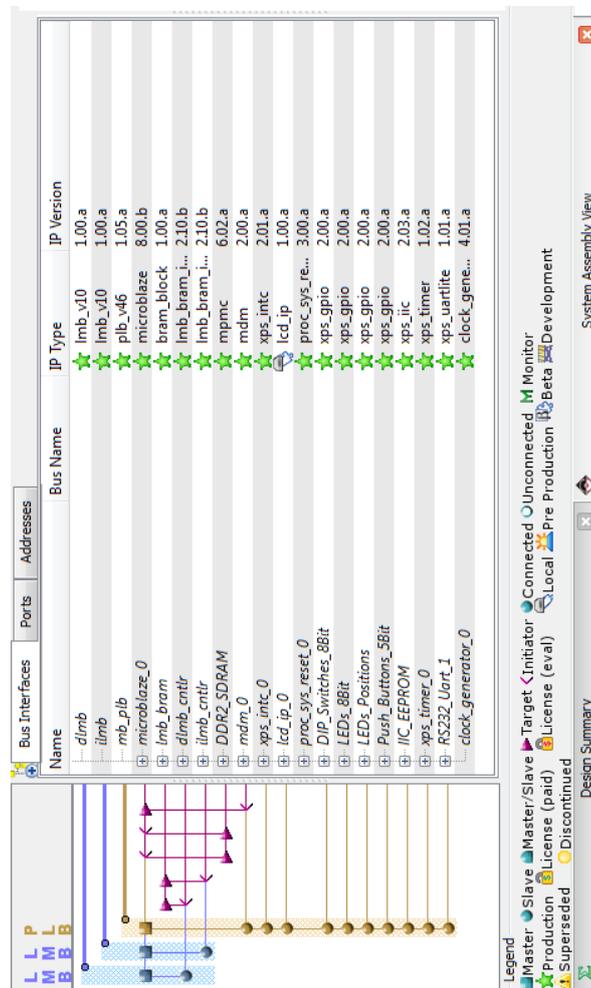


Abbildung B.1: Bus Interfaces des Embedded-Systems

Bus Interfaces		Ports	Addresses	Net	Direction	Range	Class	Frequency(Hz)	Reset Polarity	Sensitivity	IP Type
External Ports											
		dlimb									lmb_v10
		ilmb									lmb_v10
		mb_plb									plb_v46
		microblaze_0									microblaze
		lmb_bram									bram_block
		dlimb_cntlr									lmb_bram_if_cntlr
		ilmb_cntlr									lmb_bram_if_cntlr
		DDR2_SDRAM									mpmc
		mdm_0									mdm
		xps_intc_0									xps_intc
		lcd_ip_0									lcd_ip
		proc_sys_reset_0									proc_sys_reset
		DIP_Switches_8Bit									xps_gpio
		LEDs_8Bit									xps_gpio
		LEDs_Positions									xps_gpio
		Push_Buttons_5Bit									xps_gpio
		IIC_EEPROM									xps_iic
		xps_timer_0									xps_timer
		RS232_Uart_1									xps_uartlite
		clock_generator_0									clock_generator

Abbildung B.2: Ports des Embedded-Systems

Name	Net	Direction	Range	Class	Frequency(Hz)	Reset Polarity	Sensitivity	IP Type
External Ports								
dlimb								lmb_v10
mb_plb								lmb_v10
microblaze_0								microblaze
lmb_bram								bram_block
dlimb_cntlr								lmb_bram_if_cntlr
lmb_cntlr								lmb_bram_if_cntlr
DDR2_SDRAM								mpmc
mdm_0								mdm
xps_intc_0								xps_intc
lcd_ip_0								lcd_ip
proc_sys_reset_0								proc_sys_reset
DIP_Switches_8Bit								xps_gpio
LEDs_8Bit								xps_gpio
LEDs_Positions								xps_gpio
Push_Buttons_5Bit								xps_gpio
IIC_EEPROM								xps_iic
IIC2INTC_Irpt								
(BUS IF) SPLB								
(IO IF) iic_0								
Sda	No Connection	O						
Scl	Connected to BUS mb_plb	O						
Sda_I	Connected to External Ports	O						
Sda_O	Connected to External Ports	O						
Sda_T	fpga_0_IIC_EEPROM_Sda_pin	IO						
Scl_I	fpga_0_IIC_EEPROM_Scl_pin	IO						
Scl_O	No Connection	I						
Scl_T	No Connection	O						
Gpo	No Connection	O						
xps_timer_0								xps_timer
RS232_Uart_1								xps_uartlife
clock_generator_0								clock_generator

Legend

- Master/Slave
- Target
- Initiator
- Connected
- Unconnected
- Monitor
- Production
- License (paid)
- License (eval)
- Local
- Pre Production
- Beta
- Development
- Superseded
- Discontinued

Abbildung B.4: Ports des Embedded-Systems erweiterte Ansicht (Schwerpunkt: I²C)

Name	Net	Direction	Range	Class	Frequency(Hz)	Reset Polarity	Sensitivity	IP Type
External Ports								
fpga_0_RS232_Uart_1_RX_pin	fpga_0_RS232_Uart_1_RX_pin	I		NONE				
fpga_0_RS232_Uart_1_TX_pin	fpga_0_RS232_Uart_1_TX_pin	O		NONE				
fpga_0_IIC_EEPROM_Sda_pin	fpga_0_IIC_EEPROM_Sda_pin	IO		NONE				
fpga_0_IIC_EEPROM_Scl_pin	fpga_0_IIC_EEPROM_Scl_pin	IO		NONE				
fpga_0_DDR2_SDRAM_DDR2_Clk_pin	fpga_0_DDR2_SDRAM_DDR2_Clk_pin	O	[1:0]	NONE				
fpga_0_DDR2_SDRAM_DDR2_Ce_pin	fpga_0_DDR2_SDRAM_DDR2_Ce_pin	O	[1:0]	NONE				
fpga_0_DDR2_SDRAM_DDR2_Cs_n_pin	fpga_0_DDR2_SDRAM_DDR2_Cs_n_pin	O	[1:0]	NONE				
fpga_0_DDR2_SDRAM_DDR2_Odt_pin	fpga_0_DDR2_SDRAM_DDR2_Odt_pin	O		NONE				
fpga_0_DDR2_SDRAM_DDR2_Ras_n_pin	fpga_0_DDR2_SDRAM_DDR2_Ras_n_pin	O		NONE				
fpga_0_DDR2_SDRAM_DDR2_Cas_n_pin	fpga_0_DDR2_SDRAM_DDR2_Cas_n_pin	O		NONE				
fpga_0_DDR2_SDRAM_DDR2_We_n_pin	fpga_0_DDR2_SDRAM_DDR2_We_n_pin	O		NONE				
fpga_0_DDR2_SDRAM_DDR2_BankAd...	fpga_0_DDR2_SDRAM_DDR2_BankAd...	O	[1:0]	NONE				
fpga_0_DDR2_SDRAM_DDR2_Addr_pin	fpga_0_DDR2_SDRAM_DDR2_Addr_pin	O	[12:0]	NONE				
fpga_0_DDR2_SDRAM_DDR2_Dq_pin	fpga_0_DDR2_SDRAM_DDR2_Dq_pin	IO	[63:0]	NONE				
fpga_0_DDR2_SDRAM_DDR2_Dqs_pin	fpga_0_DDR2_SDRAM_DDR2_Dqs_pin	O	[7:0]	NONE				
fpga_0_DDR2_SDRAM_DDR2_Dqs_n...	fpga_0_DDR2_SDRAM_DDR2_Dqs_n...	IO	[7:0]	NONE				
fpga_0_clk_1_sys_clk_pin	dcm_clk_s	I		CLK	100000000			
fpga_0_rst_1_sys_rst_pin	sys_rst_s	I		RST		0		
lcd_ip_0_lcd_pin	lcd_ip_0_lcd	O	[0:6]	NONE				
LEDs_8Bit_GPIO_IO_0	LEDs_8Bit_GPIO_IO_0	O	[0:7]	NONE				
LEDs_Positions_GPIO_IO_0	LEDs_Positions_GPIO_IO_0	O	[0:4]	NONE				
Push_Buttns_5Bit_GPIO_IO_1	Push_Buttns_5Bit_GPIO_IO_1	I	[0:4]	NONE				
DIP_Switches_8Bit_GPIO_IO_1	DIP_Switches_8Bit_GPIO_IO_1	I	[0:7]	NONE				
lmb								lmb_v10
lmb								lmb_v10
lmb_plb								plb_v46
lmb_bram								microblaze
lmb_cntnr								bram_block
lmb_cntnr								lmb_bram_if_cntnr
lmb_cntnr								lmb_bram_if_cntnr

Abbildung B.5: Externe Ports des Embedded-Systems

Bus Interfaces		Ports		Addresses					
Instance	Base Name	Base Address	High Address	Size	Bus Interface(s)	Bus Name	Lock		
microblaze_0's Address Map									
dmb_cntlr	C_BASEADDR	0x00000000	0x0000FFFF	64K	SLMB	dmb	<input type="checkbox"/>		
ilmb_cntlr	C_BASEADDR	0x00000000	0x0000FFFF	64K	SLMB	ilmb	<input type="checkbox"/>		
DDR2_SDRAM	C_MPMC_BASE...	0x50000000	0x5FFFFFFF	256M	XCL0:XCL1	microblaze_0 IX...	<input type="checkbox"/>		
Push_Buttons_5Bit	C_BASEADDR	0x81400000	0x8140FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		
LEDs_Positions	C_BASEADDR	0x81420000	0x8142FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		
LEDs_8Bit	C_BASEADDR	0x81440000	0x8144FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		
DIP_Switches_8Bit	C_BASEADDR	0x81460000	0x8146FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		
IIC_EEPROM	C_BASEADDR	0x81600000	0x8160FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		
xps_intc_0	C_BASEADDR	0x81800000	0x8180FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		
xps_timer_0	C_BASEADDR	0x83C20000	0x83C2FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		
RS232_Uart_1	C_BASEADDR	0x84000000	0x8400FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		
mdm_0	C_BASEADDR	0x84400000	0x8440FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		
lcd_ip_0	C_BASEADDR	0xCF400000	0xCF40FFFF	64K	SPLB	mb_plb	<input type="checkbox"/>		

Legend		Design Summary	
	Slave		Master/Slave
	Production		License (paid)
	Superseded		Discontinued
	Target/Initiator		Connected
	Unconnected		Monitor
	Local		Pre Production
	Beta		Development

Abbildung B.6: Adressen des Embedded-Systems

Anhang C

9 Degrees of Freedom Sensor Stick

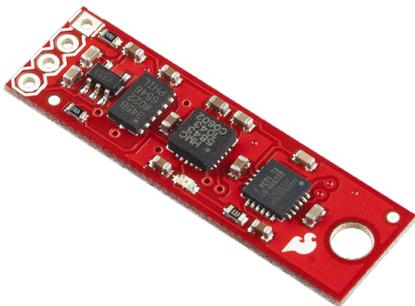


Abbildung C.1: 9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun (1) [32]



Abbildung C.2: 9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun (2) [32]

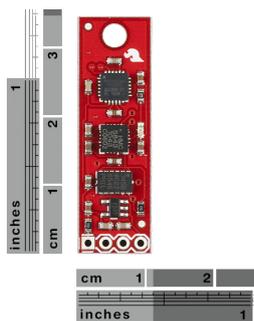


Abbildung C.3: 9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun (3) [32]



Abbildung C.4: 9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun (4) [32]

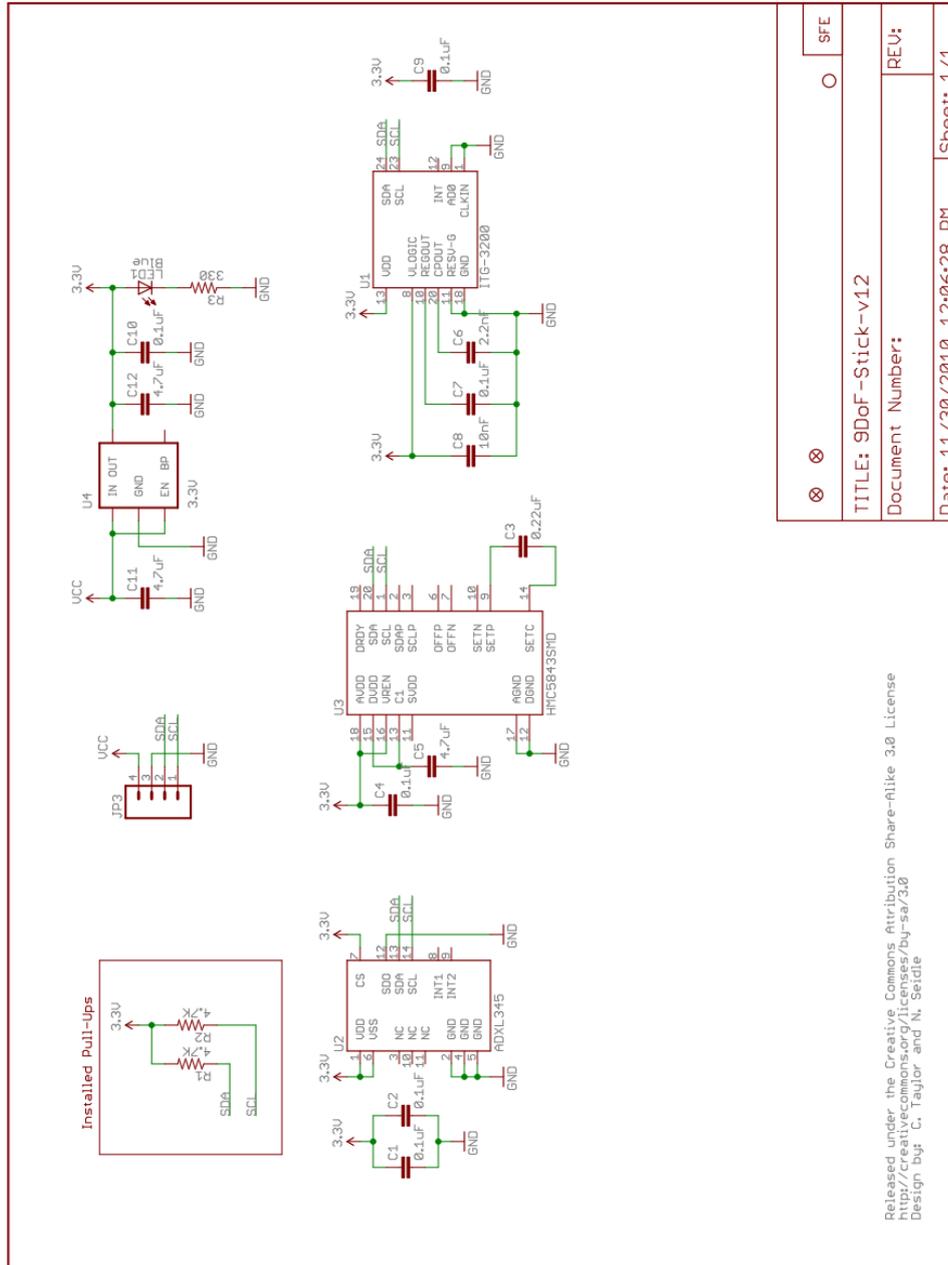


Abbildung C.5: Schematic des 9 Degrees of Freedom Sensor Stick (SEN-10321) von Sparkfun [32]

⊗	⊗	○	SFE
TITLE: 9DoF-Stick-v12			
Document Number:			
REU:			
Date: 11/30/2010 12:06:28 PM			Sheet: 1/1

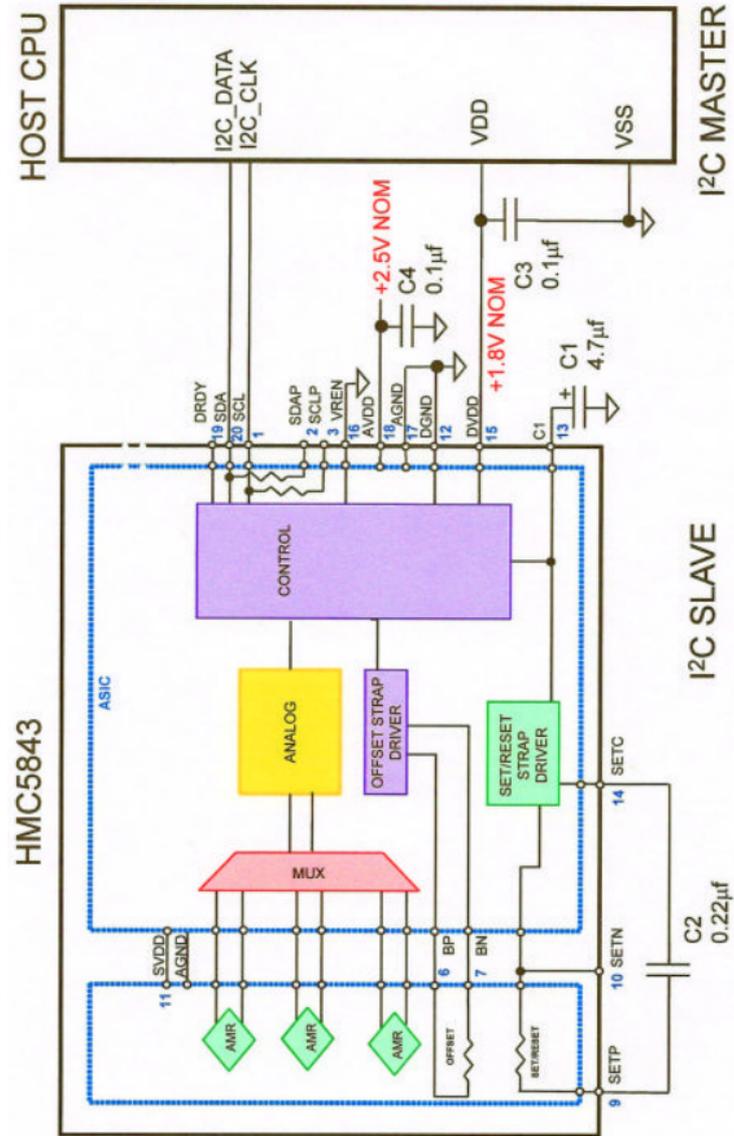


Abbildung C.6: Blockdiagramm HMC5843 [41]

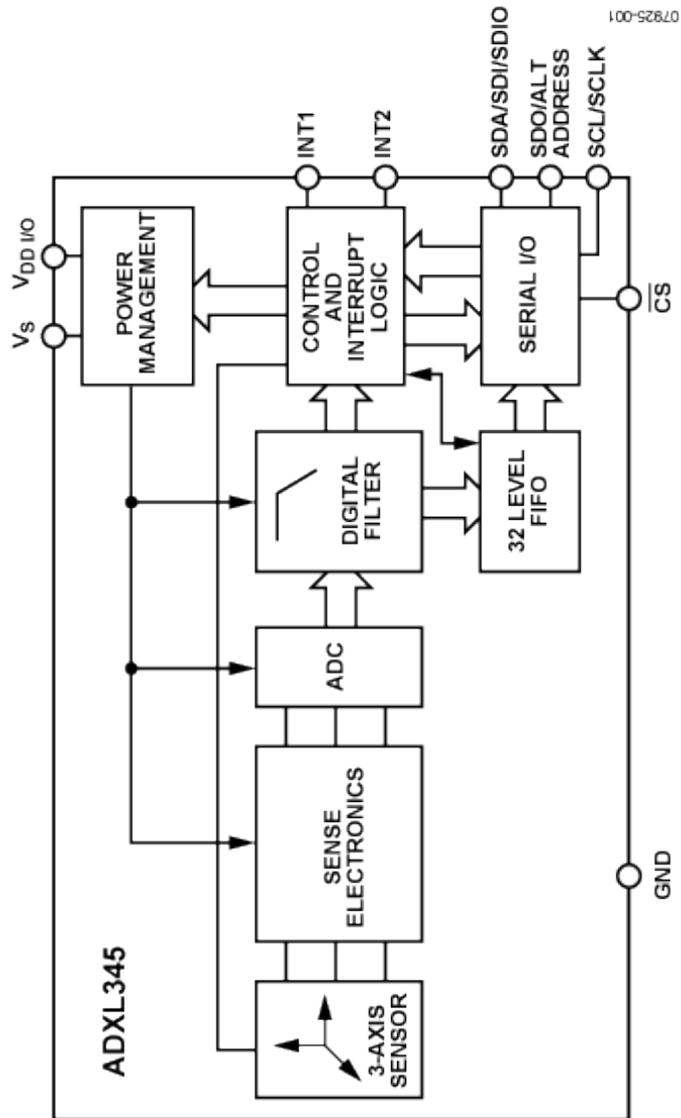


Abbildung C.7: Blockdiagramm ADXL345 [31]

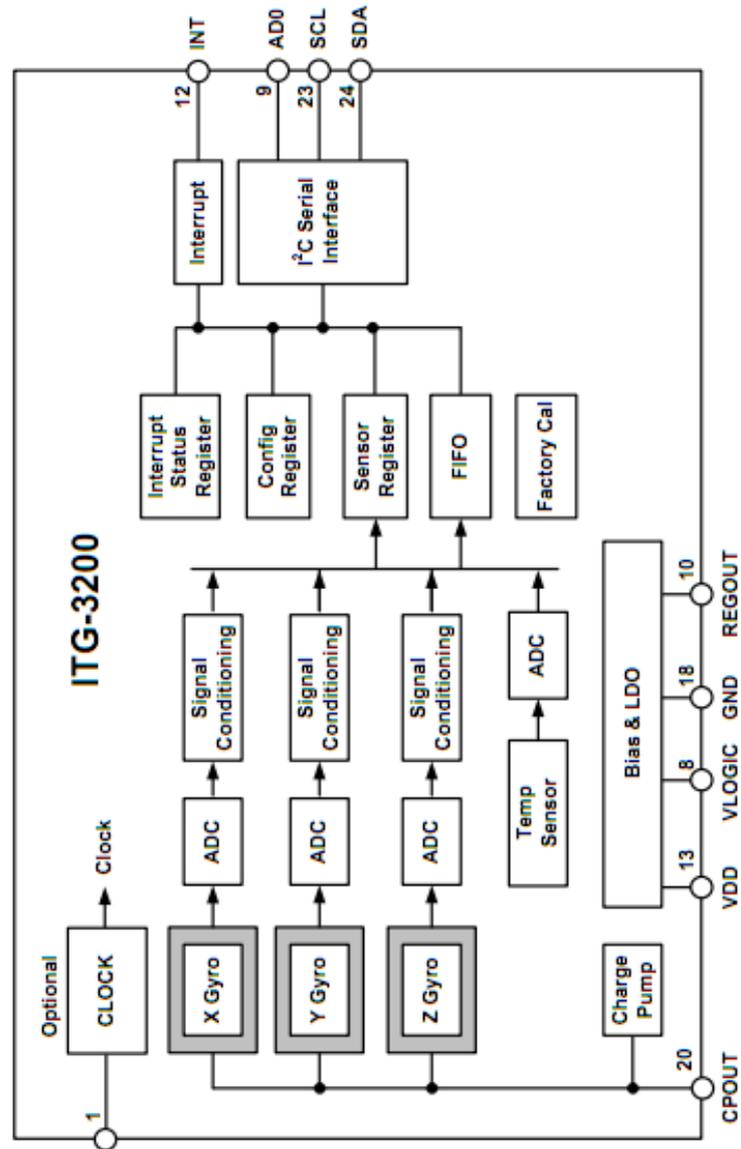


Abbildung C.8: Blockdiagramm ITG-3200 [42]

Anhang D

Embedded System

Push-Button	Funktion
Push-Button Oben	System zurücksetzen
Push-Button Rechts	Gyroskop kalibrieren
Push-Button Unten	System stoppen
Push-Button Links	Accelerometer kalibrieren
Push-Button Mitte	Kompass kalibrieren

Tabelle D.1: Push-Button Funktionsübersicht

Dip-Switch	Funktion Stellung 1	Funktion Stellung 0
Dip-Switch 1	Terminalausgabe eingeschaltet	Terminalausgabe ausgeschaltet
Dip-Switch 2	Displayausgabe eingeschaltet	Displayausgabe ausgeschaltet
Dip-Switch 3	Kompass eingeschaltet	Kompass ausgeschaltet
Dip-Switch 4	Accelerometer eingeschaltet	Accelerometer ausgeschaltet
Dip-Switch 5	Gyroskop eingeschaltet	Gyroskop ausgeschaltet
Dip-Switch 6	Roll/Pitch eingeschaltet	Roll/Pitch ausgeschaltet
Dip-Switch 7	Temperatur/Laufzeit eingeschaltet	Temperatur/Laufzeit ausgeschaltet
Dip-Switch 8	Tilt Korrektur eingeschaltet	Tilt Korrektur ausgeschaltet

Tabelle D.2: Dip-Switch Funktionsübersicht

	verfügbar	verwendet	
Number of Slice Registers	44.800	6.848	15%
Number of Slice LUTs	44.800	6.445	14%
Number of occupied Slices	11.200	3.850	34%
Number of LUT Flip Flop pairs used		9.693	
Number of bonded IOBs	640	153	23%
Number of LOCed IOBs	153	153	100%
Number of BlockRAM/FIFO	148	65	43%
Total Memory used (KB)	5.328	2.340	43%
Number of BUFG/BUFGCTRLs	32	5	15%
Number of IDELAYCTRLs	22	3	13%
Number of BSCANs	4	1	25%
Number of BUFIOs	80	8	10%
Number of DSP48Es	128	5	3%
Number of PLL_ADVs	6	1	16%

Tabelle D.3: FPGA Ressourcenverbrauch

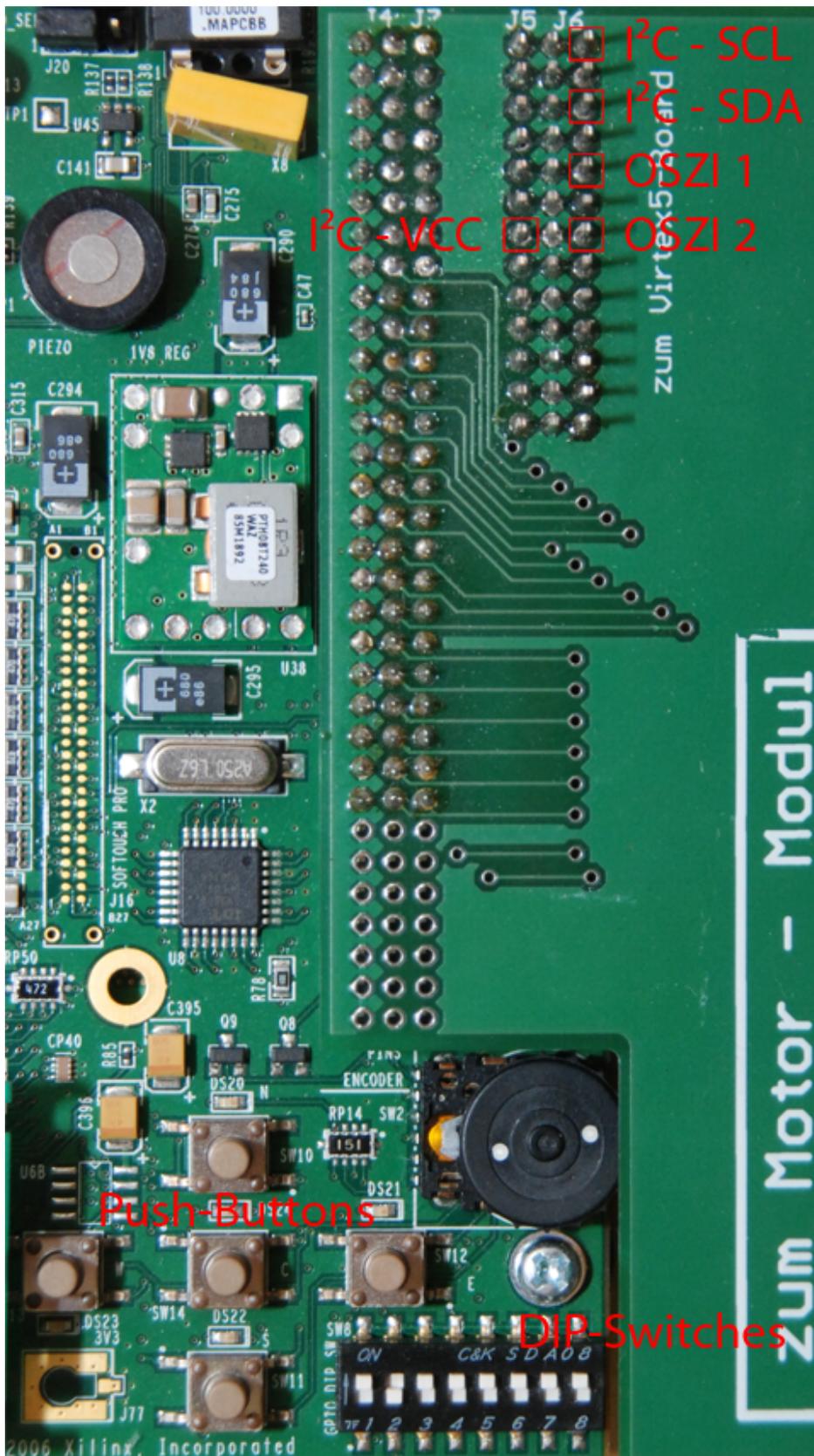


Abbildung D.1: Interfaces des Embedded-Systems

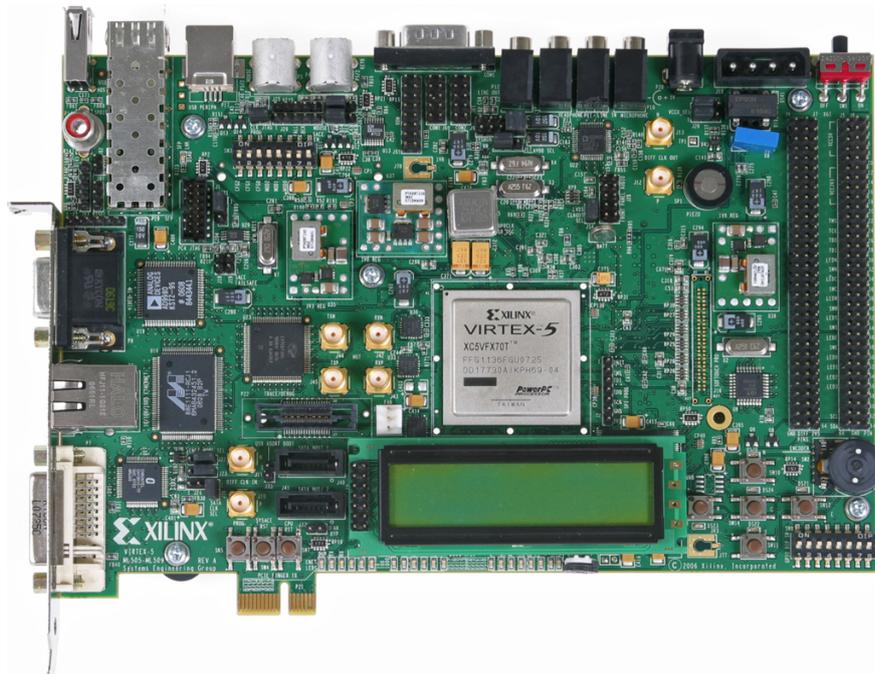


Abbildung D.2: Sicht von oben auf das ML507 Evaluationsboard [30]

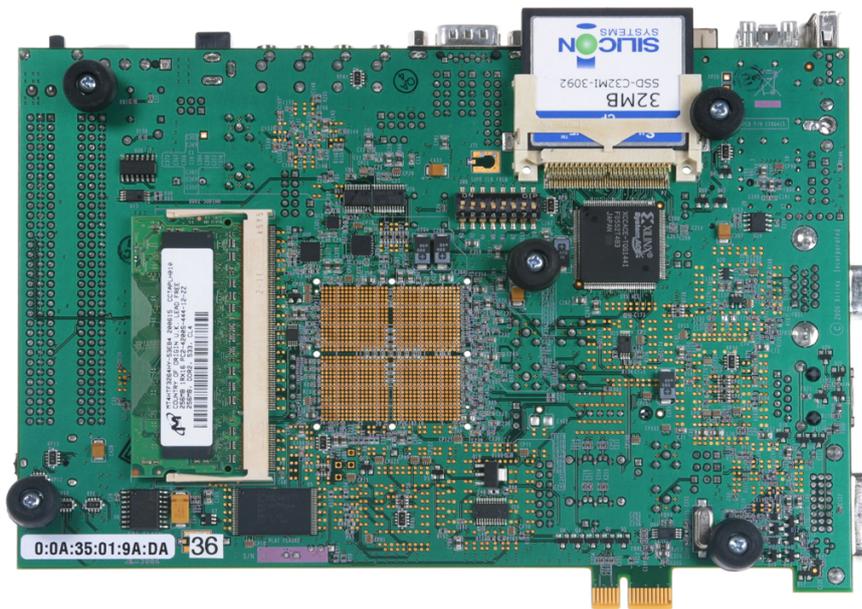


Abbildung D.3: Sicht von unten auf das ML507 Evaluationsboard [30]

Anhang E

Display- und Terminalausgaben des Systems



Abbildung E.1: Terminalausgabe des Embedded Systems



Abbildung E.2: Ausgabemodi auf das Virtex-5 LCD Displays

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §22(4) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 01. August 2011

Ort, Datum

Unterschrift