Dominik Klein, BSc

Digitale Sensorschnittstellen für automotive Anwendungen

MASTERARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

Masterstudium Elektrotechnik

eingereicht an der

Technischen Universität Graz

Betreuer

Ass.Prof. Dipl.-Ing. Dr.techn. Bernd Eichberger

Institut für Elektronische Sensorsysteme

Graz, März 2018

EIDESSTATTLICHE ERKLÄRUNG

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

_____ _____

Datum Unterschrift

STATUTORY DECLARATION

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

_____ _____

Date Signature

Danksagung

An dieser Stelle möchte ich mich bei allen Freunden und Bekannten bedanken, die mich sowohl während der Studienzeit als auch bei meiner Masterarbeit unterstützt haben. Ohne eure Motivation und Hilfe wäre das Studium um vieles schwerer gewesen.

Der größte Dank ergeht an meine gesamte Familie, vor allem an meine Eltern und Großeltern. Dank eurem Einsatz sowie der moralischen und finanziellen Unterstützung von der Schulzeit bis zum Ende des Studiums, kann ich nun stolz auf eine sehr gute Ausbildung zurückblicken. Vielen lieben Dank für alles, was ihr für mich geleistet und auf euch genommen habt. Ein besonderer Dank ergeht auch an meine Freundin Patricia, die mir in schwierigen Situationen zur Seite gestanden ist und mir immer neue Motivation zugesprochen hat.

Vielen Dank möchte ich auch an meinen Betreuer Herrn Ass.Prof. Dipl.-Ing. Dr.techn. Bernd Eichberger aussprechen. Mit Ihren nützlichen Ideen und Hilfestellungen konnten wir ein sehr spannendes und lehrreiches Projekt realisieren.

Bedanken möchte ich mich auch bei den Firmen NXP, ST Microelectronics, ELMOS, Micronas und Analog Devices für die zur Verfügung gestellten Sensoren.

Kurzfassung

Die Digitalisierung des Kraftfahrzeuges hat sich in den letzten Jahrzehnten zu einem zentralen Thema der Automobilindustrie entwickelt. Neben den zahlreichen Assistenzsystemen, die den Fahrzeuglenker im Verkehr unterstützen und einen Gewinn an Sicherheit bringen, erfahren wir täglich sowohl positive als auch leider negative Neuigkeiten über die Entwicklungen im Bereich des autonomen Fahrens. Damit die Sicherheit für die Fahrzeuginsassen und Personen außerhalb des Fahrzeugs weiter verbessert werden kann, ist eine zuverlässige Datenübertragung für die Messdaten der Sensorsysteme erforderlich. Existierende digitale Bussysteme wie CAN (Controller Area Network), FlexRay oder LIN (Local Interconnect Network) sind hier kostenmäßig nicht optimal oder haben zu geringe Datenraten. Mit den neu entwickelten Bussystemen SENT (Single Edge Nibble Transmission) und PSI5 (Peripheral Sensor Interface 5) versucht man diese Defizite zu vermeiden.

Diese Arbeit soll die wichtigsten Grundlagen über die Funktionsweise der SENT- und PSI5-Schnittstellen vermitteln. Das theoretisch erlangte Wissen kann an Hand der selbst entwickelten Hard- und Software gefestigt werden. Zur Unterstützung stehen mehrere Simulationsbeispiele sowie am Markt erhältliche Sensoren zur Verfügung. Die Erkenntnisse und Ergebnisse der getesteten Sensoren werden ebenso wie die Umsetzung der Hard- und Software in dieser Masterarbeit dokumentiert, um eine Weiterentwicklung und Verbesserung zu ermöglichen.

Abstract

The digitization of the motor vehicle turned in recent decades into a central theme of the automotive industry. Many driver assistance systems already support the driver operating the car and result in a considerable increase in safety on the road. We receive mostly positive news about the processing of autonomous driving. To improve the safety of passengers and people outside the vehicle a reliable transmission of data from sensor systems is required. Existing transmission technologies such as CAN (Controller Area Network), FlexRay or LIN (Local Interconnect Network) either suffer from high development costs or have too low data rates. Recently developed bus systems like SENT (Single Edge Nibble Transmission) and PSI5 (Peripheral Sensor Interface 5) try to bring improvements in these areas.

This thesis describes the most important basics of the SENT- and PSI5-Interfaces. The theoretical knowledge can be consolidated by a self-developed hardware and software. Several simulation examples and sensors, which are available on the market, are provided with this master thesis. The perception and results of the tested sensors are also documented in this thesis, as well as the implementation of the hardware and software to offer future developments and improvements.

Inhaltsverzeichnis

1. Einleitung 1

1.1. Allgemein 1

1.2. Ziel der Masterarbeit 2

2. Grundlagen 3

2.1. EMV im KFZ 3

2.1.1. Kopplungsarten 3

2.1.2. EMV-Normen (KFZ) 3

2.1.3. Funkstörungen 4

2.1.3.1 Einstrahlung von Störungen 4

2.1.3.2 Ausstrahlung von Störungen 5

2.1.4. Leitungsgeführte Störgrößen 5

2.1.4.1 ISO 7637-2 5

2.1.4.2 ISO 16750 10

2.1.5. Elektrostatische Entladungen 17

2.1.6. EMV-Maßnahmen 18

2.1.6.1 Allgemeines, Filter und Schutzmaßnahmen 18

2.1.6.2 Layout-Maßnahmen 19

2.2. Sensorprotokolle 23

2.2.1. SENT – Single Edge Nibble Transmission 23

2.2.1.1 SENT - Protokoll 23

2.2.1.2 Serieller Datenkanal - „Serial Messages“ 25

2.2.1.3 Physikalische Ebene 28

2.2.1.4 Applikationsbeispiele 29

2.2.2. PSI5 – Peripheral Sensor Interface 5 32

2.2.2.1 Base Standard – Übertragungsprotokoll 33

2.2.2.2 Base Standard - Betriebsarten 35

2.2.2.3 Base Standard - Physikalische Ebene 38

2.2.2.4 Base Standard – Anwendungsebene 43

2.2.2.5 Substandard – Airbag 46

2.2.2.6 Substandard – Chassis und Safety Control 50

2.2.2.7 Substandard - Powertrain 52

3. Hardware 55
3.1. Discovery-Board 56
3.2. Spannungsversorgung 57
3.3. USB, RS232, CAN 58
3.4. SENT – Schnittstelle 58
3.5. PSI5 – Schnittstelle 59
3.6. PSI5-Transceiver - E521.40 61
3.7. SENT / PSI5 – Digital-Analog-Converter 62
4. Software 63
4.1. Mikrocontroller 63
4.1.1. Allgemeiner Funktionsablauf 64
4.1.2. Timer 66
4.1.3. Interrupts 67
4.1.4. SENT 68
4.1.4.1 SENT – Übermittlung 70
4.1.4.2 SENT - Empfang 71
4.1.5. PSI5 72
4.1.5.1 PSI5 – Übermittlung 74
4.1.5.2 PSI5 – Empfang 80
4.1.5.3 PSI5-Transceiver - E521.40 82
4.1.6. USB / RS232 / CAN 88
4.1.6.1 Schnittstellenkonfiguration 89
4.1.6.2 Schnittstellenkommunikation 90
4.1.7. DAC – Digital-Analog-Converter 101
4.2. PC-Software 102
4.2.1. Allgemeines & Einstellungen 103
4.2.2. SENT 105
4.2.3. PSI5 109
5. Beispiele für SENT-/PSI5-Signale 113
5.1. SENT 113
5.1.1. Beispiele für SENT-Übertragungen 113
5.1.2. Sensorbeispiel - HAL 2830 / HAL 2833 116

5.1.3. Sensorbeispiel – KMA215 120
5.1.4. Sensorbeispiel – TLE4998S3 122
5.2. PSI5 127
5.2.1. Beispiele für PSI5-Übertragungen 127
5.2.2. Sensorbeispiel - ADXL151 133
5.2.3. Sensorbeispiel - AIS1200PS 134
6. Zusammenfassung 136
7. Literaturverzeichnis 138
8. Abkürzungsverzeichnis 141
9. Abbildungsverzeichnis 142
10. Tabellenverzeichnis 145
11. Anhang 147
11.1. µVision 147
11.2. EAGLE - Bauteilliste 156
11.3. EAGLE - Schaltpläne 162

1. Einleitung

1.1. Allgemein

Die Entwicklung des Automobils hat eine lange Geschichte hinter sich. Von den ersten dampfgetriebenen Fahrzeugen im 18. Jahrhundert, über den ersten modernen Motorwagen von Carl Benz im Jahr 1886, bis zu den heutigen Elektroautos. In den letzten Jahrzehnten spielte vor allem die Digitalisierung in der Automobilindustrie eine sehr große Rolle. Durch den Fortschritt der Technik eröffnen sich auch für die Fahrzeuge immer mehr Möglichkeiten. Nicht nur effiziente Verbrennungsmotoren mit geringem Schadstoffausstoß oder Elektroautos mit großer Reichweite sind zu wichtigen Aspekten geworden, sondern auch die Sicherheit der Fahrzeuge steht an der obersten Stelle. Die Anzahl der Assistenzsysteme steigt von Jahr zu Jahr stärker an. Beispiele sind Antiblockiersysteme, Parkassistenten, Spurhaltesysteme, Bremsassistenten und noch viele mehr. In den letzten Jahren sind außerdem einige Unternehmen, wie beispielsweise Google oder Tesla, an der Entwicklung von autonomen Fahrzeugen bestrebt. [1]

All diese Entwicklungen haben gemeinsam, dass sie zuverlässige Sensorsysteme benötigen, damit die einzelnen Steuergeräte das Fahrzeug situationsgerecht steuern können. Nicht nur die Messungen selbst müssen zuverlässig funktionieren, sondern auch die Übermittlung der gewonnenen Daten an das jeweilige Steuergerät ist ein wichtiger Faktor. Bisherige analoge Übertragungstechniken sind kostengünstig und einfach zu realisieren. Allerdings haben diese auch große Nachteile, wie die Störungsempfindlichkeit, geringe Auflösungen oder Umwandlungsfehler bei der Analog-Ditigal-Wandlung. Am Markt sind aber bereits seit vielen Jahren digitale Übertragungssysteme wie CAN (Controller Area Network) oder FlexRay im Einsatz. Diese Systeme bieten wesentliche Vorteile wie hohe Datenraten, geringe Störanfälligkeiten sowie eine Priorisierung der übertragenen der Daten. CAN und FlexRay verursachen aber gerade für einzelne kleinere Sensoren verhältnismäßig große Entwicklungskosten. Kostengünstige Bussysteme wie der LIN-Bus (Local Interconnect Network) besitzen wiederum nur geringe Datenraten, die in einigen Bereichen nicht ausreichend sind.

Aus diesen Gründen haben sich vor mittlerweile mehr als 10 Jahren in der Automobilbranche mehrere Konsortien gebildet, um für kleinere Sensorsysteme neue kostengünstige

Schnittstellen mit verbesserten Datenraten zu entwickeln. Die wichtigsten standardisierten Entwicklungen sind die SENT- (Single Edge Nibble Transmission) und PSI5-Schnittstelle (Peripheral Sensor Interface 5).

1.2. Ziel der Masterarbeit

Diese Masterarbeit soll die wichtigsten Grundlagen zur Funktionsweise der Schnittstellen SENT und PSI5 vermitteln. Auf Basis der erarbeiteten Informationen soll anschließend eine geeignete Hard- und Software (Testumgebung) entwickelt werden, damit potentielle Anwender durch Simulationsbeispiele das theoretisch erlangte Wissen erproben und festigen können. Zusätzlich sollen mehrere am Markt erhältliche Sensoren evaluiert werden. Die erlangten Erkenntnisse und Ergebnisse werden für eine spätere Weiterentwicklung bzw. Verbesserung dokumentiert.

Mögliche Einsatzgebiete der Testumgebung sind beispielsweise als Basis zur Entwicklung eigener Systeme, oder im Rahmen von Laborübungen an der Technischen Universität Graz.

2. Grundlagen

2.1. EMV im KFZ

In modernen Fahrzeugen wird heutzutage eine Vielzahl an elektronischen Systemen integriert, dies reicht von klassischen Geräten wie Radio, Funkschlüssel, Navigationssystem, bis hin zu elektronischen Anzeigen. Alle diese Systeme und Geräte können elektromagnetische Störungen produzieren bzw. auch für solche empfänglich sein. Nicht nur die elektronischen Bauteile innerhalb und außerhalb eines Fahrzeuges können elektromagnetische Störungen verursachen, sondern auch natürliche Phänomene wie Gewitter können ein Fehlverhalten der Elektronik bewirken. Aus diesem Grund ist die elektromagnetische Verträglichkeit von großer Bedeutung. [2]

2.1.1. Kopplungsarten

In der EMV bezeichnet man ein Gerät oder System welches Störungen verursacht als Störquelle und Geräte oder Systeme welche die Störungen empfangen als Störsenke. Die Störungen werden dabei über einen Kopplungspfad übertragen. Im Wesentlichen wird zwischen einer Feldkopplung und einer leitungsgeführten Kopplung unterscheiden. Allerdings können auch beide Kopplungsarten in gemischter Form sowie auch Störungen aufgrund von elektrostatischen Entladungen auftreten. [2]

Die Feldkopplung lässt sich wiederum in eine kapazitive Kopplung (aufgrund eines elektrischen Feldes zwischen zwei Leitern), eine induktive Kopplung (durch magnetische Felder zwischen den Leitern) und eine elektromagnetische Kopplung (elektromagnetischen Wellen) unterteilen. [2]

Bei der leitungsgeführten Störkopplung kann die Ursache beispielsweise eine galvanische Kopplung (bzw. Impedanzkopplung) sein. Störquellen können dabei über gemeinsame Leiterstücke Störungen einspeisen und somit andere Geräte beeinflussen. [2]

2.1.2. EMV-Normen (KFZ)

In der Europäischen Union (bzw. EWG) wurden bereits in den Jahren 1970 und 1972 mit den Richtlinien 70/156/EWG und 72/245/EWG erste Regelungen zur Funkentstörung geschaffen. Im Jahr 1995 wurden beide Richtlinien durch 95/54/EG aktualisiert. Eine umfassende Anpassung der KFZ-EMV-Richtlinien erfolgte mit der Richtlinie 2004/104/EG, welche ebenfalls nochmals in den Jahren 2005 (2005/83/EG und 2005/49/EG), 2006 (2006/28/EG) sowie 2009

(2009/19/EG) ergänzt wurde. Bauteile die nach EU-Richtlinien geprüft wurden, erhalten als Kennzeichnung ein „e“ (Kleinbuchstabe) in einem rechteckigen Logo. Allerdings können im Fahrzeug auch noch andere EU-Richtlinien oder nationale Gesetze zur Anwendung kommen. Erwähnenswert sind die ECE-Richtlinie R10 der „Economic Commission for Europe“ (Kennzeichnung mit einem „E“ als Großbuchstabe in einem runden Logo) und der Standard SAE J1113 für den amerikanischen Wirtschaftsraum.[2]

Mittlerweile erfolgt die Normung zu großen Teilen auf internationaler Ebene, in welche oftmals Teile von nationalen Normen eingeflossen sind.

2.1.3. Funkstörungen

2.1.3.1 Einstrahlung von Störungen

Die EU-Richtlinie 2004/104/EG überlässt dem Hersteller die Wahl, ob er das Gesamtfahrzeug nach der ISO 11451 auf EMV prüft oder ob er die elektromagnetische Verträglichkeit der einzelnen elektronischen/elektrischen Systeme nach ISO 11452 genehmigen lässt. [2] [3]

Bei der Prüfung nach ISO 11451-2 wird in einer Absorberhalle mit Rollenprüfstand bei einer Geschwindigkeit von 50 km/h das Verhalten des Fahrzeuges durch von außen eingestrahlte Felder überprüft. Für den Test wird eine Sendeantenne auf das Fahrzeug gerichtet, wobei die Absorber an den Wänden Reflexionen verhindern und somit eine ausschließliche Bestrahlung durch die Antenne garantieren. Für die ISO 11451-3 wird anstelle einer externen Sendeantenne die Fahrzeugantenne verwendet. Die ISO 11451-4 beschreibt das Verfahren zur „Bulk Current Injection“ (BCI abgekürzt), bei der anstelle von Feldern Störströme mit Hilfe von Stromzangen in die Kabelbäume eingekoppelt werden. Die „Bulk Current Injection“ lässt sich bis ca. 400 MHz einsetzen und zeichnet sich durch geringe Kosten aus. Sowohl Teil 3 als auch Teil 4 der ISO 11451 sind nicht in der EU-Richtlinie als Prüfmethode vorgesehen. [2] [3]

Für die Überprüfung der EMV der elektronischen Einzelsysteme kann der Hersteller nach der EU-Richtlinie zwischen einer Absorberkammer, TEM-Zelle (transversale elektromagnetische Zelle), Stromeinspeisung oder Streifenleitung in beliebigen Kombinationen wählen, allerdings muss dabei der gesamte Frequenzbereich abgedeckt werden. Für die Prüfsignale werden amplitudenmodulierte Signale (1kHz Modulation, 80% Modulationsgrad) in einem Frequenzbereich von 20 bis 800 MHz und pulsmodulierte Signale (t_{ein} 577µs, Periodendauer 4600µs) im 800 – 2000 MHz Frequenzbereich eingesetzt. [3]

2.1.3.2 Ausstrahlung von Störungen

Die europäische Norm EN 55012 (z.B. ÖVE/ÖNORM EN 55012) definiert die Grenzwerte und Messverfahren für die elektromagnetische Störaussendung durch Fahrzeuge, Boote und andere Geräte, welche durch einen Verbrennungsmotor angetriebenen werden. Die Grenzwerte sind jeweils für den Zustand des laufenden und ausgeschalteten Motors festgelegt, um Empfänger außerhalb des Testobjektes im Frequenzbereich von 30 MHz bis 1 GHz vor Funkstörungen zu schützen. Die Norm beschreibt außerdem die Anforderungen (Antennenarten, Umgebungsbedingungen, etc.) und den Ablauf der Messung. [4]

In der Norm EN 55025 wird hingegen der Schutz von Empfängern (Radio, GPS, Bluetooth, etc.) innerhalb von Fahrzeugen, Booten und anderen Geräten mit Verbrennungsmotoren behandelt. Diese Norm beschreibt Messverfahren und Grenzwerte für elektromagnetische Störaussendungen durch Komponenten, Geräte und Systeme innerhalb des Fahrzeuges. In der Norm ist ein Frequenzbereich von 150 kHz bis 2,5 GHz für die Messungen vorgesehen. Bei den Grenzwerten handelt es sich allerdings nur um Empfehlungen, welche durch entsprechende Vereinbarungen zwischen Gerätehersteller und Fahrzeughersteller abgeändert werden können. Hochfrequente Störungen sowie transiente Spannungsschwankungen auf Steuerungen bzw. Regelungen werden von dieser Norm nicht abgedeckt. [5]

2.1.4. Leitungsgeführte Störgrößen

Leitungsgeführte Störgrößen wurden in der (mittlerweile zurückgezogenen) DIN 40839 durch sechs häufig auftretende Störmuster im 12V- und 24V-Bordnetz beschrieben [2]. Diese Inhalte bildeten die Grundlagen für die ISO 7637-2 (Teil 2 der Norm) und die ISO 16750-2 [6].

2.1.4.1 ISO 7637-2

Die ISO 7637-2 beschreibt und definiert die charakteristischen Eigenschaften der am häufigsten auftretenden transienten Störgrößen im 12V- und 24V-Bordnetz. Die Norm beinhaltet Schaltungen und Impulsverläufe um ein elektronisches Gerät auf seine Funktionstüchtigkeit überprüfen zu können. Alle in diesem Kapitel verwendeten Informationen entstammen der ISO 7637-2 [6]. Anderslautende Informationsquellen werden an der entsprechenden Stelle angeführt.

Testimpuls 1

Dieser Impuls simuliert das Abschalten der Versorgungsspannung einer Induktivität. Durch diesen Impuls können elektronische Systeme parallel zur Induktivität beeinflusst werden. *Abbildung 2-1* zeigt die Schaltung und den Verlauf des Testimpulses zur Nachbildung der Störung. In der *Tabelle 2-1* sind die charakteristischen Parameter für 12V- und 24V-Systeme angeführt.

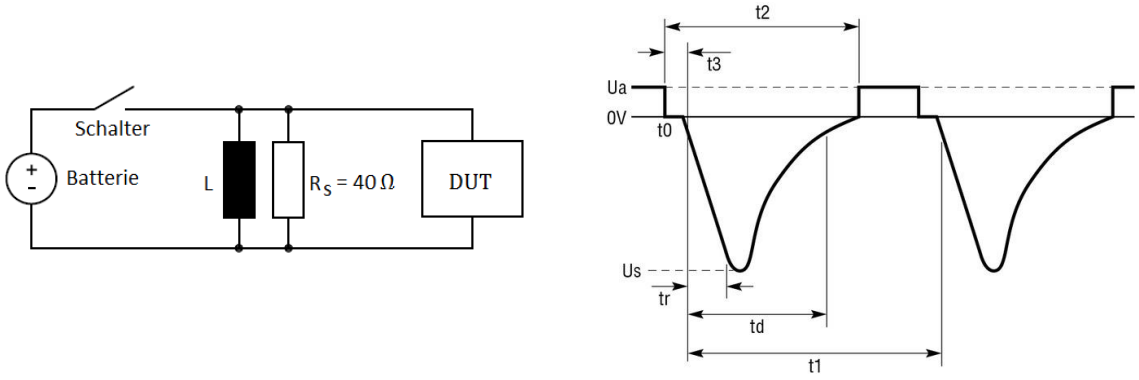

Abbildung 2-1: Prinzipschaltbild [6] ***und Verlauf*** [7] ***- Testimpuls 1***

Parameter	12V System	24V System
U_S	-75V bis -150V	-300V bis -600V
R_i	10Ω	50Ω
t_d	2ms	1ms
t_r	0,5µs bis 1µs	1,5µs bis 3µs
t_1	> 0,5s	
t_2	200ms	
t_3	< 100µs	

Tabelle 2-1: Parametertabelle – Testimpuls 1 [6]

Testimpuls 2a und 2b

Der Testimpuls 2a in *Abbildung 2-2* beschreibt die entstehenden Transienten einer seriellen Induktivität (z.B. Kabelbaum), wenn der Strom eines parallelgeschalteten Gerätes (z.B. elektronische Last) unterbrochen wird. Für den Testimpuls 2b wird der Zündschalter abgeschaltet, wodurch der Gleichstrommotor nachläuft und als Generator agiert, was zu Störgrößen am Testobjekt führt. *Tabelle 2-2* und *Tabelle 2-3* zeigen jeweils die Parameter für den Testimpuls 2a und 2b.

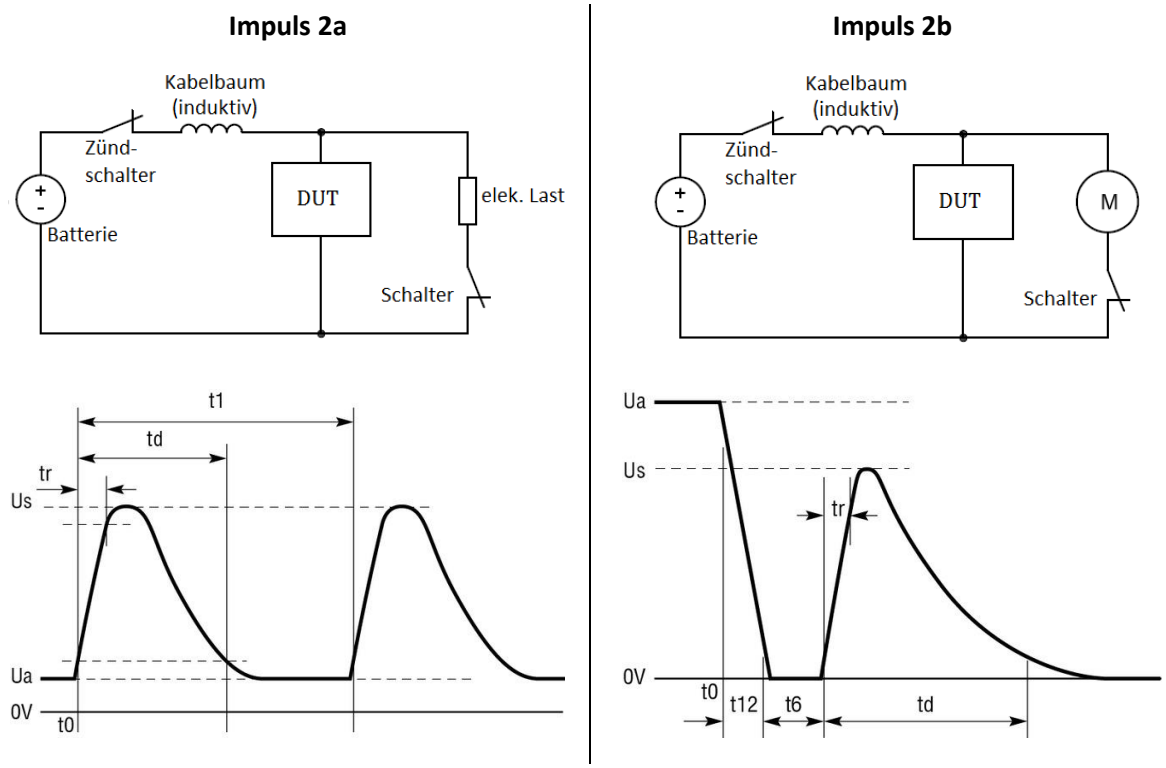

Abbildung 2-2: Prinzipschaltbild [6] ***und Impulsform*** [7] ***– Testimpuls 2a (links) und 2b (rechts)***

Parameter	12V und 24V System
U_S	+37V bis +112V
R_i	2Ω
t_d	0,05ms
t_r	0,05 µs bis 1µs
t_1	0,2s bis 5s

Tabelle 2-2: Parameter – Testimpuls 2a [6]

Parameter	12V System	24V System
U_S	10 V	20 V
R_i	0Ω bis 0,05Ω	
t_d	0,2s bis 2s	
t_{12}	1ms ± 0,5ms	
t_r	1ms ± 0,5ms	
t_6	1ms ± 0,5ms	

Tabelle 2-3: Parameter -Testimpuls 2b [6]

Testimpuls 3a und 3b

Die Impulse 3a und 3b entstehen durch Schaltvorgänge im System. Die Impulse werden durch verteilte Induktivitäten und Kapazitäten im Kabelbaum beeinflusst. Das Prinzipschaltbild zur Erzeugung dieser Störgrößen ist in *Abbildung 2-3* dargestellt.

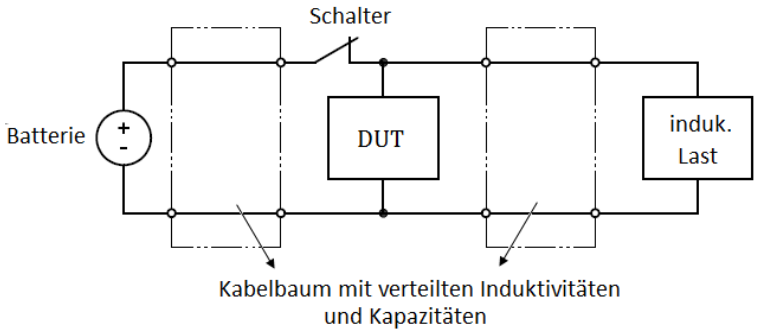

Abbildung 2-3: Prinzipschaltbild - Testimpuls 3a und 3b [6]

Abbildung 2-4 zeigt die Impulsform für die Testimpulse 3a und 3b. Der Impuls 3b entspricht betragsmäßig dem Impuls 3a, besitzt aber für die Spannung U_S ein umgekehrtes Vorzeichen (positive Flanken).

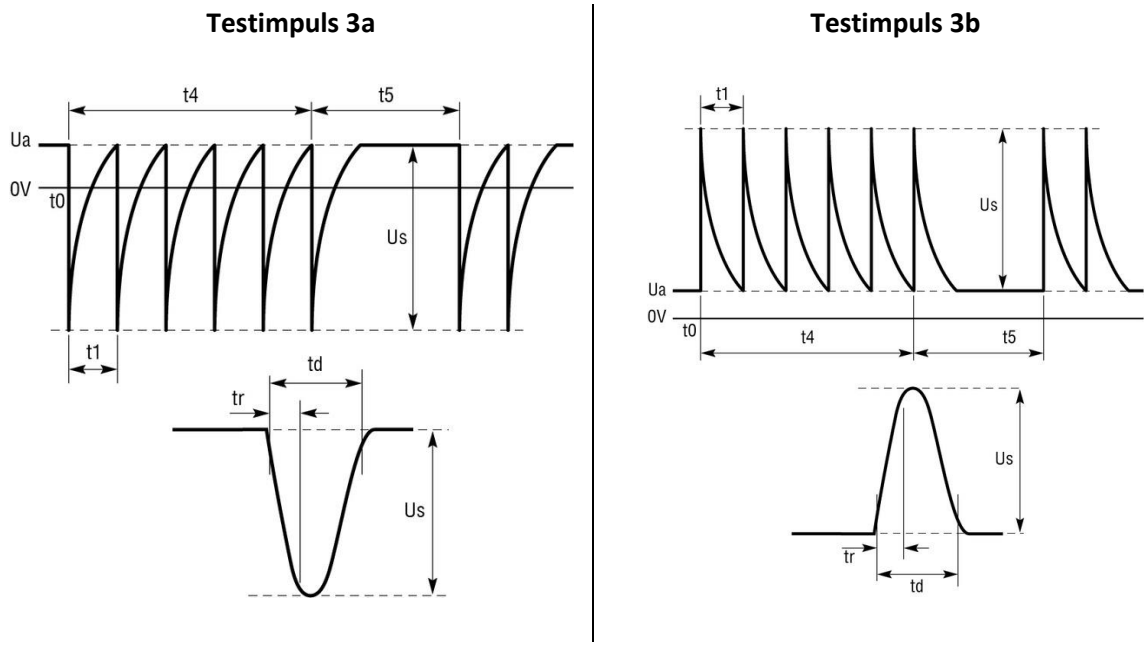

Abbildung 2-4: Impulsform - Testimpuls 3a (links) und 3b (rechts) [7]

Tabelle 2-4 zeigt die Parameter für den Testimpuls 3a. Die Parameter für den Testimpuls 3b werden durch Vertauschen des Vorzeichens von U_S (+112V bis +220V im 12V-System und +150 bis +300V im 24V-System) erhalten, alle anderen Parameter sind identisch.

Parameter	12V System	24V System
U_S	-112V bis -220V	-150V bis -300V
R_i	50Ω	
t_d	150ns ± 45ns	
t_r	5ns ± 1,5ns	
t_1	100µs	
t_4	10ms	
t_5	90ms	

Tabelle 2-4: Parameter – Testimpuls 3a [6]

Tabelle 2-5 und *Tabelle 2-6* zeigen eine Übersicht der verschiedenen Testimpulse mit dem jeweiligen Prüfgrad für die Spannung U_s, der Anzahl der Impulse und wie oft die Impulse wiederholt werden müssen. Der zu verwendende Prüfgrad und die einzustellende Testzeit wird zwischen dem Gerätehersteller und Fahrzeughersteller vereinbart.

Testimpuls	Testimpuls Prüfgrad U_s in V			Min. Anzahl an Impulsen oder Testzeit	„Burst“ Zyklus / Impuls Wiederholungszeit	
	IV	III	I / II		min.	max.
1	-150	-112	-75	500 Impulse	0,5s	> 0,5s
2a	+112	+55	+37	500 Impulse	0,2s	5s
2b	+10	+10	+10	10 Impulse	0,5s	5s
3a	-220	-165	-112	1 Stunde	90ms	100ms
3b	+150	+112	+75	1 Stunde	90m	100ms

Tabelle 2-5: Prüfgrade für die Testimpulse im 12V-System [6]

Testimpuls	Testimpuls Prüfgrad U_s in V			Min. Anzahl an Impulsen oder Testzeit	„Burst“ Zyklus / Impuls Wiederholungszeit	
	IV	III	I / II		min.	max.
1	-600	-450	-300	500 Impulse	0,5s	> 0,5s
2a	+112	+55	+37	500 Impulse	0,2s	5s
2b	+20	+20	+20	10 Impulse	0,5s	5s
3a	-300	-220	-150	1 Stunde	90ms	100ms
3b	+300	+220	+150	1 Stunde	90m	100ms

Tabelle 2-6: Prüfgrade für die Testimpulse im 24V-System [6]

2.1.4.2 ISO 16750

Die ISO 16750 beschreibt verschiedene Testbedingungen (elektrische, mechanische, chemische, etc.) unter denen elektronische Geräte und Systeme im KFZ funktionieren müssen. Es sind fünf Funktionszustände definiert, die festlegen wie stark ein elektronisches Gerät bzw. System durch eine Störgröße beeinflusst werden darf. [8]

- **Funktionszustand A:** Das Gerät/System erfüllt während und nach dem Test alle Funktionen wie vorgegeben.

- **Funktionszustand B:** Das Gerät/System erfüllt während dem Test alle Funktionen wie vorgegeben. Eine oder mehrere Funktionen können aber außerhalb der Toleranz liegen. Nach dem Test müssen wieder alle Funktionen innerhalb der zulässigen Grenzen arbeiten. Speicherfunktionen müssen im Zustand A bleiben.

- **Funktionszustand C:** Das Gerät/System erfüllt während dem Test eine oder mehrere Funktionen nicht wie vorgegeben. Nach dem Test müssen wieder alle Funktionen in den vorgesehenen Betriebszustand zurückkehren.

- **Funktionszustand D:** Das Gerät/System erfüllt während dem Test eine oder mehrere Funktionen nicht wie vorgegeben und kehrt nach dem Test solange nicht mehr in den ursprünglichen Funktionszustand zurück, bis es durch einen einfachen Eingriff zurückgesetzt wurde.

- **Funktionszustand E:** Das Gerät/System erfüllt während dem Test eine oder mehrere Funktionen nicht wie vorgegeben und kann nach dem Test nur durch eine Reparatur oder einem Austausch alle Funktionen wieder korrekt ausführen.

In der ISO 16750-2 sind eine Reihe von Tests für elektrische Beanspruchungen vorgegeben. Damit wird die korrekte Funktionsweise der elektronischen Geräte und Systeme unter den vorgegebenen Bedingungen sichergestellt. Im Folgenden werden die einzelnen Tests mit ihren wichtigsten Eckdaten für 12V- und 24V-Systeme vorgestellt [9]:

Versorgungsspannung

Das Equipment muss die vorgegebene minimale und maximale Versorgungsspannung mit Funktionszustand A erfüllen.

Überspannung

Test bei T_{max} – 20°C: Simuliert die Situation einer zu hohen Ausgangsspannung des Generators bei 20°C unter der maximalen Temperatur. Getestet wird 60 Minuten lang mit 18V bei 12V-Systemen und mit 36V bei 24V-Systemen. Der Funktionszustand muss mindestens Zustand C betragen, bei strengeren Anforderungen Zustand A.

Test bei Raumtemperatur: Simuliert eine Starthilfe (jump start) durch das Anlegen von 24V für (60 ± 6)s. Der Funktionszustand muss mindestens D betragen, bei strengeren Anforderungen B.

Überlagerte Wechselspannung

Simuliert eine überlagerte Wechselspannung an den Gleichspannungseingängen und legt den Funktionszustand A für den Test fest.

Langsame Abnahme und Zunahme der Versorgungsspannung

Simuliert die sukzessive Entladung und Aufladung der Batterie. Der Test erfolgt durch Reduzierung der minimalen Versorgungspannung auf 0V und einer anschließenden Erhöhung von 0V auf die minimale Versorgungsspannung. Der Funktionszustand soll dabei mindestens D betragen, bei strengeren Anforderungen C.

Spannungseinbrüche

Kurzzeitige Einbrüche der Versorgungsspannung

Beschreibt den Effekt, wenn eine Sicherung in einer anderen Schaltung schmilzt. *Abbildung 2-5* zeigt die anzulegende Spannung für das 12V-System. Der Verlauf für das 24V-System ist identisch, aber mit höherer Spannung. Der Funktionszustand muss mindestens B betragen.

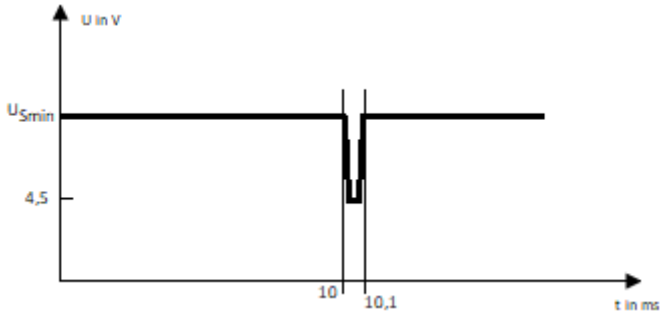

Abbildung 2-5: Kurzer Spannungseinbruch im 12V-System [9]

Rücksetzverhalten bei Spannungseinbruch

Mit diesem Test wird das Rücksetzverhalten für Geräte mit Reset-Funktion bei verschiedenen Spannungseinbrüchen überprüft. *Abbildung 2-6* zeigt den Verlauf der Testspannung, dabei wird diese ausgehend von der minimalen Versorgungsspannung um 5% reduziert und für 5s gehalten. Anschließend wird wieder die minimale Versorgungsspannung eingestellt, für mindestens 10s gehalten und daraufhin eine Funktionsüberprüfung durchgeführt. Im weiteren Verlauf wird die Spannung jedes Mal um zusätzliche 5% reduziert. Der Funktionszustand muss dabei mindestens C betragen.

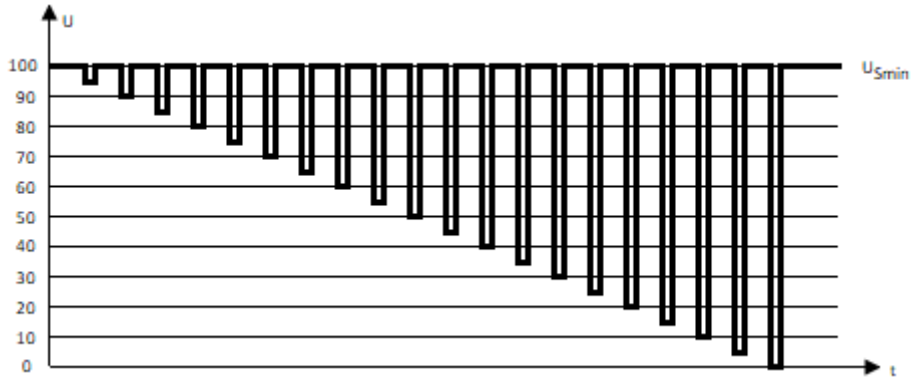

Abbildung 2-6: Verlauf der Versorgungsspannung für den Rücksetztest [9]

Startverhalten

Durch diesen Test wird das Verhalten eines Gerätes während und nach dem Anlassen des Motors überprüft. Das Startprofil in *Abbildung 2-7* wird 10-mal an allen relevanten Eingängen wiederholt, wobei zwischen den einzelnen Zyklen eine Pause von 1 bis 2 Sekunden eingelegt werden soll. Alle Funktionen die während dem Startvorgang benötigt werden, müssen sich im Funktionszustand A befinden. Dieser Test entspricht dem Testimpuls 4 wie er in einer alten Ausgabe der ISO 7637-2 [6] zu finden war.

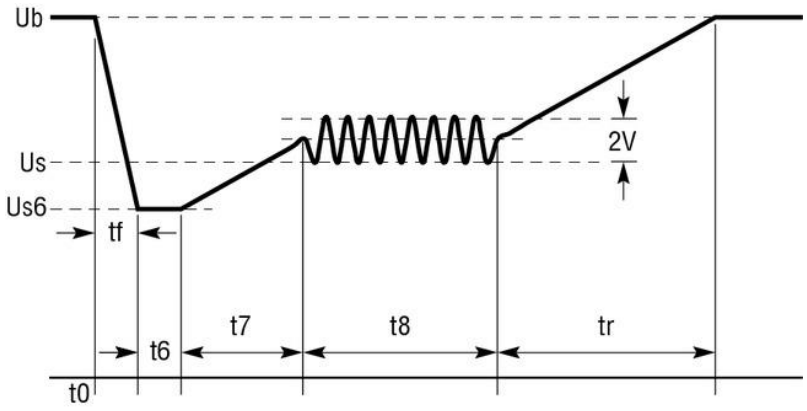

Abbildung 2-7: Spannungsverlauf beim Starten des Motors [7]

Lastabwurf („Load Dump“)

Dieser Test simuliert einen „Load Dump“-Impuls welcher entsteht, wenn eine entladene Batterie vom Generator getrennt wird, während dieser einen Ladestrom erzeugt und andere Geräte mit dem Generator elektrisch verbunden sind.

Abbildung 2-8 zeigt das Prinzipschaltbild zur Erzeugung des „Load Dump“-Impulses. Der „Load Dump“-Test wird in zwei Teile A und B unterteilt. Diese beiden Testimpulse entsprechen den Testimpulsen 5a und 5b einer alten Ausgabe der ISO 7637-2 [6]. Der Funktionszustand für beide Testimpulse muss dabei mindestens C betragen.

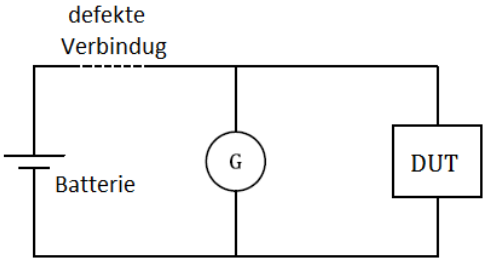

Abbildung 2-8: Prinzipschaltbild zum Lastabwurf [9]

Abbildung 2-9 und *Tabelle 2-7* zeigen den Impulsverlauf und die Parameter für den Testimpuls A eines Generators ohne zentrale „Load Dump“ - Unterdrückung.

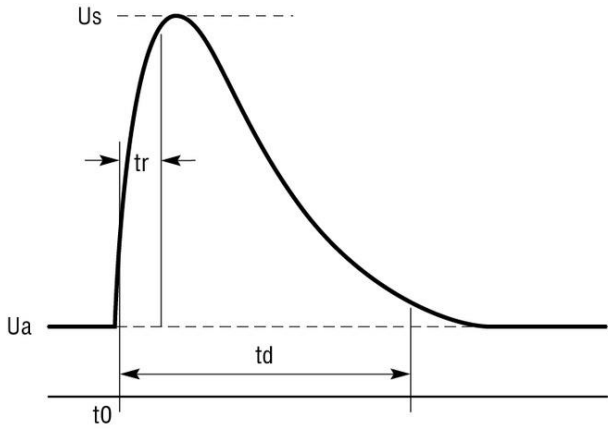

Abbildung 2-9: Testimpuls A für den "Load Dump" [7]

Parameter	12V System	24V System	Minimale Testanforderungen
U_S	79V bis 101V	151V bis 202V	10 Impulse mit einem Intervall von 1min
R_i	0,5Ω bis 4Ω	1Ω bis 8Ω	
t_d	40ms bis 400ms	100ms bis 350ms	
t_r	5ms bis 10ms	5ms bis 10ms	

Tabelle 2-7: Parameter für den Testimpuls A [9]

In *Abbildung 2-10* und *Tabelle 2-8* sind der Impulsverlauf und die Parameter für einen Generator mit zentraler „Load Dump“ – Unterdrückung dargestellt. Der Impulsverlauf zeigt deutlich wie der Spannungsimpuls unterdrückt wird.

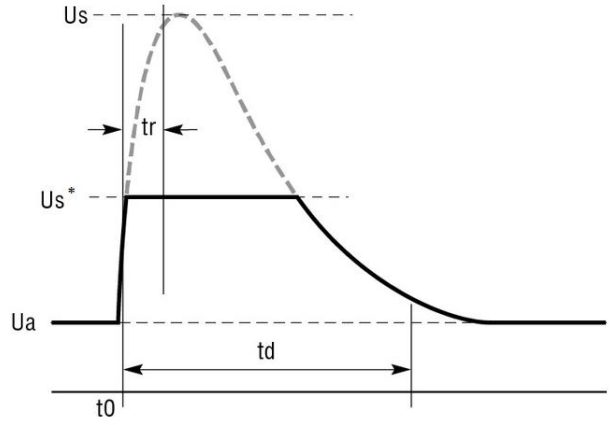

Abbildung 2-10: Testimpuls B für den "Load Dump" [7]

Parameter	12V System	24V System	Minimale Testanforderungen
U_S	79V bis 101V	151V bis 202V	5 Impulse mit einem Intervall von 1min
U_s^*	35	Definiert durch Auftraggeber (typischer Wert 58)	
R_i	0,5Ω bis 4Ω	1Ω bis 8Ω	
t_d	40ms bis 400ms	100ms bis 350ms	
t_r	5ms bis 10ms	5ms bis 10ms	

Tabelle 2-8: Parameter für den Testimpuls B [9]

Verpolung der Versorgungsspannung

Mit diesem Test wird überprüft ob das Gerät einer Verbindung mit einer vertauschten Batterie bei der Verwendung eines Starthilfegerätes standhält. Für den Test muss das Gerät wie im realen Auto verbunden und mit Sicherungen ausgestattet werden, aber ohne Generator oder Batterie.

Testfall 1: Wenn das Gerät in einem Auto ohne Sicherung für den Generator eingesetzt wird und die Gleichrichterdioden einer Verpolung für 60s standhalten, muss gleichzeitig an allen relevanten Eingängen für (60 ± 6)s eine Testspannung von 4V angelegt werden. Dieser Test gilt nicht für das 24V-System.

Testfall 2: In allen anderen Fällen muss eine Testspannung (14V bei 12V-System oder 28V bei 24V-System) mit umgekehrter Polarität gleichzeitig an allen relevanten Eingängen für (60 ± 6)s angelegt werden.

Wenn alle durchgebrannten Sicherungen ersetzt wurden, muss sich das Gerät wieder im Funktionszustand A befinden.

Bezugsmasse und Spannungsoffset

Dieser Test muss zwischen Kunde und Hersteller abgesprochen werden. Mit diesem Test wird sichergestellt, dass alle Komponenten bei zwei oder mehr Versorgungsleitungen (z.B. Masse der Spannungsversorgung und (Signal-Masse) funktionieren. Für den Test sollen alle Eingänge und Ausgänge mit entsprechenden Lasten oder Schaltungen verbunden werden, um eine Situation im Auto nachzustellen. Der Masse-/Spannungsversorgungsoffset muss

zwischen allen Massen und Versorgungsleitungen angelegt und getestet werden. Der Funktionszustand muss für alle Leitungen A betragen.

Leerlauftest

Unterbrechung einer Leitung

Mit diesem Test wird ein offener Kontakt simuliert. Das Gerät soll wie vorgesehen angeschlossen werden. Danach wird eine Leitung des Testgerätes geöffnet (Öffnungszeit (10 ± 1)s und Leerlaufwiderstand von ≥ 10 MΩ) und anschließend wieder verbunden. Der Funktionszustand des Gerätes muss mindestens C betragen.

Unterbrechung mehrerer Leitungen

Mit diesem Test wird überprüft ob das Gerät Funktionen erfüllt, wenn mehrere Leitungen des Gerätes unterbrochen werden. Die Parameter für den Test sind identisch zur Unterbrechung mit einer Leitung.

Kurzschlussfestigkeit

Für diesen Test werden die Ein- und Ausgänge des Gerätes auf deren Kurzschlussfestigkeit überprüft. Der Test wird für Signalleitungen (allen relevanten Eingängen und Ausgängen) und Lastschaltkreise durchgeführt. Für Signalleitungen und Lastschaltkreise mit elektronisch geschützten Ausgängen muss der Funktionszustand C betragen. Bei Lastschaltkreisen mit herkömmlichen Schmelzsicherungen für die Ausgänge soll sich das Gerät im Funktionszustand D befinden. Bei ungeschützten Ausgängen kann es zu Schäden kommen, weshalb der Funktionszustand hier E beträgt.

Spannungsfestigkeit

Anhand dieses Tests kann die Spannungsfestigkeit von Schaltungen und die galvanische Trennung von induktiven Bauteilen (z.B. Relais, Motor, Spulen) oder Lasten überprüft werden. Durch das bewusste Anlegen einer zu hohen Spannung an galvanisch getrennten Schaltungen kann das dielektrische Material auf Spannungsfestigkeit gegenüber hohen Spannungen beim

Abschalten von induktiven Lasten geprüft werden. Der Test wird mit einer sinusförmigen Testspannung mit einem Effektivwert von 500V für die Dauer von 60s an Klemmen und Gehäuse mit galvanischer Isolierung durchgeführt. Der Funktionszustand soll für diesen Test mindestens C betragen, außerdem darf kein Spannungsdurchschlag und kein Lichtbogen während dem Test entstehen.

Isolationswiderstand

Durch diesen Test wird ein minimaler ohmscher Widerstand sichergestellt, der Ströme zwischen galvanisch getrennten Schaltungen und leitfähigen Teilen verhindert. Dadurch werden Angaben zur Qualität des Materials und des Isolierungssystems erhalten. Der Test wird durchgeführt, indem an Klemmen und Gehäuse mit galvanischer Isolierung eine Gleichspannung von 500V für 60s angelegt wird. Für spezielle Anwendungen kann die Testspannung auch auf 100V reduziert werden, wenn dies zwischen Kunden und Hersteller abgesprochen wurde. Der Isolationswiderstand soll größer als 10MΩ sein.

2.1.5. Elektrostatische Entladungen

In der ISO 10605 sind die Prüfmethoden für **Elektrostatische Entladungen** (ESD-Test) für automotive Anwendungen genormt. Diese basiert auf dem Standard IEC 61000-4-2 und beschreibt elektrostatische Entladungen in der Fertigung, durch Servicepersonal und Fahrzeuginsassen. Überprüft werden sowohl einzelne elektronische Module als auch das komplette Fahrzeuge. Zusätzlich werden auch ESD-Tests für das Verpacken und die Hantierung der Module beschrieben. [10]

Beim ESD-Test nach IEC 61000-4-2 wird das „Human Body Model“ mit einem Hautwiderstand von 330Ω, 150pF Körperkapazität und einer maximalen Spannung von 15kV verwendet [11]. Anhand dieses Modells wird die Berührung einer aufgeladenen Person nachgebildet. Für automotive Anwendungen muss aber berücksichtigt werden, dass einerseits Werkzeuge aufgeladen sein können und andererseits das Fahrzeug als Bezugsmasse nicht Erdpotential, sondern die Masse der Fahrzeugbatterie verwendet. Deshalb testet die ISO 10605 zusätzlich auch noch mit 330pF und 2000Ω [12]. Durch diese zusätzlichen Parameter ergeben sich vier R/C-Kombinationsmöglichkeiten für den ESD-Test, dessen Prinzipschaltbild in *Abbildung 2-11* dargestellt ist. Die jeweiligen R/C-Kombinationen werden in einer ESD-Pistole durch entsprechende Prüfspitzen nachgebildet und sind davon abhängig ob innerhalb oder

außerhalb vom Fahrzeug getestet wird [12]. Der Test erfolgt mittels Kontakt- und Luftentladung, allerdings wird bei ISO 10605 mit bis zu 25kV geprüft. [12]

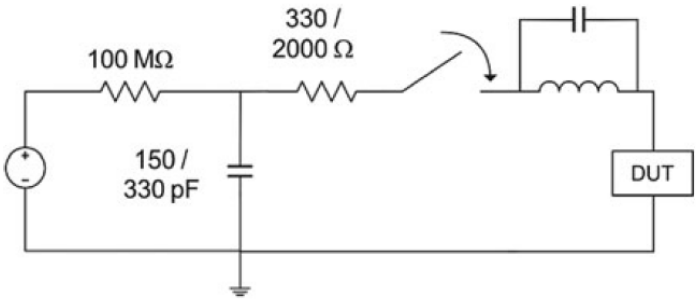

Abbildung 2-11: Prinzipschaltbild für den ESD-Test [13]

2.1.6. EMV-Maßnahmen

2.1.6.1 Allgemeines, Filter und Schutzmaßnahmen

Im Gegensatz zu Gebäuden oder Energienetz existiert im Fahrzeug keine Erdung, sondern nur der Minuspol der Batterie. Ein sternförmiger Massepunkt wäre für die elektromagnetische Verträglichkeit ideal, bedingt aber einen hohen Verkabelungsaufwand. Zur Reduktion der Impedanzkopplung sollten möglichst wenige Leitungen zusammen genutzt und die Impedanz durch große Leiterquerschnitte klein gehalten werden. Nach Möglichkeit sollten die Geräte gegen äußere elektromagnetische Felder abgeschirmt werden. Dabei ist zu beachten, dass schon sehr kleine Öffnungen in der Schirmung, beispielsweise durch Anschlüsse im metallischen Gehäuse, zu einer großen Verschlechterung der Abschirmung führen können. Aus Kostengründen werden sehr häufig nur Leitungen geschirmt und statt dem metallischen Gehäuse wird ein Kunststoffgehäuse verwendet. Die geschirmte Leitung (z.B. Koax-Kabel) besitzt meist einen dichten Schirm über der eigentlichen Signalleitung, wobei der Schirm an beiden Enden auf Masse gelegt wird. Alternativ können auch Leitungen verdrillt werden, weil dadurch die Induktion der eingestrahlten Fehler gegengleich ist und sich somit auslöscht. [2]

Elektrostatische Entladungen (ESD)

Für den Schutz vor elektrostatischen Entladungen werden hauptsächlich Suppressordioden, Polymerbauteile und Varistoren eingesetzt. Für die Suppressordioden, auch als TVS (Transient Voltage Suppressors) bezeichnet, werden Z-Dioden eingesetzt. Häufig werden diese in Form von Dioden-Arrays in einem gemeinsamen Gehäuse ausgeführt, um mehrere analoge oder digitale Leitungen, wie zum Beispiel die Datenleitungen von USB oder CAN, gleichzeitig

schützen zu können. Suppressordioden besitzen sehr kurze Ansprechzeiten, geringe Kapazitäten und Leckströme, sowie kleinstmögliche Klemmspannungen. Sie eigenen sich sowohl für geringe Datenraten als auch für High-Speed-Datenleitungen. Außerdem können sie problemlos hohen elektrostatischen Entladungen von bis zu 30kV ohne Verschlechterung der Eigenschaften standhalten. Polymerbauteile bestehen aus in Polymer eingebetteten Partikel. Sie zeichnen sich durch sehr geringe Kapazitäten und sehr kurze Ansprechzeiten aus, weshalb sie besonders gut für High-Speed Datenleitungen oder Hochfrequenzschaltungen geeignet sind. Allerdings besitzen sie auch sehr hohe Ansprechspannungen und sind daher nicht zum Schutz von integrierten Schaltungen im automotiven Bereich geeignet. Ein weiterer Nachteil ist der Verschleiß beim Klemmen. Bei Varistoren handelt es sich um spannungsabhängige Widerstände (je größer die Spannung desto kleiner der Widerstand) aus Keramik (z.B. Zinkoxid). Sie werden beispielsweise in SMD-Bauform als „Mulitlayer Varistors“ (Mehrschicht-Varistoren) sehr kostengünstig hergestellt, besitzen aber höhere Kapazitäten und sind ebenfalls von Verschleiß betroffen. Sie eigenen sich für den Schutz von Versorgungsleitungen oder Datenleitungen mit langsamen bis mittleren Datenraten. [14] [15] [16]

Für den Schutz von leitungsgeführten Störungen wie „Load Dump“, ESD oder Überspannung müssen die Amplituden begrenzt werden. Für den automotiven Bereich stehen eine Vielzahl an speziellen Schutzbauteilen zur Verfügung. Sehr häufig wird dabei eine Z-Diode oder ein Varistor in Parallelschaltung zum Bauteil eingesetzt.

Um Störungen abzuschwächen werden häufig kostengünstige passive Tiefpass-Filterschaltungen eingesetzt. Da die Störkopplung mit höheren Frequenzen zunimmt, kann dadurch ein Großteil der Störungen im Frequenzbereich oberhalb des Nutzsignales unterdrückt werden. Bei der Bauteilauswahl ist auf das zeitliche Verhalten und auf eine entsprechende Spannungsfestigkeit zu achten. Für Datenleitungen werden sehr häufig keramische SMD-Kondensatoren eingesetzt. Bei speziellen Anforderungen (z.B. Impulsfestigkeit) werden Folienkondensatoren verwendet. Als Entstörferrite (Induktivitäten) können stromkompensierte Drosseln als platzsparende und kostengünstige Variante eingesetzt werden. [11]

2.1.6.2 Layout-Maßnahmen

Bereits beim Erstellen des Layouts einer elektronischen Schaltung kann durch richtige Bauteilpositionierung für eine gute EMV gesorgt werden. Die Anschlüsse des Gerätes sollten

über möglichst kurze Wege nach außen geführt werden. Im Detail bedeutet dies, dass die Anschlüsse nicht über zusätzliche Kabel nach außen geführt, sondern direkt auf der Platine platziert werden. Filterschaltungen der Anschlüsse müssen in unmittelbarer Nähe zu den Buchsen platziert werden. [11]

Zur besseren Abschirmung sollten freie Flächen durch eine Massefläche ausgefüllt werden. Bei der Erstellung des Layouts kann eine falsche Platzierung der Bauteile zu einem Spalt in der Massefläche führen. Dieser kann im schlimmsten Fall wie eine Antenne wirken und eine Ab- bzw. Einstrahlung von Störungen begünstigen. *Abbildung 2-12* zeigt ein negatives Beispiel eines langen Spaltes in der Massefläche. Zur Verhinderung können entweder die Bauteile neu platziert werden oder die „Design Rules“ verbessert werden. [11]

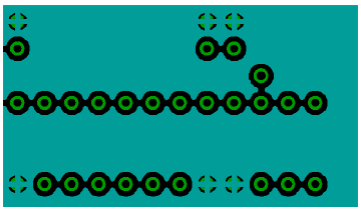

Abbildung 2-12: Durchkontaktierungen erzeugen einen Spalt [11]

Allerdings kann eine bewusste Integration eines Spaltes aus gewissen Gründen notwendig sein. Ein Beispiel ist die Trennung von Schaltungsteilen mit sehr großen geschalteten Strömen vom Rest der Schaltung. In diesem Fall ist unbedingt darauf zu achten keine Leitungen über diesen Spalt führen, da dies zu einer Antennenwirkung führen kann. In der *Abbildung 2-13* ist die richtige Handhabung zur Verlegung von Leitungen bei einem Spalt in der Massefläche dargestellt.

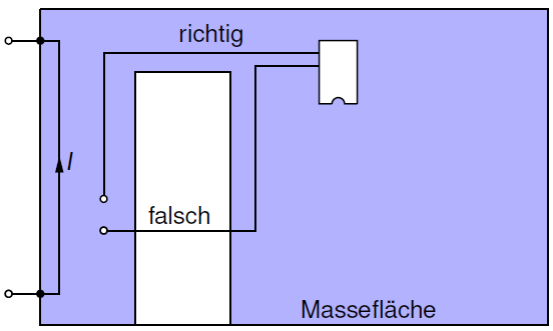

Abbildung 2-13: Vermeidung von Leitungen über Spalte in der Massefläche [11]

Durch schlecht platzierte Stützkondensatoren können diese nicht ihre maximale Wirkung erzielen, da die Induktivität der Leitungen dem Kondensator entgegenwirkt. In der *Abbildung 2-14* wird im linken Bild die schlechte Platzierung eines Stützkondensators dargestellt. Wesentlich bessere Eigenschaften erzielt der Stützkondensator durch eine Anordnung wie im rechten Bild. [11]

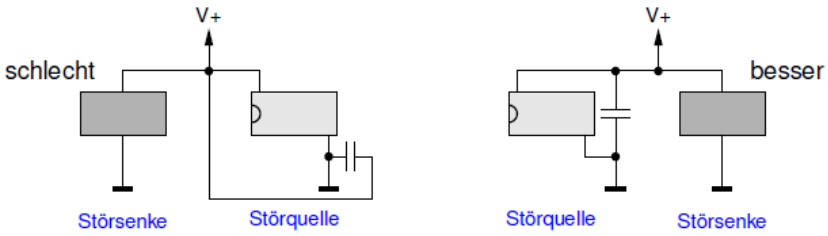

Abbildung 2-14: Stützkondensator auf möglichst kurzem Weg anordnen [11]

Aber auch bei Filterschaltungen muss ebenfalls auf eine richtige Anordnung geachtet werden. Werden die Filter aufgrund von Platzmangel wie in *Abbildung 2-15* angeordnet, kann es zu kapazitivem Übersprechen zwischen ungefilterten und gefilterten Leitung kommen. [11]

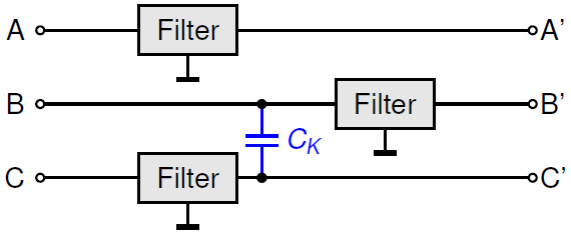

Abbildung 2-15: Fehlerhafte Anordnung der Filter aus Platzmangel [11]

Für eine geringere Störemission sollten alle Anschlüsse auf einer Seite der Platine bzw. möglichst nahe platziert werden. Eine gegenüberliegende Anordnung kann durch den Spannungsabfall eine erhöhte Störaussendung verursachen. *Abbildung 2-16* zeigt die richtige Anordnung der Anschlüsse auf der Platine. [17]

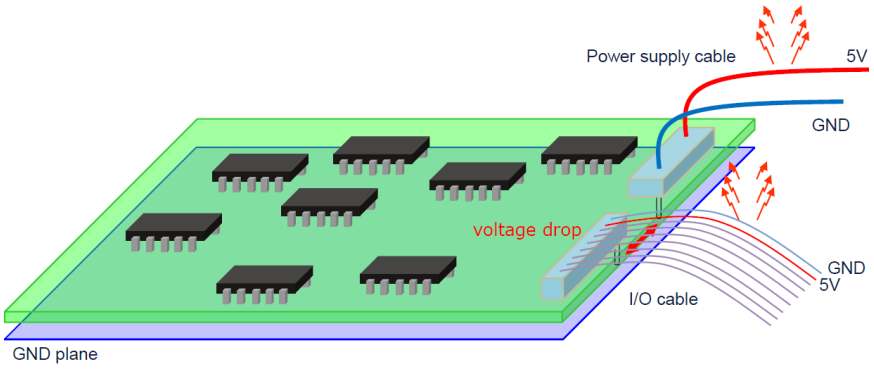

Abbildung 2-16: Anordnung der Anschlüsse auf einer Seite der Platine [17]

Störungsbehaftete Leitungen sollten nach Möglichkeit keine großen Flächen aufspannen und können zur besseren Abschirmung auch zwischen Masseleitungen eingebettet werden. *Abbildung 2-17* zeigt, wie eine Leiterbahn mit hochfrequenten Signalen zwischen Masseleitungen eingebettet wird. Bei Kabeln kann dies durch Mitführen von Bezugsleitern erfolgen, wodurch der Signalrückstrom neben dem hinlaufenden Signalstrom liegt. Diese Maßnahme führt ebenfalls zu einer Reduzierung der aufgespannten Fläche. [11]

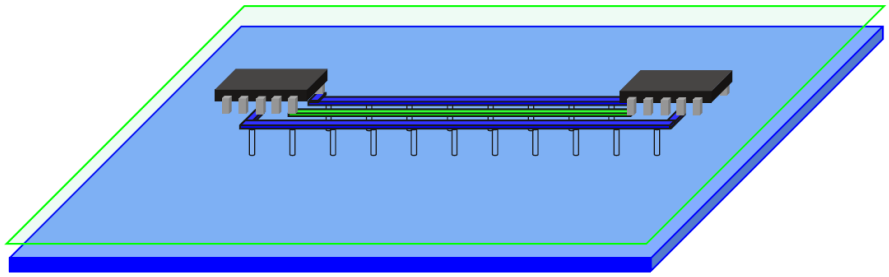

Abbildung 2-17: Leiterbahn mit hochfrequenten Signalen zwischen Masseleitungen [17]

2.2. Sensorprotokolle

In diesem Kapitel erfolgt eine Einführung in die wichtigsten Grundlagen der Sensor-Schnittstellen SENT und PSI5. Zusätzlich zu beiden genannten Übertragungsprotokollen existieren weitere Systeme wie der „Automotive Safety Restraint Bus (ASRB 2.0)“ oder das „Distributed Systems Interface (DSI)“ welche im Rahmen dieser Masterarbeit nicht behandelt wurden. [18]

2.2.1. SENT – Single Edge Nibble Transmission

Das SENT-Protokoll wurde im April 2007 von SAE (Society of Automotive Engineers) mit der Bezeichnung J2716 standardisiert und zuletzt im Jahr 2016 aktualisiert. SENT soll bisherige analoge und pulsweitenmodulierte Signale durch hochauflösende digitale Signale ersetzen bzw. als kostengünstige Alternative für bereits bestehende Bussysteme, z.B. CAN (Controller Area Network) oder LIN (Local Interconnect Network), eingesetzt werden. [19]

Die Inhalte der nachfolgenden Unterkapitel wurden dem Standard „J2716 JAN2010“ der Society of Automotive Engineers [19] entnommen. Anderslautende Informationsquellen werden an der entsprechenden Textpassage gesondert angeführt.

2.2.1.1 SENT - Protokoll

Die Übermittlung der Daten erfolgt bei SENT (Single Edge Nibble Transmission) in Form von Zeitspannen zwischen zwei fallenden Flanken der digitalen High- und Low-Pegel. Diesem Prinzip entspringt auch die Bezeichnung „Single Edge“. Bei SENT besteht die Möglichkeit in einer Botschaft zwei Signale zu je 12 Bit zu übertragen. Diese 12 Bit Werte werden dabei in Gruppen von jeweils 4 Bit („Nibble“) pro Impuls gesendet.

In der *Abbildung 2-18* ist ein Beispiel für eine mögliche SENT-Botschaft dargestellt. Jede Nachricht beginnt mit einem Synchronisationsimpuls mit fester Länge von 56 „ticks“. Ein „tick“ entspricht dabei in der Regel einer Zeitbasis von 3µs, kann bei Bedarf vom Hersteller aber auf bis zu 90µs erhöht werden. Die Toleranzen für die Zeitbasis liegen in einem Bereich von ±20%. Durch die definierte Länge von 56 „ticks“ für den Synchronisationsimpuls kann der Empfänger die exakte Zeitbasis des Senders für die weitere Auswertung bestimmen (Zeit zwischen der ersten und zweiten fallenden Signalflanke dividiert durch 56). Bei einer Zeitbasis von 3µs ergibt dies eine Zeitdauer von 168µs. Mit Hilfe der ermittelten Zeitbasis können nun alle darauffolgenden Datenwerte beim Empfänger berechnet werden.

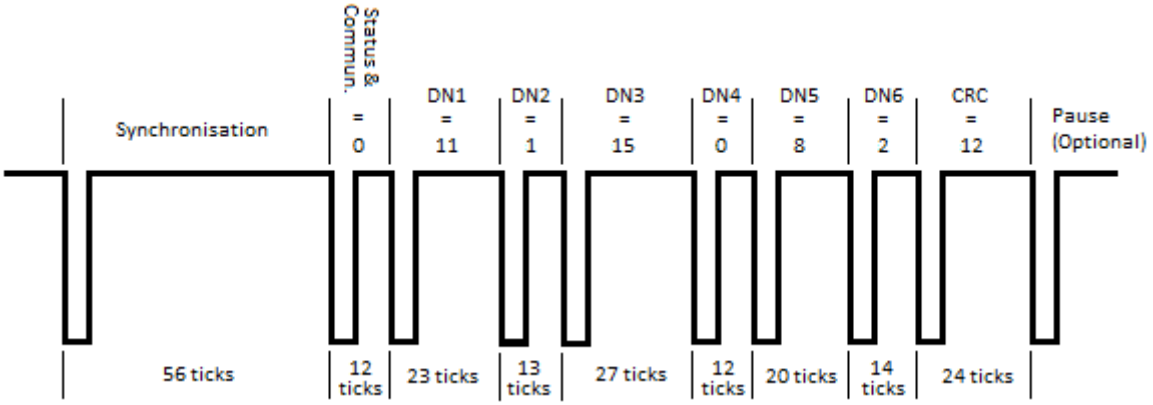

Abbildung 2-18: Beispiel für eine SENT-Botschaft mit 6 Daten-Nibble (DN) und optionaler Pause

Nach dem Synchronisationsimpuls folgt ein „Status- und Kommunikations-Nibble“ (4 Bit), welches beispielsweise zur Kennzeichnung von Fehlerzuständen oder zur Übertragung von Daten in einem seriellen Datenkanal (siehe Kapitel 2.2.1.2) verwendet werden kann. Anschließend folgen zwischen ein und sechs Daten-Nibble, welche in *Abbildung 2-18* mit „DN“ abgekürzt wurden. Die Anzahl der Daten-Nibble hängt vom jeweiligen Sensortyp (Drucksensor, Sicherheitssensor, Positionssensor, etc.) ab bzw. kann vom Sensorhersteller festgelegt werden. Die Anzahl muss aber für alle Nachrichten eines Sensors konstant bleiben. Im Beispiel von *Abbildung 2-18* entsprechen „DN1“ bis „DN3“ dem „Signal 1“ mit 12 Bit und „DN4“ bis „DN6“ dem „Signal 2“ mit 12 Bits. Abschließend folgt immer ein CRC-Nibble zur Erkennung von Übertragungsfehlern.

Die Auswertung der übertragenen Nibble (Status & Kommunikation bis zum CRC) erfolgt durch die anfangs berechnete Zeitbasis (3µs). Wie bereits beim Synchronisationsimpuls wird die Zeit zwischen den fallenden Flanken gemessen und anschließend durch die ermittelte Zeitbasis dividiert, um die Anzahl der „ticks“ zu erhalten. Jedes Nibble hat ein Minimum von 12 „ticks“ (36µs bei 3µs Zeitbasis), dies entspricht einem dezimalen Wert von 0. Da ein Nibble eine Größe von 4 Bit besitzt, beträgt die maximale Zeitdauer 27 „ticks“ (81µs bei 3µs Zeitbasis) und entspricht einem Dezimalwert von 15. In *Abbildung 2-18* ist für jedes Nibble der dekodierte Dezimalwert dargestellt. Für alle Impulse gilt, dass die Low-Periode mindestens 5 „ticks“ lang ist.

Eine SENT-Nachricht mit 24 Bit bewegt sich, in Abhängigkeit der gesendeten Datenwerte, im Rahmen von 154 bis 270 „ticks“ (462µs bis 810µs bei 3µs Zeitbasis). Die Datenrate für 24 Bit liegt, ohne Berücksichtigung der Toleranzen, bei rund 29,6 kBit/s.

Zusätzlich besteht auch noch die Möglichkeit einen Pause-Impuls an die SENT-Botschaft anzufügen. Dadurch können beispielsweise Übertragungen mit einer festen Zeitdauer von 1ms realisiert werden. Die Pause muss mindestens 12 „ticks“ und darf maximal 768 „ticks“ lang sein.

In der *Tabelle 2-9* ist der Aufbau des Status- & Kommunikations-Nibbles dargestellt. Die 4 Bits sind in zwei Bereiche aufgeteilt: Die Bedeutung von Bit Null und Eins kann je nach Applikationsbeispiel variieren und sollte dem Datenblatt des Sensorherstellers entnommen werden. Die Bits Zwei und Drei werden für einen seriellen Datenkanal verwendet. Bei diesem werden über mehrere SENT-Botschaften hinweg zusätzliche Informationen, wie beispielsweise die Seriennummer oder der Herstellercode, übermittelt. Weitere Details zum seriellen Datenkanal werden im nächsten Kapitel *2.2.1.2* erklärt.

Bit – Position	Bedeutung
0 (LSB)	Verwendung von der Applikation abhängig
1	Verwendung von der Applikation abhängig
2	Bits für den seriellen Datenkanal
3 (MSB)	Beginn der seriellen Datenübertragung = 1, die restlichen Werte sind 0 bzw. enthalten Daten

Tabelle 2-9: Aufbau des Status- & Kommunikations-Nibble

Die Berechnung des CRC-Nibble erfolgt mit dem Polynom $x^4 + x^3 + x^2 + 1$ und einem Initialisierungswert von 5 („0101“ binär). Die Berechnung erfolgt ausschließlich anhand der Daten-Nibble, das Status & Kommunikations-Nibble wird nicht in die Berechnung miteinbezogen. Dadurch können hier Bitfehler auftreten, die vom Empfänger nicht erkannt werden. Als Unterstützung zur Implementierung der CRC-Berechnung wird in der SENT-Norm [19] auf Seite 18 ein Beispiel (Matlab – Code) mit einer „Lookup table“ angegeben. Im Anhang „Appendix B.1“ werden auch mehrere Beispiele für Daten-Nibble und den daraus resultierenden CRC-Nibble zur Verfügung gestellt.

2.2.1.2 Serieller Datenkanal - „Serial Messages“

Das SENT-Protokoll bietet zusätzlich zur Übertragung der Messwerte die Möglichkeit weitere Informationen über einen seriellen Datenkanal zu übermitteln. Diese „Serial Messages“ werden, wie in *Tabelle 2-9* dargestellt, über eine bestimmte Anzahl von SENT-Nachrichten mit dem Bit 2 und 3 des Status- & Kommunikations-Nibble übertragen. Für den Empfang einer

„Serial Message“ werden daher mehrere einzelne SENT-Nachrichten benötigt, weshalb dieser Sub-Kanal auch als „Slow Channel“ bezeichnet wird. Bei den „Serial Messages“ wird zwischen den „Short Serial Messages“ und „Enhanced Serial Messages“ unterschieden.

In der *Tabelle 2-10* wird der Aufbau der „Short Serial Messages“ dargestellt. Für deren Empfang müssen insgesamt 16 einzelne SENT-Nachrichten in Folge fehlerfrei eingelesen werden. Insgesamt besteht eine „Short Serial Message“ aus 16 Bit an Informationen, welche im Bit 2 übertragen werden. Die erste Nachricht enthält das MSB, das LSB wird mit der 16. Nachricht empfangen. Die Bits der ersten vier Nachrichten enthalten eine „Message ID“ zur Kennzeichnung der seriellen Daten. Die Definition der möglichen IDs ist von der jeweiligen Applikation abhängig und wird durch den Sensorhersteller angegeben. Anschließend folgt ein Byte (8 Bit) mit Daten. Die letzten 4 Bits werden für eine CRC-Prüfsumme, zur Erkennung von Übertragungsfehlern, verwendet. Die Berechnung dieses CRC-Nibble erfolgt analog zu den herkömmlichen Sensordaten aus Kapitel 2.2.1.1. Der Beginn einer neuen „Short Serial Message“ wird durch eine „1“ im Bit 3 des Status- & Kommunikations-Nibble signalisiert, die restlichen Bits müssen den Wert Null besitzen.

SENT-Nachricht	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Status & Komm. Bit 3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Status & Komm. Bit 2	Message ID				Datenbyte								CRC			

Tabelle 2-10: Short Serial Message – Struktur

Mit der Verwendung von „Enhanced Serial Messages“ besteht die Möglichkeit größere Datenmengen gegenüber den „Short Serial Messages“ zu übermitteln. Die Übertragung erfolgt dabei ähnlich wie zuvor bei den „Short Serial Messages“. Es werden ebenfalls die Bits 2 und 3 des Status- & Kommunikations-Nibble mehrerer einzelner SENT-Nachrichten verwendet. Insgesamt müssen hier 18 einzelne SENT-Nachrichten fehlerfrei empfangen werden. Bei den „Enhanced Messages“ wird nun auch das Bit 3 zur Übertragung von Dateninformationen genutzt, das Format besitzt aber weiterhin einzelne definierte Steuerbits um beispielsweise den Start erkennen und überprüfen zu können. Der Strukturaufbau wird in der *Tabelle 2-11* dargestellt.

SENT-Nachricht	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Status & Komm. Bit 3	1	1	1	1	1	1	0	C	8 Bit ID (Bit 7-4) / 4 Bit ID (Bit 3-0)				0	8 Bit ID (Bit 3-0) / 4 Bit Daten				0
Status & Komm. Bit 2	6 Bit CRC						12 Bit Datenfeld											

Tabelle 2-11: Enhanced Serial Messages – Struktur

Das Bit 3 des Status- & Kommunikations-Nibble muss in den ersten 6 SENT-Nachrichten auf „1“ gesetzt sein, worauf zwingend eine „0“ folgt. In Nachricht 13 bzw. 18 muss dieses Bit ebenfalls auf „0“ gesetzt sein.

Mit den „Enhanced Serial Messages“ können insgesamt 20 Bit an Informationen übertragen werden. Dafür werden im Bit 3 die SENT-Nachrichten 9 bis 12 bzw. 14 bis 17 sowie im Bit 2 die SENT-Nachrichten 7 bis 18 verwendet. Mit Hilfe eines Konfigurations-Bits (Bit 3, Nachricht 8) wird angegeben, wie die Zusatzinformationen kodiert werden. Es sind folgende Varianten möglich:

- 8 Bit „Message ID“, 12 Bit Daten
- 4 Bit „Message ID“, 16 Bit Daten

In der *Tabelle 2-12* ist das Konfigurations-Bit „0“ gesetzt, in diesem wird Bit 3 als 8 Bit „Messages ID“ verwendet. Die zweite Variante ist in *Tabelle 2-13* dargestellt, hier wird Bit 3 als 4 Bit „Message ID“ sowie 4 Bits für die Daten verwendet. Es wird immer zuerst das MSB übertragen bzw. bei 16 Bit Daten beginnt das MSB im Bit 3 von SENT-Nachricht 14.

SENT-Nachricht	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Status & Komm. Bit 3	1	1	1	1	1	1	0	**0**	8 Bit ID (Bit 7-4)				0	8 Bit ID (Bit 3-0)				0
Status & Komm. Bit 2	6 Bit CRC						12 Bit Datenfeld											

Tabelle 2-12: Enhanced Serial Messages - 8 Bit ID und 12 Bit Daten

SENT-Nachricht	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Status & Komm. Bit 3	1	1	1	1	1	1	0	**1**	4 Bit ID (Bit 3-0)				0	16 Bit Daten (15-12)				0
Status & Komm. Bit 2	6 Bit CRC						16 Bit Datenfeld (Bit 11 – 0)											

Tabelle 2-13: Enhanced Serial Messages – 4 Bit ID und 16 Bit Daten

Zur Erkennung von Übertragungsfehlern wird bei den „Enhanced Messages“ eine 6 Bit CRC-Prüfsumme in den Nachrichten 1 bis 6 im Bit 2 übertragen. Auch hier wird immer zuerst das MSB empfangen.

Für die „Enhanced Serial Messages“ gibt es ein vordefiniertes Set an „Message IDs“. In der *Tabelle 2-14* ist ein Auszug einiger IDs und der zugehörigen Bedeutung angeführt. Eine vollständige Liste aller möglichen „Message IDs“ können in der SENT-Norm J2716 [19] im „Appendix D.1“ ab Seite 50 nachgelesen werden.

Alle IDs sind optional und können je nach Bedarf des Herstellers übermittelt werden. Maximal können 32 IDs für einen sich wiederholenden Zyklus verwendet werden. Es können IDs innerhalb eines Zyklus auch mehrfach verwendet werden. Wird eine „Message ID“ nicht verwendet, so muss der Empfänger die Standardwerte für die jeweilige Applikation annehmen.

Message ID	Beschreibung
0x03	Sensortype
0x04	Sensor – Konfigurationscode
0x05	Herstellercode
0x06	SENT Version

Tabelle 2-14: Beispiel für die 8 Bit „Message IDs“ bei „Enhanced Serial Messages“

Die Berechnung der 6 Bit CRC Prüfsumme erfolgt mit Hilfe des Polynoms $x^6 + x^4 + x^3 + 1$ und einem Initialisierungswert von 21 („010101“ binär). Um den CRC Wert aus der Datenübertragung ermitteln zu können, müssen die einzelnen Bits in einer bestimmten Reihenfolge anhand *Tabelle 2-15* verarbeitet werden. Die Bits werden in 4 Blöcken zu je 6 Bits unterteilt und anschließend um einen weiteren Block mit 6 Nullen ergänzt werden. Für die Berechnung steht auch hier eine Hilfestellung (Matlab-Code) auf Seite 12 sowie Beispielwerte im „Appendix B.2“ der SENT-Norm zur Verfügung.

SENT-Nachricht	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Status & Komm. Bit 3	1	1	1	1	1	1	**1**	**3**	**5**	**7**	**9**	**11**	**13**	**15**	**17**	**19**	**21**	**23**
Status & Komm. Bit 2	6 Bit CRC						**0**	**2**	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**	**22**

Tabelle 2-15: Bitreihenfolge (0 bis 23 in violett) zur Berechnung der 6 Bit CRC-Prüfsumme

2.2.1.3 Physikalische Ebene

Für die Übermittlung der SENT-Botschaften wird eine unidirektionale Datenverbindung mit nur drei Leitungen verwendet. Der Sender (in der Regel ein Sensor) wird dabei durch den Empfänger (Mikrocontroller / CPU) über eine eigene Versorgungsleitung U_+ und einen

Masseanschluss mit der nötigen Betriebsspannung von +5V versorgt. Die SENT-Nachrichten werden vom Sender zum Empfänger über eine eigene Signalleitung übertragen. In *Abbildung 2-19* wird die von der SENT-Norm empfohlene Hardwarekonfiguration dargestellt.

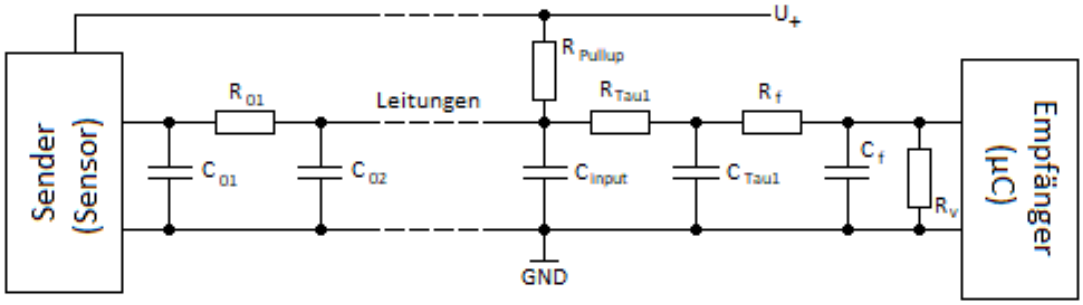

Abbildung 2-19: Empfohlene Hardwarekonfiguration für SENT

2.2.1.4 Applikationsbeispiele

Der SENT-Standard liefert im „Appendix A“ für insgesamt fünf Sensoranwendungen empfohlene Systemkonfigurationen. In der folgenden Auflistung werden die drei für diese Masterarbeit benötigten Anwendungen vorgestellt:

„Throttle Position“ (Drosselklappensensoren):

Bei diesem Applikationsbeispiel können zwei Drosselklappensensoren in einer SENT-Botschaft ihre Messwerte übertragen. Die Nibble von Sensor 2 werden dabei mit geänderter Reihenfolge +übertragen. **Achtung**: Es ändert sich hier nur die Anordnung der Nibble, die Bitreihenfolge innerhalb eines Nibbles bleibt unverändert!

- DN 1: Sensor 1 MSN
- DN 2: Sensor 1 MidN
- DN 3: Sensor 1 LSN
- DN 4: Sensor 2 LSN
- DN 5: Sensor 2 MidN
- DN 6: Sensor 2 MSN

DN … Data Nibble, MSN … Most significant Nibble, LSN … Least significant Nibble, MidN … Middle Nibble

Wenn der Sensor 1 einen internen Fehlerzustand erkennt, wird dieser Fehler im Bit 0 des Status- und Kommunikations-Nibble durch eine „1“ signalisiert. Zusätzlich werden alle Nibble

von Sensor 1 auf den Hexwert 0xF gesetzt. Ein Fehlerzustand von Sensor 2 wird im Bit 1 des Status- und Kommunikations-Nibble ebenfalls durch eine „1“ gekennzeichnet. Die Daten-Nibble erhalten in diesem Fall den Wert „0x0“.

Die Verwendung von „Serial Messages“ ist für dieses Anwendungsbeispiel nicht vorgesehen.

„Single Secure“:

Diese Sensortypen werden für diverse Sicherheitssysteme im Fahrzeug eingesetzt. Ein Beispiel für eine Anwendung sind Sensoren für Bremspedale. „Single Secure“ Sensoren verwenden nur ein Sensorsignal, die Daten-Nibble 4 und 5 sind hier als 8 Bit Sicherheitszähler ausgeführt. Zusätzlich wird in Daten-Nibble 6 die invertierte Kopie von „Sensor 1 MSN“ übermittelt. Dadurch kann der Empfänger jederzeit den lückenlosen und fehlerfreien Empfang überwachen.

- DN 1: Sensor 1 MSN
- DN 2: Sensor 1 MidN
- DN 3: Sensor 1 LSN
- DN 4: Sicherheitszähler MSN
- DN 5: Sicherheitszähler LSN
- DN 6: invertierte Kopie von DN1

Befindet sich der Sensor in einem Fehlerzustand sendet er ebenfalls im Bit 0 des Status- und Kommunikations-Nibble eine „1“. Das Bit 1 muss immer „0“ gesetzt sein. Auch hier ist die Verwendung von „Serial Messages“ nicht vorgesehen.

„Single Sensors“:

Dieser Applikationstyp wird als allgemeines Sensorsystem verwendet. Die Konfiguration der Daten-Nibble und sonstigen Regelungen sind identisch mit den „Single Secure“ Sensoren. Optional können DN4 bis DN6 auch auf den Wert „0“ gesetzt werden.

Weitere Applikationen:

Neben den hier vorgestellten Beispielen gibt es noch Anwendungsbeispiele für „Mass Air Flow“ (Luftmassenmesser) und „Pressure or Pressure/Secure“ (Druck- oder Druck/Sicherheits-Sensoren), welche im Rahmen dieser Masterarbeit nicht näher behandelt wurden. Die

detaillierten Konfigurationsmöglichkeiten für diese Anwendungen können im SENT-Standard [19] im „Appendix A“ nachgelesen werden.

2.2.2. PSI5 – Peripheral Sensor Interface 5

Das „Pheripheral Sensor Interface 5“ (PSI5) wurde vom „PSI5 Steering Committee“, welchem die Firmen Autoliv, Continental und Bosch angehören, erstmals im Jahr 2008 als freier Standard veröffentlicht. Die letzte Aktualisierung erfolgte im Februar 2018. Neben dem „PSI5 Steering Committee“ sind eine Vielzahl an großen Sensorherstellern (z.B. ELMOS, NXP, Hella, Infineon, Analog Devices, etc.) an der Entwicklung beteiligt. [20]

PSI5 wurde mit dem Ziel entwickelt, die bereits existierenden Airbag-Sensoren hinsichtlich der zuverlässigen Datenübertragung bei gleichzeitig niedrigen Implementierungskosten zu verbessern. Der PSI5-Standard ist deshalb als freier Standard kostenlos erhältlich. Die wichtigsten Eckdaten sind eine Zweidrahtleitung, Manchester-Codierung zur digitalen Datenübertragung und eine hohe Übertragungsrate. Die Einsatzmöglichkeit von PSI5 beschränkt sich nicht ausschließlich auf Airbag-Sensoren, sondern lässt sich auch für viele andere automotive Bereiche einsetzen. Im Jahr 2011 wurde der PSI5-Standard deshalb mit Version 2.0 in einen „Base Standard“ und drei Substandards aufgeteilt [20]:

- Substandard – Airbag
- Substandard – Vehicle Dynamics Control (ab 2012 “Chassis and Safety Control”)
- Substandard – Powertrain

Der Base-Standard beinhaltet allgemeine und grundlegende Spezifikationen zur Funktionsweise von PSI5, in den Substandards werden spezielle Spezifikationen bzw. Empfehlungen für den jeweiligen Einsatzbereich festgelegt.

Die Inhalte der Kapitel *2.2.2.1* bis *2.2.2.4* wurden dem PSI5-Base-Standard „PSI5 Peripheral Sensor Interface for Automotive Applications, Base Standard V2.3“ des „PSI5 Steering Committee“ entnommen [20]. Anderslautende Informationsquellen werden an der entsprechenden Textpassage gesondert angeführt.

Die Informationsquellen für die Substandards werden in den entsprechenden Unterkapiteln angeführt.

2.2.2.1 Base Standard – Übertragungsprotokoll

Im PSI5 – Protokoll werden die Sensordaten periodisch in sogenannten „Frames“ an den Empfänger (Mikrocontroller) übermittelt. In der *Abbildung 2-20* ist das vollständige Frame-Konstrukt inklusive der möglichen Datenfelder dargestellt. Anmerkung: Die Grafik wurde aus dem Base-Standard V2.2 [21] übernommen, da in der überarbeiteten Version 2.3 offensichtliche Fehler vorhanden sind.

Das PSI5-Frame besteht im Allgemeinen aus drei Teilen: zwei Startbits, einer „Payload Data Region“ (Nutzdaten) und am Ende folgt ein Paritätsbit bzw. eine 3 Bit CRC-Prüfsumme zur Fehlerüberprüfung.

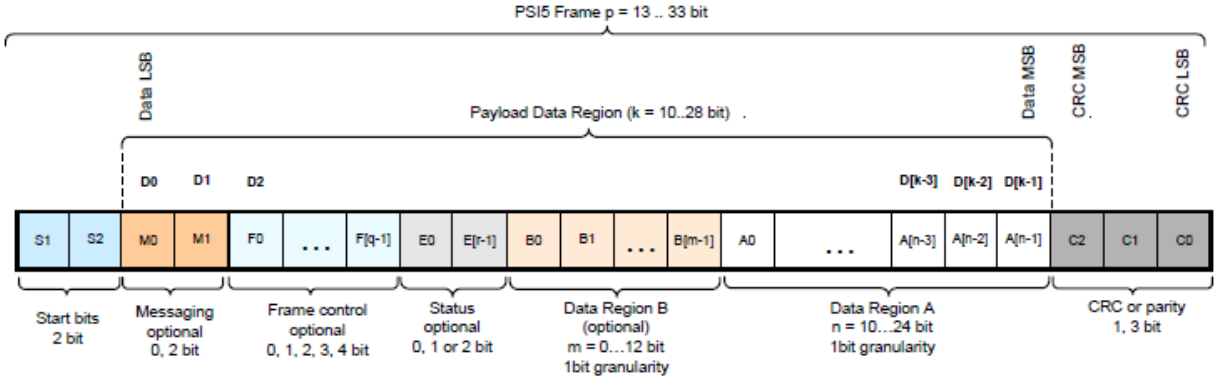

Abbildung 2-20: Vollständiges PSI5-Übertragungsprotokoll [21]

Die beiden Startbits werden immer mit dem Wert „0“ gesendet. Die Region für die Nutzdaten besitzt eine Größe von mindestens 10 und maximal 28 Bits. Je nach Sensortyp und Anwendungsbereich können unterschiedliche Datenfelder für die Nutzdatenregion verwendet werden:

- Messaging: Dieses optionale Datenfeld wird als serieller Datenkanal verwendet und wurde aus dem SENT-Standard „J2716“ [19] übernommen. Der Aufbau und die Funktionsweise ist identisch zu den „Enhanced Serial Messages“ im SENT-Standard (siehe Kapitel *2.2.1.2* bzw. *Abbildung 2-21*). Im Powertrain-Substandard Kapitel 3.1.2 wird auf die Übernahme der SENT-Spezifikation hinsichtlich des „Serial Channels“ hingewiesen [22].

- Frame-Control: Dieses optionale Datenfeld kann mit 0 bis 4 Bits verwendet werden. Es besteht die Möglichkeit in der periodischen Übertragung der Frames unterschiedliche

Messwerte zu kennzeichnen, so kann zum Beispiel Frame 1 eine Druckmessung und Frame 2 eine Temperaturmessung enthalten.

- Status: Mit diesem optionalen Feld können vom Sensor intern erkannte Fehlerzustände signalisiert werden. Es stehen zwischen 0 und 2 Bits zur Verfügung.

- Datenregion B: Dieses optionale Datenfeld kann zur Übertragung von sekundären Messwerten verwendet werden. Es stehen 0 bis 12 Bits zur Verfügung.

- Datenregion A: Dieses Datenfeld enthält die primären Sensormessdaten und ist daher für alle Übertragungen verpflichtend zu verwenden. Insgesamt können zwischen 10 und 24 Bit für dieses Datenfeld gewählt werden.

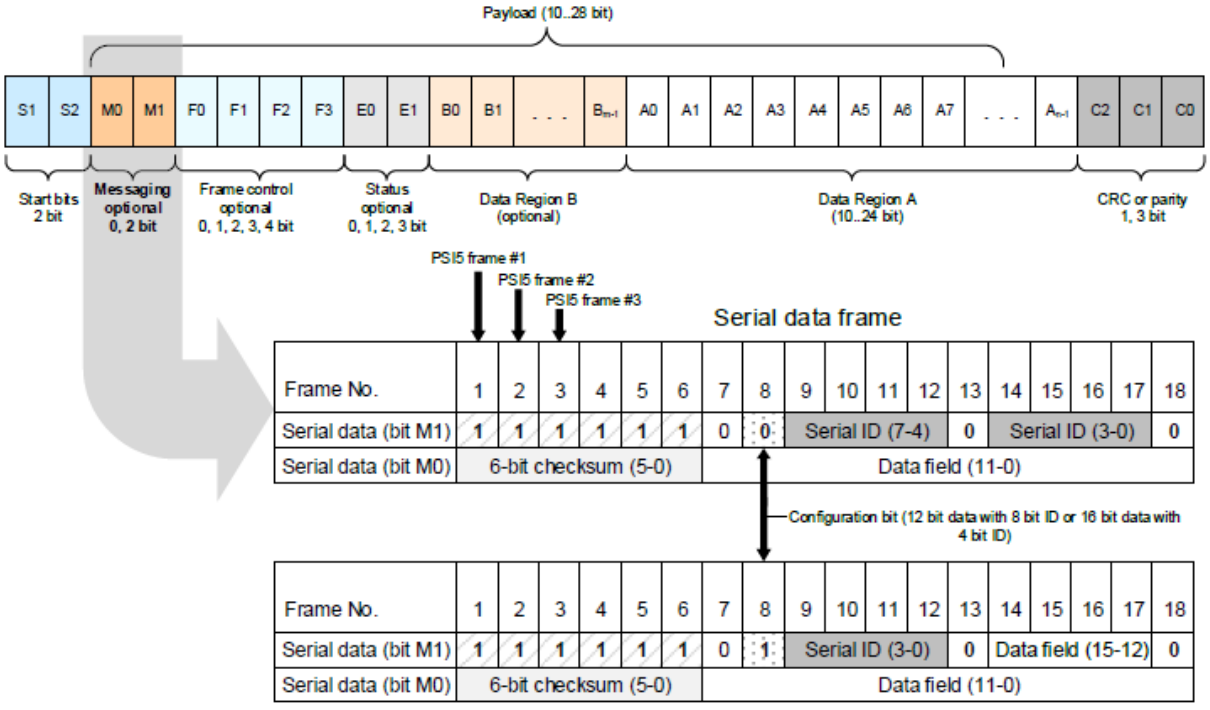

Abbildung 2-21: Identischer Aufbau der PSI5 „Enhanced Serial Messages“ wie im SENT-Standard [20]

Bei der Auswahl der Datenfelder ist darauf zu achten, dass für die Datenregion A mindestens 10 Bit verwendet werden müssen. Für die gesamte Nutzdatenregion dürfen jedoch nicht mehr als 28 Bit eingesetzt werden. Weiters muss beachtet werden, dass die Nutzdaten in der Reihenfolge LSB zu MSB und die CRC-Prüfsumme jedoch MSB zu LSB gesendet wird.

Die Überprüfung der empfangenen Nutzdaten auf Übertragungsfehler kann entweder mit einem Paritätsbit (gerade Parität) oder mit einer 3 Bit CRC-Prüfsumme erfolgen. Die Startbits werden bei der Prüfung nicht miteinbezogen. Das Paritätsbit wird in der Regel für 10 Bit Datenwerte verwendet, für längere Datenblöcke wird die CRC-Prüfsumme verwendet. Diese wird mit dem Generatorpolynom $x^3 + x + 1$ und dem Initialisierungswert 7 („111“ binär) berechnet. Im Gegensatz zu SENT wird bei PSI5 kein Beispielcode zur Implementierung zur Verfügung gestellt.

2.2.2.2 Base Standard - Betriebsarten

Bei PSI5 wird zwischen zwei Betriebsarten, einem asynchronen und synchronen Betrieb, unterschieden. Beim asynchronen Betrieb erfolgt eine unidirektionale Kommunikation vom Sensor zum Empfänger (z.B. Mikrocontroller) über eine Punkt-zu-Punkt Verbindung mit einer Zweidrahtleitung. Nachdem sich am Sensor die entsprechende Versorgungsspannung eingestellt hat, beginnt der Sensor seine Daten kontinuierlich an den Empfänger zu senden. In der *Abbildung 2-22* ist das Prinzip dieser Betriebsart dargestellt.

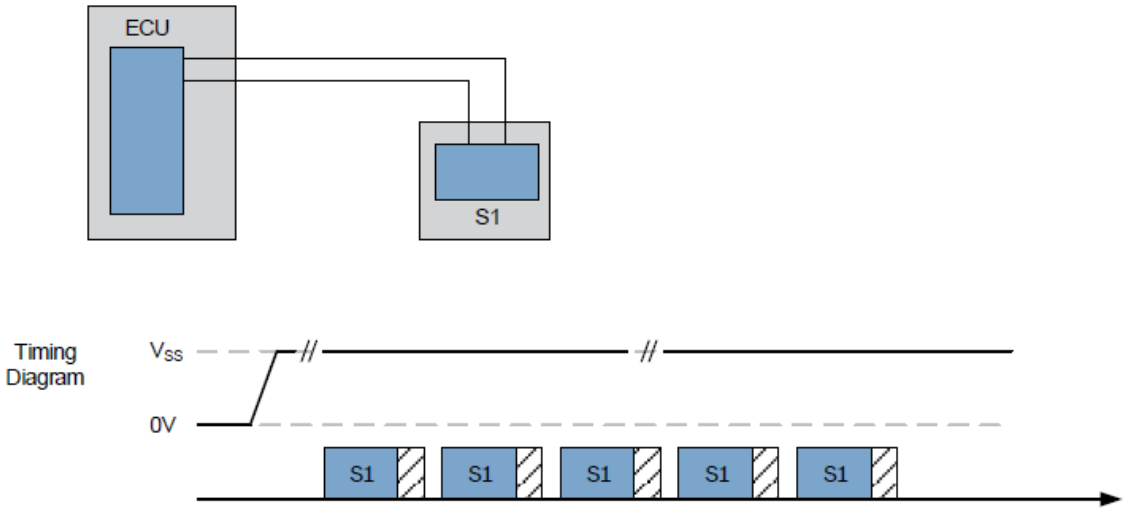

Abbildung 2-22: Asynchrone Betriebsart bei PSI5 [20]

Bei der zweiten Betriebsart handelt es sich um den synchronen Betriebsmodus. Hier kommunizieren die Sensoren in einer parallelen, universalen oder „Daisy-Chain“ Buskonfiguration mit Hilfe eines Zeitmultiplexverfahrens auf einer gemeinsamen Zweitdrahtleitung. Die Sensoren werden dabei vom Empfänger (z.B. Mikrocontroller) durch Modulation der Busspannung koordiniert bzw. synchronisiert. Es handelt sich dabei allerdings um eine eingeschränkte bidirektionale Kommunikation, da sich die Kommunikation vom Empfänger zum Sensor nur auf die Synchronisation bzw. Adressierung beschränkt.

In der *Abbildung 2-23* ist der Betriebsmodus in allgemeiner Form dargestellt. Es lässt sich erkennen, dass die Sensoren erst nach dem Synchronisationsimpuls (Impuls der Versorgungsspannung) ihre Daten übermitteln. In der gezeigten Buskonfiguration müssen die Sensoren dahingehend konfiguriert werden, dass sie nur innerhalb bestimmter Zeitslots senden, damit die Sensoren nicht zeitgleich ihre Daten übermitteln. Haben alle Sensoren ihre Daten übermittelt, wird die weitere Übertragung solange pausiert, bis der Empfänger den nächsten Synchronisationsimpuls übermittelt.

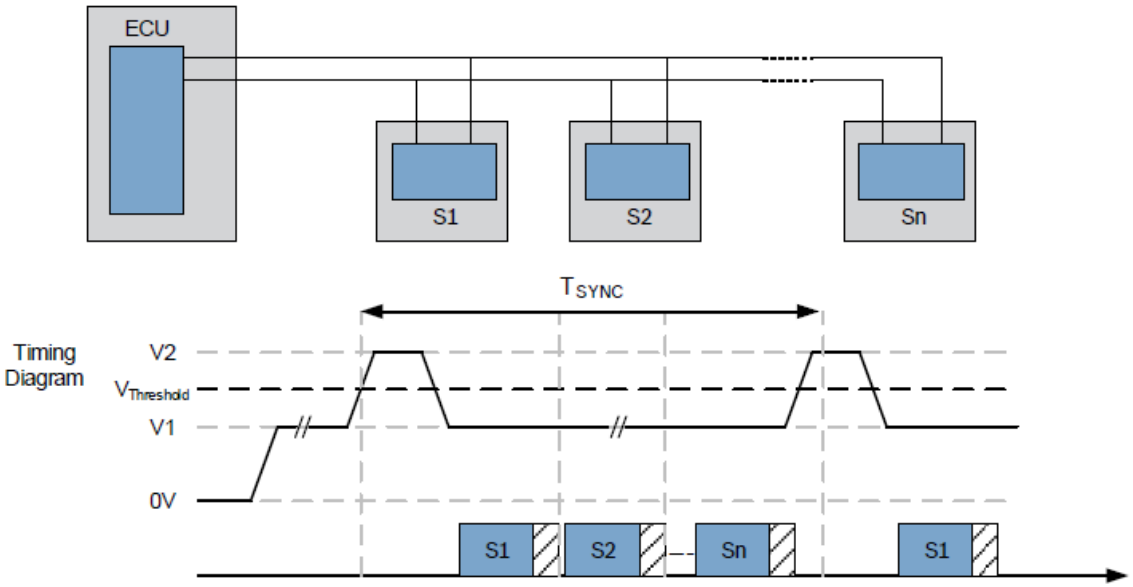

Abbildung 2-23: Synchroner Busmodus – Allgemein [20]

Bei der Verwendung des „Daisy-Chain“ Modus werden die Sensoren hintereinander geschalten, siehe *Abbildung 2-24*. Die Sensoren mit aktiver „Daisy Chain“ Funktion müssen vorab nicht konfiguriert werden, jeder Sensor wird durch den Empfänger initialisiert. Ohne Initialisierung sind bei allen Sensoren die Ausgänge gesperrt. Nachdem die Busspannung hergestellt wurde, führt der Empfänger die Initialisierung und Adressierung der Sensoren durch. Der erste Sensor erhält vom Empfänger eine entsprechende Adresse und gibt seinen Ausgang nach Abschluss der Initialisierung frei, wodurch im nächsten Schritt der zweite Sensor konfiguriert wird. Dieser Modus wurde im Rahmen der Masterarbeit nicht behandelt. Für näherer Details zur Funktionsweise wird auf den Base-Standard [20] verwiesen.

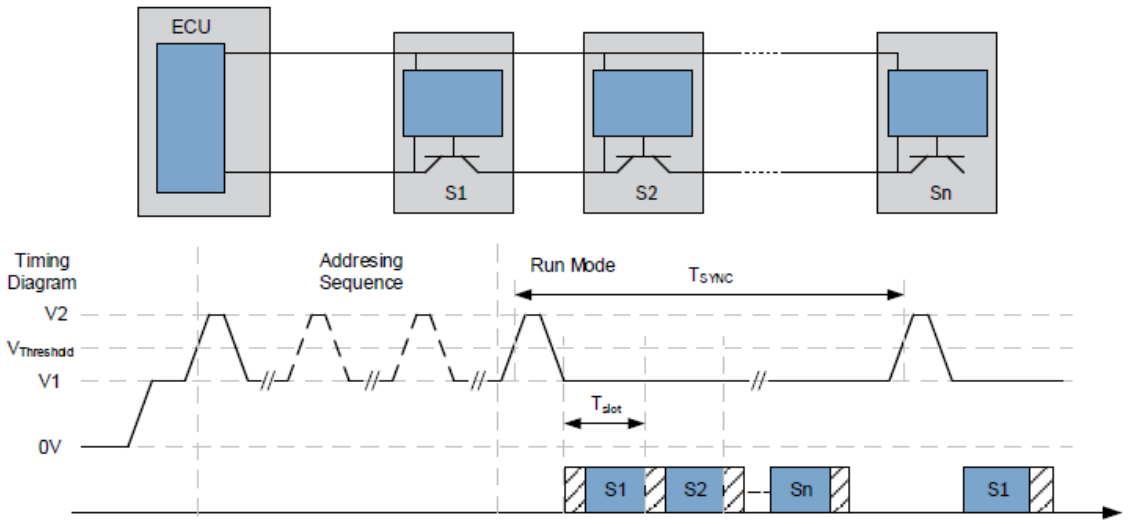

Abbildung 2-24: „Daisy Chain" Busmodus [20]

Aufgrund der unterschiedlichen Betriebsarten und Kombinationsmöglichkeiten wird ein einheitliches Schema (*Abbildung 2-25*) zur Identifizierung der verwendeten Grundkonfiguration zur Verfügung gestellt.

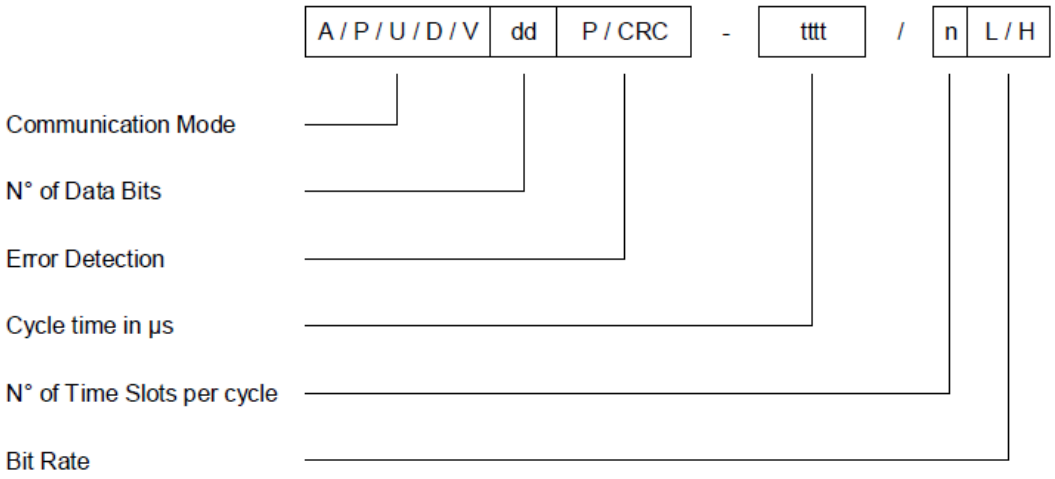

Abbildung 2-25: Bezeichnungsschema für die Systemkonfiguration [20]

- „Communication Mode"
  - A … Asynchroner Modus
  - P … Synchroner Parallelbus
  - U … Synchroner Universalbus
  - D … Synchroner „Daisy Chain" – Bus
  - V … Variabler zeitgesteuerter synchroner Modus
- „N° of Data Bits": Anzahl der Nutzdatenbits
- „Error Detection"
  - P … Paritätsbit
  - CRC … 3 Bit CRC-Prüfsumme

- „Cycle time in µs“: Gibt an in welchen Zyklus die Daten gesendet werden, z.B. der Abstand zwischen den Synchronisationsimpulsen
- „N° of Time Slots per cycle“: Anzahl der Zeitslots für einen Zyklus
- „Bit Rate“
    - L … 125 kBit/s
    - H … 189 kBit/s

Beispiel:

- „A10P-250/1L“:
  Asynchroner Betrieb mit 10 Bit, einem Paritätsbit, einem 250µs Zyklus mit einem Frame pro Zyklus und einer Bitrate von 125 kBit/s.

- „P16CRC-500/2L“:
  Synchroner Parallelbus mit 16 Bit Nutzdaten, 3 Bit CRC, einem 500µs Zyklus mit zwei Frames pro Zyklus und 125 kBit/s.

2.2.2.3 Base Standard - Physikalische Ebene

Zur Übertragung der Daten wird nur eine Zweitdrahtleitung benötigt, welche als Stromschnittstelle zur Übertragung der Sensordaten ausgeführt ist. Je nach Betriebsart kommen unterschiedliche Hardware-Konfigurationen zum Einsatz. In der *Abbildung 2-26* ist der asynchrone Betriebsmodus mit einem einzigen Sensor in einer Punkt-zu-Punkt Verbindung dargestellt. Die Werte der Bauteile können dem Base-Standard [20] entnommen werden.

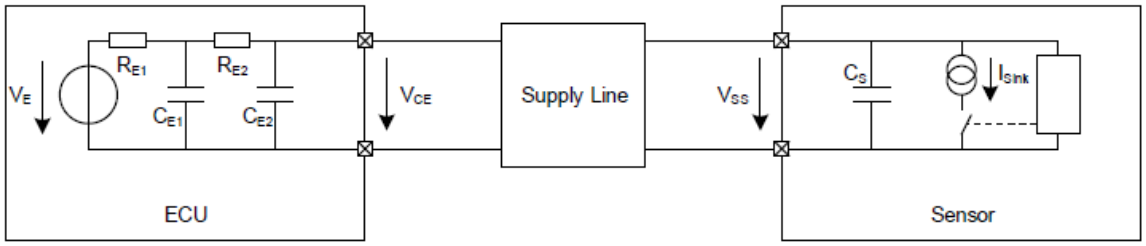

Abbildung 2-26: Asynchroner Betrieb in einer Punkt-zu-Punkt Verbindung [20]

In *Abbildung 2-27* ist der synchrone Betriebsmodus als Parallelbus dargestellt. Ein oder mehrere Sensoren werden in einer Sternschaltung an das Empfängerinterface angeschlossen. *Abbildung 2-28* zeigt hingegen den Universalbus bei dem ein oder mehrere Sensoren in einer „splice“ Konfiguration angeschlossen werden. Im Gegensatz zum Parallelbus werden die Sensoren hier nicht am Empfänger, sondern an einer beliebigen Stelle der Zweidrahtleitung verbunden. Zusätzlich gibt es beim Universalbus eine „pass-through“ Konfiguration, welche in *Abbildung 2-29* dargestellt wird.

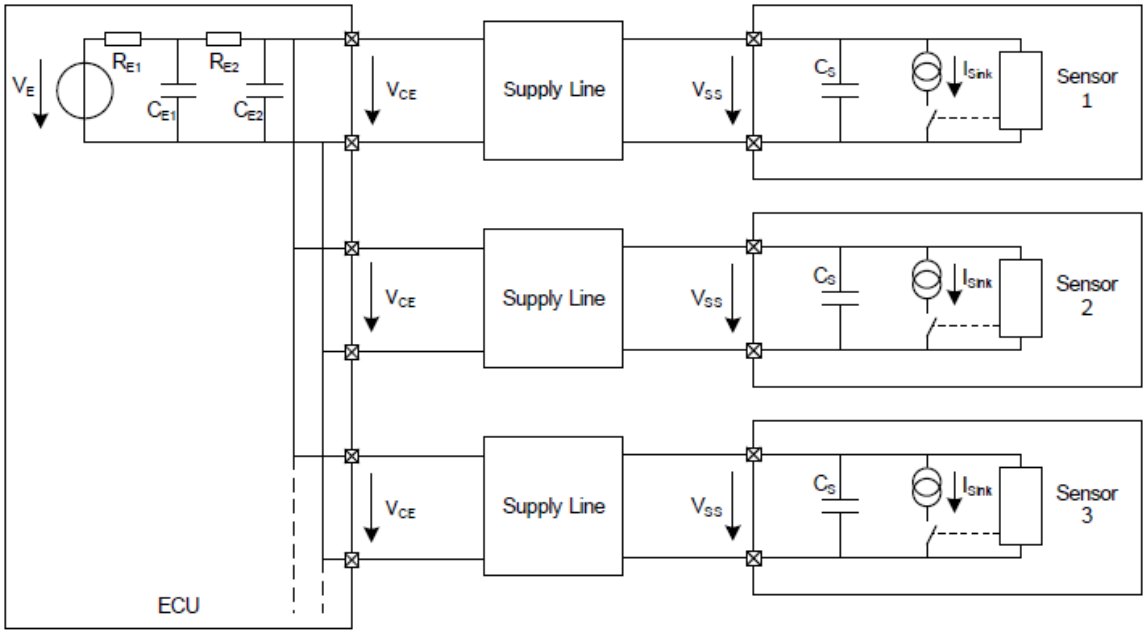

Abbildung 2-27: Synchroner Betrieb mit einer Parallelbus – Konfiguration [20]

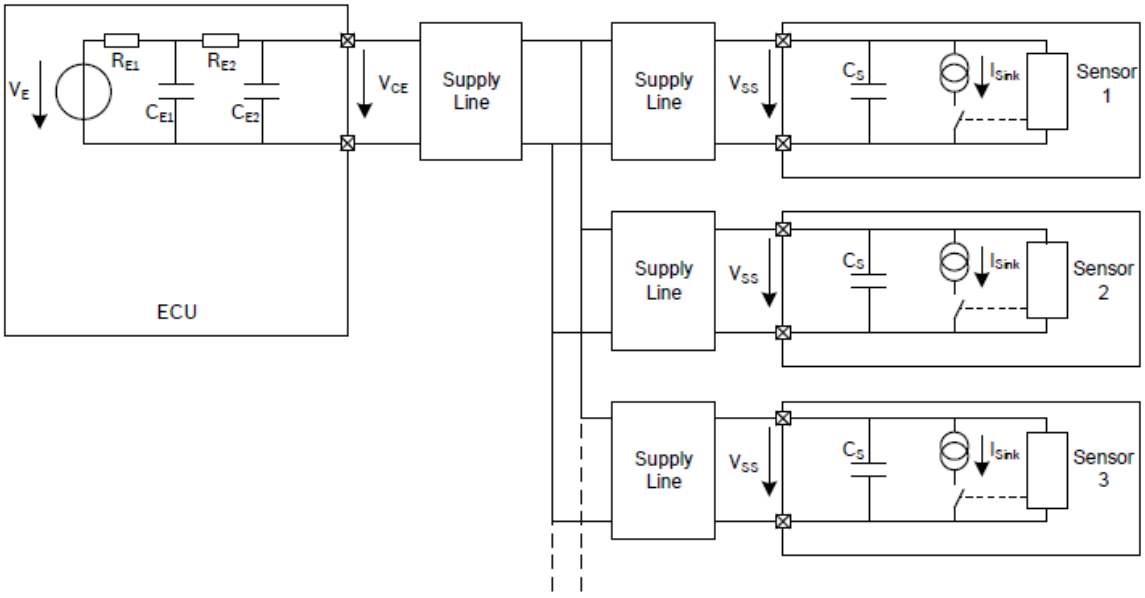

Abbildung 2-28: Synchroner Betrieb in einer Universalbus „splice“ – Konfiguration [20]

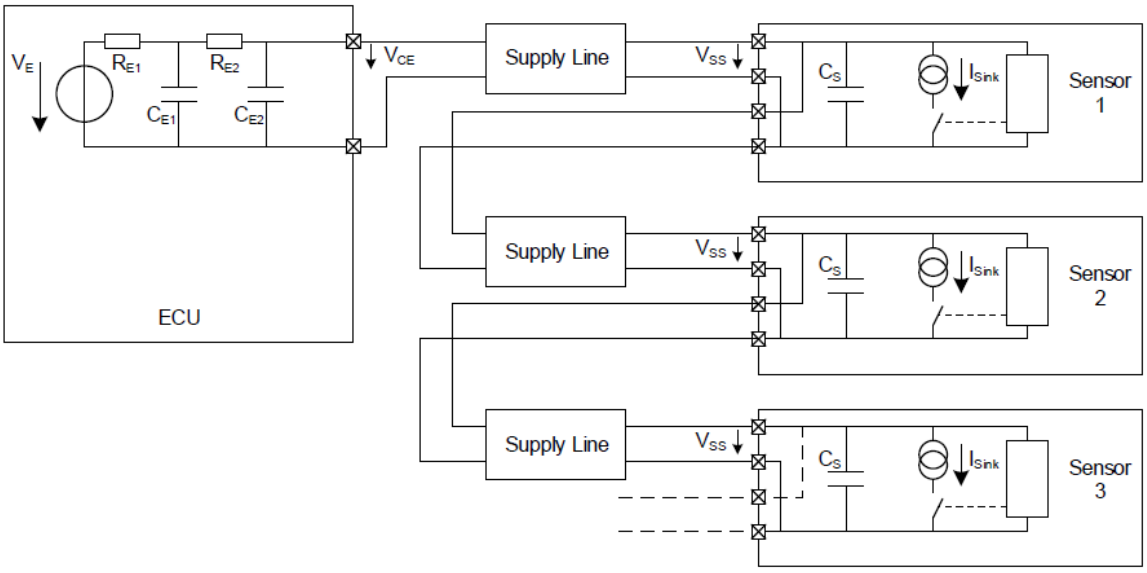

Abbildung 2-29: Synchroner Betrieb in einer Universalbus „pass through" – Konfiguration [20]

Die Versorgungsspannung am Empfänger und Sender besitzt die nachfolgenden Eigenschaften. Die Werte mit dem Index „Base" bezeichnen den statischen Ruhezustand, wenn keinerlei Kommunikation stattfindet.

- V_{SS} = 5 … 16,5 V (Standard)
- V_{SS} = 4 … 16,5 V (Low Voltage - Version)
- $V_{SS, Base}$ = 5 … 11 V (Standard)
- $V_{SS, Base}$ = 4 … 11 V (Low Voltage - Version)

- V_{CE} = 5,5 … 16,5 V (Standard)
- V_{CE} = 4,2 … 16,5 V (Low Voltage - Version)
- V_{CE} = 6,5 … 16,5 V (Increased Voltage - Version)
- $V_{CE, Base}$ = 5,7 … 11 V (Standard)
- $V_{CE, Base}$ = 4,4 … 11 V (Low Voltage - Version)
- $V_{CE, Base}$ = 6,7 … 11 V (Increased Voltage - Version)

Die Übertragung der Sensordaten an den Empfänger (Mikrocontroller) erfolgt durch eine Strommodulation an der Zweidrahtleitung mit Hilfe einer Manchester-Codierung. Wie in *Abbildung 2-30* dargestellt, erfolgt die Strommodulation indem der Ruhestrom ($I_{S, Low}$), welcher den „Low"-Pegel repräsentiert, um ΔI_S moduliert wird. Bei dem resultierenden Strom handelt es sich um den „High"-Pegel: $I_{S,High} = I_{S,Low} + \Delta I_S$. Bei der Manchester-Codierung

wird eine „Return to Zero“ Codierung verwendet. Eine logische „0“ wird durch eine steigende Flanke und eine logische „1“ durch eine fallende Flanke in der Mitte der Bit-Zeit T_{Bit} realisiert. Die Bit-Zeit T_{Bit} besitzt eine typische Zeitdauer von 8µs, was einer Datenrate von 125 kBit/s ergibt. Zusätzlich gibt es eine schnellere Übertragungsvariante mit einer Bit-Zeit von 5.3µs und einer Datenrate von 189 kBit/s.

Der Parameter $I_{S,LOW}$ liegt im Bereich von 4 bis 19mA bzw. in einer „Extended Current“ Variante im Bereich von 4 bis 35mA. Die Strommodulation ΔI_S erfolgt mit 26mA bzw. in einer „Low Power“ Variante mit 13mA.

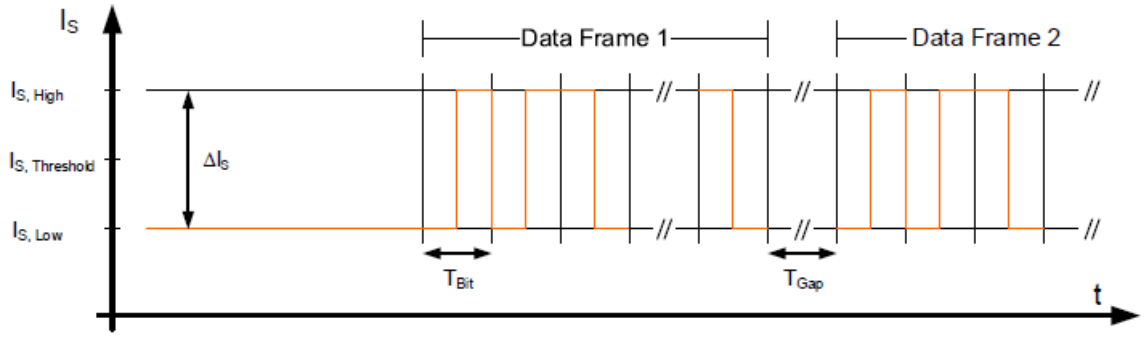

Abbildung 2-30: Strommodulation und Manchester-Codierung für die Datenübertragung [20]

Zwischen den einzelnen Frames (asynchron oder synchron) muss eine Pausenzeit von $T_{Gap} > T_{Bit}$ eingehalten werden, in der keine Kommunikation stattfindet. Dadurch kann der Empfänger das Ende von Frame 1 und den Beginn von Frame 2 erkennen: $T_{Gap} > 8{,}4\mu s$ bei $T_{Bit} = 8\mu s$ oder $T_{Gap} > 5{,}6\mu s$ bei $T_{Bit} = 5{,}3\mu s$.

Abbildung 2-31 zeigt ein Beispiel für eine Manchester-Codierte Datenübertragung mit 10 Bits Nutzdaten (Region A) und einem Paritätsbit. In diesem Beispiel lässt sich sehr gut die Bedeutung der Bits aufgrund der fallenden oder steigenden Flanken erkennen. Es ist dabei zu beachten achten, dass die Bits vom LSB zum MSB übertragen werden.

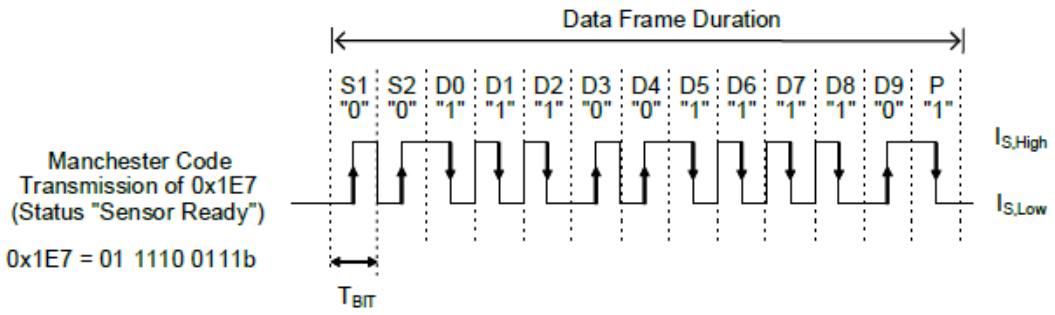

Abbildung 2-31: Beispiel für eine PSI5-Datenübertragung [20]

Wie bereits im vorherigen Kapitel *2.2.2.2* erläutert, wird im synchronen Betrieb ein Synchronisationsimpuls benötigt um die Übermittlung der verschiedenen Sensordaten zu koordinieren. Es wird dabei zwischen einem kurzen und langen Synchronisationsimpuls unterschieden, welche in *Abbildung 2-32* dargestellt sind. Erst wenn dieser Synchronisationsimpuls erfolgreich von den Sensoren erkannt wurde, dürfen diese ihre Sensordaten an den Empfänger übermitteln.

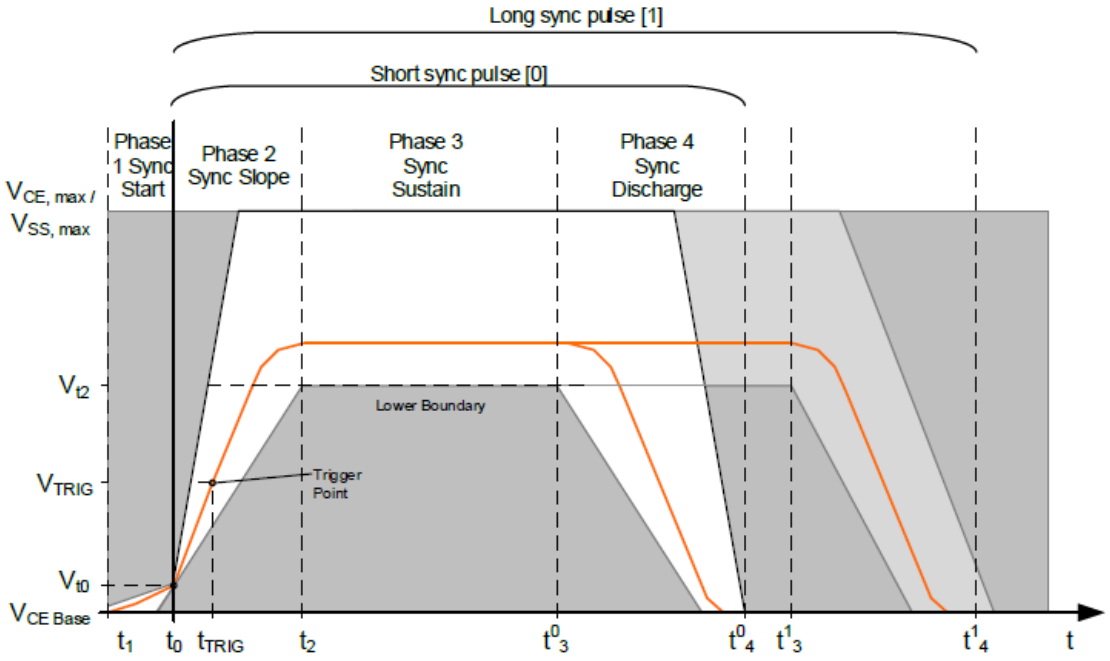

Abbildung 2-32: Parameter für den Synchronisierungsimpuls im synchronen Betrieb [20]

Für den Synchronisationsimpuls müssen dabei folgende Parameter gemäß *Abbildung 2-32* eingehalten werden:

- t_1 = -3µs
- t_0 = 0µs
- t_{TRIG} = 3.5µs (typisch)
- t_2 = max. 7µs
- t^0_3 = min. 16µs
- t^0_4 = max. 35µs
- t^1_3 = min. 43µs
- t^1_4 = max. 62µs
- $V_{t0} = V_{CE,Base} + 0{,}5V$
- V_{t2} = min. 3,5V – Standard, seit Base-Standard V2.0 auch mit min. V_{t2} = 2,5V als „reduzierter Sync-Puls“ Variante möglich

2.2.2.4 Base Standard – Anwendungsebene

Der Wertebereich der Datenregion A wird bei PSI5 in drei Bereiche unterteilt. In der *Tabelle 2-16* ist der Wertebereich für eine 10 Bit Übertragung dargestellt. Der Messwert des Sensors wird im Bereich von -480 bis +480 ausgegeben. Der Bereich von +481 bis +511 ist für Status- und Fehler-Codes (z.B. „Sensor ready“) reserviert. Der verbleibende Bereich von -512 bis -481 wird für die Initialisierung nach der Herstellung der Betriebsspannung verwendet. Bei der Initialisierung übermittelt der Sensor dem Empfänger die jeweilige Sensorkonfiguration bzw. Sensoreigenschaften.

Wert		Beschreibung	Bereich	
Dezimal	Hex			
+511	0x1FF	Reserviert (Empfänger int. Verwendung)	Status & Error	2
. .	. .			
+500	0x1F4	Sensor defekt		
. .	. .			
+489	0x1E9	„Sensor in Service Mode“		
+488	0x1E8	„Sensor Busy“		
+487	0x1E7	„Sensor Ready“		
+486	0x1E6	„Sensor Ready but Unlocked“		
. .	. .			
+480	0x1E0	Sensor - maximaler Datenwert	Sensorsignal	1
. .	. .			
+1	0x001			
0	0x000			
-1	0x3FF			
. .	. .			
-480	0x220	Sensor – minimaler Datenwert		
-481	0x21F	Status „1111“	Initialisierungswerte	3
. .	. .			
-496	0x210	Status „0000“		

-497	0x20F	„Block ID“ 16	Block IDs	
. .	. .			
-512	0x200	„Block ID“ 1		

Tabelle 2-16: 10 Bit Wertebereich für Datenregion A [20]

Das Verhalten nach dem Einschalten des Sensors hängt davon ab, ob der serielle Datenkanal verwendet wird. Ohne seriellen Datenkanal werden während der Initialisierung verschiedene Konfigurations- und Informationsparameter gesendet, erst nach Abschluss der Initialisierung werden die Messwerte übertragen. Wenn der serielle Datenkanal verwendet wird, entfällt die Übermittlung der Initialisierungsparameter. Die Sensorparameter werden gleichzeitig mit den Messwerten übertragen. Dadurch stehen die Sensorsignale wesentlich schneller zur Verfügung als bei der Initialisierungsvariante.

In *Abbildung 2-33* ist das Startverhalten mit den drei unterschiedlichen Initialisierungsphasen dargestellt. In der ersten Phase erfolgt keine Datenübertragung, der Sensor führt die interne Initialisierung seiner Systemfunktionen durch. Mit der zweiten Phase beginnt der Sensor seine internen Selbsttests und sendet die entsprechenden Sensorparameter. Je nach Ergebnis des internen Selbsttests wird in der dritten Phase „Sensor Ready“ oder „Sensor defect“ gesendet. Nach der dritten Phase beginnt die Übermittlung der Sensorsignale. Das genaue zeitliche Verhalten ist in den jeweiligen Substandards spezifiziert.

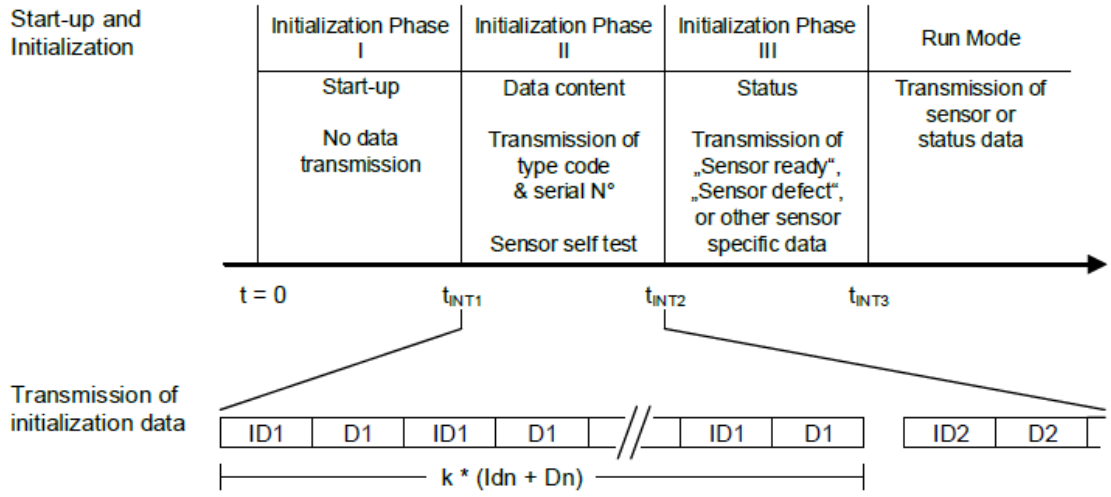

Abbildung 2-33: Initialisierung des Sensors [20]

Die grundlegenden Parameter aus Phase 2 sind im Base-Standard definiert, werden aber im jeweiligen Substandard durch weitere Parameter ergänzt. Diese Parameter werden mit Hilfe der Werte aus dem Wertereich 3 in *Tabelle 2-16* übermittelt. Für jeden Parameter gibt es definierte „Block IDs“ (16 IDs von 0x200 bis 0x20F) und die zugehörigen Datenwerte (4 Bit von 0x210 bis 0x21F). Zuerst wird zur Kennzeichnung immer die jeweilige „Block ID“ und anschließend der zugehörige Datenwert übermittelt. In der *Abbildung 2-34* ist das Prinzip dieser Übermittlung dargestellt. Der Parameter „k“ gibt an, wie oft die jeweiligen IDs und Datenwerte wiederholt werden, bevor mit der nächsten ID fortgefahren wird. Dieser Parameter wird ebenfalls in den Substandards festgelegt. Wenn die Anzahl der IDs überschritten wird, werden die IDs beginnend mit 0x200 wiederholt. Die verwendeten IDs und die Interpretation der Datenwerte können in der Regel auch aus den Datenblättern des jeweiligen Herstellers entnommen werden.

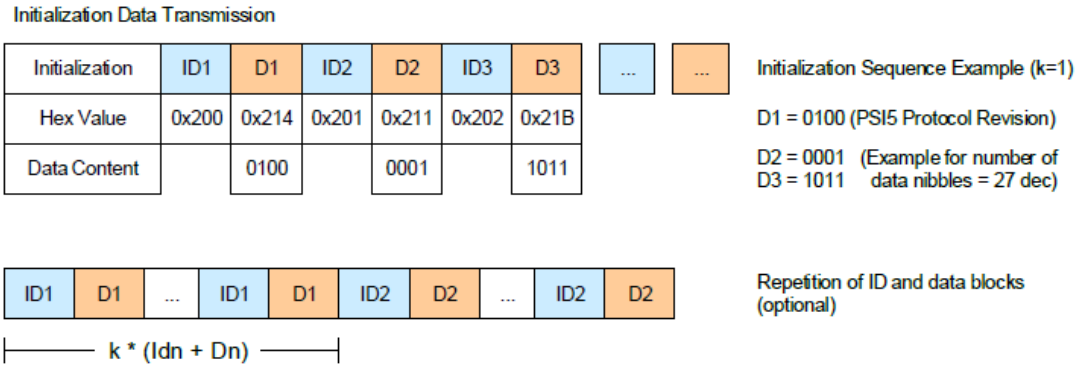

Abbildung 2-34: Initialisierungswerte in Phase 2 [20]

Im Base-Standard werden 5 Pflichtfelder mit Initialisierungswerten vorgegeben. Diese Pflichtfelder können der *Tabelle 2-17* entnommen werden. Hier muss beachtet werden, dass das Datenfeld (F1 bis F5) nur die Zugehörigkeit der Daten festlegt, für jedes Daten-Nibble D1 bis D9 muss eine „Block ID“ vergeben werden: ID1 = D1, ID2 = D2, ID3 = D3, usw. Mit dem Feld D1 wird die PSI5-Version angegeben. Bei F2 wird die Gesamtanzahl der verwendeten Daten-Nibble (D1 bis DX) angeführt. Der Hersteller-Code für F3 und der Sensortyp F4 können dem Base-Standard bzw. dem Datenblatt des Herstellers entnommen werden. Das Feld F5 beinhaltet sensorspezifische Parameter, die durch den Hersteller festgelegt werden. Zusätzlich sind in den jeweiligen Substandards weitere Datenfelder definiert.

	Header	Initialisierung		Hersteller-Code		Produkt-Code			
Datenfeld	F1	F2		F3		F4		F5	
Daten-Nibble	D1	D2	D3	D4	D5	D6	D7	D8	D9
	PSI5 Version	Anzahl Daten-Nibble		Hersteller-Code		Sensortyp		Sensorparameter	

Tabelle 2-17: Tabelle mit den Pflichtfeldern für Phase 2 [20]

Bisher wurde nur der Fall mit 10 Bit für die Datenregion A behandelt. Werden in der Datenregion A mehr als 10 Bit verwendet, dann müssen die Bereiche und Werte aus *Tabelle 2-16* nur entsprechend skaliert werden. Die zusätzlichen Bits werden als LSB-Bits an diese 10 Bit Werte angefügt.

Beispiel mit 16 Bit: Der Messwert (Bereich 1) liegt jetzt in einem Wertebereich -30720 Dezimal (0x8800 Hex) bis +30720 Dezimal (0x7800 Hex).

Für die Bereiche 2 (Initialisierung) und 3 (Status & Error) sind weiterhin nur die 10 Bit Codes gültig. Wenn mehr als 10 Bit gesendet werden, sind zur Erkennung der Codes aus Bereich 2 und 3 nur die ersten 10 MSB-Bits ausschlaggebend. Die verbleibenden Bits von Datenregion A oder B können frei verwendet werden bzw. werden in den Substandards entsprechend geregelt. Der serielle Datenkanal (M0, M1), das „Frame Control“- und „Status“-Feld sind davon nicht betroffen, diese Werte werden immer unabhängig von der Datenregion A und B übermittelt.

Beispiel mit 16 Bit Datenregion A: „Sensor Ready“ besitzt den 10 Bit Hexcode 0x1E7, skaliert auf 16 Bit ergibt es den Wert: 0x79C0. Wird nun der Wert 0x79FF empfangen, muss zuerst geprüft werden ob der Wert im Bereich 1, 2 oder 3 liegt. Da der Wert größer als 0x7800 und kleiner als 0x8000 (obere Grenze von Bereich 2) ist, teilt sich der empfangene Wert 0x79FF wie folgt auf: 10 Bit MSB = 0x1E7, 6 Bit LSB = 0x3F. Die ersten 10 Bit enthalten „Sensor Ready“, die restlichen 6 Bit wurden frei verwendet.

2.2.2.5 Substandard – Airbag

Die Inhalte dieses Kapitels wurden dem PSI5-Substandard „PSI5 Peripheral Sensor Interface for Automotive Applications, Substandard Airbag V2.3“ des „PSI5 Steering Committee“ entnommen [23]. Anderslautende Informationsquellen werden an der entsprechenden Textpassage gesondert angeführt.

Mit diesem Substandard werden für alle Airbag-Komponenten bestimmte Parameter des Base-Standards konkretisiert bzw. reglementiert. Einerseits sind nicht alle möglichen Parameter und Optionen miteinander kompatibel, und anderseits soll damit erreicht werden, dass die Implementierungen seitens der Sensor-Hersteller keine zu große Vielfalt aufweisen. Damit soll der Aufwand auf der Empfängerseite in Grenzen gehalten sowie eine möglichst große Kompatibilität zwischen den Herstellern ermöglicht werden.

Grundsätzlich kann im Airbag – Substandard der gesamte Aufbau des Übertragungsprotokolls aus Kapitel 2.2.2.1 angewendet werden. Empfohlen wird aber eine Konfiguration von 10 Bits für die Nutzdaten („Payload“) mit einem Paritätsbit. Alternativ können noch 16 Bit mit 3 CRC Bits verwendet werden. Als Betriebsarten wird der Asynchrone- und Synchrone-Modus ermöglicht. In der folgenden Liste ist eine Auswahl an möglichen Konfigurationen aufgelistet, die vollständige Liste kann dem Airbag-Substandard entnommen werden:

- A10P-250/1L
- A16CRC-500/1L
- P10P-250/1L
- P16CRC-500/2L
- D10P-500/3L

Bei der Verwendung der 10 Bit Variante kann der gesamte Wertebereich aus *Tabelle 2-16* für die Datenregion A verwendet werden. Wird die 16 Bit Variante verwendet, sind gemäß Base-Standard weiterhin die ersten 10 MSBs zur Erkennung der drei Bereiche und Codes ausschlaggebend, die verbleibenden 6 Bits können laut Base-Standard frei verwendet werden (siehe Kapitel *2.2.2.4*). Im Airbag-Substandard wird für die verbleibenden Bits (bei 16 Nutzdaten) bei den Bereichen 2 und 3 folgendes Schema festgelegt: Die verbleibenden Bits erhalten den Wert des niederwertigsten Bits aus dem 10 Bit Wertebereich.

Beispiele:

- „Block ID 1“ mit 10 Bit = 0x200 Hex → 16 Bit = 0x8000, da das LSB von 0x200 „0“ ist.
- „Block ID 2“ mit 10 Bit = 0x201 Hex → 16 Bit = 0x807F, da das LSB von 0x201 „1“ ist.

In *Tabelle 2-18* sind die wichtigsten Eckparameter für die Skalierung des 10 Bit Wertebereiches im Airbag-Substandard dargestellt. Es lässt sich erkennen, dass sich der Wertebereich 1 für

das Sensorsignal entsprechend vergrößert, im Bereich 2 und 3 aber weiterhin die 10 Bit Werte, mit der zuvor definierten Regel für die verbleibenden Bits, gültig sind.

Wert		Beschreibung	Bereich	
Dezimal	**Hex**			
+32767	0x7FFF	Reserviert (Empfänger int. Verwendung)	Status & Error	2
. .	. .			
+31231	0x79FF	„Sensor Ready“		
. .	. .			
+30720	0x7800	Sensor - maximaler Datenwert	Sensorsignal	1
. .	. .			
0	0x000			
. .	. .			
-30720	0x8800	Sensor – minimaler Datenwert		
-30721	0x87FF	Status „1111“	Initialisierungswerte	3
. .	. .			
-31744	0x8400	Status „0000“		
-31745	0x83FF	„Block ID“ 16	Block IDs	
. .	. .			
-32768	0x8000	„Block ID“ 1		

Tabelle 2-18: Wertebereich mit 16 Bit für Datenregion A im Airbag-Substandard [23]

Im Kapitel *2.2.2.4* wurden in *Abbildung 2-33* die drei unterschiedlichen Phasen der Sensorinitialisierung vorgestellt. Grundlegend ist für den Airbag-Substandard diese Initialisierungsvariante beim Sensorstart vorgesehen, da der serielle Datenkanal keinerlei Erwähnung im Substandard findet. Für die einzelnen Phasen sind die folgenden zeitlichen Abläufe und Regeln festgelegt:

Phase 1:

Die Länge dieser Phase beträgt zwischen 50 und 150ms, typischerweise werden 100ms angegeben. Während dieser Phase erfolgt keine Übermittlung von Daten, der Empfänger kann

aber bereits Synchronisationsimpulse an den Sensor senden, die aber keine Störungen hervorrufen dürfen.

Phase 2:

In dieser Phase führt der Sensor seine internen Selbsttests durch und es werden bereits die unterschiedlichen Initialisierungswerte mit der Wiederholungsrate k=4 (typisch) an den Empfänger übermittelt. Die Länge dieser Phase hängt von der Anzahl der Initialisierungswerte, der verwendeten Betriebsart und der Zykluslänge ab. Wenn der Sensor am Ende dieser Phase seine Selbsttests noch nicht abgeschlossen hat, wird die Phase entsprechend erweitert und der Sensor muss den Code „Sensor Busy“ senden. Im Kapitel *2.2.2.4* wurden in der *Tabelle 2-17* bereits die Pflichtfelder F1 bis F5 definiert. Im Substandard Airbag werden folgende zusätzliche Datenfelder definiert:

	„Application specific“											
Datenfeld	F6		F7			F8				F9		
Daten-Nibble	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	…	D32
	Sensorcode Spezifische Parameter		Sensorcode z.B. Produktversion			Produktionsdatum				Sensorinformation z.B. Seriennummer		

Tabelle 2-19: Zusätzliche Initialisierungswerte für Phase 2 der Initialisierung [23]

Eine detaillierte Beschreibung der einzelnen Felder kann im Airbag-Substandard nachgelesen werden.

Phase 3:

In Phase 3 wird das Ergebnis des internen Selbsttests übermittelt. Bei einem fehlerfreien Testergebnis wird bei Beschleunigungssensoren der Code für „Sensor Ready“ ausgegeben, wobei die Anzahl an Wiederholungen bei existierenden Sensoren sehr unterschiedlich ausfällt. Bei Drucksensoren wird nicht nur „Sensor Ready“ ausgegeben, sondern auch zusätzliche Statusinformationen wie z.B. Temperatur oder der Absolutdruck. Bei einem fehlerhaften Testergebnis wird bei beiden Arten der Code „Sensor defect“ und der jeweilige Fehlercode ständig wiederholt, bis der Sensor abgeschaltet wird. Durch die starke Variation der einzelnen Hersteller empfiehlt sich ein Blick in das jeweilige Datenblatt des Sensors.

In Phase 3 müssen zumindest 2 Nachrichten (Codes) übermittelt werden, wobei als typische Anzahl 10 Nachrichten angegeben sind. Die dritte Phase darf für maximal 200ms ausgeführt werden. Phase 3 endet sobald der erste Wert aus dem Wertebereich 1 übermittelt wird.

2.2.2.6 Substandard – Chassis und Safety Control

Die Inhalte dieses Kapitels wurden dem PSI5-Substandard „PSI5 Peripheral Sensor Interface for Automotive Applications, Substandard Chassis und Safety V2.3“ des „PSI5 Steering Committee“ entnommen [24]. Anderslautende Informationsquellen werden an der entsprechenden Textpassage gesondert angeführt.

Wie auch im Airbag-Substandard werden hier bestimmte Parameter vorgegeben. Dieser Substandard findet Anwendung für alle Systeme, die Messungen und Steuerungen zur Fortbewegung des Fahrzeugs vornehmen, z.B. Drehzahlsensoren an den Reifen, Trägheitssensoren, Bremspedalsensoren oder Sensoren für den Lenkeinschlag.

Im Gegensatz zum Airbag-Substandard wird hier eine feste Größe von 20 Bit für die Nutzdaten („Payload“) vorgegeben. Zur Fehlererkennung werden die 3 Bit CRC eingesetzt. Zusätzlich werden für die 20 Bit Nutzdaten zwei unterschiedliche Frame-Optionen angeboten:

Variante 1:

Bei der ersten Option erhält die Datenregion A 16 Bit, das „Frame-Control“-Feld 3 Bit und ein Bit wird als Statusfeld eingesetzt. Diese Variante soll für alle Sensoren mit hoher Genauigkeit eingesetzt werden. Das Statusfeld soll hier zur Kennzeichnung von Fehlerzuständen verwendet werden, um bei nicht sicherheitsrelevanten Messungen die Übertragung des Messsignals nicht durch Fehlercodes (Wertebereich 2) zu unterbrechen. Mit dem „Frame-Control“-Feld können beim synchronen Betrieb in einem Zeitslot über mehrere Zyklen unterschiedliche Signale übertragen werden, welche dann mit Hilfe der Frame-Bits unterschieden werden. Da im asynchronen Betrieb keine Zeitslots existieren, kann mit diesen Frame-Bits ebenfalls eine Unterscheidung unterschiedlicher Daten bewirkt werden.

Im Gegensatz zum Airbag-Substandard erfolgen in diesem Substandard keine Angaben über die Handhabung der verbleibenden 6 Bits aus der Datenregion für die Wertebereiche 2 und 3. Es gilt daher die Regelung aus dem Base-Standard, dass die verbleibenden 6 Bits frei verwendet werden können (siehe Kapitel *2.2.2.4*).

Variante 2:

Diese Variante wird für spezielle Airbag-Sensoranwendungen verwendet. Es sei an dieser Stelle angemerkt, dass grundsätzlich für Airbag-Sensoren der Airbag-Substandard verwendet werden muss. Bei speziellen Anwendungen die mit dem Airbag-Substandard nicht möglich sind, kann der „Chassis und Safety“ – Substandard verwendet werden. Die 20 Bit Nutzdaten werden in dieser Variante mit jeweils 10 Bit auf die Datenregion A und B aufgeteilt, dadurch können zwei verschiedene Sensorsignale in einer Übertragung gesendet werden. Für die Datenregion B kommt ebenfalls der 10 Bit Wertebereich der Datenregion A (siehe *Tabelle 2-16*) zur Anwendung. Dementsprechend muss auch für die Datenregion B die Übermittlung der Initialisierungswerte erfolgen.

Als mögliche Systemkonfigurationen bietet der „Chassis und Safety“ – Substandard folgende Optionen:

- A20CRC – 300/1L
- A20CRC – 200/1H
- P20CRC – 500/1L
- P20CRC – 500/2L (Clock-Toleranz <5%)
- P20CRC – 500/2H
- P20CRC – 500/3H (Clock-Toleranz <5%)

Wie schon im Airbag-Substandard ist auch in diesem Substandard kein serieller Datenkanal vorgesehen, weshalb die dreiphasige Initialisierungsvariante eingesetzt wird. Im Gegensatz zum Airbag-Substandard besitzt die Phase 1 eine Länge von 50 bis 200ms (typisch 100ms). Phase 2 besitzt in diesem Substandard nur ein zusätzliches Datenfeld (*Tabelle 2-20*) und die Wiederholungsrate „k“ besitzt auch hier den Wert 4. Phase 3 ist identisch mit den Regelungen im Airbag-Substandard.

	„Application specific“						
Datenfeld	F6						
Daten-Nibble	D10	D11	D12	D13	D14	D15	D16
	Sensor spezifische Informationen						

Tabelle 2-20: Datenfelder im „Chassis und Safety“ Substandard [24]

2.2.2.7 Substandard - Powertrain

Die Inhalte dieses Kapitels wurden dem PSI5-Substandard „PSI5 Peripheral Sensor Interface for Automotive Applications, Substandard Powertrain V2.3“ des „PSI5 Steering Committee“ entnommen [22]. Anderslautende Informationsquellen werden an der entsprechenden Textpassage gesondert angeführt.

Der Powertrain-Substandard wird für alle Sensoren, die für den Antrieb des Fahrzeuges zuständig sind, eingesetzt. Der Substandard sieht für diese Sensoren nur sehr wenige Einschränkungen und Regelungen vor. Für die möglichen Systemkonfigurationen gibt es nur die Vorgaben, dass eine Datenrate von 125 kBit/s sowie 3 Bit CRC verwendet werden müssen, und kein „Daisy-Chain“ – Bus angewendet werden darf. Beim Übertragungsprotokoll kann grundsätzlich der gesamte Aufbau aus Kapitel *2.2.2.1* angewendet werden. Empfohlen wird hier aber die Verwendung des seriellen Datenkanals („Messaging“ mit M0 und M1). Aus diesem Grund besitzt dieser Substandard eine geänderte Initialisierungsphase, welche in *Abbildung 2-35* dargestellt ist.

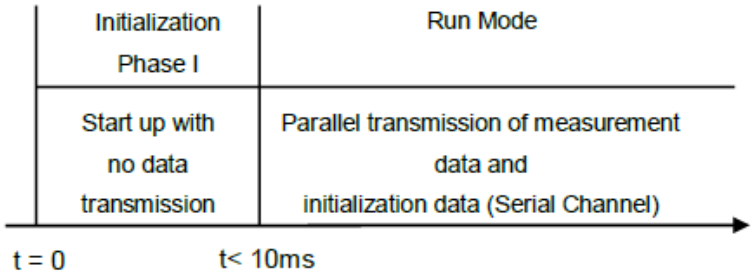

Abbildung 2-35: Initialisierung beim Powertrain-Substandard [22]

Bei diesem Substandard wird nur die Phase 1 zur Sensorinitialisierung verwendet. Die Übertragung der Initialisierungswerte entfällt und der Sensor beginnt nach max. 10ms die Messwerte zu übertragen. Die Initialisierungswerte werden gleichzeitig über den seriellen Datenkanal übertragen.

Wie bereits in Kapitel *2.2.2.1* erwähnt, ist der gesamte Aufbau des seriellen Datenkanals identisch zu den Definitionen der „Enhanced Serial Messages“ im SENT-Standard [19]. Im Powertrain-Substandard wird lediglich auf die Verwendung der PSI5-Protokoll-Version und des Herstellercodes hingewiesen, für alle anderen Informationen wird ebenfalls auf die SENT-Norm verwiesen. Vorgeschrieben wird bei Powertrain die Verwendung der „Enhanced Serial

Messages“ mit der 8 Bit „Message ID“. Der Herstellercode wird bei SENT mit der Message-ID „0x05“ übermittelt, dafür werden bei Powertrain die Zuordnungen aus „Table 30“ des PSI5-Base-Standards eingesetzt. Da bei PSI5 die Herstellercodes nur 8 Bit betragen, müssen die Codes auf die 12 Bit Daten der „Enhanced Serial Messages“ angepasst werden. Dazu wird der Herstellercode mit dem LSB startend eingetragen und die restlichen 4 Bits werden als MSBs mit dem Wert „0“ aufgefüllt. Für die Protokollversion werden die Werte aus *Tabelle 2-21* verwendet.

12 Bit Code	Beschreibung
0x000	Nicht spezifiziert
0x001 – 0x0FF	Reserviert für SENT-Protokollversion
0x100	PSI5 2.0
0x101	PSI5 2.1
0x102	PSI5 2.2
0x103	PSI5 2.3
0x104 – 0xFFE	Reserviert
0xFFF	Nicht spezifiziert

Tabelle 2-21: Werte für die PSI5-Protokoll-Version bei „Enhanced Serial Messages“ [22]

Wie im Base-Standard gelten auch im Powertrain-Substandard die Regelungen für den 10 Bit Wertebereich (siehe Kapitel *2.2.2.4* bzw. *Tabelle 2-16*). Werden mehr als 10 Bit verwendet, gilt die Regel, dass der Wertebereich entsprechend skaliert und für die Bereiche 2 und 3 die ersten 10 MSB ausschlaggebend sind. Beim Powertrain-Substandard können allerdings die verbleibenden Bits nicht frei verwendet werden, sondern müssen den Wert „0“ besitzen.

Die letzte erwähnenswerte Regelung für den Powertrain-Substandard betrifft die Datenregion B. Für diese gibt es zwei Varianten zur Auswahl:

Die Variante 1 wird als „Transparent Mode“ bezeichnet, in der alle Bits frei verwendet werden können. Als Beispiel wird die Verwendung eines Zählers angegeben.

Bei Variante 2 „Measurement Data Mode“ wird die Datenregion B zur Übermittlung von Messwerten verwendet. Acht Werte aus dem gesamten Wertebereich der Region B werden zur Anzeige von Fehlerzuständen, Initialisierungsnachrichten und spezifischen Parametern, die durch den Hersteller festgelegt werden können, verwendet. Diese Variante wurde im Zuge

des technischen Teils der Masterarbeit nicht umgesetzt, weshalb für weitere Details auf den Powertrain-Substandard verwiesen wird.

3. Hardware

Die Umsetzung der SENT- und PSI5-Schnittstelle erfolgte durch eine eigens entwickelte Platine. Zur Unterstützung diente dabei als zentrales Element das Entwicklerboard „STM32F412GDISCOVERY“ der Firma STMicroelectronics. Solche Entwicklerboards bieten neben dem eigentlichen Mikrocontroller bereits eine Vielzahl an Bauteilen (LED’s, Display, div. Schnittstellen, etc.), wodurch die Entwicklung der eigenen Hardware schneller und kostengünstiger durchgeführt werden kann. In der *Abbildung 3-1* gibt das Blockschaltbild einen groben Überblick über die verwendeten Hardwarebestandteile. Mit Ausnahme des Entwicklerboards und der externen Geräte (strichliert dargestellt) wurden die verbleibenden Schnittstellen durch die eigens entwickelte Platine realisiert. Im Wesentlichen kann die entwickelte Platine in folgende Teile unterteilt werden:

- Kommunikationsschnittstellen (RS232, CAN1, CAN2; USB/ST-LINK sind durch das Discovery-Board realisiert)
- Sensor-Schnittstellen (PSI5, SENT, ELMOS PSI5-Transceiver mit SPI)
- Analoge Signalausgabe (DAC)
- Spannungsversorgung

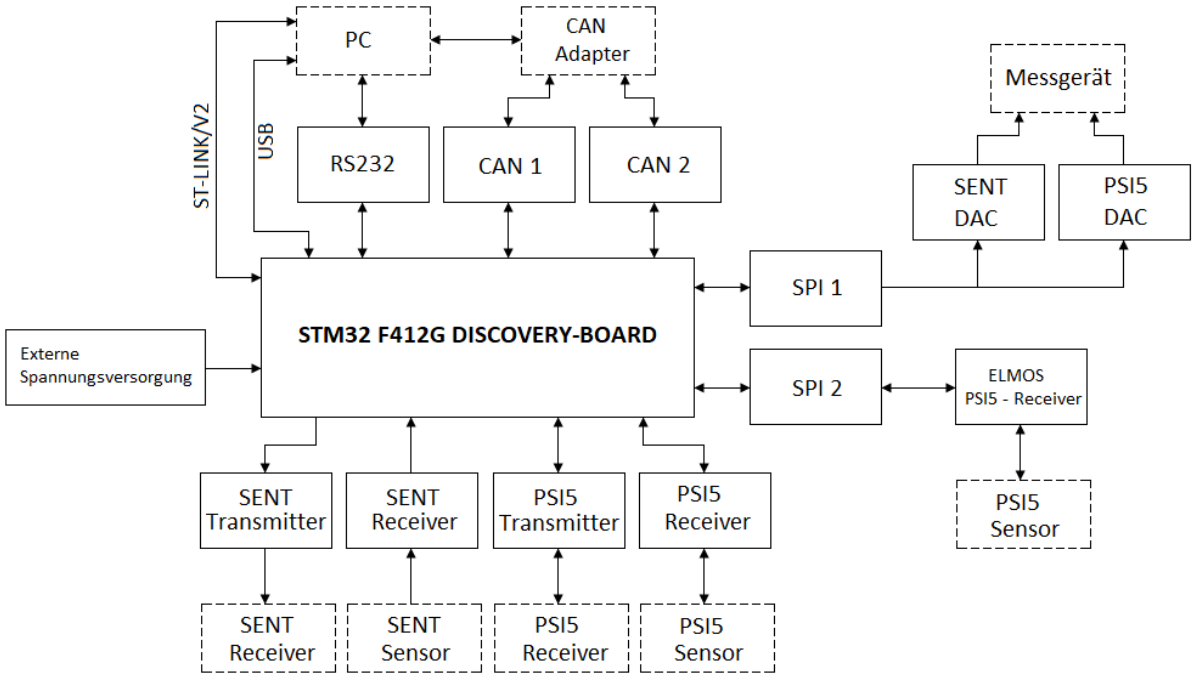

Abbildung 3-1: Blockdiagramm der verwendeten Hardware

Die Schnittstellen der Platine werden über eigene Steckverbindungen mit dem Entwicklerboard verbunden, damit diese durch den Mikrocontroller gesteuert werden können. Der detaillierte Schaltplan kann im Anhang in Kapitel *11.3* eingesehen werden. Zur Erstellung des Schaltplans und des Platinenlayouts wurde die Software „EAGLE“ in der Version 6.6.0 eingesetzt.

In den nachfolgenden Kapiteln werden die wichtigsten Funktionsgruppen aus dem Blockdiagramm der *Abbildung 3-1* näher erläutert.

3.1. Discovery-Board

Wie bereits zuvor erwähnt, wurde als zentrales Steuerelement das Entwicklerboard „STM32F412GDISCOVERY“ der Firma STMicroelectronics eingesetzt. Es folgt ein kurzer Überblick über die wichtigsten Bausteine dieses Entwicklerboards [25]:

- Mikrocontroller: STM32F412ZGT6
- 4 LED’s (Grün, Rot, Blau, Orange)
- NOR-Flash mit 128 Mbit Datenspeicher – Quad-SPI zur Datenübertragung
- USB OTG FS („USB On-the-go Fullspeed“)
- Farb-LCD Display mit 240x240 Pixel und kapazitiven Touchscreen
- ST-Link/V2-1 – Es handelt sich dabei um einen integrierten Debugger
- Joystick und RESET-Button
- Konnektor-Leiste zur externen Nutzung der Mikrocontroller-Funktionen
- Spannungsversorgung über USB bzw. ST-Link/V2-1

Das LCD-Display wird verwendet, um den Anwender über die derzeit aktiven Schnittstellen zu informieren. Der Anwender sieht somit sofort welche Kommunikationseinheiten (USB, RS232 oder CAN) und Sensorschnittstellen (SENT, PSI5 oder ELMOS PSI5-Transceiver) in Verwendung sind. Durch die vier LED’s erhält der Benutzer ebenfalls bestimmte Informationen:

- LED – Grün: Blinkt im Abstand von 1,5 Sekunden. Der Benutzer hat damit jederzeit die Kontrolle, ob das Mikrocontroller-Programm noch aktiv ist oder ob eine Störung vorliegt.

- LED – Blau: Leuchtet, wenn eine der drei Kommunikationsschnittstellen aktiv ist.

- LED – Rot: Diese LED leuchtet für ca. 1,5 Sekunden, wenn bei der Übertragung von SENT bzw. PSI5 ein Fehler aufgetreten ist.

- LED – Orange: Leuchtet für ca. 1,5 Sekunden, wenn ein Systemfehler aufgetreten ist.

Im 128 Mbit NOR-Flash werden die zuletzt eingestellten Parameter von SENT und PSI5 abgespeichert. Damit steht bei einem Neustart des Mikrocontrollers (z.B. durch Drücken des RESET-Buttons, oder bei Unterbrechungen der Spannungsversorgung) die zuletzt verwendete Konfiguration zur Verfügung. Durch Drücken des Joysticks beim Neustart wird das Laden der Konfiguration aus dem NOR-Flash übersprungen und die Standardkonfiguration geladen.

Mit dem ST-Link/V2-1 steht ein integrierter SWD-Debugger („SWD“ = „Serial Wire Debug“) für den Mikrocontroller zur Verfügung. Dabei handelt es sich um eine Schnittstelle, mit welcher einerseits der entwickelte Programmcode auf den Mikrocontroller übertragen und anderseits die Abarbeitung des Programmcodes durch den Mikrocontroller Schritt für Schritt überwacht werden kann. Zur Nutzung der SWD-Schnittstelle werden eine geeignete Entwicklungsumgebung (z.B. Keil µVision 5) sowie die passenden Treiber benötigt. Die Treiber können von der Homepage von STMicroelectronics heruntergeladen werden (Stand 13.03.2018):

http://www.st.com/content/st_com/en/products/development-tools/software-development-tools/stm32-software-development-tools/stm32-utilities/stsw-link009.html

Zusätzlich bietet der ST-Link/V2-1 einen virtuellen COM-Port (UART auf USB Adapter) und Zugriff auf den Speicher der Micro-SD-Karte [25].

3.2. Spannungsversorgung

Die 5V-Spannungsversorgung des gesamten Discovery-Boards erfolgt über den USB-Anschluss des ST-Link/V2-1. Die für den Mikrocontroller benötigten 3,3V werden durch einen am Discovery-Board vorhandenen Spannungswandler erzeugt.

Die grundlegenden Funktionsbausteine der selbst erstellten Platine benötigen ebenfalls eine 3,3 bzw. 5V Spannungsversorgung, welche vom Entwicklerboard über die Konnektorleiste zur Verfügung gestellt wird. Für den Betrieb der PSI5-Schnittstellen ist allerdings eine Spannung im Bereich von 4 bis 16,5V notwendig (siehe Kapitel *2.2.2.3*). Deshalb wird eine externe

Spannungsversorgung benötigt, die mit 15V vorgegeben ist. Das ist für diese Anwendung ausreichend, da bei der eigenen Beschaltung der minimale Pegel bei 7.5V liegt und der Synchronisationsimpuls mit 12V festgelegt wurde. Auch für den Betrieb des PSI5-Transceiver von ELMOS ist eine maximale Spannung von 15V ausreichend.

In der Regel wird der ST-Link/V2-1 Anschluss nur während der Entwicklungsphase benötigt, da die USB-Kommunikation über einen eigenen Anschluss erfolgt. Damit der Mikroprozessor bzw. das Entwicklerboard auch ohne die 5V Spannungsversorgung des ST-Link/V2-1 Anschluss betrieben werden kann, wurde zusätzlich auf der eigenen Platine ein Spannungsregler integriert. Mit diesen werden die benötigten 5V aus der externen Versorgungsspannung erzeugt. In *Abbildung 11-13* im Anhang sind einerseits der Spannungsregler und anderseits auch die zwei Konnektorleisten des Entwicklerboards mit je 50 Anschlüssen dargestellt. Diese bieten direkten Zugriff auf bestimmte Pins des Mikrocontrollers. Aufgrund von Schwierigkeiten bei der Bauteilbeschaffung für den Spannungsregler wurde eine alternative Stromversorgung implementiert (siehe *Abbildung 11-15* im Anhang).

3.3. USB, RS232, CAN

Zur Kommunikation mit dem Mikrocontroller stehen vier Schnittstellen zur Verfügung. Die USB-Schnittstelle wird direkt durch das Entwicklerboard von STMicroelectronics zur Verfügung gestellt. Auf der eigenen Platine sind zusätzlich noch eine RS232- und zwei CAN-Schnittstellen ausgeführt, welche vom Mikrocontroller gesteuert werden. Die Treiber-Bausteine in der *Abbildung 11-14* im Anhang dienen zur Signalmodulation.

Durch diese vier Schnittstellen besteht die Möglichkeit die ausgewerteten SENT/PSI5-Signale zu analysieren bzw. die Parameter für SENT/PSI5 entsprechend festzulegen.

3.4. SENT – Schnittstelle

Die Übertragung der SENT-Signale wird durch die Spannungspegel High (5V) bzw. Low (0V) realisiert (siehe Kapitel *2.2.1.3*), weshalb die dafür notwendige Hardware relativ einfach zu realisieren ist.

Die Generierung der SENT-Signale erfolgt im Mikrocontroller mit Hilfe einer entsprechenden Timer-Funktion in welcher Mikrocontroller-Pins angesteuert werden. In *Abbildung 11-15* sorgt der Pullup-Widerstand R_{26} gemeinsam mit den Transistoren T_2 und T_6 dafür, dass der SENT-Ausgang auf das Spannungslevel von 5V (High-Pegel) gezogen wird. Für den Low-Pegel steuert

das Ausgangssignal (SENT-Signal 3,3V) des Mikrocontrollers den Transistor T_1, wodurch der SENT-Ausgang auf 0V (Masse, Low-Pegel) gezogen wird. Die Übermittlung des SENT-Signals erfolgt indem der Transistor T_1 zum entsprechenden Zeitpunkt den Ausgang auf 0V zieht.

Für den Empfang wird das SENT-Signal an der Empfangsleitung entgegengenommen und einer Komparatorschaltung zugeführt. Diese sorgt dafür dass keine Störungen und Schwankungen an der Empfangsleitung ausgewertet und dem Mikrocontroller nur die definierten und angepassten High- (5V) und Low-Pegel (0V) geliefert werden. Die durch den Komparator gelieferten Signale werden anschließend an einen Mikrocontroller-Pin mit einer „Capture/Compare"-Funktion übergeben. Diese detektiert die Pegel und stellt den Zeitpunkt des Flankenwechsels in einem Interrupt zur Verfügung. Dadurch kann von der Software die Differenz zwischen den einzelnen Flanken ausgewertet werden.

Zusätzlich wurde ein Jumper JP2 (*Abbildung 11-15*) eingebaut. Damit können die selbst generierten SENT-Signale wieder eingelesen und die SENT-Übertragungen simuliert werden.

3.5. PSI5 – Schnittstelle

Die PSI5-Schnittstellen wurden im Schaltplan aufgrund des Funktionsumfangs in zwei Bereiche unterteilt. In *Abbildung 11-17* ist die Sendeeinheit der PSI5-Nachrichten und in *Abbildung 11-18* die Empfangseinheit dargestellt.

Zur Übertragung der PSI5-Signale wurden zwei Varianten für eine Stromschnittstelle realisiert. Bei beiden Varianten wird ein Ruhestrom $I_{S,Low}$ von 4mA erzeugt (siehe Kapitel *2.2.2.3*), welcher zur Erzeugung von $I_{S,HIGH}$ wahlweise um 26mA oder 13mA („Low-Power" Variante) moduliert wird. Mit Hilfe von Jumpern kann der Anwender zwischen den beiden Stromschnittstellen (Schaltungsteil mit U11 bzw. U12 - *Abbildung 11-17*) auswählen. Wird die PSI5-Schnittstelle im synchronen Betriebsmodus betrieben, muss die Übertragungseinheit den Synchronisationsimpuls (Anhebung der Busspannung um min. 2,5V bzw. 3,5V, siehe Kapitel *2.2.2.3*) erfolgreich erkennen. Zu diesem Zweck wird mit einer Komparatorschaltung (Schaltungsteil U10) die Busspannung überwacht. Allerdings können damit nur die Synchronisationsimpulse detektiert werden. Eine Erkennung ob eine ausreichende Busspannung (> 4V) vorhanden ist, ist mit dieser Schaltung nicht möglich. Für nachfolgende Adaptierungen könnte eine Messung der Busspannung mit Hilfe eines ADC-Eingangs des Mikrocontrollers realisiert werden.

Für die PSI5-Empfangsschaltung in *Abbildung 11-18* stehen ebenfalls zwei Varianten zur Auswertung des modulierten Stromes zur Verfügung. Bei beiden Varianten erzeugt der modulierte Strom an einem Strommesswiderstand (Shunt) einen Spannungsabfall. Der Shunt muss dabei so klein wie möglich sein, um den Spannungsabfall der gesamten Übertragung gering zu halten. In beiden Fällen kommt ein Widerstand mit 10 Ω zum Einsatz. Bei einer Strommodulation von ΔI_S = 26mA führt dieser zu einem Spannungsabfall von ΔU_S = 260mV. Da dieser Spannungsabfall zu gering ist, um vom Mikrocontroller entsprechend ausgewertet werden zu können, wird er mit Komparatorschaltungen in einen High- (5V) und Low-Pegel (0V) umgewandelt. Dem Mikrocontroller steht dadurch ein digitales Signal zur Auswertung an einer „Capture/Compare"-Einheit zur Verfügung. Wie schon bei SENT (Kapitel 3.4) kann diese Funktionseinheit anschließend die zeitliche Differenz zwischen den einzelnen Pegelwechseln auswerten.

Bei ersten Variante der Empfangsschaltung wird der Spannungsabfall über eine Drossel (Spule, L6) an den Komparator (U14) geführt. Mit einem Potentiometer kann die Schaltschwelle am Komparator entsprechend justiert werden. Bei der zweiten Variante wird anstelle der Drossel eine Differenzschaltung verwendet, auch hier kommt ein Potentiometer für die Schaltschwelle zum Einsatz. Ebenso wie bei der Aussendung der PSI5-Nachrichten gibt es auch für die beiden Varianten der Empfangsschaltung die Möglichkeit, mit Jumpern eine Auswahl zu treffen.

Im dritten Schaltungsteil der *Abbildung 11-18* wird die Busspannung zur Verfügung gestellt bzw. auch der Synchronisationsimpuls generiert. Mit dem Transistor T_5 kann über den Mikrocontroller (Pin PD13) die Busspannung aktiviert oder abgeschaltet werden. Die Zenerdiode D_3 liefert dabei die Busspannung von 7,5V (durch Spannungsabfälle liegt die tatsächliche Busspannung bei ca. 6,6V). Für den Synchronisationsimpuls kann der Mikrocontroller über den Pin PG9 den Transistor T9 deaktivieren, wodurch die zweite Zener-Diode D_{12} eine Spannung von 12V liefert (ca. 11,1V tatsächliche Bus-Spannung).

Der praktische Test der fertigen Platine zeigte, dass für die Aussendung der PSI5-Signale im Schaltungsteil mit U12 in *Abbildung 11-17* ein Fehler vorliegt. Durch das Verbinden dieser Übertragungseinheit mit der Empfangsschaltung (*Abbildung 11-18*) oder mit dem ELMOS-Baustein (*Abbildung 11-19*), liegt der Widerstand R_{98} = 10Ω zwischen zwei Massepunkten (GND und „PSI5-TX-GND", wobei letzteres durch die Empfangsschaltung wieder mit Masse

verbunden wird). Die Schaltung ist so nicht funktionsfähig. Diese Form der Schaltung könnte nur durch eine vollständige Potentialtrennung der Schnittstelle realisiert werden.

Bei der Empfangsschaltung mit der Drossel L6 wurde festgestellt, dass die notwendige Schaltschwelle je nach Sensor nur sehr schwer bzw. überhaupt nicht eingestellt werden kann. Durch kapazitive Einflüsse kommt es zu zeitlichen Verschiebungen bei der steigenden und fallenden Flanke der PSI5-Signale. Eine Lösung wäre die Schaltung dahingehend zu modifizieren, dass die Schaltschwelle für die fallende und steigende Flanke jeweils getrennt eingestellt werden kann. Die Auswertung der Signale mit der Differenzschaltung (U13) ist durch entsprechende Justierung des Potentiometers weitestgehend möglich. Allerdings kann es je nach Sensor vorkommen, dass die Schaltschwelle mit dem Potentiometer angepasst werden muss. Hier wäre eine Lösung ohne notwendige Anpassung der Schaltschwelle von Vorteil.

3.6. PSI5-Transceiver - E521.40

Zusätzlich zur entwickelten PSI5-Empfangsschaltung wurde auf der Platine der PSI5-Transceiver „E521.40“ der Firma Elmos Semiconductor AG integriert. Dieser Transceiver besitzt zwei unabhängige Kanäle auf denen PSI5-Signale ausgewertet werden können. Der Baustein unterstützt das PSI5-Protokoll bis zur Version 2.1, beide Datenraten von 125 kBit/s und 189 kBit/s sowie den synchronen und asynchronen Betrieb. Die Busspannung und der Synchronisationsimpuls können entweder durch den Entwickler selbst vorgegeben werden, oder durch einen internen Spannungsregler mit Ladungspumpe vom Transceiver generiert werden. Zur Kommunikation mit dem Empfänger steht wahlweise eine SPI- oder UART-Schnittstelle zur Verfügung. [26]

Die notwendige Beschaltung des ELMOS-Empfängers mit Beispielen und Empfehlungen zur Dimensionierung wird vom Hersteller im Datenblatt [26] angegeben. In *Abbildung 11-19* ist der Schaltplan zur Nutzung des Empfängers dargestellt. Zur Kommunikation wurde das SPI-Interface ausgewählt. Der Transceiver wurde so beschalten, dass nur eine externe Spannung von 15V am Sensor zur Verfügung gestellt wird. Die Erzeugung der Busspannung und des Synchronisationsimpulses erfolgt durch den internen Spannungsregler.

3.7. SENT / PSI5 – Digital-Analog-Converter

Als zusätzliche Option besitzt die Platine zwei Digital-Analog-Umsetzer „AD5620“ der Firma Analog Devices (*Abbildung 11-16*), die einen 12 Bit Digitalwert auf eine analoge Spannung von 0 bis 2,5V abbilden. Das Ausgangssignal der DACs wird zusätzlich durch einen nichtinvertierenden Verstärker auf 0 bis 5V skaliert. Damit können sehr einfach typische Sensorsignale im Bereich von 0,5V bis 4,5V ausgegeben werden.

Die Übermittlung der 12 Bit Werte erfolgt über eine SPI-Schnittstelle. Beide Digital-Analog-Umsetzer erhalten ihre Werte von einer gemeinsamen SPI-Schnittstelle des Mikrocontrollers. Mit einer „Chip-Select“ Leitung (PIN $\overline{SYNC}$) wird vom Mikrocontroller der jeweilige Baustein ausgewählt, der die Daten in Empfang nehmen soll.

Beide AD5620 haben die Aufgabe den empfangenen SENT bzw. PSI5 Wert analog auszugeben. Der AD5620 mit der Bezeichnung U_8 (*Abbildung 11-16*) wird für die SENT-Nachrichten und der Baustein U_9 für PSI5-Daten verwendet.

4. Software

In diesem Kapitel werden die wesentlichen Softwarebestandteile der Masterarbeit beschrieben. Im ersten Teil wird die Programmierung des Mikrocontrollers näher beschrieben und im zweiten Teil wird die eigens angefertigte PC-Software zur Kommunikation mit dem Mikrocontroller erklärt. Für beide Softwareprojekte werden jeweils Funktionen, Einstellungen, Einschränkungen und allgemeine Funktionsabläufe behandelt. Die detaillierte Erklärung der einzelnen Code-Abschnitte wurde im jeweiligen Programmcode dokumentiert.

4.1. Mikrocontroller

Die Programmierung des Mikrocontrollers erfolgte mit Hilfe der Software „µVision5“ der Firma „ARM Limited“ (ehemals Keil). Für diese Software steht eine kostenlose Testversion „Lite/Evaluation“ zur Verfügung (https://www.keil.com/demo/eval/arm.htm, Stand: 29.03.2018). Diese Testversion besitzt allerdings einige Einschränkungen hinsichtlich der möglichen Code-Größe (max. 32 KByte). Da die Software für dieses Projekt die Beschränkungen der „Lite/Evaluation“ überschreitet, wird zwingend eine kostenpflichtige Lizenz benötigt. Als Programmiersprache wurde für die Mikrocontroller-Software „C“ verwendet.

Die Einrichtung eines neuen Projektes mit den dafür notwendigen Einstellungen wird im Anhang in Kapitel *11.1* beschrieben.

Für das Discovery-Board (siehe Kapitel *3.1*) bzw. für die MCU-Reihe STM32F4 stehen von der Firma STMicroelectronics „Firmware-Examples“ zur Verfügung [27]. Dabei handelt es sich um Beispielprojekte in denen bestimmte Funktionen demonstriert werden. Für das Discovery-Board „STM32F412GDISCOVERY“ werden beispielsweise eine Reihe an Demoprojekten und Hilfsbibliotheken/-funktionen für die Verwendung von Display, Joystick, LED’s, NOR-Flash, usw. geliefert. Für dieses Projekt wurden die „Firmware-Examples“ in der Version 1.17.0 verwendet.

Erwähnenswert ist auch die Software „STM32CubeMX“ von STMicroelectronics (http://www.st.com/en/development-tools/stm32cubemx.html, Stand: 29.03.2018). In dieser Software kann eine grafische Konfiguration und Anpassung der µC-Parameter durchgeführt werden. Sehr hilfreich ist die Software zur „Clock-Konfiguration“, da die Software in einer grafischen Darstellung über die resultierenden „Clocks“ an den

Peripherieeinheiten informiert. Ein Benutzerhandbuch für diese Software kann im zuvor angeführten Link heruntergeladen werden.

Für die Umsetzung des Programmcodes wurden für die Systemfunktionen zur Steuerung des Mikrocontrollers sogenannte „HAL-Treiber“ (Hardware Abstraction Layer) verwendet. Diese führen weitestgehend eine Abstraktion der Mikrocontroller-Register durch, dadurch ist keine nähere Behandlung der benötigten Register während der Programmierung notwendig. Eine Portierung auf einen funktionsähnlichen Mikrocontroller wird dadurch wesentlich erleichtert. Diese „HAL-Treiber“ werden ebenfalls von STMicroelectronics im Rahmen der „Firmware-Examples“ zur Verfügung gestellt und befinden sich im Ordner: „STM32Cube_FW_F4_V1.17.0\Drivers\STM32F4xx_HAL_ Driver“. In Kapitel *11.1* wird der Import dieser „HAL-Treiber“ beschrieben. Für die speziellen Bauteile des Discovery-Boards (Display, Joystick, etc.) gibt es vorgefertigte Bibliotheken: „STM32Cube_FW_F4_V1.17.0\... …Drivers\BSP\STM32412G-Discovery“

Eine Übersicht über den Strukturaufbau und der nötigen Treiber des Mikrocontrollers kann im Anhang in Kapitel *11.1* gefunden werden.

4.1.1. Allgemeiner Funktionsablauf

Im Hauptprogramm „main.c“ werden zu Beginn alle notwendigen Variablen und Parameter definiert und mit den entsprechenden Standardwerten initialisiert. Im Anschluss erfolgt die Konfiguration und Initialisierung der HAL-Treiber bzw. Schnittstellen (Timer, Pins, SPI, CAN, USB, etc.). Nach der Initialisierung wird mit der QSPI-Schnittstelle die im NOR-Flash zuletzt hinterlegte SENT / PSI5 Konfiguration geladen. Soll die Standardkonfiguration geladen werden, muss beim Start des Mikrocontrollers der Joystick am Discovery-Board gedrückt werden. In diesem Fall wird die folgende Konfiguration geladen:

- SENT als aktive Sensorschnittstelle (gültig für Übertragung und Empfang)
    - 6 Daten-Nibble
    - Datenwert = „000000“
    - Status = „0“
    - Pause aktiviert
    - Standard-Modus
    - keine Serial Messages

- PSI5 (inkl. ELMOS-Transceiver) ist deaktiviert (gültig für TX und RX)
    - Gesamt-Bits (Nutzdaten „Payload“): 10 Bit
    - Datenregion A: 10 Bit
    - Keine Datenregion B
    - Keine Serial-Messages
    - Kein Frame-Control
    - Kein Status
    - Synchrone Betriebsart
    - Standard-Modus
    - Kanal: 0 (eigene PSI5 Empfängerschnittstelle)
    - Alle Datenwerte werden mit „0“ initialisiert

Am Display wird über den Status der jeweiligen Schnittstellen (USB, CAN, RS232, SENT, PSI5-TX, PSI5-RX, ELMOS-Transceiver) informiert (siehe auch Kapitel *3.1*), welche zum Start mit „inaktiv“ initialisiert werden.

Nachdem alle Initialisierungen abgeschlossen sind, wird in Abhängigkeit der gewählten PSI5 / SENT-Konfiguration zuerst das Display aktualisiert und dann werden die entsprechenden Timer-Funktionen und Interrupts aktiviert.

Wie in Kapitel *3.1* beschrieben, signalisiert die blinkende grüne LED (1,5s), dass die Software aktiv ist. Sollte die Software durch unvorhersehbare Umstände nicht mehr funktionsfähig sein, wäre dies durch das fehlende Blinken zu erkennen. Erkennt die Software Fehler in der Datenübertragung von SENT/PSI5 leuchtet die rote LED für 1,5s. Buffer-Überläufe, Fehler bei der Schnittstellenkommunikation von SPI, CAN, USB, etc. und sonstige Fehler wird dies durch die orange LED gekennzeichnet. Wenn eine aktive Verbindung mit dem PC (USB, RS232, CAN) besteht, wird dies einerseits am Display und anderseits durch die aktive blaue LED angezeigt.

Für alle Schnittstellen (SENT, PSI5, CAN, USB, RS232, SPI) werden Buffer verwendet, um bei Interrupts oder längeren Berechnungen keine Daten zu verlieren. Bei allen Schnittstellen werden für den Empfang von Daten Interrupts verwendet. In den Interrupt-Funktionen werden nur die Daten entgegengenommen und an die Buffer übergeben, damit die Laufzeit der Interrupts so kurz wie möglich gehalten wird. Die Verarbeitung und Auswertung der Daten

im Buffer erfolgt anschließend im Hauptprogramm „main.c“. Für die Übermittlung von Daten werden zuerst die Buffer im Hauptprogramm mit Daten versorgt. Für die eigentliche Übermittlung werden ebenfalls Interrupts verwendet, wobei dort nur noch die vorberechneten Werte aus den Buffern übergeben werden. Somit wird die Laufzeit auf ein Minimum reduziert. Beispielweise werden für die Übermittlung von SENT- und PSI5-Nachrichten die Daten vorab im Hauptprogramm berechnet, die Generierung der Signalflanken erfolgt in den Interrupts anhand der vorberechneten Bufferdaten.

Die Änderung der Systemparameter (Aktive Schnittstelle bzw. die SENT und PSI5-Konfiguration) erfolgt nur durch die entsprechende PC-Software für USB, RS232 oder CAN. Wurde eine neue Konfiguration erfolgreich empfangen, stehen für SENT und PSI5 „Active“ Variablen zur Verfügung. Besitzen diese Variablen den Wert „1“ erkennt die Software, dass eine Konfigurationsänderung vorliegt und stoppt alle Timer und Interrupts für die SENT oder PSI5 Schnittstellen. Anschließend werden alle Variablen und Buffer, welche die Übertragung / den Empfang der PSI5 bzw. SENT-Nachrichten koordinieren, wieder zurückgesetzt. Der „Active“ Parameter wird auf den Wert „2“ gesetzt. Die neuen SENT / PSI5 – Parameter werden zuerst im NOR-Flash gespeichert und danach erfolgt eine Aktualisierung des Displays. Wichtig ist, dass dieses Display-Update vor dem Beginn der Datenübertragung erfolgen muss, da der Vorgang mehrere Millisekunden benötigt und eventuell die Datenübertragungen beeinflusst. Abschließend werden die Timer und Interrupts der jeweiligen Schnittstellen aktiviert. Für die Übertragung von SENT- und PSI5-Nachrichten werden zuerst die jeweiligen Buffer vollständig mit Daten befüllt, erst anschließend wird mit der Übertragung begonnen.

In den nachfolgenden Unterkapiteln werden die Besonderheiten der wichtigsten Schnittstellen besprochen.

4.1.2. Timer

In Tabelle *Tabelle 4-1* ist eine Übersicht der verwendeten Timer inklusive der gewählten Einstellungen zu finden. Einige Timer werden für mehrere Funktionen verwendet. So wird beispielsweise der Timer 14 verwendet, um am ELMOS PSI5-Transceiver zuerst die Register und Einstellungen zurückzusetzen, anschließend werden die Registerwerte neu festgelegt und regelmäßig die Befehle für Synchronisationsimpulse und Datenabfragen vorbereitet. Die Taktfrequenzen und Zeitdauern können bei Mehrfachverwendung der Timer ggf. angepasst

werden. Die Neukonfiguration der Timer ist an den entsprechenden Stellen im Programmcode dokumentiert.

Timer	Taktfrequenz	Periode	Beschreibung
Timer 1	50 MHz	65536	SENT-TX
Timer 2	4 MHz	65536	PSI5-TX - Input Capture (fallende und steigende Flanke) zur Kennung von Synchronisationsimpulsen
Timer 3	4 MHz	65536	SENT-RX – Input Capture (fallende Flanke)
Timer 4	50 MHz	65536	PSI5-RX – Input Capture (fallende und steigende Flanke)
Timer 5	100 kHz	9000	PSI5-TX – Koordination für den asynchronen Busbetrieb
Timer 6	10 kHz	15000	LED-Timer
Timer 7	1 MHz	1	Auslösen der ELMOS-SPI-Interrupts
Timer 8	50 MHz	65536	PSI5-TX
Timer 9	100 kHz	99	PSI5-RX – Erzeugung des Synchronisationsimpulses
Timer 14	100 kHz	100	Koordination zur Steuerung der ELMOS-Befehle

Tabelle 4-1: Übersicht der verwendeten Timer

4.1.3. Interrupts

Die richtige Konfiguration der Interrupt-Prioritäten ist von entscheidender Bedeutung. Zeitkritische Funktionen, beispielsweise das Aussenden der SENT- und PSI5-Nachrichten, benötigen die höchste Priorität damit die Signale möglichst genau generiert werden. Ebenso muss die SPI-Kommunikation mit dem ELMOS-Transceiver die höchste Priorität besitzen, da sonst die SPI-Übertragung unterbrochen wird und möglicherweise am ELMOS-Transceiver bereits empfangene PSI5-Nachrichten überschrieben wurden. Die Übermittlung der ausgewerteten Daten über USB / RS232 / CAN benötigt hingegen keine hohe Priorität.

Bei der Cortex-M4 Reihe, zu der der STM32F412ZG gehört, sind insgesamt 16 Prioritäten „Preemption Priority“ möglich. Je kleiner der angegebene Wert umso größer ist die resultierende Priorität, demnach besitzt der Wert „0“ die höchste und der Wert „15“ die niedrigste Priorität. Innerhalb dieser 16 Prioritäten stehen noch sogenannte „Sub-Prioritäten“ zur Verfügung, mit denen eine Reihung innerhalb einer „Preemption Priority“ durchgeführt werden kann. Auch hier bedeutet ein niedriger Wert eine hohe Priorität. Tritt ein Interrupt auf, welcher dieselbe Priorität wie der aktive Interrupt besitzt, kommt es zu keiner Unterbrechung. Der aktuelle Interrupt wird abgearbeitet und anschließend erst der zweite

Interrupt mit derselben Priorität. Allerdings stehen insgesamt nur 4 Bit für die Vergabe von Prioritäten zur Verfügung, somit können nicht 16 Prioritäten und gleichzeitig 16 Sub-Prioritäten verwenden werden. Mit der Gruppenpriorität „PriorityGroup“ wird das Verhältnis von „Preemption Priority“ zu „Sub-Prioritäten“ angegeben. Details für die möglichen Prioritäten können im „Programming manual“ gefunden werden. [28]

Für dieses Projekt wurde die „PriorityGroup“ mit dem Wert „0“ ausgewählt, das heißt es stehen nur „Preemption Priority“ mit 16 Prioritäten zur Verfügung.

Interrupt	Preemption Priority	Beschreibung
TIM1_UP_TIM10_IRQn	0	Timer 1 Interrupt
TIM8_UP_TIM13_IRQn	0	Timer 8 Interrupt
SPI2_IRQn	0	SPI - ELMOS
TIM1_BRK_TIM9_IRQn	0	Timer 9 Interrupt
TIM8_TRG_COM_TIM14_IRQn	0	Timer 14 Interrupt
TIM2_IRQn	0	Timer 2 Interrupt
TIM3_IRQn	0	Timer 3 Interrupt
TIM4_IRQn	0	Timer 4 Interrupt
TIM5_IRQn	0	Timer 5 Interrupt
TIM7_IRQn	0	Timer 7 Interrupt
DMA1_Stream1_IRQn	1	USART3-RX – DMA Stream
OTG_FS_IRQn	13	USB Interrupt
USART3_IRQn	13	USART3 – RS232
CAN1_RX0_IRQn	13	CAN1-RX Interrupt
CAN1_TX_IRQn	13	CAN1-TX Interrupt
CAN2_RX0_IRQn	13	CAN2-RX Interrupt
CAN2_TX_IRQn	13	CAN2-TX Interrupt
SPI1_IRQn	14	SPI – AD5620
TIM6_IRQn	15	Timer 6 Interrupt

Tabelle 4-2: Übersicht der verwendeten Interrupts sortiert nach Priorität

4.1.4. SENT

Die Software der SENT-Schnittstelle unterstützt für den Empfang bzw. für die Übermittlung von SENT-Nachrichten alle grundlegenden Funktionen des SENT-Protokolls aus dem Kapitel *2.2.1*. Es besteht die Möglichkeit die Anzahl der Nibble anzupassen und beliebige Datenwerte einzustellen. Die Zeitbasis „tick“ wird mit dem Standardwert von 3µs vorgegeben. Die Aussendung kann wahlweise mit oder ohne Pause-Impuls erfolgen. Entscheidet man sich für

den Pause-Impuls, so wird jede SENT-Botschaft, unabhängig von der Anzahl der Nibble bzw. Datenwerte, mit einer Dauer von 1ms übermittelt. Beim Empfang kann die Pause eine beliebige Länge betragen, eine Überprüfung auf Einhaltung der maximalen Grenze erfolgt nicht. Die Software unterstützt auch den Versand bzw. den Empfang von „Serial Messages". Es sind sowohl „Short Serial Messages" als auch „Enhanced Serial Messages" mit beiden Konfigurationsarten (12 oder 16 Bit Daten) verfügbar. Bei den „Serial Messages" besteht allerdings die Einschränkung, dass immer nur eine „Message ID" übertragen werden kann. Die Übermittlung eines ganzen „Sets" an IDs, wie es *Tabelle 2-14* dargestellt wird, ist nicht möglich.

An dieser Stelle sei auch angemerkt, dass die Konfigurationsparameter für SENT nur bei der Konfiguration über USB, RS232 und CAN (siehe Kapitel *4.1.6*) auf korrekte Eingaben überprüft werden. Bei der Übertragung bzw. beim Empfang erfolgt keine Überprüfung der gewählten Parameter. Bei einer Änderung der Parameter im Programmcode muss deshalb darauf geachtet werden, dass diese nur innerhalb der für SENT zulässigen Grenzen festgelegt werden. Es wird dringend empfohlen die Parameter ausschließlich über die drei zuvor genannten Schnittstellen einzustellen.

Für die SENT-Kommunikation stehen unterschiedliche Betriebsmodi zur Verfügung. In *Tabelle 4-3* sind diese mit der entsprechenden Beschreibung aufgelistet. Im allgemeinen Modus können alle verfügbaren SENT-Parameter ohne Einschränkung gewählt werden. Sowohl die Bits 0 und 1 des Status- und Kommunikations-Nibble (*Tabelle 2-9*) können frei verwendet werden, als auch die Bits 3 und 4 unter der Vorrausetzung, dass keine „Serial Messages" aktiviert sind. Bei den beiden Applikationsbeispielen werden alle Vorgaben laut Kapitel *2.2.1.4* eingehalten. Für den Sensor „TLE4998S3" ist eine spezielle Berechnungsmethode hinsichtlich der CRC-Prüfsumme notwendig (siehe Kapitel *5.1.4*), weshalb für diesen Sensor ein eigener Modus für den Empfang zur Verfügung steht. Da diese Variante der CRC-Berechnung den Vorgaben der SENT-Norm widerspricht, wird eine Aussendung in dieser Form nicht unterstützt. Bei den sieben Simulationsbeispielen sind diverse Fehler in den Übertragungen eingebaut. Es wird empfohlen für diese Simulationen nur mit der eigenen Hardware (Jumper JP2 in *Abbildung 11-15* muss geschlossen werden) und Software zu arbeiten. So können die Übertragungen entsprechend analysiert und die eingebauten Fehler gefunden werden.

Modus	Senderichtung	Beschreibung
Allgemein	RX + TX	Verwendung aller verfügbaren Parameter ohne Einschränkung
Throttle Position	RX + TX	Applikationsbeispiel laut Kapitel *2.2.1.4*
Single Secure	RX + TX	Applikationsbeispiel laut Kapitel *2.2.1.4*
TLE4998S3	RX	Spezielle CRC-Berechnung siehe Kapitel *5.1.4*
Simulation 1	TX	Fehlersimulation: Zeitbasis ergibt 3,5µs, Nibble aber mit 3µs Zeitbasis übertragen.
Simulation 2	TX	Fehlersimulation: DN4 besitzt nur 11 „ticks“
Simulation 3	TX	Fehlersimulation: Synchronisationsimpuls doppelt
Simulation 4	TX	Fehlersimulation: Single Secure – DN6 besitzt nicht den inversen Wert von DN1, Counter überspringt bei jeder 10. Nachricht einen Zähler
Simulation 5	TX	Fehlersimulation: Throttle-Position – bei jeder 15. Nachricht wird im Status ein Fehler für beide Sensoren angezeigt. Zusätzlich ist bei Sensor 2 im Fehlerzustand der Wert ungleich 0.
Simulation 6	TX	Fehlersimulation: Sensor sendet abwechselnd „1E4“ und „1E5“ für die CRC-Prüfsumme, bei identischen Datenwert. Korrekt ist „1E4“.
Simulation 7	TX	Fehlersimulation: „Enhanced Serial Message“ – Bit 3 in Nachricht 13 ist nicht 0

Tabelle 4-3: Modi für die SENT-Kommunikation

4.1.4.1 SENT – Übermittlung

Wie bereits im allgemeinen Funktionsablauf im Kapitel *4.1.1* beschrieben, wird beim erstmaligen Start bzw. nach einer Konfigurationsänderung über USB/RS232/CAN der Buffer für die Aussendung mit den vorberechneten Werten gefüllt. Bei der Vorberechnung werden zuerst die Datenwerte (Status-Nibble bei „Serial Messages“, CRC-Prüfsumme, Zähler für „Secure Counter“, usw.) ermittelt und anschließend die für die Übertragung notwendigen Zeiten kalkuliert. Sobald der Buffer gefüllt ist, wird der Timer-Interrupt zur Aussendung aktiviert. Die Übertragung erfolgt indem in diesem Interrupt nur noch der Zustand des Pins (High / Low) bei jedem Aufruf invertiert wird. Der Zeitzähler des Timers wird entsprechend der vorberechneten Werte im Buffer bei jedem Interrupt neu eingestellt. Dadurch befindet sich der SENT-Ausgang der Platine für die berechneten Zeiten im jeweiligen High- oder Low-Zustand, wodurch die SENT-Übertragung zustande kommt. Die freigewordenen Plätze im Buffer werden im Hauptprogramm kontinuierlich mit den nächsten vorberechneten Zeiten aufgefüllt. Sobald der Interrupt aktiviert wurde, werden kontinuierlich die Zeiten aus dem

Buffer abgerufen, solange bis eine Konfigurationsänderung erfolgt und der Interrupt dadurch deaktiviert wird.

4.1.4.2 SENT - Empfang

Für den Empfang der SENT-Nachrichten wird der Timer 3 (siehe *Tabelle 4-1*) in einer „Capture & Compare“ Konfiguration verwendet. Da bei SENT die Zeit zwischen den fallenden Flanken ausschlaggebend ist, wurde der Timer so konfiguriert, dass nur bei einer fallenden Flanke ein Interrupt ausgelöst wird. Somit wird direkt die Zeit zwischen den fallenden Flanken gemessen. Eine Überprüfung ob mindestens 5 „ticks“ auf einem Low-Pegel liegen, erfolgt hier nicht. Es besteht aber prinzipiell die Möglichkeit, den Timer so zu konfigurieren, dass sowohl fallende als auch steigende Flanken detektiert werden. Die Auswertung müsste dann dementsprechend angepasst werden.

Wenn der Interrupt bei der fallenden Flanke ausgelöst wird, wird der aktuelle Zählerstand des Timers in einem Buffer gespeichert. Da der Zähler kontinuierlich weiterläuft, muss im Hauptprogramm „main.c“ dann jeweils der Zählerunterschied mit Hilfe des letzten Wertes berechnet werden, z.B. letzter Zählerstand = 0x100, aktueller Zählerstand = 0x200 ergibt den Wert „0x100“ zwischen der letzten und aktuellen Flanke. Dieser berechnete Zeitwert wird wiederum in einem eigenen Buffer abgelegt. Dies wird durchgeführt weil für die Berechnung der SENT-Nachrichten solange gewartet wird, bis die nötige Anzahl an Zeitwerten (Flanken) zur Berechnung einer ganzen SENT-Nachricht verfügbar sind. Beispielweise werden bei 6 Nibble insgesamt 9 Zeitwerte (Synchronisation + Status + 6 Nibble + CRC) für eine gesamte SENT-Botschaft benötigt. Sobald genügend Werte für eine SENT-Nachricht in diesem Buffer vorhanden sind, erfolgt die Überprüfung und Berechnung der Nachricht.

Bei der Berechnung wird zuerst anhand des Synchronisationsimpulses die Zeitbasis „tick“ ermittelt. Dazu wird jedes Mal die Frequenz des Timers anhand des gewählten Wertes für den „Prescaler“ berechnet. Das hat den Vorteil, dass bei einer Änderung der Frequenz des Timers keine Änderungen in der Berechnung notwendig sind, da sich die Frequenz zur Berechnung automatisch einstellt. Mit Hilfe der Frequenz des Timers wird dann der tatsächliche Zeitwert in Mikrosekunden berechnet. Zusätzlich wird überprüft ob die berechnete Zeitbasis sich innerhalb der 20% Toleranz befindet. Anschließend werden mit der Zeitbasis und der Frequenz des Timers die Anzahl der „ticks“ der einzelnen Nibble berechnet und auf Einhaltung der Grenzen (min. 12 und max. 27 „ticks“) überprüft.

Sobald alle Daten-Nibble berechnet wurden, werden diese an die jeweiligen USB / RS232 / CAN Buffer, zur Übertragung an den PC, übergeben. Erst danach wird die CRC-Prüfsumme berechnet und mit dem empfangenen Wert verglichen. Sollte die CRC-Prüfsumme abweichen, wird eine entsprechende Fehlermeldung ausgegeben. Da die Daten-Nibble aber bereits zuvor gesendet wurden, kann die Ursache für den CRC-Fehler am PC analysiert werden. Für die CRC-Berechnung wurde das in Kapitel *2.2.1.1* angesprochene Berechnungsbeispiel der SENT-Norm verwendet. Ebenso werden die spezifischen Regeln der Applikationsbeispiele „Throttle Position“ und „Single Secure“ erst im Anschluss kontrolliert.

Wie in Kapitel *5.1.4* erläutert, weicht beim Sensor „TLE4998S3“ die Berechnung der CRC-Prüfsumme von der SENT-Norm (Kapitel *2.2.1.1*) insofern ab, als zusätzlich das Status- & Kommunikations-Nibble für die Berechnung miteinbezogen wird. Deshalb gibt es den Betriebsmodus „TLE4998S3“ (*Tabelle 4-3*) für den SENT-Empfang zur Auswahl, bei welchem das Status-Nibble ebenfalls in die Berechnung einfließt, damit die Daten dieses Sensors eingelesen werden können.

Die empfangen SENT-Daten werden anschließend zur analogen Ausgabe aufbereitet. Da der „AD5620“ Baustein maximal 12 Bit unterstützt, werden maximal die ersten drei Daten-Nibble für die analoge Ausgabe benutzt. Ein Wert von „000“ entspricht dabei 2,5V am analogen SENT-Ausgang der Platine. Die Werte werden entsprechend dem Vorzeichen („signed“ Werte) addiert, d.h. bis zum Hex-Wert 0x7FF (3 Daten-Nibbe) werden die Werte addiert – die Spannung erhöht sich, ab dem Hex-Wert 0x800 werden die Werte subtrahiert – die Spannung reduziert sich von den 2,5V ausgehend. Die berechneten Werte werden einem SPI-Buffer zur Übertragung an den „AD5620“ übergeben.

Wenn „Serial Messages“ zum Einsatz kommen, werden je nach Typ 16 oder 18 Statuswerte zwischengespeichert. Sobald alle nötigen Statuswerte vorhanden sind, werden die „Short“ und „Enhanced Messages“ aus Bit 2 und 3 extrahiert und die Berechnung der Werte sowie die CRC-Prüfung durchgeführt.

4.1.5. PSI5

Bei der PSI5-Schnittstelle werden die in Kapitel *2.2.2* beschriebenen Funktionen großteils unterstützt. Es können beim Übertragungsprotokoll aus *Abbildung 2-20* alle Felder des PSI5-Frames verwendet werden. Bei der Verwendung von 10 Bit Nutzdaten wird das Paritätsbit, ab

11 Bit werden die drei CRC-Bits eingesetzt. Ebenso werden auch die „Enhanced Serial Messages“ mit 12 und 16 Bit Daten ermöglicht. Bei den Betriebsarten kann der asynchrone und synchrone Betrieb („long“ und „short“ Synchronisation) mit jeweils einem Sensor genutzt werden. Zusätzlich gibt es eine Option mit der zwei Zeitslots übermittelt bzw. empfangen werden können.

Wie schon bei SENT gilt auch hier, dass die Parameter für die PSI5-Kommunikation nur bei der Konfiguration über USB, RS232 und CAN (siehe Kapitel *4.1.6*) auf korrekte Eingaben überprüft werden. Es wird für die PSI5-Schnittstelle dringend empfohlen die Parameter ausschließlich auf diese Weise zu konfigurieren.

Neben dem allgemeinen Modus, bei welchem alle verfügbaren Parameter ohne Einschränkung genutzt werden können, gibt es für alle drei Substandards jeweils 2 Beispiele. Bei diesen handelt es sich um die in den Substandards empfohlenen Konfigurationen, damit die Besonderheiten der Substandards anhand von Praxisübungen demonstriert werden können. In einem eigenen Modus „2 Zeitslots“ werden innerhalb von einem Zyklus zwei Sensoren (Zeitslots) mit 11 Bit und „Frame Control“ simuliert. Für den Sensor „AIS1200PS“ von STMicroelectronics musste zur Auswertung ein eigener Modus konfiguriert werden, da das erste Startbit eine zu kurze Zeit aufweist. Zusätzlich gibt es für die Aussendung zwei Fehlersimulationen um, wie schon bei SENT, die Fehler zu analysieren.

Modus	**Senderichtung**	**Beschreibung**
Allgemein	RX + TX	Verwendung aller verfügbaren Parameter
Airbag 1	TX + (RX)	A10P-250/1L bzw. P10P-250/1L Parameter laut Substandard – Kapitel 2.2.2.5
Airbag 2	TX + (RX)	A16CRC-500/1L bzw. P16CRC-500/1L Parameter laut Substandard – Kapitel 2.2.2.5
Powertrain 1	TX + (RX)	125 kBit/s, 12 bis 28 Bit, Serial Messages, Asynchron oder Synchron, freie Verwendung der Felder; siehe Kapitel 2.2.2.7
Powertrain 2	TX + (RX)	125 kBit/s, 12 bis 28 Bit, Serial Messages, Asynchron oder Synchron, Region B als „Counter“; siehe Kapitel 2.2.2.7
Chassis & Safety 1	TX + (RX)	P20CRC – 500/1L, Variante 1 laut Kapitel 2.2.2.6
Chassis & Safety 2	TX + (RX)	P20CRC – 500/1L, Variante 2 laut Kapitel 2.2.2.6
2 Zeitslots	TX + (RX)	A11CRC – 500/2L, P11CRC – 500/2L, 1 Bit Frame-Control

AIS1200PS	RX	Spezielle Auswertung siehe Kapitel 5.2.3
Simulation 1	TX	Fehlersimulation: Paritätsfehler – Bit 5 invertiert zur Eingabe
Simulation 2	TX	Fehlersimulation: 12 Bit werden gesendet statt der eingestellten 11 Bit

Tabelle 4-4: Modi für die PSI5-Kommunikation

4.1.5.1 PSI5 – Übermittlung

Wie bereits bei SENT und im allgemeinen Funktionsablauf im Kapitel *4.1.1* beschrieben, wird beim erstmaligen Start bzw. nach einer Konfigurationsänderung über USB/RS232/CAN der Buffer für die Aussendung mit den vorberechneten Werten gefüllt. Bei der Vorberechnung werden zuerst die Nutzdaten („Messaging“, „Frame Control“, Status, Region B, Region A und Parität/CRC) ermittelt und anschließend die für die Übertragung notwendigen Zeiten kalkuliert.

Im Hardware-Kapitel *3.5* wurde erläutert, dass eine Erkennung des Synchronisationsimpulses möglich ist, aber keine Überwachung auf eine vorhandene Busspannung durchgeführt werden kann. Deswegen erfolgt die Aussendung entweder beim Erhalt des Synchronisationsimpulses oder im asynchronen Betrieb nach einer bestimmten Zeitdauer (90ms ohne bzw. 9ms mit „Serial Messages“). Beim ersten Aufruf bzw. nach einer Konfigurationsänderung erfolgt die Aussendung nur, wenn der Buffer vollständig gefüllt wurde (ca. 2ms bei einer Buffergröße mit 2048 Elementen). In den Substandards „Airbag“ (Kapitel *2.2.2.5*) und „Chassis & Safety“ (Kapitel *2.2.2.6*) ist für die Phase 1 eine Länge von mindestens 50ms, nach Herstellung der Busspannung, gefordert. Wenn von einem externen Empfänger ein Synchronisationsimpuls innerhalb dieser Zeitspanne gesendet wird, liefert die PSI5-Schnittstelle trotzdem die Sensordaten, die geforderten 50ms können in diesem Fall nicht eingehalten werden. Bei der eigenen PSI5-Empfangsschaltung wird der erste Synchronisationsimpuls erst nach 100ms ausgesendet, siehe Kapitel *4.1.5.2*. Es wäre zwar möglich das zeitliche Problem bei externen Empfängern ebenfalls mit einem Timer zu lösen (wie schon beim asynchronen Betrieb), aber trotzdem würde die Schnittstelle weiterhin unabhängig vom externen Empfänger agieren. Die beste Vorgehensweise wäre eine Detektion der Busspannung, beispielweise wie der Vorschlag aus dem Hardware-Kapitel *3.5* mit Hilfe eines ADCs.

Das Prinzip der Übertragung erfolgt ähnlich wie bei SENT: Im Interrupt wird der Zustand des Mikrocontroller-Pins invertiert und der Zeitzähler entsprechend der vorberechneten Werte neu eingestellt. Im Gegensatz zu SENT wird die Übertragung nicht kontinuierlich ohne Pause

durchgeführt, sondern die PSI5-Übertragung erfolgt in Form von Zyklen (synchron und asynchron). Deshalb gibt es einen zusätzlichen Buffer, der synchron mit den vorberechneten Werten läuft. Dieser Buffer besitzt einen Indikator, der den Timer nach jeder vollständigen Übertragung stoppt. Im synchronen Betrieb wird der Timer durch den Synchronisationsimpuls gestartet, im asynchronen Betrieb zeitgesteuert durch einen anderen Timer. Zur Übertragung der Daten steht derzeit nur die Standardvariante mit ΔI_s = 26mA und einer Datenrate von 125 kBit/s zur Verfügung. Für die „Low-Power" Variante müsste der Pin „PC1" statt „PC0" verwendet werden. Die Änderung der Datenrate gestaltet sich etwas aufwendiger, da einerseits die Berechnung der Zeitwerte und andererseits möglicherweise auch die Timer-Frequenz angepasst werden muss.

Im Gegensatz zu SENT steht bei PSI5 zur Berechnung der 3 Bit CRC-Prüfsumme kein Implementierungsbeispiel zur Verfügung. Das Generatorpolynom lautet $x^3 + x + 1 = 0xB$ und besitzt einen Initialisierungswert von 7. Im ersten Schritt werden die Nutzdaten in Gruppen zu 3 Bit unterteilt. Danach wird beginnend mit dem Initialisierungswert (3 Bit) durch eine „XOR" Berechnung mit „bit shifting" (verschieben der Bits) für alle Bits die resultierende CRC-Prüfsumme ermittelt.

Allgemeiner Modus:

Wenn eine Übertragungsform ohne „Serial Messages" gewählt wird, werden die Parameter des Sensors in Form von Initialisierungswerten gesendet (siehe Kapitel *2.2.2.4*). Diese Initialisierungswerte werden noch vor der Aktivierung der Timer bzw. Interrupts ermittelt und an den Übertragungs-Buffer übergeben. Im Allgemeinen Modus (siehe *Tabelle 4-4*) werden für die Phase 1 folgende Initialisierungswerte verwendet:

	Header	**Initialisierung**		**Hersteller-Code**		**Produkt-Code**			
Datenfeld	F1	F2		F3		F4		F5	
Daten-Nibble	D1	D2	D3	D4	D5	D6	D7	D8	D9
Binärcode	0110	0000	1001	0110	0001	0000	0000	0000	0000
Dezimalwert	6	0	9	6	1	0	0	0	0
Beschreibung	PSI5 V2.X	Länge D1-D9		Analog Devices		Sensortyp		Sensorparameter	

Tabelle 4-5: Initialisierungswerte für den allgemeinen Modus

Da in diesem Modus jeder beliebige Sensor simuliert werden kann, sind die Werte der Felder von F4 und F5 mit „0" belegt. Für den Sensorhersteller wurde der Code von „Analog Devices"

ausgewählt, die Übertragung steht in keinem Zusammenhang mit der Firma „Analog Devices“. Es handelt sich dabei nur um ein mögliches Beispiel um einen Initialisierungswert zu demonstrieren, da ein fiktiver Code nicht mit dem PSI5-Standard verglichen werden kann. Die Initialisierungswerte werden mit der Wiederholungsrate k = 4 übermittelt. Anschließend folgt die Phase 3 in der zehnmal der Code für „Sensor ready“ (0x1E7) übertragen wird. Werden mehr als 10 Bit Nutzdaten verwendet, so wird der jeweilige Code weiterhin mit 10 Bit MSB (Kapitel *2.2.2.4*) gesendet, die verbleibenden Bits werden frei verwendet und entsprechend der Eingabe übermittelt.

Möchte man „Serial Messages“ verwenden, so werden keine Initialisierungswerte übermittelt. Als Dateninhalt wird im allgemeinen Modus bei den „Serial Messages“ die Eingabe des Benutzers verwendet, eine Initialisierungsprozedur ist hier nicht realisiert. Diese kann aber Anhand eines Beispiels im „Powertrain“ Modus analysiert werden.

„Airbag 1 und 2“:

Bei den Modi „Airbag 1“ und „Airbag 2“ kann ausschließlich die Datenregion A mit 10 bzw. 16 Bit genutzt werden, alle anderen Felder können nicht verwendet werden. Dadurch werden nur die Initialisierungswerte verwendet. Wie im Substandard „Airbag“ empfohlen werden in Phase 2 und Phase 3, bei der Verwendung von 16 Bit, die verbleibenden Bits abhängig vom LSB des 10 Bit Codes übermittelt (siehe Definitionen Kapitel *2.2.2.5*). In der Phase 2 werden folgende Initialisierungswerte für F1 bis F5 übertragen:

	Header	**Initialisierung**		**Hersteller-Code**		**Produkt-Code**			
Datenfeld	F1	F2		F3		F4		F5	
Daten-Nibble	D1	D2	D3	D4	D5	D6	D7	D8	D9
Binärcode	0110	0010	0000	0110	0001	0000	0001	0100	1000
Dezimalwert	6	2	0	6	1	0	1	4	8
Beschreibung	PSI5 V2.X	Länge D1-D32		Analog Devices		Acceleration Sensor (high g)		± 480g	

Tabelle 4-6: Initialisierungswerte F1 bis F5 im „Airbag1“ bzw. „Airbag 2“ Modus

Auch in diesen beiden Modi wurde der Code von „Analog Devices“ nur als Beispiel verwendet. Im Gegensatz zum allgemeinen Modus wird im Feld F4 ein Beschleunigungssensor mit einem Messbereich von ±480g (F5) angegeben. Zur Übersicht wurden die für den Substandard „Airbag“ zusätzlichen Felder in einer eigenen Tabelle (*Tabelle 4-7*) angeführt. Beim Feld F6 kann

der Hersteller einen beliebigen selbstdefinierten Code angeben, dieser wurde hier mit „0“ angegeben. Als Beispiel wurde für F7 die Produktversion „001“ und für F8 das Produktionsdatum „14.02.2018“ eingestellt. Bei der Seriennummer in F9 wurde ein simpler Zähler verwendet und aus Platzgründen deshalb die Binärdarstellung nicht vollständig aufgelistet.

	„Application specific“											
Datenfeld	F6		F7			F8				F9		
Daten-Nibble	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	…	D32
Binärcode	0000	0000	0000	0000	0000	0010	0100	0100	1110	0000 0001 0010 0011 …		
Dezimalwert	0	0	0	0	1	2	4	4	E	0123456789ABCD		
Beschreibung	Sensorcode Spezifische Parameter		Produktversion „001“			Produktionsdatum 18 / 02 / 14 (14.02.2018)				Seriennummer: 0123456789ABCD		

Tabelle 4-7: Initialisierungswerte F6 bis F9 im „Airbag1“ bzw. „Airbag 2“ Modus

Wie auch schon im allgemeinen Modus wird nach Abschluss der Phase 2 der Sensorcode „Sensor ready“ mit 10 Nachrichten übertragen.

„Powertrain 1 und 2“:

Grundsätzlich können im Substandard „Powertrain“ alle Felder des PSI5-Protokolls (*Abbildung 2-20*) verwendet werden, allerdings ist hier die Verwendung der „Serial Messages“ vorgesehen. Aus diesem Grund müssen bei den Modi „Powertrain 1“ und „Powertrain 2“ die „Serial Messages“ verwendet werden. Vorgegeben wird in diesen Modi auch die Verwendung von 8 Bit „Message-ID“ mit 12 Bit Daten. Da durch die „Serial Messages“ hier keine Initialisierungswerte übertragen werden, werden die Parameter kontinuierlich und parallel zu den Messwerten mit den „Enhanced Messages“ übertragen. Die Übertragung des Messwertes erfolgt parallel dazu. In der folgenden Tabelle sind die verwendeten „Message – IDs“ aufgelistet:

Message ID	Code	Beschreibung
0x01	0x000	„Diagnostic Error Codes“: kein Fehler
0x03	0x002	Channel 1 / 2 Sensor type: Drucksensor
0x05	0x061	Herstellercode: Analog Devices
0x06	0x103	PSI5 V2.3

Tabelle 4-8: „Message ID“ in den Modi „Powertrain 1 und 2“

Wie im SENT-Kapitel *2.2.1.2* beschrieben, sind bei „Enhanced Messages“ alle IDs optional und werden vom Hersteller ausgewählt. Für die Modi „Powertrain 1“ und „Powertrain 2“ wurden 4 IDs ausgewählt und mit einem Beispielwert versehen. Auch hier ist der Hersteller Code für „Analog Devices“ nur als Beispiel angegeben.

Sollen im Powertrain mehr als 10 Bits verwenden, so gilt für die Codes aus den Wertebereichen 2 und 3 (10 Bit Codes), dass die verbleibenden Bits immer auf „0“ gesetzt werden müssen.

Beim Modus „Powertrain 1“ kann die Anzahl der Bits zwischen 12 und 28 frei verändert werden, wobei immer 2 Bits für die „Serial Messages“ vorgegeben sind. Die Datenwerte mit Ausnahme der „Serial Messages“ können ebenfalls beliebig verwendet werden. Wird im Modus „Powertrain 2“ die Datenregion B verwendet, so kann hier keine Eingabe gemacht werden. Die Datenregion B wird in diesem Modus als Sicherheitszähler verwendet.

„Chassis & Safety 1 und 2“:

Der Substandard „Chassis & Safety“ schreibt die Verwendung von 20 Bits für die Nutzdaten vor (siehe Kapitel *2.2.2.6*). Als Empfehlung werden zwei Varianten angeführt. Bei der ersten Variante teilen sich die 20 Bits in 3 Bit „Frame-Control“, 1 Bit Status und 16 Bit für die Datenregion A auf. Bei der zweiten Variante „Chassis & Safety 2“ können zwei Sensoren in einer Übertragung genutzt werden, die Nutzdaten teilen sich zu jeweils 10 Bit auf die Datenregion A und B auf. Bei beiden Modi werden für die Phase 2 Initialisierungswerte verwendet bzw. bei „Chassis & Safety 2“ werden auch in der Region B Initialisierungswerte gesendet, da es sich um einen eigenständigen Sensor handelt. Beim Modus „Chassis & Safety 1“ werden, wie im Base-Standard definiert, die verbleibenden 6 Bits für Codes aus Datenregion 2 und 3 frei verwendet.

Bei den Initialisierungswerten in *Tabelle 4-9* ist der Code für „Analog Devices“ nur ein Beispiel. Anhand der Werte lässt sich erkennen, dass es sich um einen Winkelsensor mit einem Messbereich von 90° handelt.

	Header	**Initialisierung**		**Hersteller-Code**		**Produkt-Code**			
Datenfeld	F1	F2		F3		F4		F5	
Daten-Nibble	D1	D2	D3	D4	D5	D6	D7	D8	D9
Binärcode	0110	0001	0000	0110	0001	0000	0110	0101	1010
Dezimalwert	6	1	0	6	1	0	6	5	A
Beschreibung	PSI5 V2.X	Länge D1-D16		Analog Devices		Position Sensor (Linear)		90° Winkel	

Tabelle 4-9: Initialisierungswerte F1 bis F5 im „Chassis & Safety 1“ bzw. „Chassis & Safety 2“ Modus

Das Datenfeld F6 kann vom Hersteller frei verwendet werden, weshalb hier ein fiktiver Wert verwendet wurde.

	„Application specific“						
Datenfeld	F6						
Daten-Nibble	D10	D11	D12	D13	D14	D15	D16
Binärcode	0000	0101	1010	0011	1111	0000	0001
Dezimalwert	0	5	A	3	F	0	1
Beschreibung	Fiktiver Wert: 05A3F01						

Tabelle 4-10: Initialisierungswerte F6 im „Chassis & Safety 1“ bzw. „Chassis & Safety 2“ Modus

„2 Zeitslots“:

In diesem Modus werden innerhalb eines Zyklus zwei Zeitslots (11 Bit, 1 Bit „Frame Control“) simuliert. Bei den Initialisierungswerten (identisch mit „allgemeiner Modus“) wird auch im Slot 2 eine Initialisierung durchgeführt. Zur Vereinfachung werden die Sensorwerte (Messwert) in Datenregion A invertiert im zweiten Zeitslot gesendet. Mit dem „Frame Control“-Feld wird der jeweilige Zeitslot markiert, damit diese anschließend in der PC-Software unterschieden werden können.

Wie zuvor beschrieben, endet die Übertragung eines Zeitslots, indem ein Indikator den Timer zur Aussendung stoppt. Im Modus „2 Zeitslots“ wird der Indikator erst gesetzt, wenn die Daten für den zweiten Zeitslot vorberechnet wurden. Die beiden Zeitslots müssen mit einer Pause

von T_{GAP} > 8,4µs getrennt werden (siehe Kapitel *2.2.2.3*). In diesem Modus besitzt die Pausenzeit einen Wert von T_{GAP} = 16µs, dadurch können die Zeitslots problemlos am Empfänger unterschieden werden.

Im asynchronen Betrieb sind hingegen keine Zeitslots definiert, da kein Synchronisationsimpuls existiert. Durch die Verwendung des „Frame Control“-Feldes bekommt man jedoch die Möglichkeit einzelne Frames im asynchronen Betrieb zu unterscheiden, was im Modus „2 Zeitslots“ ebenfalls simuliert werden kann.

4.1.5.2 PSI5 – Empfang

Beim ersten Start oder nach einer Konfigurationsänderung werden alle Timer und Interrupts für den Empfang deaktiviert, wie es auch bei der PSI5-Aussendung und bei der SENT-Schnittstelle der Fall ist. Die Busspannung wird dabei auf „0V“ gesetzt und bleibt auf diesem Wert bis alle notwendigen Konfigurationen übernommen sind. Danach wird die Busspannung wiederhergestellt und nach 100ms (ohne „Serial Messages“) bzw. 9ms (mit „Serial Messages“) der erste Synchronisationsimpuls ausgesendet. Aufgrund der fehlenden Erkennung der Busspannung (siehe Kapitel *4.1.5.1*) wird bei der eigenen PSI5-Implementierung dadurch das richtige Startverhalten (Phase 1 mit 50 bis 150/200ms bzw. < 10ms bei „Serial Messages“) simuliert.

Der Empfang von PSI5-Nachrichten mit der eigenen PSI5-Empfangsschaltung funktioniert grundsätzlich nach demselben Prinzip wie bei der SENT-Empfangsschaltung. Im Unterschied zu SENT werden sowohl fallende als auch steigende Flanken detektiert. Beim Auftreten des Interrupts wird neben dem Zeitunterschied zwischen den einzelnen Flanken auch der Zustand des Mikrocontroller-Pins (High/Low) in einem Buffer gespeichert. Im Hauptprogramm „main.c“ wird anschließend ebenfalls der Zeitunterschied berechnet und in einem neuen Buffer abgelegt. Zusätzlich beginnt die Auswertung der PSI5-Daten erst, wenn für eine PSI5-Nachricht genügend Zeitwerte vorhanden sind. Nachdem hier fallende und steigende Flanken detektiert werden, muss die doppelte Anzahl an Werten im Buffer vorhanden sein. Hierzu zählen die zwei Startbits sowie das Paritätsbit / CRC-Bit.

Bei der Auswertung der Zeiten erfolgt die Erkennung des PSI5-Frames anhand der Pausenzeiten. Deshalb wird geprüft ob der erste Zeitwert eine Pausenzeit von T_{GAP} > 8,4µs darstellt. Sollte dies nicht der Fall (z.B. fehlerhafte PSI5-Konfiguration, fehlerhafte PSI5-Nachricht) sein, werden solange Buffer-Werte übersprungen, bis wieder eine Pause detektiert

wird. Wenn für den ersten Wert erfolgreich eine Pause erkannt wurde, werden alle Buffer-Werte der Reihe nach berechnet um das Ende des aktuellen PSI5-Frames zu finden. Bei der Berechnung der Werte wird mit der eingestellten Timer-Frequenz der tatsächliche Zeitwert in Mikrosekunden berechnet. Im Anschluss wird überprüft, ob die Bitzeit T_{Bit} im zulässigen Toleranzbereich liegt (5% → 7,6µs bis 8,4µs). Zusätzlich wird überprüft ob alle Flanken korrekt empfangen wurden, indem der Zustand des Mikrocontroller-Pins ausgewertet wird. Im nächsten Schritt folgt die Manchester-Dekodierung, bei der die Zeitwerte in digitale Werte umgewandelt werden. Die Funktionsweise der Dekodierung wurde ausführlich an der entsprechenden Stelle im Code des Mikrocontrollers dokumentiert.

Gleich wie bei SENT werden die berechneten Daten zuerst über USB / RS232 oder CAN übertragen, erst anschließend folgt die Paritäts- bzw. CRC-Prüfung. Dadurch besteht die Möglichkeit die Ursache für den Paritäts-/CRC-Fehler zu analysieren. Die Berechnung der CRC-Prüfsumme erfolgt analog zur Berechnung bei Aussendung von PSI5-Nachrichten.

Wenn beim Empfang sehr häufig Fehler auftreten (z.B. falsche Konfiguration, oder zu häufige Übertragungsfehler) kann es durch die große Anzahl an Zeitwerten zu Buffer-Überläufen kommen. Aus diesem Grund wird bei mehr als 10 Fehlern in Folge, ohne eine einzige korrekte PSI5-Botschaft, der Empfang neuer Nachrichten für 100ms ausgesetzt. Dadurch hat die Software genügend Zeit die entsprechenden Fehlerbehandlungen durchzuführen. Sobald eine fehlerfreie Nachricht empfangen wird, wird der Fehlerzähler auf null zurückgesetzt.

Für die analoge Ausgabe am „AD5620“ (12 Bit DAC) wird nur die Datenregion A verwendet. Wie bei SENT entspricht ein Wert von „0“ einer Spannung von 2,5V. Die Daten aus Region A werden entsprechend dem positiven / negativen Dezimalwert addiert bzw. subtrahiert. Der Wertebereich für die Datenregion A kann wesentlich größer werden als die 12 Bit des „AD5620“. In diesem Fall werden nur die ersten 12 MSBs berücksichtigt. Um einen symmetrischen Spannungsbereich zu erhalten, wird der Wertebereich für einen Spannungsbereich von 0,5 bis 4,5V aufbereitet. Der angepasste Wert wird dem Buffer des „AD5620“ übergeben.

Sollten „Serial Messages“ zum Einsatz kommen werden diese abschließend berechnet, sofern 18 einzelne PSI5-Botschaften empfangen wurden. Die Auswertung und Übermittlung an die PC-Software erfolgt analog zur SENT-Schnittstelle.

In der Tabelle für die möglichen Modi (*Tabelle 4-4*) wurde bei einigen Modi der Empfang in Klammern gesetzt. Prinzipiell stehen diese Modi für den PSI5-Empfang zur Verfügung, allerdings haben sie in der derzeitigen Form keinen Einfluss auf die Auswertung. Bei den Substandards erfolgt also keine Auswertung, ob die Regeln tatsächlich eingehalten werden (z.B. Handhabung der verbleibenden Bits bei mehr als 10 Bit, oder Prüfung des Sicherheitszählers im Powertrain 2). Diese in Klammern gesetzten Modi dienen derzeit nur als Auswahlhilfe in der PC-Software, siehe Kapitel 4.2. Da die Modi bereits über USB/RS232/CAN konfiguriert werden können, wird eine spätere Integration von zusätzlichen Sicherheitsabfragen ermöglicht.

Wie in der Einleitung von Kapitel 4.1.5 erwähnt, steht der Modus „AIS1200PS“ zur Verfügung. Bei diesem Sensor „AIS1200PS“ besitzt das erste Startbit eine zu kurze Zeitdauer, weshalb es in diesem gewählten Modus ignoriert wird, um die Daten einlesen zu können. Details zum Sensorverhalten können im Kapitel 5.2.3 nachgelesen werden.

4.1.5.3 PSI5-Transceiver - E521.40

Beim Baustein „E521.40“ der Firma ELMOS Semiconductor AG besteht die Möglichkeit, die Kommunikation der PSI5-Signale (Synchronisationsimpuls, Manchester-Dekodierung, Analyse der Signalzeiten, etc.) auf einen externen Baustein auszulagern. Der Baustein wird in weiterer Folge nur kurz als „ELMOS“ bezeichnet. Der Mikrocontroller steuert und empfängt Daten vom ELMOS über eine SPI-Schnittstelle. Die Schnittstelle wurde dabei wie folgt konfiguriert:

- **Frequenz**: 3,125 MHz (bzw. Bitrate mit 3,125 Mbit/s)
- **Direction**: „2 Lines“ (Fullduplex)
- **CLK-Phase**: „2 EDGE“
- **CLK-Polarity**: „LOW“
- **Data size**: 8 Bit
- **First bit**: MSB
- **Mode**: Master
- **CRC**: Deaktiviert

Die Frequenz wurde mit 3,125 MHz gewählt, da der ELMOS eine Frequenz von 0 bis 5 MHz erlaubt, die nächst größere Frequenz von Seite des Mikrocontrollers jedoch 6,25MHz beträgt. Die Kommunikation mit dem ELMOS erfolgt durch 16 Bit Befehle. Die SPI-Schnittstelle ist aber

nur mit 8 statt 16 Bit konfiguriert. Der Grund dafür ist, dass die HAL-Treiber der SPI-Schnittstelle einen 8 Bit Buffer erwarten, aber bei der 16 Bit Konfiguration den 8 Bit Wert als 16 Bit Wert übertragen. Dadurch würden sich insgesamt 32 Bit für einen 16 Bit Befehl ergeben. Anstelle eines 16 Bit Befehls wird deshalb der Befehl in zwei Variablen mit je 8 Bit aufgeteilt und der SPI-Schnittstelle übergeben. Durch die Konfiguration mit 8 Bit für die Schnittstelle werden die zwei mal 8 Bit als 16 Bit Befehl übertragen.

Zur Kommunikation mit dem ELMOS stehen folgende SPI-Befehle zur Verfügung [26]:

- **SPI_Write_Register** – Kodierung (16 Bit): 0001 0000 0000 0000
- **SPI_Read_Register** – Kodierung (16 Bit): 0010 0000 0000 0000
- **SPI_Sync_Pulse** – Kodierung (16 Bit): 0011 0000 0000 0000
- **SPI_Get_Data_16b** – Kodierung (16 Bit): 0100 0000 0000 0000
- **SPI_Get_Data_24b** – Kodierung (16 Bit): 0101 0000 0000 0000
- **SPI_Get_Data_32b** – Kodierung (16 Bit): 0111 0000 0000 0000
- **SPI_Get_Data_48b** – Kodierung (16 Bit): 1000 0000 0000 0000
- **SPI_NOP** – Kodierung (16 Bit): 1110 0000 0000 0000
- **SPI_SW_Reset** – Kodierung (16 Bit): 1111 0000 0000 0000

Beim Befehl handelt es sich um einen 4 Bit Code beginnend beim MSB des 16 Bit Wertes. In Abhängig vom jeweiligen Befehl müssen für die verbleibenden Bits entsprechende Werte ausgewählt werden, beispielsweise müssen für den Befehl „**SPI_Write_Register“** noch die Adresse und der zu speichernde Inhalt eingefügt werden.

Da es sich beim ELMOS um einen universal einsetzbaren PSI5-Transceiver mit zwei Eingangskanälen („Channels“) handelt, müssen beide durch Programmierung der ELMOS-Register für das jeweilige PSI5-Signal konfiguriert werden. Für die grundlegende PSI5-Konfiguration stehen 3 „ASIC Register“ (0x00, 0x01 und 0x02) zur Verfügung. In diesen werden allgemeine Parameter wie die Busspannung, Datenrate oder asynchroner/synchroner Betrieb eingestellt. Jedes Register besitzt einen Wertebereich von 16 Bit, die entsprechende Unterteilung der einzelnen Register kann im Datenblatt nachgelesen werden. Für diese Arbeit wird das „ASIC Register 1“ (0x00) so konfiguriert, dass der interne Spannungsregler eine Busspannung von 6,65V liefert. Im „ASIC Register 3“ (0x02) werden die Ladungspumpe und

beider Channels aktiviert. Der synchrone Betrieb ist standardmäßig aktiviert und wird für diese Arbeit nicht verändert.

Für jeden Channel stehen sieben Register zur Konfiguration der einzelnen PSI5-Nachrichten zur Verfügung (Channel 1: 0x03 bis 0x09, Channel 2: 0x0A bis 0x10). Der Empfang wird beim Transceiver in Form von Zeitslots konfiguriert. Maximal können 6 Zeitslots verwendet werden, die in den „Channel Register 1 bis 6“ festgelegt werden. Für jeden Zeitslot kann getrennt eine Durchführung der Paritäts-/CRC-Prüfung, die Länge des Zeitslots und die Anzahl der Bits festgelegt werden. Nicht verwendete Zeitslots werden durch die Angabe von 0 Bit deaktiviert. Im Register 7 wird die Größe des Daten-Buffers pro Channel festgelegt. Die genaue Konfiguration der einzelnen Register kann im Datenblatt nachgelesen werden. Die gewählte Register-Konfiguration variiert je nach gewählter PSI5-Einstellung am Mikrocontroller. Die Software des Mikrocontrollers ermöglicht derzeit maximal 2 Zeitslots und maximal 24 Datenbits pro Zeitslot, da die SPI-Befehle mit der Anzahl der Slots und Datenbits immer komplexer werden. Die verwendeten Konfigurationen und Registerwerte sind im Mikrocontroller-Code an der entsprechenden Stelle dokumentiert.

Zusätzlich gibt es 6 Fehlerregister die ausgewertet werden können. Im ersten Fehlerregister „Global ASIC errors“ (0x25) werden allgemeine Fehlermeldungen (SPI-Fehler, Überspannung, etc.) zur Verfügung gestellt. Tritt an einem Channel ein Fehler auf, so kann der betroffene Channel mit dem Fehlerregister 0x26 erkannt werden. Für beide Channels stehen dann jeweils 2 Register (Channel 1: 0x27 + 0x28, Channel 2: 0x29 + 0x2A) zur Auswahl. In diesem Channel Fehlerregister kann der jeweilige Fehler (z.B. Frame-Fehler, Buffer-Fehlkonfiguration, Paritäts/CRC-Fehler, etc.) abgerufen werden. Der jeweilige Fehlerstatus wird erst nach dem Abruf der Registeradresse zurückgesetzt.

Für einige SPI-Befehle sind neben den 4 Bit Codes noch zusätzliche Angaben notwendig. In der *Abbildung 4-1* wird die Funktionsweise des SPI-Befehls „SPI_Write_Register**“** dargestellt. Die ersten 4 Bits geben den „Write“ Befehl an, anschließend wird mit 6 Bit die Registeradresse angegeben. Nach dem Register folgen 16 Bit mit der gewünschten Registerkonfiguration. Da pro Befehl nur 16 Bit übertragen werden können, müssen die verbleibenden 10 Bits in einem zweiten SPI-Befehl übermittelt werden. Gleichzeitig liefert der Sensor auf der „MISO“ (Master in, Slave out) Leitung immer auch die Antwort auf die jeweiligen Befehle. Im zweiten SPI-Frame „n+1“ liefert der Sensor auf der „MISO“ Leitung zuerst nochmals den Befehl sowie die

Registeradresse, anschließend folgt zur Kontrolle mit 16 Bit der übernommene Registerwert. Da für die Rückantwort insgesamt 32 Bit benötigt werden, muss im SPI Frame „n+2“ entweder der nächste Befehl (Schreibbefehl, Lesebefehl, etc.) oder ein „No Operation“ Befehl (SPI_NOP) übermittelt werden. Die letzten 6 Bit enthalten eine „XCRC“ – Prüfsumme mit der die empfangen Daten am Mikrocontroller auf Übertragungsfehler geprüft werden können. Die genaue Beschreibung der „XCRC“-Prüfsumme kann im Datenblatt nachgelesen werden.

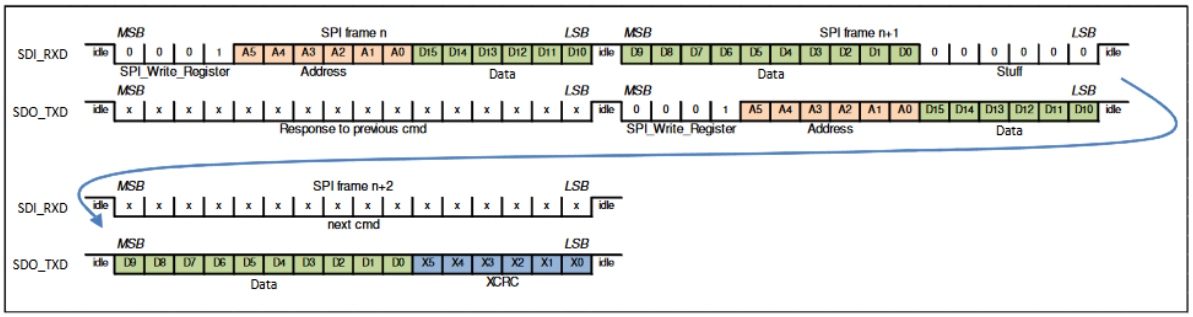

Abbildung 4-1: ELMOS SPI-Befehl „SPI_Write_Register“ [26]

Als weiteres Beispiel wird die Abfrage der PSI5-Daten in *Abbildung 4-2* dargestellt. Im ersten SPI-Frame wird der Befehl zur Abfrage von 11 Bit Sensordaten (10 Bit PSI5-Daten, 1 Paritätsbit) verwendet. Anschließend müssen der jeweilige Channel und der gewünschte Buffer angeführt werden. Der zu wählende Buffer hängt von der gewählten Buffer-Konfiguration im Register 7 und der Anzahl an Zeitslots des jeweiligen Channels ab. Im Frame „n+2“ können entweder sogenannte „Stuff-Bits“ (alle Bits sind auf „0“ gesetzt) oder optional der nächste Synchronisationsbefehl übermittelt werden. Der ELMOS liefert als Antwort wieder den Befehl, Channel, Buffer-ID sowie den zugehörigen Zeitslot und einen Indikator für potentielle PSI5-Übertragungsfehler. Anschließend werden in „n+1“ und „n+2“ die PSI5-Daten und die „XCRC-Prüfsumme“ übertragen. Zu beachten ist, dass hier die PSI5-Daten mit der Reihenfolge MSB zu LSB übertragen werden. D10 entspricht in diesem Beispiel dem Paritätsbit.

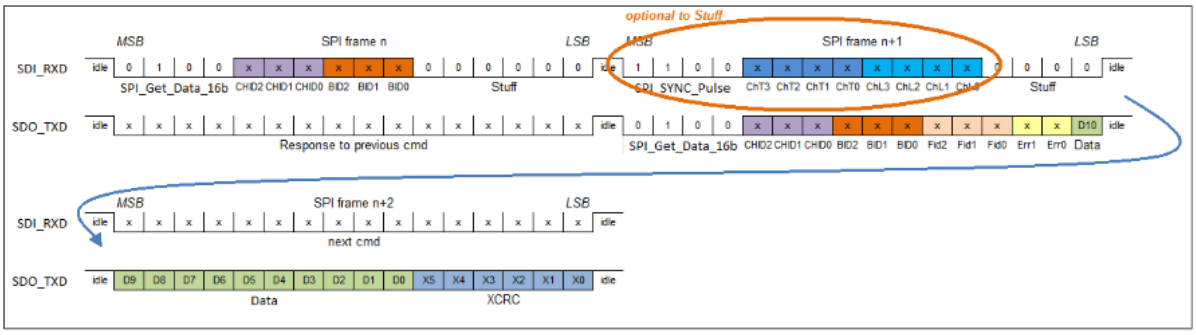

Abbildung 4-2: ELMOS SPI-Befehl „SPI_Get_Data_16b“ [26]

Es muss darauf geachtet werden, dass die PSI5-Daten abgerufen werden, bevor das nächste PSI5-Frame vom ELMOS eingelesen wird, da der Buffer sonst überschrieben wird und Daten verloren gehen. In *Abbildung 4-3* wird das gültige Zeitfenster zum Abrufen der PSI5-Daten dargestellt. Im Datenblatt wurde auch angemerkt, dass es im asynchronen Modus praktisch unmöglich ist, den Buffer synchron zu den PSI5-Daten abzurufen, weshalb für den asynchronen Modus die UART-Kommunikation empfohlen wird. Aus diesem Grund wird der asynchrone Modus beim ELMOS in dieser Masterarbeit nicht unterstützt. [26]

Alle anderen SPI-Befehle funktionieren nach demselben Prinzip, unterscheiden sich aber je nach Befehl in der Länge bzw. Anzahl der nötigen SPI-Frames. Die vollständige Funktionsweise aller Befehle kann im Datenblatt des ELMOS nachgelesen werden.

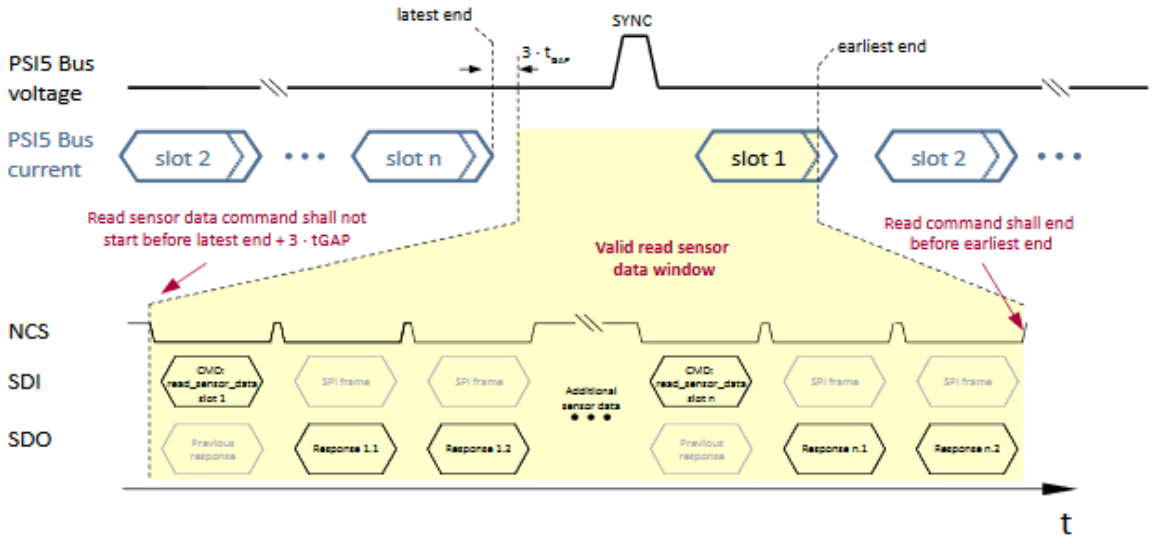

Abbildung 4-3: Zeitfenster zum Abrufen der PSI5-Daten über SPI [26]

Möchte man mehr als 11 PSI5-Bits (10 Bit Nutzdaten, 1 Paritätsbit) abrufen, verlängern sich die SPI-Frames entsprechend. Für 12 bis 19 PSI5-Bits (Nutzdaten + Parität/CRC) benötigt man den Befehl „SPI_Get_Data_24b“ und bereits insgesamt 4 SPI-Frames. Beim Befehl „SPI_Get_Data_32b“ sind bis zu 27 Datenbits möglich, er benötigt aber ebenfalls nur 4 SPI-Frames. Bei „SPI_Get_Data_48b“ würde man bereits 5 SPI-Frames benötigen. Zusätzlich wird der Befehl zur Aussendung des Synchronisationsimpulses benötigt und im Fehlerfall pro Registerabfrage (6 mögliche Register) zusätzlich 3 SPI-Frames. Eine typische SPI-Kommunikation mit „SPI_Get_Data_32b“, einem Zeitslot und PSI5-Übertragungsfehlern kann so auf bis zu etwa 230µs kommen. Diese Zeitdauer ist insofern problematisch, da der Mikrocontroller beispielsweise auf der PSI5-TX-Schnittstelle ebenfalls Daten aussendet. Die SPI-Kommunikation mit dem ELMOS darf aber nicht durch einen anderen Interrupt

unterbrochen werden, da sonst die Befehle als fehlerhaft verworfen werden. Die eigene PSI5-Aussendung, darf aber ebenfalls nicht durch einen SPI-Interrupt unterbrochen werden, sonst weichen die Bitzeiten von der zulässigen Toleranz ab. Beide Vorgänge müssen also mit jeweils höchster Interrupt-Priorität behandelt werden. Um dieses Problem zu lösen, wird zuerst das PSI5-Signal von der eigenen Schaltung gesendet, erst im Anschluss erfolgt die ELMOS-Kommunikation über SPI. Da die PSI5-Zyklen entweder 250 oder 500µs betragen, ergibt sich hier ein zeitliches Problem. Aus diesem Grund wird bis 16 Bit PSI5-Nutzdaten am ELMOS alle 500µs ein SPI-Befehl zur Auslösung des Synchronisationsimpulses übermittelt. Ab 16 Bit wird der Synchronisationsimpuls alle 750µs ausgelöst. Dadurch bleibt genügend Zeit um sowohl eine fehlerfreie PSI5-Aussendung bzw. ELMOS-Kommunikation zu realisieren. Aus den zuvor genannten Gründen sind deshalb nur maximal 24 Bit Nutzdaten (Maximum des „SPI_Get_Data_32b“ Befehls bei Berücksichtigung der 3 CRC-Bits) und zwei Zeitslots möglich, weil ansonsten die Zeiten noch weiter verschoben wären.

Die ELMOS-Kommunikation beginnt, sobald nach dem ersten Start bzw. einer Konfigurationsänderung alle anzuwendenden Operationen (Default-Werte der Buffer, Variablen erstellen, Display beschreiben, etc.) abgeschlossen sind. Dazu wird der Timer 14 aktiviert, der die zeitliche Koordination der SPI-Befehle steuert. Die Befehle werden dabei in einen eigenen Buffer für SPI-Befehle übertragen. Nach dem ersten Start bzw. nach einer Konfigurationsänderung wird am ELMOS ein Reset durchgeführt und alle Fehlerregister abgefragt. Damit wird sichergestellt, dass sich der ELMOS in einem definierten Default-Zustand befindet. Dadurch wird auch die Busspannung auf „0V“ gesetzt und externe Sensoren werden deaktiviert. Nach 10ms wird die Konfiguration der einzelnen Register vorgenommen und nach insgesamt 100ms der erste Synchronisationsimpuls ausgelöst. Timer 14 sorgt dafür, dass die Datenabfrage am ELMOS erst nach der Aussendung der eigenen PSI5-Schaltung erfolgt, um keine Interrupt-Probleme auszulösen. Der Timer 14 sorgt auch für eine periodische Auslösung des Synchronisationsimpulses nach 500 bzw. 750µs.

Die eigentliche SPI-Kommunikation wird bei jedem Interrupt von Timer 14 aktiv, indem dieser den Timer 7 aktiviert. Sobald Timer 7 aktiviert wurde, werden in dessen Interrupt die Befehle aus dem SPI-Befehls-Buffer übernommen und an die SPI-Schnittstelle übergeben. Sobald eine SPI-Kommunikation abgeschlossen wurde, wird von der SPI-Schnittstelle eine Callback-Funktion (Interrupt) aufgerufen, die das Ende der aktuellen Kommunikation kennzeichnet. In

dieser wird der Timer 7 aktiviert, um die weiteren Befehle abzuarbeiten. Dieser Vorgang wiederholt sich solange bis keine Werte mehr im SPI-Befehls-Buffer enthalten sind. Das Timing wurde dabei so gewählt, dass sich bei einem PSI5-Übertragungsfehler (Abfrage der Fehlerregister) keine Überschneidungen mit dem nächsten PSI5-Zyklus ergeben.

In der SPI-Callback-Funktion werden die vom ELMOS gesendeten Daten an einen Buffer übergeben. Zur Erkennung ob ein PSI5-Übertragungsfehler aufgetreten ist und damit eine Abfrage der Fehlerregister notwendig ist, werden die Fehlerbits aus *Abbildung 4-2* überprüft. Sollte ein Fehler angezeigt werden, wird sofort ein neuer ELMOS-Befehl zur Abfrage der Register in den SPI-Befehls-Buffer eingetragen. Wenn die Fehlerbits aus *Abbildung 4-2* keinen Fehler anzeigen, wird ein „No Operation“ Befehl eingefügt, damit auch das letzte SPI-Frame empfangen werden kann. Wie schon bei der eigenen PSI5-Empfangsschaltung wird nach mehr als 10 Fehlern in Folge der nächste Synchronisationsimpuls für 100ms ausgesetzt.

Im Hauptprogramm „main.c“ wird anschließend der Buffer mit den empfangenen ELMOS-Daten ausgewertet. Wenn die Daten keinen Fehlerindikator aufweisen, werden die PSI5-Daten extrahiert und über USB/RS232 oder CAN an den PC übertragen. Die analoge Ausgabe erfolgt identisch zur eigenen PSI5-Empfangsschaltung. Ebenso werden die „Serial Messages“ am Ende kalkuliert. Handelt es sich bei den empfangenen ELMOS-Daten um Fehlerregister, werden die Registeradresse sowie der Registerinhalt mit dem markierten Fehler an den PC übertragen.

4.1.6. USB / RS232 / CAN

Für die Kommunikation zwischen dem Mikrocontroller und einem externen Gerät (in der Regel PC) stehen auf der Platine die drei Schnittstellentypen USB, RS232 und CAN zur Verfügung. Die CAN-Schnittstelle ist doppelt ausgeführt, beide Schnittstellen besitzen die selbe Funktionsweise. In Bezug auf den Datenaustausch und die Parameter sind alle drei Schnittstellen gleichwertig. Mit jeder Schnittstelle können die SENT / PSI5-Parameter mit identischen Befehlen konfiguriert werden. Auch die Übertragung von Statusinformationen und Sensordaten erfolgt auf dieselbe Art und Weise. Sobald am Mikrocontroller auf einer der drei Schnittstellentypen Daten empfangen werden, gilt diese als die aktive Schnittstelle und alle anderen werden geschlossen. Eine aktive Verbindung wird einerseits durch die blaue LED und anderseits durch das Display gekennzeichnet. Für die Schnittstellen USB / RS232 steht eine eigene PC-Software (Kapitel *4.2*) zur Verfügung. Grundsätzlich kann jede beliebige

Software, welche die beiden Schnittstellen unterstützt, verwendet werden. Aufgrund der speziellen Kodierung der Signale wird jedoch die eigene PC-Software empfohlen. Für die CAN-Schnittstelle ist beispielweise ein USB-CAN Adapter mit entsprechender PC-Software erforderlich.

Für alle drei Schnittstellen stehen in der Mikrocontroller-Software eigene Daten-Buffer zur Verfügung, sowohl für den Empfang als auch für den Versand. Für den Versand werden die Buffer im Hauptprogramm „main.c“ mit den entsprechenden Daten (Statusinformationen, Konfigurationen, Sensordaten, etc.) gefüllt. Diese werden dann sobald als möglich im an die jeweilige Schnittstelle übergeben und von diesen eigenständig Übertragen. Werden keine Sensordaten empfangen, findet in der Regel keine Datenübertragung statt. Sollte die Verbindung zwischen PC und Mikrocontroller abbrechen, würde ein Ausfall in der PC-Software nicht bemerkt werden. Bei USB und RS232 wird deshalb zusätzlich alle 1,5s auch ein „Alive“ Status (ASCII-Wert „null“ – 0x0) übertragen. Die PC-Software kann somit einen Verbindungsabbruch feststellen.

4.1.6.1 Schnittstellenkonfiguration

Zur Umsetzung der USB-Schnittstelle stehen die in der Einleitung von Kapitel 4.1 beschriebenen „Firmware-Examples“ zur Verfügung. Mit deren Hilfe muss die USB-Schnittstelle nur noch initialisiert werden und kann dann zur Kommunikation im Interrupt-Modus verwendet werden. Die USB-Schnittstelle wird als virtueller COM-Port am PC zur Verfügung gestellt. Falls Windows den Treiber der Schnittstelle nicht automatisch installiert, kann dieser von der STMicroelectronics-Webseite heruntergeladen werden: http://www.st.com/en/development-tools/stsw-stm32102.html [29].

Bei Verbindungsproblemen kann es vorkommen, dass die Mikrocontroller-Software beim Versuch der Datenübertragung abstürzt. Um dies zu verhindern wurde ein Zähler zur Erkennung von „timeouts“ integriert. Wird der Zählerstand von 0xFFFF überschritten, so wird der erneute Versuch die USB-Daten zu senden abgebrochen und die USB-Schnittstelle geschlossen.

Bei RS232 erfolgt die Übertragung ebenfalls mit Hilfe von Interrupts. Dabei muss die Empfangsschnittstelle als DMA-Interrupt konfiguriert werden. Bei der Verwendung von HAL-Treibern und Interrupts ohne DMA können keine Daten empfangen werden. Die Ursache hierfür ist nicht bekannt. Im Gegensatz zu USB sendet die UART-Schnittstelle kontinuierlich

Daten aus, unabhängig davon ob ein Empfänger vorhanden ist. Deshalb kann die Übermittlung im Gegensatz zu USB nicht abstürzen. Die Daten werden solange an die Schnittstelle übertragen bis die Schnittstelle deaktiviert wird. Für die grundlegende Konfiguration wurden folgende Einstellungen gewählt:

- Baudrate: 921600
- Datenbits: 8 Bit
- Stoppbits: 1
- Paritätsbit: keine Parität
- Flusssteuerung (Hardware Flow Control): keine
- Modus: TX + RX

Für die CAN-Schnittstellen erfolgt die Übertragung ebenfalls über Interrupts. Sollten Fehler auftreten, so wird ein eigener „ErrorCallback“ aufgerufen, wodurch die Übertragung nicht wie bei USB abstürzen kann. Der „Prescaler“ (Frequenzteiler) wurde für eine Bitrate von 500 kBit/s konfiguriert. Die Abkürzung „TQ“ bezeichnet die Anzahl der „Zeitquanten“. Anhand dieser kann der Abtastzeitpunkt der Bitsegmente bestimmt werden. In der folgenden Auflistung sind die wichtigsten Konfigurationsparameter angeführt:

- SJW (Synchronization Jump Width): 2 TQ
- BS1 (Bit Segment 1): 2 TQ
- BS2 (Bit Segment 2): 1 TQ
- Bitrate: 500 kBit/s
- Filter: kein ID-Filter

Der Abtastpunkt ergibt einen Wert von 75%. Die Berechnung der Bitrate und des Abtastpunktes kann im Code zur CAN-Konfiguration nachgelesen werden.

4.1.6.2 Schnittstellenkommunikation

Für die Kommunikation zwischen dem Mikrocontroller und einem PC über USB/RS232/CAN werden eine Reihe von Befehlen und Parametern zur Verfügung gestellt.

In den vorhergehenden Kapiteln wurde bereits empfohlen die Konfiguration der SENT- und PSI5-Parameter nur über USB/RS232 oder CAN durchzuführen, da bei einer direkten Konfiguration im Mikrocontroller-Code keine Überprüfung der zulässigen Konfigurationen

erfolgt. In der *Tabelle 4-11* werden alle verfügbaren Befehle, welche an den Mikrocontroller übermittelt werden können, aufgelistet. ***Hinweis***: Die Auswertung erfolgt bei allen drei Schnittstellen über eine gemeinsame Funktion. Bei CAN müssen deshalb die Werte ebenfalls ASCII-Kodiert übergeben werden.

Wenn eine neue Verbindung auf einer der Schnittstellen aufgebaut werden soll, muss der Parameter „?“ an den Mikrocontroller übertragen werden. Um eine neue SENT oder PSI5-Konfiguration an den Mikrocontroller zu übergeben, muss zuerst die gewünschte Schnittstelle mit „P“ oder „S“ ausgewählt werden. Anschließend muss angegeben werden, ob die folgende Konfiguration für die Empfangs- oder Sendeschnittstelle („R“ oder „T“) verwendet wird. Es muss beachtet werden, dass bei einigen Befehlen (z.B. „$X“ für den Modus) nicht alle Parameter für RX und TX identisch sind. Mit dem Parameter „>“ kann in einer Befehlsübertragung zwischen RX/TX und TX/RX gewechselt werden. Bei PSI5 müssen bei einer Änderung der Bitanzahl immer alle Bitfelder (Gesamtanzahl, Frame, Status, Region A + B, Messaging) angegeben werden. Bei der Auswertung der übermittelten Konfigurationen wird immer überprüft, ob sich die jeweiligen Parameter innerhalb der Grenzen befinden. Beispielsweise können bei SENT nur 1-6 Daten-Nibble konfiguriert werden, andernfalls wird eine Fehlermeldung ausgegeben. Bei PSI5 hingegen wird überprüft, ob die Anzahl der Bits (Gesamtanzahl, die einzelnen Felder sowie deren Kombinationen) und die zugehörigen Datenwerte (z.B. 10 Bit Region A, der Datenwert darf nicht größer als 0x3FF sein) zulässig sind. Weiters wird auch die Einhaltung von gewissen Einschränkungen kontrolliert, z.B. dass beim ELMOS max. 24 Bit und kein asynchroner Modus verwendet werden darf.

Das Ende des aktuellen Befehlssatzes wird durch den Parameter „@“ gekennzeichnet. Dadurch übernimmt der Mikrocontroller, insofern alle Parameter korrekt sind, den ausgewählten Befehl.

Befehlsbeispiel: „PRC0*1$0&00A10A20304050>T*1$0&00A10A20304050%10002A4@“

Der erste Parameter „P“ legt fest, dass die folgende Konfiguration für die PSI5-Schnittstelle gilt. Die anderen Parameter werden wie folgt interpretiert:

- R = Empfangsschnittstelle PSI5
    - C0 = Eigene PSI5-Empfangsschaltung
    - *1 = Synchroner Betrieb „short“

    - $0 = Allgemeiner Modus
    - &00A10A20304050
        - & = Bitkonfiguration
        - 00A = Gesamtbit: 10 (A)
        - 10A = Bit - Region A: 10 (A)
        - 20 = Bit – Region B: 0
        - 30 = Bit – Frame Control: 0
        - 40 = Bit – Status: 0
        - 50 = Messaging: keine “Serial Messages”
- “>” = Verlassen der Konfiguration der Empfangsschnittstelle
- T = Sendeschnittstelle PSI5
    - 1 = Synchroner Betrieb „short“
    - $0 = Allgemeiner Modus
    - &00A10A20304050
        - & = Bitkonfiguration
        - 00A = Gesamtbit: 10 (A)
        - 10A = Bit - Region A: 10 (A)
        - 20 = Bit – Region B: 0
        - 30 = Bit – Frame Control: 0
        - 40 = Bit – Status: 0
        - 50 = Messaging: keine “Serial Messages”
    - %10002A4
        - %1 = Datenregion A
        - 0002A4 = Datenwert: 0x2A4
- @ = Befehlsende, Überprüfungen und Übernahme der Konfiguration

Befehl / Code	Schnittstelle	Beschreibung	Bemerkung
?	Allgemein	Verbindungsstart	
P	Allgemein	Auswahl PSI5	
S	Allgemein	Auswahl SENT	
R	SENT / PSI5	Auswahl RX	Empfangsschnittstelle
T	SENT / PSI5	Auswahl TX	Sendeschnittstelle

&X	SENT – RX/TX	Anzahl Nibble	X = 1-6
+	SENT – RX/TX	Pause-Impuls	aktiveren
-	SENT – RX/TX	Pause-Impuls	deaktivieren
$X	SENT – RX	Modus	X = 0 → Allgemein X = 1 → Throttle Position X = 2 → Single Secure X = 3 → TLE4998S3
$X	SENT – TX	Modus	X = 0 → Allgemein X = 1 → Throttle Position X = 2 → Single Secure X = 3 → Simulation 1 X = 4 → Simulation 2 X = 5 → Simulation 3 X = 6 → Simulation 4 X = 7 → Simulation 5 X = 8 → Simulation 6 X = 9 → Simulation 7
#X	SENT – RX / TX	Serial Messages	X = 0 → keine X = 1 → Short X = 2 → Enhanced 12 Bit Daten X = 3 → Enhanced 16 Bit Daten
%0	SENT – TX	Status & Kommunikation	X = 0-F
%1XXXXXX	SENT – TX	Daten-Nibble	X = 0-F; Die Anzahl der zu übertragenden Daten-Nibble hängt von der Anzahl der ausgewählten Nibble (Parameter &X) ab. BSP: &3 → %1000
%2XXX	SENT – TX	Short Serial Message	X = 0-F; Es müssen 12 Bit angegeben werden
%3XXXXX	SENT – TX	Enhanced Serial Messages	X=0-F; Es müssen alle 20 Bit angegeben werden; Zuerst werden die Message-ID und dann die Daten eingegeben.
*X	PSI5 – RX/TX	Busmodus	X = 0 → Asynchron X = 1 → Synchron – short X = 2 → Synchron – long
$X	PSI5 – RX	Modus	X = 0 → Allgemein X = 1 → Airbag 1 X = 2 → Airbag 2 X = 3 → Powertrain 1 X = 4 → Powertrain 2 X = 5 → Chassis & Safety 1

			X = 6 → Chassis & Safety 2 X = 7 → AIS1200PS X = 8 → 2 Zeitslots
$X	PSI5 - TX	Modus	X = 0 → Allgemein X = 1 → Airbag 1 X = 2 → Airbag 2 X = 3 → Powertrain 1 X = 4 → Powertrain 2 X = 5 → Chassis & Safety 1 X = 6 → Chassis & Safety 2 X = 7 → 2 Zeitslots X = 8 → Simulation 1 X = 9 → Simulation 2
&1XX2XX3X4X5X	PSI5 – RX/TX	Datenbits	1 = Bit Region A (X = 0-F) 2 = Bit Region B (X = 0-C) 3 = Bit Frame (X = 0-4) 4 = Bit Status (X = 0-2) 5 = Messaging • X = 0 → keines • X = 1 → 12 Bit Daten • X = 2 → 16 Bit Daten Es müssen immer alle 5 Felder angegeben werden.
%1XXXXXX	PSI5 - TX	Datenwert Region A	X = 0-F; Es müssen alle 24 Bit angegeben werden. Bei z.B. nur 10 Bit, müssen die restlichen Bit mit 0 angegeben werden;
%2XXX	PSI5 - TX	Datenwert Region B	X = 0-F; Es müssen alle 12 Bit angegeben werden. Bei z.B. nur 4 Bit, müssen die restlichen Bit mit 0 angegeben werden.
%3X	PSI5 - TX	Datenwert Frame-Control	X = 0-F
%4X	PSI5 - TX	Datenwert Status	X = 0-F
%5XXXXX	PSI5 - TX	Datenwert Serial Message	X=0-F; Es müssen alle 20 Bit angegeben werden; Zuerst werden die Message-ID und dann die Daten eingegeben.
CX	PSI5 - RX	Auswahl der PSI5-RX-Schnittstelle	X = 0 = eigene PSI5-RX-Schnittstelle X = 1 = ELMOS CH1 X = 2 = ELMOS CH2 X = 3 = EMOS CH1 + CH2

>	SENT / PSI5	RX/TX Wechsel	Ermöglicht den Wechsel zwischen RX- / TX-Parameter
@	SENT / PSI5	Befehlsende	Ende der Parameterkonfiguration

Tabelle 4-11: Befehle zur Konfiguration der SENT/PSI5-Schnittstellen

Bei einer fehlerfreien Befehlsübermittlung (Konfigurationsänderung) wird der ASCII-Code „0x06“ (Acknowledge) gesendet. Im Fehlerfall wird der jeweilige Fehlercode gesendet. Jedes Mal wenn am Mikrocontroller ein Befehl (korrekt, fehlerhaft oder unbekannt) empfangen wurde, wird eine Statusnachricht an die Gegenstelle übertragen, welche die aktuell verwendete SENT-/PSI5-Schnittstelle inkl. Konfiguration enthält. Dadurch ist der Gegenstelle nach jedem Befehl bekannt welche Konfiguration im Mikrocontroller verwendet wird.

In der *Tabelle 4-12* kann der Aufbau einer Statusnachricht für die SENT-Schnittstelle eingesehen werden. Jede Statusnachricht besteht aus 23 Zeichen. Der Beginn der Statusnachricht wird durch „?“ gekennzeichnet und durch ein „Line Feed“ (\n) beendet. Die einzelnen Parameter werden nicht durch Symbole gekennzeichnet, sondern werden anhand Ihrer Position innerhalb der 23 Zeichen festgelegt.

Beispiel für eine SENT - Statusnachricht: „?S6+036+030B0C54A015DA\n“

- RX: 6 Daten-Nibble
- RX: Pause Impuls = ja
- RX: Modus = 0 (Allgemein)
- RX: „Serial Messages“ = 3 (Enhanced Serial Messages 16 Bit Daten)
- TX: 6 Daten-Nibble
- TX: Pause Impuls = ja
- TX: Modus = 0 (Allgemein)
- TX: „Serial Messages“ = 3 (Enhanced Serial Messages 16 Bit Daten)
- TX: Status-Nibble = 0
- TX: Daten-Nibble = B0C54A
- TX: Serial-Message = 015DA

Zeichen	Wert	Bezeichnung
1	?	Beginn der Statusnachricht
2	S	Kennzeichnet die SENT-Schnittstelle
3	1-6	SENT-RX - Anzahl Nibble
4	+ / -	SENT-RX - Pause-Impuls ja / nein
5	0-3	SENT-RX-Modus (siehe *Tabelle 4-11*)
6	0-3	SENT-RX- „Serial Message“ Modus (siehe Parameter # *Tabelle 4-11*)
7	1-6	SENT-TX - Anzahl Nibble
8	+ / -	SENT-TX - Pause-Impuls ja / nein
9	0-3	SENT-TX-Modus (siehe *Tabelle 4-11*)
10	0-3	SENT-TX- „Serial Message“ Modus (siehe Parameter # *Tabelle 4-11*)
11	0-F	SENT-TX - Status-Nibble
12 – 17	0-F	SENT-TX - 6 Zeichen für Daten-Nibble
18 – 22	0-F	SENT-TX - 5 Zeichen „Serial Message“
23	\n	„Line Feed“ – Ende der Statusnachricht

Tabelle 4-12: Statusnachricht für die SENT-Konfiguration

Die Statusnachricht für PSI5 folgt demselben Prinzip wie die SENT-Schnittstelle. Im Gegensatz zur SENT-Schnittstelle werden bei der PSI5-Schnittstelle 40 Zeichen verwendet.

Beispiel für eine PSI5-Statusnachricht: „?P01C0AA422101C0AA422100000103FAF105AD3“

- RX: Channel = 0
- RX: Gesamtbits = 28 (1C)
- RX: Bitanzahl Region A = 10 (0A)
- RX: Bitanzahl Region B = 10 (A)
- RX: Bitanzahl Frame-Control = 4
- RX: Bitanzahl Status = 2
- RX: „Serial Messages“ = 2 (16 Bit Daten)
- RX: Busmodus = 1 (Synchron short)
- RX: Modus = 0 (Allgemein)
- TX: Gesamtbits = 28 (1C)
- TX: Bitanzahl Region A = 10 (0A)
- TX: Bitanzahl Region B = 10 (A)
- TX: Bitanzahl Frame-Control = 4
- TX: Bitanzahl Status = 2

- TX: „Serial Messages“ = 2 (16 Bit Daten)
- TX: Busmodus = 1 (Synchron short)
- TX: Modus = 0 (Allgemein)
- TX: Daten für Region A = 000010
- TX: Daten für Region B = 3FA
- TX: Daten für Frame-Control = F
- TX: Daten für Status = 1
- TX: Daten für „Serial Messages“ = 05AD3

Zeichen	Wert	Bezeichnung
1	?	Beginn der Statusnachricht
2	P	Kennzeichnet die PSI5-Schnittstelle
3	0-3	PSI5-RX – Channel
4-5	0-F	PSI5-RX – 2 Zeichen (8 Bit) Gesamtbits
6-7	0-F	PSI5-RX – 2 Zeichen (8 Bit) Bitanzahl Region A
8	0-F	PSI5-RX – Bitanzahl Region B
9	0-F	PSI5-RX – Bitanzahl Frame Control
10	0-F	PSI5-RX – Bitanzahl Status
11	0-2	PSI5-RX – „Serial Messages“ (0=keine, 1 = 12 Bit Daten, 2 = 16 Bit Daten)
12	0-2	PSI5-RX – Busmodus (siehe *Tabelle 4-11*)
13	0-8	PSI5-RX – Modus (siehe *Tabelle 4-11*)
14-15	0-F	PSI5-TX – 2 Zeichen (8 Bit) Gesamtbits
16-17	0-F	PSI5-TX – 2 Zeichen (8 Bit) Bitanzahl Region A
18	0-F	PSI5-TX – Bitanzahl Region B
19	0-F	PSI5-TX – Bitanzahl Frame Control
20	0-F	PSI5-TX – Bitanzahl Status
21	0-2	PSI5-TX – „Serial Messages“ (0=keine, 1 = 12 Bit Daten, 2 = 16 Bit Daten)
22	0-2	PSI5-TX – Busmodus (siehe *Tabelle 4-11*)
23	0-9	PSI5-TX – Modus (siehe *Tabelle 4-11*)
24-29	0-F	PSI5-TX – 6 Zeichen (24 Bit) Daten für Region A
30-32	0-F	PSI5-TX – 3 Zeichen (12 Bit) Daten für Region B
33	0-F	PSI5-TX – Daten für Frame Control
34	0-3	PSI5-TX – Daten für Status
35-39	0-F	PSI5-TX – Daten für „Serial Message“
40	\n	„Line Feed“ – Ende der Statusnachricht

Tabelle 4-13: Statusnachricht für die SENT-Konfiguration

Wenn eine fehlerhafte Konfiguration übermittelt, ein Übertragungsfehler bei PSI5 bzw. SENT, oder ein sonstiger Systemfehler auftritt (z.B. Buffer-Überlauf), wird ein zweistelliger Fehlercode übermittelt. Jeder Fehlercode beginnt mit einem Rufzeichen „!“ gefolgt von der jeweiligen Fehlernummer. Ausgenommen von dieser Regel sind ELMOS-Register-Fehler, da hier durch die Registeradresse und Registerwert insgesamt 9 Zeichen übermittelt werden. Alle Fehlermeldungen werden mit einem „\n“ („Line feed“) abgeschlossen. In der *Tabelle 4-14* werden alle verfügbaren Fehlercodes aufgelistet. Als Beschreibung dient die in der PC-Software (Kapitel 4.2) angezeigte Fehlermeldung. Einige Fehlermeldungen erscheinen nur an speziellen Stellen in der PC-Software, weshalb in der Spalte „Anmerkung“ zusätzliche Erklärungen eingefügt wurden (z.B. !70 erscheint nur bei einer PSI5-Konfiguration, weshalb in der PC-Software kein expliziter Hinweis auf PSI5 erfolgt).

Code	Fehlermeldung PC-Software	Anmerkung
!00	SENT Clock-Tick-Fehler	3µs Zeitbasis nicht in der 20% Toleranz
!01	SENT Nibble-Period-Fehler	Daten-Nibble nicht im Bereich von 12-27
!02	Reserve	
!03	SENT CRC-Fehler	
!04	SENT Short: Bit 3 in Frame 1 nicht 1	
!05	SENT Short: Bit 3 in Frame >1 nicht 0	
!06	SENT Short: CRC-Fehler	
!07	SENT Enhanced: Bit 3 in Frame 1-6 (bzw. 8) nicht 1	
!08	SENT Enhanced: Bit 3 in Frame 7, 13 u. 18 (bzw. 8) nicht 0	
!09	SENT Enhanced: CRC-Fehler	
!10	SENT Throttle: Bit 0 falsche Fehlercodierung	Status Bit 0 zeigt einen Fehler hat, der Sensor 1 liefert aber nicht 0xFFF
!11	SENT Throttle: Bit 1 falsche Fehlercodierung	Status Bit 1 zeigt einen Fehler hat, der Sensor 2 liefert aber nicht 0x000
!12	SENT Secure: DN 6 nicht invertiert	
!13	SENT Secure: Counter fehlerhaft	Sicherheitszähler fehlerhaft
!14 - !19	Reserve	
!20	ELMOS Fehler - Register: ADRESSE + REGISTWERT	ADRESSE = 2 Zeichen Registerstelle REGISTERWERT = 4 Zeichen Registerinhalt

!21	Pausenzeit beim Framestart zu kurz	PSI5 – Frame hat nicht mit Pause begonnen
!22	Keine Pause im aktuellen Frame	PSI5
!23	Fehlerhafte Bitzeit	PSI5 – Bitzeit von 8µs ± 5% überschritten
!24	Falscher PIN-Zustand	PSI5 - Mikrocontroller-Pin besitzt einen nicht erwarteten Zustand (z.B. Mikrocontroller hat eine Flanke nicht detektiert)
!25	PSI5-Start-Bits sind nicht Null	
!26	Anzahl der empfangenen Datenbits ist fehlerhaft!	PSI5 – die Anzahl der empfangen Bits stimmt nicht mit der Konfiguration überein
!27	Paritätsbit / CRC fehlerhaft	
!28-29	Reserve	
!30	FIFO USB Überlauf (TX)	
!31	FIFO USB Überlauf (RX)	
!32	FIFO USART Überlauf (TX)	
!33	FIFO USART Überlauf (RX)	
!34	FIFO SENT-Capture Überlauf	
!35	FIFO SENT-Data (Tx) - keine Daten	Buffer enthält keine Signale zum Versenden
!36	FIFO SENT-Data Überlauf (RX)	
!37	FIFO ELMOS Überlauf (TX)	
!38	FIFO ELMOS Überlauf (RX)	
!39	FIFO ELMOS SPI Fehler	
!40	FIFO PSI5-TX - keine Daten	Buffer enthält keine Signale zum Versenden
!41	FIFO PSI5-TX-SYNC-Capture Überlauf	
!42	ELMOS XCRC fehlerhaft	
!43	FIFO PIS5-RX-Capture Überlauf	
!44	FIFO PSI5-RX-Daten Überlauf	
!45	FIFO CAN Überlauf (RX)	
!46	FIFO CAN Überlauf (TX)	
!47 - !49	Reserve	
!50	Unbekannter Protokolltyp (S=SENT, P=PSI5)!	USB/RS232/CAN
!51	Anzahl der Nibble fehlerhaft (Werte: 1-6)!	SENT-Befehl
!52	Unbekannter SENT-Modus!	SENT-Befehl
!53	Unbekannte Typ der Serialmessage (Werte: 0-3)!	SENT-Befehl

!54	Unbekannter SENT-Parameter!	SENT-Befehl
!55	Unbekannte Senderichtung (R=Rx, T=Tx)!	SENT-Befehl
!56	Fehlerhafter TX-Datentyp (Werte: 0-3)!	SENT-Befehl
!57	Fehlerhafter TX-Status (Werte: 0-F)!	SENT-Befehl
!58	Fehlerhafter TX-Datennibble (Werte: 0-F)!	SENT-Befehl
!59	Fehlerhafte TX-Serialmessage (Werte: 0-F)!	SENT-Befehl
!60	Unbekannte Senderichtung (R=Rx, T=Tx)!	PSI5-Befehl
!61	Fehlerhafter Channel (Werte: 0-2)!	PSI5-Befehl
!62	Unbekannter PSI5-Parameter!	PSI5-Befehl
!63	Fehlerhafter PSI5-RX/TX-Datentyp (Werte: 0-5)!	PSI5-Befehl
!64	Falscher PSI5-RX/TX Bit-Parameter!	PSI5-Befehl
!65	Falscher PSI5-TX-Datenwert (Werte: 1-5)!	PSI5-Befehl
!66	Falscher PSI5-TX-Datenwert (Werte: 0-F)!	PSI5-Befehl
!67	Unbekannter Busmodus (Asynchron = 0, Synchron = 1)!	PSI5-Befehl
!68	TX-Datenwert Region A größer als Anzahl an Bits!	PSI5-Befehl
!69	TX-Datenwert Region B größer als Anzahl an Bits!	PSI5-Befehl
!70	TX-Datenwert Frame größer als Anzahl an Bits!	PSI5-Befehl
!71	TX-Datenwert Status größer als Anzahl an Bits!	PSI5-Befehl
!72	Bei ELMOS Auswahl ist nur der synchrone Modus möglich!	PSI5-Befehl
!73	PSI5 RX/TX Gesamtbitanzahl ist fehlerhaft (10-28 Bit)!	PSI5-Befehl
!74	PSI5 RX/TX Bitanzahl Region A fehlerhaft (10-24 Bit)!	PSI5-Befehl
!75	Die Anzahl der ausgewählten Bits stimmt nicht mit der Anzahl der Gesamtbits überein!	PSI5-Befehl
!76	Bei ELMOS Auswahl sind max. 20 Datenbits erlaubt!	PSI5-Befehl
!77	Unbekannter PSI5-Modus!	PSI5-Befehl
!78	Bei Chassis & Safety müssen 20 Datenbits verwendet werden!	PSI5-Befehl
!79	Bei Powertrain müssen Serial-Messages (12 Bit Daten) verwendet werden!	PSI5-Befehl

!80	Beim Modus '2 Zeitslots' müssen 11 Bit verwendet werden!	PSI5-Befehl
!81	Beim Modus '2 Zeitslots' muss 1 Frame-Bit verwendet werden!	PSI5-Befehl
!82	Beim TX-Modus '2 Zeitslots' müssen die Frame-Daten den Wert 0 haben!	PSI5-Befehl

Tabelle 4-14: Fehlercodes und Fehlermeldungen

In der *Tabelle 4-15* werden die unterschiedlichen Nachrichtentypen für die Übermittlung von empfangene SENT- / PSI5-Daten aufgelistet. Jede Nachricht beginnt mit dem Zeichen „&“ gefolgt von einer zweistelligen Nummer. Alle Nachrichten werden zusätzlich mit einem „\n“ („Line Feed“) beendet. Bei SENT-Nachrichten hängt die Anzahl der gesendeten Daten-Nibble von der konfigurierten Nibble-Anzahl ab. Bei PSI5 hingegen werden immer alle 28 Bit übertragen. Die einzelnen Bits müssen, beginnend beim LSB, anhand der PSI5-Bit-Konfiguration extrahiert werden. Nicht verwendete Bits werden als „0“ gesendet. Der Aufbau der gesendeten PSI5-Nutzdaten entspricht dabei der Darstellung in *Abbildung 2-20*, allerdings wird hier vom MSB zum LSB gesendet.

Code	Nachrichtentyp	Beschreibung
&00 X YYYYYY Z	SENT-Nachricht	X = Statuts, Y = Daten-Nibble (Y-Anzahl von Anzahl an Daten-Nibble abhängig), Z = CRC
&01 X YY Z	SENT Short Serial	X = Message-ID, YY = Daten, Z = CRC
&02 XX YY ZZZ	SENT Enhanced Serial	XX = CRC, YY= Message ID / Daten, ZZZ = 12 Bit Daten
&03 X YYYYYYY Z	PSI5-Nachricht	X = Channel, YYYYYYY = 28 Bit Nutzdaten, Z = Parität/CRC
&04 XX YY ZZZ	PSI5 Enhanced Serial	XX = CRC, YY= Message ID / Daten, ZZZ = 12 Bit Daten

Tabelle 4-15: Übermittlung empfangener SENT- / PSI5-Daten

4.1.7. DAC – Digital-Analog-Converter

Auf der eigenen Platine stehen zwei DAC zur Verfügung, um die empfangenen SENT-/PSI5-Daten als analogen Spannungswert auszugeben (siehe Kapitel 3.7). Die Ausgabe wird für beide DACs über eine gemeinsame SPI-Schnittstelle gesteuert. Der Mikrocontroller steuert über die „Chip Select“ Leitung welcher DAC die Daten in Empfang nehmen soll. Die SPI-Schnittstelle ist dabei wie folgt konfiguriert:

- **Frequenz**: 12,5 MHz (bzw. Bitrate mit 12,5 Mbit/s)
- **Direction**: „1 Line“ (Nur Übermittlung, kein Empfang)
- **CLK-Phase**: „2 EDGE“ (CPHA)

- **CLK-Polarity**: „LOW“ (CPOL)
- **Data size**: 8 Bit
- **First bit**: MSB
- **Mode**: Master
- **CRC**: Deaktiviert

Wie im Hardware-Kapitel 3.7 beschrieben, besitzt der DAC eine Auflösung von 12 Bit zur analogen Spannungsausgabe. Für die Datenübertragung benötigt der DAC jedoch einen 16 Bit Wert. Wie schon beim ELMOS (Kapitel *4.1.5.3*) wurden auch hier nur 8 Bit konfiguriert, wobei der SPI-Schnittstelle dann zwei 8 Bit Werte übergeben werden, die dann in einer 16 Bit Nachricht übertragen werden. Die 12 Bit Daten müssen dabei gemäß der *Abbildung 4-4* in das 16 Bit Feld transferiert werden. Die ersten beiden Bits „PD1 und PD0“ sind Steuerungsbit, die hier nicht weiter relevant sind und mit dem Wert „0“ angegeben werden. Anschließend folgen die 12 Bit Sensordaten, die letzten beiden Bits sind ohne Funktion und werden ebenfalls mit „0“ übertragen. [30]

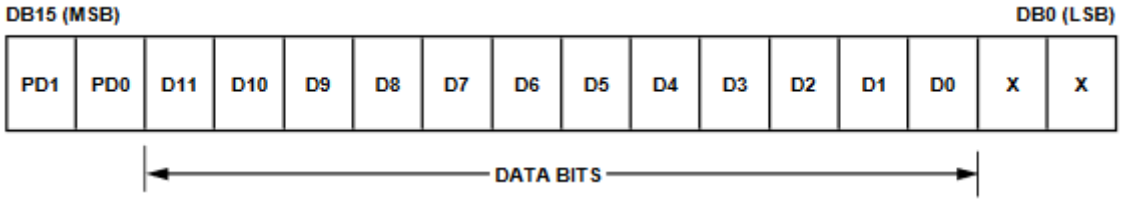

Abbildung 4-4: Bitreihenfolge für den „AD5620“ [30]

Wie in Kapitel *4.1.4.2* und *4.1.5.2* beschrieben, werden die bereits aufbereiteten Signalwerte an einen Buffer übergeben, werden jedoch zuvor entsprechend der *Abbildung 4-4* um zwei Bits nach links verschoben. Die Übermittlung an den jeweiligen DAC erfolgt im Hauptprogramm „main.c“ mittels Interrupts. Der DAC halt die Spannung am Ausgang solange aufrecht, bis ein neuer Wert übermittelt wird. Um eine unnötige Datenübertragung zu vermeiden, wird der empfangene SENT-/PSI5-Wert nur in den Buffer übernommen, wenn er sich vom letzten Datenwert unterscheidet. Dadurch findet eine Übertragung an den DAC nur bei einer Messwertänderung statt.

4.2. PC-Software

Für die Steuerung und Kommunikation über USB / RS232 mit dem Mikrocontroller wurde eine eigene PC-Software entwickelt. Grundsätzlich kann jede beliebige Software zur USB / RS232

Kommunikation verwendet werden. Aufgrund der vielfältigen Befehle und Datenkodierungen (siehe Kapitel *4.1.6.2*) kann der Datenaustausch jedoch unübersichtlich werden. Die selbstentwickelte Software abstrahiert die einzelnen Befehle und Dateninformationen soweit, dass keine Kenntnisse über die verwendeten Strukturen benötigt werden. Alle Parameter können über eine grafische Oberfläche ausgewählt werden. Die empfangenen SENT-/PSI5-Daten werden entsprechend aufbereitet und ebenfalls in der grafischen Oberfläche dargestellt.

Die Software wurde in der Programmiersprache C# entwickelt. Als Entwicklungswerkzeug wurde „Microsoft Visual Studio 2013“ verwendet. Die eigene PC-Software steht sowohl als ausführbare Datei (EXE) aber auch als Projektdatei zur Verfügung. Aufgrund der Abwärtskompatibilität kann die Projektdatei auch mit aktuelleren Versionen (2015 bzw. 2017) geöffnet werden. Für die Ausführung der EXE-Datei muss am jeweiligen PC das .NET-Framework in der Version 4 oder höher installiert sein. Der Vorteil dieser Entwicklungsumgebung ist, dass diese grafischen Elemente per „Drag & Drop“ in einer sogenannten „Design-Ansicht“ platziert werden können. Die zugehörigen Code-Elemente (Klick auf einen Button, Aktion nach einer Texteingabe) können automatisiert durch die Entwicklungsumgebung generiert werden.

Die Software wurde in drei Bereiche unterteilt: Einstellungen für die USB / RS232 Kommunikation, SENT und PSI5. Diese drei Bereiche werden in den folgenden Unterkapiteln näher vorgestellt.

4.2.1. Allgemeines & Einstellungen

Die USB-Schnittstelle wird durch den Mikrocontroller als virtueller COM-Port am PC zur Verfügung gestellt. Dadurch kann sowohl für USB als auch für RS232 ein serieller Port genutzt werden. Der serielle Port wird als Steuerelement im .NET-Framework zur Verfügung gestellt. Die Software wurde zur besseren Übersicht durch in drei getrennte Tabs unterteilt. In der *Abbildung 4-5* ist der „Tab“ mit den Einstellungen für den seriellen Port dargestellt.

Die von Windows erkannten Anschlüsse werden in einem „DropDown-Element“ zur Auswahl angeboten. Wenn der serielle Port erst nach dem Start der Software verbunden wird, kann die Auswahlliste über den Button „Port aktualisieren“ neu geladen werden. Die Parameter für die serielle Datenübertragung stehen nicht zur Auswahl bereit, da eine Änderung dieser

Parameter auch am Mikrocontroller durchgeführt werden müsste. Diese Parameter werden deshalb nur informativ angezeigt.

Beim Klick auf den Button „Port öffnen“ wird versucht die Kommunikation mit dem Mikrocontroller herzustellen, indem automatisch der Parameter „?“ (siehe *Tabelle 4-11*) gesendet wird. Wenn die Verbindung erfolgreich hergestellt wurde, sendet der Mikrocontroller eine Statusnachricht mit der aktuellen Schnittstelle (SENT/PSI5) und der derzeit verwendeten Konfiguration. Die Software wechselt in diesem Fall automatisch in den jeweiligen Tab (SENT oder PSI5). Die Verbindung kann jederzeit über den Button „Port schließen“ beendet werden.

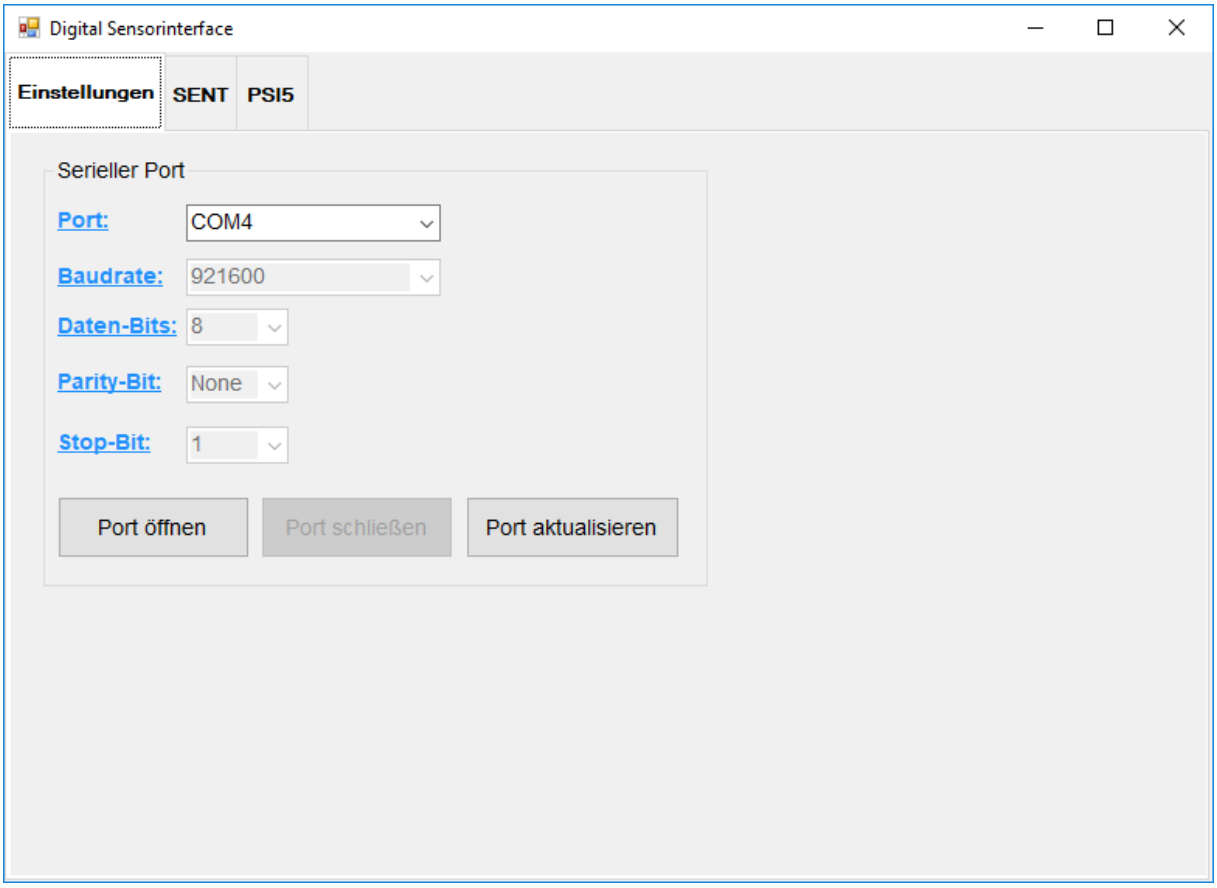

Abbildung 4-5: Einstellungen für die PC-Software

Wie im Kapitel *4.1.6* erwähnt, sendet der Mikrocontroller periodisch alle 1,5 Sekunden ein „alive“ Symbol. Die Software prüft deshalb mit Hilfe eines „Software-Timers“ alle 3 Sekunden ob dieses Symbol empfangen wurde. Sollte das nicht der Fall sein, bedeutet dies eine Unterbrechung der Verbindung zwischen PC und Mikrocontroller. Die Software versucht in diesem Fall automatisch eine neue Verbindung aufzubauen. Ist dies nicht erfolgreich, wird kein weiterer Versuch unternommen und automatisch in den Tab „Einstellungen“ gewechselt.

Diese periodische Überprüfung erfolgt jedoch nur wenn sich die Software aktiv im Vordergrund befindet. Wird ein andere Software geöffnet, so erfolgt keine Prüfung, da die Daten am seriellen Port nur mehr eingeschränkt durch das .NET-Framework im Hintergrund abgerufen werden und dadurch ungerechtfertigte Fehlermeldungen zum Verbindungsstatus erscheinen können. Sobald die Software wieder im Vordergrund läuft, wird die regelmäßige Prüfung des Verbindungsstatus wieder durchgeführt.

4.2.2. SENT

Im Tab „SENT“ können alle Parameter für die SENT-Kommunikation konfiguriert werden. Die vom Mikrocontroller empfangenen SENT-Daten werden auf der linken Seite in einer Tabelle dargestellt (siehe *Abbildung 4-6*). Jedes Nibble wird in einer eigenen Spalte dargestellt. Beim Status-Nibble erfolgt zusätzlich eine Darstellung der einzelnen Bits. Damit kann bei der Verwendung von „Serial Messages“ mit den Bits 2 und 3 leicht eine händische Berechnung durchgeführt werden. Die Spalte „Typ“ kennzeichnet zudem um welche Art von Daten es sich handelt. Bei „Serial Messages“ werden die empfangenen Daten ebenfalls in 4 Bit (Nibble) unterteilt und beginnend bei DN1 eingetragen.

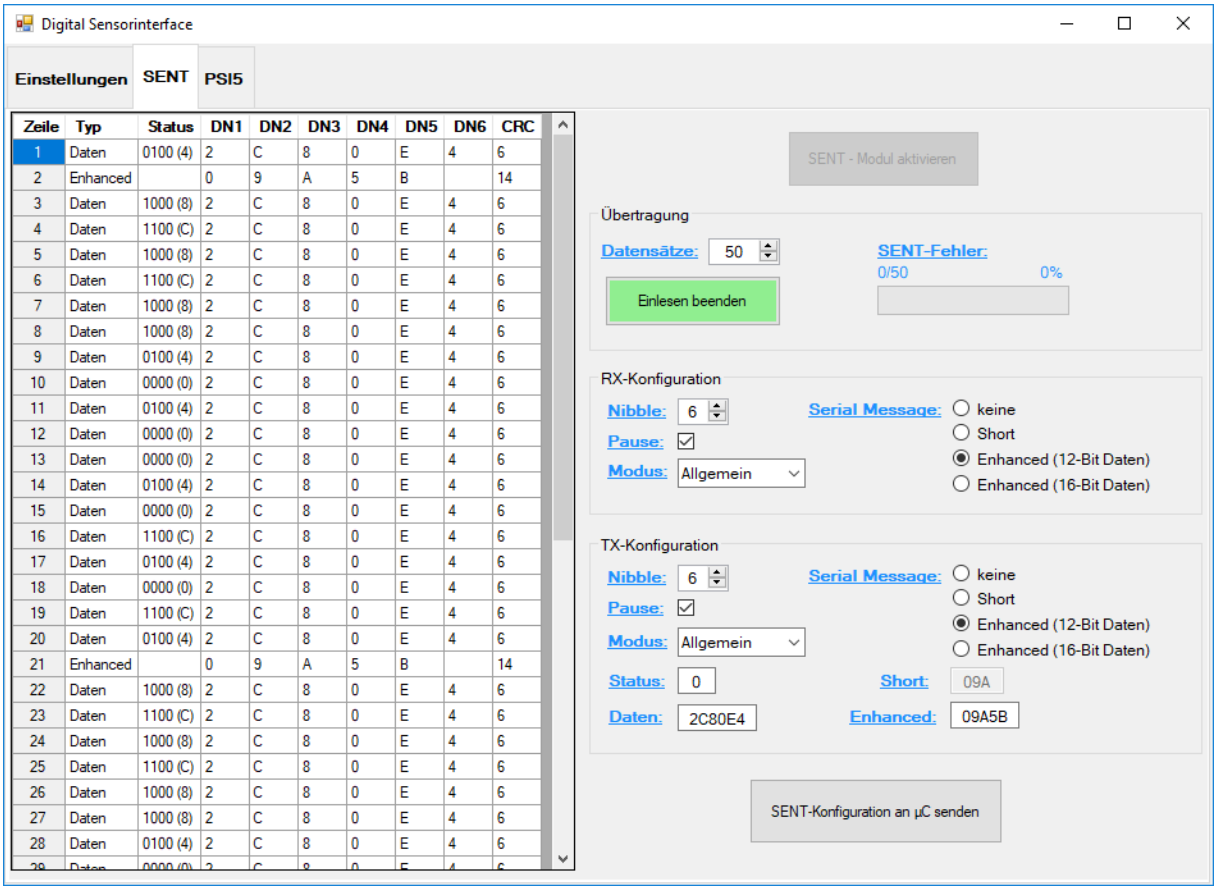

Abbildung 4-6: SENT-Kommunikation in der PC-Software

Die Konfigurationsmöglichkeiten auf der rechten Seite wurden in 3 Bereiche unterteilt. In der Gruppe „Übertragung“ kann die Anzahl der Datensätze (1-512) in der Tabelle eingestellt werden. Diese Einschränkung ist notwendig, da ein unendliches Hinzufügen der Datensätze eine Bedienung der Software unmöglich macht. Bei Überschreitung der eingestellten Anzahl wird das Hinzufügen unterbrochen und nach einer Sekunde der Tabelleninhalt über einen „Software-Timer“ gelöscht. Die Daten werden anschließend wieder bis zur nächsten Überschreitung eingelesen. Über den grünen Button „Einlesen beenden“ kann die Übernahme der Daten in die Tabelle unterbrochen werden. Die aktuell angezeigten Daten bleiben in diesem Fall ohne Veränderung in der Tabelle stehen. Der Button wird in diesem Fall in Rot dargestellt und bietet die Möglichkeit die Unterbrechung wieder aufzuheben (siehe *Abbildung 4-7*). Im Falle von Übertragungsfehlern werden diese in der Tabelle links angezeigt und zusätzlich zeigt rechts unter „Übertragung“ ein Balken die Anzahl der Fehler im Verhältnis zu positiv eingelesenen Werten in Prozent an.

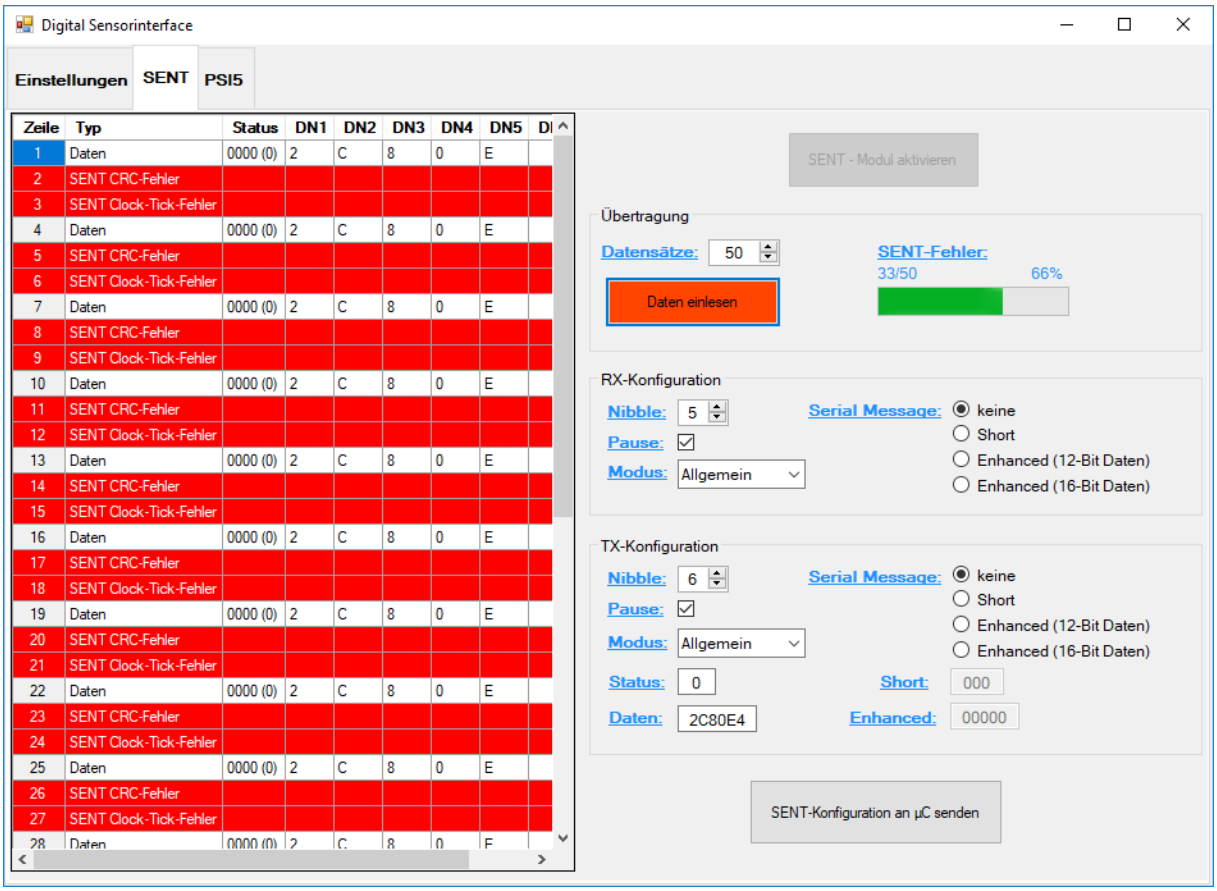

Abbildung 4-7: Darstellung von SENT-Übertragungsfehlern

In der zweiten und dritten Gruppe „RX-Konfiguration“ bzw. „TX-Konfiguration“ können die Parameter für den Empfang und Versand von SENT-Nachrichten am Mikrocontroller

eingestellt werden. Die Eingabemöglichkeit für die Anzahl an Nibble ist dabei auf 1-6 beschränkt. Im Feld „Daten“ in der „TX-Konfiguration“ können die Daten nur in Abhängigkeit der gewählten Nibble-Anzahl eingegeben werden (z.B. 2 Nibble → maximal zwei Werte). Die Eingabe der Datenwerte erfolgt im HEX-Format (0-F), andere Werte sind nicht zugelassen. Die Datenfelder für „Short“ und „Enhanced“ Nachrichten werden erst bei entsprechender Auswahl der gewünschten „Serial Message“ Form aktiviert.

Die Eingabe der Daten von „Serial Messages“ erfolgt ebenfalls als 4 Bit Werte (0-F) nach dem folgenden Schema:

- Short: 1. Stelle = Message-ID, 2. + 3. Stelle = Daten
- Enhanced 12 Bit: 1. + 2. Stelle = Message-ID, 3. bis 5. Stelle = Daten
- Enhanced 16 Bit: 1. Stelle = Message-ID, 2. Bis 6. Stelle = Daten

Beispiel aus *Abbildung 4-6*: Message-ID = 09, Daten = A58.

Die Anzeige der „Serial Messages“ erfolgt in der Tabelle Schema beginnend bei DN1 nach demselben.

Durch die grafische Konfigurationsmöglichkeit sind keine Kenntnisse über die notwendigen Befehle für die Mikrocontroller-Kommunikation erforderlich (*Tabelle 4-11*). Die Befehle werden automatisch im Hintergrund aufgrund der Eingaben erstellt. Aufgrund des Aufbaus können auch keine Parametergrenzen (z.B. mehr als 6 Nibble) verletzt werden. Auf eine korrekte Einstellung der Empfangs- und Sendeschnittstelle ist dennoch zu achten, eine Funktion zur automatischen Parametereinstellung existiert nicht. Beispiel: Für die Sendeschnittstelle werden nur 5 Nibble und kein Pause-Impuls konfiguriert. Werden die Signale wieder selbst eingelesen, muss die Konfiguration für die Empfangsschnittstelle entsprechend angepasst werden.

Mit dem Button „SENT-Konfiguration an µC senden“ kann die gewählte Konfiguration an den Mikrocontroller übermittelt werden. Die erfolgreiche Übernahme der Konfiguration wird durch den Mikrocontroller und ein Popup-Fenster in der PC-Software bestätigt.

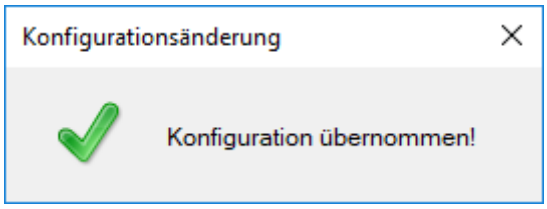

Abbildung 4-8: Popup mit positiver Konfigurationsübernahme

Wenn die SENT-Schnittstelle aktiv ist, sind die Tabelle und Konfigurationsfelder im Tab „PSI5“ deaktiviert. Wenn auf die PSI5-Schnittstelle gewechselt werden soll, muss der grüne Button „PSI5-Modul aktivieren“ betätigt werden. Sobald das PSI5-Modul am Mikrokontroller aktiviert ist, werden die Steuerelemente durch die Statusnachricht im Tab „SENT“ deaktiviert und der Button „SENT-Modul aktivieren“ wird grün.

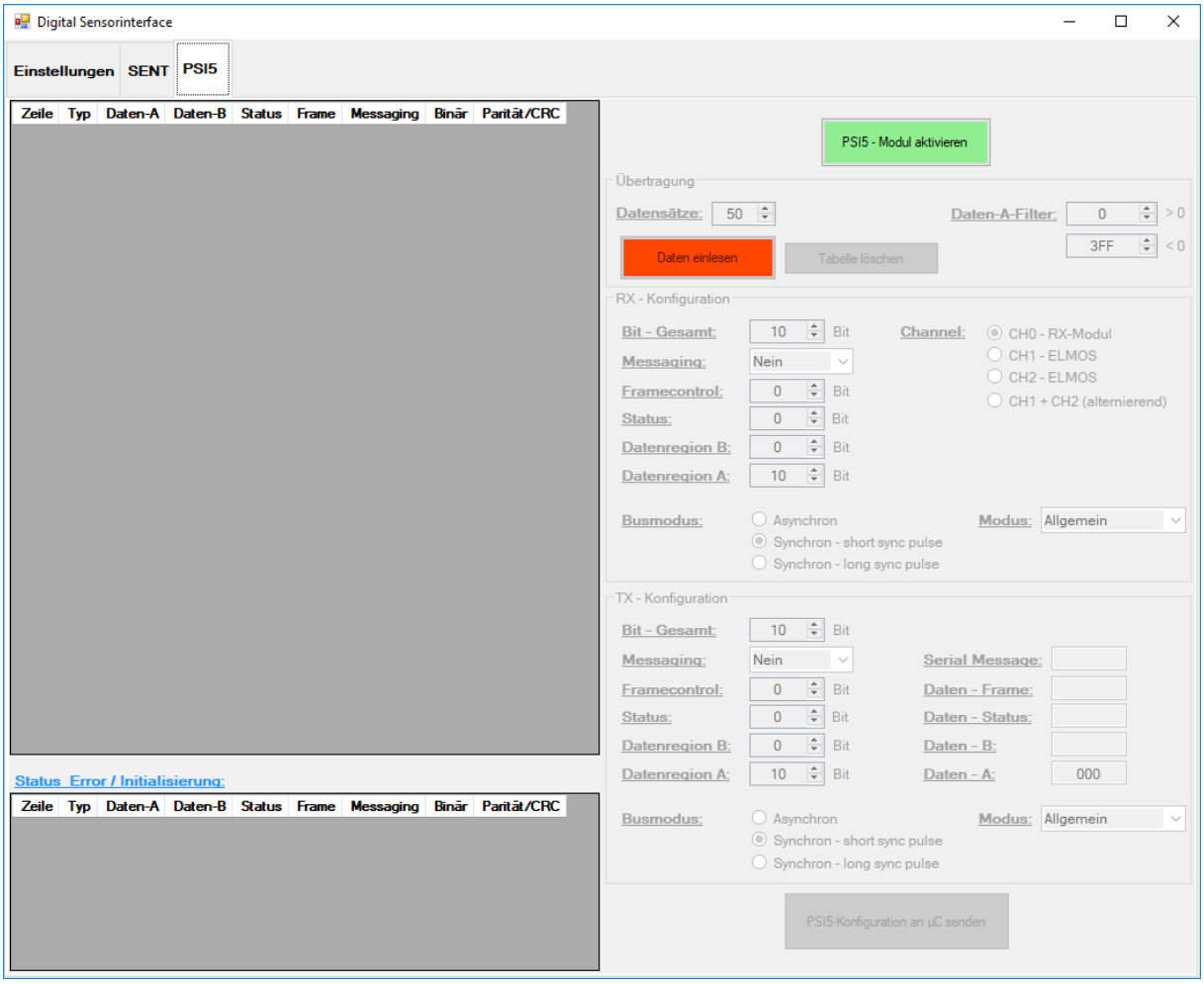

Abbildung 4-9: Bei inaktiver PSI5-Schnittstelle können keine PSI5-Parameter konfiguriert werden

4.2.3. PSI5

Der PSI5-Tab besitzt einen ähnlichen Aufbau wie SENT. In der *Abbildung 4-10* wird ein Beispiel für PSI5-Nachrichten dargestellt. Links befinden sich zwei Tabellen mit PSI5-Daten und auf der rechten Seite können die Parameter für die PSI5-Übertragung ausgewählt werden.

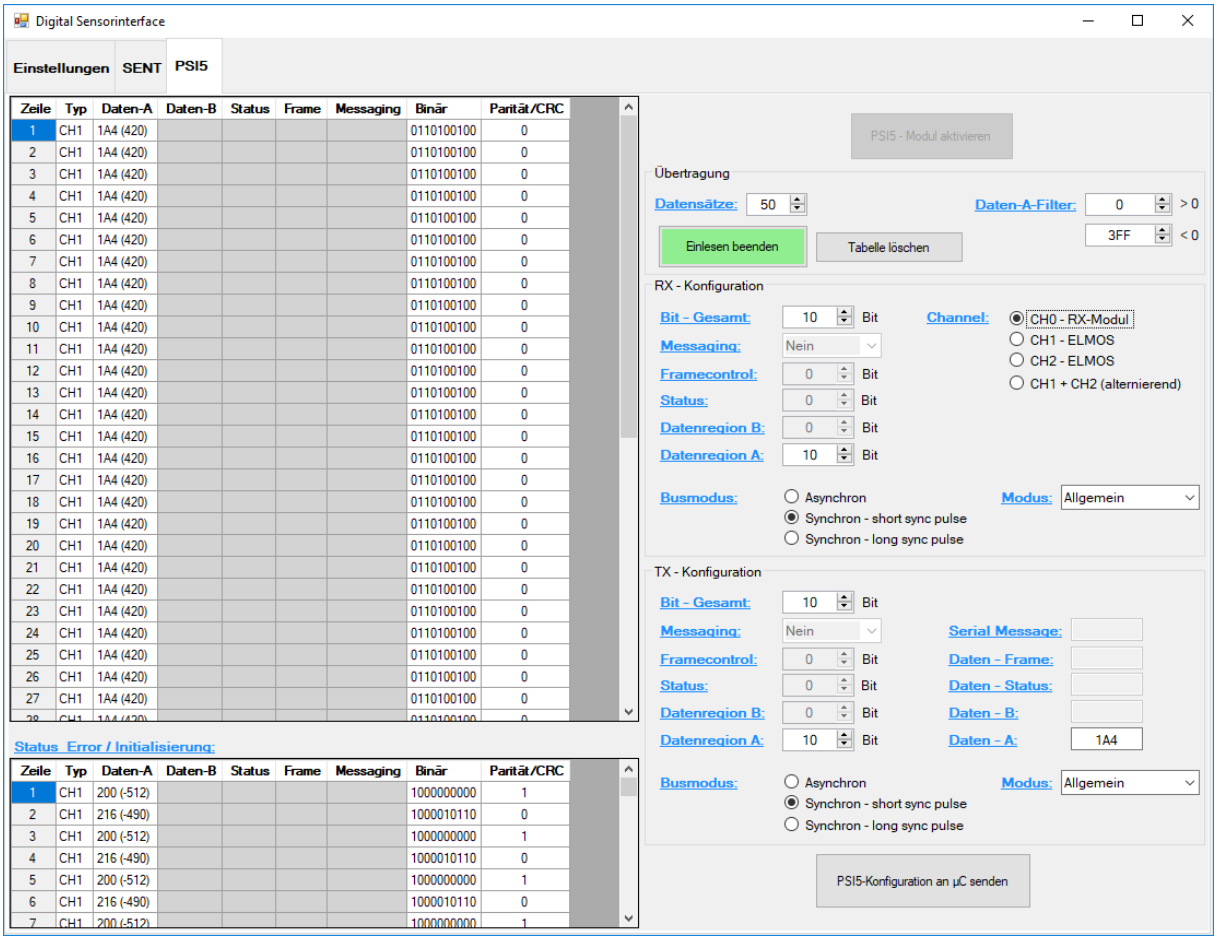

Abbildung 4-10: PSI5-Kommunikation in der PC-Software

Auch hier kann analog zu SENT die Anzahl der Tabelleneinträge festgelegt werden und das Einlesen der Daten in die Tabelle gestoppt werden. Neu ist in dieser Gruppe, dass ein Filter für die Datenregion A zur Verfügung steht. Dieser Filter ist vor allem beim Airbag-Sensor sehr hilfreich. Diese liefern ohne Beschleunigung immer den Wert „0“ (insofern der Sensor richtig ausgerichtet ist). Soll ein Aufprall mit einem Airbag-Sensor simuliert werden, beispielweise indem eine Kraft auf den Sensor einwirkt, wäre der Ausschlag der Beschleunigung nur sehr schwer festzustellen, weil im Ruhezustand sofort wieder der Wert „0“ gesendet wird. Durch die vielen Null-Messwerte wird die Tabelle sehr schnell automatisch gelöscht und die Simulationswerte gehen verloren. Wird jedoch der obere Filterwert von 0 auf z.B. 10 gestellt, so kann der Ausschlag der Simulation problemlos festgehalten und betrachtet werden. Der

obere Filter deckt alle positiven Werte ab, der untere Filter alle negativen Werte beginnend bei „-1“.

Im Gegensatz zu SENT werden beim PSI5-Tab zwei Tabellen verwendet. Die obere Tabelle stellt dabei alle Sensormesswerte aus dem Bereich 1 des 10 Bit Wertebereiches der Datenregion A dar (siehe *Tabelle 2-16*). In der unteren Tabelle werden die Initialisierungswerte sowie Status & Fehlermeldungen aus den Bereichen 2 und 3 des 10 Bit Wertebereiches dargestellt. Die Aufteilung skaliert mit der Anzahl der Bits, wie in Kapitel *2.2.2.4* beschrieben, entsprechend mit, sodass die obere Tabelle immer den Bereich 1 und die untere Tabelle den Bereich 2 und 3 abbildet. Am Beispiel der *Abbildung 4-10* kann die Aufteilung der Tabellen mit den zugehörigen Datenwerten betrachtet werden. Die einzelnen Datenfelder werden in jeweils getrennten Spalten dargestellt. Zusätzlich werden die Daten als Binärwert dargestellt. In der letzten Spalte wird immer das Paritätsbit bzw. die CRC-Prüfsumme angezeigt.

Die Unterscheidung, ob die Daten in den Bereich 1 oder 2 bzw. 3 fallen, erfolgt durch die ersten 10 Bits beginnend beim MSB der Datenregion A. Wie im Kapitel *4.1.6.2* beschrieben, übermittelt der Mikrocontroller die PSI5-Daten immer als 28 Bit Wert. Wichtig ist dabei, dass die Werte vom LSB beginnend aufgefüllt werden. Die Daten werden von der Software entsprechend der Einstellungen extrahiert. Anmerkung: Die aktuelle Konfiguration wird im Hintergrund zwischengespeichert, damit während einer Neukonfiguration die Daten nicht fehlerhaft aufgeteilt werden.

Auch hier können alle Felder und Datenwerte nur innerhalb der zulässigen Grenzen konfiguriert werden. Beispielweise muss die Bitanzahl für die Datenregion A im Bereich von 10 und 24 liegen. Bei „Bit – Gesamt“ darf nur zwischen 10 und 28 Bit gewählt werden. Wie in *Abbildung 4-11* dargestellt, kann beispielweise bei der „TX-Konfiguration“ das „Status“-Feld nur ausgewählt werden, wenn weitere Bits bei „Bit – Gesamt“ hinzugefügt werden. Auch spezifische Einschränkungen werden durch die Software realisiert. Bei der Verwendung des ELMOS sind maximal 24 Bit möglich und der asynchrone Modus kann nicht verwendet werden, weshalb dieser in *Abbildung 4-11* deaktiviert ist. Die Datenwerte werden ebenfalls entsprechend der eingestellten Bits beschränkt. Zusätzlich wird der Filter aus *Abbildung 4-10* bei einer Änderung der Datenregion A automatisch angepasst. Dies hat den Vorteil, dass er sofort die möglichen Konstellationen bzw. freien Felder ersichtlich sind und eine fehlerhafte PSI5-Konfigurationen ausgeschlossen ist.

In der *Abbildung 4-12* ist ein Beispiel für die Kombination mehrerer Datenfelder dargestellt.

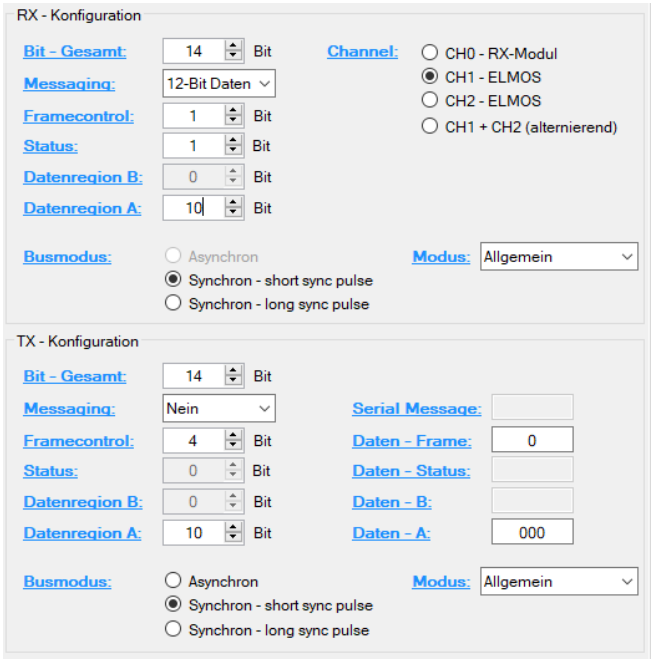

Abbildung 4-11: Konfigurationsbeispiele

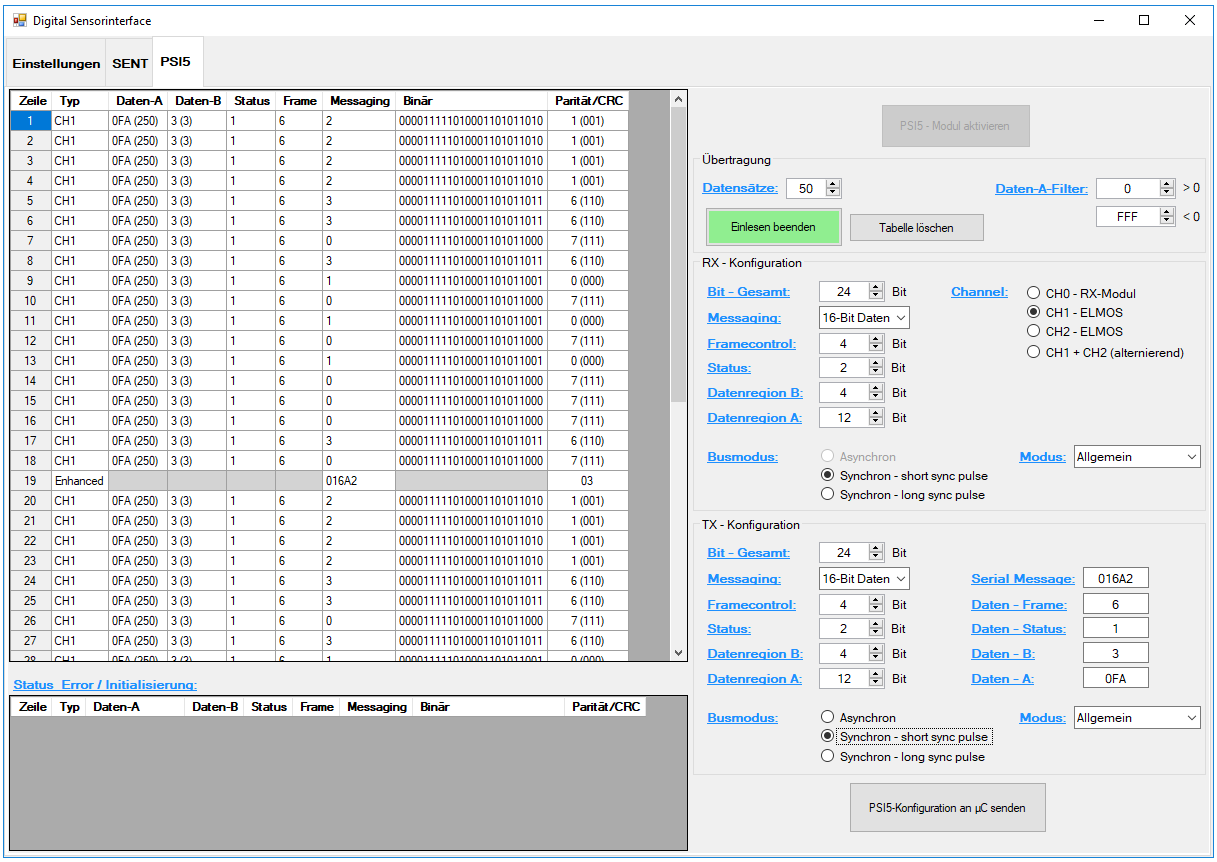

Abbildung 4-12: Darstellung bei mehreren Bits und Datenfeldern

Wie im Kapitel 4.1.5.3 beschrieben, können am ELMOS bei PSI5-Übertragungsfehler die Fehlerinformationen aus den jeweiligen Registern ausgelesen werden. Diese Fehlerregister werden an die PC-Software weitergleitet (siehe Kapitel *4.1.6.2*). Ein Beispiel ist in *Abbildung 4-13* dargestellt. In der Spalte „Typ“ wird die jeweilige Registeradresse und in der Spalte „Binär“ der 16 Bit Registerinhalt zur Verfügung gestellt. In der Spalte „Frame“ wird der Registerinhalt auch als HEX-Wert dargestellt. Mit Hilfe des Datenblattes [26] kann anschließend über die Spalte „Binär“ der Fehler identifiziert werden.

In *Abbildung 4-13* haben die Fehler folgende Bedeutung:

- Register 0x26: Fehler am „Channel 1“ (Bit 0 ist gesetzt)
- Register 0x27: Es wurde kein PSI5-Frame am „Channel 1“ im Zeitslot 1 empfangen
- Register 0x28: „leakage to GND“ – In diesem Fall wurde kein Sensor angeschlossen

Zeile	Typ	Daten-A	Daten-B	Status	Frame	Messaging	Binär	Parität/CRC
1	ELMOS Fehler - Register: 28				0200		0000001000000000	
2	ELMOS Fehler - Register: 26				0001		0000000000000001	
3	ELMOS Fehler - Register: 27				0004		0000000000000100	
4	ELMOS Fehler - Register: 28				0200		0000001000000000	
5	ELMOS Fehler - Register: 26				0001		0000000000000001	
6	ELMOS Fehler - Register: 27				0004		0000000000000100	
7	ELMOS Fehler - Register: 28				0200		0000001000000000	
8	ELMOS Fehler - Register: 26				0001		0000000000000001	
9	ELMOS Fehler - Register: 27				0004		0000000000000100	
10	ELMOS Fehler - Register: 28				0200		0000001000000000	
11	ELMOS Fehler - Register: 26				0001		0000000000000001	
12	ELMOS Fehler - Register: 27				0004		0000000000000100	
13	ELMOS Fehler - Register: 28				0200		0000001000000000	
14	ELMOS Fehler - Register: 26				0001		0000000000000001	
15	ELMOS Fehler - Register: 27				0004		0000000000000100	
16	ELMOS Fehler - Register: 28				0200		0000001000000000	
17	ELMOS Fehler - Register: 26				0001		0000000000000001	
18	ELMOS Fehler - Register: 27				0004		0000000000000100	
19	ELMOS Fehler - Register: 28				0200		0000001000000000	
20	ELMOS Fehler - Register: 26				0001		0000000000000001	
21	ELMOS Fehler - Register: 27				0004		0000000000000100	
22	ELMOS Fehler - Register: 28				0200		0000001000000000	
23	ELMOS Fehler - Register: 26				0001		0000000000000001	
24	ELMOS Fehler - Register: 27				0004		0000000000000100	
25	ELMOS Fehler - Register: 28				0200		0000001000000000	
26	ELMOS Fehler - Register: 26				0001		0000000000000001	
27	ELMOS Fehler - Register: 27				0004		0000000000000100	
28	ELMOS Fehler - Register: 28				0200		0000001000000000	

Abbildung 4-13: ELMOS - Registerfehler

5. Beispiele für SENT-/PSI5-Signale

5.1. SENT

5.1.1. Beispiele für SENT-Übertragungen

In diesem Kapitel werden mehrere Übertragungsbeispiele für die SENT-Schnittstelle dargestellt. Die Übertragungssignale wurden durch die eigene SENT-Sendeschnittstelle ausgesendet und von eigenen SENT-Empfangsschaltung wieder eingelesen. In der *Abbildung 5-1* ist ein Beispiel für eine Datenübertragung mit 6 Nibble und einem Pause-Impuls dargestellt. Für dieses Übertragungsbeispiel wurde eine Messung der einzelnen Zeitwerte durchgeführt:

Sync.	Status	DN1	DN2	DN3	DN4	DN5	DN6	CRC	Pause
µs	µs	µs	µs	µs	µs	µs	µs	µs	µs
168	36	45	69	60	72	81	42	69	358

Tabelle 5-1: Messwerte für die SENT-Übertragung in Abbildung 5-1

Mit diesen gemessenen Werten kann der übertragene Datenwert berechnet werden. Für die Berechnung muss zuerst die Zeitbasis „tick“ berechnet werden. Die Zeitbasis kann aus dem Synchronisationsimpuls (= „56 ticks“ lang) berechnet werden. In diesem Beispiel beträgt die Zeitbasis exakt 3µs. Anschließend können die einzelnen Nibble-Werte berechnet werden, indem die Werte durch die Zeitbasis dividiert und gerundet werden. Zum Schluss muss noch der verwendete Minimalwert von „12 ticks“ abgezogen werden. In der *Tabelle 5-2* werden die berechneten Werte dargestellt. Die Länge des Pause-Impuls wurde ebenfalls mitgemessen, um den festen Übertragungsrahmen von 1ms zu demonstrieren.

$$t_{tick} = \frac{Sync.}{56} = \frac{168\mu s}{56} = 3\mu s$$

$$Nibble_{Status} = round\left(\frac{Status}{t_{tick}}\right) - 12 = round\left(\frac{36\mu s}{3\mu s}\right) - 12 = 0$$

	Status	DN1	DN2	DN3	DN4	DN5	DN6	CRC
Dezimal	0	3	11	8	12	15	2	11
HEX	0	3	B	8	C	F	2	B

Tabelle 5-2: Berechnete Werte für die SENT-Übertragung in Abbildung 5-1

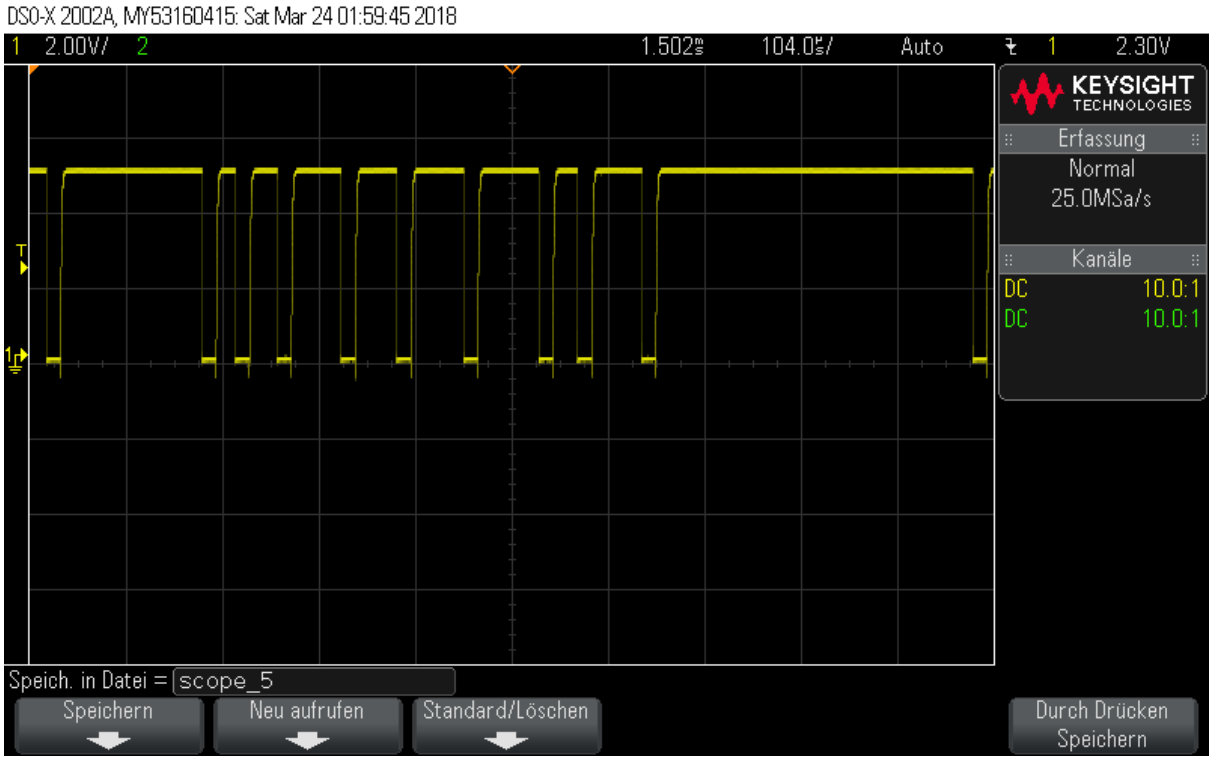

Abbildung 5-1: SENT-Übertragung – 6 Nibble mit Pause, Status = 0x0, Daten = 0x3B8CF2, CRC = 0xB

In *Abbildung 5-2* wird nochmals ein Signal mit denselben Datenwerten wie in *Abbildung 5-1* übertragen, allerdings ohne den Pause-Impuls.

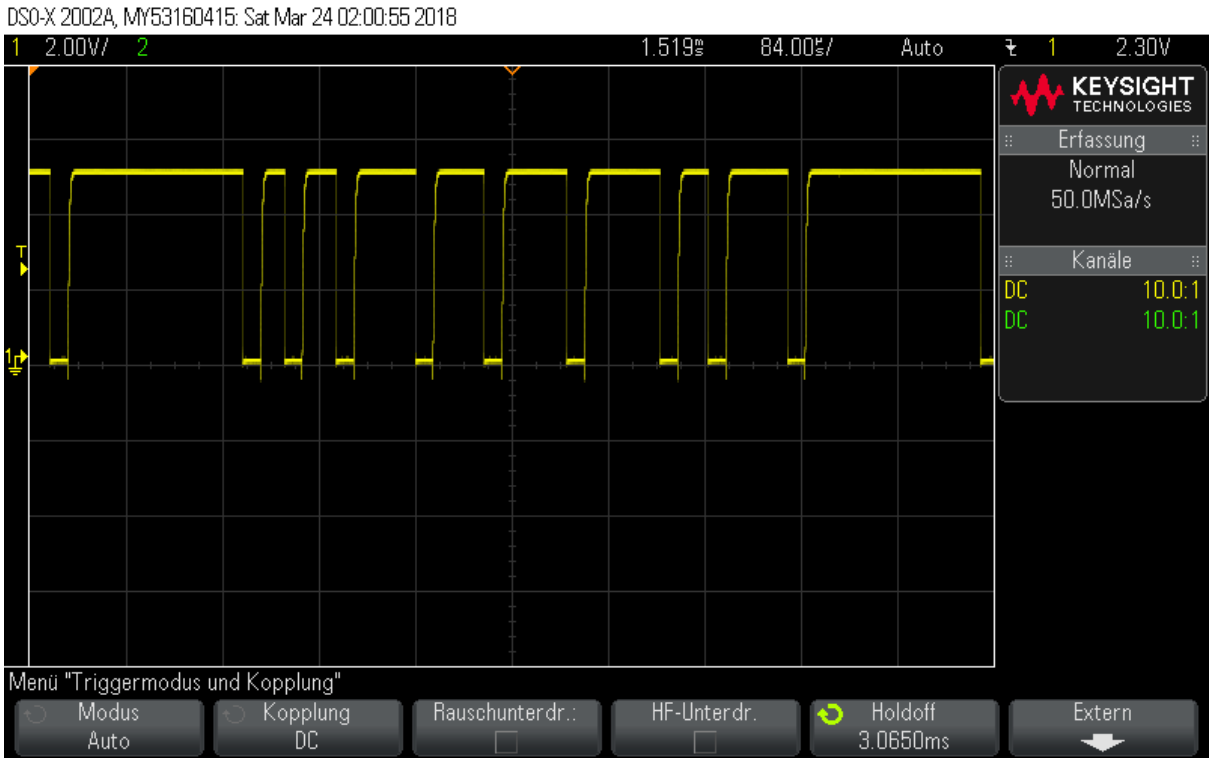

Abbildung 5-2: SENT-Übertragung – 6 Nibble ohne Pause, Status = 0x0, Daten = 0x3B8CF2, CRC = 0xB

Im Vergleich zu *Abbildung 5-1* wird in *Abbildung 5-3* ein SENT-Signal mit nur 3 Daten-Nibble dargestellt. Durch die aktivierte Pause wird auch hier die Übertragung auf eine feste Länge von 1ms ausgeweitet. In *Abbildung 5-4* ist ein Beispiel mit einem Status-Nibble ungleich Null dargestellt.

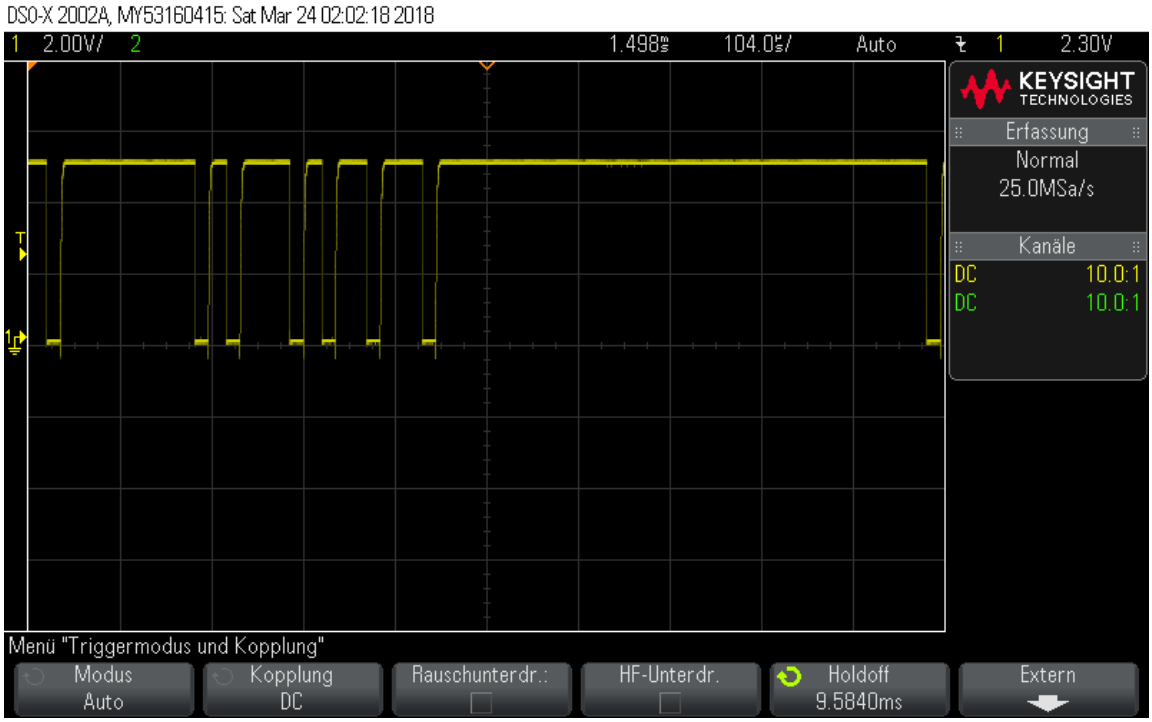

Abbildung 5-3: SENT-Übertragung – 3 Nibble mit Pause, Status = 0x0, Daten = 0xC05, CRC = 0x9

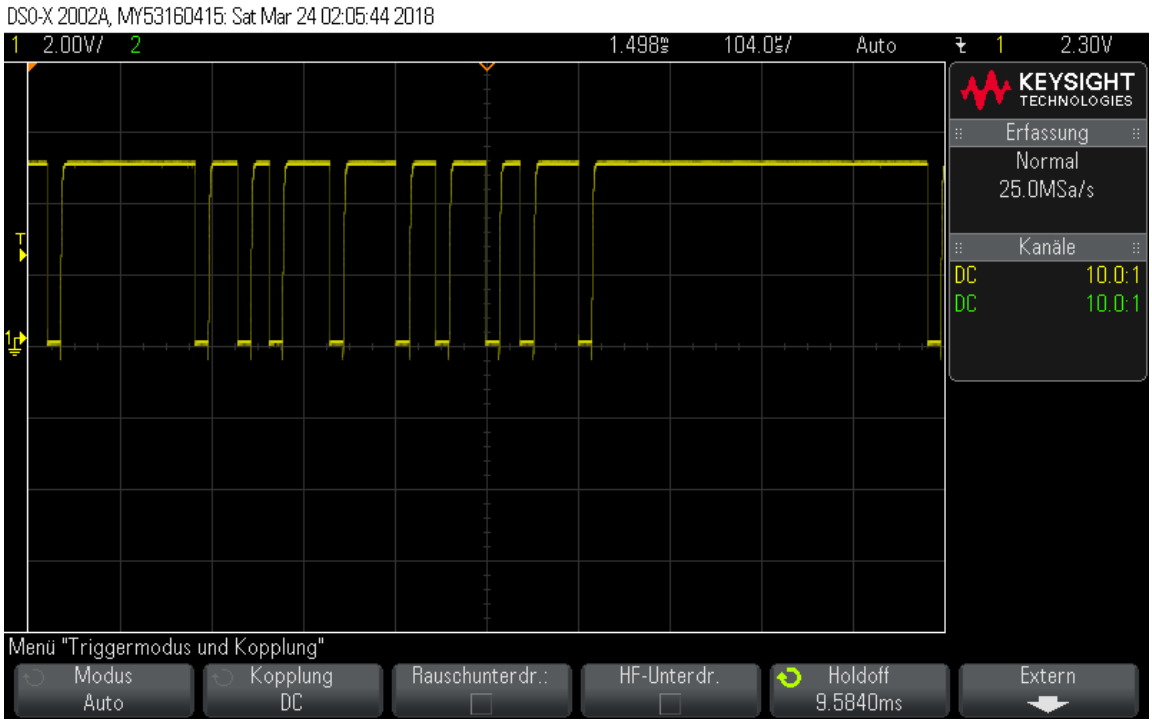

Abbildung 5-4: SENT-Übertragung – 6 Nibble ohne Pause, Status = 0x4, Daten = 0x0BD371, CRC = 0xA

5.1.2. Sensorbeispiel - HAL 2830 / HAL 2833

Als Referenz für die SENT-Übertragungen standen die Sensoren „HAL2830“ und „HAL2833“ der Firma Micronas zur Verfügung. Bei den beiden Sensoren handelt es sich um lineare Hall-Sensoren, die den SENT-Standard in der Version von 2010 unterstützen. In der Standardkonfiguration decken die Sensoren einen Messbereich von ± 40mT ab. Der HAL2830 besitzt eine Auflösung von 12 Bit (3 Daten-Nibble) und verwendet keinen Pause-Impuls. Beim HAL2833 wird hingegen eine Auflösung von 16 Bit (4 Daten-Nibble) und ein Pause-Impuls zur Verfügung gestellt. Über den Signal-Pin besteht die Möglichkeit die Sensoren zu programmieren und verschiedene Parameter wie z.B. Zeitbasis oder Messbereich anzupassen. Erwähnenswert ist, dass die Zeitbasis zwischen 2 und 17,75µs programmiert werden kann, obwohl laut SENT-Norm ein Bereich von 3 bis 90µs vorgegeben ist (Kapitel *2.2.1.1*). Im Datenblatt wird außerdem angeführt, dass der „Low-Pegel“ mindestens 4 „ticks“ lang ist (laut Norm mindestens 5 „ticks“). Beide Sensoren verwenden die „Short Serial Messages“ zur Übermittlung der Seriennummer, Temperatur und diverse interne Systemregisterwerte. Die vollständige Liste aller „Message-IDs“ und die zugehörige Bedeutung kann dem Datenblatt entnommen werden. [31]

Zur Veranschaulichung der Unterschiede zwischen den beiden Sensormodellen wurden insgesamt 4 Messungen durchgeführt. In *Abbildung 5-5* und *Abbildung 5-6* werden die Signale für den HAL2830 dargestellt. Bei der ersten Messung (*Abbildung 5-5*) wurde das Signal ohne direkten Einfluss eines magnetischen Feldes aufgenommen, im Vergleich wurde das zweite Signal in *Abbildung 5-6* einem statischen magnetischen Feld (Kühlschrankmagnet) ausgesetzt. Bei der dritten und vierten Messung wurde analog dazu der HAL2833 verwendet (*Abbildung 5-7*, *Abbildung 5-8*). Aufgrund von äußeren Umwelteinflüssen liefern die Sensoren nicht exakt den Wert „0“. Im Vergleich zum HAL2830 kann beim HAL2833 sehr gut der zusätzliche Pause-Impuls sowie das zusätzliche Daten-Nibble erkannt werden. Der Vergleich von *Abbildung 5-7* und *Abbildung 5-8* zeigt, dass auch bei verschiedenen Datenwerten immer ein konstanter Übertragungsrahmen von 1ms verwendet wird.

Die aufgenommenen Messwerte sind in der *Tabelle 5-3* dargestellt. Im Falle vom HAL2833 wurde der Pause-Impuls mitgemessen, um die gesamte Länge der Übertragung von 1ms festzuhalten. Die berechneten Werte werden in der *Tabelle 5-4* dargestellt. Die Überprüfung

mit der PC-Software zeigt, dass die berechneten Werte mit den empfangenen Werten am Mikrocontroller übereinstimmen.

Auffallend ist, dass beide Sensoren eine identische Zeitbasis von 2,696µs besitzen. Im Datenblatt ist die Standard-Zeitbasis nicht genau definiert. Da im Datenblatt die Tabelle „Table 5-3“ mit einem Wert vom 2,75µs beginnt und beide Sensoren eine Zeitbasis von 2,696µs verwenden, wurde eine Standard-Zeitbasis von 2,75µs angenommen. Aufgrund der 20% Toleranz kann der Sensor mit der eigenen Schaltung, welche nur eine Zeitbasis von 3µs unterstützt, trotzdem ohne Probleme eingelesen werden. [31]

Messung	Sensor	Sync.	Status	DN1	DN2	DN3	DN4	CRC	Pause
Nr.		µs	µs	µs	µs	µs	µs	µs	µs
1	HAL2830	151	32	32	32	35	---	43,5	---
2	HAL2830	151	43	73	69	62	---	69	---
3	HAL2833	151	43	32	32	35	52	59	596
4	HAL2833	151	43	38	70	51	46	43	561

Tabelle 5-3: Messwerte für die Sensoren HAL 2830 / HAL 2833

$$t_{tick} = \frac{Sync.}{56} = \frac{151\mu s}{56} = 2{,}696\mu s$$

$$Nibble_{Status} = round\left(\frac{Status}{t_{tick}}\right) - 12 = round\left(\frac{32\mu s}{2{,}696\mu s}\right) - 12 = 0$$

Messung	Sensor	t_{tick}	Status	DN1	DN2	DN3	DN4	CRC
Nr.		µs						
1	HAL2830	2,696	0	0	0	1	---	4
2	HAL2830	2,696	4	15	14	11	---	9
3	HAL2833	2,696	4	0	0	1	7	10
4	HAL2833	2,696	4	2	14	7	5	4

Tabelle 5-4: Berechnete Datenwerte für die Sensoren HAL 2830 / HAL 2833

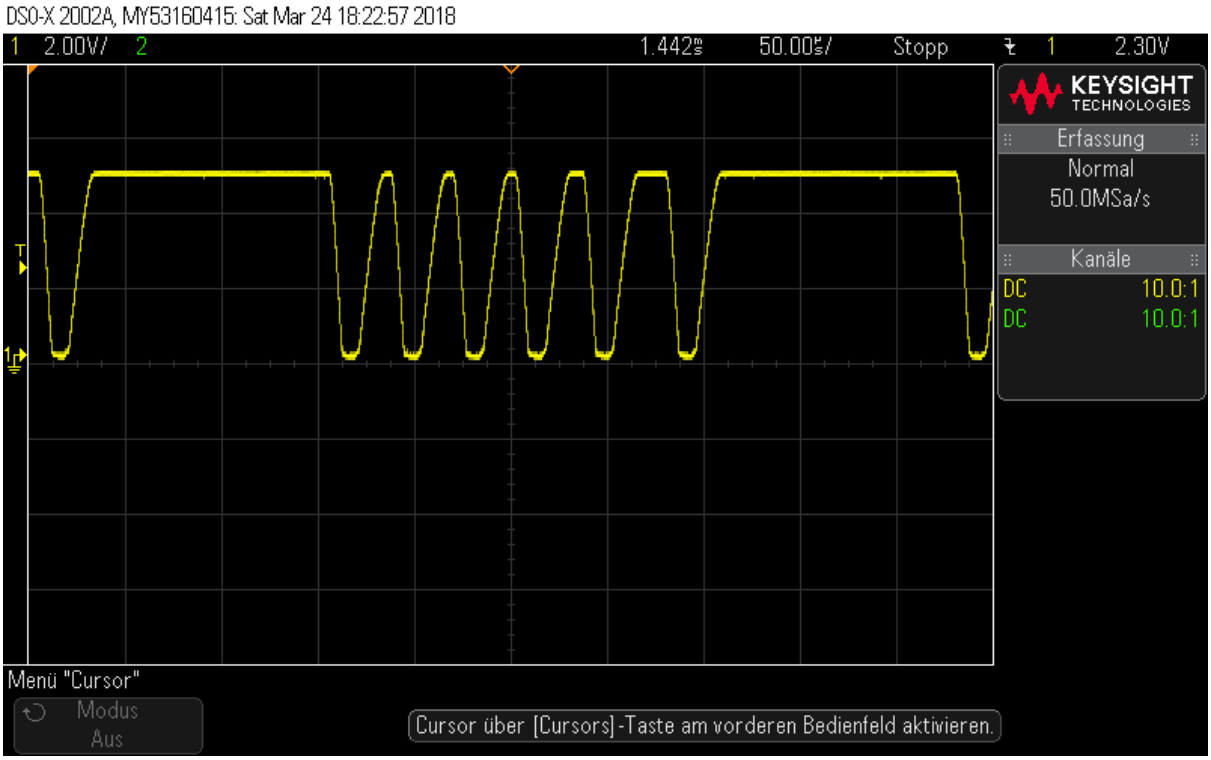

Abbildung 5-5: HAL2830 – Messung 1 – Status = 0x0, Daten = 0x001, CRC = 0x4

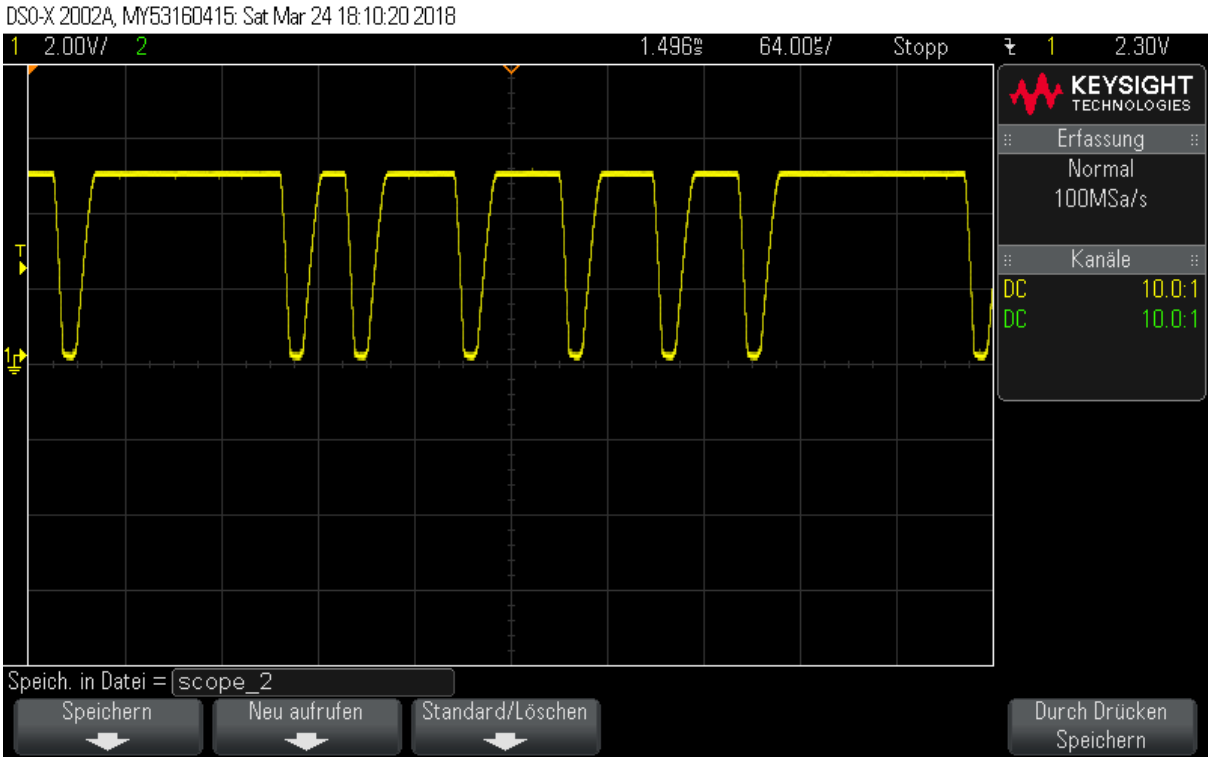

Abbildung 5-6: HAL2830 – Messung 2 – Status = 0x1, Daten = 0xFEB, CRC = 0x9

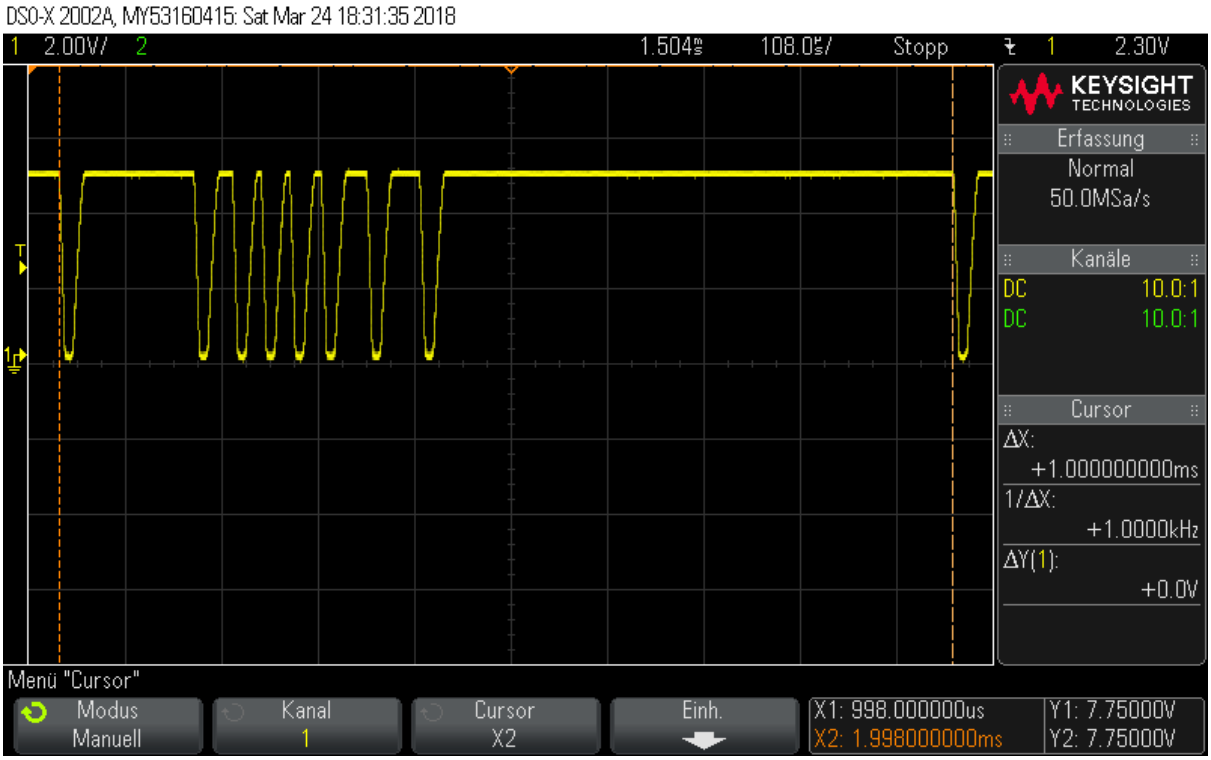

Abbildung 5-7: HAL2833 – Messung 3 – Status = 0x4, Daten = 0x0017, CRC = 0xA

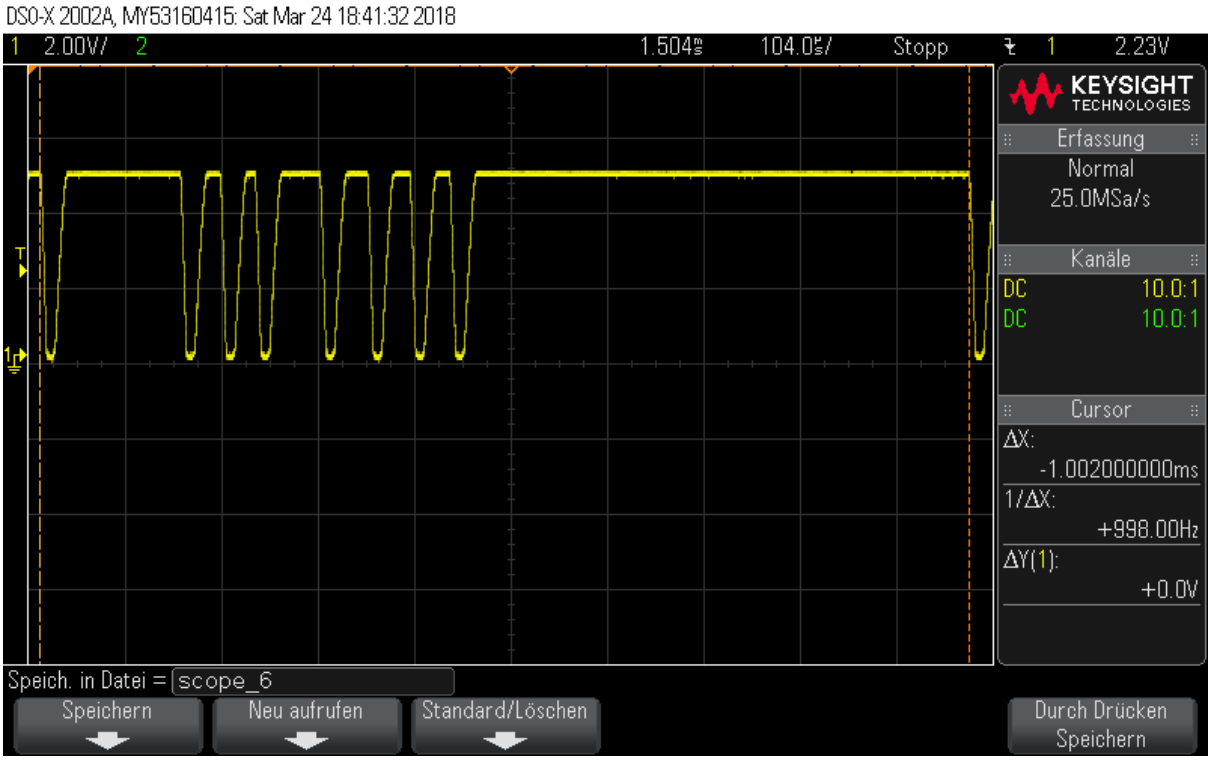

Abbildung 5-8: ***HAL2833 – Messung 4 – Status = 0x4, Daten = 0x2E75, CRC = 0x4***

5.1.3. Sensorbeispiel – KMA215

Der Sensor „KMA215“ der Firma NXP ist ein programmierbarer magnetischer Winkelsensor. Der Messbereich liegt im Bereich von 0 bis 180°. In der Standardkonfiguration verwendet der Sensor eine 3µs Zeitbasis, 6 Daten-Nibble, keine Pause und keine „Serial Messages“. Außerdem wird das „Single Secure“ Format (siehe Kapitel *2.2.1.4*) eingesetzt. Der Sensor unterstützt die SENT-Norm in der Version von 2010. Wie die Sensoren von Micronas kann auch der KMA215 über die Signalleitung programmiert werden. Er bietet dadurch die Möglichkeit einer sehr umfangreichen Konfiguration. Es kann zwischen dem „Single Secure“, „Throttle Position“ und einem „12 Bit Highspeed“ Format gewählt werden. Das „12 Bit Highspeed“ ist ein spezielles Format bei dem 4 Daten-Nibble zu je 3 Bit gesendet werden. Dieses zusätzliche Format wurde in den SENT-Standard erst mit der Version von 2016 aufgenommen, der Sensor wurde aber bereits 2013 / 2014 entwickelt. [32][33]

Für diese drei Formate können mehrere Parameter konfiguriert werden, beispielsweise die Zeitbasis, ein Pause-Impuls sowie „Enhanced Serial Messages“. Alleine beim „Single Secure“ Format sind bereits 20 verschiedene Konfigurationen angeführt. Zusätzlich kann auch noch der Nullpunkt auf die jeweilige Anwendung abgestimmt werden. [32]

Auch für diesen Sensor wurden zwei Messungen durchgeführt. Bei der ersten Messung (*Abbildung 5-9*, *Tabelle 5-5*) wurden die Werte wieder ohne direktes Magnetfeld aufgenommen, bei der zweiten Messung (*Abbildung 5-10*, *Tabelle 5-6*) wurde analog zu den anderen Sensoren ein Kühlschrankmagnet zur Erzeugung eines Magnetfeldes verwendet.

Messung	Sync.	Status	DN1	DN2	DN3	DN4	DN5	DN6	CRC
Nr.	µs	µs	µs	µs	µs	µs	µs	µs	µs
1	167	36	35	50	39	47	59	79	50
2	167	36	79	45	41	57	56	36	61

Tabelle 5-5: Messwerte für den Sensor KMA215

$$t_{tick} = \frac{Sync.}{56} = \frac{167}{56} = 2{,}98\mu s$$

$$Nibble_{Status} = round\left(\frac{Status}{t_{tick}}\right) - 12 = round\left(\frac{36\mu s}{2{,}98\mu s}\right) - 12 = 0$$

Messung	t_{tick}	Status	DN1	DN2	DN3	DN4	DN5	DN6	CRC
Nr.	µs								
1	2,98	0	0	5	1	4	6	15	5
2	2,98	0	15	3	2	7	7	0	8

Tabelle 5-6: Berechnete Datenwerte für den Sensor KMA215

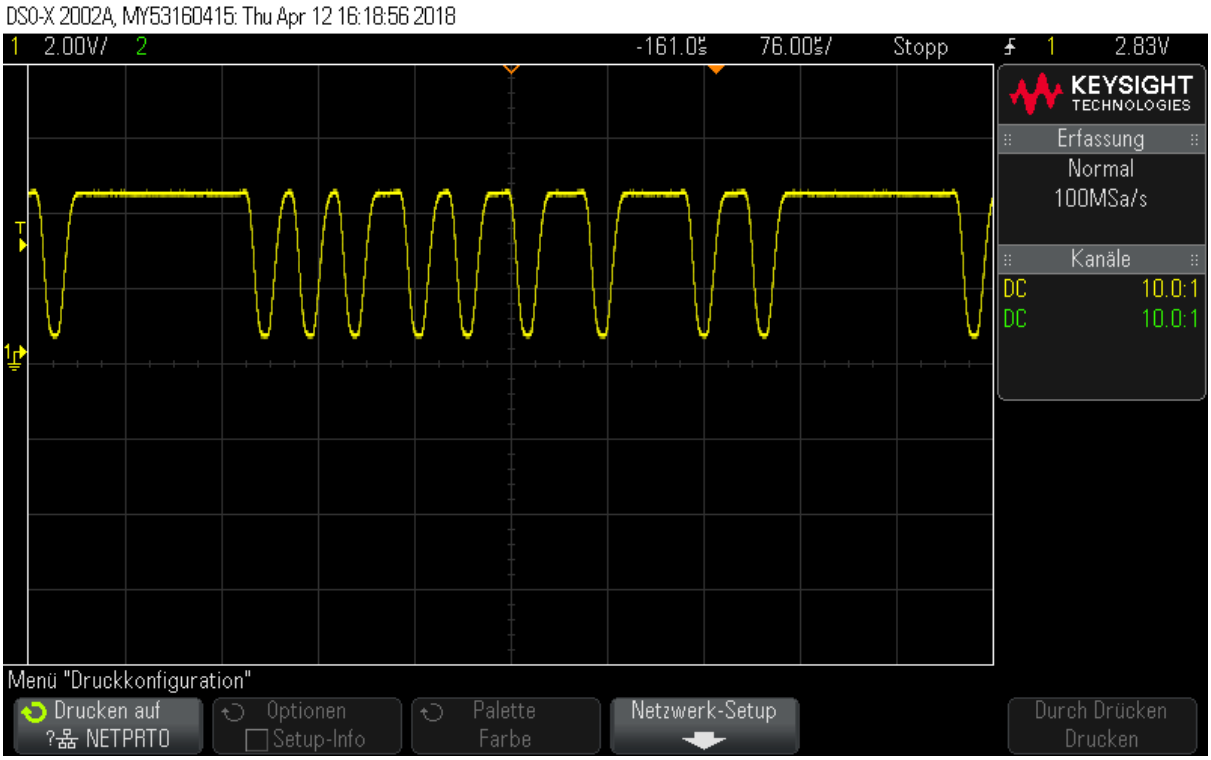

Abbildung 5-9: KMA215 – Status = 0x0, Daten = 0x05146F, CRC = 5

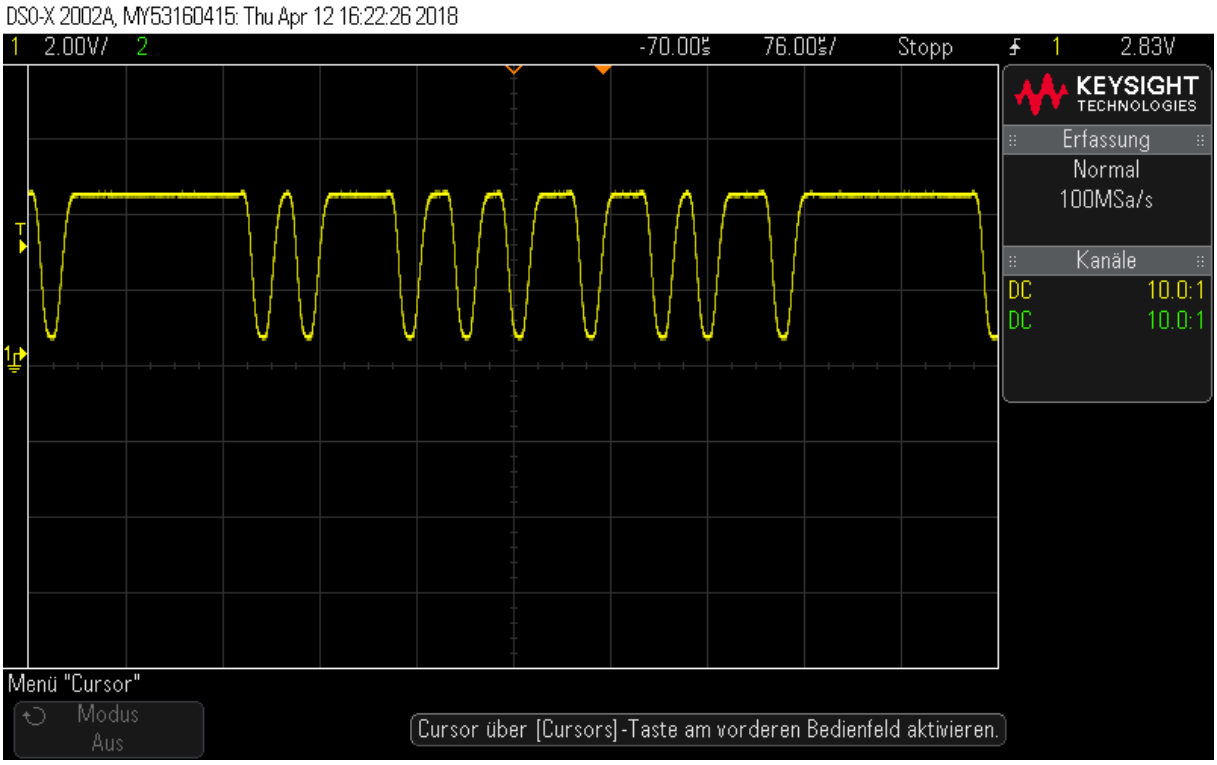

Abbildung 5-10: KMA215 – Status = 0x0, Daten = 0xF32770, CRC = 8

5.1.4. Sensorbeispiel – TLE4998S3

Für den letzten praktischen Test beim SENT-Protokoll stand der Sensor „TLE4998S3“ der Firma Infineon zur Verfügung. Auch bei diesem Sensor handelt es sich um einen programmierbaren linearen Hall-Sensor. Er unterstützt einen maximalen Messbereich von ± 200mT mit drei einstellbaren Bereichen: ± 50mT, ± 100mT und ± 200mT. Für die Übermittlung der Daten verwendet der TLE4998S3 eine Zeitbasis von 3µs, 6 Daten-Nibble, keine Pause und keine „Serial Messages“ [34]. Die 6 Daten-Nibble unterteilen sich dabei wie folgt [34]:

- DN1 – DN4 = 16 Bit Auflösung für das Hall-Signal
- DN5 – DN6 = Temperaturmesswert

Allerdings müssen bei diesem Sensor einige wichtige Punkte für das SENT-Protokoll beachtet werden. Im Gegensatz zur Definition der SENT-Norm wird im Datenblatt des TLE3998S3 die Zeit für den Low-Pegel mit 9µs angegeben, was 3 „ticks“ bei einer 3µs Zeitbasis entspricht (laut Norm mindestens 5 „ticks“). Ein weiterer, jedoch weniger gravierender Punkt ist die Handhabung des Status & Kommunikations-Nibble. Da keine „Serial Messages“ verwendet werden, wird das Status & Kommunikation-Nibble laut Datenblatt für andere Statusinformationen genutzt. In Bit 0 und 1 wird der Messbereich angeführt, in Bit 2 ein

Fehlerindikator im Falle einer Überspannung und in Bit 4 wird, um einen „Reset“ anzuzeigen, in der ersten Nachricht eine „1“ gesendet. Der wichtigste und folgenreichste Unterschied ist die Berechnung der CRC-Prüfsumme. Gemäß SENT-Standard (Kapitel *2.2.1.1*) erfolgt die Prüfung nur anhand der Daten-Nibble, das Status-Nibble wird in die Berechnung nicht miteinbezogen. Der TLE3998S3 sieht aber eine CRC-Prüfung inkl. Status-Nibble vor. Außerdem erfolgt bei der CRC-Berechnung keine abschließende XOR-Verknüpfung von vier zusätzlichen Null-Bits. Ein Praxistest konnte dieses Verhalten bestätigen, die Daten können über den herkömmlichen Weg nicht berechnet werden. Aus diesem Grund wurde für den Empfang der Modus „TLE3998S3“ geschaffen (siehe Kapitel *4.1.4.2*). Außerdem dürfen keine „Serial Messages“ aktiviert werden, da es sonst zu Fehlermeldungen aufgrund der Zweckentfremdung des Status & Kommunikation-Nibble kommt. Der Aufbau des Status- und Kommunikations-Nibble sowie die Aufteilung der Daten-Nibble kann in *Abbildung 5-11* betrachtet werden. [34]

Der Sensor wurde bereits im Jahr 2008 von Infineon auf den Markt gebracht und folglich mit einer früheren Version der SENT-Norm entwickelt. Im Zuge der Recherchen konnte nicht festgestellt werden, ob in den früheren Versionen andere Definitionen hinsichtlich Low-Pegel und CRC-Berechnung gültig waren. In der Ausgabe von 2010 ist kein Verweis auf eine notwendige Kompatibilität für Sensoren mit älteren Versionen angeführt. Die alten Ausgaben der SENT-Norm standen für diese Masterarbeit nicht zur Verfügung. Hinsichtlich der „Serial Messages“ werden laut Datenblatt bewusst die Konventionen gebrochen. [34]

Mit den geänderten Rahmenbedingungen war es in weiterer Folge möglich Messungen mit dem Sensor durchzuführen. Die Messwerte und Messergebnisse sind in *Tabelle 5-7* bzw. *Tabelle 5-8* dargestellt. Die Auswertung der Signale (DN5 und DN6 laut *Abbildung 5-11*) ergibt eine Temperatur von 24 bzw. 25°C während der Messung. Bei der ersten Messung wurde der Sensor keinem direkten Magnetfeld ausgesetzt, er lieferte den Wert von „0x8000“ (DN1-DN4). Weitere Messungen ergaben, dass ohne externem Magnetfeld alle Messwerte aufgrund äußerer Einflüsse ca. ± 6mT um diesen Wert schwanken. Der Nullpunkt des Sensors liegt offensichtlich, im Gegensatz zu den bisher getesteten Sensoren, bei einem Wert von 0x8000 (entspricht dem MSB des 16 Bit Wertes). Die SENT-Signale sind für beide Messungen in der *Abbildung 5-12* bzw. *Abbildung 5-13* grafisch dargestellt. In den Abbildungen ist deutlich der zeitlich verkürzte Low-Pegel zu erkennen.

Eine Simulation beider Messwerte mit Hilfe der PC-Software zeigt, dass die CRC-Prüfsumme gemäß SENT-Norm für die erste Messung den Wert „1“ und für die zweite Messung den Wert „12“ ergeben würde.

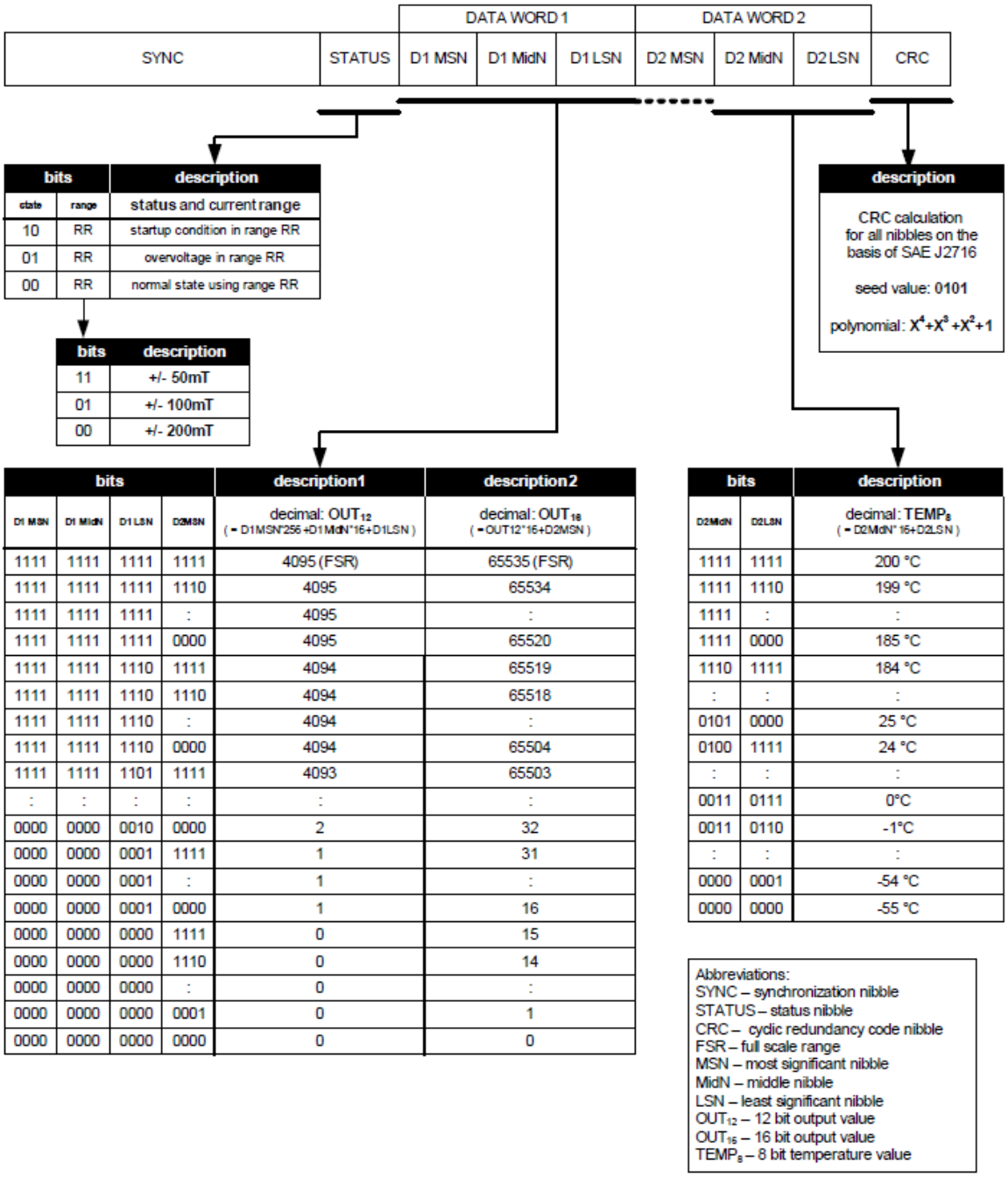

Abbildung 5-11: TLE4998S3 – Aufbau von Status- und Daten-Nibble [34]

Messung	Sync.	Status	DN1	DN2	DN3	DN4	DN5	DN6	CRC
Nr.	µs	µs	µs	µs	µs	µs	µs	µs	µs
1	159	37	57	34	34	34	49	34	34
2	159	37	42,5	34	43	73	46,5	77,5	71

Tabelle 5-7: Messwerte für den Sensor TLE4998S3

$$t_{tick} = \frac{Sync.}{56} = \frac{159}{56} = 2{,}839\mu s$$

$$Nibble_{Status} = round\left(\frac{Status}{t_{tick}}\right) - 12 = round\left(\frac{37\mu s}{2{,}839\mu s}\right) - 12 = 1$$

Messung	t_{tick}	Status	DN1	DN2	DN3	DN4	DN5	DN6	CRC
Nr.	µs								
1	2,839	1	8	0	0	0	5	0	0
2	2,839	1	3	0	3	14	4	15	13

Tabelle 5-8: Berechnete Datenwerte für den Sensor TLE4998S3

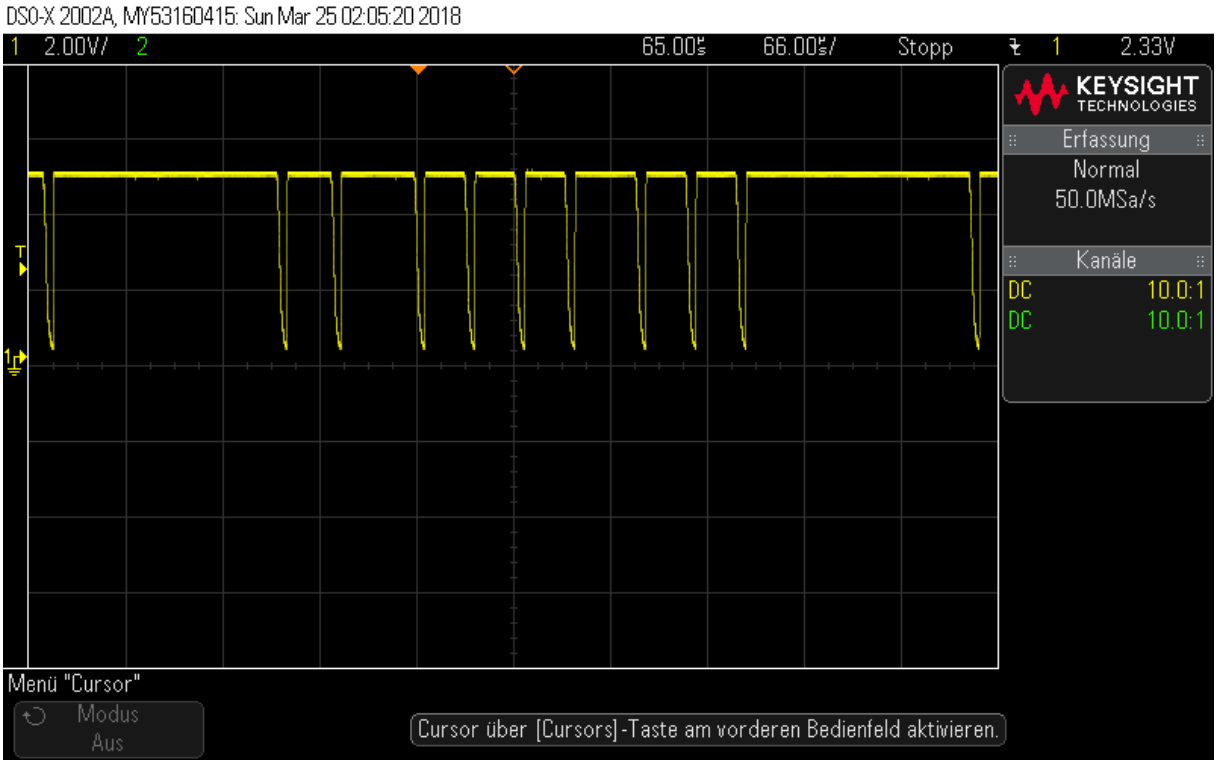

Abbildung 5-12: TLE4998S3 – Status = 0x1, Daten = 0x800050, CRC = 0x0

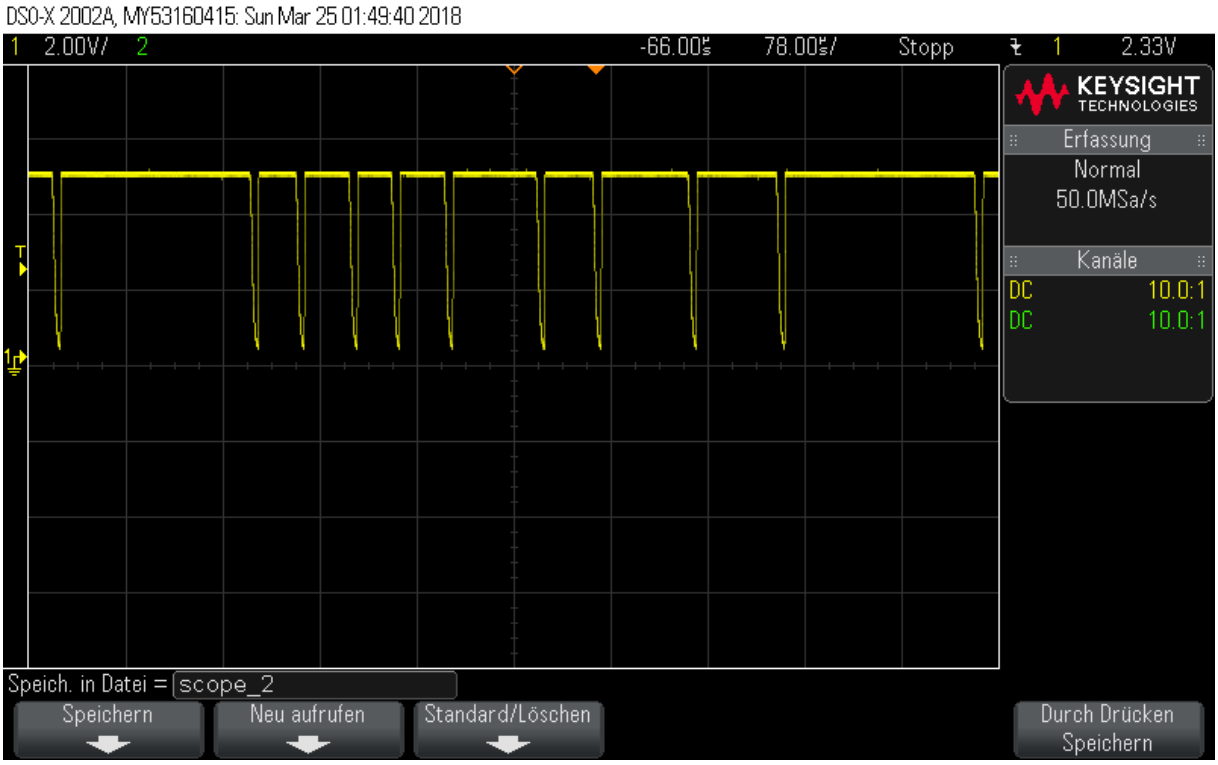

Abbildung 5-13: TLE4998S3 – Status = 0x1, Daten = 0x303E4F, CRC = 0xD

5.2. PSI5

5.2.1. Beispiele für PSI5-Übertragungen

In diesem Kapitel werden mehrere Beispiele für PSI5-Übertragungen, welche mit der selbst entwickelten Sendeschnittstelle übertragen und von der eigenen Empfangsschaltung sowie vom ELMOS-Transceiver empfangen wurden, vorgestellt.

In *Abbildung 5-14*, *Abbildung 5-15* und *Abbildung 5-16* wurde die Nachricht „Sensor Ready“ (0x1E7) im synchronen Betrieb mit einem Oszilloskop aufgenommen. Der Zyklus beträgt 250µs und die Busspannung liegt bei 6,65V. Der Synchronisationsimpuls hat in diesem Beispiel eine Länge von 21µs („short sync pulse“) und die Spannung wird auf 11,3V gehoben.

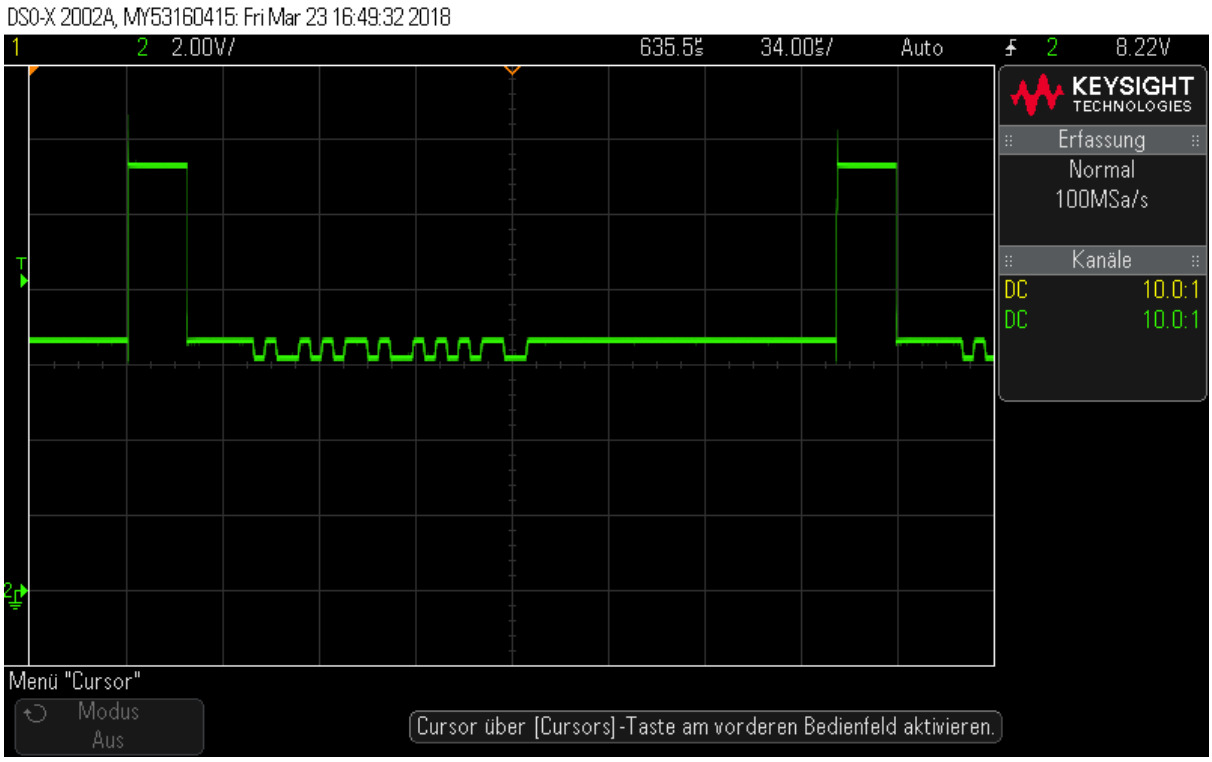

Abbildung 5-14: Synchrone PSI5-Übertragung - Nachricht: 0x1E7

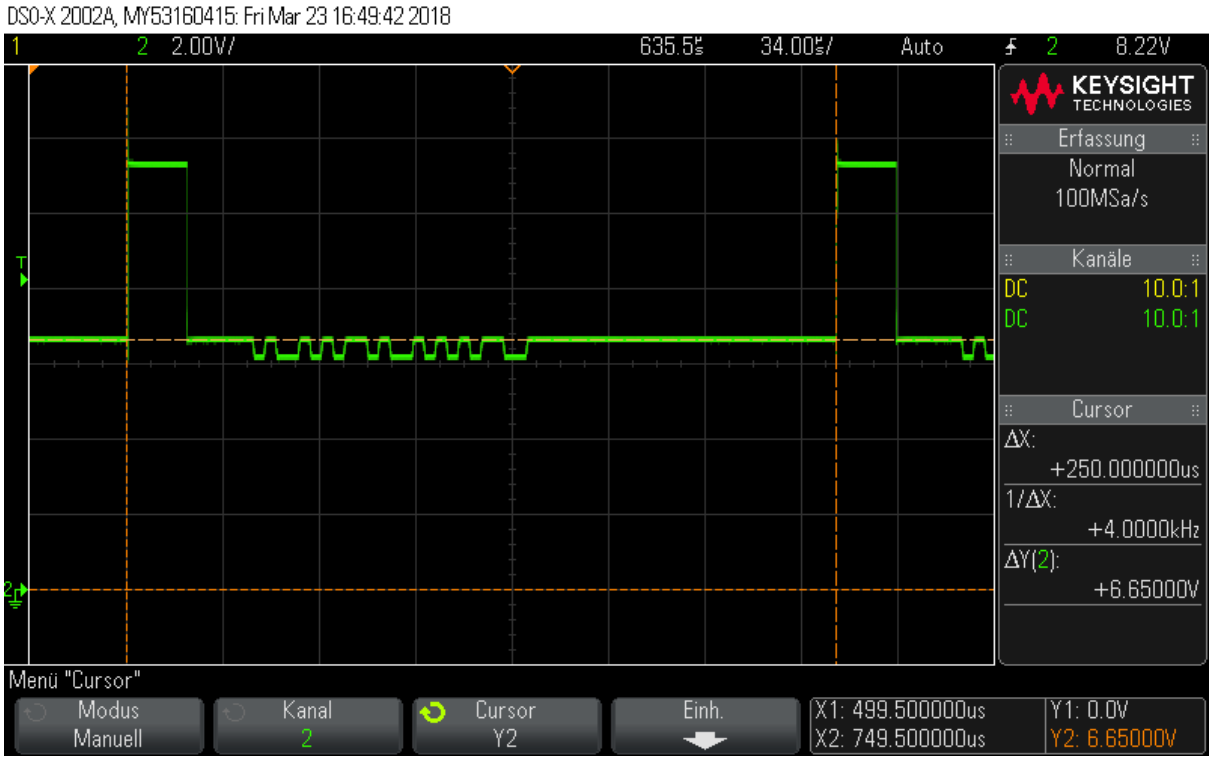

Abbildung 5-15: Messung der Busspannung bei der eigenen Sende- und Empfangsschaltung

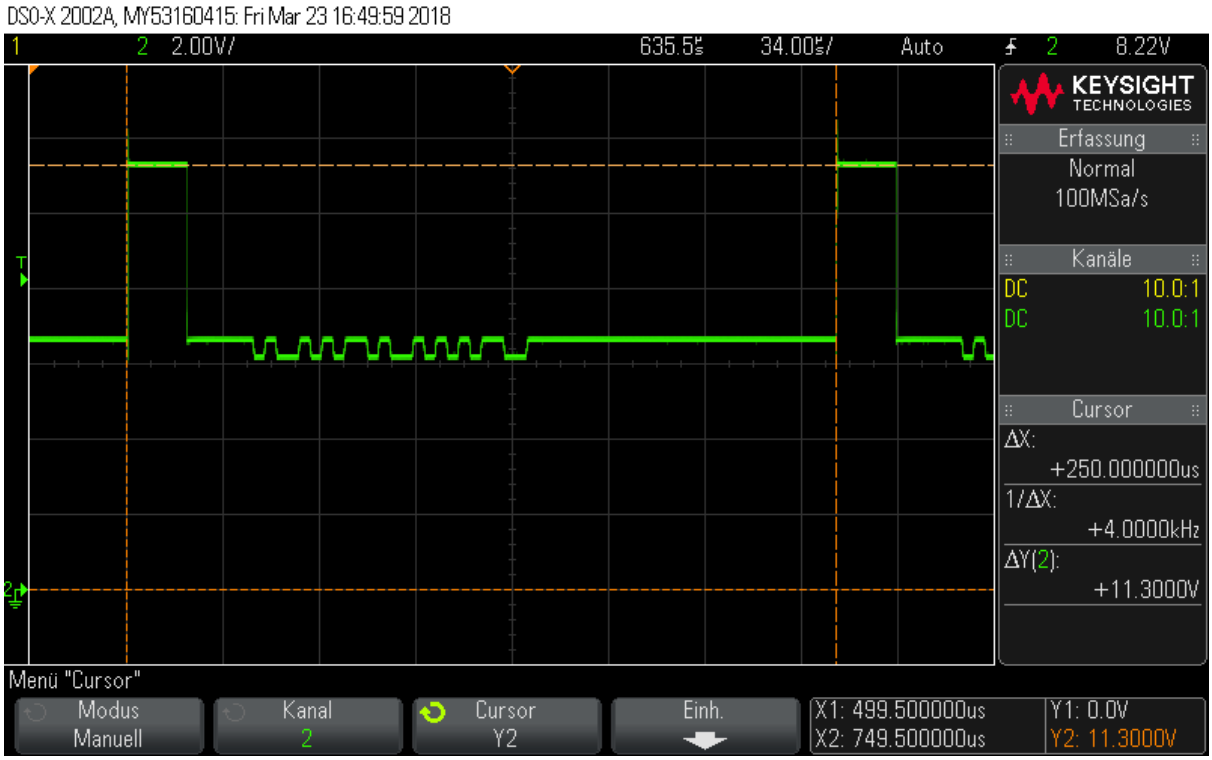

Abbildung 5-16: Messung des Spannungslevels am Synchronisationsimpuls

Für die Manchester-Dekodierung in den nachfolgenden drei Abbildungen muss beachtet werden, dass es sich hier um eine Spannungsmessung an einem Messwiderstand (10 Ω) handelt. Die eigentliche Signalübertragung erfolgt durch eine Strommodulation. Aus diesem Grund ist der Spannungsabfall am Widerstand bei einem High-Pegel $I_{S,High} = I_{S,Low} + \Delta I_S$ größer als bei einem Low-Pegel. Dies hat eine verringerte Busspannung bei einem High-Pegel zur Folge (*Abbildung 5-17*). Für die Signalauswertung muss das Signal also invertiert gegenüber *Abbildung 2-31* betrachtet werden.

Die Durchführung der Manchester-Dekodierung erfolgt indem eine steigende oder fallende Flanke mittig innerhalb der Bit-Zeit T_{Bit} = 8µs ermittelt wird. In der *Abbildung 5-17* bzw. *Abbildung 5-18* sind die Startbits „S0“ und „S1“ markiert. Die fallende Flanke (Achtung: Durch Spannungsmessung invertiert) wird als Bit-Zustand „0“ interpretiert. Der Wechsel des logischen Zustandes von „0“ auf „1“ für das Bit „D1“ lässt sich in *Abbildung 5-19* durch die steigende Flanke erkennen. Wird die Manchester-Dekodierung auf diese Art fortgesetzt, ergibt sich, nachdem die Bitreihenfolge von LSB zu MSB auf MSB zu LSB geändert wurde, der Wert „0x1E7“ mit einer Parität vom Wert „1“.

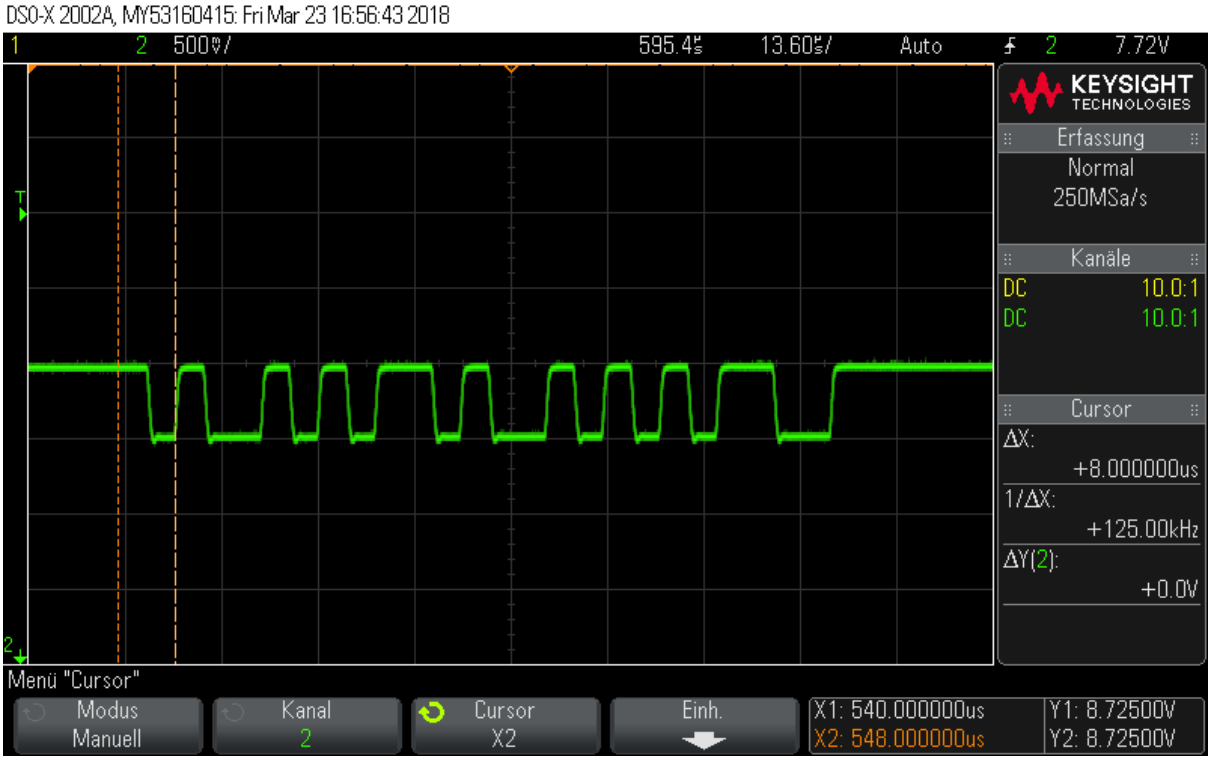

Abbildung 5-17: Startbit „S0“ mit einem Wert von „0“

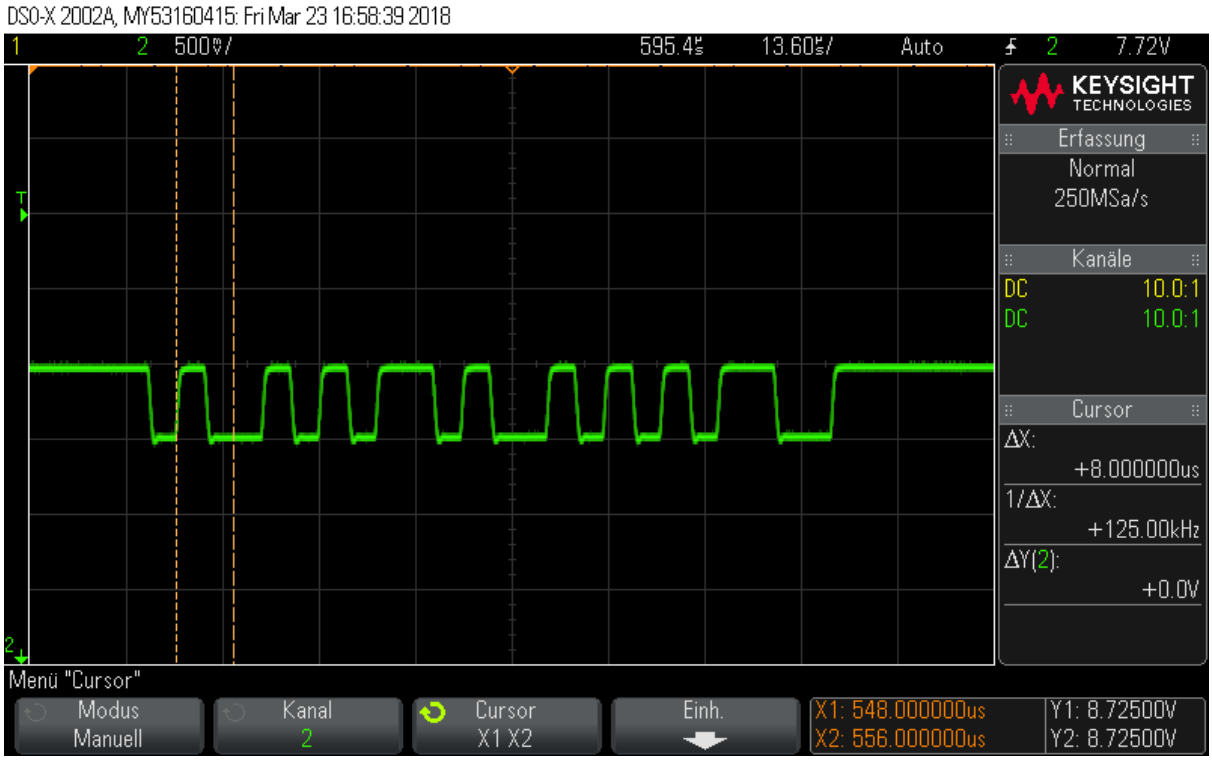

Abbildung 5-18: Startbit „S1“ mit einem Wert von „0“

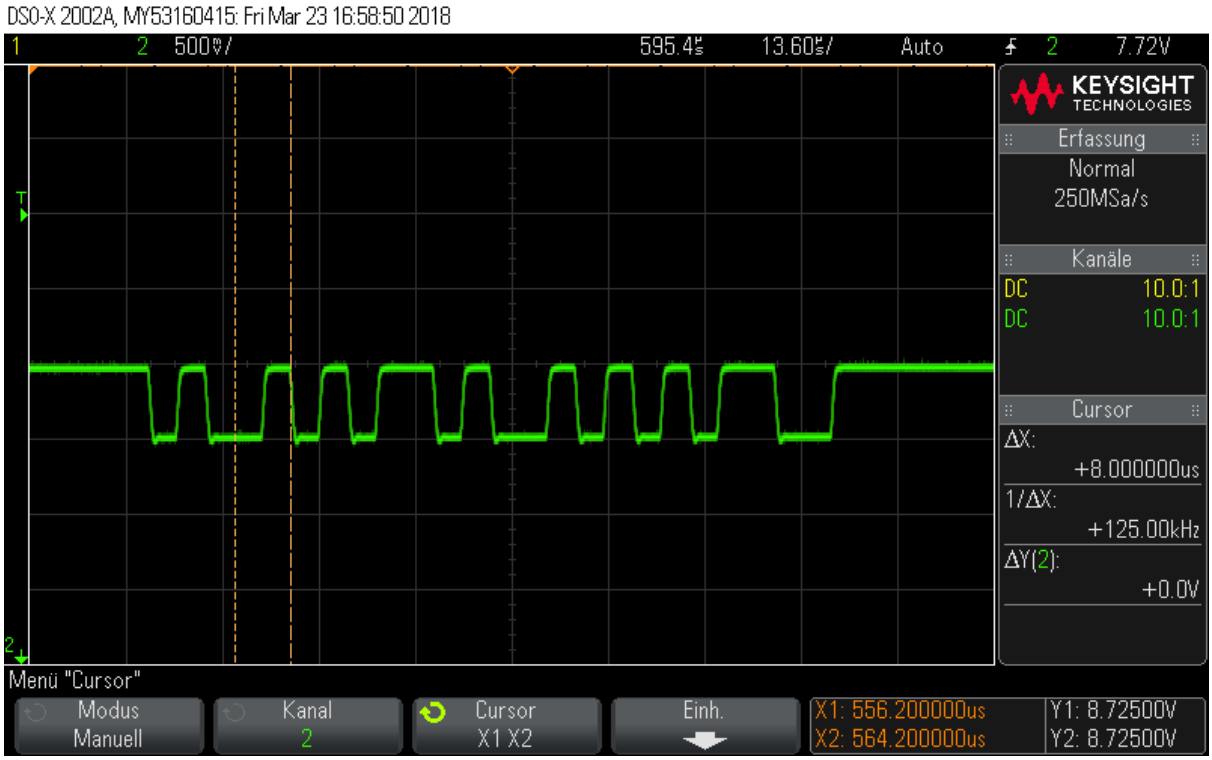

Abbildung 5-19: Bit „D1“ mit einem Pegelwechsel

Zum Vergleich der Betriebsarten wird in *Abbildung 5-20* eine PSI5-Übertragung im asynchronen Betrieb dargestellt. Die Übertragung der 10 Bit Nachricht „0x1E7“ erfolgt kontinuierlich ohne Aufforderung durch den Empfänger.

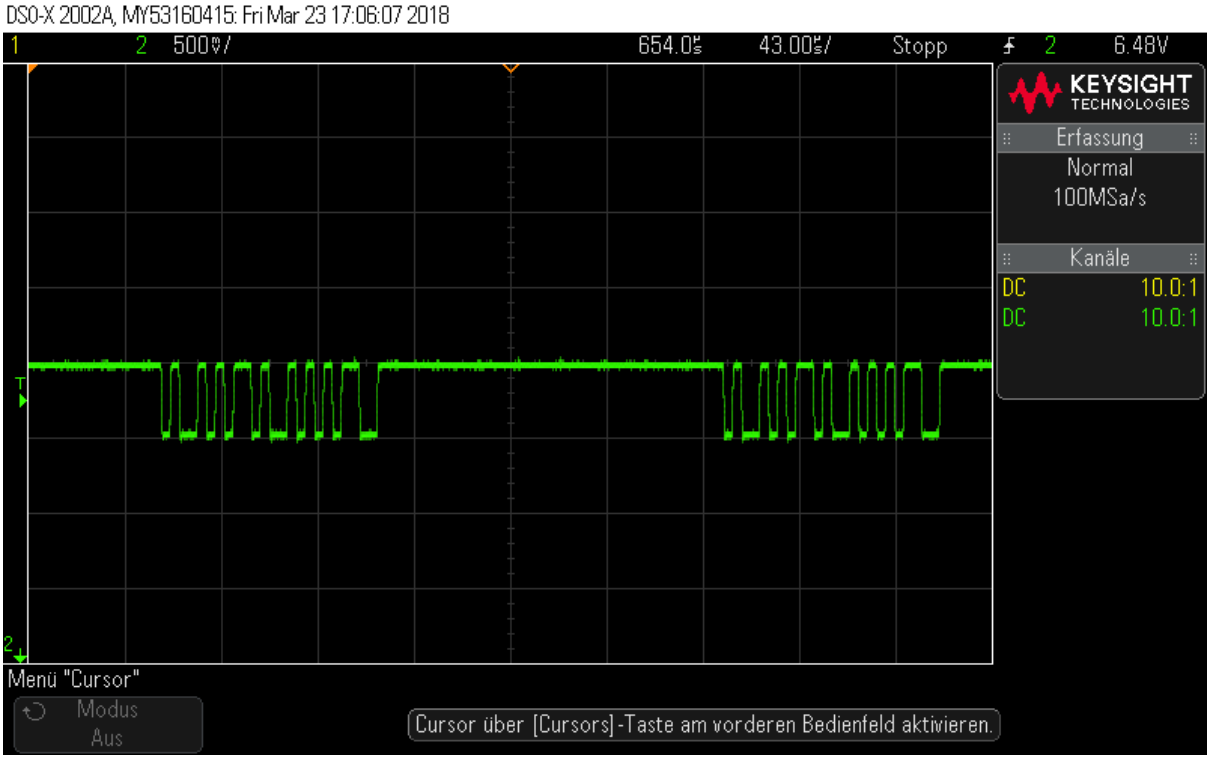

Abbildung 5-20: Asynchroner Betrieb mit der Nachricht 0x1E7

Abbildung 5-21 zeigt ein Beispiel für eine Übertragung mit 28 Bit Nutzdaten und einem Synchronisationszyklus vom 500µs. Als Konfiguration wurden 24 Bit für die Region A (Wert: 0x0B0A24) und 4 Bit für die Region B (Wert: 0x3) verwendet.

Abschließend kann noch ein Beispiel für eine synchrone Übertragung mit zwei Zeitslots in *Abbildung 5-22* betrachtet werden. Die Trennung zwischen den beiden PSI5-Frames erfolgt durch eine Pausenzeit von ca. 16µs. Wie im Kapitel *4.1.5.2* zum Modus „2 Zeitslots“ beschrieben, wird im zweiten Zeitslot immer der invertierte Wert des ersten Zeitslots gesendet. Für den ersten Zeitslot wurde die Nachricht „0x000“ übermittelt. Der invertierte Wert im Zeitslots 2 ist deutlich zu erkennen.

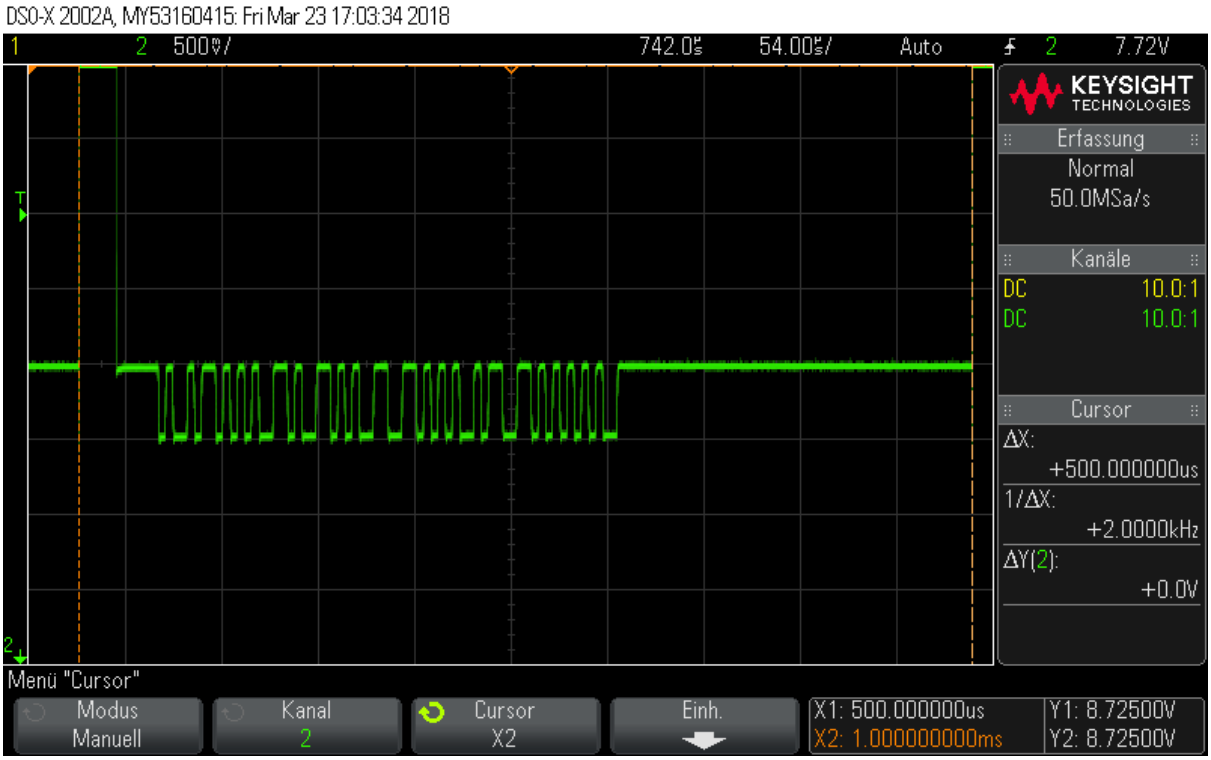

Abbildung 5-21: PSI5-Übertragungsbeispiel mit 28 Bit Nutzdaten

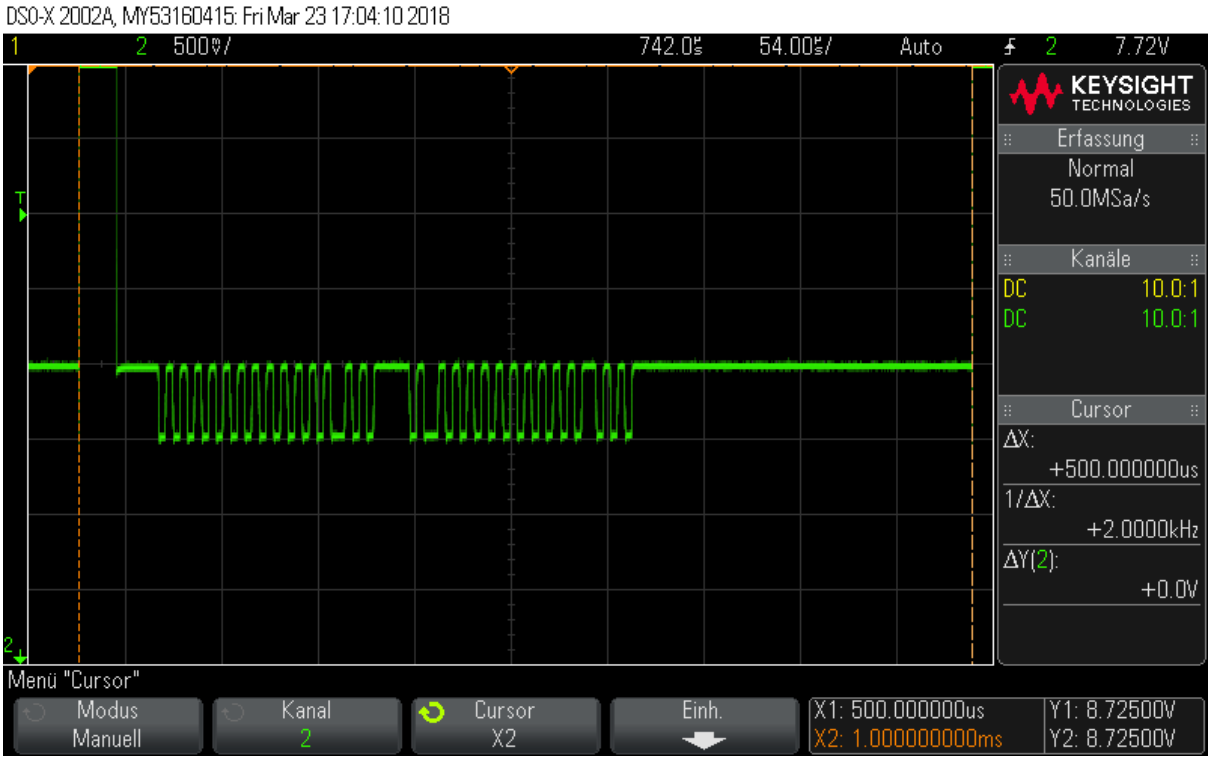

Abbildung 5-22: PSI5-Übertragungsbeispiel mit 2 Zeitslots in einem Zyklus

5.2.2. Sensorbeispiel - ADXL151

Als Referenzsensor stand für die Masterarbeit der PSI5-Sensor „ADXL151“ von Analog Devices zur Verfügung. Dabei handelt es sich um einen Airbag-Sensor mit einem konfigurierbaren Messbereich von ± 120g, ± 240g oder ± 480g. Der Sensor unterstützt den PSI5-Standard in der Version 2.1 mit asynchronem und synchronem Parallel- bzw. Daisy-Chain-Busbetrieb sowie konfigurierbaren 10 oder 16 Bit Nutzdaten. Für den Parallelbus kann der Sensor für einen bestimmten Zeitslot konfiguriert werden. In der Standardkonfiguration ist der Sensor wie folgt konfiguriert: „PSI5-P10P-250/1L“. Eine Änderung der Konfiguration ist über eine SPI-Schnittstelle möglich. [35]

Da der Sensor keine „Serial Messages“ verwendet, werden die 3 Phasen der Initialisierung durchlaufen. Die Phase 1 ist 100ms lang, anschließend folgt Phase 2 mit den Initialisierungswerte und einer Wiederholungsrate von k = 4. Bei den Initialisierungswerten muss beachtet werden, dass diese vom Anwender selbständig konfiguriert werden müssen und von der tatsächlich verwendeten Konfiguration abweichen können. Sollte der Sensor in Phase 2 noch nicht alle Selbsttests abgeschlossen haben, so wird der Code „0x1E8“ („Sensor busy“) ausgesendet. In Phase 3 wird 10-mal entweder der Code „0x1E7“ („Sensor ready“), „0x1E6“ („Sensor ready but unlocked“) oder „0x1F4“ („Sensor defect“) übermittelt. [35]

Für die Masterarbeit wurde die Standardkonfiguration des Sensors belassen. In *Abbildung 5-23* ist ein Beispiel für eine Übertragung mit dem ADXL151 dargestellt. Der Versuchsaufbau mit dem Sensor zeigte, dass Phase 2 mit dem Code „0x1E8“ verlängert wurde. Durch das Belassen des Sensors in der Standardkonfiguration wird der Code „0x1E6“ zehnmal ausgesendet. Für die Auswertung der Sensorsignale sind mit Ausnahme der Initialisierungswerte keine Besonderheiten zu beachten, die PSI5-Signale können wie spezifiziert eingelesen werden.

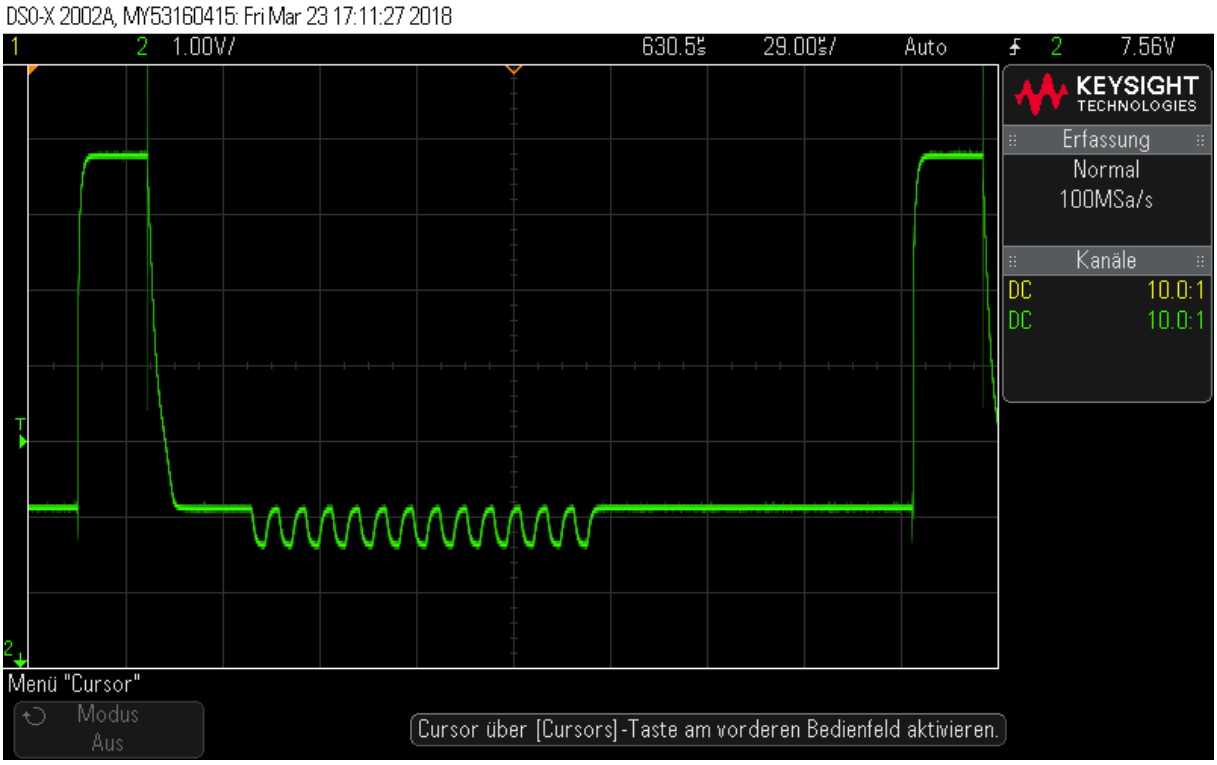

Abbildung 5-23: Beispiel für die PSI5-Übertragung am ADXL151

5.2.3. Sensorbeispiel - AIS1200PS

Beim Sensor „AIS1200PS“ der Firma STMicroelectronics handelt es sich ebenfalls um einen Airbag-Sensor. Es steht hier allerdings nur der asynchrone Modus in der Konfiguration „PSI5-A10P-224/1L“ zur Verfügung. Der Messbereich des Sensors ist mit ±200g definiert. Auch dieser Sensor verwendet die dreiphasige Initialisierungsvariante. Die Phase 1 besitzt eine Länge von 121ms, die Initialisierungswerte werden nur mit der Wiederholungsrate k=3 ausgesendet. Die Länge der zweiten Phase wird vom Hersteller mit 43ms angegeben, insofern die Selbsttests abgeschlossen sind. In der Phase 3 wird entweder der Code „0x1E7“ („Sensor ready“) oder „0x1F4“ („Sensor defect“) zweimal ausgegeben. [36]

Der Praxistest zeigte, dass der Sensor ein nicht definiertes Verhalten in Bezug auf die Strommodulation besitzt. Wie in *Abbildung 5-24* zu sehen ist, hat der Sensor neben dem eigentlichen Low- und High-Pegel zusätzlich einen Ruhepegel. Der Low-Pegel liegt dabei um ca. 718mV (ca. 7,2mA bei 10Ω Messwiderstand) tiefer als der Ruhepegel. Der Referenzsensor ADXL151 sowie die eigene PSI5-Schaltung weisen dieses Verhalten nicht auf. Wie in der Abbildung zu erkennen ist, wird für das Startbit „S0“ direkt vom Ruhepegel zum High-Pegel gewechselt. Im Gegensatz dazu wird jedoch am Ende der Übertragung beim Paritätsbit zuerst

auf den Low-Pegel zurückgewechselt und erst anschließend der Ruhepegel hergestellt. Beim Einlesen an der eigenen PSI5-Empfangsschaltung hat das Verhalten beim Startbit „S0“ zur Folge, dass die minimale Bitzeit von 7,6µs (5% Toleranz bei 8µs) unterschritten wird. Damit der Mikrocontroller die Sensordaten nicht verwirft, wurde für diesen Sensor der eigene Empfangsmodus „AIS1200PS“ (siehe *Tabelle 4-4*) geschaffen. In diesem Modus wird die Bitzeit des Startbits „S0“ nicht überprüft.

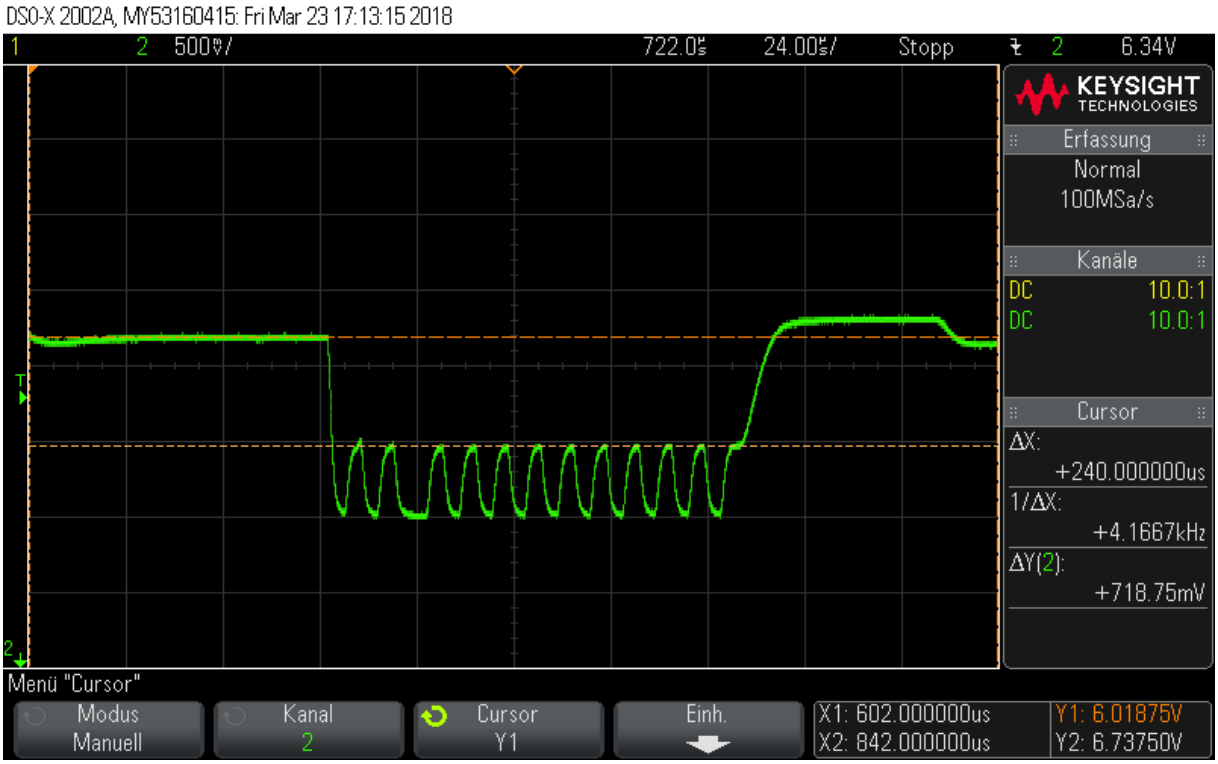

Abbildung 5-24: Beispiel für die PSI5-Übertragung am AIS1200PS

6. Zusammenfassung

Die im Laufe dieser Arbeit gesammelten Erfahrungen und Erkenntnisse zeigen, dass die beiden digitalen Sensorschnittstellen SENT und PSI5 eine zuverlässige Datenübertragung bei gleichzeitig hoher Auflösung ermöglichen.

Bei der SENT-Norm handelt es sich um ein sehr einfaches sowie schlankes Übertragungsfahren. Die Entwicklung einer entsprechenden Hard- und Software kann sehr kostengünstig realisiert werden. Die selbstentwickelte SENT-Hardware zeigte in den durchgeführten Test zuverlässige Resultate, allerdings besitzt die Platine keine ausreichenden EMV-Schutzmaßnahmen. Im Zuge einer Weiterentwicklung wäre die Integration einer Potentialtrennung sowie diverser Schutzmaßnahmen (ESD-Schutz, Verpolungsschutz, etc.) empfehlenswert. Die SENT-Software unterstützt mit dem derzeitigen Stand die wichtigsten Funktionen des SENT-Protokolls. Hinsichtlich der „Serial Messages“ wäre im nächsten Entwicklungsschritt die Implementierung eines typischen Übertragungszyklus zu empfehlen. Dabei sollte die Möglichkeit geschaffen werden, aus den vordefinierten „Message-IDs“ zu wählen und entsprechende Datenwerte festzulegen zu können. Zusätzlich könnten auch eine variable Zeitbasis, ein variabler Pausenwert sowie weitere Applikationsbeispiele integriert werden.

Wie die SENT-Sensorbeispiele im Kapitel 5.1 gezeigt haben, sind bereits einige zuverlässige Sensoren am Markt erhältlich, welche auch individuell konfiguriert werden können. Allerdings hat der praktische Test gezeigt, dass die genaue Analyse der Datenblätter äußerst wichtig ist, um abweichende Übertragungsvarianten im Vorfeld zu erkennen.

Der PSI5-Standard bietet im Vergleich zu SENT einen größeren Funktionsumfang und höhere Datenraten. Die Kosten für die Entwicklung von Hard- und Software sind gegenüber SENT höher anzusetzen. Allerdings hängen die Kosten sehr von der jeweiligen PSI5-Konfiguration ab. Bei der Verwendung des asynchronen Betriebs mit 10 Bit Nutzdaten ist der Entwicklungsaufwand deutlich geringer als bei der Unterstützung aller verfügbaren Optionen, insbesondere bei der Verwendung des „Daisy-Chain“ - Betriebsmodus. Bei der selbstentwickelten PSI5-Hardware gibt es mehrere Verbesserungsmöglichkeiten. Einerseits wäre die vielfach angesprochene Erkennung der Busspannung notwendig, anderseits fehlen auch hier geeignete Schutzmaßnahmen. Der Schaltungsteil rund um den Baustein „U12“ (PSI5-Sendeschnittstelle) in *Abbildung 11-17* ist aufgrund der fehlenden Potentialtrennung nicht

funktionsfähig. Bei den Empfangsschaltungen wäre eine Verbesserung hinsichtlich der Schaltschwellen an den Komparatorschaltungen zu empfehlen. Die Software unterstützt auch bei dieser Schnittstelle die grundlegenden Basisfunktionen für die Übermittlung der Sensordaten. Im nächsten Entwicklungsschritt wäre die Implementierung einer variablen Nutzung von Zeitslots im synchronen Betrieb zu empfehlen. Wie bei SENT könnten auch hier freie Eingaben der „Serial Messages“ umgesetzt werden. Außerdem sollten in der Mikrocontroller-Software zusätzliche Überprüfungen beim Empfang von PSI5-Daten erfolgen. Beispielsweise wäre dies eine Prüfung der unterschiedlichen Substandard-Regeln in Bezug auf die verbleibenden Bits bei Codes aus den Bereichen 2 und 3. Weiters bieten derzeit weder Hard- noch Software einen geeigneten Support für den „Daisy-Chain“ Betrieb. Für eine entsprechende Umsetzung sei an dieser Stelle angemerkt, dass diese Betriebsart einen erheblichen Entwicklungsaufwand mit sich bringt.

Der getestete PSI5-Sensor „ADXL151“ von Analog Devices lieferte sehr gute Ergebnisse, über eine SPI-Schnittstelle kann der Sensor für verschiedene Übertragungsvarianten konfiguriert werden. Allerdings müssen die Initialisierungswerte selbst konfiguriert werden, sie werden nicht anhand der gewählten Konfiguration übermittelt. Beim Sensor „AIS1200PS“ muss hingegen beachtet werden, dass der Sensor für den Ruhe- und Low-Pegel unterschiedliche Spannungen verwendet und deshalb bei der Auswertung Probleme auftreten können.

7. Literaturverzeichnis

[1] A. Wenzlaff, “MADE IN GERMANY – 125 Jahre Automobil,” 2011. [Online]. Available: https://www.muenchen.de/rathaus/dam/jcr:0b136320-6159-4ddf.../mb110403.pdf. [Accessed: 27-Mar-2018].

[2] K. Borgeest, *Elektronik in der Fahrzeugtechnik*. 2015.

[3] E. Union, *RICHTLINIE DER KOMMISSION 2004/104/EG vom 14. Oktober 2004 zur Anpassung der Richtlinie 72/245/EWG des Rates über die Funkentstörung (elektromagnetische Verträglichkeit) von Kraftfahrzeugen an den technischen Fortschritt und zur Änderung der Richtlinie 70*. 2014.

[4] ÖVE/ÖNORM, *EN 55012; Fahrzeuge, Boote und von Verbrennungsmotoren angetriebe Geräte - Funktstöreigenschaften - Grenzwerte und Messverfahren zum Schutz von außerhalb befindlichen Empfängern*. 2010.

[5] ÖVE, “EN 55025; Fahrzeuge, Boote und von Verbrennungsmotoren angetriebene Geräte - Funkstöreigenschaften - Grenzwerte und Messverfahren für den Schutz von an Bord befindlichen Empfängern,” *2009*.

[6] I. O. for Standardization, *ISO 7637-2; Road vehicles -- Electrical disturbances from conduction and coupling -- Part 2: Electrical transient conduction along supply lines only*. 2011.

[7] D. Eddleman, “LTspice Models of ISO 7637-2 & ISO 16750-2 Transients,” 2017. [Online]. Available: http://www.linear.com/solutions/7719. [Accessed: 29-Mar-2018].

[8] I. O. for Standardization, *ISO 16750-1; Road vehicles -- Environmental conditions and testing for electrical and electronic equipment -- Part 1: General*. 2006.

[9] I. O. for Standardization, *ISO 16750-2; Road vehicles -- Environmental conditions and testing for electrical and electronic equipment -- Part 2: Electrical loads*. 2012.

[10] International Organization for Standardization, “ISO 10605; Road vehicles -- Test methods for electrical disturbances from electrostatic discharge,” 2008. [Online]. Available: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=41937. [Accessed: 12-Nov-2016].

[11] H. Hartl, E. Krasser, W. Pribyl, P. Söser, and G. Winkler, *Elektronische Schaltungstechnik*. 2008.

[12] M. O’Hara, “Electrostatic Discharge Testing for Automotive Applications.” [Online]. Available: http://www.autoemc.net/Papers/Other/ESD Conformity Feb07.pdf. [Accessed: 12-Nov-2016].

[13] M. Scholz and V. Vashchenko, *System Level ESD Protection*. 2014.

[14] Littelfuse, “Electrostatic Discharge (ESD) Suppression Design Guide,” 2016. [Online]. Available: http://www.littelfuse.com/~/media/electronics/design_guides/esd/littelfuse_esd_suppression_design_guide.pdf.pdf. [Accessed: 02-Dec-2016].

[15] Littelfuse, “Application Note: ESD and Surge Circuit Protection,” 2011. [Online]. Available: http://www.littelfuse.com/~/media/electronics_technical/application_notes/esd/littelfuse_esd_and_surge_circuit_protection_overview_application note.pdf. [Accessed: 02-Dec-2016].

[16] O. Semiconductor, “Transient Overvoltage Protection,” 2008. [Online]. Available: http://www.onsemi.com/pub_link/Collateral/TND335-D.PDF. [Accessed: 02-Dec-2016].

[17] B. Deutschmann and G. Winkler, “Vorlesungsfolien - Konstruktion und Entwurf elektronischer

Geräte und Systeme; EMV gerechte Leiterplattenentwicklung," 2015.

[18] W. Zimmermann and R. Schmidgall, *Bussysteme in der Fahrzeugtechnik*. 2014.

[19] SAE-International, *SAE J2716 JAN2010; SENT - Single Edge Nibble Transmission for Automotive Applications*. 2010.

[20] PSI5 Steering Committee, "PSI5 Peripheral Sensor Interface for Automotive Applications, Base Standard V2.3," 2018. [Online]. Available: http://psi5.org/fileadmin/user_upload/01_psi5.org/04_Specification/Specifications_PDFs/v2.3/PSI5_spec_v2.3_Base.pdf. [Accessed: 09-Mar-2018].

[21] PSI5 Steering Committee, "PSI5 Peripheral Sensor Interface for Automotive Applications, Base Standard V2.2," 2016. [Online]. Available: http://psi5.org/fileadmin/user_upload/01_psi5.org/04_Specification/Specifications_PDFs/PSI5_spec_v2d2_base.pdf. [Accessed: 09-Mar-2018].

[22] PSI5 Steering Committee, "PSI5 Peripheral Sensor Interface for Automotive Applications, Substandard Powertrain V2.3," 2018. [Online]. Available: http://psi5.org/fileadmin/user_upload/01_psi5.org/04_Specification/Specifications_PDFs/v2.3/PSI5_spec_v2.3_Powertrain.pdf. [Accessed: 09-Mar-2018].

[23] PSI5 Steering Committee, "PSI5 Peripheral Sensor Interface for Automotive Applications, Substandard Airbag V2.3," 2018. [Online]. Available: http://psi5.org/fileadmin/user_upload/01_psi5.org/04_Specification/Specifications_PDFs/v2.3/PSI5_spec_v2.3_Airbag.pdf. [Accessed: 09-Mar-2018].

[24] PSI5 Steering Committee, "PSI5 Peripheral Sensor Interface for Automotive Applications, Substandard Chassis and Safety V2.3," 2018. [Online]. Available: http://psi5.org/fileadmin/user_upload/01_psi5.org/04_Specification/Specifications_PDFs/v2.3/PSI5_spec_v2.3_Chassis_and_Safety.pdf. [Accessed: 09-Mar-2018].

[25] STMicroelectronics, "32F412GDISCOVERY Data brief," 2016. [Online]. Available: http://www.st.com/content/ccc/resource/technical/document/data_brief/group0/dc/09/97/03/3f/3d/44/46/DM00280449/files/DM00280449.pdf/jcr:content/translations/en.DM00280449.pdf. [Accessed: 12-Mar-2018].

[26] Elmos Semiconductor AG, "2 Channel Multi-Mode PSI5 Transceiver - E521.40," 2016. [Online]. Available: http://www.elmos.com/fileadmin/2013/02_products/01_interface/03_psi5/e521-40_elmos_ds.pdf. [Accessed: 15-Mar-2018].

[27] STMicroelectronics, "STM32Cube MCU Package for STM32F4 series." [Online]. Available: https://my.st.com/content/my_st_com/en/products/embedded-software/mcus-embedded-software/stm32-embedded-software/stm32cube-mcu-packages/stm32cubef4.license%3d1521135192921.html. [Accessed: 15-Mar-2018].

[28] STMicroelectronics, "PM0214 Programming manual." [Online]. Available: http://www.st.com/content/ccc/resource/technical/document/programming_manual/6c/3a/cb/e7/e4/ea/44/9b/DM00046982.pdf/files/DM00046982.pdf/jcr:content/translations/en.DM00046982.pdf. [Accessed: 16-Mar-2018].

[29] STMicroelectronics, "STM32 Virtual COM Port Driver," *2018*. [Online]. Available: http://www.st.com/en/development-tools/stsw-stm32102.html. [Accessed: 20-Mar-2018].

[30] Analog Devices, "Datasheet AD5620/5640/5660," 2013. [Online]. Available: http://www.analog.com/media/en/technical-documentation/data-

sheets/AD5620_5640_5660.pdf. [Accessed: 22-Mar-2018].

[31] TDK-Micronas GmbH, “Data Sheet - HAL ® 283x - Linear Hall-Effect Sensor Family with SENT Interface,” 2016. [Online]. Available: https://www.micronas.com/de/system/files/downloads/files/HAL_283x_Linear_Hall-Effect_Sensors_with_SENT_Interface.pdf. [Accessed: 24-Mar-2018].

[32] NXP Semiconductors, “KMA215 Programmable angle sensor with SAE J2716 SENT,” 2014. [Online]. Available: http://www.nxp.com/documents/data_sheet/KMA215.pdf. [Accessed: 24-Mar-2018].

[33] SAE International, “SENT - Single Edge Nibble Transmission for Automotive Applications,” 2016. [Online]. Available: https://www.sae.org/standards/content/j2716_201604/. [Accessed: 24-Mar-2018].

[34] Infineon Technologies AG, “TLE4998S3 Programmable Linear Hall Sensor Sensors,” 2008. [Online]. Available: https://www.infineon.com/dgdl/Infineon-TLE4998S-DS-v01_00-en.pdf?fileId=db3a30431ce5fb52011d3ec7bd4c25bd. [Accessed: 24-Mar-2018].

[35] Analog Devices, *ADXL151 - Low Cost, Single Axis, PS15-Compatible Satellite Sensor*. 2015.

[36] STMicroelectronics, “AIS1200PS - MEMS acceleration sensor: single-axis with PSI5 point-to-point interface,” 2015. [Online]. Available: http://www.st.com/content/ccc/resource/technical/document/datasheet/5c/ed/d7/5d/fa/81/40/c8/DM00084958.pdf/files/DM00084958.pdf/jcr:content/translations/en.DM00084958.pdf. [Accessed: 23-Mar-2018].

8. Abkürzungsverzeichnis

SENT ..... Single Edge Nibble Transmission

PSI5 ..... Peripheral Sensor Interface 5

USB ..... Universal Serial Bus

SPI ..... Serial Peripheral Interface

CAN ..... Controller Area Network

DAC ..... Digital-to-analog converter

Jumper ..... Steckverbindungen

SWD ..... Serial Wire Debug

MCU ..... Microcontroller Unit

ADC ..... Analog-to-digital converter

USART ..... Universal Asynchronous Receiver Transmitter (z.B. RS-232)

LED ..... Light-emitting diode

µC ..... Microcontroller

ESD ..... Electrostatic discharge

SAE ..... Society of Automotive Engineers

MSB ..... Most significant bit

LSB ..... Last significant bit

DN ..... Daten-Nibble

DUT ..... Device under test

SMD ..... Surface-mount device

EMV ..... Elektromagnetische Verträglichkeit

9. Abbildungsverzeichnis

Abbildung 2-1: Prinzipschaltbild [6] und Verlauf [7] - Testimpuls 1 6
Abbildung 2-2: Prinzipschaltbild [6] und Impulsform [7] – Testimpuls 2a (links) und 2b (rechts) 7
Abbildung 2-3: Prinzipschaltbild - Testimpuls 3a und 3b [6] 8
Abbildung 2-4: Impulsform - Testimpuls 3a (links) und 3b (rechts) [7] 8
Abbildung 2-5: Kurzer Spannungseinbruch im 12V-System [9] 12
Abbildung 2-6: Verlauf der Versorgungsspannung für den Rücksetztest [9] 12
Abbildung 2-7: Spannungsverlauf beim Starten des Motors [7] 13
Abbildung 2-8: Prinzipschaltbild zum Lastabwurf [9] 13
Abbildung 2-9: Testimpuls A für den "Load Dump" [7] 14
Abbildung 2-10: Testimpuls B für den "Load Dump" [7] 14
Abbildung 2-11: Prinzipschaltbild für den ESD-Test [13] 18
Abbildung 2-12: Durchkontaktierungen erzeugen einen Spalt [11] 20
Abbildung 2-13: Vermeidung von Leitungen über Spalte in der Massefläche [11] 20
Abbildung 2-14: Stützkondensator auf möglichst kurzem Weg anordnen [11] 21
Abbildung 2-15: Fehlerhafte Anordnung der Filter aus Platzmangel [11] 21
Abbildung 2-16: Anordnung der Anschlüsse auf einer Seite der Platine [17] 22
Abbildung 2-17: Leiterbahn mit hochfrequenten Signalen zwischen Masseleitungen [17] 22
Abbildung 2-18: Beispiel für eine SENT-Botschaft mit 6 Daten-Nibble (DN) und optionaler Pause 24
Abbildung 2-19: Empfohlene Hardwarekonfiguration für SENT 29
Abbildung 2-20: Vollständiges PSI5-Übertragungsprotokoll [21] 33
Abbildung 2-21: Identischer Aufbau der PSI5 „Enhanced Serial Messages“ wie im SENT-Standard [20] 34
Abbildung 2-22: Asynchrone Betriebsart bei PSI5 [20] 35
Abbildung 2-23: Synchroner Busmodus – Allgemein [20] 36
Abbildung 2-24: „Daisy Chain“ Busmodus [20] 37
Abbildung 2-25: Bezeichnungsschema für die Systemkonfiguration [20] 37
Abbildung 2-26: Asynchroner Betrieb in einer Punkt-zu-Punkt Verbindung [20] 38
Abbildung 2-27: Synchroner Betrieb mit einer Parallelbus – Konfiguration [20] 39
Abbildung 2-28: Synchroner Betrieb in einer Universalbus „splice“ – Konfiguration [20] 39
Abbildung 2-29: Synchroner Betrieb in einer Universalbus „pass through“ – Konfiguration [20] 40
Abbildung 2-30: Strommodulation und Manchester-Codierung für die Datenübertragung [20] 41
Abbildung 2-31: Beispiel für eine PSI5-Datenübertragung [20] 41
Abbildung 2-32: Parameter für den Synchronisierungsimpuls im synchronen Betrieb [20] 42
Abbildung 2-33: Initialisierung des Sensors [20] 44
Abbildung 2-34: Initialisierungswerte in Phase 2 [20] 45
Abbildung 2-35: Initialisierung beim Powertrain-Substandard [22] 52
Abbildung 3-1: Blockdiagramm der verwendeten Hardware 55

Abbildung 4-1: ELMOS SPI-Befehl „SPI_Write_Register“ [26] 85
Abbildung 4-2: ELMOS SPI-Befehl „SPI_Get_Data_16b“ [26] 85
Abbildung 4-3: Zeitfenster zum Abrufen der PSI5-Daten über SPI [26] 86
Abbildung 4-4: Bitreihenfolge für den „AD5620“ [30] 102
Abbildung 4-5: Einstellungen für die PC-Software 104
Abbildung 4-6: SENT-Kommunikation in der PC-Software 105
Abbildung 4-7: Darstellung von SENT-Übertragungsfehlern 106
Abbildung 4-8: Popup mit positiver Konfigurationsübernahme 108
Abbildung 4-9: Bei inaktiver PSI5-Schnittstelle können keine PSI5-Parameter konfiguriert werden 108
Abbildung 4-10: PSI5-Kommunikation in der PC-Software 109
Abbildung 4-11: Konfigurationsbeispiele 111
Abbildung 4-12: Darstellung bei mehreren Bits und Datenfeldern 111
Abbildung 4-13: ELMOS - Registerfehler 112
Abbildung 5-1: SENT-Übertragung – 6 Nibble mit Pause, Status = 0x0, Daten = 0x3B8CF2, CRC = 0xB 114
Abbildung 5-2: SENT-Übertragung – 6 Nibble ohne Pause, Status = 0x0, Daten = 0x3B8CF2, CRC = 0xB 114
Abbildung 5-3: SENT-Übertragung – 3 Nibble mit Pause, Status = 0x0, Daten = 0xC05, CRC = 0x9 115
Abbildung 5-4: SENT-Übertragung – 6 Nibble ohne Pause, Status = 0x4, Daten = 0x0BD371, CRC = 0xA 115
Abbildung 5-5: HAL2830 – Messung 1 – Status = 0x0, Daten = 0x001, CRC = 0x4 118
Abbildung 5-6: HAL2830 – Messung 2 – Status = 0x1, Daten = 0xFEB, CRC = 0x9 118
Abbildung 5-7: HAL2833 – Messung 3 – Status = 0x4, Daten = 0x0017, CRC = 0xA 119
Abbildung 5-8: HAL2833 – Messung 4 – Status = 0x4, Daten = 0x2E75, CRC = 0x4 119
Abbildung 5-9: KMA215 – Status = 0x0, Daten = 0x05146F, CRC = 5 121
Abbildung 5-10: KMA215 – Status = 0x0, Daten = 0xF32770, CRC = 8 122
Abbildung 5-11: TLE4998S3 – Aufbau von Status- und Daten-Nibble [34] 124
Abbildung 5-12: TLE4998S3 – Status = 0x1, Daten = 0x800050, CRC = 0x0 125
Abbildung 5-13: TLE4998S3 – Status = 0x1, Daten = 0x303E4F, CRC = 0xD 126
Abbildung 5-14: Synchrone PSI5-Übertragung - Nachricht: 0x1E7 127
Abbildung 5-15: Messung der Busspannung bei der eigenen Sende- und Empfangsschaltung 128
Abbildung 5-16: Messung des Spannungslevels am Synchronisationsimpuls 128
Abbildung 5-17: Startbit „S0“ mit einem Wert von „0“ 129
Abbildung 5-18: Startbit „S1“ mit einem Wert von „0“ 130
Abbildung 5-19: Bit „D1“ mit einem Pegelwechsel 130
Abbildung 5-20: Asynchroner Betrieb mit der Nachricht 0x1E7 131
Abbildung 5-21: PSI5-Übertragungsbeispiel mit 28 Bit Nutzdaten 132
Abbildung 5-22: PSI5-Übertragungsbeispiel mit 2 Zeitslots in einem Zyklus 132
Abbildung 5-23: Beispiel für die PSI5-Übertragung am ADXL151 134

Abbildung 5-24: Beispiel für die PSI5-Übertragung am AIS1200PS 135
Abbildung 11-1: Auswahl / Änderung des Mikrocontrollers 147
Abbildung 11-2: Spezifische "Target" Einstellungen 148
Abbildung 11-3: Konfiguration der Ordner für den Compiler 148
Abbildung 11-4: Diverse Parameter für die Compilierung 149
Abbildung 11-5: Ordner mit diversen Hilfsdateien müssen für den Compiler eingebunden werden 149
Abbildung 11-6: Einstellungen für den Debugger 150
Abbildung 11-7: Einstellungen um den kompilierten Code in den Mikrocontroller zu übertragen 150
Abbildung 11-8: Spezifische Parameter für den „Flash Download“ 151
Abbildung 11-9: Zusätzliche Einstellungen für den Debugger 151
Abbildung 11-10: Auswahl von „Manage Project Items“ 152
Abbildung 11-11: Erstellen der Ordner und Import von vorhandenen Dateien 152
Abbildung 11-12: Alle für das Projekt notwendigen Dateien 155
Abbildung 11-13: Schaltplan Discovery-Board und Spannungsversorgung 162
Abbildung 11-14: RS232 / CAN - Schnittstellen 163
Abbildung 11-15: SENT Schnittstelle für den Empfang und die Aussendung von SENT-Nachrichten 164
Abbildung 11-16: Analoge Ausgabe der SENT/PSI Ausgabe mit Hilfe des AD5620 DAC 165
Abbildung 11-17: PSI5 – Aussendung von PSI5-Nachrichten 166
Abbildung 11-18: PSI5 – Empfangsschnittstelle 167
Abbildung 11-19: ELMOS „E521.40A1“ Schnittstelle 168

10. Tabellenverzeichnis

Tabelle 2-1: Parametertabelle – Testimpuls 1 [6] 6
Tabelle 2-2: Parameter – Testimpuls 2a [6] 7
Tabelle 2-3: Parameter -Testimpuls 2b [6] 7
Tabelle 2-4: Parameter – Testimpuls 3a [6] 9
Tabelle 2-5: Prüfgrade für die Testimpulse im 12V-System [6] 9
Tabelle 2-6: Prüfgrade für die Testimpulse im 24V-System [6] 9
Tabelle 2-7: Parameter für den Testimpuls A [9] 14
Tabelle 2-8: Parameter für den Testimpuls B [9] 15
Tabelle 2-9: Aufbau des Status- & Kommunikations-Nibble 25
Tabelle 2-10: Short Serial Message – Struktur 26
Tabelle 2-11: Enhanced Serial Messages – Struktur 27
Tabelle 2-12: Enhanced Serial Messages - 8 Bit ID und 12 Bit Daten 27
Tabelle 2-13: Enhanced Serial Messages – 4 Bit ID und 16 Bit Daten 27
Tabelle 2-14: Beispiel für die 8 Bit „Message IDs“ bei „Enhanced Serial Messages“ 28
Tabelle 2-15: Bitreihenfolge (0 bis 23 in violett) zur Berechnung der 6 Bit CRC-Prüfsumme 28
Tabelle 2-16: 10 Bit Wertebereich für Datenregion A [20] 44
Tabelle 2-17: Tabelle mit den Pflichtfeldern für Phase 2 [20] 46
Tabelle 2-18: Wertebereich mit 16 Bit für Datenregion A im Airbag-Substandard [23] 48
Tabelle 2-19: Zusätzliche Initialisierungswerte für Phase 2 der Initialisierung [23] 49
Tabelle 2-20: Datenfelder im „Chassis und Safety“ Substandard [24] 51
Tabelle 2-21: Werte für die PSI5-Protokoll-Version bei „Enhanced Serial Messages“ [22] 53
Tabelle 4-1: Übersicht der verwendeten Timer 67
Tabelle 4-2: Übersicht der verwendeten Interrupts sortiert nach Priorität 68
Tabelle 4-3: Modi für die SENT-Kommunikation 70
Tabelle 4-4: Modi für die PSI5-Kommunikation 74
Tabelle 4-5: Initialisierungswerte für den allgemeinen Modus 75
Tabelle 4-6: Initialisierungswerte F1 bis F5 im „Airbag1“ bzw. „Airbag 2“ Modus 76
Tabelle 4-7: Initialisierungswerte F6 bis F9 im „Airbag1“ bzw. „Airbag 2“ Modus 77
Tabelle 4-8: „Message ID“ in den Modi „Powertrain 1 und 2“ 78
Tabelle 4-9: Initialisierungswerte F1 bis F5 im „Chassis & Safety 1“ bzw. „Chassis & Safety 2“ Modus 79
Tabelle 4-10: Initialisierungswerte F6 im „Chassis & Safety 1“ bzw. „Chassis & Safety 2“ Modus 79
Tabelle 4-11: Befehle zur Konfiguration der SENT/PSI5-Schnittstellen 95
Tabelle 4-12: Statusnachricht für die SENT-Konfiguration 96
Tabelle 4-13: Statusnachricht für die SENT-Konfiguration 97
Tabelle 4-14: Fehlercodes und Fehlermeldungen 101
Tabelle 4-15: Übermittlung empfangener SENT- / PSI5-Daten 101
Tabelle 5-1: Messwerte für die SENT-Übertragung in Abbildung 5-1 113
Tabelle 5-2: Berechnete Werte für die SENT-Übertragung in Abbildung 5-1 113

Tabelle 5-3: Messwerte für die Sensoren HAL 2830 / HAL 2833 117
Tabelle 5-4: Berechnete Datenwerte für die Sensoren HAL 2830 / HAL 2833 117
Tabelle 5-5: Messwerte für den Sensor KMA215 120
Tabelle 5-6: Berechnete Datenwerte für den Sensor KMA215 121
Tabelle 5-7: Messwerte für den Sensor TLE4998S3 125
Tabelle 5-8: Berechnete Datenwerte für den Sensor TLE4998S3 125
Tabelle 11-1: Exportierte EAGLE - Bauteilliste 161

11. Anhang

11.1. µVision

Bei der Erstellung eines neuen Projekts müssen einige Einstellungen durchgeführt werden. Die für den Mikrocontroller notwendigen Einstellungen werden in den nachfolgenden Abbildungen dargestellt. Die Einstellungen erreicht man über den Punkt „Project > Options for Target“. Der Mikrocontroller wurde bereits bei der Erstellung des neuen Projekts ausgewählt. Die Standardbezeichnung „Target 1“ wurde auf „STM32F412ZGT“ geändert. In den Abbildungen wurden nur die relevanten Tabs dargestellt. Eine Erklärung der einzelnen Parameter kann über den Button „Help“ eingeholt werden.

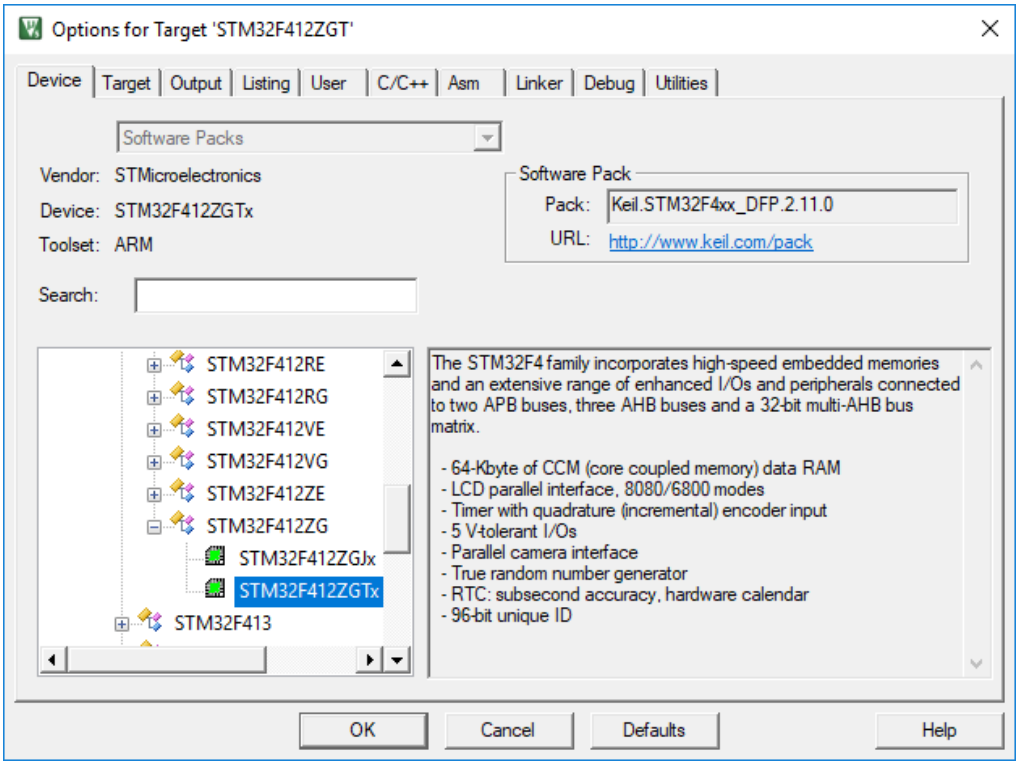

Abbildung 11-1: Auswahl / Änderung des Mikrocontrollers

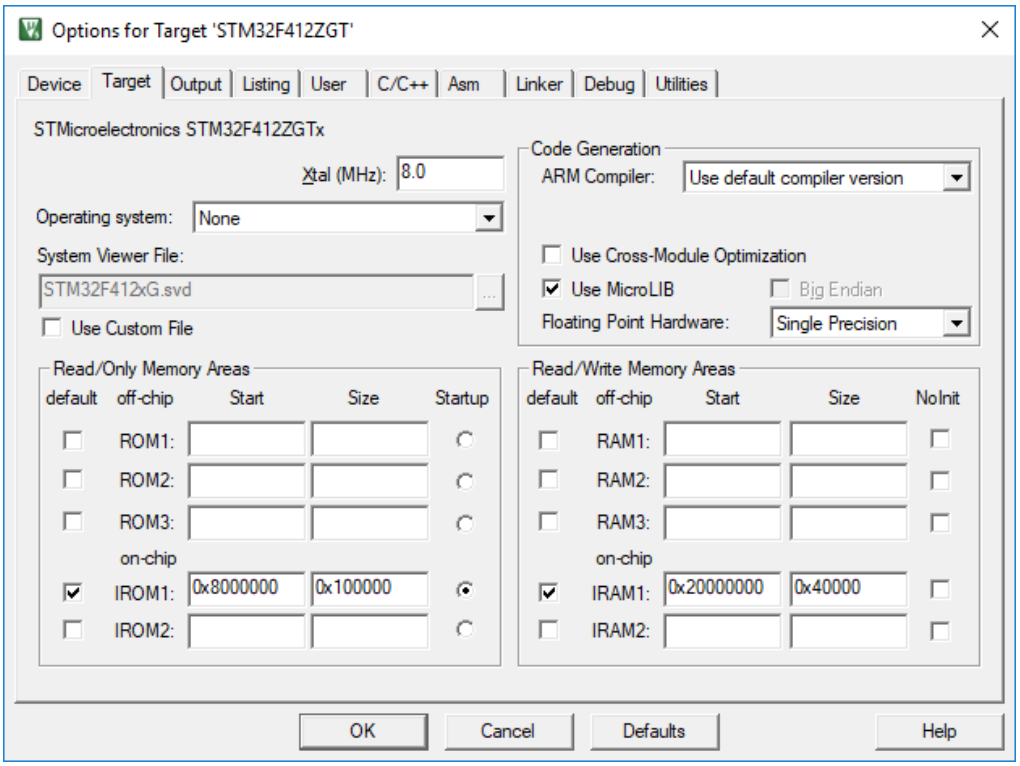

Abbildung 11-2: Spezifische "Target" Einstellungen

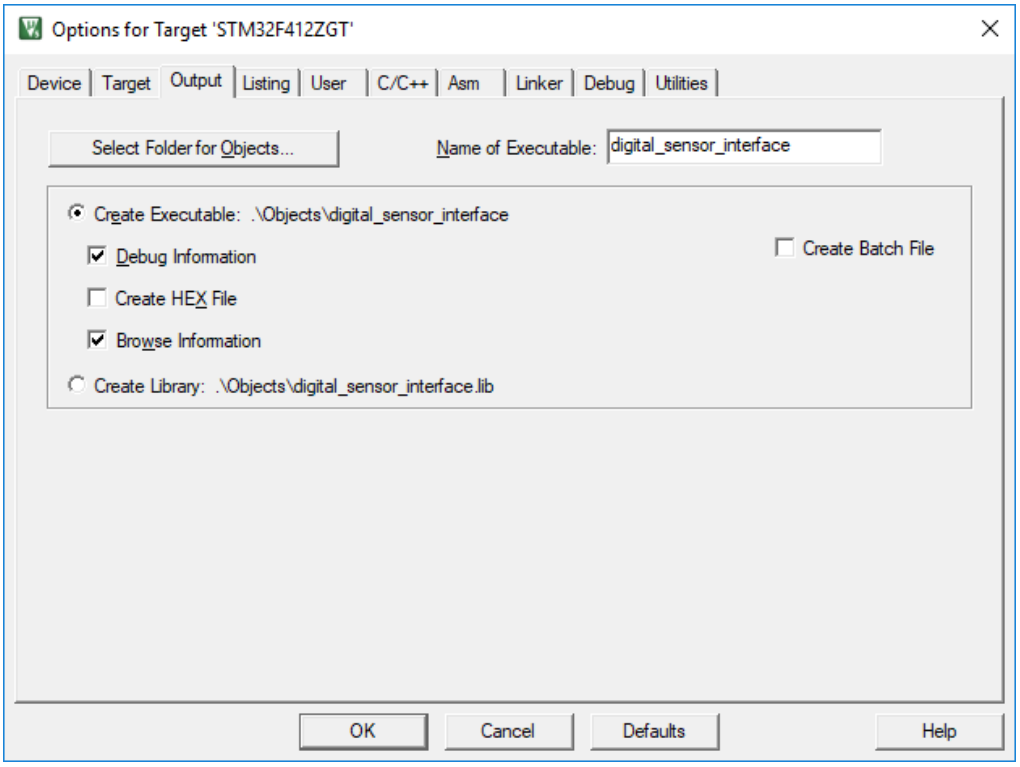

Abbildung 11-3: Konfiguration der Ordner für den Compiler

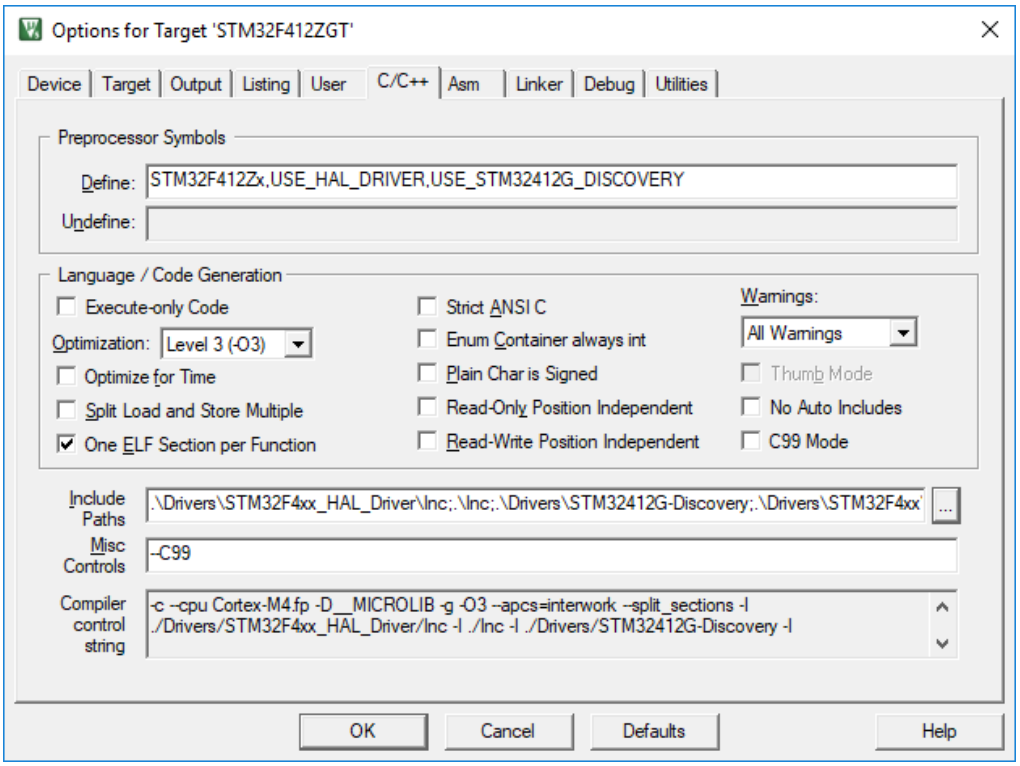

Abbildung 11-4: Diverse Parameter für die Compilierung

Die notwendigen „Include Paths“ werden in der *Abbildung 11-5* dargestellt.

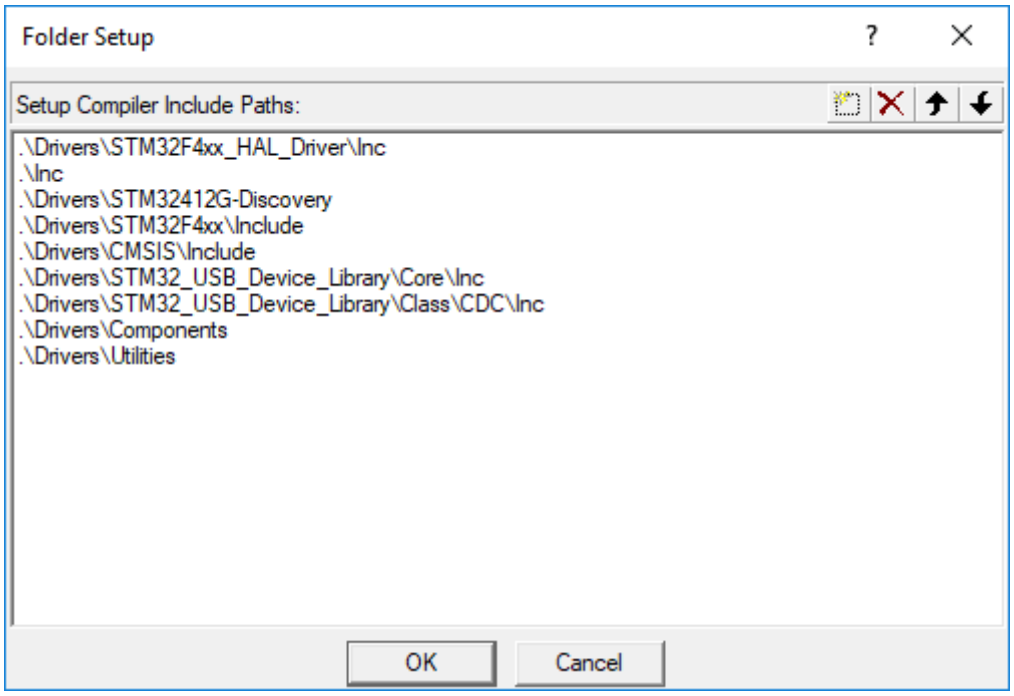

Abbildung 11-5: Ordner mit diversen Hilfsdateien müssen für den Compiler eingebunden werden

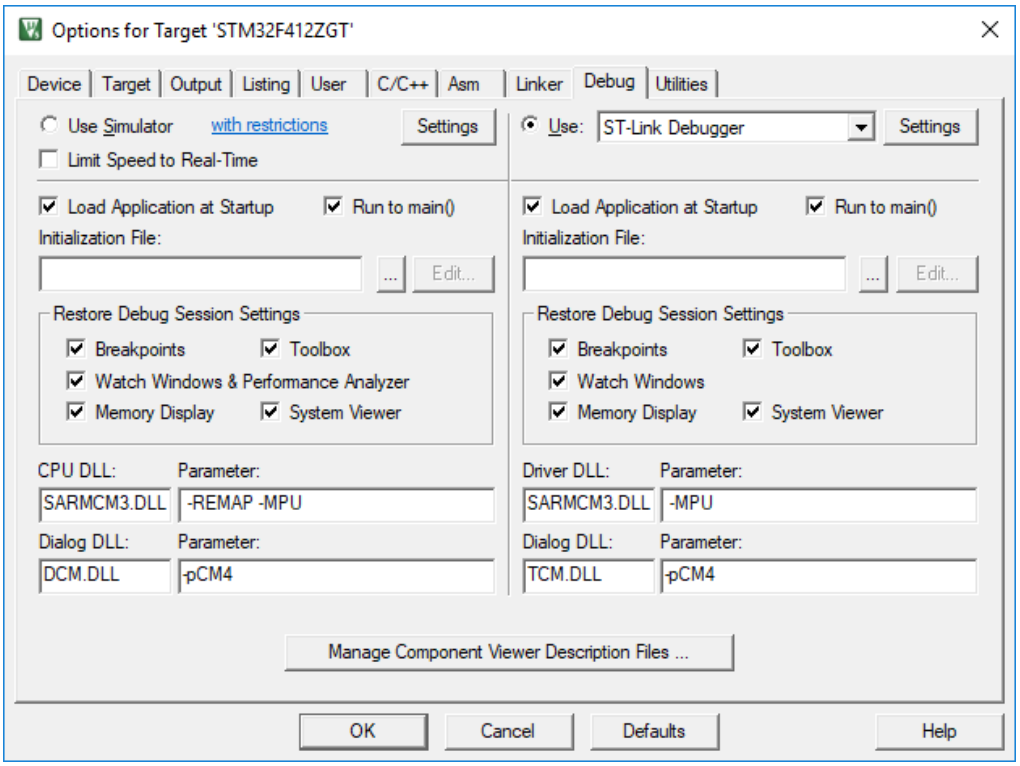

Abbildung 11-6: Einstellungen für den Debugger

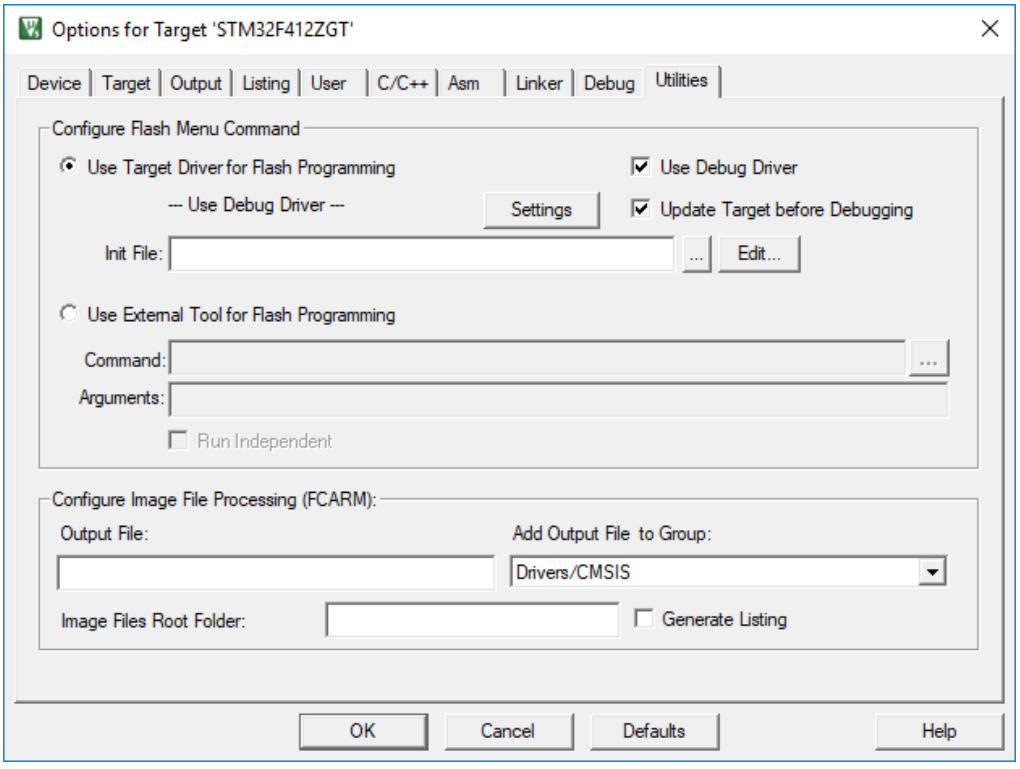

Abbildung 11-7: Einstellungen um den kompilierten Code in den Mikrocontroller zu übertragen

Für die Übertragung des kompilierten Programmcodes sind noch zusätzliche spezifische Parameter festzulegen. Diese Parameter können über den Button „Settings“ in *Abbildung 11-7* erreicht werden.

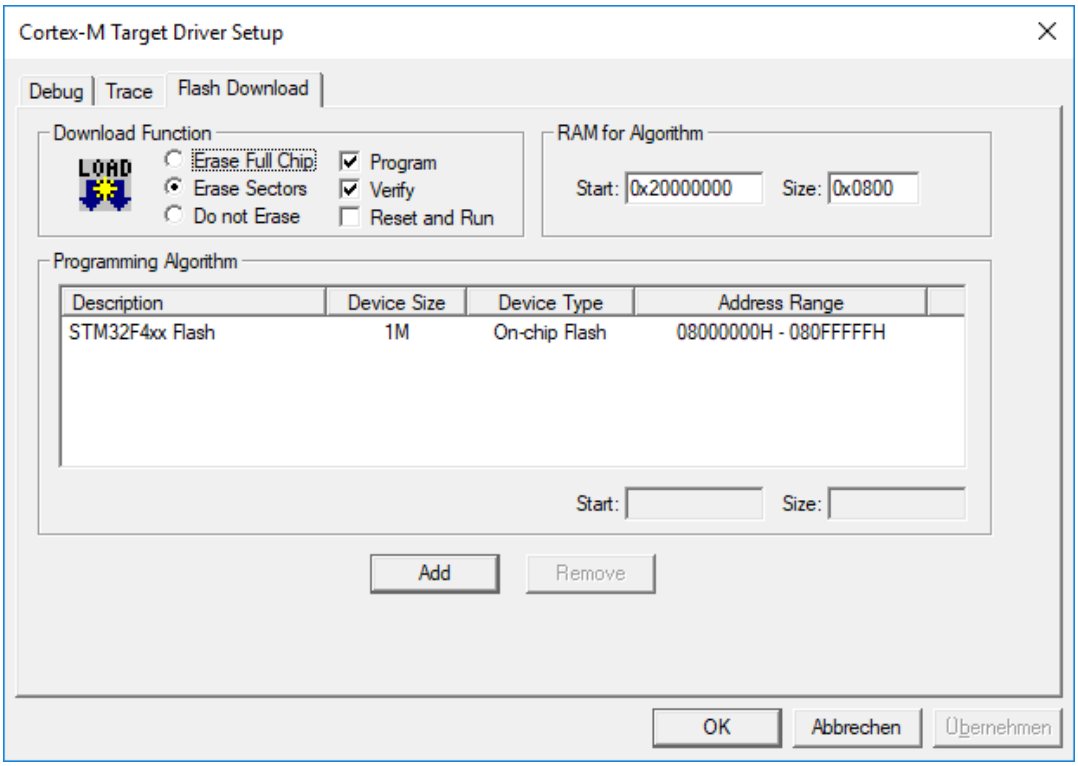

Abbildung 11-8: Spezifische Parameter für den „Flash Download“

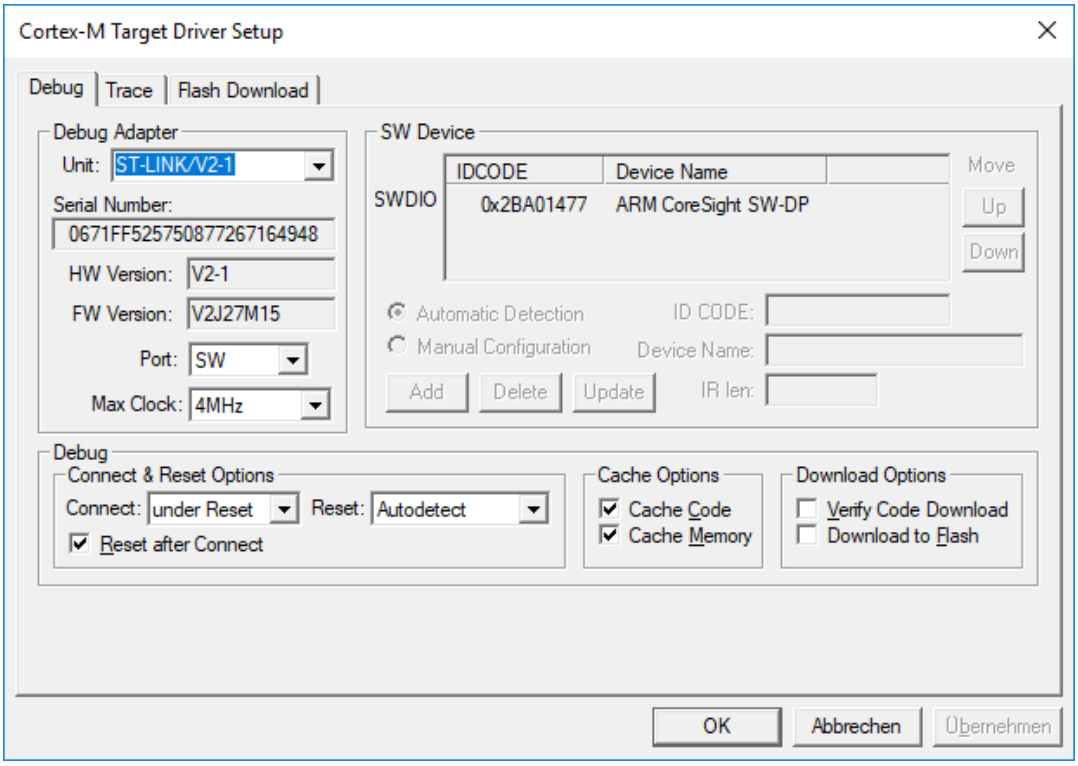

Abbildung 11-9: Zusätzliche Einstellungen für den Debugger

Wenn die Konfiguration der diversen Einstellungsparameter abgeschlossen ist, müssen noch die benötigten „C“ – Dateien in das Projekt importiert werden. Zuerst ist es notwendig eine übersichtliche Ordnerstruktur zu erstellen. Durch einen Rechtsklick auf das „Target“, in diesem Fall der „STM32F412ZGT“, kann der Punkt „Manage Project Items…“ ausgewählt werden. Anschließend werden die Ordner „Groups“ angelegt. Die einzelnen Dateien können entweder direkt hier oder später in der Projektansicht über einen Rechtsklick eingefügt werden.

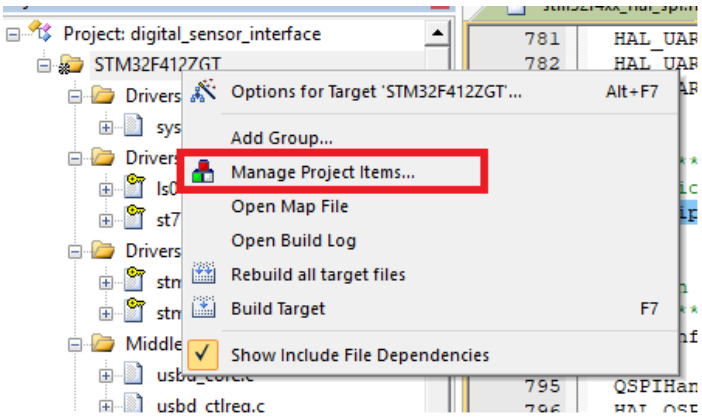

Abbildung 11-10: Auswahl von „Manage Project Items“

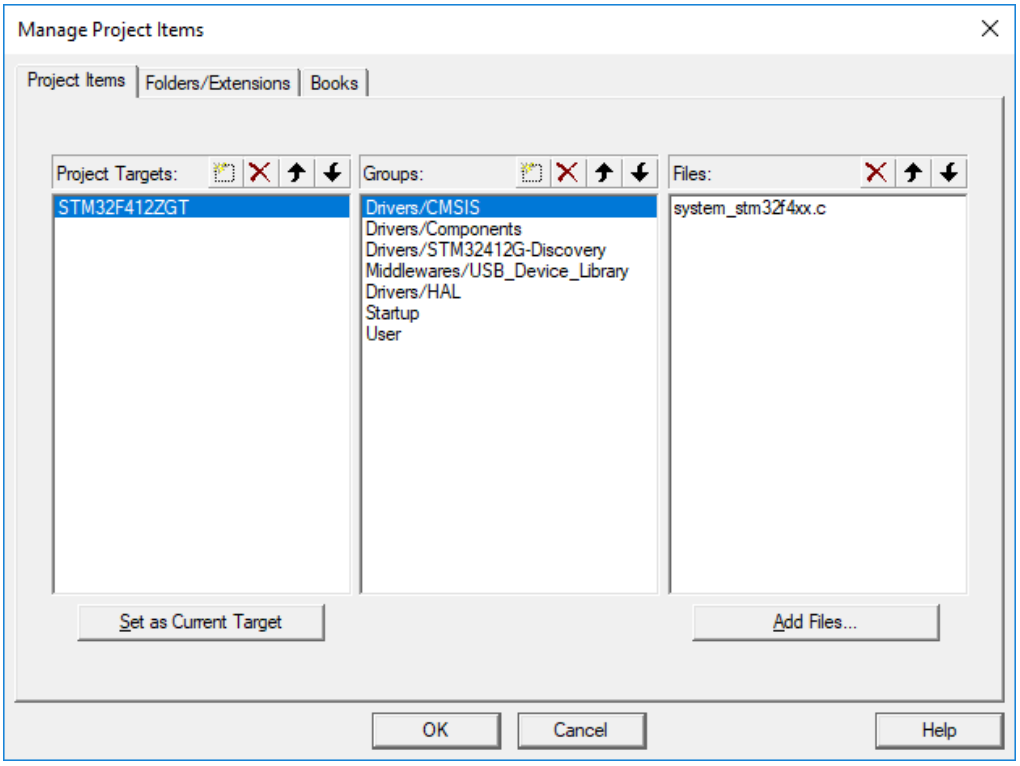

Abbildung 11-11: Erstellen der Ordner und Import von vorhandenen Dateien

Das Softwareprojekt wurde in der „µVision“ - Software in mehrere Ordner und Dateien gegliedert. Alle für den Benutzer relevanten Programmdateien befinden sich im Ordner „User“, alle anderen Dateien (HAL-Treiber, Discovery-Board-Treiber, Display-Treiber, etc.) wurden in entsprechende „Driver“ Ordner platziert. Für alle grundlegenden Schnittstellen aus dem Blockdiagramm *Abbildung 3-1* wurden im Ordner „User“ separate Dateien angelegt, um die Übersicht der einzelnen Einheiten zu bewahren.

Alle Dateien im µVision-Projektordner „User“ in *Abbildung 11-12* sind unter Windows in den µVision-Ordnern „Src“ bzw. „Inc“ zu finden. Alle restlichen Dateien können dem Windows µVision-Ordner „Drivers“ entnommen werden.

Die einzelnen vorgefertigten Systemdateien „Driver“ können den „Firmware-Examples“ entnommen werden, siehe Kapitel 4.1.

- HAL-Driver:
  - **STM32Cube_FW_F4_V1.17.0\Drivers\STM32F4xx_HAL_Driver**
- Discovery-Driver:
  - **STM32Cube_FW_F4_V1.17.0\Drivers\BSP\STM32412G-Discovery**
- Utilities (Fonts):
  - **STM32Cube_FW_F4_V1.17.0\Utilities\Fonts**
- USB-Driver (CDC-Class):
  - **STM32Cube_FW_F4_V1.17.0\Middlewares\ST\STM32_USB_Device_Library**
- CMSIS-Driver (Systemdateien):
  - **STM32Cube_FW_F4_V1.17.0\Drivers\CMSIS\Include**
- Mikrocontroller spezifische Driver:
  - **STM32Cube_FW_F4_V1.17.0\Drivers\CMSIS\Device\ST\STM32F4xx**
- LCD-Driver:
  - **STM32Cube_FW_F4_V1.17.0\Drivers\BSP\Components**

Im Main-Programm „main.c“ werden zu Beginn alle notwendigen Parameter definiert und initialisiert bzw. mit verschiedenen Voreinstellungen belegt. Im Anschluss erfolgt die Konfiguration der einzelnen Hardware-Schnittstellen, im Wesentlichen werden hier die Parameter für Timer, SPI, CAN, QSPI (NOR-Flash), USB und USART festgelegt. Die Konfiguration der Mikrocontroller-Pins, Clocks und Interrupts erfolgt in der Datei „stm32f4xx_hal_msp.c“,

welche automatisch durch die „HAL-Treiber“ aufgerufen wird. Die Interrupts werden in der Datei „stm32fxx_it.c“ bedient.

Für die einzelnen Schnittstellen stellen die „Firmware-Examples“ viele Beispiele zur Verfügung. Diese Beispiele sind dabei bereits für den jeweiligen Mikrocontroller optimiert und über ein Beispielprojekt für µVision sofort lauffähig. Diese Beispiele sind im Ordner „STM32Cube_FW_F4_V1.17.0\Projects\STM32F412G-Discovery\Examples“ zu finden. In den einzelnen Beispiel-Ordnern gibt es jeweils den Ordner „MDK-ARM“ mit den zugehörigen Projekt-Dateien. Die Software muss nur noch mit µVision kompiliert werden und kann anschließend sofort am Mikrocontroller getestet werden. Allerdings ist nicht für jede mögliche Mikrocontroller-Schnittstelle ein Beispiel verfügbar. In der Regel existieren nur Beispiele die am Discovery-Board aktiv genutzt werden können. Es gibt aber die Möglichkeit unter „STM32Cube_FW_F4_V1.17.0\Projects“ in alternativen Discovery-Boards nach entsprechenden Beispielen zu suchen. Durch die Abstraktion der HAL-Treiber müssen in der Regel nur noch die GPIO-Pins angepasst werden.

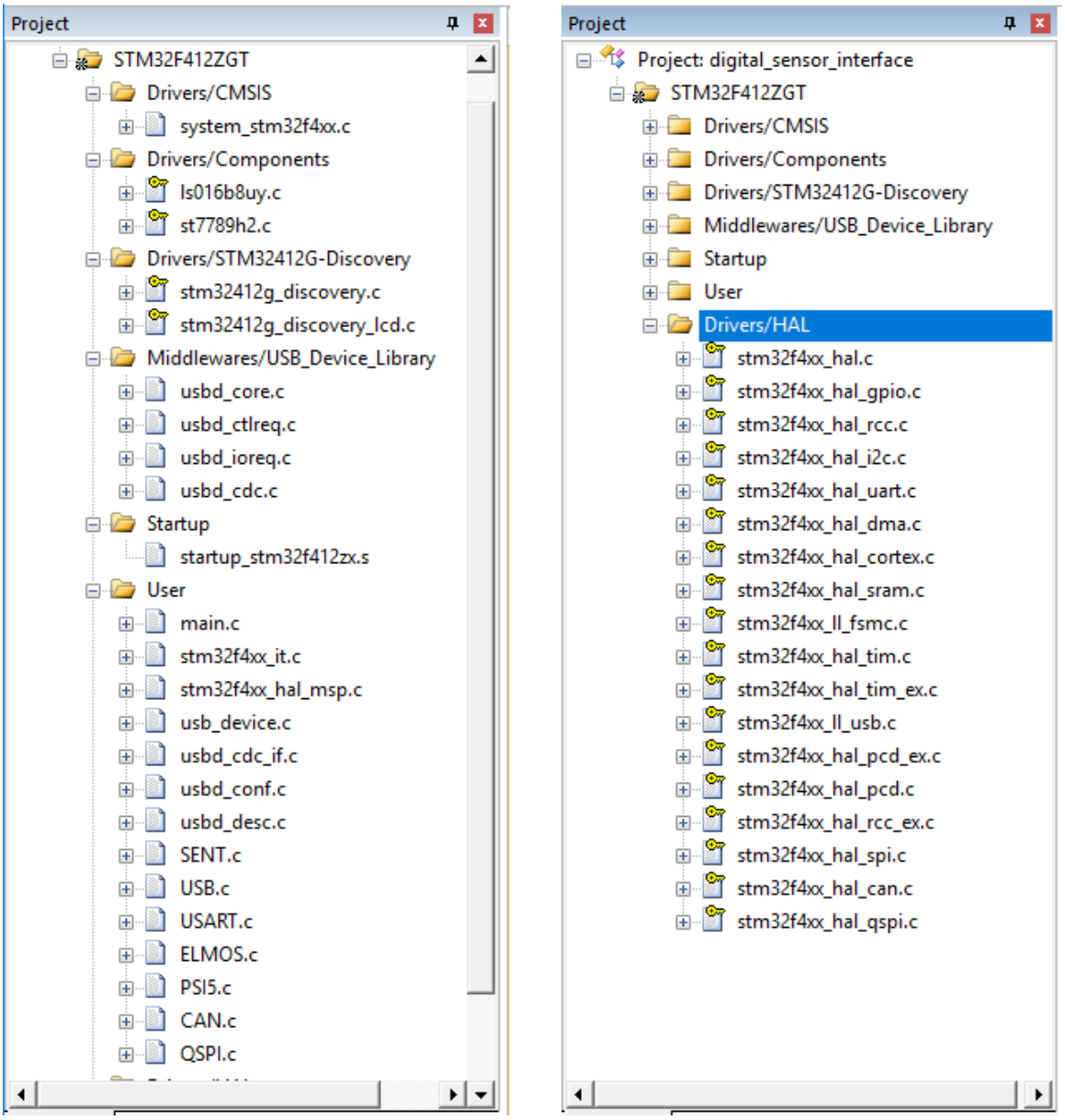

Abbildung 11-12: Alle für das Projekt notwendigen Dateien

11.2. EAGLE - Bauteilliste

Part	Value	Device	Package
µC	32F412GDISCOVERY	32F412GDISCOVERY	32F412GDISCOVERY
C1	1µ0	C_SMD_CERAMIC0805	C_0805
C2	100nF	C_SMD_CERAMIC0805	C_0805
C3	100pF	C_SMD_CERAMIC0805	C_0805
C4	100pF	C_SMD_CERAMIC0805	C_0805
C5	10µF	C_SMD_CERAMIC1206	C_1206
C6	1µ0	C_SMD_CERAMIC0805	C_0805
C7	1µ0	C_SMD_CERAMIC0805	C_0805
C8	1µ0	C_SMD_CERAMIC0805	C_0805
C9	1µ0	C_SMD_CERAMIC0805	C_0805
C10	10µ/10V	C_SMD_CERAMIC0805	C_0805
C11	100nF	C_SMD_CERAMIC0805	C_0805
C12	10µ/10V	C_SMD_CERAMIC0805	C_0805
C13	33pF	C_SMD_CERAMIC0805	C_0805
C14	100nF	C_SMD_CERAMIC0805	C_0805
C15	33pF	C_SMD_CERAMIC0805	C_0805
C16	33pF	C_SMD_CERAMIC0805	C_0805
C17	100nF	C_SMD_CERAMIC0805	C_0805
C18	10µF/50V	C_SMD_CERAMIC1206	C_1206
C19	100nF	C_SMD_CERAMIC0805	C_0805
C20	4µ7	C_SMD_CERAMIC0805	C_0805
C21	10µF/50V	C_SMD_CERAMIC1206	C_1206
C22	100nF	C_SMD_CERAMIC0805	C_0805
C23	220µF/10V	C_SMD_ELECTROLYTICSMD_6,3	D6,3X4,5
C24	1µ0	C_SMD_CERAMIC0805	C_0805
C25	1µ0	C_SMD_CERAMIC0805	C_0805
C26	100nF	C_SMD_CERAMIC0805	C_0805
C27	1µF	C_SMD_CERAMIC0805	C_0805
C28	33pF	C_SMD_CERAMIC0805	C_0805
C29	100nF	C_SMD_CERAMIC0805	C_0805
C30	330nF/25V	C_SMD_CERAMIC0805	C_0805
C31	100nF	C_SMD_CERAMIC0805	C_0805
C32	100pF	C_SMD_CERAMIC0805	C_0805
C33	100nF	C_SMD_CERAMIC0805	C_0805

C34	100nF	C_SMD_CERAMIC0805	C_0805
C35	33pF	C_SMD_CERAMIC0805	C_0805
C36	1µF	C_SMD_CERAMIC0805	C_0805
C37	1µF	C_SMD_CERAMIC0805	C_0805
C38	10µF	C_SMD_CERAMIC0805	C_0805
C39	100nF	C_SMD_CERAMIC0805	C_0805
C40	10µF/50V	C_SMD_CERAMIC1206	C_1206
C41	100pF	C_SMD_CERAMIC0805	C_0805
C42	100nF	C_SMD_CERAMIC0805	C_0805
C43	100nF	C_SMD_CERAMIC0805	C_0805
C44	10nF	C_SMD_CERAMIC0805	C_0805
C45	2.2nF	C_SMD_CERAMIC0805	C_0805
C46	10nF	C_SMD_CERAMIC0805	C_0805
C47	2.2nF	C_SMD_CERAMIC0805	C_0805
C48	220nF	C_SMD_CERAMIC0805	C_0805
C49	220nF	C_SMD_CERAMIC0805	C_0805
C50	10µF/50V	C_SMD_CERAMIC1206	C_1206
C51	220nF	C_SMD_CERAMIC0805	C_0805
C52	1µF/16V	C_SMD_CERAMIC0805	C_0805
C53	100µF/25V	C_SMD_ELECTROLYTICSMD_8	D8X6,5
C54	220µF/10V	C_SMD_ELECTROLYTICSMD_6,3	D6,3X4,5
C55	4µ7/25V	C_SMD_CERAMIC0805	C_0805
C56	100nF	C_SMD_CERAMIC0805	C_0805
C57	1µF	C_SMD_CERAMIC1210	C_1210
C58	100nF	C_SMD_CERAMIC0805	C_0805
C59	100pF	C_SMD_CERAMIC0805	C_0805
C60	100pF	C_SMD_CERAMIC0805	C_0805
C61	10nF	C_SMD_CERAMIC0805	C_0805
D1	B140	DIODE_SCHOTTKY_SMDSMA	DO-214AC
D2	BZS55C	BZS55C	1206
D3	NUP2105L	NUP2105L	SOT23
D4	NUP2105L	NUP2105L	SOT23
D5	B140	DIODE_SCHOTTKY_SMDSMA	DO-214AC
D6	SK24A	SB140	DO-214AC
D10	NUP2105L	NUP2105L	SOT23
D11	1N4007	1N4007	SUB-SMA
D12	BZS55C	BZS55C	1206

D13	BZS55C	BZS55C	1206
JP1		JP2Q	JP2Q
JP2		JP1Q	JP1
JP3		JP2Q	JP2Q
JP4		JP2Q	JP2Q
JP5		JP2Q	JP2Q
JP6	TERM	JP1Q	JP1
JP7	TERM	JP1Q	JP1
JP9		JP2Q	JP2Q
JP10		JP2Q	JP2Q
JP11		JP2Q	JP2Q
L1	WE_744227	B82790-C0	B82790-C
L2	Talema CMM101-2	B82790-C0	B82790-C
L3	47µH	SMFS7040	SMFS7040
L4	47uH	SMFS7040	SMFS7040
L6	Talema CMJ2102	B82790-C0	B82790-C
R1	10R	R_SMD0805	R_0805
R2	1K0	R_SMD0805	R_0805
R3	4K7	R_SMD0805	R_0805
R4	33R	R_SMD0805	R_0805
R5	33R	R_SMD0805	R_0805
R6	33R	R_SMD0805	R_0805
R7	1K0	R_SMD0805	R_0805
R8	1K0	R_SMD0805	R_0805
R9	1K0	R_SMD0805	R_0805
R10	150R	R_SMD0805	R_0805
R11	10R	R_SMD0805	R_0805
R12	56K	R_SMD0805	R_0805
R13	10K	R_SMD0805	R_0805
R14	150K	R_SMD0805	R_0805
R15	15K	R_SMD0805	R_0805
R16	1K0	R_SMD0805	R_0805
R17	1K0	R_SMD0805	R_0805
R18	100R	R_SMD0805	R_0805
R19	100R	R_SMD0805	R_0805
R20	1K0	R_SMD0805	R_0805
R21	10K	R_SMD0805	R_0805

R22	10K	R_SMD0805	R_0805
R23	10K	R_SMD0805	R_0805
R24	10K	R_SMD0805	R_0805
R25	18K	R_SMD0805	R_0805
R26	18K	R_SMD0805	R_0805
R27	1K0	R_SMD0805	R_0805
R28	1K0	R_SMD0805	R_0805
R29	15K	R_SMD0805	R_0805
R30	150R	R_SMD0805	R_0805
R31	1K0	R_SMD0805	R_0805
R32	100R	R_SMD0805	R_0805
R33	3K3	R_SMD0805	R_0805
R34	10R	R_SMD0805	R_0805
R35	1K0	R_SMD0805	R_0805
R40	3R3	R_SMDMINIMELF	MINIMELF
R41	0R	R_SMD0805	R_0805
R42		R_SMD0805	R_0805
R43	10R	R_SMD0805	R_0805
R44	33R	R_SMD0805	R_0805
R46	10K	R_SMD0805	R_0805
R47	120K	R_SMD0805	R_0805
R48	100R	R_SMD0805	R_0805
R50	100R	R_SMD0805	R_0805
R51	1K0	R_SMD0805	R_0805
R52	1K0	R_SMD0805	R_0805
R53	10R	R_SMD0805	R_0805
R55	10R	R_SMD0805	R_0805
R56		POT_THT01A	POT_V_01A
R57		R_SMD0805	R_0805
R58	10R	R_SMD0805	R_0805
R59	10R	R_SMD0805	R_0805
R60	820K	R_SMD0805	R_0805
R61	470R	R_SMD0805	R_0805
R62	470R	R_SMD0805	R_0805
R63	1K0	R_SMD0805	R_0805
R64	100K	R_SMD0805	R_0805
R65	10K	R_SMD0805	R_0805

R66	10K	R_SMD0805	R_0805
R67	10R	R_SMD0805	R_0805
R68	1K0	R_SMD0805	R_0805
R70	3R3	R_SMDMINIMELF	MINIMELF
R71		POT_THT01A	POT_V_01A
R72	10K	R_SMD0805	R_0805
R73		R_SMD0805	R_0805
R74	240K	R_SMD0805	R_0805
R75	1K	R_SMD0805	R_0805
R76	10K	R_SMD0805	R_0805
R77	10K	R_SMD0805	R_0805
R78	10K	R_SMD0805	R_0805
R79	10K	R_SMD0805	R_0805
R80	10K	R_SMD0805	R_0805
R81	10K	R_SMD0805	R_0805
R82		R_SMD0805	R_0805
R83	10K	R_SMD0805	R_0805
R84	10K	R_SMD0805	R_0805
R85	1K0	R_SMD0805	R_0805
R86	120K	R_SMD0805	R_0805
R87	240K	R_SMD0805	R_0805
R88	18K	R_SMD0805	R_0805
R89	10K	R_SMD0805	R_0805
R90	120K	R_SMD0805	R_0805
R91	1K0	R_SMD0805	R_0805
R96	820K	R_SMD0805	R_0805
R97	10K	R_SMD0805	R_0805
R98	10R	R_SMD0805	R_0805
R99	47R	R_SMD0805	R_0805
T1	BSS138	BSS138	SOT23
T2	BSS138	BSS138	SOT23
T4	BSP295	BSP295	SOT223
T5	BC846B	BC846B	SOT23
T6	BSS84P	BSS84P	SOT23@1
T7	BCP56-10	BCP56-10	SOT223
T8	BC846B	BC846B	SOT23
T9	BSS138	BSS138	SOT23

T10	BC846B	BC846B	SOT23
T11	BCP56-10	BCP56-10	SOT223
T12	BC556B	BC556B	TO92
T13	BC556B	BC556B	TO92
T14	BC846B	BC846B	SOT23
T15	BC846B	BC846B	SOT23
U0	E521.40_1	E521.40_1	QFN20
U1	TJA1050T	TJA1050T	SO08
U2	74HC00D	74HC00D	SO-14
U3	LMR16020PDDAR	LMR16020PDDAR	SO_08_POWER_SMD
U4	OPA2340UA	OPA2340UA	SO_08
U5	LM2903	LT1013S8	SO_08
U6	TJA1050T	TJA1050T	SO08
U7	ADM3232EARUZ	ADM3232EARUZ	TSSOP16
U8	AD5620	AD5620	MSOP08
U9	AD5620	AD5620	MSOP08
U10	LMC7211	LMC7211	SOT23-5
U11	LMC7101	LMC7101	SOT23-5
U12	LMC7101	LMC7101	SOT23-5
U13	LMV331	LMV331	SC70-5L
U14	LMV331	LMV331	SC70-5L
U15	LM2675M-ADJ	LM2675M-ADJ	SO_08
X1	STL1550WV2	STL1550WV2	1550WV2
X2	STL1550WH4	STL1550WH4	1550WH4
X3	RS232	F09HP	F09HP
X4	STL1550WH2	STL1550WH2	1550WH2
X5	CAN	M09HP	M09HP
X6	STL1550WH2	STL1550WH2	1550WH2
X7	CAN	M09HP	M09HP
X8		NYLON_02V	NYLON_02
X9		NYLON_02V	NYLON_02
X10		NYLON_02V	NYLON_02
X11		NYLON_02V	NYLON_02

Tabelle 11-1: Exportierte EAGLE - Bauteilliste

11.3. EAGLE - Schaltpläne

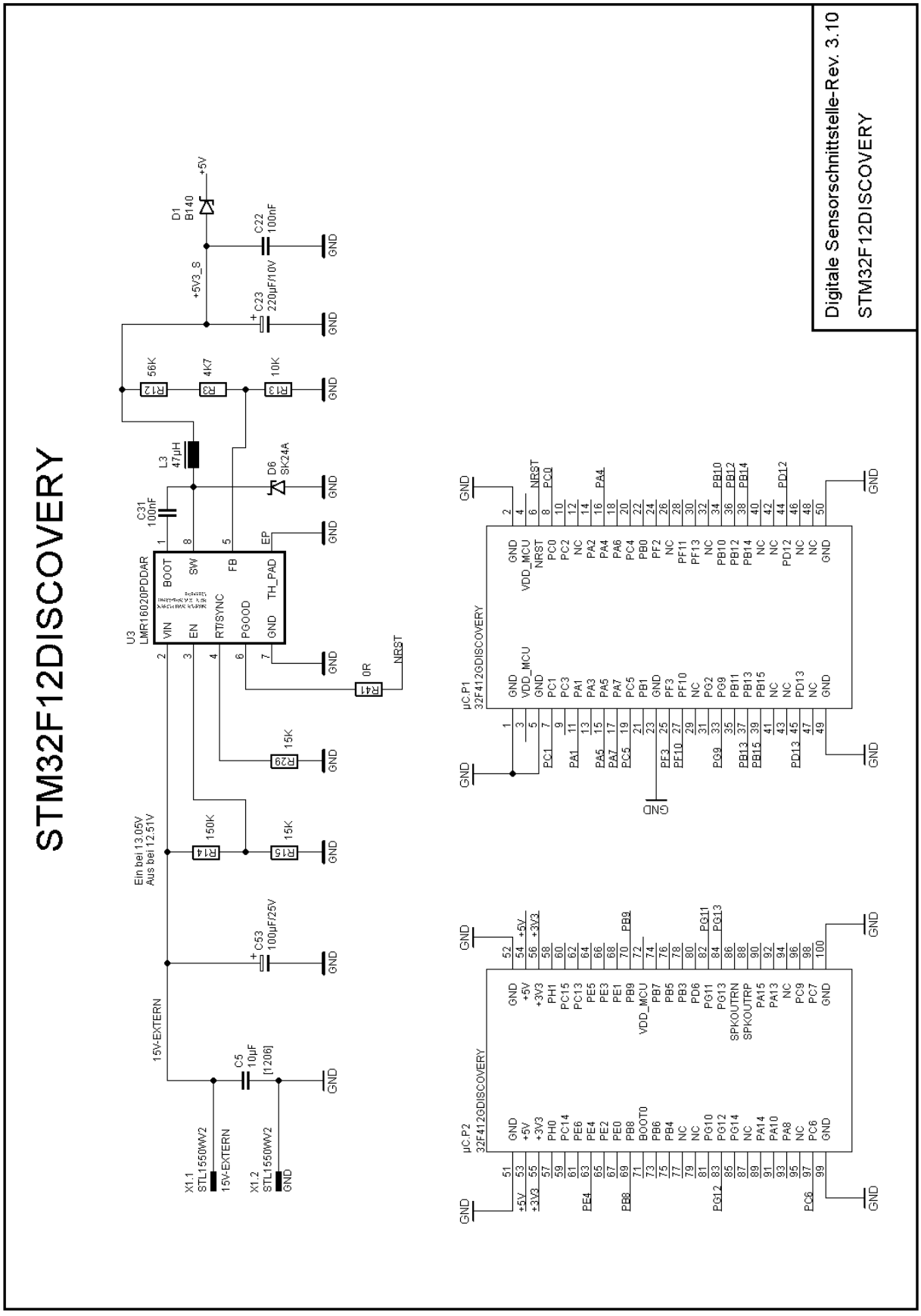

Abbildung 11-13: Schaltplan Discovery-Board und Spannungsversorgung

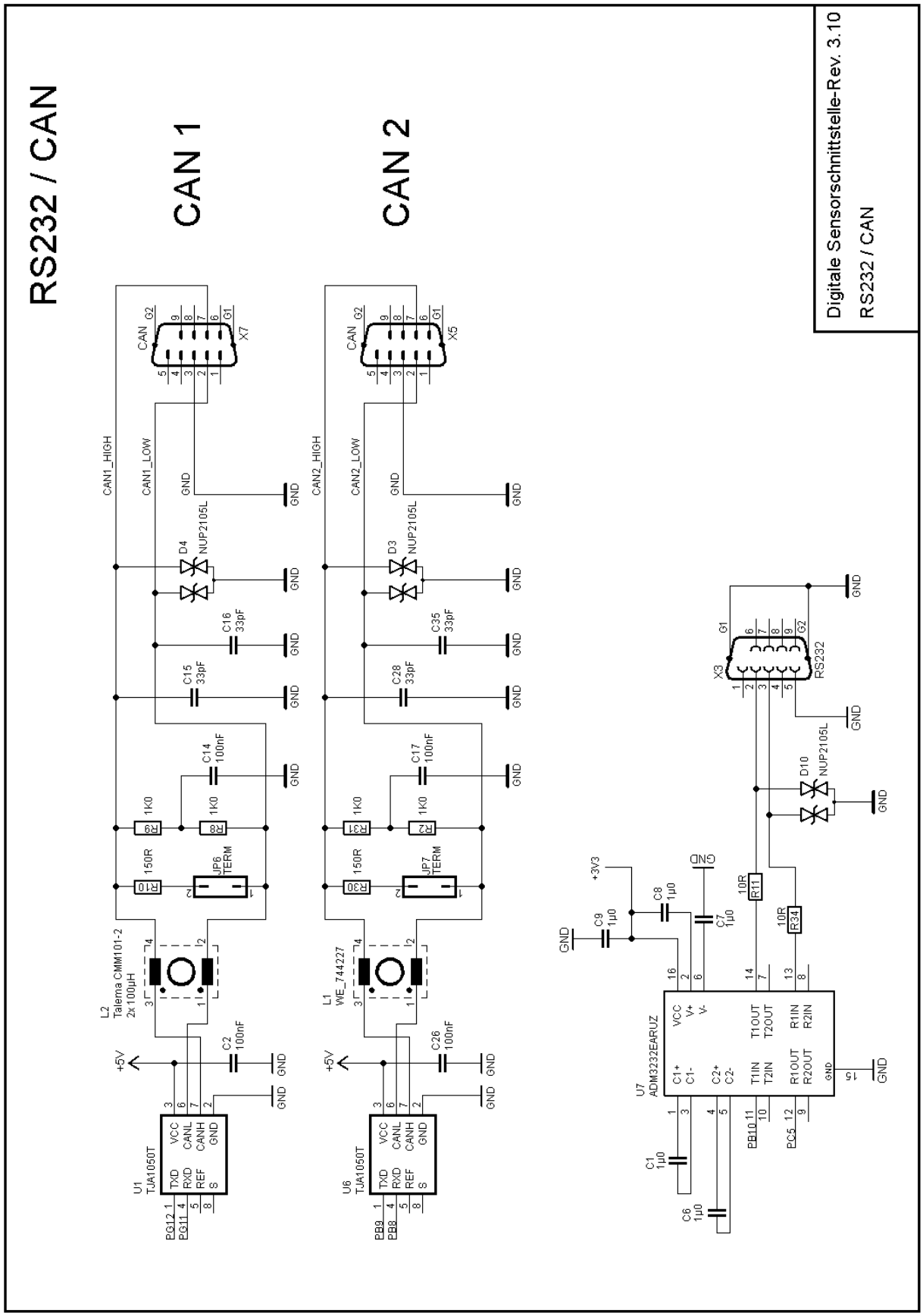

Abbildung 11-14: RS232 / CAN - Schnittstellen

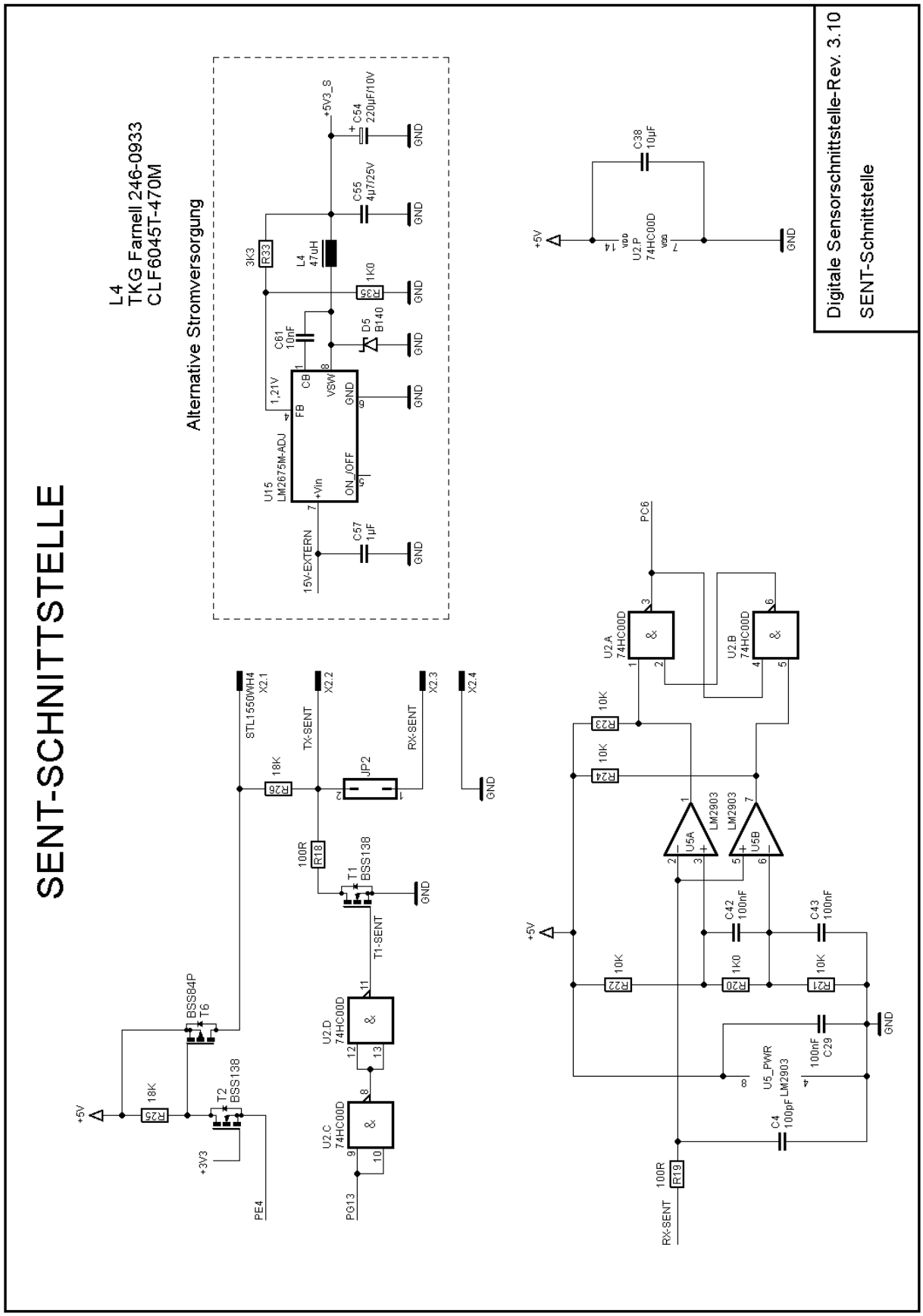

Abbildung 11-15: SENT Schnittstelle für den Empfang und die Aussendung von SENT-Nachrichten

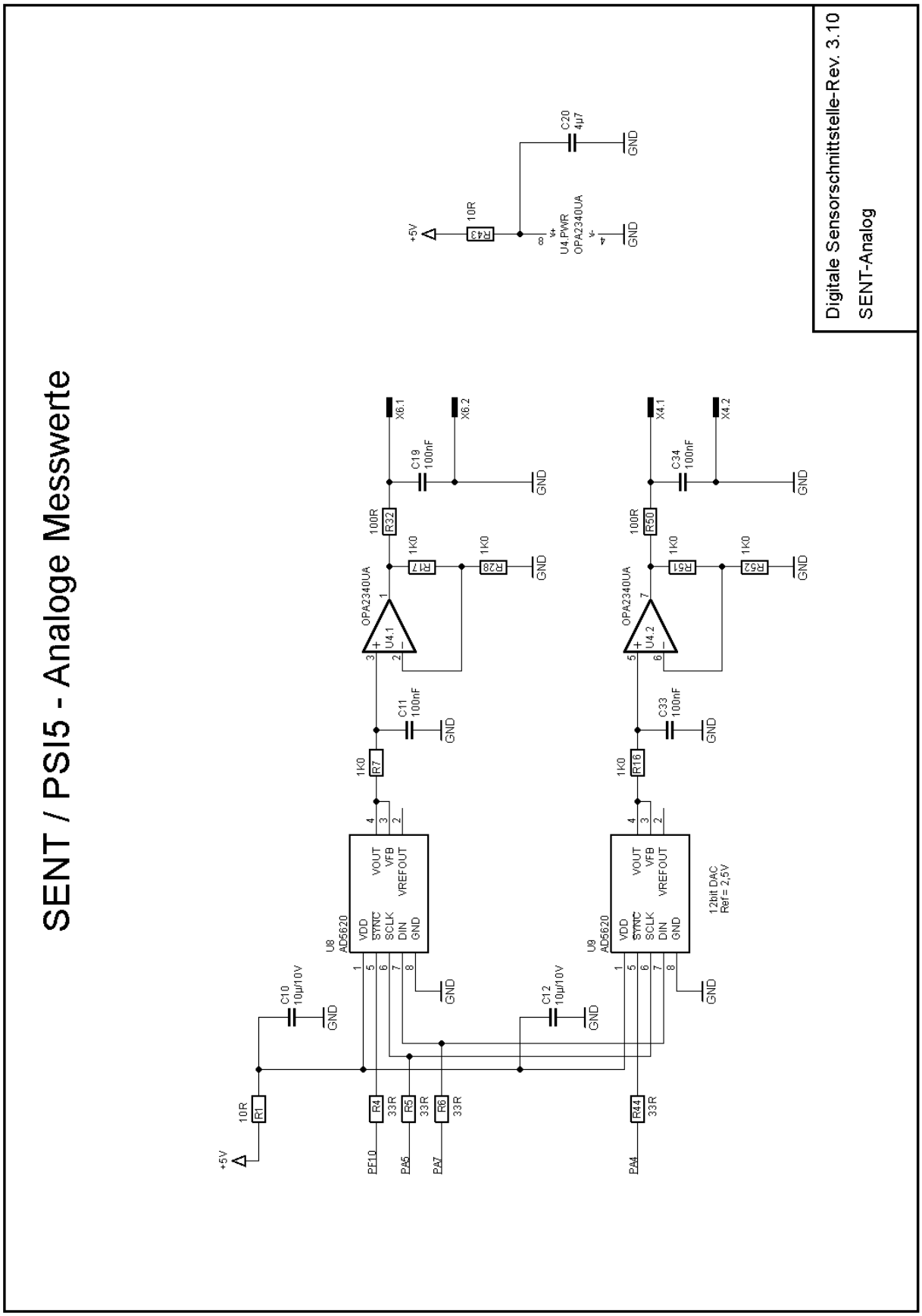

Abbildung 11-16: Analoge Ausgabe der SENT/PSI Ausgabe mit Hilfe des AD5620 DAC

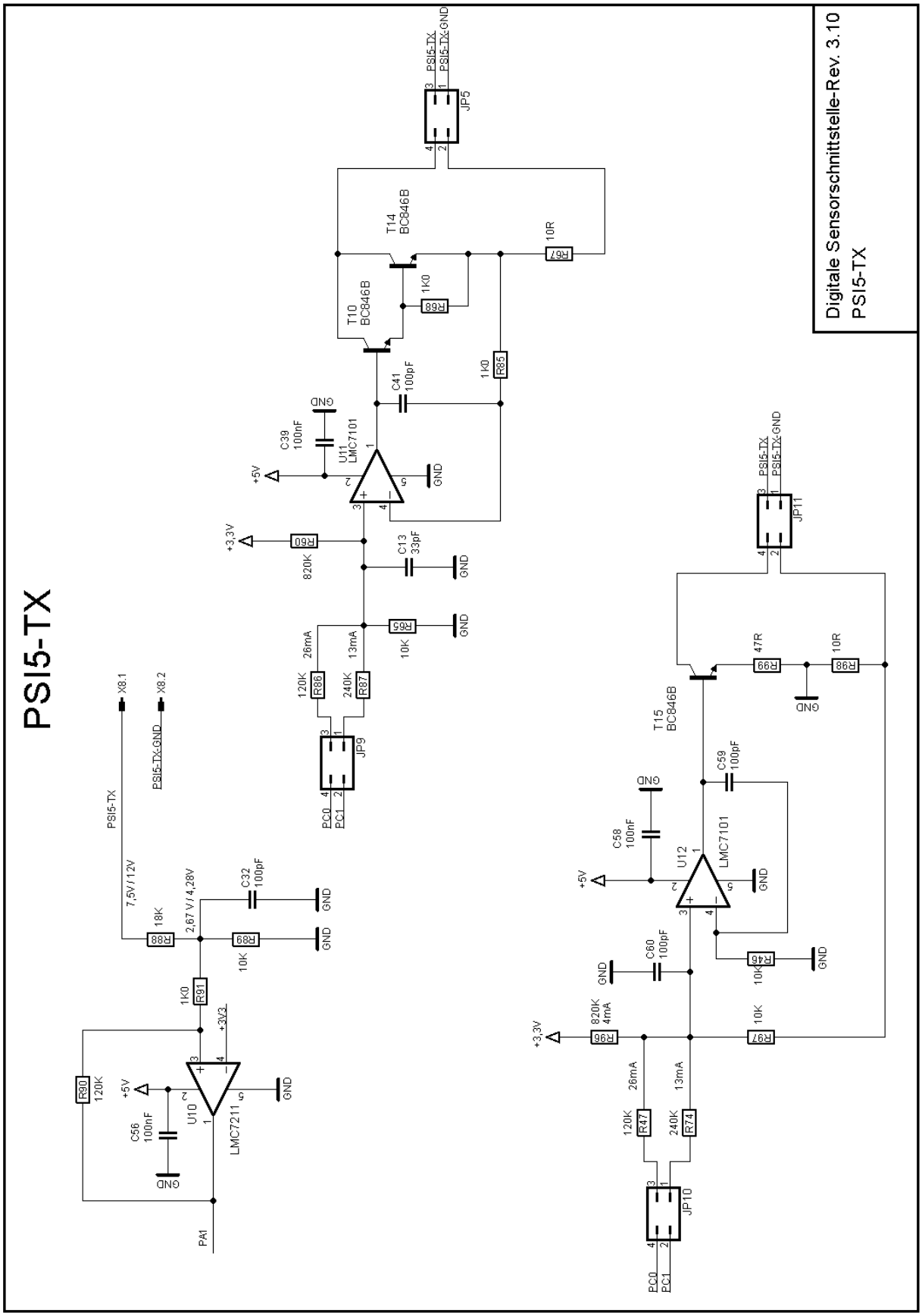

Abbildung 11-17: PSI5 – Aussendung von PSI5-Nachrichten

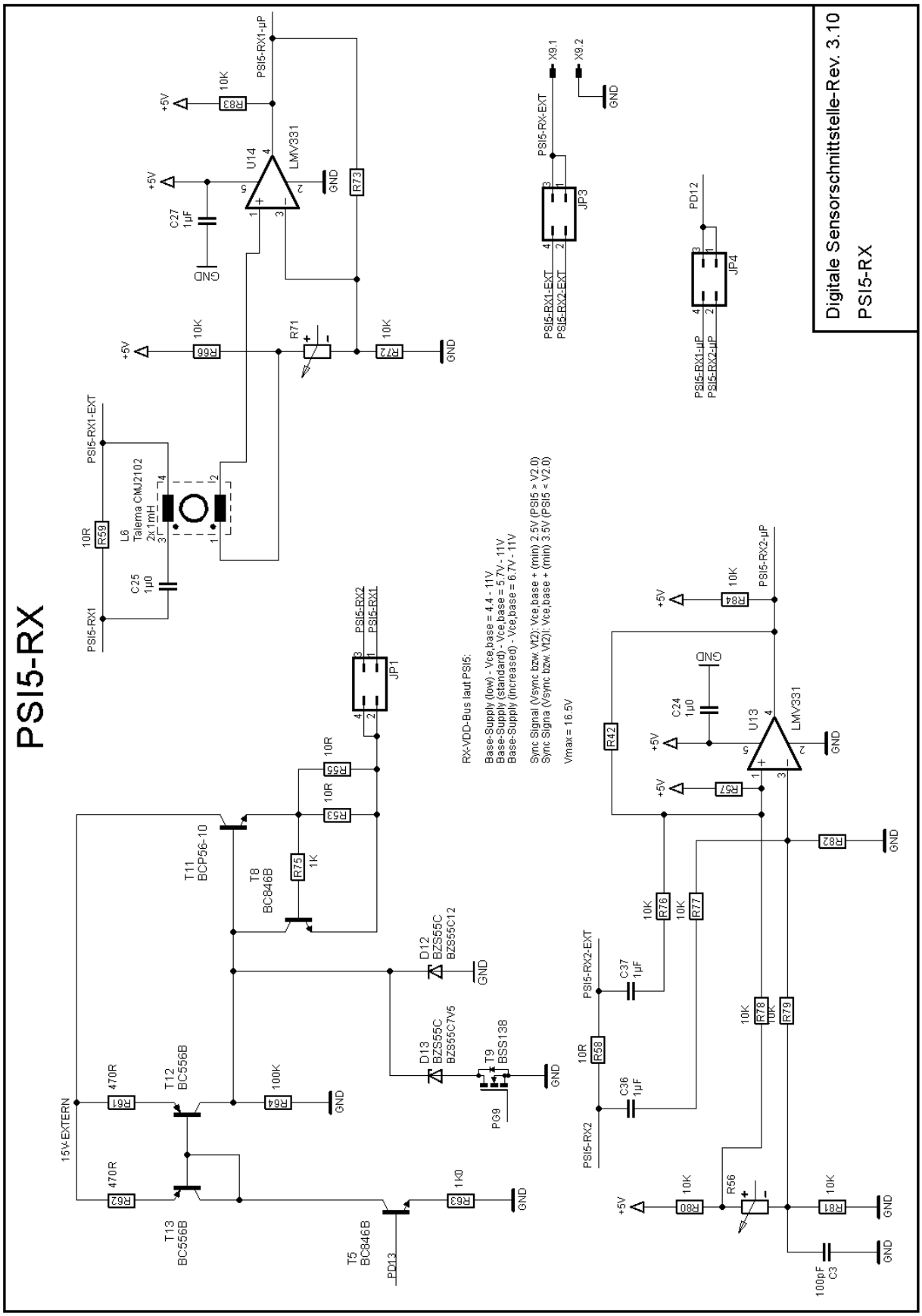

Abbildung 11-18: PSI5 – Empfangsschnittstelle

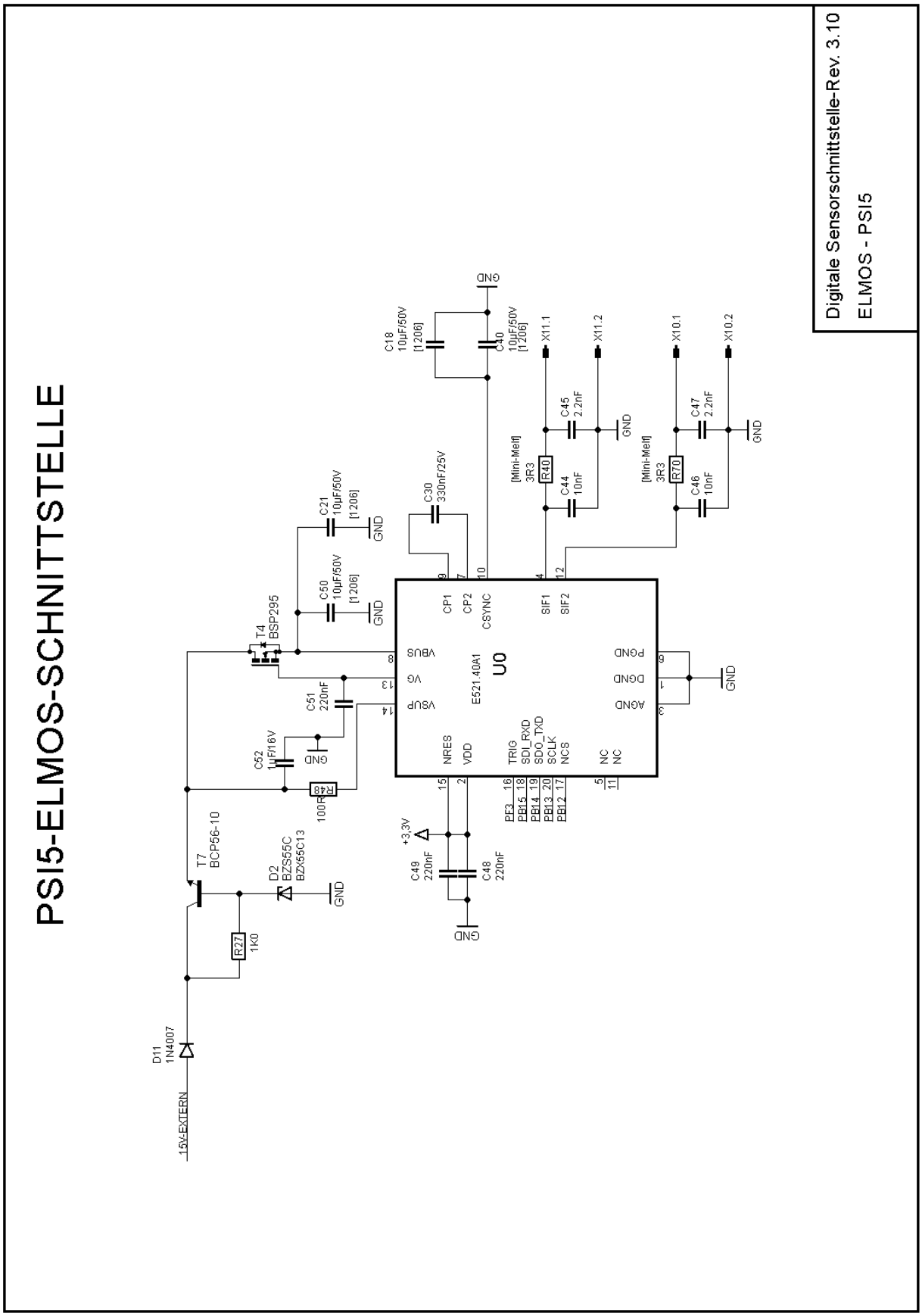

Abbildung 11-19: ELMOS „E521.40A1“ Schnittstelle