FACHBEREICH MATHEMATIK UND
NATURWISSENSCHAFTEN
BERGISCHE UNIVERSITÄT WUPPERTAL

Entwicklung eines strahlenharten Controllers für das Kontrollsystem des ATLAS-Pixeldetektors am HL-LHC

Dissertation
von
Jennifer Boek

November 2012

Für Gitte

Inhaltsverzeichnis

Kapitel 1

Einleitung

Der weltweit größte Protonenbeschleuniger, der Large Hadron Collider (= LHC) am CERN, ist seit 2009 in Betrieb. Bei Schwerpunktsenergien von bis zu $\sqrt{s} = 8\,\text{TeV}$ nehmen die Experimente Daten auf. Ein grundlegender Erfolg ist die Entdeckung eines neuen Bosons durch die Experimente ATLAS und CMS.

Das Upgrade des LHC zum High Luminosity Large Hadron Collider (= HL-LHC) gilt der Suche nach neuer Physik bis zu sehr hohen Teilchen-Massen. Aufgrund der erhöhten Luminosität können die physikalischen Prozesse mit einer besseren Statistik vermessen werden. Davon profitieren auch die physikalischen Studien des neu entdeckten Bosons, das noch eindeutig als das Higgs-Boson identifiziert werden muss.

Die Luminositätserhöhung zum HL-LHC stellt die Detektorentwicklung vor eine Herausforderung, da sich die zu erwartenden Teilchenraten stark vergrößern werden. Eine Konsequenz daraus ist der komplette Neubau des ATLAS-Spurdetektors, speziell des ATLAS-Pixeldetektors, der in Kapitel 2 ausführlicher beschrieben wird.

Da der ATLAS-Pixeldetektor komplett neu gebaut wird, werden viele grundlegende Komponenten verändert. Aus diesem Grund muss ein neues Detektorkontrollsystem (= DCS) für den zukünftigen Pixeldetektor aufgebaut werden. Die Anforderungen und der Aufbau des zukünftigen DCS werden in Kapitel 3 vorgestellt. Aus dem Konzept für das zukünftige Detektorkontrollsystem geht hervor, dass Chipdesign für die DCS-Komponenten nötig wird, um eine sichere, strahlenharte und möglichst Material reduzierte DCS-Kommunikation zum Detektor aufzubauen.

Im Rahmen dieser Arbeit wurde ein digitales Chipdesign erstellt, das eine Controller Area Network Kommunikation (= CAN-Kommunikation) beinhaltet. Diese Komponente wird DCS-Controller genannt und gewährleistet zusammen mit dem DCS-Chip die sichere DCS-Datenübertragung für jeden Betriebsmodus des Pixeldetektors.

Für ein besseres Verständnis der zu erwarteten Strahlenschädigung der DCS-Komponenten im 130 nm Prozess, werden in Kapitel 4 sowohl Einzelfehlereffekte, als auch Dosiseffekte diskutiert.

In Kapitel 5 wird ein geeignetes Datenprotokoll für die DCS-Kommunikation bestimmt, das später in dem DCS-Controller implementiert wird.

Neben der Entwicklung des DCS-Controllers (Kapitel 6) und der Verifikation der Funktionsfähigkeit (in Kapitel 7) wurden Messungen mit einem Protonenstrahl am Paul Scherrer Institut in der Schweiz durchgeführt, die bestätigen, dass die fehlerfreie Datenübertragung auch in der erwarteten Strahlungsumgebung für den HL-LHC gegeben ist (vgl. Kapitel 8 und 9).

Kapitel 2

Die Luminositätssteigerung des LHC

Die Luminosität ist das Maß für die Wahrscheinlichkeit von Protonzusammenstößen in den kollidierenden Strahlen eines Ringbeschleunigers. Sie trifft eine Aussage über die Leistungsfähigkeit des Beschleunigers. Die Luminosität für Teilchenbeschleuniger ist [1]:

$$L = \frac{n \cdot N_1 \cdot N_2 \cdot f}{A} \tag{2.1}$$

N_1 und N_2 stehen für die Anzahl der Teilchen in den kollidierenden Strahlen mit der Anzahl von n Paketen. Der Parameter f ist die Kollisionsfrequenz und A die effektive Querschnittsfläche der Teilchenpakete im Wechselwirkungspunkt.

Die integrierte Luminosität $\mathcal{L}$ gibt die über einen bestimmten Zeitraum ΔT addierte Luminosität an:

$$\mathcal{L} = \int_{\Delta T} L \cdot dt \tag{2.2}$$

Die Einheit der integrierten Luminosität ist die invertierte Fläche und wird in inversen barn (b) angegeben. Dabei entspricht $1\,\mathrm{b}^{-1} = 10^{24}\,\mathrm{cm}^{-2} = 10^{28}\,\mathrm{m}^{-2}$.

Die Anzahl der produzierten Ereignisse N_x eines bestimmten Prozesses x mit dem Wirkungsquerschnitt σ_x ist definiert als:

$$N_x = \sigma_x \cdot \mathcal{L} \tag{2.3}$$

Ziel der LHC-Experimente ist neben dem Nachweis des Higgs-Bosons die Suche nach neuen, im Standardmodell nicht enthaltenen Teilchen. Diese Teilchen haben Massen in der Größenordnung O(1 TeV), da bisherige Suchen den Bereich kleinerer Teilchenmassen bereits weitgehend ausschließen [2].

Für die Produktion dieser Teilchen werden sehr kleine Wirkungsquerschnitte erwartet. Zum einen fällt der Wirkungsquerschnitt bereits aus Dimensionsgründen proportional zu $\frac{1}{m^2}$. Weiterhin muss der Impulsbruchteil x der an der Produktion der Teilchen beteiligten Partonen in den Protonen vergleichsweise groß sein, damit Teilchen hoher Masse produziert werden können. Wegen des starken Abfalls der Strukturfunktion mit steigendem x [1], führt dies nochmal zu einer sehr starken Unterdrückung der Produktionswahrscheinlichkeit. Neue Teilchen können also nur dann entdeckt werden, wenn der Beschleuniger eine sehr hohe Luminosität hat.

Für den Large Hadron Collider, der in Kapitel 2.4 noch ausführlicher beschrieben ist, wird eine maximale Luminosität von $L = 10^{34}\,cm^{-2}s^{-1}$ angestrebt. Dabei soll über die gesamte Laufzeit des LHC eine integrierte Luminosität von $\mathcal{L} = 300\,fb^{-1}$ erreicht werden. Mit dem Luminositätsupgrade wird am zukünftigen HL-LHC dann eine fünffache Luminosität und eine zehnfache integrierte Luminosität angestrebt.

Eine hohe Luminosität ist nicht nur für die Entdeckung neuer Teilchen wichtig, sondern führt auch zur Erhöhung der Statistik und ermöglicht somit eine präzisere Vermessung schon bekannter Prozesse. Ein Beispiel hierfür ist die Vermessung des neu entdeckten Bosons, auf die aus aktuellem Anlass im Weiteren noch genauer eingegangen wird.

2.1 Das Standardmodell der Elementarteilchenphysik

Im Folgenden wird unter Verwendung der Bücher [1] und [3] das Standardmodell der Teilchenphysik beschrieben und aus diesem Grund auf die Angabe von Originalzitaten verzichtet, da diese in den Büchern zu finden sind.

Die uns bekannte stabile Materie lässt sich allein aus den Quarks und dem geladenen Lepton der ersten Familie, dem up- und down-Quark und dem Elektron aufbauen. Tatsächlich gibt es aber noch zwei weitere Familien von Teilchen mit größerer Masse. Diese werden in hochenergetischen Prozessen erzeugt, sie sind jedoch meist kurzlebig und zerfallen in die Teilchen der ersten Familie.

Eine Tabelle der elementaren Teilchen mit dessen Eigenschaften, wie Spin, Ladung und Masse ist in Abbildung 2.1 zu finden. Zu jedem Teilchen gibt es ein Antiteilchen, das die gleiche Masse und den gleichen Spin hat, allerdings umgekehrte Ladung und Parität. Neben den Fermionen mit Spin 1/2 sind in Abbildung 2.1 auch die Boso-

nen, die Spin 1 Austauschteilchen der elektromagnetischen, schwachen und starken Wechselwirkung, angegeben.

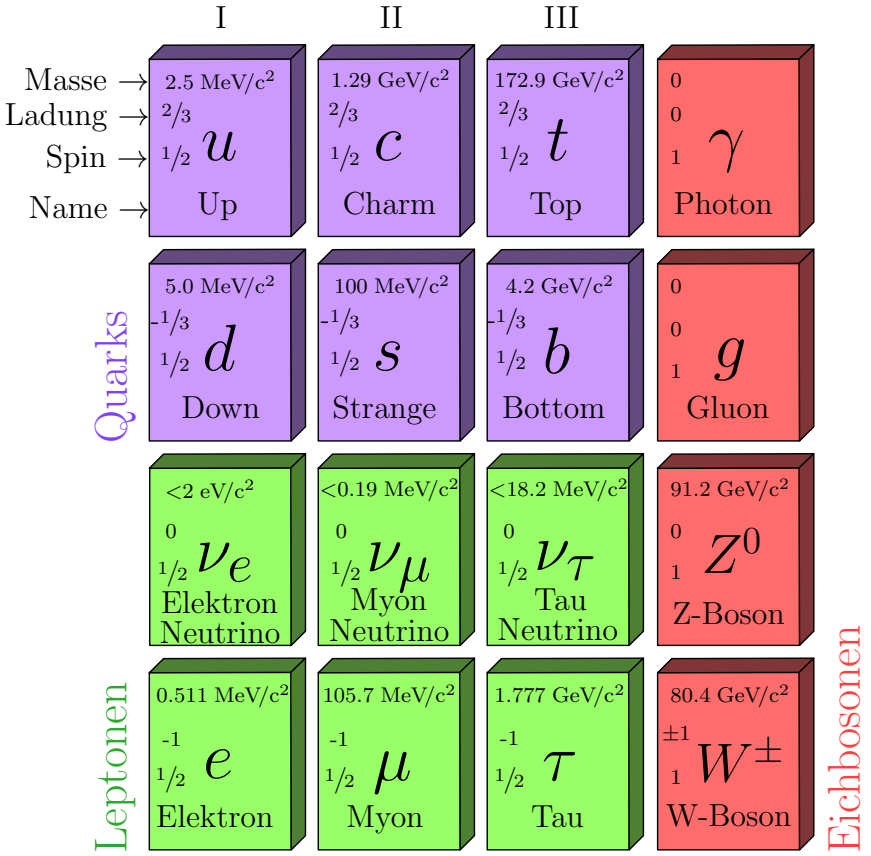


Abbildung 2.1: Die elementaren Teilchen des Standardmodells mit ihren Eigenschaften aufgeteilt in die 3 Fermionen-Familien (Quarks und Leptonen) und die Eichbosonen

Die Gravitation ist die bekannteste, aber zugleich schwächste Kraft, die erst zwischen sehr massiven Objekten spürbar wird. Das postulierte Austauschteilchen der Gravitation, das Graviton, ist noch nicht entdeckt worden und in Abbildung 2.1 nicht enthalten. Für die Experimente am LHC ist die Gravitation nicht relevant und wird deshalb hier nicht behandelt. Die elektromagnetische Kraft wirkt auf die elektrische Ladung — gleich geladene Objekte stoßen sich ab, entgegengesetzte Ladungen ziehen sich an. Die schwache Kraft ist für den β-Zerfall schwerer elementarer Teilchen verantwortlich, wie z.B. den Zerfall des Neutrons. Die starke Kraft bindet die Quarks im Hadron und gewährleistet, dass die Protonen im Atomkern sich aufgrund ihrer gleichen elektrischen Ladung nicht abstoßen. Stabile Kerne existieren, da die starke Kraft stärker als die elektromagnetische Kraft ist.

Im Standardmodell werden die drei Kräfte (starke, schwache und elektromagnetische Kraft) als Wechselwirkungen identifiziert, auf die im Folgenden genauer eingegangen wird.

Die elektromagnetische Wechselwirkung

Die elektromagnetische Wechselwirkung wird durch die Eichtheorie der Quantenelektrodynamik (QED) beschrieben. Alle elektrisch geladenen Teilchen können elektromagnetisch wechselwirken. Das Austauschteilchen ist das Photon γ, das an die elektrische Ladung koppelt. Photonen sind masselos und tragen selbst keine Ladung.

Die schwache Wechselwirkung

Die schwache Wechselwirkung wirkt auf alle Fermionen. Die Austauschteilchen der schwachen Wechselwirkung sind das $W^{\pm}$- und Z-Boson. Im Gegensatz zu den Austauschbosonen der anderen Wechselwirkungen sind W- und Z-Boson massebehaftet [3]:

$$M_W = 80,403 \pm 0,029 \,\frac{\text{GeV}}{\text{cm}^2} \qquad M_Z = 91,188 \pm 0,002 \,\frac{\text{GeV}}{\text{cm}^2} \tag{2.4}$$

Aufgrund ihrer Masse ist die Lebensdauer der Eichbosonen gering und ihre Reichweite kurz. Deshalb erscheint diese Wechselwirkung schwach.

Der bekannteste Prozess der schwachen Wechselwirkung ist der Beta-Zerfall, bei dem ein Elektron aus dem Atomkern emittiert wird, bzw. ein Neutron in ein Proton und ein Elektron unter Emission eines Elektron-Antineutrinos zerfällt: $n \rightarrow e^- + \overline{\nu_e} + p$. Die elektrisch- und farbneutralen Neutrinos sind die einzigen Teilchen, die ausschließlich schwach wechselwirken.

Die starke Wechselwirkung

Die Theorie der starken Wechselwirkung, die Quantenchromodynamik (QCD), basiert auf der Symmetriegruppe SU(3). Die starke Wechselwirkung koppelt an die Farbladung. Die drei Farben werden mit rot, grün und blau bezeichnet. Da Quarks eine Farbladung tragen, wirkt die starke Kraft auf Quarks bzw. Antiquarks.

Aus Quarks aufgebaute Teilchen werden Hadronen genannt. Da Hadronen farbneutral sind, gibt es zwei Arten: die Mesonen ($q\overline{q}$) und die Baryonen (qqq). Baryonen bestehen aus drei Quarks, die jeweils eine unterschiedliche Farbladung tragen. Mesonen bestehen aus einem Quark-Antiquark-Paar, wobei das Antiquark die Antifarbe des Quarks trägt.

Die Austauschteilchen der QCD sind die Gluonen g, die an die Farbladungen koppeln. Jedes Gluon trägt eine Farbe und eine Antifarbe. Es ergeben sich insgesamt 8 Vektorgluonen, die die Farbkraft zwischen den Quarks übertragen können. Aufgrund der Farbladung können Gluonen auch an sich selbst koppeln.

Die elektroschwache Vereinheitlichung

Die elektromagnetische und die schwache Wechselwirkung können durch eine einzige Theorie, die elektroschwache Wechselwirkung, beschrieben werden. Diese wurde von Glashow, Salam und Weinberg entwickelt und nennt sich auch GSW-Theorie.

Eine experimentelle Bestätigung der GSW-Theorie ergab sich durch den Nachweis des neutralen Stroms, der Entdeckung der W- und Z-Bosonen mit der erwarteten Masse und insbesondere den Präzisionsmessungen zur elektroschwachen Theorie bei LEP [4].

Die GSW-Theorie verwendet den Prozess der spontanen Symmetriebrechnung, welcher die Eichbosonen mit einer Masse versieht. Das wird durch das Einführen eines Isospindubletts erreicht. Der Higgs-Mechanismus sagt ein skalares, neutrales Teilchen, das Higgs-Boson, voraus.

Die fundamentalen Vektorbosonen sind das masselose Isospin-Triplett W_μ^1, W_μ^2 und W_μ^3 (aus der SU(2)-Gruppe) und das masselose Isospin-Singulett B_μ (aus der U(1)-Gruppe). Infolge der spontanen Symmetriebrechung erhalten 3 Bosonen (W_μ^+, W_μ^- und Z^0) eine Masse und ein Boson (A_μ entspricht dem γ) bleibt masselos. Diese vier Bosonen sind eine Linearkombinationen aus W_μ und B_μ [1]:

$$W_\mu^\pm = \frac{1}{\sqrt{2}} \left(W_\mu^{(1)} \pm i W_\mu^{(2)} \right) \tag{2.5}$$

$$Z_\mu = W_\mu^{(3)} \cos\theta_W - B_\mu \sin\theta_W \tag{2.6}$$

$$A_\mu = W_\mu^{(3)} \sin\theta_W + B_\mu \cos\theta_W \tag{2.7}$$

Der Mischungswinkel θ_W wird Weinberg-Winkel genannt. Er kann nicht theoretisch berechnet, sondern muss experimentell bestimmt werden.

2.2 Das Higgs-Boson

Das Higgs-Boson, genauer beschrieben in [5], koppelt an die Masse der Fermionen und Bosonen und somit bevorzugt an die schweren Teilchen des Standardmodells, wie W- und Z-Boson sowie Top- und Bottom-Quark. In Abbildung 2.2 sind die Feynmangraphen der vier hauptsächlichen Produktionsmechanismen des Higgs-Bosons in Proton-Proton Wechselwirkungen dargestellt. Diese bestehen aus der Fusion zweier

Gluonen $gg \rightarrow H$, der Vektor-Boson-Fusion $qq \rightarrow V^*V^* \rightarrow qq + H$, der assoziierten Produktion mit Vektorbosonen $q\bar{q} \rightarrow V + H$ und der assoziierten Produktion mit einem $t\bar{t}$-Paar $gg \rightarrow t\bar{t} + H$.

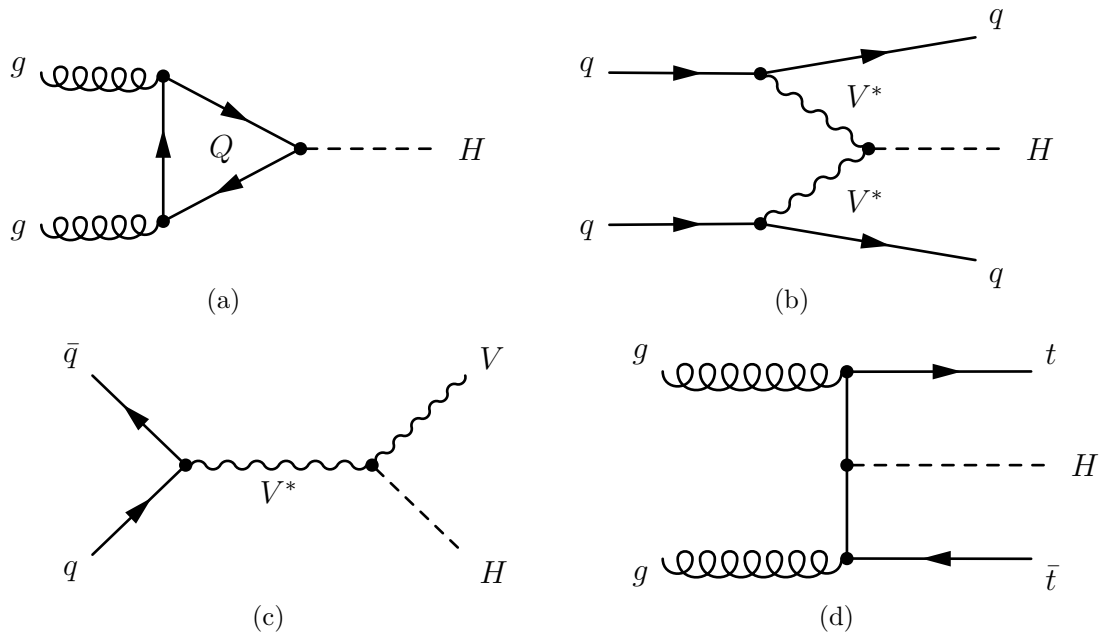


Abbildung 2.2: Vier Produktionskanäle des Higgs-Bosons veranschaulicht in Feynmangraphen. Zu sehen sind (a) die Gluon-Gluon-Fusion, (b) die Vektor-Boson-Fusion, (c) die assoziierte Produktion mit einem Vektorboson und (d) die assoziierte Produktion mit einem $t\bar{t}$-Paar.

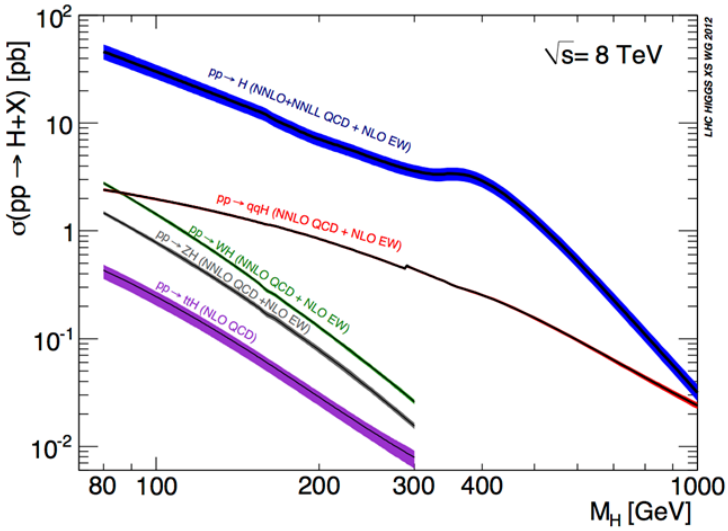


Abbildung 2.3: Produktionswirkungsquerschnitte für das Higgs-Boson bei einer Schwerpunktsenergie von $\sqrt{s}$=8 TeV als Funktion der Higgs-Masse [6]

In Abbildung 2.3 sind die Produktionswirkungsquerschnitte für das Higgs-Boson bei einer Schwerpunktsenergie von $\sqrt{s}$=8 TeV, der momentanen Schwerpunktsenergie des LHC, gezeigt. Zu sehen ist, dass die Gluon-Gluon-Fusion (pp $\rightarrow$ H) den höchsten Wirkungsquerschnitt besitzt, während die Kanäle der Vektor-Boson-Fusion (pp $\rightarrow$ qqH) etwa eine Größenordnung darunter liegen. Trotzdem ist die Vektor-Boson-Fusion für den LHC ein bedeutender Prozess, da zwei charakteristische Vorwärts-Jets erzeugt werden und keine (bzw. wenig) Kalorimeteraktivität im zentralen Bereich des Detektors erwartet wird.

Die Abbildung 2.4 zeigt den gesamten erwarteten Higgs-Massenbereich. Da das Higgs bevorzugt an schwere Teilchen koppelt, zerfällt es auch vornehmlich in diese. Für hohe Massen zerfällt das Higgs fast ausschließlich in die Eichbosonen, mit einer Verteilung von 2/3 $H \rightarrow WW$ und 1/3 $H \rightarrow ZZ$. Unterhalb der Schwelle zur Produktion eines WW-Paares ($M_H \leq 135$ GeV) dominieren die Zerfälle in ein reelles und ein virtuelles Vektorboson: $H \rightarrow WW^*$ bzw. $H \rightarrow ZZ^*$. Für den leptonischen Zerfall beider Z-Bosonen im Fall $H \rightarrow ZZ^*$ ist dieser Kanal aufgrund der 4 Leptonen im Endzustand gut vom Untergrund separierbar.

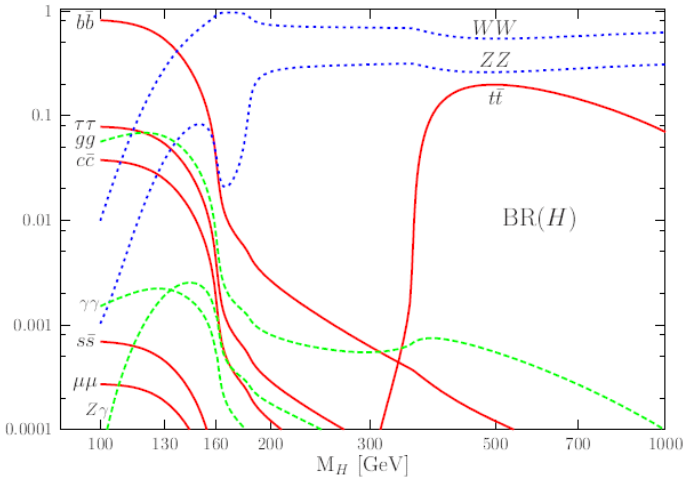


Abbildung 2.4: Die Verzweigungsverhältnisse für die Zerfallskanäle des Standardmodell-Higgs-Bosons für den gesamten erwarteten Massebereich. [5]

Im unteren Massenbereich des Higgs-Bosons bis etwa 135 GeV ist mit einem Verzweigungsverhältnis von 75 % bis 50 % der Zerfall in ein b-Quark-Antiquark-Paar $H \rightarrow b\bar{b}$ dominant (siehe dazu Abbildung 2.4). Um diesen Kanal vom hohen QCD-Untergrund des LHC selektieren zu können, werden exzellente Vertexdetektoren benötigt, die die b-Quark-Jets mit hoher Effizienz und Reinheit von „leichten“ Jets (u, d, c) identifizieren können.

Trotz seines geringen Verzweigungsverhältnisses ist der Zerfallskanal $H \rightarrow \gamma\gamma$ sehr wichtig, da er aufgrund der besonders einfachen Signatur leicht zu identifizieren ist.

Die Entdeckung eines neuen Bosons am LHC

Im Juli 2012 wurde die Entdeckung eines neuen Bosons bekannt gegeben. Die Experimente ATLAS und CMS haben mit $5{,}9\,\sigma$ bzw. $5{,}8\,\sigma$ Signifikanz ein neues Teilchen entdeckt [7, 8]:

$$m_H = 126{,}0 \pm 0{,}4\,(\text{stat}) \pm 0{,}4\,(\text{sys})\,\text{GeV} \qquad \text{(ATLAS)} \tag{2.8}$$

$$m_H = 125{,}3 \pm 0{,}4\,(\text{stat}) \pm 0{,}5\,(\text{sys})\,\text{GeV} \qquad \text{(CMS)} \tag{2.9}$$

Für das ATLAS-Experiment sind die grundlegenden Daten für die Beobachtung des Bosons die kombinierten Daten aus 2011 mit einer inversen Luminosität von $\mathcal{L} = 4{,}8\,\text{fb}^{-1}$ aufgezeichnet bei einer Schwerpunktsenergie von $\sqrt{s} = 7\,\text{TeV}$ und den Daten aus 2012 mit $\sqrt{s} = 8\,\text{TeV}$ bei $\mathcal{L} = 5{,}8\,\text{fb}^{-1}$ [7].

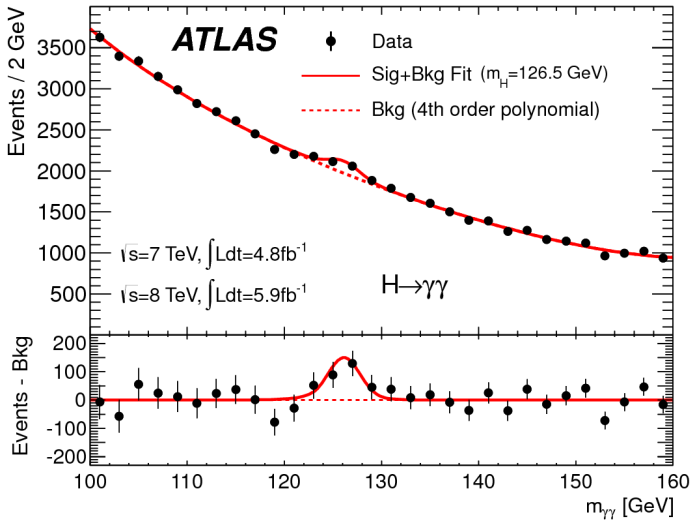


Abbildung 2.5: Invariante Massenverteilung der 2 Photon-Kandidaten für die kombinierten Daten mit $\sqrt{s} = 7\,\text{TeV}$ und $\sqrt{s} = 8\,\text{TeV}$. Das eingezeichnete Higgs-Signal entspricht einer Higgs-Masse von 126,5 GeV. Im unteren Bereich ist das verbleibende Signal nach Abzug der erwarteten Untergründe zu sehen. [7]

Abbildung 2.5 zeigt das invariante Massenspektrum für Ereignisse mit 2 Photonen. Im Massenbereich um $m_{\gamma\gamma} = 126\,\text{GeV}$ sieht man ein deutliches Signal über dem erwarteten glatten Untergrund. Eine eindeutige Entdeckung des Boson konnte durch die Kombination von drei verschiedenen Zerfallskanälen des Higgs-Bosons $H \to \gamma\gamma$, $H \to WW \to l\nu l\nu$ und $H \to ZZ \to llll$ erreicht werden.

Bei dem neuen Boson handelt es sich sehr wahrscheinlich um das Higgs-Boson. Eine direkte Bestätigung ist aber noch nicht möglich, da erst einige Eigenschaften des Higgs-Bosons bei dem entdeckten neuen Boson nachgewiesen werden konnten. Interessant ist es nun mit einer höheren Statistik weitere Zerfallskanäle des Higgs-Bosons zu erschließen und den Spin zu messen. Der Spin 1 ist bereits ausgeschlossen, da es einen Zerfall in zwei Photonen gibt. Es muss also geprüft werden, ob das Teilchen Spin 0 oder Spin 2 hat.

2.3 Physikalische Messungen am HL-LHC

Die Entdeckung und Vermessung seltener physikalischer Prozesse am LHC ist durch die bisher erreichte Luminosität begrenzt. Eine erhöhte Luminosität ist nötig, um seltene Prozesse eindeutig beobachten zu können. Der HL-LHC bietet ein breites Entdeckungspotential für Physik jenseits des Standardmodells und erlaubt es Ausschlussgrenzen im TeV-Bereich für Supersymmetrische Teilchen und andere Teilchen neuer Physik zu setzen, sowie Präzisionsmessungen des Higgs-Bosons durchzuführen. Als ein Beispiel für das Potential des HL-LHC wird im Folgenden auf die Möglichkeiten der Messungen in der Higgs-Physik eingegangen [9].

Die Messung des Higgs-Bosons am HL-LHC

Die erhöhte Luminosität ergibt eine viel bessere Statistik für bereits entdeckte Kanäle und erlaubt seltene Higgs-Boson Produktions- bzw. Zerfallskanäle genauer zu untersuchen. Es werden neben den schon zur Entdeckung des Higgs-Bosons verwendeten Zerfallskanälen $H \rightarrow \gamma\gamma$, $H \rightarrow ZZ^*$ und $H \rightarrow WW^*$ die Zerfälle des Higgs in ein Leptonpaar $H \rightarrow \tau^+\tau^-$ sowie $H \rightarrow \mu\mu$ und die assoziierte $t\bar{t}$-Higgs-Produktion mit dem Zerfall des Higgs in zwei Photonen $t\bar{t} \rightarrow H \rightarrow t\bar{t}\gamma\gamma$ genauer gemessen werden können [9].

In Abbildung 2.6 ist die erwartete Präzision der Messung der Signalstärken in den verschiedenen Zerfallskanälen für die integrierten Luminositäten des LHC von $\mathcal{L} = 300\,\mathrm{fb}^{-1}$ und des HL-LHC von $\mathcal{L} = 3000\,\mathrm{fb}^{-1}$ zu sehen. Die Zerfälle des Higgs in die Endzustände $\gamma\gamma$ und ZZ^* profitieren von der erhöhten Luminosität am stärksten. Der Zerfall $H \rightarrow b\bar{b}$ ist in der Grafik nicht enthalten, da Jet-Energieauflösung und B-Hadron Identifikationseffizienz in Jets für die hohen Luminositäten noch nicht ausreichend analysiert sind.

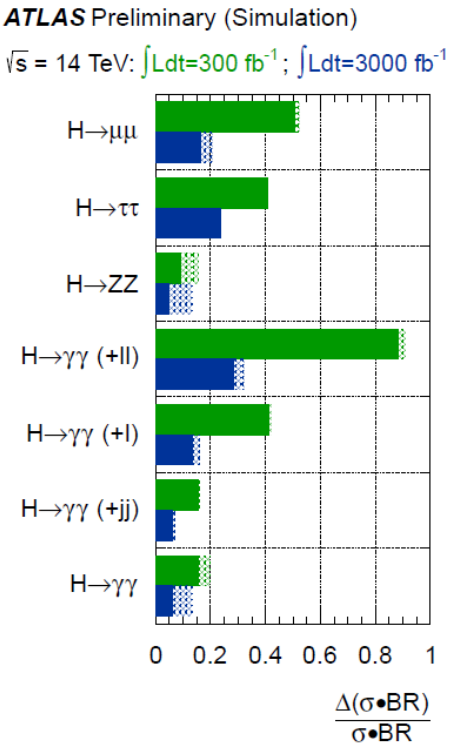


Abbildung 2.6: Erwartete Präzision der Messungen der Signalstärken für die Design Luminositäten des LHC und des HL-LHC [9]

Die Messung der Higgs-Selbstkopplung am HL-LHC

Der Test des Standardmodell-Higgs-Mechanismus ist die Messung der Higgs-Selbstkopplung am HL-LHC. Die Feynmangraphen für die Dreifach und die Vierfach-Higgs-Selbstkopplung sind in Abbildung 2.7 gezeigt.

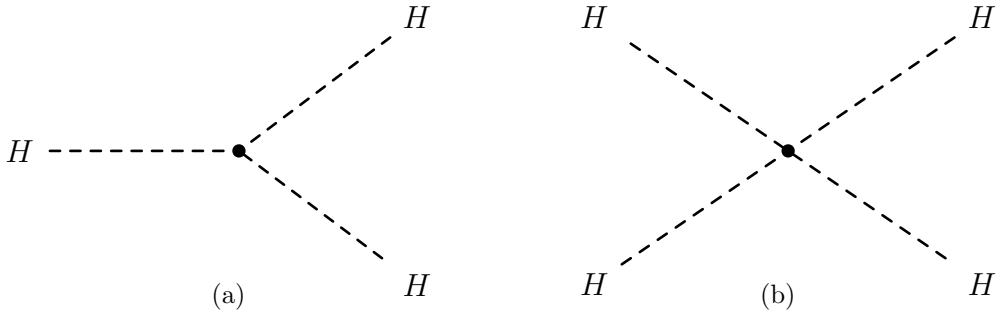


Abbildung 2.7: Die Higgs-Selbstkopplung im Standardmodell. (a) Die Dreifach-Higgs-Selbstkopplung und (b) die Vierfach-Higgs-Selbstkopplung.

Eine direkte Beobachtung der Dreifach-Higgs-Kopplung über die Detektion von Higgs-Paarproduktionen ist für die Experimente am HL-LHC möglich. Als geeignete Kandidaten für die Entdeckung der Higgs-Selbstkopplung werden die Zerfälle

der Kanäle $HH \rightarrow b\bar{b}W^+W^-$ und $HH \rightarrow b\bar{b}\gamma\gamma$ angesehen. Die Erwartung ist, mit beiden Experimenten (ATLAS und CMS) zusammen mit dem Kanal $HH \rightarrow b\bar{b}\gamma\gamma$ pro Experiment etwa $3\,\sigma$ Signifikanz zu erreichen [9].

2.4 Das Upgrade des LHC zum HL-LHC

Am europäischen Kernforschungszentrum in Genf befindet sich der weltweit größte Proton-Protonbeschleuniger, der Large Hadron Collider. An seinen vier Wechselwirkungspunkten befinden sich die Experimente ATLAS, CMS[1)], LHC-b[2)] und ALICE[3)]. Am LHC sollen in 40 MHz Protonkollisionen Schwerpunktsenergien von bis zu $\sqrt{s} = 14\,\text{TeV}$ erreicht werden. Es wird zudem eine maximale Luminosität von $L = 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$ angestrebt. Über die gesamte LHC-Betriebslaufzeit soll eine integrierte Luminosität von $\mathcal{L} = 300\,\text{fb}^{-1}$ erreicht werden [10].

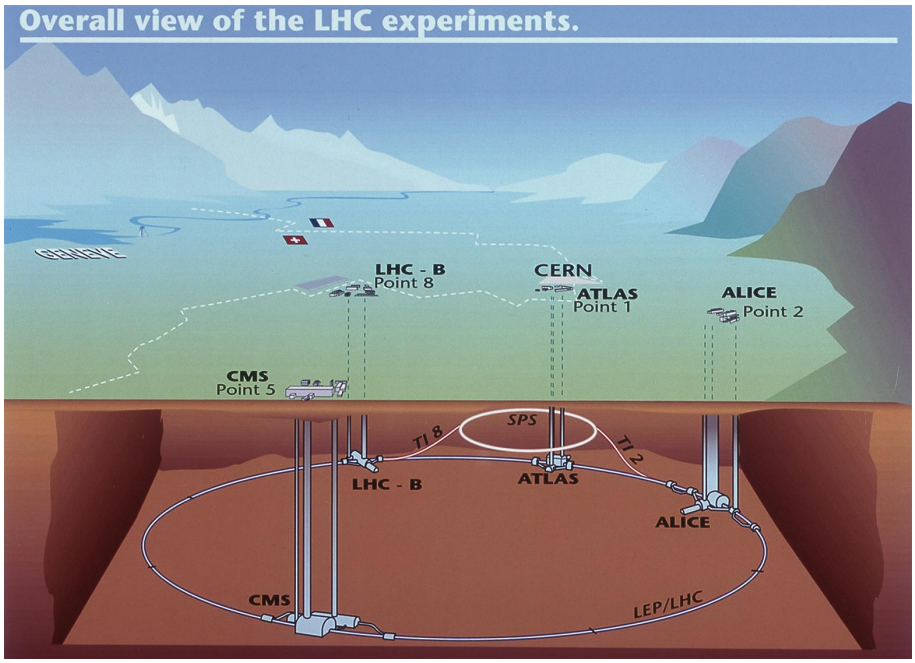


Abbildung 2.8: Der Large Hadron Collider hat einen Umfang von 27 km und befindet sich in der Nähe von Genf in etwa 100 m unter der Erde. [11]

Knapp drei Jahre nach dem Start des LHC im Jahre 2009 wurde mit den kombinierten Daten bei Schwerpunktsenergien von $\sqrt{s} = 7\,\text{TeV}$ und $\sqrt{s} = 8\,\text{TeV}$ die erste wichtige Entdeckung eines neuen Bosons gemacht.

[1)] **C**ompact **M**uon **S**olenoid
[2)] **L**arge **H**adron **C**ollider **b**eauty
[3)] **A** **L**arge **I**on **C**ollider **E**xperiment

Das Upgrade des LHC zum HL-LHC unterteilt sich in drei Phasen, die im Folgenden kurz aufgeführt sind [12]:

- **Phase 0 Upgrade:** Nach der ersten längeren Umbauphase im Jahr 2013/2014 soll der LHC die vorgesehene Schwerpunktsenergie von $\sqrt{s} = 14\,\text{TeV}$ erreichen, sowie die Design-Luminosität von $\text{L} = 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$ liefern.

- **Phase 1 Upgrade:** Nach einer angesammelten integrierten Luminosität von etwa $\mathcal{L} = 100\,\text{fb}^{-1}$ wird eine weitere Umbauphase des LHC die Verdopplung der Luminosität auf etwa $\text{L} = 2,2 \cdot 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$ zur Folge haben. Bis 2021 sollen $\mathcal{L} = 300\text{-}400\,\text{fb}^{-1}$ mit dem LHC erreicht werden.

- **Phase 2 Upgrade:** In einer dritten Phase, geplant für das Jahr 2022, wird der Umbau zum HL-LHC durchgeführt, mit dem bei gleichen Energien eine Luminosität von $\text{L} = 5 \cdot 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$ angestrebt wird. Es soll eine integrierte Luminosität von $\mathcal{L} = 3000\,\text{fb}^{-1}$ erreicht werden, davon $\mathcal{L} = 2500\,\text{fb}^{-1}$ am HL-LHC.

Die Prototypen dieser Arbeit wurden alle für das Phase 2 Upgrade des ATLAS-Detektors entwickelt.

2.5 Der ATLAS-Detektor für den HL-LHC

Um den Aufbau des zukünftigen ATLAS-Detektors für das Upgrade zum HL-LHC im Jahr 2022 besser erklären zu können, wird zunächst kurz auf den momentanen ATLAS-Detektor eingegangen.

Der ATLAS-Detektor

Der in Abbildung 2.9 gezeigte ATLAS-Detektor ist ein zylindrisch aufgebauter, um den Wechselwirkungspunkt symmetrischer Vielzweckdetektor. Er besteht aus einem inneren Spurdetektor umgeben von einem dünnen, supraleitenden Solenoid, der ein 2 Tesla Magnetfeld erzeugt, dem elektromagnetischen und dem hadronischen Kalorimeter und einem Myon-Spektrometer.

Der innere Spurdetektor dient einer sehr genauen Vertexrekonstruktion sowie der präzisen Impulsmessung geladener Teilchen. Der Pixeldetektor erreicht mit seinen ca. 80 Millionen Auslesekanälen die höchste Auflösung im gesamten ATLAS-Detektor. Die Silizium-Detektoren, der Pixel- und der Streifen-Detektor[4)] haben eine Ab-

[4)] SCT, **S**emi**c**onductor **T**racking

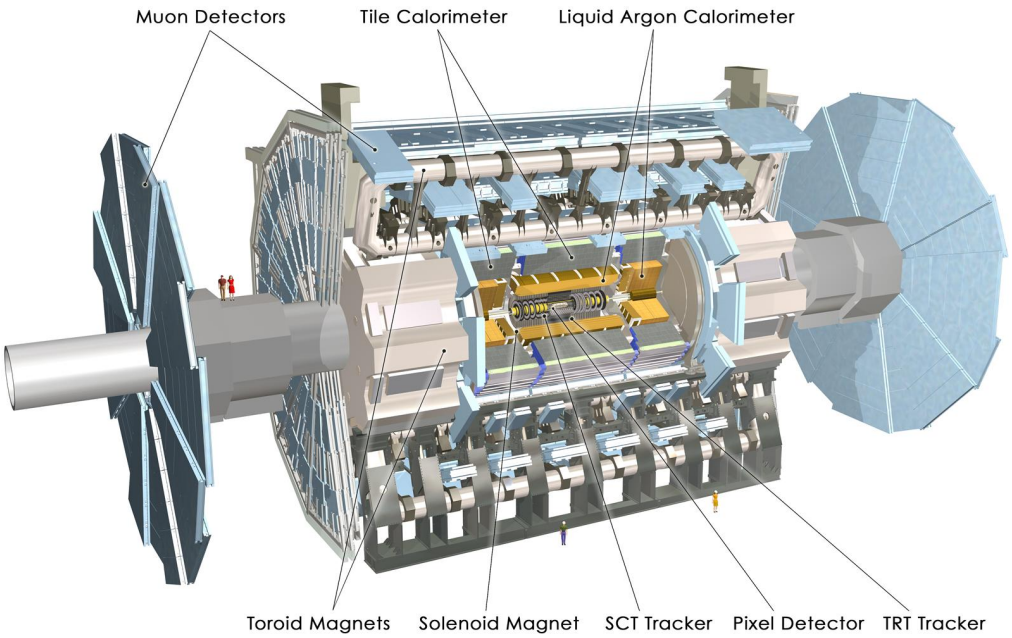


Abbildung 2.9: Der ATLAS-Detektor ist 44 m lang, hat einen Durchmesser von 22 m und ist 7000 t schwer. Es können mit dem ATLAS-Detektor Datenraten bis zu 40 MHz verarbeitet werden. [11]

deckung von $\eta \leq 2{,}5$[5)]. Auf den SCT folgt der Übergangsstrahlungsdetektor TRT[6)]. Dieser Detektor besteht aus Drahtkammer-Röhrchen mit einem Durchmesser von 4 mm, die mit Gas gefüllte Driftkammern sind. Diese liefern eine hohe Zahl an Spurpunkten, aber eine geringe Auflösung in z-Richtung. Im TRT werden geladene Teilchen nachgewiesen. Durch die Erzeugung von Übergangsstrahlung durch relativistische Teilchen im TRT können Elektronen von anderen geladenen Teilchen unterschieden werden.

Das elektromagnetische Kalorimeter besteht aus sich abwechselnden Schichten von Flüssig-Argon und einer akkordeonartig aufgebauten Bleiabsorberstruktur mit Kapton-Elektroden. Es ist aufgeteilt in Barrel- und Endkappenbereich. Die Raumwinkelabdeckung beträgt $\eta \leq 1{,}5$ im Zentralbereich und $\eta \leq 3{,}2$ im Vorwärtsbereich.

Das hadronische Kalorimeter ist unterteilt in Tile-Kalorimeter, Endkappenkalorimeter und Vorwärtskalorimeter. Das Tile-Kalorimeter besteht abwechselnd aus Stahlplatten für den Absorber und Szintillatorkacheln als aktives Material.

[5)] Die Pseudorapidität η ist eine räumliche Koordinate, die anstelle des Polarwinkels θ verwendet wird, um den Winkel eines Vektors relativ zur Strahlachse anzugeben. Die Pseudorapidität wird definiert durch $\eta = -\ln(\tan\frac{\theta}{2})$.

[6)] **T**ransition **R**adiation **T**racker

Das Endkappen- und das Vorwärtskalorimeter sind zusammengesetzt aus Kupferplatten und Flüssigargonbereichen, die den aktiven Bereich zum Messen der Hadronen darstellen. Das gesamte hadronische Kalorimeter dient der Rekonstruktion von Jets und der Messung fehlender transversaler Energie in physikalischen Prozessen und deckt einen Bereich von maximal $\eta \leq 4{,}9$ ab.

Das Myonspektrometer befindet sich im äußersten Bereich des ATLAS-Detektors, den nur noch Myonen und nur indirekt nachweisbare Neutrinos durchdringen können. Es besteht aus präzisen Spurkammern, welche zur Spurrekonstruktion der im Magnetfeld abgelenkten Myonen dienen, und aus einem separaten Trigger-Bereich, der schnelle Trigger-Informationen ($< 25\,\text{ns}$) liefert. Die verschiedenen Myonkammern nutzen alle die Gasverstärkung in Proportionalkammern oder Röhren. Drei große supraleitende Toroide erzeugen ein Magnetfeld von etwa 4 Tesla senkrecht zur Flugrichtung der Myonen. Während die Myonkammern den Bereich von $\eta \leq 2{,}7$ abdecken, sind die Trigger auf einen Bereich von $\eta \leq 2{,}4$ beschränkt.

Das Triggersystem des ATLAS-Detektors ist dreistufig und besteht zunächst aus dem Level 1 Trigger. Dieser nutzt Teile des elektromagnetischen und hadronischen Kalorimeters, sowie den Myontrigger, um physikalisch interessante Signaturen zu detektieren. Der Level 1 Trigger erlaubt eine Reduktion der Datenrate von 40 MHz auf 75 kHz. Der darauf folgende High-Level-Trigger besteht aus Level 2 Trigger und Event-Filter, auf denen eine entsprechende Software das Ereignis genauer analysiert und so die Datenmenge auf 200 Hz reduziert.

Der zukünftige ATLAS-Detektor

Durch das Upgrade des LHC zum HL-LHC wird der ATLAS-Detektor in einer etwa zehnfach höheren Strahlungsumgebung[7)] und Spurdichte im Vergleich zum gegenwärtigen Betrieb voll funktionsfähig sein müssen. Von den erforderlichen Umbaumaßnahmen ist vor allem der Spurdetektor betroffen, der im Phase 2 Upgrade komplett erneuert werden wird. Die Hauptgründe hierfür sind die schon bestehenden Strahlenschäden am Detektor und die höheren Belegungsdichten durch höhere Teilchenraten. Diese Teilchenraten führen auch zu einer Vergrößerung der Datenauslesegeschwindigkeit.

Der TRT wird die erhöhten Teilchen- und schnelleren Datenraten nicht mehr verarbeiten können, so dass dieser komplett durch einen Silizium-Detektor ersetzt werden wird.

Der zukünftige Spurdetektor wird deshalb folgendermaßen für das Phase 2 Upgrade geplant: Im inneren Bereich werden sich Pixelsensoren befinden, die von Mikrostreifen-Sensoren umgeben sind. In Abbildung 2.10 ist das Layout des geplanten Spurdetektors zu sehen. In rot sind die 4 Lagen des Pixeldetektors im Zentralbereich und die 6 Pixel-Endkappen im entsprechenden Abstand zur Strahlachse eingezeichnet.

[7)] Die genaue Strahlungsumgebung für HL-LHC wird in Kapitel 4.2.1 diskutiert.

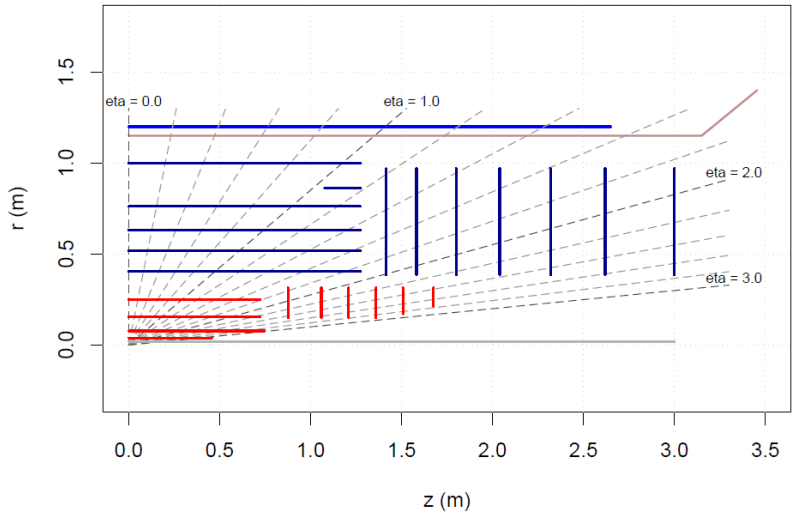


Abbildung 2.10: Das Baseline-Layout des geplanten Spurdetektors. [12] Gezeigt sind die Lagen des Barrel- und des Endkappenbereichs des Pixeldetektors (rot) und des Streifendetektors (blau) in Abhängigkeit des Abstandes zum Wechselwirkungspunkt in z-Richtung und zum radialen Abstand der Strahlachse r.

Auf den genauen Aufbau des Pixeldetektors wird in Kapitel 2.6 noch ausführlicher eingegangen. In blau sind die Lagen des Streifendetektors eingezeichnet: Es folgen 3 Streifen-Lagen (mit kurzen Streifen-Modulen) und 2 Streifen-Lagen (mit langen Streifen-Modulen) auf den Pixeldetektor im Zentralbereich. Im Vorwärtsbereich sind 7 Streifen-Endkappen geplant. Zwischen den letzten Streifen-Lagen ist in der Abbildung 2.10 noch eine kurze eingeschobene Lage eingezeichnet, der „Stub-barrel“. Dieser wird eingeführt, um ein Loch in der Treffer-Abdeckung zwischen Zentral- und Endkappenbereich des Streifendetektors zu verhindern. Die Raumwinkelabdeckung des Spurdetektors mit insgesamt etwa 700 Millionen Auslesekanälen soll $|\eta| < 2{,}5$ betragen. Um den Spurdetektor ist in Abbildung 2.10 noch eine weitere Lage in hellerem blau eingezeichnet, die aus einem Polyethylen-Moderator besteht und die den Neutronenfluss ausgehend von den Kalorimetern reduzieren soll.

Die erhöhte Teilchenrate benötigt eine Erneuerung der Ausleseelektronik der Kalorimeter. Die Front-End-Elektronik wird aufgrund der schon erfolgten Strahlenschädigung durch den LHC-Betrieb zu stark durch Strahleneffekte geschädigt sein und muss ausgetauscht werden. Somit kann sie auch gleichzeitig für die höheren Datenraten verbessert werden. Auch die Triggerelektronik wird verbessert. Es wird ein zusätzlicher Level 0 Trigger eingebaut, der die Daten auf eine Rate von 500 kHz reduzieren wird, bei einer maximalen Latenzzeit von 6 μs. Dieser Trigger wird auch Informationen vom Spurdetektor mit in die Auswahl der geeigneten Daten einfließen lassen.

Für die Myonkammern und die Myontrigger ergeben sich die gleichen Einschränkungen wie für die anderen Detektoren.

Unter der Berücksichtigung einer erhöhten Strahlenumgebung und einer viel höheren Teilchenrate muss der Detektor genauso performant sein wie der momentane ATLAS-Detektor.

2.6 Der ATLAS-Pixeldetektor für den HL-LHC

Das bisher geplante Layout des zukünftigen Pixeldetektors basiert auf den Erfahrungen mit dem momentanen Spurdetektor vereint mit den Anforderungen für den HL-LHC. Aufgrund der viel höheren Teilchenraten im Betrieb des HL-LHC muss die Geschwindigkeit der Datenauslese im Pixeldetektor erhöht sowie eine höhere Auflösung der Pixel erreicht werden, um die Belegungsdichte zu reduzieren. Es soll eine gute zwei Teilchen Separation ermöglicht werden, da die Hauptaufgabe des Pixeldetektors die Vertexrekonstruktion ist.

Grundlage für den geplanten Pixeldetektor sind die Pixelmodule, also die aktiven Sensoren, die mit der Ausleseelektronik verbunden sind. Die Sensoren sind mit den Auslesechips (Front-End-Chips) elektrisch kontaktiert. Die Module sind auf langen Trägerstrukturen, sogenannten Staves, aufgeklebt. Insgesamt soll es im Zentralbereich des Detektors 4 Lagen dieser Staves geben sowie 6 Endkappen. Darauf sollen sich Pixelmodule mit unterschiedlichen Zahlen an FE-Chips (2, 4 und 6) befinden.

Die innersten beiden Lagen des Pixeldetektors sollen separat auswechselbar aufgebaut sein und werden deshalb in der Inner Support Tube (= IST) zusammengefasst, welche einen Außenradius von 110 mm hat. Für die Module in der IST ist eine Pixelfläche von $25 \times 150\,\mu\text{m}^2$ geplant.

Wiederum separat montierbar sollen die anderen beiden Lagen im Zentralbereich des Detektors zusammen mit den Disks in der Pixel Support Tube (= PST) zusammengefasst werden. Für die PST ist ein Außenradius von 345 mm geplant. Die Pixelflächen der Module in der PST sind mit $50 \times 250\,\mu\text{m}^2$ größer als die der IST.

In Tabelle 2.1 sind die Zahlen der Pixelmodule in der Summe und pro Stave bzw. Disk Sektor zu finden. Für die Disks wurden die Zahlen für die 6-fach und die 4-fach-Module zusammengefasst. Nur die Module der innersten Lage sind Module mit 2 Sensoren.

Die Art der Sensoren ist für den geplanten Pixeldetektor noch offen. Es stehen 3 verschiedene Sensortypen zur Auswahl: Planare Sensoren, 3D-Sensoren und Diamant-Sensoren. Die planaren Sensoren sind schon vielfach eingesetzt worden und dementsprechend ist die Erfahrung mit ihnen sehr groß. Außerdem sind planare Sensoren im Vergleich die kostengünstigsten. Sie sind allerdings nicht so strahlenhart. 3D-Sensoren sind aufgrund ihrer kleineren Depletionsfläche toleranter gegenüber Strah-

Lage/Disk	Zahl der Module (2,4 und 6 Sensoren)	Module pro Stave /Disk Sektor	Zahl der FE-Chips pro Modul
Lage 1	352	22	2
Lage 2	576	36	4
Lage 3	1120	35	4
Lage 4	1785	35	4
Disk 1	288	12	6 bzw. 4
Disk 2	288	12	6 bzw. 4
Disk 3	288	12	6 bzw. 4
Disk 4	288	12	6 bzw. 4
Disk 5	296	13	6 bzw. 4
Disk 6	216	9	6 bzw. 4

Tabelle 2.1: Die Zahl der Module im zukünftigen Pixeldetektor

lenschäden. Sie benötigen nur eine geringe Depletionsspannung, sind aber in der Produktion teurer. Die Diamant-Sensoren haben den großen Vorteil, dass sie sehr strahlenhart sind und zusätzlich keine Kühlung brauchen. Von Nachteil ist, dass Diamant-Sensoren hohe Kosten verursachen — vor allem für einen Detektor mit einer solch großen Sensorzahl.

Der FE-Chip digitalisiert die Daten[8)] der Sensoren und gibt diese an die End-Of-Stave-Karte (= EoS-Karte) weiter. Für die inneren Lagen des Pixeldetektors wird aufgrund der kleineren Sensorgröße ein neues FE-Design benötigt. Die FE-Chips werden im 65 nm Prozess realisiert werden. Für die äußeren Lagen soll der sich auch in der Entwicklung befindende FEI4[9)] in modifizierter Form eingesetzt werden.

Die Datenrate beträgt voraussichtlich 2,5 GBits/s für die inneren und 640 MBit/s für die äußeren Lagen. Vom FE-Chip werden die Daten auf der EoS-Karte vom Giga-Bit Transceiver (= GBT)zusammengefasst und an den optischen Link weitergegeben, der die elektrischen Signale in optische umwandelt.

Die FE-Chips produzieren im Betrieb eine nicht vernachlässigbare Wärme. Aus diesem Grund muss der Detektor gekühlt werden. Angestrebt wird eine CO_2 Verdampfungskühlung, die über sehr dünne Röhrchen, die sich in den Staves befinden, die Wärme im Detektor abführt.

Das endgültige Layout des zukünftigen Pixeldetektors steht noch nicht fest. Momentan werden drei verschiedene Layouts diskutiert: Das Conical Layout, das Alpine Layout und das 5-Pixel-Lagen Layout.

[8)] Mithilfe der Methode der „Time Over Threshold“ wird die deponierte Ladungsmenge in den Pixel-Zellen bestimmt.

[9)] Der FEI4 wird zur Zeit für das IBL-Upgrade im Jahr 2013 entwickelt, bei dem der Pixeldetektor eine zusätzliche innere Lage erhält.

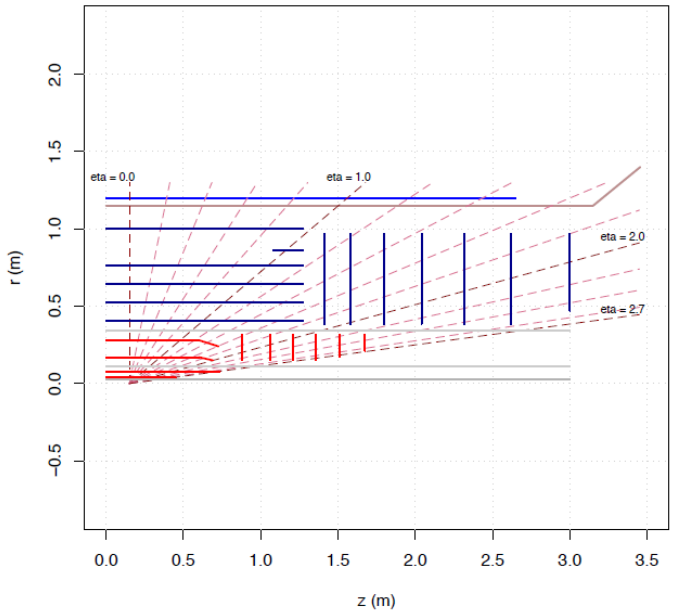


Abbildung 2.11: Das Conical Layout als eine von drei Optionen für das zukünftige Pixel-Layout [12]

Beim Conical Layout sind die Enden des Staves in Richtung der Strahlachse gebogen, wie in Abbildung 2.11 zu sehen ist. Diese Bauweise erlaubt einen größeren Radius für die äußeren Pixellagen (zur besseren Spurrekonstruktion), ohne eine zu große Lücke zu den Endkappen zu erzeugen. Des Weiteren können die Endkappen näher am Zentralbereich angebracht werden, was vorteilhaft für die mechanische Stabilität des Detektors ist. Durch die konische Bauweise können die EoS-Karten auf höhere Pseudorapiditäten (vgl. Abbildung 2.11) gebracht werden, so dass das inaktive Material im Zentralbereich reduziert wird.

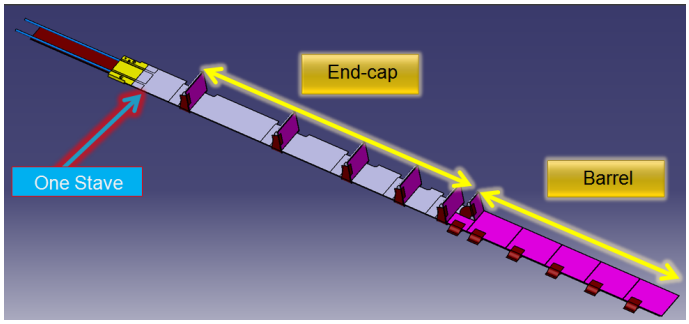


Abbildung 2.12: Veranschaulichung des Alpine Stave Layouts [13]

Bei der Option des Alpine Layout für den Pixeldetektor werden die Module des Endkappenbereichs auf eine keilförmige Trägerstruktur angebracht. Dadurch wird Mate-

rial für den Bau der Endkappen eingespart. Eine Veranschaulichung dieses Layouts ist in Abbildung 2.12 gezeigt.

Als dritte Option wird die Erweiterung um eine Pixellage in Erwägung gezogen. Ob die Staves im Zentralbereich dabei alle dieselbe Länge haben werden oder ob die äußeren Lagen kürzere Staves im Zentralbereich haben werden, wird noch diskutiert.

2.6.1 Die End-Of-Stave-Karte

Für das Phase 2 Upgrade des ATLAS-Detektors ist geplant, dass die Module eines halben Staves jeweils eine Gruppe ergeben. Jede Modul-Gruppe soll gemeinsame Leitungen für die Ein- und Ausgänge der Datenübertragung, der Versorgung mit Hochspannung und Niederspannung sowie Leitungen für das Detektor-Kontroll-System erhalten. Alle diesen elektrischen Leitungen sollen über die End-Of-Stave-Karte (= EoS-Karte) aus dem Detektor geführt werden. Es ist geplant, dass sich an jedem Ende der Staves eine EoS-Karte befinden wird. Das bedeutet, pro Stave soll es 2 EoS-Karten geben.

Zusätzlich zu den elektrischen Leitungen soll die EoS-Karte auch aktive Komponenten beinhalten. Zum Einen befindet sich dort der GBT, der die physikalischen Daten vom Detektor an den optischen Link weiter gibt und umgekehrt die Daten zu den jeweiligen Detektormodulen sendet. Es wird diskutiert, ob für den Pixeldetektor die optische Datenübertragung auf der EoS-Karte beginnen kann, da dort die Strahlungsumgebung noch zu hoch ist. Der optische Link, der für die Umwandlung der elektrischen Signale in optische zuständig ist, würde sich dann weiter außerhalb befinden. Neben dem GBT ist auch der DCS-Chip auf der EoS-Karte lokalisiert. Dieser wird Informationen über die Modultemperaturen und Spannungen erhalten und diese über den DCS-Controller zum Kontrollraum senden. Wie der DCS-Chip diese Daten erhält, steht noch nicht fest. Eine genaue Beschreibung der Funktionalität des DCS-Chips und des DCS-Controllers wird im folgenden Kapitel gegeben.

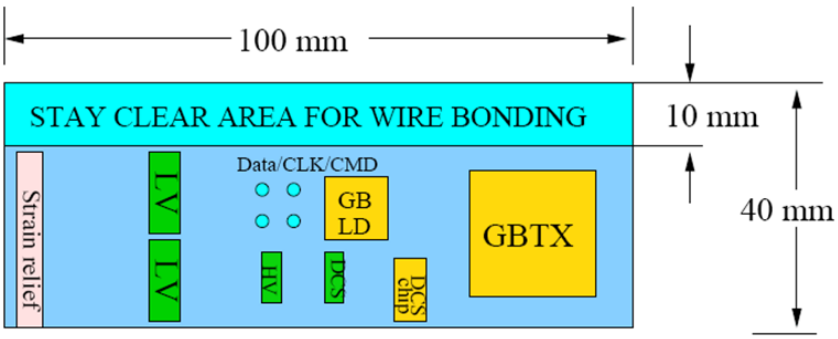


Abbildung 2.13: Schema der End Of Stave Karte [14]

Das Schema der voraussichtlichen EoS-Karte ist in Abbildung 2.13 zu sehen. In grün sind die Stecker für die Versorgungsleitungen eingezeichnet und in gelb die aktiven Komponenten, wie DCS-Chip, Giga-Bit Transceiver (GBTX) und Giga-Bit Laser Driver (GB LD). Die Größe der EoS-Karte wird voraussichtlich $40 \times 100\,\mathrm{mm}^2$ betragen. Das Layout der EoS-Karte für die innerste Pixellage wird aufgrund der größeren Zahl an Auslesekanälen etwas anders aussehen. Es wird sich auf diesen EoS-Karten die Zahl der Komponenten pro Karte erhöhen.

Kapitel 3

Das Kontrollsystem für den zukünftigen ATLAS-Pixeldetektor

Das Detektor-Kontrollsystem überwacht und steuert alle Parameter, die zum zuverlässigen Betrieb des Detektors notwendig sind. Es gewährleistet zudem ein komfortables Ein- bzw. Ausschalten des Detektors. Zusätzlich dazu reagiert das Kontrollsystem im Falle eines Fehlers und kann den Detektor kontrolliert vor Schädigung bewahren und somit eine optimale Datennahme garantieren. Diese Fehler können durch Überhitzung, einen zu hohen Stromverbrauch oder auch durch eine zu hohe Luftfeuchtigkeit verursacht werden. Tritt solch ein Fall auf, sind die Detektor-Komponenten unmittelbar gefährdet und müssen vom Kontrollsystem geschützt werden. Deshalb überwacht das Detektor-Kontrollsystem permanent Temperaturen, Spannungen und Ströme der Detektor-Komponenten.

Für die Analyse der physikalischen Daten liefert das Kontrollsystem die nötigen Informationen, um die Qualität der Daten zu bestimmen. Nur qualitativ hochwertige Daten, die wirklich physikalische Informationen enthalten, können für die Analysen verwendet werden. Dabei ist es wichtig, eine minimale integrierte Luminosität zu erreichen, und zusätzlich dürfen keine größeren Fehler im Trigger-System, bei der Datennahme oder im Detektor aufgetreten sein.

Das Detektor-Kontrollsystem ermöglicht also eine einfache Bedienung des Detektors und unterstützt den reibungsfreien Ablauf der Datennahme. Deshalb muss solch ein System sehr robust aufgebaut sein, damit es in jedem Fall einwandfrei reagieren kann und keine Fehler verursacht.

Die Schwierigkeit besteht darin, ein solches System unter der Voraussetzung aufzubauen, das Material des zukünftigen Detektors zu reduzieren, obwohl dieser eine mehr als dreifache Anzahl an Auslesekanälen haben wird. Betrachtet man dies unter dem Aspekt der um das zehnfach angestiegenen Strahlungsumgebung im Pixeldetektor, wird schnell klar, dass das Konzept des bisherigen Kontrollsystems nicht übernommen werden kann.

3.1 Anforderungen an das Detektorkontrollsystem

Wie oben schon beschrieben, garantiert das Detektor-Kontrollsystem eine reibungsfreie Datennahme sowie die Sicherheit des Detektors.

Für den DCS-Operator müssen benutzerfreundliche Software-Programme vorhanden sein, damit dieser den Detektor sicher steuern und überwachen kann. Eine direkte Rückmeldung nach der Änderung einer Betriebsgröße des Detektors, wie z.B. des Stroms oder der Spannung, ist dringend notwendig, um auf kritische Situationen umgehend reagieren zu können. Der Status der Detektorversorgung, der Kühlung sowie der Detektorumgebung muss permanent bekannt sein.

Des Weiteren muss das zukünftige ATLAS-Pixel-DCS die neuen Ansätze für die Inner TracKer (= ITK) Detektorversorgung berücksichtigen. Es sind zwei verschiedene Versorgungsansätze in der Diskussion: die serielle und die parallele Versorgung des Detektors.

Für die Komponenten innerhalb des Detektorvolumens muss zudem berücksichtigt werden, dass der Leistungsverbrauch so gering wie möglich zu halten ist, denn die vom Verbraucher verursachte Wärme muss während des Detektorbetriebs abgeführt werden, was nur begrenzt möglich ist. Außerdem soll das Kontrollsystem auch bei ausgeschaltetem Detektor[1)] funktionsfähig sein, also auch wenn keine Kühlung des Detektors vorhanden ist.

Eine weitere wichtige Anforderung an das zukünftige Detektor-Kontrollsystem ist die angestrebte Materialreduktion im Pixel-Detektorvolumen. Die Zahl der Detektor-Module verdreifacht sich knapp, wobei der zur Verfügung stehende Platz unverändert bleibt. Es muss also die Zahl der Versorgungsleitungen reduziert werden.

Aufgrund der zehnfach höheren Strahlungsumgebung des ATLAS-Detektors bei HL-LHC müssen die DCS-Komponenten im Detektor sehr strahlenhart sein. Für diese Komponenten muss die höchste Zuverlässigkeit garantiert werden, da der gesamte Pixeldetektor nach Inbetriebnahme nicht mehr erreichbar ist und keine Komponente repariert oder ausgetauscht werden kann.

Zusätzlich zu den bisherigen Anforderungen muss der Detektor vor Überhitzung geschützt werden, falls ein Fehler im Kühlsystem oder eine extreme lokale Erhitzung des Detektors auftritt.

Zuletzt ist es wichtig, dass die Integration des Pixel-DCS-Systems in das ATLAS-DCS-System gewährleistet ist, damit im späteren Betrieb mehrere ATLAS-Subdetektoren gleichzeitig überwacht und gesteuert werden können.

Die Überwachungsgrößen des zukünftigen DCS-Systems sind in Tabelle 3.1 dargestellt. Aufgrund der Anforderungen an das System variieren sie mit der Präzision, Verfügbarkeit, Granularität und Zuverlässigkeit.

[1)] Diese Funktion des Kontrollsystems ist notwendig für Tests (z.B. Kabeltests), aber auch zur Kalibration des Detektors.

Detektorbereich	Parameter	Kontrollgröße
Detektormodul	Niederspannung	Ein/Aus Überwachung von Strom und Spannung Steuerung der Betriebsparameter
	Hochspannung	Ein/Aus Überwachung von Strom und Spannung Einstellbarkeit der Betriebsspannung
	Temperatur	Überwachung der Temperatur Schutz vor Überhitzung
EoS-Karte	Niederspannung	Ein/Aus Überwachung von Strom und Spannung Steuerung der Betriebsparameter
	Temperatur	Überwachung der Temperatur
	Neu- start	Senden eines Reset-Signals Überwachung des Reset-Signals
Detektorumgebung	Temperatur	Überwachung der Temperatur
	relative Feuchte	Überwachung der Feuchteumgebung

Tabelle 3.1: Die Überwachungsgrößen des zukünftigen Detektor-Kontrollsystems

Eine DCS-Ausleseeinheit wird voraussichtlich jeweils aus Gruppen von acht Detektor-Modulen bestehen, einer sogenannten Versorgungseinheit.

3.2 Aufbau des zukünftigen DCS

Aus den Anforderungen an das zukünftige Detektor-Kontrollsystem heraus wurde das im Folgenden beschriebene System entwickelt. Dieses ist in drei Pfade unterteilt, welche sich in Verfügbarkeit und Granularität unterscheiden, so dass der Detektor einerseits vollständig automatisch gegenüber Überhitzung abgesichert ist (Safety-Pfad), aber auch möglichst detaillierte Informationen über den Zustand des Detektors jederzeit (Control- und Feedback) und auf Abruf noch detaillierter (Diagnostics-Pfad) vorliegen.

In Abbildung 3.1 ist die komplette Architektur des zukünftigen Kontrollsystems abgebildet. Im oberen Bereich ist die sich im Counting Room[2)] befindende Elektronik zu sehen, bestehend aus Datennahme-Ausleseelektronik (DAQ crates), Detektor-Kontroll-Stationen (mehrere Rechner, auf denen sich die Software für das DCS befindet) und den Versorgungseinheiten des Detektors. In ca. 100 m Entfernung be-

[2)] Es gibt bei ATLAS zwei Counting Rooms. Dies sind Hallen, in denen sich die Ausleseelektronik befindet und wo keine hohe Strahlenbelastung herrscht.

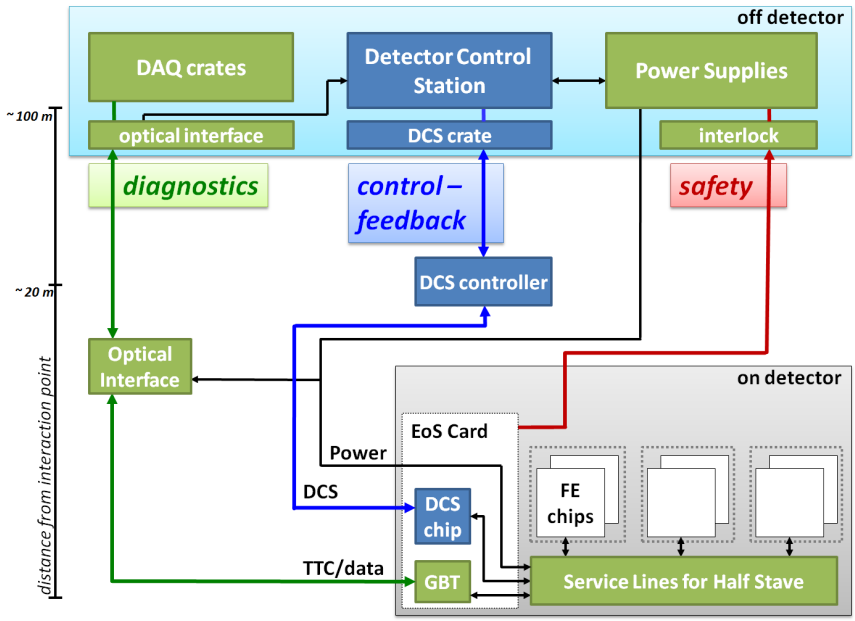


Abbildung 3.1: Der Aufbau des zukünftigen Detektorkontrollsystems [15]

findet sich das Volumen des Pixeldetektors, von dem eine Ausleseeinheit bestehend aus Half-Stave und EoS-Karte zu sehen ist.

Die drei DCS-Pfade sind in Abbildung 3.1 farblich getrennt. In grün ist der Weg der DCS-Daten eingezeichnet, welche im Diagnostics-Pfad über den optischen Datenpfad mitgesendet werden. In blau sind die Komponenten des Control und Feedback Pfades eingezeichnet und in rot ist der Safety-Pfad dargestellt. Im Folgenden wird nun im Detail auf den Aufbau und die Funktion der einzelnen DCS-Pfade eingegangen.

Der Safety-Pfad

Der Safety-Pfad, oder auch das Interlocksystem, wird als komplett eigenständiges, statisch verdrahtetes System aufgebaut. Die Ausleseelektronik ist im Kontrollraum lokalisiert, von dem aus die Leitungen zu den Temperatursensoren im Detektor geführt werden. Im Pixeldetektor werden sich jeweils ein Temperatursensor pro Versorgungseinheit und zwei Sensoren pro Kühlkreis auf der Kühlstruktur befinden. Des Weiteren könnten zusätzliche Temperatursensoren an verschiedenen kritischen Stellen, an denen die Wärme nicht sehr gut abtransportiert werden kann, eingebaut werden. Im Falle einer Überhitzung werden Teile des Detektors über das Interlocksystem direkt an den Versorgungseinheiten, die sich auch im Kontrollraum befinden, abgeschaltet. Im Normalfall sollte bei einem ungewöhnlichen Temperaturanstieg der Detektor-Komponenten schonender für die Elektronik vorgegangen und eine gewisse Abschaltreihenfolge durch ein automatisiertes DCS-Skript eingehalten werden. Ist das aber nicht der Fall, so muss das Interlocksystem eingreifen.

Das Interlocksystem muss stets betriebsbereit sein, also für jeden Zustand des Detektors (während der Installation, Kabeltests, Kalibration, Datennahme etc.) funktionieren. Außerdem muss das Interlocksystem absolut fehlerfrei reagieren können, um einerseits die Detektor-Komponenten nicht zu gefährden, aber andererseits auch nicht fälschlicherweise Teile des Detektors auszuschalten. Deshalb wird es, genauso wie das momentane Interlocksystem, komplett eigenständig und zudem unabhängig von Softwareroutinen aufgebaut werden. Das bedeutet allerdings, dass die Granularität dieses Systems sehr stark eingeschränkt werden muss, um den Leitungsverbrauch so gering wie möglich zu halten. Es ist aber auch nicht unbedingt notwendig, einzelne Detektormodule bei einem starken Temperaturanstieg ausschalten zu können, da meist auch die umliegenden Module betroffen sind. Die generelle Möglichkeit, Detektormodule einzeln abzuschalten, besteht durch den Control und Feedback Pfad.

Der Control und Feedback Pfad

Der Control und Feedback Pfad misst alle wichtigen Betriebsparameter, wie Spannungen, Ströme und Temperaturen des Detektors, die in Tabelle 3.1 genannt sind. Zusätzlich sind digitale Ausgänge zum Schalten von Detektor-Komponenten, wie z.B. einzelnen Detektor-Modulen oder des GBT, vorgesehen. Auf jedem Detektor-Modul wird sich im Pixeldetektor ein Temperatursensor befinden, so dass die kleinste Ausleseeinheit hier jeweils ein Detektor-Modul ist. Die Hochspannung wird direkt an der Versorgungseinheit im Kontrollraum gemessen, somit werden keine weiteren Leitungen im Detektor benötigt. Die Niederspannungen müssen je nach Konzept für die Detektorversorgung auf unterschiedliche Weise gemessen werden, die Konzepte hierzu werden im Folgenden noch genauer erläutert.

Um zusätzliche lange Leitungen im Detektor zu sparen, sollten Ströme und Temperaturen so nah wie möglich an den Modulen im Detektor gemessen werden. Dazu muss eine neuartige DCS-Komponente entwickelt werden, die drei wichtige Kriterien erfüllt: Die Komponente muss sehr leicht sein, zusätzlich aber auch strahlenhart, und sie sollte einen geringen Leistungsverbrauch haben. Diese Kriterien können nur durch einen speziellen, sehr kleinen Schaltkreis (ASIC, Chip) erfüllt werden, der im geeigneten Halbleiter-Prozess entwickelt wird. Das Chipdesign wurde in der Wuppertaler DCS-Gruppe bisher nicht durchgeführt, ist für das zukünftige Kontrollsystem aber unumgänglich und startet somit bei der Entwicklung des Control und Feedback Pfades.

Für das zukünftige Kontrollsystem wird an zwei unterschiedlichen Chipdesigns entwickelt: dem DCS-Chip und dem DCS-Controller. Der DCS-Chip wird die DCS-Daten im Detektor messen und sie an den DCS-Controller senden, der die Daten von mehreren DCS-Chips bündelt und an die Ausleseelektronik im Kontrollraum weiter gibt. Umgekehrt werden die DCS-Befehle vom Kontrollraum über den DCS-Controller an die DCS-Chips gesendet. Die Übertragungsstrecke beträgt insgesamt etwa 100 m vom Detektor zum Kontrollraum, dabei wird sich der DCS-Controller ca.

20 m entfernt vom DCS-Chip befinden. Die in Abbildung 3.1 gezeigte Verbindung zwischen DCS-Chip und dem DCS-Controller steht stellvertretend für ein ganzes Netzwerk aus DCS-Controllern und DCS-Chips, welches in Kapitel 3.3 ausführlich beschrieben wird.

Um eine permanente Überwachung und Steuerung des Detektors für sämtliche Betriebszustände zu realisieren, muss der Control und Feedback Pfad jederzeit zuverlässig betriebsbereit sein. Das steht im Gegensatz zum Diagnostics-Pfad, der auf Abruf zusätzliche DCS-Informationen zur Verfügung stellt.

Der Diagnostics-Pfad

Der grüne Pfad aus Abbildung 3.1 ist der Diagnostics-Pfad. Hier werden zusätzliche DCS-Informationen mit einer Auflösung pro FE-Chip über den optischen Datenstrom gesendet. Man erhält hier eine deutlich höhere Informationsdichte und braucht dabei im Detektor kein zusätzliches Material. Diese Methode ist ein weit verbreitetes Konzept für ein Detektorkontrollsystem. Der Nachteil hier ist allerdings, dass dieser Pfad nur betriebsbereit ist, wenn der komplette Detektor lauffähig ist, also der optische Datenpfad funktioniert.

Der Diagnostics-Pfad liefert Informationen über Spannungen, Ströme und Temperaturen. Die Elektronik zur Messung dieser Parameter wird im Front-End-Chip enthalten sein.

3.2.1 Der DCS-Chip

Der DCS-Chip wird sich auf der EoS-Karte befinden, die jeweils zwei Versorgungseinheiten bedienen kann. Es wird im Weiteren von der seriellen Detektorversorgung ausgegangen, da diese sehr wahrscheinlich für den Pixeldetektor verwendet werden wird. Somit werden jeweils acht Detektormodule eine Versorgungseinheit bilden, was sich in der Zahl der Ein- und Ausgänge des DCS-Chips wiederfindet.

Pro Versorgungseinheit wird es einen oder mehrere GBTs geben, so dass ein DCS-Chip mehrere GBTs schalten kann. Mit dem DCS-Chip wird es möglich sein, den GBT in einen Niedrigverbrauchsmodus zu versetzen.

Für das Ein- und Ausschalten von einzelnen Detektormodulen in einer seriell versorgten Kette sind verschiedene Ansätze vorhanden. Der Befehl zum Schalten der Detektormodule soll grundsätzlich vom DCS kommen, ob der DCS-Chip dafür verantwortlich sein wird oder diese Befehle separat geführt werden, ist noch nicht geklärt. Ein Ansatz für die Lösung dieses Problems ist in Kapitel 3.2.2 gezeigt.

Der DCS-Chip wird voraussichtlich 42 Ein- und Ausgänge erhalten. Darunter sind die Leitungen zur Temperatur- und Spannungsmessung von 2×8 Detektormodulen, zur Feuchtemessung sowie zur Überwachung und Steuerung der EoS-Karte enthal-

ten. Eine genaue Aufstellung der Ein- und Ausgänge des DCS-Chips ist in Tabelle 3.2 zu finden.

Zahl der Anschlüsse	Messung oder Steuerung von
18	Temperaturmessung der Detektormodule, je 8 NTCs teilen sich 1 Return-Leitung
16	Niederspannungen per Spannungsteiler
1	kapazitiver Feuchtesensor
1	Referenzkapazität für die Feuchtemessung
2	Temperaturmessung der EoS-Karte
2	Niederspannung der EoS-Karte
4	digitale Ausgänge (2 pro GBT) zum Schalten des GBT
Summe = 44	

Tabelle 3.2: Detaillierte Aufstellung der Ein- und Ausgänge des DCS-Chips

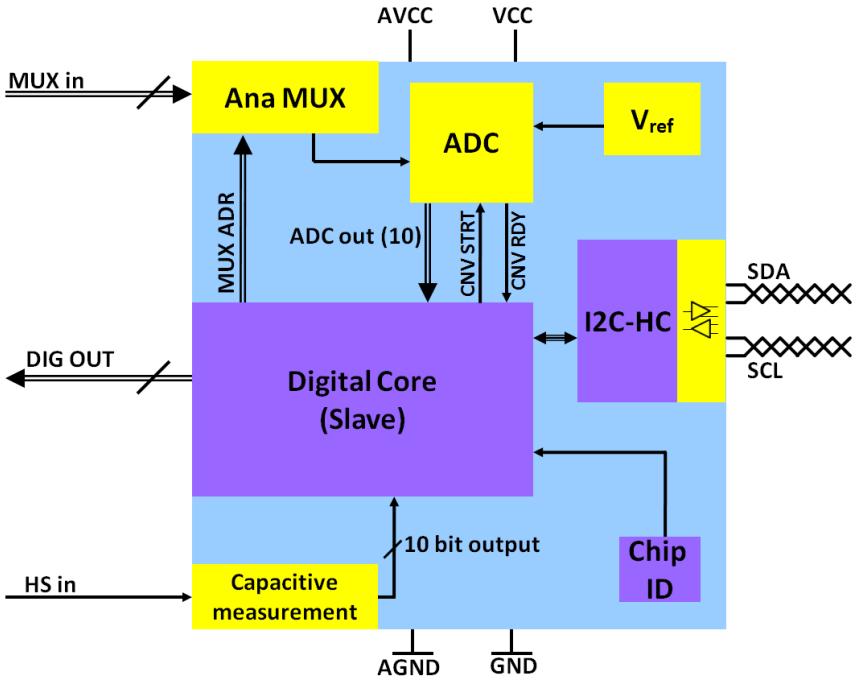


Abbildung 3.2: Schema des DCS-Chips. In gelb sind die analogen, in violett die digitalen Komponenten des DCS-Chips gezeigt. [15]

In Abbildung 3.2 ist das Blockdiagramm des DCS-Chips gezeigt. Die analogen Komponenten sind dabei gelb dargestellt. Unter ihnen befinden sich ein analoger Multiplexer und ein Analog-Digital-Konverter (= ADC) mit 10-Bit Auflösung zur Messung der Temperaturen und Niederspannungen der Detektormodule sowie eine Referenzspannungsquelle für den ADC. Für die kapazitive Feuchtigkeitsmessung werden

sich Schmitt-Trigger und Rückkopplungswiderstände im DCS-Chip befinden („Capacitive measurement“). Der Feuchtesensor und ein Referenzkondensator werden im Detektorvolumen platziert.

Der in violett dargestellte Digitalteil des DCS-Chips beinhaltet den Slave für das Master-Slave-Protokoll I2C-HC, auf welches in Kapitel 5 noch genauer eingegangen wird.

Der Slave erhält die Lese- bzw. Schreibbefehle vom DCS-Controller und führt den gewünschten DCS-Befehl aus. Zur digitalen I2C-HC-Schnittstelle gehört auch eine analoge Komponente, die die Signale für Daten und Clock auf einen differentiellen Bus treibt. Zusätzlich wird jeder DCS-Chip eine Chip-ID zur einfacheren Identifikation erhalten.

Die Funktion des DCS-Chips ist nun im Detail bekannt. Im Folgenden kann somit auf den Einsatz des DCS-Chips im Detektor eingegangen werden.

3.2.2 DCS-Konzepte für zwei unterschiedliche Ansätze der Detektorversorgung

Da bisher noch nicht klar ist, ob die serielle oder die parallele Detektorversorgung verwendet wird, gibt es zwei unterschiedliche Konzepte für die Anbindung des DCS-Chips an die Detektor-Komponenten.

DCS für die parallele Versorgung des Detektors

In Abbildung 3.3 ist das DCS für die parallele Detektorversorgung gezeigt. Zu sehen ist der auf der EoS-Karte lokalisierte DCS-Chip mit den Verbindungen zu einer Ausleseeinheit von 8 Detektormodulen. Der DCS-Chip wird zwei Ausleseeinheiten steuern und kontrollieren können, zur einfacheren Übersicht wurde auf der schematischen Zeichnung nur eine Ausleseeinheit eingezeichnet. Auch Leitungen für die von außen zugeführte Hochspannung, die das DCS direkt an den Versorgungseinheiten im Kontrollraum messen wird, sind nicht eingezeichnet worden.

Die Leitungen für die Niederspannungsversorgung verlaufen über die EoS-Karte. Deshalb kann mit dem DCS-Chip die Niederspannung pro Detektormodul durch sehr kurze Leitungen auf der EoS-Karte abgegriffen werden. Es sind keine zusätzlichen Leitungen zum Detektormodul notwendig.

Auf jedem Detektormodul wird sich ein NTC zur Temperaturmessung befinden. Zur Auslese der Temperatursensoren werden (n + 1) Leitungen benötigt, wenn n Detektormodule vorhanden sind. Die Sensoren teilen sich dabei eine gemeinsame Return-Leitung. Die voraussichtlich 9 Leitungen pro Ausleseeinheit werden nur zwischen Detektormodul und DCS-Chip benötigt und können sehr dünn sein, da es reine Messleitungen sind. Ein weiterer Temperatursensor wird sich auf der EoS-Karte oder

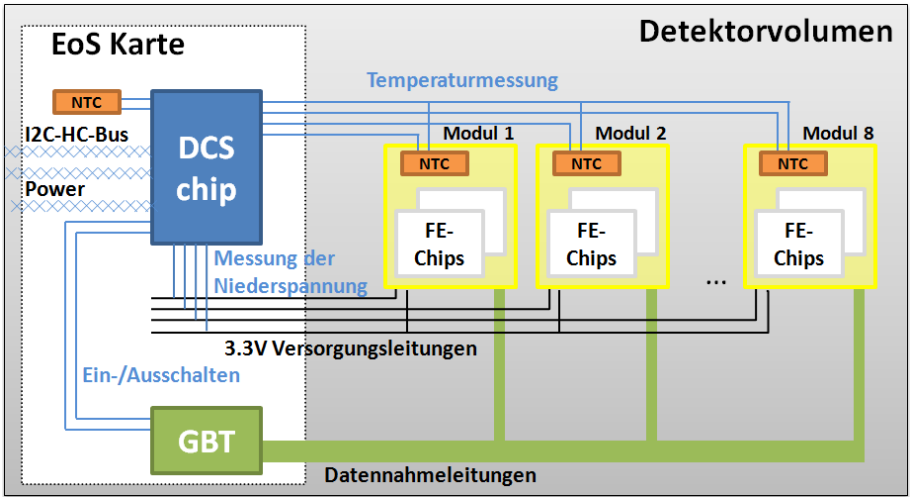


Abbildung 3.3: Das DCS für die parallele Versorgung des Pixeldetektors, wobei nur eine Ausleseeinheit eingezeichnet ist.

der Kühlstruktur befinden. Auch diesen wird der DCS-Chip durch kurze Leitungen auf der EoS-Karte auslesen können.

Auf der EoS-Karte befindet sich auch der GBT. Voraussichtlich wird dieser vom DCS-Chip gesteuert werden können. Der DCS-Chip kann den GBT zurücksetzen bzw. ein- und ausschalten oder ihn in einem Stromsparmodus starten, in dem der GBT auch bei verringerter Detektor-Kühlung betrieben werden kann.

Der DCS-Chip ermöglicht also die Spannungs- und Temperaturmessung pro Detektormodul sowie die Steuerung des GBT. Dazu werden für dieses Detektor-Konzept insgesamt (n + 1) Leitungen im Detektorvolumen benötigt.

DCS für die serielle Versorgung des Detektors

Das Konzept der seriellen Detektorversorgung reduziert die Leitungszahl im Detektorvolumen sehr stark. Ein auftretendes Problem bei diesem Ansatz der Detektorversorgung ist die Frage, wie Detektormodule einzeln ausgeschaltet werden können, ohne dem Detektor zusätzliche Leitungen zuzufügen. Es wird also ein Überbrückungsmechanismus benötigt, der einzelne Module einer seriell versorgten Kette abschalten kann.

Solch ein Mechanismus ist dringend notwendig, da ein zu stark rauschendes Detektormodul vom Bus genommen werden muss, damit die Daten der anderen Module nicht verloren gehen. Durch eine Neukalibration kann ein solches Detektormodul nach einem kurzen Stop der Datennahme eventuell wieder mit in die Auslese aufgenommen werden, dafür sollte der Überbrückungsmechanismus einfach und flexibel steuerbar sein. Für die Überwachung und Steuerung bietet sich der DCS-Chip an,

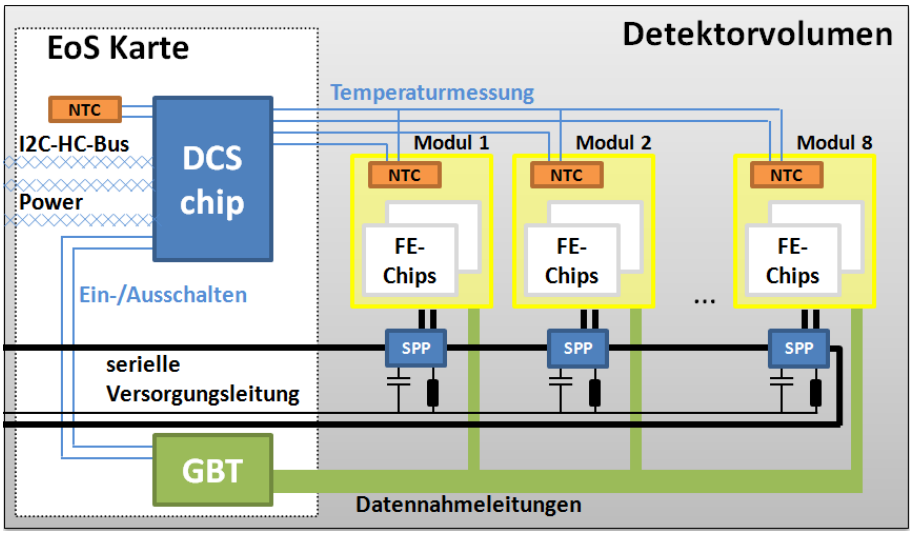


Abbildung 3.4: Das DCS für die serielle Detektorversorgung, schematisch dargestellt anhand einer Versorgungseinheit. Die Messung der Spannungen der Module vom DCS-Chip ist nicht eingezeichnet, da hierfür die genaue Methode noch bestimmt werden muss. [12]

da dieser durch kurze Leitungen auf der EoS-Karte diese Aufgaben übernehmen könnte.

Momentan ist für den Überbrückungsmechanismus der Serial-Powering-Protection-Chip (= SPP-Chip) angedacht, der über einen 1-Draht-Bus versorgt und gesteuert werden können soll. Inwiefern dieser Chip wirklich für die Überbrückung einzelner Pixelmodule in Frage kommt, wird momentan in verschiedenen Aufbauten getestet, da der SPP-Chip für den Streifen-Detektor entwickelt wurde und für den Pixeldetektor größere Ströme schalten muss.

Zwischen den einzelnen Detektormodulen herrschen durch die serielle Stromversorgung unterschiedliche Spannungsniveaus. Sollte der DCS-Chip die Niederspannung mit separaten Leitungen direkt an den Detektormodulen messen, so würde ein Spannungsteiler zusätzlich auf der EoS-Karte benötigt, um die Eingänge des DCS-Chips zu schützen. Um eine größere Materialersparnis zu erzielen, sollten auch andere Konzepte zur Messung der Niederspannung in Betracht gezogen werden. Dazu sind im Folgenden drei Konzepte vorgestellt, zu denen jeweils weitere Studien durchgeführt werden müssen:

1. Die Niederspannung könnte über die HV-Return-Leitung gemessen werden. Bisher ist allerdings noch keine HV-Return-Leitung konkret geplant worden.

2. Ein weiterer möglicher Kandidat zur Messung der Niederspannung wäre die Low Voltage Differential Signal (= LVDS) Datenleitung zwischen FE-Chip und GBT. Dazu müsste die Spannungsmessung vom FE-Chip durchgeführt

werden. Im momentanen FE-Chip sind bisher keine Entwicklungen in diese Richtung geplant.

3. Die wahrscheinlichste Möglichkeit eine Information über die Modulspannungen ohne zusätzliche Leitungen zu erhalten, wäre die Bestimmung der Spannung über die Einschaltreihenfolge durch den SPP-Chip. Dabei würde beim Einschalten der Detektormodule jeweils die Spannungsdifferenz nach jedem eingeschalteten Modul direkt an der Stromquelle messbar sein. Diese Niederspannung könnte einmal beim Einschalten der Detektormodule gemessen, während des Betriebs allerdings nicht mehr aktualisiert werden.

Aus der obigen Aufzählung wird schnell klar, dass dringend ausführliche Studien für die Messung der Niederspannung notwendig sind, wenn das Detektorsystem konkreter bekannt ist.

Wie auch für die parallele Detektorversorgung werden für die serielle Versorgung (n + 1) Leitungen für n Detektormodule zur Temperaturmessung benötigt, sowie kurze Leitungen auf der EoS-Karte zur Auslese eines zusätzlichen Temperatursensors. Der DCS-Chip wird auch für dieses Konzept den GBT ein- bzw. ausschalten oder ihn in einen Stromsparmodus versetzen können.

3.3 Der DCS-Controller-Chip

Das Netzwerk aus DCS-Controller und DCS-Chip bildet die zentrale Einheit des Detektorkontrollsystems.

Der DCS-Controller ist dafür zuständig, die Daten von bis zu 16 DCS-Chips zusammenzufassen und sie an den Kontrollraum weiterzugeben. Umgekehrt empfängt der DCS-Controller Befehle vom Kontrollraum und sendet diese an die DCS-Chips. Dabei muss die Datenübertragung in der Strahlungsumgebung absolut fehlerfrei sein. Für die Gewährleistung einer fehlerfreien Übertragung muss das Netzwerk eine Balance zwischen Redundanz der Komponenten und Minimierung der Strahlungslänge herstellen. Aus diesem Grund werden an jeden DCS-Controller vier Busse angeschlossen sein, an denen sich jeweils bis zu vier DCS-Chips befinden werden.

Das DCS-Netzwerk ist in Abbildung 3.5 gezeigt. Zu sehen ist auch, dass für die Kommunikation vom DCS-Controller zum DCS-Chip ein anderes Protokoll verwendet wird als für die Kommunikation zum Kontrollraum. Das CAN-Protokoll hat eine ausgeprägte Fehlererkennung und kann mit nur zwei Leitungen (twisted pair) Daten über lange Strecken sicher übertragen. Das für die Kommunikation zum Detektor verwendete I2C-HC-Protokoll ist ein einfach-selbstkorrigierendes Protokoll, welches in der hohen Strahlungsumgebung im Pixeldetektor eine verlustfreie, sichere Kommunikation ermöglicht. Auf den Aufbau und die Funktion der Protokolle wird in Kapitel 5 noch genauer eingegangen.

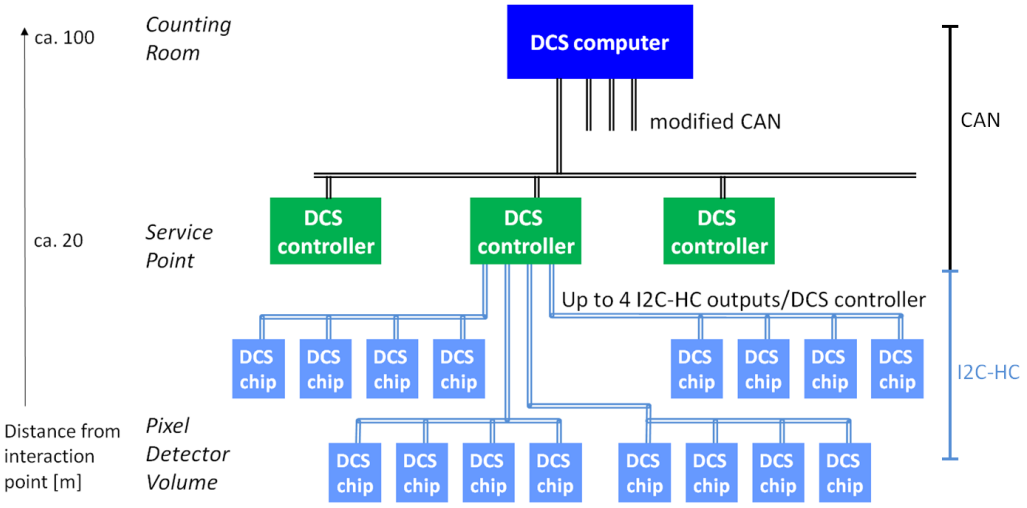


Abbildung 3.5: Das DCS-Netzwerk

Einer der Gründe für den DCS-Controller ist die begrenzte Reichweite der I2C-HC-Kommunikation. Tests zur Bestimmung der Reichweite werden in Kapitel 7.3 dargestellt. Die CAN-Datenübertragung dagegen ist dazu geeignet, für die angestrebten CAN-Bitraten von 250 kHz über eine Distanz von minimal 80 m fehlerfrei zu kommunizieren.

Ein weiterer Grund für einen DCS-Controller ist, dass dieser dazu beiträgt, die Zahl der DCS-Leitungen im Detektorvolumen zu reduzieren. Der zukünftige Pixeldetektor wird eine dreifache Zahl an Pixelzellen verglichen mit dem gegenwärtigen Pixeldetektor haben. Die zur Verfügung stehenden räumlichen Gegebenheiten werden sich dabei allerdings nicht vergrößern. Es sollte also maximal dieselbe Anzahl an Leitungen wie beim jetzigen Detektor aus dem Pixeldetektor herausgeführt werden.

Der DCS-Controller wird bewusst an einer Stelle im Detektor eingebaut, die im Notfall zugänglich ist. Während Komponenten auf der EoS-Karte nach dem Einbau nicht mehr nachträglich repariert werden können, ist das für den 20 m entfernten Service Point schon eher möglich, an dem sich der DCS-Controller befinden wird.

In Abbildung 3.6 ist der DCS-Controller schematisch dargestellt. Dabei sind die digitalen Komponenten in violett und die analogen Komponenten in gelb markiert. Die rechte Seite der Abbildung 3.6 stellt die Anbindung zum Kontrollraum über einen CAN-Bus dar. Der CAN-Knoten beinhaltet das Standard CAN-Protokoll[3)] nach den Bosch-Spezifikationen [16]. Die DCS-Befehle werden dort entgegengenommen. In der „bridge“ werden die DCS-Befehle in das I2C-HC-Protokoll übersetzt und dann mit dem I2C-HC-Master über einen Multiplexer auf einen der vier I2C-HC-Busse zum DCS-Chip im Detektor gesendet. Umgekehrt kann der DCS-Controller

[3)] Version A

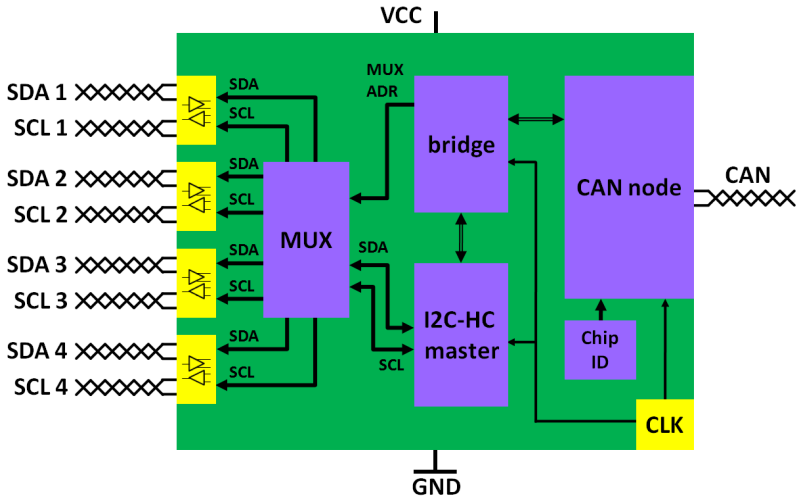


Abbildung 3.6: Schematische Darstellung des DCS-Controllers. In violett sind die digitalen, in gelb die analogen Komponenten dargestellt [15].

die zurückgelesenen Daten des DCS-Chips vom I2C-HC-Master in der bridge für das CAN-Protokoll übersetzen und dann auf dem CAN-Bus zum Kontrollraum senden.

So wie die DCS-Chips wird auch jeder DCS-Controller eine eigene Chip-ID erhalten. Damit ist die eindeutige Identifikation der Komponenten möglich, was für die Kommunikation im DCS-Netzwerk wichtig ist.

3.4 Vergleich der Leitungsanzahl des momentanen und des zukünftigen DCS

Das zukünftige Detektorkontrollsystem wurde unter dem Aspekt der größtmöglichen Sicherheit, aber auch der maximalen Materialreduktion entwickelt. Vor allem die Einschränkung, dass sich zwar die Zahl der Auslesekanäle des Detektors vergrößert, aber der Platz für Versorgungsleitungen derselbe bleibt, muss bei solch einem Design berücksichtigt werden. Im Folgenden soll gezeigt werden, dass das DCS-System für das HL-LHC Upgrade trotz einer höheren Zahl der Auslesekanäle insgesamt weniger Leitungen benötigt. Die Angaben für die Leitungszahlen des zukünftigen Pixeldetektors beruhen auf der in Kapitel 2.6 beschrieben Modularität. Für die Detektorversorgung wird bei der folgenden Berechnung vom Konzept der seriellen Detektorversorgung ausgegangen.

Der zukünftige Pixeldetektor wird insgesamt 3833 Module haben, wobei diese Zahl alle Module (ob 2-fach, 4-fach oder 6-fach Module) umfasst. Diese Module werden in Versorgungseinheiten unterteilt, die jeweils 8 Module beinhalten. Es wird insgesamt 716 Versorgungseinheiten geben. Da in den innersten beiden Lagen mehrere

Versorgungseinheiten von einer EoS-Karte aus versorgt werden, ist die Gesamtzahl der EoS-Karten mit 684 etwas geringer als die der Versorgungseinheiten.

Pro Versorgungseinheit wird ein DCS-Chip benötigt, so dass im gesamten DCS insgesamt 716 DCS-Chips eingesetzt werden. Die Zahl der DCS-Controller steht noch nicht fest. Es werden aber maximal 16 DCS-Chips und minimal 8 DCS-Chips von einem DCS-Controller im DCS-Netzwerk angesteuert. Somit werden zwischen 45 und 90 DCS-Controller zum Einsatz kommen.

Der momentane Pixeldetekor hat insgesamt 1744 Detektormodule, etwa die Hälfte der zukünftigen Auslesekanäle. Für den momentanen Detektor wird die Kabelzahl immer pro Detektormodul berechnet.

Pixeldetektor DCS	Kabelpaare Hochspannung	Kabelpaare Niederspannung	Kabelpaare für DCS
aktuell	1744	3548	5232
für HL-LHC	1432	716 + 684*	2832 + 684*

Tabelle 3.3: Gegenüberstellung der Leitungsanzahl für das gegenwärtige und das zukünftige DCS für das HL-LHC Upgrade. Die mit * markierten Zahlen sind Kabelpaare, die nur bis zur EoS-Karte geführt werden.

Für die Versorgung des Detektors mit Hochspannung wird für das momentane DCS ein Kabelpaar pro Modul benötigt. Für das zukünftige DCS werden aufgrund der seriellen Detektorversorgung pro Versorgungseinheit 2 Kabelpaare benötigt, das entspricht einem Kabelpaar pro 4 Detektormodulen. Die Gesamtzahl der Hochspannungsleitungen ist für das zukünftige DCS geringer als für das momentane, was auch in Tabelle 3.3 gezeigt ist.

Im momentanen Pixeldetektor werden pro Modul 2 Niederspannungs-Kabelpaare benötigt. Das entspricht einer Summe von 3548 Kabelpaaren. Für das zukünftige Layout, in dem drei verschiedene Modulsorten (zweifach, vierfach, sechsfach) vorgesehen sind, wird pro Versorgungseinheit ein Kabelpaar benötigt. Das entspricht insgesamt 716 Leitungen, wie in Tabelle 3.3 zu sehen ist. Die Zahlen für das momentane und das zukünftige DCS sind jedoch nicht direkt vergleichbar, da für die vierfach und die sechsfach Module ein höherer Stromverbrauch durch eine größere Zahl an FE-Chips besteht. Dementsprechend haben die Kabel ein größeres Volumen. Rechnet man dieses um auf die momentan verwendeten Niederspannungskabel, so würden statt der 716 Leitungspaare aus Tabelle 3.3 3088 Kabelpaare benötigt, was unterhalb der Zahl der Leitungen des momentanen Pixeldetektors liegt. Für die Versorgung der EoS-Karte kommt noch ein weiteres Kabelpaar pro EoS-Karte hinzu, also 684 Kabelpaare. Diese Kabelpaare werden nur bis zu den Staveenden geführt und müssen somit separat betrachtet werden.

Für das DCS werden im aktuellen Pixeldetektor pro Modul jeweils 3 Kabelpaare benötigt. Diese Leitungen sind sehr dünn, da für die beiden Niederspannungen zwei

Kabelpaare Messleitungen zur Spannungsmessung sind und das dritte Kabelpaar die Temperaturmessung realisiert. Der zukünftige Pixeldetektor hat diese hohe Granularität im verdrahteten System nicht mehr, es werden dort für die ersten beiden Lagen des Detektors jeweils 4 Kabelpaare und für die restlichen Lagen sowie den Disks 3 Kabelpaare pro Versorgungseinheit verwendet. Des Weiteren benötigt der SPP-Chip pro Versorgungseinheit ein Kabelpaar. Für die Spannungsmessung der EoS-Karte wird pro Karte ein Kabelpaar benötigt, welche, wie in Tabelle 3.3 mit * markiert, nur bis zur EoS-Karte geführt werden.

Die obige Aufzählung zeigt, dass trotz der Verdopplung der Module für den zukünftigen Detektor, die Zahl der geplanten Leitungen unterhalb der des momentanen Pixeldetektors liegt. Der geplante Aufbau des zukünftigen Detektorkontrollsystems bewährt sich also auch hier, da das System mit der gegebenen Räumlichkeit auskommt.

Kapitel 4

Strahlenschäden in digitalen integrierten Schaltungen

Durch das Luminositätsupgrade, wird sich im ATLAS-Detektor die Strahlungsumgebung etwa um das Zehnfache erhöhen. Da mehrere Komponenten des Detektorkontrollsystems dann direkt im Pixeldetektor lokalisiert sein werden, muss es gegenüber dieser erhöhten Strahlungsumgebung resistent und die DCS-Kommunikation absolut fehlerfrei sein.

Dieses Kapitel dient zum Verständnis, welche Effekte berücksichtigt werden müssen und an welchen Stellen spezielle Mechanismen zur Absicherung der DCS-Komponenten implementiert werden sollten. Da das digitale Chipdesign aus CMOS-Komponenten besteht, wird zunächst ihr Aufbau und ihre Funktion beschrieben, bevor im Detail auf die zukünftige Strahlungsumgebung am HL-LHC und die zu erwartenden Strahlenschädigungen in den Komponenten eingegangen wird.

4.1 Aufbau einer digitalen integrierten Schaltung

Eine digitale integrierte Schaltung (oder auch Digitalchip) ist heutzutage meist ausschließlich aus CMOS-Komponenten aufgebaut. CMOS steht für Complementary Metal Oxide Semiconductor. Die CMOS-Komponenten sind aus p-Kanal-MOSFETs[1)] und aus n-Kanal-MOSFETs (= PMOS und NMOS) aufgebaut. Der Aufbau eines n-Kanal MOSFET, ist in Abbildung 4.1.a gezeigt.

In einem p-dotierten Substrat (Siliziumwafer) befinden sich zwei stark n-dotierte Substratschichten, die Drain und Source genannt werden. Oberhalb dieser ist eine sehr dünne Isolationsschicht aus Siliziumdioxid angelegt, die eine ungefähre Dicke

[1)] **M**etal **O**xide **S**emiconductor **F**ield-**E**ffect **T**ransistor

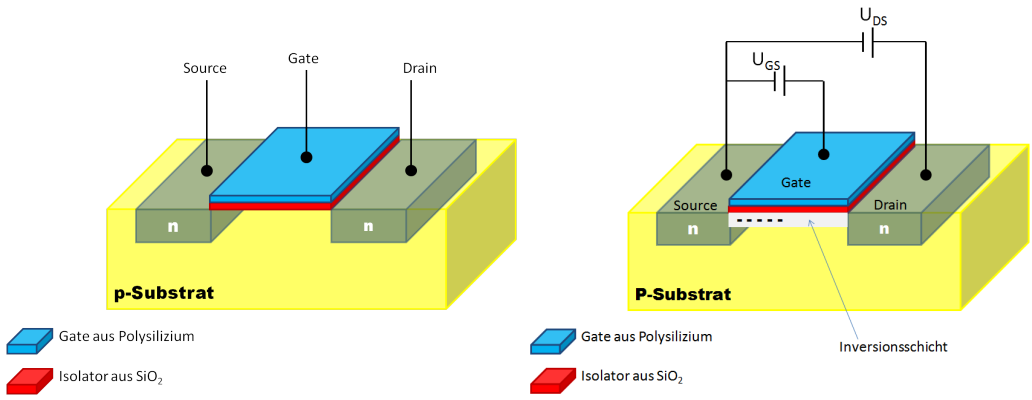


Abbildung 4.1: (a) Aufbau eines NMOS-Transistors und (b) Funktionsweise eines NMOS-Transistors

von 22 Å[2)] hat. Auf die Isolationsschicht wird das Gate aufgetragen, eine leitende Metallschicht aus Polysilizium.

Wird nun an das Gate eine positive Spannung gegenüber der Source gelegt, so bilden die Lagen aus Polysilizium, Siliziumdioxid und dotiertem Silizium den MOS-Kondensator. Durch diesen Kondensator wird nach Überschreiten einer Schwellspannung U_{GS} eine n-leitende Schicht im schwach p-dotierten Substrat geschaffen. Diese Schicht wird Inversionsschicht genannt. Zwischen Drain und Source wird eine weitere Spannung U_{DS} angelegt, durch die der Source-p-Substrat Übergang leitend wird und der Drain-p-Substrat Übergang sperrt. Wird U_{DS} angelegt und vergrößert, während U_{GS} oberhalb des Schwellwertes liegt, können freie Elektronen von Source zu Drain diffundieren und es fließt ein Strom. Dieser Prozess für einen NMOS-Transistor ist in Abbildung 4.1.b verdeutlicht.

Ein PMOS-Transistor funktioniert genau entgegengesetzt: Hierbei befinden sich die stark p-dotierten Source und Drain in einem n-dotierten Substrat.

Ein großer Vorteil von CMOS-Komponenten ist der geringe Leistungsverbrauch, denn nur beim Schaltvorgang des Transistors wird Strom benötigt. Solange U_{GS} unterhalb des Schwellenwertes liegt, fließt kein Strom.

NMOS- und PMOS-Transistoren werden in demselben Substrat hergestellt, dabei ist das n-dotierte Substrat der PMOS-Komponente eine lokale Schicht, die „n-well“ genannt wird.

Aus den NMOS- und PMOS-Transistoren lassen sich stromsparende, digitale Schaltungen aufbauen. Ein einfaches Beispiel dafür ist ein Inverter, dargestellt in Abbildung 4.2. Der Inverter besteht aus einer PMOS- und einer NMOS-Komponente. Wird auf den Eingang A die Versorgungsspannung VDD angelegt, so ist der NMOS-

[2)] Gilt für den in dieser Arbeit verwendeten 130 nm Prozess. [17]

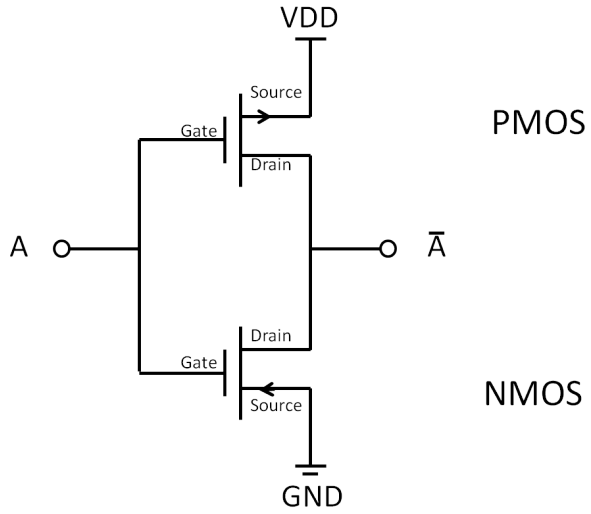


Abbildung 4.2: Aufbau eines Inverters aus CMOS-Komponenten

Transistor leitend und der Ausgang $\bar{A}$ mit GND verbunden, da zum selben Zeitpunkt der PMOS-Transistor sperrt. Liegt umgekehrt der Eingang A auf GND, so wird der Ausgang $\bar{A}$ durch den leitenden PMOS-Transistor mit der Versorgungsleitung verbunden.

4.2 Schädigung durch Teilchenstrahlen

Tritt in der CMOS-Komponente eine Ionisation auf, kann das sehr unterschiedliche Effekte haben. Diese Effekte lassen sich in zwei Arten unterteilen: in die kumulativen Effekte (= TID[3)]-Effekte) und die Einzelfehlereffekte (= SEE[4)]). Während die kumulativen Effekte erst ab einem gewissen Schwellwert messbar werden, treten die Einzelfehlereffekte zufällig verteilt im gesamten laufenden Betrieb auf.

4.2.1 Erwartete Strahlungsumgebung für den HL-LHC

Die erwartete Teilchenrate

Für Studien der Strahlenfestigkeit der DCS-Kommunikation ist die Betrachtung der Einzelfehlereffekte wichtig. Diese treten bei geladenen Hadronen mit einer Energie von E > 20 MeV durch Wechselwirkung mit Siliziumkernen im sensitiven Volumen auf. In Abbildung 4.3 ist ein Fluenzdiagramm für Hadronen mit einer Energie E > 20 MeV für den ATLAS-Pixeldetektor am HL-LHC zu sehen.

[3)] Total Ionizing Dose
[4)] Single Event Effect

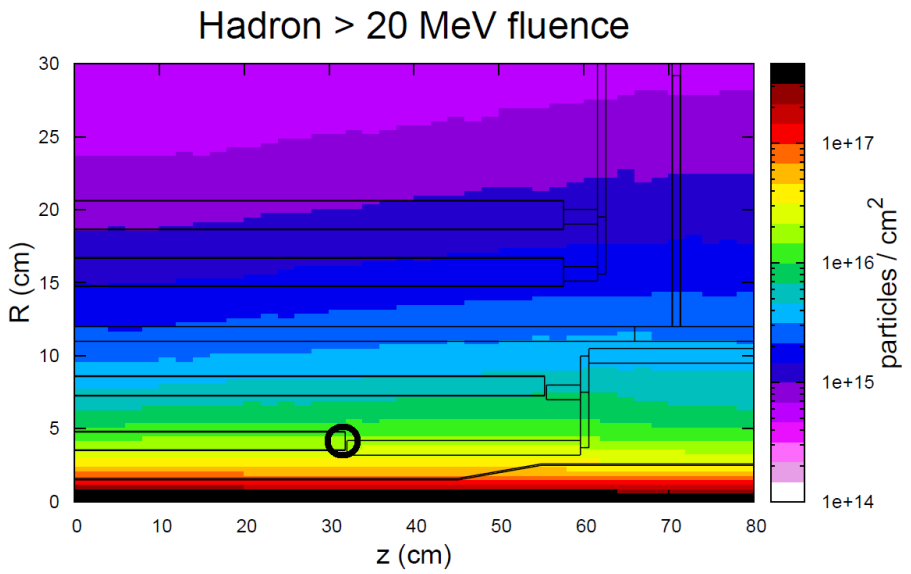


Abbildung 4.3: Erwartete Fluenz für Hadronen mit einer Energie E > 20 MeV für die vier Lagen im Zentralbereich des Pixeldetektors nach dem Luminositätsupgrade [18]

In der Region, in der die EoS-Karte der innersten Pixellage lokalisiert sein wird, wird eine Fluenz von etwa $F = 2 \cdot 10^{16} \frac{Teilchen}{cm^2}$ erwartet. Mit der angestrebten Luminosität von $L = 5 \cdot 10^{34} \frac{1}{cm^2 \cdot s}$ und der integrierten Luminosität $\mathcal{L} = 3000\,fb^{-1}$ lässt sich eine obere Grenze für den erwarteten Teilchenfluss im Bereich der innersten EoS-Karten im Pixeldetektor berechnen. Dabei gelten die Verhältnisse $F = \phi \cdot t$ und $\mathcal{L} = L \cdot t$, so dass:

$$\phi = \frac{L \cdot F}{\mathcal{L}} \tag{4.1}$$

Daraus ergibt sich ein Teilchenfluss von:

$$\phi_{EoS} = 3,3 \cdot 10^8 \frac{\text{Teilchen}}{\text{cm}^2 \cdot \text{s}} \tag{4.2}$$

Die erwartete Gesamtdosis

Die Gesamtdosis entspricht der Energie, die im Material durch radioaktive Strahlung in Form von Ionisationsenergie abgegeben wird. Ausgedrückt wird die gespeicherte Dosis in Gray bzw. Rad, wobei die Beziehung 1 Gy = 100 Rad gilt. Die Dosis wird auch hier für die Region der EoS-Karten der innersten Lage im Zentralbereich des

Pixeldetektors bestimmt, da diese dort am höchsten ist. Aus Abbildung 4.4 ergibt sich eine Gesamtdosis von etwa:

$$D_{EoS} = 10\,\mathrm{MGy} = 1000\,\mathrm{MRad} \tag{4.3}$$

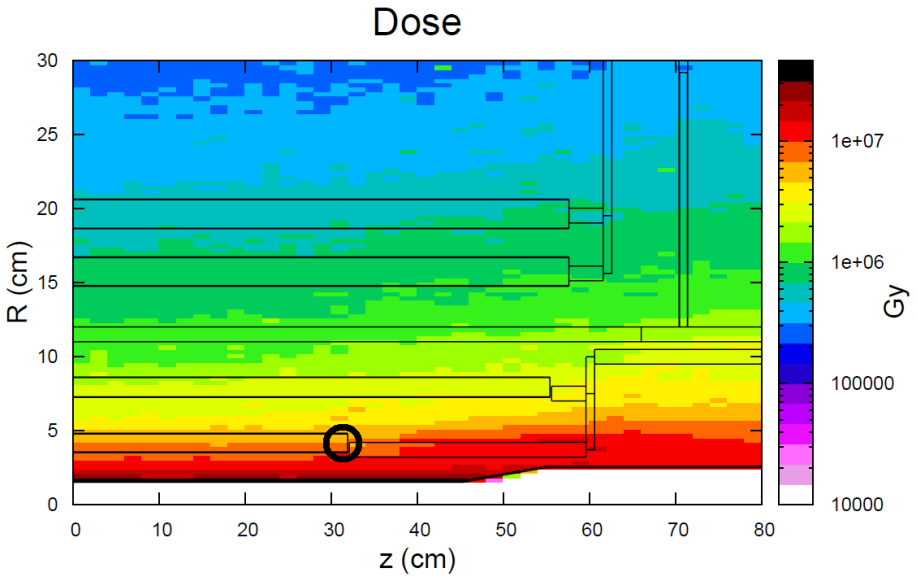


Abbildung 4.4: Erwartete Dosis für die vier Lagen im Zentralbereich des Pixeldetektors nach dem Luminositätsupgrade [18]

4.2.2 Betrachtung von Schädigungen durch Einzelfehlereffekte

Einzelfehlereffekte können zufällig auftreten und müssen somit in Wahrscheinlichkeiten ausgedrückt werden. Die Wahrscheinlichkeit für Einzelfehler hängt von der Teilchenenergie und dem Teilchenfluss sowie von der Komponente selbst ab. Dabei gibt es eine große Zahl an unterschiedlichen Effekten, die zu den SEE gehören. Diese Effekte werden klassifiziert in kurzzeitige und permanente Schädigung der Komponente. Für CMOS-Bausteine sind zwei Effekte relevant, auf die im Weiteren genauer eingegangen wird: den Single Event Upset und den Single Event Latchup.

Single Event Upset

Bei einem Single Event Upset (= SEU) wird durch indirekte Ionisation zusätzliche Ladung erzeugt. Inelastische bzw. elastische Kernwechselwirkungen mit dem Silizium im sensitiven Volumen oder in unmittelbarer Nähe dazu induzieren die

Einzelfehlereffekte durch Hadronen mit einer Mindestenergie von E = 20 MeV. Ein Einzelfehlereffekt äußert sich dadurch, dass ein in einem Register gespeichertes Bit überschrieben wird.

Ein Beispiel für die Funktionsweise eines SEU kann anhand einer statischen Speicherzelle (= SRAM) erklärt werden. Die Architektur eines SRAM wird wird für die sogenannten Register in einem ASIC verwendet. Wie in Abbildung 4.5 zu sehen, besteht das SRAM aus zwei kreuz gekoppelten Invertern, die einen eingelesenen Wert halten können, solange die Versorgungsspannung anliegt.

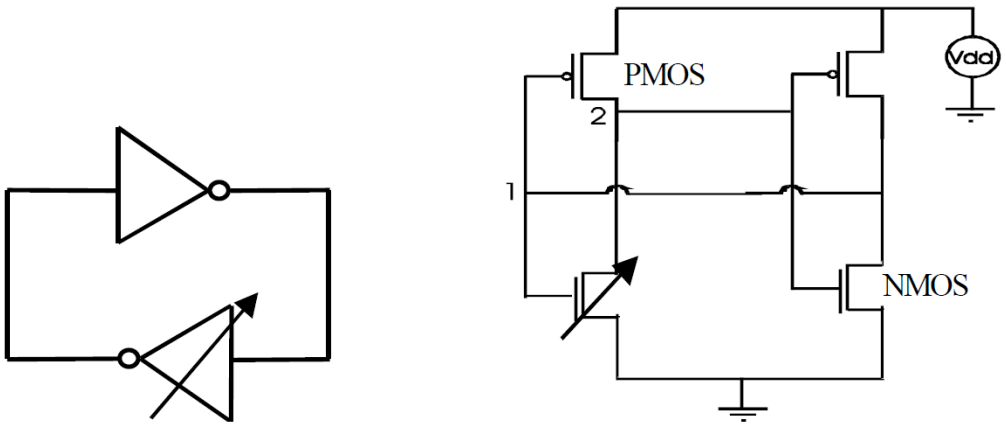


Abbildung 4.5: Ersatzschaltbild für ein SRAM. Für das Beispiel liegt an Knoten 1 0 V und an Knoten 2 die Versorgungsspannung V_{DD} an [19].

Aus der Kernwechselwirkung der Hadronen mit dem Silizium resultieren ionisierende Teilchen, die wiederum Elektronen-Loch-Paare im Silizium freisetzen. Durch das elektrische Feld driften die erzeugten Ladungen zu den Source- und Drain-Anschlüssen des betroffenen Transistors. Wird z.B. der mit einem Pfeil gekennzeichnete NMOS-Transistor aus Abbildung 4.5 getroffen, so werden die freigewordenen Elektronen vom Drain-Anschluss abgesaugt, da dort V_{DD} anliegt. Bei schnellem Absaugen und genügend hoher Ladung kann somit das Potential am Drain auf Null abfallen. Dadurch wird der zweite Inverter geschaltet, der nun die Zustandsänderung am Ausgang des getroffenen Transistors bestärkt. Die gesamte Speicherzelle ist durch die erfahrene Störung nun in einem anderen Zustand.

Die Wahrscheinlichkeit für einen SEU einer Komponente wird mithilfe des Wirkungsquerschnitts σ ausgedrückt, der in cm^2/bit angegeben wird. Dabei gilt für einen Protonenstrahl [20]:

$$\sigma = \frac{N_{SEU}}{F \cdot N_{Register}} \tag{4.4}$$

Die Zahl der aufgetretenen Einzelfehlereffekte (= N_{SEU}) ist abhängig von der Beschaffenheit der jeweiligen Komponente, weshalb jede individuell getestet werden muss. Die Fluenz F gibt die Anzahl der Teilchen pro Fläche integriert über die Zeit an. Die Zahl der im Design enthaltenen Register gibt der Parameter $N_{Register}$ an. Für den in dieser Arbeit verwendeten 130 nm Prozess wird der Wirkungsquerschnitt für Einzelfehlereffekte aus [21] mit $\sigma = 2$ bis $6 \cdot 10^{-14}\ \frac{cm^2}{bit}$ angegeben. Der Wirkungsquerschnitt für die im Rahmen dieser Arbeit entwickelten DCS-Komponenten wurde unter Bestrahlung gemessen, die Resultate sind in den Kapiteln 8.2 und 9.2 zu finden.

Um die Komponente gegen Einzelfehlereffekte abzusichern, können folgende Mechanismen eingesetzt werden:

- die redundante Auslegung von Registern mit angefügtem Majoritätsvoter
- das Einfügen von Prüfsummen
- ein direktes Neuschreiben der Daten in die Register, z.B. mit jedem Takt

Nach der Absicherung einer DCS-Komponente, wie beschrieben, muss diese nach Fertigung ausführlich unter Bestrahlung getestet werden, so dass eine Aussage über die Toleranz gegenüber SEU getroffen werden kann.

Single Event Latchup

Der Effekt des Single Event Latchup (= SEL) kann mithilfe eines Zwei-Transistor-Modells beschrieben werden, die sich parasitär zwischen einer PMOS- und einer NMOS-Komponente bilden. Dabei entsteht eine Thyristorschaltung aus einem parasitären lateralen npn- und einem parasitären vertikalen pnp-Transistor. Die Source-Drain-Gebiete des PMOS stellen hier den Emitter, der n-well die Basis und das p-Substrat den Kollektor des pnp-Transistors dar. Der laterale npn-Transistor wird erzeugt durch das Source-Drain-Gebiet des NMOS (Emitter), das p-Substrat (Basis) und den n-well (Kollektor). Somit sind die Kollektor- und Basis-Leitungen der beiden parasitären Transistoren miteinander verbunden und bilden den oben genannten Thyristor.

Unter normalen Betriebsbedingungen sind beide Transistoren gesperrt. Fließt nun ein lateraler Strom zwischen n-well und p-Substrat, induziert durch z.B. einen Teilchendurchgang, so kommt es zu Strömen entlang des Substrats bzw. des n-wells. Dadurch entsteht ein Spannungsabfall zwischen Basis und Emitter des pnp-Transistors und der resultierende Kollektor-Strom erzeugt einen Spannungsabfall im Basis-Parallelwiderstand des npn-Transistors. Wird auch hier die Basis-Emitter-Spannung ausreichend groß, so leiten nun beide parasitären Transistoren. Es folgt eine positive Rückkopplung zwischen den beiden parasitären Transistoren sowie eine dauerhaft

niederohmige Verbindung zwischen der Versorgungsspannung und dem Massepotential.

Die Folgen sind eine Fehlfunktion der Schaltung im Chip, da die Ausgänge auf einem festen Pegel liegen und nicht mehr auf Änderungen am Eingang reagieren. Im schlimmsten Fall wird der Chip durch einen zu hohen Stromfluss thermisch zerstört.

Im Chipdesign werden verschiedene Techniken angewandt, um das Bilden der parasitären Transistoren einzuschränken. Ein weiterer Schutz bietet eine Strombegrenzung während des Betriebs, so dass eine positive Rückkopplung zwischen den parasitären Transistoren verhindert werden kann [20, 22].

4.2.3 Ionisationseffekte in CMOS-Komponenten

Die Effekte, die sich aufgrund der Energiedeposition im Material ergeben und mit wachsender Dosis steigen, werden TID-Effekte genannt. Diese TID-Effekte werden hauptsächlich durch die Energiedeposition im Siliziumdioxid einer CMOS-Komponente verursacht, weil dort Elektronen-Loch-Paare in der kurzen Zeit keine Möglichkeit zur vollständigen Rekombination haben. Durch die elektrischen Felder, die in der CMOS-Komponente bei anliegenden Spannungen herrschen, driften die durch Ionisation frei gewordenen Elektronen aus der Siliziumdioxidschicht, während die Löcher sich nur innerhalb dieser Schicht bewegen können [20].

Im 130 nm-Prozess sind die empfindlichen Siliziumdioxid-Schichten die Isolationsregionen am Rand der CMOS-Komponenten. Die Gateisolation des Transistors (aus Abbildung 4.1) ist in diesem Prozess bereits so dünn, dass sie nicht zu den gemessenen TID-Effekten beiträgt. Bei größeren Prozessen aus den letzten Jahrzehnten wurde aufgrund der dickeren Isolationsschicht deutlich stärkere Schädigung durch Strahlung beobachtet.

Die Shallow Trench Isolation[5)] (= STI) sorgt für geringere Leckströme zwischen benachbarten PMOS- und NMOS-Transistoren. In der rechten Abbildung 4.6 wird folgender Prozess veranschaulicht: Die deponierte Energie erzeugt eingefangene Löcher in der STI, wodurch ein parasitärer, lateraler Transistor entsteht, der einen Stromfluss zwischen Source und Drain (in Pfeilrichtung) erzeugt. Dieser Effekt kann nur beobachtet werden, wenn der Stromfluss im parasitären Transistor größer ist als der Strom im Haupt-Transistor bei gleicher Gate-Source-Spannung.

Ein zweiter Effekt — die Bildung sogenannter Grenzflächenzustände — tritt durch das Festsetzen der Löcher an der Si/SiO_2-Grenzschicht auf. Dort werden negative Ladungen im Silizium angesammelt, die mit den Löchern rekombinieren können. Die Bildung dieser Grenzflächenzustände ist ein sehr langsam voranschreitender Prozess im Vergleich zu der Ansammlung der positiven Ladungen in der Isolation. Somit

[5)] Bei der sogenannten Grabenisolation werden die elektrisch aktiven Gebiete durch mit Siliziumdioxid gefülle Gräben voneinander isoliert [23].

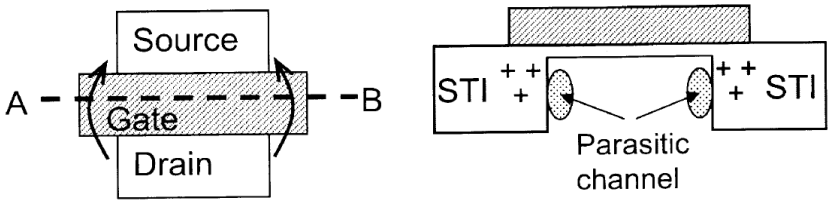


Abbildung 4.6: links: Aufsicht auf einen NMOS-Transistor, rechts: Sicht auf denselben Transistor entlang der A-B-Linie. Die durch Strahlung eingefangenen Löcher in der Shallow Trench Isolation sind durch die Pluszeichen gekennzeichnet. Diese Ladungen modifizieren das elektrische Feld am Transistor-Rand und öffnen einen leitenden Kanal. Leckströme können dann zwischen Drain und Source fließen, wie es mit den Pfeilen im linken Bild angedeutet ist [24].

entsteht ab einer geringen Dosis von einigen 100 kRad zunächst ein erhöhter Stromverbrauch, da durch die entstehenden parasitären, lateralen Transistoren Leckströme fließen. In einem Bereich zwischen 1-6 MRad tragen nun die Grenzflächenzustände signifikant zum Ladungsausgleich an den Transistor-Ecken bei, wodurch die Schwellspannung des parasitären Transistors erhöht und somit die Leckströme reduziert werden. Dies führt insgesamt zur Beobachtung einer Abnahme des Stromverbrauchs der digitalen Schaltung [24].

Eingefangene Löcher in der Siliziumdioxidschicht können durch Erwärmung des Materials wieder rekombinieren und die Fehlstellen somit ausheilen. Die Grenzflächenzustände hingegen können erst ab Temperaturen von ca. 400 °C ausgeheilt werden [20], was für den Detektor im laufenden Betrieb nicht möglich sein wird.

Eine Komponente sollte immer im Betrieb, also mit angelegter Betriebsspannung, auf TID-Effekte untersucht werden, da die angelegten elektrischen Felder die Effekte verstärken und eine Rekombination der ionisierten Teilchen im Oxidbereich des Transistors verhindern.

4.3 Abschätzung der Strahlentoleranz des 130 nm Prozesses

Für die Strahlungsumgebung des zukünftigen HL-LHC wird eine Gesamtdosis von $D_{EoS} = 1000\,\text{MRad}$ und ein Teilchenfluss von $\phi_{EoS} = 3 \cdot 10^8 \frac{Teilchen}{cm^2 \cdot s}$ erwartet, die bei der Entwicklung von elektrischen Komponenten beachtet werden müssen.

Allgemein wird die 130 nm-Technologie gegenüber den TID-Effekten als resistent für die HL-LHC-Strahlungsumgebung angesehen und deshalb auch weitreichend verwendet. Ein Beispiel dazu ist der FE-I4-Chip, der im selben Prozess entwickelt wurde. Mit ihm wurden ausführliche Bestrahlungstests mit Protonen durchgeführt, die

diese Aussage bestätigen. Die Funktion des FE-I4-Chips wurde unter Bestrahlung mit 800 MeV-Protonen bis zu einer Dosis von 200 MRad erfolgreich validiert und die Funktion auch für eine Bestrahlung mit 60 MeV-Protonen bis 800 MRad bestätigt [25].

Da allerdings noch höhere Dosen für das Luminositätsupgrade erwartet werden, sollten alle entwickelten Komponenten auch für solche Dosen mit zusätzlichem Sicherheitsfaktor validiert werden.

Zur Absicherung des Designs gegenüber Einzelfehlereffekten wird die Implementation verschiedener Mechanismen, wie in Kapitel 4.2.2 kurz beschrieben, benötigt. Eine redundante Auslegung der Register ist notwendig, um die Datenspeicherung im Chip abzusichern. Zudem sollten die Daten jeweils so schnell wie möglich erneut eingelesen werden, so dass sich die auf SEU sensitive Zeit auf ein Minimum verkürzt. Für den Datenstrom einer Kommunikation sollten Kodierungsmechanismen eingebaut werden, die einen aufgetretenen Einzelfehler detektieren können.

Es wurden Dual Interlocked Storage Cell Latches (= DICE-Latches) entwickelt, um die Absicherung gegenüber SEU für Detektorkomponenten allgemein zu vereinfachen. Diese speziellen Latches enthalten eine redundant ausgelegte Latch-Struktur, die Einzelfehlereffekte erkennen kann und die Daten durch eine gekreuzte, invertierte Schaltung entsprechend sichert. Die Toleranz gegenüber Einzelfehlereffekten wird dadurch um einen Faktor drei verbessert [26].

Leider konnten die DICE-Latches für die Prototypen des DCS-Controllers nicht verwendet werden. Sie waren zum Zeitpunkt der Layoutimplementation nicht kompatibel zur verwendeten Design-Umgebung und besaßen einen zusätzlichen Anschluss, der nicht verwendet werden konnte. Könnte diese Kompatibilität nachträglich hergestellt werden, so würden diese die für den DCS-Controller verwendete redundante Auslegung der Chipregister mit angefügtem Majoritätsvoter (Triple Modular Redundancy-Mechanismus) ersetzen, der in Kapitel 6.3 noch ausführlicher beschrieben wird.

Kapitel 5

Die Suche nach einer geeigneten DCS-Kommunikation

Aus den Anforderungen für ein zukünftiges Kontrollsystem aus Kapitel 3.1 lassen sich die Anforderungen für eine geeignete DCS-Kommunikation direkt ableiten: DCS-Daten, also Kontrollwerte des Detektors, sowie Steuerbefehle müssen sicher vom Pixeldetektor zum Kontrollraum übertragen werden. Dabei ist eine große Reichweite der Kommunikation erforderlich, da die Übertragungsstrecke ca. 100 m beträgt.

Des Weiteren muss die DCS-Datenübertragung absolut fehlerfrei sein in der hohen Strahlungsumgebung nach dem Phase 2 Upgrade des ATLAS-Detektors. Ausgehend von dem Teilchenfluss für HL-LHC und dem gemessenen SEU-Wirkungsquerschnitt aus Kapitel 9 wird pro Chip ca. ein Bitfehler alle 20 s erwartet. Diese auftretenden Fehler müssen erkannt werden und dürfen die DCS-Kommunikation nicht beeinträchtigen. Für erste Studien wird das Ziel der Robustheit für die DCS-Kommunikation zunächst auf unter einem unentdeckten Fehler in zehn Jahren Betriebszeit festgelegt. Inwiefern dieses Ziel erfüllt werden kann, wird später diskutiert.

Eine weitere Anforderung neben der Reichweite und der Strahlenhärte der DCS-Kommunikation ist die Reduktion der Strahlungslänge des Pixeldetektors. Eine zukünftige DCS-Kommunikation muss mit möglichst wenigen Leitungen auskommen. Eine komplett redundante Datenübertragung ist somit nicht möglich, auch wenn es eine einfache Lösung wäre. Ein Mechanismus, um ein Protokoll mit wenigen Datenleitungen trotzdem ausreichend gegenüber Strahlungseffekten abzusichern, ist die Kodierung. Die Hamming-Kodierung kommt für das zukünftige DCS-System zum Einsatz und wird in Kapitel 5.2 angewendet.

Unter Beachtung der beschriebenen Anforderungen wird in diesem Kapitel eine geeignete DCS-Kommunikation ausgewählt.

5.1 Casting

Die Verwendung von robusten Kommunikations-Protokollen ist z.B. aus der Automobilindustrie bekannt. Dort werden Bussysteme eingesetzt, um die Zahl der Leitungen stark zu reduzieren. Dabei muss die Kommunikation absolut fehlerfrei sein, auch unter extremen Bedingungen. Zum Beispiel sollte der Airbag im Falle eines Unfalls mit einem Fahrzeug noch funktionieren, um die Insassen zu schützen. Der Airbag darf aber unter keinen Umständen fälschlicherweise ausgelöst werden.

Ein aus der Automobilindustrie bekanntes Datenprotokoll ist das Controller Area Network (= CAN). Mit dem CAN-Protokoll kann man über große Reichweiten sehr sicher kommunizieren. Es können bis zu 8 Nutzbytes pro Datenpaket versendet werden, die durch unterschiedliche Mechanismen gegenüber Bitfehlern abgesichert werden. Insgesamt kann das CAN-Protokoll bis zu 15-fach Bündelfehler[1)] und 5-fach Bitfehler detektieren [16]. Das CAN-Protokoll wird im momentanen ATLAS-DCS erfolgreich verwendet und CAN würde somit auch eine Anbindung an das gesamte ATLAS-DCS gewährleisten. Der Umgang mit dem CAN-Bus ist bekannt und die Infrastruktur für seine Verwendung schon vorhanden. Zudem benötigt der CAN-Bus neben den Versorgungsleitungen nur eine twisted-pair-Datenleitung und würde somit den Anforderungen entsprechen.

Eine Implementation des CAN-Controllers auf jeder EoS-Karte stellt sich allerdings als problematisch heraus, da sich aufgrund der großen Reichweite und geringen Leitungszahl auf allen CAN-Konten ein eigener Taktgeber befinden muss. Es ist nicht klar, ob dieser Taktgeber den Anforderungen entsprechend platzsparend und robust gebaut werden kann, so dass er für die CAN-Kommunikation eingesetzt werden kann. Des Weiteren ist das CAN-Protokoll durch seine vielen Fehlererkennungsmechanismen ein sehr komplexes Protokoll und stellt ein größeres digitales Design dar. Das Design würde somit zu viel zusätzliches Material auf der EoS-Karte verursachen.

Das CAN-Protokoll scheint also ein geeigneter Kandidat für eine DCS-Kommunikation zu sein, kann aber aufgrund des separaten Taktgebers und der Komplexität nicht bis in den Pixeldetektor zur EoS-Karte geführt werden.

Eine reduzierte Form dieses Protokolls, wie z.B. das Local Interconnect Network (= LIN), kann für die DCS-Kommunikation nicht genommen werden, da durch die Reduzierung des Designs die meisten Fehlererkennungsmechanismen wegfallen und das Protokoll somit keine ausreichende Absicherung für eine fehlerfreie Kommunikation besitzt. Zudem sind mit dem LIN-Protokoll nur Datenraten bis zu maximal $20\,\frac{kBit}{s}$ möglich, die zu gering für eine DCS-Kommunikation sind.

Als Alternative zu CAN wird von ATLAS-DCS auch die Verwendung des Ethernet-Protokolls in Betracht gezogen. Das Ethernet-Protokoll ist ein weiteres serielles Protokoll mit einer großen Reichweite, hohen Datenrate und einer Absicherung der Da-

[1)] Bündelfehler sind mehrere aufeinanderfolgende Bits, die fälschlich invertiert wurden.

ten durch eine zyklische Prüfsumme. Hier werden pro Datenpaket immer mindestens 64 Byte versendet, worin 46 Nutzbytes enthalten sind. Insgesamt können maximal 1500 Bytes pro Datenpaket versendet werden. Diese hohe Zahl an Daten wird für das Pixel-DCS nicht benötigt, außerdem steigt mit wachsender Zahl an Daten auch die Wahrscheinlichkeit für SEU induzierte Fehler. Durch die Absicherung der zyklischen Redundanzprüfung kann das Ethernet-Protokoll Bündelfehler bis zu 31 Bit und maximal dreifach Bitfehler detektieren [27]. Diese Absicherung liegt unterhalb der Absicherung, die für das CAN-Protokoll implementiert ist. Hauptsächlich aus diesem Grund eignet sich das Ethernet-Protokoll also nicht für die DCS-Kommunikation bei ATLAS-Pixel.

Da die Möglichkeit, einen CAN-Controller auf der EoS-Karte zu realisieren, nicht besteht, müssen weitere Protokolle gefunden werden, die den Anforderungen entsprechen. Ein möglicher Ansatz dazu ist die Verwendung eines einfachen, getakteten, seriellen Protokolls, welches durch Modifikation robust gegenüber SEU gemacht werden kann. Dafür kommen sowohl das Serial Peripheral Interface (= SPI) Protokoll als auch das bidirektionale Inter-Integrated Circuit (= I2C) Protokoll in Frage. Im Vergleich dieser beiden Protokolle wird das I2C-Protokoll bevorzugt, da es weniger Datenleitungen als das SPI-Protokoll benötigt.

Mit einer geeigneten Kodierung innerhalb des I2C-Protokolls sollte es möglich sein, eine sichere DCS-Kommunikation in der hohen Strahlungsumgebung zu gewährleisten. Um die bidirektionale Datenleitung gegenüber Bitfehlern abzusichern, können die Daten auch über einen differentiellen Bus, ähnlich dem CAN-Bus, übertragen werden. Dies erhöht die Reichweite des I2C-Protokolls.

Da im DCS-System für den Control und Feedback Pfad eine Unterbrechung der Datenübertragung an einem Service Point in etwa 20 m Entfernung zu den EoS-Karten vorgesehen ist, ergibt sich für die DCS-Kommunikation die Idee, eine Kombination aus zwei Protokollen (spezielles I2C-Protokoll und CAN-Protokoll) aufzubauen: Auf den EoS-Karten soll sich ein DCS-Chip befinden, der mit dem am Service Point befindlichen DCS-Controller eine modifizierte I2C-Datenübertragung realisiert. Hierfür wird ein sich selbst korrigierendes, Haming-kodiertes I2C-Protokoll verwendet. Das Design ist nicht komplex und kann auf geringer Größe in einem 130 nm Chip realisiert werden. Da es sich um eine Master-Slave-Kommunikation handelt, wird kein externer Taktgeber benötigt. Somit kann das Design auf der EoS-Karte platziert werden. Die Zahl der I2C-Leitungen ist auch gering, es werden eine Paarleitung für die Daten und eine für den Takt benötigt. Der DCS-Controller am Service Point übersetzt die eingehenden Daten in das CAN-Protokoll, welches mehrere Informationen aus der DCS-Chip Kommunikation bündeln kann. Des Weiteren können bis zu 16 DCS-Chips an einem DCS-Controller zusammengefasst werden. Durch dieses System ist auch die Anbindung an das ATLAS-DCS gewährleistet, da dort die Kommunikation mit dem CAN-Protokoll bereits verwendet wird. Der genau Aufbau eines solchen DCS-Netzwerks wurde bereits in Kapitel 3.3 vorgestellt und in Abbildung 3.5 veranschaulicht.

5.2 Das I2C-HC-Protokoll

Für die speziellen Anforderungen an das DCS im Hinblick auf die Strahlungsumgebung, wurde ein sehr sicheres, selbstkorrigierendes Protokoll aus dem I2C-Protokoll heraus entwickelt. Das sogenannte I2C-HC-Protokoll entstand im Rahmen der Diplomarbeit von Kathrin Becker [28] durch eine geeignete Hamming-Kodierung. Zunächst werden kurz die Grudlagen des I2C-Protokolls dargestellt, bevor auf das I2C-HC-Protokoll eingegangen werden kann.

Das I2C-Protokoll

Der I2C-Bus ist ein Zweidraht-Bus mit einer seriellen Datenleitung (SDA) und einer Taktleitung (SCL). Die bidirektionale Datenleitung ist mit Pull-Up-Widerständen versehen, so dass eine logische Null auf dem Bus einen dominanten Zustand bedeutet. Die Kommunikation innerhalb des I2C-Bus erfolgt immer durch ein Master-Slave-Sytem, welches in Abbildung 5.1 dargestellt ist.

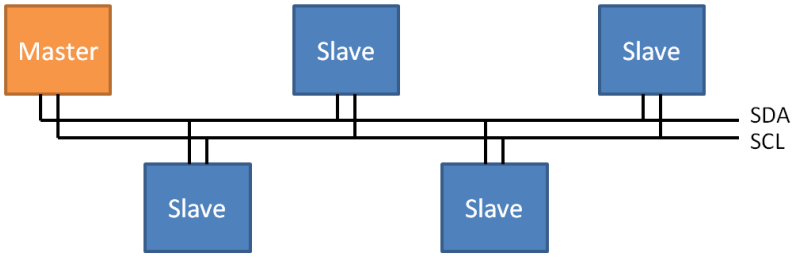


Abbildung 5.1: Das Master-Slave-System des in dieser Arbeit verwendeten I2C-Busses

Im I2C-Protokoll besteht ein Datenpaket jeweils aus 9 Bit und eventuell einer Start- bzw. Stop-Bedingung. Die Kommunikation wird gestartet, wenn der Master ein Startsignal auf den Bus sendet, und ist beendet, sobald dieser eine Stop-Bedingung schickt. Start- und Stop-Bedingung werden nur auf dem High-Pegel des Taktes gegeben, während für die Übertragung der Nutzdaten ein Wechsel auf der Datenleitung immer auf dem Low-Pegel des Taktes erfolgt. In Abbildung 5.2 ist die Übertragung eines Datenpakets mit dem I2C-Protokoll gezeigt.

Der Master sendet immer das erste Datenbyte, in dem die 7-Bit-Slave-Adresse enthalten ist. Im 8. Bit wird festgelegt, ob der Master Daten zum Slave senden wird (Low-Pegel) oder der Slave Daten an den Master senden soll (High-Pegel). Am Ende eines jeden Acht-Bit-Datenpakets wird das Acknowledge-Signal (eine logische Null) vom jeweiligen Kommunikationspartner gesendet, um den korrekten Empfang der Daten zu bestätigen. Wird nach einer Übertragung von 8 Bit kein Acknowledge-Signal empfangen, so beendet der Master den Übertragungsvorgang durch eine Stop-Bedingung. Jeder Slave erhält eine eigene Adresse, so dass zwischen den Slaves keine

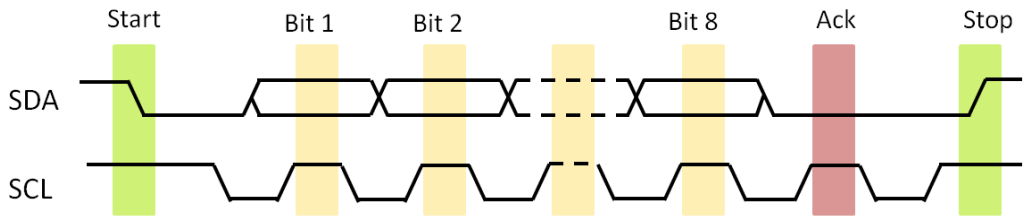


Abbildung 5.2: Die I2C-Kommunikation besteht aus 8-Bit-Datenpaketen mit angefügtem Acknowledge-Signal.

Buskollisionen auftreten können. Der Master kann allerdings mit einem speziellen „General-Call“-Befehl (Slave-Adresse „0x00“) alle Slaves gleichzeitig beschreiben.

Die Datenrate der I2C-Kommunikation liegt im Standardbereich bei 100 $\frac{kBit}{s}$.

Das I2C-HC-Protokoll

Für die DCS-Kommunikation in der Nähe des Pixeldetektors wird statistisch verteilt ein Bitfehler pro Chip alle zwei Sekunden erwartet. Um diese potenziellen Fehlerquellen abzufangen, muss eine ausgeprägte Fehlererkennung und -korrektur für Bitfehler im Protokoll enthalten sein. Dabei sollte das Protokoll mindestens zwei gleichzeitige Bitfehler pro Datenpaket finden können und zusätzlich sollten Einzelbitfehler direkt abgefangen werden. Durch diese Selbstkorrektur soll der Datentransfer minimiert, also die Neusendung von Nachrichten reduziert werden.

Im Rahmen der Diplomarbeit von Kathrin Becker [28] wurde aus dem I2C-Protokoll heraus das I2C-HC-Protokoll entwickelt. Das I2C-HC-Protokoll ist die Erweiterung des standardmäßig 8-Bit breiten I2C-Protokolls um vier Bits mit zyklischer Hamming-Kodierung. Die Implementierung eines zyklischen Hamming-Codes eignet sich sehr gut, da dieser 2-fach Bitfehler erkennen und einzelne Bitfehler automatisch korrigieren kann. Außerdem sind Enkoder und Dekoder einfach zu implementieren, da im Falle der zyklischen Hamming-Kodierung ein n-Bitwort als Polynom n-ten Grades aufgefasst wird und durch Division mit einem Generatorpolynom das sogenannte 4 Bit breite Syndrom berechnet wird, welches den vier Prüfbits entspricht. Die Polynomdivision erfolgt instantan beim Senden bzw. Empfangen des Bitwortes, so dass die 4-Bit-Prüfsumme unmittelbar an die 8-Bit-Nutzdaten angefügt werden kann.

Neben der Erweiterung der Bitzahl auf 13 Bits pro Datenblock ändert sich die Bedeutung des Acknowledge-Signals aus dem I2C-Protokoll, welches nun zusätzlich dafür verwendet wird, einen unkorrigierbaren Fehler im Protokoll anzuzeigen. Schlägt die Kommunikation aufgrund zweier Bitfehler fehl, so wird das Acknowledge verweigert und die Übertragung vom Master gestoppt. Nur in diesem Fall muss die gerade erfolgte Kommunikation wiederholt werden. In Abbildung 5.3 ist der Aufbau

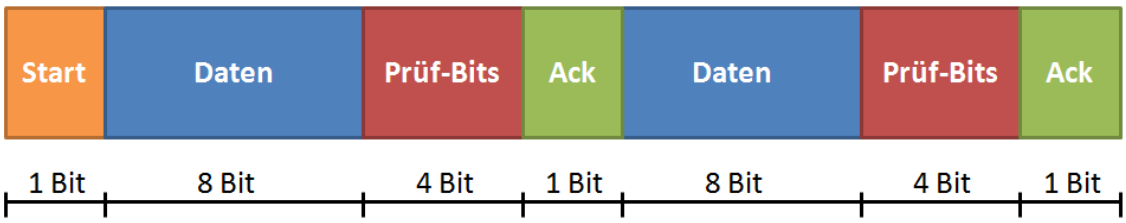


Abbildung 5.3: Datenpakete des I2C-HC-Protokolls, „Ack“ steht hier für das Acknowledge-Signal.

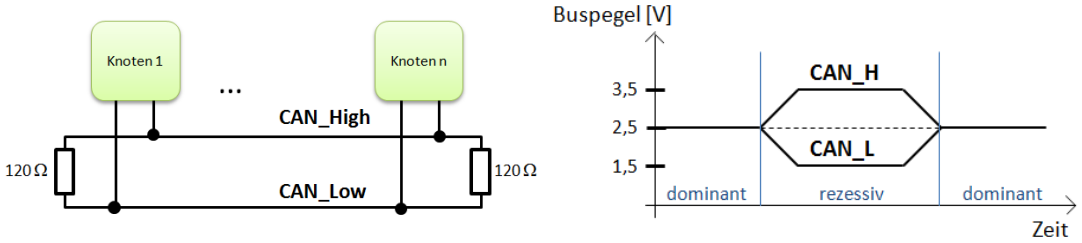
der Datenpakete in einer I2C-HC-Kommunikation schematisch dargestellt. Wie im I2C-Protokoll wird auch hier das höchstwertige Bit zuerst übertragen.

5.3 Die Grundlagen des CAN-Protokolls

Im Folgenden wird ein Überblick über den Aufbau und den wichtigsten Funktionen des CAN-Protokolls gegeben. Dabei werden viele Eigenschaften und Details der CAN-Spezifikationen weggelassen, die in [16] und in [29] genauer nachzulesen sind.

Der CAN-Bus besteht aus einer Paarleitung[2)], der CAN-High und der CAN-Low-Datenleitung, sowie zwei Leitungen für die Versorgung.

In Abbildung 5.4.a ist der Aufbau eines CAN-Bus mit den beiden 120 Ω Abschlusswiderständen[3)] gezeigt. Die CAN-Teilnehmer oder auch CAN-Knoten sind jeweils mit gleichwertiger Bedeutung an den Bus angeschlossen. Jeder CAN-Knoten verfügt über seinen eigenen Taktgeber, deshalb muss sich jeder Knoten selbst mit dem Bus synchronisieren.



(a) (b)

Abbildung 5.4: (a) Der Aufbau eines CAN-Bus mit Terminierung (b) Darstellung der rezessiven und dominanten Buspegel

[2)] Die Paarleitungen sind hier immer twisted-pair Leitungen.

[3)] Ein Abschlusswiderstand vermeidet die Signalverzerrung durch Reflexion am Leitungsende. Für twisted-pair-Leitungen sind 120 Ω passend zum Wellenwiderstand.

In Abbildung 5.4.b ist die Signalgebung auf den Bus gezeigt. Ein Zustand mit der Differenz von etwa 0 V aus CAN-High- und CAN-Low-Signal beschreibt einen dominanten Zustand, also eine logische Null. Der rezessive Zustand ergibt sich aus einer Differenz von etwa 2 V und beschreibt die logische Eins.

Das CAN-Protokoll zeichnet sich durch seine Fehlererkennungsmechanismen aus, also seiner extremen Robustheit gegenüber Bitfehlern. Um solch eine Absicherung zu gewährleisten, sind fest vorgegebene Datenformate notwendig, die eingehalten werden müssen. So gibt es im CAN-Protokoll vier unterschiedliche CAN-Rahmenformate: den Datenrahmen, den Datenanforderungsrahmen, die Fehlerrahmen und den Überlastrahmen.

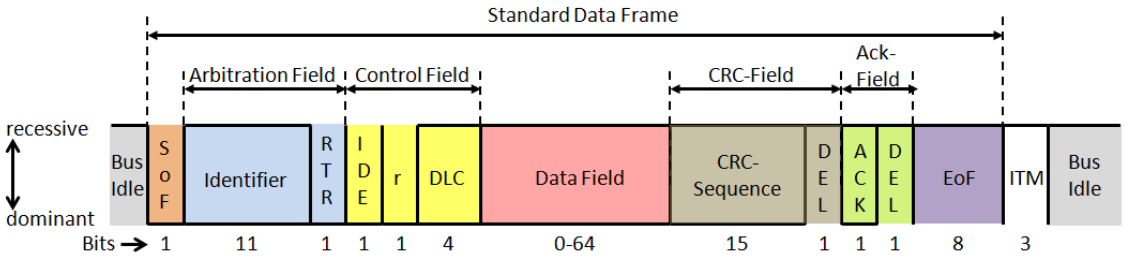


Abbildung 5.5: Der Datenübertragungsrahmen des CAN-Protokolls. Die Bedeutung der Abkürzungen sind im Text zu finden.

Das wichtigste Format ist der CAN-Datenrahmen. Innerhalb dieses Rahmens werden die CAN-Befehle mit 0 bis 64 Bit Länge Nutzdaten in einem Block von mindestens 45 Bit und höchstens 109 Bit gesendet. Ein Datenrahmen besteht, so wie in Abbildung 5.5 gezeigt, zunächst aus einem dominanten Startbit (= SOF), auf welches jeder Knoten seinen Takt synchronisiert. Danach erfolgt der 11-Bit-Identifier sowie das RTR[4)]-Bit. Dieses Bit gibt an, ob ein Datenrahmen (RTR = 1) oder ein Datenanforderungsrahmen (RTR = 0) vorliegt. Das IDE[5)]-Bit wird nur gesetzt, wenn ein Datenrahmen mit einem 29-Bit-Identifier gesendet wird. Da diese Version des CAN-Protokolls in den DCS-Komponenten nicht vorgesehen ist, ist das IDE-Bit stets Null. Auf das „r“-Bit[6)], welches zur Fehlererkennung dient, folgt der Data Length Code (= DLC). In diesen vier Bits wird die Länge der im Folgenden gesendeten Nutzdaten angegeben. Der DLC darf nur Werte von 0 bis 8 annehmen. Darauffolgend werden die Nutzdaten des Datenfeldes gesendet, diese werden mit der Prüfsumme (dem CRC-Feld) abgeschlossen. Im Anschluss an das CRC-Feld wird ein Begrenzungsbit (= DEL) gesetzt, auf welches das Acknowledge-Bit und ein weiteres Begrenzungsbit folgen. Der Datenrahmen wird mit 8 rezessiven Bits (dem Rahmenende = EoF) abgeschlossen. ITM steht in Abbildung 5.5 für den Rahmenzwischenraum, der aus 3

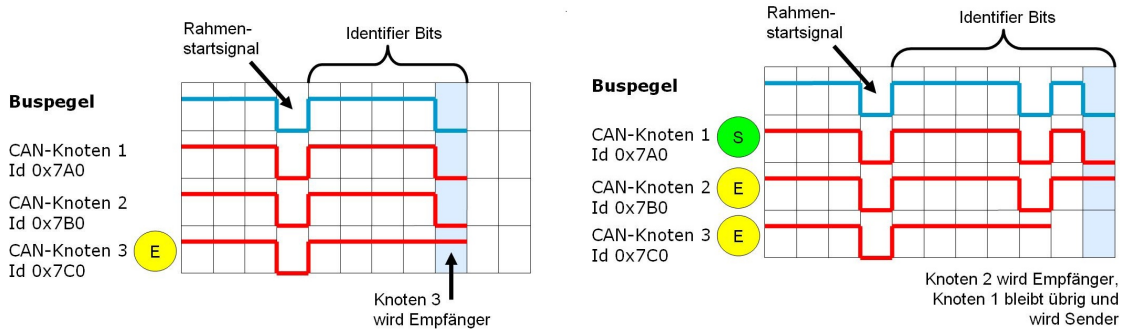
[4)] Remote Transmission Request
[5)] Identifier-Extension
[6)] Reserved Bit

rezessiven Bits besteht, bevor der CAN-Bus wieder in den betriebsbereiten Zustand gehen darf.

5.3.1 Die bitweise Arbitrierung

Da jeder CAN-Knoten gleichberechtigt Nachrichten senden darf, muss eine Regelung eingeführt werden, um Buskollisionen zu vermeiden. Durch die bitweise Arbitrierung ist festgelegt, welcher Knoten in den Empfangszustand geht und welcher Knoten eine Nachricht senden darf.

Der Identifier der CAN-Nachricht dient diesem Mechanismus. Nach dem Abschluss einer CAN-Datenübertragung kann jeder Knoten, der etwas zu senden hat, ein dominantes Startbit (= SOF) auf den Bus geben. Darauf folgt der Sendevorgang des Identifiers. Sobald ein Knoten auf dem Bus ein dominantes Bit sieht, obwohl dieser ein rezessives Bit auf dem Bus gesendet hat, muss dieser Knoten in den Empfangszustand gehen, wie Knoten 3 aus Abbildung 5.6.a.



(a) (b)

Abbildung 5.6: Die bitweise Arbitrierung: 3 Knoten senden zum Rahmenstart gleichzeitig ihren Identifier auf den Bus. (a) Knoten 3 verliert die Arbitrierung zuerst und wird Empfänger. (b) Auch Knoten 2 verliert die Arbitrierung, somit wird Knoten 1 zum Sender.

Der Arbitrierungsvorgang wiederholt sich so lange, bis spätestens am letzten Identifier-Bit ein einziger Knoten als Sender übrig bleibt (siehe Abbildung 5.6.b). Aufgrund dieses Mechanismus wird durch die Identifier eine Prioritätenreihenfolge festgelegt, da der niederwertigste Identifier stets die höchste Priorität besitzt.

5.3.2 Die Synchronisation der CAN-Bus-Teilnehmer

Im CAN-Protokoll erfolgt immer zum Start eines Übertragungsrahmens eine harte Synchronisation[7)] zwischen den Busteilnehmern zum Sender.

Je nach Bauweise des Taktgebers und dessen Fehlertoleranz kann es sein, dass eine Nachsynchronisation[8)] der Busteilnehmer mit dem gesendeten Datenstrom notwendig wird. Für Quarzoszillatoren, wie sie in den Aufbauten dieser Arbeit verwendet wurden, ist keine Nachsynchronisation erforderlich, da diese einen sehr präzisen Takt mit weniger als 0,01% Fehler erzeugen. Für strahlenhart gebaute Taktgeber wiederum, wie z.B. einen RC-Oszillator mit über 1% Fehlerrate, ist eine Nachsynchronisation dringend erforderlich.

Die Möglichkeit zur Nachsynchronisation ist nur gegeben, wenn sich der Signalpegel ändert. Dem CAN-Protokoll liegt die NRZ[9)] Bitcodierung zugrunde, bei der der Signalpegel über die Bitzeit konstant gehalten wird. Das bedeutet, dass für mehrere aufeinanderfolgende gleichwertige Bits keine Flanken für die Nachsynchronisation zur Verfügung stehen. Aus diesem Grund wird beim CAN Protokoll nach jeweils fünf aufeinander folgenden, gleichwertigen Bits ein inverses Bit in den Datenstrom eingefügt. Somit kann spätestens beim sechsten übertragenen Bit eine Nachsynchronisation erfolgen. Auf das sogenannte 'Bit-Stuffing' wird in Abschnitt 5.3.3 noch einmal eingegangen.

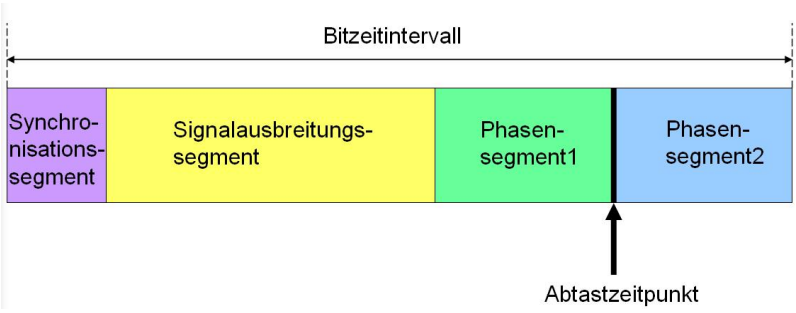


Abbildung 5.7: Aufteilung des Bitzeitintervalls für die Nachsynchonisation

In Abbildung 5.7 ist das Prinzip der Nachsynchronisation graphisch dargestellt. Im Synchronisationssegment ändert sich der Bitpegel auf dem Bus. Über das Signalausbreitungssegment werden die unterschiedlichen Signallaufzeiten innerhalb des Bussystems abgewartet. Dieses Segment ist doppelt so lang, wie die Summe aus längster Bus-Signallaufzeit und den elektrischen Schaltzeiten zwischen Sender und Empfänger. Die beiden Phasensegmente vor und hinter dem Abtastzeitpunkt sind Zeitpuffer

[7)] Eine harte Synchronisation bedeutet, dass sich alle Busteilnehmer auf das dominante Startbit synchronisieren.

[8)] Beim CAN-Protokoll entspricht das einer Synchronisation auf jede fallende Flanke im Datenstrom.

[9)] **N**on **R**eturn **T**o **Z**ero

für die Nachsynchronisation des Abtastzeitpunktes. Ist zum Beispiel der Taktgeber des Senders schneller als der des Empfängers, so muss das Phasensegment 1 verkürzt werden. Wäre der Oszillator des Senders langsamer als der des Empfängers, müsste das Phasensegment 1 verlängert werden. Die Länge der einzelnen Zeitsegmente eines Bitintervalls sind Vielfache der Frequenz des CAN-Knotens und werden in Zeitquanten angegeben.

Für den in dieser Arbeit entwickelten DCS-Controller wurden die Bitzeitintervalle folgendermaßen eingeteilt, wobei der Taktgeber von 4 MHz auf 250 kHz 16-fach heruntergeteilt wird, so dass 16 Zeitquanten pro Bitzeitintervall entstehen:

- **Synchronisationssegment:** 1 Zeitquant (entspricht $\frac{1}{4000000} = 0{,}25\,\mu s$)
- **Signalausbreitungssegment:** 6 Zeitquanten (es werden etwa 1000 ns benötigt)
- **Phasensegment 1:** 4 Zeitquanten
- **Phasensegment 2:** 5 Zeitquanten

5.3.3 Fehlererkennungsmechanismen im CAN-Protokoll

Es gibt fünf verschiedene Fehlererkennungsmechanismen im CAN-Protokoll, von denen einige nur vom Empfänger bzw. nur vom Sender angewendet werden. Erkennt ein Sender oder ein Empfänger einen Fehlerzustand, so sendet er sofort einen Fehlerrahmen, wie er schematisch in Abbildung 5.8 gezeigt ist.

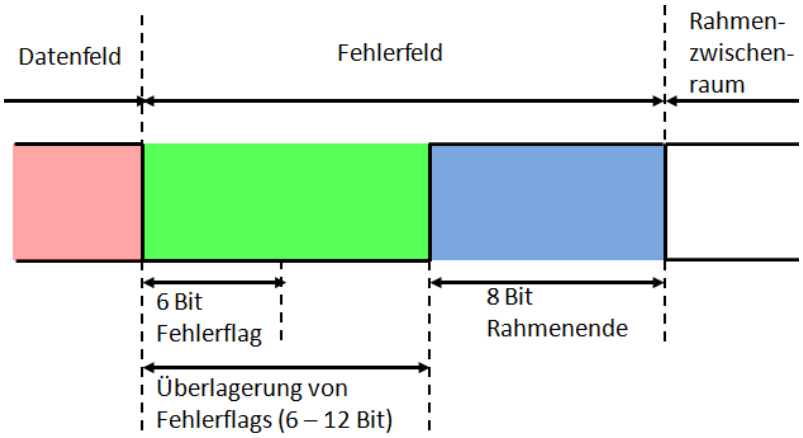


Abbildung 5.8: Der CAN-Fehlerrahmen

Ein Fehlerrahmen besteht aus einem Fehlerflag und darauffolgend aus 8 rezessiven Bits zur Erkennung des Rahmenendes. Das Fehlerflag ist aus mindestens 6 dominanten Bits aufgebaut. Diese Bits sind nur dominant, wenn der Knoten auch die

Berechtigung hat, ein Fehlersignal auf dem Bus anzuzeigen. Hat ein Knoten zu häufig Fehler empfangen, so ist davon auszugehen, dass der Knoten selbst fehlerhaft ist. Solche Knoten schalten sich selbst in einen passiven Zustand und dürfen nur noch Fehlerflags aus rezessiven Bits senden.

Tritt der empfangene Fehler im CRC-Feld auf, wird mit dem Senden des Fehlerrahmens bis nach dem Acknowledge-Bit abgewartet. Somit haben die anderen Busteilnehmer die Möglichkeit, ein positives Acknowledge-Signal auf den CAN-Bus zu geben.

Bitüberwachung

Die Bitüberwachung findet beim Sender statt. Während des Sendevorgangs überwacht der Sender permanent, ob der von ihm gesendete Bitpegel auch auf dem Bus anliegt. Nur im Arbitrierungsfeld, in dem sich Sende- bzw. Empfangszustand entscheiden, und im Acknowledge-Feld gilt die Ungleichheit zwischen gesendetem rezessiven und auf dem Bus anliegendem dominanten Bit nicht als Fehler.

Überwachung des Rahmenformats

In Abbildung 5.5 ist gezeigt, dass bestimmte Bits im CAN-Datenrahmen festgelegt sind. Dazu gehören 'r', 'DEL' und das EoF-Signal. Stellen Sender oder Empfänger ein dominantes Bit an diesen Stellen im Rahmen fest, so liegt ein Rahmenformatfehler vor.

Die zyklische Prüfsumme

Die zyklische Prüfsumme (= CRC wg. **Cyclic R**edundancy **C**heck) ermöglicht dem Empfänger, eine fehlerhafte Nachricht durch Bitfehler zu identifizieren. Dazu werden vom Sender die Bits vom Rahmenstart bis zum Ende des Datenfeldes mithilfe von Polynomdivision durch ein festes Generatorpolynom geteilt [16] und das Ergebnis wird im CRC-Feld mit 15 Bits gesendet. Der Empfänger vergleicht die gesendete Checksumme mit der selbst berechneten und kann bei einem eventuellen Unterschied bis zu fünf gleichzeitige Bitfehler und Bündelfehler bis zu 15 Bit erkennen. Tritt solch ein Fehler auf, so wird das Acknowledge-Signal verweigert und nach dem Acknowledge-Feld ein Fehlerrahmen gesendet.

Überwachung des Acknowledge-Signals

Im Acknowledge-Feld erwartet der Sender ein dominantes Acknowledge-Bit und ein rezessives Acknowledge-Begrenzungsbit. Ist das Acknowledge-Bit rezessiv auf dem

Bus, so wurde entweder die Nachricht fehlerhaft übertragen oder es existiert kein weiterer Teilnehmer am Bus.

Überwachung des Bit-Stuffings

Die zur Nachsynchronisation eingeführten Stuffbits werden vom Empfänger überwacht. Wie in Abschnitt 5.3.2 beschrieben, muss nach fünf gleichwertigen aufeinanderfolgenden Bits ein inverses Stuffbit in den Datenstrom eingefügt werden. Sollten 6 gleiche Bits in Folge empfangen werden, so liegt ein Fehlerzustand vor. Die Überwachung des Bit-Stuffings dient auch dazu, einen Fehlerrahmen zu erkennen, bei dem mindestens 6 dominante Bits in Folge gesendet werden und somit die Bit-Stuffing-Regel verletzen.

5.4 Die erwartete Fehlerrate für die DCS-Kommunikation

Die beiden Protokolle CAN und I2C-HC wurden derart ausgewählt bzw. entwickelt, dass sie in der Strahlungsumgebung nach dem Phase-II-Upgrade keinen fehlerhaften DCS-Befehl erzeugen. Durch Bitfehler in der DCS-Kommunikation könnte ein DCS-Befehl derart verändert werden, dass dieser zwar ein gültiger Befehl bleibt, aber eine andere Information enthält. So könnten im schlimmsten Fall fälschlich Teile des Pixeldetektors abgeschaltet werden, was unbedingt vermieden werden sollte.

Als Grenze der Fehlerrate wird im Folgenden für beide Protokolle weniger als ein Fehler in der gesamten HL-LHC-Betriebszeit angenommen.

Die Wahrscheinlichkeit, dass ein Bitfehler innerhalb der kritischen Zeit $t_{kritisch}$ auftritt, berechnet sich aus folgender Formel:

$$P_{SEU} = \phi_{EoS} \cdot \sigma_{130nm} \cdot N_{Register} \cdot t_{kritisch} \tag{5.1}$$

Der in der Umgebung der EoS-Karten herrschende Teilchenfluss ϕ_{EoS} ist aus Gleichung 4.2 bekannt. Der gemessene Wirkungsquerschnitt für DCS-Chip und DCS-Controller σ_{130nm} ist in Kapitel 9.2.1 und Gleichung 9.1 zu finden.

Die Zahl der Register für den CAN-Knoten liegt unterhalb der Gesamtzahl der Register des DCS-Controllers, da im Controller neben dem CAN-Knoten auch noch weitere Komponenten implementiert sind. Deshalb wird die Zahl der CAN-Register etwas geringer mit

$$N_{CANRegister} = 1300 \tag{5.2}$$

abgeschätzt. Für die Berechnung der Fehlerrate des I2C-HC-Kommunikation wird für die Zahl der Register laut [28]

$$N_{I2C-HCRegister} = 20 \tag{5.3}$$

angenommen.

Im Zeitraum, in dem die Nachricht im Chip prozessiert wird, aber auch für die Dauer des Sendens müssen die Register den Wert verlässlich abspeichern, damit kein Protokollfehler auftreten kann. Ein Bitfehler innerhalb dieser kritischen Zeit $t_{kritisch}$ wird im Register gehalten und verfälscht somit die tatsächliche Nachricht.

Für den CAN-Bus ist eine Datenrate von 250 kBit/s vorgesehen und es werden durchschnittlich für jede CAN-Kommunikation 100 Bits gesendet. Daraus ergibt sich die kritische Zeit $t_{kritischCAN}$ von:

$$t_{kritischCAN} = \frac{100\,Bits}{250000\,Bits/s} = 4 \cdot 10^{-4}\,\mathrm{s} \tag{5.4}$$

Für das I2C-HC-Protokoll ist die kritische Zeit wegen der geringeren Zahl an Daten pro Paket (= 13 Bit) kleiner, obwohl für die I2C-HC-Kommunikation eine langsamere Datenrate von 100 kHz gilt. Die kritische Zeit für das I2C-HC-Protokoll beträgt dann:

$$t_{kritischI2C-HC} = \frac{13\,Bits}{100000\,Bits/s} = 1,3 \cdot 10^{-4}\,\mathrm{s} \tag{5.5}$$

Daraus ergeben sich die folgenden Bitfehler-Wahrscheinlichkeiten innerhalb der kritischen Zeit zu:

$$P_{SEUCAN} = 8,75 \cdot 10^{-6} \qquad \text{für die CAN-Kommunikation} \tag{5.6}$$

$$P_{SEUI2C-HC} = 4,38 \cdot 10^{-8} \qquad \text{für die I2C-HC-Kommunikation} \tag{5.7}$$

Daraus lassen sich die Wahrscheinlichkeiten für einen 6-fach-Bitfehler bei CAN bzw. einen 3-fach Bitfehler für den I2C-HC-Bus berechnen. Integriert über alle Datenübertragungen während der HL-LHC-Betriebszeit und allen Chips, die im DCS-Netzwerk eingesetzt werden, lässt sich daraus eine Aussage über die Wahrscheinlichkeit der unentdeckten Fehler machen.

Es wird angenommen, dass im späteren Betrieb eine maximale DCS-Datenrate von 10 Bit/s während der gesamten Laufzeit vorliegen wird. Das ist eine obere Abschätzung der durchschnittlichen Datenrate, denn die Größenordnung der DCS-Kommunikation ist abhängig vom Betrieb des Detektors und somit zeitlich unterschiedlich. Die Zahl der DCS-Datenübertragungen in der gesamten Laufzeit beträgt:

$$N_{Daten} = 10^9 \, \frac{Bit}{10\,a} \tag{5.8}$$

Die Zahl der DCS-Chips und DCS-Controller ist in Kapitel 3.4 angegeben und beträgt für die DCS-Chips $N_{DCS-Chips} = 716$ und für die DCS-Controller maximal $N_{DCS-Controller} = 90$.

Für das CAN-Protokoll verursachen erst 6 unabhängige, gleichzeitig auftretende Bitfehler einen nicht detektierbaren Fehler. Aus der Wahrscheinlichkeit für einen einzelnen Bitfehler aus Gleichung 5.6 lässt sich die erwartete Fehlerrate für die DCS-CAN-Kommunikation berechnen zu:

$$P_{CAN} = P^6_{SEUCAN} \times N_{Daten} \times N_{DCS-Controller} = 4,04 \cdot 10^{-20} \, \frac{Bit}{10\,a} \tag{5.9}$$

Beim I2C-HC-Protokoll erzeugen drei unabhängige Bitfehler einen nicht detektierbaren Fehler im Protokoll. Die erwartete Fehlerrate für die I2C-HC-Kommunikation ergibt sich somit aus:

$$P_{I2C-HC} = P^3_{SEUI2C-HC} \times N_{Daten} \times N_{DCS-Chips} = 6,01 \cdot 10^{-11} \, \frac{Bit}{10\,a} \tag{5.10}$$

Im gesamten Kapitel wurde argumentiert, dass eine Kombination aus CAN- und I2C-HC-Protokoll die Anforderungen an eine stabile DCS-Kommunikation über eine große Reichweite und Strahlungsumgebung erfüllt. In den erwarteten Fehlerraten der Protokolle spiegelt sich dieses Verhalten wider. Aus den beiden Fehlerraten in Gleichung 5.9 und 5.10 geht hervor, dass die Zahl der auftretenden Fehler, die von den Protokollen nicht erkannt werden können, unterhalb der Grenze von einem Fehler in 10 Jahren Betriebszeit des HL-LHC liegt.

Kapitel 6

Digitales Chipdesign für das Detektorkontrollsystem

Um den Anforderungen des Detektorkontrollsystems zu entsprechen und eine strahlenharte Komponente zu entwickeln, die die DCS-Daten mehrerer DCS-Chips zusammenfassen und über ca. 100 m Entfernung fehlerfrei übertragen kann, wurde das CAN-Protokoll ausgewählt, welches in einem 130 nm Prozess mit zusätzlicher Absicherung der Datenspeicherung implementiert werden sollte.

In Kapitel 5 wurde das CAN-Protokoll als geeigneter Kandidat für die DCS-Kommunikation zum Controller ausgewählt, da es durch seine ausgeprägte Fehlererkennung in der HL-LHC Strahlungsumgebung die DCS-Daten fehlerfrei übertragen kann.

6.1 Der digitale Entwurfsprozess

Im Unterschied zum analogen Chipdesign wird beim digitalen Chipdesign die logische Schaltung mithilfe einer Hardware-Programmiersprache (HDL[1)]) beschrieben. Das Design der in dieser Arbeit vorgestellten Prototypen basiert jeweils auf der Hardware-Programmiersprache Verilog. In Abbildung 6.1 ist der digitale Entwurfsprozess dargestellt, der die Entstehung des Designs von den Spezifikationen bis hin zur Submission veranschaulicht.

Sind die Spezifikationen einer Problemstellung bekannt und wurde das Design daraufhin in Verilog programmiert, wird ein erster Design-Test vorgenommen. Die Verifikation der Funktionsweise des vorliegenden Designs kann sowohl auf Hardware-Ebene erfolgen, als auch durch geeignete Simulation mit Programmen wie NCSim von Cadence[2)].

[1)] **H**ardware **D**escription **L**anguage

[2)] Cadence Design Systems, Inc, San Jose, USA. Die Firma Cadence stellt ein Programmpaket zur Verfügung, mit dem ASICS bis zur Submission entwickelt werden können.

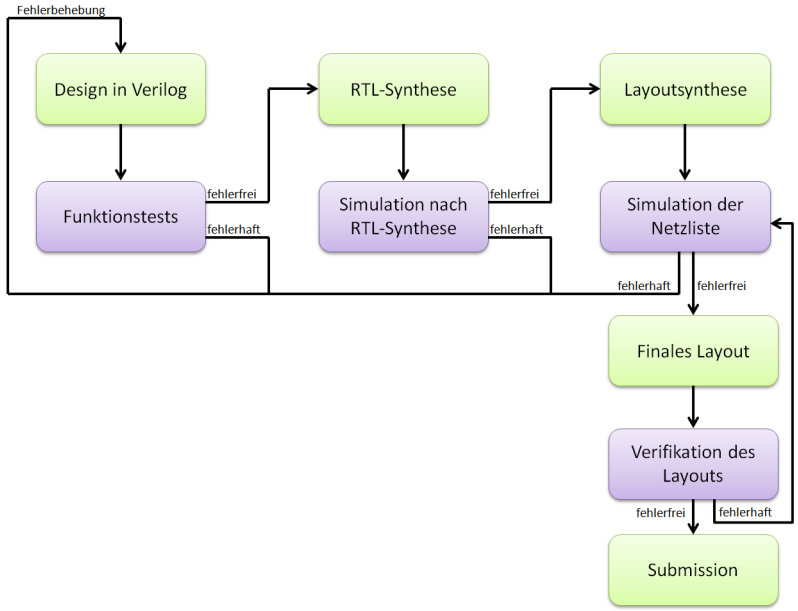


Abbildung 6.1: Der digitale Entwurfsprozess für das Design des DCS-Controllers. Die einzelnen Schritte im Prozess können jeweils nur nach erfolgreichen Tests bzw. Simulationen fortgeführt werden.

Für die Entwicklung des DCS-Controllers wurde der Verilog-Funktionstest auf Hardware-Ebene durchgeführt, da mit dem Aufbau neben der Verifikation des Verilog-Codes auch weitere DCS-Komponenten, wie z.B. der PhysLay-Chip von L. Püllen aus Kapitel 7.3, im Zusammenspiel mit den Prototypen getestet werden konnten. Für den Hardware-Test wurde das Verilog-Design auf einen programmierbaren Logikschaltkreis (XILINX-FPGA[3)] SPARTAN XC3S400A) geladen und die Tests wurden durchgeführt. Für jede Änderung im Code konnte das Programm erneut auf den FPGA geladen werden. Der genaue Aufbau und die durchgeführten Tests werden in Kapitel 6.4 beschrieben.

Entspricht der Verilog-Code den gewünschten Spezifikationen und ist dieser fehlerfrei, so wird er synthetisiert. Die RTL[4)]-Synthese erstellt den Schaltplan des Designs aus logischen Bausteinen und stellt ihre Verbindungen her, so dass das Design durch den Signalfluss zwischen den Registern spezifiziert werden kann. Die RTL-Synthese erfolgt beim Chipdesign mit dem Cadence RTL-Compiler. Nach der Synthese wird eine Simulation mit NCSim durchgeführt, die das abstrakte Design verifizieren muss. Die Signalverzögerungen werden hierbei noch nicht betrachtet. Treten in der Simulation Fehler auf, so muss der Verilog-Code entsprechend korrigiert und alle vorigen Design-Schritte müssen erneut ausgeführt werden.

[3)] **F**ield **P**rogrammable **G**ate **A**rray
[4)] **R**egister **T**ransfer **L**evel

In einem weiteren Schritt erzeugt der Cadence SoC-Encounter das digitale Layout, in welchem die logischen Bausteine nun in Form der Standardzellen aus der Bibliothek des jeweiligen Prozesses im Layout platziert und die physikalischen Verbindungen hergestellt werden. Die im späteren Chip vorliegenden Signallaufzeiten sind nun bekannt. Es folgt eine weitere Simulation mit NCSim von Cadence, in dem vor allem auf Laufzeitprobleme geprüft wird. Das digitale Layout kann nun mit Cadence Virtuoso in das finale Chip-Layout eingebunden werden. Zuletzt wird eine finale Layout-Verifikation durchgeführt, indem zusätzlich die Design-Regeln geprüft werden.

Da der Umgang mit der Software Cadence aufwendig erlernt werden muss, wurden für die in dieser Arbeit entstandenen Prototypen die RTL-Synthese, die Layoutsynthese sowie die letzten beiden Schritte des digitalen Entwurfsprozesses, also die Verbindung des digitalen Layouts mit den Pads und die abschließenden Layout-Prüfungen, von L. Püllen und P. Kind durchgeführt. Im Rahmen dieser Arbeit wurde nur mit dem Cadence-Programm NCSim gearbeitet, in dem alle Verifikationen des Verilog-Designs von der RTL-Synthese bis zur finalen Netzliste durchgeführt wurden.

6.2 Struktur der entwickelten Programme für den DCS-Controller

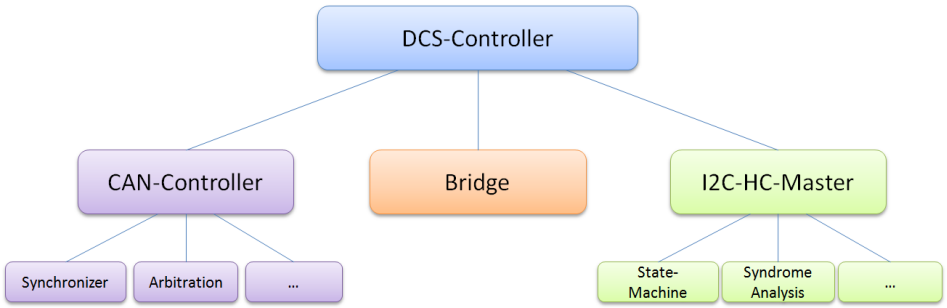


Abbildung 6.2: Die Verilog-Modulstruktur im DCS-Controller

Die Hierarchie eines Verilog-Programmes besteht immer aus einem Topmodul und den dazugehörigen Submodulen, die wiederum Submodule enthalten können. Somit entspricht die Hierarchie der Verilog-Module für den DCS-Controller der in Abbildung 6.2. Das Topmodul „DCS-Controller“ enthält die physikalischen Ein- und Ausgänge des DCS-Controllers und ruft die entsprechenden Submodule, wie den „CAN-Controller“, die „bridge“ und den „I2C-HC-Master“, auf. Dabei hat das Submodul „bridge“ die Funktion, die Nutzdaten aus dem CAN-Protokoll zu extrahieren und für die I2C-HC-Kommunikation bereitzustellen. Das Gleiche gilt auch für

den umgekehrten Kommunikationsweg. Dem „CAN-Controller“ sowie dem „I2C-HC-Master“ gehören wieder mehrere Submodule, wohingegen die „bridge“ kein weiteres Submodul enthält.

Im Rahmen dieser Arbeit wurden der „CAN-Controller“ und die „bridge“ entwickelt und programmiert. Der „I2C-HC-Master“ entstand im Rahmen der Diplomarbeit von Kathrin Becker [28], wurde aber in dieser Arbeit für den zweiten Prototypen korrigiert und erweitert. Da das Design des „I2C-HC-Masters“ in der Diplomarbeit ausführlich beschrieben ist, wird im Folgenden nicht mehr genauer auf dieses Design eingegangen.

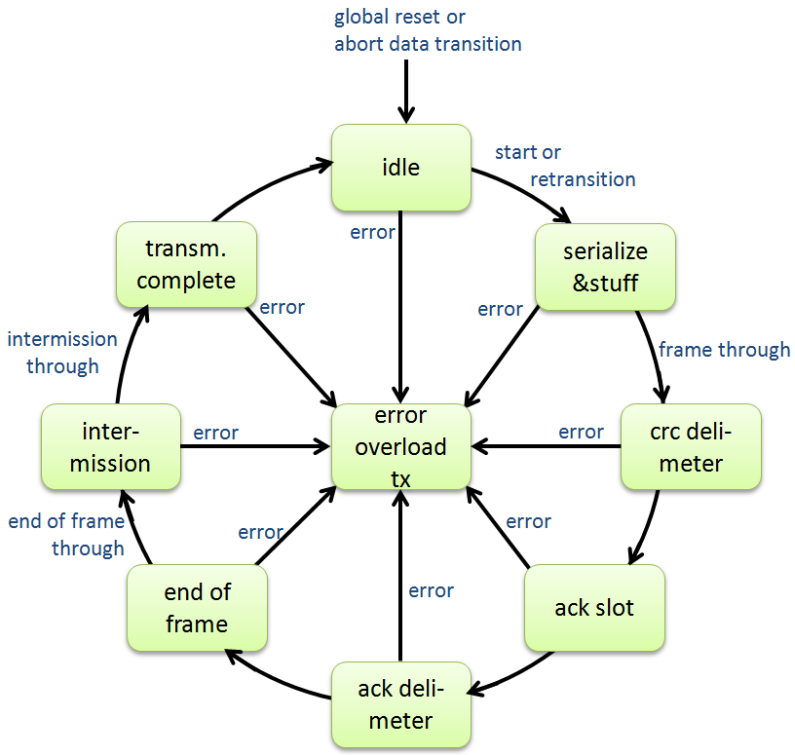


Abbildung 6.3: Schema der Zustandsmaschine für das Serialisieren der CAN-Ausgangsnachricht. In den grünen Boxen wird der jeweilige Zustand angezeigt. Die „Flags“ an den Pfeilen geben an, wann der Übergang in den nächsten Zustand möglich ist.

Die Programmierung mit Verilog basiert im Gegensatz zu den sequentiellen Programmiersprachen wie z.B. C auf dem gleichzeitigen Ablauf aller Prozesse, wie es bei elektronischen Schaltungen üblich ist. Deshalb wurden für die Realisierung des „CAN-Controllers“ in fast allen Submodulen Zustandsmaschinen implementiert. Ein Beispiel einer Zustandsmaschine ist in Abbildung 6.3 für das Serialisieren des zu sendenden CAN-Datenrahmens dargestellt. Die Zustandsmaschine ermöglicht zusammen mit 1-Bit-Variablen, sogenannten „Flags“, einen sequentiellen Ablauf des Programms. Diese entscheiden den Übergang in den nächsten Zustand, wobei diese

Flags sowohl globale als auch lokale Variablen sein können. Die vorliegende Zustandsmaschine bleibt solange im inaktiven Zustand, bis das Flag „start“ oder das Flag zur „retransition“ einer CAN-Nachricht aktiv wird. Erst dann erfolgt der Übergang von dem Zustand „idle“ in den Zustand „serialize&stuff“. Daraufhin werden nacheinander die verschiedenen Zustände durchlaufen, um eine CAN-Nachricht zu serialisieren. Zuerst werden der Stuff-Bereich mit Identifier, Kontrollfeld, Datenfeld und CRC-Prüfsumme serialisiert und die Stuff-Bits eingefügt. Die weiteren Zustände wie CRC-Begrenzungsbit, Acknowledge-Slot, Acknowledge-Begrenzungsbit, Rahmenende und Rahmen-Zwischenraum werden durchlaufen. Die Zustände weisen andere Submodule darauf hin, welcher Teil der Nachricht gerade gesendet wird. So wird im Acknowledge-Slot geprüft, ob das dominante Acknowledge-Signal der anderen CAN-Knoten auf dem Bus anliegt oder nicht. Im Falle eines Fehlers kann von jedem Zustand aus ein Übergang in den Fehlerzustand stattfinden, was in diesem Fall direkt einen CAN-Fehlerrahmen auf dem CAN-Bus hervorruft.

Globale Variablen wie das Reset-Signal, aber auch das Signal zum Abbruch des Sendevorgangs, können den Übergang der Zustandsmaschine in den inaktiven Zustand jederzeit veranlassen.

Die meisten Submodule des Designs sind aus Zustandsmaschinen aufgebaut, die insgesamt über die globalen Variablen miteinander verzahnt sind. Die globalen Variablen setzen Zustandsmaschinen in Gang, stoppen diese oder setzen weitere Variablen, die wiederum andere Zustandsmaschinen beeinflussen. Nur so ist in einer Hardwareprogrammiersprache ein solch komplexes Design möglich.

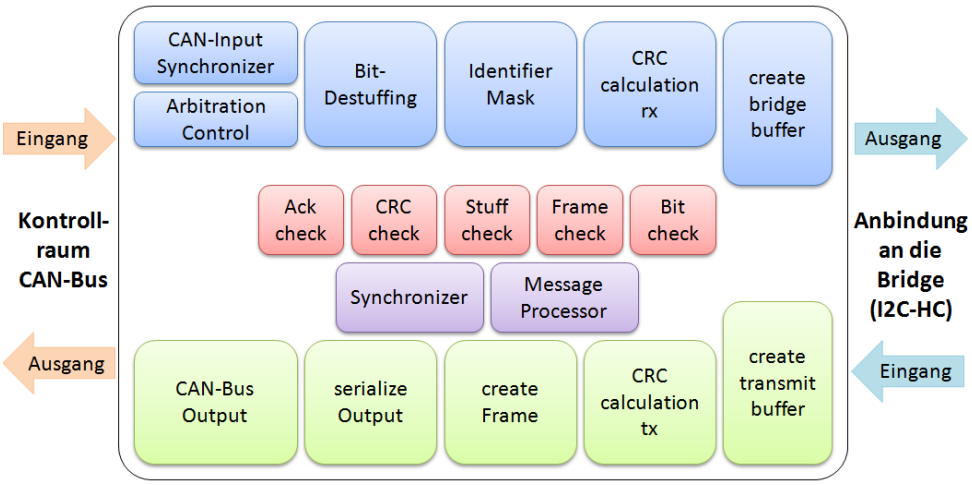


Abbildung 6.4: Die Modulstruktur des „CAN-Controllers“. Die Zustandsmaschine des Submoduls „serializeOutput“ ist in Abbildung 6.3 gezeigt worden.

Einen groben Überblick über die Komplexität der Modulstruktur des „CAN-Controllers“ gibt die Abbildung 6.4. Dort ist der Aufbau der Submodule thematisch sortiert dargestellt. Auf der linken Seite befindet sich die Anbindung zum CAN-

Bus, für die Kommunikation zu den DCS-PCs. Rechts ist die Anbindung des „CAN-Controllers“ zur „Bridge“ und damit zur I2C-HC-Kommunikation zum DCS-Chip zu sehen. Im „CAN-Controller“ sind die thematischen Bereiche wie Sender, Empfänger und die Fehlererkennungsmodule farbig unterschieden. Die blauen Submodule entsprechen dem CAN-Empfänger. Dieser nimmt die CAN-Nachricht vom Bus an und gibt die DCS-Nutzdaten an die „Bridge“ weiter, sofern die entsprechende Adresse des Controllers im Identifier angesprochen wurde. Die Module des CAN-Senders sind grün gefärbt. Der Sender erhält die Nutzdaten der I2C-HC-Kommunikation aus der „Bridge“, erzeugt daraus einen CAN-Rahmen, serialisiert diesen und sendet ihn auf den CAN-Bus. Die Module der fünf Fehlererkennungsmechanismen (siehe 5.3.3) sind rot eingefärbt. In violett sind zwei unabhängige Submodule zu sehen. Im einen ist die CAN-Synchronisation (siehe 5.3.2) realisiert und das andere ist für die Verwaltung der Systemvariablen zuständig.

Der „CAN-Controller“ enthält das Standard CAN-Protokoll. Alle Erweiterungen, die für das DCS-System notwendig sind (Fehlercodes, die genaue Festlegung des Identifiers etc.), wurden in der Bridge eingebaut. Diese strikte Trennung wurde bewusst so vorgenommen, damit der CAN-Controller als eigenständiger Baustein auch für andere Projekte genutzt werden kann, die einen strahlenharten CAN-Knoten benötigen.

Die „Bridge“, die die Verbindung zwischen den beiden Kommunikationsprotokollen herstellt, enthält die DCS-spezifischen Eigenschaften. Sie erhält aus der CAN-Nachricht den Identifier und die Datenbytes, bereitet daraus die I2C-HC-Senderegister vor und startet die Kommunikation im I2C-HC-Master. Bei diesem Vorgang werden die Daten geprüft. Ist der empfangene DCS-Befehl nicht korrekt, so wird die I2C-HC-Kommunikation nicht gestartet und ein Fehlercode generiert, der zurück in den „CAN-Controller“ geschickt wird.

Auch das Auslesen der Fehlerzähler, die Aussagen über Fehler in der CAN-Kommunikation sowie der I2C-HC-Kommunikation machen, wird in der Bridge realisiert. Vom I2C-HC-Master erhält die Bridge die Messgrößen vom Detektor bzw. die Information, ob ein Schreibvorgang erfolgreich durchgeführt wurde. Daraufhin erzeugt die Bridge den DCS konformen Identifier sowie die Datenlänge und gibt diese gemeinsam mit den Daten in bis zu zehn 8-Bit-Registern an den CAN-Controller weiter. Gleichzeitig wird der CAN-Sendevorgang gestartet. Im Falle eines I2C-HC-Übertragungsfehlers wird in der Bridge der entsprechende Fehlercode generiert und auch hier der CAN-Sendevorgang gestartet.

Die in diesem Kapitel beschriebenen Methoden und die Programmstruktur für „CAN-Controller“ und „Bridge“ wurden für das Design beider Prototypen für den DCS-Controller angewendet. Aus diesem Grund wird in der Beschreibung der beiden Prototypen nicht erneut auf die Programmstruktur eingegangen, sondern vielmehr werden ihre Unterschiede aufgezeigt und dabei auch die Tests des Designs, verschiedene Problematiken und Verbesserungen beschrieben.

6.3 Der erste Prototyp-Chip für den DCS-Controller

In diesem Kapitel wird die Entwicklung des ersten Prototypen für den DCS-Controller vorgestellt. Es wird besonders auf die Absicherung der Datenspeicherung gegenüber SEU eingegangen, da diese in den beiden in dieser Arbeit entwickelten Prototypen auf unterschiedlichen Wegen implementiert wurden. Es wird auf die im Prototypen enthaltenen Komponenten eingegangen und begründet, wieso ein weiterer Prototyp entwickelt wurde.

Das Verfahren der Triple Modular Redundancy

Neben der Implementierung des CAN-Protokolls in Verilog musste nun auch ein Weg gefunden werden, um die Register im Chip gegen Single Event Upsets abzusichern. Dafür wurde das Verfahren der Triple Modular Redundancy (= TMR) gewählt.

Bei diesem Verfahren werden die Register im Chip verdreifacht und durch einen angefügten Majoritätsvoter wird der Wert auf den Ausgang des Registers gelegt, der mehrheitlich vertreten ist. Durch dieses Verfahren können Einzelfehler korrigiert werden, erst ein Doppeltreffer der drei TMR-Register führt zu einem falschen Ausgangswert [30]. Werden die drei TMR-Register so weit wie möglich räumlich voneinander getrennt, so sinkt die Wahrscheinlichkeit für einen Doppeltreffer. Bei der Implementierung der TMR-Absicherung ist zu beachten, dass sich hierdurch die Chipfläche fast vervierfacht.

Die Implementierung einer TMR-Absicherung erhöht die Toleranz gegenüber Einzelfehlern ausreichend für die Strahlungsumgebung, die für den Betrieb nach dem Phase-II Upgrade herrschen wird. Das TMR-Verfahren sichert die Register gegenüber Doppelfehlern im redundanten Register ab, deshalb muss die Wahrscheinlichkeit berechnet werden, mit der eines der beiden anderen redundant ausgelegten Register fälschlich durch Strahlung invertiert wird:

$$P_{TMRFehler} = P_{SEUCAN} \cdot \frac{2}{100} \cdot P_{SEUCAN} = 1,53 \cdot 10^{-12} \tag{6.1}$$

Die Wahrscheinlichkeit für einen Einzelfehler in der CAN-Kommunikation P_{SEUCAN} innerhalb der kritischen Zeit in der HL-LHC-Strahlungsumgebung ist aus Gleichung 5.6 bekannt. Der Faktor $\frac{2}{100}$ ergibt sich aus der Forderung, dass für einen echten Fehler eins der beiden redundanten Register (TMR-Register) in der CAN-Nachricht verfälscht werden muss. Dabei wird angenommen, dass eine CAN-Nachricht durchschnittlich aus 100 Bits besteht.

Hier ist sehr gut zu erkennen, dass durch das TMR die Absicherung gegenüber Einzelfehlern um 6 Größenordungen verbessert wird. In Kapitel 9 bestätigt sich

dieses Verhalten, denn dort werden bei einer Teilchenrate von etwa $10^9 \frac{Teilchen}{cm^2 \cdot s}$ Fehler mit dem Design ohne TMR detektiert, während mit TMR-Absicherung kein Fehler gefunden wird.

Das TMR-Verfahren kann im digitalen Chipdesign auf verschiedene Arten implementiert werden. Für das Design des ersten Prototypen wurde das TMR im Verilog-Code realisiert, im zweiten Prototypen wurde das TMR erst während der Synthese im RTL-Compiler eingefügt.

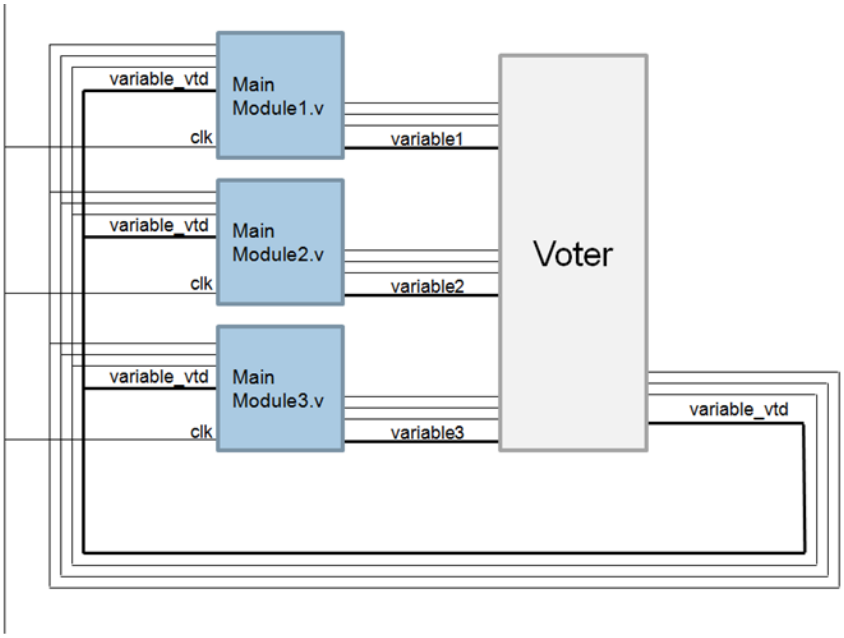


Abbildung 6.5: Funktionsprinzip des TMR im Controller-Design des ersten Prototypen

Die Implementation des TMR erfolgte in diesem Prototypen also innerhalb des Verilog-Codes. Dabei wurden die Register im Verilog-Code verdreifacht, indem die Module verdreifacht wurden, siehe Abbildung 6.5. Die Ausgänge der Verilog-Module werden auf einen Majoritätsvoter geführt, an dessen Ausgang dann die wahren Registerwerte anliegen. Sollte ein SEU den Wert eines Registers beeinträchtigen, so wird dieser Fehler automatisch korrigiert, da die anderen beiden Register durch den Majoritätsvoter den Ausgang bestimmen.

In Abbildung 6.6 ist die Verilog-Modulstruktur nach der TMR-Implementation gezeigt. Auch an dieser Abbildung ist gut zu erkennen, dass das TMR die Chipgröße fast vervierfacht, da alle Register verdreifacht werden und jeweils die Voterlogik aufgebaut werden muss.

Die Verknüpfung der dreifachen Variablen mit den Votereingängen und das Zurücksetzen der Variablen auf sich selbst ist in Verilog eine sehr aufwendige und fehleranfällige Programmierarbeit. Es muss strikt auf die korrekte Zuordnung der

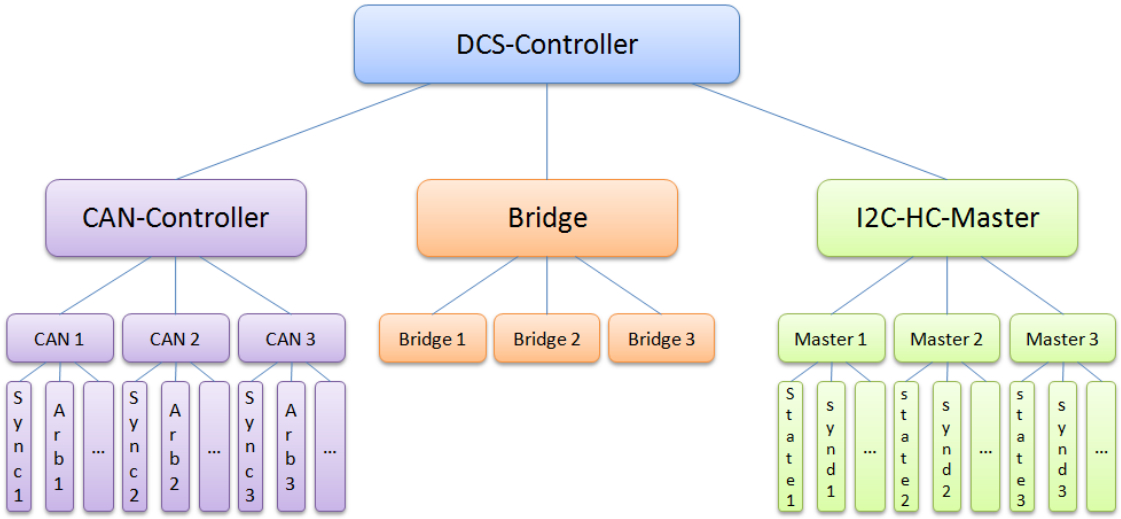


Abbildung 6.6: Die Vermehrung der Submodule aufgrund der TMR-Implementation

Variablen geachtet werden. Zudem müssen spätere Tests entwickelt werden, die die korrekte Durchführung der zugeordneten Variablen verifizieren. Die Zahl der 1519 in diesem Design enthaltenen Register, die durch das TMR-Verfahren abgesichert werden mussten, war viel zu groß, um das Design übersichtlich zu gestalten und es musste für den nächsten Prototypen ein einfacherer und sicherer Weg für die TMR-Implementation gefunden werden.

Der erste Prototyp: CoFee1

Der erste Prototyp-Chip für den **Co**ntrol- und **Fee**dback-Pfad des DCS-Systems beinhaltet sowohl den DCS-Controller als auch den ersten DCS-Chip-Prototypen im 130 nm Design. Der DCS-Chip wurde von Kathrin Becker [28] für dieses Design geschrieben. DCS-Controller und DCS-Chip sind nur aus Kostengründen in einem einzigen Chip zusammengefasst worden; man kann CoFee1 komplett isoliert voneinander als Controller oder als DCS-Chip betreiben. In Abbildung 6.7 ist eine Aufnahme des produzierten CoFee1-Chips zu sehen.

Damit im Rahmen dieser Arbeit die Entwicklung eines DCS-Controllers möglich wurde, musste innerhalb von sechs Monaten das CAN-Protokoll und die Programmierung mit einer Hardwarebeschreibungssprache wie Verilog erlernt und das gesamte Design fertiggestellt werden. Deshalb war die Überlegung, eine reduzierte Form des CAN-Protokolls u.a. durch Einführung einer festen Datenlänge oder durch Weglassen der bitweisen Arbitrierung zu implementieren und dadurch Zeit zu sparen. Dabei musste die 5-Bit-Fehlererkennung von CAN unbedingt beibehalten werden, da auf der Argumentation der Strahlenhärte des CAN-Protokolls dieses Protokoll für die DCS-Kommunikation verwendet wird. Aus dem Grund, dass die Fehlerer-

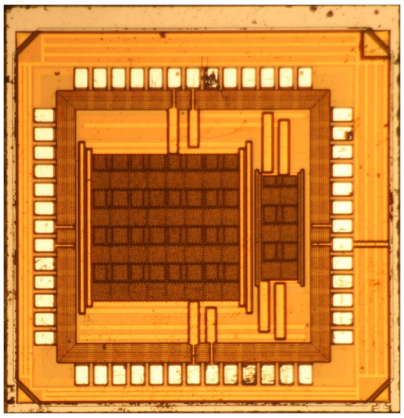


Abbildung 6.7: CoFee1- Der erste Prototyp für den DCS-Controller (linker digitaler Kern im Chiplayout) und den DCS-Chip (rechter digitaler Kern im Chiplayout)

kennung sowieso implementiert werden muss und zudem ein CAN-Controller auch für andere Anwendungsgebiete interessant ist (wie z.B. für andere ATLAS Subdetektoren, DCS-Systeme verschiedener Detektoren, aber auch zur Kommunikation in der Luft- und Raumfahrt), aber nur mit einem Standard-CAN-Protokoll eingesetzt werden kann, wurde das CAN-Protokoll nicht reduziert.

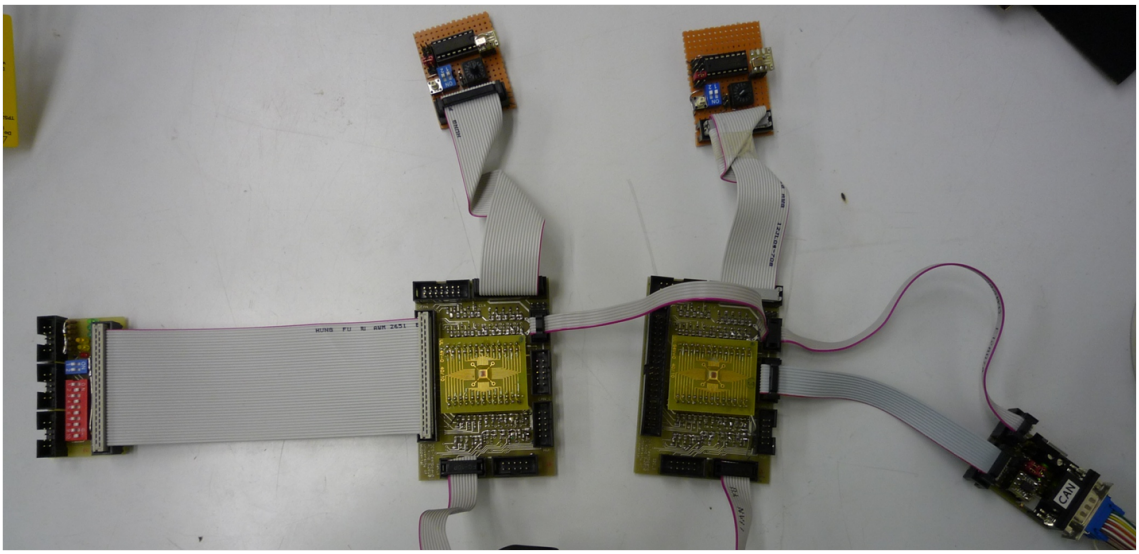


Abbildung 6.8: Aufbau des Control- & Feedback-Pfades aus den ersten Prototypen für DCS-Controller und DCS-Chip

Nach der Lieferung des DCS-Controller-Chips wurde ein Testaufbau aus zwei CoFee1-Chips aufgebaut. In Abbildung 6.8 ist der Aufbau zu sehen, links wird CoFee1 als DCS-Chip verwendet, rechts als DCS-Controller. Der CAN-Bus ist mit einer KVASER-Karte zum PC verbunden und die DCS-Kommunikation wird mit Lab-

View realisiert. Die KVASER-Karte konnte nur verwendet werden, weil ein Standard-CAN-Protokoll im DCS-Controller implementiert wurde. Dieser Aufbau verifiziert die Funktion einer DCS-Kommunikation mit Controller und DCS-Chip. Es wurden mehrere Tests durchgeführt, die zeigen, dass die Chips größtenteils funktionsfähig waren.

Allerdings wurden Fehler im CAN-Protokoll sowie im I2C-HC-Master gefunden. Die wichtigsten sind in Kapitel 6.4.1 beschrieben. Durch die Fehler in den Protokollen konnte nur ein eingeschränkter DCS-Nachrichtenraum genutzt werden, eine zufällige DCS-Kommunikation war nicht möglich. Diese Fehler zeigen, wie wichtig ausführliche Tests beim Entwurf des digitalen Designs sind. Es wurde sehr deutlich, dass viel mehr Zeit für Design-Tests aufgewendet werden muss, als es für diesen Prototypen der Fall war.

Aufgrund dieser Erfahrungen wurde das Design eines zweiten Prototypen beschlossen, in dem die Kommunikation korrigiert und einige neue Funktionen implementiert werden sollten.

6.4 Entwicklung eines weiteren Prototypen für den DCS-Controller

Der zweite Prototyp für den DCS-Controller beinhaltet eine korrigierte Version der Kommunikationsprotokolle, erweiterte Funktionen und eine komplett neue DCS-Befehlsstruktur. Das neue Design ist deutlich ausführlicher getestet als das Design für CoFee1. Dazu wurde ein Aufbau konstruiert, der den DCS-Controller mithilfe eines FPGAs realisiert und CoFee1 als DCS-Chip verwendet.

Nach der Fertigstellung des Designs wurden neben den ausführlichen Funktionstests auch Langzeittests durchgeführt sowie ein DCS-Netzwerk aufgebaut und seine Funktionsweise verifiziert. Zudem wurden ausführliche Laufzeitsimulationen gemacht, um die optimale Funktion des Chips nach der Fertigstellung zu gewährleisten. Der Übersichtlichkeit halber und mit Hinblick auf die Speicherkapazität des verwendeten FPGAs wurde auf der Grundlage des TMR-freien Designs für CoFee1 das Design für CoFee2 entwickelt.

6.4.1 Überarbeitung der Kommunikationsprotokolle

Die Testumgebung für eine fehlerfreie Programmierung

Für die Überarbeitung des DCS-Controllers wurde ein Testsystem entwickelt, um den Verilog-Code nach jeder Änderung ausführlich testen zu können. Dazu wurde der DCS-Controller mithilfe des Spartan-FPGA Evaluation Boards von Xilinx realisiert. Die Verwendung des FPGAs als DCS-Controller ermöglicht die Verifikation des sich

ständig ändernden Codes, da dieser jeweils mit dem aktuellen Design programmiert werden kann.

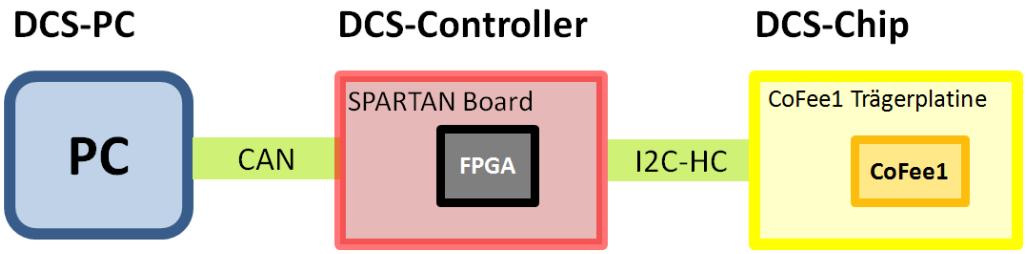


Abbildung 6.9: Schema des Testaufbaus zur Verifikation des CoFee2-Designs

Wie in Abbildung 6.9 zu sehen ist, wurde durch den Aufbau eine komplette DCS-Kommunikationskette bestehend aus DCS-PC, DCS-Controller und DCS-Chip aufgebaut. Entgegen dem variabel programmierbaren FPGA, der den DCS-Controller ersetzt, wird der DCS-Chip durch den ersten Prototyp-Chip CoFee1 realisiert. Die Funktionsweise des DCS-Chips hatte bisher keine Fehler gezeigt und erwies sich als sehr geeignet für die Tests. Außerdem konnte so aus diesem Aufbau durch Vervielfachung der Komponenten ein DCS-Netzwerk aufgebaut werden (siehe dazu 6.4.3).

Um jede strukturelle Änderung am Quellcode direkt überprüfen zu können, wurde ein Schnelltest entwickelt, der mithilfe von LabView realisiert wurde. Eine Liste mit Kommunikationsbefehlen, die verschiedene Bereiche der Kommunikation auf ihre Funktion hin testen, wurde abgearbeitet und die Kommunikation in einem Testbericht dokumentiert. Ein Schnelltest führt die folgenden Schritte durch:

1. Das Auslesen der Fehlerzähler
2. Das Auslesen aller DCS-Chip-Register
3. Das Schreiben aller DCS-Chip-Register
4. Die Provokation der sieben Fehlercodes
5. Der Test auf CAN-Nachrichten mit speziellen Identifiern (wie 0x000 und 0xFFF, in denen maximal viele Stuff-Bits enthalten sind)
6. Der Test auf Stuff-Bits an kritischen Stellen in der CAN-Nachricht (z.B. am Anfang und am Ende des CRC-Feldes)

Mithilfe des Testberichts aus diesem Schnelltest konnte ausgeschlossen werden, dass eine Änderung am Quellcode einen Fehler im Design verursacht. Der Test wurde so gewählt, dass jeder Schritt einen bestimmten Bereich der im DCS-Controller enthaltenen Funktionen abdeckt, so dass insgesamt jeweils das gesamte Design geprüft werden konnte. Es wurde immer die gesamte Liste eines Schnelltests abgearbeitet,

da die meisten Änderungen am Quellcode modulübergreifend und somit für mehrere Teile der Kommunikation durchgeführt wurden. Ein Schnelltest dauerte wenige Minuten, so dass die Durchführung des gesamten Tests jeweils nach einer strukturellen Änderung am Code gerechtfertigt war.

Die Behebung der Fehler im Code des ersten Prototypen

Im Folgenden werden beispielhaft vier ausgewählte Probleme im ersten Prototypen des DCS-Controllers vorgestellt und ihre Behebung skizziert:

- **Die fehlerhafte CAN-Synchronisation:** In CoFee1 erzeugte der CAN-Knoten durch einen Fehler in der CAN-Synchronisation nutzlose CAN-Nachrichten auf den Bus, die im späteren Aufbau die Kommunikationsgeschwindigkeit beeinträchtigen würden. Die fehlerhafte Synchronisation wurde dadurch behoben, dass das Modul für die CAN-Synchronisation komplett umgeschrieben und die CAN-Resynchronisation in diesem Modul nur auf das Wichtigste beschränkt wieder aufgenommen wurde. Für CoFee2 ist nur die harte Synchronisation auf den Rahmenstart und die Nachsynchronisation auf eine fallende Flanke im Empfangszustand auf die Phasensegmente implementiert[5)].

  Die komplette CAN-Resynchronisation aus Kapitel 5.3.2 wurde für diesen Prototypen nicht in das Design eingebaut, da nicht genügend Zeit bis zur nächsten Submission blieb, um die verschiedenen Fälle einwandfrei zu programmieren und ausführlich zu testen.

- **Bit-Stuffing-Problem:**

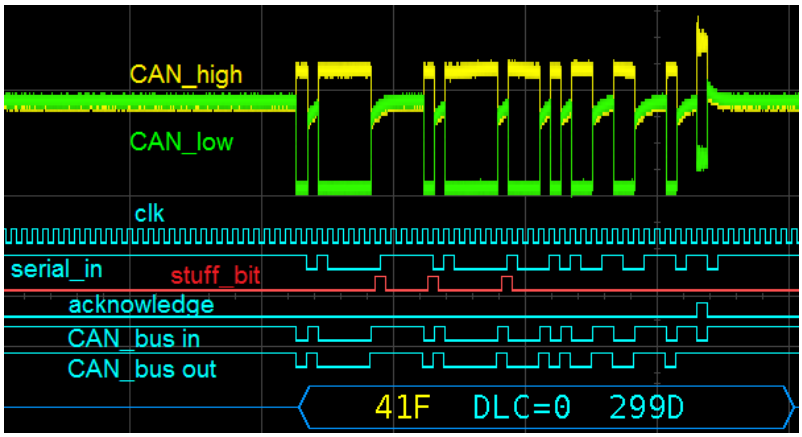


Abbildung 6.10: Spezielle CAN-Nachricht, in deren Identifier auf fünf gleiche Bits fünf invertierte Bits folgen.

[5)] Das ist nur möglich, weil der Taktgeber hier ein Quarzoszillator ist, der aufgrund seiner hohen Frequenzstabilität eine Nachsynchronisation innerhalb der CAN-Nachricht entbehrlich macht.

Durch einen Programmierfehler in einem Zähler wurden im ersten Prototypen teilweise CAN-Nachrichten mit 6 gleichen Bits in Folge erzeugt, was die Bit-Stuffing-Regel verletzt. Diese Nachrichten traten immer dann auf, wenn fünf gleichwertige Bits und darauffolgend fünf invertierte Bits im CAN-Datenrahmen enthalten waren. Nach der Korrektur können mit dem neuen Prototypen auch solche Nachrichten fehlerfrei übertragen werden. Die in Abbildung 6.10 übertragene CAN-Nachricht mit der Datenlänge Null wird vom Empfänger positiv bestätigt. In der 11-Bit Message ID dieser CAN-Nachricht (0x41F = 10000011111) tritt der für CoFee1 problematische Fall auf und wird mit dem neuen Design korrekt übertragen.

- **Zwei CAN-Nachrichten folgen direkt aufeinander:** Es kann vorkommen, dass auf dem CAN-Bus zwei Nachrichten unmittelbar aufeinander folgen. Das geschieht meist, wenn zwei Knoten gleichzeitig senden wollen. Dann verliert ein Knoten die Arbitrierung und sendet seine Nachricht erneut direkt im Anschluss an den Rahmenzwischenraum (= ITM aus Abbildung 5.5) der vorhergehenden Nachricht.

  Das bedeutet, die Empfängerstruktur muss im Zeitraum von drei Bit des ITM alle Register und Zähler zurücksetzen, um eine neue Nachricht in Empfang nehmen zu können. Für CoFee1 wurde während dieses ITM-Zeitraums missverständlich auch auf Protokollfehler geprüft, so dass diese drei Bit nicht zum Zurücksetzen der Register und Zähler verwendet wurden. Dort gab es *kein* Bit Zeit für das Zurücksetzen, so dass direkt anschließende CAN-Nachrichten diverse Fehlerrahmen erzeugten, weil die Zähler nicht korrekt zurückgesetzt und so eingehende CAN-Nachrichten stark missinterpretiert wurden. Dasselbe gilt für den Empfangszustand.

  Für das neue Design des DCS-Controllers werden im Telegrammzwischenraum jetzt alle entsprechenden Variablen für den Empfangs- bzw. Sendezustand zurückgesetzt, so dass CAN-Nachrichten direkt nach 11 Bit hintereinander gesendet bzw. empfangen werden können.

- **Absturz nach Fehlermeldung:** Wird bei der I2C-HC-Kommunikation zum DCS-Chip kein Acknowledge-Signal, also keine positive Bestätigung vom Slave empfangen, so bleibt im ersten Prototypen die State-Machine des I2C-HC-Masters im State „transmission_error“ stehen und der gesamte Chip muss erneut initialisiert werden, damit die Kommunikation wieder aufgebaut werden kann. Behoben wurde das Problem für das neue Design mit der Einführung einer zusätzlichen Bedingung in die State Machine für den I2C-HC-Master. Eine fehlerhafte I2C-HC-Kommunikation wird nun erkannt, ein spezifischer Fehlercode gesendet und die I2C-HC-State-Machine wieder in den Bereitschaftszustand gesetzt.

Nach der Behebung aller gefundenen Fehler im Design konnten zusätzliche Verbesserungen an der DCS-Kommunikation vorgenommen werden. Dazu gehört eine festgelegte DCS-Befehlsstruktur und die Information, ob eine Übertragung erfolgreich war, sowie ein Zwischenspeicher für eingehende DCS-Befehle.

6.4.2 Neue Befehlsstruktur und zusätzliche Funktionen

Eine festgelegte Aufteilung des CAN-Identifiers einer CAN-Nachricht ermöglicht eine direkte Adressierung jedes einzelnen Controllers und DCS-Chips im DCS-Netzwerk. Die 11 Bits des Identifier sind, wie in Tabelle 6.1 zu sehen ist, aufgeteilt worden.

10	9	8	7	6	5	4	3	2	1	0
dominant	Controller-Adresse					Adresse DCS-Chip				Antwortnachricht?

Tabelle 6.1: Die Struktur des Identifiers in einem DCS-Befehl. Das 10. Bit ist in diesem Design immer dominant. Eine Antwortnachricht wird gesendet, wenn das 0. Bit dominant ist.

Da im zukünftigen DCS-System maximal 16 DCS-Chips mit einem Controller verbunden sein werden, wurden 4 Bit des Identifiers für die DCS-Chipadresse reserviert. Für die Adresse der Controller werden die übrigen fünf Bit verwendet. Es werden insgesamt zwar mehr als 31 Controller im DCS-Netzwerk erwartet, diese teilen sich allerdings auf zwei Detektorhälften und somit zwei separate CAN-Busse auf. Diese Aufteilung kann mit jedem Prototypen wieder entsprechend angepasst werden und ist keinesfalls endgültig.

Durch die bitweise Arbitrierung liefert das CAN-Protokoll aber eine Prioritätenreihenfolge der Identifier (der Identifier mit dem niedrigsten Wert hat die höchste Priorität auf dem Bus). Für das DCS-System kann diese Reihenfolge gewährleisten, dass eventuell wichtigere Befehle, wie z.B. das Abschalten eines Detektormoduls, direkt ausgeführt werden und nicht erst nachdem eine Ausleseroutine abgearbeitet wurde. Dazu könnte man den eigentlichen DCS-Steuerbefehl im Identifier festlegen und die Adressierung in den ersten CAN-Datenbytes realisieren. Da aber noch nicht alle DCS-Befehle eines späteres DCS-Systems eindeutig bekannt sind und außerdem durch die Adressierung in den Datenbytes weniger Daten pro Datenrahmen versendet werden können, ist für diesen Prototypen zunächst die Aufteilung des Identifiers wie in Tabelle 6.1 beschrieben vorgesehen.

Es wird nur eine Prioritätenreihenfolge dadurch erzeugt, dass das niederwertigste Bit im Identifer eine Nachricht als Anfrage (0. Bit = rezessiv) oder als Antwort (0. Bit = dominant) definiert. Wie auch schon im Prototypen CoFee1 ist somit gewährleistet, dass eine Antwort auf dem Bus stets einer Anfrage vorgezogen wird. Das 10. Bit wird in diesem Design stets nur dominant als gültiges Bit akzeptiert, kann in zukünftigen Designs aber eine weitere Priorität hinzufügen.

Für eine vollständige Anfrage im DCS-Netzwerk werden in die ersten beiden Datenbytes der CAN-Nachricht drei Informationen mitgesendet: die Unterscheidung zwischen Lese- und Schreibbefehl, das Startregister im DCS-Chip und die Zahl der Register, die ausgelesen bzw. beschrieben werden sollen. Die genaue Aufteilung der Informationen ist in Tabelle 6.2 zu finden.

1. Datenbyte							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
R/W	x	x	x	x	Zahl zu lesender Register		
2. Datenbyte							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Adresse des Start-Registers							

Tabelle 6.2: Neue Befehlsstruktur der ersten beiden Datenbytes eines DCS-Befehls, x markiert don't care Bits.

Auch die Antwortnachricht des DCS-Controllers enthält zusätzliche Informationen: In das 1. Datenbyte wird die Adresse des DCS-Chip-Registers geschrieben, mit dem die Auslese begonnen wurde. Die Anzahl der ausgelesenen Datenbytes gibt zusätzlich Aufschluss darüber, wie viele Register ausgelesen wurden. Es gibt allerdings zwei Spezialfälle, die in dem ersten Byte der Antwortnachricht auftreten können.

1. Für Schreibbefehle an den DCS-Chip wurde eine Antwort definiert, die nach dem erfolgreichen Schreibvorgang gesendet wird. Der erste Prototyp für den DCS-Controller hatte dieses selbst definierte Acknowledge nicht implementiert, so dass das erfolgreiche Ausführen eines Schreibbefehls im Testaufbau immer extern geprüft werden musste. In diesem Prototypen wird nun eine Antwortnachricht generiert, die wie alle Antwortnachrichten den CAN-Identifier um eins reduziert und in das erste Datenbyte ein „0xAC“ schreibt, wenn ein Schreibvorgang erfolgreich ausgeführt wurde.

2. Tritt ein Kommunikationsfehler bei einer CAN-Anfragenachricht auf, so wird dem Benutzer eine Nachricht zurückgesendet, die sich im Identifier wieder auf die Anfrage bezieht (niederwertigstes Datenbyte wird dominant gesendet) und im ersten Datenbyte ein „FF“ stehen hat. Steht im ersten Datenbyte ein „FF“, so folgt im zweiten Datenbyte ein spezieller Fehlercode (momentan 1-7, siehe Tabelle 6.3), der eine Aussage über die Art des aufgetretenen Problems macht. Es kann zwischen Fehlern wie einer falschen Slaveadresse, ungültigen Datenbytelängen und einer Abfrage eines falschen Start-Registers unterschieden werden.

Um einen Eindruck zu vermitteln, wie die DCS-CAN-Kommunikation für diesen Prototyp-Chip aussehen wird, sind einige ausgewählte DCS-Befehle und ihre Antwortnachrichten in Tabelle 6.4 dargestellt.

Fehlercode	Interpretation
01	Slave Adresse ungültig
02	DLC > 7 abgefragt
03	DLC $= 0$ gesendet
04	Startregister > 11
05	Lesebefehl und DLC $\neq 2$
06	Schreibbefehl und DLC < 3
07	Schreibbefehl und Startregister > 11-DLC

Tabelle 6.3: Detaillierte Aufstellung der Fehlercodes und ihrer Interpretation im DCS-Netzwerk

Identifer	DLC	B1	B2	B3	B4	B5-8	Interpretation
Lesebefehl							
0x1FF	2	0x81	0x00	x	x	x	Anfrage der DCS-Chip-ID, DCS-Chip-Adresse: 0x0F, Adresse Controller: 0x0F
0x1FE	3	0x00	0x21	0xF4	x	x	Antwort zu obiger Anfrage: Startregister 0x00, Chip-ID 0x21F4
Schreibbefehl							
0x1FF	4	0x00	0x08	0x60	0x07	x	Schreiben von Register 0x08
0x1FE	1	0xAC	x	x	x	x	Schreibbefehl erfolgreich übertragen

Tabelle 6.4: Exemplarische DCS-Befehle und ihre Antwortnachrichten

Ein Zwischenspeicher für eingehende DCS-Befehle

Neben der Umstrukturierung der DCS-Befehle und ihrer Antwortnachrichten wurde für diesen Prototypen eine weitere wichtige Funktionen implementiert, die es ermöglicht, dass direkt aufeinander folgende DCS-Befehle im Controller gepuffert werden können. Für diesen Prototypen wurde ein Puffer von drei CAN-Nachrichten implementiert. Für die Speicherung der drei Nachrichten stehen insgesamt drei 80-Bit-Puffer zur Verfügung[6)].

[6)] Diese 80-Bit setzen sich zusammen aus 11-Bit Identifier, 4-Bit DLC, einem Bit RTR und den 64-Bit der Daten.

6.4.3 Funktionstests

Ein vollständiger Test des DCS-Controllers wäre das Senden bzw. Empfangen aller möglichen CAN-Nachrichten auf dem Bus und die Auswertung der entsprechenden Reaktion des Controllers. Diese entspricht 2^{100} Befehlen und ist mit den gegebenen Mitteln zeitlich nicht realisierbar. Aus diesem Grund wurde ein Plan mit ausgewählten Tests aufgestellt, die alle Bereiche der DCS-Befehle ausführlich testen sollen:

1. Empfangen bzw. Senden einer zeitlich annehmbaren Zahl von CAN-Nachrichten auf dem Bus
2. Spezielle Prüfung kritischer Stellen im CAN-Protokoll
3. Empfangen bzw. Senden von Zufallsnachrichten unbestimmter Länge und Abstand auf dem CAN-Bus
4. Test der Kommunikation im DCS-Netzwerk
5. Laufzeitsimulationen

Zusätzlich werden die Funktionstests in Sende- und Empfänger-Zustand aufgeteilt. Dies ermöglicht eine strukturierte Erstellung und Auswertung der Tests.

Tests des Empfängers

Die Tests des Empfängers des Controllers wurden weiterhin mit dem Aufbau aus Abbildung 6.9 realisiert, der auch schon für das Design und die Schnelltests verwendet wurde. Die Ansteuerung des Controllers erfolgte auch hier mithilfe von LabView.

Zuerst wurden alle Bit-Kombinationen spezieller Segmente einer CAN-Nachricht (wie dem Identifier und den jeweiligen Datenbytes) auf den Bus gegeben und somit getestet, ob sämtliche Nachrichten erfolgreich empfangen werden können. Beim Test des CAN-Identifiers wurden die 2^{11} Kombinationen für den Identifier jeweils mit einer CAN-Datenlänge von acht Bit und den Datenbytes mit den Einträgen 0xFF gewählt.

Beim Test der 2^8 Kombinationen der acht Datenbytes wurde immer der Identifier 0x103 verwendet, die gerade nicht im Test befindlichen Datenbytes erhielten jeweils den Wert 0xFF.

Nach dem erfolgreichen Empfangen aller getesteten Nachrichten wurden spezielle CAN-Nachrichten getestet. Dazu wurde eine CAN-Nachricht ausgewählt, die sowohl ein Stuff-Bit ganz am Ende des Datenfeldes als auch ein Stuff-Bit am Ende des CRC-Feldes beinhaltet. Die Nachricht lautet: Identifier: 0x373, DLC: 0x01, Datenbyte: A0 und CRC: 0x5720 und testet somit die beiden kritischsten Stellen für Stuff-Bits gleichzeitig. Darüber hinaus sollte der Controller eine CAN-Nachricht empfangen,

die auf ein Stuff-Bit fünf weitere Bits desselben Wertes beinhalten sollte („6er Stuff-Bit-Test“). Dieser Test prüfte vor der Submission ein letztes Mal, ob die in Kapitel 6.4.1 beschriebene Korrektur erfolgreich funktioniert. Zuletzt wurde der Sonderfall einer CAN-Nachricht mit Datenlänge Null auf den Bus gesendet. Diese Stelle ist programmiertechnisch als Sonderfall zu behandeln, da auf das Kontroll-Feld hier unmittelbar das CRC-Feld folgt und die Berechnung der Prüfsumme direkt mit der Information einer Datenlänge von Null abgeschlossen werden muss.

Für den Empfänger wurden im Weiteren alle möglichen Kombinationen aus den Adressen der Controller und der DCS-Chips geprüft und somit die Kodierung des Identifiers verifiziert. Da nach diesen statischen Tests nun auch gewährleistet werden sollte, dass über die getesteten Nachrichten hinaus alle CAN-Nachrichten erfolgreich empfangen werden können, wurde eine KVASER-Software eingesetzt, mit der CAN-Zufallsnachrichten auf den Bus gesendet werden können. Die Nachrichten sind dabei zufällig lang und werden in Bündeln von 1 bis 5 CAN-Nachrichten in zufälligen Zeitabständen zwischen 50 ms und 200 ms gesendet. Nachdem der Test mit Zufallsnachrichten fehlerfrei eine Woche durchgelaufen war[7)], wurde der Empfänger als erfolgreich getestet angesehen.

Tests des Senders

Für die Tests des Senders des Controllers wurde der bisherige Aufbau aus DCS-PC, DCS-Controller in Form eines XILINX FPGAs (auf dem SPARTAN Evaluation Board) und DCS-Chip, wie in Abbildung 6.11 zu sehen ist, erweitert. In diesem Aufbau wurde ein zweiter CAN-Bus dazu verwendet, um auf DCS-Befehl einen Ausgang des Mikrocontrollers zu setzen, der im FPGA einen Zähler inkrementierte.

Somit wurden pro Schritt zwei CAN-Befehle gesendet, die nacheinander zunächst den Zähler inkrementierten und danach den Wert des Zählers auf den Bus sendeten. Da es zeitlich nicht möglich ist, alle Kombinationen an CAN-Nachrichten zu erzeugen, wurden nacheinander alle 2048 Werte der 11-Bit-Message-ID und danach jeweils alle 255 Werte der acht Datenbytes vom Controller auf den Bus gesendet. Es wurden immer alle acht Datenbytes in der CAN-Nachricht versendet und die Datenbytes, die gerade nicht am Test beteiligt waren, auf 0xFF gesetzt. Der Identifier entsprach, sofern er nicht getestet wurde, immer demselben Wert der CAN-Anfrage, da die Anforderung des Zählerwertes für diesen Test mit Hilfe eines Datenanforderungsrahmens (RTR-Bit = 1) realisiert wurde. Die vollautomatische Ansteuerung der CAN-Komponenten erfolgte mit LabView. Der resultierende Test dauerte etwa

[7)] Es wurden nur fehlerhafte Nachrichten erzeugt, die aufgrund des Tests auftreten können. Die Zufallsnachrichten waren in seltenen Fällen bis zum DLC identisch mit der Antwortnachricht des Controllers, die Arbitrierung erfolgte allerdings nur auf den Identifier, so dass der PC und der DCS-Controller beide als Sender erkannt wurden. Sobald sich die Nachricht dann in einem Bit unterscheidet, gehen die Knoten in den Fehlerzustand, weil sie einen Bitfehler erkannt haben.

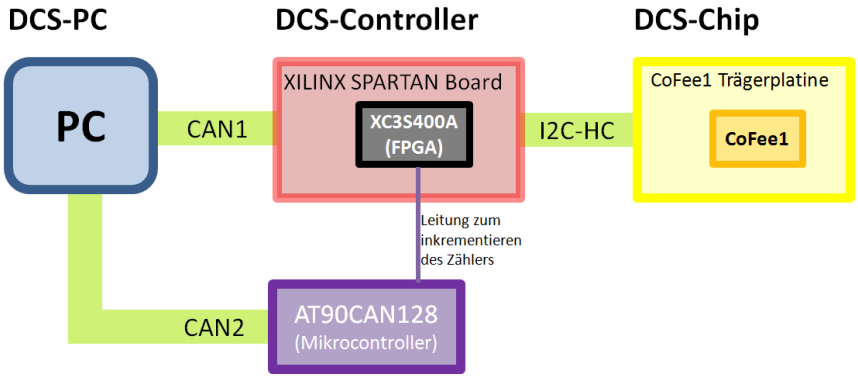


Abbildung 6.11: Hardware-Aufbau zur Realisierung der Sendertests

zwei Stunden und lieferte ein fehlerfreies Resultat des CAN-Designs in Form eines Testberichtes.

Nach diesem allgemeinen Test wurden nun Spezialfälle vom Controller auf den Bus gesendet und geprüft, ob Fehler auftreten. Dazu wurde der Controller dazu veranlasst, auf einen Befehl hin eine bestimmte Antwort — mit Inhalt des Spezialfalles — zurückzusenden. Es wurden eine Nachricht mit DLC = 0 getestet, ein „6er-Stuff-Bit-Test“[8)] durchgeführt und eine CAN-Nachricht auf den Bus gesendet mit einem Stuff-Bit unmittelbar vor der CRC-Checksumme und vor dem Acknowledge-Bit wie schon für den Test des Empfängers.

Um die korrekte Reaktion des DCS-Controllers auf DCS-Befehle hin zu testen, wurden zunächst alle Fehlercodes durch das Senden fehlerhafter DCS-Befehle erzeugt. Zusätzlich wurden alle digitalen Ausgänge in jeder Kombination gesetzt und auf die korrekte Acknowledge-Nachricht des Controllers geschaut. Außerdem wurden verschiedene ADC-Werte erzeugt[9)] und die korrekte Übertragung geprüft. Da allerdings das Senden der ADC-Werte eine normale CAN-Nachricht ohne weitere Funktionen ist, brauchte nicht jeder ADC-Wert getestet werden, da der generelle Test auf das allgemeine Senden von CAN-Nachrichten schon erfolgte.

Das Senden nach dem Mindestabstand zwischen zwei CAN-Nachrichten wurde getestet, indem die KVASER-Karte mehrere aufeinander folgende CAN-Nachrichten sendet, während der DCS-Controller gleichermaßen eine Nachricht auf den Bus sendet. In Abbildung 6.12 sind eine Aufnahme des Oszilloskops vom CAN-Bus und darunter einige digitale Signale des Controllers zu sehen. Im unteren Bereich der Abbildung findet man die Interpretation der gesendeten Nachrichten auf dem Bus. Nachdem die KVASER-Karte zwei CAN-Nachrichten auf den Bus gesendet hat, sendet der

[8)] bezogen auf den Fehler des ersten Prototypen für den DCS-Controller

[9)] mithilfe eines DIP-Schalters, der als ADC-Eingang an die CoFee1-Trägerplatine angeschlossen ist

DCS-Controller direkt nach dem Mindestabstand zwischen zwei CAN-Nachrichten von 48 μs[10)] die Antwort auf die erste KVASER-Nachricht auf den Bus.

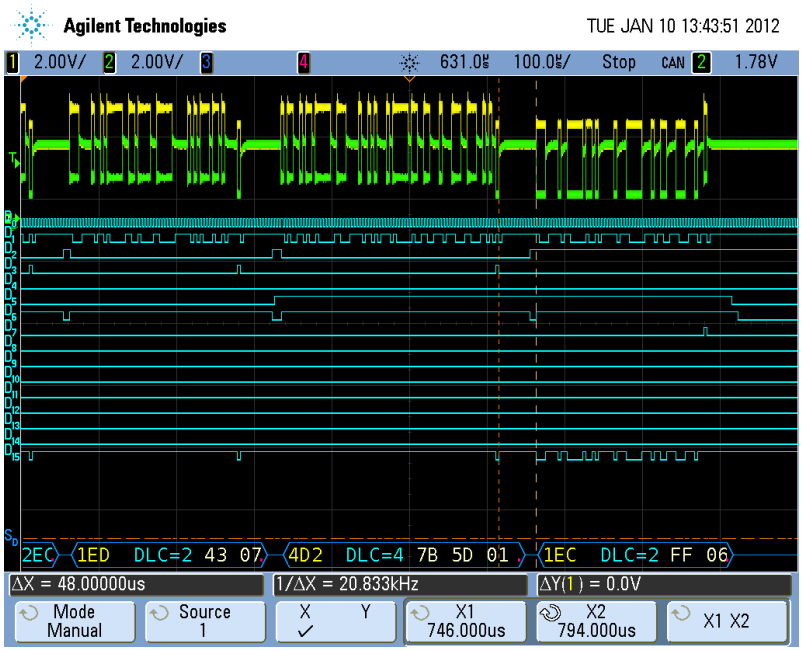


Abbildung 6.12: Erfolgreiches Senden nach dem Mindestabstand zwischen zwei CAN-Nachrichten von 11 Bit (48 μs)

Aufbau eines DCS-Netzwerks

Nachdem das System aus einem DCS-Controller und einem DCS-Chip alle Tests erfolgreich durchlaufen hatte, wurde aus den vorhandenen Komponenten ein DCS-Netzwerk aufgebaut und seine Funktion verifiziert.

Die Funktionsfähigkeit für ein System aus einem DCS-Controller und zwei bzw. vier DCS-Chips wurde geprüft, also der maximalen Anzahl an DCS-Chips pro Controller und drei Controllern mit jeweils drei DCS-Chips, wie es in Abbildung 6.13 zu sehen ist.

In allen Kombinationen wurde für jeden Controller zunächst der Schnelltest aus Kapitel 6.4.1 durchgeführt, ergänzt durch den Test mit der Datenlänge Null, und so die Funktionsweise des DCS-Netzwerkes verifiziert. Mit dem Aufbau aus Abbildung 6.13 wurde ein viereinhalb Tage langer Test mit Zufallsnachrichten durchgeführt, die in Form von 1-5 zufälligen CAN-Nachrichten mit zufälligem Zeitabstand von 50 ms bis 200 ms auftraten. Alle Tests waren erfolgreich und zeigen somit, dass aus

[10)] =1 Bit × 4 μs, bei einer Datenrate von 250 kHz und nach der Wartezeit von Ack-DEL, End of Frame und Rahmenzwischenraum

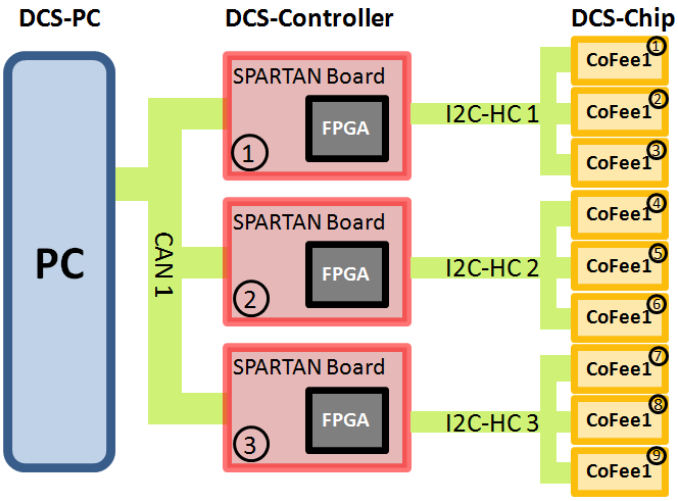


Abbildung 6.13: Aufbau eines DCS-Netzwerks mit drei DCS-Controllern und jeweils drei DCS-Chips

dem Design für den DCS-Controller ein funktionsfähiges DCS-Netzwerk aufgebaut werden kann.

6.4.4 Laufzeitsimulationen des Designs für den DCS-Controller

Die Laufzeitsimulationen nach der RTL-Synthese und nach der Layoutsynthese wurden mit dem Programm Cadence NCSim durchgeführt. Es wurden verschiedene Szenarien der DCS-Kommunikation simuliert, wie Schreibbefehle, Lesebefehle und das Senden der speziellen CAN-Nachricht, die an kritischen Stellen des Protokolls Stuff-Bits hat. Für den CAN-Controller und den simulierten zweiten CAN-Knoten wurden Clocksignale verwendet, die sich um 0,1 ns unterscheiden, um eine möglichst realistische Simulation zu erzeugen. Die Simulation zeigte in jedem Fall korrektes Verhalten, es traten keine Protokollfehler und keine Laufzeitprobleme auf.

In Abbildung 6.14 ist die Simulation der speziellen Stuff-Bit-Nachricht auf dem Bus gezeigt. Diese Nachricht wurde direkt im Anschluss an die vorhergehende gesendete Nachricht vom DCS-Controller auf den CAN-Bus gesendet. Somit wurde gezeigt, dass auch in der Simulation zwei direkt aufeinander folgende Nachrichten verarbeitet werden können. In Abbildung 6.15 ist die Simulation eines Schreibbefehls gezeigt. Zu sehen ist die CAN-Nachricht mit dem entsprechenden DCS-Befehl, die I2C-HC-Kommunikation zum DCS-Chip und die CAN-Nachricht auf dem Bus mit der enthaltenen Acknowledge-Nachricht.

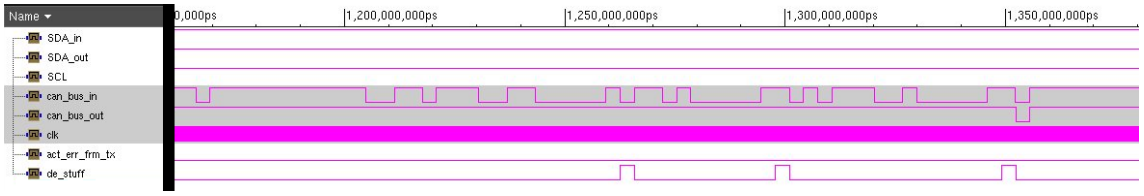


Abbildung 6.14: Beispiel der Simulation einer CAN-Nachricht auf dem CAN-Bus

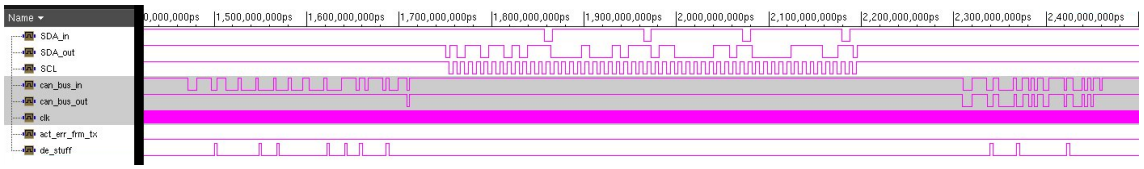


Abbildung 6.15: Beispiel der Simulation eines DCS-Schreibbefehls im DCS-Controller

6.4.5 Absicherung der Datenspeicherung gegen SEU

Im zweiten Prototypen wurde das TMR-Verfahren zur redundanten Auslegung der Chipregister nicht im Verilog-Code, sondern bei der RTL-Synthese angewendet[11)] [31]. Dabei wurde zunächst ein sogenanntes „TMR-Register“ erzeugt, welches aus drei Flip-Flops mit angehängtem Majoritätsvoter besteht. Dieses TMR-Register ist in Abbildung 6.16 gezeigt.

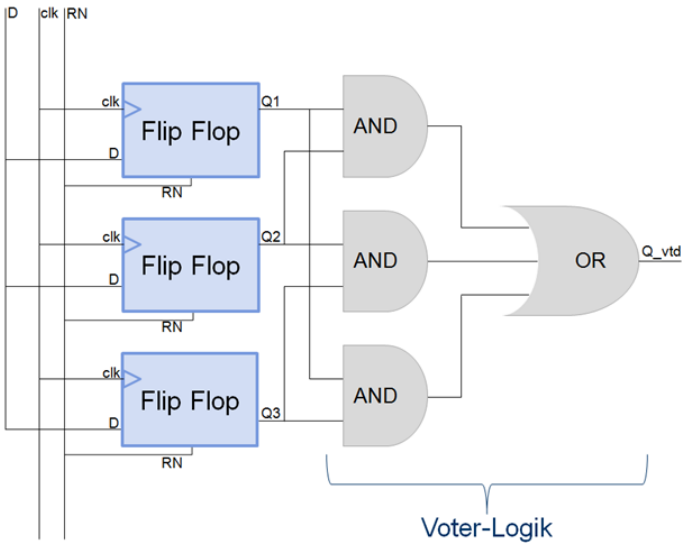


Abbildung 6.16: Darstellung eines TMR-Registers

[11)] Die Idee das TMR-Verfahren auf diese Art zu implementieren stammt von der ASIC-Gruppe der Universität Heidelberg. Besondere Hilfestellung bei der Implementation erhielt unsere Gruppe von A. Grübl.

Anhand der Voter-Logik ist zu erkennen, wie die Korrektur im Falle eines Bitfehlers funktioniert. Tritt kein Fehler auf und ändert sich auch der Wert der Flip-Flops nicht, so werden mit jedem Taktpegel die D-Eingänge der Flip-Flops immer neu mit Q_vtd beschrieben. Das verringert die kritische Zeit, in der ein Wert in einem Register abgespeichert werden muss.

Bei der Synthese wird in der erzeugten Netzliste nun jedes Register durch ein TMR-Register ausgetauscht und somit ein vollständiges TMR-Design für den DCS-Controller erzeugt.[12)] Dieses Verfahren ist leicht zu realisieren und birgt zudem wenig Fehlerquellen, da der Austausch der Register durch TMR-Register automatisiert ist. Aus dieser Sicht sollte für ein Digitaldesign immer dieser Weg der Implementation des TMR gewählt werden. Da an jedes dreifach-Flip-Flop ein Voter direkt angeschlossen ist, lässt sich allerdings kein Zähler einbauen, der direkt die Zahl der aufgetretenen Bitfehler im Chip unter Bestrahlung zählen könnte.

6.4.6 Submission zweier Designs für SEU-Studien

Für den zweiten Prototyp-Chip für den DCS-Controller wurden zwei Designs submittiert. Die beiden in Abbildung 6.17 gezeigten Chips beinhalten die Logik des DCS-Controllers einmal *ohne* TMR-Absicherung gegen SEU und einmal *mit*. Diese werden im Folgenden immer mit CoFee2_noTMR und CoFee2_TMR bezeichnet.

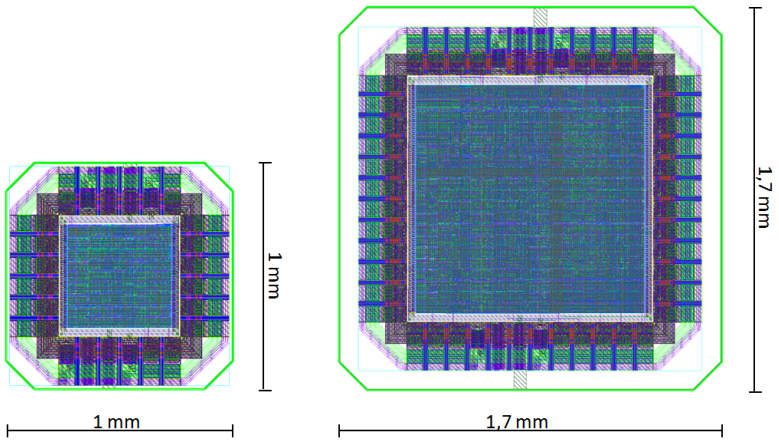


Abbildung 6.17: Submittiertes Design für den zweiten Prototyp-Chip für den DCS-Controller, links ist das Design ohne TMR (CoFee2_noTMR) und rechts mit TMR (CoFee2_TMR) zu sehen.

Bei der TMR-Implementation, wie sie für diesen Prototypen realisiert ist, konnte kein Zähler eingebaut werden, der die Anzahl an SEU in den Registern zählt. Die

[12)] Die TMR-Implementierung im RTL-Compiler wurde von Peter Kind mit Hilfe der Heidelberger ASIC-Gruppe durchgeführt.

Implementation eines solches Zählers ist sehr aufwendig und würde das Design stark vergrößern, da zu jedem der 1519 TMR-Register jeweils neben dem Majoritätsvoter auch eine Logik zur Fehlerdetektion implementiert werden müsste. Die Ausgänge jedes TMR-Registers müssten dann auf eine Zählerlogik geführt werden. Solch ein Konstrukt ist in der Netzliste des Compilers schwer zu erzeugen.

Damit trotzdem SEU-Studien erfolgen können, wurde auch das Design ohne TMR submittiert. Erwartungsgemäß sollte das TMR-freie Design während des Betriebs unter Bestrahlung Fehler aufzeigen und das Design mit TMR eine fehlerfreie Kommunikation ermöglichen. Dabei ist die Zahl der fehlerhaften Übertragungen im TMR-freien Design direkt proportional zu der Anzahl der SEU.

Auf die Funktionsweise und SEU-Studien des zweiten Prototypen für den DCS-Controller wird in den folgenden Kapiteln eingegangen.

6.5 Übersicht der Prototypen im 130 nm Prozess für das DCS

Im Rahmen dieser Arbeit wurden drei Prototypen für den DCS-Controller submittiert. Es wurden dabei aber nur zwei Iterationen des Verilog-Designs hergestellt, da sich zwei der DCS-Controller nur in der TMR-Implementation unterscheiden. Zur Vollständigkeit wird im Folgenden eine Tabelle erstellt, die alle submittierten Designs der Wuppertaler DCS-Gruppe enthält.

Chipname	digital/analog	Funktion
CoFee1	digital	DCS-Chip und DCS-Controller
PhysLay	analog	CAN-Treiber Chip
CoFee2_noTMR	digital	DCS-Controller 2. Design ohne TMR
CoFee2_TMR	digital	DCS-Controller 2. Design mit TMR
V_{Ref}-Chip	analog und digital	Referenzspannungsquelle und Schieberegister für Bestrahlungstests

Tabelle 6.5: Die submittierten Designs der Wuppertaler DCS-Gruppe. Alle Chips wurden im 130 nm Prozess produziert.

Kapitel 7

Aufbau des Control und Feedback Pfades

Nach der Produktion der beiden DCS-Controller CoFee2_noTMR und CoFee2_TMR wurden diese ausführlich auf ihre Funktionsweise geprüft. Daraufhin konnten auch einige Eigenschaften der Chips wie Temperaturabhängigkeit des Core-Stroms, ihre Reichweite der Kommunikation sowie der Leistungsverbrauch bestimmt werden. Im Folgenden wird zuerst genauer auf die eingeschränkte Funktionalität von CoFee2_TMR eingegangen und der Grund des vorliegenden Fehlers herausgearbeitet.

7.1 Der zweite Prototyp des DCS-Controllers

Aus den ersten Funktionstests für die beiden DCS-Controller ergab sich eine stark eingeschränkte Funktionalität für den Chip CoFee2_TMR. Es wurden diverse Studien durchgeführt, um das Problem zu verstehen. Eine gezielte Fehlersuche zusammen mit dem digitalen Chipdesigner S. Bonacini am CERN führte schließlich zum Verständnis der Ursache des Problems.

Aufgrund der falschen Zuweisung eines Chip-Anschlusses, an dem der externe Takt angeschlossen wird, wurde dieser nicht als Clock-Leitung definiert und somit im Chip kein sogenannter „Clock-Tree“ aufgebaut. Ein Clock-Tree dient dazu, den Takt an alle Flip-Flops gleichzeitig anzulegen, damit keine Setup- oder Hold-Fehler auftreten können. Der Clock-Tree ist derart im Chip-Schaltplan angelegt, dass alle Clock-Leitungen dieselbe Länge zu den Flip-Flops haben. Setup- und Hold-Fehler treten immer dann auf, wenn sich in einem gewissen Bereich, vor bzw. nach der Taktflanke, der Wert des Flip-Flops ändert. Durch eine unterschiedliche Länge der Clock-Leitung kann es dazu kommen, dass der Takt am Flip-Flop durch Laufzeiten Fehler auslöst. Aus diesem Grund werden ausführliche Laufzeitsimulationen gemacht, die auch für diese Chips durchgeführt wurden.

Leider gibt es bei der Simulation drei verschiedene Fälle, die simuliert werden müssen. Diese drei Fälle unterscheiden sich in den verschiedenen Kombinationen aus simulierter Betriebstemperatur und Stromverbrauch des Chips, was jeweils unterschiedliche Laufzeiten zwischen den Gattern (kleine, mittlere und sehr lange Laufzeiten) zur Folge hat. Es wurde aber nur eine Simulation für den „worst case“ Fall durchgeführt. In diesem Fall wird angenommen, dass der Chip einer hohen Temperatur ausgesetzt ist und niedrige Spannungen anliegen. Für diesen Fall war die Simulation fehlerfrei.

Für die beiden anderen Fälle, den „Normalfall“ und den Fall der optimalen Betriebsbedingungen (mit einer niedrigen Umgebungstemperatur und hoher Eingangsspannung), hätte die Simulation den fehlenden Clock-Tree aufgedeckt.

7.1.1 Probleme mit CoFee2_TMR

Aufgrund des oben beschriebenen, fehlenden Clock-Trees funktioniert der TMR abgesicherte DCS-Controller nur sehr eingeschränkt. Mit einer Versorgungsspannung von 1 V lassen sich die meisten Chips stabil derart betreiben, dass sie einen CAN-Befehl einlesen können und diesen korrekt an den DCS-Chip weitergeben. Nur das Senden eines CAN-Befehls, also die positive Bestätigung, dass ein DCS-Schreibbefehl erfolgreich war, oder die Durchführung eines Lesebefehls, schlagen fehl. Es lassen sich mit diesem Prototypen keine korrekten CAN-Nachrichten auf den Bus senden. Dieses Verhalten ist stark abhängig vom Core-Strom, je geringer dieser ist, desto stabiler wird das Verhalten. Dies macht auch Sinn, da durch eine niedrige Versorgungsspannung die Schaltzeiten der Transistoren im Chip vergrößert werden und die Bedeutung von Laufzeitproblemen geringer wird.

Mit dem TMR abgesicherten Prototypen lassen sich somit nur der CAN-Empfängerbereich und der Sendebereich des I2C-HC-Masters korrekt betreiben, was in etwa die Hälfte der Chipregister in Anspruch nimmt.

Bei dem viel kleineren Design des DCS-Controllers ohne TMR-Absicherung ist der fehlende Clock-Tree nicht so problematisch und äußert sich nicht in einer eingeschränkten Funktionalität des Chips. Im Gegenteil funktioniert dieser sehr gut, obwohl auch hier die Simulationen für die beiden nicht durchgeführten Fälle Fehler vorausgesagt hätten.

7.1.2 Funktionalität des CoFee2_noTMR

Entsprechend der vorhergehenden Tests aus Kapitel 6 funktioniert der DCS-Controller ohne TMR-Absicherung (CoFee2_noTMR) einwandfrei. Es wurde zunächst das Testprogramm aus Kapitel 6.4.1 zusammen mit dem Aufbau aus Abbildung 7.1 verwendet, in dem das Schreiben und Lesen der DCS-Chip-Register, das Verhalten der Fehlerzähler und der Fehlercodes sowie spezielle CAN-Nachrichten mit Stuff-Bits

an kritischen Stellen geprüft wurden. Für alle Chips verliefen die Tests positiv, es konnten keine Fehler gefunden werden.

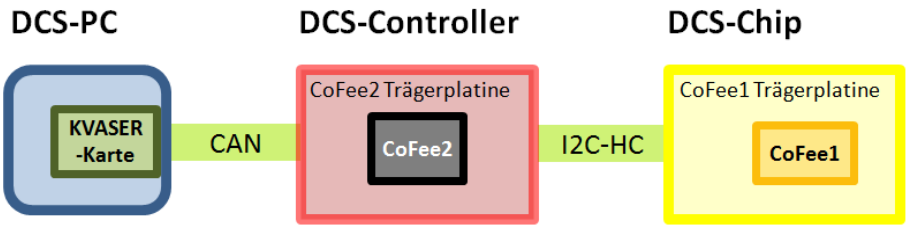


Abbildung 7.1: Schematische Zeichnung zum Aufbau der Tests mit dem nicht TMR-abgesicherten DCS-Controller

Die Fehler im ersten Prototypen des DCS-Controllers, beschrieben in Kapitel 6.4.1, wurden im zweiten Prototypen behoben. Sowohl die CAN-Synchronisation als auch das Bit-Stuffing verursachen keinerlei Fehler. Das Senden bzw. Empfangen zweier aufeinander folgender CAN-Nachrichten funktioniert entsprechend der CAN-Spezifikation. Auch der Fehler im I2C-HC-Protokoll wurde behoben.

In Abbildung 7.2 ist das positive Verhalten des DCS-Controllers während der Arbitrierung zu sehen. Die Oszilloskopaufnahme des CAN-Busses zeigt die CAN-High und CAN-Low Leitung in gelb und grün, sowie die Tx-Leitung des Controllers in blau. In der Nachricht vom DCS-Controller wird im zweiten Bit die Arbitrierung gewonnen, da der DCS-Controller mit dem Identifier „1A0“ eine höhere Priorität hat als die KVASER-Karte mit „2CB“ und sich die beiden Identifier im 2. Bit unterscheiden. Des Weiteren ist in der Abbildung 7.2 zu sehen, dass der neue Prototyp im Gegensatz zum vorherigen nun auch nach dem minimalen Nachrichtenzwischenraum von 12 Bit (=48 μs) nach dem Acknowledge-Signal der vorhergehenden Nachricht[1)] neue Nachrichten senden bzw. empfangen kann.

Nachdem die Funktionen des DCS-Controllers einzeln erfolgreich geprüft wurden, konnte nun ein Test mit Zufallsnachrichten durchgeführt werden. Es wurden 16 Stunden lang Zufallsnachrichten mit einer zufälligen Länge von 1 bis 5 CAN-Nachrichten und einem zufälligen zeitlichen Abstand gesendet. Der Controller reagiert mit der Bestätigung der Nachricht durch das Acknowledge-Signal oder antwortet mit einer entsprechenden Reaktion, wenn eine Zufallsnachricht einem DCS-Befehl entspricht. Dieser Test wurde durchgeführt, um abschließend zu bestätigen, dass der DCS-Controller den gesamten CAN-Nachrichtenraum korrekt empfangen kann.

Der DCS-Controller ohne TMR-Absicherung funktioniert entsprechend dem Design und kann für weitere Messungen, wie z.B. einen Aufbau des Control und Feedback Pfades, eingesetzt werden.

[1)] 12 Bit = 1 Bit Ack Begrenzungsbit + 8 Bit Rahmenende + 3 Bit Zwischenraum

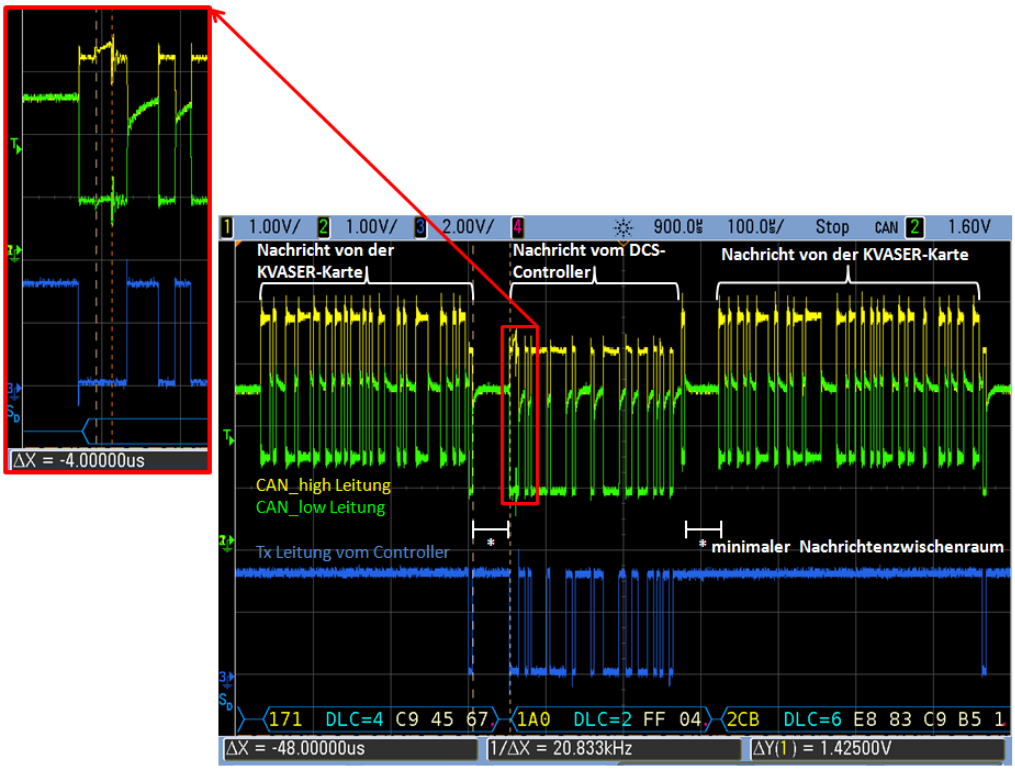


Abbildung 7.2: Oszilloskopaufnahme vom CAN-Bus mit CAN-High (gelb) und CAN-Low (grün) Leitung, sowie der Tx-Leitung des DCS-Controllers (blau). Der Controller sendet und empfängt nach dem minimalen Nachrichtenzwischenraum korrekt. In der Vergrößerung ist zu sehen, wie in der Nachricht vom DCS-Controller der Controller im 2. Bit die Arbitrierung gewinnt, weil er das dominante Bit sendet.

7.2 Leistungsverbrauch des DCS-Controllers

Für die beiden DCS-Controller wurden jeweils Messungen während des Betriebs bei einer Temperatur von 30,5 °C aufgenommen, um ihren Leistungsverbrauch zu veranschaulichen. Dazu wurde der Aufbau aus Abbildung 7.3 verwendet.

In Abbildung 7.4.a ist die Leistung des CoFee2_noTMR gegen die Messzeit aufgetragen. Die Schwankungen im Leistungsverbrauch ergeben sich aus dem Betriebsmodus des Chips, der analog zum Betriebsprogramm unter Bestrahlung, beschrieben in Kapitel 9.1, betrieben wurde. Die Eingangsspannung am Core beträgt 1,25 V. Während eines eingehenden CAN-Befehls und der I2C-HC-Kommunikation erhöht sich der Verbrauch und fällt dann wieder ab, wenn der Chip kurzzeitig inaktiv ist, bis der nächste DCS-Befehl eingelesen werden kann. Insgesamt liegt für den CoFee2_noTMR der Leistungsverbrauch bei durchschnittlich $64{,}65 \pm 0{,}13\,\mu$W. Der Leistungsverbrauch anderer CoFee2_noTMR liegt jeweils im selben Bereich mit kleinen fertigungsbedingten Abweichungen.

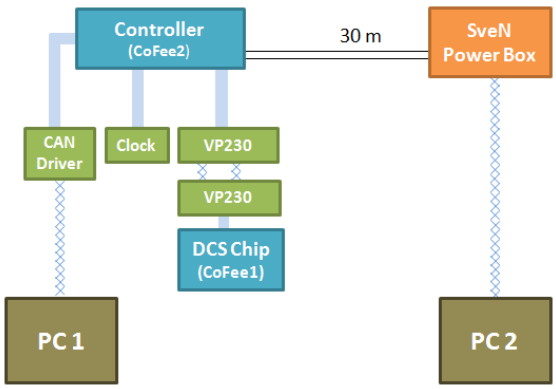
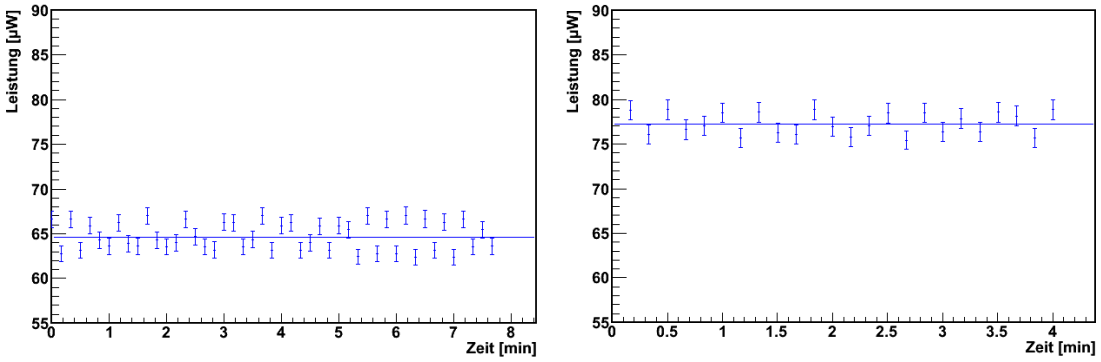


Abbildung 7.3: Aufbau zur Messung des Leistungsverbrauchs der DCS-Controller. Die Stromversorgung und Messung wurde mit der Sven-Box [32] über 30 m lange Kabel realisiert.



(a) (b)

Abbildung 7.4: Leistungsverbrauch der CoFee2-Chip-Cores (a) des DCS-Controllers ohne TMR-Absicherung und (b) des DCS-Controller mit TMR-Absicherung

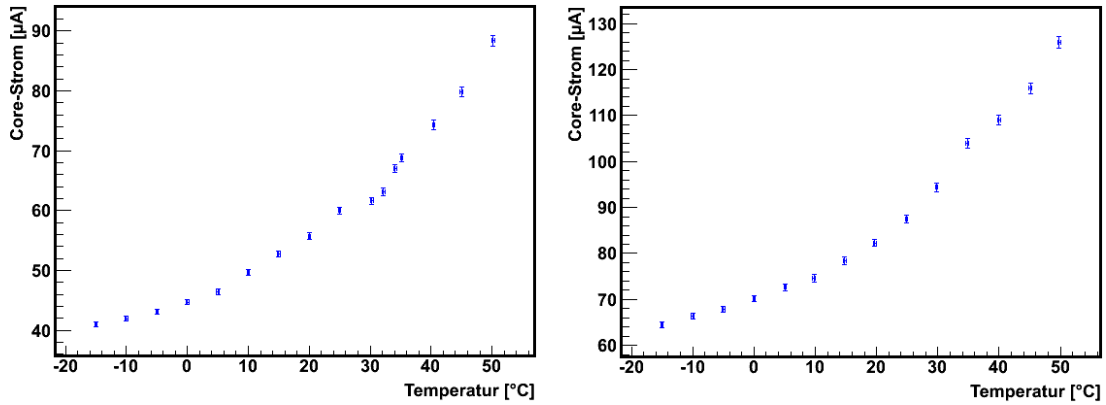
Für den CoFee2_TMR ist die Kurve des Leistungsverbrauchs in Abbildung 7.4.b gezeigt. Das Betriebsprogramm unterscheidet sich vom nicht TMR abgesicherten Controller, da CoFee2_TMR nicht komplett funktioniert und nur eingeschränkt DCS-Befehle ausführen kann. Auch arbeitet dieser Chip eher mit einer niedrigeren Versorgungsspannung, so dass dieser Chip mit 1 V versorgt wird. Im Durchschnitt verbraucht dieser Controller etwa $77{,}27 \pm 0{,}22\,\mu$W.

Vergleicht man den Leistungsverbrauch von Cofee2_noTMR mit $6{,}5\,\frac{mW}{cm^2}$ und Cofee2_TMR mit $2{,}7\,\frac{mW}{cm^2}$ mit dem FE-I4-Chip, der eine Leistung von $160\,\frac{mW}{cm^2}$ [25] verbraucht, so liegen die CoFee-Chips deutlich unterhalb des FE-I4-Chips. Für den Einbau in den Pixeldetektor ist der deutlich geringere Verbrauch der CoFee-Chips ausschlaggebend, da der FE-I4-Chip bereits eine Kühlung benötigt, die für die CoFee-Chips nicht vorgesehen ist.

7.2.1 Die Temperaturabhängigkeit des Core-Stroms

Die Abhängigkeit des Core-Stroms von der Temperatur wurde bestimmt, um unter Bestrahlung die auftretenden Effekte am Core-Strom besser verstehen zu können.

Für die Bestimmung der Temperaturabhängigkeit des Core-Stroms des DCS-Controllers wurde der Aufbau aus Abbildung 7.1 verwendet, wobei die Platine mit dem DCS-Controller in einen Wärmeschrank (MK 53) von Firma Binder platziert wurde. Mit dem Wärmeschrank sind Temperaturen von -40 °C bis zu 180 °C einstellbar, wobei der Bereich nicht komplett ausgenutzt wurde, um die Elektronik nicht zu gefährden. Die Temperatur selbst wurde nicht direkt am Chip gemessen, sondern die Umgebungstemperatur im Wärmeschrank wurde für die Messung verwendet. Es soll nur der Trend der Temperaturabhängigkeit des Core-Stroms gezeigt werden.



(a) (b)

Abbildung 7.5: Temperaturabhängigkeit des Core-Stroms (a) des CoFee2_noTMR und (b) des CoFee2_TMR

Die Ergebnisse der Messung mit beiden Designs für den DCS-Controller sind in Abbildung 7.5 zu sehen. Die Abhängigkeit des Core-Stroms von der Temperatur ist nicht vernachlässigbar. Der Stromverbrauch steigt mit der Temperatur an. So verdoppelt sich der Core-Strom bei einer Temperaturerhöhung von etwa 50 °C bei beiden Chips. Bei der Temperatur des Wärmeschranks von etwa 36 °C gab es Schwierigkeiten beim Einstellen. In beiden Graphen ist deshalb eine leichte Abweichung von der Kurve bei dieser Temperatur zu sehen.

Die Abhängigkeit des Core-Stroms von der Temperatur ist dadurch begründet, dass sich durch Erwärmung die Leitfähigkeit der Elektronen im Valenzband erhöht. Durch diese erhöhte Leitfähigkeit sinkt die Schwellspannung der CMOS-Komponenten und somit steigt der Strom bei konstanter Versorgungsspannung an.

Insgesamt liegt die in beiden Abbildungen 7.5 gemessene Erhöhung des Stromverbrauchs im erwarteten Bereich. Der maximale Stromverbrauch des DCS-Controller

im Pixeldetektor muss also auch im Hinblick auf die Betriebstemperatur ermittelt werden.

Des Weiteren ist nun bekannt, dass eine geringe Erhöhung des Core-Stroms während einer Bestrahlung nicht unbedingt auf Dosiseffekte zurückzuführen ist. Eine große Stromänderung wiederum ist sehr wahrscheinlich auf Dosiseffekten begründet.

7.3 Die Reichweite der I2C-HC-Kommunikation

Um die Ausgänge der CoFee-Chips auf die korrekten Buspegel zu bringen, wird ein CAN-Treiber benötigt. Solch eine DCS-Komponente im kommerziellen 130 nm Prozess wurde von L. Püllen entwickelt. Der PhysLay-Chip erzeugt ein differentielles Signal mit einem Spannungshub von maximal 300 mV, wie in Abbildung 7.6 zu sehen ist.

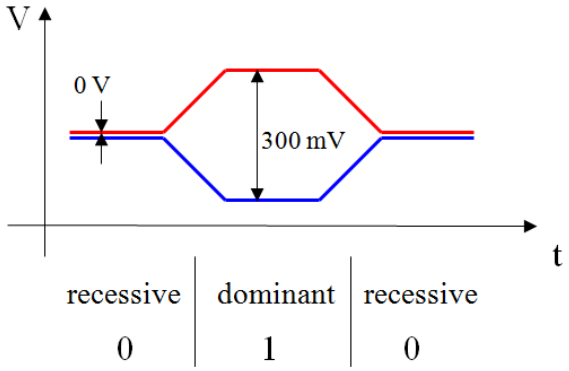
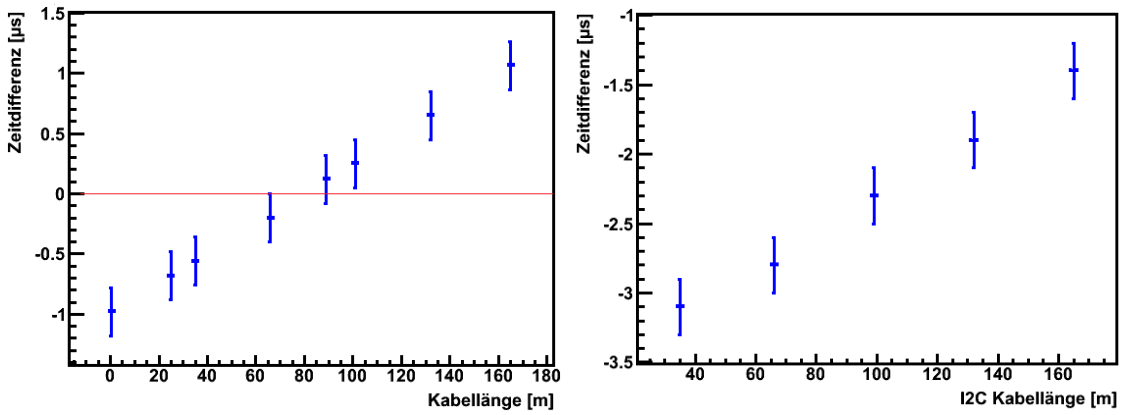


Abbildung 7.6: Veranschaulichung des differentiellen Signals auf dem Bus erzeugt durch den PhysLay-Chip [33].

Das (analoge) Design des CAN-Treibers wird im späteren DCS-Netzwerk sowohl für die CAN-Kommunikation als auch für die I2C-HC-Kommunikation eingesetzt, da beide Datenpfade über denselben, differentiellen Bus gesendet werden. Durch die differentielle Übertragung wird die Reichweite der I2C-HC-Kommunikation erhöht. Es gilt nun, die Reichweite der I2C-HC-Kommunikation in Abhängigkeit möglicher Datenraten zu bestimmen.

Für die Messung der Reichweite der I2C-HC-Kommunikation standen mehrere Kabel mit Längen bis insgesamt 165 m zur Verfügung. In den Abbildungen 7.7.a und 7.7.b ist die Zeitdifferenz des Datensignals gegenüber dem Taktsignal des I2C-HC-Protokolls aufgetragen worden. Bei einer positiven Differenz kann keine korrekte I2C-HC-Kommunikation aufgebaut werden. Für eine Datenrate von 200 kHz ist in Abbildung 7.7.a zu erkennen, dass mit einer Kabellänge über 80 m keine stabile Kommunikation mehr aufgebaut werden kann.



(a) (b)

Abbildung 7.7: Verzögerung des Datensignals gegenüber dem Taktsignal bei (a) 200 kHz und (b) 100 kHz I2C-HC-Übertragung

Für die Übertragung mit 100 kHz, zu sehen in Abbildung 7.7.b, sind keine Probleme in der Funktionalität der I2C-HC-Kommunikation bis zu 165 m Reichweite zu erwarten.

Es könnten noch durchaus größere Reichweiten bei höheren Datenraten erzielt werden, wenn im Design des I2C-HC-Masters im DCS-Controller eine Änderung vorgenommen würde. Dort wird bei eingehenden Datensignalen die ausgehende Taktflanke des DCS-Controllers als Takt verwendet. Dadurch werden die Verzögerungen noch ausgeprägter. Es wäre besser, wenn die eingehenden Daten auf den Takt des eingehenden Taktsignals aus dem Leitungstreiber getaktet werden. Die Laufzeiten zwischen Takt- und Datensignal würden dann nur noch minimal sein. Für einen nächsten Prototypen des DCS-Controllers sollte das mitberücksichtigt werden.

Da für die zukünftige DCS-Kommunikation eine Datenrate für das I2C-HC-Protokoll von 100 kHz verwendet werden wird und die Übertragung über eine Strecke von ca. 20 m erfolgt, zeigen die obigen Messungen, dass mit den vorhandenen Prototypen eine stabile I2C-HC-Kommunikation aufgebaut werden kann. Da die Übertragung bis 165 m fehlerfrei möglich ist, kann auch ein geeigneter Sicherheitsfaktor bei der Reichweite der I2C-HC-Kommunikation mit berücksichtigt werden.

7.4 Aufbau eines DCS-Netzwerks aus den vorhandenen Prototypen

Aus den Prototypen für das DCS-Netzwerk soll eine DCS-Kommunikationskette des Control und Feedback Pfades aufgebaut werden, welche in Abbildung 7.8 zu sehen ist.

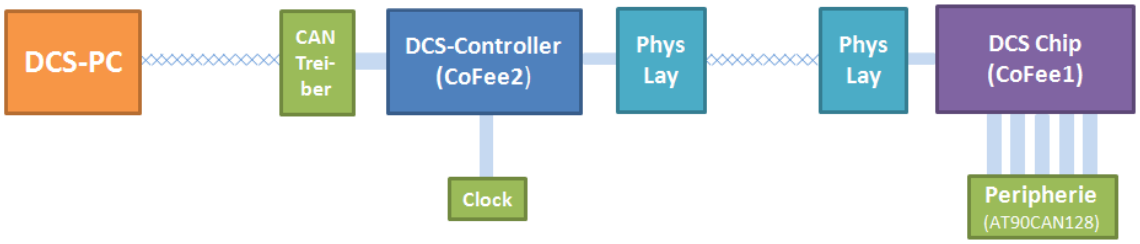


Abbildung 7.8: Aufbau des Control und Feedback Pfades. In Blautönen sind die bisherigen Prototypen dargestellt: der CoFee2-Chip für den DCS-Controller, CoFee1 für den DCS-Chip und der PhysLay-Chip

Der CoFee2-Chip stellt dabei den DCS-Controller dar, der CoFee1-Chip den digitalen Teil des DCS-Chips. Die PhysLay-Chips konnten nur für die I2C-HC-Kommunikation eingebaut werden, da nur zwei gebondete Chips auf den entsprechenden Platinen vorhanden waren. Somit musste ein kommerzieller CAN-Treiber für die CAN-Kommunikation verwendet werden. Der analoge Teil des DCS-Controllers, also der externe Taktgeber, ist auch in diesem Aufbau durch einen kommerziellen Quarz dargestellt. Der fehlende analoge Teil des DCS-Chips ist in diesem Aufbau durch einen Mikrocontroller mit entsprechender Schaltung ersetzt worden.

Die Kommunikation in diesem Aufbau lief stabil. Es konnten alle DCS-Befehle ausgeführt werden, das Testprogramm aus Kapitel 6.4.1 konnte auch in diesem Aufbau fehlerfrei durchlaufen werden. Es ist also möglich, mit den vorhandenen Prototypen einen Control und Feedback Pfad des DCS-Systems aufzubauen, der in einen Systemtest integriert werden könnte.

7.5 Zusammenfassung

Der DCS-Controller ohne TMR-Absicherung ist ein geeigneter Prototyp für die Weiterentwicklung des Control und Feedback Pfades. Es sind keine Protokollfehler bekannt und der DCS-Controller ist somit voll einsatzfähig. Auch liegen der Leistungsverbrauch des Controllers und die Temperaturabhängigkeiten seines Core-Stroms im erwarteten Bereich. Der DCS-Controller ist im Aufbau zusammen mit DCS-Chip und PhysLay-Chip funktionsfähig, so dass ein kompletter Control und Feedback Pfad aufgebaut werden kann.

Im Weiteren muss nun unter Bestrahlung verifiziert werden, dass die DCS-Datenübertragung in einer hohen Strahlungsumgebung den Anforderungen entsprechend korrekt funktioniert. Dazu werden DCS-Chip und DCS-Controller separat mit einem hohen Teilchenfluss bestrahlt und das Verhalten der DCS-Kommunikation untersucht.

Kapitel 8

Messung der SEU-Fehlerrate des DCS-Chips

In Kapitel 5 wurde das I2C-HC-Protokoll als geeignet für eine sichere Datenübertragung in der HL-LHC-Strahlungsumgebung herausgearbeitet. Um dies auch experimentell zu prüfen, wurde der erste Prototyp für den DCS-Chip in einer ersten Bestrahlungskampagne im Jahr 2011 mit Protonen bestrahlt.

Es soll eine möglichst realistische HL-LHC-Strahlungsumgebung hergestellt werden. Der Fokus wird dabei auf die Funktionalität des TMR und die Eigenschaften des I2C-HC-Protokolls gelegt, so dass die Erzeugung von Einzelfehlereffekten im Vordergrund vor der Erzeugung einer maximalen Gesamtdosis steht.[1)] Es wurde also nach einer Einrichtung mit einem möglichst hohen Teilchenfluss und relativ niederenergetischen Teilchen gesucht, um einen möglichst hohen Energieverlust der Teilchen erzielen zu können.

Ein geeigneter Kandidat für SEU-Studien ist das Paul Scherrer Institut (= PSI) in Villingen in der Schweiz. Dort befindet sich eine Einrichtung für die Bestrahlung mit Protonen, die Proton Irradiation Facility (= PIF), die von verschiedenen Forschungsgruppen benutzt werden kann. Da das PIF vielseitig einsetzbar sein muss, sind Protonenergien und Strahlintensitäten während des Betriebs veränderbar. Es können Energien zwischen 70 MeV und 250 MeV, sowie Teilchenflüsse bis maximal $\phi = 5 \cdot 10^8 \frac{Protonen}{cm^2 \cdot s}$ eingestellt werden.

Das PIF wurde aufgrund seines hohen Protonflusses, aber auch wegen der variablen Einstellbarkeit der Strahlgrößen ausgewählt. In Abbildung 8.1 ist der Aufbau für die im Folgenden beschriebene Bestrahlung gezeigt. Zu sehen ist die Experimentierhalle mit dem Strahlrohr.

[1)] Tests der Gesamtdosis sind für diesen Prototypen nicht sinnvoll, da nicht alle Komponenten strahlenhart implementiert wurden. Die Anschlusspads des Chips sind z.B. aufgrund von Kompatibilitätsproblemen Standardpads und somit nicht strahlentolerant.

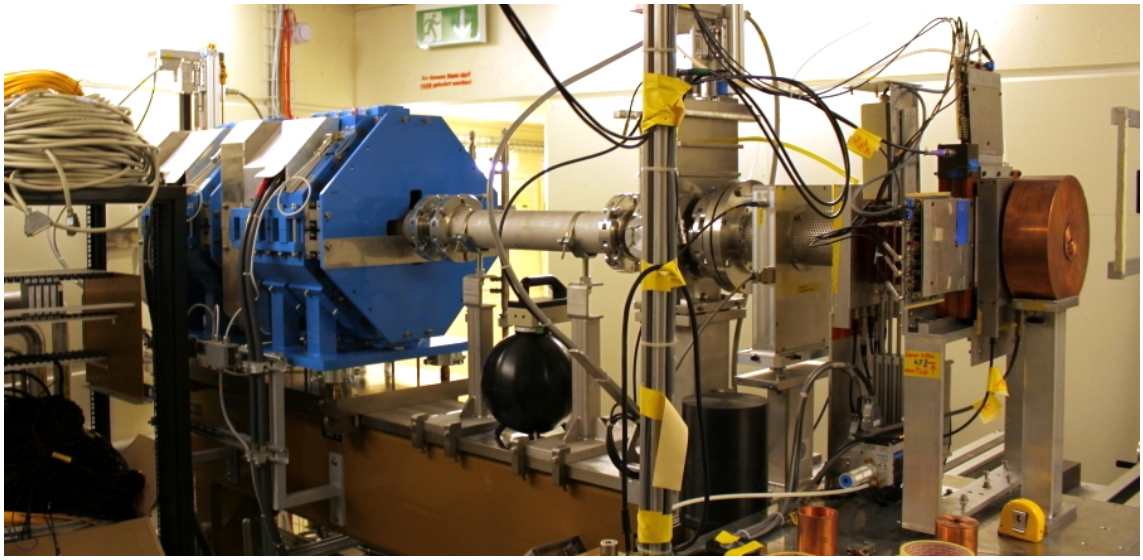


Abbildung 8.1: Experimentierhalle der Proton Irradiation Facility am PSI

In den folgenden Teilkapiteln wird nun genauer auf das Programm der Bestrahlung und die verschiedenen Ergebnisse eingegangen sowie eine untere Grenze für die Strahlenhärte der I2C-HC-Kommunikation angegeben.

8.1 Das Bestrahlungsprogramm

Da die Fehlerrate der Kommunikation des DCS-Chips ermittelt werden sollte, wurde dieser unter Bestrahlung betrieben. Dabei wurde eine möglichst realistische DCS-Kommunikation aufgebaut, in der auf die analogen Eingänge des DCS-Chips variable Werte gegeben, sowie sämtliche digitalen Ausgänge in jeder Kombination beschrieben wurden. Das gesamte System wurde gleichzeitig auch außerhalb des Strahls in einem Referenzsystem betrieben, so dass systematische Fehler in der Kommunikation ausgeschlossen werden konnten.

Um weitere systematische Fehler auszuschließen, wurde nur der DCS-Chip als zu testende Komponente mit in den Aufbau für die Bestrahlung integriert. Der DCS-Controller, der zum gleichen Zeitpunkt auch schon als Prototyp existierte, wurde bei der Bestrahlung nicht getestet. Wie in Kapitel 6.3 beschrieben, ist die Funktionalität des ersten Prototypen eingeschränkt, so dass dieser Controller in der folgenden Bestrahlung nicht eingesetzt wurde. Mithilfe eines Mikrocontrollers wurde die Funktionalität des DCS-Controllers nachgebaut. Für diese Aufgabe wurde ein Mikrocontroller des Typs AT90CAN mit dem Standard CAN-Protokoll ausgewählt, so

dass nur die DCS-Funktionen des Controllers und ein I2C-HC-Master implementiert werden mussten[2)].

Die Protonenergie am PIF wurde auf 100 MeV festgelegt. Die Teilchenflüsse wurden von $10^6 \frac{Protonen}{cm^2 \cdot s}$ bis maximal $5 \cdot 10^8 \frac{Protonen}{cm^2 \cdot s}$ eingestellt.

Für die Bestimmung des SEU-Wirkungsquerschnittes war im CoFee1-Chip kein entsprechender Mechanismus[3)] vorhanden, sondern es wurde stattdessen der sogenannte „Scan-Chain-Mechanismus“ genutzt, um die Zahl der Einzelfehlereffekte zu bestimmen.

Eine Scan Chain wird im digitalen Chipdesign dafür verwendet, die momentanen Werte der Chipregister auszulesen. In diesem Betriebsmodus wird die eigentliche Funktionalität des Chips verworfen und es werden alle Chipregister in Serie geschaltet. Diese können dann wie ein Schieberegister ausgelesen werden. Dieser Betriebsmodus dient normalerweise zur Fehlerdiagnose bei Designtests. Schreibt man in jedes Register einen bekannten Wert, wartet unter Bestrahlung eine gewisse Zeit ab und liest dann diese Register wieder aus, kann man anhand der verfälschten Register die Zahl der aufgetretenen Bitfehler bestimmen. Da die Wartezeit relativ gering gewählt wurde (2 s) und das Auslesen des Schieberegisters sehr schnell ist (etwa 6 ms), war die Wahrscheinlichkeit für Doppeltreffer in redundanten Registern sehr klein.

In Abbildung 8.2 ist der Aufbau zur Bestrahlung des DCS-Chips dargestellt. Zu sehen ist, dass zwei Chips gleichzeitig bestrahlt wurden, denn neben dem DCS-Chip wurde der PhysLay-Chip für Dosismessungen rückwärtig mit in den Strahl gebracht. Die Ergebnisse dieser Messungen sind in [34] zu finden. Der redundante Aufbau der DCS-Kommunikation ist in der Abbildung direkt zu erkennen, da das System für die Kommunikation zum DCS-Chip ein weiteres Mal im unteren Bereich der Grafik zu sehen ist sowie der PhysLay-Chip auch außerhalb des Strahls und im unteren Bereich von Abbildung 8.2 zu finden ist. Eine Fotografie mit dieses Aufbaus mit Referenzsystem in der Experimentierhalle ist in Abbildung 8.3 zu sehen.

Der Pfad der DCS-Kommunikation (die Kommunikation vom PC zum DCS-Controller und darüber zum DCS-Chip) wird durch einen zweiten Kommunikationsweg ergänzt. Dieser Weg dient nur zur Fehleranalyse während der Bestrahlung und dem Einstellen der Betriebsmoden „DCS-Kommunikation“ und „Scan-Chain-Mechanismus“. Der Wechsel zwischen den Betriebsmoden ist durch den eingezeichneten Schalter angedeutet worden, welcher die I2C-HC-Kommunikation außer Betrieb setzen kann, um in den Scan-Chain-Modus zu wechseln. Für die Steuerung wird ein weiterer Mikrocontroller eingesetzt, der in Abbildung 8.2 mit Peripherie bezeichnet ist. Mit diesem können die DCS-Chip Eingänge beschrieben sowie die digitalen Ausgänge überprüft werden. Des Weiteren kann der DCS-Chip durch die Peripherie resettet werden, falls der Chip durch die Bestrahlung in solch einen Fehlerzustand

[2)] Diese Aufgaben übernahm P. Kind.

[3)] Es hätten z.B. Zähler im TMR Mechanismus implementiert werden können, um die Zahl der verfälschten Register zu bestimmen.

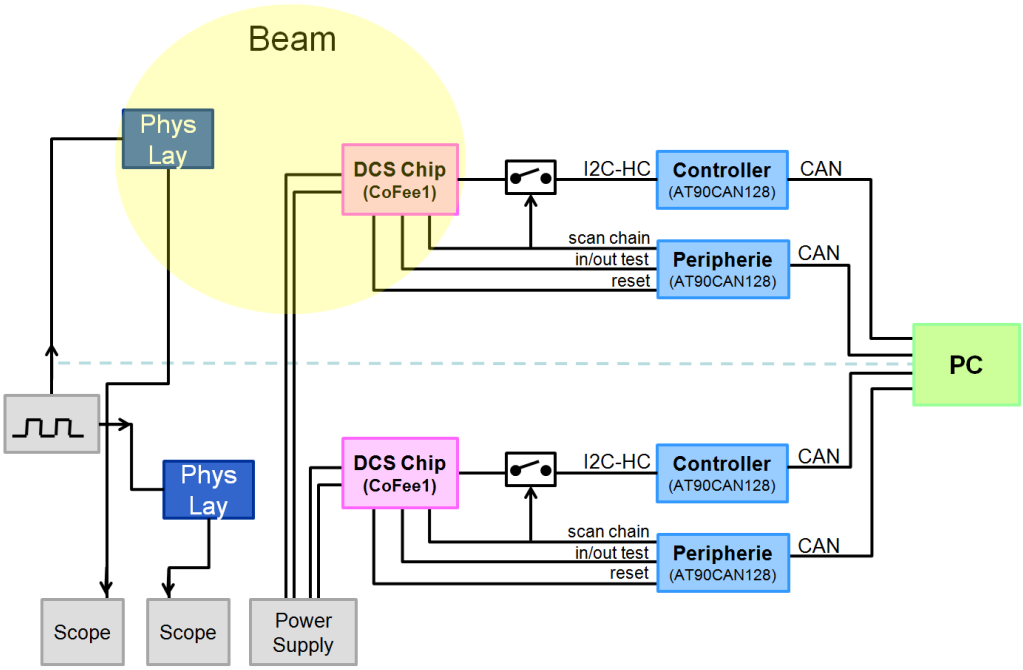


Abbildung 8.2: Aufbau zur Bestrahlung des DCS-Chips. DCS-Chip und PhysLay sind beide hintereinander, mittig im Strahl positioniert. Der Strahl trifft dabei zuerst auf den DCS-Chip.

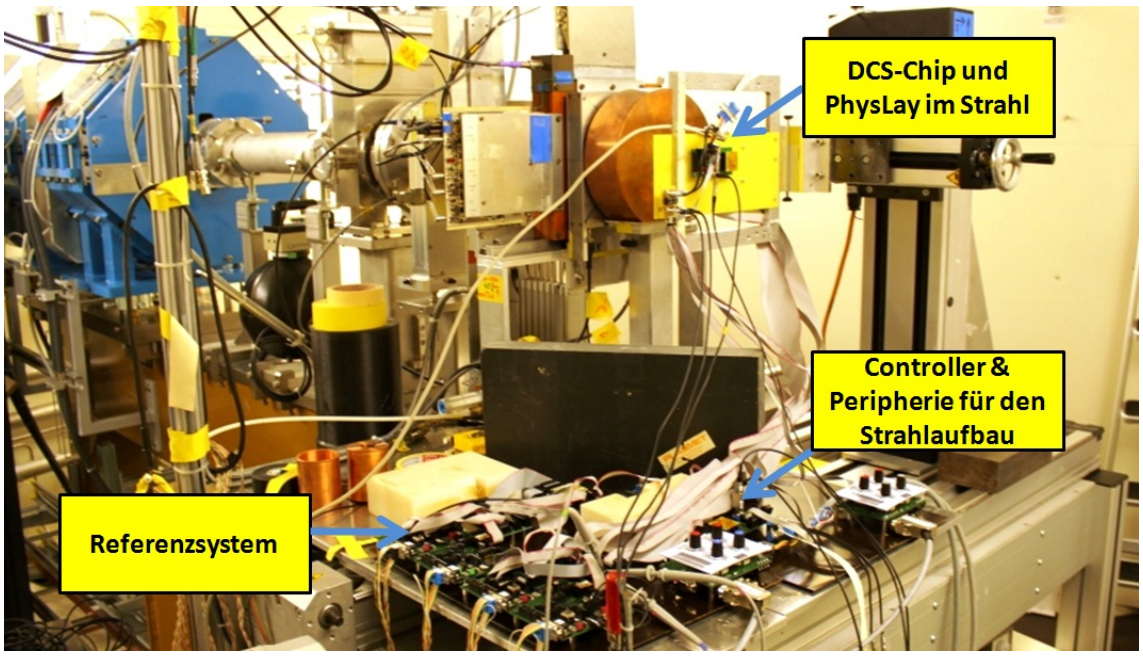


Abbildung 8.3: Fotografie des Aufbaus zur Bestrahlung in der Experimentierhalle am PSI. Links unten im Bild ist der Referenzaufbau zu sehen.

versetzt wird, dass alle Register wieder auf die Ausgangswerte zurückgesetzt werden müssen.

Während der Bestrahlung wurde kontinuierlich eine Messung des Strom- bzw. Spannungsverbrauchs des DCS-Chips durchgeführt, die in Abbildung 8.2 nicht genauer eingezeichnet wurde. Von insgesamt zehn Stunden Strahlzeit wurden fünf Stunden für die Bestimmung der Fehlerrate der DCS-Kommunikation und fünf Stunden für die Bestimmung des SEU-Wirkungsquerschnittes vorgesehen. Der genaue Bestrahlungsplan ist in Tabelle 8.1 zu finden.

Fluss $[\frac{p}{cm^2 \cdot s}]$	DCS-Chip (Referenzchip)	Betriebsmodus	Dosis [Rad]	Gesamtdosis [Rad]
10^6	7(6)	Normal	$3,36 \cdot 10^2$	
10^7	7(6)	Scan Chain	$3,80 \cdot 10^3$	
10^8	7(16)	Scan Chain	$3,33 \cdot 10^4$	
10^8	7(16)	Scan Chain	$3,33 \cdot 10^4$	
10^8	7(16)	Normal	$3,34 \cdot 10^4$	
$5 \cdot 10^8$	7(16)	Normal	$1,59 \cdot 10^5$	$2,62 \cdot 10^5$
10^8	22(6)	Scan Chain	$3,31 \cdot 10^4$	
10^8	22(6)	Normal	$3,31 \cdot 10^4$	
$5 \cdot 10^8$	22(6)	Scan Chain	$1,51 \cdot 10^5$	
$5 \cdot 10^8$	22(6)	Normal	$1,58 \cdot 10^5$	$6,37 \cdot 10^5$

Tabelle 8.1: Das Programm der Bestrahlung

Der Teilchenfluss wurde nach und nach erhöht, um jeweils sicherzustellen, dass die DCS-Kommunikation nicht aufgrund von Dosis-Effekten im Chip gestört wurde. Dies war bis zu dem höchsten Teilchenfluss von $\phi = 5 \cdot 10^8 \frac{Teilchen}{cm^2 \cdot s}$ nicht der Fall. Es wurden zwei verschiedene DCS-Chips bestrahlt, mit einer Gesamtdosis von $d_{Chip7} = 2{,}62 \cdot 10^5$ Rad und $d_{Chip22} = 6{,}37 \cdot 10^5$ Rad.

Anhand der verschiedenen Betriebsmoden kann nun eine Aussage über die Toleranz gegenüber Einzelfehlereffekten des ersten DCS-Chip-Prototypen gemacht werden. Mithilfe der Strommessung wurden Gesamtdosiseffekte im DCS-Chip beobachtet. Hierbei ist allerdings zu bemerken, dass die gemessenen Dosen deutlich unterhalb der zu erwartenden HL-LHC Dosen liegen, da der Fokus auf der Bestimmung der Toleranz gegenüber Einzelfehlereffekten lag. Die Ergebnisse werden im Folgenden dargestellt und diskutiert.

8.2 Wirkungsquerschnitt für SEU des DCS-Chips

Mithilfe des Scan-Chain-Betriebsmodus des DCS-Prototypen konnte die Rate der Einzelfehlereffekte unter Bestrahlung gemessen werden. Die genaue Kenntnis des

Wirkungsquerschnitts ist wichtig, um eine Aussage über die erwarteten Einzelfehlereffekte für die HL-LHC-Strahlungsumgebung machen zu können. Aus diesen Berechnungen kann dann theoretisch ermittelt werden, ob die DCS-Kommunikation ausreichend gegenüber Einzelfehlern abgesichert ist.

Die Bestimmung der Zahl an Einzelfehlern erfolgte in insgesamt fünf Messungen, bei denen drei unterschiedliche Flüsse[4)] verwendet wurden. Bei der Messung wurden jeweils bekannte Werte in die Chipregister geschrieben und nach Ablauf von 2 s Wartezeit die Register wieder ausgelesen. Stimmte ein Register nicht mit dem bekannten Wert überein, so konnte dies mit einem SEU gleichgesetzt werden. In die Bestimmung der Länge der Wartezeit flossen Faktoren wie die Wahrscheinlichkeit für einen Doppeltreffer und die Wahrscheinlichkeit für einen Treffer innerhalb der Schreib- bzw. Lesezeit ein.

Für die Bestimmung des SEU-Wirkungsquerschnitts wurde für alle Messungen die Anzahl von $N_{Latches} = 432$ Chipregistern verwendet. In Tabelle 8.2 sind die Messwerte und die mithilfe von Formel 4.4 berechneten Wirkungsquerschnitte dargestellt.

Fluenz $[\frac{p}{cm^2}]$	N_E	ΔN_E	$\sigma\ [\frac{cm^2}{bit}]$	$\Delta\sigma\ [\frac{cm^2}{bit}]$
$3,52 \cdot 10^{10}$	1	1	$6,58 \cdot 10^{-14}$	$6,71 \cdot 10^{-14}$
$3,53 \cdot 10^{11}$	5	1,2	$3,23 \cdot 10^{-14}$	$1,59 \cdot 10^{-14}$
$1,61 \cdot 10^{12}$	7	2,6	$1,01 \cdot 10^{-14}$	$0,43 \cdot 10^{-14}$

Tabelle 8.2: Die gemessene Zahl an Bitfehlern N_E und der berechnete Wirkungsquerschnitt σ aus Formel 4.4 in Kapitel 4.2.2, sowie ihre Fehler ΔN_E und $\Delta\sigma$

Sehr auffällig ist die kleine Zahl an detektierten Bitfehlern, die sich aufgrund der geringen Zahl der Register ergibt. Die gemessene Zahl der Bitfehler für die höchste eingestellte Fluenz von $F = 1,61 \cdot 10^{12} \frac{p}{cm^2}$ ist allerdings viel zu gering. Dasselbe Problem tritt auch bei der Bestrahlung des DCS-Controllers in Kapitel 9 auf und wird als Fehlmessung der Fluenz bei einem maximal eingestellten Strahlstrom interpretiert. Für diesen Fall liegt leider keine Korrektur vor, so dass als Konsequenz dieser Wert aus der Berechnung des Wirkungsquerschnitts herausgenommen wurde. Diese Annahme wird im Folgenden noch ausführlicher begründet.

[4)] Es wurde bei den Flüssen $\phi_1 = 10^7 \frac{p}{cm^2 \cdot s}$, $\phi_2 = 10^8 \frac{p}{cm^2 \cdot s}$, $\phi_3 = 5 \cdot 10^8 \frac{p}{cm^2 \cdot s}$ jeweils etwa 60 min lang gemessen, so dass sich die im weiteren verwendeten Fluenzen ergeben.

Da nur noch zwei Werte für die Bestimmung des Wirkungsquerschnitts zur Verfügung stehen, wird durch Mittelwertbildung der Wirkungsquerschnitt für den verwendeten 130 nm Prozess bestimmt zu:

$$\sigma = (4,91 \pm 4,15) \cdot 10^{-14} \frac{cm^2}{bit} \tag{8.1}$$

Verglichen mit dem Wert aus Kapitel 4.2.2 für den Wirkungsquerschnitt eines Designs im 130 nm Prozess $\sigma = 2$ bis $6 \cdot 10^{-14} \frac{cm^2}{bit}$ und dem Wert für den FE-I4-Chip $\sigma_{FEI4} = 5 \cdot 10^{-14} \frac{cm^2}{bit}$[5)] [35] stimmt der gemessene Wert für den DCS-Chip überein. Allerdings erkennt man an dem sehr großen Fehler die Probleme der zu geringen Statistik an gemessenen Bitfehlern und auch an durchgeführten Messungen. Der Wert für den Wirkungsquerschnitt liegt zwar im erwarteten Bereich, kann aber nicht als repräsentativ angenommen werden, da der Fehler viel zu groß ist.

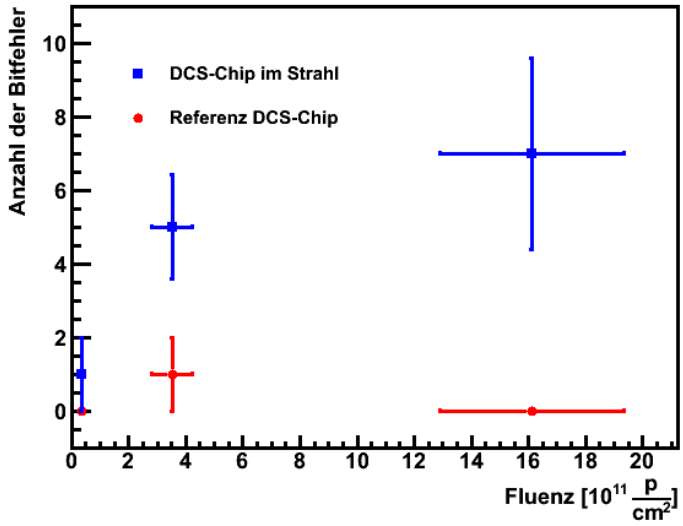


Abbildung 8.4: Gemessene Einzelfehlereffekte während der DCS-Kommunikation unter Bestrahlung für den Chip im Strahl in blau und den Chip im Referenzaufbau in rot

In Abbildung 8.4 sind die gemessenen Bitfehler abhängig von der Fluenz graphisch aufgetragen. Die blauen Messpunkte geben die Bitfehler des DCS-Chips im Strahl an, die roten Messwerte sind die Fehler, die im DCS-Chip des Referenzaufbaus gemessen wurden. Eigentlich hätten dort keine einzigen Bitfehler detektiert werden dürfen, da aber der Aufbau zu nah am Strahl stand und es eine zu hohe Streustrahlung in der Experimentierhalle gab, sieht man auch für den Referenzaufbau einen Bitfehler.

Die erwartete lineare Abhängigkeit der Zahl der Bitfehler von der Fluenz ist für den bestrahlten Chip nicht zu erkennen, weil der Wert für die höchste Fluenz zu stark

[5)] Angabe ohne Fehlerwert.

abweicht. Bei der Fluenz von $F_2 = 3{,}53 \cdot 10^{11}\ \frac{p}{cm^2}$ erfolgten drei Messungen, bei denen der Mittelwert aus diesen Messungen ($n_{E1} = 3 \pm 1{,}7$, $n_{E2} = 6 \pm 2{,}4$, $n_{E3} = 6 \pm 2{,}4$) eingezeichnet wurde. Der Fehler für die Fluenz wurde mit 20 % abgeschätzt.

Obwohl diese Methode zur SEU-Messung funktioniert, gab es bei der Durchführung einige Schwierigkeiten. Wegen technischer Probleme bei der synchronen Auslese des Schieberegisters gab es regelmäßige Wertänderungen mehrerer Bits. Besonders auffällig war ein Block von 160 Registern, deren Werte sich unkontrolliert änderten. Aus diesem Grund wurden diese Register aus der kompletten Auswertung herausgenommen, so dass aus den ursprünglich 592 Registern der Wirkungsquerschnitt nur mit 432 Registern bestimmt wurde.

Vergleicht man die Zahl der gemessenen Einzelfehlereffekte mit der Zahl der erwarteten, sieht man sehr schnell, dass eigentlich mehr Effekte hätten detektiert werden müssen. Mithilfe der Poisson-Statistik wurde die Wahrscheinlichkeit für die erwartete Zahl an SEUs berechnet.Mit einem Wirkungsquerschnitt von $\sigma = 5 \cdot 10^{-14}\ \frac{cm^2}{bit}$[6)] können für die drei verschiedenen Fluenzen, bei denen gemessen wurde, die erwarteten Zahlen an SEU aus der Poissonverteilung berechnet werden. Die Werte sind in Tabelle 8.3 zu finden.

	Fluenz $[\frac{cm^2}{bit}]$	Erwartete Anzahl an SEU aus Poissonverteilung	Gemessene SEU-Zahl aus Tabelle 8.2
a	$3{,}52 \cdot 10^{10}$	0,76	1
b	$3{,}53 \cdot 10^{11}$	7,62	5
c	$1{,}61 \cdot 10^{12}$	34,78	7

Tabelle 8.3: Bestimmung der erwarteten Anzahl an SEU aus der Poissonverteilung

Die Verteilung für die Wahrscheinlichkeiten der gemessenen Bitfehler sind in Abbildung 8.5 zu finden. In den Graphen ist zu sehen, dass für die höchste Fluenz eine deutlich zu niedrige Anzahl an Ereignissen gemessen wurde. Ein Grund hierfür könnte sein, dass der Protonenstrahl im Experiment den Chip nicht optimal getroffen hat und eventuell nur ein Teil des Strahls auf den Chip auftreffen konnte. Der Chip wurde mithilfe eines Fadenkreuzes aus Lasern so am Probehalter befestigt, dass die Strahlmitte genau auf den Chip treffen sollte. In Abbildung 8.6 ist der Ausgang des Kollimators und das Fadenkreuz zur Justage zu sehen sowie der Rahmen, an dem das Target befestigt wird. Trotzdem kann eine eventuelle Verschiebung des Strahls nicht der Fehler sein, denn der DCS-Chip wurde während der Messungen nicht ausgetauscht oder neu justiert, so dass für alle drei Messungen bei jeder Fluenz zu wenig Bitfehler hätten detektiert werden müssen. Eine weitere Fehlerquelle zeigte auch die Methode zur Bestimmung der Einzelfehlereffekte, da die Werte vieler Register stark fluktuierten und einige Register deshalb aus der Auswertung ausgeschlossen werden

[6)] Es wird dieser Wert für den Wirkungsquerschnitt angenommen, da dieser dem im selben Design angefertigten FE-I4-Chips entspricht.

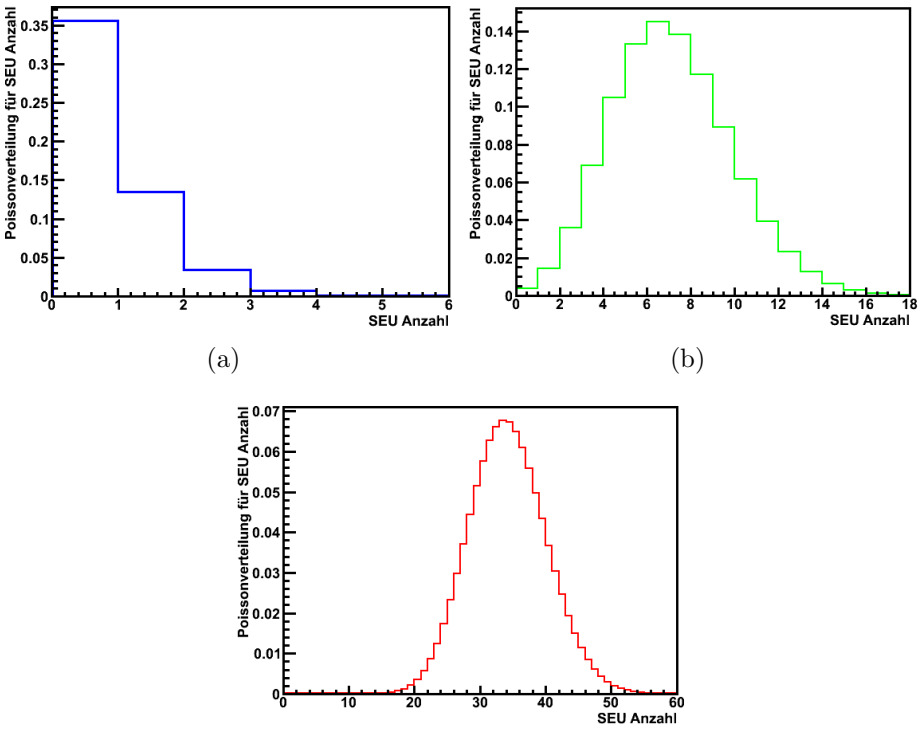


Abbildung 8.5: Poissonverteilung für die Zahl der auftretenden Einzelfehlereffekte für die Fluenzen (a), (b) und (c)

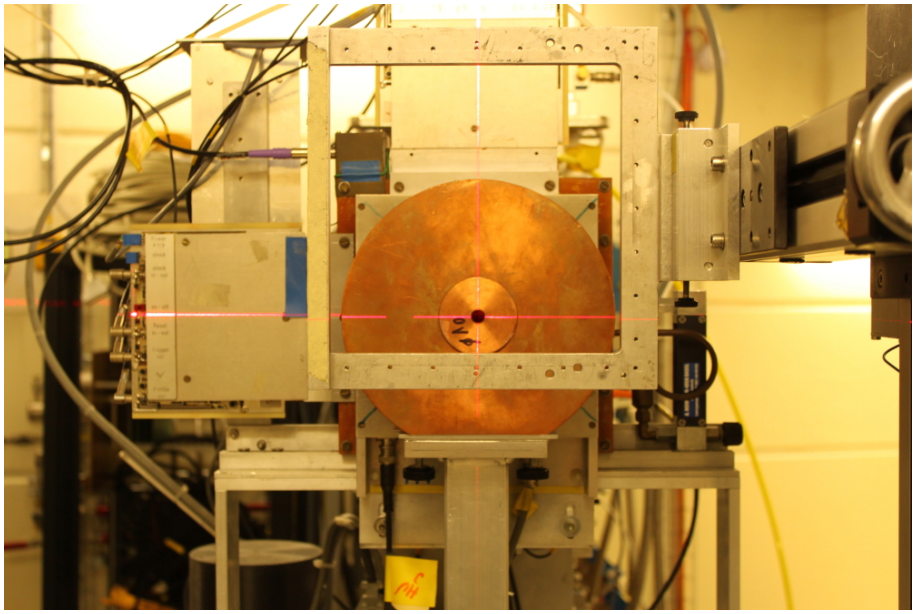


Abbildung 8.6: Fadenkreuz zur Justage des Targets in der Strahlmitte

mussten. Aber auch hier gilt, dass zwar die Auswertung sehr aufwendig war, die echten Bitfehler zu identifizieren, dennoch liegen die gemessenen Bitfehler für die beiden niedrigen Fluenzen im erwarteten Bereich und nur der Wert der höchsten Fluenz ist zu gering. Eine weitere Fehlerquelle ist die Fluenzmessung. Diese wird bei einem Strahlstrom von 0,02 nA kalibriert, während die höchste Fluenz durch den maximalen Strahlstrom von 200 nA erzeugt wird. Sehr wahrscheinlich ist es, dass für höhere Strahlströme der Wert für die Fluenz deutlich zu gering gemessen wurde. Diese Annahme bestätigt sich auch in der Bestrahlung des DCS-Controllers aus Kapitel 9.

Für eine Wiederholung der Bestimmung des Wirkungsquerschnitts sollten auf jeden Fall deutlich mehr Register zur Bestimmung der Bitfehler eingesetzt werden, um eine höhere Statistik zu erhalten. Außerdem sollte eine andere Methode gewählt werden, damit die Auswertung automatisiert werden kann. Solch eine Auswertung ist weniger fehleranfällig und zudem nicht so zeitintensiv.

8.3 Verhalten der TMR-abgesicherten I2C-HC-Kommunikation im Strahl

Derselbe DCS-Chip wurde für die Messung des SEU-Wirkungsquerschnitts und für die Bestimmung der Fehlerrate während der DCS-Kommunikation verwendet. Aus diesem Grund ist davon auszugehen, dass die Zahl der Einzelfehlereffekte in etwa dieselbe bei beiden Messungen ist — leider eine sehr geringe Zahl an zu erwarteten Bitfehlern.

Für die Bestimmung der Funktionalität der DCS-Kommunikation zum Detektor unter Bestrahlung wurde der DCS-Chip über die gesamte Strahlzeit betrieben. Ein möglicher Fehler der Kommunikation kann durch Strahlungseffekte in den Chipregistern oder auf der Busleitung auftreten und hat verschiedene Folgen. Ein Fehler auf der Busleitung verfälscht die gerade übertragenden I2C-HC-Daten. Das Hamming kodierte I2C-HC-Protokoll kann in solch einem Fall einzelne Bitfehler korrigieren und 2-Bit-Fehler detektieren. Zwei Zähler im DCS-Chip dienen dazu, die Hamming-korrigierten bzw. die detektierten Fehler zu zählen, um solche für die Bestrahlung sichtbar zu machen. Diese Zähler werden im folgenden Fehlerzähler genannt.

Einzelfehler in den Chipregistern sollten durch den TMR-Mechanismus abgefangen werden. Falls der unwahrscheinliche Fall eines Doppeltreffers in einem TMR-Register auftreten sollte, kann ein Register einen fehlerhaften Wert speichern. Dadurch kann es sein, dass der DCS-Chip eine Antwort mit fehlerhaftem Inhalt sendet oder eine falsche Prüfsumme auf den Bus gibt. Die Prüfsumme würde durch die Selbstkorrektur des Protokolls wieder korrigiert und in den Fehlerzählern sichtbar werden. Damit fehlerhafte Nachrichten vom DCS-Chip entdeckt werden können, wurden Schreibbefehle jeweils vom Peripherie-Mikrocontroller überprüft. Lesebefehle des DCS-Chips

wurden mithilfe des LabView-Programms, in dem die DCS-Kommunikation realisiert wurde, ausgewertet und dabei die empfangenen Nachrichten mit den erwarteten verglichen.

Der Ablauf der DCS-Kommunikation ist in der folgenden Aufzählung zu finden:

1. Auslesen der Fehlerzähler
2. Lesen aller DCS-Chip Register (Jeder 10. Iterationsschritt verändert die Eingangswerte am ADC und an den Latches zur Feuchtemessung.)
3. Auslesen der Fehlerzähler
4. Lesen der DCS-Chip Ausgänge
5. Auslesen der Fehlerzähler
6. Schreiben der DCS-Chip Ausgänge (Es wird geprüft, ob der Wert am Ausgang stimmt.)
7. Auslesen der Fehlerzähler
8. Lesen der DCS-Chip Ausgänge (zur Überprüfung, ob die Ausgänge korrekt geschrieben wurden)
9. Auslesen der Fehlerzähler

Es wurden zwei DCS-Chips insgesamt 5 Stunden lang bestrahlt. Die oben beschriebene Kommunikations-Reihenfolge wurde pro Strahlstunde jeweils etwa 450 Mal durchlaufen.

Chip Nr.	Fluenz $[\frac{p}{cm^2}]$	Dosis [rad]	Fluss $[\frac{p}{cm^2 \cdot s}]$	erwartete SEU-Rate
7	$3{,}58 \cdot 10^9$	$3{,}36 \cdot 10^2$	10^6	0
7	$3{,}53 \cdot 10^{11}$	$1{,}04 \cdot 10^5$	10^8	5
7	$1{,}61 \cdot 10^{12}$	$2{,}63 \cdot 10^5$	$5 \cdot 10^8$	7
22	$3{,}53 \cdot 10^{11}$	$6{,}62 \cdot 10^4$	10^8	5
22	$1{,}61 \cdot 10^{12}$	$3{,}75 \cdot 10^5$	$5 \cdot 10^8$	7

Tabelle 8.4: Aufzählung der Fluenzen, Dosen und Flüsse zweier DCS-Chips unter Bestrahlung. Es wurde jeweils 1 Stunde bestrahlt. Die erwartete SEU-Rate ergibt sich aus den *gemessenen* Werten aus Kapitel 8.2.

Während der gesamten Strahlzeit trat kein Fehler in der DCS-Kommunikation auf: Weder zählten die Fehlerzähler einen Fehler, noch wurde eine DCS-Nachricht mit fehlerhaftem Inhalt übertragen.

Es lässt sich somit sagen, dass der TMR-Mechanismus höchstwahrscheinlich funktionsfähig ist. Durch die geringe Bitfehlerrate ist die Wahrscheinlichkeit allerdings hoch, dass in der kritischen Zeit $t_{kritisch} = 10^{-4}\,\mathrm{s}$ des I2C-HC-Protokolls keine Register getroffen wurden. Ob die TMR-Absicherung allerdings für die HL-LHC-Strahlungsumgebung ausreicht, kann weiterhin nur theoretisch bestimmt werden.

8.4 Leistungsverbrauch des DCS Chips

Um Aussagen über kumulative Strahlungseffekte machen zu können, wurde der Core-Strom des DCS-Chips unter Bestrahlung gemessen.

Der DCS-Chip wird durch zwei unterschiedliche Spannungen versorgt: Der Core wird mit $V_{Core} = 1{,}25\,\mathrm{V}$, der Padframe mit $V_{Padframe} = 3{,}3\,\mathrm{V}$ versorgt. Im Durchschnitt verbraucht der DCS-Chip während des Betriebs im Labor einen Core-Strom von $I_{Core} \approx 40\,\mu\mathrm{A}$ und $I_{Padframe} \approx 90\,\mu\mathrm{A}$.

Während der gesamten Bestrahlung und unabhängig vom Betriebsmodus wurden Strom- und Spannungsmessungen des DCS-Chips durchgeführt. Die Werte wurden mithilfe von Keithley-Messgeräten automatisiert durch ein LabView-Programm ausgelesen, welches zusammen mit L. Püllen entwickelt wurde. In Abbildung 8.2 sind die Keithley-Messgeräte nicht eingezeichnet, nur die Agilent Power Supplies sind zu sehen. Über eine externe Schaltung wurden Strom- und Spannungsinformationen direkt auf den Versorgungsleitungen zum Chip abgegriffen. Die hier vorgestellten Ergebnisse stammen alle aus dem Betriebsmodus der normalen DCS-Kommunikation.

Die folgenden Abbildungen zeigen jeweils den verbrauchten Core-Strom des DCS-Chips, die Core-Spannung sowie die Temperatur am Probenhalter des Aufbaus im Strahl. Die Messungen für Strom und Spannung wurden parallel auch für den Referenzaufbau durchgeführt. Eine Temperaturmessung direkt am Chip wurde nicht durchgeführt. Es wurden die Umgebungstemperaturen mithilfe von Temperatursensoren gemessen.

In Abbildung 8.7 ist das Verhalten des Core-Stroms (a), der Core-Spannung (b) und der Umgebungstemperatur (c) zu sehen. Hier wurde derselbe DCS-Chip mit einer Fluenz von $1{,}61 \cdot 10^{12}\,\frac{p}{cm^2}$ bestrahlt, der während dieser Messung eine Dosis von $D_{Chip7} = 0{,}16\,\mathrm{MRad}$ und eine Gesamtdosis von $D_{GesamtChip7} = 2{,}6\,\mathrm{MRad}$ erhalten hat. Hierbei ist der Anstieg des Core-Stroms von $I_{CoreStartChip7} = 0{,}04\,\mathrm{mA}$ auf $I_{CoreMaxChip7} = 1{,}81\,\mathrm{mA}$ zu sehen. Nach $\mathrm{t} = 3800\,\mathrm{s}$ wurde der Strahl abgeschaltet, in der Abbildung 8.7 durch einen senkrechten Strich gezeigt. Im Graphen sieht man an dieser Stelle ein Maximum im Stromverbrauch, welcher nach Abschalten des Strahls direkt wieder abfällt. Der Verbrauch des Core-Stroms geht nach kurzer Zeit wieder auf den Ausgangswert von $I_{CoreStartChip7} = 0{,}04\,\mathrm{mA}$ zurück, was nicht im Graphen zu sehen ist, aber absolut gemessen werden konnte. Die Umgebungs-

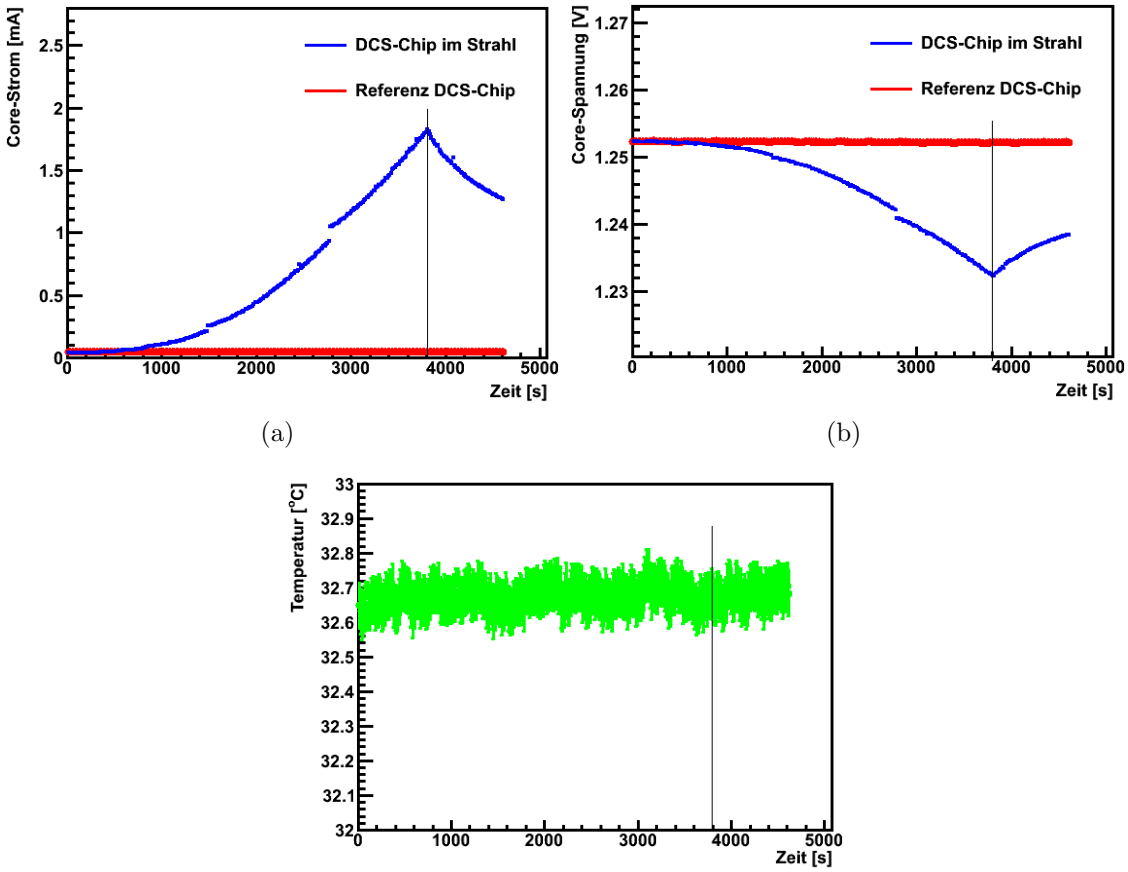


Abbildung 8.7: (a) Messung des Core-Stroms, (b) Messung der Core-Spannung und (c) Messung der Umgebungstemperatur des DCS-Chips Nr. 7, bestrahlt mit einer Fluenz von $F = 1{,}6 \cdot 10^{12} \frac{p}{cm^2}$. Die senkrechte schwarze Linie gibt den Zeitpunkt an, an dem der Strahl abgeschaltet wurde.

temperatur schwankt leicht um den Wert von etwa T = 32,65 °C, ist aber nicht mit dem Strahl korreliert.

In einer weiteren Messung wurde ein zweiter DCS-Chip insgesamt 2 Stunden bei maximaler Fluenz bestrahlt. Dabei erhielt der Chip pro Stunde eine Dosis von D = 0,16 MRad. Nach der ersten Stunde Strahlzeit stiegt der Core-Strom von ursprünglich $I_{CoreStartChip22} = 0{,}04\,\text{mA}$ auf $I_{CoreMaxChip22} = 1{,}30\,\text{mA}$ an. Durch die direkt im Abschluss erfolgende Bestrahlung konnte sich der Strom nicht wieder normalisieren und startet deshalb nicht am Ausgangswert der Messung, sondern bei etwa 1 mA und steigt während der Bestrahlung auf $I_{MaxChip22} = 5{,}16\,\text{mA}$ an. Dieses Verhalten ist in Abbildung 8.8.a zu sehen. In den Abbildungen 8.8.b und c sind jeweils der DCS-Chip Stromverbrauch und die Umgebungstemperatur des DCS-Chips aufgetragen. Nach Abschalten des Strahls, in den Abbildungen durch die senkrechte

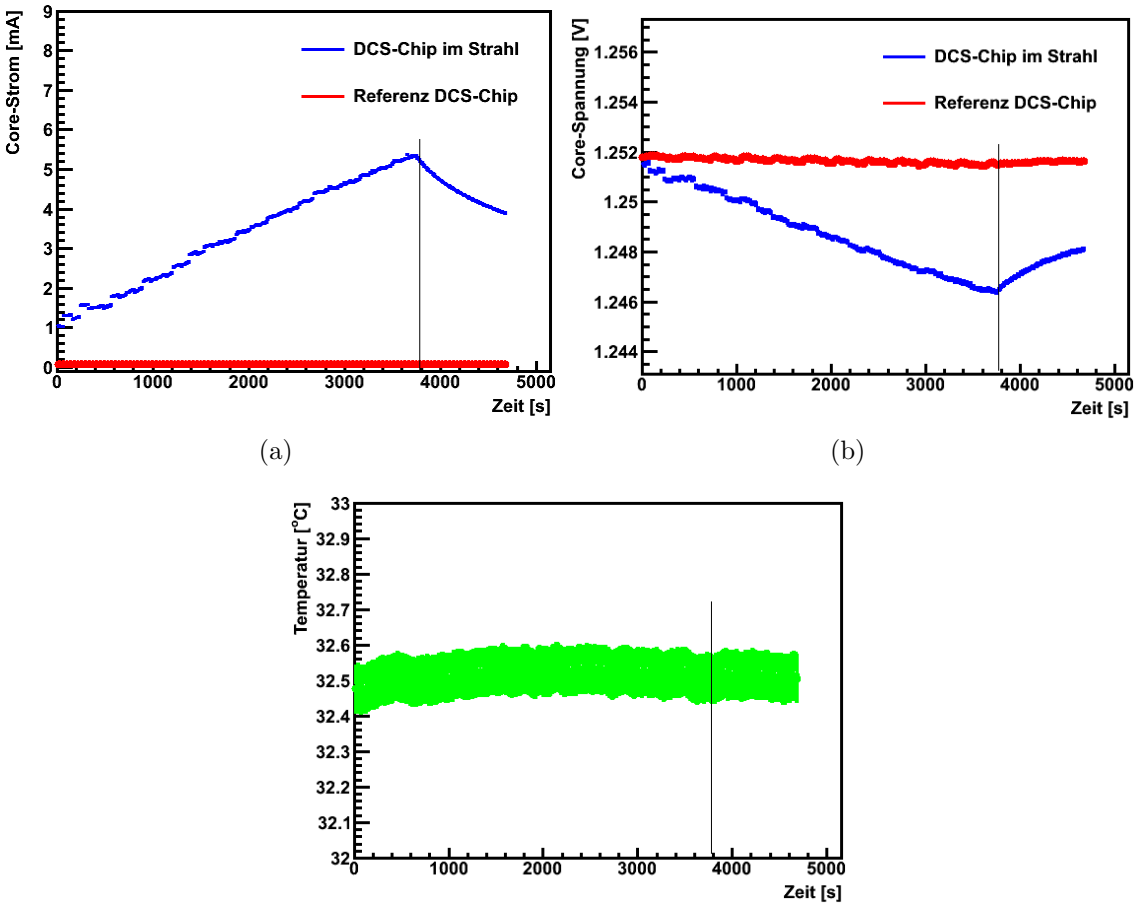


Abbildung 8.8: (a) Messung des Core-Stroms, (b) Messung der Core-Spannung und (c) Messung der Umgebungstemperatur des DCS-Chips Nr. 22, bestrahlt mit einer Fluenz von $F = 1{,}6 \cdot 10^{12} \frac{p}{cm^2}$, nachdem schon vorher eine Stunde mit derselben Fluenz bestrahlt worden ist. Aus diesem Grund startet in (a) der Core-Strom bei einem erhöhten Wert. Die senkrechte schwarze Linie gibt den Zeitpunkt an, an dem der Strahl abgeschaltet wurde.

schwarze Linie markiert, sinkt der Core-Strom wieder ab und erreicht $I_{CoreStartChip22}$, was allerdings im Graphen nicht zu sehen ist.

Die Strommessung zeigt ganz allgemein jeweils einen Anstieg des Core-Stroms unter Bestrahlung. Diese entsteht durch Ladungsansammlung in den Isolationsregionen der CMOS-Komponenten, die Leckströme erzeugen und dadurch den Stromverbrauch erhöhen, wie in Kapitel 4.2.3 beschrieben ist.

Es wurden keine Latch-Up-Effekte während der Bestrahlung oder andere permanente Schädigungen beobachtet. Der Anstieg des Stromverbrauchs im Core-Strom fiel

jeweils nach einiger Zeit ohne Strahl wieder auf den Ausgangswert zurück. Bei den erreichten Dosen entsprechen diese Beobachtungen auch der Erwartung.

8.5 Zusammenfassung

Mit der Bestrahlung des DCS-Chips sollte hauptsächlich die Strahlenfestigkeit des TMR-abgesicherten I2C-HC-Protokolls bestimmt werden. Wie erwartet wurde auch kein einziger Protokollfehler detektiert, so dass von einem funktionsfähigen TMR-Mechanismus ausgegangen werden kann. Allerdings wurde eine deutlich geringere SEU-Rate gemessen, als erwartet wurde, so dass der TMR-Mechanismus eventuell gar nicht ausgelöst wurde.

Des Weiteren konnte ein SEU-Wirkungsquerschnitt für den verwendeten 130 nm Prozess gemessen werden. Dieser hat zwar aufgrund der geringen Statistik und Unsicherheiten bei der Fluenzmessung einen großen Fehler, liegt dennoch im erwarteten Bereich anderer Designs, die im selben Prozess entwickelt wurden.

Zuletzt wurden erste Dosiseffekte, mit Dosen deutlich unterhalb der HL-LHC-Gesamtdosis, bestimmt. Diese Effekte wurden nur zur Vollständigkeit erwähnt, die Messungen der Einzelfehlereffekte standen im Vordergrund.

Kapitel 9

Validierung der Kommunikation zum Detektor unter Bestrahlung

Um die theoretische Vorhersage für die Strahlenfestigkeit des TMR-abgesicherten CAN-Protokolls im DCS-Controller zu prüfen, wurde dieser in einer zweiten Bestrahlungskampagne im Jahr 2012 an derselben Einrichtung wie zuvor der DCS-Chip mit Protonen bestrahlt. Neben dem TMR-abgesicherten Design wurde auch der nicht abgesicherte DCS-Controller bestrahlt, damit die Unterschiede der beiden Designs herausgestellt werden konnten, um diese mit den theoretischen Vorhersagen zu vergleichen.

Des Weiteren wurde ein Schieberegister bestrahlt, um den SEU-Wirkungsquerschnitt für den verwendeten 130 nm Prozess erneut mit einer besseren Statistik zu bestimmen.

Anhand der Messungen des Leistungsverbrauchs der beiden DCS-Controller können Dosiseffekte in den Chips sichtbar gemacht werden. Das Verständnis dieser Effekte trägt auch dazu bei, die Vorhersagen für das Verhalten des DCS-Controllers im späteren Betrieb bei HL-LHC zu verbessern und den 130 nm Prozess für den Detektorbetrieb zu validieren.

9.1 Das Bestrahlungsprogramm

Die Bestrahlung der unterschiedlichen Komponenten am PSI erfolgte mit verschiedenen Flüssen und Protonenergien. Die Breite des Strahls wurde vom PSI mit $2 \times 2\,\mathrm{cm}^2$ flach verteilt angegeben. Es wurden insgesamt vier verschiedene Prototypen bestrahlt:

- der DCS-Controller mit und ohne TMR Absicherung zur Bestimmung der Strahlenfestigkeit des CAN-Protokolls

- das Schieberegister zur Messung des SEU-Wirkungsquerschnittes

- der V_{Ref}-Chip, zur allgemeinen Bestimmung seines Verhaltens unter Bestrahlung

Der V_{Ref}-Chip stellt einen ersten Prototypen für eine Referenzspannungsquelle dar, die von L. Püllen entwickelt wurde. Die Referenzspannungsquelle wird im Detektor-Kontrollsystem für einen späteren ADC im DCS-Chip benötigt. Da der V_{Ref}-Chip nicht Teil dieser Arbeit ist, wird auf die Messung und die Ergebnisse dieses Chips nicht genauer eingegangen.

Das genaue Bestrahlungsprogramm ist in Tabelle 9.1 zu finden.

Teilchenfluss $[\frac{p}{cm^2 \cdot s}]$	Protonenergie [MeV]	Dosis [MRad]	Chiptyp (SR = Schieberegister)
$1{,}07 \cdot 10^9$	99,7	0,34	SR 1, V_{Ref} 6
$1{,}07 \cdot 10^9$	99,7	0,34	SR 1, V_{Ref} 6
$4{,}25 \cdot 10^9$	99,7	1,97	SR 1, V_{Ref} 6
$5{,}48 \cdot 10^8$	99,7	0,17	SR 1, V_{Ref} 6
$1{,}84 \cdot 10^8$	99,7	0,04	SR 1, V_{Ref} 6
$5{,}48 \cdot 10^8$	99,7	0,17	CoFee2_noTMR 3, V_{Ref} 4
$1{,}07 \cdot 10^9$	99,7	0,34	CoFee2_noTMR 3, V_{Ref} 4
$1{,}07 \cdot 10^9$	99,7	0,34	CoFee2_noTMR 3, V_{Ref} 4
$4{,}25 \cdot 10^9$	99,7	1,31	CoFee2_noTMR 3, V_{Ref} 4
$4{,}25 \cdot 10^9$	99,7	1,31	CoFee2_noTMR 3, V_{Ref} 4
$1{,}07 \cdot 10^9$	99,7	0,34	CoFee2_TMR 5, V_{Ref} 4
$4{,}25 \cdot 10^9$	99,7	1,29	CoFee2_TMR 5, V_{Ref} 4
$4{,}25 \cdot 10^9$	99,7	1,29	CoFee2_TMR 5, V_{Ref} 4
$4{,}25 \cdot 10^9$	99,7	1,29	CoFee2_TMR 5, V_{Ref} 4
$4{,}25 \cdot 10^9$	30,7	0,08	SR 1, V_{Ref} 4
$4{,}25 \cdot 10^9$	51,7	0,16	SR 1, V_{Ref} 4
$4{,}25 \cdot 10^9$	70,4	0,21	SR 1, V_{Ref} 4
$4{,}25 \cdot 10^9$	99,7	0,69	SR 1, V_{Ref} 4

Tabelle 9.1: Das Bestrahlungsprogramm mit Teilchenfluss, Protonenergie und Dosis. Zudem wird aufgelistet welcher Chip bestrahlt wurde. Die Strahlzeit betrug jeweils etwa eine Stunde, für die letzten vier Messungen jeweils eine halbe Stunde. Die separate Bestrahlung von CoFee2_noTMR erfolgte mit dem Betriebsprogramm von CoFee_TMR.

Aufbau der Bestrahlung des DCS-Controllers

Für die Bestrahlung beider DCS-Controller-Typen (mit und ohne TMR-Absicherung) wurde ein kompletter DCS-Pfad aufgebaut, bestehend aus DCS-PC, DCS-Controller und DCS-Chip. Um auftretende Fehler eindeutig Strahlenschädigungen zuweisen zu können, wurde gleichzeitig ein Referenzaufbau betrieben, der sich komplett in der Kontroll-Baracke befand. In Abbildung 9.1 ist der gesamte Aufbau für die Bestrahlung des DCS-Controllers zu sehen.

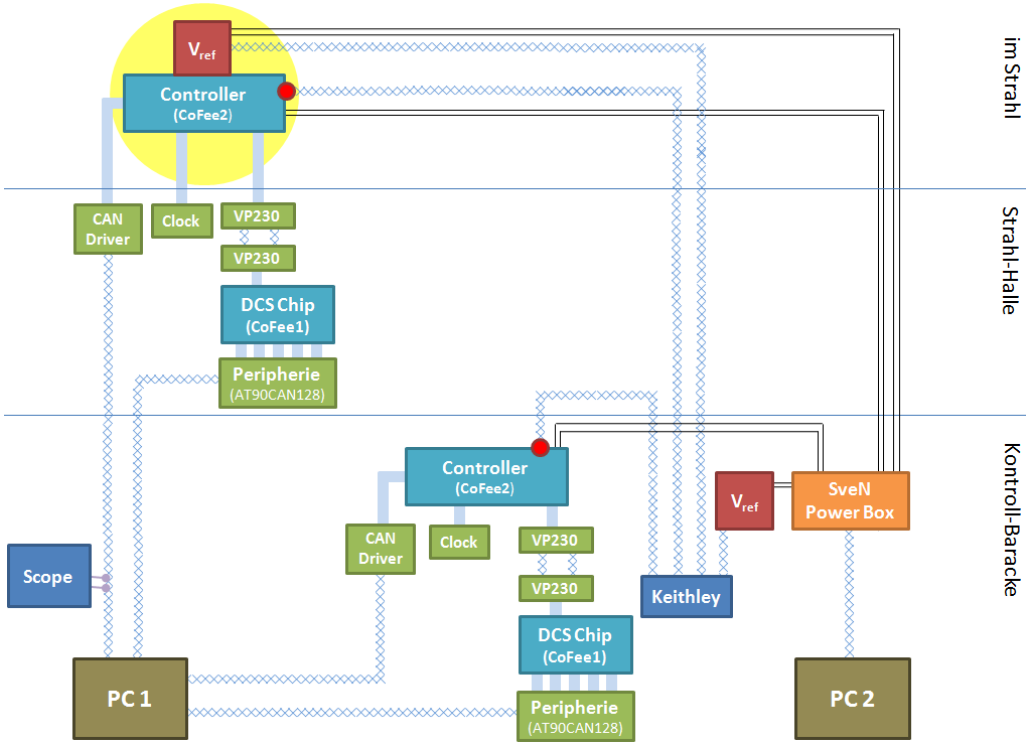


Abbildung 9.1: Aufbau zur Bestrahlung des DCS-Controllers mit eingezeichnetem Referenzaufbau

Im Strahl befinden sich DCS-Controller und V_{Ref}-Chip, die beide rückwärtig hintereinander im Strahl lokalisiert sind. Dabei wird der DCS-Controller primär bestrahlt. Auf beiden DCS-Controllern im Strahl wurden Temperatursensoren angebracht, die die Chiptemperatur sowie die Umgebungstemperatur maßen. Die Auslese der Temperatursensoren erfolgte mit dem Keithley-Messgerät. Die Bus-Treiber und der externe Taktgeber wurden mit Flachbandkabeln möglichst weit entfernt vom Strahl angebracht, um sie vor Streu-Strahlung zu schützen. Der DCS-Chip und der Peripherie-Mikrocontroller befanden sich ca. 2 m entfernt vom DCS-Controller.

Der Peripherie-Mikrocontroller ermöglichte das Schreiben der DCS-Chip Eingänge, so dass dieser unterschiedliche Werte auslesen konnte. Eine weitere Aufgabe des Peripherie-Mikrocontroller bestand darin, die DCS-Chip Ausgänge auszulesen.

Versorgt wurden Aufbau und Referenz mit der SVEN-Box[1)], die neben der Spannungsversorgung auch die Messung der Spannungen und Ströme durchführte. Die SVEN-Box wurde im Rahmen der Bachelor-Arbeit von Sven Römer [32] entwickelt und kam erstmals bei dieser Bestrahlung zum Einsatz.

Die Länge der CAN-Bus Kabel entsprechen der Entfernung zwischen Strahlhalle und Kontroll-Baracke und sind 30 m lang. Auch die Kabel zur SVEN-Box hatten für beide Aufbauten eine Länge von 30 m.

Mit diesem Aufbau wurde das Verhalten der CAN-Kommunikation des DCS-Controllers unter Bestrahlung gemessen, das in Kapitel 9.3 dargestellt ist.

Aufbau der Bestrahlung des Schieberegisters

Durch die gleichzeitige Bestrahlung von vier Schieberegistern (mit jeweils 1500 Registern), die jeweils im 130 nm Prozess hergestellt worden sind, wurde eine maximale Zahl an Registern bereitgestellt, die zur Messung von Einzelbiteffekten zur Verfügung stehen. Die vier Schieberegister wurden, wie in Abbildung 9.2 zu sehen, zu einem Viereck auf die Trägerplatine geklebt und gebondet. Es stehen somit insgesamt 6000 Register für die Messung des SEU-Wirkungsquerschnitts zur Verfügung.

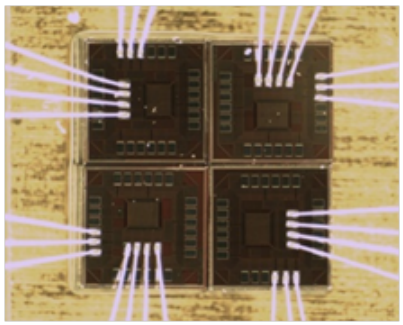


Abbildung 9.2: Anordnung der vier Schieberegister im Strahl

Auch für die Bestrahlung des Schieberegisters gab es einen Referenzaufbau in der Kontroll-Baracke, um sicherzustellen, dass Bitfehler aufgrund von Strahlung und nicht aufgrund anderer Effekte induziert wurden. Wie auch bei der Bestrahlung des Controllers wird der V_{Ref}-Chip wieder zusätzlich, aber rückwärtig, in den Strahl gebracht.

In Abbildung 9.3 ist der gesamte Aufbau sowie der Referenzaufbau zu sehen. Die Ansteuerung und Auslese des Schieberegisters erfolgte via CAN-Kommunikation vom Computer zum Peripherie-Mikrocontroller. Dieser sendete für jede Messung zunächst das Bitmuster „001100110011“ in jedes der vier Schieberegister, dann wurden die Schieberegister für eine 2-minütige Messzeit nicht angesteuert, während sie bestrahlt

[1)] **S**erielle Strom**V**ersorgung für **E**xperimentelle **N**utzung

wurden. Danach wurde das in den Schieberegistern vorhandene Bitmuster vom Mikrocontroller ausgelesen, der die Daten per CAN-Kommunikation an den Computer zurücksendete. Die Versorgung des Schieberegisters erfolgte durch die SVEN-Box. Die CAN- sowie die Versorgungskabel hatten für Aufbau und Referenz die gleichen Längen. Auch bei diesem Aufbau wurden die Chip- und die Umgebungstemperatur des Schieberegisters mithilfe von Temperatursensoren vom Keithley gemessen.

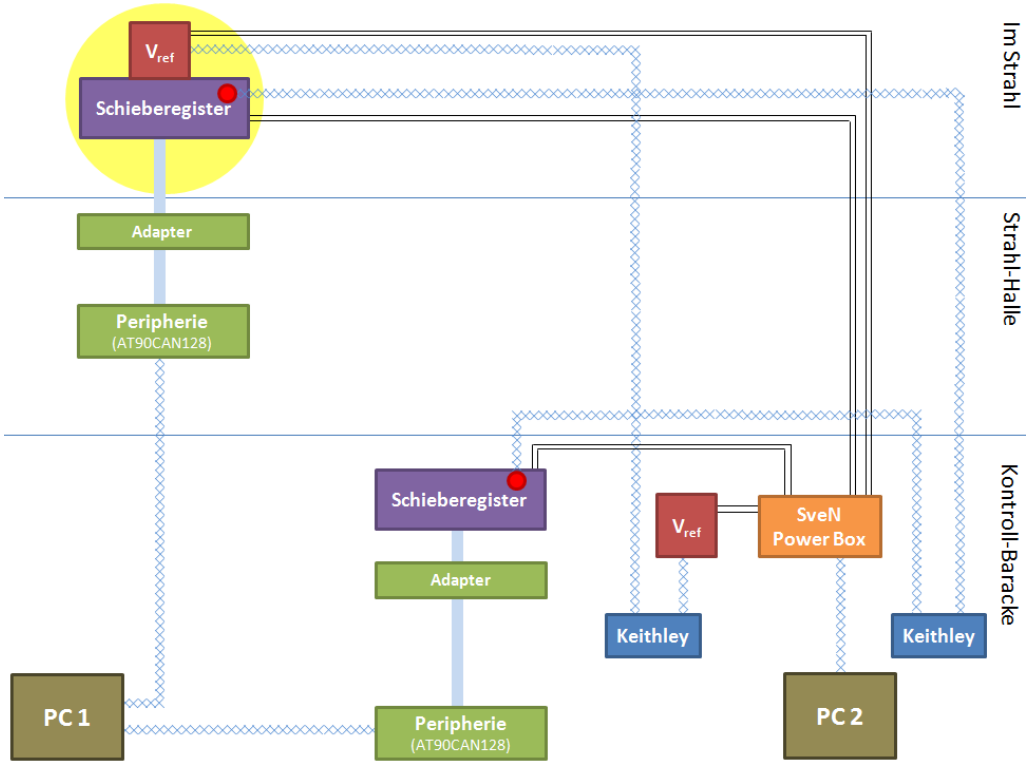


Abbildung 9.3: Aufbau zur Bestrahlung des Schieberegisters mit zugehörigem Referenzaufbau in der Kontroll-Baracke

9.2 Bestimmung des SEU-Wirkungsquerschnitts

Der SEU-Wirkungsquerschnitt gibt Aussagen über das Verhalten gegenüber Einzelfehlereffekten des Designs in einer Strahlungsumgebung. In Kapitel 4.2.2 wurden die erwarteten Wirkungsquerschnitte angegeben. Gemessen wurde der Wirkungsquerschnitt bei der Bestrahlung des DCS-Chips aus Kapitel 8 mit einer großen Unsicherheit in Gleichung 8.1 zu $\sigma = (4{,}91 \pm 4{,}15)\,10^{-14}\,\frac{cm^2}{bit}$. Mit der großen Zahl an Registern sollen mehr SEUs detektiert werden, so dass sich der Fehler der Messung reduziert.

9.2.1 Bestimmung des SEU-Wirkungsquerschnitts

Gemessen wurde der SEU-Wirkungsquerschnitt für vier unterschiedliche Flüsse bei einer konstanten Protonenergie von E = 99,7 MeV. Bei jeder Messung wurde eine Verweildauer des Bitmusters im Schieberegister von 2 Minuten eingestellt. Es wurde pro Durchlauf eine Stunde bestrahlt, so dass jeweils 30 Messungen durchgeführt werden konnten.

Die gemessenen SEU-Fehler in der Summe aller vier Schieberegister in Abhängigkeit von der Fluenz sind in Tabelle 9.2 aufgetragen.

Fluenz $[\frac{Protonen}{cm^2}]$	Summe der Bitfehler	Bitfehler SR 1	Bitfehler SR 2	Bitfehler SR 3	Bitfehler SR4
$0{,}5 \cdot 10^{12}$	96 ± 10	29 ± 5	24 ± 5	23 ± 5	20 ± 4
$1{,}7 \cdot 10^{12}$	448 ± 21	123 ± 11	109 ± 10	116 ± 11	100 ± 10
$3{,}3 \cdot 10^{12}$	980 ± 31	254 ± 16	243 ± 15	237 ± 15	246 ± 15
$13{,}3 \cdot 10^{12}$	6227 ± 80	1537 ± 39	1607 ± 40	1602 ± 40	1481 ± 38

Tabelle 9.2: Anzahl der detektierten Bitfehler in Abhängigkeit von der Fluenz. Neben der Gesamtzahl der SEU-Fehler werden auch die gemessenen Bitfehler pro Schieberegister (= SR) dargestellt.

In Abbildung 9.4 sind die Bitfehler in Abhängigkeit von der Fluenz graphisch dargestellt. Formel 4.4 sagt einen linearen Zusammenhang voraus.

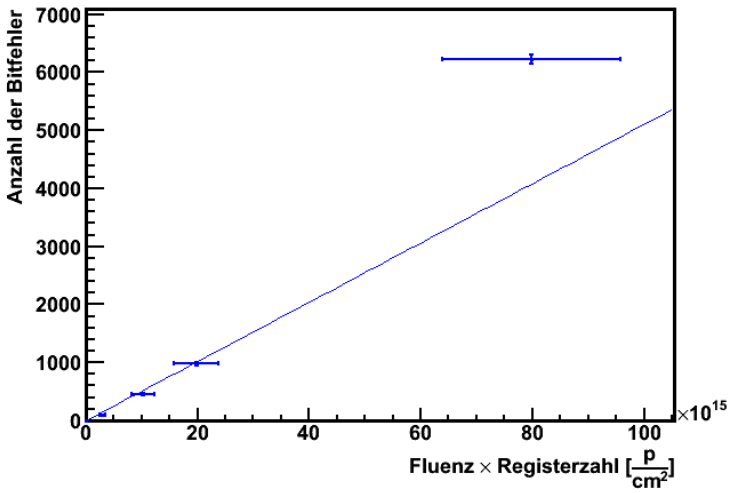


Abbildung 9.4: Die gemessenen Bitfehler in Abhängigkeit von der Fluenz multipliziert mit der Registerzahl.

Aus dem linearen Fit in Abbildung 9.4 ergibt sich der SEU-Wirkungsquerschnitt zu:

$$\sigma_{130\,nm} = (5,11 \pm 0,91) \cdot 10^{-14}\,\frac{cm^2}{bit} \tag{9.1}$$

Die Unsicherheit für die Zahl der Bitfehler ist zusammengesetzt aus statistischem und systematischem Fehler. Dabei sind im systematischen Fehler die erwarteten Bitfehler enthalten, die in der Zeit während des Schreibens und Lesens der Register auftraten und somit nicht detektiert werden konnten. Die Anzahl der möglichen Doppeltreffer wurde nicht in den systematischen Fehler eingebracht, weil diese Zahl zu klein ist.

Der Fehler für die Fluenz wird mit 20 % abgeschätzt.

Bei Betrachtung der Abbildung 9.4 ist deutlich zu erkennen, dass der Wert für die höchste Fluenz zu stark von dem linearen Zusammenhang abweicht. Es wurde entweder eine zu geringe Fluenz gemessen, oder eine zu hohe Zahl an Bitfehlern detektiert. Mit dem Referenzaufbau wurden keine Bitfehler detektiert. Somit ist es ausgeschlossen, dass zusätzliche Bitfehler, die nicht durch SEUs induziert worden sind, mit dem Strahlaufbau gemessen wurden.

Um die Fluenzmessung genauer zu untersuchen, wurde am PSI [37] eine nachträgliche Kalibration der Teilchenflüsse in Abhängigkeit vom Strahlstrom durchgeführt. Die gemessenen Teilchenflüsse werden bei einem Strahlstrom von 0,02 nA mit einem Plastik-Szintillator kalibriert. Während der Bestrahlung werden die Teilchenflüsse für Ströme bis 200 nA dann mit Ionisationskammern direkt am Strahlaustrittsfenster und hinter den Kollimatoren gemessen. Die Positionen der Ionisationskammern sind in Abbildung 9.5 gezeigt.

In Abbildung 9.6 ist die gemessene Abhängigkeit des Teilchenflusses vom Strahlstrom nach den Werten vom PSI [37] aufgetragen. In der blauen Kurve ist für die Teilchenflüsse eine Sättigung des Flusses für hohe Strahlströme zu erkennen. Laut [37] ist dies ein Problem des zu stark fokussierten Strahls. Bei einer solchen Fokussierung — die nie zuvor verwendet wurde — zeigen die Ionisationskammern eine zu hohe Belegungsdichte und messen somit einen zu kleinen Fluss. Bisher wurde in der Einrichtung nur mit geringeren Strahlströmen bis etwa 40 nA gearbeitet. Da eine lineare Abhängigkeit zwischen Strahlstrom und Teilchenfluss erwartet wird, wurde in Abbildung 9.6 eine lineare Interpolation auf Grundlage der gemessenen Werte bis zu einem Strom von 40 nA durchgeführt. Die Fitfunktion für die gemessenen Werte und für die Interpolation sind die Folgenden:

Die gemessene Abhängigkeit des Teilchenflusses y vom Strahlstrom x:

$$y = -(45707,5 \pm 4639,0)x^2 + (2,61 \pm 0,09) \cdot 10^7 \cdot x + (8,34 \pm 4,15) \cdot 10^7 \tag{9.2}$$

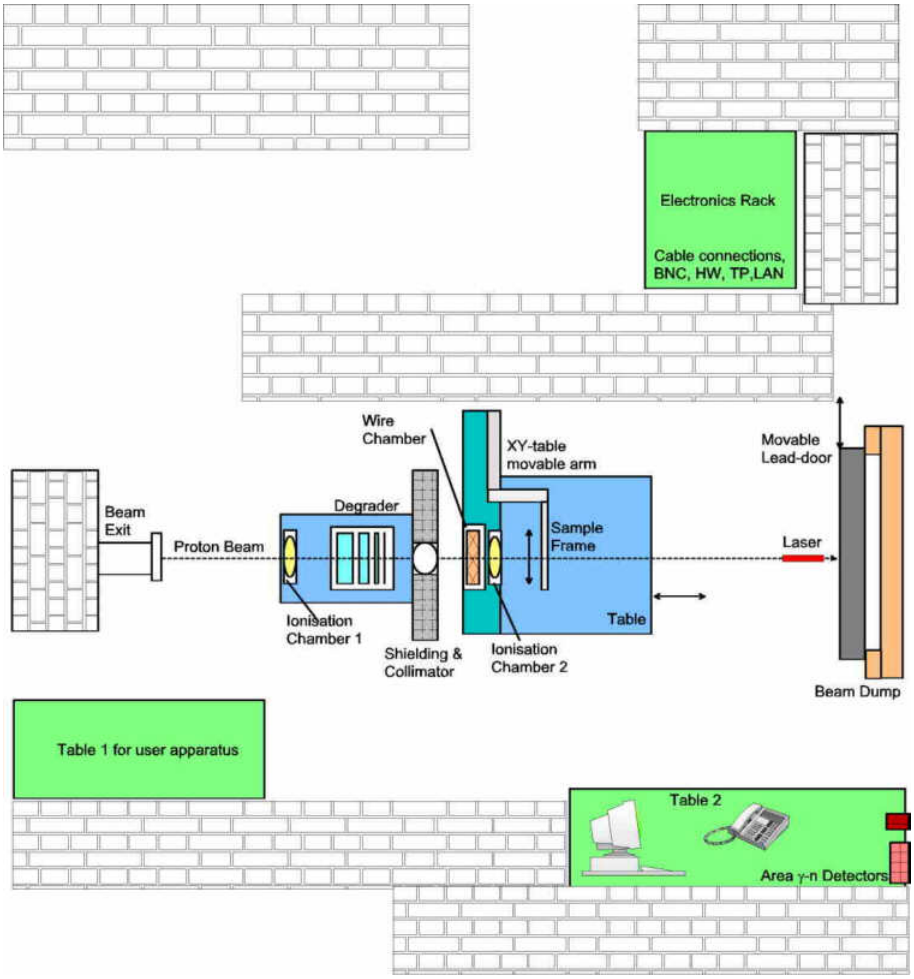


Abbildung 9.5: Schematische Zeichnung der Experimentierhalle vom PIF [36]

Lineare Interpolation:

$$y = (2,81 \pm 0,1) \cdot 10^7 \cdot x \tag{9.3}$$

Aus der Interpolation in Abbildung 9.6 ist direkt zu erkennen, dass für einen Strahlstrom von 200 nA ein deutlich höherer Fluss als der gemessene erwartet wird. Mit dem korrigierten Teilchenfluss aus der interpolierten Gerade kann ein korrigierter Wirkungsquerschnitt bestimmt werden.

Aus Abbildung 9.7 ergibt sich direkt aus der Steigung der Geraden der korrigierte SEU-Wirkungsquerschnitt zu:

$$\sigma_{korr} = (4,83 \pm 0,10) \cdot 10^{-14} \, \frac{cm^2}{bit} \tag{9.4}$$

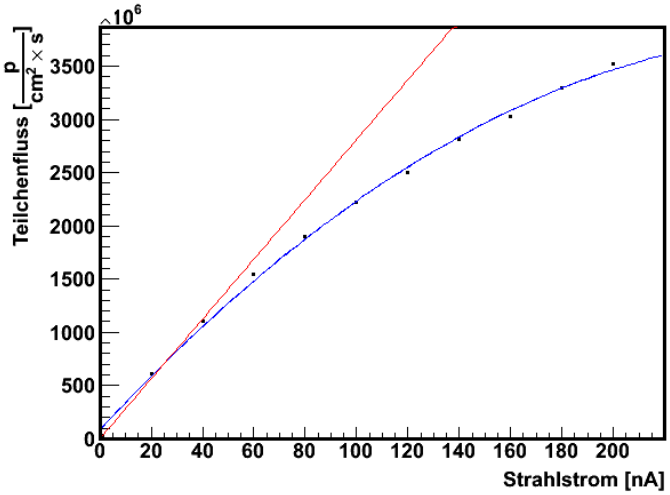


Abbildung 9.6: Gemessener Teilchenfluss in Abhängigkeit vom Strahlstrom. In rot ist die lineare Abhängigkeit des Teilchenflusses vom Strahlstrom interpoliert worden.

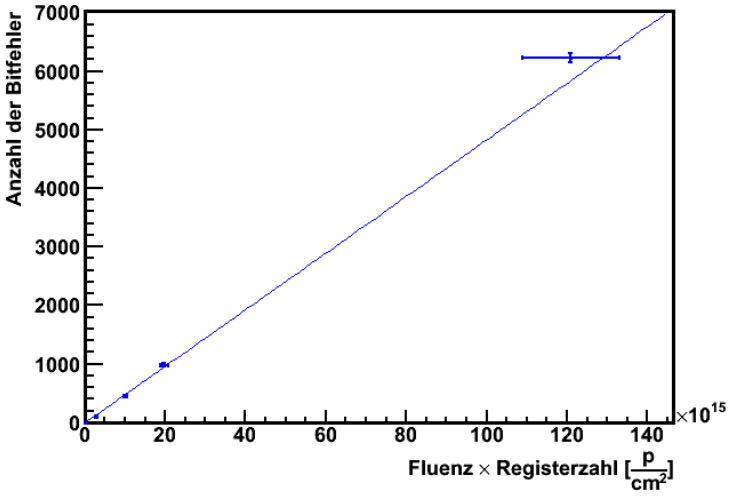


Abbildung 9.7: Die gemessenen Bitfehler in Abhängigkeit von der Fluenz multipliziert mit der Registerzahl. Für den Wert der höchsten Fluenz wurde eine Korrektur durchgeführt.

Die lineare Abhängigkeit der Bitfehler von der Fluenz ist im Graph 9.7 durch die Korrektur deutlicher geworden. Der Wirkungsquerschnitt ist im Vergleich zu 9.1 geringer, die beiden Werte stimmen im Rahmen ihres Fehlers überein. Diese Auswertung zeigt, dass durch die Sättigung der Ionisationskammern bei der Messung des Teilchenflusses ein zu geringer Wert gemessen wurde, welcher durch die angewandte Korrektur verbessert werden konnte.

Es wird im Weiteren geprüft, ob der gemessene Wert für den Wirkungsquerschnitt nicht noch einem weiteren systematischen Fehler unterliegt. Es wäre möglich, dass die vier Schieberegister nicht gleichmäßig im konstanten Bereich des Strahlprofils lokalisiert waren und deshalb verschiedene Teilchenflüsse erfuhren. Dazu ist in Abbildung 9.8 der gemessene Bitfehler für jedes Schieberegister einzeln gegen die Fluenz aufgetragen[2)]. Zur besseren Veranschaulichung wurde im unteren Graph die relative Differenz der Schieberegister zu Chip 1 aufgetragen. Die gemessenen Bitfehler der einzelnen Schieberegister sind auch in Tabelle 9.2 zu finden.

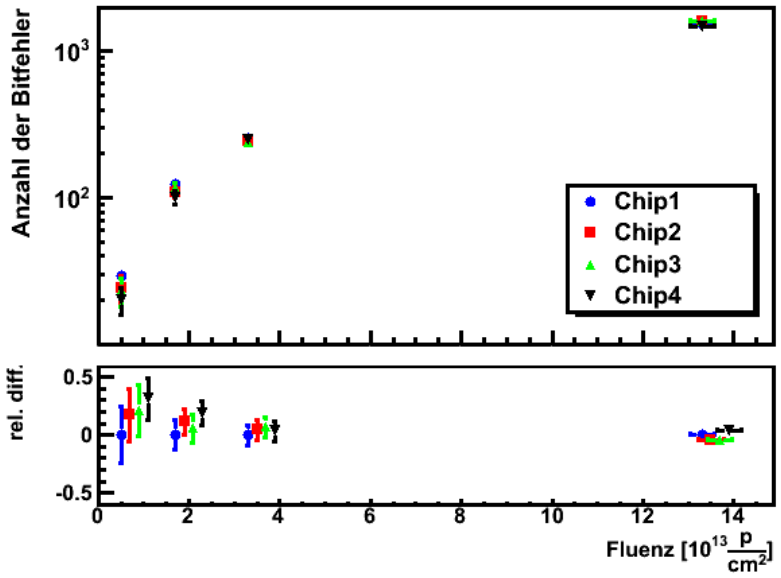


Abbildung 9.8: Bitfehler für die einzelnen Schieberegister in Abhängigkeit von der Fluenz. Im unteren Graph ist die relative Differenz zu Chip 1 aufgetragen. Für die bessere Sichtbarkeit wurden dort die Fluenzen für Chip 2 bis Chip 4 leicht erhöht dargestellt, die tatsächlichen Fluenzen entsprechen denen von Chip 1. Die im Graph verwendeten Werte stammen aus Tabelle 9.2.

Die Werte streuen leicht, stimmen aber im Rahmen des Fehlers überein. Nur für die höchste erreichte Fluenz weichen die gemessenen Bitfehler stärker voneinander ab, Chip 1 und Chip 4 streuen bei dieser Fluenz nicht mehr im Bereich des Fehlers, sondern knapp doppelt so hoch. Da die Abweichungen allerdings nicht systematisch sind und die gemessenen Bitfehler meist im Rahmen des Fehlers streuen, liegt kein weiterer systematischer Fehler des Wirkungsquerschnitts vor.

[2)] Der Wert für die höchste Fluenz wurde hier nicht korrigiert.

9.2.2 Energieabhängigkeit des SEU-Wirkungsquerschnitts

Nach der theoretischen Vorhersage aus Kapitel 4.2.2 ist anzunehmen, dass der SEU-Wirkungsquerschnitt ab einer Protonenergie von etwa 20 MeV stark ansteigt und sich relativ schnell sättigt. Für die vier verschiedenen Protonenergien wurden die Schieberegister jeweils 30 min lang mit einer Verweildauer des Bitmusters im Schieberegister von 1 min bestrahlt. Die Ergebnisse dieser Messung sind in Abbildung 9.9 gezeigt.

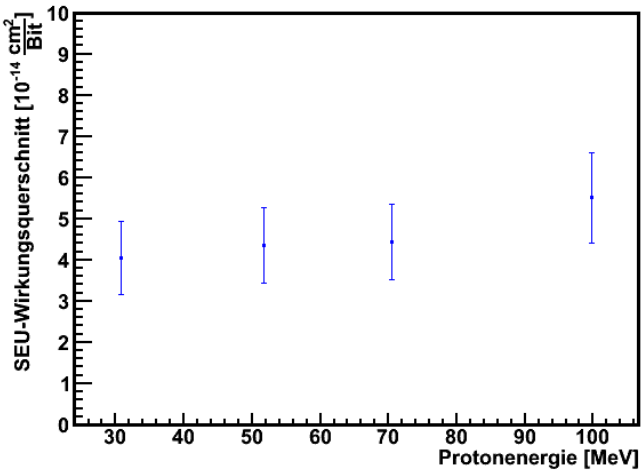


Abbildung 9.9: Abhängigkeit des SEU-Wirkungsquerschnitts von der Protonenergie

Für den Wert bei einer Energie von 99,7 MeV wird der aus Gleichung 9.1 bestimmte Wirkungsquerschnitt verwendet. Die verschiedenen Wirkungsquerschnitte unterscheiden sich für die verschiedenen Protonenergien kaum. Es kann also davon ausgegangen werden, dass sich die Sättigung des Wirkungsquerschnitts eingestellt hat. Dies bestätigt den in 9.2.1 gemessenen Wirkungsquerschnitt bei der Protonenergie von 99,7 MeV, der folglich für beliebige Protonenergien gültig ist.

9.3 Bestimmung der Strahlenhärte der CAN-Kommunikation

Zur Bestimmung der Strahlenhärte der CAN-Kommunikation wurden sowohl CoFee2_noTMR als auch CoFee2_TMR bestrahlt. Durch die eingeschränkte Funktionsfähigkeit von CoFee2_TMR, musste dieser mit einem anderen DCS-Betriebsprogramm betrieben werden als CoFee2_noTMR.

Resultate für CoFee2_noTMR

Zunächst werden die Ergebnisse für den nicht TMR-abgesicherten DCS-Controller vorgestellt.

Im DCS-Controller sind Fehlerzähler implementiert worden, die Fehler in der CAN-Kommunikation und im I2C-HC-Master zählen. Bei der CAN-Kommunikation werden entsprechend den CAN-Spezifikationen auftretende Fehler in Empfangs- und in Senderichtung gezählt. Im I2C-HC-Master werden sowohl vom Protokoll korrigierte Fehler als auch unkorrigierbare Fehler gezählt. Durch die Fehlerzähler werden Fehler im Protokoll also direkt sichtbar.

Die Kommunikation mit dem DCS-Controller bestand hauptsächlich aus dem Schreiben der DCS-Chip-Register und dem Lesen dieser. Mit dem Peripherie-Mikrocontroller aus Abbildung 9.1 konnte das korrekte Schreiben bzw. Lesen der Register extern überprüft werden. Auf diesem Weg wurde ein weiteres Mal geprüft, ob ein Fehler im Protokoll durch die Strahlung erzeugt wurde.

CoFee2_noTMR wurde mit dem folgenden Programm der DCS-Kommunikation betrieben:

1. Zum Start eines Betriebsintervalls einmalig: Lesen der Versionsnummer des Peripherie-Mikrocontrollers und Senden eines allgemeinen Resets auf DCS-Controller und DCS-Chip

2. Lesen der Fehlerzähler

3. Lesen aller DCS-Chip-Register (Jeder 10. Iterationsschritt verändert die Eingangswerte am ADC und den Latches zur Feuchtemessung. Deshalb können die empfangenen Werte der ausgelesenen Register mit den vom Peripherie-Mikrocontroller an den DCS-Chip gesendeten Werten verglichen werden. Sollten die Werte nicht übereinstimmen wird ein Fehlersignal gesetzt und eine Aufnahme vom CAN-Bus mit dem Oszilloskop gemacht.)

4. Lesen der Fehlerzähler

5. Schreiben der DCS-Chip Ausgangsregister (Mithilfe des Peripherie-Mikrocontrollers wird geprüft, ob der Wert am Ausgang stimmt.)

6. Lesen der Fehlerzähler

7. Lesen der gerade geschriebenen DCS-Chip Ausgangsregister (Es wird geprüft, ob DCS-Controller und Peripherie-Mikrocontroller dieselben Werte auslesen.)

8. Lesen der Fehlerzähler

Dieses Programm ähnelt dem aus der Bestrahlung des DCS-Chips aus Kapitel 8.3. Bei der Erstellung ging es vor allem darum, den DCS-Controller unter späteren realistischen DCS-Bedingungen zu betreiben und ihn möglichst viele DCS-Daten auslesen zu lassen, bzw. DCS-Steuerbefehle zu senden. Die Überprüfung, ob während der CAN-Kommunikation ein verfälschter DCS-Befehl gesendet wurde, erfolgt indirekt durch die Prüfung, ob die richtigen Werte ausgelesen bzw. geschrieben wurden.

Der DCS-Controller wurde entsprechend Tabelle 9.1 vier Stunden lang mit Flüssen von $\phi = 5,48 \cdot 10^8 \frac{p}{cm^2 \cdot s}$ bis zu $\phi = 4,25 \cdot 10^9 \frac{p}{cm^2 \cdot s}$ bestrahlt, was zu einer Gesamtfluenz von

$$F_{noTMR} = 9,67 \cdot 10^{12} \frac{p}{cm^2} \tag{9.5}$$

führt. Multipliziert mit dem gemessenen SEU-Wirkungsquerschnitt aus Gleichung 9.1 und der Zahl der Register von $N_{Register} = 1492$ in dieser Zeit waren

$$N_{SEU} = 737 \tag{9.6}$$

Bitfehler zu erwarten. Der oben beschriebene Programmablauf wurde insgesamt 1539 Mal ausgeführt.

Während der Bestrahlung traten verschiedene charakteristische Fehler auf:

- Insgesamt 6-mal schlug die gesamte CAN-Kommunikation fehl und der DCS-Controller musste zurückgesetzt werden, da ein Fehler in einem zentralen Register auftrat, welcher die gesamte Empfänger- bzw. Sendestruktur zerstörte.
- Es wurden 5 CAN-Empfängerfehler gezählt, was bedeutet, dass insgesamt 5 CAN-Nachrichten auf dem Bus verfälscht wurden.
- Es wurde auch ein CAN-Sendefehler gezählt, d.h. eine zu sendende CAN-Nachricht wurde durch Bitfehler verfälscht.
- Des Weiteren wurden in den Fehlerzählern selbst mehrere einfache und doppelte Bitfehler detektiert. Doppelte Bitfehler sind in diesem Fall relativ wahrscheinlich, weil sich die Register der Fehlerzähler geometrisch direkt nebeneinander befinden.
- Es wurden in der I2C-HC-Kommunikation 5 korrigierte und ein unkorrigierbarer Fehler gezählt.

Die oben aufgelisteten Fehler zeigen sehr deutlich, dass eine zusätzliche Absicherung des DCS-Controllers dringend notwendig ist, wenn der DCS-Controller in einer

Strahlungsumgebung betrieben werden soll. Bitfehler können die komplette Kommunikation stören oder auch Sende- bzw. Empfängerfehler induzieren. Zudem ist kein Verlass auf die Fehlerzähler, weil auch diese anfällig für Bitfehler sind.

Es wurde allerdings kein einziger verfälschter DCS-Befehl detektiert, d.h. es trat keine nicht detektierbare Bitfehlerzahl innerhalb der Protokolle auf. Dies wurde in Kapitel 5.4 vom CAN-Protokoll und von der I2C-HC-Kommunikation gefordert.

Im Referenzaufbau verlief während der gesamten Strahlzeit die Kommunikation mit dem DCS-Controller fehlerfrei. Es kann also davon ausgegangen werden, dass die obigen Fehler nur anhand der Strahlungsumgebung aufgetreten sind.

Eine TMR-Absicherung in solch einer hohen Strahlungsumgebung ist also absolut notwendig. Es ließ sich keine stabile CAN-Kommunikation über einen größeren Zeitraum aufbauen und die Auslese der Fehlerzähler war nicht zuverlässig. Entsprechend Gleichung 9.6 wurden 737 Bitfehler erwartet. Dass jedoch eine viel geringere Zahl an Fehlern sichtbar geworden ist, liegt an der kritischen Zeit aus Gleichung 5.4, in der ein Bitfehler einen Protokollfehler verursacht. Die kritische Zeit ist viel geringer als die Zeit zwischen zwei CAN-Befehlen.

Resultate für CoFee2_TMR

Der TMR-abgesicherte DCS-Controller wurde vier Stunden mit Flüssen von $\phi = 1{,}07 \cdot 10^9 \frac{p}{cm^2 \cdot s}$ bis zu $\phi = 4{,}25 \cdot 10^9 \frac{p}{cm^2 \cdot s}$ bestrahlt, was zu einer Gesamtfluenz von

$$F_{TMR} = 4,98 \cdot 10^{13} \frac{p}{cm^2} \tag{9.7}$$

führt. Aus der Zahl der „verwendbaren Register“ des TMR abgesicherten DCS-Controllers mit $N_{\text{funktionsfähigeReg}} = 2236$ ergibt sich eine erwartete Rate der Bitfehler zu:

$$N_{SEUtmr} = 5690 \tag{9.8}$$

Das Betriebsprogramm ist aufgrund der eingeschränkten Funktionalität von CoFee2_TMR stark verkürzt:

1. Nur einmal zum Start eines Betriebsintervalls: Auslese der Versionsnummmer der Peripherie-Mikrocontrollers und allgemeiner Reset auf DCS-Controller und DCS-Chip.
2. Schreiben der DCS-Chip-Register.

3. Verifizieren des korrekten Schreibens mithilfe des Peripherie-Mikrocontrollers.

4. Allgemeines Zurücksetzen des DCS-Controllers und des DCS-Chips[3)].

Während der gesamten Bestrahlung trat kein einziger Fehler auf. Die DCS-Kommunikation lief die gesamte Zeit stabil. Die Fehlerzähler zählten keine Fehler und ansonsten stimmten alle Nachrichten mit den erwarteten überein. Damit kann der TMR-Mechanismus als funktionsfähig angenommen werden.

Um einen direkten Vergleich zu erhalten, wurde zusätzlich eine Stunde lang der nicht TMR abgesicherte DCS-Controller CoFee2_noTMR mit dem höchsten Fluss bestrahlt, was zu einer Fluenz von

$$F_{1h} = 1,53 \cdot 10^{13} \frac{p}{cm^2} \tag{9.9}$$

führt. Da in etwa nur die Hälfte der Register durch das Programm der Kommunikation aktiv benutzt wurde, wird mit einer Registerzahl von $N_{HalbeReg} = 746$ die erwartete Rate der Bitfehler zu

$$N_{SEU1h} = 583 \tag{9.10}$$

berechnet.

Die erwarteten Bitfehler können nur einen Fehler im DCS-Controller verursachen, wenn sie in einer kurzen Zeit von etwa 4 ms einen Treffer in den entsprechenden Registern verursachen, da nach dieser Zeit ein globaler Reset alle Register wieder zurücksetzt.

Bei der einstündigen Bestrahlung traten für den nicht TMR-abgesicherten DCS-Controller insgesamt vier Bitfehler auf. Alle Fehler führten dazu, dass die Ausgänge des DCS-Chips auf Null gesetzt wurden.

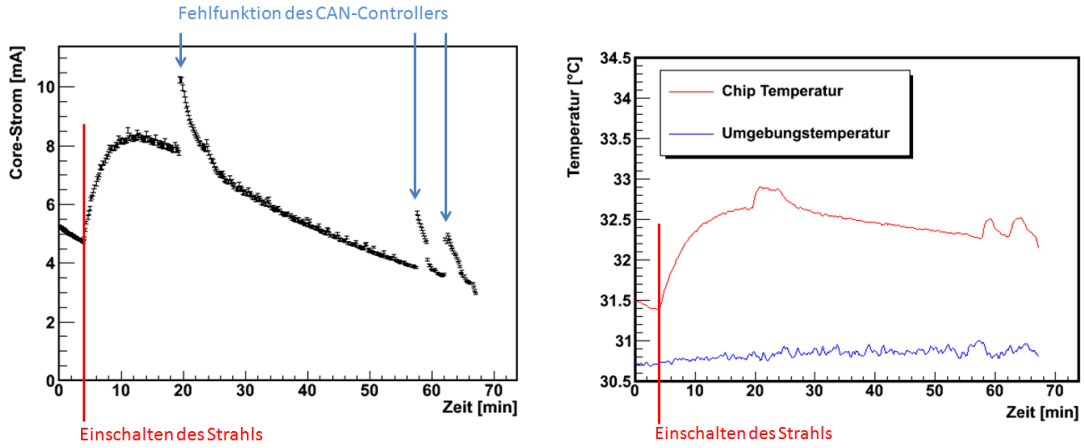
Nach dieser Messung kann also eindeutig gesagt werden, dass der TMR-Mechanismus funktioniert, da durch die viel längere Strahlzeit und höhere Zahl an Registern dort deutlich mehr Fehler aufgetreten sein müssten.

[3)] Dies ist notwendig, da der Controller eine CAN-Bestätigungsnachricht sendet, wenn ein erfolgreicher Schreibbefehl gesendet wurde. Da CoFee2_TMR keine fehlerfreien CAN-Nachrichten senden kann, wird jedes Mal der CAN-Bus blockiert, so dass der Controller erneut initialisiert werden muss.

9.4 Dosiseffekte am Beispiel des DCS-Controller-Prototypen

Anhand der Strommessung des Core-Stroms der bestrahlten DCS-Controller kann auch eine Aussage über Gesamtdosiseffekte gemacht werden. Obwohl die zu erwartende HL-LHC Strahlendosis von 570 MRad bei dieser Bestrahlung längst nicht erreicht wurde, treten ab etwa 1 MRad zwei Effekte in Erscheinung, die in den folgenden Grafiken zu sehen sind.

Stromverbrauch von CoFee2_noTMR im Strahl



(a) (b)

Abbildung 9.10: (a) Bestrahlung von CoFee2_noTMR mit höchstem Fluss, die 3 Peaks zeigen die Zeitpunkte, an denen der CAN-Bus massiv gestört wurde. (b) Das Temperaturverhalten des bestrahlten Chips CoFee2_noTMR und das Verhalten der Umgebungstemperatur.

In Abbildung 9.10.a wurde CoFee2_noTMR nach einer vorhergehenden Dosis von 0,85 MRad und einer Fluenz von F = 9,67 $\cdot$ 10^{12} $\frac{p}{cm^2}$ eine weitere Stunde mit einem Teilchenfluss von ϕ = 4,25 $\cdot$ 10^9 $\frac{p}{cm^2 \cdot s}$ bestrahlt. Wie an dem Abfall des Core-Stroms in den ersten 4 Minuten zu sehen ist, wurde der Strahl erst nach dem Start der Messung eingeschaltet, also nach diesen 4 Minuten. Der Strom steigt direkt stark an, bis ungefähr 12 Minuten Strahlzeit und bis eine Gesamtdosis des Chips von etwa 1 MRad erreicht sind. Danach sinkt der Core-Strom wieder ab. Die drei Peaks bei ca. 20, 58 und 63 Minuten stellen jeweils eine durch Bitfehler induzierte Fehlfunktion des CAN-Controllers dar, der durch sein Verhalten den Bus stört, was wiederum zu einem erhöhten Stromverbrauch im DCS-Controller führt. Nachdem der DCS-Controller zurückgesetzt wurde und die CAN-Kommunikation somit wieder korrekt

funktionierte, sank der Core-Strom unabhängig von der Bestrahlung wieder auf den Ausgangswert zurück.

Der Anstieg des Core-Stroms auf $I_{Core1MRad} = 8{,}3\,\text{mA}$ aus Abbildung 9.10.a lässt sich anhand des Effekts der eingefangenen Löcher im Quarzmaterial (der Oxidisolationsgräben an den Transistorecken), beschrieben in Kapitel 4.2.3 erklären, die Leckströme zwischen Source und Drain erzeugen. Ab etwa 1 MRad überlagert sich ein weiterer TID-Effekt, der im Oxid der Gate-Isolationsschicht stattfindet. Durch eingefangene Ladungen im Bereich der Isolationsschicht einer CMOS-Komponente wird das elektrische Feld beeinträchtigt und das Schaltverhalten des Transistors ändert sich. Die Schwellspannung erhöht sich und es fließt weniger Strom. Dieser Effekt überwiegt und reduziert somit den Core-Strom ab einem Zeitpunkt von 13 Minuten.

Nach der Bestrahlung des Chips und nach etwa 12 Stunden Ruhezeit betrug der Core-Strom $I_{CoreNachIrrad} = 175\,\mu$ A. Die Spannung am Core betrug während der gesamten Bestrahlung etwa $V_{Core} = 1{,}25\,\text{V}$. Nur an den drei Stellen, an denen der Core-Strom durch den gestörten CAN-Bus sehr hoch war, brach die Spannung kurzzeitig geringfügig ein.

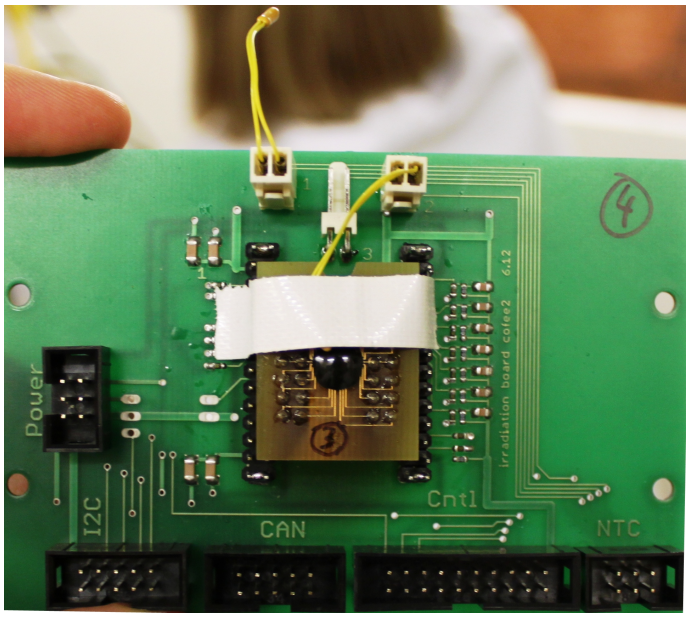
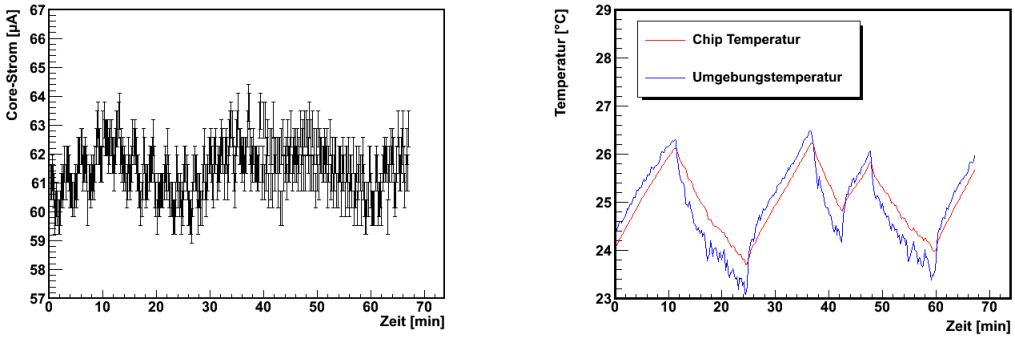


Abbildung 9.11: Temperaturmessung am Chip und der Stahlumgebung

Die Temperatur des Chips im Strahl wurde mit einem Temperatursensor gemessen, der mit Klebeband direkt auf den Rand des Pottings geklebt wurde, wie in Abbildung 9.11 gezeigt ist. Ein weiterer Temperatursensor maß die Umgebungstemperatur. Beide Temperaturen für die Bestrahlung des DCS-Controllers ohne TMR-Absicherung sind in Abbildung 9.10.b gezeigt. Der bestrahlte Chip hat ca. 2 °C Unterschied zur Umgebungstemperatur. Die Temperatur hängt von dem Stromverbrauch des Chips ab, der sich durch TID-Effekte und Störungen auf dem CAN-Bus durch SEU-Effekte

ändert. Ziemlich sicher ist allerdings, dass die gemessene Temperatur nicht der wahren Temperatur des Chips entspricht, weil diese durch den Stromverbrauch viel höher sein müsste. Das nicht sehr wärmeleitfähige Potting des DCS-Controllers schirmt die Chiptemperatur zu gut ab, so dass nur große Temperaturänderungen durch äußere Einflüsse beobachtet werden können.

Der Referenzaufbau zeigte keinerlei Veränderung des Core-Stroms während des Betriebes. Nur die bekannte Temperaturabhängigkeit des Core-Stroms ist in Abbildung 9.12 zu sehen.



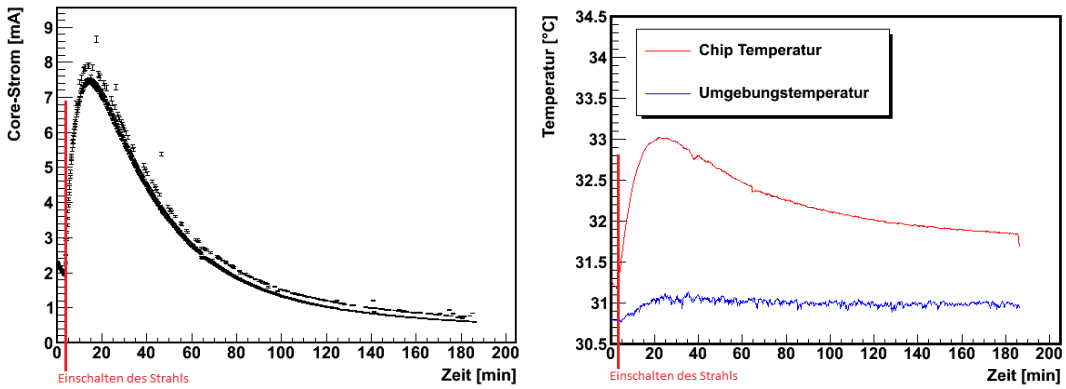
(a) (b)

Abbildung 9.12: (a) Für den Referenzchip CoFee2 noTMR sind nur die temperaturabhängigen Änderungen im Core-Strom zu sehen. (b) Die Temperatur des Chips ändert sich wegen der Klimaanlage in der Kontroll-Baracke.

Stromverbrauch von CoFee2_TMR im Strahl

Bei der Bestrahlung des DCS-Controllers mit TMR-Absicherung über drei Stunden mit dem höchsten einstellbaren Fluss von $\phi = 4{,}25 \cdot 10^9 \frac{p}{cm^2 \cdot s}$ ist der oben beschriebene physikalische Effekt aufgrund der Strahlendosis sehr gut zu erkennen. In Abbildung 9.13.a ist zu sehen, dass der Core-Strom bis zu einer Dosis von knapp 1 MRad bis auf $I_{Core1MRad} = 7{,}8\,\mathrm{mA}$ ansteigt und dann bei der weiteren Bestrahlung die restlichen zweieinhalb Stunden und einer Dosis von $D = 4{,}21\,\mathrm{MRad}$ auf $I_{Core}585\,\mu\mathrm{A}$ abfällt. Nach der Bestrahlung und einer Nacht Ruhe beträgt der Core-Strom $I_{CoreNachIrrad} = 246\,\mu\mathrm{A}$, vor der Bestrahlung betrug der Core-Strom $I_{CoreVorIrrad} = 77{,}5\,\mu\mathrm{A}$. Betrieben wurde dieser DCS-Controller mit einer Core-Spannung von 1 V, die konstant bleibt, nur beim hohen Core-Strom minimal absinkt.

In Abbildung 9.13.a ist außerdem zu sehen, dass der Chip schon vorher bestrahlt wurde (eine Stunde lang mit dem Fluss von $\phi = 1{,}07 \cdot 10^9 \frac{p}{cm^2 \cdot s}$). Mit einem senkrechten Strich ist der Zeitpunkt markiert, an dem der Strahl eingeschaltet wurde. Der Strahl wurde ca. 5 Minuten nach Start der Messung eingeschaltet, so dass die ersten 5 min der Core-Strom zunächst sinkt.



(a) (b)

Abbildung 9.13: (a) Bestrahlung des CoFee2 TMR mit höchstem Fluss (b) Das Temperaturverhalten des bestrahlten Chips CoFee2 TMR und Verhalten der Umgebungstemperatur

Die zwei scheinbar getrennten Kurven ergeben sich aus dem speziellen Betriebsmodus, der für diesen Chip gewählt wurde. Nach dem Senden eines I2C-HC-Befehls zum DCS-Chip wird der DCS-Controller jedesmal zurückgesetzt, dadurch wird der Stromverbrauch gesenkt.

Die Temperatur des Chips, zu sehen in Abbildung 9.13.b, verhielt sich auch hier wie beim anderen DCS-Controller. Bei dem Anstieg des Core-Stroms stieg auch die Temperatur leicht an, trotzdem scheint nicht die absolute Chiptemperatur gemessen worden zu sein, sondern nur der Teil, der auch das Potting entsprechend aufheizen konnte.

9.4.1 Weitere Studien

Aus den Daten zur Bestimmung des SEU-Wirkungsquerschnitts aus Kapitel 9.2.1 konnten die richtungsabhängigen Bitfehler bestimmt werden. Das in die Schieberegister geschriebene Muster (110011001100...) ermöglicht die Bestimmung der einzelnen Wirkungsquerschnitte für die Bitfehlerrichtung $\sigma_{0\to1}$ und $\sigma_{1\to0}$. Da nur die Hälfte aller Bits im Schieberegister durch einen SEU den Wert von $0 \to 1$ und umgekehrt ändern konnten, wird für die einzelnen Wirkungsquerschnitte eine Zahl der Register von $N_{Reg1/2} = 3000$ angenommen. Die Zahlen der Bitfehler in Abhängigkeit von der Richtung sind in Tabelle 9.3 aufgetragen.

Fluenz $\times$ $N_{Reg1/2}$	Bitfehler $0 \rightarrow 1$	Bitfehler $1 \rightarrow 0$
$0{,}15 \cdot 10^{16}$	28 ± 5	68 ± 8
$0{,}51 \cdot 10^{16}$	130 ± 11	318 ± 18
$0{,}99 \cdot 10^{16}$	323 ± 18	657 ± 26
$3{,}99 \cdot 10^{16}$	2030 ± 45	4197 ± 65

Tabelle 9.3: Anzahl der detektierten Bitfehler in Abhängigkeit von der Richtung. $N_{Reg1/2} = 3000$.

In Tabelle 9.3 ist deutlich zu erkennen, dass die Richtung $1 \rightarrow 0$ bevorzugt durch SEU induziert wird. Die einzelnen Wirkungsquerschnitte wurden graphisch aus Abbildung 9.14 bestimmt zu[4)]:

$$\sigma_{0\rightarrow1} = (3,23 \pm 0,36) \cdot 10^{-14} \frac{cm^2}{bit} \tag{9.11}$$

und

$$\sigma_{1\rightarrow0} = (7,26 \pm 0,38) \cdot 10^{-14} \frac{cm^2}{bit} \tag{9.12}$$

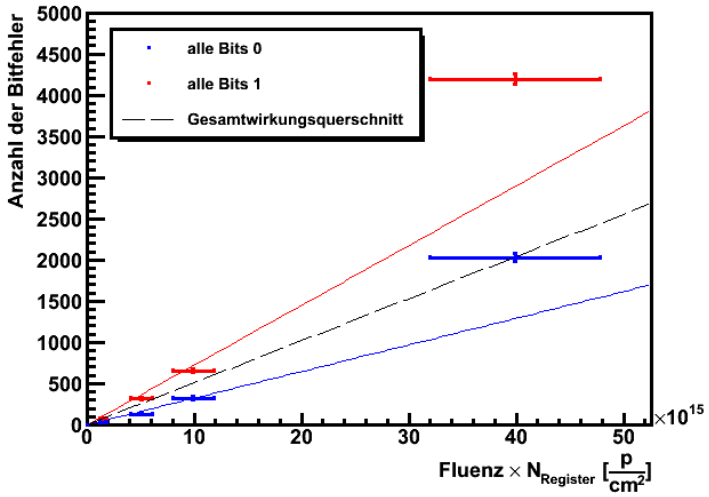


Abbildung 9.14: Bestimmung des Wirkungsquerschnitts für die Richtungen der Bitfehler von $0 \rightarrow 1$ in blau und $1 \rightarrow 0$ in rot. Die schwarze gestrichelte Linie zeigt den SEU-Gesamtwirkungsquerschnitt an.

[4)] Der Wert für die höchste Fluenz wurde hier nicht korrigiert.

Der in Abbildung 9.14 eingezeichnete SEU-Wirkungsquerschnitt aus Gleichung 9.1 beträgt: $\sigma_{130\,nm} = (5{,}11 \pm 0{,}91) \cdot 10^{-14}\,\frac{cm^2}{bit}$. Dies ergibt in etwa auch der Mittelwert aus den beiden Wirkungsquerschnitten $\sigma_{0\to1}$ und $\sigma_{1\to0}$ mit:

$$\sigma_{Mittelwert} = (5,25 \pm 0,37) \cdot 10^{-14}\,\frac{cm^2}{bit} \tag{9.13}$$

Das Verhalten der richtungsabhängigen Bitfehler ist vom Prozess abhängig. Es muss eine genaue Kenntnis der sensitiven Flächen der einzelnen CMOS-Komponenten vorliegen, um das beobachtete Verhalten verifizieren zu können. Zu diesem Zeitpunkt sind diese Studien noch nicht abgeschlossen.

9.5 Zusammenfassung

Bei der Bestrahlung des Schieberegisters am Paul Scherrer Institut konnte der SEU-Wirkungsquerschnitt bestimmt werden. Dieser wurde mit einer verbesserten Statistik gegenüber der Bestrahlung des DCS-Chips bestimmt. Auch liegt der Wert für den Wirkungsquerschnitt aus Gleichung 9.1 im Rahmen des erwarten SEU-Wirkungsquerschnitts für diesen Prozess.

Im Weiteren wurde mit der Bestrahlung des CoFee2_noTMR gezeigt, dass eine TMR Absicherung für eine Strahlungsumgebung mit Teilchenflüssen ab $\phi = 5 \cdot 10^8\,\frac{p}{cm^2 \cdot s}$ absolut notwendig ist. Es konnte keine stabile DCS-Kommunikation mit CoFee2_noTMR im Strahl hergestellt werden. Mit CoFee2_TMR hingegen konnte eine stabile Kommunikation im Strahl mit den höchsten zu erreichenden Teilchenflüssen aufgebaut werden. Auch im direkten Vergleich mit dem reduzierten Betriebsprogramm des CoFee2_TMR wurde die Notwendigkeit der TMR Absicherung sichtbar, da CoFee2_noTMR mit demselben Programm auch Fehler in der DCS-Kommunikation aufwies.

Um eine konkrete Aussage darüber treffen zu können, wie sich der SEU-Wirkungsquerschnitt einer Komponente verringert, die durch das TMR abgesichert ist, sollte eine weitere Messung durchgeführt werden. Dazu könnte dasselbe Schieberegister, welches in dieser Bestrahlung verwendet wurde, mit zusätzlicher TMR-Absicherung submittiert werden. Dies würde einen direkten Vergleich der experimentell gemessenen SEU-Wirkungsquerschnitte für Komponenten im 130 nm Prozess mit und ohne TMR-Absicherung schaffen.

Des Weiteren wurden interessante Dosiseffekte entsprechend der Theorie aus Kapitel 4.2.3 bei der Bestrahlung von CoFee2_TMR bis zu einer Dosis von 4,21 MRad sichtbar. Die Dosiseffekte sind zur Vollständigkeit erwähnt worden, genaue Dosisbetrachtungen mit den zu erwartenden HL-LHC-Dosen müssen noch durchgeführt werden.

Kapitel 10

Ergebnisse und Ausblick

In dieser Arbeit wurde das Konzept für ein zukünftiges Detektor-Kontrollsystem des Pixeldetektors für das Phase 2 Upgrade des ATLAS-Detektors vorgestellt. Dazu wird eine sichere DCS-Kommunikation im Control und Feedback Pfad in der hohen Strahlungsumgebung angestrebt. Es wird eine Kommunikation mithilfe von zwei DCS-Komponenten im 130 nm Prozess aufgebaut, die durch den DCS-Controller und den DCS-Chip realisiert wird. Mithilfe dieser Komponenten werden DCS-Befehle über mehr als 100 m fehlerfrei zum Detektor unter Verwendung einer geringen Zahl an Leitungen übertragen.

Für den in dieser Arbeit entwickelten DCS-Controller wurde das CAN-Protokoll in einer Hardware-Beschreibungssprache aufgebaut und im 130 nm Prozess implementiert. Der entwickelte DCS-Controller erwies sich aufgrund von ausgiebigen Tests und einer Bestrahlung als geeignet für den Einsatz im geplanten Detektor-Kontrollsystem. Dabei wurde speziell die Toleranz gegenüber Einzelfehlereffekten geprüft und durch den Einsatz einer TMR-Absicherung als geeignet für die HL-LHC-Strahlungsumgebung befunden.

Im Rahmen dieser Arbeit wurde auch der DCS-Chip bestrahlt und die Funktionsfähigkeit der DCS-Kommunikation zum Detektor im Hinblick auf Einzelfehlereffekte in der HL-LHC-Strahlungsumgebung erfolgreich bestätigt.

Für theoretische Berechnungen der Bitfehlerrate durch Einzelfehlereffekte wurde zudem ein SEU-Wirkungsquerschnitt für den verwendeten 130 nm Prozess bestimmt.

Aus den in dieser Arbeit entwickelten Komponenten kann ein DCS-Netzwerk aufgebaut werden. Implementiert in einem Systemtest, können dadurch Studien des Detektorbetriebs des ATLAS-Pixeldetektors ermöglicht werden.

Literaturverzeichnis

[1] D. Perkins, *Hochenergiephysik*. Addison-Wesley Publishing Company, Bonn, München, Reading Massachusetts (u.a.), 3. Auflage, 1991.

[2] The CMS and the ATLAS Collaboration, S. Worm, *Searches for Physics beyond the Standard Model*, Talk at ICHEP Melbourne (2012) .

[3] D. Griffiths, *Introduction to Elementary Particles*. WILEY-VCH Verlag Gmbh & Co. KGaA, Weinheim, 2. Auflage, 2008.

[4] ALEPH Collaboration, DELPHI Collaboration, L3 Collaboration, OPAL Collaboration, SLD Collaboration, LEP Electroweak Working Group, SLD Electroweak Group, SLD Heavy Flavour Group Collaboration, S. Schael et al., *Precision electroweak measurements on the Z resonance*, Phys.Rept. **427** (2006) 257–454, `arXiv:hep-ex/0509008 [hep-ex]`.

[5] A. Djouadi, *The Anatomy of electro-weak symmetry breaking. I: The Higgs boson in the standard model*, Phys.Rept. **457** (2008) 1–216 LPT-ORSAY-05-17, `arXiv:hep-ph/0503172 [hep-ph]`.

[6] LHC Higgs Cross Section Working Group Homepage . `https://twiki.cern.ch/twiki/bin/view/LHCPhysics/CrossSections`.

[7] ATLAS Collaboration, G. Aad et al., *Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC*, Phys.Lett. **B716** (2012) 1–29 CERN-PH-EP-2012-218, `arXiv:1207.7214 [hep-ex]`.

[8] CMS Collaboration, S. Chatrchyan et al., *Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC*, Phys.Lett. **B716** (2012) 30–61 CMS-HIG-12-028, CERN-PH-EP-2012-220, `arXiv:1207.7235 [hep-ex]`.

[9] T. ATLAS-Collaboration, *Physics at a High-Luminosity LHC with ATLAS*, ATL-PHYS-PUB-2012-001, ATL-COM-PHYS-2012-1118.

[10] *ATLAS: Detector and physics performance technical design report. Volume 1*, CERN-LHCC-99-14, ATLAS-TDR-14.

[11] Offizielle Webseite vom CERN . `www.cern.ch`.

[12] *ATLAS: Letter of Intent for the Phase-II Upgrade of the ATLAS Experiment*, CERN-LHCC-2012-022, LHCC-I-023.

[13] T. Todorov, *Presentation of the alpine layout Multi Stave Card (MSC)*, Talk at the ITK DCS-Meeting (26. June 2012) .

[14] D. Nelson, *Data transmission on strip-lines, end of stave card, and micro-twinax cable documents*, Talk (27. March 2012) .

[15] L. Püllen, K. Becker, J. Boek, S. Kersten, P. Kind, et al., *Studies for the detector control system of the ATLAS pixel at the HL-LHC*, JINST **7** (2012) C02053.

[16] R. B. GmbH, *CAN Specification Version 2.0*, . `http://www.semiconductors.bosch.de/media/pdf/canliterature/can2spec.pdf`.

[17] K. Kloukinas, *Access to ASIC design tools and foundry services at CERN for SLHC*, 2008. Talk at the TWEPP Conference (2008) .

[18] Radiation Background Simulations Twiki Page . `https://twiki.cern.ch/twiki/bin/viewauth/Atlas/RadiationBackgroundSimulations`.

[19] F. Faccio, C. Detcheverry, and M. Huhtinen, *First Evaluation of the Single Event Upset (SEU) Risk for Electronics in the CMS Experiment*, CERN-CMS-NOTE-1998-054, CMS-NOTE-1998-054.

[20] F. Faccio, *Radiation effects in the electronics for CMS*, CERN radiation tutorial . `http://lhcb-elec.web.cern.ch/lhcb-elec/papers/radiation_tutorial.pdf`.

[21] K. Hänsler, G. Anelli, S. Baldi, F. Faccio, W. Hajdas, and A. Marchioro, *TID and SEE performance of a commercial 0.13 μm CMOS technology*, Radiation and Its Effects on Components and Systems, 2003. RADECS 2003. Proceedings of the 7th European Conference on (2003) 119 –125.

[22] Wikipedia, Artikel über den Latch-Up-Effekt . `http://de.wikipedia.org/wiki/Latch-Up-Effekt`.

[23] Wikipedia, Artikel über die Grabenisolation . `http://de.wikipedia.org/wiki/Grabenisolation`.

[24] F. Faccio and G. Cervelli, *Radiation-Induced Edge Effects in Deep Submicron CMOS Transistors*, IEEE Transactions on Nuclear Science **Vol. 52, No.6** (December 2005) .

[25] M. Barbero, D. Arutinov, M. Backhaus, X.-C. Fang, L. Gonella, et al., *The FE-I4 pixel readout chip and the IBL module*, PoS **VERTEX2011** (2011) 038 SLAC-PUB-14958.

[26] M. Menouni, D. Arutinov, M. Barbero, R. Beccherle, P. Breugnon, R. Ely, D. Fougeron, M. García-Sciveres, D. Gnani, T. Hemperek, M. Karagounis, R. Kluit, A. Mekkaoui, A. Rozanov, and J.-D. Schipper, *Design and measurements of SEU tolerant latches*, CERN-2008-008.

[27] P. Koopman, *32-Bit Cyclic Redundancy Codes for Internet Applications*, DSN '02 Proceedings of the 2002 International Conference on Dependable Systems and Networks **Pages 459 - 472** (202) .

[28] K. Becker, *Design and Test of a Control Chip for the Future ATLAS Pixel Detector at the sLHC*, Diplomarbeit, Bergische Universität Wuppertal, Juli 2010.

[29] K. Etschberger, *CAN Controller-Area-Network.* Carl Hanser Verlag, München, Wien, 1994.

[30] Wikipedia, Artikel über die Triple Modular Redundancy . `http://en.wikipedia.org/wiki/Triple_modular_redundancy`.

[31] Private Kommunikation mit dem ASIC-Labor der Universität Heidelberg unter der Leitung von J. Schemmel und H. C. Schultz-Coulon .

[32] S. Römer, *Entwicklung einer Spannungsversorgung inklusive Überwachung für Bestrahlungsmessungen von DCS-Komponenten des ATLAS-Pixel-Detektors.* Bachelorarbeit, Bergische Universität Wuppertal, 2012.

[33] Diese Grafik wurde erstellt von L. Püllen, Bergische Universität Wuppertal .

[34] L. Püllen, K. Becker, J. Boek, S. Kersten, P. Kind, et al., *Studies for the detector control system of the ATLAS pixel at the HL-LHC*, JINST **7** (2012) C02053.

[35] Private Kommunikation mit M. Barbero vom Centre de Physique des Particules de Marseille .

[36] Proton Irradiation Facility High Energy Website at PSI . `http://pif.web.psi.ch/pif.htm`.

[37] Private Kommunikation mit I. Britvitch vom Paul Scherrer Institut .

Abbildungsverzeichnis

Tabellenverzeichnis

Danksagung

Hierdurch möchte ich mich ganz herzlich bei allen bedanken, die zur Verwirklichung dieser Arbeit beigetragen haben.

Ein besonderer Dank geht an meinen Doktorvater Prof. Dr. P. Mättig, der mir diese Arbeit an dem sehr spannenden Thema ermöglicht hat. Auch Prof. Dr. C. Zeitnitz möchte ich für die stets gute Zusammenarbeit und für die Unterstützung bedanken, die mir zu Teil wurde.

Zudem möchte ich mich bei allen Mitarbeitern der Gruppe Elementarteilchenphysik für die gute Zusammenarbeit und das tolle Arbeitsklima bedanken. Dabei gilt mein spezieller Dank an Susanne Kersten, Peter Kind, Lukas Püllen, Kathrin Becker und Tobias Flick. Ihr habt mich in zahlreichen Diskussionen mit euren Ideen und eurer Erfahrung immer einen Schritt weitergebracht und wart jederzeit für mich da.

Des Weiteren möchte ich ganz besonders meiner Familie danken, die durch ihre Unterstützung in moralischer Form sowie in Form von Babysitting einen wesentlichen Teil zu dieser Arbeit beigetragen hat.

Meinem Mann Thorsten und meinem Sohn Leonard danke ich sehr für eure Unterstützung während der Doktorarbeit, ihr wart für mich immer das Licht am Horizont.