

Einsatz programmierbarer Logikbausteine in der COMPASS-Ausleseelektronik

Als Diplomarbeit vorgelegt von
Marco Schierloh

Fakultät für Physik
Albert-Ludwigs-Universität
Freiburg i. Brsg.
Dezember 1998

Inhaltsverzeichnis

1	Einleitung	1
2	Die Physik von COMPASS	3
2.1	Messung der Gluonenpolarisation ΔG	4
2.2	Longitudinale Spinverteilungsfunktionen	7
2.3	Messung der Λ und $\bar{\Lambda}$ Polarisation	8
2.4	Transversale Spinverteilungsfunktionen	8
3	Der experimentelle Aufbau von COMPASS	9
3.1	Das polarisierte Target	10
3.2	Die Spektrometernagnete	11
3.3	Detektoren zur Rekonstruktion von Teilchenspuren	11
3.4	Detektoren zur Teilchenidentifizierung	15
3.5	Generation der Trigger	17
4	Das COMPASS Datenauslesesystem	19
4.1	Elektronik an den Detektoren (" <i>Frontend</i> ")	21
4.2	Das CATCH-System	22
4.3	Zwischenspeicherung und Filterung der Daten	25
5	Das CATCH1-Modul	27
5.1	Ziele der Entwicklung	27
5.1.1	Das CATCH1-Modul als Prototyp	27
5.1.2	Das CATCH1-Modul als Testmöglichkeit für Frontend Elektronik	28
5.2	Technische Realisierung	30
5.2.1	Der VMEbus	30
5.2.2	Programmierbare Logikbausteine	33
5.2.3	Die CATCH1-Platine	38
6	Erste Tests und Ergebnisse	47
7	Zusammenfassung	57
A	Schaltplan der VME-Schnittstelle	60
B	Beispiel für einen ABEL-HDL Quellcode	61

1 Einleitung

Im Verlauf ihrer Geschichte hat die europäische Organisation für Kernforschung in Genf, CERN¹, auf dem Gebiet der Hochenergiephysik eine Reihe von Experimenten hervorgebracht, die eine detaillierte Untersuchung der kleinsten Strukturen der Materie zum Ziel haben. In diesen Experimenten interessiert man sich z. B. für die Zusammensetzung der Nukleonen oder für die Produktion und den Zerfall von subatomaren Teilchen, die es in der Natur nur für kurze Zeit nach der Entstehung des Universums gegeben hat.

Im Jahr 1995 haben zwei Arbeitsgruppen von Wissenschaftlern die Durchführung zweier Experimente mit unterschiedlichen Zielsetzungen vorgeschlagen. So wollte die CHEOPS-Kollaboration [1] vor allem Baryonen mit der Quantenzahl *Charm* spektroskopisch untersuchen, während die HMC-Kollaboration [2] durch Streuung von Myonen an einem polarisierten Target einen Beitrag zur Aufklärung der Nukleonspinstruktur leisten wollte. Nachdem sich herausstellte, daß beide Experimente mit einem ähnlichen Aufbau durchgeführt werden sollten, tat man sich zusammen und gründete die COMPASS-Kollaboration um die physikalischen Ziele mit vereinten Kräften zu erreichen. Das COMPASS-Experiment wurde 1997 genehmigt und befindet sich derzeit in der Aufbauphase.

Für eine erfolgreiche Durchführung der von COMPASS angestrebten Messungen sind sehr hohe Ereignisraten unbedingt erforderlich. Diese werden aber enorme Datenmengen zur Folge haben. Aus diesem Grund wurde für das Datenauslesesystem von COMPASS ein völlig neues Konzept entwickelt, welches in der Lage sein wird, neben einer schnellen Detektorauslese auch eine Reduktion der Daten vorzunehmen, bevor diese auf Band geschrieben werden. Eine zentrale Komponente wird dabei das CATCH-System sein, welches derzeit an der Universität Freiburg entwickelt wird. Dieses auf der VMEbus-Architektur basierende System wird alle Detektorkomponenten mit Triggersignalen versorgen und auslesen. Ferner ist mit seiner Hilfe ein erstes Sortieren der Detektordaten nach Ereignisnummern möglich, bevor diese über eine einheitliche Schnittstelle zur nächsten Komponente des Datenauslesesystems weitergeleitet werden.

In der vorliegenden Arbeit soll die Entwicklung der CATCH1-Platine vorgestellt werden, bei der es sich um einen Prototypen für das CATCH-System handelt. Im Zuge der Entwicklung dieser VME-Karte wurden zwei Ziele verfolgt: zum einen soll mit ihrer Hilfe das Konzept des CATCH-Systems als Bestandteil der Datenauslese im Experiment überprüft werden. Auf der anderen Seite bietet diese Platine eine Testmöglichkeit für die bei COMPASS

¹CERN, franz.: Conseil Européen pour la Recherche Nucléaire

benötigte Elektronik zur Digitalisierung von Meßwerten. Für einen Teil des Detektorsystems, die Straw-Driftrohrechen, wird diese Elektronik ebenfalls an der Universität Freiburg entwickelt.

Im Verlauf der Arbeit wird zunächst ein kurzer Überblick über die physikalischen Ziele des Experiments gegeben. Die daraus resultierenden Anforderungen an den experimentellen Aufbau kommen in einem Abschnitt über die vorgesehenen Detektorkomponenten zur Sprache. Die Verarbeitung der Detektorinformationen wird in einem Abschnitt dargelegt, der die Struktur des Datenauslesesystems zum Inhalt hat. In diesem Zusammenhang wird insbesondere die Funktion des CATCH-Systems erläutert werden, wobei der Schwerpunkt auf der Auslese von Straw-Driftrohrechen liegt, für die es ursprünglich konzipiert war.

Der Hauptteil der Arbeit beschäftigt sich mit der CATCH1-Platine und ihren Möglichkeiten. Neben den Zielen, die bei ihrer Entwicklung verfolgt wurden, wird die technische Realisierung des CATCH1-Konzeptes vorgestellt werden, wobei auch auf eine Reihe Details der auf der Platine verwendeten Komponenten eingegangen wird. Hierbei handelt es sich vor allem um programmierbare Logikbausteine, die dem Benutzer vielseitige Möglichkeiten bei der Entwicklung neuer Hardware eröffnen.

Die Behandlung von Inbetriebnahme und ersten Tests der CATCH1-Platine ermöglicht eine Bewertung der dabei aufgetretenen Probleme und führt am Schluß der Arbeit zu einer Zusammenfassung. Hier wird ein Ausblick auf die weitere Entwicklung der CATCH-Hardware für das COMPASS-Experiment gegeben.

2 Die Physik von COMPASS

Das COMPASS-Experiment befindet sich momentan am CERN im Aufbau und soll im Jahr 2000 in der Halle 888 am SPS-Speicherring² in Betrieb gehen. Der Name COMPASS steht für *Common Muon and Proton Apparatus for Structure and Spectroscopy* [3].

Mit dem geplanten Experiment kann ein breitgefächertes Programm der Teilchenphysik durchgeführt werden. So möchte man zum einen die Nukleonenstruktur durch unelastische Streuung polarisierter Myonen an einem festen, polarisierbaren Target untersuchen. Auf der anderen Seite soll durch Verwendung von Hadronenstrahlen Spektroskopie von Teilchen mit der Quantenzahl Charm ermöglicht werden. Der Vergleich experimenteller Ergebnisse mit Vorhersagen der nicht-störungstheoretischen Quantenchromodynamik (*QCD*), von Gitterrechnungen oder Modellen, die auf chiraler Symmetrie basieren, wird helfen das Verständnis der Hadronen zu verbessern.

Unter Verwendung von Pion- oder Protonstrahlen wird die Suche nach sogenannten exotischen Zuständen ein Schwerpunkt des Experiments sein. Diese werden von der QCD vorhergesagt, sind aber experimentell noch nicht eindeutig nachgewiesen. Es handelt sich dabei um reine oder gemischte gluonische Systeme wie z. B. *Glueballs*, die nur aus Valenzgluonen bestehen. Für diese Messungen stellt das SPS Teilchenstrahlen im Energiebereich zwischen 100 GeV und einigen hundert GeV zur Verfügung. Diese Strahlen mit einer sehr hohen Teilchenrate erlauben Messungen bei einer großen integrierten Luminosität. COMPASS ist ein Experiment, das aus zwei Spektrometereabschnitten besteht, die beide über ausgezeichnete Teilchenidentifikation und Kalorimetrie verfügen werden und mit Strahlintensitäten von bis zu $2 \cdot 10^8$ Teilchen pro Spill³ zurechtkommen sollen. Ein schnelles Datenauslesesystem, für das spezielle Trigger notwendig sein werden, komplettiert die herausragende Leistung des Detektorsystems.

Die Messungen, bei denen ein Myonenstrahl verwendet wird, sollen vor allem der Frage nachgehen, wie die verschiedenen Bestandteile des Nukleons zu seinem Spin beitragen. Nach der überraschenden Entdeckung, daß der Nukleonenspin nur zu einem geringen Teil von den Quarks getragen wird, soll nun mit dem COMPASS-Experiment im speziellen der Beitrag der Gluonen zum Spin des Nukleons untersucht werden.

Dieses Teilprogramm stellt den Schwerpunkt des COMPASS-Experiments in seiner Startphase dar. Die Messungen werden mit Targets durchgeführt, die sowohl longitudinal als auch transversal polarisiert werden können. Im

²SPS: Super Protonen Synchrotron

³Ein SPS Spill dauert 2,5 Sekunden, wonach eine 12 Sekunden lange Spillpause folgt.

folgenden wird ein kleiner Überblick über die Messungen mit einem Myonenstrahl gegeben.

2.1 Messung der Gluonenpolarisation ΔG

Ausgangspunkt ist die Tatsache, daß die Ellis-Jaffe Summenregel verletzt ist, also nicht der gesamte Nukleonenspin von seinen Valenzquarks getragen wird. Obwohl es zur Erklärung dieser Tatsache unterschiedliche theoretische Ansätze gibt, ist immer noch nicht geklärt, woher dieser Effekt rührt. Ein Modell sieht vor, daß neben den Quarkspins auch polarisierte Gluonen und die Drehimpulse der Bestandteile des Nukleons eine Rolle spielen. Andererseits gibt es aber auch die Annahme, daß der Beitrag der Quarks durch negativ polarisierte s-Quarks reduziert wird.

Um zwischen diesen beiden Theorien unterscheiden zu können, soll mit dem COMPASS-Experiment die Gluonenpolarisation ΔG durch Messung der longitudinalen Spinasymmetrie bei der Produktion von Mesonen mit einem c-Quark bestimmt werden. Diese Messung basiert auf dem Nachweis von D-Mesonen über ihre hadronischen Zerfallsprodukte. Weil schwere Quarks in führender Ordnung über den Prozeß der Photon-Gluon-Fusion (PGF) produziert werden, ist dieser Prozeß besonders sensitiv auf eine mögliche Polarisation der an der Reaktion beteiligten Gluonen. Hierbei verschmelzen das vom Myon abgestrahlte virtuelle Photon mit dem Viererimpuls q und das vom Nukleon⁴ abgestrahlte Gluon (Viererimpuls k) zu einem $c\bar{c}$ -Paar, das dann in Hadronen fragmentiert (siehe Abb. 1).

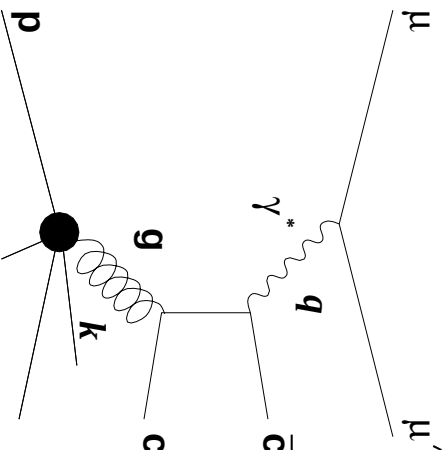


Abbildung 1: Diagramm zur Photon-Gluon-Fusion.

⁴hier: Proton

Gemessen wird nun die spinabhängige Asymmetrie A^{exp} , die durch die Zahl $N_{\vec{\sigma}}$ der Ereignisse mit c-Quarks für antiparallele ($\uparrow\downarrow$) und parallele ($\uparrow\uparrow$) Ausrichtungen der longitudinalen Myonen- und Targetspins gegeben wird:

$$A_{\mu N}^{\vec{\sigma}} = \frac{1}{P_B \cdot P_T \cdot f} \cdot A^{exp} = \frac{1}{P_B \cdot P_T \cdot f} \cdot \frac{N_{\vec{\sigma}}^{\uparrow\downarrow} - N_{\vec{\sigma}}^{\uparrow\uparrow}}{N_{\vec{\sigma}}^{\uparrow\downarrow} + N_{\vec{\sigma}}^{\uparrow\uparrow}}. \quad (1)$$

Hier sind P_B und P_T die Strahl- und Targetpolarisationen und f der Anteil polarisierbarer Nukleonen im Targetmaterial. Die Asymmetrie $A_{\mu N}^{\vec{\sigma}}$ hängt mit der Asymmetrie für virtuelle Photonen, $A_{\gamma N}^{\vec{\sigma}}$, über

$$A_{\gamma N}^{\vec{\sigma}} = \frac{1}{D} \cdot A_{\mu N}^{\vec{\sigma}} \quad (2)$$

zusammen. Hier ist D der Depolarisationsfaktor des virtuellen Photons in Bezug auf das Myon. Er läßt sich schreiben als

$$D(y) \approx \frac{1 - (1 - y)^2}{1 + (1 - y)^2}, \quad (3)$$

wobei die Skalenvariable $y = p \cdot q / k \cdot q = (E - E')/E$ den relativen Energietransfer ausdrückt. Hier sind p , q und k die Viererimpulse des Nukleons, des virtuellen Photons und des Gluons. E und E' stehen für die Energie des Myons vor bzw. nach dem Streuprozess.

$A_{\gamma N}^{\vec{\sigma}}$ läßt sich als Verhältnis vom helizitätsabhängigen Wirkungsquerschnitt $\Delta\sigma^{\gamma N \rightarrow \vec{\sigma} X}$ zum über die Helizität gemittelten Wirkungsquerschnitt $\sigma^{\gamma N \rightarrow \vec{\sigma} X}$ für die Charmproduktion darstellen. Diese wiederum können als Faltungen der elementaren polarisierten und unpolarisierten Photon-Gluon Wirkungsquerschnitte, $\Delta\sigma(\hat{s})$ und $\sigma(\hat{s})$, mit den Gluonverteilungen ΔG und G geschrieben werden [4]:

$$A_{\gamma N}^{\vec{\sigma}} = \frac{\Delta\sigma^{\gamma N \rightarrow \vec{\sigma} X}}{\sigma^{\gamma N \rightarrow \vec{\sigma} X}} = \frac{\int_{4m_\pi^2}^{2M_N E_y} d\hat{s} \Delta\sigma(\hat{s}) \Delta G(\eta, \hat{s})}{\int_{4m_\pi^2}^{2M_N E_y} d\hat{s} \sigma(\hat{s}) G(\eta, \hat{s})}. \quad (4)$$

In dieser Gleichung steht $\hat{s} = (q + k)^2$ für die invariante Masse des Systems aus Photon und Gluon; $\eta = \hat{s}/2M_N E_y$ ist der Impulsbruchteil des Gluons.

Zur Bestimmung der Asymmetrien sollen D^0 ($\bar{D}^0$) Mesonen genutzt werden, in die ca. 60% aller c-Quarks fragmentieren. Ihr Nachweis erfolgt über die hadronischen Zerfallsprodukte aus ihrem einfachsten Zerfallskanal, dem Zweikörperzerfall $D^0 \rightarrow K^- + \pi^+$, der mit einem Verzweungsverhältnis von $(4,01 \pm 0,14) \%$ stattfindet.

Im Schwerpunktsystem des D^0 Mesons steht für die aus diesem Zerfallskanal stammenden Pionen und Kaonen ein großer Impuls von $p^* = 861 \text{ MeV}/c$ zur Verfügung (siehe Abb. 2). Daher fliegen sie im Laborsystem unter großen Winkeln θ relativ zur D^0 Bewegungsrichtung auseinander. Das gilt nicht für solche Pionen oder Kaonen, die schon aus der Fragmentation oder aus Zerfällen anderer Teilchen entstanden sind. Diese Mesonen bewegen sich zu meist unter kleinen Winkeln bezüglich der Strahlrichtung.

Um den kombinatorischen Untergrund an Pionen und Kaonen möglichst klein zu halten, braucht man eine sehr gute Rekonstruktion der Teilchenspu ren, die es ermöglicht, Ereignisse mit der falschen Kinematik auszusortieren. Eine weitere Voraussetzung besteht in einer sehr guten Trennung von Pionen und Kaonen durch Teilchenidentifikation. In Abb. 3 zeigt eine Monte-Carlo Simulation, mit welchen Schnitten auf die kinematischen Variablen θ_K^* und z_D eine sehr gute Trennung von Ereignissen aus D^0 -Zerfällen und dem Un tergrund erreicht werden kann. Hier sind der θ_K^* der Winkel, unter dem das Kaon im Schwerpunktsystem des D^0 Mesons fliegt, und die Skalenvariable $z_D = E_D/\nu$ steht für den Anteil der Energie des virtuellen Photons, der vom D^0 Meson getragen wird. Dargestellt sind Schnitte auf $|\cos \theta_K^*| < 0,5$ und auf $z_D < 0,25$, die ein Verhältnis von Ereignissen zum Untergrund von ca. 1 zu 3,8 erlauben. Andere Zerfallskanäle der D^0 Mesonen, an denen mehr als zwei Zerfallsprodukte beteiligt sind, eignen sich nicht gut zum Nachweis, da die Kombinatorik hier nicht so günstig ist.

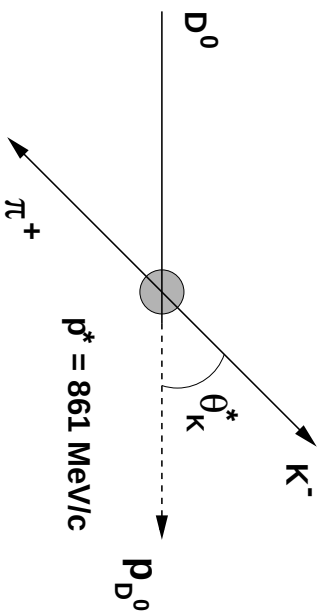


Abbildung 2: Zerfall eines D^0 Mesons in ein π^+ und ein K^- .

Der Untergrund kann noch weiter verringert werden wenn man zusätzlich D^{*+} Zerfälle berücksichtigt. Das D^{*+} zerfällt in ein π^+ und in ein D^0 . Das D^0 besitzt die oben beschriebene gut erkennbare Zerfallskinetik und man muß nur nach einem zusätzlichen Pion mit wenig Energie suchen.

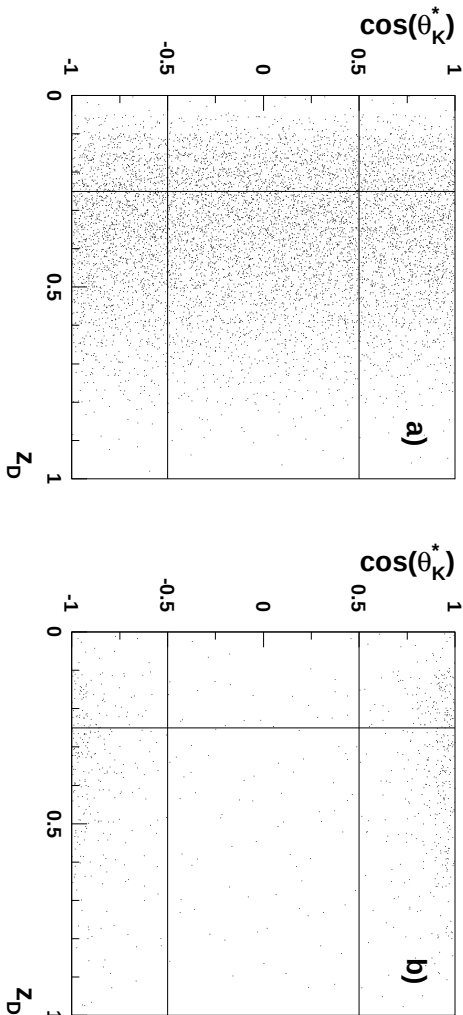


Abbildung 3: Verteilung der Ereignisse als Funktion von $\cos\theta_K^*$ und z_D für Ereignisse aus D°-Zerfällen (a) und Untergrund (b) (Monte-Carlo Simulation). Die durch Linien dargestellten Schnitte werden im Text erläutert.

Parallel zur Messung von $A_{\gamma N}^{\vec{e}_2}$ können auch inklusive⁵ Daten für die spinabhängigen Strukturfunktionen g_1 des Protons und des Deuterons aufgenommen werden, die bereits früher gemessen worden ist. Hierbei erwartet man eine deutlich bessere Statistik gegenüber früheren Experimenten am CERN und am DESY.

2.2 Longitudinale Spinverteilungsfunktionen

In Bezug auf die unterschiedlichen Modelle zur Spinstruktur des Hadrons ist es interessant den Nukleonenspin in Valenz- und Seequarkanteile aufzuteilen. Die Spinverteilungsfunktionen für die unterschiedlichen Flavours der Valenz- und der Seequarks können aus seminklusive⁶ Messungen von Hadronenasymmetrien in tiefinelastischer Streuung von polarisierten Leptonen an polarisierten Protonen- und Deuteron targets bestimmt werden. Die Hadronenasymmetrien lassen sich schreiben als

$$A_{\gamma N}^h = \frac{1}{P_B P_T f D} \frac{N_h^{\uparrow\downarrow} - N_h^{\uparrow\uparrow}}{N_h^{\uparrow\downarrow} + N_h^{\uparrow\uparrow}}, \quad (5)$$

wobei P_B und P_T wieder die Strahl- und Targetpolarisationen und f der Anteil polarisierbarer Nukleonen im Targetmaterial, sowie D der Depolarisa-

⁵in inklusiven Messungen weist man nur das gestreute Myon nach

⁶seminklusive bezeichnet den Nachweis des gestreuten Myons in Koinzidenz mit einem Hadron

tionsfaktor des virtuellen Photons sind. Der Index h bezeichnet die entsprechenden Hadrontypen π^- , π^+ , K^- , K^+ und K^0 .

Diese Messungen erfordern die Identifikation der Hadronen im Endzustand für das gesamte Impulsspektrum, und können mit guter Statistik parallel zur Messung von ΔG durchgeführt werden.

2.3 Messung der Λ und $\bar{\Lambda}$ Polarisation

Die Messung der longitudinalen Polarisation von Λ und $\bar{\Lambda}$ Baryonen kann benutzt werden um Informationen über die Polarisation der s-Quarks im Nukleon zu gewinnen. Auf diese Weise kann auch dazu beigetragen werden, zwischen den Modellen zu unterscheiden, die zum einen polarisierte Gluonen, und zum anderen negativ polarisierte s-Quarks als Erklärung für den zu geringen Anteil der Valenzquarks am Nukleonenspin angeben [5].

Weiterhin ist es möglich mit diesen Messungen die Spintransferfunktion vom getroffenen Quark zum Λ Hyperon zu bestimmen.

2.4 Transversale Spinverteilungsfunktionen

Ein weiter Teilaspekt im Rahmen des experimentellen Programms mit dem Myonenstrahl ist es, semiinklusive Daten unelastischer Streuevents an transversal polarisierten Protonen- und Deuteron targets zu nehmen. Hiermit ist es möglich die transversalen Spinverteilungsfunktionen $\Delta_T q(x) = q_T(x) - q_\downarrow(x)$ für verschiedene Quarksorten zu bestimmen. Die Pfeile $\uparrow$ und $\downarrow$ stehen für eine Quarkpolarisation parallel bzw. antiparallel zur transversalen Polarisation der Nukleonen. Bislang sind diese semiinklusive Spinverteilungsfunktionen noch nie gemessen worden.

3 Der experimentelle Aufbau von COMPASS

In Abb. 4 wird eine schematische Übersicht des COMPASS Detektorsystems gezeigt. Der Detektor besteht aus zwei Spektrometerabschnitten, die für einen präzisen Nachweis von Teilchen in einem großen geometrischen Akzeptanzbereich notwendig sind. Der erste Spektrometerabschnitt nach dem Target deckt große Winkel von maximal 200 mrad in vertikaler und 250 mrad in horizontaler Richtung ab. Teilchen, deren Spuren unter einem Winkel von weniger als 30 mrad verlaufen, werden erst im zweiten Spektrometerabschnitt detektiert. Beide Abschnitte bestehen aus Spektrometern (SM1-M/H und SM2), ringabbildenden Cerenkov Detektoren (RICH1⁷ und RICH2), Kalorimetern (ECAL1/2 und HCAL1/2) und sind mit verschiedenen Detektoren zur Rekonstruktion von Teilchenspuren ausgestattet.

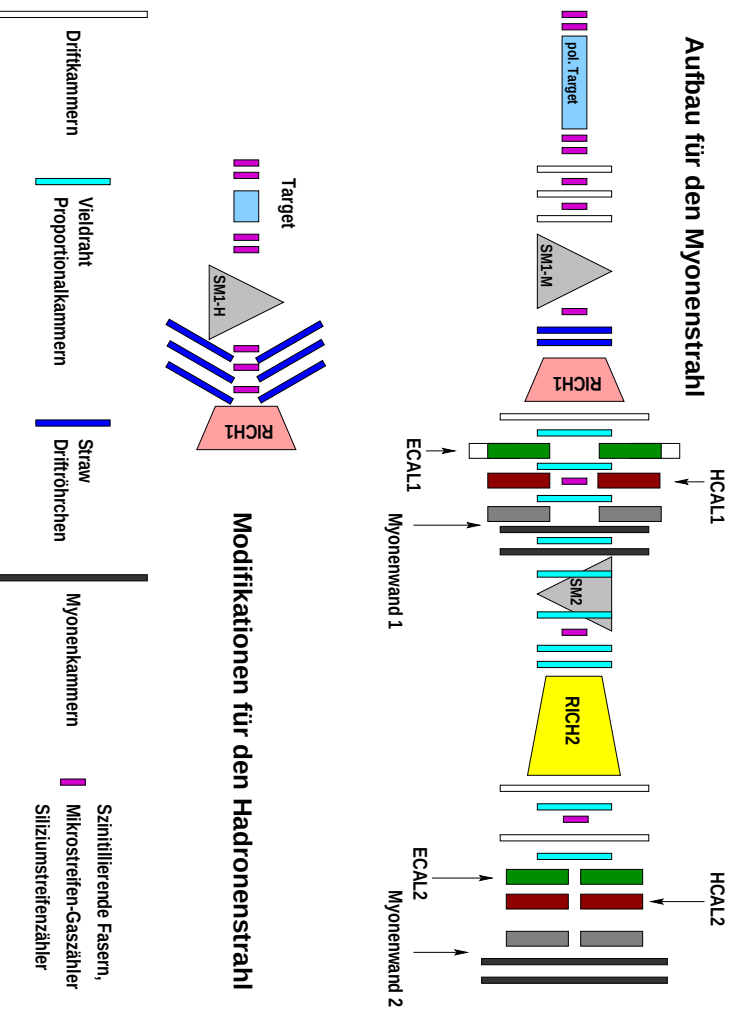


Abbildung 4: Schematische Übersicht über das COMPASS Detektorsystem.

Im mittleren Teil von Abb. 4 ist zu sehen, welche Modifikationen für die Messungen mit Hadronenstrahlen vorgenommen werden müssen. Es wird dann ein anderes Target verwendet werden, wobei gerade im Targetbereich

⁷RICH, engl.: Ring Imaging CHerenkov counter

eine exzellente Rekonstruktion der Teilchen mit Charrn, sowie ihrer Zerfallsprodukte notwendig sein wird. Bei diesen Messungen wird auch die oben erwähnte maximale Akzeptanz voll ausgenutzt werden.

Um die Zahl der nachgewiesenen Teilchen pro Auslesekanal über den ganzen Detektor möglichst konstant zu halten, sieht das Detektorkonzept eine nach außen abnehmende Granularität vor: dicht am Strahl liegende Detektoren stellen für die gleiche Fläche mehr Kanäle zur Verfügung als weiter außen positionierte Detektoren.

In der Startphase des Experiments werden noch nicht alle ursprünglich vorgesehenen Detektorkomponenten verfügbar sein. So ist z. B. geplant, die Vieldraht-Proportionalkammern zur Spurrekonstruktion hinter dem RICH1 schrittweise durch präziser messende Straw-Driftrohren zu ersetzen. Da noch nicht endgültig geklärt ist wie der Detektor nach seinem kompletten Aufbau genau aussehen wird, soll im Rahmen dieser Arbeit ein Überblick über den vorläufigen Aufbau gegeben werden, der dann auch in der Startphase des Experiments zum Einsatz kommen wird.

3.1 Das polarisierte Target

Als Target ist das existierende polarisierte Target von SMC [6] vorgesehen. Allerdings muß es mit einem größeren supraleitenden Magnetsystem ausgerüstet werden, um für die Messungen mit einem Myonenstrahl eine geometrische Akzeptanz von 180 mrad zu gewährleisten.

Die Polarisation der Nukleonen wird durch die Methode der dynamischen Kernpolarisation [7] (DNP)⁸ in einem homogenen Magnetfeld von 2,5 T bei Temperaturen unter 0,5 K erzielt. Dem Targetmaterial werden in verdünnter Form paramagnetische Zentren zugegeben, die sich mit Mikrowellen polarisieren lassen und ihre Polarisation dann auf das Targetmaterial übertragen. Dazu muß die Frequenz der Mikrowellen knapp über der Larmorfrequenz der paramagnetischen Zentren liegen. Ist die maximale Polarisation erreicht, wird diese nach Abschalten der Mikrowellen bei einer Temperatur von unter 50 mK und einem Magnetfeld von mehr als 0,5 T für lange Zeit gehalten.

Das Target besteht aus zwei zylindrischen Zellen von je 60 cm Länge und 3 cm Durchmesser. Die beiden Zellen werden immer entgegengesetzt polarisiert und können mit unterschiedlichen Materialien gefüllt werden. Als Targetmaterialien werden ⁶LiD für Messungen am Deuteron und NH₃ für Messungen am Proton verwendet. Mit diesen Materialien wird ein Polarisationsgrad von

⁸DNP, engl.: Dynamic Nuclear Polarisation

ca. 50% bzw. 85% erreicht, der mit Hilfe von Kernspinresonanzsonden kontinuierlich überwacht wird.

3.2 Die Spektrometermagnete

Die Spektrometermagnete SMI-M/H und SM2 werden zur Impulsbestimmung der nachgewiesenen Teilchen benötigt. Der Dipolmagnet SMI-M/H befindet sich im ersten Spektrometerabschnitt zwischen dem Target und dem ersten ringabbildenden Cerenkov-Detektor (RICH1). Es ist geplant, hier den existierenden MNP21 Magneten zu verwenden, der mit zusätzlichem Eisen ausgerüstet wird. Für den Betrieb mit einem Myonenstrahl (SMI-M) werden die Pol- und Jochstücke so erweitert, daß eine Öffnung von einem Meter erreicht wird. Dies entspricht vom Target aus gesehen dem Öffnungswinkel von 180 mrad. Für die Messungen mit Hadronenstrahlen (SMI-M) muß die Öffnung noch ein wenig vergrößert werden um hier die maximal verfügbare Akzeptanz zu erreichen. Die integrierte Feldstärke dieses Magneten wird bei $\int B dl = 1,0 \text{ Tm}$ liegen.

Als zweiter Magnet SM2 kommt der CERN MEP45 Dipolmagnet zum Einsatz, der schon bei dem SMC-Experiment verwendet worden war. Seine Position befindet sich zwischen dem ersten Kalorimeterabschnitt und dem zweiten Cerenkov Detektor. Dieser Magnet besitzt eine Öffnung von vertikal mal horizontal $100 \times 200 \text{ cm}^2$ und eine integrierte Feldstärke von 4.4 Tm .

3.3 Detektoren zur Rekonstruktion von Teilchenspu ren

Im Experiment stehen unterschiedliche Detektoren zur Spurrekonstruktion zur Verfügung. In der Nähe des Strahls werden relativ kleine Detektoren verwendet, die allerdings über eine große Zahl von Kanälen pro Flächeneinheit verfügen. Dazu gehören Siliziumstreifenzähler, verschiedene Typen von Mikrostreifen-Gaszählern und szintillierende Fasern.

Die äußeren Bereiche werden mit Straw-Driftrohrchen, Driftkammern und Vieldraht-Proportionalkammern ausgestattet. Im Folgenden wird eine kleine Übersicht über die einzelnen Komponenten gegeben.

- **Straw-Driftrohrchen**

Zunächst wird es diese Detektoren nur zwischen dem Spektrometernagneten SMI und RICH1 geben. Die ca. 14.000 Kanäle bestehen aus innen und außen mit Kohlenstoff beschichteten Röhren aus Kaptonfolie in welche zusätzlich noch Aluminium eingearbeitet ist. Die Röhren besitzen eine

Wandstärke von $60\text{ }\mu\text{m}$ und ihr Durchmesser beträgt in Strahlnähe $0,6\text{ cm}$ und weiter außen $1,0\text{ cm}$. Sie werden voraussichtlich mit einer Mischung aus Argon, Tetrafluorkohlenstoff und Kohlendioxid ($\text{Ar} : \text{CF}_4 : \text{CO}_2 \approx 74 : 20 : 6$) gefüllt, wobei das genaue Mischungsverhältnis dieses nicht brennbaren Gases noch nicht feststeht. In der Mitte der Röhren verläuft ein $30\text{ }\mu\text{m}$ starker Draht, an dem positive Hochspannung anliegt.

Durchquert nun ein geladenes Teilchen das Röhren, so ionisiert es auf seinem Weg Moleküle des Zählgases. Die negativ geladenen Elektronen driften im radialsymmetrischen elektrischen Feld zum Anodendraht. Auf den letzten Bruchteilen ihres Weges werden sie durch das hier sehr starke Feld so sehr beschleunigt, daß sie ihrerseits wieder Zählgas ionisieren. Die so einsetzende Elektronenlawine produziert eine Ladungswolke, die auf den Anodendraht auftrifft. Die positiv geladenen Ionen wandern langsam zur leitenden Hülle des Röhrens, die mit Masse verbunden ist und erzeugen auf dem Anodendraht durch Influenz ein meßbares Signal.

Die Driftgeschwindigkeit der Elektronen kann durch geschickte Wahl der Zählgaskomponenten variiert und über einen großen Teil ihrer Driftstrecke konstant gehalten werden, wie in Abb. 5 anhand einer Simulation der Orts-Driftzeitrelation für die voraussichtliche Gas Mischung der Straw-Driftrohren zu sehen ist.

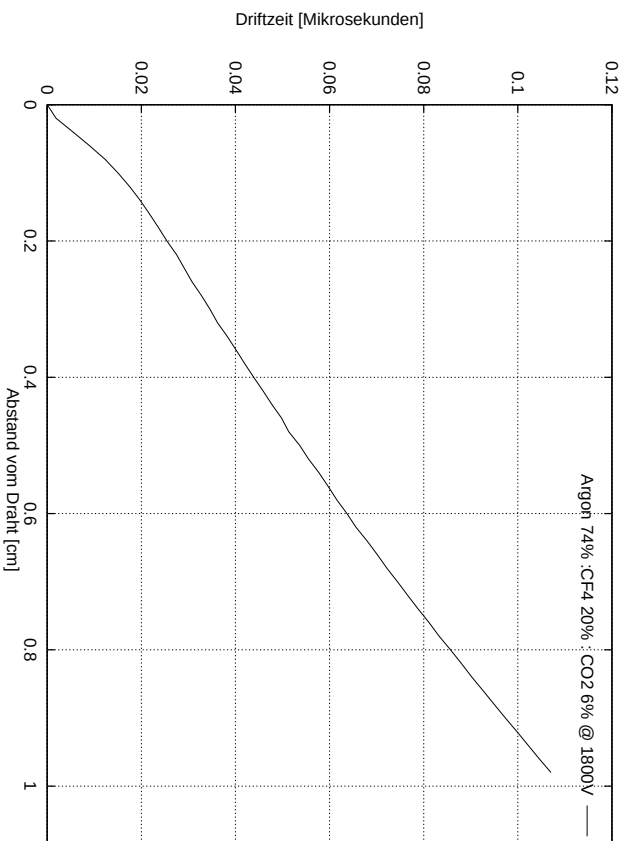


Abbildung 5: Garfield-Simulation [8] der Orts-Driftzeitrelation für die voraussichtliche Gas Mischung in den Straw-Driftrohren mit 1 cm Durchmesser. Die Anodendraftspannung wurde auf 1800 V eingestellt.

Die Messung der Driftzeit der Elektronen, also der Zeit, die zwischen einem Trigger und dem Signal auf dem Anodendraht liegt, ermöglicht die Bestimmung des kleinsten Abstandes der Teilchenspur vom Anodendraht mit einer Präzision von ca. $100\text{ }\mu\text{m}$. Die Raumkoordinaten der Teilchenspur erhält man durch Kombinationen von horizontal und vertikal ausgerichteten Driftrohren. Die Frage, ob das geladene Teilchen ein Driftrohrchen links oder rechts vom Anodendraht durchquert hat wird durch eine zweite Ebene von Straw-Driftrohrchen geklärt, die gegenüber der ersten Ebene um einen halben Driftraum verschoben ist. Die verbleibenden Ambiguitäten für Raumpunkte bei mehreren Treffern in der selben Ebene werden durch eine weitere, gegenüber den x - und y -Ebenen um einen kleinen Winkel (ca. 20°) gekippte Ebene aufgelöst.

- **Driftkammern**

Driftkammern arbeiten prinzipiell so ähnlich wie die Straw-Driftrohrchen. Auch hier gewinnt man die Ortsinformation der geladenen Teilchen durch Messung der Driftzeit der durch Ionisation entstandenen Elektronen. Allerdings wird hier die Verteilung des elektrischen Feldes durch die Anordnung von allen Anoden- und Kathodendrähten im gesamten Detektorvolumen bestimmt. Mit Hilfe des verwendeten Kammergases und einer günstigen Feldverteilung kann auch hier die Driftgeschwindigkeit der Elektronen über weite Teile ihrer Driftstrecke konstant gehalten werden.

Gegenüber den Straw-Driftrohrchen haben Driftkammern den Nachteil, daß defekte Drähte umgehend ausgetauscht werden müssen, da sie sonst durch einen Kurzschluß die ganze Kammer außer Betrieb setzen. Ein defektes Driftrohrchen hingegen kann einfach abgeschaltet werden.

- **Vieldraht-Proportionalkammern**⁹

Ein weiterer Detektortyp, der geladene Teilchen anhand der von ihnen verursachten Ionisation von Gasmolekülen nachweist, ist die Vieldraht-Proportionalkammer. Diese besitzt allerdings eine schlechtere Auflösung als eine Driftkammer, da hier die Ortsinformation nur von der Position der getroffenen Anodendrähte geliefert wird.

Bei COMPASS werden zunächst strahlabwärts vom ersten ringabbildenden Cerenkov Detektor im wesentlichen existierende Vieldraht-Proportionalkammern zur Spurrekonstruktion verwendet werden. Diese stammen von den Experimenten WA89 (Omega) und NA47 (SMC), sollen aber im Verlauf des Experimentes durch Straw-Driftrohren ausgetauscht werden.

⁹MWPC, engl.: Multi Wire Proportional Chamber

- **Detektoren im Stahlbereich**

Neben Siliziumstreifenzählern kommen in unmittelbarer Nähe des Teilchenstrahls auch Mikrostreifen-Gaszähler und szintillierende Fasern zum Einsatz. Die Mikrostreifen-Gaszähler sollen zur genauen Spurekonstruktion im Targetbereich eingesetzt werden. Ihr Aufbau ist in Abb. 6 schematisch dargestellt. Das einfallende geladene Teilchen ionisiert im 3 mm großen Konversionsbereich Gasmoleküle, woraufhin sich die positiven Ionen zur Kathode (HV1) bewegen und die Elektronen mit gleichmäßiger Geschwindigkeit zu einem feinen, unter positiver Hochspannung stehenden Gitter (HV2) driften. Erst hinter diesem Gitter findet die Gasverstärkung in einem wesentlich höheren elektrischen Feld statt. Die hier entstehende Elektronenwolke trifft auf feine Mikrostreifen auf, wo durch Influenz der positiven Ionen wieder ein Signal erzeugt wird, das Aufschluß über die Teilchenspur gibt.

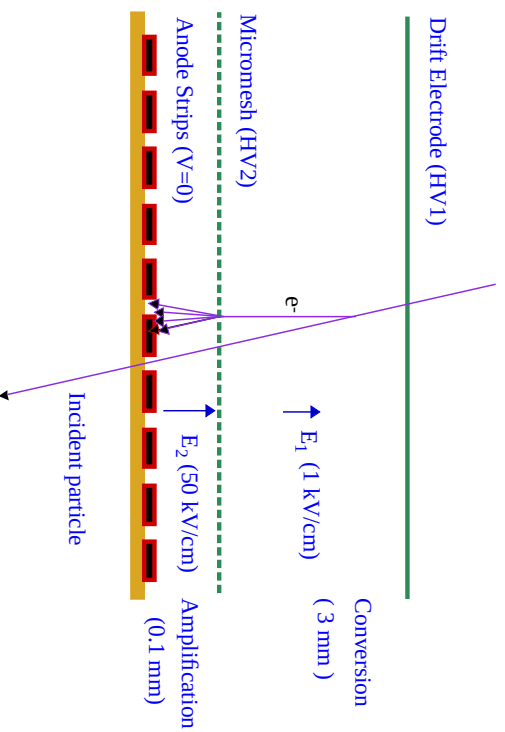


Abbildung 6: Aufbau der Mikrostreifen-Gaszähler.

Diese Detektoren haben den Vorteil, daß die in der Elektronenlawine entstehenden Ionen nur eine kurze Strecke zur Kathode zurücklegen müssen. Dadurch ist das Signal sehr kurz und die Mikrostreifen-Gaszähler sind sehr schnell wieder sensitiv. Aus diesem Grund können sie mit entsprechender Elektronik auch bei hohen Teilchenraten betrieben werden.

Werden in einem Ereignis mehrere Myonenspuren detektiert, so kann mit Hilfe der szintillierenden Fasern entschieden werden, welche Spur zum wechselwirkenden Myon gehört. Dazu wird überprüft, ob die szintillierenden Fasern vom Teilchen innerhalb eines bestimmten Zeitfensters, relativ zum ereignisauslösenden Trigger, durchquert wurden. Allerdings geschieht diese

Prüfung erst “Off-Line”, also nachdem das gesamte Ereignis auf einem Computersystem vorliegt, durch Verwendung geeigneter Software. Die Auslese der szintillierenden Fasern erfolgt einzeln über ein Pixel eines ortsauflösenden Photomultipliers.

3.4 Detektoren zur Teilchenidentifizierung

- Cerenkov Zähler

Für das COMPASS-Experiment ist eine sehr gute Trennung von Pionen, Kaonen und Protonen absolut notwendig. Die Unterscheidung dieser Teilchen wird mit Hilfe der Cerenkov Detektoren erreicht. Dabei macht man sich den Umstand zu nutze, daß geladene Teilchen beim Durchgang durch Materie oberhalb einer Schwellengeschwindigkeit

$$v_s = \frac{c}{n} \quad (6)$$

unter einem charakteristischen Winkel Cerenkov-Licht bezüglich der Richtung ihrer Bewegung emittieren. Dabei sind c die Lichtgeschwindigkeit im Vakuum und n der Brechungsindex des durchquerten Mediums.

Aus Gleichung (6) folgt ein Schwellenimpuls unterhalb dessen keine Emission von Cerenkov-Licht stattfindet:

$$p_t = \beta_t \gamma_t m c, \quad \text{mit} \quad \beta_t = \frac{1}{n} \quad \text{und} \quad \gamma_t = \frac{1}{\sqrt{1 - \beta_t^2}}. \quad (7)$$

Hieran ist zu erkennen, daß Teilchen mit gleichen Impulsen, aber unterschiedlichen Massen, Cerenkov-Licht unter verschiedenen Öffnungswinkeln

$$\theta_c = \arccos \frac{c}{nv} \quad (8)$$

abstrahlen, da sie unterschiedliche Geschwindigkeiten v besitzen.

Diese Eigenschaften nutzt man in den ringabbildenden Cerenkov Detektoren bei COMPASS aus. Abb. 7 zeigt eine schematische Ansicht des RICH1. In einem Gasbehälter strahlt ein einfallendes Teilchen Cerenkov-Licht ab, das dann über ein Spiegelsystem auf Detektoren abgebildet wird. Diese befinden sich außerhalb der geometrischen Akzeptanz des Spektrometers und besitzen eine gute Ortsauflösung. Hier erzeugen alle unter dem gleichen Winkel einfallenden Spuren einer Teilchensorte identische Ringe in der Hälfte des RICH, die sie durchqueren. Aus diesen kann dann zusammen mit dem Brechungsindex des verwendeten Gases der Öffnungswinkel des vom Teilchen

abgestrahlten Lichtkegels bestimmt werden. Dieser wiederum läßt zusammen mit dem Impuls des Teilchens¹⁰ auf seine Masse und damit auf seine Identität schließen.

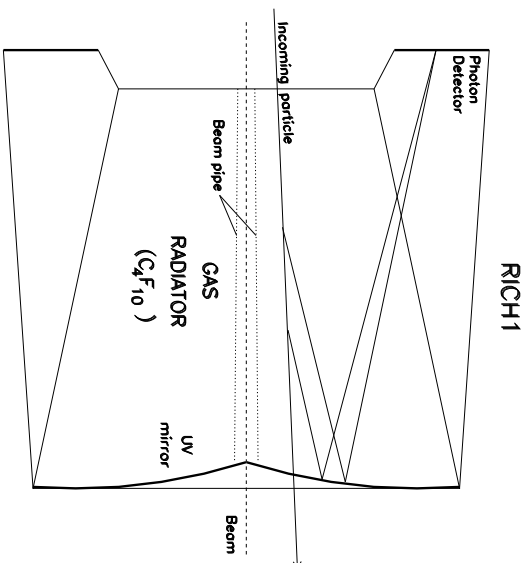


Abbildung 7: Skizze des ringabbildenden Cerenkov Detektors RICH1.

Um die Hadronen in einem großen Impulsbereich von 3 GeV/c bis 120 GeV/c mit mindestens drei Standardabweichungen (dies entspricht einer Wahrscheinlichkeit von 99,7 % für richtige Identifikation) voneinander zu trennen, werden zwei RICH-Detektoren verwendet. Dabei deckt der RICH1 den Impulsbereich von 3 GeV/c bis zu 65 GeV/c ab, während der RICH2 für den Bereich von 30 GeV/c bis 120 GeV/c zuständig ist. Die Schwellenimpulse für die Abstrahlung von Cherenkov-Licht können für beide Detektoren aus Tabelle 1 entnommen werden.

Tabelle 1: Schwellenimpulse zur Erzeugung von Cerenkov-Licht in den ringabbildenden Cerenkov Detektoren von COMPASS.

	RICH1	RICH2
Pionen	2,5 GeV/c	4,4 GeV/c
Kaonen	8,9 GeV/c	15,5 GeV/c
Protonen	17,9 GeV/c	29,5 GeV/c

¹⁰Impulse werden durch die Krümmung der Teilchenspuren unter dem Einfluß der Spektrometermagnete bestimmt

- **Kalorimeter**

Der experimentelle Aufbau von COMPASS sieht zwei Kalorimeter für Hadronen (HCAL1 und HCAL2) und zwei elektromagnetische Kalorimeter (ECAL1 und ECAL2) vor. Es ist geplant, das Kalorimeter MHC-200 [9] aus dem WA102-Experiment als Hadronenkalorimeter HCAL1 zu verwenden. Es besteht aus $200 \times 200 \times 1100 \text{ mm}^3$ großen Zellen, die aus Schichten von jeweils 25 mm Eisen und 5 mm Szintillator aufgebaut sind. Das zweite Hadronenkalorimeter wird aus $100 \times 100 \times 1200 \text{ mm}^3$ messenden Blöcken bestehen, die aber mit Schichten von jeweils 16 mm Blei und 4 mm Szintillatormaterial gefüllt sind. In beiden Fällen erfolgt die Auslese des Szintillatorlichts mit Wellenlängenschiebern und Photomultipliern. Während HCAL1 eine sehr gute Ortsauflösung besitzt (4 mm am Rand und 16 mm in der Mitte) verfügt HCAL2 über eine gute Energieauflösung von $\sigma_E/E \approx 60\%/\sqrt{E/\text{GeV}}$.

Die beiden elektromagnetischen Kalorimeter werden unbedingt für die Messungen mit Hadronenstrahlen benötigt und müssen dafür über gute Orts- und Energieauflösung verfügen. Auch hier ist geplant, bereits vorhandene Ausrüstung von WA102 zu verwenden: das aus Bleiglasblöcken bestehende GAMS-4000 Kalorimeter [10] soll im Strahlungsbereich mit strahlenharten Bleiwolframatkristallen erweitert werden.

- **Myonenwände**

Zum Nachweis der Myonen wird ihre Eigenschaft ausgenutzt, Materie leichter zu durchdringen als andere Teilchen. In beiden Spektrometerabschnitten befinden sich hinter den Kalorimetern ca. 2 m dicke Absorber aus Eisen, die dafür sorgen, daß in den dahintergelegenen Detektoren nur noch Myonen nachgewiesen werden. Diese Detektoren erlauben zusammen mit den Spurinformationen der Detektoren vor den Kalorimetern eine Rekonstruktion der Myonenspur.

Im Fall der ersten Myonenwand erfolgt der Nachweis der Myonen mit sogenannten *Plastic larocci Tubes* (PITs) [11], die eine Ortsauflösung von 1 mm besitzen. Die Myonenwand im zweiten Spektrometerabschnitt wird mit Driftrohren aus Aluminium betrieben, welche einen Durchmesser von ca. 2,5 cm besitzen.

3.5 Generation der Trigger

Zur Bildung des Triggers werden an mehreren Stellen des Detektors Hodoskope verwendet. Hierbei handelt es sich um horizontal und vertikal angeordnete

Streifen aus Szintillatormaterial, die an beiden Enden mit Photomultipliern ausgelesen werden. In Abb. 8 ist eine schematische Aufsicht auf den Detektor gezeigt, aus der die Positionen der Hodoskope ershen werden kann. Dabei stehen z. B. die Bezeichnungen H1H und H1V für die ersten Hodoskope im horizontalen bzw. vertikalen Richtung. Ein "L" am Ende der Hodoskopbezeichnung bedeutet, daß dieses in Strahlrichtung gesehen auf der linken Seite positioniert ist.

Der Level 0- oder Prettrigger erfordert lediglich eine Koinzidenz einzelner Hodoskopsegmente, während für den Level 1 Trigger zusätzlich noch eine Mindestenergie in den Hadronenkalorimeter deponiert werden muß. Der Level 1 Trigger löst die Auslese aller Detektorkomponenten aus. Es werden maximal $\text{ca. } 10^6$ Level 0 und 10^5 Level 1 Trigger pro Sekunde erwartet.

Der Myonentrigger für tiefinelastische Ereignisse zur Messung der Gluonpolarisation ΔG umfaßt eine Koinzidenz der Hodoskope H4V'' und H5V'' für kleine Energieüberträge bzw. eine Koinzidenz der Hodoskope H4VL und H5VL für große Energieüberträge. In beiden Fällen wird zusätzlich noch die Deposition einer Mindestenergie in beiden Hadronenkalorimetern gefordert.

Das *Trigger Control System* (TCS) verteilt die Triggersignale über eine optische Verbindung an alle Detektorkomponenten um die Auslese der Digitalisierungselektronik einzuleiten.

COMPASS MUON TRIGGER (top view)

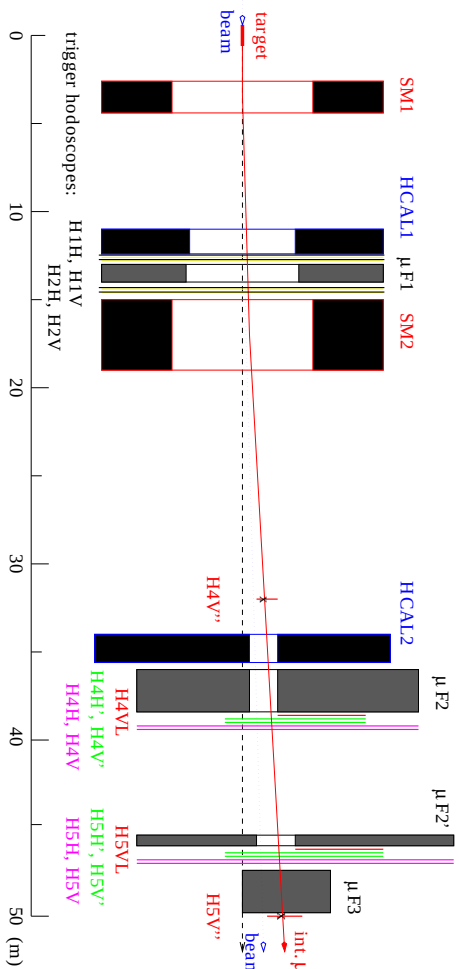


Abbildung 8: Schematische Darstellung der Triggerhodoskope.

4 Das COMPASS Datenauslesesystem

Die für das COMPASS-Experiment geplanten Messungen führen nur zu sinnvollen Ergebnissen, wenn Daten mit einer hohen Statistik gesammelt werden. Folglich ist es notwendig, bei den höchstmöglichen Strahlintensitäten eine sehr große Anzahl von guten Ereignissen aufzuzeichnen. Diese Anforderungen setzen ein Datenauslesesystem voraus, das keine, oder eine möglichst kleine Totzeit besitzt. Eine totzeitfreie Auslese ist bei COMPASS aufgrund der hohen Anforderungen an die Auflösung der digitalisierten Detektorinformationen nicht überall zu erfüllen. So führt z. B. die Digitalisierung analoger Signalamplituden oder Ladungssignale mit einer Auflösung von besser als 10 Bit eine Totzeit von 500 ns ein. Um weitere Wartezeiten zu vermeiden, werden die digital vorliegenden Daten ohne Rückmeldung zum jeweils nächsten Abschnitt des Datenaufnahmesystems weitergeleitet. Daher ist es sehr wichtig, eventuelle Übertragungsfehler am Ende dieser Kette eindeutig zu erkennen und die entsprechenden Ereignisse zu verwerfen.

Die Datenmenge, die vom Aufnahmesystem bewältigt werden muß, ist durch die Größe eines einzelnen Ereignisses und die Ereignisrate, also die Rate akzeptierter Trigger, bestimmt. Für COMPASS rechnet man mit Ereignisraten von bis zu 100 kHz. Bei einer erwarteten Ereignisgröße von ca. 30 kByte führt dies zu Datenraten von einigen Gigabyte pro SPS Spill, was sich zu einigen 100 Terabyte pro Jahr aufsummiert.

Um mögliche Störeinflüsse bei der Übertragung kleiner Signale auszuschließen und um Kosten für Kabel zu sparen sieht das Konzept der Datenaufnahme von COMPASS vor, die Informationen bereits am Detektor zu digitalisieren, soweit dies möglich ist. Verbindungen mit großer Bandbreite leiten die Daten zu Zwischenspeichern weiter, die in der Lage sind alle in einem Spill anfallenden Daten aufzunehmen. Von hier werden die Daten der verschiedenen Detektorkomponenten zwischen zwei Spills zu einem Computercuster weitergeleitet, wo sie zu kompletten Ereignissen zusammengesetzt werden. Dabei wird auch die Datenmenge reduziert, indem alle unbrauchbaren Ereignisse aussortiert werden, bevor die Daten zum CERN Rechenzentrum geschickt und auf Massenspeichermедien gesichert werden.

Diese Architektur, die den Datenstrom früh bündelt und dann über wenige Kanäle mit großer Bandbreite weiterleitet, erleichtert die Wartung des Datenaufnahmesystems. Weiterhin kann jederzeit problemlos neue Elektronik für zusätzliche Kanäle in das System eingefügt werden.

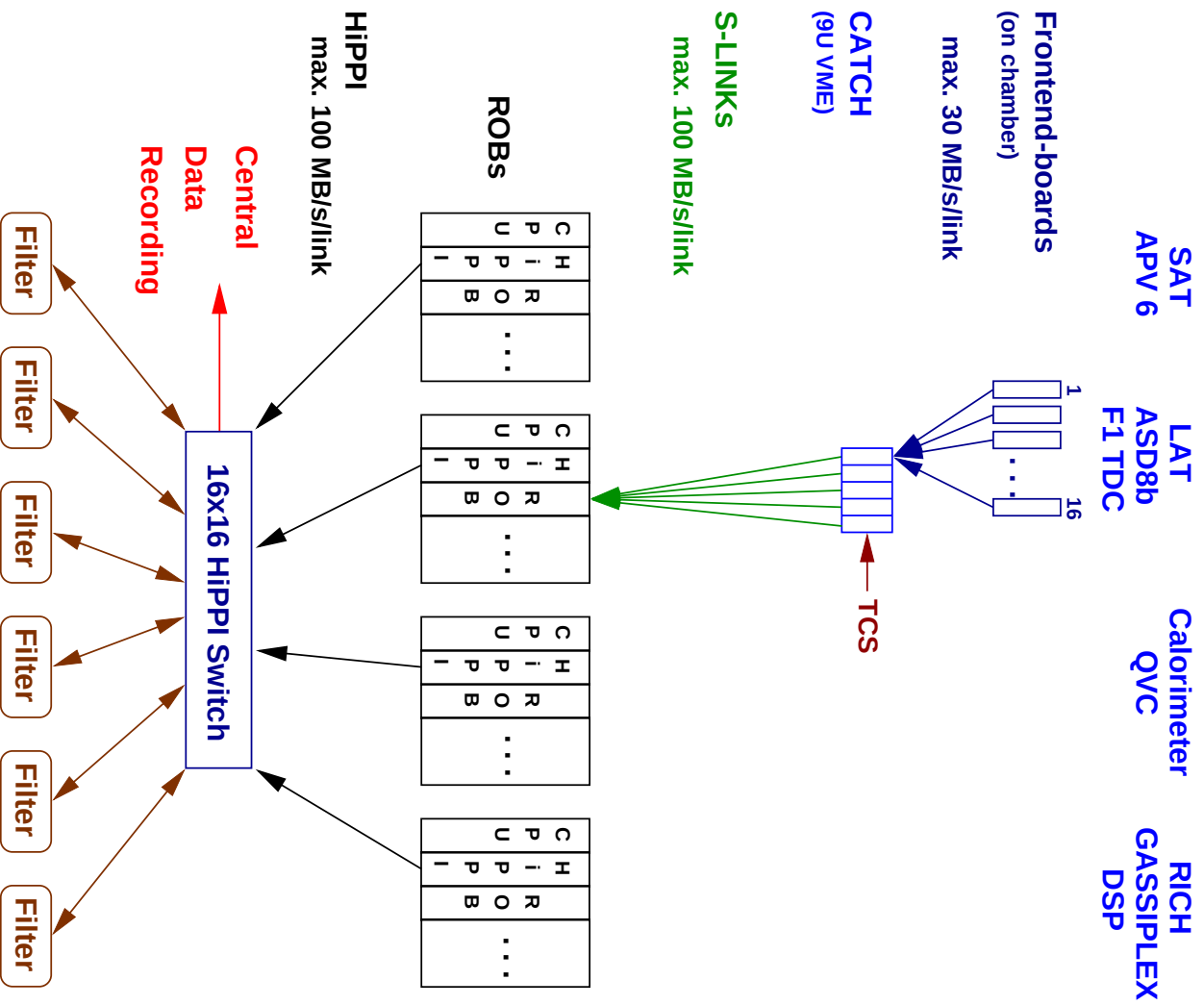


Abbildung 9: Übersicht über den Datenfluß bei COMPASS. Hier führt der Weg der Daten von den Detektoren im oberen Teil des Diagramms über vier Stufen zur Aufzeichnung auf Magnetband. Die einzelnen Stationen werden im Text näher erläutert.

Die Detektorinformationen durchlaufen auf ihrem Weg zum Magnetband vier Abschnitte, die im folgenden erläutert werden. Dabei liegt der Schwerpunkt auf der Auslese der Straw-Driftrohren, die in Abb. 9 dargestellt ist. Viele der in dieser Arbeit vorgestellten Konzepte wurden ursprünglich für diesen Teil des Datenauslesesystems entwickelt und dann später für die anderen Detektorkomponenten übernommen.

4.1 Elektronik an den Detektoren (“*Frontend*”)

Das Detektorsystem von COMPASS kann vom Standpunkt der Datenauslese in vier Typen aufgeteilt werden: Spurdetektoren für kleine Winkel (SAT: *Small Angle Tracking*), solche für große Winkel (LAT: *Large Angle Tracking*), Kalorimeter und RICH-Detektoren zur Identifikation geladener Teilchen.

Für die Digitalisierung der Detektorinformationen kommen im wesentlichen zwei Methoden zum Einsatz. Im Fall der RICH-Detektoren und der Kalorimeter werden analoge Signale mit ADC-Chips¹¹ digitalisiert, während für die Detektoren zur Spurrekonstruktion TDC-Chips¹² zum Einsatz kommen um Zeitinformationen zu digitalisieren. Die Bezeichnungen der einzelnen Chips, die bei COMPASS eingesetzt werden sollen sind in Abb. 9 unter den zugehörigen Detektorkomponenten aufgeführt. An dieser Stelle beschränkt sich die Beschreibung allerdings auf den Fall der Straw-Driftrohren, die zu den Spurdetektoren für große Winkel gehören.

Hier werden Frontend-Karten auf eine Platine am Detektor gesteckt, mit welcher die Signale der einzelnen Kanäle aus dem Detektor herausgeführt werden. Auf einer Frontend-Karte befinden sich ASD8b Vorverstärker [12] und F1-TDC-Chips [13] mit denen 64 Kanäle ausgelesen werden können. Die Vorverstärker verfügen über eine Diskriminatorstufe, die es erlaubt, nur Signale mit voreingestellter Mindestamplitude zum TDC durchzulassen. Der F1-Chip ist eine Neuentwicklung für die Auslese der Straw-Driftrohren und erlaubt die Messung von Driftzeiten mit einer sehr guten Auflösung. Zudem ist er sehr vielseitig und kann auch für die Auslese von z. B. Proportionalkammern oder Triggerhodoskopen verwendet werden.

Auf den Frontend-Karten ist ein Sender vorgesehen, der die digitalen Informationen seriell zur nächsten Stufe der Datenaufnahme, dem CATCH-System, transferiert. Als Übertragungsmedium werden S-UTP [14] Kabel eingesetzt, die auch in kommerziellen Ethernets weit verbreitet sind. Sie stellen vier differenzielle Datenleitungen zur Verfügung und werden bei COMPASS

¹¹ ADC, engl.: *Analog to Digital Converter*

¹² TDC, engl.: *Time to Digital Converter*

für Transferdaten von maximal 30 MByte/s pro S-UTP Verbindung verwendet werden.

Auch ein großer Teil der an anderen Detektoren angebrachten Frontend-Elektronik wird die Detektorinformationen digitalisieren bevor diese über S-UTP Kabel oder optische Verbindungen zum CATCH-System weitergeleitet werden. Dieses System bildet dann die gemeinsame Schnittstelle für alle Detektorkomponenten und leitet die Daten in nur noch einem einzigen Format weiter.

4.2 Das CATCH-System

Die Bezeichnung CATCH steht für *COMPASS Accumulate, Transfer and Cache Hardware*. Das CATCH-System ist auf 9U großen Platinen für ein VMEbus System¹³ implementiert und erfüllt gleichzeitig drei verschiedene Aufgaben:

1. die Auslese der Frontend-Elektronik,
2. die Verteilung von experimentweiten Befehlen, wie Triggersignalen,
3. die Initialisierung eines Teils der Frontend-Elektronik.

Um das CATCH-System so vielseitig wie möglich zu halten und um den Austausch einzelner Komponenten zu erleichtern, werden die VME-Karten so entwickelt, daß sie am detektorseitigen Dateneingang mit vier Mezzanine-Karten (gemäß dem IEEE CMC Standard [15]) bestückt werden können. Derzeit sind vier verschiedene Typen von Mezzanine-Karten vorgesehen:

- **Typ 1** wird mit vier HOTLink Empfängerchips [16] bestückt, die den seriellen Datenstrom bereits digitalisierter Detektorinformationen zur weiteren Verarbeitung parallelisieren. Diesen Kartentyp wird es in zwei Varianten geben, welche entweder die Verwendung von S-UTP Kabeln oder optischen Verbindungen für die Verbindung zur Frontend-Elektronik erlauben.
- **Typ 2** ist mit acht $\mathcal{F}1$ -TDCs ausgerüstet um Detektoren auszulesen, deren Informationen erst hier digitalisiert werden sollen.
- **Typ 3** besitzt Zähler, die ohne Totzeit ausgelesen werden können und mit einer maximalen Frequenz von 200 MHz zählen.

¹³Eine kurze Beschreibung der VMEbus-Architektur wird in Abschnitt 5.2 gegeben.

- **Typ 4** bildet die Schnittstelle zur Elektronik für die strahldefinierenden Siliziumstreifenzähler.

Für die Auslese des gesamten Detektorsystems von COMPASS werden ungefähr 160 CATCH-Platinen benötigt. Aus Tabelle 2 kann entnommen werden, wie sich diese auf die einzelnen Komponenten des Detektors verteilen. Dabei fällt auf, daß der größte Teil der CATCH-Module für die Auslese der verschiedenen Spurdetektoren zuständig ist. Von diesen werden u. a. diejenigen Module mit Mezzanine-Karten des Typs 1 ausgerüstet, welche die Daten von den Straw-Driftrohren verarbeiten.

Tabelle 2: Aufteilung der ca. 160 für COMPASS benötigten CATCH-Module auf die unterschiedlichen Detektorkomponenten.

Detektor	Zahl der CATCH-Module	Mezzanine Kartentyp
Straw-Driftrohren und Driftkammern	50	1
Vieldraht-Proportionalkammern	20	1
Myonenkammern	8	1
Mikrostreifen-Gaszähler	45	4
Scintillierende Fasern	5	2
RICH Detektoren	16	1
Kalorimeter	16	1

Im Folgenden sollen die Vorgänge auf einer CATCH-Platine wieder exemplarisch anhand der Auslese der Straw-Driftrohren beschrieben werden. In diesem Fall empfängt jede CATCH-Platine die Daten von 16 Frontend-Karten, was 1024 Kanälen entspricht. Für diesen Datentransfer wird nur eines der vier differentiellen Leitungspaare benötigt, die das S-UTP Kabel zur Verfügung stellt. Mit Hilfe der restlichen drei Leitungspaare werden Informationen vom CATCH-System zur Frontend-Elektronik übertragen.

Im Experiment wird es ein sogenanntes *Trigger Control System* (TCS) geben. Dieses hat die Aufgabe, über optische Verbindungen Triggersignale an alle Detektorkomponenten zu verteilen. Neben den Triggersignalen sind noch drei weitere Befehle vorgesehen, von denen einer ein Resetsignal ist, das zu Beginn eines Spills die Hardware an den Detektoren synchronisiert. Über die Verwendung der beiden restlichen Signale kann noch frei entschieden werden. Diese vier *TCS-Befehle* werden vom Trigger Control System

zusammen mit einer Uhr¹⁴ und der aktuellen Triggernummer an die CATCH-Module geliefert. Die Clock besitzt aus Gründen der Kompatibilität zu LHC-Experimenten eine Frequenz von 38,88 MHz und hat zwei Funktionen. Zum einen dient sie der experimentweiten Synchronisation der Elektronik. Daneben werden u. a. im Fall der Straw-Driftrohrchenaulese auf dem CATCH-Modul mit ihrer Hilfe die TCS-Befehle kodiert bevor sie über ein Leitungs-paar des S-UTP Kabels zur Frontend-Elektronik gesendet werden. Die Kodierung der TCS-Befehle ist notwendig, da alle vier Signale über eine einzige Leitung gesendet werden, wobei ihre Übertragung synchron zum Clocksignal erfolgt, für das ein eigenes Leitungs-paar auf dem Kabel verwendet wird. Auf diese Weise kann die Frontend-Elektronik die TCS-Befehle wieder entschlüsseln und ausführen.

Die Daten von der Frontend-Elektronik werden auf der CATCH-Platine gesammelt und nach Ereignissen sortiert. Dazu werden für jedes Trigger-signal sowohl auf der CATCH-Platine, als auch auf der Frontend-Karte unabhängige, lokale Triggernummern generiert. Treffen nun neue Daten von einer Frontend-Karte auf dem CATCH-Modul ein, so wird ihre lokal generierte Triggernummer mit der vom Frontend gesendeten Triggernummer verglichen und dann durch die vom Trigger Control System gelieferte, experimentweite Triggernummer ersetzt. Bevor die Daten in 32 Bit breite Worte reformatiert werden, wird ihnen noch eine Adresse hinzugefügt, die ihre Herkunft spezifiziert. Erst wenn zu einem Ereignis von allen verbundenen Frontend-Karten auf der CATCH-Platine Daten vorliegen, werden diese zusammen mit einem *Header*, welcher Informationen über Ereignis- und Spillnummer, sowie über ihre Herkunft enthält, über eine schnelle S-LINK Verbindung [17] zur nächsten Stufe der Datenaufnahme geschafft. Der S-LINK ist eine optische Verbindung, die einen maximale Datentransfer von 100 MByte pro Sekunde ermöglicht.

Die verschiedenen Frontend-Karten werden unterschiedliche Datenmen-gen produzieren. Daher muß das CATCH-System in der Lage sein, mehrere Ereignisse zwischenspeichern zu können. Mit dem CATCH-System kann auch ein “Fast Clear” höherer Stufe implementiert werden: dazu müssen die Daten eines Ereignisses solange auf dem CATCH-Module zwischengespeichert werden bis das Trigger Control System die Entscheidung liefert ob sie verwendet werden sollen oder nicht.

Um den Austausch defekter Frontend-Elektronik zu erleichtern und um Bedienungsfehler auszuschließen, soll die *Initialisierung* der Frontend-Elektronik an den Straw-Driftrohren weitgehend automatisch ablaufen. Nach Ein-

¹⁴Im folgenden wird anstelle des Wortes “Uhr” immer der Begriff “Clock” verwendet.

schalten schicken die an den Detektoren angebrachten Platinen anstelle von Daten zwei eindeutige Adressen zur Identifikation: die Seriennummer der Frontend-Platine selber und eine Nummer die ihren Steckplatz auf dem Detektor spezifiziert. Die CATCH-Module erkennen diese Adressen und fordern über den VMEbus aus einer Datenbank die notwendigen Informationen zur Initialisierung der Frontend-Elektronik an. Dazu gehören u. a. Spannungen für Diskriminatorschwellen oder der Betriebsmodus von z. B. dem *J1-TDC*. Diese Daten werden über das vierte und letzte Leitungspaar auf dem S-UTP Kabel mit einer Datenrate von 10 MBit pro Sekunde an die Frontend-Elektronik übermittelt.

Das CATCH-System nutzt den VMEbus nur als Stromversorgung und zur Übermittlung der Initialisierungsdaten von einer zentralen Datenbank an die CATCH-Module. Allerdings besteht unabhängig vom Datenaufnahmesystem weiterhin die Möglichkeit, Stichproben von Ereignissen über den VMEbus parallel zum S-LINK auszulesen um schnell zu prüfen, ob Fehler bei der Detektorauslese auftreten.

4.3 Zwischenspeicherung und Filterung der Daten

Mit maximal 64 S-LINK Verbindungen, die eine Bandbreite von ungefähr 6 GByte pro Sekunde zur Verfügung stellen, werden die Daten vom CATCH-System zu den sogenannten *Read Out Buffern* (ROB) übertragen. Dabei empfängt jeder dieser 16 Zwischenspeicher die Daten von vier S-LINK Verbindungen. Um nun alle 160 CATCH-Module mit 64 S-LINKS auslesen zu können, werden 32 dieser Verbindungen an ein einzelnes CATCH-Modul angeschlossen, während die anderen 32 S-LINKS die Daten von jeweils vier CATCH-Modulen zu den *Read Out Buffern* übertragen.

Diese großen Zwischenspeicher sind in der Lage alle in einem Spill anfallenden Daten aufzunehmen. An dieser Stelle können die einzelnen Spillpuffer erste Konsistenzchecks durchführen und überprüfen, ob alle angeschlossenen CATCH-Module gültige Daten geliefert haben, bevor sie diese in den zwölf Sekunden zwischen zwei Spills zum Datenfiltersystem weiterleiten.

Die Verbindung zwischen den Spillpuffern und dem Filtersystem ist durch einen HiPPI-Switch¹⁵ [18] implementiert. Über diesen kann von jedem Spillpuffer eine Punkt-zu-Punkt Verbindung zu einem Rechner des Filtersystems hergestellt werden, die einen Datentransfer von 100 MByte pro Sekunde erlaubt. So wird eine weitere Ordnung der Daten möglich: der HiPPI-Switch erkennt die im Übertragungsprotokoll enthaltenen Triggernummern und sendet alle zur gleichen Triggernummer gehörenden Daten zum gleichen Rechner

¹⁵Switch steht hier für Verteiler

des Filtersystems. Hier werden dann die Ereignisse endgültig nach Detektor-komponenten geordnet und auf Vollständigkeit überprüft.

Die Rechner des Datenfiltersystems reduzieren die Datenmenge, indem physikalisch unvollständige Ereignisse verworfen werden. Weiterhin wird hier entschieden, ob vollständige Ereignisse für eine spätere Auswertung genutzt werden können indem z. B. Spuren rekonstruiert werden. Besonders interessante Ereignisse könnten vom Filtersystem gekennzeichnet werden, um die Auswertung zu erleichtern.

Eine Verbindung des HiPPi-Switch leitet die gefilterten Daten zum CERN Rechenzentrum weiter, wo sie dann auf Magnetbändern gesichert werden.

5 Das CATCH1-Modul

Wie aus Abschnitt 4 hervorgeht, spielt das CATCH-System in der Datenaufnahme von COMPASS eine zentrale Rolle. Im Rahmen dieser Arbeit soll nun die CATCH1-Platine vorgestellt werden. Sie ist gewissermaßen der Prototyp der Karten, die später im Experiment eingesetzt werden.

Bevor näher auf die technische Seite dieser Platine eingegangen wird, sollen in den nächsten Abschnitten ihre wesentlichen Eigenschaften und die damit verbundenen Möglichkeiten vorgestellt werden.

5.1 Ziele der Entwicklung

Mit der Entwicklung dieser Platine wurden zwei Ziele verfolgt. Zum einen ist es im Hinblick auf das Experiment notwendig zu überprüfen, ob das Konzept der CATCH-Module technisch zu realisieren ist. Ist dies der Fall, kann mit Hilfe des Prototypen untersucht werden, an welchen Stellen Probleme auftreten könnten. Für diese Zwecke sollte die Platine möglichst viele Eigenschaften der endgültigen Version besitzen.

Auf der anderen Seite wollte man diese Karte aber auch für konkrete Anwendungen nutzbar machen. Aus diesem Grund sind auf der CATCH1-Karte Komponenten implementiert, die es im Experiment nicht mehr geben wird.

So wird es mit dem CATCH1 möglich sein, ein Testsystem aufzubauen, welches die ebenfalls in Freiburg entwickelten Platinen für die Frontend-Elektronik der Straw-Driftrohren auf ihre Funktionalität überprüft.

5.1.1 Das CATCH1-Modul als Prototyp

In seiner Eigenschaft als Prototyp für das CATCH-System von COMPASS verfügt der CATCH1 im wesentlichen über drei Eigenschaften, die auch bei der Hardware für die Auslese der Straw-Driftrohren im Experiment zu finden sein werden. Dabei handelt es sich um

1. die Auslese serieller Daten vom Frontend,
2. die Initialisierung der Frontend-Elektronik,
3. die Übermittlung von TCS-Befehlen an das Frontend.

Der CATCH1 besitzt einen HOTLink-Empfängerchip und kann damit den seriellen Datenstrom von einer Frontend-Karte auslesen. Die auf einem

S-UTP Kabel eintreffenden Daten werden in 8 Bit breite Worte umgewandelt und zur Zwischenspeicherung in einem 9 Bit breiten FIFO-Puffer weitergeleitet. Das neunte Bit wird vom HOTLink geliefert und zeigt an, ob bei der seriellen Übertragung ein Fehler aufgetreten ist. Im Gegensatz zum Experiment werden die Daten vom Frontend auf dem CATCH1 nicht weiter bearbeitet, sondern müssen direkt von einem Computersystem über den VMEbus ausgelesen und analysiert werden.

Zur Initialisierung der Frontend-Karte stellt der CATCH1 eine serielle Schnittstelle zur Verfügung, die alle für das Experiment geforderten Eigenschaften besitzt und über den VMEbus angesteuert werden kann.

Eine Schnittstelle für die Befehle, die im Experiment vom Trigger Control System geliefert werden, war in der Entwicklungsphase des CATCH1-Moduls noch nicht definiert. Aus diesem Grund wurde sie hier einfach als ECL-Eingang ausgelegt. Die vier TCS-Befehle können aber auch über den VMEbus an den CATCH1 übermittelt werden. In beiden Fällen verlaufen die Kodierung der Signale und ihre Übermittlung an die Frontend-Elektronik genauso, wie es für die Module im Experiment geplant ist. Allerdings wird das dazu notwendige Clocksignal auf der CATCH1-Platine selbst generiert und besitzt der Einfachheit halber eine Frequenz von 40 MHz.

Im Gegensatz zu den CATCH-Modulen im Experiment verhält sich der CATCH1 völlig passiv und trifft keine eigenen Entscheidungen. Abgesehen von den TCS-Befehlen wird er komplett von einem Computer über den VMEbus angesteuert. Dort müssen dann z. B. auch die Daten von der Frontend-Karte nachbearbeitet und in ein für ihre Analyse richtiges Format gebracht werden.

5.1.2 Das CATCH1-Modul als Testmöglichkeit für Frontend Elektronik

Die CATCH1-Platine bietet neben ihrer Funktion als Prototyp die Möglichkeit, einen Aufbau zum Testen von Frontend-Elektronik zu realisieren. Der in Abb. 10 gezeigte Aufbau ist in erster Linie für die Frontend-Karten der Straw-Driftrohren gedacht, die auch an der Freiburger Universität entwickelt werden. Allerdings ist das Design des CATCH1 so vielseitig gehalten, daß das Testsystem leicht an andere Anforderungen angepaßt werden kann.

Mit Hilfe der CATCH1-Platine können definierte Ladungsimpulse generiert werden, welche die Signale eines Detektors simulieren sollen. Dazu übernimmt ein Rechner über den VMEbus ein aus 64 Bit bestehendes Muster an den CATCH1.

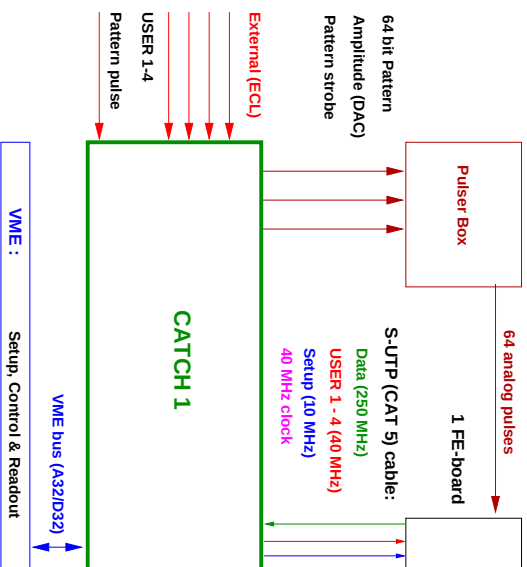


Abbildung 10: Möglicher Aufbau zum Test von Frontend-Karten.

Das CATCH1-Modul kann jeweils ein Bitmuster zur Zeit zwischenspeichern, um damit entsprechende Kanäle in einem externen Gerät zu selektieren, welches für die Generation der Ladungsimpulse zuständig ist. Diese “Pulser Box” wurde speziell auf die Anforderungen der ASD8b Vorverstärker auf den Straw-Frontend-Karten zugeschnitten und steht in zwei Varianten zur Verfügung. Beide Typen verfügen über einen AD7845 Digital-zu-Analogkonverter (DAC) der Firma Analog Devices [19], der eine Auflösung von 12 Bit besitzt und für die Generation analoger Signale zuständig ist. Diese besitzen für jeden selektierten Kanal die gleiche, über den CATCH1 festgelegte Amplitude. Die beiden Versionen der Pulser Box haben folgende Eigenschaften:

- **Typ 1** wandelt die analogen Signale mit Hilfe von Kondensatoren in Ladungsimpulse um. Für diese kapazitiv erfolgende Ladungserzeugung gilt $Q = C \cdot U$, wobei die erzeugte Ladung Q das Produkt der Kapazität C und der Signalspannung U ist. Hier sind Ladungsimpulse von bis zu 200 fC möglich.
- **Typ 2** bedient sich eines Widerstandsnetzwerkes zur Ladungsimpuls-generation, wobei für die erzeugte Ladung $Q = U/R \cdot t$ gilt. Bei einer fest vorgegebenen Signaldauer t können über den Widerstand R Ladungsimpulse von bis zu 2 pC erzeugt werden. Sollten noch größere Ladungsimpulse nötig sein, so kann die Signaldauer durch Änderungen an der CATCH1-Platine leicht erhöht werden.

Die zu testende Frontend-Karte steckt auf der Pulser Box und ist durch ein S-UTP Kabel mit dem CATCH1-Modul verbunden. Nach ihrer Initialisierung kann die Frontend-Karte die künstlichen Detektorsignale auslesen und schickt ihrerseits wieder Daten zum CATCH1. Nun kann der Computer die Daten über den VMEbus zurücklesen und mit den vorgegebenen Einstellungen vergleichen. Alle notwendigen Steuerbefehle, sowie die Triggersignale kommen entweder vom Rechner oder können über Stecker auf der Frontplatte des Moduls eingespeist werden.

Dieser Aufbau erlaubt einen weitgehend automatisierten Testablauf. Nach dem Einstecken und Verbinden einer Frontend-Karte übernimmt ein Computerprogramm alle notwendigen Schritte. Die Testergebnisse können in eine Datei geschrieben und später ausgewertet werden. Man kann überprüfen, ob alle Kanäle normal arbeiten, ob es ein Übersprechen zwischen einzelnen Kanälen gibt oder das Verhalten der Diskriminatorstufe des ASD8b für unterschiedliche Schwellenspannungen untersuchen. Damit wird es möglich sein, die vielen Frontend-Karten vor ihrem Einsatz im Experiment eingehend auf ihre Funktionstüchtigkeit zu überprüfen.

5.2 Technische Realisierung

In diesem Abschnitt wird vorgestellt, wie die CATCH1-Karte realisiert wurde. Dazu ist es notwendig, zunächst die Funktion wichtiger Bestandteile der Platine vorzustellen.

5.2.1 Der VMEbus

Der VMEbus ist eine Standardarchitektur für Computersysteme, wobei das Kürzel für *Versa Module Eurocard* steht. Er wurde 1981 von mehreren Herstellern, zu denen u. a. Motorola gehörte, entwickelt und ist in der Industrie und Forschung weit verbreitet. Kernstück ist ein Übernahmen, der Platz für bis zu 21 Module bietet, diese über Stecker und Leitungen auf seiner Rückwand miteinander verbindet und alle notwendigen Spannungen zur Verfügung stellt.

Beim VMEbus handelt es sich um eine sogenannte *Master-Slave Architektur*. Das bedeutet, daß aktive *Master*-Module Kontrolle über den Datentransferbus übernehmen können um Informationen zu oder von den *Slave*-Modulen zu transferieren. Die passiven *Slave*-Module reagieren dabei nur auf die Anfragen eines Masters. Zu den Master-Modulen zählen z. B. Karten, auf denen sich ein Mikroprozessor befindet, während Slaves einfache

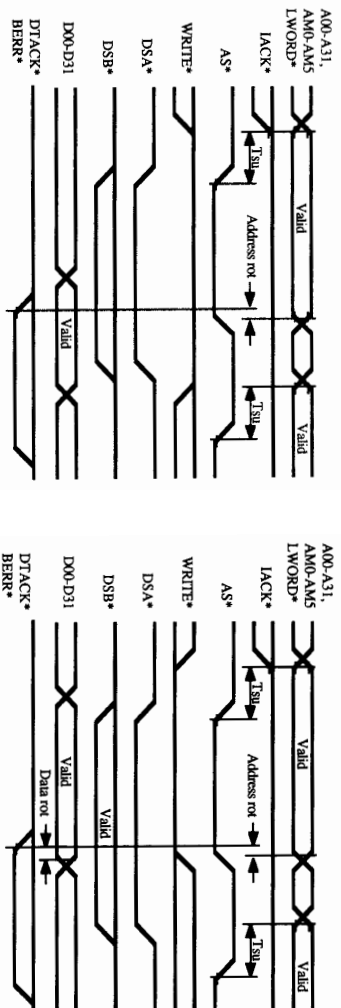


Abbildung 11: Zeitliche Abfolge der Signale auf dem VME Datentransferbus. Das linke Diagramm zeigt einen Lesezugriff, das rechte zeigt einen Schreibzugriff (Aus [20]).

Speichermodule sein können. Da ein Übernahmen mehrere Master-Module aufnehmen kann, wird der VMEbus auch als Multiprozessorbus bezeichnet. Während gebräuchliche Module in der Größe 6U erhältlich sind, werden bei COMPASS auch die größeren 9U Module verwendet. Sie bieten eine Fläche (Höhe x Tiefe) von 233 x 160 mm² (6U) bzw. 367 x 400 mm² (9U).

Seit seiner Einführung wurde der VMEbus ständig weiterentwickelt, wobei die daraus resultierenden Neuerungen jeweils in einer neuen Version der *VMEbus Spezifikationen* [21] festgehalten werden. Dabei wird darauf geachtet, daß die neue Spezifikation mit älteren Systemen kompatibel bleibt. Ferner sind hier strenge Richtlinien für die Entwicklung neuer Module enthalten, deren Aufgabe es ist, ein höchstmögliches Maß an Kompatibilität zwischen den Produkten unterschiedlicher Entwickler zu gewährleisten.

Der VMEbus besitzt eine Reihe von Unterbussen, die unterschiedliche Aufgaben übernehmen. Diese sind der *Data Transfer Arbitration Bus*, der *Priority Interrupt Bus*, der *Utility Bus* und der *Data Transfer Bus*. Während die ersten drei Unterbussse für die Zuteilung des Datentransferbusses an verschiedene Master, die Behandlung von Interrupts und die Verteilung von Systemsignalen zuständig sind, ist für diese Arbeit nur der Datentransferbus von Interesse.

Beim Datentransferbus handelt es sich um einen 32 Bit breiten Datenbus, der einen ebenfalls 32 Bit großen Adreßraum verwalten kann. Neuere Versionen verfügen für beide Fälle schon über 64 Bit breite Verbindungen. Der Datentransfermechanismus arbeitet asynchron. Das bedeutet, daß zur Koordinierung der Datenübergabe kein Taktgeber, sondern eine Reihe von

Steuersignalen verwendet wird, die beim Datenaustausch zwischen zwei Modulen einen *“Handshake”* ermöglichen. Die maximale Geschwindigkeit eines asynchronen Busses wird durch die Signallaufzeiten auf der Rückwand des Übertrahmens (diese kann innerhin bis zu einem halben Meter breit sein) und vor allem durch Pufferchips bestimmt. Diese Puffer enthalten die Schnittstelle zum VMEbus und regeln den Datenverkehr auf einem Modul. Der VMEbus kann Übertragungsraten von bis zu 40 MByte pro Sekunde (80 MByte/s für 64 Bit breite Datenverbindungen) erreichen. Es stehen insgesamt vier verschiedene Übertragungsarten zur Verfügung:

1. einfache Lese- und Schreibzyklen (*Read/Write Cycles*)
2. “unteilbare” Lese- und Schreibzyklen (*Read-Modify-Write Cycle*)
3. Blocktransfer
4. Zyklen, die nur eine Adresse übertragen (*Address-Only Cycle*)

Das zeitliche Verhalten aller für einfache Lese- und Schreibzyklen wichtigen Signale ist in Abb. 11 dargestellt. Hierzu sendet der Master zu Beginn des Zyklus die Adresse (Leitungen A00 bis A31) des anzusprechenden Slaves. Diese wird mit dem Signal AS*¹⁶ (“Adressen-Strobe”) als gültig deklariert und vom Slave entschlüsselt. Die Signale AM0 bis AM5 und LWORD* zeigen an, um welche Art von Übertragung es sich handelt, während der Zustand des Signals WRITE* angibt, ob gelesen oder geschrieben werden soll. Im Fall eines Lesezugriffs setzt nun der Master die Signale DS0* und DS1* (“Daten-Strobe”) auf Null um anzuzeigen, daß er bereit ist Daten zu empfangen. Daraufhin stellt der Slave bis zu 32 Bit auf den Leitungen D00-D31 zur Verfügung und erklärt sie mit dem Signal DTACK* für gültig. Nach Übernahme der Daten nimmt der Master DS0* und DS1* zurück, woraufhin das Zurücksetzen des Signals DTACK* durch den Slave den Zyklus beendet. Reagiert das Slave-Modul zu spät oder gar nicht auf eine Adresse, so wird der Transferzyklus mit dem Signal BERR* beendet, was einen Fehler anzeigt. Im Schnitt dauert solch ein Datentransferzyklus eine Mikrosekunde.

Bei den sogenannten “unteilbaren” Lese- und Schreibzugriffen liest ein Master aus einer Adresse Daten, kann diese bearbeiten und sofort wieder zurückschreiben, ohne daß einem anderen Master in der Zwischenzeit Zugriff

¹⁶Der Stern bezeichnet immer Signale mit inverser Logik - sie sind aktiv, wenn sie Null sind.

auf diese Adresse gewährt wird. Sollen keine Daten übertragen werden, so reicht in manchen Fällen das Senden einer Adresse. Da hier kein Handshake erfolgt sind diese Zyklen sehr schnell. Die maximale Übertragungsgeschwindigkeit kann allerdings nur mit dem Blocktransfer erreicht werden. Hier ist es nur einmal nötig das Slave-Modul zu adressieren bevor ein großer Datenblock übertragen wird.

Da u. a. die Schnittstelle zum VMEbus auf dem CATCH1 in einem programmierbaren Logikbaustein implementiert ist, sollen im folgenden einige programmierbare Chips und ihre Eigenschaften vorgestellt werden.

5.2.2 Programmierbare Logikbausteine

Die fortschreitende Entwicklung in der Halbleitertechnologie hat integrierte Schaltkreise hervorgebracht, die es dem Benutzer erlauben, die Implementierung logischer Schaltungen selber vorzunehmen. Damit ist es nicht mehr notwendig, diese Arbeiten an einen Halbleiterhersteller zu übergeben, sofern die erforderliche Logik einen gewissen Grad an Komplexität nicht überschreitet.

Programmierbare Logikbausteine verfügen über eine große Zahl von logischen Verknüpfungen in Form von sogenannten Gattern, wobei es sich z. B. um logische UND-Verknüpfungen handeln kann. Die Gatter sind in den Chips schon zu komplexeren Strukturen, wie Flip-Flops oder Multiplexern zusammengefasst, die sich nach den Anforderungen der zu implementierenden Logik miteinander verbinden lassen.

Bei der Entwicklung von Platinen eröffnen programmierbare Logikbausteine dem Benutzer eine Reihe von Vorteilen gegenüber der Verwendung von Standardchips mit TTL- oder CMOS-Technologie. So kann eine Schaltung genau auf die Anforderungen zugeschnitten werden, und nimmt dabei weniger Platinenplatz in Anspruch. Weiterhin besteht die Möglichkeit diese Chips jederzeit neu zu programmieren. Dies kann in den meisten Fällen selbst dann noch geschehen, wenn sie schon auf der Platine verlötet sind. Damit wird es leichter Schaltungsfehler zu entdecken und diese noch in der Testphase zu korrigieren. Dadurch kann der Entwicklungsprozeß erheblich beschleunigt werden.

Auf der CATCH1-Platine kommen drei verschiedene programmierbare Logikbausteine zum Einsatz, deren Eigenschaften nun etwas genauer beleuchtet werden sollen.

- ***Generic Array Logic: GAL***

Der einfachste programmierbare Baustein, der auf der CATCH1-Platine Verwendung findet, ist ein GAL der Firma Lattice. Der GAL22V10C [22] ist ein vergleichsweise kleiner Chip, mit dem sich nur Schaltungen realisieren lassen die wenig Logik erfordern. Die Bezeichnung "22V10" in seinem Namen steht für 22 mögliche Eingangssignale, denen 10 mögliche Ausgangssignale gegenüber stehen. Aufgrund seiner geringen Größe sind Signallaufzeiten kurz.

Dieser Chip verfügt über einige Flip-Flops und ist in der Lage, seine Konfiguration ohne Spannungsversorgung zu halten. Außerdem kann er neu programmiert werden, um andere Aufgaben zu übernehmen.

- ***Complex Programmable Logic Device: CPLD***

Der auf der CATCH1-Platine verwendete XC95288 [23] ist ein CPLD-Baustein der Firma Xilinx. Seine Architektur ist in Abb. 12 wiedergegeben. Sie unterteilt sich in im wesentlichen in drei Komponenten: Blöcke, die für die Ein- und Ausgabe von Signalen zuständig sind (*I/O Blocks*), Funktionsblöcke und eine Verbindungsmatrix.

Die I/O-Blöcke stellen die Verbindung zwischen interner Logik und den Beinchen des Chips her. Dabei können alle für den Benutzer verfügbaren Beinchen zur Ein- oder Ausgabe von Signalen verwendet oder deaktiviert werden. Im deaktivierten Zustand spielt es keine Rolle, was für ein Signal am Beinchen anliegt. Dies kann nützlich sein, wenn die zum Beinchen führende Leitung z. B. zu einem Datenbus gehört und von mehreren Quellen mit Signalen getrieben werden kann.

Die I/O-Blöcke sind mit den Funktionsblöcken über eine Matrix verbunden, die eine programmierbare Verbindung zweier beliebiger Blöcke erlaubt. Jeder Funktionsblock besteht aus 18 Makrozellen, welche die eigentliche Logik enthalten. Die 18 Makrozellen können zusammen 36 Signale verarbeiten. Dabei kann in jeder Makrozelle ein Ergebnis der miteinander verknüpften Signale entweder erst in einem Flip-Flop zwischengespeichert oder sofort wieder ausgegeben werden. Die 18 Ausgänge der Makrozellen können sowohl auf die Verbindungsmatrix und von dort in einen anderen Logikblock gelangen, als auch zu den I/O-Blöcken führen. Weiterhin sind spezielle Leitungen für z. B. Clocksignale vorgesehen, welche eine schnelle Verteilung dieser Signale an alle Komponenten des Chips ermöglichen.

Die Architektur dieses CPLD verfügt damit über viele Blöcke, die mit einem GAL-Baustein vergleichbar sind, allerdings noch etwas mehr Logik

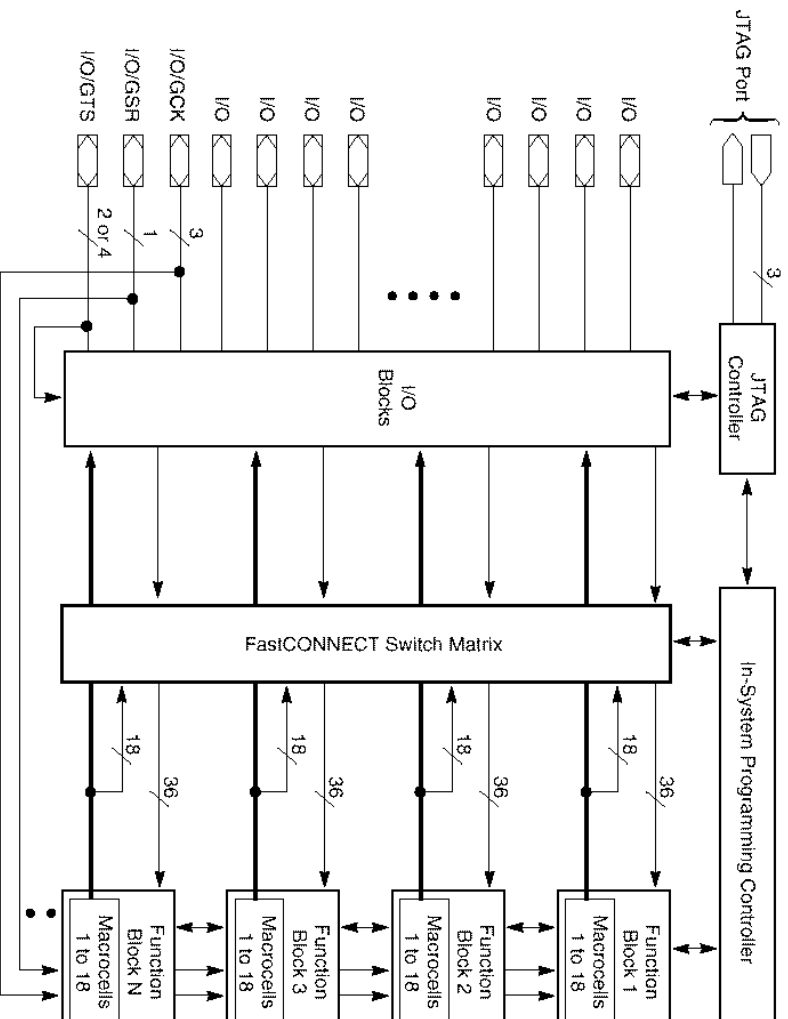


Abbildung 12: Architektur des Xilinx XC95288 CPLDs (Aus [23]).

zur Verfügung stellen. Insgesamt stehen auf dem verwendeten Chip in 288 Makrozellen 6400 Gatter zur Nutzung bereit.

Mit dieser Architektur lassen sich bereits sehr komplexe logische Schaltungen verwirklichen, die zudem den Vorteil haben, daß die Signallaufzeiten von einem Eingangs- zu einem Ausgangsbeinchen genau definiert sind. Für den kürzesten Weg durch eine einzige Makrozelle benötigt ein Signal im verwendeten Chip 15 ns. Durchläuft das Signal auf seinem Weg die Logik mehrerer Makrozellen, so erhöht sich seine Laufzeit um einen bekannten Betrag.

Einmal programmiert, behalten diese Chips ihre Informationen auch nach Ausschalten der Versorgungsspannung. Sie können allerdings elektrisch gelöscht werden und sind dann bereit, eine neue logische Schaltung aufzunehmen. Dies ist selbst dann noch möglich, wenn ein CPLD schon auf der Platine verlötet ist. Für die Programmierung werden die speziellen, mit JTAG-Port bezeichneten Beinchen verwendet.

- ***Field Programmable Gate Array: FPGA***

Diese Bausteine unterscheiden sich in ihrer Arbeitsweise grundlegend von den CPLDs. Abb. 13 zeigt die prinzipielle Struktur eines FPGAs der Firma Xilinx. Auch hier gibt es zwei wesentliche Typen funktioneller Einheiten: die konfigurierbaren Logikblöcke und Blöcke für die Ein- und Ausgabe von Signalen. Im Gegensatz zur CPLD-Architektur liegen hier die konfigurierbaren Logikblöcke selber auf der Matrix, durch die sie miteinander verbunden werden. Somit wird es nun möglich, mehrere Logikblöcke beliebiger Position auf dem Chip hintereinanderschalten, um damit sehr komplexe logische Schaltungen zu realisieren. Das hat allerdings zur Folge, daß sich das zeitliche Verhalten der Signale nicht mehr von vornherein genau festlegen läßt.

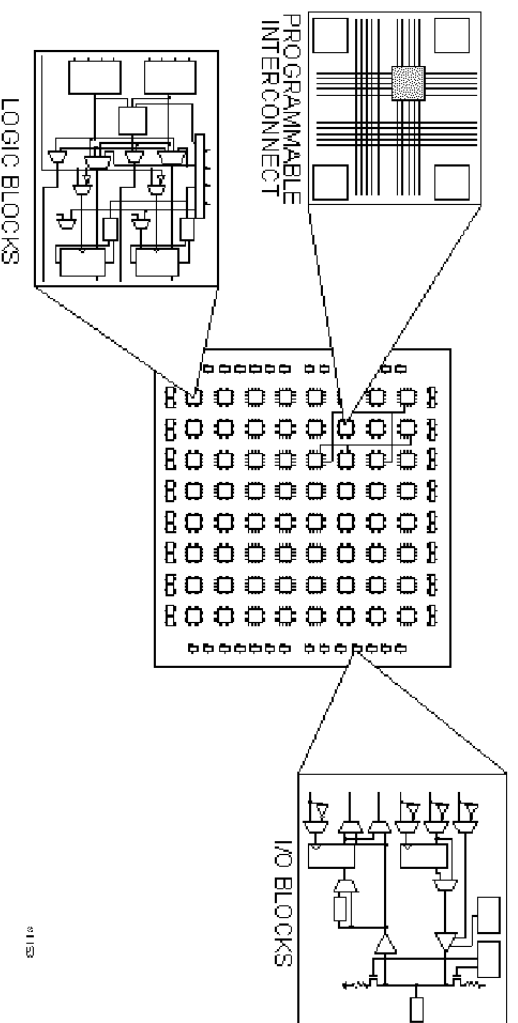


Abbildung 13: Architektur des Xilinx XC4020E FPGAs (Aus [24]).

Dafür bieten FPGAs aber eine erheblich größere Zahl von Gattern als die CPLD-Bausteine. So verfügt z. B. der auf der CATCH1-Platine verwendete Chip über ein Äquivalent von 20.000 logischen Gattern.

Da FPGAs ihre Konfiguration in SRAM-Strukturen speichern, verlieren sie bei Wegfall der Versorgungsspannung ihre Informationen und müssen bei jedem Neustart konfiguriert werden.

- **Software zur Entwicklung von Schaltungen**

Für die Entwicklung logischer Schaltungen stehen eine Reihe von Möglichkeiten zur Verfügung. Hier wurde das Softwarepaket *XACTstep* der Firma

Xilinx ausgewählt, das eine komplette Entwicklungsumgebung bietet und auf einem PC unter Windows NT läuft.

Kernstück dieser Software ist der *Project Manager*, der einen schnellen Überblick über den aktuellen Stand der Entwicklung liefert. Von hier können alle notwendigen Programme aufgerufen werden. Dazu gehört ein Editor, der eine schematische Eingabe von Schaltplänen erlaubt und über große Bibliotheken bereits vordefinierter Makros verfügt, die an die Gegebenheiten des Chips angepaßt sind, auf dem die Schaltung implementiert werden soll. Dazu können eigene Makros definiert und, zusammen mit den vorgefertigten, in hierarchischen Strukturen verarbeitet werden.

Auf der anderen Seite ist es möglich, logische Schaltungen mit Hilfe der Hardware-Beschreibungssprache *ABEL-HDL* zu definieren. Damit können auch Zustandsmaschinen verwirklicht werden, die sequentielle Schaltungsabläufe möglich machen. Eine Zustandsmaschine besteht aus einer Reihe definierter Zustände für die bestimmte Signalkombinationen festgelegt werden können. Dabei erfolgt der Wechsel von einem Zustand zum nächsten nur unter bestimmten, benutzerdefinierten Bedingungen. Mit ABEL-HDL definierte Makros können auch in schematische Schaltpläne eingebunden werden.

In Anhang A befindet sich als Beispiel eines schematischen Schaltplans die in einem XC95288 CPLD implementierte VME-Schnittstelle des CATCH1-Moduls. Ein Blocksymbol dieses Schaltplans, die in ABEL-HDL realisierte Zustandsmaschine, die den Datenverkehr auf der CATCH1-Platine regelt, kann in Anhang B gefunden werden.

Ist das Design fertig gestellt, kann die Implementierung weitgehend automatisiert ablaufen. Es ist für den Benutzer jedoch möglich, durch Vorgeben des zeitliche Verhalten von Signalen zu beeinflussen oder den Signalen bestimmte Beinchen des Chips zuzuweisen. Am Ende des Implementierungsprozesses steht dann eine Datei, die von der Software über spezielle Kabel auf den Chip heruntergeladen werden kann. Zusätzlich stehen dem Benutzer ausführliche Informationen zur Verfügung, welche die Suche nach eventuellen Fehlern erleichtern und es erlauben, die Implementierung zunächst in einem Simulator zu testen. Damit kann sichergestellt werden, daß z. B. das zeitliche Verhalten der einzelnen Signale den Anforderungen entspricht, was u. a. wichtig ist, um Kurzschlüsse zu vermeiden, welche die Hardware beschädigen oder gar zerstören könnten.

5.2.3 Die CATCH1-Platine

In diesem Abschnitt soll nun beschrieben werden, wie die Konzepte für das CATCH1-Modul technisch umgesetzt wurden. Der CATCH1 ist als 6U VME Slave-Modul ausgelegt und besteht im wesentlichen aus vier funktionellen Einheiten, deren Aufgaben Inhalt dieses Abschnitts sein werden. Dies sind

1. die Schnittstelle zum VMEbus,
2. die Kontrolleinheit zur Steuerung der Pulser Box,
3. die Einheit zur Kodierung der TCS-Befehle,
4. die Einheit für die Kommunikation mit der Frontend-Karte.

Aus dem Blockdiagramm in Abb. 14 können alle Verbindungen zwischen den einzelnen Komponenten, sowie zu den Steckern auf der Frontplatte der CATCH1-Platine entnommen werden. Dabei kann die Logik der VME-Schnittstelle mit fast allen anderen Einheiten Daten austauschen. Sie ist in der Lage, den FIFO-Puffer anzulesen und kann Daten zu einer seriellen Schnittstelle senden, welche die Initialisierungsdaten zur Frontend-Karte überträgt. Ferner ist eine bidirektionale Kommunikation mit der Kontrolleinheit für die Testpulse vorgesehen. Die Testpulskontrolleinheit schickt TCS-Befehle zur dafür vorgesehenen Kodierungseinheit und steuert über einen 80 poligen Stecker die externe Pulserbox.

In Abb. 15 wird eine Fotografie der fertigen VME-Karte gezeigt. Man kann erkennen, daß die Platine nicht mit sehr vielen integrierten Schaltkreisen bestückt ist. Der Grund dafür ist, daß die beiden funktionellen Einheiten mit der aufwendigsten Logik in jeweils einem programmierbaren Baustein Platz finden. So ist die VME-Schnittstelle in einem XC95288 CPLD implementiert, während für die Testpulskontrolleinheit ein XC4020E FPGA verwendet wird. Da die Kodierung der TCS-Befehle nicht sehr viel Logik erfordert, konnte diese Einheit in einem GAL realisiert werden.

Daneben befinden sich noch der HOTLink-Empfänger und der FIFO-Puffer für die Auslese der Frontend-Daten auf der Platine. Der serielle Sender zur Frontend-Initialisierung konnte neben der VME-Schnittstelle im CPLD implementiert werden. Der restliche Platinenplatz wird von Signalleitungen, Treiberbausteinen für Signale und verschiedenen Taktegebern eingenommen. Zu erwähnen sind noch zwei Drehschalter, mit denen die VME-Adresse des CATCH1-Moduls eingestellt werden kann.

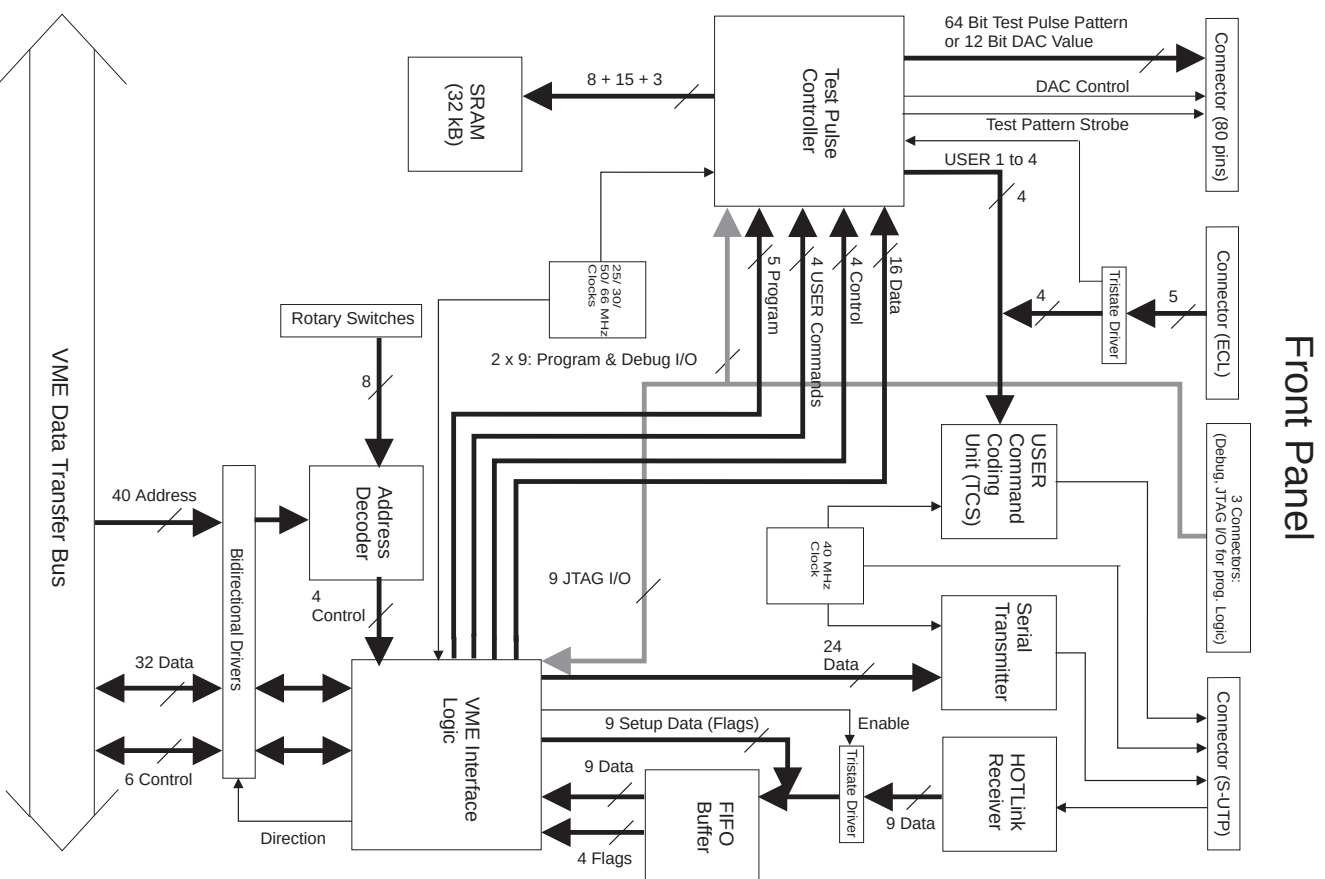


Abbildung 14: Blockdiagramm der CATCH1-Platine. Dargestellt sind alle Datenverbindungen, Stecker und die wichtigsten elektronischen Bauteile. Bei Verbindungen mit mehreren Leitungen ist jeweils ihre Anzahl angegeben. Die Logik für die Schnittstelle zum VMEbus und die Kontrolleinheit für die Testpulse sind in programmierbaren Logikbausteinen implementiert.

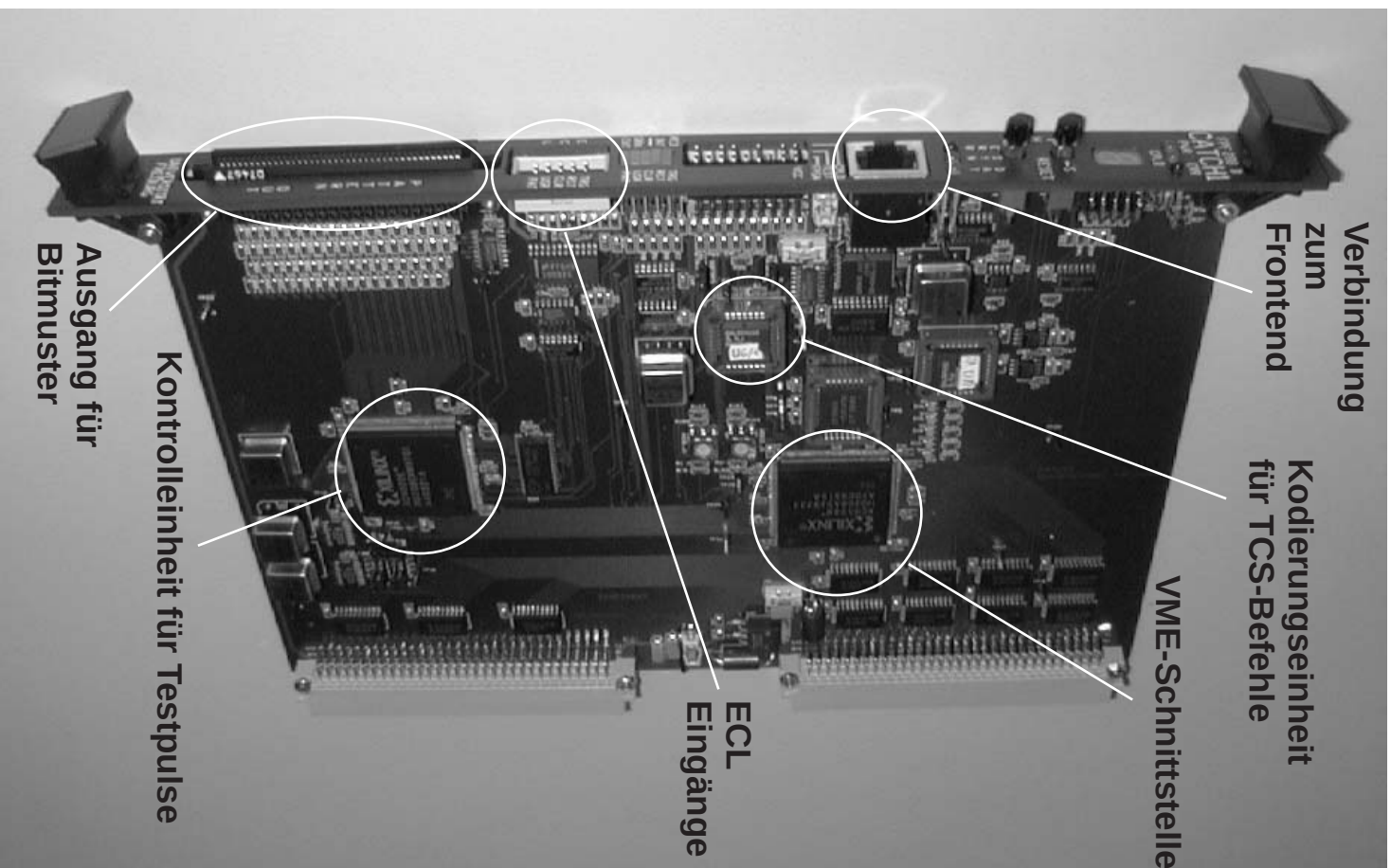


Abbildung 15: Fotografie der fertigen CATCH1-Platine. Die Positionen der wichtigsten funktionellen Einheiten sind hier markiert.

- **Die VME-Schnittstelle**

Die VME-Schnittstelle regelt den Datenaustausch zwischen allen Komponenten auf der CATCH1-Platine und dem VMEbus. Dazu ist es notwendig, daß der Rechner des VME-Systems das CATCH1-Modul für Lese- oder Schreibvorgänge mit der richtigen Adresse anspricht. In der Adresse sind zwei Informationen enthalten: bei Verwendung mehrerer Module muß unterschieden werden, mit welchem Modul der Datenaustausch stattfinden soll und zum zweiten wird festgelegt, welche Daten gelesen oder geschrieben werden sollen. Jeder CATCH1 überwacht also die VME Adreßleitungen und überprüft, ob die nachfolgenden Daten für ihn bestimmt sind oder er selber Daten zur Verfügung stellen muß. Dabei überprüft der Adreßdecoder des CATCH1 den gesamten 32 Bit großen VME Adreßbereich. Im VME Sprachgebrauch redet man davon, daß das Modul auf A32 Adressierungen reagiert (es existieren auch Module, die z. B. für A24 Adressierungen ausgelegt sind).

Um bei der Adressierung mehrere Module voneinander unterscheiden zu können, wird jeder Platine eine Basisadresse zugewiesen. Diese kann auf dem CATCH1 mit Hilfe zweier Drehschalter in einem acht Bit großen Bereich eingestellt werden, was die Adressierung von 256 Modulen erlaubt. Nun können z. B. die unteren acht Adreßbits dazu genutzt werden, verschiedene Befehle für die Ein- und Ausgabe von Daten zu definieren. Diese sogenannte Offsetadresse wird einfach zur jeweiligen Basisadresse hinzuaddiert. Für die CATCH1-Module wurde der Basisadreibereich von E0000000h bis E000FF00h gewählt.

Der Adreßdecoder der VME-Schnittstelle erkennt 13 Befehle, die in Tabelle 3 aufgeführt sind. Nachdem ein Befehl erkannt wurde, übergibt der Adreßdecoder diesen an eine Zustandsmaschine, die dann die auf dem VMEbus ankommenden Daten an die richtige Komponente auf der Platine weiterleitet oder ihrerseits Daten anfordert, um diese an den VMEbus zu übergeben. Bei jedem Datenaustausch mit dem VMEbus werden die maximal möglichen 32 Bit übertragen, was im VME Sprachgebrauch einem D32 Zugriff entspricht. Dies geschieht auch dann, wenn eigentlich nur weniger Bits am anderen Ende erwartet werden.

Der Befehl zur Auslese der Daten von der Frontend-Elektronik bewirkt, daß drei 9 Bit breite Datenworte aus dem FIFO-Puffer gelesen werden. Diese bestehen jeweils aus 8 Bit Daten und einem Bit welches eventuelle Fehler bei der seriellen Übertragung vom Frontend anzeigt. Zusätzlich zu jedem Datenwort liefert der FIFO-Puffer noch ein Bit welches angibt, ob er bei

Tabelle 3: Funktion aller vom CATCH1-Modul erkannten Befehle.

8 Bit Offsetadresse (hexadezimal)	Funktion
00	Auslese der Seriennummer des CATCH1.
08	Auslese der Frontend-Daten aus dem FIFO-Puffer.
10	Statusauslese verschiedener Komponenten auf der Platine.
18	Auslese der Testpulskontrolleinheit.
40	Übergibt Daten zur Initialisierung der Frontend-Elektronik an die serielle Schnittstelle.
48	Steuerung der Testpulskontrolleinheit.
50	Übergibt TCS-Befehle an die Kodierungseinheit.
80	Setzt Register im FIFO-Puffer.
88	Konfiguriert die Testpulskontrolleinheit.
90	Startet einen Selbsttest der HOTLink Verbindung.
98	Setzt den HOTLink in einen anderen Empfangsmodus.
C0	Löscht den FIFO-Puffer.
C8	Setzt die Testpulskontrolleinheit zurück.

der Auslese bereits leer war. Diese insgesamt 30 Bit werden in einem Wort zusammengefaßt und dann an den VMEbus weitergeleitet.

Der FIFO-Puffer verfügt noch über drei weitere Bits, die anzeigen ob er ganz gefüllt ist, oder nur noch eine bestimmte Anzahl von Datenworten enthält bzw. aufnehmen kann. Durch Setzen interner Register kann bestimmt werden, bei welcher Anzahl von Worten die beiden zuletzt erwähnten Bits gesetzt werden. Der Zustand dieser vier, sogenannten FIFO-Flags, kann mit dem Befehl zur Statusauslese abgefragt werden.

Bei der Übertragung der Initialisierungsdaten für die Frontend-Elektronik werden vom VMEbus jeweils 24 Bit an den seriellen Sender transferiert, der sich, wie bereits erwähnt, auch im CPLD-Baustein befindet. Dieser sendet das Datenwort, eingebettet zwischen zwei Start- und zwei Stopbits, zum Frontend. Da der Sender selber keine Daten empfangen kann, geschieht dies ohne

Rückmeldung oder Fehlererkennung. Der Sender arbeitet mit einer Übertragungsrate von 10 MBit pro Sekunde und benötigt somit für die Übertragung der Daten knapp 3 μ s. Daher kann es vorkommen, daß der VMEbus schon die nächsten 24 Bit schicken möchte während der Sender noch beschäftigt ist. Aus diesem Grund beinhaltet die Statusauslese auch Informationen über den aktuellen Zustand des seriellen Senders.

Eine ganz wesentliche Aufgabe der Schnittstelle zum VMEbus ist es, auch die Konfiguration der im FPGA implementierten Testpulskontrolleinheit über den VMEbus zu ermöglichen, da dies ja bei jedem Neustart geschehen muß. Zwar ist auf der Frontplatte des CATCH1-Moduls ein Stecker vorgesehen, über den der Benutzer den Konfigurationsprozeß vom PC aus vornehmen kann, aber das wird im Experiment nicht mehr zweckmäßig sein. Dann werden auf jedem der vielen CATCH-Module mehrere FPGAs zu finden sein, deren Aufgabe es sein wird z. B. den Datenstrom vom Frontend zu ordnen.

Die Konfiguration eines FPGAs geschieht, indem ein serielles Bitmuster mit Hilfe eines Clocksignals über dafür vorgesehene Beinchen auf den Chip geladen wird. Dabei ist es egal, ob das Bitmuster vom PC oder von der VME-Schnittstelle kommt. Der Befehl zur Konfiguration der Testpulskontrolleinheit erfolgt immer zusammen mit der Übermittlung eines Datenwortes. Je nach dem welches der 10 definierten Datenworte auf dem CATCH1 eintrifft, entscheidet die VME-Schnittstelle ob ein Signal am entsprechenden Beinchen des FPGAs gesetzt wird oder nicht. Die folgende Tabelle enthält die Funktionen der einzelnen Datenworte:

8 Bit Datenwort (hexadezimal)	Funktion
01	ermöglicht Konfiguration über den VMEbus
02	ermöglicht Konfiguration über den PC
04	Setzt PROG auf 0 (löscht die FPGA Konfiguration)
08	Setzt PROG auf 1 (Normalzustand)
10	Setzt DIN auf 1 (Dateneingang)
20	Setzt DIN auf 0
40	Setzt CCLK auf 1 (Clocksignal)
80	Setzt CCLK auf 0

Ob der Konfigurationsprozeß erfolgreich abgeschlossen wurde, kann mit dem Befehl zur Statusauslese überprüft werden.

- **Die Testpulskontrolleinheit**

Nach der Konfiguration der Testpulskontrolleinheit kann auf einer 16 Bit breiten Verbindung die Kommunikation mit der VME-Schnittstelle erfolgen. Das dazu verwendete Protokoll erlaubt sowohl Schreib- als auch Lesevorgänge und beinhaltet fünf Kommandos, welche das Verhalten der Testpulskontrolleinheit bestimmen. Da die Kommandos in den drei letzten Bits eines zur Testpulskontrolleinheit übertragenen Datums stecken, muß zunächst immer ein Schreibvorgang erfolgen, welcher den FPGA auf die nachfolgenden Transfers vorbereitet. Neben der Versorgung mit Bitmustern kann mit Hilfe der Kommandos das Verhalten der externen Pulser Box gesteuert werden. Eine Auflistung aller Befehle findet sich in Tabelle 4.

Die Kommandos erlauben es, über die Testpulskontrolleinheit die aktiven Kanäle der Pulser Box, sowie die Amplitude der analogen Signale auszuwählen. Nachdem nun ein Ladungsimpuls am Ausgang der Pulser Box generiert wurde, stehen drei verschiedene Möglichkeiten zur Verfügung, mit denen ein Triggersignal an die Frontend-Elektronik übermittelt werden kann, um die Auslese der Ladungsimpulse auszulösen. Wie bereits erwähnt, können die TCS-Befehle entweder über den VMEbus auf die CATCH1-Platine gelangen oder als ECL-Signale von einer externen Quelle an einen Stecker auf der Frontplatte des Moduls anliegen. In beiden Fällen muß der Benutzer das Zeitintervall zwischen Ladungsimpuls und Triggersignal selbst justieren. Diese Aufgabe kann aber auch von der Testpulskontrolleinheit erfüllt werden. Es ist möglich, die Verzögerung zwischen dem Befehl zur Konversion des Bitmusters und einem automatisch erzeugten Triggersignal in Schritten von 25 ns zu justieren. Hinzu kommt noch ein Offset von ca. 1 μ s, welcher in etwa der im Experiment zu erwartenden Verzögerung entspricht.

Auf der CATCH1-Platine ist ein 32 kByte großes SRAM-Speichermodul implementiert, welches von der Testpulskontrolleinheit angesteuert werden kann, momentan aber nicht genutzt wird. Mit einem veränderten Design der Testpulskontrolleinheit könnten hier z. B. viele Bitmuster zur Selektion der aktiven Kanäle der Pulser Box zwischengespeichert werden, um mit diesen dann in schneller Folge unterschiedliche Ladungsimpulsmuster zu generieren.

Tabelle 4: Gültige Kommandos für die Testpulskontrolleinheit.

Kommando	untere 3 Bit	Funktion
Trigger einrichten	00h	Schreibbefehl, der das Verhalten der Triggersignale einstellt und ihre Quelle auswählt.
Zählerauslese vorbereiten	01h	Schreibbefehl, der den FPGA auf die Auslese der Zähler für TCS-Befehle vorbereitet.
Zähler auslesen		Liest den ausgewählten 16 Bit Zähler aus.
DAC setzen	02h	Schreibbefehl zur Selektion der analogen Signalhöhe in der Pulser Box. Dazu werden 12 Bit in den Digital-Analogwandler geschrieben.
Bitmuster übertragen	03h	Schreibbefehl, der den Transfer des aus 64 Bit bestehenden Testmusters einleitet. Hier nach erwartet die Testpulskontrolleinheit vier Worte à 16 Bit, die die aktiven Kanäle der Pulser Box selektieren.
Bitmuster aktivieren	04h	Schreibbefehl, der die Generation der Ladungsimpulse auslöst.

- **Kodierungseinheit für TCS-Befehle**

Unabhängig von ihrer Herkunft gelangen die TCS-Befehle in diese Einheit und werden für die Übertragung zur Frontend-Elektronik vorbereitet. Dazu werden die vier Signale mit unterschiedlichen Längen versehen. Als Referenz wird ein Clocksignal mit einer Frequenz von 40 MHz verwendet, welches auf der CATCH1-Platine generiert wird (im Experiment wird das Clocksignal zusammen mit den TCS-Befehlen geliefert). Somit erhält ein Triggersignal die Länge einer Clockperiode, also 25 ns, ein Reset-Signal die Länge von zwei Clockperioden etc. Das genaue Zeitverhalten kann aus Abb. 16 ersehen werden. Der Kodierungsprozeß beginnt mit einer steigenden Flanke des Clocksignals und gibt die kodierten Signale nach einer festen Zeit wieder aus. In Abb. 16 werden das Triggersignal mit *USER1* und das Resetsignal zu Beginn eines Spill mit *USER2* bezeichnet. Wie bereits erwähnt kann über die Verwendung der restlichen beiden Signale noch frei entschieden werden.

Sollen keine TCS-Befehle übertragen werden, so kann diese Einheit auch ganz abgeschaltet werden.

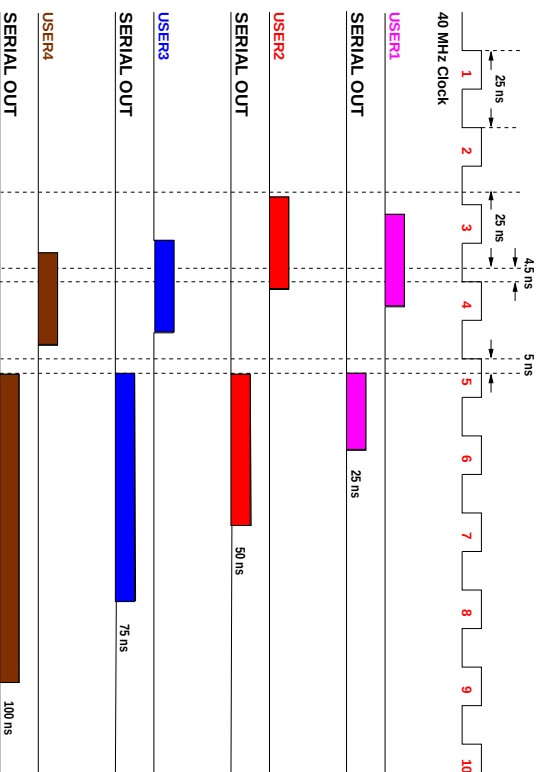


Abbildung 16: Zeitverhalten der vier TCS-Befehle bei ihrer Kodierung. Alle Signale, die mindestens 4,5 ns vor einer steigenden Flanke des Clocksignals eintreffen liegen 5 ns nach der nächsten steigenden Flanke am Ausgang der Einheit an. Bei Signalen, die später als 4,5 ns vor einer steigenden Flanke eintreffen, kann nicht mit Sicherheit gesagt werden, ob sie noch akzeptiert werden oder ihre Kodierung erst mit der nächsten steigenden Clockflanke stattfindet.

Eine ausführlichere, technische Beschreibung aller Komponenten und Signale auf der CATCH1-Platine findet sich in [25].

6 Erste Tests und Ergebnisse

Nachdem das Konzept für die CATCHI-Module feststand, war der nächste Schritt die Produktion der Platinen, wobei ihr Layout in der fakultätseigenen Elektronikwerkstatt vorgenommen wurde. Zu diesem Zeitpunkt waren auch schon die logischen Schaltungen der VME-Schnittstelle und der Testpulskontrollereinheit entwickelt und simuliert. Allerdings bereitete die Version 1.3 der Entwicklungssoftware XACT *step* hier einige Probleme, da sie zunächst keine eindeutige Beinchenzuteilung für die Signale der VME-Schnittstelle im XC95288 CPLD zu zuließ, die für ein Platinenlayout absolut notwendig ist. Die Nachfrage bei der Firma Xilinx ergab, daß es sich hier um einen Fehler in der Software handelte, welcher durch eine Neuinstallation behoben werden konnte.

Die Herstellung von vier Platinen wurde bei einer Freiburger Firma [26] in Auftrag gegeben, wo auch ihre Bestückung mit allen SMD-Bauteilen¹⁷ und deren Verlötlung vorgenommen wurde.

• Inbetriebnahme der Platine

Bei den fertigen CATCHI-Karten stellte sich zunächst die Frage, ob alle integrierten Schaltkreise, insbesondere natürlich die programmierbaren Chips, den Lötvorgang ohne Schaden überstanden hatten. Weiterhin galt es zu überprüfen, ob alle Signalleitungen richtig durchkontaktiert waren und keine Kurzschlüsse die Bauteile auf der Platine beschädigen könnten.

Die Platinen wurden Schritt für Schritt auf ihre Funktionstüchtigkeit überprüft. Da die Schnittstelle zum VMEbus Voraussetzung für alle weiteren Tests war, mußte sie zuerst in Betrieb genommen werden. Die Konfigurationsdaten für den CPLD konnten problemlos über einen speziellen Stecker auf der Frontplatte des Moduls heruntergeladen werden. In Abb. 17 befindet sich dieser Stecker mit der Bezeichnung "CPLD" neben zwei weiteren Steckern für Konfiguration und Test der Testpulskontrollereinheit im FPGA-Chip.

Nachdem so die Funktionstüchtigkeit der CPLD-Bausteine auf allen vier Platinen sichergestellt war, konnten die Module von einem Computersystem aus angesteuert werden. Für diese Zwecke stand ein VME-Übertrahnen mit einem *EUROCOM-17-5xx Dual 68040 CPU Board* der Firma ELTEC zur Verfügung, der eine Adressierung der CATCHI-Platinen mit Hilfe von Routinen in der Programmiersprache C erlaubte. Hier wurde das Betriebssystem OS9 verwendet.

¹⁷SMD, engl.: Surface Mounted Device, sehr kleine Ausführung elektronischer Komponenten wie z. B. Widerstände oder Kondensatoren, die ohne Durchbohrungen direkt auf die Leiterplattenoberfläche gelötet werden.

Die ersten Tests der VME-Schnittstelle bestanden darin, eine Leuchtdiode auf der Frontplatte einzuschalten und die Seriennummer der Platine auszulesen. Dabei wurde das Design der VME-Schnittstelle mit fünf SMD-Leuchtdioden auf der Platine kontrolliert, welche den aktuellen Zustand der Logik im CPLD anzeigten. Mit diesen Tests war es möglich, das Schnittstellendesign zu optimieren, indem das Zeitverhalten der Zustandsmaschine im CPLD etwas variiert wurde.

Der nächste Schritt bestand darin, die Kommunikation zwischen FIFO-Puffer und VME-Schnittstelle zu überprüfen. Mit einem für diese Zwecke modifizierten Design konnten Worte vom VMEbus in den FIFO-Puffer geschrieben und auch wieder ausgelesen werden. Allerdings stellte sich bei diesen Tests heraus, daß ein Widerstand auf der Platine den falschen Wert hatte: erst nach dem Einlöten eines kleineren Widerstandes konnte ein Signal zur Ansteuerung der FIFO-Register schnell genug den Zustand wechseln. In diesen Registern werden die Grenzen für das Setzen der Bits festgelegt, die einen fast vollen oder fast leeren FIFO-Puffer anzeigen.

Im folgenden Schritt wurde der FPGA-Baustein getestet. Dieser wurde zunächst über den Stecker mit der Bezeichnung "FPGA" auf der Frontplatte des CATCH1-Moduls vom PC aus konfiguriert. An dieser Stelle gab es das Problem, daß die Kommunikation zwischen Testpulskontrolleinheit und VME-Schnittstelle nicht sofort hergestellt werden konnte, was daran lag, daß ein Chip bei der Signalübertragung mit normaler, der andere Chip aber mit inverser Logik arbeitete.

Ein weiteres Problem bestand in der Synchronisation der zwischen beiden Chips ausgetauschten Signale. Für den Betrieb der CPLD und FPGA Bausteine stehen auf der CATCH1-Platine Clocksignale mit unterschiedlichen Frequenzen zur Verfügung, welche mit Hilfe von Jumpers für beide Chips separat selektiert werden können. Arbeiten die Chips bei unterschiedlichen Frequenzen, so besteht für die zwischen

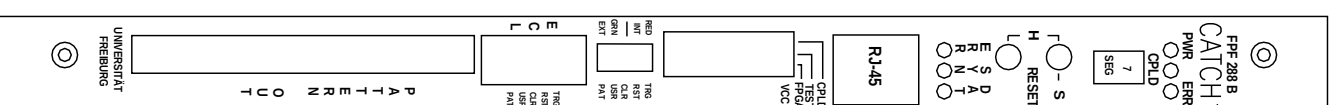


Abbildung 17: Frontplatte des CATCH1.

ihnen ausgetauschten Signale keine feste Beziehung. Das kann bei schlechter Synchronisation der Signale dazu führen, daß die logische Schaltung in einen metastabilen Zustand gerät und Fehlfunktionen auftreten.

Das Ergebnis dieser Tests war, daß die Kommunikation zwischen Testpulskontrolleinheit und VME-Schnittstelle problemlos verläuft, wenn beide Chips bei einer Frequenz von 33 MHz betrieben werden. Dies führt zu der Erkenntnis, daß bei späteren Entwicklungen darauf geachtet werden sollte, möglichst alle Komponenten mit dem gleichen Clocksignal zu versorgen um etwaigen Synchronisationsproblemen vorzubeugen.

Nach erfolgreicher Inbetriebnahme der Testpulskontrolleinheit konnten nun alle zur Verfügung stehenden Befehle eingehend in ihrer Funktion getestet werden, wozu sich eine Siebensegment-Anzeige im oberen Teil der Frontplatte als äußerst hilfreich erwiesen hat. Hier wird ständig der aktuelle Zustand der Einheit angezeigt, also ob sie z. B. weitere Bitmusterdaten erwartet oder Zählerwerte zur Auslese bereitgestellt hat. Unbekannte Befehle führen dabei zu einer Fehleranzeige. Zu diesem Zeitpunkt war es bereits möglich, die Testpulskontrolleinheit auch über den VMEbus zu konfigurieren.

Eine große Hilfe bei der Inbetriebnahme der CATCH1-Module waren auf der Platine vorgesehene Testpins. Mit ihrer Hilfe war es möglich, alle wichtigen Steuersignale mit einem Oszilloskop darzustellen und so ihr Zeitverhalten zu überprüfen.

• Test aller Datenverbindungen

Nachdem die Kommunikation zwischen den funktionellen Einheiten auf den CATCH1-Platinen sichergestellt war, konnte genauer untersucht werden, ob dabei auch die richtigen Daten übertragen werden. Hierzu wurden insbesondere alle Ein- und Ausgangssignale auf den Frontplatten der Module überprüft.

Die Bitmuster und die Steuerbefehle für die Pulser Box wurden vom Computer auf den CATCH1 übertragen und dann direkt am 80 poligen Ausgangsstecker des Moduls mit einem Oszilloskop vermessen, da die Pulser Box zu diesem Zeitpunkt noch nicht verfügbar war. Bei den Messungen wurde überprüft, ob alle 64 Bitmusterkanäle arbeiten und zusammen mit den Steuersignalen die Spezifikationen für die Pulser Box erfüllen. Dabei war es möglich, das Verhalten dieser Signale jederzeit durch Änderungen im Design der Testpulskontrolleinheit zu beeinflussen.

Da auch der Entwicklungsprozeß der Frontend-Platinen noch nicht abgeschlossen ist, wurde für die Übertragung der Daten zwischen Frontend und CATCH1 eine Testplatine verwendet, welche über einen HOTLink Senderchip verfügt und in der Lage ist, die seriellen Initialisierungsdaten, die kodierten TCS-Befehle und das Clocksignal zu empfangen.

Um eine mit dem Experiment vergleichbare Übertragung zu gewährleisten, wurde für diese Tests ein 30 m langes S-UTP Kabel verwendet. So konnten mit diesem Aufbau neben der Funktion des seriellen Senders zur Frontendinitialisierung auch das Clocksignal und die kodierten TCS-Befehle mit Hilfe eines Oszilloskops überprüft werden. Die Tests der kodierten TCS-Befehle wurden sowohl für die externen ECL-Eingänge, als auch für die vom VMEbus übermittelten Befehle durchgeführt. Ferner konnte festgestellt werden, daß die von der Testpulskontrolleinheit automatisch generierten Triggersignale das gewünschte Zeitverhalten besitzen und die, in der Testpulskontrolleinheit enthaltenen Zähler wirklich die richtige Anzahl übertragener TCS-Befehle anzeigen.

Bei dem Test der Datenübertragung von der Testplatine zum CATCHI-Modul lag das Augenmerk insbesondere auf eventuellen Übertragungsfehlern. Das Konzept des COMPASS Datenauslesesystems sieht keine Rückmeldung bei der Übertragung der Detektorinformationen vor, wobei durch Übertragungsfehler verfälschte Daten lediglich als unbrauchbar gekennzeichnet werden. Somit kann ein Übertragungsfehler auf einer einzigen S-UTP Verbindung zum Verlust des ganzen zugehörigen Ereignisses führen. Dabei wird die Wahrscheinlichkeit für durch Übertragungsfehler unbrauchbar gewordene Ereignisse durch die große Zahl der Verbindungen im Experiment dramatisch erhöht.

Für den Test dieser Datenverbindung von der Frontend-Elektronik war die Testplatine zunächst mit einem kleinen FPGGA ausgerüstet, der die Steuerung des HOTLink-Senders übernahm und diesen auch mit bekannten Daten versorgte. Vom CATCHI-Modul wurden die Daten umgehend wieder ausgelesen und mit den Vorgaben verglichen, wobei nun ein schnelleres unter *ALX Unix* laufendes *MVME 2604 Power PC* System verwendet wurde. Zunächst stellte sich heraus, daß die Verbindung zwischen beiden Platinen korrekt arbeitet und die Daten richtig auf dem CATCHI eintreffen.

Allerdings zeigte ein Dauerbetrieb, daß bei der Übertragung sporadisch Fehler auftraten. Die Fehlerrate war mit mehr als einem Fehler pro 10^9 Übertragungen zu groß und damit für das Experiment nicht mehr akzeptabel. Die erste Annahme war, daß die Übertragungsfehler durch ein falsches Zeitverhalten der Steuersignale vom FPGGA-Chip verursacht wurden. Dies konnte aber durch Änderungen der implementierten Logik bzw. Vermessen der Signale auf der Testplatine nicht bestätigt werden. Auch die Verwendung eines anderen Chips zur Generation der Daten und eine Übertragung über ein kürzeres S-UTP Kabel konnten die Fehlerrate nicht verringern.

Es stellte sich heraus, daß das Problem bei der Erdung von Testplatine und CATCHI lag: um im Experiment eine Störung der sehr empfindlichen ASD8b Vorverstärker auf der Frontend-Elektronik durch sogenannte Erd-

schleifen zu vermeiden, sollte zwischen der Frontend-Elektronik und dem CATCH-System keine Verbindung der Massen bestehen.

Somit gab es auch für den Testaufbau keine Massenverbindung zwischen Testplatine und CATCH1. Ferner waren auf der Testplatine die beiden Leitungen zur differentiellen Übertragung der HOTLink-Signale mit Hilfe zweier Kapazitäten von der CATCH1-Platine entkoppelt, wie im linken Diagramm von Abb. 18 zu sehen ist. Bei diesem Aufbau führt eine Spannungsänderung des Bezugspotentials auf der Senderseite zu einer Störung der gesendeten Signale, die auch die Kapazitäten durchdringt und auf der Empfängerseite eine falsche Dekodierung der Daten durch den HOTLink verursacht.

Das Verbinden der Massen beider Platinen durch eine ebenfalls kapazitiv entkoppelte Leitung führte zu keiner entscheidenden Verbesserung, da auch in diesem Fall Übertragungsfehler provoziert werden konnten, indem das Bezugspotential auf Seite der Testplatine schnell um mindestens ein Volt verändert wurde.

Erfolgt dagegen die Entkopplung der Signalleitungen, wie im rechten Diagramm von Abb. 18 gezeigt, mittels eines Transformators auf der Testplatine, so können Änderungen des Bezugspotentials auf der Senderseite nicht mehr durch diesen durchdringen und auf der Empfängerseite Fehler verursachen. Dies wurde für Spannungsänderungen des Bezugspotentials von bis zu 36 V getestet. Mit diesem Aufbau lag die Fehlerrate bei maximal einem einzigen Fehler pro 10^{12} Übertragungen. Da im Experiment sehr starke Magnetfelder auftreten, ist es jedoch fraglich, ob die Transformatoren auf der Frontend-Elektronik eingesetzt werden können. Eine Alternative wäre es, die Entkopplung der differentiellen Signale erst auf den CATCH-Modulen vorzunehmen. Diese Möglichkeit muß allerdings noch genauer untersucht werden. Probleme mit Übertragungsfehlern sollten bei dem Einsatz optischer Übertragungsmedien, wie sie für die Auslese der ringabbildenden Cerenkov Detektoren geplant sind, nicht auftreten.

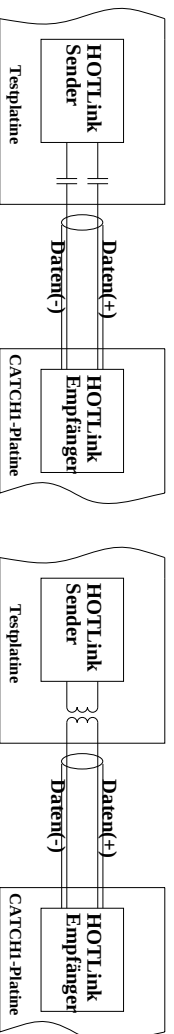


Abbildung 18: Die Entkopplung der differentiellen Signale zwischen HOTLink Sender auf der Testplatine (Frontend-Karte) und dem CATCH1-Modul kann mit Hilfe von Kapazitäten (rechtes Diagramm) oder eines Transformators (linkes Diagramm) erfolgen.

Mit dem CATCHI-Modul steht ein äußerst vielseitiges Instrument zur Verfügung, welches allen sich gestellten Anforderungen gerecht wird. Dieser Eindruck wird noch dadurch verstärkt, daß im programmierbaren FPGABaustein nicht nur die Testpulskontrolleinheit implementiert werden kann, sondern auch andere Designs möglich sind. So kann z. B. ein Testaufbau realisiert werden, der dem in Abb. 10 gezeigten ähnlich ist. Hier wird die Pulser Box von einem CATCHI-Modul gesteuert, während in einem zweiten CATCHI-Modul anstelle der Testpulskontrolleinheit Zähler implementiert sind [27], die über den 80 poligen Stecker auf der Frontplatte des Moduls mit den Ausgangssignalen der zu testenden Frontend-Karte versorgt werden. Mit diesem Aufbau ist es möglich, auch Frontend-Karten zu testen auf denen sich noch kein TDC-Chip befindet, indem die von den ASD8b Vorverstärkern gelieferten Signale gezählt und mit der Signtrate von der Pulser Box verglichen werden. Auf diese Weise können die Diskriminatorschwellen der ASD8b-Chips genau vermessen werden.

An dieser Stelle soll noch die Leistung der CATCHI-Module durch einige, in Tabelle 5 gezeigte Übertragungsraten für Zugriffe über den VMEbus belegt werden. Diese wurden ebenfalls auf dem *MVME 2604 Power PC* System ermittelt, indem die Zeit für eine sehr große Anzahl der jeweiligen Zugriffe gemessen wurde. Dabei ist zu beachten, daß es sich bei allen VME Zugriffen auf die CATCHI-Platine um einfache A32/D32 Lese- bzw. Schreibvorgänge handelt. Obwohl die Übertragungsraten mit anderen, am CERN entwickelten VME-Platinen vergleichbar sind [28], kann die Transferrate noch dramatisch gesteigert werden, indem z. B. für die Auslese des FIFO-Puffers Blocktransferzugriffe in der VME-Schnittstelle implementiert werden.

Tabelle 5: Einige VME Übertragungsraten für das CATCHI-Modul.

VME Zugriff (4 Byte pro Übertragung)	Übertragungsrate
Frontend-Daten aus dem FIFO-Puffer lesen	1,11 MHz
Statusauslese der funktionellen Einheiten	1,20 MHz
Schreiben der TCS-Befehle	2,73 MHz
Generation der Ladungsimpulse auslösen	1,92 MHz
Selektion der aktiven Kanäle und Auslösen der Ladungsimpulsgeneration (5 Zugriffe)	0,33 MHz

- **Ausgangssignale der Pulser Box**

Am Schluß dieses Abschnitts sollen die ersten Messungen der Ausgangssignale einer Pulser Box vorgestellt werden, um einen Eindruck ihrer Leistung zu geben. Für spätere Messungen zum Test von Frontend-Karten sollten die 64 Kanäle der Pulser Box in Bezug auf die von ihnen gelieferten Ladungsimpulse so homogen wie möglich sein. In den hier beschriebenen Messungen wurde die Version der Pulser Box getestet, die mit kapazitiver Ladungsimpulsgeneration arbeitet (siehe Abschnitt 5.1.2). Ihre 64 Kanäle wurden einzeln selektiert um die jeweiligen Ausgangsladungen immer auf den gleichen Kanal eines ASD8b Vorverstärkers zu geben und die Amplituden der verstärkten Signale zu messen. Eine detaillierte Beschreibung der ASD8b Vorverstärker, die auch im COMPASS-Experiment verwendet werden sollen, findet sich in [12]. Für die hier beschriebenen Messungen ist allerdings nur die Kenntnis des prinzipiellen Aufbaus des ASD8b notwendig, der in Abb. 19 dargestellt ist. Hier wird das differentielle Eingangssignal zunächst verstärkt und dann von zwei *Shaper*-Stufen in eine neue Form gebracht. Nachdem die nächste Stufe die Signallänge verkürzt (*Tail Cancellation*), besteht die Möglichkeit, das analoge Signal abzugreifen, bevor es die Diskriminatorstufe erreicht und digitalisiert wird.

Für die Messungen wurden die vom ASD8b verstärkten Ladungsimpulse auf einem digitalen Oszilloskop dargestellt, welches für jeden Meßwert ein über 64 Signale gemitteltetes Ergebnis lieferte. Es wurden vier Meßreihen für die Ladungsimpulse 2, 5, 10 und 20 fC aufgenommen, welche in Abb. 20 dargestellt sind. Obwohl die Werte der einzelnen Meßreihen etwas streuen, gibt es keine großen Ausreißer und es kann eine recht gute Homogenität der Ladungsimpulse festgestellt werden. Allerdings ist zu erkennen, daß alle vier Meßreihen die gleiche Struktur aufweisen: es sind immer die gleichen Kanäle der Pulser Box, die größere Signalamplituden erzeugen. Die Mittelwerte der Meßreihen sind in Tabelle 6 zusammen mit ihrer Standardabweichung aufgeführt.

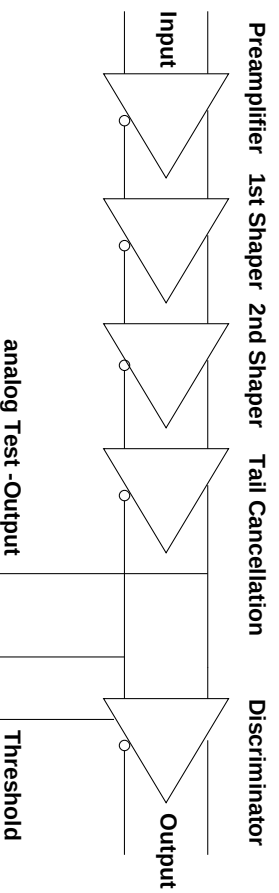


Abbildung 19: Schematische Darstellung der ASD8b Komponenten.

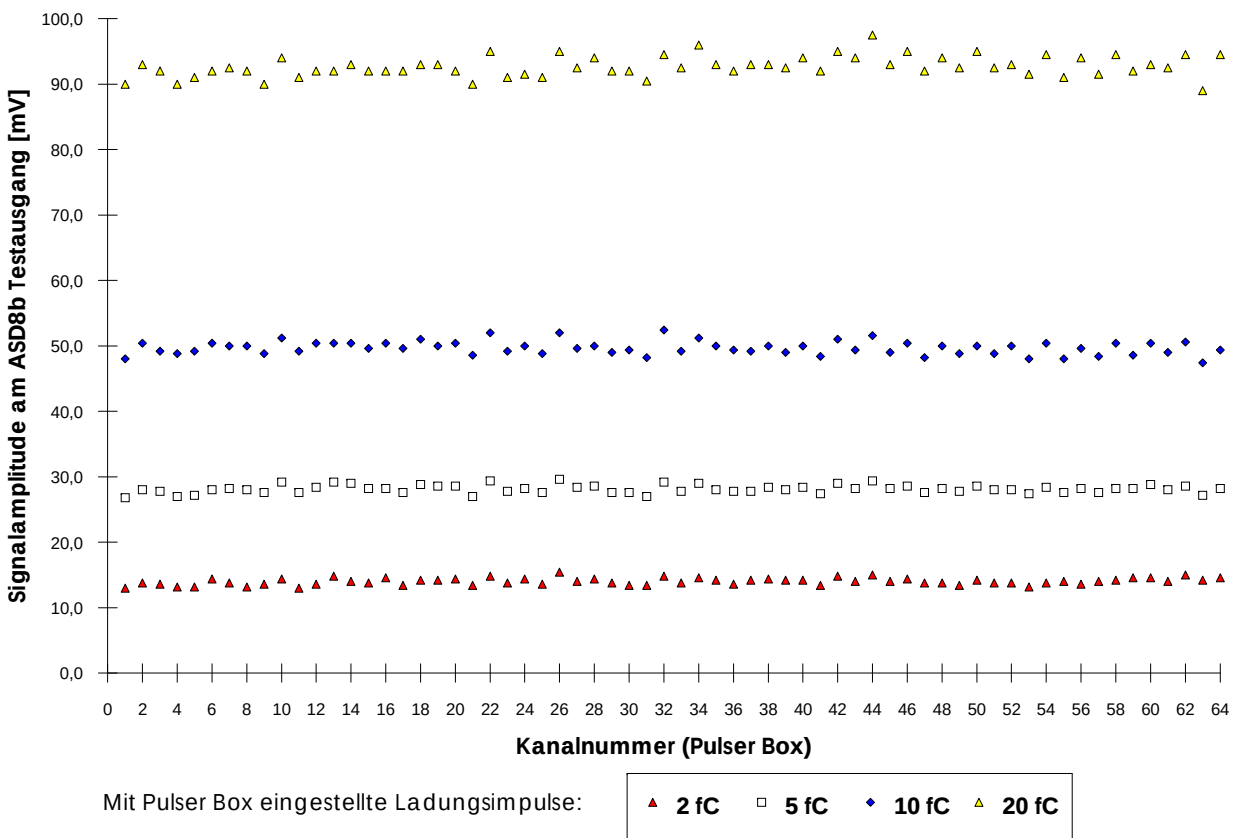


Abbildung 20: Amplituden der analogen Ausgangssignale eines ASD8b Kanals, gemessen für alle 64 Kanäle der Pulser Box bei vier verschiedenen Ladungsimpulseinstellungen.

Tabelle 6: Mittelwerte und Standardabweichungen der Meßreihen mit Ladungsimpulsen von 2, 5, 10 und 20 fC.

Ladungsimpuls	Mittelwert	Standardabweichung
2 fC	14,01 mV	0,53 mV (3,81 %)
5 fC	28,13 mV	0,64 mV (2,26 %)
10 fC	49,69 mV	1,04 mV (2,10 %)
20 fC	92,68 mV	1,60 mV (1,73 %)

Für die Schwankungen der Meßwerte gibt es zwei Gründe. Einmal können die 64 Kanäle voneinander abweichen und zum anderen spielen Meßfehler eine Rolle. Setzt man die Werte aus zwei Meßreihen ins Verhältnis zueinander, so kann man den Einfluß der Meßfehler auf die Schwankungen abschätzen. Kommen die Schwankungen aufgrund voneinander abweichender Kanäle zustande, so sollten die Verhältnisse sehr eng um ihren Mittelwert verteilt sein. Ist dies nicht der Fall, so kann man annehmen, daß die Schwankungen durch Meßfehler dominiert werden. Für das Verhältnis der Meßwerte bei Ladungsimpulsen von 20 fC und 2 fC ergibt sich eine relative Standardabweichung von 3,0 %, während man für das Verhältnis der Meßwerte bei 20 fC und 10 fC eine relative Standardabweichung von 1,4 % erhält. In beiden Fällen sind diese Werte nicht sehr viel kleiner als die relativen Standardabweichungen der Meßreihen bei 2 fC und 10 fC in Tabelle 6. Somit kann man sagen, daß Meßfehler sowohl bei Messungen mit großen, als auch mit kleinen Ladungsimpulsen einen großen Anteil an der Schwankung der Meßwerte haben.

In einem weiteren Test sollte überprüft werden, ob die Messung der Linearität der ASD8b Vorverstärker mit der Pulser Box zu ähnlichen Ergebnissen führt, wie die Einspeisung der Ladungsimpulse mit einer aufwendigen Anordnung im Labor. In Abb. 21 sind die Amplituden der analogen Signale eines ASD8b Kanals in Abhängigkeit von der Eingangsladung dargestellt. Dabei wurde eine Messung mit Ladungsimpulsen von Kanal 15 der Pulser Box durchgeführt, während bei der zweiten Messung die Ladungen im Labor direkt eingespeist wurden. Wie zu erkennen ist, kann mit der Pulser Box die Labormessung mit guter Übereinstimmung wiederholt werden. Die Abweichung der Kurven bei kleinen und mittleren Eingangsladungen sollte durch eine korrekte Kalibration der Ladungsimpulseinstellung in der Pulser Box behoben werden können. Vermutlich liefert die Pulser Box etwas kleinere Ladungen als man erwarten würde. Bei großen Eingangsladungen kommt der ASD8b in die Sättigung, was zur Folge hat, daß die beiden Kurven sich wieder annähern.

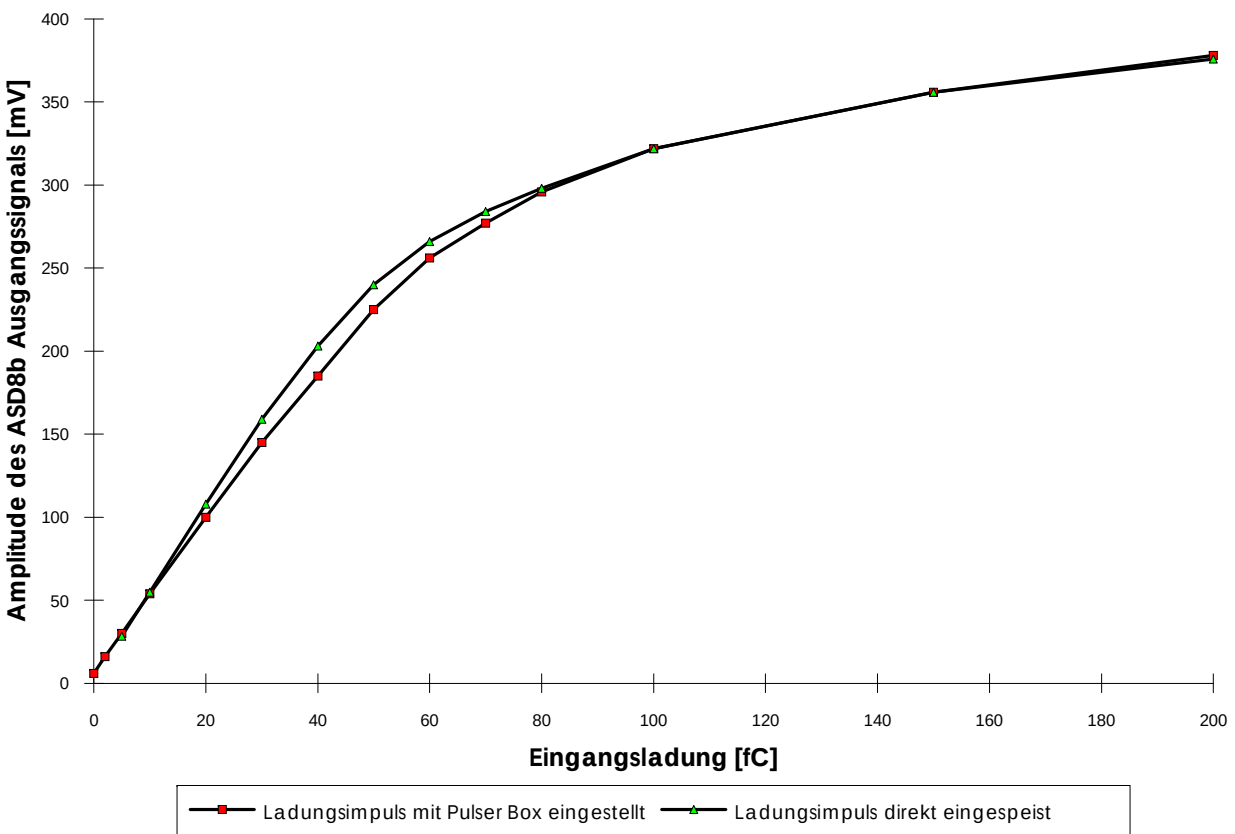


Abbildung 21: Vergleich von zwei Messungen der Linearität des ASD8b Vorverstärkers. Einmal wurden die Ladungen mit einer speziellen Anordnung direkt in den Chip eingespeist, bei der zweiten Messung handelt es sich um Ladungsimpulse, die mit Kanal 15 der Pulser Box generiert wurden.

7 Zusammenfassung

Im Rahmen dieser Diplomarbeit wurden in einer Einarbeitungsphase zunächst Messungen an einem Testmodul von Straw-Driftrohrchen durchgeführt, um die Eigenschaften des Detektors und der Elektronik zu seiner Auslese kennenzulernen. Dies geschah insbesondere im Hinblick auf das Zusammenwirken von Detektor und dem ASD8b Vorverstärker. Desweiteren wurden die Straw-Driftrohrchen mit dem Programm *Garfield* am Computer simuliert, um z. B. geeignete Gasmischungen für ihren Betrieb zu finden.

Im folgenden lag der Schwerpunkt der Arbeit bei der Implementierung der VME-Schnittstelle des CATCH1-Moduls. Da zur Entwicklung dieser Platine an der Fakultät erstmals programmierbare logische Bausteine wie CPLDs und FPGAs zum Einsatz kamen, war es notwendig, sich mit ihrer Architektur, sowie mit der Software zur Entwicklung logischer Schaltungen der Firma Xilinx vertraut zu machen. Ferner war die Einarbeitung in die Funktionsweise der VMEbus-Architektur notwendig.

Auf der Basis der VME-Schnittstelle wurde bei der Realisierung der Kommunikation zwischen den funktionellen Einheiten auf der CATCH1-Platine mitgewirkt. Nach der Fertigstellung des CATCH1-Moduls wurde dieses auf seine vollständige Funktionstüchtigkeit überprüft. Für diese Zwecke war der Einsatz elektronischer Testaufbauten notwendig. Weiterhin mußten Programme in der Programmiersprache C entwickelt werden, um das VME-Modul anzusteuern.

Zusammenfassend kann man sagen, daß die CATCH1-Platine allen Erwartungen gerecht wird. Obwohl die Frontend-Elektronik aufgrund des noch nicht verfügbaren $\mathcal{F}1$ TDC-Chips bislang nicht in der Form vorliegt, die für das Experiment vorgesehen ist, steht mit dem CATCH1 bereits eine ausgezeichnete Testmöglichkeit für Frontend-Karten zur Verfügung. Somit wird es möglich sein, alle Frontend-Platinen vor ihrem Einsatz im Experiment zeitsparend auf ihre Funktionalität zu überprüfen und z.B. geeignete Schwellenspannungen für die Diskriminatorstufen der ASD8b Vorverstärker auszumessen. Ferner ist es mit Hilfe des CATCH1-Moduls möglich, wichtige Teile des Konzeptes für die Datenauslese von COMPASS, insbesondere die Datenübertragung von den Frontend-Karten, zu überprüfen. Die bislang durchgeführten Tests haben gezeigt, daß das Konzept des CATCH-Systems realisierbar ist und eventuell auftretende Probleme überwindbar sind.

Aufbauend auf den bei der Entwicklung der CATCH1-Platinen gewonnenen Erkenntnissen kann nun die nächste Generation von CATCH-Modulen konzipiert werden, die dann alle für das Experiment wesentlichen Funktionen aufweisen wird.

Literatur

- [1] Letter of Intent: *CHarm Experiment with Omni-Purpose Setup*, CERN/SPSLC 95-27 (1995)
- [2] Letter of Intent: *Semi-inclusive Muon Scattering from a Polarized Target*, CERN/SPSLC 95-27 (1995)
- [3] COMPASS Collaboration: *A proposal for a Common Muon and Proton Apparatus for Structure and Spectroscopy*, CERN/SPSLC 96-14 (1996).
- [4] A.D. Watson, *Z. Phys. C* **12** (1982) 123.
- [5] J. Ellis, D. Kharzeev und A. Kotzinian, *Z. Phys. C* **69** (1996) 467.
- [6] J. Kyniäinen, in H. Dutz und W. Meyer, eds., *Proceedings 7th Workshop on Polarized Target Materials and Techniques*, Bad Honnef, Deutschland, 3. bis 6 Juni 1994, Nucl. Instrum. Methods A **356** (1995) 47.
- [7] A. Abragam und M. Goldman, Rep. Prog. Phys. **41** (1978) 395.
- [8] R. Veenhof, *GARFIELD: A Drift-Chamber Simulation Program*, CEA, DAPNIA/SPP, CE Saclay, 91191 Gif-sur-Yvette, Frankreich (1996).
- [9] F. Binon et al., Nucl. Instrum. Methods A **256** (1987) 444.
- [10] F. Binon et al., Preprint SPSC/84-45/p214 (CERN, Genf, 1985).
- [11] E. Iarocci, Nucl. Instrum. Methods **217** (1983) 30.
- [12] F. M. Newcomer, *A Fast, Low Power, Amplifier-Shaper-Discriminator for High Rate Straw Tracking Systems*, IEEE Transactions on Nuclear Science, Vol. 40, No. 4 (1993) 630.
- [13] G. Braun et al., *TDC Chip and Readout Driver Developments for COMPASS and LHC-Experiments, fourth workshop on electronics for LHC experiments - LEB98*, Rom, Italien, September 1998.
- [14] ISO/IEC 11801:1995 *Information technology - Generic Cabling for Customer Premises*.
- [15] IEEE P1386, *Standard for a Common Mezzanine Card Family: CMC*.

- [16] Cypress Semiconductor, *CY7B933/CY7B933 HOTLink preliminary Data Sheet*, Cypress Semiconductor High Performance Data Book, August 1, 1993.
- [17] E. van der Bij et al., *S-LINK: a Prototype of the ATLAS Readout-link, fourth workshop for LHC experiments - LEB98*, Rom, Italien, September 1998.
- [18] ANSI X3T9.3/91-005, High Performance Parallel Interface Standard.
- [19] Analog Devices, *AD7845 D/A Converter* Data Sheet
- [20] W. D. Peterson, *The VMEbus Handbook*, VFEA International Trade Association, 1992
- [21] IEEE P1014, *The VMEbus Specification*
- [22] CYPRESS Semiconductor Corporation, *Data Book* (1994)
- [23] Xilinx, *The Programmable Logic Data Book* (1996) 3-3 ff.
- [24] Xilinx, *The Programmable Logic Data Book* (1996) 4-1 ff.
- [25] G. Braun et al., *CATCH, A Test-facility for COMPASS Front-End Electronics - User Manual*, COMPASS Note 1998-16.
- [26] Creners CAD-Technik, Burkheimer Str. 3, 79111 Freiburg i. Br.
- [27] T. Schmidt, *private Mitteilung*.
- [28] M. Joos et al., *An Evaluation of Second Generation VMEbus PowerPC Based Single Board Computers Running the LynxOS Operating System*, CERN Preprint (1998).

A Schaltplan der VME-Schnittstelle

In Abb. 22 ist der Schaltplan der VME-Schnittstelle des CATCH1-Moduls zu sehen, welche in einem XC95288 CPLD der Firma Xilinx implementiert ist. Die Blocksymbole können selbst definiert und durch Leitungen miteinander oder mit den Beinchen des Chips verbunden werden. Weiterhin stellt die Entwicklungssoftware große Bibliotheken mit vordefinierten Symbolen zur Verfügung.

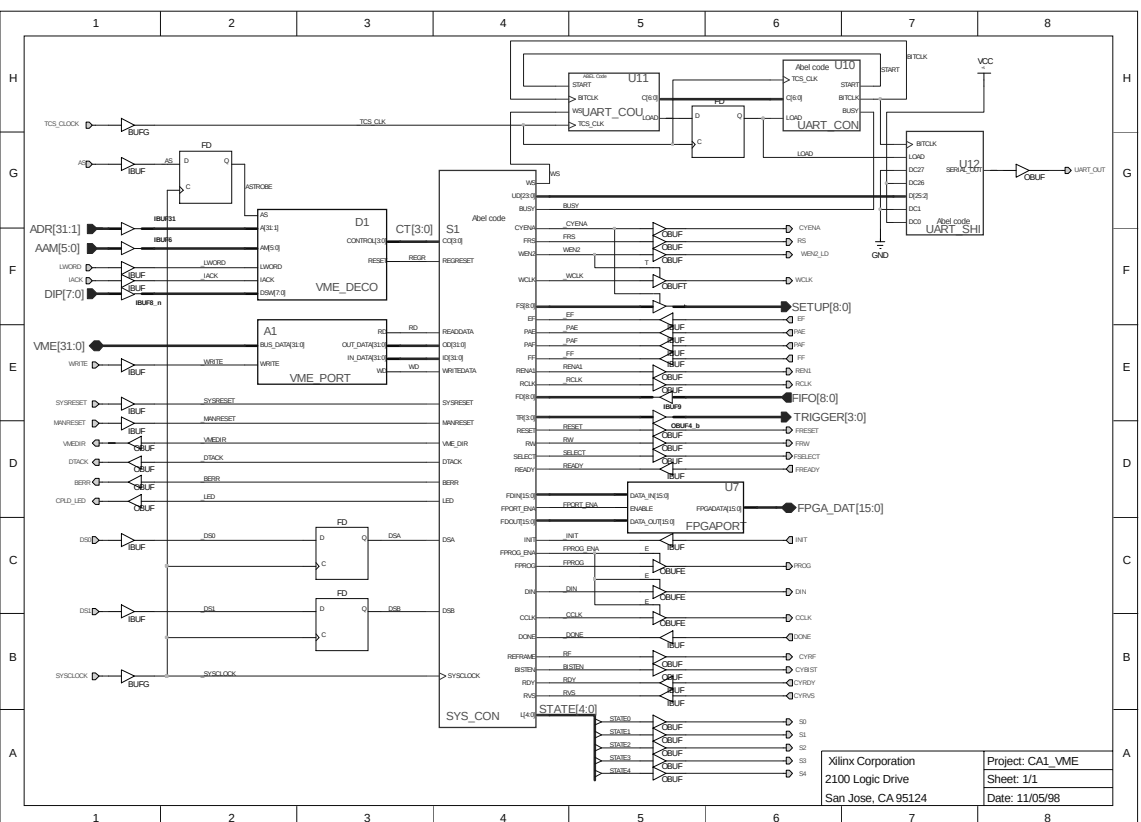


Abbildung 22: Schematischer Schaltplan der VME-Schnittstelle.

B Beispiel für einen ABEL-HDL Quellcode

Der folgende ABEL-HDL Quellcode zeigt die Zustandsmaschine in der VME-Schnittstelle, die den Datenverkehr auf der CATCHI-Platine regelt. In dieser Hardwarebeschreibungssprache werden zunächst die Signalein- und Ausgänge festgelegt. Im nächsten Abschnitt können dann mehrere Signale zu Bussen zusammengefaßt und Konstanten deklariert werden, bevor die eigentlichen Gleichungen für die logischen Verknüpfungen folgen. Das Zustandsdiagramm im hinteren Teil der Routine legt das Verhalten einzelner Signale und Bedingungen für den Wechsel zwischen Zuständen fest.

```
Module SYS_CON
Title 'VMEbus Interface_Controller as a State_Machine'
Declarations
  "PIN Assingment

  "INPUTS
    SYSCLOCK      PIN;
    SYSRESET      PIN;
    MANRESET      PIN;

    "from VMEbus
    DSA,DSB       PIN;

    "from FPGA
    INIT          PIN;
    DONE          PIN;
    READY         PIN;
    READY_LATCH   NODE istype 'reg_d,buffer';

    "from UART
    BUSY          PIN;

    "from FIFO
    FD8..FD0      PIN;
    EF_FF         PIN;
    ELATCH        NODE istype 'reg_d,buffer';
    PAE,PAF       PIN;

    "from HOTLINK
    RDY,RVS       PIN;

    "from ADD_DECO
    CO3..CO0      PIN;

    "from VME_PORT
    ID31..ID0     PIN;

  "OUTPUTS

  "to VMEbus
    RGRRESET      PIN;
    !VME_DIR      PIN;
    !DTACK        PIN;
    BERR          PIN;
    LED           PIN;
    OD31..OD0     PIN;
    READDATA      PIN;
    WRITEDATA     PIN;

    "to FPGA
    PPRG_ENA      PIN istype 'reg_d,buffer';
    PPRG          PIN istype 'reg_d,buffer';
    DIN           PIN istype 'reg_d,buffer';
    CCLK          PIN istype 'reg_d,buffer';
    TR3..TR0      PIN;
    RW            PIN;
    SELECT        PIN;
    !RESET        PIN;
    FPORT_ENA     PIN;

    "to UART
    UD23..UD0     PIN;
    WS            PIN;

    "to FIFO
    FS8..FS0      PIN;
```

```

        !FRS          PIN;
        CYENA         PIN istance 'reg_d,buffer';
        WEN2          PIN istance 'reg_d,buffer';
        WCLK          PIN;
        RCLK          PIN;
        !RENA1        PIN;

"to HOTLINK
        BISTEN        PIN istance 'reg_d,buffer';
        REFRAME       PIN istance 'reg_d,buffer';

"to VME_PORT
        FDIN15..FDIN0 PIN;
        FDOUT15..FDOUT0 PIN;

"STATE MACHINE
        TT4..TT0      NODE istance 'reg_d,buffer';
        L4..L0        PIN;

"additional registers...
        FW9..FW0      NODE istance 'reg_d,buffer';
        FX9..FX0      NODE istance 'reg_d,buffer';

"SETS

"BUSES
OUTDATA      = [OD31..OD0];
INDATA       = [ID31..ID0];
TRIGGER      = [TR3..TR0];
FPGA_DATA_OUT = [FDOUT15..FDOUT0];
FPGA_DATA_IN  = [FDIN15..FDIN0];
UART_DATA     = [UD23..UD0];
FIFO_SETUP    = [FS8..FS0];
FIFO_DATA     = [FD8..FD0];
CONTROL       = [CO3..CO0];
BOARD_STATUS  = [INIT,DONE,EF,PAE,PAF,FF,RVS,RDY,BUSY];
STATE_LEDS    = [L4..L0];
WORDA         = [FW9..FW0];
WORDB         = [FX9..FX0];

"Constants
        READ_ID       = ^b0001;
        READ_FIFO     = ^b0010;
        READ_STATUS   = ^b0011;
        READ_FPGA     = ^b0100;
        WRITE_UART     = ^b0110;
        WRITE_FPGA     = ^b0111;
        WRITE_TRIG     = ^b1000;
        PROG_FIFO      = ^b1001;
        PROG_FPGA      = ^b1010;
        PROG_LED       = ^b1011;
        PROG_RF        = ^b1100;
        PROG_BIST      = ^b1101;
        RESET_FIFO     = ^b1110;
        RESET_FPGA     = ^b1111;

"Bitpatterns for FPGA Programming
        SET_ENA        = ^h01;
        RESET_ENA      = ^h02;
        SET_PROG       = ^h04;
        RESET_PROG     = ^h08;

```

```

        SET_DIN        = ^h10;
        RESET_DIN      = ^h20;
        SET_CCLK       = ^h40;
        RESET_CCLK     = ^h80;

"Bitpatterns for LED
        LED_ON         = ^h01;
        LED_OFF        = ^h02;

"Bitpatterns for REFRAMING
        RF_ON          = ^h01;
        RF_OFF         = ^h02;

"Bitpatterns for BISTEN
        TEST_ON        = ^h01;
        TEST_OFF       = ^h02;

"States
CON_STATE      = [ TT0, TT1, TT2, TT3, TT4 ];
ZERO           = [ 0, 0, 0, 0, 0 ]; ^d00

        R_ID1         = [ 1, 0, 1, 0, 0 ]; ^d20
        R_ID2         = [ 0, 1, 1, 0, 0 ]; ^d12
        R_FIFO1       = [ 0, 0, 1, 0, 1 ]; ^d05
        R_FIFO2       = [ 0, 0, 1, 1, 0 ]; ^d06
        R_FIFO3       = [ 1, 0, 1, 0, 1 ]; ^d21
        R_FIFO4       = [ 0, 1, 0, 1, 1 ]; ^d11
        STATUS1       = [ 0, 1, 1, 1, 0 ]; ^d14
        STATUS2       = [ 1, 1, 0, 1, 1 ]; ^d27
        R_FPGA1       = [ 0, 1, 0, 0, 0 ]; ^d08
        R_FPGA2       = [ 1, 0, 0, 0, 0 ]; ^d16

        UART1         = [ 0, 0, 0, 0, 1 ]; ^d01
        UART2         = [ 0, 0, 0, 1, 0 ]; ^d02
        UART3         = [ 0, 0, 1, 0, 0 ]; ^d04
        W_FPGA1       = [ 0, 0, 0, 1, 1 ]; ^d03
        W_FPGA2       = [ 1, 1, 0, 0, 0 ]; ^d24
        TRIG1         = [ 1, 1, 1, 1, 1 ]; ^d31

        P_FIFO1       = [ 1, 0, 0, 0, 1 ]; ^d17
        P_FIFO2       = [ 1, 1, 0, 0, 1 ]; ^d25
        P_FIFO3       = [ 1, 0, 0, 1, 1 ]; ^d19
        P_FIFO4       = [ 0, 0, 1, 1, 1 ]; ^d07
        P_FIFO5       = [ 1, 1, 1, 0, 1 ]; ^d29
        P_FIFO6       = [ 1, 0, 1, 1, 0 ]; ^d22

        P_FPGA1       = [ 1, 0, 1, 1, 1 ]; ^d23
        RF1           = [ 1, 1, 0, 1, 0 ]; ^d26
        SELF_TEST     = [ 1, 0, 0, 1, 0 ]; ^d18

        RES_FIFO      = [ 0, 1, 0, 0, 1 ]; ^d09
        RES_FPGA      = [ 0, 1, 1, 0, 1 ]; ^d13

"        DATA_ACK    = [ 1, 1, 1, 0, 0 ]; ^d28
"        X           = [ 0, 1, 0, 1, 0 ]; ^d10
"        Y           = [ 0, 1, 1, 1, 1 ]; ^d15
"        Z           = [ 1, 1, 1, 1, 0 ]; ^d30

```

Equations

```

"Resets
REGRESET      = (!SYSRESET # !MANRESET);
CON_STATE.AR  = (!SYSRESET # !MANRESET);
FRS           = (!SYSRESET # !MANRESET);

"Sets
STATE_LEDS    = [EF, PAE, PAF, FF, RDY];
LED           = DSA # !DTACK;
READY_LATCH.D = 1;
BERR         = 1;
FPROG_ENA.D   = 1;
FPROG.D       = 1;
DIN.D         = 1;
CCLK.D        = 1;
CYENA.D       = 1;
WEN2.D        = 1;
WORDA[8..0].D = FIFO_DATA;
WORDB[8..0].D = FIFO_DATA;

WORDA[9].D    = EF;
WORDB[9].D    = EF;
ELATCH.D      = EF;

"Clocks
CON_STATE.CLK = !SYSCLOCK;
RCLK         = SYSCLOCK;
WCLK         = SYSCLOCK;
CYENA.CLK    = (CONTROL == PROG_FIFO);
WEN2.RE      = (CONTROL == PROG_FIFO);

State_Diagram CON_STATE

State ZERO:

    "Reading Data

    If (CONTROL == READ_ID)      Then R_ID1
    With {VME_DIR = 1; READDATA = 1;}

    Else If (CONTROL == READ_FIFO) Then R_FIFO1
    With {RENAL = 1; WORDA[9].CLK = RCLK;}

    Else If (CONTROL == READ_STATUS) Then STATUS1
    With {VME_DIR = 1; READDATA = 1; OUTDATA[15..7] = BOARD_STATUS;}

    Else If (CONTROL == READ_FPGA) Then R_FPGA1
    With {VME_DIR = 1; READDATA = 1;}

    "Writing Data

    Else If (CONTROL == WRITE_FPGA) Then W_FPGA1
    With {WRITEDATA = 1; RW = 1; FPGA_DATA_IN = INDATA[31..16];}

    Else If (CONTROL == WRITE_UART) Then UART1
    With {WRITEDATA = 1;}

    Else If (CONTROL == WRITE_TRIG) Then TRIG1
    With {WRITEDATA = 1;}

```

```

"Programming

Else If (CONTROL == PROG_FIFO) Then P_FIFO1
With {WRITEDATA = 1; "FRS = 1;"}

Else If (CONTROL == PROG_FPGA) Then P_FPGA1
With {WRITEDATA = 1;}

" Else If (CONTROL == PROG_LED) Then LED1
" With {WRITEDATA = 1;}

Else If (CONTROL == PROG_RF) Then RF1
With {WRITEDATA = 1; REGRESET = 1;}

Else If (CONTROL == PROG_BIST) Then SELF_TEST
With {WRITEDATA = 1; REGRESET = 1;}

"Resets

Else If (CONTROL == RESET_FIFO) Then RES_FIFO
With {REGRESET = 1;}

Else If (CONTROL == RESET_FPGA) Then RES_FPGA
With {RESET = 1; REGRESET = 1;}

Else ZERO;

*****
***** READ_ID *****
*****

State R_ID1:
    REGRESET      = 1;
    VME_DIR       = 1;
    READDATA      = 1;
    OD31          = 1;
    OD30          = 1;
    OD27          = 1;
    OD25          = 1;
    OD20          = 1;
    OD2           = 1;
    OD1           = 1;
    OD0           = 1;
    DTACK         = 1;
    If (!DSA # !DSB) Then R_ID1;
    Else R_ID2;
State R_ID2:
    VME_DIR       = 1;
    OD31          = 1;
    OD30          = 1;
    OD27          = 1;
    OD25          = 1;
    OD20          = 1;
    OD2           = 1;
    OD1           = 1;
    OD0           = 1;
    DTACK         = 1;
    Goto ZERO;

```

```

*****
*****      READ_FIFO      *****
*****

State   R_FIFO1:  RENAL      = 1;
                REGRESET    = 1;
                WORDA[8..0].CLK = RCLK;
                Goto R_FIFO2 With WORDB[9].CLK = RCLK;

State   R_FIFO2:  RENAL      = 1;
                WORDB[8..0].CLK = RCLK;
                Goto R_FIFO3 With {VME_DIR = 1;
                READDATA = 1; ELATCH.CLK = RCLK;}

State   R_FIFO3:  VME_DIR     = 1;
                OUTDATA[7..0] = FIFO_DATA[7..0];
                OUTDATA[15..8] = WORDB[7..0];
                OUTDATA[23..16] = WORDA[7..0];
                OUTDATA[26..24] = [FW8,FX8,FD8];
                OUTDATA[30..28] = [FW9,FX9,ELATCH];
                DTACK         = 1;
                If (!DSA # !DSB) Then R_FIFO3;
                Else R_FIFO4;

State   R_FIFO4:  Goto ZERO;

*****
*****      READ_STATUS     *****
*****

State   STATUS1:  REGRESET    = 1;
                VME_DIR      = 1;
                READDATA     = 1;
                OUTDATA[15..7] = BOARD_STATUS;
                DTACK         = 1;
                If (!DSA # !DSB) Then STATUS1;
                Else STATUS2;

State   STATUS2:  VME_DIR     = 1;
                OUTDATA[15..7] = BOARD_STATUS;
                DTACK         = 1;
                Goto ZERO;

*****
*****      READ_FPGA      *****
*****

State   R_FPGA1:  REGRESET    = 1;
                VME_DIR      = 1;
                READDATA     = 1;
                READY_LATCH.CLK = (READY & !SYSCLOCK);
                SELECT        = 1;
                If (!READY_LATCH) Then R_FPGA1;
                Else R_FPGA2 With
                OUTDATA[31..16] = FPGA_DATA_OUT;

State   R_FPGA2:  VME_DIR     = 1;
                READY_LATCH.RE = 1;
                OUTDATA[31..16] = FPGA_DATA_OUT;
                DTACK         = 1;
                If (!DSA # !DSB) Then R_FPGA2;
                Else ZERO;

```

```

*****
*****      WRITE_FPGA     *****
*****

State   W_FPGA1:  REGRESET    = 1;
                FPORT_ENA    = 1;
                READY_LATCH.CLK = (READY & !SYSCLOCK);
                RW            = 1;
                SELECT        = 1;
                FPGA_DATA_IN  = INDATA[31..16];
                If (!READY_LATCH) Then W_FPGA1;
                Else W_FPGA2;

State   W_FPGA2:  READY_LATCH.RE = 1;
                DTACK         = 1;
                If (!DSA # !DSB) Then W_FPGA2;
                Else ZERO;

*****
*****      WRITE_UART     *****
*****

State   UART1:    REGRESET    = 1;
                UART_DATA    = INDATA[0..23];
                If (BUSY) Then UART1;
                Else UART2 With WS = 1;

State   UART2:    WS          = 1;
                UART_DATA     = INDATA[0..23];
                Goto UART3;

State   UART3:    UART_DATA    = INDATA[0..23];
                DTACK         = 1;
                If (!DSA # !DSB) Then UART3;
                Else ZERO;

*****
*****      WRITE_TRIG     *****
*****

State   TRIG1:    REGRESET    = 1;
                TRIGGER       = INDATA[3..0];
                DTACK         = 1;
                If (!DSA # !DSB) Then TRIG1;
                Else ZERO;

*****
*****      PROG_FIFO      *****
*****

State   P_FIFO1:  REGRESET    = 1;
                FRS           = 1;
                Goto P_FIFO2 With WEN2.RE = 1;

State   P_FIFO2:  FIFO_SETUP[7..0] = INDATA[31..24];
                FIFO_SETUP[8] = 0;
                Goto P_FIFO3;

State   P_FIFO3:  FIFO_SETUP[7..0] = INDATA[23..16];
                FIFO_SETUP[8] = 0;
                Goto P_FIFO4;

State   P_FIFO4:  FIFO_SETUP[7..0] = INDATA[15..8];
                FIFO_SETUP[8] = 0;
                Goto P_FIFO5;

```

```

State    P_FIFO5:  FIFO_SETUP[7..0] = INDATA[7..0];
                FIFO_SETUP[8] = 0;
                Goto P_FIFO6;
State    P_FIFO6:  CYENA.RE      = 1;
                WEN2.CLK       = SYSCLOCK;
                DTACK          = 1;
                If (!DSA # !DSB) Then P_FIFO6;
                Else ZERO;

"*****
"*****          PROG_FPGA          *****
"*****

State    P_FPGA1:  REGRESET      = 1;

                FPROG_ENA.RE    = (INDATA[7..0] == RESET_ENA);
                FPROG_ENA.CLK   = (INDATA[7..0] == SET_ENA);
                FPROG.RE        = (INDATA[7..0] == SET_PROG);
                FPROG.CLK       = (INDATA[7..0] == RESET_PROG);
                DIN.RE          = (INDATA[7..0] == RESET_DIN);
                DIN.CLK         = (INDATA[7..0] == SET_DIN);
                CCLK.RE         = (INDATA[7..0] == RESET_CCLK);
                CCLK.CLK        = (INDATA[7..0] == SET_CCLK);

                DTACK          = 1;
                If (!DSA # !DSB) Then P_FPGA1;
                Else ZERO;

"*****
"*****          SWITCHING THE LED          *****
"*****

"State    LED1:    REGRESET      = 1;
"                LED.CLK        = (INDATA[7..0] == LED_ON);
"                LED.RE         = (INDATA[7..0] == LED_OFF);
"                DTACK          = 1;
"                If (!DSA # !DSB) Then LED1;
"                Else ZERO;

"*****
"*****          SWITCHING REFRAME ON/OFF          *****
"*****

State    RF1:      REFRAME.D      = 1;
                REFRAME.CLK       = (INDATA[7..0] == RF_OFF);
                REFRAME.RE        = (INDATA[7..0] == RF_ON);
                DTACK             = 1;
                If (!DSA # !DSB) Then RF1;
                Else ZERO;

"*****
"*****          TOGGLE BUILD IN SELF TEST OF HOT LINK          *****
"*****

State    SELF_TEST:BISTEN.D      = 1;
                BISTEN.CLK        = (INDATA[7..0] ==
TEST_OFF);
                BISTEN.RE = (INDATA[7..0] == TEST_ON);

                DTACK             = 1;
                If (!DSA # !DSB) Then SELF_TEST;

```

```

                Else ZERO;

"*****
"*****          RESETTING THE FIFO          *****
"*****

State    RES_FIFO:  REGRESET      = 1;
                FRS          = 1;
                Goto SELF_TEST;

"*****
"*****          RESETTING THE FPGA          *****
"*****

State    RES_FPGA:  REGRESET      = 1;
                RESET          = 1;
                Goto SELF_TEST;

"*****
"*****          DTACK for RESETS          *****
"*****

"State    DATA_ACK: goto ZERO with REGRESET = 1;

"*****
"*****          EMPTY STATES          *****
"*****

"State    X:        goto ZERO with REGRESET = 1;
"State    Y:        goto ZERO with REGRESET = 1;
"State    Z:        goto ZERO with REGRESET = 1;

End SYS_CON

```