Ein Slow-Control-System für das CHORUS-Experiment

Diplomarbeit
von
Dirk Bonekämper

Fachbereich Physik der
Westfälische Wilhelms Universität Münster
Institut für Kernphysik
August 1993

Inhaltsverzeichnis

1. Einleitung

CHORUS (Cern Hybrid Oscillation Research apparatUS) ist ein mittelgroßes Experiment am CERN. Es hat zum Ziel, Neutrinooszillationen von ν_μ nach ν_τ direkt nachzuweisen. Dazu wird ein Hybriddetektor verwendet, der aus einem Emulsionstarget und einer anschließenden elektronischen Tracker/Kalorimeterkombination besteht. Neben der physikalischen Datenerfassung gehört zum Experiment auch eine Überwachung sämtlicher anderer elektronischen Parameter, die nicht zu den physikalischen Daten beitragen. Systeme, die diese Aufgabe erfüllen, werden im allgemeinen Slow Control Systeme genannt. Dieser Name kommt daher, daß ein Slow Control System nur Parameter mißt, die sich zeitlich langsam verändern. Langsam heißt in diesem Fall langsam relativ zu den strikten Zeitbegrenzungen, mit denen sich eine Datenerfassung beschäftigen muß.

Das Slow Control System für das CHORUS-Experiment wird ca. 4000 Datenkanäle überwachen. Dabei wird gefordert, daß das System zuverlässig und leicht zu bedienen sein soll. Trotzdem muß es aber flexibel bleiben, damit neue Datenkanäle ohne Schwierigkeiten in die Überwachung aufgenommen werden können und neue Hardware leicht integriert werden kann. Um diesen Anforderungen zu genügen, wurde ein System aus voneinander unabhängigen Programmen entwickelt, die Daten durch einen definierten Mechanismus austauschen. Die Flexibilität wird dadurch gewährleistet, daß alle Parameter zu den Datenkanälen sich in Konfigurationsdateien befinden, mit denen das System initialisiert wird. So wird vermieden, daß bei jeder Änderung die Software umgeschrieben und neu kompiliert werden muß.

2. Neutrinooszillationen

2.1 Massive Neutrinos

Bis jetzt gibt es noch keine Beweise dafür, daß Neutrinos Masse haben.

Im minimalen Standardmodell wird vorausgesetzt, daß Neutrinos die Masse $\mathrm{m}_\nu = 0$ haben. Man muß dabei Quantenzahlen L_e, L_μ und L_τ einführen, um die verschiedenen Neutrinoarten zu unterscheiden.

In den über das Standardmodell hinausgehenden sogenannten Grand Unified Theories (GUTS) wird das Neutrino als Mitglied eines Multipletts zusammen mit einem positiv und einem negativ geladenen Quark sowie einem geladenen Lepton betrachtet:

$$F = \begin{bmatrix} q^+ \\ q^- \\ l \\ \nu \end{bmatrix}$$

In diesen Theorien, in denen das Neutrino ein gleichwertiges Mitglied im Multiplett ist, wäre es sogar erstaunlich, wenn es keine Masse hätte. Eine Erklärung dafür, daß die Masse der Neurinos so viel kleiner ist als die von Quarks und geladenen Leptonen, bietet dabei der See-Saw Mechanismus. In ihm wird angenommen, daß Neutrinos ihre eigenen Antiteilchen, also Majorana Teilchen sind. Für die Masse M_ν des Neutrinos ergibt sich dabei $M_\nu \approx M_d^2/M$, wobei M_d die Masse des geladenen Leptons oder eines Quarks derselben Generation ist und M in der Grössenordnung von $\sim 10^{14}\ GeV$ die Massenskala der GUT darstellt. Der See-Saw Mechanismus sagt also ein Verhältnis der Neutrinomassen von

$$m_{\nu_e} : m_{\nu_\mu} : m_{\nu_\tau} = m_e^2 : m_\mu^2 : m_\tau^2$$

oder

$$m_{\nu_e} : m_{\nu_\mu} : m_{\nu_\tau} = m_u^2 : m_c^2 : m_t^2$$

voraus. Dabei bedeuten u, c und t das Up, Charm und Top Quark.

Wenn Neutrinos eine Masse besitzen, so können Neutrinooszillationen auftreten. Dieser Mechanismus wird als eine mögliche Lösung des solaren Neutrinoproblems angesehen [Dav, Hir, Ans, Aba]. Neutrinos sind auch die einzigen bis jetzt nachgewiesenen Kandidaten für die Dunkelmaterie im Universum. Ein Neutrino mit einer Masse von $\sim 20\ eV$ würde das Massendefizit erklären. Wenn dabei die Massenverhältnisse der Neutrinos aus dem See-Saw Mechanismus betrachtet werden, dann ist das τ Neutrino der Hauptkandidat für die Dunkelmaterie, da die Massen der beiden anderen Neutrinos um einige Grössenordnungen kleiner sind [Har].

2.2 Theorie

Die Flavour-Eigenzustände ν_e, ν_μ und ν_τ der schwachen Wechselwirkung sind lineare Überlagerungen der Massen-Eigenzustände ν_m.

$$\nu_l = \sum_{m=1}^{3} U_{lm}\nu_m \qquad (l = e, \mu, \tau)$$

Die Matrix U_{lm} ist dabei die unitäre Mischungsmatrix. Sie ist analog zur Cabibbo Matrix angeordnet. Im Falle von zwei Neutrinoarten reduziert sie sich auf eine einfache 2x2 Rotationsmatrix.

$$U = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$$

Ohne Einschränkung der Allgemeinheit werden in den folgenden Formeln das Myon- und das Tauneutrino benutzt, da es die im CHORUS-Experiment wichtigen Neutrinoarten sind. Der Winkel θ bezeichnet in diesem Fall also den Mischungswinkel zwischen ν_μ und ν_τ. Die Wellenfunktionen sind orthogonal und haben die Form:

$$\begin{aligned} \nu_\mu &= \cos\theta\nu_2 + \sin\theta\nu_3 \\ \nu_\tau &= -\sin\theta\nu_2 + \cos\theta\nu_3 \end{aligned}$$

In einem Zerfall werden Eigenzustände der schwachen Wechselwirkung erzeugt, z. B. $\pi \to \mu + \nu_\mu$. Die Ausbreitung in der Raumzeit geschieht jedoch mit den charakteristischen Frequenzen der Masseneigenzustände. Es gilt:

$$\begin{aligned} \nu_m(t) &= \nu_m(0)e^{-iE_mt} \\ E_m = E(\nu_m) &= \sqrt{p_\nu^2 + M_m^2} \end{aligned} \tag{1}$$

Dabei ist $\hbar = c = 1$. Für den Fall, daß $M_m \ll p_\nu$ ist, gilt:

$$E_m \approx p_\nu + \frac{M_m^2}{p_\nu} \tag{2}$$

Wenn man zum Zeitpunkt $t = 0$ ein ν_μ erzeugt, dann ist $\nu_\mu(0) = 1$ und $\nu_\tau(0) = 0$. Dann folgt

$$\begin{aligned} \nu_2(0) &= \nu_\mu(0)\cos\theta \\ \nu_3(0) &= \nu_\mu(0)\sin\theta \end{aligned} \tag{3}$$

und

$$\nu_\mu(t) = \cos\theta\nu_2(t) + \sin\theta\nu_3(t)$$

Wenn man diese Formel mit (1) und (3) kombiniert, so erält man

$$\frac{\nu_\mu(t)}{\nu_\mu(0)} = \cos^2\theta e^{-iE_2t} + \sin^2\theta e^{-iE_3t}$$

Die Intensität beträgt

$$\frac{I_\mu(t)}{I_\mu(0)} = \left|\frac{\nu_\mu(t)}{\nu_\mu(0)}\right|^2 = 1 - \sin^2 2\theta \sin^2\left[\frac{(E_3 - E_2)t}{2}\right]$$

Mit der Definition von $\Delta m^2 = m_3^2 - m_2^2$ und der Verwendung von (2) folgt für die Wahrscheinlichkeit, am Ort L ein ν_μ zu messen

$$P(\nu_\mu \to \nu_\mu, L) = 1 - \sin^2 2\theta \sin^2\left(\frac{1,27\Delta m^2\,[(eV/c)^2]\,L\,[km]}{E\,[GeV]}\right)$$

Die Wahrscheinlichkeit, ein ν_τ zu messen, beträgt demnach

$$\begin{aligned} P(\nu_\mu \to \nu_\tau, L) &= 1 - P(\nu_\mu \to \nu_\mu, L) \\ &= \sin^2 2\theta \sin^2 \left(\frac{1,27\Delta m^2\,[(eV/c)^2]\,L\,[km]}{E\,[GeV]} \right) \end{aligned} \tag{4}$$

2.3 Die Messung von Neutrinooszillationen

Prinzipiell gibt es zwei verschiedene Methoden, nach Neutrinooszillationen zu suchen. Dies sind „Disappearance"- und „Appearance"-Experimente. In einem Disappearence-Experiment versucht man, die durch Neutrinooszillation bedingte Abnahme der Intensität eines Neutrinostrahls nachzuweisen. Das Hauptproblem bei Experimenten dieser Art besteht darin, den absoluten Neutrinofluß zu bestimmen. Größen wie der Wechselwirkungsquerschnitt oder die Detektoreffizienz, die mit einer Ungenauigkeit belegt sind, verschlechtern das Signal-Rauschverhältnis noch weiter. Aus diesem Grund benutzt man oft zwei gleichartige Detektoren, die sich in unterschiedlichen Entfernungen von der Neutrinoquelle befinden. Nachdem man die gemessenen Flüsse auf die Entfernung vom Detektor normiert hat, bestimmt man ihr Verhältnis und vergleicht das Ergebnis mit (4). Man muß dabei eine hohe Anzahl von Ereignissen nachweisen, um eine gute Statistik zu erhalten.

In einem Appearance-Experiment sucht man in einem zum Zeitpunkt t=0 reinen Strahl einer Neutrinosorte nach dem Vorkommen von Neutrinos einer anderen Generation. Eines der Probleme bei Appearance-Experimenten ist dabei die Verunreinigung des Strahls, in dem sich häufig auch Anteile der gesuchten Neutrinosorte befinden. Im Gegensatz zu den vorher genannten Disappearace-Experimenten genügt hier aber der Nachweis von nur wenigen Ereignissen der gesuchten Art, um Neutrinooszillationen nachzuweisen.

Aufgrund des kleinen Neutrinowechselwirkungsquerschnittes von $\sim 10^{-38}\, m^2/GeV$ müssen Neutrinodetektoren möglichst massiv sein, damit eine ausreichende Ereignisrate erzielt wird.

Als Ergebnis eines Neutrinooszillationsexperimentes erhält man eine

Aussage über die beiden Oszillationsparameter Δm^2 und $\sin^2 2\theta$, die über die Formel (4) miteinander verbunden sind. Die Ergebnisse führen deshalb auf sogenannte Ausschlußdiagramme, in denen die durch Δm^2 und $\sin^2 2\theta$ parametrisierte Fläche dargestellt wird. Eine entscheidende Bedeutung für den Parameterbereich, den ein Experiment mißt, hat dabei das Verhältnis L/E aus (4). Ein großes L/E bedingt eine hohe Sensitivität für kleine Δm^2, während ein kleines L/E zu einer hohen Empfindlichkeit für kleine $\sin^2 2\theta$ führt.

2.3.1 Vorherige Experimente

Alle in dieser Sektion beschriebenen Experimente haben Messungen zu der Neutrino-Oszillation $\nu_\mu \to \nu_\tau$ durchgeführt.

Das E531 Experiment am Fermilab benutzte einen Hybriddetektor, der aus einem Emulsionstarget, gefolgt von Driftkammern, einem Magneten, einem Kalorimeter und einem Myonenspektrometer bestand. Der prinzipielle Aufbau ähnelt also sehr dem CHORUS-Detektor (s. nächstes Kapitel). Es wurden zweimal Daten aufgenommen, 82 Tage lang zwischen November 1978 und Februar 1979 sowie 184 Tage zwischen November 1980 und Juni 1981. Das Emulsionstarget hatte in beiden Runs ein aktives Volumen von 22,6 l, mit einer Masse von 83 kg. Bei diesem Experiment wurde in einem gemischten Strahl aus Elektron- und Tauneutrinos nach ν_τ Wechselwirkungen gesucht. Es handelt sich also um ein Appearance-Experiment. Um ν_τ Wechselwirkungen nachzuweisen, wurde die Emulsion auf τ Zerfällen untersucht. Insgesamt wurden 3886 Neutrino Wechselwirkungen gefunden. Davon waren 104 Kandidaten für einen τ Zerfall. Nach Hintergrundsubtraktion und Cuts blieben keine Kandidaten mehr übrig. Dies führt zu der in Abb. 2.1 ausgeschlossenen Fläche für den Oszillationskanal ν_μ nach ν_τ . [Ush]

Die CDHS (CERN Dortmund Heidelberg Saclay) Kollaboration hat 1983 Messungen zu Neutrinooszillationen am CERN PS Beschleuniger durchgeführt. Es war ein Disappearance-Experiment, das zwei Detektoren benutzte. Die beiden Detektoren befanden sich in einer Entfernung von 130 und 885 m vom Target und besaßen aktive Massen von 100 und 600 t. Im ersten Detektor wurden 20000 und im zweiten 3000 Ereignisse gemessen. Die Durchschnittsenergie der Strahlneutrinos lag bei

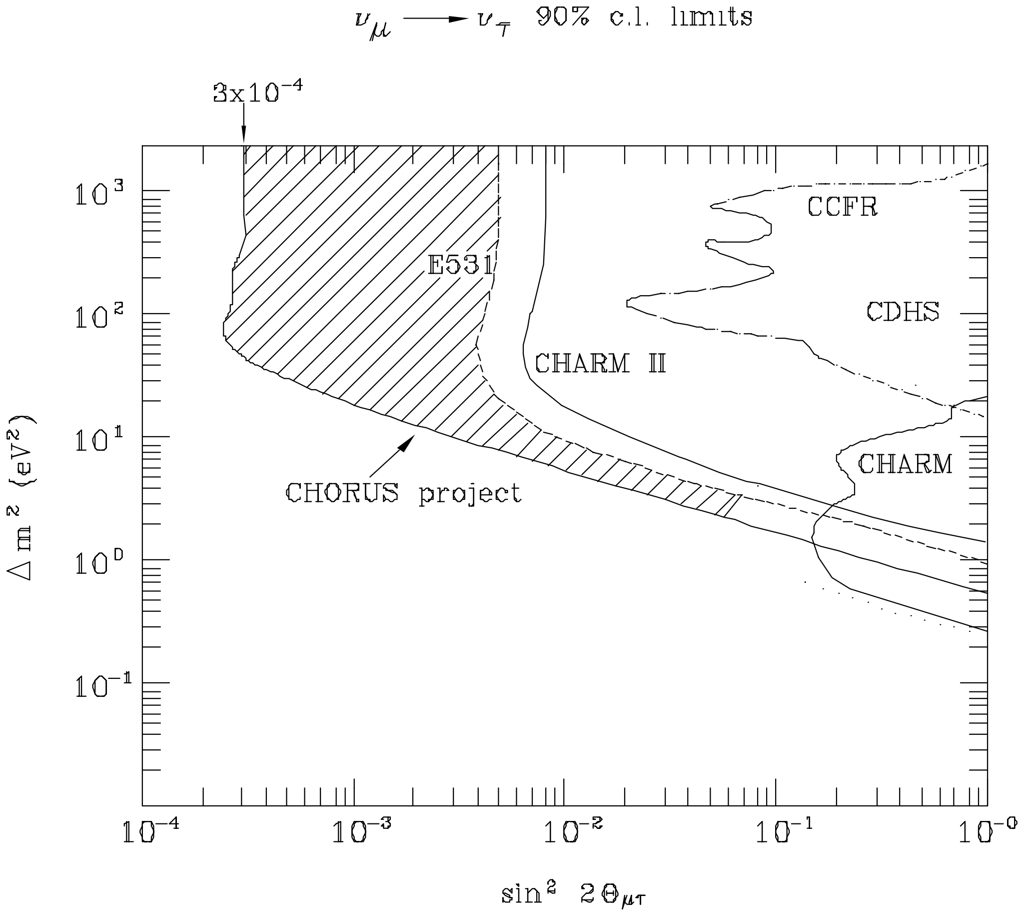

Abbildung 2.1: Ausschlußdiagramm für den Oszillationskanal $\nu_\mu \to \nu_\tau$

$\sim 1,5\,GeV$. Das nach der Normierung gewonnene Verhältnis war 1, es wurden also keine Neutrinooszillationen beobachtet. [Dyd]

Zur selben Zeit benutzte die CHARM (CERN Hamburg Amsterdam Rom Moskau) Kollaboration den gleichen CERN PS Strahl und einen ähnlichen Detektoraufbau wie die CDHS Gruppe, um ihre Messungen durchzuführen. Hier standen die zwei Detektoren in einer Entfernung von 123 und 903 m vom Target. Die aktiven Massen der Detektoren betrugen in diesem Fall 36 und 102 t. Es wurden im ersten Detektor 2000 und im zweiten 270 Ereignisse registriert. Auch hier wurden keine Hinweise auf Neutrinooszillationen gefunden. [Ber]

Wie auch die beiden vorher erwähnten Experimente benutzte die CCFR (Chikago Columbia Fermilab Rochester) Gruppe am Fermilab zwei Detektoren für ihre Messungen. Bei dem 1985 durchgeführten Experiment hatten die Detetektoren Massen von 105 und 444 t, bei einer Entfernung von 700 und 1100 m vom Target. Hier wurden 33700 und 33400 Ereignisse gemessem. Wie in den vorher beschriebenen Experimenten zeigte sich kein Hinweis auf Neutrinooszillationen. Im Gegensatz zu den beiden anderen Experimenten wurden hier jedoch Strahlneutrinos mit Energien zwischen 40 und 230 GeV verwendet. In Abb. 2.1 ist zu erkennen, daß dieses Experiment bei einem kleineren L/E auf große Δm^2 sensitiv war. [Sto]

Der CHARM-II Detektor am CERN bestand aus einem hochauflösenden Targetkalorimeter gefolgt von einem Myonenspektrometer. Zur Messung von ν_τ induzierten Ereignissen wurden 1991 Zerfälle vom Typ $\tau \rightarrow \pi\nu_\tau$ untersucht. Es wurden 124 Kandidaten für diese Zerfälle gefunden, doch alle Kandidaten konnten durch neutrale Strom-Wechselwirkungen erzeugt worden sein. Auch hier gab es keinen schlüssigen Beweis für Neutrinooszillationen. [Gru]

2.3.2 Das CHORUS-Experiment

Das CHORUS-Experiment [CHO] gehört zum Typ der Appearance-Experimente. Es sucht in einem ν_μ Strahl nach ν_τ Ereignissen. Gesucht wird dabei nach der Reaktion

$$\nu_\tau \mathrm{N} \rightarrow \tau^- X$$

mit den drei darauffolgenden Zerfallsmoden

$$\begin{aligned}
&\tau^- \rightarrow \mu^- \overline{\nu_\mu} \mu_\tau && (17.8\%)\\
&\tau^- \rightarrow \mathrm{h}^- (n\pi^0) \mu_\tau && (50.3\%)\\
&\tau^- \rightarrow (\pi^+\pi^-\pi^-)(n\pi^0)\mu_\tau. && (13.8\%)
\end{aligned}$$

Die Zahlen in den Klammern geben dabei das Branching-Ratio des Zerfallsmodus an.

Der Zerfallsknick des Tauleptons soll direkt nachgewiesen werden. Da es bei einer mittleren Lebenszeit t von ca. 0.3×10^{-12} s nur eine Stecke von ct=91 μm zurücklegt, muß ein Detektor mit einer sehr guten Ortsauflösung verwendet werden. Aus diesem Grund benutzt man Emulsion als Targetmaterial und erhält so eine Auflösung von ~ 1 μm.

Dem Emulsionstarget folgt ein elektronischer Detektor, der aus einer Gruppe von Trackern aus szintillierenden Fasern, einem magnetischen Spektrometer, einem hochauflösendem Kalorimeter und einem Myonenspektrometer besteht. Mit dieser Anordnung werden Ereignisse ausgewählt, die ein negatives Myon oder ein negatives Pion und einen fehlenden Transversalimpuls besitzen. Außerdem werden die Teilchenspuren bis zum Austrittspunkt aus der Emulsion zurückverfolgt. Dies muß mit einer hohen Ortsauflösung geschehen, um die zu scannende Fläche auf der Emulsion möglichst klein zu halten.

3. Der CHORUS-Detektor

Der CHORUS-Detektor verbindet in seinem Design altbewährte und neueste Technik. Als Target wird Emulsion verwendet, eines der am längsten bekannten Detektormaterialien. Die verwendeten Mengen sind jedoch größer als in jedem anderen Emulsionsexperiment zuvor. Andererseits benutzt man szintillierende Fasern und optoelektronische Ausleseketten, um die Teilchenbahnen zu bestimmen. Diese Technologie ist noch eine Neuentwicklung und wird zum ersten Mal in diesem Umfang in einem Detektor verwendet.

3.1 Emulsion

Das Emulsionstarget besitzt ein Totalvolumen von $230\,l$ und eine Masse von $800\,kg$. Es ist in 4 Lagen von je $1,44 \times 1,44\,m^2$ mit einer Dicke von je 2,75 cm unterteilt. Jeweils zwei dieser Lagen werden zusammen gruppiert. Weiterhin ist jede Lage in 8 Sektoren zu $0,72 \times 0,36\,m^2$ aufgeteilt. Jeder dieser Sektoren besteht aus 25 Emulsionsplatten (Sheets) (Abb. 3.2) . Eine Emulsionsplatte setzt sich aus zwei 550 μm dicken Schichten zusammen, die beidseitig auf einer 70 μm starken Polystyren Folie (Base) aufgebracht werden. Zum Scannen werden die Emulsionssektoren dann in $70 \times 70\,mm^2$ große Stapel (Tower) geschnitten. Die dabei entstehenden Mini-Plates werden dann gescannt. In den Mini-Plates liegen die Teilchenspuren fast senkrecht zu den Emulsionsplatten. Diese Konfiguration erlaubt es, schnelle semiautomatische wie auch automatische Scanning-Methoden bei der Auslese anzuwenden. Außerdem werden zwei Paare von Emulsionsplatten zwischen dem Target und dem ersten Target-Tracker in der Driftstrecke installiert. Diese Wechselplatten werden während des Runs alle drei Wochen ausgetauscht. Dadurch wird der entstehende Hintergrund gering gehalten. Das erste Paar befindet sich direkt vor dem ersten Target-Tracker. Die Spuren in diesen

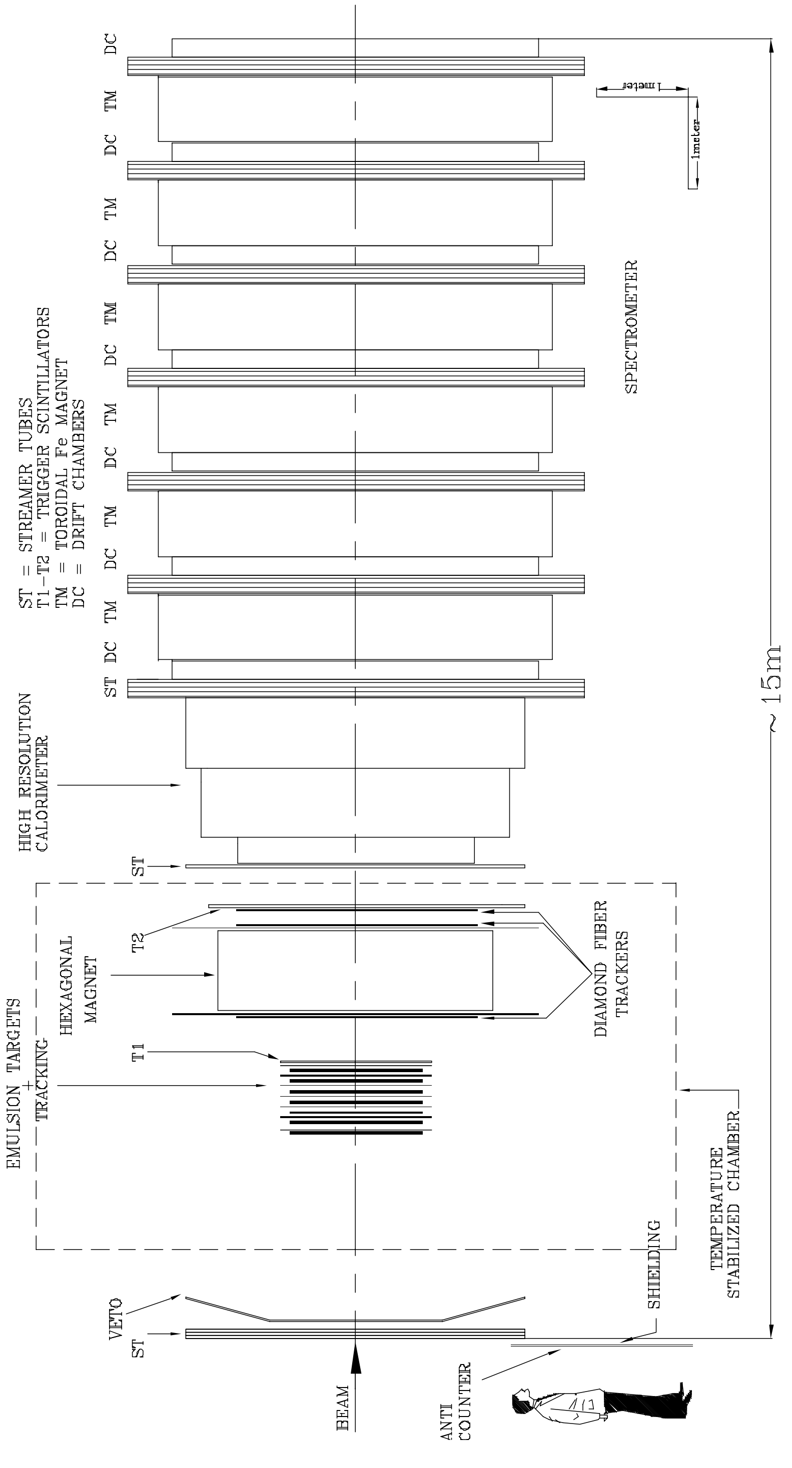

Abbildung 3.1: Der CHORUS-Detektor

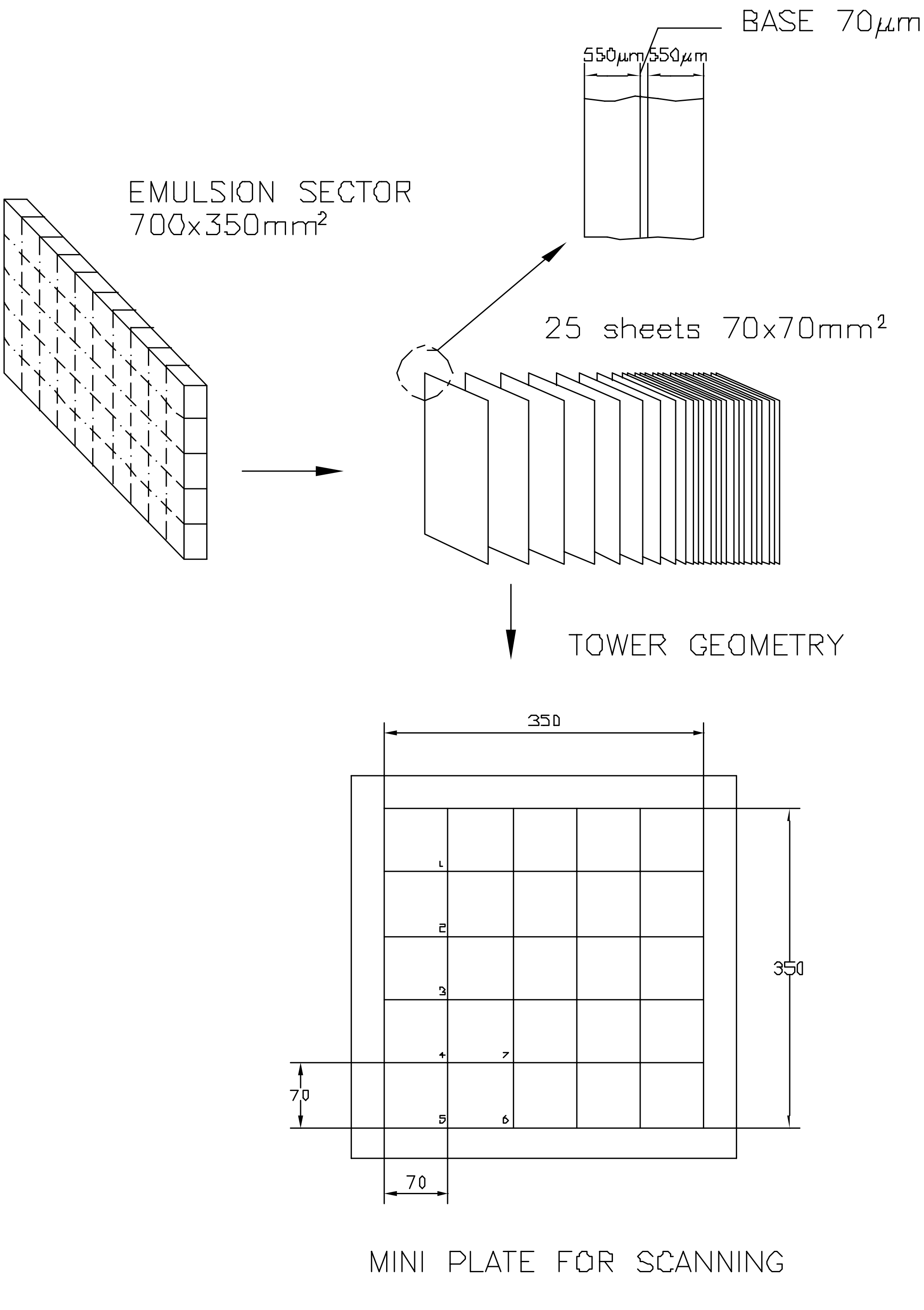

Abbildung 3.2: Aufteilung des Emulsiontargets

Wechselplatten können mit Hilfe der Target-Tracker in einer Fläche von $(300\mu m)^2$ lokalisiert werden. Automatische Auslese der Platten bestimmt die Position der Spur in einer Fläche von $(1\ \mu m)^2$. Dabei wird eine Winkelauflösung von $\Delta\theta \sim 0,2\ mrad$ erreicht. Damit auch Ereignisse mit hoher Multiplizität in den Target-Trackern aufgelöst werden können, befindet sich der erste Target-Tracker in einer Entfernung von $50\,mm$ vom Emulsionstarget. Das zweite Paar von Wechselplatten wird direkt vor dem Emulsiontarget angebracht. Hier muß eine Fläche von $(10\ \mu m)^2$ gescannt werden, um die Position der Spur auf eine Fläche von $(1\ \mu m)^2$ einzugrenzen. Auf diese Weise kann erreicht werden, daß trotz der hohen Spurdichte in der Emulsion die Auslesezeit nicht zu groß wird.

Die Emulsionslagen werden vakuumverpackt und in einen Rahmen eingebaut, in dem auch die Wechselplatten und die Target-Tracker positioniert sind. Um den unvermeidbaren Alterungsprozeß der Emulsion zu verlangsamen, befindet sich dieser Rahmen in einem Kühlraum, der auf $5°\ C$ gehalten wird. Da auch die szintillierenden Fasern bei niedrigen Temperaturen langsamer altern, befindet sich auch der Hexagonalmagnet mit den dazugehörigen Trackern in dem Kühlraum.

3.2 Tracker aus szintillierenden Fasern

Um die Teilchenspuren mit hoher Präzision in die Emulsion zurückzuverfolgen, befinden sich hinter den Emulsionsgruppen Tracker aus szintillierenden Fasern. Diese neue Technik wird verwendet, da es für das Experiment essentiell ist, die Teilchenspuren mit einer möglichst hohen Genauigkeit auf die Emulsion zurückzuverfolgen. Ein äquivalentes System aus Drahtkammern hätte eine schlechtere Zwei-Spur-Auflösung in der Nähe der Emulsion und eine bedeutend größere Länge in Strahlrichtung, wodurch die Akzeptanz des Kalorimeters und des Myonenspektrometers stark verkleinert worden wäre.

Die erste Trackerebene sitzt zwischen den zwei Emulsionlagen einer Gruppe, dahinter befindet sich eine Driftstrecke mit drei weiteren Ebenen (Abb. 3.3). Diese Anordnung wird hinter beiden Emulsionsgruppen wiederholt.

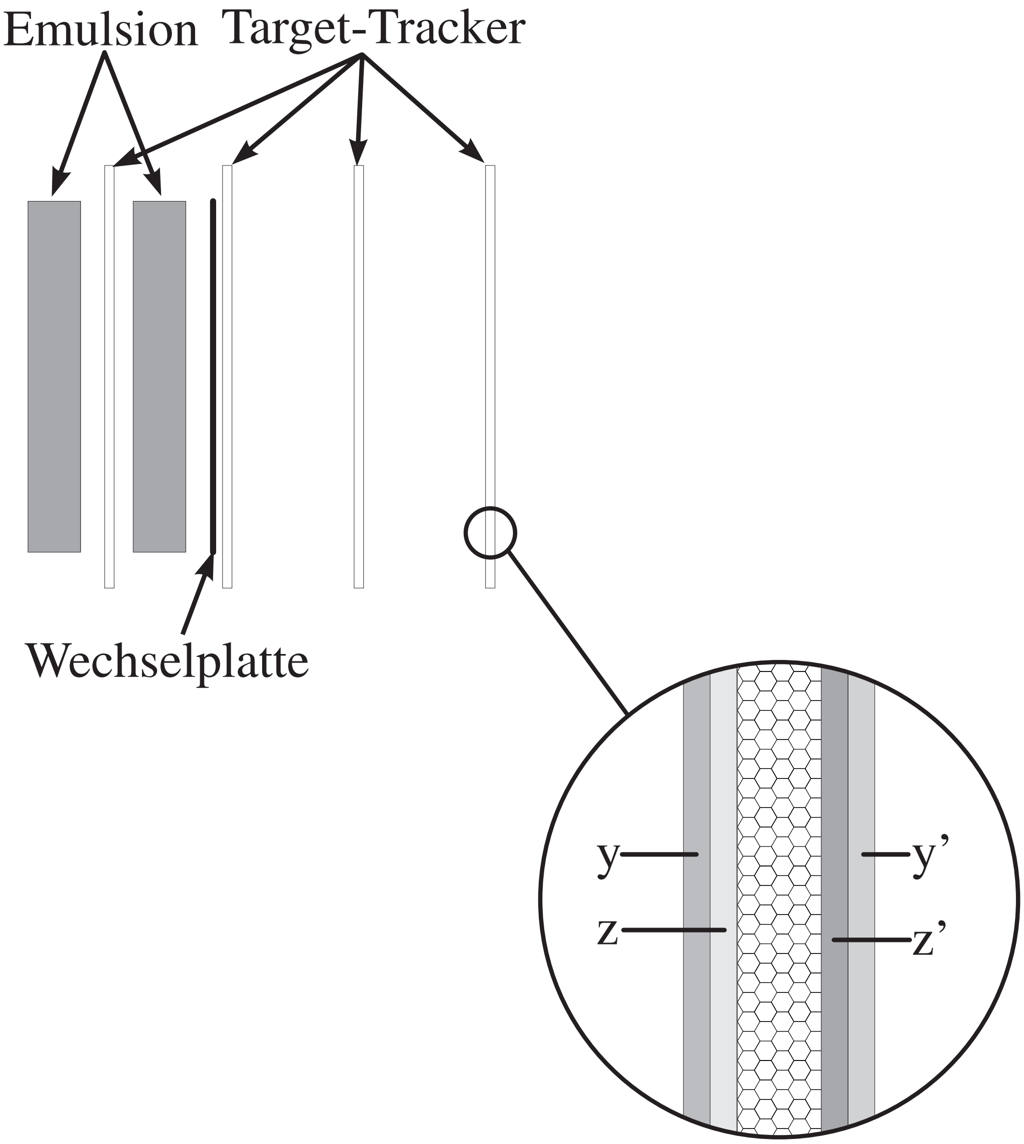

Abbildung 3.3: Die Lage der Target-Tracker

Jede Trackerebene besitzt Fasern in den 4 Projektionen y, z, y′, z′, um eine eindeutige Spuridentifizierung zu ermöglichen. Dabei sind die y′ und z′ Lagen um 8° gegenüber den y und z Lagen gedreht, wobei der Drehsinn zwischen +8° und −8° in den aufeinanderfolgenden Trackerebenen alterniert. Die Tracker zwischen den beiden Emulsionsgruppen erlauben eine Extrapolation der Spuren in die zweite Emulsionsgruppe. Somit kann ein aufwendiges Scannen durch mehrere Emulsionslagen vermieden werden. Die Tracker wurden entworfen, um eine Lokalisie-

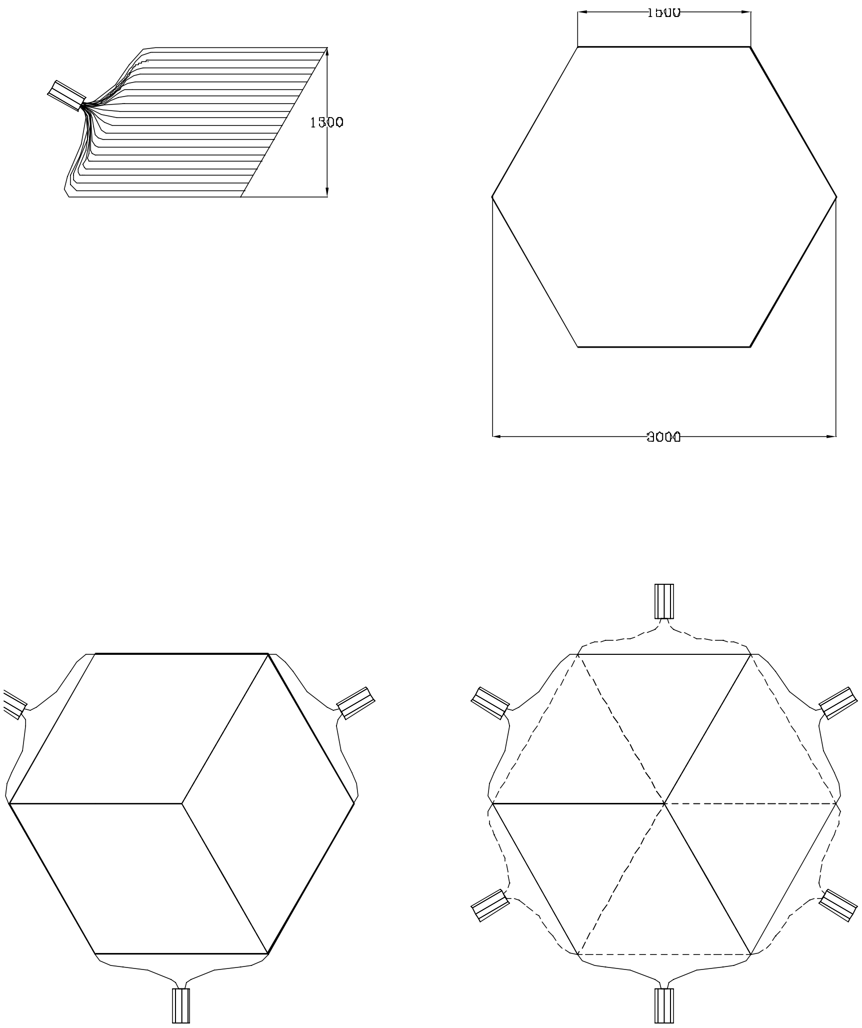

Abbildung 3.4: Der Aufbau eines Diamond-Trackers

rung der Spuren auf der Endfläche der Emulsionen in einer Fläche von $(300\mu m)^2$ und mit einer Winkelauflösung besser als $10mrad$ zu ermöglichen. Dabei soll eine gute Trennung zweier Spuren gewährleistet sein, um die hohe Spurdichte in der Nähe des Vertex aufzulösen.

Weiterhin gibt es drei hexagonale („Diamaond") Tracker, von denen sich einer vor und zwei hinter dem hexagonalen Magneten befinden. Der erste Diamond-Tracker ist 40 cm vom letzten Target-Tracker entfernt, um eine gute Winkelauflösung vor dem Magneten zu erreichen. Die Ablenkung im Magneten wird dann mit zwei weiteren Diamond-Trackern gemessen.

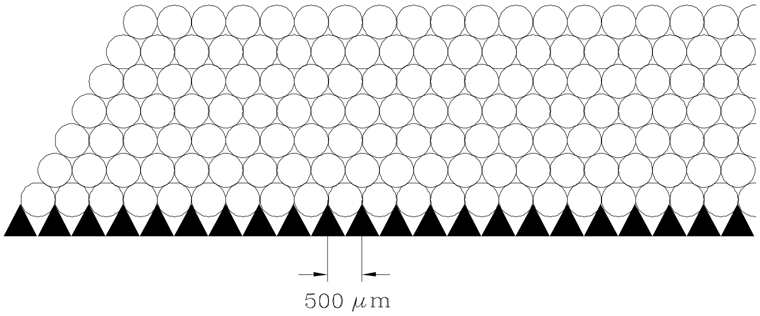

Abbildung 3.5: Der Aufbau eines Faserbündels

Jeder Diamond-Tracker besteht aus 6 Modulen (Abb. 3.4). Sie werden zu zwei Lagen zusammengesetzt. Dabei bilden die Fasern in beiden Lagen einen Winkel von 60°.

In beiden Arten von Trackern besteht jede Projektion aus 7 versetzten Lagen von szintillierenden Fasern mit einem Durchmesser von 500 μm (Abb. 3.5). Die Dicke des Claddings beträgt 3 μm. Die Fasern sind mit einer Farbe auf TiO_2 Basis zu Bündeln verklebt. Die Farbe dient gleichzeitig auch als Außenabsorber („Extramural Absorber").

Am Ende jedes Faserbündels wird ein Aluminiumspiegel mit 80% Reflektivität aufgesputtert. Das andere Ende wird poliert. Jeweils 16 Bündel werden dann an das Eintrittsfenster des ersten Bildverstärkers angekoppelt. Bei Tests in einem Pionenstrahl wurden 3 Photoelektronen pro Millimeter durchquerte Faser gemessen. Die Standardabweichung σ von der Spur betrug dabei 140 μm, die Zweispurauflösung betrug 500 μm.

Die Herstellung und Präparation der Faserbündel wird in Nagoya durchgeführt. Die fertigen Bündel werden dann am CERN zu den Target- und in Brüssel zu den Diamond-Trackern weiterverarbeitet.

Die Gesamtlänge der in den Trackern verwendeten Fasern beträgt ca. 2500 km.

3.3 Optoelektronische Ausleseketten

Zur Auslese der Faser-Tracker werden 58 Bildverstärkerketten eingesetzt. Jede Kette besteht aus 4 Komponenten (Abb. 3.6). Die erste

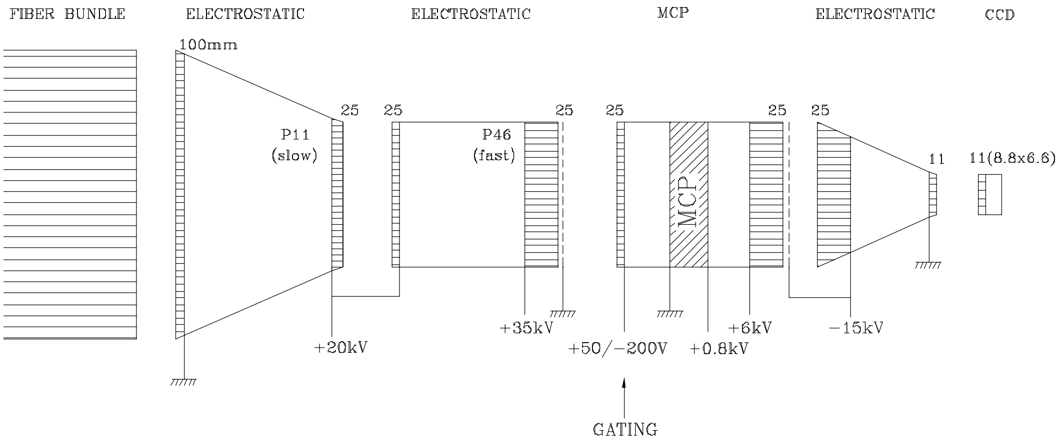

Abbildung 3.6: Eine optoelektronische Auslesekette

Stufe verkleinert das Bild von 100 mm Durchmesser auf 25 mm. Sie hat nur eine schwache Verstärkung, dafür aber eine hohe Quanteneffizienz. Die zweite Stufe ist ein elektrostatischer Verstärker, der die Größe des Bildes nicht ändert. Als dritte Stufe nimmt man ein MCP (Micro Channel Plate), das außer einer hohen Verstärkung auch noch die Möglichkeit der Triggerung durch ein externes Gate-Signal bietet. Die vierte Stufe schließlich verkleinert das Bild noch einmal von 25 mm Durchmesser auf 11 mm. Sie wird an ein CCD (Charge Coupled Device) angekoppelt, das eine Auflösung von 550×200 Pixeln besitzt. Ein Pixel hat dabei die Fläche von $16 \times 33\ \mu m^2$. Bei einer Verkleinerung des Bildes um den Faktor 9,1 bedeutet dies 6 Pixel pro Faser.
Im Experiment besteht dann die Aufgabe, die Signale der einzelnen Pixel einer Faser zuzuordnen. Dazu werden die Faserbündeln („Fibre Ribbons") am Eintrittsfenster des ersten Bildverstärkers mit dünnen Schichten aus Papier („Spacer", Dicke 0.2 mm) voneinander getrennt, um die einzelnen Faserbündel auseinander zu halten. Fünf Schichten mit eine Dicke von $0,5mm$ bestehen aus Araldit („Fiducials"). In ihnen sind jeweils 9 Plastikfasern ($\phi = 127\ \mu m$) zur Kalibration eingegossen (Abb. 3.7). Nach einer Korrektur für die Verzerrung der Bildverstärkerkette kann mit Hilfe dieser Kalibrationsfasern die absolute Position der Faserbündel bestimmt werden. Da die relative Lage der einzelnen Fasern untereinander bekannt ist, kann so die Zuordnung der Pixel zu den einzelnen Fasern berechnet werden.

In dem ersten Bildverstärker wird ein langsamer P11 Phosphor verwen-

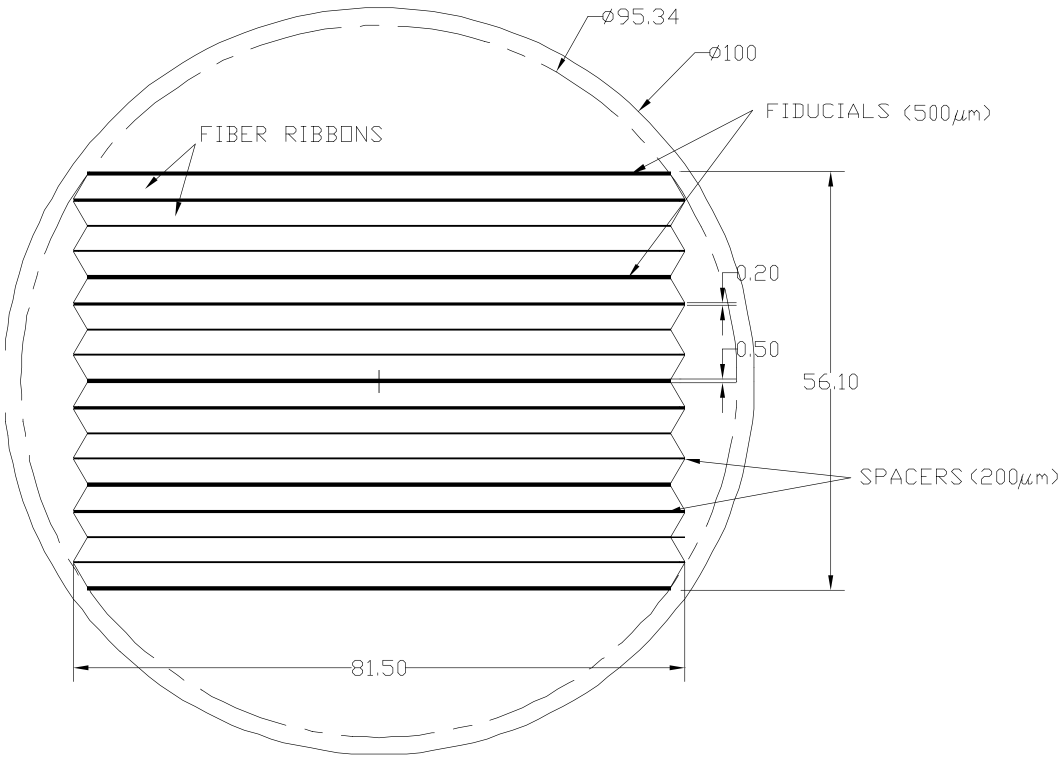

Abbildung 3.7: Anordnung der Faserbündel und der Kalibrationsfasern

det (Abklingzeit 50 μs). Alle anderen Röhren benutzen einen schnellen P46 Phosphor. Diese Anordnung erlaubt es, bei einer Triggerentscheidung von 500 ns und einer Gatelänge von 20 μs am MCP 36% des Lichtes zu messen.

Ein „Beamburst“ besteht aus zwei Spills von je 6 ms Dauer, in einem Abstand von $2,5\ s$. Pro Spill können 2 Ereignisse mit dem CCD erfaßt werden. Das erste Ereignis wird mit einer Operation (non-interlaced) in den internen Speicher des CCD transferiert, das zweite im sensitiven Bereich des Chips gespeichert. Die dabei entstehende Überlagerung von Ereignissen wird mit 8% veranschlagt. Mit Hilfe der anderen Subdetektoren kann die Überlagerung jedoch wieder herausgefiltert werden.

3.4 Der Hexagonalmagnet

Zwischen der Targetregion und dem Kalorimeter befindet sich ein Ablenkmagnet, um Impuls und Ladung der Teilchen vor dem Kalorimeter

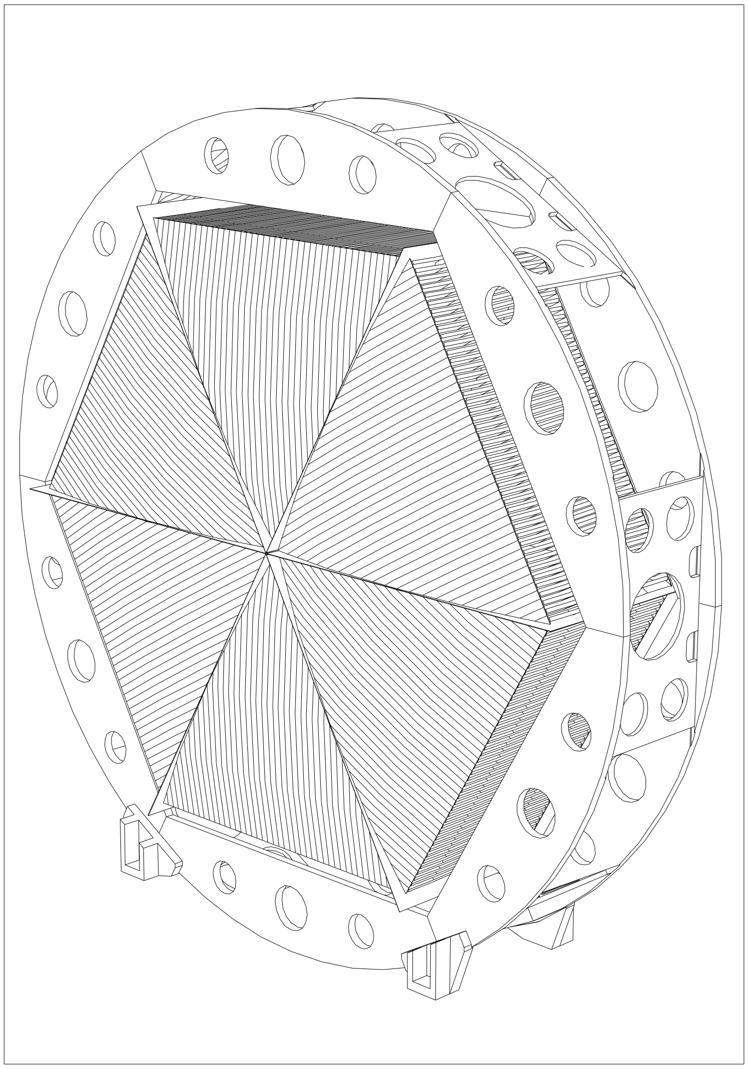

Abbildung 3.8: Der Hexagonalmagnet

zu bestimmen. Der Magnet muß verschiedene Bedingungen erfüllen. Er darf kein zu großes Streufeld haben, damit die elektrostatischen Bildverstärker in seiner Nähe betrieben werden können. Außerdem sollte er eine möglichst geringe Tiefe haben, um die Akzeptanz des Kalorimeters und des Myonenspektrometers nicht unnötig zu verringern. Weiterhin sollten die Teilchen auf ihrem Weg durch den Magneten nur wenig Material durchqueren, um die Wahrscheinlichkeit für Mehrfachstreuung zu vermindern.

Der für diese Bedingungen konstruierte Hexagonalmagnet besteht aus 6 gleichseitigen Dreiecken mit je $1,5\,m$ Seitenlänge und $0,75\,m$ Tiefe. Die

Sektoren sind in einen zylindrischen Rahmen mit $3,6\ m$ Durchmesser eingebaut. Die Windungen bestehen aus Aluminiumblechen, die alle Außenflächen eines Sektors überdecken. Das so entstehende Feld liegt in jedem Sektor parallel zum Ein- und Austrittsfenster. Es ist unabhängig vom Radius (Abb. 3.8).

Das Eintritts- und das Austrittsfenster des Magneten haben jeweils eine Stärke von $\leq 4\%$ einer Strahlungslänge. Die 6 Speichen des Magneten bilden einen toten Raum für die Impulsmessung. Wenn man über alle Spuren geladener Teilchen mit einem Impuls zwischen 2,5 und $10 GeV/c$ mittelt, so ergibt sich für den Magneten eine Akzeptanz von 85%. Aufgrund der Struktur des Neutrino Strahls wird der Magnet mit einem Duty-Factor von $\sim 4 \times 10^{-3}$ gepulst. Bei einer Leistung von $15\,kW$ wird ein Feld von $0,12\ T$ erreicht. Da der Magnet sich im Kühlraum direkt zwischen zwei Diamond-Trackern befindet, besitzt er ein eigenes Kühlsystem. Kalte Luft strömt im Inneren des Magneten an den Wicklungen entlang. Die erwärmte Luft wird dann wieder aus dem Kühlraum herausgeleitet.

Aus den kombinierten Meßfehlern der Richtungsmessungen vor und nach dem Magneten ergibt sich ein Fehler von

$\left(\frac{\Delta p}{p}\right)_{\text{Messung}} = 1,6\% \cdot p\ (GeV/c).$

Aus der Mehrfachstreuung ergibt sich ein impulsunabhängiger Meßfehler von

$\left(\frac{\Delta p}{p}\right)_{\text{Mehrfachstreung}} = 16\%.$

Quadratische Addition der beiden Werte ergibt einen totalen Fehler zwischen 16% (p=2 GeV/c) und 23% (p=10 GeV/c). Diese Auflösung ist gut genug, um das Vorzeichen der Ladung eines Teilchens mit einer Energie von bis zu 10 GeV mit einer Genauigkeit von 8 Standardabweichungen zu bestimmen. Die Impulsmessung wird gebraucht, um den Transversalimpuls der Pionen und die Reichweite von niederenergetischen Myonen (1-2 GeV) zu bestimmen.

3.5 Streamer-Kammern

Vor dem Kalorimeter, zwischen den Kalorimetersegmenten und in den Zwischenräumen des Myonenspektrometers werden CHARM II Streamer-Kammern zur Mustererkennung eingebaut.

Die Streamer-Kammern werden in Vierergruppen installiert, je zwei mit senkrechten und zwei mit waagerechten Drähten. Der Abstand der Drähte beträgt 1 cm, die Kathodenstreifen sind 2,1 cm voneinander entfernt. Dabei werden die Drähte digital (Information: Signal/kein Signal) und die Kathodenstreifen analog ausgelesen. Eine Gruppe liefert 8 gemessene Koordinaten. Durch Driftzeitmessung an Gruppen von 8 Drähten kann die Präzision noch verbessert werden. Die Meßgenauigkeit beträgt $\sigma \sim 1\,mm$.

3.6 Hochauflösendes Kalorimeter

Die Unterscheidung der Reaktionen

(i) $\nu_\tau \mathrm{N} \rightarrow (\tau^- \rightarrow \mu^- \nu_\mu \nu_\tau)\mathrm{X}$, $(\tau^- \rightarrow \pi^- \nu_\tau)\mathrm{X}$
und $(\tau^- \rightarrow \pi^- \pi^+ \pi^- (n\pi^0)\nu_\tau)\mathrm{X}$

von

(ii) $\nu_\mu \mathrm{N} \rightarrow \mu^- \mathrm{X}$ und $\nu_\mu \mathrm{N} \rightarrow \nu_\mu \mathrm{X}$

erfolgt mit Hilfe des fehlenden Transversalimpulses, der von den Neutrinos in (i) fortgetragen wird. Außerdem kann die charakteristische Korrelation des Vektors des fehlenden Transversalimpulses mit dem Richtungsvektor des hadronischen Transversalimpulses ausgenutzt werden. Bei ν_τ Reaktionen sind ihre Vektoren gegenläufig, während bei ν_μ Reaktionen die Richtungen nahezu symmetrisch sind. Aus diesem Grund wird ein Kalorimeter mit einer hohen Energieauflösung benutzt, um die Richtung und Energie des Hadronenschauers zu bestimmen.

Im Kalorimeter induzierte Schauer tendieren dazu, sich auszubreiten und zu überlappen. Daher können die Energievektoren nicht individuell gemessen werden. Es wird der Energievektor des gesamten hadronischen Endzustandes bestimmt, indem man den lateralen Schwerpunkt des Energieflusses im Kalorimeter bestimmt und mit der Vertexposi-

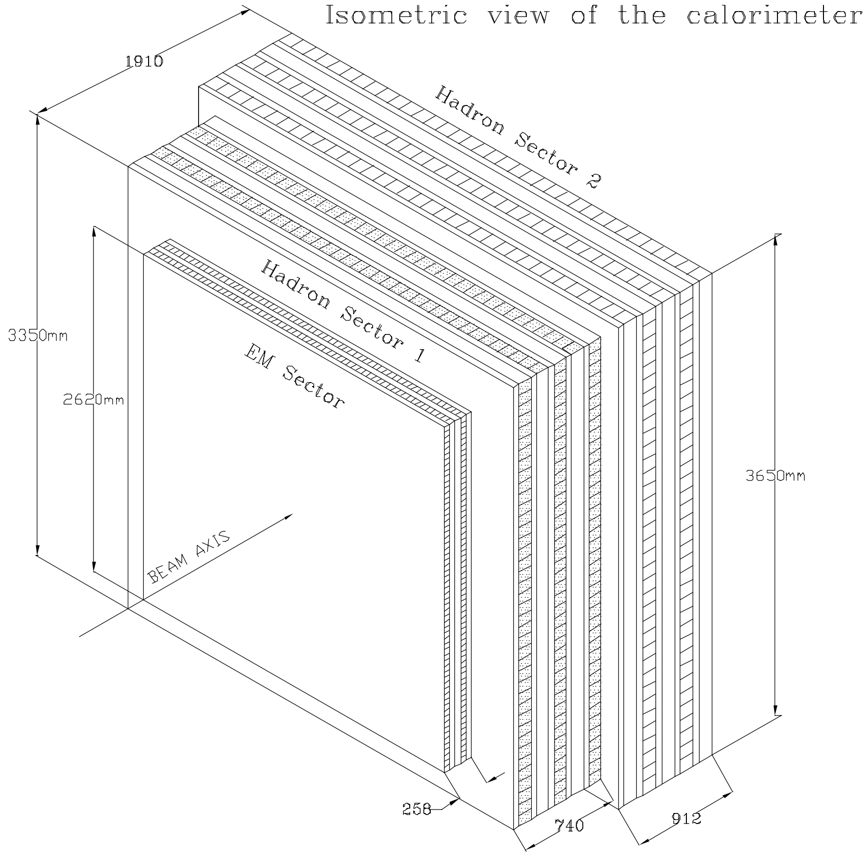

Abbildung 3.9: Das Kalorimeter

tion kombiniert, die man mit den Faser-Trackern bestimmt hat. Dementsprechend wurde die Modularität des Kalorimeters gewählt. Jede Kalorimeter-Ebene besteht aus senkrecht zum Strahl orientierten Modulen, die in zwei Projektionen abwechselnd senkrecht und waagerecht angeordnet sind. Wenn die Querschnitte der Module so gewählt werden, daß sie kleiner sind als die Schauergröße eines Hadrons, dann ist der Meßfehler einzig durch die Statistik der Energiemessung bestimmt. Aus diesem Grund wurde ein hochauflösendes Kalorimeter entworfen. Es soll außerdem das gleiche Ansprechvermögen für die Leptonen aus dem Zerfall geladener Pionen und für die Photonen aus dem Zerfall neutraler Pionen haben. Um diese Bedingungen zu erfüllen, benutzt man ein kompensierendes Bleikalorimeter mit einem Volumenverhältnis Blei zu Szintillator von 4 zu 1.

Die Sekundärteilchen aus dem τ Zerfall besitzen normalerweise eine

Energie kleiner als $5\,GeV$. Um 99% (90%) eines hadronischen Schauers zu absorbieren, benötigt man etwa 5,2 (3) Absorptionslängen.

Das Kalorimeter besteht aus drei Sektionen mit zunehmender Granularität. Sie heißen EM, HAD1 und HAD2. Die erste Sektion dient dazu, die elektromagnetische Komponente des Ereignisses zu messen. In den ersten beiden wird die hadronische Komponente gemessen, und die dritte Komponente wird benutzt, um die Restenergie zu messen, die in den ersten beiden Modulen nicht absorbiert werden konnte. (Abb. 3.9)

Die ersten beiden Sektionen (EM und HAD1) bestehen aus Blei und Szintillationsfasern (Spaghettikalorimeter). Sie können 90% eines Hadronenschauers absorbieren. Die dritte Sektion (HAD2) ist ein Sandwich-Kalorimeter aus Blei und Szintillatorstreifen, da die Messung des Endes eines Schauers eine gröbere Orts- und Energieauflösung erlaubt (Abb. 3.10).

Das Kalorimeter besteht aus vertikalen und horizontalen Modulen, mit eingeschobenen Streamer-Kammern. Diese ortsauflösenden Kammern sind notwendig, um die Spuren in dem Myonenspektrometer mit den in den Faser-Trackern gemessenen Spuren zu verbinden.

Die Module der EM Sektion bestehen aus mehreren Lagen gezogener Bleiplatten mit Rillen. In den Rillen befinden sich dann die Plastikszintillationsfasern. Ein Modul setzt sich aus 21 Lagen mit 2620 mm Länge und 740 Fasern zusammen. Es hat eine Querschnittsfläche von $40 \times 82,4\ mm^2$. An beiden Seiten werden die Fasern in zwei Bündeln geordnet, was zu einer Zellengröße von etwa $40 \times 40\,mm^2$ Querschnittsfläche führt. Jedes der vier Bündel wird dann über einen Lichtleiter aus Plexiglas an einen Photomultiplier angekoppelt. Die HAD1 Module sind ähnlich aufgebaut. Die Bleilagen sind hier 3350 mm lang, und ein Modul besteht aus 43 Lagen mit je 1554 Fasern. Hier erhält man eine Querschnittsfläche von etwa $80 \times 80\ mm^2$. Die Lichtleiter werden benötigt, um das Licht der individuellen Fasern zu mischen. Die HAD2 Module bestehen 16 mm dicken Bleiplatten, abwechselnd mit 4 mm dicken Szintillatorplatten. Sie sind 3650 mm breit, 200 mm hoch und 100 mm dick. Auch diese Module werden in zwei Gruppen ausgelesen.

Eine Gruppe mit zwei Streamer-Kammern (horizontal und vertikal) befindet sich hinter jeder zweiten Ebene von EM Modulen und hinter

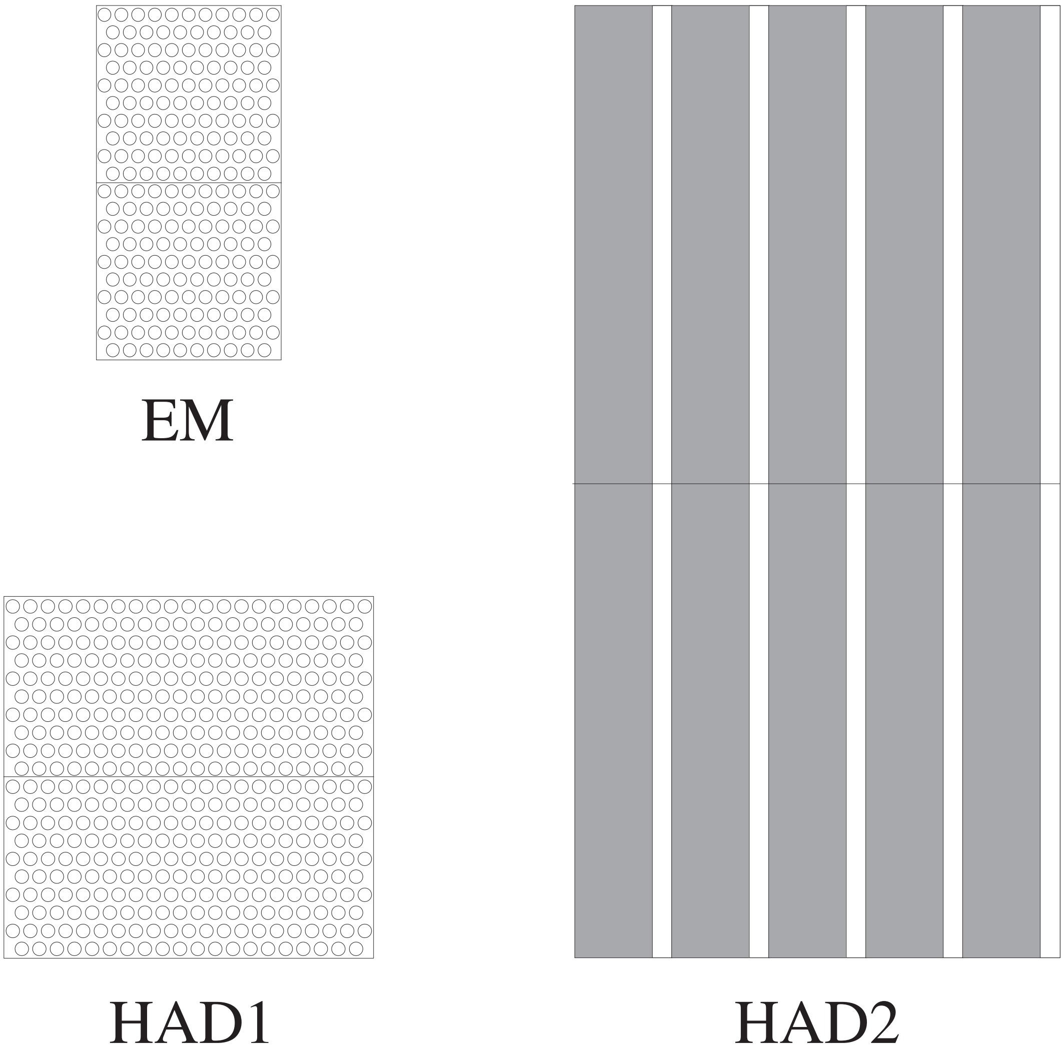

Abbildung 3.10: Die unterschiedlichen Kalorimetermodule

jeder Ebene von HAD1 und HAD2 Modulen.

Die Energieauflösung des Spektrometers wurde an Prototypmodulen gemessen. Sie beträgt für Elektronen

$\Delta(E_e)/E \approx 13\%/\sqrt{E}\ (Gev)$

und für Pionen

$\Delta(E_\mu)/E \approx 30\%/\sqrt{E}\ (Gev)$.

Die Unterscheidung von e/π kann mit einer Genauigkeit von 10^{-3} ausgeführt werden. Dies ist möglich durch den Vergleich der Schauerbreite und durch den unterschiedlichen Energieverlust in der elektronischen

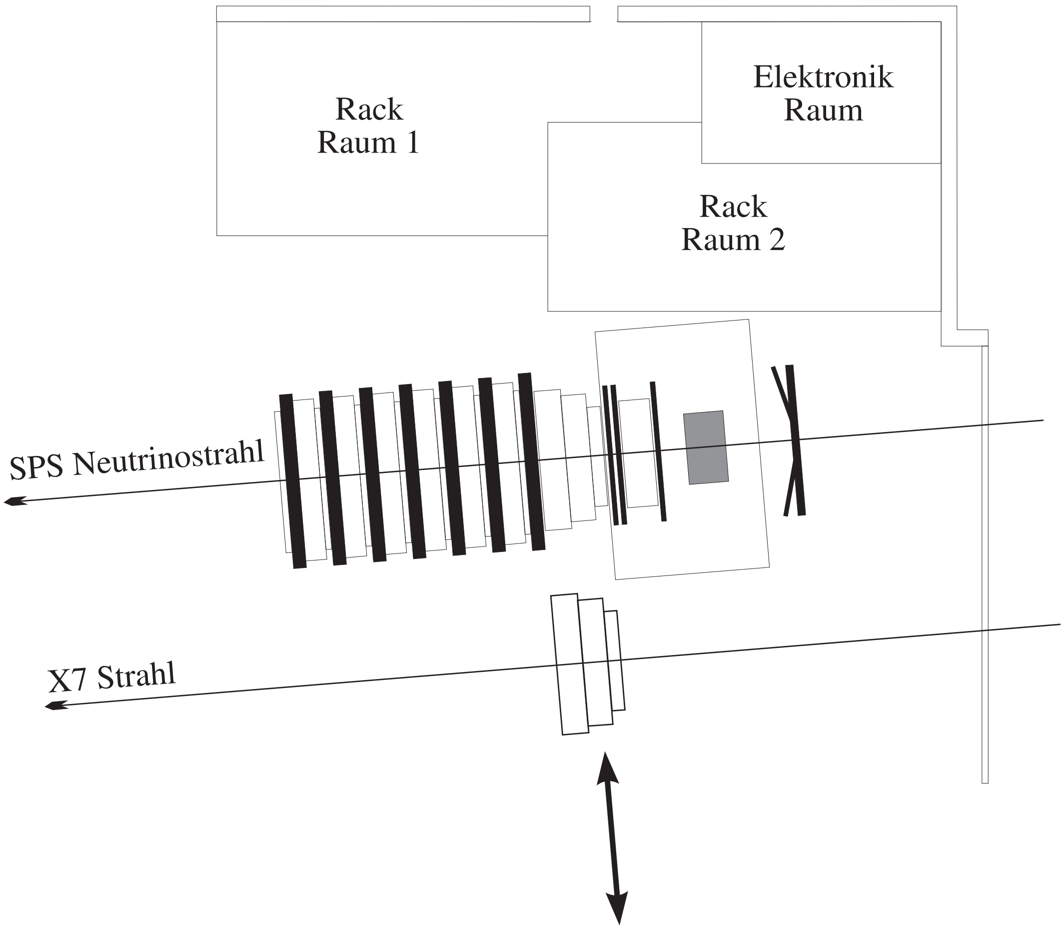

Abbildung 3.11: Die Lage des Detektors in der BEBC Halle

und der hadronischen Sektion des Kalorimeters. Dadurch können Elektronen aus konvertierten Photonen von der Rekonstruktion des Vertex ausgeschlossen werden, da ihre Spur ja auf einen falsche Vertexposition zeigt.

Die Position des Schwerpunktes eines Hadronenschauers bei einer Energie von $5\,GeV$ kann mit einer Auflösung von $\sigma \sim 1,5\,cm$ bestimmt werden. Indem man dies mit der Genauigkeit der Vertexmessung von $\sigma \sim 0,3\,cm$ kombiniert, erhält man eine Winkelauflösung von $\sigma \sim 10\,mrad$ bei einer Energie von $5\,GeV$.

Das Kalorimeter wird kalibriert und seine Qualität überwacht, indem man zwischen den Neutrino-Spills Ereignisse mit kosmischer Strahlung mißt.

Eine genaue Energieeichung wird mit Elektronen und Pionen des SPS X7 Strahls vorgenommen. Dazu wird das Kalorimeter auf Schienen aus dem Detektor herausgefahren (Abb. 3.11).

3.7 Myonenspektrometer

Das Myonenspektrometer besteht aus sechs zirkular magnetisierten Eisenmodulen mit $3,75\ m$ Durchmesser und sieben Tracking Sektionen, eines vor den Eisenmodulen, fünf in den Lücken zwischen den Modulen und eines dahinter (Abb. 3.12). Jedes Eisenmodul besteht aus zwanzig Eisenplatten mit $2,5\ cm$ Dicke. Die Wicklungen laufen von einem Loch in der Mitte radial nach außen. Die magnetische Feldstärke im Inneren des Magneten beträgt $\sim 2\,T$. Die Lücken zwischen den Platten sind mit $8\ mm$ dicken und $15\ cm$ breiten Szintillatoren ausgefüllt, die abwechselnd horizontal und vertikal geführt werden. Jeweils 5 Szintillatoren werden an einen Photomultiplier angekoppelt. Die Reichweite der Myonen in Eisen kann so mit einer Schrittweite von $12,5\ cm$ berechnet werden.

Jede der sieben Tracking Sektionen besteht aus einer Driftkammer und einer Gruppe von acht Streamer-Kammern. Jede Driftkammer besitzt drei Gruppen von Meßdrähten in den Richtungen $0°$, $+60°$ und $-60°$ bezüglich zur Horizontale. Driftzeitmessungen werden mit Gruppen von acht Drähten gemacht, was zu einer Auflösung von $1\ mm$ führt.

Aus jeder Tracking-Sektion erhält man 19 gemessene Koordinaten, die zusammengenommen den 3-dimensionalen Bahnvektor jeder Spur bestimmen. Aus einem Fit dieses Vektors an die Myonenspur erhält man den Impuls. Die Auflösung hängt dabei von der Anzahl der durchquerten Module ab. Ein $20\ GeV$ Myon, das alle sechs Module durchquert, wird mit einer Auflösung von 15% gemessen. Durchquert es nur ein Modul, so beträgt die Auflösung 30%.

Die Szintillatoren in den Eisenmodulen dienen vier Zwecken. Sie geben ein Trigger-Signal für ein eindringendes Teilchen und sie ergänzen die Messung von Hadronenenergien, die nicht vollständig im Kalorimeter dissipiert werden. Außerdem kann mit ihnen die Reichweite eines Myons, das in einem Modul gestoppt wird, sowie die durch Bremsstrah-

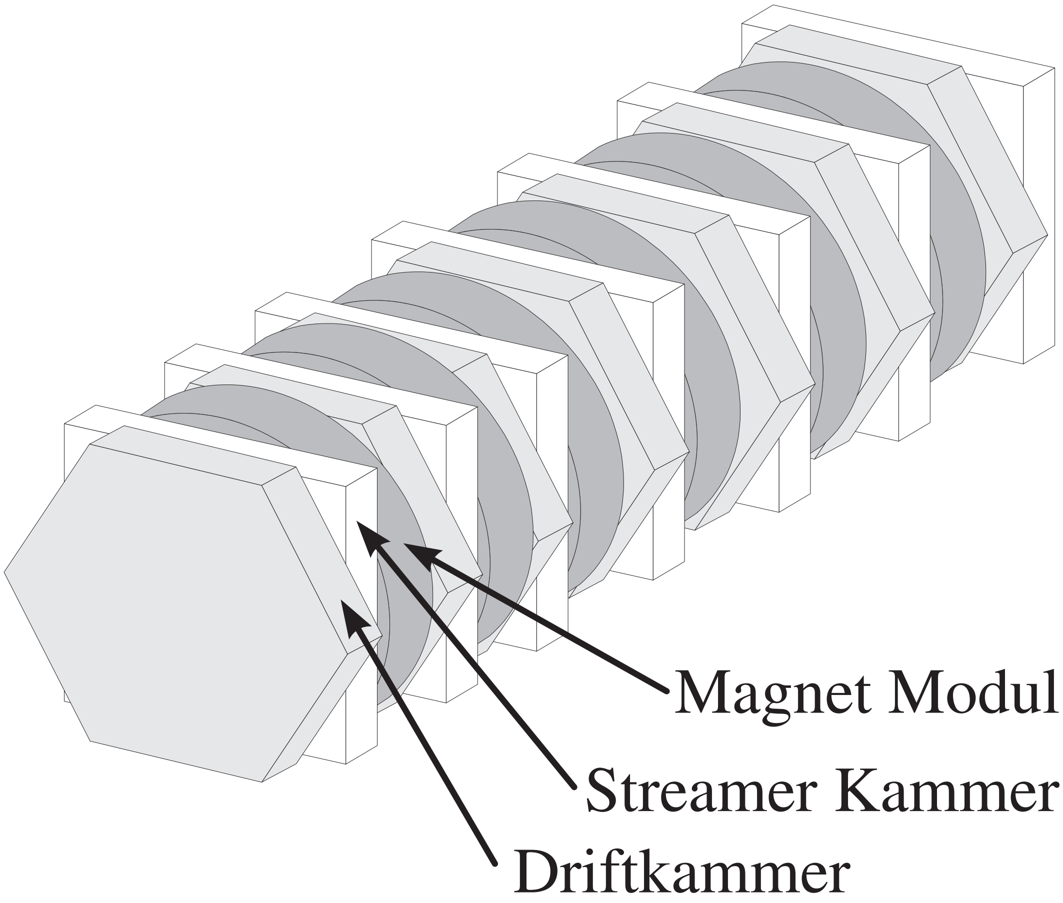

Abbildung 3.12: Das Myonenspektrometer

lung des Myons im Eisen verlorene Energie gemessen werden.

3.8 Trigger

Die Hauptaufgabe des Triggers besteht darin, Neutrinoinduzierte Ereignisse in der Emulsion zu selektieren und den Hintergrund durch kosmische Strahlung, Strahlmyonen und Wechselwirkungen außerhalb der Emulsion auszuschließen. Zusätzlich wird der Trigger für die Kalibrierung und die Ausrichtung des Detektors benötigt. Die erwartete Neutrino Ereignisrate liegt bei 0,7 Ereignissen pro Spill bei 10^{13} Protonen auf dem Be Target und einer effektiven Masse von 2400 kg.

Zwei Hodoskope werden als Neutrino Wechselwirkungs Trigger benutzt. Das erste, T1, befindet sich direkt hinter der Target Region, das zweite, T2, direkt vor dem Kalorimeter (Abb. 3.1). T1 besteht aus zwei Lagen von fünfzehn 10 cm breiten und 160 cm langen horizontal orientierten Szintillatoren. Die zwei Lagen überlappen sich und haben eine Gesamtfläche von $160 \times 160\ cm^2$. Sie sind in ihrer Dicke optimiert, um einen Verlust an Meßgenauigkeit durch Mehrfachstreuung zu minimieren. T2 besteht aus zwei Schichten von 20 cm dicken und 200 cm langen Szintillatoren. Sie sind wie in T1 horizontal orientiert.

Eine Veto-Ebene (V) aus zwanzig 20 cm breiten Szintillatoren befindet sich 2 m vor der Target-Region. Sie hat eine Fläche von $4 \times 3,2\ m^2$. Diese Größe wurde gewählt, um die volle Winkelakzeptanz der Trigger-Hodoskope zu überdecken.

Ein Neutrinowechselwirkungs-Trigger besteht aus einer Kombination von Pulsen in T1 und T2, wobei der Winkel der Spur zum Neutrinostrahl nicht mehr als 14° betragen darf. Ein Veto Trigger besteht aus Pulsen in V und T1. Dabei wird ein striktes Timing verlangt, um Veto Signale durch Rückstreuung zu vermeiden. Aus diesem Grund werden die Szintillatoren in V und T1 auch beidseitig von Photomultipliern ausgelesen. Aus den Ankunftszeiten der Pulse an den beiden Photomultipliern läßt sich dann der genaue Zeitpunkt des Teilchendurchgangs bestimmen.

Der Myonenfluß in der BEBC Halle[1] wurde während der Operation des Neutrino-Strahls gemessen. Er betrug $N_\mu \approx 500/m^2$ für ein Bündel mit 2×10^{13} Protonen. Da die Abschirmung in der Halle schwach ist, wurde sie vor dem Detektor mit Eisen verstärkt. Dadurch konnte der Myonenfluß um einen Faktor ~ 10 abgeschwächt werden. Der Abstand zwischen dem Detektor und der Abschirmung beträgt ca. 10 m, um den Hintergrund von Neutrino induzierten Partikeln gering zu halten. Die Zahl der einfallenden Myonen muß so weit wie möglich reduziert werden, damit die Spurdichte in der Emulsion nicht zu groß wird.

Wenn man die Spuren der Strahl-Myonen ausgeschlossen hat, so besteht der Hintergrund hauptsächlich aus Neutrino Ereignissen im Material um den Detektor, z.B. Stützen oder Fußboden. Diese Ereignisse erfüllen die Trigger Bedingungen, wenn mehrere Spuren in den Trackern einen

[1] Der vormalige Standort der Big European Bubble Chamber. In dieser Halle ist jetzt das CHORUS Experiment aufgebaut.

schmalen Winkel vortäuschen. Dieser Hintergrund wird minimiert, indem man die Materialdichte im Trigger-Volumen so klein wie möglich hält. Der erwartete Hintergrund durch kosmische Strahlung ist klein gegen die anderen Quellen.

3.9 Datenerfassung

Die Ereignisrate im CHORUS Experiment ist gering, im Durchschnitt weniger als ein Ereignis pro Neutrino Burst. Jedes Ereignis enthält große Datenmengen (ca. 20 Mb CCD Daten, ca. 10 Mb Daten aus Kalorimeter, Spektrometer und Streamer Kammern). Bevor alle Daten zu einem Ereignis zusammengefaßt werden, müssen sie auf einige 100kb komprimiert werden, um zu große Bandbreiten für die Datenbusse zu vermeiden.

Die Möglichkeit, bis zu 3 oder 4 Ereignisse pro Burst auszulesen, erfordert eine Obergrenze von wenigen $100\,ms$ Auslesezeit pro Ereignis. Das Datenvolumen wird in etwa verdoppelt durch Kalibrationsmessungen mit kosmischer Strahlung zwischen den Bursts. Es wird erwartet, daß alle 45 Minuten ein 200 MB Tape gefüllt wird.

Die Datenerfassung basiert auf VME-Bus Hardware, die mit dem OS9 Betiebssystem ausgestattet ist. Das Herz des Systems bildet eine zentrale CPU (Central Processing Unit), auf der das Event-Builder-Programm läuft, und das durch einen SCSI-Bus mit einer dualen 3480-Cartridge Einheit verbunden ist. Die Aufgabe der zentralen CPU ist es, Ereignisse aus den komprimierten Daten der anderen Crates zusammenzusetzen, die Ereignisse auf Bänder zu schreiben und ausgewählte Ereignisse zur Überwachung aufzubereiten.

Die Daten der Faser Tracker werden in den Kameras digitalisiert und in dafür reservierten CPUs aufbereitet. Jede dieser CPUs verarbeitet die Daten von zwei Kameras. Die Daten werden mit Hilfe des VIC-Busses (Vertical Inter Crate Bus) zwischen den Crates ausgetauscht. Es gibt Interface-Module zu CAMAC Crates, in denen ADC- (Analog Digital Converter) und TDC- (Time Digital Converter) Module sowie Trigger-Logikmodule untergebracht sind.

Jedes TDC- und ADC- Modul ist mit Ereignis-Buffern ausgestattet.

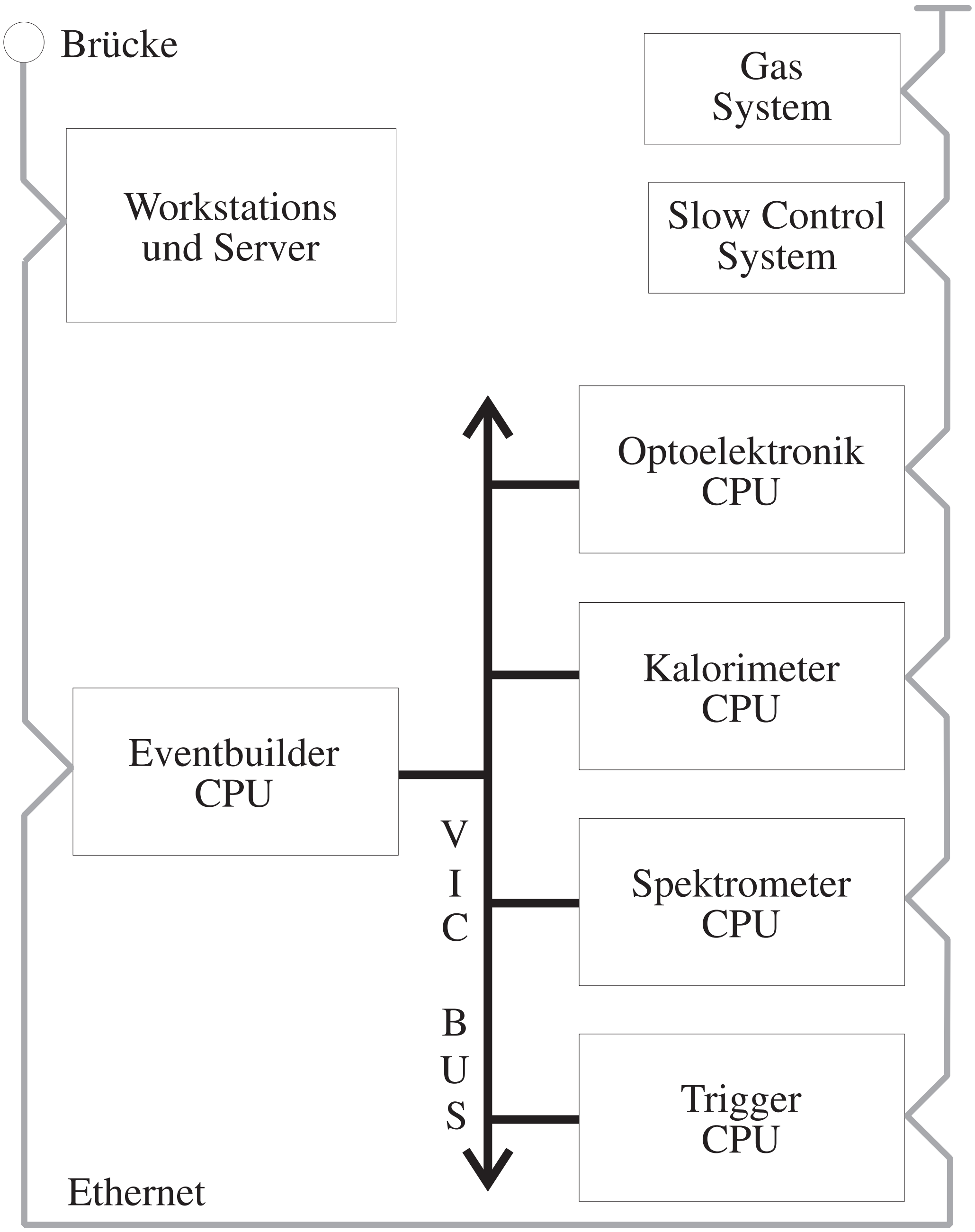

Abbildung 3.13: Das CHORUS Datenerfassungssystem

Ihre Umrechnungszeiten liegen unter 20 μs. Ereignisse werden während der Neutrino Bursts in den Buffern gespeichert. Diese werden zwischen und nach den Bursts ausgelesen.

Die Qualität der Daten wird auf Unix Workstations überwacht, die durch Ethernet mit allen VME Crates verbunden sind (Abb. 3.13). Auf ihnen werden ausgewählte Daten analysiert, Online-Kalibrationen ausgeführt und auf den OS9 Systemen erstellte Histogramme dargestellt. Für das Experiment wichtige Daten werden laufend auf den Bildschirmen der Workstations dargestellt.

4. Slow-Control

4.1 Einführung in Slow-Control

In einem physikalischen Experiment wie z.B. CHORUS wird eine große Menge von Daten aufgenommen. Sie sind notwendig, um ein physikalisches Ereignis zu messen und zu charakterisieren. Es gibt aber auch andere Parameter, die überwacht werden müssen, um ein Gelingen des Experimentes zu ermöglichen. Sie dienen zur Kontrolle der Bedingungen, unter denen ein Ereignis aufgenommen wird. Vor allem sind hier die Versorgungsspannungen der elektronischen Geräte zu nennen, aber auch Umgebungsdaten wie Temperatur, Luftdruck und Luftfeuchtigkeit. Diese Parameter werden von sogenannten Slow-Control-Systemen kontrolliert und überwacht. Sie sind hauptsächlich dafür konzipiert, eine Online-Kontrolle über den Zustand des Experimentes zu erhalten und über den aktuellen Zustand des Experimentes zu informieren. Dies beinhaltet eine Alarmfunktion, um bei auftretenden Fehlern sofort einschreiten zu können. So kann ein Verlust an physikalisch wichtigen Daten minimiert oder gegebenenfalls rekonstruiert werden. Ferner sollen Slow-Control-Systeme auch eine Hilfefunktion enthalten, um unerfahrenen Benutzern des Systems bei Alarmen zu helfen, richtig zu reagieren. Aus diesem Grund wird die Bedienung der Systeme auch möglichst einfach gehalten, damit im Falle eines Alarmes kein umständliches Nachschlagen in Bedienungsanleitungen notwendig ist.

Der Name „Slow"-Control rührt daher, daß diese Systeme sich mit Parametern befassen, die sich zeitlich normalerweise nur langsam verändern. Das Wort „langsam" ist hier zu verstehen als „langsam relativ zur Datenerfassung". In der Datenerfassung muß auf ein sehr exaktes Timing geachtet werden, um auf Trigger zu reagieren und die Daten mit einer möglichst geringen Totzeit des Detektors auszulesen. Ein Slow-Control-System dagegen befaßt sich mit Werten, die entweder zeitlich konstant sind (z.B. Versorgungsspannungen) oder nur langsam fluktuieren (z.B. Luftdruck). Im allgemeinen ist es deshalb unkritisch, ob diese Werte mit exaktem Timing ausgelesen werden oder nicht. Man nennt diese Methode ein asynchrones Auslesen, im Gegensatz zu der synchronen Auslese in der Datenerfassung.

Die Slow-Control und die Datenerfassung sind zwei völlig voneinander getrennte Systeme. Dies liegt vor allem daran, daß ein Slow-Control-System auch dann aktiviert ist, wenn die Datenerfassung abgeschaltet ist. Dies ist notwendig, um eine ständige Kontrolle über den Zustand des Experimentes zu haben. Prinzipiell sollte ein Slow-Control-System auch das erste System sein, das gestartet wird.

In einem Punkt weicht das CHORUS Slow-Control-System allerdings von dieser Philosophie ab. Wenigstens zu Anfang des Experimentes wird das System nur Werte überwachen, aber sie nicht kontrollieren, d.h. setzen. Dies ist eine Beschränkung, die aus reinen Zeitgründen gemacht wurde. Um ein System zu erstellen, das auch eine direkte Kontrolle ausübt, ist ein bedeutend größerer Zeitaufwand notwendig. Am CHORUS Slow-Control-System arbeiten nur 3 Personen, und zwei davon haben auch noch andere Aufgaben. Um die Größenordnung zu verstehen, sei hier nur angegeben, daß das Slow-Control-System des H1 Detektors am DESY mit einem Aufwand von „ nicht weniger als 8 Mannjahren“ [Gue] erstellt wurde. Die Zeit, die zum Aufbau des CHORUS Slow-Control-Systems zur Verfügung steht, beträgt jedoch nur weniger als ein Mannjahr.

4.2 Anforderungen

Ein Slow-Control-System muß mehreren Anforderungen genügen. Der erste und wichtigste Punkt ist, daß ein es verläßlich sein muß, damit seine Meldungen von den Benutzern des Systems beachtet werden. Ferner sollte es bei Inbetriebnahme der Computer automatisch initialisiert und gestartet werden, um eine ständige Kontrolle zu gewährleisten.

Die wohl wichtigste Funktion eines Slow-Control-Systems ist die Alarmfunktion. Falls ein Parameter außerhalb seiner festgelegten Grenzen liegt oder sich zu schnell ändert, so muß der Operator im Kontrollraum so bald wie möglich darauf aufmerksam gemacht werden. Dies erfordert eine gute visuelle und eventuell auch eine akustische Meldung. Die

Alarmmeldung soll dabei mitteilen, wo der Fehler aufgetreten ist und um welche Art von Fehler es sich dabei handelt. Eine Hilfefunktion ist wünschenswert.

Es muß möglich sein, einen Alarm für eine kurze Zeit zu maskieren, d.h. einen Alarm nicht anzuzeigen. Dies ist nötig, wenn z.B. ein Gerät gewartet wird. In diesem Fall sollte das System den Operator jedoch nach einer gewissen Zeitspanne fragen, ob die Maskierung weiter gültig bleiben soll, damit ein Kanal nicht versehentlich abgeschaltet bleibt. Zur Analyse der Zuverlässigkeit des Detektors sollte weiterhin auch eine Möglichkeit zur Speicherung und Analyse von Alarmen vorhanden sein.

Eine Speicherung und Darstellung der überwachten Kanäle ist eine weitere Funktion, die von einem Slow-Control-System erwartet wird. Diese Funktion, wie auch die Kontrolle und Analyse von Alarmen, soll über eine intuitiv verständliche Benutzerschnittstelle zugänglich sein, um auch unerfahrenen Benutzern eine effektive Nutzung des Systems zu ermöglichen.

Um all diese Aufgaben zu erfüllen, muß das System auch in der Lage sein, mit anderen Systemen (Strahlkontolle, Sicherheitssystem, Gassystem, Run-Kontrolle) zu kommunizieren. Falls es über Statusinformationen dieser Systeme verfügt, so kann beim Auftreten eines Ereignisses wie eines Gasalarms oder eines Not-Stops vermieden werden, den Benutzer mit einer großen Anzahl unnötiger Fehlermeldungen zu verwirren.

Es sollte ferner ohne große Schwierigkeiten möglich sein, neue zu überwachende Werte in das System zu integrieren. Dies bedeutet, daß neue Werte nur einmal und an einer Stelle eingegeben werden müssen, bevor sie dem gesamtem System bekannt sind. Ein aufwendiges erneuertes Kompilieren des Systems bei jeder kleinen Änderung soll dabei vermieden werden.

4.3 Konzeption

Als erstes wurde überprüft, ob ein kommerzielles System erhältlich ist, das die Anforderungen für das CHORUS Slow-Control-System erfüllt.

Es zeigt sich aber, daß solche Systeme zu aufwendig in ihrer Konzeptionierung und Wartung sind, um für das CHORUS-Experiment in Frage zu kommen. Außerdem arbeiten sie fast immer nur mit einer bestimmten Art von Hardware zusammen, was eine Anschaffung auch aus Kostengründen ausschloß.

Ferner wurden am CERN schon vorhandene Slow-Control-Systeme geprüft. Doch es wurde festgestellt, daß diese Systeme alle für einen bestimmten Detektor mit einer speziellen Hardware konzipiert wurden. Diese Systeme umzuschreiben und sie an den CHORUS-Detektor anzupassen hätte ähnlich viel Zeit erfordert, wie ein eigenes System zu entwickeln. Es gibt am CERN zwar eine Softwaregruppe, die versucht, ein Slow-Control-Standard-Programmpaket zu schreiben, aber es ist noch keine Software erhältlich. Hinzu kommt, daß dieses Programmpaket mehr für die in Zukunft am LHC (Large Hadron Collider) geplanten Experimente konzipiert ist, also in seinem Umfang zu aufwendig ist für ein vergleichsweise kleines Experiment wie CHORUS.

Aus diesen Gründen wurde entschieden, ein eigenes System zu entwickeln und zu programmieren.

Um ein verläßliches System zu erhalten, wurde beschlossen, die Slow-Control-Computer an das Notstromsystem anzuschließen, damit sie während eines Stromausfalls und vor allem nach dem Neustart des Stroms ihre Aufgabe erfüllen können. Die vom System benötigte Software ist dabei auf einem lokalen Server gespeichert. Der Server ist ebenso wie das zur Kommunikation notwendige lokale Ethernetsegment an die Notstromversorgung angeschlossen.

Insgesamt wurde angestrebt, das Slow-Control-System in seiner Hardware und Software so weit wie möglich kompatibel zu den Systemen der Datenerfassung zu halten. Durch diese Maßnahme vereinfachen sich viele Arbeitsvorgänge, wie z.B. die Installierung von Betriebssystem, Compiler und anderen Dienstprogrammen. Kommunikationsprogramme zum Austausch von Daten zwischen den CPUs müssen nur einmal geschrieben werden und können sowohl von der Datenerfassung als auch von der Slow-Control benutzt werden.

Damit das System variabel bleibt, werden alle zum Lesen und Darstellen eines Kanales notwendigen Informationen in Konfigurationsdateien gespeichert. Diese Dateien sind reine Textdateien. Bei dem Start des Programmes werden sie eingelesen und in die von den Programmen benötigten Formate übersetzt. Wenn ein neuer Kanal überwacht werden soll, so wird er in eine der Konfigurationsdateien eingesetzt. Dann kann man das System neu initialisieren, um den Kanal hinzuzufügen. Dadurch kann vermieden werden, das System bei jedem neuen Wert erneut zu kompilieren. Unter anderem läßt sich so auch die Anzahl der unnötigen Fehler verringern.

Wenn ein Alarm auftritt, so soll das Slow-Control-System dem Benutzer alle notwendigen Informationen über den fehlerhaften Parameter anzeigen, damit eine schnelle Reparatur und damit ein möglichst geringer Datenverlust gewährleistet wird. Dazu gehören Art des Alarms, augenblicklicher Wert des Parameters, Funktion der fehlerhaften Hardware, Position der Hardware und die Kontaktperson für den Fall, daß keine sofortige Reparatur möglich ist. Damit diese Informationen auch für einen unerfahrenen Benutzer schnell und einfach zugänglich sind, soll vermieden werden, daß zuerst in einem Handbuch die genaue Syntax eines Befehls nachgeschlagen werden muß. Deshalb wird eine graphische Benutzeroberfläche erstellt, d.h. daß alle Funktionen des Systems über eine Menüsteuerung mit Mausklicks ausgeführt werden können.

4.4 Konfigurationsdateien

Der wohl wichtigste Punkt im Entwurf des CHORUS Slow-Control-Systems besteht in der Struktur der Konfigurationsdateien, da alle anderen Programme diese Daten als Basis für ihre Operation benutzen. Wenn später Änderungen in den Konfigurationsdateien nötig sein sollten, so müssen fast alle anderen Programme umgeschrieben werden.

Um Platz zu sparen, soll darauf geachtet werden, daß die Konfigurationsdateien keine überflüssigen Daten beinhalten. Z.B. muß in einer Konfigurationsdatei für Kanäle in einem G64 System kein Speicher reserviert werden, der Parameter für die Auslese eines VME Systems enthält, da diese Informationen nicht benötigt werden, um einen Wert in einem G64 System zu lesen. Weiterhin sollten auch Parameter, die für eine große Anzahl von Kanälen gelten, nur einmal angegeben werden, anstatt für jeden Kanal einzeln aufgeführt zu werden.

Das im folgenden aufgeführte Design von Konfigurationsdateien, sowie die Programme für ihre Erstellung und Wartung, sind das Endprodukt einer langen Reihe von Entwürfen und Verbesserungen. Die Hilfsprogramme nehmen dem Anwender viele Aufgaben bei der Erstellung der Dateien ab. Durch interaktive Dialoge werden dabei die Fehlermöglichkeiten minimiert.

Die Konfigurationsdateien sind reine Textdateien, wodurch eine visuelle Kontrolle durch den Anwender erleichtert. Für das Slow-Control-System werden sie dann von einem besonderem Programm in maschinenlesbare Strukturen übersetzt.

Jede Konfigurationsdatei enthält nur Daten zu Kanälen, die mit derselben Auslesemethode gemessen werden. Die Konfigurationsdateien selbst bestehen aus einem Kopf mit allgemeinen Informationen und einer Liste mit Details über die einzelnen Kanäle. Im Dateikopf stehen die Parameter, die für alle Kanäle in der Datei gelten. Zuerst kommen allgemeinen Parameter wie der Name der Konfigurationsdatei, die Auslesemethode, Informationen über Alarm-, Speicher- und Darstellungsmodi für die in der Konfigurationsdatei definierten Kanäle. Danach folgen die für die Auslesemethode spezifischen Parameter, z.B. eine gemeinsame Basisadresse eines elektronischen Auslesemoduls oder ein Adreßmodifikator für eine spezifische Hardware. Im zweiten Teil folgen dann die Parameter, die sich für jeden Kanal ändern. Dies sind z.B. der Name eines Kanales, das untere und das obere Limit, zwischen denen der Wert des Kanals sich befinden darf, oder der Offset zur Basisadresse im Auslesmodul. Die Struktur und Zusammensetzung dieser Parameter wird dabei von der Auslesemethode festgelegt.

Alle in den Konfigurationsdateien enthaltenen Parameter sind dabei stabil, d.h. sie sollten prinzipiell für den gesamten Verlauf des Experimentes gelten. Falls sich Werte häufig ändern, wie z.B. die Sollspannungen der Photomultiplier im Triggersystem, so wird dies im Kopf der Konfigurationsdatei angezeigt, und die Werte werden dann durch einen Befehl von einer speziellen Datei eingelesen. Diese Datei wird von dem entsprechenden System, in diesem Fall also dem Trigger-System, verwaltet. Wenn Änderungen vorliegen, so wird dem Slow-Control-System durch eine Nachricht mitgeteilt, die neuen Werte zu benutzen.

In einer Masterdatei werden alle Konfigurationsdateien augelistet, die im Slow-Control System verwendet werden. Außerdem enthält die Masterdatei für jede Konfigurationsdatei das Datum der letzten Änderung sowie eine Flag. Diese Flag gibt Auskunft, ob die Konfigurationsdatei aktiv ist oder nicht. Diese Option ist nützlich bei Reparaturen von ganzen Hardwaregruppen.

4.4.1 Namen und Indizes

Vor den Namen jeder Konfigurationsdatei wird ein eindeutiger Index aus zwei Buchstaben gesetzt. Dieser Index erleichtert die Unterscheidung der verschiedenen Konfigurationsdateien und charakterisiert später alle Kanäle, die in der Konfigurationsdatei definiert werden.

Jeder Kanal, der im Slow-Control-System überwacht wird, muß einen eindeutigen Namen haben. Dies ist notwendig, da die Konfigurationsdateien sich ändern können und das System bei der Initialisierung den Kanälen automatisch Indizes zuteilt. Deshalb ist der Name die einzige Möglichkeit, einen Kanal eindeutig zu identifizieren. Um die Eingabe von Namen zu vereinfachen, wird deshalb der Index der Konfigurationsdatei vor jeden Kanalnamen in der Datei gestellt. So muß bei der Erstellung einer neuen Konfigurationsdatei nicht geprüft werden, ob der Name eines Kanals schon im System existiert. Ferner sollte der Name eines Kanales auch etwas über den Wert aussagen, der in dem Kanal überwacht wird.

Der Name einer Konfigurationsdatei, in der die Kanäle des Gassystems definiert sind, könnte z.B. `aagas` lauten. Typische Namen in dieser Konfigurationsdatei wären `aaArPres` oder `aaIBPres`. Diese Namen stehen dann für `Ar`gon-`Pres`sure (Druck) und `I`so`B`utan-`Pres`sure.

4.4.2 Transfermodus und Bearbeitungsmodi

Damit das Slow-Control-Programm weiß, wie es einen Kanal auslesen soll, wird ein Transfermodus definiert. Dabei gehört jeder Transfermodus zu einer Auslesemethode. Ein Transfermodus besteht aus einem

Buchstaben, gefolgt von einer Zahl, z.B. `c1`. Dabei steht der Buchstabe für die verwendete Hardware (`c` CAMAC, `v` VME, `g` G64 usw.). Die Zahl steht für unterschiedliche Methoden (z.B. unterschiedliche VME Module).

Weiter gibt es für jede Konfigurationsdatei einen Alarm- und Speichermodus. Diese Information besteht aus zwei Ziffern: 10 bedeutet z.B. Alarme, aber keine Speicherung der Werte; 11 hieße sowohl Alarme als auch Speicherung. Die Verwendung der Ziffern 0 und 1 ist eine Konvention, analog zum logischen Aus und Ein. Außerdem kann dieses Konzept leicht erweitert werden, falls später weiter Modi erforderlich werden.

Es kann angegeben werden, wie oft der Wert eines Kanales gelesen werden soll. Falls Speicherung gewünscht wird, so wird auch noch ein Speicherungsintervall definiert. Hier kann bestimmt werden, wie oft der Wert in eine Datenbank eingegeben werden soll, z.B. einmal pro Stunde. Außerdem müssen in diesem Fall auch noch die Darstellungsoptionen angegeben werden.

Diese Parameter stehen im Kopf der Konfigurationsdatei, sie sind also für alle Kanäle in der Datei gleich.

4.4.3 Individuelle Parameter

Für jeden Kanal wird der Name eingegeben, sowie die Ober- und die Untergrenze, zwischen denen sich der gemessene Wert bewegen darf. Außerdem wird ein Grenzwert definiert, um den sich ein Wert prozentual zwischen zwei Auslesungen ändern darf. Ein Skalierungsfaktor gibt an, wie der von der Hardware gelesene Wert mit dem realen Wert des Kanals in Beziehung steht. Dazu muß auch die Einheit angegeben werden, in der der Wert gemessen wird. Diese Parameter werden für jeden Kanal benötigt.

Der Rest der individuellen Parameter wird normalerweise durch die Auslesemethode bestimmt. Sie geben an, mit welchen Parametern die Hardware angesprochen werden muß, um den Wert des Kanals auszulesen.

4.5 Formatdateien und Auslesemethoden

In einer Formatdatei werden Auslesemethoden definiert. Dabei kann eine Formatdatei die Angaben zu mehreren Auslesemethoden enthalten.

Eine Formatdatei besteht aus zwei Teilen. Im ersten Teil werden mit einem Textstring pro Zeile alle Parameter angegeben, die in einer Auslesemethode benötigt werden können. Auf die Beschreibung jedes Parameters folgt in Klammern der Formatspecifier, mit dem der Parameter in der Programmiersprache C ausgelesen wird. Der erste Teil wird mit dem String `"-101010"` abgeschlossen. Im zweiten Teil folgen dann die Definitionen der Auslesemethoden. Jede Definition besteht dabei aus zwei Strings. Der erste String gibt den Namen der Methode an, der zweite enthält eine „character" (Buchstaben) pro Parameter, der im ersten Teil aufgeführt wird. Dieser Character gibt jeweils an, wie der entsprechende Parameter mit der Auslesemethode in Verbindung steht. Er kann die Werte `'0'`, `'1'`, `'2'` oder `'3'` annehmen.

Eine `'0'` bedeutet, daß der entsprechende Parameter in dieser Auslesemethode verwendet wird. Später wird er von dem Programm, mit dem die Konfigurationsdateien erstellt werden, ignoriert. Parameter, die einen Wert >`0` haben, werden für die Auslesemethode benötigt. Das Konfigurationsprogramm fragt dann zuerst die Parameter ab, die den Wert `'1'` haben. Diese Parameter werden nur einmal abgefragt und dann in den Kopf einer Konfigurationsdatei gesetzt. Sie gelten später also für alle Kanäle, die in der Konfigurationsdatei mit der entsprechenden Auslesemethode definiert werden. Auch Parameter mit dem Wert `'2'` werden in den Dateikopf einer Konfigurationsdatei gesetzt. Sie gelten also auch für alle Kanäle, die in der Konfigurationsdatei enthalten sind. Die mit `'2'` bezeichneten Parameter werden jedoch hinter die Parameter mit dem Wert `'1'` gestellt. Diese Unterscheidung wird vorgenommen, um Parmeter, die in jeder Auslesemethode vorkommen, von den Parametern zu unterscheiden, die spezifisch für eine besondere Methode sind. Die vierte erlaubte Character ist eine `'3'`. Parameter dieser Art werden für jeden Kanal in einer Konfigurationsdatei erneut abgefragt. Sie bilden also den Listenteil einer Konfigurationsdatei.

Da die verschiedenen Konfigurationsdateien wissen, mit welcher Methode in einer Formatdatei sie erstellt wurden, dürfen Auslesemethoden nicht geändert werden. Falls es notwendig ist, eine Auslesemethode für eine neue Konfigurationsdatei zu modifizieren, so besteht die Möglichkeit, eine geänderte Version der Auslesemethode mit einem neuen Namen an das Ende der Formatdatei anzufügen.

4.6 Hilfsprogramme zur Erstellung von Konfigurationsdateien

Eine Reihe von Hilfsprogrammen wurde erstellt, um den Umgang mit Konfigurationsdateien zu erleichtern und die Fehleranfälligkeit zu vermindern.

4.6.1 `formprog`

Eine Formatdatei wird mit einem Texteditor erstellt. Pro Zeile wird ein Textstring eingegeben, der einen Parameter für eine Auslesemethode entspricht. Danach folgt jeweils ein Buchstabe in Klammern. Dieser Buchstabe ist der Format-Specifier, den die Programmiersprache C benötigt, um den Wert in eine Variable einzulesen. Da eine Formatdatei mehrere Auslesemethoden enthalten kann, werden mehr Parameter eingegeben, als für eine einzige Auslesemethode benötigt werden. Danach wird die Formatdatei mit dem String `"-101010"` beendet.

Sobald eine Formatdatei existiert, kann das Programm `formprog` benutzt werden, um Auslesemethoden zu definieren. Eine Aulesemethode besteht dabei aus einem Namen und einem String, in dem jedem Parameter in der Formatdatei ein Character zugeordnet wird. Nach dem Start fragt `formprog`, ob eine neue Methode erstellt oder eine alte Methode abgeändert werden soll. Danach geht es die in der Formatdatei definierten Parameter Zeile für Zeile durch. Falls eine Auslesemethode modifiziert wird, so wird der alte Wert eines Parameters als Defaultwert angezeigt. Mit der Taste <`CR`> wird der Defaultwert übernommen. Ansonsten wird aus einem Menü der neue Wert des Parameters ausgewählt. Dabei sind die Tasten <`0`>, <`1`>, <`2`> und<`3`> den in der

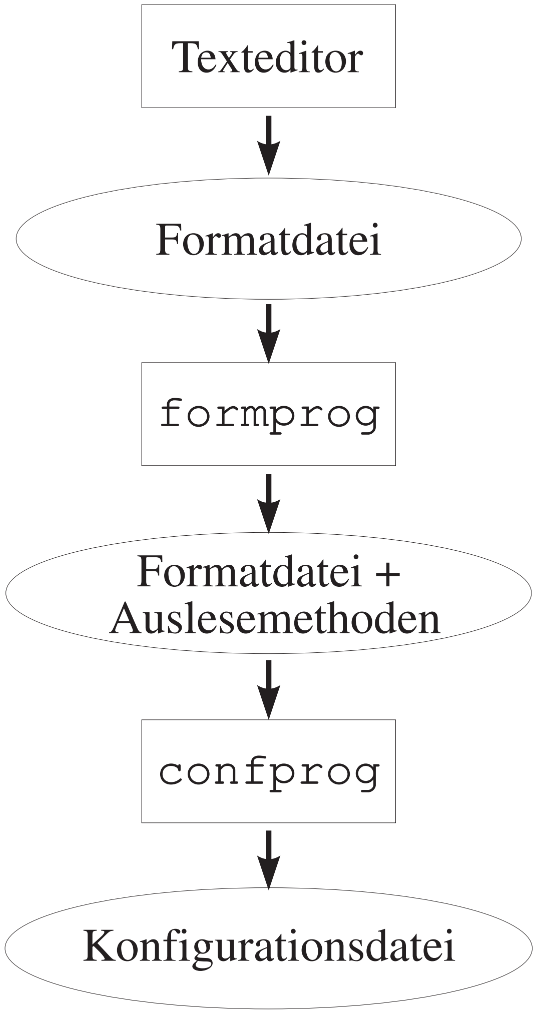

Abbildung 4.1: Die Erstellung einer Konfigurationsdatei

vorigen Sektion erklärten Möglichkeiten zugeordnet.

Nachdem das Programm alle Parameter abgefragt hat, kann die neue Auslesemethode abgespeichert werden oder es kann eine Korrektur vorgenommen werden. Im Falle einer Korrektur werden alle Parameter nocheinmal mit ihren vorher definierten Werten als Defaultwert durchgegangen. Nach dem Abspeichern fügt das Programm die neue Auslesemethode an das Ende der Formatdatei an.

Das Programm `formprog` fügt also Auslesemethoden an eine schon bestehende Formatdatei an (Abb. 4.1).

Da eine Formatdatei eine Textdatei ist, könnten neue Auslesemethoden auch mit dem Texteditor eingegeben werden. Das Programm `formprog` wurde geschrieben, um die umständliche Eingabe eines Strings mit mehr als 40 Characters zu ersetzen und damit auch die Fehlermöglichkeiten zu reduzieren.

Es folgt ein Beispiel für eine Formatdatei `formats1.def`, in der eine Auslesemethode namens `g64am` definiert wurde. Mit dieser Auslesemethode werden niedrige Spannungen ($< 100\ V$) über einen Analogen Multiplexer in einem G64 Crate gemessen. Hauptsächlich sind dies Versorgungsspannungen der verschiedenen Elektronik-Crates.

```
Date code for last modification (d)
Parameter subset monitored (s)
Number of entries in header record (d)
Number of entries in first part of header (d)
Number of entries in each parameter record (d)
Number of parameter records (d)
Formats file name (s)
Format identifier (s)
Generic file name (s)
Alarm and storage mode (s)
Read interval in mns (d)
Regular storage interval in mns (d)
Beginning of presentation path (s)
Transfer mode (s)
Name of the parameter (s)
Signal source identification (s)
Unit of parameter (s)
Scale factor to get unit from reading (f)
Lowlimit in units (f)
Highlimit in units (f)
Changelimit in percent (f)
Transfer submethod (s)
CAMAC branch for control (d)
CAMAC crate for control (d)
CAMAC slot for control (d)
CAMAC subaddress for control (d)
CAMAC branch for data transfer (d)
CAMAC crate for data transfer (d)
CAMAC slot for data transfer (d)
CAMAC subaddress for data transfer (d)
Lecroy mainframe number (d)
VME module name for control (d)
VME baseaddress for control (d)
```

```
VME offset for control (d)
VME module name for data transfer (d)
VME baseaddress for data transfer (d)
VME offset for data transfer (d)
CAEN crate number (d)
CAEN subaddress (d)
G64 module number (d)
G64 MUX station*10+card number (d)
G64 MUX board screw number 1 to 32 (d)
Channel number for control (d)
Channel number for data transfer (d)
Free parameter number 1 (d)
Free parameter number 2 (d)
Free parameter number 3 (d)
Free parameter number 4 (d)
-101010
g64am
11111111111111331333100000000000000000033000000
```

Die Formatspecifier `d`, `f` und `s` stehen dabei für jeweils einen Integer (ganzzahligen) Wert, eine Gleitkommazahl oder einen String. Die Parameter, die für die Auslesemethode `g64am` ausgewählt wurden, werden im nächsten Abschnitt anhand einer Konfigurationsdatei erklärt.

4.6.2 `confprog`

Wenn eine Formatdatei existiert, so kann mit dem Programm `confprog` eine Konfigurationsdatei erstellt oder modifiziert werden. Es bietet die Möglichkeit, eine bestehende Konfigurationsdatei auszudrucken, Parameter in ihr zu ändern oder neue Kanäle hinzuzufügen. Außerdem erleichtert es den Transfer von Daten in ein neues Format.

Das Programm erwartet den Namen einer Konfigurationsdatei als Argument. Für den Fall, daß eine neue Konfigurationsdatei angelegt werden soll, wird in der Datei `allconfig.gname` überprüft, ob der Index aus zwei Buchstaben bereits in einem anderen Namen verwendet wurde. Falls der Index schon existiert, muß ein neuer Index eingegeben werden. Anderenfalls wird der Name zu der Datei `allconfig.gname` hinzugefügt, allerdings erst nachdem die neue Konfigurationsdatei gespeichert wurde (Abb. 4.1).

Zuerst fragt das Programm nach dem Namen einer Formatdatei. Danach muß der Name einer Auslesemethode eingegeben werden. Mit Hilfe des Strings, in dem die Auslesemethode definiert wurde, werden nun alle mit `'1'` markierten Parameter abgefragt. Die Antworten werden in eine Textdatei geschrieben, mit je einer Antwort pro Zeile. Danach werden alle Parameter mit `'2'` abgefragt und abgespeichert. Schließlich beginnt das Programm eine Schleife, in der alle mit `'3'` bezeichneten Parameter abgefragt werden. Diese Schleife wird so lange durchlaufen, bis alle Kanäle der Konfigurationsdatei eingegeben wurden. Auch in diesem Programm besteht eine Möglichkeit zur Korrektur der eingegebenen Werte.

Mit Hilfe der Formatdatei `formats1.def` aus dem vorherigen Beispiel wurde eine Konfigurationsdatei `gamux.config` erstellt. Da in einer Konfigurationsdatei nur die Werte der einzelnen Parameter abgespeichert sind, wird eine kommentierte Auflistung der Datei `gamux.config` angegeben, die mit dem Programm `confprog` erstellt wurde-

```
Commented listing of gamux.config generated with format g64am
in formats1.def:
----Listing of the header record
 1:          Date code for last modification (d)  932011443
```

In dem Datums-Code wird die Anzahl der Minuten seit dem Beginn des Jahres 1993 angegeben. Diese Art eines Datums-Code wird verwendet, da für die Darstellung von gespeicherten Werten eine lineare Zeitskala benötigt wird.

```
 2:               Parameter subset monitored (s)  Prototype
                                                   for G64 MUX
                                                   monitoring
```

Dieser Kommentar zeigt an, welche Art von Kanälen in der Konfigurationsdatei definiert wird.

```
 3:          #  of entries in header record (d)  16
 4:    # of entries in first part of header (d)  16
 5:   # of entries in each parameter record (d)  7
```

Der erste Parameter gibt die Anzahl der Zeilen im Kopf der Konfigurationsdatei an. Es ist die Anzahl aller in der Auslesemethode mit '1' und '2' bezeichneten Parameter. In der vierten Zeile wird angezeigt, wie viele Parameter vom Typ '1' im Dateikopf stehen. Da in der Auslesemethode `gamux` keine Parameter vom Typ '2' definiert wurden ist dieser Wert mit dem vorhergehenden Wert identisch. In Zeile 7 wird dann die Anzahl der Einträge angegeben, die für jeden Kanal erneut gemacht werden.

```
 6:             Number of parameter records (d)  4
```

Die Anzahl der in der Konfigurationsdatei definierten Kanäle.

```
 7:                       Formats file name (s)  formats1.def
 8:                       Format identifier (s)  g64am
```

Der Name der Konfigurationsdatei und der Auslesmethode, mit der die Konfigurationsdatei erstellt wurde.

```
 9:                       Generic file name (s)  gamux
```

Der Name der Konfigurationsdatei.

```
10:                  Alarm and storage mode (s)  10
```

Alarm- und Speichermodus für die in dieser Datei definierten Kanäle (1-Alarme, 0-Keine Speicherung).

```
11:                    Read interval in mns (d)  5
```

Angabe, wie oft die Kanäle überprüft werden sollen.

```
12:         Regular storage interval in mns (d)  0
```

Angabe, wie oft der Wert der Kanäle abgespeichert werden soll. In diesem Beispiel wurde keine Speicherung verlangt.

```
13:          Beginning of presentation path (s)  prototype/test
```

Hier wird der Menüpfad angegeben, über den man später mit der Benutzeroberfläche die Werte der Kanäle abrufen kann. Das hier definierte Menü wird die Namen aller in der Datei definierten Kanäle als Eintragungen haben. Für die Speicherung von Werten in einer Datenbank wird die gleiche Menüstruktur verwendet.

```
14:                              Transfer mode (s)  g1
```

Der Transfermodus, der die Auslesemethode für das Programm zur Datenauslese definiert.

```
15:                          Unit of parameter (s)  volt
```

Die Einheit, in der der Wert eines Kanales ausgegeben wird.

```
16:                     Changelimit in percent (f)  10.0
```

Die Änderung in Prozent, um die sich der Wert eines Kanales zwischen zwei Messungen höchstens ändern darf.

Hier endet der Kopf der Konfigurationdatei. Im folgenden Teil werden die Parameter für die einzelnen Kanäle aufgeführt. Die Parameter werden am Beispiel des ersten Kanales erklärt.

```
----Listing of param records. 4 records in total
  1  1:                   Name of the parameter (s)  gac5p01001
```

Die ersten beiden Buchstaben des Namens sind der Index der Konfigurationsdatei, `ga`. Danach folgt ein Buchstabe, der die Art des überwachten Crates anzeigt (C-CAEN, N-NIMH). Danach folgen der Sollwert der Spannung (hier $5\,V$) und die Polarität der Spannung. Die letzten 5 Ziffern (01001) setzen sich aus den in Zeile 6 und 7 definierten Werten zusammen. Diese Werte bestimmen die Adresse des Kanals in dem Multiplexer-System. Die ersten 3 Ziffern enthalten den Wert aus Zeile 1-6, zusammen mit einer führenden 0 (010), und in den letzten beiden Ziffern steht der Wert aus Zeile 1-7, auch mit einer führenden 0 (01).

```
  1  2:           Signal source identification (s)  101A
```

Die Zahl gibt die Nummer des Elektonik-Racks an, in dem sich das zu überwachende Crate befindet. Der Buchstabe markiert die Lage des Crates innerhalb des Racks.

```
  1  3: Scale factor to get unit from reading (f)  1.0
```

Der Multiplexer kann nur positive Spannungen zwischen 0 und $6\,V$ messen. Der gemessene Wert wird dann mit diesem Skalierungsfaktor multipliziert, um den realen Wert des Kanales zu erhalten.

```
1  4:                      Lowlimit in units (f)  4.8
1  5:                     Highlimit in units (f)  5.2
```

Dies sind die Grenzwerte, zwischen denen sich der Wert des Kanales bewegen darf ohne einen Alarm auszulösen. Diese Werte sind die realen Werte ohne Skalierung.

```
1  6:         G64 MUX station*10+card number (d)  10
1  7:     G64 MUX board screw number 1 to 32 (d)  1
```

Diese Werte geben an, an welcher Adresse der Kanal an den Multiplexer angeschlossen ist. Die Adresse dieses Kanales lautet: Multiplexer Station 1, Terminalboard 0, Schraube 1. Insgesamt können bis zu 32 Stationen an den Multiplexer angeschlossen werden. Dabei hat jede Station 4 Terminalboards (0-3), und jedes Terminalboard 32 Anschlußschrauben.

```
2  1:                  Name of the parameter (s)  gac5n01101
2  2:           Signal source identification (s)  101A
2  3: Scale factor to get unit from reading (f)  -1.0
2  4:                      Lowlimit in units (f)  -5.2
2  5:                     Highlimit in units (f)  -4.8
2  6:         G64 MUX station*10+card number (d)  11
2  7:     G64 MUX board screw number 1 to 32 (d)  1
3  1:                  Name of the parameter (s)  gan24p01001
3  2:           Signal source identification (s)  102B
3  3: Scale factor to get unit from reading (f)  4.0
3  4:                      Lowlimit in units (f)  22.8
3  5:                     Highlimit in units (f)  25.2
3  6:         G64 MUX station*10+card number (d)  10
3  7:     G64 MUX board screw number 1 to 32 (d)  2
4  1:                  Name of the parameter (s)  gan24n01102
4  2:           Signal source identification (s)  102B
4  3: Scale factor to get unit from reading (f)  -4.0
4  4:                      Lowlimit in units (f)  -25.2
4  5:                     Highlimit in units (f)  -22.8
4  6:         G64 MUX station*10+card number (d)  11
4  7:     G64 MUX board screw number 1 to 32 (d)  2
```

4.6.3 `regrouprog`

Die Konfigurationsdateien sind Textdateien, die für eine einfache Benutzung optimiert wurden. Sie enthalten alle Informationen über die verschiedenen Kanäle des Slow-Control-Systems. Damit diese Informationen von den verschiedenen Programmen des Systems gelesen werden können, müssen sie erst in eine geeignete Struktur übersetzt werden.

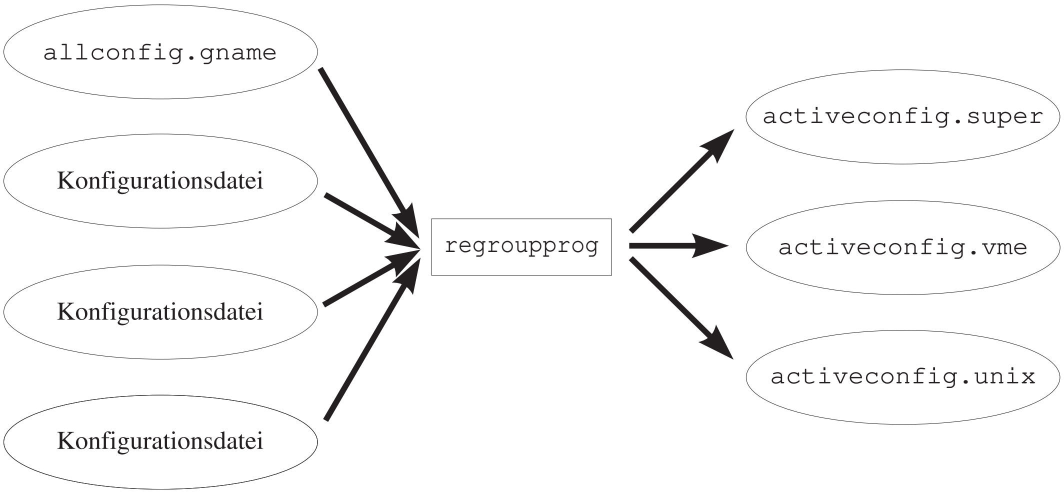

Abbildung 4.2: Das Programm `regroupprog`

Dabei kann auf alle Informationen verzichtet werden, die ein bestimmtes Programm nicht benötigt. Z.B. braucht das Darstellungsprogramm nichts über die Ausleseparameter eines Kanals zu wissen.

Das Programm `regrouprog` liest eine Textdatei ein. In dieser Datei befinden sich die Namen der Konfigurationsdateien, deren Information strukturiert werden soll. Die letzte Zeile der Datei muß den String `"-101010"` enthalten. Als Standard wird dabei die Datei `allconfig.gname` gelesen. Als Option kann man angeben, welche Konfigurationsdateien nicht verarbeitet werden sollen. Die restlichen angegebenen Konfigurationsdateien werden in alphabetischer Reihenfolge abgearbeitet.

Es werden drei Ausgabedateien erzeugt, die alle den ersten Teil des Namens gemeinsam haben. Dazu wird der Name der Eingabedatei ohne Erweiterung benutzt. Der Standardname ist `activeconfig` (Abb. 4.2). Folgenden Dateien werden erzeugt:

- name.`super`
  Diese Datei enthält die Namen aller Konfigurationsdateien, die vom Slow-Control-System bearbeitet werden. Hinter jedem Namen befindet sich der Datums-Code, der angibt, wann die Datei zuletzt geändert wurde, sowie ein Flag, der anzeigt, ob die Konfigurationsdatei aktiv ist oder nicht. Mit Hilfe dieses Flags kann man eine ganze Konfigurationsdatei ein- oder ausschalten. Die

letzte Zeile enthält den String `"-101010"`.

- name.`unix`
  In diese Datei werden alle Informationen geschrieben, die zur Darstellung und Analyse der Kanäle benutzt werden.

- name.`vme`
  Hier sind alle Parameter aufgeführt, die zur Auslese der Kanäle benötigt werden.

Wenn die Datei `allconfig.gname` nur den Namen `gamux.config` enthält, so werden die folgenden Dateien erzeugt:

`activeconfig.super`:

```
gamux 932011443 1
-101010
```

`activeconfig.unix`:

```
gamux 932011443 10 4
Prototype for G64 MUX monitoring
1 gac5p01001 101A volt 4.8 5.2 prototype/test
2 gac5n01102 101A volt -5.20 -4.80 prototype/test
3 gan24p01001 102B volt 22.8 25.2 prototype/test
4 gan24n00202 102B volt -25.20 -22.80 prototype/test
```

In der ersten Kopfzeile wird zuerst der Name der Konfigurationsdatei angegeben. Es folgen der Datums-Code der letzten Änderung, der Alarm- und Speichermodus der Kanäle, gefolgt von der Anzahl der Kanäle in der Konfigurationsdatei. In der zweiten Zeile wird dann beschrieben, welche Art von Kanälen in der Datei definiert wurde. Darauf folgt eine Zeile pro Kanal. Die Zeilen sind numeriert. Sie enthalten jeweils dem Namen des Kanals, den Ursprungsort des Wertes, die Einheit des Wertes und die Grenzwerte des Bereiches, in dem sich der Wert bewegen darf. Außerdem wird noch das Menü angegeben, über das der Wert betrachtet werden kann und in dem er gespeichert werden soll, falls dies verlangt wird. (Der Darstellungsbaum und der Menübaum für die Speicherung verwenden die gleichen Namen.) Diese Parameter werden benötigt, um Alarme anzuzeigen und Diagramme darzustellen.

`activeconfig.vme`:

```
gamux 932011443 10 g1 5 0 8 4
1 gac5p01001 1.0 4.8 5.2 10.0 10 1
2 gac5n01102 -1.0 -5.20 -4.80 10.0 11 1
3 gan24p01001 4.0 22.8 25.2 10.0 10 2
4 gan24n00202 -4.0 -25.20 -22.80 10.0 11 2
```

In der Datei `activeconfig.vme` enthält die Kopfzeile für eine Konfigurationsdatei deren Namen, den Datums-Code der letzten Änderung, den Alarm- und Speichermodus, die Transfermethode, das Leseintervall, das Speicherintervall, die Anzahl der Einträge pro Kanal sowie die Anzahl der Kanäle. Die Zeilen für die einzelnen Kanäle sind auch hier numeriert. Danach kommt der Name des Kanals, sein Skalierungsfaktor, die Unter- und die Obergrenze für den Wert des Kanales sowie die maximal erlaubte Änderung des Wertes in Prozent. Danach folgen zwei durch die Auslesemethode gegebene Parameter. Hier sind es die Adressen für den analogen Multiplexer. Da sich die Anzahl der spezifischen Parameter je nach Auslesemethode ändert, muß deshalb die Anzahl der Einträge pro Kanal in der Kopfzeile definiert werden, damit die Parameter von den Programmen richtig gelesen werden können.

4.6.4 `checkprog`

Dieses Programm erlaubt es, die Dateien name.`unix` und name.`vme` auf ihre Maschinenlesbarkeit zu überprüfen. Außerdem können mit seiner Hilfe die Informationen in diesen Dateien ausgedruckt werden. Es besteht die Möglichkeit, entweder die ganze Datei, die Informationen über eine bestimmte Konfigurationsdatei oder die Parameter eines Kanals auszudrucken.

4.6.5 `suprog`

Mit Hilfe dieses Programmes können die Aktivierungs-Flags in der Datei name.`super` geändert werden.

4.7 Hardware

Das Slow-Control-System läuft auf zwei Hardware-Plattformen. Die Konfigurations- sowie Darstellungprogramme laufen auf einer IBM RS6000 Workstation, die mit zwei 19 Zoll Monitoren ausgestattet ist. Sämtliche Sofware liegt auf dem CHORUS Online-Server, der ebenfalls eine RS6000 Maschine ist. Die Programme zur Steuerung der Meßhardware laufen auf einem Motorola 68040 CPU, das in einem VME Crate installiert ist. Das VME Datenbus System wurde als Standard für die CHORUS Datenerfassung gewählt. Aus Kostengründen wird in dem Slow-Control-System auch noch ein Crate mit dem G64 Bus System verwendet, das vom Vorgängerexperiment CHARM II übernommen wurde. Aus Standardisierungsgründen wird dieses Crate jedoch mit einem Interface zu einem VME System ausgestattet. Dieses Interface wurde für das ALEPH Slow-Control-System entwickelt. Es besteht aus einer kleinen Box mit 3 VME Einschüben, die mit einer Prozessor- und einer Ethernetkarte ausgestattet ist. Der Prozessor ist ein Motorola 68020 CPU. Somit kann auch dieses System wie ein VME System behandelt werden. Sämtliche CPUs sind über Ethernet miteinander verbunden. Auch die Kommunikation zu Maschinen außerhalb des Slow-Control-Systems erfolgt über Ethernet.

4.8 Hilfssoftware

Im gesamten Slow-Control-System wird die gleiche Software genutzt wie in der Datenerfassung. Dies geschieht hauptsächlich aus Gründen der Einfachheit. Software muß nur einmal installiert werden und die Resourcen können gemeinsam genutzt werden. Außerdem müssen Aufgaben, die sowohl in der Slow-Control als auch in der Datenerfassung anfallen, nur einmal programmiert werden.

4.8.1 Betriebssysteme

Es werden zwei Betriebssysteme verwendet. Auf den IBM RS6000 Computern wird AIX benutzt, das IBM-eigene UNIX Derivat. UNIX ist ein ein komplexes, weit verbreitetes Multiuser-Multitasking-System.

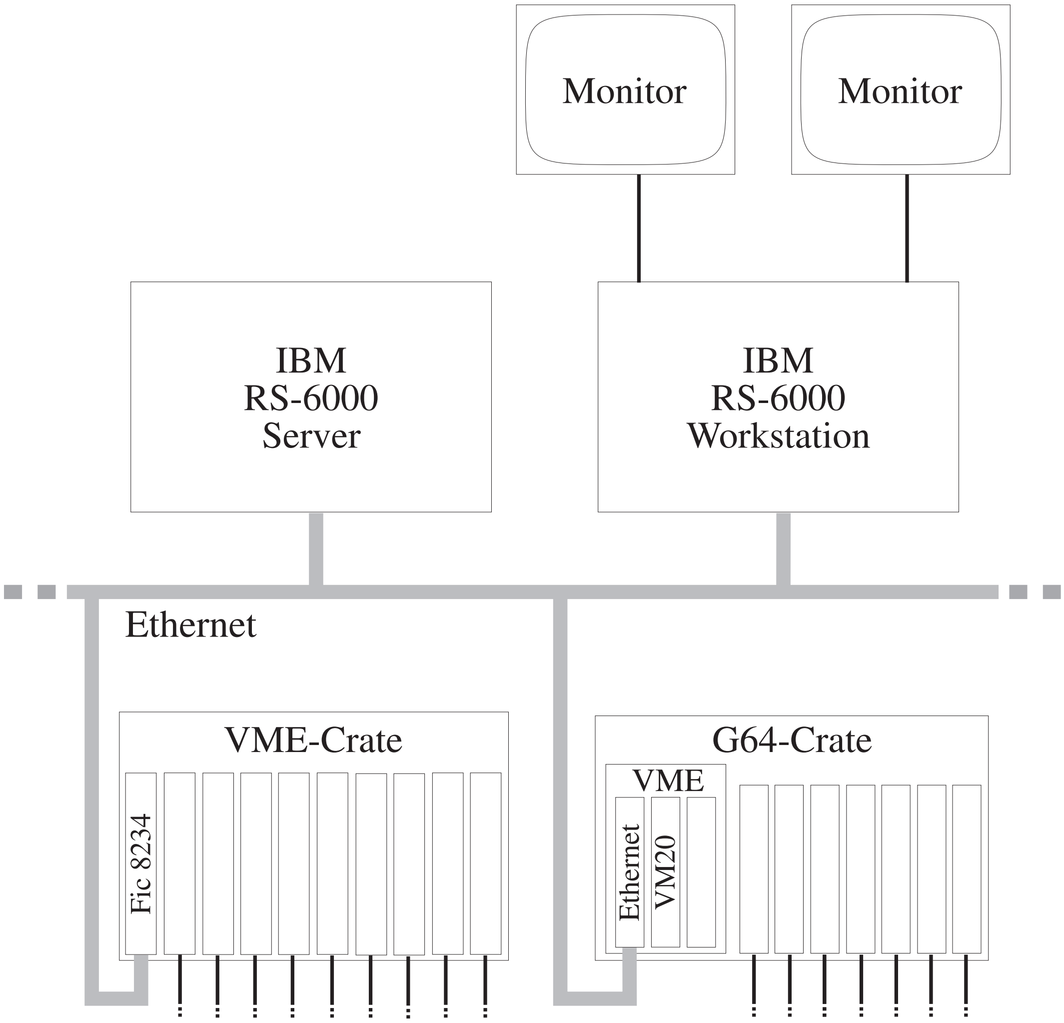

Abbildung 4.3: Das Slow-Control Hardware-System

Eine große Menge an Software ist für UNIX verfügbar. Unter anderem sind alle CERN Libraries erhältlich, sowie eine große Auswahl von „Public Domain“ (kostenlosen) Programmen. Aufgrund seiner Komplexität und des Multitasking-Systems ist UNIX nur in begrenztem Umfang für Echtzeitanwendungen geeignet. Unter UNIX laufen der Kontroll-, Speicherungs- und Alarmprozeß des Slow-Control-Systems. Für die Benutzeroberfläche dieser Prozesse können die Vorteile des UNIX Systems genutzt werden.

Die VME Prozessoren benutzen das Betriebssystem OS9. Es ist ebenfalls ein Multiuser-Multitasking-System, liegt in der Komplexität aber weit unter UNIX. Deshalb kann es bei entsprechender Programmierung auch für Echtzeitaufgaben eingesetzt werden. Der Vorteil eines Multitasking-Systems für eine Anwendung dieser Art besteht darin, daß man mehrere Prozesse, die verschiedene Aufgaben erfüllen, zur gleichen Zeit aktiv sein können. Dadurch wird die Modularität des Systems erweitert und die Programmierung vereinfacht. Außerdem hat man die Möglichkeit, das System zu überwachen, ohne dabei in den Lauf der aktiven Prozesse einzugreifen. Auf dem OS9 Sytem laufen die eigentliche Datenerfassungs- und Datenkontrollprozesse des Slow-Control-Systems.

4.8.2 Programmiersprache

Alle Programme werden in der Sprache C geschrieben. Diese bietet neben den Datenstrukturen einer höheren Programmiersprache auch noch umfangreiche Möglichkeiten zur hardwarenahen Programmierung. Dabei sind vor allem die direkte Speicheradressierung und die Funktionen zur Bitmanipulation zu nennen. Als Compiler wird der GNU C++ Compiler verwendet. Dieser Compiler ist ein Public Domain Programm. Er läuft sowohl auf dem UNIX wie auch auf dem OS9 Betriebssystem. In den Programmen der Datenerfassung werden auch die Möglichkeiten zur objektorientierten Programmierung genutzt, die die Sprache C++ bietet. Die Slow-Control-Programme benutzen jedoch nur Code in der Sprache C. Da C++ nur eine Erweiterung der Sprache C ist, verarbeitet der Compiler auch Programme im ANSI C Standard. Auch hier ist es von Vorteil, nur einen Compiler installieren und warten zu müssen.

4.8.3 Kommunikation

Ein zentraler Punkt im Design der Slow-Control-Software ist die Kommunikation voneinander unabhängiger Prozesse, die in unterschiedlichen CPUs laufen. Da diese Prozesse auf unterschiedlichen Hardwareplattformen laufen, war es klar, daß die Kommunikation über das Ethernet verlaufen muß.

Am Anfang wurde erwägt Daten und Nachrichten über Daten-Sockets der Systeme direkt zu verschicken. Mit dieser Methode wird eine direkte Verbindung zwischen Prozessen auf verschiedenen Computern hergestellt. Dabei werden Blöcke von binären Daten von einem CPU an das andere weitergereicht. Ein Satz von Testprogrammen lief zufriedenstellend, es konnten Datenblöcke zwischen einem VME System und einem UNIX System transportiert werden. Da jedoch die Datensicherheit dieser Methode und seine Zuverlässigkeit bei verschiedenen Netzauslastungen nicht kontrolliert gemessen werden konnten, wurde diese Kommunikationsmethode verworfen.

Schließlich wurde das kommerzielle Programmpaket ISIS ausgewählt, um die Kommunikation innerhalb des CHORUS Systems zu realisieren. In diesem Paket sind alle Anforderungen erfüllt, die an ein Kommunikationssystem gestellt werden. Außerdem enthält es auch einen Mechanismus, mit dem Prozesse auf verschiedenen Systemen überwacht werden können.

ISIS stellt einen Verständigungmechanismus zur Verfügung, der unabhängig von der CPU ist, auf der ein Prozeß läuft. ISIS reagiert dabei auf vorher definierte Ereignisse. Das zentrale Konzept von ISIS heißt dabei Virtuelle Synchronität. Dies bedeutet, daß alle Prozesse Ereignisse in der gleichen Reihenfolge sehen. Das am häufigsten vorkommende Ereignis in einem ISIS Prozeß ist der Erhalt einer Nachricht. Falls also von einem Kontrollprozeß aus eine Start- und darauf eine Stopnachricht geschickt werden, so garantiert ISIS, daß die Stopnachricht nie vor der Startnachricht eintrifft. Diese Garantie gilt aber nicht nur für Ereignisse, die von einem einzigen Prozess ausgehen, sondern für alle Ereignisse innerhalb eines Systems von Prozessen, das sich über mehrere CPUs und Netzwerke erstreckt. Dadurch wird die Kausalität eines Systems gewährleistet, und es kann so programmiert werden, als ob alle Ereignisse synchron ablaufen würden.

ISIS ist hauptsächlich ein System zum Austausch von Nachrichten. In ISIS wird eine Nachricht immer an Gruppen von Prozessen geschickt. Diese Prozeßgruppen bilden die Einheiten eines ISIS-Systems. Eine Gruppe kann eine beliebige Anzahl von Mitgliedern haben. Sie kann z.B. auch nur aus einem Prozeß bestehen. Die durch das System gegebene Obergrenze liegt bei 1024 Prozessen. Wenn eine Gruppe eine Nachricht erhält, so wird sie an jedes Mitglied der Gruppe ausgeliefert. Eine Nachricht enthält dabei eine Identifikationsnummer. Falls in einem Prozeß diese Nachrichtenart definiert ist, so wird ein Subprozeß gestartet, mit dem Inhalt der Nachricht als Argument.

Grundsätzlich werden in ISIS Ereignisse mit Subprozessen (Tasks) verknüpt, die auf ein Ereignis reagieren. Ein ISIS-Ereignis ist der Erhalt einer Nachricht, eine Änderung in einer Gruppenmitgliedschaft (Eintritt oder Austritt eines ISIS-Prozesses, oder auch der Ausfall eines Prozesses), eine Eingabe von der Tastatur oder ein Signal des Betriebssystems. Dabei wird durch die Verwendung von ISIS eine Programmstruktur vorgegeben. Die ersten Befehle eines ISIS-Programms bestehen aus Definitionen von ISIS-Tasks und zugehörigen ISIS-Ereignissen. Nach den Definitionen tritt das Programm in den ISIS-Main-Loop ein. Dabei wird ein Initialisierungstask aufgerufen. In diesem werden alle Daten des Programms initialisiert. Danach tritt das Programm in eine oder mehrere Prozeßgruppen ein. Sobald dieses erfolgt ist, sendet das Programm eine Meldung an das ISIS-System, daß es von nun auf Ereignisse reagieren kann. Ab diesem Zeitpunkt startet der ISIS-Main-Loop beim Auftreten eines vorher definierten Ereignisses den dazugehörigen Task. Im Gegensatz zu einer konventionellen Programmstruktur, in der immer nur ein Unterprogramm die Kontrolle über den Programmfluß haben kann, ist es in einem ISIS Programm möglich, daß mehrere ISIS-Tasks parallel laufen.

Das ISIS-Packet besteht aus einem Koordinationsprogramm und einer Funktionslibrary in der Sprache C. Das ISIS-Koordinationsprogramm läuft auf einem zentralen CPU (im Fall CHORUS auf einem der IBM Online-Server) und verwaltet die Gruppenmitgliedschaften sowie den Nachrichtenfluß innerhalb des ISIS-Systems. Anwendungsprogramme benutzen die in den ISIS-Libraries definierten Funktionen, um mit dem Kontrollprozeß zu kommunizieren. Alle ISIS-Kommunikationen benutzen dabei das Ethernet.

Nach der Installation von ISIS wurden alle für das CHORUS-System benötigten ISIS-Funktionen mit Hilfe von Testprogrammen überprüft. Außerdem wurde ein Interface zwischen ISIS und dem Buffer-Manager der Datenerfassung geschrieben, das es ermöglicht, ISIS-Nachrichten in einen Buffer zu schreiben. Dieses Interface wird hauptsächlich in den Monitoring-Programmen der Datenerfassung genutzt. Die Leistung und die Zuverlässigkeit der Testprogramme waren dabei so zufriedenstellend, daß die Struktur des Slow-Control-Systems vollkommen auf einer ISIS-Struktur basiert.

4.8.4 Datenbank

Zur Speicherung von Daten wird die HEPDB (High Energy Physics DataBase) Datenbank benutzt, ein am CERN entwickeltes Softwarepaket. Jeder Wert, der gespeichert werden soll, wird zusammen mit einem Zeitindex in eine Datenbankstruktur eingegeben. Diese Struktur besteht aus einem Verzeichnisbaum, dessen Blätter die Namen der zu speichernden Kanäle sind. Für das CHORUS Slow-Control-System wird eine Auslesefunktion geschrieben, die die Werte eines bestimmten Zeitraums in einen Array schreibt, d.h. alle Werte, für deren Zeitindex t gilt: $t_{min} < t < t_{max}$. Dieser Array wird dann mit den zugehörigen Zeitindizes an das Darstellungsprogramm übergeben.

Um große Zugriffszeiten zu vermeiden, wird eine lokale Version von HEPDB benutzt, so daß die Datenbank auch als Zwischenspeicher genutzt werden kann. Einmal pro Woche werden Daten, die älter als einen Monat sind, aus dem Datenbankdatei entfernt und auf Band geschrieben, um Speicherplatz zu sparen.

Die HEPDB Datenbank wurde ausgewählt, weil sich mit ihr das Speichern und Zurücklesen von Daten ohne großen Overhead durchführen läßt.

4.8.5 Benutzeroberfläche

Für die grahische Benutzeroberfläche standen drei Möglichkeiten zur Auswahl.

Erstens wurde es erwogen, die Benutzeroberfläche mit Hilfe der MOTIF C-Libraries zu programmieren. MOTIF ist ein Standard für Anwendungen in einer XWindows-Umgebung. In den dazugehörigen Libraries wird eine große Anzahl von Widgets (Fensterelemente: Knöpfe, Schalter, Schiebeleisten etc.) zur Verfügung gestellt, aus denen sich Anwendungen zusammensetzen lassen. Mit Hilfe dieser Libraries läßt sich jede beliebige Anwendung realisieren. Ein Nachteil dieser Methode ist jedoch die Umständlichkeit der Programmierung. Zur Initialisierung eines einzigen Widgets müssen mehrere C-Funktionen aufgerufen, ebenso wird jede Eigenschaft eines Widgets (Größe, Position, Farbe, Beschriftung und die bei einem Mausklick auszuführende Aktion) über separate Funktionsaufrufe eingestellt. Dadurch werden die Programme lang und schwer verständlich. Dazu kommt noch, daß für jede Änderung der Anwendung das gesamte Programm neu kompiliert werden muß. Aus diesen Gründen wurde eine Programmierung mit den MOTIF Libraries verworfen.

Da die Diagramme für das Slow-Control-System und die Datenerfassung mit PAW (Physics Analysis Workstation) dargestellt werden (s. nächste Sektion), wurde als zweite Möglichkeit zur Erstellung der Benutzeroberfläche das PAW-MOTIF-Interface in Betracht gezogen. Mit diesem Interface kann PAW in einer XWindows-Umgebung über eine Maus bedient werden. Es besteht auch die Möglichkeit, eine eigene Menüstruktur zu definieren. Die Programmierung dieser Menüs wird in der KUMAC Kommandosprache durchgeführt. Es wurde jedoch festgestellt, daß das PAW-MOTIF-Interface in seinen Möglichkeiten zu limitiert ist, um eine Benutzeroberfläche zu programmieren. Als einziges Schaltelement in den Menüs gibt es nur Druckschalter. Und auch für diese Schalter gibt es keine Kontrolle über ihre Positionierung. Bei der Benutzung des PAW-MOTIF-Interfaces würde deshalb auf die Möglichkeit einer intuitiven Benutzerführung verzichtet.

Als dritte Möglichkeit wurde das TK/TCL (Tool Kit/Tool Command Language) Programmpaket in Erwägung gezogen. Es ist ein Paket aus dem Public Domain Bereich. TCL ist eine interpretierte Batch Sprache, die in der mitgelieferten WISH (WIndowing SHell) Umgebung des UNIX Systems läuft. TK ist eine Funktionslibrary, die die Erstellung und Wartung von Widgets ermöglicht. Im TK sind TCL Kommandos definiert, mit denen in einer einzigen Befehlszeile ein Widget mit seinen Parametern programmiert werden kann. Die Aktion, die bei einem Mausklick auf dem Widget ausgeführt wird, ist ein TCL Kommando oder eine TCL Prozedur. TCL bietet auch ein Kommando, mit dem beliebige Prozesse aufgerufen werden können, so daß nicht nur TCL Anwendungen aufgerufen werden könnnen.

Da es sich bei TCL um eine interaktive Sprache handelt, können auch während eines Programmlaufes alle Eigenschaften eines Widgets verändert werden. Aus diesem Grund sind mit TCL geschriebene Anwendungen sehr flexibel. Durch die ständige Interpretation wird die Programmgeschwindigkeit zwar verlangsamt, doch dieser Nachteil wird von der einfachen Programmierung mehr als aufgewogen. Erste Erfahrungen mit TK/TCL zeigen, daß es für die Erstellung einer Benutzeroberfläche sehr gut geeignet ist.

Zur Kommunikation mit den Slow-Control-Prozessen kann mit einem TCL Kommando ein ISIS-Prozess gestartet werden, der dann eine Nachricht an den entsprechenden Prozeß schickt. Wenn ein Slow-Control-Prozeß Daten an die Benutzeroberfläche übergeben will, so schickt er sie in einer Nachricht an einen dafür vorgesehenen ISIS-Prozeß. Dieser Prozeß schreibt die Daten in ein UNIX Shared-Memory-Segment, das auch von der Sprache TCL aus gelesen werden kann. Dieser Mechanismus ist speziell für CHORUS entwickelt worden.

4.8.6 Graphische Darstellung

Zur Darstellung von Diagrammen innerhalb des Slow-Control-Systems wird das PAW (Physics Analysis Workstation) Paket benutzt. Dieses wurde am CERN entwickelt und bietet umfangreiche Möglichkeiten zur Analyse und graphischen Darstellung von physikalischen Daten. Um die Bedienung von PAW, das über einen sehr großen Befehlssatz verfügt, zu vereinfachen, werden alle Befehle an PAW von der graphischen Benutzeroberfläche gegeben. Dies bedeutet, daß kein Fenster für die interaktive Benutzung von PAW geöffnet wird, sondern daß auf einen Mausklick hin eine vollständige Befehlszeile an PAW zur Verarbeitung übergeben wird. Über ein Kommando-Fenster können einige Eigenschaften des Diagrammes wie Skalierung oder betrachteter Zeitraum verändert werden. Es besteht hier die Absicht, Daten auf eine einfache Weise darzustellen, ohne auf komplexe Befehlsfolgen zugreifen zu müssen. Falls eine weitergehende Bearbeitung der Daten gewünscht werden sollte, so kann die interaktive PAW Version benutzt werden.

Im Slow-Control-System werden nur zweidimensionale Histogramme benötigt. PAW ist im Grunde genommen ein zu umfangreiches Programmpaket für eine so einfache Aufgabe auszuführen. Im Rahmen der Datenerfassung werden jedoch komplexere Diagramme zur Überwachung der Online-Daten benötigt. Der oben beschriebene Mechanismus zur Darstellung von Diagrammen wurde im Rahmen der Datenerfassung entwickelt. Aus Gründen der Einfachheit werden die entsprechenden Prozeduren kopiert, um eine doppelte Entwicklungsarbeit zu vermeiden.

4.9 Software

Das Slow-Control-System für CHORUS besteht aus einer Gruppe von parallel laufenden Prozessen, die voneinander unabhängig sind. Die Kommunikation der Prozesse untereinander erfolgt dabei durch den Austausch von ISIS-Nachrichten.

Das System besteht aus drei Ebenen. In der obersten Ebene findet die Kommunikation mit dem Benutzer statt. Sie ist der einzige Bereich, der für den Endbenutzer sichtbar ist. Der zentrale Punkt in dieser Ebene ist der Bildschirm. Alle Ein- und Ausgaben werden mit Hilfe des Bildschirmes ausgeführt. Zur Realisierung der Eingabe wird dabei eine graphische Benutzeroberfläche verwendet. Das heißt, daß Kommandos aus einem Menüsystem mit Hilfe einer Maus ausgewählt werden. Diese Befehlsstruktur vermeidet ein mühsames Eintippen von längeren Kommandos, die oft erst nachgeschlagen werden müssen. Dem Benutzer stehen dabei die folgenden Befehle zur Verfügung:

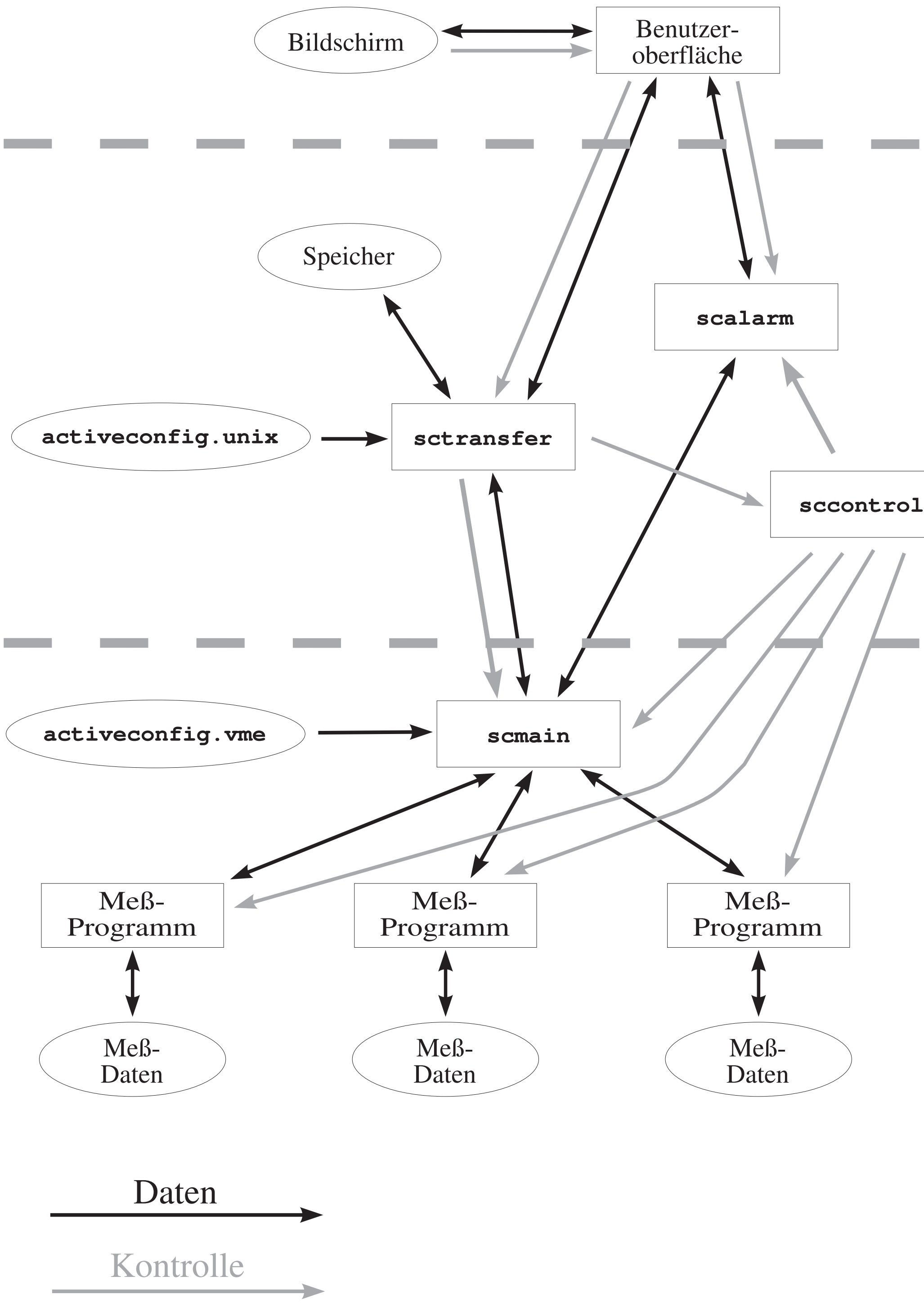

Abbildung 4.4: Layout des Software-Systems

- Alarmübersicht
  Eine Anzeige von allen Kanälen, die unter Alarm stehen.

- Zustand
  Der letzte gemessene Wert und Alarmzustand eines Kanales wird ausgegeben.

- Messung
  Das Slow Control System wird aufgefordert, einen Kanal außerhalb des normalen Turnus zu messen. Dieser Befehl ist vor allem dann nützlich, wenn eine Reparatur durchgeführt wurde und eine sofortige Kontrolle verlangt wird.

- Alarm aussetzen
  Ein Kanal wird angegeben, für den dann in den nächsten 2 Stunden keine Alarme angezeigt werden, z.B. während der Reparatur eines Gerätes. Nach Ablauf dieser Frist fragt das System nach, ob Alarme für diesen Kanal weiterhin ignoriert werden sollen. Diese Abfrage wird in regelmäßigen Abständen wiederholt, bis der Alarm erneut angestellt wird. Dies soll verhindern, daß ein Alarm aus Versehen abgeschaltet bleibt. Der Wert von 2 Stunden ist dabei willkürlich gewählt, da noch keine Praxiserfahrung mit dem System vorhanden ist.

- Alarm anschalten
  Für einen Kanal sollen Alarme wieder angezeigt werden.

- Zeitverlauf
  Falls in der Konfigurationsdatei angegeben worden ist, daß die Werte eines Kanals gespeichert werden sollen, so wird mit diesem Befehl ein Diagramm über den zeitlichen Verlauf dieser Werte erstellt. Dabei kann gewählt werden, ob die Werte des letzten Tages, der letzten Woche oder des letzten Monats angezeigt werden. Außerdem kann das Diagramm auf Wunsch ausgedruckt werden.

In dieser oberen Ebene ist es nicht möglich, dauerhafte Veränderungen an der Konfiguration des Systems durchzuführen. Dies soll verhindern, daß ein unerfahrener Benutzer versehentlich Funktionen des Systems abstellen kann, die zur Überwachung notwendig sind.

In der mittleren Ebene befindet sich die Kontrolle des Systems sowie die Datenanalyse. Alle gemessenen Daten werden hier gesammelt und können bei Bedarf gespeichert und analysiert werden. Dazu gehört auch die Erstellung und Verarbeitung der Konfigurationsdateien. Die Programme in dieser Ebene laufen auf der IBM RS6000 Workstation. Sie können nur mit dem Password des Slow-Control-Systems benutzt werden. Das Programm `sccontrol` überwacht dabei die anderen Programmen. Dieses Programm wartet, bis alle zum Betrieb des Slow-Control-Systems notwendigen Programme laufen und gibt dann ein Startsignal an alle Programme. Bei Ausfällen von Programmen kann es je nach Anforderung das System anhalten und/oder entsprechende Warnungen mitteilen. Mit diesem Programm kann dem Slow-Control-System mitgeteilt werden, das System zu pausieren, erneut zu starten, neu zu initialisieren oder zu beenden. Das Programm `sctransfer` verwaltet den Austausch von Daten zwischen den verschiedenen Programmen, dem Speicher und der Benutzeroberfläche. Seine erste Aufgabe ist es, die Datei `activeconfig.unix` zu lesen und die in ihr enthaltenen Informationen zu speichern. Im Verlauf der Messungen werden alle zu speichernden Daten an diesen Prozeß weitergeleitet. Es legt die Werte im Speicher ab und stellt sie auf Anfrage zusammen mit den Darstellungsparametern der Benutzeroberfläche zur Verfügung. Dieser Programm empfängt auch die Daten, die dem Slow-Control-System von externen Prozessen zur Verfügung gestellt werden. Dies sind z.B. Informationen über den Runstatus des Experimentes oder über den Strahlstatus. Außerdem ist in dieser Ebene das Programm `scalarm` enthalten. Es hat die Aufgabe, den Alarmzustand der verschiedenen Kanäle zu überwachen und diese Informationen an die Benutzeroberfläche weiterzuleiten. Falls sich der Alarmzustand eines Kanals ändert, so wird dies in eine Alarmdatei eingetragen. Auf Anfrage des Benutzers erstellt dieses Programm eine Liste aller unter Alarm stehenden Kanäle. Außerdem verwaltet es die Alarmzustände der verschiedenen Kanäle. Falls für einen Kanal kurzzeitig keine Alarmeldungen ausgegeben werden sollen, so werden die Alarme von `scalarm` nicht an die Benutzeroberfläche weitergeleitet. Nachdem die vorgegebene Zeit abgelaufen ist, veranlaßt `scalarm` dann die Benutzeroberfläche zu einer Abfrage über den Zustand des Kanales.

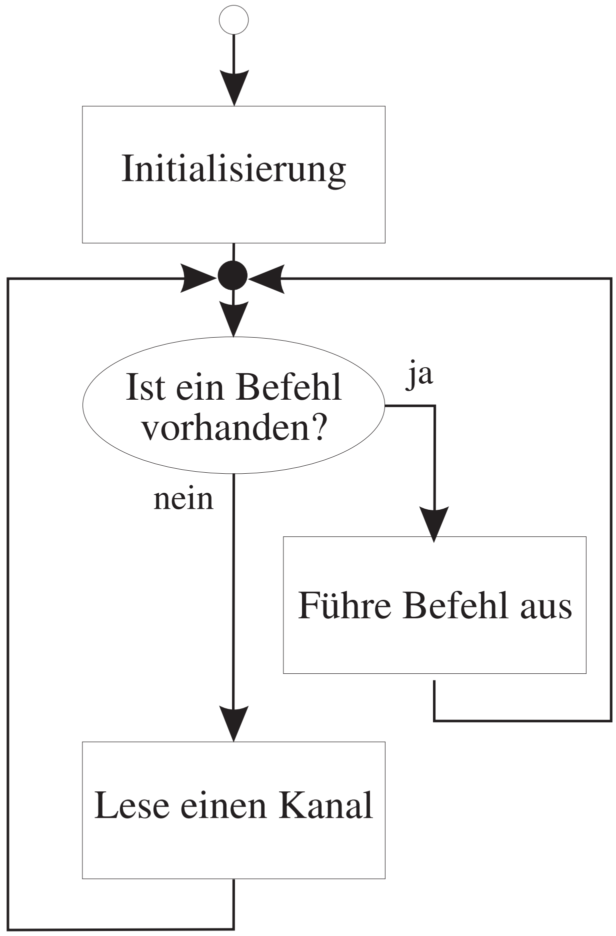

Abbildung 4.5: Struktur von `scmain`

In der dritten und untersten Ebene findet die Messung und Kontrolle der Daten statt. Das Programm zur Datenkontrolle, `scmain`, ist das Hauptprogramm des Slow-Control-Systems. Es liest alle zur Auslese notwendigen Parameter aus der Datei `activeconfig.vme` und gibt sie an Meßprogramme zur Datenerfassung weiter. Diese Meßprogramme lesen die von `scmain` gewünschen Werte und leiten sie ohne jede Kontrolle an `scmain` weiter. Dieses Programm vergleicht die gemessen Werte mit den angegebenen Ober- und Untergrenzen und berechnet die Änderung des Wertes seit seiner letzten Messung. Falls der Wert nicht im vorgegebenen Bereich liegt oder sich zu stark geändert hat, wird eine Mitteilung an `scalarm` geschickt. Danach wird überprüft, ob der augenblickliche Wert des Kanales gespeichert werden soll. Wenn dies der Fall ist, wird der aktuelle Wert an das Programm `sctransfer` weitergeleitet. Falls nicht wird der Wert nur intern für den Vergleich mit der nächsten Messung gespeichert. Nach jedem gemessenen Wert kontrolliert der `scmain`, ob ein Befehl eingegangen ist. Nachdem es den Befehl ausgeführt hat, fährt das Programm mit dem Lesen des nächsten Kanales fort.

5. Hardwareparameter

Die Aufgabe eines Slow-Control-Systems ist die Überwachung von Hardwareparametern in einem Experiment. Im folgenden Kapitel wird eine Zusammenstellung der zu überwachenden Kanäle für den CHORUS-Detektor gegeben, zusammen mit den dazu notwendigen Auslesemethoden. Es ist eine vorläufige Liste, da noch nicht alle Auslesemethoden vollständig definiert sind. Sie bietet jedoch einen guten Überblick über den Umfang des CHORUS Slow-Control-Systems.

5.1 Optoelektronik

Die 58 Bildverstärkerketten der Optoelektronik sind an CAEN Hochspannungsversorgungen angeschlossen. Die Hochspannungen und Ströme in den Spannungsversorgungen sollen von der Slow-Control überwacht werden. Dies geschieht mit einem VME HVCAEN-Controller-Modul. Insgesamt werden 408 Kanäle für diese Aufgabe benötigt.

Weiterhin sollen die Ströme in den einzelnen Komponenten der Bildverstärkerketten auch einzeln überprüft werden. Dazu werden sie in eine Frequenz umgewandelt, die proportional zur Stromstärke ist. Dies ergibt dann $4 \cdot 58 = 290$ Kanäle. Dazu kommt noch ein Kanal als Trigger für die Messung und ein Kanal, auf dem eine definierte Frequenz zur Normierung der Stromstärke gemessen wird. Diese Werte werden von G64 Scaler (Frequenzzähler) Karten gelesen.

Die Spannungsversorgungen der Optoelektronik-Crates werden von einem G64 Multiplexersystem gelesen. Dieser analoge Multiplexer wurde von der CHARM-II Kollaboration übernommen. Er ist das einzige Gerät, das Werte von mehreren Subdetektoren ausließt. Er wird dazu benutzt, um niedrige Spannungen ($< 100\ V$) zu messen. Für die Optoelektronik werden 32 Werte mit dem G64 Multiplexer überwacht.

Ferner soll das Slow-Control-System noch die Pulsform der Gating-

Impulse an den MCPs messen. Diese Aufgabe fällt ein wenig aus dem Rahmen eines Slow-Control-Systems, da hier ein zeitabhängiger Parameter überwacht werden soll. Da die Pulsform jedoch kein Wert ist, der mit den physikalischen Werten eines Ereignisses auf Band geschrieben wird, wurde entschieden, diese Werte innerhalb der Slow-Control zu messen. In Abständen von ca. 5 Minuten wird dazu jeweils die Pulsform an einer Kette ausgelesen. Danach wird zur nächsten Kette gewechselt. Bei 58 Ketten bedeutet dies, daß in etwa 6 Stunden jede Pulsform einmal nachgemessen wird. Nach den Angaben der Optoelektronikgruppe ist dieser Zeitrahmen ausreichend. Die Messungen selbst werden mit VME Modulen durchgeführt.

5.2 Kühlraum

Innerhalb des Kühlraumes ist es wichtig, die allgemeinen Umgebungswerte wie Temperatur, Luftfeuchtigkeit und Lichtintensität zu überwachen, da diese Werte kritisch für die Alterungsbedingungen der Emulsion und der Szintillationsfasern sind. Außerdem soll in der Nähe der Emulsion ein elektronisch auszulesendes Strahlungsmeßgerät installiert werden, z.B. ein Geiger-Müller-Zählrohr, das über einen ADC ausgelesen wird. Die Definition dieser Kanäle ist noch nicht vollständig. Es wird erwartet, daß ca. 30 Kanäle ausgelesen werden müssen.

5.3 Hexagonalmagnet

Für den Hexagonalmagneten sollen die folgenden Betriebsparameter überwacht werden:

- Temperatur der Windungen (12x)
- Luftfluß durch das Kühlsystem (2x)
- Pulsstrom (1x)
- Magnetfeld (1x)
- Spannungsversorgung (2x)

- Spannungsversorgung des Pulsers (2x)

Diese 20 Kanäle sollen mit einer G64 Karte ausgelesen werden.

5.4 Kalorimeter

Das Kalorimeter besteht aus 1256 Photomultipliern und 22 Streamer-Kammern. Dazu kommen 4 Streamer-Kammern in dem Zwischenraum zwischen dem Kühlraum und dem Kalorimeter. Die Streamer-Kammern werden nur digital ausgelesen. Zu ihrer Überwachung wird die Hochspannung und der Stromfluß an den Spannungsversorgungen gemessen. Dies sind 52 Kanäle. Ferner werden die Versorgungsspannungen der zur Auslese der Streamer Kammern notwendigen Crates überwacht. Der G64 Multiplexer übernimmt diese Aufgabe.

Die 1256 Spannungsversorgungen (LeCroy) der Photomultiplier werden mit Hilfe eines LeCroy-Interfaces ausgelesen. Die Versorgungsspannungen der dazugehörigen Elektronik-Crates werden ebenfalls mit dem G64 Multiplexer gemessen. Dies führt zu 66 weiteren Kanälen.

5.5 Myonenspektrometer

Die Hochspannungsversorgungen der Drift- und Streamer-Kammern des Myonenspektrometers werden mit einem VME HVCAEN- Controller gemessen. Spannung und Strom der 56 Driftkammern ergeben 112 Kanäle, dazu kommen 21 Kanäle für die Spannungsversorgungen der 7 Driftkammern. Das gleiche CAEN-Modul kann auch benutzt werden, um die Spannungsversorgungen von 6 alten Elektronik-Crates nahe dem Myonenspektrometer auzulesen.

Die 300 Szintillatoren in den Magneten werden mit Photomultipliern ausgelesen. Spannungen und Ströme an den Photomultipliern werden vom Slow-Control-System mit einem HSCAEN VME-Modul gemessen. Daraus ergeben sich 600 weitere Kanäle.

Für die Überwachung der weiteren Elektronik-Crates des Myonenspektrometers werden 645 Kanäle von dem G64 Multiplexer gemessen.

Die Kontrolle der Magneten besteht aus einem separaten System. Es ist noch nicht klar, wie die Werte an das Slow-Control-System übergeben werden sollen. Es soll ein Mechanismus definiert werden, mit dem die Werte über Ethernet verschickt werden können.

Zur Überprüfung der Gasflüsse im Myonenspektrometer soll ein neues System aus VME-Modulen eingesetzt werden. Eine genaue Definition dieses Systems liegt noch nicht vor. Im alten Kontrollsystem des Spektrometers wurden 20 Kanäle überwacht.

5.6 Trigger

Wie bei den Photomultiplier im Myonenspektrometer, so werden auch die Versorgungsspannungen der Photomultiplier an den Trigger-Ebenen mit einem HSCAEN VME-Modul ausgelesen.

Die Kontrolle der Elektronik-Crates wird wieder mit dem G64 Multiplexer ausgeführt. Insgesamt ergeben sich 220 HSCAEN- und 203 G64MUX-Kanäle.

5.7 VME-Crates

Die Versorgungsspannungen aller 140 VME-Crates, die für CHORUS in den verschiedenen Subsystemen Verwendung finden, werden mit einem HSCAEN-Controller gemessen. Dieser Controller ist auch ein VME-Modul.

5.8 Zusammenfassung

In der folgenden Tabelle werden alle Kanäle, die vom Slow-Control-System überwacht werden sollen, noch einmal aufgeführt.

Subsystem		Methode	Sp.	Anzahl	Summe
Optoelektronik	HV	HSCAEN	J	408	
	Gating	VME	?	290	
	Crates	G64MUX	N	10	708
Kühlraum		G64?	?	30	30
Hexagonalmagnet		G64	?	20	20
Kalorimeter	HV-Streamer	HVCAEN	J	52	
	LV-Streamer	G64MUX	N	68	
	HV-Szint.	LeCroy	J	1256	
	LV-Szint.	G64MUX	N	66	1442
Spektrometer	HV-Streamer	HVCAEN	J	133	
	alte Crates	HVCAEN	Y	6	
	HV-Szint.	HSCAEN	J	600	
	LV	G64MUX	N	645	
	Gas	VME?	J	20	1398
Trigger	HV-Szint.	HSCAEN	J	220	
	Crates	G64MUX	N	202	
Alle	VME Crates	HSCAEN	N	140	140
					4160

In der Spalte „Sp.“ wird angegeben, ob die entsprechenden Werte in regelmäßigen Abständen abgespeichert werden sollen, um einen Überblick über ihren zeitlichen Verlauf zu haben.

Die Endsumme von 4160 zu überwachenden Kanälen steht noch nicht genau fest, aber an der Größenordnung dieses Wertes wird sich nichts mehr ändern.

5.9 Informationen von anderen Systemen

Über das Ethernet wird das Slow-Control-System auch noch Informationen von anderen Systemen erhalten. Dazu gehört das Kontrollsystem des Strahls, das Sicherheitssystem und die Kontrollsysteme der beiden Magneten. Die Daten über die Magneten werden für die Kontrollaufgabe des Slow-Control-Systems benötigt. Die Zustände des Strahls und des Sicherheitssystems gehören prinzipiell nicht in den Aufgabenbereich des Slow-Control-Systems. Es ist jedoch nützlich, wenn das Slow-Control-System über diese Informationen verfügt, um in entsprechenden Situationen Verwirrung zu vermeiden. Wenn z.B. die Stromversorgung in der BEBC Halle zusammenbricht, so laufen der CHORUS Server und das Slow-Control-System weiter, da sie an das Notstromsystem angeschlossen sind. Wenn das Slow-Control-System über diesen Zustand Bescheid weiß, so kann es diese Information ausgeben und braucht die daraus folgenden Alarme nicht mehr anzuzeigen.

6. Zusammenfassung

Das CHORUS Slow-Control-System befindet sich noch im Aufbau. Zum Zeitpunkt dieser Diplomarbeit ist die Konzeption des Systems beendet, doch zu seiner Implementierung ist noch viel Arbeit zu leisten.

Die Entwicklung der Hilfsprogramme zur Erstellung von Konfigurationsdateien ist abgeschlossen. Von den eigentlichen Programmen des Systems gibt es zur Zeit jedoch nur Prototypen. Diese laufen zufriedenstellend und die Kommunikation zwischen ihnen verläuft reibungslos. Es fehlt allerdings noch ein großer Teil der Funktionalität. Die einzige Hardware, für die ein Meßprogramm existiert, ist im Augenblick der analoge Multiplexer in dem G64 System. Es ist noch keine Software für die Benutzeroberfläche geschrieben worden.

Zur Zeit wird am CERN die Hardware des G64 Multiplexers installiert und es werden die schon bekannten Parameter von Meßkanälen in Konfigurationsdateien eingegeben.

Es ist geplant, für den Teststrahl im Oktober 1993 ein Kernsystem fertigzustellen, das den größten Teil der vorhandene Kanäle auslesen und Alarmmeldungen generieren kann. Während des Teststrahls sollen dabei die Programme unter reellen Bedingungen auf ihre Zuverlässigkeit geprüft werden. Zu diesem Zeitpunkt wird es noch keine graphische Benutzeroberfläche geben. Dieser Teil des Slow-Control-Systems hat zuerst nur eine geringe Priorität, da es am Anfang wichtiger ist, Werte zuverlässig zu messen, als sie gefällig darzustellen.

Zum Beginn der CHORUS Meßzeit im März 1994 soll dann das gesamte System laufen. Während des Verlaufs des Experimentes kann und wird das System dann mit der Hilfe von in der Praxis gewonnenen Erfahrungen optimiert werden, um seine Funktionalität und Bedienung zu verbessern.

Literaturverzeichnis

[Dav] R. Davis Jr. et al., proc. 21st International Cosmic Ray Conf., Ed. R.J. Protheroe, Vol. 12 (University of Adelaide Press, Adelaide, Australia, 1990) 143

[Hir] K.S. Hirata et al., Kammiokande II Collaboration, Phys. Rev. Lett. 65 (1990) 1301

[Ans] P. Anselmann et al., GALLEX Collaboration, Phys. Lett. B 285 (1992) 376

[Aba] A.I. Abazov et al., SAGE Collaboration, Phys. Rev. Lett. 67 (1991) 3332

[Har] H. Harari, Phys. Lett. B 216 (1989) 413

[Ush] N. Ushida et al., Phys. Rev. Lett. 57 (1986) 2897

[Dyd] F. Dydak et al., CDHS Collaboration, Phys. Lett. B 134 (1984) 281

[Ber] F. Bergsma et al., CHARM Collaboration, Phys. Lett. B 142 (1984) 103

[Sto] I.E. Stockdale et al., CCFR Collaboration, Z. Phys. C 27 (1985) 53

[Gru] M. Gruwé et al., CHARM II Collaboration, CERN-PPE/93-nn (1993)

[CHO] Proposal to search for $\nu_\mu \to \nu_\tau$ oscillations, CHORUS Collaboration, CERN-SPSC/90-42 (1990)

[Gue] S. Guenther, P. Skavril, J. Strachota, Proc. 10th International conf. on computing in high-energy physics, Ed. C. Verkerk - CERN, Geneva, 1992 - (CERN 92-07)

Ich möchte meinen Dank ausdrücken

an H. Øverås und F. Cataneo für ihre Mitarbeit und Unterstützung bei der Erstellung des CHORUS Slow Control Systems.

an Prof. Dr. D. Frekers für die gute Betreuung der Diplomarbeit.

an D. Rondeshagen und R. Weiß für das Korrekturlesen der Arbeit.

an P. Lendermann für die prompte Beschaffung von Bildern.

an die Mitglieder der CHORUS Kollaboration, die mir bei allen Problemen und Fragen mit Rat und Tat zur Seite gestanden haben.

an meine Eltern für ihre volle Unterstützung während meines gesamten Studiums.

Ich versichere, daß ich diese Arbeit selbst verfaßt und keine weiteren Hilfsmittel außer den hier angegebenen verwendet habe.

Münster, August 1993