KfK 3712
Juni 1984

Ein schnelles Datenerfassungssystem für Spektroskopieexperimente

U. Raich
Institut für Kernphysik

Kernforschungszentrum Karlsruhe

KERNFORSCHUNGSZENTRUM KARLSRUHE

Institut für Kernphysik

KfK 3712

Ein schnelles Datenerfassungssystem für Spektroskopieexperimente*

Ulrich Raich

Kernforschungszentrum Karlsruhe GmbH, Karlsruhe

*von der Fakultät für Physik der Universität Karlsruhe genehmigte Dissertation

Als Manuskript vervielfältigt
Für diesen Bericht behalten wir uns alle Rechte vor

Kernforschungszentrum Karlsruhe GmbH
ISSN 0303-4003

ABRISS

Die Fertigstellung des Speicherrings für langsame Antiprotonen (LEAR) bei CERN erfordert, wegen der gegenüber früheren Antiprotonenstrahlen um mehrere Größenordnungen gestiegenen Datenraten, ein neues Konzept für ein Datenerfassungssystem. Für die Messung der charakteristischen Röntgenstrahlung antiprotonischer Atome wurde ein aus einem Mikroprozessor mit großem Speicher und einem Minicomputer (PDP-11) bestehendes System aufgebaut. Der Mikroprozessor ist verantwortlich für die Auslese und Vorbehandlung experimenteller Daten, während der Minicomputer die Ergebnisse des Mikroprozessors auf Magnetband schreibt und eine Datenanalyse und graphische Darstellung erlaubt. Die Funktionsweise des gesamten Aufbaus wird im Detail beschrieben und die ersten mit diesem System gemessenen Daten vorgestellt.

ABSTRACT

A FAST DATA ACQUISITION SYSTEM FOR SPECTROSCOPY EXPERIMENTS

The new storage ring for low energy antiprotons LEAR at CERN allows to generate antiproton beams with particle rates, which are several orders of magnitude higher than with older beams. This demands for a new concept of the data acquisition system. For the measurements of the characteristic X-ray spectra of antiprotonic atoms a system was developed, which consists of a microprocessor with large memory and a PDP-11 minicomputer. The microcomputer handles the readout of experimental data, while the minicomputer analyses and displays the acquired data and writes them to magnetic tape.

A detailed description of the complete system is given in this work and the first results obtained with the system are shown.

INHALTSVERZEICHNIS

Kapitel Seite

1. Einleitung 1
1.1 Exotische Atome 1
1.2 Formation antiprotonischer Atome 2
1.3 Meßbare Größen 3
1.4 Das Experiment 5
1.5 Erzeugung von Antiprotonen bei LEAR 7
1.6 Anforderungen des Experimentes an das Datenerfassungssystem . . . 8
1.7 Prinzipieller Aufbau des Datenerfassungssystems 10
1.8 Interface Experiment-Mikrocomputer 13
1.9 Kopplung Minicomputer-Mikrocomputer 14

2. Definition der Fachbegriffe und Einführung von CAMAC 17
2.1 Bus und Adressierung 17
2.2 Interrupt 19
2.3 DMA 20
2.4 Einführung in CAMAC 21

3. CAMPORT, Überblick über die verschiedenen Funktionseinheiten 26
3.1 Die beiden CAMPORT Schnittstellen 26
3.2 Funktionsblöcke CAMPORT's 27
3.3 Physikalischer Aufbau CAMPORT's 27

4. CAMPORT, Zugriff des Mikroprozessors auf CAMAC 29
4.1 Anschluß CAMPORT's an den Mikrocomputer 29
4.2 CAMAC Adressierung 30
4.3 Datenzugriff 32
4.4 Funktionsregister 34
4.5 Beispiel eines CAMAC Befehles 37
4.6 Interrupts 37

5. CAMPORT, Zugriff zum Mikrocomputer Speicher 40
5.1 CAMPORT Funktionen ansprechbar vom Minicomputer 40
5.2 Der Normalmode 42
5.3 Der DMA Mode 43
5.4 Die Statusregister 43
5.5 CAMAC Initialisierung 45

6. Das CAMPORT Steuerwerk 46
6.1 Prinzip der Mikroprogrammierung 47
6.2 Erzeugung der CAMPORT Steuersignale 52
6.3 Entstehung eines Mikroprogrammes 56

7. Der VME-Busadapter 61
7.1 Allgemeines über den VME Bus 61
7.2 Eigenschaften des VME Busses 62
7.3 Der VME Busadapter 63

8. Software 67
8.1 Das Ausleseprogramm 67
8.2 Gemessene Auslesezeiten 72
8.3 Die Kommunikationssoftware 74
8.4 Die Protokollsoftware 76
8.5 Absetzen eines Befehles 78
8.6 Anforderungen des Mikroprozessors an den Minicomputer 80
8.7 Die Verwaltungsprogramme 82

9. Erste Ergebnisse 84

10. Erweiterungen in der Zukunft 85
10.1 Der Life Display 85
10.2 Der Busspion 87
10.3 Eingabe 87

11. Weitere Anwendungsmöglichkeiten des Systemes 89
11.1 Die Vieldrahtkammer 89
11.2 Anschluß der Kammer an das Datenerfassungssystem 91
11.3 Kammerauslesesystem mit Mikrocomputer 93
11.4 Software des Kammerauslesesystemes. 93
11.5 Das Eichprogramm 95
11.6 Die normale Datennahme 96

12. Zusammenfassung 99

Anhang A: Flip Flop Einheit 101
A.1 Benutzung der Dual Flip Flop Einheit 102

Anhang B: ADC Koppler 103
B.1 Das Handshakeprotokoll 103
B.2 CAMAC Funktionen der ADC Koppler 105
B.3 Routing 105
B.4 Ausgangssignale des ADC Kopplers 107

Anhang C: Auszüge aus dem CAMPORT Mikroprogramm 108

Anhang D: Pinbelegung der Kabelverbindung CAMPORT-KDM Bus 110

Anhang E: Pinbelegung der VME Buskonnektoren 111

Verzeichnis der Programmbeispiele 113

Verzeichnis der Tabellen 114

Verzeichnis der Abbildungen 115

Index der Fachbegriffe 118

Quellenverzeichnis 121

KAPITEL 1

EINLEITUNG

1.1 Exotische Atome

Unter einem exotische Atom versteht man ein Atom, bei welchem ein Elektron der Hülle durch ein anderes negativ geladenes Teilchen (Myon, Pion, Antiproton...) ersetzt ist. Mit modernen Teilchenbeschleunigern ist es möglich, diese Elementarteilchen in großer Menge herzustellen, exotische Atome zu formen und sie durch ihre charakteristische Röntgenstrahlung nachzuweisen. Antiprotonische Atome spielen dabei wegen der hohen Masse des Antiprotons und wegen der starken Wechselwirkung, der das Antiproton ausgesetzt ist, eine besondere Rolle. Verglichen mit den Mesonen jedoch ist das Antiproton, auf Grund seiner hohen Masse, wesentlich schwerer zu erzeugen. Außerdem ist die Herstellung von Antiprotonen immer von einer hohen Produktion an Mesonen begleitet, sodaß die Antiprotonen aus einem hohen mesonischen Untergrund herausgefiltert werden müssen.

Bei konventionellen $\bar{p}$-Strahlen werden Protonen beschleunigt und auf ein Produktionstarget geschossen. Die dabei entstehenden Teilchen werden aufgesammelt und die Antiprotonen mit Hilfe von Energie- und Impulsanalyse und Cerenkov- bzw. Flugzeitmessungen herausgefiltert. An solchen Strahlen durchgeführte Experimente mit antiprotonischen Atomen weisen typische Erzeugungsraten von einigen hundert pro Sekunde auf. Seit einigen Monaten steht am europäischen Kernforschungszentrum CERN ein neuer Beschleunigerring zur Verfügung, der eine Erhöhung der Ereignisraten um einige Größenordnungen erlaubt (siehe Abschnitt 1.5). Das im folgenden beschriebene Meßsystem ist auf erste Messungen an diesem neuen Gerät zugeschnitten.

1.2 Formation antiprotonischer Atome

Die Antiprotonen des Strahles werden mit Hilfe eines Moderators abgebremst und in einem Target gestoppt. Das Antiproton geht dabei elektromagnetische Wechselwirkungen mit den Elektronen des Moderator- bzw. Targetmaterials ein. Ist die Geschwindigkeit des Antiprotons mit der der Elektronen vergleichbar, so kann es vom Coulombfeld des Targetkernes eingefangen werden. Dabei wird ein Elektron aus der Elektronenhülle des Targetkernes entfernt und die nötige Energie zur Bindung des Antiprotons am Kern benutzt. Nach dem Einfangprozess, dessen Einzelheiten wesentlich von der Ordnungszahl des Targetkernes, der Bindungsenergie der Elektronen und dem Atomradius abhängen, findet man das Antiproton in hoch angeregten Zuständen mit hohen Quantenzahlen n,l,j wieder. Durch Auger- und Röntgenübergänge geht das Antiproton schnell in stärker gebundene Zustände über. Es sind diese Röntgenübergänge, die der Messung zugänglich sind und alle Informationen über antiprotonische Atome liefern. Wegen der hohen Masse des Antiprotons sind die Energien der Bohr'schen Bahnen wesentlich größer als die der Elektronen (die Übergangsenergien liegen im Röntgenbereich, bis zu 500 keV) und die Radien der Bohr'schen Bahnen entsprechend kleiner, wie aus Formel (1) und (2) ersichtlich ist.

1.

$$E_n = -\frac{\mu c^2}{2}\frac{(\alpha Z)^2}{n^2} \qquad \mu = \frac{m \cdot M}{m+M} = \text{reduzierte Masse des } \bar{p}$$

2.

$$r_n = \frac{\hbar}{\mu c}\frac{n^2}{\alpha Z}$$

M = Masse des Kernes
m = 938.2796 MeV/c^2 = $\bar{p}$-Masse
α = 1/137 Feinstrukturkonstante
$\hbar$ = $6.58 \cdot 10^{-22}$ MeVs Planck-Wirkungsquantum
c = $3 \cdot 10^8$ m/s Vakuumlichtgeschwindigkeit

Schon ab ca. n=40 liegt die Bahn des Antiprotons näher am Kern als die innersten Bahnen der Hüllenelektronen, so daß Augerübergänge immer unwahrscheinlicher werden. Die Übergangsenergie wird nun durch Röntgenquanten abgegeben, deren Energie sich in erster Näherung aus (1) errechen läßt. Für genauere Rechnungen müssen jedoch Effekte der Quantenelektrodynamik mitberücksichtigt werden. Kommt das Antiproton sehr nahe an den Kern, so überlappt seine Wellenfunktion mit der des Kernes. Das Antiproton tritt daher auch in starke Wechselwirkung mit dem Kern, was eine Energieverschiebung des Niveaus bewirkt, verglichen mit der Energie, welche durch rein elektromagnetische Wechselwirkung erwartet wird. Außerdem kann das Antiproton nun mit einem Nukleon annihilieren, so daß dadurch die Lebensdauer des Zustandes verringert wird und dadurch die Energieunschärfe des Zustandes zunimmt. In gleicher Weise nimmt die Energieunschärfe des emittierten Röntgenquantes zu, sodaß die gemessene Röntgenlinie verbreitert ist. Die Prozesse beim Herunterkaskadieren eines Antiprotons sind für den Fall des $\bar{p}$-^{16}O in Abb. 1 dargestellt.

1.3 Meßbare Größen

Aus genauen Messungen antiprotonischer Röntgenlinien können z.B. die Masse und das magnetische Moment des Antiprotons mit hoher Präzision bestimmt werden. Von besonderem Interesse sind die starken Wechselwirkungseffekte. Sie können nur am letzten beobachtbaren Übergang gemessen werden und hängen von der Wechselwirkung des Antiprotons mit einem Nukleon und von der Verteilung der Nukleonen im Kern ab. Beides sind physikalisch sehr interessante Aspekte. Um die beiden Effekte zu trennen werden Isotopenmessungen gemacht z.B. ^{16}O, ^{17}O, ^{18}O und die Röntgenspektren miteinander verglichen. Gemessen wird die Energieverschiebung der letzten beobachtbaren Linie, die Linienform, aus der die Lorentzbreite des Übergangs bestimmt wird, sowie die durch die starke Wechselwirkung reduzierte Intensität der letzten beobachtbaren Linie.

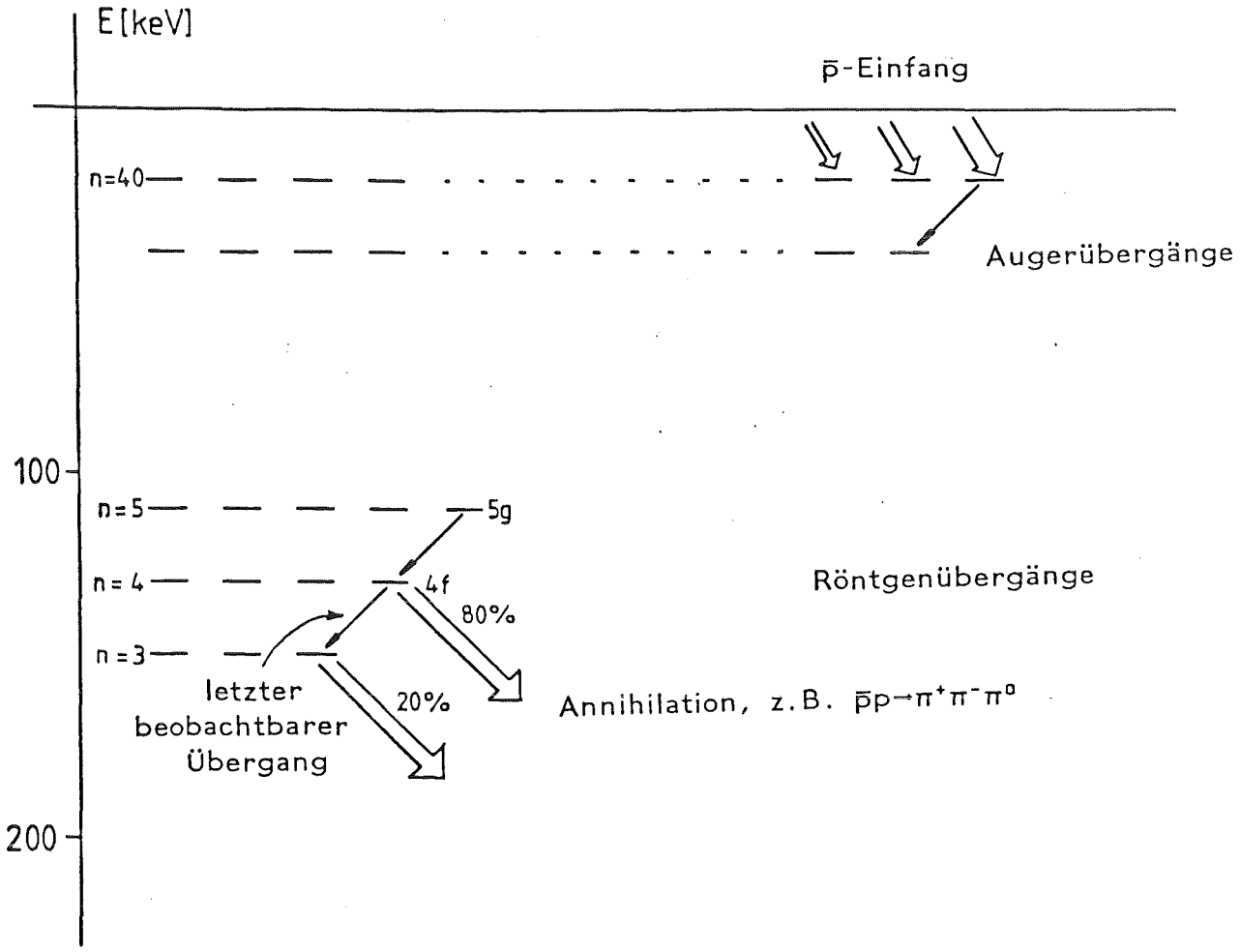


Abbildung 1: $\bar{p}$-^{16}O Energie Schema

Für mehr Informationen über die physikalische Motivation der Experimente an antiprotonischen Atomen siehe [KOC76],[POT82].

1.4 Das Experiment

Unser Experiment (PS-176) [AD180] beschäftigt sich mit der Messung starker Wechselwirkungseffekte an leichten antiprotonischen Atomen. Es befindet sich am Low Energy Antiproton Ring (LEAR) von CERN in Genf (LEAR ist im nächsten Abschnitt näher beschrieben). Bei diesem Experiment werden die Antiprotonen von LEAR, die beim Austritt aus dem Strahlrohr einen Impuls von 300 MeV/c haben, durch einen Moderator auf so kleine Geschwindigkeiten herabgebremst, daß sie im Target stoppen. Die Targets haben eine eine typische Dicke von 50-100 mg/cm^2. Der einlaufende Strahl wird durch ein Teleskop aus Szintillationszählern S1-S4 erfaßt (siehe Abb. 2). Die Targets sitzen auf einer bewegbaren Targetleiter und können mittels Computerkontrolle in sehr kurzer Zeit gewechselt werden. Die Röntgenquanten werden mit Hilfe von Halbleiterdetektoren, welche den gesamten Energiebereich der erwarteten Röntgenquanten abdecken, gemessen. Die Halbleiterdetekoren werden wegen ihrer hohen Energieauflösung benutzt.

Eines der Hauptprobleme des Experimentes ist die Unterdrückung von Untergrundereignissen, die durch Comptonstreuung hochenergetischer γ-Strahlen im Detektor hervorgerufen werden. Auch Ereignisse von geladenen Teilchen müssen unterdrückt werden. Da die Untergrundereignisse im allgemeinen ein anderes zeitliches Verhalten relativ zum einlaufenden Antiproton zeigen, läßt sich gegen diese Ereignisse diskriminieren, wenn nicht nur die Energie sondern auch das Zeitverhalten der Ereignisse gemessen wird.

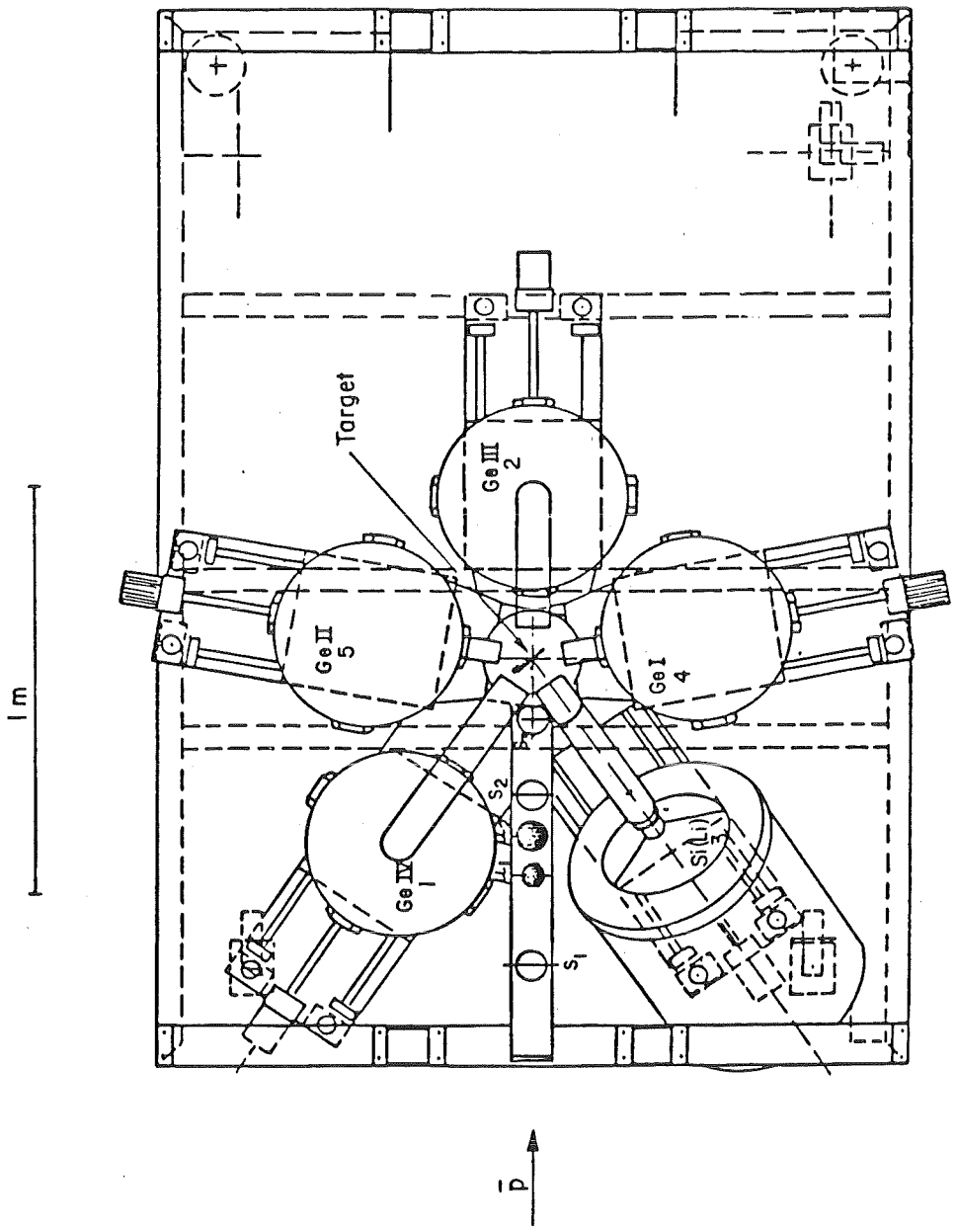


Abbildung 2: Der experimentelle Aufbau

1.5 Erzeugung von Antiprotonen bei LEAR

PS 176 gehört zu den ersten Experimenten, die am Low Energy Antiproton Ring im CERN [HAR79] durchgeführt werden. LEAR ist eine Maschine, die speziell für die Produktion von Antiprotonen niedriger Energie gebaut wurde. Das CERN Proton Synchrotron (PS) beschießt ein Wolframtarget mit Protonen von 25 GeV. Die dabei entstehenden Antiprotonen werden vom Antiproton Akkumulator (AA) aufgesammelt und gespeichert. Die produzierten Antiprotonen haben stark unterschiedliche Impulse und Einschußwinkel in den AA. Sie müssen phasenraumgekühlt werden, um nicht beim Speichern verloren zu gehen. Die Kühlung geschieht durch elektrische Felder deren Feldstärke von der mittleren Strahlposition abhängt. Teilchen, deren Bahn relativ zur Idealbahn zu weit außen liegt, werden von diesen Feldern stärker nach innen abgelenkt, während Teilchenbahnen, die zu weit innen verlaufen, mehr nach außen gedrängt werden (stochastische Kühlung). Die Antiprotonen aus dem AA werden zum PS transferiert, dort auf den Impuls von 600 MeV/c abgebremst und dann in LEAR injiziert und dort gespeichert. Eine neue Extraktionsmethode (stochastische Extraktion), bei der der Extrahierungsprozess durch Hochfrequenzrauschen herbeigeführt wird, erlaubt eine extrem lange Spilldauer (Zeit, während der Strahl für die Experimente zur Verfügung steht). LEAR kann die Experimente, nach einer Füllung mit Antiprotonen, während ca. 1 Stunde mit Strahl versorgen. Danach ensteht eine Strahlpause von ca. 10 Minuten, in der weitere Antiprotonen vom AA nach LEAR transferiert, bevor der Strahl für eine weitere Stunde zur Verfügung steht. Mit Kühlung und stochastischer Extraktion ist es möglich einen $\bar{p}$-Strahl von 10^6 Teilchen/s herzustellen, der beinahe Gleichstromcharakteristik hat und nahezu keine anderen Teilchen enthält. Die Strahldaten LEAR's sind in Tabelle 1 zusammengefaßt. Der LEAR Strahl wird unter drei Strahlzweigen verteilt, sodaß für drei Experimente je ca. $3 \cdot 10^5$ $\bar{p}$/s zur Verfügung stehen.

Tabelle 1: Daten des LEAR Strahls

- Strahlstruktur:
  Spill: ca. 1 Stunde
  Strahlpause: ca. 10 min.
- Teilchenrate: 10^6 $\bar{p}$/s
- Strahlgröße:

	minimaler Fokus	maximaler Fokus
Fokallänge	2.0 m	9.5 m
vertikal	2 x 0.6 mm	2 x 4 mm
horizontal	2 x 0.6 mm	2 x 4 mm

- Divergenz:

horizontal	± 8.3 mrad	±1.2 mrad
vertikal	±33.3 mrad	±5.0 mrad

- Emittanz:

horizontal	< 2 π mm mrad
vertikal	≈15-20 π mm mrad

- Impulsunschärfe: $\Delta p/p = 2\text{-}3 \cdot 10^{-3}$

1.6 Anforderungen des Experimentes an das Datenerfassungssystem

Mit einer Antiprotonenrate von $3 \cdot 10^5$ $\bar{p}$/s und unter Berücksichtigung des Raumwinkels, der durch die Detektoren abgedeckt wird, der Nachweiswahrscheinlichkeit der Detektoren, der Anzahl der Röntgenquanten pro

Antiproton, der Untergrundrate etc. kann eine maximale Ereignisrate von 10^4 Ereignissen/s abgeschätzt werden. Während des Spills von einer Stunde kann jeder der Halbleiterdetektoren je eine Energie- und eine Zeitinformation pro Ereignis liefern. Da dies die einzigen Daten sind, die aufgenommen werden müssen, kann die gesamte Information über das Ereignis in zweidimensionalen Spektren abgespeichert werden, falls nur ein Detektor gefeuert hat. Diese Datenstruktur ist typisch für alle Spektroskopie-Experimente, die bisher gewöhlich von Vielkanalanalysatoren ausgelesen wurden. Möchte man zweidimensionale Spektren von sehr hoher Präzision aufnehmen, so ist dies jedoch mit Vielkanalanalysatoren wegen des hohen Speicherplatzbedarfes nicht mehr möglich. Ein zweidimensionales Spektrum mit 4096 Energiekanälen und 32 Zeitkanälen benötigt schon 512 kBytes, wenn mehr als 65536 (64K) Ereignisse in einem Kanal erlaubt sein sollen.

Es ist auch möglich, daß 2 Detektoren je ein Röntgenquant von derselben Kaskade sehen. Solche Koinzidenzereignisse sind jedoch wesentlich seltener. Auch sie können mit Vielkanalanalysatoren nur in sehr beschränkter Weise behandelt werden.

Die Anforderungen des Experimentes an das Datenerfassungssystem sind in Tabelle 2 zusammengestellt.

Tabelle 2: Anforderungen an das Datenerfassungssystem

- Hohe Datenrate (10^4 Ereignisse/s) bei kleiner Wortlänge der Ereignisse
- Besondere Strahlstruktur (ca. 1 Stunde Strahl, ca. 10 Minuten Pause)
- Koinzidenzereignisse sollen auswertbar sein
- Hohe Präzision der Spektren und deshalb grosser Speicherplatzbedarf

Ein konventionelles Datenerfassungssystem, bei dem das Experiment via CAMAC[1] an einen Minicomputer angeschlossen wird, kann die genannten Anforderungen nicht erfüllen. Versucht man die Ereignisse einzeln auf Magnetband zu schreiben, so ist das System viel zu langsam. Will man dagegen die Spektren im Speicher des Minicomputers halten, so reicht der zur Verfügung stehende Speicherplatz nicht aus. Eine Erzeugung der Spektren auf Platte ist wiederum aus Geschwindigkeitsgründen nicht möglich.

Eine Auslese des Experimentes mit Vielkanalanalysatoren ist nur dann möglich, wenn auf Koinzidenzen verzichtet wird und die Anforderungen an die Energie- und Zeitauflösung nicht sehr hochgeschraubt werden (typisch 64 Energie- und 16 Zeitkanäle).

1.7 Prinzipieller Aufbau des Datenerfassungssystems

Die Schwierigkeiten, auf die man bei der Auslese mit Minicomputer stößt, sind, wie besprochen, Speicherplatzprobleme und Geschwindigkeitsprobleme. Die maximale Anzahl von Speicherzellen, die angesprochen werden können, wird durch die Anzahl der Adresslinien des Prozessors festgelegt. Bei den heutigen 16 bit Minicomputern sind dies 16 Adresslinien, was 64 kBytes Speicher anzusprechen erlaubt.

Die Geschwindigkeit des Minicomputers hängt sehr wesentlich von seinem Betriebssystem ab. Es ist selbstverständlich, daß ein in Fortran geschriebenes Datenausleseprogramm, das unter einem komfortablen Mehrbenutzersystem läuft, langsamer ist als ein Assemblerprogram, welches alleine in der Maschine existiert. Man möchte allerdings auch nicht gerne auf höhere Programmiersprachen und auf den Komfort des Betriebssystemes beim Ansprechen der Peripherie verzichten. Eine Lösung des Problems bieten die neuen 16 Bit Mikroprozessorsysteme an, die seit wenigen Jahren auf dem Markt

---

[1] Für eine Erklärung von CAMAC und der im folgenden benutzten Fachbegriffe siehe Kapitel 2.

sind. Der MC 68000 Mikroprozessor, der in dem besprochenen System verwendet wird, hat 24 Adresslinien und erlaubt daher 16 MBytes direkt anzusprechen, was für die experimentellen Spektren vollkommen ausreichend ist. Die Rechenleistung dieses Prozessors steht der eines Minicomputers nicht nach, solange keine Gleitkomma Arithmetik verlangt wird. Da bis heute keine brauchbare Echtzeitsoftware auf diesen Systemen existiert, und auch die angebotene Peripherie noch sehr beschränkt ist, liegt der Gedanke nahe, ein solches Mikroprozessorsystem mit einem Minicomputer zu koppeln. Der Mikroprozessor wird dann nur zum Auslesen der Daten und zum Generieren der Spektren benutzt, da hierzu weder ein Betriebssystem noch Peripherie notwendig sind, während der Minicomputer die fertigen Spektren übernimmt, um sie auf Magnetband zu schreiben und sie zur graphische Darstellung aufzuarbeiten. Auf diese Art werden die Schwächen der Maschinen geschickt umgangen, während ihre Stärken voll ausgenutzt werden. Das Blockschaltbild eines solchen 2 Prozessorsystemes ist in Abb. 3 gezeigt.

Es lassen sich bei der Datennahme also drei Phasen unterscheiden:

- die Auslesephase
- die Transferphase
- und die Verarbeitungsphase

Während des Spills von ca. 1 Stunde nimmt der Mikrocomputer die experimentellen Daten auf und erzeugt in seinem Speicher die zweidimensionalen Spektren. Das Ausleseprogramm ist ein sehr kurzes Assemblerprogramm und deshalb sehr schnell. Es ist auf kein unterstützendes Betriebssystem angewiesen, da es weder komplizierte mathematische Funktionen ausführen muß (Gleitkomma Arithmetik wird ebenfalls nicht benötigt) noch auf Peripheriegeräte zugreifen muß. Der große Adressraum des Mikrocomputers erlaubt ihm die benötigten großen Spektren in seinem Speicher aufzubauen, ohne auf eine Platte ausweichen zu müssen.

Die Transferphase wird gestartet sobald der Spill zu Ende ist. Während der Strahlpause müssen alle aufgesammelten Daten zum Minicomputer transferiert

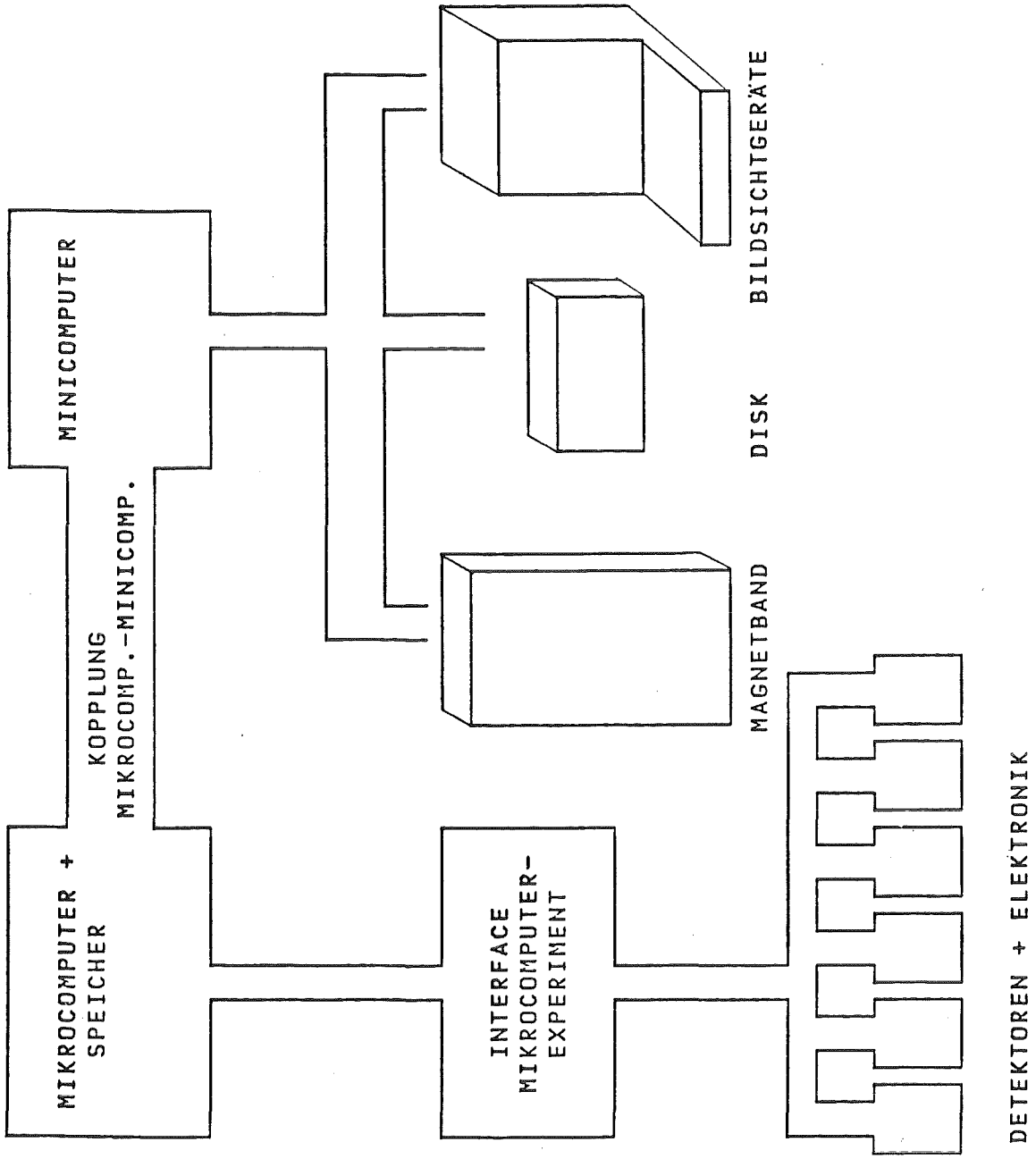


Abbildung 3: Prinzip des 2 Prozessorsystemes

werden, um dort auf Magnetband geschrieben und zur graphischen Darstellung aufbereitet zu werden.

Sobald alle Daten transferiert sind startet die Verarbeitungsphase im Minicomputer. Gleichzeitig jedoch ist der Mikrocomputer frei, neue Daten aufzunehmen, sodaß die Aufnamephase der Daten eines Spills und die Verarbeitungsphase der Daten des vorhergehenden Spills gleichzeitig ausgeführt werden.

Zwei grundsätzliche Probleme sind bei einem solchen Datenerfassungssystem zu lösen:

- Der Anschluß des Experimentes an den Mikrocomputer
- Die Kopplung des Mikrocomputers an den Minicomputer

1.8 Interface Experiment-Mikrocomputer

Wegen der großen Verbreitung von CAMAC existiert beinahe für jede erdenkliche Funktion das entsprechende CAMAC Modul. Mit einem Interface (Anpassungsschaltung), das dem Mikroprozessor erlaubt als CAMAC Controller zu fungieren und CAMAC Befehle abzusetzen, kann man alle vom Experiment anfallenden Daten vom Mikrocomputer auslesen lassen. Wegen der Vielfältigkeit von CAMAC können außer dem Erfassen der Energien (über Analog zu Digitalwandler) und der Zeiten (über Zeit zu Digitalwandler) auch noch beliebig viele andere Aufgaben ausgeführt werden, die bei einer eventuellen Erweiterung des Experimentes in der Zukunft interessant werden können. Dies macht das System sehr flexibel. Das Mikrocomputer-CAMAC Interface wurde als Lösung gewählt, weil es wesentlich einfacher erscheint ein Interface an CAMAC zu entwickeln als für jede neue Funktion eine neue Anpassungsschaltung an den Mikroprozessorbus entwerfen zu müssen. Das CAMAC Interface ist in Kapitel 4 beschrieben.

1.9 Kopplung Minicomputer-Mikrocomputer

Um sich genauere Vorstellungen über die Kopplung der beiden Maschinen machen zu können, muß man sich zunächst über die gewünschten Funktionen der Kopplung klar werden.

Man möchte vom Minicomputer aus

- den Mikrocomputerspeicher lesen und beschreiben können
- und dem Mikrocomputer Befehle erteilen können.

Außerdem soll

- der Mikroprozessor Interrupts an den Minicomputer absetzen können.

Das koppelnde Interface muß daher einen DMA (Direkt Memory Access) Mechanismus beinhalten, der den Mikroprozessor von seinem Bus abhängt, solange der Minicomputer auf den Speicher zugreift. Das Absetzen von Befehlen an den Mikrocomputer läßt sich am einfachsten durch Interrupts realisieren. Der Minicomputer muß daher imstande sein, den Mikroprozessor zu unterbrechen. Wenn der Mikroprozessor die Aufmerksamkeit des Minicomputers benötigt, weil z.B. ein Befehl fertig abgearbeitet ist, so geschieht auch dies am einfachsten durch Interrupts (hier Unterbrechen des Minicomputers). All diese Funktionen können wiederum durch ein speziell für diesen Zweck entwickeltes CAMAC Modul durchgeführt werden, da ein CAMAC Interface für den Minirechner existiert. Wie dieses CAMAC Modul im einzelnen funktioniert, wird in Kapitel 5 beschrieben.

Wie wir gesehen haben kann sowohl das Interface Mikroprozessor-CAMAC als auch die Kopplung zwischen Mini- und Mikrorechner als CAMAC Einheit implementiert werden. Es liegt daher auf der Hand, ein CAMAC Modul zu entwickeln, das beide Funktionen ausführt. Diese CAMAC Einheit wurde in unserer Gruppe entwickelt und CAMPORT benannt [MEY83]. Ein Blockschaltbild des gesamten Datenerfassungssystemes ist in Abb. 4 gezeigt. Auch der

Datenfluß während der Datennahme und während des Transfers der Spektren zum PDP-11 Minicomputer sind darin dargestellt.

Zur Steuerung von CAMPORT ist eine sehr komplexe zeitliche Abfolge von Signalen notwendig. Der CAMPORT Controller, der für die Generierung aller Steuersignale verantwortlich ist, wird in Kapitel 6 beschrieben.

Da CAMPORT zu einer Zeit entstand als es nur eine Karte mit dem MC 68000 Mikroprozessor zu kaufen gab: das 68K Design Modul (KDM Karte), wurde CAMPORT auf diese Karte zugeschnitten. Die KDM Karte ist kompatibel mit Motorola's 8 Bit Entwicklungssystemen und erlaubt die Entwicklung von Programmen für den MC 68000 mit Hilfe von 8 Bit Prozessoren. Um die volle Kapazität des MC 68000 auszunutzen (was mit dem Bus des KDM Moduls nicht möglich ist, da es sich dabei um eine Erweiterung eines einfachen 8 Bit Busses handelt) wurde von mehreren Firmen ein neues Bussystem entwickelt: der VME Bus. Ein Busadapter, der in Kapitel 7 beschrieben ist, erlaubt CAMPORT auch mit diesem Bus zusammenzuarbeiten.

Selbstverständlich werden auch Programme für die Auslese und vor allem für die Kommunikation zwischen den beiden Maschinen benötigt. Die gesamte Software wird in Kapitel 8 erklärt.

In Kapitel 10 werden Erweiterungen, die in der Zukunft geplant sind, beschrieben.

Kap. 11 fällt etwas aus dem Rahmen dieser Arbeit. Darin wird nämlich gezeigt, wie die Hardware des Systemes auch für völlig andere Anwendungen als die Emulation (Simulierung) von Vielkanalanalysatoren eingesetzt werden kann. Es wird ein Beispiel gezeigt, in dem das Mikroprozessorsystem als schneller Datenpuffer zur Auslese einer Vieldrahtkammer benutzt wird.

Im Anhang werden noch 2 CAMAC Einheiten beschrieben, die für das System entwickelt wurden, um die maximale Verarbeitungsgeschwindigkeit zu erreichen.

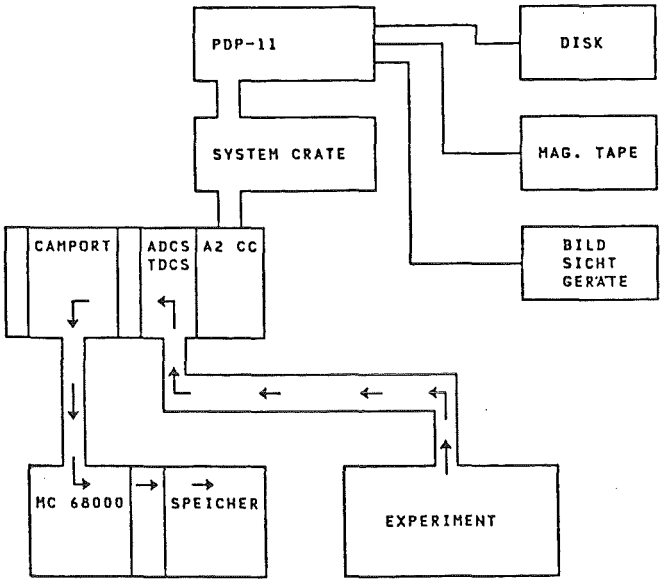


Abb. 4a Datenfluß während des LEAR Spills

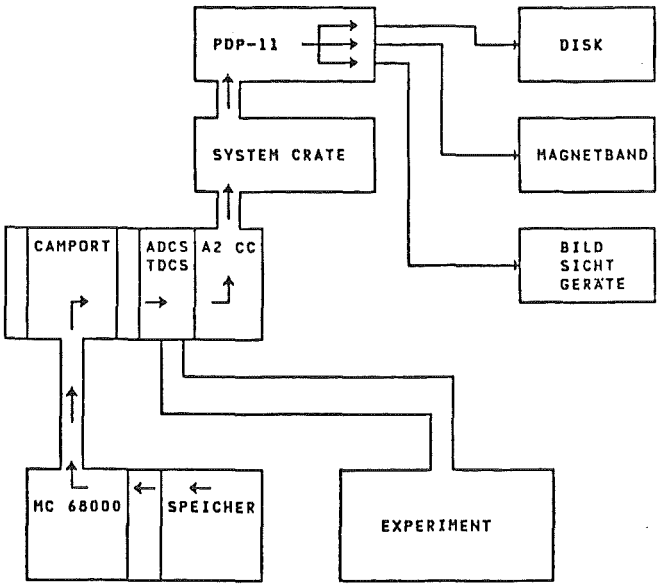


Abb. 4b Datenfluß während des Transfers der Spektren

Abbildung 4: Blockschaltbild des Datenerfassungssystems

KAPITEL 2

DEFINITION DER FACHBEGRIFFE UND EINFÜHRUNG VON CAMAC

In diesem Kapitel werden einige Fachbegriffe eingeführt, welche im folgenden benötigt werden. Eine vollständige Abhandlung ist im Umfang dieser Arbeit natürlich nicht möglich. Dafür wird auf die Fachliteratur verwiesen. Desweiteren wird eine Einführung in CAMAC gegeben, in der im wesentlichen auf den CAMAC Datenweg eingegangen wird. Der "Branchhighway" und das "System Crate" werden nur sehr flüchtig gestreift, da sie für das Verständnis der folgenden Kapitel nicht von Bedeutung sind.

2.1 Bus und Adressierung

Unter einem Bus versteht man alle benötigten elektrischen Verbindungsleitungen, die einen ordnungsgemäßen Informationsaustausch zwischen einer (oder mehreren) kontrollierenden Einheiten (Busmaster) und mehreren ausführenden Einheiten (Slaves) gewährleisten. Abb. 5 zeigt eine typische Busstruktur.

Die Buslinien werden dabei üblicherweise in 3 Kategorien zusammengefaßt:

- den Datenlinien (Datenbus)
- den Adresslinien (Adressbus)
- den Kontrollinien (Kontrollbus).

Der Austausch von Informationen zwischen Busmaster und Slaves geschieht über den Datenbus. Da jedoch mehrere Slave Einheiten am Bus angeschlossen sein können und jede Slave Einheit über mehrere Informationseinheiten verfügen kann, muß der Busmaster die Möglichkeit haben, die gewünschte Information anzuwählen (adressieren). Der gesamte vom Busmaster adressierbare Bereich wird als sein "Adressraum" bezeichnet.

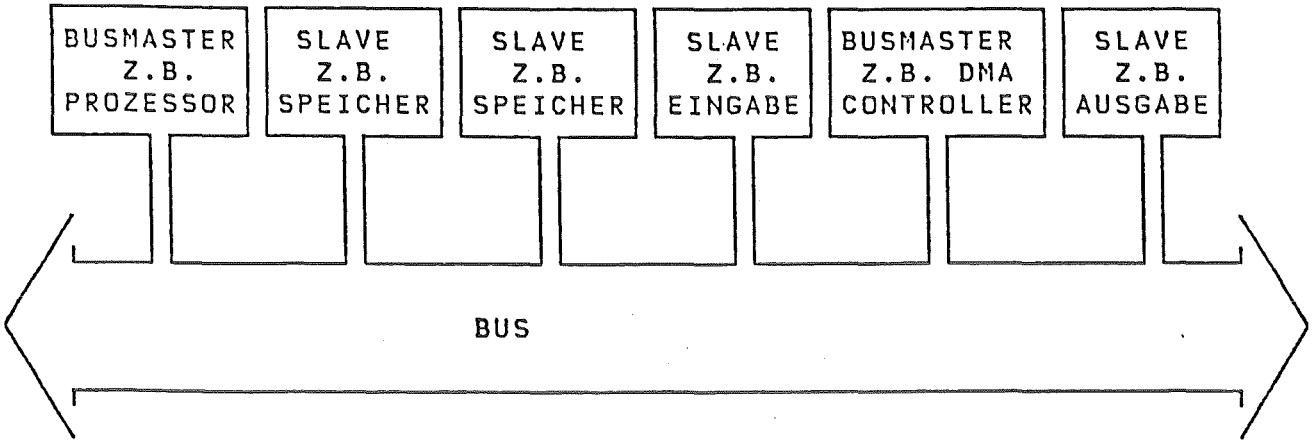


Abbildung 5: Typische Busstruktur

Die am meisten benutzte Art der Adressierung besteht darin, die Adresse in binär kodierter Form auf den Adressbus zu geben. Jede Slave Einheit muß dann feststellen ob sie angesprochen wurde (dekodieren ihrer Basisadresse) und falls ja, die gewünschte Information mit Hilfe der Adresse heraussuchen und zur Verfügung stellen. Fast alle Computer addressieren ihren Speicher und ihre I/O (Input/Output, Ein-Ausgabe) auf diese Weise. Es gibt jedoch noch eine Fülle anderer Adressierungsarten, hauptsächlich bei Instrumentierungsbussen, von denen nur die "geographische Adressierung" hier erklärt werden soll, da diese in CAMAC eingesetzt wird. Bei der geographischen Adressierung wird eine Einschubnummer angesprochen, und die Einheit, welche sich an der adressierten Stelle befindet muß die geforderte Antwort geben.

Der Kontrollbus besteht aus Signalleitungen, welche Kontroll- oder Timinginformationen übertragen. Die Schreib/Leseleitung (WRITE*), Handshakeleitungen (Eigensynchronisierungsleitungen) zur Synchronisation der

Datentransfers, Interruptleitungen und DMA Leitungen werden zum Kontrollbus gezählt.

Die Informationsübertragung zwischen Busmaster und Slave kann entweder synchron zu einem Taktsignal (synchroner Bus) oder asynchron mit Hilfe von Handshakesignalen durchgeführt werden (asynchroner Bus).

2.2 Interrupt

Die Steuerung von I/O Einheiten in einem Computersystem kann auf verschiedene Weise erfolgen. Soll z.B. ein Buchstabe von einem Terminal gelesen werden, so kann der Prozessor das Terminal solange abfragen, bis der Buchstabe vorliegt (Polling). Eine Menge Rechenzeit wird jedoch hierbei durch Warten verschwendet. Aus diesem Grund haben viele Busse extra Kontrolleitungen, welche der I/O Einheit erlauben, den Prozessor zu unterbrechen (Interruptleitungen). Sobald das Terminal einen Buchstaben weitergeben will, gibt es ein Signal auf die Interruptleitung. Der Prozessor unterbricht daraufhin sein gerade in Ausführung befindliches Programm und speichert alle Statusinformationen ab, die später benötigt werden, um das unterbrochene Programm fortsetzen zu können. Der Prozessor startet danach eine Interruptservice Routine, die den Buchstaben vom Terminal liest und verarbeitet. Eine spezielle Instruktion (RTE, Return From Exception im Falle des MC 68000) signalisiert das Ende der Interruptservice Routine. Die gespeicherte Statusinformation wird dann zurückgelesen und der Prozessor führt die Bearbeitung des unterbrochenen Programmes fort.

Sobald mehrere I/O Einheiten Interrupts absetzen können, muß der Prozessor in der Lage sein zu unterscheiden, welche Interruptservice Routine ausgeführt werden soll. Auch hier gibt es verschiedene Lösungsmöglichkeiten, von denen nur die vektorisierten Interrupts erklärt werden, da sowohl die PDP-11 als auch der MC 68000 diese Interruptart benutzen. Bei vektorisierten Interrupts bekommt jede I/O Einheit eine Nummer, die Interruptvektornummer. Sobald der Prozessor seine Statusinformation abgespeichert hat fragt er diese Nummer vom

unterbrechenden I/O Kanal ab und verwendet sie als Zeiger in der Interruptvektortabelle, in der die Startadressen sämtlicher Interruptservice Routinen abgelegt sind. Bei mehreren Interruptquellen muß außerdem eine Prioritätenliste aufgestellt werden, für den Fall, daß zwei Interrupts gleichzeitig anstehen. Dies kann dadurch geschehen, daß mehrere Interrupt Leitungen benutzt werden, wobei jeder Interrupt Leitung eine andere Priorität (Interruptebene) zugeordnet wird. Oft wird auch ein Signal (Interrupt Acknowledge, IACK) durch alle Interruptquellen geschleust. Eine I/O Einheit, die IACK empfängt, aber keinen Interrupt generiert hat, gibt das Signal an die nächste Einheit der Kette weiter. Die erste Einheit, welche IACK empfängt und einen Interrupt erzeugt hat, blockt das Signal für alle weiteren I/O Einheiten der Kette und stellt die Interruptvektornummer. Eine solche "Daisy Chain" ist in Abb. 6 gezeigt.

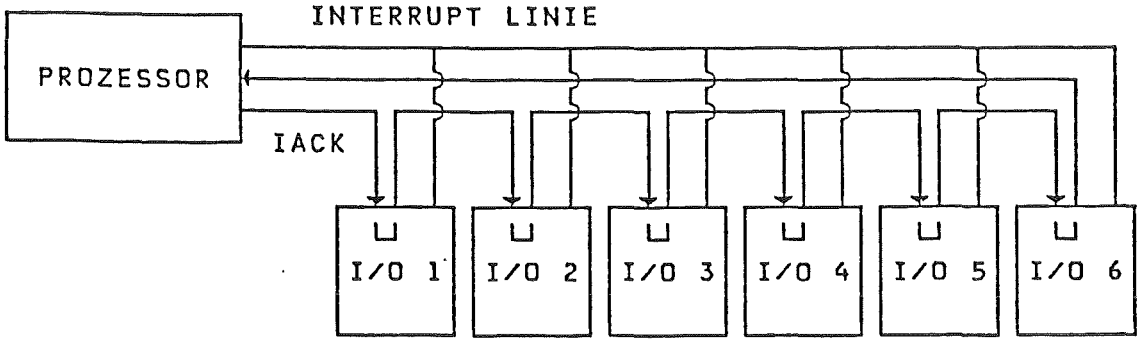


Abbildung 6: Daisy Chain

2.3 DMA

Bei einem Informationsaustausch zwischen zwei Slave Einheiten werden die Daten gewöhnlich zuerst von Slave 1 in ein Prozessorregister transferiert und dann von dort an Slave 2 weitergegeben. Findet ein solcher Transfer ohne Einfluß des Prozessors statt, so redet man von einem DMA (Direkt Memory Acess, Direkter

Speicherzugriff). Dazu muß ein anderer Busmaster, der DMA Controller, die notwendigen Adressignale erzeugen. Dies geschieht mit Hilfe des MAR (Memory Adress Register, Speicher Adress Register) im DMA Controller. Vor dem DMA wird das MAR mit der Startadresse des zu transferierenden Datenblockes (Sourceadresse) geladen. Der DMA Contoller inkrementiert das MAR nach jedem Worttransfer, sodaß die nächste Adresse erzeugt wird. Ein weiteres Register, das Word Count Register (Wortzähler), zählt die Anzahl der transferierten Worte. Dieses Register wird vor Ausführung des DMA mit der Länge des zu transferierenden Datenblockes geladen. Nach jedem Worttransfer wird das Register dekrementiert, und der DMA wird abgebrochen, sobald der Registerinhalt null wird. Man nennt diesen Datentransfer deshalb auch häufig "Blocktransfer". Die Zieladresse (Destinationadresse) ist gewöhnlich konstant (z.B ein Register im Interface zur Magnetplatte, bei einem Schreib-DMA vom Speicher auf die Platte).

Da es zu einem Buskonflikt kommen kann, wenn der DMA Controller und der Prozessor gleichzeitig versuchen auf den Bus zuzugreifen, muß, ähnlich wie bei den Interrupts, eine Prioritätenliste festgelegt werden. Der sogenannte Busarbiter (Vorrangsteuerung) entscheidet dabei, welcher der beiden Busmaster den Vorrang erhält. Wie bei den Interrupts werden entweder verschiedene Busanforderungsleitungen (Busrequest Linien) eingesetzt, wodurch man verschiedene "Busmastership Niveaus" erhält, oder die Priorität wird durch eine Daisy Chain festgelegt. Weitere Informationen zum Thema Bus, Interrupt, DMA finden sich in [DOB83].

2.4 Einführung in CAMAC

Die von uns benutzte CAMAC Konfiguration besteht aus einem System Crate (Crate = Überrahmen) der Firma Fisher, in dem sich das Interface zum PDP-11 Unibus (Bus der PDP-11), ein Interruptvektorgenerator, ein AMC DMA Controller (Autonomous Memory Channel) und ein CAMAC "Branchdriver" befindet. Der Branchdriver verbindet das PDP-11 Interface über einen speziellen Bus, den CAMAC "Branchhighway", mit maximal 7 CAMAC Crates. Jedes Crate

besitzt 25 Einschübe (Stationen), die über einen weiteren Bus, den CAMAC Datenweg, miteinander verbunden sind. Abb. 7 zeigt ein Blockschaltbild dieser CAMAC Konfiguration.

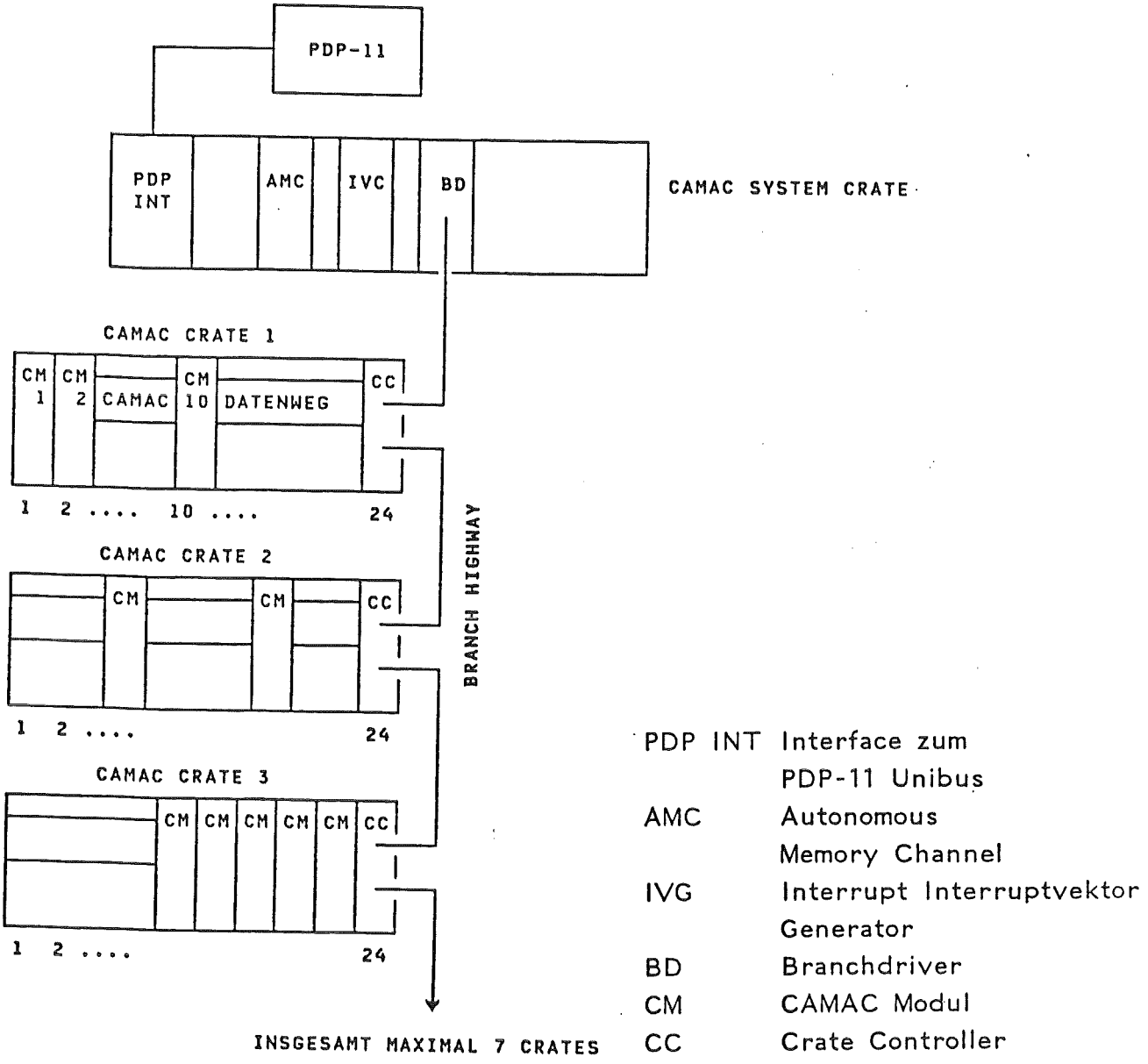


Abbildung 7: CAMAC Konfiguration

Die vom CAMAC Datenweg benutzte Adressierung ist eine Mischung aus geographischer Adressierung und binär kodierter Adressierung. Jede der Stationen, außer Station 25, wird mit Hilfe einer Stichleitung, der Stationsadresse N, angesprochen und ist daher geographisch adressiert. Alle diese Stichleitungen sind auf Station 25 geführt und können daher nur von der Einheit in dieser Station, dem Crate Controller, angesprochen werden. Der Crate Controller ist deshalb der einzige Busmaster des CAMAC Datenweges. Die Unteradressen einer CAMAC Einheit werden durch die Subadresse A bestimmt. Sie liegt in binär kodierter Form vor und muß von der CAMAC Einheit dekodiert werden.

Zum Transfer der Daten stehen je 24 Schreibleitungen W (Write lines) und 24 Leseleitungen R (Read lines) zur Verfügung. Die Informationsübertragung zwischen CAMAC Modul und Crate Controller wird mit Hilfe zweier Strobe Signale S1, S2 synchronisiert, wobei S1 für den Datentransfer und S2 zum Löschen von Registern benutzt wird.

Wird ein CAMAC Befehl an die Station N abgesetzt, so signalisiert das X Signal (X Response), ob sich in der angesprochenen Station überhaupt eine Einheit befindet. Das Q Signal (Q Response) meldet eine erfolgreiche Ausführung des Befehles.

5 Funktionscode Leitungen F wählen die gewünschte CAMAC Funktion an. Das CAMAC Modul dekodiert den Funktionscode (F-Code) auf diesen Leitungen und führt die entsprechende Aktion aus. Man unterscheidet dabei folgende Funktionsgruppen:

F0	- F7	Lesebefehle
F8	- F15	Steuerbefehle (z.B. Prüfen von Statusbits)
F16	- F23	Schreibbefehle
F24	- F31	Steuerbefehle (z.B. Sperren und Freigeben von Einheiten)

Bei den Steuerbefehlen handelt es sich um Aktionen, die den Datenbus des CAMAC Datenweges nicht benutzen. Die CAMAC Funktionen C und Z, welche

zur Initialisierung des CAMAC Crates benötigt werden, nehmen eine Sonderstellung ein. Sie werden nicht mittels F-Codes, sondern über die speziellen Datenwegleitungen C und Z erzeugt.

CAMAC Einheiten können mit Hilfe von "LAM" (Look at me) Signalen Interrupts erzeugen. Bei den LAMs handelt es sich, wie bei den N Signalen, um Stichleitungen, zu denen nur der Crate Controller Zugang hat. Die LAMs werden an das System Crate weitergegeben, wo der Interrupt erzeugt und der Interruptvektor generiert wird.

Sollen zu einem CAMAC Crate zwei Prozessoren Zugriff haben, so muß ein spezieller Crate Controller (A2 Crate Controller, A2 CC) verwendet werden, der auf seiner Rückseite über einen Stecker für Flachbandkabel verfügt. Über dieses Kabel, den "Auxiliary Bus", macht der zweite Prozessor dem A2 CC die Stationsadresse in binär kodierter Form zugänglich. Der A2 CC dekodiert die Stationsadresse und generiert das entsprechende N Signal. Abb. 8 zeigt die Adressierung einer CAMAC Einheit im Falle von CAMPORT als Interface zum zweiten Prozessor. Auch die LAMs der einzelnen Stationen werden im A2 CC kodiert und über diesen Stecker an den zweiten Prozessor weitergegeben. Auch ein einfacher Busarbiter befindet sich im A2 Controller, um einen Buskonflikt zu vermeiden, falls Prozessor 1 (über den Branchhighway) und Prozessor 2 (über den Auxiliary Bus) gleichzeitig versuchen, auf den CAMAC Datenweg zuzugreifen.

Ein zweiter Prozessor, der mit Hilfe des A2 CC CAMAC Befehle absetzen kann, wird auch als "Auxiliary Crate Controller" (ACC) bezeichnet. Weitere Informationen über CAMAC finden sich unter [EUR80a], [EUR80b], [EUR78].

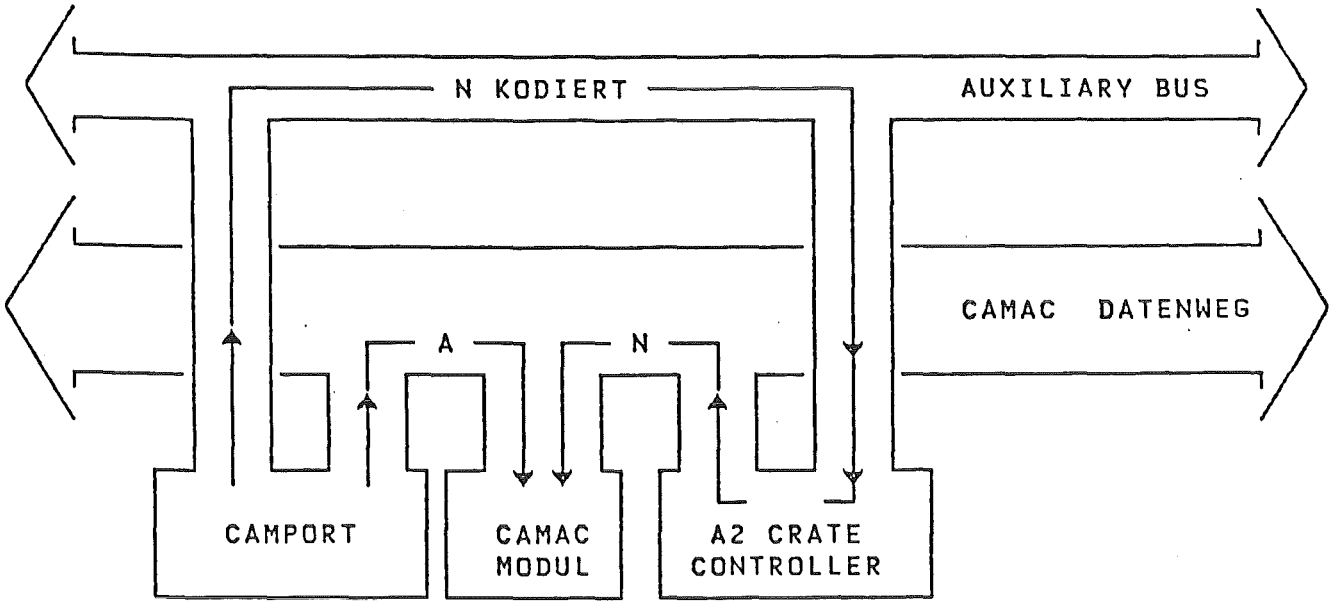


Abbildung 8: Adressierung einer CAMAC Einheit über den A2 CC

KAPITEL 3

CAMPORT, ÜBERBLICK ÜBER DIE VERSCHIEDENEN FUNKTIONSEINHEITEN

Wie bereits in der Einleitung erwähnt, bestehen bei einem 2 Prozessorsystem, in dem ein Prozessor die Auslese der experimentellen Daten und die Erzeugung von Spektren übernimmt, während der andere die aufgesammelten Spektren weiterverarbeitet, zwei prinzipielle Probleme:

- das Interface Auslesecomputer-CAMAC
- und die Kopplung Auslesecomputer-Verarbeitungscomputer

Beide Probleme werden von CAMPORT gelöst.

3.1 Die beiden CAMPORT Schnittstellen

CAMPORT kann über zwei verschiedene Ports (Schnittstellen) angesprochen werden. Zum einen erlaubt eine Flachbandkabelverbindung dem Mikrocomputer Befehle abzusetzen, welche von CAMPORT entweder als interne Funktionen interpretiert und ausgeführt werden, oder als CAMAC Funktionen erkannt und mit Hilfe eines A2 Crate Controllers an den CAMAC Datenweg weitergegeben werden. Kap. 4 beinhaltet eine detailierte Beschreibung der über diesen Port ansprechbaren CAMPORT Funktionen.

Der zweite Port CAMPORT's läßt sich über den CAMAC Datenweg ansprechen und ist daher über das System Crate mit der PDP-11 verbunden. Über diesen Port stellt CAMPORT der PDP-11 CAMAC Funktionen zur Verfügung, die ihr erlauben, auf den Mikrocomputerspeicher zuzugreifen (siehe Kap. 5). Mit Hilfe von speziell für diesen Zweck vorgesehenen Interruptbits, dem Kommunikationsinterrupt an den Mikroprozessor und dem Kommunikations-LAM (Communications LAM, CLAM) an die PDP-11, kann eine Kommunikation zwischen Mini-und Mikrocomputer etabliert werden.

3.2 Funktionsblöcke CAMPORT's

CAMPORT ist ein mikroprogrammgesteuertes Interface zwischen CAMAC und einem Mikrocomputerbus. Darin lassen sich, wie in jeder mikroprogrammierten Einheit, zwei Funktionsblöcke unterscheiden:

- das Steuerwerk und
- das Operationswerk.

Im Operationswerk werden die eigentlichen Funktionen durchgeführt, während das Steuerwerk nur die zum Funktionieren des Operationswerkes notwendigen Steuersignale erzeugt. Das Operationswerk CAMPORT's läßt sich, wie oben beschrieben, wiederum in zwei Blöcke unterteilen, von denen einer dem Mikroprozessor ermöglicht CAMAC Befehle zu erteilen, während der andere der PDP-11 Zugang zum Mikrocomputerspeicher verschafft. Abb. 9 zeigt diese Unterteilung.

3.3 Physikalischer Aufbau CAMPORT's

CAMPORT besteht aus zwei CAMAC Platinen, die in Wire Wrap Technik (Art der Verbindungstechnik) hergestellt wurden. CAMPORT belegt daher zwei CAMAC Steckplätze. Auf einer der beiden Platinen (der Sequenzerplatine) ist das gesamte Steuerwerk und Teile des Operationswerkes (die Logik für das Statusregister und das Interruptvektorregister) untergebracht, während auf der anderen Platine (der Interfaceplatine) der Hauptteil des Operationswerkes realisiert ist. Die Platinen sind so angeordnet, daß die Interfaceplatine an der rechten CAMPORT CAMAC Station und die Sequenzerplatine an der linken Station ansprechbar ist.

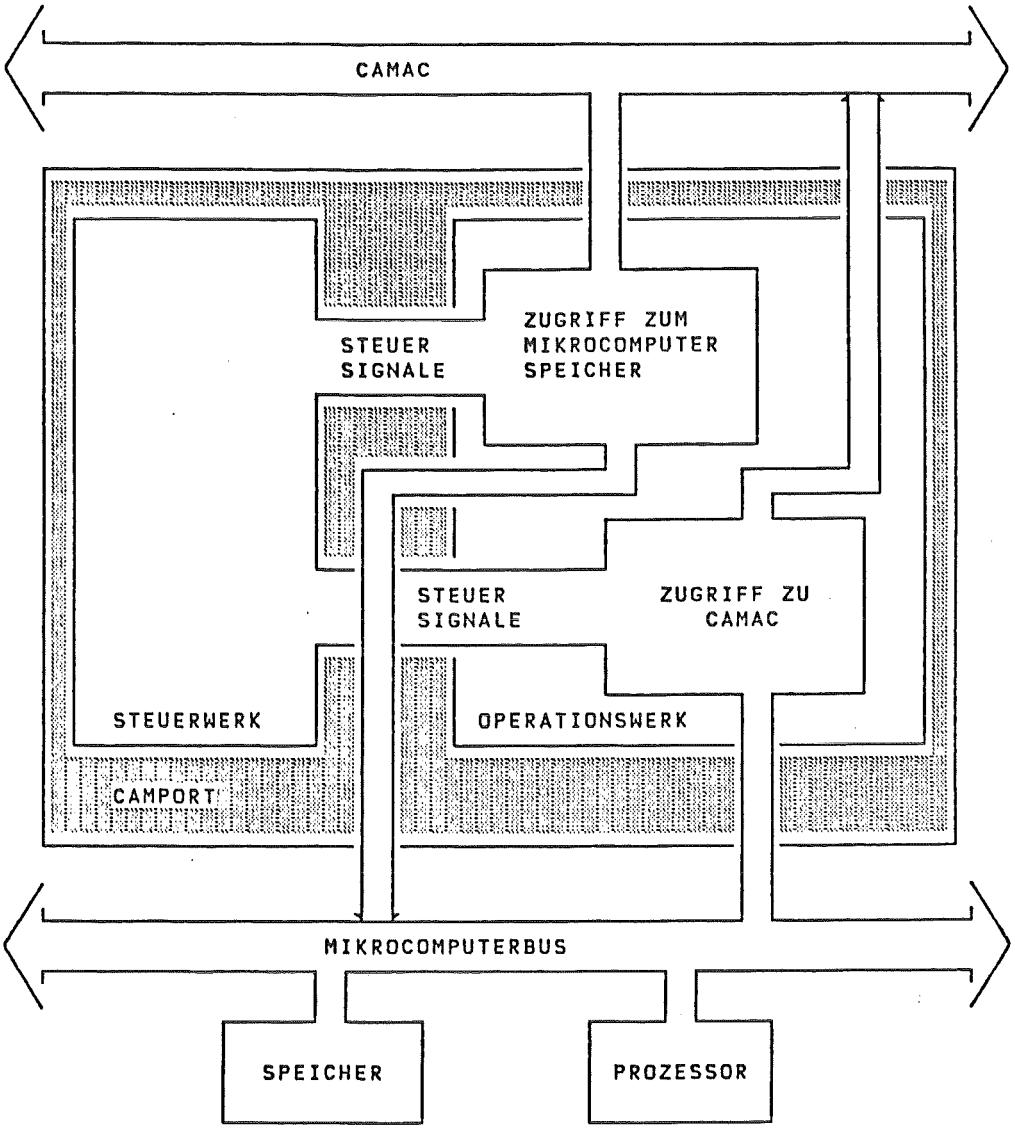


Abbildung 9: Funktionsblöcke CAMPORT's

KAPITEL 4

CAMPORT, ZUGRIFF DES MIKROPROZESSORS AUF CAMAC

In diesem Kapitel wird gezeigt, wie der Mikroprozessor mit Hilfe von CAMPORT CAMAC Befehle absetzen kann, und wie er somit die experimentellen Daten auslesen und die Spektren aufbauen kann. Es wird dabei ausschließlich auf die Hardware des Interfaces Mikrocomputer (KDM Karte) - CAMAC eingegangen. Die erforderliche Software ist in Kap. 8 dargelegt. Zunächst wird erklärt, wie CAMAC Einheiten vom Mikrocomputer adressiert werden und dann werden CAMPORT's interne Funktionen beschrieben, die vom Mikroprozessor angesprochen werden können. An Hand eines Beispieles wird die Durchführung eines CAMAC Befehles aufgezeigt.

4.1 Anschluß CAMPORT's an den Mikrocomputer

Die KDM Karte [MOT79a] kommuniziert mit der Außenwelt über einen Bus, dessen Eigenschaften in Tabelle 3 zusammengestellt sind. Der Bus der KDM Karte ist mittels zweier 50-poliger Flachbandkabel mit CAMPORT verbunden. Die Pinbelegungen dieser Kabel finden sich im Anhang D.

Um CAMAC Befehle absetzen zu können, muß der MC 68000 Zugang zu den CAMAC Adresslinien, d.h. zu den Stationslinien N und zu den Subadresslinien A, haben. Außerdem muß für Datentransfers der Datenbus des Mikrocomputers mit den CAMAC Schreib- und Leseleitungen verbunden sein und der MC 68000 muß den gewünschten Funktionscode auf die Funktionsleitungen des CAMAC Datenweges geben können. Letztlich müssen die CAMAC Strobe Signale S1 und S2 zur richtigen Zeit generiert werden.

- Asynchroner Bus
- 16 Adressleitungen (AM1-AM16) ermöglichen Zugriff zu 64K Worten (16 Bits)
- Byte Adressierung möglich über Datenstrobes LDS*,UDS*
- 16 Datenlinien (D0*-D15*)
- 4 Mhz System-Taktfrequenz
- 2 Interrupt Ebenen (I2, I7)
- DMA Möglichkeit durch

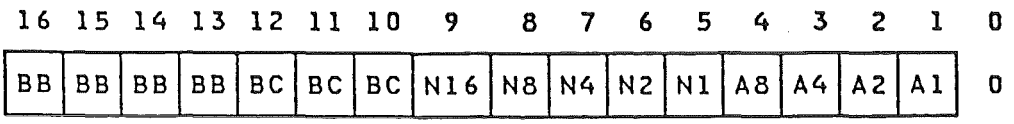
BR*	Busrequest Leitung
BG*	Busgrant Leitung
BGACK*	Bus Grant Acknowledge Leitung

* Signal ist aktiv im spannungslosen Zustand (active low)

Tabelle 3: Eigenschaften des KDM Busses

4.2 CAMAC Adressierung

Wie aus dem Blockschaltbild (Abb. 12) hervorgeht, sind die CAMAC Adressen direkt in den Adressraum der KDM Karte abgebildet (memory mapped). Die Adresslinien AM1-AM4 bilden die CAMAC Subadressen A während die Linien AM5-AM9 die Stationsadressen N generieren (siehe auch Abb. 10).
Die sieben höchst signifikanten Adressbits AM10-AM16 werden zur Dekodierung der CAMPORT Basisadresse benutzt. AM13-AM16 bestimmen die Lage von CAMAC im Adressraum des Mikrocomputers. AM10-AM12 hingegen sind durch Jumpers (Schaltstecker) in CAMPORT selektierbar und bilden eine "CAMPORT Nummer", sodaß insgesamt 8 CAMPORTs an eine KDM Karte angeschlossen werden können. (3 Adressbits erlauben 8 verschiedene Basisadressen). Die CAMPORTs können dabei auf verschiedene CAMAC Überrahmen verteilt sein. Es können sich jedoch auch mehrere CAMPORTs in einem Überrahmen befinden. In unserem



1-16 Mikrocomputer Adressbits
BB Basisadresse von CAMAC
BC Basisadresse selektierbar in CAMPORT
N1-N16 CAMAC Stationsadresse
A1-A8 CAMAC Subadresse

Abbildung 10: CAMAC Adressierung

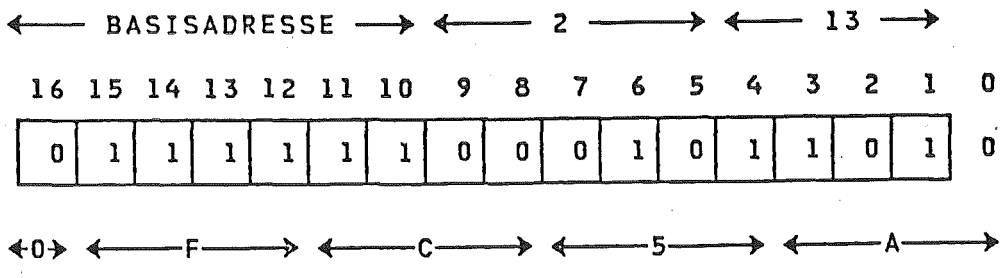


Abbildung 11: Adressierungsbeispiel

Experiment wurde die Basisadresse $E000 gewählt. Ein vorangestelltes "$" Zeichen gibt an, daß die angegebene Zahl hexadezimal[2] dargestellt ist. Es wird nur ein CAMPORT eingesetzt, dessen Jumpers so gesetzt sind, daß CAMAC den Bereich von $FC00 - $FFFF belegt (CAMPORT 7). Die zu einer CAMAC Adresse korrespondierende Adresse im Adressraum des Mikrocomputers läßt sich somit durch Zusammensetzen der CAMPORT Basisadresse, der CAMPORT Nummer und der CAMAC N und A Bits berechnen. Soll beispielsweise ein CAMAC Register mit Adresse N=2, A=13 gelesen werden, so muß der Funktionscode F0 an die Adresse $FC5A abgesetzt werden (siehe Abb. 11).

CAMPORT kann sich an jeder beliebigen CAMAC Station befinden (außer Station 25) und hat deshalb keinen direkten Zugang zu den N Leitungen des CAMAC Datenweges. Aus diesem Grund ist CAMPORT auf seiner Rückseite mittels eines Flachbandkabels mit dem A2 Crate Controller verbunden, welcher CAMPORT den indirekten Zugang zu den N Leitungen erlaubt. Dasselbe gibt für die LAM Linien (siehe auch Kap. 2).

4.3 Datenzugriff

Die CAMAC Daten werden über den Datenbus des Mikrocomputers und die Sende- bzw. Empfangspuffer transferiert und im CAMPORT Datenregister zwischengespeichert. Da sowohl das Datenregister als auch die Puffer nur 16 Bits breit sind, können nur 16 Schreib- und Leseleitungen von CAMPORT angesprochen werden. Das bedeutet, daß nur CAMAC Einheiten gelesen und beschrieben werden können, welche die Leitungen R17-R24 und W17-W24 nicht benutzen.

2

Hexadezimale Darstellung:

0000 = 0	0100 = 4	1000 = 8	1100 = C
0001 = 1	0101 = 5	1001 = 9	1101 = D
0010 = 2	0110 = 6	1010 = A	1110 = E
0011 = 3	0111 = 7	1011 = B	1111 = F

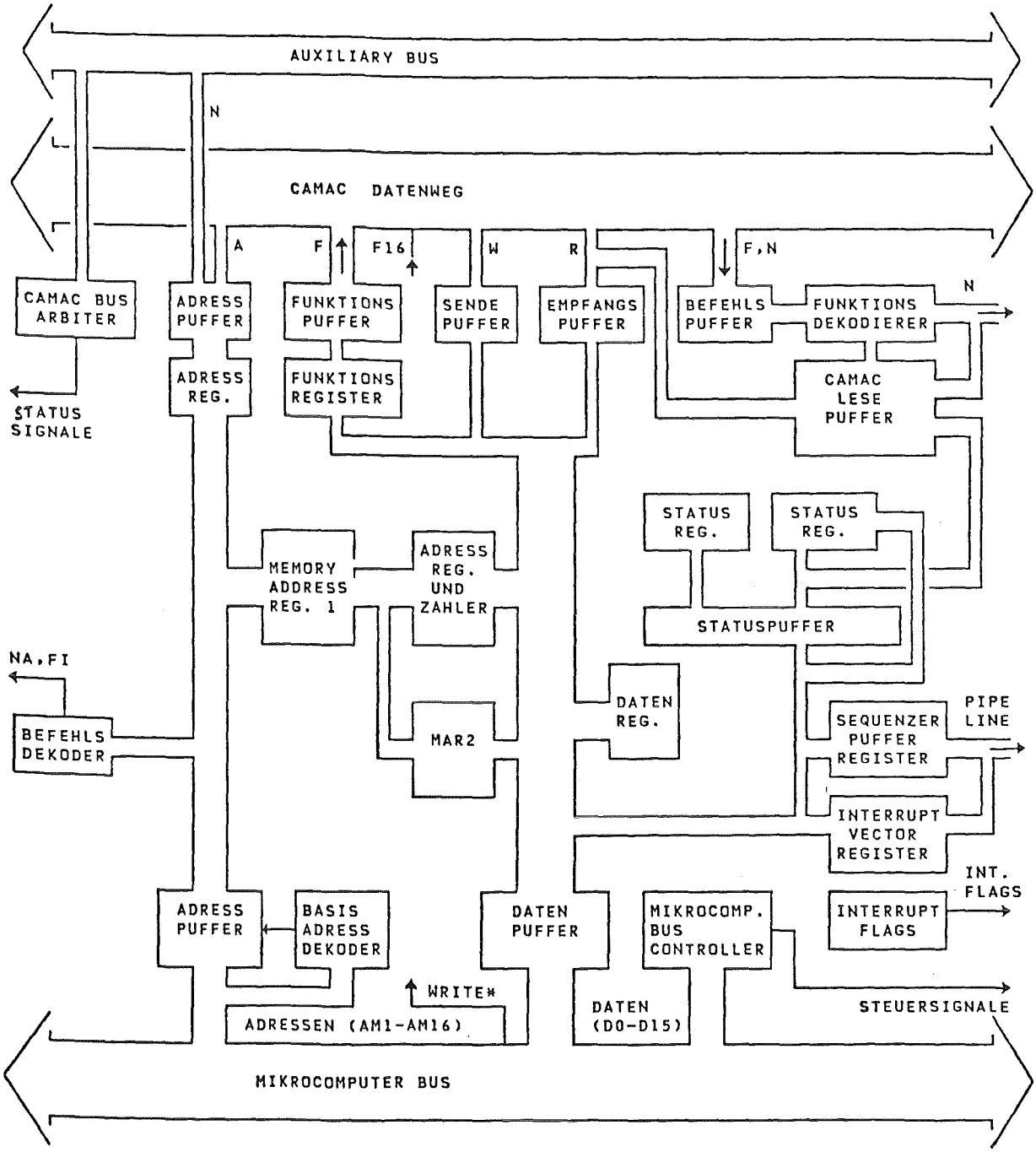


Abbildung 12: Blockdiagramm des CAMPORT Operationswerkes

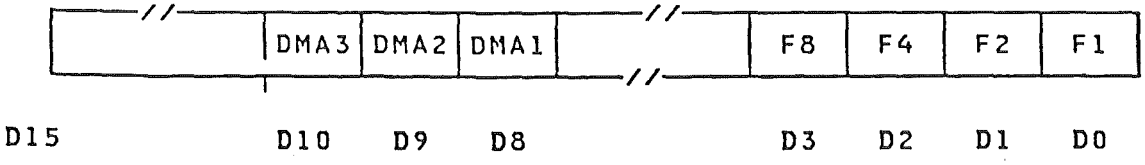
4.4 Funktionsregister

Wie bereits erläutert werden die CAMAC Stationsadressen und Subadressen von den Mikrocomputeradresslinien generiert, während die CAMAC Daten über den Datenbus transferiert werden. Die Funktionscodes jedoch können nicht gleichzeitig mit den Daten über die Datenleitungen übertragen werden, da nur 16 Leitungen zur Verfügung stehen, welche vollständig durch Datenbits ausgelastet sind. Es ist auch nicht möglich die Adresslinien zur Generierung der Funktionscodes heranzuziehen, da auch hier nicht genügend Linien zur Verfügung stehen. Die F-Codes werden daher vor der Ausführung des CAMAC Befehles über die Datenlinien des Mikrocomputer Busses und den CAMPORT Datenpuffer in das Funktionsregister in CAMPORT geladen. Erst danach findet der eigentliche CAMAC Zyklus statt, wobei der im Funktionsregister gespeicherte F-Code auf die CAMAC Funktionsleitungen gegeben wird. Selbstverständlich muß das Funktionregister nur dann geladen werden, wenn nicht schon der gewünschte F-Code in diesem Register steht. Es ist daher erstrebenswert, immer dieselben CAMAC Funktionen (z.B. nur F2 zur Auslese) zu benutzen, um ein Umladen des Funktionsregisters zu vermeiden, und somit wertvolle Auslesezeit zu sparen.

Die Zuordnung der einzelnen Funktionsregisterbits ist in Abb. 13 gezeigt. Soll beispielsweise der CAMAC Befehl F9 ausgeführt werden, so muß zunächst das Datenwort $0009 ins Funktionsregister geladen werden.

Das Funktionsbit F16, welches festlegt ob es sich um einen CAMAC Schreib- oder Lesebefehl handelt, wird direkt aus dem WRITE* Signal des KDM Busses generiert. Ein Schreibbefehl des Mikroprozessors setzt das F16 Bit und bewirkt dadurch einen CAMAC Schreibbefehl.

Mit den DMA Bits ist es möglich, CAMAC Daten ohne Einwirkung des Prozessors zwischen CAMAC Registern und dem Mikrocomputerspeicher zu transferieren. Zwei verschiedene DMA Typen sind in CAMPORT implementiert:



DMA1-3 : Bits zur Wahl des DMA Modes für DMAs von CAMAC in den Mikrocomputerspeicher
F1-F8 : CAMAC Funktionscode
D0-D15 : Datenbits des Mikrocomputers

Abbildung 13: Funktionsregister

- DMA1 Liest von einer konstanten CAMAC Adresse und schreibt das Ergebnis auf konsekutive Adressen im Mikrocomputerspeicher.

- DMA2 ist der "Q-Scan" Mode. Er erlaubt von aufsteigenden CAMAC Adressen zu lesen und das Ergebnis in konsekutive Adressen abzuspeichern. Q=0 signalisiert das Ende der Subadressen einer CAMAC Station. Daraufhin wird die Subadresse zurückgesetzt und die Stationsadresse inkrementiert. Ist Q=0 bei zwei aufeinanderfolgenden CAMAC Zyklen, so wird der DMA abgebrochen.

Weitere spezielle DMA Modes können nachträglich in CAMPORT eingebaut werden (siehe Kapitel 6).

Um das Funktionsregister laden zu können, muß CAMPORT eine "interne Funktion" besitzen, welche die Daten des MC 68000 als Funktionscode interpretiert und in das Funktionsregister abspeichert. Eine interne Funktion erkennt CAMPORT an Hand der N Signale. Ein CAMAC Überrahmen besitzt 24 Stationen. Mit den 5 N Bits können jedoch 32 Stationen angesprochen werden. CAMPORT benützt daher eine der "Pseudoadressen" (N=31) für seine internen Funktionen. Die Subadressbits wählen in diesem Fall die Art der Funktion an. Tabelle 4 zeigt ein Zusammenstellung aller vom Mikrocomputer ansprechbarer interner CAMPORT Funktionen.

N31A0(Write)	Lade CAMAC Funktionsregister
N31A1(Read)	Lese Statusregister
N31A1(Write)	Lade Statusregister
N31A2(Write)	Lade Memory Address Register
N31A2(Read)	Lese Memory Address Register
N31A3(Write)	Lade DMA Wordcount
N31A4(Write)	Setze Kommunikations LAM
N31A4(Read)	Setze Kommunikations LAM zurück
N31A5(Read)	Teste Q und X

Tabelle 4: Interne CAMPORT Funktionen ansprechbar vom MC 68000

Die CAMAC Strobesignale S1 und S2 werden von CAMPORT zur richtigen Zeit erzeugt, sodaß sich der Programmierer des Mikroprozessors nicht darum zu kümmern braucht. Das CAMPORT Steuerwerk (Kap. 6) übernimmt diese Aufgabe.

4.5 Beispiel eines CAMAC Befehles

Der Ablauf eines Camac Befehles soll an Hand eines Programmbeispieles noch einmal verdeutlicht werden. Die im Programm benutzten Assemblermnemonics sind im MC 68000 User's Manual [MOT79b] und im MC 68000 Cross Macro Assembler Manual [MOT79c] beschrieben.

In dem Beispiel soll ein Register eines Datenweg Displays, der sich in Station 1 befindet, mit dem Datenwort $AFFE beschrieben und die geschriebenen Daten zurückgelesen werden. Da der Datenweg Display den Zustand der Lese- und Schreiblinien des CAMAC Datenweges auf Leuchtdioden darstellt, können die Datentransfers visuell kontrolliert werden. Programmbeispiel 1 zeigt das zugehörige MC 68000 Assemblerprogramm.

4.6 Interrupts

Zur Interrupt getriebenen Auslese von CAMAC werden die LAMs der CAMAC Einheiten benutzt. 8 verschiedene LAM Leitungen können mit einem Prioritätsencoder verbunden werden, sodaß nur das LAM der höchsten Priorität einen Interrupt erzeugt. Außer den LAMs existieren jedoch noch weitere Interruptquellen in CAMPORT. Um die Herkunft der Interrupts unterscheiden zu können, generiert jede Quelle eine andere Interruptvektornummer. Die Interruptvektornummer wird über das Interruptvektorregister an den Mikroprozessor weitergegeben. Werden mehrere CAMPORTs eingesetzt, so muß auch unterschieden werden, welcher der CAMPORTs den Interrupt erzeugt hat. Tabelle 5 zeigt die Korrespondenz zwischen Interruptquelle und die Interruptvektornummer.

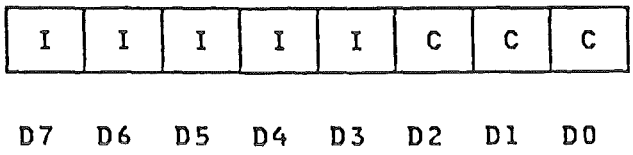
Programmbeispiel 1: Ablauf eines CAMAC Befehles

```
                                        M68mil cross macro assembler, CERN, DD-Division   20/09/83
                                               Version 3.5 from: 14th of September 1983   11:20:21

  ADDR    OPC    EXT    EXT    .    S O U R C E  -  S T A T E M E N T

000000                            IDENT       BEISPIEL
                                  ********************************************
                                  *       Beispiel eines CAMAC Befehles      *
                                  ********************************************
                                  *
          0000   FC00             CAMBAS      EQU     $FC00      Basisadresse und Camport Nummer
          0000   0001             NDIS        EQU         1      Station des Displays
          0000   0000             ADIS        EQU         0      Subadresse des Displays
          0000   001F             INTFUN      EQU        31      Interne Funktionen
          0000   0000             FUNCNR      EQU         0      Gewünschte Funktion
                                  *
                                  *           Berechne Adressen für Display
                                  *           und Funktionsregister
                                  *
          0000   FC20             DISADR      EQU         CAMBAS+32*NDIS+2*ADIS
          0000   FFE0             FUNREG      EQU         CAMBAS+32*INTFUN+2*FUNCNR
                                  *
                                  *           Lade Funktionsregister
                                  *
000000    33FC   0000             ENTRY            MOVE       #0,FUNREG
000004           0000   FFE0
                                  *
                                  *           Beschreibe und lese Display
                                  *
000008    33FC   AFFE                        MOVE       #$AFFE,DISADR
00000C           0000   FC20
000010    3039   0000   FC20               MOVE       DISADR,D0
                                  *
                                  *           Weiterverarbeitung der zurückgelesenen
                                  *           Daten
                                  *
000016                                        END        ENTRY

NO DIAGNOSTICS.
```



C	CAMPORT Identifikationsbits
I	Interruptvektornummer mit C=0

$40	Nicht implementierte Funktion
$48	Keine X-Response
$50	Keine Q-Response
$58	CAMAC Clear
$60	DMA End No Q
$68	DMA End Wordcount=0
$70	Ext. DMA (Lesen)
$78	Ext. DMA (Schreiben)
$80	LAM Priorität 1
$88	LAM Priorität 2
$90	LAM Priorität 3
$98	LAM Priorität 4
$A0	LAM Priorität 5
$A8	LAM Priorität 6
$B0	LAM Priorität 7
$B8	LAM Priorität 8

Tabelle 5: Interruptvektornummern

KAPITEL 5

CAMPORT, ZUGRIFF ZUM MIKROCOMPUTER SPEICHER

Im vorhergehenden Kapitel wurde das Mikrocomputer-CAMAC Interface erläutert. Es wurde gezeigt, wie der Mikroprozessor mit Hilfe von CAMPORT CAMAC Befehle absetzen kann, und wie er somit die experimentellen Daten auslesen und die Spektren aufbauen kann. Das zweite Problem des 2-Prozessorsystemes, die Kopplung der beiden Maschinen, wird im folgenden in Angriff genommen. Wiederum wird nur auf die Hardware eingegangen und für Softwareaspekte auf Kapitel 8 verwiesen.

5.1 CAMPORT Funktionen ansprechbar vom Minicomputer

CAMPORT kann nicht nur vom Mikroprozessor über die Flachbandkabel angesprochen werden, sondern auch von der PDP-11 über den CAMAC Datenweg. Die auf diesem Wege ausführbaren Funktionen sind in Tabelle 6 zusammengestellt. Diese CAMAC Funktionen lassen sich in 3 Gruppen aufteilen:

FxA0	Funktionen, bei denen CAMPORT nicht Busmaster des Mikroprozessorsystemes ist (Normalmode)
FxA1	Funktionen mit CAMPORT als Busmaster des Mikroprozessorsystemes (DMA Mode)

Diese beiden Gruppen von CAMAC Funktionen lassen sich an der rechten Station von CAMPORT ansprechen.

FxA0	Funktionen für das Statusregister

ansprechbar auf der linken CAMPORT Station.

Funktionen an der rechten Station

Normalmode

F0A0	Lese internen Datenpuffer
F2A0	Lese internen Datenpuffer und setze CLAM zurück
F10A0	Setze CLAM zurück
F17A0	Unterbreche MC 68000
F24A0	Sperre CLAM
F25A0	Busrequest
F26A0	Gib CLAM frei

DMA Mode

F0A1	Lese Mikrocomputerspeicher an der Adresse, die in MAR enthalten ist, mit autoinkrement (MAR+)
F1A1	Lese Mikrocomputerspeicher an MAR+, mit autodekrement des Wordcounters bis dieser 0 wird
F2A1	Lese Mikrocomputerspeicher an der Adresse, die im MAR enthalten ist (MAR)
F3A1	Lese Mikrocomputerspeicher an MAR mit autodecrement des Wordcounters bis dieser 0 wird
F9A1	Schließe Lese-DMA ab Gib Mikrocomputerbus frei und generiere DMA-End Interrupt
F11A1	Schließe Schreib-DMA ab Gib Mikrocomputerbus frei und generiere DMA-End Interrupt
F16A1	Beschreibe den Mikrocomputerspeicher an MAR+
F17A1	Beschreibe den Mikrocomputerspeicher an MAR
F18A1	Setze MAR für Lese-DMA
F19A1	Setze MAR für Schreib-DMA

Funktionen an der linken Station

F0A2	Lese Statusregister
F2A0	Lese Statusregister und setze Error LAM zurück

Tabelle 6: CAMPORT Funktionen ansprechbar über den CAMAC Datenweg

5.2 Der Normalmode

Als Normalmode wird bezeichnet, wenn CAMPORT nicht Busmaster des Mikroprozessorsystemes ist. In diesem Zustand kann CAMPORT nicht auf den Mikrocomputerspeicher zugreifen. Die einzige Möglichkeit einer Datenübertragung zwischen MC 68000 und PDP-11 ist über das Datenregister (siehe Blockschaltbild Abb. 12) gegeben, welches das letzte Datenwort enthält, das sich auf CAMPORT's internem Datenbus befand. Diese Möglichkeit wird von der Kommunikationssoftware (Kapitel 8) benutzt, um Statusinformation zu übertragen. Der MC 68000 schreibt die Statusinformation auf die Adresse $FFE8, was der Funktion N31A4, d.h. Setzen des CLAM, entspricht. Da sich zur Zeit der Ausführung dieser internen Funktion die Statusinformation auf den Datenleitungen befindet, wird sie automatisch ins Datenregister übertragen. Die PDP-11 kann dann die Information mit Hilfe der Funktion F2A0 lesen und setzt außerdem das CLAM zurück. Das CLAM wird somit zur Synchronisation der Datenübertragung benutzt.

Das CAMPORT Datenregister wird im weiteren auch mit FEDR (Fast Exchange Data Register, Register für schnellen Datenaustausch) bezeichnet, da über dieses Register ein Datenwort sehr schnell, d.h. ohne DMA Mechanismus (hierfür ist Busrequest, der DMA, Busfreigabe notwendig) übertragen werden kann.

Ein weiterer sehr wichtiger Befehl dieser Gruppe ist die Möglichkeit, den MC 68000 zu unterbrechen. Dies wird von der Kommunikationssoftware benutzt, um die Ausführung von Befehlen zu starten. Die PDP-11 schreibt mit Hilfe der Funktion F17A0 eine Interruptvektornummer ins Datenregister CAMPORT's, und setzt das Interruptbit. Abhängig von der Interruptvektornummer kann der MC 68000 verschiedene Interuptroutinen ausführen.

Um vom Normalmode in den DMA-Mode überzugehen, muß CAMPORT den Mikrocomputerbus anfordern und darauf warten, daß dieser vom Mikroprozessor freigegeben wird. Dies geschieht mit Hilfe von Funktion F25A0. Während alle CAMAC Befehle im Normalmode die Subadresse A0 belegen, ist für die Befehle des DMA Modes Subadresse A1 reserviert.

5.3 Der DMA Mode

Will man die großen Spektren im Mikrocomputerspeicher möglichst schnell zur PDP-11 transferieren so ist der 1 Wort Transfer über das FEDR sicherlich ungeeignet. Hierzu muß man mittels CAMAC direkt auf den Speicher zugreifen können. Dies ist mit Hilfe der Befehle der zweiten Gruppe möglich.

Zum Transfer der Spektren muß zunächst die Kontrolle über den Mikrocomputerbus erlangt werden (mit F25A0). Danach wird das MAR1 (Memory Adress Register 1) mit der Anfangsadresse des zu transferierenden Blockes beschrieben (F18A1). Mit Hilfe von Funktion F0A1 wird der eigentliche DMA Transfer durchgeführt, wobei nach jedem gelesenen Wort das MAR automatisch inkrementiert wird. In unserem Fall werden die Spektren in Blocks von 4095 Worten übertragen, da der Fisher AMC keine größere Blocklänge zuläßt. Nachdem alle Daten übertragen, sind wird der Mikrocomputerbus mit Hilfe von F9A1 wieder freigegeben, sodaß CAMPORT wieder in den Normalmode übergeht. Der Mikroprozessor kann dann mit der Datenaufnahme fortfahren. Das Ende des DMAs wird dem MC 68000 durch einen DMA-End Interrupt signalisiert.

5.4 Die Statusregister

Die dritte Gruppe von Funktionen bezieht sich auf die Statusregister. Diese Funktionen werden in unserem System nicht benutzt, sollen aber der Vollständigkeit wegen hier beschrieben werden.

Eines der beiden Statusregister kann vom MC 68000 gelesen werden und enthält Statusinformation für diesen Prozessor, während das andere der PDP-11 zugeordnet ist. Abb. 14 zeigt die Zuordnung der Statusbits in beiden Fällen. Diese Register sind besonders dann von Nutzen, wenn der Mikroprozessor nicht zum Aufbau von Spektren, sondern zur Vorauswertung experimenteller Daten benutzt wird. So kann der Mikroprozessor, nach der Auslese eines Ereignisses, CAMPORT vom CAMAC Datenweg isolieren, sodaß die linke Station CAMPORT's von der PDP-11 nicht mehr ansprechbar ist (keine X bzw. Q Response mehr),

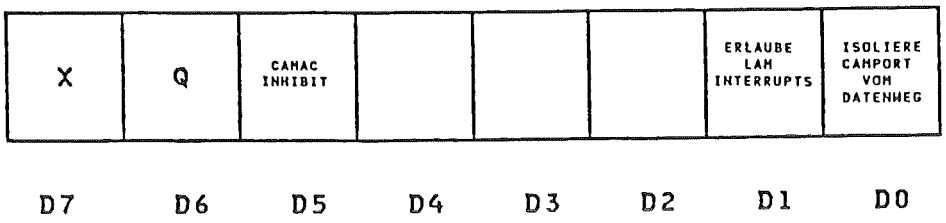


Abb. 14a: Statusregister ansprechbar vom Mikroprozessor

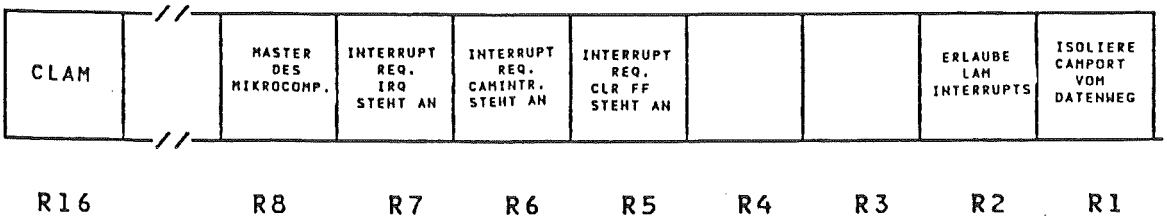


Abb. 14b: Statusregister ansprechbar vom CAMAC Datenweg

Abbildung 14: Die Statusregister

bis das Ereignis vollständig verarbeitet ist. CAMPORT ist jedoch auch vom CAMAC Datenweg isoliert, solange ein Interrupt an den MC 68000 ansteht. Das Statusregister wird an der linken Station CAMPORT's angesprochen und kann immer gelesen werden, sodaß der Grund einer Isolierung herausgefunden werden kann. Auch das "Error LAM", welches eine Fehlerbedingung in CAMPORT anzeigt (z.B. wenn eine nicht existente interne Funktion angesprochen wird) kommt von dieser Station.

5.5 CAMAC Initialisierung

Die CAMAC Befehle Z und C werden auch von CAMPORT gesondert behandelt. Das CAMAC C erzeugt einen Interrupt im Mikrocomputer und kann dazu benutzt werden, Initialisierungsroutinen, wie z.B. Löschen des Systemspeichers, zu starten. Das CAMAC Z generiert einen Reset an den Mikroprozessor, so daß dieselbe Initialisierung gestartet wird, wie sie beim Einschalten der Stromversorgung durchgeführt wird (Powerup Restart Sequence).

KAPITEL 6

DAS CAMPORT STEUERWERK

Will der Mikroprozessor einen CAMAC Befehl absetzen, so muß in CAMPORT eine ganze Reihe von Statussignalen in fester zeitlicher Abfolge erzeugt werden. Bei einem CAMAC Schreibbefehl (z.B. F16) muß bzw. müssen

- die CAMPORT Basisadresse dekodiert und die Adresspuffer geöffnet werden
- der Befehlsdekoder feststellen, daß keine interne Funktion angesprochen wurde
- die Adressen in das Adressregister übernommen werden
- die Datenpuffer geöffnet werden und die Daten im Datenregister zwischengespeichert werden
- die Handshakesignale generiert werden, um den Datentransfer zwischen MC 68000 und CAMPORT zu synchronisieren
- sichergestellt werden, daß der CAMAC Datenweg frei ist
- die CAMAC Adresse, der Funktionscode und die CAMAC Daten auf den Datenweg (bzw. an den A2 Controller) gegeben werden
- die CAMAC Strobe Signale S1 und S2 generiert werden.

Bei dieser Aufstellung von notwendigen Steueraktionen wurde vorausgesetzt, daß sich der richtige Funktionscode schon im Funktionsregister befindet.

Wollte man den Controller, der sämtliche notwendigen Steuersignale generieren muß, mit fest verdrahteter Logik aufbauen, so würde er extrem komplex und unüberschaulich. Der CAMPORT Controller generiert daher seine Steuersignale mit Hilfe eines Mikroprogrammes.

Mikroprogrammierte Controller sind noch immer ziemlich ungewohnt. In diesem Kapitel wird deshalb zunächst das Prinzip der Mikroprogrammierung an Hand eines einfachen Beispieles erklärt, bevor auf die Details des CAMPORT Controllers eingegangen wird. Am Ende des Kapitels wird gezeigt, wie ein Mikroprogramm mit Hilfe eines Mikroassemblers erzeugt wird.

6.1 Prinzip der Mikroprogrammierung

Zum Einstieg in dieses Kapitel soll die Steuerung einer Ampelanlage an einer Straßenkreuzung untersucht werden. Es wird davon ausgegangen, daß die sich kreuzenden Straßen dieselbe Verkehrsdichte haben, sodaß die Rot- und Grünphasen der beiden Ampeln dieselbe Dauer haben. Tabelle 7a zeigt die Ampelphasen, wobei die Zeit in 16 gleiche Abschnitte von 15 sec unterteilt ist. Die Grünphase dauert daher 1 min 15 während die Rotphase aus Sicherheitsgründen mit der Rotphase der anderen Ampel überlappen muß und deshalb länger dauert (1 min 45).

Eine Abfolge von Phasen, wie sie in Tabelle 7a dargestellt ist kann mit Hilfe eines Taktgenerators, eines Zählers und eines Mikroprogrammes in PROM (Programmable Read Only Memory, programmierbarer Festwertspeicher) generiert werden (siehe Abb. 15). Die Frequenz des Taktgenerators beträgt 1/15 sec^{-1}. Die Ausgänge des Zählers liefern die Adresse für das Mikroprogramm PROM, welches wie in Programmbeispiel 2 programmiert ist. Zum Zeitpunkt 0 schaltet das Mikroprogramm die grüne Lampe von Ampel 1 und die rote Lampe von Ampel 2 ein (Bit 1,6 gesetzt). Dieser Zustand bleibt erhalten bis zum Zeitpunkt 4. Zum Zeitpunkt 5 schaltet die Ampel 1 auf gelb (Bit 1=0, Bit 2=1), während Ampel 2 noch immer rot bleibt (Bit 6=1), u.s.w. Da sich der Zähler nach Zeitpunkt 15 wieder zurücksetzt, wird das Mikroprogramm wieder von vorne durchlaufen, und die Ampelphasen wiederholen sich demzufolge.

Es ist leicht zu sehen, wie die Dauer der Phasen geändert werden kann, falls die Steuerung für Straßen unterschiedlicher Verkehrsdichte eingesetzt werden soll. Es genügt nämlich eine oder mehrere Phasen (z.B 11,12) aus dem Mikroprogramm herauszunehmen und dafür neue Phasen (z.B 3,4) einzufügen. Dies wurde in Tabelle 7b bzw Programmbeispiel 2b gemacht. Es muß darauf hingewiesen werden, daß die Hardware (Taktgenerator, Zähler) dabei nicht verändert werden muß.

Die bisher beschriebene Steuerung schaltet die Ampelphasen auch dann um, wenn dies aus verkehrstechnischen Gründen nicht notwendig ist, d.h. wenn

ZEIT

	AMPEL 1	AMPEL 2
0	GRÜN	ROT
1	GRÜN	ROT
2	GRÜN	ROT
3	GRÜN	ROT
4	GRÜN	ROT
5	GELB	ROT
6	ROT	ROT
7	ROT	ROT/GELB
8	ROT	GRÜN
9	ROT	GRÜN
10	ROT	GRÜN
11	ROT	GRÜN
12	ROT	GRÜN
13	ROT	GELB
14	ROT	ROT
15	ROT/GELB	ROT

Tabelle 7a

	AMPEL 1	AMPEL 2
0	GRÜN	ROT
1	GRÜN	ROT
2	GRÜN	ROT
3	GRÜN	ROT
4	GRÜN	ROT
5	GRÜN	ROT
6	GRÜN	ROT
7	GELB	ROT
8	ROT	ROT
9	ROT	ROT/GELB
10	ROT	GRÜN
11	ROT	GRÜN
12	ROT	GRÜN
13	ROT	GELB
14	ROT	ROT
15	ROT/GELB	ROT

Tabelle 7b

Tabelle 7: Ampelphasen

ADRESSE	AMPEL 1			AMPEL 2		
	GR	GE	RO	GR	GE	RO
0	1	0	0	0	0	1
1	1	0	0	0	0	1
2	1	0	0	0	0	1
3	1	0	0	0	0	1
4	1	0	0	0	0	1
5	0	1	0	0	0	1
6	0	0	1	0	0	1
7	0	0	1	0	1	1
8	0	0	1	1	0	0
9	0	0	1	1	0	0
10	0	0	1	1	0	0
11	0	0	1	1	0	0
13	0	0	1	1	0	0
14	0	0	1	0	0	1
15	0	1	1	0	0	1

Mikroprogramm a

AMPEL 1			AMPEL 2		
GR	GE	RO	GR	GE	RO
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
0	1	0	0	0	1
0	0	1	0	0	1
0	0	1	0	1	1
0	0	1	1	0	0
0	0	1	1	0	0
0	0	1	1	0	0
0	0	1	0	0	1
0	1	1	0	0	1

Mikroprogramm b

Programmbeispiel 2: Mikroprogramm für die Ampelsteuerung

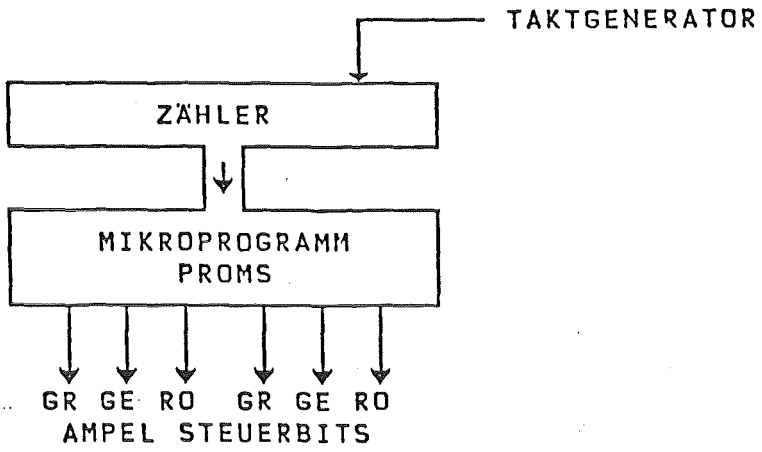


Abbildung 15: Einfache Ampelsteuerung

keine Autos an der grün werdenden Ampel warten. So kann es vorkommen, daß Autos zum Warten gezwungen werden, obwohl auf der kreuzenden Straße kein Verkehr herrscht. Um diesen Zustand zu vermeiden, werden Induktionsschleifen in den Boden verlegt, welche der Steuerung anzeigen, ob Autos warten oder nicht. Die Ampelsteuerung muß nun die Möglichkeit erhalten, abhängig von einer externen Bedingung, die Phasen umzuschalten. Die Steuerschaltung muß daher um mehrere Bauelemente erweitert werden, und die Wortbreite des Mikroprogrammes muß vergrößert werden. Zunächst wird der Zähler in Abb. 15 durch einen Adressequenz Controller (Sequenzer) ersetzt.

Der Sequenzer muß mindestens 2 Befehle verstehen:

CONT	Continue	Gehe zur nächsten Adresse über
JC	Jump Conditional	Gehe zur Adresse über, die vom Mikroprogramm angezeigt wird, falls die Bedingung wahr ist, sonst gib dieselbe Adresse noch einmal aus

Desweiteren ist ein "CC MUX" (Condition Code Multiplexer, Bedingungskodierer) notwendig, um die Signale von den Induktionsschleifen (die Condition Code Quellen) abfragen zu können. Abb. 16 zeigt die verbesserte Steuerung.

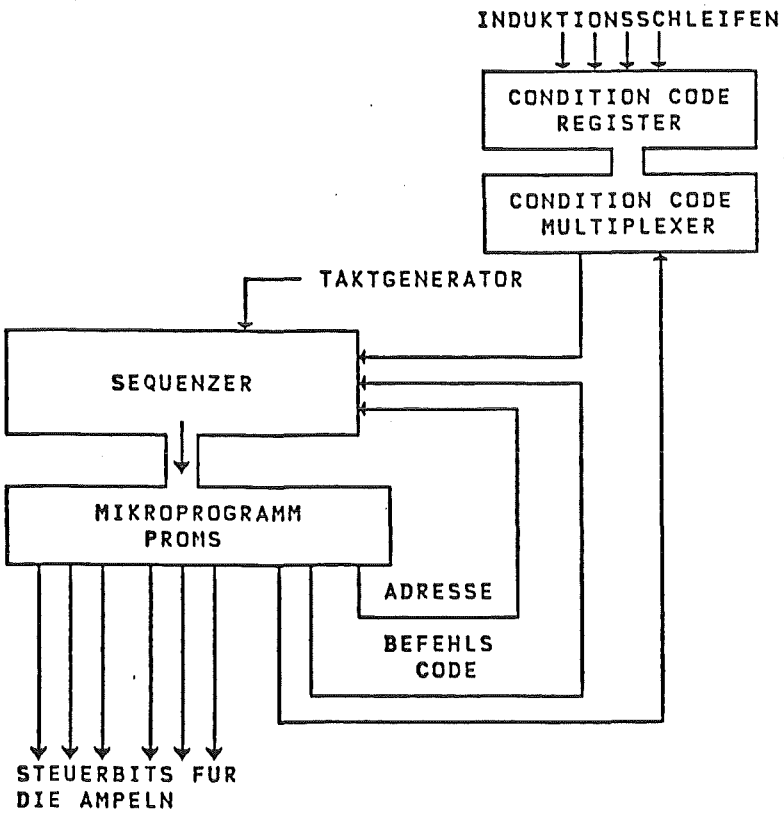


Abbildung 16: Verbesserte Ampelsteuerung

Das neue Mikroprogramm enthält nun außer den schon bekannten Bits für die Ampeln, weitere Bitfelder (Instruktionsfelder), welche der Steuerung des Sequenzers (Instruktionscode und Sprungadresse) und der Steuerung der CC MUXs dienen. Programmbeispiel 3 zeigt das neue Mikroprogramm.

	STEUERBITS FÜR SEQUENZER			AMPEL 1			AMPEL 2		
ZEIT	INSTR.	ADRESSE	CC	GR	GE	RO	GR	GE	RO
0	CONT			1	0	0	0	0	1
1	CONT			1	0	0	0	0	1
2	CONT			1	0	0	0	0	1
3	CONT			1	0	0	0	0	1
4	JC	5	1	1	0	0	0	0	1
5	CONT			0	1	0	0	0	1
6	CONT			0	0	1	0	0	1
7	CONT			0	0	1	0	1	1
8	CONT			0	0	1	1	0	0
9	CONT			0	0	1	1	0	0
10	CONT			0	0	1	1	0	0
11	CONT			0	0	1	1	0	0
12	JC	13		0	0	1	1	0	0
13	CONT			0	0	1	0	1	0
14	CONT			0	0	1	0	0	1
15	CONT			0	1	1	0	0	1

Programmbeispiel 3: Mikroprogramm der verbesserten Ampelsteuerung

Phase 0-3 läuft genauso ab wie in der alten Version. Allerdings muß dem Sequenzer ein Instruktionscode zur Verfügung gestellt werden, der ein Inkrementieren der Adresse bewirkt. Die Sprungadresse und das Bit für den CC MUX ist während dieser Phasen ohne Belang und kann einen beliebigen Zustand einnehmen. Bei Phase 4 wird zusätzlich zu den Funktionen des alten

Mikroprogrammes, nachgeschaut, ob die Induktionsschleife der kreuzenden Straße angesprochen hat. Ist dies der Fall so springt der Sequenzer auf die Adresse 5. Das bedeutet, daß das Mikroprogramm genauso fortgeführt wird wie das in der alten Version der Fall war. Hat die Induktionsschleife jedoch nicht angesprochen, so verharrt der Sequenzer auf Adresse 4 und schaltet die Ampeln nicht um.

6.2 Erzeugung der CAMPORT Steuersignale

Das Blockschaltbild des CAMPORT Controllers ist in Abb. 17 gezeigt. Man findet darin sämtliche Bauelemente der Ampelsteuerung wieder. Im Gegensatz zur Ampelsteuerung, wo nur eine Funktion durchgeführt werden mußte, die sich zyklisch wiederholte, müssen in CAMPORT mehrere unterschiedliche Steuerfunktionen erledigt werden. Das Mikroprogramm des CAMPORT Controllers ist deshalb unterteilt in mehrere Routinen, welche die verschiedenen Befehle abarbeiten. Eine zusätzliche Schaltung ist notwendig, um die Startadresse der entsprechenden Routine zu ermitteln und diese zu starten. An Hand des Blockschaltbildes wird nun im Detail erklärt, wie ein Befehl von CAMPORT abgearbeitet wird. Dabei kann der Befehl vom MC 68000, von CAMAC oder von anderen Quellen kommen.

Zunächst muß CAMPORT in den Grundzustand versetzt werden, in dem der Controller auf Befehle wartet. Dies geschieht entweder beim Einschalten des CAMAC Überrahmens oder durch ein CAMAC Z. Der Controller springt dabei auf die Adresse des Mikroprogrammes, welche die Wartefunktion durchführt. (Adresse 0 in Anhang C.2) Die Ausführung einer Mikroprogrammroutine wird getriggert, sobald ein Befehl ins Befehlsauswahlregister geschrieben wird. Dies kann der Controller mit Hilfe des CC MUX feststellen. Eine der Condition Code Quellen gibt nämlich an, ob ein Befehl ansteht oder nicht. Weitere Beispiele für Condition Code Quellen wären z.B. der CAMAC Strobe S1 und die CAMAC Q-Response, die über den CC MUX abgefragt werden können. Falls mehrere Befehle anstehen entscheidet ein Prioritätsencoder, welcher der Befehle zuerst bearbeitet wird. Es werden dabei 5 Befehlsquellen unterschieden, welche in Tabelle 8, nach Priorität geordnet, gelistet sind.

N	Funktionen ansprechbar von CAMAC Datenweg
FI	Interne Funktionen ansprechbar vom Mikroprozessor
NA	CAMAC Funktionen gesteuert von Mikroprozessor
LAM	LAMs
EXT	Flag Flip Flops

Tabelle 8: Befehlsquellen

Bei jeder dieser Befehlsquellen handelt es sich um eine ganze Gruppe von Befehlen. Die Gruppe N z.B. enthält alle CAMAC Befehle, die vom CAMAC Datenweg angesprochen werden können und damit alle Befehle von Tabelle 6. Zu jeder dieser Gruppen gehört ein Befehlsregister, in dem der gewünschte Befehl innerhalb der Gruppe spezifiziert ist. Nachdem der Controller erkannt hat, daß ein Befehl ansteht, wird das Register der Befehlsquelle mit der höchsten Priorität gelesen und mit Hilfe des Vektor Mapping PROMs die Startadresse der Mikroprogrammroutine bestimmt. Das Vektor Mapping PROM enthält die Startadressen sämtlicher zur Verfügung stehender Mikroprogrammroutinen und bildet den anstehenden Befehlscode auf die Startadresse der zugehörigen Routine ab (daher der Name "Vektor Mapping PROM"). Der Sequenzer springt daraufhin die zugeordnete Adresse an und führt die dort befindliche Routine aus. Nach Beendigung der Befehlsausführung springt der Sequenzer wieder zur Mikroprogrammroutine zurück, die auf neue Befehle wartet. Die Steuerfunktionen, die vom CAMPORT Controller durchgeführt werden müssen, sind wesentlich komplexer, als die beschriebene Steuerung der Ampeln. Trotzdem hat sich die Struktur des Steuerwerkes nur unwesentlich geändert. Nur die Breite des Mikroprogrammwortes muß erhöht werden, und ein leistungsfähigerer Sequenzer wird benötigt. Der CAMPORT Controller benutzt dehalb einen AM 2910 Mikrosequenzer, dessen Befehlssatz in Tabelle 9 zusammengestellt ist.
Mit diesem noch immer sehr kleinen Befehlssatz ist es möglich bedingte Sprungbefehle, Schleifen und sogar Unterprogramme auszuführen. Eine ausführliche Beschreibung der einzelnen Befehle findet sich im AM 2910 Datenblatt.

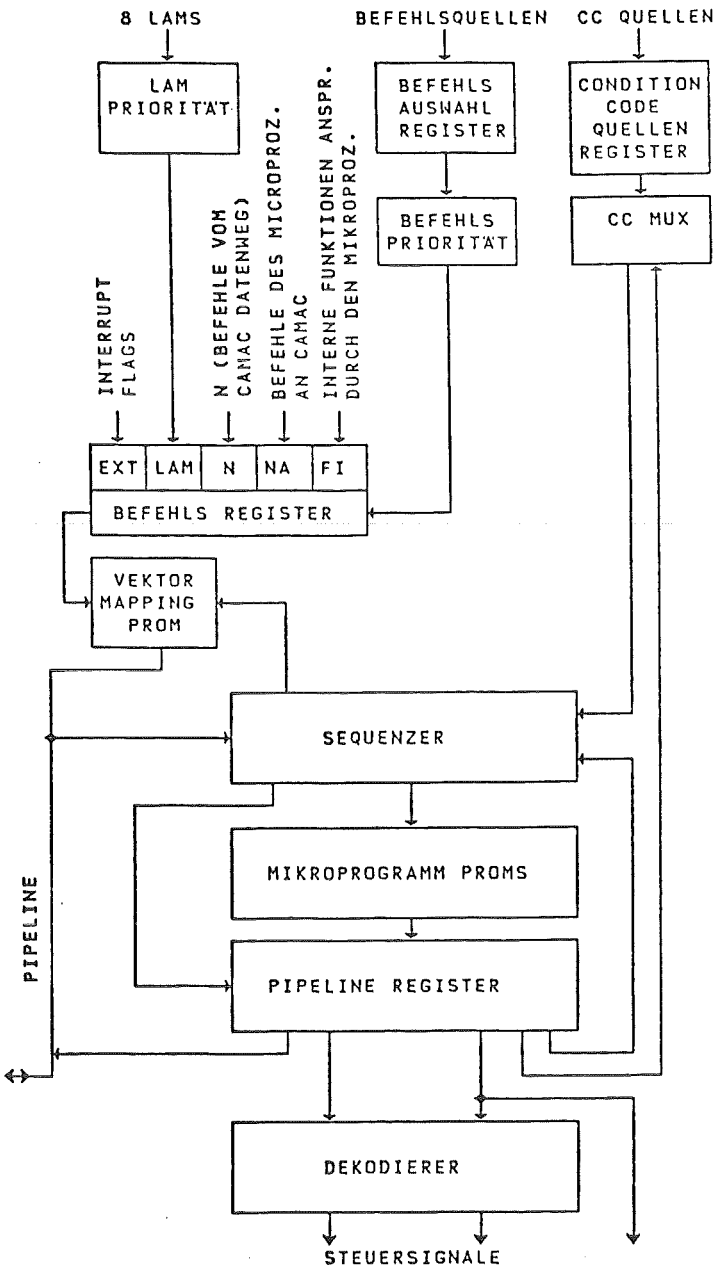


Abbildung 17: Blockschaltbild des CAMPORT Controllers

I_3-I_0	Bezeichnung	Bedeutung
0	JZ	JUMP ZERO
1	CJS	CONDITIONAL JUMP TO SUBROUTINE PIPELINE
2	JMAP	JUMP MAP
3	CJP	CONDITIONAL JUMP PIPELINE
4	PUSH	PUSH/CONDITIONAL LOAD COUNTER
5	JSRP	CONDITIONAL JUMP TO SUBROUTINE REGISTER/PIPELINE
6	CJV	CONDITIONAL JUMP VECTOR
7	JRP	CONDITIONAL JUMP REGISTER/PIPELINE
8	RFCT	REPEAT LOOP,COUNTER≠0
9	RPCT	REPEAT PL, COUNTER≠0
10	CRTN	CONDITIONAL RETURN
11	CJPP	CONDITIONAL JUMP PL + RETURN
12	LDCT	LOAD CONTER + CONTINUE
13	LOOP	TEST END LOOP
14	CONT	CONTINUE
15	TWB	THREE WAY BRANCH

Tabelle 9: Befehlssatz des AM 2910 Mikrosequenzers

Als weiterer Unterschied zur Ampelsteuerung fällt das "Pipelineregister" am Ausgang der PROMs auf. Im Gegensatz zur Ampelsteuerung ist in CAMPORT die Ausführungszeit der Befehle ein wichtiger Parameter. Das Pipelineregister erlaubt, die Ausführungsphase (Executionphase) und das Erkennen des nächsten Befehles (Fetchphase) überlappen zu lassen und dadurch die Ausführungsgeschwindigkeit annähernd zu verdoppeln. (Die Überlappung ist bei Sprungbefehlen nicht möglich. Eine Steigerung der Geschwindigkeit um einen Faktor 2 kann daher nicht erreicht werden). Die Dekodierer am Ausgang des Pipelineregisters helfen die Wortbreite des Mikroprogrammwortes einzuschränken. 8 Steuersignale können nämlich auf diese Weise mit nur drei Mikroprogrammbits erzeugt werden.

6.3 Entstehung eines Mikroprogrammes

Ein einfaches Mikroprogramm, wie das der Ampelsteuerung, kann noch von Hand geschrieben werden. Bei längeren Programmen jedoch schleichen sich leicht Fehler ein, wenn jedes Bit von Hand programmiert werden muß. Es ist daher vorteilhaft auf einen Mikroassembler zurückgreifen zu können. Das CAMPORT Kontrollprogramm wurde mit Hilfe des Signetics Mikroassemblers [SIG99] erstellt. Es handelt sich hierbei um einen Cross-Mikroassembler (Cross-Mikroassembler = Übersetzungsprogramm, das auf einem Großrechner abläuft und ein symbolisches Mikroprogramm in seine binäre Darstellung verschlüsselt. Das verschlüsselte Programm wird in PROMs geladen und im Zielsystem (CAMPORT) zum Einsatz gebracht). Als einfaches Beispiel für ein solches symbolisches Mikroprogramm soll das Programm für die verbesserte Ampelsteuerung (Abb. 16) erstellt werden.

Da jedes mikroprogrammierte Steuerwerk einen anderen Befehlssatz besitzt (andere Wortbreite, andere Bedeutung der Steuerbits) muß dem Mikroassembler zunächst die zu steuernde Maschine beschrieben werden. Dies geschieht mit Hilfe eines Definitionsfiles (Programmbeispiel 4). Darin wird die Wortbreite des Mikroprogrammwortes (mit INSTRUCTION WIDTH) und die Breite und Bedeutung des einzelnen Instruktionsfelder (mit FIELD) beschrieben. Jeder Wert, den ein Instruktionsfeld annehmen kann, bekommt einen symbolischen Namen (mit MICROP), welcher später bei der Programmierung benutzt wird. Außerdem ist es möglich, symbolischen Namen feste Werte zuzuordnen (mit EQU).

Nachdem die Bedeutung der benutzten Symbole definiert ist, kann mit der eigentlichen Programmierung begonnen werden. Eine Mikroinstruktion besteht aus einer Sprungmarke (Label), falls benötigt, und einer Folge von symbolischen Namen, die im Definitionsfile festgelegt wurden. Der Mikroassembler übersetzt die Namen in die entsprechende Bitfolge und errechnet eventuell benötigte Sprungadressen und trägt diese in die dafür vorgesehenen Felder des Mikroprogrammwortes ein. Programmbeispiel 5 zeigt das Mikroprogramm für die Ampelsteuerung. Jeder Eingabezeile folgt darin die Übersetzung des Mikroassemblers.

```
SIGNETICS BIPOLAR MICRO ASSEMBLER(A)                                PAGE 00001
  ADDR: AM2910INSTR ADRESSE CCMUX AMPEL1 AMPEL2 SPARE

00001  "
00002  ********************************************************************
00003  *                                                                  *
00004  *          Definitionsfile für das Microprogramm der               *
00005  *          verbesserten Ampelsteuerung (Abb. 15)                   *
00006  *                                                                  *
00007  *          Ulrich Raich            21. November 83                 *
00008  *                                                                  *
00009  ********************************************************************
00010
00011             Microassembler Definitions File
00012             ===============================
00013  "
00014
00015  "Festlegung der Mikroprogramm Wortbreite"
00016  INSTRUCTION WIDTH 16;
00017
00018  "Definition der Instruktionsfelder"
00019  FIELD AM2910_INSTR WIDTH 4 DEFAULT 0;"Instruktionen des Sequenzers"
00020  FIELD ADRESSE      WIDTH 4 DEFAULT 0;"Sprung Adressen"
00021  FIELD CCMUX        WIDTH 1 DEFAULT 0;"Condition Codes"
00022  FIELD AMPEL1       WIDTH 3 DEFAULT 0;"Phasen der Ampel 1"
00023  FIELD AMPEL2       WIDTH 3 DEFAULT 0;"Phasen der Ampel 2"
00024  FIELD SPARE        WIDTH 1 DEFAULT 0;"Nicht benutztes Bit"
00025  END INSTRUCTION;
00026
00027  "Definition der Sequenzer Instruktionen"
00028  MICROP   CJP  ASSIGN  AM2910_INSTR = 0011B;"Conditional Jump"
00029  MICROP   CONT ASSIGN  AM2910_INSTR = 1110B;"Continue"
00030
00031  "Definition der Ampelphasen für beide Ampeln"
00032  MICROP   ROT1      ASSIGN AMPEL1 = 001B;"Rot Phase"
00033  MICROP   GELB1     ASSIGN AMPEL1 = 010B;"Gelb Phase"
00034  MICROP   GRUN1     ASSIGN AMPEL1 = 100B;"Grün Phase"
00035  MICROP   ROT_GELB1 ASSIGN AMPEL1 = 011B;"Phase rot/gelb"
00036  MICROP   ROT2      ASSIGN AMPEL2 = 001B;"Rot Phase"
00037  MICROP   GELB2     ASSIGN AMPEL2 = 010B;"Gelb Phase"
00038  MICROP   GRUN2     ASSIGN AMPEL2 = 100B;"Grün Phase"
00039  MICROP   ROT_GELB2 ASSIGN AMPEL2 = 011B;"Phase rot/gelb"
00040
00041  "Defintion der Sprungadressen"
00042  MICROP   JUMP(LOC) ASSIGN ADRESSE = LOC;"Springe nach Loc"
00043
00044  "Test ob Umschalten der Ampel notwendig ist"
00045  MICROP   TEST(CC)  ASSIGN CCMUX = CC;"Teste Ampel1 oder Ampel2"
00046
00047  AUTO1:   EQU    0B;"Test ob Ampel 1 umgeschaltet werden muß"
00048  AUTO2:   EQU    1B;"Test ob Ampel 2 umgeschaltet werden muß"
```

Programmbeispiel 4: Definitionsfile für das Mikroprogramm der Ampelsteuerung

```
SIGNETICS BIPOLAR MICRO ASSEMBLER(A)                                PAGE 00002
  ADDR: AM2910INSTR ADRESSE CCMUX AMPEL1 AMPEL2 SPARE

00049  PROGRAM AMPEL WIDTH 16 LENGTH 16;
00050  "
00051  ************************************************************
00052  *                                                          *
00053  *          Ampel Microprogramm                             *
00054  *          Produziert das Microprogramm fUr die            *
00055  *          verbesserte Ampelsteuerung (Abb. 15)            *
00056  *                                                          *
00057  *          Ulrich Raich            21. November 83         *
00058  *                                                          *
00059  ************************************************************
00060
00061            Mikroprogramm der verbesserten Ampelsteuerung
00062            ============================================
00063  "
00064  LIST SOURCE OBJECT;"Steuert das Listing"
00065
00066  ORG 0H;"Das Programm fängt an Adresse 0 der Proms an"
00067
00068  "-----------------------------------------------------------"
00069  "Erste Ampelphase"
00070  START:  CONT  GRUN1 ROT2;"Ampel 1 ist rot, Ampel 2 ist grun"
  0000: 1110        0000    0     100    001    0
00071          CONT  GRUN1 ROT2;
  0001: 1110        0000    0     100    001    0
00072          CONT  GRUN1 ROT2;
  0002: 1110        0000    0     100    001    0
00073          CONT  GRUN1 ROT2;
  0003: 1110        0000    0     100    001    0
00074  "-----------------------------------------------------------"
00075  "Übergang zur zweiten Phase falls Autos an Ampel 2 warten"
00076          CJP TEST(AUTO2) JUMP(PHASE2) GRUN1 ROT2;
  0004: 0011        0101    1     100    001    0
00077  PHASE2: CONT  GELB1 ROT2;"Übergang zur nachsten Phase"
  0005: 1110        0000    0     010    001    0
00078          CONT  ROT1 ROT2;
  0006: 1110        0000    0     001    001    0
00079          CONT  ROT1 ROT_GELB2;
  0007: 1110        0000    0     001    011    0
00080          CONT  ROT1 GRUN2;
  0008: 1110        0000    0     001    100    0
00081          CONT  ROT1 GRUN2;
  0009: 1110        0000    0     001    100    0
00082          CONT  ROT1 GRUN2;
  000A: 1110        0000    0     001    100    0
00083          CONT  ROT1 GRUN2;
  000B: 1110        0000    0     001    100    0
00084  "-----------------------------------------------------------"
00085  "Übergang zur ersten Phase falls Autos an Ampel 1 warten"
00086          CJP TEST(AUTO1) JUMP(PHASE1) ROT1 GRUN2;
  000C: 0011        1101    0     001    100    0
00087  PHASE1: CONT  ROT1 GELB2;
  000D: 1110        0000    0     001    010    0

00088          CONT  ROT1 ROT2;
  000E: 1110        0000    0     001    001    0
00089          CONT  ROT_GELB1 ROT2;
  000F: 1110        0000    0     011    001    0
00090  "-----------------------------------------------------------"
00091  "Ab hier wiederholt sich das Programm zyklisch"
00092          END;
```

Programmbeispiel 5: Auszug aus dem Ampel Mikroprogramm

Zuletzt muß festgelegt werden, wie die Bits des übersetzten Mikroprogrammes in den PROMs angeordnet werden sollen. Dies geschieht mit Hilfe eines Formatfiles. Ein exemplarischer Formatfile für die Ampelsteuerung ist in Programmbeispiel 6 gegeben. Teil a stellt die vom Programmierer gegebene Information dar, während Teil b das vom Mikroassembler ausgegebene Listing zeigt.

Das Mikroprogramm CAMPORT's unterscheidet sich von dem Ampelbeispiel nur durch sein breiteres Mikrowort (40 Bits) und durch seine größere Länge (256 Mikroworte). Auch werden kompliziertere Adressfolgen (d.h komplexere Instruktionen des AM2910) benötigt. Der prinzipielle Aufbau des Programmes ist jedoch derselbe wie bei dem Ampelprogramm. Auschnitte aus dem Definitionsfile und dem Mikroprogramm CAMPORT's sind in Anhang C gegeben.

Ähnlich wie bei dem Programm der einfachen Ampelsteuerung (Programmbeispiel 2), wo eine Änderung der Ampelphasen nur eine Änderung des Mikroprogrammes, nicht aber der Hardware zur Folge hatte, kann man, durch Hinzufügen neuer Mikroprogrammroutinen neue Funktionen (z.B. spezielle DMA Modes) in CAMPORT implementieren.

```
SIGNETICS BIPOLAR MICRO FORMATTER(A)   FORMAT CONTROL INPUT      PAGE 00001
 LINE                          IMAGE

00001  " Ampel Format command file"
00002
00003  MEMORY WIDTH 8 LENGTH 16; "PROM 2"
00004  FORMAT WORD (HEADER=' ' DATA=HEX);
00005  OUTPUT FROM 0 TO 15 OF AMPEL;
00006      SELECT SPARE(0)
00007             AMPEL2(2)
00008             AMPEL2(1)
00009             AMPEL2(0)
00010             AMPEL1(2)
00011             AMPEL1(1)
00012             AMPEL1(0)
00013             CCMUX(0);
00014
00015  MEMORY WIDTH 8 LENGTH 16; "PROM 1"
00016  FORMAT WORD (HEADER=' ' DATA=HEX);
00017  OUTPUT FROM 0 TO 15 OF AMPEL;
00018      SELECT ADRESSE(3)
00019             ADRESSE(2)
00020             ADRESSE(1)
00021             ADRESSE(0)
00022             AM2910_INSTR(3)
00023             AM2910_INSTR(2)
00024             AM2910_INSTR(1)
00025             AM2910_INSTR(0);
00026
00027  END;
```

Programmbeispiel 6a: Eingabe

```
SIGNETICS BIPOLAR MICRO FORMATTER(A) INSTRUCTION FIELD SUMMARY  PAGE 00003
  ID  FIELD                          OBJECT MODULE

 001  AM2910INSTR(3)                 2(3)
 001  AM2910INSTR(2)                 2(2)
 001  AM2910INSTR(1)                 2(1)
 001  AM2910INSTR(0)                 2(0)

 001  ADRESSE(3)                     2(7)
 001  ADRESSE(2)                     2(6)
 001  ADRESSE(1)                     2(5)
 001  ADRESSE(0)                     2(4)

 001  CCMUX(0)                       1(0)

 001  AMPEL1(2)                      1(3)
 001  AMPEL1(1)                      1(2)
 001  AMPEL1(0)                      1(1)

 001  AMPEL2(2)                      1(6)
 001  AMPEL2(1)                      1(5)
 001  AMPEL2(0)                      1(4)

 001  SPARE(0)                       1(7)
```

Programmbeispiel 6b: Ausgabe

Programmbeispiel 6: Formatfile für die Ampelsteuerung

KAPITEL 7

DER VME-BUSADAPTER

Um ein zweidimensionale Spektrum mit 2048 Energie- und 32 Zeitkanälen abspeichern zu können, werden

2048 Energiekanäle × 32 Zeitkanäle × 4Bytes/Kanal ⇒ 256 KBytes

benötigt. Bei geringerer Auflösung geht der Speicherplatzbedarf entsprechend zurück.

In dem System, wie es im Moment benutzt wird, sind 2 MBytes für die Speicherung der Spektren von 6 Detektoren vorgesehen. Wie man aus Tabelle 3 ersieht, kann man mit dem KDM Bus jedoch nur 128 KBytes adressieren, von denen noch ein Teil für die Kommunikationssoftware (siehe Kap. 8) und für den Adressbereich CAMPORT's abgezogen werden muß. Es muß deshalb auf einen leistungsfähigeren Bus ausgewichen werden. Die Wahl fiel dabei auf den VME Bus, dessen Eigenschaften im ersten und zweiten Kapitels zusammengestellt sind.

Mit der Benutzung eines anderen Busses wird außerdem eine Anpassungsschaltung zwischen CAMPORT und dem VME Bus, der VME Busadapter, notwendig, dessen Beschreibung der dritte Teil diese Kapitels gewidmet ist.

7.1 Allgemeines über den VME Bus

Mit der Einführung des MC 68000 Mikroprozessors wurde ein neuer 16 Bit Mikrocomputerbus benötigt, der alle Möglichkeiten dieses Prozessors ausschöpft. Da sowohl die Entwicklung eines solchen Busses (auf der Seite des Herstellers) als auch das Erlernen des Busses (auf der Seite der Benutzer) einen hohen Aufwand an Zeit und Geld erfordert, sollte ein Bus entwickelt werden, der auch

mit den zukünftigen 32 Bit Prozessoren noch einsetzbar ist und somit auch in der Zukunft benutzt werden kann. Gemeinsam mit den Firmen Mostek, Signetics, Philips etc. entwickelte die Firma Motorola den VME Bus. Wegen seiner hohen Leitungsfähigkeit und wegen der großen Unterstützung von Seiten der Industrie hat sich dieser Bus auch bei den Benutzern sehr schnell durchgesetzt, sodaß heute, zwei Jahre nach der Markteinführung des Busses schon mehrere hundert verschiedene VME Platinen erhältlich sind.

Ein VME System besteht aus einem Crate im Doppeleuropa Format mit bis zu 20 Einschüben. Alle Einheiten sind untereinander über den VME Bus gekoppelt. 96 polige Stecker verbinden alle VME Karten mit einer Mutterplatine. Die Pinbelegung der Stecker findet sich im Anhang E. Der VME Bus ist an jedem Ende der Mutterplatine mit Widerständen abgeschlossen, um Reflexionen der Signale auf den Buslinien zu vermeiden und dadurch eine hohe Datenübertragungsrate zu erlauben. Eine zweite Mutterplatine kann eingesetzt werden, wenn das System von einem 32 Bit Prozessor gesteuert wird. Diese Platine enthält dann alle zusätzlich notwendigen Signalleitungen.

7.2 Eigenschaften des VME Busses

In Tabelle 10 sind die Eigenschaften des VME Busses in derselben Weise zusammengefaßt, wie dies in Tabelle 3 für den KDM Bus der Fall war. Dadurch lassen sich die beiden Busse direkt miteinander vergleichen und man sieht, wo eine Anpassungsschaltung notwendig ist.
Wie man schon aus Tabelle 10 ersieht, können sehr komplexe Systeme mit Hilfe des VME Busses aufgebaut werden. Vielprozessorsysteme können erstellt werden, leistungsfähige Mehrbenutzersysteme für Offline Betrieb können ebenso implemetiert werden wie sehr schnelle Echtzeitsysteme. Da für unsere Anwendung jedoch nur der große Adressraum und die Möglichkeit von DMA Transfers wichtig sind, wird für eine ausführliche Beschreibung des VME Busses und dessen Möglichkeiten auf das VME Specification Manual [VME82] verwiesen. Die hohe Leistungsfähigkeit des Busses erlaubt allerdings, das VME System, nach seiner Benutzung als Vielkanalanalysator, auch für viele andere Anwendungen einzusetzen.

Tabelle 10: Eigenschaften des VME Busses

- Asynchroner Bus
- 23 Adressleitungen (A1-A23) ermöglichen den Zugriff zu 8M Worten. Erweiterbar auf 31 Adressleitungen.
- Byte Adressierung möglich über Datenstrobes DS0*,DS1*
- 5 Adressmodifiers erlauben Systemspeicherbereiche und Datenspeicherbereiche vor unbefugtem Zugriff zu schützen
- 16 Datenlinien (D0-D15), erweiterbar auf 32
- 8 Mhz System Taktfrequenz
- 7 Interruptebenen
- 4 Busmastership Niveaus durch

  BR0*-BR4* Busrequest Leitungen
  BG0*-BG3* Busgrant Leitungen
  BGACK* Busgrant Acknowledge Leitung

7.3 Der VME Busadapter

Die CAMPORT Kabel, welche mit den Linien des KDM Busses 1:1 verbunden werden konnten, müssen mit Hilfe des VME Busadapters (siehe Blockschaltbild in Abb. 18) auf den VME Bus angepaßt werden.

Wie ein Vergleich der Tabellen 3 und 10 zeigt, müssen sämtliche Datensignale invertiert werden, was einfach durch invertierende Bustransceiver (bidirektionale Bustreiber) geschieht. Desweiteren müssen die Adresslinien A17-A23 dekodiert und damit die Basisadresse von CAMPORT im VME Adressraum festgelegt werden. Hierzu werden die vom VME Bus kommenden Adressignale mit dem Zustand von DIL Schaltern (Dual In Line, Schalter, deren Pinabstand mit dem integrierter Schaltkreise übereinstimmt) verglichen. Die niederwertigen Adressbits (A1-A16) und die Daten werden nur dann an CAMPORT weitergegeben, wenn die Adressbits A17-A23 mit der Stellung der DIL Schalter übereinstimmen. Nur dann generiert nämlich die Steuerschaltung die

zum Öffnen der Adresspuffer (1) und der Bustransceiver nötigen Steuersignale. Während des Transfers der Spektren erzeugt CAMPORT nur 16 Adressbits. Die restlichen Bits A17-A23 müssen ebenfalls vom VME Adapter zur Verfügung gestellt werden. Auch diese Signale werden mit Hilfe von DIL Schaltern (2) und eines Adresspuffers (2) erzeugt. Die Steuerschaltung öffnet diesen Adresspuffer nur während des DMA. Der VME Busadapter bestimmt daher auch die Lage des Speicherbereiches, zu dem CAMPORT Zugriff hat (in Kap. 8 mit Kommunikationsbereich bezeichnet). Abb. 19 zeigt die Aufteilung des VME Adressraumes. Da die Busvorrangsteuerung bei den verschiedenen VME Prozessorkarten unterschiedlich gehandhabt wird, kann für den Busrequest CAMPORT's jede der 4 BR* und BG* Leitungen durch Jumpers selektiert werden.

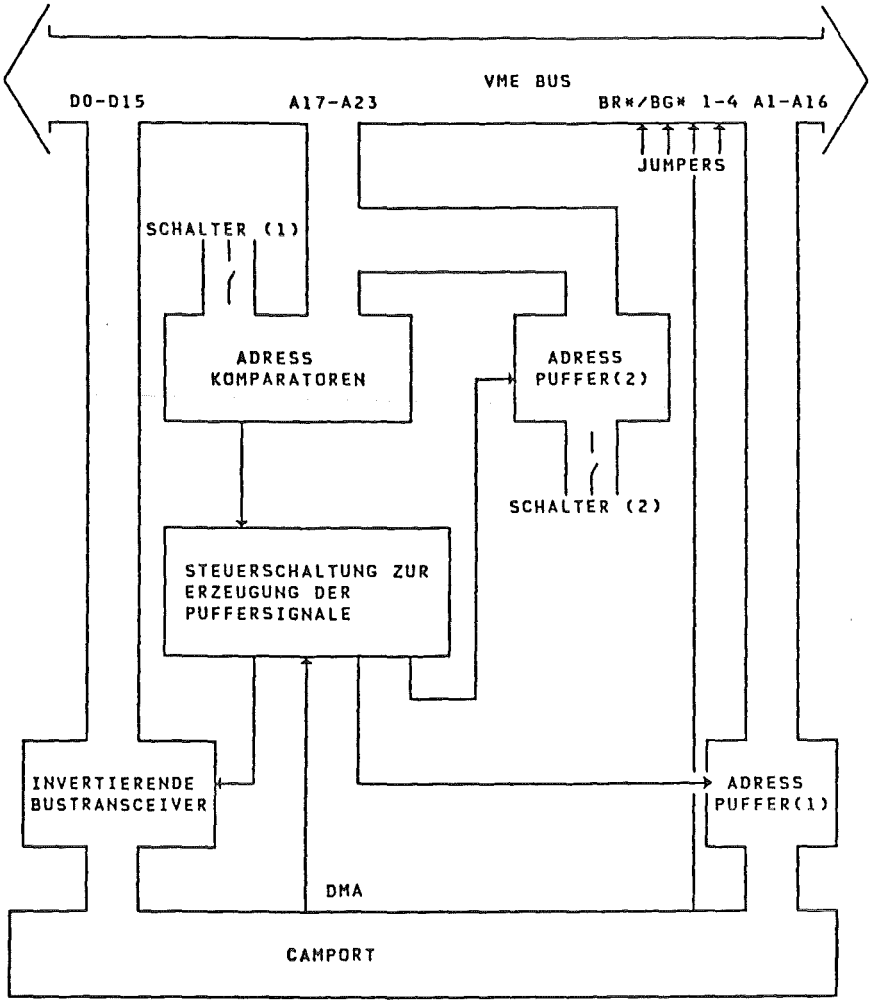


Abbildung 18: Blockschaltbild des VME Adapters

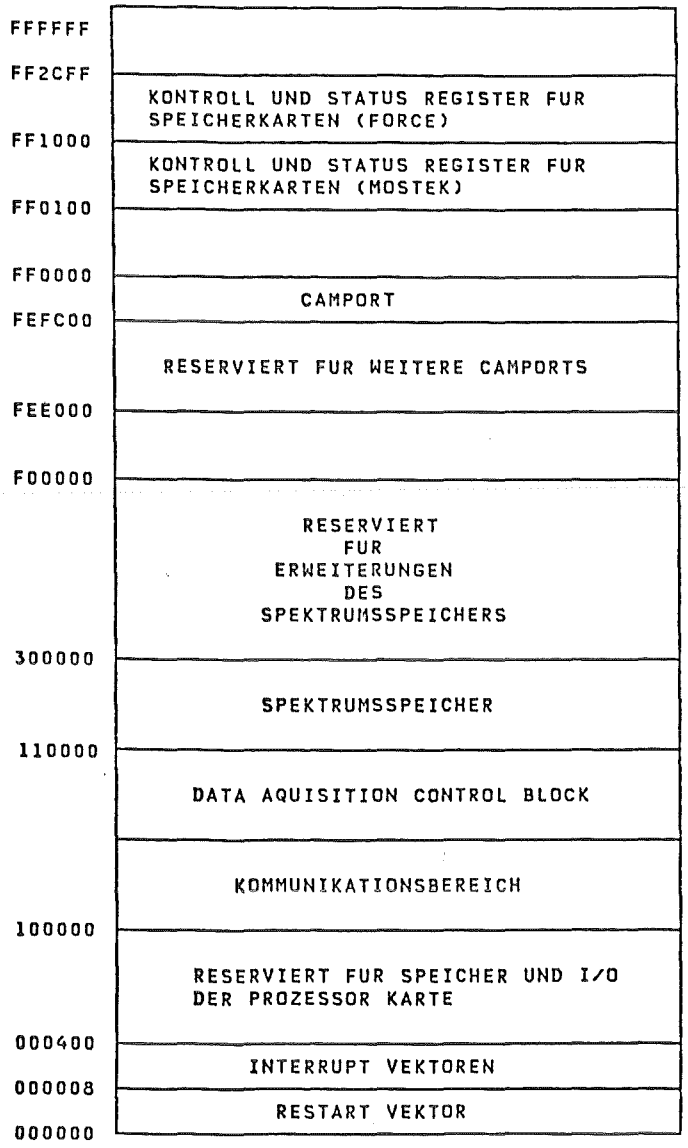


Abbildung 19: Aufteilung des Speicherbereiches im VME System

KAPITEL 8

SOFTWARE

Die Software läßt sich in drei getrennte Einheiten unterteilen.

- das Ausleseprogramm im Mikrocomputer
- die Kommunikationssoftware im Mikrocomputer und auf der PDP-11
- PDP-11 Verwaltungsprogramme zum Bandschreiben und zur Online Darstellung der Spektren.

Zu Beginn dieses Kapitels wird beschrieben, welche CAMAC Einheiten in unserem Experiment eingesetzt werden und daher ausgelesen werden müssen, und wie die Informationen im Mikrocomputer verarbeitet werden. Danach folgt eine Beschreibung der Kommunikationssoftware, welche für die Transfers der aufgesammelten Spektren zur PDP-11 verantwortlich ist. Dieser Teil der Software ist der komplexeste und findet deshalb am meisten Beachtung. Bei den Verwaltungsprogrammen auf der PDP-11 werden standard Programmiertechniken benutzt, wie sie bei jedem Datenerfassungssystem eingesetzt werden. Die Darstellung dieses Teiles ist ensprechend kurz.

8.1 Das Ausleseprogramm

Das Ausleseprogramm im Mikrocomputer ist der einzige Teil der Software, der von den Eigenschaften des Experimentes abhängt. Dieses Programm muß wissen, welche CAMAC Einheiten eingesetzt werden, wie viele Detektoren benutzt werden und welche Präzision die Spektren haben sollen. Bei einer Änderung des experimentellen Aufbaus, z.B. durch Einsatz eines zusätzlichen Detektors, muß daher nur das Ausleseprogramm geändert werden. Die bei unserem jetzigen Experiment eingesetzten CAMAC Einheiten sind in Abb. 20 gezeigt.

Wie bereits erwähnt werden 6 Halbleiterdetektoren benutzt, deren Energie- und Zeitinformationen ausgelesen werden müssen. Ein Detektor liefert einen

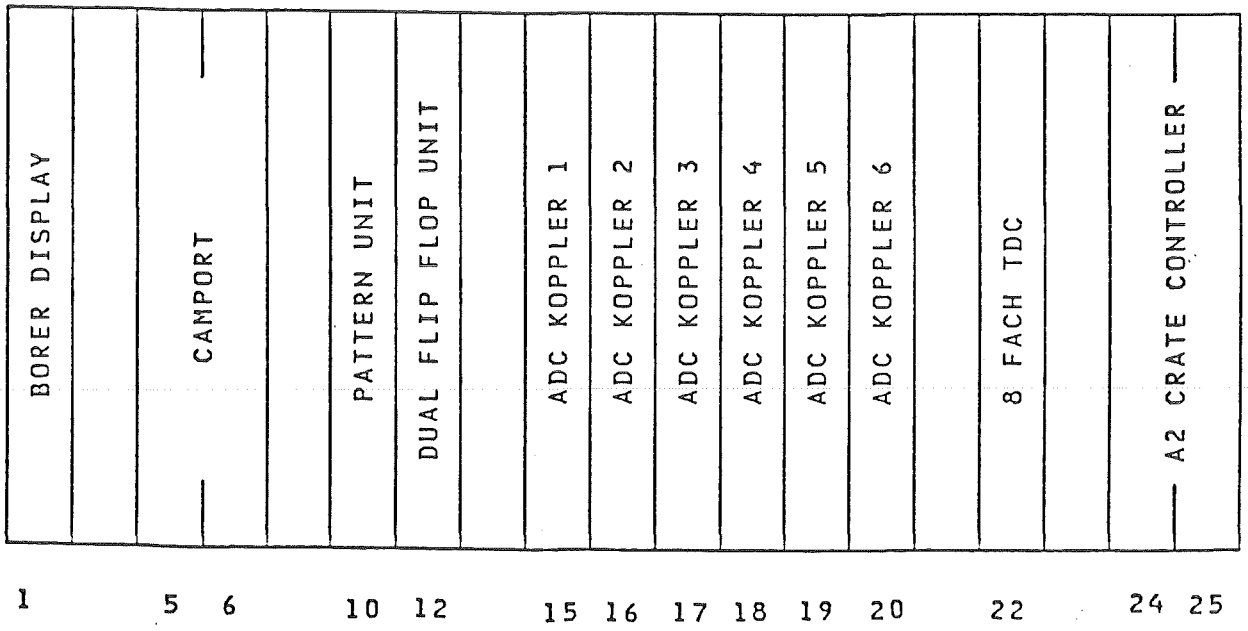


Abbildung 20: CAMAC Einheiten

Spannungsimpuls dessen Höhe proportional zur Energieabgabe des Röntgenquantes im Detektor ist. Dieser Spannungsimpuls wird verstärkt und mit Hilfe eines Analog-zu-Digitalwandlers (Analog to Digital Converter, ADC) in digitale Information umgesetzt. Auf Grund ihrer besonders guten Linearität werden NIM ADCs eingesetzt (Laben 8215). Diese ADCs benötigen spezielle Interfaces (ADC Koppler), um mittels CAMAC ausgelesen werden zu können. Eine Beschreibung der ADC Koppler befindet sich in Anhang B. Auf die gleiche Weise werden die Zeitinformationen mit Hilfe von Zeit-zu-Digitalwandlern

(Time to Digital Converter, TDC) für die Auslese aufbereitet. Dies geschieht mit einer CAMAC Einheit, welche 8 TDCs beinhaltet. Es werden zwei weitere CAMAC Moduln benutzt, um eine höhere Auslesegeschwindigkeit zu erreichen. Eine Dual Flip Flop Einheit (beschrieben in Anhang A) zeigt dem MC 68000 an, daß ein Ereignis aufgezeichnet wurde und eine Pattern-Unit ermittelt, welcher Detektor gefeuert hat. Der Ablauf der Auslese soll nun an Hand des Flußdiagrammes (Abb. 21) erläutert werden.

Ein Trigger wird generiert, sobald ein Antiproton im Target gestoppt wird und einer der Halbleiterdetektoren angesprochen hat. Der Trigger setzt das zweite Flip Flop (FF) der Dual Flip Flop Einheit. Der Ausgang des FF2 wird benutzt, um weitere Trigger zu unterdrücken, bis die Verarbeitung des Ereignisses abgeschlossen ist. Nachdem alle ADCs und TDCs konvertiert haben, wird das erste Flip Flop gesetzt, um dem Mikroprozessor anzuzeigen, daß das Ereignis zum Auslesen bereit steht. In der Triggerschleife wird dauernd der Zustand des ersten FFs abgefragt. Sobald das FF1 gesetzt, ist wird der Auslesevorgang gestartet und das FF1 zurückgesetzt (CAMAC Befehl F2). Danach wird die Pattern-Unit ausgelesen, um festzustellen welcher der Detektoren gefeuert hat. Die Bedeutung der einzelnen Pattern-Unit Bits kann aus Abb. 22 ersehen werden.

Ist Bit 15 gesetzt, so hat mehr als ein Detektor angesprochen. Es handelt sich also um ein Koinzidenzereignis. In diesem Fall wird das Pattern-Unitwort in einen Ereignispuffer geschrieben (siehe Abb. 23), der für Koinzidenzereignisse vorgesehen ist. Durch Analysieren der Pattern-Unitbits kann festgestellt werden, welcher der Detektoren angesprochen hat, d.h welcher der ADCs und TDCs ausgelesen werden muß. Alle Geräte werden mit CAMAC F2 Befehlen ausgelesen und daher sofort wieder zurückgesetzt. Die gelesenen Werte werden ebenfalls in der Ereignispuffer übertragen. Dies geschieht, solange bis alle 6 Detektorbits überprüft sind. Zum Abschluß wird das zweite FF zurückgesetzt, um weitere Trigger zu erlauben.

Meistens spricht jedoch nur ein Detektor an und das Koinzidenzbit ist nicht gesetzt. In diesem Fall wird ebenfalls das Pattern-Unitwort untersucht, um

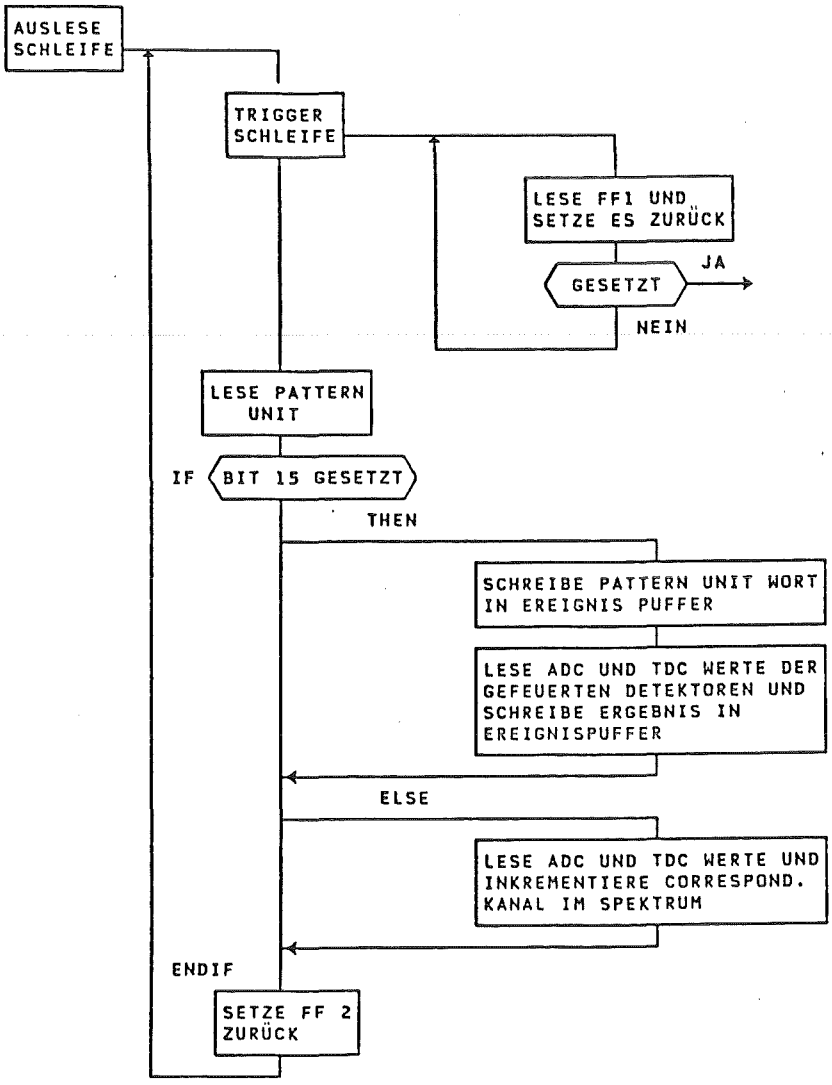
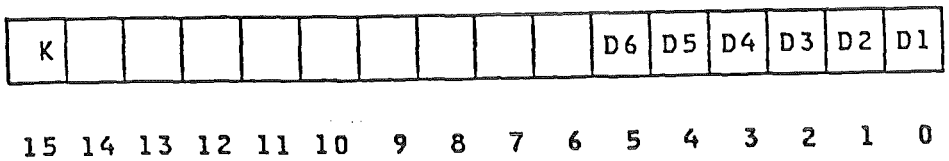


Abbildung 21: Flußdiagram des Ausleseprogrammes



K: Koinzidenzbit, gesetzt, wenn mehr als ein Detektor angesprochen hat.
Dx: Bit zeigt an daß Detektor Nr. x angesprochen hat.

Abbildung 22: Bedeutung der Pattern-Unit Bits

herauszufinden, welcher ADC und TDC ausgelesen werden muß. Mit der Detektornummer und den ADC und TDC Werten wird eine Adresse gebildet, die einem Kanal in dem zum Detektor gehörenden Spektrum entspricht. Der Kanalinhalt wird inkrementiert und die Verarbeitung des Ereignisses wird durch Zurücksetzen des zweiten FFs abgeschlossen. Von nun an wird wieder das FF 1 beobachtet, d.h. auf ein neues Ereignis gewartet.

Für die Auswertung der Spektren ist es wichtig, außer den Daten selbst, auch Informationen über deren Struktur abzulegen. So muß die Anzahl und Länge der Spektren sowie die Länge des Ereignispuffers festgehalten werden.

Da nur der Mikrocomputer die Eigenschaften des Experimentes kennt, ist es das Ausleseprogramm, welches diese Kontrollinformationen zur Verfügung stellen muß. Dies geschieht mit Hilfe eines "Data Aquisition Control Blocks" (DACB), der in Abb. 23 dargestellt ist. Dieser DACB ist Teil des Datenbereiches im VME System und wird gemeinsam mit den Spektren zum Minicomputer übertragen. Der DACB erlaubt nicht nur variable Länge des Puffers für die Koinzidenzereignisse, sondern auch variable Anzahl und Länge der Spektren. Dadurch wird es sehr einfach, neue Detektoren ins System aufzunehmen, da die Verwaltungsprogramme in der PDP-11 diese Informationen lesen können und dementsprechend mehr oder weniger Spektren behandeln können. Ein Einbau oder die Herausnahme eines Detektors hat daher nur eine Änderung des Ausleseprogrammes in Mikrocomputer zur Folge.

8.2 Gemessene Auslesezeiten

Um ein Ereignis auszulesen, sind, wie besprochen, mindestens 5 CAMAC Zugriffe notwendig. Es muß der Trigger erkannt werden (Lesen des FF1), es muß festgestellt werden, welcher Detektor gefeuert hat (Lesen der Pattern Unit) und es müssen die Daten ausgelesen werden (Lesen des ADCs und TDCs). Zuletzt muß das Datenerfassungssystem für neue Ereignisse freigegeben werden (Zurücksetzen des FF2). Die Freigabe geschieht sofort nach der Auslese der Daten, sodaß das Einsortieren in das korrespondierende Spektrum und die Verarbeitung des folgenden Ereignisses durch die Analogelektronik und durch die ADCs und TDCs parallel erfolgen kann.

Die Zeit zwischen der Ankunft eines Triggers und der Freigabe des Systemes wurde für Einzelereignisse gemessen. Sie beträgt 35 µs. Das bedeutet, daß Datenraten bis zu 10^4 Hertz ohne nennenswerte Totzeit verkraftet werden, was den Spezifikationen von Hardware-Vielkanalanalysatoren nahekommt.

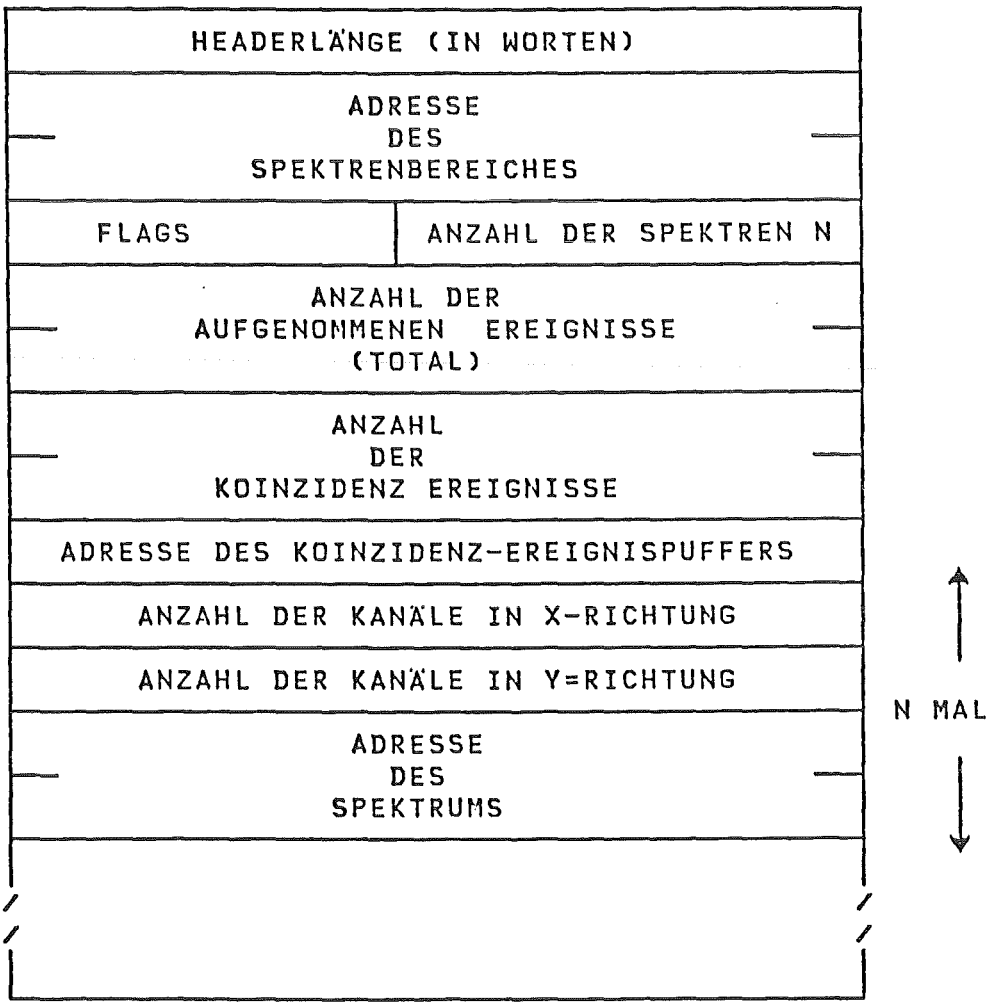


Abbildung 23: Data Aquisition Control Block

8.3 Die Kommunikationssoftware

Um das System wie einen Vielkanalanalysator aussehen zu lassen, muß man die normalen Vielkanalanalysatorfunktionen von einem PDP-11 Terminal ausführen können. So muß man die Möglichkeit haben, die Datennahme zu starten und anzuhalten, die Spektren zu transferieren und anzuschauen etc., ohne am VME System eingreifen zu müssen. Dazu ist ein Programmpaket erforderlich, das die Kommunikation zwischen VME System und PDP-11 abhandelt.

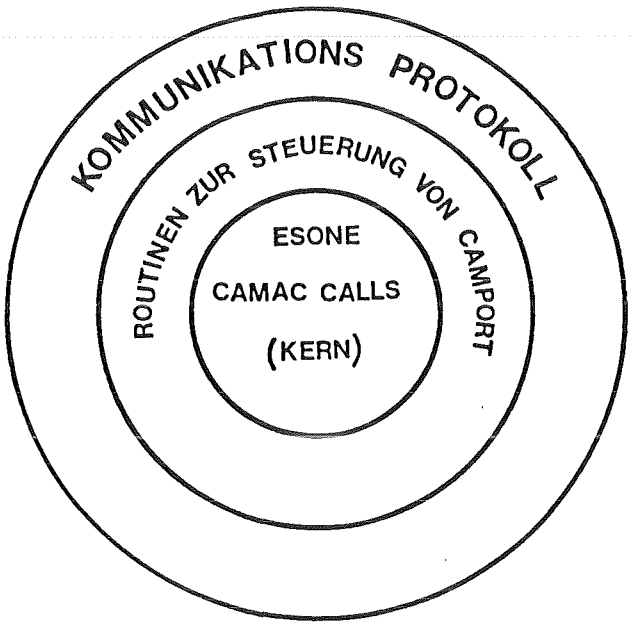


Abbildung 24: Struktur der Kommunikationssoftware

Die Kommunikationsprogramme auf der PDP-11 bestehen aus drei Niveaus (siehe Abb. 24). Der Kern wird gebildet von einer Sammlung von Routinen, welche die zur Ausführung von CAMAC Befehlen notwendigen I/O Zugriffe durchführen. Die Aufrufe der Routinen wurden vom ESONE Kommitee standardisiert und sind auf allen Computern gleich [EUR90]. (CALL CFSA(F,EXT,Q) führt die CAMAC Funktion F auf Adresse EXT aus) Das zweite Niveau ist für die Steuerung von CAMPORT zuständig und greift auf die Kernroutinen zurück. Hier werden CAMPORT's CAMAC Funktionen ausgeführt, wie Busrequest, Laden des MAR, Transfer der Daten, Busrelease, Interrupt etc. Nur die Programme dieses Niveaus kennen die Eigenschaften von CAMPORT und kümmern sich um alle Details der notwendigen Einzelaktionen. Das dritte Niveau wird durch die Protokollsoftware gebildet, welche in den nächsten drei Abschnitten ausführlich erklärt ist. Es ist dafür verantwortlich, daß der Informationsaustausch zwischen den beiden Maschinen in geordneter Weise vor sich geht. Die Protokollsoftware greift auf die unterliegenden Niveaus zurück und braucht sich daher um Hardwaredetails nicht zu kümmern. Im vierten Niveau schließlich befinden sich die Routinen, welche das Interface zum Benutzer des Systemes bilden. Soll z.B. ein Datenblock der Länge JLEN von der Adresse JLOC im VME System in den Array IAREA der PDP-11 übertragen werden, so soll sich der Programmierer nicht darum kümmern zu müssen, in welches Feld des Kommunikationsbereiches er was zu schreiben hat.
Ein Aufruf:

```
CALL UPREAD(JLOC,JLEN,IAREA)
```

soll die geforderte Aufgabe erfüllen.

Die Interfaceroutinen wie UPREAD rufen, für den Benutzer transparent, verschiedene Routinen der Protokollsoftware auf, z.B.

```
CALL UPDUMP(IARRAY,LENGHT)
```

Die Protokollroutinen wie UPDUMP beschreiben die Felder des Kommunikationsbereiches im geforderten Format, unter Benutzung der Kernroutinen.

8.4 Die Protokollsoftware

Wie bereits in der Einleitung erwähnt, werden die aufgesammelten Spektren während der Strahlpause vom VME System zur PDP-11 transferiert und dort auf Platte geschrieben. Da CAMPORT nur über 16 Adressleitungen verfügt, können maximal 128 kBytes an VME Speicher gelesen werden. Das bedeutet, daß der Mikroprozessor den zu transferierenden Teil des Spektrums zunächst in einen Speicherbereich kopieren muß, der CAMPORT zugänglich ist. Um sowohl den Kopiervorgang als auch den Transfer in geordneter Weise durchführen zu können, wurde ein Kommunikationsprotokoll mit Handshake Mechanismus erstellt. Mit diesem Protokoll ist es nicht nur möglich, die Datentransfers durchzuführen, man kann außerdem Programme in Maschinensprache von der PDP-11 in das VME System laden und dort starten und anhalten. Es können daher alle für die Datennahme notwendigen Aktionen vom Minicomputer gesteuert werden, sodaß am VME System auf ein Terminal verzichtet werden kann.

Der gesamte Datenaustausch zwischen den beiden Maschinen geht über das FEDR in CAMPORT und über den von CAMPORT ansprechbaren Speicherbereich im VME System, da dies der einzige Speicherbereich ist, zu dem sowohl die PDP-11 als auch der MC 68000 Zugang hat. Dieser Speicherbereich wird im folgenden mit "Kommunikationsbereich" bezeichnet. Er ist ausschließlich für den Austausch von Informationen zwischen Mikro- und Minicomputer reserviert und wird nicht zum Aufbau der Spektren benutzt. Abbildung 25 zeigt wie die Informationen im Kommunikationsbereich abgelegt werden. Am Anfang des Bereiches befindet sich ein Headerblock von 30 Worten Länge, der zum Austausch von Befehlen und Anforderungen benutzt wird. Der Rest ist als Datenbereich für den Transfer der Spektren vorgesehen.

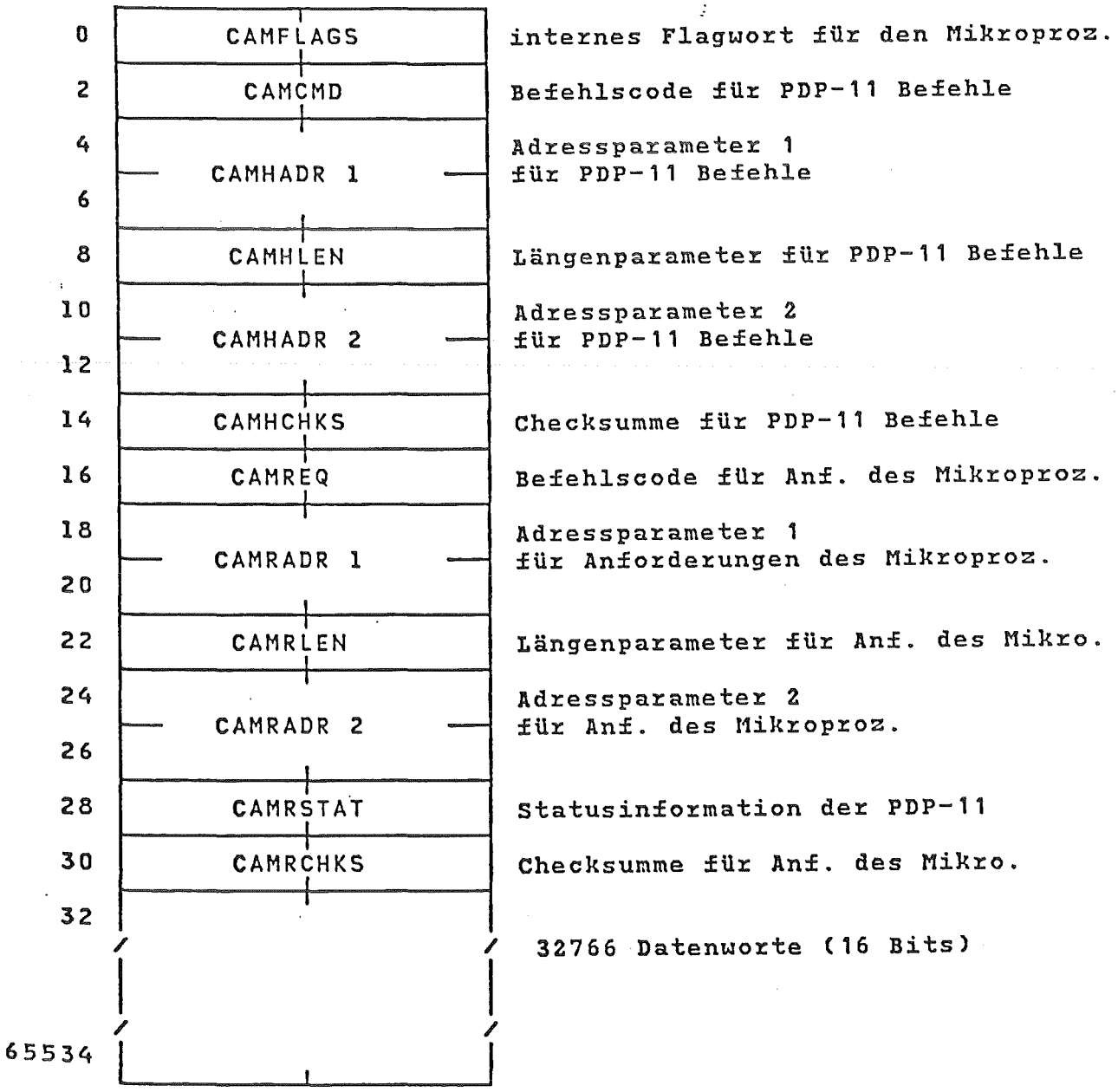


Abbildung 25: Kommunikationsbereich

8.5 Absetzen eines Befehles

Will die PDP-11 einen Befehl an den MC 68000 absetzen, so schreibt sie zunächst den Befehlscode mit den benötigten Parametern in den dafür vorgesehenen Teil des Kommunikationsbereiches. Das höchst signifikante Bit des Befehlscodes gibt an, ob eine Antwort des Mikroprozessors erwartet wird oder nicht (Bit=1: Antwort erwartet, Bit=0: Antwort nicht notwendig). Die Ausführung des Befehles wird durch einen Kommunikationsinterrupt ausgelöst. Der Mikroprozessor unterbricht daraufhin das gerade in Aktion befindliche Programm und liest, interpretiert und exekutiert den geforderten Befehl. Ist eine Rückantwort gefordert, so schreibt der MC 68000 ein Statuswort in das FEDR und setzt das CLAM. Das Statuswort besteht aus dem Befehlscode, falls das Kommando erfolgreich durchgeführt werden konnte, bzw. einem Fehlercode, wobei durch Setzen des höchst signifikanten Bits die Fehlerbedingung angezeigt wird. Der Ablauf eines solchen Befehles wird in Abb. 26 graphisch dargestellt.

Selbstverständlich müssen die Mikroprozessorprogramme, welche die Befehle bearbeiten ständig im Mikrocomputer vorhanden sein. Die Protokollsoftware ist daher in EPROMS (Erasable Programmable Read Only Memories) gespeichert. Beim Einschalten des VME Systemes wird zunächst der Spektrumsspeicher initialisiert und dann sofort die Kontrolle der Kommunikationssoftware übertragen, die auf Befehle der PDP-11 wartet. Selbst das Ausleseprogram wird erst mittels der Kommunikationsprogramme in den VME Speicher geladen und gestartet. Dadurch kann das Ausleseprogramm, was ja besonders zeitkritisch ist, in sehr schnelle RAM (Random Access Memory) Bausteine geladen und dort ausgeführt werden (150 ns Zugriffszeit), was eine Erhöhung der Auslesegeschwindigkeit zur Folge hat. Die EPROM Bausteine haben nur Zugriffszeiten von 450 ns. Außerdem müssen bei einer Änderung der experimentellen Anforderungen keine neuen EPROMs mit dem modifizierten Ausleseprogramm hergestellt werden.

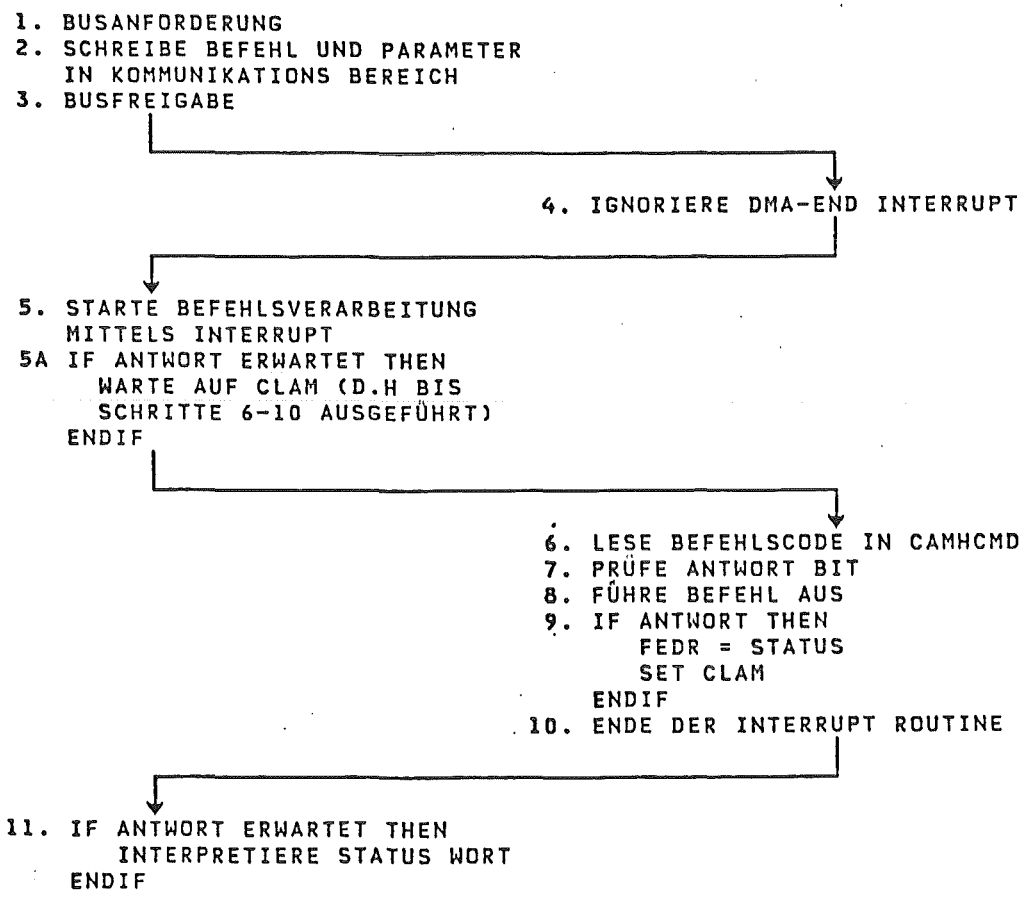


Abbildung 26: Befehl der PDP-11 an den Mikrocomputer

8.6 Anforderungen des Mikroprozessors an den Minicomputer

Es ist möglich, daß der MC 68000 zur Ausführung eines Befehles bestimmte Informationen von der PDP-11 benötigt, oder daß ein ein Fehler, der im Mikrocomputer auftritt (z.B. Paritätsfehler im Speicher) dem Minicoputer gemeldet werden muß. Eine solche Anforderung geschieht durch Setzen des CLAM in CAMPORT. Es kann zusätzlich der Grund der Anforderung im FEDR übertragen werden. Abb. 27 zeigt den zeitlichen Ablauf einer Anforderung.

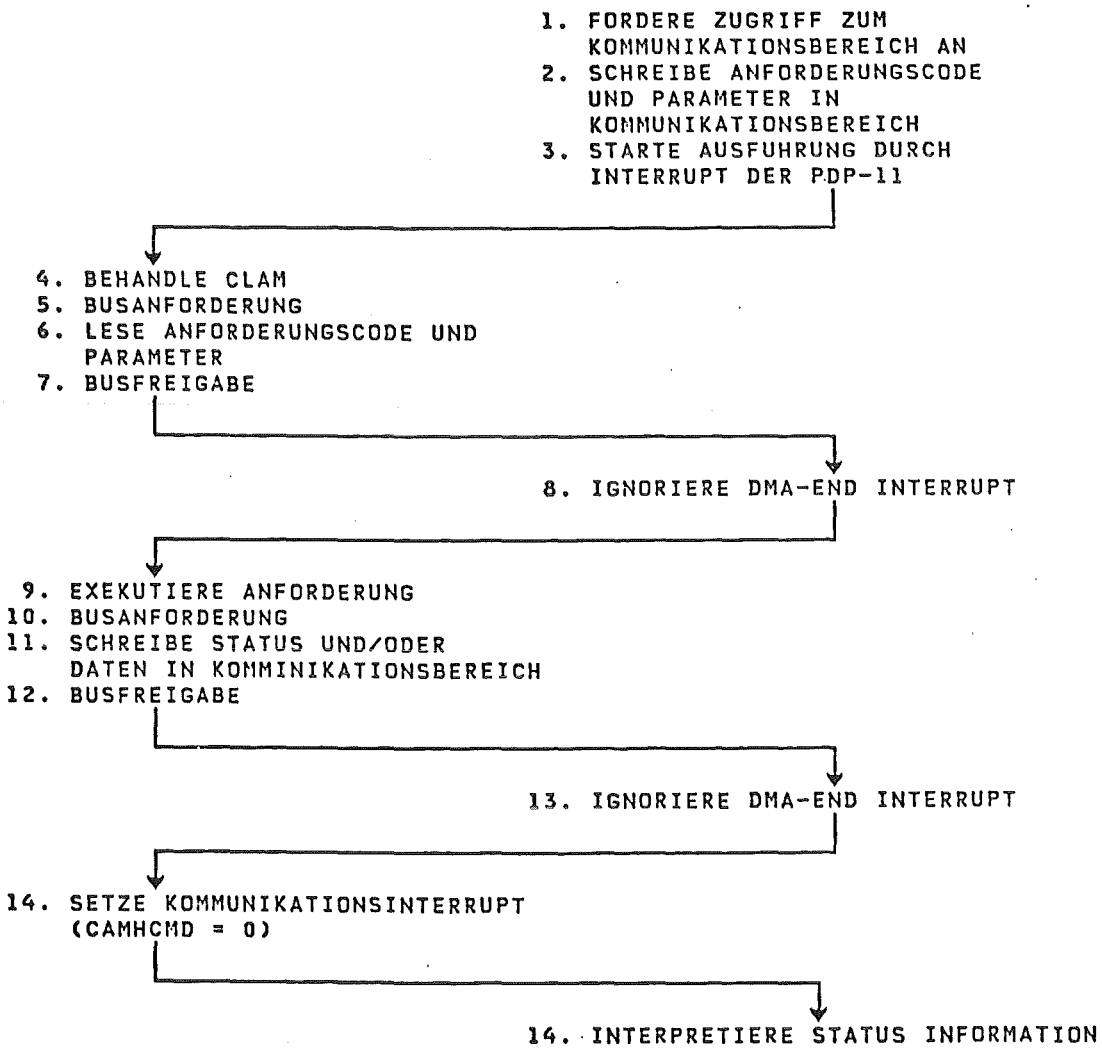


Abbildung 27: Anforderung an den Minicomputer

8.7 Die Verwaltungsprogramme

Die Spektren werden, wie im Vorhergehenden gezeigt wurde, Block für Block zur PDP-11 transferiert und dort auf Band geschrieben. In der PDP-11 werden die zweidimensionalen Spektren außerdem durch Aufaddierung in eindimensionale Energie- bzw. Zeitspektren umgewandelt und auf die Magnetplatte geschrieben. Der gesamte Transfer des 2 MBytes Speicherbereiches dauert ca. 3 Minuten, falls keine anderen Programme unter dem Mehrprogrammsystem der PDP-11 aktiv sind. Die eindimensionalen Spektren können dann auf Drucker und Graphikterminals angeschaut werden. Abb. 28 zeigt ein solches Online Spektrum.

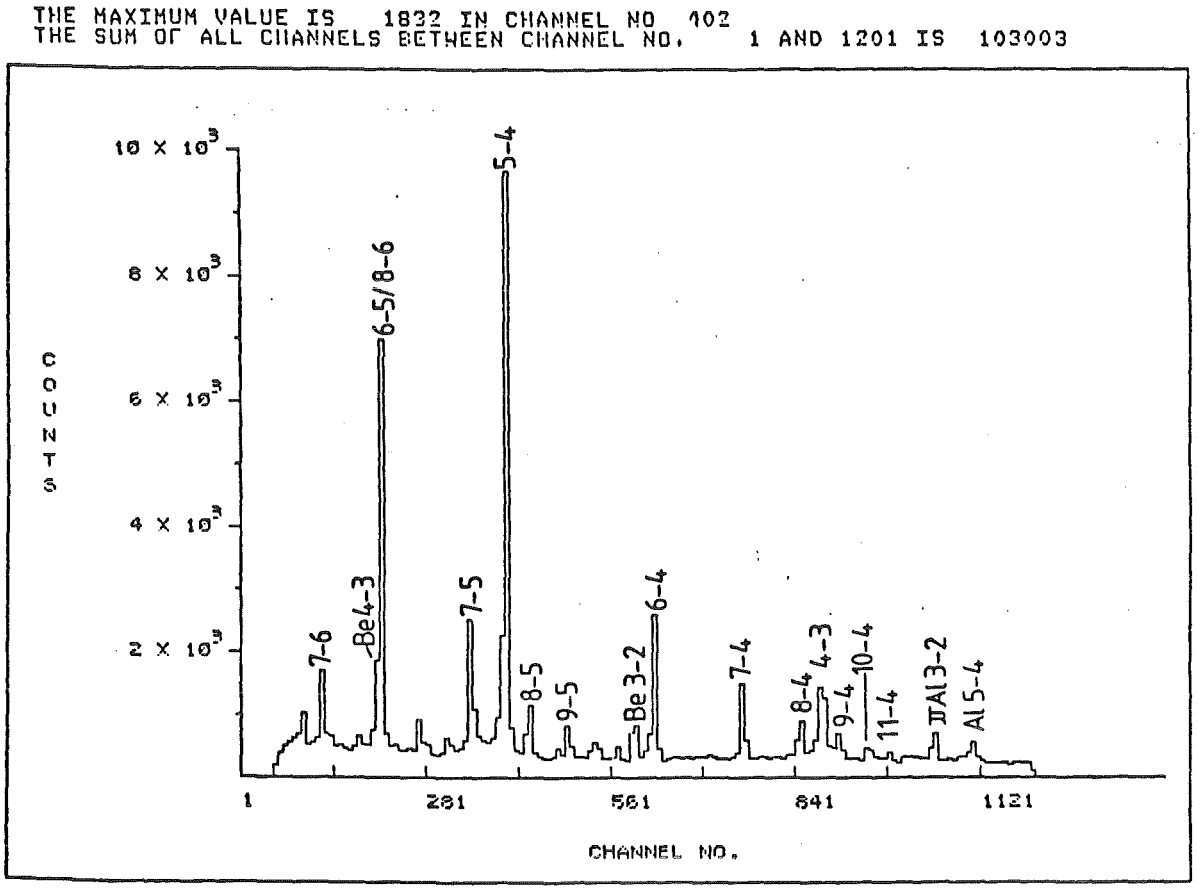


Abbildung 28: Online Spektrum von $\bar{p}^{16}O$. Erstes bei LEAR gemessenes Röntgenspektrum antiprotonischer Atome

KAPITEL 9

ERSTE ERGEBNISSE

Nach einem Testlauf am LEAR im Juli 83 wurde das Meßsystem Mitte Dezember 83 zum ersten Mal unter realen Runbedingungen eingesetzt. Abb. 28 zeigt ein Spektrum, welches während dieses Runs aufgenommen wurde. Dabei wurden Antiprotonen in einem ^{16}O (isotopenreines $H_2{}^{16}O$) Target gestoppt und die Energien der enstehenden Röntgenquanten mit Hilfe einer planaren Germaniumdiode gemessen. Die Anzahl der Ereignisse im Spektrum entspricht ca. 10^8 $\bar{p}$-Stops und einer effektiven Meßzeit von ca. einer Stunde (1 Spill). Für dieselbe Statistik wären an einem konventionellen $\bar{p}$-Strahl mehrere Monate Strahlzeit notwendig.

Das Spektrum zeigt die $\bar{p}$-Kaskade im ^{16}O. Die Linien ohne Elementangabe stammen alle aus dem Sauerstoff. Von besonderem Interesse dabei ist die durch die starke Wechselwirkung intensitätsgeschwächte und verbreiterte ^{16}O 4-3 Linie. Aus der Energie, der Intensität und der Linienform dieser Linie können Rückschlüsse auf die Natur der starken $\bar{p}$-Kern Wechselwirkung und auf Eigenschaften des ^{16}O Kernes gezogen werden (siehe Kap. 1).

Außer den Sauerstofflinien sind nur sehr kleine Beiträge von Störlinien aus dem Beryllium der Targethülle und dem Aluminium der Diodenhalterung zu erkennen.

Das Spektrum war in dieser Form Online zu sehen und enthält noch keinerlei Offline Behandlung.

Bei diesem Experiment wurde eine Ereignisrate von 2500 Ereignissen pro Sekunde bei einer Systemtotzeit von weniger als 5% gemessen. Das Datenerfassungssytem hat daher das Designsoll erfüllt und ist für weitere Messungen bereit.

KAPITEL 10

ERWEITERUNGEN IN DER ZUKUNFT

Im Gegensatz zu konventionellen Vielkanalanalysatoren steht bei dem beschriebenen Datenerfassungssystem kein Life Display (graphischer Bildschirm auf dem bewegte Bilder dargestellt werden können) zur Darstellung der Spektren zur Verfügung. Die gemessenen Daten müssen grundsätzlich zuerst zur PDP-11 transferiert werden, bevor sie auf Bildschirmgeräten dargestellt werden können. Besonders zur Einstellung der Experimentierelektronik ist ein Life Display jedoch sehr von Nutzen, da der Effekt jeder Änderung sofort sichtbar wird. Es ist daher geplant, einen Life Display zu entwickeln und in das System zu inkorporieren.

10.1 Der Life Display

Ein Life Display besteht im wesentlichen aus einem großen Speicher (Bildspeicher), dessen Inhalt 1:1 auf einen Bildschirm abgebildet wird. Bei jedem Durchlauf des Elektronenstrahls über den Schirm wird der aktuelle Inhalt des Speichers auf den Bildschirm kopiert, sodaß jedes gesetzte Bit im Speicher einen weißen Punkt auf dem Bildschirm erzeugt. Abb. 29 zeigt die Korrespondenz zwischen Speicher und Bildschirm.

Da man den Bildspeicher in wesentlich kürzerer Zeit beschreiben kann als zum Auffrischen des Bildschirms benötigt wird, ergibt sich der Eindruck eines bewegten Bildes (daher der Name Life Display). Die einfachste Weise einen Life Display in das bestehende System einzubauen, besteht darin, einen Bildspeicher zur Verfügung zu stellen, der vom Mikroprozessor angesprochen werden kann. Außerdem wird ein Interface für den Bildschirm benötigt.

Dadurch wird der Mikroprozessor jedoch mit dem Auffrischen des Bildspeichers zusätzlich belastet und damit die Auslesegeschwindigkeit gedrosselt. Jede zusätzliche Funktion, die in den Life Display eingebaut wird,

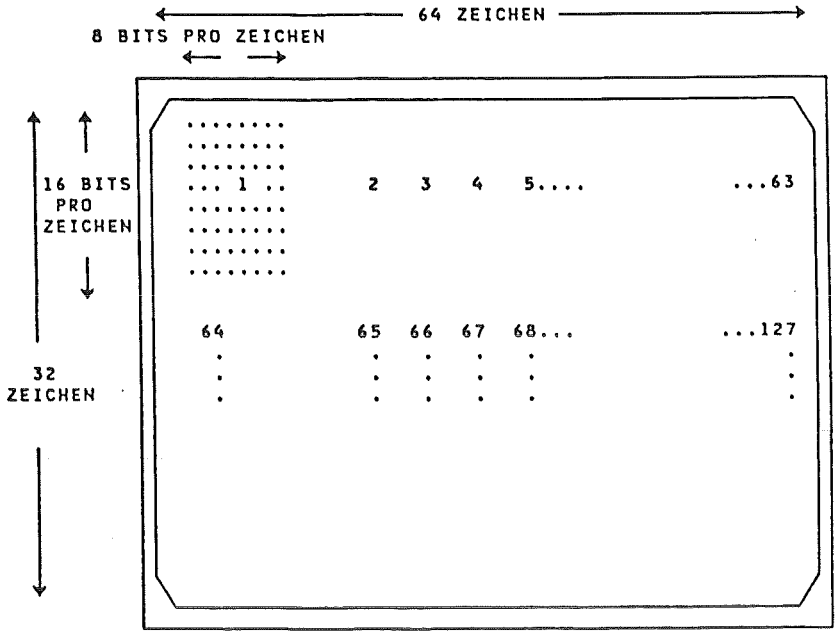


Abbildung 29: Korrespondenz zwischen Speicher und Bildschirm

resultiert in einer Verringerung der Auslesegeschwindigkeit des gesamten Systemes. Es ist daher vorteilhaft, den Life Display mit einem eigenen Prozessor auszustatten, der vollkommen unabhängig von Ausleseprozessor arbeitet.

10.2 Der Busspion

Für den Prozessor des Life Displays besteht das Problem, herauszufinden, wann ein neues Ereignis in das darzustellende Spektrum einsortiert wird, d.h wann der Bildspeicher aufgefrischt werden muß. Da der Ausleseprozessor auf den Spektrumsspeicher zugreifen muß, um das ausgelesene Ereignis einzusortieren, kann der Prozessor des Life Displays diesen Zugriff durch "Spionieren" auf dem VME Bus feststellen. Um verschiedene Full Scales der Spektren zu erlauben, baut der Life Display seine eigenen lokalen Spektren auf, indem er, bei jedem Zugriff des Ausleseprozessors auf das darzustellende Spektrum, die angesprochene Adresse herausfindet und den entsprechenden Kanal in seinem lokalen Spektrum inkrementiert. Es wird dann dieses lokale Spektrum vom Life Display dargestellt. Selbstverständlich ist es nicht möglich, 2-dimensionale Spektren mit voller Präzision, wie sie im Spektrumspeicher vorliegen, lokal im Life Display aufzubauen. Durch geschickte Wahl der Adresskomparatoren im Busspion ist es jedoch möglich, verschiedene Arten von 1-dimensionalen Spektren darzustellen. So können z.B. die Projektionen der 2-dimensionalen Spektren auf die Energie- bzw. Zeitachse dargestellt werden. Außerdem sind Schnitte durch das Spektrum sowohl in der Zeit- als auch in der Energieachse möglich.

10.3 Eingabe

Um die Art des Spektrums anwählen zu können, ist eine Eingabetastatur mit 24 Tasten vorgesehen. Damit kann außerdem die Full Scale des Spektrums, Spreizen des Spektrums, Cursor etc. gehandhabt werden. Die Möglichkeiten des Life Displays hängen nur noch von der Phantasie (und Zeit) des Programmierers ab, der das Kontrollprogramm für den Prozessor des Life Displays schreibt. Abb. 30 zeigt ein Blockschema des geplanten Life Displays.

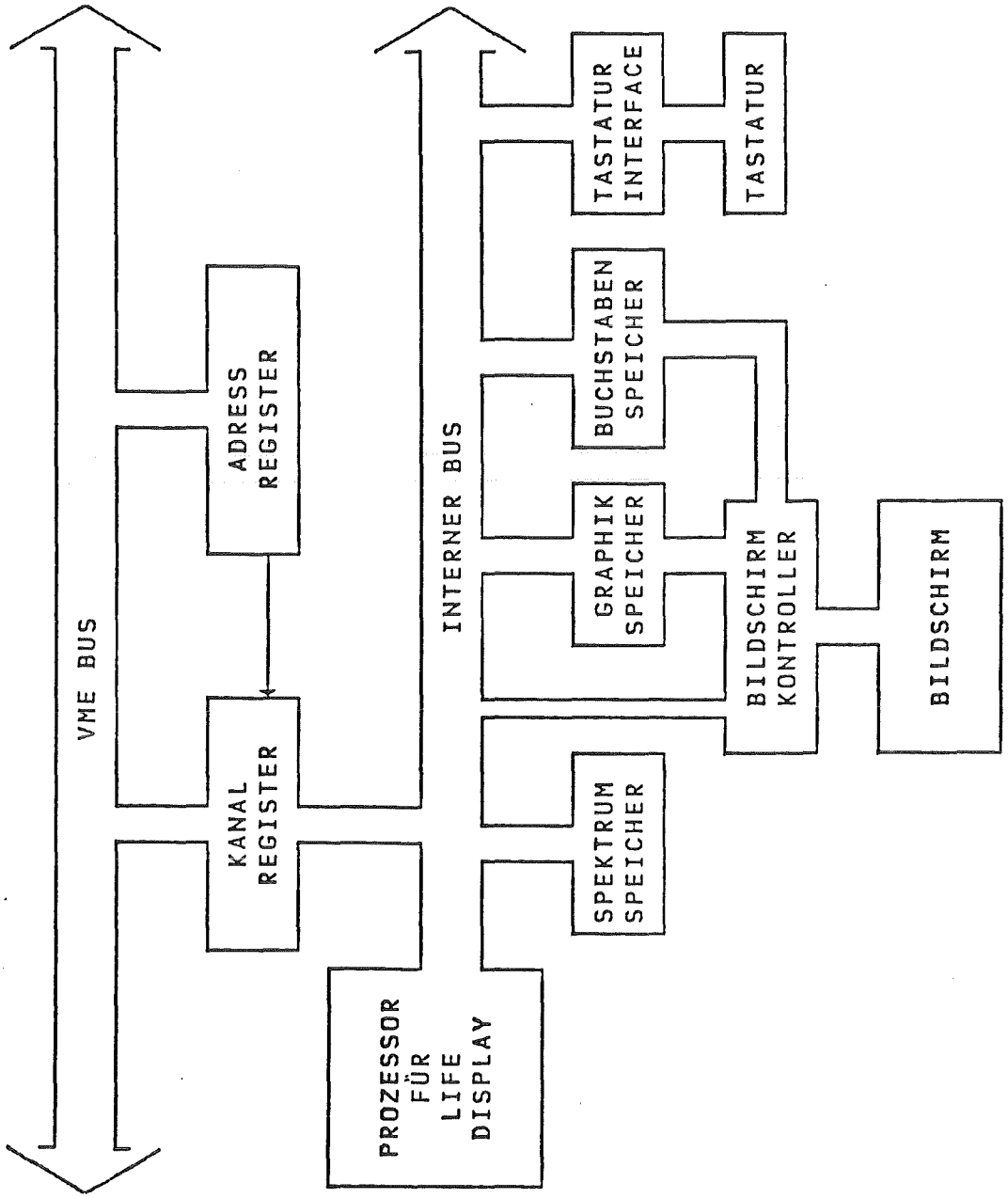


Abbildung 30: Block Diagramm des geplanten Life Displays

KAPITEL 11

WEITERE ANWENDUNGSMÖGLICHKEITEN DES SYSTEMES

Bisher wurde gezeigt, wie mit dem VME System, CAMPORT, CAMAC, PDP-11 etc. ein Datenerfassungssystem aufgebaut werden kann, das mehrere Vielkanalanalysatoren hoher Präzision emuliert. Dies jedoch ist nur eine von vielen Anwendungsmöglichkeiten, die mit der oben genannten Hardware möglich sind.

Um die hohe Flexibilität des Systemes zu demonstrieren, wird in diesem Kapitel eine Anwendung beschrieben, bei der eine Vieldrahtkammer ausgelesen werden mußte, die an einem gepulsten Strahl arbeitete (K23 am CERN PS, 1 Puls pro 2 sec.). Auch bei dieser Anwendung wurde besonderer Wert auf hohe Auslesegeschwindigkeit gelegt.

11.1 Die Vieldrahtkammer

Bei der Vieldrahtkammer handelte es sich um eine zylindrische Driftkammer. Sie bestand aus 15 Moduln, von denen jedes 5 Auslesedrähte enthielt. In Abb. 31 ist ein solches Modul gezeigt.

Ein durch die Driftkammer fliegendes Teilchen ionisiert das in der Kammer befindliche Gas (Primärionisation). Durch elektrische Felder innerhalb der Kammer werden die bei der Primärionisation entstehenden Elektronen zum positiven Auslesedraht gezogen, während die Ionen zur negativen Kammerwand abdriften. Da in der Nähe des sehr dünnen Auslesedrahtes (15 µm) sehr hohe Felder herrschen, kann dort das angezogene Elektron zwischen zwei Stößen mit Gasatomen so viel Energie aufnehmen, daß es das Gasatom wiederum ionisieren kann (Sekondärionisation). Die dabei entstehenden Elektronen werden wiederum so stark beschleunigt, daß sie das Gas von neuem ionisieren. Dadurch entsteht eine Elektronenlawine, deren am Auslesedraht induziertes Signal gemessen werden kann.

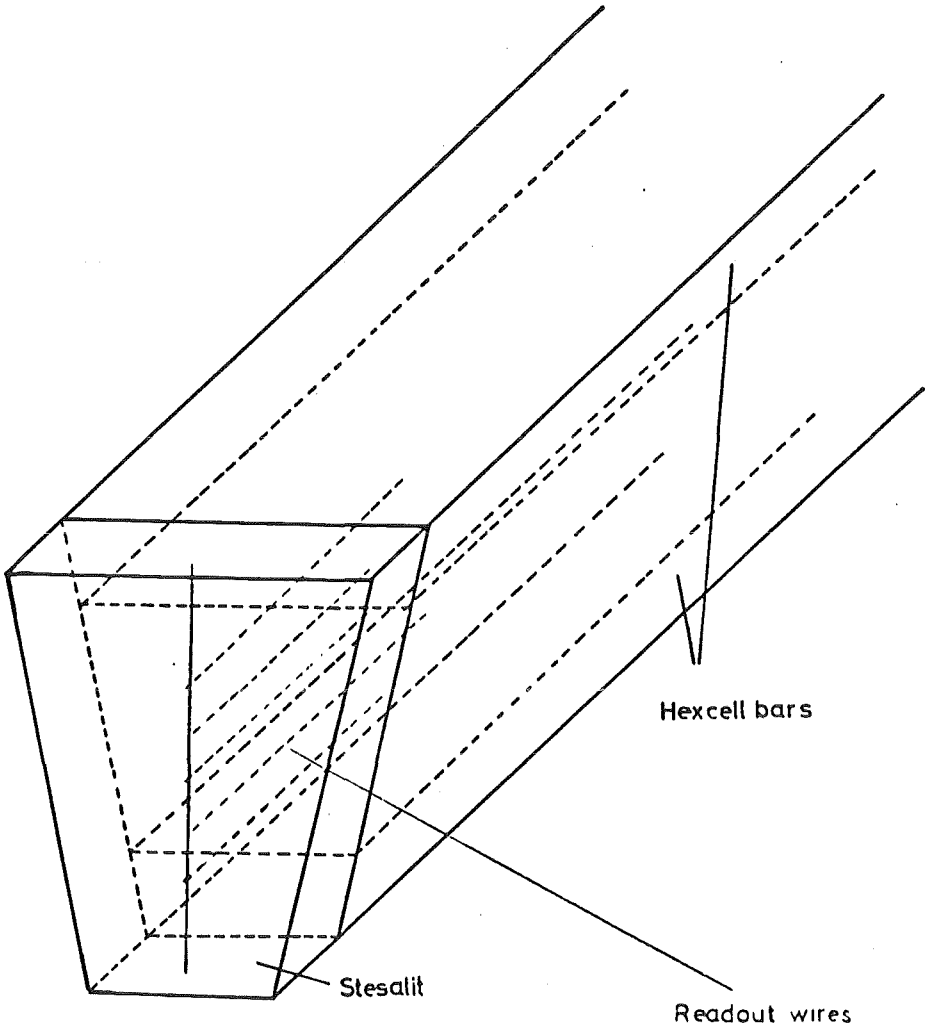


Abbildung 31: Driftkammer Modul

Bei der Driftkammer wird die Zeit zwischen dem Durchfliegen des Teilchens und der Ankunftszeit der Elektronenlawine am Auslesedraht gemessen. Da die Driftgeschwindigkeit der Elektronen aus der Primärionisation nahezu konstant ist, kann aus der gemessenen Driftzeit auf den Abstand des Teilchentrajektorie zum Auslesedraht geschlossen werden. Um unterscheiden zu können, ob das Teilchen links oder rechts vom Draht die Kammer durchquerte, sind die Auslesedrähte um jeweils 250 µm zur Mitte versetzt. Ein Fit durch die gemessenen Zeiten erlaubt dann die Zweideutigkeit aufzulösen.

Bei zylindrischen Kammern hat man außerdem das Problem, den Ort der Trajektorien entlang des Auslesedrahtes festzulegen. In der benutzten Kammer wird deshalb ein Widerstandsdraht als Auslesedraht verwendet (ca. 2 kΩ/m) und das Signal an beiden Enden des Drahtes abgegriffen. Das Verhältnis der gemessenen Ladungen läßt damit auf den Entstehungsort der Ladung schließen. Jeder Draht benötigt daher einen Verstärker an beiden Enden. Das Signal der Verstärker muß mittels ADCs in digitale Informationen umgewandelt werden. Außerdem muß ein Signal für die Zeitmessung abgegriffen werden, das nach einem Diskriminator in einen TDC gegeben wird. Abb. 32 zeigt die für einen Draht notwendige Elektronik. Eine gute Einführung in die Funktionsweise von Vieldrahtkammern wird in [SAU77] gegeben.

11.2 Anschluß der Kammer an das Datenerfassungssystem

Wie oben gesehen werden für jeden Draht 2 ADCs und ein TDC benötigt. Es mußten daher für die insgesamt 75 Drähte 150 ADCs und 75 TDCs zur Verfügung stehen. Als ADCs wurden LRS 2249W 12-fach ADCs verwendet, während für die Zeitmessungen ein Drifttimedigitizer (DTD LRS 2770A) eingesetzt wurde. Der DTD konvertiert die Zeiten der gefeuerten Drähte (max. 40) und schreibt die kodierte Drahtadresse mit in das Datenwort. Das Datenerfassungssystem muß daher 150 ADC Kanäle und bis zu 40 TDC Worte auslesen, die Ergebnisse auf Band schreiben und zur graphischen Darstellung in Spektren einsortieren. Im Gegensatz zur Anwendung als Vielkanalanalysator kann ein Wort bei der Kammerauslese bis zu 190 Informationsstücke enthalten,

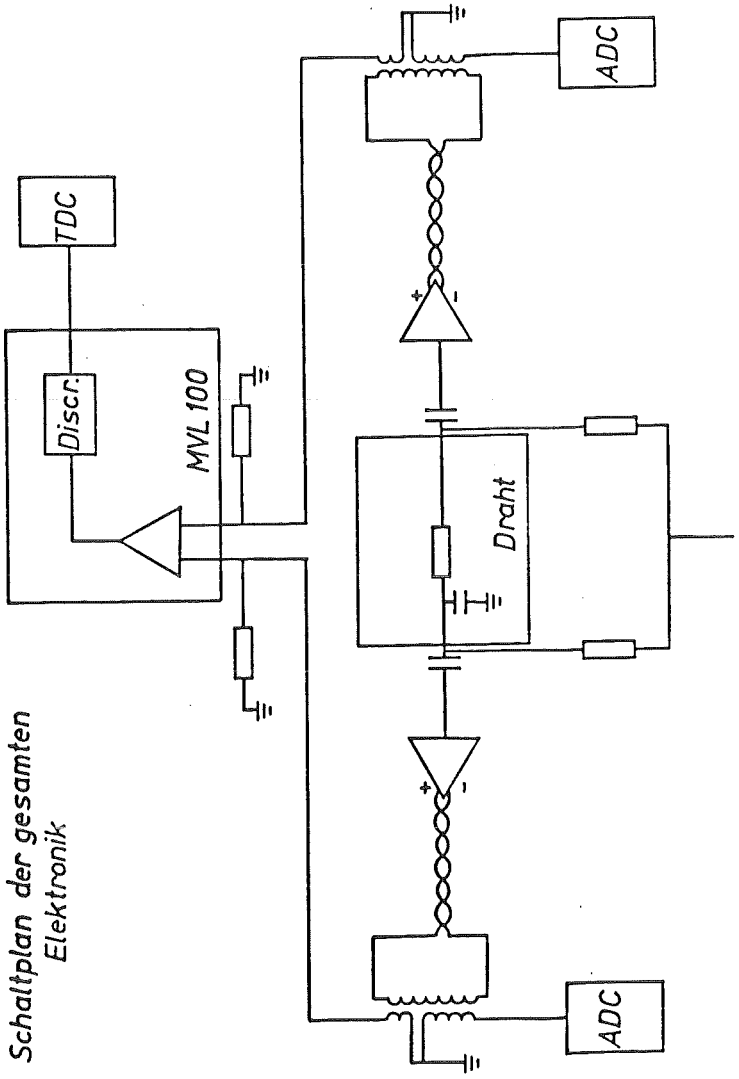


Abbildung 32: Elektronik für einen Draht

sodaß die Einzelereignisse (und nicht zweidimensionale Spektren) auf Band geschrieben werden müssen.

Wird ein Minicomputer zur Auslese und Verarbeitung der Ereignisse eingesetzt, so werden die 150 ADC Kanäle im DMA Mode gelesen, während die TDC Werte wegen der Eigenschaften des DTD einzeln erfasst werden. Wegen der relativ langen Zugriffszeiten des PDP-11 Echtzeitsystemes auf CAMAC (ca. 2-3 ms pro CAMAC Befehl) ist es mit einem solchen System nicht möglich mehr als 10 Ereignisse pro PS Puls aufzunehmen.

11.3 Kammerauslesesystem mit Mikrocomputer

Bei der Auslese mit Mikroprozessor werden die Daten während eines PS Pulses aufgenommen und im VME System zwischengespeichert. Der Mikrocomputer wird also als schneller Datenpuffer eingesetzt. Für diese Anwendung wurde ein Speicher von 48 Kbytes eingesetzt, der für 130 Ereignisse Platz bot und daher eine Verbesserung der Ausleserate um einen Faktor 10 erlaubt.

11.4 Software des Kammerauslesesystemes.

Die Programme für die Kammerauslese lassen sich folgendermaßen unterteilen:

- Ausleseprogramm im Mikrocomputer
- Dialogprogramm für die Steuerung durch den Benutzer (enthält auch die Routinen zum Aufsammeln von Online Spektren (Sampling) und für deren graphische Darstellung auf Terminals)
- Programm für Datentransfer und Schreiben auf Magnetband
- Programm zur Eichung der Elektronik

An Hand von Programmbeispiel 7 soll nur der zeitliche Ablauf dieser Programme beschrieben werden.

	MENU TASK		RUN TASK		EICHUNG
Benutzer →	Start der Menu Task				
	startet Run Task	→	Setzt Datenpuffer auf und verschafft sich Zugang dazu		
	wartet 5 s				
	verschafft sich Zugang zum Datenpuffer				
	startet Dialog mit Benutzer				
Benutzer →	startet Datennahme falls Eichung gewünscht				
	startet Eichprogramm	→			Eicht alle ADCs und TDCs schreibt Ergebnisse auf Magnetband
	wartet auf Rückmeldung				
	Startet normale Datennahme			←	Meldet Ende der Eichung

Programmbeispiel 7: Zeitlicher Ablauf der Kammerauslesepprogramme

Der gesamte Programmkomplex wird angeworfen, indem die Menu-Task (Task = Programm, das unter einem Multiprogrammsystem läuft) gestartet wird. Die Menu-Task startet zunächst die Run-Task, welche einen Datenpuffer anlegt, zudem mehrere Tasks zugreifen können (dynamischer Common Bereich) und dann darauf wartet, daß die Datennahme eingeschaltet wird. Die Menu-Task verschafft sich daraufhin Zugang zu dem Datenpuffer (mapped sich in den Datenpuffer) und stellt dem Benutzer ein Menu zur Verfügung mit dem die Datennahme mit oder ohne Eichung gestartet werden kann. Das Menu erlaubt außerdem, das Einsortieren der gelesenen Daten in Online Spektren (z.B. Verteilung der gefeuerten Drähte, Multiplizitäten, Zeitspektren und Verteilung der Ereignisse entlang der Drähte) anzuschalten und die Anzahl der Ereignisse in den Spektren festzulegen. Auch die Auswahl der darzustellenden Spektren wird über das Menu gesteuert.

Sobald der Benutzer die Datennahme einschaltet wird nachgeschaut, ob eine Eichung durchgeführt werden soll und, falls ja, gestartet.

11.5 Das Eichprogramm

Das Eichprogramm muß sowohl die Eichung der Pulshöhen als auch die Eichung der Zeitkreise vornehmen. Für die Eichung der Pulshöhen werden je 100 Eichpulse wachsender Pulshöhe in 10 Schritten erzeugt und auf den Eingang der Verstärker (siehe Abb. 32) eingekoppelt. Für die Erzeugung der Pulse wird ein programmierbarer Pulsgenerator benutzt (LRS 1976). Für jeden der 10 Eichpunkte wird der Mittelwert der am ADC gemessenen Werte errechnet und auf Band geschrieben.

Derselbe Pulser kann benutzt werden, um zwei Pulse von variablem zeitlichen Abstand zu erzeugen und damit die TDCs des DTD zu eichen. Die TDC Werte werden auf dieselbe Weise behandlet wie die ADC Werte.

Nachdem die Eichung abgeschlossen ist, meldet die Eich-Task den erfolgreichen Abschluß der Eichung an die Menu-Task und stoppt. Die Menu-Task startet daraufhin die normale Datennahme.

11.6 Die normale Datennahme

In Programmbeispiel 8 ist der zeitliche Ablauf der Kammerausleseprogramme während der normalen Datennahme gezeigt. Während des PS Pulses nimmt der Mikroprozessor die Daten der Kammer auf und schreibt sie in seinen lokalen Speicher. Sobald der Speicher voll ist, wird dies der Run-Task mit Hilfe des CLAM gemeldet. Die Run-Task verschafft sich dann den alleinigen Zugriff zum Datenpuffer und wirft den Transfervorgang der Daten zwischen Mikrocomputer und PDP-11 an. Sobald sich die Daten im Speicher der PDP-11 befinden, schreibt die Run-Task die Daten auf Magnetband.

Die Menu-Task kann während des normalen Runs zwei Zustände einnehmen:

- den Dialog Mode oder
- den "Sampling" (Sammeln und Einsortieren von Daten in Online Spektren) Mode.

In beiden Fällen werden die während eines PS Pulses vom Mikrocomputer aufgesammelten Daten transferiert und auf Magnetband geschrieben.

Im Dialog Mode steht dem Benutzer ein Menu zur Verfügung, welches ihm alle implementierten Funktionen anzeigt. Mit Hilfe dieses Menus kann er alle zur Steuerung des Experimentes notwendigen Aktionen anwählen. Er kann den Run abbrechen, einen neuen starten, die zuvor aufgenommenen Spektren anschauen, angeben wieviele Ereignisse in neu aufzunehmende Spektren einsortiert werden sollen und das Einsortieren starten (Übergang zum Sampling Mode).

Während des Samplings verschwindet das Menu und neue Steuerfunktionen können erst dann wieder durchgeführt werden, wenn die angewählte Statistik in den Online Spektren erreicht ist. Dann nämlich kehrt die Menu-Task automatisch wieder in den Dialog Mode zurück. Abb. 8 zeigt den Ablauf während des normalen Runs, wobei sich die Menu-Task im Sampling Mode befindet.

Solange das Sampling aktiv ist, schaut die Menu-Task nach, ob ein neuer Datenblock transferiert wurde, und falls ja, kopiert sie einen Teil der Daten aus dem Common Bereich in einen Speicherbereich, zu dem nur sie Zugang hat. Mit Hilfe dieser Daten werden die Online Spektren gefüllt. Sobald sich die gewünschte Anzahl Ereignisse in den Spektren befindet, geht die Menu-Task wieder in den Dialog Mode über, in dem die Ergebnisse auf Graphikterminals begutachtet werden können.

Sobald das Band zu Ende ist, wird die Datennahme gestoppt und das Band zurückgespult, sodaß das System für einen neuen Run bereit ist.

MENU-TASK	RUN-TASK
Wartet auf CLAM (Mikrocomputer Datenpuffer voll)	
	Erhält exklusiven Zugriff auf Common Bereich
	Transferiert Daten vom Mikroprozessor in Common Bereich
	Gibt Common Bereich frei
	Schreibt Daten auf Magnetband
Fordert Zugriff zu Common Bereich an	
Test ob neue Daten	
Falls ja setzt alle Zeiger zurück	
Kopiert Ereignisse in lokalen Puffer	
Gibt Zugriff zu Common Bereich frei	
Sortiert Ereignisse in Spektren ein bis gewünschte Statistik erreicht	

Programmbeispiel 8: Normale Datennahme mit Sampling

KAPITEL 12

ZUSAMMENFASSUNG

In der vorliegenden Arbeit wird ein schnelles Datenerfassungssystem vorgestellt, mit dem eine Vielzahl von Vielkanalanalysatoren emuliert werden kann. Es wurde für das LEAR Experiment PS 176 entwickelt, ist jedoch für alle Spektroskopieexperimente einsetzbar, welche normalerweise mit Vielkanalanalysatoren ausgelesen werden. Die hohe Auslesegeschwindigkeit wird dadurch erreicht, daß für die Auslese der experimentellen Daten ein dedizierter Mikroprozessor eingesetzt wird. Wegen des großen Adressbereiches des Mikrocomputers lassen sich Spektren von extrem hoher Auflösung im Speicher halten. Der von allen Ausleseaufgaben befreite Minicomputer bearbeitet Verwaltungsaufgaben wie das Speichern der Spektren auf Magnetband und die graphische Darstellung der Spektren auf Bildsichtgeräten. Dies geschieht parallel zur Auslese.

Im Gegensatz zum Vielkanalanalysator können mit diesem System auch Koinzidenzereignisse ohne Probleme verarbeitet werden.

Das System ist der Strahlstruktur von LEAR optimal angepasst. Während des LEAR Spills werden die experimentellen Daten vom Mikrocomputer aufgenommen, während gleichzeitig die Daten des vorhergehenden Spills vom Minicomputer verarbeitet werden. In der Strahlpause werden die aufgesammlten Daten des Mikrocomputers zum Minirechner übertragen. In dieser Arbeit wird eine Lösung der beiden grundsätzlichen Probleme eines solchen Systemes:

- dem Interface Experiment-Mikrocomputer
- und der Kopplung Mikrocomputer Minicomputer

gegeben. Dabei wird sowohl auf Hardware- wie auch auf Softwareaspekte eingegangen. Besonderer Wert wird auf die verschiedenen Möglichkeiten gelegt, das System auf maximale Auslesegeschwindigkeit zu trimmen.

Die erzielten Ergebnisse sind:

- 35 µs Auslesezeit pro Ereignis
- Verarbeitung von Koinzidenzereignissen ist möglich
- 2 MBytes Spektrumsspeicher erlaubt extrem gute Auflösung der Spektren (typisch 32 Zeit- und 2096 Energiekanäle)
- Der Spektrumsspeicher ist ausbaubar auf maximal 14 MBytes
- Extrem hohe Flexibilität
  Die Größe der Speicherbereiche, d.h die Auflösung der Spektren der einzelnen Detektoren, ist einfach zu ändern. Der Einbau neuer Detektoren oder das Herausnehmen von Detektoren aus dem System ist problemlos.
- Die Hardware ist auch für völlig andere Anwendungen, z.B. als schneller Pufferspeicher oder zur Vorauswertung von Einzelereignissen einsetzbar.

Das Datenerfassungssystem hat, verglichen mit kommerziell erhältlichen Vielkanalanalysatoren, einen extrem großen Speicherbereich für die Spektren (512 K Kanäle (1 K = 1024) verglichen mit maximal 64 K Kanälen eines konventionellen Vielkanalanalysators). Auch ist die Zuordung von Speicherbereichen zu den Spektren wesentlich flexibler und erlaubt daher eine optimale Anpassung an die experimentellen Anforderungen. Die Auslesegeschwindigkeit und damit die maximale Ereignisrate, die von dem System verkraftet werden kann, ist mit Vielkanalanalysatoren vergleichbar. Im Gegensatz zum Vielkanalanalysator kann das System jedoch auch Koinzidenzereignisse ohne Mühe verarbeiten. Es kann außerdem für völlig andere Anwendungen, wie Voranalyse von Online Daten oder als schneller Datenpuffer (siehe Kap. 11) eingesetzt werden.

Das System wurde während einer ersten LEAR Strahlzeit eingesetzt und hat erfolgreich erste physikalisch wertvolle Daten aufgenommen.

ANHANG A

FLIP FLOP EINHEIT

Das Ausleseprogramm im Mikrocoputer benutzt ausschließlich F2 CAMAC Befehle, um die ADCs und TDCs zu lesen. Es ist daher erwünscht, auch die Triggererkennung mit F2 Befehlen durchzuführen, da auf diese Weise das CAMPORT Funktionsregister nicht umgeladen werden muß und wertvolle Auslesezeit gespart werden kann. Es muß daher eine Triggereinheit zur Vefügung gestellt werden, welche ein Flip Flop enthält, das von der Triggerlogik gesetzt und vom Computer mit Hilfe eines F2 Befehles gelesen und zurückgesetzt werden kann. Dabei muß darauf geachtet werden, daß das FF nur dann zurückgesetzt wird, wenn der Trigger vom Mikroprozessor erkannt wurde. Da bei einem F2 Lesebefehl das Datenwort während S1 transferiert wird, das FF aber erst während S2 gelöscht wird, bleiben Trigger, welche zwischen S1 und S2 das FF erreichen unerkannt (siehe Abb. 33), falls nicht die notwendigen Vorkehrungen getroffen werden.

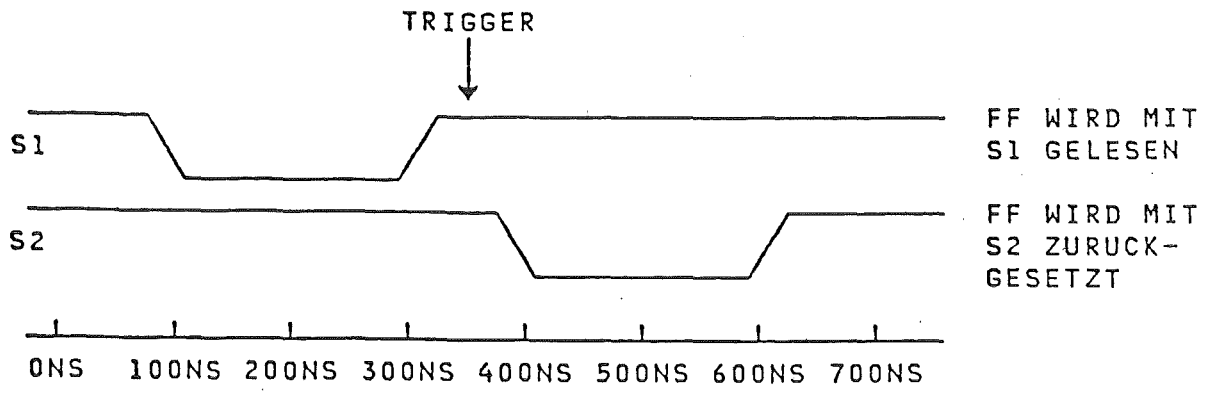


Abbildung 33: Unerkannte Trigger

Ein zweites FF wird häufig zur Unterdrückung von Triggern während der Verarbeitungszeit eines Ereignisses benötigt (Busy Schaltung).

A.1 Benutzung der Dual Flip Flop Einheit

Das Modul besitzt je einen Eingang für die FFs und einen "Common Set" Eingang, mit dem beide FFs gleichzeitig gesetzt werden können. Außerdem ist für jedes FF ein Ausgangssignal "OUT" und dessen invertiertes Signal "OUT*" über Lemo Buchsen zugänglich. Alle Ein- und Ausgänge haben NIM Pegel. Zwei LEDs (Light Emitting Diode) zeigen den Zustand der FFs an.

Die Einheit versteht folgende CAMAC Befehle:

- F0 Lesen des FF
- F2 Lesen und zurücksetzen des FF
- F16 Beschreiben des FF (nur zu Testzwecken)

wobei FF 1 über die Subadresse A=0 und FF 2 über A=1 angesprochen wird. Bit 0 im gelesenen Datenwort reflektiert den Zustand des FF (Bit=1: FF gesetzt). Beim Befehl F2 wird das FF nur dann zurückgesetzt, falls Bit 0 im gelesenen Datenwort gesetzt war. Die Schaltpläne und eine ausführlichere Beschreibung der Einheit finden sich unter [RAI83].

ANHANG B

ADC KOPPLER

Wegen ihrer sehr guten Linearität und ihrer hohen Konversionsgeschwindigkeit werden in unserem Experiment NIM ADCs vom Typ Laben 8215 bzw Silena 7411 benutzt. Um diese ADCs auslesen zu können, werden CAMAC Einheiten (ADC Koppler) benötigt, denen die konvertierten Daten der ADCs übertragen werden können und die dann vom Mikroprozessor, mit Hilfe von CAMAC Befehlen, ausgelesen werden können. Der ADC ist über ein multipol "Twisted Pair" Kabel mit dem ADC Koppler verbunden. Die Pinbelegung des Kabels ist in Tabelle 11 festgehalten.

B.1 Das Handshakeprotokoll

Um den Datentransfer zwischen ADC und Koppler in geordneter Weise durchführen zu können, wurde vom ADC Hersteller ein Handshakeprotokoll festgelegt, das in Abb. 34 dargestellt ist. Sobald der ADC seine Konversion abgeschlossen hat und die Daten im Register des ADCs vorliegen, sendet er einen DATA RDY* (Data Ready) Puls an den ADC Koppler. Ist das LAM Bit des Kopplers nicht gesetzt, so antwortet dieser mit einem TRANSFER Puls, der dem ADC anzeigt, daß der Koppler bereit ist, die Daten zu übernehmen. Der ADC öffnet daraufhin seine Datenpuffer. Der ADC Koppler übernimmt die Daten zur Zeit des Übernahmepulses und setzt das LAM Bit. Der Übernahmepuls wird 200 ns nach dem TRANSFER Puls im Koppler generiert, da zu diesem Zeitpunkt die Daten des ADCs auf den Datenleitungen stabil sind. Der Koppler schließt den Datentransfer mit einem RESET Puls ab. Der RESET Puls löscht das Datenregister des ADCs und bewirkt die Zurücknahme des DATA RDY* Signales. Um auch ADCs anschließen zu können, welche zwar dasselbe Protokoll benutzen, jedoch invertierte TRANSFER und RESET Pulse erwarten (z.B. Intertechnique CT103), kann die Polarität dieser beiden Signale mittels Jumpern (T-T* bzw. R-R*) gewählt werden.

Koppler (Cannon)	Signal	Farbe	Laben (AMP)	Silena (Cannon)
1	Masse	weiß-rot	HH(50)	36
2	D0*	rot(weiß-rot)	A(13)	1
3	D1*	braun(weiß-rot)	B(38)	2
4	D2*	orange(weiß-rot)	C(1)	3
5	D3*	gelb(weiß-rot)	D(26)	4
6	D4*	grün(weiß-rot)	E(14)	5
7	D5*	blau(weiß-rot)	F(39)	6
8	D6*	violett(weiß-rot)	H(2)	7
9	D7*	grau(weiß-rot)	J(27)	8
10	D8*	weiß(weiß-rot)	K(15)	9
11	D9*	schwarz(weiß-rot)	L(40)	10
12	D10*	rot(weiß-braun)	M(3)	11
13	D11*	braun(weiß-braun)	N(28)	12
14	D12*	orange(weiß-braun)		13
15	D13*			
16	n/c			
17	DATA RDY*	gelb(weiß-braun)	f(32)	21
18	RESET	grün(weiß-braun)	m(33)	22
19	TRANSFER	blau(weiß-braun)	k(8)	23
20	TDT*	violett(weiß-braun)	X(30)	15
21	UPPER DISC	weiß(weiß-braun)		27
22	ENABLE	grau(weiß-braun)	h(20)	20
23	Masse	weiß-braun	HH(50)	36
24	Masse	weiß-braun	HH(50)	36
25	+ 5V			

Tabelle 11: Pinbelegung der Kabelverbindung zwischen ADC und ADC Koppler

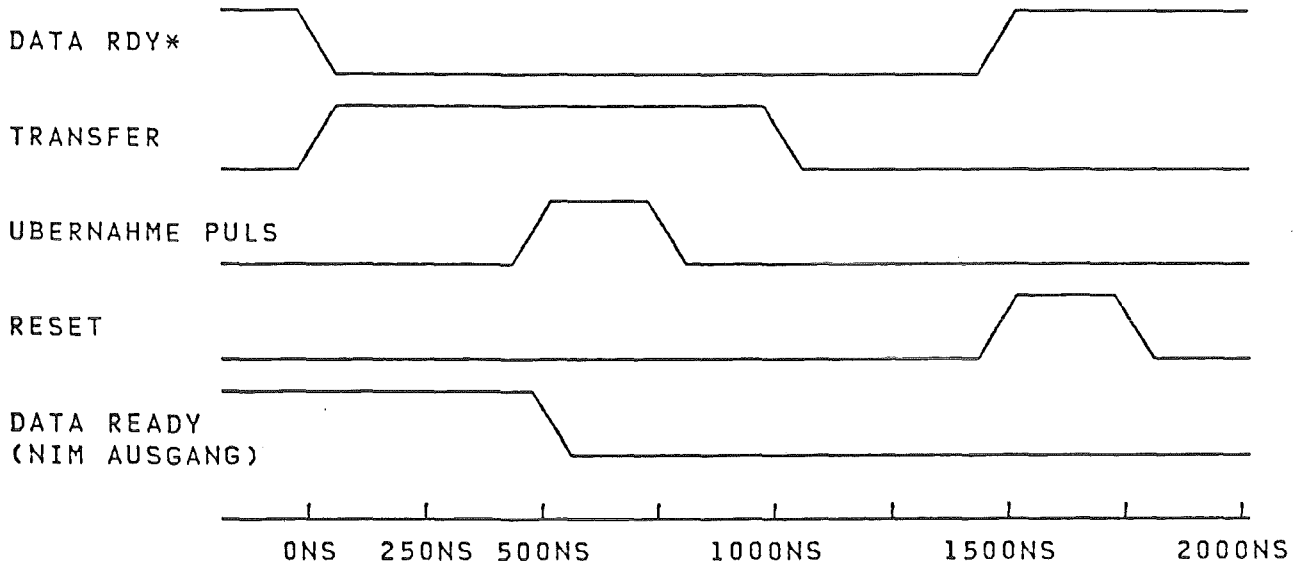


Abbildung 34: Handshakeprotokoll zwischen ADC und ADC Koppler

B.2 CAMAC Funktionen der ADC Koppler

Tabelle 12 enthält eine Aufstellung aller CAMAC Funktionen, welche vom ADC Koppler verstanden werden.

B.3 Routing

In Spektroskopieexperimenten werden die ausgelesenen Daten als Adresse bzw. Kanalnummer interpretiert, dessen Inhalt inkrementiert wird. Schreibt man die Anfangsadresse des Spektrums (die Spektrumsnummer) mit in das Datenwort, so kann dadurch die Auslesegeschwindigkeit bedeutend gesteigert werden, da die

F0	Lesen des Datenregisters
F2	Lesen des Datenregisters und Löschen des Datenregisters und des LAM Bits
F8	Testen des LAM Bits
F10	LAM zurücksetzen
F24	Sperren des ADCs
F26	Freigeben des ADCs
C	Löschen des Datenregisters und LAMs
Z	Löschen des LAMs und Sperren des ADCs

Tabelle 12: CAMAC Funktionen des ADC Kopplers

Spektrumsnummer nicht erst mit Hilfe der CAMAC Stationsadresse errechnet werden muß. Dies ist im Koppler mittels DIL Schaltern möglich.

Bei manchen Anwendungen ist es auch erwünscht, abhängig von Zusatzbedingungen (z.B. zusätzliche Detektoren oder Koinzidenzbedingungen) die Daten eines ADCs in verschiedene Spektren einzusortieren. Auch dies kann mit dem ADC Koppler mittels externen Routingbits bewerkstelligt werden.

Der ADC Koppler speist 16 CAMAC Leseleitungen (R1-R16), von denen R1-R8 grundsätzlich ADC Datenbits sind. Für die Bits R9-R16 kann mit Hilfe von 8 Jumpern (R/S-D) gewählt werden, ob es sich um ADC Bits oder um Routingbits handelt. Weitere 8 Jumpers (R-S) wählen entweder die DIL Schalter oder externe NIM Signale, die über Lemo Buchsen dem Koppler zugeführt werden, als Routingbits an. Abb. 35 zeigt die verschiedenen Routing Möglichkeiten. Werden externe Routing Signale verwendet, so kann die Zeit, in der das Datenregister des Kopplers mit Routingbits beschrieben werden kann, durch ein GATE Signal eingeschränkt werden. Die Routingbits werden in diesem Fall nur während der Dauer des GATE Signales akzeptiert. Mit einem weiteren Jumper kann angewählt werden ob mit oder ohne GATE gearbeitet wird (GATE on-off).

		Jumpers R/S-D	Jumpers R-S
1.	ADC Datenbits:	D	ohne Einfluß
2.	DIL Schalter :	R/S	S
3.	Externe Routing Signale:	R/S	S

Abbildung 35: Routing Möglichkeiten

B.4 Ausgangssignale des ADC Kopplers

Manche der internen Signale des Kopplers können auch extern für Test- oder sogar für Triggerzwecke nützlich sein. Folgende 4 Signale werden im ADC Koppler auf NIM Niveaus konvertiert und auf Lemobuchsen herausgeführt:

DATA READY	Daten liegen im Koppler Datenregister vor
RESET	Datentransfer abgeschlossen
TDT	gesamte Totzeit (Total Dead Time)
UPPER DISC	Die obere Diskriminatorschwelle hat angesprochen (nur bei Silena ADCs)

Das DATA READY Signal kann dazu benutzt werden, die Auslese zu starten, falls die ADCs die größte Konversionszeit benötigen.

ANHANG C

AUSZÜGE AUS DEM CAMPORT MIKROPROGRAMM

```
SIGNETICS BIPOLAR MICRO ASSEMBLER(A)                                    PAGE 00001
  ADDR' INSTR ES  SRCE ED DEST CCMUX  VECTOR  INC ISOC ISOM CLEAR DS AS WRITE IDTACK ONAF OMARB SPARE
                                                                                                    "
00001   ***************************************************************************
00002   *                                                                         *
00003   *                                                                         *
00004   *          C A M P O R T     M I C R O P R O G R A M                      *
00005   *                                                                         *
00006   *                                                    November 1981        *
00007   ***************************************************************************
00008
00009                          Micro Assembler Definition File
00010                          ===============================
00011                  *
00012
00013
00014   "CAMPORT MICRO-INSTRUCTION WIDTH DEFINITION"
00015   INSTRUCTION WIDTH 40;
00016
00017
00018   "CAMPORT MICRO-INSTRUCTION FIELD DEFINITIONS"
00019   FIELD INSTR     WIDTH 4 DEFAULT 0; "AMD2910 Instruction"
00020   FIELD ES        WIDTH 1 DEFAULT 0; "Enable Source"
00021   FIELD SRCE      WIDTH 5 DEFAULT 0; "Source"
00022   FIELD ED        WIDTH 1 DEFAULT 0; "Enable Destination"
00023   FIELD DEST      WIDTH 4 DEFAULT 0; "Destination"
00024   FIELD CCMUX     WIDTH 4 DEFAULT 0; "Condition Code Mux Input"
                        //
00044   "***************  AMD 2910 MICROPS  ***************"
00045
00046   MICROP JZ       ASSIGN INSTR      = 0000B;
00047   MICROP CJS      ASSIGN INSTR      = 0001B;
00048   MICROP JMAP     ASSIGN INSTR      = 0010B;
00049   MICROP CJP      ASSIGN INSTR      = 0011B;
00050   MICROP PUSH     ASSIGN INSTR      = 0100B;
                        //                //
00063   "***************  DESTINATION  MICROPS  ***************"
00064
00065   MICROP LF       ASSIGN DEST = 1  ED = 1; "Load Function"
00066   MICROP LMAR     ASSIGN DEST = 2  ED = 1; "Load MAR"
00067   MICROP LMARB    ASSIGN DEST = 3  ED = 1; "Load MAR Buffer"
00068   MICROP INCMAR   ASSIGN DEST = 4  ED = 1; "Increment MAR"
                        //               //
00106   "***************  CONDITION CODE MULTIPLEXER CODES  ***************"
00107
00108   SUM:    EQU     0;      "Test sum"
00109   ACL:    EQU     1;      "Test ACL"
00110   GI:     EQU     2;      "Test Grant in"
00111   NAFI:   EQU     3;      "Test NAF"      //
                //              //
```

C.1: Auszug aus dem Definitions File CAMPORT's

```
SIGNETICS BIPOLAR MICRO ASSEMBLER(A)                                    PAGE 00007
  ADDR: INSTR ES  SRCE ED DEST CCMUX  VECTOR  INC ISOC ISOM CLEAR DS AS WRITE IDTACK ONAF OMARB SPARE

00219  PROGRAM CAMPORT WIDTH 40 LENGTH 256;
00220  LIST SOURCE OBJECT;
00221
00222  "
00223     ***************************************************************
00224     *                                                             *
00225     *           C A M P O R T     M I C R O P R O G R A M         *
00226     *                                                             *
00227     *                                                November 1981 *
00228     ***************************************************************
00229
00230                                  Program Source File
00231                                  ===================
00234  "
00235
00236  ORG 00;  "Program Start"
00237
00238  "-------------------------------------------------------------------------------------------"
00239  "Quiescent Condition"
00240
00241  ZERO:      CJV;
  0000: 0110  0  00000 0  0000 0000  00000000 0   0    0    0     0  0  0     0      0    0     000.
00242
00243  "-------------------------------------------------------------------------------------------"
00244  "Non-Implemented Function"
00245
00246  NIF:       ∂CONT(1,1,1,0) IRQ ∂LIV(VNIMP);
  0001: 1110  1  01010 1  1001 0000  01000000 1   1    1    0     0  0  0     0      0    0     000
00247            ∂CJP(1,1,1,0) ∂TST(TRUE) SMIRQ ∂BR(EOR);
  0002: 0011  1  01111 0  0000 1111  11111101 1   1    1    0     0  0  0     0      0    0     000
00248
00249  "-------------------------------------------------------------------------------------------"
00250  "Read Status (Internal Function at N31)"
00251
00252  A1R:       ∂CJP(1,1,1,0) ∂TST(TRUE) OSTAT LDATA ∂BR(EOR);
  0003: 0011  1  10001 1  1111 1111  11111101 1   1    1    0     0  0  0     0      0    0     000
00253            ∂JZ(0,0,0,0); "Free location"
  0004: 0000  0  00000 0  0000 0000  00000000 0   0    0    0     0  0  0     0      0    0     000
00254            ∂JZ(0,0,0,0); "Free location"
  0005: 0000  0  00000 0  0000 0000  00000000 0   0    0    0     0  0  0     0      0    0     000
00255
00256  "-------------------------------------------------------------------------------------------"
00257  "Set Communications LAM (Internal Function at N31)"
00258
00259  A3R:       ∂CJP(1,1,1,0) ∂TST(TRUE) SETCLAM ∂BR(EOR);
  0006: 0011  1  10101 0  0000 1111  11111101 1   1    1    0     0  0  0     0      0    0     000
00260
00261  "-------------------------------------------------------------------------------------------"
00262  "Clear Communications LAM (Internal Function at N31)"
00263
00264  A4R:       ∂CJP(1,1,1,0) ∂TST(TRUE) CLRCLAM ∂BR(EOR);
```

C.2: Auszug aus dem CAMPORT Mikroprogramm

ANHANG D

PINBELEGUNG DER KABELVERBINDUNG CAMPORT-KDM BUS

Kabel "Alpha"		Kabel "Beta"	
Pin Nummer	Signal	Pin Nummer	Signal
1	AM01	1	D00*
3	AM02	3	D01*
5	AM03	5	D02*
7	AM04	7	D03*
9	AM05	9	D04*
11	AM06	11	D05*
13	AM07	13	D06*
15	AM08	15	D07*
17	AM09	17	D08*
19	AM10	19	D09*
21	AM11	21	D10*
23	AM12	23	D11*
25	AM13	25	D12*
27	AM14	27	D13*
29	AM15	29	D14*
31	AM16	31	D15*
33	n/c	33	LDS*
35	n/c	35	UDS*
37	IRQ2*	37	DTACK*
39	IRQ7*	39	WRITE*
41	RESET*	41	n/c
43	BUSDRIVE*	43	IACKOUT*
45	AS*	45	n/c
47	BR*	47	BGOUT*
49	BGACK*	49	n/c
Alle geraden Pins	GND	Alle geraden Pins	GND

n/c nicht verbunden

ANHANG E

PINBELEGUNG DER VME BUSKONNEKTOREN

Pin number	Reihe A	Reihe B	Reihe C
1	D00	BBSY*	D08
2	01	BCLR*	D09
3	D02	ACFAIL*	D10
4	D03	BG0IN*	D11
5	D04	BG0OUT*	D12
6	D05	BG1IN*	D13
7	D06	BG1OUT*	D14
8	D07	BG2IN*	D15
9	GND	BG2OUT*	GND
10	SYSCLK	BG3IN*	SYSFAIL*
11	GND	BG3OUT*	ERR*
12	DS1*	BR0*	SYSRESET*
13	DS0*	BR1*	LWORD*
14	WRITE*	BR2*	AM5*
15	GND	BR3*	A23
16	DTACK*	AM0	A22
17	GND	AM1	A21
18	AS*	AM2	A20
19	GND	AM3	A19
20	IACK*	GND	A18
21	IACKIN*	SERCLK	A17
22	IACKOUT*	SERDAT	A16
23	AM4	GND	A15
24	A07	IRQ7*	A14
25	A06	IRQ6*	A13
26	A05	IRQ5*	A12
27	A04	IRQ4*	A11

28	A03	IRQ3*	A10
29	A02	IRQ2*	A09
30	A01	IRQ1*	A08
31	-12V	+5V STDBY	+12V
32	+5	+5V	+5V

VERZEICHNIS DER PROGRAMMBEISPIELE

Programmbeispiel		Seite
1.	Ablauf eines CAMAC Befehles	38
2.	Mikroprogramm für die Ampelsteuerung	48
3.	Mikroprogramm der verbesserten Ampelsteuerung	51
4.	Definitionsfile für das Mikroprogramm der Ampelsteuerung	57
5.	Auszug aus dem Ampel Mikroprogramm	58
6.	Formatfile für die Ampelsteuerung	60
7.	Zeitlicher Ablauf der Kammerausleseprogramme	94
8.	Normale Datennahme mit Sampling	98

VERZEICHNIS DER TABELLEN

Tabelle		Seite
1.	Daten des LEAR Strahls	8
2.	Anforderungen an das Datenerfassungssystem	9
3.	Eigenschaften des KDM Busses	30
4.	Interne CAMPORT Funktionen ansprechbar vom MC 68000	36
5.	Interruptvektornummern	39
6.	CAMPORT Funktionen ansprechbar über den CAMAC Datenweg	41
7.	Ampelphasen	48
8.	Befehlsquellen	53
9.	Befehlssatz des AM 2910 Mikrosequenzers	55
10.	Eigenschaften des VME Busses	63
11.	Pinbelegung der Kabelverbindung zwischen ADC und ADC Koppler	104
12.	CAMAC Funktionen des ADC Kopplers	106

VERZEICHNIS DER ABBILDUNGEN

Abbildung Seite

1. $\bar{p}$-^{16}O Energie Schema 4
2. Der experimentelle Aufbau 6
3. Prinzip des 2 Prozessorsystemes 12
4. Blockschaltbild des Datenerfassungssystems 16
5. Typische Busstruktur 18
6. Daisy Chain 20
7. CAMAC Konfiguration 22
8. Adressierung einer CAMAC Einheit über den A2 CC 25
9. Funktionsblöcke CAMPORT's 28
10. CAMAC Adressierung 31
11. Adressierungsbeispiel 31
12. Blockdiagramm des CAMPORT Operationswerkes 33
13. Funktionsregister 35
14. Die Statusregister 44
15. Einfache Ampelsteuerung 49

16. Verbesserte Ampelsteuerung 50

17. Blockschaltbild des CAMPORT Controllers 54

18. Blockschaltbild des VME Adapters 65

19. Aufteilung des Speicherbereiches im VME System 66

20. CAMAC Einheiten 68

21. Flußdiagram des Ausleseprogrammes 70

22. Bedeutung der Pattern-Unit Bits 71

23. Data Aquisition Control Block 73

24. Struktur der Kommunikationssoftware 74

25. Kommunikationsbereich 77

26. Befehl der PDP-11 an den Mikrocomputer 79

27. Anforderung an den Minicomputer 81

28. Online Spektrum von $\bar{p}^{16}O$. 83

29. Korrespondenz zwischen Speicher und Bildschirm 86

30. Block Diagramm des geplanten Life Displays 88

31. Driftkammer Modul 90

32. Elektronik für einen Draht 92

33. Unerkannte Trigger 101

34. Handshakeprotokoll zwischen ADC und ADC Koppler 105

35. Routing Möglichkeiten 107

INDEX DER FACHBEGRIFFE

ADC ... siehe Analog zu Digital Wandler
Adressierung ... 17, 30
Adressraum ... 17
Analog zu Digital Wandler ... 68
Antiproton Akkumulator (AA) ... 7
Autonomous Memory Channel (AMC) ... 21, 43
Auxiliary Crate Controller ... 24

Bus ... 17, 29
Busarbiter ... 21
Busmaster ... 17
Busrequest ... 21

CAMAC ... 21
CAMAC A2 Crate Controller ... 24
CAMAC C und Z ... 23, 45
CAMAC Crate ... 21
CAMAC Funktionscode (F-Code) ... 23
CAMAC Q-Response ... 23
CAMAC Stationsadresse N ... 23, 29
CAMAC Strobesignale S1,S2 ... 23, 29, 36
CAMAC Subadresse ... 23
CAMAC X-Response ... 23
CAMPORT Funktionsregister ... 34
CAMPORT's interne Funktionen ... 36
CAMPORT's Statusregister ... 43
CERN ... 1
CLAM ... siehe Kommunikations-LAM
Condition Code Multiplexer (CC MUX) ... 50
Controller ... siehe Steuerwerk

Data Aquisition Control Block (DACB) ... 72
Definitionsfile ... 56
DIL Schalter ... 63
Direct Memory Access (DMA) ... 20

Erasable Programmable Read Only Memory (EPROM) ... 78
Extraktion, stochastische ... 7

Fast Exchange Data Register (FEDR) ... 42
Formatfile ... 59

Handshake ... 18
Hexadezimal ... 32

Interrupt ... 19, 37
Interruptservice Routine ... 19
Interruptvektornummer ... 19
Interruptvektortabelle ... 20

Kühlung, stochastische ... 7
KDM Karte ... 29
Koinzidenz ... 69
Kommunikations-LAM ... 26
Kommunikationsbereich ... 64

Low Energy Antiproton Ring (LEAR) ... 5, 7

Memory Adress Register (MAR) ... 21
Mikroassembler ... 56
Mikroinstruktion ... 56

Operationswerk ... 27

Pipelineregister ... 55
Port ... 26

Random Access Memory (RAM) ... 78

Slave ... 17
Steuerwerk ... 27

TDC ... siehe Zeit zu Digital Wandler

Vektor Mapping PROM ... 53
VME Bus ... 61

Zeit zu Digital Wandler ... 68

QUELLENVERZEICHNIS

AD180 Study of X-ray and γ-ray spectra from antiprotonic atoms
at the slowly extracted antiproton beam of LEAR
CERN PSCC 80-104, INIS-MF-6908
L. Adiels, G. Alberis, G. Backenstoss, I. Bergström, P. Blüm,
K. Fransson, D. Gotta, R. Guigas, H. Koch, M. Meyer,
P. Pavlopoulos, H. Poth, J. Repond, U. Raich, B. Richter,
M. Suffert, L. Tauscher, D. Tröster, K. Zioutas

AMD79 The AM2900 Family Data Book
Seite 2-135 - 2-159
Advanced Micro Devices Inc. 1979

DOB83 Bus Basics
R. W. Dobinson CERN Geneva
Proceedings of the 1982 CERN School of Computing
CERN 83-03 Seite 156-194 (1983)

EUR80a Euratom Bericht EUR 4100e(1980)

EUR80b Euratom Bericht EUR 4600e(1980)

EUR78 Euratom Bericht EUR6500e(1978)

HAR79 Conceptual study of a facility for low energy antiproton
experiments.
W. Hardt, L. Hoffmann, P. Lefevre, D. Möhl, G. Plass und
D. Simon CERN Genf
Proceedings of the joint CERN KfK workshop on physics with
cooled low energy antiprotons
KfK Bericht Nr. 2836, Seite 1-26

KOC76 Muonic and hadronic atoms
H. Koch
Proceedings of the summer school about nuclear and particle physics at intermediate energies, J.B. Warren (ed.)
Plenum Press New York und London 1976
Seite 87 (1976)

MEY83 M. Meyer, S.Carius, P.Schmid
Unveröffentlichter Bericht

MÖH79 Possibilities and limits with cooling in LEAR
D. Möhl CERN Genf
Proceedings of the joint CERN KfK workshop on physics with cooled low energy antiprotons
KfK Bericht Nr. 2836 Seite 27-62

MOT79a MC68000 Design Module User's Guide
Motorola Inc. 1979

MOT79b MC68000 16 Bit Microprocessor
User's Manual
Motorola Inc. 1979

MOT79c M68000 Cross Macro Assembler Reference Manual
Motorola Inc. 1979

POT82 Physics with antiprotonic atoms
H. Poth
Physics at LEAR with low-energy cooled Antiprotons
U. Gastaldi, R. Klapisch (ed.)
Plenum publishing corporation (1984)

RAI83 U. Raich, T. Köhler, J. Hauth
Unveröffentlichter Bericht

SAU77 Principles of Operation of Multiwire Proportional Chambers
F. Sauli CERN
CERN 77-09 (1977)

SIG79 Signetics Cross Mikroassembler Reference Manual
Signetics Inc. 1977

VME82 VME Bus Specification Manual
Motorola Inc. 1982