MAX-PLANCK-INSTITUT FÜR PLASMAPHYSIK

GARCHING BEI MÜNCHEN

Studie zur Erfassung und Auswertung von Experimentdaten

Joachim Steuerwald
Joachim Steuerwald jr.

IPP R/4o April 1983

Die nachstehende Arbeit wurde im Rahmen des Vertrages zwischen dem Max-Planck-Institut für Plasmaphysik und der Europäischen Atomgemeinschaft über die Zusammenarbeit auf dem Gebiete der Plasmaphysik durchgeführt.

Abstract

The development of diagnostics for application by large-scale-experiments, accompanied by progress in the field of data acquisition leads to a growing amount of data to be handled by the central processor. Thereby, 10 MBytes of data for each experiment is estimated to be typical. In conjunction, the present organisation of data-acquisition and -analysis cannot be assumed to satisfy future safety- and timing - requirements. Therefore, a detailed study of the context has been recognized to be necessary, and is presented within this paper.

Following the presentation of the systems GALE and EDDAR, currently in use at the MPI fuer Plasmaphysik in Garching, the requirements to a data-acquisition and -analysis system of the specified dimension will be defined in detail. In addition a model for communications is designed, and the context in which it should be embedded will be presented. Subsequently, essential interfaces are described in detail.

As far as needed, the representation of data will be discussed, and certain methods to reduce the execution time of transactions are presented.

Many details concerning the structure of participating virtual machines are presented and discussed. In conjunction, the monitoring and control of data flow by a central virtual machine is a main topic.

A further chapter describes the layout of terminal devices, in use for the controlling and monitoring of experiment data transmission, as well as methods to illustrate system flow control.

Final remarks, in conjunction with results of performance tests and an overview about further developments complete the discussion of elementary details.

Main emphasis is given to the discussion of the abilities of the system, because the CPU-Time for data-base transactions used by the system is very low. It is pointed out, that a large- scale computer with a speed of 6 MIPS is able to schedule two large-scale experiments with the described characteristics.

Vorwort

Der juengere der Autoren ist Student der Informatik an der TU Muenchen. Er hat die Untersuchungen fuer die vorliegende Arbeit mit Eifer und Sachverstand unterstuetzt. In seinem und dem Interesse der Leser, die in der Informatik nicht zu Hause sind, wurde der Rahmen weit genug gesteckt. Der eilige Leser kann, ohne den Faden zu verlieren, umfangreiche Eroerterungen uebergehen.

Dieser Arbeit wurden mehrjaehrige Erfahrungen bei der Uebertragung und Auswertung von Experimentdaten mit dem System EDDAR zu Grunde gelegt. In diesem Zusammenhang ist besonders Herrn Prof. Hertweck fuer Anregungen und Foerderung zu danken. Weitere Unterstuetzung erhielten wir durch viele z. T. sehr intensive Diskussionen mit den Herren Burgess, Gassmann, Hellauer, Heimann, Kroiss, Ruhs, Smeulders, Stolz und Zimmermann.

Alle Untersuchungen wurden im Rechenzentrum des Max-Planck-Instituts fuer Plasmaphysik in Garching durchgefuehrt. Es wurde ausschliesslich die installierte Hardware im normalen Betrieb verwendet. CPU- und Verweilzeiten wurden stets in CMS Maschinen auf dem Rechner SIEMENS 7880 bei betriebsueblicher Belastung gemessen.

Das System COMET wurde ausschliesslich mit der auf dieser Anlage zur Verfuegung stehenden Hard- und Software entwickelt. Basissprache ist FORTRAN 77 (VS FORTRAN), wegen der fuer grosse Datenmengen erforderlichen Optimierung wurden entscheidende Routinen in ASSEMBLER entwickelt.

Garching, den 14.3.1983

1. Uebersicht

1.1 Konzept der virtuellen Maschinen

Jedem Benutzer einer virtuellen Maschine (VM) /14/ stehen ueber virtuelle Zugriffsmethoden alle Resourcen eines physikalischen Rechnersystems zur Verfuegung. Die einzelnen VM (Fig. 1) werden ohne (virtuelles) Sharing der Resourcen unabhaengig voneinander betrieben. Mehrfachzugriff auf Files

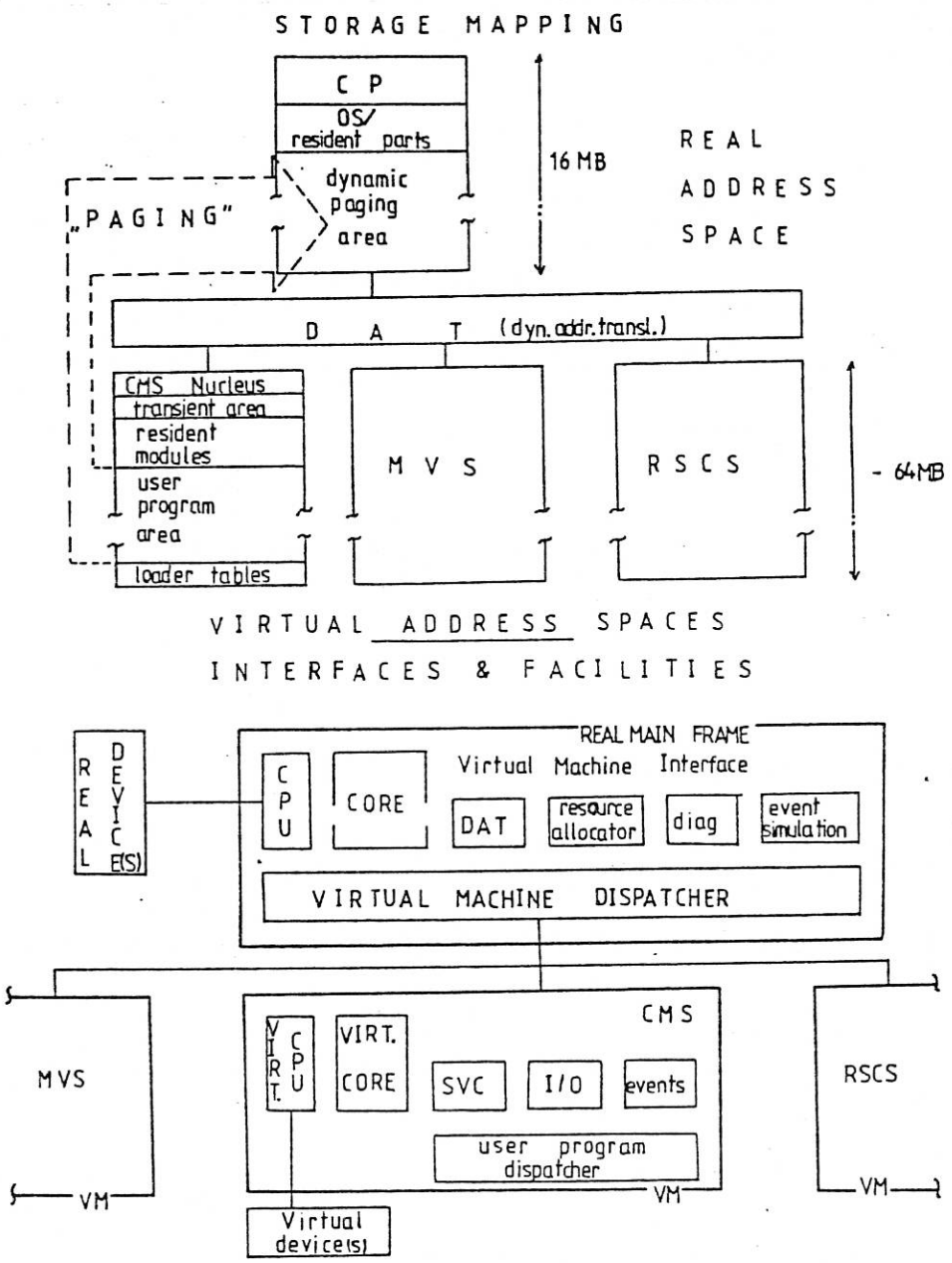


FIG 1: PRINZIP VIRTUELLER MASCHINEN

laesst sich auf der Ebene des Basis-Systems (bei : IBM Control Program /8/, abgekuerzt CP) durch Systemdienste (z. B. CP-Command LINK) realisieren, wenn den beteiligten Maschinen die virtuelle Adresse dieser Files bekannt ist. Der Datenaustausch zwischen virtuellen Maschinen durch eine Prozessor-Prozessor-Datenuebertragung, ist vor allem aus folgenden Gruenden vorzuziehen:

- es wird keine Information benoetigt, mit der die Herkunft und Art der Daten beschrieben wird,
- die Synchronisation des Zugriffs wird wesentlich vereinfacht,
- der Benutzer hat keine I/O-Aktivitaet zu initialisieren.

Zu diesem Zweck wurde im CP ein Interface zur Kommunikation (bei IBM : Virtual Machine Communication Facility /8/, abgekuerzt VMCF) entwickelt.

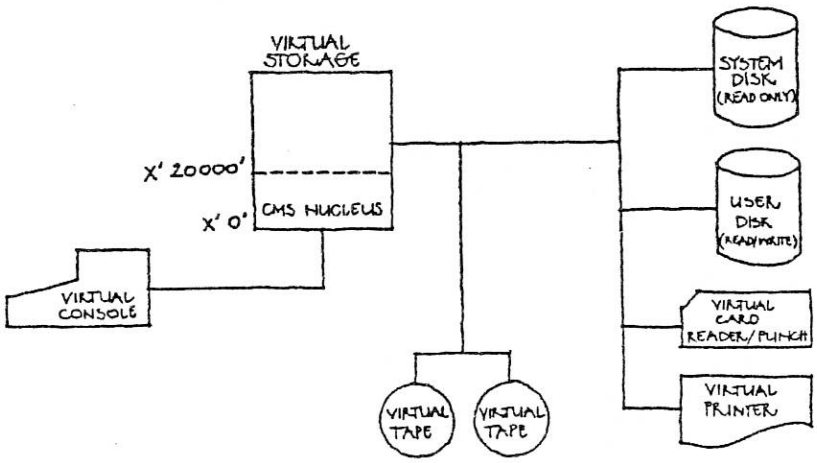


FIG 2: BEISPIEL EINER VIRTUELLEN MASCHINE (CMS)

Waehrend das CP alle physikalischen Resourcen verwaltet und die Simulation der teilweise sehr unterschiedlichen VM uebernimmt, gilt im Einzelnen fuer den Betrieb einer VM (Beispiel Fig. 2) :

- sie wird ausschliesslich durch das jeweils implementierte Operating System (OS) betrieben (z. B. MVS, AMOS, CMS, RSCS),
- jedes OS verkehrt nur ueber eine allen gemeinsame Schnittstelle mit dem CP (z. B. DIAGNOSE-Instruktion),
- die aktuelle Benutzer-Schnittstelle wird ausschliesslich durch das OS der relevanten VM verifiziert,
- Anwendungs-Programme stehen staendig unter Kontrolle

des OS der betreffenden VM,

- abgesehen von Restriktionen, die bei der Generierung der VM festgelegt werden, kann jedes Anwendungs-Programm Kontakte zum OS und CP aufnehmen.

Das Konzept der virtuellen Maschinen erlaubt daher fuer beliebige Anwendungs-Programme, unabhaengig vom uebergeordneten OS, die Erschliessung aller VMCF-Funktionen. Auf diese Weise laesst sich ohne Ruecksicht auf die Struktur einzelner Maschinen ein virtuelles Rechnernetz zur Erledigung vielfaeltig strukturierter Multi-Tasks simulieren. Damit steht ein Werkzeug zur Verfuegung, das die vollen Moeglichkeiten der Maschinen-Architektur ausschoepft.

Aus einer Studie ueber die Nutzung der mit dem Konzept der VM gegebenen Moeglichkeiten, angewendet auf die Archivierung und Auswertung der Experimentdaten beim Max-Planck-Institut fuer Plasmaphysik in Garching /1/, entstand das Programm-System COMET (Controlling and Monitoring Experiment Data Transmission).

Unter Beruecksichtigung

- der Erfahrungen aus dem Betrieb des Programm-Systems EDDAR /21/,
- neuer Technologien, die durch die Betriebs-Systeme (CP, CMS) und ihre Komponenten (z. B. XEDIT) /9/ zur Verfuegung stehen,
- der Erfahrungen aus Berichten (JET /16/, TFTR), weiterer Literatur /2,11,17/ und
- der Leistungsfaehigkeit des Betriebssystems

wurden Routinen entwickelt, die

- ein hohes Mass an Betriebssicherheit,
- gute Uebersicht ueber das Betriebsgeschehen,
- Beschleunigung der Aktionen,
- eine hohe Qualitaet des Bedienungskomforts und
- schnelle und komfortable Auswertung der Experimentdaten

garantieren. Die folgenden Abschnitte sollen im wesentlichen einen Ueberblick ueber

- die aktuelle Situation des Datenflusses,
- die wichtigsten Ergebnisse der grundlegenden Studie /1/,
- das basale Monitor-Konzept und
- vielfaeltige Moeglichkeiten der Nutzung der zur Verfuegung stehenden Hard- und Software

geben.

1.2 Aktuelles Kommunikationssystem

Das aktuelle Datenerfassungs-Konzept ist in Fig. 3 dargestellt. Die Messdaten werden beim Experiment mit externen

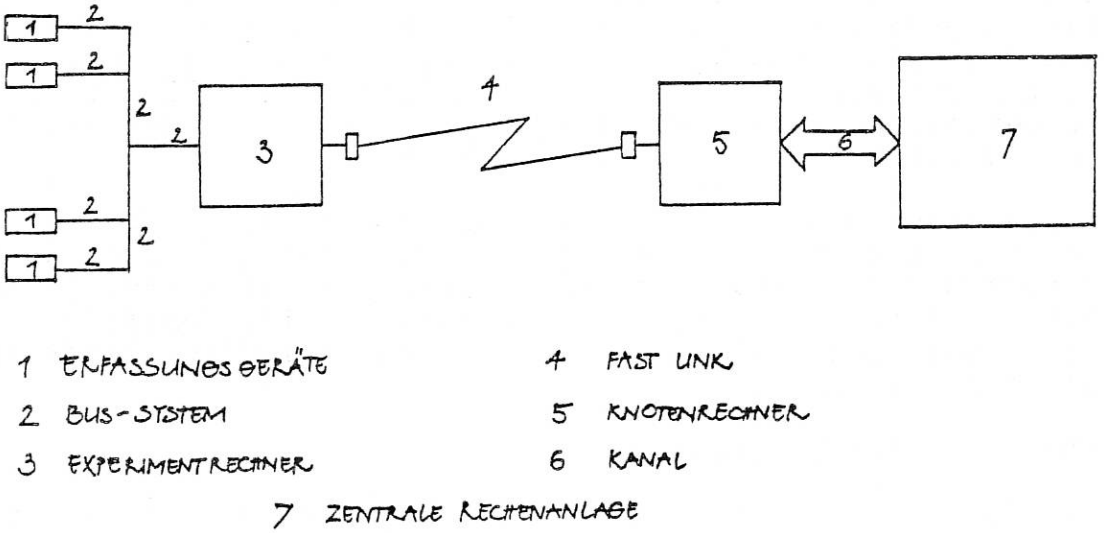


FIG 3: KOMMUNIKATION EXPERIMENT-RECHENANLAGE

Speichergeraeten (1) erfasst, ueber ein Bussystem (2) von einem Prozessrechner (3) gesammelt, zu GALE-Datenfiles /13/ zusammengestellt und in 512-Byte-Units ueber eine Datenleitung (4) gesendet. Zur Ueberwachung des Handshake wird mit zwei Bytes die Laenge des Blocks und mit zwei weiteren Bytes die Checksum mitgeliefert. Die Datenpakete werden von einem

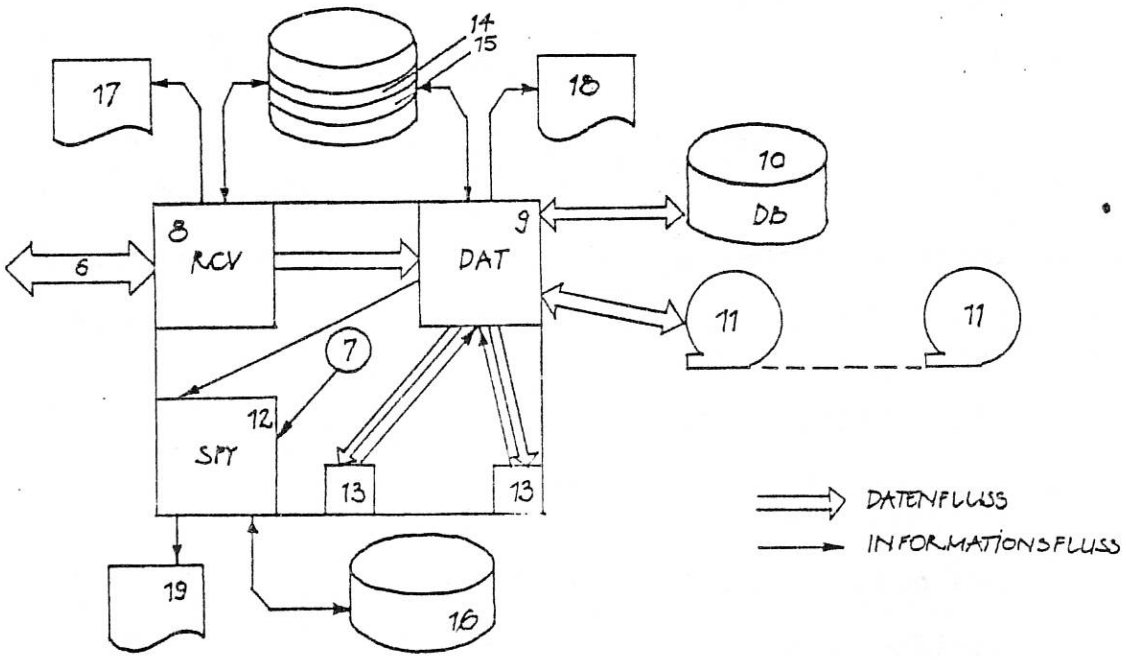


FIG 4: EINZELHEITEN DES AKTUELLEN KOMMUNIKATIONSMODELLS

Knotenrechner (5) angenommen und geprueft. Je nach Ergebnis der Pruefung wird die Uebertragung wiederholt oder das Datenpaket ueber einen Kanal (6) an den Grossrechner (7) weitergegeben.

Zum Lesen, Konvertieren, Pruefen und Archivieren der Daten, sowie zur Ueberwachung des Datenflusses und zur Auswertung sind mehrere VM (Fig. 4), die durch durch das Conversational Monitoring System (CMS) betrieben werden, eingerichtet.

Die VM <exp>RCV (8) (<exp> steht fuer die dreistellige Identifikation des Experiments am Zentralrechner, z. B. ASD fuer ASDEX, WSA fuer W VII a usw.) liest Datenpakete vom Kanal (6), konvertiert die Daten von der Zeichendarstellung des Experimentrechners zu der des Grossrechners, prueft kennzeichnende Daten und die File-Struktur und uebergibt der Datenbank, die von der VM <exp>DAT (9) verwaltet wird, ueber VMCF (Pfad 8-9) jeweils ein vollstaendiges GALE-Datenfile.

Jede weitere VM (13) des Grossrechners kann ueber VMCF (Pfad 13-9) Daten der Datenbank anfordern, die Benutzer sind nicht auf CMS-Maschinen, obwohl bevorzugt, festgelegt.

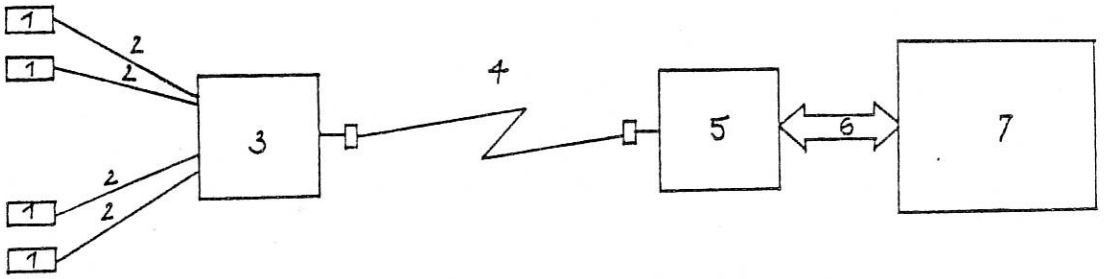
Einer dritten VM, <exp>SPY (12), wird die Ablage jedes Datenfiles in der Datenbank und die Anforderung und Uebergabe von Daten (Pfad 9-12) an den Benutzer gemeldet. Sie informiert ausserdem auf Anfrage ueber den Betriebszustand des Grossrechners.

Der Datenfluss Kanal (6) - VM <exp> RCV (8) - VM <exp> DAT (9) - VM des Benutzers (13), bei dem die Datenbank von anderen VM nur ueber VMCF angesprochen werden kann, ist nach bisherigen Erfahrungen, abgesehen von der Organisation und Ausfuehrung der Transaktionen, optimal. Globale Information fliesst ueber VMCF von der VM <exp> DAT (9) zur VM <exp> SPY (12), differenzierte Information wird im Console-Spoolfile in den VM <exp>RCV, <exp>DAT und <exp>SPY gespeichert und kann daher nicht simultan sichtbar gemacht werden.

1.3 Nachteile und Ausblick

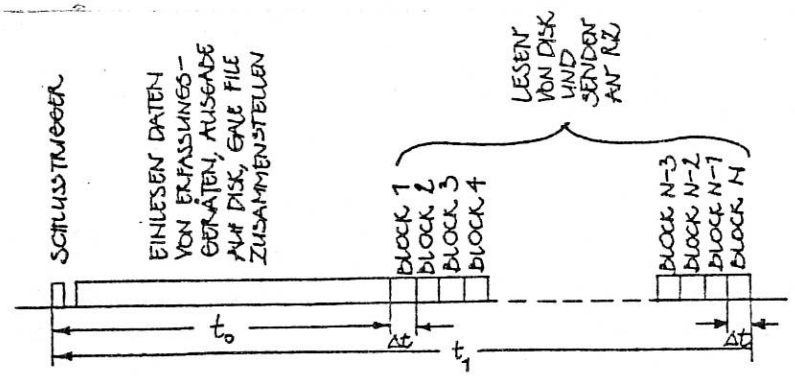
Die GALE-Filestruktur wurde wegen der Kompatibilitaet mit der Datenarchivierung beim Experiment fuer die Datenbank im RZ uebernommen. Die Wartezeit zwischen Versuchsende und Praesenz der Daten am Bildschirm des Experimentators, die in dieser Organisation mit der Datenmenge proportional waechst, darf einen Maximalwert, der sich aus Schussfrequenz und Zeitbedarf fuer eine optimale Auswertung errechnen laesst, nicht ueberschreiten. Setzt man eine Weiterentwicklung der Sofortauswertung als wuenschenswert voraus, dann wird trotz Minimalisierung der Auswerte-Programme (besonders bei I/O- und graphischen Operationen) die Datenmenge eines GALE-Files zwei MBytes nicht ueberschreiten duerfen. Die Datenerfassung

und -uebertragung muss dann in mehreren Datenstroemen



1 ERFASSUNGSGERÄTE
2 LINK ZUM EXPERIMENTRECHNER
3 MULTIPLEXER
4 FAST LINK
5 KNOTENRECHNER
6 KANAL
7 ZENTRALE RECHENANLAGE

FIG 5A : AUFTEILUNG IN DATENSTRÖME - ÜBERTRAGUNG -



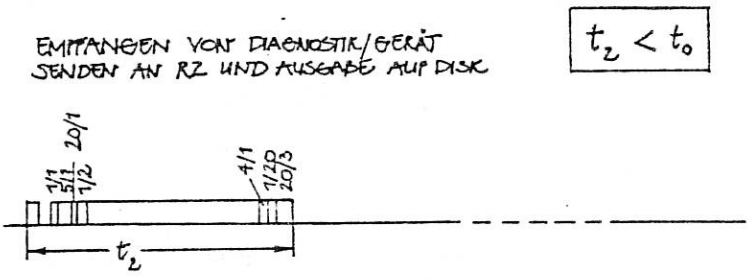


FIG : 5 B: AUFTEILUNG IN DATENSTRÖME -ZEITBEDARF-

(Fig. 5a und 5b) erfolgen, um diese Wartezeit wesentlich zu verringern. Die serielle Verarbeitung der Daten im RZ (Lesen, Konversion, Uebergabe an die Datenbank und an den Benutzer) ist mit den gleichen Nachteilen behaftet. Diese Aktionen betrachtet man unter drei Gesichtspunkten:

(1) Erweiterung der Datenbank,

(2) Umstellung des Send-Receive-Prozesses und

(3) Optimierung der Funktionen.

(1) Erweiterung der Datenbank. Die mit dem Programm-System DATMON (Autor: R. Lathe) /12/ in der VM WSADAT (fuer Experiment W VII a) eingefuehrte Methode (Fig. 6)

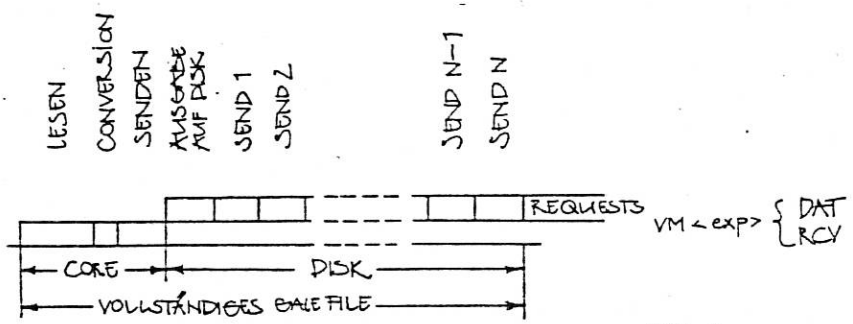


FIG. 6

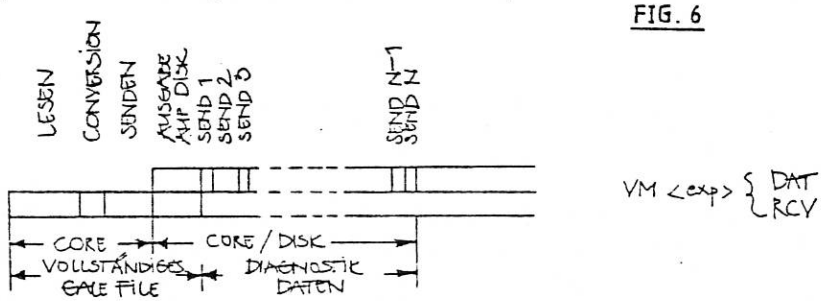


FIG. 7

FIG 6: AKTUELLE ÜBERGABE - ANNAHME VERFAHREN (WVIIa)
FIG 7: AKTUELLE ÜBERGABE - ANNAHME VERFAHREN (ASDEX)

- DATMON schreibt vollstaendiges GALE-File (bislang ca. 160 kBytes) auf die A-Disk der VM WSADAT,

- DATMON uebertraegt das vollstaendige GALE-File in den Kernspeicherbereich des Auswerte-Programms in der VM des Benutzers,

musste fuer das Experiment ASDEX neu konzipiert werden. Das Programm-System EMPFN (Autor: J. Steuerwald) /18/ in der VM ASDDAT arbeitet mit kleineren Einheiten:

- EMPFN schreibt vollstaendiges GALE-File (bislang ca. 1,6 MBytes) auf die A-Disk der VM ASDDAT,

- EMPFN uebertraegt auf Anforderung die Daten einer Diagnostik aus dem Kernspeicherbereich der VM <exp>DAT in

den des Benutzer-Prozesses.

Inzwischen werden aehnliche Ueberlegungen fuer das Experiment W VII a angestellt, nachdem die Groesse des GALE-Files auf etwa 500 KBytes ausgedehnt werden soll.

Auch dann gilt noch der einleitend dargestellte Nachteil: die Wartezeit waechst proportional zur Datenmenge.

Die Einbeziehung des Kernspeichers in die Datenbank

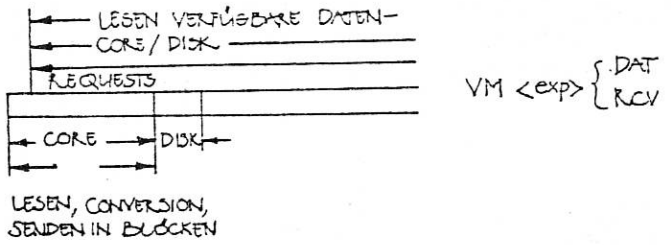


FIG 8: ÜBERGABE - ANNAHME VERFAHREN IM PROZESS COMET

bringt weitere wesentliche Zeitersparnis (Fig. 7). Der Benutzer kann so frueher bedient werden.

(2) Umstellung des Send-Receive-Prozesses. Mit der Aufteilung in mehrere Datenstroeme (Fig. 5a und 5b) ist ein weiterer bedeutender Zeitgewinn verbunden. Nach den bisherigen Vorschlaegen sollen die Daten einer Diagnostik gesammelt und (ohne Zusammenstellung eines GALE-Files) gesendet werden. Das Multiplexing mehrerer Datenstroeme erhoeht die Geschwindigkeit der Datenerfassung und -uebertragung, trotzdem muss beim jetzigen Verfahren auf die Komplettierung einer Diagnostik gewartet werden.

Die Umstellung von der Synchronisation des Prozesses durch Senden einer vollstaendigen Diagnostik auf das Lesen vorhandener Daten (Fig. 8) in der Art der Abhandlung von I/O-Prozessen erlaubt den Zugriff auf die Experimentdaten zum fruehest moeglichen Zeitpunkt.

Das Senden der Experimentdaten nach

- Lesen der Erfassungsgeraete,
- Ausgabe auf Disk beim Experiment,
- Lesen vom Disk des Experiments und
- Zusammenstellung des GALE-Datenfiles beim Experiment

wird mit der Tatsache begruendet, dass der Adressraum des

Prozessrechers am Experiment zu klein ist, um die Zusammenstellung des GALE-Datenfiles (Fig. 9 und 10) aus dem GALE Configuration File im Kernspeicher zu vollziehen.

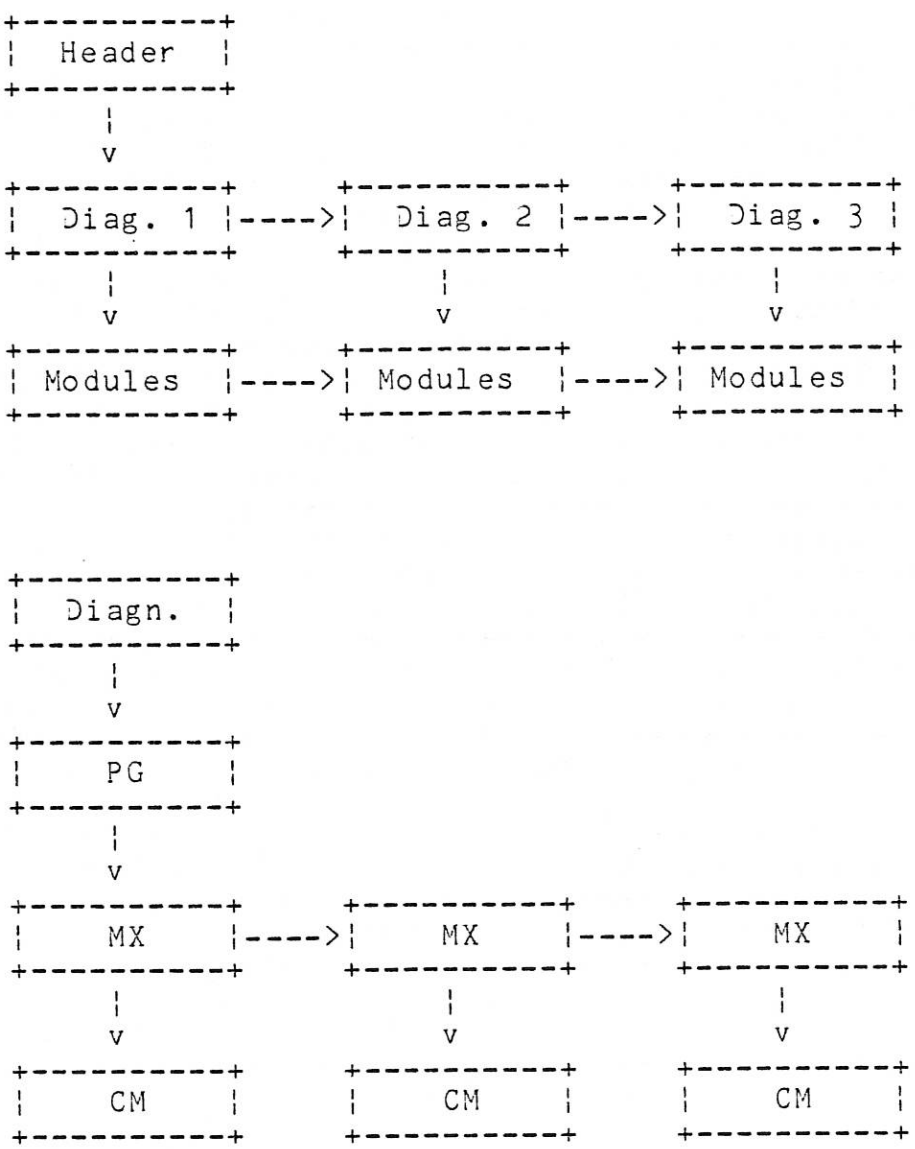


Fig. 9 GALE-File (Logik)

Das Senden kann selbstverstaendlich auch waehrend des Lesens der Erfassungsgeraete durchgefuehrt werden:

- das Senden und die gleichzeitige Ausgabe auf Disk bedeuten im Prozessrechner des Experiments keine zeitliche Verzoegerung,
- die Abbildung des GALE-Datenfiles beim Experiment auf das GALE-Datenfile im RZ muss wegen der Flexibilitaet der Fileorganisation und der Zugriffsroutinen nicht identisch, sondern bijektiv sein,
- die Zusammenstellung der Experimentdaten und des Configuration Files zum GALE-Datenfile kann im RZ wegen der Bearbeitung im Kernspeicher ohne I/O-Operationen (wegen des groesseren Adressraums) und der maschinellen Ueber-

legenheit des Grossrechners schneller erfolgen als beim Experiment.

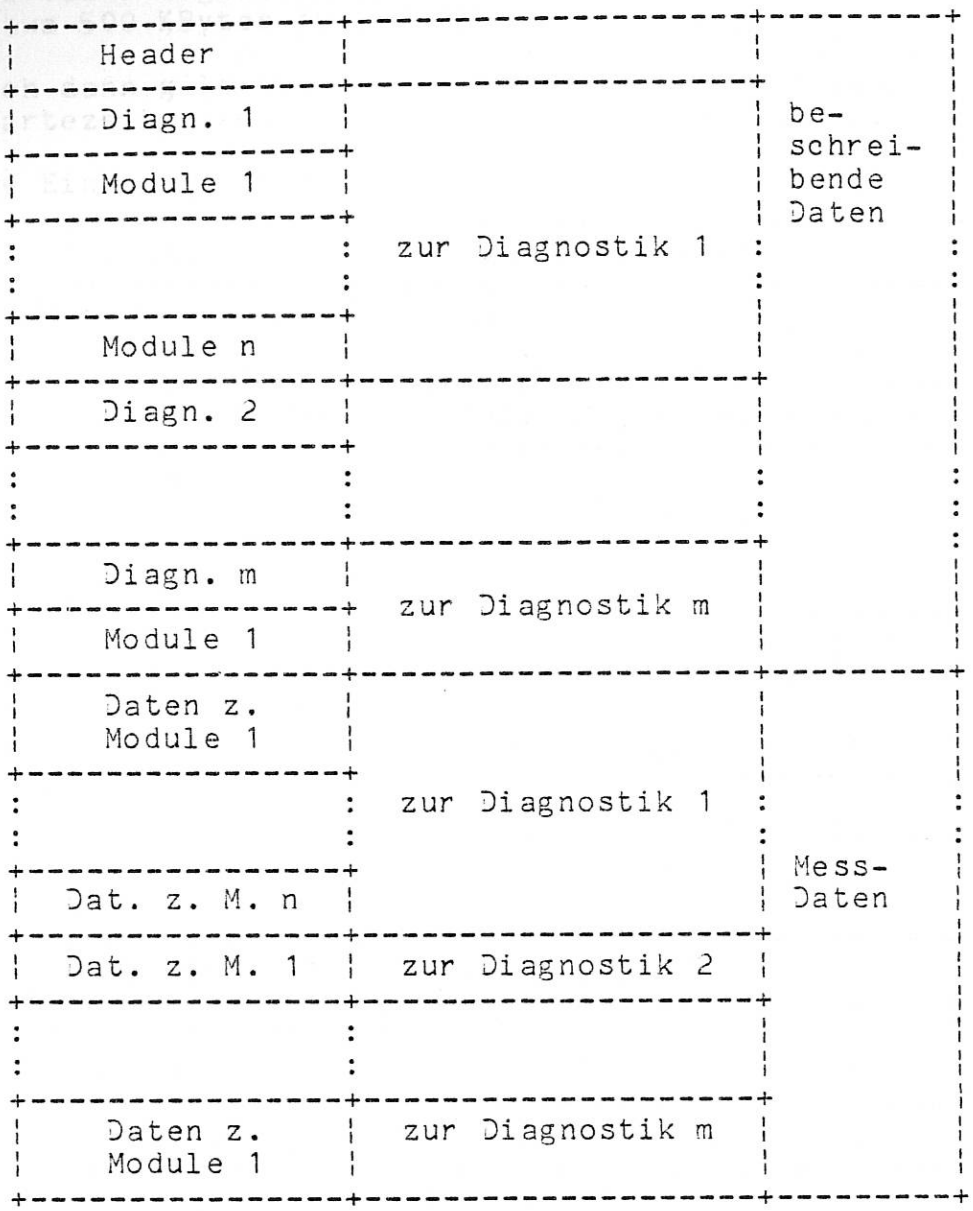


Fig. 10 GALE-File (Speicherbelegung)

Beim Experiment werden die Versuchsdaten ebenfalls archiviert. Dazu wird ein zweites GALE-Datenfile erzeugt. Diese Doppelarbeit kann wegen der fruehzeitigen Praesenz der Daten beim Benutzer u. U. vertreten werden.

Dann kann aber auch das GALE Configuration File (ca. 60 KBytes), die Grundlage des GALE-Datenfiles, das die Diagnostik- und Geraete-Daten fuer den jeweiligen Versuch enthaelt, schon mit dem Starttrigger des laufenden Versuchs gesendet werden (die diesen Versuch betreffenden Daten muessen schon vor dem Starttrigger eingegeben worden sein).

Der Vorteil des GALE-Datenfiles ist die feststehende Struktur der Beschreibung des Experiments, der Diagnostiken

und der Geraetedaten. Dieser Teil kann im RZ sofort konvertiert, an die VM <exp>DAT gesendet und an die Benutzer verteilt werden. Die ersten, darauf basierenden Programmschritte koennen dann schon vor Schussende ausgefuehrt werden.

(3) Optimierung der Funktionen. Nicht nur die aus der Entwicklung der Art der Datenuebertragung und des Programmpakets EDDAR erklaerbaren Umwege (z. B. umstaendliche Konversion der Experimentdaten) sind zu bereinigen, sondern auch neue Technologien (z. B. Interrupt-gesteuerte Prozesse, re-enterable Routinen), mit neuen Releases des Betriebs-Systems zur Verfuegung gestellte Dienste des Herstellers und die aus dem bisherigen Betrieb gewonnenen Erfahrungen sollten mit allen Mitteln genutzt werden, um die Verfuegbarkeit der Experimentdaten fuer den Benutzer zu verbessern. Dabei koennen umfangreiche Teile des Programmpakets EDDAR ohne wesentliche Aenderungen uebernommen werden. Viele Einzelheiten sind entworfen und getestet, ein hoher Aufwand an Man-Power ist dafuer nicht mehr erforderlich. Ein wesentlicher weiterer Vorteil der Weiterentwicklung des Systems EDDAR zum System COMET ist die Moeglichkeit der stufenweisen Erweiterung waehrend des Schussbetriebs.

Die ankommenden Daten werden seriell gespeichert, wegen der Erfuellung spaeterer Anforderungen werden die Datenfragmente jeder Diagnostik durch Pointer zu Listen verbunden. Bei konsequenter Durchfuehrung kann somit die Verarbeitung des ersten gesendeten Datenblocks (den Header eines Schussfiles bekommen alle Benutzer), unabhaengig von der Datenmenge, in der ersten Sekunde nach Schlusstrigger beginnen.

Moeglichkeiten zum Eingriff in das laufende System bestehen bisher wegen der autonomen Arbeitsweise der einzelnen Maschinen des VM-Tripels nicht. Bei Zwischenfaellen muss das System in den Anfangszustand versetzt und neu gestartet werden. Damit kann der Verlust wertvoller Information und Daten verbunden sein.

1.4 Anforderungen im Dauerbetrieb

Von einem System, das im Dauerbetrieb stoerungsfrei arbeiten soll, erwartet man daher

- umfassende, simultane Information mit guter Uebersicht,
- einfache, leicht erlernbare Bedienung,
- weitgehende Automatisierung,
- stoerungsfreie Koordinierung des Arbeitsablaufs.

Differenzierte Information, bisher in den Console-Spoolfiles (Fig. 4: 14, 15, 16) gespeichert, kann Entscheidungen nur durch sofortige Ausgabe unterstuetzen. Andrerseits muss die Informationsflut auf das Wesentliche beschraenkt bleiben, um die Uebersicht ueber das Geschehen zu bewahren. Trotzdem muss es moeglich sein, gezielt spezielle Auskuenfte zu erhalten, Fehlerquellen einzukreisen und Recovery einzuleiten, oder Anweisungen zur Abhilfe bei Stoerungen zu geben.

Anstelle verschiedener Console-Spoolfiles, die keine maschinelle Auswertung erlauben, sollten alle Ereignisse in einem Logfile aufgezeichnet werden um sie mit geeigneten Auswerte-Programmen zu analysieren und so

- Schwachstellen im System,
- notwendige Aenderungen in der Organisation der Daten und des Datenflusses,
- Fehleinschaetzungen beim Systementwurf,
- Fehler in der Bedienung und
- Engpaesse aus Statistiken der Benutzung

rechtzeitig zu erkennen und geeignete Massnahmen einzuleiten.

Die Beschaffung der durch Migrate ausgelagerten Daten soll unabhaengig von den bisher vorgestellten VM im Hintergrund erfolgen, um den primaeren Datenstrom nicht zu behindern, soll aber trotzdem simultan ablaufen koennen. Das Warten auf Zuteilung einer Bandeinheit und Auflegen des angeforderten Bandes stoeren den laufenden Betrieb erheblich. Daher wird das VM-Tripel durch Einfuehrung der VM <exp>MIG zu einem Quadrupel erweitert. Sie uebernimmt das Auslagern der Daten und die Wiederbeschaffung, wann auch immer angeforderte Daten off-line sind.

Bemerkung: Als vierte VM kann bei entsprechender Organisation auch die VM AMOS vorgesehen werden.

Die Betriebsmodi des Systems koennen unter zwei Aspekten betrachtet werden:

a) Kombination der VM. Als zentrale VM muss <exp>SPY stets in Betrieb sein. Ausserdem sind folgende Kombinationen moeglich :

- <exp>RCV - <exp>DAT - <exp>MIG an Schusstagen oder
- <exp>DAT - <exp>MIG an Nicht-Schusstagen.

b) Betriebsmodi der VM. Man unterscheidet fuer jede VM des Quadrupels :

- normalen Betrieb (Terminal disconnected)
- Testbetrieb (Terminal connected)
- normalen Betrieb mit Testausdrucken (Terminal disconnected).

Dabei werden im normalen Betrieb alle notwendigen Updates durchgefuehrt, dagegen im Testbetrieb nicht. Im normalen Betrieb mit Testausdrucken werden gezielt zusaetzliche Ausdrucke zum Einkreisen von Fehlerquellen ausgegeben.

Die bisherigen Erfahrungen legen die Entwicklung einer

zentralen Prozedur nahe, die nicht nur den Start jeder VM des Quadrupels unterstuetzt, sondern auch die Ueberwachung des Datenflusses und daraus resultierende Eingriffe in das System ermoeglicht.

In diesem Zusammenhang ist auch das Abfangen von Katastrophen-Situationen zu sehen. Dabei soll der betroffene Prozess nicht mit dem Module DMSABN (Abend-Routine) terminiert werden. Die Abhandlung abnormer Ereignisse, wie z. B. Program-Interrupts, wird durch das Macro SPIE der relevanten VM ueberlassen.

1.5 Erweitertes Kommunikationsmodell

Ohne Aenderungen am Kommunikationsmodell lassen sich diese Ziele nicht erreichen. Der Datenfluss Kanal - VM <exp> RCV - VM <exp> DAT - VM des Benutzers, kann, abgesehen von der Ausfuehrung der Transaktionen ohne Aenderung uebernommen werden.

Bemerkung: Die Benutzung des CMS-Filesystems ist nicht obligatorisch, seit ueber VMCF Verbindungen zu anderen VM (MVS, AMOS) bestehen.

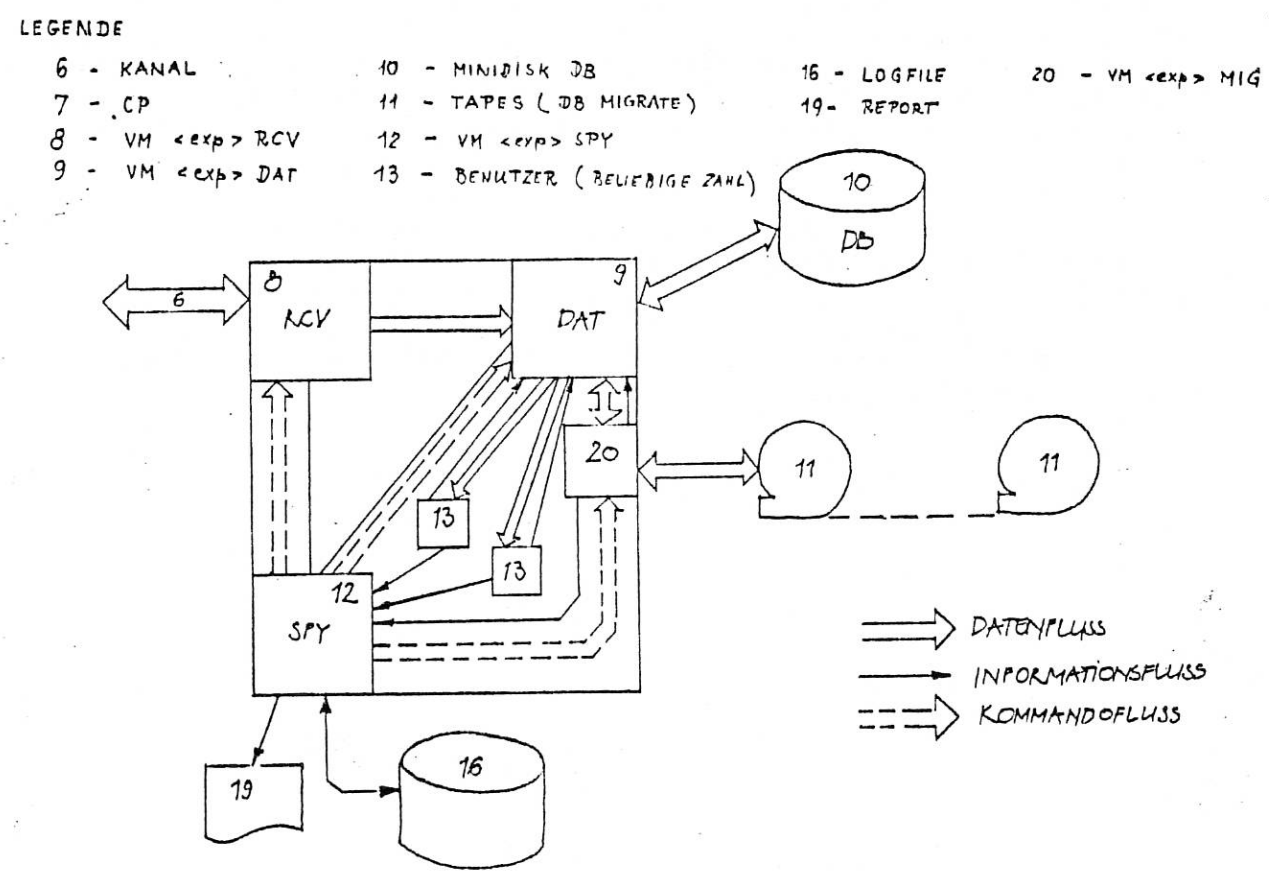


FIG 11: ERWEITERTES KOMMUNIKATIONSMODELL

Der Informationsfluss wird unter Zusammenfassung aller Informationen auf einem zentralen Bildschirm, der VM <exp> SPY, neu gestaltet (Fig. 11). Auf diese Art werden wesentliche Forderungen, wie

- simultane Information,
- zentrales Logfile und
- Nutzung der Information fuer sofortige Entscheidungen

realisiert.

Daraus folgt als zusaetzliches Hilfsmittel die zentrale Steuerung (CC). Sie erlaubt, von der VM <exp>SPY aus

- automatischen Start der Betriebs-Programme aller VM mit freier Wahl der Kombination und des Betriebsmodus,
- Umstellung des Betriebsmodus waehrend des laufenden Betriebs, Anhalten und Restart einzelner Betriebs-Programme,
- Anforderung zusaetzlicher Information ueber die angeschlossenen VM (z. B. TRACE, Testausdrucke usw.),
- Wiederholung von Teilen der Betriebs-Programme von Checkpoints oder frei gewaehlten Adressen aus oder Restart.

Entscheidungen koennen in vielen Faellen erst nach Analyse der Test-, Trace-Ausdrucke oder Dumps gefaellt werden. Bei der Komplexitaet des Systems ist nicht nur eine transparente Fehleranalyse von Bedeutung, sondern auch die Bereitstellung uebersichtlicher Checklisten und Anweisungen, die die Fehlerbereinigung unterstuetzen.

Es ist leicht einzusehen, dass mit geringfuegigen, nicht entscheidenden Aenderungen jede VM dieser Aufgabe auch durch einen Prozess in einem umfassenden Programm ersetzt werden kann. Einer Anwendung im Zusammenhang mit dem von Hertweck /3/ vorgestellten Datenerfassungs- und -auswerte-System steht daher nichts im Wege.

2. Umgebung

2.1 Informationsfluss

Die Vielfalt der Informationen, die zur Ueberwachung und Gestaltung des Datenflusses notwendig sind, erfordert eine Betrachtung im Einzelnen. Dabei darf nicht uebersehen werden, dass aus Gruenden der Effizienz der Austausch an Information auch direkt zwischen Benutzern und Datenbank erfolgt. Andrerseits kann die VM <exp>DAT aus mehreren Gruenden nicht mit den VM der Benutzer ueber das CP-Command LINK gekoppelt werden. Der damit uebliche Service ist daher ueber VMCF anzubieten. Die in diesem Abschnitt in Klammern stehenden Zahlenpaare stellen Pfade der Fig. 11 dar.

a) (<exp>RCV,<exp>DAT) - <exp>SPY (8-12,9-12)

- Betriebszustand VM (logon, disconnected, idle, busy usw.),
- Startphase eines Betriebs-Programms,
- Verweilen in Program-Units des Betriebs-Programms,
- Nachrichten ueber Fehler und Stoerungen,
- HELP- oder Error-Management zur Beseitigung von Fehlern oder Stoerungen,
- Ausdrucke von Test-Units der Betriebs-Programme,
- Nachrichten ueber den Status der Betriebs-Programme (Zustand der Daten, Speicherung der Daten, letzter Checkpoint, der ohne Fehler erreicht wurde usw.),
- Endphase eines Betriebs-Programms.

b) <exp>DAT - <exp>SPY (9-12)

- Liste der in einem vorgegebenen Zeitintervall erzeugten Datenfiles,
- freier Speicherplatz,
- im aktuellen Modus moegliche Dienste,
- Uebergabe angeforderter oder automatisch gelieferter Daten an die Benutzer,
- Meldung ueber Ausfuehrung von Diensten (z. B. Back-up, Migrate, Erzeugung von Strukturvektoren usw.),
- Warnungen oder Reaktion auf Warnungen.

c) Benutzer - <exp>SPY (13-12)

- Zustand der Benutzer-VM (logon, disconnected, autolocate, logoff usw.)

d) System - <exp>SPY (7-12)

- Zustand des Betriebs-Systems,
- Zuordnung von Geraeten, Kanaelen usw.,
- Belegung der VMCF-Queue,
- Kommunikation mit PDP11-Monitor des Experiments <exp>.

e) <exp>SPY - System (12-16,16-12,12-19)

- Logfile (als Spoolfile abgelegt, wird in festen Zeitintervallen eingelesen, auf Disk ausgegeben und gedruckt).

f) <exp>SPY - Benutzer (12-13)

- Modus der Datenbank (logon, Schussbetrieb, kein Schussbetrieb, logoff usw.).

g) <exp>MIG - <exp>SPY (20-13)

- Zuordnung Bandeinheit (attached, detached),
- Information ueber angefordertes Band (angefordert, aufgelegt, Band-, Filenummer),
- angefordertes Schussfile (Schussnummer, Laenge usw.).

Manche dieser Anforderungen implizieren, wie auch die im naechsten Abschnitt genannten Moeglichkeiten zum Eingriff in die Arbeit untergeordneter VM, eine Anpassung oder Umstellung der relevanten Betriebs-Programme.

2.2 Virtual Machine Communication Facility

Diese Einrichtung des Basis-Systems (CP) stellt alle Funktionen (Fig. 12) zur Verfuegung, die zur Einleitung, Durchfuehrung, Ueberwachung und zum Abschluss gezielten Nachrichten- oder Datentransfers notwendig sind.

Die in der TXTLIB W7LIB (Autor: R. Lathe) verfuegbaren Funktionen koennen als Subroutinen in Assembler, FORTRAN IV, PASCAL und (mit Vorsicht) auch in FORTRAN 77 aufgerufen werden.

Das Warten auf Interrupts (Subroutine VMCWFR) ist wegen grundsaetzlicher Umgestaltung des Interrupt-Handling (man vergleiche Abschnitt 2.5) durch das erheblich effizientere Unterprogramm WAIT der TXTLIB PTRLIB (Autoren J. Steuerwald und J. Steuerwald jr.) ersetzt worden. Weitere Funktionen, wie SENDX zur Uebergabe der Daten und CANCEL,

VMCF-Funktion		W7LIB	CPTEST
AUTHORIZE		VMCOPN	VMCFCO
UNAUTHORIZE		VMCCLS	VMCFDI
SEND	*	VMCSND	VMCWFW
SEND/RECEIVE	*		VMCFWA
SENDX	*		VMCFWA
RECEIVE	*	VMCRCV	VMCFWA
REPLY	*		
IDENTIFY	*		
REJECT	*		
CANCEL			
QUIESCE			
RESUME			

* Funktionen mit Protokoll-Doppelwort VMCPUSE /8/

Fig. 12 VMCF-Funktionen

RESUME und IDENTIFY, die fuer die Steuerung des Datenflusses von Bedeutung sind, mussten angefuegt werden. Alle VMCF-Funktionen ausser CANCEL, QUIESCE, RESUME, AUTHORIZE und UNAUTHORIZE, erlauben den Transfer eines Doppelwortes (IBM-Notation: VMCPUSE), das die Gestaltung eines High-Level-Protokolls ermoeglicht. Die in Fig. 12 mit * versehenen Funktionen erlauben (mit oder ohne gleichzeitige Uebergabe

Program Modules	Aufruf der Software-Funktionen des High-Level-Protocol
EXEC	CONNECT, ATTACH, LINK and ACCESS von Devices usw.
Doppelwort VMCPUSE	Mit VMCF-Funktionen gesendet, enthaelt die Parameter fuer das High-Level-Protocol
VMCF-Funktionen	Funktionen zur Abhandlung des Hardware-Protocol
DIAGNOSE X'83'	Maschinen-Instruktion mit Subcodes (der Subcode bestimmt das Kommunikations-Protokoll

Fig. 13 High-Level-Protocol zu VMCF

von Daten) den Austausch von 8 Bytes Information in beliebiger Organisation, wie Function-Codes, Messages, Error-Codes, die in vereinbarter Weise zu interpretieren sind. Die Moeglichkeiten der Interpretation sind den kommunizierenden Partnern ueberlassen und erlauben vom Senden einer einfachen

Message bis zum komplizierten Operationscode alle Zwischenstufen.

Die bisher uebliche, ausschliessliche Benutzung der VMCF-Routinen zum Transfer von Schussfiles fuehrte zu der in Fig. 14 dargestellten Organisation des Protokoll-Doppelwortes (VMCPUSE).

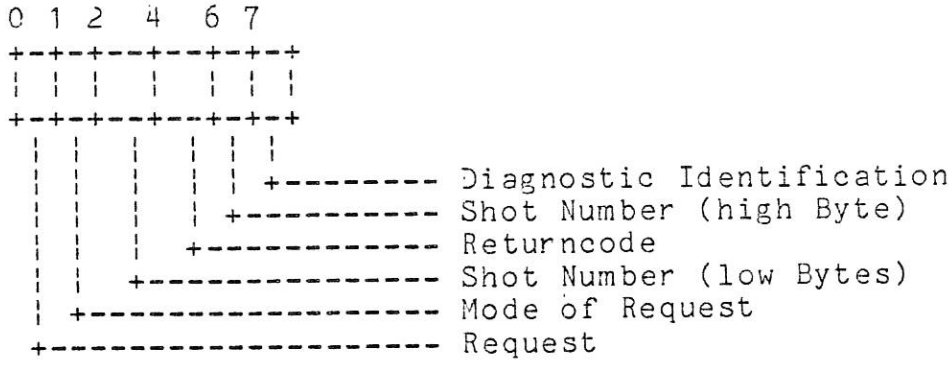


Fig. 14 Organisation Protokoll-Doppelwort (VMCPUSE)

Die Requests 0...10 (Fig. 15) sind unter Benutzung dieser Organisation

Request	Funktion
0	Locate File
1	Autolocate
2	Copyfile
3	Delete File
4	Set mode
5	File to Tape
6	File from Tape
7	Show Status
8	Finish
9	No Request
10	Illegal Request

Fig. 15 Fuer Datenbank (W VII a) belegte Requests

belegt. Eine andere Interpretation kann fuer Requests > 10, insbesondere in der Anwendung fuer zentrale Kommandos (CC), unterlegt werden (man vergleiche z. B. Fig. 16)

Die Performance war bisher unbefriedigend. Bei grossen Datenmengen oder bei haeufigem Aufruf von VMCF-Routinen empfiehlt IBM /8/ :

- Aenderung der LOADLIST, um den Module DMKVMC resident zu machen,
- die Parameterliste der VMCF-Funktion auf der gleichen

Page wie die DIAGNOSE-Instruktion zu halten,

- re-entrant Routinen fuer VMCF zu entwickeln,

- CRO Bit 31 staendig auf "true" zu setzen und die Steuerung der VMCF-Routinen durch das Bit 7 des PSW durchzufuehren,

- zur Uebergabe von Nachrichten und Daten, soweit moeglich, SENDX anzuwenden. In einigen Faellen ist dazu auch IDENTIFY geeignet.

2.3 Erweiterter Datenfluss

Die Benutzer erwarten Dienste bezueglich der Experimentdaten, die bislang nur durch Ankoppeln der Datenbank mit CP-Command LINK zur Verfuegung gestellt werden konnten. Dieses Verfahren sichert nicht zu jeder Zeit die volle Information, da die Kopie der Directory der Datenbank in der VM des Benutzers nicht staendig auf dem neuesten Stande gehalten wird. Daher sind die durch Ankoppeln einer VM verfuegbaren Operationen als Requests (Fig. 16) mit dem in Fig. 14 dargestellten Protokoll-Doppelwort einzufuehren.

Request	Funktion
21	Status von Schussfiles
22	Datum von Schussfiles
23	Schussfiles eines Zeitintervalles
24	Schussfile im Kernspeicher
31	Listfile (CMS-Command)
32	Type (CMS-Command)
41	Wiederbeschaffung ausgelagerter Schussfiles
42	Tape- und Filenummer von Schussfiles
43	in Gebrauch befindliche Tapes
44	Migrate
51	Lock Schussfiles
52	Unlock von locked Schussfiles
53	List of on-line Shots not read since <dat>
54	Number of Requests for on-line Shots

Fig. 16 Requests fuer weitere Dienstleistungen

Der Verkehr der VM <exp>DAT zur VM <exp>MIG und in der Gegenrichtung wird von der VM <exp>DAT aus gesteuert.

2.4 Eingriffsmoeglichkeiten

Die bisher mit dem Programmpaket EDDAR waehrend des Experimentbetriebs gesammelten Erfahrungen vermitteln wohl im wesentlichen das Bild eines stabilen Betriebes, koennen aber nicht darueber hinwegtaeuschen, dass durch abnormale Randbedingungen der Datenfluss an beliebiger Stelle unterbrochen werden kann. In manchen Faellen (z. B. beim Ausfall eines Datenerfassungsgeraetes) wird es genuegen, wie in Abschnitt 2.1 dargestellt, Informationen darueber zu erhalten.

Andrerseits kann es notwendig werden, Fehlerquellen einzukreisen oder zu beseitigen. Diese Aktionen koennen selbstverstaendlich, wie bisher, in der relevanten VM in der ueblichen Weise durchgefuehrt werden. Die dazu notwendigen I/O-Aktivitaeten belasten jedoch so stark, dass

- der Datenfluss erheblich gestoert wird,
- die Uebersicht ueber den Datenfluss verloren geht und
- die Fehlersuche durch eingeschraenkte Information oder Aenderung der Randbedingungen erschwert wird.

Sicherer ist die Fehlersuche in einem Betriebszustand, der die uneingeschraenkte Uebersicht ueber den vollstaendigen Daten- und Informationsfluss sichert. Dazu muss der Monitor mit allen angeschlossenen VM in voller Aktion bleiben. Dann muss aber durch Eingabe von Commands an der Konsole des Monitors, in den Programmablauf untergeordneter VM eingegriffen werden, um den Datenfluss wieder in Gang zu setzen. Um die Eingabedaten bei Wiederholung von Program-Units nicht mehrfach beschaffen zu muessen, sind nach signifikanten Operationen Checkpoints einzurichten, von denen aus die relevanten Daten mehrfach abgerufen werden koennen. Dabei kann ggf. auf die staendige Besetzung der Checkpoints verzichtet werden, man kann sie auch wahlweise oder zu Testzwecken benutzen. Zum Auffinden von Fehlern koennen, begrenzt auf bestimmte Programm-Abschnitte, Test-Ausdrucke erforderlich werden. Im Extremfalle kann Debugging erwuenscht sein, ohne die VM <exp> RCV abzukoppeln. Diese Aktionen koennen von der VM <exp> SPY mit Monitor COMET ueber VMCF eingeleitet und am zentralen Bildschirm ueberwacht werden. Die fuer diese Anwendung entworfene Organisation des Protokoll-Doppelwortes (VMCPUSE) und die zugehoerigen Operationsschluessel sind in Fig. 17 und 18 dargestellt.

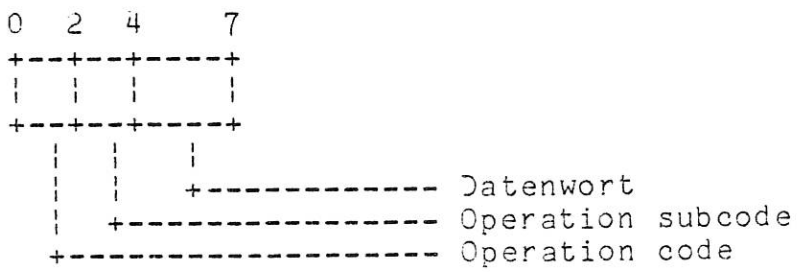


Fig. 17 Protokoll-Doppelwort (VMCPUSE)

OP - Code Nr	OP - Code Bedeutung	Op-Subcode Nr	Op-Subcode Bedeutung	Datenwort
60	Start	1	Normalbetrieb	-
		2	Testbetrieb	Fuer jede Test-Unit 1 Bit
		3	Normal- und Testbetrieb	
61	Stop	1	ohne Logoff	-
		2	Logoff	
		4	Print Fehler	1 Bit je Fehler
		8	Print Statistik	-
62	Betriebs-modus		wie Start	wie Start
63	Trace	1	General Regs *)	-
		2	Instructions	-
		4	Branch	1 Bit je Branch
		8	Adstop	Adresse
64	Dump	1	Kernspeicher *)	-
		2	File *)	-
65	Organi-zation Core			
66	Organi-zation Dataflow			
67	Go Back *)	1	Recovery-Unit	Nr. of Unit
		2	To Address	Address
69	Command *)	1	CP	
		2	CMS	

*) als zusaetzliche Information werden Daten gesendet :
63 - 1 Registernummern | 67 Startcommand
64 - 1 Speicheradressen | 69 Command-Text
- 2 File-Identifier

Fig. 18 Beispiele fuer Op-Codes des CC

2.5 Interrupt-Handling

Einige Leistungen des Monitors, wie

- zeitgebundene Funktionen (TIMER - external Interrupt),
- Kommunikation zwischen VM (VMCF - external Interrupt),
- Kommunikation mit der Konsole (CONSOLE - I/O-Interrupt),
- externe Speicherung (DISK - I/O-Interrupt),
- Kommunikation mit den Betriebs-Systemen (CMS, CP) ueber Systemschnittstelle (SVC-Interrupt),
- externe Datenuebertragung (div. Kanaele - I/O-Interrupt)

erfordern die Annahme und Bearbeitung von Interrupts. Die fuer diese Leistungen entworfenen Routinen behandeln die Interrupt-Quelle meist nur sequentiell und periodisch. Die Anwendung ist begrenzt, der Zeitbedarf meist hoeher als vertretbar. Durch

- geeignete Maskierung des PSW (gleichzeitige Annahme unterschiedlicher Interrupts),
- Verarbeitung aller Interrupts gemaess ihrer Bedeutung (nach Subcode),
- Zuruecksetzen des Systems in den vorhergehenden Zustand nach Verarbeitung des Interrupts (sonst kann eine Voraussage ueber das Systemverhalten nicht mehr gemacht werden).

sind erhebliche Verbesserungen moeglich. Fuer das Monitorkonzept ergeben sich daher folgende Forderungen:

- Wahl der Maskierung des PSW so, dass stets parallel jeder relevante Interrupt auftreten kann,
- die relevanten Interrupts muessen eindeutig und gemaess der erwarteten Ausfuehrung beantwortet werden,
- die Entries aller interrupt-verarbeitenden Routinen muessen staendig eindeutig definiert sein,
- die Anzahl der zu verarbeitenden Operationen in diesen Routinen muss minimalisiert sein (ggf. unter Aufgabe der Verallgemeinerung).

Relevant sind fuer den Monitor COMET

- I/O-Interrupts, im wesentlichen fuer jedes I/O-Geraet (wegen Verwendung der DIAGNOSE-Instruktion in diesem

Zusammenhang werden keine Disk-I/O-Interrupts zur VM reflektiert),

- External-Interrupts fuer VMCF-Kommunikation und (verschiedene) Timer,

- SVC-Interrupts fuer Operationen, die im System-Modus und ohne Unterbrechung durchgefuehrt werden muessen.

Die Unterscheidbarkeit beruht auf zusaetzlicher Information, die das System, einhergehend mit jedem Interrupt, in fest definierten Speicherbereichen ablegt. Im einzelnen lassen sich

- der Typ des Interrupts und ausserdem bei

- I/O die Kanalnummer und die Geraete-Adresse,

- EXTERNAL die Subcodes und

- SVC die SVC-Nummer

erkennen.

Der Ansatz fuer Vereinfachung und Minimalisierung liegt (aufgabenbedingt) im Bereich der I/O-Aktivitaeten und der Methoden des Systemkontakts. Geeignete Verfahren zur Straffung des Zeitbedarfs sind :

- Ignorieren der Console-Output-Interrupts,

- Verwendung des Macro STAX (MACLIB PTRLIB) das ueber SVC96 die Spezifikation einer virtuellen Adresse ermoeglicht, an die das System nach Annahme eines ATTN-Interrupts (IXTLIB PTRLIB) die Kontrolle uebergibt (so kann beim Lesen des Terminalbuffers -im Zustand VM READ- die DIAGNOSE-Instruktion verwendet werden).

- Verwendung der DIAGNOSE-Instruktion fuer I/O, erlaubt Lesen des Terminalbuffers und Ausgabe des Bildschirm-Inhaltes in einem Zuge, davon wird die Unterscheidbarkeit der Interrupt-Subclass nicht beeintraechtigt (z. B. zur Erkennung von Interrupts, die durch PF-Keys ausgeloest werden).

Die Vereinfachung der Systemkontakte wird durch

- Verwendung der DIAGNOSE-Instruction statt bisher SVC in CP und

- BRANCH statt SVC 202 in einfachen Faellen in CMS

erreicht.

Die Notwendigkeit der Anwendung der genannten Methoden wird sofort klar, wenn man beachtet, dass die Synchronisierung des Monitors mit den angeschlossenen VM ausschliesslich ueber Interrupts erfolgt. Durch die vorgestellten Methoden wurden beispielsweise die in Fig. 19 gezeigten Ergebnisse erzielt. Die Auswirkungen auf

- CPU-Anteil,
- Antwortzeit-Verhalten,
- Verhaeltnis elapsed Time/CPU-Time und
- Paging Rate

sind bei konsequenter Anwendung der in diesem Abschnitt vorgestellten Methoden beachtlich.

Operation	konventionell	minimalisiert
Console I/O Interrupts	22	1
Schreib-operationen	22	1
Maschinen-operationen	ca. 5000	20
ATTN Interrupt Maschinen-operationen	ca. 700	ca. 70

Fig. 19 Auswirkungen der Minimalisierung

2.6 Zentrale Sammelstelle

In einem System mehrerer VM kann die Ueberwachung des Arbeitsablaufs nur im Zusammenhang gesehen werden. Damit ist die Einrichtung einer zentralen Konsole, an der alle Informationen des Systems zusammengefasst und analysiert werden, gerechtfertigt.

Ueber den Ablauf signifikanter Arbeitsgaenge bei der Archivierung und Verarbeitung der Experimentdaten wird daher von den beteiligten VM an den Monitor COMET (in VM <exp>SPY) berichtet. Dieser Informationsfluss laesst Schluesse ueber den Betriebszustand der verwalteten VM und des Systems zu und gestattet bei geeigneter Gestaltung im Falle von Stoerungen fruehzeitiges Eingreifen.

Die vielgestaltige, umfassende Information des Monitors kann auch von allgemeinem Interesse sein. Dazu ist der Anschluss beliebiger Terminals ueber das DIAL-Command vorgesehen.

Beim Eingang in die jeweilige Information-Unit eines Programms wird die zugeordnete Figur des Datenfluss-Diagramms auf dem Bildschirm (s. auch Abschnitt 8.2) der zentralen VM <exp>SPY aufgehellt. Fehler die waehrend der Verarbeitung auftreten, werden sofort angezeigt. Beim Verlassen der Information-Unit wird die relevante Figur abgeblendet, Datenmenge und Verweilzeit eingetragen.

Einige Fehlermeldungen muessen mit Eingriffen in das System beantwortet werden. Sind dem Operator die notwendigen Aktionen nicht bekannt, kann er durch Betaetigen der PF12-Taste eine HELP-Message anfordern. Auf Anfrage ist dazu die Kennziffer des Fehlers (dreistelliger Code bestehend aus einem Zeichen fuer die Program-Unit und zwei Zeichen fuer die laufende Nummer des Fehlers) einzugeben. Diese Anweisungen findet man auch im COMET Operator's Guide /19/.

2.7 Schnittstellen ausserhalb RZ

Die fuenf Schnittstellen (Fig. 3)

- Erfassung der Daten in externen Speichergeraeten (1),
- Uebergabe der Daten von externen Speichergeraeten an den Experimentrechner (2-3),
- Uebergabe der Daten vom Experimentrechner an den "Fast Link" (3-4),
- Uebergabe der Daten vom "Fast Link" an den Knotenrechner (4-5),
- Uebergabe der Daten vom Knotenrechner an den Kanal (5-6)

werden nicht vom Grossrechner ueberwacht. Die Ueberwachung der Hardware durch Testprogramme, die zur Wartungszeit oder bei Fehlern zu starten sind, sollte obligatorisch sein. Ausserdem sollte beim Experiment ein Monitor, aehnlich COMET, der zumindest die ersten drei Operationen beobachtet, moeglicherweise aber auch die Verbindung zum Monitor COMET herstellt, eingefuehrt werden.

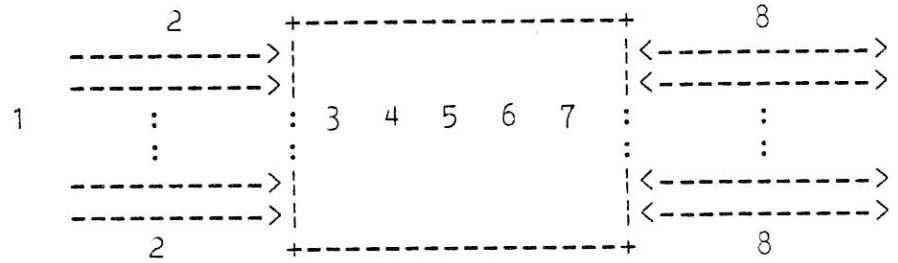
Der Knotenrechner (5) liefert ein Protokoll, das im Maschinenraum der zentralen Rechenanlage jederzeit eingesehen werden kann. Die Taetigkeit dieses Knotenrechners sollte jedoch, nach jetzigen Erkenntnissen, vom Monitor aus ueberwacht werden koennen.

3. Datenorganisation

3.1 Schnittstellen

Bei den ersten Ueberlegungen zu dieser Arbeit wurden keine bestimmenden Einfluesse der Struktur der Experimentdaten auf die Gestaltung des Systems erwartet. In den vorhergehenden Kapiteln wurde daher, sofern notwendig, auf das GALE-Datenfile /13/ Bezug genommen. Natuerlich arbeitet der Monitor COMET mit jeder beliebigen Form eines Files oder kann, wie gezeigt, in dieser Richtung angepasst werden. Sofern die Experimentdaten in Datenbanken integriert werden sollen, ist jegliche hierarchische Ordnung stoerend. Bei Suchaktionen muss der "Dienstweg" eingehalten werden. Die Struktur des GALE-Files wurde fuer Datenmengen unter 200 KBytes in linearer Organisation entworfen.

Das Ergebnis der Studie /1/ deutet, besonders fuer grosse Datenmengen, auf sinnvollere Strukturen. Um volle Flexibilitaet der Experimente bezueglich Datenerfassung und -auswertung zu sichern, sind Schnittstellen gemaess Fig. 20 zu definieren.



1 Datenerfassungsgeraete
2 Datenwege
3 Fast Link
4 Knotenrechner
5 Kanal
6 Zentralrechner
7 Datenbank
8 VM der Benutzer

Fig. 20 Schnittstellen Experiment - RZ

Durch diese Anordnung sind die Experimente in der Organisation der Diagnostiken, der Wahl der Erfassungs-Geraete und dem Aufbau von Datenwegen vollstaendig frei, die Daten koennen zu beliebiger Zeit und in freier Folge zusammen mit einem geeigneten Protokoll an den "Fast Link" uebergeben werden. Die Abnahme zu verarbeitender Daten aus der Datenbank im RZ und die Ablieferung der Ergebnisdaten soll ebenfalls in freier Organisationsform moeglich sein. Dem Benutzer wird ermoeglicht, Datenfamilien seiner Wahl in verschiedenen Stufen der Verarbeitung ohne Einbusse an Effizienz zu bilden. Diese letzte Forderung gilt bei den fuer die Zukunft zu erwartenden Datenmengen ebenfalls fuer das RZ, um CPU-, Verweil- und Response-Zeiten zu optimieren.

An der Schnittstelle beim Experiment (linke Seite der Fig. 20) koennen mehrere Datenstroeme aufgenommen werden. Die Rohdaten sollten wegen der schnelleren Verfuegbarkeit vom Experiment ohne Aufenthalt durch Organisations-Arbeiten, Vorverarbeitung usw. an den "Fast Link" uebergeben werden.

	aktuell	geplant
Serial Highway *1	200	200
PDP 11 *2	40	-
Fast Link	100	500
Knotenrechner	400	1000
Kanal	1000	1000
EDDAR *3	1000	1000

Alle Angaben in KBytes/s
*1 Netto-Leistung, Brutto-Leistung 520
*2 Einschraenkung durch Organistaionsarbeiten
*3 einschl. Organisationsarbeiten

Fig. 21 Leistungsdaten

Die Zeittabellen des Anhangs A-4 zeigen, dass auch bei normalem Betrieb des Grossrechners eher die Leistung der externen Hardware erschoepft ist als die des Rechners. Erst bei mehreren Datenstroemen, die ueber verschiedene Leitungen gesendet werden, kann Gleichgewicht eintreten. Sofern mehrere Knotenrechner beteiligt sind, ist im System COMET das Lesen von mehreren Kanaelen vorgesehen.

3.2 Datenbank

Die an die Datenbank gestellten Anforderungen werden durch eine relationale Datenbank /22/ am besten erfuellt. Dabei spielt die Organisation der Relationen zuerst nicht die entscheidende Rolle, vielmehr bestimmen die Verteilung der Daten auf den externen Speichern und die Zugriffsmethoden die Effizienz der Transaktionen.

Die Benutzer verlangen ausgewaehlte Daten unterschiedlich strukturierter Datenfamilien. Kleinste logische Einheit sind die Daten eines Geraetes, die stets in gleicher Form und Bedeutung gegeben sind. Sie wurde schon fuer die Bereitstellung der Daten umfangreicher Diagnostiken beim Experiment ASDEX geplant. Bei der Auswertung koennen sie bei Datenmengen unter 256 KBytes ohne Umstaende zur Diagnostik zusammengefasst werden, selbst wenn die Verarbeitung unter unterschiedlichen Aspekten erfolgt.

Eine Loesung vom GALE-Datenfile -oder vom Begriff des Datenfiles ueberhaupt- als Menge unterschiedlicher (innerer) Strukturen, ist bezueglich der Speicherung in einer Datenbank unumgaenglich. Die Daten jeder logischen Einheit werden im wesentlichen mit den gleichen -oder mit aehnlichen- Operationen aufbereitet. Selbstverstaendlich sind GALE-Datenfiles oder die Zusammenfassung von Diagnostikdaten ebenfalls Datenfamilien, das Ueberleben als virtuelles File ist gesichert.

Bemerkung: Die GALE-Datenfiles des Experiments W VII a (bisher ca. 160 KBytes) wurden von jedem Benutzer ohne gegenseitige Behinderung zur Verarbeitung in Auswerteprogrammen vollstaendig eingelesen . Nach Vergroesserung des Files auf ca. 200 KBytes ergeben sich Schwierigkeiten. Die hierarchische Struktur muss daher aufgebrochen werden.

Fuer die Beschreibung der Datenstrukturen (Domain Deskriptoren) wurde das GALE-Namelistfile /13/ in Listenstruktur entworfen. Schnelle Algorithmen zum Durchsuchen -auch langer Listen- sind allgemein bekannt. Aenderungen im Entwurf sind ausschliesslich

- in der Organisation der Namelist Control Blocks (NCB) (z. B. mehr Platz fuer Text) und

- in den Methoden des Update (s. Abschnitt 8.3)

notwendig. Schliesslich sind die NCB nicht nur zum Update von Geraete- und Einstelldaten (und der Steuerung) zu benutzen, sondern auch zur Adressierung von Variablen in Auswerte-Prorammen. Die dazu notwendigen Systemfunktionen sind zu entwerfen und bereitzustellen /20/.

3.3 Organisation der Datenbank

Die Effizienz der Transaktionen in einer Datenbank haengt von

- den Methoden der Anforderung der Daten,

- den Zugriffsmethoden,

- der Organisation der Listenanfaenge und dem Zusammenhang mit dem GALE Configuration File und

- der Verteilung der Daten auf die externen Speicher ab.

(1) Methoden der Anforderung von Daten. Experimentdaten werden aus drei verschiedenen Situationen heraus angefordert:

- zwischen zwei Versuchen,

- nach Versuchsserien und

- spontan.

Zwischen zwei Versuchen werden die Experimentdaten aus dem Kernspeicher der Datenbank ueber VMCF an die Benutzer ueber-

tragen Der Zugriff auf die unterschiedlichsten Daten ist problemlos. Das Auswerteprogramm des Benutzers muss auf das Eintreffen der relevanten Daten warten. Eine Abkuerzung dieser Wartezeit kann durch

- Senden der Verwaltungdaten mit dem Starttrigger und
- Legen von Datenpfaden

abgekuerzt werden. Der erste Punkt bedarf keiner Erlaeuterung. Sofern die Auswerte-Programme bezueglich der Anforderung von Daten nicht staendig geaendert werden, werden die Daten in vorhersehbarer Reihenfolge und Menge angefordert. Andrerseits kann unter der Voraussetzung mehrerer Datenstroeme nicht angenommen werden, dass die Daten in gleichbleibender Reihenfolge verfuegungsbereit sind.

Die daraus resultierende Wartezeit kann durch Einfuehrung des abstrakten Objektes "Datenpfad" optimiert werden.

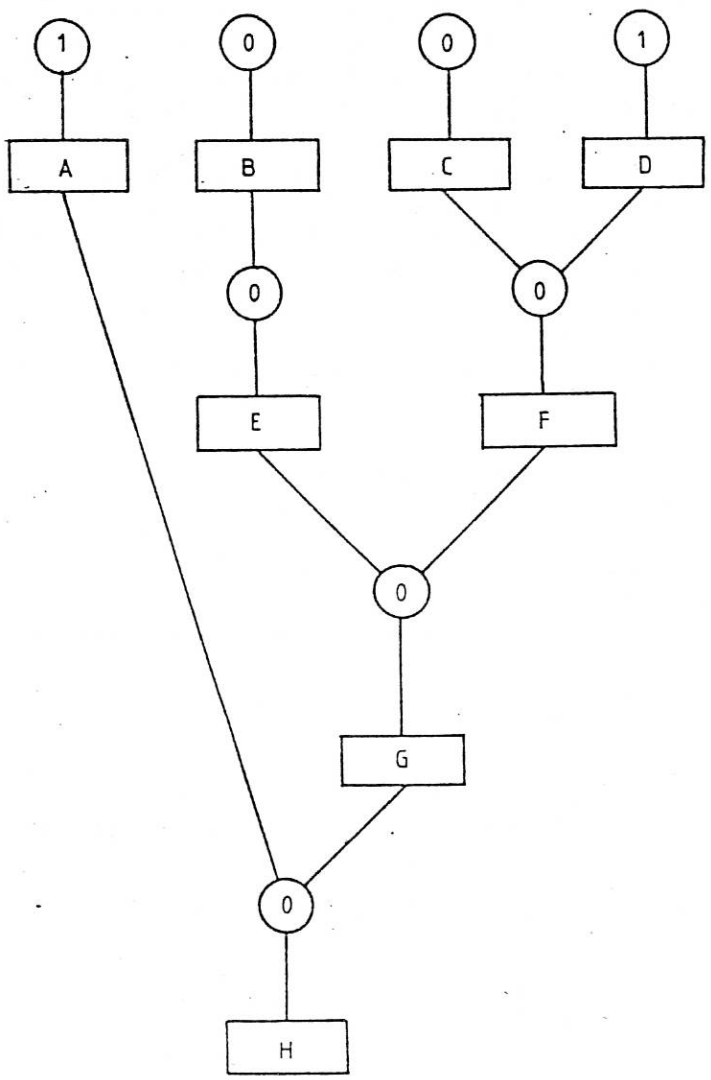


FIG 22: DATENPFAD

Definition: Als Datenpfad (Fig. 22) bezeichnet man das Petri-Netz eines Programmes, in dem die Programm-Bloecke t-Knoten und die damit assoziierten Daten s-Knoten sind.

Definition: Ein endliches Petri-Netz ist ein Relativ (S,T;Q,R) mit folgenden Bedeutungen:

(1) S und T sind endliche disjunkte Mengen. Die Elemente von S heissen s-Knoten (oder Stellen), die Elemente von T t-Knoten (oder Transitionen).

(2) Q ist eine zweistellige Relation ueber S x T.

(3) R ist eine zweistellige Relation ueber T x S.

(4) Es ist eine Markierung m : S -> (0,1) gegeben.

s-Knoten werden durch einen Kreis, t-Knoten durch Rechtecke oder Quadrate dargestellt.

Zur Aufstellung des Datenpfades wird ein Programm in Programm-Bloecke gemaess der Datenabhaengigkeit zerlegt. Jedem Programm-Block ordnet man einen t-Knoten, jeder Menge von Eingabedaten zu einem Programm-Block einen s-Knoten (der einen Semaphor simuliert) zu. Die Ausgabedaten eines vorausgehenden Programm-Blocks koennen ganz oder teilweise Eingabedaten eines folgenden sein.

Bemerkung: Ein mit 1 besetzter s-Knoten am Anfang des Petri-Netzes erwartet keine Daten. Jeder mit 0 besetzte Knoten wird bei Eingang der Daten auf 1 geschaltet. Ist ein s-Knoten mit 1 besetzt, kann der relevante Programm-Block ausgefuehrt werden.

Ein Ablauf-Diagramm des Beispiels Fig. 22 ist in Fig. 23 dargestellt:

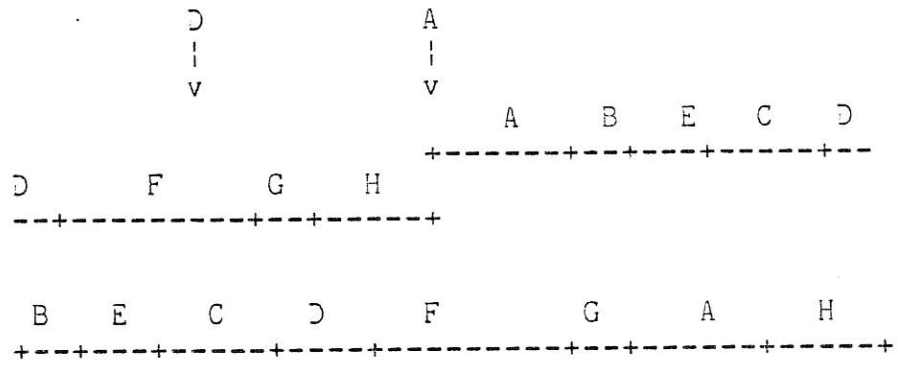


Fig.23 Zeitverlauf mit und ohne Datenpfad

Ohne Datenpfad. Ohne Ruecksicht auf das Eintreffen der Daten werden die Programm-Bloecke in der Reihenfolge

A-B-E-C-D-F-G-H

durchgefuehrt.

Mit Datenpfad. Es wird mit B-E-C begonnen. Treffen zuerst die Daten zu D ein, dann folgt D-F und danach ggf. Warten auf A, schliesslich A-H. Treffen die Daten zu A zuerst ein, dann wird A verarbeitet, warten auf D und endlich D-F-G-H.

Auf der naechst hoeheren Ebene kann man auf diese Art Datenpfade des gesamten Experiments waehrend des Schussbetriebes oder der Serien-Auswertung aufstellen und ggf. mit VMCF Multiple Send weiter erhebliche Zeit gewinnen.

Bei der Auswertung von Versuchsserien liegen die Verhaeltnisse teilweise anders:

- die Auswertung ist nicht strikt zeitgebunden,
- die Daten muessen teilweise von externen Speichern (Platte, Massenspeicher, Band) beschafft werden,
- daraus folgt eine grosse Zahl von I/O-Operationen.

In diese Ueberlegungen ist einzubeziehen, dass die Schusstage der Experimente alternieren. An Schusstagen des einen Experiments werten die Diagnostiker des anderen Versuchsserien aus. Die dabei erforderliche grosse Anzahl I/O-Operationen beeinflusst, auch wegen der Menge der Daten den Rechenbetrieb -und damit auch das aktive Experiment- erheblich. Eine Minimalisierung der I/O-Operationen wird sich deutlich bemerkbar machen. Es laesst sich zeigen, dass auch in diesem Zusammenhang der Datenpfad zur Optimierung durch rechtzeitige Bereitstellung der Daten beitraegt.

Art der I/O-Routinen	Lesen		Schreiben	
	CPU msec	Ela sec	CPU msec	Ela sec
FORTRAN 1)	1436,753	27,504	2177,524	58,651
System-Macros 2)	17,000	3,138	32,000	3,327
VMCF 3)	0,057	0,056	0,980	0,065

1.) FORTRAN mit binaeren I/O-Operationen aus VS FORTRAN
2.) mit einmaligem Aufruf einer Assembler Routine fuer Macro FSREAD oder FSWRITE
3.) mit DIAGNOSE-Instruktion
Ela - Verweilzeit
I/O-Operationen auf 3350 Disk
Zeiten fuer Uebertragung von 1 MByte Daten

Fig. 24 Zeiten fuer I/O-Operationen

Die spontane Anforderung von Daten kann wegen mangelnder Vorausschaubarkeit auftretender Strukturen nicht geplant

werden.

(2) Zugriffsmethoden. Zugriffe auf Daten mit I/O-Routinen hoeherer Sprachen sind zeitraubend. In CMS wurden fuer EDDAR die vom Hersteller zur Verfuegung gestellten System-Macros /7/ verwendet. Die CPU- und Verweilzeiten sind in Fig. 24 gegenueber gestellt. Man schliesst daraus:

- die Zugriffsroutinen (GETCH, GTSHOT beim Experiment ASDEX, GETDAT usw. beim Experiment W VII a) koennen erheblich beschleunigt werden,

- die System-Macros sind, wo immer moeglich, einzusetzen und

- die Bereitstellung der Daten muss sorgfaeltig geplant werden.

Als Sonderfall ist die Uebertragung der Daten des aktuellen Schusses zu sehen. Sie erfolgt aus dem Kernspeicher der Datenbank in erheblich kuerzeren Zeiten (Fig. 24). Die Uebertragung selbst wird stets durch VMCF ausgefuehrt. Damit erlangen die in Abschnitt 2.2 vorgestellten VMCF-Funktionen und ihre Optimierung grosse Bedeutung.

(3) Verteilung der Daten. Man hat nicht nur zu entscheiden, auf welchen externen Speichergeraeten (Platte, Massenspeicher, Band) die Daten abgelegt werden, sondern auch die Verteilung innerhalb der einzelnen Geraete zu organisieren, um Anzahl und Art der Zugriffe und der mechanischen Bewegungen zu optimieren.

Auch in diesem Zusammenhang spielen die Datenpfade eine wichtige Rolle. Durch geeignete Statistiken wird Konstanz oder Fluktuation des Benutzerverhaltens deutlich. Daraus koennen Schluesse fuer die Optimierung der Speicherung gezogen werden.

Zu den hier diskutierten Punkten liegen, besonders bezueglich der erwarteten grossen Datenmengen, keine Erfahrungen vor. Entscheidungen koennen erst nach eingehenden Untersuchungen, die hiermit vorgeschlagen werden, getroffen werden.

Bemerkung: Eine DB fuer ausgewertete Daten ist von gleich grosser Bedeutung, kann jedoch in diesem Rahmen aus zeitlichen Gruenden nicht diskutiert werden. Ohne Mitarbeit der beteiligten Experimentalphysiker ist an die Entwicklung eines DB-Systems dieser Anwendungsbreite nicht zu denken. Es empfiehlt sich, die Vorbereitungen bald zu beginnen.

4. Struktur der VM <exp>RCV

4.1 Prozess PETER

Im aktuellen System EDDAR werden Interrupts in den Unterprogrammen GETBLK (Autor: H. Hackl, I/O-Interrupt vom Kanal) und VMCWFR (Autor: R. Lathe, EXTERNAL Interrupt von VMCF) sequentiell abgehandelt. Bisher wurde noch keine Konfliktsituation beobachtet. Sie wurden durch gewisse Vorsichtsmassnahmen beim Programmentwurf weitgehend vermieden. Einige Stoerungen, fuer die es bislang keine Erklaerung gab, koennten jedoch darauf zurueckzufuehren sein, dass nur sequentielles determiniertes Auftreten von Interrupts erlaubt ist.

Zur Steuerung des Datenflusses von einer zentralen VM aus, sind weitere Interrupts noetig. Ein solches System kann nur dann stoerungsfrei arbeiten, wenn der Ablauf der Programme durch diese Interrupts gesteuert wird. Das Flussdiagramm in Fig. 25 zeigt das Resultat dieser Ueberlegungen

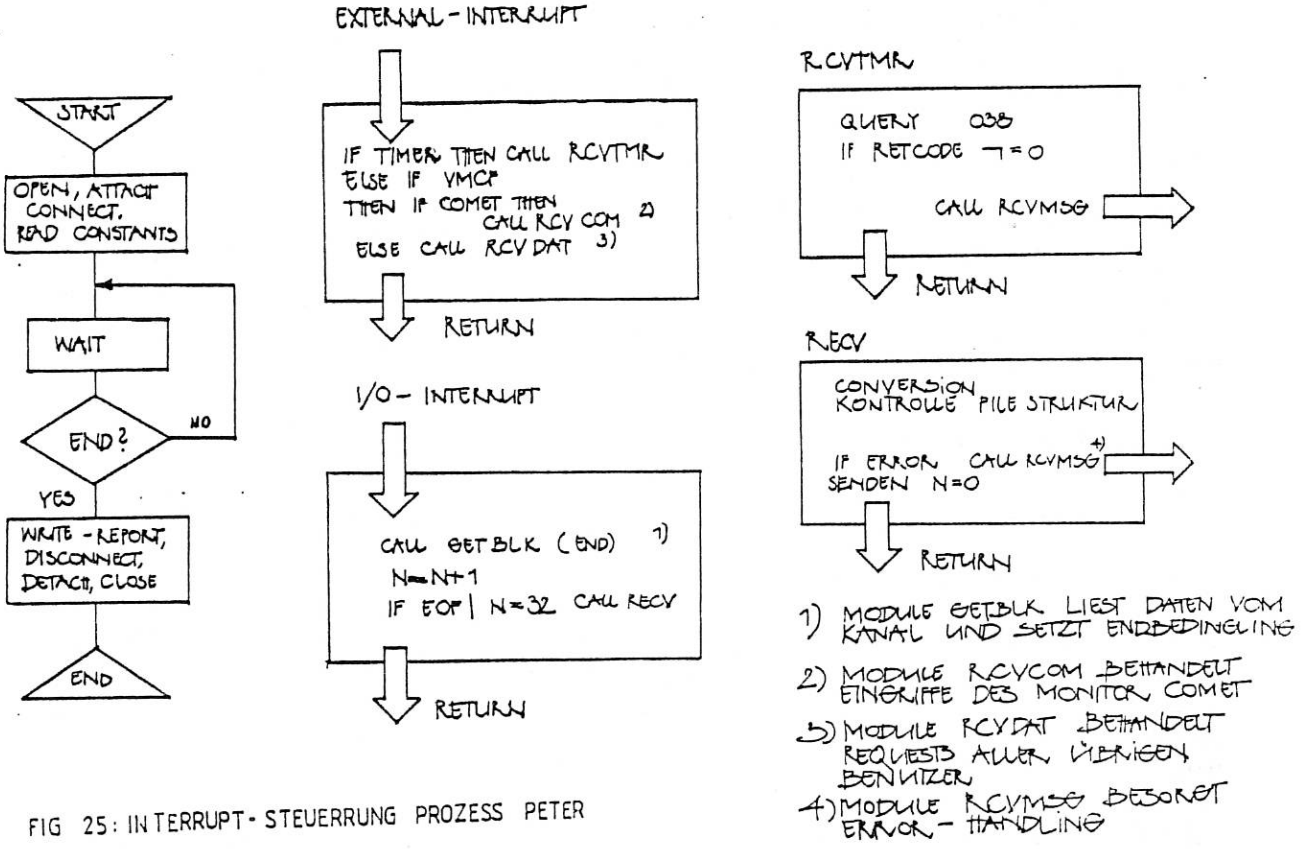


FIG 25: INTERRUPT - STEUERRUNG PROZESS PETER

fuer den Prozess PETER (die Arbeitsweise der Moduln wird eingehend im COMET System Programmer's Guide /20/ dargestellt).

a) Start. Fuehrt alle OPEN, ATTACH, ACCESS, CONNECT und

AUTHORIZE Anweisungen aus, prueft Status und Lesbarkeit der Systemdaten und Directories und liest die permanenten Daten ein.

b) WAIT und Endabfrage. Der Prozess PETER verweilt fast staendig, nur unterbrochen durch Interrupts und das darauffolgende Interrupt-Handling, in dieser Program-Unit. Nach der Verarbeitung jedes Interrupts wird die Endabfrage, fuer die der Condition Code entweder durch Eingabe vereinbarter Daten oder ueber VMCF durch die zentrale VM gesetzt wird, angesteuert. Ein in regelmaessigen Abstaenden erzeugter TIMER-Interrupt fragt die Lesebereitschaft der Subroutine GETBLK ueber das CP-Command QUERY 038 (Kanal 0, Device 38 : virtuelle Adresse des Datenlieferanten) ab.

c) Exit. Diese Program-Unit gibt veraenderte Files aus, schliesst alle Files, beendet das Programm (RELEASE, DETACH, DISCONNECT, CLOSE und UNAUTHORIZE) und veranlasst die VM <exp>SPY, das Schussprotokoll auszugeben.

4.2 Informationsfluss

Die Organisation des Datenflusses (Fig. 8 und 11) erlaubt Informationen ueber Zwischenstadien (Lesen, Konversion, Senden) nicht mehr. Man unterscheidet, wie in Abschnitt 4.1 schon gezeigt, statt dessen:

(1) Anfangszustand,

(2) Programmstart und Vorbereitungen,

(3) Wait,

(4) Verarbeitung der Daten, Send und

(5) Abschluss und Exit.

(1) Anfangszustand. Die VM <exp> RCV kann drei Zustaende annehmen:

- nicht bereit,
- warten und
- Prozess PETER gestartet.

Der Zustand "nicht bereit" wird durch Abdunkelung des Datenpfades in dem zugehoerigen Bildschirmsegment dargestellt. Im Zustand "warten" ist das Startprogramm fuer den Prozess PETER in Betrieb, der Prozess PETER kann dann gestartet werden.

(2) Programmstart und Vorbereitungen. Diese Information-Unit meldet den Verlauf des Programmstarts. Betriebsbereitschaft oder Fehler werden dem Monitor angezeigt. Die Uhrzeit der Beendigung dieses Abschnitts ist waehrend des Schussbetriebs staendig auf dem Bildschirm sichtbar.

(3) Wait. Der Prozess PETER (Einzelheiten des Daten- und

Informationsflusses s. Fig. 26) wird fast staendig in dieser Information-Unit verweilen. Um Unsicherheiten des Operators zu vermeiden, wird die Lesebereitschaft in Abstaenden von 1 Minute geprueft. Sobald diese Abfrage mit Timeout beantwortet wird, wird die auf dem Bildschirm fuer WAIT aufgehellte Figur in "blink"-Zustand versetzt (moeglich nur mit Zusatzeinrichung ECSA) und eine entsprechende Fehlernachricht aus-

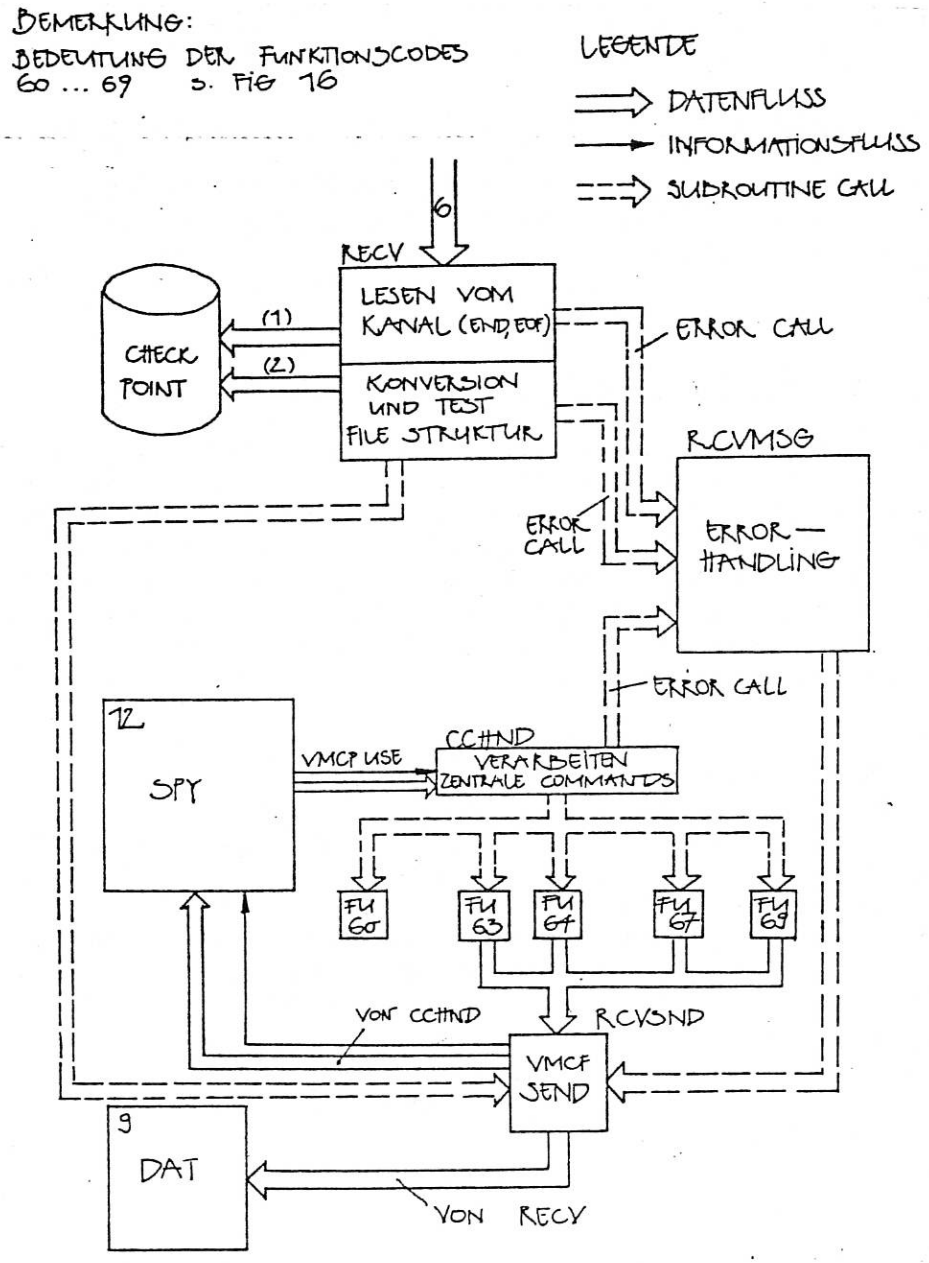


FIG 26: Übersicht VM (exp) RCV

gegeben. Bei einem Interrupt, der WAIT unterbricht, wird die fuer diese Program-Unit aufgehellte Figur auf dem Bildschirm abgeblendet und die Durchfuehrung des jeweiligen Interrupt-Handlers durch Aufleuchten der relevanten Figur auf dem Bildschirm angezeigt.

(4) Verarbeitung der Daten und Senden. Die Verarbeitungszeit fuer Lesen, Konversion und Senden ist so kurz, dass Einzelanzeige nicht moeglich ist. Man muss sich auf die Angabe

- des Verweilens in diesem Abschnitt,
- der Versuchsnummer (bzw. Schussnummer),
- der Fehler (mit Blocknummer) und
- der Beendigung dieses Abschnitts mit Datenmenge

beschraenken. Bei Beginn der VMCF-Uebertragung an die VM <exp> DAT wird auch die dafuer vorgesehene Figur aufgehellt.

(5) Abschluss und Exit. Bei Eintritt in diese Information-Unit wird ebenfalls die relevante Figur des Prozess-Flussplans auf dem Bildschirm aufgehellt. Fehler beim Abschluss von Files und die Uhrzeit der Beendigung werden auf den Bildschirm ausgegeben.

4.3 Testhilfen und Recovery

Bislang ist automatisches Recovery nicht geplant. Um manuelle Eingriffe zu steuern, muessen Testhilfen, die das Einkreisen von Fehlerquellen erlauben, und Checkpoints, ab denen Programm-Units wiederholt werden koennen, vorgesehen werden. Zusaetzlich sollten Routinen fuer die Zeitmessung nach Bedarf an- oder abgeschaltet werden koennen. Diese Einrichtung wird beim Restart des Prozesses nach Abbruch des Programmes oder Systemzusammenbruch ebenfalls von Vorteil sein.

Die bisherigen Erfahrungen zeigen, dass die im Programmpaket EDDAR vorgesehenen Testhilfen im wesentlichen zum Auffinden von Fehlern ausreichen. Aus Gruenden der besseren Uebersicht ist jedoch eine Unterteilung der Programme in Test-Units, die gezielt angesprochen werden koennen, erwuenscht. Durch Definition eines Vektors logischer Groessen im Programm und deren Steuerung von der zentralen Konsole aus, kann Information aus wohldefinierten Testbereichen abgerufen werden.

Durch die Struktur des Prozesses sind als Checkpoints in natuerlicher Weise

(1) Kanal,

(2) Lesen und

(3) Konversion und Test des Datenfiles

definiert.

(1) Kanal. Bislang wird die Uebertragung von Datenbloecken im Knotenrechner ueberwacht, bei Fehlern die Uebertragung eines Blocks wiederholt. Beim Auftreten weiterreichender Stoerungen muss die Uebertragung des Schussfiles wiederholt werden. Durch Beobachtung ueber laengere Zeit und eine Kosten-Nutzen-Analyse sollte entschieden werden, ob an dieser Stelle durch einen leistungsfaehigeren Knotenrechner ein Checkpoint eingerichtet werden kann, der im Stoerfalle durch

Zwischenspeicherung der Daten Hilfe bieten kann.

Bemerkungen:

(a) Ein leistungsfaehigerer Knotenrechner bietet ausserdem Testhilfen fuer die Ueberwachung und das Testen der Uebertragungs-Hardware.

(b) Das Senden zusammenhaengender Files ist -auch in diesem Zusammenhang- nicht zu empfehlen.

(2) Lesen. Besteht der Checkpoint (1) nicht, muss bei Lesefehlern die Uebertragung eines vollstaendigen Files durch den Experiment-Rechner wiederholt werden. Andrerseits ist es angebracht, die uebertragenen Daten (noch vor der Konversion) durch Speichern auf einer DASD (Direct Access Storage Device - Minidisk) zu sichern (Fig. 26). Daraus ergeben sich zwei Vorteile:

- bei Lesefehlern muessen nur die fehlerhaften Bloecke wiederholt werden,

- bei Fehlern in der Konversion, abnormal Exit des Prozesses oder Zusammenbruch des Systems kann mit diesem Checkpoint fortgefahren werden.

Bei Uebertragung eines einzigen Datenstroms bleibt zwischen zwei Bloecken genuegend Zeit, fuer mehrere Datenstroeme sind genaue Untersuchungen des zeitlichen Verlaufs erforderlich.

(3) Konversion und Test der Datenstruktur. Sofern beim Test der Datenstruktur oder bei der Konversion Fehler auftreten, kann nach deren Beseitigung auf die Daten des Checkpoint (2, Fig. 26) zurueckgegriffen werden. Werden die Daten von der VM <exp> DAT nicht angenommen, muessen sie ebenfalls auf DASD gerettet werden. Fuer den Notbetrieb sollte dieser Checkpoint von einer dafuer vorgesehenen VM betrieben werden, die dann die Benutzer bedient.

Bemerkung: Zur Steigerung der Effizienz wird empfohlen, von der Subroutine GETBLK nur die Bytes 3...514 abzunehmen.

4.4 Eingriffe

Der Start der Prozesse PETER (Verarbeitung von Experimentdaten) und STRGEN (Erzeugung von Struktur-Vektoren) wird vom Monitor COMET aus eingeleitet. Die VM <exp> RCV wartet dazu mit EXEC RCVWT auf einen EXTERNAL Interrupt (von VMCF) und interpretiert das Startcommand. Je nach Code wird der Prozess PETER mit Hilfe des EXEC STRTPTR oder der Prozess STRGEN durch EXEC STRGSTRT gestartet. In allen anderen Faellen verbleibt die VM <exp> RCV im Wartezustand. Wird, aus welchen Guenden auch immer, die LOGOFF-Prozedur durchgefuehrt, muss die VM <exp> RCV durch LOGON wieder aktiviert, das EXEC RCVWT neu gestartet und ggf. das Startcommand von Monitor COMET aus wiederholt werden. Selbstverstaendlich koennen die EXEC STRTPTR und STRGSTRT auch von der VM <exp> RCV aus ohne Umwege manuell gestartet werden.

Die Eingabedaten fuer den Betriebszustand werden bei automatischem Start mit Hilfe einer eingegebenen Kennziffer vom Monitor erzeugt und (ohne weitere Abfragen) an den Console-Stack der VM <exp> RCV uebergeben. Bei manuellem Start sind die durch die Prozesse (im Startabschnitt) vorgegebenen Fragen (Normalbetrieb und/oder Test, Disconnect, Eingabe Testvektor usw.) nacheinander zu beantworten.

4.5 Folgerungen

Die im Abschnitt 5.2 entwickelte Struktur des Prozesses PETER in der VM <exp> RCV erlaubt Erweiterungen und Aenderungen aller Aktionen zur Ueberwachung und Steuerung des Datenflusses mit einfachen Mitteln. Dann bleiben die Abschnitte unberuehrt, die die Experimentdaten verarbeiten. Die Eingriffe koennen auf das Wesentliche beschraenkt und die Entwicklung den Erfahrungen jederzeit angepasst werden.

Die Informationen, die die Eingriffe am laufenden Prozess vermitteln, beanspruchen die volle Aufmerksamkeit des Operators, sodass die Ueberwachung des restlichen Datenstroms zum Zeitpunkt eines Eingriffes -obwohl wuenschenswert- nicht mehr moeglich ist. Das Layout des Bildschirms ist dieser Gegebenheit anzupassen, ggf. auch der Anschluss eines weiteren Bildschirms vorzusehen.

5. Struktur der VM <exp> MIG

5.1 Backup von Experimentdaten

Ein stetig wachsender Teil der Datenbank muss auf Baender ausgelagert werden und von dort aus, falls angefordert, wieder beschafft zu werden. Im aktuellen System EDDAR sind zwar geeignete Prozesse vorhanden, die Benutzung aber umstaendlich und nicht einheitlich.

Der Entwurf des Prozesses DATMON fuer das Experiment W VII a wurde seinerzeit ueberstuerzt durchgefuehrt, sodass notwendige Ueberlegungen hinsichtlich dieses Teils der Datenbank unterblieben. Auch der Prozess EMPFN fuer das Experiment ASDEX konnte nur mit mangelhafter Vorbereitung entwickelt werden. Fuer das Backup stehen daher derzeit zwei stand-alone Prozesse

- SAMIGR zum Auslagern der Schussdaten von Platte auf Band und
- PRORET zum Wiederbeschaffen der ausgelagerten Daten

zur Verfuegung. Sie werden jeweils auf Anforderung -in einigen Faellen unter Schwierigkeiten- manuell gestartet. Die Entwicklung zum weitgehend automatischen Backup wurde inzwischen eingeleitet.

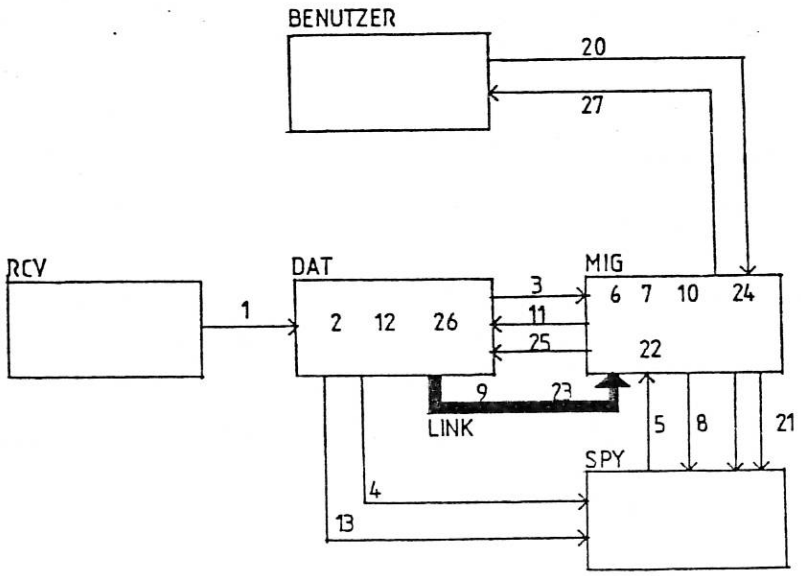


FIG 27: ZUSAMMENWIRKEN DER VM BEI AUSGELAGERTEN DATEN

Legende zu Fig. 27

1 Schuss 1 mit Request 2
2 aktuelles Migrate Intervall
3 Migrate Intervall mit Request 44
4 Request 44 mit Anforderung der Startdaten
5 Startdaten mit Request 45/1
6 Adresse Migrate Intervall und Startdaten in Warteschlange
7 Warteschlange verarbeiten
8 Information ueber Zustand Bandeinheit, Band, Schussnummer
9 Lesen Schussdaten von VM <exp> DAT
10 Update Migrate Directory
11 Migrate Directory mit Request 5
12 Update File Directory
13 Fertigmeldung
20 Senden Schussvektor mit Request 41
21 Request 41 mit Anforderung von Schussdaten
5 Startdaten mit Request 45/2
22 Adresse Schussvektor mit Requestor und Startdaten in Warteschlange
7 Warteschlange verarbeiten
8 Information ueber Zustand Bandeinheit, Band, Schussnummer
23 Beschaffen File Directory
24 Lesen Schussdaten vom Band
25 Senden Schussdaten
26 Abspeichern Schussdaten in Datenbank
27 Fertigmeldung an Benutzer

5.2 Prozesse MIGRAT und BACKUP

Das Zusammenwirken der Prozesse MIGRAT und BACKUP mit den

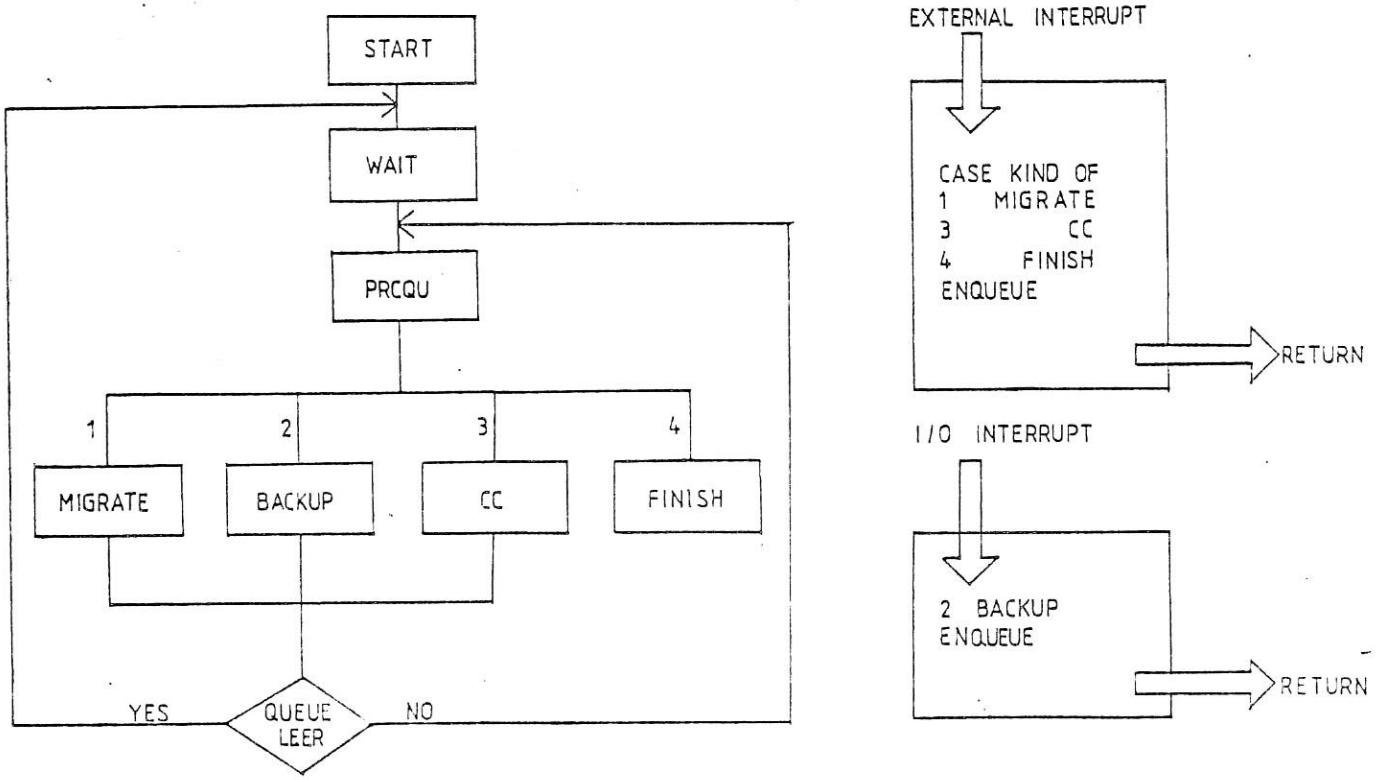


FIG 28: PROZESSE IN VM <exp> MIG

Prozessen DATMON und COMET ist in Fig. 27 dargestellt. Jeder dieser Prozesse wird nach Erkennen des fuer ihn erzeugten Interrupts gestartet. Die Wartezeit auf Zuordnung einer Bandeinheit oder Auflegen des Bandes kann sehr unterschiedlich sein, sodass staendig alle Interrupts zugelassen sein und in eine Warteschlange eingereiht werden muessen. Aus diesen Ueberlegungen folgt das Flussdiagramm Fig. 28.

(1) START. Dieser Programmteil fuehrt die VMCF-Funktion AUTHORIZE aus.

(2) WAIT. Der Prozess befindet sich fast staendig in diesem Programmteil, nur unterbrochen durch Interrupts. Nach jedem Interrupt wird entweder der unterbrochene Prozess fortgesetzt oder der Programmteil ENQUEU angesteuert.

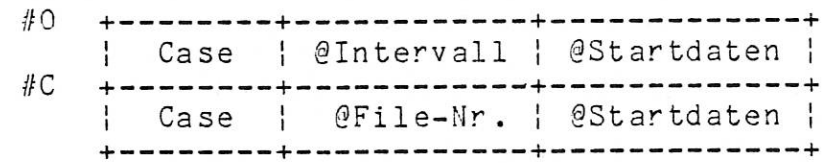


Fig. 29 Queue-Elemente

(3) ENQUEU. Jeder Request wird als Queue-Element (Fig. 29) in die Warteschlange eingereiht und danach entweder der unterbrochene Prozess fortgesetzt oder die Warteschlange weiter abgearbeitet. Eine Endbedingung wird durch Ueberschreitung der vorgegebenen Uhrzeit und nach Abarbeiten aller Requests erzeugt.

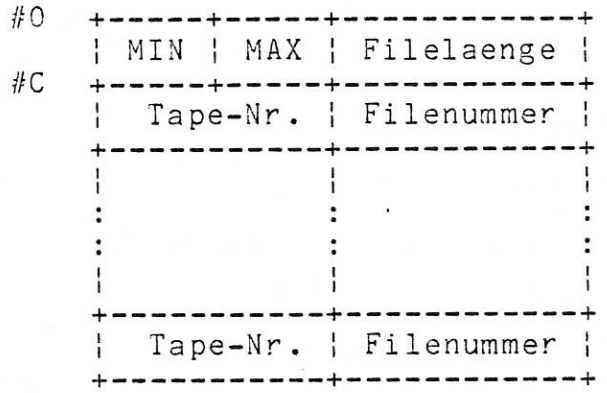


Fig. 30 Migrate Directory

(4) MIGRAT. Durch Request 44 (s. Fig. 16), der gleichzeitig das Migrate-Intervall sendet, werden

- die Startdaten (s. Abschnitt 1.4) von VM <exp> SPY angefordert,
- Case 1, die Adresse des Migrate-Intervalls und die Adresse der Startdaten als Qeue-Item in die Warteschlange eingereiht,

- nach Erledigung der davorliegenden Requests der Prozess MIGRAT gestartet und

- VM <exp> DAT durch CP-Command LINK und CMS-Command ACCESS mit der VM <exp> MIG gekoppelt.

Er fordert, falls notwendig, ueber eine Message an VM MSG eine Bandeinheit an und laesst gemaess Filedirectory FILEINFO ein Band auflegen. Anschliessend werden alle auszulagernden Files auf Band ausgegeben , Band- und Filenummer in der Migrate Directory (Fig. 30) eingetragen. Bei Erkennen des Bandendes wird

- das aktuelle Band zurueckgespult und entladen,

- das naechste Band der Filedirectory FILEINFO ueber Message an die VM MSG angefordert,

- die Filedirectory FILEINFO berichtigt und

- danach weitere Files ausgegeben.

Nach Abarbeiten der Migrate Directory wird

- mit CMS-Command RELEASE und CP-Command DETACH die Kopplung zur VM <exp> DAT aufgehoben,

- das zuletzt beschriebene Band entladen und

- falls kein weiterer Request in der Warteschlange vorhanden ist, die Bandeinheit freigegeben.

Anschliessend wird, je nach Zustand, einer der Programmteile WAIT oder ENQJEJ aufgerufen.

(5) BACKJP. Mit dem EXEC RETRVL erzeugt der Benutzer in seiner VM ein Punchfile, das an die VM <exp> MIG gesendet und dort als Readerfile gespoolt wird. Durch I/O-Interrupt des virtuellen Readers wird die Programmeinheit ENQJEJ, die

- die Startdaten von der VM <exp> SPY beschafft,

- Case 2, die Adressen der Anforderung und der Startdaten in die Warteschlange eintraegt,

aufgerufen. Sobald dieser Item abgearbeitet werden soll, wird der Prozess BACKJP gestartet. Zu Beginn wird durch CP-Command LINK und CMS-Command ACCESS die VM <exp> DAT gekoppelt, die Filedirectory kopiert, danach mit CMS-Command RELEASE und CP-Command DETACH die VM <exp> DAT wieder abgekoppelt.

Aus allen zu dieser Zeit vorliegenden Requests der Warteschlange wird eine sortierte Folge der angeforderten Schussnummern generiert. Es wird geprueft, ob sie on-line oder off-line sind oder nicht uebertragen wurden. Band- und Filenummern der off-line Schuesse werden sortiert und danach ueber eine Message an die VM MSG, falls noetig, eine Bandeinheit angefordert, in der gleichen Weise die notwendigen Baender verlangt und schliesslich die eingelesenen Daten

ueber VMCF in Bloecken von 256 KBytes an VM <exp> DAT uebertragen. Der Bandwechsel wird in gleicher Weise wie im Prozess MIGRAT durchgefuehrt.

Abschliessend wird, je nach Zustand, einer der Programmteile WAIT oder ENQUEU aufgerufen.

(6) CC - Central Commands. Dieser Programmteil verarbeitet Requests von der VM <exp> SPY fuer Start oder Eingriffe in die VM <exp> MIG. Naehere Einzelheiten im COMET System Programmer's Guide.

(7) FINISH. Dieser Programmteil wird nach einer vorbestimmten Uhrzeit bei leerer Warteschlange aufgerufen. Im wesentlichen wird die VMCF-Funktion UNAUTHORIZE durchgefuehrt.

Zur besseren Absicherung der Datenbestaende auf Band ist -auch in CMS- die Pruefung der Label der aufgelegten Baender und die Entfernung der Schreibringe nach Bandwechsel sicherzustellen.

5.3 Informationsfluss

Zur Uebersicht ueber den Datenfluss erscheint auf dem Bildschirm der VM <exp> SPY der Name der aktiven Program-Unit oder, falls VM <exp> MIG nicht gestartet oder ausgefallen ist, "OFFLINE". Weitere Angaben sind nur fuer MIGRAT oder BACKUP erforderlich :

- Anfordern einer Bandeinheit und die Nachricht DEV 181 ATTACHED AT TIME hh:mm:ss,
- Anfordern eines Bandes mit Namen und Zeit der Anforderung, bzw. Zeit des Auflegens,
- auszulagerndes Intervall mit Angabe des aktiven Files,
- Bandwechsel und Anforderung von Bandnachschub.

5.4 Testhilfen und Recovery

Die im aktuellen System EDDAR vorhandenen Testhilfen reichen im wesentlichen aus. Einige Utilities fuer die Pruefung von Filestrukturen, wie sie bei on-line Files ueblich sind, muessen eingefuehrt werden. Wegen unvermeidbarer Stoerungen ist Recovery der Directories durch

- Kopieren im Ausgangszustand,
- Ausgabe der Aenderungen

vorgesehen. Datenfiles koennen jederzeit wieder beschafft werden, weitere Massnahmen sind nicht erforderlich.

5.5 Eingriffe

Die bisherigen Beobachtungen zeigen, dass Eingriffe von aussen fuer

- Start und Anhalten der Prozesse,
- Messages an die VM MSG und
- gewisse Bandoperationen

notwendig sind. Mit den im CC vorgesehenen Operation-Codes sind alle Forderungen abgedeckt.

6. Struktur der VM <exp> DAT

6.1 Uebersicht

Die VM <exp> DAT verwaltet die Experimentdaten -auch nach der Auslagerung- und fuehrt alle Transaktionen, die fuer Empfang und Anforderung von Daten erforderlich sind, ueber VMCF-Funktionen aus. Die Schreiberlaubnis liegt ausschliesslich bei dieser VM, Leseerlaubnis besteht fuer alle VM--Benutzer. Aus dieser Situation ergibt sich eine Vielzahl von Requests, die grosse Datenmengen zu bewegen haben. Ausserdem muss diese VM staendig in Betrieb sein, um auch ausserhalb des Schussbetriebs fuer die Benutzer bereit zu stehen. Eine Verknuepfung mit anderen VM (insbesondere der Benutzer) ueber CP-Command LINK ist wegen der damit verbundenen Nachteile nicht vorgesehen.

Der Datenstrom bewegt sich in gerader Linie von der VM <exp> RCV ueber die VM <exp> DAT zu den VM der Benutzer. Zwischen zwei Schuessen befindet sich jeweils ein File im Kernspeicher.

Man unterscheidet bei Datenbanken allgemein

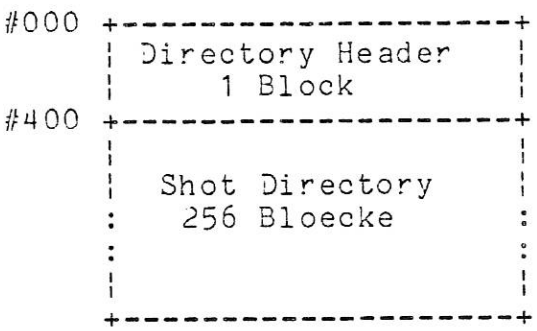
- Datenbasis (DB) und
- Datenverwaltung,

Elemente, die, wie sich in der Praxis gezeigt hat, ohne Nachteile in einer VM integriert werden koennen.

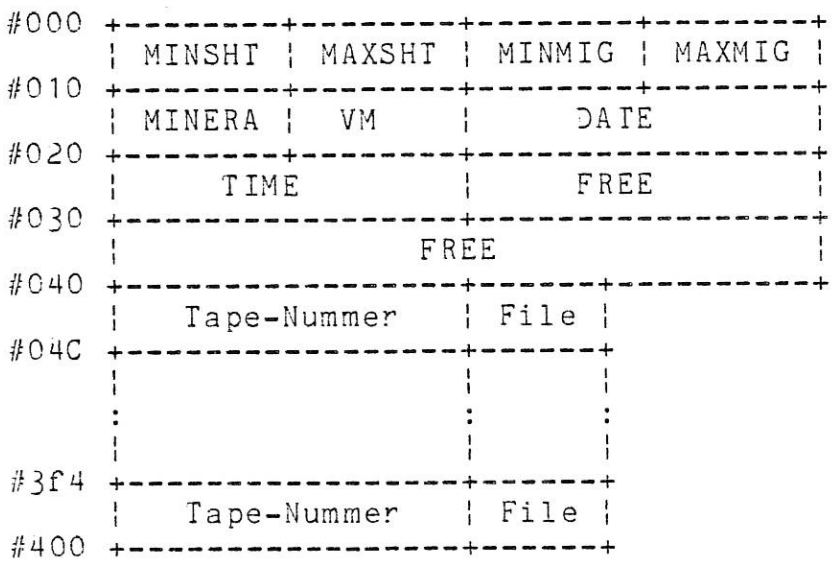
(1) Datenbasis. Sie umfasst den Kernspeicher, Platten, Massenspeicher und Baender, auf denen die Daten eines Experiments gespeichert sind. Techniken zur Verwaltung mehrerer Volumes als Minidisks in CMS fuer den on-line Teil der Datenbank wurden schon im System EDDAR mit Erfolg angewendet. Die Organisation der DB kann erheblichen Einfluss auf die Zugriffszeiten ausueben, sodass staendige Ueberwachung durch geeignete Statistiken, die von der Datenverwaltung gefuehrt werden, notwendig ist. Es ist durchaus denkbar, dass die Organisation der DB von Zeit zu Zeit entsprechend der durch die Auswertung der Statistiken gewonnenen Erkenntnisse (andere Auswertung, zusaetzliche Daten, geaendertes Benutzerverhalten, System-Aenderungen) angepasst werden muss. Die von der VM <exp> RCV ueber VMCF angelieferten Experimentdaten werden auf Platte (mit CMS-Macro FSWRITE) ausgegeben, stehen aber im Kernspeicher zur Verfuegung. Das Lesen der Daten -gleichgueltig, von welchem Speicher- wird von der DB mit schnellen Transaktionen (MOVE, CMS-Macros FSREAD, RDTAPE usw.), die Weitergabe an die Benutzer ueber VMCF-Funktionen durchgefuehrt.

(2) Datenverwaltung. Es sind drei unterschiedliche Funkti-

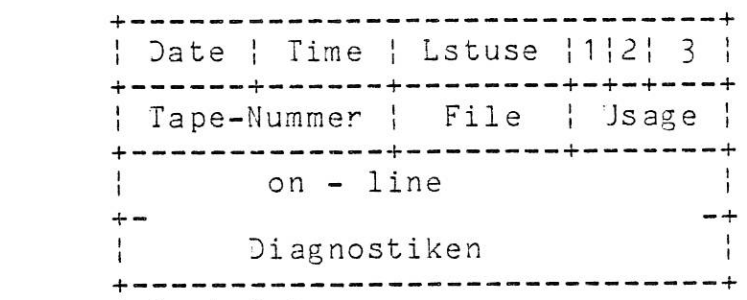
onen zu erfuellen:



a) Uebersicht



b) Header



1 Lock Byte
2 On-line Byte
3 free

c) Shot Directory

Fig. 31 Organisation der File Directory

- Verwaltung externer Speicher,
- Anforderungen der Benutzer,

- Service und Utilities.

(a) Verwaltung externer Speicher. Sie wird in der File Directory Fig. 31 abgewickelt.

Bei dieser Organisation werden 4096 Schuesse in einer File Directory verwaltet. Haelt man staendig drei Directories on-line, so ist Information ueber mindestens 8192 Schuesse zu jeder Zeit greifbar. Nicht aktuelle File Directories werden auf Band ausgelagert.

Die bisher uebliche Praxis, auf allen Minidisks der DB maximale Menge an Daten zu speichern und fuer den aktuellen Schuss erst kurz vor dem Abspeichern Platz zu schaffen, hat sich bewaehrt und kann beibehalten werden. Das on-line Byte wurde zusaetzlich eingefuehrt, da mehr als bisher mit der Wiederbeschaffung ausgelagerter Files gerechnet werden muss.

Die relevante File Directory ist staendig im Kernspeicher; es sind keine I/O-Operationen notwendig, um Information ueber aktuelle Files zu erhalten.

Nach Ende des Schussbetriebs oder spaetestens, sobald weniger als eine vorbestimmte Zahl Schussfiles geloescht werden kann, wird Migrate durch Request 44 mit Uebersendung des relevanten Migrate Intervalls an die VM <exp> MIG gestartet. Die Rueckgabe der notwendigen Information mit der Migrate Directory, die danach wieder in die File Directory eingeblendet wird, erfolgt mit Request 5 von der VM <exp> MIG.

(b) Anforderungen der Benutzer. Man unterscheidet zwei Moeglichkeiten:

- automatische oder beschraenkte automatische Anforderung,

- vom Benutzer gesteuerte Anforderung.

Im ersten Fall erwartet der Benutzer zu Anfang des Schussbetriebes oder nach abgeschlossener Auswertung eines Schusses eine Diagnostik des naechsten Schusses. Da er nicht weiss, wann diese Daten greifbar sind, muss dieser Request in eine Warteschlange eingereiht werden. Diese Moeglichkeit ist allerdings auf den Schussbetrieb beschraenkt. Er wird sofort nach Eingang dieser Daten des folgenden Schusses ueber VMCF aus dem Kernspeicher befriedigt. Im anderen Falle (z. B. weitere Diagnostiken des aktuellen Schusses oder frueherer Schuesse) wird ein Request aus der VM des Benutzers gesendet und von der VM <exp> DAT sofort ausgefuehrt. Die verlangten Daten werden, falls notwendig, von der Platte eingelesen oder aus dem Kernspeicher beschafft und dem Benutzer ueber VMCF uebergeben.

Eine Ausnahme muss bei der Anforderung ausgelagerter Daten gemacht werden. Die schon im Abschnitt 5.1 geschilderten Umstaende fuehren zu getrennter Behandlung.

(c) Utilities und Service. Bislang sind die Requests 21 ... 54 (Fig. 16) vorgesehen. Mit ihnen wird im wesentlichen Information ueber Datenfiles beschafft. Dieser Service ist

ohne Schwierigkeiten kuenftigen Entwicklungen anzupassen.

6.2 Prozess DATMON

Er verwaltet die Datenbank und steht somit mit allen angeschlossenen VM -auch aller am Rechenbetrieb beteiligten Benutzer- ueber VMCF in Verbindung. In diesem Zusammenhang ist die Fileorganisation noch ohne entscheidende Bedeutung, vielmehr soll die Struktur des Prozesses diskutiert werden. Die bisherigen Ueberlegungen bezueglich des Zeitablaufs bei der Datenuebertragung und -verarbeitung wurden mit der Feststellung abgeschlossen, geeignete Teilmengen der Experimentdaten durchzureichen. Ein Beispiel zu diesen Ueberlegungen ist in Fig. 32 dargestellt. Daraus folgt sofort, dass die

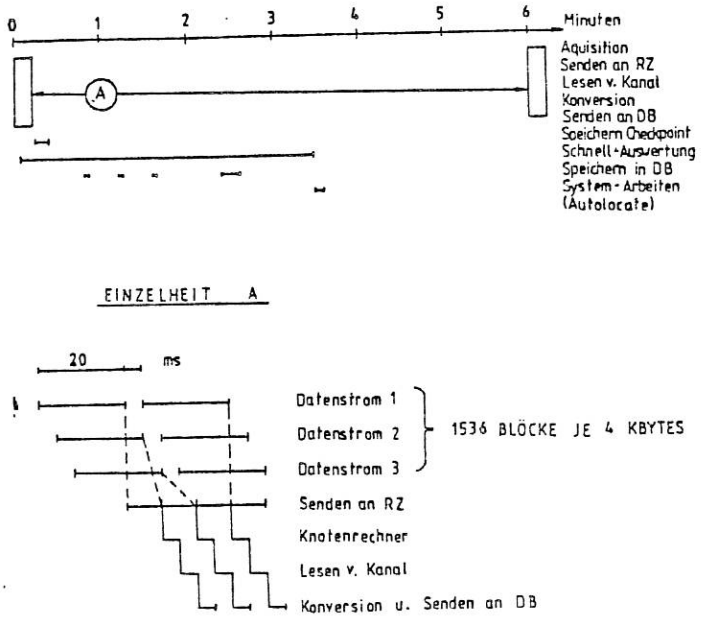


FIG 32: ZEITLICHER VERLAUF ÜBERTRAGUNG-AUSWERTUNG

Requests der Benutzer -gleichgueltig, ob automatisch oder vom Benutzer gesteuert-, fuer das Abspeichern in der Datenbasis und fuer Backup in eine Warteschlange eingereiht werden muessen, um die Benutzer im Dialog optimal zu bedienen. Requests fuer die Uebergabe der Daten aus der VM <exp> RCV muessen selbstverstaendlich ohne Aufenthalt ausgefuehrt werden, Unterbrechungen sind daher nicht erlaubt.

In Anbetracht der fuer die Datenbank belegten Requests (Fig. 15) existieren vorerst folgende Program-Units:

- START,
- WAIT,
- COPYFILE,

- ENQUEU,
- Locate, Autolocate,
- Migrate, Backup,
- Utilities,
- CC und
- Close.

(1) START. Die staendig im Kernspeicher befindlichen Daten, wie File Directory und User Directory werden eroeffnet und eingelesen, die VM wird fuer VMCF autorisiert und der VM <exp> SPY der Start, die Startzeit und, wenn noetig, aufgetretene Fehler mitgeteilt. Als Verwaltungsprogramm einer Datenbank soll soll der Prozess DATMON staendig in Betrieb sein.

(2) WAIT. Der Prozess DATMON verweilt, sofern keine Interrupts abgehandelt werden, in diesem Abschnitt. Eine Endabfrage, wie in anderen Prozessen, existiert nicht. Der Prozess wird durch CC (Stop-Command) von der VM <exp> SPY aus angehalten.

(3) COPYFILE. Man unterscheidet Annahme der Messdaten, Suchen nach Speicherplatz und das Abspeichern in die Datenbasis. Der EXTERNAL-Interrupt der VM <exp> RCV, mit dem die Daten angeliefert werden, wird -ohne jegliche Unterbrechung- sofort abgehandelt. Die ankommenden Daten werden (im derzeitigen Stadium der Planung, spaetere Aenderung nicht ausgeschlossen) gemaess dem Configuration File und den Angaben ueber Diagnostik-, Geraet- und Blocknummer, beginnend in einer dadurch definierten Speicherzelle im Kernspeicher abgelegt (s. Anlage A-1). So ist schnelles und einfaches Wiederauffinden sichergestellt. Das Abspeichern dagegen erfolgt (wieder nach dem derzeitigen Stand der Planung) in Diagnostikfiles in 1024-Byte-Bloecken (s. Anlage A-2) ueber Requests, die in eine Warteschlange eingereiht werden. Die Datensicherung wird durch

- Kopieren der Daten in die VM der Benutzer und
- Ablage einer Kopie auf der A-Disk der VM <exp> RCV

durchgefuehrt. Die letzte Kopie kann vom Prozess DATMON auf Kommando aus der VM <exp> SPY von der VM <exp> RCV gelesen werden.

(4) ENQUEU. Alle weiteren Requests werden in eine Warteschlange eingereiht und danach sequentiell ausgefuehrt.

(5) Locate, Autolocate. Beide Operationen unterscheiden sich durch Request- und Schuss-Nummer. Fuer Autolocate gilt:

Request: 1, Schuss-Nummer: 0,

bei Eintreffen des naechsten Schusses wird dessen Schussnum-

mer als aktuelle Schussnummer eingesetzt. Fuer Locate (Request 0) muss eine vorhandene Schussnummer angegeben sein. In beiden Faellen muss eine Diagnostik-Nummer bekannt sein, da jede Anforderung als Beschaffung der vollstaendigen Daten einer Diagnostik (maximal 256 KBytes) interpretiert wird. Sofern eine Diagnostik mehr Daten umfasst, enthaelt die Anforderung ausserdem die Block-Nummer das Beginns der Folgedaten.

Die VM der Benutzer sendet Requests aus dem Unterprogramm READR des Leseprogramms GTSHOT.

(6) Migrate, Backup. Nach Beendigung des Schussbetriebes durch Schuss "1") wird ein Request mit dem relevanten Migrate Intervall an die VM <exp> MIG abgesetzt. Er startet den Prozess MIGRAT, der nach Beendigung des Auslagerns die Daten der Migrate Directory mit Request 5 an die VM <exp> DAT zurueckgibt.

Zu jeder beliebigen Zeit koennen ausgelagerte Schuesse wieder angefordert werden. Sobald in der VM <exp> MIG der Prozess BACKUP gestartet werden soll, wird mit Request 41 der relevante Ausschnitt der File Directory angefordert. Die vom Band gelesenen Schuesse werden in Bloecken von 256 KBytes von der VM <exp> MIG an die VM <exp> DAT uebergeben. Sie werden in aehnlicher Art wie mit COPYFILE der Datenbank uebergeben.

(7) Utilities. Eine Reihe von Utilities (s. Fig. 16) ist fuer Auskuenfte ueber den Datenbestand in der Datenbasis vorgesehen. Sie lassen sich ohne Schwierigkeit erweitern. Die Benutzer fordern die Dienste ueber den Module SHOW (in der VM <exp>) an. Die Wahl der Utilities ist benutzerfreundlich (Menu) gestaltet, die Ausgabe der Information beschleunigt, die Belastung des Rechners optimiert.

(8) CC. Wie auch in den anderen VM laesst sich der Datenfluss der VM <exp> DAT von der VM <exp> SPY aus ueber zentrale Kommandos steuern. Einzelheiten findet man im COMET System Programmer's Guide.

(9) FINISH. Diese Program-Unit schliesst nur den VMCF-Verkehr ab. Ausgabe und Schliessen von Files entfaellt, da jede Aenderung waehrend des Betriebs abschliessend ausgefuehrt wird.

6.3 Informationsfluss.

Die VM <exp> DAT ist im aktuellen System EDDAR zur Kontrolle des Datenflusses, gemaess der folgenden Figur 33 stets in Kontakt.

(1) VM <exp> RCV. Beide VM haben zeitlich voneinander unabhaengige und scharf begrenzte, nicht synchronisierbare Funktionen zu erfuellen. Im System EDDAR entstand aus diesem Zusammenhang im SEND/RECEIVE Zyklus (Fig. 33) Schwierigkeiten, die nur durch zeitliche Einschraenkungen geloest werden konnten. Im System COMET entfaellt die Meldung, sodass von der VM <exp> DAT zur VM <exp> RCV keine Information fliesst.

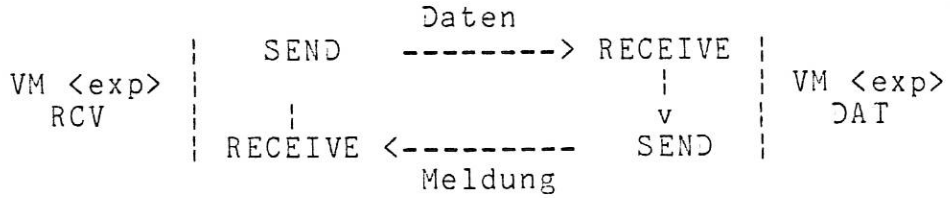


Fig. 33 SEND/RECEIVE Zyklus in EDDAR

(2) VM der Benutzer. Die Benutzer fordern Dienste an und erhalten sie ohne weitere Bestaetigung. Zwischen den VM der Benutzer und der VM <exp> DAT werden keine Informationen ausgetauscht, andrerseits dient die VM <exp> DAT der VM <exp> SPY als Quelle der Information ueber die Benutzer.

(3) VM <exp> MIG. Auch in diesem Falle werden nur Daten, keine Information ausgetauscht.

(4) VM <exp> SPY. Diese zentrale VM erhaelt staendig Informationen ueber

- das Schussfile im Kernspeicher,
- Schussfiles des Schusstages,
- Funktion der Programmteile,
- Anforderungen der Benutzer,
- Anfordern Migrate mit Migrate Intervall,
- Ablage Schussfiles durch Backup,
- Nachrichten ueber CC.

6.4 Testhilfen und Recovery

Der Prozess DATMON besteht aus den im Abschnitt 6.2 genannten Program-Units, deren jede ebenfalls Recovery- und Test-Unit ist.

(1) COPYFILE. Recovery des Schussfiles ist wegen Speicherung auf der A-Disk der VM <exp> RCV moeglich. Directories muessen vor Beginn des Update oder zu festen Zeiten gerettet und die Daten des Update gespeichert werden. Als Testhilfen werden Ausdrucke der Filestruktur, der I/O-Aktivitaeten und des freien Speicherplatzes benoetigt.

(2) Migrate. Recovery der Directories ist in COPYFILE beschrieben. Man erwartet Testhilfen durch Ausdruck des Migrate Intervalles, zusaetzliche Information ueber fehlende Schussfiles bei der Rueckgabe mit Fehlerindex.

(3) Backup. Recovery der Anforderung und fehlender Teile von

Files oder Fortsetzung der Wiederbeschaffung ausgelagerter Experimentdaten.

6.5 Eingriffe

Ausser dem Start und dem Anhalten der VM koennen weitere Eingriffe notwendig werden:

- Loeschen von VMCF-Requests,
- Anhalten oder Wiederanfahren der Queue,
- Abschalten von Benutzern mit nicht erfuellbaren Datenanforderungen,
- Loeschen stoerender Files,
- Festhalten oder Freigabe von Files,
- Migrate nicht ausgelagerter Files

usw. Die vorgeschlagenen Operationen im CC erlauben diese Eingriffe ohne Schwierigkeiten, sofern die Program-Units dazu ausgelegt sind. Das System kann an weitere Anforderungen angepasst werden.

6.6 Ausblick

Die VM <exp> DAT ist durch die Vielzahl der Requests naturgemaess stark belastet. Im Gegensatz zu Untersuchungen mit ungeeigneten Mitteln kann gemaess Zeitmessungen im Anhang A-4 bei Normalbetrieb der Rechenanlage als sicher angenommen werden, dass keine Engpaesse entstehen, Abschaetzungen werden weiter unten vorgestellt. Mehrere, nicht zeitbedingte Gruende legen es jedoch nahe, eine zweite (Duplikat-) VM zu generieren.

(1) Stoerungen in CMS (OS der VM). Es wurden mehrfach Stoerungen beobachtet, die sich kurzfristig, jedoch nicht innerhalb einer Schusspause, beseitigen lassen. Durch Umschalten auf eine Duplikat-VM kann der Datenfluss aufrecht erhalten werden.

(2) Entwicklung. In einer zweiten VM kann die Weiterentwicklun des Systems betrieben und getestet werden. Bei Stoerungen kann ohne Aufenthalt umgeschaltet werden.

(3) Untersuchung von Datenstrukturen. Die Struktur der Datenfiles, besonders beim Entmultiplexen von Datenstroemen, Lesen von mehreren Kanaelen, Verarbeiten mehrerer Experimente in einer VM, kann, wie die Beobachtung zeigt, zerstoert sein. In einer Duplikat-VM koennen geeignete Untersuchungen ohne Stoerung des Betriebs durchgefuhert werden.

Im derzeitigen Stadium der Entwicklung laesst sich noch nicht voraussagen, ob die in diesem Kapitel beschriebenen Aufgaben ohne Stoerungen von einer einzelnen VM wahrgenommen werden koennen. Gegebenenfalls ist der Einsatz einer zentra-

len VM mit Satelliten, von denen jeder einen oder mehrere Prozesse (parallel) abwickelt, moeglich. Bislang ist jedoch an einen Rechner mit mehreren Prozessoren oder einen Rechnerverbund nicht gedacht. In beiden Faellen sollten grundlegende neue Untersuchungen angestellt werden.

7. Struktur der VM <exp> SPY

7.1 Allgemeines

Die Struktur dieser VM ergibt sich im Wesentlichen aus den Eigenschaften der in den vorhergehenden Kapiteln vorgestellten VM. Ihre Organisation im aktuellen System EDDAR laesst viele Wuensche offen:

- die Uebersicht ueber den Datenfluss ist zu keiner Zeit gesichert,
- die Information ueber Nebensaechlichkeiten ist zu aufdringlich,
- Der Wechsel des Bildschirminhaltes erfolgt in zu kurzen Zeitabstaenden.

usw. Der Gestaltung der Texte auf dem Bildschirm (s. auch Kapitel 8) wurde besondere Aufmerksamkeit geschenkt. Dabei wird an Schusstagen auf Information ueber den Schussbetrieb und die automatische Beschaffung der Daten, an den anderen Tagen auf die Ueberwachung der Transaktionen der Datenbank besonderer Wert gelegt. Weitere, in Einzelheiten gehende, Informationen (z. B. Betriebszustand des Zentralrechners) werden nur auf besondere Anforderung (DIAL) auf einen weiteren Bildschirm geliefert. Diese Massnahme ist notwendig, um den Ueberblick ueber den Datenstrom in keinem Zeitpunkt zu verlieren.

7.2 Prozess PEEK

Er wird, wie alle anderen Prozesse im System COMET, durch Interrupts gesteuert. Hauptaufgaben sind:

- Darstellung des Datenstroms,
- Anzeige von Stoerungen,
- Darstellung wichtiger Meldungen,
- Beschaffen von Information zur Erkennung und Beseitigung von Stoerungen,
- Starten, Anhalten und Eingreifen in Prozesse der angeschlossenen VM,
- Aufzeichnen der Informationen der angeschlossenen VM im Logfile,
- Auswerten des Logfiles.

(a) Darstellung des Datenstroms. Jede der angeschlossenen VM des Systems, die VM der Benutzer gemeinsam, wird durch einen Bereich des Bildschirms der VM <exp> SPY repraesentiert. Ueber die Darstellung der Information, die von den angeschlossenen VM ueber VMCF geliefert wird, in den einzelnen Bereichen wird im naechsten Kapitel berichtet.

(b) Anzeige wichtiger Meldungen. Bei der Darstellung der Information auf dem Bildschirm der VM <exp> SPY blieben bisher Messages fast unbeachtet, da der Bildschirminhalt jede Sekunde erneuert wird. Gewisse Meldungen (z. B. VM Neustart) koennen aber grosse Bedeutung fuer das Ueberleben wichtiger Information haben. Im System COMET ist Sorge dafuer getragen, dass Messages erhalten bleiben. Zusaetzlich sollte der Bildschirm mit Alarmeinrichtung versehen sein.

(c) Beschaffen von Information. Solange der Datenstrom ohne Stoerung fliesst, wird weitere Information im allgemeinen wenig sinnvoll sein. Im Falle von Stoerungen, z. B.:

- naechster Schuss bleibt aus,
- Information wird nicht uebertragen,
- Fehler bei der Schuss-Uebertragung,
- Systemzusammenbruch,

kann umfassende Information ueber Art der Stoerung, Ursachen, Umgebung usw. von grosser Bedeutung sein. Gleichzeitig sollte die Uebersicht ueber den Datenfluss erhalten bleiben. Deshalb wurde die Moeglichkeit geschaffen, gezielte Information auf einem weiteren Bildschirm mit der DIAL-Funktion anzufordern. Sie kann, auch bei schnellem Wechsel, durch PF-Taste zum Lesen festgehalten werden, eine weitere PF-Taste erlaubt das Ausdrucken des Bildschirminhaltes auf dem Drucker.

(d) CC. Fuer Eingriffe und die Steuerung des Datenflusses verbleibt auf dem staendig angeschlossenen Bildschirm ein Kommandobereich. Er kann, ohne Verlust der auf dem Bildschirm dargestellten Information, wenn noetig, erweitert werden. Dazu wird u. U. die Information anderer Bereiche komprimiert.

Umfassende Information ueber Anwendung und gezielten Einsatz der CC steht ueber PF-Taste (als HELP-Funktion) bereit. Die vom Hersteller (bei HELP-Funktionen) zur Verfuegung gestellten Funktionen der PF-Tasten (DOWN, 1/2 DOWN usw.) werden fuer die Steuerung von Texten ebenfalls eingesetzt.

Nach den bisherigen Erfahrungen kann der Datenfluss durch diese Massnahmen auch in schwierigen Situationen aufrecht erhalten oder schnell wieder in Gang gesetzt werden. Der Neustart angeschlossener VM ist in wenigen Sekunden moeglich, jede Phase des Starts wird durch geeignete Information begleitet und ueberwacht. Ausserdem sind ausreichende Moeglichkeiten vorgesehen, um bei Stoerungen gezielt die notwendige Information zu erhalten. Weiterhin lassen sich Modalitaeten, wie Blockgroesse, Time-Out usw. dem Betriebsverhal-

ten des zentralen Rechners anpassen.

(e) Logfile. Die von den angeschlossenen VM uebermittelten Informationen ueber die Stationen des Datenflusses werden mit Angabe der benoetigten Zeit, Datenmenge, aufgetretene Fehler usw. im Logfile abgelegt. Es kann auf Anforderung abschnittsweise (Seiten, Zeilen) ausgedruckt werden, um die Uebersicht nicht zu verlieren. Das Logfile wird taeglich nach Ende des Schussbetriebs (durch Schuss "1") oder an schussfreien Tagen 17.00 Uhr auf Band ausgegeben.

(f) Auswertung des Logfiles. Ueberlegungen zur Auswertung des Logfiles wurden bisher nicht angestellt. Die Erfahrungen des Herstellers und die Literatur sollten dazu herangezogen werden.

7.3 Ausblick

Die Gestaltung der VM <exp> SPY kann derzeit noch nicht endgueltig festgelegt werden. Die bisher geplanten Eigenschaften werden beim Experiment ASDEX im laufenden Betrieb getestet und den gewonnenen Erfahrungen angepasst. Zu gegebener Zeit wird ueber die Entwicklung berichtet. Gewisse Einschraenkungen koennen u. U. wegen des Zeitbedarfs notwendig werden. Andrerseits sind wesentliche Funktionen durch Nutzung der gegebenen Moeglichkeiten und durch Weglassen unnoetiger Schnoerkel so beschleunigt worden, dass Erweiterungen im Informationsangebot moeglich sind. Eine Simulation des vorgeschlagenen Systems scheint uns zu aufwendig und zeitraubend. Einerseits liegt nicht genuegend Erfahrung vor, um eine geeignete Simulation des geplanten Netzwerkes zu entwerfen, andrerseits hoffen wir, die gewonnene Zeit fuer die Entwicklung des Systems COMET gewinnbringend verwenden zu koennen.

8. Layout

8.1 Vorwort

Die in den vorhergehenden Kapiteln vorgestellte dezentrale Organisation der Experiment-Maschinen RCV, DAT, MIG, USER erfordert bei den gegebenen Rahmenbedingungen (grosse Datenmengen, hohe Uebertragungsgeschwindigkeiten, schnelle Zugriffsmethoden), insbesondere im Hinblick auf den erweiterten Datenfluss zukuenftiger Experimente, hohe Zuverlaessigkeit und Stabilitaet

- des Betriebssystems
- der Betriebsprogramme und
- der Hardware,

um das Zusammenwirken und das Betriebsverhalten der Experiment-Maschinen optimieren zu koennen.

Die relationale Struktur des Netzes der virtuellen <exp> VM (siehe "Datenpfad") erfordert aus diesem Grunde Moeglichkeiten, durch "externe Einfluesse", wie den Betriebszustand des Zentralrechners, bedingte Engpaesse und Fehlverhalten festzustellen, Fehlerquellen einzukreisen, und entsprechende Gegenmassnahmen einzuleiten. Diese Notwendigkeit fuehrte zu dem vorgestellten Informationsfluss. Wie dazu bereits bemerkt wurde kann wegen der (zeitlichen) Minimalisierung der Betriebsprogramme, selbst bei bestmoeglicher Hardware-Unterstuetzung (z.B. schnelle Graphik), und auch aus Gruenden der Uebersichtlichkeit, nur mehr gezielt Information ueber ganze Verarbeitungsabschnitte aufgezeichnet werden. Um dieses Mindestmass an Kontrollinformation gewaehrleisten zu koennen muss bei der hohen Interrupt-Belastung (I/O, VMCF) der SPY- Maschine auf leistungsfaehige graphische Hardware zurueckgegriffen werden. Unter den gegebenen Vorraussetzungen (Benutzung von CMS) bieten die Modelle 3278.x von IBM bei Anschluss an geeignete Controller (IBM 3274-1B bzw. -1D mit ECSA = Extended Character Set Adapter) ein verhaeltnismaessig breites Spektrum hardwaremaessig unterstuetzter Funktionen. Es umfasst im einzelnen:

Extended Highlighting

- Intensified Display (2 Kontraststufen)
- Blink Data (Blinkrate 2Hz)
- Reverse Video
- Underscore

Programmed Symbols

- Definition von max. 6 Zeichensaetzen zu je 190 Zeichen (Spezifikation durch Angabe der Matrix (9x16))

Alternate Screen Size

- wahlweise 1920 oder 3564 Bytes bei Modell 5

Program Function Keys

- wahlweise 12 oder 24

Selector Light Pen

Die genannten Funktionen werden softwaremaessig initialisiert und sind ueber die Verwendung geeigneter Channel- Commands bzw. Instream -Orders einfach zu implementieren /6/ Insbesondere ist die Belegung (Programmierung) der PF -Keys durch den Benutzer frei waehlbar. Sie koennen im Zusammenhang mit geeigneten READ -Operationen (READ MODIFIED) /6,8/ zur Minimierung des informationsflusses (Max. 3 Bytes) 3278 - Controller - Channel - Memory verwendet werden, und tragen dadurch im wesentlichen zur Entlastung der Controller bzw. Kanaele sowie zur Senkung der I/O Wartezeiten bei. Die einfache Adaptierung in CMS ueber die DIAGNOSE - Instruktion ermoeglichte eine schnelle Implementation und eine umfangreiche Testphase. Im Vergleich zu FORTRAN I/O Routinen wurden mit den neu entwickelten Standard - Routinen (RDCON, WRCON) folgende Zeiten gemessen:

	WRITE 1 PAGE		SHORT READ	
	CPU (us)	ELA (us)	CPU (us)	ELA (us)
FORTRAN	6037	24318	--nicht moeglich--	
RDCON WRCON	55	623	18	501

Elapsed Times bei einer VM mit Prioritaet 99 und einem CPU-Anteil von 10 %

Fig.34 Vergleich FORTRAN I/O - ASSEMBLER I/O

Die im Zusammenhang mit der vorgestellten Hardware erreichbaren und vertretbaren Moeglichkeiten erstrecken sich ueber:

- FORM FILL
- FULL SCREEN DISPLAY
- FULL SCREEN EDIT

- SCHNELLE GRAPHIK (+ gute Aufloesung, ca. 300000 Punkte)
- STRUCTURED DISPLAY

Nicht zuletzt ermoeglicht die damit verbundene Minimalisierung der I/O Operationen erst die Gestaltung des zentralen COMET- Bildschirms. Das Limit der dabei in der Zeiteinheit durchfuehrbaren Operationen (auf Bildschirm) errechnet sich aus Faktoren wie:

- Kapazitaet des Controllers
- Kapazitaet des Kanals
- mittlere Belastung des Prozessors
- Geschwindigkeit des Terminals
- I/O Belastung des Gesamtsystems
- Anzahl der Bildschirme je Controller
- Anzahl der Controller je Kanal
- Anzahl der aktuellen Benutzer mit interaktiver Aktivitaet
- Interrupt-Belastung der SPY-Maschine
- Deadline-Prioritaet im Dispatch-Prozess
- Prioritaet der virtuellen Maschine
- Menge der pro Operation transferierten Daten

Im Hinblick auf diese Eckdaten (siehe Fig.35) muss die Frequenz der zu erwartenden Bildschirmoperationen geplant werden.

Controller	450000 Bytes/sec (READ) / 600000 (WRITE)
Channel	ca. 1 - 1,2 MB / sec
Belastung Prozessor	typisch ca. 90 % bei 2-5 Paging - Operationen / sec
I/O Belastung	typisch ca. 80-200 I/O Operationen / sec
Bildschirme	32 je Controller
Interrupts SPY	nicht vorhersehbar

Fig.35 Rahmendaten fuer die Planung der I/O Operationen

Eventuell kann diese aber durch geeignete Hardware- Unterstuetzung (z.B. zusaetzliche Kanaele oder Kanaele mit queuing capacity) gesteigert werden. Im vorgegebenen Rahmen kann mit Ruecksicht auf die

- Uebersichtlichkeit
- bessere Ausnutzung von Subsystemen (Subchannel availability)
- verbesserte interaktive Charakteristik (nach dem Prinzip konkurrierender Prozesse)

bei Verwendung von 3 zentralen Bildschirmen bereits die volle Betriebssicherheit und Uebersicht ueber das Gesamtsystem gewaehrleistet werden. Da CMS die Definition zusaetzlicher virtueller Konsolen erlaubt (CP Command DEFINE) mussten zu diesem Zweck die Standard I/O Routinen (RDCON, WRCON) nur um den Parameter der virtuellen Adresse erweitert werden. Eine Konsole wird dabei jeweils beim Start der VM <exp> SPY, die anderen beiden mittels des Commands CP DIAL initialisiert. Den Informationsfluss sowohl waehrend der Initialisierung als auch waehrend I/O Taetigkeit zeigt Fig.36:

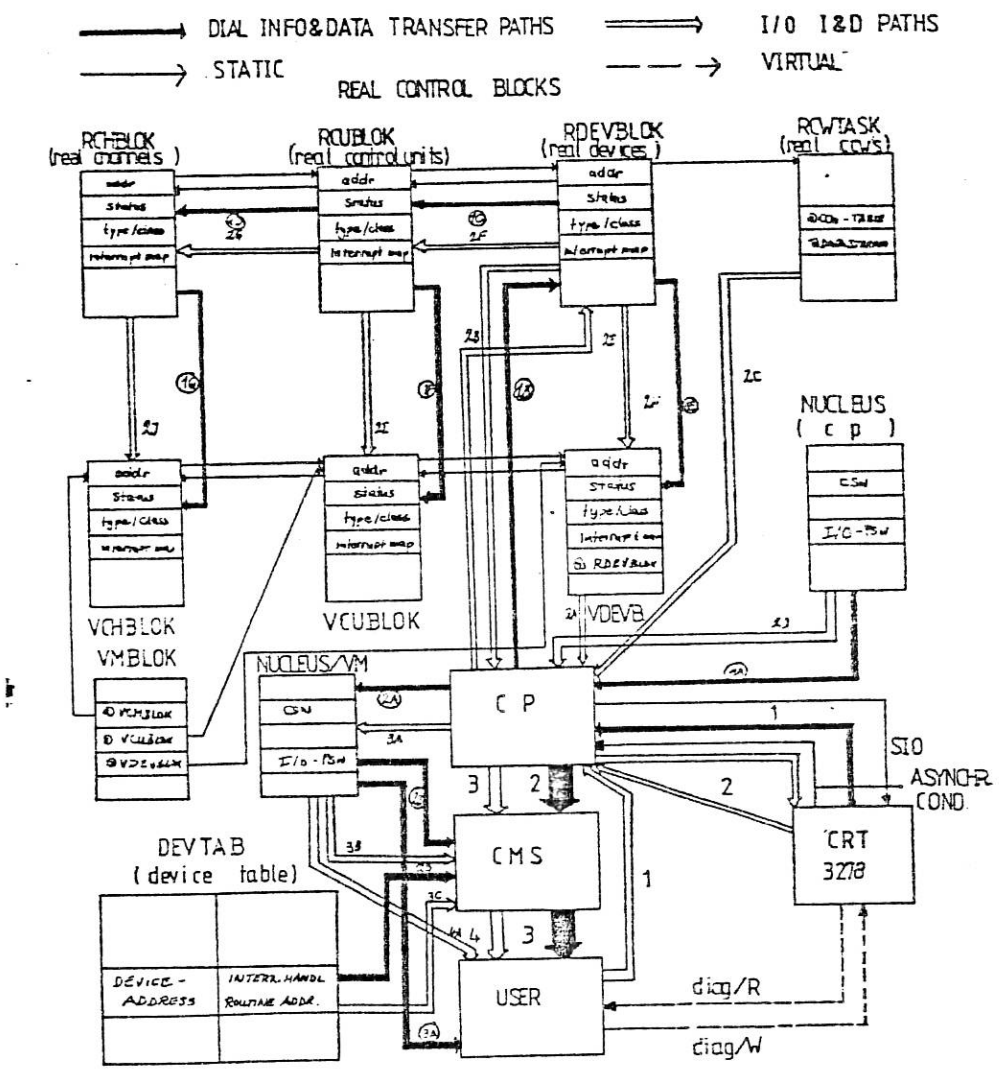


FIG 36 : DIAL / IO - INFORMATIONSFLUSS

8.2 Darstellung des Informationsflusses

Bei Verwendung von 3 zentralen Konsolen wird folgende Strukturierung der Datenstroeme (graphische Information) unterlegt:

1. SPY-System-Operator-Communication-Area (SPYCOM)

fuer

- Eingabe von zentralen Kommandos
- Beantwortung von QUERYs abhaengiger VMs
- Uebersicht ueber den Betriebszustand des Systems (d.h. des Netzes der VM RCV, DAT, MIG, SPY und deren externen Geraeten)
- detaillierte Information ueber den Betriebszustand des Zentralrechners (PEEK- z.B. Auslastung CPU und STORAGE, PAGING,I/O, etc.)

2. SPY-System-Status-Display

zur Darstellung aller Informationen ueber Informationsfluss, Status und externer Geraete der VM RCV, MIG, DAT. Im einzelnen beinhaltet dies:

- Zustand der VM (OFFLINE, WAIT,...)
- Zustand externer Geraete (TAPE, CHANNEL, INTERDATA, ...)
- Zustand interner Queues (BACKUP, MIGRATE, LOCATE, ...)
- Fehler in Verarbeitungsabschnitten (Uebertragung, Konversion,...)
- Statistik ueber Verarbeitungsabscnitte (Zeiten, Datenmengen,...)

3. SPY-Test-Output-Display

zum Display von Testausdrucken, umfangreichen Statistiken und Antworten auf zentrale Kommandos. Da dieser Bildschirm nur im Testfall benoetigt wird, kann auf ihm auch die SPY - PEEK Version laufen.

Zur Vereinfachung der Start und Restart - Prozeduren wird SPYCOM permanent der System - Konsole der VM <exp> SPY (Adresse 01F) zugeordnet. Die Initialisierung erfolgt somit entweder durch die Logon oder durch die Reconnect - Prozedur. SPYSTAT und SPYTEST koennen hingegen nach den Bemerkungen von 9.1 in einfacher Weise als DIAL - Konsolen implemen-

tiert werden. Die Initialisierung erfolgt dann als zentrales Kommando (einerseits durch Zuordnung einer virtuellen Adresse, andererseits durch den DIAL - IN Prozess). Die vorgenommene Strukturierung des Informationsflusses wird gerechtfertigt durch:

- bessere Ausnuetzung von I/O Wartezeiten
- bessere Uebersicht
- die Forderung nach zentraler Bedienung des Rechnernetzes

Um die Funktionsfaehigkeit der zentralen Steuerung einzusehen hat man dazu nur folgendes zu beachten: Die VMCF - Uebertragung ist voll transparent und gestattet mithin, bei geeigneter Auslegung des Protokoll - Doppelwortes, die Gestaltung eines High - Level - Protokolls, wie dies in virtuellen Rechnernetzen benoetigt wird. Somit koennen auch Kommandos, Modules, EXEC - Prozeduren, etc. in jeder beliebigen VM initialisiert werden, solange nur ein fest vorgegebenes Protokoll zwischen den teilhabenden VM vereinbart ist. Der Ueberwacher an der zentralen SPY - Konsole (SPYCOM) kann also, wie jeder CMS - Benutzer von der System - Konsole fuer seine Maschine, zentral CMS und CP - Kommandos fuer beliebige untergeordnete virtuelle Maschinen abgeben, ohne Einschraenkungen in der Maechtigkeit des Befehlsvorrates und ohne dass er die Simulation bemerkt. Im einzelnen bedeutet dies die Verfuegbarkeit bewaehrter Testhilfen (CP-TRACE, CMS-DEBUG), die Ermoeglichung der Kommunikation mit verteilten Prozessen (ueber QUERY oder zentrales Kommando), sowie direkte Einsicht in den Status verschiedener Prozesse.

Welche ausgewaehlte Information daraus auf den logischen Einheiten SPYCOM und SPYSTAT zur Verfuegung gestellt wird ist im Einzelfall der jeweiligen Betriebssituation anzupassen. Dazu unterscheidet man:

- Normalbetrieb
- Testbetrieb
- Recovery, Restart
- Central Command Modus

Im Hinblick auf einen "ruhigen" Bildschirm (max. 1-2 Operationen / sec.) und gute Uebersicht ist sowohl eine weitere logische Strukturierung des Datenflusses (SCREEN SPLITTING), sowie die Konservierung ausgewaehlter graphischer Information (SCREEN SAVE) notwendig. Zusaetzlich muss im Bedarfsfall, wegen der oft sehr hohen Geschwindigkeit der Transaktionen, auf die Wiedergabe detaillierter Information verzichtet werden. Stattdessen koennen sinnvoll nur der Beginn, das Ende und statistische Daten ueber groessere Verarbeitungsabschnitte aufgezeichnet werden. In diesem Rahmen kann nun ohne Ruecksicht auf Recovery, Restart und Testmodus die Bildschirmgestaltung im Normalbetrieb geplant werden. Einziger Anhaltspunkt fuer eine logische Strukturierung ist dabei die Identifikation der beteiligten VM : RCV, DAT, MIG. Nach

umfangreichen Zeitmessungen und Erfahrungen ueber die Stabilitaet der Ausfuehrungszeiten (siehe Anhang) kann aus den Forderungen der entsprechenden Kapitel leicht die aufzuzeichnende Information entnommen werden:

VM <exp> MIG:

Wegen der (problembedingt) langsameren Transaktionen im Bandbetrieb kann, solange noch kein Anschluss an schnelleren Massenspeicher vorhanden ist, detaillierte Information ueber

- die verwendete Bandeinheit (Zuordnung der realen Einheit, Aufforderung und Auflegen des Bandes)
- die verschiedenen Queues (Backup, Migrate, Anzahl Requests, Anzahl angeforderter Schuesse)
- den bearbeiteten Request (Backup, Migrate, Bandnummer, Filenr., Schussnummer, Laenge)
- Fehlersituationen (Fehler beim Lesen/Schreiben auf BAND, falsches Label)

wiedergegeben werden. Darueber hinaus wird der jeweilige Status der VM - MIG angezeigt. Man unterscheidet:

- OFFLINE <logoff oder VMCF nicht autorisiert>
- ONLINE <Anfangszustand vor Start der Betriebsprogramme>
- IDLE_WAIT <Warte-Zustand ohne Requests>
- TAPE_WAIT <Warte-Zustand bedingt durch Bandeinheit>
- BACKUP <fuehrt BACKUP - Request aus>
- MIGRATE <fuehrt MIGRATE- Request aus>

Der Start und das Anhalten der Betriebsprogramme, wird wahlweise manuell oder durch zentrales Kommando (bevorzugt) durchgefuehrt. Die entsprechenden Phasen werden durch Zeit, Ursache und Fehlernachrichten dokumentiert.

VM <exp> RCV:

Wie in Kapitel 4 erlaeutert, verweilt der Prozess PETER ausschliesslich Interrupt - gesteuert in der WAIT - Phase. Wegen der schnellen Kanal- bzw. VMCF- Transaktionen kann dabei nicht jede Transition der WAIT in die READ- bzw. SEND - Phase aufgezeichnet werden. Vielmehr muss man sich darauf beschraenken den Wechsel zwischen zwei Verarbeitungsphasen (etwa Zusammenstellung einer Diagnostik oder groessere Bloecke) unter Angabe des Eintritts, der Beendigung, der Anzahl gelesener und uebertragener Bloecke , und der aufgetretenen Fehler, zu dokumentieren. WAIT wird demnach erst bei groesserer IDLE - Zeit (TIMEOUT einige wenige Sekunden) wieder angezeigt. Zusaetzlich wird staendig der Status der Einheit 038 (INTERDATA) angezeigt. Die Statusanzeige beinhaltet

mithin:

- OFFLINE
- ONLINE
- IDLE_WAIT
- 038_WAIT <Warte-Zustand bedingt durch Einheit 038>
- PETER

Bei Exit durch "Schuss 1" oder zentrales Kommando wird zusaetzlich die Ausgabe des aktuellen Schussprotokolls bestaetigt.

VM <exp> DAT:

Aehnlich wie bereits bei der VM <exp> RCV kann wegen der kurzen STATE - TRANSITION - Zeiten der Prozess ENQUEUE nicht, und der Prozess WAIT nur bei laengerer IDLE-Zeit aufgezeichnet werden. Waehrend des laufenden Schussbetriebs werden daher folgende Abschnitte dokumentiert:

- LOCATE (jeder Request)
- AUTOLOCATE (als Statistik)
- COPYFILE (als Statistik)
- Ablage Schussfiles durch BACKUP (als Statistik)

Nach Beendigung des Schussbetriebs muss zusaetzlich der Eintritt in die MIGRATE - Phase mit Angaben ueber MIGRATE- Intervall etc., bestaetigt werden. Im einzelnen wird dadurch folgende Information verfuegbar gemacht:

- Beginn des Schussintervalls (mit Zeit)
- Ende des Schussintervalls (mit Zeit)
- fehlende Schuesse im Intervall (Nummern)
- Erster / letzter Block des laufenden Schussfiles (COPYFILE)
- Erste/letzte Diagnostik des laufenden Schussfiles (AUTOLOCATE)
- Locate Request (mit Schussnummer und Zeit)
- Anzahl Locate Requests in der Queue

Exit und Startphase werden durch die VM <exp> SPY durch zentrales Kommando eingeleitet. Mit der vorgestellten Organisation ergibt sich eine Bildschirmorganisation, wie sie etwa Fig. 37 zeigt:

```
<===> System COMET <===> SPYSTAT <===> CONNECTED AT 08:03:09 04/04/83 <=======>
ASDMIG --- Status = WAIT          UNIT 181 attached to 275 at 09:05:50 NOT READY
TAPE   --- M118AZ   NOT MOUNTED  Req = BACKUP Shot = 37369 File = 152 by ASDDAT
QUEUES --- Backup =    8 REQ FOR   22 FILES    Migrate =    0 REQ FOR   0 FILES
ERRORS ---
+-----------------+-----------------+-------------------+
| START (08:38:55) | BACKUP (09:06:55) | TP_WAIT (09:06:55) |
+-----------------+-----------------+-------------------+
ASDRCV --- Status = WAIT          UNIT 038 attached to 080 at 09:02:38     READY
SHOT   --- 37528  First Block transm : 09:32:58  Last Block transm : 09:33:58
ERRORS --- CVM038-DID181
+-----------------+---------------------+
| START (08:04:38) | IDLE_WAIT (09:33:59) |
+-----------------+---------------------+
ASDDAT --- Status = AUTOLOCATE              Shot = 37528
RCV    --- First block received: 09:32:58  Last block received : 09:33:58
SND    --- First diag: 09:33:01   Did 100  Last diag:
QUEUES --- LOCATE =    22 REQ FOR    22 FILES
SHOTS  --- First Shot = 37523  09:14:54    Last Shot = 37527  09:28:56
MISSING SHOTS : 37525,37526
ERRORS ---
+-----------------+---------------------+
| START (08:04:02) | AUTOLOCATE (09:32:58) |
+-----------------+---------------------+
```

Fig.37 : Organisation des SPYSTAT

Im Testbetrieb ergeben sich dazu keine Aenderungen, da die Ausgabe der TEST - Ausdrucke mit Uhrzeit / Datum / VM-ID / Testunit auf SPYCOM oder SPYTEST erfolgt, und in Form und Inhalt von der ausfuehrenden Testunit bestimmt wird. Im Falle des RECOVERY, das automatisch eingeleitet werden soll, und des Checkpoint- RESTARTS, per zentralem Kommando, ist die Bildschirmgestaltung den veraenderten Gegebenheiten anzupassen. Das Recovery umfasst im einzelnen bei:

VM <exp> MIG

- Wiederbeschaffung der BACKUP und MIGRATE Queues (von DISK)
- Wiederbeschaffung der SPOOL - Files (durch Betriebssystem)
- Wiederbeschaffung der Master-File-Directory (MFD) von VM <exp> DAT

VM <exp> DAT

- Wiederbeschaffung der LOCATE -Queue (von DISK)
- Bestimmung der aktuellen Schussnummer (von RCV)
- Wiederbeschaffung des aktuellen Schusse (von RCV)
- Wiederbeschaffung fehlender Bloecke (von RCV)
- Wiederbeschaffung der Master-File-Directory (MFD) von zusaetzlicher CMS-Maschine

VM <exp> RCV

- Wiederbeschaffung fehlender Bloecke (von Experiment)
- Wiederbeschaffung des aktuellen Schussfiles (von Experiment)

Fuer naehere Einzelheiten wird auf den COMET System Programmer's Guide /20/ verwiesen.

Nach konkreten Fehlersituationen, oder nach einem Systemzusammenbruch im Zentralrechner soll dagegen per zentralem Kommando die Moeglichkeit gegeben sein, an bestimmten Checkpoints fortzusetzen. Eventuell benoetigte Startdaten koennen von dem betreffenden Prozess per QUERY ueber die Operator - Konsole verlangt werden. Moeglichkeiten geeignete Checkpoints anzulegen ergeben sich aus den Ablaufschemata der betreffenden Prozesse. Als Beispiele seien hier genannt:

VM <exp> MIG

- TAPE- Reset (REWIND)
- System - Reset (loeschen der Queues, WAIT status)
- Kopieren der Spoolfiles in die Queue
- Retten der Queues
- Start des Queue-Schedulers

VM <exp> DAT

- Retten der Master-File-Directory
- Start des letzten Requests
- Start der AUTOLOCATE - Prozedur
- Start der LOCATE - Prozedur
- System-Reset
- Retten der Queue

VM <exp> RCV

- 038 - Reset
- Start der Lese Prozedur
- System - Reset

Fuer naehere Einzelheiten, die Status der Maschinen, Startdaten, sowie eine vollstaendige Beschreibung aller Checkpoints betreffen, wird auf den COMET System Programmer's Guide /20/ verwiesen.

8.3 Update des GALE - Namelist - Files

Im Vorwort (8.1) wurde auf die Moeglichkeiten und Vorteile eines leistungsfaehigen graphischen Subsystems hingewiesen. Als naheliegende Anwendungen boten sich Full Screen Operationen (EDIT oder DISPLAY). Die Erfahrung (der Autoren) mit wichtigen (haeufig benutzten) Full Screen Komponenten im CMS (XEDIT , HELP) zeigt, dass die Ausnutzung der gegebenen

graphischen Moeglichkeiten grosse Vorteile in Bezug auf

- Benutzerfreundlichkeit
- Bedienunsgskomfort
- Performance (durch Senkung des I/O Aufkommens) hat.

Gleichwertige und vor allem geeignete Produkte stehen, obwohl wuenschenswert, fuer das Update der GALE - Namelist - Files nicht zur Verfuegung. Vorhandene Komponenten wie XEDIT sind wegen einschraenkender Faktoren, wie:

- Formatierung (des Files aus Gruenden der Lesbarkeit)
- Beschraenkung der Record und Filegroessen (aus Gruenden der Uebersichtlickeit)
- ungeeignete Suchverfahren
- relativ komlizierte Bedienung

nicht geeignet. Im Gegensatz dazu bietet eine dem vorgegebenen Rahmen (einfache Spezifikation von Diagnostik, Geraet, Variablen und Einstelldaten) angepasste Full Screen Facility bessere Moeglichkeiten:

- voellige Freiheit der physikalischen Datenorganisation
- Implementation effizienterer, fuer die Problemstellung geeigneterer Suchverfahren
- formatfreie Eingabe, etc.

Seitdem CP Full Screen Operationen auch fuer andere Bildschirme unterstuetzt, kann anstelle des IBM 3278 auch graphische Hardware anderer Hersteller (TEKTRONIX, u.a.) verwendet werden.

8.4 Neugestaltung des Prozesses PEEK

In einem System, das vom Konzept her von Zeit zu Zeit die Kapazitaet des Zentralrechners auslastet, besteht in natuerlicher Weise der Bedarf, zu beliebiger Zeit den Betriebszustand des Zentralrechners zu beobachten, um Engpaesse und Fehlverhalten der Betriebsprogramme zu erkennen, oder gar Ansatzpunkte fuer weitere Optimierungen zu finden. Die vorhandenen Programm- Produkte (PEEK, AUTOMON) sollten dazu jedoch, wegen verschiedener Nachteile, nicht verwendet werden:

PEEK

- ungenuegende Detailinformation
- zu starke I/O Belastung

AUTOMON

- zu unruhiger Bildschirm

- ungeeignete Darstellung (zu wenig aufgezeichnete Maschinen)

Beide Programme entnehmen ihre Information der VM SYSMON der als zentraler Monitor Informationen ueber Ereignisse im Betrieb sammelt, und von jeder beliebigen VM ueber VMCF angesprochen werden kann. Die von SYSMON zur Verfuegung gestellte Information umfasst (fuer jeden aktiven Benutzer):

- CPU Anteil (dazu prozentual Anteil Benutzer und Betriebssystem)
- Paging Rate
- Anzahl I/O Operationen
- Aktuelle Arbeitsgroesse der VM (in PAGES)
- Anzahl im Speicher gehaltener PAGES
- Prioritaetsstufe des Benutzers
- Queue des laufenden Prozesses (Q1,Q2,Q3)
- Status der virtuellen Maschine (RUNABLE, ECMODE, BCMODE, FAVOURED, IOWAIT, IDLEWAIT, TRC, INSTWAIT,...)
- gegenwaertige Maximalgroesse der VM in K Bytes

Die beschriebenen Daten sind Mittelwerte ueber ein fest vorgegebenes Monitor - Intervall. Das Sammeln von Informationen ueber bestimmte reale Ereignisse im Zentralrechner, wird vom Basis - Betriebssystem (VM 370) unterstuetzt, und ist System - Programmierern ueber das CP Command MONITOR zugaengich. Die zugrunde liegenden Assembler - Monitor Call (MC) Operationen /8,9/ sind fest in die Routinen des Betriebssystems (VM/SP) integriert. Das Sammeln der Daten geschieht dabei auf zwei Arten:

- direkt per Monitor Call (MC) oder
- periodisch (Timer Interrupt)

In VM/SP sind 8 verschiedene Klassen des MC-Aufrufs implementiert. Man unterscheidet:

- PERFORM (Timer) --- Auslastung der System-Resourcen
- RESPONSE (MC) --- Terminal I/O - Antwortzeiten
- SCHEDULE (MC) --- Queue - Bearbeitung - Resourcen- Verteilungs - Strategie
- USER (Timer) --- Benutzer - Profile der Resourcen- Auslastung
- INSTSIM (MC) --- Simulation privilegierter Operationen - CP Ueberhang
- DASTAP (Timer) --- DASD und Tape Aktivitaet

- SEEKS (Timer) --- DASD I/O Requests - Armbewegungen

- SYSPROF (MC) --- Profil der Resourcen - Auslastung

Die Interpretation und Reduktion der anfallenden Daten kann im wesentlichen automatisch durch System-Routinen (VM/370: Performance / Monitor Analysis Program /6/) geschehen.

Die anfordernde VM erhaelt die Information von SYSMON als Print - File in den virtuellen Card - Reader, das bereits in der vorstehend beschriebenen Weise formatiert ist. Fuer die weitere Nutzung kann daher das gegebene Format, ohne Konversion, direkt weiter verwendet werden. Hingegen wird im Normalfall nicht die Information ueber jeden Benutzer benoetigt, es interessieren im allgemeinen nur

- die Daten der VM MIG, DAT, SPY, RCV

- sowie die Gesamtbelastung des Systems

Zusaetzlich definieren wir folgende Anforderungen:

- moegliche Aenderung des Monitorintervalls

- festhalten (konservieren) eines angezeigten Status

- Display aller oder ausgewaehlter virtueller Maschinen

- Erstellen von Hardcopys eines angezeigten Status

- Setzen verschiedener High-Water-Marks, bei deren Ueberschreitung die Daten der betreffenden virtuellen Maschinen eingeblendet werden

- automatische Inbetriebnahme des Prozesses bei Ueberschreitung einer High - Water - Mark (mit ggf. verschiedenen Intervallen und Schusssynchron)

Fig. 38 zeigt einen Vorschlag fuer die Neugestaltung, Prozess CMON:

```
<===> System CONEZ <===> CMON <===> CONNECTED AT 08:08:12 04/04/83 <==========>
Interval =  10 SEC  Modus =  EXP  Hold =  NONE  Reason =  USER  Time = 09:14:06

USERID-> %CPU %CP %USR ISEC PSEC WSS RES USEC DRUM PRI VMSIZE Q EXCTN-STATUS
TOTAL     84  24   60  148   11

ASDMIG   .99 .99  .00   12   .0  49  51   .0    0  90  2048K 3 CMS,RUNABLE
ASDDAT   IDLE.00  .00   .0   .0  66  66   .0    0  10  5120K . CMS,IDLEWAIT
ASDRCV   IDLE.00  .00   .0   .0  33   0   .0    0  10  4096K . CMS,IDLEWAIT
ASDSPY   IDLE.00  .00   .0   .0  27   0   .0    0  99  2048K . CMS,INSTWAIT

WSADAT   IDLE.00  .00   .0   .0 114 114   .0    0  10  2048K . CMS,IDLEWAIT
WSARCV    .02 .02  .00   .0   .0  75  75   .0    0  10  1024K . CMS,IDLEWAIT
WSASPY    10   1    9   .0  1.2  53  53    17   0  99  1024K . CMS,IDLEWAIT

MARKS     30  10   20   80   10  REACHED BY THE FOLLOWING MACHINE(S)

MVS       43   3   40   25  6.7 1076 076   .0    0  99  8192K 3 ECM,RUNABLE

PF 1 = EXIT                 PF 2 = CHANGE TIMER      PF 3 = CHANGE MODUS
PF 4 = CHANGE REASON        PF 5 = HOLD              PF 6 = RESUME
PF 7 = PRINT                PF 8 = FORWARD           PF 9 = FORWARD 1/2
PF10 = BACKWARD             PF11 = BACKWARD 1/2      PF12 = TOP
```

Fig.38 : Bildschirmgestaltung beim Prozess CMON

9. Abschliessende Bemerkungen

9.1 Benutzer-Software

Es wurden viele Anstrengungen gemacht, alle Aktionen des Systems zu beschleunigen und die Responsezeit spuerbar zu verbessern. Der Anteil des Systems an der CPU-Zeit fuer Datenverwaltung und Transaktionen ist -normalen Auswertebetrieb vorausgesetzt- wie in Abschnitt 9.3 gezeigt wird, nur gering. Das aktuelle System EDDAR wurde teilweise schon in Richtung auf das System COMET weiterentwickelt, die Wartezeit auf die Experimentdaten, soweit nicht von der Zeit fuer die Erfassung abhaengig, wesentlich verkuerzt. Ausschlaggebend fuer die Gestaltung des Rechenbetriebes sind daher im wesentlichen die Benutzer-Programme, die teilweise die gewonnene Zeit wieder aufbrauchen. Ihrer Optimierung sollte daher besondere Aufmerksamkeit geschenkt werden.

Beispiel: Plotten von Rohsignalen

Der Mittelwert von n Signalen (n als Zweier-Potenz angenommen) wird durch 10 Operationen gebildet (Zaehler loeschen, Division durch n, Berechnung der Punktkoordinaten, Real--Integer-Konversion). Der als Integer eingelesene Messwert wird vorher mit 5*n Operationen aufbereitet (Integer-Real--Konversion und Addition). Ohne die zweifache Konversion hat man fuer den Mittelwert von n Signalen 8 Operationen (Zaehler loeschen, Shift und Ermittlung der Bildpunkt-Koordinaten), je Signal erfolgt nur eine Addition. Daher stehen 5*n+10 Operationen nur n+8 Operationen gegenueber. Fuer das Plotten von 128K Signalen und n=256 wird daher

- 1,101 sec bei zweimaligem Konvertieren
- 0,225 sec ohne Konversion

an CPU-Zeit gebraucht. Bei den gegebenen Voraussetzungen kann man so eine Verminderung der CPU-Zeit je Versuch um bis zu 100 sec erhalten, da das Erstellen von Plots bei der Schnellauswertung bevorzugt wird.

In der Praxis zeigt sich bei vielen anderen Problemen, dass durch geeignete

- Programmplanung und -organisation,
- Auswahl und Programmierung der Verfahren
- Datenorganisation

zum Teil eklatante Einsparungen moeglich werden.

Diese Situation kann durch eine umfassende Bibliothek opti-

mierter Auswerte-Routinen unterstuetzt werden. Besondere Bedeutung erhaelt diese Massnahme durch die stetig wachsenden Datenmengen. Die derzeitige Gestaltung der Auswerte-Programme wird ohne diese Massnahmen mit Sicherheit neue Engpaesse hervorrufen.

9.2 Entwicklung des Systems

Im Abschnitt 2.3 wurde erwaehnt, dass die Entwicklung des Systems COMET aus dem System EDDAR ohne Umstaende zu bewaeltigen ist. Auch kann jede VM im Zusammenhang mit dem von Hertweck /3/ vorgestellten Auswerte-System als Prozess eingefuehrt werden.

Einige der vorgestellten Einzelheiten wurden inzwischen im System EDDAR realisiert und sind erprobt. Eine schrittweise Erweiterung stoert den laufenden Schussbetrieb nicht, jeder Schritt kann beim Experiment ASDEX getestet und eingefuehrt werden. Damit kann bei Beginn des Experiments W VII AS bereits ein erprobtes System, das mit Sicherheit grosse Datenmengen bewaeltigt, zur Verfuegung gestellt werden.

Umfangreiche Auswertungen sollten -ggf. nach Vorbereitung im Dialog- im Batch-Betrieb durchgefuehrt werden. Dazu ist das CMS-Batch-System bei der geplanten Organisation ehestens geeignet. Sollte jedoch ein Anschluss an AMOS/2 moeglich sein, um auch den Massenspeicher zu integrieren, dann stuende, solange vorhanden, MVS zur Verfuegung.

9.3 Abschaetzungen

Die diesem Abschnitt zu Grunde gelegten Leistungen wurden aus Protokollen ueber den Schussbetrieb des vergangenen Jahres (*1) oder aus neueren Messungen (*2, s. Anhang A-4) im Normalbetrieb mit unterschiedlichen Belastungen gemittelt. Dabei wurden Fremdleistungen der CPU mit 50 ... 80 % -durch die VM AMOS, MVS, CRAYFE und der Benutzer des Experiments W VII a waehrend des Schussbetriebs- festgestellt. Beim Lesen vom Kanal (*3) wurden die bisher gemessenen Zeiten eingesetzt, die Blockgroesse jedoch auf 4096 Bytes erhoeht.

Fuer die Abschaetzung der erforderlichen Rechnerleistung wurden die folgenden Voraussetzungen zu Grunde gelegt:

- 6 MBytes Experimentdaten je Versuch, davon 4 MBytes zur Schnell-Auswertung zwischen zwei Versuchen, 2.5 MBytes zur regelmaessigen Serien-Auswertung ,

- beim Schussbetrieb 50 % der CPU-Leistung fuer Experiment W VII a Serien-Auswertung, bei Serien-Auswertung 30 % CPU-Leistung fuer Experiment W VII a Schussbetrieb,

- 60 Operationen je Byte Experimentdaten im Mittel fuer die Schnell-Auswertung, 300 Operationen im Mittel fuer die Serien-Auswertung, 6 MIPS-Rechner,

- 40 Benutzer bei der Schnell-Auswertung je 4 VMCF-Re-

quests, davon je 1 Request mit 100 bzw. 50 KBytes aus dem Kernspeicher und 2 Requests mit je 50 KBytes aus der Datenbank,

- 20 Benutzer bei der Serien-Auswertung, je Benutzer 8 VMCF-Requests mit insgesamt 400 KBytes, im Mittel aus 40 Files.

(1) Abschaetzung der Schnell-Auswertung

VM <exp> RCV	CPU (msec)
Lesen vom Kanal (*3) 1536 Bloecke * 4 KB	3806,21
Konversion (*1), Entmultiplexen Datenstroeme und Kanaele (4 KB)	2371,58
Senden (*1,*2) 48 mal 128 KBytes	8,70
Abspeichern zum Checkpoint 48 * 128 KBytes	120,00
VM <exp> DAT	
Empfangen 48 * 128 KB	86,40
Abspeichern in Datenbank 48 * 128 Kbytes	120,00
160 Benutzer-Requests	6,24
40 * 100 KBytes aus Kernspeicher senden	160,00
40 * 50 KBytes aus Kernspeicher senden	80,00
80 * 50 KBytes aus Datenbank senden	488,00
Summe System	7247,13

Fig. 39 CPU-Leistung fuer die VM des Systems

Der zeitliche Zusammenhang ist in Fig. 32 dargestellt. Danach ist die Schnell-Auswertung eines Versuchs binnen 3,7 min abgeschlossen.

Die verlangten 80 I/O-Operationen werden derzeit auf IBM 3350 Direct Access Storage /4/ ausgefuehrt. Daraus folgt als mittlere Zeit fuer eine Armbewegung 17,5 msec und als mittlere Zugriffszeit 8,4 msec. Ferner ist zu beachten, wie oben

schon ausgefuehrt, dass die I/O-Operationen von der CPU nur angestossen werden, die Datenbeschaffung jedoch autonom von der Steuereinheit ausgefuehrt wird. Trotzdem muessen die

VM der Benutzer	CPU (sec)
160 Requests senden	0,12
160 * Daten empfangen	0,29
10 MBytes Daten verarbeiten	100,00
Summe Benutzer	100,41

Fig. 40 CPU-Leistung fuer die VM der Benutzer

Warte- und Lesezeiten ueberprueft werden. Ein Schussfile belegt ca. 10 Zylinder, daher sind -der Optimierung durch CP zufolge- maximal 18 Armbewegungen anzusetzen. Die mittlere Wartezeit tritt hoechstens 80 mal auf :

I/O-Operationen	CPU (sec)
18 Armbewegungen	0,32
80 mittl. Umlaufzeiten	0,67
Ausgabe 12 MBytes	10,00
Einlesen 4 MBytes	3,33
Summe I/O	14,32

Fig. 41 Wartezeiten auf I/O-Operationen

Der Vergleich der Summen System und Benutzer mit der Summe I/O erlaubt den Schluss, dass alle I/O-Operationen rechtzeitig abgeschlossen werden, ohne dass ein Benutzer-Programm auf deren Abschluss zu warten hat.

(2) Abschaetzung der Serien-Auswertung

Auch in diesem Falle zeigt der Vergleich der Summen System und Benutzer gegen die Summe der I/O-Aktivitaeten, dass die Auswerte-Programme nicht auf deren Beendigung warten muessen.

VM <exp> DAT	CPU (sec)
6400 Requ. empfangen	0,25
6400 * 50 KBytes lesen aus Datenbank	39,04
6400 * 50 KB senden	4,69
Summe System	43,98

Fig. 42 CPU-Leistung fuer die VM des Systems

VM der Benutzer	CPU (sec)
6400 Requests senden	4,69
6400 * 50 KBytes von VM <exp> DAT lesen	39,04
Verarbeitung 320 MB	16000,00
Summe Benutzer	16043,73

Fig. 43 CPU-Leistung fuer VM der Benutzer

Auch bei der Serien-Auswertung muss die Wartezeit auf Beendigung der Disk-Operationen untersucht werden:

I/O-Operationen	CPU (sec)
320 Armbewegungen	0,56
6400 mittl. Umlaufzeit	53,76
320 MBytes Daten von Datenbank lesen	266,00
Summe I/O	320,32

Fig. 44 Wartezeiten bei I/O-Operationen

In diesem Zusammenhang werden als weitere Moeglichkeiten zur Rationalisierung der Auswertung

- Ausruestung mit IBM 3380 Direct Access Storage /5/,
- Ausbau des Kernspeichers im zentralen Rechner auf 32 MByte,

- Fast Queuing fuer I/O-Operationen,
- Formatieren der Datenbank mit 4096 Bytes,
- Nutzung der Datenpfade,
- zusaetzlicher Einsatz des CRAY 1 Rechners

vorgeschlagen. Die Beschaffung der notwendigen Information fuer die Serien-Auswertung im Zuge der Schnell-Auswertung oder bei einem Vorlauf im Dialog erlaubt die Vernetzung der Programme fuer die Serien-Auswertung und ihre Verarbeitung im Stapelbetrieb. Auf diese Art koennen komplexe Datenpfade zur Optimierung des Rechenbetriebs erheblich beitragen.

Ferner koennen im Tagesbetrieb die Vorauswertungen so gestaltet werden, dass bei Nacht der zentrale Rechner im Stapelbetrieb voll ausgelastet ist. Damit sind nicht, wie in der Abschaetzung angenommen, 300, sondern 900 Operationen je Byte Experimentdaten moeglich. So koennen komplexe Auswertungen durchgefuehrt werden, wenn gleichzeitig bei der Schnell- und Vorauswertung gewonnene komprimierte oder ausgewertete Daten in geeigneter Weise gespeichert und bei der Serienauswertung als Eingabe verwendet werden.

Eine abschliessende Bemerkung scheint angebracht. In einer Studie zum Ausbau des ASDEX Datenerfassungs-Systems /15/ werden Zeitmessungen am zentralen Rechner (SIEMENS 7880) diskutiert, die dessen mangelnde Leistung -insbesondere bei I/O-Operationen- belegen sollen. Dabei sind mehrere entscheidende Fehler unterlaufen:

- I/O-Operationen werden, wie oben schon dargelegt, von der CPU nur angestossen, aber autonom durch die Steuereinheit abgewickelt, der Rechner somit nicht belastet, ausschliesslich die erforderlichen Wartezeiten koennen den Betrieb behindern,
- das Verhaeltnis Verweilzeit/CPU-Zeit haengt von dem Anteil der gemessenen VM an der gesamten CPU-Zeit ab; weder die virtuelle CPU-Zeit, noch ihr Anteil an der reellen CPU-Zeit wurden gemessen oder angegeben,
- die endgueltige Leseroutine fuer Experiment ASDEX wurde nicht verwendet; die in der Studie vorgestellten Zeiten basieren daher auf einem in absehbarer Zeit zu ersetzenden Provisorium.

Eine sachliche Diskussion ist auf dieser Grundlage nicht moeglich, zumal die Wiederholung der Messungen unter pruefbaren Bedingungen mit geeigneten Routinen ein um mehr als eine Groessenordnung guenstigeres Ergebnis lieferte.

Literatur-Verzeichnis

/1/ Burgess J., Hellauer G., Steuerwald J.: Untersuchungen zur Erfassung und Auswertung von Experimentdaten am Inststitut fuer Plasmaphysik in Garching (nicht veroeffentlicht)

/2/ Chu W. W.: Introduction in the Special Issue of Distributed Processing Systems. IEEE Trans. on Computers, Vol. C-29 (1980), pp. 1037 - 1038

/3/ Hertweck F.: Konzept eines Erfassungs- und Auswertesystems fuer die Experimente im IPP, Vortrag v. 27.1.83 (nicht veroeffentlicht).

/4/ IBM (Herausgeber) : Reference Manual for IBM 3350 Direct Access Storage, GA26-1638, 3rd Edition, April 1977, San Jose

/5/ IBM (Herausgeber) : IBM 3380 Disk Storage Description and User's Guide, GA26-1664

/6/ IBM (Herausgeber) : IBM 3270 Information Display System Component Description, GA27-2749

/7/ IBM (Herausgeber) : IBM Virtual Machine Facility/370, CMS Command and Macro Reference, GC20-1818, New York 1980

/8/ IBM (Herausgeber) : IBM Virtual Machine/System Product, System Programmer's Guide, SC19-6203, New York 1981

/9/ IBM (Herausgeber) : IBM Virtual Machine/System Product, System Product Editor User's Guide, SC24-5220, New York 1980

/10/ IBM (Herausgeber) : IBM Virtual Macine Facility/370: Performance/Monitor Analysis Program

/11/ Labiak W, Minor E.: The Software Design of a General Purpose Data Acquisition and Control System. IEEE Trans. Nucl. Sci., Vol NS-28 (1981)

/12/ Lathe R.: Gale Data File Manager (CMS Version), VM RFL, File DTM DOC, IPP (1979)

/13/ Lathe R., Mueller E.: GALE System Programmer's Handbook IPP R/27, Februar 1978

/14/ Madnik S. E., Donovan J. J.: Operating Systems, Mc Graw-Hill, New York 1974

/15/ Ruhs N.: Ausbau des ASDEX Datenerfassungs-Systems (nicht veroeffentlicht)

/16/ Saffert J.: ISIS User's Guide, CODAS Handbook 103.6, JET Abingdon

/17/ Speckert G., Grant C., Guenther D.: Toward a Generalized Computer System: a Condensed Analysis, Proc. 9th Symp. on Eng. Problems, Chicago (1981)

/18/ Steuerwald J.: EMPFN, Data Monitor fuer eine Datenbank wissenschaftlicher Daten (in Vorbereitung)

/19/ Steuerwald J., Steuerwald jr. j.: COMET Operator's Guide (in Vorbereitung)

/20/ Steuerwald J., Steuerwald jr. J.: COMET System Programmer's Guide (in Vorbereitung)

/21/ Steuerwald J., Tichmann Ch.: EDDAR System Survey, IPP R/34, November 1979

/22/ Wedekind, H.: Datenbanksysteme I und II, Mannheim-Wien-Zuerich, BI-Wissenschaftsverlag (1974)

Verzeichnis der Figuren

(1) zu Kapitel 1

(2) zu Kapitel 2

(3) zu Kapitel 3

(4) zu Kapitel 4

(5) zu Kapitel 5

(6) zu Kapitel 6

(7) zu Kapitel 8

(8) zu Kapitel 9 (Sofortauswertung)

(9) zu Kapitel 9 (serienauswertung)

(10) zum Anhang

Anhang A-1

Vorschlag fuer die Organisation der Diagnostik-Files

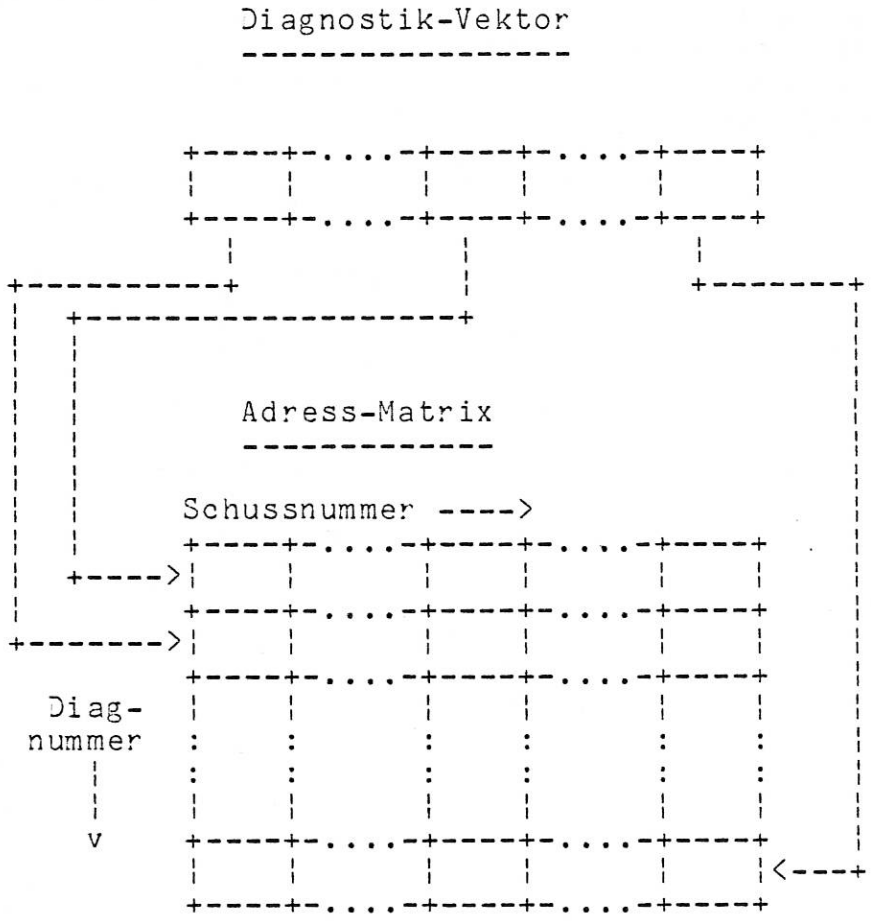


Fig. A1 Organisation der Pointer fuer Diagnostik-Files

(1) Diagnostikvektor. 256 Integer*4, die Zeilenadresse der Diagnostik i in der Adressmatrix steht im Element i des Diagnostik-Vektors.

(2) Adress-Matrix. Matrix mit 512 Spalten (fuer 512 Schuesse) und n Zeilen (n bestimmt durch die Zahl der gueltigen Adressen im Diagnostik-Vektor). Jedes Element der Matrix enthaelt die Blockadresse des Datenanfangs der gesuchten Diagnostik im relevanten Schuss.

Dazu sind Diagnostik-Files (Groesse richtet sich nach dem Datenanfall) definiert, die ueber die Allocation Map des CMS-Filesystems verwaltet werden. In dieser Organisation koennen Diagnostikdaten wegen der Uebereinstimmung der Block- und Recordgroesse ueber CMS-Macro FSREAD unmittelbar eingelesen werden.

Diese Anordnung ergibt sich aus der Tatsache, dass waehrend

des Schussbetriebes Auswertungen vorwiegend in der Spalte, bei der Serienauswertung dagegen in der Zeile vorgenommen werden.

Bemerkung: Es wird vorgeschlagen, die Schussnummer in Zukunft als Identifikation des Zeitintervalls zwischen Starttrigger eines Schusses bis zum Starttrigger des naechsten Schusses zu verwenden. Dazu ist es ausschliesslich noetig, mit dem Starttrigger die zentrale Zeit (in Mikrosekunden) zu senden und zu speichern. Auf diese Art koennen sporadische und langzeitige Diagnostiken ohne Schwierigkeiten praezise eingeordnet werden. Datenanforderungen beziehen sich dann auf absolute Zeitangaben.

Anhang A-2

Entmultiplexen von Datenstroemen

(1) Allgemeine Daten

- statisch,
- dynamisch

Statische Daten werden nur bei (seltenen) Aenderungen gesendet, dynamische bei jedem Schuss.

(2) Statische Daten Sie enthalten u. a.

- angeforderte Anzahl Datenitems,
- Laenge eines Datenitems in Bytes,
- Datenformat und den
- Pointer auf den Datenanfang.

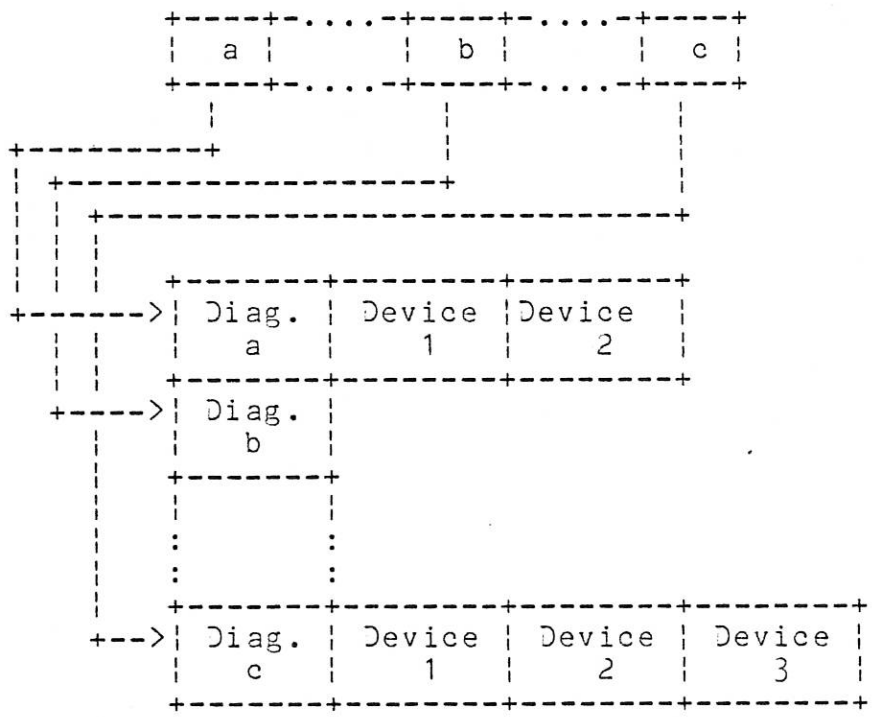


Fig. A2 Adress-Vektor fuer Datenanfaenge

(3) Pointer auf Datenanfaenge

- gemaess Definition in 2.
- Pointer-Vektor der Datenanfaenge (zusaetzlich bei Uebergabe der statischen Daten erzeugt). Jedes Element

enthaelt 2 Pointer (Integer*4 : zum Datenanfang und zum Strukturvektor).

(4) Strukturvektor Er wird aus den Daten zu 2. beim Senden der statischen Daten gebildet.

Anhang A-3

Datenpfad

Zur Definition eines Datenpfades wird dem Namen einer Variablen <variable> eine Liste von Pfaditems zugeordnet. Jedes Pfaditem enthaelt Angaben ueber:

- Experimentnamen,
- Schussnummer,
- Diagnostiknummer oder -namen,
- Geraetenummer,
- Zeitintervall.

Jeder dieser Begriffe kann durch Variablennamen dargestellt sein. Beim Aufruf des Datenpfades durch

read (<variable>)

wird festgestellt, ob die angeforderten Daten on-line, off-line oder nicht vorhanden sind. Sind Daten nicht vorhanden, wird das Programm abgebrochen, bei off-line Daten wird es angehalten, bis die Daten beschafft sind. In allen anderen Faellen wird eine Program-Unit gestartet, die keine Daten erwartet und die Daten fuer den naechsten Programmteil bereitgestellt oder Daten beschafft und danach wie oben weiter verfahren.

Bei Ausbau der zentralen Rechenanlage auf 32 MBytes Kernspeicher wird dieses Verfahren besonders wirksam.

Bemerkung: Das Verfahren zum Zurueckweisen von Abschnitten, deren Daten nicht vollstaendig vorhanden sind, kann ohne Schwierigkeiten von AMOS/2 uebernommen werden.

Anhang A-4

Zeitmessungen

(1) Auswertung der Protokolle des Schussbetriebs

Arbeitsgang	CPU (msec)
Lesen vom Kanal in Bloecken zu 512 Bytes	2,478
Konversion je Block zu 512 Bytes	0,193
Senden an VM <exp> DAT je Block zu 512 B	0,034

Fig. A3 Zeitmessungen aus Protokollen

(2) Messungen der Arbeitsgaenge

Arbeitsgang	CPU (mysec)
Senden VMCF-Message	733
Empfang VMCF-Message	39
Senden Daten mit VMCF	733
Emfang Daten mit VMCF	1800
Schreiben Daten auf Disk (MByte)	50000
Lesen Diagnostik von Disk (100 KByte)	6000
Lesen Diagnostik aus Kernspeicher	2000
Konversion mit Entmultiplexen (1 KByte)	700

Fig. A4 gemessene Zeiten

Bemerkung: Die Verweilzeiten wurden gleichzeitig gemessen.

Sie liessen sich stets -in angemessenem Rahmen- aus dem Verhaeltnis virtueller CPU-Zeit zu reeller CPU-Zeit und den vorgegebenen Prioritaeten erklaeren. Die Nachrechnung ergab, dass ausserdem durch I/O-Aktivitaeten keine zusaetzlichen Wartezeiten auftraten, sofern nicht ungewoehnliche Belastungen mit dem Fremdverbrauch an CPU-Zeit verbunden sind. Man kann daher auf die Angabe der Verweilzeit verzichten und sie durch den Anteil der virtuellen an der reellen CPU-Zeit ausdruecken.