KERNFORSCHUNGSANLAGE JÜLICH GmbH

Zentralinstitut für Angewandte Mathematik

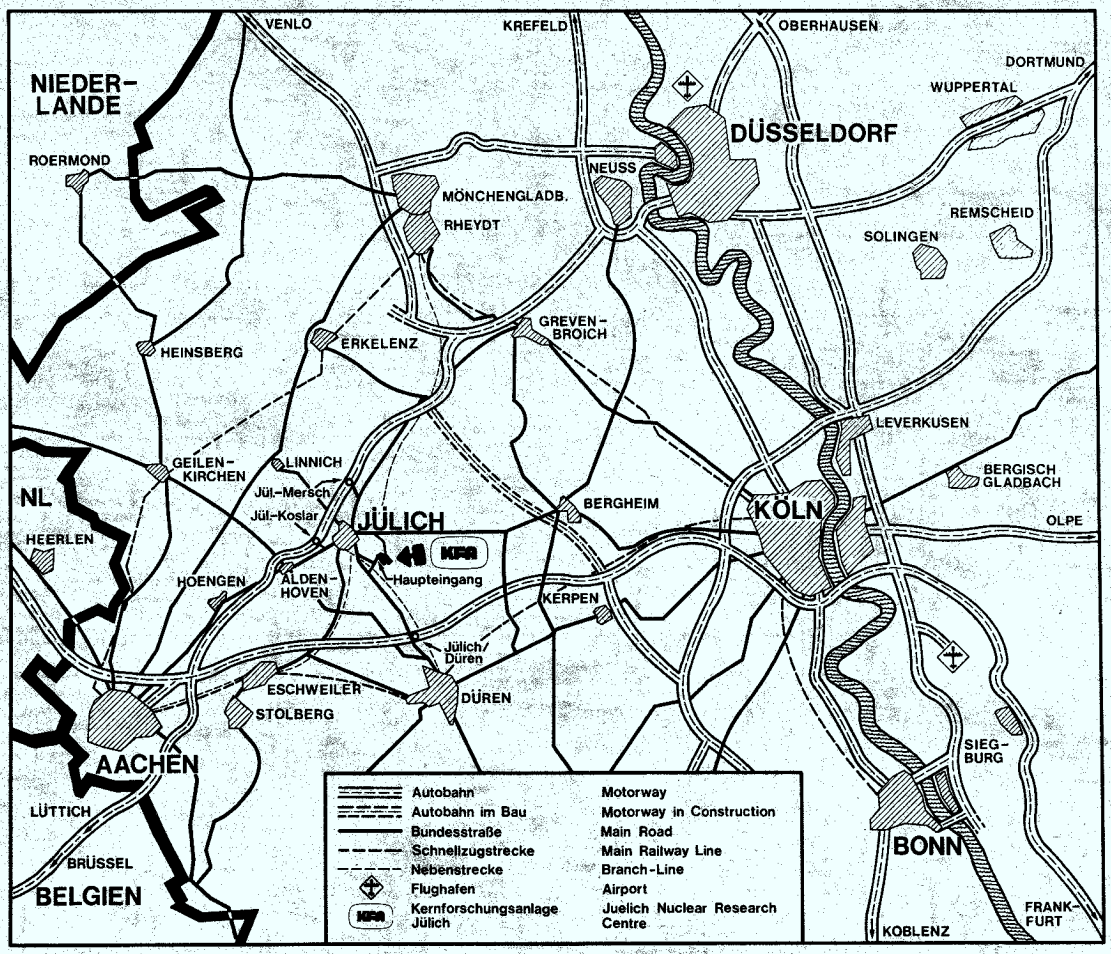
Der JOKER-Monitor für das Betriebssystem MVS

– Kommunikationsfunktionen –

von

P. Schiffeler, H. Schumacher

Jül - 1630
Dezember 1979
ISSN 0366-0885



Als Manuskript gedruckt

Berichte der Kernforschungsanlage Jülich - Nr. 1630

Zentralinstitut für Angewandte Mathematik Jül - 1630

Zu beziehen durch: ZENTRALBIBLIOTHEK der Kernforschungsanlage Jülich GmbH

Postfach 1913 · D-5170 Jülich (Bundesrepublik Deutschland)

Telefon: 02461/611 · Telex: 833556 kfa d

Der JOKER-Monitor für das Betriebssystem MVS

– Kommunikationsfunktionen –

von

P. Schiffeler*, H. Schumacher

*IBM Deutschland

Zusammenfassung

Über das Kopplungssystem JOKER können Experimentrechner verschiedener Fabrikate mit dem Zentralrechner verbunden werden. Der vorliegende Bericht beschreibt die Anwendungsmöglichkeiten des JOKER-Systems unter dem IBM-Betriebssystem MVS. Den Benutzern des JOKER-Systems wird eine Reihe von Prozeduren, Utilities und Unterprogrammen zur Verfügung gestellt, die es ihnen sowohl im Batch- als auch im Timesharing-Betrieb gestatten, sich der Funktionen des JOKER-Netzes zur Datenübertragung und -manipulation auf einfache Weise zu bedienen. Statusinformationen über Systemzustand sowie über die Datenbestände können abgerufen werden.

Abstract

The coupling system JOKER has been designed and implemented to connect experimental computers of various types with the central computing system. The present report describes the facilities of the system JOKER under the IBM Operating System MVS. A variety of procedures, utilities and subroutines is given to the users, which permits them to utilize in an easy way the functions of the JOKER-Net for data transfer and manipulation both in batch- and timesharing-mode. Status information concerning the system as well as the data base is provided by the users.

Inhaltsverzeichnis

1 Aufbau und Wirkungsweise von JOKER unter MVS

1.1 Einleitung

Das Online-System JOKER zur Kopplung von Experimentrechnern verschiedener Fabrikate mit einem zentralen Timesharing-Rechner ist zu Beginn des Jahres 1972 in Betrieb genommen worden /1/. In seiner achtjährigen Betriebszeit hat es sich als sehr zuverlässig und benutzerfreundlich erwiesen.
Bedingt durch die Betriebssystempolitik der IBM erschien es notwendig, das Kopplungssystem JOKER nicht nur unter dem Betriebssystem TSS (Time Sharing System) /4/ , sondern auch unter dem Betriebssystem MVS (Multiple Virtual System) /5/ sowohl im Batch- als auch im Timesharing-Betrieb betreiben zu können.
Da die Systemphilosophie des MVS zum Teil stark von der des TSS abweicht, war es nicht möglich, den zentralen Softwarekomplex (Monitor, Prozeduren, Kommandos und Subroutinen) ohne tiefgreifende Umgestaltung im logischen Aufbau zu übertragen. Aus diesem Grunde wurde im Jahre 1976 eine Neukonzipierung beschlossen. Dadurch wurde es möglich, einerseits günstige Eigenschaften des MVS auszuschöpfen und andererseits die während der mehrjährigen Betriebszeit als sinnvoll erkannten Funktionsänderungen und notwendigen Funktionserweiterungen von vornherein bereits im Konzept zu berücksichtigen. Zusätzlich konnten einige bis jetzt bestehende Beschränkungen beseitigt werden.

Der Benutzer von JOKER sollte sich auf die Auswertung seiner Daten konzentrieren können. Deshalb werden ihm die nötigen Funktionen durch Prozeduren, Utilities und Subroutinen zur Verfügung gestellt. Bei der Neukonzipierung unter MVS wurde besonderer Wert darauf gelegt, so weit wie möglich die gleichen Namen wie unter TSS zu verwenden. Vom Benutzer werden folgende Funktionen von JOKER gefordert:

Gesicherter Datentransfer von und zum Experimentrechner
Datenzugriff und -manipulation
Statusinformationen über System und Daten
Hilfsmittel bei der Suche nach Bedienungsfehlern

Die Erfahrungen haben gezeigt, daß es im wesentlichen zwei Anwendungsformen gibt:

(1) Die Daten werden am Experiment über einen längeren Zeitraum gesammelt und dann zur späteren Auswertung und Archivierung zum Zentralrechner geschickt.

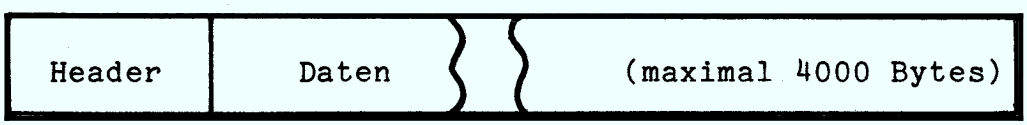
(2) Der Zentralrechner wird mit in den Echtzeitbetrieb eingeschlossen, d.h. die anfallenden Meßdaten werden direkt zum Zentralrechner übertragen und sofort ausgewertet. Die Ergebnisse werden zum Experimentrechner zurückgeschickt und können so bei der Steuerung des Meßablaufes mitberücksichtigt werden.

1.2 Aufbau und Identifikation der Datensätze

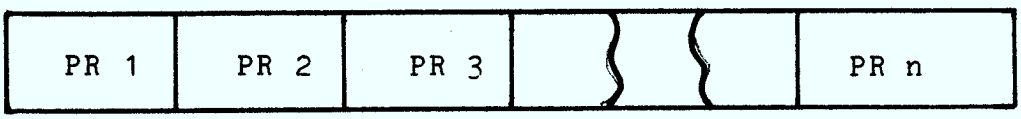
Der Experimentator sendet und empfängt seine Daten mit Hilfe von Treibern bzw. geeigneten Ein-/ Ausgaberoutinen, die es gestatten, mit dem Übertragungsnetz in gleicher Weise zu verkehren wie mit anderen Peripherie-Geräten. Für die meisten der für PDP8- und PDP11-Rechner angebotenen Betriebssysteme stehen Treiber zur Verfügung.

1.2.1 Übertragung zum Zentralrechner

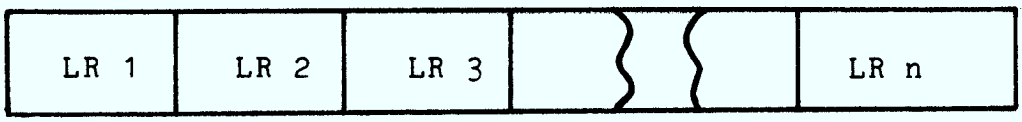
Für die Organisation seiner Daten steht dem Experimentator folgende hierarchische Strukturierungsmöglichkeit zur Verfügung (Abb.1) :
-physikalischer Record-
-logischer Record-
-Reading-
Die kleinste Einheit ist der physikalische Record (PR). Dieser besteht aus einem Header von 14 Bytes und den eigentlichen Daten. Die maximale Datenlänge für einen Transfer beträgt 4000 Bytes. Ein oder mehrere physikalische Records werden zu einem logischen Record (LR) zusammengefaßt. Ein oder mehrere logische Records bilden das Reading (RDG). Dieses wird in der Regel (Ausnahme: ´Fast dialog´) mit eindeutiger Identifikation in einen Dataset geschrieben. Erst wenn es dort vollständig abgespeichert ist, kann der Experimentator darauf zugreifen.



<---- Physikalischer Record (PR) ---->



<---- Logischer Record (LR) ---->



<---- Reading (RDG) ---->

Abb. 1 : Struktur der Datensätze

Der Header eines physikalischen Records dient dem Zentralrechner zur Identifizierung der Daten. Sein Aufbau ist in Abb. 2 dargestellt.

Record-Nr.	Indikatoren	RDG-Nr.	frei	frei	User-Nr.	Exp-Nr.

-Record-Nr.: Nummer des physikalischen Records. Sie muß bei eins beginnen und sollte innerhalb eines Readings fortlaufend numeriert werden.

-Indikatoren: Beschreibung siehe nächste Seite

-RDG-Nr.: Nummer des Readings, zu dem der physikalische Record gehört.

-User-Nr.: Benutzer-Identifikation im JOKER-System

-Exp-Nr.: Experiment-Identifikation im JOKER-System

Abb. 2 : Aufbau des Headers

Änderungen gegenüber dem derzeitigen Header gibt es nur im Indikatorwort. Der Aufbau ist in Abb. 3 dargestellt. Die Bits 13 bis 16 dürfen nicht verwendet werden, da es auch Experimentrechner (PDP8) mit 12 Bit Wortlänge gibt und der Header für alle angeschlossenen Rechner einen identischen Aufbau haben soll.

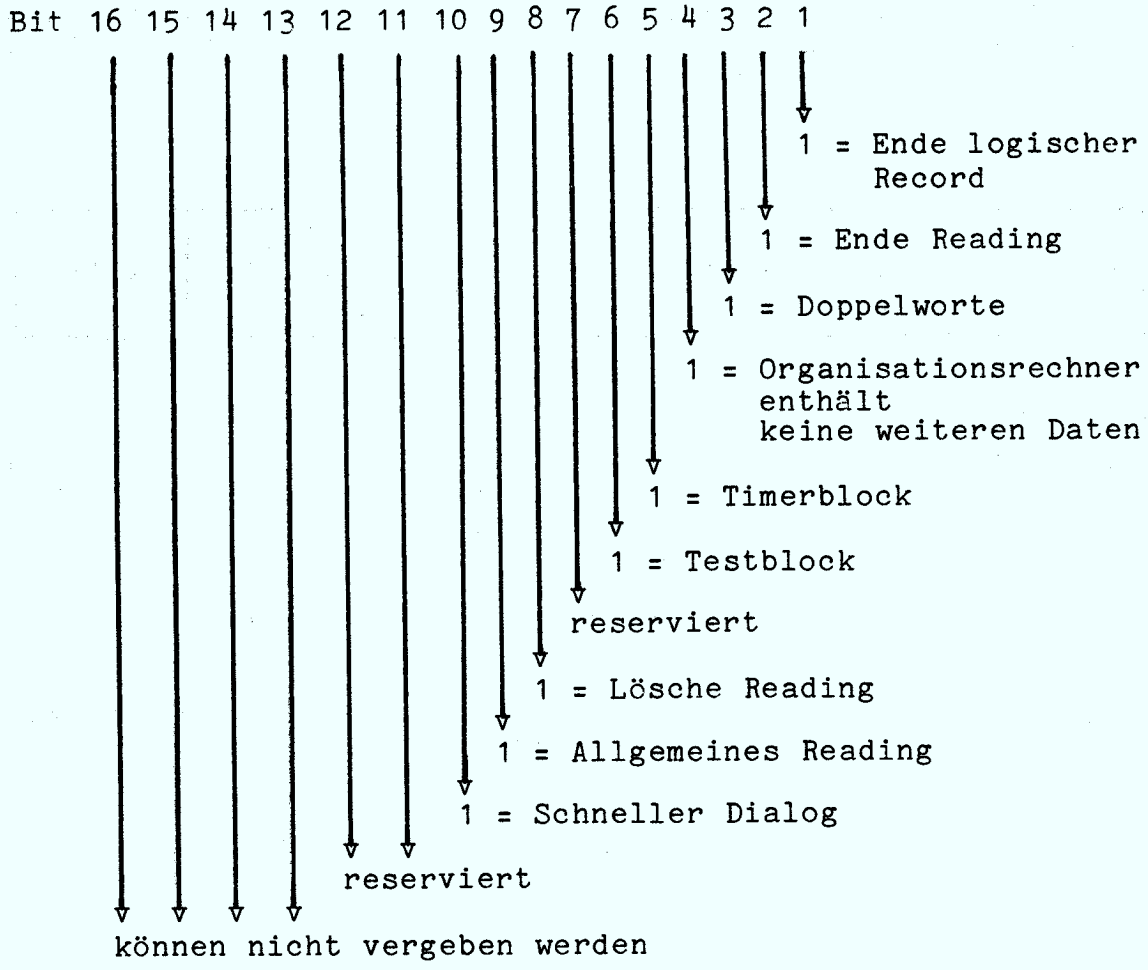


Abb. 3 : Aufbau des Indikatorwortes

Die geänderten Bits haben dabei die folgenden Bedeutungen :

(1) Bit 8: Lösche Reading

Dieses Bit wird in Verbindung mit der Record-Nummer 1 dann gesetzt, wenn ein schon vorhandenes, abgeschlossenes Reading mit dem gleichen Namen gelöscht und ein Reading mit dem gleichen Namen neu anlegt werden soll. Welche Konsequenzen das Setzen dieses Bits sonst noch hat, ist aus der Entscheidungstabelle (Abb. 6) ersichtlich (Vgl. Abschnitt 1.4).

(2) Bit 9: Allgemeines Reading

Durch das Setzen dieses Bits wird dem Monitor mitgeteilt, daß der Benutzer einen Batch-Job starten möchte. Der Header eines solchen Readings enthält ein Data-Set-Suffix. Mit diesem Suffix wird ein Dataset-Name gebildet. Dieser Dataset muß vorher vom Benutzer erstellt werden und alle nötigen

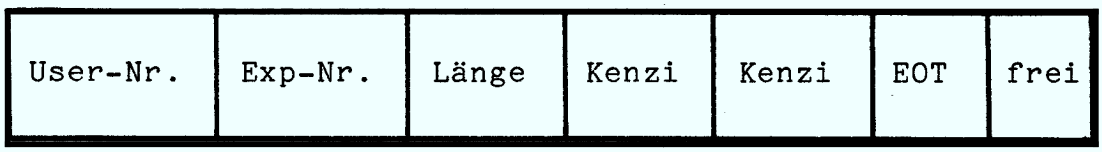
Job-Control-Language-Statements enthalten. Der Inhalt dieses Datasets wird dem MVS als Batch-Job zugeführt (Siehe Abschnitt 1.5.3).

(3) Bit 10: Schneller Dialog

Dieses Bit (=1) besagt, daß der Experimentator ein Reading bestehend aus einem maximal 4000 Bytes langen Record sofort auswerten möchte. Der Record wird nicht auf der dem JOKER-System fest zugeordneten Platte sichergestellt. Um den Zugriff zu den Daten zu beschleunigen, wird eine spezifische Eigenart des MVS ausgenutzt. Der Record wird in einem Puffer der Common Service Area (CSA) , das ist ein für alle zugänglicher Bereich, bereitgestellt (Siehe Abschnitt 1.5.2).

1.2.2 Übertragung zum Experimentrechner

Normalerweise wird beim Senden der Daten zum Experimentrechner kein Header benötigt. Zur Identifizierung reicht die Experimentrechner-Hardware-Adresse aus. Da inzwischen Experimentrechner mit Multi-User-Betriebssystemen angeschlossen sind, wird auch hier ein 14 Bytes langer Header vorangestellt. Sein Aufbau ist in Abb. 4 dargestellt.



-User-Nr.: Benutzer-Identifikation im JOKER-System

-Exp-Nr.: Experiment-Identifikation im JOKER-System

-Länge: Länge der Daten

-Kenzi: Kennung des Header für ein Multi-User-Betriebssystem im Experimentrechner

-EOT: Ende des Datensatzes

Abb. 4 : Aufbau des Headers

1.3 Organisation der Kontroll- und Datenblöcke

Wie jedes andere System hat auch das JOKER-System eine Vielzahl von Kontrollblöcken. Diese sind in /2/ ausführlich beschrieben. In der Common Service Area (CSA) des MVS befinden sich alle wichtigen Kontrollblöcke. Zu diesen gehören der Common System Block (CSB) und der Common Reading Block (CRB). Diese Blöcke enthalten:

User-Slots	(CSB)
Experiment-Slots	(CSB)
Reading-Slots	(CRB)
TTR-Slots	(CRB)

Da der Benutzer seine Daten zu jeder beliebigen Zeit senden kann, ohne daß im Zentralrechner ein von ihm gestartetes Programm die Daten in Empfang nimmt, dienen diese Slots dazu, die im Header angegebenen Informationen mit den Angaben, die sich in diesen Slots befinden, zu vergleichen. Wenn alle Angaben übereinstimmen, können die Daten auf die Platte abgespeichert werden.
User- und Experiment-Slots werden angelegt, wenn ein Benutzer zum System zugelassen wird (Join-Prozess), Reading- und TTR-Slots dagegen, wenn die Daten auf der Platte sichergestellt werden.
Diese Slots dienen einmal zur Kennzeichnung der Daten und zum anderen als eine Art Directory.

Jeder User-Slot enthält u.a. folgende Informationen:

-User-Nr.

Diese Nummer wird einmalig beim Anlegen des User-Slots für einen bestimmten Benutzer vergeben. Zur Identifikation des Daten-Records muß sie im Header angegeben werden.

-Userid

Beim Zugriff auf die Daten wird das Userid des Anforderers mit dem aus dem Task-Control-Block des MVS-Systems verglichen, so daß sichergestellt ist, daß jeder Benutzer nur seine eigenen Daten bekommt.

-Einen Pointer zum ersten Experiment-Slot des Benutzers.

Ein Benutzer kann ein oder mehrere Experimente gleichzeitig betreiben. Für jedes dieser Experimente wird beim Join-Prozess ein Experiment-Slot angelegt.

Ein Experiment-Slot enthält u.a:

-Exp-Nr.

Sie dient zur Identifizierung des Experiments und muß im Header angegeben werden.

-Experimentrechner-Hardware-Adresse

Diese wird vom Organisationsrechner im Header eingetragen. Durch den Vergleich von User-Nr., Exp-Nr. und Experimentrechner-Hardware-Adresse können die Daten nicht nur eindeutig einem Benutzer zugeordnet werden, sondern es kann in gewissem Umfang auch eine Rechtmäßigkeitskontrolle durchgeführt werden.

-Pointer zum nächsten Experiment-Slot des Benutzers.

-Pointer zum ersten Reading-Slot dieses Experiments.

Jedes Experiment kann ein oder mehrer Readings besitzen. In jedem Reading-Slot wird u.a. festgehalten:

-RDG-Nr.

Sie dient zur Identifizierung des Datensatzes und muß im Header angegeben werden.

-Pointer zum nächsten Reading-Slot desselben Experiments.

-Pointer zum aktuellen Datenblock im Kernspeicher.

-Indikatoren, die den Zustand des Readings festhalten.

-Pointer zum ersten TTR-Slot.

Ein TTR-Slot enthält die Informationen über den Bereich, in dem sich die Daten auf der Platte befinden. Je nach der Menge der Daten können weitere, miteinander verkettete TTR-Slots vorhanden sein. Diese Slots sind notwendig, da keine IBM-Standard-Zugriffsmethode für den Verkehr mit der Platte verwendet wird.

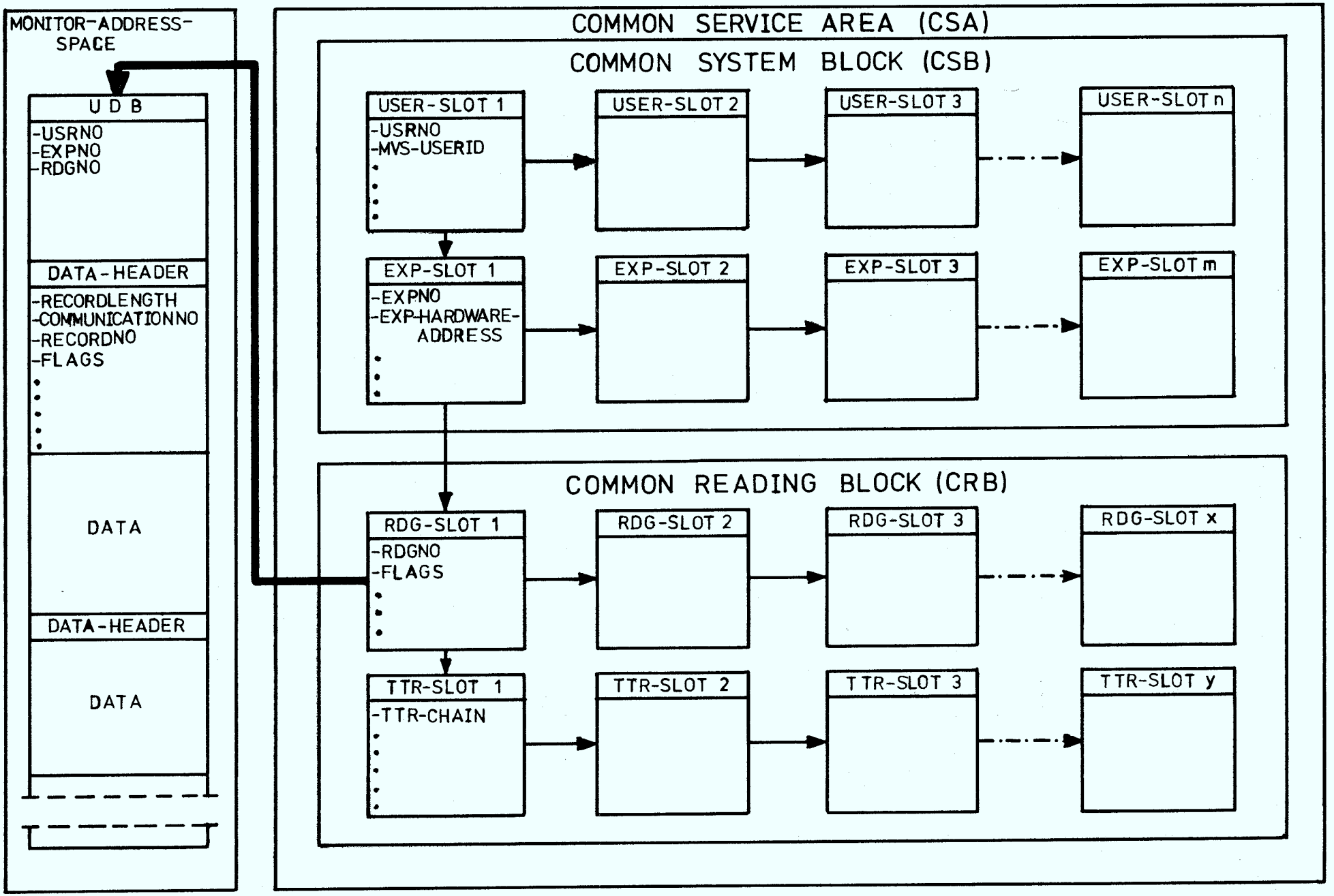


ABB. 5: Kontroll-und Datenblöcke

Jeder RDG-Slot hat einen Pointer zu einem 4k Datenblock, dem User-Data-Block (UDB). Dort werden die Daten eines Readings im Kernspeicher gesammelt. Wenn ein solcher Block voll oder ein Reading abgeschlossen ist, wird er auf die Platte geschrieben. Im zugeordneten TTR-Slot wird dann ein entsprechender Eintrag gemacht.

Der UDB besteht aus einem UDB-Header und einem oder mehreren physikalischen Records. Der UDB-Header enthält u.a. die User-Nr., Exp-Nr. und die RDG-Nr., um den Datensatz eindeutig identifizieren zu können. Danach folgen die einzelnen Records. Auch diese haben einen Header, in dem sich u.a. die Record-Länge des nachfolgenden Records und die Indikatoren befinden.

Die Verwendung von User-Nr., Exp-Nr. und RDG-Nr. hat die gleiche Funktion wie ein Dataset-Name. Diese Darstellung wurde gewählt, da bei den verschiedenen Experimentrechnertypen die Darstellung der Festkommazahlen gleich ist, die Zeichendarstellung jedoch nicht.

1.4 Regeln für die Record-Nummer

Mit einer Record-Nummer (im nachfolgenden als RNO bezeichnet) von 1 im Header eines physikalischen Records wird dem JOKER-Monitor bekanntgegeben, daß ein neues Reading anzulegen ist. Zur Kontrolle der Vollständigkeit sollte RNO in jedem folgenden Record eines Readings um 1 inkrementiert werden. Unregelmäßigkeiten in der Numerierung werden dem Benutzer beim späteren Zugriff auf dieses Reading mitgeteilt.

Record-Nummer \ Status Rdg	offen	abgeschlossen	nicht vorhanden
RNO = 1	.Lösche altes Reading .Eröffne neues Reading	.´Delete-Flag´ an im Header: wie unter ´offen´ und RNO = 1 .´Delete-Flag´ aus im Header: Lösche Record	Eröffne neues Reading
RNO ≠ 1	Teste Record-Nummer-Sequenz Folgende Flags können im Reading-Slot gesetzt werden: ´Out of sequence´ ´Increment > 1´	Lösche Record	Lösche Record

Abb. 6 : Entscheidungstabelle

Beim Eintreffen eines Records können drei verschiedene Zustände des zugehörigen Readings vorliegen:
-offen,
-abgeschlossen, oder
-nicht vorhanden.
In Abhängigkeit von der Record-Nummer werden vom JOKER-Monitor folgende Maßnahmen durchgeführt (Abb. 6) :

1.4.1 Record-Nummer RNO = 1

a) Ein Reading gleichen Namens existiert und ist offen.

Das alte Reading wird gelöscht und ein neues mit gleichem Namen wird angelegt.

b) Ein Reading gleichen Namens existiert und ist ordnungsgemäß abgeschlossen.

Falls im Header des ankommenden Records das ´Delete-Bit´ gesetzt ist, wird das abgeschlossene Reading gelöscht und ein neues Reading gleichen Namens, beginnend mit dem ankommenden Record, angelegt. Andernfalls wird der ankommende Record für ungültig erklärt und nicht gespeichert.

c) Es existiert kein Reading gleichen Namens (Normalfall).

Es wird ein neues Reading angelegt.

1.4.2 Record-Nummer RNO ≠ 1

a) Das Reading existiert und ist offen.

Es wird $\Delta RNO = RNO_n - RNO_{n-1}$ gebildet. Falls $\Delta RNO \leq 1$ wird ´Out of sequence´, falls $\Delta RNO > 1$ wird ´Increment > 1´ im Reading-Slot festgehalten. Der Record wird in jedem Fall gespeichert.

b) Das Reading ist abgeschlossen.

Der Record wird gelöscht.

c) Es existiert kein Reading mit gleichem Namen.

Der Record wird gelöscht.

1.4.3 Offene Readings

Jeder Benutzer darf nur ein offenes Reading pro Experiment haben. Soll ein neues Reading, mit anderem Namen, für dieses Experiment eröffnet werden (RNO = 1 im Header), wird das offene Reading zwangsweise abgeschlossen und das neue Reading angelegt. Auch dieser Sachverhalt wird im Reading-Slot festgehalten und dem Benutzer beim späteren Zugriff auf dieses Reading mitgeteilt.

1.5. Datenarten und ihre Handhabung

Der Benutzer schickt seine Daten vom Experimentrechner über den Organisationsrechner zum JOKER-Monitor im Zentralrechner. Dieser trennt Header und Daten. Er analysiert den Header und macht Plausibilitätskontrollen. Eventuelle Fehler werden im Header angezeigt. Die Daten lassen sich in drei Gruppen einteilen:

(1) der normale Datenblock
(2) der schnelle Dialog
(3) das allgemeine Reading.

1.5.1 Der normale Datenblock

Je nach Art des Records wird entweder

- ein Reading angelegt
- ein Record im UDB gespeichert
- ein Reading abgeschlossen
- ein Reading gelöscht
- oder der ungültige Record gelöscht.

a.) Anlegen eines neuen Readings

Soll ein Reading neu angelegt werden, so werden in der Common Service Area (CSA) des MVS ein Reading-Slot und der erste TTR-Slot aufgebaut. Anschließend wird aus der Puffer-Queue ein freier 4k-Bytes großer Puffer geholt, ein UDB-Header erzeugt und eingetragen. Danach werden die Daten mit einem vorangestellten Data-Header in diesen UDB übertragen. Die Adresse des Puffers wird dabei im Reading-Slot gespeichert.

b.) Abspeichern eines Folge-Records

Existiert bereits ein Reading für den Record, wird aus dem Reading-Slot die zugehörige Puffer-Adresse geholt, und die Daten werden in den Puffer übertragen. Wird dabei der Puffer voll, so wird er auf der Platte sichergestellt und anschließend ein neuer Puffer für dieses Reading bereitgestellt. Die neue Puffer-Adresse wird wieder im Reading-Slot abgespeichert. Konnten die

Daten nur teilweise vom vorhergehenden Puffer aufgenommen werden, so werden die Restdaten, nach Bildung des UDB-Headers, in diesen neuen Puffer übertragen.

c.) Abschließen eines Readings

Es gibt mehrere Bedingungen, die zum Abschluß eines Readings führen. Im Normalfall ist es das gesetzte 'End of Reading'-Bit im Indikatorwort des Headers. Dann werden die Daten des Records in den aktuellen Puffer übertragen und auf der Platte sichergestellt. Zusätzlich wird ein 'Reading closed'-Bit im Reading-Slot gesetzt.
Eine zweite Abschlußbedingung ist das Vorhandensein eines offenen Readings, während der Benutzer versucht, für das gleiche Experiment ein weiteres Reading mit anderem Namen zu eröffnen.
Das offene Reading wird dann zwangsweise abgeschlossen. Dies geschieht genau so wie oben beschrieben, wobei zusätzlich im Reading-Slot diese Zwangsbeendigung gekennzeichnet wird.

d.) Löschen eines Readings

Wird ein Record mit der Record-Nummer RNO = 1 geschickt, während das zugehörige Reading noch offen ist, wird angenommen, daß der Benutzer den bisher übertragenen Teil des Readings löschen und ein neues Reading mit dem gleichen Namen neu erstellen möchte.
Um ein Reading als gelöscht zu kennzeichnen, wird ein 'Delete'-Bit im Reading-Slot gesetzt.
Damit sind automatisch alle Blöcke, im Kernspeicher sowie auf der Platte, die zu diesem Reading gehören, ungültig geworden und können freigegeben werden.

Ein Reading wird auch gelöscht, wenn im Header mit RNO = 1 das 'Delete'-Bit gesetzt ist und ein Reading gleichen Namens bereits vorhanden und abgeschlossen ist. Auch in diesem Fall wird das 'Delete'-Bit im Reading-Slot des alten Readings gesetzt.

In beiden Fällen wird in einer 'Delete Reading'-Tabelle ein Eintrag gemacht, um dem JOKER-Monitor beim endgültigen Löschen der Blöcke das Durchsuchen aller im System befindlichen Readings zu ersparen.

e.) Löschen eines ungültigen Records

Ein Record ist ungültig, wenn die im Header angegebene Benutzer- bzw. Experiment-Identifikations-Nummer unbekannt ist.
Die anderen Fälle, in denen ein Record für ungültig erklärt wird, können Abb. 6 entnommen werden.
Bei jedem Record -ob gültig oder ungültig- wird auf jeden Fall der Header gespeichert. Dies dient der

statistischen Auswertung und der Analyse von Fehlerbedingungen, die im Header festgehalten werden.

Von jedem Record -ob gültig oder nicht- wird außerdem eine im Header enthaltene Bestätigungsnummer in eine Tabelle eingetragen. Wenn der Record ordnungsgemäß verarbeitet worden ist, wird sie zum Organisationsrechner geschickt. Dort kann der korrespondierende Record dann gelöscht werden.

1.5.2 Der schnelle Dialog

Der schnelle Dialog gestattet dem Benutzer eine sofortige Auswertung seiner Daten. Sie werden nicht auf der Platte zwischengespeichert. Dabei müssen folgende Voraussetzungen erfüllt sein:

- Der Benutzer hat genau ein Auswerteprogramm für den schnellen Dialog gestartet. Wenn kein Auswerteprogramm gestartet ist, gehen die Daten verloren.
- Es soll ein Dialog im strengen Sinne durchgeführt werden, d.h. nach jedem Lesen sollen zur Synchronisation auch jeweils Daten zurückgeschickt werden.
- Zugelassen sind nur Readings mit einem physikalischen Record von maximal 4000 Bytes.

Bei der Kommunikation zwischen JOKER-Monitor und Benutzerprogramm wird bei diesem Datenzugriff die Funktion 'Intertaskkommunikation' ausgenutzt. In der Common Service Area des MVS befindet sich eine 'User Communication'-Tabelle (UCT). Sie dient als Bindeglied zwischen JOKER-Monitor und Benutzerprogramm. Für jeden Benutzer wird in dieser UCT ein Entry mit seiner User-Nummer reserviert. Fordert das Benutzerprogramm Daten an, bevor sie der JOKER-Monitor erhalten hat, wird im zugehörigen UCT-Entry die Exp-Nr., die RDG-Nr., eine ECB-Adresse und die Adresse des Adressraumes (ASCB) eingetragen. Der JOKER-Monitor schreibt nach Empfang der Daten die Pufferadresse in den UCT-Entry und aktiviert die Benutzer-Task durch ein XMPOST (Cross Memory Post). Der Puffer wird von jetzt an für den schnellen Dialog dieses Benutzers verwendet. Das Benutzerprogramm entnimmt diesem in der CSA befindlichen Puffer die Daten. Sendet der Benutzer neue Daten, werden die alten überschrieben, unabhängig davon, ob die alten bereits ausgelesen waren oder nicht.

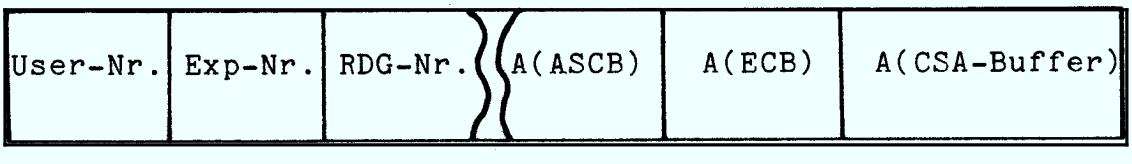


Abb. 7 : U C T - Entry

1.5.3 Das allgemeine Reading

Durch die Funktion des 'allgemeinen Readings' hat der Experimentator die Möglichkeit von seinem Experimentrechner Batch-Jobs im Zentralrechner zu starten. Es wird durch das Setzen des Bits 'allgemeines Reading' und die Angabe eines Dataset-Suffix im Header gekennzeichnet. Dadurch kann der Benutzer angeben, welche Funktionen er ausgeführt haben möchte. Mit Hilfe des Dataset-Suffix wird nun folgender Dataset-Name aufgebaut: Userid.Batchxx.Data (xx ist das Suffix). In diesen Dataset hat der Experimentator vorher alle Job-Control-Language-Statements eingetragen, um die gewünschte Funktion in einem Batch-Job ausführen zu lassen. Der Inhalt dieses Datasets wird über einen Internal-Reader dem MVS als Job zugeführt, nachdem vorher Teile der JCL überprüft worden sind. Eventuelle Benutzer- oder Systemfehler werden im User-Slot festgehalten. Vom JOKER-Monitor wird noch ein Service-Programm hinzugefügt, das sicherstellt, daß für einen Benutzer jeweils nur ein solcher Job im MVS-System gleichzeitig gestartet ist.
Wenn ein Benutzer ein 'allgemeines Reading' schickt, wird zuerst geprüft, ob für ihn vom JOKER-Monitor schon ein Batch-Job gestartet ist. Sollte dies der Fall sein, so wird diese Anforderung ignoriert. Ansonsten wird der gewünschte Batch-Job gestartet.

1.6. Der Datenzugriff

Zum Lesen und Schreiben der Daten werden Subroutinen bereitgestellt. Subroutinen haben den Vorteil, daß bei einem Systemwechsel nur wenige Änderungen erforderlich sind. Außerdem kann das JCL-Statement für den Dataset entfallen, da diese Angaben als Parameter an die Subroutine weitergereicht werden.
Der Zugriff zu den Daten wird durch verschiedene READ-Subroutinen ermöglicht, die sich einer eigenen Zugriffsmethode bedienen. Die Daten können nur sequentiell gelesen werden. Zu ihrer Identifizierung genügen die User-Nr., die Exp-Nr. und die RDG-Nr. Die Daten werden in einen vom Benutzer angegebenen Puffer übertragen. Ein Return-Code gibt Auskunft über den Status der ausgeführten Funktion.
Mit Hilfe von WRITE-Subroutinen werden Daten zum Experimentrechner geschickt. Ein Return-Code gibt an, ob die Übertragung erfolgreich war oder nicht.

1.7. Synchronisation der Zugriffe zu den Kontrollblöcken

Da verschiedene Tasks auf die Kontrollblöcke zugreifen können, entstehen kritische Abschnitte. Diese müssen synchronisiert werden. Dieses Problem wurde durch die Hardware-Instruktion ´Compare and Swap´ gelöst. Da diese Instruktion nicht unterbrechbar ist, kann zwischen Einleitung und Beendigung der Operation keine andere Aktion ablaufen. Als Semaphor dient dazu das ´Interlock Field´. Da das Benutzerprogramm im Normalfall nur Informationen abrufen, der JOKER-Monitor hingegen Änderungen vornehmen möchte, wurde folgender Synchronisationsmechanismus vereinbart:
Der JOKER-Monitor setzt das ´Interlock Field´ negativ und wartet bis alle Benutzerprogramme ihre Arbeit beendet haben. In dieser Zeit kann kein neues Benutzerprogramm mehr auf die Kontrollblöcke zugreifen. Nach Beendigung seiner Arbeit setzt der JOKER-Monitor das ´Interlock Field´ auf Null. Die Benutzerprogramme erhöhen zu Beginn das ´Interlock Field´ um 1. Bei Beendigung wird es um 1 erniedrigt. Wenn sich der JOKER-Monitor im Wartezustand befindet und kein Benutzer arbeitet, hat es also den Wert Null.
Der einzige Fall, wo ein ähnlicher Mechanismus angewandt wird, um Benutzerprogramme zu synchronisieren, tritt beim Löschen der Daten auf. Dazu wird in der ´Delete Reading´-Tabelle (DRT) in einem Entry ein entsprechender Eintrag gemacht. Da jeder Entry nur einmal vergeben werden darf, besteht diese Tabelle aus einem Header und den Entries. Im Header gibt es einen Pointer zum ´current´ Entry. Auf diesen Pointer wird nun mit Hilfe von ´Compare and Swap´ die Adresse des nächsten Entries gesetzt. Am Return-Code kann man erkennen, ob die Operation erfolgreich war. Ansonsten muß sie wiederholt werden. Durch ein XMPOST wird dem JOKER-Monitor mitgeteilt, daß er das im DRT-Entry eingetragene Reading löschen soll. Die Benutzer-Task wartet dann solange bis sie nach Abschluß der Arbeit vom JOKER-Monitor durch ein XMPOST wieder aktiviert wird.

2 Anleitung zur Benutzung

2.1 Ein-/ Ausgabe

2.1.1 Eingaberoutinen

(1) READ

Der Aufruf erfolgt durch:

```
CALL READ (Expno,Rdgno,Toaddr,Maxlng,Actlng,Rc)
```

Mit diesem Unterprogramm kann der Experimentator Daten aus dem angegebenen Reading lesen. Bei Erfolg erhält er einen logischen (nicht physikalischen!) Record. Der Benutzer kann gleichzeitig bis zu zehn Readings geöffnet haben und daraus lesen. Um den Lesevorgang ordnungsgemäß zu beenden, d.h. das Reading wieder zu schließen, muß man nach Return-Code=10 das Unterprogramm R D G E N D aufrufen oder bis Return-Code=2 weiterlesen.

Parameter, Bedeutung und Deklarationen :

Expno : Nummer des Experiments, dem die gewünschten Daten zugeordnet sind. Die Experimentnummer muß dem JOKER-System bekannt sein.
Integer*4

Rdgno : Nummer eines Readings, das von der Platte gelesen werden oder auf dessen Erstellung gewartet werden soll.
Integer*4

Toaddr : Einfach-dimensionierte Größe von beliebigem Typ (meistens Logical*1). Sie enthält nach erfolgreichem Aufruf von R E A D byte-weise einen logischen Record.

Maxlng : Diese Größe gibt an, wieviele Bytes der übertragene, logische Record maximal enthalten kann.
Integer*4

Actlng : In diese Größe wird von R E A D die genaue
------ Anzahl Bytes des gerade gelesenen, logischen Records eingetragen.
Integer*4

Rc : An diese Variable gibt R E A D einen Return-Code
-- zurück, der auf jeden Fall vom Benutzer abgefragt werden sollte. Diese Variable ist logisch in zwei Halbworte gegliedert.
Das erste Halbwort enthält Hinweise auf mögliche Benutzerfehler bei der Datenstrukturierung.
Das zweite Halbwort gibt Auskünfte über den Erfolg bei der Ausführung von R E A D.
Integer*4

Zusätzlich zu diesen Return-Codes druckt R E A D noch eine Fehlermeldung aus.

Es gibt folgende Return-Codes und Fehlermeldungen:
---

1. Halbwort

Rc = 32 : Increment not one

Das Reading wurde nicht mit aufsteigenden Record-Nummern und einem Increment von 1 geschickt.

Rc = 16 : Increment out of sequence

Das Reading wurde nicht mit aufsteigenden Record-Nummern geschickt.

Rc = 8 : Reading forced

Das Reading wurde zwangsweise beendet, da es noch nicht abgeschlossen war,als ein neues geschickt wurde.

2. Halbwort

Rc = 0 : R E A D war erfolgreich.

Rc = 1 : False dsname

Das Reading wurde nicht gefunden.

Rc = 2 : End of dataset

Das Reading war beim letzten Lesen zu Ende und wird nun abgeschlossen.

Rc = 3 : JOKER-Monitor not active

Der JOKER-Monitor ist noch nicht gestartet.

Rc = 4 : JOKER-Monitor just active,try later

Der JOKER-Monitor hat den Zugriff zu allen Kontrollblöcken gesperrt, da er gerade mit notwendigen Verwaltungsarbeiten beschäftigt ist.

Rc = 5 : User not joined

Dieser Benutzer ist dem JOKER-System nicht bekannt.

Rc = 6 : Exp not joined

Dieses Experiment ist dem JOKER-System nicht bekannt.

Rc = 7 : More than 10 RDG´s are not allowed

Die maximale Anzahl der Readings, aus denen der Benutzer gleichzeitig lesen kann, beträgt 10. Es wurde gerade versucht, sie zu überschreiten.

Rc = 8 : Systemerror

Fehler im JOKER-Monitor.

Rc = 9 : Record larger than record field

Das Feld ´Toaddr´ ist zu klein.
(Actlng > Maxlng)

Rc = 10 : Last record of reading

Beim letzten Record des Readings erhält man diese zusätzliche Mitteilung. Um das Reading nun ordnungsgemäß abzuschließen, sollte man das Unterprogramm R D G E N D aufrufen.

Rc = 11 : Permanent I/O error

Übertragungsfehler, die beim Lesen von der Platte aufgetreten sind.

(2) TREAD

Der Aufruf erfolgt durch:

CALL TREAD (Expno,Rdgno,Toaddr,Maxlng,Actlng,Rc,Time)

Durch dieses Unterprogramm, bei dem die ersten sechs Parameter Reihenfolge und Bedeutung genau der Parameterliste des Unterprogramms R E A D entsprechen und die nur um den letzten Parameter 'Time' erweitert wurde, kann der Benutzer eine Maximalzeit angegeben, die T R E A D auf die Erstellung eines Readings warten soll. Alle Return-Codes entsprechen denen des R E A D , insbesondere wird nach Ablauf einer erfolglosen Wartezeit auch Rc = 1 zurückgegeben.

Parameter, Bedeutung und Deklaration :

Time : Muß die maximale Wartezeit in Einheiten von hundertstel Sekunden enthalten. Diese Zeit muß kleiner sein als die im MVS-System bei der Generierung festgelegte maximale Wartezeit eines Programms.
Integer*4

(3) RDGEND

Der Aufruf erfolgt durch:

CALL RDGEND(Expno,Rdgno)
========================

Durch den Aufruf von R D G E N D wird das durch die Parameter spezifizierte Reading ordnungsgemäß abgeschlossen und der gleiche Zustand wie bei einem R E A D mit Return-Code=2 erzeugt.

Parameter, Bedeutung und Deklarationen :

Expno : Nummer des Experiments, dem die gewünschten Daten zugeordnet sind. Die Experimentnummer muß dem JOKER-System bekannt sein.
Integer*4

Rdgno : Nummer des Readings, das abgeschlossen werden soll.
Integer*4

(4) SREAD

Der Aufruf erfolgt durch:

CALL SREAD(Expno,Rdgno,Toaddr,Maxlng,Actlng,Rc,Userid)
==

Durch dieses Unterprogramm, dessen erste sechs Parameter in Reihenfolge und Bedeutung genau der Parameterliste des R E A D entsprechen und die nur um den letzten Parameter 'Userid' erweitert wurde, können Daten aus Readings anderer Experimentatoren gelesen werden.

Voraussetzung :

Der Experimentator hat erlaubt, daß seine Daten auch anderen Benutzern zugänglich gemacht werden. Dazu wird ein von ihm benannter Experiment-Slot als ´SHARED´ gekennzeichnet.

Einschränkung :

Es kann nur aus einem bereits existierenden Reading gelesen und nicht auf die Erstellung eines Readings gewartet werden.

Parameter, Bedeutung und Deklaration :

Userid : Userid des Experimentators, aus dessen Readings
--- Daten gelesen werden sollen.

Anmerkung: Diese Routine kann von jedem Benutzer auf-
--- gerufen werden. Er braucht dem JOKER-System nicht bekannt zu sein.

(5) DREAD

Der Aufruf erfolgt durch:

CALL DREAD (Expno,Rdgno,Toaddr,Maxlng,Actlng,Rc)
==========

Durch dieses Unterprogramm, dessen Parameter in Reihenfolge und Bedeutung genau der Parameterliste des R E A D entsprechen, kann der Benutzer sofort einen maximal 4000 Bytes langen Record aus der CSA übernehmen. Da die Daten nicht im Experiment-Dataset zwischengespeichert werden, ergibt sich gegenüber dem normalen Ablauf eine spürbare Zeitersparnis. Gedacht ist diese Funktion zur Abwicklung eines schnellen Dialogs (fast dialog) zwischen Experimentrechner und Zentralrechner. Die Menge der Daten je Transfer wurde auf genau einen Record von maximal 4000 Bytes pro Reading begrenzt. Da für die Weitergabe der Daten an das

Benutzerprogramm nur ein Puffer bereit steht, ist eine Synchronisation erforderlich, um das Überschreiben noch nicht an das Programm übergebener Daten zu verhindern. Die normale Anweisungsfolge in einem Programm mit D R E A D sieht etwa folgendermaßen aus:

```
          .
          .
          .
      call write                  Zur Synchronisation von
                                  Zentralrechner und Experi-
                                  mentrechner.
          .
lies: call dread                  Maximal 4000 Bytes
                                  schnell in Empfang nehmen
          .
          .                       und
          .                       auswerten.
          .
          .
      call write                  Ausgewertete Daten zum
                                  Experimentrechner zwecks
                                  weiterer Verarbeitung zu-
                                  rückschicken.
          .
      go to lies
          .
          .
      programmende
```

2.1.2 Ausgaberoutinen

(1) WRITE

Der Aufruf erfolgt durch:

```
CALL WRITE (Usrno,Expno,Fromad,Lng,Rc)
====================================
```

Mit diesem Unterprogramm schickt der Benutzer Daten vom Zentralrechner zu seinem Experimentrechner.

Parameter, Bedeutung und Deklarationen :

Usrno : Benutzer-Identifikation im JOKER-System.
----- Integer*4

Expno : Experiment-Identifikation im JOKER-System.
Integer*4

Fromad: Dimensionierte Größe beliebigen Typs (meistens Logical*1), die Byte für Byte die zu übertragende Information enthält.

Lng : Anzahl der zu übertragenden Bytes des Feldes 'Fromad' (maximal 4000 Bytes).
Integer*4

Rc : An diese Variable gibt W R I T E einen Return-Code zurück. Er gibt Auskunft über den Erfolg bei der Ausführung von W R I T E.
Integer*4

Es gibt folgende Return-Codes und Fehlermeldungen:

Rc = 0 : Übertragung zum Experimentrechner war erfolgreich.

Rc = 1 : Receiver buffer too small.

Der zur Aufnahme der Daten bereitgestellte Speicherbereich im Experimentrechner ist zu klein.

Rc = 2 : Receiver not ready.

Experimentrechner meldet sich trotz mehrerer Versuche nicht.

Rc = 3 : Permanent I/O error.

Drei Übertragungsversuche waren ohne Erfolg.

Rc = 4 : JOKER-Monitor not active

Der JOKER-Monitor ist nicht gestartet.

Rc = 5 : Unitcheck: time out

Experimentrechner-Interface ist defekt.

Rc = 10 : User not joined.

Dieser Benutzer ist dem JOKER-System nicht bekannt.

Rc = 11 : Exp not joined.

Dieses Experiment ist dem JOKER-System nicht bekannt.

Rc = 12 : More than 50 exp's are not allowed.

Um den Zugriff zu den Kontrollblöcken zu minimieren, werden beim ersten Aufruf von W R I T E alle notwendigen Informationen in eine Tabelle kopiert. Da diese 50 Entries hat, erfolgt bei mehr als 50 Experimenten diese Fehlernachricht.

Rc = 13 : No chance for getmain.

Im MVS-System ist kein Platz mehr für den Ausgabepuffer.

Rc = 14 : Record larger than record field.

Der zu übertragende Record ist größer als 4000 Bytes.

Rc = 15 : Userid and usrno are different.

Das Userid und die Benutzer-Identifikations-Nummer stimmen nicht überein.

(2) XWRITE

Der Aufruf erfolgt durch:

CALL XWRITE (Usrno,Expno,Fromad,Lng,Rc,Com)

Durch dieses Unterprogramm, dessen erste fünf Parameter in Reihenfolge und Bedeutung genau der Parameterliste des W R I T E entsprechen und die nur durch den letzten Parameter 'Com' erweitert wurde, kann der Benutzer Daten zu einem externen Experimentrechner mit einem Multi-User-Betriebssystem (z.B. RSX-11M) schicken. In diesem Fall muß dort ein spezieller Treiber installiert sein.
Dem eigentlichen Datentransfer wird ein Header vorausgeschickt, der seine Informationen aus den Parametern Usrno, Expno, Lng und Com bezieht.
Der von X W R I T E erzeugte Return-Code ist bis auf den Fall Rc = 0 um 100 größer als beim normalen W R I T E

Parameter, Bedeutung und Deklaration :

Com : Dimensionierte Größe beliebigen Typs (meistens Logical*1) , die 4 Bytes Information für den im Experimentrechner installierten Treiber enthält (z.B. Ende des Datasets) .

2.2. Löschen von Daten: USRPURG

Der Benutzer kann seine Readings, die sich in der Datenbank befinden, durch einen Prozedur-Aufruf, durch einen Batch-Job oder durch einen Unterprogrammaufruf löschen.

a.) Prozedur-Aufruf für Timesharing-Benutzer:

```
EXEC 'SYS4.JOKLNK2(USRPURG) E(Expno) RF(Rdgnof)
                            RL(Rdgnol) RA(All)'
```

b.) Batch-Job:

```
//      Jobkarte
//JOBLIB   DD DSN='SYS4.JOKLNK2',DISP=SHR
//         EXEC PGM=USRPURG,PARM='E(Expno) RF(Rdgnof)    *
//         RL(Rdgnol) RA(All)'
//SYSPRINT DD SYSOUT=A
/*       EOJ
```

Parameter:

Expno :	Experiment-Identifikations-Nummer
Rdgnof:	Nummer des ersten zu löschenden Readings
Rdgnol:	Nummer des letzten zu löschenden Readings
All	Alle Readings der angegebenen Experiment-Identifikations-Nummer werden gelöscht.

Es sind nur folgende Parameterkombinationen möglich:

(1)	...E(10) RF(1)	Das Reading mit der Identifikations-Nummer 1 der Experiment-Identifikations-Nummer 10 wird gelöscht.
(2)	...E(2) RF(2) RL(4)	Die Sequenz der Readings mit den Identifikations-Nummern 2 - 4 der Experiment-Identifikations-Nummer 2 wird gelöscht.
(3)	...E(1) RA(all)	Alle Readings mit der Experiment-Identifikations-Nummer 1 werden gelöscht.

Wenn die Readings ordnungsgemäß aus der Datenbank entfernt worden sind, wird die folgende Nachricht ausgedruckt:

Rdg(s) deleted

Folgende Fehler werden angezeigt :

(1) RDGxxxxx not found. (xxxxx = RDG-Nr.)

Das zu löschende Reading existiert nicht.

(2) False parameter or syntax check.

Beim Überprüfen der Parameter wurde ein Fehler entdeckt.

Dieses Utility kann auch als Unterprogramm aufgerufen werden:

CALL PURG (Expno,Rdgnof,Rdgnol,Rdgnoa,Rc)

Alle Parameter haben die Spezifikation Integer*4

Bedeutung des letzten Parameters Rc:

Rc = Return-Code: 0 Rdg(s) deleted
128 Fehler

2.3 Statusinformationen

2.3.1 Systemzustand: USRSTAT

Mit dieser Prozedur kann sich der Benutzer Informationen über den Zustand des JOKER-Systems, der vom JOKER-Monitor für ihn gestarteten Jobs und seines Experiments beschaffen.

Prozedur-Aufruf für Timesharing-Benutzer:

```
EXEC 'SYS4.JOKLNK2(USRSTAT) Parm'
```

Parameter:

Als Parametereingabe ´Parm´ gibt es vier Möglichkeiten:

Mon: Der Zustand des JOKER-Monitors wird ausgedruckt.

JOKER-Monitor (not) active.
Connection to PDP11 disabled.

Job: Der Zustand des vom JOKER-Monitor gestarteten
--- Jobs wird ausgedruckt.

JOB userid** executing.
JOB awaiting execution.
JOB abnormally ended.
JOB from internal reader not found.
Open of JCL dataset failed.
Allocation of JCL dataset failed reason code=xxx.
JCL dataset allocated to another task.
JCL dataset not found.
Wrong userid in jobcard of JCL dataset.
JOB not found.

Exp: Der Zustand der Verbindung zum Experimentrechner
--- wird ausgedruckt:

Line for experiment xxxx enabled.
Line for experiment xxxx disabled.

Normalerweise ist die Verbindung zum Experimentrechner durchgeschaltet. Nur wenn der Benutzer den für ihn im Zentralrechner auf der Platte reservierten Speicherplatz überschritten hat, wird die Verbindung unterbrochen.

Das Weglassen des Parameters ´Parm´ bewirkt, daß alle drei Statusinformationen ausgedruckt werden.

2.3.2 Informationen über die Datenbestände: USRRDG

Mit dieser Prozedur kann der Benutzer Informationen über seine Readings erhalten, die sich in der Datenbank befinden.

a.) Prozedur-Aufruf für Timesharing-Benutzer:

```
EXEC 'SYS4.JOKLNK2(USRRDG) U(Usrno) E(Expno) R(Rdgno)'
```

b.) Batch-Job:

```
//      Jobkarte
//JOBLIB   DD DSN='SYS4.JOKLNK2',DISP=SHR
//      EXEC PGM=USRRDG,PARM='U(Usrno) E(Expno) R(Rdgno)'
//SYSPRINT DD SYSOUT=A
/*          EOJ
```

Parameter:

Usrno : Benutzer-Identifikations-Nummer

Expno : Experiment-Identifikations-Nummer

Rdgno : Reading-Identifikations-Nummer

Diese Prozedur kann mit vier verschiedenen Parameterkombinationen aufgerufen werden.

(1)	...U(10)	Alle Informationen über Readings mit der Benutzer-Identifikations-Nummer 10 werden ausgedruckt.
(2)	...U(2) E(1)	Alle Informationen über Readings mit der Benutzer-Identifikations-Nummer 2 und der Experiment-Identifikations-Nummer 1 werden ausgedruckt.
(3)	...U(1) E(1) R(1)	Alle Informationen, die zu der Reading-Identifikations-Nummer 1 von der Experiment-Identifikations-Nummer 1 der Benutzer-Identifikatiosnummer 1 gehören, werden ausgedruckt.
(4)	ohne Parameter	wie (1)

Folgende Informationen werden angezeigt :

```
Reading opened.
Reading saved.
Reading forced.
Increment of recordnumbers >  1.
Recordnumbers out of range.
Reading contains xx pages  (xx = number)
```

Die Parameterkombination (3) kann auch als Unterprogramm aufgerufen werden:

```
CALL CKRDG(Usrno,Expno,Rdgno,Rc)
```

Alle Parameter haben die Spezifikation Integer*4.
Bedeutung des letzten Parameters Rc:

```
Rc = Return-Code: 0    Reading vollständig vorhanden.
                  1    Reading ist angelegt.
                  2    Reading ist nicht vorhanden.
                  3    Systemfehler
```

2.3.3 Übertragungsinformationen und Fehleranalyse: USRHDR

Mit dieser Prozedur kann der Benutzer Informationen über Header erhalten, deren Adressen sich noch im Header-Block befinden.

a.) Prozedur-Aufruf für Timesharing-Benutzer:

```
EXEC 'SYS4.JOKLNK2(USRHDR) E(Exphda) A(Number) C(y)'
```

b.) Batch-Job:

```
//      Jobkarte
//JOBLIB   DD DSN='SYS4.JOKLNK2',DISP=SHR
//       EXEC PGM=USRHDR,PARM='E(Exphda) A(Number) C(y)'
//SYSPRINT DD SYSOUT=A
/*         EOJ
```

Parameter:

Number: Anzahl der gewünschten Header

Exphda: Experiment-Hardware-Adresse, von der die Header geschickt wurden.

Y: : Die in den Message-Bytes enthaltenen Informationen werden erklärt.

Diese Prozedur kann mit zwei verschiedenen Parameterkombinationen aufgerufen werden.

(1)	...E(EE00) A(5)	Die wichtigsten Informationen von maximal 5 Headern mit der Experiment-Hardware-Adresse EE00 werden ausgedruckt.
(2)	...E(4100) A(3) C(y)	Die wichtigsten Informationen von maximal 3 Headern mit der Experiment-Hardware-Adresse 4100 werden ausgedruckt, und die in den Message-Bytes enthaltenen Informationen werden erklärt.

In den Message-Bytes werden folgende Informationen angezeigt:

80xx: Falsche Benutzer-Identifikations-Nummer

40xx: Falsche Experiment-Identifikations-Nummer

20xx: Falsche Experiment-Hardware-Adresse

04xx: Ein existierendes Reading mit gleicher Reading-
10xx: Nummer wurde durch diesen Record gelöscht.
14xx:

01xx: Dieser Record wurde gelöscht, weil ein Reading mit
02xx: dieser Reading-Nummer schon existierte.
03xx:

xx80: Dieser Record wurde gelöscht, weil er nicht die Record-Nummer 1 hatte und kein offenes Reading mit dieser Reading-Nummer existierte.

xx10: Falscher Funktions-Code

2.4 Spezielle Anwendungen

2.4.1 Ausgabe eines Textes

Wegen der beschränkten Experimentrechnerperipherie ist es in vielen Fällen unbequem, umfangreichere Texte am Experimentrechner ausdrucken zu lassen. Deshalb kann der Benutzer diese Datasets über das JOKER-System zum Zentralrechner schicken. Durch die Prozedur U S R L S T x werden diese Datasets dann vom Zentralrechner ausgedruckt. Für x ist 8 oder 11 einzusetzen, je nachdem, ob es sich um Datasets von einer PDP8 oder einer PDP11 handelt, wobei 11 auch für Datasets eines anderen Experimentrechnertyps mit 16 Bit ASCII-Code benutzt werden kann.

a.) Prozeduraufruf für Timesharing-Benutzer:

```
EXEC 'SYS4.JOKLNK2(USRLSTx) E(Expno) RF(Rdgnof)
                            RF(Rd RL(Rdgnol) RP(Func)'
```

b.) Batch-Job:

```
//     Jobkarte
//JOBLIB   DD DSN='SYS4.JOKLNK2',DISP=SHR
//        EXEC PGM=USRLSTx,PARM='E(Expno) RF(Rdgnof)     *
//        RL(Rdgnol) RP(Func)'
//SYSPRINT DD SYSOUT=A
/*        EOJ
```

Parameter:

Expno : Experiment-Identifikations-Nummer

Rdgnof: Erstes oder einziges auszudruckendes Reading

Rdgnol: Letztes auszudruckendes Reading

Func: = y Die Readings werden anschließend gelöscht.

n Die Readings werden anschließend nicht gelöscht.

2.4.2 Funktionssteuerung durch C R E A T E

Es hat sich gezeigt, daß die Benutzer des JOKER-Systems daran interessiert sind, eine Reihe von 'Standard'-Dienstleistungen in Anspruch nehmen zu können, ohne sich eines Timesharing-Terminals bedienen zu müssen.

Zu diesen Dienstleistungen gehören z.B.:

- das Auslagern und Löschen von Daten aus einem Experiment-Dataset,
- das Ausdrucken von auf einem Experimentrechner erstellten Texten,
- das Abrufen von Datensätzen aus dem Zentralrechner.

Um dem Wunsch, von einem Experimentrechner aus verschiedene Dienstleistungen in Anspruch nehmen zu können, gerecht zu werden, wurde das Programmpaket CREATE /3/ erstellt, das von jedem gestarteten Job aus aufgerufen und durch zusätzlich übertragene Parameter gesteuert werden kann. Aufgrund eines vom Benutzer in einem sogenannten CREATE-Reading gesendeten Funktionsindikators wird die auszuführende Funktion vom Programm CREATE erkannt, und die entsprechenden Aktionen werden eingeleitet.

Es gibt 2 Typen von Funktionen, die durch CREATE ausgeführt werden können:

1.) Standard-Funktionen
Diese Funktionen werden systemseitig zur Verfügung gestellt und stehen jedem Benutzer, der den entsprechenden Funktionsindikator absetzt, zur Verfügung. Für diese Standard-Funktionen sind die Kennzahlen 1 bis 49 reserviert.

2.) Benutzerspezifische Funktionen
Hierbei hat jeder Benutzer die Möglichkeit, für jeden der Funktionsindikatoren (das sind die Kennzahlen 50 bis 99) eine eigene, ihm individuell zugeordnete Funktion zu definieren.

Das CREATE-Programm wird durch ein 'allgemeines' Reading gestartet. Anschließend wird ein zuvor übertragenes CREATE-Reading gelesen und die darin spezifizierte Funktion ausgeführt.

2.5. Literaturhinweise

/1/ D. Conrads, H.E. Moritz, R. Mühlstroh:
JOKER - Ein System zur Kopplung von Experimentrechnern verschiedener Fabrikate mit einem zentralen Time-sharing-Rechner,
Kernforschungsanlage Jülich , JÜL - 1004 - MA , 1973

/2/ W. Boenke, N. Dick:
Der JOKER-Monitor für das Betriebssystem MVS
-Systembeschreibung-
Kernforschungsanlage Jülich , JÜL - 1629 - MA , 1979

/3/ J. Docter, H.Peters, S. Rafflenbeul:
CREATE - Ein Programmpaket zur Steuerung von Großrechner-Funktionen über das JOKER-Netz,
Kernforschungsanlage Jülich , JÜL - SPEZ 56 , 1979

/4/ IBM System/360 Time Sharing System (TSS)
Concepts and Facilities,
IBM Form C28-2003-4, January 1970

/5/ OS/VS2 MVS Overview,
IBM Form GC28-0984, July 1978