KERNFORSCHUNGSZENTRUM

KARLSRUHE

August 1976 KFK 2317

Institut für Neutronenphysik und Reaktortechnik
Projekt Schneller Brüter

Das Karlsruher Programmsystem KAPRØS
Teil Ia
Kurzes KAPRØS-Benutzerhandbuch

H. Bachmann, S. Kleinheins

GESELLSCHAFT
FÜR
KERNFORSCHUNG M.B.H.

KARLSRUHE

GESELLSCHAFT FÜR KERNFORSCHUNG M. B. H.

KARLSRUHE

KERNFORSCHUNGSZENTRUM KARLSRUHE

KFK 2317

Institut für Neutronenphysik und Reaktortechnik

Projekt Schneller Brüter

Das Karlsruher Programmsystem KAPRØS
Teil Ia
Kurzes KAPRØS-Benutzerhandbuch

H. Bachmann, S. Kleinheins

Gesellschaft für Kernforschung mbH, Karlsruhe

Zusammenfassung

Die Regeln zur Handhabung des Karlsruher modularen Programmsystems KAPRØS sind in knapper Form für Benutzer von KAPRØS und für Programmierer von KAPRØS-Moduln zusammengestellt. Dabei werden die Syntax der KAPRØS-Eingabe, die Interpretation der Ausgabe, die Aufrufe der Systemroutinen und die Entschlüsselung ihrer Fehlermeldungen, die Anforderungen, denen die Moduln genügen müssen, und schließlich das Starten von KAPRØS-Jobs auf einer IBM/370-168 beschrieben.

Short KAPRØS Users' Manual

Summary

The rules for handling the Karlsruhe modular program system KAPRØS are compiled in a concise manner for the needs of KAPRØS users and KAPRØS module programmers. The syntax of the KAPRØS input, the interpretation of the output, the system routine calls and the decoding of the error messages, the requirements to which modules are subjected, and the submission of KAPRØS jobs on an IBM/370-168 are described.

Überblick über die Dokumentation des Karlsruher Programmsystems KAPRØS

Teil I: Übersicht und Vereinbarungen.
Einführung für Benutzer und Programmierer.
KFK 2253 (1976)

(Allgemeine Einführung in KAPRØS; Beschreibung für Benutzer und Programmierer von KAPRØS-Moduln.)

Teil Ia: Kurzes KAPRØS-Benutzerhandbuch.
KFK 2317 (1976)

(Zusammenfassung der Regeln für die Benutzung von KAPRØS.)

Teil II: Dokumentation des Systemkerns.
KFK 2254 (1976)

(Beschreibung des Systemkerns für Systemprogrammierer.)

Teil III: Kurzbeschreibungen der KAPRØS-Moduln.
KFK-Bericht in Vorbereitung.

(Beschreibung von existierenden KAPRØS-Moduln für Benutzer.)

Inhalt

		Seite
1.	Einleitung	1
2.	KAPRØS-Eingabe	2
2.1	Variable Größen der KAPRØS-Eingabe	3
2.2	Teil 1 der KAPRØS-Eingabe (*CØMPILE- und *LINK-Eingabe)	4
2.3	Teil 2a der KAPRØS-Eingabe (*KSIØX-Eingabe)	7
2.4	Teil 2b der KAPRØS-Eingabe (*GØ-Eingabe)	11
3.	KAPRØS-Ausgabe	12
4.	Systemroutinen	15
4.1	KSINIT (Modul-Initialisierung)	15
4.2	KSPUT, KSGET, KSCH, KSDLT, KSPUTP, KSGETP, KSCHP (Schreiben, Lesen usw. von Datenblöcken)	15
4.3	KSEXEC (Aufruf von Moduln)	19
4.4	KSDD (Anlegen oder Freigeben von Puffern)	21
4.5	KSDAC (Charakteristiken von Direct-Access-Dateien)	23
4.6	KSARC (Archivieren von Datenblöcken)	23
4.7	KSMØVE (Einflußnahme auf Datenblock-Speicherung)	24
4.8	KSLADY und KSLØRD (Einflußnahme auf Modul-Speicherung)	25
4.9	KSCC (Behandlung von Fehler- und Nachrichtencodes)	27
4.10	KSDUMP (Erzeugen von KAPRØS-Dumps)	27
4.11	Fehlercodes	28
4.12	Fehlermeldungen und Warnungen	34
4.13	Completion Codes	34
5.	Anforderungen an KAPRØS-Moduln	35
5.1	Allgemeines	35
5.2	Prüfmoduln	37
5.3	Druckmoduln	38
5.4	Lesemoduln	38
5.5	Steuermoduln	39
6.	Benutzungshinweise	40
6.1	JCL-Karten für KAPRØS-Jobs	40
6.2	Regiongröße	42
6.3	Aufnahme von Moduln in die KAPRØS-Modulbibliothek	43
6.4	Benutzerarchive	44
6.5	Programmkonstanten und -beschränkungen	44
Anhang:	Einige Definitionen und Abkürzungen	45
Literatur		50

1. Einleitung

Dieser Bericht enthält in knapp zusammengefaßter Form die Regeln für die Handhabung des Systemkerns des Karlsruher Programmsystems KAPRØS, Stand Frühjahr 1976 (Implementierung auf der Rechenanlage IBM/370-168 des Kernforschungszentrums Karlsruhe, Betriebssystem ØS-MVT). Es wurde versucht, alles zusammenzutragen, was ein Benutzer von KAPRØS bei der Durchführung von Rechnungen und ein Programmierer bei der Erstellung von KAPRØS-Moduln wissen und beachten muß. Im einzelnen sind das die Syntax der KAPRØS-Eingabe, die Interpretation der Ausgabe, die Aufrufe der Systemroutinen und die Entschlüsselung ihrer Fehlermeldungen, die Konventionen, denen die Moduln genügen müssen, und das Starten von KAPRØS-Jobs. Der Systeminhalt, d. h. die in der Modulbibliothek verfügbaren KAPRØS-Moduln, wird dagegen im vorliegenden Bericht nicht beschrieben.

Es muß betont werden, daß der Bericht nicht als Einführung in KAPRØS vorgesehen ist und die ausführliche KAPRØS-Dokumentation /2,3/ nicht ersetzen kann und will. Vielmehr ist er als kurze Arbeitsunterlage zum Nachschlagen für Anwender gedacht, die sich auf andere Weise, z. B. durch Lesen der einführenden und vollständigen Beschreibung /2/, schon mit dem System vertraut gemacht haben. Dort findet der Leser neben einer ausführlichen Erklärung der in KAPRØS verwendeten Begriffe (die hier im Anhang der Vollständigkeit halber nochmals kurz zusammengestellt sind) auch Beispiele vor.

Das vorliegende Handbuch ist eine ergänzte und verbesserte Fassung eines internen Berichts von 1974. Auch für die Zukunft ist vorgesehen, Änderungen des Systemkerns, die für Anwender relevant sind, durch Herausgabe von Nachträgen zu diesem Handbuch oder durch Neufassungen zu dokumentieren.

Zum Druck eingereicht am 5.7.1976

2. KAPRØS-Eingabe

Die KAPRØS-Eingabe besteht aus den folgenden Teilen in der angegebenen Reihenfolge:

Teil 1: Testmoduln
Teil 2a: Spezifizierung der Externblöcke; Blockdaten
Teil 2b: Spezifizierung des Steuermoduls

Jeder dieser Teile kann fehlen.

Die Gliederung und Steuerung der Eingabe erfolgt durch KAPRØS-Steueranweisungen vom Format

*Operation [Operand] [Kommentar] [1)]

mit den Operationen CØMPILE, LINK, KSIØX und GØ.
Außerdem gibt es die Endeanweisung

*$*$ [Kommentar].

Die Steueranweisungen stehen auf KAPRØS-Steuerkarten (ab Spalte 1), die von der Standardeingabe eingelesen werden. Eine Steueranweisung kann sich über mehrere Steuerkarten erstrecken. Wenn auf den letzten Parameter im Operanden einer Steuerkarte ein Komma folgt, erwartet das KSP eine Fortsetzungskarte. Eine Fortsetzungskarte muß mit *$b beginnen und darf nicht leer sein. Für die Fortsetzung einer Fortsetzungskarte gilt das gleiche. Eine Steueranweisung gilt als abgeschlossen, wenn auf den letzten Parameter mindestens ein Blank (im Falle von Blocknamen mindestens ein Blank nach dem 16. Zeichen des Blocknamens) oder die Spalte 72 folgt. Hinter dem abschließenden Blank oder ab Spalte 72 stehende Zeichen werden als Kommentar betrachtet. Mit *$b beginnende Karten, die nicht als Fortsetzungskarten erwartet werden, sind Kommentarkarten. Kommentarkarten und Leerkarten können beliebig zwischen anderen Karten (auch Blockdatenkarten) stehen.

Anm.: Hinter der KAPRØS-Eingabe kann auf der Standardeingabe moduleigene Eingabe folgen.

---

1) In eckigen Klammern [ ] stehender Text darf weggelassen werden; Alternativen werden durch einen senkrechten Strich | getrennt.

2.1 Variable Größen der KAPRØS-Eingabe

a) (Einfache) Blocknamen bestehen aus bis zu 16 alphanumerischen Zeichen oder Blanks, deren erstes Zeichen ein alphabethisches sein muß. Blanks sind signifikante Zeichen!

b) Modulnamen bestehen aus bis zu 6 alphanumerischen Zeichen, deren erstes ein alphabetisches sein muß.

c) Spezifikationen bestehen aus bis zu 24 alphanumerischen Zeichen oder Punkten, deren erstes Zeichen ein alphabetisches sein muß. Eine vollständige Spezifikation setzt sich zusammen aus

Jobname Startdatum Startzeit

oder aus

FTnn Kennzeichen Startdatum Startzeit,

ohne Zwischenraum hintereinander geschrieben. Dabei gilt:

Jobname	=	8 Zeichen; z.B. INR308KS
Startdatum	=	8 Zeichen; z.B. 19.05.74
Startzeit	=	8 Zeichen; z.B. 23.59.59
Kennzeichen	=	4 Zeichen; z.B. 6M2G

Anm.: Alphabethische Zeichen sind A,...,Z,$.

d) Index, Satzzahl, n, nn und mm sind positive ganze Zahlen. (Indices sollen aus Speicherplatzgründen so klein wie möglich gewählt werden).

e) Formatfreie Blockdaten werden in einer der folgenden Schreibweisen angegeben (Definition und Speicherung wie in Fortran):

Integer-Konstanten, z.B.	1 -30 +466
Real-Konstanten , z.B.	1.1 -3.14 +0.1E-6
Real*8-Konstanten , z.B.	1.1D+0 -3.14D+0 +0.1D-6
Hexadezimalkonstanten,z.B.	Z3340 ZCF317 ZABC
Complex-Konstanten, z.B.	(1.34,4.17) (+1.6,-0.7E-3)
Complex*16-Konstanten,z.B.	(+1.7D+3, -0.6E-4)
Literalkonstanten,z.B.	'ABCDE'

Außerdem ist noch eine Schreibweise für Literalkonstanten möglich, bei der jeweils 5 Zeichen, rechts aufgefüllt mit 3 Blanks, in ein Doppelwort gefüllt werden, z.B.

@ABCDE@

Ferner ist die Wiederholung einer Konstanten möglich, z.B.

100 *0.1E-6

Auf einer Blockdatenkarte können beliebig viele Blockdaten stehen, jeweils durch mindestens ein Blank getrennt. Eine Blockdatenkarte wird von Spalte 1 bis 71 einschließlich interpretiert.
Die Kombinationen ƀ,ƀ und ƀ.ƀ auf einer Blockdatenkarte werden als Literalkonstante 'KSKS' interpretiert (zur Strukturierung der Blockdaten).

2.2 Teil 1 der KAPRØS-Eingabe

Mit den *CØMPILE- und *LINK-Anweisungen wird das Compilieren oder Assemblieren und Linken der Testmoduln gesteuert. Sind keine Testmoduln in der Eingabe vorhanden, so werden alle im KAPRØS-Job benötigten Moduln in der Modulbibliothek gesucht.
Formate:

*CØMPILE A|G|H[,UNIT=nn][,Subparameter]
*LINK [Subparameter]

Wenn der UNIT-Parameter vorhanden ist, muß er an der angegebenen Stelle stehen.
Mit A,G oder H wird der Assembler H, der Fortran G1 Compiler oder der Fortran H Extended Compiler spezifiziert. Als weitere Parameter können die Subparameter von Assembler oder Compiler und Linkage Editor angegeben werden. Wegen des UNIT-Parameters s.u.

a) Für jeden aus Assembler- und/oder Fortran-Routinen zusammengesetzten Testmodul benötigt man die folgende Eingabe:

```
*CØMPILE A,LØAD,NØDECK[,weitere Subparameter]
Quellkarten aller Assembler-Routinen
*$*$
*CØMPILE G|H[,Subparameter]
Quellkarten aller Fortran-Routinen
*$*$
*LINK MAP,LIST[,weitere Subparameter]
[Eingabekarten des Linkage Editors]
bENTRY  Name der Hauptroutine
bNAME   Name des Testmoduls
*$*$
```

Stehen die Quellkarten einer oder mehrerer Routinen auf Dateien, die durch DD-Karten mit den DD-Namen FTnnFyyy,...,FTmmFyyy beschrieben sind, so ist die obige *CØMPILE-Eingabe abzuwandeln:

```
*CØMPILE A,UNIT=nn,LØAD,NØDECK[,weitere Subparameter]
[Quellkarten weiterer Assembler-Routinen]
*$*$
*CØMPILE G|H,UNIT=mm[,Subparameter]
[Quellkarten weiterer Fortran-Routinen]
*$*$
......
```

Anm.: Auf einer Datei können jeweils nur Assemblerroutinen oder nur Fortran-Routinen stehen. Der Inhalt einer Datei wird als Ganzes dem Assembler oder Compiler zugeführt. Die Dateinummern nn,...,mm dürfen nicht 40,...,50 sein.

Stehen die Eingabekarten des Linkage Editors (evtl. einschl. der ENTRY-Karte) auf einer Datei, die durch eine DD-Karte mit einem beliebigen DD-Namen beschrieben ist, so ist die obige *LINK-Eingabe abzuwandeln:

```
.....
*LINK MAP,LIST[,weitere Subparameter]
bINCLUDE ddname[(Membername)]
[bENTRY Name der Hauptroutine]
bNAME  Name des Testmoduls
*$*$
```

b) Steht ein Testmodul als Load-Modul auf einer (partitioned) Datei, die durch eine DD-Karte mit einem beliebigen DD-Namen beschrieben ist, so entfällt die *CØMPILE-Eingabe und es bleibt

```
*LINK MAP,LIST[,weitere Subparameter]
bINCLUDE ddname(Membername)
[Eingabekarten des Linkage Editors]
bENTRY Name der Hauptroutine
bNAME  Name des Testmoduls
*$*$
```

Die Eingabekarten des Linkage Editors (benötigt z.B., wenn der Modul Overlay-Struktur hat) können wie bei a) auch von einer Datei kommen. Ferner können durch eine *CØMPILE-Eingabe wie bei a) Routinen zum Load-Modul dazugelinkt oder Routinen des Load-Moduls ersetzt werden. Falls ein in der KAPRØS-Modulbibliothek LØAD.KSBI (s.6.3) stehender Modul auf diese Weise modifiziert als Testmodul in den KAPRØS-Job eingebracht werden soll, sieht die Eingabe wie folgt aus:

```
.....
*LINK MAP,LIST[,weitere Subparameter]
bINCLUDE KSBIB(Membername des Moduls)
bINCLUDE ddname(Membername der Overlaystruktur-Karten)
bENTRY  Name der Hauptroutine
bNAME   Name des Testmoduls
*$*$
```

Anm.: Wie in den obigen Mustern gezeigt, sollte pro *LINK-Anweisung nur e i n Testmodul definiert werden.

2.3 Teil 2a der KAPRØS-Eingabe

Mit den *KSIØX-Anweisungen werden die Externblöcke spezifiziert.
Format:

*KSIØX DBN=Blockname[,weitere Parameter]

Abgesehen vom DBN-Parameter ist die Stellung der Parameter beliebig.

a) Für jeden Karteneingabe-DB benötigt man die Eingabe:

*KSIØX DBN=Blockname[,IND=Index],TYP=CARD,
PM|PMN=Prüfmodulname|KETT[,LM|LMN=Lesemodulname]
Blockdatenkarten des DB
[*$ *$]

Stehen die Blockdatenkarten (einschließlich der Endekarte, falls vorhanden) auf einer sequentiellen Datei, die durch eine DD-Karte mit dem DD-Namen FTnnFyyy beschrieben ist, so benötigt man die Steueranweisung:

*KSIØX DBN=Blockname[,IND=Index],TYP=CARD,UNIT=nn,
PM|PMN=Prüfmodulname|KETT[,LM|LMN=Lesemodulname]

Wenn der LM- oder LMN-Parameter fehlt, werden die Blockdaten vom KSP formatfrei (s.2.1.e) bis zur Endekarte einschließlich (oder bis zur nächsten Steuerkarte ausschließlich) eingelesen und in die Lifeline geschrieben. Wenn der LM- oder LMN-Parameter angegeben ist, müssen die Blockdaten (einschließlich einer evtl. vorhandenen Endekarte) vom Lesemodul (s.5.4) eingelesen und in die Lifeline geschrieben werden.
Wegen des PM- oder PMN-Parameters s.u.

Anm.: Auf einer Datei können die Blockdatenkarten mehrerer Karteneingabe-DB stehen. Formatfreie Blockdaten, die vom KSP eingelesen werden sollen, müssen auf Dateien jeweils mit Endekarten abgeschlossen werden.

b) Für jeden Druckausgabe-DB benötigt man die Steueranweisung:

*KSIØX DBN=Blockname[,IND=Index],TYP=PRINT,
PM|PMN=Druckmodulname|KETT

Wegen des PM- oder PMN-Parameters s.u.

c) Für jeden Archiveingabe-DB aus dem Generellen Archiv benötigt man die Steueranweisung:

```
*KSIØX  DBN=Blockname[,IND=Index],TYP=ARCI
        [,SPEC=Jobname[Startdatum[Startzeit]]]
        [,DBNA=Alter Blockname] [,INDA=Alter Index]
        [,PM|PMN=Prüfmodulname|KETT]
```

Für jeden Archiveingabe-DB aus einem Benutzerarchiv, das durch eine DD-Karte mit dem DD-Namen FTnnFyyy beschrieben ist,benötigt man die Steueranweisung:

```
*KSIØX  DBN=Blockname[,IND=Index],TYP=ARCI,
        SPEC=FTnn[Kennzeichen[Startdatum[Startzeit]]]
        [,DBNA=Alter Blockname] [,INDA=Alter Index]
        [,PM|PMN=Prüfmodulname|KETT]
```

Im SPEC-Parameter wird Jobname, Startdatum und Startzeit des KAPRØS-Jobs angegeben, der den gesuchten DB ins Generelle Archiv schrieb, bzw. außer Startdatum und Startzeit ein Kennzeichen, unter dem der gesuchte DB in das Benutzerarchiv geschrieben wurde. Im DBNA- und im INDA-Parameter wird der Blockname und der Index angegeben,unter dem der DB im Archiv gesucht werden soll. Wenn der DB mehrfach im Archiv steht, gilt der zeitlich zuletzt geschriebene. Wenn der SPEC-Parameter unvollständig ist oder fehlt, wird der letzte im Archiv stehende DB (mit dem angegebenen Blocknamen und Index) gesucht, auf den die Spezifikation, soweit angegeben, zutrifft. (Die vollständige Spezifikation der gefundenen DB wird ins Protokoll gedruckt.) Wenn der DBNA- und/oder der INDA-Parameter fehlt, wird der DB unter dem im DBN-Parameter angegebenen Blocknamen und/oder dem im IND-Parameter angegebenen Index im Archiv gesucht. Wegen des PM- oder PMN-Parameters s. u.

d) Für jeden Archivausgabe-DB für das Generelle Archiv benötigt man die Steueranweisung:

```
*KSIØX  DBN=Blockname[,IND=Index],TYP=ARCØ
        [,PM|PMN=Druckmodulname|KETT]
```

Für jeden Archivausgabe-DB für ein Benutzerarchiv, das durch eine DD-Karte mit dem DD-Namen FTnnFyyy beschrieben ist, benötigt man die Steueranweisung:

```
*KSIØX  DBN=Blockname[,IND=Index],TYP=ARCØ,
        SPEC=FTnn[Kennzeichen] [,PM|PMN=Druckmodulname|KETT]
```

Wegen des PM- oder PMN-Parameters s.u.

e) Für jeden Alten Restart-DB benötigt man die Steueranweisung:

```
*KSIØX  DBN=Blockname[,IND=Index],TYP=RESI
        [,SPEC=Jobname[Startdatum[Startzeit]]]
        [,DBNA=Alter Blockname] [,INDA=Alter Index]
        [,PM|PMN=Prüfmodulname|KETT]
```

Im SPEC-Parameter wird Jobname, Startdatum und Startzeit des KAPRØS-Jobs angegeben, der den gesuchten DB in die Restart-Lifeline schrieb. Im DBNA- und im INDA-Parameter wird der Blockname und der Index angegeben, unter dem der DB in der Restart-Lifeline gesucht werden soll. Wenn der DB mehrfach in der Restart-Lifeline steht, gilt der zeitlich zuletzt geschriebene. Wenn der SPEC-Parameter unvollständig ist oder fehlt, wird der letzte in der Restart-Lifeline stehende DB (mit dem angegebenen Blocknamen und Index) gesucht, auf den die Spezifikation, soweit angegeben, zutrifft. (Die vollständige Spezifikation der gefundenen DB oder Teil-DB wird ins Protokoll gedruckt.) Wenn der DBNA- und/oder der INDA-Parameter fehlt, wird der DB unter dem im DBN-Parameter angegebenen Blocknamen und/oder dem im IND-Parameter angegebenen Index in der Restart-Lifeline gesucht.
Wegen des PM- oder PMN-Parameters s. u.

f) Für jeden Neuen Restart-DB benötigt man die Steueranweisung:

```
*KSIØX  DBN=Blockname[,IND=INDEX],TYP=RESØ
        [,PM|PMN=Prüfmodulname|KETT]
```

Wegen des PM- oder PMN-Parameters s.u.

Anm.: Anstelle von TYP=RESI kann (wie bisher) auch TYP=REA m i t Angabe des SPEC-Parameters geschrieben werden; anstelle von TYP=RESØ kann auch TYP=REA geschrieben werden. Ein Neuer Restart-DB ist vom Startzeitpunkt des KAPRØS-Jobs ab 7 Tage lang anderen Jobs als Alter-Restart-DB zugänglich.

g) Für jeden Scratch-DB benötigt man die Steueranweisung:

*KSIØX DBN=Blockname[,IND=INDEX] [,TYP=SCDB]
[,PM|PMN=Prüfmodulname|KETT]

Wegen des PM- oder PMN-Parameters s.u.

Anmerkungen:

α) Wenn der IND-Parameter fehlt, wird IND=1 angenommen.

β) Der PM- oder PMN-Parameter m u ß bei Karteneingabe- und Druckausgabe-DB vorhanden sein; bei Archiv-, Restart- und Scratch-DB k a n n er vorhanden sein. Bei Karteneingabe-, Archiveingabe-, Restart- und Scratch-DB gibt er den Namen des Prüfmoduls an (s.5.2); bei Druckausgabe- und Archivausgabe-DB gibt er den Namen des Druckmoduls an (s. 5.3). Wenn statt eines Modulnamens das Schlüsselwort KETT steht, so wird der DB vom gleichen Prüf- oder Druckmodul verarbeitet, wie der DB auf der unmittelbar folgenden *KSIØX-Anweisung, und zwar mit diesem verkettet. Durch mehrfache Verwendung des Schlüsselwortes KETT können beliebig viele DB verkettet von einem Prüf- oder Druckmodul verarbeitet werden (s. 5.2 und 5.3). Dabei können Karteneingabe-, Archiveingabe-, Restart- und Scratch-DB gemischt von einem Prüfmodul verarbeitet werden; Druckausgabe- und Archivausgabe-DB können gemischt von einem Druckmodul verarbeitet werden.

γ) Auf den *KSIØX-Anweisungen aller Typen kann zusätzlich noch der MØDQ-Parameter angegeben werden:

[,MØDQ=n,Modulname$_1$,...,Modulname$_n$]

Er gibt an, daß der betr. DB nur für die aufgeführten Moduln qualifiziert ist, d.h. nur den aufgeführten Moduln zugänglich sein soll.
(Anm.: Man vergesse nicht, einen DB ggf. auch für den Prüf-, Druck- oder Lesemodul zu qualifizieren.)

δ) Die Dateinummern nn dürfen nicht 40,...,50 sein.

2.4 Teil 2b der KAPRØS-Eingabe

Die *GØ-Anweisung schließt die KAPRØS-Eingabe ab und spezifiziert den Steuermodul (s. 5.5). Fehlt die *GØ-Anweisung, so wird die KAPRØS-Eingabe durch eine Endekarte oder durch Dateiende (EØF) auf der Standardeingabe abgeschlossen; der KAPRØS-Job wird dann nach der Eingabeprüfung abgebrochen.

Format:

*GØ SM=Steuermodulname[,RL=Satzzahl] [,ML=0|1|2|3]

Im RL-Parameter wird die Anzahl der auf der Restart-Lifeline zu reservierenden Sätze (zu je 766 Worten) angegeben.

Anm.: Es ist durchaus möglich, DB in die Restart-Lifeline zu schreiben, ohne daß dort vorher Platz reserviert wurde; man riskiert dann jedoch, daß bei Platzmangel in der Restart-Lifeline der KAPRØS-Job mittendrin abgebrochen werden muß, während er andernfalls schon nach der Eingabeprüfung abgebrochen würde. Evtl. zuviel reservierter Platz wird am Ende des KAPRØS-Jobs wieder freigegeben.

Im ML-Parameter wird angegeben, ob alle KAPRØS-Mitteilungen ins Protokoll gedruckt werden sollen (ML=0), ob KAPRØS-Nachrichten unterdrückt werden sollen (ML=1), ob KAPRØS-Nachrichten und -Warnungen unterdrückt werden sollen (ML=2) oder ob KAPRØS-Nachrichten, -Warnungen und -Fehlermeldungen unterdrückt werden sollen (ML=3). Wenn der ML-Parameter fehlt, wird ML=0 angenommen.

Anm.: KAPRØS-Mitteilungen, die die Verarbeitung der Eingabe betreffen, die die Aufnahme von DB in die Restart-Lifeline oder in ein Archiv bestätigen oder die die Beendigung des KAPRØS-Jobs betreffen, werden unabhängig vom Wert des ML-Parameters ins Protokoll gedruckt.

3. KAPRØS-Ausgabe

Die KAPRØS-Ausgabe besteht aus dem Protokoll, das sich wie folgt aufbaut:

a) Im Kopf des Protokolls werden neben der Überschrift "KAPRØS (KARLSRUHER PRØGRAMM-SYSTEM)" sowie Versionsnummer, Literatur- und anderen Hinweisen der Jobname, das Startdatum und die Startzeit des KAPRØS-Jobs ausgedruckt. Diese 3 Angaben (die mit der Überschrift auch im Kopf der Standardausgabe ausgedruckt werden) sind maßgebend zur Identifizierung von Restart- und Archiv-DB, die von dem KAPRØS-Job erstellt werden, wenn sie in späteren KAPRØS-Jobs wieder benötigt werden. Anschließend wird die KAPRØS-Eingabe protokolliert. Um die Eingabe von evtl. KAPRØS-Mitteilungen abzuheben, beginnt der Ausdruck der Eingabekarten auf Druckposition 11; die Spalte 72 der Eingabekarte, ab welcher deren Inhalt als Kommentar interpretiert wird, wird mit einem Schrägstrich / überdruckt.

b) Während der Verarbeitung der Eingabe können vom KSP KAPRØS-Mitteilungen ins Protokoll gedruckt werden; ebenso später von den von den Moduln aufgerufenen Systemroutinen. Es gibt 3 Arten von Mitteilungen:

   α) KAPRØS-Nachrichten: "KS-NACHRICHT:..."
   β) " -Fehlermeldungen: "KS-FEHLER....."
   γ) " -Warnungen: "KS-WARNUNG....."

   Die Mitteilungen werden ab Druckposition 1 ausgedruckt.

   Die meisten KAPRØS-Nachrichten sind selbsterklärend. Mit einer Nachricht wird z. B. das Anlaufen und das Abschließen jedes Moduls gemeldet. Die dabei ausgedruckten CPU- und Verweilzeiten gelten für die Zeiträume, in denen der betr. Modul aktiv war (ohne die Zeiten, in der evtl. gerufene Moduln höherer Stufe aktiv waren). Mit einer Nachricht wird auch das Schreiben eines DB in ein Archiv und das Schreiben und Ändern jedes Teil-DB in der Restart-Lifeline bestätigt. Dabei bedeuten die ausgedruckten Daten:

Blockname, Index, ndb bzw.
Blockname, Index, kdb, izw

mit ndb = Gesamtlänge des DB, kdb = Relativadresse des Teil-DB, izw = Wortzahl des Teil-DB. Blockname und Index sind gleich demjenigen einfachen Blocknamen und Index, unter dem der DB in das Archiv bzw. die Restart-Lifeline geschrieben wurde. Im Falle von Archivausgabe- und von Neuen Restart-DB ist das immer der Externblockname des DB; im Falle der Archivierung mit der Systemroutine KSARC ist das der im KSARC-Aufruf verwendete Blockname. Mit einer Nachricht, die die obigen Daten und die vollständige Spezifikation des DB enthält, wird auch der Anschluß eines unvollständig spezifizierten Archiveingabe-DB und eines unvollständig spezifizierten Alten Restart-DB an den KAPRØS-Job gemeldet (s. 2.3.c und e).

KAPRØS-Fehlermeldungen und -Warnungen sind selbsterklärend, wenn sie vom KSP ausgedruckt werden. Wenn sie von den Systemroutinen ausgedruckt werden, sind sie gemäß (4.12) zu entschlüsseln.

c) Nach fehlerfreiem Abschluß eines KAPRØS-Jobs oder nach Jobabbruch wegen eines Fehlers werden unter der Überschrift "KS-JØB-STATISTIK" die wichtigsten Daten des Jobs ins Protokoll gedruckt (und in dieser Form auch auf die Jobstatistik-Datei von KAPRØS gebracht). Die einzelnen Rubriken sind folgendermaßen zu interpretieren:

JØB-NAME	Jobname } des KAPRØS-Jobs;
ST-DATUM	Startdatum } des KAPRØS-Jobs;
ST-ZEIT	Startzeit } des KAPRØS-Jobs;
T(CPU)G	gesamte CPU-Zeit des KAPRØS-Jobs (in Sekunden);
T(CPU)M	CPU-Zeit der Moduln(in Sekunden);
T(CPU)C	CPU-Zeit von Compiler, Assembler und Linkage-Editor (in Sekunden);
T(VW)G	Verweilzeit des KAPRØS-Jobs (in Sekunden);
S	maximale Schachtelungstiefe der Moduln im KAPRØS-Job;
REG	angeforderte Region des Jobs (in K Bytes);

IL(F)	unbenutzte Region während der Go-Phase des KAPRØS-Jobs (in K Bytes);
SL(A)	Anzahl der angeforderten Sätze der Scratch-Lifeline;
SL(B)	Anzahl der benötigten Sätze der Scratch-Lifeline;
RL(A)	Anzahl der reservierten Sätze der Restart-Lifeline;
RL(B)	Anzahl der benötigten Sätze der Restart-Lifeline;
GA(B)	Anzahl der benötigten Sätze des Generellen Archivs;
F-CØDE	Fehlercode, der zum Abbruch des KAPRØS-Jobs führte (0 bei fehlerfreiem Abschluß);
F-MØDUL	Name des Moduls, in dem der Fehler auftrat (mit einem Pluszeichen + versehen, wenn es ein Bibliotheksmodul war; Blank, wenn der Fehler im KSP bemerkt wurde).

Anm.: Ein-/Ausgabefehler liefern negative Fehlercodes; Codes kleiner 100 bedeuten Nachrichtencodes; eine STØP-Anweisung liefert den Code 100; Codes größer als 100 s. 4.11; Codes CC xxx bedeuten Completions Codes.

d) Nach Jobabbruch wegen eines Fehlers in einem Modul, wegen einer STØP-Anweisung oder wegen eines Completion-Codes, der von KAPRØS abgefangen werden konnte, wird vor der Jobstatistik ein KAPRØS-Dump mit der Überschrift "KS-DUMP (0, KSSTØP 0)" ins Protokoll gedruckt. Dieser Dump kann manchmal bei der Fehlersuche helfen. Für den Benutzer wichtig sind die Ausdrucke unter BT und LT. In der BT (σ) finden sich die Blocknamen aller im Modul σ-ter Stufe, $1 \leq \sigma$, verwendeten DB. Hinter dem einfachen Blocknamen folgt eine Zahl n und dann n Paare von Zahlen $k\ell tr_i$, τ_i, i=1,...,n, die den Indices 1...n zum einfachen Blocknamen zugeordnet sind. Jedes Paar verweist auf einen Eintrag in der LT, und zwar auf den mit dem $k\ell tr$-ten Wort beginnenden Eintrag in der LT(τ). In ihm ist unter anderem verzeichnet, ob der DB schon erstellt ist (Spalte 4 ungleich 0) und ggf. wie groß er ist (Inhalt von Spalte 4 gleich Gesamtanzahl ndb der Worte).

e) Wenn das Protokoll nicht mit der Jobstatistik endet, ist das ein Zeichen dafür, daß der KAPRØS-Job wegen eines Completion Codes abgebrochen wurde, der nicht von KAPRØS abgefangen werden konnte.

f) Hinter der Jobstatistik kann noch eine Botschaft AN ALLE KAPRØS-BENUTZER folgen.

4. Systemroutinen

In der folgenden Übersicht über die Systemroutinen und ihre Parameter sind überstrichene Parameter Eingabe (Argumente) der Routinen, unterstrichene Parameter Ausgabe (Resultate) der Routinen, über- und unterstrichene Parameter beides.
Eingabeparameter können direkt als Konstante im Routinenaufruf eingesetzt werden; Ausgabeparameter oder Parameter, die Ein- und Ausgabe sind, müssen als Namen von Variablen oder Feldern eingesetzt werden.
Alle nicht ausdrücklich anders bezeichneten Größen sind Integer-Größen.
Literalkonstanten werden in Variable oder Felder (oder direkt in die Parameterlisten) linksbündig eingesetzt und rechts auf die volle Länge der Konstanten mit Blanks aufgefüllt.

4.1 KSINIT

Die Systemroutine KSINIT, die Teil jedes KAPRØS-Moduls sein muß, muß am Anfang jedes Moduls zur Initialisierung /3/ aufgerufen werden; sie liefert dabei auch einige Größen als Ausgabeparameter an:

CALL KSINIT(tc, dtcmax, ne, na, nd)

tc = Anfangs-CPU-Zeit des KAPRØS-Jobs in Sekunden (Realvariable).
dtcmax = CPU-Zeit, die für den KAPRØS-Job zur Verfügung steht; in Sekunden (Realvariable).
ne = Dateinummer der Standardeingabe (zur Zeit gleich 5).
na = " " Protokollausgabe(" " " 42).
nd = " " Standardausgabe (" " " 6).

Anm.: Die CPU-Zeit dtcr, die zu einem beliebigen Zeitpunkt im KAPRØS-Job noch zur Verfügung steht, erhält man durch die Fortran-Anweisung dtcr=dtcmax-ZEIT(tc).

4.2 KSPUT, KSGET, KSCH, KSDLT, KSPUTP, KSGETP, KSCHP

Mit KSPUT, KSGET, KSCH werden DB oder Teile von DB aus moduleigenen Feldern in die Lifeline geschrieben, aus der Lifeline in moduleigene Felder gelesen oder in der Lifeline durch Überschreiben aus moduleigenen Feldern geändert (Übertragungstechnik):

```
CALL KSPUT (name, ind, ifeld, kdb, izw, iq)
CALL KSGET (name, ind, ifeld, kdb, izw, iq)
CALL KSCH  (name, ind, ifeld, kdb, izw, iq)
```

Mit KSPUTP wird in der Internen Lifeline Platz für einen DB reserviert, in den im Anschluß an den Aufruf der DB geschrieben werden kann. Mit KSGETP wird ein DB in die Interne Lifeline gebracht, wo er im Anschluß an den Aufruf gelesen oder geändert werden kann. Mit KSCHP wird KAPRØS mitgeteilt, daß das Schreiben oder Lesen des DB nach einem KSPUTP- bzw. KSGETP-Aufruf beendet ist, und der in der Internen Lifeline festgehaltene DB wieder freigegeben werden kann (Zeigertechnik):

```
CALL KSPUTP (name, ind, izw, ifeldb, ip, iq)
CALL KSGETP (name, ind, izw, ifeldb, ip, iq)
CALL KSCHP  (name, ind, iq)
```

Mit KSDLT werden DB in der Lifeline gelöscht:

```
CALL KSDLT  (name, ind, iq)
```

name = Einfacher Blockname des DB (s.2.1) (Literalkonstante als Inhalt von 4 aufeinanderfolgenden Worten).

ind = Index zum Blocknamen des DB (s.2.1).

kdb = Relativadresse des Teiles eines DB im DB (1, wenn der ganze DB geschrieben, gelesen oder geändert werden soll).

izw = Wortzahl des Teiles eines DB oder des ganzen DB.

ifeld = (Integer-)Feld der Dimension $\geq$ izw, in dem die zu schreibenden Worte stehen oder in das die zu lesenden Worte übertragen werden sollen.

ifeldb = (Integer-)Feld von beliebiger Dimension, auf das sich der Zeiger ip beziehen soll.

ip = Zeiger zum 1. Wort des DB, bezogen auf ifeldb (d.h. in ifeld(ip) steht das erste Wort des DB).

iq = Fehlercode.

Anmerkungen:

a) In Übertragungstechnik können DB in Teilen erstellt und als sog. Teil-DB in die Lifeline geschrieben werden. DB können auch in Teilen aus der Lifeline geholt oder dort geändert werden. Die Teile brauchen nicht mit den in der Lifeline evtl. physikalisch getrennt stehenden Teil-DB zusammenfallen und werden deshalb DB-Teile genannt.

b) KSPUT darf nur für DB oder Teil-DB aufgerufen werden, die noch nicht in der Lifeline stehen. Umgekehrt darf KSGET und KSCH nur für DB oder DB-Teile aufgerufen werden, die schon in der Lifeline stehen, und zwar beliebig oft.

c) In Zeigertechnik ist die Verarbeitung von Teilen eines DB nicht möglich. Mit KSPUTP wird in der Internen Lifeline Platz für das 1. bis izw-te Wort eines DB reserviert; KAPRØS nimmt dann an, daß diese Worte im Anschluß an den Aufruf auch beschrieben werden. Mit KSGETP werden die existierenden Teil-DB so in die Interne Lifeline übertragen, wie sie im vollständigen DB angeordnet sind; KAPRØS nimmt dann an, daß noch fehlende Teil-DB im Anschluß an den Aufruf ergänzt werden (dies gilt nicht für Alte Restart-DB). Insbesondere ist auch das Verlängern eines DB in Zeigertechnik nicht möglich. In Übertragungstechnik ist das Ergänzen und Verlängern eines DB (mit KSPUT) nicht möglich, solange in irgendeinem Modul noch ein Zeiger zum DB gesetzt ist.

d) KSPUTP darf nur für DB aufgerufen werden, die noch nicht in der Lifeline stehen. Umgekehrt darf KSGETP nur für DB aufgerufen werden, von denen mindestens ein Teil-DB in der Lifeline steht, und zwar beliebig oft.

e) Mit KSPUTP und KSGETP werden die DB so in die Interne Lifeline gelegt, daß sie auf einer Doppelwortgrenze des Kernspeichers beginnen /2/.

f) Ein mit KSPUTP oder KSGETP für einen DB gesetzter Zeiger bleibt bis zu einem KSCHP- oder KSDLT-Aufruf für diesen DB gesetzt, oder längstens bis zum Ende des Moduls, in dem der Zeiger gesetzt wurde. Modulaufrufe mit KSEXEC/KSLADY beeinflussen die im rufenden Modul gesetzten Zeiger nicht; dagegen heben KSLØRD-Aufrufe die Zeiger aller DB auf, die im rufenden Modul bisher gesetzt wurden. Die Werte der Zeiger zu gleichen DB in verschiedenen Moduln unterscheiden sich im allgemeinen; deshalb muß jeder Modul alle benötigten Zeiger selbst setzen.

g) DB können mit KSDLT nur in dem Modul gelöscht werden, in dem sie lokal sind.

h) Mit einem KSGETP-Aufruf wird die Gesamtlänge ndb des DB in izw angeliefert. Mit einem KSGET-Aufruf ist es möglich, die Länge und zusätzlich die Aufteilung des DB in Teil-DB zu erfahren, wenn kdb $\leq$ 0 beim Aufruf eingesetzt wird (was zu einem Fehlercode 5 04 40 führt!):

Wenn kdb=0: Gesamtlänge des DB ndb → izw.

Wenn kdb<0: $kdb_i \rightarrow kdb$, $izw_i \rightarrow izw$, wo kdb_i und izw_i die Relativadresse und die Wortzahl des ersten Teil-DB sind, für den $kdb_i \geq |kdb|$ ist; wenn es keine solchen Teil-DB gibt: 0→kdb, ndb→izw.

i) Alte Restart-DB können nur gelesen (mit KSGET oder KSGETP), nicht geändert oder ergänzt werden. Neue Restart-DB können in beiden Techniken geschrieben, gelesen und geändert werden. Es ist jedoch zu beachten, daß ein mit KSPUTP in der Internen Lifeline erstellter Neuer Restart-DB erst beim Aufheben des Zeigers in die Restart-Lifeline übertragen wird; ein mit KSGETP in die Interne Lifeline übertragener Neuer Restart-DB wird in der Restart-Lifeline "vergessen" und beim Aufheben des Zeigers dort neu geschrieben (Platz- und Zeitverschwendung!)

j) Wenn ein KSPUTP- oder KSGETP-Aufruf auf einen Fehlercode führt, wird ip auf einen sehr großen Wert gesetzt, sodaß die irrtümliche Benutzung von ip als Zeiger auf einen Addressing-Fehler und damit zum Jobabbruch führt. Wenn ein KSPUTP-Aufruf auf den Fehlercode 9 00 05 führt,kann aus dem Wert von ip die Höchstzahl izw_{max} von Worten eines DB entnommen werden, für den zur Zeit Platz in der Internen Lifeline ist, wenn alle DB, zu denen keine Zeiger gesetzt sind, ausgelagert werden:

$$izw_{max} = ip-50\ 000\ 000$$

k) Mit der Routine KSPUTP ist im Rahmen eines KAPRØS-Jobs in gewisser Weise eine dynamische Speicheraufteilung durch Moduln möglich. Dazu reserviert sich ein Modul sämtlichen vorhandenen Platz in der Internen Lifeline durch Erstellen eines Dummy-DB in Zeigertechnik:

```
IZW = 50 000 000
CALL KSPUTP ('DUMMY..........',1,IZW,IFELDB,IP,IQ)
CALL KSCC(1,IQ)
IZW = IP-IZW
CALL KSPUTP ('DUMMY..........',1,IZW,IFELDB,IP,IQ)
```

Im Bereich IFELDB(IP)....IFELDB(IP+IZW-1) kann nun frei gearbeitet werden.

4.3 KSEXEC

Mit KSEXEC werden Moduln aufgerufen und wahlweise DB an diese weitergegeben oder von diesen übernommen.

a) CALL KSEXEC ($\overline{\text{modul}}$,iq)

b) CALL KSEXEC ($\overline{\text{modul}}$, $\overline{\text{n}}$, $\overline{\text{0}}$, $\overline{\text{names}_1}$, $\overline{\text{namea}_1}$,...,
$\overline{\text{names}_n}$, $\overline{\text{namea}_n}$, iq)

c) CALL KSEXEC ($\overline{\text{modul}}$, $\overline{\text{n}}$, $\overline{\text{m}}$, $\overline{\text{names}_1}$, $\overline{\text{namea}_1}$,...,
$\overline{\text{names}_n}$, $\overline{\text{namea}_n}$, $\overline{\text{indfld}_1}$,...,$\overline{\text{indfld}_m}$, iq)

modul = Name des gerufenen Moduls (s.2.1) (Literalkonstante als Inhalt von 2 aufeinanderfolgenden Worten).

names_i = Einfacher Standardname eines DB im gerufenen Modul (s.2.1) (Literalkonstante als Inhalt von 4 aufeinanderfolgenden Worten).

namea_i = Einfacher aktueller Name eines DB im rufenden Modul (s.2.1) oder das Schlüsselwort 'KSIØX' (Literalkonstante als Inhalt von 4 aufeinanderfolgenden Worten).

indfld_i = (Integer-)Feld der Dimension ≥ 3, in dem die Indices (s.2.1) inda_i (Anfangsindex zu names_i), inde_i (Endindex zu names_i), indv_i (Verschiebeindex zu namea_i) stehen.

n,m = Anzahl der Namenspaare bzw. Indexfelder; $m \leq n$.

iq = Fehlercode.

Anmerkungen:

α) Durch KSEXEC-Aufrufe nach b) werden also die Blocknamen

names_i, 1	namea_i, 1
(im gerufenen Modul)	(im rufenden Modul)

einander zugeordnet.

Durch KSEXEC-Aufrufe nach c) werden die Blocknamen

names_i, ind_i	namea_i, $\text{ind}_i + \text{indv}_i$
(im gerufenen Modul)	(im rufenden Modul)

für $\text{ind}_i = \text{inda}_i,\ldots,\text{inde}_i$

einander zugeordnet.

Dabei darf auch $indv_i \leq 0$ sein, solange dadurch $ind_i+indv_i>0$ bleibt.

β) Durch KSEXEC-Aufrufe nach b) und c) mit dem Schlüsselwort 'KSIØX' können auch die Blocknamen

$names_i,ind_i$	$names_i,ind_i$
(im gerufenen Modul)	(Externblock)

für $ind_i=1$ bzw. $ind_i=inda_i,\ldots,inde_i$ einander zugeordnet werden ($indv_i$ muß hier 0 sein). Bei KSEXEC-Aufrufen vom Steuermodul aus können auch mit der Form α) Externblöcke zugeordnet werden (s.5.5).

γ) In KSEXEC-Aufrufen werden nur die Blocknamen von DB einander zugeordnet; die zugehörigen DB brauchen zum Zeitpunkt des Aufrufs noch nicht zu existieren. Zugeordnete Externblocknamen müssen durch *KSIØX-Anweisungen definiert worden sein.

δ) In KSEXEC-Aufrufen der Form α) kann statt des Parameters $names_i$ auch der Parameter $namezmodulz_i$ stehen, gleich dem einfachen Standardnamen $namez_i$ eines DB in einem Zielmodul (genau 16 Zeichen) und dem Namen $modulz_i$ des Zielmoduls (bis zu 6 Zeichen), getrennt durch einen Punkt (Literalkonstante als Inhalt von 5 oder 6 aufeinanderfolgenden Worten). Diese Zuordnung hat für den gerufenen Modul keine Wirkung. Vielmehr werden dadurch die Blocknamen

$namez_i,1$	$namea_i,1$
(im Zielmodul)	(im rufenden Modul)

bzw. die Blocknamen

$namez_i,ind_i$	$namea_i,ind_i+indv_i$
(im Zielmodul)	(im rufenden Modul)

für $ind_i=inda_i,\ldots,inde_i$ einander zugeordnet. Die Zuordnung wird jedoch erst bei einem nachfolgenden Zielmodulaufruf der Form β) wirksam.

ε) Mit KSEXEC ist rekursiver Modulaufruf möglich.

4.4 KSDD

Mit KSDD werden Puffer moduleigener Dateien angelegt oder freigegeben:

CALL KSDD (nart,nfile,nform,dummy,iq)

nart = 0: Anlegen eines Puffers, der bis zur expliziten Freigabe erhalten bleiben soll;
1: Anlegen eines Puffers, der bis zur expliziten Freigabe oder bis zum Ende des Moduls, in dem er angelegt wurde, erhalten bleiben soll;
-1: Freigeben eines Puffers.

nfile = ± Dateinummer einer Fortran-Datei; positiv für eine sequentielle Datei; negativ für eine Direct-Access-Datei.

nform = Formatangabe der Fortran-Datei: positiv, wenn die Datei unformatiert beschrieben oder gelesen wird; negativ, wenn die Datei mit Format beschrieben oder gelesen wird.

Bei Nicht-Fortran-Dateien steht in nfile und nform die erste bzw. zweite Hälfte des DD-Namens der Datei als Literalkonstante.

dummy = unbenutzter Parameter.

iq = Fehlercode.

Anmerkungen:

a) Dateinummern dürfen nicht 40,...,47,50 sein.

b) Vor der erstmaligen Verwendung einer moduleigenen Datei durch READ-,WRITE-,REWIND-,BACKSPACE- und ENDFILE-Anweisungen müssen für sie durch einen KSDD-Aufruf mit nart=0 oder nart=1 Puffer angelegt werden. (Dies gilt auch für Dummy-Dateien.) Fortran-Dateien werden dabei eröffnet. Nicht-Fortran-Dateien werden nicht eröffnet; auf sie muß im Modul unmittelbar nach dem KSDD-Aufruf ein ØPEN-Makro ausgeführt werden, um Fehler im Systemkern zu vermeiden.

c) Vor dem Anlegen der Puffer einer Fortran-Direct-Access-Datei muß eine DEFINE-FILE-Anweisung ausgeführt werden (unnötig bei Direct-Access-Dateien mit statischem Puffer; s. 5.1.h).

d) Angelegte Puffer werden durch einen KSDD-Aufruf mit nart=-1 freigegeben; andernfalls automatisch am Ende des Moduls, in dem sie angelegt wurden (wenn sie mit nart=1 eröffnet wurden) oder am Jobende (wenn sie mit nart=0 eröffnet wurden). Dynamische Puffer von Fortran-Direct-Access-Dateien werden immer am Ende des Moduls, in dem sie angelegt wurden, freigegeben; statische Puffer immer am Jobende; unabhängig vom Wert von nart, mit dem sie eröffnet wurden. Fortran-Dateien werden beim Freigeben abgeschlossen; sequentielle Dateien werden "rewinded". Nicht-Fortran-Dateien werden ebenfalls abgeschlossen, falls auf sie noch kein CLØSE-Makro ausgeführt wurde.

e) Statische Puffer von Fortran-Direct-Access-Dateien können durch einen KSDD-Aufruf nicht freigegeben werden. Dynamische Puffer von Direct-Access-Dateien dürfen nach der Freigabe durch einen KSDD-Aufruf oder am Modulende nicht erneut angelegt werden. Wenn eine Direct-Access-Datei in mehr als einem Modul benutzt werden soll, muß sie daher statische Puffer bekommen. (S. 5.1.h.)

f) ENDFILE-Anweisungen und END-Ausgänge in READ-Anweisungen geben die betr. Puffer frei und schließen die Datei ab. Vor weiteren READ- und WRITE-Anweisungen auf die Datei (nach REWIND- und BACKSPACE-Anweisungen mit alter Sequence Number, sonst mit erhöhter Sequence Number) müssen die Puffer der Datei durch einen KSDD-Aufruf erneut eröffnet werden.

g) Bei REWIND-Anweisungen sorgt KAPRØS dafür, daß die Puffer der Datei nicht freigegeben werden; erneutes Anlegen durch einen KSDD-Aufruf ist unnötig.

4.5 KSDAC

Mit KSDAC können die Charakteristiken moduleigener Direct-Access-Dateien mit statischen Puffern (s. 5.1.h) abgefragt werden:

CALL KSDAC (nfil,nrec,lrec,iav,iq)

nfil = Dateinummer.
nrec = Anzahl der Sätze der Datei.
lrec = Satzlänge der Datei in Bytes.
iav = Wert der assoziierten Variablen der Datei.
iq = Fehlercode.

4.6 KSARC

Mit KSARC werden DB (die keine Externblöcke sein müssen, aber auch Externblöcke beliebigen Typs sein können) aus der Lifeline in ein Archiv übertragen. Im Gegensatz zur Archivierung von Archivausgabe-DB durch das KSP, die erst am Ende eines (fehlerfreien) KAPRØS-Jobs erfolgt, werden DB mit KSARC sofort archiviert.

CALL KSARC (name,ind,n,kenn,iq)

name = Einfacher Blockname des DB (s. 2.1) (Literalkonstante als Inhalt von 4 aufeinanderfolgenden Worten).

ind = Index zum Blocknamen des DB (s. 2.1).

n = 0, für das Generelle Archiv; oder Dateinummer (ungleich 40,..,50) eines Benutzerarchivs.

kenn = Kennzeichen des DB (s. 2.1), wenn er in ein Benutzerarchiv übertragen werden soll; beliebig, wenn er in das Generelle Archiv übertragen werden soll (Literalkonstante).

iq = Fehlercode.

4.7 KSMØVE

Mit KSMØVE können DB aus der Externen in die Interne Lifeline übertragen werden oder umgekehrt aus der Internen in die Externe Lifeline. Da KAPRØS automatisch die Verteilung der DB vornimmt, sollte KSMØVE nur in Ausnahmefällen verwendet werden.

CALL KSMØVE (k,name,ind,iq)

k = 1: Übertragen des DB aus der EL in die IL;
-1: Übertragen des DB aus der IL in die EL;
0: Abfragen der Höchstzahl von Worten, für die zur Zeit Platz in der IL ist, ohne daß DB aus der IL ausgelagert werden müßten; sie wird in k zurückgegeben.

name = Einfacher Blockname des DB (s. 2.1) (Literalkonstante als Inhalt von 4 aufeinanderfolgenden Worten).

ind = Index zum Blocknamen des DB (s. 2.1).

iq = Fehlercode.

Anmerkungen:

a) Beim Übertragen eines DB aus der EL in die IL bleibt der DB in der EL erhalten. Beim Übertragen eines DB aus der IL in die EL wird der DB in der IL gelöscht.

b) KSMØVE-Aufrufe für DB, zu denen Zeiger gesetzt sind, sind wirkungslos.

c) DB werden nur dann aus der EL in die IL übertragen, wenn dort Platz ist, ohne daß DB aus der IL ausgelagert werden müßten (d. h. in den Fällen, in denen DB auch mit KSGET in die IL gebracht würden).

4.8 KSLADY und KSLØRD

KSLADY ist eine Erweiterung der Systemroutine KSEXEC (s. 4.3). Mit KSLADY werden Moduln aufgerufen und DB an diese weitergegeben oder von diesen übernommen; wahlweise kann dabei das Auslagern des rufenden Moduls aus dem Kernspeicher unterdrückt werden. Mit KSLØRD können Moduln vorab in den Kernspeicher geladen werden, die dort bis zum expliziten Löschen stehen bleiben sollen. KSLADY und KSLØRD sollten nur in Ausnahmefällen verwendet werden, wenn häufiges Laden oder Aus- und Rücklagern von Moduln zu hohen CPU- und Peripheriekosten führen würde.

CALL KSLADY ($\overline{k}$,$\overline{modul}$,...)

k = 0: Aufruf des Moduls modul unter Auslagern aller im Kernspeicher stehenden aktivierten Moduln;

-1: Aufruf des Moduls modul unter Auslagern des letzten im Kernspeicher stehenden aktivierten Moduls (entspricht einem Aufruf von KSEXEC);

-2: Aufruf des Moduls modul unter Auslagern der beiden letzten im Kernspeicher stehenden aktivierten Moduln;

. . . . . .

1: Aufruf des Moduls modul ohne Auslagern von Moduln.

modul
:
} Parameter und ihre Bedeutung wie bei KSEXEC.

CALL KSLØRD ($\pm\overline{n}$,$\overline{modul_1}$,...,$\overline{modul_n}$,$\underline{iq}$)

n = Anzahl der Modulnamen in der Parameterliste;
+ n bedeutet Laden der Moduln;
- n bedeutet Löschen der Moduln.

$modul_i$ = Name eines Moduls (s. 2.1) (Literalkonstante als Inhalt von 2 aufeinanderfolgenden Worten).

iq = Fehlercode.

Anmerkungen:

a) Die mit einem KSLØRD-Aufruf zu ladenden Moduln werden in der Reihenfolge, in der die Modulnamen in der Parameterliste stehen, in den Kernspeicher hinter die schon dort stehenden Moduln (wozu auch der rufende Modul gehört) geladen. Das Löschen von Moduln muß in umgekehrter Reihenfolge geschehen, d. h. der KSLØRD-Aufruf für die zuletzt geladenen Moduln muß zuerst kommen, usw.; dabei spielt jedoch die Reihenfolge der Modulnamen in der Parameterliste keine Rolle. Versuche, Moduln mehrfach zu laden oder zu löschen, werden ignoriert.

b) Mit KSLØRD geladene Moduln können nicht gelöscht werden, solange sie noch aktiviert sind. Alle mit KSLØRD geladenen Moduln werden am Ende des Moduls, in dem sie geladen wurden, automatisch gelöscht, falls sie nicht schon vorher explizit durch KSLØRD-Aufrufe gelöscht wurden.

c) Mit KSEXEC oder mit KSLADY und $k \leq 0$ gerufene Moduln dürfen noch nicht im Kernspeicher stehen; sie werden nach dem Durchlaufen gelöscht. Mit KSLADY und $k=1$ gerufene Moduln können schon im Kernspeicher stehen; in diesem Fall werden sie nach dem Durchlaufen nicht gelöscht.

d) Durch KSEXEC-Aufruf oder durch KSLADY-Aufruf mit $k \leq 0$ auszulagernde aktivierte Moduln dürfen nicht mit KSLØRD geladen worden sein oder andere Moduln mit KSLØRD geladen haben.

e) Aktivierte Moduln, die nicht ausgelagert sind oder nicht durch KSEXEC-Aufruf oder KSLADY-Aufruf mit $k \leq 0$ ausgelagert werden, dürfen nicht rekursiv aufgerufen werden.

f) KSLØRD-Aufrufe implizieren KSCHP-Aufrufe für alle DB, zu denen zum Zeitpunkt des Aufrufs im rufenden Modul Zeiger gesetzt sind /3/.

4.9 KSCC

Mit KSCC wird der Nachrichtencode gesetzt, abgefragt oder gelöscht, sowie der interne Fehlercode gelöscht.

a) Setzen des Nachrichtencodes auf den Wert iq:

CALL KSCC ($\overline{-1}$,$\overline{iq}$)

b) Abfragen des Nachrichtencodes (Übertragen des Wertes nach iq):

CALL KSCC ($\overline{0}$,iq)

c) Löschen des Nachrichtencodes oder internen Fehlercodes iq:

CALL KSCC ($\overline{1}$,$\overline{iq}$)

Anmerkungen:

Der Nachrichtencode kann auf einen Wert zwischen 1 und 99 gesetzt werden. Codes zwischen 90 und 99 haben zur Folge, daß der Job abgebrochen wird. Beim Löschen muß iq mit dem Wert des Nachrichtencodes (iq<100) oder des internen Fehlercodes (iq≥100) übereinstimmen.

4.10 KSDUMP

Mit KSDUMP können Moduln zur Hilfe bei der Fehlersuche KAPRØS-Dumps (s. 3) erzeugen.

CALL KSDUMP ($\overline{0}$,$\overline{m1}$,$\overline{m2}$,$\overline{m3}$)

m1,m2,m3 = Konstanten, die in der Überschrift des KAPRØS-Dumps zur Identifizierung desselben im Format A4,A4,I3 ausgedruckt werden.

4.11 Fehlercodes

Die Systemroutinen setzen den Fehlercode iq auf folgende Werte:

0: kein Fehler
xxyyzz: Fehler der Klasse A oder B
-xxyyzz: Fehler der Klasse C

Im Falle eines Fehlers der Klasse A oder B wird auch der interne Fehlercode auf den Wert xxyyzz gesetzt. Fehler der Klasse A oder B veranlassen den Ausdruck einer Fehlermeldung. Bei Fehlern der Klasse A wird der KAPRØS-Job dann abgebrochen. Fehler der Klasse C veranlassen den Ausdruck einer Warnung.

a) xx bezeichnet die Systemroutine, in der der Fehler auftrat:
   01 KSINIT
   02 KSEXEC/KSLADY
   03 KSDD
   04 KSCC
   05 KSGET
   06 KSPUT
   07 KSCH
   08 KSGETP
   09 KSPUTP
   10 KSCHP
   11 KSDAC
   12 KSDLT
   13 KSARC
   14 KSLØRD
   15 KSMØVE

b) yy bezeichnet die Stellung des Parameters in der Aufrufliste der Systemroutine, der den Fehler verursacht hat (00, wenn der Fehler keinem Parameter zugeordnet werden kann).

c) zz bezeichnet die Art des Fehlers (in der folgenden Aufstellung sind jeweils auch die Klasse des Fehlers und die Systemroutinen, die ihn setzen können, angegeben sowie bei Fehlern der Klasse C die Aktionen der Systemroutinen):

03 Der Kernspeicher reicht für rückzulagernde Zeiger-DB nicht aus, da inzwischen die Tabellen zu lang sind.
(A; KSEXEC/KSLADY)

04 Der Kernspeicher reicht für einen rückzulagernden Modul nicht aus, da inzwischen die Tabellen zu lang sind.
(A; KSEXEC/KSLADY)

05 Im Kernspeicher ist nicht genügend Platz, um einen DB in Zeigertechnik zu bearbeiten.
(B; KSPUTP, KSGETP)

06 SL-Überlauf.
(A; KSEXEC/KSLADY, KSLØRD, KSPUT, KSPUTP, KSGETP, KSMØVE, KSARC, KSDD)

07 RL-Überlauf.
(A; KSEXEC/KSLADY, KSLØRD, KSPUT, KSCHP)

08 Kernspeicherüberlauf.
(A; KSEXEC/KSLADY, KSLØRD, KSPUT, KSPUTP, KSARC, KSDD)

09 Ein gerufener Modul findet im Kernspeicher keinen Platz.
(A; KSEXEC/KSLADY, KSLØRD)

11 Ein Blockname ist nicht zu finden.
(B; KSGET, KSCH, KSDLT, KSGETP, KSCHP, KSMØVE, KSARC)

12 Inkonsistenter Zielmodulname bei 'KSIØX' als aktuellem Namen.
(C; KSEXEC/KSLADY) - Der Zielmodulname wird ignoriert.

15 Ein Index zum Blocknamen ist kleiner/gleich Null.
(B; KSEXEC/KSLADY, KSPUT, KSGET, KSCH, KSDLT, KSPUTP, KSGETP, KSCHP, KSMØVE, KSARC)

16 Ein Index zum Blocknamen ist nicht zu finden.
(B; KSGET, KSCH, KSDLT, KSGETP, KSCHP, KSMØVE, KSARC)

17 Ein Anfangsindex ist größer als ein Endindex.
(B; KSEXEC/KSLADY)

18 Von Null verschiedener Verschiebeindex bei 'KSIØX' als aktuellem Namen.
(C; KSEXEC/KSLADY) - Der Verschiebeindex wird ignoriert.

19 Versuch einen schon geladenen Modul in Auslagerungstechnik aufzurufen oder einen nicht ausgelagerten aktivierten Modul rekursiv aufzurufen.
(A; KSEXEC/KSLADY)

20 Mehrfache Zuordnung von aktuellen Namen zu einem Standardnamen.
(B; KSEXEC/KSLADY)

21 Ein Standardname in einem Modul 1.Stufe oder bei 'KSIØX' als aktuellem Namen ist als Externblockname nicht definiert.
(C; KSEXEC/KSLADY) - Ein Scratch-DB mit dem Standardnamen wird als Externblock zugeordnet.

22 Versuch, einen Modul auszulagern, der durch einen KSLØRD-Aufruf geladen worden ist.
(A; KSEXEC/KSLADY)

23 Die Stufe eines gerufenen Moduls ist größer als die vorgesehene maximale Schachtelungstiefe s_{max}=22 der Moduln.
(B; KSEXEC/KSLADY)

24 Versuch, mehr als 10 Moduln zu laden.
(B; KSLØRD)

25 Versuch, mehr Moduln als vorhanden auszulagern.
(A; KSLADY)

26 Versuch, einen Modul auszulagern, der mit KSLØRD-Aufrufen andere Moduln geladen hat.
(A; KSEXEC/KSLADY)

27 Ein zu löschender Modul ist nicht der zuletzt durch einen KSLØRD-Aufruf geladene.
(B; KSLØRD)

28 Versuch, einen schon geladenen und aktivierten Modul zu laden, oder einen noch aktivierten Modul zu löschen.
(B; KSLØRD)

29 Ein gerufener Modul ist in den Bibliotheken nicht zu finden.
(B; KSEXEC/KSLADY, KSLØRD)

30 Überflüssiges Aufheben eines Zeigers (er ist in keinem Modul gesetzt).
(C; KSCHP) - Der Versuch des Aufhebens wird ignoriert.

31 Überflüssiges Aufheben eines Zeigers (er ist im rufenden Modul nicht gesetzt).
(C; KSCHP) - Der Versuch des Aufhebens wird ignoriert.

32 Überflüssiges Setzen eines Zeigers.
(C; KSGETP) - Der Wert des Zeigers wird zurückgegeben.

33 Versuch, einen DB, zu dem ein Zeiger gesetzt ist, zu ergänzen.
(B; KSPUT)

34 Versuch, einen DB, zu dem ein Zeiger gesetzt ist, in die EL zu übertragen.
(C; KSMØVE) - Der DB bleibt in der IL stehen.

35 Versuch, einen Alten Restart-DB zu ergänzen oder zu ändern.
(B; KSPUT, KSCH)

36 Versuch, einen nichtlokalen DB zu löschen.
(C; KSDLT) - Der Löschversuch wird ignoriert.

37 Unkorrekter Versuch, den Nachrichtencode oder internen Fehlercode zu löschen.
(C; KSCC) - Der Nachrichtencode bzw. der interne Fehlercode wird gelöscht.

38 Versuch, den Nachrichtencode auf einen unzulässigen Wert zu setzen.
(C; KSCC) - Der Setzversuch wird ignoriert.

39 Versuch, einen schon gelöschten Nachrichtencode oder internen Fehlercode zu löschen.
(C; KSCC) - der Löschversuch wird ignoriert.

40 Eine Relativadresse ist kleiner/gleich Null.
(B; KSPUT, KSGET, KSCH) - Wenn von KSGET gesetzt: (4.2.h) beachten!

41 Eine Wortzahl ist kleiner/gleich Null.
(B; KSPUT, KSGET, KSCH, KSPUTP)

42 Versuch, einen leeren DB zu lesen oder zu ändern.
(B; KSGET, KSCH, KSGETP, KSMØVE, KSARC)

43 Versuch, Teile eines DB, die vor oder zwischen den vorhandenen Teil-DB liegen, zu lesen oder zu ändern (evtl. Relativadresse zu klein). - Die Worte der vorhandenen Teil-DB werden gelesen bzw. geändert.
(B; KSGET, KSCH)

44 Versuch, Teile eines DB, die hinter den vorhandenen Teil-DB liegen, zu lesen oder zu ändern (evtl. Wortzahl zu groß). - Die Worte der vorhandenen Teil-DB werden gelesen bzw. geändert.
(B; KSGET, KSCH)

45 Versuch, einen schon vorhandenen Teil-DB zu überschreiben.
(B; KSPUT, KSPUTP)

49 Leere Teile eines DB wurden (mit Nullen gefüllt) in ein Archiv übertragen.
(C; KSARC)

50 Zu einer Direct-Access-Datei gibt es keine DD-Karte.
(B; KSDAC)

51 Versuch, Puffer für die Standardeingabe- oder -ausgabedatei anzulegen oder freizugeben.
(C; KSDD) - Der Versuch wird ignoriert.

52 Eine Direct-Access-Datei soll nach dem Schließen wieder eröffnet werden.
(B; KSDD)

53 Unzulässige Dateinummer.
(B; KSDD, KSARC)

54 Überlauf der Datei-Tabellen.
(A; KSDD)

55 Anlaufen des END-Parameters in einer READ-Anweisung oder Ausführung einer ENDFILE-Anweisung, ohne daß Puffer eröffnet sind.
(B; KSDD)

56 Versuch, einen statischen Puffer einer Direct-Access-Datei zu löschen.
(C; KSDD) - Der Löschversuch wird ignoriert.

57 Versuch, einen schon gelöschten Puffer zu löschen.
(C; KSDD) - Der Löschversuch wird ignoriert.

58 Für eine Direct-Access-Datei wurde auf der DD-Karte DISP=MØD angegeben.
(C; KSDD) - Der Puffer wird eröffnet.

60 Tabellenüberlauf in KSINIT.
(A; KSINIT)

61 Versuch, einen schon in der IL stehenden DB in die IL zu übertragen.
(C; KSMØVE) - Der Versuch wird ignoriert.

62 Versuch, einen nicht in der IL stehenden DB in die EL zu übertragen.
(C; KSMØVE) - Der Versuch wird ignoriert.

63 In der IL ist kein Platz für einen aus der EL zu übertragenden DB.
(C; KSMØVE) - Der Übertragungsversuch wird ignoriert.

64 Dateiende beim Schreiben auf ein Archiv.
(B; KSARC)

zz≥80: Mögliche Ursachen für solche Fehler sind: Nach KSPUTP- oder KSGETP-Aufrufen Schreiben bzw. Ändern von mehr als izw Worten in Zeigertechnik (Systemtabellen wurden zerstört); Verwenden von moduleigenen Dateien ohne vorheriges Anlegen von Puffern durch KSDD-Aufruf; Modifizieren von Bibliotheksmodulnohne entsprechende Korrektur der Modullänge im Modulverzeichnis; Lesefehler auf Systemdateien und Archiven.
(A; KSEXEC/KSLADY,KSLØRD,KSPUT,KSGET,KSCH,KSDLT,KSPUTP,KSGETP, KSCHP,KSMØVE,KSARC,KSDD)

Anm.: Die Fehlercodes -2yy12, -2yy18, -2yy21 werden nicht an die Variable iq in der Parameterliste von KSEXEC/KSLADY weitergegeben.

4.12 Fehlermeldungen und Warnungen

a) KSPUT, KSGET, KSCH, KSDLT:

KS-FEHLER
KS-WARNUNG } iq; name ind [kdb izw]

b) KSPUTP, KSGETP, KSCHP, KSMØVE:

KS-FEHLER
KS-WARNUNG } iq; name ind [izw]

c) KSEXEC/KSLADY:

KS-FEHLER iq [; modul [names namea modulz inda inde indv ind]]

KS-WARNUNG iq; names modulz indv (für iq = -2yy12 und -2yy18)

KS-WARNUNG iq; names modul ind (für iq = -2yy21)

d) KSDD,KSDAC: Selbsterklärende Fehlermeldungen und Warnungen

e) KSARC:

KS-FEHLER
KS-WARNUNG } iq; name ind n

f) KSLØRD:

KS-FEHLER iq [; modul]

g) KSCC:

KS-WARNUNG -4yyzz; iq

4.13 Completion Codes

Completion Codes 0C1,0C4,0C5 können in KAPRØS-Jobs aus den in Abschnitt 4.11 unter zz≥80 angegebenen Ursachen entstehen; ferner wenn Parameter in Systemroutinenaufrufen fehlen oder falsch eingesetzt sind, wenn KSINIT nicht aufgerufen wird (s. 5.1.d), usw.

5. Anforderungen an KAPRØS-Moduln

5.1 Allgemeines

a) Ein KAPRØS-Modul besteht aus einer oder mehreren mit dem Assembler H, dem Fortran G1 Compiler und/oder dem Fortran H Extended Compiler übersetzten Routinen. Die Systemroutine KSINIT muß Bestandteil jedes Moduls sein (bei Testmoduln wird sie vom KSP, bei Bibliotheksmoduln vom aufnehmenden Dienstprogramm zum Modul gelinkt).

b) Alle Routinen eines Moduls müssen als Subroutinen geschrieben sein. MAIN-Programme sind nicht erlaubt, weil darin enthaltene RETURN-Anweisungen wie STØP-Anweisungen wirken. Der Name der Hauptroutine des Moduls muß auf der ENTRY-Karte in der *LINK-Eingabe (s.2.2 und 6.3) angegeben werden.

c) KAPRØS-Moduln können (außer über die Lifeline oder moduleigene Dateien) über eine Art Common, der aus 5 aufeinanderfolgenden Worten, den sog. Aufrufparametern, besteht, miteinander korrespondieren. Dazu werden bis zu 5 Variablennamen, jeweils in Schrägstriche eingeschlossen, in der Aufrufliste der Hauptroutinen der Moduln angegeben. Vom KSP werden die entsprechenden 5 Aufrufparameter vor dem Aufruf der Prüf-, Druck- und Lesemoduln sowie des Steuermoduls mit Null initialisiert (bei Lesemoduln wird der 1. Aufrufparameter auf die Nummer der Eingabedatei gesetzt, s. 5.4).

d) Als eine der ersten ausführbaren Anweisungen in jedem Modul (auf jeden Fall vor jeder I/Ø-Anweisung und jedem Systemroutinen- oder Bibliotheksroutinenaufruf) muß die Systemroutine KSINIT aufgerufen werden (s.4.1).

e) KAPRØS-Moduln sollen keine STØP- oder STØP i-Anweisungen enthalten, sondern über eine RETURN-Anweisung oder eine Kette von RETURN-Anweisungen ins KSP zurückspringen. Beim Auftreten einer STØP- oder STØP i-Anweisung oder beim Setzen des Nachrichtencodes auf 90,...,99 wird, ebenso wie bei Fehlern der Klasse A und bei nicht gelöschten Fehlern der Klasse B (s.f) der KAPRØS-Job abgebrochen, ohne daß Druckausgabe-DB gedruckt oder Archivausgabe-DB ins Archiv gebracht werden; auch können Neue Restart-DB, zu denen Zeiger gesetzt sind, verlorengehen.

f) Moduln müssen nach dem Aufruf der Systemroutinen (außer KSINIT, KSCC, KSDUMP) den Fehlercode iq abfragen. Bei positiven Fehlercodes (das sind solche der Klasse B) muß, nachdem auf den Fehler sinnvoll reagiert wurde, der interne Fehlercode durch Aufruf von KSCC gelöscht werden; andernfalls wird der Job beim nächsten Aufruf einer Systemroutine oder am Modulende abgebrochen. Negative Fehlercodes (das sind solche der Klasse C) können ignoriert werden.

g) KAPRØS-Moduln können die Region nicht selbst verwalten, da diese von KAPRØS verwaltet wird. Der einzige Weg zu einer dynamischen Speicheraufteilung durch Moduln führt über die Systemroutine KSPUTP (s. 4.2.k).

h) Wegen g) müssen vor der Verwendung einer moduleigenen Datei deren Puffer angelegt und nach der Verwendung wieder freigegeben werden. Das geschieht durch Aufruf der Systemroutine KSDD (s. 4.4). Für sequentielle Fortran-Dateien wird der Pufferplatz d y n a m i s c h , d. h. erst beim KSDD-Aufruf bereitgestellt. Für Fortran-Direct-Access-Dateien kann der Pufferplatz dynamisch bereitgestellt werden (s. jedoch 4.4.e); er kann aber auch s t a t i s c h , d. h. am Jobanfang für die ganze Dauer des KAPRØS-Jobs bereitgestellt werden. Das wird vom KSP veranlaßt, wenn die Dateinummer 48 oder 49 ist, wenn der Dateiname mit KSDA beginnt oder wenn der Dateiname gleich GRUBA,JBGRUC, REMØ,GRØUCØ,KNDF,KEDAK3 oder MØXTØT ist. DEFINE FILE-Anweisungen für solche Dateien sind unnötig; die Charakteristiken können durch Aufruf der Systemroutine KSDAC (s. 4.5) erfragt werden. Zur Eröffnung der Datei ist auch hier ein KSDD-Aufruf notwendig; eine Freigabe der Puffer vor Jobende ist nicht möglich. Für Nicht-Fortran-Dateien wird der Pufferplatz immer statisch bereitgestellt; das wird vom KSP veranlaßt, wenn der DD-Name nicht Blank oder STEPLIB ist oder nicht mit FT,PGM,KS oder SYS beginnt.

i) Die Dateinummern 40,...,47,50 dürfen für moduleigene Dateien nicht verwendet werden; für 48,49 gelten die Besonderheiten nach h).

j) Der Aufruf der Bibliotheksroutine FSPIE ist wirkungslos.

5.2 Prüfmoduln

Jeder syntaxfehlerfreie und nichtleere Karteneingabe-DB wird, wenn auch die zugehörige *KSIØX-Anweisung fehlerfrei war, vor dem Anlaufen des Steuermoduls von dem Prüfmodul geprüft, dessen Namen im PM- oder PMN-Parameter der *KSIØX-Anweisung angegeben ist. Das gleiche gilt für nichtleere Archiveingabe-DB und nichtleere Alte Restart-DB, wenn die zugehörigen *KSIØX-Anweisungen den PM- oder PMN-Parameter enthalten.
Mit Hilfe der Verkettung (s. 2.3.β) können mehrere DB durch e i n e n Prüfmodul geprüft werden. Außerdem kann die von Karteneingabe-, Archiveingabe- und/oder Alten Restart-DB stammende Eingabe im Prüfmodul auch umgearbeitet und in Form von Scratch- und/oder Neuen Restart-DB in die Lifeline geschrieben werden. Voraussetzung dazu ist, daß die *KSIØX-Anweisungen der Scratch- und Neuen Restart-DB den PM- oder PMN-Parameter enthalten. Die verkettete Prüfung entfällt, wenn wenigstens ein DB syntaxfehlerbehaftet oder leer ist.
Die Prüfmoduln werden vom KSP nach der Verarbeitung aller *KSIØX-Karten für die einzeln oder verkettet zu prüfenden DB aufgerufen. Die Standardnamen in einem Prüfmodul, die das KSP den Blocknamen der zu prüfenden oder zu erstellenden Externblöcke zuordnet, hängen von der Art des in der letzten *KSIØX-Anweisung einer Verkettung verwendeten Schlüsselwortes ab:

a) Bei 'PM=Prüfmodulname' sind als Standardnamen im Prüfmodul die Blocknamen der Externblöcke zu verwenden.

b) Bei 'PMN=Prüfmodulname' sind als Standardnamen im Prüfmodul die Blocknamen 'KSTEST_____', Indices 1,...,n, zu verwenden, die den Blocknamen der Externblöcke (hier n) in der Reihenfolge der zugehörigen *KSIØX-Anweisungen entsprechen.

Unter den Standardnamen muß der Prüfmodul die zu prüfenden DB aus der Lifeline lesen (durch Aufruf der Systemroutine KSGET oder KSGETP). Der Prüfmodul prüft die Blockdaten des DB, druckt bei Fehlern eine Diagnose in das Protokoll und teilt dem KSP durch Setzen des Nachrichtencodes auf einen Wert zwischen 1 und 89 (durch Aufruf der Systemroutine KSCC) mit, daß ein Fehler festgestellt wurde. Das KSP bricht dann nach der Eingabe-

prüfung aller DB den KAPRØS-Job ab. Evtl. zu erstellende DB werden unter ihren Standardnamen in die Lifeline geschrieben (durch Aufruf der Systemroutinen KSPUT oder KSPUTP).

5.3 Druckmoduln

Jeder nichtleere Druckausgabe-DB wird nach dem Durchlaufen des Steuermoduls von dem Druckmodul ausgedruckt, dessen Namen im PM- oder PMN-Parameter der zugehörigen *KSIØX-Anweisung angegeben ist. Das gleiche gilt für nichtleere Archivausgabe-DB, wenn die zugehörigen *KSIØX-Anweisungen den PM- oder PMN-Parameter enthalten.
Mit Hilfe der Verkettung (s. 2.3.β) können mehrere DB durch e i n e n Druckmodul gedruckt werden. Das verkettete Drucken entfällt, wenn wenigstens ein DB leer ist.
Die Druckmoduln werden vom KSP für die einzeln oder verkettet zu druckenden DB aufgerufen. Die Standardnamen in einem Druckmodul, die das KSP den Blocknamen der zu druckenden Externblöcke zuordnet, hängen von der Art des in der letzten *KSIØX-Anweisung einer Verkettung verwendeten Schlüsselwortes ab:

a) Bei 'PM=Druckmodulname' sind als Standardnamen im Druckmodul die Blocknamen der Externblöcke zu verwenden.

b) Bei 'PMN=Druckmodulname' sind als Standardnamen im Druckmodul die Blocknamen 'KSTEST_____', Indices 1,...,n, zu verwenden, die den Blocknamen der Externblöcke (hier n) in der Reihenfolge der zugehörigen *KSIØX-Anweisungen entsprechen.

Unter den Standardnamen muß der Druckmodul die zu druckenden DB aus der Lifeline lesen (durch Aufruf der Systemroutinen KSGET oder KSGETP).
Der Druckmodul druckt die Blockdaten des DB ins Protokoll oder auf die Standardausgabe.

5.4 Lesemoduln

Blockdaten eines Karteneingabe-DB, die nicht formatfrei vom KSP eingelesen und in die Lifeline gebracht werden sollen, werden, wenn auch die zugehörige *KSIØX-Anweisung fehlerfrei war, von dem Lesemodul, dessen

Namen im LM- oder LMN-Parameter der *KSIØX-Anweisung angegeben ist, eingelesen und in die Lifeline gebracht.
Die Lesemoduln werden vom KSP unmittelbar nach der Verarbeitung der zugehörigen *KSIØX-Anweisungen aufgerufen. Der Standardname in einem Lesemodul, den das KSP dem Blocknamen des einzulesenden Externblocks zuordnet, hängt von der Art des in der *KSIØX-Anweisung verwendeten Schlüsselwortes ab:

a) Bei 'LM=Lesemodulname' ist als Standardname im Lesemodul der Blockname des Externblocks zu verwenden.

b) Bei 'LMN=Lesemodulname' ist als Standardname im Lesemodul der Blockname 'KSTEST_____',Index 1, zu verwenden.

Das KSP setzt die Nummer der Eingabedatei, auf der die Blockdaten stehen (im Normalfall die Nummer der Standardeingabe), in den ersten Aufrufparameter des Moduls ein. Der Lesemodul muß die Blockdaten des DB mit dem entsprechenden Format von der Eingabedatei mit Fortran-READ-Anweisungen lesen, die Blockdaten ins Protokoll drucken und unter dem obigen Standardnamen (durch Aufruf der Systemroutine KSPUT oder KSPUTP) in die Lifeline schreiben. Wenn die Blockdaten des DB von einer Endekarte abgeschlossen werden, muß der Lesemodul diese mitlesen. Wenn der Lesemodul einen Lesefehler oder Dateiende (EØF) auf der Eingabedatei entdeckt, muß er eine dementsprechende Mitteilung ins Protokoll drucken und den Nachrichtencode auf 88 bzw. 89 setzen (durch Aufruf der Systemroutine KSCC); ähnlich, wenn der Lesemodul entdeckt, daß der DB leer ist (Nachrichtencode: irgendein Wert zwischen 1 und 87). Die Syntaxprüfung soll erst durch den Prüfmodul erfolgen.

5.5 Steuermoduln

Wenn die gesamte KAPRØS-Eingabe fehlerfrei verarbeitet wurde, ruft das KSP den Steuermodul auf, der auf der *GØ-Anweisung spezifiziert wurde. Das KSP setzt beim Aufruf des Steuermoduls die Blocknamen aller Externblöcke, die keine Modulqualifikation haben oder die für den Steuermodul explizit qualifiziert sind (s. 2.3.ð), als Standardnamen ein.

Anm.: Wenn es zwei oder mehr gleichnamige Externblöcke ohne Modulqualifikation oder mit Qualifikation für den Steuermodul gibt, führt der Aufruf des Steuermoduls zu einem Fehler.

6. Benutzungshinweise

6.1 JCL-Karten für KAPRØS-Jobs

Zum Rechnen von KAPRØS-Jobs gibt es die katalogisierten Prozeduren KSCLG (vorzugsweise für KAPRØS-Jobs mit einer Compile-/Link- und einer Go-Phase) und KSG (für KAPRØS-Jobs mit einer Go-Phase allein). Damit werden die folgenden JCL-Karten für einen oder mehrere KAPRØS-Jobs benötigt:

```
//Jobkarte (wegen der Regiongröße s. 6.2)
/*FØRMAT PR,DDNAME=FT42FOO1
[/*FØRMAT PR,DDNAME=SYSPRINL]                                    (a)
[/*FØRMAT PR,DDNAME=KSAPRINT]                                    (b)
//  EXEC KSCLG|KSG                          }
[//K.FT44FOO1 DD SPACE=(3064,msl)]                               (c)
[//K.FTxxFyyy DD .....]                                          (d)
.....                                       }
[//K.ddname DD  .....]                      }   Erster KAPRØS-   (e)
.....                                       }   Job
[//K.SYSUDUMP DD SYSØUT=A]                                       (f)
//K.SYSIN DD *                              }
KAPRØS-Eingabe                              }
[Moduleigene Eingabe]                       }
/*                                          }
//  EXEC KSCLG|KSG                          }   Zweiter KAPRØS-
.....                                       }   Job usw.
/*                                          }
//
```

Anmerkungen:

a) Die Karte (a) wird nur in der Prozedur KSCLG benötigt.

b) Die Karte (b) wird nur in der Prozedur KSCLG benötigt, wenn Assembler-Routinen in Testmoduln enthalten sind.

c) Die Karte (c) wird nur benötigt, wenn die Satzzahl msl (zu je 766 Worten) der Scratch-Lifeline verschieden vom Standardwert 150 gewünscht wird.

d) Die Karten (d) (mit xx ungleich 40,...,47,50) werden für Eingabedateien, die Assembler-, Compiler- oder Linkage-Editor-Eingabe (s. 2.2) oder Blockdaten (s. 2.3) enthalten, für Benutzerarchive (s. 6.4) oder für moduleigene Fortran-Dateien (s. 5.1.h) benötigt.

e) Die Karten (e) (mit ddname ungleich FTxxFyyy) werden für moduleigene Nicht-Fortran-Dateien (s. 5.1.h) benötigt.

f) Die Karte (f) wird nur benötigt, wenn im Falle eines Completion Codes ein vollständiger Dump ausgedruckt werden soll; ein Kurzdump (s. SNAP-Macro Instruction mit Parametern SDATA=(CB,Q), PDATA=(PSW,REGS,SA) in: IBM System/360 Operating System, Supervisor and Data Management Macro Instruction) wird bei einem Completion Code automatisch ausgegeben.

6.2 Regiongröße

Der KAPRØS-Systemkern (einschließlich Bibliotheksroutinen) belegt im Kernspeicher 80 K Bytes. Von ØS-Puffern und ØS-Routinen wird weiterer Kernspeicherplatz belegt. Die benötigte Region errechnet sich dann wie folgt:

Für die Compile-/Link-Phase:

Systemkern, ØS-Puffer und -Routinen I	106 K Bytes
Platzbedarf des Assemblers, Compilers oder Linkage Editors, und zwar der Bedarf des größten der verwendeten Programme	☐ K Bytes
	Summe

Für die Go-Phase:

Systemkern, ØS-Puffer und -Routinen II	132 K Bytes
Platzbedarf des größten Moduls (oder größte Länge aller gleichzeitig im Kernspeicher stehenden Moduln, s. 4.8), aufgerundet auf 2 K Bytes	☐ K Bytes
Platzbedarf der Internen Lifeline (mindestens 4 K Bytes)	☐ K Bytes
Platzbedarf für Puffer von Eingabedateien mit Blockdaten, von Benutzerarchiven und von moduleigenen Dateien	☐ K Bytes
	Summe

Anmerkungen:

a) Der Platzbedarf statischer Puffer (s. 5.1.h) von Fortran-Direct-Access-Dateien und von Nicht-Fortran-Dateien muß auch in der Compile-/Link-Phase mit eingerechnet werden.

b) Der in der Go-Phase nicht ausgenützte Kernspeicherplatz der Region wird, auf K Bytes abgerundet, in der Jobstatistik unter der Überschrift IL(F) ausgedruckt (s. 3).

c) Der Platzbedarf in Bytes der Puffer einer Datei ist gegeben durch

Blocklänge (BLKSIZE) × Anzahl der Puffer (BUFNØ) + 8,

aufgerundet auf die nächste durch 2048 teilbare Zahl.

6.3 Aufnahme von Moduln in die KAPRØS-Modulbibliothek

Ausgetestete KAPRØS-Moduln sollen vom Ersteller in die KAPRØS-Modulbibliothek LØAD.KSBI gebracht werden. Dazu gibt es die katalogisierte Prozedur KSUPDA. Mit ihr wird die folgende Eingabe benötigt:

```
//Jobkarte (Region: 62 K Bytes plus Assembler, Compiler oder Linkage Editor)
//  EXEC KSUPDA
[//K.FTxxFyyy DD .....]                    [Für Dateien mit Assembler-,
.....                                      Compiler- oder Linkage-Editor-
                                           Eingabe (s. 2.2)]
//K.SYSIN  DD  *
*UPDATE
nn                                         Anzahl der Moduln, ≤ 10
mmmmmmmbbeeedddbbbbbbk   }
.....                    }                 Insgesamt nn Angaben (s.u.)
mmmmmmmbbeeedddbbbbbbk   }                 für das Modulverzeichnis
*CØMPILE .....           }
.............            }
*LINK ........           }                 Eingabe des 1. Moduls (s. 2.2)
.............            }
bNAME mmmmmmm[(R)]       }
*$*$                     }
.................        }                 Eingabe des 2. bis nn-ten
                         }                 Moduls (s. 2.2)
*END
/*
//
```

Bedeutung der Angaben für das Modulverzeichnis:

mmmmmmm Modulname.

eee Anzahl der moduleigenen Dateien (999, wenn unbestimmt).

ddd Anzahl der verwendeten DB (999, wenn unbestimmt).

bbbb Benutzernummer des Erstellers.

k 0, wenn Erstaufnahme des Moduls (R auf der bNAME-Karte entfällt);
1, wenn Modifizierung des Moduls (R auf der bNAME-Karte muß stehen).

Anm.: KSUPDA-Läufe sind Aufnahmeläufe, die wegen eines Expiration Date der Modulbibliothek LØAD.KSBI nur mit Genehmigung gerechnet werden können.

6.4 Benutzerarchive

Benutzerarchive zur langfristigen Speicherung von DB können auf Platte oder Band als sequentielle Dateien mit Satzformat VBS angelegt werden. Vor seiner Verwendung muß ein Benutzerarchiv mit einem Anfangssatz (enthält 5 Worte Benutzeridentifikation und im 6. Wort die Satzlänge in Worten) und einem Endesatz (enthält in 5 Worten die Literalkonstante 'ARCHIVENDE _____') initialisiert werden. Hierzu sowie zum Ausdrucken und Zusammenschieben von Benutzerarchiven, gibt es Dienstprogramme der KAPRØS-Verwaltung /3/. Der Dateiname sollte KSA1 enthalten.

6.5 Programmkonstanten und -beschränkungen

Satzlänge der Scratch-Lifeline und der Restart-Lifeline:	766 Worte
Satzlänge des Generellen Archivs:	329 Worte
Standard-Satzzahl der Scratch-Lifeline:	150
Satzzahl der Restart-Lifeline:	3600
Satzzahl des Generellen Archivs:	4500
Maximale Schachtelungstiefe der Moduln:	22
Maximale Anzahl von Testmoduln:	22
Zeit, während der Alte Restart-DB in der Restart-Lifeline zugänglich sind (vom Startzeitpunkt des erstellenden Jobs ab zum Startzeitpunkt des zugreifenden Jobs gerechnet):	7 d
Zeit, nach der nicht mehr zugängliche Alte Restart-DB in der Restart-Lifeline gelöscht werden:	24 h
Maximale Anzahl gleichzeitig eröffneter moduleigener Dateien:	40
Maximale Anzahl von Fortran-DA-Dateien mit statischem Puffer:	10
Maximale Anzahl von Nicht-Fortran-Dateien:	5
Maximale Anzahl von mit der Systemroutine KSLØRD geladenen Moduln:	10
Reservierte Dateinummern:	40,...,47,50

Anhang: Einige Definitionen und Abkürzungen

Aktiv, aktiviert: Der Modul mit der höchsten Stufe einer Modulschachtelung ist aktiv; die anderen Moduln der Modulschachtelung sind lediglich aktiviert.

Aktueller Name: Ist ein DB im Modul σ-ter Stufe nichtlokal, so heißt der ihm beim Aufruf des Moduls σ-ter Stufe zugeordnete Blockname eines Moduls niedrigerer Stufe der aktuelle Name des DB im Modul σ-ter Stufe.

Archiv: Ein Archiv ist eine permanente Datei zur Speicherung von DB, die mit der Lifeline eines KAPRØS-Jobs ausgetauscht werden können; die gespeicherten DB heißen Archiv-DB.

Anm.: Permanent heißt, daß die DB erst bei Bedarf durch ein Dienstprogramm gelöscht werden.

> Generelles Archiv: Das Generelle Archiv ist ein von KAPRØS verwaltetes Archiv.
> Benutzerarchiv: Ein Benutzerarchiv ist ein vom Benutzer verwaltetes Archiv.

Auslagern, Rücklagern: Beim Aufruf eines Moduls wird normalerweise der rufende Modul aus dem Kernspeicher auf eine sequentielle Datei ausgelagert, um Platz für den gerufenen Modul zu machen. Nach dem Ablauf des gerufenen Moduls wird der alte Modul wieder in den Kernspeicher rückgelagert. Bei Platzmangel im Kernspeicher können ferner in der IL stehende DB auf die SL ausgelagert werden.

Blockname: Ein Blockname ist ein Konstantenpaar, bestehend aus dem einfachen Blocknamen und dem Index zum Blocknamen.

> Einfacher Blockname: Ein einfacher Blockname ist eine Literalkonstante aus maximal 16 alphanumerischen Zeichen oder Blanks, deren erstes Zeichen ein alphabetisches Zeichen ist.
>
> Anm.: Zu den alphabetischen Zeichen zählt in KAPRØS das Dollarzeichen $.

Index zum Blocknamen: Ein Index zum Blocknamen ist eine positive ganzzahlige Konstante.

Datenblock (DB): Ein Datenblock ist eine logisch zusammengehörige, aber nicht notwendigerweise physikalisch zusammenstehende, Folge von Worten, die durch einen Blocknamen gekennzeichnet ist und von KAPRØS verwaltet wird.

Anm.: Der Blockname eines Datenblocks kann in verschiedenen Moduln einer Schachtelung verschieden sein.

Teil-Datenblock (Teil-DB): Die physikalisch getrennt stehenden Teile eines Datenblocks werden als Teil-Datenblöcke bezeichnet.

Datenblock-Teil (DB-Teil): Jede logisch zusammengehörige Folge von Worten aus einem Datenblock wird als Datenblock-Teil bezeichnet.

Anm.: Der Begriff "DB-Teil" schließt als Sonderfall den Begriff "DB" mit ein. Dasselbe gilt für den Begriff "Teil-DB".

Blockdaten: Der Inhalt eines DB besteht aus den Blockdaten des DB.

Druckmodul: Ein Druckmodul ist ein Modul, der vom KSP aufgerufen wird, um Externblöcke zu drucken.

Externblock: Ein DB heißt Externblock, wenn er ein Karteneingabe-, Druckausgabe-, Archiveingabe-, Archivausgabe-, Alter oder Neuer Restart- oder Scratch-DB ist; er ist auf Stufe 0 lokal.

Interner Fehlercode: Der interne Fehlercode ist eine Variable im Systemkern, in der von den Systemroutinen gesetzte Fehlercodes festgehalten werden.

KAPRØS-Job: Ein KAPRØS-Job ist ein Job-Step im Sinne des IBM/370-Betriebssystems, in dem der Systemkern aufgerufen wird. Er besteht zeitlich im allgemeinen aus einer Compile-/Link-Phase und aus einer Go-Phase.

KAPRØS-Steuerprogramm (KSP): Das KAPRØS-Steuerprogramm ist der Teil des Systemkerns, unter dessen Regie die KAPRØS-Eingabe verarbeitet wird, die Lese-, Prüf-, Druckmoduln und der Steuermodul aufgerufen werden.

Lesemodul: Ein Lesemodul ist ein Modul, der vom KSP aufgerufen wird, um Blockdaten eines Karteneingabe-DB in die Lifeline zu lesen.

Lifeline: Der Speicherbereich, in dem alle Teil-DB eines KAPRØS-Jobs stehen, die zu einem betrachteten Zeitpunkt existieren, heißt die Lifeline des KAPRØS-Jobs zu diesem Zeitpunkt. Zur Lifeline gehören auch einige Tabellen, die als Inhaltsverzeichnis der Teil-DB in der Lifeline dienen.

Anm.: Der Begriff "Lifeline" entspricht den Begriffen "Data Pool" oder "Data Base" in anderen Programmsystemen.

Interne Lifeline (IL): Der im Kernspeicher stehende Teil der Lifeline heißt Interne Lifeline.

Externe Lifeline (EL): Der nicht im Kernspeicher stehende Teil der Lifeline heißt Externe Lifeline.

Anm.: DB können gleichzeitig in der IL und in der EL stehen.

Scratch-Lifeline (SL): Der Teil der EL, der auf einer temporären Datei des KAPRØS-Jobs steht, heißt Scratch-Lifeline.

Restart-Lifeline (RL): Der Teil der EL, der auf einer semi-permanenten allen KAPRØS-Jobs zugänglichen Datei steht, heißt Restart-Lifeline; die gespeicherten DB heißen Restart-DB.

Anm.: Semi-permanent heißt, daß die DB nach einer festgelegten Frist automatisch gelöscht werden.

Lokal, nichtlokal: Ein DB heißt lokal im Modul σ-ter Stufe, wenn er dort verwendet wird und nicht schon in einem Modul niedrigerer Stufe verwendet wurde. Ein DB heißt nichtlokal im Modul σ-ter Stufe, wenn er dort verwendet wird und schon in einem Modul niedrigerer Stufe verwendet wurde.

Anm.: Jeder DB ist demnach in genau einem Modul einer Schachtelung lokal. Im KSP gibt es nur lokale DB. "Verwenden" wird hier gebraucht im Sinne von Schreiben, Lesen, Weitergeben an einen Modul höherer Stufe, Übernehmen von einem Modul höherer Stufe.

Modul: Ein Modul ist ein abgeschlossenes, beliebig komplexes Programm, das die KAPRØS-Konventionen erfüllt und dem ein Modulname zugeordnet ist. Er ist ein Load Module im Sinne des IBM/370-Betriebssystems, in einfacher oder Overlay-Struktur, und Member eines Partitioned Dataset.

Anm.: Die Mindestkonventionen für einen Modul sind: a) Der Modul muß die Systemroutine KSINIT als Subroutine enthalten; b) Im Modul muß die Systemroutine KSINIT am Anfang aufgerufen werden. "Abgeschlossen" bedeutet, daß alle im Modul benützten Subroutinen und Commons im Modul enthalten sind, außer den im Systemkern stehenden Systemroutinen (und einigen anderen zentralisierten Routinen).

Bibliotheksmodul: Ein Bibliotheksmodul ist ein in der KAPRØS-Modulbibliothek stehender und im Modulverzeichnis festgehaltener Modul, der allen KAPRØS-Jobs zugänglich ist.

Testmodul: Ein Testmodul ist ein in der KAPRØS-Eingabe eines KAPRØS-Jobs definierter Modul, der nur für die Dauer dieses KAPRØS-Jobs exisitiert.

Modulname: Ein Modulname ist eine Literalkonstante aus maximal 6 alphanumerischen Zeichen, deren erstes ein alphabetisches Zeichen ist.

Anm.: Zu den alphabetischen Zeichen zählt in KAPRØS auch das Dollarzeichen $.

Nachrichtencode: Der Nachrichtencode ist eine Variable im Systemkern, die mit der Systemroutine KSCC gesetzt oder gelöscht werden kann, um von Modul zu Modul Nachrichten weiterzugeben.

Prüfmodul: Ein Prüfmodul ist ein Modul, der vom KSP aufgerufen wird, um Externblöcke zu prüfen.

Standardname: Wird ein DB im Modul σ-ter Stufe verwendet, so heißt der ihm dort zugeordnete Blockname der Standardname des DB im Modul σ-ter Stufe.

Steuermodul: Der Steuermodul ist der Modul, der vom KSP aufgerufen wird, wenn die KAPRØS-Eingabe verarbeitet ist und die Lese- und Prüfmoduln abgearbeitet sind.

Stufe: Werden Moduln vom KSP oder von Moduln aufgerufen, so heißt die Stellung eines Moduls in der Modulschachtelung die Stufe des Moduls (die vom KSP aufgerufenen Moduln haben die Stufe 1; die von Moduln 1. Stufe aufgerufenen Moduln haben die Stufe 2; usw.).

Systemkern: Der Systemkern ist ein Load Module im Sinne des IBM/370-Betriebssystems. Er besteht aus dem KAPRØS-Steuerprogramm und den Systemroutinen (außer KSINIT).

Systemroutinen: Die Systemroutinen sind Routinen im Systemkern (außer KSINIT), die von den Moduln aufgerufen werden können, um mit anderen Moduln wechselzuwirken, um Datenblöcke zu behandeln, um die Dateiverarbeitung zu ermöglichen und um Fehler zu behandeln.

Übertragungstechnik: Übertragungstechnik ist eine Methode zum Schreiben, Lesen oder Ändern von DB der Lifeline, in der DB oder Teile von DB von Systemroutinen aus moduleigenen Feldern in die Lifeline oder umgekehrt übertragen werden.

Zeigertechnik: Zeigertechnik ist eine Methode zum Schreiben, Lesen oder Ändern von DB der Lifeline, in der die Blockdaten des DB vom Modul direkt in die IL an die durch Zeiger bezeichnete Adresse geschrieben oder von dort gelesen werden.

Abkürzungen:
- DB - Datenblock
- EL - Externe Lifeline
- IL - Interne Lifeline
- KSP - KAPRØS-Steuerprogramm
- RL - Restart-Lifeline
- SL - Scratch-Lifeline

Literatur:

/1/ H. Bachmann, G. Buckel, W. Höbel, S. Kleinheins:
The Modular Program System KAPRØS for Efficient Management of Complex Reactor Calculations.
Proc. of Conf. on Computational Methods in Nuclear Engineering, Charleston, South Carolina, April 15-17, 1975.
CØNF-750413, Vol. II, S. V1-V10

/2/ G. Buckel, W. Höbel:
Das Karlsruher Programmsystem KAPRØS - Teil I: Übersicht und Vereinbarungen. Einführung für Benutzer und Programmierer.
KFK 2253 (1976)

/3/ H. Bachmann, S. Kleinheins:
Das Karlsruher Programmsystem KAPRØS - Teil II:
Dokumentation des Systemkerns.
KFK 2254 (1976)