KfK 3702 B
April 1984

Das Programm zur Überwachung und Steuerung der Vakuumanlage des Karlsruher Magnetspektrographen „Little John“

D. Manger
Institut für Kernphysik

Kernforschungszentrum Karlsruhe

KERNFORSCHUNGSZENTRUM KARLSRUHE

Institut für Kernphysik

KfK 3702 B

DAS PROGRAMM ZUR ÜBERWACHUNG UND STEUERUNG DER VAKUUMANLAGE DES KARLSRUHER MAGNETSPEKTROGRAPHEN "LITTLE JOHN"

D. Manger

Kernforschungszentrum Karlsruhe GmbH, Karlsruhe

Kernforschungszentrum Karlsruhe GmbH
ISSN 0303-4003

Zusammenfassung

Es wird ein in FORTRAN 4 geschriebenes, auf einer NOVA 2 mit 64 kByte Hauptspeicher lauffähiges Multitasking-Programmsystem beschrieben, das über CAMAC den Istzustand, die Schaltbefehle und die Zustandsänderungen einer komplexen Vakuumanlage überwacht sowie Prozeduren zum automatischen Ein- und Ausschalten der Anlage enthält.

THE CONTROL PROGRAM FOR THE VACUUM SYSTEM OF THE KARLSRUHE MAGNETIC SPECTROGRAPH "LITTLE JOHN"

Summary

A real-time, multi-tasking program system written in FORTRAN 4 and running on a NOVA 2 computer with a main memory capacity of 64 kB is described which is intended to surveille the actual state, the manual control commands and the changes of state of a complex high vacuum installation via CAMAC and to perform automatic coldstart and shutoff procedures.

Inhalt

1. Einleitung

Nach dem gegenwärtigen Stand der Technik ist es üblich, Vakuumpumpen von Hand zu fahren und gegen Fehlbedienungen durch Hardware-Verriegelungen zu schützen. Nach diesem Konzept werden nicht nur einzelne Laborpumpstände, sondern z.B. auch die umfangreichen Vakuumsysteme des Karlsruher Isochronzyklotrons und des Strahlführungssystems betrieben.

Der Neuaufbau einer größeren Experimentiereinrichtung wie des Magnetspektrographen "Little John" /1/ mit einem komplexen Vakuumsystem, das im Gegensatz zu den genannten Vakuumanlagen nicht ausschließlich von Stammpersonal, sondern intentionsgemäß auch von Gästen bedient werden soll, bot die Möglichkeit, erstmals eine Reihe von Gesichtspunkten zu spezifizieren und in konsequenter Weise zu verwirklichen, die vornehmlich die Betriebssicherheit der Anlage betreffen. Dazu gehören u.a. echte, simultane Zustandsmeldungen aller Ventile, Pumpen und Vakuummeßstellen sowie eine Software-Überwachung sowohl des Anfangszustandes der Vakuumanlage als auch aller Befehle und Zustandsänderungen. Verglichen mit gelöteter Logik hat ein Rechnerprogramm den Vorteil größerer Flexibilität, d.h. leichterer Änderungsmöglichkeit bei Anlagenerweiterungen, wie sie de facto bereits während der Installationsphase eintraten, sowie bei Änderung von Überwachungskriterien. Voraussetzung für eine derartige Rechnerüberwachung sind nicht nur manuell, sondern auch elektrisch setzbare Befehlsgeber.

Für eine Prozeßsteuerung der geschilderten Art gibt es u.W. bisher kein Vorbild. Eine wesentliche Vorarbeit bestand daher in der Erarbeitung zunächst eines verbalen Konzepts, das in Entscheidungstabellen fixiert wurde und dessen logische Richtigkeit und programmiertechnische Durchführbarkeit an einem reduzierten Modellfall demonstriert wurde /2/. Anschließend wurde ein konzeptionell verbesserter und auf die Gesamtanlage ausgedehnter Programmentwurf erstellt /3/ und schrittweise implementiert. Als Rechner stand ein Kleinrechner (NOVA 2) mit nur 64 KByte Hauptspeicher zur Verfügung. Vor Ablauf des Jahres 1983 konnte erstmals das lauffähige Programm in seiner Gesamtheit vorgestellt werden. In Ref. /4/ ist die Hard- und Software der Vakuumanlage im Zusammenhang dargestellt.

Im folgenden werden die Programmarchitektur und der Ablauf der einzelnen Programmeinheiten beschrieben. Bezüglich der Software-Tools, der Programmiergesichtspunkte und der Begründung der anwendungsspezifischen Programmlogik wird auf Ref. /3/ verwiesen. Mit Ref. /5/ steht eine kurzgefaßte Benutzeranleitung zur Verfügung.

2. Hardware-Gegebenheiten

Die Bezeichnungen der einzelnen Aggregate der Vakuumanlage sind aus Abb. 1 ersichtlich. Tab. 1 gibt die vollständigen Pinbelegungen der Steckverbindungen der CAMAC-Input- und Output-Module an. Es bedeuten: "B" = Befehle, "M" = Meldungen, "T_" = Betriebsbereitschaft der Diffusionspumpen, "A" = Schaltpegel, "$\overline{R}$" = Resetpegel.

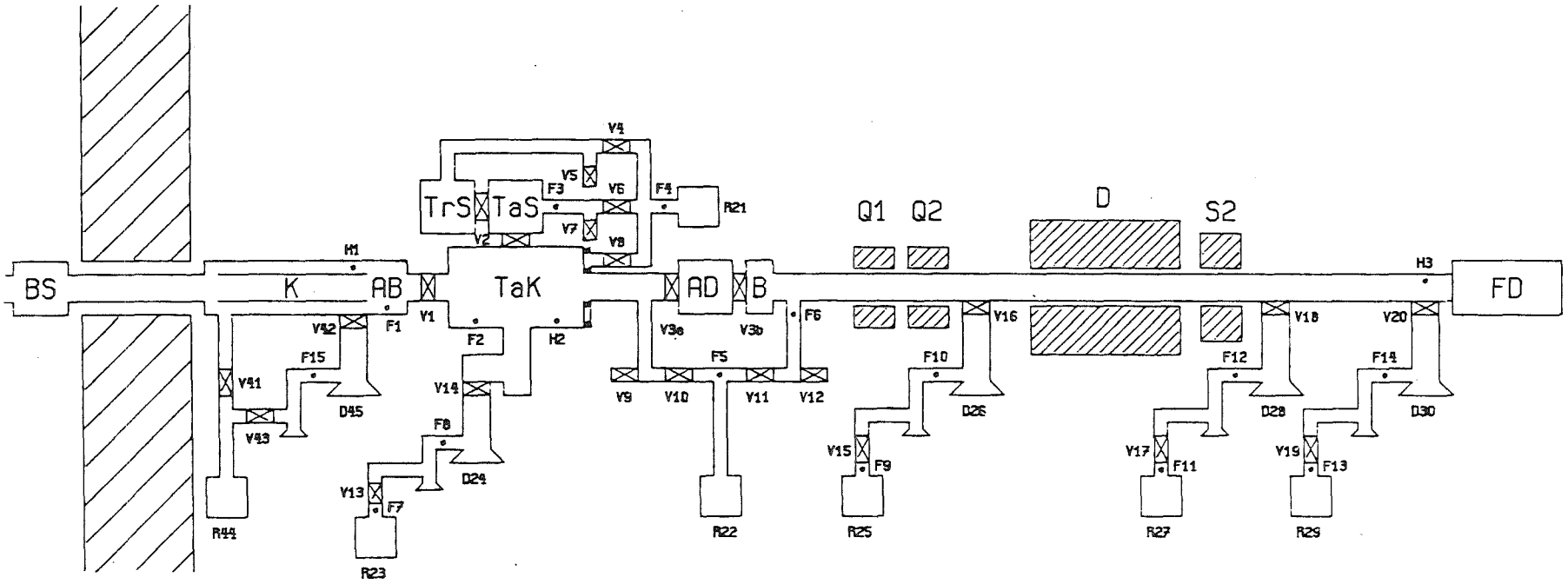

Abb. 1: Schema der Vakuumanlage

V: Ventil
R: Rotationspumpe
D: Diffusionspumpe
F: Feinvakuummeßstelle
H: Hochvakuummeßstelle

BS: Beamstop
K: Kryofalle
AB: Aktive Blende
TaK: Targetkammer
TaS: Targetschleuse

TrS: Transportschleuse
AD: Akzeptanzdetektor
B: Passive Blende
FD: Fokalebenendetektor

Tab. 1: Pin-Belegungen der CAMAC-Module

Pin	CAMAC-Module 3473					CAMAC-Module 3270	
I =	L = 1	L = 2	L = 3	L = 4	L = 5	L = 1	L = 2
1	B V1	B D24	M V1	M D24	F1 <	A V1	R V1
2	B V2	B D26	M V2	M D26	F2 <	A V2	R V2
3	B V3	B D28	M V3	M D28	F3 <	A V3	R V3
4	B V4	B D30	M V4	M D30	F4 <	A V4	R V4
5	B V5	B D45	M V5	M D45	F5 <	A V5	R V5
6	B V6	B R21	M V6	M R21	F6 <	A V6	R V6
7	B V7	B R22	M V7	M R22	F7 <	A V7	R V7
8	B V8	B R23	M V8	M R23	F8 <	A V8	R V8
9	B V9	B R25	M V9	M R25	F9 <	A V9	R V9
10	B V10	B R27	M V10	M R27	F10 <	A V10	R V10
11	B V11	B R29	M V11	M R29	F11 <	A V11	R V11
12	B V12	B R44	M V12	M R44	F12 <	A V12	R V12
13	B V13		M V13		F13 <	A V13	R V13
14	B V14		M V14		F14 <	A V14	R V14
15	B V15		M V15		F15 <	A V15	R V15
16	B V16		M V16			A V16	R V16
17	B V17	RE/HA	M V17	M T24	H1 <	A V17	R V17
18	B V18	QUITT.	M V18	M T26	H2 <	A V18	R V18
19	B V19		M V19	M T28	H3 <	A V19	R V19
20	B V20		M V20	M T30		A V20	R V20
21	B V41		M V41	M T45		A V41	R V41
22	B V42		M V42			A V42	R V42
23	B V43		M V43			A V43	R V43
24							
25							
26						A D24	R D24
27						A D26	R D26
28						A D28	R D28
29						A D30	R D30
30						A D45	R D45
31						A R21	R R21
32						A R22	R R22
33						A R23	R R23
34						A R25	R R25
35						A R27	R R27
36						A R29	R R29
37						A R44	R R44
38						RE.BEREIT	
39						READY	
40						VERRIEGELT	
41						RE.-REAKT.	
43							
:							
50							

3. Programmarchitektur

Im Zuge der Implementierungsarbeiten stellte es sich als notwendig heraus, die in Ref. /3/ vorgeschlagene Programmarchitektur in einigen Punkten zu ändern. Hierfür waren vornehmlich 2 Gründe maßgeblich:

a) Das Programmsystem hatte bereits in der (in Ref. /3/ definierten) Ausbaustufe "a" (Anlagenabschnitt "Targetkammer" allein) die für Programme insgesamt zur Verfügung stehende Speicherkapazität von ca. 36 KByte nahezu erreicht.

b) Die Protokollausgaben auf Terminalschirm und Drucker sowie die zahlreichen IF-Abfragen führten zu Abbruch durch STACK OVERFLOW, da u.a. für jedes WRITE-Format und für jedes IF-Statement Run-Time-Stack benötigt wird.

Zur Erläuterung: In einem Multi-Tasking-Programm wird der Run-Time-Bereich in gleich große "Segmente" aufgeteilt, deren Anzahl gleich der Anzahl der Tasks ist. Außer dem Run-Time-Stack enthält jedes Segment noch einen "SP-Stack" und einen "NUMBER-Stack". Alle globalen Variablen und Felder, die nicht im COMMON-Bereich stehen, werden im Run-Time-Stack abgespeichert. Hierzu gehören außer den oben genannten Fällen auch z.B. die Argumente bei Unterprogrammaufrufen.

Einsparungen von Programmspeicherplatz und Run-Time-Stack konnten durch folgende Maßnahmen erreicht werden:

3.1 Untergliederung in Hauptprogramme

Im Gegensatz zu dem in Ref. /3/ beschriebenen Entwurf enthält das Programmsystem (Abb. 2) jetzt 4 Hauptprogramme:

LJ1 (Initialisierung von Anlagen- und Programmparametern),
MALJ2 (Automatischer Kaltstart der Vakuumanlage),
LJ2 (Überwachung und Steuerung),
VA1 (Anfangsprüfung).

Im folgenden werden zunächst LJ1, LJ2 und VA1 behandelt. Die Kaltstartprozedur MALJ2 wird in Abschn. 4.4 beschrieben.

Während der Übergang von LJ1 nach MALJ2 bzw. nach LJ2 durch Chaining erfolgt (Einbahnstraße!), geschieht der Wechsel von LJ2 nach VA1 und umgekehrt durch die Statements CALL SWAP und CALL BACK. Durch die Herausnahme der speicherplatzaufwendigen Anfangsprüfung aus LJ2 wurde der Hauptspeicher um nahezu 4 KByte entlastet. Dieser Gewinn wird allerdings dadurch erkauft, daß der Rechner für die Dauer der Anfangsprüfung für andere Aufgaben, z.B. Überwachung der Magneterregungen oder Positionierungen, blockiert ist.

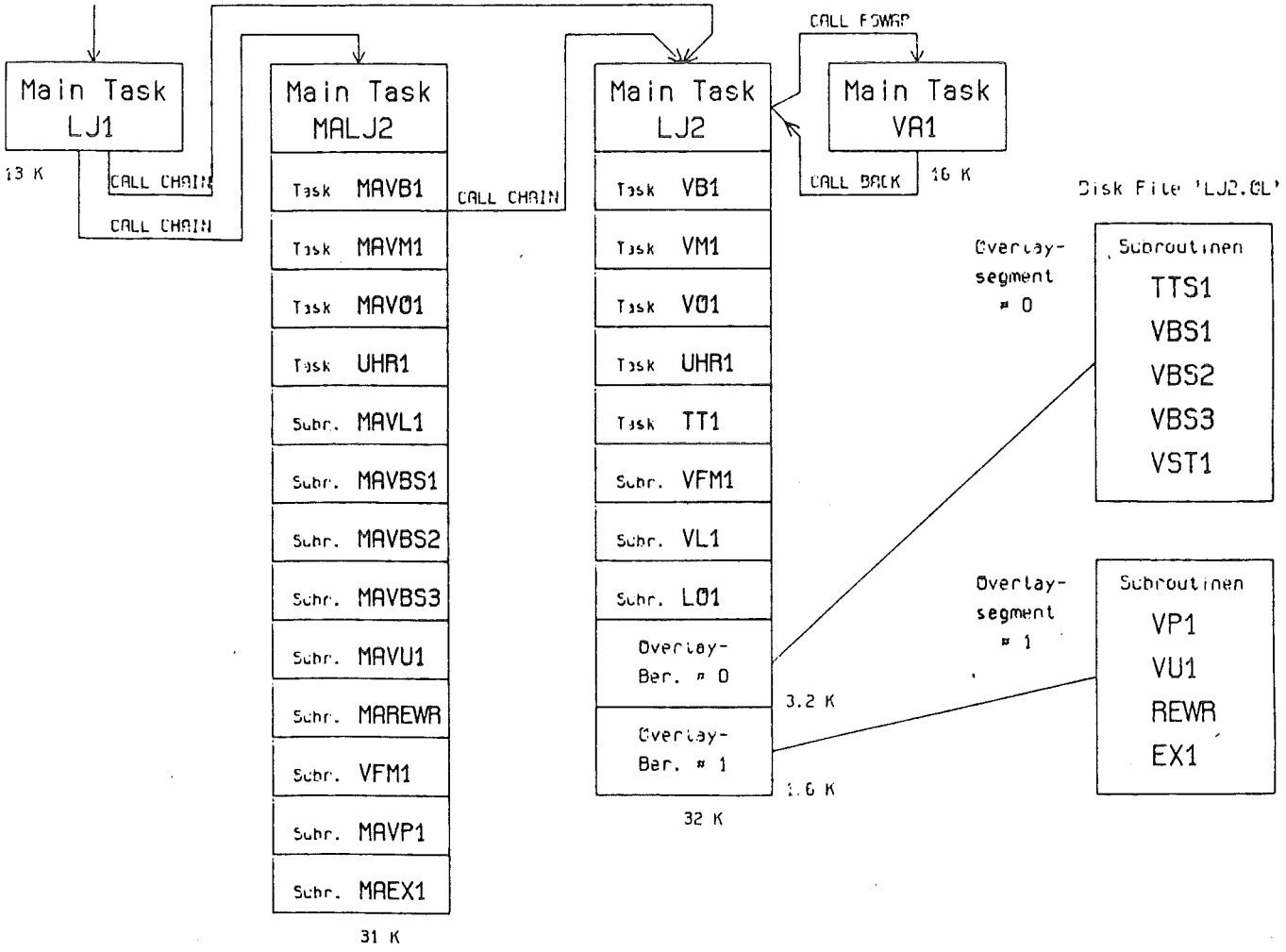

Abb. 2: Struktur des Programmsystems zur Steuerung der Vakuumanlage des Magnetspektrographen Little John. (Alle Speicherplatzangaben in Bytes für die gebundenen Programme)

LJ1 und VA1 sind Single-Tasking-Programme, LJ2 hingegen ist ein Multitasking-Programm mit Overlay-Struktur, siehe Abb. 2.

Alle Tasks wurden von der Programmlogik her optimiert, wodurch ebenfalls Speicherplatz eingespart werden konnte. Weiterhin wurden nach Möglichkeit zeitunkritische Programmteile separiert und als Overlays deklariert.

3.2 Overlay-Struktur

Die Größe der (des) vom Betriebssystem im Hauptspeicher angelegten Overlay-Bereiche(s) richtet sich nach dem größten Overlay-Programm; dabei wird jeweils ein Vielfaches von 256 2-Byte-Worten reserviert. Um den Hauptspeicherbedarf klein zu halten, ist daher darauf zu achten, daß die Größe des größten Overlays in einem Segment möglichst klein ist. Unter Berücksichtigung programmlogischer Restriktionen, die eine Unterteilung in beliebig viele und beliebig kleine Overlays verbieten, wurden gegenüber dem ursprünglichen Vorschlag /3/ folgende Verbesserungen in der Overlay-Struktur durchgeführt:

a) Die Befehlsprüfung, die ursprünglich als eine einzige Programmeinheit konzipiert war, wurde in eine (hauptspeicherresidente) Service-Task VB1 und in 3 Overlay-Subroutinen VBS1, VBS2 und VBS3 aufgeteilt, die jenachdem, von welchem CAMAC-Input-Modul der Befehls-Interrupt kam, von der Task VB1 (mittels LO1) geladen werden.

b) In gleicher Weise wurde die Bedienung von Tastatur-Interrupts zerlegt in die (hauptspeicherresidente) Service-Task TT1 sowie die Overlay-Subroutinen TTS1 (Kommando-Eingabe) und VST1 (Statusausgabe), wobei wiederum nur die Service-Task asynchron angesprungen wird.

c) Das Overlay-Segment # 1 wurde um die Overlay-Subroutine REWR (READ/WRITE) erweitert, welche die im COMMON stehenden Parameter auf Platte schreibt oder von Platte liest.

Zur Synchronisation der Overlay-Programme, die bekanntlich nicht überschrieben werden dürfen, ehe sie vollständig abgelaufen sind, wurden in der Laderoutine LO1 Flaggen eingeführt, weil die vom System angebotenen Synchronisationshilfen nicht zuverlässig funktionierten.

Insgesamt erbrachte die Verbesserung der Overlay-Struktur eine Entlastung des Hauptspeichers um nahezu 3 KByte. LJ2 belegt 32 KByte, sodaß für eventuelle spätere Zusatzaufgaben noch 4 ca. KByte Hauptspeicher verfügbar sind.

3.3 Task-Prioritäten

Die Prioritäten der einzelnen Tasks von LJ2 wurden entsprechend dem Konzept in Ref. /3/ beibehalten:

Task-Name	Task-Identifikations-Nr.	Task-Priorität
LJ2	-	0 (defaultmäßig zugeteilt)
VM1	1	10
VO1	2	20
VB1	3	30
TT1	4	40
UHR1	5	50

Diese Prioritätenstaffelung, die während des gesamten Programmablaufs unverändert bleibt, garantiert, daß die Abarbeitung eines Anlagen-Interruptes (VM1) nicht durch die Abarbeitung eines manuellen Interrupts (VB1, TT1), und daß die Abarbeitung eines Befehls-Interrupts (VB1) nicht durch die Abarbeitung eines TT-Interrupts (TT1) unterbrochen werden kann.

4. Programmbeschreibungen

Die im folgenden beschriebene Ausbaustufe des Programms bezieht sich auf die Steuerung der kompletten, in Abb. 1 dargestellten Vakuumanlage des Magnetspektrographen Little John. Die hierfür gültige Entscheidungslogik für die Befehlsprüfung, die Abschaltautomatik und die Anfangsprüfung ist den Tabellen 2 (S. 21), 3 (S. 51) und 4 (S.69) zu entnehmen, die eine Erweiterung der entsprechenden Tabellen 2, 3 und 4 aus Ref. /6/ um die neu hinzugekommenen Pumpsektionen D26 und D30 darstellen.

Jeder einzelnen Programmbeschreibung ist als Anhang das betreffende Programmlisting beigefügt. Die maßgeblichen Statements sind durch Zeilenkommentare erläutert.

4.1 MAIN-Task LJ1 (Anlagendefinition)

Das Programmsystem LJ1-LJ2-VA1 mit seinen Unterprogrammen (Tasks, Subroutinen) ist so ausgelegt, daß die dem Experimentator zugänglichen und eventuell von ihm veränderbaren Anlagenparameter als globale, extern gespeicherte Variable geführt werden, die gegebenenfalls nur an einer einzigen Stelle, u.z. in den Deklarations-Statements von LJ1, umzudefinieren sind. Es sind dies die folgenden "primären" Anlagenparameter:

MASKE(2,3) enthält an den Bitpositionen eine 1, die den Pins der Steckverbindungen der CAMAC-Input-Module #3 bis #5 (Tab. 1, S. 3) entsprechen, wenn das betreffende Aggregat bzw. die betreffende Meßstelle existiert bzw. von LJ2 berücksichtigt werden soll, sonst eine 0;

ICR(2) die (an den CAMAC-Crate-Kontrollern einstellbaren) Nummern der CAMAC-Crates im Rack des E.R. II und im Rack # 7 in der Vorhalle);

N3473(5) die Stationsnummern der 5 CAMAC-Input-Module (Kinetic Systems Mod. 3473) im Rack # 7 in der Vorhalle);

N3072(2) die Stationsnummern der 2 CAMAC-Output-Module (Kinetic Systems Mod. 3072) im Rack # 7 in der Vorhalle);

NUHR die Stationsnummer der CAMAC-Uhr im Rack des E.R. II.

Hinweis: Nach einer eventuellen Änderung der Initialisierungswerte muß LJ1 neu kompiliert

FORT LJ1 ↓ und gelinkt

LLJ1 ↓ werden.

Als endgültig und unveränderlich werden hingegen die festverdrahteten Pinbelegungen der CAMAC-Input- und Output-Steckverbindungen nach Tab. 1, S. 3, angesehen; eine Änderung der Pinbelegungen würde tiefgreifende Programmänderungen erforderlich machen.

LJ1 berechnet aus den oben genannten primären Anlagenparametern die von MALJ2, LJ2 und VA1 häufig gebrauchten REAL-Adressen der CAMAC-Submodule als "sekundäre" Anlagenparameter. Aus Gründen der Speicherökonomie werden LJ2 und VA1 von der Berechnung dieser abgeleiteten Größen entlastet.

Außer den primären und sekundären Anlagenparametern wird in LJ1 noch der globale LOGICAL-Programmparameter LPT definiert, mit dessen Hilfe das Line-Printer- (Schnelldrucker-) Protokoll unterdrückt werden kann. Weitere, intern benutzte Programmparameter siehe Seite 9, 10 und 13.

Der Programmablauf von LJ1 kann wie folgt skizziert werden:

* Deklarationen und Initialisierungen (mit Ausnahme von SIAB, s. unten)

* Aufforderung zur Eingabe, ob die Pumpsektion D30 vorhanden ist (wenn nicht, werden die Meldungen V19, V20, R29, D30, T30, F13, F14 maskiert), ob die Targettransportschleuse angeschlossen ist (wenn nicht, werden die Meldungen V4 und V5 maskiert) sowie des LOGICAL-Parameters LPT;

* Zeittakt der Real-Time-Clock verifizieren (falls nicht 1 oder 10 oder 100 msec, erfolgt Abbruch);

* Löschen eventuell gesetzter, (für alle Module eines Crates geltender) Inhibit-Bits (d.s. Sperren) für Crate # 1 und # 2; Disablen des branch demand outputs für Crate # 1 und # 2;

* Überprüfen, ob sich die beiden 3072-Output-Module an ihren vorgesehenen Plätzen (Stationen im Crate) befinden; (die 3473-Input-Module haben keinen Funktionscode für eine derartige Identifikationsprüfung);

* REAL-Adressen der CAMAC-Submodule berechnen; CAMAC-Uhr disablen;

* Anlagen- und Programm-Parameter in den Platten-File "VPA.DA", Channel Nr. 1, schreiben; (die Datenübergabe mittels externer Zwischenspeicherung ist hier unvermeidlich, weil die Kommunikation zwischen geswappten und gechainten Programmen durch (unlabeled) Common nur möglich ist, wenn alle Programme entweder single-tasking- oder multitasking-Programme sind, nicht aber eine Kombination aus beiden.)

* Aufforderung zur Eingabe, ob zu Beginn das automatische Kaltstartprogramm MALJ2 gestartet werden soll.

* MALJ2 bzw. LJ2 laden (CALL CHAIN); wenn CALL CHAIN erfolgreich war, läßt sich die (in der Subroutine CHAIN und in den meisten CAMAC-Subroutinen als letztes Argument vorkommende) Erfolgsvariable IER nicht mehr abfragen, da LJ1 bereits durch MALJ2 bzw. LJ2 überschrieben ist; daher ist die Abfrage IF(IER.NE.1)... für den Fehlerfall entbehrlich.

"VPA.DA" ist ein Block (Record) von 512 Bytes mit folgendem Inhalt:

MASKE(2,3)	definiert die existierenden Aggregate / Meßstellen, siehe oben (im DATA-Statement zunächst m i t Berücksichtigung der Pumpsektion D30 und der Targettransportschleuse)
A3473(2,5)	REAL-Adressen der 5 Input-Submodule zu IA=0 und IA=1, siehe oben (IA ist die A-Adresse des CAMAC-CNAF-Kommandos)
A3072(2,2)	REAL-Adressen der 2 Output-Submodule zu IA=0 und IA=1, siehe oben
AUS(2,2)	REAL-Darstellung der an CAMAC auszugebenden Worte zu IA=0 und IA=1
IALT(2,5)	INTEGER-Darstellung der Bitmuster der 5 Input-Module vor Eintreffen eines Interrupts
INEU(2,5)	INTEGER-Darstellung der Bitmuster der 5 Input-Module nach Eintreffen eines Interrupts
IAUS(2,2,2)	INTEGER-Darstellung der an die Output-Module # 1 und # 2 (3. Dimension), Submodule # 1 (Pin 1 bis 24) und # 2 (Pin 26 bis 29) (2. Dimension) in jeweils 2 16-Bit-INTEGER-Worten (1. Dimension) auszugebenden Pegel
N3473(5)	Stationsnummern der 5 Input-Module, siehe oben
MIV(5)	Vorwahlzeiten (in min) der 5 Stoppuhren, siehe 4.2.5

IDT(5)	IDT(1) Monat IDT(2) Tag IDT(3) Jahr IDT(4) und IDT(5) (unbenutzt)
ACR(2)	REAL-Adressen zu N=30, IA=10 der beiden CAMAC-Crates am E.R. II und in der Vorhalle, siehe oben
AUHR	REAL-Adresse zu IA=0 der CAMAC-Uhr, siehe oben
ICR(2)	Die an den CAMAC-Crate-Kontrollern am E.R. II und in der Vorhalle einstellbaren Nummern, siehe oben
N3072(2)	Stationsnummern der beiden Output-Module, siehe oben
NUHR	Stationsnummer der CAMAC-Uhr im Crate am E.R. II, siehe oben
IC	Interrupt-Code
IFDEL	Taktfrequenz der Systemuhr in 1/sec
LPT	LOGICAL-Flagge zur Unterdrückung des Protokolls auf dem Schnelldrucker, siehe oben
STAT	LOGICAL-Flagge zur Rückmeldung von TTS1 (4.2.2.1) an TT1 (4.2.2), ob das Kommando 'VSTA' (Status-Ausgabe) gegeben wurde
IHA	LOGICAL-Flagge (.TRUE. bei Handbetrieb)
SIAB	LOGICAL-Flagge zur Unterscheidung, ob mit VO1 (4.2.4) die Sicherheitsabschaltung oder eine gezielte Abschaltung eingeleitet werden soll; SIAB wird erst in LJ2 initialisiert
IDEL(2,2)	Den Worten IAUS(1,1,1), IAUS(2,1,1), IAUS(1,2,1) und IAUS(2,2,1) korrespondierende INTEGER-Worte, deren Bits bei der Ausgabe von IAUS() für die Dauer von 1 sec = 1 gesetzt werden, um VM1 (4.2.3) zu ermöglichen, zwischen erwarteten und spontanen Zustandsänderungen zu unterscheiden
MAPRO(60)	Die ersten 40 Elemente von MAPRO() werden in LJ1 mit den Aggregatnummern (1,2,...,45), die folgenden 6 Elemente mit den Strings 'T', 'V', 'D', 'R', 'F', "H" initialisiert; die restlichen Elemente bleiben unbenutzt.MAPRO() dient zur Vereinfachung der FORMAT-Statements bei der Protokoll- und Status-Ausgabe.
MAFLA(12)	INTEGER-Flaggen (Werte 0 oder 1) zur Programmsteuerung: MAFLA(1) zeigt an, ob der Overlay-Bereich # 0 frei ist, siehe 4.2.7 MAFLA(2) zeigt an, ob der Overlay-Bereich # 1 frei ist, siehe 4.2.7 MAFLA(3) übergibt das Ergebnis der Anfangsprüfung (PERMISSIBLE / PROHIBITED) von VA1 (4.3) an LJ2 (4.2) MAFLA(4) Sperre für Befehlsprüfung bei Ausgabe (über Output-Modul # 2) von RESET-Impulsen (RESET-Pegel Setzten und nach 0.1 wieder Rücksetzen) MAFLA(5) bewirkt, daß VM1 (4.2.3) während des Ablaufs von VO1 (4.2.4) und MALJ2 (4.4) übersprungen wird MAFLA(6) bricht VO1 (4.2.4) ab; wird durch Drücken einer der Funktionstasten F1 - F5 gesetzt und in VO1 wieder gelöscht MAFLA(7) bis MAFLA(12) (unbenutzt)
MABIM(5)	Minutenzähler für die 5 Diffusionspumpen (werden in UHR1 hochgezählt und in VO1 gelöscht)

```
C
C       ***************************************************************
C       *  MAIN LJ1                                    MANGER JANUAR 1984  *
C       *  INITIALISIERUNG ALLER KONSTANTEN UND VARIABLEN ANLAGEN UND      *
C       *  EXPERIMENTPARAMETER                                             *
C       ***************************************************************
        COMPILER NOSTACK
        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
        INTEGER LWORT,MWORT
        DIMENSION IDAT(3),MASK30(2,3)
        LOGICAL LPT,STAT,IHA,SIAB
        LPT=.FALSE.

        DATA MASKE/177K,177777K,37K,7777K,7K,77777K/
        DATA MASK30/163K,177777K,27K,5767K,7K,47777K/
        DATA NUHR/20/,ICR/1,2/,N3072/9,10/,N3473/15,16,17,18,19/,MIV/5*0/
        DATA MAPRO/1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,41,42,43,
     *             24,26,28,30,45,21,22,23,25,27,29,44,24,26,28,30,
     *             "T","V","D","R","F","H"/

C       ========================  TERMINALEINGABE  ====================================
10      WRITE(10,1000)
1000    FORMAT("                          D30-SUBSECTION CONNECTED?   (YES/NO):",Z)
        READ(11,1050) MWORT
1050    FORMAT(A2)
        IF(MWORT.EQ."<131><105>") GOTO 25
        IF(MWORT.EQ."<116><117>") GOTO 20
        GOTO 10                                          ; FALSCHE EINGABE
20      DO 21 L=1,3
        DO 22 K=1,2
        MASKE(K,L)=(MASKE(K,L).AND.MASK30(K,L))
22      CONTINUE
21      CONTINUE

25      WRITE(10,1060)
1060    FORMAT("                          TRANSPORT SLUICE CONNECTED? (YES/NO):",Z)
        READ(11,1050) MWORT
        IF(MWORT.EQ."<131><105>") GOTO 30
        IF(MWORT.EQ."<116><117>") GOTO 26
        GOTO 25                                          ; FALSCHE EINGABE
26      CALL ICLR(MASKE(2,1),3)
        CALL ICLR(MASKE(2,1),4)

C       ===========================================================================
30      WRITE(10,1100)
1100    FORMAT("                          LINE PRINTER PROTOCOL?      (YES/NO):",Z)
        READ(11,1050) LWORT
        IF(LWORT.EQ."<131><105>".OR.LWORT.EQ."<116><117>") GOTO 40
        GOTO 30                                          ; FALSCHE EINGABE
40      IF(LWORT.EQ."<131><105>") LPT=.TRUE.

C       ==============     REALTIME-CLOCK VERIFIZIEREN ============================
        CALL GFREQ(IFREQ)
        IF(IFREQ.GE.1.AND.IFREQ.LE.3) GOTO 100
        TYPE "*** LJ1: ILLEGAL REALTIME-CLOCK FREQUENCY<7>",IFREQ
        GOTO 200
100     IFDEL=10**(IFREQ-1)

C       ========== GEMEINSAME STEUERSIGNALE FUER ALLE CAMAC MODULE==============
        DO 105 I=1,2
        CALL CAL92(A,ICR(I),30,9)          ; REMOVE DATAWAY INHIBIT
        CALL CAL93(24,A,DUMMY,IQ)          ;    "       "       "
```

```
C
      CALL CAL92(A,ICR(I),30,10)          ; CAMAC BRANCH DEMAND OUTPUT
105   CALL CAL93(24,A,DUMMY,IQ)           ;        DISABLEN

C     ================  3072 - MODULE IDENTIFIZIEREN  ==============================

      DO 110 I=1,2
      CALL CAL92(A,ICR(2),N3072(I),15)
      CALL CAL93(1,A,DUMMY,IQ)
      IF(IQ.EQ.1) GOTO 110
      TYPE "*** LJ1 : OUTPUTREGISTER  3072 #",I," NOT ON PLACE<7>"
      GOTO 200
110   CONTINUE

C     ================   ADRESSEN DER CAMAC-SUBMODULE   ============================

      DO 120 L=1,5                                      ; L= INPUT MODUL 1 BIS 5
      DO 120 K=1,2                                      ; K= 1(A=0) INPUTREGISTER
120   CALL CAL92(A3473(K,L),ICR(2),N3473(L),K-1)        ; K= 2(A=1) MEMORYREGISTER

      DO 130 L=1,2                                      ; L= OUTPUT MODUL 1 UND 2
      DO 130 K=1,2                                      ; K= 1 AUSGABE PIN 1-24
130   CALL CAL92(A3072(K,L),ICR(2),N3072(L),K-1)        ; K= 2 AUSGABE PIN 26-49

      CALL CAL92(AUHR,ICR(1),NUHR,0)                    ;DISABLE CAMAC UHR
      CALL CAL93(24,AUHR,0,IQ)                          ;   "     "    "
      CALL CAL92(ACR(1),ICR(1),30,10)                   ;CNA ADRESSE FUER BRANCH-
      CALL CAL92(ACR(2),ICR(2),30,10)                   ;DEMAND OUTPUT DIS/ENABLEN

C     ================   PARAMETER AUF PLATTE SCHREIBEN  ============================

      CALL OPEN(1,"VPA.DA",2,IER)                       ; FILE  OEFFNEN
      IF(IER.EQ.1) GOTO 140
      TYPE "*** LJ1 : FILE VPA.DA NOT OPENED<7>",IER
      GOTO 200

140   CALL WRBLK(1,1,MASKE(1,1),1,IER)                  ; PARAMETER SCHREIBEN
      IF(IER.EQ.1) GOTO 150
      TYPE "*** LJ1 : PARAMETERS ARE NOT WRITTEN ON FILE VPA.DA<7>",IER
      GOTO 200

150   CALL CLOSE(1,IER)
      IF(IER.EQ.1) GOTO 160
      TYPE "*** LJ1 : FILE VPA.DA NOT CLOSED<7>",IER
      GOTO 200
160   WRITE(10,1300)
1300  FORMAT("                          AUTOMATIC COLD STARTING?    (YES/NO):",Z)
      READ(11,1350) MWORT
1350  FORMAT(A2)
      IF(MWORT.EQ."<131><105>") GOTO 170
      IF(MWORT.EQ."<116><117>") GOTO 180
      GOTO 160
170   CALL CHAIN("MALJ2.SV",IER)
      GOTO 190
180   CALL CHAIN("LJ2.SV",IER)
190   TYPE "*** LJ1: MALJ2/LJ2 NOT CHAINED",IER
200   STOP
      END
```

IDUM(70)	IDUM(1) - IDUM(15)	Minutenzähler für die Feinvakuummeldungen F1 - F15
	IDUM(16) - IDUM(70)	Dummy-Feld zum Auffüllen des Records auf 512 Bytes. (Zum Schreiben und Lesen dienen die NOVA-FORTRAN-IV-Runtime-Subroutinen WRBLK bzw. RDBLK.)

4.2 MAIN-Task LJ2 (Interrupt-Erkennung)

LJ2 übernimmt von LJ1 über den Platten-File "VPA.DA", Channel-Nr. 1, sämtliche in 4.1 aufgeführten Anlagen- und Programmparameter und speichert sie zwecks Weitergabe an die Unterprogramme (Tasks und Subroutinen) von LJ2 in einen benannten COMMON-Bereich (COMMON-Name /PRM/), der in LJ2 und in allen seinen Unterprogrammen den gleichen Aufbau wie "VPA.DA" hat, siehe 4.1.

In den zum Multitasking-Programm LJ2 gehörenden Tasks wird für die REAL-Adressen der Submodule und für die die Bitmuster haltenden INTEGER-Worte und deren Indizes folgende Nomenklatur benutzt (die Zahlen vor und hinter den Doppelpunkten bedeuten die kleinsten und die größten Werte, die die betreffenden Indizes annehmen können):

A3473 (K,L)	Wertebereich: J = ,	K = 1:2,	L = 1:5
INEU (J, L)	1:2	(1)	1:5
IALT (J, L)	1:2	(2)	1:5
MASKE (J, L-2)	1:2	(1)	3:5
A3072 (K,L)		1:2	1:2
IAUS (J,K,L)	1:2	1:2	1:2
IDEL (J,K)	1:2	1:2	

Hierbei bezeichnet:

* J den Index des INTEGER-Wortes (siehe Ref. /3/):
  J = 1 für Pin 17 bis 24 (3473-Input-Module) bzw.
  " " 17 " 24 und Pin 42 bis 49 (3072-Output-Module) und
  J = 2 " " 1 " 16 (3473-Input-Module) bzw.
  " " 1 " 16 und Pin 26 bis 41 (3072-Output-Module);

* K die Adresse des Submoduls
  für 3473-Input-Module: K = 1 für IA = 0 (Input-Register)
  " " " " : 2 " 1 (Memory-Register
  " 3072-Output " : 1 " Ausgabe auf Pins 1 bis 24
  " " " " : 2 " " " " 26 " 49

* L die laufende Nr. der Input-/Output-Module innerhalb des Crates # 2 (von links nach rechts)

Der Laufindex für die Bit-Nr. in den INTEGER-Worten (mit 1 beginnend) wird mit I, die Bit-Nr. selbst (mit 0 beginnend) demnach mit (I-1) bezeichnet.

LJ2 gliedert sich in folgende Programmschritte:

* Overlay-File ("LJ2.OL"; Channel-Nr. 2) öffnen;

* Datum, Uhrzeit und Nachricht "LJ2 LOADED" ausgeben;

* Parameter von Platten-File ("VPA.DA"; Channel-Nr. 1) lesen;

* Subtasks aktivieren (CALL ITASK) einschließlich der TT-Task (CALL TTIT) und sofort wieder suspendieren (CALL HOLD); über die Argumente von ITASK erhalten die Tasks die bereits in 3.3 genannten Identifikationsnummern und Prioritäten.

* Alle (5) 3473-Input-Register lesen, -Memory-Register updaten und -LAM's löschen (Subroutine VL1, 4.2.6);

* Die Ist-Zustände "M" der Aggregate (Input-Module #3 und # 4, siehe Tab. 1) in die INTEGER-Ausgabe-Worte IAUS(1:2,1:2,1) für die Schaltpegel "A" übertragen und nicht mit dem Istzustand übereinstimmende Befehls-FF's updaten (Subroutine VU1, 4.2.2.3);

* Bit RECHNER BEREIT (LED-Anzeige) setzen;

* IAUS() ausgeben;

* Bit RECHNER/HAND (Input-Modul # 2) prüfen, IHA=.TRUE. oder .FALSE. setzen und Betriebsart protokollieren. Bei Rechnerbetrieb die aktuellen Parameter in den Platten-File "VPA.DA" schreiben und die MAIN-Task VA1 (4.3) durch Swappen laden. Nach erfolgter Anfangsprüfung die (eventuell von VA1 veränderten) Parameter von Platte lesen. Das Prüfergebnis (INITIAL STATE PERMISSIBLE / PROHIBITED) wird mittels der INTEGER-Flagge MAFLA(3) übergeben; im Fall MAFLA(3)=0 wird die Sicherheitsabschaltung (4.2.4) eingeleitet, im Fall MAFLA(3)=1 wird das READY-Bit gesetzt und ausgegeben;

Nach diesen Vorbereitungen beginnt die Interrupt-Erkennung und Verzweigung. LJ2 bleibt auch während des Handbetriebs interrupt-fähig, um auf die Umschaltung HAND -> RECHNER reagieren zu können. Im Handbetrieb werden nach jedem Befehls-Interrupt (erkennbar am Interrupt-Code) die Ausgabeworte IAUS() aktualisiert, sodaß bei der Umschaltung HAND -> RECHNER die aktuellen Ausgabepegel zur Verfügung stehen.

* Interrupt-Betrieb bei RDOS anmelden (CALL CA190); Crate # 1 und # 2 disablen;

* LAM-Bits der Input-Module # 1 bis 5 sowie der CAMAC-Uhr löschen;

* Input-Module # 1 bis 5 sowie CAMAC-Uhr enablen;

* ITWAIT aufrufen, worauf sich LJ2 bis zum Eintreffen eines Interrupts selbst suspendiert;

* Nach dem Eintreffen eines Interrupts wird LJ2 automatisch wieder ready und, weil es die höchste Priorität besitzt, sofort executing; LJ2 kann anhand des Interrupt-Codes unterscheiden, ob es sich um

  a) einen Anlagen-Interrupt,
  b) einen Befehls-Interrupt,
  c) einen TT-Interrupt,
  d) einen Interrupt der CAMAC-Uhr oder
  e) einen illegalen Interrupt

  handelt. Die Reihenfolge der Abfragen ist wie angegeben; sie richtet sich weder nach der laufenden Nr. der CAMAC-Module noch nach der mutmaßlichen Häufigkeit der einzelnen Fälle, sondern nach deren Dringlichkeit. LJ2 startet daraufhin die entsprechende Service-Task.

  Der Sonderfall, daß das LAM nicht durch einen Befehl, sondern durch Betätigen des Schlüsselschalters RECHNER/HAND ausgelöst wurde, wird bereits in LJ2 abgefangen. Hierbei wird jedoch nicht der neue Zustand des Bits RECHNER/HAND mit dem alten Zustand verglichen, sondern unter Verwendung der LOGICAL-Flagge IHA nachfolgendem Schema verfahren:

IHA	LAM-Quelle Inp.Modul #	Neuer Zust. Bit RE/HA	Betr.-artänd.	Rechnereaktion
.TRUE.	1	irrelevant	HA->HA	IAUS() updaten
"	2	0	HA->HA	" "
"	2	1	HA->RE	IHA=.FALSE. Ausg. "COMPUTER CONTROL" Anfangsprüfung
.FALSE.	1	irrelevant	RE->RE	Befehlsprüfung
"	2	1	RE->RE	"
"	2	0	RE->HA	IHA=.TRUE. Ausg. "MANUAL CONTROL"

  Der andere Sonderfall, daß das VERRIEGELT-Signal quittiert wurde (ebenfalls CAMAC-Input-Modul # 2), erfordert keine Service-Task.

* Im Falle des TT-Interrupts 330 (Funktionstasten F1 bis F5) wird keine Service-Task gestartet, sondern nur die Flagge MAFLA(6) gesetzt (siehe V01, 4.2.4).

* Nach dem Starten der betreffenden Service-Task (im Fall e) direkt) springt LJ2 5 Programmschritte (Interrupt bei RDOS anmelden (CALL CA190)) zurück.

```
C
C       ***************************************************************
C       *           MAIN LJ2, MANGER                                  *
C       *           INTERRUPTERKENNUNG UND VERZWEIGUNG                *
C       *           FEBRUAR 1984                                      *
C       ***************************************************************
        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

        EXTERNAL TT1,UHR1,VM1,VB1,VO1

        LOGICAL LPT,STAT,IHA,SIAB
        DATA IAUS/8*0/,MIV/5*0/
        SIAB=.TRUE.

C       ========================================  PARAMETER VON PLATTE LESEN  ========

        CALL OVOPN(2,"LJ2.OL",IER)              ; OVERLAY FILE OEFFNEN
        IF(IER.EQ.1) GOTO 100
        TYPE"*** LJ2: OVERLAY FILE NOT OPENED<7>",IER
        STOP
100     CALL LO1(9,0,0,2)                       ; PARAMETER EINLESEN

C       ========================================  TASKS AKTIVIEREN  ==================

        CALL TTIT(11)                           ; TELETYPEHANDLER STARTEN
        CALL ITASK(VM1,1,10,IER)                ; VM1 AKTIVIEREN
        IF(IER.NE.1) CALL LO1(10,0,4,0)         ; FEHLERAUSGANG ZU EX1
        CALL HOLD(1,IER)                        ; SUSPENDIEREN
        IF(IER.NE.1) CALL LO1(10,0,10,0)        ; FEHLERAUSGANG ZU EX1
        CALL ITASK(VO1,2,20,IER)                ; DAS GLEICHE MIT VO1
        IF(IER.NE.1) CALL LO1(10,0,5,0)
        CALL HOLD(2,IER)
        IF(IER.NE.1) CALL LO1(10,0,10,0)
        CALL ITASK(VB1,3,30,IER)                ; DAS GLEICHE MIT VB1
        IF(IER.NE.1) CALL LO1(10,0,6,0)
        CALL HOLD(3,IER)
        IF(IER.NE.1) CALL LO1(10,0,10,0)
        CALL ITASK(TT1,4,40,IER)                ; DAS GLEICHE MIT TT1
        IF(IER.NE.1) CALL LO1(10,0,7,0)
        CALL HOLD(4,IER)
        IF(IER.NE.1) CALL LO1(10,0,10,0)
        CALL ITASK(UHR1,5,50,IER)               ; DAS GLEICHE MIT UHR1
        IF(IER.NE.1) CALL LO1(10,0,8,0)
        CALL HOLD(5,IER)
        IF(IER.NE.1) CALL LO1(10,0,10,0)
        CALL PRI(5)

C       ========================================  DATUM  ============================  =
        CALL DATE(IDT(1),IER)
        IF(LPT) WRITE(12,1200) (IDT(2),IDT(1),IDT(3),K=1,7)
1200    FORMAT(" ",7(I2,".",I2,".",I2,"   ")/72("="))

C       ========================================  MEMORY UPDATEN UND LAM-STATUS LOESCHEN  ============

        CALL LO1(7,0,12,0)                      ; PROTOKOLLAUSDRUCK
        DO 105 I=1,5                            ; MEMORY UPDATEN UND CLEAR
105     CALL VL1(I)                             ; LAM STATUS
        CALL LO1(8,0,0,0)                       ; BFF'S UPDATEN
        CALL ISET(IAUS(2,2,1),12)               ; BIT "RECHNER BEREIT" SETZEN
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)   ; AUSGABE RECHNER BEREIT
        IF(.NOT.ITEST(INEU(1,2),0)) GOTO 120    ; RECHNER / HANDBETRIEB ?
```

```
C       ================  RECHNERBETRIEB/HANDBETRIEB ============================

        CALL LO1(7,0,1,0)                                   ; PROTOKOLL RECHNERBETRIEB
        IHA=.FALSE.
        CALL LO1(9,0,0,1)                                   ; DATEN SCHREIBEN FUER SWAP
        CALL FSWAP("VA1.SV")                                ; SWAP ANFANGSPRUEFUNG
        CALL LO1(9,0,0,2)                                   ; DATEN WIEDER LESEN
        IF(MAFLA(3).EQ.0) CALL RELSE(2,IER)                 ; SIAB BEI UNZULAESSIG
        IF(IER.NE.1) CALL LO1(10,0,5,0)
        MAFLA(3)=1
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)               ; READY-BIT AUSGEBEN
        GOTO 130

C       ========================================================================

120     CALL LO1(7,0,2,0)                                   ; PROTOKOLL HANDBETRIEB
        IHA=.TRUE.

C       ==========  HAT SICH INZWISCHEN WAS GETAN  =============================

130     DO 160 I=3,5                                        ; ZUSTANDSAENDERUNG M ODER F
        CALL CAL93(27,A3473(1,I),DUMMY,IQ)                  ; SEIT LETZTEM UPDATEN DER
        IF(IQ.EQ.0) GOTO 160                                ; MEMORY - REGISTER ??
        CALL VL1(I)
        CALL RELSE(1,IER)                                   ; WENN JA,VM STARTEN
        IF(IER.NE.1) CALL LO1(10,0,13,0)
        GOTO 170
160     CONTINUE

C       ============  IAUS IMMER SCHOEN AKTUELL HALTEN  ========================

165     IAUS(2,1,1)=INEU(2,3)                               ; IM HANDBETRIEB MUSS NACH
        IAUS(2,2,1)=INEU(2,4)                               ; JEDEM BEFEHL DAS AUSGABE-
        IAUS(1,1,1)=INEU(1,3)                               ; WORT UPGEDATED WERDEN
        CALL ISET(IAUS(2,2,1),12)
        CALL ISET(IAUS(2,2,1),13)
        AUS(1,1)=FLOINT(IAUS(1,1,1))
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(1,1),AUS(1,1),IQ)
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)

C       ================== ACHTUNG IN DIE STARTLOECHER ==========================

170     CALL CA190                                          ; INTERRUPTS BEI RDOS ANMELDEN
        CALL CAL93(26,ACR(1),DUMMY,IQ)                      ; BRANCH DEMAND ENABLEN
        CALL CAL93(26,ACR(2),DUMMY,IQ)                      ;    "       "      "
        DO 175 I=1,5
        CALL CAL93(26,A3473(1,I),DUMMY,IQ)                  ; ENABLE MODUL - LAMS
175     CALL CAL93(10,A3473(1,I),DUMMY,IQ)                  ; CLEAR  MODUL - LAMS
        CALL CAL93(10,AUHR,DUMMY,IQ)                        ; CLEAR  LAM UHR
        CALL CAL93(26,AUHR,DUMMY,IQ)                        ; ENABLE LAM UHR
        CALL ITWAIT(IC)                                     ; AUF INTERRUPT WARTEN

C       ====================  JETZT GEHT DIE POST AB  ==========================

        DO 180 I=1,5                                        ; ZUSTANDSAENDERUNG ODER
        I1=6-I                                              ;       BEFEHL ?
        IF(IC.EQ.(900+N3473(I1))) GOTO 190                  ; WENN JA ZU 190
180     CONTINUE
        GOTO 220                                            ; ANSONSTEN TT1 STARTEN

C       ========================================================================
```

```
C
190     CALL VL1(I1)                                    ; GEAENDERTES REGISTER LESEN
        IF(IHA.AND.IC.NE.(900+N3473(2))) GOTO 165 ; WEITERHIN HAND?
        GOTO (200,205,210,210,210),I1                   ; SONST VERZWEIGUNG ZU :
C       ================================================================================

200     CALL RELSE(3,IER)                               ; BEFEHLSPRUEFUNG
        IF(IER.NE.1) CALL LO1(10,0,14,0)
        GOTO 170
C       ================================================================================

205     IF(.NOT.ITEST(INEU(1,2),0)) GOTO 206    ; HANDBETRIEB NEU,DANN ZU 206
        IF(.NOT.IHA) GOTO 200                           ; WENN VORHER HAND WAR,IST JETZT
        IHA=.FALSE.                                     ; RECHNERBETRIEB
        CALL LO1(7,0,1,0)                               ; PROTOKOLLIEREN
        CALL LO1(9,0,0,1)                               ; PARAMETER SCHREIBEN
        CALL CA199                                      ; INTERRUPTS SPERREN
        CALL FSWAP("VA1.SV")                            ; ANFANGSPRUEFUNG
        CALL LO1(9,0,0,2)                               ; PARAMETER LESEN
        IF(MAFLA(3).EQ.0) CALL RELSE(2,IER)     ; SIAB, WENN UNZULAESSIG
        IF(IER.NE.1) CALL LO1(10,0,5,0)
        MAFLA(3)=1
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)   ; READY-BIT AUSGEBEN
        GOTO 170
C       ================================================================================

206     IF(IHA) GOTO 170
        CALL LO1(7,0,2,0)                               ; HANDBETRIEB NEU
        IHA=.TRUE.
        CALL ICLR(IAUS(2,2,1),13)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)   ; READY-BIT CLEAREN
        GOTO 170                                        ; INTERRUPTFAEHIG BLEIBEN
C       ================================================================================

210     CALL RELSE(1,IER)                               ; MELDUNGEN CHECKEN
        IF(IER.NE.1) CALL LO1(10,0,13,0)
        GOTO 170
C       ================================================================================

220     IF(IC.NE.327) GOTO 230                  ; ESCAPE - INTERRUPT ?
        CALL RELSE(4,IER)                               ; TT1 STARTEN
        IF(IER.NE.1) CALL LO1(10,0,15,0)
        GOTO 170
C       ================== MINUTENINTERRUPT DER CAMAC-UHR ==============================

230     IF(IC.NE.(900+NUHR)) GOTO 240           ; MINUTEN - INTERRUPT ?
        CALL RELSE(5,IER)                               ; READY UHR1
        IF(IER.NE.1) CALL LO1(10,0,16,0)
        CALL CAL93(10,AUHR,DUMMY,IQ)                   ; CLEAR LAM UHR
        GOTO 170
C       ================================================================================

240     IF(IC.EQ.330) GOTO 241                  ; F1-F5 INTERRUPT
        TYPE *                          ILLEGAL INTERRUPT*
        GOTO 170
241     MAFLA(6)=1
        GOTO 170

        END
```

4.2.1 Task VB1 (Service-Task für Befehls-Interrupts)

VB1 identifiziert einen über die 3473-Input-Module # 1 oder # 2 durch Tastenbetätigung auf der aktiven Bedientafel gegebenen Befehl zum Schalten von Ventilen oder Pumpen. VB1 wird von LJ2 mittels

CALL RELSE(3,IER)

gestartet und übernimmt über die COMMON-Variablen IC, INEU() und IALT() den Interrupt-Code sowie die bereits in die 16-Bit-INTEGER-Worte konvertierten Bitmuster nach und vor Eintreffen des Interrupts. VB1 gliedert sich in folgende Programmschritte:

* Wenn der Befehls-FF, der das LAM verursacht hat, nicht durch Drücken der betreffenden Befehls-Taste, sondern durch Ausgabe des Reset-Signals umgeschaltet wurde (dann ist die Flagge MAFLA(4) = 1), wird MAFLA(4) wieder genullt, und VB1 springt nach einer Verzögerung von 0.2 sec (d.i. die doppelte Dauer des RESET-Signals) direkt zum letzten Programmschritt.

* Als nächstes findet eine Vorentscheidung statt darüber, ob das Bit VERRIEGELT gleich 1 oder 0 ist. Im ersten Fall wird dies protokolliert mit

  LOCKED

  Der betreffende Befehls-Flipflop wird auf den ursprünglichen Zustand zurückgesetzt und VB1 wird suspendiert.

* Durch bitweisen Vergleich zwischen INEU() und IALT() wird festgestellt, welches Bit (Nr. 0 bis 15) sich in welchem Wort (Indizes (2,1), (1,1) oder (2,2)) geändert hat. Je nach dem betroffenen Wort wird eine der 3 Overlay-Subroutinen VBS1 / VBS2 / VBS3 (4.2.1.1 bis 4.2.1.3) aufgerufen. Die Bit-Nr. wird als Argument I übergeben. Bei Änderung des VERRIEGELT-Bits auf 0 (durch Quittieren) wird lediglich die (vorher gesetzte) Software-Sperre wieder aufgehoben.

* CALL SUSP und, sobald VB1 wieder executing wird, Rücksprung an den Anfang von VB1.

```
C
C       ********************************************************************
C       **  TASK     VB1                                                   **
C       **  TASK ZUR UEBERPRUEFUNG UND AUSFUEHRUNG DER MANUELLEN           **
C       **  BEFEHLE.          MANGER FEBRUAR 1984                          **
C       ********************************************************************

        TASK VB1

        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2) ,N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LFT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
        LOGICAL LFT,STAT,IHA,SIAB

C       ================== RESET-SIGNAL ODER BEFEHL? ==========================

1       IF(MAFLA(4).NE.1.AND.MAFLA(5).NE.1) GOTO 2 ; RESET ODER SIAB?
        CALL FDELY(2*IFDEL)                        ; 0.2 SEC=2*RESET-DAUER
        MAFLA(4)=0
        GOTO 50

C       ========== SOFTWARESPERRE WENN VERRIEGELT NICHT QUITTIERT ============

2       IF (.NOT.ITEST(INEU(1,2),1)) GOTO 3        ; VERRIEGELT QUITTIERT?
        TYPE*                      LOCKED*
        CALL LO1(8,0,8,0)                          ; WENN NEIN, RESET BEFEHL
        GOTO 50
3       IF(IC.EQ.(900+N3473(2))) GOTO 10

C       ================ BEFEHLSIDENTIFIKATION V1 - V16 =======================

        DO 5 I=1,16
        IF(ITEST(INEU(2,1),I-1).NE.ITEST(IALT(2,1),I-1)) GOTO 6
5       CONTINUE
        GOTO 20
6       CALL LO1(2,0,0,I)                          ; VBS1 LADEN
        GOTO 50

C       ================ BEFEHLSIDENTIFIKATION PUMPEN =========================

10      DO 15 I=1,12
        IF(ITEST(INEU(2,2),I-1).NE.ITEST(IALT(2,2),I-1)) GOTO 16
15      CONTINUE
        GOTO 50
16      CALL LO1(4,0,0,I)                          ; VBS3 LADEN
        GOTO 50

C       ================ BEFEHLSIDENTIFIKATION V17 - V43 ======================

20      DO 25 I=1,7
        IF(ITEST(INEU(1,1),I-1).NE.ITEST(IALT(1,1),I-1)) GOTO 26
25      CONTINUE
        GOTO 50
26      CALL LO1(3,0,0,I)                          ; VBS2 LADEN

50      CALL SUSP
        GOTO 1
        END
```

Tab. 2: Bedingungen für die Zulässigkeit von Befehlen und zugehörige Rechnerreaktionen

	V1 auf	V2 auf	V3 auf
	V14 auf V6 zu	V14 auf	
	V42 " V7 "	V16 "	
		V18 "	
Reakt.			

	V4 auf	V4 zu	V5 auf	V6 auf	V6 zu	V7 auf	V8 auf	V8 zu
	R21 ein	-	V4 zu	R21 ein	-	V6 zu	R21 ein	-
	V5 zu			V4 zu		V2 "	V4 zu	
	V6 "			V8 "			V6 "	
	V8 "			V7 "			F4 gut	
	F4 gut			V2 "				
				F4 gut				
Reakt.		R21 aus			R21 aus			R21 aus

	V9 auf	V10 auf	V10 zu	V11 auf	V11 zu	V12 auf
	V1 zu	V1 zu	-	V3 zu	-	V3 zu
	V2 "	V2 "		V10 "		V11 "
	V3 "	V3 "		V12 "		V18 "
	V10 "	V9 "		V18 "		
	V11 "	V11 "		R22 ein		
		V14 "		F5 gut		
		R22 ein				
		F5 gut				
Reakt.			R22 aus		R22 aus	

	V13 auf	V13 zu	V14 auf	V15 auf	V15 zu	V16 auf
	V14 zu	D24 ¬bb	V1 z|a|a|a|a	V16 zu	D26 ¬bb	V3 z|a|a|a|a
	R23 ein		V2 |z|a|z|a	R25 ein		V2 |z|a|z|a
	F7 gut		V3 |z|z|a|a	F9 gut		V1 |z|z|a|a
			V41 |z|z|z|z			V41 | | |z|z
			V6 | |z| |z			V6 | |z| |z
			V11 | | |z|z			V10 |z|z|z|z
			V10 zu			V11 zu
			D24 bb			D26 bb
			(F2 gut)			(F6 gut)
			F8 "			F10 "
Reakt.		R23 aus			R25 aus	

	V17 auf	V17 zu	V18 auf	V19 auf	V19 zu	V20 auf
	V18 zu	D28 ¬bb	V3 z|a|a|a|a	V20 zu	D30 ¬bb	V3 z|a|a|a|a
	R27 ein		V2 |z|a|z|a	R29 ein		V2 |z|a|z|a
	F11 gut		V1 |z|z|a|a	F13 gut		V1 |z|z|a|a
			V41 | | |z|z			V41 | | |z|z
			V6 | |z| |z			V6 | |z| |z
			V10 |z|z|z|z			V10 |z|z|z|z
			V11 zu			V11 zu
			D28 bb			D30 bb
			(F6 gut)			(F6 gut)
			F12 "			F14 "
Reakt.		R27 aus			R29 aus	

	V41 auf	V41 zu	V42 auf	V43 auf	V43 zu
	R44 ein		V1 z|a|a|a|a	V42 zu	D45 aus
	V43 zu		V2 |z|a|z|a	R44 ein	
			V3 |z|z|a|a	V41 zu	
			V6 | |z| |z		
			V10 |z|z|z|z		
			V11 | | |z|z		
			V41 zu		
			D45 bb		
			F1 gut		
			Fkr "		
Reakt.		R44 aus			R44 aus

	R21 ein	R21 aus	R22 ein	R22 aus	R44 ein	R44 aus	D45 ein	D45 aus
	-	V6 zu	-	V10 zu	-	V41 zu	V42 zu	V42 zu
		V8 "		V11 "		V43 "	R44 ein	
							V43 auf	
							Fkr gut	
Reakt.								

	R23 ein	R23 aus	D24 ein	D24 aus	R25 ein	R25 aus	D26 ein	D26 aus
	-	V13 zu	R23 ein	V14 zu	-	V15 zu	R25 ein	V16 zu
			V13 auf				V15 auf	
			F7 gut				F10 gut	
			F8 "				F9 "	
			V14 zu				V16 zu	
Reakt.								

	R27 ein	R27 aus	D28 ein	D28 aus	R29 ein	R29 aus	D30 ein	D30 aus
	-	V17 zu	R27 ein	V18 zu	-	V19 zu	R29 ein	V20 zu
			V17 auf				V19 auf	
			F12 gut				F14 gut	
			F11 "				F13 "	
			V18 zu				V20 zu	
Reakt.								

4.2.1.1 VBS1 (Overlay-Subroutine zur Befehlsprüfung V1 bis V16)

VBS1 prüft den von VB1 identifizierten Befehl aus der Befehlsgruppe V1 bis V16 auf Zulässigkeit gemäß der Befehlsprüfliste (Tab. 2). Bei Zulässigkeit gibt VBS1 den Schaltpegel "A" über den CAMAC-Output-Modul # 1 an das Leistungsteil Ü7, bei Unzulässigkeit hingegen ein impulsförmiges RESET-Signal (Dauer 0.1 sec) über den CAMAC-Output-Modul # 2 an den betreffenden Befehls-FF im Steuergerät Ü6 sowie das VERRIEGELT-Signal aus. Die Befehle werden nach folgendem Beispiel protokolliert:

hh:mm:ss B V 9: 0 oder:

hh:mm:ss B V 9: 1 PROHIBITED

VBS1 liegt im Overlay-Segment # 0 des Platten-Files "LJ2.OL", hat den Overlay-Namen OVB1, die Identifikations-Nr. ION=2 (siehe 4.2.7), besitzt ein explizites Argument (I) und wird ausschließlich von VB1 aufgerufen mittels

CALL LO1(2,0,0,I)

siehe 4.2.7. (I-1) ist die Nr. des zu prüfenden Befehls-Bits. Benötigt wird die aktuelle COMMON-Variable INEU() mit den Zustandsmeldungen der CAMAC-Input-Module # 3 bis # 5.

VBS1 gliedert sich in folgende Schritte:

* Initialisierung der Befehlsprüflisten und der Ausblendmasken für nichtrelevante Zustandsmeldungen in den Feldern ILIS(5,16) und IMAS(5,16), IL14(5,5) und IM14(5,5) sowie IL16(5,5) und IM16(5,5). Die Feldelemente entsprechen in ihrem Aufbau den die Bitmuster haltenden INTEGER-Worten der aktuellen Istzustandsmeldungen INEU(). Für Befehle, für die nur ein einziger Istzustand zulässig ist, werden die jeweiligen Prüflisten sowie die Ausblendmasken für nichtrelevante Zustandsmeldungen in ILIS(5,16) und IMAS(5,16) initialisiert. Für Befehle, für die mehrere zulässige Zustandskombinationen existieren, wurden gesonderte Prüflisten erstellt (hier IL14(5,5) und IM14(5,5) für V14 sowie IL16(5,5) und IM16(5,5) für V16).

* Vorentscheidung, ob das betreffende Ventil geschlossen werden soll oder ob es sich um Ventil V14 bzw. V16 handelt. Im ersteren Fall wird sofort zum drittletzten Programmschritt gesprungen; in den beiden letzteren Fällen wird ein Programmschritt übersprungen und B V14 bzw. B V16 wird gesondert überprüft.

* Die Überprüfung eines Befehls vom Typ "Ventil auf", für den gemäß Tab. 2 nur eine einzige erlaubte Zustandskombination existiert, erfolgt in 2 geschachtelten DO-Schleifen. Die äußere läuft von L=1,5 für die die Bitmuster haltenden INTEGER-Worte der 3473-Input-

Module sowie die Vergleichsworte ILIS() und IMAS(). Die innere läuft von I=1,16 für den Einzelbit-Test. Das Argument von VBS1 wird direkt zur Auswahl der für den Befehl zuständigen 5 Worte verwendet. Bits, die für die Prüfung nicht relevant sind, werden durch die Maske IMAS() ausgeblendet, indem der Vergleich übersprungen wird. Bei Identität der Vergleichs-Bits wird zum drittletzten Programmabschnitt (Ausgabe von "A") gesprungen, im anderen Fall zum letzten Programmabschnitt (Ausgabe von "R").

* Die Prüfungen von B V14 bzw. B V16, für die, gemäß Tab. 2, 5 erlaubte Zustandskombinationen existieren, erfolgen in 3-fach geschachtelten DO-Schleifen. Die zusätzlichen äußeren Schleifen durchlaufen die verschiedenen zulässigen Kombinationen. Die Aufgaben der mittleren und der inneren DO-Schleifen entsprechen der im vorigen Abschnitt beschriebenen Prüfung.

* Setzen des dem (zulässigen) Befehl "AUF" entsprechenden Bits im Ausgabewort IAUS(J,K,1) und Ausgabe. Unmittelbar vor der Ausgabe wird in IDEL(J,K) das dem Befehl entsprechende Bit = 1 gesetzt, wodurch VM1 (4.2.3) angezeigt wird, daß ein Zeitintervall von 1 sec abläuft, während dessen VB1 und damit auch VBS1 suspendiert sind (CALL FDELY). Liegt nach Ablauf des Zeitintervalls die dem Befehl kongruente Rückmeldung M vor, wird das Bit in IDEL(J,K) wieder gecleart und VBS1 beendet. Bei fehlender Rückmeldung wird die Protokollzeile

  NOT DONE! CHECK THE VALVE / PUMP

  ausgegeben, und das entsprechende Befehls-Flipflop wird zurückgesetzt.

* Nullen des dem (stets zulässigen) Befehl "ZU" entsprechenden Bits im Ausgabewort IAUS(J,K,1) und Ausgabe. Wie vorstehend, jedoch erfolgt gegebenenfalls entsprechend Tab. 2 zusätzlich Mitabschaltung nicht mehr benötigter Rotationspumpen (RECHNERAKTION) durch Ausgabe des entsprechenden RESET-Signals über den Output-Modul # 2.

* Resetten des (als unzulässig befundenen) Befehls (wird über die Overlay-Subroutine VU1 (4.2.2.3) getätigt) und Ausgabe des VERRIEGELT-Signals sowie der Protokollzeile z.B.

  B V 9: 1 PROHIBITED

* RETURN

```
C
C     ****************************************************************************
C     **  OVERLAY VBS1                MANGER FEBRUAR 1984                       **
C     **  SUBR ZUR UEBERPRUEFUNG UND AUSFUEHRUNG DER MANUELLEN                  **
C     **  BEFEHLE.                                (VENTILE 1 BIS 16)            **
C     ****************************************************************************
      COMPILER NOSTACK
      OVERLAY OVB1
      SUBROUTINE VBS1(IS)

      COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LFT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

      LOGICAL LFT,STAT,IHA,SIAB
      INTEGER ILIS(5,16),IMAS(5,16),IL14(5,5),IM14(5,5),IL16(5,5),IM16(5,5),
     *  IN(5)

      DATA ILIS/40K,20000K,8*0K,2K,120000K,3*0K,3*0K,40K,10K,5*0K,3*0K,
     *          40K,10K,8*0K,40K,10K,8*0K,100K,20K,0K,0K,0K,100K,20K,
     *          8*0K,200K,100K,5*0K,3*0K,400K,400K,5*0K/

      DATA IMAS/40K,20000K,4*0K,140K,3*0K,2K,120000K,3*0K,0K,260K,0K,40K,
     *          10K,0K,10K,3*0K,0K,302K,0K,40K,10K,
     *          0K,42K,4*0K,40K,0K,40K,10K,0K,23007K,4*0K,22407K,0K,100K,20K,
     *          2K,5004K,0K,100K,20K,2K,2004K,4*0K,20000K,0K,200K,100K,5*0K,
     *          0K,100000K,0K,400K,400K,5*0K/

      DATA IL14/0K,0K,1K,0K,202K,0K,1K,1K,0K,202K,0K,3K,1K,0K,202K,0K,5K,
     *          1K,0K,202K,0K,7K,1K,0K,202K/

      DATA IM14/0K,1001K,1K,0K,202K,20K,1007K,1K,0K,202K,20K,1047K,1K,0K,
     *          202K,20K,3007K,1K,0K,202K,20K,3047K,1K,0K,202K/

      DATA IL16/0K,0K,2K,0K,1040K,0K,4K,2K,0K,1040K,0K,6K,2K,0K,1040K,0K,5K,
     *          2K,0K,1040K,0K,7K,2K,0K,1040K/

      DATA IM16/0K,2004K,2K,0K,1040K,0K,3007K,2K,0K,1040K,0K,3047K,2K,0K,
     *          1040K,20K,3007K,2K,0K,1040K,20K,3047K,2K,0K,1040K/

      IN(1)=INEU(1,3)
      IN(2)=INEU(2,3)
      IN(3)=INEU(1,4)
      IN(4)=INEU(2,4)
      IN(5)=INEU(2,5)

C     ========      BEFEHLSPRUEFUNG, VERZWEIGUNG AUF/ZU ====================

      IF(.NOT.ITEST(INEU(2,1),IS-1)) GOTO 550        ; VENTIL ZU ?
      IF(IS.EQ.14) GOTO 30                           ; V14 GEHT GETRENNT
      IF(IS.EQ.16) GOTO 70                           ; V16 GEHT GETRENNT
C     ============================================================================

      DO 20 L=1,5                                    ; L=5 REGISTER
      DO 10 I=1,16                                   ; I=16 BITS
      IF(.NOT.ITEST(IMAS(L,IS),I-1)) GOTO 10         ; UNERWUENSCHTE AUSBLENDEN
      IF(ITEST(ILIS(L,IS),I-1).NE.ITEST(IN(L),I-1)) GOTO 600
10    CONTINUE
20    CONTINUE
      GOTO 500                                       ; ZULAESSIG!
```

```
C       ========================= V 14 ==========================================

30      DO 60 L=1,5                                          ; 5 MOEGLICHKEITEN
        DO 50 M=1,5                                          ; 5 REGISTER
        DO 40 I=1,16                                         ; 16 BITS
        IF(.NOT.ITEST(IM14(M,L),I-1)) GOTO 40
        IF(ITEST(IL14(M,L),I-1).NE.ITEST(IN(M),I-1)) GOTO 60
40      CONTINUE
50      CONTINUE
        GOTO 500                                             ; ZULAESSIG!
60      CONTINUE
        GOTO 600                                             ; UNZULAESSIG!
C       ========================= V 16 ==========================================

70      DO 100 L=1,5                                         ; 5 MOEGLICHKEITEN
        DO 90 M=1,5                                          ; 5 REGISTER
        DO 80 I=1,16                                         ; 16 BITS
        IF(.NOT.ITEST(IM16(M,L),I-1)) GOTO 80
        IF(ITEST(IL16(M,L),I-1).NE.ITEST(IN(M),I-1)) GOTO 100
80      CONTINUE
90      CONTINUE
        GOTO 500                                             ; ZULAESSIG!
100     CONTINUE
        GOTO 600                                             ; UNZULAESSIG!

C       ================  BEFEHL VENTIL AUF AUSGEBEN ==================================

500     CALL LO1(7,41,8,IS)
        CALL ISET(IAUS(2,1,1),IS-1)                          ; IN IAUS UEBERTRAGEN
        CALL ISET(IDEL(2,1),IS-1)                            ; MESSAGE AN VM1
        AUS(1,1)=FLOINT(IAUS(1,1,1))
        CALL CAL93(16,A3072(1,1),AUS(1,1),IQ)
        CALL FDELY(20*IFDEL)
        IF(IDEL(2,1).NE.0) GOTO 590                          ; AUSGEFUEHRT ?
        GOTO 900
C       ============  BEI VENTIL ZU ROTATIONSPUMPE MIT AUS  ========================

550     IF(IS.EQ.4.OR.IS.EQ.6.OR.IS.EQ.8) GOTO 551
        IF(IS.EQ.10.OR.IS.EQ.11) GOTO 552
        IF(IS.EQ.13) GOTO 553
        IF(IS.EQ.15) GOTO 554
        GOTO 560
C       ============  EINZELPRUEFUNG ZU  ============================================

551     CALL ICLR(IAUS(2,2,1),5)                             ; R21 AUS
        CALL ISET(IDEL(2,2),5)
        GOTO 555
552     CALL ICLR(IAUS(2,2,1),6)                             ; R22 AUS
        CALL ISET(IDEL(2,2),6)
        GOTO 555
553     IF(ITEST(IN(3),0)) GOTO 610
        CALL ICLR(IAUS(2,2,1),7)                             ; R23 AUS
        CALL ISET(IDEL(2,2),7)
        GOTO 555
554     IF(ITEST(IN(3),1)) GOTO 610
        CALL ICLR(IAUS(2,2,1),8)                             ; R25 AUS
        CALL ISET(IDEL(2,2),8)
555     MAFLA(4)=1                                           ; FLAGGE RESET
```

```
C
C       ==========  BEFEHL VENTIL ZU AUSGEBEN UND GEGEBENENFALLS R AUS ==========

560     CALL LO1(7,41,9,IS)
        CALL ICLR(IAUS(2,1,1),IS-1)
        CALL ISET(IDEL(2,1),IS-1)
        AUS(1,1)=FLOINT(IAUS(1,1,1))
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(1,1),AUS(1,1),IQ)
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        CALL FDELY(20*IFDEL)
        IF(IDEL(2,1).NE.0.AND.IDEL(2,2).NE.0) GOTO 590
        CALL LO1(8,0,0,0)
        GOTO 900
C       ===== RESET BEFEHL ==========================================================
590     CALL LO1(7,0,13,0)                              ; NOT DONE PROTOKOLLIEREN
        IDEL(2,1)=0
        IDEL(2,2)=0
        GOTO 620
600     CALL LO1(7,41,10,IS)                            ; PROHIBITTED PROTOKOLLIEREN
        GOTO 620
610     CALL LO1(7,41,11,IS)                            ;       "           "
620     CALL ISET(IAUS(2,2,1),14)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        CALL ICLR(IAUS(2,2,1),14)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        CALL LO1(8,0,0,0)
900     RETURN

        END
```

4.2.1.2 Subroutine VBS2 (Overlay-Subroutine zur Befehlsprüfung der Ventile V17 bis V43)

Wie VBS1 mit folgenden Abweichungen:

* Geprüft wird die Befehlsgruppe B V17 bis B V43, siehe Tab. 1.

* VBS2 hat den Overlay-Namen OVB2, die Identifikations-Nr. ION=3 (siehe 4.2.7) und wird aufgerufen:

  CALL LO1(3,0,0,I)

* Gesondert überprüft werden "B V18 AUF", "B V20 AUF" und "B V42 AUF" (3-fache DO-Schleifen; Prüftabellen IL18(), IM18(); IL20(), IM20(); IL42(), IM42())

4.2.1.3 Subroutine VBS3 (Overlay-Subroutine zur Befehlsprüfung von R21 bis R44 und D24 bis D45)

Wie VBS1 mit folgenden Abweichungen:

* Geprüft wird die Befehlsgruppe B R21 bis B R44 und B D24 bis B D45, siehe Tab. 1.

* VBS3 hat den Overlay-Namen OVB3, die Identifikations-Nr. ION=4 (siehe 4.2.7) und wird aufgerufen:

  CALL LO1(4,0,0,I)

* Die Rotationspumpen einerseits und die Diffusionspumpen andererseits werden jeweils in 2-fach DO-Schleifen überprüft.

```
C
C     ***************************************************************************
C     **  OVERLAY VBS2              MANGER FEBRUAR 1984                          *
C     **  SUBR ZUR UEBERPRUEFUNG UND AUSFUEHRUNG DER MANUELLEN                   *
C     **  BEFEHLE.                                (VENTILE 17  BIS 43 )          *
C     ***************************************************************************

      COMPILER NOSTACK
      OVERLAY OVB2
      SUBROUTINE VBS2(IS)

      COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

      LOGICAL LPT,STAT,IHA,SIAB
      INTEGER ILIS(5,7),IMAS(5,7),IN(5),IM18(5,5),IL18(5,5),IM42(5,5),
     *  IL42(5,5),IL20(5,5),IM20(5,5)

      DATA ILIS/3*0K,1000K,2000K,8*0K,2000K,10000K,8*0K,4000K,9*0K,4000K,0K/
      DATA IMAS/2K,0K,0K,1000K,2000K,5*0K,10K,2*0K,2000K,10000,5*0K,100K,0K,
     *          0K,4000K,6*0K,60K,2*0K,4000K,0K/
      DATA IL18/0K,0K,4K,0K,4040K,0K,4K,4K,0K,4040K,0K,6K,4K,0K,4040K,0K,
     *          5K,4K,0K,4040K,0K,7K,4K,0K,4040K/
      DATA IM18/0K,2004K,4K,0K,4040K,0K,3007K,4K,0K,4040K,0K,3047K,4K,0K,
     *          4040K,20K,3007K,4K,0K,4040K,20K,3047K,4K,0K,4040K/
      DATA IL20/0K,0K,10K,0K,20040K,0K,4K,10K,0K,20040K,0K,6K,10K,0K,
     *          20040K,0K,5K,10K,0K,20040K,0K,7K,10K,0K,20040K/
      DATA IM20/0K,2004K,10K,0K,20040K,0K,3007K,10K,0K,20040K,0K,3047K,10K,
     *          0K,20040K,20K,3007K,10K,0K,20040K,20K,3047K,10K,0K,20040K/
      DATA IL42/0K,0K,20K,0K,40001K,20K,1K,20K,0K,40001K,0K,3K,20K,0K,
     *          40001K,0K,5K,20K,0K,40001K,0K,7K,20K,0K,40001K/
      DATA IM42/20K,1K,20K,0K,40001K,20K,1007K,20K,0K,40001K,20K,1047K,20K,
     *          0K,40001K,20K,3007K,20K,0K,40001K,20K,3047K,20K,0K,40001K/

      IP=IS+16                                  ; FUER AUFRUF VON VP1
      IN(1)=INEU(1,3)
      IN(2)=INEU(2,3)
      IN(3)=INEU(1,4)
      IN(4)=INEU(2,4)
      IN(5)=INEU(2,5)

C     ================== VERZWEIGUNG ZUR EINZELPRUEFUNG ======================

      IF(.NOT.ITEST(INEU(1,1),IS-1)) GOTO (510,550,550,550,520,550,530),IS

      IF(IS.EQ.2) GOTO 30                       ; V18 WIRD GETRENNT GEPRUEFT
      IF(IS.EQ.4) GOTO 110                      ; V20 EBENFALLS
      IF(IS.EQ.6) GOTO 70                       ; V42 EBENFALLS

C     ===========================================================================

      DO 20 L=1,5                               ; 5 REGISTER
      DO 10 I=1,16                              ; 16 BITS
      IF(.NOT.ITEST(IMAS(L,IS),I-1)) GOTO 10
      IF(ITEST(ILIS(L,IS),I-1).NE.ITEST(IN(L),I-1)) GOTO 600
10    CONTINUE
20    CONTINUE
      GOTO 500
```

```
C       ================== VENTIL 18 ==========================================
30      DO 60 L=1,5                                 ; 5 MOEGLICHKEITEN
        DO 50 M=1,5                                 ; 5 REGISTER
        DO 40 I=1,16                                ; 16 BITS
        IF(.NOT.ITEST(IM18(M,L),I-1)) GOTO 40
        IF(ITEST(IL18(M,L),I-1).NE.ITEST(IN(M),I-1)) GOTO 60
40      CONTINUE
50      CONTINUE
        GOTO 500                                    ; ZULAESSIG!
60      CONTINUE
        GOTO 600                                    ; UNZULAESSIG

C       ================== VENTIL 42 ==========================================
70      DO 100 L=1,5                                ; WIE OBEN
        DO 90 M=1,5
        DO 80 I=1,16
        IF(.NOT.ITEST(IM42(M,L),I-1)) GOTO 80
        IF(ITEST(IL42(M,L),I-1).NE.ITEST(IN(M),I-1)) GOTO 100
80      CONTINUE
90      CONTINUE
        GOTO 500
100     CONTINUE
        GOTO 600

C       ================== VENTIL 20 ==========================================
110     DO 140 L=1,5                                ; WIE OBEN
        DO 130 M=1,5
        DO 120 I=1,16
        IF(.NOT.ITEST(IM20(M,L),I-1)) GOTO 120
        IF(ITEST(IL20(M,L),I-1).NE.ITEST(IN(M),I-1)) GOTO 140
120     CONTINUE
130     CONTINUE
        GOTO 500
140     CONTINUE
        GOTO 600

C       ==========    BEFEHL AUF AUSGEBEN   ==================================
500     CALL LO1(7,41,8,IF)
        CALL ISET(IAUS(1,1,1),IS-1)
        CALL ISET(IDEL(1,1),IS-1)
        AUS(1,1)=FLOINT(IAUS(1,1,1))
        CALL CAL93(16,A3072(1,1),AUS(1,1),IQ)
        CALL FDELY(20*IFDEL)
        IF(IDEL(1,1).NE.0) GOTO 590
        GOTO 900

C       ==========  BEFEHL ZU PRUEFEN ==========================================
510     IF(ITEST(IN(3),2)) GOTO 610                 ; D28 BETRIEBSBEREIT?
        CALL ICLR(IAUS(2,2,1),9)                    ; R27 AUSSCHALTEN
        CALL ISET(IDEL(2,2),9)
        GOTO 531
520     CALL ICLR(IAUS(2,2,1),11)                   ; R44 AUSSCHALTEN
        CALL ISET(IDEL(2,2),11)
        GOTO 531
530     IF(ITEST(IN(4),4)) GOTO 610                 ; D45 AUS?
        CALL ICLR(IAUS(2,2,1),11)                   ; R44 AUSSCHALTEN
        CALL ISET(IDEL(2,2),11)
531     MAFLA(4)=1                                  ;FLAGGE FUER RESET
```

```
C
C       ==========  BEFEHL ZU AUSGEBEN ==========================================
550     CALL LO1(7,41,9,IF)
        CALL ICLR(IAUS(1,1,1),IS-1)
        CALL ISET(IDEL(1,1),IS-1)
        AUS(1,1)=FLOINT(IAUS(1,1,1))
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(1,1),AUS(1,1),IQ)
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        CALL FDELY(20*IFDEL)
        IF(IDEL(1,1).NE.0.AND.IDEL(2,2).NE.0) GOTO 590
        CALL LO1(8,0,0,0)
        GOTO 900
C       =========   RESET BEFEHL ================================================
590     CALL LO1(7,0,13,0)                          ; NOT DONE PROTOKOLLIEREN
        IDEL(1,1)=0
        IDEL(2,2)=0
        GOTO 620
600     CALL LO1(7,41,10,IF)                        ; PROHIBITED PROTOKOLLIEREN
        GOTO 620
610     CALL LO1(7,41,11,IF)                        ;       "           "
620     CALL ISET(IAUS(2,2,1),14)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        CALL ICLR(IAUS(2,2,1),14)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        CALL LO1(8,0,8,0)
900     RETURN
        END
```

```
C     ****************************************************************************
C     **  OVERLAY VBS3                MANGER FEBRUAR 1984                       *
C     **  SUBR ZUR UEBERPRUEFUNG UND AUSFUEHRUNG DER MANUELLEN                  *
C     **  BEFEHLE.                                  (PUMPEN)                    *
C     ****************************************************************************

      COMPILER NOSTACK
      OVERLAY OVB3
      SUBROUTINE VBS3(IS)

      COMMON/FRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

      LOGICAL LPT,STAT,IHA,SIAB
      INTEGER ILIS(5,5),IMAS(5,5),IN(5),IM(5,12),IL(5,12)

      DATA ILIS/0K,10000K,0K,200K,300K,0K,40000K,0K,400K,1000K,1K,0K,0K,
     *          1000K,6000K,4K,2*0K,2000K,30000K,100K,0K,0K,4000K,40000K/
      DATA IMAS/0K,30000K,0K,200K,300K,0K,140000K,0K,400K,1000K,3K,0K,0K,
     *          1000K,6000K,14K,2*0K,2000K,30000K,140K,0K,0K,4000K,40000K/
      DATA IL/60*0K/
      DATA IM/0K,20000K,4*0K,100000K,3*0K,2K,4*0K,10K,4*0K,40K,5*0K,240K,
     *        4*0K,3000K,4*0K,10000K,4*0K,40000K,3*0K,1K,4*0K,4K,4*0K,
     *        120K,4*0K/

      IN(1)=INEU(1,3)
      IN(2)=INEU(2,3)
      IN(3)=INEU(1,4)
      IN(4)=INEU(2,4)
      IN(5)=INEU(2,5)
      ID=42
      IF(IS.GT.5) ID=43
      IP=IS+23

C     ================    BEFEHLSPRUEFUNG   ====================================

      IF(.NOT.ITEST(INEU(2,2),IS-1)) GOTO 30        ; PUMPE AUS?
      IF(IS.GE.6) GOTO 500                          ; IST`S 'NE ROTATIONSPUMPE

C     ============================================================================

      DO 10 K=1,5                                   ; PRUEFUNG DIFF-PUMPE
      DO 20 I=1,16
      IF(.NOT.ITEST(IMAS(K,IS),I-1)) GOTO 20
      IF(ITEST(ILIS(K,IS),I-1).NE.ITEST(IN(K),I-1)) GOTO 600
20    CONTINUE
10    CONTINUE
      GOTO 500

C     ============================================================================

30    CONTINUE                                      ; PRUEFUNG PUMPE AUS
      DO 40 K=1,5
      DO 50 I=1,16
      IF(.NOT.ITEST(IM(K,IS),I-1)) GOTO 50
      IF(ITEST(IL(K,IS),I-1).NE.ITEST(IN(K),I-1)) GOTO 610
50    CONTINUE
40    CONTINUE
      GOTO 550
```

```
C       ====================  BEFEHL EIN AUSGEBEN ==================================

500     CALL LO1(7,ID,8,IF)
        CALL ISET(IAUS(2,2,1),IS-1)
        CALL ISET(IDEL(2,2),IS-1)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        CALL FDELY(20*IFDEL)
        IF(IDEL(2,2).NE.0) GOTO 590
        GOTO 900

C       ====================== BEFEHL AUS AUSGEBEN  ==================================

550     CALL LO1(7,ID,9,IF)
        CALL ICLR(IAUS(2,2,1),IS-1)
        CALL ISET(IDEL(2,2),IS-1)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        CALL FDELY(20*IFDEL)
        IF(IDEL(2,2).NE.0) GOTO 590
        GOTO 900

C       ====================  RESET BEFEHL  ========================================

590     CALL LO1(7,0,14,0)
        IDEL(2,2)=0
        GOTO 620
600     CALL LO1(7,ID,10,IF)
        GOTO 620
610     CALL LO1(7,ID,11,IF)
620     CALL ISET(IAUS(2,2,1),14)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        CALL ICLR(IAUS(2,2,1),14)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
        MAFLA(4)=1                                  ; FLAGGE FUER RESET
        CALL LO1(8,0,8,0)                           ; UPDATEN
900     CALL LO1(8,0,8,0)
        MAFLA(4)=0
        RETURN
        END
```

4.2.2 Task TT1 (Service-Task für Tastatur-Interrupts)

TT1 bedient mit Unterstützung durch die (Nicht-FORTRAN-)Task TTIT /7/ Interrupts von derjenigen der beiden Terminal-Tastaturen des E.R. II, die on-line geschaltet ist. (Wenn beide Tastaturen on-line geschaltet sind, ist Eingabe von jeder der beiden Tastaturen aus möglich.) In der vorliegenden Programmversion werden nur die Interrupt-Codes 327 (Drücken der Escape-Taste) und 330 (Drücken einer der Funktionstasten F1 bis F5, siehe 4.2) akzeptiert. Alle anderen TT-Interrupts werden bereits in LJ2 (4.2) mit der Fehlermeldung

**** LJ2: ILLEGAL INTERRUPT

protokolliert.

Nach dem Drücken der ESC-Taste lädt TT1 zunächst die Overlay-Subroutine TTS1 (4.2.2.1) mittels

```
CALL LO1(1,0,0,0)
```

siehe 4.2. TTS1 fordert die Eingabe eines von 4 vorgesehenen Kommandos an. Wenn VSTA eingegeben wird, setzt TTS1 die Flagge STAT=.TRUE. und kehrt nach TT1 zurück. TT1 ruft seinerseits in Abhängigkeit von der Flagge STAT die Overlay-Subroutine VST1(4.2.2.2) auf mittels

```
CALL LO1(5,0,0,0)
```

Mit Suspendierung und (sobald TT1 wieder executing ist) Rücksprung zum Anfang endet TT1.

TT1 hat die Task-Identifikations-Nr. 4 und die Priorität 40.

Aufruf:

```
COMMON /PRM/ .... (benötigt werden IC, LPT, STAT)
 :
CALL RELSE(4,IER)
```

```
C
C       ************************************************************************
C       *     SERVICE-TASK FUER TELETYPE INTERRUPTS                            *
C       *     MOMENTAN NUR VAKUUMANLAGE, SPAETER AUCH FUER MAGNETE USW.        *
C       *     MANGER FEBRUAR 1984                                              *
C       ************************************************************************

        TASK TT1

        COMMON/FRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAFRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

        LOGICAL LPT,STAT,IHA,SIAB

1       CALL LO1(1,0,0,0)                              ; TTS1 LADEN

        IF(STAT) CALL LO1(5,0,0,0)                     ; VST1 LADEN

        CALL SUSP

        GOTO 1

        END
```

4.2.2.1 Subroutine TTS1 (Kommandoeingabe über Tastatur)

TTS1 ermöglicht über die Terminal-Tastatur die Eingabe von Kommandos, für die keine eigenen Befehls- oder Sensorboard-Tasten vorhanden sind. Für die Vakuumanlage kommen in Betracht:

* Protokollausgabe aller Istzustände "M", "F", "H" auf dem Terminalschirm und auf dem Drucker (LPT);

* Updaten der Ausgabepegel "A" und der Befehls-FF's gemäß dem aktuellen Istzustand der Anlage nach Unregelmäßigkeiten, z.B. Umbauten, verlorenen Interrupts usw.;

* Gezieltes Einleiten der automatischen, zeitlich gestuften Abschaltung der gesamten Vakuumanlage;

* Ordnungsgemäße Beendigung von LJ2.

In TTS1 sind diese 4 genannten Optionen verwirklicht.

Nach dem Drücken der ESC-Taste erscheint auf dem Terminalschirm die Aufforderung:

COMAND (VSTA/VUPD/VOFF/EXIT):

Die eingegebenen Kommandostrings dürfen von beliebiger Länge sein, doch werden nur die ersten 4 Characters eingelesen und geprüft. Leereingabe (bloßes Drücken der RETURN-Taste) führt zur sofortigen Suspendierung von TT1. Stimmen die ersten 4 Characters nicht mit "VSTA", "VUPD", "VOFF", "EXIT" oder " " überein, dann wird die Aufforderung wiederholt.

Das Kommando VSTA bewirkt Ausgabe der in der Variablen INEU() gehaltenen Istzustände "M", "F", "H" der Vakuumanlage in Tabellenform auf dem Terminalschirm und/oder auf dem Drucker (Subroutine VST1, 4.2.2.2). Der Aufruf dieser Subroutine, die mit TTS1 im gleichen Overlay-Segment # 0 liegt (Abb. 2), erfolgt erst nach Rückkehr zu TT1 von TT1 aus, siehe 4.2.2, sodaß gegenseitiges Überschreiben vermieden wird.

Das Kommando VUPD dient dazu, mit Hilfe der Subroutine VU1 (4.2.2.3) sowohl die Befehls-FF's (über die Reset-Impulse "R") als auch die Steuerpegel "A" mit dem aktuellen Istzustand der Vakuumanlage in Übereinstimmung zu bringen. Dies kann in dem (voraussichtlich seltenen) Fall nützlich sein, daß während der (sehr kurzen) Interrupt-Totzeiten, während derer der CAMAC-Branch disabled ist, Anlagen-Interrupts verloren gehen, oder daß von sehr schnell aufeinanderfolgenden ("geschachtelten") Anlagen- Interrupts nur der 1. Folge-Interrupt bedient wird, die anderen aber ignoriert werden. VUPD wird mit CALL LO1(8,0,0,0) geladen.

Das Kommando VOFF bewirkt die beabsichtigte Abschaltung (im Gegen-

satz zur Sicherheitsabschaltung) der gesamten Vakuumanlage (nicht etwa einzelner Sektionen) durch Ready-machen der Task VO1 (4.2.4).

Das Kommando EXIT bewirkt die ordnungsgemäße Beendigung von LJ2 durch Aufruf der Subroutine EX1 (4.2.2.4).

Mit Return zu TT1 wird TTS1 beendet.

TTS1 hat den Overlay-Namen OTT1, die Identifikations-Nr. ION=1 (siehe 4.2.7), liegt im Overlay-Segment # 0 und wird aufgerufen mittels

```
COMMON /PRM/ .... (benötigt werden IC, STAT, SIAB)
 :
CALL LO1(1,0,0,0).
```

```
C
C       ************************************************************************
C       *      SUBROUTINE  FUER TELETYPE INTERRUFTS                            *
C       *      MANGER FEBRUAR 1984                                             *
C       ************************************************************************
        COMPILER NOSTACK
        OVERLAY OTT1
        SUBROUTINE TTS1

        COMMON/FRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

        LOGICAL LPT,STAT,IHA,SIAB
        EXTERNAL LO1,OVP1,VP1,OVST1,VST1
        INTEGER IWORT(2)

        IF(IC.EQ.327) GOTO 100                          ; ESCAPE - INTERRUPT ?
        GOTO 900
C       ==================    KOMMANDOEINGABE   ==========================

100     WRITE(10,1000)
1000    FORMAT("                          COMMANDO :(VSTA/VUPD/VOFF/EXIT): ")
        READ(11,1100) IWORT
1100    FORMAT(2A2)

C       ==================        STATUS AUSGEBEN ========================

        IF(IWORT(1).NE."<126><123>".OR.IWORT(2).NE."<124><101>") GOTO 120
        STAT=.TRUE.
        GOTO 900

C       ====================    UPDATEN   ==================================

120     IF(IWORT(1).NE."<126><125>".OR.IWORT(2).NE."<120><104>") GOTO 150
        CALL LO1(8,0,0,0)
        GOTO 900

C       ================    VAKUUMANLAGE AUSSCHALTEN ======================

150     IF(IWORT(1).NE."<126><117>".OR.IWORT(2).NE."<106><106>") GOTO 160
        SIAB=.FALSE.
        CALL  LO1(7,0,3,0)
        CALL RELSE(2,IER)
        IF(IER.NE.1) CALL  LO1(10,3,13,0)
        GOTO 900

C       ==================    LJ1  BEENDEN  ==============================

160     IF(IWORT(1).NE."<105><130>".OR.IWORT(2).NE."<111><124>") GOTO 170
        CALL LO1(7,0,5,0)
        CALL LO1(10,3,18,0)
        GOTO 900
C       ==================  LEEREINGABE  ================================

170     IF(IWORT(1).NE."<40><40>".OR.IWORT(2).NE."<40><40>") GOTO 100

900     RETURN
        END
```

4.2.2.2 Subroutine VST1 (Statusausgabe)

VST1 gibt den aktuellen Istzustand der Vakuumanlage auf dem Terminalschirm und/oder auf dem Drucker nach folgendem Muster aus:

Tab. 5: Vollständige Status-Ausgabe der Vakuumanlage

hh:mm:ss ACTUAL STATE OF THE VACUUM SYSTEM

V 1= 1	V11= 0	V41= 0	D24= 1	F 1= 1	F11= 1
V 2= -	V12= 0	V42= 1	T24= 1	F 2= 1	F12= 1
V 3= 1	V13= 1	V43= 1	D26= 1	F 3= 1	F13= 1
V 4= -	V14= 1		T26= 1	F 4= 1	F14= 1
V 5= -	V15= 1	R22= 0	D28= 1	F 5= 0	F15= 1
V 6= 0	V16= 1	R23= 1	T28= 1	F 6= 1	
V 7= 0	V17= 1	R25= 1	D30= 1	F 7= 1	
V 8= 1	V18= 1	R27= 1	T30= 1	F 8= 1	H 1= 1
V 9= 0	V19= 1	R29= 1	D45= 1	F 9= 1	H 2= 1
V10= 0	V20= 1	R44= 1	T45= 1	F10= 1	H 3= 1

Als Beispiel sind die Zustände eingetragen, die dem derzeitigen Ausbauzustand der Vakuumanlage entsprechen und die bei der Betriebsart "Hochvakuum-pumpen" vorzufinden sind.

Die Kurzangaben haben folgende Bedeutung:

Tab. 6: Kurzbezeichnungen der Istzustände im Statusprotokoll

	Ventile	Rotationspumpen	Diffusionspumpen	
	V__	R__	D__	T__
0	zu	aus	Heizung aus	nicht betriebsbereit
1	auf	ein	" ein	betriebsbereit

	Feinvakuum- / Hochvakuum-Wächter
0	Druck größer als die eingestellte Schwelle (Vakuum schlecht)
1	" kleiner " " " " (Vakuum gut)

Für Ist-Zustandsmeldungen von Aggregaten und Meßstellen, die in LJ1 (4.1) mittels des Anlagenparameters MASKE() als nichtexistent ausgeblendet sind, wird, wie in Tab. 5 gezeigt, ein Minuszeichen ausgegeben.

VST1 kann auch unter Umgehung der Task TT1 direkt angesprungen werden. Diese Möglichkeit wird von LJ2 (4.2) (beim Umschalten HAND -> RECHNER und umgekehrt) und von VM1 (4.2.3) (zur Dokumentation des Anlagenzustands nach einer Zustandsänderung, die zur Sicherheitsabschaltung führte) genutzt.

Nach der Druckausgabe setzt VST1 die Flagge STAT auf .FALSE. und kehrt zum aufrufenden Programm zurück.

VST1 hat den Overlay-Namen OVST1, die Identifikations-Nr. ION=5 (siehe 4.2.7), befindet sich im Overlay-Segment # 0 und wird aufgerufen mittels

```
COMMON /PRM/ .... (benötigt werden (MASKE(), INEU(), LPT, STAT)
 :
CALL LO1(5,0,0,0).
```

4.2.2.3 Subroutine VU1 (Updaten der Befehls-FF's)

Die Subroutine VU1 dient dazu, den aktuellen Istzustand (Meldungen "M") der Vakuumanlage einzulesen (INEU()) und sowohl die Befehs-FF's als auch die Steuerpegel "A" mit INEU() in Übereinstimmung zu bringen, wie es von der Steuer-Hardware /8/ verlangt wird. VU1 wird u.a. von LJ2 (4.2) aufgerufen, um den Fall abzudecken, daß bei ausgefallenem Steuerrechner im RECHNER-Betrieb (das Bit READY bleibt gesetzt und die Multiplexer RECHNER/HAND in Ü6 bleiben auf RECHNER geschaltet /8/) die Befehls-FF's durch Tastenbetätigung umgeschaltet werden und dann nicht mehr mit dem Ist-Zustand der Vakuumanlage übereinstimmen. Ferner wird VU1 am Ende von VB1 (4.2.1) herangezogen, um bei geschachtelten Befehls-Interrupts die zwar schon umgeschalteten, aber nicht mehr zur Wirkung kommenden Befehls-FF's wieder zu resetten. Während des Updatens werden die CAMAC-Input-Module # 1 und # 2 disabled. Die RESET-Impulse dauern 0.1 sec.

Aufruf:

```
COMMON /PRM/ .... (benötigt werden A3473(), IALT(), INEU(), A3072(),
                                                    IAUS(), IDEL())
 :
CALL LO1(8,0,IA2,0)
```

Das Argument IA2, das von LO1 an VU1 übergeben wird, hat folgende Bedeutung:

IA2 = 0: MAFLA(4) wird in VU1 gleich 0 gesetzt;
IA2 ¬= 0: MAFLA(4) " " " " 1 " .

Zur Erläuterung: VU1 wird dann ohne Unterdrückung der Befehlprüfung aufgerufen, wenn VU1 zum Updaten der Memory-Register der CAMAC-Input-Module und nicht zum Updaten der Befehls-Flipflops benutzt wird.

```
C
C       ***************************************************************
C       * SUBROUTINE VST1    TERMINAL/DRUCKER-AUSGABE MANGER JANUAR 1984     *
C       ***************************************************************
        COMPILER NOSTACK
        OVERLAY OVST1
        SUBROUTINE VST1

        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
        LOGICAL LPT,STAT,IHA,SIAB,L1,L2,I1,I2
        EXTERNAL LO1,OVP1,VP1
        DIMENSION IST(16,2,3)
        DO 110 K=1,3                                    ; AKTUELLE ZUSTAENDE ZU-
        DO 110 J=1,2                                    ;       WEISEN
        DO 110 I=1,16
        M=K+2
        L1=ITEST(MASKE(J,K),I-1)
        L2=ITEST(INEU(J,M),I-1)
        IF(L1.AND.L2) IST(I,J,K)="1"
        IF(L1.AND..NOT.L2) IST(I,J,K)="0"
        IF(.NOT.L1) IST(I,J,K)="-"
110     CONTINUE
        CALL LO1(7,0,4,0)                               ; KOPFZEILE
C       ===============================================================
        DO 120 N=10,12,2
        WRITE(N,1200)                                   ; AUSGABE
     *  IST(1,2,1),IST(11,2,1),IST(5,1,1),IST(1,2,2),IST(1,2,3),IST(11,2,3),
     *  IST(2,2,1),IST(12,2,1),IST(6,1,1),IST(1,1,2),IST(2,2,3),IST(12,2,3),
     *  IST(3,2,1),IST(13,2,1),IST(7,1,1),IST(2,2,2),IST(3,2,3),IST(13,2,3),
     *  IST(4,2,1),IST(14,2,1),IST(6,2,2),IST(2,1,2),IST(4,2,3),IST(14,2,3),
     *  IST(5,2,1),IST(15,2,1),IST(7,2,2),IST(3,2,2),IST(5,2,3),IST(15,2,3),
     *  IST(6,2,1),IST(16,2,1),IST(8,2,2),IST(3,1,2),IST(6,2,3),
     *  IST(7,2,1),IST(1,1,1),IST(9,2,2),IST(4,2,2),IST(7,2,3),
     *  IST(8,2,1),IST(2,1,1),IST(10,2,2),IST(4,1,2),IST(8,2,3),IST(1,1,3),
     *  IST(9,2,1),IST(3,1,1),IST(11,2,2),IST(5,2,2),IST(9,2,3),IST(2,1,3),
     *  IST(10,2,1),IST(4,1,1),IST(12,2,2),IST(5,1,2),IST(10,2,3),IST(3,1,3)
1200    FORMAT(11X,35("=")/
     *  11X,"V1  : ",A2,3X,"V11 : ",A2,3X,"V41 : ",A2,3X,"D24 : ",A2,3X,"F1  : "
     *  A2,3X,"F11 : ",A2/
     *  11X,"V2  : ",A2,3X,"V12 : ",A2,3X,"V42 : ",A2,3X,"T24 : ",A2,3X,"F2  : "
     *  A2,3X,"F12 : ",A2/
     *  11X,"V3  : ",A2,3X,"V13 : ",A2,3X,"V43 : ",A2,3X,"D26 : ",A2,3X,"F3  : "
     *  A2,3X,"F13 : ",A2/
     *  11X,"V4  : ",A2,3X,"V14 : ",A2,3X,"R21 : ",A2,3X,"T26 : ",A2,3X,"F4  : "
     *  A2,3X,"F14 : ",A2/
     *  11X,"V5  : ",A2,3X,"V15 : ",A2,3X,"R22 : ",A2,3X,"D28 : ",A2,3X,"F5  : "
     *  A2,3X,"FKR : ",A2/
     *  11X,"V6  : ",A2,3X,"V16 : ",A2,3X,"R23 : ",A2,3X,"T28 : ",A2,3X,"F6  : "
     *  A2/
     *  11X,"V7  : ",A2,3X,"V17 : ",A2,3X,"R25 : ",A2,3X,"D30 : ",A2,3X,"F7  : "
     *  A2/
     *  11X,"V8  : ",A2,3X,"V18 : ",A2,3X,"R27 : ",A2,3X,"T30 : ",A2,3X,"F8  : "
     *  A2,3X,"H1  : ",A2/
     *  11X,"V9  : ",A2,3X,"V19 : ",A2,3X,"R29 : ",A2,3X,"D45 : ",A2,3X,"F9  : "
     *  A2,3X,"H2  : ",A2/
     *  11X,"V10 : ",A2,3X,"V20 : ",A2,3X,"R44 : ",A2,3X,"T45 : ",A2,3X,"F10 : "
     *  A2,3X,"H3  : ",A2)
        IF(.NOT.LPT) GOTO 130
120     CONTINUE
130     STAT=.FALSE.
        RETURN
        END
```

```
C
C       ***************************************************************
C       *       SUBROUTINE VU1     ZUM UPDATEN DER BEFEHLS-FLIP-FLOPS     *
C       *                          MANGER FEBRUAR 84                      *
C       ***************************************************************

        COMPILER NOSTACK
        OVERLAY OVU1
        SUBROUTINE VU1(IARG)
        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LFT,STAT,IHA,SIAB,IDEL(2,2),MAFRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

        LOGICAL LFT,STAT,IHA,SIAB,I1,I2,I3,I4
C       ======================================================================

        DO 10 I=1,5
10      CALL VL1(I)                                  ; EINLESEN 5 INPUTREGISTER
        CALL CAL93(24,A3473(1,1),0,IQ)               ; DISABLE LAM REGISTER 1
        CALL CAL93(24,A3473(1,2),0,IQ)               ; DISABLE LAM REGISTER 2
C       ======================================================================

        IAUS(1,1,2)=INEU(1,3)                        ; AUSGABEWORTE UPDATEN
        IAUS(2,1,2)=INEU(2,3)
        IAUS(2,2,2)=INEU(2,4)
        I3=INEU(1,3)
        I4=INEU(1,1)
        IF(IARG.EQ.8) MAFLA(4)=1                     ; FLAGGE FUER RESET
        DO 20 K=1,2                                  ; RESET-AUSGABEWORTE UPDATEN
        I1=INEU(2,K+2)
        I2=INEU(2,K)
20      IAUS(2,K,2)=(I1.OR.I2).AND.(.NOT.I1.OR..NOT.I2)
        IAUS(1,1,2)=(I3.OR.I4).AND.(.NOT.I3.OR..NOT.I4)

C       ======================================================================

        AUS(1,2)=FLOINT(IAUS(1,1,2))
        AUS(2,2)=FLOINT(IAUS(1,2,2))
        CALL CAL93(16,A3072(1,2),AUS(1,2),IQ)        ; BEFEHLS-FF'S KORRIGIEREN
        CALL CAL93(16,A3072(2,2),AUS(2,2),IQ)
        CALL FDELY(1*IFDEL)
        IAUS(2,1,2)=0                                ; RESET-PEGEL WIEDER NULLEN
        IAUS(2,2,2)=0                                ;       "          "      "
        IAUS(1,1,2)=0                                ;       "          "      "
        AUS(1,2)=FLOINT(IAUS(1,1,2))
        AUS(2,2)=FLOINT(IAUS(1,2,2))
        CALL CAL93(16,A3072(1,2),AUS(1,2),IQ)        ; JETZT WIRD RESET ERST WIRKSAM
        CALL CAL93(16,A3072(2,2),AUS(2,2),IQ)
C       ======================================================================

        CALL VL1(1)                                  ; MEMORY UPDATEN
        CALL VL1(2)
        CALL CAL93(10,A3473(1,1),0,IQ)               ; LAM-STATUS LOESCHEN
        CALL CAL93(10,A3473(1,2),0,IQ)               ;       "          "
        CALL CAL93(26,A3473(1,1),0,IQ)               ; LAM REQUEST WIEDER ENABLEN
        CALL CAL93(26,A3473(1,2),0,IQ)               ;  "       "          "

        RETURN
        END
```

4.2.2.4 Subroutine EX1 (LJ2 beenden)

EX1 dient zur ordnungsgemäßen Beendigung von LJ2, sei sie beabsichtigt oder fehlerbedingt. Im letzteren Fall gibt EX1 den Namen der Programmeinheit, in der der Fehler aufgetreten ist, sowie die Art des Fehlers in Klartext auf dem Terminalschirm (nur auf diesem) sowie ein akustisches Signal ("bell") aus, z.B.:

**** LJ2: CHANNEL 1 NOT OPENED

Das Format der Fehlermeldung wird durch das Argument IA2 bestimmt. Die Beendigung von LJ2 beinhaltet im einzelnen:

* LED's "RECHNER BEREIT" und "RECHNERBETRIEB" löschen;

* File "LJ2.OL" (Channel-Nr. 2) schließen;

* CAMAC-Module disablen;

* LAM-Generator der CAMAC-Uhr disablen;

* CAMAC-Crates disablen;

* CAMAC bei RDOS abmelden (CALL CA199)

EX1 liegt im Segment # 1 des Files "LJ2.OL" (Abb. 2). EX1 hat die Identifikations-Nr. ION=10, siehe 4.2.7, und wird mittels der Subroutine LO1 (4.2.7) geladen und aufgerufen:

```
COMMON /PRM/ .... (benötigt wird IAUS(), AUHR, A3473(), A3072(), ACR())
 :
CALL LO1(10,0,IA2,0)
```

```
C
C       ************************************************************************
C       *       SUBROUTINE EX1 ZUM BEENDEN VON LJ2 UND ZUR AUSGABE             *
C       *                      VON FEHLERMELDUNGEN                             *
C       *                      MANGER FEBRUAR 1984                             *
C       ************************************************************************

        OVERLAY OEX1
        SUBROUTINE EX1(I)

        COMMON/FRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAFRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
        LOGICAL LPT,STAT,IHA,SIAB

C       =======  FEHLERPROTOKOLL BEI UNERWUENSCHTEM BEENDEN ====================

        GOTO (10,20,30,40,50,60,70,80,900,100,110,900,130,140,150,160,170,
     *  900),I

10      TYPE "*** LJ2: CHANNEL 1 NOT OPENED<7>"
        GOTO 900
20      TYPE "*** LJ2: FILE VPA NOT READ<7>"
        GOTO 900
30      TYPE "*** LJ2: CHANNEL 1 NOT CLOSED<7>"
        GOTO 900
40      TYPE "*** LJ2: TASK VM1  NOT ACTIVATED<7>"
        GOTO 900
50      TYPE "*** LJ2: TASK VO1 NOT ACTIVATED<7>"
        GOTO 900
60      TYPE "*** LJ2: TASK VB1  NOT ACTIVATED<7>"
        GOTO 900
70      TYPE "*** LJ2: TASK TT1  NOT ACTIVATED<7>"
        GOTO 900
80      TYPE "*** LJ2: TASK UHR1 NOT ACTIVATED<7>"
        GOTO 900
100     TYPE "*** LJ2: ONE OF THE SUBTASKS NOT SUSPENDED<7> "
        GOTO 900
110     TYPE "*** LJ2: TASK TT1  NOT READIED<7>"
        GOTO 900
130     TYPE "*** LJ2: TASK VM1 NOT READIED<7>"
        GOTO 900
140     TYPE "*** LJ2: TASK VB1 NOT READIED<7>"
        GOTO 900
150     TYPE "*** LJ2: TASK TT1 NOT READIED<7>"
        GOTO 900
160     TYPE "*** LJ2: TASK UHR1 NOT READIED<7>"
        GOTO 900
170     TYPE "*** LJ2: ILLEGAL INTERRUPT"
        GOTO 900


C       ====== LJ2 BEENDEN                      ================================
C       ====== DER REIHE NACH :                 ================================
C       ====== LED RECHNER BEREIT LOESCHEN      ================================


900     CALL ICLR(IAUS(2,2,1),12)
        CALL ICLR(IAUS(2,2,1),13)
        AUS(2,1)=FLOINT(IAUS(1,2,1))
        CALL CAL93(16,A3072(2,1),AUS(2,1),IQ)
```

```
C       ===== OVERLAY FILE SCHLIESSEN ==========================================

        CALL CLOSE (2,IER)
        IF(IER.NE.1) TYPE "*** EX1: CHANNEL #2 NOT CLOSED<7> "

C       ===== CAMAC-LAMS DISABLEN ==============================================

        DO 910 I=1,5
910     CALL CAL93(24,A3473(1,I),DUMMY,IQ)

C       ===== CAMAC-UHR-LAM-GENERATOR AUSSCHALTEN ==============================

        CALL CAL93(24,AUHR,DUMMY,IQ)

C       ===== CAMAC-BRANCH DISABLEN ============================================

        CALL CAL93(24,ACR(1),DUMMY,IQ)
        CALL CAL93(24,ACR(2),DUMMY,IQ)

C       ===== CAMAC BEI RIOS ABMELDEN ==========================================

        CALL CA199

        STOP
        END
```

4.2.3 TASK VM1 (Service-Task für Anlagen-Interrupts)

VM1 identifiziert bei einem Anlagen-Interrupt, welcher der 3072-Input-Module # 3 bis # 5 den LAM-Request gesendet hat und welche der möglichen Zustandsänderungen:

```
M V__  1/0
M R__  1/0
M D__  1/0
M T__  1/0
F__    </>
H_     </>
```

vorliegt. Das Scannen erfolgt in der angegebenen Reihenfolge.

Die Unterscheidung zwischen erwarteten oder harmlosen spontanen Zustandsänderungen einerseits und gefährlichen spontanen Zustandsänderungen andererseits geschieht nach folgender Strategie:

* Die Meldungen H_ >/< werden lediglich protokolliert.

* Alle Meldungen vom Typ "auf", "ein", "betriebsbereit", und "Feinvakuum gut" werden als befehlsbedingt bzw. als harmlos eingestuft.

* Für die Meldungen "Feinvakuum schlecht" gilt allgemein:
  Die Diffusionspumpen sind bei einem Druckanstieg von F1, F2, F3 bzw. F6 bereits hardwaremäßig geschützt: Das Ventil über der/den betroffenen Diffusionspumpen fällt zu, und VM1 leitet daraufhin die Sicherheitsabschaltung ein. Für die Rotationspumpen wird ein vorübergehender Druckanstieg von 5 min in Kauf genommen: Da VM1 keine Information über den zeitlichen Verlauf und über die Größe des Druckanstiegs hat, kann es nicht zwischen betrieblich zulässigen bzw. erwarteten Druckanstiegen, z.B. beim Anpumpen von Atmosphärendruck herab, beim Anheizen einer Öldiffusionspumpe, beim Ausheizen der Kryopumpe, bei Leckratenmessungen usw., einerseits und Betriebsstörungen wie Lufteinbruch, Wassereinbruch, Folienriß usw. andererseits unterscheiden.
  Die Meldungen F15 >, F7 >, F8 >, F9 >, F10 >, F11 >, F12 >, F13 > und F14 > werden als harmlos angesehen, wenn 5 min nach Eintreffen der Meldung die zugehörige Pumpe ausgeschaltet bzw. das zugehörige Ventil geschlossen ist oder der Feinvakuumdruck inzwischen wieder gut geworden ist.

  Im einzelnen gelten folgende Festlegungen:

Druckanstieg	gilt als harmlos, wenn nach 5 min					
F1 >	V41	zu	oder	inzwischen	wieder	F1 <
F2 >	V10	"	"	"	"	F2 <
F3 >	V6	"	"	"	"	F3 <
F4 >	R21	aus	"	"	"	F4 <
F5 >	R22	"	"	"	"	F5 <
F6 >	V11	zu	"	"	"	F6 <
F7 >	R23	aus	"	"	"	F7 <
F8 >	V13	zu	"	"	"	F8 <

Druckanstieg	gilt als harmlos, wenn nach 5 min				
F9 >	R25	aus	oder inzwischen wieder		F9 <
F10 >	V15	zu	"	"	F10 <
F11 >	R27	aus	"	"	F11 <
F12 >	V17	zu	"	"	F12 <
F13 >	R29	aus	"	"	F13 <
F14 >	V19	zu	"	"	F14 <
F15 >	V42	"	"	"	F15 <

Alle diese, um 5 min verzögerten Zulässigkeitsprüfungen von F__ > erfolgen in VFM1 (4.2.3.1).

* Jede Meldung, die innerhalb 1 sec (IDEL()) nach dem entsprechenden vorausgegangenen Befehl auftritt, wird als befehlsbedingt angesehen.

* Alle anderen Meldungen leiten die Sicherheitsabschaltung ein (bei unzulässigen Druckanstiegen erst 5 min nach Eingang der Meldung, siehe VFM1, 4.2.3.1).

Alle Meldungen werden protokolliert, entweder

```
hh:mm:ss  V 9= 0                                              oder:

hh:mm:ss  F 2= 0
          SAFETY SHUT OFF IN PROGRESS
```

VM1 ist hauptspeicherresident (Abb. 2); es hat die Task-Identifikations-Nr. 1, die Priorität 10 und wird wie folgt aufgerufen:

```
COMMON /PRM/ .... (benötigt werden IALT(), INEU(), MAPRO(), MAFLA(),
                                                    IDUM(), A3473())
 :
CALL RELSE(1,IER)
```

Der Programmablauf gliedert sich in folgende Schritte:

* Identifizieren, welches Bit sich in den einzelnen Input-Modulen # 5 bis # 3 geändert hat und in welcher Richtung es sich geändert hat;

* VP1 (Ausgabe der Protokollzeile) laden mittels
  CALL LO1(7,Aggregatkürzel, Protokolltext, Aggregatnummer)
  LO1 gibt die letzten 3 Argumente an VP1 weiter.

* Abfragen der Zeitbedingungen: Läuft 1-sec-Zeitintervall (4.2.1.1, 4.2.4) noch oder nicht?

* Prüfen, ob inzwischen ein neuer LAM-Request ansteht: Wenn der Versuch, ein eventuelles LAM der CAMAC-Input-Module # 3 bis # 5 zu löschen, erfolgreich ist (erkenntlich am Rückgabeargument IQ = 1), wird an den Anfang von VM1 zurückgesprungen; sonst:

* CALL SUSP und, sobald VM1 wieder executing wird, Rücksprung an den Anfang von VM1.

```
C
C       ***************************************************************
C       *   TASK VM1     ZUR IDENTIFIKATION ALLER MELDUNGEN UND EVENTUELLEN     *
C       *                EINLEITUNG DER SICHERHEITSABSCHALTUNG                  *
C       *                MANGER FEBRUAR  1984                                   *
C       ***************************************************************
        TASK VM1
        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LFT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
        LOGICAL LFT,STAT,IHA,SIAB

1       IF(MAFLA(5).EQ.1) GOTO 920              ; MELDUNGEN UNTERDRUECKEN

C       ================    MELDUNGEN F IDENTIFIZIEREN  ================================

        DO 5 I=1,15
        IF(ITEST(INEU(2,5),I-1).AND..NOT.ITEST(IALT(2,5),I-1)) GOTO 10
        IF(.NOT.ITEST(INEU(2,5),I-1).AND.ITEST(IALT(2,5),I-1)) GOTO 15
        GOTO 5
10      CALL LO1(7,44,6,I)                      ; MELDUNG GUT PROTOKOLLIEREN
        GOTO 5
15      CALL LO1(7,44,7,I)                      ; MELDUNG SCHLECHT PROTOKOLLIEREN
        IDUM(I)=5                               ; ZEITVERZOEGERUNG FUER CHECK
5       CONTINUE
        IALT(2,5)=INEU(2,5)

C       ======= AENDERUNG DER BETRIEBSBEREITSCHAFT DER DIFF-PUMPEN ================

        DO 200 I=24,28
        IF(ITEST(INEU(1,4),I-24).AND..NOT.ITEST(IALT(1,4),I-24)) GOTO 20
        IF(.NOT.ITEST(INEU(1,4),I-24).AND.ITEST(IALT(1,4),I-24)) GOTO 25
        GOTO 200
20      CALL LO1(7,40,6,I)                      ; MELDUNG BB PROTOKOLLIEREN
        GOTO 200
25      CALL LO1(7,40,7,I)                      ;     "     NBB       "
200     CONTINUE
        IALT(1,4)=INEU(1,4)

C       ================  M V1 BIS M V16 IDENTIFIZIEREN ============================

        DO 300 I=1,16
        IF(ITEST(INEU(2,3),I-1).AND..NOT.ITEST(IALT(2,3),I-1)) GOTO 30
        IF(.NOT.ITEST(INEU(2,3),I-1).AND.ITEST(IALT(2,3),I-1)) GOTO 35
        GOTO 300
30      CALL LO1(7,41,6,I)                      ; AUF PROTOKOLLIEREN
        CALL ICLR(IDEL(2,1),I-1)                ; RUECKMELDUNG AN VBS1
        GOTO 300
35      CALL LO1(7,41,7,I)                      ; ZU        "
        IF(.NOT.ITEST(IDEL(2,1),I-1)) GOTO 800
        CALL ICLR(IDEL(2,1),I-1)                ; RUECKMELDUNG AN VBS1
300     CONTINUE
        IALT(2,3)=INEU(2,3)

C       ================  M V17  BIS M V43 IDENTIFIZIEREN ===========================

        DO 500 I=17,23
        IF(ITEST(INEU(1,3),I-17).AND..NOT.ITEST(IALT(1,3),I-17)) GOTO 40
        IF(.NOT.ITEST(INEU(1,3),I-17).AND.ITEST(IALT(1,3),I-17)) GOTO 45
        GOTO 500
40      CALL LO1(7,41,6,I)                      ; AUF PROTOKOLLIEREN
        CALL ICLR(IDEL(1,1),I-17)               ; RUECKMELDUNG AN VBS2
        GOTO 500
45      CALL LO1(7,41,7,I)                      ; ZU        "
```

```
C
        IF(.NOT.ITEST(IDEL(1,1),I-17)) GOTO 800
        CALL ICLR(IDEL(1,1),I-17)                ; RUECKMELDUNG AN VBS2
500     CONTINUE
        IALT(1,3)=INEU(1,3)

C       ======================  M D24  BIS M R44  IDENTIFIZIEREN ==================

        J=42
        DO 600 I=24,35
        IF(ITEST(INEU(2,4),I-24).AND..NOT.ITEST(IALT(2,4),I-24)) GOTO 50
        IF(.NOT.ITEST(INEU(2,4),I-24).AND.ITEST(IALT(2,4),I-24)) GOTO 55
        GOTO 600
50      IF(I.GT.28) J=43                         ; EIN PROTOKOLLIEREN
        CALL LO1(7,J,6,I)
        CALL ICLR(IDEL(2,2),I-24)                ; RUECKMELDUNG AN VBS3
        GOTO 600
55      IF(I.GT.28) J=43
        CALL LO1(7,J,7,I)                        ; AUS PROTOKOLLIEREN
        IF(.NOT.ITEST(IDEL(2,2),I-24)) GOTO 800
        CALL ICLR(IDEL(2,2),I-24)                ; RUECKMELDUNG AN VBS3
600     CONTINUE
        IALT(2,4)=INEU(2,4)

C       ================  MELDUNGEN H IDENTIFIZIEREN  ==============================

        DO 700 I=1,3
        IF(ITEST(INEU(1,5),I-1).AND..NOT.ITEST(IALT(1,5),I-1)) GOTO 60
        IF(.NOT.ITEST(INEU(1,5),I-1).AND.ITEST(IALT(1,5),I-1)) GOTO 65
        GOTO 700
60      CALL LO1(7,45,6,I)                       ; GUT PROTOKOLLIEREN
        GOTO 700
65      CALL LO1(7,45,7,I)                       ; SCHLECHT PROTOKOLLIEREN
700     CONTINUE
        IALT(1,5)=INEU(1,5)
        GOTO 900

C       ======= UNZULAESSIGE ZUSTANDSAENDERUNG , SICHERHEITSABSCHALTUNG ==========

800     CALL RELSE(2,IER)                        ; VO1 AKTIVIEREN
        IF(IER.NE.1) CALL LO1(10,0,0,0)

C       ==========================================================================

900     DO 910 I=3,5                             ; ZUSTANDSAENDERUNG M ODER F
        CALL CAL93(27,A3473(1,I),DUMMY,IQ)       ; SEIT LETZTEM UPDATEN DER
        IF(IQ.EQ.0) GOTO 910                     ; MEMORY - REGISTER ??
        GOTO 1                                   ; DANN VON VORNE

910     CONTINUE
920     CALL SUSP
        GOTO 1
        END
```

4.2.3.1 Subroutine VFM1

VFM1 überprüft nach einer Verzögerung von 5 min die Zulässigkeit der Feinvakuummeldungen "schlecht" (siehe 4.2.3) wie folgt:

Beim Eintreffen einer Meldung "Fi >" (i = 1 bis 15) wird in VM1 IDUM(i)=5 gesetzt und dadurch eine 5-min-Stoppuhr gestartet, siehe UHR1, 4.2.5). Nach Ablauf der 5-min-Frist wird VFM1 von UHR1 aufgerufen. In VFM1 wird die betreffende Feinvakuummeldung auf Zulässigkeit überprüft. Bei Unzulässigkeit wird die Sicherheitsabschaltung eingeleitet.

VFM1 wird wie folgt aufgerufen:

```
COMMON /PRM/ .... (benötigt wird INEU())
 :
CALL VFM1(IV)                                                    mit
```

IV = Nummer der Feinvakummeldung.

```
C
C       ***************************************************************************
C       *    SUBROUTINE FUER DIE VERZOEGERTE UEBERPRUEFUNG DER FEINVAKUUM-        *
C       *    MELDUNGEN                                                            *
C       *    MANGER    NOVEMBER 1983                                              *
C       ***************************************************************************

        SUBROUTINE VFM1(IV)

        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LFT,STAT,IHA,SIAB,IDEL(2,2),MAFRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

        LOGICAL LFT,STAT,IHA,SIAB

        IF(.NOT.ITEST(INEU(2,5),IV-1)) GOTO (1,2,3,4,5,6,7,8,9,10,11,12,
     *  13,14,15),IV
        RETURN

1       IF(.NOT.ITEST(INEU(1,3),4)) RETURN                    ; F1/V41
        GOTO 20
2       IF(.NOT.ITEST(INEU(2,3),9)) RETURN                    ; F2/V10
        GOTO 20
3       IF(.NOT.ITEST(INEU(2,3),5)) RETURN                    ; F3/R21
        GOTO 20
4       IF(.NOT.ITEST(INEU(2,4),5)) RETURN                    ; F4/R21
        GOTO 20
5       IF(.NOT.ITEST(INEU(2,4),6)) RETURN                    ; F5/R22
        GOTO 20
6       IF(.NOT.ITEST(INEU(2,3),10)) RETURN                   ; F6/V11
        GOTO 20
7       IF(.NOT.ITEST(INEU(2,4),7)) RETURN                    ; F7/R23
        GOTO 20
8       IF(.NOT.ITEST(INEU(2,3),12)) RETURN                   ; F8/V13
        GOTO 20
9       IF(.NOT.ITEST(INEU(2,4),8)) RETURN                    ; F9/R25
        GOTO 20
10      IF(.NOT.ITEST(INEU(2,3),14)) RETURN                   ; F10/V15
        GOTO 20
11      IF(.NOT.ITEST(INEU(2,4),9)) RETURN                    ; F11/R27
        GOTO 20
12      IF(.NOT.ITEST(INEU(1,3),0)) RETURN                    ; F12/V17
        GOTO 20
13      IF(.NOT.ITEST(INEU(2,4),10)) RETURN                   ; F13/R29
        GOTO 20
14      IF(.NOT.ITEST(INEU(1,3),2)) RETURN                    ; F14/V19
        GOTO 20
15      IF(.NOT.ITEST(INEU(1,3),6)) RETURN                    ; F15/V43

20      CALL RELSE(2,IER)
        IF(IER.NE.1) CALL LO1(10,0,5,0)
        RETURN

        END
```

Tab. 3: Sicherheits- und Handabschaltung

Aggregat	Sicherheitsabschaltung				Handabschaltung			
	0 sec	5 sec	2 min	maximal 60 min	0 sec	5 sec	2 min	60 min
V41	zu	=	=	=	zu	=	=	=
V43	zu	auf	zu 1)	zu 6)	zu	auf	zu 6)	zu
V42	zu	=	=	=	zu	=	=	=
V1	zu	=	=	=	zu	=	=	=
V2	zu	=	=	=	zu	=	=	=
V3	zu	=	=	=	zu	=	=	=
V4	zu	=	=	=	zu	=	=	=
V5	zu	=	=	=	zu	=	=	=
V6	zu	=	=	=	zu	=	=	=
V7	zu	=	=	=	zu	=	=	=
V8	zu	=	=	=	zu	=	=	=
V9	zu	=	=	=	zu	=	=	=
V10	zu	=	=	=	zu	=	=	=
V11	zu	=	=	=	zu	=	=	=
V12	zu	=	=	=	zu	=	=	=
V13	zu	auf	zu 2)	zu 7)	zu	auf	zu 7)	zu
V14	zu	=	=	=	zu	=	=	=
V15	zu	auf	zu 3)	zu 8)	zu	auf	zu 8)	zu
V16	zu	=	=	=	zu	=	=	=
V17	zu	auf	zu 4)	zu 9)	zu	auf	zu 9)	zu
V18	zu	=	=	=	zu	=	=	=
V19	zu	auf	zu 5)	zu 10)	zu	auf	zu 10)	zu
V20	zu	=	=	=	zu	=	=	=
R44	ein	=	aus 1)	aus 6)	ein	=	aus 6)	aus
R21	aus	=	=	=	aus	=	=	=
R22	aus	=	=	=	aus	=	=	=
R23	ein	=	aus 2)	aus 7)	ein	=	aus 7)	aus
R25	ein	=	aus 3)	aus 8)	ein	=	aus 8)	aus
R27	ein	=	aus 4)	aus 9)	ein	=	aus 9)	aus
R29	ein	=	aus 5)	aus 10)	ein	=	aus 10)	aus
D45	bleibt	=	aus 1)	aus 6)	bleibt	=	aus	=
D24	bleibt	=	aus 2)	aus 7)	bleibt	=	aus	=
D26	bleibt	=	aus 3)	aus 8)	bleibt	=	aus	=
D28	bleibt	=	aus 4)	aus 9)	bleibt	=	aus	=
D30	bleibt	=	aus 5)	aus 10)	bleibt	=	aus	=

---

1) nur wenn nach 2 min Fkryo schlecht, sonst unverändert
2) nur wenn nach 2 min F8 schlecht, sonst unverändert
3) nur wenn nach 2 min F10 schlecht, sonst unverändert
4) nur wenn nach 2 min F12 schlecht, sonst unverändert
5) nur wenn nach 2 min F14 schlecht, sonst unverändert
6) nur wenn D45 nicht betriebsbereit, sonst unverändert
7) nur wenn D24 nicht betriebsbereit, sonst unverändert
8) nur wenn D26 nicht betriebsbereit, sonst unverändert
9) nur wenn D28 nicht betriebsbereit, sonst unverändert
10) nur wenn D30 nicht betriebsbereit, sonst unverändert

4.2.4 Task VO1 (Abschaltautomatik)

Die Service-Task VO1 (Vacuum System Off) dient zum gezielten oder abnormalen Abschalten der Vakuumanlage gemäß Tab. 3. Wegen der Dringlichkeit der Sicherheitsabschaltung liegt VO1 nicht im Overlay, sondern ist hauptspeicher-resident und hat nächst VM1 (4.2.3) die höchste Priorität. Während des Ablaufs von VO1 werden VM1 und VB1 übersprungen (MAFLA(5)).

Die Abschaltung erfolgt in folgenden Einzelschritten:

* Das zu Beginn von VO1 initialisierte INTEGER-Feld IOFF(J,K) mit J = 1:2 und K = 1:2, das die Bitmuster für die Abschaltpegel "A" enthält, in das Ausgabefeld IAUS(J,K,1) übertragen;

* Die Ist-Zustände der Diffusionspumpen, die gemäß der oben zitierten Tabelle zunächst unverändert bleiben sollen, in das gleiche Ausgabefeld IAUS(J,K,1) übertragen;

* IAUS(J,K,1) ausgeben; die Ausgabe erfolgt in 2 Schritten im Abstand von 5 sec (zuerst Ventile zu und Pumpen ein, dann Ventile auf);

* Nach Ablauf von 5 sec das ebenfalls zu Beginn von VO1 initialisierte INTEGER-Feld IOFV(J,K) in das Ausgabefeld IAUS(J,K,1) übertragen und IAUS() wie oben ausgeben;

* Die Reset-Bits im Ausgabefeld IAUS(J,K,2) derart setzen oder clearen, daß diejenigen Befehl-FF's, deren Schaltzustand nach den Sofortmaßnahmen vom Schaltzustand der zugehörigen Aggregate abweichen würden, resettet werden, die anderen nicht; dies erfolgt mittels der ohnehin vorhandenen Subroutine VU1 (4.2.2.3), die mittels
  CALL LO1(8,0,0,0)
  aufgerufen wird.

* Für jede der 5 Diffusionspumpen wird, falls sie eingeschaltet ist, eine eigene Stoppuhr gestartet und auf 60 min vorgewählt (UHR1, (4.2.5). Die Ventile zur Rotationspumpe und die Rotationspumpe selbst werden ausgeschaltet, sobald die Diffusionspumpe nicht mehr betriebsbereit ist, spätestens jedoch nach Ablauf der 60-min-Frist, siehe unten.

* Bei einer Sicherheitsabschaltung (Parameter SIAB=.TRUE.) werden nach 2 min (CALL FDELY) diejenigen Diffusionspumpen ausgeschaltet, die eingeschaltet sind und in denen der Feinvakuumdruck schlecht ist, ebenso die zugehörige Rotationspumpe und das Vorvakuumventil; ist der Feinvakuumdruck nach 2 min gut, bleiben sie eingeschaltet; bei einer gezielten Abschaltung hingegen werden nach 2 min sämtliche Diffusionspumpen ausgeschaltet;

* Bei jedem min-Interrupt wird geprüft, ob eine der 5 Stoppuhren noch aktiv und ob die Flagge MAFLA(6) noch gesetzt ist. Im letzteren Fall wird MAFLA(6) wieder genullt und VO1 wird suspendiert.

* Wenn die betreffende Diffusionspumpe nicht mehr betriebsbereit ist, spätestens jedoch nach 60 min, werden das Ventil zur Vorpumpe und die Vorpumpe selbst ausgeschaltet.

* CALL SUSP und, sobald VO1 wieder executing wird, Rücksprung an den Anfang von VO1.

Da VM1 während des Ablaufs von VO1 übersprungen wird, werden auch keine Einzelmeldungen protokolliert. Lediglich zu Beginn und am Ende von VO1 erscheinen die Nachrichten:

```
hh:mm:ss  VACUUM SHUT OFF IN PROGRESS

hh:mm:ss  VACUUM SHUT OFF FINISHED
```

VO1 hat den Task-Namen VO1, die Task-Identifikations-Nr. 2, die Priorität 20 und wird wie folgt aufgerufen:

```
COMMON /PRM/ ....  (benötigt werden A3072(), INEU(), IAUS(), IDEL(),
                                IFDEL, MIV(), MABIM(), SIAB, MAFLA())
 :
CALL RELSE(2,IER)
```

```
C
C       ************************************************************************
C       *     TASK V01        MANGER  FEBRUAR 1984                             *
C       *    TASK ZUR ABSCHALTUNG DER VAKUUMANLAGE (SICHERHEITSABSCHALTUNG)    *
C       ************************************************************************
        COMPILER NOSTACK
        TASK V01
        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
        LOGICAL LPT,STAT,IHA,SIAB
        INTEGER IOFF(2,2),IOFV(2)

        DATA IOFF/0K,0K,0K,137600K/
        DATA IOFV/105K,50000K/

C       ================    IOFF IN IAUS UEBERNEHMEN  ==( 0 SEKUNDEN )============

10      IF(SIAB) CALL LO1(7,0,15,0)
        TYPE "                          IN ORDER TO INTERRUPT THE SHUT OFF TASK
     *                                  PRESS ONE OF F1 TO F5 KEY"
        MM=0
        DO 100 K=1,2
        DO 100 J=1,2
100     IAUS(J,K,1)=IOFF(J,K)

C       ================  ISTZUSTAND DER DIFF.-PUMPEN IN IAUS UEBERTRAGEN  =====
        DO 110 I=1,5
110     IF(ITEST(INEU(2,4),I-1)) CALL ISET(IAUS(2,2,1),I-1)
120     CONTINUE
        MAFLA(5)=1
C       ================== 1.AUSGABE  , ===========================================
        DO 140 K=1,2
        AUS(K,1)=FLOINT(IAUS(1,K,1))
140     CALL CAL93(16,A3072(K,1),AUS(K,1),IQ)
        CALL FDELY(20*IFDEL)
        CALL LO1(8,0,8,0)                                   ; BFF UPDATEN

C       ================== 2.AUSGABE  , ======( 5 SEKUNDEN )=====================

        IF(MM.EQ.1) GOTO 146                                ; SCHON DURCHLAUFEN?
        CALL FDELY(30*IFDEL)
        IAUS(1,1,1)=IOFV(1)
        IAUS(2,1,1)=IOFV(2)
        MM=1
        GOTO 120
146     MM=0
        CALL FDELY(1200*IFDEL)
        IF(.NOT.SIAB) GOTO 151                              ; BEI VOFF

C       ==========BEI SIAB,  NACH 120 SEC DRUCK UEBER DIFF.-PUMPEN ABFRAGEN ===

        IF(ITEST(INEU(2,5),7)) GOTO 161                     ; F8 GUT?
        CALL ICLR(IAUS(2,2,1),0)                            ; SEKTION D24 AUS
        CALL ICLR(IAUS(2,1,1),12)
        CALL ICLR(IAUS(2,2,1),7)
161     IF(ITEST(INEU(2,5),9)) GOTO 162                     ; F10 GUT?
        CALL ICLR(IAUS(2,2,1),1)                            ; SEKTION D26 AUS
        CALL ICLR(IAUS(2,1,1),14)
        CALL ICLR(IAUS(2,2,1),8)
162     IF(ITEST(INEU(2,5),11)) GOTO 163                    ; F12 GUT?
        CALL ICLR(IAUS(2,2,1),2)                            ; SEKTION D28 AUS
        CALL ICLR(IAUS(1,1,1),0)
        CALL ICLR(IAUS(2,2,1),9)
```

```
C
163     IF(ITEST(INEU(2,5),13)) GOTO 164                    ; F14 GUT?
        CALL ICLR(IAUS(2,2,1),3)                            ; SEKTION D30 AUS
        CALL ICLR(IAUS(1,1,1),2)
        CALL ICLR(IAUS(2,2,1),10)
164     IF(ITEST(INEU(2,5),14)) GOTO 153                    ; F15 GUT?
        CALL ICLR(IAUS(2,2,1),4)                            ; SEKTION D45 AUS
        CALL ICLR(IAUS(1,1,1),6)
        CALL ICLR(IAUS(2,2,1),11)
        GOTO 153                                            ; ZUR AUSGABE

C       ========== BEI IOFF DIFF-PUMPEN AUSSCHALTEN ==========================
151     DO 152 I=1,5
152     CALL ICLR(IAUS(2,2,1),I-1)                          ; DIFF.-PUMPEN AUS

C       ===========  GEMEINSAME AUSGABE FUER VOFF UND SIAB=====================

153     DO 154 K=1,2
        AUS(K,1)=FLOINT(IAUS(1,K,1))
154     CALL CAL93(16,A3072(K,1),AUS(K,1),IQ)
        CALL FDELY(10*IFDEL)
        CALL LO1(8,0,8,0)

C       ============STOPPUHREN STARTEN WENN D AUS=============================

        DO 150 I=1,5                                        ; STOPPUHREN STARTEN
        IF(.NOT.ITEST(INEU(2,4),I-1)) MIV(I)=3
150     CONTINUE
        GOTO 200

C       ==========  JEDE MIN PRUEFEN OB EINE STOPPUHR NOCH LAEUFT  ===========

160     CALL FDELY(300*IFDEL)
        DO 170 I=1,5
        IF(MIV(I).GT.0) GOTO 200
170     CONTINUE
        GOTO 999

C       ============   FALLS D24 KALT, V13 ZU UND R23 AUS  ====================

200     IF(MAFLA(6).EQ.1) GOTO 270
        IF(ITEST(INEU(1,4),0)) GOTO 210
        MIV(1)=0
        MABIM(1)=0
        CALL ICLR(IAUS(2,2,1),0)
        CALL ICLR(IAUS(2,1,1),12)
        CALL ICLR(IAUS(2,2,1),7)

C       ====== FALLS D26 KALT,  V15 ZU UND R25 AUS  ==========================

210     IF(ITEST(INEU(1,4),1)) GOTO 220
        MIV(2)=0
        MABIM(2)=0
        CALL ICLR(IAUS(2,2,1),1)
        CALL ICLR(IAUS(2,1,1),14)
        CALL ICLR(IAUS(2,2,1),8)

C       ====== FALLS D28 KALT,  V17 ZU UND R27 AUS  ==========================

220     IF(ITEST(INEU(1,4),2)) GOTO 230
        MIV(3)=0
        MABIM(3)=0
        CALL ICLR(IAUS(2,2,1),2)
        CALL ICLR(IAUS(1,1,1),0)
        CALL ICLR(IAUS(2,2,1),9)
```

```
C
C       ====== FALLS D30 KALT,  V19 ZU UND R29 AUS  ================================
230     IF(ITEST(INEU(1,4),3)) GOTO 240
        MIV(4)=0
        MABIM(4)=0
        CALL ICLR(IAUS(2,2,1),3)
        CALL ICLR(IAUS(1,1,1),2)
        CALL ICLR(IAUS(2,2,1),10)
C       ====== FALLS D45 KALT UND V41 ZU , V43 ZU UND R44 AUS     ***********
240     IF(ITEST(INEU(1,4),4)) GOTO 250
        MIV(5)=0
        MABIM(5)=0
        CALL ICLR(IAUS(2,2,1),4)
        CALL ICLR(IAUS(1,1,1),6)
        CALL ICLR(IAUS(2,2,1),11)
C       ========================  AUSGABE  ==========================================
250     CONTINUE
        CALL ISET(IAUS(2,2,1),15)                    ; RECHNERREAKTION
        DO 260 K=1,2
        AUS(K,1)=FLOINT(IAUS(1,K,1))
260     CALL CAL93(16,A3072(K,1),AUS(K,1),IQ)
        CALL ICLR(IAUS(2,2,1),15)
        CALL FDELY(10*IFDEL)
        CALL LO1(8,0,8,0)
        CALL FDELY(10*IFDEL)
        IF(.NOT.SIAB.AND.INEU(2,4).EQ.0) GOTO 999
        GOTO 160
270     MAFLA(6)=0                                   ; ABBRUCHFLAGGE LOESCHEN
        DO 271 I=1,5                                 ; STOPPUHREN ZURUECKSETZEN
271     MIV(I)=0
999     MAFLA(5)=0                                   ; VM1 WIEDER FREIGEBEN
        MAFLA(4)=0                                   ; VB1 WIEDER FREIGEBEN
        CALL LO1(7,0,16,0)                           ; ENDE SIAB PROTOKOLLIEREN
        SIAB=.TRUE.
        CALL SUSP
        GOTO 10
        END
```

4.2.5 Task UHR1 (Mehrfachstoppuhr)

Die 20-fach-Software-Stoppuhr wird einerseits benötigt, um bei der automatischen Abschaltung der Vakuumanlage (VO1, 4.2.4) die Abkühlzeiten der 5 Diffusionspumpen von maximal 60 min einhalten zu können, und andererseits, um bei den 15 Feinvakuummeldungen "schlecht" ein 5-min-Intervall zu starten, nach dessen Ablauf die Feinvakuummeldung erst überprüft wird. UHR1 besteht im wesentlichen aus 20 unabhängigen Rückwärtszählern, die durch die Minuten-Interrupts der CAMAC-Uhr fortgeschaltet werden. Die Zeitauflösung beträgt mithin 1 min. Jeder einzelne Zähler kann, unabhängig von den anderen, über die COMMON-Argumente MIV(1:5) bzw. IDUM(1:15) vorgewählt bzw. resetted werden. Nach Ablauf der vorgewählten Zeit liefern die Software-Stoppuhren kein spezielles LAM, vielmehr muß das Programm, das diese Zeitzähler verwendet, bei jedem CAMAC-Uhren-Interrupt abfragen, ob die vorgewählte Zeit bereits abgelaufen ist.

UHR1 hat den Task-Namen UHR1, die Task-Identifikations-Nr. 5, die Priorität 50 und wird wie folgt aufgerufen:

```
COMMON /PRM/ ....  (benötigt werden MIV(), IDUM())
 :
CALL RELSE(5,IER)
```

Die COMMON-Argumente haben folgende Bedeutung:

MIV(5): : Countdown-Zähler in min für die Abkühlzeit der Diffusionspumpen;
MIV(i)=0 bedeutet: die betreffende Stoppuhr ist inaktiv

IDUM(1) bis IDUM(15): Minutenzähler für die Feinvakummmeldungen F1 bis F15.

```
C       ***************************************************************************
C       ** TASK UHR1              MANGER JANUAR 1984                              **
C       ** MEHRFACHSTOPPUHR FUER VERZOEGERTE REAKTIONEN                           **
C       ***************************************************************************

        TASK UHR1

        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

        LOGICAL LPT,STAT,IHA,SIAB
        EXTERNAL VFM1

C       ======== STOPPUHR FUER VERZOEGERTEN FEINVAKUUM-CHECK ======================

1       DO 5 I=1,15
        IV=I
        IDUM(I)=IDUM(I)-1
        IF(IDUM(I).EQ.0) CALL VFM1(IV)
5       CONTINUE

C       ======== STOPPUHREN FUER DIE 5 DIFFUSIONSPUMPEN ===========================

        DO 100 NR=1,5
        IF(MIV(NR).EQ.0) GOTO 100
        MABIM(NR)=MABIM(NR)+1
100     CONTINUE

C       ===========================================================================

        CALL SUSP
        GOTO 1
        END
```

4.2.6 Subroutine VL1 (CAMAC-Input-Modul lesen)

Die hauptspeicher-residente Subroutine VL1 liest die Bitkonfiguration des "Memory-Registers" und diejenige des "Input-Registers" eines der 5 3473-Input-Module, updatet das Memory-Register und löscht das LAM-Bit.

Als Eingabe benötigt VL1 die laufende Nr. L des Input-Moduls, die als Argument übergeben wird (Beschränkung: 1 <= L <= 5; wird von VL1 nicht geprüft) sowie die REAL-Adressen A3473(1,L) und A3473(2,L) der Submodule IA = 0 und IA = 1 des L. Input-Moduls, die über das benannte COMMON (siehe 4.2) übergeben werden. Die mit CALL CAL93 /7/ in je eine REAL-Zahl konvertierten Bitmuster des Memory- und des Input- Registers werden mit CALL IDFIX /7/ in je 2 INTEGER-Zahlen IALT(1,L),IAL(2,L) bzw. INEU(1,L),INEU(2,L) zurückverwandelt. Dabei ist die Reihenfolge der Indizes wichtig, da beim Data-General-FORTRAN, im Gegensatz zum IBM-FORTRAN, mehrdimensionale Felder so gespeichert werden, daß sich die Indizes in der Reihenfolge von links nach rechts ändern.

Aufruf:

```
COMMON /PRM/ ....  (benötigt werden A3473(), IALT(), INEU())
 :
CALL VL1(L)
```

```
C     ********************************************************************
C     ** SUBROUTINE VL1                                                 **
C     ** UNTERPROGRAMM ZUM EINLESEN EINES DER 3473-INPUTREGISTER        **
C     ** MANGER FEBRUAR 1984                                            **
C     ********************************************************************

      SUBROUTINE VL1(L)

      COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
      LOGICAL LPT,STAT,IHA,SIAB

C     ========================= MEMORY-REGISTERS LESEN ===================================

      CALL CAL93(0,A3473(2,L),R,IQ)
      CALL IDFIX(IALT(1,L),R)

C     ====== AKTUELLE EINGANGSPEGEL LESEN,MEMORY UPDATEN,LAM LOESCHEN =========

      CALL CAL93(2,A3473(1,L),R,IQ)
      CALL IDFIX(INEU(1,L),R)

C     ====================================================================================

      RETURN

      END
```

4.2.7 Subroutine LO1 (Overlay laden)

Die hauptspeicher-residente Subroutine LO1 lädt eine Overlay-Subroutine aus einem Overlay-Segment des auf Platte liegenden Files "LJ2.OL" in den zugehörigen, von RLDR (relocatable loader des NOVA-Betriebssystems) angelegten Overlay-Bereich des Hauptspeichers, siehe Abb. 2. Der Overlay-File muß bereits im Oberprogramm geöffnet worden sein. Die Channel-Nr. wurde in LJ2 (4.2) willkürlich zu 2 festgesetzt; sie ist keine globale Variable. Das Laden (CALL OVLOD) erfolgt "conditional" (3. Argument von OVLOD # -1). d.h., das Laden von Platte unterbleibt, wenn das zu ladende Overlay bereits geladen ist, sodaß unnötige Plattenzugriffe vermieden werden. Falls geladen werden muß und der Overlay-Bereich vorher nicht freigemacht worden wäre, würde die aufrufende Task suspendiert werden. Da dieser Mechanismus erfahrungsgemäß nicht immer funktioniert, werden das Laden und die Freigabe mit Overlay-Flaggen MAFLA(1) bzw. MAFLA(2) gesteuert, siehe unten.

LO1 wird aufgerufen:

```
CALL LO1(ION,IA1,IA2,IA3)
```

Das 1. Argument ION ist ein INTEGER-Code für den (im OVERLAY-Statement vergebenen) Overlay-Namen. ION darf Werte zwischen 1 und 10 annehmen (diese Beschränkung wird von LO1 nicht geprüft):

ION	= 1	entspricht	dem Overlay-Namen	OTTS1
	2	"	"	OVBS1
	3	"	"	OVBS2
	4	"	"	OVBS3
	5	"	"	OVST1
	6	(nicht benutzt)		
	7	"	"	OVP1
	8	"	"	OVU1
	9	"	"	OREWR
	10	"	"	OEX1

Die übrigen 3 Argumente dienen zur Übergabe von Argumenten an die Overlay-Subroutinen.

LO1 gliedert sich in folgende Programmschritte:

* Abfrage, in welchen Overlay-Bereich geladen werden soll.

* Abfrage, ob der betreffende Bereich zur Benutzung frei ist (MAFLA(1) für Bereich 0, MAFLA(2) für Bereich 1). Ist der Bereich noch nicht freigegeben (Flagge=1), wird das rufende Programm für 0.1 Sekunde suspendiert (CALL FDELY), um im Anschluß daran die Flagge erneut abzufragen. Dieser Versuch wird beliebig oft wiederholt; bei Nichterfolg versackt LO1 ohne Fehlernachricht.

* Ist der Bereich frei, wird die über das Argument ION angewählte

Overlay-Subroutine geladen. Vorher wird die für den Bereich zuständige Flagge gleich 1 gesetzt.

* Nach Beendigung des Overlay-Programms wird die Flagge wieder gecleart, um den Bereich wieder zur Benutzung freizugeben.

In LO1 sind die Fehlermeldungen sowie die Reaktionen auf Fehler explizit programmiert, da die Subroutine EX1 (4.2.2.4) selbst im Overlay liegt, d.h. gegebenenfalls von LO1 zu laden wäre.

```
C
C     ***************************************************************
C     **     SUBROUTINE ZUM LADEN DER OVERLAY-FILES                * *
C     **     AUSWAHL DER OVERLAYS UEBER ARGUMENTE                  * *
C     *      MANGER Januar 1984                                     **
C     ***************************************************************


      SUBROUTINE LO1(ION,IBU,IARG,IAG)
      COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
      LOGICAL LPT,STAT,IHA,SIAB
      EXTERNAL OTT1,TTS1,OEX1,OVB1,VBS1,OVP1,VP1,EX1,VU1,OVU1,VST1,
     *  OVST1,OVB2,VBS2,OVB3,VBS3,OREWR,REWR

      IF(ION.GE.7) GOTO 2                          ;OVERLAYFLAGGEN ABFRAGEN
1     IF(MAFLA(1).EQ.0) GOTO 5
      CALL FDELY(1*IFDEL)
      GOTO 1
2     IF(MAFLA(2).EQ.0) GOTO 5
      CALL FDELY(1*IFDEL)
      GOTO 2
5     GOTO(10,20,30,40,50,60,70,80,90,100),ION   ; OVERLAY LADEN

C     ====================    OVERLAYS LADEN ================================
10    MAFLA(1)=1
      CALL OVLOD(2,OTT1,1,IER)
      IF(IER.NE.1) GOTO 150
      CALL TTS1
      GOTO 140
20    MAFLA(1)=1
      CALL OVLOD(2,OVB1,1,IER)
      IF(IER.NE.1) GOTO 151
      CALL VBS1(IAG)
      GOTO 140
30    MAFLA(1)=1
      CALL OVLOD(2,OVB2,1,IER)
      IF(IER.NE.1) GOTO 152
      CALL VBS2(IAG)
      GOTO 140
40    MAFLA(1)=1
      CALL OVLOD(2,OVB3,1,IER)
      IF(IER.NE.1) GOTO 153
      CALL VBS3(IAG)
      GOTO 140
50    MAFLA(1)=1
      CALL OVLOD(2,OVST1,1,IER)
      IF(IER.NE.1) GOTO 154
      CALL VST1
      GOTO 140
60    MAFLA(1)=1                                   ; INZWISCHEN DUMMY
      GOTO 140
70    MAFLA(2)=1
      CALL OVLOD(2,OVP1,1,IER)
      IF(IER.NE.1) GOTO 155
      CALL VP1(IARG,IBU,IAG)
      GOTO 141
80    MAFLA(2)=1
      CALL OVLOD(2,OVU1,1,IER)
      IF(IER.NE.1) GOTO 156
      CALL VU1(IARG)
      GOTO 141
```

```
C
90      MAFLA(2)=1
        CALL OVLOD(2,OREWR,1,IER)
        IF(IER.NE.1) GOTO 157
        CALL REWR(IAG)
        GOTO 141
100     MAFLA(2)=1
        CALL OVLOD(2,OEX1,1,IER)
        IF(IER.NE.1) GOTO 158
        CALL EX1(IARG)
        GOTO 141
140     MAFLA(1)=0
        RETURN
141     MAFLA(2)=0
        RETURN
C       ========================    FEHLERAUSGANG   ====================================

150     TYPE "*** LO1 : OVERLAY SUBROUTINE TTS1 NOT LOADED"
        GOTO 160
151     TYPE "*** LO1 : OVERLAY SUBROUTINE VBS1 NOT LOADED"
        GOTO 160
152     TYPE "*** LO1 : OVERLAY SUBROUTINE VBS2 NOT LOADED"
        GOTO 160
153     TYPE "*** LO1 : OVERLAY SUBROUTINE VBS3 NOT LOADED"
        GOTO 160
154     TYPE "*** LO1 : OVERLAY SUBROUTINE VST1 NOT LOADED"
        GOTO 160
155     TYPE "*** LO1 : OVERLAY SUBROUTINE VF1  NOT LOADED"
        GOTO 160
156     TYPE "*** LO1 : OVERLAY SUBROUTINE VU1  NOT LOADED"
        GOTO 160
157     TYPE "*** LO1 : OVERLAY SUBROUTINE REWR NOT LOADED"
        GOTO 160
158     TYPE "*** LO1 : OVERLAY SUBROUTINE EX1 NOT LOADED"

C       ===== BEI LOAD-ERROR KANN PROGRAMM NICHT UEBER EX1 GESTOPPT WERDEN ========

160     TYPE "***       FATAL ERROR, PROGRAM STOPPED"
        STOP

        END
```

4.2.8 Subroutine VP1 (Protokollzeile ausgeben)

VP1 gibt pro Aufruf auf dem Terminalschirm und wahlweise (abhängig von dem über COMMON übergebenen globalen Programmparameter LPT) auf dem Drucker ($LPT) 1 Protokollzeile aus, die aus der Uhrzeit in der Form hh:mm:ss und einem (englischen) Text besteht, dessen Format durch 3 Argumente bestimmt wird.

VP1 hat den Overlay-Namen "OVP1", die Identifikations-Nr.ION=7 (siehe 4.2.7) und liegt zusammen mit VU1 (4.2.2.3), REWR (4.2.9) und EX1 4.2.2.4) im Segment # 1 des Files "LJ2.OL" (Abb. 2). Das Laden und der Aufruf von VP1 erfolgt mittels der Subroutine LO1 (4.2.7):

```
COMMON /PRM/ ....  (benötigt werden LPT, MAPRO())
 :
CALL LO1(7,IA2,IA3,IA4)
```

Die Argumente haben folgende Bedeutung:

IA2: Auswahl des Protokolltextes
IA3: Index eines Feldelements von MAPRO() zur Ausgabe von 'V', 'R', 'D', 'T', 'F', 'H'
IA4: Index eines Feldelements von MAPRO() zur Ausgabe der Aggregatnummer

4.2.9 Subroutine REWR (Lesen von und Schreiben auf Platte)

REWR liest das Plattenfile "VPA.DA" (4.1) in den COMMON-Bereich /PRM/ (4.2) oder umgekehrt. REWR besitzt ein explizites Argument zur Auswahl der jeweiligen Aktion. Der Ablauf ist: Öffnen des Files "VPA.DA", Lesen oder Schreiben, File Schließen, Return. Bei Lese-/Schreib-Fehlern wird der Versuch nicht wiederholt.

REWR hat den Overlay-Namen OREWR, die Identifikations-Nr. ION=9 (siehe 4.2.7) und liegt im Segment # 1 des Plattenfiles "LJ2.OL". Das Laden und der Aufruf erfolgt mittels der Subroutine LO1 (4.2.7):

```
COMMON /PRM/ ....  (benötigt wird der gesamte COMMON-Bereich)
 :
CALL LO1(9,0,0,IA4)
```

```
C
C       ********************************************************************
C       **  OVERLAY VF1                                                   **
C       **  UNTERPROGRAMM ZUM  AUSDRUCK EINER PROTOKOLLZEILE              **
C       **  MANGER FEBRUAR 1984                                           **
C       ********************************************************************
        COMPILER NOSTACK
        OVERLAY OVF1
        SUBROUTINE VF1(IPR,IBU,IAG)

        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LFT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

        LOGICAL LFT,STAT,IHA,SIAB
        INTEGER IZEIT(3)

C       ========== UHRZEIT PROTOKOLLIEREN ==========

        DO 900 I=10,12,2
        CALL TIME(IZEIT,IER)
        WRITE(I,1000)IZEIT(1),IZEIT(2),IZEIT(3)
1000    FORMAT(" ",2(I2,":"),I2,"   ",Z)

C       =======  PROTOKOLLZEILE ANWAEHLEN  ==========

        GOTO (110,120,130,140,150,160,170,180,190,200,210,220,230,240,
     *  250,260),IPR
C       ====================================================================

110     WRITE(I,1100)
1100    FORMAT("              COMPUTER CONTROL, INITIAL CHECK UP FOLLOWS")
        GOTO 899
C       ====================================================================

120     WRITE(I,1200)
1200    FORMAT("              MANUAL CONTROL")
        GOTO 899
C       ====================================================================

130     WRITE(I,1300)
1300    FORMAT("        <7>      SHUT-OFF PROCEDURE IN PROGRESS <3><16>!!<17>")
        GOTO 899
C       ====================================================================

140     WRITE(I,1400)
1400    FORMAT(" ACTUAL STATE OF THE VACUUM-SYSTEM :")
        GOTO 899
C       ====================================================================

150     WRITE(I,1500)
1500    FORMAT("              END OF PROGRAM")
        GOTO 899
C       ====================================================================

160     WRITE(I,1600) MAPRO(IBU),MAPRO(IAG)
1600    FORMAT(" ",11X,A2,I2," =   1    ")
        GOTO 899
C       ====================================================================

170     WRITE(I,1700) MAPRO(IBU),MAPRO(IAG)
1700    FORMAT(" ",11X,A2,I2," =   0")
        GOTO 899
C       ====================================================================
```

```
C
180     WRITE(I,1800) MAPRO(IBU),MAPRO(IAG)
1800    FORMAT(" ","B ",A2,I2," : 1")
        GOTO 899
C       ================================================================================
190     WRITE(I,1900) MAPRO(IBU),MAPRO(IAG)
1900    FORMAT(" ","B ",A2,I2," : 0")
        GOTO 899
C       ================================================================================
200     WRITE(I,2000) MAPRO(IBU),MAPRO(IAG)
2000    FORMAT(" ","B ",A2,I2," : 1  PROHIBITED")
        GOTO 899
C       ================================================================================
210     WRITE(I,2100) MAPRO(IBU),MAPRO(IAG)
2100    FORMAT(" ","B ",A2,I2," : 0  PROHIBITED")
        GOTO 899
C       ================================================================================
220     WRITE(I,2200)
2200    FORMAT("              LJ2 LOADED")
        GOTO 899
C       ================================================================================
230     WRITE(I,2300)
2300    FORMAT("        <7>      NOT DONE! CHECK THE VALVE")
        GOTO 899
C       ================================================================================
240     WRITE(I,2400)
2400    FORMAT("        <7>      NOT DONE! CHECK THE PUMP")
        GOTO 899
C       ================================================================================
250     WRITE(I,2500)
2500    FORMAT("              SHUT-OFF PROCEDURE IN PROGRESS <3><16>!!<17><7>")
        GOTO 899
C       ================================================================================
260     WRITE(I,2600)
2600    FORMAT("              <7>SHUT-OFF PROCEDURE FINISHED")
C       ================================================================================
899     IF(.NOT.LPT) GOTO 910                    ;BEI LPT PROT AUF TT UND LPT
900     CONTINUE
910     RETURN
        END
```

```
C
C     ***************************************************************************
C     *  OVERLAY REWR                                                           *
C     *  SUBROUTINE ZUM LESEN UND SCHREIBEN DER PARAMETER VON/AUF               *
C     *  PLATTENFILE VPA.DA                                                     *
C     *  MANGER JANUAR 1984                                                     *
C     ***************************************************************************

      OVERLAY OREWR
      SUBROUTINE REWR(IAG)

      COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)

      LOGICAL LPT,STAT,IHA,SIAB

C     =============================================================================

      CALL OPEN(1,"VPA.DA",2,IER)                          ;DATENFILE OEFFNEN
      IF(IER.EQ.1) GOTO 10
      TYPE "*** REWR : CHANNEL 1 NOT OPENED<7>",IER
      STOP
10    GOTO(20,30),IAG                                      ;SCHREIBEN ODER LESEN?

C     =============================================================================

20    CALL WRBLK(1,1,MASKE(1,1),1,IER)                     ;DATEN SCHREIBEN
      IF(IER.EQ.1) GOTO 40
      TYPE "*** REWR : WRITE-ERROR ,FILE VPA.DA<7>",IER
      STOP

C     =============================================================================

30    CALL RDBLK(1,1,MASKE(1,1),1,IER)                     ;DATEN LESEN
      IF(IER.EQ.1) GOTO 40
      TYPE "*** REWR : READ-ERROR ,FILE VPA.DA<7>",IER
      STOP

C     =============================================================================

40    CALL CLOSE(1,IER)                                    ;KANAL SCHLIESSEN
      IF(IER.EQ.1) RETURN
      TYPE "*** REWR : CHANNEL 1 NOT CLOSED<7>",IER
      STOP

      END
```

Tab. 4: Zulässige Zustände bei der Anfangsprüfung

Funkt.einh.:	1	2	3	4	5	6	7	8	9	10	11
V1	zu	auf	zu	auf	zu	auf	zu	auf	Abgetrennte Sektionen		
V2	zu	zu	auf	auf	zu	zu	auf	auf			
V3a,b	zu	zu	zu	zu	auf	auf	auf	auf	Kryof.	T.schl.	Sp.kamm
Abgetrennt von allen Subsektionen	V9 zu V10 " V14 " *	V9 zu V10 " V14 " * V41 " V42 " *	V6 zu V7 " V8 a|z R21 e|a V9 zu V10 " V14 " *	V6 zu V7 " V8 a|z R21 e|a V9 zu V10 " V14 " * V41 " V42 " *	V9 zu V10 " V11 " V12 " V14 " * V16 " * V18 " * V20 " * R22 aus	V9 zu V10 " V11 " V12 " V14 " * V16 " * V18 " * V20 " * V41 " V42 " * R22 aus	V6 zu V7 " V8 a|z R21 e|a V9 zu V10 " V11 " V12 " V14 " * V16 " * V18 " * V20 " * R22 aus	V6 zu V7 " V8 a|z R21 e|a V9 zu V10 " V11 " V12 " V14 " * V16 " * V18 " * V20 " * V41 " V42 " * R22 aus	V41 zu V42 " *	V6 zu V7 " V8 a|z R21 e|a	V11 zu V12 " V16 " * V18 " * V20 " * @
Belüften	V9 auf V10 zu V14 " *	V9 auf V10 zu V14 " * V41 zu V42 " *	V6 zu V7 " V8 a|z R21 e|a V9 auf V10 zu V14 " * V41 " V42 " *	V6 zu V7 " V8 a|z R21 e|a V9 auf V10 zu V14 " * V41 " V42 " *	V9 a|z|a V12 z|a|a V10 zu V11 " V14 " * V16 " * V18 " * V20 " * R22 aus	V9 a|z|a V12 z|a|a V10 zu V11 " V14 " * V16 " * V18 " * V20 " * V41 " V42 " * R22 aus	V6 zu V7 " V8 " V9 a|z|a V12 z|a|a V10 zu V11 " V14 " * V16 " * V18 " * V20 " * R21 aus R22 "	V6 zu V7 " V8 a|z R21 e|a V9 a|z|a V12 z|a|a V10 zu V11 " V14 " * V16 " * V18 " * V20 " * V41 " V42 " * R22 aus	-	V6 zu V7 " V8 a|z R21 e|a	V11 zu V12 auf V16 zu * V18 " * V20 " * @
Vorevakuieren	V9 zu V10 auf V11 zu V14 " * R22 ein	-	V6 zu V7 " V9 " V10 auf V11 zu V14 " * R22 ein	-	-	-	-	-	V41 auf V42 zu * R44 ein	V6 auf V7 zu V8 " R21 ein	V11 auf V12 zu V16 " * V18 " * V20 " * @
Hochvakuumpumpen	V9 zu V10 " V13 auf V14 " R23 ein D24 " T24 bb F8 gut	V9 zu V10 " V13 auf V14 " V41 zu V42 auf V43 " R23 ein R44 " D24 " D45 " T24 bb T45 " F8 gut F15 "	V6 zu V7 " V8 a|z R21 e|a V9 zu V10 " V13 auf V14 " R23 ein D24 " T24 bb F8 gut	V6 zu V7 " V8 a|z R21 e|a V9 zu V10 " V13 auf V14 " V41 zu V42 auf V43 " R23 ein R44 " D24 " D45 " T24 bb T45 " F8 gut F15 "	V9 zu V10 " V11 " V12 " V13 auf V14 " V15 " V16 " V17 " V18 " V19 " V20 " R22 aus R23 ein R25 " R27 " R29 " D24 " D26 " D28 " D30 " T24 bb T26 " T28 " T30 " F8 gut F10 " F12 " F14 "	V9 zu V10 " V11 " V12 " V13 auf V14 " V15 " V16 " V17 " V18 " V19 " V20 " V41 zu V42 auf V43 " R22 aus R23 ein R25 " R27 " R29 " R44 " D24 " D26 " D28 " D30 " D45 " T24 bb T26 " T28 " T30 " T45 " F8 gut F10 " F12 " F14 " F15 "	V6 zu V7 " V8 a|z R21 e|a V9 zu V10 " V11 " V12 " V13 auf V14 " V15 " V16 " V17 " V18 " V19 " V20 " V41 zu V42 auf V43 " R22 aus R23 ein R25 " R27 " R29 " R44 " D24 " D26 " D28 " D30 " D45 " T24 bb T26 " T28 " T30 " F8 gut F10 " F12 " F14 " F15 "	V6 zu V7 " V8 a|z R21 e|a V9 zu V10 " V11 " V12 " V13 auf V14 " V15 " V16 " V17 " V18 " V19 " V20 " V41 zu V42 auf V43 " R22 aus R23 ein R25 " R27 " R29 " R44 " D24 " D26 " D28 " D30 " D45 " T24 bb T26 " T28 " T30 " T45 " F8 gut F10 " F12 " F14 " F15 "	V41 zu V42 auf V43 " R44 ein D45 " T45 bb F15 gut	-	V11 zu V12 " V15 auf V16 " V17 " V18 " V19 " V20 " R25 ein R27 " R29 " D26 " D28 " D30 " T26 bb T28 " T30 " F10 gut F12 " F14 " @
Überprüf. abgetr. Sektionen	Sp. 9 Sp. 10 Sp. 11	Sp. 10 Sp. 11	Sp. 9 Sp. 11	Sp. 11	Sp. 9 Sp. 10	Sp. 10	Sp. 9	-			

*) Überprüfung abgetrennter Diffusionspumpen:

V42 zu	V14 zu	V16 zu	V18 zu	V20 zu
V43 z|a	V13 z|a	V15 z|a	V17 z|a	V19 z|a
R44 a|e	R23 a|e	R25 a|e	R27 a|e	R29 a|e
D45 a|e	D24 a|e	D25 a|e	D28 a|e	D30 a|e

@) Überprüfung von R22:
"R22 ein" nur erlaubt, wenn V10 oder (exklusiv) V11 auf, sonst Sicherheitsabschaltung

4.3 MAIN-Task VA1 (Anfangsprüfung)

VA1 führt nach dem Laden von LJ2, falls die aktive Bedientafel auf RECHNER-Betrieb geschaltet ist, sowie nach dem Umschalten von HAND- auf RECHNER-Betrieb, falls LJ2 geladen ist, die Anfangsprüfung durch, d.h. VA1 prüft, ob der vorgefundene Ist-Zustand der Vakuumanlage (Meldungen "M" und "F__") einem der in Tab. 4 definierten, in VA1 in den Feldern

IGL1() bis IGL8()	(Gesamtpr. Liste f. d. Funktionseinh.	# 1bis 8)
IGM1() bis IGM8()	(Gesamtpr. Maske " "	# 1 bis 8)
IKRL() und IKRM()	(Liste und Maske " "	# 9)
ITSL() und ITSM()	(" " " " "	# 10)
ISPL() und ISPM()	(" " " " "	# 11)

gespeicherten und initialisierten zulässigen Zustände entspricht, siehe unten.

Tab. 4 ist folgendermaßen zu verstehen:

Jenachdem, welche der Ventile V1; V2; V3a,b geöffnet bzw. geschlossen sind, ist die Zentralsektion "Targetkammer" mit keiner, mit einer oder mit mehreren ihrer Nachbarsektionen "Kryofalle", "Targetschleuse", "Spektrographenkammer" verbunden. Diese 8 Hauptfälle sind in den Spalten 1 bis 8 von Tab. 4 behandelt. Die von der Zentralsektion abgetrennten Hauptsektionen, an die in der unteren Zeile von Tab. 4 erinnert wird, werden in den Spalten 9 bis 11 betrachtet. Wenn z.B. V1 und V3a,b geöffnet sind und V2 geschlossen ist (Spalte 6), gibt es akut 2 selbständige Funktionseinheiten, nämlich die Kombination "Targetkammer + Kryofalle + Spektrographenkammer" sowie die abgetrennte Sektion "Targetschleuse".

Für jede selbständige Funktionseinheit sind in Tab. 4 diejenigen Zustände der einzelnen Aggregate angegeben, die den folgenden 4 als sinnvoll und als einzig zulässig erachteten Betriebsarten entsprechen:

A) Abgetrennt von allen Subsektionen (Atmosphäre, Vorvakuumpumpe, Hochvakuumpumpe),
B) Belüften,
C) Vorevakuieren,
D) Hochvakuumpumpen.

Zu Beginn des Rechnerbetriebs prüft das Programm jede einzelne Funktionseinheit daraufhin, ob die vorgefundenen Anfangszustände einer dieser 4 zulässigen Betriebsarten entsprechen; ausgenommen von dieser Prüfung sind irrelevante Zustände (sie sind in Tab. 4 nicht aufgeführt und in VA1 durch IGM1() bis IGM8(), IKRM(), ITSM() bzw. ISPM() gekennzeichnet) sowie eventuell nichtexistierende Aggregate (sie sind durch MASKE(), siehe 4.1, gekennzeichnet). Falls nur eine der Funktionseinheiten einen unzulässigen Anfangszustand aufweist, wird als Standardmaßnahme die Sicherheitsabschaltung (VO1, 4.2.4) eingeleitet.

Auf folgende Besonderheiten sei eigens hingewiesen:

a) Wenn eine zu einer Funktionseinheit gehörende Subsektion selbst wieder aus mehreren Untersektionen besteht, z.B. R21 mit V6 und V8 oder die Diffusionspumpen mit ihren Rotationspumpen, dann gibt es für die betreffende Funktionseinheit u.U. mehrere zulässige Betriebszustände. Diese sind teils im Hauptteil von Tab. 4 enthalten, teils (für abgetrennte Diffusionspumpen) in einer Nebentabelle angegeben.

b) Die Rotationspumpe R22, die die Abschnitte Targetkammer und Spektrographenkammer wahlweise, jedoch nocht gleichzeitig bedienen soll, wird gesondert anhand der Ventilstellungen V10 und V11 geprüft.

c) Für Funktionsheiten, an die mehr als 1 Diffusionspumpe angeschlossen sind, (Spalten # 3, 4, 5, 6, 7, 8), und für die Sektion "Spektrographenkammer" (Spalte 11), die wahlweise mit 2 Diffusionspumpen (D26, D28) oder mit 3 Diffusionspumpen (zusätzlich D30) ausgerüstet sein wird, gilt in der Betriebsart "Hochvakuumpumpen", daß alle vorhandenen Diffusionspumpen in Betrieb und die zugehörigen Ventile geöffnet sein müssen.

VA1 ist ein eigenständiges Hauptprogramm, das in den oben genannten Fällen von LJ2 (4.2) geladen und zur Ausführung gebracht wird (CALL SWAP). VA1 übernimmt von LJ2 über das Platten-File "VPA.DA" alle Anlagen- und Programm-Parameter, von denen allerdings nur INEU(J,L), IAUS(J,K,L), MAFLA(3) (Anfangsprüfung Flagge) und MASKE() benötigt werden.

Zu Beginn von VA1 werden die Prüflisten I_L_() und die Masken I_M_() gemäß Tab. 4 initialisiert, u.z. nach folgendem Schema:
Für jede Funktionseinheit (Spalte 1 bis 11 von Tab. 4) sind 2 eigene Prüffelder I_L_() und I_M_() angelegt. Z.B. ist für die Funktionseinheit 1 (V1 zu, V2 zu, V3 zu) die Prüfliste im Feld IGL1(5,4) initialisiert; die zugehörige Ausblendmaske, mit deren Hilfe nichtrelevante Zustände von der Prüfung ausgenommen werden, ist entsprechend in IGM1(5,4) festgelegt. Die Dimensionen dieser Felder haben folgende Bedeutung: 5 bezieht sich auf die 5 INTEGER-Worte, die die Bitmuster der zulässigen Anfangszustände enthalten, wie sie für den Vergleich mit den 5 die Istzustände enthaltenden INTEGER-Worten INEU(1,3), INEU(2,3), INEU(1,4), INEU(2,4) und INEU(2,5), vgl. Tab. 1, Seite 3 sowie bezüglich der Indexbedeutungen 4.2, Seite 14, benötigt werden. Die Dimension 4 gibt an, daß es für die betrachtete Funktionseinheit 4 verschiedene zulässige Anfangszustände gibt.

VA1 gliedert sich in folgende Schritte:

* Lesen der Parameter vom Platten-File "VPA.DA"

* Entscheidung, um welche der 8 Hauptfälle (Spalte # 1 bis 8) es sich handelt, und Verzweigung zu dem betreffenden Prüfabschnitt. Diese 8 Prüfabschnitte sind in ihrem Ablauf identisch und unterscheiden sich nur durch die verwendeten Felder IGL1() bis IGL8() und IGM1() bis IGM8().

  Die Zulässigkeitsprüfung erfolgt in 3 geschachtelten DO-Schleifen: In der inneren DO-Schleife, die den Einzelbit-Vergleich der Feldele-

mente von INEU() und IGL_() durchführt, wird die Prüfung übersprungen, wenn entweder das betreffende Aggregat nicht existiert oder wenn sein Zustand irrelevant ist. Die mittlere DO-Schleife läuft über die jeweils 5 zu vergleichenden Worte (1. Dimension der Felder); sobald nur ein einziges Bit von INEU() nicht mit dem entsprechenden Bit von IGL_() übereinstimmt, wird der Laufindex der äußeren DO-Schleife, die die verschiedenen zulässigen Zustandskombinationen (2. Dimension der Felder) durchläuft, um 1 erhöht. Wenn aber die innere und die mittlere DO-Schleife vollständig durchlaufen sind, ohne daß ein Bitunterschied festgestellt wurde, wird ein Zeiger, der auf die Prüfabschnitte für die abgetrennten Sektionen hinweist, gesetzt; diese Prüfungen und gegebenenfalls die Prüfung für R22 werden in analoger Weise durchgeführt. Wenn die äußere DO-Schleife vollständig durchlaufen ist, so bedeutet das, daß der vorgefundene Istzustand mit keiner der zulässigen Zustandskombinationen übereinstimmt.

* Bei z u l ä s s i g e m Prüfergebnis wird die Protokollzeile

INITIAL STATE PERMISSIBLE

ausgegeben und die Flagge MAFLA(3) zur Übergabe des Prüfergebnisses an LJ2 wird gesetzt;

Bei u n z u l ä s s i g e m Prüfergebnis wird die Protokollzeile

INITIAL STATE PROHIBITED

ausgegeben, die Flagge MAFLA(3) wird gecleart und die LOGICAL-Flagge SIAB gleich .TRUE. gesetzt.

In beiden Fällen werden das READY-Bit in IAUS(2,1,1) gesetzt, der COMMON-Bereich /PRM/ wird in das Platten-File "VPA.DA" geschrieben, und VA1 kehrt mittels CALL BACK zu LJ2 zurück. Aufgrund MAFLA(3)=0 wird von LJ2 die Sicherheitsabschaltung (4.2.4) eingeleitet.

```
C
C       ************************************************************************
C       *    MAIN VA1          (ANFANGSPRUEFUNG VAKUUMANLAGE)                  *
C       *    PROGRAMM WIRD VON LJ2  GE- SWAPPED        MANGER JANUAR 1984      *
C       ************************************************************************

      COMPILER NOSTACK
      COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAFRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
      LOGICAL LPT,STAT,IHA,SIAB,IV1,IV2,IV3

      DIMENSION IGM1(5,4),IGL1(5,4),IGM2(5,3),IGL2(5,3),IGM3(5,7),IGL3(5,7),
     *  IGM4(5,6),IGL4(5,6),IGM5(5,5),IGL5(5,5),IGM6(5,5),IGL6(5,5),
     *  IGM7(5,7),IGL7(5,7),IGM8(5,7),IGL8(5,7),IKRM(5,3),IKRL(5,3),
     *  ITSM(5,5),ITSL(5,5),ISPM(5,4),ISPL(5,4),IN(5),MA(5)

C       ---------- INITIALISIERUNG DER PRUEFLISTEN ----------------------------

      DATA IGL1/5*0K,0K,400K,0K,0K,0K,0K,1000K,100K,0K,0K,0K,30000K,201K,
     *          200K,1K/
      DATA IGM1/0K,21400K,0K,0K,0K,0K,21400K,0K,0K,0K,0K,23400K,100K,
     *          0K,0K,0K,31400K,201K,200K,1K/

      DATA IGL2/5*0K,0K,400K,0K,0K,0K,140K,30000K,4221K,40200K,21K/
      DATA IGM2/60K,21400K,0K,0K,0K,60K,21400K,0K,0K,0K,160K,31400K,
     *          4221K,40200K,21K/

      DATA IGL3/5*0K,0K,200K,40K,0K,0K,0K,400K,0K,0K,0K,0K,600K,40K,0K,0K,
     *          0K,1000K,100K,0K,0K,0K,30200K,241K,200K,1K,
     *          0K,30000K,201K,200K,1K/
      DATA IGM3/0K,21740K,40K,0K,0K,0K,21740K,40K,0K,0K,60K,21740K,
     *          40K,0K,0K,60K,21740K,40K,0K,0K,0K,23540K,100K,0K,0K,
     *          0K,31740K,241K,200K,1K,0K,31740K,241K,200K,1K/

      DATA IGL4/5*0K,0K,200K,40K,0K,0K,0K,400K,0K,0K,0K,0K,600K,40K,
     *          0K,0K,140K,30000K,4221K,40200K,21K,140K,30200K,4261K,
     *          40200K,21K/
      DATA IGM4/60K,21740K,40K,0K,0K,60K,21740K,40K,0K,0K,60K,21740K,
     *          40K,0K,0K,60K,21740K,40K,0K,0K,160K,31740K,4261K,
     *          40200K,21K,160K,31740K,4261K,40200K,21K/

      DATA IGL5/5*0K,0K,400K,0K,0K,0K,0K,4000K,0K,0K,0K,0K,4400K,0K,
     *          0K,0K,17K,170000K,3617K,25200K,17K/
      DATA IGM5/12K,127400K,100K,0K,0K,12K,127400K,100K,0K,0K,12K,
     *          127400K,100K,0K,0K,12K,127400K,100K,0K,0K,17K,177400K,
     *          3717K,25200K,17K/

      DATA IGL6/5*0K,0K,400K,0K,0K,0K,0K,4000K,0K,0K,0K,0K,4400K,0K,
     *          0K,0K,157K,170000K,7637K,65200K,37K/
      DATA IGM6/72K,127400K,100K,0K,0K,72K,127400K,100K,0K,0K,72K,
     *          127400K,100K,0K,0K,72K,127400K,100K,0K,0K,177K,
     *          177400K,7737K,65200K,37K/

      DATA IGL7/5*0K,0K,200K,40K,0K,0K,0K,400K,0K,0K,0K,0K,4000K,
     *          0K,0K,0K,0K,4400K,0K,0K,0K,17K,170000K,3617K,25200K,
     *          17K,17K,170200K,3657K,25200K,17K/
      DATA IGM7/12K,127740K,140K,0K,0K,12K,127740K,140K,0K,0K,12K,
     *          127740K,140K,0K,0K,12K,127740K,140K,0K,0K,12K,
     *          127740K,140K,0K,0K,17K,177740K,3757K,25200K,17K,17K,
     *          177740K,3757K,25200K,17K/
```

```
C
      DATA IGL8/5*0K,0K,200K,40K,0K,0K,0K,400K,0K,0K,0K,0K,4000K,
     *         0K,0K,0K,0K,4400K,0K,0K,0K,157K,170000K,7637K,
     *         65200K,37K,157K,170200K,7677K,65200K,37K/
      DATA IGM8/72K,127740K,140K,0K,0K,72K,127740K,140K,0K,0K,72K,
     *         127740K,140K,0K,0K,72K,127740K,140K,0K,0K,72K,
     *         127740K,140K,0K,0K,177K,177740K,7777K,65200K,37K,
     *         177K,177740K,7777K,65200K,37K/

      DATA IKRL/5*0K,20K,0K,4000K,0K,0K,140K,0K,4020K,40000K,20K/
      DATA IKRM/60K,4*0K,60K,0K,4000K,0K,0K,160K,0K,4020K,40000K,20K/

      DATA ITSL/5*0K,0K,200K,40K,0K,0K,5*0K,0K,200K,40K,0K,0K,
     *         0K,40K,40K,0K,0K/
      DATA ITSM/0K,340K,40K,0K,0K,0K,340K,40K,0K,0K,0K,340K,40K,0K,0K,
     *         0K,340K,40K,0K,0K,0K,340K,40K,0K,0K/

      DATA ISPL/5*0K,0K,4000K,0K,0K,0K,0K,2000K,100K,0K,0K,17K,
     *         140000K,3416K,25000K,16K/
      DATA ISPM/12K,106000K,0K,0K,0K,12K,106000K,0K,0K,0K,12K,
     *         106000K,100K,0K,0K,17K,146000K,3416K,25000K,16K/

      CALL OPEN(1,"VPA.DA",2,IER)                    ;DATENFILE OEFFNEN
      IF(IER.EQ.1) GOTO 20
10    TYPE "*** VA1 : FILE VPA.DA NOT OPENED OR READ/WRITE ERROR<7>",IER
      STOP
20    CALL RDBLK(1,1,MASKE(1,1),1,IER)               ;PARAMETER LESEN
      IF(IER.NE.1) GOTO 10
30    CALL CLOSE(1,IER)
      IF(IER.NE.1) GOTO 10
      IN(1)=INEU(1,3)                                ;LEIDER NOTWENDIG
      IN(2)=INEU(2,3)
      IN(3)=INEU(2,4)
      IN(4)=INEU(2,5)
      IN(5)=INEU(1,4)
      MA(1)=MASKE(1,1)
      MA(2)=MASKE(2,1)
      MA(3)=MASKE(2,2)
      MA(4)=MASKE(2,3)
      MA(5)=MASKE(1,2)

C     ==== VERZWEIGUNG, ZUSTAND VON V1,V2,V3 ALS VERZWEIGUNGSKRITERIUM =======

      IV1=ITEST(INEU(2,3),0)
      IV2=ITEST(INEU(2,3),1)
      IV3=ITEST(INEU(2,3),2)
      IF(.NOT.IV1.AND..NOT.IV2.AND..NOT.IV3) GOTO 100
      IF(IV1.AND..NOT.IV2.AND..NOT.IV3) GOTO 200
      IF(.NOT.IV1.AND.IV2.AND..NOT.IV3) GOTO 300
      IF(IV1.AND.IV2.AND..NOT.IV3) GOTO 400
      IF(.NOT.IV1.AND..NOT.IV2.AND.IV3) GOTO 500
      IF(IV1.AND..NOT.IV2.AND.IV3) GOTO 600
      IF(.NOT.IV1.AND.IV2.AND.IV3) GOTO 700
      IF(IV1.AND.IV2.AND.IV3) GOTO 800

C     ------------------------  V1=ZU ,V2=ZU ,V3=ZU  ------------------------------
100   DO 110 J=1,4
      DO 120 K=1,5
      DO 130 I=1,16
      IF(.NOT.ITEST(MA(K),I-1)) GOTO 130
      IF(.NOT.ITEST(IGM1(K,J),I-1)) GOTO 130
      IF(ITEST(IGL1(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 110
130   CONTINUE
120   CONTINUE
      IHF=1
```

```
      GOTO 900
110   CONTINUE
      GOTO 955
C     ------------------------  V1=AUF,V2=ZU ,V3=ZU  ------------------------------------
200   DO 210 J=1,3
      DO 220 K=1,5
      DO 230 I=1,16
      IF(.NOT.ITEST(MA(K),I-1)) GOTO 230
      IF(.NOT.ITEST(IGM2(K,J),I-1)) GOTO 230
      IF(ITEST(IGL2(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 210
230   CONTINUE
220   CONTINUE
      IHF=2
      GOTO 910
210   CONTINUE
      GOTO 955
C     ------------------------  V1=ZU ,V2=AUF,V3=ZU  ------------------------------------
300   DO 310 J=1,7
      DO 320 K=1,5
      DO 330 I=1,16
      IF(.NOT.ITEST(MA(K),I-1)) GOTO 330
      IF(.NOT.ITEST(IGM3(K,J),I-1)) GOTO 330
      IF(ITEST(IGL3(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 310
330   CONTINUE
320   CONTINUE
      IHF=3
      GOTO 900
310   CONTINUE
      GOTO 955
C     ------------------------  V1=AUF,V2=AUF,V3=ZU  ------------------------------------
400   DO 410 J=1,6
      DO 420 K=1,5
      DO 430 I=1,16
      IF(.NOT.ITEST(MA(K),I-1)) GOTO 430
      IF(.NOT.ITEST(IGM4(K,J),I-1)) GOTO 430
      IF(ITEST(IGL4(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 410
430   CONTINUE
420   CONTINUE
      IHF=4
      GOTO 920
410   CONTINUE
      GOTO 955
C     ------------------------  V1=ZU ,V2=ZU ,V3=AUF  ------------------------------------
500   DO 510 J=1,5
      DO 520 K=1,5
      DO 530 I=1,16
      IF(.NOT.ITEST(MA(K),I-1)) GOTO 530
      IF(.NOT.ITEST(IGM5(K,J),I-1)) GOTO 530
      IF(ITEST(IGL5(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 510
530   CONTINUE
520   CONTINUE
      IHF=5
      GOTO 900
510   CONTINUE
      GOTO 955
C     ------------------------  V1=AUF,V2=ZU ,V3=AUF  ------------------------------------
600   DO 610 J=1,5
      DO 620 K=1,5
      DO 630 I=1,16
      IF(.NOT.ITEST(MA(K),I-1)) GOTO 630
      IF(.NOT.ITEST(IGM6(K,J),I-1)) GOTO 630
      IF(ITEST(IGL6(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 610
630   CONTINUE
620   CONTINUE
```

```
C
        IHF=6
        GOTO 910
610     CONTINUE
        GOTO 955
C       ------------------------  V1=ZU  ,V2=AUF,V3=AUF  ------------------------------------
700     DO 710 J=1,7
        DO 720 K=1,5
        DO 730 I=1,16
        IF(.NOT.ITEST(MA(K),I-1)) GOTO 730
        IF(.NOT.ITEST(IGM7(K,J),I-1)) GOTO 730
        IF(ITEST(IGL7(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 710
730     CONTINUE
720     CONTINUE
        IHF=7
        GOTO 900
710     CONTINUE
        GOTO 955
C       ------------------------  V1=AUF,V2=AUF,V3=AUF  ------------------------------------
800     DO 810 J=1,7
        DO 820 K=1,5
        DO 830 I=1,16
        IF(.NOT.ITEST(MA(K),I-1)) GOTO 830
        IF(.NOT.ITEST(IGM8(K,J),I-1)) GOTO 830
        IF(ITEST(IGL8(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 810
830     CONTINUE
820     CONTINUE
        GOTO 950
810     CONTINUE
        GOTO 955
C       ------------------------ UEBERPRUEFUNG KRYOFALLE ------------------------------------
900     DO 901 J=1,3
        DO 902 K=1,5
        DO 903 I=1,16
        IF(.NOT.ITEST(MA(K),I-1)) GOTO 903
        IF(.NOT.ITEST(IKRM(K,J),I-1)) GOTO 903
        IF(ITEST(IKRL(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 901
903     CONTINUE
902     CONTINUE
        IF(IHF.EQ.3) GOTO 920
        IF(IHF.EQ.7) GOTO 940
        GOTO 910
901     CONTINUE
        GOTO 955
C       ------------------- UEBERPRUEFUNG TARGETKAMMER ------------------------------------
910     DO 911 J=1,5
        DO 912 K=1,5
        DO 913 I=1,16
        IF(.NOT.ITEST(MA(K),I-1)) GOTO 913
        IF(.NOT.ITEST(ITSM(K,J),I-1)) GOTO 913
        IF(ITEST(ITSL(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 911
913     CONTINUE
912     CONTINUE
        IF(IHF.EQ.5.OR.IHF.EQ.6) GOTO 940
        GOTO 920
911     CONTINUE
        GOTO 955
C       ---------------- UEBERPRUEFUNG SPEKTROMETERKAMMER ------------------------------------
920     DO 921 J=1,4
        DO 922 K=1,5
        DO 923 I=1,16
        IF(.NOT.ITEST(MA(K),I-1)) GOTO 923
        IF(.NOT.ITEST(ISPM(K,J),I-1)) GOTO 923
        IF(ITEST(ISPL(K,J),I-1).NE.ITEST(IN(K),I-1)) GOTO 921
923     CONTINUE
```

```
C
922     CONTINUE
        GOTO 940
921     CONTINUE
        GOTO 955
C       ---------- ABGETRENNTE DIFF-PUMPEN UEBERPRUEFEN ----------

940     IF(ITEST(IN(1),5)) GOTO 941                          ;V42 AUF?
        IF((.NOT.ITEST(IN(3),11).AND..NOT.ITEST(IN(1),6).AND..NOT.
     *  ITEST(IN(3),4)).OR.(ITEST(IN(3),11).AND.ITEST(IN(1),6).AND.
     *  ITEST(IN(3),4))) GOTO 941
        GOTO 955
941     IF(ITEST(IN(2),13)) GOTO 942                         ;V14 AUF?
        IF((.NOT.ITEST(IN(3),7).AND..NOT.ITEST(IN(2),12).AND..NOT.
     *  ITEST(IN(3),0)).OR.(ITEST(IN(3),7).AND.ITEST(IN(2),12).AND.
     *  ITEST(IN(3),0))) GOTO 942
        GOTO 955
942     IF(ITEST(IN(1),1)) GOTO 943                          ;V18 AUF?
        IF((.NOT.ITEST(IN(3),9).AND..NOT.ITEST(IN(1),0).AND..NOT.
     *  ITEST(IN(3),2)).OR.(ITEST(IN(3),9).AND.ITEST(IN(1),0).AND.
     *  ITEST(IN(3),2))) GOTO 943
        GOTO 955
943     IF(ITEST(IN(2),15)) GOTO 944                         ;V16 AUF?
        IF((.NOT.ITEST(IN(3),8).AND..NOT.ITEST(IN(2),14).AND..NOT.
     *  ITEST(IN(3),1)).OR.(ITEST(IN(3),8).AND.ITEST(IN(2),14).AND.
     *  ITEST(IN(3),1))) GOTO 944
        GOTO 955
944     IF(ITEST(IN(1),3)) GOTO 945                          ;V20 AUF?
        IF((.NOT.ITEST(IN(3),10).AND..NOT.ITEST(IN(1),2).AND..NOT.
     *  ITEST(IN(3),3)).OR.(ITEST(IN(3),10).AND.ITEST(IN(1),2).AND.
     *  ITEST(IN(3),3))) GOTO 945
        GOTO 955
C       ---------- UEBERPRUEFUNG R22/V10/V11 ----------

945     IF(ITEST(IN(3),6).AND.ITEST(IN(2),9).AND..NOT.ITEST(IN(2),10)) GOTO 950
        IF(ITEST(IN(3),6).AND.ITEST(IN(2),10).AND..NOT.ITEST(IN(2),9)) GOTO 950
        IF(.NOT.ITEST(IN(3),6).AND..NOT.ITEST(IN(2),10).AND..NOT.
     *  ITEST(IN(2),9)) GOTO 950
        GOTO 955
C       ---------- ZULAESSIG ----------
950     MAFLA(3)=1                                  ; UEBERGABE AN LJ2
        TYPE "                          INITIAL STATE PERMISSIBLE"
        CALL ISET(IAUS(2,2,1),13)                   ; BIT READY SETZEN
        GOTO 990
C       ---------- UNZULAESSIG ----------
955     MAFLA(3)=0                                  ; UEBERGABE AN LJ2
        TYPE "                          INITIAL STATE PROHIBITED"
990     CALL ISET(IAUS(2,2,1),13)                   ; BIT READY SETZEN, WEGEN
        AUS(2,1)=FLOINT(IAUS(1,2,1))                ;          SIAB
C       ------------------------------------------------------------
        CALL OPEN(1,"VPA.DA",2,IER)                 ; PARAM. SCHREIBEN FUER SWAP
        IF(IER.NE.1) GOTO 10
        CALL WRBLK(1,1,MASKE(1,1),1,IER)
        IF(IER.NE.1) GOTO 10
        CALL CLOSE(1,IER)
        IF(IER.NE.1) GOTO 10

        CALL BACK
        END
```

4.4 MAIN-Task MALJ2 (Automatischer Kaltstart der Vakuumanlage)

Wie aus Abb. 2 ersichtlich ist, kann LJ2 entweder direkt von LJ1 aus oder auf dem Umweg über das (mit allen seinen Unterprogrammen hauptspeicherresidente) Multi-Tasking-Programm MALJ2 gestartet werden.

Durch den fakultativen Aufruf des Programms MALJ2 (Schlüsselschalter HAND/RECHNER in Stellung RECHNER!) wird ein automatischer Kaltstart der Vakuumanlage besorgt, der vom ausgeschalteten Zustand aus die 4 Hauptsektionen Kryofalle, Targetkammer, Targetschleuse und Spektrographenkammer vorevakuiert, das Zwischenvakuum (für die Doppeldichtungen) einschaltet und die Diffusionspumpen in Betrieb setzt. Der Benutzer kann dann zu gegebener Zeit die Dichtheit der genannten Sektionen anhand der inzwischen erfolgten Druckanstiege beurteilen, die Ventile über den Diffusionspumpen und zwischen den Hauptsektionen öffnen und die Penning-Vakuummeter einschalten.

Die von der Subtask MAVB1 generierte Folge von Schaltbefehlen ist nicht funktions-, sondern zeitgesteuert, wobei den einzelnen Zeitintervallen Erfahrungswerte zugrundeliegen. Insbesondere wird vor dem Einschalten der Diffusionspumpenheizungen kein expliziter Lecktest für die einzelnen Subsektionen "Diffusionspumpen" durchgeführt. Jeder einzelne der 24 automatisch generierten Befehle wird jedoch, genau so wie in LJ2 jeder manuelle Befehl, auf Zulässigkeit geprüft und gegebenenfalls zurückgewiesen. Ein als unzulässig befundener Befehl wird nach einer Verzögerungszeit wiederholt; nach dem 3. erfolglosen Versuch wird die Sicherheitsabschaltung eingeleitet.

MALJ2 erkennt und bedient alle Zustandsänderungen, soweit sie von der Vakuumanlage herrühren (alle anderen Interrupts werden ignoriert) und reagiert gegebenenfalls mit der Sicherheitsabschaltung. Die Kaltstartautomatik läßt sich per Programm nicht unterbrechen, es sei denn durch Umlegen des Schlüsselschalters HAND/RECHNER auf HAND.

MALJ2 verwendet als Subtasks und Subroutinen eine Reihe der im Abschn. 4.2 beschriebenen, von LJ2 benutzten Unterprogramme, teilweise mit erheblichen Modifikationen. In diesen Fällen unterscheiden sich die Namen entsprechender Programme durch die vorgesetzten Buchstaben MA. Im folgenden ist nur das Quellprogramm MAVB1 wiedergegeben, das die automatische Befehlsfolge generiert.

Eine Anfangsprüfung wird beim Kaltstart nicht durchgeführt. Die Anfahrautomatik beginnt in jedem Falle wie beschrieben ganz von vorn, auch wenn beim Start von LJ1/MALJ2 die Vakuumanlage bereits teilweise oder vollständig eingeschaltet sein sollte.

Die Benutzung des automatischen Kaltstarts ist zu empfehlen:

a) wenn die Vakuumanlage noch ausgeschaltet und "kalt" ist;

b) wenn alle Subsektionen "Diffusionspumpen" als dicht gelten dürfen;

c) wenn während der Anfahrzeit (etwa 4 min bis zum automatischen Laden (Chaining) von LJ2 auf weitere Aktivitäten des E.R. # 2 verzichtet werden kann.

```
C       ********************************************************************************
C       **  TASK  MAVB1                                                                 *
C       **  AUTOMATIKPROGRAMM VAKUUM PUMPEN                                             *
C       **  MANGER FEBRUAR 1984                                                        **
C       ********************************************************************************
        COMPILER NOSTACK
        TASK MAVB1

        COMMON/PRM/MASKE(2,3),A3473(2,5),A3072(2,2),AUS(2,2),
     *  IALT(2,5),INEU(2,5),IAUS(2,2,2),N3473(5),MIV(5),IDT(5),ACR(2),AUHR,
     *  ICR(2),N3072(2),NUHR,IC,IFDEL,LPT,STAT,IHA,SIAB,IDEL(2,2),MAPRO(60),
     *  MAFLA(12),MABIM(5),IDUM(70)
        EXTERNAL MAS1,MAS2,MAS3,MAEX1,MAVP1
        LOGICAL LPT,STAT,IHA,SIAB

        IF(ITEST(INEU(1,2),0)) GOTO 10
        CALL MAVP1(2)
5       IF(ITEST(INEU(1,2),0)) GOTO 10
        GOTO 5
10      CALL MAVP1(1)
C       ================================================================================
        CALL FDELY(50*IFDEL)

        CALL MAS3(12,1)                         ; R44 EIN
        CALL FDELY(50*IFDEL)
        CALL MAS3(6,1)                          ; R21 EIN
        CALL FDELY(50*IFDEL)
        CALL MAS3(7,1)                          ; R22 EIN
        CALL FDELY(50*IFDEL)
        CALL MAS3(8,1)                          ; R23 EIN
        CALL FDELY(50*IFDEL)
        CALL MAS3(9,1)                          ; R25 EIN
        CALL FDELY(50*IFDEL)
        CALL MAS3(10,1)                         ; R27 EIN
C       ================================================================================
        CALL FDELY(300*IFDEL)
        CALL MAS1(6,1)                                  ;V6      "
        CALL FDELY(50*IFDEL)
        CALL MAS2(5,1)                                  ;V41     AUF
        CALL FDELY(50*IFDEL)
        CALL MAS1(10,1)                                 ;V10     "
        CALL FDELY(50*IFDEL)
        CALL MAS1(13,1)                                 ;V13     "
        CALL FDELY(50*IFDEL)
        CALL MAS1(15,1)                                 ;V15     "
        CALL FDELY(50*IFDEL)
        CALL MAS2(1,1)                                  ;V17     "
C       ================================================================================
        CALL FDELY(300*IFDEL)
C       ================================================================================
        CALL MAS1(6,2)                                  ;V6      ZU
        CALL FDELY(50*IFDEL)
        CALL MAS2(5,2)                                  ;V41     "
        CALL FDELY(50*IFDEL)
        CALL MAS1(10,2)                                 ;V10     "
        CALL FDELY(100*IFDEL)
        CALL MAS1(8,1)                                  ;V8      AUF
        CALL FDELY(50*IFDEL)
        CALL MAS1(11,1)                                 ;V11     AUF
        CALL FDELY(50*IFDEL)
        CALL MAS2(7,1)                                  ;V43     AUF
C       ================================================================================
        CALL FDELY(300*IFDEL)                   ; WENN F OK, DANN WEITER
C       ================================================================================
```

```
      CALL MAS3(1,1)                          ; D24 EIN
      CALL MAVF1(3)
      CALL FDELY(50*IFDEL)
      CALL MAS3(2,1)                          ; D26 EIN
      CALL MAVF1(4)
      CALL FDELY(50*IFDEL)
      CALL MAS3(3,1)                          ; D28 EIN
      CALL MAVF1(5)
      CALL FDELY(50*IFDEL)
      CALL MAS3(5,1)                          ; D45 EIN
      CALL MAVF1(7)
C     ================================================================================
      CALL FDELY(50*IFDEL)
      CALL MAVF1(8)
      MAFLA(10)=10                            ; FLAGGE FUER EX1
      CALL MAEX1                              ; NUR FUER INTERRUPTS SPERREN
      CALL CHAIN("LJ2.SV",IER)
      STOP
      END
```

4.5 Protokoll-Beispiel

Das nachfolgende Protokollbeispiel zeigt einen normalen, vollständigen Evakuierungsvorgang der Vakuumanlage des Magnetspektrographen unter Verwendung der Kaltstart- und der Abschalt-Automatik, so wie in der Benutzeranleitung /5/ beschrieben.

```
18.02.84  18.02.84  18.02.84  18.02.84  18.02.84  18.02.84  18.02.84
==========================================================================
10:27:20              COLD STARTING PROCEDURE IN PROGRESS!
                      WAIT FOR MESSAGE TO CONTINUE
10:30:51              ====================
                      SECTION D24 READY
10:30:59              ====================
                      SECTION D26 READY
10:31: 6              ====================
                      SECTION D28 READY
10:31:13              ====================
                      SECTION D45 READY
                      ====================
10:31:18              COLD STARTING PROCEDURE FINISHED!
10:31:21              LJ2 LOADED
10:31:21              COMPUTER CONTROL, INITIAL CHECK UP FOLLOWS
                      INITIAL STATE PERMISSIBLE
10:47:22              T26 =  1
10:49:21              T28 =  1
10:55:34              T24 =  1
10:56:22              T45 =  1
11:01:11  B V14 : 1
11:01:12              V14 =  1
11:01:14              F 8 =  1
11:01:23  B V18 : 1
11:01:23              V18 =  1
11:01:24              F12 =  0
11:01:27  B V16 : 1
11:01:28              V16 =  1
11:01:39              F12 =  1
11:01:45  B V42 : 1
11:01:45              V42 =  1
11: 2: 4              H 1 =  1
11: 2: 7              H 2 =  1
11: 2: 8              H 3 =  1
11:12:53              SHUT OFF PROCEDURE IN PROGRESS!
12: 1:17              SHUT OFF PROCEDURE FINISHED
12: 2: 7              END OF PROGRAM
```

5. Änderung der Entscheidungslogik

Bei Änderungen der in Tab. 2 und in Tab. 4 festgelegten und in den Programmeinheiten VBS1, VBS2, VBS3 sowie VA1 initialisierten Entscheidungstabellen für die Befehlsprüfung und für die Anfangsprüfung ist zu unterscheiden, ob lediglich ein bereits bestehender Initialisierungswert zu korrigieren ist, oder ob eine Entscheidungstabelle um neue zulässige Zustandskombinationen erweitert werden soll.

5.1 Korrektur der Anfangsprüftabelle

Eine Änderung eines einzelnen zulässigen Anfangszustands ist in VA1 an einer einzigen Stelle, u.z. im DATA-Statement der betreffenden Funktionseinheit (vgl. 4.3), durchzuführen. Soll z.B. in der Betriebsart "Vorevakuieren" der Funktionseinheit 1 (Tab. 4, Spalte 1) eine Korrektur vorgenommen werden, die eines der Ventile V17 bis V43, d.h. das 1. der 5 INTEGER-Worte, betrifft, so ist die Initialisierung des jeweils 11. Elements der Felder IGL1() und IGM1() zu korrigieren. (Der Betriebsart "Vorevakuieren" gehen 2 zulässige Anfangszustände für diese Funktionseinheit, nämlich "Abgetrennt" und "Belüften", voraus, und für jeden zulässigen Anfangszustand sind im DATA-Statement 5 INTEGER-Worte oktal initialisiert.)

5.2 Erweiterung der Anfangsprüftabelle

Soll die Anzahl zulässiger Anfangszustande z.B. um 1 weiteren zulässigen Anfangszustand erweitert werden, dann ist VA1 an 3 Stellen zu ändern:

* Im DIMENSION-Statement muß die 2. Dimension der Felder I_L_() und I_M_() um 1 erhöht werden;

* Im DATA-Statement müssen die 5 zusätzlichen Oktalzahlen, die die neu hinzugekommenen zulässigen Anfangszustände beschreiben, sowohl in I_L_() als auch in I_M_() eingefügt werden;

* Für die äußere DO-Schleife (z.B. Statement Nr. 100 für die Funktionseinheit # 1) muß der Schleifenzähler um 1 erhöht werden.

5.3 Korrektur der Befehlsprüftabelle

Eine Änderung der Bedingungen für einen einzelnen zulässigen Befehl ist entweder im DATA-Statement der Programmeinheit VBS1 (Ventile V1 bis V16) oder der Programmeinheit VBS2 (Ventile V17 bis V43) oder der Programmeinheit VBS3 (Pumpen) durchzuführen. Existiert für einen Befehl nur eine einzige zulässige Zustandskombination, z.B. für "V13 AUF", dann ist diese in den Feldern ILIS() und IMAS() initialisiert; existieren jedoch für einen Befehl mehrere zulässige Zustandskombinationen, z.B. für "V14 AUF", dann sind diese in den Feldern ILn() und IMn() initialisiert, wobei n für die Aggregatnummer steht (im Beispiel: n = 14).

5.4 Erweiterung der Befehlsprüftabelle

Soll die Befehlsprüftabelle um eine weitere zulässige Zustandskombination für einen Befehl, für den bereits mehrere zulässige Kombinationen existieren, erweitert werden, dann ist VBS1 bzw. VBS2 bzw. VBS3, wie oben VA1, an 3 Stellen zu ändern: im DIMENSION-Statement, im DATA-Statement und im DO-Statement. Besteht jedoch bislang nur eine einzige zulässige Zustandskonbination für den betreffenden Befehl, dann muß die Erweiterung nach folgendem Schema durchgeführt werden:

* Die bisherigen 5 INTEGER-Worte in ILIS() und IMAS() sind zu nullen;

* Für das Aggregat mit der Nummer n sind die Felder ILn() und IMn() neu anzulegen (DIMENSION ...) und zu initialisieren (DATA ...);

* Es sind ein IF-Statement und eine komplette DO-Schleife nach dem Muster z.B. von V14 hinzuzufügen.

5.5 Parallelaufgaben zur Vakuumsteuerung

Zusätzlich zur Steuerung der Vakuumanlage und ohne Beeinträchtigung dieser vorrangigen Aufgabe soll LJ2 später eventuell weitere Aufgaben wie Positionierungen, Verwaltung von Datenbanken, Aufbereiten und Darstellen von Statusparametern auf einem Videoterminal, Hilfsrechnungen u.ä. übernehmen. Nach der derzeitigen Vorstellung kann dies mittels Service-Tasks entsprechend geringer Priorität erfolgen, die einen von LJ2 zu identifizierenden Anlagen-Interrupt (über CAMAC), System-Interrupt (z.B. Uhr) oder manuellen Interrupt (z.B. über Tastatur, Sensorboard, Lightpen) bedienen. Hierfür sind noch ca. 4 KByte Hauptspeicher verfügbar.

Die Einbindung neuer Funktionen in ein bestehendes Real-Time-Programmsystem bedeutet einen tieferen Eingriff und erfordert dessen Verständnis sowie Einfühlungsvermögen. Für funktionell erweiterte Programmversionen sollten tunlichst Namen mit anderen Endnummern, z.B. LJ3 oder LJ21, verwendet werden.

Ich danke Herrn Prof. Dr. G. S c h a t z und Herrn Prof. Dr. H. R e b e l für die Ermöglichung dieser Arbeit sowie den Herren J. B i a l y, Dr. J. B u s c h m a n n, Dr. H. J. G i l s, H. H e i n z m a n n, B. K ö g e l, G. L u d w i g, W. S e i t h, Dr. T. T h o u w und S. Z a g r o m s k i für wertvolle Diskussionen und freundliche Unterstützung.

6. Referenzen

/1/ H.J. Gils, J. Buschmann, M. Heinz, J. Krisch, H. Rebel, S. Zagromski:
in Report KfK 3427, Kernforschungszentrum Karlsruhe (1982), S. 167.

/2/ D. Manger:
unveröffentlichte Diplomarbeit, Fachhochschule Karlsruhe (1982).

/3/ J. Buschmann, D. Manger:
unveröffentlicher Bericht, Kernforschungszentrum Karlsruhe (1983).

/4/ J. Buschmann, H.J. Gils, J. Krisch, G. Ludwig, D. Manger, H. Rebel, W. Seith, and S. Zagromski:
Report KfK 3681 B, Kernforschungszentrum Karlsruhe (in Vorbereitung)

/5/ D. Manger:
unveröffentlichte Benutzeranleitung, Kernforschungszentrum Karlsruhe (1984).

/6/ D. Manger, J. Buschmann:
unveröffentlichter Bericht, Kernforschungszentrum Karlsruhe (1982).

/7/ G. Ehret:
unveröffentlichte Programmbeschreibung, Kernforschungszentrum Karlsruhe (1978).

/8/ W. Seith, J. Buschmann:
unveröffentlichter Bericht, Kernforschungszentrum Karlsruhe (1983).