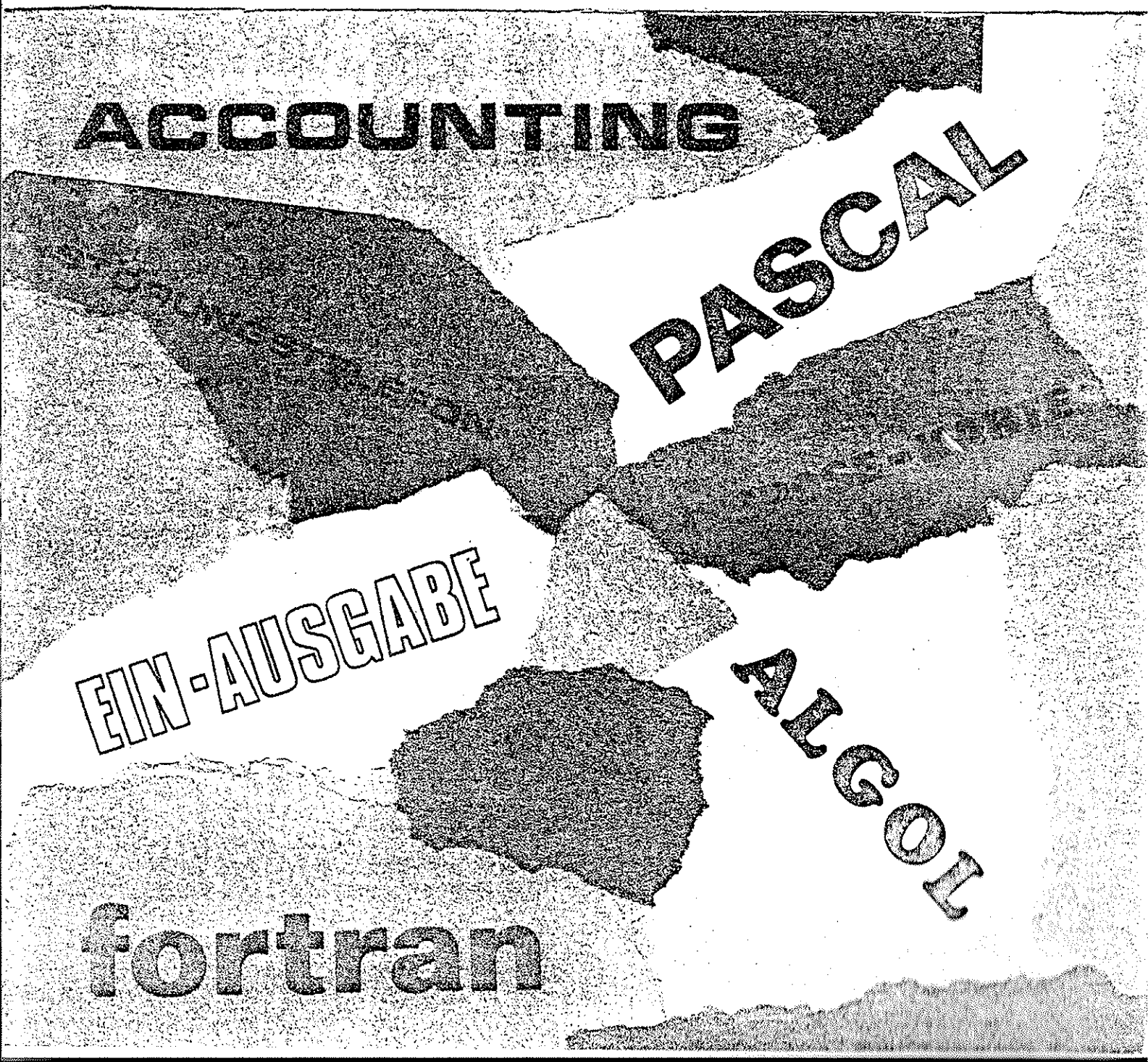
HEISSER DRAHT

RECHENZENTREN
UNI-TH WIEN
CYBER 73-74

NUMMER 11

MÄRZ 1975



HEISSER DRAHT

GEMEINSAMES MITTEILUNGSBLATT DER RECHENZENTREN
DER UNIVERSITÄT UND TECHNISCHEN HOCHSCHULE WIEN

Wie Sie vielleicht am geänderten Titelblatt bemerkt haben, wird künftig der "Heiße Draht" als ein gemeinsames Mitteilungsblatt der Rechenzentren der Universität und der Technischen Hochschule Wien erscheinen. Damit sollen die Benutzer der beiden CDC-Rechner über die in vielen Belangen gemeinsamen Einrichtungen der beiden Rechenzentren informiert werden, sodaß auch in organisatorischer Hinsicht der Kompatibilität der beiden Rechenanlagen in vermehrtem Ausmaß Rechnung getragen wird.

Wir hoffen, durch diese Maßnahme den Benutzern, die vielfach beide Rechner in Anspruch nehmen, eine bessere Information über die Serviceleistungen der Rechenzentren bieten zu können. Damit wollen wir die vielfältigen Benutzerinteressen, in einem für alle Beteiligten fruchtbaren Gespräch vereinigen.

Für Anregungen und Kritik hinsichtlich der Gestaltung dieses Mitteilungsblattes sind wir Ihnen jederzeit dankbar.

Bodenseher Grafendorfer Rastl Schornböck

Eigentümer, Herausgeber und Verleger:
Rechenzentren UNI - TH Wien, Cyber 73-74.
Für UNI-Wien: Universitätsstraße 7, 1010 Wien
Für TH-Wien: Gußhausstraße 27-29, 1040 Wien
Für den Inhalt verantwortlich: Manfred Weiss.
Druck: Österr. Hochschülerschaft der TH-Wien.

Nr. 11

IN MEMORIAM ERICH BUKOVICS

Der schmerzliche Verlust, den die Technische Hochschule Wien durch das Ableben von o.Prof.Dr.Erich Bukovics schlagartig erlitt, ist auch für das Interfakultäre Rechenzentrum, dessen Vorstand Bukovics durch viele Jahre angehörte, von großer Tragweite. Hat sich doch Bukovics dem Aufbau, dem Betrieb und der Erweiterung des Interfakultären Rechenzentrums mit großer Hingabe gewidmet, wie er allen Aufgaben in führender Stellung im Kollegium, als mehrjähriger Rektor und Prorektor, im Rechenzentrum und Institut hoher Sorgfalt beigemessen und die Lehre der Mathematik stets wahrhaft vorbildlich gestaltet hat.

Alle, die mit Bukovics in fachlichem oder organisatorischem Kontakt standen, bewunderten neben seinen wissenschaftlichen und fachlichen Leistungen seinen klaren Verhandlungsstil und seinen Blick für das Wesentliche jeder Problematik. Wir werden Erich Bukovics als leuchtendes Vorbild in bleibender Erinnerung behalten.

Vorstand, Leitung und Mitarbeiter
des Interfakultären Rechenzentrums
der Technischen Hochschule Wien

MÄRZ 1975

EINGABE

BETRIEBSZEITEN DES INTERFAKULTÄREN RECHENZENTRUMS DER TECHNISCHEN HOCHSCHULE WIEN

RECHENANLAGE CYBER 74

Batch-Betrieb :	MO	:	14.00	-	24.00	Uhr
	DI - FR	:	00.00	-	06.00	Uhr
			08.30	-	24.00	Uhr
	SA	:	00.00	-	06.00	Uhr
Letzte Eingabe	:	MO - FR	um 21.00		Uhr	
Intercom-Betrieb:	MO	:	14.00	-	21.00	Uhr
	DI - FR	:	09.00	-	21.00	Uhr

DATENSTATION CD 1700 (HAUPTGEBÄUDE)

Montag - Freitag	:	09.00	-	20.00	Uhr

LOCHERRAUM NEUES GEBÄUDE

Montag - Freitag	:	08.00	-	21.00	Uhr

LOCHERRAUM HAUPTGEBÄUDE

Montag - Freitag	:	08.00	-	21.00	Uhr

PROGRAMMBERATUNG

Montag	:	14.00	-	16.00	Uhr
Dienstag - Freitag	:	09.30	-	12.00	Uhr
		14.00	-	16.00	Uhr

SEKRETARIAT

Montag - Freitag	:	09.00	-	15.00	Uhr

AUSGABE

hd

MÄRZ 1975

EINGABE

BETRIEBSZEITEN DES INTERFAKULTÄREN RECHENZENTRUMS DER UNIVERSITÄT WIEN

RECHENANLAGE CYBER 73

Batch-Betrieb:	MO, MI, FR	:	08.00	- 22.00	Uhr
	DI, DO	:	11.00	- 22.00	Uhr
	SA	:	08.00	- 15.00	Uhr
Intercom-Betrieb:	MO, MI, FR	:	08.00	- 20.00	Uhr
	DI, DO	:	11.00	- 20.00	Uhr
	SA	:	08.00	- 14.00	Uhr

Fallweise werden am Freitag von 07.00 - 11.00 Uhr Systemarbeiten durchgeführt. In diesem Fall entfällt der Produktionsbetrieb.

PROGRAMMBERATUNG

Montag - Freitag	:	09.30	- 12.00 Uhr
		14.00	- 17.00 Uhr

SEKRETARIAT

Montag - Freitag	:	09.00	- 12.00 Uhr

AUSGABE

hd MÄRZ 1975

HINWEISE

ZUR SPARSAMEN VERWENDUNG DES KERNSPEICHERPLATZES

An der CYBER 73 des interfakultären Rechenzentrums der Universität Wien stehen derzeit 48 k Kernspeicher (für das Betriebssystem und Benutzerprogramme) zur Verfügung. Die Feldlänge eines Programmes kann maximal etwa 100000 (oktal) Worte (zu 60 Bit) betragen. Erfahrungsgemäß macht die Adaptierung großer FORTRAN-Programme an der CYBER 73 häufig Schwierigkeiten, wenn der Kernspeicherbedarf zu groß ist. An der CYBER 74 des Rechenzentrums der Technischen Hochschule Wien kann ein Programm bis zu 170000 (oktal) Worte beanspruchen, doch beeinträchtigt die Größe des Programmes die Turnaroundzeiten wesentlich. Es gibt jedoch einige Möglichkeiten, wie man ohne allzu großen Aufwand Kernspeicherplatz einsparen kann.

1) Verlegen von Feldern ins unbenannte Common (blank common)

Während des Ladevorganges muß im Kernspeicher genug Platz sowohl für das gesamte Programm (einschließlich der dimensionierten lokalen Variablen, der Ein/Ausgabepuffer, der vom System hinzugefügten Routinen usw.), als auch für den Loader selbst sein. Ein unbenanntes Common wird am Ende der Programmfeldlänge hinter den Programmcode geladen. Man nützt daher einen Teil des Platzes aus, den während des Ladevorganges der Loader selbst belegt, wenn man große Felder in ein unbenanntes Common verlegt.

2) Einsparung bei der Vergabe der Ein/Ausgabepuffer

Für jede Datei, die im PROGRAM-Statement genannt wird, legt das System automatisch einen Puffer in der Größe von oktal 1001 (CYBER 73) bzw. 2002 (CYBER 74)(=513 bzw. 1026 dezimal) Worten an. In vielen Fällen kann man (ohne wesentliche Verlangsamung) mit weniger Puffer auskommen. Man erreicht dies,indem man im PROGRAM-Statement die Angabe

```
filename=buffersize
```

einfügt, also z.B.:

```
PROGRAM XYZ (TAPE2=65,TAPE4=129,TAPE8=513)
```

Die minimalen Puffergrößen für READ/WRITE sind die Größen der Physical Record Unit (PRU) plus EINS, also für:

Datei auf Platte, formatiert (insbesondere die Dateien INPUT, OUTPUT, PUNCH)	65 Worte
SCOPE-Tape, formatiert	129 Worte
Datei (Platte oder Band), unformatiert ..	513 Worte

S-Tape ..... 513 Worte

3) Verwenden von BUFFER IN/OUT

Falls man Felder in unformatierter Form auf einen Externspeicher zwischenspeichert, so kann man anstelle von READ bzw. WRITE die Anweisungen BUFFER IN/OUT verwenden. Damit werden die Daten ohne Verwendung eines Puffers übertragen

```
BUFFER IN (u,p) (a,b)
BUFFER OUT (u,p) (a,b)
```

u .... Einheitennummer, entsprechend TAPEu
p .... Ø oder 1 (Angabe der Parität für Siebenspurbänder, sonst belanglos)
a .... erstes Wort des zu übertragenden Records
b .... letztes Wort des zu übertragenden Records

Beispiel:

```
DIMENSION REC(512)
BUFFER OUT(12,Ø) (REC(1),REC(512))
```

In diesem Fall wird ohne zusätzliche Puffer direkt von Feld REC auf TAPE12 übertragen. Um den automatisch reservierten Puffer freizugeben, muß in PROGRAM-Statement die Puffergröße im Null angegeben werden

```
PROGRAM XYZ (TAPE12=Ø)
```

Im Zusammenhang benötigt man die eingebaute Funktion UNIT zur Abfrage des Dateizustandes; genauere Informationen darüber sind den FORTRAN EXTENDED-Handbüchern zu entnehmen.

4) Packen von Zeichen oder Zahlen im Kernspeicher

Häufig kann man große Einsparungen an Kernspeicherplatz erzielen, wenn man in jedem Kernspeicherwort mehr als ein Zeichen bzw. mehr als eine große Zahl unterbringt. Man kann z. B. in einem Kernspeicherwort zehn Zeichen speichern; falls man die einzelnen Zeichen benötigt, können sie unter Anwendung der Anweisungen ENCODE/DECODE oder unter Verwendung der eingebauten Funktionen SHIFT, MASK, AND, OR, NOT wieder verfügbar gemacht werden.

Beispiel:

```
C     EINLESEN VON 8Ø ZEICHEN
      DIMENSION KARTE(8)
      READ(5,1) KARTE
    1 FORMAT(8A1Ø)
```

um z. B. das vierzehnte Zeichen aus dem Feld KARTE wieder zu gewinnen, kann man schreiben:

```
      DECODE(8Ø,2,KARTE) IZEICH
    2 FORMAT(13X,R1)
```

oder auch

```
IZEICH=SHIFT(KARTE(2),24).AND.77B
```

In ähnlicher Weise kann man beim Speichern ganzer Zahlen verfahren, die einen vorgegebenen kleinen Zahlenbereich haben. Zum Speichern einer Zahl zwischen 0 und 7 benötigt man z. B. nur 3 Bit; man bringt also 20 solcher Zahlen in einem 60-Bit-Wort unter.

5) Verwendung von IGETP und IPUTP

Um dem Benützer das Packen und Entpacken ganzer Zahlen mit gegebenen Wertebereich zu erleichtern, sind an der CYBER 73 (derzeit nicht an der CYBER 74!) die beiden Routinen IGETP und IPUTP implementiert worden. Diese ermöglichen es, kleinere Einheiten als 60-Bit-Worte im Kernspeicher anzusprechen.

Aufrufe:

```
CALL IPUTP(ibit,array,index,ivalue)
ivalue = IGETP(ibit,array,index)
```

Damit können Einheiten zu ibit Bits angesprochen werden.
ibit muß ein Teiler von 60 sein.
IPUTP speichert die Zahl ivalue in die index-ten Einheit im Feld array.
IGETP liefert den Zahlenwert, der im Feld array in der index-ten Einheit gespeichert ist.
array ist ein Datenfeld, das entsprechend dimensioniert werden muß.

Beispiel: Es sollen 3000 Zahlen zwischen 0 und 1000 gespeichert werden. Dazu sind jeweils 10 Bits ausreichend, man kann also 6 Zahlen pro Maschinenwert speichern. Man benötigt daher insgesamt statt 3000 nur 500 Maschinenworte.

```
      PROGRAM GETPUT(INPUT=65,OUTPUT=65)
      DIMENSION FELD(5ØØ)
      DO 1 K=1,3000
      READ 4,M
4     FORMAT (I3)
1     CALL IPUTP(1Ø,FELD,K,M)
      DO 2 K=1,3ØØØ
X     N=IGETP(1Ø,FELD,K)
2     PRINT 3,N
3     FORMAT...
```

Das Unterprogramm IGETP/IPUTP gibt bei fehlerhafter Verwendung die Fehlermeldung "ERROR IN IGETP/IPUTP" und bricht das Programm ab. Dies ist z. B. der Fall, wenn ein zu speichernder Wert zu groß ist, sodaß signifikante Bits verloren gehen würden.

6) Die EQUIVALENCE-Anweisung

Die EQUIVALENCE-Anweisung kann häufig ebenfalls verwendet werden, um Kenrspeicherplatz einzusparen, wenn nämlich zwei Felder während des Programmablaufs nicht gleichzeitig benötigt werden. Z.B.:

```
      DIMENSION FELD1(1ØØØ),FELD2(8ØØ)
      EQUIVALENCE (FELD1,FELD2)
```

Damit benötigt FELD2 keinen zusätzlichen Speicher mehr, kann jedoch nur in Programmteilen verwendet werden, in denen eine Änderung der Werte in FELD1 keine Rolle spielt.

7) Ablegen von Feldern auf Externspeicher

Häufig ist es möglich, anstelle eines großen Feldes im Kernspeicher einen Externspeicher zu verwenden. Bei sequentiellem Zugriff kann man die Anweisungen WRITE, REWIND und READ verwenden.

Bei der Verwendung der Routinen READMS/WRITMS zum Direktzugriff ist allerdings zu beachten, daß für jedes Record am Externspeicher ein Wort für eine Indexeintragung im Kernspeicher benötigt wird.

Beispiel:

```
      PROGRAM  XYZ (TAPE15)
      DIMENSION INDEX(3ØØØ),DAT (25)
      CALL OPENMS (15,INDEX,3ØØØ,Ø)
      DO 1 I=1,2999
    1 CALL WRITMS(15,DAT ,25,I)
```

Um also 2999 Records zu schreiben, benötigt man im Kernspeicher ein Indexfeld der Länge 3000. Auch für dieses Feld kann Kernspeicherplatz eingespart werden (etwa EQUIVALENCE oder Auslagerung des Feldes).

Für andere Dateiorganisationsformen steht die direkte Verwendung des Record Manager in FORTRAN zur Verfügung (siehe FORTRAN-Manual Kapitel III-6 und RECORD MANAGER Reference Manual).

8) Overlays und Segmente

Schließlich kann man fast immer den zur Verfügung stehenden Kernspeicher besser ausnützen, wenn man das Programm in Overlays oder Segmente zerlegt. Informationen über die Technik und Handhabung von Overlays findet man im FORTRAN-Manual (Kapitel I,12), die Segmentierung ist im LOADER Reference Manual ausführlich beschrieben. Die Hauptunterschiede zwischen Overlays und Segmenten sind die folgenden:

- Overlays müssen innerhalb des Programms durch ein OVERLAY-Statement definiert und explizit mit Hilfe des Aufrufes CALL OVERLAY geladen werden. Die Parameterübergabe zwischen den Overlays erfolgt ausschließlich über COMMON.

Segmente werden durch Loader-Steuerkarten definiert; das Programm selbst bleibt dabei völlig unverändert und das Laden erfolgt automatisch beim Aufruf eines Unterprogrammes. Die Parameterübergabe ist auch über Parameterlisten möglich.

Bei Problemen im Zusammenhang mit dem Kernspeicherbedarf, Programmierung von Überlagerungsstrukturen und Verwendung von Externspeichern usw. steht die Programmberatung der Rechenzentren gerne zur Verfügung.

IM blick punkt

BETRIEBSMITTEL-ABRECHNUNG AN DER CYBER 74

Wie bereits im November 1974 angekündigt, ist nun mit der automatischen Kontrolle des Betriebsmittelverbrauches begonnen worden. Die folgenden Ausführungen sollten die wichtigsten Maßnahmen erläutern.

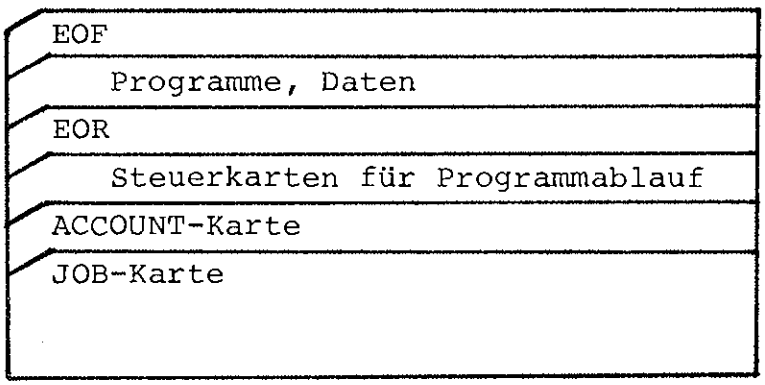
1. ANSUCHEN

Für jedes Projekt, das an der Rechenanlage CYBER 74 gerechnet wird, muß um eine Account-Nr. angesucht werden (Neuansuchen). Sind die bewilligten Betriebsmittel verbraucht, so muß um Verlängerung angesucht werden.

Die nötigen Formulare sind in der Programmberatung oder im Sekretariat erhältlich.

2. STEUERKARTEN FÜR DAS ACCOUNTING

2.1 Aufbau eines Jobs



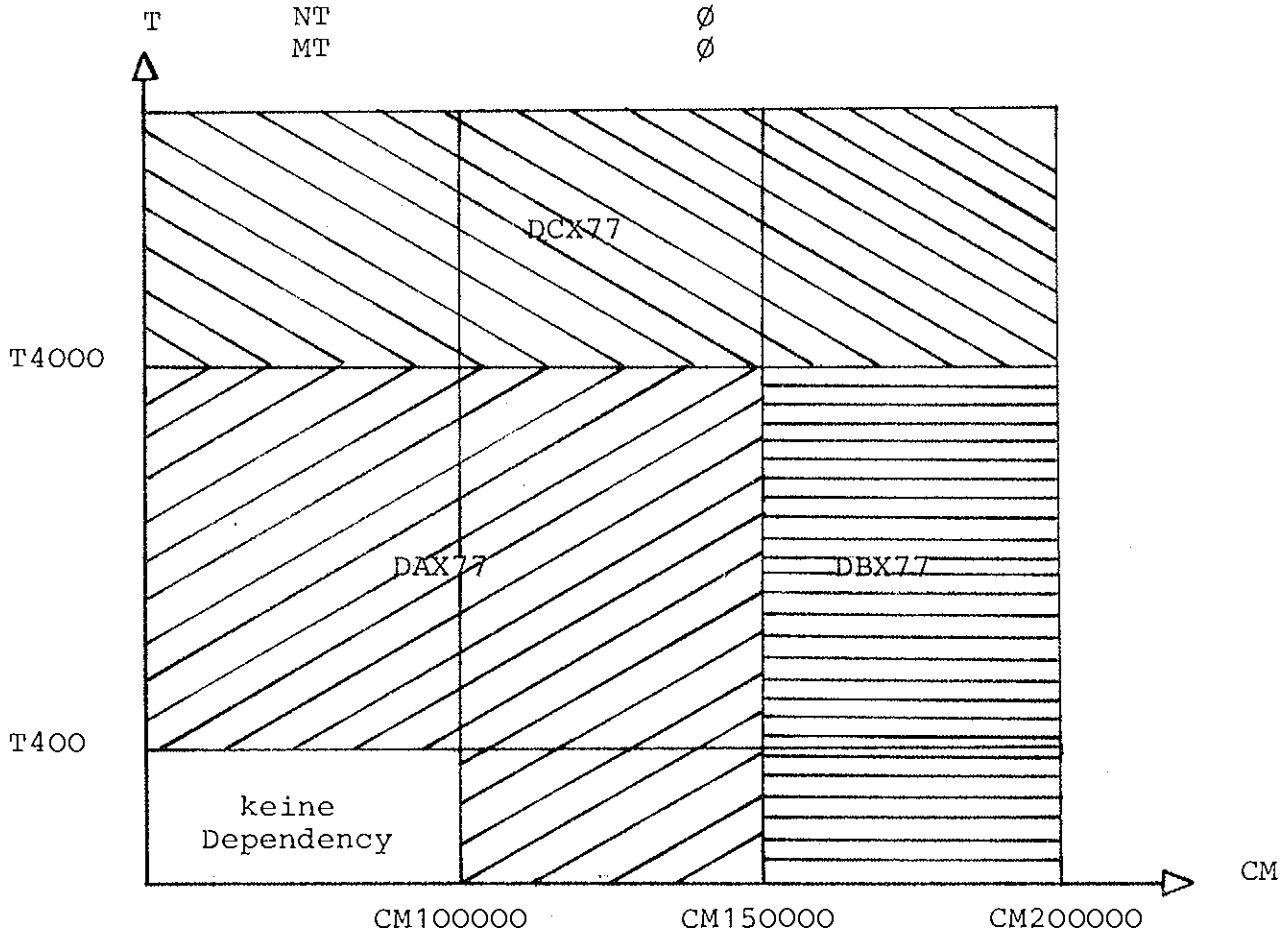
2.2 JOBKARTE

Neben den von SCOPE erlaubten Parametern auf der Jobkarte (mit Ausnahme von P für Priorität) muß der D-Parameter (Dependency) angegeben werden, wenn die Angaben

der Jobkarte einer Jobklasse mit Dependency entsprechen (Achtung: diese Angaben dürfen die bewilligten Werte der Limits pro Job nicht überschreiten!)

Standardwerte

CM	100 000 Worte (oktal)
T	400 Sekunden (oktal)
NT	Ø
MT	Ø



Beispiel: job,T100,CM160000,DBX77.

Wenn der Ausdruck mehr als 150.000 Zeichen (ca. 50 Seiten) beträgt, muß

- bei der Dependency anstelle von X ein P angegeben werden
  Beisp.: job,T100,CM160000,DBP77.
- eine DISPOSE-Karte verwendet werden
  Beisp.: DISPOSE,OUTPUT,P2=CO1.

2.3 ACCOUNT-KARTE

Formaler Aufbau: ACCOUNT,accnr,password,SUP.

Alle unterstrichenen Angaben müssen angegeben werden.

IM blick punkt

BETRIEBSMITTEL-ABRECHNUNG AN DER CYBER 74

Wie bereits im November 1974 angekündigt, ist nun mit der automatischen Kontrolle des Betriebsmittelverbrauches begonnen worden. Die folgenden Ausführungen sollten die wichtigsten Maßnahmen erläutern.

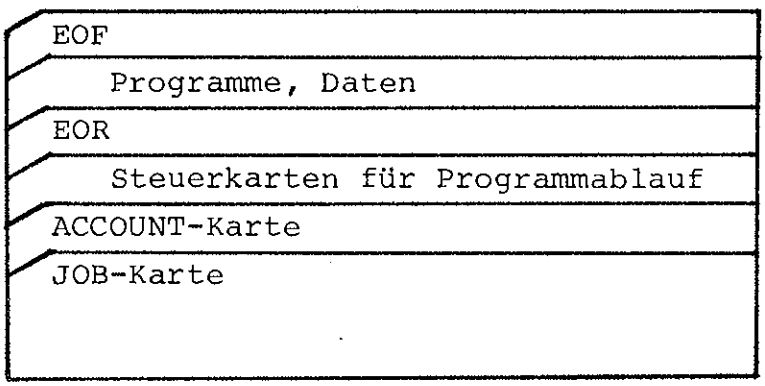
1. ANSUCHEN

Für jedes Projekt, das an der Rechenanlage CYBER 74 gerechnet wird, muß um eine Account-Nr. angesucht werden (Neuansuchen). Sind die bewilligten Betriebsmittel verbraucht, so muß um Verlängerung angesucht werden.

Die nötigen Formulare sind in der Programmberatung oder im Sekretariat erhältlich.

2. STEUERKARTEN FÜR DAS ACCOUNTING

2.1 Aufbau eines Jobs



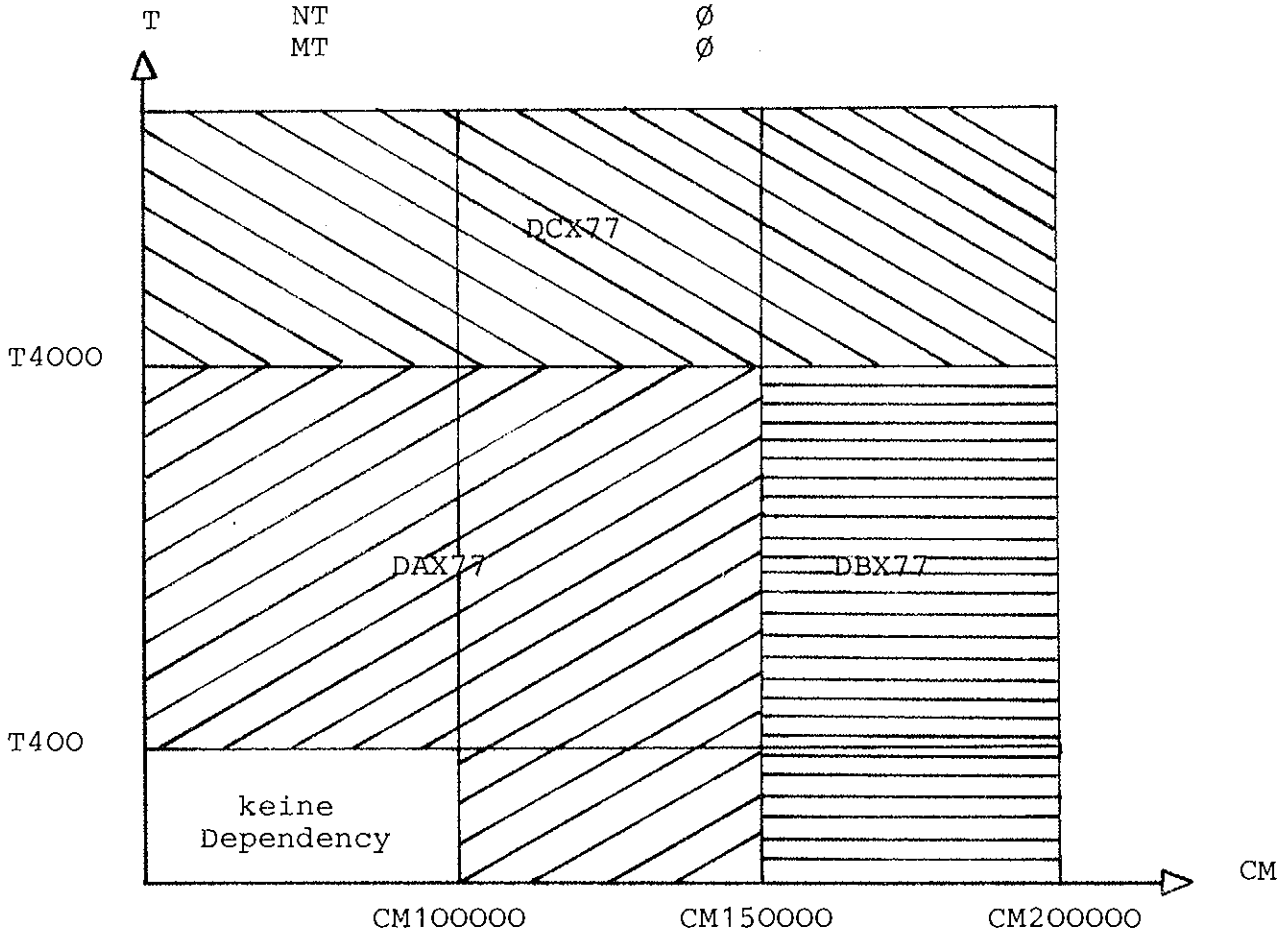
2.2 JOBKARTE

Neben den von SCOPE erlaubten Parametern auf der Jobkarte (mit Ausnahme von P für Priorität) muß der D-Parameter (Dependency) angegeben werden, wenn die Angaben

der Jobkarte einer Jobklasse mit Dependency entsprechen (Achtung: diese Angaben dürfen die bewilligten Werte der Limits pro Job nicht überschreiten!)

Standardwerte

CM	100 000 Worte (oktal)
T	400 Sekunden (oktal)
NT	Ø
MT	Ø



Beispiel: job,T100,CM160000,DBX77.

Wenn der Ausdruck mehr als 150.000 Zeichen (ca. 50 Seiten) beträgt, muß

- bei der Dependency anstelle von X ein P angegeben werden
  Beisp.: job,T100,CM160000,DBP77.
- eine DISPOSE-Karte verwendet werden
  Beisp.: DISPOSE,OUTPUT,P2=CO1.

2.3 ACCOUNT-KARTE

Formaler Aufbau: ACCOUNT,accnr,password,SUP.

Alle unterstrichenen Angaben müssen angegeben werden.

Existiert zu einer Accountnummer ein Password und wird diese Nummer im Batch-Betrieb verwendet, so muß das Password auf der Accountkarte aufscheinen.

Beispiel: ACCOUNT,accnr,passw.

Terminalbenutzer, die im Batch-Betrieb ohne Password arbeiten wollen, müssen um eine zusätzliche Account-Nummer ansuchen.

Ist das automatisch erscheinende SYSBULL bereits bekannt und soll es für die weiteren Läufe unterdrückt werden, so hat die Accountkarte folgende Form:

ACCOUNT,accnr,passw,SUP.
oder
ACCOUNT,accnr,,SUP.

je nachdem, ob für diese Accountnummer ein Password existiert oder nicht.

3. MELDUNGEN DES SYSTEMS IM ZUSAMMENHANG MIT DEM ACCOUNTING

Bei formalen Fehlern auf der Accountkarte oder bei verbrauchten Betriebsmitteln wird im Dayfile eine der folgenden Meldungen ausgegeben.

3.1 Fehlermeldungen

Fehlermeldung	mögliche Ursache
INCORRECT SYNTAX IN ACCOUNT-CARD	fehlende Accountkarte (vertauscht?)
INCORRECT SYNTAX IN ACCOUNT-NUMBER	Accountnummer hat nicht die Form A99999999 A...Buchstabe 9...Ziffer
INCORRECT SYNTAX IN PASSWORD	Password hat nicht die Form AXXXXXXXX A...Buchstabe X...beliebiges Zeichen
INCORRECT SYNTAX IN SUP	Lochfehler
NO TERMINATOR ON ACCOUNT-CARD	Punkt oder schließende Klammer fehlt am Ende der Account-Karte
INVALID ACCOUNT-NUMBER	Accountnummer existiert nicht oder wurde nicht rechtzeitig verlängert
INCORRECT PASSWORD	falsches oder fehlendes Passwort
INCORRECT JOBCLASS	falsche oder fehlende Dependency
SS RESOURCE EXCEEDED	bewilligte Systemsekunden verbraucht
NO PERMISSION BLOCK MATCHES JOB-CARD PARAMETERS	Parameter der Jobkarte übersteigen vergebene Werte
RB LIMIT EXCEEDED	zuviele Recordblocks belegt

4.2 Warnungen

Warnung	mögliche Ursache
SS RESOURCE 90 % EXCEEDED	Systemsekunden bereits bis zu 90 % verbraucht

UP TO DATE UPDATE

STÖRUNGSTELEFON

Die Schwierigkeiten, die bisher bei der Installierung einer eigenen Störungsnummer aufgetreten sind, konnten nun endlich behoben werden.

DIE NEUE STÖRUNGSNUMMER DER CYBER 74 LAUTET:

65-43-50

Unter dieser Nummer können von 09.00 - 22.00 Uhr dem Rechenzentrum der TH-Wien alle Störungen bekanntgegeben werden. Wir bitten in diesem Zusammenhang die Kunden, im Falle einer Störung nicht mehr beim Konsoloperator anzurufen, da dies den Betrieb der Rechenanlage behindern würde.

SOFTWARE

SPRACHPROZESSOREN

ALGOL

GILT NUR FÜR CYBER 74

(.) Datum, Tageszeit:

Ab sofort steht auch ALGOL-Benutzern sowohl die Tageszeit als auch das Datum zur Verfügung. Dazu müssen folgende Pseudodeklarationen vorgenommen werden:

```
procedure date(x);integer x;fortran adate;
procedure time(x);integer x;fortran atime;
```

Nach dem Aufruf

```
z.B.: integer a,b;
      :
      date(a);time(b);
      :
```

kann der aktuelle Parameter (a bzw. b) entweder ausgedruckt (nur im Format 8A),

```
z.B.:  output(61,"("/,"("AUSFUEHRUNG AM ")",8A,"(" UM ")",8A,/")",a,b);

       Druckbild:
       AUSFUEHRUNG AM jj/mm/tt UM hh.mm.ss
```

oder als Integerzahl (eventuell für Sortiervorgänge) weiterverwendet werden.

(.) Achtung:

Beim Aufruf von FORTRAN-Unterprogrammen dürfen nur

einfache Variable und Feldnamen,

die entweder lokale Variable oder value-Parameter sein müssen, als aktuelle Parameter verwendet werden.

4.1 Externe PASCAL-Prozeduren:

4.1.1 Deklaration

```
procedure name(<parameterliste>); extern;
function  name(<parameterliste>); <type>; extern;
```

Diese PASCAL-Prozeduren (Funktionen) müssen getrennt vom Hauptprogramm übersetzt werden, und zwar in Form eines Hauptprogramms, das nur Prozedur(Funktions-)deklarationen (eventuell auch Typdeklarationen von Typen der Parameterlisten der Prozeduren(Funktionen)) und einen leeren Programmrumpf besitzt:

4.1.2 Beispiel

```
(*$E+       *)
program dummy(output);
     procedure 1;
     :
     procedure n;
     :
     function 1;
     :
     function n;
begin(*dummy body*) end.
```

Die Option E+ muß unbedingt verwendet werden (siehe Option-Beschreibung).

4.2 Externe FORTRAN-Prozeduren:

4.2.1 Deklaration

```
procedure name (<parameterliste>); fortran;
function  name (<parameterliste>); <type>;  fortran;
```

(.) Bei der Übersetzung eines FORTRAN-Unterprogramm-Aufrufes wird eine FTN-Calling-Sequence aufgebaut (Adresse der Parameterliste in A1, Abschluß der Parameterliste durch ein Nullwort).

(.) Achtung:

Unterschiede in der Speicherung von mehrdimensionalen Feldern in FORTAN (=spaltenweise) ergeben die Einschränkung, daß entweder

(..) vor der Übergabe von PASCAL-Feldern (zeilenweise Speicherung) an FORTRAN-Unterprogramme die Speicherreihenfolge geändert oder

(..) die zeilenweise Speicherung der PASCAL-Felder im FORTRAN-Unterprogramm berücksichtigt werden muß.

4.2.2 Pseudodeklaration für FORTRAN-Unterprogramme

Aufgrund einiger Inkompatibilitäten in der Darstellung gewisser Typen in PASCAL bzw. FORTRAN ergeben sich folgende Korrespondenzen zwischen FORTRAN-Unterprogrammen und entsprechender Pseudodeklarationen:

Parametertyp im FORTRAN-Programm	Parametertyp in der entspr. PASCAL-Pseudedeklaration	B e m e r k u n g
INTEGER	integer	bei var-Parametern(Call-by-reference) dieses Typs kann auch -Ø (minus Null) vom FORTRAN-Unterprogramm zurückgegeben werden. Dieser Wert sollte aber (durch Addition von Ø) im PASCAL-Programm eliminiert werden.
REAL	real	
DOUBLE	record	
COMPLEX	p1:real; p2:real	
	end;	
LOGICAL	integer	.TRUE. ... negativer Wert .FALSE... positiver Wert
Felder	array[..]of..	siehe auch Bemerkung in 4.2.1
SUBROUTINE	procedure	
FUNCTION	function	Das Resultat darf nicht vom FORTRAN-Typ COMPLEX oder DOUBLE sein! Bezüglich -Ø siehe obige Bemerkung.

5. Korrespondenzen formale-aktuelle Parameter in PASCAL-Prozeduren (Funktionen):

PASCAL unterscheidet zwischen

(.) var-Parametern (Call-by-Reference);

der entsprechende aktuelle Parameter muß eine Variable sein

und

(.) Value-Parametern (Call-by-Value);

der entsprechende aktuelle Parameter muß ein Ausdruck sein (wobei eine Variable ein Spezialfall davon ist).

Achtung: (.) Prozeduren(Funktionen) als aktuelle Parameter dürfen selbst nur Value-Parameter besitzen.

(.) Elemente von packed-Strukturen dürfen nicht als aktuelle Parameter verwendet werden!

6. Standard-Prozeduren und Funktionen

6.1 Filehandlingprozeduren:

put(f) schreibt die Buffervariable f↑ auf das File f; put(f) ist nur möglich, wenn eof(f) den Wert true hatte. Nach der Ausführung der Prozedur put(f) bleibt eof(f) true und f↑ ist undefiniert.

get(f) weist der Buffervariablen f↑ den Wert des nächsten Elements des Filesfzu. Ist kein Element mehr vorhanden, so wird eof(f) true; get(f) ist nur erlaubt, wenn eof(f) false war.

reset(f) weist der Buffervariablen f↑ den Wert des ersten Elements des Files f zu; eof(f) wird false, wenn das File f nicht leer ist, andernfalls wird f undefiniert und eof(f) bleibt true.

rewrite(f) ersetzt den laufenden Wert von f durch das Leerfile, eof(f) wird true und das File f kann (neu) erstellt werden.

page(f) veranlaßt den Drucker, eine neue Seite zu beginnen, bevor die nächste Zeile des Textfiles f ausgedruckt wird.

6.2 Prozeduren zur dynamischen Generierung bzw. Rückgabe von Speicherplatz.[1)]

new(p) weist der Pointer-Variablen p die Adresse der neu generierten Variablen v zu.(p ist vom Typ ↑v) Ist v vom Typ record mit Varianten, so kann die Form

new(p,t1,t2,..,tn) verwendet werden; (t1,..,tn) sind die jeweiligen Tagfield-Werte, die lückenlos in der richtigen Reihenfolge angegeben werden müssen.

dispose(p) gibt (den Speicherplatz für) die Variable v mit der Adresse p zurück.[2)]

dispose(p,t1,..,tn) wie oben (analog zur zweiten Form von new)

---

1) derzeit keine Freispeichersammlung (Garbage collection) implementiert.

2) wegen 1) keine echte Rückgabe.

6.3 Prozeduren für Daten-transfer

pack(a,i,z) Sei
a: array[m..n]of T und
z: packed array[u..v]of T

wobei n-m≥v-u dann
bewirkt pack(a,i,z):

```
for j:=u to v do z[j]:=a[j-u+i]
```

unpack(z,a,i) bewirkt

```
for j:=u to v do a[j-u+i]:=z[j]
```

6.4 Arithmetische Funktion

abs(x) (Real- o. Integer) Absolutwert der Real- oder Integervariablen x.

sqr(x) (Real- o. Integer)Wert des Quadrats der Real- oder Integervariablen x.

sin(x) Realwert des Sinus' der Real- o. Integervariablen x.

cos(x) Cosinus

arctan(x) Arcus tangens

exp(x) Exponentialfunktion

ln(x) Natürlicher Logarithmus

sqrt(x) Quadratwurzel

6.5 Bool'sche Funktionen

odd(x) x.. Integervariable
true... x ist eine ungerade Zahl
false.. x ist eine gerade Zahl

eoln(f) true... während des Lesens vom Textfile f wurde das Ende der Zeile erreicht.

eof(f) true... während des Lesens vom File f wurde das Ende des Files erreicht.

6.6 Andere Funktionen

trunc(x) x.. muß eine Realgröße sein. Das Ergebnis ist die größte Integerzahl $\leq x$ (für $x \geq \emptyset$) bzw.

die kleinste Integerzahl $\geq x$ (für $x < \emptyset$)

round(x) x.. muß eine Realgröße sein.

$$round(x) = \begin{cases} trunc(x+\emptyset.5) & \text{für } x \geq \emptyset \\ trunc(x-\emptyset.5) & \text{für } x < \emptyset \end{cases}$$

ord(x) Gibt die Positionsnummer der Variablen x innerhalb der Menge der Werte des Typs von x an.

```
Bsp.: typ t=(rot,blau,gelb,grün);
      var x:t;
          i:integer;
       x:=gelb;
       i:=ord(x);
```

so hat i den Wert 3.

chr(x) x muß eine Integergröße sein. Ergibt das Zeichen, das den Displaycode x hat.

succ(x) x sei ein beliebiger Skalartyp (ausgenommen real); das Resultat ist der Nachfolger von x (wenn vorhanden)

z.B.: succ(gelb) ergibt grün

pred(x) x...beliebiger Skalartyp (ausgenommen real); das Resultat ist der Vorgänger von x (wenn vorhanden)

z.B.: pred(blau) ergibt rot

7. Zusätzliche vordefinierte Prozeduren und Funktionen

7.1 Prozeduren

date(a) weist der alpha-Variablen a das laufende Datum zu.
Form: ␣jj/mm/tt␣

halt beendet die Ausführung des Programmes und gibt einen Post-Mortem-Dump aus

time(a) weist der alpha-Variablen a die momentane Uhrzeit zu.
Form: ␣hh.mm.ss.

linelimit(f,x) Beendet das Programm, wenn mehr als x Zeilen auf das Textfile f ausgeschrieben werden sollten. (x.. arithmetischer Ausdruck).

message(s) Bewirkt das Schreiben des Strings s ins Dayfile. (x sollte 40 Zeichen nicht überschreiten).

7.2 Funktionen

card(x) ergibt die Kardinalzahl der Menge (set) x, d.h. die Anzahl der Elemente der Menge.

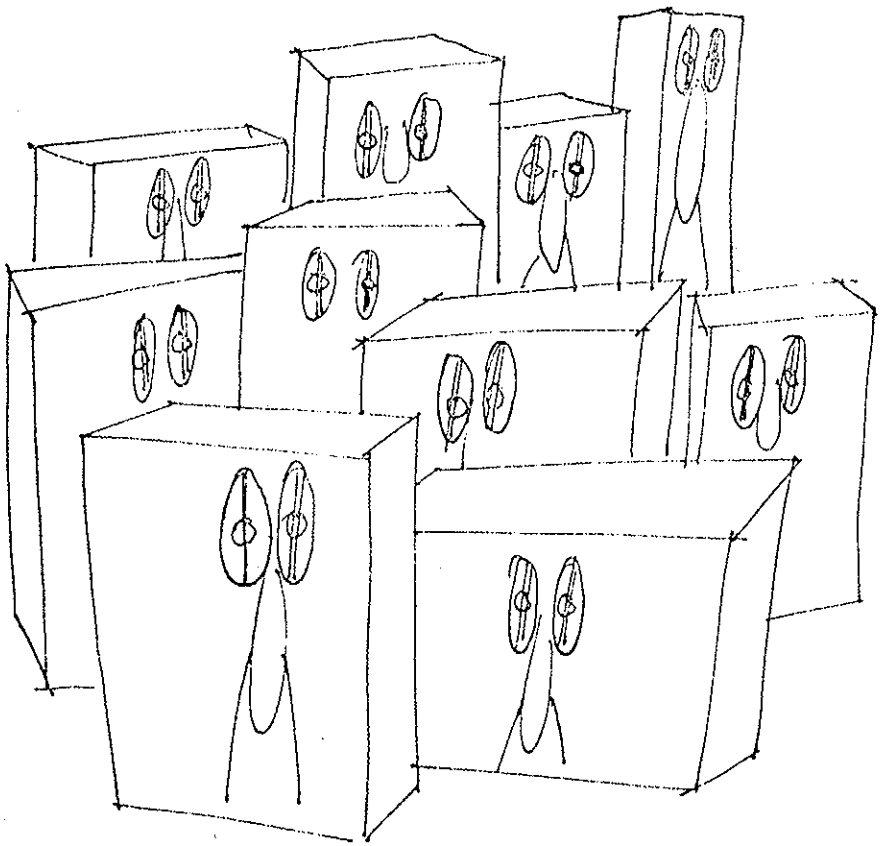
clock liefert die seit Jobbeginn vergangene CPU-Zeit in Millisekunden

expo(x) liefert den Exponenten der Gleitkommazahl x (Maschinendarstellung) als Integerwert.

undefined(x) liefert den Wert

true wenn der Realwert von x "out of range" oder "indefinit" ist,

false sonst.



Widerwillige Computer

SOFTWARE

GILT NUR FÜR CYBER 74

F O R T R A N

ÄNDERUNG VON PARAMETERN AUF DER FTN-KARTE

1) Die Standardwerte

für die Parameter R und A auf der FTN-Karte werden geändert!

1.1 Referencemap: neu:R=Ø

Der neue Standardwert soll den Papierverbrauch des Rechenzentrums verringern. Die Ausgabe einer Referencemap ist nur in ganz wenigen Fällen sinnvoll. Bevor Sie hier Papier verschwenden, sollten Sie eventuell die Programmberatung aufsuchen.

1.2 Job Abart: neu:A gesetzt

Treten bei der Übersetzung fatale Fehler auf, so wird nach der Compilation die Ausführung des Jobs abgebrochen (bzw. zur nächsten EXIT-Karte übergegangen).

Für die einfache

FTN.

Steuerkarte werden also jetzt folgende Werte angenommen:

FTN,B=LGO,L=OUTPUT,R=Ø,A,I=INPUT,PL=5ØØØ,OPT=1,OL=Ø,EL=1.

2) Neue Parameter

Wie bereits im letzten "Heißen Draht" angekündigt sind neue Parameter auf der FTN-Karte möglich.

2.1 Programmausführung GO

Ist der Parameter GO auf der FTN-Karte angegeben, so erfolgt nach dem fehlerfreien (keine fatalen Fehler) Übersetzen des Programms unmittelbar das Laden und die Ausführung des Programms ohne expliziten Loaderaufruf.

Die Steuerkarte

FTN,GO.

hat dadurch dieselbe Wirkung wie

FTN.
LGO.

2.2 Listparameter L,R,EL

a) L=lfn, das Programmlisting wird auf das File lfn geschrieben. Mit L=Ø kann das Listing unterdrückt werden (es werden nur mehr fatale Fehler auf das File OUTPUT geschrieben).

Umfang und Inhalt des Listings kann durch die folgenden Parameter bestimmt werden:

b) R=n Ø≤n≤3 bestimmt den Umfang der Referencemap

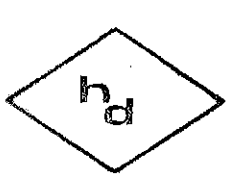
c) Der Parameter EL bestimmt den Umfang des Fehlerlistings (Error-Level)

EL=
- F nur fatale Fehler werden gelistet
- I zusätzlich werden auch informative Meldungen gelistet
- A zusätzlich werden auch NON-ANSI Meldungen ausgegeben

Beispiel:

```
FTN,L=XTAPE,EL=A.
```

Auf dem File XTAPE werden das Programm und alle Fehlermeldungen (FE,I,ANSI) gelistet, keine Referencemap.



MÄRZ 1975

dispose karte

Mit Hilfe einer Dispose-Karte kann vom Benutzer die Art der Ausgabe eines Files angegeben werden (stanzen, drucken auf verschiedenen Druckern).

Diese Zeilen sollen Ihnen die Anwendungen der Dispose-Karte erläutern.

1) DISPOSE für Ausgabefiles mit mehr als 150 000 Zeichen

Wenn Sie auf dem zentralen Drucker mehr als 150 000 Zeichen drucken wollen, müssen Sie eine DISPOSE-Karte verwenden.

```
 75/02/11 SCOPE 3.4.1 PSR 376 TH-WIEN   74/09/23
10.21.21.RZTTL3U  FROM **
10.21.21.IP  00000320 WORDS - FILE INPUT  , DC 00
10.21.21.RZTTL.   FTNLIBRARY TEST
10.21.28.ATTACH,FTTN,ID=
10.21.28.PFN IS
10.21.28.FTTN
10.21.28.PF CYCLE NO. = 002
10.21.30.EDITLIB.
10.21.46.FTN.
10.21.50.      .073 CP SECONDS COMPILATION TIME
10.21.50.LIBRARY,FTTN.
10.21.50.LGO.
10.21.53.
10.21.53.OP  00000640 WORDS - FILE OUTPUT , DC 40   <--
10.21.53.CPA       .989 SEC.          .989 ADJ.
10.21.53.IO       1.103 SEC.         1.103 ADJ.
10.21.53.CM      32.266 KWS.         1.968 ADJ.
10.21.53.SS                          4.062
10.21.53.PP       8.297 SEC.     DATE 75/02/11
10.21.53.EJ  END OF JOB, **


**********          RZTTL3U  //// END OF LIST ////
**********          RZTTL3U  //// END OF LIST ////
```

Die Zeile OP des Dayfiles gibt Ihnen den Umfang des Ausgabefiles in Worten an. Dabei enthält ein Wort 10 Zeichen. Sie dürfen also maximal 15 000 Worte ohne DISPOSE-Karte ausgeben.

Wenn Sie diesen Wert überschreiten, so verwenden Sie folgende Steuerkarte:

DISPOSE,lfn,PR=CØ1.

Diese Karte bewirkt, daß das File lfn (ohne Dayfile) nicht sofort gedruckt wird, sondern in eine besondere Warteschlange eingereiht wird. Sie erhalten unter Ihrem Jobnamen zwei Printouts, das Dayfile sofort und den eigentlichen Printout meist erst am nächsten Tag.

Einfaches Anwendungsbeispiel:

```
JOB,DAP77.          richtige Dependency nicht vergessen!
ACCOUNT,acctnr,pw,CHP.
FTN,PL=1ØØØØØ.
LGO.
DISPOSE,OUTPUT,PR=CØ1.
eor(7/8/9)

    FTN-Programm

eor(7/8/9)

    Daten

eof(6/7/8/9)
```

Wenn Ihr Job noch Fehler enthalten könnte, verwenden Sie die Steuerkartenfolge:

```
.
.
.
LGO.
DISPOSE,OUTPUT,PR=CØ1.
EXIT,S.
DISPOSE,OUTPUT,PR=CØ1.
eor(7/8/9)
.
.
.
```

2) Drucken eines Files auf dem zentralen Drucker mit speziellen Wünschen

Bei Abarbeitung der Dispose-Karte wird das File lfn in eine spezielle Warteschlange eingeordnet und mit einer Kennzeichnung versehen. Dafür dienen die Steuerkarten:

```
DISPOSE,lfn,PR=C.
```

bzw.:

```
DISPOSE,lfn,PR=Cnn.
```

für nn sind außer Ø1 (mehr als 150 000 Zeichen OUTPUT) noch die Werte:

Ø2	weißes Papier
Ø4	persönliche Abholung des Printouts
Ø8	8 Zeilen/Zoll
Ø9	Printout nicht falten

möglich.

3) Drucken eines Files auf der Datenstation im Hauptgebäude

Files können auch auf der Datenstation gedruckt werden. Diese Druckfiles dürfen allerdings nicht mehr als 150 000 Zeichen enthalten. Längere Files müssen auf dem zentralen Drucker gedruckt werden.

Verwenden Sie für die Ausgabe auf der Datenstation die Steuerkarte

```
DISPOSE,lfn,PR=IAB.
```

Jobs, die auf der Datenstation eingelesen werden, werden automatisch auch auf der Datenstation ausgegeben.

4) Outputqueues in Intercom

Die obigen Dispose-Karten lassen sich natürlich auch in Intercom verwenden, darüber hinaus können Sie mit

```
DISPOSE,lfn,PR=Ixx.
```

das File lfn in die Outputqueue des Terminals xx (UserId) einordnen