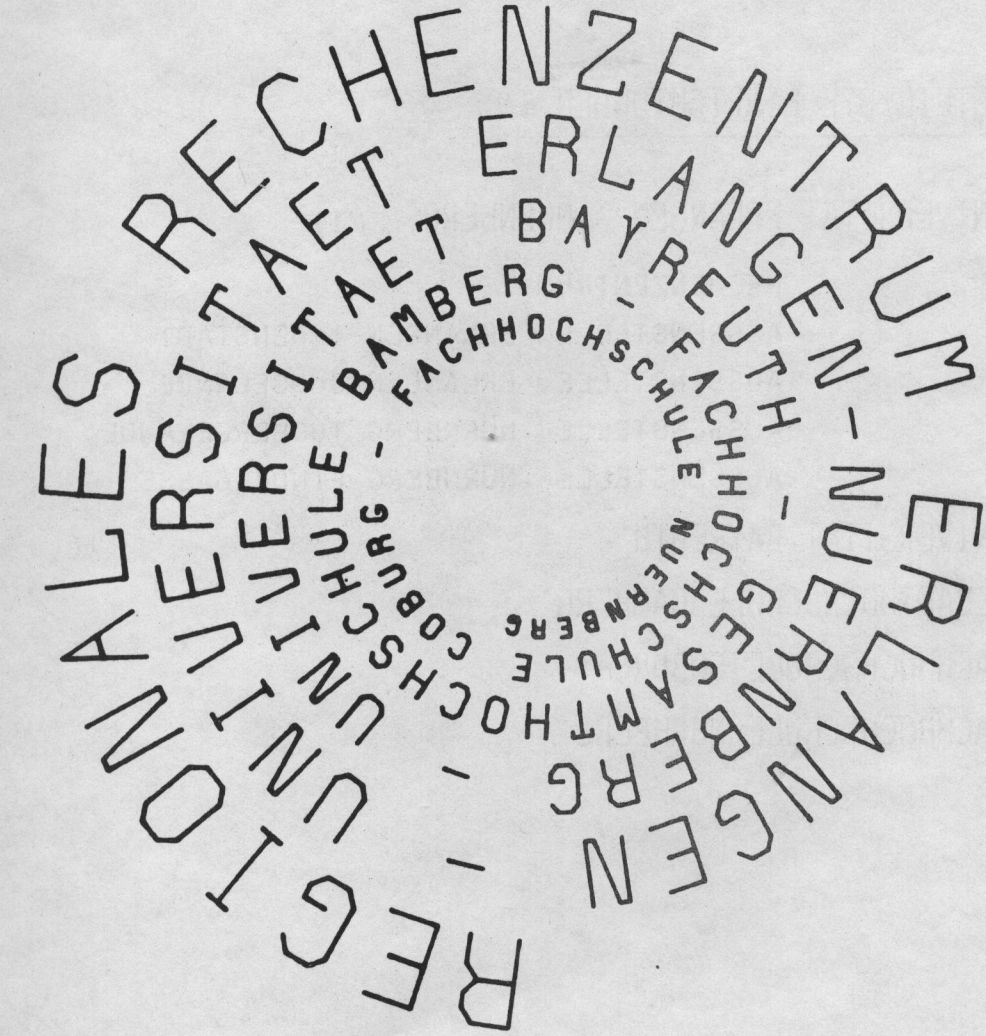
R R Z E
-G. Büttner-
Martensstr. 1

8520 Erlangen

RRZE

BENUTZERINFORMATION



BI 19 -ERLANGEN- 3. DEZEMBER 1979

R R Z E
REGIONALES RECHENZENTRUM
MARTENSSTRASSE 1

852o ERLANGEN

TEL: o9131 / 85 7o 31 - 85 7o 32

BETEILIGTE EINRICHTUNGEN :

UNIVERSITÄT ERLANGEN - NÜRNBERG MIT

RECHENZENTRUM
AUSSENSTELLE ERLANGEN INNENSTADT
AUSSENSTELLE ERLANGEN SÜDGELÄNDE
AUSSENSTELLE NÜRNBERG TUCHERGELÄNDE
AUSSENSTELLE NÜRNBERG FINDELGASSE

UNIVERSITÄT BAYREUTH

GESAMTHOCHSCHULE BAMBERG

FACHHOCHSCHULE COBURG

FACHHOCHSCHULE NÜRNBERG

HERAUSGEGEBEN VOM REGIONALEN RECHENZENTRUM ERLANGEN

Inhaltsverzeichnis

1 **Aktuelle Informationen** 2

1.1 Benutzerkolloquium 2
1.1.1 Einladung zum Benutzerkolloquium am 12.2.1980 2
1.1.2 Protokoll des Benutzerkolloquium am 13.11.1979 2

1.2 Kommando UMBENENNE am TR440 3

1.3 BMDP/CYBER 3

1.4 Plotteranschluß in PASCAL (CYBER) 4

1.5 Asynchronanschlüsse ohne STX 4

1.6 Magnetbandbearbeitung im Dialog (TR440) 5

2 **Neues Cyber-Betriebssystem: NOS Release 5** 7

2.1 Änderungen bei Systemkommandos 7
2.1.1 Dezimalangaben bei Zeit-und SRU-Grenzwerten 7
2.1.2 CATLIST-Ausgabe 7
2.1.3 Erweiterungen bei REWIND,RETURN,CLEAR 7

2.2 Änderungen bei Telex-Kommandos 7
2.2.1 TIME-LIMIT und SRU-LIMIT Meldung 8
2.2.2 Dumps/Exchange Package 8
2.2.3 ENQUIRE-/STATUS-Kommando 8
2.2.4 XEDIT 8
2.2.5 Neues Subsystem FORTRAN 8
2.2.6 CYBER-INTERACTIVE-DEBUG 8

2.3 Änderungen der Compiler 9
2.3.1 ALGOL 5 9
2.3.2 BASIC 9
2.3.3 COBOL 5 9
2.3.4 FTN4 9
2.3.5 FTN5 10

2.4 Änderungen des CYBER-RECORD-MANAGER 10

2.5 Neuer Compiler für FORTRAN 77 10

2.6 Konvertierungsprogramm F45 12

1 Aktuelle Informationen

1.1 Benutzerkolloquium

1.1.1 Einladung zum Benutzerkolloquium am 12.2.1980

Das nächste Benutzerkolloquium findet am Dienstag, dem 12.2.1980 um 16 hct im H4 statt. Alle Benutzer des RRZE sind dazu herzlich eingeladen. Bitte besprechen Sie Ihre Fragen möglichst vorher mit einem Benutzervertreter oder mit der Leitung des RRZE.

1.1.2 Protokoll des Benutzerkolloquium am 13.11.1979

Tagesordnung:
1. Bericht des RZ-Leiters
2. Stand der Erweiterung
3. Neue Betriebssystemversionen
4. Anwendersoftware
5. Sonstiges

Top 1: Bericht des Rechenzentrumsleiters (Dr. Wolf)

Dr. Wolf gab eine Übersicht über den Zustand und die Auslastung der beiden Rechenanlagen im 3. Quartal 1979. Als Vergleichswerte wurden die Daten vom gleichen Zeitraum des Jahres 1978 herangezogen. Im einzelnen ergaben sich folgende Werte:

	Berichtszeitraum 3.Quartal 1979		Vergleichszeitraum 3.Quartal 1978	
	Cyber	TR440	Cyber	TR440
CPU-Stunden	1482	2973	1461	1625
Betriebszeit (h)	2115	1558	1952	1379
Auslastung (%)	68	64	75	39
Ausfallrate (%)	3	10	3	10
Mittl. Fehlerabst. (h)	88	42	70	40

Top 2: Stand der Erweiterung (Dr. Wolf)

Der Erweiterungsantrag des RRZE wurde von der DFG positiv entschieden, so daß u.a. nunmehr die Cyber 172 zur Cyber 173 ausgebaut werden kann. Die Finanzierung erfolgt über das Regionalprogramm, wobei die Bundesmittel im Haushalt 1979 des BMFT bereitgestellt werden. Die Zustimmung des Staatsministeriums für Unterricht und Kultus gilt als relativ sicher.

Mit der Installation der Erweiterungen wurde unter dem Vorbehalt der endgültigen Genehmigung bereits begonnen, da die Arbeiten zum Jahresende abgeschlossen sein müssen. Genaueres über Art und Umfang der Erweiterungen ist in BI 18 nachzulesen.

Im Rahmen der Umrüstung werden sich Störungen im Rechenbetrieb nicht vermeiden lassen. Es empfiehlt sich also für die nächste Zeit besonders, die Anschläge und Rundschreiben zu beachten!

Top 3: Neue Betriebssystemversionen (Seybold, Müller, Zeitler)

Hiervon sind in erster Linie die Benutzer der Cyber betroffen: Mit "Release 5" wurde von CDC eine neue Version des Betriebssystems sowie aller Compiler ausgeliefert. Damit die Überraschung nicht zu groß wird (vgl. ähnliche Aktionen in der Vergangenheit!), werden die Benutzer gebeten, von der Möglichkeit, das System vorab zu testen, regen Gebrauch zu machen.

Nähere Einzelheiten zu "Release 5" sind BI 18 und 19 zu entnehmen. Ansonsten wird, auch was die Modifikationen an der TR betreffen, auf die nächsten Benutzer-Informationen verwiesen.

Top 4: Anwendersoftware (Cramer)

Das hierzu Gesagte kann ebenfalls in BI 18 nachgelesen werden.

Top 5: Sonstiges

Als Termin für das nächste Benutzerkolloquium wurde der 12.2.1980 festgelegt.

1.2 Kommando UMBENENNE am TR440

Das Kommando wurde aufwärtskompatibel geändert. Ab sofort ist es möglich, mittels UMBENENNE den Koordinationstyp einer Datei zu verändern, d.h. aus einer P-Datei (private Datei) eine G-Datei (Gemeinschaftsdatei) zu machen und umgekehrt.

Syntax des Kommandos:

```
 #UMBE,DATEI=Dateialt´Neudatei,PROGRAMM=...
Neudatei:= [(KT)] [Dateineu]
KT:= P oder G
Dateineu:= wie im TR440 Kommandobuch beschrieben
```

Beispiel: #UMBE,Datei=QUARK´(G)

1.3 BMDP/CYBER

Die BMDP-Programme und die Kommandoprozedur BIMED wurden so modifiziert,daß es jetzt möglich ist, in einem BMDP-Lauf Daten von einer Datei einzulesen und einen BMDP-File zu erzeugen.

Erweiterter BMDP-Aufruf:

POP,BIMED.PROG=bmdpxy,SPACE=ib,TRANSF=lfn1,FUN=lfn2,CON=lfn3, TRANS1S=lfn4, INPUT=lfn5,OUTPUT=lfn6,DATA=lfn7,SAVE=lfn8

Die Prozedur BIMED wurde um den Parameter SAVE erweitert.

Bei der Dateneingabe (Rohdaten oder BMDP-File) von einem File ist DATA=lfn7 und im INPUT-Paragraphen UNIT=10 zu setzen, bei der Erzeugung eines BMDP-Files muß SAVE=lfn8 und im SAVE-Paragraphen UNIT=20 sein. lfn7 und lfn8 müssen verschiedene lokale Files sein, die UNIT-Nummern müssen eingehalten werden. Die Ausgabe eines BMDP-Files mit eigener Fortran-Transformation erfolgt mit SAVE=lfn8 und WRITE(20, ...) ohne SAVE-Paragraphen. Die Ausgabe in eigenem Format mit

gleichzeitiger Erzeugung eines BMDP-Files ist nicht möglich.

1.4 Plotteranschluß in PASCAL (CYBER)

Für den neuen PASCAL-Compiler an der CYBER wurde die Möglichkeit geschaffen, das Erlanger Grafik-System (EGS) anzuschließen. Es ist dazu lediglich der Aufruf einer zusätzlichen FORTRAN-Routine FTNPLT vor Beginn des Zeichnens nötig; sie stellt u.a. den File GRAFIK zur Verfügung. Nach Abschluß der Zeichnung wird der Aufruf der FORTRAN-Routine FTNOUT empfohlen, diese schreibt EGS-Fehlermeldungen (falls vorhanden) auf den File FTNOUT.

Hier die Skizze eines zeichnenden PASCAL-Programms:

```
JOB.
USER,...
CHARGE,...
ATTACH,EGSLIB/UN=UNRZPB.
POP,PASCALI.
PASCAL.
LIBRARY,EGSLIB,PASCLIB. (Reihenfolge der Bibliotheken wichtig!!)
LGO.
SAVE,GRAFIK.
REWIND,FTNOUT.
COPY,FTNOUT.
POP,GRPOST.GER=PR8
-EOR-
PROGRAM PASCALEGS(OUTPUT);
....
PROCEDURE FTNPLT;FORTRAN;
PROCEDURE FTNOUT;FORTRAN;
PROCEDURE PLOTCL;FORTRAN;
PROCEDURE DINAH(I:INTEGER);FORTRAN;
PROCEDURE MOVE(X,Y:REAL);FORTRAN;
....
BEGIN
    FTNPLT;
    DINAH(4);
    ....
    PLTCL;
    FTNOUT
END.
-EOI-
```

1.5 Asynchronanschlüsse ohne STX

Seit dem 13.11.79 gibt es ein neues DNSP (Datenübertragungs- und Netzsteuerprogramm im DFV-Rechner DUET).

Die Behandlung der Asynchron-Anschlüsse wurde wie folgt abgeändert:

1) Das Kennzeichnen des Beginns einer Eingabe mit dem Steuerzeichen

STX (Tasten CTRL + B) ist nicht mehr nötig. Statt wie bisher die Folge

(STX)... text ...(ETX)

einzugeben, genügt jetzt allein die Folge

... text ...(ETX) .

2) Ausnahmen:
   - Die leere Eingabe muß auch jetzt in der alten Form:

     (STX) (ETX)

     eingegeben werden.
   - Wenn eine Eingabe mit dem Steuerzeichen ACK beginnen soll, muß weiterhin eine STX vorangestellt werden:

     (STX) (ACK)...text...(ETX),

     d.h. ACK als erstes Zeichen bedeutet nicht den Beginn einer Eingabe. Diese unbedeutende Einschränkung ist nötig, da die beiden im Rechenzentrum aufgestellten Diablo-Typenraddrucker auf ein ETX, das sie von der DUET empfangen, mit ACK antworten, und dadurch die Ausgabe unterbrochen würde.

3) Der Benutzer kann wie bisher auch mit vorangestelltem STX arbeiten, muß jedoch beachten, daß bisher alle Eingabezeichen vor dem STX weggeworfen wurden, dise jetzt aber schon mit zum Eingabetext genommen werden.

Wir hoffen, daß diese Änderung Ihnen den Umgang mit dem TR440 erleichtert und bitten Sie, sich bei Fehlern und Schwierigkeiten mit möglichst genauer Beschreibung (Hardcopy des Fernschreibers etc.) an die Aufsicht zu wenden.

1.6 Magnetbandbearbeitung im Dialog (TR440)

Die Bearbeitung von Magnetbändern im Dialog ist jetzt möglich geworden. Beachten Sie bitte die folgenden Hinweise:

1) Die Geräteplanung wurde aus der KFK entfernt, und in den BEN&VW gelegt. Planungsalgorithmus:

   - Die angeforderten Geräte müssen physikalisch vorhanden sein
   - Die Summe der Geräteanforderungen > 1 muß erfüllbar sein

   Der Planungsalgorithmus wird bei den Kommandos XBA, XBG und BEDARF durchlaufen. Ist das angeforderte Gerät physikalisch nicht vorhanden, so erfolgt eine Fehlermeldung; trifft der 2. Teil der Bedingung nicht zu, so erfolgt im Gespräch eine Fehlermeldung, im Abschnitt wartet der Auftrag auf das Freigeben der Berechtigungen durch andere Aufträge.

2) Zuteilung von Geräten (Band aufspannen) ist ab sofort auch im Dialog möglich; d.h. die Kommandos:

   KOPIERE,...INFORMIERE,...EINSCHLEUSE,..etc

   mit Träger =B60(...) sind von Konsolen aus absetzbar.

   Tritt bei der Zuteilung ein Geräteengpaß auf, so erfolgt im Dialog eine Fehlermeldung (FS ´5B´ =DIE ANGEFORDERTEN BETRIEBSMITTEL SIND MOMENTAN NICHT EINPLANBAR); im Abschnitt wartet der

Auftrag auf das Freiwerden von Geräten, die von anderen Aufträgen belegt sind. Der gleiche Fehlerschlüssel wird auch im Gespräch zurückgeliefert, wenn das EXDKZ von einem anderen Auftrag belegt ist.

3) Der Entschlüssler (PS&ENTSCHL) setzt bei jeder Konsolausgabe

GIB KOMMANDOS:

einen SSR ab, der das Abspannen (Freigeben) aller von diesem Auftrag belegten Bandgeräte bewirkt; damit wird das ungewollte Belegen eines oder mehrerer Bandgeräte verhindert. Braucht man im Dialog mehrmals dasselbe Band, so empfiehlt es sich, als FKZ im XBG-Kommando sechs Gleichheitszeichen (FKZ=======) anzugeben; damit wird erreicht, daß das Band logisch freigegeben, aber nicht abgespannt wird. Man erspart damit den Operateuren viel Laufarbeit!

Da durch die Angabe FKZ======= das Band auch bei Abschnittsende nicht abgespannt wird, kann ein Folgeauftrag, der mit demselben Band arbeiten möchte, ohne Operateureingriff sofort darauf zugreifen.

4) Sind Gerätenachforderungen mittels Kommando BEDARF im Abschnittsbetrieb momentan nicht erfüllbar und hat der Auftrag schon ein Gerät zugeteilt bekommen, so erfolgt auch hier ein Abbruch des Kommandos (Potentieller Deadlock).

2 Neues Cyber-Betriebssystem: NOS Release 5

2.1 Änderungen bei Systemkommandos

2.1.1 Dezimalangaben bei Zeit-und SRU-Grenzwerten

Die Angaben bzw. Kommandos
Tnnnn auf der Jobkarte
SETASL,nnnnn.
SETJSL,nnnnn.
SETTL,nnnnn.
T,nnnnn bei TIME-LIMIT in Telex
S,nnnnn bei SRU-LIMIT in Telex
bedeuten jetzt eine dezimale Sekunden-bzw. SRU-(Pfennig-) Angabe.
Beispiel: Jobkarte: BEI,T600. 10 Minuten Rechenzeit
Voreinstellung ist nach wie vor 64 Sekunden.
Eine oktale Angabe kann durch nnnnnB verwendet werden. Die Feldlängenangabe CMnnnnnn ist nach wie vor oktal, außer wenn die Ziffer 8 oder 9 vorkommt oder ein D angehängt wird.

2.1.2 CATLIST-Ausgabe

Bei CATLIST. kommt eine alphabetisch sortierte Liste der File-Namen, die nach Indirect- bzw. Direct-Access unterteilt ist, sowie die Angabe des Gesamtplatzverbrauches.

Bei CATLIST,LO=F. wird die Länge der einzelnen Files in PRU´s angegeben. Ein PRU enthält 640 Zeichern = 64 Worte
16 PRU´s sind 10240 Zeichen = 1024 Worte (1K).

2.1.3 Erweiterungen bei REWIND,RETURN,CLEAR

Das Kommando
REWIND,*,lfn1,lfn2,...
bewirkt das Rückspulen aller Files des Jobs mit Ausnahme der angegebenen Files lfn1,...
Analog wirken die beiden folgenden in ihrer Wirkung identischen Kommandos:
RETURN,*,lfn1,lfn2,...
CLEAR,*,lfn1,...
Die Ausnahmeliste kann bei diesen Kommandos auch leer sein,
z.B. REWIND,*. oder RETURN,*.(= CLEAR,*. = CLEAR.)

2.2 Änderungen bei Telex-Kommandos

2.2.1 TIME-LIMIT und SRU-LIMIT Meldung

Jetzt mit zweiter erklärender Textzeile. Die Eingabe ist auf T,nn bzw. S,nn oder die leere Eingabe beschränkt. Jede andere Eingabe führt zur Wiederholung der Meldung.

2.2.2 Dumps/Exchange Package

Diese werden auf den File ZZZDUMP geschrieben, wenn der OUTPUT-File ein Terminal ist.

2.2.3 ENQUIRE-/STATUS-Kommando

Etwas erweiterte Ausgabe, ebenso bei leerer Eingabe und aktivem Programm.

2.2.4 XEDIT

- Aufruf jetzt in allen Subsystems möglich (nicht nur im BATCH).
- Bei Aufruf mit nicht existierendem File und weggelassenem c-Parameter nicht mehr "EMPTY OR FILE NOT FOUND/NAME EDIT FILE?" sondern "EMPTY FILE/CREATION ASSUMED". Falls der Filename vertippt war, ist XEDIT mit "STOP" zu beenden und neu aufzurufen.
- Die Befehlsfolge
  ?? del $
  ?? I $ einzufügende Zeile
  bewirkt nicht mehr das gewünschte, der INPUT-Befehl ist jetzt in der Form
  ??+I $ einzufügende Zeile
  einzugeben.

2.2.5 Neues Subsystem FORTRAN

Für FTN5 gibt es ein neues Subsystem FORTRAN. Das Subsystem FTNTS aktiviert nach wie vor den FTN4-Compiler (mit TS Option)

2.2.6 CYBER-INTERACTIVE-DEBUG

Unterstützt jetzt auch quellsprachenbezogen BASIC-und FTN5-Proramme.
Bekannte Änderungen:
- CID-Kommando SAVE bewirkt kein Rewind mehr.
- Das interne Format des ZZZZZDT-Files wurde geändert.

2.3 Änderungen der Compiler

Genaue Angaben zu den Änderungen der einzelnen Compiler finden Sie in der jeweils neuesten Fassung der entsprechenden Reference Manuals.

Zur Zeit sind noch nicht bei allen Manuals die neuesten Fassungen vorhanden. Das erste Exemplar wird immer in der Beratung zugänglich gemacht, und kann dort eingesehen bzw. erfragt werden.

2.3.1 ALGOL 5

Enthält aufwärtskompatibel neue Features(insbesondere Control-Card-Parameter). Alte Binärdecks können nicht weiterverwendet werden.

2.3.2 BASIC

Enthält aufwärtskompatibel erweiterte String-Handling-Möglichkeiten.

2.3.3 COBOL 5

Enthält aufwärtskompatibel neue Features(in Richtung des vorgeschlagenen neuen Standards).

2.3.4 FTN4

Ist nach wie vor vorhanden für Programme in FORTRAN 66.

Es enthält jetzt einen Post Mortem Dump. Die dafür notwendigen Tabellen werden generiert, wenn der neue Steuerkartenparameter ´PMD´ beim FTN-Aufruf angegeben wird. Beim Programmstart ist dann anzugeben:
LGO,*OP=option bzw. LGO,file1,...,filen,*OP=option.
wobei option sein kann:
*OP=T: Kurzform des Dumps am Terminal
*OP=F: Voller Dump auf File PMDUMP(Voreinstellung)
*OP=A: Auch Programme,die nicht in der aktuellen
Aufrufverschachtelung sind werden gedumpt.
Die Kombinationen OP=TA bzw.OP=FA sind möglich. Der Post Mortem Dump kann bei OVERLAY-Programmen benutzt werden, jedoch nicht bei Segment-Programmen. Weitere Einzelheiten stehen im FTN-Reference-Manual bzw. im Anhang dieser BI (bei POST MORTEM DUMP PARAMETERS auf dem EXECUTION CONTROL STATEMENT von FTN5).

2.3.5 FTN5

Neuer Compiler für den neuen FORTRAN 77-Standard. (s.u.)

2.4 Änderungen des CYBER-RECORD-MANAGER

Die Advanced-Access-Methods enthalten größere Änderungen, insbesondere bei den Dateiorganisationen Direct-Access und Actual-Key. Genaue Hinweise bitte in der Beratung erfragen. Die Änderungen betreffen nicht die Basic-Access-Methods (BAM), die im allgemeinen verwendet werden.

2.5 Neuer Compiler für FORTRAN 77

a) Seit April 1978 gibt es eine neue FORTRAN-Norm. Dieser Standard wird als FORTRAN 77 bezeichnet.

Die wesentlichen Änderungen von FORTRAN 77 gegenüber FORTRAN 66 sind:
- neuer Datentyp CHARACTER
- PARAMETER-Anweisung
- Wegfall vieler Beschränkungen (Mixed Mode, Indexausdrücke)
- Kontrollstruktur IF-THEN-ELSE
- Erweiterte DO Loops (Real-Laufvariable, rückwärtslaufende Schleifen, Ausdrücke für Anfangs-und Endwert)
- endlich genormte neue Möglichkeit der Ein-/Ausgabesteuerung (u.a. Random Zugriff, Dateiendeabfrage, dynamisches Zuordnen von Kanalnummer und Datei)

Eine Erläuterung der neuen Sprachmöglichkeiten ist in der BI 16 vom 12.2.1979 gegeben worden.

In der neuen Betriebssystemversion ist ein Compiler für FORTRAN 77 vorhanden. Er wird mit
FTN5,parameter.
aufgerufen.

Eine komplette Beschreibung der Parameter findet sich im Anhang, ebenso eine Zusammenfassung der Sprachsyntax von FTN5. Das FTN5-Manual ist im Druck und wird Mitte Dezember in der Aufsicht des RRZE erhältlich sein.

b) Sehr viele FTN4-Programme können unverändert vom FTN5-Compiler übersetzt werden, vor allem diejenigen in denen auf maschinenspezifische Möglichkeiten und Tricks verzichtet wurde. Es gibt ein Umstellungsprogramm F45 (s.u.), das die Anweisungen in FTN4-Programmen umstellt bzw. anmerkt, die vom FTN5-Compiler nicht verarbeitet werden können.

Der FTN5-Compiler umfaßt nicht nur den FORTRAN 77-Standard, sondern er enthält auch etliche Extensions, die auch im FTN4 vorhanden waren um die Umstellung zu erleichtern, auch wenn solche Extensions durch die neuen Elemente von FORTRAN 77 überflüssig wären(z.B. ENCODE, EOF(kan)).

c) Vorsicht ist geboten bei Verwendung von **FTN4-Programmbibliotheken** (NAGLIB,EGSLIB) zu FTN5-Hauptprogrammen.

Dies geht nur, wenn die Bibliotheksprogramme weder Ein-/Ausgabebefehle noch Bezüge auf diese enthalten. Auf jeden Fall ist folgende Ladersequenz strikt einzuhalten(LDSET-Karten unbedingt am Anfang):

```
LDSET,LIB=userlib/....
LDSET,LIB=EGSLIB/.... (benötigte FTN4-Bibliotheken in der
                      richtigen Hierarchie)
LOAD,LGO. (FTN5-Programme)
LOAD,divfs (eventuell weitere FTN5-Programme, z.B. EGSFTN5)
....
NOGO(ABS) bzw. EXECUTE
```

Bei der Benutzung der EGSLIB ist der Laderbefehl
LOAD,EGSFTN5.
einzufügen.

Diese Datei muß vorher mit
GET,EGSFTN5/UN=UNRZPB.
geholt werden; sie enthält die FTN5-Fassung der ein-/ausgebenden Teile der EGSLIB.

Welche Programme in den anderen UNRZPB-Programmbibliotheken unverträglich sind, ist noch nicht vollständig ausgetestet worden. Bitte melden Sie auftretende Schwierigkeiten in der Beratung.

2.6 Konvertierungsprogramm F45

a) Dieses Programm dient dazu, Programme in FTN4 auf FTN5 umzustellen. Gleichzeitig wird eine Liste erzeugt, die die geänderten Zeilen angibt, sowie Hinweise auf Statements gibt, die von Hand umgestellt oder überprüft werden müssen.

Als Ausgabe kann gewählt werden:
- das volle umgestellte Programm oder
- nur die umgestellten Zeilen
- die umgestellten Zeilen zusammen mit passenden Update-Directiven

Enthält der Eingabefile Zeilen, die länger als 80 Zeichen sind (COMPILE-File), so wird stets die letzte Ausgabeart gewählt.

Sowohl die Eingabe als auch die Ausgabe können unabhängig voneinander in Standard-Form oder Sequenced-Format (Parameter SEQ bei FTN, Zeilennummern vorne) erfolgen.

b) Die Parameter von F45 sind:

CI=idname	*IDENT idname bei UPDATE-Korrekturen
I=lfn1	Eingabedatei, Voreinstellung INPUT
I	bedeutet I=COMPILE
L=lfn2	Protokolldatei, Voreinstellung OUTPUT
LO	Es werden nicht nur die Änderungsmeldungen sondern auch die alte Fassung und die Ausgabedatei auf lfn2 gedruckt.
MD	Maschinenabhängige Anweisungen werden gekennzeichnet
P=lfn3	Ausgabedatei, Voreinstellung PUNCH
PO=S	Ausgabesteuerung, nur die umgestellten Zeilen werden auf lfn3 ausgegeben.
PO=F	das volle Programm wird ausgegeben (außer bei COMPILE-File-Eingabe, s.o.)
PO=M	die umgestellten Zeilen werden zusammen mit den UPDATE-Directiven ausgegeben.
SI	Eingabedatei ist in Sequenced-Format, Voreinstellung: je nach Aussehen der ersten Karte der Eingabedatei
SI=0	Eingabedatei ist in Standard-Format
SO	Ausgabedatei ist in Sequenced-Format, Voreinstellung: gleich der Eingabedatei
SO=0	Ausgabedatei ist in Standard-Format

c) F45 macht automatisch u.a. folgende Umstellungen:
(* bedeutet: evtl. noch Umstellung per Hand)

Eingabe Ausgabe

Daten und Ablaufsteuerung

multiple Statements	Einzelzeilen
Fortsetzungskarten nach Leerzeilen	Einzeilzeilen
Oktalkonstanten nnnB	O"nnn"
Long-Hollerith-Konstanten	evtl. neue Syntax *,auf CHARACTER umstellen
Exponentiation A**B**C	(A**B)**C
komplexe Operanden in Vergleichsausdrücken	evtl. wird REAL(.) eingefügt
Logische Abkürzungen z.B. .A.	werden ergänzt .AND.
Minuszeichen vor Oktal- oder Hollerithkonstanten -nnnB	in DATA Compile-Time-Komplementbildung bzw. .NOT. O"nnn"
Numbered Common Block	wird Named Common Block
DATA Statements	evtl. neue Syntax
Computed GOTO	CALL GOTOER wird dahinter eingefügt
Arithmetisches IF mit komplexem Ausdruck	REAL(.) wird eingefügt
Two-Branch-IF	wird vervollständigt
END-Statement	wird falls fehlend eingefügt
C$-Debug Statements	werden entfernt

OUTPUT-Statements

WRITE fn,iolist	PRINT fn, iolist
PRINT(u,fn)iolist	WRITE(u,fn) iolist
PUNCH(u,fn)iolist	WRITE(u,fn) iolist
PRINT*,4HTEST oder PRINT*,"TEST"	PRINT*,´TEST´
Redundante Klammern in i/o-list	werden entfernt

Format-Spezifikationen

weggelassene Kommas	werden eingefügt
-nX	TLn
Ew.dDe	Ew.dEe
"TEXT" oder *TEXT*	´TEXT´
T0	T1

READ-Statements

READ(u,f) iolist	READ(u,fn,END=n) iolist n .... *)n auf das Sprungziel der folgenden EOF-Abfrage umstellen (EOF(u) geht zur Zeit noch)

Prozeduren

alternate RETURN	neue Syntax bei SUBROUTINE,RETURN und CALL
Dummy Arguments bei ENTRY	werden analog zu SUBROUTINE-Statement eingefügt
Namenskonflikte mit neuen Intrinsic Functions	EXTERNAL... wird eingefügt
Zufallszahlengenerator RANF (DUMMY)	RANF()

d) Folgende Sprachkonstrukte in FTN4 bedürfen der Überarbeitung per Hand, darauf wird teilweise durch F45-Meldungen hingewiesen:

- DO-Loops mit Endwert .LT. Anfangswert (werden in Fortran 77 null mal ausgeführt)
- Extended Range bei DO-Loops (in Fortran 77 verboten)
- List-Directed Input, bei dem durch mehrere READ-Statements von derselben Zeile gelesen werden soll
- List-Directed Output (produziert etwas anderes Listenformat)
- Hollerith-Data in INTEGER-Variablen bei List-Directed-Output
- Variables Format in einzelnen INTEGER-Variablen
- A-und O-Format bei komplexen Variablen
- Eingabe im H-Format
- ´V´- und ´=´ Format-Spezifikationen
- variables Format (wie bei c) umstellen)
- Intrinsic Functions als aktuelle Parameter (teilweise)
- Typenkonflikte beim Aufruf von Statement-Functions
- Prozeduren SLITE und SLITET
- Aufruf von DATE, JDATE, TIME, CLOCK mit CALL (jetzt parameterlose Function vom Typ CHARACTER)
- Aufruf von SECOND mit CALL (Jetzt parameterlose Function vom Typ REAL)
- Prozedur IOCHEC

This section describes the FTN5 control statement options, the FORTRAN output listing formats, and the execution control statement.

FTN5 CONTROL STATEMENT

The FORTRAN compiler is called from the system library and executed by an FTN5 control statement. The FTN5 control statement calls the compiler, specifies the files to be used for input and output, and indicates the type of output to be produced. This control statement can have any of the formats shown in figure 11-1. Some examples of FORTRAN control statements are as follows:

```
FTN5(ET=W,LO=R,S=0)

FTN5,I=INF,L=LIST.
```

Format 1:

```
FTN5(p[,p] . . . )
```

Format 2:

```
FTN5.
```

Format 3:

```
FTN5,p[,p] . . . .
```

Figure 11-1. FTN5 Control Statement

PARAMETERS

The optional parameters $p_1,\ldots,p_n$ must be separated by commas and can be in any order. If no parameters are specified, FTN5 is followed by a period or right parenthesis. If a parameter list is specified, it must conform to the syntax for job control statements as defined in the operating system reference manual, with the added restriction that a comma is the only valid parameter delimiter. Columns following the right parenthesis or period can be used for comments; they are ignored by the compiler, but are printed on the dayfile.

If the parameter name only is specified the second default value is used. First and second defaults for all parameters are listed in table 11-1. First default values are used for omitted parameters. First defaults are set when the system is installed; since installations can change default values, the user should determine what default values are in effect at the user's particular installation. Second defaults are not installation-changeable.

Unrecognizable parameters cause compilation to terminate. Conflicting options either are resolved or cause compilation to terminate, depending on the severity of the conflict; this resolution is indicated in a dayfile entry.

The values of the B, DB, E, ET, G, I, LO, ML, PD, PN, PS, S, and X parameters are passed to COMPASS when intermixed COMPASS subprograms are present.

Binary Value Parameters

A binary value parameter has two possible settings, on or off. The binary value parameters are indicated in table 11-1.

Specified Value Parameters

Specified value parameters provide the compiler with specific values for such things as file names, page size, and so forth For parameters where multiple values are allowed, the values are separated by slashes. The specified value parameters are indicated in table 11-1.

Multiple Binary Value Parameters

Certain binary value parameters can have multiple values. The form of a multiple binary value parameter is:

```
parameter=op [/op] ...
```

where op is a binary value indicating on or off.

If a multiple binary value parameter appears, the compiler applies the following steps:

1. An initial value is selected for each option.
2. The option list is scanned from left to right.
3. -op deselects the specified option.
4. op selects the specified option.
5. 0 deselects all previously selected options.

For example:

```
FTN5 (LO)
```

produces a source listing, reference map and attributes list (second default is LO=S/R/A).

```
FTN5 (LO=0/A)
```

or

```
FTN5 (LO=-S/-R/A)
```

produces only an attributes list, deselecting the normal second default.

TABLE 11-1. DEFAULTS FOR FTN5 CONTROL STATEMENT

Parameter	First Default (parameter omitted)	Second Default (keyword only)
ANSI	ANSI=0	ANSI=T
ARG	ARG=0	ARG=-COMMON/FIXED
B	B=LGO	B=BIN
BL	BL=0	BL
CS	CS=USER	CS=FIXED
DB	DB=0	DB=TB/SB/SL/ER/ST/PMD
DO	DO=0	DO=OT
DS	DS=0	DS
E	E=OUTPUT	E=ERRS
EL	EL=T	EL=F
ET	ET=0	ET=F
G	G=0	G=SYSTEXT
GO	GO=0	GO
I	I=INPUT	I=COMPILE
LCM	LCM=D	LCM=I
LO	LO=S/A	LO=S/A/R
ML	ML=0	ML=0
OPT	OPT=0	OPT=2
PD	PD=6	PD=8
PL	PL=5000	PL=50000
PN	PN=0	PN
PS	PS=6 if PD=60; PS=8 if PD=80	None
PW	PW=136 (PW=72 for connected file)	PW=72
QC	QC=0	QC
REW	REW=0	REW=I/B
ROUND	ROUND=A/S/M	ROUND=A/S/M/D
S	S=SYSTEXT if G omitted S=0 if G specified	S=SYSTEXT if G omitted S=0 if G specified
SEQ	SEQ=0	SEQ
X	X=OLDPL	X=OPL

Multiple binary value parameters are indicated in table 11-1. The initial values are indicated in the parameter descriptions.

Multiple Appearance of Parameters

The G and S parameters are the only parameters which can have multiple appearances in a FTN5 control statement. If any other parameter appears more than once, compilation terminates.

PARAMETER OPTIONS

Following are descriptions of the options for each of the FTN5 control statement parameters.

ANSI Diagnostics

The ANSI parameter specifies whether use of non-ANSI extensions to FORTRAN are to be diagnosed and if so, how severely. Valid options are:

omitted	Same as ANSI=0.
ANSI	Same as ANSI=T.
ANSI=0	No ANSI diagnostics are generated.
ANSI=T	ANSI errors are treated as trivial errors.
ANSI=F	Non-ANSI usages result in a fatal error. All ANSI warning diagnostics become fatal.

Refer to the EL parameter for an explanation of trivial and fatal diagnostics.

ARG Argument List Attributes

The ARG parameter is a multiple binary value parameter that specifies attributes of external procedure argument lists generated by the compiler. Valid options are:

omitted	Same as ARG=0.
ARG	Same as ARG=-COMMON/FIXED.
ARG=0	Same as ARG=-FIXED/-COMMON; both options are deselected.

ARG=op [/op]

where op is one of the following:

COMMON	Argument lists generated for external procedures will be of the form required for interlanguage communication. Specification of COMMON implies -FIXED.

FIXED — All references in the FORTRAN program to a given external procedure have the same number of arguments. (The compiler-generated argument lists will not contain a zero terminator.)

Initial value is ARG=0. Specification of ARG=COMMON/FIXED is not permitted.

B Binary Output File

The B parameter specifies the name of the compiler output file. Valid options are:

omitted — Same as B=LGO.

B — Same as B=BIN.

B=0 — No binary output file is produced. Cannot be specified with GO.

B=lfn — Compiler-generated binary code is output on the file lfn.

The B parameter conflicts with the QC parameter.

BL Burstable Listing

The BL parameter controls page ejects in the listing produced by the compiler. Valid options are:

omitted — Same as BL=0.

BL — Generates output listing that is easily separable into components by issuing page ejects between source listing, cross-reference-attributes map, and object code listing. Also ensures that each program unit listing contains an even number of pages, issuing a blank page at the end if necessary.

BL=0 — Generates listings in compact format by minimizing page ejects.

CS Collating Sequence

The CS parameter specifies the weight table to be used for the evaluation of character relational expressions. Valid options are:

omitted — Same as CS=USER.

CS — Same as CS=FIXED.

CS=USER — User-specified weight table.

CS=FIXED — Fixed weight table. (Refer to appendix A.)

DB Debugging Options

The DB parameter is a multiple binary value parameter that selects debugging options. Valid options are:

omitted — Same as DB=0.

DB — Same as DB=TB/SB/SL/ER/ID/ST/PMD.

DB=0 — All options are deselected.

DB=op [/op]

where op is one of the following:

TB — A full error traceback occurs upon detection of an execution time error. This option causes arguments to intrinsic functions to be passed by reference.

SB — Subscript bounds checking is performed.

SL — Character substring expressions are checked to ensure that the substring references are within the string.

ER — Selects object time reprieve of execution errors. A message identifying the program unit and line number containing the error is output.

ID — A line number table, symbol table, and special stylized object code (required by CYBER Interactive Debug) are generated along with the binary code. CYBER Interactive Debug uses the tables while processing the user program to determine variable locations, source line locations, and other useful debugging information. This option must be specified if the special FORTRAN features of CYBER Interactive Debug are to be used and debug mode has not been turned on by the DEBUG control statement. DB=-ID overrides a previous DEBUG control statement specification. DB=ID requires OPT=0.

ST — Same as DB=ID except that the stylized object code is not generated.

PMD — Must be specified if the Post Mortem Dump facility is to be used. Symbol tables are written to separate files that are accessed by Post Mortem Dump so that a symbolic analysis of error conditions, variable names and values, and traceback information can be written to an output file.

Initial value is DB=TB/SB/SL/ER/ST/PMD. Any of the preceding options conflict with ARG=FIXED.

DO Loop Control

The DO parameter specifies the manner in which DO loops are to be interpreted by the compiler. Valid options are:

omitted — Same as DO=0.

DO — Same as DO=OT.

DO=0 — Trip count must be less than 131 071 and minimum trip defaults to zero.

DO=op [/op]

where op is one of the following:

LONG — Permits the trip count to exceed 131 071.

OT — Sets the minimum trip count for DO loops to one. This option can result in faster program execution.

Most DO loops have a trip count of at least one and will execute correctly under either option. However, if a DO loop has a trip count of zero and DO=OT is specified, the program might not execute correctly. The effects of this parameter can be overridden by the C$ DO directive.

DS Directive Suppression

The DS parameter suppresses the recognition of C$ directives. Valid options are:

omitted	Same as DS=0.
DS	All C$ directives are treated as comments.
DS=0	All C$ directives are recognized and processed.

E Error File

The E parameter specifies the name of the file to receive error information. Valid options are:

omitted	Same as E=OUTPUT.
E	Same as E=ERRS.
E=lfn	In the event of an error of EL-specified severity or higher, the error line and diagnostic are written to lfn. If the L (full listing) parameter is specified, this information is also written to the file specified by the L parameters. E=0 is an error.

EL Error Level

The EL parameter indicates the severity level of errors to be printed on the output listing. The levels are ordered by increasing severity. Specification of a particular level selects that level and all higher levels. Valid options are:

omitted	Same as EL=T.
EL	Same as EL=F.
EL=T	Lists trivial errors. The syntax of these errors is correct but the usage is questionable.
EL=W	Lists warning errors. These are errors where the syntax is incorrect but the compiler has made an assumption (such as inserting a comma) and continued.
EL=F	Lists fatal errors. A fatal error prevents the compiler from processing the statement where the error occurred.
EL=C	Lists catastrophic errors. These errors are fatal to compilation; the compiler is unable to continue processing the current program unit. Compilation continues with a subsequent program unit.

ET Error Terminate

The ET parameter specifies the action to be taken by the compiler when compilation has completed. If an error of the specified level or higher occurs, the job skips to an EXIT (NOS) or EXIT(S) (NOS/BE). Valid options are:

Omitted	Same as ET=0.
ET	Same as ET=F.
ET=0	The job continues even if errors are encountered.
ET=T	Skips if errors of severity T or higher are detected.
ET=W	Skips if errors of severity W or higher are detected.
ET=F	Skips if errors of severity F or higher are detected.
ET=C	Skips if errors of severity C are detected.

Refer to the EL parameter for a description of error severity levels.

Level T and W errors result in an executable binary file. Level F and C errors result in a binary that will abort the loader.

G Get System Text File

The G parameter specifies the name of a file to be read to obtain a system text for intermixed COMPASS subprograms. Valid options are:

omitted	Same as G=0.
G	Same as G=SYSTEXT
G=0	No system text is loaded.
G=lfn	Loads the system text from file lfn.
G=lfn-recname	Loads the system text from record recname on file lfn. (The hyphen is required separator notation.)

Up to seven system texts can be specified, separated by slashes. Multiple occurrences of this parameter are permitted. A G=0 specification is ignored if any other G option is specified.

GO Automatic Execution

The GO parameter specifies automatic loading and executing. Valid options are:

omitted	Same as GO=0.
GO	The binary output file is loaded and executed after compilation.
GO=0	The binary output file is not loaded and executed after compilation.

The GO option conflicts with the QC and B=0 options.

I Input File

The I parameter specifies the name of the file containing the input source code. Valid options are:

omitted	Same as I=INPUT.
I	Same as I=COMPILE.

I=lfn — Source code to be compiled is contained in file lfn. Compilation ends when an end-of-section, end-of-partition, or end-of-information is encountered. I=0 is an error.

LCM Extended Memory (LCM or ECS Storage Access)

The LCM parameter specifies the requirements on address size for data in extended memory. Refer to section 2, LEVEL Statement, for further information. Valid options are:

omitted	Same as LCM=D.
LCM	Same as LCM=I.
LCM=D	Specifies direct mode addressing. Provides more efficient code for accessing data assigned to extended memory. Extended memory field length must not exceed 131 071 words.
LCM=I	Specifies indirect mode (21-bit) addressing. This is the most efficient mode when extended memory field length exceeds 131 071 words. If a single common block exceeds 131 071 words, an error results.
LCM=G	Specifies giant mode addressing. Required if any single common block is larger than 131 071 words. LCM=G automatically selects DO=LONG.

LO Listing Options

The LO parameter specifies the information that is to appear on the output listing file (L parameter). Multiple options can be specified. Valid options are:

omitted	Same as LO=S/A.
LO	Same as LO=S/A/R.
LO=0	No O, R, A, M, or S information appears on the output listing.
LO=op [/op] ...	

where op is one of the following:

O	Output object code (COMPASS mnemonics) is listed on the output file.
R	A cross-reference map (described later in this section) is written to the output file.
A	A list of program entities (variables, common blocks) and their attributes (data type, class, etc.) is written to the output file.
M	A map, showing the correlation of program entities and physical storage, is written to the output file.
S	A source listing of the program is written to the output file.

Initial value is LO=S/A. The effects of the LO parameter can be controlled by the C$ LIST directive, described later in this section.

ML MODLEVEL Micro

The ML parameter specifies the value of the MODLEVEL micro used by COMPASS. Valid options are:

omitted	Same as ML=0.
ML	Same as ML=0.
ML=0	The current Julian date, in the form yyddd (where yy is the year and dd is the number of the day within the year), is used for the MODLEVEL micro.
ML=str	The string str is used for the MODLEVEL micro; nnn consists of 1 through 7 letters or digits.

OPT Optimization Level

The OPT parameter specifies the level of optimization performed by the compiler. Valid options are:

omitted	Same as OPT=0.
OPT	Same as OPT=2.
OPT=0	Minimum optimization is performed, resulting in fastest compilation. OPT=0 is required for DB=ID.
OPT=1	Intermediate optimization is performed.
OPT=2	High optimization is performed, resulting in slower compilation.
OPT=3	Potentially unsafe optimizations in addition to all OPT=2 optimizations are performed.

In optimizing mode, optimizations can be performed in two ways: by the compiler and by the user. User optimization includes not only the standard methods that represent good programming practice, but also certain specific methods that enable the compiler to optimize more effectively. Source code optimization and object code optimization are discussed in the following paragraphs.

OPT=0 Compilation

In the OPT=0 compilation mode, compile time evaluations are made of constant subexpressions; redundant instructions and expressions within a statement are eliminated.

OPT=1 Compilation

In the OPT=1 compilation mode, the following optimizations take place in addition to those in OPT=0:

1. Redundant instructions and expressions within a sequence of statements are eliminated.
2. PERT critical path scheduling is done to utilize the multiple functional units efficiently.
3. Subscript calculations are simplified, and values of simple integer variables are stored in machine registers throughout loop execution, for innermost loops satisfying all of the following conditions:

   Having no entries other than by normal entry at the beginning of the loop

5. Program logic should be kept simple and straightforward; program unit length should be less than about 600 executable statements.

6. The use of dummy arguments (formal parameters) and variable dimensions should be avoided if possible; common or local variables should be used instead.

PD Print Density

The PD parameter specifies print density for all printable output (L and E files). The destination printer must be capable of supporting the specified density. For interactive connected files, PD options are suppressed. Valid options are:

omitted	Same as PD=6.
PD	Same as PD=8.
PD=6	Compiler output is printed at six lines per inch, single spaced.
PD=8	Compiler output is printed at eight lines per inch, single spaced.

PL Print Limit

The PL parameter specifies the maximum number of records (print lines) that the executing program can write to file OUTPUT. This parameter is operative only when appearing on an FTN5 statement used to compile a main program. Valid options are:

omitted	Same as PL=5000.
PL	Same as PL=50000.
PL=n	Output must not exceed n lines; n is a decimal integer consisting of one through 10 digits.

PN Pagination

The PN parameter specifies page numbering options for the compiler output listing. Valid options are:

omitted	Same as PN=0.
PN	Page numbering is continuous from program unit to program unit, including intermixed COMPASS output.
PN=0	Page numbers begin at 1 for each program unit.

PS Page Size

The PS parameter specifies the number of lines to be included on a printed page of the output listing. Valid options are:

omitted	Same as PS=60 if PD=6. Same as PS=80 if PD=8.
PS=n	Specifies the maximum number of printed lines; n must not be less than 4.

PW Page Width

The PW parameter specifies the width of an output line. Valid options are:

omitted	For a connected listing (L) or error (E) file, same as PW=72. For all other output files, same as PW=136.
PW	Same as PW=72.
PW=n	Printed lines are to contain n characters; n is a decimal integer and must not be less than 50 or greater than 136. Lines shorter than 136 characters are reformatted rather than truncated, as described later in this section under Listings.

QC Quick Syntax Check

The QC parameter specifies that the compiler is to perform a quick syntax check of the source program. Valid options are:

omitted	Same as QC=0.
QC	The compiler performs a full syntactic scan of the program, but no binary code is produced. No code addresses are provided if a reference map is requested. QC compilation is substantially faster than normal compilation; but it must not be selected if the program is to be executed.
QC=0	Quick syntax check is not performed.

The QC option conflicts with the B, GO, and LO=O/M options.

REW Rewind Files

The REW parameter specifies the files to be rewound prior to compilation. Valid options are:

omitted	Same as REW=0.
REW	Same as REW=I/B.
REW=0	No files are rewound.

REW=op[/op]...

where op is one of the following:

I	Rewinds the input file (specified by the I parameter).
E	Rewinds the error file (specified by the E parameter).
B	Rewinds the binary output file (specified by the B parameter).
L	Rewinds the output file (specified by the L parameter).

Initial value is REW=I/B/E.

ROUND Rounded Arithmetic Options

The ROUND parameter specifies which arithmetic operations are to be performed using rounded arithmetic. This parameter controls only the in-line object code compiled for arithmetic expressions; it does not affect computations performed by library subroutines, intrinsic functions, or input/output routines. Valid options are:

omitted — Same as ROUND=A/S/M.

ROUND — Same as ROUND=A/S/M/D.

ROUND=op [/op] ...

ROUND=0 — No rounding is performed.

where op is one of the following:

A — All addition operations are rounded.

S — All subtraction operations are rounded.

M — All multiplication operations are rounded.

D — All division operations are rounded.

Initial value is ROUND=0.

S System Text File

The S parameter specifies the name of the system text to be read by the compiler. Valid options are:

omitted	Same as S=SYSTEXT if G parameter is not specified. Same as S=0 if G parameter is specified.
S	Same as S=SYSTEXT if G parameter is not specified. Same as S=0 if G parameter is specified.
S=0	System text file is not loaded when COMPASS is called to assemble any intermixed COMPASS subprograms.
S=sname	Specifies the system text name to be sname and searches the global library set.
S=lib-sname	Searches the library named lib for the system text named sname. (The hyphen separating lib and sname is required.)

Multiple names can be specified by separating them with slashes, up to a maximum of seven names. Multiple occurrences of this parameter are permitted. An S=0 specification is ignored if any other S option is specified.

SEQ Sequenced Input

The SEQ parameter specifies source file sequencing format. Valid options are:

omitted	Same as SEQ=0.
SEQ	The source input file is in sequenced line format.
SEQ=0	The source input file is in standard FORTRAN format.

X External Text Name

The X parameter specifies the name of the file from which the COMPASS assembler reads the external text when it encounters an XTEXT directive in the intermixed COMPASS program. Valid options are:

omitted	Same as X=OLDPL.
X	Same as X=OPL.
X=lfn	COMPASS assembler reads external text from filelfn.

The X parameter is intended for use with intermixed COMPASS subprograms only.

FTN5 CONTROL STATEMENT EXAMPLES

Some examples of FTN5 control statements are as follows:

Example 1:

FTN5 (ET=F,EL=F,GO,L=SEE,LO=M/R,S=0)

selects the following options:

ET=F	Skip to an EXIT(NOS) or EXIT(S) (NOS/BE) control statement if fatal errors occur during compilation.
EL=F	Fatal diagnostics only are listed.
GO	Generated binary object file is loaded and executed at end of successful compilation.
L=SEE	Listed output appears on file SEE.
LO=M/R	Reference map and storage map are listed in addition to source listing and attributes list.
S=0	When COMPASS is called to assemble an intermixed COMPASS subprogram, it does not read in a system text file.

Example 2:

FTN5 (GO,DB=ID)

selects the following options:

GO	The program is loaded and executed after compilation.
DB=ID	Tables are generated for use by CYBER Interactive Debug.

Example 3:

FTN5.

selects default options. Refer to table 11-1 for a summary of the default options.

EXECUTION CONTROL STATEMENT

Optional parameters can be included on the control statement that calls into execution a program compiled by FORTRAN. This control statement is normally either the name of the file to which the binary object code was written (LGO is the default) or an EXECUTE statement specifying the name of the main entry point of the program (the name used on the PROGRAM statement or START. if the PROGRAM statement was omitted). The parameters that can be included on this control statement are of four kinds: file names, print limit specification (PL), user parameters, and Post Mortem Dump options.

FILE NAME SUBSTITUTION

FORTRAN 5 provides a method of substituting file names at execution time. File names declared on the PROGRAM statement are associated with files of the same name unless the user substitutes a different name. For example, with the PROGRAM statement:

```
PROGRAM TEST1(INPUT,OUTPUT,TAPE1,TAPE2)
```

the execution time file names would be:

```
INPUT

OUTPUT

TAPE1

TAPE2
```

Note that specification of file names on the PROGRAM statement is optional; the same associations would occur if the statement PROGRAM TEST1 were used.

However, file names on the PROGRAM statement can be changed for the execution of a program by substitution in the execution control statement. A one-to-one correspondence exists between parameters on this statement and parameters in the PROGRAM statement. For example, using the preceding PROGRAM statement, an execution control statement of the form:

```
LGO(,,DATA,ANSW)
```

would cause the following files to be used:

File name	File used
INPUT	INPUT
OUTPUT	OUTPUT
TAPE1	DATA
TAPE2	ANSW

If a file name in the PROGRAM statement is equivalenced, the logical file name is the file to the right of the equals sign. A corresponding file name in the execution control statement is ignored. For example, using the control statement:

```
PROGRAM TEST3(INPUT,OUTPUT,
*TAPE1=OUTPUT,TAPE2,TAPE3)
```

and the execution control statement:

```
LGO(,,DATA,ANSW)
```

would cause the following substitutions:

File name	File used
INPUT	INPUT
OUTPUT	OUTPUT
TAPE1	OUTPUT
TAPE2	ANSW
TAPE3	TAPE3

The user should not substitute the same file name for two different file names on the PROGRAM statement.

PRINT LIMIT SPECIFICATION

A parameter can be specified on the execution control statement to regulate the maximum number of records that can be written at execution time on the file OUTPUT. The *PL parameter has the same form as the PL parameter specified at compilation time on the FTN5 control statement. If specified on the execution control statement, the PL parameter overrides the value specified either explicitly or by default at compilation time.

The print limit parameter (specified either at compilation time or at execution time) is operative only on files with the name OUTPUT in the first word of its corresponding file information table. Thus, if a file name declared in the PROGRAM statement is superseded at execution time by the file name OUTPUT as described previously, the print limit parameter will be operative on the original file name. Conversely, if the file name OUTPUT is superseded at execution time by another file name, the effect of the print limit parameter is nullified. Some examples of *PL parameter usage are as follows:

```
LGO(*PL=2000)

EXECUTE(,FILE1,OUTPUT,FILE2,*PL=1000)
```

USER PARAMETERS

User parameters that specify values to be accessible from the user program can be included on the execution control statement. These parameters must appear after any file names, *PL specification, and PMD parameters. The format of a user parameter is as follows:

name [=value]

name — Parameter name consisting of 1 through 7 numbers or letters.

value — Optional parameter value; string consisting of numbers, letters, or the character *. Can also be a literal, containing any character, delimited by dollar signs ($).

An example of an execution control statement containing user parameters is as follows:

```
LGO (INFILE,*PL=1000,V1=25,V2=ABCD)

LGO (X=$@@$)
```

User parameters can be accessed from the FORTRAN program using the GETPARM subroutine call described in section 7.

POST MORTEM DUMP PARAMETERS

The Post Mortem Dump facility, described in section 10, provides a symbolic dump of program variables, an error analysis, and traceback information on the occurrence of a program abort. Parameters can be specified on the execution control statement to control Post Mortem Dump output and to specify limits on array subscripts.

Post Mortem Dump Output Parameter

The Post Mortem Dump output parameter specifies the destination and format of the dump. This parameter has the following format:

*OP=list

The option list consists of one or more of the following, not separated by separators:

A — All active routines are included in the dump. An active routine is one that has been executed but is not necessarily in the traceback chain.

F — Valid for interactive jobs only; Post Mortem Dump output is sent to file PMDUMP when the job is executed with file OUTPUT connected. The following message appears at the terminal at the time of the dump:

```
*POST PROCESSOR OUTPUT WILL BE
    FOUND ON THE FILE PMDUMP*
```

T — Valid for interactive jobs only; a condensed form of the dump is sent to the terminal. File OUTPUT must be connected. The following message appears after the dump:

```
*FOR A FULL ANALYSIS ON THE FILE
    PMDUMP, *

*RERUN JOB, REPLACING LGO CARD
    BY LGO,*OP=F.*
```

If the *OP parameter is omitted, dumps are sent to file PMDUMP when executing from a terminal with file OUTPUT connected. When the dump occurs, the following message appears at the terminal:

```
*POST PROCESSOR OUTPUT WILL BE FOUND ON
    THE FILE PMDUMP.*

*FOR A SUMMARY AT THE TERMINAL,

*RERUN JOB, REPLACING LGO CARD BY
    LGO,*OP=T.
```

An example of a Post Mortem Dump output control parameter is as follows:

LGO (*OP=AF)

Subscript Limit Specification

Subscript limits can be specified on the execution control statement to control the printing of arrays by Post Mortem Dump facility. This has the same effect as a PMDARRY call (section 10). The subscript limit parameter has the following format:

*DA=i+j+k+l+m+n+o

The integers i, j, k, l, m, n, and o specify the maximum values of the subscripts of arrays to be printed; i through o represent the first through seventh subscripts respectively. Subscript limits can be omitted from the list to control printing as follows:

If o is omitted, 7-dimensional arrays are not printed.

If n and o are omitted, 6- through 7-dimensional arrays are not printed.

If m through o are omitted, 5- through 7-dimensional arrays are not printed.

If l through o are omitted, 4- through 7-dimensional arrays are not printed.

If k through o are omitted, 3- through 7-dimensional arrays are not printed.

If i only is specified, only 1-dimensional arrays are printed.

For example, if the control statement:

LGO (*DA=2+5)

is used, only 1- and 2-dimensional arrays will be dumped. If the statement

DIMENSION ARAY(20,20)

appears in the source program, then the following elements will be printed by Post Mortem Dump:

```
ARAY(1,1), ARAY(2,1)
ARAY(1,2), ARAY(2,2)
ARAY(1,3), ARAY(2,3)
ARAY(1,4), ARAY(2,4)
ARAY(1,5), ARAY(2,5)
```

LANGUAGE SUMMARY D

The following symbols are used in the descriptions of the FORTRAN 5 statements: Page

v	variable name, array name, or array element
sl	statement label
iv	integer variable
name	symbolic name
u	input/output unit specifier:
	integer expression with a value of 0 through 999; Boolean expression containing a display code file name in L format
fs	format specifier
iolist	input/output list
ios	input/output status specifier
recn	record number

Other symbols are defined individually in the statement descriptions.

ASSIGNMENT

v = arithmetic expression	3-8
Boolean v = Boolean expression	3-9
character v = character expression	3-8
logical v = logical or relation expression	3-8

MULTIPLE ASSIGNMENT

v[=v]... = expression	3-9

TYPE DECLARATION

INTEGER v[,v]...	2-2
REAL v[,v]...	2-2
DOUBLE PRECISION v[,v]...	2-2
COMPLEX v[,v]...	2-2
BOOLEAN v[,v]...	2-2
LOGICAL v[,v]...	2-3
CHARACTER [*length] v [*length] [,v [*length]]...	2-3
IMPLICIT type(ac [,ac]...) [,type(ac [,ac]...)]...	2-4

ac Is a single letter, or range of letters represented by the first and last letter separated by a hyphen, indicating which variables are implicitly typed.

EXTERNAL DECLARATION

EXTERNAL name [,name]... 2-9

INTRINSIC DECLARATION

INTRINSIC name [,name]... 2-10

STORAGE ALLOCATION

type array(d) [,array(d)]... 2-4

DIMENSION array(d) [,array(d)]... 2-4

type Is INTEGER, CHARACTER, BOOLEAN, REAL, COMPLEX, DOUBLE PRECISION, or LOGICAL.

d Is one through seven array bound expressions separated by commas, as described in section 2.

COMMON [/[name]/]nlist [[,]/[name]/nlist]... 2-6

nlist Is a list of variables or arrays, separated by commas, to be included in the common block.

DATA nlist/clist/ [[,]nlist/clist/]... 2-11

nlist Is a list of names to be initially defined. Each name in the list can take the form:

variable
array
element
substring
implied DO list

clist Is a list of constants or symbolic constants specifying the initial values. Forms for list items are described in section 2.

EQUIVALENCE (nlist) [,(nlist)]... 2-7

nlist Is a list of variable names, array names, array element names, or character substring names. The names are separated by commas.

LEVEL n,name [,name]... 2-8

n Is an unsigned integer constant, or symbolic constant, with the value 0, 1, 2, or 3 indicating the storage level.

PARAMETER (name=exp [,name=exp] ...) 2-5

exp Is a constant or constant expression.

SAVE name [,name]... 2-9

FLOW CONTROL

GO TO sl 4-1

GO TO (sl [,sl]...) [,] expression 4-1

GO TO iv [[,](sl [,sl]...)] 4-2

ASSIGN sl TO iv 4-2

IF (arithmetic or Boolean expression) sl_1,sl_2,sl_3 4-2

IF (logical expression) statement 4-3

IF (logical expression) THEN 4-3

ELSE IF (logical expression) THEN 4-3

ELSE 4-3

END IF 4-4

DO sl [,] v=e_1,e_2 [,e_3] 4-5

e_1,e_2,e_3 Are indexing parameters. They can be integer, real, double precision, or Boolean constants, symbolic constants, variables, or expressions.

PAUSE [n] 4-9

STOP [n] 4-10

n Is a string of 1 through 5 digits, or a character constant.

END 4-10

MAIN PROGRAM

PROGRAM name (fpar [,fpar]...) 6-1

fpar Is a file declaration in one of the following forms:

filename

filename=buffer length

filename=/record length

filename=buffer length/record length

alternate name=filename

SUBPROGRAM

SUBROUTINE name [(argument [,argument]...)] 6-3

[type] FUNCTION name([argument [,argument]]...) 6-4

type Is BOOLEAN, CHARACTER, INTEGER, REAL, COMPLEX, DOUBLE PRECISION, or LOGICAL.

BLOCK DATA name 6-3

STATEMENT FUNCTION

name ([argument [,argument]]...)=expression 6-5

SUBROUTINE CALL

CALL name [(argument [,argument]...)] 6-9

FUNCTION REFERENCE

name ([argument, [argument]]...) 6-10

ENTRY POINT

ENTRY name [(argument [,argument]...)] 6-6

RETURN

RETURN [expression] 6-10

FORMATTED INPUT/OUTPUT

READ ([UNIT=] u, [FMT=] fs [,IOSTAT=ios] [,ERR=sl] [,END=sl]) [iolist]	5-4
READ fs [,iolist]	5-4
WRITE ([UNIT=] u, [FMT=] fs [,IOSTAT=ios] [,ERR=sl]) [iolist]	5-5
PRINT fs [,iolist]	5-5
PUNCH fs [,iolist]	5-5

UNFORMATTED INPUT/OUTPUT

READ ([UNIT=] u [,IOSTAT=ios] [,ERR=sl] [,END=sl]) [iolist]	5-22
WRITE ([UNIT=] u [,IOSTAT=ios] [,ERR=sl]) [iolist]	5-22

LIST DIRECTED INPUT/OUTPUT

READ ([UNIT=] u, [FMT=] * [,IOSTAT=ios] [,ERR=sl] [,END=sl]) [iolist]	5-22
READ * [,iolist]	5-22
WRITE ([UNIT=] u, [FMT=] * [,IOSTAT=ios] [,ERR=sl]) [iolist]	5-24
PRINT * [,iolist]	5-24
PUNCH * [,iolist]	5-24

DIRECT ACCESS INPUT/OUTPUT

READ ([UNIT=] u, [FMT=] fs [,IOSTAT=ios] [,ERR=sl] [,REC=recn]) [iolist]	5-29
WRITE ([UNIT=] u, [FMT=] fs [,IOSTAT=ios] [,ERR=sl] [,REC=recn]) [iolist]	5-30

NAMELIST INPUT/OUTPUT

NAMELIST /name/v [,v]...[/name/v [,v]...]...	5-25
READ ([UNIT=] u, [FMT=] name [,IOSTAT=ios] [,ERR=sl] [,END=sl])	5-26
READ name	5-26
WRITE ([UNIT=] u, [FMT=] name [,IOSTAT=ios] [,ERR=sl])	5-27
PRINT name	5-27
PUNCH name	5-27

name Is a NAMELIST group name.

BUFFER INPUT/OUTPUT

BUFFER IN (u,p) (a,b)	5-28
BUFFER OUT (u,p) (a,b)	5-29

p Is an integer constant or variable:

zero = even parity

nonzero = odd parity

a Is the first word of the data block to be transferred.

b Is the last word of the data block to be transferred.

INTERNAL DATA TRANSFER

ENCODE (c,fs,v)iolist 5-36

DECODE (c,fs,v)iolist 5-36

v	Is the starting location of the record to be transferred.
c	Specifies the number of characters to be transferred to or from each record.

FORMAT SPECIFICATION

sl FORMAT (flist) 5-5

flist Is a list of items, separated by commas, having the following forms:

[r]ed
ned
[r](flist)

ed	Is a repeatable edit descriptor.
ned	Is a nonrepeatable edit descriptor.
r	Is a nonzero unsigned integer constant repeat specification.

EDIT DESCRIPTORS

srEw.d	Single precision floating-point with exponent.	5-8
srEw.dEe	Single precision floating-point with specified exponent length.	5-8
srFw.d	Single precision floating-point without exponent.	5-10
srDw.d	Double precision floating-point with exponent.	5-11
srGw.d	Single precision floating-point with or without exponent.	5-10
srGw.dEe	Single precision floating-point with or without specified exponent length.	5-10
rIw	Decimal integer.	5-8
rIw.m	Decimal integer with specified minimum number of digits.	5-8
rLw	Logical.	5-14
rA	Character with variable length.	5-13
rAw	Character with specified length.	5-13
rRw	Boolean.	5-14
rOw	Octal.	5-15
rOw.m	Octal with minimum digits and leading zeros.	5-15
rZw	Hexadecimal.	5-15
rZw.m	Hexadecimal with minimum digits and leading zeros.	5-15
s	Is an optional scale factor of the form kP.	
r	Is an optional repetition factor.	
w	Is an integer constant indicating field width.	
d	Is an integer constant indicating digits to right of decimal point.	
e	Is an integer constant indicating digits in exponent field.	
m	is an integer constant indicating minimum number of digits in field.	
BN	Blanks ignored on numeric input.	5-13

BZ	Blanks treated as zeros on numeric input.	5-13
SP	+ characters produced on output.	5-13
SS	+ characters suppressed on output.	5-13
S	+ characters suppressed on output.	5-13
nX	Skip n spaces.	5-16
Tn	Tabulate to nth column.	5-17
TRn	Tabulate forward.	5-17
TLn	Tabulate backward.	5-17
nH	Hollerith output.	5-16
"..."	Hollerith output.	5-16
'...'	Character string output.	5-16
:	Format control.	5-20
/	End of FORTRAN record.	5-17

FILE POSITIONING

BACKSPACE ([UNIT=] u[,IOSTAT=ios] [,ERR=sl])	5-38
BACKSPACE u	5-38
REWIND ([UNIT=] u [,IOSTAT=ios] [,ERR=sl])	5-37
REWIND u	5-37
ENDFILE ([UNIT=] u [,IOSTAT=ios] [,ERR=sl])	5-38
ENDFILE u	5-38

FILE STATUS

OPEN ([UNIT=]u [,IOSTAT=ios] [,ERR=sl] [,FILE=fin] [,STATUS=sta] [,ACCESS=acc] [,FORM=fm] [,RECL=rl] [,BLANK=blnk] [,BUFL=bl])	5-30
INQUIRE ({[UNIT=]u / FILE=fin} [,IOSTAT=ios] [,ERR=sl] [,EXIST=ex] [,OPENED=od] [,NUMBER=num] [,NAMED=nmd] [,NAME=fn] [,ACCESS=acc] [,SEQUENTIAL=seq] [,DIRECT=dir] [,FORM=fm] [,FORMATTED=FMT] [,UNFORMATTED=unf] [,RECL=fcl] [,NEXTREC=nr] [,BLANK=blnk])	5-32
CLOSE ([UNIT=] u [,IOSTAT=ios] [,ERR=sl] [,STATUS=sta])	5-32

OVERLAYS

OVERLAY ([fname,] i,j [,orig] [,OV=m])	9-3
fname — Is the name of the file on which the overlay is to be written.	
i,j — Are the overlay level numbers.	
orig — Specifies the origin of the overlay.	
m — Optional specification of number of higher level overlays.	
CALL OVERLAY (fname,i,j [,recall] [,k])	9-3
recall — Is the recall parameter.	
k — Indicates location of fname.	

Also see CYBER Loader reference manual