EDV - ZENTRUM TECHNISCHE UNIVERSITÄT WIEN Hybridrechenanlage	EAI SIMSTAR
	Operatoranleitung
	3. Auflage, Februar 1988 FB

SIMSTAR Operatoranleitung

1. SIMSTAR einschalten

- Power Supplies (2. Kasten von links, vorne) bleiben in der Regel "on". Wenn Einschalten notwendig, dann nicht gleichzeitig, sondern ein Power Supply nach dem anderen. Das Hardware-Diagnostikprogramm *SATS* sollte frühestens eine halbe Stunde nach dem Einschalten gestartet werden.

- Start-Taste bei der rechten Platte drücken (1. Kasten von links, vorne), mit Bootstrap und Log-On warten, bis grünes Lämpchen nicht mehr blinkt.

- Systemkonsole einschalten.

- Tally Printer off-line, einschalten, dann erst on-line drücken.

2. Bootstrap (auf der Systemkonsole)

Ein Bootstrap ist nur dann notwendig, wenn

- auf der Systemkonsole z.B. bei Eingabe von **<CR>** // ausgeschrieben wird (die Eingabe von **<@@P>** kann dann entfallen),

- bei Eingabe von **<@@A>** auf der Systemkonsole kein Log-On möglich ist,

- sich das laufende System aufhängt.

Eingabe:	Bedeutung:
<@@P> (bzw. **Reset-Taste** im SIMSTAR oberhalb der Platten)	
HALT	Halt
RST	Reset
CLE	Clear
IPL=802	Bootstrap von Fixed Disk (160 MB, rechts im Schrank), **800** für Removeable Disk (80 MB, links im Schrank)
TE	Time Enter, **Datum** und **Uhrzeit** eingeben, Format: Tag-Monat-Jahr Stunde:Minute z.B.: 04-06-86 09:45 (mit führenden Nullen!)
<CR>	Carriage Return

3. Tägliche Initialisierungsprozedur

- Downloading der LCP Firmware (siehe 7.)

- Log-On auf der Systemkonsole:
  <@@A>
  ownername
  password

- Initialisieren der Terminalleitungen, falls vorher kein Bootstrap erfolgte:
  INIT (siehe auch 8.8)

- Log-Off:
  X Exit *TSM*

4. SIMSTAR abschalten

- Log-On auf der Systemkonsole (siehe 3.),
  Kontrolle, daß sonst kein Benutzer mehr eingeloggt ist,
  Log-Off (siehe 3.).

- Platten abdrehen (Start-Tasten drücken, Platten stehen, wenn grüne Lämpchen erloschen).

- Systemkonsole abschalten.

- Tally Printer erst off-line drücken, dann abschalten.

5. SIMSTAR Backup

- Log-On auf der Systemkonsole (siehe 3.)

- Starten des Backups:

  BACKUPUSER Wöchentliches Backup der User Directories (aller Nicht-RZ-Benutzer)

  BACKUPUSER dir Backup einer bestimmten User Directory (auf Wunsch des Benutzers)

  BACKUPSYS Backup aller bzw. einer bestimmten System Directory
  BACKUPSYS dir *(SYSTEM, SERVICE, SIMSTAR, SIMSTAR1, ACSLSYS, LIB)*,
  nur mit Ownername *SYSTEM* möglich.

  Es wird jeweils ausgeschrieben, welche Platte eingespannt werden muß. Wenn die Platte ready ist, ist **R** einzutippen. Die Platte wird dann gemounted. Bevor mit dem Kopieren der Files begonnen wird, muß man nochmals **R** eintippen.
  Während des Kopierens kann abgebrochen werden, indem man **<@@A>** eintippt und die folgende Frage mit **A** (Abort) beantwortet. Die Platte wird am Ende bzw. nach dem Abbruch automatisch dismountet. Abschließend erfolgt eine Meldung, ob das Backup erfolgreich war oder abgebrochen wurde.

- Log-Off (siehe 3.)

- Platte ausspannen und Backup im SIMSTAR Backup-Protokoll eintragen.

6. SIMSTAR Hardware-Diagnostik

- Log-On auf der Systemkonsole (siehe 3.)

- **Tally umschalten !** - Bei der Exekution des Diagnostikprogramms *SATS* muß das Tally für das Protokoll im Communications Rack als LCP Printer angeschlossen sein. Baudrate: 1200 Baud.

- Kontrolle mit **WHO** , daß sonst kein Benutzer hybrid rechnet. Die Tasknames *TSM, EDIT, FORT77, CATALOG* oder *VOLMGR* sind erlaubt. Wird die Diagnostik dennoch gestartet, so kann trotzdem nichts passieren, *SATS* und *HYBSYS* sind abgesichert.

- Starten des Diagnostiklaufes:
  SE setzt die Directory *SERVICE* ,
  BATCH HWDIAG startet Batch File mit Commands zur Exekution von *SATS* .

- Log-Off nach dem Ende der Diagnostik (siehe 3.)

- Tally wieder als *TY7EA2* anschließen.

- Diagnostik-Protokoll an Fritz weiterleiten.

7. Downloading der LCP Firmware

Geschieht nach einem Aus- und Wiedereinschalten der Stromversorgung automatisch, ist aber bisweilen auch während des Betriebs notwendig, wenn Probleme beim hybriden Rechnen auftreten.

Downloading erfolgt im laufenden System durch Drücken der schwarzen **Download-Taste** im unteren Bereich der Platine im Slot 5 rechts unterhalb der Plattenstationen. Während des Drückens leuchten im oberen Bereich dieser Platine zwei Lämpchen auf. Nach ca. 20 Sekunden erscheinen auf der Systemkonsole die Meldungen

P1 LCP FIRMWARE FILE CREATED: mm/dd/yy
P1 FIRMWARE DOWNLOAD COMPLETED
P1 H/W MACRO STATUS TABLE RESTORED

Erst danach kann (und sollte) auf der Systemkonsole wieder Befehlseingabe erfolgen.

7a. Wiederherstellung der Makrofiles nach Absturz oder KILL

- Downloading der LCP Firmware (siehe 7.)

- **SMU** :
  Initialize Connection Matrix(C), Macros(M), Tables(T)

- **SATS** :
  Generate current PSP Inventory File *PSP1CURR*

8. System Operating

Abgesehen von einigen Meldungen, die auf der Systemkonsole ausgeschrieben werden, hat die Operatorkonsole keine spezielle Bedeutung für das Operating.

Informationen über die Benutzung des Systems erhält man nur, wenn man sich als eigener Benutzer einloggt und dann bestimmte Befehle eingibt. Das Einloggen muß aber nicht auf der Systemkonsole erfolgen, sondern kann von jedem Terminal durchgeführt werden.

8.1 Log-On

Wakeup Character (auch *Attention Character*) eingeben:
  <@@A> auf Systemkonsole
  <^E> (Control E) auf anderen Terminals
ownername
password

System meldet sich mit *TSM>*

8.2 Operating Programme und Befehlsformat

Es gibt im wesentlichen zwei Programme für das Operating:
  TSM Terminal Services Manager, läuft sofort nach Log-On
  OPCOM Operator Communications, muß extra gestartet werden durch Eingabe von **OPCOM<CR>** , meldet sich mit *??*

Beenden der Programme mit **X** bzw. **EXIT** :
  ?? **X** *OPCOM* wird beendet, *TSM* ist wieder aktiv
  TSM> **X** Log-Off

TSM -Befehle können nur im *TSM* eingegeben werden, können ein vorangestelltes **$** -Zeichen haben und sind immer mit **<CR>** abzuschließen.

OPCOM -Befehle sind ebenfalls mit **<CR>** abzuschließen, können aber nicht nur im *OPCOM* , sondern auch gleich im *TSM* eingegeben werden, wenn ihnen ein ! -Zeichen vorangestellt wird.

Im folgenden sind *OPCOM* -Befehle immer mit ! -Zeichen angegeben (zur Unterscheidung von *TSM* -Befehlen).

8.3 Überblick über Benutzer, Queues, Platten

WHO liefert eine Liste aller eingeloggten Benutzer mit Terminaladresse, Ownernamen, Tasknamen, Project, Volume und Directory

!LIST listet alle laufenden Tasks mit Tasknummer, Tasknamen, Ownernamen, Terminalidentifikation, Priorität, Status (Tabelle siehe *MPX-32 Manual* , Vol.I, pp.2-12/13) und Swap-Status

!LIST PRINT listet alle Printerfiles in der Spool Queue
- mit Jobnummer, Jobname, Ownername, etc. bei Batch Jobs,
- mit Tasknummer, Taskname, Ownername, etc. bei Programmen, die nicht im Batch Environment laufen (z.B. Programme, die durch Aufruf ihres Programmnamens exekutiert werden).

!STATUS VOLU listet die zur Verfügung stehenden Disk Volumes mit Volumenamen, Drive (*DM0802* - Fixed Disk, *DM0800* - Removeable Disk), maximalem und momentan freiem Speicherplatz (in Bytes) sowie Angaben über die Verwendung (Public Volume, Shared Use, etc.)

STATUS aktiviert einen Makro gleichen Namens, der die obigen vier Befehle enthält und exekutiert.

8.4 Anhalten und Abbrechen eines Tasks

Bei selbst gestarteten, nicht interaktiven Programmen kann jederzeit durch Eingabe des *Wakeup Characters* (**<@@A>** bzw. **<^E>**) ein Break-Interrupt für den Task bewirkt werden. Es erfolgt eine entsprechende Meldung am User Terminal. Durch Eingabe eines **C** (für Continue) kann der Task fortgesetzt werden, durch ein **A** (für Abort) wird er abgebrochen.

Fremde bzw. auf den Wakeup Character nicht reagierende Programme können durch *OPCOM* -Befehle angehalten, fortgesetzt und abgebrochen werden, die allerdings die Kenntnis des Tasknamens oder der Tasknummer voraussetzen. Daher muß zuvor **!LIST** exekutiert werden.

!HOLD T,tnam
!HOLD tnum Der Task mit dem Namen *tnam* bzw. der Tasknummer *tnum* wird angehalten.

!CONT T,tnam
!CONT tnum Der Task mit dem Namen *tnam* bzw. der Tasknummer *tnum* wird fortgesetzt.

!ABORT T,tnam
!ABORT tnum Der Task mit dem Namen *tnam* bzw. der Tasknummer *tnum* wird beendet, wobei aber etwa ein ausstehender I/O Request noch durchgeführt wird.
Ein Programm sollte immer zuerst mit **ABORT** abgebrochen werden, da **ABORT** ein sauberes Beenden des Tasks erlaubt (besonders wichtig bei hybriden Programmen!).

!KILL T,tnam
!KILL tnum Der Task mit dem Namen *tnam* bzw. der Tasknummer *tnum* wird beendet, wobei alle ausstehenden I/O Requests abgebrochen werden.
Dieser Befehl sollte nur verwendet werden, wenn ein Task mit **ABORT** nicht abgebrochen werden kann, da bei **KILL** ausstehende Operationen nicht regulär beendet werden.

8.5 Anhalten und Abbrechen einer Liste

8.5.1 Listen mit dem Programm *BPRINT*

Listen, die mit dem Druckprogramm *BPRINT* am Tally gedruckt werden (Macros **SPOOL** und **SPOOLF**), können angehalten, fortgesetzt und abgebrochen werden, indem das Programm *BPRINT* angehalten, fortgesetzt bzw. abgebrochen wird. (Es können auch Programme, die noch nicht zu drucken begonnen haben, abgebrochen werden.)

!LIST BPRINT listet alle laufenden *BPRINT* -Tasks mit Tasknummer, Tasknamen, Ownernamen, Terminalidentifikation, Priorität, Status und Swap-Status.
Der momentan druckende *BPRINT* -Task hat den Status *SWIO* , wartende Tasks haben den Status *SUSP* .

!HOLD tnum *BPRINT* mit der Tasknummer *tnum* wird angehalten.

!CONT tnum *BPRINT* mit der Tasknummer *tnum* wird fortgesetzt.

!ABORT tnum *BPRINT* mit der Tasknummer *tnum* wird abgebrochen.

8.5.2 Listen in der Printer Spool Queue

Listen in der Printer Queue können ebenfalls angehalten, fortgesetzt und abgebrochen werden. Spool Files, die erst in der Printer Queue stehen und noch nicht gedruckt werden, können aus der Queue gelöscht werden. Dazu muß aber vorher mit **!LIST PRINT** die Printer Queue gelistet werden.

!HOLD PRINT Der Spool Output auf dem Printer wird angehalten.

!CONT PRINT Der Spool Output auf dem Printer wird fortgesetzt.

!DEPR Der momentan auf dem Printer ausgegebene Spool File wird abgebrochen.

!DEPR T,tnam
!DEPR tnum Alle Spool Files für den Task mit dem Namen *tnam* bzw. der Tasknummer *tnum* werden gelöscht. Files in der Queue werden gelöscht, momentan gedruckte Files abgebrochen.

!DEPR T,jnam
!DEPR J,jnum Alle Spool Files für den Job mit dem Namen *jnam* im *$JOB* -Statement bzw. mit der Job Sequence Nummer *jnum* werden gelöscht. Files in der Queue werden gelöscht, momentan gedruckte Files abgebrochen.

8.6 Nachricht an einen Benutzer

Es kann eine Nachricht an einen eingeloggten Benutzer oder an alle Terminals geschickt werden:

SIGNAL owner Es kann nach Eingabe des Befehls eine Nachricht (max. 80 Zeichen) für den Benutzer *owner* eingegeben werden.

SIGNAL Die eingegebene Nachricht wird an alle Terminals geschickt. Wenn auf einem Terminal kein Benutzer eingeloggt ist, erscheint die Meldung erst beim nächsten Log-On.

8.7 Richtigstellen des Datums und der Zeit

Das Datum und die Uhrzeit können richtiggestellt werden mit

!ENTER tt-mm-jj hh:mm (führende Nullen, z.B.: 04-06-86 09:45)
nur mit Ownername *SYSTEM* möglich

8.8 Initialisieren einer Terminalleitung

Eine bestimmte oder alle acht Terminalleitungen können, sofern sie nicht in Verwendung stehen, initialisiert werden. Dies geschieht automatisch beim Bootstrap und ist in der Regel im normalen Betrieb nicht notwendig. Eine Liste möglicher Fehlermeldungen findet sich im *MPX-32 Rel. 3.2B Manual* , Vol.III, p.10-32 (im Zimmer von Fritz).

INIT Initialisiert alle Terminalleitungen

INIT 7EAi Initialisiert die Terminalleitung *7EAi*, i=0,...,7

Die Einstellungen der Terminalleitungen befinden sich auf dem File *LOGONFLE* in der Directory *SYSTEM* .

LIST LOGONFLE listet den File mit den Terminalleitungsmerkmalen.

Wenn eine Leitung anders eingestellt werden soll, muß die betreffende Zeile im File geändert und **INIT** für diese Leitung exekutiert werden. Dies geschieht am einfachsten durch den Makro-Aufruf **INITDEV i** (siehe eigene Beschreibung).

Die derzeitige Zuordnung der acht Terminalleitungen ist wie folgt:

- *CA7EA0* - Mininet (Tunix), kein Log-On möglich
- *CA7EA1* - Datenleitung zum Pacer, kein Log-On möglich
- *TY7EA2* - Tally, Spool Device
- *TY7EA3* - Modem (300 Baud)
- *TY7EA4* - Terminalraum (T8)
- *TY7EA5* - TUNET (HYB1.7)
- *TY7EA6* - TUNET (HYB1.8)
- *TY7EA7* - TUNET (HYB1.9)

Systemkonsole:

- *CT7EFC*

8.9 Mounten und Dismounten einer Platte

Die Wechselplatte auf dem Drive *DM0800* (linker Drive, 80 MB) muß, falls der Bootstrap mit **IPL=802** durchgeführt wurde, extra gemountet werden. Das gilt ebenso für die Fixed Disk auf Drive *DM0802* (rechter Drive, 160 MB), falls der Bootstrap mit **IPL=800** gemacht wurde.

Beim Mounten einer Platte muß ihr Volume-Name angegeben werden. Die Volume-Namen der Platten sind folgende:

- *TUSYS* - Fixed Disk
- *USERSAVE* - User Backup Disk (inkl. RZ-Users)
- *SIMSYS* - System Backup Disk und alle EAI System Release Disks
- *SOURCE* - Disk mit EAI Source Files

Eine Platte kann für public use gemountet sein, sie kann dann von allen Benutzern (ohne spezielle Vorkehrungen) angesprochen werden:

MOUNT volume ON DM080x OPTIONS=PUBLIC NOMSG

Eine solche Platte kann nicht dismountet werden, d.h. bei einem Plattenwechsel muß ein neuer Bootstrap gemacht werden:

- System stoppen (**<@@P>** oder **Reset-Taste** im SIMSTAR)
- Platte wechseln
- neuer Bootstrap

Eine Platte kann aber auch nur von einem (oder mehreren) Benutzern gemountet werden:

MOUNT volume ON DM080x OPTIONS=NOMSG

Die Platte bleibt bis zum Ende der Terminal Session gemountet, beim Log-Off wird sie automatisch dismountet. Es kann die Platte aber auch schon vor dem Log-Off dismountet werden:

DISMOUNT volume

Mit **!STATUS VOLU** (siehe 8.3) kann man sich jederzeit einen Überblick über die gemounteten Platten verschaffen.

EDV - ZENTRUM TECHNISCHE UNIVERSITÄT WIEN Hybridrechenanlage	EAI SIMSTAR
	Benützungsanleitung
	3. Auflage, Februar 1988 FB

SIMSTAR Benützungsanleitung

SIMSTAR Benützungsberechtigungen

Ansuchungsformulare für SIMSTAR Benützungsberechtigungen sind beim Operator erhältlich und auch dort abzugeben.

Der Benützer kann sich einen Benützernamen (maximal 8 Zeichen) aussuchen. Als Password erhält jeder Benützer bei der Vergabe der Benützungsberechtigung *DEFPW* . Das Password kann vom Benützer laufend geändert werden, indem er, wenn die Eingabe des Passwords verlangt wird, zuerst das alte Password, einen Beistrich und dann das neue Password (maximal 8 Zeichen) eingibt.

Jeder Benützer erhält eine Directory auf der *TUSYS* -Platte, auf der er seine Files abspeichern kann. Die Directory ist unter dem Benützernamen ansprechbar.

Backups von Benützerdirectories werden jeden Montag gemacht. Zwischendurch kann beim Operator - etwa nach größeren Veränderungen - ein zusätzliches Backup der Benützerdirectory gewünscht werden.

Terminal-Session

Nach dem Einschalten des Terminals sind je nach Terminal-Typ eventuell off-line Initialisierungen durchzuführen. Der entsprechende TUNET-Server meldet sich nach Eingabe von **<CR>** z.B. mit

HYB1>

Durch Eingabe von

DO SIM <CR>

wird eine Verbindung zum EAI SIMSTAR hergestellt. Es wird ausgeschrieben, welche Netzverbindung hergestellt wird, z.B.:

Querying Primary Name Server...
Connecting... session 1 -- connected to sim
session 1 with sim resumed

Falls keine Verbindung zum SIMSTAR hergestellt werden kann (wenn z.B. alle Anschlüsse belegt sind), wird

Connecting... Remote is busy

ausgeschrieben und man muß warten, bis ein Anschluß frei wird.

Nach der Eingabe von

<^E>

meldet sich das Betriebssystem MPX des EAI SIMSTAR Systems mit

MPX-32 RELEASE 3.2B TUSYSE01 TERMINAL SERVICES MANAGER
(C) COPYRIGHT 1983, GOULD INC., COMPUTER SYSTEMS DIVISION.
ENTER YOUR OWNERNAME:

Jetzt muß der **Benützername** eingegeben werden, abgeschlossen durch **<CR>** .

Nach

ENTER KEY

muß das **Password** eingegeben werden, ebenfalls abgeschlossen durch **<CR>** . Die eingegebenen Zeichen sind dabei auf dem Bildschirm nicht sichtbar.

Ist das Password oder der Benützername nicht richtig, dann wird

UNAUTHORIZED NAME OR KEY
TRY AGAIN:

ausgegeben und die Eingabe kann wiederholt werden.

NAME IN USE

zeigt an, daß unter demselben Benützernamen bereits ein Benützer eingestiegen ist.

Bei korrektem Benützernamen und korrektem Password meldet sich das System z.B. mit

```
**********************************************************************
*       H Y R I D R E C H E N A N L A G E   T U   W I E N        *
*       SIMSTAR Software Revision E01              23-11-87      *
**********************************************************************
***                      aktuelle Nachrichten                      ***
SELECT LOGIN
```

Nun wird der File *LOGIN* auf der Benützerdirectory exekutiert. In diesen File kann man Befehle und Programmaufrufe (z.B. zum Initialisieren) schreiben, die man bei jedem Einstieg exekutiert haben will.

Falls kein File *LOGIN* auf der Benützerdirectory vorhanden ist, wird

**FILE NOT FOUND*

ausgeschrieben.

Als nächstes meldet sich der Terminal Services Manager (TSM) als eingabebereit

TSM>

Nun können TSM-Commands eingegeben werden bzw. TSM-Makros aufgerufen werden.

TSM-Commands und Makros

Im folgenden sind die gebräuchlichsten TSM-Commands und Command-Makros angeführt. Die HYBSYS-Makros zum Übersetzen eines Modells, zum Generieren eines Overlays und zum Starten eines Simulationsprogrammes sind im HYBSYS PTRAN User Manual beschrieben. Zu Beginn erfolgt jedoch eine Erklärung des sogenannten *Wakeup Characters* , der im MPX-System eine spezielle Bedeutung hat.

Wakeup Character (auch Attention Character)

Der *Wakeup Character* **<^E>** (auch *Attention Character* genannt) stellt eine Verbindung zum Betriebssystem bzw. zu einem exekutierten Programm her.

Das beginnt bereits beim Login-Vorgang, wo sich beim Eintippen von **<^E>** das Betriebssystem MPX meldet mit der Aufforderung, den Benützernamen einzugeben.

Bei einem laufenden Programm bewirkt die Eingabe von **<^E>** eine Unterbrechung und die Bildschirmmeldung

*** BREAK ** ON:taskname AT:location CPU TIME = n SEC.*
CONTINUE, ABORT, DEBUG, OR HOLD?

Bei Eingabe von **C** (Continue) wird das Programm weiter exekutiert, bei Eingabe von **A** (Abort) abgebrochen.

Wird ein Programm exekutiert und man erhält keine Reaktion auf dem Bildschirm, so kann es sein, daß das Programm auf bestimmte Resourcen (Memory, File, etc.) wartet. Beim Eintippen von **<^E>** erfolgt in diesem Fall eine Meldung, worauf das Programm wartet:

TASK state. CONTINUE OR DELETE?

Bei Eingabe von **C** (Continue) wird das Programm weiter exekutiert, bei Eingabe von **D** (Delete) abgebrochen.

Information auf dem Benützerterminal

Mit dem TSM-Makro **NEWS** erhält man eine kurze Übersicht und Beschreibung der gebräuchlichsten TSM-Makros.

NEWS

Der TSM-Makro **STATUS** schreibt alle gerade eingeloggten Benützer, alle CPU-Aktivitäten sowie Informationen über die Printer Queue und den Disk-Status aus.

STATUS

Filemanipulation

Files werden mit einem Pathnamen angesprochen, der im allgemeinen folgende Form hat:

@volume(directory)resource

volume bezeichnet dabei die Platte, *directory* den Namen der Directory und *resource* den Filenamen.

Die Bezeichnungen für Platte (ist immer *TUSYS*) und Directory können weggelassen werden, wenn der Benützer Files seiner eigenen Benützerdirectory anspricht. Ein Filename kann bis zu 16 Zeichen lang sein. Manche Programme verlangen Filenamen mit bestimmten Zeichen (z.B. FORTRAN-Compiler).

Beispiele für Pathnamen:

@TUSYS(USER9)MAINPR.FOR
(USERDP)TEST
MODEL.H

Der TSM-Makro **DIR** listet alle Filenamen der Benützerdirectory auf dem Benützerterminal.

DIR

Die Filenamen sind alphabetisch sortiert. Jede Zeile sieht folgendermaßen aus:

LOG FILE @TUSYS (Directory)Filename Größe

Es wird immer nur eine Bildschirmseite ausgeschrieben. Falls die Directory noch mehr Files enthält, steht am Ende der Bildschirmseite

ENTER CR FOR MORE

Die Eingabe eines **<CR>** setzt die Liste fort, jedes andere Zeichen bricht die Liste ab.

Eine Auswahl von Filenamen erhält man, wenn man

DIR path

eingibt, wobei der Parameter *path* die auszuwählenden Files kennzeichnet. Die Verwendung von * ermöglicht dabei das Markieren von Stellen im Filenamen, an denen beliebige Zeichen stehen können:

z.B.:

DIR Y*	alle Filenamen, die mit *Y* beginnen
DIR *.FOR	alle Filenamen, die mit *.FOR* enden
DIR *1*	alle Filenamen, die *1* enthalten

Der TSM-Makro **DEL** löscht einen File.

DEL path

path ist der Pathname des zu löschenden Files. Wie beim Makro **DIR** kann * zum Markieren von Stellen im Filenamen, an denen beliebige Zeichen stehen können (z.B. **DEL *.OBJ**), verwendet werden.

Der TSM-Makro **COP** kopiert einen File.

COP path1 path2

path1 ist der Pathname des zu kopierenden Files. Nach dem Kopieren ist der File dann auch unter dem Namen *path2* vorhanden.

Der TSM-Makro **REN** ermöglicht das Umbenennen von Files.

REN path1 path2

Der File mit dem Pathnamen *path1* erhält den Pathnamen *path2* .

Generieren, Editieren und Übersetzen eines Programmes

Der TSM-Command **EDIT** startet den Editor, der das Generieren, Editieren und Abspeichern von Programm-, Daten- und Text-Resourcen ermöglicht. Siehe Editor-Beschreibung.

Der TSM-Makro **FOR** übersetzt ein FORTRAN-Programm (FORTRAN 77 Compiler).

FOR path

Das Source-Programm muß *path.FOR* heißen. Das Object wird unter *path.OBJ* abgespeichert. Die Liste kommt auf *path.LST* .

Falls ein FORTRAN-Programm syntaktisch fehlerhaft war, wird

COMPILATION ERRORS WERE DETECTED. ERRORS DESCRIBED IN LISTED OUTPUT (LO)

ausgeschrieben. Dann kann man z.B. mithilfe des Befehls **LIST** (siehe unten) den File *path.LST* auf dem Terminal listen. Fehlerhafte Zeilen im FORTRAN-Programm sind durch *** gekennzeichnet.

Generieren und Manipulieren von Object-Libraries

Eine Object-Library kann ein oder mehrere Functions oder Unterprogramme im Object-Format enthalten. Eine Object-Library *lpath* besteht aus zwei Files: dem File mit den Object-Programmen *lpath.LIB* und dem Directory-File *lpath.DIR* .

Das Generieren einer Object-Library erfolgt mit dem TSM-Makro **LIBCRE** .

LIBCRE lpath path

Aus dem Object-File *path.OBJ* wird eine Object-Library *lpath* generiert (Files *lpath.LIB* und *lpath.DIR*).

Der TSM-Makro **LIBUPD** fügt die Functions und Unterprogramme des Object-Files *path.OBJ* in der Object-Library *lpath* (Files *lpath.LIB* und *lpath.DIR*) hinzu. Bereits in der Library enthaltene Functions und Unterprogramme werden durch die neuen Versionen ersetzt.

LIBUPD lpath path

Der TSM-Makro **LIBDEL** löscht die Function bzw. das Unterprogramm *module* aus der Object-Library *lpath* .

LIBDEL lpath module

Der TSM-Makro **LIBLIS** listet die Namen der Functions und Unterprogramme der Object-Library *lpath* auf dem Benützerterminal.

LIBLIS lpath

Binden und Rechnen eines Programmes

Der TSM-Makro **LINK** bindet einen Object-File zu einem exekutablen Programm.

LINK path [lpath]

Der Object-File *path.OBJ* wird mit der Library *lpath* (Angabe von lpath optional) zu einem exekutablen Programm gebunden, das den Namen *path* erhält. Automatisch wird die Library *TULIB* dazugebunden. Die Ausgabe erfolgt auf *path.LST* .

Ein exekutables Programm wird durch Angabe seines Pathnamens gestartet:

path

Listen von Files

Auf dem Benützerterminal kann ein File mit dem Befehl

LIST pathname

gelistet werden. Die Ausgabe stoppt nach jeder Bildschirmseite mit

ENTER CR FOR MORE

Die Eingabe eines **<CR>** setzt die Liste fort, jedes andere Zeichen bricht ab.

Der TSM-Makro **SPOOL** listet einen Source File (unformatierter File) auf dem Tally-Printer.

SPOOL dire path [copies]

dire ist der Name der Directory, *path* der Name des Files, unter *copies* kann optional angegeben werden, wie oft der File gelistet werden soll. Am Ende der Liste erfolgt ein Kontrollausdruck mit Benützername und Zeitangaben.

Der TSM-Makro **SPOOLF** listet einen List File (formatierter File) auf dem Tally-Printer.

SPOOLF dire path [copies]

dire ist der Name der Directory, *path* der Name des Files, unter *copies* kann optional angegeben werden, wie oft der File gelistet werden soll. Am Ende der Liste erfolgt ein Kontrollausdruck mit Benützername und Zeitangaben.

Beenden der Terminal-Session

Der TSM-Command **X** beendet eine Terminal Session am SIMSTAR. Es wird ein Zeitprotokoll der Terminal Session ausgeschrieben:

z.B.:

CPU EXECUTION TIME = 00 HOURS- 00 MINUTES- 04.66 SECONDS
TOTAL CONNECT TIME = 00 HOURS- 09 MINUTES- 06.92 SECONDS

RING IN FOR SERVICE

Das MPX-System ist verlassen, die TUNET-Verbindung ist noch aufrecht. Man könnte durch Eingabe von **<^E>** wieder in das MPX-System einsteigen.

Durch Eingabe von **<^_>** gelangt man wieder in den Terminal Server, der sich mit

HYB1>

meldet. Zur Beendigung der Terminal Session muß

DC <CR>

eingegeben werden. Die Rechnerverbindung wird für andere Benützer freigegeben. Der Server schreibt z.B. aus:

Disconnecting ... session 1 -- disconnected from sim(....)

EDV - ZENTRUM TECHNISCHE UNIVERSITÄT WIEN Hybridrechenanlage	EAI SIMSTAR
	Text Editor
	2. Auflage, Februar 1988 FB

SIMSTAR Text Editor

Der MPX-32 Text Editor (**EDIT**) ermöglicht das Generieren und Manipulieren von Textfiles. Der Benützer führt die Edit-Operationen auf einem Arbeitsfile (*work file*) aus. Source-Text kann von einem permanenten Plattenfile auf den Arbeitsfile kopiert werden bzw. nach dem Editieren auf einen Plattenfile gespeichert werden. Der Arbeitsfile ist aber auch ein permanenter Plattenfile, der beim Beenden von **EDIT** nicht verloren geht.

Adressierungstechniken

Der Zugriff auf Text basiert auf Zeilennummern, die auf den einzelnen Textzeilen im Arbeitsfile vorhanden sind.

Es gibt verschiedene Arten anzugeben, auf welche Zeilen sich ein Editierbefehl bezieht: bestimmte Zeilennummer, Zeilenbereich, Gruppe von Zeilen und/oder Zeilenbereichen. Anstelle einer Zeilennummer dürfen auch bestimmte Buchstaben bzw. Schlüsselworte verwendet werden, wie z.B.

F oder **FIRST**	erste Zeile
L oder **LAST**	letzte Zeile
A oder **ALL**	alle Zeilen

Zeilen- und Bereichsadressierung:

Um eine bestimmte Zeile anzusprechen, ist die Zeilennummer **lnr** anzugeben. Zeilenbereiche werden durch **lnr1/lnr2** festgelegt, wobei *lnr1* und *lnr2* die erste und letzte Zeile des Bereichs angeben. Es ist erlaubt, entweder *lnr1* oder *lnr2* wegzulassen, also **/lnr2** oder **lnr1/** einzugeben. In diesem Fall gilt die entsprechende Bereichsgrenze des letzten Zeilenbereichs beim letzten Editierbefehl.

Gruppe von Zeilen und/oder Zeilenbereichen:

Eine Gruppe ist eine beliebige Kombination von Zeilennummern und/oder Zeilenbereichen, jeweils getrennt durch einen Beistrich, z.B.

1,50/60,75,209/L

Einschränkung auf Zeilen mit bestimmtem Inhalt:

Durch Angabe einer Zeichenkette **\string** (eingeschlossen von Backslashes) kann in einem Zeilenbereich oder in einer Gruppe der Zugriff auf jene Zeilen eingeschränkt werden, die die angegebene Zeichenkette enthalten, z.B.

1,50/60,75,209/L \TRAP

Es ist nicht erforderlich, bei jedem Befehl Zeilen zu spezifizieren, da bei fehlender Zeilenangabe (mit gewissen Einschränkungen) die beim letzten Befehl angegebenen Zeilen angesprochen werden.

Zeilennummern

Zeilennummern können im Bereich von 0 bis 9999.999 liegen, wobei maximal drei Ziffern nach dem Dezimalpunkt möglich sind. Beim Generieren von neuen Zeilen ist es sinnvoll, nur ganze Zahlen zu verwenden, um die Möglichkeit von nachfolgenden Einfügungen zu haben.

Wenn **EDIT** Zeilennummern generiert (z.B. mit **COLLECT, MOVE, COPY**), dann erfolgt die Zeilennummernvergabe entsprechend der angegebenen Startnummer und dem durch die am wenigsten signifikante Stelle der Startnummer bestimmten Inkrement.

Für die erste Zeile eines Files wird standardmäßig die Zeilennummer 1 gewählt. Die Zeilennummern 0 bis 0.999 sind aber auch gültige Zeilennummern und können für Einfügungen vor der Zeile 1 verwendet werden.

Die Zeilennummern werden am Beginn der Zeilen ausgeschrieben, sie werden jedoch in den Spalten 73-80 jeder Textzeile gespeichert. Der Text in jeder Zeile kann also maximal 72 Zeichen umfassen. Der Arbeitsfile kann wahlweise mit oder ohne Zeilennummern auf einen permanenten File gespeichert werden.

Mit dem Befehl **USE** wird üblicherweise ein Plattenfile auf den Arbeitsfile kopiert. Wenn der File im Text Editor erstellt wurde und mit den Zeilennummern gespeichert wurde, dann werden diese Nummern übernommen. Bei unnummerierten Files werden in die Spalten 73-80 neue Zeilennummern geschrieben. Enthält der Plattenfile Zeilen mit mehr als 72 Zeichen (z.B. Listen von Prozessoren), so werden die Zeilen auf 72 Zeichen abgeschnitten und in die Spalten 73-80 ebenfalls neue Zeilennummern geschrieben.

STARTEN DES TEXT EDITORS

Die vom Editor bearbeitbaren Arbeitsfiles haben Namen der Form *fc*WRKFL* , wobei *fc* ein zwei Zeichen langer Filecode ist, der beim Starten des Text Editors anzugeben ist. Der Editor wird vom *TSM* aus gestartet mit dem Befehl

TSM> **EDIT fc**
bzw.
TSM> **EDIT** (in diesem Fall wird *fc* vom Editor extra verlangt).

Existiert kein Arbeitsfile mit dem angegebenen Filecode, so wird ein neuer Arbeitsfile mit diesem Namen generiert. (Man kann mehrere Arbeitsfiles - mit unterschiedlichen Filecodes - auf der Directory haben.) Es erfolgt eine Meldung, daß der Arbeitsfile leer ist, anschließend meldet sich der Editor mit

EDT>

und ist bereit für die Eingabe von Befehlen. Man kann dann

- mit dem Befehl **USE** einen Textfile zum Editieren auf den Arbeitsfile kopieren, oder
- mit dem Befehl **COLLECT** neue Textzeilen generieren.

Gibt es auf der aktuellen Directory einen File mit dem Namen *fc*WRKFL* , so wird dieser Arbeitsfile angesprochen und es erfolgt eine Meldung, ob der Inhalt des Files gelöscht, auf einen permanenten Plattenfile abgespeichert oder (ohne nachfolgendes Abspeichern) geändert worden ist (siehe auch Befehl **SHOW**). Man kann dann

- den aktuellen Inhalt des Arbeitsfiles editieren, oder
- mit dem Befehl **USE** einen neuen Textfile zum Editieren auf den Arbeitsfile kopieren, oder
- den Arbeitsfile löschen (Befehl **CLEAR**) und mit dem Befehl **COLLECT** neue Textzeilen generieren.

BEFEHLE DES TEXT EDITORS

Die im folgenden beschriebenen Befehle stellen nicht den gesamten Befehlssatz des Text Editors dar. Es werden auch bei den einzelnen Befehlen nicht alle möglichen Options angeführt. Die Einschränkungen bedeuten jedoch keine Beeinträchtigung bei der Benützung des Editors, sie wurden ausschließlich im Interesse einer einfachen, kurzen Beschreibung getroffen.

Die Befehle des Text Editors können auch abgekürzt eingegeben werden. Der notwendige Teil des Befehlswortes wird in Großbuchstaben, der Rest des Befehlswortes in Kleinbuchstaben angegeben. **COLlect** bedeutet, daß der Befehl **COLLECT** auch in der abgekürzten Form **COL** geschrieben werden kann. Teile des Befehls, die wahlweise eingegeben werden können, erscheinen in eckigen Klammern.

Generieren von Textzeilen

Mit dem Befehl **COLLECT** werden Textzeilen auf einem leeren Arbeitsfile erzeugt bzw. weitere Textzeilen zu einem nichtleeren Arbeitsfile hinzugefügt:

COLlect [lnr]

Beginnend mit der Zeile mit der Nummer *lnr* werden neue Zeilen in den Arbeitsfile aufgenommen. Es wird die Zeilennummer *lnr* ausgeschrieben, dann ist der Text mit einem abschließendem Carriage Return (**<CR>**) einzugeben. Sodann wird die nächste Zeilennummer ausgeschrieben, und die nächste Textzeile muß eingegeben werden. Ein Blank mit anschließendem Return generiert eine Leerzeile.

Das Einfügen von Zeilen wird beendet, indem man nach dem Ausschreiben der nächsten Zeilennummer sofort ein Carriage Return eingibt. Der Befehl wird automatisch beendet, wenn der Editor beim Generieren der nächsten Zeilennummer auf eine bereits existierende Zeilennummer stößt.

Wird für *lnr* eine existierende Zeilennummer angegeben, so wird diese Zeile ausgeschrieben und die Einfügung nach dieser Zeile vorgenommen. Wird *lnr* weggelassen, fügt der Editor die Zeilen nach der letzten (mit einem früheren **COLLECT**) eingefügten Zeile ein bzw. am Ende des Files, wenn noch kein **COLLECT** durchgeführt wurde.

Beispiel:

```
EDT> COL 22.01
22.01  neuer text
22.02  neuer text
22.03  neuer text
22.04  <CR>
EDT> LIST 22/23
22.    text
22.01  neuer text
22.02  neuer text
22.03  neuer text
23.    text
EDT>
```

Auflisten von Textzeilen

Der Befehl **LIST** gestattet das Auflisten von Zeilen des aktuellen Arbeitsfiles oder auch eines permanenten Textfiles in der Benützerdirectory. Dieser Befehl wird implizit exekutiert, wenn kein Befehlswort angegeben wird, d.h. **LISt** kann weggelassen werden.

[LISt] [lgroup] [\string\] [path]

lgroup kann eine Zeilennummer, ein Zeilenbereich oder eine Gruppe von Zeilen und/oder Zeilenbereichen sein. Bei Angabe von *\string* werden nur die Zeilen, die die angegebene Zeichenkette enthalten, gelistet.

Wird kein *path* angegeben, so werden die gewünschten Zeilen aus dem Arbeitsfile gelistet. Bei Angabe von *path* werden die Zeilen aus dem File *path* in der Benützerdirectory gelistet, ohne daß der aktuelle Arbeitsfile irgendwie verändert wird. Eine Einschränkung auf die in *lgroup* angegebenen Zeilen erfolgt in diesem Fall nur dann, wenn die Zeilen des Files mit den Zeilennummern gespeichert sind (siehe Befehl **STORE**), sonst kann nur der gesamte File ohne Nummern gelistet werden. Eine Auswahl mittels Angabe von *\string* kann jedoch in jedem Fall erfolgen.

Beispiel:
```
EDT> COL
13.   text
14.   text
15.   <CR>
EDT> LIST
13.   text
14.   text
EDT> A
 1.   text
 2.   text
 .
 .
13.   text
14.   text
EDT>
```

Löschen von Textzeilen

Der Befehl **DELETE** erlaubt das Löschen von Zeilen aus dem Arbeitsfile:

DELete [lgroup] [\string\]

lgroup kann eine Zeilennummer, ein Zeilenbereich oder eine Gruppe von Zeilen und/oder Zeilenbereichen sein. Bei Angabe von *\string* werden nur die Zeilen, die die angegebene Zeichenkette enthalten, gelöscht.

Beispiel:
```
EDT> DEL 15/20
EDT> 13/22
13.   text
14.   text
21.   text
22.   text
EDT>
```

Verändern von Textzeilen

Ändern von einzelnen Zeichen

Der Befehl **MODIFY** kann verwendet werden, wenn einzelne Zeichen in einer Zeile durch andere Zeichen zu ersetzen sind:

MODify [lgroup] [\string\]

lgroup kann eine Zeilennummer, ein Zeilenbereich oder eine Gruppe von Zeilen und/oder Zeilenbereichen sein. Bei Angabe von *string*\ können nur die Zeilen, die die angegebene Zeichenkette enthalten, modifiziert werden.

Es wird die erste Zeile von *lgroup* ausgeschrieben und der Cursor unterhalb dieser Zeile am Bildschirm positioniert. Mit Hilfe der Leertaste kann man die nicht zu verändernden Zeichen übergehen und an den Stellen, wo Änderungen vorzunehmen sind, die richtigen Zeichen eingeben. Ein überflüssiges Zeichen kann durch Eingabe von ^ durch ein Blank ersetzt werden. (Dies gilt aber nur vor dem Ende des Textes in der Zeile. ^-Zeichen, die nach dem Ende des Textes eingetippt werden, erscheinen so in der modifizierten Zeile.) Mit der Backspace-Taste kann auf vordere Zeichen zurückgegangen werden, etwaige Änderungen in den hinteren Spalten werden dadurch rückgängig gemacht und müssen nochmals durchgeführt werden.

Bei Eingabe eines Carriage Return bleiben die rechts des Cursors stehenden Zeichen unverändert. Es wird zunächst zur Kontrolle die geänderte Zeile und dann die nächste zu ändernde Zeile von *lgroup* ausgeschrieben, und so weiter, bis zur letzten Zeile von *lgroup* .

Die Eingabe eines Carriage Return unmittelbar am Beginn einer Zeile bricht den Befehl ab und läßt die restlichen Zeilen von *lgroup* unverändert. Will man also in einer Zeile keine Änderung durchführen und den Befehl noch nicht abbrechen, so muß man zumindest ein Leerzeichen und dann erst das Carriage Return eintippen.

Beispiel:
```
EDT> MOD 10/15
10.   EDV-ZENTRUMMTU WIEN
10.                ^<CR>
10.   EDV-ZENTRUM TU WIEN
11.   ABT. HYRBIDRECHENANLAGE
11.          BR<CR>
11.   ABT. HYBRIDRECHENANLAGE
12.   A-1040  WIEN
12.   <CR>
EDT>
```

Ersetzen von Zeichenketten

Mit dem Befehl **CHANGE** kann eine Zeichenkette durch eine eventuell auch kürzere oder längere Zeichenkette ersetzt werden, wobei die restlichen Zeichen in der Zeile entsprechend verschoben werden:

CHAnge [lgroup] \string\\[newstring]

lgroup kann eine Zeilennummer, ein Zeilenbereich oder eine Gruppe von Zeilen und/oder Zeilenbereichen sein. *string* ist die zu ersetzende Zeichenkette, *newstring* die Zeichenkette, die für *string* eingesetzt wird.

Bei fehlendem *newstring* (bei 3 aufeinanderfolgenden Backslashes) wird die Zeichenkette *string* gelöscht. Die geänderten Zeilen werden zur Kontrolle am Bildschirm aufgelistet.

Führt eine Ersetzung zu einer Zeile mit mehr als 72 Zeichen, so wird die Zeile, abgeschnitten auf 72 Zeichen, gelistet und der Benützer gefragt, ob dies in seinem Sinn ist. Wird als Antwort **Y** (Yes) eingegeben, so erfolgt die Ersetzung und das Abschneiden der überzähligen Zeichen, und es wird mit der nächsten Zeile von *lgroup* fortgesetzt. Die Antwort **N** (No) verändert die Zeile nicht und beendet den Befehl.

Beispiel:
```
EDT>  CHA A \IV\\IWERT\
14.    IWERT = IWERT + 1
15.    CALL UP (IWERT)
63.    DO 250 I = 1, IWERT
EDT>
```

Ersetzen von Zeilen

Existierende Zeilen im Arbeitsfile können mit dem Befehl **REPLACE** durch andere Textzeilen ersetzt werden:

REPlace [lgroup] [\string\]

lgroup kann eine Zeilennummer, ein Zeilenbereich oder eine Gruppe von Zeilen und/oder Zeilenbereichen sein. Bei Angabe von *\string* werden nur die Zeilen, die die angegebene Zeichenkette enthalten, ersetzt.

Es wird die erste Zeile von *lgroup* ausgeschrieben und der Cursor unterhalb dieser Zeile am Bildschirm positioniert. Es ist nun die neue Textzeile einzutippen und mit einem Carriage Return abzuschliessen. Dann wird die nächste zu ersetzende Zeile von *lgroup* ausgeschrieben, und so weiter, bis zur letzten Zeile von *lgroup* .

Die Eingabe eines Carriage Return unmittelbar am Beginn einer Zeile bricht den Befehl ab und läßt die restlichen Zeilen von *lgroup* unverändert. Um eine Zeile durch eine Leerzeile zu ersetzen, muß man zumindest ein Leerzeichen und dann erst das Carriage Return eintippen.

Beispiel:
```
EDT>  REP 24
24.    The replace command
24.    Der REPLACE-Befehl
EDT>
```

Einfügen von Zeichen am Zeilenbeginn

Ein oder mehrere Zeichen können mit dem Befehl **PREFACE** am Beginn einer existierenden Zeile eingefügt werden:

PREface [lgroup] [\string\]

lgroup kann eine Zeilennummer, ein Zeilenbereich oder eine Gruppe von Zeilen und/oder Zeilenbereichen sein. Bei Angabe von *\string* erfolgt das Einfügen nur bei Zeilen, die die angegebene Zeichenkette enthalten.

Es wird die erste Zeile von *lgroup* ausgeschrieben und der Cursor unterhalb dieser Zeile am Bildschirm positioniert. Es sind nun die am Beginn der Zeile einzufügenden Zeichen einzutippen und mit einem Carriage Return abzuschliessen. Dann wird zunächst zur Kontrolle die geänderte Zeile und anschließend die nächste zu ändernde Zeile von *lgroup* ausgeschrieben, und so weiter, bis zur letzten Zeile von *lgroup* .

Die Eingabe eines Carriage Return ohne ein vorangegangenes Zeichen bricht den Befehl ab und läßt die restlichen Zeilen von *lgroup* unverändert.

Führt eine Einfügung zu einer Zeile mit mehr als 72 Zeichen, so wird die Zeile, abgeschnitten auf 72 Zeichen, gelistet und der Benützer gefragt, ob dies in seinem Sinn ist. Wird als Antwort **Y** (Yes) eingegeben, so erfolgt die Einfügung und das Abschneiden der überzähligen Zeichen, und es wird mit der nächsten Zeile von *lgroup* fortgesetzt. Die Antwort **N** (No) verändert die Zeile nicht.

Beispiel:
```
EDT>  PRE 15
15.    ZEICHEN AM ZEILENBEGINN
15.    EINFUEGEN VON <CR>
15.    EINFUEGEN VON ZEICHEN AM ZEILENBEGINN
EDT>
```

Anfügen von Zeichen am Zeilenende

Ein oder mehrere Zeichen können mit dem Befehl **APPEND** am Ende einer existierenden Zeile angefügt werden:

APPend [lgroup] [\string\]

lgroup kann eine Zeilennummer, ein Zeilenbereich oder eine Gruppe von Zeilen und/oder Zeilenbereichen sein. Bei Angabe von *\string* erfolgt das Anfügen nur bei Zeilen, die die angegebene Zeichenkette enthalten.

Es wird die erste Zeile von *lgroup* ausgeschrieben und der Cursor am Ende dieser Zeile am Bildschirm positioniert. Es sind nun die anzufügenden Zeichen einzutippen und mit einem Carriage Return abzuschliessen. Dann wird die nächste zu ändernde Zeile von *lgroup* ausgeschrieben, und so weiter, bis zur letzten Zeile von *lgroup* .

Die Eingabe eines Carriage Return ohne ein vorangegangenes Zeichen bricht den Befehl ab und läßt die restlichen Zeilen von *lgroup* unverändert. Will man also in einer Zeile keine Änderung durchführen und den Befehl noch nicht abbrechen, so muß man ein Leerzeichen und dann erst das Carriage Return eintippen.

Führt das Anfügen zu einer Zeile mit mehr als 72 Zeichen, so werden die überzähligen Zeichen nicht mehr ausgeschrieben und es wird der Benützer gefragt, ob die abgeschnittene modifizierte Zeile gewünscht wird. Wird als Antwort **Y** (Yes) eingegeben, so erfolgt das Anfügen bis zum 72. Zeichen in der Zeile, und es wird mit der nächsten Zeile von *lgroup* fortgesetzt. Die Antwort **N** (No) verändert die Zeile nicht.

Beispiel:
```
EDT>  APP 15
15.    ANFUEGEN VON ZEICHEN
                                  AM ZEILENENDE<CR>
EDT>  <CR>
15.    ANFUEGEN VON ZEICHEN AM ZEILENENDE
EDT>
```

Verschieben von Textzeilen

Eine oder mehrere Textzeilen können mit dem Befehl **MOVE** von einer Stelle des Arbeitsfiles an eine andere Stelle verschoben werden. Jede Zeile, die richtig an die neue Stelle verschoben wurde, wird danach von der ursprünglichen Position gelöscht.

MOVe [lrange] [\string\] [TO lnr]

lrange kann eine Zeilennummer oder ein Zeilenbereich sein. Bei Angabe von *\string* werden nur die Zeilen, die die angegebene Zeichenkette enthalten, verschoben.

lnr ist jene Zeilennummer, die die erste verschobene Zeile an der neuen Position erhält. Der Editor generiert für die nachfolgenden verschobenen Zeilen an der neuen Position, ausgehend von *lnr* , Zeilennummern mit dem Inkrement 1., .1, .01 oder .001, abhängig von der am wenigsten signifikanten Stelle von *lnr* . Die Zeilennummer *lnr* darf vor der Ausführung des Befehls **MOVE** nicht existieren. Wird *lnr* nicht angegeben, so werden die Zeilen an das Ende des Files verschoben.

Trifft der Editor bei der Generierung von Zeilennummern auf eine bereits bestehende Zeilennummer, dann wird abgebrochen und die Nummer der letzten nicht mehr verschobenen Zeile ausgeschrieben. Es kann dann mit einem neuen **MOVE** mit geeignetem *lnr* der restliche Teil, beginnend bei der ausgeschriebenen Zeilennummer, verschoben werden.

Beispiel:

```
EDT> MOV 60/62 TO 58.1
EDT> 58/63
58.    text
58.1   verschobener text  (frühere Zeile 60)
58.2   verschobener text  (frühere Zeile 61)
58.3   verschobener text  (frühere Zeile 62)
59.    text
63.    text
EDT>
```

Kopieren von Textzeilen

Bestehende Textzeilen können mit dem Befehl **COPY** an eine andere Stelle des Arbeitsfiles kopiert werden. Die zu kopierenden Zeilen können Zeilen des Arbeitsfiles sein, sie können aber auch von einem bereits mit dem Befehl **STORE** abgespeicherten Textfile kommen.

COPy [lrange] [\string\] [[FROM] path] [TO lnr]

lrange kann eine Zeilennummer oder ein Zeilenbereich sein. Fehlt *lrange* , so werden alle Zeilen kopiert. Bei Angabe von *\string* werden nur die Zeilen, die die angegebene Zeichenkette enthalten, kopiert.

Die Angabe von *path* entfällt, wenn Zeilen innerhalb des Arbeitsfiles kopiert werden. Anderenfalls spezifiziert *path* den Textfile, von dem die Zeilen kopiert werden sollen. Um spezielle Zeilen des Files ansprechen zu können, müssen die Zeilen mit den Zeilennummern gespeichert sein (vgl. Befehl **STORE**). Soll der ganze File in den Arbeitsfile kopiert werden, dann dürfen die Zeilennummern auch fehlen. Eine Auswahl mittels Angabe von *\string* kann jedoch in jedem Fall erfolgen.

lnr ist jene Zeilennummer, die die erste kopierte Zeile erhält. Der Editor generiert für die nachfolgenden kopierten Zeilen, ausgehend von *lnr* , Zeilennummern mit dem Inkrement 1., .1, .01 oder .001, abhängig von der am wenigsten signifikanten Stelle von *lnr* . Die Zeilennummer *lnr* darf vor der Ausführung des Befehls **COPY** nicht existieren. Wird *lnr* nicht angegeben, so werden die Zeilen an das Ende des Files kopiert.

Trifft der Editor bei der Generierung von Zeilennummern auf eine bereits bestehende Zeilennummer, dann wird abgebrochen und die Nummer der letzten nicht mehr kopierten Zeile ausgeschrieben. Es kann dann mit einem neuen **COPY** mit geeignetem *lnr* der restliche Teil, beginnend bei der ausgeschriebenen Zeilennummer, kopiert werden.

Beispiele:
```
EDT> COP 93/95 TO 90.5
EDT> 90/96
90.    text
90.5   kopierter text  (wie Zeile 93)
90.6   kopierter text  (wie Zeile 94)
90.7   kopierter text  (wie Zeile 95)
91.    text
92.    text
93.    text
94.    text
95.    text
96.    text
EDT>

EDT> COP 3/5 FORMUL TO 1.01
FORMUL    *FILE* (USER FILE)
EDT> 0/2
 0.5   text
 1.    text
 1.01  kopierter text  (FORMUL Zeile 3)
 1.02  kopierter text  (FORMUL Zeile 4)
 1.03  kopierter text  (FORMUL Zeile 5)
 2.    text
EDT>
```

Neunumerierung der Textzeilen

Mit dem Befehl **NUMBER** kann eine neue Durchnumerierung der Textzeilen im Arbeitsfile vorgenommen werden. Dies ist sehr nützlich, wenn durch das Löschen, Einfügen, Verschieben und Kopieren von Textzeilen die Zeilennummern unübersichtlich geworden sind.

NUMber [lnr] [BY inc]

lnr ist die neue Zeilennummer der ersten Textzeile (standardmäßig 1). *inc* ist das Inkrement für die Generierung der folgenden Zeilennummern (standardmäßig ebenfalls 1, unabhängig von der am wenigsten signifikanten Stelle von *lnr*).

Beispiel:
```
EDT> 0/2
 0.5   text
 1.    text
 1.01  text
 1.02  text
 1.03  text
 2.    text
```

```
EDT>  NUM
EDT>  A
 1.     text  (frühere Zeile 0.5)
 2.     text  (frühere Zeile 1.)
 3.     text  (frühere Zeile 1.01)
 4.     text  (frühere Zeile 1.02)
 5.     text  (frühere Zeile 1.03)
 6.     text  (frühere Zeile 2.)
 .
 .
```

Abspeichern des Arbeitsfiles

Der momentane Inhalt des Arbeitsfiles kann mit dem Befehl **STORE** auf einen permanenten Plattenfile in der Benützerdirectory gespeichert werden. Das ist etwa notwendig, wenn die Textzeilen von einem Programm verarbeitet werden sollen (z.B. Sourcetext für einen Compiler, Eingabedaten für ein Programm). Der Arbeitsfile bleibt dabei unverändert.

STOre [path] [UNN] [SCRatch]

path spezifiziert den (eventuell zu generierenden) permanenten File, auf den gespeichert werden soll. Bei fehlendem *path* wird der zuletzt mit **STORE** oder **USE** angesprochene File angenommen. Mit der Option **UNN** (Unnumbered) werden die Textzeilen ohne Zeilennummern abgespeichert. Die Compiler können Zeilen mit und ohne Nummern verarbeiten.

Die Option **SCRatch** ermöglicht das Überschreiben des Files *path* , wenn *path* bereits existiert. Ohne diese Option wird der Benützer bei existierendem File *path* gefragt, ob er den File wirklich überschreiben möchte:

filename, SCRATCH = N

Bei Eingabe eines Carriage Return oder **N** (No) wird der Befehl abgebrochen, bei **Y** (Yes) wird der File *path* mit dem Inhalt des Arbeitsfiles überschrieben.

Beim Abspeichern auf einen File werden die folgenden Informationen ausgegeben:

volume (directory) filename ownername size type
*volume (directory) fc*WRKFL ownername size type*
*fc*WRKFL STORED xx LINES*

volume ist der Name des Plattendrives, *directory* die Benützerdirectory und *filename* der Name des abgespeicherten Files (*path*). *ownername* ist der Benützername, *size* die Filegröße und *type* der Filetyp (*EE,ED* für Textfiles, *FE* für Arbeitsfiles). *fc* ist der Filecode des bearbeiteten und abgespeicherten Arbeitsfiles und *xx* die Zeilenanzahl.

Beispiel:

```
EDT>  STO FLINKG SCR
TUSYS (STEFAN) FLINKG      STEFAN      4  EE
TUSYS (STEFAN) WS*WRKFL    STEFAN     80  FE
WS*WRKFL STORED   26 LINES
EDT>
```

Laden eines Files in den Arbeitsfile

Mit dem Befehl **USE** wird ein permanenter Plattenfile zum Editieren auf den Arbeitsfile kopiert. Der Plattenfile kann ein bereits früher mit dem Editor bearbeiteter und dann abgespeicherter File oder ein beliebiger anderer Textfile sein.

USE path

path spezifiziert den permanenten File, der auf den aktuellen Arbeitsfile kopiert werden soll.

Wenn Veränderungen im Arbeitsfile vorgenommen wurden und der aktuelle Stand des Arbeitsfiles vor dem Aufruf von **USE** nicht mit **STORE** auf einen Plattenfile gespeichert wurde, wird der Benützer gefragt, ob er den Arbeitsfile wirklich mit dem File *path* überschreiben möchte:

CLEAR = N

Bei Eingabe eines Carriage Return oder **N** (No) wird der Befehl abgebrochen, bei **Y** (Yes) wird der Arbeitsfile mit dem Inhalt des Files *path* überschrieben.

Wenn der angegebene File kein mit dem Editor erzeugter Textfile ist (z.B. Listausgabe eines Compilers), dann erfolgt die Meldung

FILE TYPE NOT ED, EE, OR CO, PROCESS IT (Y,N)?

Wenn man weiß, daß der File Textinformation enthält, kann man versuchen, ihn zu benützen. Compilerlisten werden dabei auf 72 Spalten abgeschnitten.

Beispiel:
```
EDT> USE FLINKG
CLEAR=N  Y<CR>
EDT>
```

Löschen des Inhalts des Arbeitsfiles

CLEAR löscht alle Zeilen des Arbeitsfiles und kann verwendet werden, wenn ein neuer Text (ein neues Programm) erstellt werden soll:

CLEar

Der Befehl ist äquivalent zum Befehl **DEL A** (Delete All).

Beispiel:
```
EDT> CLE
EDT> SHOW
TUSYS (STEFAN) FLINKG        STEFAN      4  EE
TUSYS (STEFAN) WS*WRKFL      STEFAN     80  FE
WS*WRKFL CLEAR
EDT>
```

Status des Arbeitsfiles, Fileübersicht

Der Befehl **SHOW** liefert den Status des Arbeitsfiles und wahlweise auch eine Liste der Files des Benützers.

SHOw

SHOW ohne Parameter liefert den Namen des angesprochenen permanenten Plattenfiles und Informationen über den Arbeitsfile:

volume (directory) filename ownername size type
*volume (directory) fc*WRKFL ownername size type*
*fc*WRKFL status xx LINES*

status gibt den Status des Arbeitsfiles an:

CHANGED bedeutet, daß der Arbeitsfile editiert wurde und die geänderte Version noch nicht auf einen Plattenfile gespeichert wurde,
SAVED heißt, daß der Arbeitsfile gespeichert ist und seit dem Abspeichern nicht weiter editiert worden ist,
CLEAR gibt an, daß der Arbeitsfile keine Textzeilen enthält.

Die Bedeutung der übrigen Informationen ist wie beim Befehl **STORE** .

SHOw path
SHOw FILes

Diese beiden Aufrufe des Befehls **SHOW** liefern Informationen über den File *path* bzw. über alle Files in der Benützerdirectory in der Form

volume (directory) filename ownername size type

Die Auflistung der Files kann mit **<^E>** (Control E) abgebrochen werden.

Beispiele:
```
EDT> SHOW
TUSYS (STEFAN) FLINKG        STEFAN      4  EE
TUSYS (STEFAN) WS*WRKFL      STEFAN     80  FE
WS*WRKFL CHANGED  27 LINES
EDT> SHOW LIBCON
TUSYS (STEFAN) LIBCON        STEFAN     76  CA
EDT>
```

Liste der letzten ausgeführten Befehle

Der Befehl **COMMAND** listet die letzten 4 vom Benützer gegebenen Editor-Befehle:

COMmand

Bei weniger als 4 vorangegangenen Befehlen ist die Liste entsprechend kürzer.

Beispiel:
```
EDT> COM
1/20
DEL 2,5/7
APP 9
A
EDT>
```

BEENDEN DES TEXT EDITORS

Der Befehl **EXIT** beendet den Text Editor und gibt die Kontrolle wieder zurück an den *TSM* .

eXit

Der aktuelle Arbeitsfile bleibt erhalten und kann bei einem neuerlichen Aufruf des Editors weiter editiert werden. Soll der editierte Text von anderen Programmen verwendet werden (z.B. Compilieren eines Source Programms), so muß er vor dem Ausstieg aus dem Editor auf einen permanenten Plattenfile gespeichert werden (Befehl **STORE**).

Beispiel:
EDT> **X**
TSM>

Ausgabe eines SIM-Plotfiles am Bildschirm (Tektronix 4010/4014)

Aufruf

Der TSM-Makro **VIEW** gibt einen SIM-Plotfile (von ACSL oder HYBSYS erzeugt) auf einem Tektronix 4010/4014 kompatiblen Terminal aus.

VIEW filename

Parameter

filename	Plotfilename	(der File *filename.PLT* muß existieren)

Bemerkung

Die File-Extension .PLT darf nicht angeben werden.

Ausgabe eines SIM-Plotfiles am Plotter

Aufruf

Der TSM-Makro **PLOT** gibt einen SIM-Plotfile (von ACSL oder HYBSYS erzeugt) auf dem Houston Data Plotter aus.

PLOT username filename

Parameter

username	Benützername	(= Name der Directory, in der sich der Plotfile befindet)
filename	Plotfilename	(der File *filename.PLT* muß existieren)

Bemerkungen

- Die File-Extension .PLT darf nicht angeben werden.
- Da mehrere Benützer nicht gleichzeitig Daten zum Plotter senden können, muß der Plotversuch nach der Fehlermeldung

  Other user is plotting, please try later

  nach ca. 5 Minuten wiederholt werden.

Die Verwendung von ACSL

Aufruf

Der TSM-Makro **ACSL** startet ACSL (Advanced Continuous Simulation Language).

ACSL model [mode]

Parameter

model — Modellname

mode — Jobsteuerung
A ... all (der File *model.CSL* muß existieren)
- Modell übersetzen und ausführen
R ... run (der File *model.X* muß existieren)
- Modell ausführen
default=A wenn der File *model.X* nicht existiert
default=R wenn der File *model.X* existiert

Bemerkungen

- Nach Änderungen in der Modelldefinition (File: model.CSL) m u ß entweder der Parameter *mode* angegeben, d.h. auf **A** gesetzt werden,

  ACSL model A

  oder der File *model.X* gelöscht werden. Dann kann die Simulation mit

  ACSL model

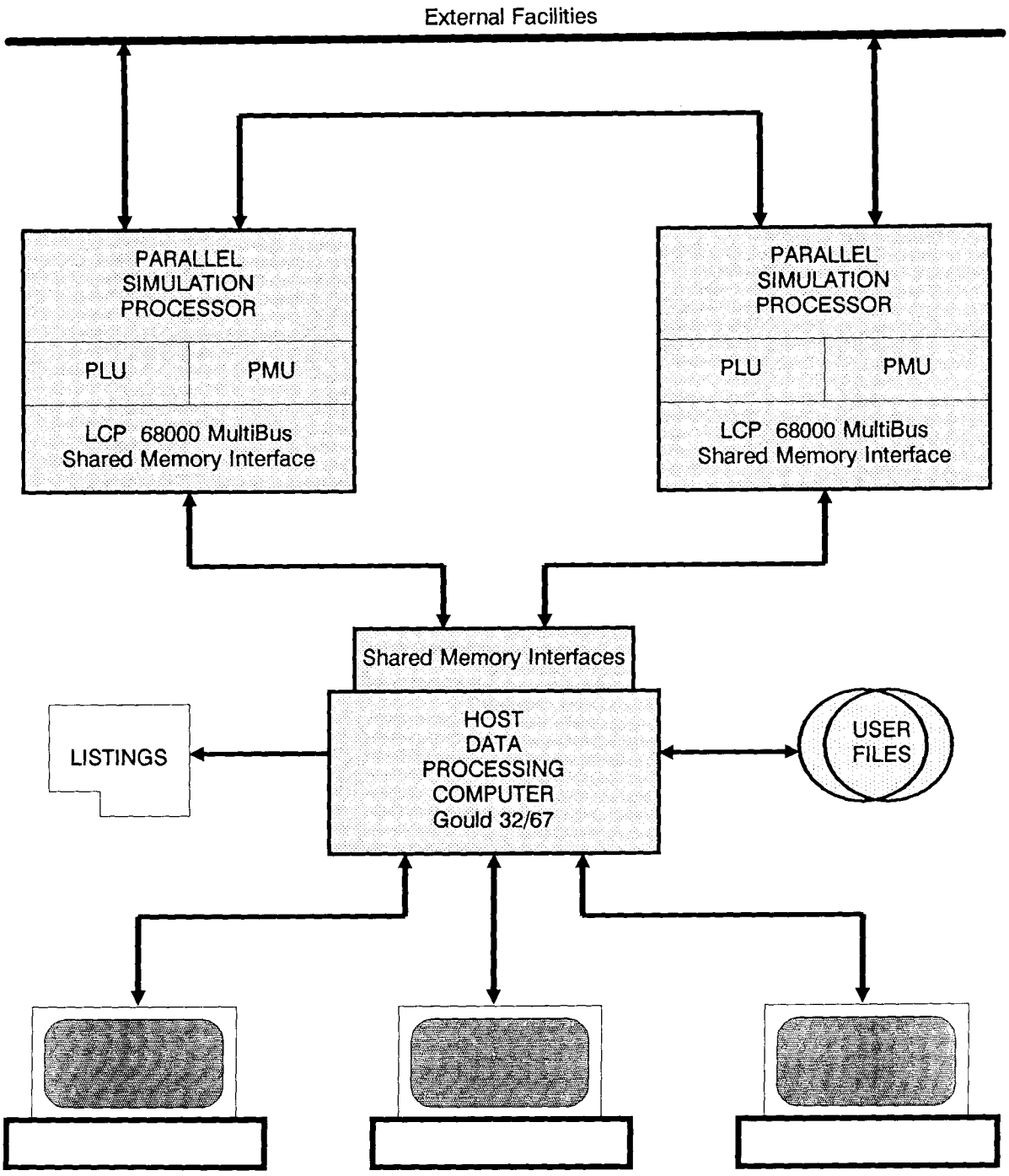
  gestartet werden.

- Die Standardeinstellungen für einige Systemparameter sind am Beginn der Ausführung des Modells durch Laufzeitkommandos nach Wunsch zu verändern:

```
TCWPRN  ...  Anzahl der Spalten für die Ausgabe          (Default=132)
PRNPLT  ...  Ausgabe der Graphikbefehle am Printer       (Default=.T.)
CALPLT  ...  "Line"-Plots                                (Default=.F.)
STRPLT  ...  "Strip"-Plots                               (Default=.F.)
PLT     ...  Steuerung der Graphik-Ausgabe               (Default=9)
             PLT=1    Ausgabe auf Bildschirm (Tektronix 4010/4014)
             PLT=2    Ausgabe auf Bildschirm (Tektronix 4010/4014)
                      und SIM-Plotfile (model.PLT)
             PLT=3    Ausgabe auf SIM-Plotfile (model.PLT)
```

Z.B.: **SET TCWPRN=80,PRNPLT=.F.,CALPLT=.T.,PLT=2**

SIMSTAR Systemarchitektur



für ABB Kurs

89/5/12 IH ABBSYAR.CHP

Interner SIMSTAR-Einführungskurs (85/11/25)

SIMSTAR einschalten

- Power Supplies (2. Kasten von links, vorne) bleiben in der Regel "on". Wenn Einschalten notwendig, dann nicht gleichzeitig, sondern ein Power Supply nach dem anderen. Vor der Diagnostik (SATS) etwa eine halbe Stunde warten.

- Platten: Ready-Switch bei beiden Platten (1. Kasten von links, vorne) drücken, warten bis grünes Licht nicht mehr blinkt.

Bootstrap auf der Systemkonsole

Eingabe	Bedeutung
@@P (bzw. Reset-Taste im SIMSTAR oberhalb der Platten) HALT RST CLE	Reset des Systems, auch bei Aufhänger bei laufendem System zu machen.
IPL=802	Bootstrap von fixed disk (removeable disk = 800)
TE	Time Enter, Datum und Uhrzeit eingeben, Format: Tag-Monat-Jahr Stunde:Minute Z.B.: 25-11-85 09:45 mit führenden Nullen
Carriage Return	

Diagnostik SATS starten

@@A	Login auf Systemkonsole (sonst CTL E)
SERVICE	Ownername für Diagnostik
Carriage Return	kein Keyword

Achtung: Zur Exekution von SATS muß das Tally als List Output Device angeschlossen sein (hellgraues Kabel), wo das Diagnostik-Protokoll erfolgt (ablegen in Ordner "SIMSTAR Diagnostiks").

BATCH MORGEN	Starten des Batch Files mit Commands zur Exekution von SATS

Bemerkungen

Nach jedem Plattenwechsel muß ein Bootstrap gemacht werden.

Genauere Beschreibung in SIM-HINTS (SIM HINT #2).

Der TSM-Command HELP listet alle von Heinz erstellten Command-Makros:

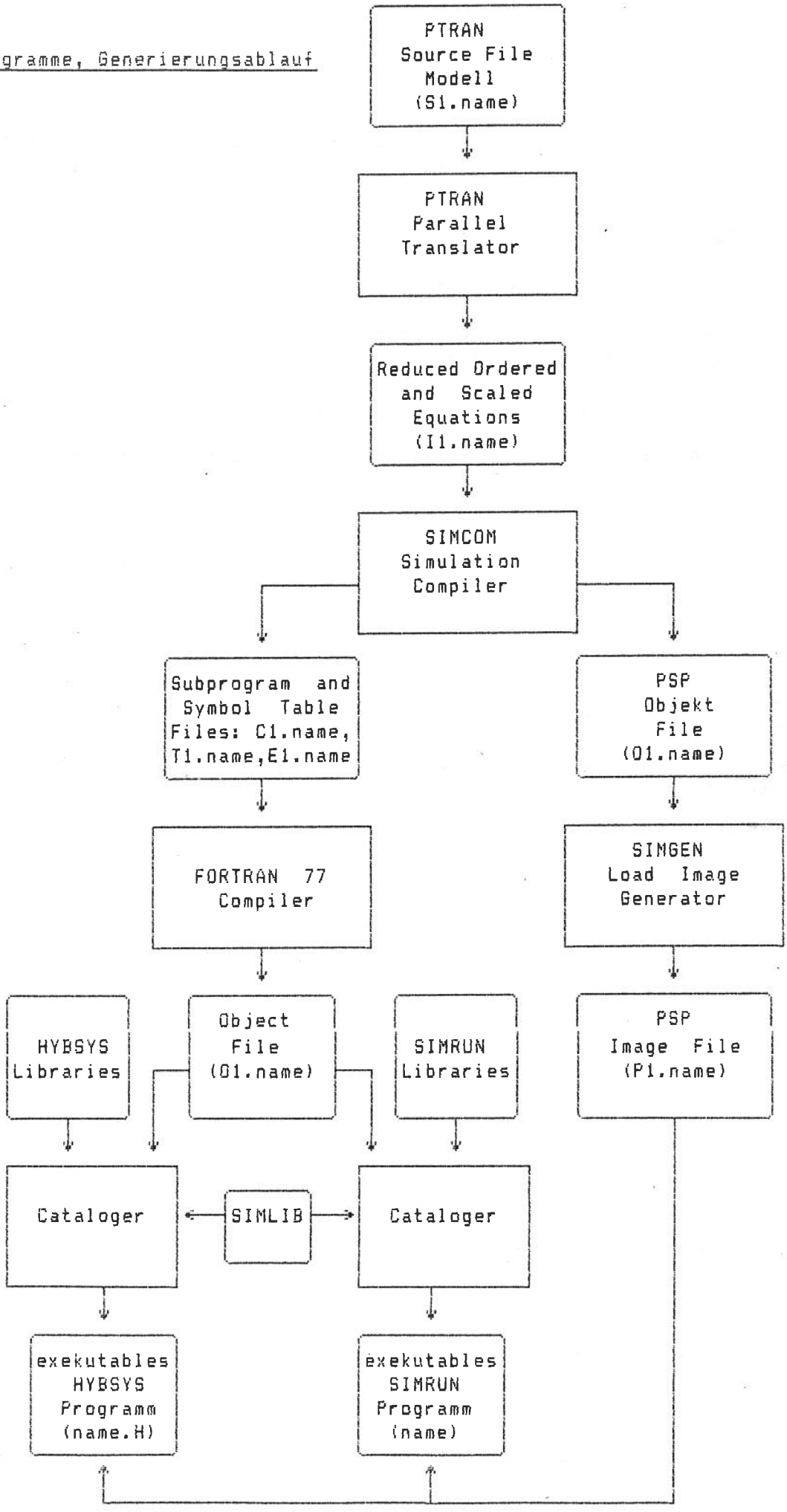
COP path1 path2	COPY FILE path1 TO path2
DEL path	DELETE FILE path
DIR	LIST FILES OF CURRENT DIRECTORY
DIR path	LIST FILES DEFINED BY path
FOR path	FORTRAN COMPILER FOR path.FOR
LIBCRE path1 path2	CREATE LIBRARY path2, INSERT OBJECT path1
LIBDEL module path	DELETE module FROM LIBRARY path
LIBLIS path	LIST MODULES & STATISTICS OF LIBRARY
LIBUPD path1 path2	UPDATE OBJECT path1 IN LIB path2
LINK path1 path2	CATALOG path1.OBJ, PROGRAM IS path1 path2 COULD BE A LIBRARY
MAC path	MACROASSEMBLE path.MAC
NEWS	TYPES SYSTEM NEWS
REN path1 path2	RENAME FILE path1 TO path2
STATUS	SYSTEM STATUS
nn	$USER name, LIKE FB,DS,HS,HH,HY SS=SYSTEM, SE=SERVICE, SI=SIMSTAR

ACSL INSTALLED!
THE FILE LOGIN IN USER DIRECTORY IS EXECUTED AT LOGIN

Beispiel: Exekutieren eines FORTRAN-Programmes

	Files	
- Erstellen des Source-Programmes mit dem Editor (Beschreibung des Editors siehe Manual MPX.32 Vol II)	Source File	name.FOR
- Übersetzen mit dem Makro FOR FOR name	Objekt File List File	name.OBJ name.LST
Der List File kann mit dem Editor gelistet werden.		
- Binden des Objekts mit dem Makro LINK LINK name[lib] Hier kann eine Library lib dazugebunden werden	exekutables Programm	name
- Exekutieren des Programmes durch Eingabe des Filenamens name		

Hybridprogramme, Generierungsablauf



HYBSYS

Nach dem Start des HYBSYS-Programmes meldet sich HYBSYS mit dem bekannten Bindestrich. Beim Laden wird ein Basic Function Check und ein Balance Check gemacht.

Als erster Befehl ist

HPINI zur Initialisierung des HP-Terminals einzugeben

Das Modell muß in PTRAN deklariert werden.
Es sind nahezu alle Befehle zur Modelluntersuchung wie in HYBSYS 5 implementiert

Bei Modelländerung muß ein neuer PTRAN-Lauf gemacht werden.

Ausführliche Dokumentation im vorläufigen User Manual für HYBSYS 6 PTRAN.