Deutsch

Fujitsu Software

BS2000 OS DX Diagnosehandbuch

Benutzerhandbuch

Stand der Beschreibung:
BS2000 V21.0B

Ausgabe November 2025

Kritik… Anregungen… Korrekturen…

Die Redaktion ist interessiert an Ihren Kommentaren zu diesem Handbuch. Ihre Rückmeldungen helfen uns, die Dokumentation zu optimieren und auf Ihre Wünsche und Bedürfnisse abzustimmen.

Sie können uns Ihre Kommentare per E-Mail an bs2000.info@fujitsu.com senden.

Zertifizierte Dokumentation nach DIN EN ISO 9001:2015

Um eine gleichbleibend hohe Qualität und Anwenderfreundlichkeit zu gewährleisten, wurde diese Dokumentation nach den Vorgaben eines Qualitätsmanagementsystems erstellt, welches die Forderungen der DIN EN ISO 9001: 2015 erfüllt.

Copyright und Handelsmarken

Inhaltsverzeichnis

Diagnosehandbuch

1 Einleitung

Will man die Ursachen für Ablaufprobleme im Betriebssystem bzw. in einem Anwenderprogramm ermitteln, dann braucht man Informationen über den Zustand des Betriebssystems bzw. des Anwenderprogramms zum Zeitpunkt des Fehlverhaltens und kurz davor.

Das Diagnosehandbuch beschreibt die in BS2000 vorhandenen Mittel, mit denen ein Betreiber des Betriebssystems BS2000 Informationen über auftretende Software-Fehler und sonstige Programmablaufdaten erhält, sicherstellt und mit denen er sie auswerten kann.

1.1 Zielsetzung und Zielgruppen des Handbuchs

Das Diagnosehandbuch richtet sich an System-Programmierer, die bereits eine gewisse Systemkenntnis besitzen, die Systembetreuung und den Service. Es enthält die Beschreibungen der für die Diagnose wichtigen Software-Produkte und -Komponenten.

Das Handbuch wendet sich sowohl an privilegierte als auch an nichtprivilegierte Anwender von BS2000.

1.2 Konzept des Handbuchs

Dieses Handbuch beschreibt die Software-Diagnoseverfahren in BS2000.

Im Detail werden die in BS2000 vorhandenen Dienstprogramme behandelt, mit denen Informationen über auftretende Software-Fehler und sonstige Programmablaufdaten sichergestellt und ausgewertet werden können.

Ausgenommen ist das Software-Produkt AID. AID ist im Handbuch „AID“ [1] beschrieben.

Ergänzende Produkt-Informationen

Aktuelle Informationen, Versions-, Hardware-Abhängigkeiten und Hinweise für Installation und Einsatz einer Produktversion enthält die zugehörige Freigabemitteilung. Solche Freigabemitteilungen finden Sie online unter http://bs2manuals.ts.fujitsu.com.

1.3 Änderungen gegenüber dem Vorgänger-Handbuch

Die vorliegende Ausgabe des Diagnosehandbuchs enthält gegenüber dem Vorgängerhandbuch folgende, wichtige Änderungen:

- Das Handbuch wurde an BS2000 V21.0 angepasst.
- Kapitel CDUMP: Ein Systemdump auf Band kann mit dem Dienstprogramm PERCON auf Band kopiert werden.
- Kapitel DAMP
  - Neuer Ausgabeformat-Typ ASC.
  - Neue Standardprozedur LIST_CONTROL_BLOCK.
  - STATUS-Elemente, die zur Unterstützung von inzwischen nicht mehr benutzten HSI-Typen verwendet wurden, wurden entfernt.

1.4 Darstellungsmittel

Die Metasyntax der in diesem Handbuch beschriebenen Anweisungen finden Sie im Handbuch „Kommandos“ [8]. Operanden, die sich nur an Anwender mit Privilegierung richten, sind in den Formaten grau unterlegt.

Wegen der häufigen Nennung der Bezeichnungen, werden der Einfachheit und Übersichtlichkeit halber folgende Abkürzungen gebraucht:

- **BS2000-Server** für die Server mit /390-Architektur und die Server mit x86-Architektur. Diese Server werden mit dem entsprechenden BS2000-Betriebssystem betrieben.
- **/390-Server** für die Server Unit /390 der Fujitsu Server BS2000 SE Serie
- **x86-Server** für die Server Unit x86 der Fujitsu Server BS2000 SE Serie
- **SE Server** für die Fujitsu Server BS2000 SE Serie (Server Units /390 und x86)

Die Zeichenfolgen `<date>`, `<time>` und `<version>` bezeichnen in Beispielen die aktuellen Ausgaben für Datum, Uhrzeit und Version eines Software-Produkts, wenn die Beispiele sonst Datums-, Zeit- und Versions-unabhängig sind.

In diesem Handbuch werden folgende Darstellungsmittel verwendet:

Dieses Zeichen kennzeichnet Hinweise auf wichtige Informationen

Dieses Zeichen kennzeichnet einen Warnhinweis, der auf die Möglichkeit des Datenverlustes oder anderer ernsthafter Schäden an Daten hinweist.

[ ] Literaturhinweise werden im Text in Kurztiteln angegeben. Der vollständige Titel jeder Druckschrift, auf die durch eine Nummer verwiesen wird, ist im Literaturverzeichnis hinter der entsprechenden Nummer ufgeführt.

`Eingabe` In Anwendungsbeispielen sind Eingaben an das System und Ausgaben des Systems in `Schreibmaschinenschrift` dargestellt.

2 Software-Diagnoseverfahren in BS2000

Informationen über den Programmablauf erhält man generell auf zwei verschiedene Arten: durch Protokollieren des Programmablaufs und durch Sicherstellen des Speicherinhalts zum Zeitpunkt eines tatsächlichen oder vermuteten Fehlverhaltens. Nachfolgend werden beide Verfahren und ihre Realisierung in BS2000 beschrieben.

Kommando	**Bedeutung**
ACTIVATE-SNAPSHOT	Dump-Erzeuger SNAP aktivieren
ADD-ASE-ELEMENT	ASE-Element deklarieren
ADD-USER	Steuerung von Hardware- bzw. Linkage-AUDIT benutzerspezifisch zulassen, Testprivileg benutzerspezifisch festlegen
CANCEL-JOB	u.a. Steuerung der Dump-Ausgabe von CDUMP
CHANGE-SERSLOG-FILE	SERSLOG-Datei wechseln
CREATE-DUMP	User- bzw. Systemdump anfordern
DEACTIVATE-SNAPSHOT	Dump-Erzeuger SNAP deaktivieren
EXIT-JOB	u.a. Steuerung der Dump-Ausgabe von CDUMP
FORCE-JOB-CANCEL	u.a. Steuerung der Dump-Ausgabe von CDUMP
HOLD-HARDWARE-AUDIT	Hardware-AUDIT-Modus unterbrechen
HOLD-LINKAGE-AUDIT	Linkage-AUDIT-Modus unterbrechen
MODIFY-ASE-PARAMETERS	globale ASE-Einstellungen ändern
MODIFY-FILE-ATTRIBUTES	für eine Benutzerkennung mit dem Privileg HARDWARE-MAINTENANCE dateispezifisch den Zugriff auf mehrbenutzbare Dateien steuern
MODIFY-TEST-OPTIONS	AID-Testprivilegierungswerte tasklokal ändern, Hardware- bzw. Linkage-AUDIT tasklokal zulassen, Userdump-Ausgabe steuern. AID-Test von anderen Tasks unter der eigenen Benutzerkennung mit niedriger Testprivilegierung steuern
MODIFY-USER-ATTRIBUTES	Steuerung von Hardware- bzw. Linkage-AUDIT benutzerspezifisch zulassen. Testprivileg benutzerspezifisch ändern
REMOVE-ASE-ELEMENT	ASE-Element löschen
RESUME-HARDWARE-AUDIT	Unterbrochenen Hardware-AUDIT-Modus fortsetzen
RESUME-LINKAGE-AUDIT	Unterbrochenen Linkage-AUDIT-Modus fortsetzen
SHOW-ASE-ELEMENT	ASE-Element anzeigen
SHOW-ASE-LOGGING	Daten des internen ASE-Loggings anzeigen

SHOW-ASE-PARAMETERS	globale ASE-Einstellungen anzeigen
SHOW-ASE-STATUS	ASE-Statusinformationen anzeigen
SHOW-AUDIT-STATUS	Statusinformationen zu Linkage- und Hardware-AUDIT anzeigen
SHOW-HARDWARE-AUDIT	Hardware-AUDIT-Tabelle auf SYSOUT oder SYSLST anfordern
SHOW-LINKAGE-AUDIT	Linkage-AUDIT-Tabelle auf SYSOUT oder SYSLST anfordern
SHOW-SERSLOG-STATUS	Informationen über SERSLOG anfordern
SHOW-SNAPSHOT-STATUS	Informationen über SNAP-Dump ausgeben
SHOW-TEST-OPTIONS	Taskspezifische Einstellungen für Test und Diagnose ausgeben
SHOW-TRACE-STATUS	Eigenschafts- und Zustandsinformationen über System-Traces anfordern
START-DAMP	DAMP starten
START-ELFE	ELFE aufrufen
START-HARDWARE-AUDIT	Hardware-AUDIT-Modus starten
START-LINKAGE-AUDIT	Linkage-AUDIT-Modus starten
START-NDM-DIAGNOSIS	NDMDUMP aufrufen
START-SERSLOG	SERSLOG aktivieren
START-TRACE	Schaltbaren Ablaufverfolger aktivieren
STOP-HARDWARE-AUDIT	Hardware-AUDIT-Modus beenden und AUDIT-Tabelle freigeben
STOP-LINKAGE-AUDIT	Linkage-AUDIT-Modus beenden und AUDIT-Tabelle freigeben
STOP-SERSLOG	SERSLOG deaktivieren
STOP-TRACE	Aktivierten Ablaufverfolger abschalten
Makro	**Bedeutung**
AUDIT	Hardware- und Linkage-AUDIT-Funktionen anwenden
BKPT	Übergabe der Steuerung an das System
CDUMP2	Speicherauszug veranlassen, ohne dass das Programm beendet wird
TERM	Programm und Auftragsabschnitt beenden; ggf. einen Speicherauszug veranlassen

Tabelle 1: Schnittstellenübersicht für die Diagnoseverfahren in BS2000

2.1 Software-Ablauf protokollieren

Das Protokollieren des Software-Ablaufs ist eine Vorsorgemaßnahme, mit der man im Falle eines Programmfehlers Erkenntnisse über die Fehlerursache erhält. In einem solchen Protokoll werden ständig, auch bereits während des normalen (fehlerfreien) Programmablaufes, bestimmte Informationen sichergestellt. Das können Speicheradressen, Namen von durchlaufenen Modulen oder auch Zeilennummern etc. sein. Tritt ein Fehlerereignis auf, dann lässt sich anhand des Protokolls die Fehlerursache ggf. örtlich eingrenzen. Da jedoch auf diese Weise innerhalb kurzer Zeit große Datenmengen anfallen, die nicht unbedingt für eine Fehleranalyse gebraucht werden, ist es sinnvoll, die Daten nach einer gewissen Zeit wieder freizugeben. Dies erreicht man zum Beispiel durch zyklisches Überschreiben, wobei immer nur die letzten n Einträge im Protokoll erhalten bleiben und so im Falle eines Fehlers dennoch die in die Fehlersituation führenden Schritte im Protokoll vermerkt sind.

Die in BS2000 benutzten Verfahren zum Protokollieren sind das Error-Logging und das Tracing.

2.1.1 Software-Error-Logging

Unter dem Begriff „Software-Error-Logging“ versteht man das Ablegen von Fehlereinträgen (Fehlerrecords) in eine Logbuch-Datei. Dies geschieht immer dann, wenn ein Software-Problem erkannt wurde. Das Software-Modul, bei dessen Durchlaufen der Fehlerzustand auftritt, veranlasst den Logbuch-Eintrag und legt fest, welchen Inhalt er hat.

Das Erzeugen und Ablegen von Fehlereinträgen in eine Logbuch-Datei übernehmen in BS2000 die Komponenten ELS und SERSLOG. ELS protokolliert Hardware-Fehler und ist im Handbuch „ELSA“ [3] beschrieben. Software-Fehler werden von SERSLOG festgehalten. SERSLOG wird im Kapitel „SERSLOG Software-Fehler in SERSLOG-Datei sicherstellen“ beschrieben.

2.1.2 Tracing

Unter dem Begriff „Tracing“ versteht man das Protokollieren von Abläufen. Dabei werden bestimmte Daten aufgezeichnet, unabhängig davon, ob ein Software-Problem vorliegt oder nicht. Diese Daten bleiben eine gewisse Zeit greifbar, so lange nämlich, bis sie wieder von aktuellen Daten überschrieben werden. Der TRACE-MANAGER (siehe Kapitel „TRACE- MANAGER Diagnoseinformation während des Systemlaufs sammeln“) verwaltet solche Protokolle.

Während des normalen Betriebs werden einige Traces standardmäßig protokolliert. In kritischen Fällen können noch weitere Traces dazugeschaltet werden. Manche dieser zuschaltbaren Traces sind jedoch sehr zeitintensiv.

Eine Sonderform des Tracings ist die Funktion AUDIT (siehe Kapitel „AUDIT Adressen ausgeführter Sprungbefehle protokollieren“ (AUDIT Adressen ausgeführter Sprungbefehle protokollieren)). Mit AUDIT werden alle erfüllten Sprungbefehle notiert. Diese Funktion steht auch dem nichtprivilegierten Benutzer zur Verfügung.

> ¡ Das Einschalten zusätzlicher Traces oder des systemweiten AUDIT kann die Betriebssystemperformance erheblich beeinträchtigen und sollte daher nur mit Bedacht eingesetzt werden.

2.2 Speicherinhalte sicherstellen

Mit einem Speicherabzug (Dump) stellt man Inhalte, z.B. aus Registern, Realspeichern, Adressräumen oder Dateien, zum Zeitpunkt eines tatsächlichen oder vermuteten Fehlverhaltens sicher. Ausgelöst wird der Dump entweder automatisch durch eine Fehlererkennungsinstanz des Betriebssystems bzw. des laufenden Programms oder durch das Eingeben eines entsprechenden Kommandos am Terminal bzw. an der Konsole. Abhängig von der Bedeutung des aufgetretenen Fehlers unterscheidet man in BS2000

- den Gesamtdump unter Beendigung des Betriebssystems und
- den Teildump ohne Beendigung des Betriebssystems.

Sind Ursache und Wirkung des Fehlers nicht eingrenzbar und ist ein unverzichtbarer Teil des Betriebssystems beeinträchtigt, müssen für die Diagnose sämtliche Haupt- und Hintergrundspeicherbereiche von BS2000 sichergestellt werden. Einen solchen Gesamtdump erstellt das Programm SLED (**S**elf **L**oading **E**mergency **D**ump, siehe Kapitel „SLED-Dump"). SLED arbeitet unabhängig von BS2000.
Wurde der Gesamtdump erstellt, dann wird das Betriebssystem neu geladen. Bei entsprechender Vorgabe erfolgen das Sicherstellen des Speicherinhalts und das Neustarten des Betriebssystems vollautomatisch.

Lässt sich der Fehler auf eine Task eingrenzen, braucht nur der von der Task belegte Teil des Speichers sichergestellt zu werden. Ein Teildump umfasst daher im Wesentlichen nur den Benutzeradressraum der Fehlertask, wenn der Fehler im Benutzerprogramm aufgetreten ist. Ist andererseits der Fehler in einer Funktion des Betriebssystems aufgetreten, ohne sich jedoch auf andere Tasks auszuwirken, werden auch Bereiche des Betriebssystemadressraums sichergestellt.

Während die fehlerhafte Task nach der Erstellung eines solchen Dumps von der Fehlerbehandlungsroutine meist beendet wird, laufen das Betriebssystem und alle anderen Tasks während der Dumperstellung und auch anschließend weiter.

In BS2000 lassen sich Teildumps (System-, Area- oder Userdumps) mit dem Makro CDUMP2 erzeugen. CDUMP2 (siehe Kapitel „CDUMP Area-, User- oder Systemdump ausgeben" (CDUMP Area-, User- oder Systemdump ausgeben)) arbeitet asynchron, d.h. simultan zu anderen Tasks unter der Kontrolle von BS2000. Der Umfang des mit CDUMP2 erzeugten Teildumps ist abhängig zum einen von der Operandenangabe und zum anderen von der Privilegierung der aufrufenden Task.

Eine besondere Form des Teildumps ist der SNAP-Dump (siehe Kapitel „SNAP-Dump"). SNAP stellt den Klasse-1- und Klasse-3-Speicher des Betriebssystems und die Namens- und Einsprungsliste aller Betriebssystemmodule (EOLDTAB) sicher.

SNAP wird vom Betriebssystem aus dem Zustand TPR oder SIH aufgerufen. Das geschieht im Allgemeinen dann, wenn sich ein irregulärer Betriebssystemzustand einstellt, der jedoch nicht so schwer wiegend ist, dass der Systemlauf beendet werden muss. Nach dem Aufruf hält SNAP BS2000 automatisch für maximal 24 Sekunden an, erstellt unabhängig von BS2000 den Dump und setzt das Betriebssystem anschließend wieder in Gang.

2.3 Dumps und Protokoll-Daten auswerten

Das Auswerten sichergestellter Daten kann bei allen Dumps und Protokollen zu einem beliebigen späteren Zeitpunkt erfolgen.

Die von SERSLOG erstellte Fehlerprotokoll-Datei wird mit dem Programm ELFE ausgewertet. ELFE (siehe Kapitel „ELFE SERSLOG-Datei aufbereiten und auswerten") ist in der Lage, die einzelnen Einträge nach verschiedenen Kriterien zu sortieren und auszugeben. ELFE bietet auch statistische Funktionen (welcher Fehler erscheint wie oft etc.).

Ablaufprotokolle (Traces) werden nicht in eigene Dateien geschrieben, sondern in einen Hauptspeicherbereich und können nicht separat ausgegeben werden. Vielmehr werden sie beim Erzeugen eines Dumps ebenfalls sichergestellt und befinden sich somit in der Dumpdatei. Bei der Dumpauswertung können sie dann ausgegeben werden. Eine Ausnahme bilden die AUDIT-Traces. Diese können mit den SHOW-Kommandos der AUDIT-Funktion im Dialog ausgegeben werden.

Ein SLED-Dump kann mit DAMP ausgewertet werden.
DAMP (siehe Kapitel „DAMP Diagnoseobjekte auswerten") ist ein dialogorientiertes Aufbereitungsprogramm, d.h., die Diagnose wird im Dialog am Bildschirm durchgeführt. So können mit DAMP beispielsweise Verkettungen interaktiv verfolgt, Tabellen ausgegeben und abhängig von deren Inhalt dann ggf. weitere Diagnoseschritte unternommen werden.
Auch Teildumps, die von SNAP oder CDUMP (System-, User- oder Areadumps) erzeugt wurden, lassen sich mit DAMP interaktiv am Bildschirm auswerten.

Für die Diagnose sind Systemkenntnisse erforderlich.

2.4 Datenschutz

- Datenschutz bei der Dumperzeugung
- Datenschutz für eine Dumpdatei
- Datenschutz für Protokolldateien
- Datenschutz bei der Diagnose im aktiven System

2.4.1 Datenschutz bei der Dumperzeugung

Für die Erzeugung und Speicherung eines Dumps können Sie Vorkehrungen bezüglich des Datenschutzes treffen. Mit der Kennzeichnung von Speicherseiten als SECRET PAGES können Sie verhindern, dass diese Seiten aus dem System-, User- oder Areadump in die Dumpdateien übernommen werden.
Voraussetzung dafür ist, dass der Systemparameter DUMPSEPA bei der Systemeinleitung über Parameterservice (siehe Handbuch „Einführung in die Systembetreuung“ [6]) entsprechend eingestellt worden ist. SECRET PAGES werden mit dem Makro CSTAT (siehe Handbuch „Makroaufrufean den Ablaufteil“ [4]) bzw. $CSTA als solche gekennzeichnet. Die Kennzeichnung ist vor allem bei sensiblen Daten sinnvoll. Dabei ist zu prüfen, ob die Daten für die Diagnose wichtig sein können.
Dumps, die schutzwürdige Daten enthalten, werden auf besondere Kennungen abgelegt (SYSUSER, SYSDUMP, SYSSNAP, TSOS). Die Zugriffsberechtigungen darauf vergibt die Systembetreuung.

2.4.2 Datenschutz für eine Dumpdatei

Für Dumpdateien bestehen dieselben Schutzmöglichkeiten wie für alle anderen Dateien auch, nämlich die Standard-Zugriffskontrolle (USER-ACCESS/ACCESS), Basic-ACL (einfache Zugriffskontroll-Liste) und Guards. Dumpdateien lassen sich daher durch das Festlegen und Vergeben geeigneter Zugriffsrechte gegen unbefugten Zugriff schützen.

2.4.3 Datenschutz für Protokolldateien

Es ist nicht möglich, zu protokollierende Daten, etwa für SERSLOG-Fehlerprotokolle, auf ihre Schutzwürdigkeit zu prüfen. Protokolldateien sollten daher genauso gegen Zugriff geschützt werden wie Dumpdateien (siehe oben).

2.4.4 Datenschutz bei der Diagnose im aktiven System

Die Diagnose im aktiven System ist mit dem Software-Produkt AID (siehe Handbuch „AID“ [1]) oder mit DAMP (siehe Kapitel „DAMP Diagnoseobjekte auswerten“) möglich. Durch Gestalten der entsprechender Systemparameter und Vergabe von Testprivilegien kann der Systemverwalter den Datenschutz beeinflussen (siehe Handbuch „Einführung in die Systembetreuung“ [6]).

3 AUDIT Adressen ausgeführter Sprungbefehle protokollieren

Mit der Funktion AUDIT können Sie Adressen ausgeführter Sprungbefehle aufzeichnen, um ggf. den Programmlauf nachträglich zu verfolgen. Auf diese Weise lässt sich nicht nur die eigene Task protokollieren, sondern es besteht auch die Möglichkeit, AUDIT in einer Fremdtask einzuschalten, die sich beispielsweise in einer Schleife befindet oder die als Batchtask abläuft.

Es gibt zwei Arten von AUDIT-Funktionen:

- Hardware-AUDIT
- Linkage-AUDIT

Während der Hardware-AUDIT die Absprungadressen für jeden ausgeführten Sprungbefehl in einer Sprungfolgetabelle hinterlegt, so schreibt der Linkage-AUDIT bei Ausführung bestimmter Branch- und Load-Befehle die Sprungzieladressen in eine Sprungfolgetabelle.Für den Hardware- und Linkage-AUDIT werden getrennte Tabellen angelegt, die bei der Ausgabe in der Kopfzeile entsprechend gekennzeichnet werden.
Hardware-AUDIT und Linkage-AUDIT können unabhängig voneinander eingeschaltet werden. Es können die Sprungadressen eines Prozesses der eigenen Task (z.B. Contingency-Prozess), des gesamten Laufs einer Task oder aller Tasks des laufenden Systems und bei Linkage-AUDIT zusätzlich die Sprungadressen eines Prozessors protokolliert werden.

Es besteht die Möglichkeit den Hardware- und den Linkage-AUDIT für eine ganze Session zu verbieten oder zu erlauben. Im STARTUP-Parameter-Service steht dazu der Parameter AUDALLOW=YES/NO zur Verfügung. Es kann nur Hardware- und Linkage-AUDIT gemeinsam zugelassen oder verboten werden. Ein bereits eingeschalteter prozessorlokaler Linkage-AUDIT wird wieder ausgeschaltet, wenn der Parameter AUDALLOW=NO gesetzt wurde.

Die Funktion AUDIT wird auf Kommandoebene (siehe die nachfolgenden Abschnitte und das Handbuch „Kommandos“ [8]) und als Makro (AUDIT, siehe Handbuch „Makroaufrufe an den Ablaufteil“ [4]) angeboten.

Hardware- und Linkage-AUDIT können zusätzlich user- oder taskbezogen über die Kommandos ADD-USER, MODIFY-USER-ATTRIBUTES und MODIFY-TEST-OPTIONS zugelassen oder verboten werden.

Siehe dazu auch Handbuch „Kommandos“ [8].

```
AUDIT=*PARAMETERS(...)

    *PARAMETERS(...)

        |  HARDWARE-AUDIT = *UNCHANGED/ *NOT-ALLOWED/ *ALLOWED
        |  LINKAGE-AUDIT = *UNCHANGED/ *NOT-ALLOWED/ *ALLOWED
```

Der Parameter *UNCHANGED ist für das Kommando ADD-USER nicht gültig.

Ein Hardware- bzw. Linkage-AUDIT-Aufruf wird bei fehlender Zulassung mit einem Returncode (Makroaufruf) oder einer Meldung (Kommando) abgewiesen.

> **i** Auf den x86-Servern fehlen die Firmware-Voraussetzungen zur Sprungverfolgung in Programmen. Deshalb ist der Hardware-AUDIT nicht verfügbar. Die Kommandos HOLD-HARDWARE-AUDIT, RESUME-HARDWARE-AUDIT, SHOW-HARD-WARE-AUDIT, START-HARDWARE-AUDIT und STOP-HARDWARE-AUDIT werden mit der Meldung `IDA0020` abgewiesen. Für Makroaufrufe zum Hardware-AUDIT wird der Returncode X'00000000' zurückgeliefert. Möglichkeiten zur Analyse von TU-Programmen mit Adress-Stops und Ablaufverfolgung bietet auch das Produkt AID. Siehe Handbuch „AID“ [1].

3.1 Hardware-AUDIT

Bei eingeschaltetem Hardware-AUDIT werden alle ausgeführten Sprungbefehle mit der Absprungadresse in der Hardware-AUDIT-Tabelle hinterlegt. Diese Tabelle ist 256 Bytes groß (entspricht 64 Einträgen zu je einem Wort) und wird zyklisch überschrieben.

Auf Benutzeranforderung können die Daten der AUDIT-Tabelle in einer Sicherungstabelle vor dem Überschreiben gerettet werden. Diese zusätzliche Tabelle ist für den TU-AUDIT maximal 64 KB groß. Für den TPR-AUDIT (nur für den privilegierten Aufrufer unter der Benutzerkennung TSOS) wird an Stelle der 256 Byte großen AUDIT-Tabelle ausschließlich eine auf 4 KB vergrößerte AUDIT-Tabelle, die gleichzeitig Sicherungstabelle ist, angelegt.

Die Sicherungstabelle und die vergrößerte AUDIT-Tabelle werden zyklisch überschrieben. Sie können mit dem Kommando SHOW-HARDWARE-AUDIT und durch Angabe der entsprechenden Operanden beim AUDIT-Makro ausgegeben werden.

Kommando	**Bedeutung**
HOLD-HARDWARE-AUDIT	Hardware-AUDIT-Modus unterbrechen
RESUME-HARDWARE-AUDIT	Unterbrochenen Hardware-AUDIT-Modus fortsetzen
SHOW-AUDIT-STATUS	Statusinformationen zu Linkage- und Hardware-AUDIT anzeigen
SHOW-HARDWARE-AUDIT	Hardware-AUDIT-Tabelle auf SYSOUT oder SYSLST anfordern
START-HARDWARE-AUDIT	Hardware-AUDIT-Modus starten
STOP-HARDWARE-AUDIT	Hardware-AUDIT-Modus beenden und hardware AUDIT-Tabelle freigeben
Makro	**Bedeutung**
AUDIT	Hardware- und Linkage-AUDIT-Funktionen anwenden

Tabelle 2: Schnittstellenübersicht für den Hardware-AUDIT

3.2 Linkage-AUDIT

Bei eingeschaltetem Linkage-AUDIT wird die Ausführung der Befehle BASR, BALR, BASSM und BAKR mit der Aufzeichnung der Sprungzieladressen in einer 4 KB großen Sprungfolgetabelle (Linkage-AUDIT-Table) protokolliert. Diese Tabelle wird zyklisch überschrieben und kann bei TU-Anwendungen analog zum Hardware-AUDIT in eine bis 64 KB große Sicherungstabelle geschrieben werden (siehe "Hardware-AUDIT"). In TPR ist das Anlegen einer Sicherungstabelle nicht möglich.
Ist der Linkage-AUDIT aktiv, kommt es zu einer Perfomancebeeinträchtigung des Funktionszustandes, für den der Linkage-AUDIT eingeschaltet ist.
Im Vergleich zum Hardware-AUDIT treten die mit Linkage-AUDIT protokollierten Sprünge wesentlich seltener auf als bedingte Sprünge, die beim Hardware-AUDIT protokolliert werden.

AUDIT-Steuerung über Parameterservice

Der Linkage-AUDIT ermöglicht das CPU-spezifische Einschalten des Linkage-AUDIT für alle CPUs bzw. alle logischen Maschinen einer Server-Konfiguration. Pro CPU wird eine Trace-Tabelle im privilegierten Klasse-3-Speicher angelegt, die während der gesamten Session erhalten bleibt. Der Linkage-AUDIT kann in der Startup-Phase über den Parameterservice eingeschaltet werden.

Die Steuerung erfolgt mittels Parametersatz SYSOPT-IPL, über den unterschiedliche Vorgaben für die Systemeinleitung festgelegt werden können. Für AUDIT gilt der Parameter LINKAGE-AUDIT-SCOPE, der innerhalb des SYSOPT-IPL-Satzes wie folgt definiert ist:

```
/BS2000 PARAMS
/BEGIN SYSOPT-IPL
LINKAGE-AUDIT-SCOPE=NO/INTERRUPT-HANDLING/SYSTEM-LEVEL
/EOF
/END-PARAMS
```

Bedeutung der Operanden

LINKAGE-AUDIT-SCOPE = NO
Der Linkage-AUDIT wird nicht eingeschaltet (Standardwert).

LINKAGE-AUDIT-SCOPE = INTERRUPT-HANDLING
Der Linkage-AUDIT wird für den Funktionszustand SIH eingeschaltet.

LINKAGE-AUDIT-SCOPE = SYSTEM-LEVEL
Der Linkage-AUDIT wird für die Funktionszustände TPR und SIH eingeschaltet.

Zusätzlich zu dieser speziellen AUDIT-Steuerung ist eine allgemeine Steuerung des AUDIT über den Systemparameter AUDALLOW und über diverse BS2000-Kommandos möglich (siehe dazu "AUDIT Adressen ausgeführter Sprungbefehle protokollieren").

AUDIT-Steuerung in der laufenden Session

Zum Ein- bzw. Ausschalten in der laufenden Session stehen dem Systemverwalter die Kommandos START-LINKAGE-AUDIT, STOP-LINKAGE-AUDIT, RESUME-LINKAGE-AUDIT sowie der Makro AUDIT zur Verfügung.

Kommando	Bedeutung
HOLD-LINKAGE-AUDIT	Linkage-AUDIT-Modus unterbrechen

RESUME-LINKAGE-AUDIT	Unterbrochenen Linkage-AUDIT-Modus fortsetzen
SHOW-AUDIT-STATUS	Statusinformationen zu Linkage- und Hardware-AUDIT anzeigen
SHOW-LINKAGE-AUDIT	Linkage-AUDIT-Tabelle auf SYSOUT oder SYSLST anfordern
START-LINKAGE-AUDIT	Linkage-AUDIT-Modus starten
STOP-LINKAGE-AUDIT	Linkage-AUDIT-Modus beenden und Linkage-AUDIT-Tabelle freigeben
Makro	**Bedeutung**
AUDIT	Hardware- und Linkage-AUDIT-Funktionen anwenden

Tabelle 3: Schnittstellenübersicht für den Linkage-AUDIT

4 CDUMP Area-, User- oder Systemdump ausgeben

Der Makro CDUMP2 (siehe Handbuch „Makroaufrufe an den Ablaufteil“ [4]) erstellt unter der Steuerung einer eigenen Task einen Speicherabzug (Dump). Je nach Operandenangabe handelt es sich dabei um einen Area-, User- oder Systemdump (siehe Abschnitt „Dumpformen"). Der Makro generiert eine 31-Bit-Schnittstelle. Weiterentwicklungen werden nur noch für CDUMP2 erfolgen. Zur Generierung einer 24-Bit-Schnittstelle muss weiterhin der alte Makro CDUMP verwendet werden.

Die Dumps werden unaufbereitet in eine PAM-Datei auf Platte oder Band abgelegt. Eine Verteilung auf mehrere Magnetbänder ist nicht möglich. Mit dem Kommando COPY-FILE können Dumps von Band auf Platte kopiert werden (siehe Beispiel "Systemdump"). Die Struktur der Dump-Datei entspricht dem SLEDFILE-Format. Alle von CDUMP erzeugten Dumparten können mit dem Diagnoseprogramm DAMP (siehe Kapitel „DAMP Diagnoseobjekte auswerten") ausgewertet werden.

Mit dem Kommando CREATE-DUMP (siehe Handbuch „Kommandos“ [8]) wird auf Kommandoebene ein Userdump eingeleitet.

Area- und Userdumps werden unter der Benutzerkennung der Aufrufertask (Fehlertask) katalogisiert, es sei denn, sie enthalten Speicherseiten mit besonderen Attributen (siehe "Einfluss von Seitenattributen") oder Daten, die lesegeschützt sind. In diesem Fall werden sie unter der Kennung SYSUSER katalogisiert. Systemdumps werden in jedem Fall unter der Kennung SYSDUMP katalogisiert. Bei Speicherplatzmangel für die Benutzerkennung der Aufrufertasks oder die Kennung SYSUSER wird eine Fehlermeldung ausgegeben und die Dumperzeugung abgebrochen. Für die Kennung SYSDUMP wird der Speicher automatisch erweitert und die Dumpausgabe fortgesetzt. Wird dabei Space Saturation Level 5 erreicht, werden an Stelle des Dumps nur noch Error-Logging-Einträge erzeugt.

Im Batch- oder Prozedurbetrieb werden User- oder Areadumps nur erstellt, wenn vorher das Kommando MODIFY-TEST-OPTIONS DUMP=YES eingegeben wurde.

Kann auf Grund eines bei der Dumpbearbeitung auftretenden Systemfehlers kein Userdump erstellt werden, versucht CDUMP einen Systemdump zu erstellen. Wenn auch dies scheitert, erstellt CDUMP einen SERSLOG-Eintrag, der neben dem TCB (Task Control Block) und dem PCB (Program Control Block) noch den CDUMP-Arbeitsbereich mit der TTSAV (CDUMP-interne Daten) enthält.

Kommando	**Bedeutung**
ADD-USER	Testprivileg benutzerspezifisch festlegen
CANCEL-JOB	Steuerung der Dump-Ausgabe von CDUMP
CREATE-DUMP	User- bzw. Systemdump anfordern
EXIT-JOB	Steuerung der Dump-Ausgabe von CDUMP
FORCE-JOB-CANCEL	Steuerung der Dump-Ausgabe von CDUMP
MODIFY-TEST-OPTIONS	Userdump-Ausgabe steuern
MODIFY-USER-ATTRIBUTES	Testprivileg benutzerspezifisch ändern
SHOW-TEST-OPTIONS	Taskspezifische Einstellungen für Test und Diagnose ausgeben
SHOW-USER-ATTRIBUTES	Benutzerspezifische Einstellungen für Test und Diagnose ausgeben

Makro	Bedeutung
CDUMP2	Speicherauszug veranlassen, ohne dass das Programm beendet wird
TERM	Programm und Auftragsabschnitt beenden; ggfs. einen Speicherauszug veranlassen

Tabelle 4: Schnittstellenübersicht für die Dump-Steuerung

4.1 Dumpformen

- Areadump
- Userdump
- Systemdump

4.1.1 Areadump

Ein Areadump wird mit der Operandenangabe SCOPE=*AREA beim Makro CDUMP2 angefordert (nur zulässig aus dem Bereich Task-User, TU). Er enthält die Bereiche aus dem Klasse-5- und Klasse-6-Speicher, die beim Aufruf ausgewählt wurden und wird unter der Benutzerkennung des Aufrufers abgelegt (Ausnahmen: siehe Abschnitt „Einfluss von Seitenattributen" (Einfluss von Seitenattributen)).

Einige Systemparameter beeinflussen ebenfalls die Ausgabe der Speicherbereiche (siehe Abschnitt „Steuerung durch Systemparameter").

Über den Operanden DSCTRL können Bereiche aus einem Datenraum (Data Space) adressiert werden, die in den Areadump aufgenommen werden sollen.

In der Standard-Einstellung MODE=*STD enthalten Areadumps neben den angegebenen Bereichen auch

- Modul AIDSYSD
- Bereiche mit COMAREA
- TU-PCB
- TCB

Bei Angabe von MODE=*EXP enthalten Areadumps auch

- Bereiche mit COMAREA
- Modul AIDSYSD
- Trace Dump List
- TTSAV (CDUMP-interne Daten)
- TU-AUDIT-Tabelle
- Binder/Lader-Metadaten
- Speicherbereiche mit den Tabellen
  XVT (Executive Vector Table)
  SVMT (System Virtual Memory Table)
  UVMT (User Virtual Memory Table)
  TCB (Task Control Block)
  TU-PCBs (Process Control Block)
- Seite, auf welche die im Operanden PC angegebene Referenzadresse zeigt

Dateiname des Areadumps

CDUMP legt den Areadump unter einer der folgenden Kennungen ab:

- unter der Benutzerkennung des Aufrufers, falls dieser berechtigt ist, alle im Dump ausgegebenen Daten zu lesen.
- unter der Systemkennung SYSUSER, wenn der Dump lesegeschützte Daten enthält (z. B. Programme, die mit einem Lesekennwort geschützt sind, das der Benutzer nicht in die Passwort-Tabelle der Task eingetragen hat). Dann kann nur der Systemverwalter dem Benutzer den Zugriff ermöglichen, z. B. mit

  ```
  /MODIFY-FILE-ATTRIBUTES dateiname, -
  /   PROTECTION=*PARAMETERS(USER-ACCESS=*ALL-USERS,READ-PASSWORD=readpass)
  ```

Der Dateiname eines Areadumps hat den Aufbau:

- `:catid:$userid.SYS.ADUMP[.jobname].tsn.i`,
  falls die Datei unter der Benutzerkennung des Aufrufers abgelegt wird
- `:catid:$SYSUSER.SYS.ADUMP[.jobname].tsn.i.userid`,
  falls die Datei unter der Systemkennung SYSUSER abgelegt wird

Dabei bedeuten:

catid	Katalogkennung des Public-Volume-Sets, auf dem der Dump abgelegt wurde
userid	Benutzerkennung des Aufrufers
jobname	Jobname des Auftrags, maximal achtstellig
tsn	vierstellige TSN der Fehlertask
i	fünfstellige laufende Nummer des Areadumps

4.1.2 Userdump

Ein Userdump wird mit der Operandenangabe SCOPE=*USER beim Makro CDUMP2 oder mit dem Kommando CREATE-DUMP angefordert. Er umfasst den ganzen Benutzeradressraum, also den Klasse-5- und den Klasse-6-Speicher.
Einige Systemparameter beeinflussen die Ausgabe der Speicherbereiche (siehe Abschnitt „Steuerung durch Systemparameter").

Ausgabemöglichkeiten mit CDUMP2-Operanden:

- benutzerspezifische Datenräume (Data Spaces) mit dem Operanden DS
- benutzerspezifische DIV-Fenster mit dem Operanden DIV
- eingelagerte POSIX-Dateien mit dem Operanden MMAP

Zeigt der Fehlertask-Befehlszähler in den Systemadressraum, werden zusätzlich alle über Mehrzweckregister und PCs (bis zu 256 PCBs der Fehlertask) referenzierten Seiten, inklusive 5 Seiten vorher und 5 Seiten nachher, ausgegeben.

Ein Userdump enthält ferner folgende Speicherbereiche:

- Modul AIDSYSD
- Trace Dump List
- TTSAV (CDUMP-interne Daten)
- TU-AUDIT-Tabelle
- Binder/Lader-Metadaten
- System-Trace-Tabelle
- Speicherbereiche mit den Tabellen
  XVT (Executive Vector Table)
  SVMT (System Virtual Memory Table)
  UVMT (User Virtual Memory Table)
  TCB (Task Control Block)
  TU-PCBs (Process Control Block)
- Seite, auf welche die im Operanden PC angegebene Referenzadresse zeigt
- JIT390-Verwaltungsdaten

> ¡ Die System-Trace-Table wird beim Userdump mit ihrem Inhalt, der zum Zeitpunkt des CDUMP-Aufrufs besteht, zwischengespeichert und bei der Dump-Erstellung in den Dump eingearbeitet (an der Stelle, an der die Table auch im System steht).

Dateiname eines Userdumps

CDUMP legt den Userdump unter einer der folgenden Kennungen ab:

- unter der Benutzerkennung des Aufrufers, falls dieser berechtigt ist, alle im Dump ausgegebenen Daten zu lesen.

- unter der Systemkennung SYSUSER, wenn der Dump lesegeschützte Daten enthält (z. B. Programme, die mit einem Lesekennwort geschützt sind, das der Benutzer nicht in die Passwort-Tabelle der Task eingetragen hat). Enthält der Dump mindestens eine Seite, die privilegiert, aber nicht „common readable“ ist, wird der Userdump ebenfalls auf die Kennung SYSUSER ausgegeben.

  Wenn der Dump unter $SYSUSER abgelegt ist, kann nur der Systemverwalter dem Benutzer den Zugriff ermöglichen, z. B. mit

  ```
  /MODIFY-FILE-ATTRIBUTES dateiname, -
  /   PROTECTION=*PARAMETERS(USER-ACCESS=*ALL-USERS,READ-PASSWORD=readpass)
  ```

Der Dateiname eines Userdumps hat den Aufbau:

- `:catid:$userid.DUMP[.jobname].tsn.i`,
  falls die Datei unter der Benutzerkennung des Aufrufers abgelegt wird
- `:catid:$SYSUSER.DUMP[.jobname].tsn.i.userid`,
  falls die Datei unter der Systemkennung SYSUSER abgelegt wird

Dabei bedeuten:

catid	Katalogkennung des Public-Volume-Sets, auf dem der Dump abgelegt wurde
userid	Benutzerkennung des Aufrufers
jobname	Jobname des Auftrags, maximal achtstellig
tsn	vierstellige TSN der Fehlertask
i	fünfstellige laufende Nummer des Userdumps

4.1.3 Systemdump

Ein Systemdump wird mit der Operandenangabe SCOPE=*SYSTEM beim Makro CDUMP2 angefordert und stets unter der Kennung SYSDUMP katalogisiert.
Er umfasst den gesamten Klasse-6-, Klasse-5-, Klasse-3- und Klasse-1-Speicher mit Ausnahme der Seiten, die zu „Secret Pages“ erklärt wurden. Einige Systemparameter beeinflussen die Ausgabe der Speicherbereiche (siehe Abschnitt „Steuerung durch Systemparameter" (Steuerung durch Systemparameter)).

Die Bereiche mit den Tabellen

- EXVT
- SVMT
- UVMT
- TCB
- PCB-Stack
- JCB
- TTSAV
- P1-AUDIT

werden automatisch mit dem Klasse-5-, Klasse-3- und Klasse-1-Speicher ausgegeben.Vom Klasse-4-Speicher werden alle Datenseiten außer „Secret Pages“ ausgegeben.

Ferner umfasst der Systemdump solche Seiten des Klasse-4- und Klasse-2-Speichers, die bis zu einem Abstand von 5 Seiten vor und 5 Seiten nach einer Referenzadresse liegen und die Referenzseite selbst. Auf Referenzadressen verweisen die Befehlszähler (PC) aus den PCBs und der Trace Table sowie die Mehrzweckregister der PCBs und die Börsenregister der PCBs. Ausgenommen davon sind Speicherseiten mit dem Attribut „Secret Pages“.

Ein Systemdump enthält außerdem:

- die Module: AIDSYSD, EOLDTAB, DMCHD, NSISINF und CLASS2OP
- den Bereich Trace Dump List
- die Systemdateien REPLOG und SERSLOG
  (sie werden bis zum Last Page Pointer jeweils in eigenen Dumpabschnitten hinterlegt)

Mit dem Operanden SNAP im Makro CDUMP2 kann die Zwischenspeicherung des Klasse-1-, Klasse-3- und des residenten Klasse-4-Speichers vor der eigentlichen Dumperstellung angefordert werden (nur privilegierte Anwender). Dieser so genannte SNAP-Dump wird bei der Erstellung des Systemdumps in den Dump eingearbeitet.

Der Operator erhält eine Meldung (`IDA0N52`) und kann daraufhin entscheiden, ob der Systemdump auf Platte oder Band ausgegeben werden soll. Die Ausgabe dieser Meldung kann mit den Systemparametern DUMPCTRL und DUMPSD# unterdrückt werden.

Ein Systemdump kann normalerweise nur aus dem privilegierten Systembereich (TPR) angefordert werden. Doch auch als nichtprivilegierter Anwender können Sie einen Systemdump anfordern, falls Sie zuvor Ihre Leseprivilegierung mit dem Kommando

```
/MODIFY-TEST-OPTIONS PRIVILEGE=*PARAMETERS(READ=m,WRITE=1)
```

auf einen Wert m >= 3 eingestellt haben. Dazu sind Sie nur berechtigt, wenn Ihnen diese Privilegierungsmöglichkeit im Benutzerkatalog eingeräumt wurde.
Ein Userdump kann bei vorliegender Privilegierung (siehe oben) in einen Systemdump umgewandelt werden (Kommando MODIFY-TEST-OPTIONS Operand DUMP=*SYSTEM). Die Meldung `IDA0N45` wird unterdrückt. Der Operator kann steuern, ob der Systemdump auf Platte oder auf Band ausgegeben wird.
Beim Systemdump wird, abhängig vom Wert des Systemparameters DUMPCTRL, bei einem abnormalen Dumpabbruch die Meldung `IDA0N99` ausgegeben.

Ein Systemdump, der auf Band ausgegeben wurde, muss mit dem Kommando COPY-FILE oder mit dem Dienstprogramm PERCON auf Platte kopiert werden. Anschließend ist eine Aufbereitung mit dem Diagnoseprogramm DAMP möglich.

Beispiel 1

```
/IMPORT-FILE SUPPORT=*TAPE
           VOLUME=<volume>,DEVICE=<device>,FILE-NAME=<dateiname> ————————  (1)
/ADD-FILE-LINK LINK=DMCOPY11,FILE-NAME=<dateiname>,ACCESS-METHOD=*UPAM,
           BUFFER-LENGTH=*STD(2) ————————————————————————————————————————  (2)
/ADD-FILE-LINK LINK=DMCOPY22,FILE-NAME=<ausgabedateiname>,
           ACCESS-METHOD=*UPAM,BUF-LEN=*STD(2) ——————————————————————————  (3)
/COPY-FILE FROM-FILE=<originaldateiname>,TO-FILE=<ausgabedateiname>————————  (4)
```

(1) Eintragen der Banddatei im Systemkatalog
(2) Vergabe des Linknamens DMCOPY11 für die Banddatei
(3) Vergabe des Linknamens DMCOPY22 für die Ausgabedatei
(4) Mit COPY-FILE wird anschließend der Systemdump von Band auf Platte opiert

Siehe auch Beschreibung von COPY-FILE im Handbuch „Kommandos“ [8].

Beispiel 2

```
/CREATE-FILE <output-filename> ———————————————————————————————————————  (1)
/ADD-FILE-LINK LINK-NAME=<link-name>,FILE-NAME=<output-filename>,-
/    ACCESS-METHOD=*PAM,BUFFER-LENGTH=*STD(SIZE=1)
/START-PERCON
//ASSIGN-INPUT-FILE FILE=*TAPE-FILE(NAME=<filename>) ——————————————————  (2)
//ASSIGN-OUTPUT-FILE FILE=*DISK-FILE(NAME=<output-filename>) ——————————  (3)
//START-CONVERSION ———————————————————————————————————————————————————  (4)
//END
```

(1) Erstellen der Ausgabedatei
(2) Angabe der Eingabedatei auf Band
(3) Angabe der Ausgabedatei auf Platte
(4) Konvertieren der Eingabedatei auf Band in die Ausgabedatei auf Platte

Dateiname eines Systemdumps

CDUMP legt einen Systemdump unter der Systemkennung SYSDUMP ab und erzeugt den Dateinamen nach folgendem Muster:

```
:catid:$SYSDUMP.{ ABSOLU | module }.pc.ec.tsn.date.time
```

Dabei bedeuten

catid	Katalogkennung des Public-Volume-Sets, auf dem der Dump abgelegt wurde. Wurde der Dump auf Band ausgegeben, erscheint keine Katalogkennung.
modul	Name des Moduls, in dem der Speicherauszug aktiviert wurde, max. 8 Zeichen
ABSOLU	Wird eingesetzt, wenn kein Modulname existiert
pc	Adresse im Befehlszähler (relativ zum Modulanfang bzw. absolut)
ec	Ereigniscode (sedezimal). Wurde der Systemdump mit dem Kommando CREATE-DUMP eingeleitet, hat der Ereigniscode ec den Wert 'C8'
tsn	TSN der Task, die den Speicherauszug aktiviert hat.
datum	Datum in der Form Dyymmdd (D=Aufmerker für Datumsbeginn, yy=Jahr, mm=Monat, dd=Tag).
uhrzeit	Uhrzeit in der Form hhmmss (hh=Stunde, mm=Minuten, ss=Sekunden)

4.2 Einfluss von Seitenattributen

Speicherseiten sind mit Attributen versehen. Folgende Seitenattribute wirken sich auf das Verhalten von CDUMP aus:

- **Secret Pages**

  Sie werden vom jeweiligen Benutzer mit dem Makro CSTAT als solche gekennzeichnet und sind dann besonders geschützt.
  Je nachdem, wie der Systemparameter DUMPSEPA (siehe Abschnitt „Steuerung durch Systemparameter") eingestellt ist, werden alle Secret Pages, nur Secret Pages aus bestimmten Speicherklassen oder gar keine Secret Pages in die Dumpdatei mit aufgenommen.

- **Trusted Pages**

  Sie sind mit einem Speicherschutzschlüssel versehen, der sich vom Speicherschlüssel des Benutzers unterscheidet.
  Solche Seiten einer Task werden wie folgt behandelt:

  Wird ein **Area- oder Userdump** aus einem Programmteil aufgerufen, der als „trusted“ gekennzeichnet ist, werden die Trusted Pages mit in den Dump aufgenommen und dieser unter der Kennung SYSUSER abgelegt. Wird der Area- oder Userdump aus einem Programmteil aufgerufen, der den normalen Speicherschlüssel des Benutzers besitzt, werden die Trusted Pages nicht mit in den Dump aufgenommen (Ausnahme: Common-Readable-Pages, s. u.) und im Dump als „not accessible“ gekennzeichnet. Der Area- oder Userdump wird dann auf der Benutzerkennung abgelegt.

  In einen **Systemdump** werden alle Trusted Pages aufgenommen.

- **Common-Readable-Pages**

  Sie werden vom Eigentümer mit dem Makro CSTAT gegebenenfalls als solche markiert (Klasse-6-Speicher) und sind dann allgemein zugänglich. Common-Readable-Pages werden auch dann im Dump sichergestellt, wenn sie als „Trusted Pages“ markiert sind.
  Sind sie jedoch (gleichzeitig) als „Secret Pages“ markiert, unterbleibt die Sicherstellung.

- **Read-protected Bereiche**

  Sind Programme oder Programmteile, die als besonders schützenswert (z.B. readprotected) gekennzeichnet sind, Bestandteil des Dumps, wird der entsprechende Userdump oder Areadump nicht auf die Kennung des Benutzers sondern auf die Kennung SYSUSER ausgegeben.

  Ein Programm oder Programmteil ist z. B. dann besonders schützenswert, wenn es/er aus einer Datei (nach) geladen wird, die mit einem Lesepasswort geschützt ist, aber der Benutzer dieses Passwort nicht mit dem Kommando /ADD-PASSWORD angegeben hat.

- **Freshly-obtained Seiten**

  Es sind vom Benutzer angeforderte und für ihn reservierte Seiten (allokierte Seiten), die jedoch noch nicht verwendet wurden.
  Freshly-obtained Seiten werden von CDUMP nicht in die Dumpdatei übertragen, sondern nur in der Indexstruktur als „freshly-obtained“ für den Dumpauswerter gekennzeichnet.

- **DIV-Seiten**

  DIV-Seiten stellen einen Datenbereich aus einer Datei dar, der im virtuellen Adressraum abgebildet und bearbeitet wird.
  Solche Seiten können auf Wunsch des Benutzers Bestandteil des Userdumps sein (Operand DIV im Makro CDUMP2 mit SCOPE=*USER). Wurde der Operand DIV im Makroaufruf nicht angegeben, so bestimmt der mit dem Operanden DATA-IN-VIRTUAL im Kommando MODIFY-TEST-OPTIONS eingestellte Wert das Verhalten bezüglich DIV-Seiten bei einem Userdump.

In einem Systemdump sind DIV-Seiten generell enthalten.

- **MMAP-Seiten**
  MMAP-Seiten sind Datenbereiche von POSIX-Dateien, die im virtuellen Adressraum abgebildet und bearbeitet werden. Diese Seiten können auf Wunsch des Benutzers Bestandteil des Userdumps sein (Operand MMAP im Makro CDUMP2). Wurde der Operand MMAP im Makroaufruf nicht angegeben, so bestimmt der mit dem Operanden MEMORY-MAP im Kommando MODIFY-TEST-OPTIONS eingestellte Wert das Verhalten bezüglich MMAP-Seiten bei einem Userdump.
  In einem Systemdump sind MMAP-Seiten generell enthalten.

Hinweis

Benutzerspeicherbereiche können auch in Datenräumen (Data Spaces) liegen. Alle Seiten eines Datenraumes besitzen dieselben Seitenattribute, mit Ausnahme der Attribute „allocated“ und „freshly obtained“. (Bei Seiten eines Programmraumes ist dies nicht der Fall.)

- Speicherbereiche aus Datenräumen können auf Wunsch des Benutzers Bestandteil des **Userdumps** sein (Operand DS im Makro CDUMP2 mit SCOPE=*USER). Wurde der Operand DS im Makroaufruf nicht angegeben, so bestimmt der mit dem Operanden DATA-SPACES im Kommando MODIFY-TEST-OPTIONS eingestellte Wert das CDUMP-Verhalten bezüglich Datenräumen in einem Userdump.
- In einem **Areadump** sind genau diejenigen Bereiche aus Datenräumen enthalten, welche der Benutzer beim CDUMP2-Aufruf angegeben hat (Operand DSCTRL im Makro CDUMP2 mit SCOPE=*AREA).
- In einem **Systemdump** sind nur solche Bereiche aus Datenräumen enthalten, welche das Betriebssystem über eine Schnittstelle ($DMPDEF(I)) bei CDUMP angemeldet hat.

4.3 Steuerung der CDUMP-Funktionen

Die CDUMP-Funktionen können durch Systemparameter und taskspezifische Einstellungen gesteuert werden.

4.3.1 Steuerung durch Systemparameter

Die Systemparameter werden im Startup-Parameterservice eingestellt und wirken sich systemweit auf die Dumpausgabe aus.

- Durch eine entsprechende Einstellung für den Systemparameter **DUMPSEPA** kann die Ausgabe besonders geschützter Speicherbereiche (Secret Pages, siehe "Einfluss von Seitenattributen") des Klasse-1- bis Klasse-6-Speichers unterdrückt werden.
- Die Ausgabe von privilegiertem Klasse-5-Speicher bei Area- und Userdumps kann durch Setzen des Systemparameters **DUMPCL5P** gesteuert werden.
- Mit dem Systemparameter **DUMPSD#** wird festgelegt, wie viele Systemdumps pro Session ohne Operator-Dialog ausgegeben werden sollen.
- Mit dem Systemparameter **DUMPCTRL** können folgende Funktionen gesteuert werden:
  - Duplikatserkennung bei Systemdumps (standardmäßig abgeschaltet)
  - Operatorloser Betrieb (standardmäßig abgeschaltet)
  - Ausgabe der Meldung `IDA0N99` (standardmäßig abgeschaltet)
  - Ausgabe der Meldung `IDA0N52` (standardmäßig eingeschaltet)
  - Verwendung der Operanden IOPERF= HIGH und IOUSAGE=WRITE beim FILE-Aufruf für die Dump-Ausgabedatei (standardmäßig eingeschaltet)
- Die Ausgabe des Klasse-6-Speichers bei Systemdumps kann durch den Systemparameter **DUMPSREF** gesteuert werden.
- Der Systemparameter **RDTESTPR** legt den Maximalwert der Lese-Testprivilegierung fest.

  Zur Erstellung eines Systemdumps aus dem Funktionszustand TU bzw. zur Umwandlung eines Area- oder Userdumps in einen Systemdump ist ein Wert >= 3 erforderlich.
- Wurde der Systemparameter **DESTLEV** >= **2** gesetzt, werden die Dumpdateien unter den Kennungen SYSDUMP und SYSUSER mit dem Operanden DESTROY-BY-DELETE=YES eingerichtet.

Weitergehende Informationen über die möglichen Werte und Operanden zu den Systemparametern entnehmen Sie bitte dem Handbuch „Einführung in die Systembetreuung“ [6].

4.3.2 Steuerung durch taskspezifische Einstellungen

Taskspezifische Festlegungen für die Dump-Erstellung können mit dem Kommando MODIFY-TEST-OPTIONS (siehe Handbuch „Kommandos“ [8]) getroffen werden.

Über den Operanden USERDUMP-OPTIONS ist die Ausgabe von User- und Areadumps für die eigene Task oder für andere Tasks unter der eigenen Benutzerkennung (diese Einschränkung gilt nicht für Kennungen mit dem Privileg TSOS) steuerbar.
Der Benutzer kann festlegen:

- ob auftretende User- oder Areadumps gezogen bzw. in Systemdumps (DUMP=*SYSTEM) umgewandelt werden sollen
- auf welchem Pubset die Dumps abgelegt werden sollen
- ob Dump-Duplikate unterdrückt werden sollen
- wieviele Dumps maximal gezogen werden sollen
- ob DIV-Fenster, Datenräume bzw. POSIX-Memory-Mapping-Bereiche im Userdump enthalten sein sollen

Über die Einstellung der tasklokalen Testprivilegierung mit dem Operanden PRIVILEGE kann gesteuert werden, ob:

- User- oder Areadumps in Systemdumps umgewandelt werden dürfen (DUMP=*SYSTEM)
- beim CDUMP2-Makro oder bei der Meldung `IDA0N45` Systemdumps angefordert werden dürfen

Dafür ist – entweder in der Task, in der der Dump auftritt, oder in der Task, die DUMP=*SYSTEM eingestellt hat, – eine Lese-Testprivilegierung >= 3 erforderlich.

Voraussetzung für die Einstellung einer tasklokalen Lese-Testprivilegierung >= 3 ist, dass für die Benutzerkennung, unter der die Task abläuft, ein ausreichend großer Wert vorliegt (siehe Kommandos ADD-USER bzw. MODIFY-USER-ATTRIBUTES, Operand TEST-OPTIONS).

4.4 Dumpspezifische Operanden in BS2000-Kommandos

Kommandos ADD-USER / MODIFY-USER-ATTRIBUTES

Mit dem Kommando ADD-USER legt die Systemverwaltung fest, welches Testprivileg der Benutzer maximal einstellen kann, siehe Handbuch „Kommandos“ [8]:
`ADD-USER TEST-OPTIONS=*PARAMETERS(READ-PRIVILEGE=...,MODIFICATION=...)`.

Mit dem Kommando MODIFY-USER-ATTRIBUTES kann die benutzerspezifische Testprivilegierung geändert werden.

> ¡ Sie können sich die aktuellen Einstellungen zur Privilegierung benutzerspezifisch mit dem Kommando SHOW-USER-ATTRIBUTES, Operand TEST-OPTIONS (und taskspezifisch mit dem Kommando SHOW-TEST-OPTIONS, Operand INFORMATION=*PRIVILEGE) anzeigen lassen.

Kommando MODIFY-TEST-OPTIONS

Mit dem Kommando MODIFY-TEST-OPTIONS kann der Benutzer taskspezifische Einstellungen für Test und Diagnose vornehmen.

Operand DUMP

Mit dem Operanden DUMP des Kommandos MODIFY-TEST-OPTIONS wird festgelegt, wie nach Ausgabe der Meldung `IDA0N51 PROGRAM INTERRUPT AT LOCATION ...` mit dem CDUMP-Speicherabzug des Benutzers verfahren wird. Ferner wird geregelt, ob ein User- oder Areadump in einen Systemdump umgewandelt werden soll.

> ¡ Sie können sich die aktuellen Einstellungen zur Ausgabe von User- und Areadumps mit dem Kommando SHOW-TEST-OPTIONS, Operand INFORMATION=USERDUMP-OPTIONS anzeigen lassen.

*DUMP=*STD (im Dialogbetrieb)*

CDUMP gibt folgende Meldung aus:

```
IDA0N45 DUMP DESIRED? REPLY (Y=USER-/AREADUMP TO DISK;
                      Y,<VOLUMETYPE>=USER-/AREADUMP TO TAPE;
                      Y,SYSTEM=SYSTEMDUMP TO DISK;
                      N=NO)
```

Antwortet der Benutzer mit N, so wird der Speicherabzug unterdrückt. Auf die Antwort Y oder Y,SYSTEM gibt CDUMP den Speicherabzug aus und meldet:
`IDA0N53 DUMP BEING PROCESSED. PLEASE HOLD ON`
Bei der Antwort Y,SYSTEM wird ein User- oder Areadump in einen Systemdump umgewandelt, wenn der Benutzer eine Lese-Testprivilegierung >= 3 besitzt. Bei Angabe von <VOLUMETYPE> wird der Dump auf Band ausgegeben.
Die Meldung `IDA0N53` wird unterdrückt, wenn CDUMP über das Kommando CANCEL-JOB DUMP= aufgerufen wird (siehe "Dumpspezifische Operanden in BS2000-Kommandos").

*DUMP=*STD (im Batch-Betrieb und in Prozeduren)*

Die Ausgabe des Speicherabzugs wird unterdrückt und folgende Meldung ausgegeben:
`IDA0N48 TASK/SYSTEM SETTINGS PROHIBIT DUMP`

*DUMP=*YES*

Die User- und Areadumps werden erstellt und folgende Meldung auf SYSOUT ausgegeben: `IDA0N53 DUMP BEING PROCESSED. PLEASE HOLD ON`
Die Meldung `IDA0N45` wird unterdrückt.

*DUMP=*NO*

Die Erstellung von User- und Areadumps wird unterdrückt und an Stelle der Meldung `IDA0N45` wird folgende Meldung auf SYSOUT ausgegeben:
`IDA0N47 DUMP PROHIBITED BY /MODIFY-TEST-OPTIONS COMMAND`

*DUMP=*SYSTEM*

Ein User- oder Areadump wird in einen Systemdump umgewandelt, wenn der Benutzer eine Lese-Testprivilegierung >= 3 besitzt.
Die Meldung `IDA0N45` wird unterdrückt.

Operand DUMP-CONTENTS

Mit dem Operanden DUMP-CONTENTS des Kommandos MODIFY-TEST-OPTIONS kann die Aufnahme von DIV-Fenstern, Datenräumen und POSIX-MMAP-Bereichen in den Userdump gesteuert werden. Um diese Steuerung wirksam zu machen, darf der Operand DUMP nicht auf *NO oder *SYSTEM gesetzt sein.

*DUMP-CONTENTS=*PARAMETERS(DATA-IN-VIRTUAL=*NO/*STD/*YES)*

Mit dem Operandenwert *NO ist keine Seite des DIV-Bereichs im Userdump enthalten. Die Operandenwerte *STD und *YES sind hier gleichbedeutend und veranlassen eine Aufnahme von DIV-Fenstern in den Userdump.

*DUMP-CONTENTS=*PARAMETERS(DATA-SPACES=*NO/*STD/*YES)*

Mit dem Operandenwert *NO ist keine Seite des DS-Bereichs im Userdump enthalten. Die Operandenwerte *STD und *YES sind hier gleichbedeutend und veranlassen eine Aufnahme von Datenräumen in den Userdump.

*DUMP-CONTENTS=*PARAMETERS(MEMORY-MAP=*STD/*YES/*NO)*

Die Angabe *STD oder *YES veranlasst die Aufnahme von POSIX-Seiten in den Userdump, mit der Angabe *NO wird keine Seite des Bereichs MMAP in den Userdump aufgenommen.

Operand TSN

Mit dem Operanden TSN kann gesteuert werden, für welche Task die Einstellungen für die Dumpausgabe geändert werden sollen.

*TSN=*OWN*

Die Änderungen an den Einstellungen für die Dumpausgabe gelten für die eigene Task.

TSN=<alphanum-name 1..4>/ <c-string 1..4>

Die Änderungen an den Einstellungen für die Dumpausgabe gelten für die Task mit der angegebenen TSN. Anwender, die nicht das TSOS-Privileg besitzen, können die Einstellungen nur für Tasks modifizieren, die unter der eigenen Benutzerkennung ablaufen.

Operand OUTPUT-PUBSET

Mit dem Operanden OUTPUT-PUBSET kann der Pubset bestimmt werden, auf dem User- oder Areadumps abgelegt werden.

*OUTPUT-PUBSET=*DEFAULT-PUBSET*

Ein User- oder Areadump wird auf dem Default-Pubset abgelegt, der für die Benutzerkennung der verursachenden Task vereinbart ist, wenn der Dump keine schutzwürdigen Daten enthält. Andernfalls wird der Dump auf dem Default-Pubset der Benutzerkennung SYSUSER abgelegt.

OUTPUT-PUBSET=<cat-id 1..4>

Ein User- oder Areadump wird auf dem angegebenen Pubset abgelegt.

Falls der Dump nicht auf dem angegebenen Pubset abgelegt werden kann, z. B. weil dort kein Speicherplatz (mehr) verfügbar ist, wird er auf dem Default-Pubset abgelegt.

Operand MAXIMUM-NUMBER

Mit dem Operanden MAXIMUM-NUMBER kann ein Maximalwert für die Anzahl zu erstellender User- und Areadumps festgelegt werden.

*MAXIMUM-NUMBER=*UNLIMITED*

Die maximale Anzahl der User- und Areadumps ist nicht begrenzt.

MAXIMUM-NUMBER=<integer 1..255>

Die Anzahl der User- und Areadumps wird auf den angegebenen Wert begrenzt.

*Operand SUPPRESS-DUPLICATES=*NO/*YES*

Mit dem Operanden SUPPRESS-DUPLICATES wird festgelegt, ob die Ausgabe eines User- oder Areadumps unterdrückt wird, falls es sich um ein Duplikat eines bereits erstellten Dumps handelt. Ein Dump wird als Duplikat betrachtet, wenn im selben Programm an derselben Stelle ein Dump mit demselben Unterbrechungsgewicht aufgetreten ist.

Operand PRIVILEGE

Mit dem Operanden PRIVILEGE=*PAR(READ=...) kann die taskspezifische Lese-Testprivilegierung eingestellt werden.
Ein nicht privilegierter Anwender kann nur dann einen Systemdump anfordern, wenn er ein Leseprivileg >= 3 hat (siehe Kommando ADD-USER).

Kommando CANCEL-JOB

Das Kommando CANCEL-JOB bricht einen Auftrag ab. Der Operand DUMP steuert, ob für den abzubrechenden Auftrag ein Userdump auszugeben ist bzw. ob ein gerade in Bearbeitung befindlicher User- oder Areadump abgebrochen werden soll.

Operand DUMP

Der Operand DUMP vereinbart, ob für den abzubrechenden Auftrag ein Dump auszugeben ist.

*DUMP = *NO*

Es wird kein Speicherauszug angefordert. Ist für den abzubrechenden Auftrag jedoch bereits ein Dump in Bearbeitung, wird dieser vollständig erstellt.

*DUMP = *STD*

Wurde mit dem Kommando MODIFY-TEST-OPTIONS der Operand DUMP=*YES oder DUMP=*STD eingestellt, wird ein Dump gezogen (auch wenn *STD ein *NO bedeuten würde, was im Batch- oder Prozedurbetrieb der Fall ist). Bei explizitem DUMP=*NO wird kein Dump gezogen. Bei DUMP=*SYSTEM wird ein Systemdump gezogen.

*DUMP = *CANCEL-RUNNING-DUMP*

Ist für den abzubrechenden Auftrag ein User- oder Area-Dump in Bearbeitung, wird dieser unverzüglich abgebrochen und die Dumpdatei gelöscht.

Kommando MODIFY-SYSTEM-PARAMETERS

Mit dem Kommando MODIFY-SYSTEM-PARAMETERS kann der Systemverwalter die CDUMP-Systemparameter einstellen.

4.5 Ablaufmeldungen

Der Operator wird an der Konsole über die Ursache des System-Speicherauszugs informiert.

In Abhängigkeit des CDUMP2-Parameters DIAG wird folgende Meldung ausgegeben oder nicht (aber nur bei Anforderung des Systemdumps über den Makro CDUMP2) :

```
IDA0N50     SYSTEMDUMP BEI ADRESSE '(insert)' AUFGERUFEN
```

```
insert = { <CDUMP-Adr>.(modul+nnnnn),EC=...[,ELSN=elsnr#] |
           <CDUMP-Adr>.(ABSOLUT),EC=...[,ELSN=elsnr#] }{ [,CODE=code] |
           [,INSERT=insert] }
```

Grundsätzlich wird folgende Meldung ausgegeben:

```
IDA0N51     PROGRAMM-UNTERBRECHUNG BEI ADRESSE '(insert)'
```

```
insert = { <Fehleradr>.(modul+nnnnn),EC=...[,ELSN=elsnr#] |
           <Fehleradr>.(ABSOLUT),EC=...[,ELSN=elsnr#] }{ [,CODE=code] |
           [,INSERT=insert] }
```

`CDUMP-Adr`	sedezimale CDUMP-Aufrufadresse
`Fehleradr`	sedezimale Fehleradresse
`modul`	Modulname
`EC`	Ereigniscode
`ELSN`	Nummer des Error-Logging-Sequenz-Blockes
`CODE`	Ursache des Speicherauszuges
`INSERT`	Zeichenstring zur Spezifizierung der Ursache

Danach wird der Operator in der Regel mit folgender Meldung gefragt, ob er eine Speicherausgabe wünscht und welches Ausgabemedium (Platte oder Band) er wählt:

```
IDA0N52   SYSTEMDUMP AUSGEBEN? ANTWORT (EOT=AUSGABE AUF PLATTE;
          <volumetyp>=AUSGABE AUF BAND; N=NEIN)
```

Der Operator kann folgende Eingaben machen:

<tsn>.EOT	->	Ausgabe auf Platte
<tsn>.<gerätebezeichnung>	->	Ausgabe auf Platte
<tsn>.N	->	Speicherauszug unterdrücken

Für <gerätebezeichnung> sind Angaben wie im Operanden DEVICE-TYPE des Kommandos CREATE-FILE (siehe Handbuch „Kommandos“ [8]) erlaubt, z.B.:

`TA`	beliebiges Bandgerät
`TAPE`	beliebiges Bandgerät
`T-C4`	Bandgerät vom Volume-Typ TAPE-C4

Alternativ dazu führt auch die Beantwortung der folgenden Meldung (die beim Auftreten eines User-/Areadumps nach SYSOUT ausgegeben wird) zur Auswahl des Ausgabemediums:

```
IDA0N45  SOLL EIN DUMP AUSGEGEBEN WERDEN? ANTWORT:
         (Y=USER-/AREADUMP AUF PLATTE;
         Y,<volumetype>=USER-/AREADUMP AUF BAND;
         Y,SYSTEM=SYSTEMDUMP; N=NEIN) ?
```

Die Meldungen werden erneut ausgegeben, wenn eine ungültige Bandgerätebezeichnung eingegeben wurde.

Wird eine Bandgerätebezeichnung eingegeben, die gültig, das angeforderte Gerät aber nicht verfügbar ist, so wird an der Konsole die Meldung `IDA0N58` so lange wiederholt, bis eine korrekte Eingabe erfolgt ist.

```
IDA0N58   BANDGERAETE-TYP NICHT VERFUEGBAR? ANTWORT:
          (EOT=AUSGABE AUF PLATTE;
          <volumetype>=AUSGABE AUF BAND; N=UNTERDRUECKE BANDAUSGABE) ?
```

Nach der Meldung `IDA0N52` bzw. `IDA0N58` folgen Meldungen der Geräteverwaltung und des Datenverwaltungssystems zur Steuerung der Dumpausgabe (z.B. MOUNT-Meldungen, Anforderungen des Datenträgers usw.).

Tritt bei der Ausgabe des Speicherauszugs ein Fehler im System auf, so wird in einer Meldung (`IDA0N63`) ein Fehlerhinweis eventuell mit einem Fehlercode ausgegeben. Gegebenenfalls wird die Ausgabe abgebrochen (`IDA0N61`-Meldung).

Die ordnungsgemäße Beendigung der Ausgabe des System-Speicherauszugs wird mit der folgenden Meldung angezeigt:

```
IDA0N54  SYSTEMDUMP AUSGEGEBEN IN DATEI '$SYSDUMP....'
```

Wurde der Dump auf Band ausgegeben, muss er mit dem Kommando COPY-FILE oder mit dem Dienstprogramm PERCON auf Platte kopiert werden, bevor er mit DAMP bearbeitet wird.

5 DAMP Diagnoseobjekte auswerten

Hinweise zur Versionsdarstellung in diesem Kapitel:

- DAMP wird als Kurzform für **DAMP V21.0** (BS2000 V21.0) verwendet
- Die Zeichenfolge `<version>` bezeichnet in Beispielausgaben die aktuelle Version von DAMP, also in dieser Version von DAMP: `<version>=V21.0A00`.
- Die Zeichenfolge `<ver>` bezeichnet in Dateinamen die aktuelle Version von DAMP, also in dieser Version von DAMP: `<ver>=210`.
- Die Zeichenfolge `<ver>` bezeichnet in Beispielen auch die aktuelle Version von BS2000 (`<ver>=21.0`), eines Dumperzeugers, von VM2000, von Systemprogrammen oder eines Diagnoseobjekts.

5.1 Leistungsbeschreibung

DAMP ist ein Programm zur Auswertung von Diagnoseobjekten im Dialogbetrieb. Das Diagnoseobjekt kann eine Dumpdatei oder ein aktives BS2000-System sein.

Dumpdateien, die ein BS2000-System enthalten, sowie das aktive BS2000-System können mit DAMP virtuell und symbolisch ausgewertet werden. Darüber hinaus wird auch eine reale Auswertung von Dumpdateien, die ein fremdes Betriebssystem oder ein BS2000-System mit beschädigten BS2000-Strukturen enthalten, unterstützt. Sind die Dumpdatei-Strukturen beschädigt, so ist eventuell noch eine „Notauswertung“ als PAM-Datei möglich.
Um mit DAMP zu arbeiten, sind gewisse System- und Diagnose-Kenntnisse erforderlich. Eine geführte Diagnose am Bildschirm ist mit DAMP nicht möglich.

Das hier beschriebene Programm **DAMP** ist auf folgenden Servern bzw. mit folgenden Betriebssystem-Versionen ablauffähig :

- auf /390-Servern: ab BS2000/OSD-BC V11.0
- auf x86-Servern[1]: ab BS2000/OSD-BC V11.0 (OSD/XC V11.0)

[1]Auf x86-Servern läuft DAMP emuliert ab

DAMP kann Dump-Dateien unabhängig von der aktiven BS2000-Version verarbeiten, wenn sie auf folgenden Servern erstellt wurden:

- auf /390-Servern ab BS2000/OSD-BC V8.0
- auf x86-Servernab BS2000/OSD-BC V8.0 (OSD/XC V4.0)

Zur Auswertung von Dumpdateien aus BS2000 V21.0 benötigt DAMP die Zugriffsmethode ANITA V21.0. ANITA V21.0 ist mit einer Korrekturlieferung für BS2000/OSD-BC ab V11.0 verfügbar.

5.1.1 Diagnoseprotokoll

Alle Diagnoseaktionen sowie alle Bildschirmausgaben können protokolliert werden. Die Ein- und Ausgaben werden dabei in eine Datei geschrieben. Durch Auswerten des Protokollausdrucks bzw. durch Wiedergabe der protokollierten Sitzung am Bildschirm lässt sich der Diagnoseweg nachträglich verfolgen.

5.1.2 Listen erzeugen

Da die Diagnose interaktiv am Bildschirm durchgeführt wird, kann man weitgehend auf das Erzeugen von Listen verzichten. Sollen dennoch Listen erzeugt werden, lassen sich Art und Menge der zu druckenden Informationen bequem auswählen.

5.1.3 Diagnosevorgänge automatisieren

Häufig wiederkehrende Aktionen können automatisiert werden. Eine automatische Voranalyse beispielsweise ist in der Lage, die diagnoserelevanten Seiten eines Dumps herauszufinden und die Fehlerkomponente einzugrenzen. Mit der Diagnosesprache PRODAMP kann der Diagnostiker beliebige, auch an Entscheidungen gebundene Anweisungen zu Diagnoseprozeduren zusammenstellen, diese in Bibliotheken sammeln und nach Bedarf aufrufen.

5.1.4 Weitere Funktionen

Sollen zusätzliche Meldungs- und Fehlerdateien wie etwa CONSLOG oder SERSLOG für die Diagnose herangezogen werden, können Sie den EDT als Unterprogramm aufrufen. Ferner können PRODAMP-Prozeduren mithilfe des EDT erstellt oder verändert werden.

DAMP kann Speicherbereiche aufbereiten, zum Beispiel im Layout von BS2000-Dummy-Sections. Dazu greift DAMP auf Elemente einer Symbolbibliothek zurück.

Neben der Standard-Symbolbibliothek des BS2000-Grundausbaus können Sie für spezielle Analyseanwendungen auch eigene, private Symboldateien verwenden, die Sie mit DAMP selbst erzeugen können.

5.1.5 Verhalten im Falle eines Programm- oder Systemfehlers

Tritt ein irregulärer Programm- oder Systemzustand auf, so gibt DAMP eine kurze Diagnoseinformation auf SYSOUT aus und beendet sich dann mit Speicherauszug (Userdump).

Ist jedoch der Auftragsschalter 30 gesetzt und läuft DAMP im Dialogmodus, so erfolgt eine Rückfrage, ob ein Speicherauszug veranlasst werden soll.
Ist der Auftragsschalter 30 gesetzt und läuft DAMP im Batchmodus, so unterbleibt der Speicherauszug.

Vermuten Sie eine Endlosschleife in DAMP, so sollte DAMP durch Drücken der Taste K2 unterbrochen werden. Mit dem Kommando INFORM-PROGRAM ist der weitere Ablauf steuerbar (siehe Eingabe von DAMP-Anweisungen über das Systemkommando INFORM-PROGRAM auf "Auf Systemebene"). Insbesondere führt die Eingabe von `/INFORM-PROGRAM MSG='?'` zu einem geführten SDF-Dialog und `/INFORM-PROGRAM MSG='*DUMP'` beendet das DAMP-Programm mit einem Speicherauszug.

5.1.6 Auswertbare Diagnoseobjekte

- Aktives System
- Dumpdateien
- PAM-Datei als Diagnoseobjekt

5.1.6.1 Aktives System

Das Diagnoseobjekt „aktives System“ (CURRENT SYSTEM) enthält immer ein BS2000-System. Bei der Diagnose ist zu berücksichtigen, dass das System während der Diagnosesitzung weiterarbeitet und somit einem permanenten Wandel unterliegt. Für die Diagnose des aktiven Systems ist die Lese-Testprivilegierung 8 erforderlich (siehe Handbuch „Kommandos“ [8], Kommandos MODIFY-TEST-OPTIONS und SHOW-TEST-OPTIONS).

5.1.6.2 Dumpdateien

DAMP kann nicht nur Dumpdateien auswerten, die ein BS2000-System enthalten, sondern grundsätzlich alle Dateien, die auf Grund ihrer Metadaten von DAMP als BS2000-Dumpdatei erkannt und real ausgewertet werden können. Ein BS2000-System kann virtuell und symbolisch ausgewertet werden.

SLED und SNAP

Die Diagnoseobjekte SLED und SNAP werden von den gleichnamigen Dumpdatei-Erzeugern erstellt und sind virtuell oder real auswertbar. SNAP enthält immer das Betriebssystem BS2000. Ein SLED kann auch für fremde Betriebssysteme erstellt werden. Beide Arten von Dumpdateien werden im Ruhezustand des Systems erzeugt:

- SLED nach einem Systemabbruch
- SNAP während einem kurzzeitigen Anhalten des Systems (maximal 24 Sekunden)

Der Umfang ist jeweils über Parameter steuerbar, wobei beim SNAP wegen der Zeitgrenze deutliche Einschränkungen bestehen.

Ein SLED kann mehrere Produkte enthalten:

- VM2000-Gesamtsled

  Ein VM2000-Gesamtsled ist eine SLED-Datei, die Daten des Produkts VM2000 sowie Daten von VM2000-Gastsystemen enthält. VM2000-Gastsysteme laufen als virtuelle Maschinen (VM) auf einem VM2000-System ab. Sie können sowohl BS2000 als auch fremde Betriebssysteme enthalten.
- SLED vom SLED

  Ein SLED vom SLED entsteht, wenn vor Ablauf des SLEDs bereits ein weiterer SLED geladen wurde. Ein SLED vom SLED ist eine Datei, die Daten des Produkts SLED sowie Daten des zuvor abgelaufenen Produkts bzw. der zuvor abgelaufenen Produkte (bei VM2000) enthält.

Systemdump, Userdump und Areadump

Die Diagnoseobjekte SYSDUMP, USERDUMP und AREADUMP werden vom Dumpdatei-Erzeuger CDUMP erstellt. Sie enthalten immer Daten von BS2000 und sind nur virtuell auswertbar. Die Daten werden während des Systemablaufs erzeugt und enthalten Speicherbereiche von einer Task und ausgewählte Systembereiche. Beim Systemdump werden Diagnoseunterlagen für den Systemdiagnostiker, beim Userdump solche für den nichtprivilegierten BS2000-Anwender erstellt. Ein Areadump enthält vom nichtprivilegierten Anwender festgelegte Speicherbereiche.

SELF-LOADER

Jede beliebige, mit DAMP real verarbeitbare Dumpdatei kann vom Anwender mit der Eigenschaft SELF-LOADER geöffnet werden. Hiermit wird die reale Auswertung von Dumpdateien (SLEDs und SNAPs), die ein fremdes Betriebssystem oder ein BS2000-System mit beschädigten BS2000-Strukturen enthalten, ermöglicht.

5.1.6.3 PAM-Datei als Diagnoseobjekt

Auch Dateien, die nicht das BS2000-Dumpformat haben, können (mit gewissen Einschränkungen) mit DAMP verarbeitet werden. Dazu müssen sie mit der Eigenschaft PAM geöffnet werden. Diese Funktion ist hauptsächlich als „Notauswertung“ von beschädigten Dumpdateien gedacht.

5.1.7 Online-Hilfetexte

Während einer Diagnosesitzung können Sie bei Bedarf Hilfe bei DAMP anfordern. Wenn Sie in eines der Eingabefelder ein Fragezeichen (?) eintragen und die DUE-Taste drücken, liefert DAMP die zu diesem Feld passende Beschreibung. Dazu schaltet DAMP in den EDT um. Durch Blättern können Sie sich dort auch weitergehende Informationen zu DAMP besorgen. Nach Beenden des EDT können Sie Ihre Diagnosesitzung fortsetzen. Voraussetzung für diese Hilfe-Funktion ist der EDT-Modus '@VDT F2'. Dieser steht nur auf den Datensichstationen 9763 zur Verfügung.

Online-Hilfetexte stehen in deutscher und englischer Sprache zur Verfügung.

Eine weitere (weniger komfortable) Möglichkeit der Hilfe bietet das Help-Fenster (siehe "Das Help-Fenster (W1)").

5.1.8 Verwendete Begriffe

DAMP-Bildschirm	Alle Daten, die von DAMP auf einem Bildschirm gezeigt werden. Auf einem DAMP-Bildschirm können gleichzeitig Inhalte eines oder mehrerer Diagnosefenster dargestellt werden.
Diagnosefenster	Attribute eines Diagnosefensters sind die Bezeichnung (W0 - W9, W21 - W99), der Inhalt und die Darstellung des Inhalts. Einem Teil der Diagnosefenster (W0 - W3) sind feste Inhalte zugeordnet, dem Rest (den Dumpfenstern W4 - W9 und W21 - W99) können unterschiedliche Inhalte zugewiesen werden. Auf dem DAMP-Bildschirm ist im Allgemeinen nur ein Ausschnitt des gesamten Inhalts des Diagnosefensters dargestellt.
Dumpfenster	Die Fenster mit der Bezeichnung W4 - W9 und W21 - W99 sind die Dumpfenster. Sie werden auch, je nach Fensterinhalt, als Standard-Dumpfenster oder Spezialfenster bezeichnet.

5.2 Bildschirmformat

5.2.1 Bildschirmmaske

Die DAMP-Bildschirmmaske hat einen einheitlichen Aufbau, wobei folgende Zeilen immer die gleiche Bedeutung haben:

1 **Titelzeile**
Anzeige der DAMP-Version und von Metainformation über das Diagnoseobjekt

2-3 **Meldungszeilen**
Ausgabezeilen für DAMP- und Systemmeldungen

4-22 **Diagnosefeld**
Anzeige eines oder mehrerer Diagnosefenster mit jeweils einer oder mehreren Kopfzeilen und einer Trennzeile zwischen den Fenstern.

23 **Kommandozeile**
Eingabezeile für DAMP-Anweisungen und Systemkommandos

24 **Key-Zeile**
Anzeige der Belegung der 9 Diagnosefenster W1 bis W9, die mit den P-Keys (P-Tasten) einstellbar sind (die Belegung der übrigen Diagnosefenster wird nur im Fenster W0 angezeigt).

```
Zeile
1       DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>
2
3
4       Analyzed Object: BS2000   V<ver>            TID=000A00CB TSN=HSMS W2,PLK,L 8
5       Dumpfile: :1DQM:$DIAGDUMP.A0550438.DHSIHSM@.00502.50.HSMS.D06081 on /390-HSI/VM
6       MemSize: 256.0 MB  ShareB: 00C00 UserXB: 01000 SysB: 71000
7
8       PCB#  PCB-Ad    IS-LNK    SR         Program Counter          SVC/IW      A_MODE
9          1  739073A8  72F91008  070C0C00   71542CBE=CDUMPF1 +0117E  EE=$PNUP        31
10         2  72F91008  732DC578  070C0C00   7139ECB0=NRTINIT +01670  1A=CDUMP        31
11         3  732DC578  00000000  070C0C00   7C584BC2=DHSIHSM@+00502  ED=TPR-Ter      31
12      ------------------------------------------------------------------------------
13      ETCB                            +00344=720E2AEC TID=000A00CB          W5,CBA,L 3
14      344 ETCBSTA : 739073A8                 | 348 ETCBTRC@: 00000000 =     0
15      34C ETCBCTRC: 71FEC420                 | 350 ETCBADM1: 00000000 =     0
16      ------------------------------------------------------------------------------
17      DHSIHSM@                        +004FC=7C584BBC TID=000A00CB          W4,ASS,L 6
18      7C584BBC (04FC):   41 10 DD88        =   LA    R1,3464(,R13)
19      7C584BC0 (0500):   0A ED             =   SVC   237                   (TPR-Ter)
20      7C584BC2 (0502):   98 A6 D0D0        =   LM    R10,R6,208(R13)
21      7C584BC6 (0506):   07 FE             =   BR    R14
22      7C584BC8 (0508):   00060202          =   DC    X'00060202'
23      CMD:
24      Key: 1=Help 2=Plk 3=PCB 4=U7C584 5=ETCB   6=EXVT   7=MEMA   8=SUSY   9=FIND
```

Bild 1: Die DAMP-Bildschirmmaske

Die Maske in Bild 1 hat folgenden Aufbau:
Titelzeile (Zeile 1), zwei Meldungszeilen (2-3), Statusfenster (W2) in der Länge 8 Zeilen (4-11), Trennzeile (12), Fenster W5 im symbolischen Format in der Länge 3 Zeilen (13-15), Trennzeile (16), Fenster W4 im Assembler-Layout in der Länge 6 Zeilen (17-22), Kommandozeile (23) und Key-Zeile (24).

Die einzelnen Zeilen der DAMP-Bildschirmmaske haben folgende Funktion:

Die Titelzeile (Zeile 1)

Die Titelzeile zeigt an:

- die DAMP-Version

- den Typ des Diagnoseobjekts
- die Version des Dumperzeugers (nur bei einer Dumpdatei)
- Name und Version des im Diagnoseobjekt enthaltenen Produkts
- Tag und Uhrzeit der Erstellung des Diagnoseobjekts

Im zweiten Feld (Typ des Diagnoseobjekts) können folgende Informationen stehen:

- CURRENT SYSTEM
- SLED (einschließlich VM2000-Gesamtsled und SLED vom SLED)
- SNAP
- SYSDUMP
- USERDUMP
- AREADUMP
- SELFLOADER
- PAM FILE

Siehe auch Abschnitt „Auswertbare Diagnoseobjekte".

Enthält ein Diagnoseobjekt (zum Beispiel eine VM2000-SLED-Datei) Daten von mehreren Produkten, so bezieht sich Name und Version auf das Produkt, das unmittelbar nach dem Öffnen angezeigt wird (bei einer VM2000-Datei ist dies das Produkt VM2000). Diese Information bleibt auch dann in der Titelzeile erhalten, wenn anschließend ein anderes Produkt aus dem Objekt ausgewählt wird.

Weitere Informationen über das Diagnoseobjekt können im Modus INF im Status-Fenster (W2) ausgegeben werden.

```
DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>
```

Bild 2: Titelzeile bei einem Systemdump als Diagnoseobjekt

```
DAMP <version> CURRENT SYSTEM from BS2000(<ver>)   <date> <time>
```

Bild 3: Titelzeile beim aktiven System als Diagnoseobjekt

Die Meldungszeilen (Zeile 2 und 3)

Die Meldungszeilen zeigen Meldungen des DAMP-Systems an. Mit den Programm-Tasten P14 bzw. P15 können Sie in der Meldungs-Historie zurück- bzw. vorblättern.

Das Blinken der Meldungen kann mit der Benutzeroption „Blinking“ ein- oder ausgeschaltet werden (siehe „Column separator (list)" (Modifikationen durch den Benutzer (Spezialfenster OPTIONS))).

```
DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>
DMP8751 CLASS  1 SEARCH INTERRUPTED; <F3> CONTINUE, <F1> CANCEL

FIND - Command                                SYS=000A00CB          21,D  ,L16
```

Bild 4: Meldungen des DAMP-Systems in den Zeilen 2 und 3

Im EDT-Modus werden diese Meldungen auf den beiden letzten Datenzeilen des EDT-Bildschirmes ausgegeben.

Das Diagnosefeld (Zeilen 4 bis 22)

Im Diagnosefeld werden die gewünschten Diagnosefenster (siehe "Diagnosefenster") eingeblendet. Sie enthalten jeweils eine oder mehrere Kopfzeilen und Information aus dem Diagnoseobjekt oder Helptexte. Die Trennzeile kann optional ausgeschaltet werden. Die Ersatzzeichen für nichtabdruckbare Zeichen sowie für Fenster- und Spalten-Trennzeichen können je nach generiertem Terminal eingestellt werden.

Die Kommandozeile (Zeile 23)

Die Kommandozeile beginnt mit CMD: In der Kommandozeile werden die DAMP-Anweisungen entgegengenommen (mögliche Anweisungen siehe "Programmanweisungen"). Dort können auch die zulässigen Systemkommandos eingegeben werden.

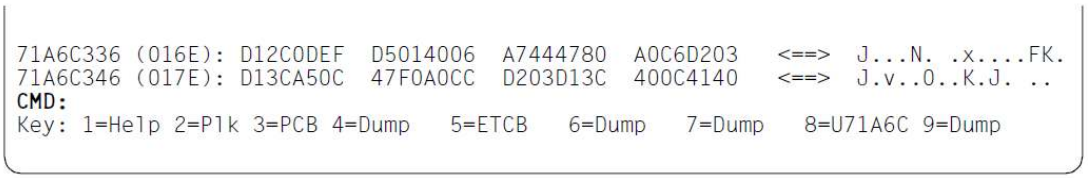


Bild 5: Eingabebereich für DAMP-Anweisungen und Systemkommandos

Führt das in der DAMP-Maske eingegebene Systemkommando zu einer Systemmeldung, so wird diese in den Zeilen 2 und 3 (Meldungszeilen) ausgegeben.

Durch Eingabe eines Fragezeichens bzw. eines Anweisungsnamens gefolgt von „?“ wird in den geführten SDF-Dialog verzweigt. In Bildschirmmasken kann die Anweisung durch Ausfüllen der entsprechenden Felder vervollständigt und durch Auslösen der Execute-Funktion an DAMP übergeben werden.

In die Kommandozeile können direkt BS2000-Systemkommandos eingegeben werden. Sollte es wegen Namensgleichheit mit DAMP-Anweisungen zu Unklarheiten kommen, kann durch Voranstellen eines Kommandolabels, z.B. /.LABEL, bestimmt werden, dass die Eingabe als Systemkommando interpretiert wird.

Die Key-Zeile (Zeile 24)

In der Key-Zeile wird die aktuelle Belegung der 9 Diagnosefenster W1 bis W9, die mit den P-Keys (P-Tasten) einstellbar sind, angezeigt, unabhängig davon, ob die Fenster auf dem Bildschirm sichtbar sind oder nicht. Die Belegung der übrigen Diagnosefenster wird nur im Fenster W0 angezeigt. Die Beschreibung der Diagnosefenster finden Sie auf "Diagnosefenster".

Die Dumpfenster (W4 - W9 und W21 - W99) sind zunächst frei verfügbare Standard-Dumpfenster. Für die Dumpfenster W4 - W9 ist dies jeweils in der Keyzeile an der Anzeige „Dump“ erkennbar.

Wurde mit einem der Dumpfenster (W4 - W9) gearbeitet und ist dieses Fenster kein Spezialfenster, erscheint an der zugehörigen Position in der Key-Zeile die Seite des System- oder Benutzerspeichers, aus der momentan ein Ausschnitt im Fenster gezeigt wird.

Folgende Anzeige wird in der Key-Zeile gesetzt:

Snnnnn	bei Systemspeicher
Unnnnn	bei Benutzerspeicher
Dnnnnn	bei Datenräumen

Wird der Ausschnitt im symbolischen Format gezeigt, so erscheint der Kontrollblock-Name (ggf. gekürzt auf 6 Zeichen).

Bei Ausgabe von Hardware-Informationen wird im zugehörigen Feld der Key-Zeile folgende Anzeige gesetzt:

Rnnnnn	für Ausgabe mit Real-Adressierung (ASEL=RM); nnnnn: Adresse relativ zum dazugehörigen 4GB-Segment
Annnnn	für Ausgabe mit Absolut-Adressierung (ASEL=ABS); nnnnn: Adresse relativ zum dazugehörigen 4GB-Segment
Hnnnnn	für Ausgabe der Hardware-System-Area (ASEL=HSA)
PSSnnn	für Ausgabe des Processor-Saved-Status (ASEL=PSS); nnn: Prozessornummer
snnnnn	bei Ausgabe einer Dumpfile-Section (ASEL=SCT)

Wird einem Fenster eine Spezialfunktion zugewiesen, so erscheint diese Funktion ebenfalls an der zugehörigen Stelle in der Key-Zeile. Angezeigt wird

AUDI	bei	SHOW-EDITED-INFORMATION INFORMATION=*AUDIT-TABLE-EDIT
FILE	bei	SHOW-EDITED-INFORMATION INFORMATION=*DUMPED-SYSTEM-FILE
FIND	bei	START-PATTERN-SEARCH
LIST	bei	START-LIST-GENERATION
MEMA	bei	SHOW-EDITED-INFORMATION INFORMATION=*MEMORY-ATTRIBUTES
OPTS	bei	START-OPTION-DIALOG
PROC	bei	START-PRODAMP-EDITOR
SUSY	bei	SHOW-EDITED-INFORMATION INFORMATION=*SUBSYSTEM-INFORMATION
TABL	bei	SHOW-EDITED-INFORMATION INFORMATION=*TASK-TABLES
TRAC	bei	SHOW-EDITED-INFORMATION INFORMATION=*TRACE-TABLE-EDIT

Damit lässt sich jederzeit erkennen, welche Diagnosefenster mit welchem Speicherausschnitt belegt und welche Diagnosefenster für eine Neuzuweisung noch frei verfügbar sind. Mit der Anweisung `SHOW-EDITED-INFORMATION INFORMATION=*STORAGE-EDIT` kann die Verwendung als Spezialfenster wieder aufgehoben werden, es wird dann wieder „Dump“ angezeigt.

```
068 ETCB@19@ : 70F8E000                | 06C ETCB@20@ : 70F81F40
070 ETCB@21@ : 00000000 =     0        | 074 ETCB@22@ : 00000000 =     0
CMD:
Key: 1=Help 2=Plk 3=PCB 4=S71234 5=D705FE 6=U70F12 7=ETCB   8=Dump   9=TRAC
```

Bild 6: Key-Zeile

Zeile 24 ist die Key-Zeile, sie zeigt:

- Fenster 1 bis 3 haben die feste Zuordnung als Help-, Status- bzw. Stack-Fenster
- Fenster 4 enthält die virtuelle Systemseite 71234
- Fenster 5 enthält die virtuelle Seite 705FE eines Datenraums
- Fenster 6 enthält die Benutzerseite 70F12
- Fenster 7 enthält den TCB der aktuellen Task in symbolischer Aufbereitung
- Fenster 8 ist nicht belegt
- Fenster 9 ist mit der Trace-Table belegt

5.2.2 Diagnosefenster

Für die Diagnose am Bildschirm stehen insgesamt 89 Diagnosefenster zur Verfügung. Die Fenster werden mit W0 bis W9 bzw. W21 bis W99 bezeichnet (W10 bis W20 sind nicht als Diagnosefenster nutzbar). Es handelt sich dabei um

- das Übersicht-Fenster (W0)
- das Help-Fenster (W1)
- das Status-Fenster (W2)
- das Stack-Fenster (W3)
- die Dumpfenster (W4 - W9 und W21 - W99)

Auf einem Bildschirm sind maximal 19 Zeilen eines oder mehrerer Fenster einschließlich der Trenn- und Kopfzeile (n) darstellbar.

Den Fenstern können im Laufe der Diagnosesitzung Inhalte zugeordnet werden:
Das Fenster W0 enthält die aktuelle Belegung aller 89 Diagnosefenster, dem Fenster W1 ist fest die Online-Hilfe von DAMP zugeordnet, die Fenster W2 und W3 werden beim Öffnen eines BS2000-Diagnoseobjektes belegt und die Fenster W4 bis W9 und W21 bis W99 sind schließlich frei belegbar, z.B. mit den Anweisungen `SHOW-EDITED-INFORMATION` oder `START-PATTERN-SEARCH`.

Anzahl, Reihenfolge und Länge der auf dem Bildschirm erscheinenden Fenster können vom Benutzer mit der Anweisung `MODIFY-SCREEN-LAYOUT` gesteuert werden. Eine eingestellte Zuordnung bleibt für ein Fenster auch dann erhalten, wenn es momentan nicht sichtbar ist. Alternativ zu der Anweisung `MODIFY-SCREEN-LAYOUT` kann ein Fenster (in der aktuell gültigen Länge) auch durch Drücken einer P-Taste (Fenster W1 bis W9) bzw. durch Eingabe der Fensternummer (0 - 9, 21 - 99) in der Kommandozeile und Drücken von DUE sichtbar gemacht werden.

Weitere Informationen zum Diagnosefenster siehe auch "Diagnosefenster verändern".

5.2.2.1 Das Übersicht-Fenster (W0)

Das Übersicht-Fenster (W0) enthält stets die aktuelle Belegung der verfügbaren Diagnosefenster W0 bis W9 und W21 bis W99. Da W10 bis W20 nicht als Diagnosefenster genutzt werden dürfen (um Konfliktsituationen mit der bisherigen Verwendung der P-Tasten P10 bis P20 zu vermeiden), sind diese als „reserviert“ gekennzeichnet.

Ist ein Diagnosefenster aktuell nicht belegt, so wird dies im Fenster W0 durch die Ausgabe von Leerzeichen angezeigt. Ansonsten wird die aktuelle Fenster-Belegung angezeigt. Die Anzeige erfolgt analog zur Anzeige für die Fenster W1 bis W9 in der Key-Zeile (siehe Beschreibung der „Key-Zeile (Zeile 24)“ auf "Bildschirmmaske"). Da im Fenster W0 zur Belegungsanzeige mehr Zeichen zur Verfügung stehen als in der Key-Zeile, wird hier bei Speicherbereichen die Adresse (an Stelle der Seitennummer) angezeigt.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


DAMP Window Assignment                                                W0.WIN.L19
00 = WINDOWS    01 = HELP       02 = PLK        03 = PCB        04 = U70FFD328
05 = S728D73E8  06 =            07 = U70FFCFC8  08 = D00000000  09 =
10-20: reserved 21 = EXVT       22 = ETCB       23 = S728BA380  24 =
25 =            26 =            27 =            28 =            29 =
30 =            31 =            32 =            33 =            34 =
35 =            36 =            37 =            38 =            39 =
40 =            41 =            42 =            43 =            44 =
45 =            46 =            47 =            48 =            49 =
50 =            51 =            52 =            53 =            54 =
55 =            56 =            57 =            58 =            59 =
60 =            61 =            62 =            63 =            64 =
65 =            66 =            67 =            68 =            69 =
70 =            71 =            72 =            73 =            74 =
75 =            76 =            77 =            78 =            79 =
80 =            81 =            82 =            83 =            84 =
85 =            86 =            87 =            88 =            89 =
90 =            91 =            92 =            93 =            94 =
95 =            96 =            97 =            98 = TRACCMD    99 = FINDCMD
CMD:
Key: 1=Help 2=Plk 3=PCB 4=U70FFD 5=S728D7 6=Dump   7=U70FFC 8=D-0000 9=Dump
```

Bild 7: Ausgabe im Übersicht-Fenster (W0)

Durch Markieren einer Fensternummer mit MAR und anschließendes Drücken von DUE kann das ausgewählte Fenster auf dem Bildschirm sichtbar gemacht werden. Alternativ dazu kann die Fensterauswahl auch wie unter „Diagnosefenster“ (siehe "Diagnosefenster") beschrieben erfolgen.

5.2.2.2 Das Help-Fenster (W1)

Im Help-Fenster können Sie sich über Funktionen und Bedienung von DAMP informieren. Alle Informationen sind nach Kapitel und Unterkapitel gegliedert. Das jeweils gerade ausgegebene Kapitel wird in der Kopfzeile („Kapitel nnnn“) angezeigt.
Die Hilfe-Information steht in Deutsch und Englisch zur Verfügung.

Schlagworte in der Help-Information sind hellgesteuert und markierbar. Nach Markieren eines Schlagwortes mit MAR und anschließendem Drücken der Taste DUE erscheint das Kapitel mit der Erklärung dieses Schlagwortes.

Innerhalb des Help-Fensters stehen Ihnen die folgenden Blätterfunktionen zur Verfügung:

- an den Anfang/das Ende eines Kapitels mit der Eingabe „--“ oder „++“
- n Zeilen zurück/vor mit der Eingabe „-n“ oder „+n“
- eine Fensterlänge zurück/vor mit der Eingabe „-“ / F1 oder „+“ / F3

```
DAMP <version>  No Object opened in BS2000 V<ver>  <date> <time>


H E L P    Kapitel 0001 / DEUTSCH --------------------------------     W1,TXT,L19
                        D A M P   Version <version>
                Dump Analysis and Maintenance Program.

Dieses Programm dient zur Auswertung von Diagnoseobjekten (Dumpdateien und
aktives System) im Dialog. Die folgenden Dumpdateitypen werden unterstuetzt:
SLED (auch VM2000-Gesamtsled), SNAP, Systemdump, Userdump und Areadump.
Zur Diagnose des aktiven Systems ist die Lese-Testprivilegierung 8 erforderlich.

  Weitere Informationen: Inhalt / Stichwoerter markieren oder Kapitel angeben,
                         Blaettern mit --, ++, - (F1), + (F3), -n, +n
  Beginn der Auswertung: Anweisung oder '?' in CMD-Zeile eingeben,
                         Diagnosefenster mit P-Taste oder Fenster-Nr. auswaehlen

Durch Eingabe eines '?' in ein Eingabefeld eines Diagnosefensters erhalten Sie
direkte Hilfe zu diesem Feld (nur bei DSS 9763).

Note for English users:
By entering 'ENGLISH' in the header line you can change the language!

CMD:
Key: 1=Help 2=Plk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=Dump   9=Dump
```

Bild 8: Ausgabe im Help-Fenster (W1)

Das Help-Fenster hat die Eingabefelder „Kapitel“, „Sprache“ und „Fensterlänge“, die sich in der Kopfzeile befinden (siehe nächste Seite):

- Sie können jedes Kapitel durch Eingeben der Kapitelnummer in das Eingabefeld „Kapitel“ direkt ansprechen.
- Die Sprache der Ausgabe im Help-Fenster, der Online-Hilfetexte (die bei der Eingabe von „?“ in ein Eingabefeld angezeigt werden) und der DAMP-Meldungen können Sie durch Eingabe von DEUTSCH oder ENGLISH (Standard) im Eingabefeld „Sprache“ umschalten. Die Sprache können Sie auch über das Spezial-Fenster OPTIONS (siehe "Modifikationen durch den Benutzer (Spezialfenster OPTIONS)") ändern.
- Im Feld „Fensterlänge“ wird die Länge des Help-Fensters ein- bzw. ausgegeben.

Bild 9: Eingabefelder in der Kopfzeile des Help-Fensters

Neben der Nutzung des Help-Fensters bietet DAMP auch die – in vielen Fällen komfortablere – Möglichkeit, Hilfetexte durch Eingabe eines Fragezeichen ('?') in ein Eingabefeld eines der Diagnosefenster abzurufen (siehe "Online-Hilfetexte"). Die '?'-Hilfe steht allerdings nur auf den Datensichtstationen 9763 (mit EDT-Modus '@VDT F2') in Deutsch und Englisch zur Verfügung.

5.2.2.3 Das Status-Fenster (W2)

Das Status-Fenster gibt einen Überblick über das geöffnete Diagnoseobjekt. Die ersten vier Zeilen geben allgemeine Informationen über die Art des Diagnoseobjekts und die Umgebung während der Erstellung (z.B. BS2000-Version, Maschinentyp oder Speicherausbau).

Das Status-Fenster wird nach dem Öffnen des Diagnoseobjekts automatisch angezeigt und hat die Eingabefelder „TID“, „TSN“, „Modus-Auswahl“ und „Fensterlänge“.

Bild 10: Eingabefelder in der Kopfzeile des Status-Fensters

Modus-Auswahl

Im Eingabefeld „Modus-Auswahl“ für das Status-Fenster (W2) kann durch verschiedene Modi die Informationsausgabe beeinflusst werden.
Es werden die Modi INF, TSK, PLK und SLK angeboten.

INF Zusätzlich zu den standardmäßigen Ausgaben in den ersten drei Zeilen des Status-Fensters werden weitere Informationen zu Dumperzeuger, Diagnoseobjekt und ausgewähltem Produkt geliefert. Der Modus INF wird automatisch eingestellt, wenn die Dumpdatei mehr als ein Objekt (Produkt) enthält, z.B. bei einem VM2000-Gesamtsled.

DAMP kann die folgenden Objekte auswerten:

- aktives BS2000-System
- System-, User- und Areadumps mit Dumpobjekt BS2000
- SNAP-Dump mit Dumpobjekt BS2000
- SLED mit Dumpobjekt BS2000
- SLED mit Dumpobjekt VM2000 (VM2000-Gesamtsled)
- SLED mit Dumpobjekt SLED (SLED vom SLED / Dump vom SLED)
- SLED mit sonstigem Dumpobjekt wie z.B. SIR
- virtuelle Maschine (VM) unter VM2000
- Vorgängersystem im SLED
  (Vorgängersysteme sind BS2000, VM2000 oder andere Systeme)

Beispielfenster zu den genannten Objekttypen finden Sie ab "Das Status-Fenster (W2)".

Durch den Eintrag INF in der Kopfzeile kann der Modus durch den Anwender eingestellt werden. Wurde in Dumpdateien mit mehreren Objekten (VM2000-Gesamtsled, SLED vom SLED / Dump vom SLED) eine Objekt-Auswahl getroffen, hebt „-“ / F1 im Modus INF die Auswahl wieder auf.

Im Modus INF werden soviel Informationen wie möglich gezeigt. Eine Ausnahme ist der so genannte SELF-LOADER (siehe "Bearbeitung von SLEDs ohne BS2000-Struktur").
Die Informationen sind zunächst:

- der Name der Dump-Datei und das HSI des analysierten Objekts
- die Speichergrößen des analysierten Objekts
- der CPU-Typ, gefolgt von `Virtual Machine`, wenn es sich um eine virtuelle Maschine (VM2000) handelt (nach einer Live Migration wird hier usätzlich der neue Systemname, gefolgt von der Zeichenfolge `(after Live-Mig)` ausgegeben)
- der Typ und die Version des Dumperzeugers des zu analysierenden Objekts
- der Inhalt der Produkt-ID; dazu gehört der Name und die Version des Produktes und, sofern vorhanden, die Adresse des sog. Dump-Testaments (enthält interne SLED-Informationen).

Durch Markieren (siehe Abschnitt „Markieren" (Diagnosefenster verändern)) der Adresse des Dump-Testaments kann der Speicherinhalt des Dump-Testaments in einem Standardfenster mit Adressierungsmodus RM bzw. ABS ausgegeben werden.

Abhängig vom Dumpdateityp kommen dazu die folgenden Informationen:

- Für System-, User- und Areadump der vollständige Inhalt der Dummy-Section „Dumptitle“.
- Bei einem SLED
  - Der Inhalt der BS2000-Crash-Meldung mit abdruckbarem Text.
    Eine VM2000-Crash-Meldung wird von DAMP nicht erkannt.
  - Der Inhalt des Time of Day Registers wird aufbereitet ausgegeben.
  - Informationen über weitere Dumpobjekte z.B. PREVIOUS SYSTEM oder DUMPED SYSTEM. Beim Dumpobjekt VM2000 wird eine Gesamtübersicht der virtuellen Maschinen im Gesamtsystem angeboten. Das gewünschte Dumpobjekt wird durch Markieren ausgewählt.
  - Eine Meldung, wenn ein STARTUP-Dump vorliegt.
- Bei einem SNAP-Dump
  - Die Adresse der SNAP-Informationen.
  - Der Inhalt der SNAP-Meldung.
  - Die Informationen über den Aufruf von $SNAP, z.B. der Funktionsbereich, aus dem der Aufruf gestartet wurde, ggf. die TSN des Aufrufers, die Adresse des SNAP-Aufrufs und die Anfangsadresse des GP-Registersatzes.
- Wenn die zu analysierende Datei als PAM-Datei geöffnet wird, erhält man in diesem Modus automatisch die Informationen über die geöffnete Datei, z.B. den Dateinamen, die Dateigröße und den Last-Page-Pointer.

Beispielfenster zu den verschiedenen Dumpdateitypen im Modus INF

```
DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>


Analyzed Object: BS2000     <ver>               TID=00010001 TSN=     W2,INF,L 8
Dumpfile: :1DQM:$DIAGDUMP.QM122.05.LOI.SLED.ABGQN406-8 on X86-HSI/VM
MemSize:   3.8 GB  ShareB: 00D00 UserXB: 01000 SysB: BF000
CPU: X86SU-  300-160F / Virtual Machine X86SU-  300-160F (after Live-Mig)
Generator Name: SLED     (STD)          Generator Version: <ver>
Product Name:   BS2000                  Product Version:   <ver>

Dumptitle:   TSN-HSMS   ELSN-          SYSTEMDUMP   PC- 7C584BC2(DHSIHSM@+00502
             )   EC-50   VERS-<ver>   DUMP-TIME   <time>  <date>




--------------------------------------------------------------------------------
```

Bild 11: Informationsbildschirm im Status-Fenster (W2). Dumperzeuger CDUMP. Dumpobjekt BS2000 (SU x86 nach Live Migration)

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date>  <time>


Analyzed Object: BS2000   <ver>                 TID=00010001 TSN=     W2,INF,L13
Dumpfile: :1DQM:$DIAGDUMP.A0610398.SLED.ABGSE21B.170208 on /390-HSI/VM
MemSize: 512.0 MB  ShareB: 00C00 UserXB: 01000 SysB: 71000
CPU: 390SU-  700-70 / Virtual Machine
Generator Name: SLED     (ALL )         Generator Version: <ver>
Product Name:   BS2000                  Product Version:   <ver>

Time of creating SLED:  <date>  <time>

ID of Crash Message: NRTT501 SETS.      Crash ID: NRTC515
Crash Insert: SVC ERROR AT NIA F1A67C44

Crash Caller: F1A67C44 = ETMOSVC  + 00204

--------------------------------------------------------------------------------
```

Bild 12: Informationsbildschirm im Status-Fenster (W2). Dumperzeuger SLED. Dumpobjekt BS2000 (SU /390 nach abnormaler Systembeendigung)

```
DAMP <version> SNAP(<ver>) from BS2000(<ver>)       <date> <time>


Analyzed Object: BS2000   V<ver>            TID=00010001 TSN=     W2,INF,L13
Dumpfile: :2OS6:$SPMO.MEN.SNAP.WILLI on /390-HSI/VM
MemSize: 256.0 MB  ShareB: 00C00 UserXB: 01000 SysB: 71000
CPU: 7.500- S210-40 / Virtual Machine
Generator Name: SNAP                  Generator Version: <ver>
Product Name:   BS2000                 Product Version:   V<ver>

SNAPID: NSPTEST
SNAP Insert:

Return Code of SNAP call: SNAP successfully processed
SNAP from SIH                          Address of SNAP call: 00000040
Address of SNAP internal data: 729A2800
--------------------------------------------------------------------------------
```

Bild 13: Informationsbildschirm im Status-Fenster (W2). Dumperzeuger SNAP. Dumpobjekt BS2000 (S-Server)

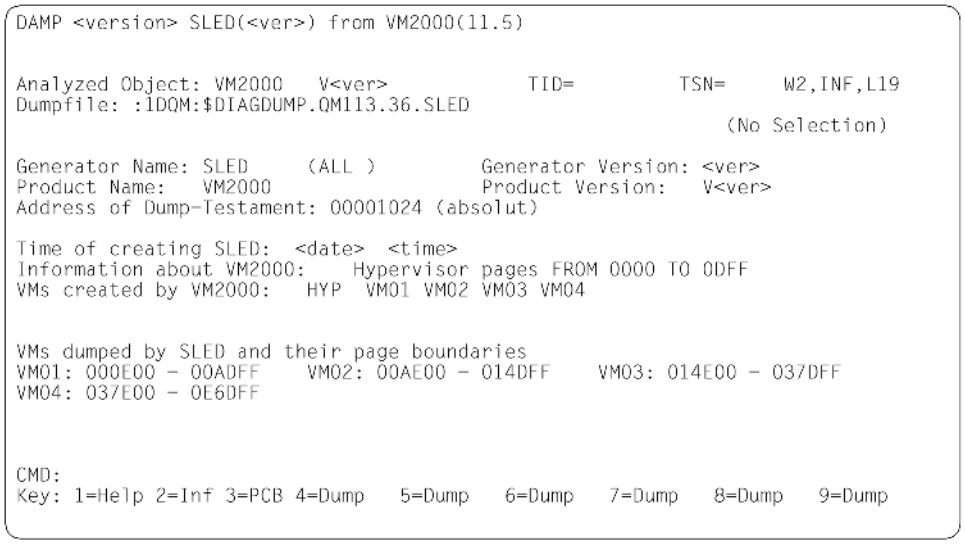


Bild 14: Informationsbildschirm im Status-Fenster (W2). Dumperzeuger SLED. Dumpobjekt VM2000

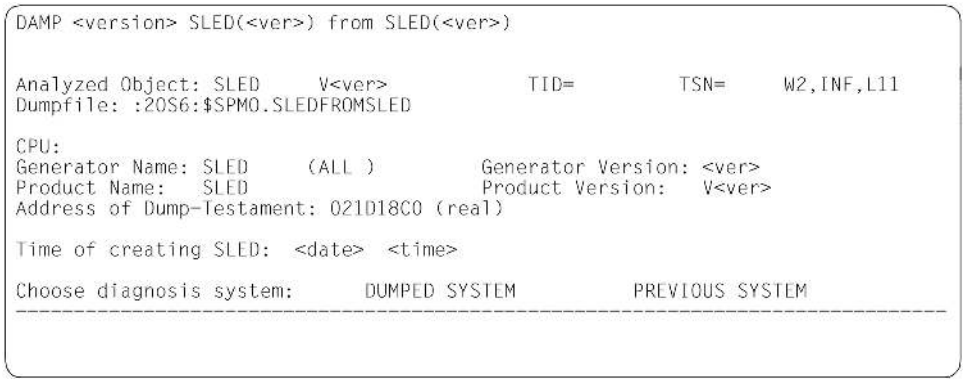


Bild 15: Informationsbildschirm im Status-Fenster (W2). Dumperzeuger SLED. Dumpobjekt SLED

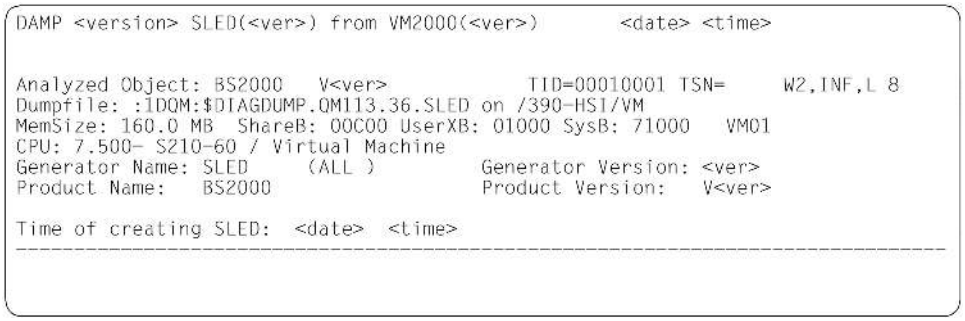


Bild 16: Informationsbildschirm im Status-Fenster (W2). Dumperzeuger SLED. Dumpobjekt BS2000 in der Monitor-VM

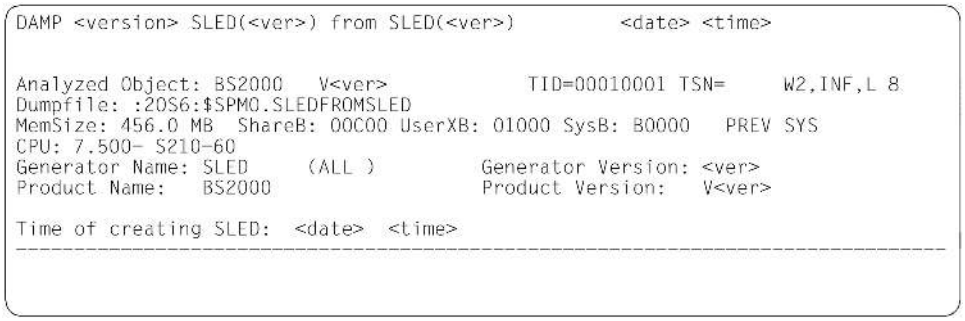


Bild 17: Informationsbildschirm im Status-Fenster (W2). Vorgängersystem. Dumpobjekt BS200

TSK Der Modus wird automatisch eingestellt, wenn ein SLED oder SNAP mit dem Dumpobjekt BS2000 oder das aktive System analysiert werden soll und Informationen über mehrere Tasks vorhanden sind. Es werden zunächst nur die ersten 14 Tasks im Status-Fenster (W2) ausgegeben. Jede Zeile enthält die Information für eine Task und kann markiert werden.
In der Key-Zeile ist der Modus durch den Eintrag TSK gekennzeichnet.

Durch die Eingabe von +, -, ++, --, +n, -n in die Kommandozeile und Drücken von DUE kann in der Taskliste geblättert werden. Anstatt „+“ DUE bzw. „-“ DUE kann die Taste F3 bzw. F1 betätigt werden.

> ¡ Bei der Diagnose des aktiven Systems wird die Taskliste nur beim Positionieren mit „--“ auf den Anfang (Anzeige der TID 0001) aktualisiert.

Durch Taskbeendigung bzw. -erzeugung können bei der Diagnose des aktiven Systems auch in anderen Fenstern Inkonsistenzen auftreten, die bei Bedarf durch eine Erneuerung der Taskliste aktualisiert werden müssen.

Beim Systemdump kann die Ausgabe der Taskliste durch Eingabe von „TSK“ im Feld Modus-Auswahl eingestellt werden (voreingestellt ist der Modus PLK für die 'Fehlertask'). Die Taskliste umfasst auch beim Systemdump alle Tasks aus dem zu diagnostizierenden BS2000-System. Durch Markieren einer Zeile kann eine Task zur weiteren Diagnose ausgewählt werden.

> ! **ACHTUNG!**
> Der Systemdump enthält systemglobale Daten aller Tasks, allerdings nur die tasklokalen Daten der 'Fehlertask'. Bei der Auswertung der Systemdaten (z.B. der PCB-Verkettungen) der übrigen Tasks ist äußerste Vorsicht geboten, da diese Tasks während der Erstellung des Systemdumps nicht angehalten werden.

Die Taskliste kann nach verschiedenen Kriterien sortiert werden. Dazu müssen Sie in der Überschrift-Zeile eine Spalte markieren. Die Sortierung erfolgt aufsteigend, gemäß dem Inhalt der ausgewählten Spalte. Voreingestellt ist die Sortierung nach TID (1. Spalte).

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>
DMP2307 TID 00010001 HAS BEEN SELECTED AS CURRENT TID

Analyzed Object: BS2000   V<ver>            TID=         TSN=     W2,TSK,L19
Dumpfile: :SLED:$DUMPFILE.SLED.CS503K on /390-HSI/VM
MemSize: 512.0 MB  ShareB: 00C00 UserXB: 01000 SysB: 71000

     TID  TSN Typ Q-PND Act SVC/IW   Current PCR     Caller SVC/PCR
00010001  TSC PA  12 17 FA=$BOWT     JSM@@@@@+00316
00010002 CLOG SYS 12 17 FA=$BOWT     NBRCLOG +003D0
00010003  RMM PA  12 17 FA=$BOWT     EMMRMRCM+02018
00010004 HERS SYS 11  6 EB=$PEND     EHERSPT +001BC
00010005  PT5 PA  13  4 59=VPASS     ETMPT5  +0027C
00010006  PT6 PA  12 17 FA=$BOWT     JSSTASK@+00952
00010007  PGE PA   4 17 FA=$BOWT     DJPGER  +010FC
00010008  UCO PA   4 17 FA=$BOWT     NBRMAIN +00048
00010009  REK PA  12 17 FA=$BOWT     ETMRK2B +00190
0001000A  VMM PA   0  0 F1=$PNDT     NRTSEH  +01662  48=Pag. Err  MESCPUSS+2593C
0001000B  MSG PA  12 17 FA=$BOWT     ETMRK2F +000EA
0001000C  KTT PA   4  4 59=VPASS     NBCADM  +01C5C
0001000D RUNT SYS 12 17 FA=$BOWT     ECCLP   +007D2
0001000E SEST SYS 12 17 FA=$BOWT     NBESSWR@+036FE
CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=Dump   9=Dump
```

Bild 18: Taskübersicht im Status-Fenster (W2)

PLK Der Modus wird automatisch eingestellt, wenn ein Area-, User- oder Systemdump bearbeitet wird, der nur eine Task enthält. Er kann auch im Modus TSK durch Markieren einer Taskzeile oder durch Eingabe einer TID/TSN eingestellt werden. Bei der Auswahl einer Task genügt der rechte Teil der 4-Byte-TID.
Es werden dann alle zu dieser Task gehörenden PCBs mit der dazugehörigen Information ausgegeben. Durch das Markieren einer PCB-Zeile erfolgt die aufbereitete Ausgabe dieses PCBs im Stack-Fenster (W3).
In der Key-Zeile wird das Symbol PLK angezeigt.

Benutzer-PCBs sind in der Ausgabezeile mit einem „*“ markiert.

In der PCB-Übersicht für x86-Objekte wird in der Spalte A_MODE zusätzlich zum Adressierungsmodus noch der Kontext-Typ angegeben. Dabei steht CIS für einen PCB mit /390-Kontext und X86 für einen PCB mit x86-Kontext. Folgende Ausgaben sind bei diesen Objekten möglich:

- Bei x86-Objekten:
  CIS 31 oder CIS 24, falls der PCB einen /390-Kontext besitzt (im /390-Modus (d.h. emuliert) ablaufender Code mit 31- bzw. 24-Bit-Adressierung)
- Bei x86-Objekten:
  X86 32, X86 31 oder X86 24, falls der PCB einen x86-Kontext besitzt (im x86-Modus (d.h. native) ablaufender Code mit 32-, 31- bzw. 24-Bit-Adressierung)

Aus dem ersten PCB erfolgt die Rückkehr zur Taskübersicht mit F1/ „-“ oder durch Eingabe von TSK im Feld Modus-Auswahl.

```
DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>


Analyzed Object: BS2000   V<ver>            TID=00010065 TSN=2WG7 W2,PLK,L 9
Dumpfile: :LOU3:$SYSDUMP.EVENT#SA.000D4.C8.2WG7 on X86-HSI
MemSize:   5.5 GB  ShareB: 00C00 UserXB: 01000 SysB: BF000
Job: MENCHER /TSOS    /ADMINSTR Cmd: CREATE-D Prg: SYSPRG.DAMP.<ver>
PCB#  PCB-Ad    IS-LNK    SR        Program Counter          SVC/IW      A_MODE
   1  C33271E0  C3327768  070C0C04  C0BE9094=CDUMPF1 +02B14  EE=$PNUP    X86 32
   2  C3327768  C3327590  070C0C04  C1D687E6=NSCDUMP +00206  1A=CDUMP    X86 32
*  3  C3327590  00000000  07ED2C00  010072D4                 5C=BKPT     CIS 31
   4  C33273B8  00000000  070C0C04  C05A456E=ETMSF   +008EE  E9=$FNAT    X86 32
--------------------------------------------------------------------------------
```

Bild 19: PCB-Übersicht im Status-Fenster (W2), Dump-Datei mit x86-Objekt

SLK In diesem Modus wird die Kette der Aufrufe über den TPR-Program-Manager (SPL-Linkage) ausgegeben. Das Modus-Feld in der Kopfzeile des Fensters muss dazu mit dem Symbol SLK überschrieben werden. Durch das Markieren einer der angezeigten Stack-Zeilen erfolgt die aufbereitete Ausgabe dieses Program-Manager-Stacks im Stack-Fenster (W3). In der Key-Zeile wird das Symbol SLK angezeigt.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


Analyzed Object: BS2000   V<ver>            TID=000402FB TSN=0VGY W2,SLK,L19
Dumpfile: :SLED:$DUMPFILE.SLED.CS507K.1015 on /390-HSI
MemSize:   4.0 GB  ShareB: 00C00 UserXB: 01000 SysB: 71000
Job: NKI10882/DMS10   /A        Cmd: CALL-PRO Prg:
Stk#  Stack-Ad  ADF       Params (R1) Caller (R14)           ADF-Ind
   1  6F3D22D0  6F3D2328  6F3D2398    F14477FA=ECTYP   +037FA N     current
   2  6F3D2428  6F3D2480  6F3D23D0    F144587A=ECTYP   +0187A N
   3  6F3D2580  6F3D25D8  6F3D2FE4    F1444D82=ECTYP   +00D82 N
   4  6F3D2740  6F3D2798  6F3D2FE4    FF71922C=NMHRPUT +00BAC N
   5  6F3D3160  6F3D31B8  7F719F8A    FF71CD22=NMHRSPL +00A22 N
   6  6F3D3AB0  6F3D3B08  6F3D3BE0    F154BEFC=CDUMPF3 +02D7C N
   7  6F3D3C58  6F3D3CB0  00000002    F15453A4=CDUMPF1A+00B24 N
   8  6F3D3D80  6F3D3DD8  70FA2000    F1541E3E=CDUMPF1 +002FE N
   9  6F3FC2D8  6F3FC330  00000000    00000000                N
  10  6F3FD590  6F3FD5E8  6F3FEFE8    FD586B74=CLIKREA@+03E34 N
  11  6F3FF238  6F3FF290  6F3FFF68    FD56BBEA=CLIISL@@+007FA N
  12  70FDA870  70FDA8C8  70FDB1DC    F1390018=NLKISLS +00558 O
  13  70FDA938  70FDA990  70FDB1DC    FF23F56A=SSMLIBR@+0056A N
  14  70FDBE60  70FDBEB8  70FDD1B8    FF24BF8E=JSYLIBR +0004E N
CMD:
Key: 1=Help 2=Slk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=Dump   9=Dump
```

Bild 20: Kette der Program Manager Aufrufe im Status-Fenster (W2)

Aus dem ersten Stack erfolgt die Rückkehr zur Taskübersicht mit F1/ „-“ oder durch Eingabe von TSK im Feld `Modus-Auswahl`.

Blättern im Statusfenster

Die Eingaben +, -, ++, --, +n, -n sowie die Funktionstasten F3 (blättert um eine Fensterlänge vorwärts, dies entspricht +) und F1 (blättert um eine Fensterlänge rückwärts, dies entspricht -) stehen zur Verfügung. Näheres hierzu siehe „Blättern in einem Diagnosefenster" (Diagnosefenster verändern).

Hinweis zum Vorwärtsblättern

Beim Vorwärtsblättern (+, ++, +n, F3) bleibt man am Ende der jeweiligen Liste stehen, es erfolgt kein automatisches Wiederaufsetzen am Listenanfang.

Hinweise zum Rückwärtsblättern

- F1 und die Eingabe „-“ blättern in den Modi PLK und SLK zur Taskliste zurück, wenn aktuell der erste PCB bzw. der erste Stack auf dem Fenster sichtbar ist. Durch die Modus-Auswahl TSK kehrt man ebenfalls zur Taskübersicht zurück. Dabei wird die aktuelle Task die erste Task in der Übersicht.
- Bei der Diagnose des aktiven Systems wird im Modus TSK durch das Blättern mit „--“ auf die erste Task die Taskliste neu aufgebaut.
- Wurde in Dumpdateien mit mehreren Objekten eine Auswahl des Dumpobjekts getroffen, hebt F1 bzw. die Eingabe „-“ im Modus INF die Auswahl wieder auf.

5.2.2.4 Das Stack-Fenster (W3)

Das Stack-Fenster zeigt den Inhalt des ersten bzw. des im Fenster W2 ausgewählten und markierten TU- bzw. TPR-Stack (Modus PCB) oder eines TPR-Program-Manager-Stack (Modus SPL). Die weiteren Informationen sind abhängig vom im Status-Fenster (W2) gewählten Modus.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


SYS-PCB (31BIT)    #   1       Addr: 73B001D8   TID=000402FB TSN=0VGY W3,PCB,L11
PC  : 7103B916=ETMBON1 +001D6    SVC: FA = $BOWT    LNK: 74135AE8  ISL: 7460AAE8
R00 : 00000000 =      0                R01 : F103B803=ETMBON1 +000C3
R02 : 74692008=ETCB-2FB                R03 : 71000800=NCTXVT  +00800
R04 : 7103C548=ETMBON1 +00E08          R05 : 73B001D8
R06 : 71247B40=ETMBOWK                 R07 : 71248A00=ETMBOWK +00EC0
R08 : 6F3D2328                         R09 : 6F3D2398
R10 : 7103B740=ETMBON1                 R11 : 7F7194C8=NMHRPUT +00E48
R12 : 80800102                         R13 : 71248A38=ETMBOWK +00EF8
R14 : F103B908=ETMBON1 +001C8          R15 : 71021974=NLCNLMAN+03234
Bourse Caller: 714477FA=ECTYP   +037FA
-------------------------------------------------------------------------------
```

Bild 21: Stack-Fenster (W3) mit PCB

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


SPL-Stack          #   1       Addr: 6F3D22D0   TID=000402FB TSN=0VGY W3,SPL,L10
ADF : 6F3D2328   (User)          ***  current   *** LNK: 6F3D2428  ADFI:  P  NYN
R00 : 74692008=ETCB-2FB                R01 : 6F3D2398
R02 : C2404040                         R03 : FF718986=NMHRPUT +00306
R04 : 714458E8=ECTYP   +018E8          R05 : 00000001 =      1
R06 : 00000002 =      2                R07 : 7348D600
R08 : 7A8E3A90                         R09 : 6F3D25D8
R10 : 71447748=ECTYP   +03748          R11 : 7F7194C8=NMHRPUT +00E48
R12 : 7100DA78=NLKSYSPM+014B8          R13 : 6F3D2328
R14 : F14477FA=ECTYP   +037FA          R15 : 7103B740=ETMBON1
-------------------------------------------------------------------------------
```

Bild 22: Stack-Fenster (W3) mit Program-Manager-Stack

Das Stack-Fenster (W3) hat die Eingabefelder „Stack#“, „TID“, „TSN“, „Stack-Auswahl“ und „Fensterlänge“.

Bild 23: Eingabefelder in der Kopfzeile des Stack-Fensters

Um von einem in das andere Ausgabeformat zu wechseln, überschreiben Sie die Angabe „PCB“ im Eingabefeld „Stack-Auswahl“ in der Kopfzeile des Stack-Fensters mit „SPL“ bzw. umgekehrt und drücken anschließend die Taste DUE.

Registerinhalte werden nach Möglichkeit als Adressen bzw. Dezimalwerte interpretiert und ausgegeben. Jedes Feld der Ausgabe ist markierbar.

Um einen Speicherbereich in einem der Standard-Dumpfenster (W4-W9,W21-W99) darzustellen, markieren Sie das Adressfeld und weisen Sie ihm ein Ausgabefenster zu, siehe Abschnitt „Markieren" (Diagnosefenster verändern)

Ist in dem angezeigten PCB das Access-Register-Mode-Flag (AR-Mode-Flag) gesetzt und enthält das gleichnamige Access-Register einen Wert (ALET) ungleich null, so führt das Markieren eines Mehrzweckregisters gleich zur Zuweisung des entsprechenden Datenraumes und dessen Anzeige im gewünschten Fenster.
Die Access-Register liegen im Stack-Fenster (W3) „hinter“ den Mehrzweckregistern, falls das AR-Mode-Flag im PCB gesetzt ist.
Ist das AR-Mode-Flag im PCB nicht gesetzt, dann kann man die Access Register im PCB unter dem symbolischen Namen ESTKARx (x=0,1,...15) finden.

Hinweise

- Ein PCB mit x86-Kontext wird mit allen 16 Registern des x86-Modus dargestellt. Ein Beispiel dafür zeigt Bild 24. Da in den Registern r12 - r15 ein /390-Kontext enthalten sein kann, werden diese Register in /390-Register-Notation angezeigt (R12 - R15). Alle anderen Register tragen die x86-spezifischen Bezeichnungen.

  Die Inhalte der /390-Register R0 - R11 werden von der Firmware nicht in x86-Registern abgespeichert, sondern im ASSTRAN-Speicher. Diese Bereiche werden im x86-Modus ebenfalls angezeigt.

  > **i** Aufgrund von ASSTRAN-Optimierungen werden die Speicherbereiche, aus denen DAMP sich die Inhalte der /390-Register besorgt, nicht bei jeder Änderung sofort aktualisiert. Im x86-Modus können daher die /390-Register nicht zuverlässig zur Diagnose genutzt werden. Allein die x86-Register erlauben in diesem Modus eine zuverlässige Diagnose!

  In einer speziellen Kopfzeile wird der Typ des dargestellten Register-Kontexts angezeigt. Durch Markieren können Sie auch den gewünschten Kontext auswählen:

CISC	/390-Register in Wortlänge
x86	x86-Register in Doppelwortlänge

  Sie können auch mit „>“ oder „<“ zum folgenden oder vorhergehenden Kontext blättern.

  Die spezielle Kopfzeile enthält zusätzlich die markierbare Adresse der CSA (Context Save Area). Sie ist in Bild 24 am linken Rand mit einem Pfeil markiert.

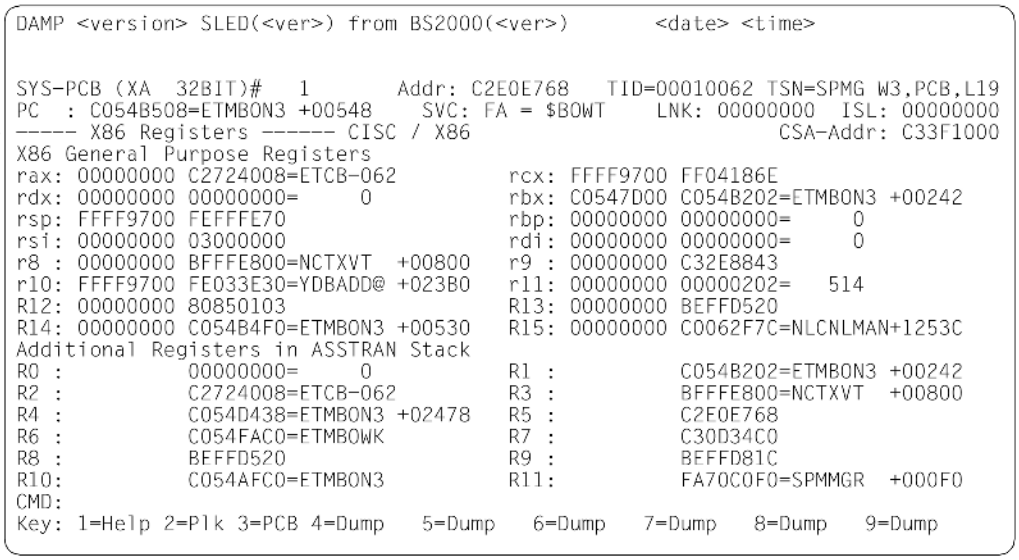


Bild 24: Beispiel für das Stack-Fenster eines PCB mit x86-Kontext

5.2.2.5 Die Dumpfenster (W4 - W9 und W21 - W99)

Diesen Fenstern kann der Benutzer verschiedene Funktionen zuordnen, voreingestellt ist die Nutzung als Standard-Dumpfenster für die aufbereitete Ausgabe von Speicherbereichen. Dies entspricht der Anweisung `SHOW-EDITED-INFORMATION INFORMATION=*STORAGE-EDIT, WINDOW=<w>.`

Die Fenster W4 - W9 und W21 - W99 können für Spezialausgaben als sog. Spezialfenster verwendet werden. Die Ausgabe erfolgt mit der Anweisung `SHOW-EDITED-INFORMATION` und der Angabe des vorgesehenen Dumpfensters sowie der gewünschten Aufbereitung. Andere Anweisungen, wie z.B. `START-PATTERN-SEARCH`, öffnen ebenfalls ein Spezialfenster (Beschreibung ab "System-Trace-Table ausgeben (Spezialfenster TRACE)"). Wird kein bestimmtes Fenster angegeben, legt DAMP die Ausgabe auf das nächste freie Fenster.

Mit der Anweisung `SHOW-EDITED-INFORMATION INFORMATION=*STORAGE-EDIT` wird das belegte Fenster wieder für eine Standardausgabe verfügbar gemacht.

In den Fenstern W4 - W9 und W21 - W99 werden die folgenden Anweisungen unterstützt:

```
SHOW-EDITED-INFORMATION INFORMATION=*AUDIT-TABLE-EDIT
SHOW-EDITED-INFORMATION INFORMATION=*STORAGE-EDIT
SHOW-EDITED-INFORMATION INFORMATION=*DUMPED-SYSTEM-FILE
START-PATTERN-SEARCH
START-LIST-GENERATION
SHOW-EDITED-INFORMATION INFORMATION=*MEMORY-ATTRIBUTES
START-OPTION-DIALOG
START-PRODAMP-EDITOR
SHOW-EDITED-INFORMATION INFORMATION=*SUBSYSTEM-INFORMATION
SHOW-EDITED-INFORMATION INFORMATION=*TASK-TABLES
SHOW-EDITED-INFORMATION INFORMATION=*TRACE-TABLE-EDIT
```

Die Zuweisung einer Funktion an ein Diagnosefenster ist auch über folgende Kurzform möglich:

```
ATT[ACH] window#, function
```

Für `window#` ist die gewünschte Fenster-Nummer (4...9, 21...99) anzugeben.

Für `function` werden die folgenden Eingaben unterstützt:

`AUDI, AUDIT`	Informationen über AUDIT-Tabellen ausgeben
`DUMP`	Standard-Dumpfenster wiederherstellen
`FILE`	Informationen über Systemdateien/Sections
`FIND`	Zeichenketten suchen
`LIST`	Listen erzeugen und ausdrucken
`MEMA, MEMATTR`	Speicherattribute ausgeben
`OPTS, OPTIONS`	Benutzeroptionen modifizieren
`PROC, PRODAMP`	PRODAMP-Prozeduren verwenden

SUSY	Informationen über Subsysteme ausgeben

TABL, TABLE	Tabellen taskspezifischer Werte ausgeben
TRAC, TRACE	System-Trace-Table ausgeben

Beispiele

`ATT 4,DUMP` oder `ATT 99,FIND`

Die Nutzung der Fenster W4 - W9 und W21 - W99 als Standard-Dumpfenster

Die Standard-Dumpfenster können Speicherausschnitte des Diagnoseobjekts im Dump-Format, Sedezimal-Format, Zeichen-Format, Assembler-Format oder symbolischen Format ausgeben.
Außer im Zeichen- und Assembler-Format können Sie Speicherausschnitte in einem der Standard-Dumpfenster darstellen, indem Sie Adressfelder markieren und ihnen Ausgabefenster zuweisen, siehe Abschnitt „Markieren" (Diagnosefenster verändern).

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


ASAFNAT                          +00000=7F2774C0 SYS=000402FB          W4,D  ,L19
7F2774C0 (0000): 05A058F0  A36E0DEF  5510A372  4770A018   <==>  ???0t>????t?????
7F2774D0 (0010): 58F0A376  0DEF47F0  A0345510  A37A4770   <==>  ?0t????0????t:??
7F2774E0 (0020): A0340DE0  47F0E008  7103F264  41A0E012   <==>  ?????0????2?????
7F2774F0 (0030): 58E0E004  07FE0700  4110A03E  47F0A04E   <==>  ?????????????0?+
7F277500 (0040): 00061202  FFFFFFFF  02C9C4C1  F0F1F5F7   <==>  ????~~~~?IDA0157
7F277510 (0050): 0AED0000  00000000  05A05860  23449640   <==>  ???????????-??o
```

Bild 25: Ausgabe eines Bereichs im Dump-Format

Bei Objekten von x86-Servern können Sie Adressen, die im x86-HSI im Format Little Endian angezeigt werden, durch Markieren und Drücken von F4 vor Ausgabe des zugehörigen Speicherbereichs von DAMP in das BS2000-Adressformat (Big Endian) konvertieren lassen (siehe "Diagnosefenster verändern").

5.2.2.6 Eingabefelder der Standard-Dumpfenster (W4 - W9 und W21 - W99)

Die Standard-Dumpfenster haben in der Kopfzeile die Eingabefelder „Symbolische Adresse“, „Relativadresse“, „Absolutadresse“, „ASEL“, „ASID“, „Ausgabeformat“ und „Fensterlänge“.

In den Spezialfenstern, die mit der Anweisung `SHOW-EDITED-INFORMATION` aktiviert werden, sowie in den Fenstern FIND, LIST, OPTS oder im PRODAMP-Fenster sind weitere oder andersartige Eingaben möglich (siehe ab "System-Trace-Table ausgeben (Spezialfenster TRACE)").

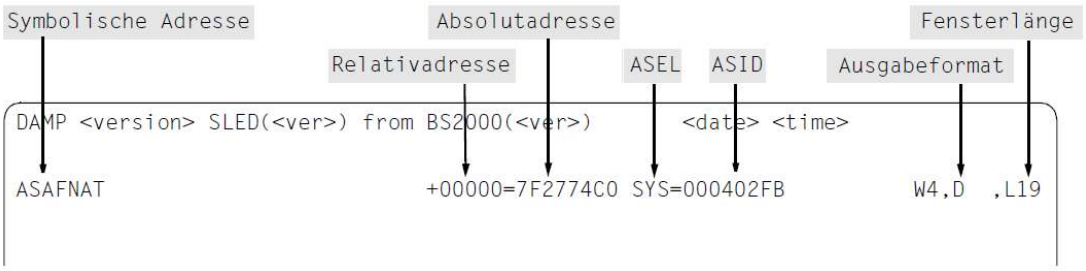


Bild 26: Eingabefelder in der Kopfzeile der Standard-Dumpfenster

Symbolische Adresse

Im Eingabefeld „Symbolische Adresse“ (siehe Bild 26) können Modulnamen, Kontrollblock-Namen und auch Kontrollblock-Feldnamen eingegeben werden.
Angezeigt wird bei der Ausgabe standardmäßig der Modul-Name. Kann der Speicherbereich keinem Modul zugeordnet werden, werden Leerzeichen angezeigt.
Wird ein Kontrollblock-(Feld)-Name angezeigt (Ausgabeformat CBA/CBM), dann kann der Modulname sichtbar gemacht werden durch Eingabe von „NAM“ in das Feld „Ausgabeformat“.
Es können auch Namen von CSECTs aus Subsystemen oder aus einem evtl. geladenen Benutzerprogramm angegeben werden. Sofern die aktuelle Task an das nichtprivilegierte Subsystem angeschlossen bzw. die CSECT in dem Benutzerprogramm enthalten ist, wird der Speicherbereich automatisch lokalisiert. Bei geladenen privilegierten Subsystemen lokalisiert DAMP auch dann, wenn die Task nicht angeschlossen ist.
Wird die modulrelative Darstellung nicht gewünscht, kann sie durch Eingabe von „ALT“ in das Feld „ASEL“ abgeschaltet werden. DAMP relativiert dann alle Adressen zur aktuellen Anfangsadresse in diesem Fenster. In diesem Format wird die Ausgabe zudem über die Modulgrenze hinweg fortgesetzt.

Zur Kennzeichnung, dass die modulrelative Darstellung abgeschaltet wurde, wird der Modulname gelöscht.
Diese Darstellung bleibt beim Blättern und bei Eingabe einer Relativadresse oder Absolutadresse erhalten.

Relativadresse

Angezeigt wird die momentane Distanz zwischen Modulanfang und „Absolutadresse“. Bei Veränderung der Relativadresse innerhalb eines CSECT/Kontrollblock-Bereiches ändern sich nur Relativadresse und Anfangsadresse des Fensterinhaltes („Absolutadresse“). Überschreitet diese Distanz jedoch den Modulbereich, werden die relativen und absoluten Adressausgaben automatisch angepasst. Das Gleiche passiert beim Blättern im Dumpfenster.

Absolutadresse

Angezeigt wird die Anfangsadresse des momentanen Fensterinhaltes.

Ausgabeformat

Es stehen die folgenden Ausgabeformate zur Verfügung:

D	Dump-Format (Defaultwert bzw. nach Eingabe von „D“). Pro Bildschirmzeile werden 16 Bytes sedezimal und abdruckbar ausgegeben. Die 4 Wortfelder im sedezimalen Format sind markierbar.
ASC	Dump-Format(ASCII); Pro Bildschirmzeile werden 16 Bytes sedezimal und abdruckbar in ASCII-Form ausgegeben. Die 4 Wortfelder im sedezimalen Format sind markierbar.
HEX	Hexadezimal-Format (nach Eingabe von „H“). Pro Bildschirmzeile werden 32 Bytes sedezimal ausgegeben. Alle 8 Wortfelder sind markierbar.
CHR	Zeichen-Format (nach Eingabe von „C“). Pro Bildschirmzeile werden 64 Bytes als abdruckbare Zeichen ausgegeben, nichtabdruckbare Zeichen erscheinen als Wischzeichen. Durch Einstellung des Benutzeroption „Trash character“ (siehe "Modifikationen durch den Benutzer (Spezialfenster OPTIONS)") kann auch ein anderes abdruckbares Ersatzzeichen gewählt werden.
ASS	Befehlsdarstellung (nach Eingabe von „A“). Pro Bildschirmzeile wird ein Befehl „disassembliert“ und im Maschinencode ausgegeben. Bei x86-Objekten wählt DAMP den Modus der Disassemblierung entsprechend des Prozessormodus der CSECT (PMODE-Byte). Sollte dies zu unsinnigen Ausgaben führen, kann der Modus CAS oder XAS explizit eingestellt werden. Bei der Ausgabe zeigt ASS an, dass die /390-Disassemblierung verwendet wurde.
CAS	Eingabe: die /390-Disassemblierung soll verwendet werden (als Ausgabe erscheint dann ASS).
XAS	Eingabe: die x86-Disassemblierung soll verwendet werden. Ausgabe: die x86-Disassemblierung wurde verwendet.
CBA	Symbolische Ausgabe mit automatischem Lokalisieren des Kontrollblocks (Control-Block-Automatic).
CBM	Symbolische Ausgabe mit manuellem Lokalisieren des Kontrollblocks (Control-Block-Manual).
NAM	Symbolische Ausgabe, wobei im Feld Symbol an Stelle des Kontrollblock-Namens der Name des Moduls angezeigt wird, in dem der Kontrollblock liegt.

Fensterlänge

Angezeigt wird die aktuelle Fensterlänge einschließlich der Kopfzeile. Die durch Benutzeroption veranlasste Trennzeile wird bei der Ausgabe im Feld „Fensterlänge“ nicht mitgezählt. Eingabewerte > 19 werden auf die Maximallänge 19 gekürzt.

Mit den Programmtasten P10 bis P12 (Terminal 9750) lässt sich der Cursor auf die Eingabefelder „Fensterlänge “ der folgenden Diagnosefenster positionieren:

P10	positioniert auf „Fensterlänge“ des ersten Diagnosefensters auf dem Bildschirm,
P11	positioniert auf „Fensterlänge“ des zweiten Diagnosefensters auf dem Bildschirm,

P12 positioniert auf „Fensterlänge“ des dritten Diagnosefensters auf dem Bildschirm

sofern mindestens drei Fenster am Bildschirm gezeigt werden.

Werden nur zwei Fenster angezeigt, positioniert P12, bei nur einem angezeigten Fenster positioniert auch P11 auf die Kommandozeile.

ASEL und ASID

Die in den Standard-Dumpfenstern gezeigten Speicherbereiche können Ausschnitte aus folgenden Bereichen sein:

- eines virtuellen Adressraums
- des Hauptspeichers (reale/absolute Adressierung)
- der Hardware-System-Area (HSA)
- eines gesicherten Prozessorstatus (PSS)
- aus Datenräumen
- aus Dumpfile-Sections (SCT)

Die Datenräume existieren neben den task- und systemspezifischen virtuellen Adressräumen und stellen quasi eine Vervielfältigung der virtuellen Adressräume dar.

Bei DAMP kann auch ein task- oder systemspezifischer Adressraum als Datenraum eingestellt werden. In diesem Fall verzichtet DAMP auf eine modulspezifische Relativierung.

Adressierung der Datenräume

Zusätzlich zu den 16 Mehrzweckregistern hat jeder Prozess noch weitere 16 Access-Register. Abhängig von der Stellung eines Schalters (AR-Mode) werden diese Access-Register (bis auf Register 0) zusätzlich zur Adressierung von Speicherbereichen herangezogen.

Das zum Basisregister gleichnamige Access-Register wird, wenn sein Inhalt ungleich null ist, zur Adressierung eines Datenraums verwendet. Innerhalb dieses Datenraums, der bis zu 2 GByte groß sein kann, wird wie üblich wieder mit Basisregister, Indexregister und Distanz adressiert.

Für die Adressierung erhalten die Access-Register einen sog. ALET (Access-List-Entry-Token), der für einen Adressraum (Task- oder Systemadressraum) den Datenraum eindeutig bestimmt. Systemweit wird der Datenraum durch die sog. SPID (Space-Identification) eindeutig bestimmt.

Die Namen ASEL (Address-Space-Selector) und ASID (Address-Space-Identifier) bezeichnen Felder, die folgende Symbole aufnehmen können:

ASEL	ASID	Bedeutung der Symbole
TID	<tid> (sedezimal)	Der Adressraum ist ein Benutzeradressraum und wird durch die <tid> spezifiziert.
TSN	<tsn> (string)	Der Adressraum ist ein Benutzeradressraum und wird durch die <tsn> spezifiziert.
SYS	wird ignoriert	Der Adressraum ist der Systemadressraum.

ALT	<alet>-<tid> (sedezimal)	Der Adressraum ist ein Datenraum, der durch <alet> (plus <tid> bei Benutzer-Datenräumen) spezifiziert wird.
SPI	<space-id> (sedezimal)	Der Adressraum ist ein Datenraum, der durch die (systemweite) <space-id> spezifiziert wird.
RM	<segm> (sedezimal)	Der Adressraum ist der reale Hauptspeicher im ausgewählten Objekt. <segm> bezeichnet das 4GB-Segment (0, 1, ...), in dem die Adresse liegt.
ABS	<segm> (sedezimal)	Der Adressraum ist der absolute Hauptspeicher im ausgewählten Objekt. <segm> bezeichnet das 4GB-Segment (0, 1, ...), in dem die Adresse liegt. (Nur bei VM2000-Gesamtsled)
PSS	<prozessor> (sedezimal)	Der Adressraum ist der Processor-Saved-Status des angegebenen Prozessors.
HSA	wird ignoriert	Der Adressraum ist die Hardware-System-Area.
SCT	<section-name> (string)	Der Adressraum ist eine Dumpfile-Section, die durch <section-name> spezifiziert wird.

Die Felder „ASEL“ und „ASID“ sind überschreibbar und hellgesteuert.

Bei Angabe von ALET plus TID muss Letztere mit einem Bindestrich an ALET angefügt werden. Soll ein task- oder systemspezifischer Adressraum als Datenraum eingestellt werden, braucht lediglich „ALT“ als „ASEL“ (plus <tid> als „ASID“) eingegeben werden.

Standardmäßig wird für Bereiche aus dem Benutzeradressraum die TID im Feld „ASID“ angezeigt. Will der Anwender die TSN zur Anzeige bringen, muss er „TSN“ in das Feld „ASEL“ eintragen.

Bei Ausgabe von Bereichen des Systemadressraums wird das Feld „ASEL“ auf SYS umgeschaltet. Die dem Fenster noch zugewiesene TID wird weiterhin im Feld ASID angezeigt. Diese TID kann wie bisher durch Eingabe von „TID“ im Feld „ASEL“ und <tid> im Feld „ASID“ verändert werden.

In einem VM2000-Gesamtsled kann mit der Eingabe von „ABS“ - beginnend beim Hypervisor - absolut adressiert werden, unabhängig davon, ob eine VM ausgewählt wurde. Mit der Eingabe „RM“ wird innerhalb einer ausgewählten VM adressiert.

Abkürzen der Eingaben

Alle Eingaben in das Feld „ASEL“ dürfen soweit abgekürzt werden, dass sie eindeutig bleiben.
Die TID darf bei der Eingabe in das Feld „ASID“ generell abgekürzt werden, soweit die bezeichnete Task hiermit eindeutig bestimmt ist. In der Regel sind die letzten vier Stellen der TID ausreichend. Ausgegeben wird stets die vollständige TID.

Beispiele für Eingaben in die Felder „symbolische Adresse“ und „Ausgabeformat“

Im Folgenden sollen anhand unterschiedlicher Eingabekombinationen in die Felder „symbolische Adresse“ und „Ausgabeformat“ einige wichtige Anwendungen beschrieben werden. Die Eingaben in das Feld „Ausgabeformat“ sind bis auf Eindeutigkeit abkürzbar.

- Lokalisieren eines automatisch auffindbaren Kontrollblocks (z.B. EXVT)

  Eingabe: Kontrollblockname im Feld „symbolische Adresse“

  Ausgabe: Kontrollblockname im Feld „symbolische Adresse“
  CBA im Feld „Ausgabeformat“

- Lokalisieren eines manuell auffindbaren Kontrollblocks

  Annahme: Der Speicherbereich ist schon eingestellt.

  Eingabe: Kontrollblockname im Feld „symbolische Adresse“

  Ausgabe: Kontrollblockname im Feld „symbolische Adresse“
  CBM im Feld „Ausgabeformat“

- Lokalisieren eines Feldes im aktuell angezeigten Kontrollblock

  Annahme: Der Kontrollblockname ist im Fenster schon eingestellt.

  Eingabe: Feldname im Feld „symbolische Adresse“

  Ausgabe: Kontrollblockname im Feld „symbolische Adresse“
  CBM oder CBA im Feld „Ausgabeformat“

- Lokalisieren eines Feldes in einem „automatischen“ Kontrollblock

  Eingabe: Feldname im Feld „symbolische Adresse“

  Ausgabe: Kontrollblockname im Feld „symbolische Adresse“
  CBA im Feld „Ausgabeformat“

- Manuelles Lokalisieren eines „automatischen“ Kontrollblocks

  Annahme: Der Speicherbereich ist schon eingestellt.

  Eingabe: Kontrollblockname im Feld „symbolische Adresse“
  CBM im Feld „Ausgabeformat“ oder Eingabe im Feld „Absolutadresse“

  Ausgabe: Kontrollblockname im Feld „symbolische Adresse“
  CBM im Feld „Ausgabeformat“

- Überlagerung eines Bereichs mit einem Kontrollblock ab Feldnamen

  Annahme: Der Speicherbereich ist schon eingestellt.

  Eingabe: Feldname im Feld „symbolische Adresse“
  CBM im Feld „Ausgabeformat“

  Ausgabe: Kontrollblockname im Feld „symbolische Adresse“
  CBM im Feld „Ausgabeformat“

- Überlagerung eines Bereichs mit einem Kontrollblock ab Relativadresse

  Annahme: Der Speicherbereich ist schon eingestellt.

  Eingabe: Kontrollblockname im Feld „symbolische Adresse“
  Relativadresse (rel. zum Kontrollblockanfang)

  Ausgabe: Kontrollblockname im Feld „symbolische Adresse“
  CBM im Feld „Ausgabeformat“

- Anzeigen des Moduls, in dem ein Kontrollblock liegt

  Eingabe: NAM im Feld „Ausgabeformat“

  Ausgabe: Modulname im Feld „symbolische Adresse“
  NAM im Feld „Ausgabeformat“

- Umschalten von symbolischer Darstellung auf Dumpformat

  Eingabe: D im Feld „Ausgabeformat“

  Ausgabe: Modulname im Feld „symbolische Adresse“
  D im Feld „Ausgabeformat“

- Anzeige eines Bereichs in einem Modul

  Eingabe: Modulname (eventl. plus Relativadresse) im Feld „symbolische Adresse“

  Ausgabe: Modulname im Feld „symbolische Adresse“
  Anzeige im Feld „Ausgabeformat“ bleibt; CBA, CBM oder NAM werden zu D

- Umschalten vom Dumpformat auf disassembliertes Format

  Eingabe: ASS im Feld „Ausgabeformat“

  Ausgabe: Modulname im Feld „symbolische Adresse“
  ASS im Feld „Ausgabeformat“

5.3 Bedienung

5.3.1 Grundfunktionen

- DAMP aufrufen
- Programmablauf steuern
- Diagnoseobjekt zuweisen und öffnen
- Diagnosefenster verändern
- DAMP unterbrechen und fortsetzen
- DAMP beenden

5.3.1.1 DAMP aufrufen

DAMP wird in dem System, an dem sich das Diagnoseobjekt befindet, mit **/START-DAMP** aufgerufen. DAMP wird dann von der unter IMON hinterlegten Benutzerkennung gestartet. Das Ladeprogramm lädt je nach gewählter Funktion die Verarbeitungsmodule aus der eingestellten Modulbibliothek nach.

Ist DAMP in einem System nicht standardmäßig installiert, oder soll DAMP mit speziellen Benutzeroptionen (siehe ab "Modifikationen durch den Benutzer (Spezialfenster OPTIONS)") gestartet werden, muss **/START-EXECUTABLE-PROGRAM** mit dem Namen der gegebenenfalls angepassten Ladephase von DAMP (Auslieferungsname: SYSPRG.DAMP.<ver>) verwendet werden.

Für die Analyse im aktiven System muss die aufrufende Benutzerkennung die Lese-Testprivilegierung 8 besitzen. Die Privilegierung muss mit dem Kommando MODIFY-TEST-OPTIONS vorher aktiviert werden.

Im **Dialogbetrieb** werden nach Aufruf von DAMP die programmierbaren Tasten automatisch mit Funktionen belegt. Anschließend erscheint die DAMP-Bildschirmmaske mit dem HELP-Fenster (ausgenommen im Batch- und Prozedurbetrieb).

DAMP kann auch als **Batchjob** gestartet werden (siehe "Batch- und Prozedurbetrieb, Anweisungsfolgen"). Die Eingaben werden dann von SYSDTA gelesen und wie Eingaben in die Kommandozeile bearbeitet. Diese Funktion ist überwiegend für das Erzeugen von Listen einsetzbar.

DAMP kann auch im **Prozedurbetrieb** gestartet werden (siehe "Batch- und Prozedurbetrieb, Anweisungsfolgen"). Diese Funktion eignet sich vor allem bei einem standardmäßigen Vorablauf einer „Diagnose am Bildschirm“.

5.3.1.2 Programmablauf steuern

Sie steuern den Programmablauf über die automatisch programmierten P-Tasten, die K-Tasten, die F-Tasten, durch Markieren mit der Taste MAR und über Anweisungen.

5.3.1.3 Diagnoseobjekt zuweisen und öffnen

Das Diagnoseobjekt (Dumpdatei oder System) lässt sich auf zwei Arten zuweisen:

- Mit der Anweisung `OPEN-DIAGNOSIS-OBJECT dumpfilename` wird ein SLED, SNAP-Dump, System-, User- oder Areadump zur Diagnose zugewiesen. Es empfiehlt sich, diesen Dumpdateien vor oder während des Programmlaufs die Linknamen #0,#1,...,#9 zuzuweisen. Dann lässt sich die Anweisung folgendermaßen vereinfachen:

  `OPEN-DIAGNOSIS-OBJECT #n (n = 0,...,9)`

  Die Angabe von teilqualifizierten Dateinamen oder Wildcards im Dateinamen ist erlaubt. Wenn eine Datei durch Teilqualifizierung bzw. den Wildcardnamen eindeutig bestimmt ist, öffnet DAMP diese Datei.

  Mit dem Parameter KIND-OF-OBJECT kann bestimmt werden, ob das Dumpobjekt als BS2000-Objekt, SELF-LOADER oder als PAM-Datei geöffnet werden soll. Siehe dazu auch Anweisung `OPEN-DIAGNOSIS-OBJECT` "OPEN-DIAGNOSIS-OBJECT Eröffnen eines Diagnoseobjekts für die Bearbeitung".

  Mit der Anweisung `OPEN-DIAGNOSIS-OBJECT *SYSTEM` weisen Sie das aktive System als Diagnoseobjekt zu. Die aufrufende Benutzerkennung muss dazu die Lese-Testprivilegierung 8 besitzen.
- Über die List-Maske lässt sich eine Dumpdatei auswählen, zuweisen und öffnen, und zwar auch dann, wenn man keine Liste ausdrucken möchte. In einem Feld können Sie den Namen der Dumpdatei auch teilqualifiziert oder mit Wildcardsyntax angeben und dann aus den vom System gefundenen Dateien die gewünschte auswählen. Näheres siehe unter „Listen erzeugen und ausdrucken (Spezialfenster LIST)“.

5.3.1.4 Diagnosefenster verändern

Da die Diagnosefenster im Diagnosefeld der Bildschirmmaske eingeblendet werden, können sie nicht länger sein als das Diagnosefeld, d.h., die Maximallänge beträgt 19 Zeilen. Die Diagnosefenster können jedoch verkürzt werden, wenn beispielsweise mehrere Fenster gleichzeitig im Diagnosefeld eingeblendet werden sollen. Die Mindestlänge eines Diagnosefensters beträgt 2 Zeilen.

Die Reihenfolge, die Länge, der Inhalt und das Ausgabeformat lassen sich beeinflussen durch:

- die Anweisung `MODIFY-SCREEN-LAYOUT`
- die Programmtasten P1 bis P9 oder die Eingabe der Fensternummer (0...9, 21...99) in die Kommandozeile
- Markieren von Adressfeldern, Schlagworten und bestimmten Ausgabezeilen (Task-, PCB-, Trefferzeilen etc.)
- Vorwärts- und Rückwärtsblättern mit --, ++, -, +, -n, +n sowie den Tasten F3 / F1 oder den entsprechenden Programmtasten (siehe „Blättern in einem Diagnosefenster")
- Eingaben in die Eingabefelder der Kopfzeilen
- die Anweisungen `SHOW-EDITED-INFORMATION`, `START-PATTERN-SEARCH`, `START-LIST-GENERATION`, `START-OPTION-DIALOG`, `START-PRODAMP-EDITOR`

Die Diagnosefenster sind zu Beginn der Bearbeitung mit Standardwerten besetzt:

- Standardlänge bei den Diagnosefenstern ist 19 Zeilen. Nur bei den Fenstern W2 und W3 wird die Fensterlänge zunächst der tatsächlichen Informationsmenge angepasst.
- Das Ausgabeformat bei den Dumpfenstern (W4 - W9 und W21 - W99) ist standardmäßig Dump-Format (Anzeige D). Diese Fenster sind als unbenutzt (keine Speicherinhalte) gekennzeichnet.
- Die Ausgabe im Format „RM“ oder „ABS“ wird verwendet, wenn keine virtuelle Adressierung möglich ist. Es wird dann die Seite „0“ angezeigt.

Nach Aufruf von DAMP wird das HELP-Fenster (W1) mit der Kapitelübersicht der Help-Informationen gezeigt.

Bei Änderung des Diagnoseobjekts durch die Anweisung `OPEN-DIAGNOSIS-OBJECT` werden alle zuvor getroffenen Zuweisungen an die Diagnosefenster zurückgesetzt. Erhalten bleiben lediglich die Einstellungen für Fensterlänge und Ausgabeformat. Eine Ausnahme bilden das LIST- und das PRODAMP-Fenster, die auch beim Wechsel des Diagnoseobjekts erhalten bleiben.

Bildschirminhalt wiederherstellen

Wird durch eine Leitungsstörung oder eine Operator-Meldung der Bildschirminhalt verschoben, kann mit K3 der vorherige Zustand wiederhergestellt werden.

Die Anweisung MODIFY-SCREEN-LAYOUT

Mit der Anweisung `MODIFY-SCREEN-LAYOUT` zieht man das angegebene bzw. die angegebenen Fenster an den Anfang der Fensterfolge und legt auf Wunsch neue Fensterlängen fest (siehe "MODIFY-SCREEN-LAYOUT Neue Reihenfolge und Größe der Diagnosefenster festlegen").

Die Programmtasten P1 - P15

Mit den Programmtasten P1 bis P9 lassen sich die zugehörigen Fenster W1 bis W9 an den Anfang der Fensterfolge vorziehen und ins Diagnosefeld einblenden. Nach dem Markieren eines Adressfeldes wird der zugehörige Speicherbereich dem Diagnosefenster zugewiesen, das über eine der Tasten P1 bis P9 ausgewählt wird.

Mit den Programmtasten P10 bis P12 positioniert man den Cursor auf das Eingabefeld „Fensterlänge“ der drei ersten angezeigten Diagnosefenster. Werden weniger als drei Fenster am Bildschirm gezeigt, so positionieren die Tasten P12 und gegebenenfalls auch P11 auf die Kommandozeile.

Wird eine Eingabe mit der Programmtaste P13 übertragen, leitet DAMP die Eingabe unbesehen an eine eventuell aktive PRODAMP-Prozedur weiter. Damit ist es möglich, für PRODAMP-Anwendungen eine eigene Benutzeroberfläche zu schreiben.

Die Programmtasten P14 und P15 können Sie zum Blättern in der DAMP-Meldungs-Historie verwenden. Mit P14 können Sie in den Meldungszeilen (Zeile 2 und 3) zurückund mit P15 vorblättern.

Die Programmtasten P1 bis P15 werden beim Aufruf von DAMP automatisch geladen. Sind die Programme der P-Tasten (etwa nach einem Task-Wechsel über OMNIS) verlorengegangen, können Sie die P-Tasten neu laden. Dazu unterbrechen Sie das Programm mit der Taste K2 und geben anschließend das Kommando RESUME-PROGRAM ein.

Bei Datensichtstationen mit der Funktion „Lesen P-Bereiche“ (ab DSS 9762) sichert DAMP vor dem Beschreiben der Programmtasten den aktuellen Inhalt, falls die DAMP-Option **Save P-Keys = yes** nicht umgestellt wurden (siehe "Modifikationen durch den Benutzer (Spezialfenster OPTIONS)"). Bei jeder Unterbrechung von DAMP mittels K2 und bei Beendigung von DAMP wird der zuletzt gesicherte Inhalt der Programmtasten wieder geladen. Änderungen der Programmtasteninhalte bei DAMP-Unterbrechungen werden bei anschließendem `/RESUME-PROGRAM` gesichert.

Alternative zu den Programmtasten P1 - P9 und P13

Alternativ zu den Programmtasten P1 bis P9 sowie P13 kann die zugehörige Nummer der Programmtaste (1 bis 9 oder 13) in die Kommandozeile eingegeben und mit Drücken der Taste DUE aktiviert werden.

Blättern in einem Diagnosefenster

Mit den Funktionstasten F3 und F1 können Sie innerhalb eines Fensters vor- und zurückblättern. Diese Funktion von F1 / F3 ist auch auf die beiden letzten Programmtasten abgebildet: bei 20 Programmtasten (Terminal 9750) entspricht P19 = F1 und P20 = F3.

In der Regel werden auch die Eingaben +, -, ++, --, +n, -n zum Blättern unterstützt.

+	blättert um eine Fensterlänge vorwärts
-	blättert um eine Fensterlänge rückwärts
++	blättert zum Ende der Liste, des Kapitels, des Moduls
--	blättert an den Anfang der Liste, des Kapitels, des Moduls
+n	blättert um n Zeilen vorwärts
-n	blättert um n Zeilen rückwärts

Werden mehrere Fenster am Bildschirm gezeigt, wirkt das Blättern auf das Fenster, mit dem zuletzt gearbeitet wurde. Mit F1 oder „-“ wird zunächst nur bis zum Anfang des jeweiligen Moduls, des Kontrollblocks oder des Kapitels zurückgeblättert. Mit einem weiteren Blättern kann die Grenze überblättert werden.

Markieren

Der Inhalt von Standard-Dumpfenstern ist in der Regel ein Ausschnitt aus dem System- oder Benutzerspeicher mit Befehls-, Daten- und Adressfeldern. Durch das Markieren eines Adressfeldes - d.h. durch Positionieren des Cursors in das Adressfeld und Drücken der Taste MAR - wird diese Adresse als Anfangsadresse des gleichen bzw. eines neuen Dumpfensters festgelegt. Ebenso können Adressen im Stack-Fenster (W3) und diverse Daten in funktionsgebundenen Spezial-Fenstern (FIND, AUDIT etc.) markiert werden.

Die Ausgabe einer markierten Speicherstelle erfolgt nach Möglichkeit nicht in das Fenster, das die Markierung enthält.
Um einer markierten Adresse ein bestimmtes Standard-Dumpfenster zuzuweisen, drücken Sie nach dem Markieren eine der Programmtasten P4 bis P9 (für die Ausgabe in einem der Fenster W4 bis W9) oder geben Sie in der CMD-Zeile die Fensternummer (4..9, 21..99) ein und drücken anschließend DUE. Standard-Dumpfenster können auch über die Anweisung MODIFY-SCREEN-LAYOUT zugewiesen werden.

Wenn Sie nach dem Markieren kein Dumpfenster auswählen, also nur DUE drücken, dann wählt DAMP das nächste Standard-Dumpfenster, das im Diagnosefeld eingeblendet ist. Wenn kein anderes Dumpfenster mehr im Diagnosefeld zur Verfügung steht, dann wählt DAMP das aktuell eingestellte Standard-Dumpfenster.

Abhängig von der Anzahl der im Diagnosefeld eingeblendeten Fenster kann die Ausgabe gleichzeitig in bis zu 6 Standard-Dumpfenster erfolgen. Wenn mehr Felder markiert sind, dann werden die überzähligen Felder ignoriert.

Hellgesteuerte Schlagworte im Help-Fenster (W1) oder die Task- bzw. die PCB-Zeilen im Status-Fenster (W2) lassen sich ebenfalls markieren. Nach dem Drücken der Taste DUE erhalten Sie genauere Information über den markierten Begriff bzw. die markierte Task oder den PCB.

In Standard-Dumpfenstern ist zudem das Markieren der ersten Spalte in jeder Zeile möglich. Die so markierte Zeile wird bei der nächsten Ausgabe an den Anfang des Fensters positioniert.

Durch nochmaliges Drücken der Taste MAR werden Markierungen zurückgenommen.

Umwandeln der Adressdarstellung bei Kontexten im x86-HSI

Auf x86-Servern liegen Speicheradressen bei x86-Kontexten im Format „Little Endian“ vor. BS2000 verwendet das Format „Big Endian“ bei der Adressdarstellung. Im laufenden Betrieb konvertiert die Firmware die Adressen beim HSI-Übergang jeweils in das richtige Format.

DAMP stellt bei der Aufbereitung von BS2000-Tabellen in Spezialfenstern (z.B bei der Aufbereitung eines PCB mit x86-Kontext in W3) die Adressen automatisch im Format Big Endian von BS2000 dar.

Wenn aber ein Speicherbereich mit x86-Kontext in ein Standard-Dumpfenster ausgegeben wird, dann wird der angegebene Bereich von DAMP unverändert ausgegeben. Dies ist auch dann der Fall, wenn eine DSECT über den Bereich gelegt wird. Durch Markieren und Drücken von F4 können Sie eine Adresse vor der Ausgabe des zugehörigen Speicherbereichs von DAMP konvertieren lassen. Wenn ein Standard-Dumpfenster als Ausgabefenster eingestellt ist, dann wird der Bereich ab der konvertierten Adresse in das ausgewählte Fenster ausgegeben. Die Adressdarstellung an der markierten Stelle wird aber **nicht** umgewandelt.

Die möglichen Ausgabefenster sind im Abschnitt „Markieren" beschrieben.

F4 wirkt nur bei Objekten des x86-HSI.

5.3.1.5 DAMP unterbrechen und fortsetzen

Mit der Taste K2 können Sie jederzeit in die Kommandoebene umschalten und die Arbeit mit DAMP unterbrechen. In der Kommandoebene sind alle Systemkommandos erlaubt.

Mit dem Kommando RESUME-PROGRAM kehren Sie in das Programm DAMP zurück. Sie können jedoch auch von der Kommandoebene aus die Arbeit mit DAMP fortsetzen, und zwar über die STXIT-Unterbrechungsroutine mit dem Kommando INFORM-PROGRAM (siehe Abschnitt „Auf Systemebene“).

5.3.1.6 DAMP beenden

DAMP wird beendet durch

- Eingabe der Anweisung **END** in der Kommandozeile
- durch Drücken der Taste K1. Eine Rückfrage erfolgt nur bei entsprechend eingestellter Benutzeroption (siehe "Modifikationen durch den Benutzer (Spezialfenster OPTIONS)").

Daneben stehen noch weitere Möglichkeiten zum Beenden von DAMP zur Verfügung:

- Drücken der Taste K2 (unterbrechen des laufenden Programms und Wechsel in die Kommandoebene) und Eingeben des Kommandos INFORM-PROGRAM mit einem der folgenden Texte:

`/INFORM-PROGRAM MSG='*END'`
`/INFORM-PROGRAM MSG='*HALT'`
`/INFORM-PROGRAM MSG='*TERMINATE'`
	Beenden ohne Speicherabzug

`/INFORM-PROGRAM MSG='*DUMP'` `/INFORM-PROGRAM MSG='*TERMD'`	Beenden mit Speicherabzug

5.3.2 Diagnosedaten ausgeben

Da es für die Lösung eines Software-Problems im Allgemeinen kein „Kochrezept“ gibt, kann Ihnen diese Beschreibung nur sagen, welche Informationen Sie in einem Diagnoseobjekt finden können, und wie Sie sie finden. Wonach Sie suchen, sollten Sie daher selbst wissen.

Die Diagnose wird eingeleitet durch den Aufruf des Programms DAMP. Es erscheint das HELP-Fenster (W1) mit der Kapitelübersicht.

Der weitere Ablauf der Diagnose wird durch DAMP-Anweisungen, Funktionstasten, Veränderungen von Eingabefeldern und Markieren von Markierungsfeldern gesteuert.

Das Einschalten eines Diagnosemitschnitts ist sinnvoll, damit zu einem späteren Zeitpunkt die Diagnoseaktionen von der gleichen oder einer anderen Diagnoseinstanz nachvollzogen werden können.

5.3.2.1 Automatisches Interpretieren der Ausgabedaten

DAMP versucht bei der Ausgabe des Status- und Stack-Fensters sowie bei der Ausgabe eines Speicherbereichs in symbolischer Form (Überlagerung mit einer DSECT), den Inhalt der einzelnen Felder sinnvoll zu interpretieren. Insbesondere wird versucht, Adressen zu relativieren, d.h. in der Form Modul + Distanz darzustellen. Dabei gelten folgende Regeln:

In Dumpfenstern
wird die Anfangsadresse des Fensters modulrelativ interpretiert. Außerdem beziehen sich alle ausgegebenen Relativadressen auf den Anfang des im Fenster sichtbaren Moduls. Der Inhalt wird maximal bis zum Ende des Moduls aufbereitet.

Wird diese Darstellung nicht gewünscht, z.B. wenn man die Relativadressen auf den Anfang einer Tabelle beziehen möchte, die mitten in einem Modul liegt, sind folgende Schritte durchzuführen:

- Positionieren auf den Tabellenanfang,
- Eingabe von „ALT“ in das Feld „ASEL“.

DAMP nimmt dann die aktuelle Anfangsadresse des Fensters als Basis für die Relativierung. Außerdem wird die Ausgabe auch über das Modul-Ende hinaus fortgesetzt.

In Diagnoseobjekten werden Adressen generell als 31-Bit-Adressen (/390-Objekte) bzw. 32-Bit-Adressen (x86-Objekte) interpretiert. Wird dies nicht gewünscht, muss der Bereich mit einer DSECT überlagert und wie für symbolisch aufbereitete Bereiche verfahren werden (siehe „Abschnitt „Symbolische Ausgabe“).

In TU-PCBs
werden Adressen je nach dem im PCB eingestellten Adressierungsmodus als 24-Bit-, 31-Bit- oder 32-Bit-Adressen interpretiert. 32-Bit-Adressen sind nur bei x86-Objekten möglich. Für die Relativierung werden Module aus den angeschlossenen nicht-privilegierten Subsystemen sowie CSECTs eines evtl. geladenen Benutzerprogramms berücksichtigt. Die Mehrzweckregister 0 und 1 werden generell nicht relativiert.

Ist in dem gezeigten PCB das Access-Register-Mode-Flag (AR-Mode-Flag) gesetzt und enthält das gleichnamige Access-Register einen Wert (ALET) ungleich null, so führt das Markieren eines Mehrzweckregisters gleich zur Zuweisung des entsprechenden Datenraums und dessen Anzeige im gewünschten Fenster.

In TPR-PCBs
werden Adressen als 31-Bit-Adressen (/390-Objekte) bzw. 32-Bit-Adressen (x86-Objekte) interpretiert. Für die Relativierung werden Module aus dem Control-Program (CP) und aus all denjenigen privilegierten Subsystemen berücksichtigt, die geladen sind.

Für TPR-PCBs, in denen das AR-Mode-Flag gesetzt ist, wird analog zu den TU-PCBs (siehe oben) verfahren.

In symbolisch aufbereiteten Speicherbereichen
werden Adressen generell als 31-Bit-Adressen (/390-Objekte) bzw. 32-Bit-Adressen (x86-Objekte) interpretiert. In Spezialfällen (etwa wenn Benutzer-Parameterlisten betrachtet werden) kann dies unerwünscht sein. In diesem Fall lässt sich mit der Anweisung

```
MODIFY-OBJECT-ASSUMPTIONS ADDRESSING-MODE=*PAR(<control-block>, *NXS/ *XS31)
```

für den angegebenen Kontrollblock eine Interpretation der Adressen im 24-Bit- bzw. 31-Bit-Modus vereinbaren. Abhängig davon, ob sich die Daten im Benutzerspeicher oder im Systemspeicher befinden, werden für die Relativierung alle Module bzw. nur Module aus dem Control-Program und aus Klasse-4-Subsystemen berücksichtigt.

In funktionsgebundenen Fenstern
wie dem TRACE-, FIND-, SUSY oder TABL-Fenster werden Adressen generell wie bei Speicherbereichen interpretiert.

Ausnahme

Lediglich beim AUDIT-Fenster wird beim TU-AUDIT die Relativierung wie beim TU-PCB durchgeführt.

5.3.2.2 Status-Informationen ausgeben

Eröffnen Sie eines der Diagnoseobjekte SLED, SNAP-Dump oder aktives System bzw. geben Sie nach Eröffnen eines Systemdumps den Modus TSK im Status-Fenster (W2) ein, so erhalten Sie Status-Informationen über vorkommende Tasks. Diese Datenmenge überschreitet in der Regel die maximale Fensterlänge. Durch Blättern können die restlichen Daten ausgegeben werden.

Markieren einer Task-Zeile bzw. Eintragen der gewünschten Task in das entsprechende Eingabefeld der Kopfzeile und Drücken der Taste DUE bewirken, dass Informationen über alle PCBs dieser Task (Modus PLK) auf dem Bildschirm ausgegeben werden. Ist die eingegebene TSN kürzer als vier Zeichen, so wird sie automatisch mit führenden Nullen auf vier Zeichen erweitert. Beginnt eine TSN mit Leerzeichen (etwa bei Systemtasks), so müssen diese geschrieben werden.

Die Fensterlänge des Status-Fensters (W2) bei PCB-Ausgabe wird selbsttätig auf die notwendige Größe eingestellt. Wird der erste PCB angezeigt, führt F1 wieder zurück zur Taskübersicht.

Bei System-, Area- und Userdumps wird beim Öffnen der Dumpdatei sofort der Modus PLK eingestellt, d.h. die Liste der PCBs der verursachenden Task ausgegeben.

Durch Eingabe von „SLK“ im Modus-Feld werden die TPR-Program-Manager-Stacks ausgegeben.

Im Modus INF wird stets das aktuell geöffnete Objekt beschrieben. Falls bei Dumpdateien mit mehreren Objekten eine Auswahl möglich ist, erfolgt sie durch das Markieren des gewünschten Objekts. Mit Einstellen des Modus INF im Statusfenster und Drücken der Taste F1 kann diese Auswahl rückgängig gemacht werden.

Es ist stets möglich, durch Eingabe in das Modus-Feld zwischen den verschiedenen Modi (INF, TSK, PLK, SLK) des Statusfensters zu wechseln.

Nach dem Markieren eines PCB wird der Stack dieses PCB im Stack-Fenster (W3) ausgegeben. Wird kein PCB markiert, aber das Stack-Fenster durch die Programmtaste P3 bzw. die Anweisung `MODIFY-SCREEN-LAYOUT` aktiviert, so wird der Stack des ersten PCB ausgegeben.

5.3.2.3 Stack-Inhalt ausgeben

Inhalte von Programmsteuerblöcken (PCBs) werden im Stack-Fenster (W3) ausgegeben. Jedes im Stack-Fenster gezeigte Adressfeld ist markierbar.

Liegt ein **System-, User- oder Areadump** vor, gilt Folgendes:
Um sich den ersten PCB dieser Task ausgeben zu lassen, genügt das Drücken der Taste P3. Es erscheint daraufhin das Stack-Fenster (W3) mit dem Inhalt des ersten PCBs (voreingestellt).
Soll der Inhalt eines anderen PCBs ausgegeben werden, drücken Sie zunächst die Taste P2. Sie erhalten daraufhin das Status-Fenster (W2) mit der PCB-Liste. Markieren Sie die Zeile mit dem gewünschten PCB und drücken anschließend die Taste P3. Nun erscheint das Stack-Fenster (W3) mit dem Inhalt des markierten PCBs.

Liegt ein SLED oder SNAP vor, muss zunächst eine Task ausgewählt werden. Standardmäßig ist die Task 1 eingestellt. Die Auswahl erfolgt durch Markieren einer Task-Zeile im Status-Fenster (W2) und Übertragen mit DUE . Nun erscheint die PCB-Liste der markierten Task im Status-Fenster. Markieren Sie die Zeile mit dem gewünschten PCB und drücken anschließend die Taste P3. Nun erscheint das Stack-Fenster (W3) mit dem Inhalt des markierten PCBs.

Um sich die Daten ausgeben zu lassen, auf die die Register des PCBs verweisen, markieren Sie den Befehlszähler bzw. ein Adressfeld in den Registern des PCB. Durch anschließendes Drücken einer der Programmtasten P4 bis P9 weisen Sie die markierte Adresse einem Dumpfenster zu. Alternativ kann die Auswahl des Dumpfensters über die Anweisung MODIFY-SCREEN-LAYOUT oder durch Angabe der Fensternummer (4...9, 21...99) in der Kommandozeile und Übertragen mit DUE erfolgen.

Sie können bis zu sechs Adressen gleichzeitig markieren und Dumpfenstern zuweisen. Dazu geben Sie zunächst die Anweisung `MODIFY-SCREEN-LAYOUT` zum Beispiel in der folgenden Form an:

```
MODIFY-SCREEN-LAYOUT FIRST=3(SIZE=10), SEC=4(SIZE=2), THIRD=5(SIZE=2),
                     FOURTH=6(SIZE=2)
```

Die Fenster erscheinen in der angegebenen Reihenfolge und Länge. Durch anschließendes Markieren von drei Adressfeldern des PCB und Drücken der Taste DUE werden die markierten Adressen den Dumpfenstern W4, W5 und W6 zugeordnet.

Wollen Sie sich statt der PCB-Information die TPR-Program-Manager-Information ausgeben lassen, tragen Sie

- „SLK“ im Eingabefeld „Modus-Auswahl“ des Status-Fensters (W2) (siehe "Das Status-Fenster (W2)") bzw.
- „SPL“ im Eingabefeld „Stack-Auswahl“ des Stack-Fensters (W3) ein.

5.3.2.4 Systemtabellen ausgeben

Folgende Systemtabellen können von DAMP im zugewiesenen Diagnoseobjekt automatisch lokalisiert und im „symbolischen Format“ (entsprechend ihrem DSECT-Layout) auf ein Dumpfenster ausgegeben werden:

Bezeichnung	Abk.	DSECT-Name
Executive Vector Table	XVT	EXVT
Task Control Block	TCB	ETCB
Job Control Block	JCB	EJCB
System Virtual Memory Table	SVMT	EVSMT
User Virtual Memory Table	UVMT	EVUMT

Tabelle 5: Automatisch lokalisierbare Systemtabellen

Nach Eingabe des DSECT-Namens im Eingabefeld „symbolische Adresse“ eines Dumpfensters (W4 - W9 bzw. W21 - W99) wird der lokalisierte Speicherbereich ab Tabellenanfang symbolisch aufbereitet ausgegeben: mit Distanzadresse, DSECT-Feldnamen, Feldinhalt und möglicher Interpretation gemäß Felddefinition. Bei den taskspezifischen Tabellen TCB, JCB und UVMT ist bei SLED- und SNAP-Dateien unter Umständen auch die gewünschte Task einzugeben. Ansonsten wird die Task 1 bzw. die zuletzt angesprochene Task als Default-Wert genommen.

```
DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>


EVSMT                            +00000=711A5000 TID=000A00CB          W4,CBA,L19
000 EVSMID  : E2E5D4E3                          | 004 EVSMVER : 00F4F6F0
008         : 0000000000000000                  | 010 EVSMRASZ: 00010000 = 65536
014 EVSMRASM: 00010000 = 65536                  | 018 EVSMRASI: 00010000 = 65536
01C EVSMMMLM: 0000FFFF = 65535                  | 020 EVSMHFR#: 0000FFFF = 65535
024 EVSM#AFR: 00010000 = 65536                  | 028 EVSMC1FR: 000003F2 =  1010
02C EVSMMBFR: 00000000 =     0                  | 030 EVSMPPTP: 710E0100 = EMMENTAL+40100
034 EVSMPPTE: 710E0900 = EMMENTAL+40900
038 EVSMPPTA: 710E0900 = EMMENTAL+40900
03C EVSMPPAT: 710E0000 = EMMENTAL+40000
040 EVSMPPAE: 710E0040 = EMMENTAL+40040
044         : 00000000                          | 048 EVSMPPID: 0000000080000600
050 EVSMPPTI: 02                                | 051         : 00
052 EVSMPPSH: 000A     =    10                  | 054 EVSMPPM1: 0FFFE000
058 EVSMPPM2: 00001FFF =  8191                  | 05C EVSMPPRS: 0100     =   256
05E EVSMPPVS: 2000                              | 060 EVSMPPES: 0020     =    32
062         : 0000                              | 064 EVSMPPTR: 000E0900
068         : 00000000                          | 06C EVSMPAGP: 0000D149 = 53577
070 EVSM#CFR: 0000D0E8 = 53480                  | 074 EVSM#CFB: 0000D0E8 = 53480
CMD:
Key: 1=Help 2=Plk 3=PCB 4=EVSMT  5=Dump   6=Dump   7=Dump   8=Dump   9=Dump
```

Bild 27: Ausgabe der SVMT im symbolischen Format; Anzeige: EVSMT in der Keyzeile

An Stelle der in Tabelle 5 genannten DSECT-Namen ist auch das Eingeben eines Feldnamens aus diesen DSECTs möglich. Die Ausgabe beginnt dann mit der entsprechenden Speicheradresse.

Man kann innerhalb der DSECT mit den Tasten F3/F1 oder +/-/++/-- blättern. Dabei kommt man jedoch nicht aus dem DSECT-Bereich heraus. Das Verlassen der DSECT ist nur durch Umschalten auf ein anderes Layout im Feld „Ausgabeformat“ möglich.

Weitere Systemtabellen werden nicht automatisch lokalisiert. Sie müssen über Adressverkettungen selbst aufgefunden werden. Eine symbolische Ausgabe dieser Tabellen kann dann durch Überlagerung mit einer DSECT aus der Symboldatei erreicht werden (siehe Abschnitt „Symbolische Ausgabe").

Es ist auch eine „versetzte“ Überlagerung möglich: dabei wird der Speicherbereich nicht im Format der gesamten DSECT, sondern erst ab dem eingegebenen DSECT-Feldnamen aufbereitet.

5.3.2.5 Prozessor-lokale Bereiche ausgeben

Sie können sich die **lokalen XVTs** bei Multiprozessoren auf folgende Weise ausgeben lassen:

- Nach dem Öffnen eines SLED- oder SNAP-Dumps tragen Sie den Feldnamen EXVTLMD1 in das Eingabefeld „symbolische Adresse“ eines Dumpfensters und drücken DUE. Daraufhin wird ein Bereich aus der globalen XVT in diesem Fenster ausgegeben.
- Mit der Anweisung `MODFIY-SCREEN-LAYOUT` stellen Sie mehrere freie Dumpfenster bereit.
- Markieren Sie die Adressen in den Feldern EXVTLMD1, EXVTLMD2, ... und drücken Sie DUE; es werden die lokalen XVTs im Dump-Format auf den neuen Fenstern ausgegeben.
- Überlagerung der Speicherbereiche mit dem Format der EXVT ab Feld EXPROCAR (Beginn des lokalen Teils). Hierzu muss der Feldname EXPROCAR ins Eingabefeld „symbolische Adresse“ und das Schlüsselwort „CBM“ ins Eingabefeld „Ausgabeformat“ eingegeben werden.

Zur Vereinfachung bietet DAMP folgende alternative Möglichkeiten:

- Geben Sie die „Pseudo“-CSECT EXVT-XXX in das Feld „symbolische Adresse“ ein und drücken Sie DUE. XXX ist die 3-stellige sedezimale Nummer der gewünschten logischen Maschine.
- Ersetzen Sie EXVT-XXX durch EXVT oder EXPROCAR **und** geben Sie das Schlüsselwort „CBM“ ins Feld „Ausgabeformat“ ein. Anschließend drücken Sie DUE.

Beispiel

Ausgabe der lokalen XVTs bei Multiprozessoren

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


EXVT                            +00CE8=710014E8 SYS=00010001          W4,CBA,L 3
CE8 EXVTLMD1 : 71F9D800 = EXVT-001     | CEC EXVTLP02 : 71F9E800 = EXVT-002
CF0 EXVTLP03 : 71F9F800 = EXVT-003     | CF4 EXVTLP04 : 71FA0800 = EXVT-004
-------------------------------------------------------------------------------
EXVT                            +00000=71F9D800 SYS=00010001          W5,CBM,L11
000 EXPROCAR : E7E5E3F4                | 004          : 00000000
008 EXVTTSCE : 0000000000000000        | 010 EXVTCWKA : 00000000 =     0
014          : 00000000000000000000000000000000000000000000000000000000000000000000
034          : 00000000000000000000000000000000000000000000000000000000000000000000
054          : 00000000000000000000000000000000000000000000000000000000000000000000
074          : 00000000000000000000
080 EXVTTSPD : 7111E000 = EMMINIT1+3A000
084 EXVTTSTD : 0010307F                | 088 EXVTTST@ : 71103000 = EMMINIT1+1F00
08C EXVTASTO : 0011F040                | 090 EXVTSSPD : 00000000 =     0
094 EXVTSSTD : 00000000 =     0        | 098 EXVTSST@ : 00000000 =     0
-------------------------------------------------------------------------------
EXVT                            +00000=71F9E800 SYS=00010001          W6,CBM,L11
000 EXPROCAR : E7E5E3F4                | 004          : 00000000
008 EXVTTSCE : 0000000000000000        | 010 EXVTCWKA : 00000000 =     0
OMD:
Key: 1=Help 2=Tsk 3=PCB 4=EXVT   5=EXVT   6=EXVT   7=Dump   8=Dump   9=Dump
```

Bild 28: Ausgabe der lokalen XVTs im symbolischen Format

Außerdem können Sie eine **Prozessor-lokale PSA** (Processor Storage Area) automatisch lokalisieren, indem Sie die „Pseudo“-CSECT PSA-XXX eingeben.

5.3.2.6 Hardware-Informationen ausgeben

Bei SLED-Dateien lokalisiert DAMP die Hardware-Informationen und gibt diese auf einem Diagnosefenster aus.

Durch Eingeben des entsprechenden Schlüsselwortes im Eingabefeld „ASEL“ und eventuell eines Parameters im Eingabefeld „ASID“ eines Dumpfensters W4 - W9 bzw. W21 - W99 wird durch DAMP der entsprechende Speicherbereich lokalisiert und im eingestellten Ausgabeformat auf diesem Dumpfenster ausgegeben.
Folgende Eingaben sind möglich:

- in ASEL: „HSA“
  in ASID: Eingabe wird ignoriert

- in ASEL: „PSS“
  in ASID: <prozessor-nummer> (sedezimal)

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


                                +00000=  000000 PSS=0                 W4.D  ,L19
00000000 (00000):E2E3C1E3  0600E500  00000000  00000000   <==>  STATV
00000010 (00010):00000000  00000000  00000000  00000000   <==>
00000020 (00020):31020001  21600000  DC400000  00000000   <==>  -
00000030 (00030):00000000  00000000  00000000  00000000   <==>
00000040 (00040):0125C000  00280000  040A0000  80FFFFFF   <==>  ~~~
00000050 (00050):FFFFFFFE  78F9AB00  FFFFFFFF  FFFFF000   <==>  ~~~9~~~~~~0
00000060 (00060):00000000  00000000  00000000  00000000   <==>
00000070 (00070):00000000  00000000  00000000  00000000   <==>
00000080 (00080):0CB5FC40  0040A07F  00173440  00000000   <==>    "
00000090 (00090):00000000  00120200  FF000000  0010507F   <==>  ~&"
000000A0 (000A0):00000000  00000000  00000000  00000000   <==>
000000B0 (000B0):00000000  0040A07F  FF000000  70FD8BE8   <==>   "~}Y
000000C0 (000C0):00000000  00000000  00000000  00000000   <==>
000000D0 (000D0):00000000  00000000  00000000  00000000   <==>
000000E0 (000E0):00000040  711B9011  F12663D4  711B9000   <==>   1M
000000F0 (000F0):00000000  00000000  17D78400  711B9DD8   <==>  PdQ
00000100 (00100):0003F480  7103F480  71000000  71000800   <==>  44
00000110 (00110):71266550  711B9E20  711B9000  00000000   <==>  &
CMD:
Key: 1=Help 2=Tsk 3=PCB 4=PSS-0 5=Dump   6=Dump   7=Dump   8=Dump   9=OPTS
```

Bild 29: Ausgabe des „Processor Saved Status“ im Dumpfenster W4

5.3.2.7 Speicherbereiche ausgeben

Speicherbereiche, die nicht automatisch lokalisierbar sind, werden durch Markieren entsprechender Adressfelder, durch explizite Eingabe der Adresse in die Eingabefelder eines Dumpfensters oder durch Weiterblättern in einem angezeigten Speicherbereich aufgefunden.

Der Speicherbereich wird im zuletzt eingestellten Ausgabeformat ausgegeben. Das Ausgabeformat kann in der Kopfzeile geändert werden. Dadurch ändert sich jedoch auch die Menge der angezeigten Informationen:

Format	Anzeige	Eingabe	Ausgabelänge pro Zeile	Maximalinfo pro Fenster
Dump	D	d,D	16 Byte	288 Byte
ASCII	ASC	asc,ASC	16 Byte	288 Byte
Hexa	HEX	h,H	32 Byte	576 Byte
Zeichen	CHR	c,C	64 Byte	1152 Byte
Assembler	ASS, XAS ASS XAS	a,A cas,CAS x,X	1 Stmt	
symbolisch	CBA CBM NAM	cba,CBA cbm,CBM n,N	max. 32 Byte	

Tabelle 6: Zusammenhang zwischen Ausgabeformat und ausgegebener Informationsmenge

Bei der Erstbelegung eines Dumpfensters wird standardmäßig Dump-Format D mit der Fensterlänge 19 eingesetzt.

Die symbolische Ausgabe kann nicht durch Änderung im entsprechenden Eingabefeld erreicht werden. Sie wird für die Ausgabe lokalisierbarer Tabellen bzw. für die Überlagerung eines Speicherbereichs mit einer DSECT (Angabe einer DSECT im Eingabefeld „symbolische Adresse“) automatisch eingestellt.

5.3.2.8 Symbolische Ausgabe

Ist ein Speicherbereich in Form einer Standard-DSECT strukturiert, dann kann DAMP die Ausgabe dieses Bereichs im Layout dieser DSECT durchführen. (Der Begriff DSECT steht im Folgenden für die Strukturen der verschiedenen Programmiersprachen, so z.B. für ASS-DSECTs, MODELs (SPL), STRUCTUREs (C).) Die entsprechenden Felder der DSECT werden gemäß ihrem Daten-Typ interpretiert. Numerische Werte werden sowohl sedezimal als auch als Adresse oder Dezimalwert ausgegeben, Zeichenketten nur als solche.

Da die Definitionen der Feldnamen in den benutzten DSECTs nicht immer ihrer Bedeutung entsprechen (z.B. CL4 statt AL4), können vereinzelt nicht ganz sinnvolle Interpretationen erscheinen. Eine Änderung dieser Darstellung erreicht man, indem man die DSECTs ändert und anschließend eine neue bzw. geänderte Symboldatei generiert (siehe „Abschnitt „Private Symbolelemente verwenden").

Neben den automatisch lokalisierbaren globalen und taskspezifischen Tabellen kann jeder beliebige Speicherbereich im Layout einer DSECT ausgegeben werden. Hierfür stehen zwei Varianten zur Verfügung:

Überlagern mit einer vollständigen DSECT

Um einen Speicherbereich mit einer DSECT zu überlagern, muss der Anfang des ausgegebenen Speicherbereiches dem DSECT-Anfang entsprechen. Eintragen des DSECT-Namens in das Eingabefeld „symbolische Adresse“ und Eingabe des Schlüsselworts „CBM“ ins Eingabefeld „Ausgabeformat“ führt zur Ausgabe im symbolischen Format. Die Eingabe des Schlüsselworts „CBM“ kann entfallen, wenn die DSECT von DAMP nicht automatisch lokalisiert werden kann.

Wird der Speicherbereich durch Eingeben einer absoluten Adresse in der Kopfzeile lokalisiert, kann auch gleich ein DSECT-Name mitgegeben werden. Der Speicherbereich wird dann sofort im symbolischen Layout ausgegeben.

```
DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>


ESTK                              +00000=739073A8 TID=000A00CB          W5,CBM,L19
000 ESTKTBLH: D7C3C2                    | 003          : 04
004 ESTKSIND: 84                        | 005 ESTKHSI  : 01
006 ESTKIND : 01                        | 007 ESTKIND1: 01
008 ESTKAFIN: 60                        | 009 ESTKEIAF: 00
00A ESTKPCBL: 01C8                      | 00C ESTKSLNK: 72F91008
010 ESTKISLK: 72F91008                  | 014 ESTKTSF1: 00
015 ESTKTSF2: 00                        | 016 ESTKTSS  : 00
017          : 00                       | 018 ESTKAUDM: 00000000 =     0
01C ESTKEXRT: 713974A0 = NLMLOCK5+0CA0  | 020 ESTKCLK  : 00000000 =     0
024 ESTKCCLK: 0000FFFF = 65535          | 028 ESTKSTXC: FF
029 ESTKSTXB: FF                        | 02A ESTKP1EV: EF
02B ESTKP1SE: FF                        | 02C ESTKPOST: 71E63140
030 ESTKBOID: 000A00CB                  |
034          : 000A00CBF1A941860006000500000000000000000
048 ESTKLPSC: 00000000 =     0          | 04C ESTKBRSE: 00000000 =     0
050 ESTKBRS@: 71249400 = ETMBOWK +18C0  | 054 ESTKCONF: 80
055 ESTKCLEV: EF                        | 056 ESTKSTID: 00
057 ESTKPLVL: 00                        | 058          : 0000000000000000
CMD:
Key: 1=Help 2=Plk 3=PCB 4=ESMFHD 5=ESTK   6=Dump   7=Dump   8=Dump   9=FIND
```

Bild 30: Ausgabe eines Stack im symbolischen Format. Der DSECT-Name wird in der Keyzeile angezeigt

Der Inhalt des Feldes „Ausgabeformat“ gibt an, ob eine DSECT von DAMP automatisch lokalisiert worden ist (Inhalt CBA = Control Block Automatic) oder ob der Benutzer die DSECT positioniert hat (Inhalt CBM = Control Block Manual). Im Falle von CBM bedeutet dies, dass bei Eingabe eines Feldnamens der überlagerten DSECT im Eingabefeld „symbolische Adresse“ innerhalb des DSECT-Formats geblättert wird. Ein eventuell mögliches automatisches Lokalisieren auf den Feldnamen der entsprechenden Systemtabelle ist ausgeschaltet.

Die Zuweisung einer DSECT zu einem Fenster wird gespeichert. Wenn man also die gleiche Struktur in einem andern Speicherbereich noch einmal lokalisiert hat (etwa bei verketteten Listen) und dem Diagnosefenster diesen Speicherbereich z.B. durch Markieren und Drücken der entsprechenden P-Taste zuweist, wird nicht auf Dump-Layout umgeschaltet, sondern der Speicher gleich im Layout der eingestellten DSECT symbolisch aufbereitet.

Dieser Effekt tritt allerdings auch dann ein, wenn man dem Fenster irgendeinen anderen Speicherbereich zuweist. Dieser Bereich wird dann im Layout der alten, vermutlich unpassenden DSECT aufbereitet. Man muss dann entweder die richtige DSECT eintragen oder durch Eingabe von D im Feld „Ausgabeformat“ der Kopfzeile auf Dump-Layout umschalten.

Überlagerung mit einer versetzten DSECT

Ist ein Speicherbereich nur im Layout eines Teilabschnittes einer DSECT strukturiert (z.B. die lokalen XVTs bei Multiprozessoren, siehe Bild 28 (Prozessor-lokale Bereiche ausgeben)), so ist eine versetzte Überlagerung notwendig. Hierzu ist die Eingabe des entsprechenden DSECT-Feldnamens im Eingabefeld „symbolische Adresse“ zusammen mit der Eingabe „CBM“ im Feld „Ausgabeformat“ erforderlich. Die Eingabe des Schlüsselwortes „CBM“ kann entfallen, wenn die DSECT nicht automatisch lokalisiert werden kann.

Damit können Sie sich auch das Rückwärtsblättern auf die Anfangsadresse einer Tabelle sparen. Ist der entsprechende Feldname des angezeigten Speicherbereichanfangs bekannt, so überlagert man den ausgegebenen Bereich wieder durch Angabe des Feldnamens.

Innerhalb der DSECT können Sie mit F1 / F3 und +/-/++/-- hin und her blättern, jedoch nur bis zum Beginn oder zum Ende der DSECT. Dabei bleibt das symbolische Layout erhalten.

Listen der angebbaren DSECTs für alle unterstützten BS2000-Versionen finden Sie ab "Liste der DSECTs aus den Standard-Symboldateien". Ferner können auch DSECTs aus privaten Symboldateien benutzt werden (Näheres siehe „Abschnitt „Private Symbolelemente verwenden"").

> ¡ Bei der symbolischen Ausgabe wird als Spalten-Trennzeichen X'4F' verwendet. Bei einigen Terminal-Einstellungen bzw. Zeichensätzen von Druckern wird dieses Zeichen nicht als senkrechter Strich „|“ abgebildet. Zur Umstellung wird der Benutzerparameter „Column separator (list)“ angeboten (siehe "Modifikationen durch den Benutzer (Spezialfenster OPTIONS)").

Überlagern mit der Pseudo-DSECT WORDLIST

Wenn ein Speicherbereich nicht im Layout einer überlagerungsfähigen DSECT beschrieben ist, aber Adressreferenzen zu Systembereichen enthält oder vermuten lässt, so können diese Adressen symbolisch mit Modulname + Distanz aufbereitet werden, und zwar mit der Pseudo-DSECT WORDLIST.

Für jedes Wort des Speicherbereiches wird dabei die Assembler-Deklaration `DS AL4` angenommen. Wenn der Wortinhalt dies formal zulässt, dann wird er als Modulname + Distanz ausgegeben.

So können Sie zum Beispiel den Konstantenbereich eines Moduls mit WORDLIST überlagern, um die verwendeten Extern-Adressen modulrelativ aufzubereiten.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


WORDLIST                        +00000=AFFEA4FC TID=000101E1          W4,CBM,L19
000 WORD000 : B13DF520 = DPSUSTA +00A0 | 004 WORD001 : B13EB7B8 = DPSUSTA@+B1F8
008 WORD002 : B13DF520 = DPSUSTA +00A0 | 00C WORD003 : AFFA0000
010 WORD004 : AFFEA3C0                 | 014 WORD005 : B13EBAC8 = DPSUSTA@+B508
018 WORD006 : B13E3010 = DPSUSTA@+2A50 | 01C WORD007 : AFFEFB6E
020 WORD008 : AFFEA720                 | 024 WORD009 : B1019140 = DKCMSPL@+3B80
028 WORD010 : B10124A0 = DKCMSPL +00A0 | 02C WORD011 : B0427D80 = NLKSYSPM+1F40
030 WORD012 : AFFEA378                 | 034 WORD013 : B1019C00 = DKCMSPL@+4640
038 WORD014 : B1021258 = DKCMSPL@+BC98 | 03C WORD015 : AFFEF006
040 WORD016 : B101A040 = DKCMSPL@+4A80 | 044 WORD017 : AFFEAF48
048 WORD018 : 00000003 =     3         | 04C WORD019 : 00000001 =     1
050 WORD020 : B03FE160 = NCTXVT  +2160 | 054 WORD021 : 00000004 =     4
058 WORD022 : AFFEA3C0                 | 05C WORD023 : B13E3064 = DPSUSTA@+2AA4
060 WORD024 : B13E2AC8 = DPSUSTA@+2508 | 064 WORD025 : AFFEFB6E
068 WORD026 : AFFEA720                 | 06C WORD027 : AFFEA720
070 WORD028 : B10128B0 = DKCMSPL +04B0 | 074 WORD029 : B10226E8 = DKCMSPL@+D128
078 WORD030 : B10124A0 = DKCMSPL +00A0 | 07C WORD031 : B0427D80 = NLKSYSPM+1F40
080 WORD032 : AFFEA378                 | 084 WORD033 : B1022A54 = DKCMSPL@+D494
088 WORD034 : B1020B48 = DKCMSPL@+B588 | 08C WORD035 : AFFEF006
CMD:
Key: 1=Help 2=Plk 3=PCB 4=WORDLI 5=Dump   6=Dump   7=Dump   8=Dump   9=Dump
```

Bild 31: Aufbereitung mit der DSECT WORDLIST; Anzeige in der Key-Zeile: WORDLIST

> Bei Klasse-6-Speicher werden nur Adressen aus dem Benutzerprogramm und angeschlossenen nicht-privilegierten Subsystemen relativiert. Will man in Ausnahmefällen auch Adressen aus dem Control Program (CP) und privilegierten Subsystemen relativiert erhalten, muss der Speicherbereich durch Eingabe von „ALT“ im Feld ASEL als Datenraum gekennzeichnet werden.

Behandlung von Substrukturen

In Kontrollblöcken, die in einer Hochsprache definiert worden sind, sind Substrukturen zulässig. Mit DAMP können derartige Substrukturen „aufgeklappt“ und „zugeklappt“ werden.

Bei der Ausgabe eines Kontrollblocks werden die Substrukturen zunächst ignoriert, d.h. die betroffenen Felder werden wie ein „ARRAY OF BYTE“ ausgegeben. Die Namen der Felder, die Substrukturen enthalten, werden durch ein vorangestelltes Zeichen „*“ markiert.
Wird dieses Zeichen mit einem „+“ überschrieben, wird die entsprechende Substruktur „aufgeklappt“, bei einem „-“ wird die Substruktur „zugeklappt“.

Über die Wahl der DUE-Taste (DUE1 oder DUE2) wird bestimmt, ob das Aufklappen „schön“ oder „komprimiert“ erfolgen soll.

- DUE1 „schönes“ Aufklappen. Pro Zeile wird nur ein Datenfeld der Substruktur unter dem übergeordneten Namen aufbereitet. Die Feldnamen werden eingerückt.
- DUE2 „komprimiertes“ Aufklappen. Aufbereitung der Substruktur wie ein normaler Kontrollblock. Die Substruktur wird lediglich an der entsprechenden Fensterposition eingefügt.

```
DAMP <version> USERDUMP(<ver>) from BS2000(<ver>)   <date> <time>


ESMFHDR                         +00000=0125AB64 TID=000400E4          W4,CBM,L19
000*IF_ID     : 00580201                | 004*RETURNCODE: 0E400009
```

Bild 32: Zugeklappte fiktive Substruktur

```
DAMP <version> USERDUMP(<ver>) from BS2000(<ver>)   <date> <time>


ESMFHDR                          +00000=0125AB64 TID=000400E4          W5,CBM,L19
000+IF_ID     :
  000 UNIT    : 0058     =    88
  002 FUNCTION: 02
  003 VERSION : 01
004+RETURNCODE:
  004+STRUCTURED_RC:
    004+SUBCODE       :
      004 SUBCODE2: 0E
      005 SUBCODE1: 40
    006 MAINCODE      : 0009
```

Bild 33: „Schön“ aufgeklappte Substruktur

```
DAMP <version> USERDUMP(<ver>) from BS2000(<ver>)   <date> <time>


ESMFHDR                          +00000=0125AB64 TID=000400E4          W7,CBM,L19
000+UNIT    : 0058     =    88          | 002 FUNCTION: 02
003 VERSION : 01                       | 004+SUBCODE2: 0E
005 SUBCODE1: 40                       | 006 MAINCODE       : 0009
```

Bild 34: „Komprimiert“ aufgeklappte Substruktur

5.3.2.9 Ausgabe im Assembler-Layout

Ein in einem Dumpfenster ausgegebener Speicherbereich kann disassembliert und im Assembler-Layout ausgegeben werden. Dabei werden Datensequenzen, die eindeutig nicht als Befehle interpretiert werden können, als DC-Konstanten gezeigt. Sollen Teile des disassemblierten Datenbereichs als DC-Konstanten angezeigt werden, dann sind die entsprechenden Zeilen in der Spalte des Opcodes zu markieren. Nach DUE wird dann die Disassemblierung dieser Befehle rückgängig gemacht und die entsprechenden Stellen werden als Konstanten ausgegeben.

Das Assembler-Format wird durch die Angabe „ASS“, „CAS“ oder „XAS“ im Eingabefeld „Ausgabeformat“ bestimmt. Bei der Eingabe „ASS“ bestimmt DAMP über die CSECT-Attribute automatisch, ob /390- oder x86-Befehle ausgegeben werden sollen. Sie können (z.B. bei fehlenden oder fehlerhaften CSECT-Attributen) das Format der Disassemblierung aber auch selbst bestimmen; bei der Eingabe „CAS“ werden /390-Befehle und bei „XAS“ werden x86-Befehle ausgegeben.

Bei der Ausgabe liefert DAMP im Feld „Ausgabeformat“ entweder „ASS“ (/390-Format) oder „XAS“ (x86-Format) zurück.
Im Ausgabeschirm sind Adresspegel und Opcodes markierbar, Opcodes aber nur bei /390-Objekten.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


ASTRACE                          +0004C=C0ADE70C SYS=00010001          W4,XAS,L19
C0ADE70C (0004C):448B5C2448         = mov         r11d,DWORD PTR [rsp+X'48'(R1)]
C0ADE711 (00051):90                 = nop
C0ADE712 (00052):458D5570           = lea         r10d,[R13+X'70']
C0ADE716 (00056):41C60226           = mov         BYTE PTR [r10],X'26'
C0ADE71A (0005A):41C6420101         = mov         BYTE PTR [r10+X'01'],X'01'
C0ADE71F (0005F):418BF3             = mov         esi,r11d
C0ADE722 (00062):0FCE               = bswap       esi
C0ADE724 (00064):41897204           = mov         DWORD PTR [r10+X'04'],esi
C0ADE728 (00068):44895C2420         = mov         DWORD PTR [rsp+X'20'(R11)],r11
                                   d
C0ADE72D (0006D):41BF24000000       = mov         R15d,X'00000024'
C0ADE733 (00073):6641C1CF08         = ror         R15w,X'08'
C0ADE738 (00078):6645897A02         = mov         WORD PTR [r10+X'02'],R15w
C0ADE73D (0007D):4489542448         = mov         DWORD PTR [rsp+X'48'(R1)],r10d
C0ADE742 (00082):41BF8C027FC0       = mov         R15d,X'C07F028C'
C0ADE748 (00088):4C0BBC2440010000   = or          R15,QWORD PTR [rsp+X'00000140'
                                   ]
C0ADE750 (00090):448D3503000000     = lea         R14d,[X'C0ADE75A']
CMD:
Key: 1=Help 2=Tsk 3=PCB 4=SC0ADE 5=Dump   6=Dump   7=Dump   8=Dump   9=Dump
```

Bild 35: Beispiel für disassemblierte Ausgabe von x86-Code

Mit der Anweisung `USE-REGISTER` (siehe "USE-REGISTER Registerverwendung für Disassemblierung festlegen") kann bei /390-Code, nicht jedoch bei x86-Code, eine symbolische Anzeige der Befehlsadressen gesteuert werden. Dadurch werden die Adressbefehle nicht mit Register und Distanz, sondern als modulrelative Adressen oder in Form eines Feldnamens aus der angegebenen DSECT dargestellt.

Das für die Disassemblierung verwendete /390-Befehlsformat wird von DAMP automatisch auf Grund der zu Grunde liegenden CPU-Serie gewählt. Diese Voreinstellung lässt sich mit der Anweisung `MODIFY-OBJECT-ASSUMPTIONS` ändern. Verfügbar sind die Befehlsformate 1 bis 5 (siehe Beschreibung `MODIFY-OBJECT-ASSUMPTIONS` auf "MODIFY-OBJECT-ASSUMPTIONS Ändern der Standardeinstellungen für das Diagnoseobjekt").

Die verwendeten `USE-REGISTER`-Anweisungen gelten für alle Disassemblierungen des gleichen Moduls, auch in mehreren Dumpfenstern. Durch die Anweisung `DROP-REGISTER` (siehe "DROP-REGISTER Registerverwendung für Disassembler festlegen") können diese Zuweisungen aufgehoben werden.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


ASTRACE                          +0047E=7151563E SYS=00010001          W5,ASS,L19
7151563E (047E):   D2 03 9008 B010   =   MVC   8(4,R9),WORD004
71515644 (0484):   D2 03 9014 B018   =   MVC   20(4,R9),WORD006
7151564A (048A):   D5 03 B01C A098   =   CLC   WORD007(4),LOC#04F0
71515650 (0490):   47 70 A05C        =   BNZ   LOC#04B4
71515654 (0494):   58 40 A09C        =   L     R4,LOC#04F4
71515658 (0498):   95 01 4A28        =   CLI   2600(R4),1
7151565C (049C):   47 70 A052        =   BNZ   LOC#04AA
71515660 (04A0):   D2 03 9018 A0A8   =   MVC   24(4,R9),LOC#0500
71515666 (04A6):   47 F0 A058        =   B     LOC#04B0
7151566A (04AA):   D2 03 9018 A0AC   =   MVC   24(4,R9),LOC#0504
71515670 (04B0):   47 F0 A062        =   B     LOC#04BA
71515674 (04B4):   D2 03 9018 B01C   =   MVC   24(4,R9),WORD007
```

Bild 36: Ausgabe im Assembler-Format; folgende Anweisungen wurden gegeben:

USE-REG MOD-NA=ASTRACE,REG=10,FOR=*MOD-BASE(DISPL=X'458')

USE-REG MOD-NA=ASTRACE,REG=11,FOR=*CONTR-BLOCK(NAME=WORDLIST)

Mit der Anweisung `ADD-LIST-OBJECTS WINDOW=<w>` können Sie den disassemblierten Speicherbereich für die Ausgabe auf Liste anmelden.

5.3.2.10 Ausgabe in real adressierte Bereiche

Real adressierte Bereiche sind in SLED- und SNAP-Dumps sowie in VM2000-Gesamtsleds nach Auswahl einer virtuellen Maschine enthalten.

Liegt die Anfangsadresse eines Speicherbereichs als Real-Adresse vor, so kann dieser Bereich ohne Adressumsetzung direkt auf ein Dumpfenster ausgegeben werden. Dazu tragen Sie in der Kopfzeile eines Dumpfensters in das Eingabefeld „ASEL“ das Schlüsselwort `RM` und in das Eingabefeld „Absolutadresse“ die gewünschte Realadresse ein.

DAMP stellt automatisch als 4GB-Segment „0“ ein, wenn das Feld „ASID“ nicht versorgt wird. Soll eine reale Adresse größer als 4GB ausgegeben werden, muss als „ASID“ das zugehörige 4GB-Segment und als „Absolutadresse“ die relative Distanz der Adresse zum Segmentanfang eingegeben werden.

Sie erhalten die Ausgabe im eingestellten Ausgabeformat. Sie können vorher und nachher aber auch auf die anderen Ausgabeformate umschalten.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


                              +00000=  001000 RM =00000000          W8.D  ,L19
001000 (0000) :  D7C9C401  00000000  00000000  C2E2F2F0   <==>  PID?????????BS20
001010 (0010) :  F0F04040  E5F1F94B  F0C1F0F0  C7F10000   <==>  00  V19.0A00G1??
001020 (0020) :  00000000  E7E5E3F4  FFFFFFFF  72AFA008   <==>  ????XVT4~~~~????
001030 (0030) :  7DD2C040  710011BC  FFFFFFFF  71001264   <==>  'K? ???\~~~~????
001040 (0040) :  713CE380  710011A4  7FA8BCF0  72C6A0E0   <==>  ??T????u"y\0?F??
001050 (0050) :  710011F8  71001648  FFFFFFFF  710011F8   <==>  ???8????~~~~???8
001060 (0060) :  FFFFFFFF  FFFFFFFF  FFFFFFFF  7FD5E000   <==>  ~~~~~~~~~~~~"N??
001070 (0070) :  71001160  7FAE23A8  FFFFFFFF  71294D40   <==>  ???-"??y~~~~??(
001080 (0080) :  FFFFFFFF  FFFFFFFF  FFFFFFFF  FFFFFFFF   <==>  ~~~~~~~~~~~~~~~~
001090 (0090) :  7D6E0A10  FFFFFFFF  7FD3C940  FFFFFFFF   <==>  '>??~~~~"LI ~~~~
0010A0 (00A0) :  7F295710  71001148  7FD1B000  FFFFFFFF   <==>  "???????"J??~~~~
0010B0 (00B0) :  7100114C  71001154  FFFFFFFF  FFFFFFFF   <==>  ???<????~~~~~~~~
0010C0 (00C0) :  FFFFFFFF  7FD3EF80  FFFFFFFF  FFFFFFFF   <==>  ~~~~"L??~~~~~~~~
0010D0 (00D0) :  FFFFFFFF  71F8F8E0  7F302000  7100161C   <==>  ~~~~?88?"???????
0010E0 (00E0) :  FFFFFFFF  FFFFFFFF  FFFFFFFF  71001278   <==>  ~~~~~~~~~~~~????
0010F0 (00F0) :  72925000  7289DC60  72899CC0  7DC20A60   <==>  ?k&??i?-?i??'B?-
001100 (0100) :  733E4D90  FFFFFFFF  FFFFFFFF  FFFFFFFF   <==>  ??(?~~~~~~~~~~~~
001110 (0110) :  710014E0  FFFFFFFF  00000000  00000000   <==>  ????~~~~????????
CMD:
Key: 1=Help 2=Tsk 3=PCB 4=S714CB 5=S71515 6=Dump   7=Dump   8=R-0001 9=Dump
```

Bild 37: Ausgabe ab realer Adresse 1000; Anzeige R-0001 in der Key-Zeile

Sinnvoll ist die Ausgabe in Real-Adressierung etwa in den Tabellen des Memory Management, bei der Analyse von CCW-Ketten, in festen Hardware-Bereichen (z.B. Save Areas) oder bei „Überschreibern“.

Ausgaben in Real-Adressierung erfolgen bei der Standard-Listaufbereitung nur für die standardmäßig vorgesehenen HW-Bereiche. Andere Bereiche können bei Bedarf mit `ADD-LIST-OBJECTS` zusätzlich für die Ausgabe auf Liste angemeldet werden.

5.3.2.11 Ausgabe in Absolut-Adressierung

Absolute Adressen liegen bei einem VM2000-Gesamtsled vor. Mit absoluten Adressen kann das gesamte VM2000-System adressiert werden.

Ein Bereich mit absoluten Adressen kann in einem Dumpfenster ausgegeben werden. Dazu tragen Sie in der Kopfzeile des Dumpfensters in das Eingabefeld „ASEL“ das Schlüsselwort ABS und in das Eingabefeld „Absolutadresse“ die gewünschte absolute Adresse ein.

Bei einer Absolutadresse größer als 4GB geben Sie im Feld „ASID“ das zugehörige 4GB-Segment ein und im Feld „Absolutadresse“ die relative Distanz der Adresse zum Segmentanfang.

Sie erhalten die Ausgabe im eingestellten Ausgabeformat. Vorher oder nachher kann ein anderes Ausgabeformat eingestellt werden.

```
DAMP <version> SLED(<ver>) from VM2000(<ver>)


                                +00000=  001000 ABS=00000000          W4,D  ,L19
001000 (0000) :  D7C9C401  00001024  00000000  E5D4F2F0   <==>  PID?????????VM20
001010 (0010) :  F0F04040  E5F1F14B  F0C1F1F0  F0F00000   <==>  00  V11.0A1000??
001020 (0020) :  00000000  00470101  FFFFFFFF  D7C9C401   <==>  ????????~~~~PID?
001030 (0030) :  00000000  00000000  E2E8E2E2  E3C1D9E3   <==>  ????????SYSSTART
001040 (0040) :  E5F1F94B  F0C1F0F0  C7F10601  00000000   <==>  V19.0A00G1??????
001050 (0050) :  00000000  00000010  E85CE3C5  E2E3C1D4   <==>  ????????Y*TESTAM
001060 (0060) :  C5D5E35C  00000000  00E00000  00E8C4D6   <==>  ENT*?????????YDO
001070 (0070) :  D4C1C9D5  00E00000  0A000000  01E8C4D6   <==>  MAIN?????????YDO
001080 (0080) :  D4C1C9D5  0AE00000  0A000000  02D5C4D6   <==>  MAIN?????????NDO
001090 (0090) :  D4C1C9D5  14E00000  23000000  03D5C4D6   <==>  MAIN?????????NDO
0010A0 (00A0) :  D4C1C9D5  80037E00  800AF000  04D5C4D6   <==>  MAIN??=???0??NDO
0010B0 (00B0) :  D4C1C9D5  00000000  00000000  D5E8C4D6   <==>  MAIN????????NYDO
0010C0 (00C0) :  D4C1C9D5  00000000  00000000  D5E8C4D6   <==>  MAIN????????NYDO
0010D0 (00D0) :  D4C1C9D5  00000000  00000000  D5E8C4D6   <==>  MAIN????????NYDO
0010E0 (00E0) :  D4C1C9D5  00000000  00000000  D5E8C4D6   <==>  MAIN????????NYDO
0010F0 (00F0) :  D4C1C9D5  00000000  00000000  D5E8C4D6   <==>  MAIN????????NYDO
001100 (0100) :  D4C1C9D5  00000000  00000000  D5E8C4D6   <==>  MAIN????????NYDO
001110 (0110) :  D4C1C9D5  00000000  00000000  D5E8C4D6   <==>  MAIN????????NYDO
CMD:
Key: 1=Help 2=Inf 3=PCB 4=A-0001 5=Dump   6=Dump   7=Dump   8=Dump   9=Dump
```

Bild 38: Ausgabe ab absoluter Adresse 1000

5.3.2.12 Ausgabe von Dumpfile-Sections

Dumpfile-Sections sind Abschnitte von Dumpdateien, deren Strukturen vom Erzeuger des Dumps (SLED, SNAP oder CDUMP) aufgebaut werden.

Ein Bereich einer Dumpfile-Section kann in einem Dumpfenster ausgegeben werden. Dazu tragen Sie in der Kopfzeile des Dumpfensters in das Eingabefeld „ASEL“ das Schlüsselwort „SCT“ und in das Eingabefeld „ASID“ den gewünschten Section-Namen ein.

Sie erhalten die Ausgabe im eingestellten Ausgabeformat. Sie können vorher und nachher aber auch auf die anderen Ausgabeformate umschalten.

DAMP unterstützt die Ausgabe von Dumpfile-Sections mit sequentieller, homogener oder gemischter Struktur, jedoch keine Dumpfile-Sections mit inhomogener Struktur.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


                              +00000=  001000 SCT=SLEDMEM           W7,D  ,L19
001000 (0000) :  10480100  01000000  00000000  C1C4E2F1   <==>              ADS1
001010 (0010) :  010ED000  010ED400  010EDC00  D4E2C7C2   <==>         M     MSGB
001020 (0020) :  E4C6C67A  01111000  E2C4E37A  01145000   <==>  UFF:    SDT:  &
001030 (0030) :  0116D500  0116D600  00000000  00000000   <==>    N   0
001040 (0040) :  00000000  00000000  00000000  00000000   <==>
001050 (0050) :  00000000  00000000  00000000  00000000   <==>
001060 (0060) :  00000000  00000000  00000000  00000000   <==>
001070 (0070) :  00000000  00000000  40D9C9C7  C8E3E240   <==>           RIGHTS
001080 (0080) :  D9C5E2C5  D9E5C5C4  D7D7D7D7  D7D7D7D7   <==>  RESERVEDPPPPPPPP
001090 (0090) :  010010A0  010013F0  00350000  00000000   <==>         0
0010A0 (00A0) :  01001000  D5E2C9C9  D7D3C8E6  F000F000   <==>       NSIIPLHW0 0
0010B0 (00B0) :  01010000  C5E3E2E5  D7404040  F002F700   <==>       ETSVP   0 7
0010C0 (00C0) :  0103F700  D5E2C9C4  C5D94040  F0006400   <==>     7 NSIDER  0
0010D0 (00D0) :  01045B00  D5E2C9C5  E7D4C7E3  F0003000   <==>     $ NSIEXMGT0
0010E0 (00E0) :  01048B00  D5E2C9C9  D7D3C3D6  F000A800   <==>       NSIIPLC00 y
0010F0 (00F0) :  01053300  D5E2C9C9  D7D3C4D4  F000F800   <==>       NSIIPLDM0 8
001100 (0100) :  01062B00  D5E2C9C9  D7D3C4E3  F0004000   <==>       NSIIPLDT0
001110 (0110) :  01066B00  D5E2C9C9  D7D3C9C8  F0006400   <==>     , NSIIPLIH0
CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=s-0001 8=Dump   9=Dump
```

Bild 39: Ausgabe der Dumpfile-Section SLEDMEM; Anzeige S-0001 in der Key-Zeile

5.3.2.13 Verkettungen verfolgen

Haben Sie in einem Dumpfenster den Beginn einer längeren Verkettung gefunden, die Sie verfolgen wollen, so empfiehlt sich folgendes Vorgehen:

- Festhalten des Verkettungsbeginns in einem Dumpfenster (z.B. W4) und Zuweisung zweier freier oder nicht mehr benötigter Dumpfenster (z.B. W6 und W8) mit der Anweisung
  `MODIFY-SCREEN-LAYOUT FIRST=6(SIZE=5),SECOND=8(SIZE=5),THIRD=4`
- Markieren der ersten Verkettungsadresse in W4 + DUE (drittes Fenster am Bildschirm). Im Fenster W6 erscheint die markierte Adresse.
- Abwechselndes Markieren der nächsten Verkettungsadresse in den beiden oberen Fenstern W6 und W8, bis die gewünschte Adresse erreicht wird.

Dieses Vorgehen hat den Vorteil, dass man bei Fehlbedienung jederzeit auf der letzten und auch auf der ersten Verkettungsadresse wieder aufsetzen kann.

Wollen Sie auf diese Sicherheit verzichten und brauchen Sie zur weiteren Bearbeitung nur einen bestimmten Speicherabschnitt in der Verkettungsreihe, so empfiehlt sich folgendes Vorgehen:

- Belegen des gesamten Bildschirms mit dem Fenster, in dem die erste Verkettungsadresse steht (z.B. W4) durch die Anweisung
  `MODIFY-SCREEN-LAYOUT FIRST=4(SIZE=19)`
- Markieren der Verkettungsadresse im (einzigen) Fenster W4, bis Sie zum gewünschten Speicherabschnitt kommen.

Damit erreichen Sie, dass nicht ungewollt das Fenster mit den Verkettungsadressen belegt wird, dessen alter Inhalt für die weitere Bearbeitung noch benötigt wird.

Günstig ist dabei, dass eine einmal für ein Fenster eingestellte DSECT gespeichert wird. Wenn Sie also nacheinander verschiedene Elemente der Kette auf das gleiche Fenster legen, werden die Bereiche sofort im symbolischen Layout der eingestellten DSECT aufbereitet.

5.3.2.14 System-Trace-Table ausgeben (Spezialfenster TRACE)

Mit der Anweisung `SHOW-EDITED-INFORMATION` können Sie sich die System-Trace-Table aufbereitet in ein bestimmtes Fenster ausgeben lassen.

```
SHOW-EDITED-INFORMATION INFORMATION=*TRACE-TABLE-EDIT, WINDOW=<w>
```

Nach dem Aufruf werden die Trace-Table-Einträge aller im Diagnoseobjekt enthaltenen Tasks ausgegeben. Das Eingabefeld „Taskauswahl“ kann mit einer <tid> oder mit ALL modifiziert werden. In das Feld „LM“ kann die Nummer einer logischen Maschine oder ALL eingegeben werden.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


SEQ# GP LM =ALL TID PS P_COUNTER      IDENTIFIER     (TID=ALL     )   W9,TRC,L19
   1 00 01          SI ETMSLMG +00058 CRSH NRTC516
  ?2 00 01          SI ETMSLMG +00058 PROG 58 Inv Opcd PCB=00000000 R13=00000000
   3 00 01          WT SYSTEM IDLE    I/O  0000   00C3 CSW=00929008 CSW=0C000000
   4 00 01 00010002 TP*ETMBON1 +001D6 SVC  FA $BOWT    LOC=F1BE43B8 BRS=7125C8C0
?  5 00 01 0001000E TP DQPAM   +01764 SVC  D7 $XCPW    PCB=71175918  R1=733C5028
?  6 00 01 00010008 TP*ETMBON1 +001D6 SVC  FA $BOWT    LOC=F1BEE230 BRS=7125C880
   7 00 01 0001000C TP*ETMBON1 +001D6 SVC  FA $BOWT    LOC=F12A86CE BRS=7125C740
   8 00 01 0001000C TP NBCCNTS +001F0 SVC  D5 $EXCP    PCB=71180720  R1=71FF9F00
   9 00 01 00010008 TP*ETMBON1 +001D6 SVC  FA $BOWT    LOC=F1BEE230 BRS=7125C880
? 10 00 01 000100C1 TP*ETMBON1 +001D6 SVC  FA $BOWT    LOC=FC688E82 BRS=73895B00
  11 00 01 000100C1 TP NEHMSTAT+0007E SVC  EA $FNDT    PCB=73286008  R1=70FFAA10
  12 00 01 00010075 TP*ETMBON3 +001EA SVC  FA $BOWT    LOC=FF512072 BRS=7125D440
? 13 00 01 00010075 TP EMMREQM1+004E0 SVC  F6 $UNMASK  LOC=F12188CC ATT=80EF0000
  14 00 01 00010075 TP DISTRIB +00F72 SVC  EA $FNDT    PCB=732861D8  R1=6EC83020

CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=TRAC   9=OPTS
```

Bild 40: Ausgabeformat der System-Trace-Table

Neben der Taskauswahl können Sie die Fensterlängen verändern und innerhalb eines Fensters blättern. Näheres hierzu siehe „Blättern in einem Diagnosefenster“ (Diagnosefenster verändern).

Die Einträge in „SEQ#“ sowie alle Felder, die Adressen enthalten, sind markierbar.
Mit der Sequenznummer wird die Adresse des System-Trace-Table-Eintrages markiert. Dies ist nützlich, wenn man sich den vollständigen Eintrag anschauen möchte.

In der GP- und der LM-Spalte der Traceliste werden die CPU-Nummer und die Nummer der logischen Maschine sedezimal angezeigt.

> **i** Standardmäßig werden die Trace-Einträge durch DAMP auf Grund des mitgeschriebenen Zeitstempels chronologisch zusammengemischt. Auf Wunsch kann jedoch durch Eingabe einer Nummer in das LM-Feld der Kopfzeile die Aufbereitung für eine bestimmte logische Maschine veranlasst werden. Es ist zu beachten, dass bei Auswahl einer logischen Maschine nicht gleichzeitig eine Task ausgewählt werden kann (und umgekehrt).

5.3.2.15 Ausgabe von Speicherattributen (Spezialfenster MEMATTR)

Die Aufbereitung der Speicherattribute erfolgt durch Eingabe der Anweisung

```
SHOW-EDITED-INFORMATION INFORMATION=*MEMORY-ATTRIBUTES, WINDOW=<w>
```

In diesem Spezialfenster wird die Tabelle mit den Attributen der allokierten virtuellen Speicherseiten ausgegeben. Außerdem können Sie sich die Speicherklassengrenzen anzeigen lassen.

Die Ausgabe kann sowohl für einen bestimmten Benutzeradressraum als auch auf Wunsch für den Systemadressraum veranlasst werden. Die Attribute können symbolisch oder im Original-Ausgabeformat der VAT (Virtual Attribute Table) dargestellt werden.

Um diese Möglichkeiten zu realisieren, gibt es in der Kopfzeile des Fensters die beiden Felder „ASN“ und „Layout-Feld“.
Im Feld „ASN“ kann eine TID oder der String „System“ eingetragen werden, um den Adressraum zu kennzeichnen.
Im Feld „Layout-Feld“ können folgende Kürzel angegeben werden:

- „LIM“ für die Speicherklassengrenzen
- „SYM“ für die symbolische Darstellung der virtuellen Attribute
- „HEX“ für die sedezimale Darstellung des ersten Bytes der virtuellen Attribute
- „HX2“ für die sedezimale Darstellung des zweiten Bytes der virtuellen Attribute

Bei der Ausgabe der Seitenattribute wird für jeweils 16 Seiten eine Zeile beschrieben, in der für jede Seite ein drei Zeichen langer Eintrag geschrieben wird. Dieser enthält entsprechend dem eingestellten Ausgabeformat entweder eine symbolische Darstellung der Attribute dieser Seite (Memory-Class, Partial-Page-Indicator, Privilege-Indicator) oder sedezimal aufbereitet das Original-Byte aus der VAT.

Drei Punkte (...) zeigen an, dass diese Seite nicht allokiert ist. Wenn die virtuellen Attribute nicht zugreifbar sind (Normalfall für Areadumps, aber auch bei anderen Dumptypen möglich), gibt (...) nur an, dass die Seite nicht in der Dumpdatei enthalten ist.

Ein Fragezeichen und zwei Punkte (?..) zeigen an, dass die Seite in der Dumpdatei enthalten ist, aber die virtuellen Attribute nicht zugreifbar sind.

Ein Stern (*) in der ersten Spalte einer Zeile bedeutet, dass sich seit der letzten Ausgabe eines Eintrags die Einträge der vorangehenden Seiten nicht geändert haben. Die Attribute für diese Seiten werden in der vorangehenden Zeile ganz rechts angezeigt.

Memory-Pool-Bereiche sind am rechten Rand durch den String „Pool“ gekennzeichnet.

Die Bedeutung der Symbole bei der SYM-Ausgabe bzw. der Bits bei der HEX- und HX2-Ausgabe ist in der Kopfzeile erläutert.

Bei der Ausgabe von Seitenattributen kann in der ersten Zeile im Feld „PAGE“ eine Seitennummer eingegeben werden. Die Aufbereitung beginnt dann bei der angegebenen Seite. Seitennummern (bis zu 8 Sedezimalziffern) liegen im Bereich X'0' bis X'1FE00000'.

```
DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>


The Symbols mean: <Class>,(P)artial-Pg,(N)onpriv-Pg       ASN=SYSTEM   W9,SYM,L19
 PAGE  :  0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F  Remarks
 00C0X : ShrBase  : 00C00000
 00C0X : ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...
*00F8X : ... ... ... ... ... ... ... ... ... ... ... ... 4P- 4P- ... 4--
*00FFX : 4-- 4-- 4-- 4-- 4-- 4-- 4-- 4-- 4-- 4-- 4-- 4-- 4-- 4-- ... ...
 7100X : C11Base  : 71000000
 7100X : 1-- 1-- 1-- 1-- 1-- 1-- 1-- 1-- 1-- 1-- 1-- 1-- 1-- 1-- 1-- 1--
*713FX : 1-- 1--
 713FX : C12Base  : 713F2000
 713FX :         2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2--
*71E0X : 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2-- 2--
 71E1X : C12Limit : 71E0FFFF
 71E1X : ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...
*71E6X : ... 3-- 3-- 3-- 3P- 3P- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3--
*71ECX : 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3P- 3P- 3-- 3-- 3-- 3-- 3-- 3--
*71EEX : 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3P- 3P- 3-- 3--
*71FBX : 3P- 3P- 3P- 3P- 3P- 3P- 3P- 3P- 3P- 3P- 3P- 3P- 3P- 3P- 3P- 3P-
 71FCX : 3P- 3P- 3P- 3P- 3P- 3P- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3-- 3--
CMD:
Key: 1=Help 2=Plk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=Dump   9=MEMA
```

Bild 41: Ausgabe von Speicherattributen

```
DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>


Class Limits for System / User Space :                  ASN=SYSTEM   W9,LIM,L19
 PAGE    Space                Start      End

 00C00 : Shared Code        = 00C00000 - 00FFFFFF

 71000 : Class 1            = 71000000 - 713F1FFF

 713F2 : Class 2            = 713F2000 - 71E0FFFF

 71E10 : Class 3/4          = 71E10000 - 7FFFEFFF
```

Bild 42: Ausgabe von Speicherklassengrenzen

5.3.2.16 Tabellen taskspezifischer Werte ausgeben (Spezialfenster TABLE)

Die Funktion TABLE ermöglicht die übersichtliche Ausgabe von taskspezifischen Werten aller im Diagnoseobjekt enthaltenen Tasks.

Nach dem Aufruf

```
SHOW-EDITED-INFORMATION INFORMATION=*TASK-TABLES, WINDOW=<w>
```

wird das gewünschte Dumpfenster W4 - W9 bzw. W21 - W99 als TABLE-Fenster an erster Stelle in der aktuellen Fensterlänge, aber noch ohne Ausgabewerte in das Diagnosefeld eingeblendet.

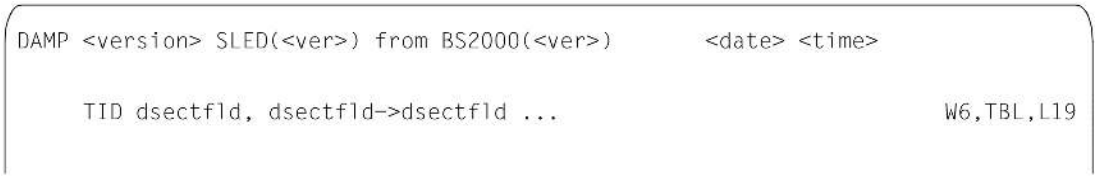


Bild 43: Dumpfenster nach Aufruf der Funktion TABLE

Anschließend können Sie in die Kopfzeile eine Liste von Feldnamen der taskspezifischen DSECTs ETCB, EJCB und EVUMT, getrennt durch Leerzeichen, eingeben. Die nachfolgende Ausgabe besteht dann in je einer Zeile pro Task mit den Inhalten dieser Felder.

Sollen taskspezifische Datenfelder aus weiteren Kontrollblöcken dargestellt werden, muss der Weg zu diesem Datenfeld eindeutig beschrieben werden. In die Kopfzeile des Fensters trägt man eine Kette von Feldnamen in der folgenden Form ein:

feldname1 -> feldname2 -> feldname3 -> ... -> feldnamex

Der Feldname feldname1 muss aus einer der oben genannten automatisch lokalisierbaren DSECTs stammen und auf den Anfang der Struktur zeigen, die feldname2 enthält. feldname2 zeigt wiederum auf den Anfang der Struktur, die feldname3 enthält u.s.w. Das gewünschte Feld feldnamex steht am Ende der Kette.

Enthält einer der angegebenen Feldnamen den Wert 0 oder ist die bezeichnete Adresse nicht allokiert, so bricht DAMP die Auflösung der Feldnamen-Kette ab.

Will man beispielsweise untersuchen, in welcher Task mehr als drei Dateien geöffnet sind, so kann folgende Konstruktion verwendet werden:

ETCBTFT -> IDMFRLNK -> IDMFRLNK -> IDMFRLNK -> IDMFRLNK

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


     TID ETCBTSN  ETCBTFT->IDMFRLNK->IDMFRLNK->IDMFRLNK->IDMFRLNK     W6,TBL,L19
000100DD    FTCP 70F9A000->70F9A0A8->70F9A150->70F9A1F8->00000000
000100DE    OS91 70F44000->
000900DF    XAE8 00000000->
000900E0    XAEY 00000000->
000900E1    XAE9 00000000->
000D00E2    OTTK 70F9A000->
000900E3    PR03 70F60000->00000000->
000100E4    PR1C 70F60000->00000000->
000D00E5    OTTB 70F9A000->70F9A0A8->70F9A150->70F9A1F8->70F9A2A0
000100E6    XAD7 00000000->
000D00E7    OTTQ 70F9A000->70F9A0A8->70F9A150->70F9A1F8->70F9A2A0
000400E8    OTR6 00000000->
000E00E9    OTTT 70F9A000->
000A00EA    OTRM 00000000->
000200EB    OTVE 70F9A000->70F9A0A8->70F9A150->70F9A1F8->70F9A2A0
000200EC    OTVF 70F9A000->70F9A0A8->70F9A150->70F9A1F8->70F9A2A0
000900ED    OTRT 00000000->
000200EE    OTT8 70F9A000->70F9A0A8->70F9A150->70F9A1F8->70F9A2A0
CMD:
Key: 1=Help 2=Plk 3=PCB 4=ETCB   5=ESTK   6=Dump   7=Dump   8=TABL   9=Dump
```

Bild 44: Ausgabe von Verkettungen, Angaben in der Eingabezeile:

ETCBTSN(C) ETCBTFT->IDMFRLNK->IDMFRLNK->IDMFRLNK->IDMFRLNK

Die TSN wird zusätzlich abdruckbar ausgegeben

Das Ausgabeformat der Felder entspricht ihrer Definition in der DSECT. Es kann jedoch durch explizite Angabe eines Formatzeichens in Klammern hinter dem Feldnamen geändert werden. Akzeptiert werden die Formatzeichen

- C Darstellung im Zeichenformat,
- X Darstellung im Sedezimal-Format und
- I Darstellung als Integer-Zahl.

Mit den Eingaben F3/+, ++, F1/- , --, +n und -n kann geblättert werden. Näheres hierzu siehe „Blättern in einem Diagnosefenster" (Diagnosefenster verändern).

5.3.2.17 Informationen über Subsysteme ausgeben (Spezialfenster SUSY)

Mit der Funktion SUSY kann sich der DAMP-Anwender folgende Informationen ausgeben lassen:

- über den BS2000-Nucleus (Control Program, CP)
- über alle Subsysteme, die mit DSSM geladen wurden
- über alle geladenen Benutzerprogramm-Kontexte

Das Control Program und die Benutzerprogramm-Kontexte erhalten von DAMP Pseudo-Subsystemnamen. Das Control Program erhält den Subsystemnamen 'CP' und alle Benutzer-Kontexte den Subsystemnamen 'USERPROG'. Gibt es mehrere geladene Benutzer-Kontexte in einer Task, so ist über eine von DAMP intern vergebene 'USERPROG'- Versionsnummer CTXNRxxx ('xxx': laufende Nr. 001, 002, ...) die Eindeutigkeit der Subsystembezeichnung gewährleistet.

In Userdumps sind Informationen über folgende Subsysteme enthalten:

- alle geladenen Benutzer-Kontexte,
- alle an die Task angeschlossenen nicht privilegierten Subsysteme.

In Areadumps fehlen diese Informationen in der Regel.

Bei Systemdumps, SLEDs, SNAPs und beim aktiven System werden Informationen über folgende Subsysteme ausgegeben:

- Control Program
- alle geladenen privilegierten und nicht privilegierten Subsysteme

Der Aufruf lautet:

```
SHOW-EDITED-INFORMATION INFORMATION=*SUBSYSTEM-INFORMATION, WINDOW=<w>
```

Zum Layout des erzeugten Fensters und zur Erläuterung der Felder siehe "Informationen über Subsysteme ausgeben (Spezialfenster SUSY)".

Im SUSY-Fenster kann mit F3/+, ++, F1/ -, --, +n und -n geblättert werden. Näheres hierzu siehe „Blättern in einem Diagnosefenster" (Diagnosefenster verändern).

> ENTRY-Namen werden im SUSY-Fenster nicht unterstützt.

Die Kopfzeilen des SUSY-Fensters

In den Kopfzeilen des SUSY-Fensters befinden sich mehrere Eingabefelder, in die Sie die auszugebenden Subsysteme, Holder-Tasks oder CSECTs eintragen können.

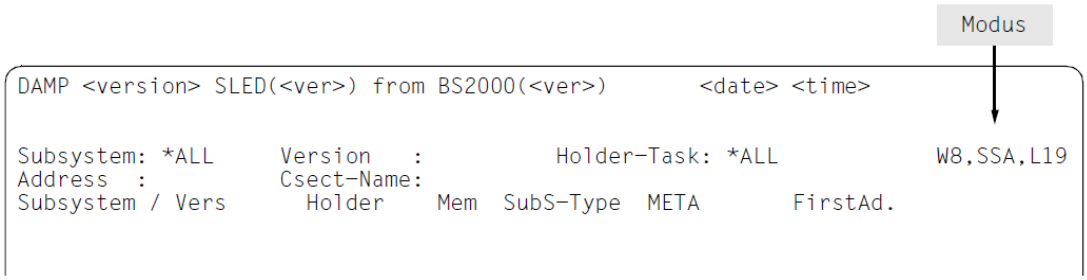


Bild 45: Die Kopfzeilen des SUSY-Fensters

Eingabemöglichkeiten in die Kopfzeilen des SUSY-Fensters

Es ist eine kombinierte Eingabe in die verschiedenen Felder möglich, d.h. die in vorhergehenden Arbeitsschritten erfolgten und immer noch angezeigten Einträge in die Eingabefelder behalten ihre Gültigkeit und zusätzlich können neue Einträge vorgenommen werden.

- Feld **Subsystem**

  Wird in kein anderes Feld etwas eingegeben, so erhält man die Liste der CSECTs dieses Subsystems.
  Die Eingabe von *USER entspricht der Eingabe von USERPROG.
  Wird zusätzlich ein CSECT-Name oder eine Adresse eingegeben, so wird ausschließlich danach in dem angegebenen Subsystem gesucht.
  Wird Subsystem=*ALL und z.B. ein CSECT-Name eingegeben, so wird dieser in allen Subsystemen gesucht.
  Wird nur Subsystem=*ALL eingegeben, so wird auf die Subsystem-Liste umgeschaltet, wobei ein eventueller Eintrag im Feld „Holder-Task“ einschränkend berücksichtigt wird.Durch Markieren des Subsystem-Namens in einer der Ausgabezeilen wird die entsprechende CSECT-Liste ausgegeben.

  Die Kontext-Namen werden im Übersichtsfenster alternativ zu den sonstigen Informationen über die Subsysteme ausgegeben. Dazu muss man entweder die Version eines Subsystems markieren oder „CTX“ im Modusfeld eingeben.

- Feld **Version**

  In dieses Feld kann die Version des gewünschten Subsystems eingegeben werden.

- Feld **Holder-Task**

  Tragen Sie in das Feld „Holder-Task“ eine TID ein, so wird die Liste der Subsysteme auf diejenigen verkürzt, deren Holder-Task mit der angegebenen TID übereinstimmt. Den gleichen Effekt erreicht man durch Markieren der unter „Holder“ angeführten TID in einer der Ausgabe-Zeilen.

  Gleichzeitig wird versucht, die aktive Task zu wechseln (die mit der angegebenen TID wird ggf. aktiviert). Ein Nebeneffekt kann dann das Erscheinen oder Verschwinden eines Benutzerprogramms am Ende der Subsystemliste werden.
  Die ursprüngliche Übersicht erscheint wieder, wenn in dieses Feld *ALL oder in das Feld „Modus“ SSA eingetragen wird.

- Feld **Csect-Name**

  Wird in das Feld ein CSECT-Name eingetragen, so wechselt das Format, und es wird eine Liste aller Subsysteme ausgegeben, in denen diese CSECT vorkommt.

  Die Subsystemliste erhält man zurück, wenn man in das Feld „Subsystem“ *ALL einträgt.

  Diese Funktion ist auch für Module des CP erlaubt, etwa um sich deren Identifikationsfeld (ETPND) ausgeben zu lassen.

- Feld **Address**

  In das Feld „Address“ können Sie eine Adresse eintragen, die lokalisiert, d.h. in Modul + Distanz umgerechnet werden soll. Die Ausgabe hat das Format der CSECT-Liste und enthält alle Module, in denen diese Adresse liegen könnte.

- Feld **Modus**

  Dieses Feld ist ein kombiniertes Ein-/Ausgabefeld und hat folgende Bedeutung:

  SSA Eine Liste aller Subsysteme wird ausgegeben.

  SSC Es werden die mit der aktuellen Task verbundenen Subsysteme angezeigt.

  SSH Eine Liste aller Subsysteme, deren Holder-Task mit der angegebenen TID übereinstimmt, wird ausgegeben (nur Ausgabefeld).

  CTX Für alle Subsysteme wird der Kontext-Name ausgegeben.

  INF CTX wird rückgängig gemacht.

  CS2 Das Layout der CSECT-Liste wird umgeschaltet. Statt des ETPND wird für jede CSECT der PMode und das HSI-Byte ausgegeben. DieserModus wird nur für Diagnoseobjekte von x86-Servern unterstützt.

  CS1 oder CSE CS2 wird rückgängig gemacht.

  EDT Die den aktuellen Einstellungen des SUSY-Fensters entsprechenden Informationen werden in der gesamten Länge in den aktuellen EDT-Bereich ausgegeben.

  LST Die gesamten Subsystem- und CSECT-Informationen des Objekts werden in den aktuellen EDT-Bereich ausgegeben.

  Aktuelle Task ist (im SLED oder SNAP) diejenige Task, die vom Diagnostiker zuletzt ausgewählt wurde (durch Markierung im Fenster W2 oder durch explizite Eingabe einer TID oder TSN etc.).

Layout der Subsystem-Liste

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


Subsystem: *ALL     Version   :          Holder-Task: *ALL            W8,SSA,L19
Address  :          Csect-Name:
Subsystem / Vers      Holder    Mem  SubS-Type  META       FirstAd.
CP        / <ver>     00000000  SYS  Nuc                   71000000
AIDSYSA   / <ver>     0001001E  SYS  Priv       7DDF6000   73882000
ANITA     / <ver>     0001001E  SYS  Priv       79B86000   79B94000
ASTI      / <ver>     0001001E  SYS  Priv       7D7CA000   7D7F8000
BCAM      / <ver>     0001001E  SYS  Priv       7F423000   73411000
BCAM-SM2  / <ver>     0001001E  SYS  Priv       7AD95000   79D1F000
BLSSERV   / <ver>     0001001E  SYS  Priv       7F629000   7F537000
BLSSYS    / <ver>     0001001E  SYS  Priv       7FAC9000   7FABA000
CALENDAR  / <ver>     0001001E  SYS  Priv       7FAE8000   7F82B000
CAPRI     / <ver>     0001001E  SYS  Priv       7D553000   7D43C000
CCOPY     / <ver>     0001001E  SYS  Priv       7BE67000   7C786000
CMX-TP    / <ver>     0001001E  SYS  Priv       7BE69000   7CDAD000
COSMOS    / <ver>     000200FF  SYS  Priv       79BE7000   76971000
CPR       / <ver>     0001001E  SYS  Priv       7D568000   7CD94000
DAB       / <ver>     00010069  SYS  Priv       7D458000   7389E000
DCAM      / <ver>     0001001E  SYS  Priv       7D27A000   7D3F1000
CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=SUSY   9=Dump
```

Bild 46: Übersicht über Subsysteme

Die einzelnen Spalten der ausgegebenen Subsystem-Liste im Bild 46 haben folgende Bedeutung:

- Überschrift **Subsystem**
  Name des Subsystems.
  Wird in dieser Spalte ein Feld markiert, schaltet die Ausgabe auf das Layout der CSECT-Liste für dieses Subsystem um.
- Überschrift **Vers**
  Version des Subsystems.
  Wird in dieser Spalte ein Feld markiert, wird auf die Ausgabe des Kontext-Namens für das markierte Subsystem umgeschaltet.
- Überschrift **Holder**
  TID der Holder-Task (wenn vorhanden).
  Wird in dieser Spalte ein Feld markiert, erscheint eine verkürzte Subsystem-Liste mit allen Subsystemen, die von dieser Task „gehalten“ werden.
- Überschrift **Mem**
  Speicherbereich, in den das Subsystem geladen wurde. Es wird „SYS“ für Systemspeicher oder „USR“ für Benutzerspeicher ausgegeben.
- Überschrift **SubS-Type**
  Art des Subsystems.
  In dieser Spalte können die Bezeichnungen Nuc, Priv, NPriv, TU, Usr-Ctx, Pool-Ctx, Task-Loc und undefined stehen.
- Überschrift **META**
  Adresse der Metadaten der Zugriffsmethode ANITA.
  Wird in dieser Spalte ein Feld markiert und anschließend z.B. auf eine der Tasten P4 - P9 gedrückt, erscheinen in dem gewünschten Fenster die ANITA-Metadaten für dieses Subsystem. Dieser Bereich enthält eine Liste von Ladeinformationen, die für das Subsystem aufgebaut wurden.
  Für das Subsystem CP ist dieses Feld leer.
  Für Subsysteme, die vor DSSM geladen wurden, gibt es ebenfalls keine Anzeige.
- Überschrift **First Ad.**
  Anfangsadresse des Subsystems.
  Dieses Feld enthält die niedrigste Adresse der gefundenen CSECTs des Subsystems. Sind keine Ladeinformationen zu dem Subsystem vorhanden, ist die Anfangsadresse nicht versorgt.

Layout der CSECT-Liste

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


Subsystem: AIDSYSA  Version   : 19.0     Holder-Task: 0001001D        W8,CSE,L19
Address   :         Csect-Name:
Subsystem/Version Address  = Module   + Reladd  Length    ETPND-Info
                  73199000 = ASASHS   +      0  00001E00  ASASHS   897A 20080730
                  7EEF0000 = ASAIDENT +      0  00000600  ASAIDENT 866A 20080730
                  7EEF0600 = ASYTPRV  +      0  00000EC0  ASYTPRV  810A 20080730
                  7EEF14C0 = ASAFNAT  +      0  00000980  ASAFNAT  890A 20080730
                  7EEF1E40 = ASACMD   +      0  00000980  ASACMD   8901 20080730
                  7EEF27C0 = ASACMD@@ +      0  00001B40  ASACMD@@ 8901 20080730
                  7EEF4300 = ASASSD   +      0  00000080  ASASSD   8901 20080730
                  7EEF4380 = ASAENAT  +      0  00000240  ASAENAT  8991 20080730
                  7EEF45C0 = ASAENAT@ +      0  000017C0  ASAENAT@ 8991 20080730
                  7EEF5D80 = ASAEVT   +      0  000002C0  ASAEVT   8911 20080730
                  7EEF6040 = ASAEVT@@ +      0  00002A00  ASAEVT@@ 8911 20080730
                  7EEF8A40 = ASAUTIL  +      0  00000900  ASAUTIL  8971 20080730
                  7EEF9340 = ASAUTIL@ +      0  00001900  ASAUTIL@ 8971 20080730
                  7EEFAC40 = ASASHC   +      0  00000900  ASASHC   8971 20080730
                  7EEFB540 = ASASHC@@ +      0  0000AE80  ASASHC@@ 8971 20080730
                  7EF063C0 = ASASH2   +      0  00000880  ASASH2   8971 20080730
CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=SUSY   9=Dump
```

Bild 47: Liste der CSECTs des Subsystems AIDSYSA

Die einzelnen Spalten der CSECT-Liste im Bild 47 haben folgenden Inhalt:

- Überschrift **Subsystem**
  enthält den Namen des Subsystems nur dann, wenn CSECTs aus verschiedenen Subsystemen angezeigt werden. Ansonsten ist dieses Feld leer und es gilt der Name des in der Kopfzeile angezeigten Subsystems.
- Überschrift **Address**
  enthält die Adresse der CSECT oder eine Adresse innerhalb der CSECT. Letzteres ist nur dann der Fall, wenn die Liste durch eine Eingabe im Feld „Address“ erzeugt wurde. Dieses Feld ist markierbar. Wird anschließend z. B. auf eine der Tasten P4 - P9 gedrückt, dann wird das gewählte Fenster auf die markierte Adresse positioniert.
- Überschrift **Module** und **Reladd**
  enthält die Adresse aus der Spalte „Address“ modulrelativ umgerechnet.
- Überschrift **Length**
  enthält die Modul-Länge.
- Überschrift **ETPND-Info**
  enthält die ETPND-Information mit Modulname, Versionsnummer und Übersetzungsdatum.

  Die Länge des Moduls kann null sein, wenn es sich um einen Großmodul handelt. Die ETPND-Information kann in System-, User- oder Areadumps fehlen, wenn die zugehörige virtuelle Seite nicht im Diagnoseobjekt enthalten ist, weil der Dumperzeuger nur die referenzierten Coding-Seiten abspeichert.

  Im Standard-Fenster können CSECTs aller Subsysteme beim Lokalisieren von Speicherbereichen angegeben werden. Auch die automatische Relativierung von Adressfeldern berücksichtigt die CSECTs aller Subsysteme. Zum Lokalisieren von CSECTs ist SUSY also nicht mehr erforderlich. Dies gilt jedoch nur, solange die CSECT-Namen im Bereich aller Subsysteme eindeutig sind. Ist die Eindeutigkeit nicht gegeben (z.B. in den Subsystemen NKVT, NKVD oder bei Koexistenz verschiedener Versionen des gleichen Subsystems), wird bei diesem Verfahren stets der erste Treffer angezeigt. Eine gezielte Lokalisierung ist dann nur über SUSY oder die Anweisung SEARCH-IN-SUBSYSTEM (siehe "SEARCH-IN-SUBSYSTEM CSECT-Suche in einem Subsystem") möglich.

- Überschrift **PMODE/HSI**
  enthält bei Diagnoseobjekten von x86-Servern Informationen über den Prozessormodus, in dem das Coding abläuft, sowie die Art der Codegenerierung. Diese Ausgabe erfolgt beim Modus CS2.

  Die möglichen Ausgaben haben folgende Bedeutung:

PMODE	**Bedeutung**
CISC	emuliert im /390-Modus
X86	native auf einem x86-Server
HSI	**Bedeutung**
R	mixed binary_no
U	mixed binary_yes
00	BLS-Information ist nicht verfügbar bzw. obsolet. Bei CISC-Coding wird generell dieser Wert angezeigt.

In den Übersichts-Fenstern von SUSY kann man mit den üblichen Tasten bzw. Eingaben blättern. Dabei werden die Blätterfunktionen +/F3, ++, - / F1, --, +n, -n unterstützt.

> ¡ Es gibt Klasse-5-Subsysteme, die in der Holder-Task und in den angeschlossenen Nutzer-Tasks jeweils einen anderen Adressraumstreifen belegen. Die von DAMP zur Lokalisierung benutzten Informationen enthalten dabei jeweils die Adressen, die in einer Nutzer-Task gültig sind. In der Holder-Task findet man jedoch an diesen Stellen nicht die erwarteten Module.

5.3.2.18 Informationen über Systemdateien sowie Sections der Dumpdatei (Spezialfenster FILE)

Die Funktion FILE dient zur Übersicht, Anzeige, Listen-Ausgabe und Generierung von in Dump-Dateien enthaltenen (gesicherten) Systemdateien.

Zusätzlich erhält man eine Übersicht aller nicht leeren Abschnitte (sog. Sections) der Dumpdatei. Alle Systemdateien sind als Sections in der Dumpdatei abgelegt.

Nach Eingabe von

```
SHOW-EDITED-INFORMATION INFORMATION=*DUMPED-SYSTEM-FILE, WINDOW=<w>
```

wird das gewünschte Dumpfenster W4 - W9 bzw. W21 - W99 als oberstes Fenster in der aktuellen Fensterlänge in Form eines Übersichts- und Auswahlschirms im Modus INF ausgegeben. In diesem Schirm sind alle in der Dumpdatei enthaltenen Sections aufgelistet. Es kann mit den üblichen Scroll-Kommandos geblättert werden.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


Section   Available Modi    * Overview and Selection Screen *         W8,INF,L19
          Mark mode of wanted section to select  - or enter right above
          "LST" to list this overview, "ALL" for more, "INF" for compact info.
SLEDLOG   DSP  EDT  LST  GEN
          Loggingfile of dump generator
CONSLOG   DSP  EDT  LST  GEN
          Dumped Systemfile  $SYSAUDIT.SYS.CONSLOG.<date>.075.001
EQUISAMQ  GEN
          Dumped Systemfile  $TSOS.EQUISAMQ
HELFILE   GEN
          Dumped Systemfile  $TSOS.SYS.HEL.<date>.065621
MSCFTRAC  GEN
          Dumped Systemfile  $TSOS.SYS.MSCF-TRACE.<date>.166.075.001
REPLOG    DSP  EDT  LST  GEN
          Dumped Systemfile  $SYSAUDIT.SYS.REPLOG.<date>.075.01
SERSLOG   GEN
          Dumped Systemfile  $TSOS.SYS.SERSLOG.<date>.075.01
SJMSFILE  GEN
          Dumped Systemfile  $TSOS.SJMSFILE.WORK
CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=FILE   9=OPTS
```

Bild 48: Layout des erzeugten Übersichts- und Auswahlschirms

Im Modus-Feld sind folgende Eingaben möglich:

ALL Umschaltung in eine detaillierte Übersicht. Sie enthält eine Aufbereitung der gesicherten Katalogeinträge der enthaltenen Systemdateien.

LST Ausgabe der detaillierten Übersicht nach *SYSLST.

INF Zurückschalten in den kompakten Übersichts- und Auswahl-Modus.

Direkt neben dem Sectionnamen werden die für diese Section verfügbaren Bearbeitungsmodi und eine Kurzbeschreibung der Section ausgegeben. Durch Markieren eines Bearbeitungsmodus wird die zugehörige Section ausgewählt und bearbeitet.

Die markierbaren Bearbeitungsmodi sind:

DSP	Inhalt der Section im Diagnosefenster anzeigen.
LST	Inhalt der Section nach *SYSLST ausgeben.
EDT	EDT aufrufen und den Inhalt der Section in einen EDT-Bereich einlesen.
GEN	Aus der gesicherten Systemdatei eine Datei erzeugen.

> *Hinweis zur Generierung von Systemdateien*
>
> DAMP führt keine automatische Konvertierung der Dateiformate durch!
> Es muss z.B. für eine Datei mit PAMKEYs eine dieses Format unterstützende Platte zugreifbar sein. Der Hinweis ist nicht gültig für die Generierung von REPLOG, CONSLOG oder SLEDLOG.
> Mit ADD-FILE-LINK und dem Sectionnamen als Linknamen können für die Generierung Datenträger, Name usw. vereinbart werden. Existiert kein Linkname, generiert DAMP die Datei unter der Aufruferkennung mit einem von DAMP automatisch vergebenen Namen.

Wurde ein Modus für eine Section markiert, für die auch der Modus DSP verfügbar ist, so ändert sich nach Ausführung der angestoßenen Bearbeitung (z.B. nach Rückkehr aus dem EDT) das Layout des Fensters zum „Layout bei ausgewählter Section“. In diesem „Layout bei ausgewählter Section“ bestehen in der Kopfzeile folgende Eingabemöglichkeiten:

Eingabemöglichkeiten in der Kopf-Zeile bei ausgewählter Section

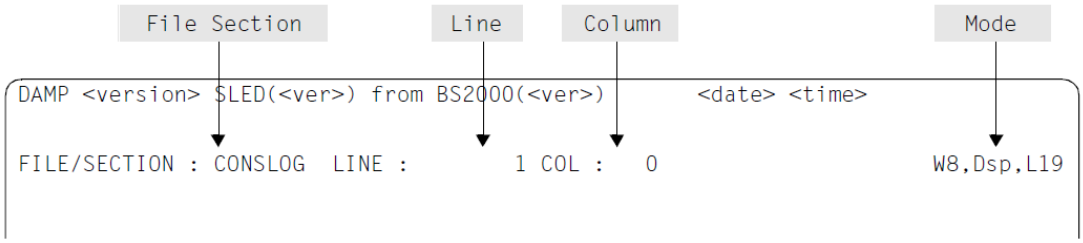


Bild 49: Eingabemöglichkeiten in die Kopfzeile des FILE-Fensters bei ausgewählter Section

- Feld **FILE/SECTION**
  hier wird der Name der aktuell ausgewählten Section ausgegeben und der Name einer anderen Section kann eingegeben werden.
- Feld **Line**
  hier kann die Nummer der ersten anzuzeigenden Zeile eingetragen werden.
- Feld Column
  hier kann die Nummer der ersten anzuzeigenden Spalte eingetragen werden.
- Feld **Mode**
  es gibt die folgenden sechs Modi:

Dsp	Informationen im Fenster anzeigen (Voreinstellung)
Edt	EDT aufrufen und Informationen in den aktuellen EDT-Arbeitsbereich übernehmen
Gen	Generieren als BS2000-Datei
Lst	Informationen nach SYSLST ausgeben
Inf	Zurückschalten in den Übersichts- und Auswahlschirm
All	Zurückschalten in den Modus ALL des Übersichts- und Auswahlschirms

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


FILE/SECTION : REPLOG   LINE :      140 COL :   0                      W8,Dsp,L19
 REP 001F4 103 X'92A106B2A2E1001826F6000892F80016' 0000 FA0434308692 2#1DPSIMPO
 REP 00204 103 X'37180001A2F8001626590180032000008' 0000 0A0434308692 2#1DPSIMPO
 REP 00214 103 X'00000000'                         0000 FA0434308692 2#1DPSIMPO
 REP 004FC 112 X'24090006000E682591B800003404007A' 26AD FA0434354251 2#1NASDSPP@
 REP 0050C 112 X'13040015000000009 1B800003404004B' 25A4 BA0434354251 2#1NASDSPP@
 REP 0051C 112 X'1304001100000000025AD00012529FFFF' 0320 EA0434354251 2#1NASDSPP@
 REP 0052C 112 X'1520FFF5000000000129482610000000A' 2405 3A0434354251 2#1NASDSPP@
 REP 0053C 112 X'00000000'                         25A6 3A0434354251 2#1NASDSPP@
 REP 0458C 114 X'1000'                             1704 EA0434434307 2#1NSPSERV@
 REP 02DC8 142 X'92D80014330400081080003700000000' 92D8 DA0434520058 2#1DOCPFA@@
 REP 02DD8 142 X'92D80010340400011704000300000000' 92D8 0A0434520058 2#1DOCPFA@@
 REP 02DE8 142 X'1000000C00000000'                 1000 3A0434520058 2#1DOCPFA@@
 REP 02E28 142 X'8EB700801000001E00000000'         2405 DA0434520058 2#1DOCPFA@@
 REP 0303C 142 X'8EB7016C16E0010F00000000'         8EB7 2A0434520058 2#1DOCPFA@@
 REP 03480 142 X'8EB700801000FF0800000000'         0000 EA0434520058 2#1DOCPFA@@
 REP 07B2C 119 X'267901100320000800000000'         8EF6 AA0434806307 2#1DPSCMGR@
 REP 001B0 119 X'8EF6000412C000040000000026590F00' 0000 3A0434806307 2#1DPSCMGR
 REP 001C0 119 X'0320000800000008E570FE002E0B009' 0000 DA0434806307 2#1DPSCMGR
CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=FILE   9=OPTS
```

Bild 50: Layout des FILE-Fensters nach Auswahl der Systemdatei REPLOG ab Zeile 140

5.3.2.19 Informationen über AUDIT-Tabellen (Spezialfenster AUDIT)

Die Funktion AUDIT dient zur Anzeige der in einem Diagnoseobjekt enthaltenen AUDIT-Tabellen (Hardware- und Linkage-AUDIT).

Nach Eingabe von

```
SHOW-EDITED-INFORMATION INFORMATION=*AUDIT-TABLE-EDIT, WINDOW=<W>
```

wird das gewünschte Dumpfenster W4 - W9 bzw. W21 - W99 als erstes Fenster in der aktuellen Fensterlänge angezeigt. Sind im Dump AUDIT-Tabellen enthalten, wird eine davon ausgewählt und angezeigt. Die Kopfzeile des Fensters enthält mehrere Eingabefelder, mithilfe derer man die verschiedenen AUDIT-Tabellen-Arten und -Bereiche sowie die gewünschte Task auswählen kann.

Neben dem Hardware-AUDIT gibt es den Linkage-AUDIT in den Ausprägungen prozessorlokal (SIH oder SIH+TPR) und tasklokal (TPR und TU). Die Auswahl der AUDIT-Tabellen durch DAMP nach der Anweisung `SHOW-EDITED-INFORMATION INFORMATION=AUDIT-TABLE-EDIT` geschieht in der Reihenfolge „tasklokaler Hardware-AUDIT (TPR) -> tasklokaler Hardware-AUDIT (TU) -> tasklokaler Linkage-AUDIT (TPR) -> tasklokaler Linkage-AUDIT (TU) -> prozessorlokaler Linkage-AUDIT“. Angezeigt wird die erste gefundene Tabelle.

Der Hardware-AUDIT wird nur auf /390-Servern unterstützt.

Eingabemöglichkeiten in der Kopf-Zeile

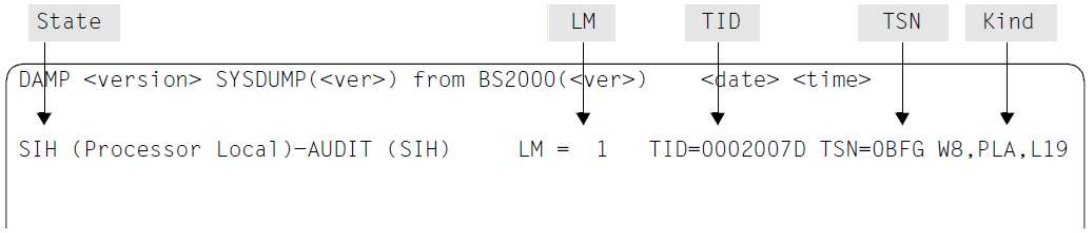


Bild 51: Eingaben in die Kopfzeile des AUDIT-Fensters

- Feld **State**
  hier kann der Programmstatus der gewünschten AUDIT-Tabellen angegeben werden:

  SIH System-Interrupt-Handling (nur bei „Kind = PLA“)
  TPR Taskprivileged (nur bei „Kind = LKA“ oder „Kind = HWA“)
  TU Taskunprivileged (nur bei „Kind = LKA“ oder „Kind = HWA“)

- Feld **LM**
  gibt die sedezimale Nummer der LM (nur bei „Kind = PLA“) an.
- Feld **TID**
  gibt den Task-Identifier (nur bei „Kind = LKA“ oder „Kind = HWA“) an.
- Feld **TSN**
  gibt die Task-Sequence-Number (nur bei „Kind = LKA“ oder „Kind = HWA“) an.
- Feld **Kind**
  gibt die Art der AUDIT-Tabellen an.

  PLA Processor local Linkage AUDIT
  LKA Task local Linkage AUDIT
  HWA Hardware AUDIT

In der AUDIT-Tabellen-Ausgabe werden die Blätterfunktionen +/ F3, ++, - / F1, +n, -n unterstützt. Näheres hierzu siehe „Blättern in einem Diagnosefenster" (Diagnosefenster verändern).

Alle Adressen sind markierbar. Die Darstellung der Adressen ist virtuell und wenn möglich symbolisch (Modulname + Distanz) aufbereitet.

Bei Anzeige des prozessorlokalen Linkage-AUDIT werden die Eingabefelder „TID“ und „TSN“ nicht ausgewertet.

Wird eine AUDIT-Tabelle ausgegeben, zeigt die zweite Zeile die Adressen des jeweiligen AUDIT-Verwaltungsbereiches („EXVTLAUD“) und der jeweiligen AUDIT-Tracetabelle („AuditTable“). Das Feld „AuditTable“ erhält den Zusatz „(current)“, wenn der angezeigteAUDIT zum Dumpzeitpunkt eingeschaltet war. Wurde der angezeigte AUDIT vor dem Dumpzeitpunkt mit dem Kommando HOLD-LINKAGE-AUDIT oder HOLD-HARDWARE-AUDIT in den Zustand DISCONTINUE versetzt, wird das Feld „AuditTable“ mit „(obsolet)“ ergänzt.

Layout eines AUDIT-Fensters

```
DAMP <version> SYSDUMP(<ver>) from BS2000(<ver>)    <date> <time>


SIH (Processor Local)-AUDIT (SIH)      LM =  1   TID=0002007D TSN=0BFG W8.PLA.L19
EXVTLAUD=72408C40   AuditTable (current)=7241E000      ">" indicates LAST branch
> F10053A0 NLCNLMAN+003A0 |  F10450E0 EMMPGSRV+00860 |  F1064AC0 ETMPI$X +00000
  F1062648 EMMPLATO+00288 |  F1064DE8 ETMPI$X +00328 |  F1064AC0 ETMPI$X +00000
  F1062648 EMMPLATO+00288 |  F106084C ETMTIM$X+002CC |  F1062648 EMMPLATO+00288
  F1006E30 NLCNLMAN+01E30 |  F1006CAC NLCNLMAN+01CAC |  F1064F72 ETMPI$X +004B2
  F1056158 ETMPSUBR+01298 |  F1006E30 NLCNLMAN+01E30 |  F1055C40 ETMPSUBR+00D80
  F10559B8 ETMPSUBR+00AF8 |  F1055660 ETMPSUBR+007A0 |  F1060EA0 ETMTIM$X+00920
  F1056028 ETMPSUBR+01168 |  F1056080 ETMPSUBR+011C0 |  F1006CAC NLCNLMAN+01CAC
  F1055790 ETMPSUBR+008D0 |  F1063C58 ETMPSHEL+00258 |  F1031186 NDISERV +02546
  F10058F0 NLCNLMAN+008F0 |  F10053A0 NLCNLMAN+003A0 |  F10450E0 EMMPGSRV+00860
  F1045000 EMMPGSRV+00780 |  F1025320 EMMPGFIX+022A0 |  F10450E0 EMMPGSRV+00860
  F1045000 EMMPGSRV+00780 |  F1025320 EMMPGFIX+022A0 |  F10450E0 EMMPGSRV+00860
  F1045000 EMMPGSRV+00780 |  F1025320 EMMPGFIX+022A0 |  F10450E0 EMMPGSRV+00860
  F1045000 EMMPGSRV+00780 |  F1025320 EMMPGFIX+022A0 |  F10450E0 EMMPGSRV+00860
  F1045000 EMMPGSRV+00780 |  F1025320 EMMPGFIX+022A0 |  F10450E0 EMMPGSRV+00860
  F1045000 EMMPGSRV+00780 |  F1025320 EMMPGFIX+022A0 |  F10450E0 EMMPGSRV+00860
  F1045000 EMMPGSRV+00780 |  F1025320 EMMPGFIX+022A0 |  F10450E0 EMMPGSRV+00860
  F1045000 EMMPGSRV+00780 |  F1025320 EMMPGFIX+022A0 |  F100521C NLCNLMAN+0021C
CMD:
Key: 1=Help 2=Plk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=AUDI   9=Dump
```

Bild 52: Informationen über AUDIT-Tabellen

5.3.2.20 Zeichenketten suchen (Spezialfenster FIND)

Mit der Funktion FIND können Sie Zeichenketten im gesamten Bereich des Diagnoseobjekts suchen lassen. Dabei kann der Suchbereich nach Speicher-Intervall, Ladeeinheiten bzw. Speicherklassen definiert werden.
DAMP unterstützt die Suche mit einem und mit zwei Suchbegriffen. Bei der Suche mit zwei Suchbegriffen muss der Abstand zwischen dem Anfang des ersten und dem Anfang des zweiten Suchbegriffs angegeben werden.

Bezogen auf den ersten Suchbegriff kann der Anwender eine spezielle Speicherausrichtung angeben. Für beide Suchbegriffe werden von DAMP verschiedene Formate von Zeichenketten (Sedezimal-, Character-, String- und Text-Format) und veränderbare Wildcard-Symbole unterstützt. Durch Angabe einer maximalen Anzahl von Treffern kann die Suche eingeschränkt werden.

Voraussetzung für die Ausführung der FIND-Funktion ist ein geöffnetes Diagnoseobjekt. Aufgerufen wird die Suche nach Zeichenketten durch die Anweisung

```
START-PATTERN-SEARCH WINDOW=<w>
```

Nach dem Aufruf erscheint zuerst das FIND-Auswahlfenster. Dieses Auswahlfenster dient zur Festlegung des Suchbereichs und der Suchbegriffe. Durch eine entsprechende Eingabe in den Feldern „ASEL“ und „ASID“ kann in jedem von DAMP unterstützten Adressraum gesucht werden (siehe auch "Eingabefelder der Standard-Dumpfenster (W4 - W9 und W21 - W99)").

- im virtuellen Adressraum (ASEL = TSN | TID),
- im Datenraum (ASEL = ALT | SPI),
- im realen Adressraum (ASEL = RM),
- im absoluten Adressraum (ASEL = ABS),
- im Processor Saved Status (ASEL = PSS),
- in der Hardware System Area (ASEL = HSA),
- in einer Dumpfile-Section (ASEL = SCT).

Dem FIND-Fenster liegen folgende Eingabe-Prinzipien zu Grunde:

- Alle Eingabedaten bleiben nach Ausführung der FIND-Funktion erhalten und sind für die nächste Funktionsausführung voreingestellt; es kann immer auf der vorherigen Spezifikation aufgesetzt werden.
- Alle Eingabefelder werden bei der Funktions-Ausführung so interpretiert, wie sie auf dem Bildschirm angezeigt werden.
- Vor der Funktionsausführung muss nur ein Eingabefeld modifiziert werden. Eine Ausnahme stellt lediglich ein neues FIND-Fenster dar: hier müssen zumindest der Suchbereich und der 1. Suchbegriff definiert werden.
- Zum Zurücksetzen von Eingaben auf „nicht spezifiziert“ dürfen ausschließlich Leerzeichen verwendet werden. Nil-Zeichen (X'00') dürfen dazu nicht benutzt werden. (Ausnahmen bilden die beiden Suchbegriffe.)

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


FIND - Command                                  TID=00010001          W9,D  ,L19
   Interval     :    Start     =           End     =
or Load Unit    :    Scope     =  ALL       CLASS4    PRIV      NONPRIV   USER
                     Subsystem =           Version =
                     Module    =
or Memory Class :     ALL    C11    C12    C13-PP    C14-PP    C15-PP    C16-FP
                                           C13-FP    C14-FP    C15-FP    C16-MP
                                                     C14-NP    C15-MP

                     Wildcard Symbol = *           Alignment  (B/H/W/D/P)  = H
                     Number of Hits  = 18          Count only (Y/N)        = N
Output Area: *SYSOUT

1.Search Strg C:
Offset :
2.Search Strg C:

Cancel possible with K2 + /INFORM-PROGRAM MSG='CANCEL' (/INTR CANCEL).

CMD:
Key: 1=Help 2=Plk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=Dump   9=FIND
```

Bild 53: Auswahlmaske für die Suche im virtuellen Adressraum

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


FIND - Command                                ALT=00000000-00010017 W9,D  ,L19
   Interval     :    Start     =           End     =









                     Wildcard Symbol = *           Alignment  (B/H/W/D/P)  = H
                     Number of Hits  = 18          Count only (Y/N)        = N
Output Area: *SYSOUT

1.Search Strg C:
Offset :
2.Search Strg C:

Cancel possible with K2 + /INFORM-PROGRAM MSG='CANCEL' (/INTR CANCEL).

CMD:
Key: 1=Help 2=Plk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=Dump   9=FIND
```

Bild 54: Auswahlmaske für die Suche in den übrigen Adressbereichen (am Beispiel ASEL=ALT)

Die Ausgabe erfolgt standardmäßig am Bildschirm und in das gleiche Diagnosefenster wie die Eingabe. Der Anwender kann als Ausgabemedium auch eine Datei, *SYSLST oder *EDT angeben (Output Area). Weiter hat er die Möglichkeit, nur die Anzahl der gefundenen Treffer ausgeben zu lassen und die Trefferliste zu unterdrücken („Count only=Y“).

Spezifikation des Suchbereichs

Bei der Auswahl des Suchbereichs im virtuellen Adressraum können ein Speicher-Intervall (**Interval**), eine oder mehrere Ladeeinheiten (**Load Unit**) bzw. eine oder mehrere Speicherklassen (**Memory Class**) angegeben werden. Alle drei Suchbereichs-Spezifikationen sind disjunkt: es muss also immer genau eine Spezifikation gültig sein, Verknüpfungen werden nicht unterstützt.
In den anderen Adressräumen kann nur ein **Interval** angegeben werden.

Die Auswahl des aktuell gültigen Suchbereichs erfolgt durch DAMP:

- ist genau ein Suchbereich spezifiziert - explizit spezifiziert oder bei dem vorhergehenden FIND-Aufruf voreingestellt - so wird dieser ausgewählt;
- sind zwei Suchbereiche spezifiziert, von denen genau einer explizit spezifiziert wurde, so wird dieser ausgewählt;
- sind mehrere Suchbereiche explizit spezifiziert, wird der Aufruf der Funktion mit Ausgabe einer Meldung zurückgewiesen.

Ein Suchbereich ist genau dann explizit spezifiziert, wenn die Modifikationen sich nicht ausschließlich auf das Überschreiben von Feldern mit Leerzeichen beschränken.
Nach der Auswahl eines Suchbereichs werden alle Werte der anderen Suchbereiche von DAMP implizit zurückgesetzt, ein explizites Zurücksetzen ist nicht erforderlich.

Für den virtuellen Adressraum (ASEL=TSN | TID) wird die Suche in einem ausgewählten Adressraum (ASID=<tsn>|<tid>) sowie in allen im Objekt enthaltenen Adressräumen (ASID=*ALL) unterstützt. Bei ASID=<tsn>|<tid> können Suchbereiche aus dem Benutzer- und Systemspeicher eingegeben werden, bei ASID=*ALL ist nur die Eingabe von Bereichen, die vollständig im Benutzeradressraum liegen, erlaubt.

- Suchbereich **Interval**

  Feld **Start** gibt die Startadresse des Suchbereichs an
  Feld **End** gibt die Endadresse des Suchbereichs an

  Der Suchbereich ist nur dann vollständig definiert, wenn sowohl Start- als auch Endadresse spezifiziert sind. Es gibt keine Voreinstellungen.

- Suchbereich **Load Unit**
  Dieser Suchbereich spezifiziert die Module von BLS- bzw. DSSM-Ladeeinheiten.

  Die Eingrenzung des Suchbereichs erfolgt hierarchisch in folgenden Ebenen:

  „Scope“ -> „Subsystem“ -> „Version“ -> „Module“.

  Falls „Load Unit“ als Suchbereich eingestellt wird, müssen nicht alle verfügbaren Felder spezifiziert werden. Für jedes Feld wird, falls es nicht spezifiziert ist, eine implizite Annahme getroffen.

  Auswahlleiste **Scope** Spezifiziert ein oder mehrere BLS- bzw. DSSM-Ladebereiche

ALL	alle Module (aus CP, allen Subsystemen und dem Benutzerprogramm)
CLASS4	alle Module aus dem Systemadressraum, außer CP
PRIV	alle Module aus CP und den privilegierten Subsystemen
NONPRIV	alle Module aus den nichtprivilegierten Subsystemen und dem Benutzerprogramm
USER	alle Module aus dem Benutzerprogramm

  Falls „Load Unit“ als Suchbereich eingestellt und „Scope“ nicht angegeben wurde, wird implizit „Scope=ALL“ angenommen.Bei NONPRIV bzw. USER hängt das Ergebnis der Suche davon ab, ob die eingestellte Task an die Subsysteme angeschlossen ist bzw. ob die Task ein Programm geladen hat.

Feld **Subsystem**	Spezifiziert ein oder mehrere Subsysteme. Subsystemnamen können in einer Länge von bis zu 8 Zeichen eingegeben werden. Zur Angabe von mehreren Subsystemen werden Wildcards unterstützt: das Symbol „*“ ersetzt hierbei beliebig viele Zeichen des Namens, während „/“ genau ein Zeichen ersetzt. Falls „Load Unit“ als Suchbereich eingestellt und „Subsystem“ nicht angegeben wurde, wird implizit „Subsystem=*“ (alle Subsysteme aus dem spezifizierten „Scope “) angenommen.
Feld **Version**	Spezifiziert eine oder mehrere Subsystem-Versionen. Versionen können in einer Länge von bis zu acht Zeichen eingegeben werden. Zur Angabe von mehreren Versionen werden Wildcards unterstützt (analog zum Feld „Subsystem“). Falls „Load Unit“ als Suchbereich eingestellt und „Version“ nicht angegeben wurde, wird implizit „Version=*“ (alle Versionen der spezifizierten Subsysteme) angenommen
Feld **Module**	Spezifiziert ein oder mehrere Module. Modulnamen können in einer Länge von bis zu 32 Zeichen eingegeben werden. Zur Angabe von mehreren Modulen werden Wildcards unterstützt (analog zum Feld **Subsystem**). Falls „Load Unit“ als Suchbereich eingestellt und „Module“ nicht angegeben wurde, wird implizit „Module=*“ (alle Module der spezifizierten Subsysteme) angenommen.

- Suchbereich **Memory Class**

  Dieser Suchbereich spezifiziert eine oder mehrere Speicherklassen als Suchbereich(e). Die Speicherklassen werden durch Ankreuzen ausgewählt, wobei alle möglichen Kombinationen erlaubt sind. Die Auswahl von Speicherklassen aus dem Benutzeradressraum gilt jeweils nur für die aktuell eingestellte Task.

  Es werden folgende (Sub-)Speicherklassen unterstützt:

Feld		gewählter Bereich	
ALL		alle Speicherklassen	
CL1	Class-1-Memory	residente Systemmodule	System-adress-raum
CL2	Class-2-Memory	seitenwechselbare Systemmodule	
CL3PP	Class-3-Partial-Pages	residente Teilseiten	
CL3FP	Class-3-Full-Pages	residente Vollseiten	
CL4PP	Class-4-Partial-Pages	seitenwechselbare Teilseiten	
CL4FP	Class-4-Full-Pages	seitenwechselbare Vollseiten	
CL4NP	Class-4-Nonpriv-Pages	nicht privilegierte Klasse-4-Seiten	
CL5PP	Class-5-Partial-Pages	privilegierte Teilseiten	Benutzer-adress raum
CL5FP	Class-5-Full-Pages	privilegierte Vollseiten	
CL5MP	Class-5-Memory-Pool	Klasse-5-Memory-Pool	
CL6FP	Class-6-Full-Pages	nicht privilegierte Vollseiten	
CL6MP	Class-6-Memory-Pool	Klasse-6-Memory-Pool	

Tabelle 7: Speicherbereiche (Auswahl erfolgt durch Ankreuzen)

Spezifikation der Suchbegriffe

Es ist die Angabe von einem Suchbegriff bzw. von zwei Suchbegriffen möglich. Bei der Angabe von zwei Suchbegriffen muss der Anwender den Abstand zwischen dem Anfang des ersten und dem Anfang des zweiten Suchbegriffs angeben. Die Spezifikation der Suchbegriffe erfolgt über die Felder **1.Search Strg**, **Offset** und **2. Search Strg**.

Über das Feld **Alignment** kann, bezogen auf den ersten Suchbegriff, eine Speicherausrichtung spezifiziert werden. Außerdem kann das Feld **Wildcard Symbol** vom Anwender verändert werden.

- Feld **1.Search Strg**
  spezifiziert einen (den ersten) Suchbegriff. Dieser Suchbegriff muss immer angegeben sein. Er besteht aus maximal 64 Zeichen, denen ein Byte als Format-Typ vorangestellt ist.

  Es werden folgende Formate unterstützt:

X	Sedezimal-Format	erlaubt sind die Zeichen 0..9 und A..F
C	Character-Format	Umwandlung der Klein- in Großbuchstaben beim Suchbegriff und anschließender Vergleich
S	String-Format	ohne Umwandlung
T	Textformat	Umwandlung der Groß- in Kleinbuchstaben, sowohl beim Suchbegriff als auch beim Suchbereichsinhalt, und anschließender Vergleich

  Voreingestellt ist das Format C.

  Beim Suchbegriff werden Wildcards unterstützt. Das Wildcard-Symbol entspricht dem unter „Wildcard Symbol“ definierten Zeichen. Es kann an beliebiger Stelle des Suchbegriffs verwendet werden und ersetzt jeweils genau ein Zeichen im Suchbegriff.
- Feld **Offset**
  spezifiziert den Abstand zwischen dem Anfang des „1.Search Strg“ und dem Anfang des „2.Search Strg“. Der „Offset“ wird als Sedezimalwert angegeben. Um den Offset auf „nicht spezifiziert“ zu setzen, muss das Eingabefeld mit Leerzeichen überschrieben werden.
- Feld **2.Search Strg**
  spezifiziert den zweiten Suchbegriff. Der „2.Search Strg“ ist analog zum „1.Search Strg“ definiert. Er wird nur dann für die Suche berücksichtigt, wenn ein „Offset“ angegeben ist.
- Feld **Wildcard Symbol**
  spezifiziert ein Zeichen, das im „1.Search Strg“ und im „2.Search Strg“ als Wildcard-Symbol verwendet werden darf. Das Wildcard-Symbol ist mit „*“ voreingestellt und kann vom Anwender verändert werden. Erlaubt sind alle Zeichen außer Ziffern, Buchstaben und Leerzeichen.
- Feld **Alignment**
  spezifiziert eine Speicherausrichtung, bezogen auf den „1.Search Strg“.

  Es werden folgende Ausrichtungs-Typen unterstützt:

B	Bytegrenze
H	Halbwortgrenze
W	Wortgrenze
D	Doppelwortgrenze
P	Seitengrenze

  Voreingestellt ist das Format H.

Seitengrenze (P) bedeutet in der Regel, dass die Suche auf 4-KB-Grenze ausgerichtet wird. Die einzige Ausnahme stellt die Suche in Objekten dar, die als PAM-Dateien geöffnet wurden. In diesem Falle wird auf 2-KB-Grenze gesucht.

Spezifikation der Ausgabe bei der FIND-Funktion

Über das Feld „Output Area“ wird das Ausgabemedium festgelegt. Standardmäßig erfolgt die Ausgabe am Bildschirm und in das gleiche Diagnosefenster wie die Eingabe. Über das Feld „Number of Hits“ wird die maximale Anzahl der Treffer für einen Suchlauf eingestellt und „Count only“ legt fest, ob die Trefferliste ausgegeben werden soll oder nur die Anzahl der Treffer. Das Format der Ausgabe ist über das „Modus-Feld“ in der Kopfzeile wählbar.

- Feld **Output Area**
  spezifiziert das Ausgabemedium.

  Es werden folgende Ausgabemedien unterstützt:

*SYSOUT	Bildschirm, gleiches Diagnosefenster wie bei der Eingabe
<filename>	Datei
*SYSLST	Systemdatei *SYSLST
*EDT	EDT-Fenster, das zuletzt verwendet wurde, bzw. EDT-Fenster 0

  Voreingestellt ist *SYSOUT.

- Feld **Number of Hits**
  spezifiziert die maximale Anzahl der Treffer, nach denen die Suche unterbrochen und die gefundenen Treffer ausgegeben werden sollen. Die Treffer werden als Dezimalwert eingegeben. Die maximale Anzahl von ausgegebenen Treffern ist eingeschränkt durch die Größe des Ausgabefensters.

- Feld **Count only**
  spezifiziert, ob die Trefferliste ausgegeben werden soll oder nur die Anzahl der Treffer.

  Es werden folgende Angaben unterstützt:

N	Trefferliste
Y	nur Anzahl gefundener Treffer

  Voreingestellt ist Count only=N, d.h. die Ausgabe der Trefferliste.

- Feld **Modus** in der Kopfzeile

  Es werden folgende Ausgabeformate unterstützt:

D(MP)	normales Dump-Format
C(HR)	Zeichen-Format
H(EX)	Sedezimal-Format

  Voreingestellt ist das Format D(MP).

Das Ausgabeformat kann im Auswahl- und im Ausgabefenster eingegeben werden.

Ausgabefenster der FIND-Funktion

Nach dem Starten der FIND-Funktion mit Output Area = *SYSOUT (dies ist Voreinstellung) werden im gleichen Diagnosefenster die Treffer ausgegeben. Wurde **ein** Suchbegriff angegeben, wird für jeden Treffer eine Zeile ausgegeben; wurden **zwei** Suchbegriffe angegeben, so belegt ein Treffer zwei Zeilen, wobei in der jeweils zweiten Zeile der „Offset“ der Ausgabe vorangestellt ist.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


FIND - Command                                  TID=000A01EB          W8,D  ,L19
724FD008 (ETCB-002+00000) E3C3C240 00010002 C8C5D9E2 00008000 = TCB ????HERS????
724FD016 = * + 0000000E   80000400 00000000 00000000 00000000 = ??????????????????
724FD7A8 (ETCB-004+00000) E3C3C240 00010004 C3D3D6C7 00008000 = TCB ????CLOG????
724FD7B6 = * + 0000000E   80000400 00000000 00000000 00000000 = ??????????????????
72500008 (ETCB-00D+00000) E3C3C240 0001000D E3C1D7F1 00008000 = TCB ????TAP1????
72500016 = * + 0000000E   80000400 00000000 00000000 00000000 = ??????????????????
```

Bild 55: FIND-Ausgabefenster (bei der Suche mit zwei Suchbegriffen)

Die Suche wird unterbrochen, wenn das Fenster gemäß der aktuellen Fensterlänge ganz mit Trefferzeilen gefüllt ist oder die im Eingabefeld „Number of Hits“ eingegebene maximale Anzahl von Treffern erreicht ist. Sie kann dann mit +/ F3 fortgesetzt oder mit - / F1 abgebrochen werden.

Zudem kann durch K2 mit anschließendem /INFORM-PROGRAM MSG='*CANCEL' ein Abbruch der Stringsuche erzwungen werden. Es werden dann die bisher gefundenen Treffer angezeigt.

Umfasst der Suchbereich Seiten, die nicht im Diagnoseobjekt enthalten sind, so wird dies durch Meldungen angezeigt, die Suche wird aber nicht abgebrochen.

Die Adresse des gefundenen Suchmusters kann in jeder Trefferzeile markiert und der Speicherbereich in einem Dumpfenster ausgegeben werden, siehe Abschnitt „Markieren" (Diagnosefenster verändern).

In den Ausgabemodi D und HEX sind im Ausgabebereich die einzelnen Wörter markierbar und können als Anfangsadressen den einzelnen Dumpfenstern zugeordnet werden.

5.3.3 Modifikationen durch den Benutzer (Spezialfenster OPTIONS)

Unabhängig von der Einstellung bei Auslieferung oder durch die Systemverwaltung kann der Benutzer die Benutzeroptionen für seine DAMP-Anwendung entsprechend einstellen. Einstellbar sind die Kennungen für die Pfadnamen der von DAMP benötigten Dateien und weitere Optionen.

Standardnamen

Im Abschnitt „Software- und Hardware-Voraussetzungen" sind alle Produkt-Dateien von DAMP mit ihren Release-Namen und ihrer Bedeutung aufgeführt.

Unter dem Standardnamen einer Produktdatei wird der von IMON bereitgestellte Pfadname verstanden.

Die Standardnamen der System-Symbolbibliothek bzw. der System-PRODAMP-Bibliothek sind die festen Pfadnamen $TSOS.SYSSMB.DAMP bzw. $TSOS.SYSDMP.DAMP.

DAMP arbeitet stets mit diesen Standardnamen. In einem individuell erzeugten DAMP können andere Namen eingestellt werden (siehe „Einstellbare Parameter").

Einstellen der Benutzeroptionen

Der DAMP-Benutzer kann Benutzeroptionen individuell für seine Anwendung einstellen.Die Einstellung /Veränderung der Benutzeroptionen erfolgt nach dem Laden von DAMP durch die Anweisung `START-OPTION-DIALOG`. Mit dieser Anweisung wird ein Fenster eröffnet, in dem die entsprechenden Einstellungen durch Überschreiben und Markieren geändert werden können. In der Key-Zeile erscheint die Anzeige OPTS für das verwendete Diagnosefenster.

Die Benutzeroptionen können bei jedem DAMP-Aufruf temporär verändert werden, in der Kopfzeile des Diagnosefensters ist im Feld „Ausgabeformat“ standardmäßig **TMP** eingetragen.

Für eine permanente Einstellung der Benutzeroptionen beginnen Sie am zweckmäßigsten damit, das ausgelieferte Ladeprogramm SYSPRG.DAMP.<ver> in eine benutzereigene Datei, mit dem Namen „DAMP“, zu kopieren. Anschließend rufen Sie DAMP mit dem neuen Programmnamen auf, setzen die Anweisung START-OPTION-DIALOG ab, stellen Ihre Benutzeroptionen ein und überschreiben das Feld „Ausgabeformat“ mit **SAV**. Nach Drücken der Taste DUE wird die Prozedur S.PRC.DAMP.<ver>.OPTIONS erzeugt. Die Prozedur muss nach Beendigung von DAMP mit `/call-proc s.prc.damp.<ver>.options` gestartet werden; sie modifiziert das angegebene Ladeprogramm.

```
DAMP <version>  No Object opened in BS2000 V<ver>  <date> <time>


DAMP user options                                                  W9,TMP,L19

Userids:  SYSLNK / SYSDMP   =  *STD        SYSPAR  (REDUCE)  =  *STD
          SYSSMB            =  *STD        SYSLNK  (ANITA)   =  *STD
          SYSMSH / SYSSDF   =  *STD

Window separation:       yes/no           Window separator          =  -/X'60'
Column separator (screen)   =  |/X'4F'     Column separator (list)   =  |/X'4F'
Trash character             =  ?/X'07'

Message:  Language          =  ENGLISH     Blinking:               yes/no

Lines per list page         =   65

K1 check-back:           yes/no            Save P-Keys:            yes/no

PRODAMP:  Source      =  *STD
          Object      =  *STD

CMD:
Key: 1=Help 2=Plk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=OPTS   9=OPTS
```

Bild 56: OPTS-Fenster

Einstellbare Parameter

Folgende Benutzeroptionen stehen in DAMP zur Verfügung, die bei Auslieferung voreingestellten Werte sind fett gedruckt.

Der Wert *STD für eine der unten angegebenen Userids bedeutet, dass der jeweilige Standardname (siehe "Modifikationen durch den Benutzer (Spezialfenster OPTIONS)") als Pfadname für die dadurch beschriebene Datei verwendet wird.

Wird als Wert eine Kennung eingetragen, sucht DAMP die entsprechende Datei ausschließlich unter der eingetragenen Kennung.

User IDs: SYSLNK/SYSDMP = ***STD** | <userid>

Gemeinsame Kennung für die Nachladebibliothek von DAMP und die System-PRODAMP-Bibliothek. Die System-PRODAMP-Bibliothek enthält unter anderem die PRODAMP-Routinen der automatischen Vordiagnose.

SYSSMB = ***STD** | <userid>

Benutzerkennung für die System-Symbolbibliothek, auf die beim Laden der Symbole defaultmäßig zugegriffen wird.

SYSMSH / SYSSDF = ***STD** | <userid>

Gemeinsame Benutzerkennung der Hilfedateien, der Meldungsdateien und der SDF-Benutzersyntaxdateien mit den DAMP-Anweisungen.

SYSLNK (ANITA) = ***STD** | <userid>

Benutzerkennung, unter der die Nachladebibliothek von ANITA katalogisiert ist. Die Angabe einer Kennung bewirkt, dass die Bibliothek SYSLNK.ANITA unter der angegebenen Kennung gesucht wird. Eine evtl. geladene alte Version einer ANITA-Modulbibliothek wird entladen und eine neue Version geladen. (Die Zugriffsmethode ANITA wird von DAMP bei Zugriffen auf Dumpdateien und das aktive System benutzt.)

Window separation: **yes** | no

Die Diagnosefenster werden mit/ohne Strichzeile voneinander getrennt.

Window separator = **- / X'60'**

Fenster-Trennzeichen für die Ausgabe auf Terminal.

Column separator (screen) = **| / X'4F'**

Spalten-Trennzeichen für die Ausgabe auf Terminal, weil X'4F' nicht auf allen Terminals als „|“ dargestellt wird.

Column separator (list) = **| / X'4F'**

Spalten-Trennzeichen für die List-Ausgabe, weil X'4F' nicht auf allen Druckern mit „|“ ausgegeben wird.

Trash character = **. / X'07'**

Ersatzzeichen für nicht abdruckbare Zeichen auf Terminal. Möglich sind X'00', X'07' und abdruckbare Zeichen.

Message: Language = **ENGLISH** | DEUTSCH

Sprache, in der die Texte im Help-Fenster, die Online-Hilfetexte und die DAMP-Meldungen ausgegeben werden. Die Sprache kann auch über das Help-Fenster (W1) eingestellt werden, siehe "Das Help-Fenster (W1)".

Blinking: yes | **no**

Die Ausgabe der DAMP-Meldungen in den Zeilen 2 und 3 blinkend oder nicht blinkend.

Lines per list page = **65**

Legt fest, wie viele Zeilen bei der List-Ausgabe auf eine Seite gedruckt werden sollen.

K1 check-back: **yes** | no

Legt fest, ob nach dem Drücken der K1-Taste DAMP sofort oder erst nach einer Rückfrage beendet wird.

Save P-Keys: **yes** | no

Legt fest, ob die P-Tasten vor dem Beschreiben durch DAMP gesichert werden und bei Unterbrechungen sowie beim Beenden von DAMP restauriert werden sollen. Diese Option wird nur bei bestimmten Datensichtstationen (vom Typ 976x) ausgewertet.

PRODAMP: Source = ***STD** | <filename>

Bibliothek, in der die Benutzer-Quellprogramme abgespeichert werden.

Object = ***STD** | <filename>

Bibliothek, in der die Benutzer-Objekte abgespeichert werden.

In beiden Fällen steht *STD für die Bibliothek SYS.USRDMP.DAMP.<ver> der Ablaufkennung.

> Eine temporäre Einstellung für die Benutzer-PRODAMP-Bibliotheken wird nicht sofort wirksam. Sie ist lediglich eine Voreinstellung für den Wert *STD in den Operanden SOURCE-LIBRARY bzw. OBJECT-LIB-RARY der DAMP-Anweisung ASSIGN-PRODAMP-LIBRARIES. Sollen die Einstellungen sofort wirksam werden, ist diese Anweisung im Anschluss entsprechend einzugeben.

5.3.4 Weitere Funktionen

- EDT als Unterprogramm aufrufen
- Diagnosesitzung protokollieren und wiedergeben
- Dateien mit PAM-Format bearbeiten
- Bearbeitung von SLEDs ohne BS2000-Struktur
- Private Symbolelemente verwenden
- Private Assembler-Benutzerroutinen schreiben

5.3.4.1 EDT als Unterprogramm aufrufen

Mit der Anweisung EDIT-FILE kann der EDT als Unterprogramm aufgerufen werden. Es stehen dann die EDT-Funktionen zur Verfügung. Siehe dazu auch Handbuch „EDT“ [2]. Dadurch können, parallel zur Bearbeitung der Dumpdatei in den Diagnosefenstern, zusätzliche Unterlagen-Dateien wie CONSLOG, SERSLOG, HERSFILE etc. zur Diagnose am Bildschirm benutzt werden.

Für einige Funktionen benutzt DAMP selbst den EDT:

- Die Deskriptorenliste aus der automatischen Voranalyse wird im EDT-Bereich 8 abgelegt.
- Im FILE-Spezialfenster können System-Dateien und Dump-Sections mit dem EDT bearbeitet werden.
- Zum Editieren von Prozeduren und für Ausgaben laufender Prozeduren nutzt die Diagnosesprache PRODAMP den EDT.

Im EDT stehen alle EDT-Funktionen zur Verfügung bis auf die EDT-Anweisungen @LOAD und @EXEC. Sie werden generell abgewiesen.

Meldungen des EDT und des DAMP-Systems werden in der EDT-Programmebene in der letzten bzw. vorletzten Datenzeile des EDT-Schirmes ausgegeben. Ging ein Teil des Bildschirminhalts verloren, kann mit K3 der vorherige Zustand wiederhergestellt werden.

Durch Drücken der Taste K1 kehren Sie in die DAMP-Programmebene zurück. Im F-Modus können Sie aber auch HALT oder END und im L-Modus @HALT oder @RET eingeben.

Die Anweisung HALT kann auch mit folgenden Operanden eingegeben werden:

HALT @	Ausgabe des aktuellen EDT-Anweisungs-Symbols in einer DAMP-Meldungszeile
HALT msg	Ausgabe der Zeichenkette „msg“ in der DAMP-Meldungszeile
HALT #msg	Ausgabe der Zeichenkette „msg“ in der DAMP-Kommandozeile

Dies gilt ebenso für @HALT und @RET (aber nicht für END).

5.3.4.2 Diagnosesitzung protokollieren und wiedergeben

Sämtliche Bildschirmein-/ausgaben können protokolliert, d.h. in eine Datei geschrieben werden.

- Die Protokolldatei kann mit der Anweisung `PRINT-LOGGING-FILE` druckaufbereitet werden. Der Ausdruck wird ebenfalls über dieses Kommando angestoßen (Beschreibung der Anweisung, siehe "PRINT-LOGGING-FILE Aufbereiten und Ausdrucken einer Logging-Datei").
- Die Diagnoseschritte können zum Beispiel von einem anderen Diagnostiker nachvollzogen und überprüft werden.
- Die bisherigen Diagnoseschritte können vom gleichen Bearbeiter nachvollzogen werden, wenn die Diagnose beabsichtigt oder unerwartet unterbrochen wurde.

Diagnosesitzung protokollieren

Das Protokollieren der Diagnosesitzung wird auf Programmebene mit der Anweisung `LOG-SESSION` und auf Systemebene mit dem BS2000-Kommando `INFORM-PROGRAM MSG='*LOG-SESSION'` aktiviert.

Der Dateiname der Protokolldatei wird auf Programmebene mit der Anweisung `LOG-SESSION LOGGING-FILE=filename` oder auf Systemebene mit dem BS2000-Kommando `ADD-FILE-LINK LINK-NAME=DAMPLOG, FILE-NAME=filename` vereinbart.

Nach dem Schließen der Logging-Datei wird der Linkname DAMPLOG freigegeben.
Ist keine solche Vereinbarung getroffen worden, wird der Dateiname automatisch generiert, und zwar nach dem Muster `S.LOG.DAMP.<ver>.<date>.<time>`.

Der Mitschnitt wird auf Programmebene mit der Anweisung `STOP-LOGGING` und auf Systemebene mit dem BS2000-Kommando `INFORM-PROGRAM MSG='*STOP-LOGGING'` beendet.

Bei Weiterleitung des Problems an die nächste Diagnoseinstanz sollte die Logging-Datei in Dateiform mitgegeben werden.

Diagnosesitzung ausdrucken

Ein Diagnose-Mitschnitt wird mit der Anweisung `PRINT-LOGGING-FILE` ausgedruckt (Beschreibung der Anweisung, siehe "PRINT-LOGGING-FILE Aufbereiten und Ausdrucken einer Logging-Datei").

Diagnosesitzung wiedergeben

Jede mit DAMP erstellte Protokolldatei kann von der gleichen oder einer anderen Diagnoseinstanz mit DAMP wiedergegeben werden.

Die Diagnose-Wiedergabe wird auf Programmebene mit der Anweisung `REPEAT-SESSION <loggingfilename>` und auf Systemebene mit dem Kommando `INFORM-PROGRAM MSG='*REPEAT-SESSION (<loggingfilename>)'` gestartet.

Alle Ein- und Ausgaben des protokollierten DAMP-Dialogs werden am Bildschirm gezeigt und müssen mit DUE oder K3 quittiert werden.

Das Drücken der Taste K2 bewirkt einen Wechsel in den Systemmodus. Befindet man sich im Systemmodus, erreicht man die Rückkehr zur Wiedergabe-Funktion durch das Kommando `RESUME-PROGRAM`.

Mit K1 kann die Wiedergabe vorzeitig abgebrochen werden.

Wird die gesamte Protokolldatei abgespielt, endet die Wiedergabe in der Ebene, in der das Protokoll gestartet wurde.
Wurde die Wiedergabe mit `REPEAT-SESSION <loggingfilename>` gestartet, erscheint am Ende der Wiedergabe die DAMP-Bildschirmmaske.

Wurde die Wiedergabe aus dem Systemmodus mit dem Kommando `INFORM-PROGRAM MSG='*REPEAT-SESSION (<loggingfilename>)'` gestartet, befindet man sich am Ende der Wiedergabe im Systemmodus. Die DAMP-Diagnose kann in diesem Fall mit dem Kommando `RESUME-PROGRAM` fortgesetzt werden.

5.3.4.3 Dateien mit PAM-Format bearbeiten

Auch Dateien, die nicht das BS2000-Dumpformat haben, können (mit gewissen Einschränkungen) analysiert werden. Diese Funktion ist hauptsächlich für „Notauswertung“ von beschädigten Dumpfiles gedacht.

Mit `OPEN-DIAGNOSIS-OBJECT filename(KIND-OF-OBJECT=*PAM)` lässt sich eine beliebige Plattendatei im PAM-Format öffnen. Die Datei selbst kann natürlich auch eine SAM- oder ISAM-Datei sein.

Bei der Bearbeitung sind alle DAMP-Funktionen möglich, die keine BS2000-Struktur der Diagnosedatei voraussetzen, d.h., möglich ist

- die Ausgabe von PAM-Seiten der Datei in den üblichen Formaten (D, H, C, ...) auf verschiedene Dumpfenster,
- das Zuweisen einer beliebigen Symboldatei (per Hand) zur symbolischen Aufbereitung der Ausgabe,
- die selektive Stringsuche (`START-PATTERN-SEARCH`) mit Wildcards, wobei der Suchbereich eingeschränkt werden kann,
- die Ausgabe von aufbereiteten PAM-Seitenbereichen auf SYSLST,
- die Anwendung von Prozeduren, die in der Diagnosesprache PRODAMP geschrieben sind.

Bei der Adressierung gibt es in PAM-Dateien folgende Unterschiede gegenüber BS2000-Dateien:

- PAM-Seitennummern

  PAM-Seitennummern ersetzen die für BS2000-Dumps übliche modulrelative Adressierung. Die PAM-Seite wird in der Form P-XXXXXXXX (Seitennummer sedezimal) ab Spalte 1 der Kopfzeile eines Dumpfensters eingegeben. Die erste Seite der Datei ist die Seite P-00000001.
- Absolute Adressen

  Bei absoluter Adressierung wird die gesamte Datei als unstrukturierter „Stream“ von Bytes aufgefasst. Die absolute Adresse nummeriert innerhalb dieses Streams die Bytes (mit 0 beginnend) durch. Die absolute Adresse des ersten Bytes von Seite P ergibt sich also aus A = (P - 1) * 2048. Absolute Adressen können in der Spalte 40 der Kopfzeile eines Dumpfensters eingegeben werden.

Beim Markieren von Adressfeldern wird die Adressierung über PAM-Seiten-Nummern benutzt, d.h. die rechten drei Bytes des markierten Worts werden als Seiten-Nummer interpretiert und diese Seite wird dem entsprechenden Fenster zugewiesen. Dies entspricht der Methode, z.B. bei Dumpfiles Verkettungen über PAM-Seiten zu repräsentieren.

Bei der Ausgabe von Bereichen auf SYSLST werden ebenfalls nur vollständige PAM-Seiten ausgegeben. Im LIST-Fenster müssen daher Seitennummern (ohne „P-“) angegeben werden.

Bei der Stringsuche kann auf zwei Arten gesucht werden:

- unter Angabe einer Seitennummer nur auf der angegebenen Seite
- unter Angabe der Grenzen als absolute Adressen in einem Bereich.

> **i** Bei großen Dateien kann die Länge eines Datenabschnittes die 4-GB-Grenze überschreiten. Da die in DAMP verwendeten absoluten Adressen nicht größer werden können, wird intern eine Segmentnummer mitgeführt, mit der die 4-GB-Segmente unterschieden werden. Bei Angabe einer PAM-Seiten-Nummer wird automatisch auf das richtige Segment umgeschaltet. Die absoluten Adressen verstehen sich dann relativ zum Anfang des Segments. Bei `START-PATTERN-SEARCH` kann eine Segmentnummer für die Suche explizit angegeben werden.

Das Stack-Fenster (W3) und die meisten funktionsgebundenen Fenster von DAMP setzen voraus, dass das Diagnoseobjekt eine BS2000-Struktur hat. Für die Bearbeitung von Dateien im PAM-Format kommen diese Fenster daher nicht in Betracht.

Das Status-Fenster (W2) enthält bei der Bearbeitung von PAM-Dateien Informationen über die geöffnete Datei selbst, so z.B. Dateigröße und Last-Page-Pointer.

5.3.4.4 Bearbeitung von SLEDs ohne BS2000-Struktur

Mit DAMP können auch SLEDs verarbeitet werden, die von einem anderen Betriebssystem als BS2000 stammen (z. B. von IPL, BOOT, STARTUP oder von SLED).

Eine beliebige Dumpdatei kann bewusst ohne virtuelle Adressierung mit der Anweisung `OPEN-DIAGNOSIS-OBJECT <filename> (KIND-OF-OBJECT=*SELF-LOADER)` geöffnet werden.

Für die Bearbeitung der als SELF-LOADER geöffneten Dumps bietet DAMP keine automatische Aufbereitung. Alle Adressen werden als reale Adressen interpretiert. Die Bereiche des Hauptspeichers können nur über reale Adressen angesprochen werden.

Es sind folgende Funktionen möglich:

- Ausgabe der Speicherseiten in den üblichen Formaten (D, H, C, ...) auf verschiedene Dumpfenster.
- Zuweisen einer beliebigen Symboldatei (per Hand) zur symbolischen Aufbereitung der Ausgabe.
- Selektive Stringsuche (FIND-Funktion) mit Wildcards, mit der Möglichkeit den Suchbereich einzuschränken.
- Ausgabe von Seitenbereichen auf SYSLST im üblichen DAMP-Layout.
- Anwendung von PRODAMP-Prozeduren, durch die es möglich ist, SLEDs ohne BS2000-Struktur komfortabel auszuwerten.

5.3.4.5 Private Symbolelemente verwenden

Standardmäßig wird mit DAMP die Bibliothek SYSSMB.DAMP.<ver> ausgeliefert und in die Bibliothek $TSOS.SYSSMB.DAMP eingemischt. Sie enthält die von DAMP benötigten und weitere häufig benutzte DSECTs. Eine Übersicht über die DSECTs erhalten Sie im Abschnitt „Liste der DSECTs aus den Standard-Symboldateien".

Für Spezialdiagnosen können zusätzlich Symbolelemente generiert, erweitert oder geändert und anschließend für die Diagnose zugewiesen werden, zum Beispiel

- DSECT-Tabellen für DCM
- DSECTs für Datenstrukturen, die in einem TU-Programm verwendet werden (zur Auswertung von Userdumps dieses Programms).

Beim Öffnen des Diagnoseobjekts wird von DAMP automatisch das entsprechende BS2000-Systemversions-Symbolelement zugewiesen.
Dieser Automatismus lässt sich ausschalten, indem bei der Anweisung OPEN-DIAGNOSIS-OBJECT explizit ein Symbolelement angegeben wird, das für die Verarbeitung des zu öffnenden Objekts herangezogen wird. Es ist jedoch zu beachten, dass die BS2000-Standard-Symbole in diesem Element enthalten sein müssen.

Weitere Symbolelemente können mit der Anweisung ADD-SYMBOLS zugewiesen werden. Bei Angabe einer DSECT werden dann alle zugewiesenen Symbolelemente durchsucht, wobei mit dem zuletzt eingegebenen Symbolelement begonnen wird.

Beim Wechsel des Objekts werden alle Zuweisungen zurückgesetzt.

Private Symbolelemente erzeugen

Ein privates Symbolelement erzeugen Sie folgendermaßen:

- Assemblieren der zusätzlichen oder geänderten DSECTs mit TEST-SUPPORT=*AID bzw. mit *COMOPT ISD (nach der letzten DSECT sollte eine Dummy-CSECT in den Source aufgenommen werden, da der Assembler ansonsten eine falsche Länge für die letzte DSECT berechnet).

  Bei SPL-Models sollte für den Compiler die Option *COMOPT SYMTEST=ALL angegeben werden.

  Ist bereits ein Modul mit Symbolinformationen vorhanden, muss nicht neu übersetzt werden.

  C-Strukturen müssen mit TEST-SUPPORT=YES übersetzt werden. Für jedes zu generierende Symbol (=Typ) muss ein Pointer definiert werden, da der C-Compiler nur den Namen der Variablen ablegt. Strukturen und Arrays werden auf diese Weise unterstützt. Die Pointer sollten in der gleichen Reihenfolge wie die Strukturen, auf die sie zeigen, definiert werden. Nur so können die Bezüge zwischen den Strukturen und den Pointern ausgewertet werden, zudem wird so Speicher gespart. DAMP unterscheidet beim Suchen nach einem Symbol nicht zwischen Groß- und Kleinschreibung, sodass sich die Namen von Hauptstrukturen (DSECTs) nicht nur durch die Groß- und Kleinschreibung unterscheiden dürfen. Deshalb prüft der Symbolgenerator die generierten Hauptstrukturen auf Eindeutigkeit in diesem Sinn ab, behält die jeweils erste und eliminiert die nächsten.
- Nach dem Aufruf von /START-DAMP-SYMBOL-GENERATOR wird im Dialog abgefragt, ob Symbole generiert werden sollen (Eingabe: „g“) oder ob Symbolinformationen ausgegeben werden sollen (Eingabe: „i“). Bei Eingabe von „g“ lautet die nächste Abfrage, welches Modul aus welcher Bibliothek die Symbolinformationen enthält.
  Diese Symbolinformationen werden dann als Element vom Typ X einer PLAM-Bibliothek abgelegt. Der Name der Bibliothek und des Elements werden im Dialog abgefragt.

Die Bibliothek für Standard-BS2000-Symbole hat den festen Namen `$TSOS.SYSSMB.DAMP`. Der Elementname ist gleich dem Namen des Produkts, auf das sich die Symbole beziehen, ebenso geht die Elementversion aus der Version des Produkts hervor (z.B.: BS2000/210 für BS2000 V21.0A).

Ist eine Symbolbibliothek gleichen Namens unter der aktiven Kennung bereits vorhanden, können die neu übersetzten DSECTs in die bestehende Bibliothek eingefügt werden. Ist ein Symbolelement gleichen Namens in der angegebenen Bibliothek bereits vorhanden, kann das Element wahlweise ersetzt werden oder durch die neuen Informationen ergänzt werden.

- Ggf. Kopieren der Symbolbibliothek oder des Symbolelements unter die gewünschte Benutzerkennung oder in die gewünschte Bibliothek und Einstellen der Benutzeroption SYSSMB (siehe "Modifikationen durch den Benutzer (Spezialfenster OPTIONS)").

Anhand zweier Beispiele soll die Verwendung von `/START-DAMP-SYMBOL-GENERATOR` erläutert werden.

Beispiel 1

```
/start-damp-symbol-generator
You wish to :  - Generate symbols ?                              -->  g
               - Get information about symbols ?                 -->  i
*g
  Creation of a DAMP-Symbolfile.
  Please give name of :    - library with object module or
                           - old symbol file for conversion
*my.object.lib
  Please enter name and type of object module
  (e. g. 'MODNAME/R' [type R is default])
  In C it is the R-element with "@" as termination.
  In SPL it is the 8 B long R-element with "@" as termination.
*dmpbs2a/r
  Element DMPBS2A/@/R
  from library MY.OBJECT.LIB successfully opened.
  Symbolic information will be taken from LSD-cards.
  The symbol information is from BS2000 V210, PVLU E1.
  Proposal :  The element BS2000/210.E1
              will be generated in the library
              SYSSMB.DAMP
  Please enter one of the following answers :
  Y[ES]      -> You accept the proposal.
  L[IBR]     -> You will further be asked for the name
                of the output library where the element
                BS2000/210.E1
                will be generated.
  U[SER]     -> By user, you will further be asked
                for the name of the output library and
                for the name and version of the element.
  P[ROD]/N[O]-> You will further be asked for the name,
                the version and the PVLU of the product.
                Lib : SYSSMB.<prod>.<vers>.<PVLU>
                El : <prod>/<vers>.<PVLU>
  I[NPUT]    -> Output library/element/version =
```

```
                Input library/element/version.
                Output element type = X.
*u
  Please enter valid names !!!
  1. -> Enter library name :
*my.symbol.lib
  2. -> Enter element name :
*my_element
  3. -> Enter element version :
*210
  Output Symbol Library : MY.SYMBOL.LIB
  Output Symbol Element : MY_ELEMENT/210
  Element MY_ELEMENT/210 from
  library MY.SYMBOL.LIB successfully opened.
  Starting to write symbol element.
  There are to be generated 75 structures.
  Symbol element written.
  There have been written 75 structures into the file.
  Program terminated normally.
```

Beispiel 2

Mit `/START-DAMP-SYMBOL-GENERATOR` kann man sich auch die in Symbolelementen abgelegten DSECTs, Strukturen und Symbole auf dem Bildschirm oder in einer Datei auflisten lassen.

```
/start-damp-symbol-generator
  You wish to :  - Generate symbols ?                          -->  g
                 - Get information about symbols ?             -->  i
*i
Please enter the library name :
*syssmb.damp.<ver>
You wish to :  - list the library elements and the contained symbols
                 on screen ?                                   -->  s
               - write the names of the DSECTs into a file ?   -->  n
               - write a DSECT, converted to a Pascal-Record,
                 into a file ?                                 -->  r
               - search for a symbol with wildcards ?          -->  w
               - write the alphabetical list of symbols into EDT-->  a
               - show the global info of a symbol element      -->  g
               - go to EDT ?                                   -->  @
               - assign a new library ?                        -->  l
               - terminate the program ?                       -->  e
*s
The library SYSSMB.DAMP.<ver> contains the following elements :
BS2000/190        BS2000/200        BS2000/210
BS2000-USER/190   BS2000-USER/200   BS2000-USER/210
NSDI0/190         NSDI0/200         NSDI0/210
STATUS/006        STATUS/010        XA2000/190
XA2000-USER/190   XA2000/200        XA2000-USER/200
```

```
XA2000/210        XA2000-USER/210
You wish to :  - list the symbols    --> name/version
               - stop this function  --> *e, *end
*BS2000/210
   ASAVDSSM                           ASIMDBHD
   ASIPUCON                           BS_CTX_VECTOR_REC_MDL
   CTX_VECTOR_REC_MDL                 DBL_OPTIONS_COM_MDL
   DBL_OPTIONS_P_C_MDL                DBL_OPTIONS_S_P_MDL
   DSTE                               DWQE
   DWQH                               EBWL
   ECSA                               ECSE
   ECSX                               ECTLP
...
You wish to :  - list the library elements and the contained symbols
                 on screen ?                                    -->  s
               - write the names of the DSECTs into a file ?    -->  n
               - write a DSECT, converted to a Pascal-Record,
                 into a file ?                                  -->  r
               - search for a symbol with wildcards ?           -->  w
               - write the alphabetical list of symbols into EDT-->  a
               - show the global info of a symbol element       -->  g
               - go to EDT ?                                    -->  @
               - assign a new library ?                         -->  l
               - terminate the program ?                        -->  e
*e
Program terminated normally.
```

5.3.4.6 Private Assembler-Benutzerroutinen schreiben

Bei speziellen Anforderungen können Sie eigene Benutzerroutinen zur Listenaufbereitung oder zur Spezialauswertung erzeugen und über die Anweisungen LOAD-MODULE und START-MODULE von DAMP aus aufrufen. Damit machen Sie sich allerdings abhängig von der Struktur der Dumpdatei und der zu Grunde liegenden BS2000-Version. Mit der Diagnosesprache PRODAMP gibt es diese Abhängigkeiten nicht. Innerhalb von PRODAMP können Assemblerroutinen mithilfe der PRODAMP-Funktion ENTER-MODULE aufgerufen werden, mit dem Vorteil, dass Diagnosedaten als Parameter übergeben werden können.

Beim Schreiben privater Benutzerroutinen sind folgende Schnittstellen einzuhalten:

Register 1	enthält die Adresse des Parameterstrings (max. 80 Zeichen), der bei der Anweisung START-MODULE angegeben werden kann.
Register 13	enthält die Adresse einer 18-Wort-Save-Area, die von DAMP zur Verfügung gestellt wird und nach VMOS-Konventionen benutzt werden kann.
Register 14	enthält die Rückkehradresse
Register 15	enthält die Ansprungadresse

Alle Register sind vor Rückkehr nach DAMP auf ihren ursprünglichen Wert zurückzusetzen.

Das gerufene Modul darf keinerlei Voraussetzungen über den Zustand des Diagnoseobjekts machen, das zu diesem Zeitpunkt unter DAMP in Bearbeitung ist. DAMP bietet auch keinerlei Schnittstellen an, die von externen Routinen genutzt werden können. Ferner werden die externen Prozeduren im 31-Bit-Modus aufgerufen, sodass sie zumindest einen 31-Bit-fähigen Adapter enthalten müssen. Der Benutzermodul kann in eine beliebige Modulbibliothek eingetragen werden. Vor dem Aufruf der Routine mit START-MODULE muss das Modul zunächst mit der Anweisung LOAD-MODULE nachgeladen werden.

Ist das Benutzermodul in der Nachlade-Bibliothek von DAMP enthalten, kann die Anweisung LOAD-MODULE entfallen.

Bei Problemen aus dem Bereich der Datenfernverarbeitung kann man die Benutzerroutine DCM anstoßen, womit man Aufbereitungen der DCM-Tabellen erhält.

5.4 Listen erzeugen und ausdrucken (Spezialfenster LIST)

Trotz komfortabler Diagnose am Bildschirm ist es mitunter sinnvoll, sich Daten aus einem Dump auf Papier auszudrucken. Hat man etwa mit PRODAMP Spezialauswertungen durchgeführt und Strukturen, die im Speicher verteilt sind, übersichtlich zusammengefasst, können Sie sich anschließend die Ergebnisse beispielsweise auf einem Drucker des lokalen Rechners oder an einem externen Rechner auf Papier ausgeben lassen. Natürlich lässt sich auch ein kompletter Speicherabzug auf Papier ausgeben. Angesichts des Diagnosekomforts, den DAMP bietet, dürfte ein bis zu 1,5 m hoher Papierstapel aber wohl eher nur noch Demonstrationswert haben. Die Aufbereitung der Ausgabe kann entweder im Dialog oder über Batch-Anweisungen gesteuert werden.

5.4.1 Steuerung der Listenausgabe im Dialogbetrieb

Nach dem Eingeben der Anweisung `START-LIST-GENERATION` erscheint die List-Maske im letzten freien Diagnosefenster. War schon eine Dumpdatei geöffnet, so ist das Feld **Dumpfile** mit dem Namen der geöffneten Datei vorbesetzt. War keine Dumpdatei geöffnet oder soll eine andere als die geöffnete Datei für den Druck aufbereitet werden, ist zunächst eine Datei auszuwählen (siehe "Datei auswählen"). Umfang und Inhalt der zu druckenden Liste wird anschließend durch das Markieren bzw. Ausfüllen der verschiedenen Felder der Maske bestimmt. Bei der Bearbeitung eines Areadumps ist die Angabe von Auswahlkriterien wirkungslos, da nur die zum Areadump gehörenden Standardtabellen und die angeforderten Bereiche aufbereitet werden.

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


LIST - Command                                  SYS=0                 W8,LST,L19

Dumpfile = :SLED:$DUMPFILE.SLED.CS507K
Listfile = *SYSLST                                                REMOTE: YES/NO

FUNCTION : OPN/LST/LSTALL/RESET           SELECT : INF/SYS/MIN/ALL OR
           ------------------------------------------------------------------
SELECT FROM |TRACES: ALL   STT   TM    NO                                      |
            |MAPS  : ALL   CS    CRI   NO                                      |
            |TABLES: ALL   XVT   TCB   PCB     SPL        TFT        AUD     NO|
            |MEMORY: ALL   CL1   CL2   CL3     CL4        CL5        CL6     NO|
            |                          PP FP   PP FP NP   PP FP MP   FP MP     |
            |MODULE:                                                           |
           ------------------------------------------------------------------
            |PAGES         FROM:       TO:                                     |
           ------------------------------------------------------------------
WINDOW:
DIAG: YES/NO   DESCR: YES/NO              PROC:

CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=LIST   9=Dump
```

Bild 57: Die List-Maske

Während und nach dem Abarbeiten des Druckauftrages bleibt die List-Maske auf dem Bildschirm. Durch den Druckauftrag wird die letzte bearbeitete Dumpdatei nicht geschlossen und kann weiter bearbeitet werden. Durch den Wechsel in ein anderes Diagnosefenster wird die LIST-Verarbeitung unterbrochen. Eine Rückkehr in die List-Maske wird durch Drücken der entsprechenden P-Taste oder durch die erneute Anweisung `START-LIST-GENERATION` veranlasst.

Wird keine weitere Druckaufbereitung mehr gewünscht, kann das Fenster, das mit der List-Maske belegt ist, durch die Anweisung `SHOW-EDITED-INFORMATION *STORAGE-EDIT` wieder für andere Ausgaben frei gemacht werden.

Die List-Maske erscheint nur im Dialogbetrieb auf dem Bildschirm. Die jeweils gültigen Einstellungen sind hellgesteuert hervorgehoben, d.h., nach dem Markieren und anschließendem Druck auf die Taste DUE wird das markierte Feld hell, die ggf. verbleibenden Alternativfelder (wie bei SYS/MIN/ALL) in normaler Helligkeit angezeigt.

5.4.1.1 Datei auswählen

Die gewünschte Datei wird durch Eintragen eines voll- oder teilqualifizierten Dateinamens in das Feld „Dumpfile“ der List-Maske ausgewählt. Dabei ist innerhalb des Dateinamens die Benutzung von Wildcards erlaubt. Zusätzlich wird der String „$TSN“ innerhalb des Dateinamens durch die TSN der Aufrufer-Task ersetzt. Letzteres erleichtert besonders das Auffinden aktuell erzeugter Benutzer-Dumps.

Bild 58: Feld für die Datei-Auswahl

Das Absenden des veränderten Fensters mit DUE führt zum Aufbau einer internen Liste von Dateinamen und zur Anzeige des ersten passenden Dateinamens in der List-Maske. Innerhalb der Dateinamen-Liste kann mit F3 / + bzw. F1 / - vorwärts und rückwärts geblättert werden, bis die gewünschte(n) Dumpdatei(en) gefunden ist (sind).

Die Datei-Auswahl selbst hat keine Auswirkung auf die evtl. gerade geöffnete Dumpdatei. Erst das Markieren einer der Funktionen OPN, LST, LSTALL führt zum Schließen der geöffneten Dumpdatei und zum Öffnen der ausgewählten Datei(en).

5.4.1.2 Ausgabeort der Liste auswählen

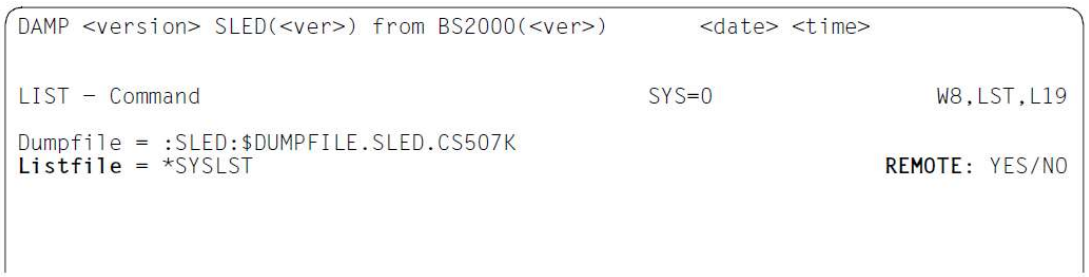


Bild 59: Felder zur Bestimmung des Ausgabeorts der erzeugten Liste

Zur Angabe des Ausgabeorts einer erzeugten Liste sind der Schalter REMOTE und das Feld „Listfile“ vorgesehen. Standardmäßig wird die Liste auf SYSLST am eigenen Rechner ausgegeben.

Durch Markieren des Schalters REMOTE kann die Ausgabedatei mit File-Transfer an einen fremden Rechner übergeben werden.

YES Ist YES markiert, werden neue Eingabefelder in den Zeilen 7 und 8 angeboten, in die Optionen für den File-Transfer angegeben werden müssen (siehe Funktion auswählen). Es gibt keine Vorbelegung für diese Optionen.

Der Name der Ausgabedatei, die per File-Transfer verschickt werden soll, wird automatisch von DAMP generiert (SYSLST.DAMP.<ver>.<date>.<time>). Der Name enthält den aktuellen Zeitstempel, damit keine Datei auf der Zielkennung zerstört wird.

Im Feld „Partner“ muss der Zielrechner eingetragen werden. Die File-Transfer-Berechtigung kann über den Namen eines FTAC-Profils oder explizit durch Benutzerkennung, Abrechnungsnummer und Passwort angegeben werden.

NO Ist NO (Voreinstellung) markiert, wird die Liste am eigenen Rechner ausgegeben. Dafür kann im Feld „Listfile“ der Name der Ausgabedatei eingetragen werden.

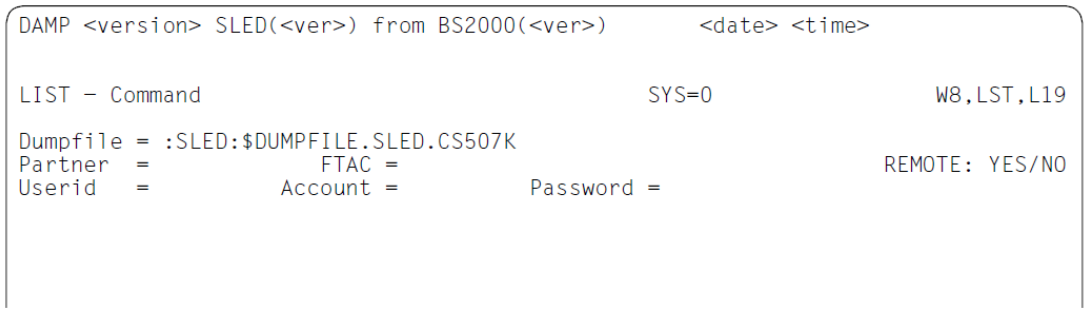


Bild 60: Optionen für den File-Transfer (nach dem Markieren von REMOTE: YES)

5.4.1.3 Funktion auswählen

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>

LIST - Command                                  SYS=0                W8,LST,L19

Dumpfile = :SLED:$DUMPFILE.SLED.CS507K
Listfile = *SYSLST                                                REMOTE: YES/NO

FUNCTION : OPN/LST/LSTALL/RESET           SELECT : INF/SYS/MIN/ALL OR
```

Bild 61: Die Möglichkeiten der Funktionsauswahl

Das Markieren einer der unter FUNCTION aufgeführten Alternativen und Übertragen mit DUE bestimmt, was mit der gerade ausgewählten Dumpdatei geschehen soll:

OPN
: Die bei Dumpfile angezeigte Dumpdatei wird geöffnet. Vorher wird eine eventuell noch offene Dumpdatei geschlossen. Dadurch kann man sich vor dem Auslösen der Listenausgabe im Dialog noch einmal davon überzeugen, dass man die richtige Dumpdatei ausgewählt hat. Das Feld OPN kann auch generell zum Öffnen von Dumpdateien benutzt werden (als Alternative zur Anweisung `OPEN-DIAGNOSIS-OBJECT`).

  Bei Dumpdateien mit mehreren Objekten (VM2000-Gesamtsled, SLED vom SLED / Dump vom SLED) wird hier - im Gegensatz zum Öffnen mit `OPEN-DIAGNOSIS-OBJECT` im Dialog - das auszuwertende Objekt automatisch ausgewählt.

LST
: Die im Feld „Dumpfile“ angezeigte Dumpdatei wird ausgedruckt. Dabei werden alle in der List-Maske hellgesteuerten sowie alle markierten Parameter wirksam.

LSTALL
: Alle in der Dateiliste enthaltenen Dumpdateien (mit BS2000-Objekt) werden ausgedruckt. Dabei werden für alle Dateien die in der List-Maske hellgesteuerten und die markierten Parameter wirksam.

RESET
: alle Parameter der List-Maske werden auf die Voreinstellungswerte zurückgesetzt. Gleichzeitig wird die ggf. vorhandene Liste von Dateinamen gelöscht.

> **i** Die Funktionen OPN, LST, und LSTALL schließen eine bereits geöffnete Dumpdatei, wenn sich die Ausgabe nicht auf diese Datei bezieht.

5.4.1.4 Task auswählen

Die Auswahl einer Task in einem SLED bzw. SNAP erfolgt durch Überschreiben des Feldes „SYS“ in der LIST-Maske. Hier können die erlaubten ASEL-ASID-Kombinationen eingetragen werden (siehe "Eingabefelder der Standard-Dumpfenster (W4 - W9 und W21 - W99)").
Bei Angabe einer Task werden nur die taskspezifischen Bereiche dieser Task ausgegeben (als ob von dieser Task ein Systemdump gezogen worden wäre).

In der LIST-Maske ist bei einer TSN-Angabe auch die Eingabe der Schlüsselwörter ***ALL** oder ***ERR** möglich.

*ALL	Es wird bei einer SLED- oder SNAP-Datei eine Task-Aufbereitung für alle aktiven Tasks veranlasst, während ansonsten standardmäßig nur die Bereiche der von DAMP ausgewählten Fehlertasks ausgegeben werden.
*ERR	Es wird die Suche nach der möglichen Error-Task mittels der Vordiagnose-Routine DIAG veranlasst. Diese Eingabe entspricht dem Markieren von YES im Feld DIAG (siehe "Felder für Vordiagnose und Fehlerdeskriptoren").

5.4.1.5 Listenumfang festlegen

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


LIST - Command                                  SYS=0                 W8,LST,L19

Dumpfile = :SLED:$DUMPFILE.SLED.CS507K
Listfile = *SYSLST                                                REMOTE: YES/NO

FUNCTION : OPN/LST/LSTALL/RESET           SELECT : INF/SYS/MIN/ALL OR
           ------------------------------------------------------------------
SELECT FROM |TRACES: ALL   STT   TM    NO                                     |
            |MAPS  : ALL   CS    CRI   NO                                     |
            |TABLES: ALL   XVT   TCB   PCB     SPL         TFT         AUD     NO|
            |MEMORY: ALL   CL1   CL2   CL3     CL4         CL5         CL6     NO|
            |                          PP FP   PP FP NP    PP FP MP    FP MP    |
            |MODULE:                                                            |
            ------------------------------------------------------------------
            |PAGES         FROM:       TO:                                      |
            ------------------------------------------------------------------
WINDOW:
DIAG: YES/NO   DESCR: YES/NO              PROC:

CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=LIST   9=Dump
```

Bild 62: Die globalen Auswahlparameter für den Listenumfang

Der Umfang der Liste wird durch das Markieren der SELECT-Felder festgelegt. Dabei können Sie zwischen vier globalen Ausgabeformen (**INF/SYS/MIN/ALL**) und einer gezielten Ausgabe anzugebender Bereiche (**SELECT FROM**) wählen. Im ersten Fall entscheidet DAMP, im zweiten Fall Sie über die Auswahl der auszugebenden Bereiche. Um diese Unterscheidung augenfällig zu machen, befinden sich die nur in Verbindung mit **SELECT FROM** erlaubten detaillierten Auswahl-Parameter im inneren Rahmen der LIST-Maske. Jede Eingabe oder jedes Markieren in diesem Bereich führt automatisch zur Umschaltung auf SELECT FROM.

Die Parameter für die globale Ausgabe haben folgende Bedeutung:

INF
: Die Informationen, die im Status-Fenster (W2) im Modus INF aufbereitet wurden, werden ausgegeben.

SYS
: Es wird nur die Systemübersicht ausgegeben. Je nach Einstellung der Parameter **DIAG** und **DESCR** gehört dazu auch die Ausgabe der Fehlerdeskriptoren (siehe "Felder für Vordiagnose und Fehlerdeskriptoren").

MIN
: Es wird eine Minimum-Liste ausgegeben (siehe "Bestandteile und Umfang der ausgegebenen Listen"). Diese Liste enthält in der Regel alle für eine erste Diagnose erforderlichen Daten. Falls erforderlich, können Sie mit weiteren gezielten Listen (SELECT FROM) die Lücken schließen. Voreinstellung.

ALL
: Es wird eine vollständige Liste ausgegeben (siehe "Bestandteile und Umfang der ausgegebenen Listen"), d.h. bei System- und Benutzer-Dumps werden alle von CDUMP sichergestellten Seiten auf der Liste ausgegeben.

> ℹ Die komplette Liste eines Systemdumps ergibt einen Papierstapel von ca. 1,5m Höhe.

Ansonsten werden nur die explizit ausgewählten Bereiche innerhalb der Umrahmung ausgegeben. Die Erläuterung der Optionen finden Sie in Abschnitt „Einzelne Ausgabebereiche auswählen".

5.4.1.6 Einzelne Ausgabebereiche auswählen

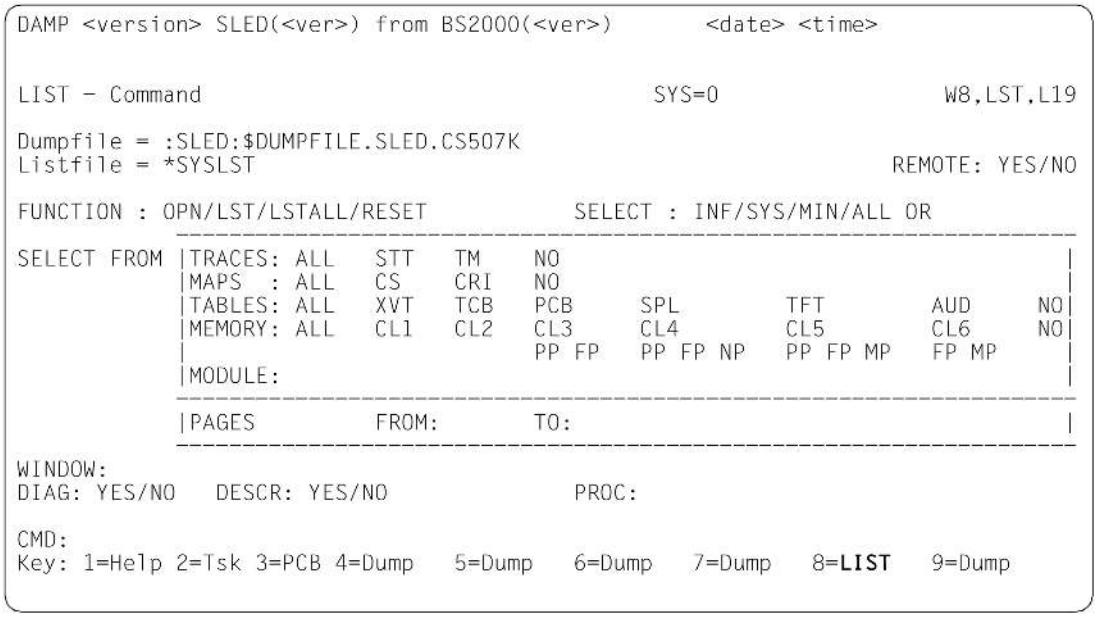


Bild 63: Einzelne markierbare Ausgabebereiche

Die durch Umrahmung hervorgehobenen Parameter dienen zur expliziten Auswahl einzelner Bereiche, die ausgegeben werden sollen. Jede Eingabe oder Markierung innerhalb dieses Rahmens führt automatisch zur Umschaltung auf SELECT FROM. Umgekehrt macht jedes Markieren eines Parameters außerhalb des SELECT-Rahmens die Einstellungen innerhalb dieses Rahmens unwirksam. Die Einstellungen werden aber nicht „vergessen “, sondern können durch Markieren von SELECT FROM wieder aktiviert werden.

Ein Teil der Felder wird durch Markierung oder Eintragungen für Speicherbereichsangaben benutzt. Für diese so genannten Memory-Optionen muss bei der Auswertung von SLED- bzw. SNAP-Dateien und Angabe von task-lokalem Speicher eine Task-Auswahl getroffen werden.

Folgende Optionen stehen zur Auswahl:

TRACES legt fest, welche System-Traces ausgegeben werden sollen:

ALL	alle Traces
STT	die System-Trace-Table
TM	die Übersicht der vom Trace-Manager verwalteten Traces
NO	kein Trace

MAPS legt fest, welche Information zur Lokalisierung von Adressen und Korrekturen ausgegeben werden sollen:

ALL	die CSECT-Maps und die Rep-Informationen
CS	nur die CSECT-Maps
CRI	nur der Inhalt der SAVEREP- bzw. REPLOG-Section
NO	keine Maps

TABLES wählt die auszugebenden System-Tabellen aus.
Die Ausgabe erscheint dabei im gleichen Format, wie bei der impliziten Ausgabe von System-Tabellen während einer Minimum- oder Voll-Listenausgabe. Die gewünschten Tabellen wählen Sie durch Markieren aus. Markieren Sie bei einer SLED- bzw. SNAP-Datei Task-lokale Tabellen (TCB, PCB etc.), muss eine Task-Auswahl getroffen werden (siehe Abschnitt „Task auswählen").

MEMORY grenzt die Speicherbereiche, die in der Liste im Dump-Format ausgegeben werden sollen, genau ein. Während die Standard-Listenausgabe
(= Minimum) nur Speicherseiten ausgibt, die durch Mehrzweckregister aus PCBs oder Save-Areas referenziert werden, können hier zusammenhängende Speicherbereiche ausgegeben werden.
Die Kürzel der auszugebenden Speicherbereiche (z.B. CL1) müssen dazu lediglich markiert und anschließend mit DUE übertragen werden. Die
Kürzel werden genauso interpretiert wie bei der Suche von Zeichenketten (siehe "Zeichenketten suchen (Spezialfenster FIND)").

Für alle MEMORY-Parameter gilt, dass bei Auswertung von SLED- bzw. SNAP-Dateien und Angabe von task-lokalem Speicher gleichzeitig eine Task-Auswahl getroffen werden muss (siehe Abschnitt „Task auswählen").

Weitere Memory-Optionen sind:

PAGE FROM legt eine Seite als Untergrenze für den auszugebenden Speicherbereich fest. Dabei wird für reale /absolute Adressen (ASEL=RM/ABS) der Inhalt des Eingabefeldes ASID berücksichtigt.

PAGE TO legt eine Seite als Obergrenze für den auszugebenden Speicherbereich fest. Dabei wird für reale /absolute Adressen der Inhalt des Eingabefeldes ASID berücksichtigt. Seitennummern (bis zu 8 Sedezimalziffern) liegen im Bereich X'0' bis X'1FE00000'.

WINDOW gibt ein Standard-Dumpfenster an, dessen Inhalt, ein Speicherbereich, ausgegeben werden soll. Zum Listen eines Fensters muss dieses gefüllt und daher ein Dump geöffnet sein; LSTALL darf in Verbindung mit dieser Option nicht verwendet werden. Bei symbolischer Ausgabe wird bis DSECT-Ende ausgegeben, bei den anderen Ausgabeformaten bis Seitengrenze. Eventuell muss die Ausgabe nach dem Weiterschalten der Ausgabeadresse nochmals wiederholt werden.
Folgende Fenster-Formate können ausgegeben werden:

- Speicherbereiche im Dump-, Sedezimal- und Zeichen-Format
- Speicherbereiche im Assembler-Format (disassembliert)
- Ausgabe in Real- und Absolut-Adressierung
- symbolische Ausgabe
- Hardware-Informationen

MODULE spezifiziert ein Modul, das in seiner gesamten Länge ausgegeben werden soll.

5.4.1.7 Felder für Vordiagnose und Fehlerdeskriptoren

```
DAMP <version> SLED(<ver>) from BS2000(<ver>)       <date> <time>


LIST - Command                                  SYS=0                 W8,LST,L19

Dumpfile = :SLED:$DUMPFILE.SLED.CS507K
Listfile = *SYSLST                                                REMOTE: YES/NO

FUNCTION : OPN/LST/LSTALL/RESET           SELECT : INF/SYS/MIN/ALL OR
           ------------------------------------------------------------------
SELECT FROM |TRACES: ALL   STT   TM    NO                                     |
            |MAPS  : ALL   CS    CRI   NO                                     |
            |TABLES: ALL   XVT   TCB   PCB     SPL         TFT         AUD     NO|
            |MEMORY: ALL   CL1   CL2   CL3     CL4         CL5         CL6     NO|
            |                          PP FP   PP FP NP    PP FP MP    FP MP    |
            |MODULE:                                                            |
            ------------------------------------------------------------------
            |PAGES         FROM:       TO:                                      |
            ------------------------------------------------------------------

WINDOW:
DIAG: YES/NO   DESCR: YES/NO              PROC:

CMD:
Key: 1=Help 2=Tsk 3=PCB 4=Dump   5=Dump   6=Dump   7=Dump   8=LIST   9=Dump
```

Bild 64: Felder für die Listenausgabe markieren

Weitere Parameter der Listenausgabe werden in den Feldern DIAG und DESCR festgelegt. Es steht jeweils der Wert YES oder NO zur Verfügung.

DIAG Der Standardwert für dieses Feld ist NO.
Ist YES markiert, wird die automatische Vordiagnose angestoßen. Durch die Vordiagnose werden eine Anzahl von Speicherseiten markiert, die bei einer nachfolgenden Minimum-Listenausgabe mit ausgegeben werden. Die automatische Vordiagnose wird für Benutzerdumps noch nicht unterstützt.

DESCR Der Standardwert für dieses Feld ist NO.
Ist YES markiert, wird in der Systemübersicht eine Liste von Fehler-Deskriptoren ausgegeben. DESCR=YES setzt DIAG=YES voraus. Umgekehrt kann eine automatische Vordiagnose ohne Ausgabe von Deskriptoren sinnvoll sein, denn durch die Vordiagnose wird eine Anzahl von Speicherseiten referenziert, die bei einer nachfolgenden Minimum-Auswertung dann automatisch mit ausgegeben werden.

Nicht erlaubte Kombinationen der Felder DIAG und DESCR werden automatisch umgesetzt.

5.4.1.8 PRODAMP-Prozeduren oder Aufbereitungsprogramme verwenden

Unabhängig von der Bereichsauswahl kann im Feld PROC der Name einer PRODAMP-Prozedur oder eines Aufbereitungsprogramms eingetragen werden. Die Prozedur bzw. das Programm wird am Ende der Listenaufbereitung gestartet und die Listen werden zusätzlich zu den im SELECT-Rahmen gewählten erzeugt. Die Prozedur oder das Programm muss in übersetzter Form als Objekt in der Benutzer-PRODAMP-Bibliothek stehen. Die Bibliothek muss ggf. mit der Anweisung `ASSIGN-PRODAMP-LIBRARIES` eingestellt werden.

5.4.1.9 Aufbereitungsprogramm verwenden

Das Programm DMP_ANALYSE_DMS_TABLES

Auch unabhängig von der durch die automatische Voranalyse erkannten Fehlerkomponente kann durch Eintrag von *DMS in das PROC-Feld der LIST-Maske eine besondere DMS-Aufbereitung veranlasst werden. Diese enthält die taskspezifischen DMS-Tabellen (TFT, TPR-FCB, TU-FCB) der betroffenen Task.
Die Aufbereitung erfolgt über die PRODAMP-Prozedur DMP_ANALYSE_DMS_TABLES, die mit DAMP standardmäßig ausgeliefert wird.

Das Programm MEMCNTRL

Bei Problemen der Speicherplatzbelegung, Adressraum-Engpässen u.a. kann man die Auswertungen der Adressraum-Belegung durch das Programm MEMCNTRL innerhalb der DAMP-Anwendung anstoßen. Das Programm MEMCNTRL wird in der System-PRODAMP-Bibliothek ausgeliefert. Diese ist vor Aufruf des Programms mit der Anweisung `ASSIGN-PRODAMP-LIBRARIES OBJECT-LIBRARY=*PRODAMP-SYSTEM-LIBRARY` einzustellen.

Das Programm NDM

Bei Problemen aus dem Bereich des Device-Managements können innerhalb der DAMP-Anwendung Aufbereitungsprozeduren für die NDM-Tabellen angestoßen werden. Durch einen entsprechenden Eintrag in das PROC-Feld der LIST-Maske werden auf SYSLST die Auswertungen ausgegeben. Für weitere Informationen zu diesem Thema siehe auch Kapitel „NDMDAMP Erzeugung von Diagnoseunterlagen“.

Vor Aufruf des Programms NDM ist ebenso wie beim Programm MEMCNTRL die System-PRODAMP-Bibliothek als OBJECT-LIBRARY einzustellen.

> ¡ Die Ausgabe auf Liste ist in der Regel keine ausreichende Unterlage bei der Weiterleitung einer Fehlermeldung. Generell sollte zu allen Fehlermeldungen die Dump-Datei auf Datenträger mitgeliefert werden, um auch einer nachfolgenden Instanz eine Diagnose mit DAMP zu ermöglichen.

5.4.2 Steuerung der Listenausgabe im Batch- oder Prozedurbetrieb

Die Steuerung des Listenumfangs wird in drei Schritten realisiert:

1. Einleiten der Listenerzeugung mit der Anweisung START-LIST-GENERATION
2. Auswahl der Speicherbereiche mit den Anweisungen ADD-LIST-OBJECTS und REMOVE-LIST-OBJECTS
3. Starten der Listenausgabe mit PRINT-LIST

Es können mehrere ADD-LIST-OBJECTS- und REMOVE-LIST-OBJECTS-Anweisungen in beliebiger Reihenfolge zwischen START-LIST-GENERATION und PRINT-LIST eingeschoben werden. Die ausgewählten Bereiche werden aufaddiert und erst beim abschließenden PRINT-LIST wirksam.

Die Anweisung START-LIST-GENERATION enthält alle Angaben zum Eingabemedium, nämlich die Namen der auszuwertenden Dump-Dateien, mit der Anweisung PRINT-LIST wird das Ausgabemedium (SYSLST oder Datei) festgelegt.
Die Einzelheiten sind den Beschreibungen der entsprechenden Anweisungen zu entnehmen.

Jede von DAMP erzeugte Liste enthält am Ende der Auswertung eine Aufstellung der ausgewählten Optionen in Form von ADD-LIST-OBJECTS-Anweisungen. Es können also z.B. im Dialog die gewünschten Optionen durch Markieren eingestellt und die äquivalenten Anweisungen aus der erzeugten Liste entnommen werden.

Beispiele für Anweisungsfolgen

An den folgenden Beispielen soll deutlich werden, wie eine Anweisungsfolge im DAMP-Batch- oder Prozedurmodus aufgebaut sein kann.
Die Anweisungsfolgen, die mit START-LIST-GENERATION beginnen und mit PRINT-LIST schließen, könnten natürlich auch in einer Datei abgelegt sein, die schließlich beim Ablauf mit der DAMP-Anweisung START-STATEMENT-SEQUENCE zugewiesen wird.

Beispiel 1

```
/BEGIN-PROCEDURE
/ASSIGN-SYSDTA TO-FILE=*SYSCMD
/MODIFY-JOB-SWITCHES ON=5
/START-DAMP
START-LIST-GENERATION FILES-TO-EVALUATE=DUMP.HUGO ————————————————————  (1)
ADD-LIST-OBJECTS TASK-INFORMATION=*PARAMETERS(SELECT=C'RP01', -
                 INFORMATION=*PARAMETERS(CONTROL-BLOCKS=*TCB, -
                 PAGES=*INTERVAL(FROM=X'0',TO=X'FFF'))) ————————————————  (2)
ADD-LIST-OBJECTS GLOBAL-INFORMATION=*PARAMETERS(CONTROL-BLOCKS=*XVT, -
                                                MODULE=DOPEN) ——————————  (3)
PRINT-LIST OUTPUT=#AUSWERT ————————————————————————————————————————————  (4)
END
/MODIFY-JOB-SWITCHES OFF=5
/END-PROCEDURE
```

(1) Mit START-LIST-GENERATION wird die auszuwertende Dump-Datei festgelegt.
(2) Für die SRPM-Task soll der TCB und die Seiten 0 bis FFF ausgegeben werden.
(3) Die globalen Objekte XVT und das Modul DOPEN sollen ausgegeben werden.
(4) Die Ausgabe erfolgt in die temporäre Datei #AUSWERT.

Beispiel 2

```
/BEGIN-PROCEDURE
/ASSIGN-SYSDTA TO-FILE=*SYSCMD
/ADD-FILE-LINK FILE=SYSDUMP.VON.GESTERN, LINK=#1 ————————————————————  (1)
/MODIFY-JOB-SWITCHES ON=5
/START-DAMP
OPEN-DIAGNOSIS-OBJECT OBJECT=*#1 ————————————————————————————————————  (2)
START-LIST-GENERATION ———————————————————————————————————————————————  (3)
PRINT-LIST ——————————————————————————————————————————————————————————  (4)
END
/MODIFY-JOB-SWITCHES OFF=5
/END-PROCEDURE
```

(1) Mit `/ADD-FILE-LINK` wird die auszuwertende Dump-Datei zugewiesen.

(2) Durch die `OPEN-DIAGNOSIS-OBJECT`-Anweisung wird die Dump-Datei zur DAMP-Bearbeitung eröffnet.

(3) Die Listenausgabe wird eingeleitet. Eine `FILE-TO-EVALUATE`-Zuweisung (siehe Beispiel 1) kann entfallen, da die eröffnete und die zu bearbeitende Datei identisch sind.

(4) Die Ausgabe erfolgt nach SYSLST.

Die Auswahl der auszugebenden Bereiche übernimmt DAMP (Minimumausgabe).

5.4.3 Bestandteile und Umfang der ausgegebenen Listen

Bestandteile und Umfang der ausgegebenen Listings sind davon abhängig

- aus welchem Dump sie stammen. Die verschiedenen Dump-Formen, beispielsweise ein SLED-Dump und ein Area-Dump, unterscheiden sich bereits bezüglich Inhalt und Umfang voneinander. Entsprechend unterschiedlich sind auch die jeweiligen Listen.
- welche Listing-Art ausgewählt wird (Minimum oder All)

Welche Arten von Listings es gibt und aus welchen Bestandteilen sie zusammengesetzt sind, zeigt die nachfolgende Übersicht.

Übersicht über den Umfang der Minimum- und Vollauswertung

Listentyp **Bestandteile**	**Area**	**User Min**	**User All**	**Sys Min**	**Sys All**	**SLED Min**	**SLED All**	**SNAP**
Allg. Dumpdatei-Informationen	A	A	A	A	A	A	A	A
Systemübersicht	T	T	T	A	A	A	A	T
Tracemanager	-	-	-	A	A	A	A	T
Deskriptoren	-	-	-	A	A	A	A	-
CSECT-MAP privileg.	-	-	-	A	A	A	A	A
REPLOG	-	-	-	A	A	A	A	-
Trace-Table (Total)	-	A	A	A	A	A	A	A
Klasse1-Speicher	-	-	-	R	A	R	A	A
Klasse2-Speicher	-	-	-	R	A	R	A	-
Klasse3/4-Speicher	-	-	-	R	A	R	A	T
AUDIT-Tabellen (prozessorlokal)	-	-	-	A	A	A	A	-
CSECT-MAP n.priv.	-	A	A	A	A	-	-	-
Trace-Table (Task)	-	A	A	A	A	E	E	E
TCB	A	A	A	A	A	E	E	E
TFT	-	D	D	D	D	D	D	-
PCB	A	A	A	A	A	E	E	E
SPL - Stacks	-	-	-	A	A	E	E	-
Audit - Tabellen	-	A	A	A	A	E	E	-
Klasse6-Speicher	T	R	A	R	A	-	-	-
Klasse5-Speicher	T	A	A	A	E	E	E	-

Erklärung:
- \- nicht enthalten
- A enthalten
- T Teilbereiche enthalten
- R nur referenzierte Seiten enthalten
- D nur bei besonderer DMS-Auswertung
- E je einmal für jede Error-Task bzw. für jede explizit ausgewählte Task enthalten

Tabelle 8: Bestandteile und Umfang der ausgegebenen Listings

Beschreibung der einzelnen Listenabschnitte:

Ausgabe der allgemeinen Dumpdatei-Informationen

Der Abschnitt enthält die Ausgaben des Statusfensters im INF-Modus und ist insbesondere dann wichtig, wenn die Dumpdatei mehrere Objekte enthält. Es wird das aktuell ausgewählte Objekt angezeigt.

Der Abschnitt ist in jedem der Modi SYS, MIN und ALL unter SELECT enthalten, kann aber mit SEL=INF gezielt angefordert werden.

Ausgabe der Systemübersicht

Es folgt in jeder SLED-, SNAP- und Systemdump-Liste, die mit SEL=MIN/ALL angefordert wird, eine Systemübersicht. Diese ist identisch mit der gezielt über SEL=SYS angestoßenen Systemübersicht.

Die Systemübersicht lässt erkennen,

- welches Systemumfeld zum Zeitpunkt des Dumps vorhanden war,
- welche globalen Systemprobleme ggf. zu diesem Zeitpunkt bestanden haben und
- welche weiteren Systemdateien zum Diagnoseumfeld gehören.

Im Einzelnen enthält die Systemübersicht folgende Angaben:

- Server-Typ
- Speicherausbau
- Anzahl aktiver Prozessoren
- BS2000-Dateiname
- Ladername
- Name der aktiven SERSLOG-Datei
- Name der aktiven CONSLOG-Datei
- Übersicht über die Software-Konfiguration (Subsysteme)
- Übersicht über die vom Tracemanager verwalteten eingeschalteten Traces
- Eingestellte Systemparameter
- Hinweise auf globale Systemprobleme (Sättigungszustände, FORCE-JOB-CANCEL etc.)
- Eventuell die durch die automatische Voranalyse erstellten Fehler-Deskriptoren
- Bei SLED-Listen die gesamte System-Trace-Table, die prozessorlokalen Linkage-AUDIT-Tabellen, die Taskübersicht und die maschinentypabhängigen Hardware-Bereiche
- Bei Systemdumps und eventuell bei User- bzw. Areadumps die taskspezifisch aufbereitete System-Trace-Table, die Prozesskontrollblöcke (PCBs) und Program-Manager-Stacks der Task, den TCB und die tasklokalen AUDIT-Tabellen

Ausgabe der CSECT-Liste

Die CSECT-Liste wird pro Subsystem und Version für privilegierte und nicht privilegierte Subsysteme ausgegeben. Dabei werden die CSECTs nach Adressen sortiert. Die ETPND-Modulinformation wird ebenfalls ausgegeben.

Die nach Namen sortierte CSECT-Liste enthält aus Gründen der Papierersparnis nur die zugehörigen Adressen der CSECTs und dient lediglich als Querverweisliste (Name -> Adresse).

> Die standardmäßig erstellte Liste enthält beim SLED bzw. SNAP nicht die Modul-Versionsnummern. Letztere erhält man durch explizites Markieren von CS oder ALL im MAPS-Feld des LIST-Fensters. Das Erstellen einer solchen Liste dauert allerdings mehrere Minuten.

Ausgabe der REP-Information

Die im Dump enthaltene REP-Information (REPLOG bzw. SAVEREP in älteren Betriebssystem-Versionen) wird ausgegeben.

Ausgabe der System-Trace-Table

Die System-Trace-Table wird sowohl vollständig ausgegeben, als auch als Auszug aller Einträge der betroffenen Task (einmal pro task-spezifischer Auswertung). Die Identifizierung gleicher Einträge ist über die Nummerierung der Trace-Einträge gewährleistet.

Ausgabe der PCBs

Die PCBs werden aufbereitet ausgegeben. Anschließend an die aufbereiteten PCBs wird die PCB-Kette noch einmal unaufbereitet (im Dumpformat) ausgegeben.

Ausgabe der SPL-Stacks

Die SPL-Stacks werden aufbereitet ausgegeben.

Ausgabe der Systemtabellen

Es werden die XVT und der TCB unaufbereitet (im Dumpformat) ausgegeben. Die Adressen werden dabei zum Tabellenanfang relativiert.

JCB und JTBP werden nicht ausgegeben. Daten aus den genannten Tabellen werden nur auszugsweise aufbereitet und ausgegeben (wie etwa der Name des geladenen Programms).

Ausgabe der TFT und tasklokaler DMS-Tabellen

Die TFT und die tasklokalen DMS-Tabellen werden nur dann ausgegeben, wenn für den Parameter PROC *DMS ausgewählt wurde oder wenn die automatische Voranalyse als Fehlerkomponente das DMS erkannt hat. Globale DMS-Tabellen werden generell nicht von DAMP aufbereitet.

Ausgabe der Speicherseiten

Die Speicherseiten werden auf folgende Weise ausgegeben:

- Minimum-Ausgabe (Voreinstellung):
  Die Speicherseiten der Speicherklassen 1,2,3,4 und 6 werden nur dann ausgegeben, wenn sie über ein Register eines PCBs, SPL-Stacks oder einer Savearea referenziert sind.
  Als markiert gelten auch die Speicherseiten, die während einer vorangegangenen Diagnose im Dialog, implizit beim Ablauf der automatischen Vordiagnose oder bei der Initialisierung von DAMP angesprochen wurden.
  Der Klasse-5-Speicher wird abhängig vom Inhalt des Dumpfiles vollständig ausgegeben.
  Als „secret-pages“ gekennzeichnete Seiten sind abhängig vom Systemparameter DUMPSEPA im Dumpfile enthalten oder nicht enthalten.

- Voll-Ausgabe:
  Es werden alle Seiten der Speicherklassen 1,2,3,4,5,6 ausgegeben, sofern sie im Dumpfile vorhanden sind.

5.5 Abläufe automatisieren

- Automatische Voranalyse
- Batch- und Prozedurbetrieb, Anweisungsfolgen
- Automatisieren mit PRODAMP

5.5.1 Automatische Voranalyse

Mit DAMP werden standardmäßig PRODAMP-Prozeduren ausgeliefert, die einen ersten Ansatz zu einer automatischen Voranalyse darstellen. Die bei der Analyse durchlaufenen Diagnoseschritte und die hierbei ermittelten Diagnoseerkenntnisse führen

- zu einer internen Liste von referenzierten und diagnose-relevanten Speicherseiten,
- zu einer Deskribierung der Fehlerursache und Fehlerumgebung
- und zum Versuch einer Fehlereingrenzung auf die Fehlerkomponente, bei SLED- und SNAP-Dateien auch auf die Errortask.

Die Ergebnisse der Voranalyse beeinflussen die Steuerung der nachfolgenden Dump-Druckaufbereitung, indem zusätzliche Tabellen-Ausgaben veranlasst werden und die Liste auf die Errortask beschränkt bleibt.

Starten der Voranalyse

- bei einer List-Auswertung (`START-LIST-GENERATION`) (siehe "Felder für Vordiagnose und Fehlerdeskriptoren") oder
- durch Aufruf einer PRODAMP-Prozedur mit der Anweisung `START-PRODAMP-PROGRAM NAME=DIAG`

Mit der Anweisung `OPEN-DIAGNOSIS-OBJECT` muss vorher ein SLED, SNAP oder Systemdump zugewiesen werden.

Die zugehörigen PRODAMP-Routinen werden in der Datei $TSOS.SYSDMP.DAMP zur Verfügung gestellt. Beim Start der PRODAMP-Routine DIAG mit `START-PRODAMP-PROGRAM` ist zu beachten, dass diese System-PRODAMP-Bibliothek als aktuelle Benutzer-PRODAMP-Bibliothek zugewiesen ist. Ggf. ist die Bibliothek mit der Anweisung `ASSIGN-PRODAMP-LIBRARIES` zuzuweisen. Die aktuelle Einstellung kann mit der Anweisung `SHOW-PRODAMP-LIBRARIES` ermittelt werden.

Die Deskribierung (= Kurzbeschreibung des Problems in Form einer Kette von Deskriptoren, also beschreibenden Stichworten) ist eine bewährte Methode zur Duplikats-Erkennung. Die mit DAMP durchgeführte automatische Deskribierung liefert eine Reihe von solchen Deskriptoren.

Diese Deskriptoren werden ausgegeben

- **auf Papier** am Anfang der Ausgabeliste (im Rahmen der Systemübersicht), wenn die Voranalyse innerhalb einer List-Auswertung erfolgt
- **auf das Terminal** in den EDT-Bereich 8, zu erreichen mit der DAMP-Anweisung EDT und der EDT-Anweisung „[$]8“.

Der Aufruf der Voranalyse über `START-PRODAMP-PROGRAM NAME=DIAG` verursacht keine implizite Druckaufbereitung.

Die automatische Voranalyse versucht auch, das vorliegende Problem einer Fehlerkomponente zuzuordnen. Gehört diese Fehlerkomponente beispielsweise zum Subsystem DMS (Allocator, Catalog-Management, Open-Close, Tape-Processing, Zugriffsmethoden außer ISAM), so wird der Deskriptor DMS in die Deskriptorenkette aufgenommen.

Die Erkennung einer Fehlerkomponente aus dem Subsystem DMS hat Auswirkungen auf die Druckaufbereitung: Neben den Systemtabellen der Standard-Liste werden taskspezifische DMS-Tabellen ausgegeben.

Bei SLED- und SNAP-Dateien steuert die automatische Voranalyse den Umfang der Listenaufbereitung, wenn eine eindeutige Error-Task erkannt wird (z.B. $CRASH oder Program Error in Pre-allocated Task). Bei der Standard-Aufbereitung wird dann eine taskspezifische Auswertung dieser Task angestoßen.

Eine Voranalyse für task-spezifische Probleme bei SLED- oder SNAP-Dateien (z.B. für die Tasks, die zum SLED-Zeitpunkt auf der Rückfrage an den Benutzer stehen „`DUMP desired? YES/NO`“), lässt sich über die Task-Auswahl in der List-Maske DIAG=YES/NO steuern. Bei dieser Anwendung wird der Deskriptor TASK.ONLY aufgenommen.

> **i** Die automatische Voranalyse kann weder alle Probleme richtig erkennen, noch die erkannten Probleme bis in die Tiefe durchanalysieren. Insbesondere bei SLED-Analysen und hier wiederum speziell bei Deadlock-Problemen kommt die automatische Voranalyse oft zu wenig brauchbaren Resultaten.

5.5.2 Batch- und Prozedurbetrieb, Anweisungsfolgen

DAMP lässt sich auch aus einer BS2000-Prozedur heraus oder als Batchjob starten bzw. betreiben. So kann eine Prozedurdatei zum Beispiel die entsprechenden Kommandos ADD-FILE-LINK für die zuzuweisende Dump- und die Protokolldatei enthalten, entsprechende DAMP-Anweisungen für die Zuweisung von eigenen Symboldateien, für das Einschalten des Loggings oder den Aufruf von Dumpanalyse- und Listenaufbereitungs-Routinen. Vor dem Aufruf des Programms DAMP ist in der Prozedur der Auftragsschalter 5 zu setzen.

> ¡ Folgt im Prozedurbetrieb auf eine DAMP-Anweisung ein Systemkommando, das nicht in der DAMP-Programmebene zugelassen ist (siehe "Auf Systemebene"), geht DAMP vom Prozedur- in den Dialogbetrieb über und setzt den Auftragsschalter 5 zurück. Am Bildschirm wird dann der letzte DAMP-Ausgabeschirm angezeigt. Im Batchbetrieb führt das nicht zugelassene Systemkommando zum Abbruch des Auftrags.

Ständig wiederkehrende Anweisungsfolgen, die während der Diagnose mit DAMP notwendig sind, können ebenfalls in einer Datei zusammengefasst werden. Diese Datei kann im Dialogbetrieb, aber auch im Batch- und Prozedurbetrieb mit der Anweisung `START-STATEMENT-SEQUENCE` aktiviert werden. Nach Abarbeitung der Anweisungsfolge im Dialogbetrieb erscheint die zuletzt veranlasste Ausgabe auf dem Bildschirm. Im Prozedur- bzw. Batchbetrieb wird die Abarbeitung der Prozedur- bzw. Batchdatei fortgesetzt.

> ¡ Bei der Anweisung `START-STATEMENT-SEQUENCE` wird SYSDTA vorübergehend auf die angegebene Datei gelegt. Falls die Zuweisung von SYSDTA in dieser Prozedur geändert wird (auch z.B. innerhalb einer mit `START-MODULE` aufgerufenen Prozedur), können unerwartete Folgen eintreten. Insbesondere sollte die Anweisung `START-MODULE DCM` nicht in einer Anweisungsdatei gegeben werden, da dies zu Endlosschleifen führen kann.

Beispiele für Anweisungsfolgen

Inhalt der Datei DAMP.STATEMENT.DCM

`LOG-SESSION`	Logging einschalten
`OPEN-DIAGNOSIS-OBJECT OBJECT=#1`	Dumpdatei zuweisen
`ADD-SYMBOLS MY.SYMBOLS(PCS(<ver>))`	private Symboldatei PCS
`START-PATTERN-SEARCH`	Stringsuche vorbereiten
`SHOW-EDITED-INFO INFO=*TRACE, WINDOW=8`	Trace-Ausgabe auf Fenster W8
`MODIFY-SCREEN FIRST=3(10), SECOND=4(8)`	Fensterfolge für 1. Bildschirm

5.5.3 Automatisieren mit PRODAMP

PRODAMP (PROzedursprache für DAMP) ist eine an Pascal orientierte Sprache zur Formulierung von Diagnose-Algorithmen in DAMP. Mit PRODAMP können Sie an Entscheidungen gebundene Anweisungen, die sonst einzeln von Hand eingegeben werden müssten, in eine Prozedur schreiben und automatisch ablaufen lassen. So können Sie zum Beispiel Verkettungen bis zu einer Struktur verfolgen, die ein gesuchtes Datum enthält, Tabellen durchsuchen und die darin enthaltenen Werte (z.B. arithmetisch) verarbeiten, oder Fragen der Art: „Hält diese Task einen Lock“ automatisch beantworten lassen.

PRODAMP wird im Abschnitt „PRODAMP“ ausführlich beschrieben.

5.6 Programmanweisungen

5.6.1 Auf der Programmebene

DAMP-Anweisungen sind am Bildschirm in der Kommandozeile einzugeben. Die Eingaben im Dialog werden unterstützt durch die Dialogschnittstelle SDF. Die Metasyntax zu den Anweisungen ist im Handbuch „Kommandos“ [8] enthalten.
Längere Anweisungen müssen eindeutig abgekürzt eingegeben werden, da die Kommandozeile nur 75 Zeichen aufnehmen kann.
Im Batch- und Prozedurbetrieb sind die Anweisungen die einzig zulässigen Eingaben, die von SYSDTA gelesen und verarbeitet werden.

Folgende DAMP-Anweisungen, untergliedert nach Anwendungsgebieten, stehen dem Anwender zur Verfügung:

Behandlung des Diagnoseobjektes

OPEN-DIAGNOSIS-OBJECT	Eröffnen eines Diagnoseobjektes für die Bearbeitung
ADD-SYMBOLS	Zuweisen von Symbolen für die Ausgabe
MODIFY-OBJECT-ASSUMPTIONS	Ändern der Standardeinstellungen für das Diagnoseobjekt

Steuerung der Darstellung

MODIFY-SCREEN-LAYOUT	Neue Reihenfolge und Größe der Diagnosefenster festlegen
SHOW-EDITED-INFORMATION	Ausgabe von speziell aufbereiteten Diagnosedaten
USE-REGISTER	Darstellung für die disassemblierte Ausgabe festlegen
DROP-REGISTER	Eine mit USE-REGISTER getroffene Einstellung rückgängig machen

Aufzeichnen und Abspielen einer DAMP-Sitzung

LOG-SESSION	Diagnosemitschnitt einschalten; eine Logging-Datei wird erzeugt
REPEAT-SESSION	Diagnosemitschnitt wiederabspielen
PRINT-LOGGING-FILE	Aufbereiten und Ausdrucken einer Logging-Datei
STOP-LOGGING	Beendet den mit LOG-SESSION begonnenen Diagnosemitschnitt

Unterstützung automatisierter Diagnoseabläufe

1. PRODAMP

SHOW-PRODAMP-LIBRARIES	Anzeige der aktuellen PRODAMP-Bibliotheken
ASSIGN-PRODAMP-LIBRARIES	Zuweisen von Bibliotheken für den PRODAMP-Compiler und/oder PRODAMP-Editor
START-PRODAMP-EDITOR	Editor für den PRODAMP-COMPILER aufrufen
COMPILE-PRODAMP-PROCEDURE	PRODAMP-Prozeduren außerhalb des PRODAMP-Editors übersetzen

START-PRODAMP-PROGRAM	Laden und Starten eines PRODAMP-Programms
RESUME-PRODAMP-PROGRAM	Fortsetzen eines unterbrochenen PRODAMP-Programms

2. Externe Unterprogramme über VMOS-Linkage

LOAD-MODULE	Externes Unterprogramm laden
START-MODULE	Externes Unterprogramm starten

3. DAMP-Prozeduren

START-STATEMENT-SEQUENCE	DAMP-Anweisungen aus Datei lesen und ausführen

Listenerzeugung

START-LIST-GENERATION	Einleiten der Listenausgabe
ADD-LIST-OBJECTS	Umfang der Listenausgabe festlegen durch Hinzufügen von Bereichen
REMOVE-LIST-OBJECTS	Umfang der Listenausgabe festlegen durch Ausschluss von Bereichen
PRINT-LIST	Starten der Listenausgabe mit Angabe des Ausgabeziels

Weitere Anweisungen

START-PATTERN-SEARCH	Einleiten der Stringsuche
START-OPTION-DIALOG	Einstellen der Benutzeroptionen
EDIT-FILE	Aufrufen des EDT als Unterprogramm
SHOW-LAST-STATEMENT	Anzeige der letzten DAMP-Anweisung
END	Beenden von DAMP
SEARCH-IN-SUBSYSTEM	Beschränken der CSECT-Suche auf ein bestimmtes Subsystem
SHOW-SUBSYSTEM-FOR-SEARCH	Anzeigen des für die CSECT-Suche ausgewählten Subsystems

DAMP-Anweisungen über das Systemkommando INFORM-PROGRAM

Operand MSG	Kommunikation mit DAMP vom System aus

Die Steuerung von DAMP-Funktionen über das Systemkommando INFORM-PROGRAM ist im Abschnitt Abschnitt „Auf Systemebene" beschrieben.

In den folgenden Abschnitten sind die DAMP-Anweisungen in alphabetischer Reihenfolge beschrieben.

5.6.1.1 ADD-LIST-OBJECTS Umfang der Listenausgabe festlegen

Mit der Anweisung ADD-LIST-OBJECTS wird der Umfang der Listenausgabe festgelegt. Alle Aufrufe dieser Anweisung, die zwischen START-LIST-GENERATION und PRINT-LIST erfolgen, werden gesammelt und beim abschließenden PRINT-LIST berücksichtigt. Ebenso werden alle Aufrufe der Anweisung REMOVE-LIST-OBJECTS registriert und berücksichtigt.
Da die Anweisung recht komplex ist, empfiehlt sich bei Auswahl mehrerer Objekte, diese auch auf mehrere ADD-LIST-OBJECTS-Anweisungen zu verteilen.

Format

```
ADD-LIST-OBJECTS

GLOBAL-INFORMATION = *NONE / *INF / *STD / *OVERVIEW / *ALL-MEMORY-AREAS / *PARAMETERS(...)
    *PARAMETERS(...)
        |   TRACES = *NONE / *ALL / list-poss(2): *SYSTEM-TRACE-TABLE / *TRACE-MANAGER-TABLES
        |   ,MAPS = *NONE / *ALL / list-poss(2): *CSECT-MAPS / *CONCISE-REP-INFORMATION
        |   ,CONTROL-BLOCKS = *NONE / *ALL / list-poss(2): *XVT / *AUDIT-TABLES
        |   ,MEMORY-AREAS = *NONE / *ALL / list-poss(9): *CL1 / *CL2 / *CL3 / *CL3-PARTIAL-PAGES /
        |                           *CL3-FULL-PAGES / *CL4 / *CL4-PARTIAL-PAGES / *CL4-FULL-PAGES /
        |                           *CL4-NON-PRIVILEGED
        |   ,PAGES = *NONE / <x-string 1..8>(...) / *INTERVAL(...)
        |       <x-string 1..8>(...)
        |           |   SPACE = *VIRTUAL-MEMORY / *REAL-MEMORY (...) / *ABSOLUTE-MEMORY (...) /
        |           |               *PAM-PAGES / *HARDWARE-SYSTEM-AREA /
        |           |               *PROCESSOR-SAVED-STATUS(...) / *ALET(...) / *SPID(...)
        |           |       *REAL-MEMORY(...)
        |           |           |   SEGMENT = x'0' / <x-string 1..8>
        |           |       *ABSOLUTE-MEMORY(...)
        |           |           |   SEGMENT = x'0' / <x-string 1..8>
        |           |       *PROCESSOR-SAVED-STATUS(...)
        |           |           |   CPU-NUMBER = <integer 0..31> / <x_string 1..2>
        |           |       *ALET(...)
        |           |           |   IDENTIFIER = <x-string 1..8>
        |           |       *SPID(...)
        |           |           |   IDENTIFIER = <x-string 1..16>
        |       *INTERVAL(...)
        |           |   FROM = <x-string 1..8>
```

```
        |           |   ,TO = <x-string 1..8>
        |           |   ,SPACE = *VIRTUAL-MEMORY / *REAL-MEMORY(...) / *ABSOLUTE-MEMORY(...) /
        |           |                *PAM-PAGES / *HARDWARE-SYSTEM-AREA /
        |           |                *PROCESSOR-SAVED-STATUS(...) / *ALET(...) / *SPID(...)
        |           |       *REAL-MEMORY(...)
        |           |           |   SEGMENT = x'0' / <x-string 1..8>
        |           |       *ABSOLUTE-MEMORY(...)
        |           |           |   SEGMENT = x'0' / <x-string 1..8>
        |           |       *PROCESSOR-SAVED-STATUS(...)
        |           |           |   CPU-NUMBER = <integer 0..31> / <x_string 1..2>
        |           |       *ALET(...)
        |           |           |   IDENTIFIER = <x-string 1..8>
        |           |       *SPID(...)
        |           |           |   IDENTIFIER = <x-string 1..16>
        |   ,MODULE = *NONE / <name 1..32>
,TASK-INFORMATION = *NONE / *PARAMETERS(...)
    *PARAMETERS(...)
        |   SELECT = *ERROR-TASK / *ALL-TASKS / <x-string 1..8> / <alphanum-name 1..4> / <c-string 1..4>
        |   ,INFORMATION = *STD / *INF / *OVERVIEW / *ALL-MEMORY-AREAS / *PARAMETERS(...)
        |       *PARAMETERS(...)
        |           |   TRACES = *NONE / *SYSTEM-TRACE-TABLE
        |           |   ,MAPS = *NONE / *USER-CSECTS
        |           |   ,CONTROL-BLOCKS = *NONE / *ALL / list-poss(5): *TCB / *PCBS / *SPL-STACKS /
        |           |                                *TFTS / *AUDIT-TABLES
        |           |   ,MEMORY-AREAS = *NONE / *ALL / list-poss(7): *CL5 / *CL5-PARTIAL-PAGES /
        |           |                                *CL5-FULL-PAGES / *CL5-MEMORY-POOLS / *CL6 /
        |           |                                *CL6-FULL-PAGES / *CL6-MEMORY-POOLS
        |           |   ,PAGES = *NONE / <x-string 1..8>(...) / *INTERVAL(...)
        |           |       <x-string 1..8>>(...)
        |           |           |       SPACE = *VIRTUAL-MEMORY / *ALET(...)
        |           |           |           *ALET(...)
        |           |           |           |           IDENTIFIER = <x-string 1..8>
        |           |       *INTERVAL(...)
        |           |           |   FROM = <x-string 1..8>
        |           |           |   ,TO = <x-string 1..8>
```

```
        |           |           |   ,SPACE = *VIRTUAL-MEMORY / *ALET(...)
        |           |           |       *ALET(...)
        |           |           |           |   ,IDENTIFIER = <x-string 1..8>
        |           |   ,MODULE = *NONE / <name 1..32>
,USER-LIST-PROCEDURE = *NONE / <name 1..32 with-under> / <structured-name 1..32>
,WINDOW = *NONE / <integer 4..99>
,DUMP-DIAGNOSIS = *NONE / *PREANALYSIS(...)
    *PREANALYSIS(...)
        |   ERROR-DESCRIPTION = *NO / *YES
```

Operandenbeschreibung

GLOBAL-INFORMATION = *NONE / *INF / *STD / *OVERVIEW / *ALL-MEMORY-AREAS / *PARAMETERS(...)
Mit diesem Operanden werden globale, d.h. nicht taskspezifische Bereiche für die Listenausgabe ausgewählt.

GLOBAL-INFORMATION = *NONE
Die Listenausgabe soll keine globalen Speicherbereiche beinhalten.

GLOBAL-INFORMATION = *INF
Die Listenausgabe soll allgemeine Informationen über das zu analysierende Objekt enthalten.

GLOBAL-INFORMATION = *STD
Es wird eine Minimum-Liste ausgegeben. Diese Liste enthält in der Regel alle für eine erste Diagnose erforderlichen Daten (siehe Abschnitt „Bestandteile und Umfang der ausgegebenen Listen").

GLOBAL-INFORMATION = *OVERVIEW
Es wird eine Systemübersicht ausgegeben (Inhalt siehe Abschnitt „Bestandteile und Umfang der ausgegebenen Listen").

GLOBAL-INFORMATION = *ALL-MEMORY-AREAS
Es wird eine vollständige Liste ausgegeben (Inhalt siehe Abschnitt „Bestandteile und Umfang der ausgegebenen Listen"). Insbesondere werden bei System-, User- und Areadumps alle von CDUMP sichergestellten Seiten auf Liste ausgegeben.

GLOBAL-INFORMATION = *PARAMETERS(...)
Mit diesem Operanden können globale Bereiche einzeln ausgewählt werden.

> **TRACES = *NONE / *ALL / list-poss(2): *SYSTEM-TRACE-TABLE /*TRACE-MANAGER-TABLES**
> Legt fest, welche System-Traces ausgegeben werden sollen.
>
> **TRACES = *NONE**
> Es sollen keine System-Traces ausgegeben werden. Voreinstellung.
>
> **TRACES = *ALL**
> Es werden alle System-Traces ausgegeben.
>
> **TRACES = list-poss(2): *SYSTEM-TRACE-TABLE / *TRACE-MANAGER-TABLES**
> Die System-Trace-Table und eine Übersicht der vom Trace-Manager verwalteten Traces können einzeln oder in Form einer Liste ausgewählt werden.

MAPS = *NONE / *ALL / list-poss(2): *CSECT-MAPS / *CONCISE-REP-INFORMATION
Legt fest, welche Informationen über geladene CSECTs bzw. über den Korrekturstand des Systems ausgegeben werden soll.

MAPS = *NONE
Es sollen keine Maps ausgegeben werden.

MAPS = *ALL
Sowohl CSECT-Maps als auch die REPLOG sollen ausgegeben werden, sofern sie im Diagnoseobjekt vorhanden sind.

MAPS = list-poss(2): *CSECT-MAPS / *CONCISE-REP-INFORMATION
Die CSECT-Maps und die REPLOG können einzeln oder in einer Liste gemeinsam ausgewählt werden.

CONTROL-BLOCKS = *NONE / *ALL / list-poss(2): *XVT / *AUDIT-TABLES
Legt fest, welche Tabellen ausgegeben werden sollen.

CONTROL-BLOCKS = *NONE
Es werden keine Tabellen ausgegeben.

CONTROL-BLOCKS = *ALL
Es werden die XVT- und AUDIT-Tabellen ausgegeben.

CONTROL-BLOCKS = list-poss(2): *XVT / *AUDIT-TABLES
Die globale Systemtabelle XVT und die AUDIT-Tabellen können einzeln oder in einer Liste gemeinsam ausgewählt werden.
Die XVT wird, falls angegeben, unaufbereitet ausgegeben; bei den AUDIT-Tabellen werden, falls vorhanden, die prozessorlokalen Linkage-AUDIT-Tabellen ausgegeben.

MEMORY-AREAS = *NONE / *ALL / list-poss(9): *CL1 / ... / *CL4-NON-PRIVILEGED
Legt bestimmte globale Bereiche des virtuellen Adressraums fest, die über ihre Speicherklasse identifiziert und vollständig im Standard-Dump-Format ausgegeben werden. Voreinstellung: Keine Ausgabe. Der Operandenwert *ALL fasst alle in der Liste aufgeführten Speicherbereiche zusammen.

CL1:	Residente Systemmodule
CL2:	Seitenwechselbare Systemmodule
CL3:	Residente Tabellen und Subsystemmodule
CL3-PARTIAL-PAGES:	Residente Teilseiten
CL3-FULL-PAGES:	Residente Vollseiten
CL4:	Seitenwechselbare Tabellen und Subsystemmodule
CL4-PARTIAL-PAGES:	Seitenwechselbare Teilseiten
CL4-FULL-PAGES:	Seitenwechselbare Vollseiten
CL4-NON-PRIVILEGED:	Nicht-privilegierte System-Seiten

PAGES = *NONE / <x-string 1..8>(...) / *INTERVAL(...)
Gezielte Auswahl von Speicherseiten oder Speicherbereichen des Adressraums. Voreinstellung: Keine Ausgabe.

PAGES = <x-string 1..8>(...)
Gezielte Auswahl einer Speicherseite.

SPACE = *VIRTUAL-MEMORY / *REAL-MEMORY(...) / *ABSOLUTE-MEMORY(...) / *PAM-PAGES / *HARDWARE-SYSTEM-AREA / *PROCESSOR-SAVED-STATUS(...) / *ALET(...) / *SPID(...)
Legt fest, auf welche Art von Speicher sich die Seitenangabe bezieht.

SPACE = *VIRTUAL-MEMORY
Keine Angabe ist gleichbedeutend mit *VIRTUAL-MEMORY, d.h. standardmäßig wird der virtuelle Speicher zu Grunde gelegt.

SPACE = *REAL-MEMORY(...)
Bezeichnet den Hauptspeicher (reale Adressierung).

SEGMENT = x'0' / <x-string 1..8>
Bezeichnet das zur realen Seite gehörige 4GB-Segment (0,1,...). Voreingestellt ist das Segment 0.

SPACE = *ABSOLUTE-MEMORY(...)
Bezeichnet hostabsolute Adressen (z.B. bei vollständiger VM2000-SLED-Datei).

SEGMENT = x'0' / <x-string 1..8>
Bezeichnet das zur absoluten Seite gehörige 4GB-Segment (0,1,...). Voreingestellt ist das Segment 0.

SPACE = *PAM-PAGES
Die Seitenangabe bezieht sich auf PAM-Seiten.

SPACE = *HARDWARE-SYSTEM-AREA
Bezeichnet den hardware-nahen Speicherbereich HSA (Hardware-System-Area, enthält z.B. Tabellen zur Kommunikation zwischen CPU und IO-Prozessor). Die angegebene Seite muss innerhalb der HSA liegen.

SPACE = *PROCESSOR-SAVED-STATUS(...)
Die Seitenangabe bezieht sich auf eine Processor-Saved-Status-Area.

CPU-NUMBER = <integer 0..31>
Dezimale CPU-Nummer, wenn die Seite 0 der Processor-Saved-Status-Area ausgegeben werden soll.

CPU-NUMBER = <x-string 1..2>
Sedezimale CPU-Nummer, wenn die Seite 0 der Processor-Saved-Status-Area ausgegeben werden soll.

SPACE = *ALET(...)
Bezeichnet Speicherseiten aus Datenräumen (Erweiterung des virtuellen Adressraums des Systems). Über den ALET (ACCESS-LIST-ENTRY-TOKEN) wird der Datenraum für den System-Adressraum identifiziert.

IDENTIFIER = <x-string 1..8>
Der 4 Byte lange ALET wird sedezimal angegeben.

SPACE = *SPID(...)
Bezeichnet Speicherseiten aus Datenräumen (Erweiterung des virtuellen Adressraums des Systems). Über die SPID (Space-Identification) wird der Datenraum systemweit identifiziert.

IDENTIFIER = <x-string 1..16>
Die 8 Byte lange SPID wird sedezimal angegeben.

PAGES = *INTERVAL(...)
Auswahl mehrerer Speicherseiten durch Angabe eines Intervalls.

FROM = <x-string 1..8>
Bezeichnet die erste Speicherseite des Speicherbereichs.

TO = <x-string 1..8>
Bezeichnet die letzte Speicherseite des Speicherbereichs.

SPACE = *VIRTUAL-MEMORY / *REAL-MEMORY(...) / *ABSOLUTE-MEMORY(...) / *PAM-PAGES / *HARDWARE-SYSTEM-AREA / *PROCESSOR-SAVED-STATUS(...) / *ALET(...) / *SPID(...)
Legt fest, auf welche Art von Speicher sich die Seitenangabe bezieht.

SPACE = *VIRTUAL-MEMORY
Keine Angabe ist gleichbedeutend mit VIRTUAL-MEMORY, d.h. standardmäßig wird der virtuelle Speicher zu Grunde gelegt.

SPACE = *REAL-MEMORY(...)
Bezeichnet den Hauptspeicher (reale Adressierung).

> **SEGMENT = x'0' / <x-string 1..8>**
> Bezeichnet das zur realen Seite gehörige 4GB-Segment (0,1,...). Voreingestellt ist das Segment 0.

SPACE = *ABSOLUTE-MEMORY(...)
Bezeichnet hostabsolute Adressen (z.B. bei vollständiger VM2000-SLED-Datei).

> **SEGMENT = x'0' / <x-string 1..8>**
> Bezeichnet das zur absoluten Seite gehörige 4GB-Segment (0,1,...). Voreingestellt ist das Segment 0.

SPACE = *PAM-PAGES
Die Seitenangabe bezieht sich auf PAM-Seiten.

SPACE = *HARDWARE-SYSTEM-AREA
Bezeichnet den hardware-nahen Speicherbereich HSA (Hardware-System-Area); die angegebenen Seiten müssen innerhalb der HSA liegen.

SPACE = *PROCESSOR-SAVED-STATUS(...)
Die Seitenangabe bezieht sich auf eine Processor-Saved-Status-Area.

> **CPU-NUMBER = <integer 0..31>**
> Dezimale CPU-Nummer, wenn die Seite 0 der Processor-Saved-Status-Area ausgegeben werden soll.

> **CPU-NUMBER = <x-string 1..2>**
> Sedezimale CPU-Nummer, wenn die Seite 0 der Processor-Saved-Status-Area ausgegeben werden soll.

SPACE = *ALET(...)
Bezeichnet Speicherseiten aus Datenräumen (Erweiterung des virtuellen Adressraums des Systems). Über den ALET (ACCESS-LIST-ENTRY-TOKEN) wird der Datenraum für den System-Adressraum identifiziert.

> **IDENTIFIER = <x-string 1..8>**
> Der 4 Byte lange ALET wird sedezimal angegeben.

SPACE = *SPID(...)
Bezeichnet Speicherseiten aus Datenräumen (Erweiterung des virtuellen Adressraums des Systems). Über die SPID (Space-Identification) wird der Datenraum systemweit identifiziert.

IDENTIFIER = <x-string 1..16>
Die 8 Byte lange SPID wird sedezimal angegeben.

MODULE = *NONE / <name 1..32>
Bezeichnet den gesamten Speicherbereich, der von dem genannten System-Modul belegt wird.
Voreinstellung: Keine Ausgabe dieses Speicherbereichs.

TASK-INFORMATION = *NONE / *PARAMETERS(...)
Mit diesem Operanden werden taskspezifische Bereiche ausgewählt.

TASK-INFORMATION = *NONE
Die Listenausgabe soll keine taskspezifischen Speicherbereiche beinhalten.

TASK-INFORMATION = *PARAMETERS(...)
Gezielte Auswahl von taskspezifischen Speicherbereichen.

SELECT = *ERROR-TASK / *ALL-TASKS / <x-string 1..8> /<alphanum-name 1..4> / <c-string 1..4>
Auswahl der Task, deren Speicherbereiche ausgegeben werden sollen.

SELECT = *ERROR-TASK
Auswahl der Error-Task. Bei System-, User- und Areadumps ist dies die einzige im Dumpfile enthaltene Task. Bei Sleds wird die Error-Task durch die automatische Voranalyse innerhalb DAMP bestimmt.

SELECT = *ALL-TASKS
Bei einem Sled wird die Task-Aufbereitung aller aktiven Tasks veranlasst.

SELECT = <x-string 1..8>
Die gewünschte Task wird durch die 4 Byte lange TID sedezimal angegeben.

SELECT = <alphanum-name 1..4>
Die gewünschte Task wird durch einen alphanumerischen Namen bezeichnet, der als TSN der Task interpretiert wird.

SELECT = <c-string 1..4>
Die gewünschte Task wird durch einen Zeichenstring bezeichnet, der als TSN der Task interpretiert wird.

INFORMATION = *STD / *INF / *OVERVIEW / *ALL-MEMORY-AREAS /*PARAMETERS(...)
Steuert den Umfang der Ausgabe für taskspezifische Daten.

INFORMATION = *STD
Es wird eine Minimum-Liste ausgegeben. Diese Liste enthält in der Regel alle für eine erste Diagnose erforderlichen Daten (siehe Abschnitt „Bestandteile und Umfang der ausgegebenen Listen").

INFORMATION = *INF
Die Listenausgabe soll allgemeine Informationen über die ausgewählte Task enthalten.

INFORMATION = *OVERVIEW
Es wird eine Systemübersicht ausgegeben (Inhalt siehe Abschnitt „Bestandteile und Umfang der ausgegebenen Listen").

INFORMATION = *ALL-MEMORY-AREAS
Es wird eine vollständige Liste ausgegeben (Inhalt siehe Abschnitt „Bestandteile und Umfang der ausgegebenen Listen"). Insbesondere werden bei System-, User- und Areadumps alle von CDUMP sichergestellten Seiten auf Liste ausgegeben.

INFORMATION = *PARAMETERS(...)
Mit diesem Operanden können taskspezifische Bereiche einzeln ausgewählt werden.

TRACES = *NONE / *SYSTEM-TRACE-TABLE
Legt fest, ob taskspezifische Informationen aus der System Trace Table ausgegeben werden sollen. Voreinstellung: Keine Ausgabe.

MAPS = *NONE / *USER-CSECTS
Steuert die Ausgabe von Informationen über die geladenen User-CSECTs. Informationen über User-CSECTs können nur bei User- und Areadumps ausgegeben werden, wenn die Binder-/Lader-Informationen, die im Klasse-5-Speicher liegen, im Objekt enthalten sind. Voreinstellung: Keine Ausgabe.

CONTROL-BLOCKS = *NONE / *ALL / list-poss(5): *TCB / ... / *AUDIT-TABLES
Legt fest, ob die taskspezifischen Kontrollblöcke im Standard-Dump-Format ausgegeben werden sollen.

CONTROL-BLOCKS = *NONE
Keine Ausgabe der taskspezifischen Kontrollblöcke. Voreinstellung.

CONTROL-BLOCKS = *ALL
Alle taskspezifischen Kontrollblöcke sollen ausgegeben werden.

CONTROL-BLOCKS = list-poss(5): *TCB / *PCBS / *SPL-STACKS / *TFTS /*AUDIT-TABLES
Gezielte Auswahl taskspezifischer Kontrollblöcke. Die Blöcke können einzeln oder in Form einer Liste ausgewählt werden. TCB (Task Controll Block), PCBS (Process Controll Blocks), SPL-STACKS, (TPR-Program-Manager-Stacks), TFTS (Task File Tables) und AUDIT-TABLES (Hardware- und Linkage-AUDIT).
Bei Angabe von PCBS werden die PCBs noch zusätzlich aufbereitet ausgegeben.

MEMORY-AREAS = *NONE / *ALL / list-poss(7): *CL5 / ... /*CL6-MEMORY-POOLS
Bezeichnet Bereiche des virtuellen Task-Adressraums, die durch Angabe ihrer Speicherklasse ausgewählt und vollständig im Standard-Dump-Format ausgegeben werden sollen.

MEMORY-AREAS = *NONE
Keine speziellen Bereiche des virtuellen Task-Adressraums werden ausgewählt.

MEMORY-AREAS = *ALL
Alle Speicherbereiche des Klasse-5- und Klasse-6-Speichers sollen im Standard-Dump-Format ausgegeben werden.

MEMORY-AREAS = list-poss(7): *CL5 / *CL5-PARTIAL-PAGES /*CL5-FULL-PAGES / *CL5-MEMORY-POOLS / *CL6 / *CL6-FULL-PAGES /*CL6-MEMORY-POOLS
Die Bereiche des Klasse-5- und Klasse-6-Speichers können einzeln oder in Form einer Liste ausgewählt werden.

CL5:	Gesamter Klasse-5-Speicher
CL5-PARTIAL-PAGES:	Privilegierte Teilseiten des Klasse-5-Speichers
CL5-FULL-PAGES:	Privilegierte Vollseiten des Klasse-5-Speichers
CL5-MEMORY-POOLS:	Memory-Pool-Seiten des Klasse-5-Speichers
CL6:	Gesamter Klasse-6-Speicher
CL6-FULL-PAGES:	Nicht privilegierte Vollseiten des Klasse-6-Speichers
CL6-MEMORY-POOLS:	Memory-Pool-Seiten des Klasse-6-Speichers

PAGES = *NONE / <x-string 1..8>(...) / *INTERVAL(...)
Gezielte Auswahl von Speicherseiten, die ausgegeben werden sollen.

PAGES = *NONE
Keine Angabe ist gleichbedeutend mit *NONE, d.h. es sollen keine speziellen Speicherseiten ausgegeben werden.

PAGES = <x-string 1..8>(...)
Gezielte Auswahl einer Speicherseite durch eine sedezimale Seitenangabe.

SPACE = *VIRTUAL-MEMORY / *ALET(...)
Legt fest, auf welche Art von Speicher sich die Seitenangabe bezieht.

SPACE = *VIRTUAL-MEMORY
Keine Angabe ist gleichbedeutend mit *VIRTUAL-MEMORY, d.h. standardmäßig wird der virtuelle Speicher zu Grunde gelegt.

SPACE = *ALET(...)
Bezeichnet Speicherseiten aus Datenräumen (Erweiterung des virtuellen Adressraums des Systems). Über den ALET (ACCESS-LIST-ENTRY-TOKEN) wird der Datenraum für den Task-Adressraum identifiziert.

IDENTIFIER = <x-string 1..8>
Der 4 Byte lange ALET wird sedezimal angegeben.

PAGES = *INTERVAL(...)
Auswahl mehrerer Speicherseiten durch Angabe eines Intervalls.

FROM = <x-string 1..8>
Bezeichnet die erste Speicherseite des Speicherbereichs.

TO = <x-string 1..8>
Bezeichnet die letzte Speicherseite des Speicherbereichs.

SPACE = *VIRTUAL-MEMORY / *ALET(...)
Legt fest, auf welche Art von Speicher sich die Seitenangabe bezieht.

SPACE = *VIRTUAL-MEMORY
Keine Angabe ist gleichbedeutend mit *VIRTUAL-MEMORY, d.h. standardmäßig wird der virtuelle Speicher zu Grunde gelegt.

SPACE = *ALET(...)
Bezeichnet Speicherseiten aus Datenräumen (Erweiterung des virtuellen Adressraums des Systems). Über den ALET (ACCESS-LIST-ENTRY-TOKEN) wird der Datenraum für den Task-Adressraum identifiziert.

IDENTIFIER = <x-string 1..8>
Die 4 Byte lange ALET wird sedezimal angegeben.

MODULE = *NONE / <name 1..32>
Name eines Moduls aus dem geladenen Benutzerprogramm, das vollständig im Standard-Dump-Format ausgegeben werden soll. Keine Angabe ist gleichbedeutend mit *NONE, d.h. es wird kein Modul zugewiesen.

USER-LIST-PROCEDURE = *NONE / <name 1..32 with-under> / <structured-name 1..32>
Bezeichnet den Namen eines PRODAMP-Programms, das sich in der aktuell eingestellten Benutzer-PRODAMP-Objektbibliothek befindet. Am Ende der Listenausgabe wird das Programm automatisch gestartet.

WINDOW = *NONE / <integer 4..99>
Das angegebene Fenster wird im aktuell eingestellten Layout auf Liste ausgegeben. Unterstützt werden die Fensternummern 4 - 9 und 21 - 99, die Fensternummern 10 - 20 sind 'reserviert'. Es können nur Fenster ausgegeben werden, die mit einem Standard-Format belegt sind (Dump-Format, Hex-Format, Char-Format, assembliert oder symbolisch).

DUMP-DIAGNOSIS = *NONE / *PREANALYSIS(...)
Legt fest, ob eine automatische Voranalyse der Diagnosedaten durchgeführt werden soll. Entspricht der Eingabe DIAG=YES/NO im LIST-Fenster, d.h. die Voranalyse-Prozedur DIAG wird gestartet oder nicht.
Die Diagnoseerkenntnisse über die diagnose-relevanten Speicherseiten, Fehlerursache, Fehlerumgebung und Fehlereingrenzung beeinflussen den Umfang der nachfolgenden Listenaufbereitung.

DUMP-DIAGNOSIS = *NONE
Es wird keine Voranalyse durchgeführt.

DUMP-DIAGNOSIS = *PREANALYSIS(...)
Die Voranalyse-Prozedur DIAG wird angestoßen. DIAG ist eine PRODAMP-Prozedur aus der System-PRODAMP-Bibliothek.
Die automatische Voranalyse wird für User- und Areadumps nicht unterstützt.

> **ERROR-DESCRIPTION = *NO / *YES**
> Legt fest, ob am Anfang der Ausgabeliste, im Rahmen der Systemübersicht, die Fehlerdeskriptoren im Format des Retrieval-Systems SIS ausgegeben werden sollen. Diese Deskriptoren können dann zur Duplikatsrecherche mit SIS verwendet werden.
>
> **ERROR-DESCRIPTION = *NO**
> Es werden keine Fehlerdeskriptoren ausgegeben, jedoch werden die bei der Voranalyse referenzierten und diagnose-relevanten Speicherseiten bei einer folgenden Minimum-Auswertung automatisch mitausgegeben.
>
> **ERROR-DESCRIPTION = *YES**
> Es werden Fehlerdeskriptoren ausgegeben.
> Entspricht der Eingabe DESCR=YES im LIST-Fenster, d.h. es wird die Voranalyse-Prozedur DESCR gestartet. DESCR ist eine PRODAMP-Prozedur aus der System-PRODAMP-Bibliothek.

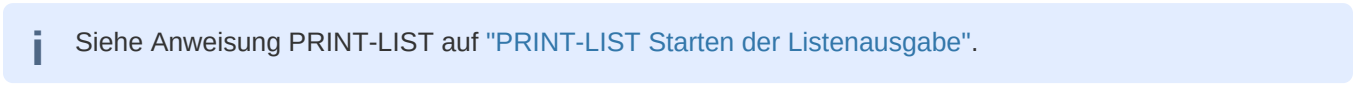
Beispiel

```
START-LIST-GENERATION
ADD-LIST GLOBAL-INFORMATION=*PAR(CONTROL-BLOCKS=*XVT)————————————————  (1)
ADD-LIST GLOBAL-INFORMATION=*PAR(MEMORY-AREAS=(*CL1,*CL3))————————————  (2)
ADD-LIST TASK-INFORMATION=*PAR(X'00040333,
         INFO=(CONTROL-BLOCKS=*TCB,PAGES=X'14'(*ALET=X'05')))————————————  (3)
ADD-LIST TASK-INFORMATION=*PAR(3UVW,INFO=*PAR(MEM=*CL6))————————————————  (4)
PRINT-LIST
```

(1)/(2) Ausgabe systemglobaler Informationen: XVT, Klasse-1- und Klasse-3-Speicher

(3) Für die Task mit TID = X'00040333' soll der TCB sowie die Seite 14 aus dem Datenraum mit ALET X'00000005' ausgegeben werden.

(4) Für die Task mit TSN 3UVW soll der Klasse-6-Speicher ausgegeben werden.



Siehe Anweisung PRINT-LIST auf "PRINT-LIST Starten der Listenausgabe".

5.6.1.2 ADD-SYMBOLS Zuweisen von Symbolen für die Ausgabe

Mit der Anweisung ADD-SYMBOLS werden zusätzliche Symbolelemente für die Aufbereitung bei symbolischer Ausgabe zugewiesen.

Format

```
ADD-SYMBOLS

 LIBRARY = *STD(...) / <filename 1..54 without-gen-vers>(...)

     *STD(...)

         |    ELEMENT = <filename 1..54 without-cat-user-gen-vers>(...)

         |         <filename>(...)

         |              |    VERSION = <composed-name_1..24_with-under>

     <filename 1..54 without-gen-vers>(...)

         |    ELEMENT = <filename 1..54 without-cat-user-gen-vers>(...)

         |         <filename>(...)

         |              |    VERSION = <composed-name_1..24_with-under>
```

Operandenbeschreibung

LIBRARY = *STD(...) / <filename 1..54 without-gen-vers>(...)
Gibt den Namen der Symbolbibliothek an, die das Symbolelement enthält.

LIBRARY = *STD(...)
Der Operandenwert *STD steht für die Standard-Symbolbibliothek von DAMP.

ELEMENT = <filename 1..54 without-cat-user-gen-vers>(...)
Bezeichnet den Namen des Elements, das die Symbole enthält. Die standardmäßig mit DAMP ausgelieferten Symbolelemente tragen den Namen des Produkts, zu dem sie gehören, z.B. BS2000 oder DAB.

VERSION = <composed-name_1..24_with-under>
Gibt die Versionsbezeichnung des Symbolelements an. Die standardmäßig mit DAMP ausgelieferten Symbolelemente tragen die Versionsbezeichnung der Produktversion, für die sie gültig sind, z.B. 200 oder 200.x1 mit x=A,B,... .

LIBRARY = <filename 1..54 without-gen-vers>(...)
Name der Symbolbibliothek.

ELEMENT = <filename 1..54 without-cat-user-gen-vers>(...)
Bezeichnet den Namen des Elements, das die Symbole enthält. Die standardmäßig mit DAMP ausgelieferten Symbolelemente tragen den Namen des Produkts, zu dem sie gehören, z.B. BS2000 oder DAB.

VERSION = <composed-name_1..24_with-under>
Gibt die Versionsbezeichnung des Symbolelements an. Die standardmäßig mit DAMP ausgelieferten Symbolelemente tragen die Versionsbezeichnung der Produktversion, für die sie gültig sind, z.B. 200 oder 200.x1 mit x=A,B,... .

Beispiele

```
ADD-SYMBOLS SYSSMB.DAMP.<ver>(BS2000(210.H1))
ADD-S *STD(BS2000(210))
```

5.6.1.3 ASSIGN-PRODAMP-LIBRARIES Zuweisen von Bibliotheken für den PRODAMP-Compiler und PRODAMP-Editor

Mit der Anweisung ASSIGN-PRODAMP-LIBRARIES kann für PRODAMP-Sourcen und PRODAMP-Objekte je eine Bibliothek zugewiesen werden.

Format

ASSIGN-PRODAMP-LIBRARIES
SOURCE-LIBRARY = *UNCHANGED / *STD / <filename 1..54 without-gen-vers>/ *PRODAMP-SYSTEM-LIBRARY ,OBJECT-LIBRARY = *UNCHANGED / *STD / *SOURCE-LIBRARY / <filename 1..54 without-gen-vers> / *PRODAMP-SYSTEM-LIBRARY

Operandenbeschreibung

SOURCE-LIBRARY = *UNCHANGED / *STD / <filename 1..54 without-gen-vers> / *PRODAMP-SYSTEM-LIBRARY
Bezeichnet den Namen der Bibliothek, die der PRODAMP-Editor zum Suchen und Ablegen von PRODAMP-Sourcen verwenden soll.

SOURCE-LIBRARY = *UNCHANGED
Der aktuelle Name der Bibliothek wird nicht geändert.

SOURCE-LIBRARY = *STD
Es wird die im OPTION-Fenster festgelegte Benutzer-Bibliothek gewählt, für welche wiederum der Standard SYS.USRDMP.DAMP.<ver> vorgegeben ist.

SOURCE-LIBRARY = <filename 1..54 without-gen-vers>
Der angegebene Bibliotheksname wird für die PRODAMP-Sourcen verwendet.

SOURCE-LIBRARY = *PRODAMP-SYSTEM-LIBRARY
Bezeichnet die PRODAMP System-Bibliothek. Dies ist bei Standardinstallationen die Bibliothek $TSOS.SYSDMP.DAMP.

OBJECT-LIBRARY = *UNCHANGED / *STD / *SOURCE-LIBRARY / <filename 1..54 without-gen-vers> / *PRODAMP-SYSTEM-LIBRARY
Bezeichnet den Namen der Bibliothek, die der PRODAMP-Compiler zum Ablegen bzw. das PRODAMP-Runtime-System zum Laden von PRODAMP-Objekten verwenden soll.

OBJECT-LIBRARY = *UNCHANGED
Der aktuelle Name der Bibliothek wird nicht geändert.

OBJECT-LIBRARY = *STD
Es wird die im OPTION-Fenster festgelegte Benutzer-Bibliothek gewählt, für welche wiederum der Standard SYS.USRDMP.DAMP.<ver> vorgegeben ist.

OBJECT-LIBRARY = *SOURCE-LIBRARY
Source- und Objektbibliothek sind identisch.

OBJECT-LIBRARY = <filename 1..54 without-gen-vers>
Der angegebene Bibliotheksname wird für die PRODAMP-Objekte verwendet.

OBJECT-LIBRARY = PRODAMP-SYSTEM-LIBRARY
Bezeichnet die PRODAMP System-Bibliothek. Dies ist bei Standardinstallationen die Bibliothek $TSOS.SYSDMP.DAMP.

Beispiele

```
ASSIGN-PRODAMP-LIBRARIES SOURCE=LIB.FOR.PRODAMP, OBJECT=*SOURCE
ASSIGN MY-SOURCE-LIB, MY-OBJECT-LIB
```

5.6.1.4 COMPILE-PRODAMP-PROCEDURE PRODAMP-Prozedur übersetzen

Mit der Anweisung COMPILE-PRODAMP-PROCEDURE wird eine PRODAMP-Prozedur aus einer PRODAMP-Bibliothek (also außerhalb des PRODAMP-Editors) im Dialog oder in einem Batch-Job übersetzt.

Format

```
COMPILE-PRODAMP-PROCEDURE

 SOURCE = *PRODAMP-USER-SOURCE-LIBRARY(...) / *PRODAMP-SYSTEM-LIBRARY(...) /
                  <filename 1..54 without-gen-vers>(...)
      *PRODAMP-USER-SOURCE-LIBRARY(...)
           |     ELEMENT = <name 1..32 with-under>
           |     VERSION = *HIGHEST-EXISTING / <integer 1..999>
      *PRODAMP-SYSTEM-LIBRARY(...)
           |     ELEMENT = <name 1..32 with-under>
           |     VERSION = *HIGHEST-EXISTING / <integer 1..999>
      <filename 1..54 without-gen-vers>(...)
           |     ELEMENT = <name 1..32 with-under>
           |     VERSION = *HIGHEST-EXISTING / <integer 1..999>
 ,OBJECT-LIBRARY = *PRODAMP-USER-OBJECT-LIBRARY / *PRODAMP-SYSTEM-LIBRARY /
                        *SOURCE-LIBRARY / <filename 1..54 without-gen-vers>
```

Operandenbeschreibung

SOURCE = *PRODAMP-USER-SOURCE-LIBRARY(...) / *PRODAMP-SYSTEM-LIBRARY(...) / <filename 1..54 without-gen-vers>(...)
Bezeichnet die PRODAMP-Bibliothek, aus der die PRODAMP-Prozedur entnommen werden soll.

SOURCE = *PRODAMP-USER-SOURCE-LIBRARY(...)
Es wird die aktuell eingestellte PRODAMP-Benutzer-Source-Bibliothek verwendet.

SOURCE = *PRODAMP-SYSTEM-LIBRARY(...)
Es wird die PRODAMP-System-Bibliothek verwendet.
Dies ist bei Standardinstallationen die Bibliothek $TSOS.SYSDMP.DAMP.

SOURCE = <filename 1..54 without-gen-vers>(...)
Es wird die Bibliothek mit dem angegebenen Namen verwendet.

ELEMENT = <name 1..32 with-under>
Name des Elements das kompiliert werden soll.

VERSION = *HIGHEST-EXISTING / <integer 1..999>
Version des Elements das kompiliert werden soll.

OBJECT-LIBRARY = *PRODAMP-USER-OBJECT-LIBRARY / *PRODAMP-SYSTEM-LIBRARY / *SOURCE-LIBRARY / <filename 1..54 without-gen-vers>
Bezeichnet die PRODAMP-Bibliothek, in die das PRODAMP-Programm gespeichert werden soll.

OBJECT-LIBRARY = *PRODAMP-USER-OBJECT-LIBRARY
Das PRODAMP-Programm wird in der aktuell eingestellten PRODAMP-Benutzer-Objekt-Bibliothek gespeichert.

OBJECT-LIBRARY = *PRODAMP-SYSTEM-LIBRARY
Das PRODAMP-Programm wird in der PRODAMP-System-Bibliothek gespeichert.
Dies ist bei Standardinstallationen die Bibliothek $TSOS.SYSDMP.DAMP.

OBJECT-LIBRARY = *SOURCE-LIBRARY
Das PRODAMP-Programm wird in der angegebenen Source-Bibliothek gespeichert.

OBJECT-LIBRARY = <filename 1..54 without-gen-vers>
Das PRODAMP-Programm wird in der Bibliothek mit dem angegebenen Namen gespeichert.

Beispiel

```
COMPILE-PRODAMP-PROCEDURE SOURCE=MYLIB(ELEMENT=MYPROG),OBJECT-LIBRARY=*SOURCE
```

5.6.1.5 DROP-REGISTER Registerverwendung für Disassembler festlegen

Mit der Anweisung DROP-REGISTER werden Registervereinbarungen, die mit der Anweisung USE-REGISTER getroffen wurden, wieder rückgängig gemacht. Bei disassemblierter Darstellung des angegebenen Moduls werden die Assembler-Befehle, die das angegebene Register verwenden, wieder mit Basis und Distanz aufbereitet (keine symbolische Darstellung der Befehlsadressen).

Die disassemblierte Ausgabe für x86-Code wird von der Anweisung DROP-REGISTER und der korrespondierenden Anweisung USE-REGISTER nicht unterstützt. Es können keine x86-Register angegeben werden.

Format

DROP-REGISTER
MODULE-NAME = <text 1..32> ,REGISTER = *ALL / <integer 0..15>

Operandenbeschreibung

MODULE-NAME = <name 1..32>
Bezeichnet das Modul bzw. den Kontrollblock (DSECT), für das/den die Registervereinbarungen zurückgesetzt werden sollen.

REGISTER = *ALL / <integer 0..15>
Bezeichnet ein oder mehrere Register, für die bei disassemblierter Ausgabe wieder die Basis-Distanz-Schreibweise verwendet werden soll. Bei *ALL wird bei allen Registern, die vorher in einer Anweisung USE-REGISTER genannt wurden, wieder die Basis-Distanz-Schreibweise verwendet.

REGISTER = <integer 0..15>
Bezeichnet ein /390-Mehrzweckregister.

Beispiele

```
DROP-REGISTER MODULE-NAME=DOPEN, REGISTER=12
DROP-REGISTER DOPEN, *ALL
```

5.6.1.6 EDIT-FILE Laden des EDT als Unterprogramm

Mit der Anweisung EDIT-FILE wird EDT als Unterprogramm aufgerufen und eine eventuell angegebene Datei in den EDT-Arbeitsbereich geladen.

Format

EDIT-FILE
NAME = *NONE / <filename 1..54 without-gen-vers>

Operandenbeschreibung

NAME = *NONE / <filename 1..54 without-gen-ver>
Bezeichnet den Namen einer Datei, die in den EDT-Arbeitsbereich gelesen werden soll. Falls der Arbeitsbereich schon belegt ist, wird zwar in den EDT verzweigt, aber die neue Arbeitsdatei nicht eingelesen. In diesem Fall erscheint eine Fehlermeldung in der EDT-Meldungszeile.

Beispiel

```
EDIT-FILE FILE.HUGO
```

5.6.1.7 END Beenden von DAMP

Mit der Anweisung END wird das Programm DAMP beendet.

Die Anweisung besitzt keine Operanden.

5.6.1.8 LOAD-MODULE Externes Unterprogramm laden

Mit der Anweisung LOAD-MODULE wird ein Modul aus einer Bibliothek geladen. Voraussetzung hierfür ist, dass das Modul über die VMOS-Linkage aufgerufen werden kann. Das Modul kann anschließend über die Anweisung START-MODULE beliebig oft zur Ausführung gebracht werden. Der DAMP-Anwender kann also selbst externe Prozeduren schreiben und diese auf die beschriebene Art unter DAMP ablaufen lassen.
Ein mit LOAD-MODULE geladenes Modul wird erst bei der Beendigung von DAMP entladen.
LOAD-MODULE kann innerhalb PRODAMP zur Performanceverbesserung und höheren Flexibilität von ENTER_MODULE eingesetzt werden (siehe "Standardprozeduren").

Format

LOAD-MODULE

```
LIBRARY = *STD(...) / *PRODAMP-USER-OBJECT-LIBRARY(...) / *PRODAMP-SYSTEM-LIBRARY(...) /
          <filename 1..54 without-gen-vers>(...)

        *STD(...)
                |       ELEMENT = <name 1..8>

        *PRODAMP-USER-OBJECT-LIBRARY(...)
                |       ELEMENT = <name 1..8>

        *PRODAMP-SYSTEM-LIBRARY(...)
                |       ELEMENT = <name 1..8>

        <filename 1..54 without-gen-vers>(...)
                |       ELEMENT = <name 1..8>
```

Operandenbeschreibung

LIBRARY = *STD(...) / *PRODAMP-USER-OBJECT-LIBRARY(...) / *PRODAMP-SYSTEM-LIBRARY(...) / <filename 1..54 without-gen-vers>(...)
Bezeichnet den Namen der Bibliothek, aus welcher das Modul geladen werden soll.

LIBRARY = *STD(...)
Die Nachladebibliothek von DAMP wird zugewiesen.

ELEMENT = <name 1..8>
Name des Moduls, der geladen werden soll.

LIBRARY = *PRODAMP-USER-OBJECT-LIBRARY(...)
Das Modul wird aus der aktuellen Benutzer-PRODAMP-Bibliothek geladen.

ELEMENT = <name 1..8>
Name des Moduls, das geladen werden soll.

LIBRARY = *PRODAMP-SYSTEM-LIBRARY(...)
Das Modul wird aus der aktuellen System-PRODAMP-Bibliothek geladen. Dies ist bei Standardinstallationen die Bibliothek $TSOS.SYSDMP.DAMP.

ELEMENT = <name 1..8>
Name des Moduls, das geladen werden soll.

LIBRARY = <filename 1..54 without-gen-vers>(...)
Name der Bibliothek.

ELEMENT = <name 1..8>
Name des Moduls, der geladen werden soll.

Beispiele

```
LOAD-MODULE *STD(MYOWNUTI)
LOAD-MODULE ELEM=MYOWNUTI
```

5.6.1.9 LOG-SESSION Diagnosemitschnitt einschalten

Mit der Anweisung LOG-SESSION werden alle auf diese Anweisung folgenden Ein- und Ausgaben der DAMP-Sitzung in die angegebene Datei mitgeschnitten.

Format

LOG-SESSION
LOGGING-FILE = *STD / <filename 1..54>

Operandenbeschreibung

LOGGING-FILE = *STD / <filename 1..54>
Name der Datei, in welche die mitgeschnittenen Ein- und Ausgaben protokolliert werden. Bei Angabe von *STD generiert DAMP den Dateinamen `S.LOG.DAMP.<ver>.<date>.<time>`.

Beispiel

```
LOG-SESSION LOGGING-FILE=LOG.HUGO
```

5.6.1.10 MODIFY-OBJECT-ASSUMPTIONS Ändern der Standardeinstellungen für das Diagnoseobjekt

Mit der Anweisung MODIFY-OBJECT-ASSUMPTIONS kann der Anwender die von DAMP selbstständig getroffenen Annahmen über die Daten des Diagnoseobjekts ändern. So ist es beispielsweise möglich, für die Disassemblierung einen anderen als den von DAMP gewählten Assembler-Befehlssatz einzustellen.

Format

```
MODIFY-OBJECT-ASSUMPTIONS

 ADDRESSING-MODE = *UNCHANGED / *PARAMETERS(...)

     *PARAMETERS(...)

         |  CONTROL-BLOCK = <name 1..32 with-under> / <structured-name 1..32>

         |  ,MODE = *STD / *XS31 / *NXS
```

Operandenbeschreibung

ADDRESSING-MODE = *UNCHANGED / *PARAMETERS(...)
Für die unter CONTROL-BLOCK angegebene Datenstruktur kann die Interpretation der Adressfelder geändert werden. Standardmäßig interpretiert DAMP alle Adressen als 31-Bit-Adressen (bei /390-Objekten) oder 32-Bit-Adressen (bei x86-Objekten). Ein Umschalten auf 24-Bit- oder 31-Bit-Adressen kann erwünscht sein.

ADDRESSING-MODE = *PARAMETERS(...)
Der Adressierungsmodus der unter CONTROL-BLOCK angegebenen Datenstruktur kann verändert werden.

CONTROL-BLOCK = <name 1..32 with-under> / <structured-name 1..32>
Bezeichnet den Namen der Datenstruktur (DSECT, Model, etc.), für die der neue Adressierungsmodus gelten soll.

MODE = *STD / *XS31 / *NXS
Bezeichnet den neuen Adressierungsmodus. Standardmäßig (*STD) werden die Adressen von DAMP als 31-Bit-Adressen (bei /390-Objekten) oder 32-Bit-Adressen (bei x86-Objekten) interpretiert. *XS31 steht für 31-Bit-Adressierung, *NXS für 24-Bit-Adressierung.

Beispiel

```
MODIFY-OBJECT-ASSUMPTION ADDRESSING-MODE=(ID1FCB,*NXS)
```

5.6.1.11 MODIFY-SCREEN-LAYOUT Neue Reihenfolge und Größe der Diagnosefenster festlegen

Mit der Anweisung MODIFY-SCREEN-LAYOUT wird die Reihenfolge und Größe der auf dem Bildschirm angezeigten Diagnosefenster verändert.

Format

```
MODIFY-SCREEN-LAYOUT

 FIRST-WINDOW = *UNCHANGED / <integer 0..99>(...)

      <integer>(...)

           |     SIZE = *UNCHANGED / <integer 2..19>

 ,SECOND-WINDOW = *UNCHANGED / <integer 0..99>(...)

      <integer>(...)

           |     SIZE = *UNCHANGED / <integer 2..19>

 ,THIRD-WINDOW = *UNCHANGED / <integer 0..99>(...)

      <integer>(...)

           |     SIZE = *UNCHANGED / <integer 2..19>

 ,FOURTH-WINDOW = *UNCHANGED / <integer 0..99>(...)

      <integer>(...)

           |     SIZE = *UNCHANGED / <integer 2..19>

 ,FIFTH-WINDOW = *UNCHANGED / <integer 0..99>(...)

      <integer>(...)

           |     SIZE = *UNCHANGED / <integer 2..19>

 ,SIXTH-WINDOW = *UNCHANGED / <integer 0..99>(...)

      <integer>(...)

           |     SIZE = *UNCHANGED / <integer 2..19>

 ,SEVENTH-WINDOW = *UNCHANGED / <integer 0..99>(...)

      <integer>(...)

           |     SIZE = *UNCHANGED / <integer 2..19>

 ,EIGHTH-WINDOW = *UNCHANGED / <integer 0..99>(...)

      <integer>(...)

           |     SIZE = *UNCHANGED / <integer 2..19>
```

```
,NINTH-WINDOW = *UNCHANGED / <integer 0..99>(...)

    <integer>(...)

        |    SIZE = *UNCHANGED / <integer 2..19>
```

Operandenbeschreibung

FIRST-WINDOW = *UNCHANGED / <integer 0..99>(...)...NINTH-WINDOW = *UNCHANGED / <integer 0..99>(...)
Das angegebene Fenster wird in der Reihenfolge der Fenster an derjenigen Stelle eingeordnet, die durch den Operandennamen bezeichnet wird (FIRST-,SECOND-, ..., NINTH-WINDOW). Beispielsweise wird durch SECOND-WINDOW=5 das Fenster mit der Nummer 5 an zweiter Position auf dem Bildschirm angezeigt, immer vorausgesetzt, dass noch genügend Platz auf dem Bildschirm vorhanden ist. Unterstützt werden die Fensternummern 0 - 9 und 21 - 99, die Fensternummern 10 - 20 sind 'reserviert'.

SIZE = *UNCHANGED / <integer 2..19>
Für das angegebene Fenster wird eine neue Größe festgelegt. Angaben von 2 bis 19 Zeilen sind möglich.

Beispiel

```
MODIFY-SCREEN-LAYOUT FIRST-WINDOW = 9(SIZE=4),
                     SECOND-WINDOW = 33,
                     THIRD-WINDOW = 4
```

Einschränkung

Werden die Operandennamen FIRST-WINDOW, ... NINTH-WINDOW im Anweisungsaufruf verwendet, dürfen diese nur in lückenloser Reihenfolge verwendet werden. Es muss mit dem Operanden FIRST-WINDOW begonnen werden.
Folgende Aufrufe sind also nicht erlaubt:

```
MODIFY-SCREEN-LAYOUT THIRD-WINDOW = 1(SIZE=4)
MODIFY-SCREEN-LAYOUT FIRST-WINDOW = 4,
                     THIRD-WINDOW = 5
```

Es empfiehlt sich für die Eingaben Stellungsoperanden zu benutzen:

```
MODIFY-SCREEN-LAYOUT 9, 4, 6, 33(2)
```

5.6.1.12 OPEN-DIAGNOSIS-OBJECT Eröffnen eines Diagnoseobjekts für die Bearbeitung

Mit der Anweisung OPEN-DIAGNOSIS-OBJECT wird ein Diagnoseobjekt (BS2000-Dump, SELF-LOADER-Dump, PAM-Datei, aktives BS2000-System) zur Analyse zugewiesen.

Zum Öffnen eines BS2000-Objekts benötigt DAMP die Standard-BS2000-Symbole der jeweiligen Version.

Format

OPEN-DIAGNOSIS-OBJECT

```
OBJECT = *SYSTEM(...) / *#0(...) / *#1(...) /... / *#9(...) / <filename 1..54 without-gen-vers with-wild(80)>(...)
    *SYSTEM(...)
        |  KIND-OF-SYSTEM = *BS2000
    *#0(...) / *#1(...) /... / *#9(...)
        |  KIND-OF-OBJECT = *STD / *BS2000 / *SELF-LOADER / *PAM
    <filename 1..54 without-gen-vers with-wild(80)>(...)
        |  KIND-OF-OBJECT = *STD / *BS2000 / *SELF-LOADER / *PAM
,SYMBOLS = *STD(...) / <filename 1..54 without-gen-vers>(...)
    *STD(....) / <filename 1..54>(...)
        |  ELEMENT = *BS2000(...) / <filename 1..54 without-cat-user-gen-vers>(...)
        |      *BS2000(...)
        |          |  VERSION= *STD / <composed-name_1..24_with-under>
        |      <filename 1..54 without-cat-user-gen-vers>(...)
        |          |  VERSION= *STD / <composed-name_1..24_with-under>
```

Operandenbeschreibung

OBJECT = *SYSTEM(...) / *#0(...) / *#1(...) / ... / *#9(...) / <filename 1..54 without-gen-vers with-wild(80)>(...)
Bezeichnet den Typ des Diagnoseobjekts.

OBJECT = *SYSTEM(...)
Mit *SYSTEM wird das aktive System als Diagnoseobjekt zugewiesen. Für den Zugriff auf das aktive System muss eine Testprivilegierung vorhanden sein, die vom Systemverwalter mit dem Kommando MODIFY-USER-ATTRIBUTES für den Benutzer eingerichtet ist und vor Programmstart mit MODIFY-TEST-OPTIONS PRIVILEG=*PAR(READ=8,WRITE=1) aktiviert wird.

KIND-OF-SYSTEM = *BS2000
Der Operand ermöglicht die Diagnose im aktiven BS2000-System.

OBJECT = *#0(...) / *#1(...) / ... / *#9(...) / <filename 1..54 without-gen-vers with-wild(80)>(...)
Eine Datei wird als Diagnoseobjekt zugewiesen. Die Schlüsselwörter *#0 bis *#9 (bzw. kurz nur #0 bis #9) werden als Linknamen interpretiert, wenn vorher das Kommando ADD-FILE-LINK mit diesen Linknamen eingegeben wurde. Ferner können alle bei SHOW-FILE-ATTRIBUTES erlaubten Wildcards angegeben werden. Wird durch die Wildcardangabe bzw. durch Teilqualifizierung eine Datei eindeutig bestimmt, wird diese geöffnet, andernfalls wird die Meldung ausgegeben, dass die Angabe unterqualifiziert war. Enthält der Dateiname den Ausdruck $TSN, wird dieser durch die TSN der Task ersetzt, unter welcher DAMP gerade läuft.

KIND-OF-OBJECT = *STD / *BS2000 / *SELF-LOADER / *PAM
Gibt DAMP die Datenstruktur des Diagnoseobjekts bekannt.

KIND-OF-OBJECT = *STD
Bei Dumpdateien mit mehreren Objekten (z.B. bei einem VM2000-Gesamtsled) wird im Dialog zunächst das Status-Fenster (W2) im Modus INF mit der möglichen Auswahl angezeigt. Durch Markieren kann das gewünschte Diagnoseobjekt ausgewählt werden. Ansonsten wirkt *STD wie *BS2000.

KIND-OF-OBJECT = *BS2000
DAMP sucht zur Diagnose aus der Dumpdatei ein BS2000-Objekt. Wird kein BS2000-Objekt gefunden, wird die Dumpdatei als SELF-LOADER geöffnet.

KIND-OF-OBJECT = *SELF-LOADER
Es können beliebige Dateien mit Dumpfile-Struktur geöffnet werden. DAMP bietet für SELF-LOADER-Dumps keine Automatismen an. Alle Adressen werden als reale Adressen interpretiert, sodass lediglich reale Speicherbereiche in den üblichen Formaten dargestellt werden können (siehe auch "Bearbeitung von SLEDs ohne BS2000-Struktur"). Weitergehende Auswertungen sind nur über PRODAMP-Prozeduren zu verwirklichen.

KIND-OF-OBJECT = *PAM
Dateien ohne Dumpfile-Struktur können nur als PAM-Datei geöffnet werden.
Auf die Daten einer PAM-Datei kann mit den im Abschnitt „Dateien mit PAM-Format bearbeiten" beschriebenen Mechanismen zugegriffen werden.

SYMBOLS = *STD(...) / <filename 1..54 without-gen-vers>(...)
Normalerweise muss der Operand SYMBOLS nicht versorgt werden.
Es sind dann die Default-Einstellungen (SYMBOLS=*STD(*BS2000(*STD))) wirksam, die bewirken, dass DAMP die BS2000-Symbole aus der Standard-Bibliothek lädt, die zu dem geöffneten BS2000-Objekt passen. Wurde KIND-OF-OBJECT=*SELF-LOADER oder *PAM verwendet, werden keine Symbole geladen. Wird der Operand SYMBOLS mit anderen Werten verwendet, versucht DAMP immer (auch für KIND-OF-OBJECT=*SELF-LOA-DER oder *PAM) Symbole zu laden.

Mit dem Operanden SYMBOLS werden die Symbole zugewiesen, die DAMP zum symbolischen Öffnen des ausgewählten Diagnoseobjekts benötigt. Zusätzliche Symbole für die anschließende Bearbeitung können mit der Anweisung ADD-SYMBOLS zugewiesen werden.

SYMBOLS = *STD(...) / <filename>(...)
Als Bibliothek wird die Standard-Symbolbibliothek gewählt, falls *STD angegeben wird.

Sollen für ein BS2000-Objekt andere Symbole verwendet werden oder werden für ein Nicht-BS2000-Objekt spezielle Symbole benötigt, kann eine Symbolbibliothek vereinbart werden, die das Symbolelement enthält.

ELEMENT = *BS2000(...)
Der Elementname BS2000 wird gewählt. In der Standard-Symbolbibliothek befinden sich unter diesem Elementnamen die Symbole für die Analyse eines BS2000-Systems.

VERSION = *STD
Die zum geöffneten Diagnoseobjekt passende Elementversion wird gewählt.

VERSION = <composed-name_1..24_with-under>
Die angegebene Elementversion wird gewählt.
Die Symbole in der Standard-Symbolbibliothek haben die Versionsbezeichnung des Produkts, für das sie gültig sind.

ELEMENT = <filename 1..54 without-cat-user-gen-vers>(...)
Name des Elements, das die Symbole enthält. Die mit DAMP ausgelieferten Symbolelemente haben den Namen des Produkts, zu dem sie gehören.

VERSION = *STD
Die zum geöffneten Diagnoseobjekt passende Elementversion wird gewählt.

VERSION = <composed-name_1..24_with-under>
Die angegebene Elementversion wird gewählt.
Die Symbole in der Standard-Symbolbibliothek haben die Versionsbezeichnung des Produkts, für das sie gültig sind.

Beispiele

```
OPEN-DIAGNOSIS-OBJECT OBJECT=*SYSTEM
OPEN OBJECT=$SLED.SLED.1234,SYMBOLS=syssmb.damp.<ver>(BS2000(210))
```

5.6.1.13 PRINT-LIST Starten der Listenausgabe

Mit der Anweisung PRINT-LIST wird eine mit den Anweisungen START-LIST-GENERATION, ADD-LIST-OBJECTS oder REMOVE-LIST-OBJECTS vorher zusammengestellte Objektmenge ausgegeben. Die Ausgabe kann in eine Datei bzw. nach SYSLST erfolgen.

Format

PRINT-LIST
OUTPUT = *SYSLST / <filename 1..54>

Operandenbeschreibung

OUTPUT = *SYSLST / <filename 1..54>
Bezeichnet das Ausgabeziel.

OUTPUT = *SYSLST
Die Ausgabe wird standardmäßig nach SYSLST geleitet.

OUTPUT = <filename 1..54>
Die Ausgabe erfolgt in eine katalogisierte Datei (SAM-Datei).

Hinweis

Das Ausdrucken einer Liste ist im Batch-Betrieb folgendermaßen abzuwickeln:

```
START-LIST-GENERATION ——————————————————————————————————————  (1)
ADD-LIST-OBJECTS ... ———————————————————————————————————————  (2)
ADD-LIST-OBJECTS ...
REMOVE-LIST-OBJECTS ...
PRINT-LIST ... ——————————————————————————————————————————————  (3)
```

(1) Einleiten der Listenerzeugung
(2) Zusammenstellen des Listenumfangs
(3) Ausdruck der Liste

Im Dialogbetrieb bewirkt die Anweisung START-LIST-GENERATION die Anzeige des LIST-Fensters, in dem die Zusammenstellung des Listenumfangs durch Markieren und Ausfüllen von Masken vorgenommen werden kann. An Stelle der Anweisung PRINT-LIST kann im Dialog auch ein entsprechendes Feld markiert werden.

5.6.1.14 PRINT-LOGGING-FILE Aufbereiten und Ausdrucken einer Logging-Datei

Mit der Anweisung PRINT-LOGGING-FILE wird das Ausgabelayout einer Logging-Datei festgelegt und der Ausdruck der Datei angestoßen.

Format

```
PRINT-LOGGING-FILE

 LOGGING-FILE = <filename 1..54 without-gen-vers>

 ,OUTPUT-FILE = *STD / <filename 1..54 without-gen-vers>

 ,LAYOUT = *STD / *TOMDOC / *OPTIONS(...)

     *OPTIONS(...)

         |   LINES-PER-PAGE = <integer 1..100>

         |   ,LOWER-CHARACTERS = *YES / *NO

         |   ,NIL-CHARACTERS = <x-string 1..2> / <c-string 1..1>

         |   ,TRASH-CHARACTERS = <x-string 1..2> / <c-string 1..1>

 ,PRINT = *NO / *YES / *ERASE
```

Operandenbeschreibung

LOGGING-FILE = <filename 1..54 without-gen-vers>
Bezeichnet den Namen der zu bearbeitenden Logging-Datei.

OUTPUT-FILE = *STD / <filename 1..54 without-gen-vers>
Bezeichnet den Namen der zu erzeugenden Ausgabedatei.
*STD bedeutet Ausgabe nach SYSLST.

LAYOUT = *STD / *TOMDOC / *OPTIONS(...)
Bezeichnet das Format der Ausgabedatei.

LAYOUT = *STD
Setzt Standardwerte für das Format der Ausgabedatei.

LAYOUT = *TOMDOC
Die Ausgabedatei wird mit Steuerzeichen des Textverarbeitungssystems TOMDOC aufbereitet.

LAYOUT = *OPTIONS(...)
Das Layout der Ausgabedatei kann mit nachfolgenden Optionen eingestellt werden.

LINES-PER-PAGE = <integer 1..100>
Bestimmt die Anzahl der Zeilen, die auf eine Seite gedruckt werden sollen.

LOWER-CHARACTERS = *YES / *NO
Legt fest, ob Kleinbuchstaben ausgedruckt werden (*YES) oder ob sie vor dem Ausdrucken in Großbuchstaben umgewandelt werden sollen (*NO).

NIL-CHARACTERS = <x-string 1..2> / <c-string 1..1>
Legt die Darstellung des NIL-Zeichens fest.

TRASH-CHARACTERS = <x-string 1..2> / <c-string 1..1>
Legt die Darstellung des Schmierzeichens fest.

PRINT = *NO / *YES / *ERASE
Gibt an, ob die erzeugte Ausgabedatei ausgedruckt und anschließend gelöscht werden soll.

PRINT = *NO
Die erzeugte Ausgabedatei wird nicht ausgedruckt. Voreinstellung.

PRINT = *YES
Die erzeugte Ausgabedatei wird ausgedruckt, aber anschließend nicht gelöscht.

PRINT = *ERASE
Die erzeugte Ausgabedatei wird ausgedruckt und anschließend gelöscht.

Beispiele

```
PRINT-LOGGING-FILE LOGGING-FILE=LOG.HUGO,OUTPUT-FILE=LOG.ABC.EDITED,PRINT=*ERASE
```

5.6.1.15 REMOVE-LIST-OBJECTS Steuerung der Listenausgabe

Mit der Anweisung REMOVE-LIST-OBJECTS können die vorher mit ADD-LIST-OBJECTS zugewiesenen Bereiche von der Listenausgabe wieder ausgenommen werden.
Die Möglichkeiten der Rücknahme sind jedoch relativ pauschal, d.h. es lässt sich z.B. die Ausgabe der TASK-INFORMATION nur für bestimmte Tasks zurücknehmen. Sollen für eine Task nur einzelne Bereiche von der Ausgabe ausgeschlossen werden, muss zuerst die gesamte Task mit REMOVE-LIST-OBJECTS zurückgenommen werden. Im Anschluss daran müssen diejenigen Bereiche mit ADD-LIST-OBJECTS zugewiesen werden, die ausgegeben werden sollen. Alle für die Ausgabe nicht gewünschten Bereiche werden nicht genannt.

Format

REMOVE-LIST-OBJECTS
GLOBAL-INFORMATION = *NONE / *ALL / *TRACES / *MAPS / *CONTROL-BLOCKS / *MEMORY-AREAS / *PAGES / *MODULE ,TASK-INFORMATION = *NONE / *ALL / <x-string 1..8> / <alphanum-name 1..4> / <c-string 1..4> ,USER-LIST-PROCEDURE = *NONE / *ALL / <name 1..32 with-under> / <structured-name 1..32> ,WINDOW = *NONE / *ALL / <integer 4..99> ,DUMP-DIAGNOSIS = *NONE / *ALL

Operandenbeschreibung

GLOBAL-INFORMATION = *NONE / *ALL / *TRACES / *MAPS / *CONTROL-BLOCKS / *MEMORY-AREAS / *PAGES / *MODULE
Die Ausgabe der globalen Informationen kann vollständig oder gezielt für bestimmte Bereiche unterdrückt werden.

GLOBAL-INFORMATION = *NONE
Falls keine Auswahl getroffen wird, werden die mit ADD-LIST-OBJECTS vereinbarten Bereiche ausgegeben.

GLOBAL-INFORMATION = *ALL
Mit dieser Angabe werden nicht nur alle mit ADD-LIST-OBJECTS vereinbarten Bereiche von der Ausgabe ausgenommen, sondern die voreingestellten Standardbereiche (siehe Mini-Auswertung) werden ebenfalls nicht ausgegeben.

GLOBAL-INFORMATION = *TRACES
Die mit ADD-LIST-OBJECTS vereinbarten globalen Traces werden nicht ausgegeben.

GLOBAL-INFORMATION = *MAPS
Informationen über die geladenen CSECTs bzw. über den Korrekturstand des Systems, die mit ADD-LIST-OBJECTS für die Ausgabe ausgewählt wurden, werden nicht ausgegeben.

GLOBAL-INFORMATION = *CONTROL-BLOCKS
Alle Kontrollblöcke, die mit ADD-LIST-OBJECTS für die Ausgabe ausgewählt wurden, werden nicht ausgegeben.

GLOBAL-INFORMATION = *MEMORY-AREAS
Die mit ADD-LIST-OBJECTS für die Ausgabe ausgewählten Bereiche des virtuellen Adressraums werden nicht ausgegeben.

GLOBAL-INFORMATION = *PAGES
Seiten bzw. Bereiche des Adressraums (virtuell, Hauptspeicher, HSA, ...), die mit ADD-LIST-OBJECTS für die Ausgabe ausgewählt wurden, werden nicht ausgegeben.

GLOBAL-INFORMATION = *MODULE
Der Speicherbereich des angegebenen Moduls, der mit ADD-LIST-OBJECTS für die Ausgabe ausgewählt wurde, wird nicht ausgegeben.

TASK-INFORMATION = *NONE / *ALL / <x-string 1..8 > /<alphanum-name 1..4> / <c-string 1..4>
Die Ausgabe von taskspezifischen Informationen kann vollständig oder gezielt für bestimmte Tasks unterdrückt werden.

TASK-INFORMATION = *NONE
Falls keine Auswahl getroffen wird, werden die mit ADD-LIST-OBJECTS vereinbarten Informationen der dort angegebenen Tasks ausgegeben.

TASK-INFORMATION = *ALL
Mit dieser Angabe werden nicht nur alle mit ADD-LIST-OBJECTS vereinbarten Informationen der dort angegebenen Tasks von der Ausgabe ausgenommen, sondern die Informationen der standardmäßig voreingestellten Tasks werden ebenfalls unterdrückt.

TASK-INFORMATION = <x-string 1..8 >
Der angegebene sedezimale Wert wird als TID der gewünschten Task interpretiert, deren Informationen nicht ausgegeben werden sollen.

TASK-INFORMATION = <alphanum-name 1..4>
Der angegebene alphanumerische Name wird als TSN der gewünschten Task interpretiert, deren Informationen nicht ausgegeben werden sollen.

TASK-INFORMATION = <c-string 1..4 >
Der angegebene Zeichenstring wird als TSN der gewünschten Task interpretiert, deren Informationen nicht ausgegeben werden sollen.

USER-LIST-PROCEDURE = *NONE / *ALL / <name 1..32 with-under> /<structured-name 1..32>
Bezeichnet den Namen eines PRODAMP-Programms aus der aktuell eingestellten PRODAMP-Objektbibliothek, das am Ende der Listenausgabe automatisch gestartet werden soll.

USER-LIST-PROCEDURE = *NONE
Falls keine Auswahl getroffen wird, werden die mit ADD-LIST-OBJECTS vereinbarten Benutzerprogramme nach Ende der Listenausgabe automatisch gestartet.

USER-LIST-PROCEDURE = *ALL
Keines der mit ADD-LIST-OBJECTS vereinbarten Benutzerprogramme wird am Ende der Listenausgabe gestartet.

USER-LIST-PROCEDURE = <name 1..32 with-under> / <structured-name 1..32>
Das angegebene Benutzerprogramm wird am Ende der Listenausgabe nicht gestartet.

WINDOW = *NONE / *ALL / <integer 4..99>
Bezeichnet ein Fenster, dessen Layout mit ADD-LIST-OBJECTS für die Ausgabe auf Liste ausgewählt wurde.
Es können alle oder gezielt einzelne Fenster von der Ausgabe ausgeschlossen werden.

WINDOW = *NONE
Alle mit ADD-LIST-OBJECTS vereinbarten Fenster werden auf Liste ausgegeben.

WINDOW = *ALL
Keines der mit ADD-LIST-OBJECTS vereinbarten Fenster wird auf Liste ausgegeben.

WINDOW = <integer 4..99>
Das angegebene Fenster wird nicht auf Liste ausgegeben. Unterstützt werden die Fensternummern 4 - 9 und 21 - 99, die Fensternummern 10 - 20 sind 'reserviert'.

DUMP-DIAGNOSIS = *NONE / *ALL
Der Aufruf der automatischen Voranalyse kann abgeschaltet (*ALL) werden.

Beispiel

```
REMOVE-LIST-OBJECTS TASK-INFORMATION=X'0004000E',WINDOW=*ALL
```

Nach der Anweisung PRINT-LIST wird die Auswahl der auszudruckenden Bereiche wieder auf die standardmäßige Voreinstellung zurückgesetzt. Der Umfang der Standard-Ausgabe-Liste kann jedoch mit der Anweisung REMOVE-LIST-OBJECTS noch weiter reduziert werden (siehe Beschreibung der Operanden GLOBAL-INFORMATION und TASK-INFORMATION).

5.6.1.16 REPEAT-SESSION Diagnose-Mitschnitt wiederabspielen

Mit der Anweisung REPEAT-SESSION werden die in einer Datei mitgeschnittenen Ein- und Ausgaben einer DAMP-Sitzung wieder abgespielt.

Format

REPEAT-SESSION
LOGGING-FILE = <filename 1..54>

Operandenbeschreibung

LOGGING-FILE = <filename 1..54>
Gibt den Namen der Datei an, die den abzuspielenden Mitschnitt enthält.

Beispiel

```
REPEAT-SESSION LOGGING-FILE=LOG.HUGO
```

5.6.1.17 RESUME-PRODAMP-PROGRAM Fortsetzen eines unterbrochenen PRODAMP-Programms

Mit der Anweisung RESUME-PRODAMP-PROGRAM wird ein unterbrochenes PRODAMP-Programm fortgesetzt. War das Programm nicht unterbrochen, wird es von Anfang an neu gestartet.

Format

RESUME-PRODAMP-PROGRAM
NAME = *INTERRUPTED / <integer 4..99> / <structured-name 1..32 / <name 1..32 with-under> ,PARAMETERS = *NONE / list-poss(32): <integer -2147483648..2147483647> / <x-string 1..8> / <c-string 1..80>

Operandenbeschreibung

NAME = *INTERRUPTED / <integer 4..99> / <structured-name 1..32> /<name 1..32 with-under>
Gibt an, welches PRODAMP-Programm fortgesetzt werden soll.

NAME = *INTERRUPTED
Falls mehrere Programme unterbrochen worden sind, wird das zuletzt unterbrochene Programm fortgesetzt.

NAME = <integer 4..99>
Bezeichnet die Nummer eines Fensters, das dem PRODAMP-Compiler zugewiesen sein sollte. Das Objekt, das vom Compiler bereits erzeugt wurde, soll fortgesetzt werden. Unterstützt werden die Fensternummern 4 - 9 und 21 - 99, die Fensternummern 10 - 20 sind 'reserviert'.

NAME = <structured-name 1..32> / <name 1..32 with-under>
Bezeichnet den Namen eines Programms, das unter diesem Namen mit der Anweisung START-PRODAMP-PROGRAM bereits gestartet wurde oder unter diesem Namen vom PRODAMP-Compiler fehlerfrei übersetzt vorliegt.

PARAMETERS = *NONE / list-poss(32): <integer -2147483648..2147483647> /<x-string 1..8> / <c-string 1..80>
Mit diesem Operanden kann eine Liste von bis zu 32 Parametern an das unterbrochene PRODAMP-Programm übergeben werden. Die Parameter können numerisch, sedezimal oder als Charakter-String vereinbart werden. Zur Übernahme der Werte durch dasPRODAMP-Programm siehe "Pseudostrukturen".

Beispiele

```
RESUME-PRODAMP-PROGRAM NAME=*INTERRUPTED, PAR=(X'10000','HUGO')
RESUME-PRODAMP-PROGRAM PROC1,(12,13,14)
```

5.6.1.18 SEARCH-IN-SUBSYSTEM CSECT-Suche in einem Subsystem

Mit der Anweisung SEARCH-IN-SUBSYSTEM kann die CSECT-Suche auf ein einziges Subsystem eingeschränkt werden. Die Einschränkung kann mit der Anweisung auch wieder aufgehoben werden. Wird keine Subsystem-Version angegeben, wird das erste Subsystem mit dem angegebenen Namen aus der Subsystem-Liste verwendet. Alternativ zur Angabe SUBSYSTEM/VERSION kann der eindeutige Kontextname angegeben werden.

Für die Relativierung von Adressen werden aber trotzdem alle Subsysteme berücksichtigt.

Format

```
SEARCH-IN-SUBSYSTEM

SUBSYSTEM = *ALL / <name 1..8>(...) /<c-string 1..8>(...) / *CONTEXT(...)

    <name 1..8>(...)

        |    VERSION = *FIRST-FOUND / <filename 1..8> / <c-string 1..8>

    *CONTEXT(...)

        |    CONTEXT = <text 1..32>
```

Operandenbeschreibung

SUBSYSTEM = *ALL / <name 1..8>(...) / <c-string 1..8>(...) / *CONTEXT(...)
Gibt den Subsystemnamen an.

SUBSYSTEM = *ALL
Mit der Angabe wird eine zuvor getroffene Einschränkung wieder aufgehoben.

SUBSYSTEM = <name 1..8>(...) / <c-string 1..8>(...)
Bezeichnet den Namen des Subsystems, wie er im SUSY-Fenster ausgegeben wird.

> **VERSION = *FIRST-FOUND / <filename 1..8> / <c-string 1..8>**
> Angabe der Version des gewählten Subsystems. Wird keine Subsystem-Version angegeben, wird das erste Subsystem mit dem angegebenen Namen aus der Subsystem-Liste verwendet.

SUBSYSTEM = *CONTEXT(...)
Angabe des dem Subsystem entsprechenden Kontextnamens.

> **CONTEXT = <text 1..32>**
> Der hier anzugebende Kontextname kann dem SUSY-Fenster entnommen werden (siehe Einstellung CTX auf "Informationen über Subsysteme ausgeben (Spezialfenster SUSY)").

5.6.1.19 SHOW-EDITED-INFORMATION Ausgabe von aufbereiteten Diagnosedaten

Mit der Anweisung SHOW-EDITED-INFORMATION werden aufbereitete Diagnosedaten in ein angegebenes Dumpfenster ausgegeben.

Format

```
SHOW-EDITED-INFORMATION

 INFORMATION = *STORAGE-EDIT / *AUDIT-TABLE-EDIT / *TRACE-TABLE-EDIT / *TASK-TABLES /
                       *SUBSYSTEM-INFORMATION / *DUMPED-SYSTEM-FILE / *MEMORY-ATTRIBUTES /
                       *SPECIAL(...)

     *SPECIAL(...)

         |    NAME = <structured-name 1..255>

 ,WINDOW = *NEXT-FREE / <integer 4..99>
```

Operandenbeschreibung

INFORMATION = *STORAGE-EDIT / *AUDIT-TABLE-EDIT / *TRACE-TABLE-EDIT /*TASK-TABLES / *SUBSYSTEM-INFORMATION / *DUMPED-SYSTEM-FILE /*MEMORY-ATTRIBUTES / *SPECIAL(...)
Gibt die Aufbereitungsart der Diagnosedaten an.

INFORMATION = *STORAGE-EDIT
Die Daten werden im Layout des DAMP-Standard-Fensters aufbereitet. Eine bereits vorher gewählte Spezialaufbereitung der Daten wird rückgängig gemacht.

INFORMATION = *AUDIT-TABLE-EDIT
Sind in einem Dump AUDIT-Tabellen (sowohl Hardware- als auch Linkage-AUDIT) enthalten, wird die erste gefundene Tabelle aufbereitet und in dem angegebenen Fenster angezeigt.
Im Audit-Fenster besteht die Möglichkeit, sich anschließend andere AUDIT-Tabellen anzeigen zu lassen. Weitere Informationen siehe "Informationen über AUDIT-Tabellen (Spezialfenster AUDIT)".

INFORMATION = *TRACE-TABLE-EDIT
Bei Sleds und SNAP-Dumps wird die gesamte, bei Systemdumps die taskspezifische Trace-Table aufbereitet und in dem angegebenen Fenster angezeigt. Weitere Informationen siehe "System-Trace-Table ausgeben (Spezialfenster TRACE)".

INFORMATION = *TASK-TABLES
Für alle im System aktiven Tasks wird der Inhalt mehrerer frei wählbarer Felder in einer tabellarischen Übersicht angezeigt. Weitere Informationen siehe "Tabellen taskspezifischer Werte ausgeben (Spezialfenster TABLE)".

INFORMATION = *SUBSYSTEM-INFORMATION
Von den im System gerade aktiven Subsysteme werden ausgewählte Daten aufbereitet und ausgegeben. Weitere Informationen siehe "Informationen über Subsysteme ausgeben (Spezialfenster SUSY)".

INFORMATION = *DUMPED-SYSTEM-FILE
Anzeigen, Editieren und Generieren von Systemdateien und Dumpsections. Weitere Informationen siehe "Informationen über Systemdateien sowie Sections der Dumpdatei (Spezialfenster FILE)".

INFORMATION = *MEMORY-ATTRIBUTES
Informiert über die Speicherbelegung und Speicherattribute des aktuellen Adressraums. Weitere Informationen siehe "Tabellen taskspezifischer Werte ausgeben (Spezialfenster TABLE)".

INFORMATION = *SPECIAL(...)
Mit diesem Operandenwert können weitere Spezialaufbereitungen ad hoc verfügbar gemacht werden. Hierzu muss der Name eines nachladbaren DAMP-Moduls angegeben werden, der diese Spezialfunktion verwirklicht. Das Modul wird aus der Bibliothek SYSLNK.DAMP.<ver> nachgeladen.

NAME = <structured-name 1..255>
Bezeichnet den Modulnamen aus der Bibliothek SYSLNK.DAMP.<ver>, der die gewünschte Spezialfunktion verwirklicht.
Dieser Operand wird nur noch aus Kompatibilitätsgründen angeboten.

WINDOW = *NEXT-FREE / <integer 4..99>
Bezeichnet das Fenster, in dem die aufbereiteten Diagnose-Daten ausgegeben werden sollen. Unterstützt werden die Fensternummern 4 - 9 und 21 - 99, die Fensternummern 10 - 20 sind 'reserviert'.
Die Fenster 4 - 9 werden bei Angabe von *NEXT-FREE in der Reihenfolge 9, 8, ..., 4 verwendet.

Beispiel

```
SHOW-EDITED-INFORMATION INFORMATION=*TRACE-TABLE-EDIT, WINDOW=99
```

5.6.1.20 SHOW-LAST-STATEMENT Anzeigen der letzten DAMP-Anweisung

Mit der Anweisung SHOW-LAST-STATEMENT wird die letzte eingegebene DAMP-Anweisung in der Kommandozeile von DAMP noch einmal angezeigt, um gegebenenfalls verändert und erneut abgeschickt zu werden.

Die Anweisung besitzt keine Operanden.

Durch mehrmalige Eingabe der Anweisung SHOW-LAST-STATEMENT können Sie in der Anweisungs-Historie zurückblättern.

5.6.1.21 SHOW-PRODAMP-LIBRARIES Anzeigen der PRODAMP-Bibliotheken

Mit der Anweisung SHOW-PRODAMP-LIBRARIES werden die Namen der aktuell für PRODAMP zugewiesenen Benutzer-Bibliotheken in den Meldungszeilen des DAMP-Bildschirms angezeigt.

Die Anweisung besitzt keine Operanden.

5.6.1.22 SHOW-SUBSYSTEM-FOR-SEARCH Anzeigen des eingestellten Subsystems

Mit der Anweisung SHOW-SUBSYSTEM-FOR-SEARCH kann das mit SEARCH-IN-SUBSYSTEM eingestellte Subsystem angezeigt werden.

Die Anweisung besitzt keine Operanden.

5.6.1.23 START-LIST-GENERATION Einleiten der Listenausgabe

Mit der Anweisung START-LIST-GENERATION wird die Ausgabe einer Liste eingeleitet. Die Auswahl der auszugebenden Bereiche erfolgt über das LIST-Fenster oder über die Anweisungen ADD-LIST-OBJECTS bzw. REMOVE-LIST-OBJECTS. Die eigentliche Ausgabe der Liste muss mit der Anweisung PRINT-LIST angestoßen werden.

Format

```
START-LIST-GENERATION

 FILES-TO-EVALUATE = *CURRENT / *#0 / *#1 / ... / *#9 / <filename 1..54 without-gen-vers with-wild(80)>(...)

       <filename>(...)

             |       EVALUATE = *FIRST-MATCH / *ALL-MATCHES

 ,WINDOW = *NEXT-FREE / <integer 4..99>
```

Operandenbeschreibung

FILES-TO-EVALUATE = *CURRENT / *#0 / *#1 / ... / *#9 / <filename 1..54 without-gen-vers-with-wild(80)>(...)
Bezeichnet die Dateien, für die eine Listenaufbereitung erfolgen soll. Im Dialog ist dieser Operand wirkungslos.

FILES-TO-EVALUATE = *CURRENT
Die zurzeit geöffnete Dumpdatei (*CURRENT) wird verwendet. Voreinstellung.

FILES-TO-EVALUATE = *#0 / *#1 / ... / *#9
Die Datei, die dem ausgewählten Linknamen mit dem Kommando ADD-FILE-LINK zugeordnet wurde, wird verwendet.

FILES-TO-EVALUATE = <filename 1..80 without-gen-vers-with-wild(80)>(...)
Durch Angabe eines vollqualifizierten Dateinamens mit Wildcards können mehrere Dateien ausgewählt werden.

> **EVALUATE = *FIRST-MATCH / *ALL-MATCHES**
> Passen mehrere Dateien zu dem mit Wildcards angegebenen Dateinamen, legt der Operand EVALUATE fest, ob nur die erste Datei (*FIRST-MATCH) oder alle Dateien (*ALL-MATCHES) ausgewertet werden sollen. Voreinstellung: Nur die erste Datei wird ausgewertet.

WINDOW = *NEXT-FREE / <integer 4..99>
Operand nur im Dialog erlaubt
Bezeichnet das Fenster, in dem die Auswahlmaske für die Listenerzeugung angezeigt werden soll (LIST-Fenster). Standardmäßig (*NEXT-FREE) wird das nächste freie Fenster als LIST-Fenster verwendet. Unterstützt werden die Fensternummern 4 - 9 und 21 - 99, die Fensternummern 10 - 20 sind 'reserviert'.
Die Fenster 4 - 9 werden bei Angabe von *NEXT-FREE in der Reihenfolge 9, 8, ..., 4 verwendet.

Beispiele

```
START-LIST-GENERATION FILES=$SYSDUMP.*DCLOSE*(EVAL=*ALL)
S-L-G FILES=*CURRENT, WINDOW=98
```

5.6.1.24 START-MODULE Externes Unterprogramm starten

Mit der Anweisung START-MODULE wird ein externes Unterprogramm über VMOS-Linkage aufgerufen. Falls das Unterprogramm noch nicht geladen ist, versucht DAMP es aus der Bibliothek SYSLNK.DAMP.<ver> zu laden. Unterprogramme aus anderen Bibliotheken müssen DAMP mit der Anweisung LOAD-MODULE vorher bekannt gegeben werden. Tritt im Unterprogramm ein Fehler auf, führt dies zwar zum Abbruch des Unterprogramms, jedoch nicht von DAMP.

Befindet sich das Unterprogramm in einer Endlosschleife, kann es mit K2 und anschließender Eingabe des BS2000-Kommandos `INFORM-PROGRAM MSG='*CANCEL'` abgebrochen werden. DAMP wird dann normal fortgesetzt. Siehe auch die Anweisung LOAD-MODULE auf "LOAD-MODULE Externes Unterprogramm laden".

Format

START-MODULE
NAME = <name 1..8> ,PARAMETERS = <cmd-rest 0..4096>

Operandenbeschreibung

NAME = <name 1..8>
Bezeichnet den Namen des externen Unterprogramms, das aufgerufen werden soll.

PARAMETERS = <command-rest 0..4096>
Definiert eine Zeichenkette, die unverändert an das Unterprogramm übergeben wird, d.h. das aufgerufene Programm bekommt in Register R1 die Adresse der Zeichenkette mitgeliefert.

Beispiel

```
START-MODULE NAME=DCM, PARAMETERS=WAS SIE WOLLEN
```

5.6.1.25 START-OPTION-DIALOG Einstellen der Benutzeroptionen

Die Anweisung START-OPTION-DIALOG bringt eine Auswahlmaske zur Anzeige (OPTIONS-Fenster), in der die globalen Default-Werte für DAMP eingestellt werden können.
Die Anweisung START-OPTION-DIALOG ist nur im Dialog erlaubt.

Weitere Informationen siehe "Modifikationen durch den Benutzer (Spezialfenster OPTIONS)".

Format

START-OPTION-DIALOG
WINDOW = *NEXT-FREE / <integer 4..99>

Operandenbeschreibung

WINDOW = *NEXT-FREE / <integer 4..99>
Bezeichnet das Fenster, in dem die Auswahlmaske (OPTIONS-Fenster) zum Einstellen der Default-Werte angezeigt werden soll.
Bei Angabe des Default-Werts *NEXT-FREE wird das nächste freie Fenster als OPTIONS-Fenster verwendet.
Unterstützt werden die Fensternummern 4 - 9 und 21 - 99, die Fensternummern 10 - 20 sind 'reserviert'.
Die Fenster 4 - 9 werden bei Angabe von *NEXT-FREE in der Reihenfolge 9, 8, ..., 4 verwendet.

Beispiel

```
START-OPTION-DIALOG WINDOW=4
```

5.6.1.26 START-PATTERN-SEARCH Einleiten der Stringsuche

Die Anweisung START-PATTERN-SEARCH bewirkt, dass in einem gewünschten Fenster die Auswahlmaske für die selektive Stringsuche angezeigt wird.

Weitere Informationen siehe "Zeichenketten suchen (Spezialfenster FIND)".

Format

START-PATTERN-SEARCH
WINDOW = *NEXT-FREE / <integer 4..99>

Operandenbeschreibung

WINDOW = *NEXT-FREE / <integer 4..99>
Bezeichnet das Fenster, in dem die Auswahlmaske (FIND-Fenster) angezeigt werden soll. Bei Angabe des Default-Werts *NEXT-FREE wird das nächste freie Fenster verwendet. Unterstützt werden die Fensternummern 4 - 9 und 21 - 99, die Fensternummern 10 - 20 sind 'reserviert'.
Die Fenster 4 - 9 werden bei Angabe von *NEXT-FREE in der Reihenfolge 9, 8, ..., 4 verwendet.

Beispiel

```
START-PATTERN-SEARCH WINDOW=5
```

5.6.1.27 START-PRODAMP-EDITOR Editor für den PRODAMP-Compiler laden

Mit der Anweisung START-PRODAMP-EDITOR wird ein DAMP-Spezialfenster (PRODAMP-Fenster) zugewiesen, in dem PRODAMP-Prozeduren ediert, compiliert und ausgeführt werden können.
Es besteht die Möglichkeit, mit Aufruf des Fensters zugleich eine PRODAMP-Source aus einer Datei in dieses Fenster einzulesen (siehe Operand SOURCE).
Zur Arbeit mit dem PRODAMP-COMPILER, siehe Abschnitt „Mit Prozeduren arbeiten (Spezialfenster PROC)".

Format

START-PRODAMP-EDITOR
WINDOW = *NEXT-FREE / <integer 4..99> ,SOURCE = *NONE / <filename 1..54>

Operandenbeschreibung

WINDOW = *NEXT-FREE / <integer 4..99>
Bezeichnet das Fenster, in dem das PRODAMP-Fenster angezeigt werden soll. Bei Angabe des Default-Werts *NEXT-FREE wird das nächste freie Fenster als PRODAMP-Fenster verwendet. Unterstützt werden die Fensternummern 4 - 9 und 21 - 99, die Fensternummern 10 - 20 sind 'reserviert'.
Die Fenster 4 - 9 werden bei Angabe von *NEXT-FREE in der Reihenfolge 9, 8, ..., 4 verwendet.

SOURCE = *NONE / <filename 1..54>
Bezeichnet den Namen einer Datei, deren Inhalt als Source in das PRODAMP-Fenster eingelesen werden soll.
Voreinstellung: Keine Datei wird zugewiesen.

Beispiel

```
START-PRODAMP-EDITOR WINDOW=6, SOURCE=MY.PRODAMP.SOURCE
```

5.6.1.28 START-PRODAMP-PROGRAM Laden und Starten eines PRODAMP-Programms

Mit der Anweisung START-PRODAMP-PROGRAM wird ein PRODAMP-Programm aus einer PRODAMP-Bibliothek geladen und gestartet.

Format

```
START-PRODAMP-PROGRAM

 NAME = <name 1..32 with-under> / <structured-name 1..32> / *LIBRARY-ELEMENT(...)

     *LIBRARY-ELEMENT(...)

         |    LIBRARY = *PRODAMP-USER-OBJECT-LIBRARY / *PRODAMP-SYSTEM-LIBRARY /

         |                  <filename 1..54 without-gen-vers>

         |    ELEMENT = <name 1..32 with-under> / <structured-name 1..32>

,PARAMETERS = *NONE / list-poss(32): / <integer -2147483648..2147483647> / <x-string 1..8> /

                       <c-string 1..80>
```

Operandenbeschreibung

NAME = <name 1..32 with-under> / <structured-name 1..32> / *LIBRARY-ELEMENT(...)
Bezeichnet den Namen des PRODAMP-Programms.

NAME = <name 1..32 with-under> / <structured-name 1..32>
Es wird ein PRODAMP-Programm aus der aktuell eingestellten PRODAMP-Benutzer-Objekt-Bibliothek geladen und gestartet. Der Name des Programms ist identisch mit dem Namen des Bibliothekselements.

NAME = *LIBRARY-ELEMENT(...)
Es wird ein PRODAMP-Programm aus der ausgewählten PRODAMP-Bibliothek geladen und gestartet. Der Name des Programms ist identisch mit dem Namen des Bibliothekselements.

LIBRARY = *PRODAMP-USER-OBJECT-LIBRARY / *PRODAMP-SYSTEM-LIBRARY / <filename 1..54 without-gen-vers>
Bezeichnet die PRODAMP-Bibliothek, aus der das Programm geladen werden soll.

LIBRARY = *PRODAMP-USER-OBJECT-LIBRARY
Das PRODAMP-Programm wird aus der aktuell eingestellten PRODAMP-Benutzer-Objekt-Bibliothek geladen.

LIBRARY = *PRODAMP-SYSTEM-LIBRARY
Das PRODAMP-Programm wird aus der PRODAMP-System-Bibliothek geladen. Dies ist bei Standardinstallationen die Bibliothek $TSOS.SYSDMP.DAMP.

LIBRARY = <filename 1..54 without-gen-vers>
Das PRODAMP-Programm wird aus der angegebenen PRODAMP-Bibliothek geladen.

ELEMENT = <name 1..32 with-under> / <structured-name 1..32>
Bezeichnet ein Element aus einer PRODAMP-Bibliothek. Der Name des Elements ist identisch mit dem Namen des PRODAMP-Programms, das geladen und gestartet werden soll.

PARAMETERS = *NONE / list-poss(32): <integer -2147483648..2147483647> / <x-string 1..8> / <c-string 1..80>
Mit diesem Operanden kann eine Liste von bis zu 32 Parametern an das PRODAMP-Programm übergeben werden. Die Parameter können numerisch, sedezimal oder als Charakterstring vereinbart werden. Zur Übernahme der Werte durch das PRODAMP-Programm siehe "Pseudostrukturen".

Beispiel

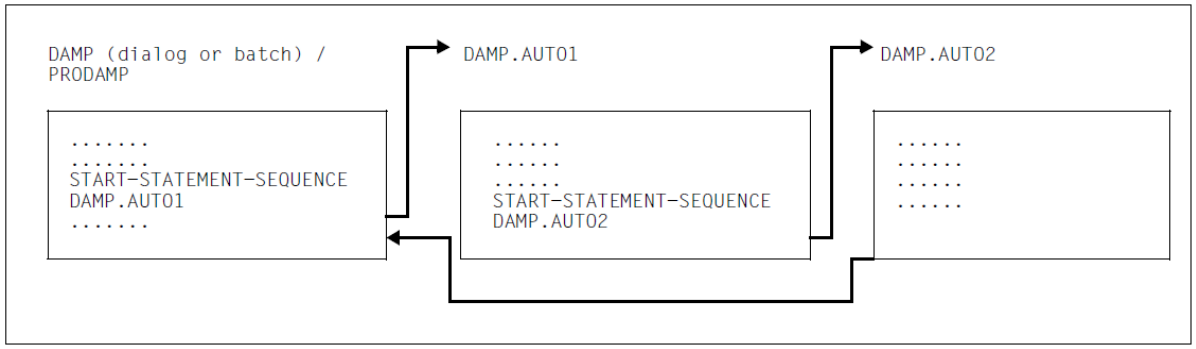
```
START-PRODAMP-PROGRAM NAME=TEST, PARAMETER = (1,2,X'ED','HUGO')
```

Hinweise

- Im Gegensatz zur Anweisung ASSIGN-PRODAMP-LIBRARIES wird beim Operanden *LIBRARY-ELEMENT die PRODAMP-Benutzer-Objekt-Bibliothek nur für den Zeitraum des Ablaufs des PRODAMP-Programms umgeschaltet, bei Beendigung des Programms wird der vorherige Zustand wieder restauriert.
- Nach Beendigung des mit START-PRODAMP-PROGRAM gestarteten PRODAMP-Programms werden alle für diesen Lauf geladenen PRODAMP-Objekte wieder entladen. Auf Grund von Unterprogrammaufrufen können weitere Objekte nachgeladen worden sein. Eine Unterbrechung mit der PRODAMP-Anweisung INTERRUPT führt nicht zu einer Entladung.
- Wird jedoch in einem PROC-Fenster (siehe "Mit Prozeduren arbeiten (Spezialfenster PROC)") ein PRODAMP-Programm ausgeführt oder fehlerfrei übersetzt, wird dieses Objekt zusammen mit den nachgeladenen Objekten erst dann entladen, wenn das zugehörige PROC-Fenster geschlossen oder in das Modusfeld „New“ eingetragen wird.
- Stellt DAMP beim Laden eines PRODAMP-Objekts fest, dass ein gleichnamiges Objekt bereits geladen ist, wird das Objekt nicht neu geladen. Dies gilt auch für das implizite Nachladen bei Aufruf eines Unterprogramms.

5.6.1.29 START-STATEMENT-SEQUENCE DAMP-Anweisungen aus Datei lesen

Mit der Anweisung START-STATEMENT-SEQUENCE werden DAMP-Anweisungen, die in einer Datei abgelegt sind, aus dieser gelesen und anschließend ausgeführt. Alle DAMP-Ausgaben werden solange zurückgestellt, bis die letzte Anweisung gelesen wurde.Enthält die Datei selbst eine START-STATEMENT-SEQUENCE-Anweisung, die eine Anweisungsfolge in einer weiteren Datei aufruft, werden die Anweisungen aus dieser Datei gelesen. Die restlichen Anweisungen aus der ursprünglichen Datei werden dann jedoch nicht mehr ausgeführt. Deshalb ist eine START-STATEMENT-SEQUENCE-Anweisung in einer solchen Datei nur als letzte Anweisung sinnvoll.



Format

START-STATEMENT-SEQUENCE
FILENAME = <filename 1..54>

Operandenbeschreibung

FILENAME = <filename 1..54>
Bezeichnet die Datei, aus der DAMP-Anweisungen gelesen werden sollen.

5.6.1.30 STOP-LOGGING Diagnosemitschnitt beenden

Mit der Anweisung STOP-LOGGING wird der mit der Anweisung LOG-SESSION begonnene Diagnosemitschnitt beendet.

Diese Anweisung besitzt keine Operanden.

5.6.1.31 USE-REGISTER Registerverwendung für Disassemblierung festlegen

Mit der Anweisung USE-REGISTER wird für die disassemblierte Ausgabe festgelegt, dass in dem angegebenen Modul ein bestimmtes Register als Basis für einen Bereich des Moduls bzw. für einen bestimmten Kontrollblock (DSECT) angenommen werden soll. Dies hat zur Folge, dass in der Befehlsaufbereitung die Operanden nicht mehr mit Basis und Distanz angegeben werden, sondern als relative Distanz innerhalb des Moduls bzw. über den Feldnamen aus dem zugehörigen Kontrollblock.
Die disassemblierte Ausgabe für x86-Code wird von der Anweisung USE-REGISTER und der korrespondierenden Anweisung DROP-REGISTER nicht unterstützt. Es können keine x86-Register angegeben werden.

Format

```
USE-REGISTER

 MODULE-NAME = <text 1..32>

 ,REGISTER = <integer 0..15>

 ,FOR = *MODULE-BASE(...) / *CONTROL-BLOCK(...)

     *MODULE-BASE(...)

         |  DISPLACEMENT = 0 /<integer -2147483648..2147483647> / <x-string 1..8>

     *CONTROL-BLOCK(...)

         |  NAME = <structured-name 1..32> / <name 1..32 with-under>
```

Operandenbeschreibung

MODULE-NAME = <name 1..32>
Bezeichnet den Namen des Moduls, innerhalb dessen die Vereinbarung für die disassemblierte Ausgabe gelten soll.

REGISTER = <integer 0..15>
Bezeichnet das /390-Mehrzweckregister, das als Basisregister verwendet werden soll.

FOR = *MODULE-BASE(...)
Das Register soll Basisregister für den angegebenen Modul sein.

> **DISPLACEMENT = 0 / <integer -2147483648..2147483647> / <x-string 1..8>**
> Bezeichnet innerhalb des angegebenen Moduls die relative Anfangsadresse des Bereiches, für den das Register als Basis verwendet werden soll.

FOR = *CONTROL-BLOCK(...)
Das Register soll Basisregister für einen Kontrollblock sein.

> **NAME = <structured-name 1..32> / <name 1..32 with-under>**
> Gibt den Namen des Kontrollblocks an, für den das Register als Basis dienen soll.

Beispiele

```
USE-REGISTER MODULE-NAME=DOPEN, REGISTER=10, FOR=*MODULE-BASE(DISPLACEMENT=X'1000')
USE-REGISTER MODULE=DCLOSE, REG=4, FOR=*CONTROL-BLOCK(NAME=EXVT)
```

5.6.2 Auf Systemebene

DAMP-Anweisungen über das Systemkommando INFORM-PROGRAM

Wird das Programm DAMP mit der Taste K2 unterbrochen, lassen sich DAMP-Funktionen auch über das BS2000-Kommando INFORM-PROGRAM ansprechen.
Die Kommunikation mit DAMP erfolgt über den Operanden MSG des Kommandos.

Format: INFORM-PROGRAM

```
INFORM-PROGRAM

 MSG = *NO / <c-string 1..64>

 ,JOB-IDENTIFICATION = *OWN / *TSN(...) / *MONJV(...)

      *TSN(...)

           |    TSN = <alphanum-name 1..4>

      *MONJV(...)

           |    MONJV = <filename 1..54 without-gen>
```

Die Eingabe im Systemmodus erfolgt in der Form `/INFORM-PROGRAM MSG='<function>'`

Format: MSG = '<function>'

```
INFORM-PROGRAM MSG='<function>'

 FUNCTION = *RESUME / HALT / *END / *TERMINATE / *DUMP / *TERMD / *ESCAPE / *BREAK /
                    *CANCEL / *LOG-SESSION / *STOP-LOGGING / *REPEAT-SESSION(...) /
                    *MODIFY-PAGE-ACCESS(...)

     *REPEAT-SESSION(...)

          |    LOGGING-FILE = <filename 1..54>

     *MODIFY-PAGE-ACCESS(...)

          |    AREA = <x-string 1..8> / <name 1..8>(...)

          |         <name>(...)

          |                   |    DISPLACEMENT = 0 / <integer 0..9> / <x-string 1..8>

          |    ,STATE = *READ-ONLY / *WRITEABLE
```

Operandenbeschreibung

FUNCTION = *RESUME / *HALT / *END / *TERMINATE / *DUMP / *TERMD / *ESCAPE / *BREAK / *CANCEL / *LOG-SESSION / *STOP-LOGGING / *REPEAT-SESSION(...) / *MODIFY-PAGE-ACCESS(...)
Bezeichnet die Funktionen, die mit dem Kommando INFORM-PROGRAM angesprochen werden können.

FUNCTION = *RESUME
Setzt DAMP an der mit K2 unterbrochenen Stelle fort.

FUNCTION = *HALT / *END / *TERMINATE
Beendet DAMP ohne Speicherauszug. Wird DAMP irregulär beendet und war zum Unterbrechungszeitpunkt eine START-MODULE-Anweisung aktiv, so bricht das Kommando `/INFORM-PROGRAM MSG='*HALT'` lediglich das externe Unterprogramm ab. Das Programm DAMP kann fortgesetzt werden.

FUNCTION = *DUMP / *TERMD
Beendet DAMP mit einem Speicherauszug.

FUNCTION = *ESCAPE / *BREAK
Zeigt die mit K2 unterbrochene Stelle an.

FUNCTION = *CANCEL
Bricht gegebenenfalls die aktuelle DAMP-Funktion ab.

FUNCTION = *LOG-SESSION
Schaltet den Diagnosemitschnitt ein.

FUNCTION = *REPEAT-SESSION(...)
Der Diagnosemitschnitt wird wieder abgespielt.

LOGGING-FILE = <filename 1..54>
Bezeichnet den Namen der Datei, in welche die Ein- und Ausgaben der DAMP-Sitzung protokolliert worden sind.

FUNCTION = *STOP-LOGGING
Schaltet den Diagnosemitschnitt aus.

FUNCTION = *MODIFY-PAGE-ACCESS(...)
Ändert den Zustand von Speicherseiten in dem Adressraum, in dem DAMP abläuft.

AREA = <x-string 1..8> / <name 1..8>(...)
Bezeichnet einen Bereich im virtuellen Adressraum, dessen Attribute geändert werden sollen.

AREA = <x-string 1..8>
Ein sedezimaler String wird als Nummer einer 4K-Seite interpretiert.

AREA = <name 1..8>(...)
Eine alphanumerische Eingabe wird als ein Modulname aus dem DAMP-System interpretiert.

DISPLACEMENT = 0 / <integer 0..9> / <x-string 1..8>
Bezeichnet eine Adresse innerhalb des Moduls, die relativ zum Modulanfang ist. Es können nur die Attribute der Seite geändert werden, zwischen deren Anfang und Ende die angegebene Adresse liegt.

STATE = *READ-ONLY / *WRITEABLE
Legt fest, ob der angegebene Speicherbereich überschreibbar gesetzt werden soll.

ℹ Mit `/INFORM-PROGRAM MSG='?'` gelangen Sie in den geführten SDF-Dialog von DAMP; „FUNCTION“ wird also wie jede andere DAMP-Anweisung behandelt.

Beispiele

```
/INFORM-PROGRAM MSG='FUNCTION=*CANCEL'
```

Eingabe von Systemkommandos in die Kommandozeile

Es können grundsätzlich alle BS2000-Systemkommandos in die Kommandozeile eingegeben werden. Der Benutzer ist selbst dafür verantwortlich, auf Kommandos zu verzichten, die eine Beendigung des Programms zur Folge haben (z.B. EXIT-JOB).

Werden abgekürzte Kommandos eingegeben, die einer abgekürzten DAMP-Anweisung entsprechen, so wird immer die DAMP-Anweisung ausgeführt. Durch Voranstellen eines Labels, z.B. /.LABEL, kann erreicht werden, dass die Eingabe als Systemkommando interpretiert wird.

Ausgaben von Systemkommandos werden in die DAMP-Meldungszeilen ausgegeben.

5.7 PRODAMP

5.7.1 Einführung

PRODAMP (PROzedursprache für DAMP) ist eine an Pascal orientierte Sprache zur Formulierung von Diagnose-Algorithmen in DAMP. PRODAMP läuft unter DAMP und nutzt die dort gebotenen Funktionen, wie etwa symbolisches Ansprechen von Datenstrukturen oder die Ausgabe auf Bildschirmfenster in verschiedenen Formaten.

Mit PRODAMP können Sie an Entscheidungen gebundene Anweisungen, die sonst einzeln von Hand eingegeben werden müssten, in eine Prozedur schreiben und automatisch ablaufen lassen. Dazu gehört zum Beispiel das Verfolgen von Verkettungen bis zu einer Struktur, die ein gesuchtes Datum enthält, das Durchsuchen von Tabellen und die (z.B. arithmetische) Verarbeitung der darin enthaltenen Werte, die automatische Beantwortung von Fragen etwa der Art: „Hält diese Task einen Lock“ u.s.w.

Beispiel

Angenommen, Sie haben häufiger Probleme zu analysieren, bei denen das Programm auf Grund eines DMS-Fehlers abstürzt und Sie als Unterlage lediglich einen Userdump erhalten. Um die betroffene Datei und den vom DMS gelieferten Fehlercode „von Hand“ zu ermitteln, sind folgende Schritte erforderlich:

1. Zuweisen des Dumpfiles
2. Auswählen des PCB, der den DMS-Makro eingegeben hat
3. Markieren von Register 1 in diesem PCB
4. Zuweisen des von Register 1 adressierten Bereichs (FCB) auf ein anderes Fenster
5. Überlagern des Bereichs mit der DSECT des FCB
6. Positionieren auf das Feld ID1FILE (Dateiname)
7. Positionieren auf das Feld ID1ECB (Fehlercode)

Die Schritte 3 bis 7 lassen sich in eine PRODAMP-Prozedur fassen und anschließend immer wieder automatisch abspulen.

Wurde die Prozedur beispielsweise unter dem Namen DMSERR abgelegt, reduzieren sich die Aktivitäten zum Ermitteln des DMS-Fehlercodes auf folgende Schritte:

1. Zuweisen des Dumpfiles
2. Auswählen des PCB, der den DMS-Makro eingegeben hat (zur Versorgung von CURRENT.PCB)
3. Eingeben der DAMP-Anweisung `//START-PRODAMP-PROGRAM DMSERR`

Die PRODAMP-Prozedur DMSERR könnte etwa folgendermaßen aussehen:

```
FNAM := ' '*54;
ERR := 0;
P := CURRENT.PCB;
FCB@ := P.ESTKGR1;
FNAM := FCB@.ID1FILE;
ERR := FCB@.ID1ECB ;
MESSAGE ( 'DMS-FEHLER '+HEX_STRING(ERR)+' FUER DATEI '+FNAM );
```

Diese Prozedur leistet sogar noch etwas mehr als die oben aufgeführten Einzelschritte, da die gewünschte Information gleich aufbereitet ausgegeben wird und nicht im Ausgabe-Fenster herausgefunden werden muss.

Eine in der PRODAMP-Sprache geschriebene Prozedur (PRODAMP-Source) ist nicht direkt ausführbar, sondern muss zunächst in ein PRODAMP-Programm (PRODAMP-Objekt) übersetzt werden. Die Übersetzung kann in einem PRODAMP-Fenster durchgeführt werden, das im Übrigen eine vollständige Entwicklungsumgebung für PRODAMP-Prozeduren bietet. Mit der DAMP-Anweisung START-PRODAMP-EDITOR wird dieses Fenster aufgerufen. Neben dem Editieren, Übersetzen und „ad hoc“ Ausführen von Prozeduren, können hier Sources und erzeugte Objekte in eine PRODAMP-Bibliothek abgelegt werden. Die DAMP-Anweisung START-PRODAMP-PROGRAM ermöglicht schließlich die Ausführung von PRODAMP-Programmen aus einer PRODAMP-Bibliothek.

PRODAMP kann also Routinearbeiten übernehmen, die anfallen, bevor man zum eigentlichen Kern des Problems vorgedrungen ist.

5.7.2 Syntax

Die Syntax von PRODAMP orientiert sich an bekannten blockorientierten Sprachen wie Pascal, Modula-2 oder ADA. Gegenüber diesen Sprachen ist PRODAMP jedoch einfacher gehalten. So gibt es keinen Deklarationsteil für Typen, Variable und Konstanten. Dadurch können auch „ad-hoc“ Algorithmen zur Lösung eines speziellen Problems formuliert werden, die nicht so vollständig ausgearbeitet sind, wie es sonst bei Programmen notwendig ist.

Die Syntax spiegelt wider, dass es neben den bekannten Sprachelementen, die man in den genannten Programmiersprachen findet, in PRODAMP Sprachelemente gibt, die auf die speziellen Diagnose-Erfordernisse zugeschnitten sind. So lassen sich zum Beispiel Diagnose-Daten symbolisch ansprechen, wobei die Feldnamen aus den zugehörigen Symboldateien entnommen werden. Es lassen sich die Meta-Eigenschaften derartiger Symbole verändern oder die Formate der Ausgabe-Fenster festlegen.

Eine vollständige Definition der Syntax ist im Abschnitt „Syntax-Diagramme“ zu finden.

5.7.3 Sprachelemente

- Lexikalische Elemente
- Operatoren
- Datentypen
- Symbole
- Variable
- Ausdrücke
- Anweisungen
- Pseudostrukturen
- Vordefinierte Variablen
- Standardprozeduren
- Standardfunktionen

5.7.3.1 Lexikalische Elemente

Zeichenvorrat

Der Zeichenvorrat von PRODAMP besteht aus

- Buchstaben,
- Sonderzeichen,
- Ziffern und
- Trennzeichen.

Bei den Buchstaben unterscheidet PRODAMP nur bei String-Literalen und in Kommentaren zwischen Groß- und Kleinschreibung.

Trennzeichen

Dort, wo Namen, Zahlen etc. nicht durch Sonderzeichen getrennt sind, müssen zur Unterscheidung Trennzeichen eingefügt werden. Trennzeichen sind das Leerzeichen und Kommentar.

Ein Kommentar kann beliebige Zeichen enthalten und wird beidseitig durch Anführungszeichen (") begrenzt. Er hat keinen Einfluss auf den Programmablauf, sondern dient nur dem besseren Verständnis beim Lesen.

Namen

Namen (Bezeichner, Identifikatoren) sind zur Identifizierung der verschiedenen Größen, die in einer Prozedur benutzt werden können (Variable, Unterprozeduren etc.), erforderlich.

Sie setzen sich zusammen aus Buchstaben, Ziffern, den Zeichen $, # oder @ und dem nur einzeln zulässigen Unterstrich "_“.

Das erste Zeichen muss ein Buchstabe oder eines der Zeichen $, # oder @ sein, das letzte Zeichen darf kein Unterstrich sein. Außerdem dürfen Namen kein Wortsymbol (Operator, Name einer Anweisung etc.) sein. Die maximale Länge von Namen beträgt 31 Zeichen.

Beispiele

```
HUGO,
X123,
A_EINS,
@LABEL,
T#1234,
DIES_IST_EIN_SEHR_LANGER_NAME
```

Länge von Source-Zeilen

Ein Zeilenende hat keine Bedeutung innerhalb der PRODAMP-Sprache. Trotzdem sollte beachtet werden:

> ¡ Im Editor sollten nur Programmzeilen mit höchstens 72 Zeichen erstellt werden. Enthält eine Zeile weniger als 72 Zeichen, so darf sie nicht mitten in einem Bezeichner, einem Literal etc. enden.

Wenn die PRODAMP-Source für die Übersetzung in ein PRODAMP-Fenster geladen wird, werden Zeilen mit mehr als 72 Zeichen umgebrochen. Anschließend werden Zeilen mit weniger als 72 Zeichen mit Leerzeichen auf 72 Zeichen aufgefüllt und die Kleinschreibung außerhalb von Strings und Kommentaren durch Großschreibung ersetzt.

5.7.3.2 Operatoren

In PRODAMP sind die Operatoren

=	(gleich),
<>	(ungleich),
<	(kleiner),
<=	(kleiner gleich),
>	(größer),
>=	(größer gleich),
+	(plus),
-	(minus)
*****	(mal)
/	(geteilt durch) und
MOD	(Modulo-Operation)

erlaubt.

Mehrere Bedingungen können durch die logischen Operatoren

AND,
OR und
NOT

verknüpft werden. Da die Operatoren unterschiedliche Priorität besitzen (siehe "Operatoren"), sind gegebenenfalls Klammern zu setzen. Die Operatoren AND und OR arbeiten kurzschließend, d.h. die Auswertung der Bedingung wird abgebrochen, sobald der Wahrheitswert feststeht.

Für Bitmuster-Ausdrücke gibt es den Operator

IN,

mit dem sich einzelne Bits und Bit-Kombinationen testen lassen. Er liefert genau dann den Wert TRUE, wenn alle getesteten Bits in dem Bitmuster gesetzt sind.

Priorität der Operatoren

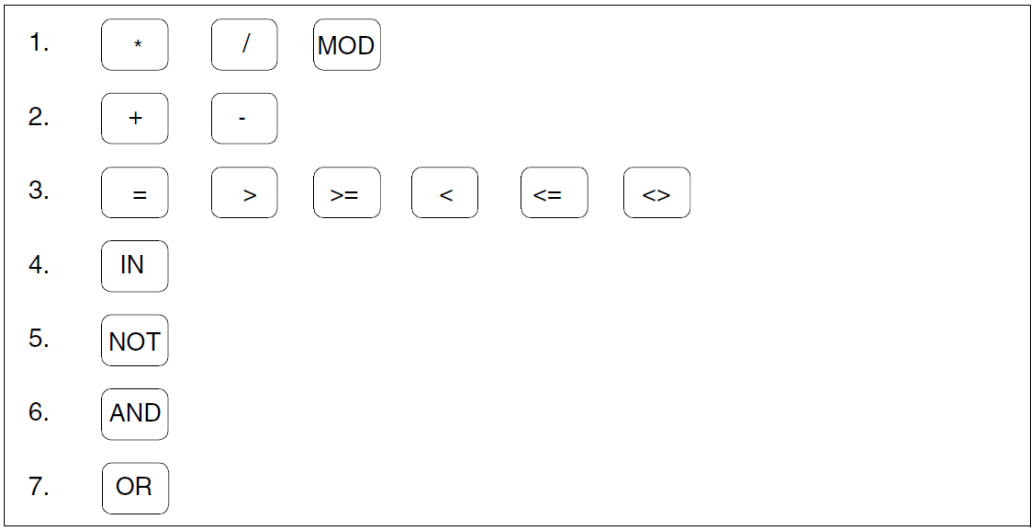


Bild 65: Priorität der Operatoren

5.7.3.3 Datentypen

PRODAMP kennt folgende drei Datentypen:

- den **numerischen Datentyp** der Länge 1 bis 4,
- den **String** der Länge 1 bis 133 und
- das **Bitmuster** der Länge 1 bis 4 Byte.

Der **numerische Datentyp** der Länge 4 ist der einzige numerische Datentyp für Variablen und Konstanten und heißt kurz numerischer Datentyp. Er umfasst die ganzen Zahlen von -2147483648 bis +2147483647. Die Zahl -2147483648 wird als Literal allerdings nur in der sedezimalen Form (X'80000000') akzeptiert.

Er wird auch zur Adressierung von Diagnose-Daten benutzt, wobei zur Adressbildung bei /390-Objekten 31 Bit, bei x86-Objekten alle 32 Bit benutzt werden.

Der Adressierungsmodus ist für PRODAMP eine HSI-abhängige Konstante.

Die numerischen Datentypen der Länge 1 bis 3 treten nur bei Symbolen auf und sind die numerische Interpretation von Feldern der Länge 1 bis 3 Byte. Bei numerischen Datentypen der Länge 2 wird der Inhalt als vorzeichenbehaftete Zahl (arithmetische Halbwortarithmetik) aufgefasst.

Numerische Literale können dezimal oder sedezimal angegeben werden.

Beispiele

```
123,
-1234,
X'0AFFE'
```

Der **String** dient zur Darstellung von Zeichenketten, die durch Hochkommata begrenzt werden; innerhalb eines Strings gelten zwei aufeinander folgende Hochkommata als Ersatzdarstellung für ein Hochkomma. Die maximale Länge eines Strings beträgt bei PRODAMP 133 Zeichen. Enthält ein String nicht abdruckbare Zeichen, kann er als sedezimale Zeichenkette vereinbart werden; hierbei bildet - wie in Pascal üblich - die Zeichenfolge #' den linken und das Hochkomma ' den rechten Begrenzer.

Beispiele

```
'1',
'Langes String-Literal',
'Wert: X''01''',
#'C100C600C600C500';
```

Bitmuster sind 1 bis 4 Byte lange Daten, in denen jedes einzelne Bit ansprechbar ist. Bitmuster-Literale werden entweder binär (bis zu einem Byte Länge) oder sedezimal definiert. Die Schreibweise bei binärer Darstellung entspricht der Assembler-Schreibweise. Bei sedezimaler Darstellung wird, um Verwechslungen mit sedezimalen numerischen Literalen zu vermeiden, die Zeichenfolge P' als linker und das Hochkomma ' als rechter Begrenzer verwendet.

Außerdem gibt es die symbolischen Bitmuster-Konstanten TRUE und FALSE.

Beispiele

```
B'10101001'  entspricht P'A9'
TRUE         entspricht P'01' bzw. B'1'
FALSE        entspricht P'00' bzw. B'0'
```

Verträglichkeit von Datentypen

Die folgende Tabelle enthält die erlaubten Typmischungen für Zuweisungen (siehe "Anweisungen") und Vergleichsoperationen.

rechts -> links | \V	undefiniert	numerisch	Bitmuster	String	
undefiniert	arithmet.	arithmet.	Bitmuster	String	*)
numerisch	arithmet.	arithmet.	arithmet.	-----	
Bitmuster	Bitmuster	-----	Bitmuster	-----	
String	String	-----	-----	String	

Tabelle 9: Zulässige Typmischungen für Zuweisungen und Vergleichsoperationen

*) Die Einträge in dieser Zeile gelten nur für Zuweisungen. Bei Vergleichen muss man ggf. durch Umgruppieren dafür sorgen, dass der linke Operand nicht „undefiniert“ ist.

Als Fundamentalregel gilt: der Typ eines Ausdrucks wird durch den Typ des ersten Terms bestimmt.

Bei Zuweisungen ist unter „links“ die Variable links vom Zuweisungsoperator zu verstehen, unter „rechts“ der Ausdruck rechts davon. Im Schnittpunkt ist zu sehen, ob die Zuweisung erlaubt oder verboten ist und von welchem Typ das Ergebnis ist (Letzteres ist nur interessant, wenn der Typ vorher undefiniert war).

Bei Vergleichsoperationen ist unter „links“ der Ausdruck links vom Vergleichsoperator und unter „rechts“ der Ausdruck rechts davon zu verstehen. Im Schnittpunkt steht, ob der Vergleich erlaubt und wie er zu interpretieren ist. (Bei einem arithmetischen Vergleich ist beispielsweise der Operator IN nicht zulässig.)

In Ausdrücken sind nur zwei Arten der Typmischung erlaubt:

<numerisch><operator><Bitmuster> (ergibt numerisch) und
<String><operator><numerisch> (ergibt String),

wobei im zweiten Fall nur der Operator „*“ akzeptiert wird (Stringvervielfältigung). Ausdrücke, in denen der erste Term undefiniert ist, bekommen den Typ „numerisch“.

Symbole haben grundsätzlich den Typ „undefiniert“, weil auf die Symboldatei erst zum Zeitpunkt des Prozedurablaufs zugegriffen wird. Das kann zu ungewünschten Effekten führen:

Beispiel

Interessiert man sich für das rechte Bit im Feld EXVTAUDI, will die höherwertigen Bits ausblenden und schreibt

```
MY_BITS:=.EXVTAUDI-P'FE';

IF MY_BITS=P'01' THEN
```

wird EXVTAUDI arithmetisch gesetzt, das Bitmuster-Literal arithmetisch uminterpretiert und als Ergebnis kommt für MY_BITS entweder der arithmetische Wert -253 (X'FFFFFF03') heraus (wenn das Bit gesetzt ist) oder -254 (wenn es nicht gesetzt ist). Die anschließende Abfrage wird vom Compiler ebenfalls akzeptiert, weil er MY_BITS als arithmetisch ansieht und das Bitmuster wieder arithmetisch umdeutet (siehe Tabelle 9). Das war jedoch nicht beabsichtigt.

Das gewünschte Ergebnis lässt sich hingegen folgendermaßen gewinnen:

```
MY_BITS:=P'0'+.EXVTAUDI-P'FE';

IF MY_BITS=P'01' THEN
```

Alternativ können Sie durch Vorinitialisieren von Variablen dafür Sorge tragen, dass bei Verwendung von Symbolen die richtigen Operationen generiert werden.

Beim Zugriff auf Daten des Diagnoseobjekts über Symbole, deren Typ zum Compilierungszeitpunkt noch nicht bekannt sein kann, bedarf es also einiger Überlegung, um vor Überraschungen sicher zu sein. Schematisches Umweisen der Diagnosedaten auf initialisierte Variable beispielsweise würde solche Probleme von vornherein ausschließen.

Die nachfolgende Tabelle gibt an, wie die Datentypen, die in den Symboldateien in Form von LSD-Codes (Assembler-Datentypen) hinterlegt sind, in die von PRODAMP verwendeten Datentypen umgesetzt werden.

LSD code	Assemb. Typ	Daten Länge	PRODAMP Typ	Daten Beschreibung
00	C	n	STRING	Zeichen
01	Z	n		entpackte Dezimalzahl
03	E	4		Gleitkommazahl, einfach genau
04	D	8		Gleitkommazahl, doppelt genau
05	P	n		gepackte Dezimalzahl
0F		n		Maschinenbefehl
10		n		CSECT, COM, DSECT, XDSEC
06	H	2	NUMERIC	Binärwert mit Vorzeichen
07	F	4	NUMERIC	Binärwert mit Vorzeichen
0A	Y	2	PATTERN	Binärwert ohne Vorzeichen
	A	4	NUMERIC	Binärwert ohne Vorzeichen

0A	X {	1 - 3 4 sonst	PATTERN NUMERIC STRING	} Binärwert ohne Vorzeichen

Tabelle 10: Umsetzung von Assembler-Datentypen in PRODAMP-Datentypen

5.7.3.4 Symbole

Mit Symbolen greift man auf Daten des zu Grunde liegenden Diagnoseobjekts (Dumpfile, System) oder auf Metadaten des Programmes DAMP zu. Ein Symbol besitzt einen Namen beginnend mit einem Punkt, eine Relativadresse, einen Typ und eine Länge.

Symbole können nicht beliebig benannt werden, sondern müssen

- dem Programm DAMP aus der Symboldatei bekannt sein,
- mit der Anweisung ARRANGE innerhalb der PRODAMP-Prozedur erzeugt werden oder
- intern, also vonseiten der Programmiersprache PRODAMP definiert sein.

Name, Relativadresse, Typ und Länge eines Symbols sind in den DAMP-Symboldateien bzw. den privaten, mit ADD-SYMBOLS zugewiesenen Symboldateien hinterlegt. Die Relativadresse bezieht sich immer auf den Anfang der Struktur (DSECT), in der das betreffende Symbol enthalten ist. Daraus ergibt sich, dass über ein Symbol nur dann auf eine Datenstruktur des Diagnoseobjekts zugegriffen werden kann, wenn man die Basis-Adresse der betreffenden Struktur angibt. Das wird auch in der Syntax für ein Symbol berücksichtigt.

Beispiele

```
.ETCBTFT
```

„.ETCBTFT“ ist durch den Punkt als Symbol erkennbar. Die explizite Angabe einer Basis-Adresse ist nicht erforderlich, da der TCB zu den durch DAMP automatisch lokalisierbaren Strukturen gehört (wie auch JCB, UVMT, SVMT oder EXVT).

```
A_FCB.ID1FILE
```

„A_FCB“ ist eine Variable, die die Basis-Adresse der betreffenden Struktur - hier eines TU-FCB - enthält. „ID1FILE“ ist ein Feldname aus der DSECT „ID1FCB“.

```
PTR.NKLCB_MDL.COPY_PARAMETER.USER_ALLOCATION.WAIT_FACTOR
```

Bei diesem Beispiel handelt es sich um Symbole in Substrukturen; diese werden durch Angabe der Substruktur-Schachtelung oder durch Aufruf der PRODAMP-Standardprozedur REFERENCE spezifiziert.
Die Angabe der DSECT (NKLCB_MDL) ist nur dann notwendig, wenn das erste Symbol (COPY_PARAMETER) hinsichtlich aller in der Symboldatei enthaltenen DSECTs nicht eindeutig ist.

Folgende Namen sind bei PRODAMP als Bezeichner reserviert und können nicht für Variable verwendet werden:

```
ABS-ADDRESSING  ADDRESS         ALET            AND             ARRANGE
ASEL            COMMAND         CPU             CURRENT         DEC_BINARY
DEC_STRING      DMP_#REFRESH    DO              DSECT           DUMP_MEMORY
ELSE            ELSIF           END             ENTER_MODULE    EXTRACT
FALSE           FOLLOW          HEX_BINARY      HEX_STRING      HSA
IF              IN              INFIELDS        INSERT          INTERRUPT
ITN             LAYOUT          LENGTH          LIST            LOCATION
MESSAGE         MOD             NAME            NEW_TASK        NEXT_WINDOW
NOT             NUMBER          NUMERIC         OFF             OFFSET
ON              OPC_TABLE       OR              OUTPUT          PARAMETER
PATTERN         PCB             PCK_BINARY      READ            READ_WINDOW
REAL            REFERENCE       RELATIVE        RETURN          SET_HEADER
SPID            STRING          SVC_TABLE       THEN            TID
TRACE           TRUE            TSN             TYPE            UNDEFINED
UNSIGNED_OFF    UNSIGNED_ON     VIRTUAL         WHILE           WINDOW
```

```
WRITE
```

Folgende Namen sind bei PRODAMP als Symbolbezeichner gekoppelt mit CURRENT und INFIELDS reserviert. Sie können, falls sie nicht in der oben angeführten Liste enthalten sind, zwar als Variable, jedoch nicht zur Bezeichnung von Symbolen verwendet werden.

ADDRESS	ALET	ASEL	ASID	ATYPE
COMMAND	CONFIGURATION	CPU	CSMA	DTYPE
DUMPTIME	ERROR	FILENAME	HSA	ITN
LAYOUT	LENGTH	LEVEL	MARK1	MARK2
MARK3	MARK4	MARK5	MARK6	PARAMETER
PCB	PTYPE	RELATIVE	SEGMENT	SPID
STACK	SYMBOL	TIME	TID	TSN
VERSION	WNDNO	WNDTSK		

PRODAMP-Prozeduren mit diesen Namen können nicht als Benutzer-Unterprogramm aufgerufen werden.

Alle Bezeichner, die mit DMP_ beginnen, sind für künftige PRODAMP-Erweiterungen reserviert und sollten daher nicht in Benutzerprogrammen verwendet werden.

5.7.3.5 Variable

Eine Variable ist ein Objekt, auf das während des Programmlaufs Werte übertragen werden können. Sie gehört einem der Datentypen an und kann gelesen oder beschrieben werden.

Variable können den Datentyp String fester Länge (bis zu 133 Zeichen), numerisch (4 Byte) oder Bitmuster (1 bis 4 Bytes) besitzen.
Zum Übersetzungszeitpunkt wird der Datentyp einer Variablen durch die statisch erste Zuweisung festgelegt, und zwar durch die Bestimmung des Datentyps der rechten Seite der Zuweisung. Näheres siehe Tabelle 9 (Datentypen).
Zum Ablaufzeitpunkt führt ein lesender Zugriff auf eine Variable, die noch nicht dynamisch beschrieben wurde, zu einem Laufzeitfehler und zum Abbruch des PRODAMP-Programms.

5.7.3.6 Ausdrücke

Ausdrücke sind Rechenvorschriften, die nach ihrer Auswertung einen Wert liefern. Man erhält sie durch die Verknüpfung von Operanden mit Operatoren (siehe Abschnitt „Syntax- Diagramme"). Die Bedeutung der Operatoren zeigt die folgende Tabelle:

	numerisch	**String**	**Bitmuster**
+	Addition	Verkettung	Mengenvereinigung
-	Subtraktion		Mengendifferenz
*	Multiplikation	Vervielfältigung *)	Durchschnitt
/	Division		
MOD	Modulo-Operation		

*) Der zweite Operand muss in diesem Fall numerisch sein.

Tabelle 11: Bedeutung der Operatoren

In PRODAMP sind die Operatoren also mehrdeutig (überladen), das heißt, die Operatoren werden je nach Operandentyp unterschiedlich interpretiert.

Beispiele

```
33 + 16                 Ergebnis: 49
X'10' * 4               Ergebnis: 64
'System' + 'absturz'    Ergebnis: 'Systemabsturz'
#'C1' * 5               Ergebnis: 'AAAAA'
B'1001' + B'11'         Ergebnis: B'1011' oder P'0B'
P'1A' * B'1001'         Ergebnis: B'1000' oder P'08'
```

Die Modulo-Operation liefert den ganzzahligen Rest einer Divison.

Beispiele

```
29 MOD 7 ergibt 1
35 MOD 11 ergibt 2
```

> Der Operand MOD lässt sich auch benutzen, um beispielsweise Adressen „von Hand“ auf 24 Bit umzustellen.
>
> *Beispiel*
>
> `X'887C0A0E' MOD X'01000000' Ergebnis: X'007C0A0E'`

Ausdrücke können in üblicher Weise mit Klammern gruppiert und beliebig kombiniert werden (im Rahmen der Typverträglichkeit).

Beispiele

```
X * Y + 3 * ( A - B ) + X'ABC'
A_FCB.ID2IND1 - P'80'            Alle Bits aus ID2IND1 bis auf ID2DUMMY
NUM - ( NUM / 16 ) * 16
('a'+'b'*2) * 4                  Ergebnis: 'abbabbabbabb'
```

5.7.3.7 Anweisungen

Eine PRODAMP-Prozedur besteht aus Anweisungen. Jede Anweisung wird durch Semikolon abgeschlossen. Da es bei PRODAMP keine Deklarationen gibt, generiert jede Anweisung Code, der erst beim Ausführen der Prozedur interpretiert wird.

In PRODAMP gibt es folgende Anweisungen:

Anweisungsname	Wirkung
ARRANGE	Symboleigenschaften vereinbaren
FOLLOW	Variablen überwachen
IF	Bedingte Anweisungen geben
INTERRUPT	Prozedur unterbrechen
RETURN	Prozedur verlassen
TRACE	Ablaufverfolgung steuern
WHILE	Programmschleife bilden
Prozeduraufruf	Prozedur aufrufen
Zuweisung	Wert zuweisen

Tabelle 12: Übersicht über die PRODAMP-Anweisungen

ARRANGE
Symboleigenschaften vereinbaren

Mit der Anweisung ARRANGE können Sie Namen, Länge, Relativadresse und Typ von Symbolen dynamisch vereinbaren.
Mit ARRANGE WINDOW kann veranlasst werden, dass ein festlegbares Dumpfenster als oberstes Fenster am Bildschirm erscheint; ferner kann vereinbart werden, dass Werte aus der PRODAMP-Prozedur in die Eingabefelder des Fensters eingetragen werden. Die Schlüsselworte und ihre Zuordnung entnehmen Sie bitte Bild 66.

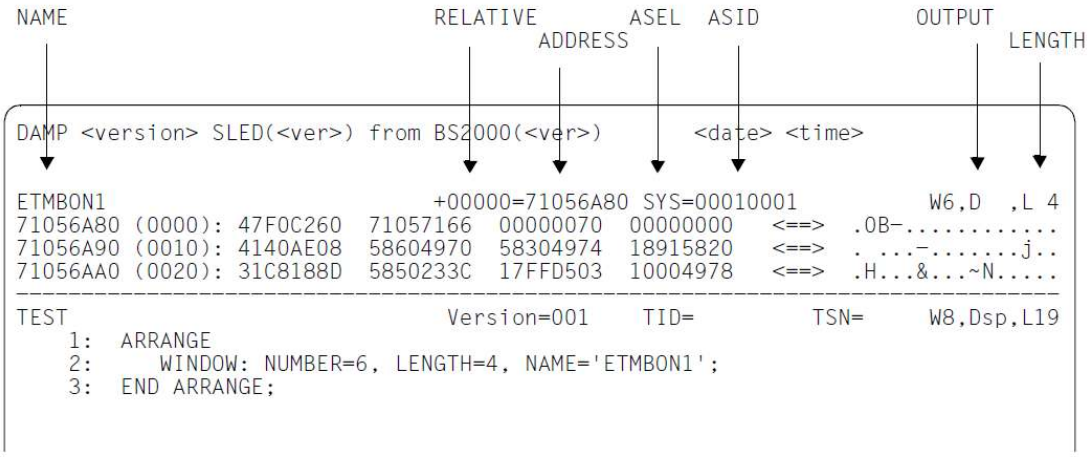


Bild 66: Kopfzeile eines Dumpfensters

ARRANGE WINDOW

Mit ARRANGE WINDOW wird das angegebene Fenster an die Spitze der Fenster-Kette in DAMP geholt. Nach mehreren ARRANGE-WINDOW-Vereinbarungen für verschiedene Fenster und anschließender Bildschirm-Ausgabe erscheint das zuletzt genannte Fenster als oberstes Fenster am Bildschirm. Die anderen Fenster folgen in umgekehrter Reihenfolge, soweit auf Grund der eingestellten Länge möglich.

Für das Help-Fenster (W1), das PROC-Fenster und andere Spezialfenster wird die Angabe ARRANGE WINDOW zurückgewiesen. Für das Status-Fenster (W2) und das Stack-Fenster (W3) lassen sich die Parameter TID, TSN und LENGTH vereinbaren, für W3 außerdem noch PCB, mit dem ein PCB aus der PCB-Kette des Tasks über seine laufende Nummer ausgewählt wird.

ARRANGE WINDOW wirkt genauso wie eine Eingabe in die Kopfzeile eines Diagnosefensters. Jeder dort möglichen Eingabe entspricht ein Schlüsselwort der Anweisung ARRANGE. Lediglich der Parameter NUMBER, mit dem die gewünschte Fenster-Nummer angegeben wird, durchbricht diese Regel.

ARRANGE

Datenstrukturen, die in keiner Symboldatei hinterlegt sind, können Sie mit der Anweisung ARRANGE ad-hoc verfügbar machen. ARRANGE ist auch immer dann erforderlich, wenn unstrukturierte Daten des Diagnoseobjekts angesprochen werden sollen (siehe „Beispiel 5 zu ARRANGE").

Die mit ARRANGE (um)definierten Symbole sind nur innerhalb der PRODAMP-Prozedur bekannt. Sie werden nicht in den Symbol-Binärbaum von DAMP eingesetzt. Demzufolge werden diese Änderungen auch bei der Überlagerung eines Fensters mit einer DSECT nicht wirksam.

Neue Symbole müssen vollständig definiert werden, d.h. LENGTH, TYPE und RELATIVE sind explizit zu versorgen. Andernfalls versucht PRODAMP, die fehlenden Angaben aus der Symboldatei zu ergänzen, und bricht die Ausführung ab, wenn das Symbol nicht gefunden wurde.

Umgekehrt können Sie schon festgelegte Eigenschaften gezielt zurücksetzen, wenn man für sie das Schlüsselwort UNDEFINED angibt. Beim nächsten Zugriff auf das Symbol wird dann die fehlende Eigenschaft aus der Symboldatei ergänzt.

Beispiel 1 zu ARRANGE WINDOW

```
ARRANGE WINDOW:
NUMBER=5, ADDRESS=0, LENGTH=8, TID=X'ABC', OUTPUT='ASS' ;
END ARRANGE ;
```

Im Fenster 5 soll der Benutzerspeicher der Task ABC ab Adresse 0 dargestellt und disassembliert werden (OUTPUT='ASS'). Ferner wird vereinbart, dass das Fenster 8 Zeilen lang sein soll.

Beispiel 2 zu ARRANGE WINDOW

```
ARRANGE WINDOW:
NUMBER=7, NAME='IDMFILE', DSECT = 'IDMTFT', OUTPUT='CBM', ADDRESS=TFT_AD;
END ARRANGE ;
```

Im Fenster 7 soll ein Speicherbereich ab der Adresse TFT_AD dargestellt und in Form der DSECT IDMTFT symbolisch aufbereitet werden. Dabei wird die DSECT so positioniert, dass die DSECT IDMTFT an die Basisadresse TFT_AD gelegt wird und das Feld IDMFILE in der 1.Zeile des Fensters zu liegen kommt. Die Fensteranfangsadresse beträgt TFT_AD + Offset(IDMFILE).

Beispiel 3 zu ARRANGE

```
ARRANGE
.ESTK#ICL : TYPE = STRING;
END ARRANGE;
IF 'P' = STK1.ESTK#ICL THEN ...
```

Das Feld ESTK#ICL aus der DSECT ESTK wird umdefiniert. Es ist als DS XL 1 definiert, wird also als Bitmuster interpretiert, enthält aber einen Buchstaben. Durch die Zuweisung des Typs STRING können Sie im Folgenden dieses Feld als Buchstaben interpretieren.

Beispiel 4 zu ARRANGE

```
ARRANGE
.STDHD : RELATIVE = 0, LENGTH = 8, TYPE = STRING ;
.PAR_1 : RELATIVE = 8, LENGTH = 4, TYPE = NUMERIC ;
.PAR_2 : RELATIVE =12, LENGTH = 1, TYPE = STRING ;
.PAR_3 : RELATIVE =13, LENGTH = 1, TYPE = PATTERN;
END ARRANGE ;
P_LATTE := .... ;
FRST_PAR := P_LATTE.PAR_1 ; ....
```

Innerhalb PRODAMP wird eine DSECT konstruiert, etwa für eine neu entwickelte Schnittstelle, deren Parameterliste noch nicht offiziell übergeben wurde. Mit den neu definierten Namen lassen sich dann die Felder der Parameterliste symbolisch ansprechen, wobei natürlich immer die entsprechende Basisadresse mitzugeben ist. Auf analoge Weise werden die Parameter zwischen PRODAMP-Prozeduren übergeben.

Beispiel 5 zu ARRANGE

```
ARRANGE
.BYTE : TYPE=NUMERIC,LENGTH=1,RELATIVE=0;
END ARRANGE;
INT_AD := STK.ESTK#ICR;          "Adresse des Paging-Errors"
OPCODE := INT_AD.BYTE;           "Maschinenbefehl an dieser Stelle"
IF OPCODE = X'D2' THEN ......
```

Mit ARRANGE lassen sich unstrukturierte Daten des Diagnosobjekts ansprechen (beispielsweise Coding).

Beispiel 6 zu ARRANGE

```
ARRANGE
  .HALBWORT: OFFSET = 0, LENGTH = 2, TYPE = NUMERIC;
END ARRANGE
WORT := PTR.HALBWORT
```

Ist ein Feld als NUMERIC mit LENGTH=2 definiert, dann ist insbesondere darauf zu achten, dass die Zuweisung des Feldinhaltes zu einer Variablen unter Berücksichtigung des Vorzeichens geschieht.
Enthält HALBWORT an der Adresse PTR etwa die Buchstabenfolge C'AB', dann entspricht dies nach Definition des Feldes nicht dem numerischen Wert X'C1C2' = 49602 sondern dem Wert X'FFFFC1C2' = -15943, der anschließend der Variablen WORT zugewiesen wird. Soll das Vorzeichen des Inhalts unberücksichtigt bleiben, dann ist das Feld mit TYPE=PATTERN zu definieren. Allgemein gilt für TYPE=NUMERIC die Regel, dass bei gerader Länge das Vorzeichen berücksichtigt wird und bei ungerader Länge der übertragene Wert immer positiv ist.

Beispiel 7 zu ARRANGE

```
ARRANGE
  .WAIT_FACTOR: LENGTH = 1, TYPE = STRING,
                REFERENCE = .NKLCB_MDL.COPY_PARAMETER.USER_ALLOCATION
END ARRANGE;
```

Die Elemente einer Substruktur können ebenfalls mit ARRANGE umdefiniert werden. Hierzu ist die Bezeichnung REFERENCE zu verwenden, um das Symbol innerhalb einer Referenzkette zu lokalisieren.

FOLLOW
Variable überwachen

Mit der Anweisung FOLLOW können Sie Variablen überwachen. Die Variable muss dabei vorher deklariert, d.h. initialisiert sein.

Nach dem Durchlaufen der Anweisung FOLLOW werden alle Zuweisungen an die genannte Variable in einen EDT-Bereich (standardmäßig Bereich 8) protokolliert. Die Ausgabe enthält die Zeilennummer, in der die Variable zugewiesen wird, den Variablennamen sowie deren Wert nach der Zuweisung.

Darüberhinaus wird jede Änderung von CURRENT.ERROR protokolliert.

Wird im Prozedurverlauf der EDT-Bereich mit WRITE („@PROC XX“) gewechselt, erfolgen die Ausgaben der Variablenüberwachung ebenfalls in den neuen EDT-Bereich.

FOLLOW ist bei der Fehlersuche in PRODAMP-Prozeduren sehr hilfreich.

> ¡ Die Überwachung einer Variablen kann nur ein-, aber nicht abgeschaltet werden. Sie wird erst am Prozedurende oder nach Verlassen der Prozedur mit RETURN beendet.

Beispiel zu FOLLOW

Die Prozedur

```
1)  S := ' '*8;
2)  STR := '12XY34';
3)  NUM := ' ';
4)  I := 0;
5)  A := 0; FOLLOW A;
6)  WHILE I <= 5 DO
7)    EXTRACT ( NUM,STR,I );
8)    INSERT ( NUM,S,3 );
9)    A := DEC_BINARY ( S );
10)   I := I + 1;
11)  END WHILE;
```

erzeugt folgendes Protokoll:

```
TEST (0) %STMT 9: 'A'              <- 1 ( X'00000001' )
TEST (0) %STMT 9: 'A'              <- 2 ( X'00000002' )
TEST (0) %STMT 9: 'A'              <- 1 ( X'00000001' )
TEST (0) %STMT 10: 'CURRENT.ERROR' <- 4
TEST (0) %STMT 9: 'A'              <- 1 ( X'00000001' )
TEST (0) %STMT 9: 'A'              <- 3 ( X'00000003' )
TEST (0) %STMT 10: 'CURRENT.ERROR' <- 0
TEST (0) %STMT 9: 'A'              <- 4 ( X'00000004' )
```

Dabei wird eine Änderung von CURRENT.ERROR, die durch eine Standard-Prozedur verursacht wird, erst nach dem Statement ausgegeben, das sie verursacht hat. Sie sehen an dem Beispiel ebenfalls, dass DEC_BINARY einen undefinierten Wert liefert, wenn der String keine Dezimalzahl ist.

IF
Bedingte Anweisungen geben

Mit der Anweisung IF können Sie bedingungsabhängige Anweisungen geben. Die zu erfüllende Bedingung wird durch Verknüpfung mehrerer Ausdrücke mit **logischen Operatoren** festgelegt (siehe "Operatoren").

Beispiel 1 zu IF

```
IF PCB.ESTKGR15 <> 0 THEN
  RC := PCB.ESTKGR15 MOD 256 ;
  MESSAGE ( 'Returncode '+HEX_STRING(RC)+' bei $REQM.' );
END IF;
```

Ist der Inhalt im Feld ESTKGR15 ungleich null, ist im Register 15 also ein Returncode hinterlegt, dann wird die anschließend angegebene Meldung auf dem Bildschirm ausgegeben.

Beispiel 2 zu IF

```
IF 'TSOS ' = .EJTPUSR THEN
  IF 'HELGA' + ' '*49 = .EJTPXPRG THEN
    DANGER := 100 ;
  END IF ;
ELSIF 'SERVICE ' = .EJTPUSR THEN
  DANGER := 180 ;
ELSE
  DANGER := 0 ;
END IF;
```

Benutzer und verwendetes Programm werden in den Feldern EJTPUSR und EJTPXPRG abgefragt. Abhängig vom Ergebnis wird der Wert der Variablen DANGER gesetzt.

Beispiel 3 zu IF

```
EVIPLFLG := B'00000100' ; "SYSTEM LOADING COMPLETED"
EVCL2REQ := P'02' ; "CLASS 2 MEMORY REQUESTED"
IF EVIPLFLG + EVCL2REQ IN .EVSVMIND THEN ........
```

Es werden zwei Felder definiert, und geprüft, ob der Inhalt dieser Felder im Feld EVSVMIND enthalten ist.

> ¡ Beispiel 3 zeigt noch etwas anderes: Es handelt sich um konkrete Equates aus der SVMT-DSECT. Da DAMP die Equates nicht in der Symboldatei hinterlegt, müssen die entsprechenden Bitmuster in PRODAMP definiert werden.

INTERRUPT
Prozedur unterbrechen

Mit der Anweisung INTERRUPT unterbrechen Sie eine Prozedur und verzweigen zum Programm DAMP. Nun können beliebige DAMP-Anweisungen gegeben werden. Die Rückkehr in die Prozedur erreichen Sie mit der Anweisung RESUME-PRODAMP-PROGRAM.

INTERRUPT bietet sich beispielsweise an, wenn man der Reihe nach eine Kette von Datenstrukturen inspizieren will. Steht die Anweisung INTERRUPT zum Beispiel in einer Schleife, die die Datenstrukturen der Reihe nach lokalisiert und einem Bildschirm-Fenster zuweist, können Sie zusammen mit der Anweisung RESUME-PRODAMP-PROGRAM praktisch von Kettenglied zu Kettenglied blättern.

Beispiel zu INTERRUPT

```
INTERRUPT WINDOW=4
```

Der Parameter WINDOW= bewirkt eine Auffrischung des gesamten DAMP-Bildschirms. Werden mit INTERRUPT nacheinander verschiedene Ausgaben auf das gleiche Fenster gelegt, so ist dieser Effekt oft störend. Man kann das Auffrischen dadurch vermeiden, dass man INTERRUPT ohne explizite Fensterangabe verwendet, es werden dann nur die geänderten Felder des Bildschirms neu geschrieben.

RETURN
Prozedur vorzeitig verlassen

Die Anweisung RETURN in einer PRODAMP-Prozedur veranlasst die Rückkehr zu ihrem Aufrufer, also entweder in eine andere Prozedur oder in den DAMP-Dialog.

RETURN wird auch vom Compiler implizit am Prozedurende generiert.

RETURN WINDOW
Prozedur vorzeitig beenden

Mit der Anweisung RETURN WINDOW=<fensternummer> wird im Gegensatz zum einfachen RETURN nicht nur die aktuelle Prozedur verlassen, sondern die gesamte Aufrufhierarchie abgebrochen. Zusätzlich spezifiziert <fensternummer> die Nummer des Diagnosefensters, das im Dialog nach Abbruch der Prozedur an oberster Stelle im DAMP-Schirm angezeigt werden soll.

Beispiel zu RETURN

```
RETURN WINDOW = CURRENT.WNDNO;
```

Die Prozedur soll vollständig abgebrochen und das Diagnosefenster eingestellt werden, das im Feld CURRENT.WNDNO angegeben ist.

TRACE
Ablaufverfolgung steuern

Mit den Anweisungen TRACE ON und TRACE OFF wird die Ablaufverfolgung ein- bzw. ausgeschaltet. Alle zwischen TRACE ON und TRACE OFF durchlaufenen Sourcezeilen-Nummern werden in einen Bereich des EDT (standardmäßig Bereich 8) protokolliert. Die beiden Anweisungen können an beliebiger Stelle in eine Prozedur eingefügt werden.

Wird im Laufe der Prozedur mit WRITE („@PROC XX“) der EDT-Prozedurbereich gewechselt, wird auch die Trace in den neuen Bereich ausgegeben.

Das Ein- und Ausschalten des Trace ist abhängig von der dynamischen Reihenfolge der Statements.

TRACE ist bei der Fehlersuche in PRODAMP-Prozeduren sehr hilfreich.

Beispiel zu TRACE

```
PROZEDUR PROCNAME:
1) IF CURRENT.LEVEL < 4 THEN
2)    TRACE ON;
3)    I := 1234;
4)    PROCNAME;
5)    I := 5678;
6) END IF;
```

führt beim Ablauf zur Protokollierung folgender Zeilennummern:

```
PROCNAME (0) %STMT 3
PROCNAME (0) %STMT 4
PROCNAME (1) %STMT 3
PROCNAME (1) %STMT 4
PROCNAME (2) %STMT 3
PROCNAME (2) %STMT 4
PROCNAME (3) %STMT 3
PROCNAME (3) %STMT 4
PROCNAME (3) %STMT 5
PROCNAME (3) %STMT 6
PROCNAME (3) %STMT 7
PROCNAME (2) %STMT 5
PROCNAME (2) %STMT 6
PROCNAME (2) %STMT 7
PROCNAME (1) %STMT 5
PROCNAME (1) %STMT 6
PROCNAME (1) %STMT 7
PROCNAME (0) %STMT 5
PROCNAME (0) %STMT 6
PROCNAME (0) %STMT 7
```

Dabei wird zunächst der Prozedurname ausgegeben und in Klammern der Wert von CURRENT.LEVEL, um die Unterprogrammebenen der Prozedur unterscheiden zu können. Anschließend wird die Zeilennummer des durchlaufenen Statements ausgegeben. PRODAMP generiert am Ende einer Prozedur implizit immer ein RETURN Statement. Deswegen taucht im Protokoll STMT 7 auf, das in der obigen Prozedur nicht enthalten ist.

Ist der Trace eingeschaltet, werden weitere TRACE ON Anweisungen ignoriert. Das erste TRACE OFF schaltet daher den Trace ab.

WHILE
Programmschleifen bilden

Mit der Anweisung WHILE ist es möglich, Schleifen zu bilden.

Beispiel zu WHILE

```
WHILE ADDR <= MAX DO
  IF ADDR.TABVAL = BAD_VAL THEN
    RETURN ;
  END IF ;
  ADDR := ADDR + TAB_LEN ;
END WHILE ;
```

Solange die Variable ADDR nicht größer wird als die Variable MAX, wird die anschließend angegebene Anweisungsfolge immer wieder abgearbeitet.

Zuweisung

Die Zuweisung ist die einfachste Art der Anweisung. Zuweisungsoperator ist das Zeichen „:=“. Links vom Zuweisungsoperator muss ein Variablenname stehen, rechts muss ein Ausdruck, d.h. ein Variablenname, ein Symbol, ein Literal, ein Funktionsaufruf oder ein arithmetischer Ausdruck stehen.

Der Datentyp einer Variablen wird durch die statisch erste Zuweisung (statisch, d.h. in der Reihenfolge der Source) an diese Variable definiert, gemäß Tabelle 9 (Datentypen). Der Datentyp der rechten Seite wird, unter Berücksichtigung bereits erfolgter Typzuordnungen, bestimmt und übernommen. Ist der Datentyp der rechten Seite undefiniert, wird er auf numerisch gesetzt. Ein Datentyp ist undefiniert, wenn z.B. der erste Operand ein Symbol ist; alle Symbole, auch die mit der PRODAMP-Anweisung ARRANGE definierten Symbole, haben zum Übersetzungszeitpunkt den Wert „undefiniert“.

Beispiele

```
A_FCB  := X'ABC';
FNAM   := A_FCB.ID1FILE;
TEXT   := 'FCB konnte nicht gefunden werden.';
XY     := FALSE;
```

In den obigen Beispielen wird (falls es sich bei den Zuweisungen um die statisch ersten Zuweisungen zu den Variablen handelt) A_FCB als numerisch, TEXT als String der Länge 33 und XY als Bitmuster festgelegt.

Da zum Zeitpunkt der Compilierung noch nicht festgestellt werden kann, welchen Typ das Symbol ID1FILE hat, wird FNAM standardmäßig zum numerischen Datentyp (andernfalls müsste bereits bei der Compilierung die Symboldatei zugewiesen werden, die erst beim Prozedurablauf benutzt werden soll. Dies müsste auch dann erfolgen, wenn die Prozedur für eine völlig andere BS2000-Version gedacht ist). Variable sollten daher besser durch die Zuweisung eines Anfangswertes deklariert werden, wenn ihnen erst später über ein Symbol ein Datum aus dem Diagnoseobjekt zugewiesen werden soll, das auch nichtnumerischer Art sein kann.

Beispiele

```
A_FCB  := X'ABC' ;
FNAM   := ' ' * 54 ;
FNAM   := A_FCB.ID1FILE ;
```

Auf der rechten Seite einer Zuweisung dürfen auch Funktionsaufrufe (Standardfunktionen) und Ausdrücke stehen.

> ¡ Wird ein String einem bereits initialisierten String anderer Länge zugewiesen, so wird bei kürzerem Ziel-String der Quell-String abgeschnitten, bei längerem Zielstring mit Leerzeichen aufgefüllt.

5.7.3.8 Pseudostrukturen

Mit den Pseudobasen CURRENT und PARAMETER können Sie auf Betriebsdaten des Programms DAMP zugreifen.

Die Pseudobasis CURRENT

In Diagnose-Algorithmen werden nicht nur die Daten des Diagnoseobjekts, sondern mitunter auch „Betriebsdaten“ des Programms DAMP benötigt. Diese erreicht man über die Pseudobasis CURRENT.betriebsdatum. Die verfügbaren Informationen sind nachfolgend erklärt.

> ¡ Bis auf CURRENT.ALET, CURRENT.ATYPE, CURRENT.ERROR, CURRENT.SPID und CURRENT.SEGMENT sind die Felder nicht überschreibbar. Zuweisen eines Wertes auf nur lesbare Felder führt zu einem Compilierungsfehler.

CURRENT.ALET

Beim Zugriff auf Datenräume (vgl. CURRENT.ATYPE) muss CURRENT.ALET mit dem gewünschten Wert für ALET (numerischer Wert der Länge 4) versorgt werden. Die zugehörige TID muss mit der Standardprozedur NEW_TASK (siehe "Standardprozeduren") eingestellt werden.

CURRENT.ATYPE

Standardmäßig werden in PRODAMP Adressen als virtuell interpretiert. Es gibt aber auch die Möglichkeit, den Realspeicher oder die Hardware System Area (HSA) sowie Datenräume zu adressieren. Hierzu dienen das Pseudo-Symbol CURRENT.ATYPE und die Pseudokonstanten VIRTUAL, REAL, HSA, ABS_ADDRESSING, ALET und SPID, die man an dieses Pseudosymbol zuweisen kann. Bei der Zuweisung wird definiert, welcher Speicher in den folgenden Statements adressiert werden soll.

Durch Zuweisung von ABS_ADDRESSING an das Pseudosysmbol CURRENT.ATYPE kann im VM2000-Gesamtsled, z.B. zur Auswertung des Hypervisors, absoluter Speicher adressiert werden.

Beispiel 1 zu CURRENT.ATYPE: Adressieren der HSA

```
HSA_START  := CURRENT.HSA;
TEST_VALUE := 0;
ARRANGE
  .TEST_SYMBOL : TYPE=NUMERIC,LENGTH=1,RELATIVE=0;
END ARRANGE ;
CURRENT.ATYPE := HSA;
TEST_VALUE := HSA_START.TEST_SYMBOL;
IF CURRENT.ERROR <> 0 THEN
  MESSAGE ( 'Hardware System Area ist nicht adressierbar !!' );
ELSE
  ....
END IF;
CURRENT.ATYPE := VIRTUAL;
```

Einige Einschränkungen müssen beachtet werden: der eingestellte Speichertyp gilt nur beim Ansprechen von Speicherbereichen über Symbole und ist auch nur lokal in der aktuellen Prozedur wirksam. Sollen aufgerufene Prozeduren auch auf Realspeicher, HSA, absoluten Speicher oder Datenräume zugreifen, muss lokal in diesen Prozeduren der Speichertyp umgeschaltet werden. Ausnahme zu dieser Regel ist die vordefinierte Prozedur DUMP_MEMORY (siehe "Standardprozeduren"), welche es erlaubt, auch Real-, HSA-Speicher, absoluten Speicher oder Bereiche aus Datenräumen direkt auf Liste auszugeben.

Bei den Angaben ALET und SPID müssen vor dem Zugriff über CURRENT noch die gewünschten Werte für ALET bzw. SPID gesetzt werden, wie in dem folgenden Beispiel gezeigt wird:

Beispiel 2 zu CURRENT.ATYPE:

```
ARRANGE
  .TEST.SYMBOL : TYPE = NUMERIC, LENGTH = 4, OFFSET = 0;
END ARRANGE;
CURRENT.ATYPE := ALET;
CURRENT.ALET := X'01010003';
PTR := 0;
OUT := PTR.TEST_SYMBOL;
IF CURRENT.ERROR <> 0 THEN
  MESSAGE ( 'Zugriff-Fehler, ALET = '
          + HEX_STRING(CURRENT.ALET,8) + ', TID = '
          + HEX_STRING(CURRENT.TID,8) );
ELSE
...
END IF;
CURRENT.ATYPE := VIRTUAL;
```

Bei der Angabe von ALET wird zum Zugriff immer die aktuell eingestellte TID verwendet, die gegebenenfalls mit der Standardprozedur NEW_TASK (siehe "Standardprozeduren") neu gesetzt werden kann.

Weiter werden Symbole, die automatisch lokalisiert werden, **immer** virtuell adressiert. Will man den Realspeicher, den absoluten Speicher oder die HSA adressieren, muss also immer eine Basis-Adresse angegeben werden (wie im obigen Beispiel).

Es ist aber mehr erlaubt als nur die einfache Zuweisung. Das Pseudo-Symbol CURRENT.ATYPE hat intern den Typ numerisch. Man muss also nicht explizit VIRTUAL, REAL, HSA, ABS-ADDRESSING, ALET oder SPID angeben. Man kann zum Beispiel auch mit dem Wert Berechnungen ausführen (wenig sinnvoll), den aktuellen Wert in andere Variablen übernehmen oder zum Beispiel einen Speichertyp als Parameter für eine Prozedur mitgeben. Wird ein ungültiger Wert an CURRENT.ATYPE zugewiesen, wird dies nur zur Laufzeit bemerkt, und die Prozedur abgebrochen. Zur Zeit der Compilierung werden die Werte nicht auf Gültigkeit geprüft.

Beispiel 3 zu CURRENT.ATYPE

```
CURRENT.ATYPE:=35
```

Die Zuweisung ist syntaktisch korrekt.

Beispiel 4 zu CURRENT.ATYPE: Parameter-Übergabe

Es wird eine Prozedur aufgerufen, die den Inhalt der Realadresse 0 als abdruckbaren String zurückliefert.

```
AUFRUFER:
  VALUE := '        ';
  PRINTABLE_VALUE ( 0, REAL, VALUE );
PROZEDUR PRINTABLE=VALUE :
  LOC := 0;
  ARRANGE
    .ADDR     : TYPE=NUMERIC, LENGTH=4, RELATIV=0;  'PARAMETER 1'
    .SPEICHER : TYPE=NUMERIC, LENGTH=4, RELATIV=4;  'PARAMETER 2'
    .ZIEL     : TYPE=STRING,  LENGTH=8, RELATIV=8;  'PARAMETER 3'
    .INHALT   : TYPE=NUMERIC, LENGTH=4, RELATIV=0;
  END ARRANGE
  CURRENT.ATYPE := PARAMETER.SPEICHER;
  LOC := PARAMETER.ADDR;
  PARAMETER.ZIEL := HEX_STRING ( LOC.INHALT, 8 );
  RETURN;
```

CURRENT.CPU

CURRENT.CPU enthält den Namen der CPU aus dem Diagnoseobjekt als String der Länge 8. Für eine als PAM-Datei geöffnete Datei wird die CPU des aktiven Systems ausgegeben.

CURRENT.CONFIGURATION

CURRENT.CONFIGURATION enthält die Bezeichnung der Hardware-Konfiguration aus dem Diagnoseobjekt als String der Länge 21 (z.B. 7.500- S210-60). Die Ausgabe entspricht der Bezeichnung beim Kommando /SHOW-SYSTEM-INFORMATION. Für eine als PAM-Datei geöffnete Datei wird die Konfiguration des aktiven Systems ausgegeben.

CURRENT.CSMA

CURRENT.CSMA enthält die (absolute) Anfangsadresse des Common-Shadow-Memory (CSMA), also einen numerischen Wert der Länge 4.

CURRENT.DTYPE

CURRENT.DTYPE enthält einen numerischen Wert der Länge 1 Byte (8 Bit), der das geöffnete Medium beschreibt. Dieser Wert hat folgendes Format:

Es bedeutet:

Medium = 0 (B'0000') : Medium nicht definiert
Medium = 1 (B'0001') : System
Medium = 2 (B'0010') : Objekt kann ausgewählt werden
Medium = 3 (B'0011') : Dump
Medium = 4 (B'0100') : PAM-Datei
Medium = 5 (B'0101') : Selbstlader

Der Fall Medium = 2 kann nur im Dialog auftreten, wenn die Dumpdatei mehrere Objekte enthält und im INF-Schirm noch keine abschließende Auswahl getroffen wurde (z.B. VM2000-Gesamtsled, SLED vom SLED). Es sollte daher für diesen Fall ein Programmabbruch vorgesehen und im INF-Schirm eine Auswahl getroffen werden. Wurde im INF-Schirm keine Auswahl getroffen, ist ein Zugriff auf das Objekt (nur VM2000-Gesamtsled) mit PRODAMP nur im absoluten Adressierungsmodus möglich (CURRENT.ATYPE = ABS_ADDRESSING).

Im Fall Medium = 3 (0011) sind auch f und Dump besetzt. Es bedeutet:

Dump = 0 (B'000') : Sled
Dump = 1 (B'001') : Systemdump
Dump = 2 (B'010') : Userdump

Ist f mit 1 besetzt (nur im Falle von Sled und Userdump), dann handelt es sich bei Sled um eine Snap-Datei und bei Userdump um einen Area-Dump.

Beispiel zu CURRENT.DTYPE:

Will man prüfen, ob es sich bei dem geöffneten Medium um einen Userdump handelt, ist beispielsweise folgende Abfrage möglich:

```
IF  CURRENT.DTYPE / 16 = 3  AND
    CURRENT.DTYPE MOD 8 = 2
THEN
```

CURRENT.DUMPTIME

CURRENT.DUMPTIME enthält Datum und Uhrzeit der Dumpdatei im Format String der Länge 19: jjjj-mm-tt hh:mm:ss. Bei der Diagnose des aktiven Systems wird das aktuelle Datum mit Uhrzeit geliefert.

CURRENT.ERROR

Entdeckt PRODAMP eine Inkonsistenz, die sich intern abfangen lässt, die also nicht den Abbruch der PRODAMP-Prozedur erfordert, wird CURRENT.ERROR auf einen Wert ungleich 0 gesetzt. Fehlt beispielsweise eine gewünschte Seite im Dumpfile, rechtfertigt das noch keinen Abbruch der Prozedur. Wird hingegen ein angegebenes Symbol nicht in der zugeordneten Symboldatei gefunden, bricht PRODAMP ab, da es sich dabei meist um einen Tippfehler handeln dürfte, der im Source der PRODAMP-Prozedur zu korrigieren ist.

Jeder Zugriff auf ein Datum des Diagnoseobjekts kann im Prinzip zu einem Fehler führen, der über CURRENT.ERROR abgefragt werden sollte, da andernfalls der weitere Ablauf der PRODAMP-Prozedur nicht vorhersehbar ist. Nur wenn nacheinander auf mehrere Objekte innerhalb der gleichen Datenstruktur zugegriffen wird und man sicher ist, dass die Datenstruktur die Speicherseite nicht überschreitet, können Sie sich aus Gründen der Performance und der Übersichtlichkeit darauf beschränken, CURRENT.ERROR nach dem ersten Zugriff abzufragen.

> ¡ Der Fehlercode, der u.U. in CURRENT.ERROR hinterlegt wird, ist für die Diagnose unerheblich. Sie sollten CURRENT.ERROR daher nur gegen 0 vergleichen.

Durch Zuweisen des Wertes 0 (CURRENT.ERROR := 0) kann CURRENT.ERROR explizit zurückgesetzt werden. Dies ist jedoch in vielen Fällen unnötig, da PRODAMP den CURRENT.ERROR in folgenden Situationen automatisch zurücksetzt:

- Beim Eintritt in eine Prozedur (CURRENT.ERROR wird prozedurlokal geführt).

- Bei einem erfolgreichen lesenden Zugriff mit einem Symbol auf Daten des Diagnoseobjekts oder auf Betriebsdaten von DAMP (siehe Abschnitt „Symbole").

  Beispiel

  `X:=.ETCBTID, Z:= CURRENT.DTYPE, Y:=PARAMETER .P1`

  Ausnahme: Der lesende Zugriff auf CURRENT.ERROR.
- Bei erfolgreicher Nutzung einer der Standardfunktionen von DAMP.
  Ausnahme: Der Aufruf von PATTERN verändert CURRENT.ERROR nicht.
- Bei erfolgreicher Ausführung der Standardprozeduren NEW_TASK oder READ.

Bei erfolgreicher Ausführung wird der CURRENT.ERROR von allen anderen Standardprozeduren und Zuweisungen innerhalb PRODAMP nicht zurückgesetzt.

Beispiel zu CURRENT.ERROR

```
TEST := P2_FCB.ID2CFLID ; "PSEUDO-HARDVALIDATION"
IF CURRENT.ERROR = 0 THEN
  "GEFAHRLOSER ZUGRIFF MOEGLICH"
  IF P2_FCB.ID2IND1 = ID2PRIVC THEN
    IF P2_FCB.ID2LOCK = ID2OUTL THEN
    "DO SOMETHING"
    END IF ;
  END IF ;
END IF ;
```

CURRENT.FILENAME

CURRENT.FILENAME enthält den Namen des gerade geöffneten Diagnoseobjekts als String der Länge 54. Falls das aktive System untersucht wird, enthält der String nur Leerzeichen.

CURRENT.HSA

CURRENT.HSA enthält bei SLED-Dateien die (absolute) Anfangsadresse der Hardware-System-Area, also einen numerischen Wert der Länge 4. Da nur SLED-Dateien eine Hardware-System-Area besitzen, ist bei anderen Dumpdateien der Inhalt von
CURRENT.HSA=0.

CURRENT.ITN

siehe CURRENT.TID.

CURRENT.LEVEL

Das Pseudosymbol CURRENT.LEVEL enthält einen numerischen Wert der Länge 1, der die Schachtelungstiefe bei Unterprogrammaufrufen angibt. Diejenige Prozedur, bei der ein RETURN in die DAMP-Umgebung zurückführt, hat CURRENT.LEVEL 0, eine von dieser Prozedur aufgerufene Unterprozedur hat LEVEL 1 usw.

CURRENT.PCB

Das Pseudosymbol CURRENT.PCB enthält die Adresse des im Diagnosefenster 3 eingestellten aktuellen PCB, also einen numerischen Wert der Länge 4. Ist dort kein PCB eingestellt, enthält das Feld den Wert -1 (X'FFFFFFFF').

CURRENT.PTYPE

CURRENT.PTYPE enthält einen numerischen Wert der Länge 1 Byte, der den Ablaufmodus von DAMP anzeigt. Folgende Werte sind möglich:

CURRENT.PTYPE = 0 (B'00000000') : DAMP läuft im Dialogmodus ab
CURRENT.PTYPE = 1 (B'00000001') : DAMP läuft im Prozedurmodus ab
CURRENT.PTYPE = 2 (B'00000010') : DAMP läuft im Batchmodus ab

CURRENT.SEGMENT

CURRENT.SEGMENT wird benötigt, wenn mit sehr großen Adressen (Adressbreite größer als 32 Bit) auf das Diagnoseobjekt zugegriffen werden soll; zwei Situationen sind zu unterscheiden:

1. Zugriff mit großen realen oder absoluten Adressen.

   In CURRENT.SEGMENT ist das 4 GB Segment zu übergeben, welches bei jedem folgenden Lesen mit CURRENT.ATYPE=REAL oder =ABS_ADDRESSING, d.h. bei realer oder absoluter Adressinterpretation, automatisch berücksichtigt wird.

2. PAM-Zugriff auf eine große Datei.

   Ist das geöffnete Diagnoseobjekt eine PAM-Datei, müssen zur Lokalisierung von Daten absolute Adressen angegeben werden, die sich aus der PAM-Seitennummer P und der relativen Distanz D aus folgender Formel ergeben:

   A = (P - 1)*2048 + D mit 0 <= D <= 2047

   Bei großen PAM-Dateien (es sind maximal 534 773 760 Seiten möglich) kann A mehr als 4 Bytes benötigen. Eine PAM-Datei wird daher in insgesamt 256 Segmente mit jeweils 2 097 152 Seiten eingeteilt.

   Zum Auffinden von Daten müssen in einer PRODAMP-Prozedur immer das 4-GB-Segment (in CURRENT.SEGMENT) und die relative Adresse innerhalb des Segments angegeben werden.

   Zur Berechnung des Segments für eine gegebene PAM-Seite sollte die vorzeichenlose Arithmetik (UNSIGNED_ON) wie im unten angegebenen Beispiel verwendet werden.

   CURRENT.SEGMENT wird prozedurlokal geführt und beim Start auf 0 gesetzt.

   In den meisten Fällen (Adressen < 4 GB) ist eine explizite Zuweisung daher nicht notwendig. Eine Versorgung mit einem negativen oder einem Wert größer als 255 führt stets zu einem Laufzeitfehler.

   Beispiel 1 zu CURRENT.SEGMENT: Umrechnung PAM-Seitennummer zu Segment und Adresse

```
UNSIGNED_ON;
   " Die PAM-Seite PAM_PAGE ist umzurechnen in 4 GB-Segment-Nummer
   - mit der CURRENT.SEGMENT unmittelbar versorgt wird -
   und eine 32-Bit breite Adresse 'BYTE_ADDRESS'"
IF PAM_PAGE > X'1FE00000' THEN
   " Jede BS2000 Datei hat hoechstens X'1FE00000' Seiten "
MESSAGE ('** FEHLER: PAMSEITE ZU GROSS **');
RETURN WINDOW=CURRENT.WNDNO;
END IF;
CURRENT.SEGMENT := (PAM_PAGE-1)/X'200000';
   "dividiert durch 2 hoch (32-11)"
BYTE_ADDRESS    := (PAM_PAGE-1)*X'800';
   "mult. mit 2 hoch 11"
```

Beispiel 2 zu CURRENT.SEGMENT: Absolutes Lesen von Adresse X'1 8000 0000'

```
CURRENT.A_TYPE  := ABS_ADDRESSING;
   CURRENT.SEGMENT := 1;
   A:=X'80000000';
   "Adresse im 4 GB-Segment"
   ARRANGE .FULLWORD: LENGTH=4,TYPE=NUMERIC,OFFSET=0;
   Wert := A.FULLWORD;
   "Zugriff auf ein Wort an Adresse X'180000000'"
   IF CURRENT.ERROR = 0 THEN
    "u.s.w"
```

CURRENT.SPID

CURRENT.SPID muss beim Zugriff auf Datenräume (vgl. CURRENT.ATYPE) mit dem gewünschten Wert für SPID (String der Länge 8) versorgt werden.

CURRENT.TIME

CURRENT.TIME enthält die seit LOGON vergangene CPU-Zeit in Millisekunden, als numerischen Wert der Länge 4. Wird der Maximalwert (ca. 25 Tage) überschritten, dann wird der Maximalwert zurückgeliefert und gleichzeitig CURRENT.ERROR ungleich 0 gesetzt.

CURRENT.TID, CURRENT.TSN und CURRENT.ITN

Die Pseudosymbole CURRENT.TID und CURRENT.TSN erlauben den Zugriff auf die aktuelle Task. CURRENT. ITN bedeutet das rechte Halbwort des TID, das die Task zum Diagnose-Zeitpunkt eindeutig bestimmt.

Bei System-, User und Areadumps ist die aktuelle Task stets die Dump-Task und kann nicht geändert werden. Bei SLED-, SNAP-Dumps oder im aktiven System wird die aktuelle Task durch Auswahl einer Task im Status-Fenster (W2), durch Eingabe einer TID (bzw. ITN) oder TSN in der Kopfzeile eines Fensters oder von PRODAMP aus durch Aufruf der Prozedur NEW_TASK festgelegt. Die jeweils eingestellte aktuelle Task ist über CUR-RENT.TID, CURRENT.ITN bzw. CURRENT.TSN ansprechbar. Alle Zugriffe zu taskspezifischen Tabellen (TCB, JCB etc.) sowie zu Adressen im Benutzeradressraum beziehen sich immer auf die eingestellte aktuelle Task.

> ¡ CURRENT.TID und CURRENT.ITN sind numerische Typen, CURRENT.TSN ist ein Stringtyp der Länge 4.

CURRENT.TSN

siehe CURRENT.TID.

CURRENT.VERSION

Das Pseudosymbol CURRENT.VERSION enthält die aktuelle BS2000-Version des Diagnoseobjekts als String der Länge 4 in der Form XX.X, also z.B. 21.0. Hierbei handelt es sich um die Version des aktuellen Systems, falls eine Datei als normale PAM-Datei geöffnet ist.

CURRENT.WNDNO

Das Pseudosymbol CURRENT.WNDNO enthält die Nummer des „aktuellen“ Diagnosefensters, d.h. desjenigen Fensters, das bei der folgenden Bildschirmausgabe als oberstes am Bildschirm erscheinen würde. Da durch ARRANGE die Reihenfolge der Fenster verändert wird, kann man CURRENT.WNDNO zum Beispiel dazu benutzen, bei Fehlern in der Fensterzuweisung wieder das PRODAMP-Fenster als aktuelles Fenster einzustellen.

Die Pseudobasis PARAMETER

PRODAMP-Prozeduren können beim Aufruf aus der DAMP-Programmebene, wie auch beim Aufruf als Unterprozedur aus einer PRODAMP-Prozedur, Parameter übergeben werden (siehe auch Abschnitt „Mit Prozeduren arbeiten (Spezialfenster PROC)").Beim Aufruf wird aus den Parametern ein Parameterbereich gebildet, der die Werte der Parameter aneinander gereiht enthält. In den Parameterbereich werden numerische Datenwerte und Bitmuster stets rechtsbündig in ein 4 Byte langes Feld eingetragen.

Der statisch erste Aufruf aus einer PRODAMP-Prozedur heraus dient dem Compiler zur Definition des Parameterbereiches. Der Aufruf setzt sich aus dem Namen der gerufenen Prozedur und einer geklammerten Liste der aktuellen Parameter zusammen.
Von PRODAMP werden auch Unterprozeduren ohne Parameter unterstützt. Rekursiver Aufruf ist erlaubt.

Um auf die Parameter zugreifen zu können, müssen mit der PRODAMP-Anweisung ARRANGE Symbole vereinbart werden. Über die Pseudobasis PARAMETER können die Symbole angesprochen werden.

Beispiel 1 zur Pseudobasis PARAMETER

Die Prozedur GETOPC soll den an der Adresse MOD + ADD gefundenen Opcode im Symbol OPC zurückliefern.

Prozedur GETOPC:

```
ARRANGE
.MOD : TYPE = STRING, RELATIVE = 0, LENGTH = 8;
.ADD : TYPE = NUMERIC,RELATIVE = 8, LENGTH = 4;
.OPC : TYPE = NUMERIC,RELATIVE =12, LENGTH = 4;
END ARRANGE ;
ARRANGE
.BYTE : TYPE=NUMERIC,RELATIVE=0,LENGTH=1;
END ARRANGE ;
A := ADDRESS ( PARAMETER.MOD,'CP' ) + PARAMETER.ADD ;
PARAMETER.OPC := A.BYTE ;
```

Prozeduraufruf:

```
MOD := 'DOPEN' ;
ADD := X'4ADC' ;
GETOPC ( MOD, ADD, OPC ) ;
IF OPC = X'47' THEN
.....
```

Den Strukturvariablen PARAMETER.XX darf ein Wert zugewiesen werden. Dieser Wert wird den Parameter-Variablen zugewiesen, die von der rufenden Prozedur an gleicher Stelle verwendet werden.

> ¡ Zum Übersetzungszeitpunkt sind Typen und Längen der Parametersymbole und aller anderen Strukturvariablen noch nicht bekannt (siehe Abschnitt „Mit Prozeduren arbeiten (Spezialfenster PROC)").

Beispiel 2 zur Pseudobasis PARAMETER

Trotz anderer Angabe werden die Daten mit den Standardwerten „numerisch“ und „Länge 4“ übertragen:

```
ARRANGE
  .DST:TYPE=STRING,RELATIVE=0,LENGTH=8;
  .SRC:TYPE=STRING,RELATIVE=8,LENGTH=8;
END ARRANGE
PARAMETER.DST:=PARAMETER.SRC;
```

Die richtige Zuweisung erreicht man hier nur durch das Einführen einer Hilfsvariablen:

```
STR:=' ' * 8;
STR:=PARAMETER.SRC;
PARAMETER.DST:=STR;
```

Die Parameter werden bei einem Aufruf aus der DAMP-Programmebene über die Anweisungen RESUME-PRODAMP-PROGRAM oder START-PRODAMP-PROGRAM übergeben. Es können jedoch in diesem Fall keine Ergebnisse in die aufrufende Ebene ausgegeben werden. Das ist nur möglich, wenn eine Prozedur eine Unterprozedur aufruft. Es lassen sich aber durchaus Prozeduren schreiben, die sowohl beim Aufruf aus der Programmebene als auch als Unterprogramm sinnvoll arbeiten, wenn man via CURRENT.LEVEL abfragt, auf welcher Ebene man sich befindet.

Es werden also alle Parameter nach der Methode „call by reference“ übergeben, d.h., es wird die Adresse des Parameters an die Prozedur übergeben. Literale und Ausdrücke können aber auch direkt als Parameter übergeben werden. Die entsprechenden Strukturvariablen können in der gerufenen Prozedur auch beschrieben werden, transportieren aber keine Werte zurück.

Mit ARRANGE können verschiedene Parameterlisten definiert werden, die dann jeweils nach einem INTERRUPT der Pseudobasis PARAMETER überlagert werden können. Es müssen zwar bei einem RESUME-PRODAMP-PROGRAM alle Parameter angegeben werden, die nach dem zugehörigen INTERRUPT ausgewertet werden, es lässt sich jedoch ein „geführter Dialog“ programmieren.

Beispiel 3 zur Pseudobasis PARAMETER

```
ARRANGE
.TSN : TYPE=STRING,LENGTH=4, RELATIVE=0;
END ARRANGE;
MESSAGE ( 'Bitte TSN eingeben. (RESUME fenster,''tsn'')' ) ;
INTERRUPT;
TASK := PARAMETER.TSN ;
ARRANGE
.NUM : TYPE=NUMERIC,LENGTH=4,RELATIVE=0;
END ARRANGE;
MESSAGE ( 'Bitte Anzahl Durchläufe angeben. (RESUME fenster,num)');
INTERRUPT;
DURCHL := PARAMETER.NUM ;
```

Beim symbolischen Zugriff mit der Pseudobasis PARAMETER wird lediglich überprüft, ob sich das zuzugreifende Datum innerhalb des vom Aufrufer mitgegebenen Parameterbereichs befindet; wenn dies nicht der Fall ist, erfolgt ein Programmabbruch mit entsprechender Laufzeitfehlermeldung (siehe Beispiele 4 und 5).

Andere Fehler - die Typen der Aufrufparameter passen nicht zu den in der gerufenen Prozedur mit ARRANGE definierten Feldern - können von DAMP nicht erkannt werden, und führen somit zur Laufzeit zu unvorhersehbaren Effekten.

Mit der Standardfunktion LENGTH kann die Gesamtlänge des Parameterbereichs in der gerufenen Prozedur abgefragt werden, z. B. um in der gerufenen Prozedur auf fehlende Parameter zu reagieren.

Beispiel 4 zur Pseudobasis PARAMETER: Überprüfung der Länge des Parameterbereichs

Das folgende PRODAMP-Programm soll mit zwei Parametern, einer Adresse und einem optional anzugebenden Counter aufgerufen werden. Ist der 2. Parameter nicht versorgt, soll stattdessen mit einer Voreinstellung gerechnet werden.

```
"Layout des Parameterblocks (2. Parameter darf fehlen )"
ARRANGE .ADDR_OF_STRUCT : OFFSET=0,LENGTH=4,TYPE=NUMERIC; "1.Parameter"
  .COUNTER : OFFSET=4,LENGTH=4,TYPE=NUMERIC; "2.Parameter"
END ARRANGE;
" Uebernahme der Parameter in die Variablen ADDR_OF_STRUCT und COUNTER. "
" Ist Counter nicht angegeben, wird COUNTER=100 gesetzt."
L:=LENGTH('*PARAMETER','DS');
IF CURRENT.ERROR <> 0 THEN
MESSAGE (' Programm abgebrochen. DAMP ab Version V4.2 erforderlich ');
RETURN WINDOW=CURRENT.WNDNO;
ELSIF L=4 THEN
ADDR_OF_STRUCT:= PARAMETER.ADDR_OF_STRUCT;
COUNTER       := 100;
ELSIF L=8 THEN
ADDR_OF_STRUCT:=PARAMETER.ADDR_OF_STRUCT;
COUNTER:=PARAMETER.COUNTER;
ELSE
MESSAGE ('Programm abgebrochen. Falsche Parameterversorgung');
END IF;
" u.s.w. ( ADDR_OF_STRUCT und COUNTER sind nun versorgt )"
```

Beispiel 5 zur Pseudobasis PARAMETER: Künstlicher Parameterbereich

In manchen Fällen ist es nützlich, dass für ein PRODAMP-Programm ein Arbeitsbereich in dem Parameterbereich vorgesehen wird, auf den dann „frei“ zugegriffen werden kann. Dies lässt sich sehr einfach dadurch realisieren, dass sich das Programm ein zweites Mal, nun mit erweitertem Parameterbereich aufruft:

```
PROZEDUR XY
"Parameterlayout"
ARRANGE
  .ADDR_IN: TYPE=NUMERIC,LENGTH=4,OFFSET=0;
  .WORKAREA: TYPE=STRING,LENGTH=64,OFFSET=4;
END ARRANGE;
"Bei direktem Aufruf soll nur ADDR_IN versorgt sein"
L:=LENGTH('*PARAMETER','DS');
WORKAREA := #'00'*64;
IF L = 4 THEN
  XY(PARAMETER.ADDR_IN,WORKAREA); "rekursiver Aufruf"
```

```
ELSIF L <> 4+64 THEN
  MESSAGE(' Falsche Parameterversorgung oder DAMP vor V4.2 ');
END IF;
"Nun steht dem Programm XY im Parameterbereich eine mit binaer Null
initialisierte WORKAREA zur Verfügung. "
```

5.7.3.9 Vordefinierte Variablen

In PRODAMP stehen zwei vordefinierte Variablen OPC_TABLE und SVC_TABLE zur Verfügung, mit denen ein Zugriff auf DAMP-interne Tabellen geregelt werden kann. Zunächst können diese Variablen wie normale numerische Variablen eingesetzt werden, d.h. es können ihnen Werte zugewiesen werden und sie können in arithmetischen Ausdrücken verwendet werden. Es ist jedoch zu beachten, dass die Länge dieser Variablen kleiner als die von normalen numerischen Variablen ist und damit nicht jeder numerische Wert zugewiesen werden kann. Ein zugewiesener Wert dient aber gleichzeitig auch als Index zur Lokalisierung eines Eintrages innerhalb einer DAMP-Tabelle, mit dem sich dann Symbole, die diesen Eintrag beschreiben, adressieren lassen.

OPC_TABLE

Diese Variable ist 2 Bytes lang und dient als Index für einen Eintrag in der DAMP-Tabelle, die den Befehlscode beschreibt und intern zur Disassemblierung verwendet wird. Ist der Inhalt von OPC_TABLE kleiner als 256 (nur 1 Byte belegt), so wird dieser als Operationscode eines Befehls aufgefasst. Ist der Inhalt größer als 255 (2 Bytes belegt), werden hierunter die ersten beiden Bytes eines Befehls verstanden, von denen das zweite ein Subcode ist. Ein Eintrag in der Befehlstabelle wird durch folgende DSECT beschrieben:

```
INST     DSECT
INSTTYPE DS    X           INSTRUCTION TYPE
INSTNO   EQU   0              NO VALID INSTRUCTION
INSTRR   EQU   4              RR INSTRUCTION
INSTRX   EQU   8              RX INSTRUCTION
INSTRS   EQU   12             RS INSTRUCTION
INSTSI   EQU   16             SI INSTRUCTION
INSTSS   EQU   20             SS INSTRUCTION
INSTUN   EQU   24             UNKNOWN INSTRUCTION TYPE
INSTFLAG DS    X           FLAG
INSTPRIV EQU   X'80'          PRIVILEGED OPERATION
INSTSVAL EQU   X'40'          SUBFUNCTION VALID/AVAILABLE
INSTSVMN EQU   X'20'          SUBFUNCTION MNEMONIC VALID
INSTPSMN EQU   X'10'          PSEUDO MNEMONIC AVAILABLE
INSTFPI  EQU   X'08'          FLOATING POINT INSTRUCTION
INSTSPEC EQU   X'04'          SPECIAL OPERATION
INSTADW  EQU   X'03'          ACCESS DOUBLE WORD
INSTAWD  EQU   X'02'          ACCESS WORD
INSTAHW  EQU   X'01'          ACCESS HALFWORD
INSTXCPT DS    X           EXCEPTIONS
INSTOP1  EQU   X'80'          OPERAND 1 EXCEPTION
INSTOP1M EQU   X'40'          OPERAND 1 = MASK/R0=0 IF RR,RX
INSTOP1E EQU   X'20'          OPERAND 1 = EVEN/EXTENDED
INSTOP2  EQU   X'10'          OPERAND 2 EXCEPTION
INSTOP2M EQU   X'08'          OPERAND 2 = MASK/R0=0 IF RR
INSTOP2E EQU   X'04'          OPERAND 2 = EVEN/EXTENDED
INSTOP3  EQU   X'02'          OPERAND 3 EXCEPTION
INSTOP3M EQU   X'01'          OPERAND 3 = MASK
INSTOPC  DS    X           OPERATION CODE
INSTSVC  EQU   X'0A'          OPERATION CODE = SVC
INSTRS2  EQU   X'20'          RS INSTRUCTION WITH 2 SIZES
```

```
 INSTOMN  DS    CL5         INSTRUCTION MNEMONIC
 INSTO2M  DS    X           MASK FOR OPERAND 2
 INSTFCT  DS    0XL6        SUBFUNCTION
 INSTFCD  DS    X           FUNCTION DISPLACEMENT
 INSTMSK  DS    X           FUNCTION MASK
 INSTFCU  EQU   X'F0'          FUNCTION CODE IN UPPER HALFBYTE
 INSTFCL  EQU   X'0F'          FUNCTION CODE IN LOWER HALFBYTE
 INSTFCF  EQU   X'FF'          FUNCTION CODE IN FULL BYTE
 INSTPTR  DS    A           FUNCTION POINTER
          ORG   INSTFCT
 INSTFPT  DS    X           FUNCTION PSEUDO TYPE (RR ONLY)
 INSTFTM  EQU   X'FC'          FUNCTION MASK FOR PSEUDO TYPE
 INSTFTD  EQU   X'03'          FUNCTION MASK FOR DISPLACEMENT
 INSTFMN  DS    CL4         FUNCTION MNEMONIC
 INSTILEN EQU   *-INST      ITEM LENGTH
```

Das folgende Beispiel soll verdeutlichen, wie durch die Variable OPC_TABLE Einträge in der entsprechenden DAMP-Tabelle angesprochen werden können. Voraussetzung ist, dass eine Disassemblierungstabelle zugewiesen ist, wie es beim Öffnen eines Diagnoseobjekts geschieht. Ist dies nicht der Fall, sollte eine solche Tabelle mit der Anweisung MODIFY-OBJECT-ASSUMPTIONS bekannt gegeben werden.

Beispiel zu OPC_TABLE

```
OPC_TABLE := X'B223';
MNEMO     :=  ' '*7;
PSEUDO    :=  'none';
INSTSVAL  := P'40';
INSTSVMN  := P'20';
INSTTYPE  := OPC_TABLE.INSTTYPE;
IF CURRENT.ERROR <> 0 THEN
   MESSAGE ( 'Keine Befehlstabelle verfuegbar.' );
   RETURN;
END IF;
IF INSTTYPE = 0 THEN
   MNEMO := 'invalid';
ELSIF INSTTYPE = 24 THEN
   MNEMO := 'unknown';
ELSE
   MNEMO := OPC_TABLE.INSTOMN;
END IF;
IF INSTSVAL + INSTSVMN IN OPC_TABLE.INSTFLAG THEN
   IF OPC_TABLE > 255 THEN
      PSEUDO := OPC_TABLE.INSTFMN;
   ELSE
      PSEUDO := '    ';
   END IF;
END IF;
MESSAGE ( 'Mnemonic: '+MNEMO+' Pseudo: '+Pseudo );
```

SVC_TABLE

Diese Variable ist 1 Byte lang und dient als Index für einen Eintrag in der DAMP-internen SVC-Tabelle, die für jeden SVC eine 8 Bytes lange Mnemonic enthält. Für diese Einträge ist in der Symboldatei keine DSECT verfügbar. Die Einträge müssen mittels einer ARRANGE-Anweisung beschrieben werden, wie folgendes Beispiel verdeutlicht:

Beispiel zu SVC_TABLE

```
SVC_TABLE := X'5C';
ARRANGE .MNEMO : TYPE = STRING, LENGTH = 8, OFFSET = 0;
END ARRANGE;
MNEMO := ' '*8;
MNEMO := SVC_TABLE.MNEMO;
IF CURRENT.ERROR <> 0 THEN
   MESSAGE ( 'Keine SVC-Tabelle verfuegbar.' );
ELSE
   MESSAGE ( 'SVC ' + HEX_STRING(SVC_TABLE,2)+' = '+MNEMO );
END IF;
```

5.7.3.10 Standardprozeduren

Übersicht

Die in PRODAMP enthaltenen Standardprozeduren sind nur innerhalb von Prozeduren aufrufbar. Die Syntax der Prozeduraufrufe entspricht der Pascal-Syntax.

In PRODAMP gibt es folgende Standardprozeduren:

Prozedurname	Funktion
COMMAND	Eingeben von DAMP-Anweisungen aus einer Prozedur heraus
DMP_#REFRESH	Erneuern der Datenbereiche
DUMP_MEMORY	Speicherbereich auf SYSLST ausgeben
ENTER_MODULE	Schnittstelle zwischen PRODAMP-Prozeduren und Assembler-Modulen
EXTRACT	Manipulieren von Strings
INSERT	Manipulieren von Strings
LIST	String auf SYSLST ausgeben
LIST_CONTROL_BLOCK	Kontrollblock auf SYSLST ausgeben
MESSAGE	Ausgeben einer Meldung
NEW_TASK	Einstellen einer anderen 'aktuellen' Task
NEXT_WINDOW	Schaltet in einer PRODAMP-Prozedur auf das nächste sichtbare Fenster des DAMP-Bildschirms
READ	Lesen aus einem EDT-Bereich
READ_WINDOW	Unterbricht eine PRODAMP-Prozedur und ermöglicht Eingaben bzw. Markierungen in einem Diagnosefenster
REFERENCE	Definition eines Symbols als Element einer Substruktur
SET_HEADER	Überschrift für die Listenausgabe erzeugen
UNSIGNED_ON	Vorzeichenlose Arithmetik einschalten
UNSIGNED_OFF	Vorzeichenlose Arithmetik ausschalten
WRITE	Schreiben in einen EDT-Bereich

Tabelle 13: Übersicht über PRODAMP-Standardprozeduren

Die aufgeführten Namen sollten nicht für eigene Prozeduren verwendet werden, da dies zu Fehlinterpretationen des Programms führen kann.

COMMAND
DAMP-Anweisungen eingeben

Mit der Standardprozedur COMMAND können Sie aus einer PRODAMP-Prozedur heraus DAMP-Anweisungen eingeben, beispielsweise um eine private Symboldatei zuzuweisen.

Prozeduraufruf

Operation	Operanden
COMMAND	(text)

Operandenbeschreibung

text gibt den Text der DAMP-Anweisung an. „text“ muss ein String-Ausdruck sein und die Anweisung in der Form enthalten, wie sie im DAMP-Batch- bzw. Prozedurbetrieb eingeben würde.

Nicht akzeptiert werden als „text“:

- REPEAT-SESSION
- RESUME-PRODAMP-PROGRAM
- SHOW-LAST-STATEMENT
- START-OPTION-DIALOG
- START-PATTERN-SEARCH
- START-PRODAMP-EDITOR
- START-PRODAMP-PROGRAM
- START-STATEMENT-SEQUENCE

DMP_#REFRESH
Speicherbereich erneuern

Mit der Standardprozedur DMP_#REFRESH können Sie PRODAMP-interne Datenbereiche erneuern. Dies kann bei der Diagnose des aktiven Systems hin und wieder erforderlich sein.

Prozeduraufruf

Operation	Operanden
DMP_#REFRESH	

DUMP_MEMORY
Speicherbereich ausgeben

Mit der Standardprozedur DUMP_MEMORY lässt sich ein Speicherbereich in einem der DAMP-üblichen Formate auf SYSLST ausgeben. Alle Parameter müssen numerische Ausdrücke sein.

Prozeduraufruf

Operation	Operanden
DUMP_MEMORY	(address,relad,length)

Operandenbeschreibung

address — bezeichnet die Anfangsadresse des Bereichs. Abhängig vom Wert von CURRENT.ATYPE wird virtueller- (Default), realer-, absoluter-, HSA- Speicher oder Bereiche aus Datenräumen ausgegeben.

Bei großen realen und absoluten Adressen kann in „address“ nur ein Wert innerhalb eines 4 GB-Segments angegeben werden. Das zugehörige Segment ist in CURRENT.SEGMENT (siehe "Pseudostrukturen") zu spezifizieren.
CURRENT.SEGMENT wird prozedurlokal geführt und mit „0“ voreingestellt.

relad — gibt die Distanz zur Startadresse an. Setzt man „relad“ auf 0, werden Relativadressen zum Anfang des Bereichs ausgegeben (dieses Format wird von DAMP etwa zur Ausgabe des TCB benutzt).
Gibt man für „relad“ einen negativen Wert an, werden keine relativen Adressen ausgegeben (von DAMP etwa für Ausgabe von Voll-Seiten benutzt).

length — gibt die Länge des auszugebenden Bereichs an.

Beispiel

```
A := .ETCBTFT; DUMP_MEMORY (A, 0, LENGTH ('IDMTFT','DS'))
```

Die erste TFT wird in der Länge der DSECT „IDMTFT“ auf SYSLST ausgegeben (siehe dazu auch die Standardfunktion LENGTH, "Standardfunktionen").

ENTER_MODULE
Aufruf von Modulen

Die Standardprozedur ENTER_MODULE dient als Schnittstelle zwischen PRODAMP-Prozeduren und Assembler-Modulen. Es können aber auch von anderen Sprachen erzeugte Module angesprungen werden, sofern diese die unten beschriebenen Konventionen beachten.

Prozeduraufruf

Operation	Operanden
ENTER_MODULE	(modul, par1, par2, ...)

Operandenbeschreibung

modul — verweist auf das *modul*, das aufgerufen werden soll. Dieses *modul* wird als R-Element in der als Objekt-Bibliothek definierten PRODAMP-Bibliothek erwartet. *modul* muss ein Ausdruck vom String-Typ sein, der den Namen von *modul* in Großschreibung enthält. Nur die ersten 8 Zeichen werden ausgewertet; enthält *modul* weniger als 8 Zeichen, so wird mit Leerzeichen auf 8 Zeichen aufgefüllt.

par1, par2, ... — ist die Liste der Parameter, die dem aufgerufenen *modul* zum Zugriff verfügbar gemacht werden sollen.
Jeder Parameter muss Bezeichner einer PRODAMP-Variablen oder (allgemeiner) ein PRODAMP-Ausdruck sein. Diese Ausdrücke oder Variablen können von beliebigem Typ sein. Die Liste par1, par2, ... kann leer sein.

Beim Aufruf von *modul* sind die Register wie folgt gesetzt:

R1 zeigt auf einen von PRODAMP versorgten Parameterbereich
R13 zeigt auf einen von PRODAMP angelegten Bereich von 18 Worten (72 Bytes), in dem die Register gesichert werden können
R14 enthält die Rücksprungadresse
R15 enthält die Ansprungadresse

Zum Rücksprungzeitpunkt erwartet PRODAMP, dass die Register R1 bis R12 die Werte enthalten, die zum Zeitpunkt des Aufrufs gesetzt waren.

Der Parameterbereich hat folgendes Format:

Byte 0-1 — Gesamtlänge des Parameterbereichs.

Byte 2-3 — Enthält den Wert 0.

Byte 4-11 — Name des angesprungenen Moduls.

Byte 12-n — Enthält als Übertragungsbereich die Werte der Parameter par1, par2,... lückenlos aneinander gereiht.

- Numerische Parameter und Parameter vom Typ Bitmuster haben dabei immer die Länge 4.

- Die Länge eines Parameters vom Typ String hängt von seiner Definition in der PRODAMP-Prozedur ab (1-133 Bytes).

Der gesamte Parameterbereich darf die Länge einer 4K-Seite (4096 Bytes) nicht überschreiten, d.h. die Gesamtlänge des Übertragungsbereichs darf nicht größer als 4084 Bytes sein.

Beim Rücksprung wird jede als Parameter übergebene Variable mit dem zugehörigen Wert aus dem Übertragungsbereich aktualisiert. Damit wird durch ENTER_MODULE also auch ein schreibender Zugriff auf PRODAMP-Variablen möglich.

Auch ein Modul, das auf andere Weise bereits geladen wurde, kann mit ENTER_MODULE gestartet werden. So kann über die PRODAMP-Prozedur COMMAND die DAMP-Anweisung LOAD-MODULE eingegeben werden, in der eine Ladebibliothek angegeben werden kann.

Ein angesprungenes Modul wird beim Rücksprung zur PRODAMP-Prozedur nur dann entladen, wenn er nicht mit LOAD-MODULE geladen wurde. Das Laden von Modulen mit der DAMP-Anweisung LOAD-MODULE (auch innerhalb von PRODAMP über die COMMAND-Anweisung möglich) kann die Laufzeit von PRODAMP-Prozeduren daher erheblich reduzieren, wenn ein Modul häufiger aufgerufen wird. Außerdem kann in der Anweisung LOAD-MODULE eine Ladebibliothek angegeben werden.

Beispiele

Beispiel einer PRODAMP-Prozedur, die den ASSEMBLER-Modul „TEST“ aufruft, von dem der String „EINGABE“ in „AUSGABE“ geändert wird:

PRODAMP-Prozedur

```
STR := 'EINGABE';
ENTER_MODULE ( 'TEST', STR );
MESSAGE ( STR );
```

ASSEMBLER-Modul TEST

```
TEST     CSECT
TEST     AMODE ANY
         USING *,15
         STM   14,12,12(13)
         MVC   12(3,1),='AUS'
         LM    14,12,12(13)
         BR    14
         END
```

Das Beispiel zeigt auch, wie Sie Register retten und zurückladen können.
(Da TEST keine Register verändert, ist dies hier eigentlich unnötig.)

Beispiel einer PRODAMP-Prozedur zur Laufzeitreduzierung

```
COMMAND ('LOAD-MODULE *P-U-O-L(TEST)');
"Der Fehlerfall führt zum Laufzeitfehler (Abbruch)"
 MESSAGE ('Modul TEST wurde geladen');
"Bei ENTER_MODULE erfolgt nun kein Entladen"
N := 0;
STR := ' '*4;
WHILE (N<100) DO
    ENTER_MODULE ('TEST',STR); N := N+1;
END WHILE;
```

EXTRACT
Strings manipulieren

Die Standardprozedur EXTRACT überträgt aus einem Quellstring von einer anzugebenden Position an so viele Zeichen in einen Zielstring, wie in den Zielstring hineinpassen. Ist die Länge des Zielstrings größer als die Anzahl der zu übertragenden Zeichen, bleiben die restlichen Zeichen des Zielstrings unverändert.

Prozeduraufruf

Operation	Operanden
EXTRACT	(ziel,quelle,position)

Operandenbeschreibung

ziel — muss ein Bezeichner einer bereits initialisierten Variablen vom String-Typ sein. Diese Variable enthält den Zielstring.

quelle — gibt den Quellstring (in Form eines Ausdrucks vom String-Typ) an.

position — gibt die Position des ersten zu übertragenden Zeichens im Quellstring an. Das erste Zeichen des Quellstrings hat dabei die Position 0.

Beispiele

```
A := 'XXXX' ;
EXTRACT ( A, 'Ausgabe fuer TSN 1234 unter TSOS',17 );
```

Nach Ausführung der Anweisungen enthält A den Text „1234“.

```
A := 'funktionieren';
EXTRACT ( A, 'Das wird wohl nicht klappen.',20 );
```

Diese Anweisungsfolge ergibt „klappen.ieren“ als Inhalt von A.
In beiden Beispielen wird angenommen, dass A durch die jeweils angegebene Anweisung initialisiert wurde.

INSERT
Strings manipulieren

INSERT ersetzt die Zeichen eines Zielstrings ab einer anzugebenden Stelle durch die Zeichen eines Quellstrings. Es werden solange Zeichen ersetzt, bis das letzte Zeichen aus dem Quellstring übertragen oder das letzte Zeichen des Zielstrings ersetzt wurde.

Prozeduraufruf

Operation	Operanden
INSERT	(quelle,ziel,position)

Operandenbeschreibung

quelle — gibt den Quellstring an. (Ausdruck vom Typ String)

ziel — muss ein Bezeichner einer bereits initialisierten Variablen vom String-Typ sein. Diese Variable enthält den Zielstring.

position — gibt die Position des Zeichens an, von dem ab der Zielstring überschrieben werden soll.
„position“ gilt relativ zum Anfang des Zielstrings. Das erste Zeichen im Zielstring hat also Position 0.

Beispiele

```
A := 'Ausgabe fuer TSN XXXX unter der Kennung $$$$$$$$.';
INSERT ('1234',A,17); INSERT ('TSOS ',A,40);
```

Nach Ausführung der Anweisungen enthält A den Text:
„Ausgabe fuer TSN 1234 unter der Kennung TSOS.“

```
A := 'Das wird wohl nicht klappen.'
INSERT ( 'funktionieren.',A,20 );
```

Diese Anweisungsfolge ergibt für A den Inhalt:
„Das wird wohl nicht funktion“

In beiden Beispielen wird angenommen, dass A durch die jeweils angegebene Anweisung initialisiert wurde.

LIST
Strings auf SYSLST ausgeben

Mit der Standardprozedur LIST können Sie einen String nach SYSLST ausgeben.

Prozeduraufruf

Operation	Operanden
LIST	(string[,skiplines])

Operandenbeschreibung

string	gibt den auszugebenden String an.
skiplines	legt fest, wie viele Leerzeilen vor dem String eingefügt werden sollen. 0 <= skiplines <= 15. Keine Angabe von skiplines bedeutet skiplines = 0.

LIST_CONTROL_BLOCK
Kontrollblock auf SYSLST ausgeben

Die Standardprozedur LIST_CONTROL_BLOCK kann für die Ausgabe der Daten des angegebenen Kontrollblocks der im Moment geöffneten Dialogfensters auf SYSLST in der symbolischen Maske verwendet werden (siehe dazu Beispiel im Abschnitt "Symbolische Ausgabe"). Alle Substrukturen und Arrays werden zum Stand “revealed” gebracht. Alle Parameter außer cbname müssen numerische Ausdrücke sein.

Prozeduraufruf

Operation	Operanden
LIST_CONTROL_BLOCK	(cbname,address,relad,length)

Operanden

cbname — Gibt den Namen des anzuzeigenden Kontrollblocks an (Ausdruck vom Typ String). Die Symboldatei, die den gewünschten Kontrollblock enthält, muss, falls es sich nicht um eine Standardsymbolbibliothek handelt, mit der Anweisung ADD-SYMBOLS geladen werden (siehe dazu Anweisung ADD-SYMBOLS), bevor die Anweisung benutzt werden kann.

address — Gibt die Startadresse des Bereiches für den Kontrollblock an.
Abhängig vom Adresstyp werden virtuelle (= Standardeinstellung), reale, absolute oder HSA Speicherbereiche oder Data-Space-Bereiche ausgegeben. Im Falle von großen realen und absoluten Adressen, kann für "address" nur ein Wert innerhalb eines 4 GB Segmentes angegeben werden. Das betreffende Segment muss unter CURRENT-SEGMENT angegeben werden. CURRENT.SEGMENT wird auf Basis der lokalen Prozedur versorgt und ist auf "0" voreingestellt.
Hinweis: CURRENT ist eine lokale Referenz in DAMP, die dazu benutzt wird, auf Daten in einem DAMP-Programm zuzugreifen. Der mögliche Wertebereich für "address" umfasst X'00000000' und X'FFFFFFFF'. Die maximale Länge des Feldes beträgt 4 Byte.

Falls eine falsche Adresse und/oder Distanz angegeben wird, wird der Kontrollblock mit falschen Werten angezeigt.

relad — Gibt die Distanz von der Startadresse an. Falls “relad”auf 0 gesetzt ist, werden die Adressen relativ zum Anfang des Bereiches ausgegeben.
Falls ein negativer Wert für “relad” angegeben wird, wird dieser automatisch auf 0 gesetzt. Mögliche Eingabewerte sind die gleichen wie unter "address".

length — Gibt die Länge des auszugebenden Kontrollblocks an.

Beispiel

```
LIST_CONTROL_BLOCK ('FAFG', X'BAD01400', 0, LENGTH ( 'FAFG', 'DS' ));
```

Nach erfolgreicher Ausführung der Anweisung enthält SYSLST folgende Informationen:

```
DSECT : FAFG          + REL : 00000       = ADDR : BAD01400   LEN : 00170
000 TIME_STAMP        : 00000000 = 0      ! 004 CCAD              : BAD00000
008 FTAD              : BAD01570          ! 00C SPAD              : 00000000 = 0
010 FAMEMPTR          : F70DDE40          ! 014 SRB               : 81FB0270
018 COMB              : D14B0234          ! 01C ABTB              : 00000000 = 0
020+WAITBOID :
  020 WAITBOID(1)     : 81BD0263
  024 WAITBOID(2)     : 82060266
  028 WAITBOID(3)     : 8315023C
02C CMXB              : 00000000 = 0      ! 030 PWRD              : 00000000 = 0
034 TSN               : F0F6D9D3 = '06RL' ! 038 SW                : 00000000
03C DPRIM             : 00000240 = 576    ! 040 DSECOND           : 000000C0 = 192
044 DUSID             :'TSOS '            ! 04C IDUSID            :'        '
054 FGGILIST          : 00000000 = 0      ! 058 FN_GENERATED      : 01
```

```
059 TD_GENERATED      : 01                ! 05A TSK_GENERATED     : 01
05B+FLAGS :
  05B RESERVED        : 000006
05E IN_BUF_SIZE       : 0541 = 1345       ! 060 OUT_BUF_SIZE      : 0541 = 1345
062                   : 0000              ! 064 DBLISTPTR         : 00000000 = 0
068 FAHOSTLPTR        : 00000000 = 0      ! 06C CHARTEMPFILE      : 40 = ' '
06D BS2VERS           :'V18.0A0000'       ! 077 UNUSED            : 00 = ' '
078+TRACE             :
  078+HDR             :
    078 DEEP_OK           : 00
    079 RECORDING_POSITION: 00
    07A SS_NUMBER         : 00
    07B FILLER2           : 00
  07C RECORD          :'                                          '
  0BD                 :'                                          '
  0FE                 :'                                          '
  13F                 :'                                          '
16C INF_BUF_NUMBER    : 0004 = 4           ! 16E OF_BUF_NUMBER    : 0008 = 8
```

MESSAGE
Meldung am Bildschirm ausgeben

Die Standardprozedur MESSAGE ohne den optionalen, numerischen Parameter line bewirkt, dass ein angegebener Text in einer der beiden Meldungszeilen der DAMP-Bildschirmmaske (Zeilen 2 und 3) ausgegeben wird, sobald die PRODAMP-Prozedur verlassen und der Bildschirm aufgefrischt wird.
Werden dem Parameter line die Werte 1 oder 2 zugewiesen, wird der angegebene Text **sofort** in der Bildschirmmaske ausgegeben.

> *Parameter line wird nicht angegeben:*
>
> Sobald der Puffer für die beiden Meldungszeilen voll ist, werden weitere Meldungen ignoriert. Es empfiehlt sich daher, nach dem Aufruf von MESSAGE eine der Anweisungen INTERRUPT oder RETURN folgen zu lassen. Bei INTERRUPT können Sie nach der Kenntnisnahme des Textes die Prozedur ggf. mit RESUME fortsetzen. Soll es sich um eine Fehlermeldung handeln und die Prozedur abgebrochen werden, empfiehlt sich RETURN.
>
> *Parameter line wird angegeben:*
>
> Auf diese Weise können in länger laufenden PRODAMP-Prozeduren Zwischenmeldungen ausgegeben werden, die über den aktuellen Bearbeitungsstand der Prozedur Auskunft geben.

Prozeduraufruf

Operation	Operanden
MESSAGE	(text[,line])

Operandenbeschreibung

text gibt den auszugebenden Text an.

line steht für einen numerischen Ausdruck, der 0, 1 oder 2 ergibt. Die Ziffern bezeichnen die Fehlerzeilen, in welche die Meldungen geschrieben werden sollen.
1. Fehlerzeile -> 2. Bildschirmzeile
2. Fehlerzeile -> 3. Bildschirmzeile
Wird für line der Wert 0 festgelegt, entspricht dies dem Prozeduraufruf MESSAGE (text).

NEW_TASK
Aktuelle Task einstellen

Mit der Standardprozedur NEW_TASK wird eine neue „aktuelle Task“ im Sinne von DAMP eingestellt. Dies ist nur für SLED-, SNAP-Dumps und das aktive System sinnvoll. Alle Zugriffe auf task-spezifische Tabellen (ETCB, EJCB etc.) oder auf Adressen im Benutzerspeicher beziehen sich anschließend auf die neue Task.

Prozeduraufruf

Operation	Operanden
NEW_TASK	(task[,map])

Operandenbeschreibung

task gibt die neue aktuelle Task an.

task muss ein String-Ausdruck oder ein numerischer Ausdruck sein; für beide Möglichkeiten wirkt NEW_TASK unterschiedlich.

Ist task ein *String-Ausdruck*, so muss er eine TSN enthalten (maximal 4 Zeichen). Gibt es eine Task mit dieser TSN, so wird sie als aktuelle Task eingestellt, andernfalls wird CURRENT.ERROR gesetzt.

Ist task ein *numerischer Ausdruck*, so werden die letzten 3 Halbbytes des Wertes dieses Ausdrucks als ITN interpretiert. Existiert im Diagnoseobjekt eine aktive Task mit dieser ITN, so wird sie eingestellt. Andernfalls wird die (in der TLT) folgende aktive Task eingestellt. Nur wenn auf diese Weise keine Task gefunden wurde, wird CURRENT_ERROR gesetzt.
Falls der numerische Ausdruck speziell ein Bezeichner einer numerischen Variablen ist, wird in ihr die eingestellte TID zurückgegeben. Auf diese Weise können Sie nacheinander alle aktiven Tasks durchgehen (siehe dazu das Beispiel 3, „Aktuelle Task ändern“ auf "Mit Prozeduren arbeiten (Spezialfenster PROC)").

map gibt an, ob die bereits vorhandene Systemübersicht (CSECT-Map) um die Übersicht der nicht privilegierten Subsysteme der neuen Task erweitert werden soll.
Für den Parameter map können die Werte TRUE oder FALSE eingesetzt werden. Mit TRUE wird die zusätzliche CSECT-Map angefordert; keine Angabe ist gleichbedeutend mit FALSE.

> ¡ Die Zuweisung TRUE ist zum einen mit Performance-Einbußen verbunden und kann zum anderen bei Systemen mit sehr vielen aktiven Tasks leicht zum Speicherüberlauf führen. Der optionale Parameter map sollte also nur verwendet werden, wenn er wirklich benötigt wird.

Beispiel

```
NEW_TASK ( '0A33' ); ——————————————————————————————————————  (1)
TASK_ID := X'AB'; ————————————————————————————————————————  (2)
NEW_TASK ( TASK_ID );
```

(1) Es wird zunächst die Task mit der TSN 0A33 eingestellt. Falls diese Task nicht existiert wird CURRENT.ERROR gesetzt.

(2) Dann wird die Task mit der ITN X'AB' als aktuelle Task eingestellt. Falls diese nicht existiert, wird die in der TLT folgende Task (etwa X'AE') eingestellt und deren TID in TASK_ID zurückgemeldet. Nur wenn X'AB' nicht aktiv ist und keine weiteren aktiven Tasks in der TLT vorhanden sind, wird CURRENT.ERROR gesetzt.

NEXT_WINDOW
Schaltet auf das nächste sichtbare Fenster

Der Aufruf von NEXT_WINDOW setzt einen vorher erfolgten Aufruf von READ_WINDOW voraus und stellt die Pseudosymbole INFIELDS.xxx für das nächste Diagnosefenster des bei READ_WINDOW gelesenen Bildschirms zur Verfügung. Nähere Informationen unter READ_WINDOW.
Bietet der DAMP-Bildschirm kein weiteres sichtbares Fenster, wird das Pseudosymbol CURRENT.ERROR auf einen Wert ungleich 0 gesetzt.

> ¡ Unabhängig vom gerade betrachteten Fenster kann auf die Anweisungszeile des DAMP-Bildschirms immer über das Pseudosymbol INFIELDS.COMMAND zugegriffen werden.

Prozeduraufruf

Operation	Operanden
NEXT_WINDOW	

Beispiel

```
READ_WINDOW; ———————————————————————————————————————————— (1)
WHILE (CURRENT.ERROR = 0) DO ———————————————————————————— (2)
   WRITE (DEC_STRING (INFIELDS.WNDNO)); ————————————————— (3)
   NEXT_WINDOW; ————————————————————————————————————————— (4)
END WHILE;
```

(1) Die Prozedur wird unterbrochen. Nach Drücken der Taste P13 ist die Prozedur wieder aktiv und die Eingaben in den letzten Bildschirm sind intern abgespeichert worden. Das oberste Diagnosefenster des Bildschirms ist das aktuelle Diagnosefenster, auf welches sich INFIELDS.xxx bezieht.

(2) Abbruch, falls NEXT_WINDOW kein weiteres sichtbares Fenster anzeigt.

(3) Hier können die Pseudosymbole INFIELDS.xxx für das aktuelle Fenster ausgewertet werden. In dem Beispiel wird die Fensternummer in den EDT-Bereich geschrieben; da INFIELDS.WNDNO stets existiert, ist eine Auswertung von CURRENT.ER-ROR nicht nötig.

(4) Umschalten auf das nächste Fenster.

READ
Aus einem EDT-Bereich lesen

Die Standardprozedur READ liest sequenziell aus dem aktuellen EDT-Bereich und ordnet den gelesenen Satz einer Stringvariablen text zu. Ist die Stringvariable noch nicht initialisiert, wird ein String maximaler Länge (133 Byte) eingerichtet. Ist die EDT-Zeile zu kurz, wird sie mit Leerzeichen aufgefüllt. Ist sie zu lang, werden überschüssige Zeichen ignoriert.

Neben dem Zugriff auf Diagnose-Daten, die nicht im Diagnoseobjekt zu finden sind (etwa der Repfile), können Sie diese Funktion z.B. auch dazu verwenden, Tabellen-Layouts in separaten Dateien zu hinterlegen und in die PRODAMP-Prozedur einzulesen. Damit erspart man sich aufwändige Initialisierungen.

> ¡ Von einer abwechselnden Verwendung der Prozeduren WRITE und READ ist abzuraten, da durch WRITE eventuell die intern von EDT eingestellte aktuelle Zeilennummer verändert wird. Ein nachfolgendes READ kann unter Umständen eine falsche Zeile liefern.

Prozeduraufruf

Operation	Operanden
READ	(text)

Operandenbeschreibung

text die Stringvariable, der der gelesene Text zugeordnet werden soll.
Maximal 133 Zeichen lang.

READ_WINDOW
Unterbricht eine PRODAMP-Prozedur und ermöglicht Eingaben bzw. Markierungen in Diagnosefenstern

Die Standardprozedur READ_WINDOW führt zu einer Unterbrechung der PRODAMP-Prozedur. Erst nach Drücken der Taste P13 wird die PRODAMP-Prozedur wieder aktiv. In der Zwischenzeit kann in den DAMP-Bildschirmen beliebig gearbeitet werden.
Nach Aktivierung durch die Taste P13 werden die Eingaben in den letzten DAMP-Bildschirm nicht an den DAMP-Bildschirm, sondern an PRODAMP übertragen. Über die Pseudosymbole INFIELDS.xxx (siehe unten) werden Eingaben in das oberste Diagnosefenster des Bildschirms sowie Eingaben in der Kommandozeile verfügbar. Mithilfe der Standardprozedur NEXT_WINDOW kann auf die Eingaben eines eventuell nächsten Diagnosefensters (im gleichen Bildschirm) zugegriffen werden. Ein mehrmaliges Aufrufen der Prozedur NEXT_WINDOW stellt die Eingaben in allen Diagnosefenstern des Bildschirms (von oben nach unten) zur Verfügung.

Neben Eingaben können mittels READ_WINDOW einige Größen bestimmt werden, die den Fenstern zugeordnet sind (wie z.B. Fensternummer). Nicht alle Eingaben sind verfügbar, insbesondere Eingaben in die meisten Spezialfenster. Einzelheiten entnehmen Sie der folgenden Aufzählung, welche die zulässigen Pseudosymbole INFIELDS.xxx enthält.

Der Anwender hat die Möglichkeit, sich eine eigene Benutzeroberfläche für DAMP zu programmieren, indem er in eine PRODAMP-Prozedur READ_WINDOW bzw. NEXT_WINDOW Prozeduren einbaut und zur Datenübertragung stets die Taste P13 verwendet. Hiermit können neue, anwendereigene DAMP-Anweisungen realisiert werden.

Prozeduraufruf

Operation	Operanden
READ_WINDOW	

Auf folgende Pseudosymbole kann zugegriffen werden:

INFIELDS.ADDRESS
enthält eine eventuelle Eingabe in das Feld **Absolutadresse** in der Kopfzeile des betrachteten Fensters. Numerischer Wert der Länge 4.

INFIELDS.ASEL
enthält eine eventuelle Eingabe in das Feld **ASEL** (Address-Space-Selector) in der Kopfzeile des betrachteten Fensters. String der Länge 3.

INFIELDS.ASID
enthält eine eventuelle Eingabe in das Feld **ASID** (Address-Space-Identifier) in der Kopfzeile des betrachteten Fensters. String der Länge 17.

INFIELDS.COMMAND
enthält eine eventuelle Eingabe in der DAMP-Anweisungszeile. String der Länge 72.

INFIELDS.LAYOUT
enthält eine eventuelle Eingabe in das Feld **Fenster-Layout** in der Kopfzeile des betrachteten Fensters. String der Länge 3.

INFIELDS.LENGTH
enthält eine eventuelle Eingabe in das Feld **Länge** in der Kopfzeile des betrachteten Fensters. Numerischer Wert der Länge 1.

INFIELDS.MARK1 bis **INFIELDS.MARK6**
enthält eventuell Adressen, die im betrachteten Fenster markiert worden sind. Numerischer Wert der Länge 4 (pro Adresse).

INFIELDS.RELATIVE
enthält eine eventuelle Eingabe in das Feld **Relativadresse** in der Kopfzeile des betrachteten Fensters. Numerischer Wert der Länge 4.

INFIELDS.STACK
enthält eine eventuelle Eingabe in das Feld **Stack-Nummer** in der Kopfzeile des betrachteten Stack-Fensters (W3). Numerischer Wert der Länge 4.

INFIELDS.SYMBOL
enthält eine eventuelle Eingabe in das Feld **Symbol** in der Kopfzeile des betrachteten Fensters. String der Länge 31. (Aus Kompatibilitätsgründen wird das letzte (d.h. das 32ste) Zeichen des Feldes Symbol ignoriert.)

INFIELDS.TID
enthält eine eventuelle Eingabe in das Feld **TID** in der Kopfzeile des betrachteten Dumpfensters. Numerischer Wert der Länge 4.

INFIELDS.TSN
enthält eine eventuelle Eingabe in das Feld **TSN** in der Kopfzeile des betrachteten Dumpfensters. String der Länge 4.

INFIELDS.WNDNO
enthält die Fensternummer des betrachteten Fensters. Numerischer Wert der Länge 1.

INFIELDS.WNDTSK
enthält die TID, zu welcher die Daten des betrachteten Fensters gehören. Numerischer Wert der Länge 4.

Hinweise

- Erfolgt eine Abfrage des Pseudosymbols, ohne dass eine Feldeingabe im DAMP-Bildschirm vorausging, wird das Pseudosymbol CURRENT.ERROR auf einen Wert ungleich 0 gesetzt. Die Pseudosymbole WNDNO und WNDTSK werden implizit gesetzt, wobei WNDNO immer und WNDTSK auf jeden Fall dann gültig ist, wenn das Fenster Daten einer Benutzertask enthält. Handelt es sich um Systemdaten, enthält WNDTSK einen ungültigen Wert (CURRENT.ERROR wird gesetzt).
- Wird die mit READ_WINDOW unterbrochene PRODAMP-Prozedur mit der Anweisung RESUME-PRODAMP-PROGRAM fortgesetzt, führt jeder Zugriff auf eines der Pseudosymbole zum Abbruch der Prozedur. Eine entsprechende Meldung wird ausgegeben.
- Die Daten, welche in den Pseudosymbolen abgespeichert sind, werden durch eine normale Unterbrechung der PRODAMP-Prozedur mit der Anweisung INTERRUPT nicht zerstört. Sie stehen nach einem Wiedereinstieg in die Prozedur (mit der Anweisung RESUME-PRODAMP-PROGRAM) weiterhin zur Verfügung.
  Bei jedem Neustart einer PRODAMP-Prozedur wird jedoch der Datenbereich, auf den die Pseudobase INFIELDS zugreift, ungültig gesetzt. Dies ist auch der Fall, wenn zwischen den Anweisungen INTERRUPT und RESUME-PRODAMP-PROGRAM eine andere Prozedur gestartet wird.

- Unabhängig von der Anzahl der auf dem Bildschirm angezeigten Fenster, werden insgesamt nur 6 Markierungen übernommen und mit den Prozeduren READ_WINDOW bzw. NEXT_WINDOW dem jeweiligen Fenster zugewiesen. Über die Pseudosymbole INFIELDS.MARK1 bis INFIELDS.MARK6 sind die Markierungen in der PRODAMP-Prozedur verfügbar.

Beispiel

Die folgende Prozedur wartet im Hintergrund, dass ein Adressfeld markiert und mit der Taste P13 übertragen wird. Für diesen Fall gibt die Prozedur das erste an der markierten Adresse gelegene Wort in der Meldungszeile aus.

```
ARRANGE
 .WORD : OFFSET = 0, LENGTH = 4, TYPE = NUMERIC;
END ARRANGE;
B := 0;
WHILE B=B DO
  CURRENT.ERROR := 0;
  READ_WINDOW;
  WHILE CURRENT.ERROR = 0 DO
    A := INFIELDS.MARK1;
    MESSAGE ( HEX_STRING(A,8)+': '+HEX_STRING(A.WORD,8) );
    NEXT_WINDOW;
  END WHILE;
END WHILE;
```

REFERENCE
Lokalisierung eines Symbols, das Element einer Substruktur ist

Die Standard-Prozedur REFERENCE wird nur in Verbindung mit den Standardfunktionen ADDRESS und LENGTH verwendet. Sie dient zur Angabe von Referenzen, wenn es sich bei dem mit ADDRESS oder LENGTH zu behandelnden Symbol um das Element einer Substruktur handelt. Die Prozedur REFERENCE muss für jedes Symbol, das auf dem „Weg“ zum gesuchten Element liegt, in der entsprechenden Reihenfolge aufgerufen werden (siehe Beispiel). Symbole sind dabei als String der maximalen Länge 32 anzugeben. Sie werden durch die REFERENCE-Aufrufe gesammelt, jedoch nicht auf ihre Gültigkeit überprüft. Erst beim Aufruf der Funktionen ADDRESS bzw. LENGTH werden die Strings überprüft.

> ¡ Alle REFERENCE-Aufrufe gelten nur lokal innerhalb einer Prozedur. Sind die Aufrufe beim Verlassen der Prozedur nicht durch Aufruf von ADDRESS oder LENGTH aufgelöst worden, wird der weitere Prozedurablauf abgebrochen. Nach einer Auflösung wird der referenzierte Weg gelöscht. Wird das Element später noch einmal benötigt, müssen neue REFERENCE-Aufrufe gegeben werden.

Prozeduraufruf

Operation	Operanden
REFERENCE	(symbol)

Operandenbeschreibung

symbol gibt das zu lokalisierende Symbol an. String der Länge 31.

Beispiel

```
NKLCB_MDL.COPY_PARAMETER.USER_ADMINISTRATION.WAIT_FACTOR ————————————  (1)
REFERENCE ( 'NKLCB_MDL' );
REFERENCE ( 'COPY_PARAMETER' );
REFERENCE ( 'USER_ADMINISTRATION' ); ————————————————————————————————  (2)
A := ADDRESS ( 'WAIT_FACTOR', 'RF' ); ———————————————————————————————  (3)
```

(1) Dieses SPL-Strukturfeld ist auszuwerten.

(2) Die dreimalige Anwendung der Prozedur REFERENCE bezeichnet den Weg („NKLCB_MDL. COPY_PARAMETER.USER_ADMINISTRATION“), der zum Symbol WAIT_FACTOR führt.

(3) liefert die Adresse des Symbols WAIT_FACTOR, die relativ zum ersten Symbol (hier NKLCB_MDL) der bezeichneten Referenzkette berechnet wird. Mit der Angabe „RF“ wird angezeigt, dass zur Auffindung dieses Symbols REFERENCE-Aufrufe notwendig waren.

SET_HEADER
Listenüberschrift erzeugen

Mit SET_HEADER lässt sich eine Überschrift für die Listenausgabe erzeugen. Die Überschrift wird erstmals nach Aufruf von SET_HEADER ausgegeben und dann bei jedem folgenden Seitenwechsel, bis der Text durch ein neues SET_HEADER geändert wird.

Prozeduraufruf

Operation	Operanden
SET_HEADER	(string,skiplines,reservelines)

Operandenbeschreibung

string	muss ein Ausdruck vom Typ „String“ sein und enthält den Text für die Überschrift.
skiplines	muss ein numerischer Ausdruck sein und gibt die Anzahl der Zeilen an, die vor dem erstmaligen Ausdruck der Überschrift freigelassen werden sollen. 0 <= skiplines <= 255.
reservelines	muss ein numerischer Ausdruck sein und wirkt wie folgt: bleiben auf der aktuellen Seite der Liste weniger als „reservelines“ Zeilen bis zum nächsten Seitenwechsel, wird vor dem erstmaligen Ausdruck der Überschrift ein Seitenwechsel gemacht. 0 <= reservelines <= 255. Die Angabe „255“ erzwingt einen Seitenwechsel.

Beispiel

```
SET_HEADER ('TEXT',0,255); ————————————————————————————————————  (1)
SET_HEADER ('TFT FUER TASK'+CURRENT.TSN, 2, 20); ——————————————————  (2)
```

(1) erzwingt einen Seitenwechsel vor der Überschrift.

(2) Es werden vor der Überschrift 2 Zeilen freigelassen (nach der Überschrift wird immer eine Zeile freigelassen). Wenn auf der aktuellen Seite weniger als 20 Zeilen frei sind, wird ein Seitenvorschub gemacht.

UNSIGNED_ON und UNSIGNED_OFF
Vorzeichenlose Arithmetik ein- bzw. ausschalten

Wenn mit Adressen gerechnet werden soll, ist die vorzeichenbehaftete Arithmetik manchmal nachteilig. Für den numerischen Datentyp von PRODAMP kann durch Aufruf einer Standardprozedur zwischen vorzeichenbehafteter und vorzeichenloser Arithmetik gewählt werden.

Bei vorzeichenbehafteter Interpretation von 32-Bit-Daten hat das führende Bit die Wertig-keit -2^{31}, bei vorzeichenloser Interpretation die Wertigkeit 2^{31}.

Eine PRODAMP-Hauptroutine läuft zunächst mit vorzeichenbehafteter Arithmetik (UNSIGNED_OFF, Voreinstellung für PRODAMP). Vorzeichenlose Arithmetik muss mit der Prozedur UNSIGNED_ON eingeschaltet werden. Sie kann mit UNSIGNED_OFF wieder ausgeschaltet werden.

Bei Aufruf einer Unter-Routine wird dieser arithmetische Ausführungsmodus (vorzeichenbehaftet, vorzeichenlos) in die Unter-Routine vererbt. Eine Änderung des arithmetischen Ausführungsmodus in einer gerufenen Routine hat keine Auswirkung auf den arithmetischen Ausführungsmodus in der rufenden Routine.

Der arithmetische Ausführungsmodus beeinflusst nur das Rechnen mit dem arithmetischen Datentyp der Länge 4 (32-Bit-Daten). Das Adressieren von Datenobjekten wird nicht beeinflusst.

Vorzeichenbehaftete Arithmetik (UNSIGNED_OFF)

Das Rechnen in vorzeichenbehafteter Arithmetik wird in PRODAMP wie bisher durchgeführt. Im Fehlerfall (Überlauf bei Addition, Subtraktion und Multiplikation, Division durch null) wird der PRODAMP-Lauf mit Laufzeitfehler abgebrochen.

Vorzeichenlose Arithmetik (UNSIGNED_ON)

Ein Überlauf bei Addition, Subtraktion und Multiplikation wird ignoriert (das Ergebnis ist „modulo 2^{32} “ korrekt), insbesondere führt ein Überlauf **nicht** zu einem Laufzeitfehler.
Bei Division durch null wird der PRODAMP-Lauf mit Laufzeitfehler abgebrochen. Vergleiche werden bei vorzeichenloser Arithmetik binär als „logische Vergleiche“ durchgeführt.

Bei den Standardfunktionen „DEC_BINARY - Dezimalzahl umwandeln“ und„DEC_STRING - Numerische Werte umwandeln“ wird das Einschalten der vorzeichenlosen Arithmetik mit UNSIGNED_ON ignoriert.

Prozeduraufruf

Operation	Operanden
UNSIGNED_ON	
UNSIGNED_OFF	

WRITE
In einen EDT-Bereich schreiben

Die Standardprozedur WRITE bewirkt, dass ein Text in einen EDT-Bereich (standardmäßig der Bereich 8) ausgegeben wird. Damit ist es möglich, z.B. eine tabellarische Aufbereitung von Systemgrößen in diesen EDT-Bereich schreiben zu lassen und anschließend im EDT auszuwerten.

Der EDT-Ausgabebereich lässt sich mit WRITE („@PROC nn“) jederzeit ändern.

> Strings, die mit dem Symbol „@“ beginnen, werden als EDT-Anweisung interpretiert. Mit „@WRITE "dateiname"“’ können Sie also den Inhalt des EDT-Bereiches in eine Datei sicherstellen.
> In der WRITE-Prozedur können jedoch nur solche EDT-Anweisungen angegeben werden, die auch der F-Modus des EDT akzeptiert. Andere Anweisungen können nur über den Umweg einer EDT-Prozedurdatei eingegeben werden.
>
> PRODAMP simuliert die EDT-Anweisungen @**PROC** und @**END**. Allerdings sind hierbei folgende Einschränkungen zu beachten:
>
> - Es können nur die Prozedurdateien 1 bis 9 verwendet werden.
> - Wird die Nummer einer gültigen Prozedurdatei erkannt, erfolgt keine weitere Syntaxprüfung. Der Rest der Kommandoeingabe wird ignoriert.
> - Das Kommando @END darf nicht abgekürzt werden.
> - Eine Kettung von @PROC und @END mit anderen EDT-Kommandos ist nicht möglich.
>
> Werden alte PRODAMP-Prozeduren, die noch EDT-Kommandos des L-Modus verwenden, unter DAMP erneut übersetzt, sind sie nicht mehr ablauffähig.
>
> Die Standardprozedur WRITE verändert unter Umständen die intern vom EDT eingestellte aktuelle Zeilennummer. Folgt später die Prozedur READ, erhalten Sie dann einen anderen Zeileninhalt als erwartet. Sie sollten daher WRITE und READ nicht abwechselnd verwenden.

Prozeduraufruf

Operation	Operanden
WRITE	(text)

Operandenbeschreibung

text gibt den Stringausdruck an, der in den EDT-Bereich geschrieben werden soll.

5.7.3.11 Standardfunktionen

Die in PRODAMP enthaltenen Standardfunktionen sind nur innerhalb von Prozeduren aufrufbar. Die Syntax der Funktionsaufrufe entspricht der Pascal-Syntax.

Funktionsname	Funktion
ADDRESS	liefert die Adresse eines Moduls oder eines Kontrollblocks
DEC_BINARY	umwandeln eines Dezimalstrings in eine numerische Variable
DEC_STRING	dezimale Druckaufbereitung
HEX_BINARY	umwandeln eines Sedezimalstrings in eine numerische Variable
HEX_STRING	sedezimale Druckaufbereitung
LENGTH	Länge eines Moduls, Kontrollblocks oder Parameterbereichs ausgeben
LOCATION	liefert den Modulnamen zu einer Adresse
PATTERN	wandelt einen numerischen Wert in einen Wert vom Typ PATTERN um
PCK_BINARY	entpacken gepackter Zahlen aus dem Diagnoseobjekt

Tabelle 14: Übersicht über PRODAMP-Standardfunktionen

Die aufgeführten Namen sollten nicht für eigene Prozeduren verwendet werden, da dies zu Fehlinterpretationen des Programms führen kann.

ADDRESS
Liefert die Adresse eines Moduls oder eines Kontrollblocks

Die Standardfunktion ADDRESS liefert die Adresse eines Moduls. Die Ergebnis-Variable muss daher numerisch sein. Falls die Ergebnis-Variable noch nicht initialisiert ist, bekommt sie den Typ „numerisch“ und die Länge 4. ADDRESS kann auch dazu benutzt werden, um Relativadressen von Feldern in einem Kontrollblock zu ermitteln.

Module eines Benutzerprogramms sind im Allgemeinen nur in Userdumps zu finden.

Funktionsaufruf

Operation	Operanden
ADDRESS	(modname, susyname)

Operandenbeschreibung

modname — gibt den Namen des gesuchten Moduls an. 'modname' darf höchstens 32 Zeichen enthalten, Leerzeichen vor oder nach dem Modulnamen sind aber erlaubt.

> ¡ In Stringtypen wird Groß- und Kleinschreibung unterschieden. *modname* muss also in Großbuchstaben angegeben werden.

susyname — gibt das Subsystem an, in dem das betreffende Modul lokalisiert werden soll.
susyname ist ein String, der entweder den Namen des Subsystems oder eine der folgenden symbolischen Bezeichnungen enthält:

CP	es wird im Control-Program, d.h. unter allen Cl1- und Cl2-Modulen gesucht.
*PRIV	es wird in allen privilegierten Subsystemen gesucht.
*NONPRIV	es wird in allen nichtprivilegierten Subsystemen gesucht.
*USER	es wird nur unter den Benutzer-CSECTs gesucht.
*ALL	es wird überall gesucht.

> ¡ Soll die Relativadresse eines Kontrollblocks ermittelt werden, muss *modname* den Feldnamen und *susyname* die Zeichenfolge „DS“ enthalten. Wird für *modname* der Name eines Symbols angegeben und handelt es sich bei diesem Symbol um das Element einer Substruktur, das über REFERENCE-Aufrufe lokalisiert wurde, so ist für *susyname* die Zeichenfolge „RF“ einzutragen (siehe Beispiel bei der Standardprozedur REFERENCE "Standardprozeduren").

Beispiel

```
A := ADDRESS ('DOPEN', 'CP') ;
MODULE := 'FAUTEM' ;
A := ADDRESS (MODULE, 'ARCHIVE');
A := ADDRESS ('EXVTDSSM','DS');
```

Wird das Modul oder das Symbol nicht gefunden, dann erhält das Pseudo-Symbol CURRENT.ERROR einen Wert î 0. In diesem Fall ist das zurückgelieferte Ergebnis undefiniert.

DEC_BINARY
Dezimalzahl umwandeln

Die Standardfunktion DEC_BINARY interpretiert den Inhalt eines Strings als Dezimalzahl und gibt sie als Datum vom Typ „numerisch der Länge 4“ zurück. Dabei werden Leerzeichen vor und nach den relevanten Zeichen ignoriert.

Falls der String keinen gültigen Dezimalwert enthält oder wenn der Wert außerhalb der Grenzen für numerische Variable liegt, wird CURRENT.ERROR gesetzt. Das Ergebnis von DEC_BINARY ist in diesem Fall undefiniert.

Das Einschalten der vorzeichenlosen Arithmetik mit der Standardprozedur UNSIGNED_ON wird ignoriert.

Funktionsaufruf

Operation	Operanden
DEC_BINARY	(string);

Operandenbeschreibung

string gibt die Stringvariable an, die umgewandelt werden soll.

DEC_STRING
Numerische Werte umwandeln

Die Standardfunktion DEC_STRING wandelt eine angegebene Zahl in einen Dezimalstring um. Das Einschalten der vorzeichenlosen Arithmetik mit der Standardprozedur UNSIGNED_ON wird ignoriert.

Funktionsaufruf

Operation	Operanden
DEC_STRING	(zahl[,länge])

Operandenbeschreibung

zahl gibt die umzuwandelnde Zahl an.

länge gibt die Länge des erzeugten Strings an. Die Dezimalzahl wird in den String rechtsbündig eingetragen und nach links mit Leerzeichen aufgefüllt. Reicht die angegebene Länge nicht aus, werden führende Zeichen abgeschnitten.

Ist keine Länge angegeben, werden nur die signifikanten Zeichen zurückgegeben (kompaktes Format ohne führende Leerzeichen).

Ausnahme:

X := DEC_STRING(I) wirkt wie X := DEC_STRING(I,L), wenn X eine bereits initialisierte Stringvariable der Länge L ist.

Nutzt man DEC_STRING ohne Angabe einer Länge zur Initialisierung einer Variablen oder als Parameter in einem für eine eigene PRODAMP-Unterprozedur zu übergebenden Ausdruck, so werden 10 Bytes für das Ergebnis reserviert. In beiden Fällen ist es besser, eine Länge explizit anzugeben.

Beispiel

```
Auftrag:                                    Ergebnis:
X := 'XXXXXXXX'; ———————————————————————————————————————  (1)
X := DEC_STRING (123);                      '     123'
X := 'XXX';
X := DEC_STRING (123456);                   '456' ——————————————  (2)
X := 'ABC' + DEC_STRING (12) + 'XYZ';       'ABC12XYZ        ' ———  (3)
X := 'ABC' + DEC_STRING (12,5) + 'XYZ';     'ABC   12XYZ' ———————  (4)
```

(1) Initialisierung von X als String der Länge 8;
(2) Initialisierung von X als String der Länge 3;
(3) Keine Initialisierung von X; es ergibt sich ein String der Länge 3 + 10 + 3;
(4) Keine Initialisierung von X, aber der Operand „länge“ bei DEC_STRING erhält den Wert 3 + 5 + 3.

HEX_BINARY
Sedezimalzahl umwandeln

Die Standardfunktion HEX_BINARY interpretiert einen angegebenen String als Sedezimalzahl und reicht das Ergebnis als Datum vom Typ „numerisch der Länge 4“ zurück. Dabei werden Leerzeichen vor und nach den relevanten Zeichen ignoriert.

Falls der String keinen gültigen Sedezimalwert enthält oder wenn der Wert außerhalb der Grenzen für numerische Variable liegt, wird CURRENT.ERROR gesetzt. Das Ergebnis von HEX_BINARY ist in diesem Fall undefiniert.

Funktionsaufruf

Operation	Operanden
HEX_BINARY	(string);

Operandenbeschreibung

string gibt die Stringvariable an, die umgewandelt werden soll.

HEX_STRING
Numerische Werte umwandeln

Die Standardfunktion HEX_STRING wandelt eine angegebene Zahl in einen Sedezimalstring um.

Funktionsaufruf

Operation	Operanden
HEX_STRING	(zahl[,länge])

Operandenbeschreibung

zahl gibt die umzuwandelnde Zahl an.

länge gibt die Länge des erzeugten Strings an. Die Sedezimalzahl wird in den String rechtsbündig eingetragen und nach links mit der Ziffer Null aufgefüllt.
Reicht die angegebene Länge nicht aus, werden führende Zeichen abgeschnitten.

Ist keine Länge angegeben, werden nur die signifikanten Zeichen zurückgegeben (kompaktes Format ohne führende Leerzeichen). Analog zu DEC_STRING gibt es Ausnahmefälle, wenn keine Länge angegeben ist.
Der Compiler reserviert dann gegebenenfalls 8 Stellen für das Ergebnis von HEX_STRING.

Beispiele

```
Auftrag:                                    Ergebnis:
X := 'XXXXXXXX';
X := HEX_STRING (123)                       '0000007B'
X := 'XXX';  "zu kurz"
X := HEX_STRING (4097)                      '001'
X noch nicht initialisiert
X := HEX_STRING (123,8)                     '0000007B'
X noch nicht initialisiert
X := HEX_STRING (123)                       '0000007B'
```

LENGTH
Länge eines Moduls, Kontrollblocks oder Parameterbereichs ausgeben

Die Standardfunktion LENGTH kann in drei Varianten aufgerufen werden:

- Variante 1: Ausgegeben wird die Länge eines Moduls
- Variante 2: Ausgegeben wird die Länge eines Kontrollblocks (DSECT/SPL-Struktur, -Unterstruktur, -Feld)
- Variante 3: Ausgegeben wird die Länge des Parameterbereichs, mit dem die aktuelle Prozedur aufgerufen wurde

Das Ergebnis ist vom Typ „numerisch“ und hat die Länge 4.

Funktionsaufruf

Operation	Operanden
LENGTH	(modname, susyname);

Operandenbeschreibung

modname

Variante 1: „modname“ ist ein Modul-Name, dessen Länge zu bestimmen ist. „modname“ darf höchstens 32 Zeichen enthalten, Leerzeichen vor oder nach dem Modul-Namen sind aber erlaubt.

Variante 2: „modname“ ist der Name eines Kontrollblocks, dessen Länge zu bestimmen ist. Maximal 32 Zeichen, sowie zusätzliche Leerzeichen vor oder nach dem Namen sind erlaubt.

Variante 3: Variante 3: „modname“ ist der String „*PARAMETER“

susyname

Variante 1: Der Name des Subsystems, in dem das Modul gesucht werden soll.

Folgende Werte des Strings „susyname“ haben eine Sonderbedeutung:

CP	Es wird im Control-Program, d.h. unter allen Cl1- und Cl2-Modulen gesucht.
*PRIV	Es wird in allen privilegierten Subsystemen gesucht.
*NONPRIV	Es wird in allen nichtprivilegierten Subsystemen gesucht.
*USER	Es wird nur unter den Benutzer-CSECTs gesucht.
*ALL	Es wird überall gesucht.

Variante 2: „susyname“ enthält „DS“ oder „RF“.
Damit wird angezeigt, dass die Länge eines Kontrollblocks bzw. Feldes zu bestimmen ist. „RF“ ist dann zu nutzen, wenn das Feld Element einer Substruktur ist und zur Lokalisierung REFERENCE Aufrufe benutzt werden.

Variante 3: „susyname“ enthält „DS“ und „modname“ ist der String „*PARAMETER“.

Hinweise

1. Die Aufrufform LENGTH('*PARAMETER','DS') setzt den Wert von CURRENT.ERROR immer auf 0 und liefert die Gesamtlänge des Parameterbereichs zurück, mit dem die aktuelle Prozedur aufgerufen wurde. Diese Länge ist 0, wenn der Aufruf ohne Parameter erfolgte.

   Beispiele zur Anwendung von LENGTH('*PARAMETER','DS') finden Sie ab "Pseudostrukturen".

2. Wird LENGTH für ein Modul oder einen Kontrollblock aufgerufen, so kann durch anschließende Auswertung von CURRENT.ERROR bestimmt werden, ob dieses Modul bzw. der Kontrollblock überhaupt gefunden wurde. Damit ergibt sich die Möglichkeit, vor einem symbolischen Zugriff mit einem Kontrollblock-Feld die Existenz dieses Kontrollblock-Feldes in den aktuell geladenen Symbolelementen zu überprüfen und entsprechend zu reagieren.

Beispiele

```
L:=LENGTH('NCTVXVT','CP');
L:=LENGTH('EXVT','DS');
L:=LENGTH('EXVTPRD','DS');
L:=LENGTH('*PARAMETER','DS');
L:=LENGTH('MYDSECT','DS');
IF CURRENT.ERROR <> 0 THEN
" Fehlerfall. Z.B. mit Nachladen der benoetigten Symbole über "
" COMMAND ('ADD-SYMBOLS ...') fortfahren oder Fehlerausgang   "
END IF;
```

LOCATION
Liefert den Modulnamen zu einer Adresse

Die Standardfunktion LOCATION gibt den Namen des Moduls aus, in dem sich eine angegebene Adresse befindet. Das Ergebnis muss daher einer Stringvariablen zugewiesen werden.

Der Modulname wird in einer maximalen Länge von 32 Zeichen übertragen. Ist die Stringvariable zur Aufnahme des Modulnamens zu kurz, wird abgeschnitten. Ist sie länger als der Modulname, wird der Rest mit Leerzeichen aufgefüllt. Ist die Zielvariable noch nicht initialisiert, bekommt sie den Typ „String“ und die Länge 8.

Entspricht der angegebene Ausdruck keiner Adresse aus einem Modul, setzt LOCATION das Pseudo-Symbol CURRENT.ERROR. Liegt die Adresse im Benutzerspeicher, erhält die Zielvariable den Wert „ABSOLUTE“. Ist das nicht der Fall, erhält sie den Wert „?“.

Die Distanz relativ zum Modulanfang kann anschließend mit der Standardfunktion ADDRESS ermittelt werden.

Funktionsaufruf

Operation	Operanden
LOCATION	(adresse, susyname);

Operandenbeschreibung

adresse — die Adresse des zu lokalisierenden Moduls.

susyname — gibt das Subsystem an, in dem das Modul lokalisiert werden soll.
susyname ist ein String, der entweder den Namen des Subsystems oder eine der folgenden symbolischen Bezeichnungen enthält:

CP	es wird im Control-Program, d.h. unter allen Cl1- und Cl2-Modulen gesucht.
*PRIV	es wird in allen privilegierten Subsystemen gesucht.
*NONPRIV	es wird in allen nichtprivilegierten Subsystemen gesucht.
*USER	es wird nur unter den Benutzer-CSECTs gesucht.
*ALL	es wird überall gesucht.

Beispiel

```
NAM := LOCATION ( ADDR, 'CP' );
REL := ADDR - ADDRESS ( NAM, 'CP' ) ;
```

PATTERN
Umwandlung eines numerischen Wertes

Die Funktion PATTERN wandelt einen numerischen Wert in einen Wert vom Typ PATTERN um. Existiert die Ergebnis-Variable noch nicht, dann erhält sie die Länge 4.

Eine solche Umwandlung wird benötigt, wenn z.B. in einer Schleife Daten dahingehend überprüft werden sollen, ob bestimmte Bits gesetzt sind oder nicht. Da keine Operationen verfügbar sind, die z.B. aus P'01' den Wert P'02' erzeugen, ist dies nur über die Umwandlung von numerischen Werten möglich.

Funktionsaufruf

Operation	Operanden
PATTERN	(num)

Operandenbeschreibung

num gibt den umzuwandelnden numerischen Wert an.

Beispiel

```
L := 1;
U := 1;
WHILE L < 8 DO
   IF PATTERN ( U ) IN .EXVTULM THEN
      .......
      .......
   END IF;
   U := U * 2;
   L := L + 1;
END WHILE;
```

PCK_BINARY
Gepackte Zahlen entpacken

PCK_BINARY formt gepackte zu ungepackten Zahlen um. Das Ergebnis ist numerisch von der Länge 4.

Enthält der String keine gepackte Zahl, wird CURRENT.ERROR gesetzt.

Funktionsaufruf

Operation	Operanden
PCK_BINARY	(string)

Operandenbeschreibung

string gibt die zu entpackende Zahl an.

5.7.4 Mit Prozeduren arbeiten (Spezialfenster PROC)

Jede PRODAMP-Prozedur kann sowohl aus der DAMP-Programmebene heraus als auch als Unterprogramm aufgerufen werden. Aus der DAMP-Programmebene wird die Prozedur dabei entweder nach der Compilierung durch Angeben der Option Mode=Xqt, durch die Option Mode=Go, durch die DAMP-Anweisung RESUME-PRODAMP-PROGRAM mit der Möglichkeit zur Parameter-Übergabe oder durch die DAMP-Anweisung START-PRODAMP-PROGRAM aufgerufen, falls die PRODAMP-Prozedur in Objekt-Form in einer Bibliothek vorliegt. Auch hierbei ist Parameter-Übergabe möglich.

Als Unterprogramm lässt sich eine PRODAMP-Prozedur nur dann aufrufen, wenn sie in Objekt-Form in einer Bibliothek oder bereits übersetzt in einem PROC-Fenster vorliegt. Der Prozeduraufruf erfolgt dann durch Nennung des Prozedur-Namens sowie ggf. der Liste der vorhandenen Parameter. Der Prozedur-Name ist identisch mit dem Elementnamen des PRODAMP-Objekts in der Bibliothek (siehe Abschnitt „Private Prozeduren archivieren").

Private Prozeduren erstellen

Das Erstellen von PRODAMP-Prozeduren besteht aus den Komponenten Editieren und Compilieren.

Wollen Sie eine Prozedur erstellen, müssen Sie zunächst einem der Dumpfenster den PRODAMP-Compiler zuweisen. Das geschieht mit der Anweisung

```
START-PRODAMP-EDITOR WINDOW=<w>, SOURCE=filename
```

WINDOW	bestimmt dabei das Fenster (4 bis 9 bzw. 21 bis 99), in dem die PRODAMP-Prozedur editiert, compiliert und gegebenenfalls ausgeführt werden soll;
SOURCE	gibt eine Datei an, die bereits eine PRODAMP-Quelldatei enthält. Diese wird sofort in das angegebene Fenster eingelesen. Diese Angabe ist optional.

> ¡ Ein mit einer PRODAMP-Prozedur belegtes Fenster wird beim Wechsel der Dumpdatei durch die DAMP-Anweisung OPEN-DIAGNOSIS-OBJECT nicht zurückgesetzt.

Eingabefelder im PROC-Fenster

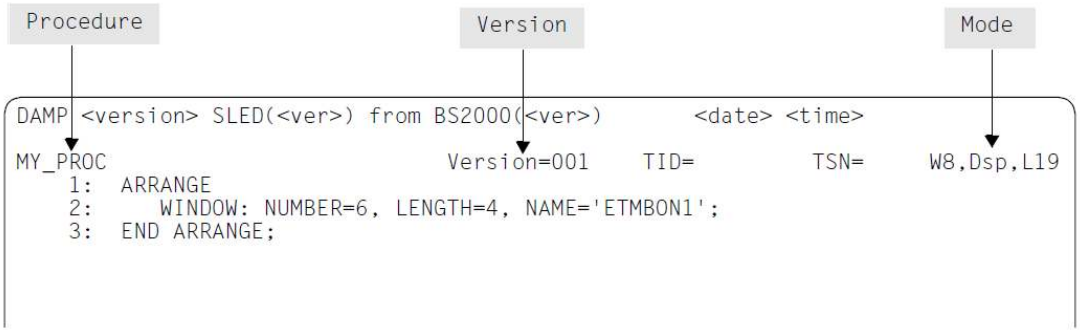


Bild 67: Dumpfenster mit zugewiesenem PRODAMP-Compiler

In einer eingelesenen oder geschriebenen Prozedur können Sie wie im EDT hin- und herblättern oder auf eine Zeile positionieren (+nn, -nn, #nn). Ferner sind Eingaben sowohl in der Kopfzeile als auch in den Textzeilen möglich. Durch Eingaben in der Kopfzeile (Procedure, Version und Mode) werden PRODAMP-Funktionen angestoßen, während Eingaben in den Programmzeilen als Änderungen dieser Zeilen angesehen werden.

Bedeutung der drei Felder in der Kopfzeile:

Mode gibt an, was im Dumpfenster ausgegeben wird oder welche Aktion der PRODAMP-Compiler gerade durchführt bzw. durchführen soll. Eintragen einer der nachfolgend aufgeführten Zeichenfolgen und übertragen mit der Taste DUE löst die zugehörige Aktion aus.

=Beg (begin); positioniert das Fenster auf Zeile 1 der Quelldatei.

=Cmp (compile); startet den Compiler für den aktuellen Inhalt des Fensters.

=Dsp (display); Standardeinstellung zum Ändern von Quelldatei-Zeilen innerhalb des PRODAMP-Fensters.

=Edt (edit); übernimmt die aktuelle PRODAMP-Prozedur in einen EDT-Bereich. Größere Änderungen an der PRODAMP-Quelldatei sollten immer im EDT vorgenommen werden. Im PRODAMP-Fenster sind lediglich einfache Editier-Funktionen (blättern, überschreiben, löschen) möglich. Nach Beendigung des Editors mit END, HALT oder RET wird die editierte Quelldatei in das PRODAMP-Fenster übernommen. Der EDT-Bereich wird anschließend gelöscht.

=Go bewirkt „compile + execute + begin“.

=Inf (Inform); Es wird ein Inhaltsverzeichnis der in der eingestellten Source-Bibliothek vorhandenen Source-Elemente ausgegeben und eine durch Lck (Lock) gesetzte Sperre aufgehoben. Falls in das Feld „Procedure“ das Schlüsselwort OBJECT eingetragen wird, schaltet die Anzeige auf das Inhaltsverzeichnis der Objekt-Elemente um, umgekehrt erhält man durch Eintrag von SOURCE wieder die ursprüngliche Darstellung.

=Lck (Lock); wirkt wie Read, jedoch wird zusätzlich das gelesene Element in der Source-Bibliothek für konkurrierende Aufrufe gesperrt. Die Sperre wird wieder durch Update, New oder Inform aufgehoben. New und Inform verändern dabei den ursprünglichen Inhalt der Bibliothek nicht. Nur Update überschreibt die alten Daten.

=New löscht das PRODAMP-Fenster zur Aufnahme einer neuen PRODAMPProzedur.
Eine durch Lck (Lock) gesetzte Sperre wird ohne Rückfrage aufgehoben.

> Das PRODAMP-Fenster wird ohne Rückfrage gelöscht.

=Rea (Read); Aus der eingestellten Source-Bibliothek wird ein Element gelesen. Elementname und Version werden wie bei M=Wrt aus den Feldern „Procedure“ und „Version“ der Kopfzeile entnommen. Wird keine Version spezifiziert, greift DAMP standardmäßig auf die höchste Version zu.

=Sav (Save); Speichert ein PRODAMP-Objekt in der eingestellten Objektbibliothek ab. Als Elementname wird der im Feld Procedure gegebene Name, als Version die im Feld Version angegebene Zahl verwendet. Ein Element gleichen Namens und gleicher Version wird ohne Rückfrage überschrieben.

=Upd (Update); In der eingestellten Source-Bibliothek wird das Element mit dem im Feld Procedure angegebenen Namen und der im Feld Version spezifizierten Version überschrieben. Eine durch Lck (Lock) gesetzte Sperre wird aufgehoben. Falls das Element mit dieser Version noch nicht existierte, wird eine Fehlermeldung ausgegeben. In diesem Fall muss mit Mode=Wrt gearbeitet werden.

=Wrt (Write); Der aktuelle Source wird in die mit ASSIGN-PRODAMPLIBRARIES eingestellte bzw. in die Default-Bibliothek geschrieben. Als Elementname wird der im Feld „Procedure“ angegebene Name, als Version die im Feld Version angegebene Versionsnummer verwendet. Wird keine Version spezifiziert, setzt DAMP standardmäßig 001 ein. Ein bereits existierendes Element gleichen Namens und gleicher Version wird nicht überschrieben. Wird Überschreiben gewünscht, muss Mode=Upd verwendet werden.

=Xqt (execute); führt eine compilierte PRODAMP-Prozedur aus.

Procedure zeigt den Namen der Prozedur an, die gerade ausgegeben wird.

Version gibt die Version der Prozedur mit dem angegebenen Namen an.

Tipps und Hinweise

Für das erste Experimentieren mit PRODAMP empfiehlt sich folgendes Vorgehen:

- Starten von PRODAMP mit START-PRODAMP-EDITOR
- Wechsel in den EDT über Mode=Edt
- Edieren der PRODAMP-Prozedur im EDT
- Evtl. Sichern der Prozedur mit WRITE
- Rückkehr nach DAMP mit HALT
- Compilieren der Prozedur mit Mode=Cmp
- Evtl. Korrektur von Syntaxfehlern direkt im PROC-Fenster oder wieder im EDT
- Ausführen der syntaktisch fehlerfreien Prozedur mit Mode=Xqt
- Debuggen der Prozedur (falls erforderlich)

Häufige Fehler

- Ein automatisch lokalisierbares Symbol wird ohne den zu einem Symbol gehörenden Punkt geschrieben. PRODAMP erkennt dann kein Symbol, sondern eine Variable.
- Bei Verwendung von Symbolen wird nicht bedacht, dass Typ und Länge der Symbole erst beim Ablaufzeitpunkt bekannt sind. Wenn eine nicht initialisierte Variable zugewiesen wird, kann sie demnach nur den Default-Typ (numerisch mit Länge 4) bekommen. Sie sollten daher immer alle Variablen durch Initialisieren deklarieren.
- Sie haben versäumt, nach einem Zugriff zum Diagnoseobjekt das Pseudo-Symbol CURRENT.ERROR abzufragen und arbeiten mit unsinnigen Werten weiter.
- Sie haben die Regeln für das Abschneiden bzw. Auffüllen beim Zuweisen von unterschiedlich langen String-Variablen nicht beachtet.
- Beim Wechseln der Objekt-Bibliothek werden Objekte, die aus einer früheren Objekt-Bibliothek geladen sind, im Speicher nicht gelöscht. Ist also eine Prozedur namens PROC mit der Anweisung `//START-PRODAMP-PROGRAM` aus einer früheren Objekt-Bibliothek geladen worden, dann wird nach dem Wechsel der Objekt-Bibliothek mit START-PRODAMP-PROGRAM PROC noch immer die alte Prozedur aufgerufen, auch wenn in der neuen Bibliothek eine andere Prozedur mit dem gleichen Namen vorhanden ist.

Adressbereinigung

PRODAMP führt beim Adressierungsmodus 31 während der Ausführung in folgenden Fällen Adressbereinigungen durch:

- wenn eine numerische Variable als Basis für einen symbolischen Zugriff genutzt wird.

Beispiel

```
P := X'82CD0000';
A := P.ESTKGR0;
```

Hier wird im zweiten Statement zur Bildung der Adresse je nach Adressierungsmodus nur X'02CD0000' bzw. X'82CD0000' hergenommen.

- wenn die numerische Variable als Eingabe für die Standardfunktion LOCATION benutzt wird.

> **i** Der Adressierungsmodus ist für DAMP eine globale Konstante, die vom HSI abhängt. Bei /390-Servern verwendet DAMP 31-Bit-Adressierung. Bei x86-Servern verwendet DAMP 32-Bit-Adressierung. Insbesondere bei der Anwendung von PRODAMP-Prozeduren auf Benutzerprogramme kann es daher nötig sein, dort aufgefundene Adressen mit dem Operator MOD „von Hand“ auf 24-Bit zu bereinigen.
>
> *Beispiel*
>
> `X'887C240C' MOD X'01000000'` ergibt `X'007C240C'`.

Parameter-Übergabe an PRODAMP

Parameter können Sie auf vier Wegen übergeben:

- mit RESUME-PRODAMP-PROGRAM
- mit START-PRODAMP-PROGRAM
- durch Eintragen in das Prozedurfenster, anschließendes Compilieren und Starten. Dazu schreiben Sie Ihre Prozedur so, dass die Variablen, die die Parameter aufnehmen sollen, in den ersten Zeilen der Prozedur stehen. Arrangieren Sie nun mit der Anweisung MODIFY-SCREEN-LAYOUT die Fenster so, dass vom PRODAMP-Fenster nur die obersten Zeilen (mit den Parameter-Variablen) sichtbar bleiben. Wenn Sie nun im PRODAMP-Fenster die Werte für die Parameter mit den aktuellen Werten überschreiben und die Option „Go“ (Compile and Go) auswählen, wird die Prozedur mit den aktuellen Werten neu compiliert und ausgeführt.
- mit der PRODAMP-Standard-Prozedur READ_WINDOW

Auswirkungen auf die Listenausgabe

Alle Seiten, die während des Ablaufs einer PRODAMP-Prozedur im Diagnoseobjekt angesprochen werden, gelten für eine nachfolgende Listenausgabe als referenzierte Seiten (ähnlich wie auch die im DAMP-Dialog angesprochenen Seiten). Sie werden also bei der Minimum-Listenausgabe automatisch mit ausgegeben. Man kann diesen Effekt dazu ausnutzen, gezielt alle die Seiten in PRODAMP anzusprechen, die man unbedingt ausgedruckt haben möchte.

Verwendung des EDT als „Fensterersatz“

Mit den PRODAMP-Anweisungen WRITE und READ, die ein Schreiben in bzw. Lesen aus einem EDT-Bereich ermöglichen, lassen sich auf Umwegen auch quasiformatierte Dialogausgaben programmieren. Dazu sind lediglich folgende PRODAMP-Anweisungen erforderlich:

```
WRITE ('Nutzinformation');
....
WRITE ('Nutzinformation');
WRITE ('@COL 80 O & C'' ''');    'Kürzen aller Zeilen auf max 80'
WRITE ('@PRO9');                 'Umschalten auf Prozedurbereich 9'
WRITE ('@DEL');                  'Löschen'
WRITE ('@@PRINT 1-.$VN');        'Eintragen der PRINT-Anweisung'
WRITE ('@END');                  'Zurückschalten des Prozedurbereiches'
WRITE ('@DO9');                  'Ausgabe der Zeilen auf dem Bildschirm'
```

Das Kürzen der Zeilen ist erforderlich, weil die Anweisung WRITE den zu schreibenden PRODAMP-String bis zur maximalen Länge (133) mit Leerzeichen auffüllt.

Beispiel: Eine geführte Eingabe

```
ABC := ' '*4; XYZ := ' '*10;
WRITE ('@PRO9');
WRITE ('@DEL');
WRITE ( '@@CREATE1READ''BITTE ABC EINGEBEN''' );
WRITE ('@END');
WRITE ('@DO9');
READ ( ABC );
WRITE ( '@DEL' );
WRITE ('@PRO9');
WRITE ('@DEL');
WRITE ( '@@CREATE1READ''BITTE XYZ EINGEBEN''' );
WRITE ('@END');
WRITE ('@DO9');
READ ( XYZ );
WRITE ( '@DEL' );
```

Die WRITE '@D ' sind dabei erforderlich, damit die Anweisung READ wieder beim ersten Satz des EDT-Bereichs aufsetzt.
Die oben dargestellte Anweisungssequenz lässt sich verkürzt folgendermaßen formulieren:

```
ABC := ' '*4; XYZ := ' '*10;
WRITE ('@PRO9');
WRITE ('@D');
WRITE ( '@@CREATE1READ''BITTE ABC EINGEBEN''' );
WRITE ( '@@CREATE2READ''BITTE XYZ EINGEBEN''' );
WRITE ('@END');
WRITE ('@DO9');
READ ( ABC );
READ ( XYZ );
WRITE ( '@D' );
```

Die Sätze werden mit READ immer in der tatsächlichen Länge des EDT-Records eingelesen. Die restlichen Zeichen der String-Variablen bleiben unverändert.

Ferner sollten die String-Variablen generell initialisiert werden, weil andernfalls die Standardlänge 133 eingesetzt wird.

Listenausgabe mit der Anweisung COMMAND

Die mit COMMAND eingegebenen DAMP-Anweisungen müssen so formuliert sein, als seien sie im Batch- bzw. Prozedurbetrieb eingegeben worden. Das wirkt sich insbesondere auf das Ausdrucken von Listen aus. Die dazu erforderlichen Angaben müssen Sie daher mit der Anweisung ADD-LIST-OBJECTS machen.

Beispiel

```
COMMAND ('START-LIST-GENERATION');         "Umschalten in den LIST-Modus"
COMMAND ('ADD-LIST-OBJECTS GLOBAL=OVERVIEW');
COMMAND ('ADD-LIST-OBJECTS TASK=(C''UCON'')');
COMMAND ('PRINT-LIST');
```

Private Prozeduren aufrufen

Ein in der eingestellten Objekt-Bibliothek abgelegtes PRODAMP-Objekt kann durch die DAMP-Anweisung

`START-PRODAMP-PROGRAM procname, PAR=(par1, par2, ...)`

direkt gestartet werden. Dazu ist keine Kenntnis des Sources oder Zuweisen eines PRODAMP-Fensters erforderlich. Dabei ist „procname“ der Elementname des auszuführenden Objekts. Es wird immer die höchste vorhandene Version des Objekts ausgeführt. Die möglichen Parameter können numerisch, dezimal oder sedezimal, bzw. als String (in Hochkommata) angegeben werden. Reihenfolge und Typ der Parameter muss dem in der Prozedur definierten Parameter-Bereich entsprechen.

Beispiel

```
ARRANGE
.P1 : RELATIVE=0,LENGTH=4,TYPE=NUMERIC;
.P2 : RELATIVE=4,LENGTH=1,TYPE=STRING;
.P3 : RELATIVE=5,LENGTH=4,TYPE=NUMERIC;
.P4 : RELATIVE=9,LENGTH=9,TYPE=STRING;
END ARRANGE;
N := PARAMETER.P1;
IF 'X' = PARAMETER.P2 THEN
....
END IF;
```

Eine Prozedur, die die oben aufgeführten Parameter verwendet, ist demnach etwa mit

`START-PRODAMP-PROGRAM procnam, PAR = (1234,'Z ',X 'AEFF','ABCDEFGHI')`

aufzurufen, wobei dann dem Parameter P1 der Wert 1234, P2 der Buchstabe „Z“, P3 die Sedezimalzahl X'AEFF' und P4 der String „ABCDEFGHI“ zugewiesen wird.

Die Parameter müssen in der Anweisung ARRANGE fortlaufend angegeben und dürfen nicht ausgerichtet werden. Dabei werden alle in der Anweisung START-PRODAMP-PROGRAM angegebenen numerischen Parameter (dezimal oder sedezimal) rechtsbündig auf ein 4 Byte langes Feld abgebildet.

> Numerische Werte lassen sich prinzipiell im Nachhinein auch als Bitmuster oder als Sedezimalstring deuten.
>
> Bei der Abarbeitung einer DAMP-Anweisung wird der ganze Eingabestring in Großbuchstaben umgesetzt. Das gilt auch für die Parameter.

Private Prozeduren unterbrechen

Mit den Anweisungen INTERRUPT (siehe "Anweisungen") und RETURN (siehe "Anweisungen") können Sie die Prozedurbearbeitung unterbrechen.

Bei beiden Anweisungen können Sie ein Fenster angeben, das bei der Unterbrechung in der DAMP-Bildschirmmaske erscheinen soll. Wird kein Fenster angegeben, erscheint das aktuelle Fenster.

Mit RESUME-PRODAMP-PROGRAM wird die unterbrochene Prozedur an der Unterbrechungsstelle fortgesetzt. War keine Prozedur in Aktion, wird die im PRODAMP-Fenster geladene Prozedur von Anfang an gestartet.

Ablauffehler erkennen und entfernen

Erkennt der PRODAMP-Interpreter während des Prozedurlaufs einen Fehler, so bricht er die PRODAMP-Prozedur ab und gibt zwei Meldungen in den Meldungszeilen des DAMP-Bildschirms aus. Die erste Meldung enthält den Namen der abgebrochenen PRODAMP-Prozedur und die Nummer der Prozedurzeile, in der der Fehler aufgetreten ist. Die zweite Meldung beschreibt den Fehler.

> Durch Drücken der Taste K2 und anschließendes Eingeben der Anweisung`/INFORM-PROGRAM MSG='*CANCEL'` lässt sich ein Laufzeitfehler provozieren. Damit können Sie z.B. Endlos-Schleifen abbrechen und erhalten ebenfalls eine Meldung mit dem Namen der Prozedur und der Fehlerzeile.

Für die Fehlerdiagnose bietet PRODAMP zwei Verfahren an, die Ablaufverfolgung (siehe "Anweisungen") und das Überwachen von Variablen (siehe "Anweisungen").

Private Prozeduren archivieren

PRODAMP-Sources und -Objekte können in PLAM-Bibliotheken als Element-Typ S bzw. Typ C abgelegt und von dort wieder geladen werden. Module, die über die Prozedur ENTER_MODULE angesprochen werden, werden als Element-Typ R erwartet.
Wenn nichts anderes angegeben, wird für alle Typen die gleiche Bibliothek mit dem Namen SYS.USRDMP.DAMP.<ver> verwendet.
Die Element-Typen C und R müssen in der gleichen Bibliothek enthalten sein.

Mit der DAMP-Anweisung ASSIGN-PRODAMP-LIBRARIES können Sie eigene Bibliotheken zuweisen, und zwar mit

```
ASSIGN-PRODAMP-LIBRARIES [SOURCE=source-lib] [,OBJECT=object-lib]
```
oder
```
ASSIGN-PRODAMP-LIBRARIES SOURCE=liname, OBJECT=*SOURCE
```

Die genannten Bibliotheken werden anschließend für den gewünschten Typ zugewiesen (bei OBJECT=*SOURCE wird die Bibliothek für alle Typen zugewiesen). Zur Umschaltung auf die Standard-Bibliothek kann statt SYS.USRDMP.DAMP.<ver> auch *STD angegeben werden. Hiermit wird auf die Bibliothek umgeschaltet, die in den Benutzeroptionen eingestellt ist (siehe Abschnitt „Benutzeroptionen einstellen“). Die aktuellen Einstellungen für die PRODAMP-Bibliotheken können mit der Anweisung SHOW-PRODAMP-LIBRARIES ausgegeben werden.

PRODAMP-Sources werden ausschließlich über das PRODAMP-Fenster abgelegt und geladen, und zwar durch Eintragen eines Befehlskürzels in das Feld Mode der Kopfzeile (siehe "Mit Prozeduren arbeiten (Spezialfenster PROC)").

Mode=Wrt	(Write); Der aktuelle Source wird in die mit ASSIGN-PRODAMP-LIBRARIES eingestellte bzw. in die Default-Bibliothek geschrieben.
Mode=Rea	(Read); Aus der eingestellten Source-Bibliothek wird ein Element gelesen.
Mode=Upd	(Update); Das angegebene Element wird überschrieben.
Mode=Inf	(Inform); Das Inhaltsverzeichnis der angegebenen Source-Bibliothek wird ausgegeben.

PRODAMP-Objekte können über das PRODAMP-Fenster erzeugt werden, aber nur über die DAMP-Anweisung START-PRODAMP-PROGRAM geladen (und gleichzeitig gestartet) werden.

Im PRODAMP-Fenster dient zum Abspeichern eines Objekts (nach erfolgreicher Compilierung) das Befehlskürzel Sav im Feld Mode:

Mode=Sav	(Save); Speichert ein PRODAMP-Objekt in der eingestellten Objektbibliothek ab.

Beispiele

Bei den folgenden Beispielen handelt es sich um konkrete Anwendungen der Diagnose-Sprache, die jeweils einen Aspekt besonders beleuchten.

Beispiel 1: HEX-Rechner

Dieses sehr einfache Beispiel realisiert eine Rechenfunktion mithilfe von PRODAMP. Mit der Anweisung MODFIY-SCREEN-LAYOUT sollte vorher dafür gesorgt werden, dass die erste Zeile des PRODAMP-Fensters am Bildschirm sichtbar ist. Durch Änderung des Ausdrucks und die Option „Go“ in der Kopfzeile erreicht man, dass das „Rechenergebnis“ sedezimal und dezimal in der 2. Zeile des DAMP-Bildschirms angezeigt wird. Zugriffe auf das Diagnoseobjekt erfolgen in diesem Beispiel nicht.

```
A := X'14' + X'3B' * 24 ; "HIER DIE GEWUENSCHTE RECHNUNG EINTRAGEN"
MESSAGE ('ERGEBNIS HEX: '+HEX_STRING(A)+', DEZ: '+DEC_STRING(A) );
```

Beispiel 2: Durchsuchen der TFT-Kette der aktuellen Task

Die nachfolgend aufgeführte Prozedur durchsucht die TFT-Kette der aktuellen Task. Für jede TFT wird auf Fenster 4 eine Ausgabe der TFT im Format der TFT-DSECT veranlasst. Mit RESUME wird zur nächsten TFT „geblättert“. Es können aber auch andere DAMP-Anweisungen gegeben werden.

```
IF CURRENT.TID = 0 THEN
  MESSAGE ( 'No TID/TSN given' );
  RETURN;
END IF;
TFT := .ETCBTFT; ──────────────────────────────────────────────────────── (1)
IF TFT = 0 THEN
  MESSAGE ( 'No TFT found' );
  RETURN;
END IF;
RET_WND := CURRENT.WNDNO; ─────────────────────────────────────────────── (2)
```

```
ARRANGE
  WINDOW: NUMBER=4,DSECT='IDMTFT',NAME='IDMFRLNK'; ———————————————— (3)
END ARRANGE;
WHILE TFT <> 0 DO
  ARRANGE
    WINDOW: NUMBER =4,ADDRESS = TFT; ———————————————————————————— (4)
  END ARRANGE;
  INTERRUPT; ———————————————————————————————————————————————————— (5)
  TFT := TFT.IDMFRLNK;
END WHILE;
RETURN WINDOW = RET_WND; ———————————————————————————————————————— (6)
```

(1) Der Anker der TFT-Kette steht in ETCBTFT. Da der TCB von DAMP automatisch lokalisierbar ist, ist die Angabe einer Basis-Adresse nicht notwendig. Es wird die Task genommen, die im PRODAMP-Fenster bei TID bzw. TSN angegeben wurde.

(2) Die Nummer des aktuellen Fensters (meist das PRODAMP-Fenster) wird sichergestellt, um später dieses Fenster wieder als oberstes einstellen zu können.

(3) Für das Fenster 4 werden die Einstellungen, die sich während des Ablaufs nicht ändern, außerhalb der Schleife vereinbart. Der NAME wird angegeben, weil die TFT-DSECT mit einem EQU * beginnt, was zu einem Zerreißen des ersten Feldes führen würde.

(4) Innerhalb der Schleife wird lediglich die Adresse für das Fenster neu vereinbart. Die übrigen Einstellungen (auch die Nummer) bleiben aus der ersten Anweisung ARRANGE erhalten. Auf eine Abfrage von CURRENT.ERROR wurde hier verzichtet, da ein nicht allokierter Speicherbereich bei Ausgabe des Fensters automatisch zu der Fehlermeldung „Requested memory area not accessible“ führt. Als Default wird die aktuelle Task (d.h. die Task, die im PRODAMP-Fenster eingestellt wurde) gezeigt. Man könnte als redundante Information noch TID=CURRENT.TID spezifizieren.

(5) INTERRUPT bewirkt die Unterbrechung der Prozedur und die Anzeige der aktuellen Fenster am Bildschirm. Auf Grund der Anweisung ARRANGE ist dabei das Fenster 4 das oberste Fenster. Die Prozedur kann vom Anwender dann mit RESUME fortgesetzt werden, was zur Anzeige der nächsten TFT führt.

(6) Nach Ausgabe der letzten TFT wird zum vorher gesicherten Fenster zurückverzweigt.

Beispiel 3: Aktuelle Task ändern

Wie man vermeidet, bei der SLED-Analyse mit DAMP im Statusfenster „endlos“ zu blättern, bis man die Task mit einer bestimmten Eigenschaft gefunden hat, zeigt folgendes Beispiel: Es veranschaulicht, wie man mithilfe von PRODAMP nach Tasks sucht, die einen Systemdump erzeugt haben und diese nacheinander zum jeweils aktuellen Task macht. D.h., nach Ablauf der Prozedur wird das DAMP-Statusfenster (2) so positioniert, dass die PCB-Kette der ausgewählten Task4 angezeigt wird.

```
TASK := 0;
WHILE CURRENT.ERROR = 0 DO ———————————————————————————————————  (1)
NEW_TASK ( TASK );
  IF CURRENT.ERROR=0 THEN
    IF .ETCBCDSY <> 0 THEN ——————————————————————————————————  (2)
      ARRANGE WINDOW: NUMBER = 2,TID=TASK; END ARRANGE; ——————————  (3)
      INTERRUPT ;
    END IF;
  END IF;
TASK := TASK + 1; ————————————————————————————————————————————  (4)
END WHILE;
```

(1) Da die Prozedur NEW_TASK CURRENT.ERROR setzt, wenn keine weitere Task mehr gefunden werden kann, ist dies das Abbruchkriterium für die Schleife über alle aktiven Tasks.

(2) Das Feld ETCBCDSY enthält die Anzahl der Systemdump-Anforderungen für diese Task. Dieses Feld ist als TCB-Feld durch DAMP automatisch lokalisierbar, wobei jeweils der TCB der aktuellen Task angesprochen wird. Dieser wurde mit NEW_TASK richtig eingestellt.

(3) Durch ein ARRANGE für Fenster 2 mit Angabe der TID wird das Fenster auf den Eintrag für diese Task positioniert.

(4) Für den Scan muss die TID um eins erhöht werden. NEW_TASK liefert dann die nächste aktive Task.

Beispiel 4: Speicherbereiche auf SYSLST ausgeben

Dieses Beispiel soll veranschaulichen, wie es mit den Standardprozeduren DUMP_MEMORY und SET_HEADER möglich ist, beliebige Speicherbereiche auf SYSLST auszugeben:

```
TFT@ := .ETCBTFT;
WHILE TFT@ <> 0 DO
  P2FCB@ := TFT@.IDMP2FL;
  IF P2FCB@ <> 0 THEN
    SET_HEADER ( '*** P2-FCB FUER FILE '+TFT@.IDMFILE+' ****', 2, 10);
    DUMP_MEMORY ( P2FCB@, 0, LENGTH( 'ID2FCB','DS' ) );
  END IF;
  TFT@ := TFT@.IDMFRLNK;
END WHILE;
```

Diese Prozedur gibt für alle offenen Dateien den P2-FCB auf Liste aus.

5.7.5 Syntax-Diagramme

Mithilfe der Syntax-Diagramme lassen sich alle zulässigen PRODAMP-Konstruktionen ableiten. Umgekehrt sind nicht alle ableitbaren Konstruktionen zulässig, denn es sind zusätzlich noch die Typ-Verträglichkeit und etwaige Einschränkungen bezüglich der Vergabe von Namen zu berücksichtigen. Dies sind allerdings auch keine syntaktischen Eigenschaften im engeren Sinn, denn ein auf Grund etwa von Typ-Unverträglichkeit illegaler Ausdruck kann durch andere Wahl der verwendeten Namen in einen legalen Ausdruck verwandelt werden.

Um eine zu umfangreiche Darstellung der Diagramme zu vermeiden, gilt Folgendes: Verbindungslinien zwischen den Kartuschen repräsentieren ein Trennzeichen (siehe "Lexikalische Elemente"). Nur vor und nach einem Sonderzeichen darf das Trennzeichen weggelassen werden. In Diagrammen, deren Kopf mit Doppel-Linien umrahmt ist, dürfen keine Trennzeichen gesetzt werden.

Der Einstieg in die Syntaxdiagramme erfolgt über den Begriff „PRODAMP-Prozedur“. Anschließend sind alle zur Definition der „PRODAMP-Prozedur“ benutzten Begriffe alphabetisch aufgeführt.

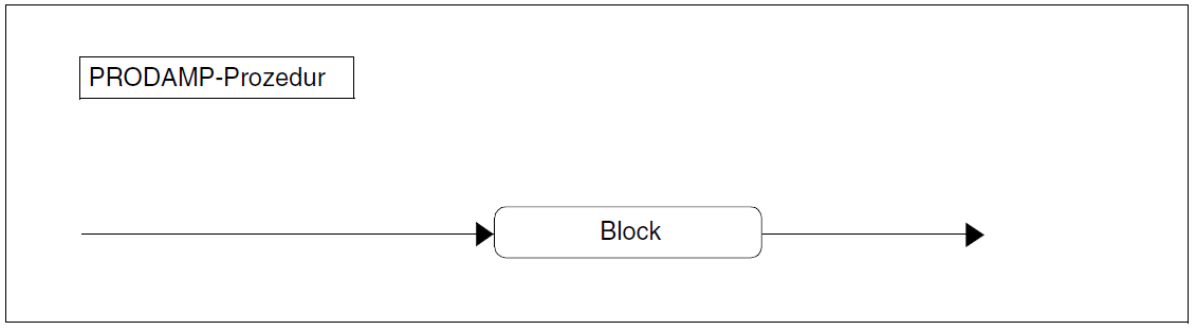


Bild 68: PRODAMP-Prozedur

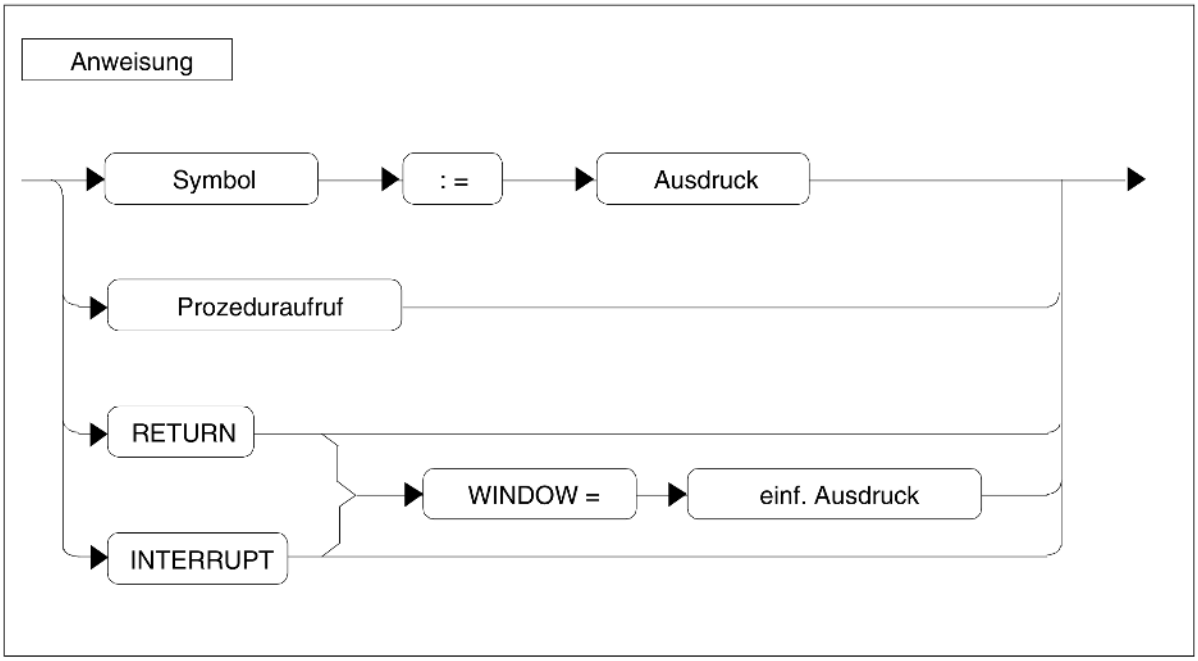


Bild 69: Anweisung

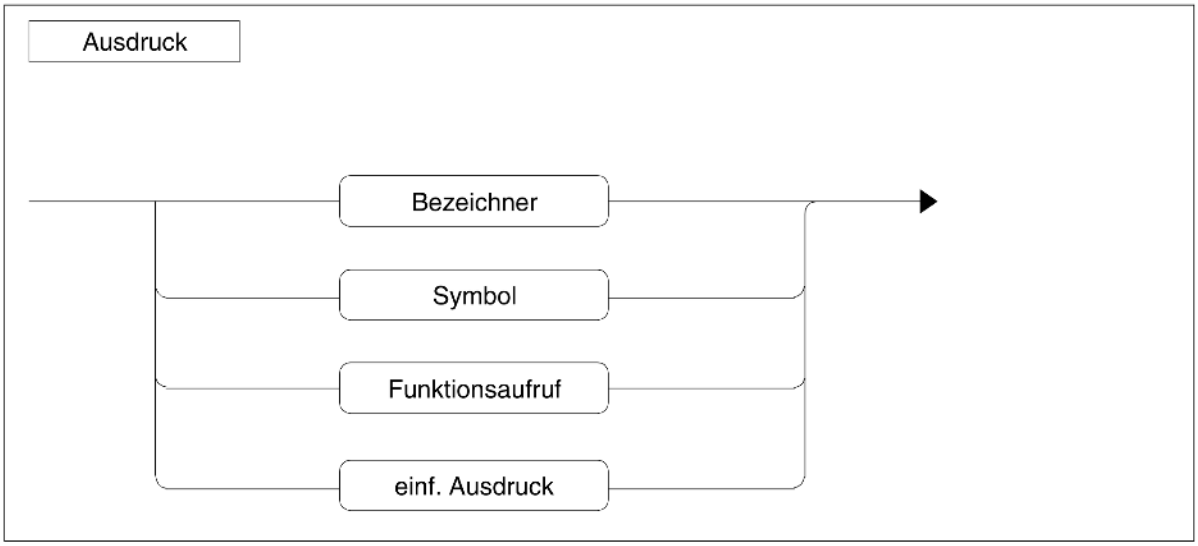


Bild 70: Ausdruck

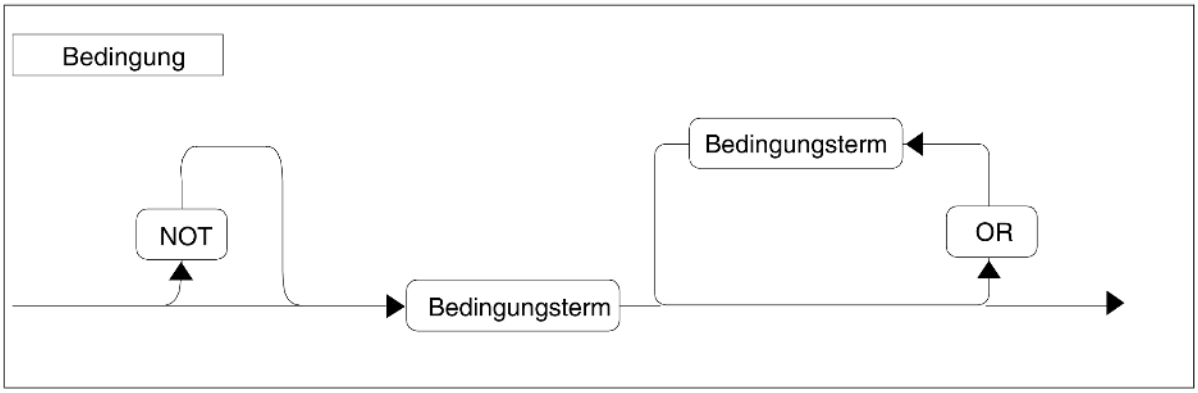


Bild 71: Bedingung

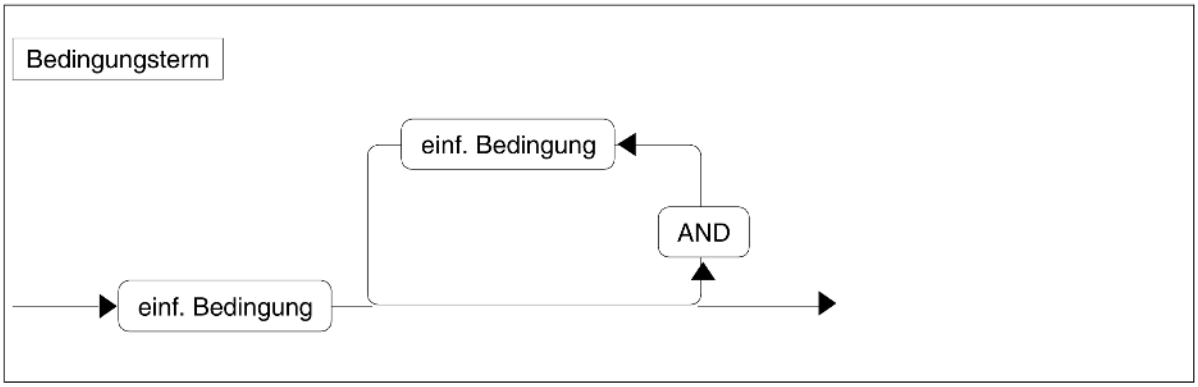


Bild 72: Bedingungsterm

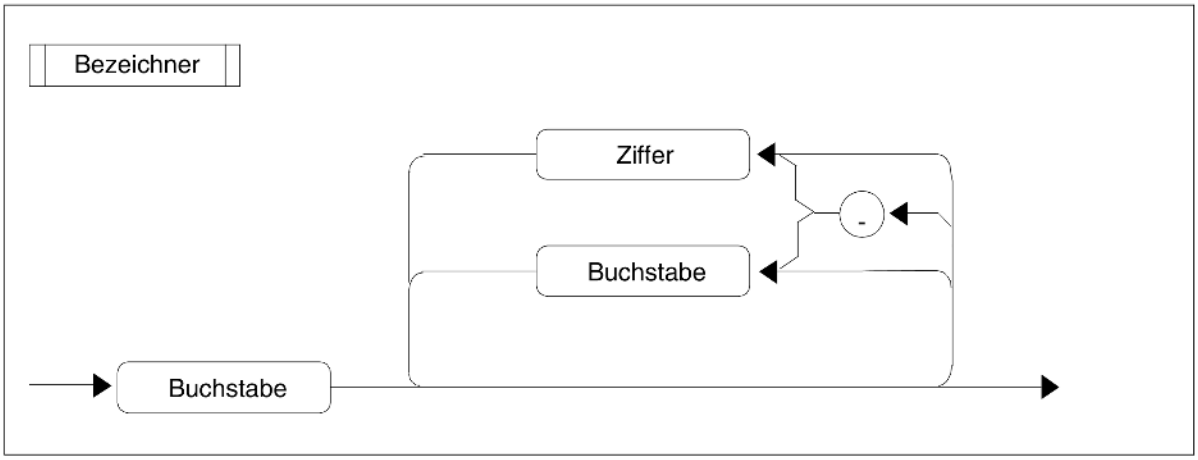


Bild 73: Bezeichner

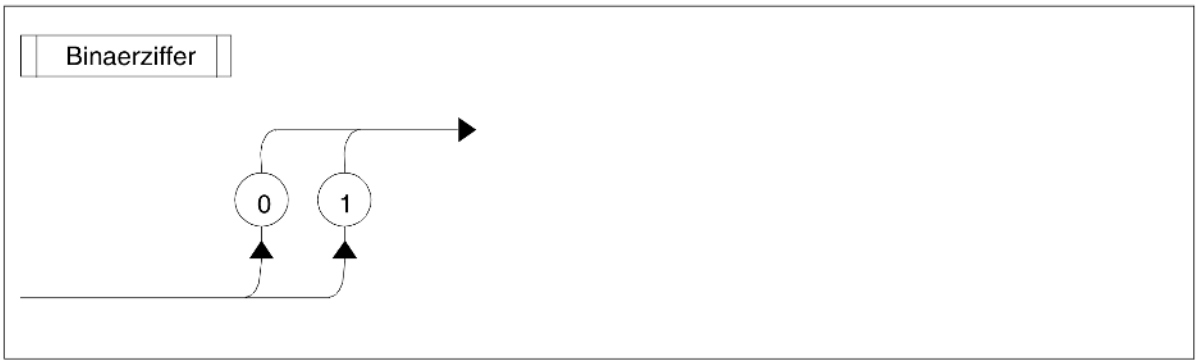


Bild 74: Binärziffer

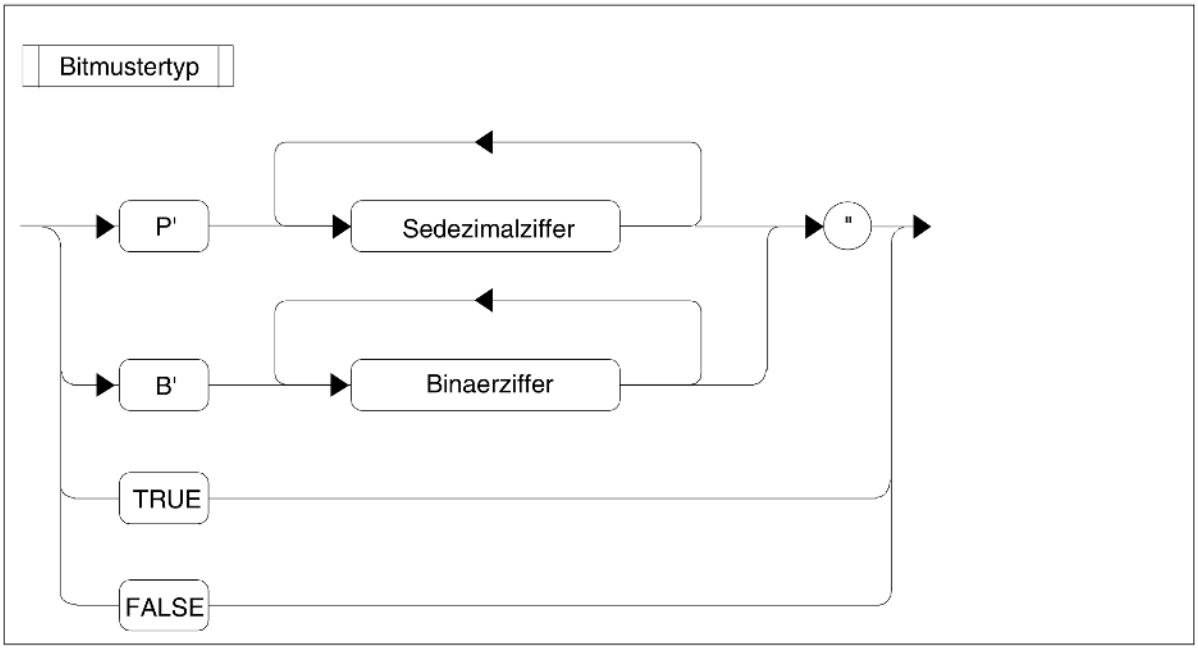


Bild 75: Bitmustertyp

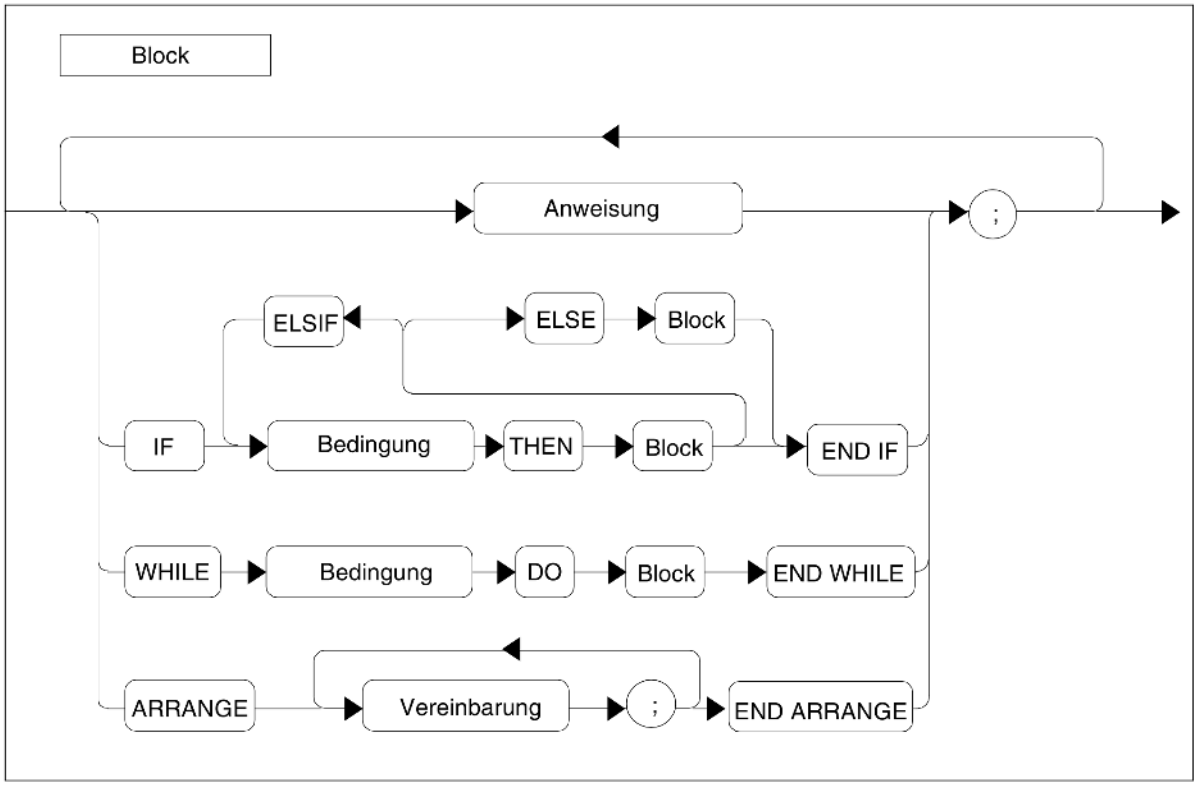


Bild 76: Block

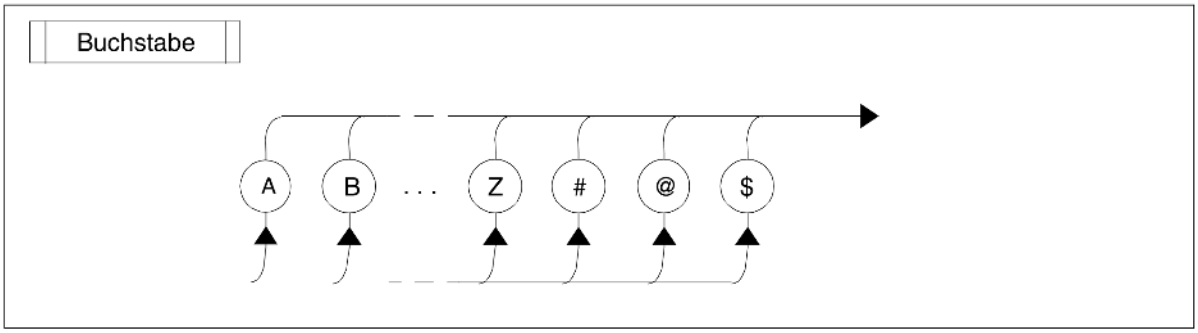


Bild 77: Buchstabe

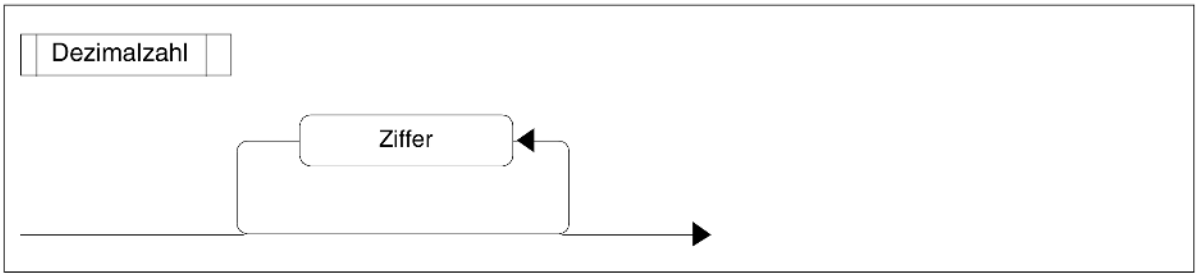


Bild 78: Dezimalzahl

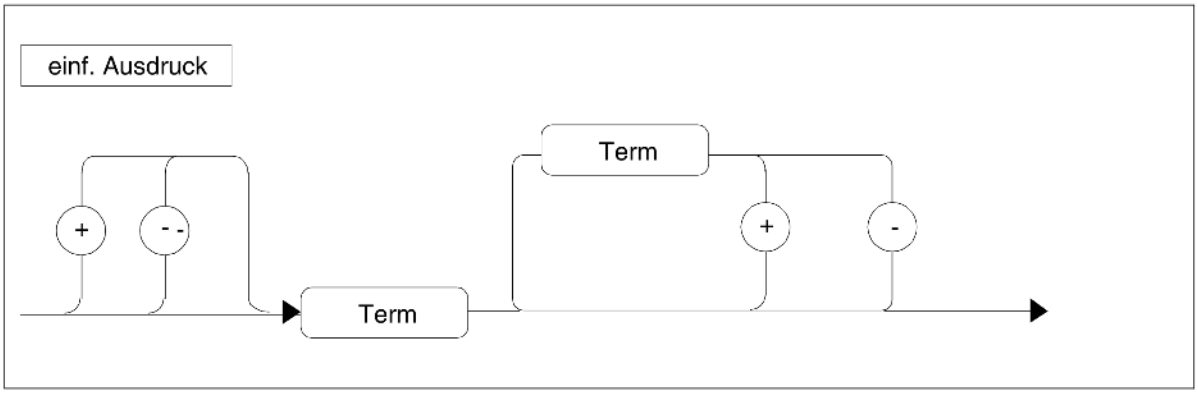


Bild 79: Einfacher Ausdruck

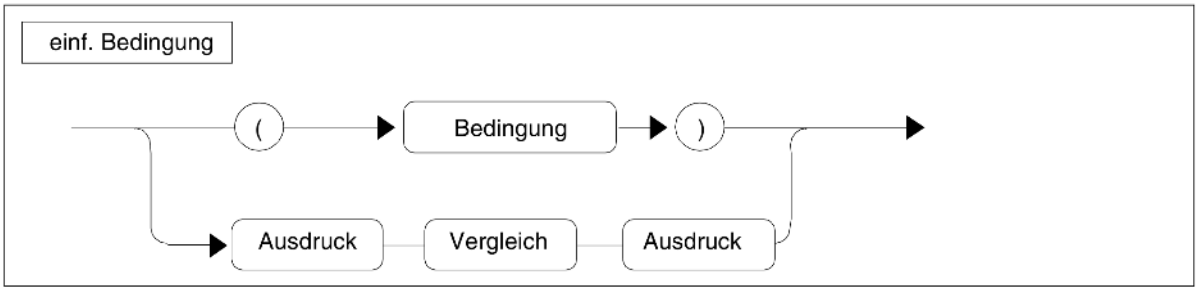


Bild 80: Einfache Bedingung

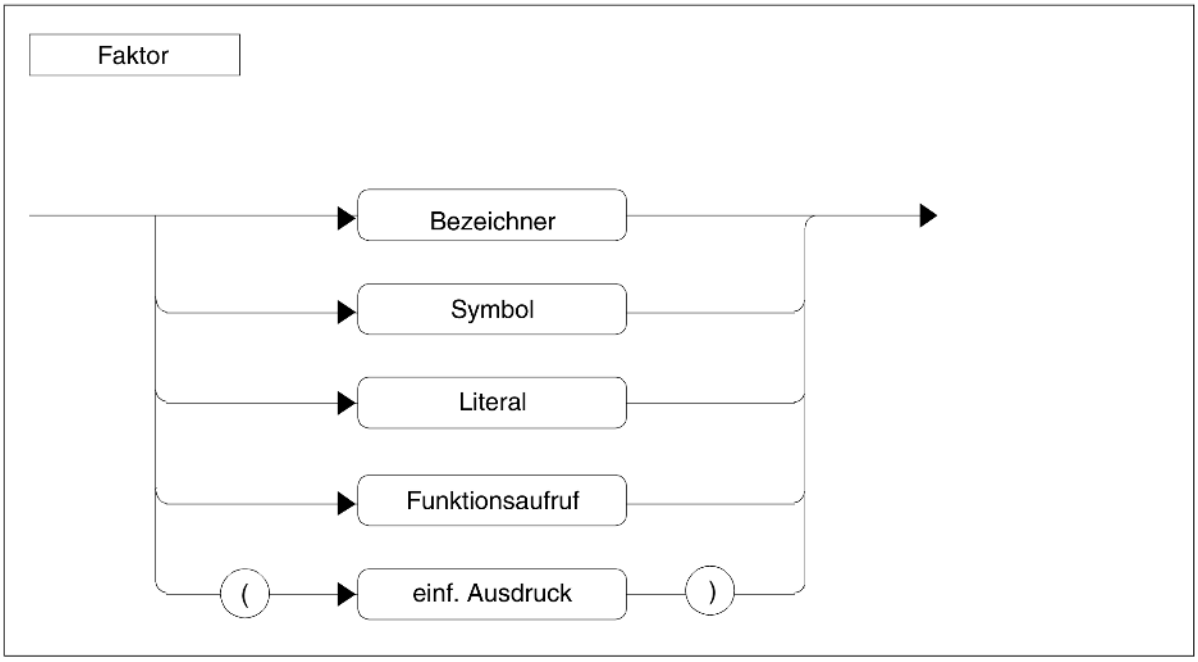


Bild 81: Faktor

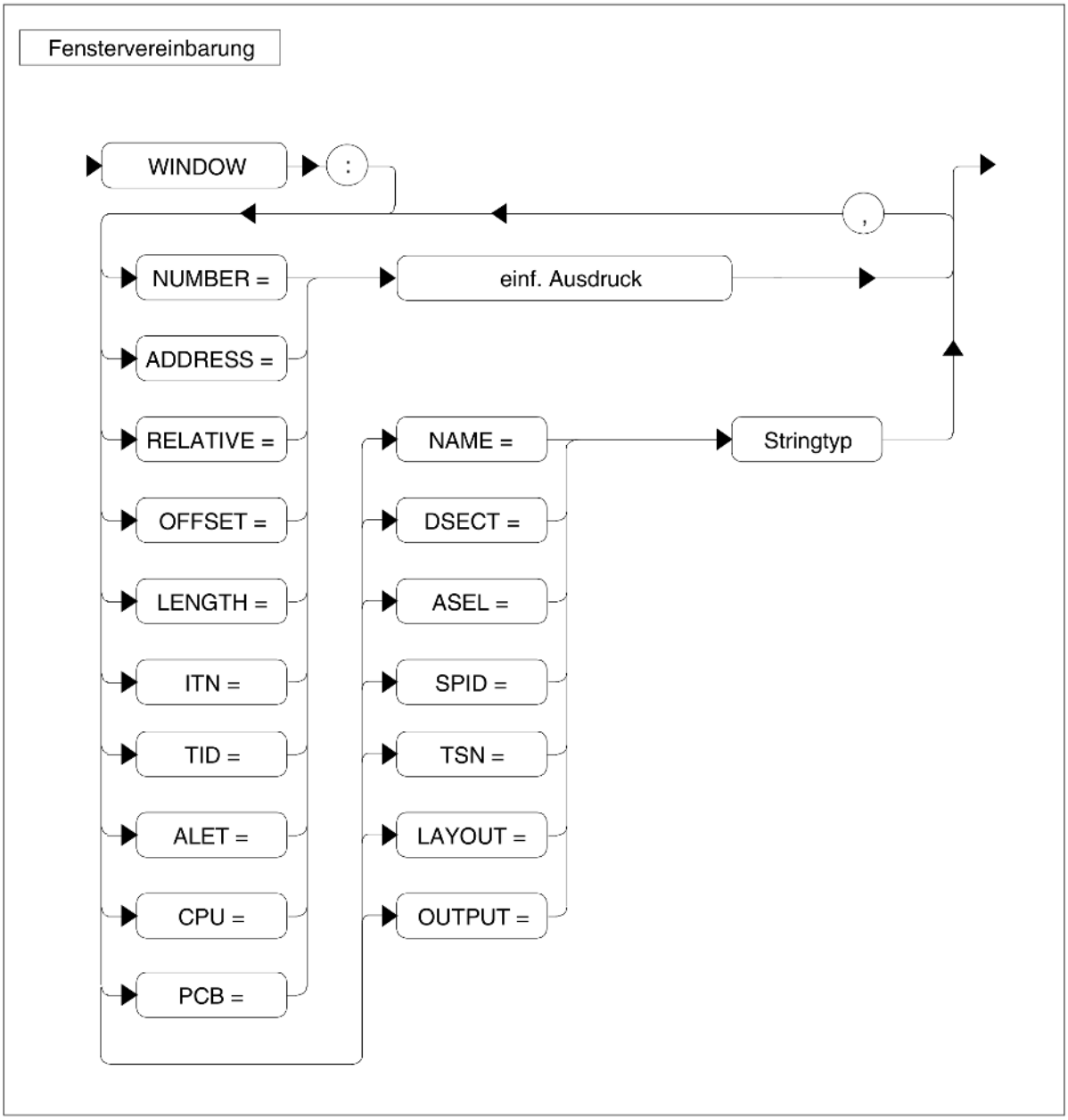


Bild 82: Fenstervereinbarung

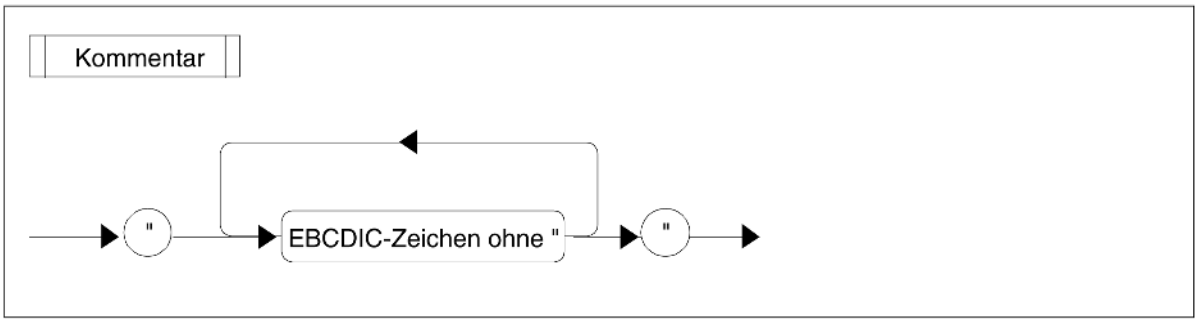


Bild 83: Kommentar

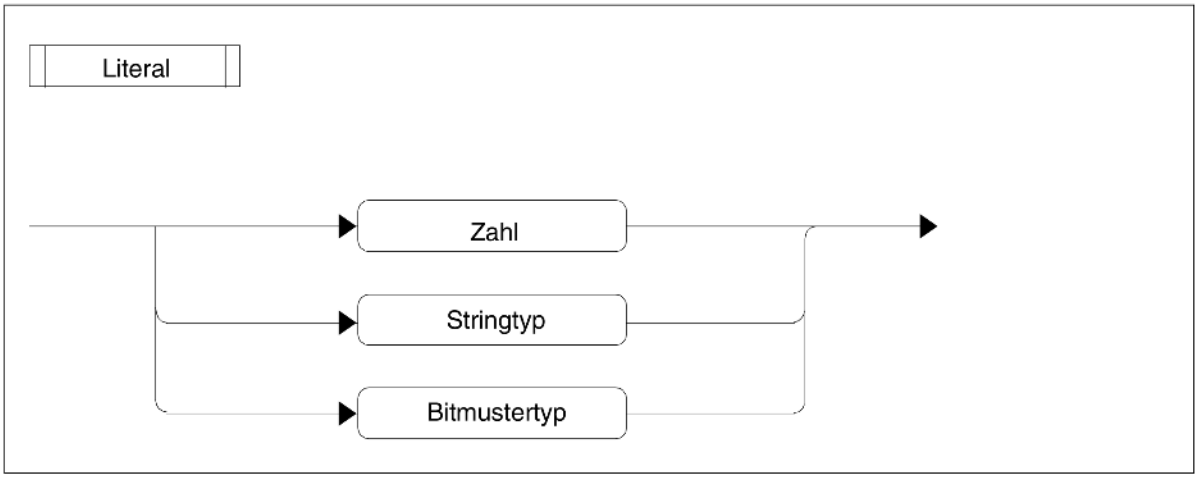


Bild 84: Literal

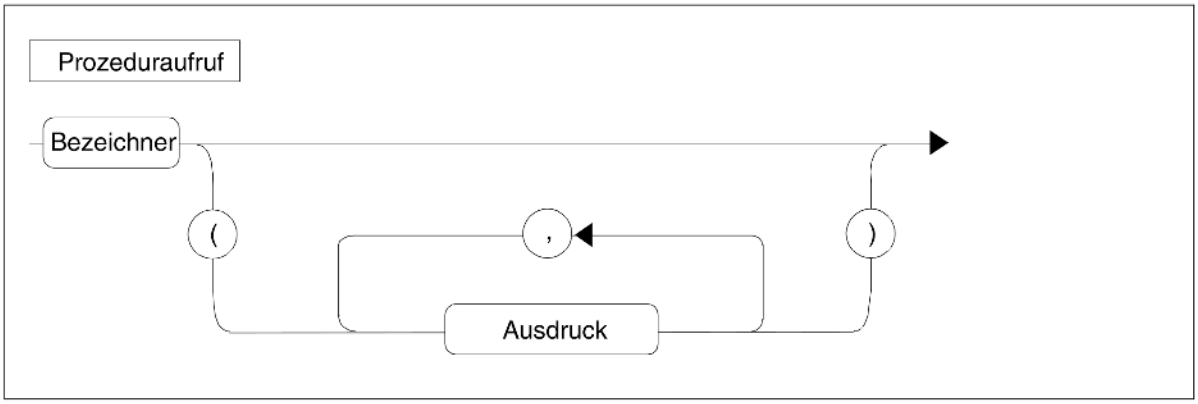


Bild 85: Prozeduraufruf

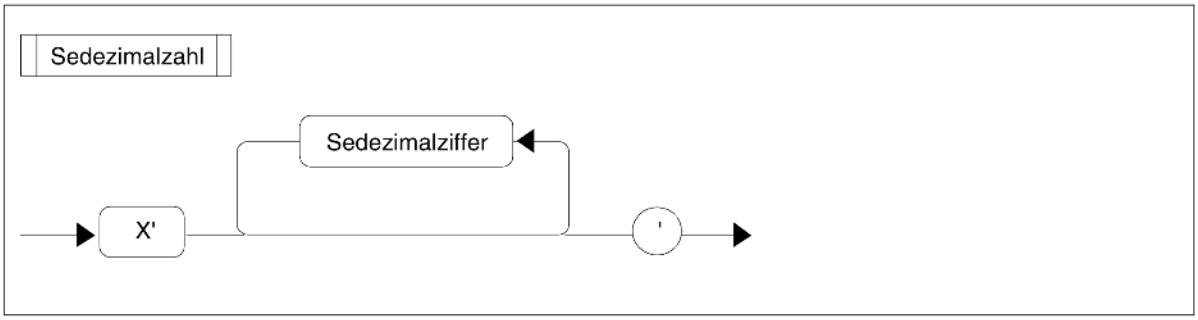


Bild 86: Sedezimalzahl

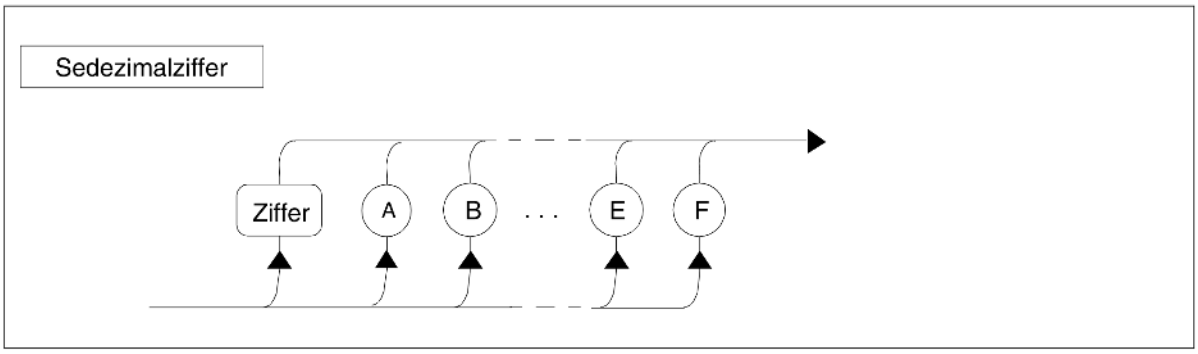


Bild 87: Sedezimalziffer

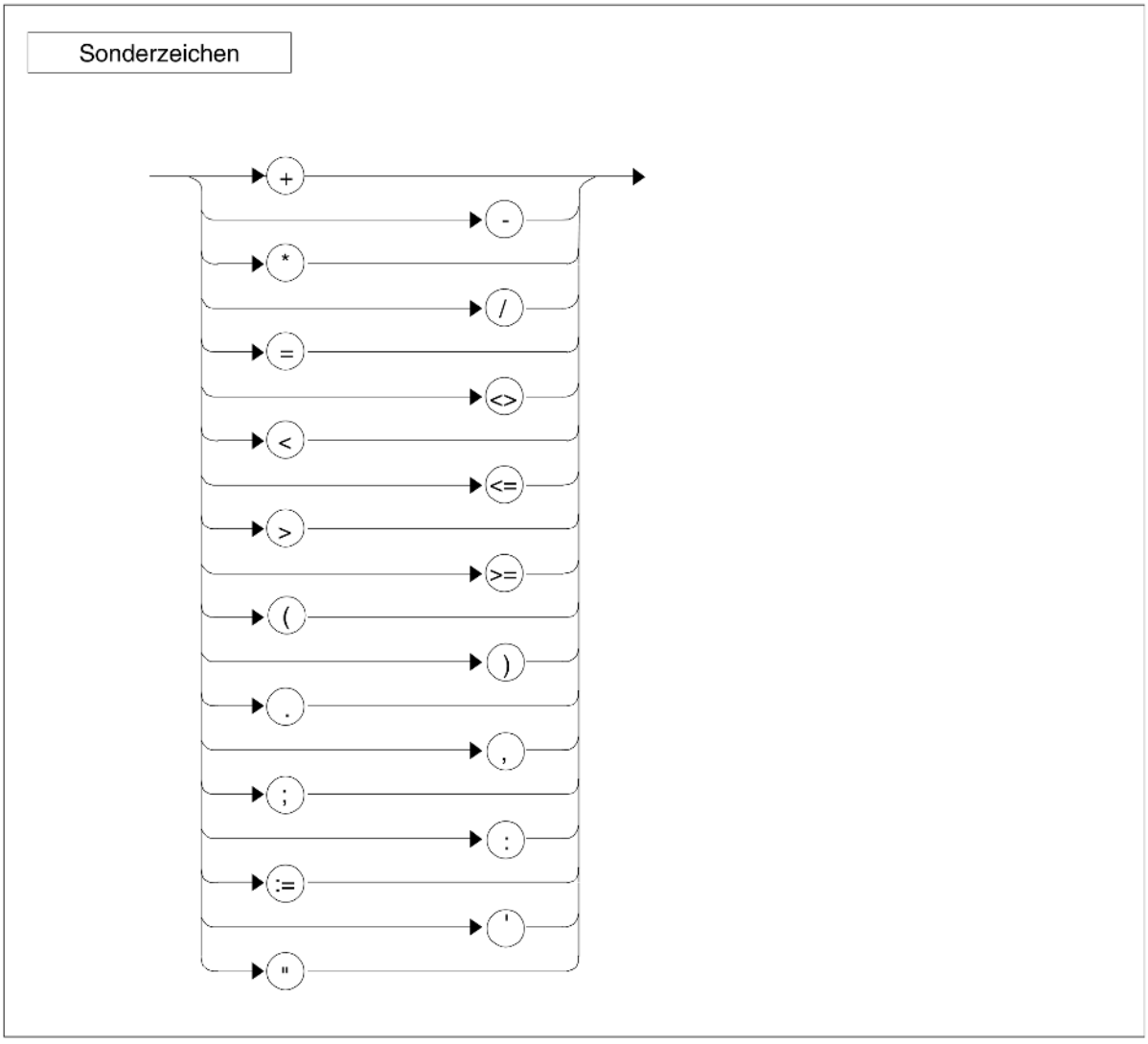


Bild 88: Sonderzeichen

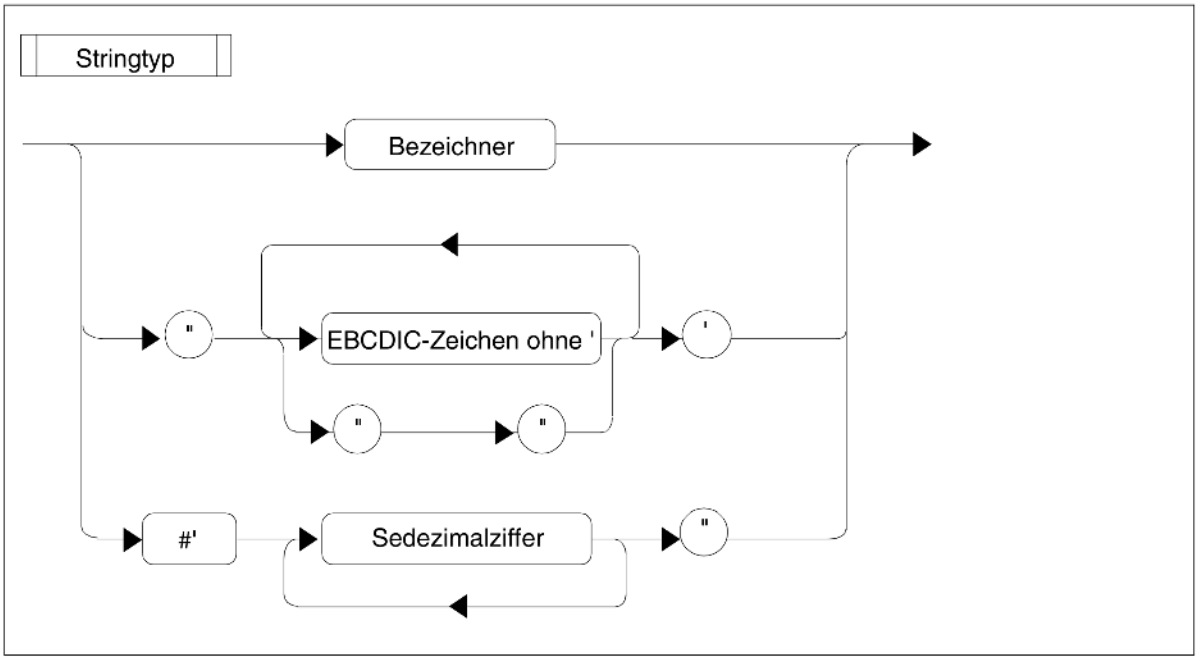


Bild 89: Stringtyp

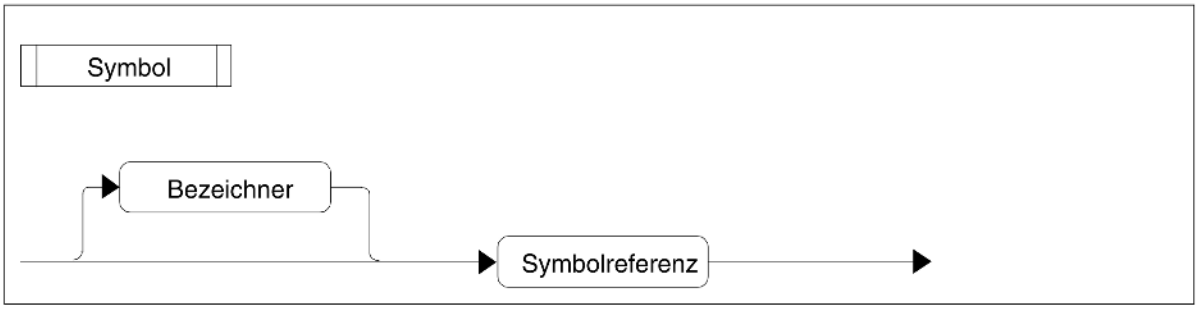


Bild 90: Symbol

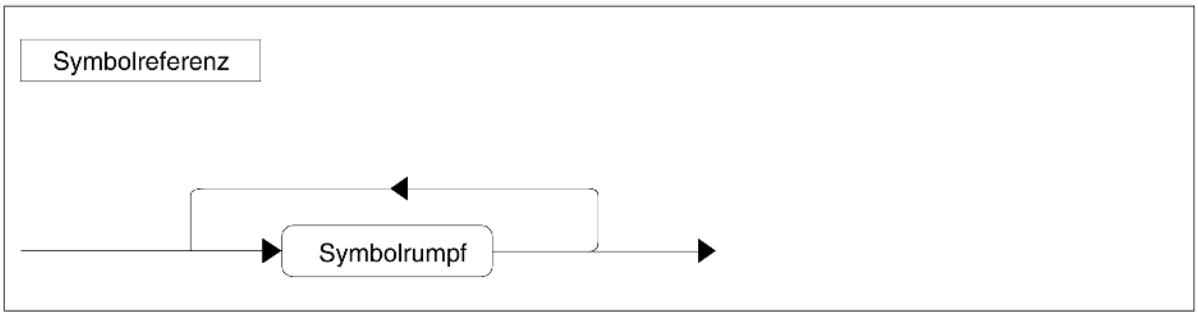


Bild 91: Symbolreferenz

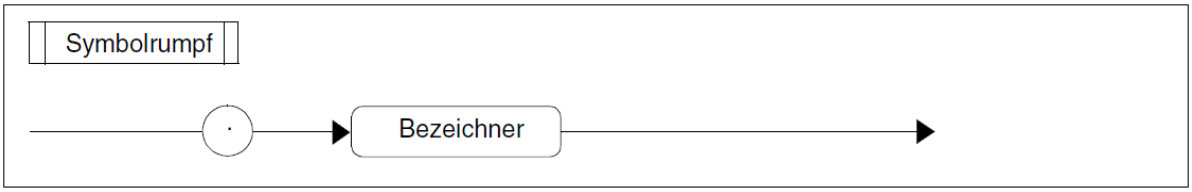


Bild 92: Symbolrumpf

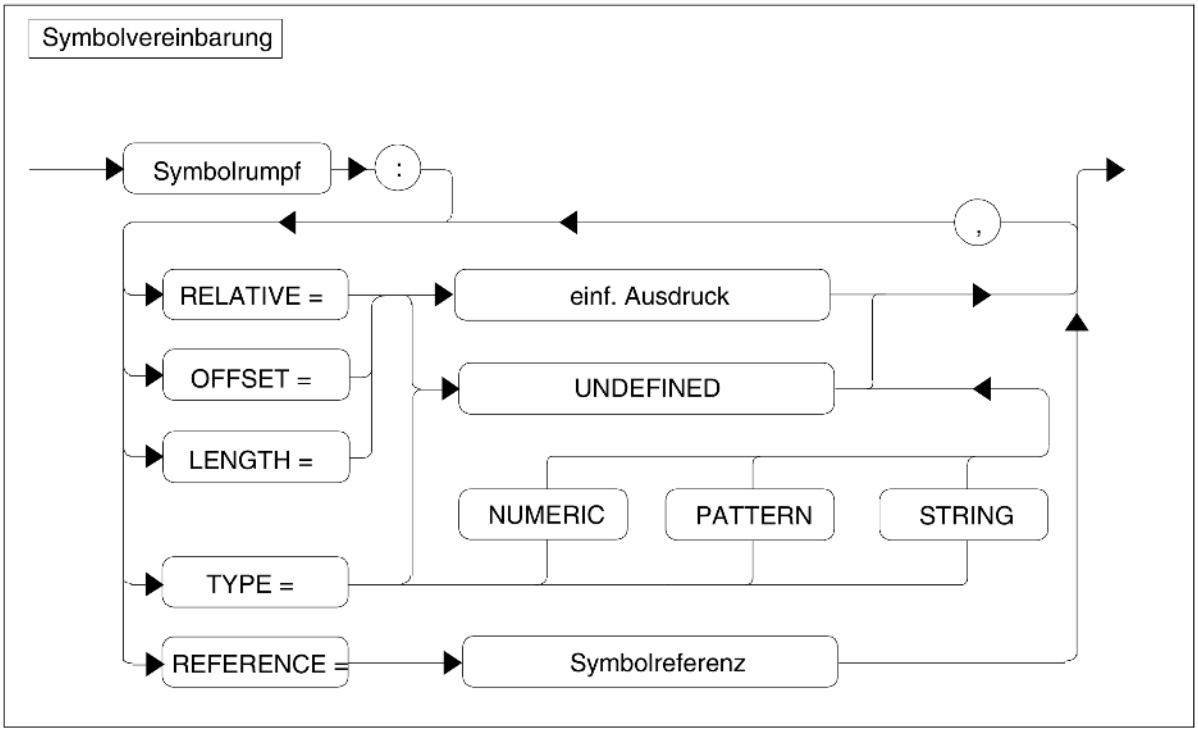


Bild 93: Symbolvereinbarung

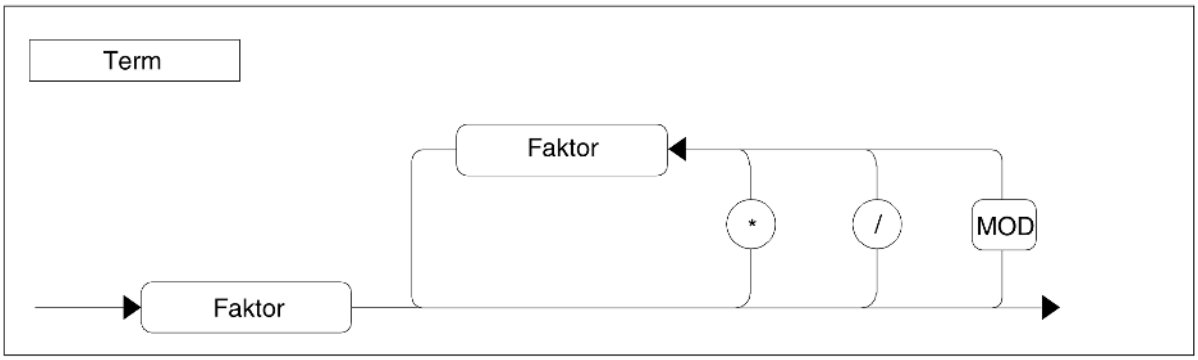


Bild 94: Term

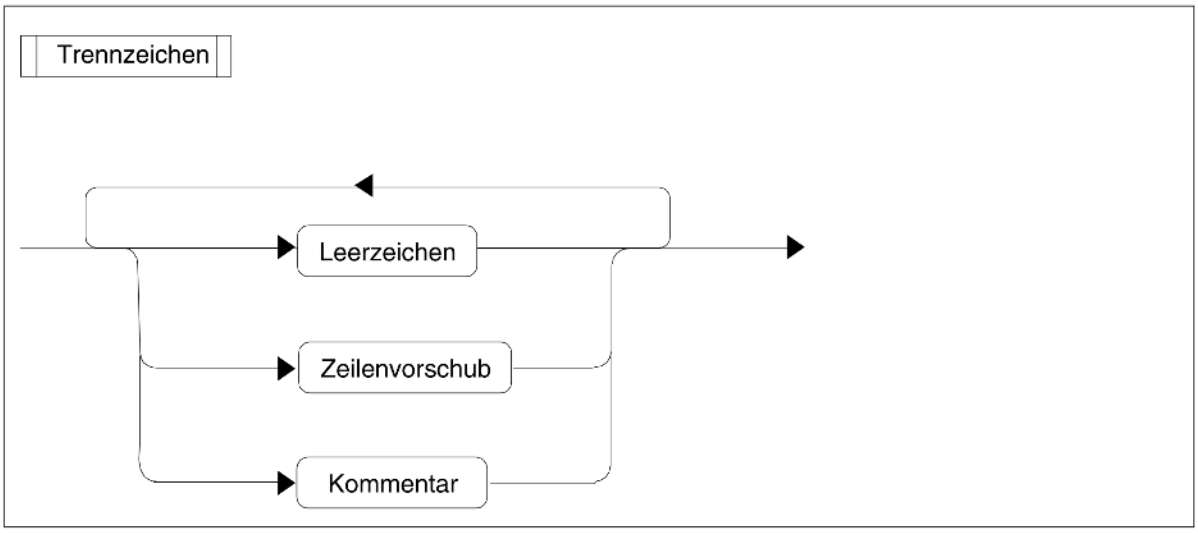


Bild 95: Trennzeichen

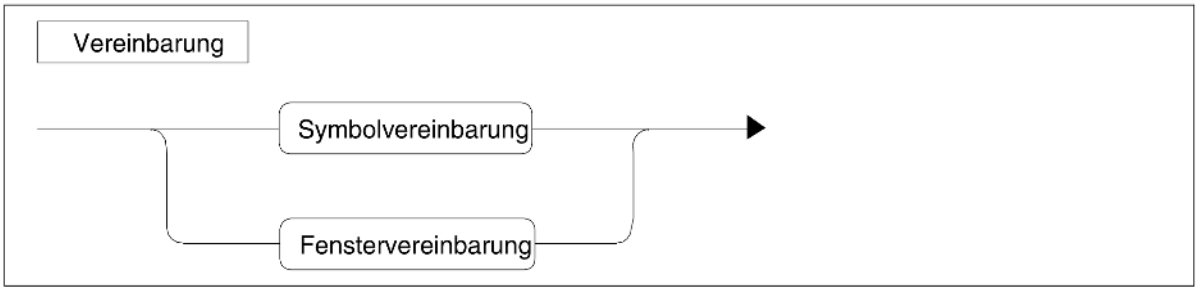


Bild 96: Vereinbarung

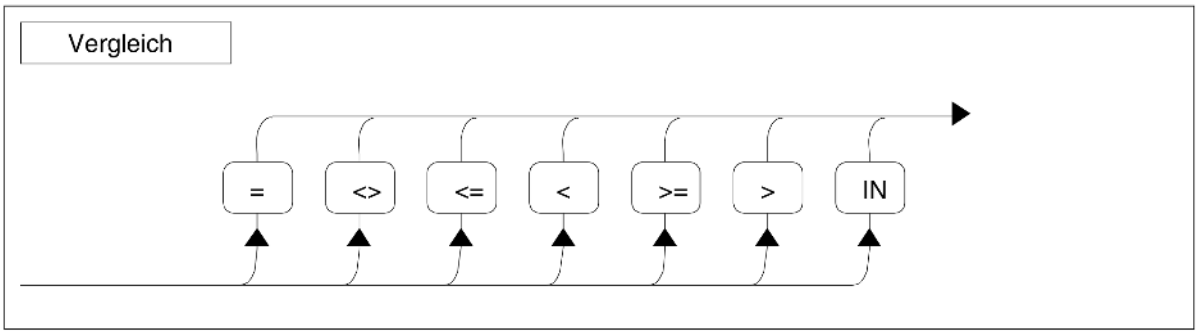


Bild 97: Vergleich

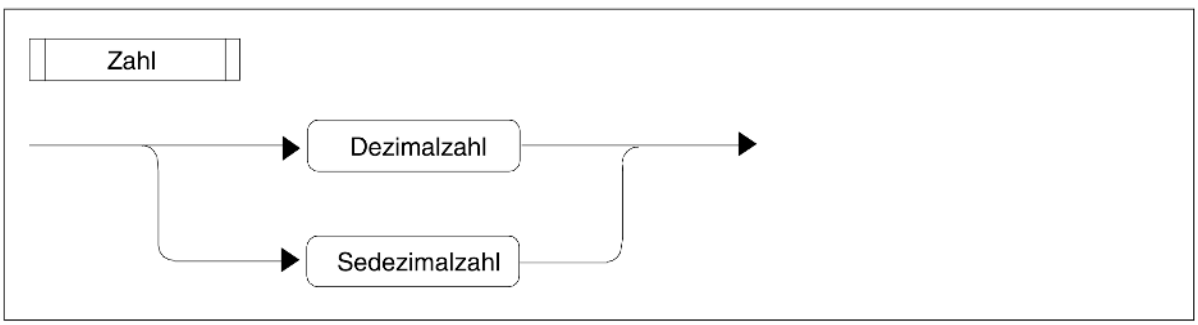


Bild 98: Zahl

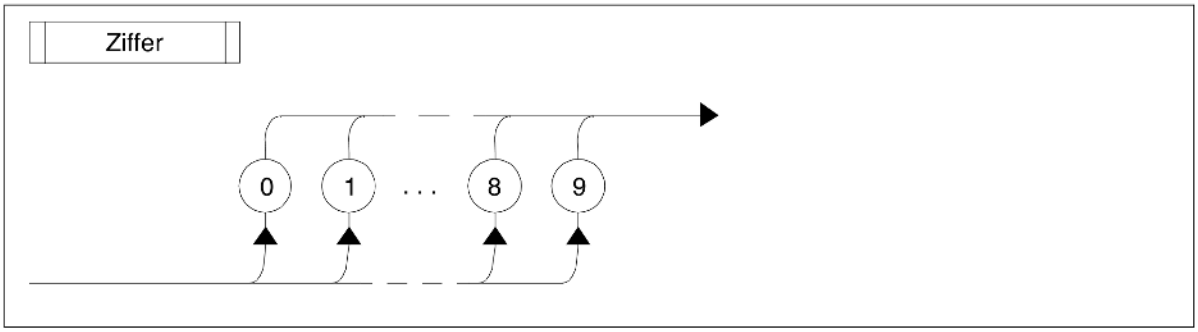


Bild 99: Ziffer

5.8 Software- und Hardware-Voraussetzungen

Installation

In der folgenden Tabelle sind alle mit DAMP (Release-Unit DAMP) ausgelieferten Produkt-Dateien enthalten, die für die Arbeit mit DAMP benötigt werden.

Für jedes Release-Item ist die logische ID (für IMON), der Release-Name und die Funktion aufgeführt. Die Release-Items sind in der Installationsdatei SYSSII.DAMP.<ver> enthalten.

Logische ID	Release-Name	Funktion
SYSSII	SYSSII.DAMP.<ver>	Installationsdatei für DAMP
SYSSDF	SYSSDF.DAMP.<ver>	SDF-Syntax-Datei mit den Kommandos START-DAMP und START-DAMP-SYMBOL-GENERATOR
SYSLNK	SYSLNK.DAMP.<ver>	Nachladebibliothek von DAMP
SYSPRG	SYSPRG.DAMP.<ver>	Ladeprogramm von DAMP
SYSPRG.SYMBOLS	SYSPRG.DAMP.<ver>.SYMBOLS. GEN	Symbolgenerator zur Erzeugung privater Symbole
SYSMSH.D	SYSMSH.DAMP.<ver>.D	Online-Hilfe deutsch
SYSMSH.E	SYSMSH.DAMP.<ver>.E	Online-Hilfe englisch
SYSMES	SYSMES.DAMP.<ver>	Meldungsdatei
SYSSDF.USER	SYSSDF.DAMP.<ver>.USER	Benutzer-SDF-Syntaxdatei mit den DAMP-Anweisungen. Die Syntaxdatei wird von DAMP aktiviert.
SYSSMB	SYSSMB.DAMP.<ver>	Ausgelieferte Bibliothek mit DAMP/BS2000-Symbolen. Nur für Auslieferungszwecke und nicht direkt zu verwenden. Siehe Hinweis Nr. 3.
SYSDMP	SYSDMP.DAMP.<ver>	Ausgelieferte Bibliothek mit DAMP-PRODAMP-Programmen. Enthalten sind unter anderem die PRODAMP-Programme für die Vordiagnose. Die Datei dient nur Auslieferungszwecken und ist nicht direkt zu verwenden. Siehe Hinweis Nr. 3.

Außerdem sind die beiden nachfolgend aufgeführten Dateien für das Arbeiten mit DAMP unbedingt erforderlich (siehe Hinweis Nr. 3).

Dateiname	Funktion
$TSOS.SYSSMB.DAMP	Symbol-Systembibliothek
$TSOS.SYSDMP.DAMP	PRODAMP-Systembibliothek

Hinweise

1. Die Datei SYSPRG.DAMP.<ver> ist auch als Element des Typs C und SYSPRG.DAMP.<ver>.SYMBOLS.GEN unter dem Namen SYMGEN als Element des Typs L in der SYSLNK-Bibliothek enthalten, aus der sie mit den START-Kommandos gestartet werden. Die zusätzliche Auslieferung in Dateiform erfolgt aus Kompatibilitätsgründen.
2. DAMP benutzt bei Zugriffen auf Dumpdateien und das aktive System die Zugriffsmethode ANITA. Die Zugriffsmethode muss ordnungsgemäß installiert sein.
3. Hinweis zur Symbol-Systembibliothek und PRODAMP-Systembibliothek: Standardmäßig (SYSSMB=*STD im OPTS-Fenster) erwartet DAMP die zum Öffnen des Diagnoseobjekts benötigten Symbole in der Datei mit dem festen Namen $TSOS.SYSSMB.DAMP (ohne Versionssuffix!). Bei Installation/Aktualisierung eines BS2000-Systems mit IMON werden in diese Datei nicht nur die von DAMP benötigten Symbole, sondern auch die Symbole weiterer offizieller Produkte eingemischt.

   Analoges gilt für die PRODAMP-Systembibliothek bei Standardinstallation (SYSLNK/SYSDMP=*STD im OPTS-Fenster); die PRODAMP-Systembibliothek hat den festen Namen $TSOS.SYSDMP.DAMP.

DAMP ist versions-entkoppelt

Siehe Abschnitt „Leistungsbeschreibung".

DAMP ist versions-unabhängig

Siehe Abschnitt „Leistungsbeschreibung".

DAMP kann Dump-Dateien bearbeiten, die in anderen BS2000-Versionen und auf anderen BS2000-Servern erstellt wurden.

Zur Auswertung von Dumpdateien aus BS2000 V21.0 muss im System die Bibliothek SYSLNK.ANITA zugreifbar sein und diese die Zugriffsmethode ANITA V21.0A enthalten.

Für die Diagnose des aktiven Systems BS2000 V21.0 muss das Subsystem ANITA V21.0 installiert sein. Es wird von DAMP bei Bedarf automatisch gestartet.

Aufruf von DAMP von anderen Kennungen aus

Das Programmsystem DAMP mit den oben genannten Dateien kann unter einer beliebigen oder unter mehreren Benutzerkennungen eingerichtet werden.

Bei privater Installation ist lediglich darauf zu achten, dass die Dateien mehrbenutzbar (USER-ACCESS=*ALL-USERS) katalogisiert werden und die Installationskennungen (falls nicht TSOS) mit START-OPTION-DIALOG eingestellt werden (siehe Abschnitt „Modifikationen durch den Benutzer (Spezialfenster OPTIONS)").

Weitere Merkmale

Über `LOAD-MODULE` können eigene Auswerteroutinen aus beliebigen Lademodul-Bibliotheken nachgeladen und mit `START-MODULE` gestartet werden (siehe Anweisung `LOAD-MODULE` auf "LOAD-MODULE Externes Unterprogramm laden").

Jeder Benutzer kann für seine Anwendungen geeignete Standardwerte einstellen (siehe „Einstellen der Benutzeroptionen" (Modifikationen durch den Benutzer (Spezialfenster OPTIONS))).

Voraussetzungen für den Zugriff auf das aktive System

Für den Zugriff auf Informationen im aktiven System wird eine Testprivilegierung benötigt.

Diese ist vom Systemverwalter mit folgendem Kommando einzurichten:

```
ADD-USER USER-IDENTIFICATION=userid,...,
         TEST-OPTIONS=*PARAMETERS(READ-PRIVILEGE=8,
         WRITE-PRIVILEGE=1[,MODIFICATION=*CONTROLLED])
```

Vor dem Aufruf von DAMP muss die Testprivilegierung mit folgendem Kommando aktiviert werden:

```
MODIFY-TEST-OPTIONS PRIVILEGE=*PARAMETERS(READ=8,WRITE=1)
```

Unterstützte Terminals

Unterstützt wird die 9750-Emulationen von PCs.

> ¡ Das Benutzen von KPAC=4 unter OMNIS ist verboten, wenn an einem der Partner DAMP läuft, da DAMP die K4-Sequenz zu eigenen Zwecken benutzt.

5.9 Liste der DSECTs aus den Standard-Symboldateien

Die für die Auswertung eines BS2000-Speicherauszugs oder des aktives BS2000-Systems benötigten Symbole werden im Folgenden aufgeführt. Die Symbole müssen als LMS-Elemente vom Typ X in der beim Öffnen verwendeten Symbolbibliothek vorliegen.

Standardmäßig erwartet DAMP die Symbole in der Symbol-Systembibliothek, dies ist bei einer Standardinstallation von DAMP die Datei mit festem Pfadnamen $TSOS.SYSSMB.DAMP.

LMS		**Benötigt für Objekte aus BS2000 OSD/BC Version**	**HSI**
Elementname	**Version**		
BS2000	200	20.0A / V11.0A	/390
BS2000-USER	200	20.0A / V11.0A	/390
XA2000	200	20.0A / V11.0A	x86
XA2000-USER	200	20.0A / V11.0A	x86
BS2000	210	21.0A	/390
BS2000-USER	210	21.0A	/390
XA2000	210	21.0A	x86
XA2000-USER	210	21.0A	x86
STATUS	*	Abhängig von dem Typ und der Version des DUMP-Generators	
NSDI0	*		

Tabelle 15: DAMP-Symbole in der Bibliothek $TSOS.SYSSMB.DAMP

Die Elemente STATUS und NSDI0 werden nur für Sonderfälle benötigt; sie sind in folgenden Versionen in der System-Symbolbibliothek vorhanden:

```
STATUS:   006/010
NSDI0:    200/210
```

Die Unterscheidung zwischen BS2000- und XA2000-Elementen ergibt sich daraus, dass in einigen hardwarenahen DSECTs HSI-spezifische Unterschiede vorliegen.

Es wird (neben STATUS und NSDI0) ein BS2000- und ein BS2000-USER-Element (bzw. ein XA2000- und ein XA2000-USER-Element) benötigt.

Beim standardmäßigen Öffnen eines Diagnoseobjekts (keine Angabe bzw. Voreinstellungen für SYMBOLS in OPEN-DIAGNOSIS-OBJECT) lädt DAMP die benötigten Symbole automatisch. Die im Folgenden aufgelisteten DSECTs und SPL-Strukturen sind dann in den DAMP-Fenstern und in PRODAMP-Programmen für die Analyse verfügbar.

DSECTs mit fett gedruckten Namen in der folgenden Aufstellung können automatisch lokalisiert werden.

Elemente BS2000/200 und XA2000/200 (für BS2000 V20.0 = BS2000 OSD/BC V11.0 mit /390-HSI bzw. x86-HSI)

Assembler DSECTs

ASAVDSSM	ASIMDBHD	ASIPUCON	DSTE	DWQE	DWQH
EBWL	ECSA	ECSE	ECSX	ECTLP	EERLWA
EGCTRAC	**EJCB**	EJTBP	EMICWA	EMMDDSMD	EMRCWA
EOLDTBLE	EORD	EPDR	ERTWA	ESOFWA	ESTK
ETCB	ETLT	ETMGPT	ETRAC	**EVSMT**	**EVUMT**
EVYVWA	**EXVT**	FPTA34	FPTE34	HCTX	IAR
IBW	IDMTFT	ID1FCB	ID1FCBE	ID2FCB	IELS
INTE	INTEMMP	INTESMP	ML	NDXMTE	NDXT
NEHX$MDL	NI0SC	NLKDEXT	NLKDHEA	NLKDSAV	NLKSJDS
NOTEDS	NRXDPL	NRXIPL	NRXPPL	NSCDH	NSCDL
NSDI0	NSDI1	NSDLPLD	NSDPPLD	NSDTPLD	NSICONFT
NSISWID	PDTHDR	PDTREC	PPTE	PSA	RECBUFF
STRCWA	TERMMSG	TLTE	TTSAVE	VATE	XD1FCB

SPL-Strukturen

BS_CTX_VECTOR_REC_MDL	CTX_VECTOR_REC_MDL
DBL_OPTIONS_COM_MDL	DBL_OPTIONS_P_C_MDL
DBL_OPTIONS_S_P_MDL	LIBRARY_TAB_D_MDL
LU_CTX_VECTOR_REC_MDL	LU_MEM_POOL_VECTOR_REC_MDL
MEM_POOL_REC_MDL	NSIVR_MDL
NTFHOOK_MDL	PBMM_ATTR_MDL
PROGRAM_LOAD_LIST_MDL	RECORD_D_MDL
RECORD_H_MDL	TABLE_D_MDL
TASK_TAB_MDL	VERSION_MDL

Elemente BS2000-USER/200 und XA2000-USER/200 (für BS2000 V20.0 = BS2000 OSD/BC V11.0 mit /390-HSI bzw. x86-HSI)

Assembler DSECTs

CSTMP	DDZCCB	DECRCOD	DECRNAM	DRPVST	DSFTB
DSHED	DSPTB	DSSTB	DSUTB	DS3BCB	DS4LBL
DS6STK	EACQ	EBVDT	ECSE	EGCARIGT	EGCMXLDS
EGCW_MDL	EGSTRAC	ENRTPL	EPDMM	EPPT	ESTK
ETCOMEV	ETCOMTBL	ETCOPRTL	ETMCH	ETMMH	GARE
IBO	ICACFCP	ICACFDP	ICACFFP	ICACFHP	ICACFMP
ICACFPP	ICAEE4	ICO	IDBCHAPL	IDBCOPPL	IDBERAPL
IDBFSTPL	IDBPFLPL	IDBPFLPX	IDBRELPL	IDCEG	IDCES
IDCEXS	IDEE3	IDJES	IDJEXT	IDJEXT2	IDKCATPL
IDPBTAPL	IDQPAMPL	IDVTS	INST	LFCB	NAR
NDVESPDS	NEHX$MDL	NERRLOCK	NLOCK	NLPT	NLWALOCK
NRTSEHDT	NSCB	NSPAPLD	NSPR	NSUBLOCK	NTIM
NTRCLOCK	NVPSPL	RKLOG	SD	SPAD	SPOD
WORDLIST	XDBFSTPL	XDPBTAPL	XDQPAMPL	XRD	

SPL-Strukturen

$JCBRW_PL_MDL	$SSMCEO_PL_MDL
$SSMENT_PL_MDL	$SSMERA_PL_MDL
ADDPLNK_MDL	CREPOOL_MDL
DELPOOL_MDL	ECSA_MDL
EAM_MDL	ESMFHDR
ESMIFID_MDL	ESMRETC_MDL
ESTK_MDL	NSIVR_MDL
PAM_MDL	REMPLNK_MDL
VERSION_MDL	

Elemente BS2000/210 und XA2000/210 (für BS2000 V21.0 mit /390-HSI bzw. x86-HSI)

Assembler DSECTs

ASAVDSSM	ASIMDBHD	ASIPUCON	DSTE	DWQE	DWQH
EBWL	ECSA	ECSE	ECSX	ECTLP	EERLWA
EGCTRAC	**EJCB**	EJTBP	EMICWA	EMMDDSMD	EMRCWA
EOLDTBLE	EORD	EPDR	ERTWA	ESOFWA	ESTK
ETCB	ETLT	ETMGPT	ETRAC	**EVSMT**	**EVUMT**
EVYVWA	**EXVT**	FPTA34	FPTE34	HCTX	IAR
IBW	IDMTFT	ID1FCB	ID1FCBE	ID2FCB	IELS
INTE	INTEMMP	INTESMP	ML	NDXMTE	NDXT
NEHX$MDL	NI0SC	NLKDEXT	NLKDHEA	NLKDSAV	NLKSJDS
NOTEDS	NRXDPL	NRXIPL	NRXPPL	NSCDH	NSCDL
NSDI0	NSDI1	NSDLPLD	NSDPPLD	NSDTPLD	NSICONFT
NSISWID	PDTHDR	PDTREC	PPTE	PSA	RECBUFF
STRCWA	TERMMSG	TLTE	TTSAVE	VATE	XD1FCB

SPL-Strukturen

BS_CTX_VECTOR_REC_MDL	CTX_VECTOR_REC_MDL
DBL_OPTIONS_COM_MDL	DBL_OPTIONS_P_C_MDL
DBL_OPTIONS_S_P_MDL	LIBRARY_TAB_D_MDL
LU_CTX_VECTOR_REC_MDL	LU_MEM_POOL_VECTOR_REC_MDL
MEM_POOL_REC_MDL	NSIVR_MDL
NTFHOOK_MDL	PBMM_ATTR_MDL
PROGRAM_LOAD_LIST_MDL	RECORD_D_MDL
RECORD_H_MDL	TABLE_D_MDL
TASK_TAB_MDL	VERSION_MDL

Elemente BS2000-USER/210 und XA2000-USER/210 (für BS2000 V21.0 mit /390-HSI bzw. x86-HSI)

Assembler DSECTs

CSTMP	DDZCCB	DECRCOD	DECRNAM	DRPVST	DSFTB
DSHED	DSPTB	DSSTB	DSUTB	DS3BCB	DS4LBL
DS6STK	EACQ	EBVDT	ECSE	EGCARIGT	EGCMXLDS
EGCW_MDL	EGSTRAC	ENRTPL	EPDMM	EPPT	ESTK
ETCOMEV	ETCOMTBL	ETCOPRTL	ETMCH	ETMMH	GARE
IBO	ICACFCP	ICACFDP	ICACFFP	ICACFHP	ICACFMP
ICACFPP	ICAEE4	ICO	IDBCHAPL	IDBCOPPL	IDBERAPL
IDBFSTPL	IDBPFLPL	IDBPFLPX	IDBRELPL	IDCEG	IDCES
IDCEXS	IDEE3	IDJES	IDJEXT	IDJEXT2	IDKCATPL
IDPBTAPL	IDQPAMPL	IDVTS	INST	LFCB	NAR
NDVESPDS	NEHX$MDL	NERRLOCK	NLOCK	NLPT	NLWALOCK
NRTSEHDT	NSCB	NSPAPLD	NSPR	NSUBLOCK	NTIM
NTRCLOCK	NVPSPL	RKLOG	SD	SPAD	SPOD
WORDLIST	XDBFSTPL	XDPBTAPL	XDQPAMPL	XRD	

SPL-Strukturen

$JCBRW_PL_MDL	$SSMCEO_PL_MDL
$SSMENT_PL_MDL	$SSMERA_PL_MDL
ADDPLNK_MDL	CREPOOL_MDL
DELPOOL_MDL	ECSA_MDL
EAM_MDL	ESMFHDR
ESMIFID_MDL	ESMRETC_MDL
ESTK_MDL	NSIVR_MDL
PAM_MDL	REMPLNK_MDL
VERSION_MDL	

5.10 DAMP-Meldungen

Die Meldungen von DAMP haben die Meldungsklasse DMP. Informationen über einzelne Meldungen erhalten Sie im laufenden Betrieb mit /HELP-MSG-INFORMATION.

6 NDMDAMP Erzeugung von Diagnoseunterlagen

NDMDAMP ist ein PRODAMP-Prozedurpaket, das innerhalb DAMP die für NDM relevanten Daten aus einem SLED, Systemdump oder dem aktiven System extrahiert und auswertet.

Zur Steuerung des Ausgabeumfangs der Dump-Funktion werden drei Möglichkeiten angeboten:

- eine Normalauswertung,
- eine aufwändigere Maximalauswertung oder
- eine (auf das jeweilige Problem zugeschnittene) eingeschränkte Auswertung.

NDMDAMP kann im Dialog, direkt über DAMP oder über vorgefertigte ENTER-Jobs aufgerufen werden. Je nach Aufrufart unterscheidet sich die Art der Parametrisierung.

6.1 Aufruf von NDMDAMP

NDMDAMP kann auf verschiedene Arten aufgerufen werden:

- im Dialog über Kommando START-NDM-DIAGNOSIS

  Das Kommando START-NDM-DIAGNOSIS steht nach dem Kommando MOD[IFY]-SDF[-OPTIONS] [$TSOS.]SYSSDF.NDMDAMP.<ver>.USER zur Verfügung.

- aus DAMP

  Beim Aufruf aus DAMP können die Standard-Auswertungen oder Sonder-Auswertungen gewählt werden. Die Einstellungen der Optionen müssen explizit in die PRODAMP-Prozedur NDM eingetragen werden.

- mit vorgefertigten ENTER-Jobs

  Die LMS-Bibliothek SYSENT.NDMDAMP.<ver> wird mit sechs ENTER-Jobs ausgeliefert, die unterschiedliche Auswertungen bieten.

6.1.1 START-NDM-DIAGNOSIS NDM-Daten auswerten

Funktionsbeschreibung

Das Kommando START-NDM-DIAGNOSIS wertet die NDM-Daten aus dem aktiven System, aus einem Systemdump oder aus einem SLED-Dump mittels NDMDAMP aus.

Format

START-NDM-DIAGNOSIS

OBJECT = ***SYSTEM** / <filename 1..54> / ***LINK**(...)

***LINK**(...)

| **LINK-NAME** = <filename 1..8 without-gen>

,INFORMATION = ***STD** / ***ALL / *TRACE / *DRV / *DRV-ALL / *EXECUTION-TRACE** / ***PAR**AMETERS(...)

***PAR**AMETERS(...)

| **IO-CONTROL-DATA** = ***STD** / ***NO / *ALL / *ALL-WITH-EXECUTION-TRACE**

| ,**BAVOLMON** = ***STD** / ***NO / *ALL / *ALL-WITH-EXECUTION-TRACE**

| ,**NKA** = ***STD** / ***NO / *ALL / *ALL-WITH-EXECUTION-TRACE**

| ,**NKV** = ***STD** / ***NO / *STD-WITH-EXECUTION-TRACE**

| ,**NKS** = ***STD** / ***NO / *ALL / *ALL-WITH-EXECUTION-TRACE**

| ,**NKR** = ***STD** / ***NO / *ALL / *ALL-WITH-EXECUTION-TRACE**

| ,**DRV** = ***STD** / ***NO / *STD-WITH-EXECUTION-TRACE**

| ,**TRACE** = ***STD** / ***NO / *STD-WITH-EXECUTION-TRACE**

,**ENVIRONMENT** = ***STD** / ***PAR**AMETERS(...)

***PAR**AMETERS(...)

| **PROGRAM-NAME** = ***STD** / <filename 1..54 without-gen>

| **,SYMBOL-LIBRARY** = ***STD** / <filename 1..54 without-gen>

| **,PRODAMP-LIBRARY** = ***STD** / <filename 1..54 without-gen>

| **,NUMBER-OF-RESTARTS** = **1** / <integer 0..5>

| **,OUTPUT** = ***STD** / ***SYSLST** / <filename 1..54>

Operandenbeschreibung

OBJECT = *SYSTEM / <filename 1..54> / *LINK(...)
Gibt an, welche Dump-Datei ausgewertet werden soll.

OBJECT = *LINK(...)

LINK-NAME = <filename 1..8 without-gen>
Die Dump-Datei wird über ihren Link-Namen angegeben.

INFORMATION =
Gibt den Umfang der Dump-Auswertung an.

INFORMATION = *STD
Maximal-Auswertung ohne EXPDT, ohne tasklokale Daten von NKA und NKS sowie ohne NKR-Module.

INFORMATION = *ALL
Maximal-Auswertung.

INFORMATION = *TRACE
Nur NDM-Traces.

INFORMATION = *DRV
Nur DRV- und NDM-Trace.

INFORMATION = *DRV-ALL
NKA, NKV, DRV, IO-CONTROL ohne EXPDT, BAVOLMON und NDM-Traces.

INFORMATION = *EXECUTION-TRACE
Maximal-Auswertung mit eingeschaltetem PRODAMP-Trace. Hierzu muss der EDT verfügbar sein.

INFORMATION = *PARAMETERS(...)

IO-CONTROL-DATA = *STD / *NO / *ALL / *ALL-WITH-EXECUTION-TRACE
Auswertung der IO-CONTROL-Daten. Bei *ALL wird zusätzlich der Datenmodul EXPDT ausgegeben.

BAVOLMON = *STD / *NO / *ALL / *ALL-WITH-EXECUTION-TRACE
Auswertung der Basic-Volume-Überwachung. Bei *ALL wird für NDIVT Zusatzinformation ausgegeben.

NKA = *STD / *NO / *ALL / *ALL-WITH-EXECUTION-TRACE
Auswertung des NKALLOC (NDM Allocator). Bei *ALL werden zusätzlich tasklokale Daten ausgegeben.

NKV = *STD / *NO / *STD-WITH-EXECUTION-TRACE
Auswertung der NKVMOUNT (NDM-Datenträger-Überwachung).

NKS = *STD / *NO / *ALL / *ALL-WITH-EXECUTION-TRACE
Auswertung des NKSECRES (NDM Secure). Bei *ALL werden zusätzlich tasklokale Daten ausgegeben.

NKR = *STD / *NO / *ALL / *ALL-WITH-EXECUTION-TRACE
Auswertung der NKRECONF (NDM-Rekonfigurationsverwaltung). Bei *ALL werden zusätzlich NKR-Module ausgegeben.

DRV = *STD / *NO / *STD-WITH-EXECUTION-TRACE
Auswertung des DRV (Dual Recording by Volume).

TRACE = *STD / *NO / *STD-WITH-EXECUTION-TRACE
Auswertung des NDM-Trace.

ENVIRONMENT = *STD / *PARAMETERS(...)
Definition der Dump-Umgebung.

ENVIRONMENT = *PARAMETERS(...)

PROGRAM-NAME = *STD / <filename 1..54 without-gen>
Angabe der DAMP-Datei.
Bei *STD wird /START-DAMP verwendet.

SYMBOL-LIBRARY = *STD / <filename 1..54 without-gen>
Name der Symbolbibliothek.
Bei *STD wird die System-Bibliothek von DAMP verwendet: SYSDMP.DAMP.

PRODAMP-LIBRARY = *STD / <filename 1..54 without-gen>
Name der Prodamp-Bibliothek mit NDMDAMP.
Bei *STD wird die System-Bibliothek von DAMP verwendet: SYSDMP.DAMP.

NUMBER-OF-RESTARTS = 1 / <integer 0..5>
Anzahl der Versuche, nach Fehlern in NDMDAMP mit der nächsten Prozedur wieder aufzusetzen. Wird ein Wert größer 0 angegeben, muss der EDT verfügbar sein.

OUTPUT = *STD / *SYSLST / <filename 1..54>
Name der Ausgabedatei.
Bei *STD wird SYSLST.NDMDAMP.<date>.<time> verwendet.

6.1.2 Aufruf von NDMDAMP aus DAMP

Beim Aufruf aus DAMP können Standard-Auswertungen oder Sonder-Auswertungen gewählt werden.

Standard-Auswertungen

Standard-Auswertungen sind die im Kommando START-NDM-DIAGNOSIS, Operand INFORMATION beschriebenen Auswertungen (siehe "START-NDM-DIAGNOSIS NDM-Daten auswerten"), ausgenommen der von INFORMATION=*PARAMETERS(...).

Voraussetzungen:

- Die Dump-Datei (bzw. das laufende System als Dump-Objekt) muss geöffnet sein.
- Die System-Symbolbibliothek ($TSOS.SYSSMB.DAMP) muss verfügbar sein und die Elemente NDM, NDMNUC und DRV für die auszuwertende Systemversion enthalten. Andernfalls sind sie folgendermaßen zuzuweisen:

  ```
  ADD-SYMBOLS LIBRARY = *STD / <datei> (ELEMENT = NDM / NDMNUC / DRV)
  ```

- Die System-PRODAMP-Bibliothek ($TSOS.SYSDMP.DAMP) muss die NDMDAMP-PRODAMP-Objekte enthalten.

  Die System-PRODAMP-Bibliothek ist in DAMP nicht standardmäßig voreingestellt. Deswegen muss sie in der Regel folgendermaßen zugewiesen werden:

  ```
  ASSIGN-PRODAMP-LIBRARIES OBJECT-LIBRARY = *PRODAMP-SYSTEM-LIBRARY
  ```

  oder kurz

  ```
  A-P-L O=*P.
  ```

  Statt des Schlüsselwortes *PRODAMP-SYSTEM-LIBRARY kann der Pfadname der Bibliothek, die die NDMDAMP-PRODAMP-Objekte enthält, angegeben werden.
- Die logische Systemdatei SYSLST kann, falls gewünscht, einer Datei zugewiesen werden.

Durch die Eingabe von

```
START-PRODAMP-PROGRAM NAME = NDM / NDMTRACE / NDMALL / NDMSTD / DRV / DRVALL
```

in der DAMP-Kommandozeile werden die Prozeduren gestartet.

Die PRODAMP-Prozeduren entsprechen dem mit dem Operanden INFORMATION auswählbaren Auswerteumfang des Kommandos /START-NDM-DIAGNOSIS. Der Auswerteumfang von NDM entspricht dabei dem von NDMSTD, wobei für NDM ein Restart möglich ist, für NDMSTD dagegen nicht.

Das Wiederaufsetzen der Auswertung nach Fehlern ist, ausgenommen für NDMSTD, mit

```
START-PRODAMP-PROGRAM NAME = NDM-RESTART
```

möglich. Für den Restart muss der EDT verfügbar sein, da die Arbeitsdatei 9 alle für den Restart relevanten Daten enthält. Diese dürfen vor dem Restart-Aufruf nicht modifiziert oder gelöscht werden.

Sonder-Auswertungen

Sonderauswertungen sind die im Kommando START-NDM-DIAGNOSIS mit INFORMATION=*PAR individuell einstellbaren Auswertungen.

Voraussetzungen:

- Die Dump-Datei (bzw. das aktive System als Dump-Objekt) muss geöffnet sein.
- Die System-Symbolbibliothek muss verfügbar sein und die Elemente NDM, NDMNUC und DRV für die auszuwertende Systemversion enthalten. Andernfalls ist eine entsprechende Bibliothek bei der Modifikation der Source NDM (siehe unten) anzugeben.
- Die System-PRODAMP-Bibliothek ($TSOS.SYSDMP.DAMP) muss die NDMDAMP-PRODAMP-Objekte sowie die Source NDM enthalten. Andernfalls sind sie folgendermaßen zuzuweisen:

  ```
  ASSIGN-PRODAMP-LIBRARIES SOURCE-LIBRARY = <prodamp-bibliothek>,
  ASSIGN-PRODAMP-LIBRARIES OBJECT-LIBRARY = <prodamp-bibliothek>
  ```

- Die logische Systemdatei SYSLST kann, falls gewünscht, einer Datei zugewiesen werden.

Die Einstellungen der Optionen müssen entsprechend der folgenden Beschreibung explizit in die PRODAMP-Prozedur NDM eingetragen werden.

- Nachladen von PRODAMP:

  ```
  START-PRODAMP-EDITOR [kurz PROC] <fenster-nr> (4-9 bzw. 21-99)
  ```

- Einlesen der Source NDM:
  - Überschreiben „(procedure/index-identifier)“ mit „NDM“
  - Setzen von MODE (Feld hinter „W<fenster-nr>“) von „Dsp“ auf „Rea“
  - Abschicken mit DUE
- Modifizieren der Source NDM:

  Die Source NDM besteht im Wesentlichen aus dem Aufruf der PRODAMP-Prozedur NDM_MAIN mit entsprechenden Parametern für die Symbolbibliothek und den einzelnen Einheiten.
  Im PRODAMP-Fenster werden die gewünschten Werte aus dem angegebenen Wertebereich eingetragen. Dabei dürfen die Längen der einzelnen Parameter, die durch die Positionen der Hochkommata vorgegeben sind, nicht verändert werden.
  Die Bedeutung der einzelnen Parameter und der möglichen Werte ist in der Source auf den Folgeseiten erläutert. Mit „+“ oder „-“ kann im PRODAMP-Fenster geblättert werden.
- Übersetzen und Ausführen der Source NDM:
  - Setzen MODE=Go
  - Abschicken mit DUE

  Die Source mit den eingetragenen Parametern wird übersetzt und ausgeführt.

6.1.3 Aufruf von vorgefertigten ENTER-Jobs

Die LMS-Bibliothek SYSENT.NDMDAMP.<ver> wird mit folgenden ENTER-Jobs ausgeliefert, die unter der Benutzerkennung TSOS ablaufen:

ENTER-Job	Auswertung	Dateiname	Restart möglich
STD	Normal Keine Verwendung von EDT und Jobvariablen	SYSLST.NDMDAMP	-
NDM	Normal Zur Erzeugung von Datum und Uhrzeit werden Jobvariablen ($SYSJV.DATUM, $SYSJV.TIME) verwendet.	SYSLST.NDMDAMP.<date>.<time>	x
NDMALL	Maximal	SYSLST.NDMDAMP.<date>.<time>	x
NDMTRACE	Trace-Daten von NDM und BAVOLMON	SYSLST.NDMDAMP.<date>.<time>	x
DRV	DRV und Traces	SYSLST.NDMDAMP.<date>.<time>	x
DRVALL	alle DRV-relevanten Teile	SYSLST.NDMDAMP.<date>.<time>	x

Tabelle 16: Vorgefertigte ENTER-Jobs für NDMDAMP

Für alle ENTER-Jobs, bei denen ein Restart möglich ist, muss der EDT zur Verfügung stehen.

Die ENTER-Jobs verwenden nicht SDF-P. Sie verlangen die Standard-Dateinamen von DAMP, z.B. die Symbol-Dateien in $TSOS.SYSSMB.DAMP. Wurden bei der Installation von DAMP nicht die Standard-Namen vergeben, müssen sie angepasst werden.

Anstatt der ENTER-Jobs kann auch die Prozedur SYSSPR.NDMDAMP.<ver> im Batch benutzt werden, wenn sie mit dem Kommando ENTER-PROCEDURE aufgerufen wird. Dabei muss mindestens der Operand OBJECT=*SYSTEM angegeben werden.

6.2 Fehlerbehandlung bei der Auswertung

Nach einem Abbruch der Auswertung durch einen Fehler in NDMDAMP, kann ein Restart beim nächsten Teilschritt erfolgen. Für den Restart muss EDT verfügbar sein, da alle Daten, die zum Wiederaufsetzen nach der Abbruchstelle erforderlich sind, in der EDT-Arbeitsdatei 9 hinterlegt sind.

Der EDT ist auch erforderlich, wenn zur Diagnose von NDMDAMP-Fehlern der PRODAMP-Trace eingeschaltet wird. Die Trace-Daten zum NDMDAMP-Ablauf werden in der EDT-Arbeitsdatei 8 hinterlegt.
Sie werden am Ende der Auswertung in die Datei NDMDAMP.TRACE übertragen, falls kein Restart erfolgte. Der Inhalt der EDT-Arbeitsdatei 8 wird anschließend gelöscht.

Bei einem Restart und eingeschaltetem PRODAMP-Trace werden die in der EDT-Arbeitsdatei 8 hinterlegten Daten in die Datei NDMDAMP.TRACE.RESTART.<break#> übertragen. Der im Dateinamen angegebene Teilstring <break#> bezeichnet die interne Nummer derjenigen PRODAMP-Prozedur, bei der die Auswertung abgebrochen wurde.
Um eine Zuordnung von verschiedenen Restarts und PRODAMP-Traces zu bekommen, wird auch in der SYSLST-Ausgabe eine entsprechende <break#> an der Abbruchstelle angegeben.
Die nach dem letzten Restart bis zum Ende der Auswertung erzeugten Trace-Daten werden in der Datei NDMDAMP.TRACE hinterlegt.

Da die Trace-Dateien (NDMDAMP.TRACE, NDMDAMP.TRACE.RESTART.<break#>) sehr umfangreich werden können, kann es beim Abspeichern der Trace-Daten zu Fehlern kommen, wenn nicht genügend Speicherplatz zur Verfügung steht.
NDMDAMP bricht dann die weitere Auswertung ab.

6.3 Installation

NDMDAMP wird mit IMON installiert.

Bei fehlerhafter Installation kann NDMDAMP nicht korrekt ablaufen. Im Folgenden sind die häufigsten Fehler und deren Behebung zusammengefasst:

- DAMP meldet, dass ein PRODAMP-Objekt nicht gefunden wird (DMP4002).

  Bezieht sich die Fehlermeldung auf das Objekt NDM oder NDM_RESTART, so ist die PRODAMP-Bibliothek von NDMDAMP nicht in die allgemeine Prodamp-Bibliothek ($TSOS.SYSDMP.DAMP) eingemischt.
- NDMDAMP meldet „Modul NKATSOSM nicht gefunden“ und bricht ab.

  In diesem Fall wird entweder das Dump-Objekt nicht unterstützt oder die Symbolbibliothek für BS2000 ist nicht korrekt eingespielt.
- NDMDAMP meldet „Keine Symboldatei zuweisbar“ und bricht ab.

  In diesem Fall sind die Symbolelemente für NDM (NDMNUC und DRV , je nach gewähltem Umfang) nicht in die allgemeine bzw. explizit angegebene Symbol-Bibliothek eingemischt.

6.3.1 Release Items von NDMDAMP

LOGICAL-ID	Beschreibung	Default-Name
SYSSDF.USER	Syntaxdatei mit Kommando START-NDM-DIAGNOSIS	$.SYSSDF.NDMDAMP.<ver>.USER
SYSSPR	S-Prozedur	$.SYSSPR.NDMDAMP.<ver>
SYSENT	ENTER-Job	$.SYSENT.NDMDAMP.<ver>
SYSDMP	PRODAMP-Prozeduren, die in die System-PRODAMP-Bibliothek von DAMP gebracht werden müssen	$.SYSDMP.NDMDAMP.<ver>
SYSSII	Enthält die Beschreibung der Release Items	$.SYSSSI.NDMDAMP.<ver>

Tabelle 17: Release Items von NDMDAMP

6.3.2 Von NDMDAMP verwendete Logical Units

NDMDAMP wertet nicht mehr die Logical Units von DAMP aus, sondern verwendet die von DAMP eingestellten Default-Namen:

- DAMP-Startprogramm

  Bei DAMP=*STD wird die Anweisung START-DAMP verwendet.

- Symbolbibliothek

  Bei der Einstellung SYMBOLS=*STD wird die System-Symbolbibliothek von DAMP verwendet ($TSOS.SYSSMB.DAMP).

- PRODAMP-Bibliothek

  Bei der Einstellung PRODAMP=*STD wird die System-PRODAMP-Bibliothek von DAMP verwendet ($TSOS.SYSDMP.DAMP).

7 ELFE SERSLOG-Datei aufbereiten und auswerten

Das Dienstprogramm ELFE bereitet den Inhalt einer SERSLOG-Datei bzw. aller zu einem Systemlauf gehörenden SERSLOG-Dateien auf.

Eine SERSLOG-Datei besteht aus einzelnen Datensätzen, die SERSLOG (siehe Kapitel „SERSLOG Software-Fehler in SERSLOG-Datei sicherstellen“) beim Auftreten eines Fehlerereignisses geschrieben hat.

Das Programm ELFE verschafft Ihnen einen Überblick über die protokollierten Fehlerereignisse, indem es eine Zusammenfassung der in der Datei enthaltenen Fehlerereignistypen und deren Häufigkeit erstellt und ausgibt. Näheres über bestimmte Merkmale solcher Fehlerereignisse (z.B. Fehlerereignistyp, verursachende TSN, Zeitpunkt des Fehlerereignisses) können Sie sich mit ELFE gezielt ausgeben lassen. Die zugehörigen Einträge können Sie sich entweder komplett oder in einer Kurzform auf dem Bildschirm oder (über SYSLST) auf dem Drucker ausgeben lassen.

Zu jedem Fehlereintrag ist auch eine allgemeine Beschreibung vorhanden, die den entsprechenden Fehlerereignistyp (Rectype) erläutert und Diagnosehinweise gibt (Ausgabe wahlweise (über SYSLST) auf Drucker oder Bildschirm).

ELFE verarbeitet jede SERSLOG-Datei, unabhängig davon, mit welcher Betriebssystemversion diese erstellt wurde. Bei der Verarbeitung darf die SERSLOG-Datei nicht aktiv sein.

7.1 Software- und Hardware-Voraussetzungen

Die Bibliothek mit den Beschreibungen und Diagnosehinweisen für SERSLOG-Dateien hat den Standardnamen

`SYSLNK.ELFE.190`	für SERSLOG-Dateien der BS2000 OSD/BC V10.0
`SYSLNK.ELFE.200`	für SERSLOG-Dateien der BS2000 OSD/BC V11.0
`SYSLNK.ELFE.210`	für SERSLOG-Dateien der BS2000 V21.0
`SYSLNK.ELFE.210`	Bibliothek enthält das LLM ELFE

> ¡ Es ist nicht möglich, im gleichen ELFE-Lauf zwei Bibliotheken auszuwerten. Die Zuweisung einer zweiten Bibliothek wird mit der Fehlermeldung `ELF0012` zurückgewiesen.

Unterstützte Terminal-Typen und Listing-Formate

ELFE unterstützt alle aktuellen Terminal-Typen.
Die letzte Zeile des Bildschirms ist für die Eingaben an das Programm bestimmt.
Die Daten werden auf SYSLST mit 66 Zeilen zu 132 Spalten pro Seite ausgegeben.

Speichern von SERSLOG-Dateien

SERSLOG-Dateien werden mit der logischen Blockgröße STD(1) erzeugt.

Verwendung von Alias-Namen

Der Benutzer kann mithilfe von ACS (Alias Catalog Service) Alias-Namen für SERSLOG-Dateien und für die Beschreibungsbibliothek der Rectyps definieren. Diese Alias-Namen können dem Programm ELFE übergeben werden.
Wird ein Alias-Name einer nicht existierenden SERSLOG-Datei zugewiesen, verwendet ELFE an Stelle des wirklichen Dateinamens den Alias-Namen, um z.B. Fehler zu kommentieren, die bei Bearbeitung dieser Datei auftreten.

Meldungen von ELFE

Die Meldungen von ELFE haben die Meldungsklasse ELF. Informationen über einzelne Meldungen erhalten Sie im laufenden Betrieb mit /HELP-MSG-INFORMATION.

7.2 Bedienung

Das Programm ELFE wird mit **/START-ELFE** aufgerufen.

Mit `OPEN dateiname` wählen Sie die SERSLOG-Datei aus und öffnen sie bzw. wählen einen bestimmten Systemlauf aus und öffnen dessen Dateien. Mit der Anweisung `LIBRARY` weisen Sie eine Auswertebibliothek zu. Anschließend können Sie die Fehlerereigniseinträge näher untersuchen. `STOP` bzw. `END` beendet das Programm.

Anweisung	**Wirkung**
C(ONT)	Bearbeiten eines Systemlaufs, dessen Hilfsdateien noch vorhanden sind
D(ISPLAY)	Anzeigen von Informationen am Bildschirm
E(ND)	ELFE beenden
H(ELP)	Kurzinformation über ELFE-Anweisungen auf dem Bildschirm ausgeben
K(EEP)	Hilfsdateien bei Programmende oder Wechsel des Systemlaufs erhalten
L(IBRARY)	Zuweisen einer Auswertebibliothek
O(PEN)	Öffnen von SERSLOG-Dateien
P(RINT)	Fehlereinträge ausdrucken
S(TOP)	ELFE beenden

Tabelle 18: Übersicht über die ELFE-Anweisungen

7.2.1 CONT SERSLOG-Datei bzw. Systemlauf weiter auswerten

Mit der Anweisung CONT können Sie die Auswertung einer SERSLOG-Datei bzw. eines Systemlaufs fortsetzen. Die Auswertung muss bereits vorher unter derselben Benutzerkennung durchgeführt worden sein. Außerdem müssen die dabei angelegten Hilfsdateien noch vorhanden sein (siehe Anweisung KEEP, "KEEP Hilfsdateien erhalten"). Die Anweisung CONT beendet ggf. die laufende Auswertung einer anderen SERSLOG-Datei bzw. eines anderen Systemlaufs.

Format

Operation	Operanden
C(ONT)	sss

Operandenbeschreibung

sss dreistellige dezimale Nummer des Systemlaufs

7.2.2 DISPLAY Fehlereinträge auf dem Bildschirm ausgeben

Mit der Anweisung DISPLAY können Sie sich aus der SERSLOG-Datei Informationen über das System oder Fehlereinträge auf dem Bildschirm ausgeben lassen. Die Fehlereinträge können Sie durch Angeben entsprechender Operanden nach verschiedenen Kriterien auswählen, z.B. Fehlerereignistyp, TSN oder Uhrzeit. Die ausgewählten Einträge können dann entweder komplett mit Fehlerumgebungsdaten oder als Kurzfassung mit nur den wichtigsten Angaben ausgegeben werden.

Format

Operation	Operanden
D(ISPLAY)	{ INFO | LOCMAP | FULL | ADDRESS=addresse | MODULE=modulname | DESCRIPT[,RECTYPE=rectype] | SHORT | SUMMARY | { ELSN={ elsn | elsn1-elsn2 } | [TSN=tsn] | [TID=tid] } [,...] [,SHORT] | TIMESTP=date1/time1:date2/time2 | RECTYPE=rectype }

Operandenbeschreibung

ADDRESS=addresse	gibt den Namen des Moduls aus, innerhalb dessen sich die angegebene Adresse befindet, sowie die Distanz zum Modulanfang. „adresse“ ist hexadezimal in einer Länge von 4 Byte anzugeben.
DESCRIPT	gibt Beschreibungen und Diagnosehinweise zu den angegebenen Fehlerereignistypen (Rectypes) aus. DESCRIPT sollte zusammen mit dem Operanden RECTYPE angegeben werden, sonst werden alle in der Beschreibungsbibliothek vorhandenen Beschreibungen ausgegeben. Der Operand DESCRIPT wird nur akzeptiert, wenn eine Beschreibungsbibliothek zugewiesen ist (siehe Anweisung LIBRARY, "LIBRARY Beschreibungsbibliothek zuweisen").
ELSN={ elsn | elsn1-elsn2 }	die auszugebenden Einträge sollen anhand der Error Log Sequence Number (ELSN) ausgewählt werden. Die Einträge wurden bei der Erstellung der SERSLOG-Datei fortlaufend nummeriert. „elsn“, „elsn1“ und „elsn2“ sind hexadezimal anzugeben. „elsn2“ muss größer sein als „elsn1“. Die Angabe „elsn1-elsn2“ wählt alle Einträge aus, deren ELSN zwischen diesen Grenzen (einschließlich der angegebenen Werte) liegt.
FULL	gibt alle Informationen über das System, die Location-Map der Module, alle Einträge nach aufsteigender ELSN geordnet und das SUMMARY aus.
INFO	gibt Informationen über das System (Version, Generierungsdatum, etc.) aus.
LOCMAP	die Liste der Systemmodule soll ausgegeben werden. Sie enthält zwei Teillisten, die eine nach Modulnamen geordnet, die andere nach Moduladressen.

MODULE=modulname	gibt die Anfangsadresse, die Länge und die Versionsnummer des angegebenen Moduls aus.
RECTYPE=rectype	Die auszugebenden Einträge werden anhand ihres Fehlerereignistyps (Rectype) ausgewählt. Ausgegeben werden sie in der Reihenfolge der ELSN. Sind nicht alle 7 Stellen angegeben, dann werden alle Einträge ausgegeben, deren Fehlerereignistyp mit der angegebenen Zeichenfolge anfängt (r[e[c[t[y[p[e]]]]]]).
SHORT	gibt die Kurzform der ausgewählten SERSLOG-Einträge aus. Der Operand SHORT kann mit allen Auswahl-Operanden kombiniert werden. Ist SHORT der einzige Operand, werden alle Einträge der SERSLOG-Datei in der Kurzform ausgegeben.
SUMMARY	gibt eine Liste aus, die alle in der aktuellen Datei bzw. im aktuellen Systemlauf enthaltenen Fehlerereignistypen und die Häufigkeit ihres Auftretens nennt.
TID=tid	die auszugebenden SERSLOG-Einträge sollen anhand der TID ausgewählt werden. Ausgegeben werden die Einträge in aufsteigender Reihenfolge der ELSN.

TIMESTP= date1/time1:date2/time2

Die auszugebenden Einträge sollen anhand der ELSN, die während des Zeitintervalls date1/time1 bis date2/time2 in die SERSLOGDatei geschrieben wurde, ausgewählt werden.
Die Ausgabe erfolgt in aufsteigender Reihenfolge der ELSN. Das Datum und die Zeit werden im SDF-Format angegeben. Der obere Grenzwert date2/time2 muss höher oder gleich dem unteren Grenzwert date1/time1 sein.

Beispiel

```
DISPLAY TIMESTP=2008-10-25/12:45:00:2008-10-25/14:00:00
```

TSN=tsn	Die auszugebenden Einträge sollen anhand der TSN ausgewählt werden; sie werden in der Reihenfolge der ELSN ausgegeben. Bei der Angabe von nichtnumerischen TSNs und gleichzeitig von weniger als 4 Zeichen wird links mit Leerzeichen aufgefüllt.

Beispiel

Die Anweisung

```
DISPLAY TSN=1234,RECTYPE=NRT,SHORT
```

bewirkt, dass die Kurzform aller SERSLOG-Sätze mit der TSN 1234 ausgegeben wird, deren RECTYPE mit der Zeichenfolge „NRT“ anfängt. Die Sätze werden nach der ELSN sortiert nach SYSOUT ausgegeben.

Übersteigt die auszugebende Datenmenge die Anzahl der Bildschirmzeilen, so wird jeweils ein Bildschirm ausgegeben und es erscheint am unteren Bildschirmrand die Meldung `ENTER '+' OR NEW COMMAND`.
Die Eingabe von „+“ oder eine Leereingabe (nur DUE) setzen dann die Ausgabe fort. Eine neue Anweisung wird ausgeführt und beendet die vorherige Ausgabe.

Die folgenden Anweisungen bewirken im Detail:

STOP/END:	Abbruch der aktuellen Anweisung, der Bearbeitung des Systemlaufs oder der Datei und des ELFE-Laufs.
OPEN/CONT:	Abbruch der aktuellen Anweisung und der Bearbeitung des Systemlaufs. Beginn mit dem angegebenen Systemlauf.
PRINT/DISPLAY:	Abbruch der aktuellen Anweisung und Beginn mit der Bearbeitung der angegebenen Anweisung.

Ausnahme

Die Anweisung

{ DISPLAY } { ADDRESS= | MODULE= }

und

die Anweisungen KEEP, HELP und LIBRARY
unterbrechen die Bearbeitung der laufenden Informationsausgabe lediglich für die Ausgabe der angeforderten Daten. Anschließend kann die davor begonnene Informationsausgabe fortgesetzt werden.

Alle anderen Operanden der Anweisung DISPLAY beenden die davor begonnene Informationsausgabe.

7.2.3 END ELFE beenden

Die Anweisung END beendet den Programmlauf und bricht die laufende Bearbeitung ab. Gleichzeitig werden die angelegten Hilfsdateien gelöscht, es sei denn, dies wurde vorher mit der Anweisung KEEP unterbunden.

Format

Operation	Operanden
E(ND)	

7.2.4 HELP Kurzinformation über ELFE-Anweisungen ausgeben

Mit der Anweisung HELP können Sie sich Kurzinformationen über eine der bei ELFE möglichen Anweisungen am Bildschirm ausgeben lassen. Wird HELP ohne Operand eingegeben, wird über alle Anweisungen informiert.

Format

Operation	Operanden
H(ELP)	[anweisung]

Operandenbeschreibung

anweisung Anweisung, über die Informationen ausgegeben werden sollen.

7.2.5 KEEP Hilfsdateien erhalten

ELFE arbeitet mit den Hilfsdateien S.SERSLOG.sss.ELSN, S.SERSLOG.sss.INFO, S.SERSLOG.sss.MODULE und S.SERSLOG.sss.ADDRESS. Diese Hilfsdateien werden normalerweise mit Beendigung des Programms ELFE gelöscht. Mit der Anweisung KEEP jedoch erreichen Sie, dass diese Hilfsdateien über das Programmende hinaus erhalten bleiben, etwa dann, wenn Sie die Arbeit unterbrechen müssen.

Ausnahme

Wurde bei der Anweisung OPEN entweder kein Parameter oder ein Dateiname angegeben, dann wird die Anweisung KEEP zurückgewiesen.

Format

Operation	Operanden
K(EEP)	

7.2.6 LIBRARY Beschreibungsbibliothek zuweisen

Mit der Anweisung LIBRARY weisen Sie eine Bibliothek mit Beschreibungen und Diagnosehinweisen zu den Fehlerereignisstypen zu.

Format

Operation	Operanden
L(IBRARY)	dateiname

Operandenbeschreibung

dateiname — Name der zuzuweisenden Bibliothek, siehe "Software- und Hardware-Voraussetzungen". Es ist nicht möglich, innerhalb eines ELFE-Laufes zwei Bibliotheken zuzuweisen.

Die Zuweisung einer weiteren Bibliothek wird mit der Fehlermeldung `ELF0012` zurückgewiesen.

7.2.7 OPEN Auszuwertende Datei zuweisen und öffnen

Mit der Anweisung OPEN legen Sie die auszuwertenden Dateien fest und öffnen sie. Dabei erzeugt ELFE die benötigten Hilfsdateien.
Sie können OPEN mit einem Dateinamen, mit der dreistelligen Nummer eines bestimmten Systemlaufs (ggf. zusammen mit der laufenden Nummer der Datei) oder ohne Operanden angeben. ELFE sucht sich dann die entsprechende Datei und öffnet sie. Diese muss ggf. auf der eigenen Benutzerkennung abgelegt sein.
Die Anweisung OPEN beendet eine bereits laufende Bearbeitung einer Datei bzw. eines Systemlaufs und leitet die Bearbeitung der angegebenen Datei bzw. des angegebenen Systemlaufs ein.

Format

Operation	Operanden
O(PEN)	[{ sss[,nn][,STD-NAME=NEW/OLD] | dateiname }]

Operandenbeschreibung

sss — Dreistellige dezimale Nummer des Systemlaufs.
Es werden alle unter der Aufrufer-Benutzerkennung vorhandenen Dateien mit dem Standardnamen SYS.SERSLOG.yyyy-mm-dd.sss.nn zur Auswertung herangezogen, wobei „sss“ die angegebene Nummer des Systemlaufs ist und „yyyy-mm-dd“ sowie „nn“ beliebig sind. Die Datumsangabe im Standardnamen befindet sich standardmäßig im neuen Format (siehe Operand STD-NAME).

nn — zweistellige dezimale laufende Nummer einer SERSLOG-Datei innerhalb eines Systemlaufs.
Werden sowohl „sss“ als auch „nn“ angegeben, so wird die unter der Aufrufer-Benutzerkennung vorhandene Datei mit dem Standardnamen SYS.SERSLOG.yyyy-mm-dd.sss.nn ausgewertet, wobei „sss“ die angegebene Nummer des Systemlaufs, „nn“ die angegebene laufende Nummer der Datei innerhalb des angegebenen Systemlaufs und „yyyy-mm-dd“ beliebig sind. Die Datumsangabe im Standardnamen befindet sich standardmäßig im neuen Format (siehe Operand STD-NAME).

STD-NAME=

- NEW — Voreinstellung. Die Datumsangabe im Dateinamen wird in der Form yyyy-mm-dd gewählt.
- OLD — Die Datumsangabe im Dateinamen wird in der Form yy.mm.dd gewählt.

dateiname — Name der auszuwertenden SERSLOG-Datei.

Sie können bei der Anweisung OPEN die Operanden weglassen, wenn Sie vor dem Aufruf des Dienstprogrammes ELFE eine Datei mit dem Linknamen SERSLOG in die Task-File-Tabelle (TFT) eintragen (Kommando ADD-FILE-LINK). ELFE öffnet dann die mit dem Linknamen SERSLOG verknüpfte Datei.

Fortsetzung der Bildschirmausgabe

Übersteigt die auszugebende Datenmenge die Anzahl der Bildschirmzeilen, so wird jeweils ein Bildschirm ausgegeben, anschließend erhält der Benutzer die Kontrolle zur Fortsetzung oder zum Abbruch der laufenden Option.
Die Eingabe von „+“ oder eine Leereingabe (nur DUE) setzen dann die Ausgabe fort.

Eingabemöglichkeiten und Auswirkung:

S/END:	Abbruch der aktuellen Anweisung, der Bearbeitung des Systemlaufs oder der Datei und des ELFE-Laufs.
O/C:	Abbruch der aktuellen Anweisung und der Bearbeitung des Systemlaufs. Beginn mit dem angegebenen Systemlauf.
P/D:	Abbruch der aktuellen Anweisung und Beginn mit der Bearbeitung der angegebenen Anweisung. Wurde während eines Öffnungsprozesses ein Kommando eingegeben und akzeptiert, wird die Sitzung als eröffnet betrachtet. Die Auswertung erfolgt mit den bisher eröffneten Dateien.
H/L	Unterbrechung der aktuellen Anweisung zur Bearbeitung dieser Aufforderung. Anschließend wird die ursprüngliche Anweisung weiterbearbeitet.
+	Fortsetzung der aktuellen Anweisung.

Parallelaufrufe

Wird ELFE parallel aufgerufen, kann es wegen gleicher Hilfsdateinamen zum DMS-Fehler 05B1 kommen. Das Problem kann durch Angabe der Nummer des Systemlaufs bei der Anweisung OPEN umgangen werden (OPEN 003,STD-NAME=NEW).

7.2.8 PRINT Fehlereinträge ausdrucken

Mit der Anweisung PRINT können Sie sich Informationen über das System oder Fehlereinträge aus der SERSLOG-Datei (über SYSLST) auf Drucker ausgeben lassen.
Durch Angeben entsprechender Operanden können Sie die Informationen nach verschiedenen Kriterien auswählen, z.B. Fehlerereignistyp, TSN oder Uhrzeit. Die ausgewählten Einträge können dann entweder komplett mit Fehlerumgebungsdaten oder als Kurzfassung mit nur den wichtigsten Angaben ausgegeben werden.

Format

Operation	Operanden
P(RINT)	{ INFO | LOCMAP | FULL | ADDRESS=addresse | MODULE=modulname | DESCRIPT[,RECTYPE=rectype] | SHORT | SUMMARY | { ELSN={ elsn | elsn1-elsn2 } | [TSN=tsn] | [TID=tid] } [,...] [,SHORT] | TIMESTP=date1/time1:date2/time2 | RECTYPE=rectype }

Die Operanden der Anweisung PRINT sind identisch mit denen der Anweisung DISPLAY (siehe "DISPLAY Fehlereinträge auf dem Bildschirm ausgeben").

7.2.9 STOP ELFE beenden

Die Anweisung STOP beendet den Programmlauf und bricht die laufende Bearbeitung ab. Gleichzeitig werden die angelegten Hilfsdateien gelöscht, es sei denn, dies wurde vorher mit der Anweisung KEEP unterbunden.

Format

Operation	Operanden
S(TOP)	

8 SERSLOG Software-Fehler in SERSLOG-Datei sicherstellen

Das Software-Error-Logging von BS2000 besteht aus zwei Teilen, der Sicherstellung von Daten über auftretende Software-Fehler und der Aufbereitung dieser Daten. Das Sicherstellen von Daten über auftretende Software-Fehler geschieht mit der Betriebssystemfunktion SERSLOG. Dabei werden bestimmte Daten über aufgetretene Software-Fehler in eine Datei, die SERSLOG-Datei, geschrieben. Um die Leistungsfähigkeit des Betriebssystems nicht zu beeinträchtigen, werden die Daten dabei nicht weiter aufbereitet. Mit dem Dienstprogramm ELFE (Error Log File Evaluation, "ELFE SERSLOG-Datei aufbereiten und auswerten") können Sie diese Daten aufbereiten und auswerten.

Die nachfolgende Übersicht enthält alle Kommandos, mit denen Operator und Systemverwalter das Software-Error-Logging beeinflussen können. Die Kommandos sind im Handbuch „Kommandos“ [8] detailliert beschrieben.

Kommandos	Anwendung
CHANGE-SERSLOG-FILE	Schließt die aktuelle SERSLOG-Datei und eröffnet eine neue
SHOW-SERSLOG-STATUS	Informiert über Zustand des Error-Logging und Namen der SERSLOG-Datei
START-SERSLOG	Startet Software-Error-Logging und eröffnet eine SERSLOG-Datei
STOP-SERSLOG	Schließen der SERSLOG-Datei

Tabelle 19: Übersicht über die SERSLOG-Kommandos

SERSLOG-Datei

Eine SERSLOG-Datei besteht aus einzelnen Datensätzen, die SERSLOG beim Auftreten eines Fehlerereignisses geschrieben hat. Jeder Datensatz besteht aus Error Log Sequence Number (ELSN), Bezeichnung des Fehlerereignistyps (Rectype), TSN, TID, Bezeichnung des Moduls, der diesen Eintrag ausgelöst hat, Uhrzeit des Fehlerereignisses sowie Daten aus der Umgebung des Software-Fehlers.

Sie wird während des System-STARTUP unter der Benutzerkennung $TSOS eröffnet, wenn das Software-Error-Logging aktiviert wird. Der Name der SERSLOG-Datei bildet sich nach folgendem Schema:

`SYS.SERSLOG.yyyy-mm-dd.xxx.nn` bzw. `SYS.SERSLOG.yy.mm.dd.xxx.nn` (je nach Einstellung des Systemparameters FMTYFNLH, siehe hierzu Handbuch „Einführung in die Systembetreuung“ [6]).

`yyyy-mm-dd`	Datum der Dateieröffnung
`xxx`	Nummer des zugehörigen Systemlaufs
`nn`	laufende Nummer der SERSLOG-Datei (01 bis 99, beginnt beim STARTUP immer mit 01). Wird nn größer als 99, dann wird der Zähler wieder auf 01 gesetzt, wobei die erste SERSLOG-Datei überschrieben wird.

Die SERSLOG-Datei hat keinen Schreibschutz.

Bei Beendigung des Systemlaufs wird die SERSLOG-Datei geschlossen und das Software-Error-Logging beendet. Die aktuelle SERSLOG-Datei wird von SLED mit ausgegeben.

Nur Operator und Systemverwalter können das Software-Error-Logging beeinflussen, d.h. aktivieren (START-SERSLOG), deaktivieren (STOP-SERSLOG) und wechseln der SERSLOG-Datei (CHANGE-SERSLOG-FILE). Mit dem Kommando SHOW-SERSLOG-STATUS können Informationen über das Software-Error-Logging angefordert werden.

9 ASE Auxiliary SERSLOG Extensions

Das Subsystem ASE (Auxiliary SERSLOG Extensions) ermöglicht eine automatische Überwachung von kritischen Systemzuständen, die sich in SERSLOG-Ereignissen widerspiegeln. Für diese Ereignisse können Schwellwerte festgelegt werden, bei deren Überschreitung die Ereignisse auf eine der folgenden Arten protokolliert werden: in einen internen Puffer, durch Meldung an der Konsole und/oder via Remote Service. Diese Protokollierung kann auf ausgewählte SERSLOG-Ereignisse eingeschränkt werden.

Die nachfolgende Übersicht enthält alle Kommandos, mit denen der Systemverwalter das Software-Error-Logging beeinflussen kann. Die Kommandos sind im Handbuch „Kommandos“ [8] detailliert beschrieben.

Kommando	**Funktion**
ADD-ASE-ELEMENT	deklariert ein ASE-Element
MODIFY-ASE-PARAMETERS	ändert globale ASE-Einstellungen
REMOVE-ASE-ELEMENT	löscht ASE-Elemente
SHOW-ASE-ELEMENT	zeigt ASE-Elemente an
SHOW-ASE-LOGGING	zeigt Daten des internen ASE-Loggings an
SHOW-ASE-PARAMETERS	zeigt globale ASE-Einstellungen an
SHOW-ASE-STATUS	liefert ASE-Statusinformation

Tabelle 20: Übersicht über die ASE-Anweisungen

10 SLED-Dump

Wenn Ursache und Wirkung eines Software-Fehlers nicht eingegrenzt werden können oder ein unverzichtbarer Teil des Betriebssystems beeinträchtigt ist, dann muss das Betriebssystem beendet und die für die Diagnose wichtigen Speicherbereiche von BS2000 sichergestellt werden. Einen solchen Gesamtdump erstellt das Programm SLED (Self Loading Emergency Dump).

SLED arbeitet unabhängig vom Betriebssystem BS2000.
Nach dem Ablauf von SLED muss das Betriebssystem neu geladen werden.

SLED kann automatisch ablaufen (unbedienter Betrieb) oder mit Steuerung durch den Operator (bedienter Betrieb).

Der Ablauf von SLED kann über eine SLED-Parameterdatei gesteuert werden.

SLED schreibt eine Dumpdatei (SLEDFILE) auf Platte oder Band. Diese Datei enthält alle angeforderten und erreichbaren Daten, die zur späteren Auswertung durch das Aufbereitungsprogramm DAMP erforderlich sind.

Wurde SLED als „DUMP vom SLED“ geladen, so werden die von IPL-EXEC und SLED benutzten Speicherbereiche ausgegeben.

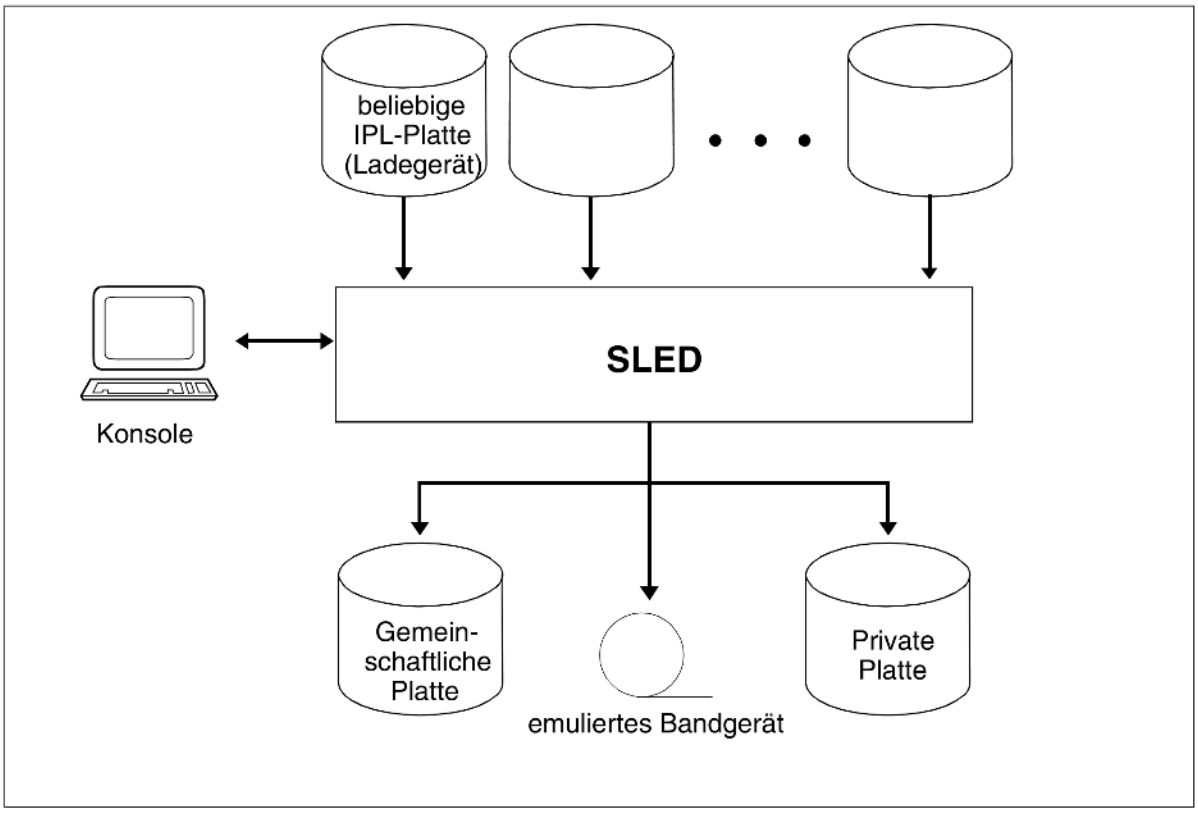


Bild 100: Gerätekonfiguration für SLED

10.1 Laden und Initialisieren von SLED

SLED läuft als Anwendungsprogramm unter dem IPL-EXEC. Bevor SLED geladen wird, wird zuerst das IPL-EXEC geladen und initialisiert.

Bei diesen Ladevorgängen muss berücksichtigt werden, ob es sich um den ersten SLED nach einem abgebrochenen Systemlauf, um eine SLED-Wiederholung oder um einen Dump vom SLED handelt.

Vor dem ersten SLED-Laden sind die Speicherbereiche zu sichern, die von BOOT, IPL und SLED benutzt werden. Dies geschieht teilweise per Firmware, indem Datenbereiche in Sicherstellungsbereiche im Speicher oder in den Serviceprocessor kopiert werden, zum größeren Teil per Software, indem Datenbereiche auf die IPL-Platte in eine Sicherungsdatei geschrieben werden (SLEDSAVE bzw. BOOTSAVE), bevor sie von BOOT, IPL oder SLED benutzt werden.

Die SLED-Einleitung ist vom Server-Typ abhängig. Sie ist ausführlich in den Handbüchern für die einzelnen Server-Typen beschrieben.

Für den Ladevorgang werden folgende Dateien auf der Ladeplatte gesucht, die im SVL der Platte mit SIR verankert sein müssen:

- $TSOS.SYSPRG.BOOT.DSKnnn.SAVE (BOOTSAVE)
- $TSOS.SYSPRG.IPL.DSKnnn (IPL-Phase)
- $TSOS.SYSREP.IPL.DSKnnn (Korrekturen für den IPL)
- $TSOS.SYSREP.SLED.DSKnnn (Korrekturen für den SLED)
- $TSOS.SYSPRG.SLED.DSKnnn.SAVE (SLEDSAVE)

Vor dem Laden des SLED sollten bereits alle im weiteren Verlauf benötigten Platten bereitgestellt und online geschaltet werden.

Wenn SLED geladen und gestartet ist, werden zunächst einige Konsistenzprüfungen durchgeführt. Es wird geprüft:

- ob die Version des geladenen SLED mit der Version des ladenden IPL übereinstimmt;
- welches System zuvor geladen war und falls es sich um BS2000 handelt (auch unter VM2000 oder beim DUMP vom SLED), ob dessen Version mit der Version des SLED übereinstimmt;
- ob ein nicht sichergestellter Teil des Hauptspeichers überschrieben wurde.

SLED führt diese Konsistenzprüfungen unabhängig davon durch, ob der Speicherauszug im bedienten oder unbedienten Betrieb abläuft. Unbedient heißt, dass der SLED über Vorgaben in der SLED-Parameterdatei oder über Standard-Vereinbarungen automatisch, ohne Eingriffe des Operators abläuft. Bedient heißt, dass der SLED den Operator zur Eingabe oder Korrektur von Optionen auffordert und der SLED im Dialog abläuft.

Entsprechende Warnungen über den Ablauf der Konsistenzprüfung werden auf der Konsole ausgegeben und in der SLED-Ausgabedatei protokolliert.

Die Ausgabe der Diagnosedaten kann entweder auf gemeinschaftliche oder Privatplatten, oder auf Band erfolgen.

Im bedienten SLED muss folgende Meldung beantwortet werden:

```
NSD1003 STANDARD SLED ? REPLY (Y; N; EOT=Y)
```

Standard-SLED

Mit der Antwort Y bzw. <EOT> auf die Meldung NSD1003 wählt der Operator den Standard-SLED aus. Die Antwort zieht folgendes Standard-Verhalten von SLED nach sich:

1. Die Platten des Home-Pubsets und die Paging-Platten werden im bedienten wie im unbedienten Betrieb auf Verfügbarkeit und Zustand überprüft (Online-Scan). Nicht vorhandene (offline-) Platten werden über die Meldung NSD1400 protokolliert. Werden im weiteren Verlauf Daten von den fehlenden Platten benötigt, so muss nach dem Zuschalten dieser Platten SLED erneut geladen werden; das Zuschalten benötigter Platten während des SLED-Laufes ist nicht möglich.

   Je mehr Pubsets der SLED zu beachten hat (den Home-Pubset des zu dumpenden Systems, den Lade-Pubset des SLED und evtl. noch einen Pubset für die Parameterdatei und einen für die SLED-Ausgabedatei), desto länger dauert der Online-Scan.
2. Die Standardwerte für die (im nachfolgenden beschriebenen) Parameter MODE und TASK werden als EOT initialisiert. Unabhängig davon, ob ein Systemabsturz vorausgegangen ist, wählt SLED die Einstellung MODE=STD (siehe Seite "Ausgabe in eine Dumpdatei") und TASK=STD (siehe Seite "Ausgabe in eine Dumpdatei").

   Nach Abschluss der SLED-Initialisierungsphase ist bekannt, ob
   - alle Platten des Home-Pubsets des abgebrochenen Systemlaufs online sind
   - gemeinschaftliche Platten aus verschiedenen Pubsets online sind
   - alle beim abgebrochenen Systemlauf verwendeten Paging-Platten online sind

Der Operatordialog im Standard-SLED wird mit der Meldung NSD5200 (Zuweisen einer Parameterdatei) fortgesetzt.

Non-Standard-SLED

Mit der Antwort N auf die Meldung NSD1003 wählt der Operator den Non-Standard-SLED aus. Er wünscht einen erweiterten Dialog mit SLED zur Steuerung des Ablaufs und erhält die folgenden zusätzlichen Meldungen:

```
NSD0900 ONLINE SCAN ? REPLY (Y; N; IPL-CONF=I; GENERAL ONLINE SCAN=X; EOT=Y)
```

SLED fragt, ob ein Online-Scan, d.h. eine Überprüfung der erreichbaren Gerätekonfiguration, durchgeführt werden soll.

Y IPL führt einen Online-Scan für jeden benötigten Pubset durch.
Das Verhalten entspricht dem Standard-SLED.

N IPL soll keinen Online-Scan durchführen.
In diesem Fall steht nur die SLED-Ladeplatte zur Verfügung. Der Operator sollte diesen Wert nur dann wählen, falls SLED mit Online-Scan oder der IPL-CONF-Auswertung nicht ablauffähig sein sollte.

I Statt der Durchführung eines Online-Scans wird die System-spezifische Partition in der Datei $TSOS.SYSDAT.IPL-CONF.<ver>, in der die aktuelle Konfiguration für die Systemeinleitung hinterlegt ist, ausgewertet. Ist die Partition in der Datei nicht vorhanden oder treten bei der Verarbeitung Fehler auf, wird wiederum ein Online-Scan für die benötigten Pubsets angestoßen. Nach erfolgreicher Abarbeitung kennt SLED den Home-Pubset, die Paging-Platten und die SLED-Ladeplatte, wenn die zugehörigen Platten bereits beim Hochfahren des Systems zugeschaltet waren und damit in der IPL-CONF-Datei eingetragen werden konnten.

X IPL führt einen Online-Scan für alle Platten durch.

> ℹ Der allgemeine Online-Scan ist vom SLED nicht für große Systeme verwendbar, da nicht mehr als max. 1290 Plattengeräte-Eintragungen verwaltet werden können. Diese Option sollte nur in Sonderfällen gewählt werden.

Nach der Beantwortung erscheinen zwei weitere Meldungen, auf die der Operator antworten muss. Die Antworten entscheiden über den Umfang der SLED-Datei.

```
NSD3001 SPECIFY NOEDIT MODE.
        REPLY (STD; NSF; REAL; ALL; EOT=STD; - (BACKTRACK))
```

Beschreibung dieser Meldung siehe "Ausgabe in eine Dumpdatei".

```
NSD3002 SELECT TASKS.
        REPLY (STD; NONE; ALL; (TSN LIST); EOT=STD; - (BACKTRACK))
```

Beschreibung dieser Meldung siehe "Ausgabe in eine Dumpdatei".

Fehlerfälle für Standard- und Non-Standard-SLED

Steht die Sysres (Systemplatte) nicht zur Verfügung, kann SLED zwar ablaufen, aber es können keine Systemdateien (TSOSCAT, Logging-Dateien, usw.) und eventuell keine Daten des Seitenwechselbereichs gesichert werden.

Steht der Home-Pubset nur teilweise zur Verfügung, so können einige Systemdateien möglicherweise nur unvollständig gesichert werden.

Nicht verfügbare Paging-Platten führen ebenfalls zu unvollständigen Diagnoseunterlagen.

SLED ermittelt das Fehlen einer der benötigten Platten und protokolliert dies mit der Meldung `NSD1400`. Der Operator kann dann die fehlenden Platten zuschalten und den SLED-Lauf wiederholen.

SLED-Wiederholung

Von einer SLED-Wiederholung spricht man, wenn nach einem SLED-Lauf erneut ein SLED geladen und gestartet wird, um den Dump des zuvor abgebrochenen Systemlaufs zu erzeugen. Dies kann z.B. erforderlich sein, wenn aus Versehen zunächst ein SLED geladen wurde, der mit der Version des abgebrochenen Systems nicht kompatibel ist, oder wenn während der SLED-Initialisierung benötigte Platten nicht verfügbar (online) waren.

Bei einer SLED-Wiederholung müssen also die in BOOTSAVE und SLEDSAVE sichergestellten Bereiche weiterverwendet und dürfen nicht erneut gesichert werden.

Sowohl Firmware als auch Software versuchen, eine SLED-Wiederholung zu erkennen, und unterdrücken in diesem Fall das Sichern der Datenbereiche. Dadurch gehen bei einer Dump-Wiederholung keine Daten verloren. Wenn der erneut geladene SLED nicht zur selben Betriebssystemversion gehört wie der erste SLED, kann es passieren, dass die SLED-Wiederholung nicht erkannt wird und in diesem Fall ein Teil der Diagnosedaten verloren geht.

Dump vom SLED

Wenn bei einem SLED-Lauf ein Fehler auftritt (Meldung `NSD1002`), kann es erforderlich sein, einen Dump vom SLED zu erstellen: es wird erneut ein SLED geladen und gestartet, um damit Informationen über den fehlerhaften SLED zu erzeugen.

Obwohl also bereits ein SLED geladen war, muss in diesem Fall die Sicherstellung der Speicherbereiche, in die der SLED geladen wird, erneut durchgeführt werden, da diese Daten für die SLED-Diagnose benötigt werden. Sowohl in der Firmware als auch in der Software muss also folgendes berücksichtigt werden:

- Bei einer SLED-Wiederholung ist die Sicherstellung der von BOOT, IPL und SLED überschriebenen Daten bereits erfolgt und wird **nicht** erneut durchgeführt.
- Bei einem Dump vom SLED wird die Sicherstellung der von BOOT, IPL und SLED überschriebenen Daten (SLED-Daten) **jetzt** erneut durchgeführt.

Deshalb sind besondere Vorkehrungen zu treffen, wenn ein Dump über den Ablauf der Dump-Funktion (Dump vom SLED) erstellt werden muss:

- Im VM2000-Betrieb erfolgt dies beim Start der VM (`/START-VM`) durch die Angabe des Parameters `UNLOCK-SAVEAREA=*YES`
- Auf x86-Servern erfolgt dies bei Systemstart mit `ipl parameter [d|u]: u` (UNLOCK)
- Bei /390-Servern müssen dafür verschiedene Aktionen durchgeführt werden. Die Auswahl der Aktionen ist abhängig vom Server. Zur ausführlichen Beschreibung siehe die Handbücher für die einzelnen Server-Typen.

  Zu den Aktionen können z.B. gehören:
  - CPU stoppen
  - Registerinhalte protokollieren
  - Adress-Stop auf real X'4000' setzen
  - bei Multiprozessor-Systemen START/STOP-Modus auf `TARGET CPU` setzen
  - Start der Dump-Funktion
  - nachdem der Adress-Stop wirksam wird, die reale Speicherstelle X'1800' mit X'00' überschreiben
  - Adress-Stop wieder zurücksetzen
  - CPU starten
  - weiterer Ablauf wie Dump-Funktion
  - nach beendeter Dump-Funktion START/STOP-Modus zurücksetzen

Funktionsauswahl

Die Auswahl des Ausgabemediums erfolgt durch Beantwortung folgender Meldung:

```
NSD3000 SPECIFY OUTPUT DEVICE.
        REPLY (DPUB; DPRIV; TAPE; EOT=DPUB; - (BACKTRACK))
```

Mögliche Antworten:

`DPUB`	Ausgabe auf gemeinschaftliche Platte (Standardwert)
`DPRIV`	Ausgabe auf Privatplatte
`T[APE]`	Ausgabe auf Band

10.2 Ausgabe in eine Dumpdatei

SLED erstellt eine Dumpdatei (SLEDFILE), die mit Hilfe des Dump-Auswerters DAMP aufbereitet und ausgewertet werden kann.

Bestimmung der Ausgabedaten

Der Umfang der Ausgabedaten, die in die SLEDFILE geschrieben werden sollen, wird durch die Parameter MODE (als Antwort auf die Meldung `NSD3001`) und TASK (als Antwort auf die Meldung `NSD3002`) festgelegt. Der Parameter MODE bestimmt die Auswahl der Speicherseiten, die in die SLEDFILE aufgenommen werden sollen. Der Parameter TASK bestimmt die Tasks, deren Adressräume gesichert werden sollen.

Die Parameter MODE=ALL und TASK=ALL werden automatisch eingestellt, wenn eine der folgenden Bedingungen zutrifft:

- der Hauptspeicher kleiner als 128 MB ist
- die Systemtabellen für die Seitenauswahl zerstört sind
- die Produkt-ID oder das Dumptestament einen Fehler enthält

> **i** Die Parameter MODE und TASK können beim Standard-SLED (d.h. automatischer SLED oder die Antwort auf `NSD1003` ist `EOT` oder `Y`) nur durch Voraus-Eingabe oder über die Parameterdatei angegeben werden. Werden die beiden Parameter nicht angegeben, legt SLED die Werte selber fest (implizite `EOT`-Antwort).
> Es wird empfohlen, die Parameter MODE und TASK nicht anzugeben und somit die Festlegung dieser Werte dem SLED zu überlassen.

Seitenauswahl mit dem Parameter MODE

```
NSD3001 SPECIFY NOEDIT MODE.
        REPLY (STD; NSF; REAL; ALL; EOT=STD; - (BACKTRACK))
```

Abhängig von der Antwort werden folgende Seiten vom Hauptspeicher und dem Seitenwechselspeicher (Paging-Area) ausgegeben:

`EOT` (keine Eingabe)

Der Wert für diesen Parameter wird von SLED bestimmt.
Unabhängig davon, ob zuvor ein Systemabsturz (SETS) vorlag, wählt SLED `MODE=STD`. Bei Systemabsturz ist normalerweise gewährleistet, dass die verursachende Task oder das verursachende Modul im Umfang der Ausgabe enthalten ist.

`STD` Es werden folgende Seiten ausgegeben:

- Seiten der privilegierten Data Spaces
- Klasse-1- bis Klasse-4-Speicherseiten (Systemadressraum)
- Klasse-5-Speicherseiten aller ausgewählten Tasks
- Klasse-6-Speicherseiten aller ausgewählten System- und SVC79-Tasks
- Klasse-6-Speicherseiten aller in der TSN-Liste angegebenen Tasks
- residente[*)] Klasse-6-Speicherseiten aller TIC (Task in Control)

- residente*) Klasse-6-Speicherseite 0 aller ausgewählten Tasks

*)„resident“ bedeutet in diesem Zusammenhang, dass sich die Seite im Hauptspeicher befindet.

`NSF` (No System Files)

Wie `STD`, jedoch ohne die Systemdateien, die bei `STD` zusätzlich gesichert werden, wenn diese Daten erreichbar sind.

`REAL` Alle Hauptspeicherseiten (der nachfolgende Parameter TASK wird ignoriert), es werden keine im Seitenwechselspeicher befindlichen Daten gesichert.

`ALL` Zusätzlich zu den bei `MODE=STD` ausgewählten Seiten wird der gesamte Hauptspeicher ausgegeben.

> ¡ Der SLED-Dump kann bei `MODE=ALL` einen sehr großen Umfang erreichen!

Taskauswahl mit dem Parameter TASK

Zusätzlich zum System-Adressraum (Klasse 1 bis 4) werden in Abhängigkeit vom Wert des Task-Parameters die Adressräume der angegebenen Tasks gesichert.

```
NSD3002 SELECT TASKS.
        REPLY (STD; NONE; ALL; (TSN LIST); EOT=STD; - (BACKTRACK))
```

`EOT` (keine Eingabe)

Der Wert für diesen Parameter wird von SLED bestimmt.
Unabhängig davon, ob zuvor ein Systemabsturz (SETS) vorlag, wählt SLED `MODE=STD.`

`STD` Erfasst werden:

- alle TIC (Task in Control)
- alle Systemtasks
- alle privilegierten (SVC-79-)Tasks
- alle CDUMP-In-Progress-Tasks und Dump-Tasks
- alle Tasks in der Warteschlange Q10 (Permanently Pended)
- alle Tasks aus den TIC-Trace-Tabellen (das sind die letzten 64 Tasks, die einer logischen Maschine zugeteilt waren)

`NONE` Alle TIC (Task in Control) einer CPU

`ALL` Alle Tasks

`<tsn1>,<tsn2>,...,<tsn8>`

Zusätzlich zu den unter `STD` aufgezählten Tasks werden die in dieser Liste angegebenen Tasks (maximal 8) im Dump gesichert.

Inhalt der SLEDFILE bei MODE = STD/ALL

1. STATUS-Abschnitt (CPU-Status)

2. MAINMEM-Abschnitt: ausgewählte Hauptspeicherseiten
3. HSA-Abschnitt (nur bei /390-Servern, die native betrieben werden)
4. VM2HYPVS-Abschnitt (VM2000-Hypervisor auf /390-Servern, falls ein SLED in einem VM2000-Gastsystem erstellt wird)
5. IOHIOSDP (Bus-Dumpdatei); nur auf x86-Servern
6. FIRMWARE-Abschnitt: Firmware-Code und -Daten (nur bei x86-Servern)
7. PAGEPHYS-Abschnitt: ausgewählte Seiten des Seitenwechselbereichs
8. PROTKEYS: Speicherschutzschlüssel

   Zusätzlich, falls die Daten erreichbar sind und zuvor BS2000, IPL, SYSTART, VM2000 oder SLED selbst geladen war, können enthalten sein:
9. TSOSCAT: Systemkatalog
10. EQUISAMQ: SPOOL-Auftragswarteschlange
11. SJOBPOOL: Job-Management-Auftragswarteschlange
12. REPLOG: Kann auf die Datei REPLOG nicht zugegriffen werden, sichert SLED die Datei SAVEREP, die nur die BS2000-Korrekturen enthält.
13. CONSLOG: letzte Konsole-Protokolldatei dieser Session
14. CONSLOG1: erste Konsole-Protokolldatei dieser Session
15. CONSLOG2: vorletzte Konsole-Protokolldatei dieser Session
16. HELFILE: Hardware-Error-Loggingdatei HEL
17. SERSLOG: letzte Software-Error-Loggingdatei dieser Session
18. SERSLOG1: erste Software-Error-Loggingdatei dieser Session
19. SERSLOG2: vorletzte Software-Error-Loggingdatei dieser Session
20. MSCFTRAC: MSCF-Tracedatei
21. SJMSFILE: JMS-Datei
22. PAGELOG-Abschnitt: Tabelle der beim Systemabbruch gesicherten Tasks
23. SLEDMEM-Abschnitt: IPL- und SLED-Codierung des aktuellen SLED-Laufs
24. SLEDLOG-Abschnitt: Aufzeichnung des SLED-Dialogs

Bei Angabe von `REAL` enthält die SLEDFILE die Punkte 1, 2, 3, 4, 5, 6, 8, 23 und 24.
Bei Angabe von `NSF` fehlen in der SLEDFILE die Punkte 9 bis 21.

Einen maximalen Dump kann der Operator durch die Angabe MODE=ALL und TASK=ALL als Antwort auf die Meldungen `NSD3001` bzw. `NSD3002` veranlassen.

Ausgabe auf ein emuliertes Bandgerät

Auf allen BS2000-Servern sind zwei Bandgeräte konfiguriert, die von der Management Unit (SU /390), vom SKP (S-Server) oder von X2000 (SU x86) emuliert werden. Eines der Bandgeräte arbeitet real auf Basis des eingebauten CD/DVD-Laufwerks. Das andere arbeitet auf Basis einer Datei, die im Dateisystem der MU, des SKP oder von X2000 abgelegt wird. Zusätzlich können weitere, auf Dateibasis arbeitende Bandgeräte konfiguriert werden.

Für die SLED-Ausgabe muss ein auf Dateibasis arbeitendes emuliertes Bandgerät verwendet werden. Bandgeräte, die auf Basis eines CD/DVD-Laufwerks arbeiten, können für die Ausgabe von SLED nicht genutzt werden.

> ¡ Die SLED-Ausgabe auf ein emuliertes Bandgerät ist für Situationen vorgesehen, in denen eine SLED-Datei auf Platte nicht verfügbar ist. Die SLED-Ausgabe ist insbesondere bei großen Systemkonfigurationen, in denen Folgebänder benötigt werden, nicht komfortabel und erfordert eine gewisse Vorbereitung.

Das Band des Bandgeräts muss bereits initialisiert, d.h. mit Standard-Kennsätzen (VOL1, HDR1 und HDR2) versehen sein. Weder die Archivnummer (VSN) noch die Schreibdichte können verändert werden. Es wird außerdem geprüft, ob das im Kennsatz HDR1 eingetragene Verfallsdatum erreicht ist.

> ¡ Das im vorkonfigurierten Bandgerät sichtbare Band ist bei älteren Servern und Firmware-Ständen noch nicht initialisiert.
> Wenn die fehlende Initialisierung nicht nachgeholt wird (z.B. mit dem Dienstprogramm INIT), dann ist eine spätere SLED-Ausgabe auf das Band nicht möglich.

Die SLED-Ausgabedatei auf Band hat immer den Namen SLEDFILE.

SLED fordert mit den Meldungen `NSD3800` und `NSD3822` zwei Angaben an: die Archivnummer (VSN) und die Geräte-Identifikation (Gerätemnemonik).

1. Archivnummer (VSN)
   Die Angabe der VSN kann voll- oder teilqualifiziert erfolgen. Als Wildcard-Symbol (nur am Ende der Eingabe erlaubt) wird das Zeichen `*` verwendet. Wird nur `*` als VSN angegeben, akzeptiert SLED alle Bänder, sofern sie mit Standardetiketten versehen sind.
2. Geräte-Identifikation
   Als Geräte-Identifikation wird die Gerätemnemonik `mn` angegeben. SLED prüft, ob das angegebene Gerät existiert und für die Ausgabe verwendet werden kann. Ist die Angabe nicht gültig, wiederholt SLED die Forderung nach der Geräte-Identifikation.

   Diese Meldungen erscheinen ebenso, wenn **ein** Band zur Ausgabe nicht ausreicht und ein Folgeband verwendet werden muss. Falls die Archivnummer mit `*` spezifiziert wurde, gilt diese Angabe auch für die folgenden Bänder.

   Alle Folgebänder müssen auf demselben Gerät montiert werden.

Das „Einlegen“ eines Folgebandes erfolgt aufgrund der Emulation nur durch folgende Aktionen:

1. Sichern der zur Bandemulation gehörigen Datei, z.B. durch Herunterladen auf einen PC.
2. Überschreiben der zur Bandemulation gehörigen Datei mit einer vorbereiteten Datei, die ein leeres Band mit einer anderen VSN repräsentiert. Die geschieht z.B. durch Hochladen von einem PC.

   Die Vorgehensweise für beide Schritte ist in der Betriebsanleitung des jeweiligen Servers beschrieben.

Die beschriebenen Dateien sollten aus Gründen der Datensicherheit nach der Auswertung überschrieben (z.B. durch erneutes Initialisieren des Bandes) oder physikalisch gelöscht werden.

Ausgabe auf Privatplatte

Bei der Ausgabe auf Privatplatte muss die SLED-Ausgabedatei (SLEDFILE) ganz auf einer Platte liegen und darf nicht über mehrere Platten verteilt sein. Die Datei muss bereits eingerichtet und hinreichend groß sein.

Der Operator wird aufgefordert, die Archivnummer der Platte anzugeben. Anschließend wird mit `NSD3410` nach der Geräteadresse der Platte gefragt.
Will der Operator nur den Gerätenamen verwenden, kann er die Meldung `NSD3400` mit `*` oder `<EOT>` beantworten.
In der Parameterdatei müsste stehen: `VSN=*, DEV=mn`

Wenn nun die Ausgabeplatte definiert und gefunden ist, wird nach dem Namen der Ausgabedatei gefragt. Bei der Ausgabe auf Privatplatte erfolgt der Dateizugriff nicht über den Systemkatalog, sondern ausschließlich über die F1-Kennsätze auf der Platte. Daher ist in diesem Fall die Angabe einer Katalogkennung (catid) nicht sinnvoll und wird als Fehler zurückgewiesen.

> ¡ Wenn ein Katalogeintrag für die Datei auf Privatplatte existiert, so wird dieser nicht aktualisiert. Nach dem Einrichten einer SLED-Ausgabedatei auf Privatplatte muss der zugehörige Katalogeintrag mit dem Kommando EXPORT-FILE FILE-NAME= filename gelöscht werden.
>
> SLED-Dateien dürfen nicht auf DRV-Privatplatten eingerichtet werden.

Ausgabe auf gemeinschaftliche Platten

Bei der Ausgabe auf eine gemeinschaftliche Platte muss zunächst festgestellt werden, auf welchem Pubset die Ausgabedatei liegt. Die Ausgabedateien für den SLED dürfen auch außerhalb des Home-Pubsets liegen, jedoch nur auf Platten bzw. Pubsets, die als IPL-Platte bzw. Home-Pubset geeignet wären, also z.B. nicht auf SM-Pubsets. Auch die Ausgabe auf Shared-Pubsets wird zurückgewiesen.

Der Pubset der SLEDFILE wird aus dem Dateinamen hergeleitet, nach dem als erstes gefragt wird. Dabei gilt:

1. Wurde der Dateiname mit der Katalogkennung angegeben, ist dadurch bereits der Pubset der SLEDFILE angegeben.
2. Wurde der Dateiname ohne Katalogkennung angegeben (oder wurde durch leere Eingabe der Standardname spezifiziert), so wird versucht, den Pubset über eine der beiden folgenden Standard-Regeln zu bestimmen:
   a. SLED wurde von einer gemeinschaftlichen Platte geladen. Dann ist der Pubset, zu dem diese Platte gehört, der Pubset der SLEDFILE.
   b. SLED wurde von einer Privatplatte geladen, aber alle gemeinschaftlichen Platten, die online sind, gehören zu einem einzigen Pubset. Dieses ist dann der Pubset der SLEDFILE.

   Läßt sich keine dieser Regeln anwenden, dann wurde SLED von einer Privatplatte geladen und es sind gemeinschaftliche Platten aus verschiedenen Pubsets oder überhaupt keine gemeinschaftlichen Platten online. In diesem Fall wird der Operator aufgefordert, die Katalogkennung der SLED-Ausgabedatei anzugeben.

Wenn der Pubset der SLEDFILE bekannt ist, wird zuerst die zugehörige Sysres und dann alle übrigen Platten des Pubsets gesucht. SLED kann nur weiterarbeiten, wenn alle Platten des Pubsets der SLEDFILE online sind. Fehlen Platten – dies wird mit der Meldung `NSD1400` angezeigt –, muss nach dem Zuschalten der betroffenen Platten SLED erneut geladen werden (SLED-Wiederholung!).

Anschließend wird versucht, die spezifizierte Ausgabedatei zu finden. Dazu muss ein benutzbarer Katalog mit der angegebenen Katalogkennung zur Verfügung stehen.

> ¡ Wird an dem betreffenden System das Software-Produkt HSMS eingesetzt, muss die Systembetreuung dafür Sorge tragen, dass die zur Ausgabe bestimmte Datei nicht wegen längerer Nichtbenutzung automatisch „verdrängt“ wurde und damit nicht mehr zugreifbar ist.
>
> Ein Pubset für SLED-Ausgabe darf bei SLED-Betrieb nicht von einem laufenden System importiert sein.

Überprüfung der SLED-Ausgabedatei

Wenn die Ausgabedatei (SLEDFILE) bestimmt und gefunden ist, wird sie auf ihre Verwendbarkeit überprüft:

1. Sie darf nicht durch ein Kennwort geschützt sein.
2. Sie darf nicht gegen Schreibzugriffe geschützt sein. Es muss also ACCESS=WRITE gelten.
3. Das Verfallsdatum (EXPIRATION-DATE) muss erreicht sein.
4. Sie soll groß genug sein, um wenigstens den Abzug des Hauptspeichers, der CONSLOG- und SERSLOG-Dateien und der Hardware-Daten aufnehmen zu können. Sehr großer Hauptspeicher wird von SLED in mehreren Portionen geschrieben, sodass der Dump auch dann durch DAMP aufbereitet werden kann, wenn der Hauptspeicher nicht vollständig gesichert werden konnte.
   Trotzdem sollte eine zu kleine SLED-Datei verhindert werden, da gerade jene Daten fehlen könnten, die für die Diagnose des Fehlers erforderlich sind.
5. Wenn Ausgabe auf gemeinschaftliche Platten verlangt worden ist, darf die Datei nicht auf Privatplatte eingerichtet sein.
6. Die Ausgabe auf Shared-Pubsets und SM-Pubsets ist verboten und wird vom IPL zurückgewiesen.

Ist eine dieser Bedingungen nicht erfüllt, so wird nach einer entsprechenden Meldung (`NSD32xx`) erneut der Name der SLED-Ausgabedatei angefordert.

Ist die SLED-Datei nicht logisch leer, wird sie nur benutzt, wenn der Operator eine entsprechende Nachfrage (Meldung `NSD3204`) positiv beantwortet, andernfalls wird erneut nach dem Namen der SLED-Ausgabedatei gefragt.

Größe von SLED-Ausgabedateien

Soll die SLED-Ausgabe auf Platte erfolgen, so muss dort eine ausreichend große Datei zur Verfügung stehen. Die Ausgabedatei darf auch größer als 32 GB (jedoch maximal 256 GB) sein und auf dem Home-Pubset liegen.

Da SLED ohne Unterstützung des Datenverwaltungssystems arbeitet, erfolgt beim SLED-Lauf keine dynamische Dateierweiterung. Beim Einrichten einer SLED-Ausgabedatei muss also die Angabe im SPACE-Operanden des Kommandos CREATE-FILE genügend groß sein.

Der Umfang des Speicherauszugs, d.h. die Größe der SLED-Ausgabedatei, ist generell von vielen Faktoren abhängig, die vor dem Dump-Zeitpunkt unbekannt sind. Zum Dump-Zeitpunkt kann die Größe der Ausgabedatei über die Parameter MODE (Seiten, die in die Datei aufgenommen werden sollen) und TASK (Tasks, deren Adressräume gesichert werden sollen) gesteuert werden.

Sollen durch Setzen des Parameters TASK=ALL die Adressräume aller Tasks erfasst werden, wird nicht nur die Laufzeit des SLED, sondern auch die Dateigröße entscheidend beeinflusst. TASK=ALL kann ein Vielfaches der Größe von TASK=STD bedeuten.
Bei großen Hauptspeichern oder Systemdateien fällt auch der Parameter MODE ins Gewicht. Der Wert MODE=ALL sollte in diesen Fällen nur bei genügend großer Datei gewählt werden.

Zieht man den (Standard-)Fall MODE=STD, TASK=STD in Betracht, haben folgende Größen hierbei einen wesentlichen Einfluss:

A: Größe des benutzten Systemadressraums

B: Größe der Systemdateien TSOSCAT, EQUISAMQ, REPLOG, CONSLOG[x], SJOBPOOL, HEL, SERSLOG[x], SJMSFILE, MSCFTRAC, ...

n: Anzahl der Verarbeitungsprozessoren

t: Anzahl der unterschiedlichen Tasks, die in der TIC-Tabelle als Tasks in Contol geführt werden (maximal 64 Einträge je Prozessor)

C: Größe des benutzten Klasse-5-Adressraums einer Task

Der Einfluss dieser Größen ergibt sich aus der Formel (A + B + n * t * C)

Problematisch bei der Berechnung der Dateigröße sind hierbei vor allem die Größen t und C. Setzt man für diese einfach die jeweilige Obergrenze ein, kommt man wieder zu in der Praxis unbrauchbar großen Werten. Allgemein gültige Durchschnittswerte lassen sich nur schwer finden, da diese stark von der Anwendung und der Auslastung des Systems abhängen (hier kann man gegebenenfalls die durchschnittlichen Ergebnisse von openSM2-Messungen einfließen lassen).

Zusammenfassend ist eine Empfehlung bezüglich der Größe der Ausgabedatei schwer zu treffen. In den meisten Fällen (mit Ausnahme von TASK=ALL) wird der Benutzer jedoch mit der doppelten Hauptspeichergröße auskommen.

Faustregel für MODE=STD/ALL, TASK=STD

Größe der `SLEDFILE = 2 * Größe des Hauptspeichers`.

Beispiel für einen SLED-Lauf

```
%S.NSI00E3  IPL-REPS READ: 0; EXECUTED: 0
%S.NSI1100  IPL DEVICE = HIP6.1; IPL PATH = B93E (MN=B93E)
%S.NSI1163  LOCAL DATE = <date>, TIME = <time> FROM SVP (MESZ)
%S.NSI00E3  SLED-REPS READ: 0; EXECUTED: 0
?S.NSD1003  STANDARD SLED ? REPLY (Y; N; EOT=Y)
s.
%S.NSD1000  SLED VERSION <version> LOADED FROM HIP6.1 TO 021D7000
%S.NSI3135  IPL DISK-SETUP READ FROM IPL-CONF PREPARED <date> <time>
%S.NSD1604  WARNING: SLEDSAVE ON VOLUME HIP6.1 TOO SMALL
%S.NSD1111  PRODUCT-ID OF DUMPED SYSTEM: BS2000 <version>
?S.NSD5200  SPECIFY NAME OF SLED PARAMETER FILE. REPLY (NO FILE=EOT; FILENAME;
STANDARD NAME=STD; END)
s.
?S.NSD1113  DO YOU WANT TO CHANGE CURRENT SLED RUNTIME LIMIT OF 045 MINUTES ? REPLY
(Y; N; EOT=N; - (BACKTRACK))
s.
?S.NSD3000  SPECIFY OUTPUT DEVICE. REPLY (DPUB; DPRIV; TAPE; EOT=DPUB; -
(BACKTRACK))
s.tape
?S.NSD3800  SPECIFY VSN OF SLED OUTPUT TAPE. REPLY (VSN; VSN*; - (BACKTRACK))
s.cs563k
?S.NSD3822  SPECIFY MN OF SLED OUTPUT TAPE CS563K. REPLY (MN;  - (BACKTRACK))
s.me
%S.NSD3810  TAPE CS563K ON DEVICE ME INITIALISED AS T6250
%S.NSD5000  DEFAULT TAKEN: MODE=STD
%S.NSD5000  DEFAULT TAKEN: TASK=STD
%S.NSD1112  SLED RUNTIME LIMIT SET TO   45  MINUTES
%S.NSD1800  STATUS   SECTION : TERMINATED. LAST BLOCK = 8. TIME = <time>
```

```
%S.NSD1800  PSA      SECTION : TERMINATED. LAST BLOCK = 18. TIME = <time>
%S.NSD1701  MAINMEM  SECTION : PAGE SELECTION STARTED. TIME = <time>
%S.NSD1702  MAINMEM  SECTION : DUMP STARTED. TIME = <time>
%S.NSD1800  MAINMEM  SECTION : TERMINATED. LAST BLOCK = 80772. TIME = <time>
%S.NSD1702  VM2HYPVS SECTION : DUMP STARTED. TIME = <time>
%S.NSD1800  VM2HYPVS SECTION : TERMINATED. LAST BLOCK = 82442. TIME = <time>
%S.NSD1800  IOHIOSDP SECTION : TERMINATED. LAST BLOCK = 82708. TIME = <time>
%S.NSD1701  PAGEPHYS SECTION : PAGE SELECTION STARTED. TIME = <time>
%S.NSD1702  PAGEPHYS SECTION : DUMP STARTED. TIME = <time>
%S.NSD1800  PAGEPHYS SECTION : TERMINATED. LAST BLOCK = 84776. TIME = <time>
%S.NSD1800  PROTKEYS SECTION : TERMINATED. LAST BLOCK = 84906. TIME = <time>
%S.NSD1800  TSOSCAT  SECTION : TERMINATED. LAST BLOCK = 101074. TIME = <time>
%S.NSD1800  EQUISAMQ SECTION : TERMINATED. LAST BLOCK = 101114. TIME = <time>
%S.NSD1800  SJOBPOOL SECTION : TERMINATED. LAST BLOCK = 101148. TIME = <time>
%S.NSD1800  REPLOG   SECTION : TERMINATED. LAST BLOCK = 102364. TIME = <time>
%S.NSD1800  CONSLOG  SECTION : TERMINATED. LAST BLOCK = 102588. TIME = <time>
%S.NSD1800  HELFILE  SECTION : TERMINATED. LAST BLOCK = 103000. TIME = <time>
%S.NSD1800  SERSLOG  SECTION : TERMINATED. LAST BLOCK = 174636. TIME = <time>
%S.NSD1800  MSCFTRAC SECTION : TERMINATED. LAST BLOCK = 174712. TIME = <time>
%S.NSD1800  SJMSFILE SECTION : TERMINATED. LAST BLOCK = 174782. TIME = <time>
%S.NSD1800  PAGELOG  SECTION : TERMINATED. LAST BLOCK = 174786. TIME = <time>
%S.NSD1800  SLEDMEM  SECTION : TERMINATED. LAST BLOCK = 176308. TIME = <time>
%S.NSD1800  SLEDLOG  SECTION : TERMINATED. LAST BLOCK = 176314. TIME = <time>
%S.NSD1802  SLED OUTPUT COMPLETED
?S.NSD5200  SPECIFY NAME OF SLED PARAMETER FILE. REPLY (NO FILE=EOT; FILENAME;
STANDARD NAME=STD; END)
s.end
%S.NSD1001  SLED TERMINATED
```

10.3 SLED-Steuerung

Die Steuerung des SLED kann mit verschiedenen Mitteln realisiert werden.

- Im bedienten SLED (= manueller SLED) wird ein Operatordialog durchgeführt. Der SLED wird entweder durch die in der – vom Operator anzugebenden – Parameterdatei vereinbarten Anweisungen gesteuert oder durch die Eingabe einzelner Parameter durch den Operator.
- Beim unbedienten SLED (= automatischer SLED) wird kein Operatordialog durchgeführt. Der SLED wird über die Standard-SLED-Parameterdatei $TSOS.SYSPAR.SLED.<ver> oder durch die Auswertung von Default-Werten gesteuert.
- Die Verwendung asynchroner Kommandoeingaben kann Informationen über den SLED-Lauf ausgeben.

SLED-Laufzeitbegrenzung

SLED hat eine Laufzeitbegrenzung von 45 Minuten voreingestellt. Diese Voreinstellung kann im Dialog geändert werden, indem die folgende Meldung mit Y beantwortet wird:

```
NSD1113 DO YOU WANT TO CHANGE CURRENT SLED RUNTIME LIMIT OF (&00) MINUTES ?
        REPLY (Y; N; EOT=N; - (BACKTRACK))
```

Auf die Antwort Y wird folgende Meldung ausgegeben:

```
NSD1114 SET RUNTIME LIMIT.
        REPLY (1-999 (MINUTES); N (CURRENT LIMIT); EOT=N; - (BACKTRACK))
```

Es kann eine Laufzeitbegrenzung von 1 bis 999 Minuten angegeben werden.

SLED bestätigt die eingestellte Laufzeitbegrenzung mit der Meldung `NSD1112`.

Ist die Zeitbegrenzung eingeschaltet, wird nach der Sicherung des Hauptspeichers das erste Mal und anschließend nach jedem Abschnitt geprüft, ob die angegebene Zeitschranke erreicht wurde. Wenn ja, wird der SLED beendet, nachdem noch die SLEDMEM- und die SLEDLOG-Abschnitte geschrieben wurden.

> ¡ Während des Speicherabzugs des Seitenwechselbereichs wird nach der Sicherung von jeweils 2 GB die Zeitbegrenzung überprüft und ggf. die Speicherung vorzeitig beendet. Es kann also vorkommen, dass die Daten des Seitenwechselbereichs nicht in vollem Umfang abgezogen werden.

Eine vorzeitige Beendigung von SLED wegen Laufzeitbegrenzung wird durch die Meldungen `NSD1804` und `NSD1803` angezeigt.

SLED-Dialog

Eingabe von Parametern im Voraus

Bei einigen Meldungen des Anfangsdialogs ist es möglich, zusammen mit der Beantwortung der gerade gestellten Frage weitere Parameter im Voraus einzugeben, die normalerweise in weiteren Dialogschritten abgefragt würden oder für die der SLED Standardwerte einsetzen würde.

Beispiel

```
NSD3000 SPECIFY OUTPUT DEVICE.
        REPLY (DPUB; DPRIV; TAPE; EOT=DPUB; - (BACKTRACK))
```

Diese Meldung kann folgendermaßen beantwortet werden:

`TAPE` Ausgabe auf Band

`TAPE,VSN=vsn`

Die Archivnummer des Ausgabebandes wird im Voraus bekannt gemacht

`TAPE,VSN=vsn,DEV=mn,MODE=NSF`

Die Archivnummer des Ausgabemediums und der (Teil-)Umfang der Ausgabedaten werden im Voraus bekannt gemacht

`TAPE,VSN=vsn,DEV=mn,MODE=NSF,TASK=ALL`

Alle Parameter für die Ausgabe auf Band werden im Voraus bekannt gemacht

Die Syntax ist also wie bei der Parameterliste eines BS2000-Kommandos, wobei höchstens ein Stellungsparameter und optional einige Schlüsselwortparameter vorkommen dürfen. Welche Kombinationen von Stellungs- und Schlüsselwortparametern jeweils erlaubt sind, ist bei den einzelnen Meldungen beschrieben.

Die Eingabe darf keine Zwischenräume enthalten. Der erste auftretende Zwischenraum wird als Ende der Eingabe interpretiert.

Ist der Stellungsparameter nicht angegeben (also nur Schlüsselwortparameter oder kein Parameter), so wird für ihn der Standardwert angenommen.

Ist ein Schlüsselwortparameter ohne Wert angegeben (z.B. `MODE=`), so wird für ihn der Standardwert eingesetzt. Welcher Standardwert angenommen wird, ist durch die Meldung definiert, durch die der entsprechende Parameter ohne Voraus-Eingabe erfragt werden würde.

Bei der Voraus-Eingabe von Parametern können Fehler auftreten, die dazu führen, dass die Eingabe ignoriert wird. Eine Meldungsausgabe informiert darüber.

Rückzugsmöglichkeit

Bei einigen Meldungen des Anfangsdialogs ist ein Rückzug möglich, indem als Antwort „-“ oder „- -“ eingeben wird.

Die Eingabe von „-“ bewirkt, dass die letzte Eingabe rückgängig gemacht wird (einfacher Rückzug).

Die Eingabe von „- -“ bewirkt, dass alle bisher gemachten Eingaben rückgängig gemacht werden. Mit der Meldung `NSD5200`, der ersten Meldung des Anfangsdialogs, wird der Dialog fortgesetzt. Bei einem SLED mit Parameterdatei wird nach dieser Eingabe die weitere Bearbeitung der Parameterdatei gestoppt.

In beiden Fällen werden im voraus eingegebene Parameter nach entsprechender Meldung (`NSD5003`) ignoriert.

Besteht die Möglichkeit eines einfachen Rückzugs, ist dies im Reply-Teil der Meldung aufgeführt:
`NSDxxxx ... REPLY (...; - (BACKTRACK))`

SLED mit Parameterdatei

Alle Anweisungen, die für den Ablauf beim bedienten oder automatischen SLED benötigt werden, können in Form einer Datei parametrisiert werden. In dieser Parameterdatei, die auf einer gemeinschaftlichen Platte angelegt werden muss, werden alle Eingaben an SLED hinterlegt, wobei die syntaktische Überprüfung der Angaben erst während des SLED-Laufs erfolgt.
Die SLED-Parameterdatei darf nicht leer sein und muss folgende Dateiattribute besitzen:

FILE-STRUC=SAM
BUF-LEN=STD oder (STD,2)
REC-FORM=V
BLK-CONTR=PAMKEY oder DATA

Alle Parameter für einen SLED-Lauf müssen in einem Satz der Parameterdatei stehen. Die einzelnen Parameter werden durch Kommas getrennt. Die Zeichenfolge darf keine Leerzeichen enthalten. Jeder Parameter muss mit dem entsprechenden Schlüsselwort eingeleitet werden. Die Reihenfolge der Parameter ist unerheblich. Kleinbuchstaben und nicht abdruckbare Zeichen dürfen in den Parametersätzen nicht enthalten sein.

Beim automatischen SLED werden die Parameter aus der Parameterdatei mit dem Namen $TSOS.SYSPAR.SLED.<ver> eingelesen, so dass auch in diesem Fall die flexible Steuerung des Speicherauszugs gewährleistet ist.

> **i** Wird an dem betreffenden System das Software-Produkt HSMS eingesetzt, muss die Systembetreuung dafür Sorge tragen, dass die Parameterdatei nicht wegen längerer Nichtbenutzung automatisch verdrängt (migriert) wird und damit nicht mehr zugreifbar ist.

Zuweisung der Parameterdatei

Die Entscheidung, ob und welche Parameterdatei für den SLED-Lauf verwendet werden soll, trifft der Operator durch die Beantwortung der Meldung `NSD5200`:

```
NSD5200 SPECIFY NAME OF SLED PARAMETER FILE.
        REPLY (NO FILE=EOT; FILENAME; STANDARD NAME=STD; END)
```

`FILENAME`	Die Zuweisung einer Parameterdatei für einen SLED-Lauf erfolgt durch Beantwortung der Meldung mit einem gültigen, vollqualifizierten Dateinamen. Bei Verzicht auf Katalog- bzw. Benutzerkennung, wird jeweils der Standard (Katalogkennung des Home-Pubsets bzw. $TSOS) angenommen.
`STD`	Durch Beantwortung der Meldung mit `STD` wird die Standard-SLED-Parameterdatei des Systems $TSOS.SYSPAR.SLED.<ver> verwendet.
`EOT` (keine ingabe)	Eine leere Eingabe (`EOT`) als Antwort auf die Meldung bewirkt, dass keine Parameterdatei verwendet wird. SLED fordert im bedienten Dialog mit dem Operator die benötigten Eingaben von der Konsole an.

Bearbeitung der Parameterdatei

Parametersätze, die bereits abgearbeitet wurden und für die bereits eine nicht-leere Ausgabedatei auf Magnetplatte existiert, werden bei der nächsten SLED-Anforderung übergangen (im automatischen Betrieb ohne Nachfrage, im bedienten Betrieb dann, wenn folgende Meldung mit `N` beantwortet wird):

```
NSD3204 SLED OUTPUT FILE (&00) IS NOT EMPTY.
        OVERWRITE ? REPLY (YES=Y; NO=N)
```

Dadurch ist es möglich, mit einer Parameterdatei, die mehrere Parametersätze enthält, auf mehrere Systemabbrüche (über einen längeren Zeitraum) zu reagieren.

Alle Parameter für einen Lauf müssen in einer Zeile stehen, z.B.

- bei Ausgabe auf Privatplatte:

  `OUTPUT=DPRIV,FILE=...,MODE=,TASK=STD,VSN=*,DEV=D6`

- bei Ausgabe auf gemeinschaftlicher Platte:

  `OUTPUT=DPUB,FILE=...,MODE=,TASK=STD`

- bei Ausgabe auf ein emuliertes Band:

  `OUTPUT=TAPE,VSN=SLED*,DEV=M0,MODE=NSF,TASK=(1EF0,1431,2EE4,5QA1)`

Das Verarbeitungsende ist erreicht, wenn eine SLEDFILE geschrieben wurde oder ein Parametersatz mit der Zeichenfolge `OUTPUT=END` erkannt wird. Nachfolgende Sätze werden dann ignoriert und SLED beendet sich. Beim automatischen SLED wird das Laden des Systems eingeleitet.

Das Dateiende wird durch eine entsprechende Meldung an der Konsole angezeigt. Im bedienten Betrieb gibt SLED erneut die Meldung `NSD5200` aus. Beim automatischen SLED wird das Laden des Systems eingeleitet.

Fehlerverhalten im bedienten Betrieb (manueller SLED)

- Tritt während der Abarbeitung eines Parametersatzes ein Fehler durch fehlende oder fehlerhafte Angaben auf, so gibt SLED die zugehörige Anforderung aus, den Parameter von der Konsole einzugeben.
  Wird der Fehler korrigiert bzw. die fehlende Angabe nachgereicht, fährt SLED mit der Abarbeitung der Datei fort. Entscheidet sich der Operator für die Möglichkeit des Rückzugs, wird die weitere Bearbeitung abgebrochen.
- Werden während der Verarbeitung leere oder solche Sätze erkannt, die Kleinbuchstaben bzw. nicht abdruckbare Zeichen enthalten, fordert SLED den Operator mit der Meldung `NSD5245` auf, den weiteren Ablauf festzulegen. Die Verarbeitung dieser Parameterdatei kann entweder abgebrochen oder mit dem nächsten Satz fortgesetzt werden.

Fehlerverhalten im unbedienten Betrieb (beim automatischen SLED)

- Ist die Datei $TSOS.SYSPAR.SLED.<ver> auf dem Home-Pubset nicht vorhanden oder für SLED nicht auffindbar (z.B. bei zerstörtem TSOSCAT), verfährt SLED wie beim automatischen SLED ohne Parameterdatei (siehe "SLED-Steuerung").
- Besitzt die Parameterdatei nicht die geforderten Dateiattribute, wird sie abgewiesen und in den bedienten Betrieb gewechselt.
- Tritt während der Abarbeitung eines Parametersatzes ein Fehler durch fehlende oder fehlerhafte Angaben auf, wird die Parameterdatei nicht weiter bearbeitet und auf den bedienten Betrieb umgeschaltet.
- Leere Parametersätze einer Parameterdatei werden ignoriert.
- Fehlen die MODE- oder TASK-Parameter, so werden diese durch SLED bestimmt.

Automatischer SLED

Der automatische SLED bietet die Möglichkeit, ohne Operatoreingriffe einen Speicherauszug zu erstellen und anschließend das System neu zu laden.

Der SLED wird im automatischen Modus geladen, wenn beim Systemabbruch die Funktion „automatischer Restart“ eingeschaltet und SLED als Dumperzeuger eingestellt ist (siehe Kommando SET-RESTART-OPTIONS MODE=*ON (...), DUMP=*SLED). Der automatische SLED wird von der Sysres des Home-Pubsets des abgebrochenen Systems geladen. Mit dem Kommando SHOW-RESTART-OPTIONS können Informationen über die Steuerung des automatischen Restarts ausgegeben werden.

Grundsätzlich kann der Ablauf für den SLED auf zwei Arten automatisiert werden:

1. Parameterdatei

   Zunächst sucht SLED die Standard-Parameterdatei $TSOS.SYSPAR.SLED.<ver> auf dem Home-Pubset. Wenn diese Datei gefunden wird und bearbeitet werden kann, verwendet SLED die dort spezifizierten Angaben.
   Parametersätze, die nicht-leere Plattendateien beschreiben, werden übergangen.

2. Einsatz von Default-Werten

   Findet SLED die Standard-Parameterdatei $TSOS.SYSPAR.SLED.<ver> nicht, werden für den Ablauf des automatischen SLED folgende Default-Vereinbarungen bestimmt:

   - Dumpdatei: `$TSOS.SLEDFILE`
   - Parameter Mode: `EOT`
   - Parameter Task: `EOT`

   Dies entspricht den folgenden Anweisungen in der Parameterdatei:

   ```
   OUTPUT=DPUB,FILE=$TSOS.SLEDFILE,MODE=,TASK=
   OUTPUT=END
   ```

Fehlerfall

Bei schwerwiegenden Fehlern wechselt SLED vom automatischen SLED in den bedienten Betrieb.
Dies gilt auch für den Fall, dass mit einer Parameterdatei gearbeitet wird und bei der Abarbeitung eines Parametersatzes fehlende oder fehlerhafte Angaben erkannt werden.
Wird beim automatischen SLED ein Fehler in der Plattenverfügbarkeit erkannt (Meldung `NSD1400`), sind die zugehörigen Daten für den SLED nicht verfügbar.

Voraussetzungen für den operatorlosen Ablauf

- SLED muss die gleiche Version haben wie das Betriebssystem.
- Alle Plattengeräte, auf denen gemeinschaftliche Platten bzw. Paging-Platten montiert sind, müssen betriebsklar sein.
- Die Datei $TSOS.SLEDFILE bzw. die in der Parameterdatei zugeordnete Datei
  - muss eingerichtet sein,
  - muss auf dem Home-Pubset angelegt sein (nur bei $TSOS.SLEDFILE),
  - darf nicht gegen Schreibzugriff geschützt sein,
  - darf nicht durch ein Kennwort geschützt sein,
  - sollte der zweifachen Hauptspeichergröße entsprechen,
  - muss logisch leer sein,
  - muss das Verfallsdatum erreicht haben,
  - darf nicht auf DRV-Platten liegen.

Automatischer Wiederanlauf des Systems

Der automatische Wiederanlauf des Systems nach Ablauf des automatischen SLED ist in folgenden Fällen gewährleistet:

- Der automatische SLED ist fehlerfrei gelaufen und alle erreichbaren Daten konnten in eine SLED-Ausgabedatei geschrieben werden.

- Der automatische SLED arbeitet ohne Parameterdatei und die Ausgabedatei $TSOS.SLEDFILE war nicht leer. In diesem Fall werden keine Daten abgezogen und das System wird sofort neu geladen.
- Beim automatischen SLED mit Parameterdatei werden nicht-leere Ausgabedateien auf Magnetplatte solange ignoriert, bis ein SLEDFILE erstellt wurde oder das Ende der Parameterdatei erreicht ist oder ein Parametersatz erkannt wird, der die Zeichenfolge `OUTPUT=END` enthält.

  Dies ermöglicht, mehrere SLED-Ausgabedateien zur Verfügung zu stellen, die dann in unterschiedlichen SLED-Läufen nacheinander beschrieben werden.
- Die Parameterdatei enthält nur einen Parametersatz mit der Anweisung `OUTPUT=END`. Hierdurch behält man sich die Option vor, einen automatischen Wiederanlauf des Systems ohne Erstellung von Diagnosedaten einzuleiten.

Asynchrone Kommando-Eingaben

Unter einer asynchronen Eingabe versteht man bei SLED jede Eingabe von der Konsole, bei der es sich nicht um die Beantwortung einer Meldung handelt. In Anlehnung an die BS2000-Kommandos beginnen alle asynchronen Eingaben mit Schrägstrich (/).

Diese Kommandos sind nicht an jeder beliebigen Stelle des SLED-Laufs möglich. Im Allgemeinen sind sie nur dann zugelassen, wenn der Anfangsdialog abgeschlossen ist und die Verarbeitung des Speicherauszugs bereits begonnen hat.

Die folgende asynchrone Eingabe wird bearbeitet; alle anderen werden mit einer entsprechenden Meldung abgewiesen.

Kommando STATUS

Mit diesem Kommando erhält der Operator Hinweise darüber, wie weit der SLED-Lauf schon fortgeschritten ist.

Es wird mitgeteilt, welchen Block SLED zuletzt geschrieben hat. Dadurch hat der Operator die Möglichkeit, auch während eines Abschnitts festzustellen, wie weit der SLED-Lauf gekommen ist.

SLED-Meldungen

Die SLED-Meldungen beginnen mit der Schlüsselnummer `NSDxxxx`.

Die für `xxxx` stehende Zahl lässt sich folgendermaßen zuordnen:

09xx Ablauf Nicht-Standard-Sled

1xxx Ablauf:

- 10xx Start und Beendigung
- 11xx Inkonsistenzen
- 12xx Interne Fehler
- 14xx Plattenverfügbarkeit
- 16xx Plattenzugriff
- 18xx Dump-Umfang
- 19xx Automatischer SLED

3xxx Ausgabe:

- 30xx Allgemein
- 32xx Plattendatei
- 33xx Plattenfehler
- 34xx Privatplatte
- 36xx gemeinschaftliche Platten
- 38xx Band
- 39xx Bandfehler

5xxx Eingabe-Verarbeitung:

- 50xx Voraus-Eingabe von Parametern
- 52xx Parameterdatei
- 55xx Spezifische Checks
- 56xx asynchrone Eingaben

7xxx Abschnitte der SLED-Ausgabe:

- 72xx Hauptspeicher
- 73xx HSA
- 74xx PSA
- 76xx Systemdateien
- 78xx Virtuelle Räume
- 79xx SLEDLOG

10.4 Extrahieren des IOHDUMP aus einem SLED

Auf x86-Servern werden die Diagnosedaten IOHDUMP in die BS2000-SLED-Datei übernommen. Dort werden sie in der Section IOHIOSDP abgelegt. Diese Daten dienen zur Diagnose von HSI-Fehlern. Sie können wie folgt verarbeitet werden:

```
/START-DAMP
//OPEN-DIAGNOSIS-OBJECT OBJECT <sledfile>
//SHOW-EDITED-INFORMATION INFORMATION=*DUMPED-SYSTEM-FILE ————————————  (1)
//END
/SHOW-FILE-ATTR *IOHIOSDP*,CREATION-DATE=*TODAY ——————————————————————  (2)
```

(1) Die Section IOHIOSDP wird angezeigt. Auf sie kann ausschließlich die Funktion „GEN“ angewendet werden. Das Markieren dieser Funktion und die anschließende Bestätigung mit der DUE-Taste bewirkt, dass DAMP diese Daten extrahiert und sie in einer PAM-Datei in BS2000 ablegt. Der Name dieser Datei wird in den ersten beiden Zeilen des DAMP-Bildschirms angezeigt.

(2) Der Name der soeben erzeugten Datei wird angezeigt.

Diese Datei kann nun folgendermaßen von BS2000 nach X2000 übertragen werden:

```
/START-FTP
open <system> ————————————————————————————————————————————————————  (1)
<userid> —————————————————————————————————————————————————————————  (2)
<password>———————————————————————————————————————————————————————  (3)
bin———————————————————————————————————————————————————————————————  (4)
put <filename> <tar_archive_name>————————————————————————————————  (5)
quit
```

(1) Aufbau der Verbindung zum X2000-System unter Angabe der IP-Adresse oder des symbolischen Namens des Systems.

(2) Eingabe der Kennung im X2000-System, auf die die Daten übertragen werden sollen.

(3) Eingabe des Passwortes dieser Kennung.

(4) **Binäre** Übertragung der PAM-Datei ist erforderlich, da andernfalls die Dateistruktur zerstört wird.

(5) Übertragung der Datei `<filename>` in die Datei `<tar_archive_name>` auf dem Zielsystem. `<tar_archive_name>` darf keine Katalog- oder Benutzerkennung enthalten.

Zur weiteren Verarbeitung der Datei sind unter X2000 folgende Schritte notwendig:

```
tar tf <tar_archive_name> ————————————————————————————————————————  (1)
tar xf <tar_archive_name> <filename> —————————————————————————————  (2)
uncompress <filename> ————————————————————————————————————————————  (3)
```

(1) Inhalt des tar-Archivs mit dem Namen `<tar_archive_name>` auflisten

(2) Gewünschte Datei auswählen und aus dem tar-Archiv extrahieren

(3) Bei Bedarf können jetzt die Diagnosedaten noch dekomprimiert werden

Danach können die Diagnose-Dateien wie gewohnt weiterverarbeitet werden:

- IOHDUMP kann mit MDEBUG bearbeitet werden (IOHDUMP wird mit `sd` als Dumpfile eröffnet):
- SYSDB-Trace und PRKDUMP können mit `exs` bzw. `exp` extrahiert werden; auf die IOH-Daten kann mit speziellen MDEBUG-Anweisungen zugegriffen werden.

11 SNAP-Dump

Der Dumperzeuger SNAP ist zwischen CDUMP und SLED einzuordnen. SNAP ist Bestandteil von BS2000 OSD /BC.

Während der Dumperstellung durch CDUMP läuft das Betriebssystem weiter und kann unter Umständen sicherzustellende Daten verfälschen. SLED beendet das Betriebssystem; man benötigt zur Datensicherstellung und zum Wiederanlauf des Systems relativ viel Zeit.

SNAP hält das Betriebssystem nur für maximal 24 Sekunden an, stellt bestimmte Speicherbereiche sicher (s.u.) und setzt das Betriebssystem anschließend wieder in Gang. SNAP arbeitet unabhängig von BS2000 und verfälscht die sicherzustellenden Diagnoseinformationen daher nicht.

SNAP wird vom Betriebssystem aus dem Zustand TPR oder SIH über die $SNAP-Schnittstelle aufgerufen. Das geschieht im Allgemeinen dann, wenn sich ein inkonsistenter Betriebssystemzustand einstellt, der jedoch nicht so schwerwiegend ist, dass der Systemlauf beendet werden muss.

Mit dem Systemparameter SNAPTIME kann gesteuert werden, wann SNAP die Steuerung wieder an BS2000 abgibt. Der Default-Wert ist 24 Sekunden. Dies entspricht dem Maximalwert bei SNAP, da sonst der Systemzustand „BS2000 beendet“ eintreten könnte. Abhängig von SNAPTIME kann der SNAP-Dump nur einen begrenzten Umfang haben. Der maximale Dumpumfang hängt in erster Linie von der Datenrate der Ein-/Ausgabe und von der Geschwindigkeit der Platte, auf der die Dump-Ausgabedatei $TSOS.SNAPFILE enthalten ist, ab. Mit den von SNAP unterstützten Servern und Platten kann maximal 1 GB sichergestellt werden.

Ein SNAP-Dump beinhaltet folgende Daten:

- Klasse-1-, Klasse-3- und wahlweise residenter Klasse-4-Speicher (resident bedeutet in diesem Zusammenhang: Die Seite befindet sich im Hauptspeicher)

  > **i** In BS2000/OSD-BC ab V9.0 ist auch Hauptspeicher oberhalb 2 GB und damit ggf. der gesamte residente Klasse-4-Speicher im SNAP-Dump enthalten.

- Namens- und Einsprungsliste aller Betriebssystemmodule (EOLDTAB)
- Verwaltungsdaten
- Hardware-Status-Register der laufenden logischen Maschine

Der Ablauf des Programms SNAP wird durch Meldungen an der Konsole protokolliert.Die Meldungen von SNAP haben die Meldungsklasse NSP. Informationen über einzelne Meldungen erhalten Sie im laufenden Betrieb mit dem Kommando HELP-MSG-INFORMATION.

Die SNAP-Funktion wird insbesondere auch vom Dumperzeuger CDUMP genutzt, um den Klasse-1-, Klasse-3- und residenten Klasse-4-Speicher konsistent (unverfälscht) zu sichern.

11.1 SNAP-Dateien

Damit SNAP ablaufen kann, muss auf dem Home-Pubset eine Systemdatei mit dem Namen SNAPFILE eingerichtet sein. Sie muss mindestens 16 MB groß sein, die maximale Größe ist 1 GB. Empfohlen wird ein Wert von mindestens 144 MB. Sie darf nicht von einem anderen Pubset kopiert werden.

Die SNAPFILE wird bei der Systemgenerierung mit dem Dienstprogramm SIR oder bei der Aktivierung von SNAP mit dem Kommando ACTIVATE-SNAPSHOT angelegt. Sie wird unter der Benutzerkennung TSOS eingerichtet. Änderungen an der SNAPFILE (einrichten, Größe ändern, löschen) dürfen im laufenden Betrieb nur mit den Kommandos ACTIVATE-SNAPSHOT und DEACTIVATE-SNAPSHOT vorgenommen werden, siehe das Handbuch „Kommandos“ [8].

Wenn die SNAPFILE nicht vorhanden ist, dann wird sie beim Startup (Parameter SNAP-ACTIVE-SWITCH=ON) in Standardgröße bzw. bei Ausführung des Kommandos ACTIVATE-SNAPSHOT in der angegebenen Größe angelegt.

SNAP schreibt die Diagnosedaten in die SNAPFILE. Um für aufeinander folgende SNAP-Aufrufe diese Datei wieder geleert und verfügbar zu haben, wird der Inhalt der SNAPFILE nach Reaktivierung von BS2000 automatisch und mit hoher Priorität in eine dynamisch erzeugte Dumpdatei geschrieben. Solche Dumpdateien sind unter der Benutzerkennung SYSSNAP katalogisiert und erhalten den Namen

```
SNAP.snap-id.datum.uhrzeit
```

Dabei bedeuten

SNAP	Identifikation als SNAP-Ausgabe
snap-id	7-stellige id aus dem zugehörigen SNAP-Aufruf
datum	Datum in der Form: yyyy.mm.dd
uhrzeit	Uhrzeit in der Form: hh.mm.ss

Die Benutzerkennung SYSSNAP wird beim FIRST-STARTUP automatisch eingerichtet. Der verfügbare Plattenspeicher für diese Kennung sollte mindestens doppelt so groß sein wie die Größe der $TSOS.SNAPFILE.

Die Datei $TSOS.SNAPFILE kann von keinem Dumpauswerter direkt verarbeitet werden. Beim Transfer der Daten von $TSOS.SNAPFILE nach $SYSSNAP entsteht ein Dateiformat, das der Dumpauswerter DAMP verarbeiten kann. Ist das System nicht mehr in der Lage, die Datei $TSOS.SNAPFILE nach $SYSSNAP zu übertragen, wird diese mit dem nächsten Startup oder bei ACTIVATE-SNAPSHOT automatisch in eine verarbeitbare Datei nach $SYSSNAP übertragen. Bei einem DRV-Home-Pubset kann die SNAPFILE-Datei in der nächsten Session möglicherweise nicht mehr umgewandelt werden; dieser halb fertige SNAP-Dump geht dann verloren.

Die SNAPFILE-Datei wird verworfen, wenn sie in der vorherigen Session im Rahmen eines Systemdumps (CDUMP) erstellt wurde.

Das Kopieren bzw. das Umwandeln der SNAPFILE wird nur bei aktivierter SNAP-Funktion durchgeführt.

Solange die SNAPFILE nicht geleert ist, wird kein weiterer SNAP-Aufruf angenommen.

11.2 Aktivieren und Deaktivieren von SNAP

Der Parameter SNAP-ACTIVE-SWITCH=ON/OFF im Startup-Parameterservice legt fest, ob SNAP in der laufenden Session sofort aktiviert wird, siehe Handbuch „Systembetreuung“ [6]. Bei SNAP-ACTIVE-SWITCH=OFF steht SNAP für diese Session zunächst nicht zur Verfügung. SNAP-Aufrufe werden mit einem entsprechenden Returncode beendet. SNAP kann später dynamisch durch die Kommandos ACTIVATE- und DEACTIVATE-SNAPSHOT beliebig oft aktiviert oder deaktiviert werden.

Die Kommandos zu SNAP sind im Handbuch „Kommandos“ [8] detailliert beschrieben.

Kommando	**Bedeutung**
ACTIVATE-SNAPSHOT	Dump-Erzeuger SNAP aktivieren
DEACTIVATE-SNAPSHOT	Dump-Erzeuger SNAP deaktivieren
SHOW-SNAPSHOT-STATUS	Informationen über SNAP ausgeben

Tabelle 21: Kommandoübersicht für SNAP

Die Kommandos ACTIVATE- und DEACTIVATE-SNAPSHOT werden asynchron ausgeführt. Die Meldung `NSP4000` bestätigt die korrekte Annahme des Kommandos. Mit dem Kommando SHOW-SNAPSHOT-STATUS können Sie die geänderten Einstellungen prüfen.

11.3 Einschränkungen

SNAP läuft auf allen BS2000-Servern ab.

Die bisher gesicherten Daten werden abgeschlossen und der fragmentarische SNAP-Dump dem Diagnostiker in folgenden Fällen zur Verfügung gestellt:

- Die von SNAP sicherzustellenden Daten (maximal 1 GB) können nicht innerhalb der von SNAPTIME eingestellten Zeit (Voreinstellung: 24 sec) geschrieben werden
- Die SNAPFILE-Datei ist zu klein
- Ein interner Fehler tritt auf

Es wird die Meldung `NSP1010` ausgegeben. Bei schwerwiegenden Fehlern im SNAP wird der SNAP-Vorgang abgebrochen.

Es gibt in BS2000 Systeminstanzen, die erkennen, ob das BS2000-System länger als eine bestimmte Zeit arbeitsunfähig ist. SNAP deaktiviert BS2000, um unter einer SNAP-eigenen Ablaufsteuerung Diagnosedaten zu sichern. Dies bedeutet aber noch nicht, dass BS2000 lebensunfähig ist. Bei der Festlegung von Zeitintervallen für die Ausfallerkennung eines System muss also berücksichtigt werden, dass trotz der Inaktivität von BS2000 das SNAP-EXEC als Stellvertreter für BS2000 noch arbeitet.

Insbesondere sei hier auf das Produkt HIPLEX MSCF hingewiesen, bei dem die Zeiten für die Lebendüberwachung eines BS2000-Systems mit dem MSCF-Konfigurationsparameter FAIL-DETECTION-LIMIT festgelegt werden können.

BS2000 OSD/BC und SNAP unterstützen einen Arbeitsspeicher von mehr als 2 GB. Damit ist neben dem Klasse-1- und Klasse-3-Speicher ggf. auch der gesamte residente Klasse-4-Speicher im SNAP-Dump enthalten, wenn das im Systemparameter SNAPTIME eingestellte Zeitlimit nicht überschritten wird und die SNAPFILE-Datei ausreichend groß ist. Der Systemparameter SNAPTIME ist auf das Intervall 8 bis 24 Sekunden begrenzt. Bei Einstellung eines zu kleinen Wertes wird auf 8 Sekunden aufgerundet, bei einem zu großen Wert auf 24 abgerundet.

11.4 Automatischer SNAP

Der SNAP wird im automatischen Modus aktiviert, wenn beim Systemabbruch die Funktion „automatischer Restart“ eingeschaltet ist und SNAP als Dumperzeuger eingestellt ist (siehe Kommando SET-RESTART-OPTIONS MODE=*ON(..., DUMP=*SNAP)).

Der automatische SNAP bietet die Möglichkeit, ohne Operatoreingriffe einen Speicherauszug zu erstellen und anschließend das System neu zu laden. Gegenüber dem automatischen SLED bietet der automatische SNAP den Vorteil, dass das System nach kurzer Zeit wieder neu geladen wird und dass trotzdem die wichtigsten Diagnosedaten gesichert wurden.

Mit dem Kommando SHOW-RESTART-OPTIONS können Informationen über die Steuerung des automatischen Restarts ausgegeben werden.

12 TRACE-MANAGER Diagnoseinformation während des Systemlaufs sammeln

Der TRACE-MANAGER erfasst zentral alle dezentral realisierten Komponententraces. Unter „Trace“ (Ablaufverfolger) versteht man hier das Sammeln von Diagnoseinformationen während des Systemlaufs an vorab definierten diskreten Orten des Systems. Die Information besteht dabei aus aufeinander folgenden Einzelsätzen mit komponenten- oder systemspezifischem Inhalt, wie Systemzustandsvariable, Programmdaten, Parameterliste, Zeitstempel etc. Traces werden bei der Erstellung von Speicherabzügen ebenfalls sichergestellt und verbessern dadurch die Möglichkeiten der BS2000-Systemdiagnose im Rahmen der Dumpauswertung.

Die Traces können tasklokal oder systemglobal erstellt werden. Die Speicherbereiche zur Ablage der Information (Tracepuffer) liegen dementsprechend im Task- oder im Systemadressraum. Tasklokale Traces besitzen einen Tracepuffer und dazugehörige Verwaltungsdaten innerhalb einer jeden Task. Systemglobale Traces benötigen nur einen Puffer und einen Satz von Verwaltungsdaten für das gesamte System.
Der TRACE-MANAGER verwaltet die Tracepuffer nicht selbst, sondern hält nur Verweise auf diese in seinen Verwaltungstabellen. Daher müssen die Trace-Eigentümer Lage und Größe der Puffer dem TRACE-MANAGER mitteilen. Da bei Traces in der Regel sehr viele Informationen in sehr kurzer Zeit anfallen, werden die Tracepuffer zyklisch überschrieben. Die im Puffer enthaltenen Daten werden mit SLED oder CDUMP sichergestellt und mit DAMP ausgewertet.

Der TRACE-MANAGER unterstützt die Auswertung durch Bereitstellung der Trace-Dump-List (TDL), die Pufferdeskriptoren und einige Tracebeschreibungsdaten (Trace-ID und Adresse des nächsten bzw. des letzten Eintrags) enthält. Bei einem Speicherabzug mit SLED kann der TRACE-MANAGER diese Liste nicht erstellen, da das Betriebssystem nach dem Laden von SLED nicht mehr arbeitet. In diesem Fall übernimmt das Auswerteprogramm DAMP diese Aufgabe.

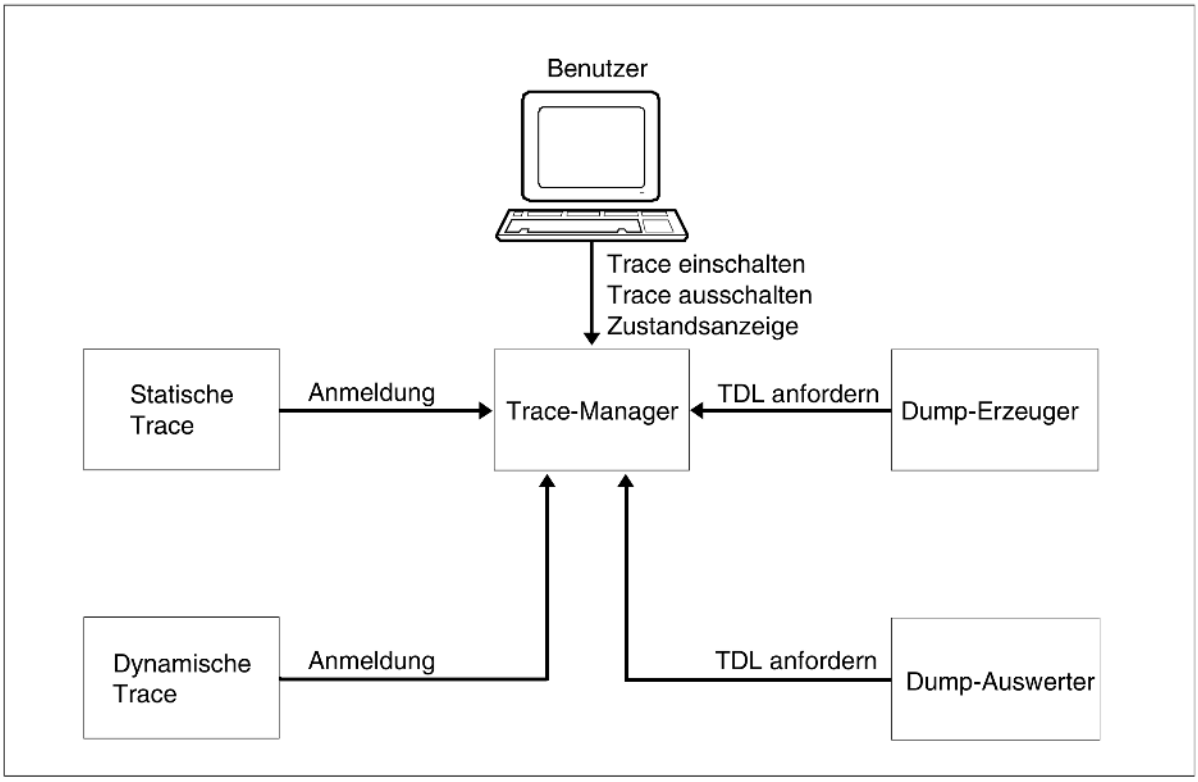


Bild 101: TRACE-MANAGER-Funktionen

Zur zentralen Verwaltung der Traces werden unter der Kontrolle des TRACE-MANAGER verschiedene Daten in Tabellen geführt. Diese Daten lassen sich in drei funktionell verschiedene Gruppen einteilen, die in verschiedenen Tabellen geführt werden. Die Verwaltungsdaten unterscheiden sich in

- Adresslisten,
- Betriebsdaten und
- Beschreibungsdaten.

Wie diese Daten und Tabellen miteinander verkettet sind, zeigt Bild 102.

Zentrale Verwaltungstabellen sind die Adresslisten. Die Adressliste für systemglobale Traces ist in der zentralen BS2000-Tabelle XVT verankert, eine taskspezifische Adressliste für die tasklokalen Traces ist im TCB der jeweiligen Task verankert. Die Adressliste enthält pro Trace einen Satz fester Länge (8 Byte), bestehend aus Adresse des Betriebsdatenblocks sowie einigen Trace-Zustandsanzeigen. Der Adressliste ist ein Listenkopf (Header) vorangestellt, der neben tabellenbeschreibenden Daten auch die Adresse der TDL enthält.

Jeder definierte Trace erhält einen Eintrag in der TDL, in welchem u.a. die Adresse des Tracepuffers und der Name des Trace und Subsystems sowie die Version des Subsystems enthalten sind.

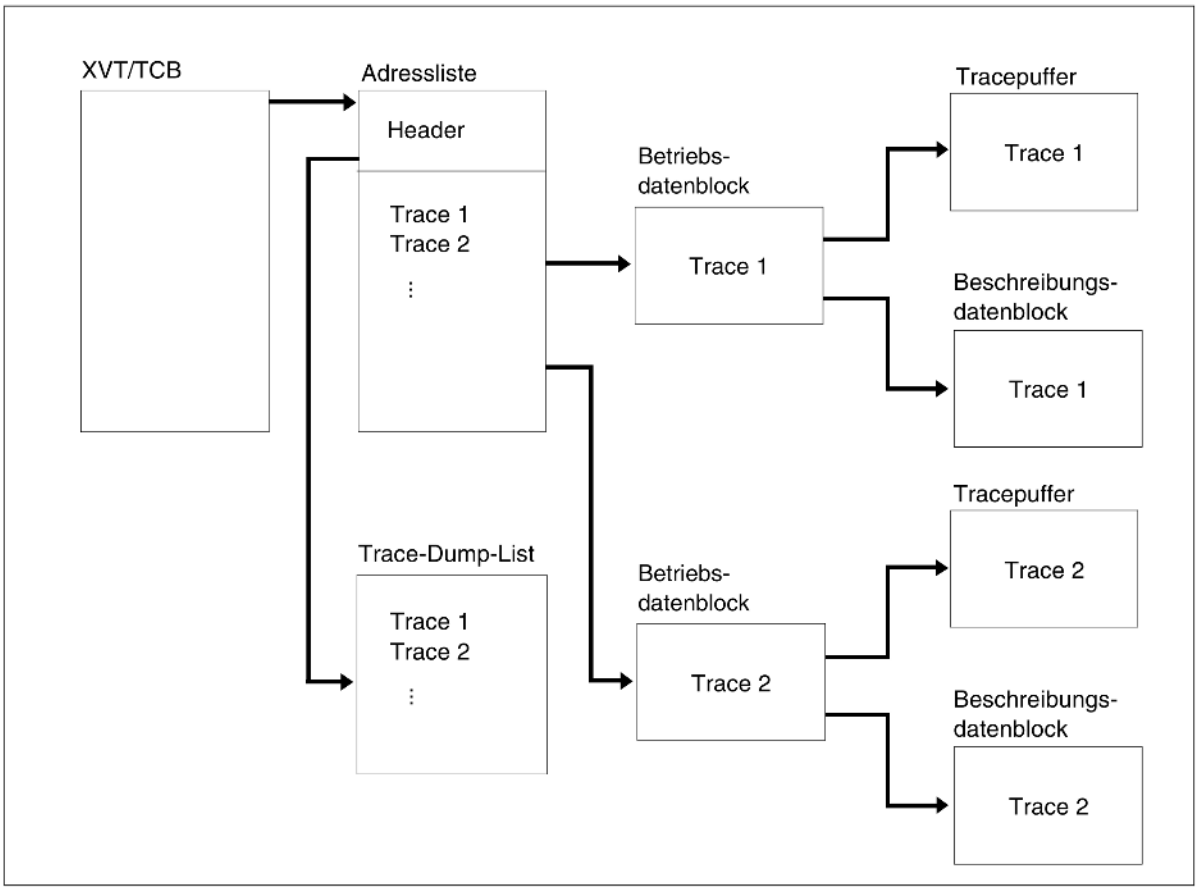


Bild 102: TRACE-MANAGER-Tabellen

Der Betriebsdatenblock enthält alle Daten, die benötigt werden, um möglichst effizient auf den Tracepuffer bzw. auf den nächsten freien Satz im Tracepuffer zugreifen zu können (Pufferdeskriptoren). Im Betriebsdatenblock ist für jede Task die Adresse des entsprechenden Beschreibungsdatenblocks hinterlegt, der an den Betriebsdatenblock gekettet ist. Dieser enthält alle Daten, die zur Verwaltung durch den TRACE-MANAGER sowie zur Aufbereitung der Tracepuffer benötigt werden, wie Beschreibung des Tracepuffer-Layouts, die Trace-ID usw.

Die systemglobalen Traceverwaltungstabellen werden während der Systemeinleitung eingerichtet. Dabei wird der TRACE-MANAGER aufgerufen und initialisiert dann alle für die Verwaltung von systemglobalen Traces erforderlichen Tabellen.
Es besteht die Möglichkeit, neue Traces nachträglich (nach Systemeinleitung) mit den entsprechenden Makros dynamisch zu definieren und zu initialisieren. Hierfür enthalten die Traceverwaltungstabellen Dummy-Einträge fester Länge, um neue Traces aufnehmen zu können (`ï` dynamisch definiert).

Zum Einrichten der tasklokalen Verwaltungstabellen wird der TRACE-MANAGER vom Basissystem bei jeder Erzeugung einer neuen Task aufgerufen. Wird die Task beendet, wird der TRACE-MANAGER ebenfalls aufgerufen, um die tasklokalen Verwaltungstabellen freizugeben. Während eines System-RESTART werden die tasklokalen Verwaltungstabellen gelöscht und durch jene Tabellen ersetzt, die zum Checkpoint-Zeitpunkt gespeichert wurden.
Auch tasklokale Traces können nachträglich dynamisch definiert und initialisiert werden. Die systemglobale Trace-Adressliste verfügt hierzu über freie Einträge für die tasklokalen Traces. Der für die tasklokalen Traceverwaltungstabellen notwendige Speicherplatz wird während deren Installation initialisiert.

Neben der dynamischen Definition von Traces ist auch die dynamische Unterdrückung der Traces in der SHUTDOWN-Routine des Subsystems möglich.

Der TRACE-MANAGER benötigt für die Definition eines Traces Informationen, die er in den systemglobalen Verwaltungstabellen hinterlegt. Diese Informationen können dem TRACE-MANAGER statisch (Daten sind in den Trace-Tabellen als Code bereits vorhanden) oder dynamisch (mit Makro) übergeben werden.

Der TRACE-MANAGER wird über folgende Operator- oder Systemverwalterkommandos gesteuert. Die Kommandos sind im Handbuch „Kommandos“ [8] detailliert beschrieben.

Kommando	**Beschreibung**
SHOW-TRACE-STATUS	Informationen über System-Traces ausgeben
START-TRACE	Trace aktivieren
STOP-TRACE	Trace deaktivieren

Tabelle 22: Übersicht über die Kommandos für den TRACE-MANAGER

13 Online-Wartung

Die Online-Wartung umfasst folgende Aufgaben:

- Führen und Auswerten der Hardware-Fehlerstatistikdatei (siehe Dienstprogramm ELSA im Handbuch „ELSA“ [3]).
- Ablauf von Statistik- und Trace-Programmen unter Steuerung von BS2000 parallel zu den Benutzerprogrammen.

Die Online-Wartung in BS2000 erfolgt unter einer Benutzerkennung mit dem Privileg HARDWARE-MAINTENANCE. Standardmäßig ist dies die Benutzerkennung SERVICE, die mit der gleichnamigen Abrechnungsnummer beim First-Start eingerichtet wird.

Zusätzlich hat jeder Dateieigentümer die Möglichkeit, für eine Benutzerkennung mit dem Privileg HARDWARE-MAINTENANCE dateispezifisch den Zugriff auf mehrbenutzbare Dateien zu steuern.
Dies ist mit dem Operanden `USER-ACCESS` des Kommandos `/MODIFY-FILE-ATTRIBUTES` (oder Makro CATAL) möglich. Wenn für eine Datei die Zugriffsberechtigung `SPECIAL` vergeben wird, dann ist sie für alle Benutzerkennungen **einschließlich** der Kennung(en) mit dem Privileg HARDWARE-MAINTENANCE zugänglich.

Sollen unter SERVICE katalogisierte Programme (z.B. ELSA) unter einer Kennung mit dem Systemprivileg HARDWARE-MAINTENANCE ablaufen, so muss auch für diese Programme im Kommando `/MODIFY-FILE-ATTRIBUTES` der Operandenwert `USER-ACCESS=*SPECIAL` vereinbart werden.
Diese Zugriffserlaubnis wird durch ein weiteres Kommando `/MODIFY-FILE-ATTRIBUTES` (oder Makro CATAL) mit dem Operanden `USER-ACCESS=*ALL / *OWNER-ONLY` zurückgesetzt.

14 Fehler- und Protokolldateien

Fehler- und Protokolldateien sind für die Systembetreuung wichtige Hilfsmittel beim Aufspüren von Hard- oder Software-Fehlern sowie beim Verfolgen des gesamten Nachrichtenverkehrs zwischen Konsolen, berechtigten Benutzerprogrammen und dem System.

Es werden die Fehlerdateien SERSLOG und HEL sowie die Protokolldateien CONSLOG und RESLOG beschrieben.

Die von den Fehler- und Protokolldateien gelieferten Daten werden durch Diagnosefunktionen und -Programme ausgewertet.

14.1 Hardware Error Logging-Datei HEL

Von der Funktion Error Logging System (ELS) werden folgende, diagnoserelevante Ereignisse erfasst:

- Fehlerereignisse
  - Maschinenfehler
  - Fehler im Zusammenhang mit I/O-Unterbrechungen
  - Fehler in peripheren Geräten
- Sonstige Ereignisse
  - Statistikinformation von peripheren Geräten
  - Statistikinformation der CPU-Firmware
  - Prozessor-Kontext-Sicherung
  - Informationen von Test- und Diagnoseprogrammen (TDP)

Die Ereignisse werden von der Systemtask HEL registriert und in der automatisch erzeugten zentralen Statistikdatei dokumentiert. Der Name der HEL-Datei ist folgendermaßen aufgebaut (yyyy-mm-dd.hhmmss: Erstellungs-Datum und -Uhrzeit):

$TSOS.SYS.HEL.yyyy-mm-dd.hhmmss

Die Auswertung der HEL-Dateien übernimmt das BS2000-Dienstprogramm ELSA.

Die HEL-Task und das Auswerteprogramm von BS2000 (ELSA) können gleichzeitig auf die Statistikdatei zugreifen. ELSA ist ausführlich im Handbuch „ELSA“ [3] beschrieben.

Der Service kann unter einer Benutzerkennung, die mit dem Privileg HARDWARE-MAINTENANCE ausgestattet ist, folgende Kommandos zur Steuerung der Abspeicherung von HEL-Sätzen, zur Steuerung der Überwachung und zur Unterstützung von RemoteServiceverwenden:

Kommando	**Bedeutung**
CHANGE-HEL-FILE	Aktuelle HEL-Datei schließen und eine neue Datei eröffnen
MODIFY-HEL-CHECK	Schwellwertüberwachung steuern
MODIFY-HEL-LOGGING	Abspeicherung der HEL-Sätze steuern
MODIFY-HEL-TELESERVICE-ALARM	Vereinbarungen über Fehlerschwellwerte für Remote Service-Nachricht treffen
SHOW-HEL-CHECK	Informationen über die Schwellwertüberwachung anfordern
SHOW-HEL-LOGGING	Informationen über die Protokollsätze anfordern
SHOW-HEL-STATUS	Allgemeine Informationen über HW-Error-Logging anfordern
SHOW-HEL-TELESERVICE-ALARM	Informationen über die eingestellten Remote Service-Parameter anfordern
START-HEL-LOGGING	Systemfunktion HW-Error-Logging aktivieren und Logging-Datei öffnen

STOP-HEL-LOGGING	Systemfunktion HW-Error-Logging beenden und Logging-Datei schließen

Die Systembetreuung kann unter der Kennung TSOS die Kommandos `/SHOW-HEL-STATUS` sowie `/START-` und `/STOP-HEL-LOGGING` verwenden.

Die Kommandos sind ausführlich im Handbuch „Kommandos“ [8] beschrieben.

14.2 Software Error Logging-Datei SERSLOG

Zur Erleichterung der Diagnose von BS2000-Fehlern wird die Funktion „Software Error Logging“ (SERSLOG) zur Verfügung gestellt. Damit können von verschiedenen Routinen BS2000-Fehlerinformationen in die SERSLOG-Datei geschrieben werden.
Während des STARTUP wird Error Logging aktiviert, wobei die SERSLOG-Datei eröffnet wird. Der Name der SERSLOG-Datei ist folgendermaßen aufgebaut:

SYS.SERSLOG.yyyy-mm-dd.xxx.nn

yyyy-mm-dd	Datum, an dem die Datei eröffnet wurde
xxx	Nummer des aktuellen Systemlaufs
nn	laufende Nummer der SERSLOG-Datei (01-99; beginnt bei Startup immer mit 01)

Die SERSLOG-Datei hat keinen Schreibschutz.

Bei der Systembeendigung wird die SERSLOG-Datei geschlossen und Error Logging beendet. Die (zuletzt) aktuelle SERSLOG-Datei wird von SLED mit ausgegeben.

Error Logging kann nur vom Operator und der Systembetreuung beeinflusst werden. Sie können die SERSLOG-Datei wechseln und Error Logging aktivieren oder deaktivieren. Wird nn größer als 99, wird der Zähler wieder auf 01 gesetzt, wobei die erste SERSLOG-Datei überschrieben wird, sofern das Datum und die Nummer des Systemlaufs übereinstimmen.

Kommando	**Bedeutung**
CHANGE-SERSLOG-FILE	Aktuelle SERSLOG-Datei schließen und neue Datei eröffnen
SHOW-SERSLOG-STATUS	Informationen über den Zustand von SERSLOG und Namen der Protokolldatei anfordern
START-SERSLOG	Funktion SERSLOG aktivieren
STOP-SERSLOG	Funktion SERSLOG beenden

Eine automatische Überwachung von kritischen Systemzuständen, die sich in SERSLOG-Ereignissen widerspiegeln, ist mit ASE möglich. Siehe dazu Kapitel „ASE Auxiliary SERSLOG Extensions".

14.3 Protokolldatei CONSLOG

Der gesamte Nachrichtenverkehr zwischen Konsolen, berechtigten Benutzerprogrammen und dem System wird in einer Protokolldatei aufgezeichnet. Davon ausgeschlossen sind die letzten Meldungen eines Systemlaufs, die beim Zurückschreiben des Home-Pubsets auf die Platte ausgegeben werden. Zusätzlich zum Konsoldialog wird das Kommando CHANGE-CONSLOG-FILE auch protokolliert, wenn es von einer Datensichtstation eingegeben wurde.

Kommando	**Bedeutung**
CHANGE-CONSLOG-FILE	Aktuelle Protokolldatei schließen und neue Datei eröffnen
SHOW-CONSLOG-ATTRIBUTES	Zustand der Systemprotokollierung und Name der Protokolldatei ermitteln
SET-CONSLOG-READ-MARK	Lesezugriff auf die aktuelle CONSLOG-Datei ermöglichen, ohne diese vorher schließen zu müssen
TURN	Auswerten der aktuellen Protokolldatei

Die Protokollierung wird automatisch während der Systemeinleitung gestartet.
Eine Meldung informiert, welche Protokolldatei eröffnet wurde.

Der Dateiname für die Protokolldatei ist abhängig von den Systemparametern NBKESNR und FMTYFNLG. Mit diesen Parametern können für die Kennungen SYSAUDIT oder TSOS die folgenden Dateinamenskonventionen eingestellt werden:

neues Format:	SYS.CONSLOG.jj.mm.tt.xxx.nnn SYS.CONSLOG.jjjj-mm-tt.xxx.nnn
altes Format:	SYS.CONSLOG.jj.mm.tt.xxx.nn SYS.CONSLOG.jjjj-mm-tt.xxx.nn
jj.mm.tt / jjjj-mm-tt	neues Format: Datum, an dem die Datei eröffnet wurde altes Format: Datum, an dem der Systemlauf gestartet wurde
xxx	Nummer des aktuellen Systemlaufs
nnn	laufende Nummer der Protokolldatei (001-999 pro Tag, an dem die CONSLOG-Datei gewechselt wird)
nn	laufende Nummer der Protokolldatei (01-99 pro Systemlauf)

Mit dem Systemparameter NBKESNR kann die Systembetreuung festlegen, ob die CONSLOG-Datei unter der Benutzerkennung TSOS oder SYSAUDIT katalogisiert wird, und ob insgesamt 99 CONSLOG-Dateien pro Systemlauf oder 999 pro Tag erzeugt werden können.

Mit dem Systemparameter FMTYFNLG kann die Systembetreuung festlegen, ob im Namen von Logging-Dateien das Jahr 2-stellig (ohne Jahrhundertangabe, in Form von jj.mm.tt) oder 4-stellig (mit Jahrhundertangabe, in Form von jjjj-mm-tt) hinterlegt wird.

Tritt beim Beschreiben der Protokolldatei ein unbehebbarer DVS-Fehler auf, so wird die aktuelle Protokolldatei geschlossen und eine neue Protokolldatei mit der Seriennummer +1 eröffnet.

Wird die Fehlermeldung

```
DMS0541  PLATTENSPEICHERMANGEL, ....
```

ausgegeben und

- ist der Systemparameter NBLOGENF (Console Logging erzwingen) gesetzt, erhält der Operator zusätzlich die Fragemeldung

  ```
  NBR0953 ERROR DURING CONSOLE-LOGGING PROCESSING. REPLY (R=RETRY; H=HALT).
  ```

  Wird auf der Kennung, auf der die CONSLOG-Datei hinterlegt wird (TSOS oder SYSAUDIT), Speicherplatz geschaffen, kann die Frage mit `R` beantwortet werden. Das Console-Logging wird mit der Protokolldatei Seriennummer +1 fortgesetzt. Datensätze gehen nicht verloren. Gibt der Operator `H` ein, wird der Systemlauf beendet.
- ist der Systemparameter NBLOGENF nicht gesetzt, wird das Console-Logging abgeschaltet. Der Operator wird mit der Meldung `NBR0906` informiert, dass Console-Logging nicht aktiv ist. Wird nun Speicherplatz auf der betreffenden Kennung geschaffen, kann mit dem Kommando CHANGE-CONSLOG-FILE das Console-Logging wieder aktiviert werden. Die Datensätze bis zum Wiedereinschalten der Console Logging gehen verloren. Um anzuzeigen, dass die Protokollierung nicht vollständig ist, wird die neue Protokolldatei mit der Seriennummer +2 eröffnet.

> ¡ Die von der TURN-Verarbeitung aus der SYS.CONSLOG-Datei ausgegebenen Meldungen werden nicht mehr in die Protokolldatei aufgenommen, um mehrfache Protokollierung zu vermeiden.

Außerdem kann die Systembetreuung während des Systemlaufs die aktuelle Protokolldatei schließen und eine neue Protokolldatei eröffnen.
Mit dem Kommando SET-CONSLOG-READ-MARK kann die Systemverwaltung die aktuelle, noch geöffnete Protokolldatei lesbar machen und z.B. kopieren.

Auf alle geschlossenen Protokolldateien kann während des Systemlaufs zugegriffen werden (z.B. mit dem Kommando PRINT-DOCUMENT).
Die maximale Anzahl der Protokolldateien während eines Systemlaufs beträgt 99, wenn die laufende Nummer der Protokolldatei 2-stellig beantragt wurde, bzw. 999 pro Tag, wenn die laufende Nummer 3-stellig beantragt wurde.
Ein automatischer Wechsel der Protokolldatei (z.B. bei Tageswechsel) findet nicht statt. Wird die max. Anzahl überschritten, erhält der Operator eine Meldung und das Beschreiben der Protokolldatei wird bis zum Ende des Systemlaufes eingestellt. Das Überschreiten der maximalen Anzahl der Protokolldateien kann mit dem Sytemparameter NBLOGENF verhindert werden.

Im Abstand von ca. 25 Sätzen wird ein kurzer Datumssatz geschrieben. Findet während eines Systemlaufs ein Tageswechsel statt, so wird dies in der Protokolldatei durch den Satztyp „Tageswechsel“ angezeigt.
Bei der Umstellung von Winter- auf Sommerzeit und umgekehrt wird ein Tageswechselsatz mit der neuen Jahreszeit-Information in die Protokolldatei geschrieben.
Dieser Satztyp wird zusätzlich als erster und letzter Satz in die Protokolldatei eingetragen. Er enthält dann das Datum der Dateieröffnung bzw. -schließung sowie die Anzahl der in dieser Session eröffneten CONSLOG-Dateien.
Wird die Protokolldatei durch einen Fehler abgebrochen, so ist nicht gewährleistet, dass als letzter Satz ein Tageswechselsatz eingetragen ist.
In der letzten Protokolldatei eines Systemlaufs entfällt der letzte Tageswechselsatz.
Das Exportieren des Home-Pubsets kann nicht mehr in der CONSLOG-Datei protokolliert werden.

Der Name der aktuellen Protokolldatei kann mit dem Kommando SHOW-CONSLOG-ATTRIBUTES ausgegeben werden.

Aufbau der Protokolldatei

CONSLOG-Dateien werden immer als SAM-Dateien erzeugt.

Format eines Satzes

Nachricht

Empfänger	Leer-zeichen	Nachrich-tentyp	Absender	-	Auftrags-kennzeichen	. oder # 1)	Tageszeit	Leerzeichen 6)	Text
1-4	5	6	7-10	11	12-14	15	16-21	22	23...

Tageswechselsatz

Leerzeichen	T 2)	CLOG 3)	Leer-zeichen	. oder # 1)	Tageszeit	Leerzeichen	***	Datum 4)	***	Leer-zeichen
1-5	6	7-10	11-14	15	16-21	22-23	24-26	27-36	37-39	40

Fortsetzung:

Anzahl der CONSLOG pro Session	Leerzeichen	***	Leerzeichen	Zeitzone 5)	Leerzeichen	***...***
41-46	47	48-51	52	53-61	62	63-128

1. Ob das Trennzeichen ein Punkt (Voreinstellung) oder ein # ist, wird durch den Systemparameter SECSTART beeinflusst (siehe Handbuch „Einführung in die Systembetreuung“ [6]).
2. T ist das Kennzeichen für Tageswechselsatz
3. CLOG ist die Task, die die Meldungen aller Konsolen in die Protokolldatei schreibt
4. Format des Datums: jjjj-mm-tt oder **jj.mm.tt
   (abhängig vom Systemparameter FMTYFLNG, siehe Handbuch „Einführung in die Systembetreuung“ [6])
5. Abweichung der lokalen Zeit von UTC in Stunden und Minuten; Format der Zeitzonenangabe: UTC±hh:mm
6. Bei Nachrichtentyp „Antwort“ steht an Stelle 22 ein Punkt oder Doppelpunkt (. oder :)

Datumssatz

Jahr	-	Monat	-	Tag
1-4	5	6-7	8	9-10

Im Abstand von 25 Sätzen wird ein Datumssatz eingetragen.
Die Darstellung gilt nur für die Belegung des Systemparameters FMTYFNLG mit dem Wert 4; bei FMTYFNLG = 2 ist das Jahr nur 2-stellig und der Datumssatz entsprechend 2 Byte kürzer.

Für Empfänger und Absender sind folgende Einträge möglich:

- mnemotechnischer Gerätename der Konsole in Klammern

- Anwendungsname
- Berechtigungsschlüssel (nur Empfänger)
- Auftragsnummer (TSN) einer Benutzer- oder Systemtask, z.B. einer OPRT

Für Nachrichtentyp sind folgende Einträge möglich:

%	Systemmeldung, die keine Antwort verlangt
?	Systemmeldung, die eine Antwort verlangt, die auch vom Operator gegeben werden kann
&	Zusatzinformation-Anforderung, verlangt Antwort vom Kommandogeber
;	Systemmeldung, die eine Antwort verlangt, die nur eine Task geben kann
+	Kommandoergebnis
!	Kommandoabschlussmeldung
*	Fehlermeldung
E	Emergency-Ausgabe (Meldung, Frage oder Antwort auf eine Emergency-Frage)
R	Antwort auf eine Frage (Meldungstyp ? oder &)
/	Kommando

Auswerten und Sicherstellen der Protokolldatei

Es gehört zu den Aufgaben der Systembetreuung, die Protokolldateien vorangegangener Systemläufe auszuwerten und zu sichern.
Da die Protokolldatei als SAM-Datei katalogisiert ist, kann sie z.B. mit EDT-Prozeduren ausgewertet werden.

Die aktuelle Protokolldatei lässt sich auch mit dem TURN-Kommando auswerten.
Dabei lassen sich Meldungen nach verschiedenen Merkmalen aussuchen:

- Tag
- Uhrzeit
- Ziel
- Quelle

Bei Einsatz des Software-Produktes SECOS können mit der Komponente SATUT auch CONSLOG-Dateien ausgewertet werden. Die CONSLOG-Meldungen werden dazu in einen SAT-Protokolldatensatz umgewandelt. Der Kurzname für den Ereignis-Typ ist immer CLG (siehe Handbuch „SECOS“ [9]).

Auszüge aus der Protokolldatei

Auszug 1: Systeminformation einholen

```
OPRT /(CB)-000.133328 SH-SYS-INF
(CB) +XACK-000.133328 CONFIGURATION                    = 7.500- S210-30
(CB) +XACK-000.133328 CPU-ID-LIST :    ADR 0           = 2002051222600000
(CB) +XACK-000.133328                  ADR 1           = 2012051222600000
(CB) +XACK-000.133328                  ADR 2           = 2022051222600000
(CB) +XACK-000.133328 HSI-ATT :        TYPE            = IX
(CB) +XACK-000.133328                  ASF             = YES
(CB) +XACK-000.133328                  OPERATION-MODE  = VM2000
(CB) +XACK-000.133328 MEMORY-SIZE                      = 2 GB
(CB) +XACK-000.133328 MINIMAL-MEMORY-SIZE              = 1 GB
```

```
(CB) +XACK-000.133328 BS2000-ID :      NAME              = J10BXS
(CB) +XACK-000.133328                  VERSION           = V21.0A00J1
(CB) +XACK-000.133328                  SERVICE-PACK      = SP 21.1
(CB) +XACK-000.133328                  CREATED           = <date> <time>
(CB) +XACK-000.133328 IOCONF-ID :      NAME              = S2100512
(CB) +XACK-000.133328                  VERSION           = V21.0A00
(CB) +XACK-000.133328                  CREATED           = <date> <time>
(CB) +XACK-000.133328                  FORMAT            = IORSF01
(CB) +XACK-000.133328 IPL-TIME                           = <date> <time>
(CB) +XACK-000.133328 SYSTEM-CONF :    SYSID             = 93
(CB) +XACK-000.133328                  HOME-PUBSET       = SBZ7
(CB) +XACK-000.133328                  HOST-NAME         = D017ZE15
(CB) +XACK-000.133328                  VM-INDEX          = 5
(CB) +XACK-000.133328                  VM-NAME           = VM11SU39
(CB) +XACK-000.133328                  SYSTEM-NAME       = *NONE
(CB) +XACK-000.133328                  SYSPAR-BS2-SEL    = *STD
(CB) +XACK-000.133328                  LIVE-MIG-COUNT    = 0
(CB) +XACK-000.133328 VM2000-VERSION                     = V11.5A
(CB) +XACK-000.133328 VM2000-MONITOR-  BS2000-VERSION    = V20.0A0000
(CB) +XACK-000.133328    SYSTEM:       HOST-NAME         = D017ZE14
(CB) +XACK-000.133328 SYSTEM-TIME-     ZONE              = +01:00
(CB) +XACK-000.133328    PARAMETER:    SEASON            = S
(CB) +XACK-000.133328                  SEASON-DIFFERENCE = 01:00
(CB) +XACK-000.133328                  PREV-CHANGE-DATE  = <date> <time>
(CB) +XACK-000.133328                  NEXT-CHANGE-DATE  = <date> <time>
(CB) +XACK-000.133328                  SYNCHRONIZATION   = SERVER-CONN-EXT-REF
(CB) +XACK-000.133328                  EPOCH             = 08
(CB) ! UCO-000.133328 % NBR0740 COMMAND COMPLETED 'SH-SYS-INF';
                        (RESULT: SC2=000, SC1=000, MC=CMD0001); DATE: <date>
```

Auszug 2: Verschiedene Operatorkommandos eingeben

```
:
OPRT /(CB)-000.110956 SHMSG
  <* % UCO-000.110956  %  NBR0970 OPERATOR TASK WITH TSN 'XACK' CREATED FOR
                          CONSOLE '(CB)'
(CB) +XACK-000.110956  %  NBR0031 NO MESSAGE OUTSTANDING ON THE CONSOLE
(CB) ! UCO-000.110956  %  NBR0740 COMMAND COMPLETED 'SHMSG';
                         (RESULT: SC2=001, SC1=000, MC=CMD0001); DATE: <date>
  <E %XACL-000.111003  %  NBR0797 APPLICATION '@001' CONNECTED WITH '$CONSOLE',
                          PROCESSOR NAME 'D017ZE14', STATION NAME 'OMS00062'
2008-10-18
OPRT /@001-000.111005 REQ-OPER-ROLE SYSADM
  <* % UCO-000.111005  %  NBR0970 OPERATOR TASK WITH TSN 'XACM' CREATED FOR
                          CONSOLE '@001'
@001 +XACM-000.111005  %  NBR0980 OPERATOR ROLE 'SYSADM' ASSIGNED TO
                          OPERATOR ID 'SYSOPR'
@001 ! UCO-000.111005  %  NBR0740 COMMAND COMPLETED 'REQ-OPER-ROLE';
                          (RESULT: SC2=000, SC1=000, MC=CMD0001); DATE: <date>
  <* %DIAA-000.111017  %  TIA0300 $DIALOG APPLICATION CORRECTLY STARTED ON
                          HOST *STDHOST
  <G %IOR1-000.111019  %  NKR0175 CONFIGURATION UPDATE STARTED.
OPRT /@001-000.111019 SH-DEV A007
@001 +XACM-000.111019 MNEM DEV-TYPE CONF-STATE POOL VSN DEV-A PHASE ACTION
@001 +XACM-000.111019 A007 FTAPE1   DETACHED    NO         FREE
```

```
@001 ! UCO-000.111019  %  NBR0740 COMMAND COMPLETED 'SH-DEV';
                          (RESULT: SC2=000, SC1=000, MC=CMD0001); DATE: <date>
OPRT /(CB)-000.111047 ASR ADD,CS=C0,CD=ALL
(CB) ! UCO-000.111047  %  NBR0740 COMMAND COMPLETED 'ASR';
                          (RESULT: SC2=000, SC1=000, MC=CMD0001); DATE: <date>
  <J %0AFW-000.111055  %  JMS0154 'TSOS' LOGGED ON FOR 'PGTD0666/STAT0C33'.
                          JOB NAME 'QE13END'. CALLER '(NONE)'. TID 00020033
OPRT /@001-000.111113 ATT A007
  <G % MSG-000.111113  %  NKR0042 'DEVICE     =A007': ATTACH ACCEPTED
  <G % MSG-000.111113  %  NKR0040 'DEVICE     =A007' ATTACHED
@001 ! UCO-000.111113  %  NBR0740 COMMAND COMPLETED 'ATT';
                          (RESULT: SC2=000, SC1=000, MC=CMD0001); DATE: <date>
  <J %0AFX-000.111122  %  JMS0154 'TSOS' LOGGED ON FOR 'PGTD0666/STAT0C50'.
                          JOB NAME 'QE13END'. CALLER '(NONE)'. TID 00020031
OPRT /@001-000.111128 SH-DEV A007
@001 +XACM-000.111128 MNEM DEV-TYPE CONF-STATE POOL VSN DEV-A PHASE ACTION
@001 +XACM-000.111128 A007 FTAPE1   ATTACHED    NO         FREE
@001 ! UCO-000.111128  %  NBR0740 COMMAND COMPLETED 'SH-DEV';
                          (RESULT: SC2=000, SC1=000, MC=CMD0001); DATE: <date>
  <E %XACO-000.111134  %  NBR0797 APPLICATION '@002' CONNECTED WITH
'$CONSOLE',
                          PROCESSOR NAME 'D017ZE14', STATION NAME 'OMS00068'
  <J %0AFZ-000.111134  %  JMS0154 'TSOS' LOGGED ON FOR 'PGTD0666/STAT0C53'.
                          JOB NAME 'QE13END'. CALLER '(NONE)'. TID 0002002C
  <J %0AFY-000.111205  %  JMS0154 'TSOS' LOGGED ON FOR 'MCP0242C/STAT0690'.
                          JOB NAME 'SQ13LUEN'. CALLER '(NONE)'. TID 0002002D
:
```

14.4 Protokolldatei RESLOG

BS2000-Server bieten die Möglichkeit, über die nominelle Server-Leistung hinaus zeitlich begrenzt weitere, bereits vorinstallierte CPUs zuzuschalten („Extra-CPUs“, siehe Handbuch „Einführung in die Systembetreuung“ [6]).

Das Subsystem RESLOG protokolliert das Zu- und Wegschalten von Extra-CPUs in der RESLOG-Protokolldatei.

Das Subsystem RESLOG

Das Subsystem RESLOG (RESource LOGging) ist Bestandteil des Grundausbaus.

> ¡ Im VM2000-Betrieb prüft RESLOG, ob es im Monitorsystem unter VM2000 läuft und ob Extra-CPUs vorhanden sind.

Das Subsystem RESLOG bietet zwei Kommandoschnittstellen für die Arbeit mit der RESLOG-Protokolldatei an:

Kommando	Bedeutung
CHANGE-RESLOG-FILE	Wechsel der Logging-Datei
START-RESLOG-EVALUATION	Auswertung der Logging-Datei

Die RESLOG-Protokolldatei

RESLOG legt unter der Kennung TSOS eine Logging-Datei mit folgendem Namen an:

SYS.RESLOG.<server-id>

Die <server-id>, abgeleitet aus der CPU-ID, identifiziert einen Server weltweit eindeutig. Mit dem Kommando SHOW-SYSTEM-INFORMATION können alle CPU-IDs einer Konfiguration ausgegeben werden. Sie unterscheiden sich nur durch eine fortlaufende Nummerierung voneinander, die sich je nach Architektur an unterschiedlichen Stellen der CPU-ID befindet. Diese Stellen werden mit „0“ belegt: Das Ergebnis ist die Server-ID.

Diese Logging-Datei bleibt normalerweise während der gesamten Session geöffnet. Eine bereits vorhandene Logging-Datei wird fortgeschrieben, falls ihre Server-ID identisch ist.

Die Logging-Datei kann mit dem Kommando CHANGE-RESLOG-FILE gewechselt werden.
Die aktuelle Datei wird dann geschlossen und in SYS.RESLOG.<server-id>.<date> mit Datum des Close-Zeitpunktes umbenannt. Anschließend wird eine neue Datei mit dem Namen SYS.RESLOG.<server-id> geöffnet.

Die Logging-Datei enthält verschiedene Datensätze:

- Mit dem Starten von RESLOG wird immer als Erstes ein Start-Datensatz geschrieben. Der Start-Datensatz enthält Datum, Uhrzeit, BS2000- und RESLOG-Version. Er enthält auch die Server-ID, die Anzahl der Extra-CPUs und die CPU-Nummern der Extra-CPUs, die zugeschaltet (ATTACHED) sind.
- Beim Beenden des Subsystems (also auch bei Shutdown) und beim Wechseln der Logging-Datei wird als letzter Datensatz ein Stop-Datensatz geschrieben. Der Stop-Datensatz enthält neben Datum und Uhrzeit auch den Grund für das Schließen der Logging-Datei (Stop/Change).
- Mit jedem ATTACH-/DETACH-DEVICE bzw. ATTACH-/DETACH-VM-RESOURCES einer Extra-CPU wird ein CPU-Datensatz mit Datum, Uhrzeit, CPU-Nummer und ATTACHED/DETACHED-Kennzeichen geschrieben.

- Als weiterer Datensatz wird einmal pro Stunde der Alive-Datensatz geschrieben bzw. aktualisiert, damit bei einem Systemabsturz noch konsistente Daten zur Verfügung stehen.
  Der Alive-Datensatz befindet sich normalerweise nur in der geöffneten Datei und wird von einem CPU- oder Stop-Datensatz überschrieben.
  Lediglich im Crash-Fall kann sich als letzter Datensatz ein Alive-Datensatz in der Datei befinden, der beim nächsten Starten von RESLOG nicht überschrieben wird.

Auswerten der RESLOG-Protokolldatei

Mit dem Kommando START-RESLOG-EVALUATION wird die RESLOG-Auswertung gestartet. Das Ergebnis kann in komprimierter oder detaillierter Form am Bildschirm oder in eine Datei ausgegeben werden.

Die RESLOG-Auswertung läuft als ein eigenes Programm ab, das intern geladen, gestartet und beendet wird.

> **Achtung!**
> Wird die RESLOG-Auswertung aus einem anderen Programm heraus aufgerufen (z.B. nach Unterbrechung mit der K2-Taste), wird dieses Programm entladen. Es ist nicht möglich, nach Beendigung der RESLOG-Auswertung in dieses Programm zurückzukehren (z.B. mit //RESUME).

Der Systembetreuer bestimmt Ausgabemenge und Ausgabeziel der RESLOG-Auswertung. Dazu stehen ihm die verschiedenen Operanden des Kommandos START-RESLOG-EVALUATION zur Verfügung:

- Operand RESLOG-FILE=*CURRENT/<dateiname>/*FROM-FILE(...)
  Es können die aktuelle, eine einzelne oder eine Liste von RESLOG-Dateien ausgewertet werden.
  Bei Angabe einer Liste von Dateien gelten folgende Bedingungen:
  - In einem Auswertelauf können nur RESLOG-Dateien eines Servers (also RESLOG-Dateien mit derselben Server-ID) ausgewertet werden.
  - Die RESLOG-Dateien sind in zeitlich aufsteigender Reihenfolge anzugeben.
- Operand PERIOD=*INTERVAL(...)
  Die Auswertung kann über den gesamten oder einen definierten Zeitraum erfolgen.
- Operand INFORMATION=*SUMMARY/*ALL
  Die Ausgabe erfolgt als kurze Zusammenfassung oder zusätzlich mit der Auflistung jedes einzelnen ATTACH /DETACH-Vorgangs.
- Operand OUTPUT=*SYSOUT/<dateiname>
  Die Ausgabe erfolgt nach SYSOUT oder in eine Datei.

Beispiel 1: Summary-Ausgabe der aktuellen RESLOG-Protokolldatei nach SYSOUT

```
/START-RESLOG-EVALUATION [RESLOG-FILE=*CURRENT,INF=*SUMMARY,OUTPUT=*SYSOUT]
------------------------------------------------------------------------------
                           RESLOG EVALUATION
------------------------------------------------------------------------------
START DATE             : <date>       END DATE  : <date>
------------------------------------------------------------------------------
BS2000 VERSION         : 21.0A00             SERVER-ID : 1002000121900000
RESLOG VERSION         : 21.0A00
------------------------------------------------------------------------------
NUMBER EXTRA CPUS      :       1
NUMBER DAYS IN USE     :       1
------------------------------------------------------------------------------
TIME WITHOUT DATA      :       1 (hours) (2%)
------------------------------------------------------------------------------
```

Bedeutung der Ausgaben:

START DATE	Beginn des Auswertezeitraums
END DATE	Ende des Auswertezeitraums
BS2000 VERSION	Betriebssystemversion zu Beginn des Auswertezeitraums
RESLOG VERSION	Version der RESLOG-Datei zu Beginn des Auswertezeitraums
SERVER-ID	weltweit eindeutige Server-ID des Servers
NUMBER EXTRA CPUS	Anzahl der Extra-CPUs zu Beginn des Auswertezeitraums
NUMBER DAYS IN USE	Anzahl der Tage, an denen Extra-CPUs genutzt wurden; Es wird jeder Kalendertag für jede Extra-CPU gezählt, an dem diese ATTACHED war.
TIME WITHOUT DATA	Zeit in Stunden/Tage, in denen RESLOG nicht aktiv war; Es werden die Stunden/Tage zwischen den STOP- und den folgenden START-Datensätzen gezählt. Zusätzlich wird die Zeit als Prozentwert, bezogen auf den gesamten Auswertezeitraum, ausgegeben.

Beispiel 2:
Detaillierte-Ausgabe der aktuellen RESLOG-Protokolldatei in die Datei PROT.EXTRA-CPU.002

```
/START-RESLOG-EVALUATION INFORMATION=*ALL, OUTPUT=PROT.EXTRA-CPU.002

:
<Ausgabe wie bei INF=*SUMMARY, siehe Beispiel 1> :
------------------------------------------------------------------------
                    DETAILED STATISTICS OF EXTRA-CPUS
--------+-------------------------+-------------------------+----------------
 CPU NR !      ATTACH TIME        !      DETACH TIME        !     DURATION
--------+-------------------------+-------------------------+----------------
  02    !  <date>       <time>    !  <date>          <time> !      1 (days)
        !                         !                         !
  02    !  <date>       <time>    !  <date>          <time>*!      1 (days)
        !                         !                         !
NO DATA !  <date>       <time>    !  <date>          <time> !      1 (hours)
------------------------------------------------------------------------
```

Bedeutung der Ausgaben:

CPU NR	CPU-Nummer der Extra-CPU oder – wenn RESLOG nicht geladen ist – NO DATA
ATTACH TIME/ DETACH TIME	Datum und Uhrzeit des ATTACH- bzw. DETACH-Vorgangs einer Extra-CPU (falls sich unter CPU NR eine CPU-Nummer befindet) oder Datum und Uhrzeit zu Beginn eines Zeitraums, in dem RESLOG nicht aktiv war (falls sich unter CPU NR der Text NO DATA befindet)
DURATION	Zeitraum in Stunden oder Tagen, bezogen auf ATTACH/DETACH TIME • Ausgabe für Extra-CPU: Es wird die Anzahl der Kalendertage des Zeitraums zwischen ATTACH und DETACH der Extra-CPU ausgegeben. • Ausgabe für NO DATA: Es wird die Anzahl Stunden oder Tage (abgerundet) des Zeitraums ausgegeben, in dem RESLOG nicht aktiv war.

Hinter den Zeiten bei ATTACH TIME und DETACH TIME kann ein * (Stern) ausgegeben werden mit folgenden Bedeutungen:

- Der genaue Zeitpunkt des ATTACH ist nicht bekannt, weil beim Starten von RESLOG die Extra-CPU bereits ATTACHED war.
- Der genaue Zeitpunkt des DETACH ist nicht bekannt, weil die Datei geschlossen oder gewechselt oder das Subsystem abnormal beendet wurde, während eine Extra-CPU ATTACHED war.
- Der tatsächliche Zeitpunkt des ATTACH/DETACH wird nicht ausgegeben, weil der Auswertezeitraum nach einem ATTACH beginnt und/oder vor einem DETACH endet.
- Es wird die aktuelle Datei ausgewertet und eine Extra-CPU ist ATTACHED.
  Bei DETACH TIME wird das Datum des letzten Alive- oder CPU-Datensatzes geschrieben.

Meldungen mit Remote Service

Zur Kontrolle der vertraglich vereinbarten Nutzungsdauer von Extra-CPUs werden alle ATTACH/DETTACH-Vorgänge für Extra-CPUs via Remote Service an den Hersteller gemeldet. Diese Meldungen dienen auch einer möglichen Überprüfung der von RESLOG protokollierten Daten.
Im Normalfall soll aber eine Überprüfung der vertraglich vereinbarten Nutzungsdauer nur anhand der RESLOG-Daten erfolgen. Deshalb muss sichergestellt werden, dass RESLOG ständig läuft.

Auch RESLOG schickt Meldungen via Remote Service an den Hersteller, falls es sich im Fehlerfall abnormal beendet. Folgende Meldungen sind dafür vorgesehen:

```
NPR0001  SUBSYSTEM RESLOG KONNTE NICHT INITIALISIERT WERDEN
         Nähere Informationen siehe SERSLOG.

NPR0002  SUBSYSTEM RESLOG IST NACH ERFOLGREICHER INITIALISIERUNG ABNORMAL
         BEENDET WORDEN.
         Nähere Informationen siehe SERSLOG.
```

15 Abkürzungen

ACS	Alias Catalog Service
AID	Advanced Interactive Debugging
ASE	Auxiliary SERSLOG Extensions
CCB	Channel Control Block
CCW	Channel Command Word
CLTF	Common Log Task Facility
CPU	Central Processing Unit
CSECT	Coding Section
DAB	Disk Access Buffer
DCAM	Data Communication Access Method
DCM	Data Communication Methods
DMS	Data Management System
DSECT	Dummy Section
DSSM	Dynamic Subsystem Management
DVS	Daten-Verwaltungs-System
EDT	Editor in BS2000
ELFE	Error Logging File Evaluation
ELSN	Error Log Sequence Number
EOLDTAB	Systemmodultabelle
ETPND	Identifikationstabelle des Subsystems
EXVT/XVT	CP
FCB	Executive Vector Table
FDDRL	File Control Block
HSA	Fast Disk Dump and Reload
HSI	Hardware System Area
IPL	Hardware-Software Interface
ISAM	Initial Program Load
	Indexed Sequential Access Method

ITN	Internal Task Number
JCB	Job Control Block
JTBP	Job-to-be-processed-Block
JTBPX	Job-to-be-processed-Block-Extension
NDM	Nucleus Device Management
PAM	Primary Access Method
PCB	Program Control Block
PSA	Processor Save Area
PSA	Prefixed Storage Area
PSW	Program Status Word
SDITT	Start Device Interrupt Trace Table
SERSLOG	Software Error Logging
SIH	System Interrupt Handling
SIR	System Install and Restore
SLED	Self Loading Emergency Dump
SPL	System Programming Language
SVC	Supervisor Call
SVMT	System Virtual Memory Table
TCB	Task Control Block
TDL	Trace Dump List
TFT	Task File Table
TIC	Task in Control
TID	Task Identity
TLT	Task Location Table
TPR	Task privileged
TSN	Task Sequence Number
TTSAV	System Trace Table
TU	Task unprivileged
UVMT	User Virtual Memory Table
VAT	Virtual Attribute Table

16 Literatur

Die Handbücher finden Sie im Internet unter http://manuals.ts.fujitsu.com. Handbücher, die mit einer Bestellnummer angezeigt werden, können Sie auch in gedruckter Form bestellen.

[1] **AID**
Advanced Interactive Debugger
Basishandbuch

[2] **EDT**
Anweisungen
Benutzerhandbuch

[3] **ELSA**
Error Logging System Analysis
Benutzerhandbuch

[4] **BS2000 OS DX**
Makroaufrufe an den Ablaufteil
Benutzerhandbuch

[5] **BS2000 OS DX**
Systeminstallation
Benutzerhandbuch

[6] **BS2000 OS DX**
Einführung in die Systembetreuung
Benutzerhandbuch

[7] **BS2000 OS DX**
Dienstprogramme
Benutzerhandbuch

[8] **BS2000 OS DX**
Kommandos
Benutzerhandbuch

[9] **SECOS**
Security Control System - Beweissicherung
Benutzerhandbuch