CPG

Programmiererhandbuch

I n h a l t s v e r z e i c h n i s Seite

CPG-Information

Dieses Handbuch wird als Anleitung für die Verwendung des CPG herausgegeben.

Anschrift:	Lattwein GmbH Otto-Brenner-Straße 25 52353 Düren
Telefon:	02421 81051
Telefax:	02421 82127
Internet:	http://www.lattwein.de
E-Mail:	service@lattwein.de

Innerhalb der unten angegebenen Arbeitszeit steht den Benutzern des CPG ein zentraler Telefon-Wartungsdienst zur Verfügung, der bemüht ist, alle Fragen zu beantworten, die in diesem Handbuch nicht behandelt sind.

Arbeitszeit:	8:30 – 16:00 Uhr

H L 1 - Hierarchical Language One. 1100

HL1 ist eine Programmiersprache, die anstelle von Programmen einzelne in sich logisch abgeschlossene Programmbausteine erstellt, die vom Programmierer zu Baugruppen kombiniert werden können.

Die Bausteine werden jeweils für sich einzeln kompiliert und in der Bibliothek für ausführbare Programme gespeichert.

HL1-Programme können sowohl für die Online- als auch für die Batch-Verarbeitung eingesetzt werden. Insbesondere kann ein und derselbe HL1-Baustein sowohl aus Online- als auch aus Batchanwendungen aufgerufen werden.

HL1-Programme sind unabhängig von der verwendeten Basissoftware.

HL1-Bausteine sind objektkompatibel, bei einem Wechsel des TP-Monitors oder Datenbanksystems ist deshalb keine neue Umwandlung erforderlich.

HL1 und CPG 1105

HL1 ist eine Weiterentwicklung der Programmiersprache CPG.

CPG ist infolge dessen eine Untermenge von HL1, und zwar der Teil, der sich auf die Erstellung der Programmbausteine bezieht.

Der HL1-Erweiterungsteil regelt die Steuerung und ermöglicht die Verknüpfung und Kommunikation zwischen den einzelnen Bausteinen sowie die Batch-Programmierung.

Die Programmierung der Anwendungsprogramme erfolgt in der Programmiersprache CPG. Bestehende CPG-Programme können in einer HL1-Installation sowohl als eigenständige Programme weiter verwendet, als auch in das HL1-System integriert werden.

Abschnitt 2 Service-Funktionen 2000

Datenfelder 2100

Feldnamen und Felddefinitionen 2110

Jedem Feld in einem CPG-Programm ist ein Bereich in der TWA zugeordnet. Ein Feld kann explizit oder implizit definiert werden.

Die explizite Definition von Feldern erfolgt entweder mit den E- oder den D-Bestimmungen von CPG (D- und E-Bestimmungen schließen sich gegenseitig aus). Felder können implizit in Eingabebestimmungen oder als Ergebnisfeld in den Rechenbestimmungen definiert werden.

Alle in der TWA abgestellten Felder sind entweder alphanumerisch oder gepackt numerisch. Immer wenn ein Feld definiert wird, verlangt CPG die folgenden Angaben:

a) Feldname
b) Feldlänge
c) Format (numerisch oder alphanumerisch)
d) Falls numerisch: Die Anzahl Dezimalstellen.

Feldnamen können bis zu sechs Stellen lang sein. Das erste Zeichen muss ein Buchstabe von A - Z sein oder das Dollarzeichen ($). Die weiteren Zeichen müssen Buchstaben (A - Z) oder Ziffern (0 - 9) sein.

Die Zeichenfolge "CPG" auf den ersten drei Stellen eines Feldnamens ist reserviert. Die Benutzung ist nur mit den im Abschnitt 2130 angegebenen CPG-internen Feldern zugelassen.

Hier ist zu beachten, dass ein Feldgruppenname einschließlich Komma und Subskript maximal sechs Stellen lang sein kann. Dies bedeutet effektiv, dass Feldgruppennamen maximal vierstellig sein können, falls mit Einzelelementen gearbeitet wird.

Die maximale Länge eines alphanumerischen Feldes beträgt 256 Bytes. Die maximale Länge eines numerischen Feldes beträgt 8 Bytes gepackt, d.h. 15 Ziffern.

Alle Felder sind standardmäßig alphanumerisch, wenn nichts anderes angegeben wird. Felder werden als numerisch gekennzeichnet, indem angegeben wird, dass sie 0 oder mehr Dezimalstellen besitzen.

Soll eine physische Beziehung zwischen zusammengehörigen Feldern hergestellt werden, so sollten diese in den E- oder D-Bestimmungen definiert werden. In diesen Bestimmungen werden auch Feldgruppen und Overlay-Felder definiert.

Durch Datenstrukturen können ebenfalls Felder und Feldgruppen in Gruppen oder Sätzen zusammengefasst werden.

Wird ein Feld in einer I-Bestimmung spezifiziert, ohne dass es zuvor in einer E- oder D-Bestimmung definiert wurde, so gilt es als implizit definiert. CPG errechnet die entsprechende Länge aufgrund der angegebenen ersten und letzten Stelle und ordnet Speicherplatz dafür in der TWA zu. Ist eine bestimmte Anzahl Dezimalstellen für das Feld spezifiziert (d.h. 0 oder mehr), so behandelt CPG es als gepacktes Feld.

Werden die Eingabedaten für ein Feld als binär oder im ungepackten Format gekennzeichnet, so wandelt CPG sie automatisch in das gepackte Format um, bevor es sie in dem entsprechenden Feld in der TWA abspeichert.

Bei Datenstrukturen sind numerische Felder nur in gepacktem Format zugelassen.

Ein Feld wird in den Rechenbestimmungen implizit definiert, wenn sein Name als Ergebnisfeld einer Rechenoperation angegeben ist und in den Spalten für die Feldlänge eine Eintragung vorgenommen wird.

Ist im CPG-Programm zuvor ein Feld mit demselben Namen explizit oder implizit mit anderer Feldlänge definiert worden, so wird ein Syntaxfehler erkannt und während der CPG-Übersetzungsphase kommentiert. Sollte das Feld numerisch sein, so muss für die Anzahl Dezimalstellen ein Wert von 0 oder größer eingetragen werden. Sind die für das Feld definierten Attribute nicht kompatibel mit der Operation, bei der es benutzt wird, so wird ebenfalls ein Fehler angezeigt.

Folgende Hinweise zu Felddefinitionen müssen beachtet werden:

a) Alle Felder müssen irgendwo in dem Programm, in dem sie benutzt werden, explizit oder implizit definiert werden. (Mit Ausnahme der CPG-internen Felder).

b) Ein Feld kann nicht in einer Weise benutzt werden, die seinen definierten Attributen widerspricht.

Feldgruppen 2120

Feldgruppen sind Gruppen von Feldern gleicher Länge und gleichen Typs, die beispielsweise über eine DO-Schleife indiziert verarbeitet werden sollen.

Definition

Die Definition einer Feldgruppe erfolgt ausschließlich in der erweiterten Dateizuordnung (siehe 4300) oder in der Datenbeschreibung. Alphanumerische Feldgruppen werden mit Blanks, numerische mit Nullen vorformatisiert.

In einem CPG-Programm sind maximal 100 Feldgruppen unterstützt.

Eingabe

Feldgruppen werden in den Ausgabebestimmungen wie Einzelfelder behandelt. Beim Ausgeben auf Bildschirm-Dateien wird die letzte Stelle des ersten Feldgruppen-Elements angegeben. Die folgenden Elemente werden automatisch in den folgenden Bildschirm-Zeilen angezeigt. Für Plattendateien wird die letzte Position des letzten Feldes der Feldgruppe

angegeben. Die Feldgruppe wird dann immer komplett ausgegeben.

Für Plattendateien kann wahlweise die letzte Position des ersten oder des letzten Feldes angegeben werden. Die Feldgruppe wird dann in beiden Fällen komplett eingelesen.

Es können ganze Feldgruppen oder Einzelfelder mit festem Index eingelesen werden. Einzelfelder mit variablem Index können nicht eingelesen werden.

Werden ganze Feldgruppen mit einem Aufbereitungsschlüssel ausgegeben, so sind die Einzelfelder zu der erforderlichen Stellenzahl für die Aufbereitung noch zusätzlich durch 2 Blanks getrennt.

Ausgabe

Bei Ausgabe einer Feldgruppe auf einen Bildschirm wird nur die letzte Position des ersten Feldes angegeben. Alle anderen Felder werden in den gleichen Positionen der Folgezeile ausgegeben.

Es können ganze Feldgruppen oder Einzelfelder mit festem Index ausgegeben werden. Einzelfelder mit variablem Index können nicht ausgegeben werden.

Verarbeitung

Folgende Operationen der Rechenbestimmungen sind indizierbar:

alle numerischen Operationen

CAB	Compare and Branch
CAS	Compare and Branch Subroutine
CBS	Compare and Branch Subroutine
COMP	Vergleichen
CONVT	Konvertieren eines alphanumerischen Feldes
DELC	Zeichen entfernen
DO	Schleifen
DOUxx	Schleifen
DOWxx	Schleifen
EDIT	Feld aufbereiten
ELIM	Zeichen löschen
FILL	Alphafeld füllen
IF	Wenn - Abfrage
MOVE	Rechtsbündig übertragen
MOVEA	Feldgruppe übertragen
MOVEL	Linksbündig übertragen
MOVEN	Numerisch übertragen
REPLC	Blank durch Zeichen ersetzen
SCAN	Alphafeld nach einer Zeichenfolge durchsuchen
SELCT	Feld auswählen

D.h. in einem gültigen Operanden einer dieser Operationen kann anstatt eines Feldnamens der Name einer Feldgruppe angegeben werden.

Dabei gelten folgende Regeln:

1. Feldgruppe mit festem Index

Wurde eine Feldgruppe mit dem Namen FGR definiert, so bewirkt eine Operation:

```
C                     Z-ADDX        FGR,7
```

dass der Wert 'X' in das siebte Feld der Feldgruppe FGR addiert wird. Dabei ist zu beachten, dass die Gesamtlänge der Eintragung nicht größer als sechs Stellen sein darf. Der Name der Feldgruppe darf also in diesem Falle maximal vier Stellen groß sein.

Wird die Feldgruppe in Faktor 1 oder Faktor 2 eingetragen, so stehen maximal 10 Stellen für Feldgruppenname plus Index zur Verfügung.

Beispiele:

```
C           FGRNAM,12 ADD  WERT      FGR,IN
C           WERT      ADD  FGRNAM,12 FGR,IN
C           WERT      ADD  SUM       FGR,12
```

2. Feldgruppe mit variablem Index

Wurde eine Feldgruppe mit dem Namen FGR definiert, so bewirkt eine Operation:

```
C                     Z-ADDX         FGR,N
```

dass der Wert X in das n-te Feld der Feldgruppe FGR addiert wird, wobei das Feld mit dem Namen 'N' eine ganze Zahl zwischen 1 und der Anzahl der für die Feldgruppe definierten Felder enthalten muss. Jeder andere Wert führt zum abnormalen Abbruch des Programms. Die Gesamtlänge der Eintragung darf in Faktor 1 und 2 nicht größer zehn Stellen und im Ergebnisfeld nicht größer als sechs Stellen sein.

```
C           FGRNAM,INDADD  WERT      FGR,IN
C           FGR,INDEX ADD  WERT      FGR,IN
C           WERT      ADD  FGRNAM,INDFGR,IN
C           WERT      ADD  FGR,INDEX FGR,IN
```

3. Feldgruppe ohne Index

Wurde eine Feldgruppe mit dem Namen FGR definiert, so bewirkt die Operation:

```
C                     Z-ADD0         FGR
```

dass alle Elemente dieser Feldgruppe auf Null gelöscht werden. Wurde eine zweite Feldgruppe mit dem Namen FG2 definiert, so bewirkt die Operation:

```
C                     Z-ADDFGR       FG2
```

dass alle Elemente der Feldgruppe FGR in die entsprechenden

Elemente der Feldgruppe FG2 addiert werden, wenn die Anzahl der Elemente für beide Feldgruppen gleich groß ist. Ist eine der beiden Feldgruppen kleiner als die andere, so werden nur so viele Elemente addiert, wie für die kleinste Feldgruppe definiert wurden.

CPG-interne Felder 2130

CPG stellt über verschiedene Felder die Verbindung zu den entsprechenden Bereichen des TP-Steuerprogramms sicher. Vor Verwendung dieser Felder im Anwendungsprogramm sollte der Programmierer sich jedoch vergewissern, ob das von ihm verwendete TP-Steuerprogramm diese Bereiche zur Verfügung stellt. Ist dies nicht der Fall, so bietet die CPG-Methodenbank in den meisten Fällen die Möglichkeit, diese Bereiche selbst anzulegen und damit den Servicegrad des TP-Steuerprogramms zu erweitern.

CPG-interne Felder sind bereits vom Compiler vorgegeben und dürfen daHer vom Programmierer nicht definiert werden. Folgende Felder stehen zur Verfügung:

CPGATR Variables Attribut.

Definition bei Bildschirmausgaben mit der Eintragung 'V' in Spalte 38. In diesem einstelligen Feld wird das gewünschte Attribut eingetragen.

CPGBZL Bezugszahlenleiste

Dieses Feld ist ein 100-stelliges alphanumerisches Feld und enthält die Bezugszahlen 00 bis 99.

CPGCOM Common Area.

Dieser Bereich kann über EDIT- und SELCT-Operationen bei Programmverbindungen mit Command Level als Zwischenspeicher verwendet werden. Zur Zeit beträgt die maximale Länge 4080 Bytes.

CPGCSA Common System Area.

Dieser Bereich kann über EDIT- und SELCT-Operationen verändert oder abgefragt werden. Die in der CPGCSA stehenden Daten stehen unabhängig vom Programm oder Terminal allen Systembenutzern zur Verfügung. Die Länge des Feldes entspricht der Länge der im TP-Steuerprogramm generierten Common Work Area. (Anhang der Common System Area). Die maximale Länge beträgt 3584 Stellen.

CPGCUR Variable Cursor Position.

Definition bei Bildschirmausgaben mit Eintragung 'V' in Spalte 39. In diesem vierstelligen Feld wird die gewünschte Cursorposition eingetragen.

CPGCxx Dateizustand siehe Abschnitt 2260.

CPGDAT 0TTMMJJJJC (num. 8,0 gepackt)

CPGDID Variable Destination-Id.

Nur bei Druckern mit Eintragung 'V' in Spalte 39. In diesem vierstelligen Feld wird der variable Name des Druckers eingetragen.

CPGDLV DL/I variable Operanden.

Sollen Serviceprogramme auf verschiedene DL1-Segmente zugreifen, so können im Feld CPGDLV (Länge 40 alpha) alle Parameter zu den Operationen QSSA, USSA und DLI variabel gesetzt werden. Siehe Beispiel Seite 2335.

CPGDRC DL/I Return-Code.

Dieser Bereich enthält den DL/I Return-Code nach einem DL/I-Call. Siehe Abschnitt 2330.

CPGEDS Ende der Datenübergabe.

CPGEOJ Das interne Feld CPGEOJ (4,0 Stellen) erlaubt es, einen variablen Returncode (größer 0) an das VSE oder z/OS bei Programmende zu übergeben. Das Feld CPGEOJ kann z.B. mit Z-ADD oder mit MOVE gefüllt werden.

CPGFIS Verschiebung bei Dateieingaben

Nur bei Dateien mit Eintragung 'V' in Spalte 43 der I-Dateikarte. In diesem fünfstelligen numerischen Feld wird der Verschiebungsfaktor eingetragen.

CPGFRC File Return-Code nach Datei-Operationen.

Bei der Programmierung von Datei-Operationen kann auf Schalter ganz verzichtet werden. Nach den Operationen enthält das zwei Byte lange Alphafeld CPGFRC die folgenden Return-Codes, die anstelle von Schaltern abgefragt werden können.

DK	Duplicate Key nach READ auf eine AIX-Datei
DR	Duplicate Record nach dem Hinzufügen
EF	End of File nach READ, READ-BACK und SETLL auf VSAM,nach READ auf Temporary Storage, Transient Data, sequentielle Dateien, Reader und Bänder.
EF	FIND nicht gefunden.
NC	Not closed nach CLOSE
NF	Not found nach CHAIN oder CHECK, OPEN, CLOSE
NO	Not open nach OPEN oder CHECK
OD	Open Disabled nach CHECK
Blank	wenn keine Besonderheit festgestellt wurde.

Der Schalter NI/NO = no Input/Output kann nicht über CPGFRC abgefragt werden.

CPGIFC Interface Communication Area.

Die Interface-Communication-Area ist ein 32-stelliges Feld zur Aufnahme von Informationen für das Steuerprogramm-Interface der Methodenbank, das über EDIT- und SELCT-Operationen verändert oder abgefragt werden kann. Mit Hilfe der Operation 'IFC' (Interface-Call) kann der Programmierer dadurch Operationen im teuerprogramm-Interface selbst modifizieren.

Einzelheiten siehe auch (5400).

CPGKxx Datei-Schlüsselfeld

Das CPGKxx-Feld wird pro Platten-Datei in der jeweiligen Schlüssellänge angelegt. 'xx' gibt die Nummer der F-Karte an. Während der Dateiverarbeitung wird im CPGK-Feld der zuletzt bearbeitete Schlüssel gespeichert. Bei VSAM-ESDS Dateien wird die RBA und bei VSAM-RRDS Dateien die relative Satznummer gespeichert.

CPGMCI Feldgruppenindex für Cursorposition (im QSF)

Dieses dreistellige numerische Feld enthält den Index der Feldgruppe, die im Feld CPGMCU beschrieben wurde.

CPGMCU Cursor-Position (im QSF)

Dieses 6-stellige Alphafeld enthält den Namen des Feldes, in das der Cursor gesetzt werden soll. Ist dieses Feld leer, wird der Cursor auf das in QSF definierte Feld gesetzt.

CPGMFI Feldgruppenindex bei der Bildschirmeingabe (QSF).

Dieses dreistellige numerische Feld enthält den Index eines Feldgruppenelementes, das bei der Bildschirmeingabe per Lichtstift oder Cursorpositionierung ausgewählt wurde. Steht der Cursor in einem Feldgruppenelement mit festem Index, z.B. FG,3, so steht der Index 3 ebenfalls in CPGMFI zur Verfügung.

CPGMFN 1. Feldname bei der Bildschirmeingabe (QSF).

Nach der Bildschirmeingabe steht in diesem 6-stelligen Alphafeld der Name des Feldes, das per Lichtstift oder Cursorpositionierung ausgewählt wurde.
UDATE bzw. UTIME werden nicht übergeben.

Steht der Cursor auf einer Konstanten mit einem Eintrag bei alternativer Eingabe, so wird der Name des alternativen Eingabefeldes in CPGMFN gesetzt.

2. Begrenzen der MAP-Ausgabe auf bestimmte Zeilen

Dazu wird '$QV' in dieses Feld gesetzt und CPGMLC (wie unten beschrieben) gefüllt.

CPGMLC 1. Feldposition bei Lichtstiftauswahl/Cursorposition bei der Bildschirmeingabe. (QSF)

Dieses vierstellige Alphafeld enthält in den beiden ersten Stellen die Zeile und in den Stellen 3-4 die Spalte des ausgewählten Feldes.

2. Von-Zeile/Bis-Zeile bei Ausgabe von Teil-Maps.

Um die Mapausgabe auf bestimmte Zeilen zu begrenzen, bringt man vor der Ausgabe Von-Zeile und Bis-Zeile in das Feld. (CPGMLC='0205': Ausgabe Zeile 2 bis 5). Das Feld CPGMFN muss gleichzeitig mit '$QV' gefüllt sein.

CPGMPF Dieses zweistellige Feld wird bei einer Map-Programm-funktion, wenn eine Funktionstaste betätigt wurde, mit dem entsprechenden Tastenkürzel gefüllt.

CPGMRC Dieses zweistellige alphanumerische Feld prüft Felder auf alphabetischen, numerischen und alphanumerischen Inhalt. Die Prüfergebnisse sind ' ' für korrekten Inhalt, 'IC' für 'invalid character' und 'NI' für 'no input'.

CPGNLS National Language Support.

CPGNLS enthält ein D bei deutschen Installationen oder ein E bei englischen (laut Kundenkonfiguration).

CPGPGG CPG-intern

CPGPGI Enthält Informationen zum aktiven Programm:

Stelle 1 - 4 aktuelles CPG-Release
Stelle 5 - 10 Datum der letzten Umwandlung (TTMMJJ)
Stelle 11 - 14 Uhrzeit der letzten Umwandlung
Stelle 15 - 19 Kundennummer
Stelle 20 - 28 Kurzname

CPGPGM Phasenname aus der H-Karte, achtstellig alphanumerisch.

CPGPIW CPG-intern

CPGQxx Satznummer bei Temporary Storage Queues

Siehe Abschnitt 2195.

CPGRxx Internes Dateirettfeld

Das CPGRxx-Feld wird pro Platten-Datei mit DS F (4 Byte) angelegt. Das erste Byte beinhaltet den Status der Datei. (Siehe Kapitel 2260, letzter Abschnitt). Bytes 2 - 4 beinhalten die Adresse der File Work Area der Datei.

CPGSIN Systeminformationen siehe Operation COMRG.

CPGSMN Segment-Name und Key-Feedback-Area.

Dieser Bereich enthält die Key-Feedback-Area nach einem DL/I-Call. Siehe Abschnitt 2330.

CPGTCA Task Control Area

Dieser Bereich adressiert die Task Control User Area. In einem Standard CPG-Programm beginnen ab Position 117 die Userfelder.

CPGTCT Terminal Control Table.

Dieser Bereich kann über EDIT- und SELCT-Operationen verändert oder abgefragt werden. Die in der CPGTCT stehenden Daten stehen unabhängig vom verwendeten Programm dem Benutzer eines bestimmten Terminals für die gesamte Laufzeit zur Verfügung. Die Länge des Feldes entspricht

der bei der Generierung der Terminal Control Table definierten 'User-Area-Length'. Die maximale Länge beträgt 255 Stellen, von denen CPG intern die ersten zehn Stellen benutzt. Im CPG-Programm sind also maximal 245 Stellen benutzbar.

CPGTDI Variable Destination bei Transient Data.

Nur bei Transdt mit Eintragung 'V' in Spalte 39 der F-Karte. In diesem 4-stelligen Feld wird der variable Name der Destination bei Transient-Data-Verarbeitung eingetragen.

CPGTID Terminal Identification.

In dieses vierstellige Feld wird vom CPG der Name des Terminals aus der Terminal-Control-Table für Abfragen im Programm zur Verfügung gestellt. Bei Batchausführung enthält das Feld den Wert Blank. Bei Non-Terminal-Online-Verarbeitung hat das Feld den Wert X'00000000'.

CPGTIO Terminal-I/O-Area.

Dieser Bereich kann über eine SELCT-Operation unmittelbar nach Aufruf eines Programms abgefragt werden. Er dient dazu, Daten zusammen mit der Transaction-Identification einzulesen. Die Trans-Id steht dabei in Position 1 bis 4 der CPGTIO oder von 4 bis 7, je nachdem ob vom leeren Bildschirm eine Trans-Id eingegeben wurde oder aus einem vorformatierten Feld.

CPGTIM Uhrzeit numerisch.

Dieses Feld enthält die Uhrzeit in numerischer (4 Bytes gepackter) Form in einem 6-stelligen Feld (OHHMMSSC) in Stunden, Minuten und Sekunden. Dieses Feld kann zum Ausrechnen von Zeitintervallen in den Rechenbestimmungen benutzt werden. Das Feld wird beim Programmaufruf und durch die Operation TIME aktualisiert. Da das Feld in der Methodenbank liegt, kann eine Aktualisierung auch von außen erfolgen. Falls diese nicht gewünscht wird, so kann bei der TIME-Operation mit einer Eintragung im Ergebnisfeld die Uhrzeit gerettet werden. Eine Verarbeitung mit dem Aufbereitungsschlüssel 'Y' ist möglich.

CPGTIS Terminal-Id simuliert

CPG5-Anwendungen laufen nicht an einem Terminal ab. Zur eindeutigen Identifizierung der Browser-Session stellt CPG diese Id als 8-stelliges alphanumerisches Feld zur Verfügung.

CPGTSN Variabler Dateiname bei Temporary Storage.

Nur bei Storage mit Eintragung 'V' in Spalte 39 der F-Karte. In diesem 8-stelligen Feld wird der variable Dateiname bei Temporary-Storage-Verarbeitung eingetragen.

CPGTWA für HL1-Bausteine (siehe Seite 2661).

CPGVRL Länge eines Satzes einer VSAM-Datei mit variabler Satzlänge (HL1-Batch, sowie Bänder und sequentielle Dateien). (5,0 Stellen).

Nach CHAIN, READ, READB stellt CPGVRL die Satzlänge des gelesenen Satzes zur Verfügung. Nur bei VSAM-Dateien mit variabler Satzlänge. Ein 'V' in Spalte 38 der O-Karte in Verbindung mit ADD/ALG bewirkt, dass der Wert aus CPGVRL die Satzlänge der Datensätze bestimmt.

Datum und Uhrzeit 2136

CPG stellt für Datum und Uhrzeit je ein 8-stelliges alphanumerisches Feld zur Verfügung. Diese Felder können vom Programmierer in den Rechen- oder Ausgabebestimmungen unter den Namen 'UDATE' und 'UTIME' beliebig benutzt werden. Außerdem steht die Uhrzeit nach der Operation TIME noch in numerischer Form im Feld CPGTIM zur Verfügung.

CPGDAI enthält das Tagesdatum numerisch in der Form '0JJJJMMTTC'.

Im Hinblick auf den Jahrtausendwechsel ist diese Datumsform sinnvoll, da sie durch die vollständige Angabe der Jahreszahl vergleichbar und durch den Aufbau sortierbar ist.

CPGDAT enthält das Tagesdatum in der Form '0TTMMJJJJC', also mit vierstelliger Jahreszahl. Als achtstelliges numerisches Feld kann CPGDAT mit Edit-Code 'Y' aufbereitet werden.

CPGTIM enthält die Uhrzeit gepackt in der Form '0HHMMSSC'

UDATE enthält das Tagesdatum in der Form: 'TT.MM.JJ'.

UDATEC enthält das Jahrhundert in der Form: 'JJ' (alpha)

UDATEI enthält das Tagesdatum in der Form: 'JJJJMMTT' (alpha 8)

UDAY enthält den aktuellen Tag : '0TTC' (numerisch)

UMONTH enthält den aktuellen Monat: '0MMC' (numerisch)

UTIME enthält die Uhrzeit in der Form 'HH.MMUHR'.

UYEAR enthält das aktuelle Jahr : '0JJC' (numerisch)

Speicherungsarten 2140

Alphanumerische Felder 2141

Alphanumerische Felder können alle druckbaren und nicht druckbaren Zeichen aufnehmen. Jedes Zeichen belegt ein Byte im Speicher. Alphanumerische Felder werden vor Beginn der Verarbeitung auf Blank initialisiert.

Numerische Felder 2142

Numerische Felder können nur Ziffern und gegebenenfalls ein Minuszeichen enthalten. Numerische Felder werden immer in gepackter Form gespeichert, das heisst in einem Byte werden jeweils 2 Ziffern untergebracht. Ein Halbbyte wird reserviert für das Vorzeichen. Numerische Felder werden vor Beginn der Verarbeitung auf Null initialisiert.

Datenfelder, mit denen Rechenoperationen durchgeführt werden sollen, müssen immer numerisch definiert sein.

Zur Vermeidung unnötiger Konvertierungen wird empfohlen, numerische Felder bei der Zwischenspeicherung auf Platteneinheiten ebenfalls in gepacktem Format auszugeben.

Natürlich ist auch die Ausgabe in ungepacktem Format möglich. In diesem Fall belegt jede Ziffer ein Byte. Das Vorzeichen wird zusammen mit der letzten Ziffer im rechtesten Byte des Feldes untergebracht. Dies kann dazu führen, dass diese Ziffer bei der Ausgabe in einen Buchstaben umgewandelt wird.

CPG gibt allen ungepackten numerischen Feldern, wenn sie positiv sind, automatisch ein Vorzeichen mit, das die letzte Ziffer wieder lesbar macht.

Achtung: Wenn bei der ungepackten Ausgabe numerische Felder teilweise durch andere Felder überlagert werden, muss diese Automatik durch Eintragung von "S" in Spalte 40 der H-Karte ausgeschaltet werden.

Binäre Felder 2143

Numerische Felder können binär ausgegeben und wieder eingelesen werden. Die binäre Ausgabe erfolgt durch Eintragung eines 'B' in Spalte 44 der Ausgabebestimmungen. Die Eingabefelder werden als binär interpretiert, wenn Spalte 43 der Eingabebestimmungen ein 'B' enthält.

Werden Felder binär eingelesen, so werden 2-Byte-Eingaben mit 4 Stellen numerisch und 4-Byte-Eingaben mit 9 Stellen numerisch definiert. Das ist RPG-kompatibel.

Für numerische Felder bis zu 4 Stellen werden bei der Ausgabe 2 Byte große Felder angelegt und für Felder bis zu 9 Stellen 4 Byte große Binärfelder.

Bei Bildschirm- und Drucker-Ein- und -Ausgaben dürfen keine binären Felder benutzt werden.

Logisch gepackte Felder 2144

Neben der Möglichkeit, numerische Felder in gepacktem Format zu speichern, bietet CPG eine weitere Einsparung von Plattenspeicherkapazität durch die Verarbeitung logisch gepackter Felder an.

Ein logisch gepacktes Feld ist ein Feld, dessen numerischer Inhalt gepackt und ohne Vorzeichen gespeichert wird.

Beispiel: Die Speicherung der Zahl 4711 erfordert:

Ungepackt	4 Bytes	F4 F7 F1 F1
Gepackt	3 Bytes	04 71 1C
Logisch gepackt	2 Bytes	47 11

Datenfelder die logisch gepackt werden sollen, dürfen nur positive nu-

merische Werte enthalten. Die logisch gepackte Form gilt nur für die Speicherung in externen Speichern. Zur Verarbeitung im Anwendungsprogramm werden die Daten wieder in das gepackte Format zurück übertragen.

Numerische Felder werden logisch gepackt ausgegeben, wenn Spalte 44 der zugehörigen Ausgabebestimmung ein 'L' enthält.

Numerische Felder werden logisch gepackt eingelesen, wenn Spalte 43 der zugehörigen Eingabebestimmung ein 'L' enthält.

Ungepackte numerische Felder 2145

Numerische Felder werden im Hauptspeicher (TCA) immer gepackt. Durch das Einlesen vom Bildschirm erhält ein numerischer Wert deshalb intern im letzten Halbbyte den Buchstaben 'F' für die Zone.

Wird ein numerischer Wert aus dem Hauptspeicher auf Platte ungepackt ausgegeben, so wird die Zone grundsätzlich auf 'F' gesetzt. Der Wert 123 steht dann beispielsweise auf der Platte als F1 F2 F3.

Beim Einlesen von der Platte wird die Zone nicht geändert.

Diese Regeln für das Zonenhalbbyte des gepackten Feldes können durch die Eintragung 'S' in Spalte 40 der H-Karte beeinflusst werden.

Der Eintrag bewirkt, dass beim Einlesen sowohl vom Bildschirm als auch von der Platte grundsätzlich die Zone 'C' angenommen wird. Bei der Ausgabe auf Platte wird die Zone nicht verändert.

Bei ungepackter Ausgabe steht in diesem Fall die Zahl 123 auf Platte als F1 F2 C3, also 12C.

vorher	C	F	C	F
nach Bildschirm-Eingabe	F	F	C	C
nach Platten-Eingabe	C	F	C	C
nach Platten-Ausgabe	F	F	C	F
H-Karte	Spalte 40 = ' '		Spalte 40 = 'S'	

Bildschirm-Eingabefelder 2150

Im Gegensatz zu Eingabefeldern anderer Dateien wird ein Bildschirmeingabefeld mit einer READ-Operation nur dann eingelesen, wenn das Feld vor der Operation am Bildschirm verändert wurde. Hardwareseitig wird dies dem Steuerprogramm durch ein 'Modified Data Tag' mitgeteilt. Das 'Modified Data Tag' kann jedoch auch vom Programmierer durch den Aufbereitungsschlüssel gesetzt werden. In diesen Fällen wird das Eingabefeld auch gelesen, wenn es vorher nicht verändert wurde.

. Ein einzulesendes Feld wird identifiziert durch die mit der Eingabenachricht übertragene Adresse des 1.Bytes dieses Feldes, d.h. es können nur solche Felder eingelesen werden, die vorher bereits einmal auf dem Bildschirm ausgegeben (vorformatisiert) wurden.

Der Programmierer muss dabei beachten, dass die erste Stelle des vorher ausgegebenen Feldes mit der Definition des Feldes in der Eingabebestimmung übereinstimmt, andernfalls wird das Feld nicht eingelesen.

Die sicherste Methode zur Erfüllung dieser Bedingung ist die Vorformatisierung eines Feldes durch eine Blankkonstante in der Länge des einzulesenden Feldes.

Bei numerischen Feldern kann jedoch je nach Anwendung die Forderung bestehen, das Feld zur Aufnahme von Aufbereitungszeichen (Komma, Punkt Vorzeichen) größer als die definierte Stellenzahl vorzuformatisieren. Der Programmierer muss in diesen Fällen beachten, dass das erste Byte des Feldes zwischen Ein- und Ausgabebestimmungen übereinstimmt.

Numerische Felder können mit gerader oder ungerader Stellenzahl definiert werden. Da numerische Felder intern in gepacktem Format verarbeitet werden, sollten diese Felder möglichst mit ungerader Stellenzahl definiert werden. Besteht dennoch die Notwendigkeit, ein Feld mit gerader Stellenzahl zu definieren, so muss der Programmierer beachten, dass bei Ausgabe mit Schablone das Feld automatisch um eine Stelle vergrößert wird; das heisst, die Schablone muss immer eine ungerade Anzahl von Stellen aufnehmen können.

Bei numerischen Feldern wird die Dezimalstellenanpassung vom CPG wie folgt durchgeführt: Dezimalstellen werden ab dem ersten eingegebenen Komma oder nach der letzten gültigen Ziffer dem definierten Eingabefeld angepasst.

Werden keine Dezimalstellen eingegeben, so werden nötigenfalls fehlende Dezimalstellen als Nullwerte eingefügt. (Siehe Beispiel 3 und 5). Zuviel eingegebene Dezimalstellen werden ignoriert, wie im Beispiel 2 zu sehen ist. Werden mehr Ziffern eingegeben als ein numerisches Feld aufnehmen kann, wird ab der angenommenen Kommastelle das Feld angepasst. Siehe hierzu auch das Beispiel 3 und 6.

Wird ein Feld vom Bildschirm gelesen, das vorher mit einer Schablone dorthin ausgegeben wurde, so ist unabhängig von der Felddefinition das formatisierte Feld um ein Byte größer als die Länge der Schablone.

Beispiel :

FELDA wird numerisch definiert mit 4 Stellen, davon 2 Dezimalstellen.
FELDB wird numerisch definiert mit 5 Stellen, davon 2 Dezimalstellen.

```
             I                              110 1132FELDA
             I                              209 2132FELDB
```

Die Felder FELDA und FELDB sollen vom Bildschirm gelesen werden; dazu muss der Bildschirm vorher 'initialisiert', d.h. zur Aufnahme der Daten vorformatisiert werden. Dies erfolgt durch die vorherige Ausgabe eines entsprechenden Feldes (Maske) auf den Bildschirm.

Die Maske muss in jedem Fall einen Aufbereitungsschlüssel mit der Feldeigenschaft 'ungeschützt' erhalten.

Das erste Byte der Maske muss mit dem 1.Byte der Feldeingabe übereinstimmen. Die Maske muss alle möglichen Eingabewerte aufnehmen können. Da CPG eine automatische Dezimalstellen-Anpassung vornimmt, muss die Maske für FELDA mindestens 4 Stellen plus 2 Dezimalstellen und für FELDB mindestens 5 Stellen plus 2 Dezimalstellen groß sein.

Feld		Pos.	Maske	Eingabe	Feldinhalt	Num.
	A	115	' '	'12,34 '	12,34	1
				'1,2345'	1,23	2
				'123 '	23,00	3
	A	215	' '	'0,1234 '	0,12	4
				'123 '	123,00	5
				'1234567'	567,00	6
FELDA	A	116	' , '	' 4,25'	4,25	7
FELDA	A	118	' 0 , CR'	' 12,34CR'	12,34-	8
FELDB	A	215	' , '	' 4,25'	4,25	9
FELDB	A	217	' 0 , CR'	' 12,34CR'	12,34-	10

Die Maske beginnt bei FELDA immer auf Position 110
bei FELDB immer auf Position 209

Jede andere Position hat zur Folge, dass das Feld vom Bildschirm nicht eingelesen wird.

Eine Überlagerung von Eingabefeldern ist daher beim Bildschirm nur möglich, wenn die Felder in der ersten Position übereinstimmen. Andernfalls muss die Überlagerung durch Definition der Felder in der erweiterten Dateizuordnung oder Datenbeschreibung erfolgen. Siehe auch (4300).

Zwischenspeicherung von Daten 2160

Online-Anwendungen erfordern mitunter eine stufenweise Verarbeitung der Daten durch verschiedene Programme, die nacheinander ablaufen. Da beim Aufruf eines neuen Programms die von diesem Programm benutzten Datenfelder in der Regel neu formatisiert, d.h. gelöscht werden, erfordert eine solche 'Verkettung' von Programmen eine Zwischenspeicherung der Daten in einem Bereich, der auch nach Beendigung eines Programms erhalten bleibt.

Die Zwischenspeicherung kann auf verschiedene Arten erfolgen:

1. Zwischenspeichern von Daten in der Transaction Work Area.

Die Transaction Work Area (TWA) wird bei jedem Programmaufruf neu formatisiert. Jedoch kann der Programmierer in Spalte 27 bis 30 der H-Karte die Anzahl Bytes der WA beginnend vom Anfang eintragen, die nicht neu formatisiert werden soll.

Bei dieser Form der Zwischenspeicherung sind die Regeln für Programmverbindungen (2280) zu beachten.

2. Zwischenspeichern von Daten in der Terminal-Tabelle.

Mit Hilfe der EDIT-Operation kann der Programmierer Daten in der TCT des Steuerprogramms abstellen, die zu beliebigen Zeitpunkten mit beliebigen Programmen, aber nur vom selben Terminal wieder abgerufen werden können. Der Abruf erfolgt mit Hilfe einer SELCT-Operation. Das Ergebnisfeld heisst in beiden Fällen 'CPGTCT' und braucht im Programm nicht definiert zu werden.

Beispiel:

```
Daten speichern:     C                    EDIT          CPGTCT

                     OCPGTCT  F
                     O                             WERT      20P
```

Das numerische Feld 'WERT' wird in gepacktem Format in der TCT gespeichert. Die letzte Stelle des Feldes steht auf Position 20 des Feldes 'CPGTCT'.

```
Daten abrufen:       ICPGTCT  F
                     I                        P  16  202WERT

                     C                    SELCT         CPGTCT
```

Das in den Positionen 16-20 abgestellte numerische Feld wird in das Feld WERT übertragen. Das Feld war in der TCT gepackt gespeichert. Das Feld 'WERT' wird mit 2 Dezimalstellen definiert.

Die Größe des Feldes 'CPGTCT' wird bei der Generierung der Terminal-Tabelle (TCT) festgelegt, und zwar muss der Parameter, der die User Area Length beschreibt, um 10 Bytes größer sein als die höchste in einem Programm verwendete Ausgabe-Position. Im obigen Beispiel also Position 20 + 10 Bytes ergibt eine TCT User Area Length von mindestens 30 Bytes.

Die TCT-Parameter, die bei der TCT-Generierung zu berücksichtigen sind, sind im Installationsteil dieses Handbuchs (Kapitel 9000) oder im separaten Installationshandbuch beschrieben.

3. Zwischenspeichern von Daten in der CSA.

Mit der oben für die TCT beschriebenen Methode können auch Daten in der Common-System-Area (CSA) zwischengespeichert werden. Diese Daten können dann zu beliebigen Zeitpunkten mit beliebigen Programmen von beliebigen Terminals wieder abgerufen werden. Hierzu gilt das oben für die TCT beschriebene Beispiel, wenn anstelle von CPGTCT der Feldname CPGCSA eingesetzt wird.

Die Größe des Feldes CPGCSA wird bei der Generierung der System-Initialisierungs-Tabelle des jeweiligen TP-Monitors festgelegt. Der CSA-Size-Parameter muss um mindestens 16 Bytes größer sein als die höchste in einem Programm verwendete Ausgabeposition, da CPG intern die ersten 16 Stellen benutzt. (Siehe Installationshandbuch bzw. Kapitel 9000).

4. Zwischenspeichern von Daten auf dem Bildschirm.

Der Bildschirm kann ebenfalls als Zwischenspeicher benutzt werden, wenn die Daten am Ende eines Programms dorthin ausgegeben werden, jedoch kann das Folgeprogramm diese Daten nur dann lesen, wenn bei der Ausgabe ein Aufbereitungsschlüssel mit der Feldeigenschaft 'Modified-Data-Tag-gesetzt' spezifiziert wurde und das Programm nach 'CPGEND' verzweigt.

5. Temporary Storage und Temporary Storage Queues siehe Kapitel 2190.

6. Zwischenspeichern von Daten auf Transient Data.

Sequentiell ausgegebene Daten können auf Transient Data zwischengespeichert werden. CPG verwendet Transient Data in jedem Fall zur Zwischenspeicherung von Drucker-Ausgaben. Der Programmierer kann diesen Service auch für eigene Anwendungen nutzen. Er sollte dabei jedoch die Auswirkungen auf die Performance übersehen können. Die Programmierung ist nicht anders als für Platten-Dateien. Die entsprechende Eintragung in die Spalten 40 - 46 der Dateizuordnung ist 'TRANSDT'.

7. Zwischenspeichern von Daten auf Platten-Dateien.

Bei zu kleinem Hauptspeicher (hohe Paging Rate) sollte in Zweifelsfällen auch die Zwischenspeicherung von Daten auf Platten-Dateien erwogen werden, um gegebenenfalls den Hauptspeicher zu entlasten.

8. Zwischenspeichern in der Common Area (bei Command Level Programmen)

Nur im Command Level steht die Common Area für das Zwischenspeichern von Daten zur Verfügung. Die Common Area ist 4080 Bytes groß. CPG kann über das Feld CPGCOM und die Operationen EDIT und SELCT mit der Common Area kommunizieren.

Die Länge der übergebenen Common Area kann durch SELCT CPGPIW ermittelt werden. Sie steht dort binär gepackt in den Stellen 239-240.

Zwischenspeichern von Daten auf Temporary Storage. 2190

Der Programmierer kann einen Bereich im Hauptspeicher oder in einem angeschlossenen Hilfsspeicher für die Zwischenspeicherung von Daten verwenden, soweit der verwendete TP-Monitor dies zulässt.

CPG ordnet ihnen einen symbolischen Namen (bis zu acht Zeichen) zu, und unter diesem Namen werden sie zwischengespeichert, aufgesucht und freigegeben. Der Name setzt sich wie folgt zusammen:

XXXXYYYY

X = Terminalkennzeichen
Es kann die Verarbeitungsart 'Independent' gewählt werden, d.h. der Name des Zwischenspeicherbereichs soll terminal-unabhängig werden. In diesem Fall ist X = '****'. Erreicht wird diese Verarbeitungsart durch ein 'I' in Spalte 17 der Satzbestimmung in den O-Karten oder ein 'I' in Spalte 32 der F-Karte.

Y = Storagename aus F-Karte

Dazu muss dieser Bereich in einer Dateizuordnungskarte mit einem beliebigen vierstelligen Namen definiert werden. Die Einheit (Spalte 40-46) lautet dabei 'STORAGE'. Die Länge des reservierten Speichers ergibt sich aus der Eintragung im Feld Satzlänge.

Außerdem bietet CPG die Möglichkeit, in Spalte 39 der F-Karte ein 'V' (variabel) einzutragen. Bei der Programmausführung muss im Feld CPGTSN der Storage-Name (max. 8 Stellen) übertragen werden, der verarbeitet werden soll. Dies sollte nach Möglichkeit bei einer None-Terminal Task eingesetzt werden.

Es darf nicht die gleiche Datei (Storage) mit verschiedener Satzlänge verwendet werden. Dies würde zu Storage Violations führen.

Um Schwierigkeiten zu vermeiden, die durch doppelt vergebene Namen entstehen können, sollten Benennungskonventionen eingeführt werden, an die sich die Programmierer halten müssen.

Ein temporärer Nachrichtenbestand sollte zum frühest möglichen Zeitpunkt freigegeben werden, damit für diesen Zweck nicht zu viel Speicher belegt wird.

Für die Ein- und Ausgabebestimmungen gelten dabei die Regeln wie für Platten-Dateien.

Die Daten können eine einzelne Informationseinheit darstellen, oder sie werden einem Nachrichtenbestand hinzugefügt.
Wenn ein Nachrichtenbestand angelegt werden soll, sind die Bedingungen bei Temporary Storage Queuing einzuhalten. Siehe Abschnitt 2195.

Bei der Ausgabe oder schon in der F-Karte kann durch eine Eintragung bestimmt werden, ob der Satz im Hauptspeicher bleiben oder auf eine Plattendatei ausgelagert werden soll und ob die Daten nur vom eigenen oder von allen Terminals gelesen werden können.

Spalte 17 der O-Karte bzw. Spalte 32 der F-Karte bedeutet:

Blank Die Daten können nur vom eigenen Terminal gelesen werden und werden im Hauptspeicher gespeichert.

I Die Daten können von allen Terminals gelesen werden und werden im Hauptspeicher gespeichert.

A Die Daten können nur vom eigenen Terminal gelesen werden und werden im Hilfsspeicher gespeichert.

In den Rechenbestimmungen können temporär gespeicherte Daten mit einer READ-Operation gelesen werden. Dabei kann der Programmierer durch eine Eintragung in Spalte 53 der READ-Operation bestimmen, ob der Bereich freigegeben oder für spätere READ-Operationen wieder verwendet werden soll. Die Eintragung in Spalte 53 bedeutet:

Blank Der Bereich wird nach dem Lesen freigegeben.
S Der Bereich wird nach dem Lesen nicht freigegeben.

Der Bereich wird jedoch in jedem Fall wieder freigegeben, wenn eine erneute Ausgabe unter demselben Namen erfolgt.

Bei Temporary Storage Queuing wirkt der Eintrag Blank in Spalte 53 genauso wie der Eintrag 'S', da das Löschen mit der speziellen Operation PURGE durchgeführt werden muss.

Temporary Storage Queuing 2195

Es gibt fünf Verarbeitungsmöglichkeiten:

1. Eine Queue füllen

2. Einzelne Elemente verändern

3. Sequentiell lesen

4. Direkt lesen

5. Ganze Queue löschen

Bei Temporary Storage Queuing muss in Spalte 16 der Dateizuordnungskarte (F-Karte) ein 'Q' eingetragen werden.

```
FSTOR   UQ   F    100               STORAGE
```

Für jede Queue wird intern ein CPGQXX-Feld fünfstellig numerisch angelegt. XX ist die Nummer der Dateizuordnung.

In Stelle 19 ist immer ein 'F' einzutragen, variable Satzlängen sind nicht unterstützt und werden wie feste Satzlängen behandelt.

Funktion 1 Queue füllen:

```
C               EXCPT                   20

OSTOR    EADD 20
O                            13 'QUEUE FÜLLEN'
```

Dieses Beispielprogramm zeigt, wie eine Queue gefüllt wird.

Es können maximal 32.000 Sätze in einer Queue gespeichert werden.
Der Benutzer ist bei einer Überschreitung verantwortlich, es erfolgt keine Fehlermeldung.
Nach einer Ausgabe ist im Feld CPGQXX die relative Positionsnummer verfügbar.
Wird in der Ausgabe statt EADD nur 'E' in Spalte 15 verwendet, so werden die Sätze auch dann hinzugefügt, wenn das Feld CPGQXX vorher auf Null gelöscht wurde.

Ein ausführliches Beispielprogramm finden Sie auf Seite 8120.

Funktion 2 einzelne Elemente verändern:

```
  ISTOR  KF
  I                                1    100  SATZ

  C               Z-ADD4        LFNR  50
  C      LFNR     READ STOR
  C   EF           GOTO NOTFND
  C                EXCPT                  21
  C      NOTFND   TAG

  OSTOR    E    21
  O                     SATZ   100
  O                              6 'UPDATE'
  O                     LFNR   100 P
```

Dieses Beispielprogramm zeigt, wie einzelne Elemente einer Queue verändert werden. Wird ein ausgewähltes Element nicht aufgefunden, so wird die Bedingung EF gesetzt. Der gesamte Datensatz aus der Eingabe sollte als erste Ausgabebestimmung vor den Update-Feldern mit ausgegeben werden.

In diesem Beispiel wird das vierte Element verändert, bei NOT FOUND wird die Ausgabe nicht durchgeführt.

Hinweis: Im Gegensatz zu der UPDATE-Funktion bei Dateien ist bei der Ausgabe auf Temporary Storage der Ausgabebereich generell auf Blank gelöscht. Soll der ursprüngliche Satz bei der Ausgabe erhalten bleiben, so ist er aus der Eingabe in die Ausgabe zu übertragen. In dem Beispiel werden dann die Stellen 1 - 6 des Satzes durch die Konstante 'UPDATE' und die Stellen 98 - 100 durch die gepackte 'LFNR' überschrieben.

Funktion 3 sequentiell lesen:

```
  C               Z-ADD1        KEY   50
  C      KEY      READ STOR                1. Satz       oder
  c      1        READ STOR                1. Satz
  C      READ     TAG
  C               READ STOR                Folgesatz
  C  EF           GOTO EOF
```

Dieses Beispielprogramm zeigt, wie eine Queue sequentiell gelesen wird.
Mit einer Read-Operation ohne Faktor 1 werden die Elemente der Reihe nach sequentiell gelesen. Um jedoch bei einem bestimmten Satz (z.B. Satz 1) mit dem Lesen zu beginnen, so ist der erste Satz gezielt mit

einem definierten Key zu lesen.

Funktion 4 direkt lesen:

```
  C               Z-ADD5        KEY   50
  C      KEY      READ STOR
```

Dieses Beispiel zeigt, wie aus einer Queue direkt gelesen wird.

Es wird das entsprechende Element, das in Faktor 1 der READ-Operation definiert ist, gelesen. Anders als bei Datei-Operationen muss der Programmierer bei den folgenden Reads dafür sorgen, dass im Key jeweils die gewünschte Satznummer enthalten ist, da hier mit 'KEY READ STOR' immer eine direkte Verarbeitung beschrieben wird. Daher können die Operationen CHAIN, RNDOM, SETLL usw. hier nicht verwendet werden.

```
  C               Z-ADD0        KEY   50
  C      READ     TAG
  C      KEY      ADD  1        KEY
  C      KEY      READ STOR
  C  EF           GOTO EOF
  C               ...
  C               GOTO READ
  C      EOF      TAG
```

Dieses Beispiel zeigt, wie eine gesamte Queue direkt gelesen werden kann.

Funktion 5 Queue löschen:

```
  C               PURGE STOR
```

Dieses Beispiel zeigt, wie eine Queue gelöscht wird.
Mit der Operation PURGE wird die in Faktor 2 definierte Queue gelöscht.
Beim Löschen einer Storage Queue wird der belegte Bereich freigegeben und dem verfügbaren dynamischen Speicherbereich zugeschlagen.
Es gibt keine Möglichkeit, nur ausgewählte Sätze aus einem Nachrichtenbestand freizugeben, daher ist beim Read die Spalte 53 nicht unterterstützt. Alle Sätze bleiben solange erhalten, bis der Bereich mit PURGE gelöscht wird.

Simulation des Queuing 2198

Mit der Eintragung S in der File-Karte bzw. im Data Dictionary erreicht man, dass ein TS-Bereich vom Programmierer nach der Einzelsatzlogik verarbeitet werden kann, dass vom CPG aber eine Queue generiert wird, die aus einem Satz besteht.

Beispiel:

```
    FNAME    US  F     100        STORAGE
```

Diese Funktion wird vor allem bei der Migration nach ESA verwendet, da hier nur noch TS-Queues verwendet werden sollten.

Programme 2200

Programmaufbau 2210

Vergleiche hierzu auch Abschnitt DC-Funktionen, Hauptspeicher-Management.

Register-Zuordnung 2211

Registerzuordnung der Standard-Version des CPG.

bei Programmen der Standard-Version mit 'M' in Spalte 47 der H-Karte werden die Register wie folgt belegt:

Register 0	nicht benutzt
Register 1	CPG-Workreg, Arbeitsregister
Register 2-7	Basisregister. Soweit nicht benutzt frei verfügbar.
Register 6	TWA bei 12K TWA
Register 7	TWA bei 8K TWA
Register 8	CPG-Subreg, Unterprogrammregister
Register 9	CPG-Getreg, Putreg, Ein-/Ausgaberegister
Register 10	CPG-Cioabar, Ein-/Ausgabebereichsregister
Register 11	CPG-Methbar, Methodenbank-Register
Register 12	CPG-TCA-Register und TWA
Register 13	CPG-CSA-Register bzw. bei ESA EIB-Storage
Register 14	CPG-Methodenbank-Verzweigungsregister
Register 15	CPG-Countreg, Zählregister

Bei der Verwendung von CPG-Registern, etwa in Assembler-Unterroutinen, sind die Register vorher vom Programmierer zu retten und nach Beendigung der Routine wieder zu laden.

Für Programme, die mit 'C' in Spalte 47 der H-Karte umgewandelt werden, gilt die unter CICS beschriebene dynamische Registerzuordnung.

Dokumentation 2240

Kommentare 2241

Ein Stern in Spalte 7 einer beliebigen CPG-Karte listet den Inhalt von Spalte 7 bis 74 dieser Karte als Kommentar in der Umwandlungsliste an und übernimmt den Kommentar ebenfalls in das generierte Assemblerprogramm.
In der Umwandlungsliste sind zwei Kommentarformen möglich:

1. Die normale Kommentarform (wie bei RPG). Hierbei enthält Spalte 6 die Kartenart der jeweiligen Bestimmung, z.B. 'C' bei Kommen-

taren in den Rechenbestimmungen.

2. Die gespreizte Kommentarform. Je ein Zeilentransport erfolgt vor und nach dem Drucken, wenn Spalte 6 blank bleibt.

Sprache 2242

Kommentare im generierten Assembler-Programm werden in englischer Sprache ausgegeben. Die Sprache der Kommentare und der Diagnostik im CPG-Programm kann der Programmierer frei wählen, wenn er die Texttabelle (Copy Book 'CPG*CTXE') in die von ihm gewünschte Sprache übersetzt und nach Änderung des letzten Buchstabens mit // OPTION CATAL und PHASE CPGSXE,* (wobei für 'E' oben genannte Buchstabe eingesetzt wird) neu katalogisiert. (* steht für CPG-Release z.B. 2.5 = 4)

Die Übersetzung wird dann immer angezogen, wenn dieser Buchstabe in Spalte 21 der H-Karte eingetragen wird.

Datenbeschreibungskarte 2243

Eine bessere Dokumentation der Datenfelder kann erreicht werden, wenn die Felder in einer Datenbeschreibungskarte definiert und beschrieben werden. Siehe auch (4400).

Blockschaltbild 2244

Bei Eintragung eines 'F' (Flow-Chart) in Spalte 15 der H-Karte wird rechts neben der Umwandlungsliste ein Blockdiagramm des CPG-Programms angelistet, das das Lesen des Programms auch für den Außenstehenden erleichtern soll.

Das Blockdiagramm hat folgenden Aufbau:

Blockdiagramm	PhaseX		von der H-Karte erzeugte Überschrift
Bildschirm			Dateizuordnung L3270
Platte			Dateizuordnung DISK
DATUM ---------	11	16	E- oder D-Karte, Überlagerung TWA
TAG	11	12	
MONAT	13	14	
JAHR	15	16	

```
INPUT--------------PLATTE      Eingabedatei INPUT (Plattendatei)
       X              75       Feld X ist 75 Stellen lang (ALPHA)
       Y               7,2     Feld Y ist 7 Stellen,davon 2 Dez.
       FG      10 *    5       Feldgruppe 10 * 5 Stellen lang

-------------------------      Beginn der Rechenbestimmungen

ANFANG-------I                 Anfang TAG
            I                  Rechenbestimmung ohne Verzweigung
           IFI                 EDIT oder SELCT Operation
01 ---      I                  Schachtelungstiefe bei DO, IF und END
            I
02 ---      I                  Eingerückte   Schachtelungstiefe  bei
 02 ---     I                  ELSE, BREAK und CONT.
02 ---      I
01 ---      I
           /// ALL             EXCPT
            I======UPRO        EXSR SUBR  oder EXPR UPRO
            I
            I------OUT         GOTO OUT oder EXITP OUT
   INPUT   ///                 Lesen Datei INPUT
            I
           < >-----ANFANG      GOTO ANFANG wenn... (z.B. 15 ein)
            I
           /// OUTPUT          UPDATE NEWRC OUTPUT
            I
            -------ANFANG      GOTO Anfang ohne Bedingung

-------------------------      Ende der Rechenbestimmungen

OUTPUT-------------PLATTE      Ausgabedatei OUTPUT (Plattendatei)
   X         1       75       Feld X beginnt auf Stelle 1, ist 75 lg
   Y        76        7,2     Feld Y beginnt auf 76 ist num 7St/2Dez
   FG        1 10 *   5       Feldgruppe beginnt auf 1 ist 10*5 lang
```

Blockdiagramm bei Überlagerungen mit ORG:

Die flexibelsten Möglichkeiten zur Redefinition und Überlagerung in der Data Division bietet der Befehl ORG. In der Data Division gibt das Blockdiagramm pro Feld die Von- und die Bis-Position an. Beim ORG-Befehl wird stattdessen angezeigt: 1. die nächste Von-Position, wenn kein ORG codiert worden wäre und 2. die neue Von-Position als Ergebnis des ORG-Befehls.

```
Beispiel:  Source Code                    BLOCKDIAGRAMM       PHASE

D        F1           10                        F1          1   10
D        F2         5 10                        F2         11   60
D        F3            5                        F3         61   65
D        F4            5                        F4         66   70
D        F3    ORG                       ORG    F3         71   61
D        F5            5                        F5         61   65
D        F5    ORG                       ORG    F5         66   61
D        F6            5                        F6         61   65
D              ORG                       ORG                66   71
D        F7           10                        F7         71   80
```

Liststeuerung 2245

Der Programmierer hat die Möglichkeit, die CPG-Umwandlungsliste seinen Wünschen anzupassen. Dazu dienen zunächst die Eintragungen in den Spalten 11, 15, 16, 21 und 41 der H-Karte, die unter der Formularbeschreibung für diese Karte beschrieben sind.

Die Eintragung '/EJECT' ab Spalte 7 einer beliebigen CPG-Karte bewirkt einen Formularvorschub auf die nächste Seite der Umwandlungsliste.

Der Ausdruck einer CPG-Umwandlungsliste kann mit /NOLIST (ab Spalte 7 einer beliebigen CPG-Karte) unterbrochen und mit /LIST wieder gestartet werden. Wird ein CPG-Fehler erkannt, so wird die NOLIST-Funktion aufgehoben.

Programmierer-Checkliste 2246

Im Anschluss an die Umwandlung wird eine Programmierer-Checkliste gedruckt, die den Programmierer auf bestimmte Voraussetzungen des TP-Steuer-Programms hinweisen und Auskunft über die wichtigsten Programmdaten geben soll.

Die Checkliste hat folgenden Aufbau:

```
TITEL                                   KD-NR KURZNAME       01.04.83
_____________________________________________________________________

        PROGRAMMIERER CHECKLISTE

               ENTHÄLT DIE CICS PCT EINE EINTRAGUNG MIT DEM PARAMETER
                                    PROGRAM=CPGTST UND
                                    TWASIZE=0166 ODER GRÖßER

               ENTHÄLT DIE CICS PPT EINE EINTRAGUNG MIT DEM PARAMETER
                                    PROGRAM=CPGTST

               ENTHÄLT DIE CICS FCT EINE EINTRAGUNG MIT DEM
                                    DATEINAMEN PLATTE


       PROGRAMMDATEN

               PROGRAMMGRÖßE = 3.500 BYTES (CIRCA)

               TWA GRÖßE     =   166 BYTES

               TIOA GRÖßE    =     4 BYTES


               DEFINIERTE SCHALTER

               01 17 25 C1 T3 P2 DE
```

EXTERNE PROGRAMMVERBINDUNGEN

```
EXIT:  'TRID'
EXIT:  PHASEX
EXPR:  PHASEY
PROG:  QPGMODUL                                                    *
EXHM:  MODUL
```

Die Eintragungen in den Programm- und Datei-Steuertabellen gelten für andere TP-Steuerprogramme sinngemaeß.

Die Angabe der Programmgröße ist eine Circa-Angabe. Die exakte Programmgröße kann durch Umrechnung der bei 'CPGPND' angegebenen Adresse ermittelt werden.

Die TWA-Größe ist die Größe des erforderlichen Arbeitsspeichers. Sie wird exakt gerechnet. In der TWA-Größe sind jedoch vom Benutzer eingefügte Copy-Books nicht enthalten. Eine Kontrolle ist auch immer dann erforderlich, wenn die Daten-Ein- und -Ausgabe über Datasets erfolgt, da unterschiedliche Datenbank-Systeme unterschiedliche Bereiche in der TWA belegen. Die TWA-Größe errechnet sich aus der Adresse bei CPGTND minus der Länge der TCA, die normalerweise 256 Bytes groß ist.

Die Terminal-IO-Area (TIOA) darf maximal 4080 Bytes groß sein. Eine größere TIOA führt zu Programmfehlern. Wenn der Wert das Maximum übersteigt - dies kann vorkommen, wenn viele kleine Felder auf den Bildschirm ausgegeben werden - ist die Terminal-Ausgabe zu teilen.

Alle im Programm verwendeten Schalter werden in sortierter Reihenfolge in der Checkliste angelistet.

Alle externen Programmverbindungen (maximal 100) werden am Ende der Checkliste in der beschriebenen Form angelistet. Der Programmierer muss dabei sicherstellen, dass bei Verwendung der EXITP oder EXPR-Operation mit dem Phasennamen in Faktor 2 keine Schleifen entstehen, da diese zu erheblichen Performance-Verlusten führen können.

Bei fehlerfreien Programmen können in bestimmten Fällen noch Warnings am Ende der Checkliste erscheinen. Siehe Abschnitt 6920.

Cross Reference 2247

Im Anschluss an die Umwandlung kann auf Wunsch eine Cross-Reference-Liste ausgedruckt werden. Hierzu ist in der H-Karte Spalte 16 ein 'X' einzutragen.

Die Cross-Reference-Liste gibt Auskunft darüber, bei welchen CPG-Statement-Nummern die folgenden Programm-Elemente verwendet wurden:

- Dateinamen
- Bezugszahlen und Anzeiger
- Alphanumerische Konstanten
- Datenfelder, Feldgruppen und Tabellen
- Numerische Konstanten
- Sprungmarken, interne und externe Unterprogramme
- Operationen

Zusätzlich werden die im Programm verwendeten TOP-Maps sowie die Eintragungen der Standard-Header-Karte (Phase CPGSTH) in der Cross-Refer-

ence aufgelistet.

Programme mit Cross-Reference benötigen bei der Umwandlung eine Size von mindestens 256K.

QSF und QTF Reference

Wurden im Programm MAP- oder LIST-Operationen verwendet, so können die benutzten QSF-Maps oder die QTF-LIST-Dokumente am Ende der Umwandlungsliste protokolliert werden.

Programm-Size 2248

Wird in Spalte 15 ein 'P'(Programmsize) eingetragen,so wird das Blockschaltbild nicht angelistet.Die Listausgabe hat dann folgenden Aufbau:

```
Phase --- +++ ------
Phase -10 ++4 ----14
```

Die erste Ziffer gibt an, wieviel Bytes an Assembler-Codes bei dem jeweiligen CPG-Statement generiert werden.

Die zweite Ziffer gibt an, wieviel Bytes an Literals bei dem jeweiligen CPG-Statement generiert werden

Die letzte Ziffer gibt an, wieviel Bytes im generierten Assemblerprogramm bis zu dem jeweiligen CPG-Statement benötigt werden. Dieser Wert kann gegenüber dem Assemblerprogramm größer sein, da mehrfach vorkommende Literale vom Assembler nur einmal definiert werden.

Programmablauf 2250

Schalter 2260

Der Programmablauf des CPG kann mit Hilfe von Bezugszahlen gesteuert werden. Die Bezugszahlen 01 bis 99 können vom Programmierer gesetzt oder gelöscht werden oder als Ergebnis-Bezugszahlen bei Rechen- und Vergleichsoperationen verwendet werden.

Neben den Schaltern 01 bis 99 bietet CPG die Terminal-Schalter T1 bis T9 und die systemweiten Schalter C1 bis C9 an, die wie normale Bezugszahlen gesetzt oder gelöscht werden können.

Die Schalter T1, T2 usw. bis T9 sind programmunabhängige Schalter in der Terminal-Tabelle. Der Schalter T3 kann zum Beispiel im Programm 1 gesetzt, im Programm 2 und 3 abgefragt und in Programm 4 gelöscht werden. Die Schalter sind dabei terminalabhängig, das heisst, ein Schalter, der von Terminal 1 gesetzt wurde, kann nicht von Terminal 2 abgefragt werden. Zu beachten ist jedoch, dass T1- bis T9-Schalter nur durch den Operationscode 'SETON' gesetzt werden können.

Bei HL1-Batch sind die Schalter U1-U9 unterstützt. Sie können anstelle der Schalter T1-T9 benutzt werden (T1=U1 usw). Durch das JCL-Statement // UPSI xxxxxxxx können die Schalter U1-U8 (bzw. T1-T8) zur Ausführungszeit gesetzt werden.

Die Schalter C1, C2 usw. bis C9 sind programmunabhängige Schalter in der Common System Area. Ihre Anwendung ist ähnlich wie die der T-Schalter, jedoch sind die C-Schalter außerdem noch terminalunabhängig, das heisst, ein Schalter C4, der von Terminal 1 mit Programm 1 gesetzt wurde, kann von Terminal 3 im Programm 4 abgefragt werden.

Bei HL1-Batch sind die Schalter C1-C9 nicht unterstützt.

Mit Hilfe der T- und C-Schalter können folglich Bedingungen für Folgeprogramme gesetzt und Bedingungen aus vorherigen Programmen abgefragt werden.

Darüber hinaus können folgende Schalter vom Terminal-Benutzer zur Ausführungszeit des Programms durch Tastendruck gesetzt werden. Diese Schalter können vom Programmierer nur abgefragt werden:

P1	X'F1'	Programmfunktionstaste 1
P2	X'F2'	Programmfunktionstaste 2
P3	X'F3'	Programmfunktionstaste 3
P4	X'F4'	Programmfunktionstaste 4
P5	X'F5'	Programmfunktionstaste 5
P6	X'F6'	Programmfunktionstaste 6
P7	X'F7'	Programmfunktionstaste 7
P8	X'F8'	Programmfunktionstaste 8
P9	X'F9'	Programmfunktionstaste 9
PA	X'7A'	Programmfunktionstaste 10
PB	X'7B'	Programmfunktionstaste 11
PC	X'7C'	Programmfunktionstaste 12
Q1	X'C1'	Programmfunktionstaste 13
Q2	X'C2'	Programmfunktionstaste 14

Q3	X'C3'	Programmfunktionstaste 15		
Q4	X'C4'	Programmfunktionstaste 16		
Q5	X'C5'	Programmfunktionstaste 17		
Q6	X'C6'	Programmfunktionstaste 18		
Q7	X'C7'	Programmfunktionstaste 19		
Q8	X'C8'	Programmfunktionstaste 20		
Q9	X'C9'	Programmfunktionstaste 21		
QA	X'4A'	Programmfunktionstaste 22		
QB	X'4B'	Programmfunktionstaste 23		
QC	X'4C'	Programmfunktionstaste 24		
A1	X'6C'	Programmabruftaste	PA1	Bei der Verwendung von PA-
A2	X'6E'	Programmabruftaste	PA2	Tasten werden keine Daten
A3	X'6B'	Programmabruftaste	PA3	vom Bildschirm eingelesen.
DE	X'7D'	Daten-Freigabe		
SP	X'7E'	Lichtstift-Abfrage / Cursor-Select-Taste (Pos. Ausw.)		
CL	X'6D'	Löschtaste		

Diese Tasten bleiben auch bei Programmverbindungen bis zur nächsten Bildschirmeingabe erhalten.

Die Schalter P1-P9, Q1-QC, A1-A3, DE, SP und CL sind im Batch nicht unterstützt.

Bei HL1-Batch ist der NI-Schalter nicht unterstützt.

Bei der Verarbeitung von Plattendateien werden die folgenden Schalter gesetzt, wenn die entsprechende Bedingung auftritt. Die Schalter können vom Programmierer unmittelbar nach der Ein-/Ausgabe-Operation abgefragt werden.

EF	Datei-Ende Transient Data (Queue Zero)
EF	Datei-Ende bei sequentieller Plattenverarbeitung.
EF	Bei Dateiende bei Temporary Storage Queuing bzw. Fehler
EF	Wenn zum Hinzufügen bei Temporary Storage Queuing kein Platz mehr vorhanden ist.
EF	Bei Datei-Anfang, bei READB (nur VSAM)
EF	Bei der FIND-Operation wurde das Tabellenende erreicht.
EF	Wenn mit OPEN im Batch eine leere VSAM Input Datei eröffnet wurde.
EF	Wenn mit TWALD versucht wurde, eine TWA zu laden, die vorher nicht mit TWASV gerettet wurde, oder deren Größe sich gegenüber der aktuellen TWA verändert hat. Das kann geschehen, wenn eine neue Programmversion eingesetzt wurde.
EF	Wenn mit LOADT versucht wurde, einen nicht gesicherten Bildschirm (SAVET) zu laden.
EF	Wenn bei Programmaufrufen mit EXITP Programmname oder EXPR das angegebene Programm nicht in der PPT liegt.
EF	Wenn bei Programmaufrufen mit EXITI entweder die aufgerufene Task oder der angesprochene Bildschirm nicht in der entsprechenden CICS-Tabelle liegt.
DR	Doppelter Satz beim Hinzufügen.
DR	Doppelter Schlüssel bei READ Alternativindex.

Bei sequentieller Plattenverarbeitung wird vom CPG der Schalter 'EF' (End of File) gesetzt, wenn der letzte Satz der Datei gelesen wurde. Dieser Schalter kann vom Programmierer ebenfalls nur abgefragt werden.

Beim Hinzufügen von Sätzen zu einer Isam- oder Vsam-Datei wird der Schalter DR gesetzt, wenn ein Satz mit gleichem Schlüssel bereits vorhanden ist. Der Schalter wird bei der nächsten 'ADD'-Ausgabe wieder gelöscht.

Tritt der Fehler DR beim sequentiellen Lesen eines Pfades einer VSAM-Datei auf, so bedeutet dies, dass mehrere Sätze mit gleichem Key vorhanden sind. Wird der letzte Satz einer Gruppe gelesen, so wird der DR-Schalter gelöscht.

Nach einer Bildschirmeingabe kann der Schalter IC (Invalid Character) abgefragt werden, der anzeigt, ob bei der Eingabe in eines der eingelesenen numerischen Felder ein nicht numerisches Zeichen gefunden wurde (z.B. Buchstabe o für Null). Der Schalter muss jedoch unmittelbar nach der READ- oder EREAD-Operation abgefragt werden.

Nach einer Bildschirmeingabe kann der Schalter NI (No Input) abgefragt werden. Werden vom Bildschirm keine Daten eingelesen, so ist der Schalter NI gesetzt. Der Schalter muss jedoch unmittelbar nach der READ- oder EREAD-Operation abgefragt werden.

Nach einer Ausgabe mit EXCPT kann der Schalter 'NO' (No Output) abgefragt werden. Der Schalter wird gesetzt, wenn der EXCPT-Befehl nicht zu einer Ausgabe führt.

Der Schalter OF zeigt in Batchprogrammen bei Druckausgaben an, dass die Überlaufzeile überschritten wurde.

Der Inhalt von Bezugszahl '00' und CPGCxx hat folgende Bedeutung:

X'F0' Sequentielles lesen
X'F8' EF-Bedingung
X'04' DR-Bedingung
X'02' CHAIN U
X'01' ETC, Shadow File busy

Gruppenstufen 2270

Gruppenstufen (L0 bis L9) werden weitgehend wie beim RPG unterstützt, jedoch kennt CPG keine Detail- oder Totalausgaben. Ausgaben zur Gruppenzeit werden durch eine EXCPT-Anweisung aus den Total-Rechenbestimmungen verursacht.

Inkompatibilitäten zu RPG sind auf der Folgeseite aufgeführt.

Anders als beim RPG erfolgt der Gruppenwechsel nicht vor Beginn der Rechenbestimmungen, sondern unmittelbar bei der READ- oder CHAIN-Operation für eine Datei, d. h. die Rechenbestimmungen werden an dieser Stelle bei einem Gruppenwechsel zur Durchführung der Gruppenberechnungen unterbrochen. Die READ- oder CHAIN-Operation wirkt dabei wie die Operation EXSR. In den Total-Rechenbestimmungen darf kein Dateizugriff mit Gruppenstufenschalter durchgeführt werden.

Gruppenstufen können bei Platten-Dateien und bei Datasets verwendet werden. Ebenso können Gruppenstufen bei der Batch-Ausführung auch bei READER, TAPE und bei sequentiellen Plattendateien verarbeitet werden. Gruppenstufenschalter sind bei HL1-Dataset-Verarbeitung nicht unterstützt.

Der Gruppenwechsel ist kompatibel zur RPG-Logik. Bei End-Of-File werden alle Gruppenschalter L1-L9 angesetzt und die Total-Rechenbestimmungen durchlaufen. Vor der Verarbeitung des ersten Satzes erfolgt keine Total-Verarbeitung.

Beispiel für Gruppenstufen:

```
  1    01 001 IPLATTE SF
  2    01 002 I                                    1   7 KDNR  L2
  3    01 003 I                                    8  15 AUTRAGL1
  4    01 004 I                                   16  80 SATZ

  5    02 001 C           LESEN     TAG
  6    02 002 C                     DEBUG
  7    02 003 C           KDNR      READ PLATTE
  8    02 004 C   EF                GOTO ENDE
  9    02 004 C   L1                Z-ADD0    ANZPOS  30
 10    02 005 C   L1      ANZAUF    ADD  1    ANZAUF  30
 11    02 006 C           ANZPOS    ADD  1    ANZPOS
 12    02 007 C                     DEBUG
 13    02 008 C                     EREADBILD
 14    02 009 C                     GOTO LESEN
 15    02 011 C           ENDE      TAG
 16    02 010 CL1                   DEBUG
 17    02 011 CL1                   EREADBILD           02
 18    02 012 CL2                   EREADBILD           03

 19    03 001 OBILD    E E      01
 20    03 002 O                         SATZ     180
 21    03 003 O        E E      02
 22    03 004 O                         ANZPOS   104 '   '
 23    03 005 O                                  115 'POSITIONEN'
 24    03 006 O        E E      03
 25    03 007 O                         ANZAUF   104 '   '
 26    03 008 O                                  114 'AUFTRÄGE'
```

Verfolgen wir den Ablauf des Programms mit Hilfe der programmierten DEBUG-Operationen, so erhalten wir folgendes Ergebnis:

STMT	L1	L2	01	02	03	KDNR	AUFTRAG	ANZPOS	ANZAUFT	SATZ
6								0	0	
12	L1	L2				004711	08150001	1	1	1
6	L1	L2				004711	08150001	1	1	1
12						004711	08150001	2	1	2
6						004711	08150001	2	1	2
16	L1					004711	08150001	2	1	2
12	L1					004711	08150002	1	2	3
6	L1					004711	08150002	1	2	3
12						004711	08150002	2	2	4
6						004711	08150002	2	2	4
16	L1	L2				004712	08150002	2	2	4
12	L1	L2				004712	08150003	1	1	5
6	L1	L2				004712	08150003	1	1	5
12						004712	08150003	2	1	6

Unterschiede zum RPG:

1. In einem Verarbeitungsschritt kann ein Gruppenstufenschalter nur einem Feld zugeordnet werden, d.h.

   n i c h t möglich ist (wegen L2)

```
 IDATEI   KF
 I                                      3   8 FELD1 L2
 I                                     12  14 FELD2 L2
 I                                     15  17 FELD3 L1
```

   möglich ist natuerlich

```
 IDATEI   KF
 I                                      3   8 FELD1 L2
 I                                     15  17 FELD3 L1
 IDATEI2  KF
 I                                      9  14 FELD2 L2
 I                                     15  17 FELD3 L1
```

2. Werden Gruppenstufenschalter bei mehreren verschiedenen Feldern angewandt, so müssen diese Felder die gleichen Feldeigenschaften haben, also gleich lang sein und vom gleichen Feldtyp (alpha / numerisch) sein.

   N i c h t möglich ist (wegen unterschiedlicher Länge bei L2)

```
 IDATEI   KF
 I                                      3   8 FELD1 L2
 I                                     15  17 FELD3 L1
 IDATEI2  KF
 I                                     12  14 FELD2 L2
 I                                     15  17 FELD3 L1
```

3. In Spalte 7 und 8 der C-Karten sind nur die Einträge L0 bis L9 unterstützt, also z.B. n i c h t LR.

'Total LR' kann aber dennoch genutzt werden, indem man stattdessen Schalter L9 abfragt (vorausgesetzt, dass L9 in der Eingabe nicht eingetragen ist).

4. N i c h t unterstützt ist der Einsatz von Gruppenstufenschaltern in Kombination mit der Prüfung von Zeichen im Datensatz.

   Die im folgenden beschriebene Verarbeitung führt deshalb zu nicht vorhersehbaren Ergebnissen für die Schalter L1 und L2, weil jeweils nur eines der beiden Felder eingelesen wird.

```
IDATEI   KF   02    2 C2
I                                      1   5 KEY   L2
I        KF   01    2 C1
I                                     21  25 KDNR  L1
```

Programmverbindungen 2280

Mit den Operationen EXITD, EXITI, EXITP, EXITT und EXPR werden Programmverbindungen hergestellt. Das gleiche gilt auch für die Operation EXHM in einem HL1-Programm

Über die Operation EXITP kann der Programmierer von einem Programm in ein anderes verzweigen.

Wenn dabei nicht auf Daten oder Bezugszahlen zurückgegriffen werden soll, so trägt der Programmierer in Faktor 2 den 4-stelligen Transaktionsschlüssel des Programms ein, in das verzweigt werden soll, und zwar in der Form:

```
EXITP'XXXX'
```

Dabei ist XXXX der Transaktionsschlüssel. Das Folgeprogramm wird danach in den Hauptspeicher geladen und der Speicherplatz für das Ursprungsprogramm wird freigegeben.

Will der Programmierer im zweiten Programm auf Feldinhalte des ersten Programms zurückgreifen, so hat er folgende Möglichkeiten:

1. Übernahme der gesamten TWA ins Folgeprogramm.

Hierzu wird in Faktor 2 der EXITP-Operation der Phasenname des aufzurufenden Programms eingetragen, in der Form:

```
EXITPXXXXXXXX
```

wobei XXXXXXXX der 8-stellige Phasenname ist. Bei dieser Art der Verzweigung muss der Programmierer folgendes beachten: Die Transaction-Work-Areas (TWA) beider Programme werden vom TP-Steuerprogramm übereinandergelegt und müssen folglich für beide Programme deckungsgleich sein. Der Programmierer muss dazu

1. die Anzahl der Stellen eintragen, die aus dem Vorprogramm übernommen werden soll.

2. Die Felder des Ursprungsprogramms müssen in der er-

weiterten Dateizuordnung des 2. Programms definiert werden, und zwar in der Reihenfolge, wie sie im Ursprungsprogramm definiert wurden. Hier kann in Zweifelsfällen die TWA-Auflösung am Ende der Assembler-Umwandlungsliste weiterhelfen.

3. In der PCT (Programm Control Tabelle) des TP-Steuerprogramms muss die größte in Programmverbindungen benutzte TWA-Size für die Task eingetragen sein.

Bei dieser Verbindung ist zu beachten, dass die Rückkehr zum Ursprungsprogramm nicht mit der gleichen EXITP-Operation erfolgen darf, da hierbei der vom aufrufenden Programm benutzte Speicher nicht freigegeben werden kann. Wegen der Problematik dieser Methode sollte möglichst einer der unter (2160) beschriebenen Methoden der Vorzug gegeben werden.

Vor Benutzung der EXITP- bzw. EXPR-Operation werden alle Dateien mit dem Befehl RNDOM*ALL intern freigegeben.

Bei der Eintragung 'S' in Spalte 53 entfällt die Einschränkung, dass die Operation EXPR nicht in einer Subroutine verwendet werden darf.

Außerdem empfehlen wir, im aufgerufenen Programm (bei EXPR) ebenfalls vor dem Rücksprung alle Dateien mit dem Befehl RNDOM freizugeben. Im aufrufenden Programm wird dann mit den Operationen CLEAR und INDOF die Verarbeitung fortgesetzt. (Daten evtl. über temporären Speicher weitergeben).

Besonderheit bei der Bezugszahlenbehandlung bei EXPR mit Spalte 53 = S

```
    C                     EXPR PROG2              S
```

PROG2 wird mit diesem Befehl aus PROG1 aufgerufen. Zur Zeit des Aufrufs sind die Schalter 01 und 02 gesetzt. Nach dem EXPR sind 01 und 02 auch in PROG2 gesetzt. In PROG2 wird dann Schalter 99 gesetzt. Nach dem Rücksprung in PROG1 ist dort der gleiche Zustand zu beobachten wie vor dem EXPR: Schalter 01 und 02 sind gesetzt, 99 nicht.

Wird allerdings in der H-Karte von PROG2 in den Spalten 27 bis 30 die Anzahl zu übernehmender Stellen aus der TWA von PROG1 angegeben, so werden alle in PROG2 gesetzten Schalter beim Rücksprung nach PROG1 übertragen: Im Beispiel wären also 01, 02 und 99 gesetzt.

Die Operation EXITD entspricht der Operation EXITI, wobei zusätzlich noch die Möglichkeit besteht, Daten auszutauschen.

Die Operation EXHM entspricht der Operation EXPR, wobei die Daten jedoch über einen Datenkanal und nicht mehr über die Transaction Work Area ausgetauscht werden. Jedes Unterprogramm benutzt dabei seine eigene private Workarea. Die EXHM-Operation ist nur in der HL1-Version unterstützt.

Bei den Programmverbindungen EXITT und EXITP unter Command Level Steuerung kann die CICS-Common Area benutzt werden. Diese Area wird im aufrufenden Programm mit EDIT CPGCOM aufbereitet und kann im aufgerufenen Programm mit SELCT CPGCOM aufgerufen werden. Die maximale Größe der Common Area ist 4000 Bytes.

Ein- und Ausgabe 2300

Dateinamen 2301

Dateinamen sind wie bei RPG maximal 8 Stellen lang. Das 1. Zeichen muss ein Buchstabe von A bis Z, ein '$' oder ein '#' Zeichen sein. Andere Sonderzeichen sind nicht zulässig.

Dateinamen dürfen nicht mit Feldnamen übereinstimmen. Die Prüfung für eine CPG-Fehlermeldung kann nur von Stelle 1 bis 6 vorgenommen werden.

In einem Programm dürfen maximal 100 Dateien definiert werden.

Schlüssel 2304

Bei den Operationen CHAIN, DELET, READ, READP, UPDAT, SETLL und WRITE muss ein Schlüsselfeld oder Schlüsselwert in Faktor 1 der Rechenbestimmungen angegeben werden.

Ein alphanumerischer Schlüsselwert kann als in Hochkommata eingeschlossene Konstante spezifiziert werden. Eine solche Konstante kann maximal 8 Bytes lang sein. Ist eine angegebene Konstante kürzer als die für die Datei in den Dateizuordnungen definierte Schlüssellänge, so wird der Rest mit X'00' formatisiert.

Hat der Schlüssel numerisches (d.h. gepacktes) Format, so kann der Schlüsselwert nicht als dezimale Konstante angegeben werden.

Wenn der Schlüssel als in der TWA definiertes Feld angezogen wird, sollte die Länge des Feldes der für die Datei in den Dateizuordnungen definierten Länge entsprechen. Ist es kürzer als die Schlüssellänge, so werden die restlichen Bytes mit X'00' formatisiert. Ist es länger als die Schlüssellänge, so wird es abgeschnitten (d.h. rechtsbündige Bytes werden ignoriert).

Wenn Dateien mit Schlüsselfeldern im gepackten Dezimalformat verarbeitet werden, muss mit Vorsicht vorgegangen werden. Für die IBM-Hardware gelten die Hexadezimalwerte 'C' und 'F' bei der Verarbeitung numerischer Felder als positives Vorzeichen. Dies kann zu einem scheinbaren Nichtauffinden des Satzes führen, wenn im Dateischlüssel das Vorzeichen 'F' codiert ist und im Programm ein Feld mit dem Vorzeichen 'C' spezifiziert ist. Das Vorzeichen des Schlüsselfeldes kann mit der MLLZO-Operation korrigiert werden.

Ist das Vorzeichen des Satzschlüssels in der Datei ein 'C' und steht im Programmfeld ein 'F' (wenn es z.B. von einem Bildschirm in ein numerisches Feld eingelesen wurde), kann das Vorzeichen des Programmfeldes mit Hilfe einer Z-ADD-Operation in ein 'C' umgewandelt werden.

Zusammengesetzte Schlüsselfelder können mit Hilfe der EDIT-Funktion oder durch Definition von Overlay-Feldern oder durch Datenstrukturen aufgebaut werden.

Datenbankfunktionen 2305

Sequentieller oder wahlfreier Zugriff 2306

Eine in einem CPG-Programm definierte DISK-Datei kann entweder wahlfrei (d.h. für jeden gelesenen Satz wird ein eindeutiger Schlüssel angegeben) oder sequentiell verarbeitet werden. Die Wirkung der CPG-Dateioperationen hängt bis zu einem gewissen Grade davon ab, wie eine Datei verarbeitet wird.

Zu Beginn einer Transaktion wird davon ausgegangen, dass alle für das Programm definierten DISK-Dateien wahlfrei (RANDOM) verarbeitet werden sollen. Die in diesem Modus unterstützten Dateioperationen sind CHAIN, WRITE, UPDAT, DELET, EXCPT und RNDOM.

Wenn eine READ-, READP- oder READB-Operation für eine DISK-Datei ausgeführt wird, wird automatisch für diese Datei sequentieller Verarbeitungsmodus gesetzt.

In diesem Modus werden die Dateioperationen READ, READP, READB, SETLL, und CHAIN unterstützt.

Sobald eine Datei auf den sequentiellen Modus umgestellt worden ist, bleibt sie darin, bis eine RNDOM-Operation für diese Datei ausgeführt wird. Diese bewirkt, dass die Datei wieder auf den wahlfreien Zugriff umgeschaltet wird.

Dateioperationen im sequentiellen Zugriff 2307

Die sequentielle Verarbeitung einer DISK-Datei wird durch Ausführung einer READ-, READB-, (nur VSAM) oder READP-Operation eingeleitet. Die Datei verbleibt im sequentiellen Modus, bis die wahlfreie Verarbeitung für die Transaktion mit Hilfe einer RNDOM-Operation wieder hergestellt wird.

Diese Leseoperationen dienen zur sequentiellen Verarbeitung einer ganzen Datei oder bestimmter Teile einer Datei (sprungsequentielle, generische Verarbeitung oder Suchlauf).

Mit der READB-Operation wird eine VSAM-Datei in Schlüsselfolge rückwärts abgearbeitet (d.h. in umgekehrter Schlüsselfolge).

Der Schlüsselwert dient lediglich zur Positionierung der Datei beim Lesen des ersten Satzes, wenn die sequentielle Verarbeitung gestartet wird. Bei einer READ- oder READP-Operation ist dies der erste Satz in der Datei, dessen Schlüssel gleich oder größer als der angegebene Schlüssel ist. Bei einer READB-Operation ist dies der Satz, dessen Schlüssel gleich dem spezifizierten Schlüssel ist. Ist der Schlüsselwert größer als der des letzten Satzes in der Datei (READ, READP) oder kleiner als der des ersten Satzes in der Datei (READB), so wird der Dateiende-Anzeiger (EF) 'EIN' gesetzt. Dieser muss unmittelbar nach der Leseoperation abgefragt werden.

Sobald die sequentielle Verarbeitung eingeleitet ist, lesen alle nachfolgenden Leseoperationen den nächstfolgenden Satz in der Datei. Dies geschieht ohne Rücksicht auf den tatsächlich angegebenen Schlüsselwert. Es ist ein Fehler, den Schlüssel wegzulassen.

Wenn das Dateiende festgestellt wird, wird der EF-Anzeiger 'EIN' gesetzt, und die Eingabebestimmungen für die Datei werden nicht ausgeführt. Das Programm wird fortgesetzt mit der nächsten Anweisung in den Rechenbestimmungen hinter der Leseoperation. Werden für die Datei weitere sequentielle Leseoperationen ausgeführt, so werden keine Sätze gelesen und der EF-Anzeiger wieder 'EIN' gesetzt.

Bei VSAM muss die EF-Bedingung unbedingt abgefragt werden. Wird dies vergessen, so meldet CPG-Beim nächsten Lesen den Systemfehler SEQ EF. Siehe Abschnitt 6950.

Gemischte Verarbeitung 2308

Durch die Kombination der Operationen READ, READP, RNDOM, SETLL und CHAIN kann ein und dieselbe Datei innerhalb desselben Programms sowohl wahlfrei als auch sequentiell verarbeitet werden.

SAM - Sequentielle Dateien 2311

Die Verarbeitung sequentieller Dateien ist im Online-Betrieb nicht möglich, da bei einem ungeplanten Systemabbruch die Datei nicht mehr abgeschlossen werden kann und damit alle Daten verloren gehen würden.

Die TP-Monitore bieten daher für diese Dateien meist einen eigenen Service an, der von CPG durch Transient Data (siehe 2160) unterstützt wird. Eine bessere Performance wird jedoch u.E. durch vorformatisierte ISAM- oder VSAM-Dateien erreicht.

Im HL1-Batch sind sequentielle Dateien unterstützt.

DAM - Direkter Zugriff 2312

DAM-Dateien werden vom CPG wie normale ISAM-Dateien behandelt.

Bei dieser Zugriffsart wird der Zugriff jedoch sehr an den physischen Plattentyp gebunden. Der Schlüssel einer DA-organisierten Datei wird immer errechnet und danach in binären oder gezont dezimalen Werten dem System mitgeteilt.

Bevor der Zugriff zu einer DA-Datei erstellt wird, muss geprüft werden, in welcher Weise der Zugriff zu erfolgen hat. Es gibt zwei grundsätzlich unterschiedliche Methoden, dies durchzuführen: Erstens den Zugriff über physische oder relative Satzadressierung und zweitens über den physischen Schlüssel (KEYED).

Beim Zugriff über den physischen Schlüssel ist dieser abhängig von der physischen Spuradresse und muss der Länge entsprechen, die in der

File Control Tabelle eingetragen ist.

Es gibt folgende Möglichkeiten, auf DA-Dateien zuzugreifen:

(1) Relative Spur und Satznummer (RELTYPE=HEX)
Dies setzt einen dreistelligen binären Schlüssel voraus. Die zwei ersten Stellen enthalten die relative Spur und die dritte Stelle enthält die relative Satznummer innerhalb der Spur. (Satznummer 0 ist nicht zulässig.) Format = TTTTRR

(2) Relative Spur und Satznummer (RELTYPE=DEC)
Bei diesem Zugriff wird ein achtstelliger gezonter Schlüssel vorausgesetzt. Die ersten sechs Stellen enthalten die relative Spurnummer (relativ zu 0) und die Stellen sieben und acht enthalten die relative Satznummer innerhalb der Spur. (Satznummer 0 ist nicht zulässig.) Format = TTTTTTRR

(3) Physische Plattenadresse.
Bei diesem Format wird ein achtstelliger binärer Schlüssel verwendet. Die ersten drei Stellen enthalten binär Null, die folgenden zwei Bytes die Nummer des CYL., die folgenden zwei Bytes die Spurnummer innerhalb des CYL., und das letzte Byte die relative Satznummer. Format ist also : 000000CCCCHHHHRR.

Der binäre Schlüssel kann über die EDIT-Operation entsprechend aufbereitet werden. Die Dateizuordnung muss dabei die Schlüssellänge enthalten, die ihrer Zugriffsart entspricht.

Achtung: Diese Verarbeitungsform ist nur im CICS Macro Level unterstützt.

ISAM-Dateien 2313

ISAM-Dateien werden grundsätzlich wie VSAM-Dateien verarbeitet.

1. Umschaltung auf sequentielle oder wahlfreie Verarbeitung in einem Programm.

Der Dateizugriff erfolgt durch die Operationen CHAIN und READ, wobei CHAIN einen wahlfreien Zugriff und READ einen sequentiellen Zugriff bewirkt.

Die erste in einem Programm durchlaufene READ-Operation schaltet die Verarbeitungsart von wahlfrei auf sequentiell um. Eine CHAIN- oder SETLL-Operation, die danach erfolgt, liest keine Daten, sondern setzt lediglich einen Pointer für die nächste READ-Operation. Jede folgende READ-Operation liest dann immer den logisch folgenden Satz. Mit der Operation RNDOM kann danach wieder auf wahlfreie Verarbeitung umgeschaltet werden, die dann solange gilt, bis erneut eine READ-Operation erfolgt.

2. Dateischutz bei Update.

Update-Operationen sollten nur bei wahlfreier Verarbeitung durchgeführt werden. Wenn ein Satz für ein UPDATE gelesen wird, so kann dieser Satz durch Eintragung eines U in Spalte 53 der CHAIN-Operation gegen Doppelbeschriftung geschützt werden. Das U bewirkt, dass dieser

Satz solange für ein weiteres UPDATE gesperrt wird, bis er zurückgeschrieben wird. Wird der Satz nicht zurückgeschrieben, so kann er vom Programmierer durch eine RNDOM-Operation wieder freigegeben werden. Wurde ein Satz mit CHAIN U gelesen, so darf kein ADD, kein READ und kein READB für einen solchen Satz durchgeführt werden.

Bei Programmende erfolgt eine automatische Freigabe aller noch gesperrten Sätze.

3. Operationen WRITE und UPDAT

Für Satzlängen bis 256 Bytes oder für Einzelfelder (z.B. Bestand) kann mit Hilfe dieser Operationen Codierarbeit gespart werden. Siehe Operationen (3000) und Beispiel (8000).
Für ungeblockte ISAM-Dateien gilt für den TP-Monitor CICS die Einschränkung (siehe CICS), dass Ausgaben nur mit EXCPT durchgeführt werden können.

VSAM-Dateien 2314

VSAM-Dateien werden wie ISAM-Dateien behandelt. Lediglich ein 'V' in Spalte 32 der zugehörigen Dateizuordnung definiert die Datei als VSAM Datei. Dies braucht jedoch nicht eingetragen zu werden, da es intern wie ISAM gehandhabt wird.

Es können Dateien mit Satzlängen von maximal 8172 Bytes verarbeitet werden.

Bei ISAM oder VSAM-Dateizugriffen unter CICS gelten bestimmte Regeln, wie zum Beispiel nach einem READ darf kein CHAIN U durchgeführt werden. Wenn gegen solche Regeln verstoßen wird, erscheint die Fehlermeldung Programmfehler. Siehe hierzu Abschnitt 6950.

In der CICS-FCT muss Recform = Blocked eingetragen werden.

VSAM-Dateien mit variabler Satzlänge werden in den Eingabe-, Rechen- und Ausgabebestimmungen wie Dateien mit fester Satzlänge behandelt.

In der Dateizuordnung muss dabei in Spalte 19 ein 'V' für variable Satzlänge und im Feld Satzlänge die Länge des größten Satzes eingetragen werden.

Anmerkung:

Die Eingabebestimmungen verarbeiten den Satz feldweise, auch wenn der Satz kürzer ist als die in den Eingabebestimmungen angegebene letzte Stelle. Dies kann zu Programmfehlern führen, wenn das Feld gepackt ist. In den Eingabebestimmungen sollten deshalb Auswahlkriterien wie z.B. eine Satzart angewandt werden.

VSAM Dateien, die im CICS mit READ, READB oder CHAIN U/P gelesen werden, belegen einen VSAM String, solange die TASK besteht, oder bis der String mit RNDOM freigegeben wird.

VSAM-Alternativindizes 2316

Eine VSAM-Datei kann mit einem Alternativschlüsselfeld gelesen werden. Dieses Leseverfahren mit Alternativschlüsseln erfordert einen weiteren Eintrag in der Dateitabelle (FCT).
Wird ein Satz im Basiscluster hinzugefügt und existiert der Alternativschlüssel (unique) bereits, so wird der Schalter DR angesetzt.

Der DR-Anzeiger wird 'EIN' gesetzt, wenn eine Datei mit nicht eindeutigem Schlüssel gelesen wird. Das Feld CPGFRC wird hierbei auf 'DK' (Duplicate Key) gesetzt.

In einem nachfolgenden Lesen mit eindeutigem Schlüssel wird der DR-Anzeiger 'AUS' gesetzt.

VSAM-ESDS/RRDS 2316

Wird bei einem Dateizugriff ein numerisches Keyfeld eingetragen, so erfolgt die binäre Aufbereitung automatisch. Wenn ein alphanumerischer Schlüssel eingetragen wird, ist der Programmierer für die binäre Aufbereitung verantwortlich.

Beispiele für die RRDS-Verarbeitung siehe Seite 8070 und 8071.

Beispiele für die ESDS-Verarbeitung siehe Seite 8080 und 8081.

VSAM-Dateien in Zugangsfolge 2317

Dateien in Zugangsfolge (auch eingabesequentielle Dateien genannt, ESDS – Entry Sequenced Data Sets) werden unterstützt. Wenn einer Datei in Zugangsfolge Sätze hinzugefügt werden, werden diese an das Ende der Datei angehängt. Ein Schlüssel bei einer EXCPT- oder UPDAT-Operation wird ignoriert.

Wenn ein Satz gelesen wird, muss die relative Byteadresse (RBA) des Satzes als Schlüssel angegeben werden. Die RBA wird als Summe der Bytes sämtlicher zuvor aus dieser Datei gelesenen Sätze berechnet. Dies ist nur bei Sätzen fester Länge ohne weiteres möglich.

In Spalte 31 der Dateizuordnungen muss ein 'R' codiert sein, das die relative Byteadressierung definiert.

Um sicherzustellen, dass die hinzugefügten Sätze auch tatsächlich auf Platte gespeichert werden, sollte nach einem ADD oder NEWRC der spezielle Satz mit einer CHAIN-Operation gelesen werden. Als Key wird das Feld CPGKxx angegeben, wobei xx für die laufende Nummer der Dateizuordnungskarte steht.

DL/I-Dataset 2318

Die Dataset-Logik mit Assembler-Datasets kann heute als veraltet angesehen werden. Sie wurde schon vor Jahrzehnten abgelöst durch die HL1-Datasets im CPG3.

Für die DLI-Verarbeitung ist heute ausschließlich die Verarbeitung mit den Schnittstellen-Operationen DLI, QSSA und USSA zu empfehlen.

DL/I-Datenbanken können aber noch in alten Programmen über die Dataset-Logik des CPG verarbeitet werden wie im Kapitel Dataset beschrieben.

Bei der CPG-Installation wurden folgende Copy-Books mitgeliefert:

A.CPG*DLD	CPG-Beispielprogramm mit DL/I
A.CPG*DL1	DL/I-Dataset

Zur Generierung des DL/I-Datasets muss folgendes Programm erstellt werden:

```
// JOB DATASET
// OPTION DECK
// EXEC ASSEMBLY
        COPY CPG*DL1                * = Releasesuffix
        END
/*
/&
```

Die bei der Umwandlung gestanzten Karten müssen anschließend in die Relocatable Library katalogisiert werden.

Anschließend muss der Datenbankprocessor erstellt bzw. um den Eintrag DL/I erweitert werden.

Dataset 2319

Die Dataset-Logik mit Assembler-Datasets kann heute als veraltet angesehen werden. Sie wurde bereits vor Jahrzehnten abgelöst durch die HL1-Datasets im CPG3.

CPG bietet dem Programmierer die Möglichkeit der Datenbank-unabhängigen Programmierung, das heisst es können logische Dateien angesprochen werden, die physisch nicht existieren. Für eine solche Datei wird der Name Dataset verwendet.

Ein Dataset ist eine Assembler-Routine, die die entsprechenden Ein- und Ausgabe-Befehle für eine oder mehrere Dateien oder die Zugriffe zu einem beliebigen Datenbank-System enthält und die für eine bestimmte Anwendung erforderlichen Daten zu einem logischen Datensatz (Dataset) zusammenstellt.

Der Benutzer kann damit sowohl auf alle denkbaren Datenbanksysteme zugreifen, als auch sein eigenes Datenbank-System entwerfen. Die Dataset-Routinen ermöglichen eine klare Trennung zwischen dem Anwendungsprogramm und der Datenbank. Dataset-Routinen sind austauschbar ohne Änderung der Anwendungsprogramme, die auf diese Routine zugreifen.

Die Umstellung von einer Datei-Organisation auf eine Datenbank-Organisation kann ohne Aufwand für die Anwendungsprogrammierung wie folgt ablaufen:

Soll die Datei 'PLATTE' in eine DL/I-Datenbank integriert werden, so muss zunächst der physische Datei-Zugriff in ein Dataset verlegt werden. Dazu wird ein Dataset für den 1:1 Datei-Zugriff erstellt.

1. Erstellen des Datenbank-Processors.

Der Datenbank-Processor ist eine Methodenbank für die Datenbank-Verwaltung, der im wesentlichen aus der Verknüpfung aller vorhandenen Datasets zusammen mit einer Sprungtabelle besteht.

Mit dem folgenden Programm kann ein Datenbank-Processor für diese Anwendung erstellt werden. Die Erweiterung um ein weiteres Dataset erfordert dann lediglich die Einfügung einer entsprechenden Karte im Copy CPGUCDBP der durch den Kommentar gekennzeichneten Stelle.

```
// JOB GENDBP
// OPTION CATAL
   PHASE CPGDBP,*
// EXEC ASSEMBLY
         COPY  CPG*CDBP
         END
/*
// EXEC LNKEDT
/&


   CATALS A.CPGUCDBP
         BKEND
*---------------------------------------------------------------*
*PLATTE  DC    CL8'PLATTE',V(DPLATTE)        PLATTE BEISPIEL
*DLI     DC    CL8'DLI',V(CPGSDLX)           DL/I BEISPIEL
*---------------------------------------------------------------*
        BKEND
```

2. Laden des Datenbank-Processors.

Der Datenbank-Processor muss wie die Methodenbank vor Beginn der Verarbeitung geladen werden. Der Ladevorgang ist abhängig vom TP-Monitor und erfolgt in derselben Weise wie das Laden der Methodenbank. (Siehe CPG3-Installatinsanweisung Abschnitt 9125, Stelle 6).

3. Anpassung der Programme.

Alle Programme, die auf die Datei PLATTE zugreifen, können jetzt nacheinander ohne Zeitdruck von der physischen auf die logische Datei umgestellt werden. Die Umstellung erfolgt dadurch, dass in der Dateizuordnung für die Datei PLATTE die Eintragung 'DISK' in den Spalten 40 bis 46 durch die Eintragung 'DATASET' ersetzt und das Programm neu umgewandelt wird. Da die Datei PLATTE auch physisch noch existiert, können Programme mit 'DISK' oder 'DATASET' für beliebige Zeit parallel verarbeitet werden. Die Umstellung kann also ohne Termindruck erfolgen.

4. Anpassung der Datenbank.

Wenn alle Programme auf Dataset umgestellt sind, kann die Dataset-Rou-

tine PLATTE, die jetzt nur einmal im Datenbank-Processor liegt, so umgestellt werden, dass die für das Dataset PLATTE erforderlichen Daten z.B. einer DL/I-Datenbank entnommen werden. Nach dem Test wird die Routine im Datenbank-Processor ausgetauscht. Das Anwendungsprogramm bleibt dabei unverändert. Eine Rückkehr zur alten Version ist ohne Programm-Änderungs-Aufwand jederzeit möglich. Ebenfalls ohne Programmänderung ist eine spätere Umstellung auf ein anderes Datenbanksystem möglich.

HL1-Datasets

HL1-Benutzer können auch Datasets als HL1-Module erstellen. Die Programmierung erfolgt mit normalen CPG-Instruktionen. Beschreibung siehe Abschnitt 5085.

Datenbanken

SQL
DL1 sind standardmäßig unterstützt.
VBOMP/EDN

Alle anderen Datenbanken können mit der CALL-Operation (Seite 3130) verarbeitet werden.

VBOMP 2325

Das Datenbank-System VBOMP wird von CPG direkt, das heisst ohne die vorherige Erstellung einer Dataset-Routine im Datenbank-Prozessor, unterstützt.

Eine genaue Beschreibung liefert das VBOMP-Handbuch. Das folgende Beispiel soll nur einen Überblick über die Arbeitsweise der VBOMP Verbindung geben.

Anwendungsbeispiel:

```
  1    01 001 FMASTER  U         200 07         VBOMP

  2    01 050 E    VDBTWA             <--- (nur für CICS)
  3    02 001 IMASTER  KF
  4    02 002 I                                  1   7 FILE
  5    02 003 I                                  8  11 PRCIND
  6    02 004 I                                 12  15 RBA
  7    02 004 I                                 16  17 ERROR
  8    02 005 I                                 18  20 INDIND
  9    02 007 I                                 21  27 KEY
 10    02 008 I                                 29 200 DATEN

 11    03 001 C           KEY       VBOMPMASTER    MRAN
 12    03 002 C                     VSLCTMASTER
       03 002 C*                    VERARBEITUNG
 20    03 022 C                     EXCPT                      01
 21    03 023 C           KEY       VBOMPMASTER    MUPD

 22    04 001 OMASTER  E       01
 23    04 002 o                        DATEN    200
```

Beschreibung:

1 Die Dateizuordnung entspricht der Dateizuordnung für ein Dataset wobei als Einheit VBOMP eingetragen wird. Das Feld Blocklänge bleibt frei. Die Felder Satzlänge und Schlüssellänge müssen eine Eintragung enthalten.

2 Als erste E-Karte muss das Copybuch VDBTWA in die User TWA aufgenommen werden. Dies gilt nur für Online-Programme! Bei Batch-Programmen darf kein TWA Copy Book benutzt werden.

3 Die Eingabe-Satzbestimmung entspricht der Satzbestimmung für die normale Plattendatei.

4 Stelle 1 bis 7 des Datensatzes enthält den Dateinamen, in diesem Falle 'MASTER'.

5 Stelle 8 bis 11 enthält den Processindicator.

6 Stelle 12 bis 15 enthält die RBA auf die zugegriffen wird/wurde.

7 Stelle 16 bis 17 enthält den Error Code.

9 Ab Stelle 21 steht der Schlüssel in der Länge wie in der Dateizuordnung angegeben. In diesem Falle 7 Stellen, das heisst von 21 bis 27.

10 Auf der nächsten verfügbaren Stelle, in diesem Falle Stelle 29, beginnen die Datenfelder des Benutzers, die hier zu einem Feld mit dem Namen DATEN zusammengefasst sind.

11 Mit dieser Operation wird ein Satz RANDOM gelesen (MRAN). Die Operation arbeitet dabei wie eine CHAIN-Operation, allerdings werden die Daten nicht in die TWA des Users übertragen, sondern in einen Zwischenspeicher je Datei.

12 Mit dem Befehl VSLCT werden die Daten aus dem Zwischenspeicher in die TWA übertragen. Dieser Bereich wird vom VBOMP verwaltet.

Zwischen Statement 12 und 20 erfolgt die Verarbeitung der Daten, je nach Anwendung.

20 Die Ausgabebestimmungen mit dem Schalter 01 werden ausgeführt. Im Gegensatz zur Ausgabe für eine Plattendatei werden dabei die Daten jedoch lediglich im Hauptspeicher verändert. Der Satz wird erst mit der folgenden Operation zur Platte zurückgeschrieben.

21 Der veränderte Satz wird auf die Platte zurückgeschrieben (MUPD).

22 Die Ausgabe entspricht der Ausgabe für die Plattendatei.

23 Die Ausgabe der Datenfelder erfolgt sinngemäß wie die Eingabe. Dabei ist zu berücksichtigen, dass die ersten 15 Stellen VBOMP-interne Daten und die folgenden Stellen das Schlüsselfeld enthalten.

Programmierung

In HL1-Batch-Anwendungen kann VBOMP und VSLCT codiert werden, um EDN-Datenbanken im Batch zu verarbeiten. Dabei ist zu beachten:

1. Der EDN-Prefix für Batch ist anders aufgebaut als für Online-Anwendungen. Daher ergibt sich bei VSLCT eine Verschiebung der Daten. Dies ist der EDN-Literatur zu entnehmen.

2. Das Copy VDBTWA darf nicht in einer E-Karte eingetragen werden, da HL1-Batch keine Verschiebung der TWA zulässt.

Beispiel:

```
H                                          B     Batch/EDN
FMASTER  I   F      228   6      VBOMP
D        EDNPFX    0 22          EDN Prefix
D        EDNFIL       7              Filename
D        EDNPI        4              Process Indicators
D        EDNEB        2              Error Bytes
D        EDNRBA       4              RBA
D        EDNIND       5              Indicators
D        KEY          6          EDN Key
D        DATEN      200          EDN Datensatz
IMASTER  KF
I                                     1  22 EDNPFX
I                                    23  28 KEY
I                                    29 228 DATEN
C           KEY       VBOMPMASTER   MRAN
```

```
     C                   VSLCTMASTER
     C           EDNEB   CABEQ'RN'     NOTFND
     C                   ...
     C           NOTFND  TAG
```

DL/I 2330

Das Datenbank-System DL/I wird über ein CPG-DL/I-Interface direkt unterstützt, soweit der eingesetzte TP-Monitor dies zulässt. Für CICS und Shadow steht ein entsprechendes Interface zur Verfügung. DL/I ist sowohl online als auch im Batch unterstützt.

Die Unterstützung umfasst:

- Volle Integration der normalen CPG-Datei-Verarbeitungs-Logik.
- Automatische PSB-Verwaltung
- Interne Auffindung logischer Fehler
- Volle Kompatibilität der Quellenprogramme für verschiedene TP-Steuerprogramme.

Voraussetzungen

Der Einsatz des Interface setzt die Installation eines in den TP-Monitor integrierten DL/I-Systems voraus.

Direkte DL/I Unterstützung ab CPG-Release 1.1.

Im CPG wird der Zugriff zur Datenbank DL/I in den Rechenbestimmungen über die Befehle QSSA, USSA und DLI unterstützt. CPG baut dabei automatisch die internen DL/I-alls auf, die vom Programmierer mit wenigen Operationen aufbereitet werden. CPG verwaltet automatisch PCBs und SSAs. Vom Programmierer können über interne CPG-Felder Return-Codes des DL/I abgefragt werden. CPG setzt den Return-Code ins Feld CPGDRC, welches alphanumerisch und vier Stellen groß definiert ist. Ebenfalls kann über ein internes Feld die Key-Feedback-Area verarbeitet werden. Ein Beispiel für DL/I-Verarbeitung soll die Arbeitsweise verdeutlichen

1. Einträge in der F-Karte.

Es ist für alle Zugriffe zu DL/I eine F-Karte pro Programm notwendig Diese F-Karte sieht wie folgt aus.

```
     FMYPSBNM U   F       7            DLI
```

Der Dateiname muss gleich dem verwendeten PSB-Namen sein. In der FCT muss die Zugriffsberechtigung für das verwendete Programm aufgenommen sein. Die Satzlänge ist fest, wird als Dummy-Eintrag mit 7 Stellen eingetragen. Der Eintrag unter Device (Spalte 40-46) muss DLI sein.

Einträge in der Eingabe.

```
     IMYROOT      DS                          112        (max.  112)
     I                                      1   8 MYCUNR (Segmentlg)
     I                                      2   8 MYCUN7
     I                                      9  33 MYCUNM
```

Die Strukturen der Segmente werden als Datenstruktur in den Eingabebe-

stimmungen beschrieben. Es kann sich auch um eine Überlagerung in den E- oder D- Bestimmungen handeln. Wichtig ist, dass DL/I in der Länge des DL/I-Segmentes einliest. Daher sollte der Bereich immer in der maximalen Größe zur Verfügung gestellt werden. Siehe auch Satzbestimmung bei MYROOT Datenstruktur im vorangehenden Beispiel.

DL/I Aufbau der SSA.

Segment Search Areas werden mit den Befehlen QSSA und USSA intern aufbereitet. Ein einfacher qualifizierter SSA-Aufbau sieht wie folgt aus:

```
     C           'MYROOT  'QSSA 'MYCUNR  'KDNR    8  EQ
```

Im Faktor1 wird der Name des Segmentes eingetragen. Der Segmentname muss in Hochkommata eingeschlossen sein und darf maximal acht Zeichen lang sein.Im Faktor2 wird, ebenfalls in Hochkommata, der Name des sensitiven Argumentes definiert. Im Ergebnisfeld wird der Feldname des programmabhängigen Schlüsselfeldes definiert. Das Feldlängenfeld wird benötigt, um die Anzahl der linksbündig verwendeten Stellen dieses Schlüsselfeldes zu definieren. Der Schalter auf größer enthält einen Vergleichswert. Mögliche Einträge sind: EQ, NE, GT, LT, GE, LE. Unter Runden (Spalte 53) kann ein 'A' oder ein 'O' eingetragen werden, über diesen Eintrag können boolesche Vergleichsoperationen über ein SSA gesetzt werden. 'A' heisst logisch-und-verknüpft, 'O' heisst logisch-oder-verknüpft. Eine solche Verknüpfung zeigt das nächste Beispiel :

```
     C           'MYROOT  'QSSA 'MYCUN1  'NAME    1  EQ
     C                    QSSA 'MYCUCY  'ORT     1 AGE
```

Der Zugriff verläuft mit einem SSA auf das Segment MYROOT, wenn der Name des Kunden in der ersten Stelle gleich Name und der Ort des Kunden größer oder gleich dem eingegebenen Ort ist.

Unqualifiziertes SSA.

```
     C           'MYROOT  'QSSA 'MYCUNR  'KDNR    5  EQ
     C           'MYCADR  'USSA
```

Bei einem unqualifizierten SSA-Aufruf wird in Faktor1 der Segment-Name wie beim qualifizierten SSA eingetragen. Die Operation heisst USSA. Alle anderen Einträge sind ungültig und führen zu einer Fehlermeldung.

DL/I-Aufruf über die DLI-Operation.

Der eigentliche Call zum DL/I wird über die Operation DLI erzeugt. Hierzu erst einmal ein Beispiel :

```
     C           GU        DLI            MYROOT  2  10
```

Im Faktor1 wird der DL/I-Call angegeben. Als Operation muss DLI eingetragen werden. Faktor2 bleibt normalerweise frei (kann u.U. einen von der F-Karte unterschiedlichen Psbnamen enthalten).Im Ergebnisfeld wird der Name des Ein-/Ausgabebereiches definiert. In diesem Fall der Name der Datenstruktur in der Eingabe. Die '2' im Feldlängenfeld gibt die Nummer des verwendeten PCBs an. Der Schalter auf Bedingung größer wird gesetzt, wenn der DL/I-Return-Code ungleich ' ' , 'GA' oder 'GK' von DL/I gesetzt wird.

Es kann im Ergebnisfeld auch der Name eines Feldes definiert werden, das nur in den D- oder E- Bestimmungen definiert wurde, aber keine Datenstruktur darstellt. Zugriffe zum DL/I , die keine Daten übertragen, brauchen aus Kompatibilitätsgründen ebenfalls einen Eintrag im Ergebnisfeld, der dann beispielsweise ein Dummyfeld der Länge 1 sein kann. Dazu als Beispiel ein Termination-Call :

```
     C           TERM      DLI            DUMMY   1 10
```

Es wird auf die im Zugriff befindliche DB ein Termination-Call durchgeführt.

Beachte:

Beim Termination-Call wird vom DLI ein Syncpoint gesetzt. Daraus folgt für die Programmierung im CPG, dass wie bei der CPG-Operation SYNCP beschrieben, vor einem Termination-Call alle im Programm verwendeten Dateien mit RNDOM freigegeben werden müssen.

Abfrage Return-Code und KFBA.

```
     ICPGSMN  F
     I                                       1   8 SEGNAM
     I                                       9  10 SEGLEV
     I                                   B  11  140KFBALE
     I                                      15  18 SENSGS
     I                                      19 146 KFBA
     C           CPGDRC    COMP '  '                1313
     C                     SELCT          CPGSMN
```

Die Key-Feedback-Area wird bei jedem DL/I-Aufruf mit den aktuellen Werten gefüllt. Der Aufbau dieser Area wird oben beschrieben. Ebenfalls gefüllt wird das interne Feld CPGDRC, in dem linksbündig in 2 Stellen der DL/I-Return-Code gesetzt wird.

Zugriff zu DL/I über datenbankunabhängige Programmierung.

Eine andere Möglichkeit der Verarbeitung wird über die Datenbankunabhängige Programmierung gegeben. Im CPG ist ein Standardzugriff auf ein Dataset 'DLI' standardmäßig enthalten und im folgenden beschrieben. Ein Beispiel in Ihrer CPG-Source Library zeigt einen Vergleich zwischen beiden Methoden. (Siehe A.CPG*DLD)

Eintragung in der H-Karte

Spalte 48 der H-Karte muss eine Eintragung enthalten, die die jeweilige Adressierungs-Routine des zugehörigen TP-Monitors aufruft. Dies ist normalerweise 'D' für CICS und 'E' für Shadow. Falls gewünscht, kann dieser Wert in die Standard H-Karte übernommen werden.

Dateizuordnung

Die Dateizuordnung muss eine Datei mit dem Namen 'DLI' enthalten.
Die Satzlänge sollte der Länge des größten Segments entsprechen (oder einer Gruppe von Segmenten für einen Pfad-Aufruf).
Die Schlüssellänge ist 8 (Spalte 29 bis 30). Die Einheit (Spalte 40 bis 46) ist Dataset.

Erweiterte Dateizuordnung

Die erweiterte Dateizuordnung muss die Definitionen für die folgenden sechs Felder enthalten: DLFUNC, SSA1, SSA2, SSA3, SSA4 und SSA5.

DLFUNC ist ein 16 Byte großer Speicherbereich zur Übertragung von Informationen zwischen CPG und DL/I.

Die Bytes 1 bis 8 dieses Bereichs sind reserviert für die interne Steuerung durch CPG.

Bytes 9 bis 12 enthalten die CALL-Funktion des DL/I. Bei der Rückkehr von einer DL/I-Anfrage enthalten die ersten beiden Bytes den Returncode von DL/I, die rechten beiden Bytes Blank.

Bytes 13 bis 14 dieses Bereichs enthalten in einem gepackten numerischen Feld die Nummer des gewünschten PCBs in der PCB-Liste.

Bytes 15 bis 16 enthalten in einem gepackten numerischen Feld die Anzahl der im DL/I CALL benutzten SSAs. Eine typische Eintragung ist die folgende:

```
E                    DLFUNC      0 16
E                    RES            8
E                    CALL           4
E                    PCBNUM         3 0
E                    SSANUM         3 0
E                    SSA1        0 80
E                    SEGNAM         8
E                    SUCH          72
E                    SSA2        0  1
E                    SSA3        0  1
E                    SSA4        0  1
E                    SSA5        0  1
E                    DUMMY          1
```

Die Felder SSA1 bis SSA5 sind Arbeitsbereiche, die DL/I-Segmentsuchparameter enthalten. Die Standardversion des CPG-DL/I-Interface lässt bis zu fünf SSAs zu, kann jedoch vom Benutzer erweitert werden. Alle fünf SSAs müssen definiert werden, auch wenn sie nicht benutzt werden.

Wenn nicht alle SSAs benutzt werden, sollten die restlichen mit '0' in den Spalten 36 bis 39 und einer Länge von 1 definiert werden. Die Länge und das Format jeder SSA ist anwendungsabhängig.

Eingabebestimmungen

Wenn Daten aus einer DL/I-Datenbank gelesen werden sollen, müssen diese in den Eingabebestimmungen für eine Datei 'DLI' beschrieben werden. Bei der Rückkehr von einem 'GET CALL' gibt das Interface das oder die Segmente zum CPG-Programm zusammen mit entsprechenden Standard-Informationen des PCB zurück. Der Bereich, in dem die Daten abgestellt werden, enthält die Daten und Informationen in folgender Reihenfolge:

```
Stelle   1 -  8   PCB Name
         9 - 10   Segment Hierarchie Level Indicator
        11 - 18   Segment-Name
        19 - 22   Länge der Key-Feedback-Area
        23 - 26   Anzahl der Segmente
        27 -150   Key-Feedback-Area
```

```
151 -154   Nicht benutzt
155 -  n   Segment Inhalte des DL/I Calls
```

Ein Beispiel zeigt die Eingabebestimmungen für mehrfache Segmentverarbeitung. Die Stellen 11 bis 18 der Eingabe, welche den Segmentnamen enthalten, werden geprüft. Ist die Prüfung positiv, so wird der zugehörige Schalter gesetzt und die Eingabe verarbeitet.

```
IDLI     AA  12  11 CA  12 CR  13 CT
I        AND     14 CT  15 CX  16 CT
I                                     155 162 ATNR
I                                     163 222 TEXT
I        AA  13  11 CA  12 CR  13 CT
I        AND     14 CV  15 CF  16 CB
I                                     155 162 VERFGB
I                                   P 240 2440GKAUFT
I                                   P 245 2480ÜBRIG
I        AA
I                                     155 162 SEGKEY
I                                      11  16 SEGNAM
```

- Wenn ein 'ARTTXT' (Spalten 21 bis 41) Segment gelesen wird, wird der Schalter 12 gesetzt und die Felder 'ATNR' und 'TEXT' werden gefüllt.

- Wenn ein 'ARTVFB' Segment gelesen wird, wird der Schalter 13 gesetzt und die Felder 'VERFGB', 'GKAUFT' und 'ÜBRIG' werden gefüllt.

- Die Felder der letzten Eingabebestimmung werden immer gefüllt, unabhängig von der Segmentart.

Rechenbestimmungen

DL/I-Calls werden in den Rechenbestimmungen durch 'CHAIN' oder 'EXCPT' Anweisungen für das DL/I-Dataset ausgeführt. Vor Ausführung einer solchen Operation muss der Programmierer sicherstellen, dass der Bereich 'DLFUNC' folgende Werte enthält:

1. Die gewünschte DL/I-CALL Funktion.

2. Die Nummer des gewünschten PCB in der PCB-Liste

3. Die Anzahl der für diesen CALL benutzten SSAs. Wenn keine SSAs benutzt werden, muss Null eingetragen werden.

Wenn SSAs benutzt werden, müssen diese vor der CHAIN- oder EXCPT-Operation gefüllt werden.

Mit der CHAIN-Operation kann jede DL/I-Call-Funktion ausgeführt werden. Faktor 1 muss den Namen eines 8-Byte Feldes enthalten, in welchem der Name des benutzten PCB steht. Die CHAIN-Operation erfordert in den Spalten 54-55 eine Bezugszahl, auch wenn sie hier nicht benutzt wird. Bei der Rückkehr aus einer CHAIN-Operation werden die Eingabebestimmungen wie üblich verarbeitet. Der DL/I-Return-Code wird im DLFUNC Bereich gespeichert, und zwar in den linken 2 Stellen vom Feld CALL.

Die EXCPT-Operation wird benutzt, um 'REPL' oder 'ISRT' DL/I-Funktionen zu verarbeiten.

Ausgabebestimmungen

Die Ausgabebestimmungen werden benutzt, um Daten in der DL/I-Datenbank zu erzeugen oder zu verändern. Segment-Daten müssen ab Stelle 155 des jeweiligen Bereichs gespeichert werden. Nach Übertragung der Daten in den Ausgabebereich setzt CPG den im DLFUNC-Bereich angegebenen Call ab. Der PSB übernimmt dabei die Funktion des Schlüssels, das heisst, der bei der letzten CHAIN-Operation benutzte PSB gilt auch für die Ausgabe. Wenn keine CHAIN-Operation durchgeführt wurde, muss der Programmierer den Namen des gewünschten PSB in das interne CPG-Schlüsselfeld des DL/I-Datasets bringen. Dieses Feld hat den Namen CPGKNN, wobei NN die laufende Nummer der Dateizuordnung ist,in der das DL/I-Dataset definiert wurde. Ist z.B. 'DLI' die zweite Datei in der Dateizuordnung, so heisst das Schlüsselfeld 'CPGK02'.

Fehlerbehandlung

Logische Fehler werden im Interface erkannt und führen zu einem Programm-Abbruch. Eine entsprechende Fehlermeldung wird über den TP-Monitor auf den Bildschirm oder die System-Konsole ausgegeben.

Beachte: Termination-Call

Beim Termination-Call wird vom DLI ein Syncpoint gesetzt. Daraus folgt für die Programmierung im CPG, dass wie bei der CPG-Operation SYNCP beschrieben, vor einem Termination-Call alle im Programm verwendeten Dateien mit RANDOM freigegeben werden müssen.

Variable Einträge bei DL/I-Operationen 2336

Vom Release 1.3 an können die Einträge der DL/I-Operationen variabel gehandhabt werden.

Dazu muss im Programm eine 40-stellige Datenstruktur mit Namen CPGDLV zur Verfügung gestellt werden, deren einzelne Subfelder die alternativen Werte enthalten. Sind zur Ausführungszeit des Befehls die Subfelder der Datenstruktur ungleich blank bzw. Null, so wird ihr Wert statt des Wertes im DLI-,QSSA oder USSA-Statement angenommen.

Für diese Verarbeitungsart muss in Spalte 53 der Operation DLI, QSSA oder USSA ein V für 'variabel' eingetragen werden.

Die Datenstruktur muss folgenden Aufbau haben :

```
ICPGDLV      DS
I                                        1   8 SEGNAM  Segmentname
I                                        9  16 SENKEY  sensit.Feld
I                                    P  17  180KEYLEN  KEY-Länge
I                                       19  20 COMPAR  Operator
I                                       21  24 EXTENS  Erweiterung
I                                       25  25 ANDOR   AND / OR
I                                       27  30 FUNCT   DLI-Call
I                                       31  38 PSBNAM  PSB-Name
I                                    P  39  400PCBNUM  PCB-Nummer
```

Beispiel: Der Befehl

```
C           'ARTIROOT'QSSA 'ARKO1   'KEY     7  GE
```

kann somit auch wie folgt dargestellt werden:

```
C                     MOVE 'ARTIROOT'SEGNAM
C                     MOVE 'ARKO1  ' SENKEY
C                     MOVE 'GE'      COMPAR
C                     Z-ADD7         KEYLEN
C           'XXX     'QSSA 'YYY     'KEY     1 VEQ
```

Sind Subfelder der Datenstruktur CPGDLV nicht gefüllt, so wird der Wert des QSSA-Statements angenommen.

Datenview-Verarbeitung 2340

Datenview-Definition 2341

Eine Datenview ist eine Zusammenstellung von Daten für eine bestimmte Aufgabenstellung.

Eine View kann beliebig aus Feldern und Sätzen verschiedener Dateien zusammengesetzt sein. Die Logik einer View unterscheidet sich von anderen Speicherungsformen dadurch, dass grundsätzlich jedes ihrer Elemente als Schlüsselfeld dienen kann.

Realisierung einer Datenview 2342

In relationalen Datenbanken wird eine Datenview online erstellt und bleibt für die Zeit ihrer Verarbeitung im Hauptspeicher.

Um diese Verarbeitungsform zu simulieren, kann folgender Weg gewählt werden: Mit einem CPG- oder CPG3..Query-Programm wird eine Datenview erstellt und auf der Datei CPGWKV abgestellt. Die genaue Vorgehensweise ist ab Abschnitt 7530 beschrieben.

Die View wird erst dann aus der Datei in den Hauptspeicher geladen, wenn sie in einem Programm angesprochen wird. Die geladene View bleibt dann bis zum folgenden Shut-Down des TP-Monitors im Hauptspeicher.

Die erstellte View hat die Form einer Tabelle mit der Besonderheit, dass jede Spalte der Tabelle als Schlüsselbegriff dienen kann.

Verarbeitung von Datenviews 2343

Zur Verarbeitung von Datenviews mit CPG muss beachtet werden:

- Für die View muss eine FILE-Bestimmung angelegt werden. Als Einheit muss TABLE angegeben werden, als Satzlänge die Summe der Felder der Tabelle und als Schlüssellänge das Maximum der Längen der möglichen Schlüsselfelder.

- Eine Datenview wird ähnlich wie eine Datei verarbeitet; alle Felder, die aus der Tabelle eingelesen werden sollen (insbesondere alle Schlüsselfelder), müssen in den Eingabebestimmungen unter dem Namen der Tabelle eingelesen werden.

- Gelesen wird die View mit dem Befehl FIND. Faktor 1 enthält das Schlüsselelement, Faktor 2 den Namen der View (bis zu vier Stellen). Eine Bezugszahl für den Vergleich auf Gleichheit zeigt an, ob das Suchargument aus Faktor 1 in der View gefunden wurde.

- Soll während der Verarbeitung wieder am Anfang der Tabelle aufgesetzt werden, so wird die sequentielle Verarbeitung der View mit der Operation RNDOM beendet. Wurde ein mit FIND gesuchtes Element nicht gefunden, wird automatisch am Tabellenanfang wieder aufgesetzt. Zudem wird in diesem Fall der Schalter EF gesetzt.

Beispiel:

Eine View wird aus Daten der Artikel- und Kundenstammdatei erstellt.

Beide Dateien haben nur jeweils einen Schlüssel: Die Artikelnummer bzw. Kundennummer. Diese Schlüssel werden im folgenden als 'primary key(s)' bezeichnet.

Mit der erstellten View können die Daten nicht nur nach den primary keys, sondern auch nach beliebig vielen anderen, im folgenden als 'secundary keys' bezeichneten Schlüsseln durchsucht werden. Beispiele für solche secundary keys sind Postleitzahl, erste Stelle der Postleitzahl oder Vertreternummer.

Beim Erstellen der View ist nun abzuwägen, wie viele 'Spalten' die generierte Tabelle haben soll. Insbesondere sollte zur sinnvollen Nutzung der View-Verarbeitung darauf geachtet werden, dass die View die primary keys der Dateien enthält, aus deren Daten sie sich zusammensetzt. Somit ist sichergestellt, dass nach einem erfolgreichen FIND auf sämtliche Daten (mit CHAIN) zugegriffen werden kann, die mit der View in Beziehung stehen. Vgl. auch Beispiel 33, Kapitel 8.

DC-Funktionen 2350

Hauptspeicher-Management 2351

DC-Programme bestehen im Gegensatz zu Batch-Programmen immer aus mehreren Teilen. Die erste Unterteilung erfolgt in einen Instruktionsteil und einem Datenteil, für die jeweils unterschiedliche Regeln gelten.

1) Instruktionen

Der Instruktionsteil ist so aufgebaut, dass er von mehreren Benutzern gleichzeitig verwendet werden kann. Dazu muss das Programm quasi-reentrant geschrieben sein, dass heisst zwischen zwei Macro-Instruktionen, die die Steuerung an den übergeordneten TP-Monitor übergeben,müssen die Programme so beschaffen sein, dass der Urzustand für den folgenden Benutzer wieder hergestellt wird, dass heisst es dürfen keinerlei Daten, Schalter oder Weichen im Programmtext abgespeichert werden. CPG-Programme erfüllen diese Anforderung.

Instruktionsteile von Programmen werden von fast allen bekannten TP-Monitoren beim ersten Aufruf in den Speicher geladen und bleiben dort bis zum Abbruch der Online-Verarbeitung (Shut Down) liegen, wenn nicht vom System festgestellt wird, dass der vorhandene Speicher nicht ausreicht (Short on Storage Bedingung).

2) Daten

Bei den Daten unterscheiden wir je nach Zuordnung verschiedene Klassen.

2A) Programmabhängige Daten

Alle programmabhängigen Daten werden in einer Transaction Work Area gespeichert, die beim Aufruf eines Programms einem Bildschirm zugeordnet wird und solange reserviert wird, bis das Programm entweder durch die Ausführung der letzten Rechenbestimmung oder durch Betätigung der Löschtaste der Bildschirmtastatur normal oder nach einem Fehler abnormal beendet wird.

Die TRANSACTION WORK AREA (TWA) wird dem Bildschirm temporär für die Dauer der Arbeit zugeordnet und kann damit mehrfach im Speicher sein, und zwar für jeden Bildschirm, der das Programm aufruft einmal.

2B) Ein- und Ausgabe-Daten

Bei der Ein- oder Ausgabe von Daten wird unmittelbar vor der Ausführung der Ein- oder Ausgabeoperation ein Hauptspeicher-Segment angefordert, in das bei Ausgabe die Daten aus der Transaction Work Area übertragen werden und aus dem nach Ausführung der Eingabe-Instruktion die Daten in die TWA übertragen werden. Danach wird dieser Bereich in der Regel sofort freigegeben, so dass Ein- und Ausgabebereiche den Hauptspeicher nur für extrem kurze Zeiten belasten.

Bei einigen Operationen müssen jedoch diese Bereiche bis zum Ende des Verarbeitungsvorgangs erhalten bleiben. Diese Ausnahmen sind:

Sequentielles Lesen von Platten	Der Bereich bleibt bis zum Ende der sequentiellen Verarbeitung (RNDOM oder Programmende) erhalten.
CHAIN für UPDATE	Der Bereich bleibt bis zum Zurückschreiben des Satzes erhalten bzw.bis zum RNDOM für diese Datei.
Lichtstiftunterbrechung	Der Bereich bleibt bis zur nächsten Ein-/Ausgabe-Operation für ein Terminal erhalten.

2C) Programmunabhängige Daten

Schließlich können CPG-Programme auch Daten untereinander austauschen. Die Daten müssen dazu in programmunabhängigen Bereichen zwischengespeichert werden. Die verschiedenen Verfahren hierzu sind im Abschnitt Zwischenspeicherung von Daten näher beschrieben.

SQL/DS 2360

CPG unterstützt SQL bzw. DB2 in der Produktstufe CPG3. Hierzu ist ein separates Installationsband erforderlich.

Die Programmierung der SQL-Befehle entspricht der normalen SQL-Syntax.

Im folgenden wird ein Miniprogramm aufgelistet,mit dem ein Zugriff auf die SQL-Tabelle KUNDEN durchgeführt wird. Es werden die Spalten KKDNR und KFIRMA gelesen und am Bildschirm angezeigt.

```
      H                             L   C    SQL TESTPROGRAMM   TST026
      FBILD
      DSQ      BEGIN DECLARE SECTION
      D        USER           8
      D        PASSW          8
      D        KDNR           5 0
      D        FIRMA         30
      DSQ      END DECLARE SECTION
      C                     MOVEL'SQLDBA  'USER
      C                     MOVEL'SQLDBAPW'PASSW
      CSQ CONNECT :USER IDENTIFIED BY :PASSW
      CSQ DECLARE C2 CURSOR FOR
      CSQ     SELECT KKDNR,KFIRMA
      CSQ     FROM KUNDENA
      CSQ     WHERE KKDNR  < 3000
      CSQ OPEN C2
      C           A100      TAG
      CSQ FETCH C2
      CSQ     INTO  :KDNR,:FIRMA
      C           SQCODE    COMP 0                    6060
      C                     EREADBILD
      C  NP1                GOTO A100
      CSQ CLOSE C2
      CSQ COMMIT WORK
      OBILD    E E
      O                                    250 'TEST SQL'
      O                          KDNR      340
      O                          FIRMA     440
      O                60        SQCODE    640
 /*
```

Programmierhilfen 2400

Entscheidungstabellen 2410

CPG ermöglicht die direkte Verarbeitung von Entscheidungstabellen in den Rechenbestimmungen.

Durch den Operationsschlüssel 'BEGDT' wird von der RPG-Logik auf Entscheidungstabellen-Logik umgeschaltet. Alle RPG-Operationen sind solange ungültig, bis die Entscheidungstabelle durch einen Befehl 'ENDDT' abgeschlossen wird.

Eine Entscheidungstabelle besteht im oberen Teil aus Bedingungen, im unteren Teil aus Aktionen. Alle Bedingungen einer Entscheidungstabelle müssen unmittelbar aufeinander folgen, alle Aktionen ebenfalls.

Der Operationscode '-----' bedeutet, dass hier die Bedingungen enden und die Aktionen beginnen.

Die Spalten 43 bis 74 enthalten die Verknüpfungsleiste. Mögliche Eintragungen sind im Bedingungsteil je verfügbare Spalte:

Y	für 'YES' oder Bedingung erfüllt,
N	für 'NO' oder Bedingung nicht erfüllt,
BLANK	für: Bedingung ist nicht ausschlaggebend.

Mögliche Eintragungen für Aktionen sind ebenfalls je verfügbare Spalte:

Blank	Aktion wird nicht ausgeführt,
Nicht Blank	Aktion wird dann ausgeführt, wenn alle senkrecht in der gleichen Spalte darüber stehenden Bedingungen erfüllt,bzw. je nach Eintragung (N) nicht erfüllt sind.

Siehe folgendes Beispiel.

Formularbeschreibung.

Allgemeine Eintragungen:

Spalte 6 Kartenart.

'C' muss eingetragen werden.

Spalte 7-17 Bleibt frei

Spalte 43-74 Verknüpfungsleiste (siehe oben).

Bedingungen.

Spalte 18-27 Faktor 1

Enthält einen gültigen Feldnamen, den Namen eines Feldgruppen-Elementes (FG,I) oder eine Konstante.

Spalte 33-42 Faktor 2

Enthält einen gültigen Feldnamen, den Namen eines Feldgruppen-Elementes (FG,I) oder eine Konstante.

Spalte 28-32 Operationsschlüssel.

Gültige Eintragungen sind:

>	oder 'HIGH'	größer als
		Faktor 1 ist größer als Faktor 2.
<	oder 'LOW'	kleiner als
		Faktor 1 ist kleiner als Faktor 2.
=	oder 'EQUAL'	gleich.
		Faktor 1 ist gleich Faktor 2.

ACHTUNG: Bei allen Vergleichsoperationen müssen Faktor 1 und Faktor 2 entweder beide numerisch oder beide alphanumerisch definiert sein.

Aktionen.

Spalte 18-27 Bleibt frei.

Spalte 28-32 Operationsschlüssel.

Gültige Eintragungen sind:

GOTO	Verzweigen nach.
EXSR	Ausführen Unterprogramm.
EXCPT	Ausgabe

Aus einer Entscheidungstabelle darf jedoch keine zweite Entscheidungstabelle mit EXSR aufgerufen werden.

Beispiel.

Eine Kunden-Datei wird sequentiell verarbeitet, dabei sollen auf einem Bildschirm alle Kunden angezeigt werden, deren Umsatz größer als 10000 Euro und deren Soll-Saldo größer als der Umsatz ist, oder deren Umsatz größer als 10000 Euro und deren Saldo größer als ihr Kreditlimit ist. Alle anderen Kunden werden nicht angezeigt.

```
C           WEITER    TAG
C                     READ BILD
C           LESEN     TAG
C                     READ KUNDEN

C           LABEL     BEGDT
C           UMSATZ    >    10.000    YY
C           SALDO     >    UMSATZ    Y
C           SALDO     >    KREDIT     Y
C           SALDO     >    10.000      Y
C                     -----
C                     EXSR PRUEF       X
C                     EXCPT           XXX
C                     EXCPTKREDIT      X
C                     GOTO WEITER     XXX
C                     GOTO LESEN         X
C                     ENDDT
```

Achtung:

Da die Spalte 43 bis 74 die Verknüpfungsleiste beinhaltet, darf keine andere Eintragung vorgenommen werden z. B. 8-stelliger Phasenname oder eine Nummerierung von Stelle 73-78.

Feldaufbereitung 2420

CPG bietet als zusätzlichen Service die Möglichkeit, Felder über die Ausgabebestimmungen aufzubereiten. CPG kann jedes beliebige Feld bis zu einer Länge von 256 Bytes wie einen Ausgabesatz aufbereiten. Der Operationsschlüssel in den Rechenbestimmungen für diese Operation lautet 'EDIT'. Das Ergebnisfeld enthält einen gültigen Feldnamen. In den Ausgabebestimmungen muss dieses Feld als Dateinamen spezifiziert werden. Spalte 15 enthält dabei ein 'F' wie Feldaufbereitung. Der Zeilenbestimmung folgen die Feldbestimmungen wie bei jeder anderen Datei. Felder ohne Aufbereitungsnamen werden grundsätzlich mit verarbeitet.

Beispiel:

Rechenbestimmungen:

```
C                     EDIT POSTEN    ZEILE
C                     ...
C                     EDIT SUMMEN    ZEILE
```

Ausgabebestimmungen:

```
OZEILE    F
O                         POSTEN
O                         KDNR       5
O                         RBETR J    20
OZEILE    F
O                         SUMMEN
O                                    6 'GESAMT'
O                         RBGES J    20
OZEILE    F
O                         NR    Z    79
```

Beispiel:

Rechenbestimmungen:

```
C                     EDIT           ZEILE  70
```

Ausgabebestimmungen:

```
OZEILE    F
O                           KDNR       7
O                           NAME      33
O                           ORT       55
O                           UMSATZ    70 ' .   . 0 ,  -'
```

Diese Möglichkeit erspart dem Programmierer vor allem bei der Bildschirmdatei die Mehrfachdefinition gleicher Datenfelder z.B. beim Blättern in einem Datenbestand.

Umgekehrt können mit der Operation 'SELCT' die einzelnen Felder aus dem aufbereiteten Feld wieder herausgelöst werden. Dies erfolgt mit Hilfe einer Rechenbestimmung mit dem Operationscode 'SELCT', deren Ergebnisfeld den Namen des aufbereiteten Feldes enthält. In den Eingabebestimmungen muss dieses Feld als Dateinamen spezifiziert werden. Spalte 15 der Eingabesatzbestimmung enthält dabei ein 'F' und Spalte 16 muss blank sein. Der Satzbestimmung folgen die gewünschten Feldbestimmungen wie bei jeder anderen Datei.

Beispiel:

Eingabebestimmungen:

```
IZEILE   F
I                                        1   7 KDNR
I                                        8  33 NAME
I                                       38  55 ORT
```

Rechenbestimmungen:

```
C                     SELCT          ZEILE
```

Nach Ausführung der Operation enthält das Feld 'KDNR' die Bytes 1 bis 7, das Feld 'NAME' die Bytes 8 bis 33 und das Feld 'ORT' die Bytes 38 bis 55 des Feldes 'ZEILE'.

Copy Funktion 2430

CPG ermöglicht das Abspeichern von Programmteilen in der System-Source-Library. Die Regeln entsprechen dabei denen der gleichnamigen RPG-Autoreport-Funktion.

Die Eintragung /COPY R.XBOOK (auch A,XBOOK) ab Spalte 7 eines beliebigen CPG-Statements fügt dabei den unter dem Namen 'XBOOK' in der Source-Library 'R' katalogisierten Programmteil ins CPG-Programm ein. Die eingefügten Statements werden in der Listausgabe in Spalte 7 mit 'C' gekennzeichnet.

Die Copy-Funktion erfordert einen Vorlauf mit der Phase CPGPP (CPG-Preprocessor) und das Anlegen einer zusätzlichen Procedure 'CPG'. Der Ablauf ist in (6200) näher beschrieben.

Abweichend vom RPG werden bei CPG-Copy Books nur diejenigen Eingabebestimmungen aus einem Copy-Book ins Programm übernommen, die auch tatsächlich benutzt werden.

Ein Copy Book darf immer nur eine CPG-Kartenart (z.B. Eingabebestimmungen) enthalten.

Diagrammausgabe 2440

CPG erlaubt die Ausgabe einer numerischen Feldgruppe (maximal 13 Stellen) auf einen Bildschirm oder Drucker als Balkendiagramm gemäß folgendem Beispiel.

Eine Feldgruppe mit dem Namen 'UMS' wird definiert mit 12 Feldern, je 5 Stellen und 0 Dezimalstellen. Die Feldgruppe soll bei Bezugszahl 15 auf den Bildschirm als Diagramm ausgegeben werden. Dazu sind folgende Ausgabebestimmungen erforderlich:

```
1   ODATEI   E        15
2   O                         UMS     DIAG
```

ZU 1: Spalte 6 enthält die Eintragung 'O' für Output.

Die Spalten 7 - 14 enthalten einen gültigen Dateinamen. Die Datei muss in der Dateizuordnung als Bildschirm oder Drucker-Datei definiert sein.
Die Spalten 23-31 können eine oder mehrere Bezugszahlen enthalten.

ZU 2: Spalte 6 enthält die Eintragung 'O' für Output.

Die Spalten 32 - 37 enthalten den Namen einer Feldgruppe. Die Feldgruppe muss in der erweiterten Dateizuordnung als numerische Feldgruppe definiert sein.

Die Zahl der Dezimalstellen muss Null sein. Gegebenenfalls muss der Programmierer eine entsprechende Anpassung vornehmen.
Die Spalten 40 bis 43 enthalten die Konstante 'DIAG'.

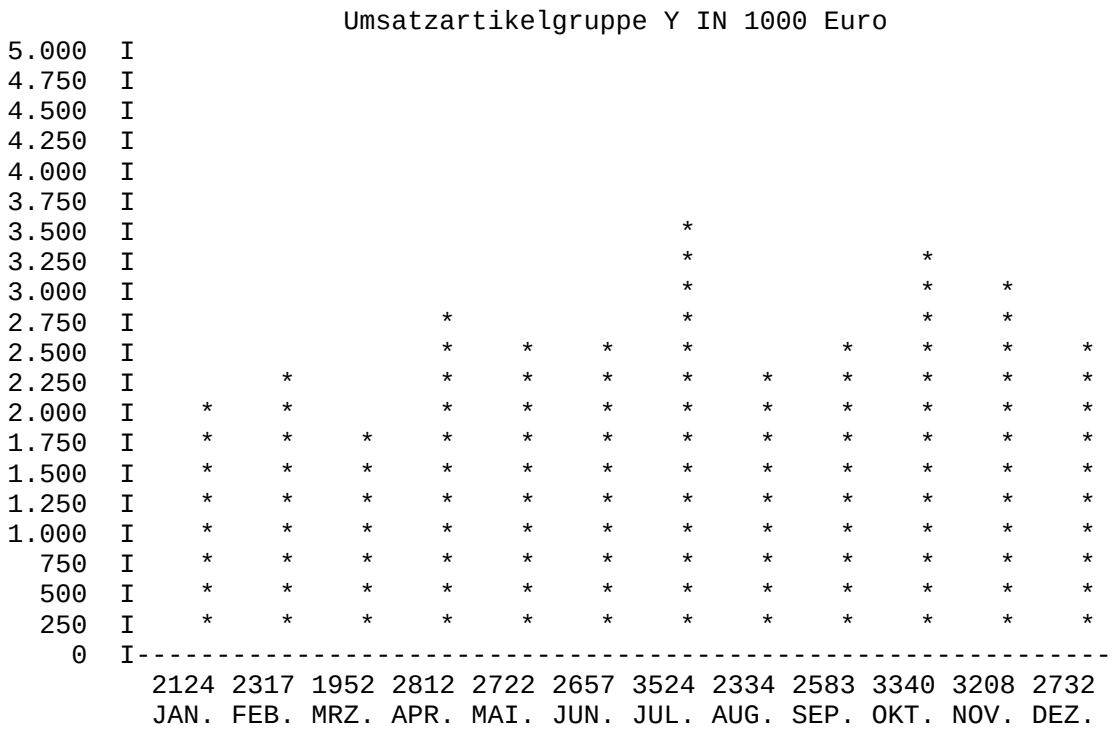
CPG versucht,auf Grund dieser Angaben ein Diagramm dieser Feldgruppe auf der spezifizierten Einheit auszugeben.
Benutzt werden hierzu die Zeilen 2 bis 23 des Bildschirms.

Ist die Gesamtstellenzahl der Feldgruppe größer als 69, so wird bei der Umwandlung die Meldung 'Feldgruppe zu groß' ausgegeben. In diesem Falle muss der Programmierer, z.B. mit Hilfe einer Division durch eine Zehnerpotenz die Feldgruppe entsprechend verkleinern.

Das Diagramm kann in Zeile 1 und 24 um beliebige Texte ergänzt werden. Zusatztexte in den Zeilen 2 bis 23 werden ignoriert, wenn sie vor der Feldgruppe spezifiziert werden, anderenfalls wird das Diagramm durch diese Texte überschrieben.

Eine Ziffer in Spalte 39 gibt an, wieviel senkrechte Sternzeilen jeder Balken des Diagramms breit ist. Bei fehlender Eintragung wird '1' eingesetzt.

In Spalte 45 bis 47 kann eine 1 Byte große Konstante, z.B. 'I', eingetragen werden. In diesem Falle werden im unten stehenden Beispiel alle Sterne durch diese Konstante (I) ersetzt.



Strukturierte Programmierung 2450

Es wurden folgende Operationen im CPG aufgenommen:

- BREAK (beenden einer DO-Schleife)
- CASxx (vergleiche und verzweige in Subroutine-Gruppe)
- CONT (unterbrechen einer DO-Schleife)
- DO (führe aus)
- DOUxx (führe aus..bis)
- DOWxx (führe aus..während)
- ELSE (sonst..führe aus)
- END (end)
- ENDDO (DOxxx Ende)
- ENDIF (IFxx Ende)
- IFxx (wenn..dann)

Die DO-Operationen erlauben eine einfache oder mehrfache Ausführung einer Gruppe von Rechenoperationen,beginnend mit dem Wert in Faktor 1, der bei jedem Durchlauf um den Wert, der in der zugehörigen END-Operation bestimmt ist, erhöht wird,bis der Grenzwert in Faktor 2 erreicht ist. Die Definition einer zugehörigen END-Operation ist weiter unten erklärt.
Im Ergebnisfeld kann ein numerisches Feld eingetragen werden, welches den laufenden Index der DO-Schleife enthält.

Die DOUxx- und DOWxx-Operationen erlauben eine einfache oder mehrfache Ausführung einer Gruppe von Rechenoperationen, basierend auf dem Ergebnis des Vergleichs von Faktor 1 und Faktor 2.

Eine IFxx- Operation bewirkt die Ausführung einer Gruppe von Rechenoperationen, basierend auf den Vergleichsergebnissen von Faktor 1 und Faktor 2.

Eine DO-, DOUxx-, DOWxx-, oder IFxx- Operation bilden zusammen mit einer END-Operation eine DO-Gruppe. Die END-Operation,bezeichnet als zugehöriges Ende der DO-Gruppe, beendet jede DO-Gruppe oder bewirkt die erneute Ausführung der DO-Gruppe. Eine IFxx- und eine IFxx- Operation mit einer Else-Operation muss ebenfalls mit einer END-Operation abgeschlossen werden. Die Kodierregeln für die DOUxx-, DOWxx- und IFxx-Operationen sind die gleichen wie bei den Vergleichsoperationen in diesem Kapitel.

xx In den DOUxx-, DOWxx- und IFxx-Operationen kann sein:

xx	Bedeutung
GT	Faktor 1 ist größer als Faktor 2
LT	Faktor 1 ist kleiner als Faktor 2
EQ	Faktor 1 ist gleich Faktor 2
NE	Faktor 1 ist ungleich Faktor 2
GE	Faktor 1 ist größer oder gleich Faktor 2
LE	Faktor 1 ist kleiner oder gleich Faktor 2

Enthält eine DO-Gruppe eine andere vollständige DO-Gruppe, werden beide zusammen als geschachtelte DO-Gruppe bezeichnet. DO-Gruppen können bis zu 40 Ebenen tief verschachtelt werden.

Der Programmierer ist dafür verantwortlich, dass nicht mehr als 999 IF- oder DO-Operationen im Programm verwendet werden.

Im Folgenden wird ein Beispiel einer 3-stufigen DO-Gruppe gezeigt:

```
. . . . . DO
.
.        ..... DO
.        .
.        ..... END
.
.    . . . . . IF
.    .
.    .   ..... DO
.    .   .
.    .   ..... END
.    .
.    . . . . . ELSE
.    .
.    .   ..... IF
.    .   .
.    .   ..... END
.    .
.    . . . . . ENDIF
.
. . . . . ENDDO
```

Folgendes ist bei der Programmierung von DO-Gruppen zu beachten:

. Jede geschachtelte DO-Gruppe muss vollständig innerhalb einer DO-Gruppe höherer Ebene enthalten sein.

. Jede DO-Gruppe muss beides enthalten, eine DO- oder IFxx-Operation und eine END-Operation.

. Eine Verzweigung in eine DO-Gruppe von außerhalb kann zu nicht erwünschten Ergebnissen führen.

BREAK-Operation 2455

Die Operation BREAK beendet die aktuelle DO-, DOU- oder DOW-Schleife und verzweigt hinter das zugehörige END-Statement.

CONT-Operation 2457

Die Operation CONT unterbricht die Verarbeitung der aktuellen DO-, DOU oder DOW-Schleife.

CONT verzweigt vor das END-Statement der Schleife und somit wieder zurück zur Schleifenbedingung. Die Statements zwischen CONT und END werden also nicht ausgeführt, die Schleife wird aber entsprechend der programmierten Schleifenbedingung weiterhin durchlaufen.

DO-Operation 2460

Die DO-Operation funktioniert in folgender Weise:

In der DO-Bestimmungszeile werden die bedingenden Anzeiger/Bezugszahlen abgefragt. Treffen die Bedingungen zu, dann wird die DO-Operation ausgeführt.

Treffen die Bedingungen nicht zu, verzweigt die Programmsteuerung zur nächsten ausführbaren Rechenbestimmung nach der zugeordneten END-Bestimmung.

Treffen die bedingenden Anzeiger/Bezugszahlen bei einer DO-Bestimmung zu, wird der Anfangswert (Faktor 1) in den Index (Ergebnisfeld) übertragen.

Der Index (Ergebnisfeld) wird mit dem Begrenzungswert (Faktor 2) verglichen. Wenn der Index größer ist als der Begrenzungswert, verzweigt die Programmsteuerung zu der Rechenbestimmung, die der zugeordneten END-Bestimmung unmittelbar folgt.

Ist der Index kleiner oder gleich dem Begrenzungswert (Faktor 2), werden die Rechenbestimmungen zwischen der DO-Bestimmung und der zugeordneten END-Bestimmung ausgeführt.

Die bedingenden Anzeiger/Bezugszahlen in einer zugeordneten END-Bestimmung werden geprüft. Treffen die Bedingungen nicht zu, geht die Programmsteuerung weiter an die Rechenbestimmung, die der END-Bestimmung unmittelbar folgt.

Wenn die bedingenden Anzeiger/Bezugszahlen bei einer END-Bestimmung zutreffen, wird der Erhöhungsfaktor (Faktor 2 der END-Bestimmung) zum Index (Ergebnisfeld der DO-Bestimmung) addiert und die Programmsteuerung verzweigt zur DO-Operation und vergleicht erneut den Begrenzungswert.

Zum Beenden einer DO-Schleife kann der Begrenzungswert geändert werden. Der Index ist nicht veränderbar. BREAK und CONT ermöglichen zusätzlich zur Schleifenbedingung eine Veränderung des Ablaufs.

DOU-Operation 2461

Die DOU-Operation funktioniert in folgender Weise:

Es erfolgt eine Prüfung auf bedingende Anzeiger/Bezugszahlen in der Zeile der DOU-Bestimmung.Ist die Prüfung der bedingenden Anzeiger/Bezugszahlen negativ, so verzweigt die Steuerung zu der nächsten ausführbaren Rechenoperation nach der zugeordneten END-Bestimmung. Ist die Prüfung der bedingenden Anzeiger/Bezugszahlen positiv, wird die DO-Gruppe ausgeführt. (Die Rechenbestimmungen zwischen der DOU- und der END-Anweisung werden ausgeführt).

Es erfolgt eine Prüfung der bedingenden Anzeiger/Bezugszahlen in der zugeordneten END-Bestimmung.
Wird aufgrund der Prüfung der bedingenden Anzeiger/Bezugszahlen die END-Bestimmung nicht ausgeführt, verzweigt die Steuerung zu der Rechenoperation, die der zugeordneten END-Bestimmung unmittelbar folgt.

Ergibt die Prüfung der bedingenden Anzeiger/Bezugszahlen in der zugeordneten END-Bestimmung, dass die END-Bestimmung ausgeführt wird, werden Faktor 1 und Faktor 2 der DOU-Operation miteinander verglichen, um zu entscheiden, ob die spezifizierte Beziehung besteht. Ist die Beziehung ungleich, wird die DO-Gruppe weiter ausgeführt.

Die DO-Gruppe wird solange ausgeführt, bis die festgelegte Beziehung erfüllt ist. Die Steuerung verzweigt danach zu der Rechenoperation, die der zugeordneten END-Bestimmung unmittelbar folgt.

DOW-Operation 2462

Die DOW-Operation funktioniert in folgender Weise:

Es erfolgt eine Prüfung auf bedingende Anzeiger/Bezugszahlen in der Zeile der DOW-Operation. Ist die Prüfung der bedingenden Anzeiger/Bezugszahlen negativ, verzweigt die Steuerung zu der nächsten ausführbaren Rechenoperation nach der zugeordneten END-Bestimmung. Ist die Prüfung der bedingenden Anzeiger/Bezugszahlen positiv, vergleicht die DOW-Operation Faktor 1 mit Faktor 2, um zu entscheiden, ob eine Beziehung besteht.

Ist die Beziehung ungleich, verzweigt die Steuerung zur nächsten ausführbaren Rechenoperation, die der zugeordneten END-Bestimmung unmittelbar folgt.
Ist die Beziehung erfüllt, wird die DO - Gruppe ausgeführt. (Die Rechenbestimmungen zwischen der DOW- und der END-Anweisung werden ausgeführt).

Ergibt die Prüfung der bedingenden Anzeiger/Bezugszahlen in der zugeordneten END - Bestimmung, dass die END - Bestimmung nicht ausgeführt wird, verzweigt die Steuerung zur nächsten ausführbaren Rechenoperation, die der zugeordneten END-Bestimmung folgt. Ergibt die Prüfung

der bedingenden Anzeiger/Bezugszahlen in der END-Bestimmung, dass die END-Bestimmung ausgeführt werden kann, verzweigt die Steuerung zurück zur DOW-Operation.

Die DO-Gruppe wird solange ausgeführt, bis die festgelegte Beziehung nicht mehr besteht. Die Steuerung verzweigt danach zu der Rechenoperation, die der zugeordneten END-Bestimmung unmittelbar folgt.

ELSE-Operation 2463

Eine ELSE-Operation gibt den Anfang derjenigen Rechenbestimmungen an, die angeführt werden, wenn die Prüfungen der zugehörigen IF-Operation nicht zutreffen.

Bedingende Anzeiger sind nicht erlaubt.

END-Operation 2464

Die END-Operation muss eingetragen werden, um eine DO, DOUxx, DOWxx oder eine IFxx-ELSE Gruppe abzuschließen.

In Faktor 2 kann ein Erhöhungswert > 0 als Konstante oder Variable eingetragen werden. Der Defaultwert ist 1.

IF-Operation 2465

Die IF-Operation funktioniert in folgender Weise:

Bedingende Anzeiger/Bezugszahlen können verwendet werden. Faktor 1 und Faktor 2 müssen entweder ein alphanumerisches oder ein numerisches Feld, einen Feldnamen oder ein Feldgruppenelement enthalten. Eintragungen in Faktor 1 und Faktor 2 müssen gleichen Typs sein: Entweder beide alphanumerisch oder beide numerisch.

Bei numerischem Vergleich sind beide Faktoren indizierbar, bei alphanumerischem Vergleich nur einer der beiden Faktoren.

In der IF-Anweisung werden die bedingenden Anzeiger abgefragt. Treffen die Bedingungen nicht zu, so wird im Programm hinter der entsprechenden END-Anweisung fortgesetzt. (Dies gilt auch, wenn eine ELSE-Anweisung vorhanden ist).

Besteht die Beziehung zwischen Faktor 1 und Faktor 2 nicht, verzweigt die Programmsteuerung zu der Recheninstruktion, die der zugehörigen END-bzw. ELSE-Anweisung unmittelbar folgt.

Bei einer END-Anweisung, die einer IF-Anweisung zugeordnet ist, müssen die Stellen für bedingende Anzeiger/Bezugszahlen leer sein.

Eine END-Bestimmung muss eingetragen werden, um eine IF-Operation abzuschließen. Folgt einer IF-Bestimmung eine ELSE-Bestimmung, muss die END-Anweisung nach der ELSE-Bestimmung und nicht nach der IF-Bestimmung eingetragen werden.

IF Condition : Bezugszahlenabfrage 2467

Die IF-Operation kann auch benutzt werden, um Bezugszahlen abzufragen.

Dabei bleiben Faktor 1 und Faktor 2 frei.

Die Verarbeitungslogik ist bei diesem Einsatz der Operation IF grundsätzlich anders. Es können bis zu drei Bedingungen angegeben werden, die logisch und-verknüpft sind. Sind alle Bedingungen zutreffend, wird der IF-Zweig durchgeführt; ist ein ELSE-Zweig vorhanden und mindestens eine Bedingung trifft nicht zu, so wird der ELSE-Zweig ausgeführt.

Beispiel: C BZ BZ BZ IF

Daten-Strukturen 2470

CPG erlaubt, einen Bereich im Speicher zu bestimmen sowie das Anlegen von Feldern - genannt Subfelder - innerhalb dieses Bereiches. Dieser Bereich im Speicher wird Datenstruktur genannt.Eine Datenstruktur kann benutzt werden, um

. denselben internen Bereich mehrfach unter Verwendung verschiedener Datenformate zu beschreiben,

. mit einem Feld zu rechnen und seine Inhalte zu verändern,

. ein Feld in Subfelder zu teilen, ohne MOVE oder MOVEL-Befehle zu benutzen,

. eine Datenstruktur und seine Subfelder auf die gleiche Art zu beschreiben, wie ein Satz definiert wird,

. nicht zusammenhängende Daten in zusammenhängende interne Speicherbereiche zu gruppieren.

Datenstrukturanweisungen werden in den Eingabebestimmungen beschrieben. Die Bestimmungen für Datenstrukturanweisungen sind:

Stelle 6	I.
Stelle 7 - 12	Name der definierten Datenstruktur.
Stelle 19 - 20	DS.
Stelle 48 - 51	Länge der Datenstruktur (wahlweise). Diese Eintragung muss rechtsbündig vorgenommen werden (z.Z. maximale Länge 999).

Folgende Regeln sind beim Spezifizieren von Datenstrukturanweisungen zu berücksichtigen:

. Der Datenstrukturname muss ein symbolischer Name mit maximal 6 Stellen sein. Der Name kann nur auf einer Datenstrukturspezifikation erscheinen und kann überall angesprochen werden, wo ein alphanumerisches Feld erlaubt ist.

. Alle Eintragungen für eine Datenstruktur und ihre Subfelder müssen

zusammen erscheinen; sie können nicht mit Eintragungen für andere Datenstrukturen gemischt werden.

. Die Länge einer Datenstruktur kann wie folgt sein:
  - Die Länge, angegeben in den Eingabefeldbestimmungen, wenn der Datenstrukturname ein Eingabefeldname ist.
  - Die höchste Bis-Stelle eines Subfeldes innerhalb einer Datenstruktur, wenn der Datenstrukturname kein Eingabefeld ist.
  - Die in den Stellen 48 bis 51 der Datenstrukturanweisung angegebene Länge.

. Die Länge der Datenstruktur wird bestimmt durch die erste Anweisung im Programm, die eine Länge der soeben beschriebenen Arten definiert. Nachfolgende anderslautende Längenangaben sind ungültig.

. Eine Datenstruktur und ein Subfeld einer Datenstruktur können nicht denselben Namen haben.

. Wird eine SELCT-Operation für eine Datenstruktur benutzt,so müssen die SELCT-Eingabe-Bestimmungen **vor** den Datenstruktur-Bestimmungen liegen.

. Ist ein Feld in einer Datenstruktur definiert, so darf es nicht in den E- oder D-Karten unterdefiniert werden. (Struktur innerhalb E- oder D-Karten).

Datenstruktursubfeld-Bestimmungen 2480

Die Subfelder einer programmbeschriebenen Datenstruktur müssen unmittelbar auf die Datenstrukturanweisung folgen, zu der sie gehören.
Die Bestimmungen für Subfelder sind folgende:

Stelle	6	I.
Stelle	43	P: besagt,dass das Subfeld in gepacktem Format dargestellt wird. (Muss bei numerischen Feldern eingetragen werden).
Stelle	44 - 51	Zwei 1-bis 4-stellige Zahlen. Die Stellen 44-47 enthalten die Anfangsposition,die Stellen 48-51 die Endpositionen des Subfeldes. Diese Eintragungen müssen rechtsbündig sein. Führende Nullen können ausgelassen werden.
Stelle	52	0-9: Gibt die Zahl der Dezimalstellen eines numerischen Feldes oder einer Feldgruppe an. Leer: Bedeutet, dass das Feld alphanumerisch ist.

Verwendung einer Datenstruktur, um Subfelder innerhalb eines Feldes zu definieren:

```
    IFILEIN  KF
    I                                    3  18 PARTNO
    I                                   19  29 NAME
    I                                   30  40 PATNO
```

```
I                                          41  61 DR
IPARTNO      DS
I                                           1   4 MFG
I                                           5  10 DRUG
I                                          11  13 STRNTH
I                                        P 14  160COUNT
```

Die Datenstruktursubfelder können über den Namen oder über die Subfelder PARTNO, MFG, DRUG, STRNTH oder COUNT angesprochen werden.

Verwendung einer Datenstruktur zur Zusammenstellung von Feldern:

```
IFILEIN  KF
I                                           3  10 PARTNO
I                                          11  160QTY
I                                          17  20 TYPE
I                                          21  21 CODE
I                                          22  25 LOCATN
IPRTKEY      DS
I                                           1   4 LOCATN
I                                           5  12 PARTNO
I                                          13  16 TYPE
```

Wird eine Datenstruktur benutzt, um Felder zu gruppieren, können Felder von nicht benachbarten Stellen auf dem Eingabesatz zusammenliegend gruppiert werden.Auf diesen Bereich kann dann durch den Datenstrukturnamen und/oder den individuellen Subfeldnamen zugegriffen werden.

Zur Zeit können maximal 500 Eintragungen in die Datenstrukturtabelle aufgenommen werden. Eine Erweiterung ist bei Bedarf möglich.
Siehe Copy CPG*CDTB. (* = Releasesuffix).

Optimierungsfunktionen 2500

Optimierung der TWA-Größe 2501

Standardmäßig wird vom CPG die TWA-Größe minimiert. Felder, die in der Input Division (implizit) definiert sind, werden nicht in die TWA übernommen, wenn sie weder in der Procedure Division noch in der Output Division nochmals angesprochen werden.

Die Optimierungsfunktion wird nicht aktiviert für Felder, die explizit in den D-Karten vereinbart wurden.

Die Optimierungsfunktion wird auch nicht wirksam bei Datenstruktur-Subfeldern. Die nicht optimierten Felder werden intern als 'used' gekennzeichnet; Datenstruktur-Subfelder sind also grundsätzlich 'used'.

Die Optimierung kann aufgehoben werden, indem in Spalte 16 der H-Karte ein 'D', 'S', 'Y' oder 'Z' eingetragen wird.

Methodenbank 2510

Die CPG-Standardversion fasst alle häufiger vorkommenden Routinen in einer internen Methodenbank zusammen. Diese Programmierweise reduziert den Hauptspeicherbedarf und entlastet damit die TP-Partition und somit die Gesamtanlage.

Die Methodenbank muss unmittelbar nach dem Laden des Systemsteuerprogramms als 1. Programm geladen werden. (9616) Zeigt ein Beispiel für dieses Ladeprogramm. Die Methodenbank muss bis zum Ende der Verarbeitung in der Partition verbleiben.

Den zentralen Teil der Methodenbank bildet das Steuerprogramm-Interface. Es enthält alle Ein- und Ausgabemakros des TP-Steuerprogramms.

Eine Methodenbank besteht auch für die Batchverarbeitung mit CPG- oder HL1-Batchprogrammen. Ein Teil der Batch-Methodenbank ist das Batch-Interface. Hier sind die Makros für die Verbindung mit dem Betriebssystem VSE oder z/OS zusammengefasst. CPG- oder HL1-Batchprogramme sind damit weitgehend unabhängig von dem verwendeten Betriebssystem.

Einzelheiten über die Methodenbank siehe Abschnitt Installation.

Systemunabhängige Programmierung 2520

Das Steuerprogramm-Interface bewirkt eine exakte Trennung zwischen dem Anwendungsprogramm und dem TP-Steuerprogramm. Das Anwendungsprogramm enthält kein Ein- oder Ausgabe-Makro mehr und ist damit unabhängig von jedem Release-Wechsel des Betriebssystems oder TP-Steuerprogramms, d. h. CPG-Programme brauchen bei einem Release-Wechsel in der Regel nicht umgewandelt zu werden. Umgewandelt wird nur das Steuerprogramm-Interface.

Datenbankunabhängige Programmierung 2530

CPG bietet in der Ausbaustufe CPG3 den Service, 'Dateien' zu verarbeiten, die in der angesprochenen Form physisch auf der Platte nicht vorhanden sein müssen. Zwischen dem codierten Zugriff im Anwendungsprogramm und dem physischen Zugriff kann ein CPG-geschriebenes Modul zum Ablauf kommen. In solchen Modulen (Datasets) können die Dateizugriffe vollständig beschrieben sein, so dass die Anwendungsprogramme von Dateizugriffen völlig frei sind.

Bei Änderungen des Dateizugriffs, zum Beispiel von VSAM-KSDS mit fester Satzlänge auf variable Satzlänge müssen dann nicht sämtliche betroffene Anwendungsprogramme geändert und umgewandelt werden, sondern nur das eine Dataset-Modul, das den Zugriff auf die Datei enthält.

Übertragungs-Optimierung 2540

Bei Bildschirmen, die über eine Übertragungsleitung (Remote) an das System angeschlossen werden, führt CPG falls gewünscht eine Optimierung der zu übertragenden Nachricht in der Form durch, dass eine Folge von mehr als drei gleichen Zeichen (z.B. Blanks) mit einem Command Repeat to Address übertragen wird. Da diese Optimierung andere kleinere Systeme durch zusätzliche CPU-Zeit belasten kann, wird die Entscheidung für den Einsatz dieses Features dem Benutzer überlassen.

Die Optimierung kann bei der CPG-Installation (siehe CPGURSIT) aktiviert werden.

Regeln bei transaktionsorientierter Programmierung 2550

. Die Lösch-Taste oder eine andere Programmfunktion für das Programmende muss vom Programmierer abgefragt werden.

. Für den transaktionsorientierten Programmaufruf steht in erster Linie die Operation EXITT zur Verfügung.

. Die Bildschirmdaten werden mit der Operation MAP ins Programm übertragen.

. Die Daten der TWA (Transaction Work Area) stehen nach dem Task-Ende nicht mehr zur Verfügung. Die für den Programmablauf relevanten Daten müssen zwischengespeichert werden (z.B. in einer Temporary Storage Queue) und beim Start der nächsten Task wieder in das Programm eingelesen werden. Es ist zu beachten, ob diese Zwischenspeicherbereiche zum Ende der Anwendung gelöscht werden sollen.

  Am Taskende werden die VSAM-Strings freigegeben. Das ist zum Beispiel zu beachten, wenn eine VSAM-Datei sequentiell gelesen wird.

In alten CPG-Programmen, die noch nicht mit QSF-Masken arbeiten, ist folgendes zusätzlich zu beachten:

. Es darf kein READ Terminal in den Rechenbestimmungen definiert werden.

  Die Eingabeübertragung vom Bildschirm erfolgt normalerweise automatisch, sie kann jedoch auch mit der Operation READI gezielt durchgeführt werden.

. Wenn beim Programmende die Tastatur entriegelt werden soll, so ist noch ein EXCPT auf den Bildschirm durchzuführen z. B.

```
OBILD    E E    01
O                          AC2480
```

. Es darf nur eine Bildschirmdatei im Programm definiert werden.

QSF - Quick Screen Facility 2570

QSF ist als Programmpaket im Service Level CPG2 enthalten. Es bietet die Möglichkeit, Bildschirm-Maps interaktiv zu erstellen und zu warten. QSF greift dabei auf eine Schnittstelle zu, die es dem Benutzer erlaubt, alle im Programm definierten Felder oder eine beliebige Auswahl daraus auf dem Bildschirm nach seinen Erfordernissen anzuordnen.

Aufbau und Veränderung der Maps erfolgt dabei interaktiv und setzt keine Programmumwandlung voraus, auch dann nicht, wenn sonstige im Programm definierte Felder in die Map neu aufgenommen werden.
Farben, Texte,Feldeigenschaften und Positionen können durch einfaches Positionieren des Cursors und entsprechende Eintragungen über das QSF verändert werden.

Das Programm enthält weder Ein- noch Ausgabe-Bestimmungen für den Bildschirm. CPG stellt dem QSF die erforderliche Schnittstelle über die Operationen MAP, MAPD, MAPI, MAPO und MAPP zur Verfügung.

QSF ist beschrieben im Handbuch CPG2-Serviceprogramme.

QLF - Quick List Facility 2580

QLF ist als Programmpaket im Service Level CPG4 enthalten. Es bietet die Möglichkeit, Listbilder interaktiv zu erstellen und zu warten.

QLF greift dabei auf eine Schnittstelle zu, die es dem Benutzer erlaubt, alle im Programm definierten Felder oder eine beliebige Auswahl daraus nach seinen Erfordernissen zu einem Listbild anzuordnen.

Aufbau und Veränderung der Listen erfolgt dabei interaktiv und setzt keine Programmumwandlung voraus, auch dann nicht,wenn sonstige im Programm definierte Felder neu in das Listbild aufgenommen werden.

Das Programm enthält keine Ausgabebestimmungen mehr für den Drucker.

CPG stellt dem QLF die erforderliche Schnittstelle über die Operation LIST für die Online- und Batchprogrammierung zur Verfügung.

Voraussetzung für QLF ist die Installation des Textverarbeitungssystems QTF.

QLF ist beschrieben im Handbuch QTF.

Produktivitätssteigernde Werkzeuge in CPG2 und CPG3 2700

Die Programmiersprache 'CPG' liefert das Grundwerkzeug zur Erzielung der größtmöglichen Produktivität bei der Entwicklung von Bildschirmanwendungen.

Darüber hinaus kann der Benutzer die Produktivität noch weiter steigern durch Einsatz der Produkte CPG2 bis CPG5.

CPG2 stellt zusätzlich zur nicht spaltengebundenen Syntax folgende Programmierwerkzeuge und -hilfsmittel zur Verfügung:

Entwicklungssystem der 4. Generation

QSF	-	Screen Designer zur programmexternen und interaktiven Beschreibung der Ein- und Ausgabe des Bildschirms.
QFF	-	schnelle Dateianzeige in selbsterstelltem Bild ohne Programmierung
Data Dictionary	-	für Datei- und Feldbeschreibungen

Werkzeuge für den Programmierer

Temporary Storage	-	zum Anzeigen und Löschen von TS-Bereichen
Transient Data	-	zum Handling des TD-Zwischenspeichers
CPGWRK	-	zur Verarbeitung der Arbeitsdatei CPGWRK
QDF	-	interaktive Testhilfe für Programmierer
Dump Service	-	Hauptspeicherauszug am Bildschirm
Manual	-	Online-Anzeigeprogramme der Handbücher

CICS- und System-Werkzeuge

Indicator Service	-	programmexternes Setzen und Löschen der T- und C-Schalter
PCT/PPT	-	Anzeigeprogramme für CICS-Tabellen
Print	-	Optimierung der Online-Druckausgaben

Für die freie Schreibweise steht das CPG2-Handbuch zur Verfügung, für die Programmierwerkzeuge das Handbuch 'CPG2..Serviceprogramme'.

CPG3 stellt für verschiedene Funktionen folgende Werkzeuge zusätzlich zu den Serviceprogrammen des CPG2 zur Verfügung:

Entwicklungssystem der 4. Generation

Query	-	Auswertungen für Endbenutzer
Chart	-	Graphische Darstellung von Zahlengruppen

Werkzeuge für den Programmierer

Ditto	-	Online-BDateien anzeigen und ändern
Hardcopy	-	Systemkonsole am Bildschirm anzeigen
Newcop	-	Aktivierung neu compilierter Programme
QTS	-	komfortable Erstellung von Datenviews

Bürokommunikation

Mail	-	Übermittlung von Nachrichten
Message	-	Übertragung von Nachrichten
Task	-	Zeitabhängiger Aufruf von Transaktionen
Zeit Information	-	über Datum, Wochentag, Kalendertag etc.
	-	Textverarbeitung siehe Zusatzprodukt QTF

Datenschutz

Sign on	-	Zugriffskontrolle für Bildschirmanwendungen

CICS- und System-Werkzeuge

CICS-Commands	-	erfassen und ausführen
PCT/PPT	-	Hilfsmittel zur Pflege der CICS-Tabellen
FCT	-	Hilfsmittel zur Pflege der CICS-Tabelle
File	-	aktuelle CICS FCT anzeigen
File Check	-	Dateiprüfung ob eröffnet (open)
Power Reader Service	-	Übertr. Jobs in die Power Reader Queue
Response	-	Ermittlung der Antwortzeiten
Storage	-	Anzeige der Speicherbelegung durch CICS

CPG3 ermöglicht darüberhinaus die Batchprogrammierung und die Programmierung in Bausteinen mit HL1.

Für die Serviceprogramme steht das Handbuch CPG3-Serviceprogramme zur Verfügung.

Testhilfen 2800

DEBUG-Operation 2810

Mit dem Operationsschlüssel 'DEBUG' kann der Programmierer an beliebiger Stelle des Programms ein 'Testbild' erzeugen, das alle gesetzten Bezugszahlen von 01 bis 99 seines Programms an dieser Stelle anzeigt.

Die Handhabung erläutert das folgende Beispiel.

Beispiel Testhilfe (Auszug).

```
127    03 110 C                    DEBUG
128    03 120 C   P5               SETON                    30

134    03 180 C                    SETON                    111213
135    03 190 C                    SETON                    1415
136    03 200 C                    SETON                    202122
137    03 210 C 30                 DEBUG
138    03 220 C           X        COMP Y                   514950
139    03 230 C 30                 DEBUG
140    03 240 C 50                 GOTO GL
141    03 250 C 49                 GOTO KL
142    03 260 C 51                 GOTO GR

178    04 150 C           GL       TAG
179    04 160 C   30               DEBUG

194    05 050 C           KL       TAG
195    05 060 C 30                 DEBUG
```

Dieses Beispiel bewirkt, dass das Programm bei Statement 127 unbedingt angehalten wird. Der Programmierer kann hier entscheiden, ob das weitere Programm ohne Unterbrechung ablaufen soll, oder ob der Ablauf auf dem Bildschirm durch Testbilder dokumentiert werden soll.

Jede Taste ungleich PF5 lässt dass weitere Programm normal ablaufen, bei Betätigung der Taste PF5 wird die Bezugszahl 30 gesetzt. Diese Bezugszahl bewirkt, dass bei allen folgenden DEBUGs Testbilder angezeigt werden und zwar in der Reihenfolge des Programmablaufs.

Achtung: Die DEBUG-Operation wird voll durch die Operation SDUMP abgedeckt. Bei Command-Level Programmen wird SDUMP statt DEBUG ausgeführt. (siehe folgende Seiten).

Beispiel: Das erste Testbild wird ausgegeben bei Statement 127.
Funktionstaste PF5 wird betätigt.
2. Testbild bei Statement 137.
3. Testbild bei Statement 139.
X ist kleiner als Y.
4. Testbild bei Statement 195.

Das 4. Testbild hat folgendes Format:

```
**********************************************************************
                            T E S T - H I L F E
**********************************************************************
BEZUGSZAHLEN EIN
11 12 13 14 15 20 21 22 30 49


**********************************************************************
STATEMENT-NR  195
```

Special Terminal Dump 2830

Die Operation 'SDUMP' beinhaltet die Operation 'DEBUG' und kann von beliebiger Stelle des zu testenden Programms aufgerufen werden.

In der zweiten Informationszeile wird unterschiedlich zur DEBUG-Operation die Programmadresse, der Sdump-Code und die zuletzt betätigte Funktionstaste angezeigt.

Als erstes erscheint folgende Maske:

```
**********************************************************************
                          T E S T - H I L F E
**********************************************************************
BEZUGSZAHLEN EIN
70


**********************************************************************
PROG.-ADRESSE 00029A            CODE ABCD          FUNKTIONSTASTE DE
**********************************************************************
```

Nach Betätigung der Datenfreigabetaste erscheint ein Terminaldump mit folgendem Format:

```
Programm = TST005   TCA                                 03.04.78  15.28UHR
E         F         0         1          3         4         5
002DBAA0  0023F180  0000001C  00      0A 502DA80A  502DA80A  502DB80A
502DC80A  502DD80A  802DBBE6  00      00 002D8B08  0023F080  001F3E40
6         7         8         9          B         C         D

00000000  0023F000  00000000         000  0023F080  ..0.....  ........
00000010  00000000  002082C0         000  0023F090  ........  ....d...
00000020  701F6A46  0023F18         BAA0  0023F0A0  ......1.  ........
00000030  502D9B1E  502D980          6952  0023F0B0  ........  ........
00000040  502DC80A  502DD8         2F720  0023F0C0  ..H...0.  ...U..7.
00000050  011EBCA8  001EA2         23300  0023F0D0  ........  ........
00000060  402D9D72  0023F         2DBAA0  0023F0E0  ......1.  ........
00000070  402D9B1E  502D         02DB80A  0023F0F0  ........  ........
00000080  00220095  00          00000000  0023F100  ........  DV......
00000090  00000000  0           0023F739  0023F110  ........  ......7.
000000A0  502D9D00  0           002DBAA0  0023F210  ......1.  ........
000000B0  502D9B1E           0  502DB80A  0023F130  ........  ........
000000C0  502DC80A          E4  4022F720  0023F140  ..H...Q.  ...U..7.
000000D0  0022EF40         000  00000000  0023F150  ..?.....  ........
000000E0  0000000          000  00000000  0023F160  ........  ........
000000F0  000000          0000  00000000  0023F170  ........  ........

SDUMP = ...  +           DRUCKER = ....         PF4 = PRINT   7 SB 8SF
```

In der 1. Zeile wird der Phasenname des Programms (hier TST005) und Bereich angezeigt, auf den der Terminal-Dump erstmals zugreift. Beim 1. Aufruf ist dies immer die Task-Control-Area (TCA).

In den vier folgenden Zeilen wird der Inhalt der Register E,F,0,1 usw bis B,C,D angezeigt, wobei jeweils links über bzw. unter dem Registerinhalt die Registernummer angezeigt wird. Im obigen Beispiel bedeutet:

```
F          Der Inhalt des Registers  15 (F)  war vor  Eintritt  in den
0023F180   SDUMP  = 0023F180.
```

Die folgenden 16 Zeilen geben jeweils 256 Bytes des Hauptspeichers in hexadezimaler und Character-Schreibweise wieder. Die Spalten 2,3,4 und 5 dieses Abschnitts zeigen dabei den Speicherinhalt hexadezimal an.

Die Spalten 7 und 8 zeigen den Inhalt in Klartext, wobei alle nicht druckbaren Zeichen einschließlich der Sonderzeichen mit einem hexadezimalen Wert kleiner als 'C1' durch einen Punkt ersetzt werden.

Spalte 1 dieses Abschnitts zeigt die relative Adresse zum jeweiligen Anfangspunkt, im obigen Beispiel zur TCA-Adresse.
Spalte 6 dieses Abschnitts zeigt jeweils zur relativen Adresse in Spalte 1 die absolute Hauptspeicheradresse an.

Über die Programmfunktionstaste DE können die jeweils nächsten 256 Bytes angefordert werden. Dabei kann im Dump beliebig weit geblättert werden. Mit PF3 wird der SDUMP verlassen.

Über die Programmfunktionstaste PF7 kann entsprechend zurückgeblättert werden. Mit CL wird das Programm abgebrochen.

Die letzte Zeile bietet schließlich die Möglichkeit, den Dump von jeder beliebigen Stelle der CPU anzuzeigen. Dabei hat der Programmierer die Möglichkeit, auf folgende Bereiche direkt aufzusetzen:

```
ADR   Adresse XXXXXX
CSA   Common System Area
CIO   CPG Input-Output-Area
CWA   Bereich Common Work-Area
END   Ende Sdump                  Das Aufsetzen erfolgt durch Einga-
HLB   HL1 Library                 be einer der  nebenstehenden  Kon-
IFC   Interface Com. Area         stanten  bei: SDUMP = ...  und Be-
MBK   Methodenbank                tätigung der Datenfreigabe-Taste.
PRG   Anwendungsprogramm
PWA   Private Work Area
SIT   Anzeige der CPGURSIT/Kundenkonfiguration
TCA   Task Control Area
TCT   TCT  User    Area
```

Dabei kann der Programmierer bei + ... noch einen Verschiebungsfaktor angeben, z.B. TCA + 000100 ist die Transaction Work Area.

Bei der Eingabe SDUMP + kann auch mit einer verkürzten Eingabe die Anzeige gestartet werden. Im Klartext werden auch Kleinbuchstaben angezeigt.

Wird bei SDUMP = ... die Konstante 'ADR' eingetragen, so muss das folgende Feld (+ .....) eine gültige Adresse enthalten. Diese Adresse, die auch außerhalb der TP-Partition liegen kann, gilt dann nach Druck der DE-Taste als Startadresse für den Terminal-Dump.

Bei Drucker = ... kann der Programmierer schließlich die Destination-Id des Druckers angeben, auf den der Bildschirminhalt nach Druck der PF4-Taste ausgegeben werden soll.

Wird beim SDUMP Drucken (PF4) ausgewählt und keine Drucker-Id eingegeben, so wird die letzte Anzeige wiederholt. Der SDUMP wird nicht abgebrochen.

Wenn ein Online-Ausdruck durchgeführt werden soll, so muss die Transaktion TPDU oder TPDC (Transient Data Programmm) in der CICS DCT aktiviert sein.

Programmfunktionstasten:

DE	Blättern vorwärts um 256 Bytes.
PF2	Neuaufsetzen.
PF3	Ende
PF4	Drucken.
PF7	Blättern rückwärts um 256 Bytes.
PF8	Blättern vorwärts um 256 Bytes.
CL	Programm abbrechen
andere	Ende SDUMP

Quick Debug Facility 2833

Für CPG2-Benutzer steht außerdem das QDF zur Fehlersuche und als Testhilfe zur Verfügung.

Einschränkungen 2900

Ungeblockte Dateien

Für ungeblockte ISAM-Dateien verwendet das CICS unterschiedliche Ein-/Ausgabebereiche für die reine CHAIN-Operation (ohne UPDATE) und für alle übrigen Operationen. Daher gilt die unter ISAM aufgestellte Regel, dass eine Datei sowohl Random als auch sequentiell im gleichen Programm verarbeitet werden kann,in diesem Falle nicht.

Für VSAM-Dateien wird empfohlen, die Datei in der FCT mit dem Parameter RECFORM=(FIXED,BLOCKED) anzugeben.

T1 - T9 Schalter dürfen in der Ausgabe bei der Feldaufbereitung nicht eingesetzt werden. Das gleiche gilt im Batch für die Schalter U1-U9.

Einschränkungen der CICS-Version 2905

Einschränkungen der Subsetversion (Eintragung H-Karte Spalte 47 = C) (Macro Level) oder D (Command Level).

Für die CICS-Subsetversion des CPG gelten gegenüber der Standardversion folgende Einschränkungen:

In den Rechenbestimmungen werden die Operationen CHECK, DEQ, ENQ, EXITD, LOKUP, MAP, NEWRC, TESTN, UPDAT, VBOMP, VSLCT, READB, READI, IFC, CALL, UCTRAN, USSA, QSSA, DLI nicht unterstützt.

Die Operation CHAIN ist nur in der RPG-Form möglich, d.h. U in Spalte 53 ist hierbei nicht erlaubt.

Die CPG-Felder CPGTIM und CPGTIO stehen nicht zur Verfügung.

Für Bezugszahlen in den Feldausgabebestimmungen für eine Bildschirmdatei gelten Einschränkungen. Siehe hierzu Formularbeschreibung der Ausgabebestimmungen.

Der Terminaldump (SDUMP) kann nicht bei einer beliebigen Adresse aufsetzen. Der Programmcode und die Funktionstaste wird nicht angezeigt.

Bei Fehlern im IOCS während der Ausführung der Programme (z.B Datei nicht eröffnet) erfolgt Programmabbruch ohne Fehlermeldung.

Temporary Storage ist nicht unterstützt.

Edit-Codes für Terminal und Drucker Buffer-Mode nicht unterstützt.

Siebenfarbige Ausgabe nicht unterstützt.

Map-Ausgabe nicht unterstützt.

Die Eingabe von Kleinbuchstaben ist nicht unterstützt.

Bei der DSPLY-Operation gibt es Einschränkungen.Siehe Abschnitt 3170.

Die Eintragung TWA Copy in der E-Karte ist nicht unterstützt.

Bei der EXITI-Operation ist die Nonterminal-Task nicht unterstützt.

READ für Transient Data ist nicht unterstützt.

Bei der Operation EXPR, EXITP und EXITT ist eine Eintragung im Ergebnisfeld nicht unterstützt.

Variabler Dateiname bei Transient Data ist nicht unterstützt.

Die Löschtaste 'CL' kann in der CICS-Version nicht abgefragt werden.

47 'D' EXEC CICS Version. Beim Programmstart wird automatisch keine Bildschirmeingabe durchgeführt. Außerdem kann keine Ausgabe auf Transient Data durchgeführt werden.

Einschränkungen bei Assembler-Dataset-Verarbeitung 2910

Die Operationen WRITE, DELET und UPDAT sind nicht unterstützt.

Die Dataset Verarbeitung in Verbindung mit Include-Version ist nicht unterstützt.

Einschränkungen 'S' Spalte 32 der H-Karte 2920

Die Operationen WRITE, DELET und UPDAT sind für VSAM-Dateien nicht unterstützt.

Einschränkungen bei Command-Level Programmen 2930

Die Operation DEBUG ist nicht unterstützt.

Einschränkungen der HL1 Version 2940

Die Operation CLEAR darf in HL1-Bausteinen nicht verwendet werden.

In einem Hauptprogramm, bzw. HL1-Modul dürfen maximal 99 verschiedene Module aufgerufen werden.

Einschränkungen bei HL1-Batch 2950

Siehe Abschnitt 6310.

ESA-Mode 2970

Alle softwareseitigen Verbesserungen, die ESA bietet, sind im Command Level verwirklicht. In Zukunft wird der Macro Level nicht mehr unterstützt, genauer

- in VSE/ESA-Umgebungen in den CICS/ESA Releases größer 2.2
- in MVS/ESA-Umgebungen ab CICS/ESA Release 3.2

Wenn in diesem Kapitel vom ESA-Mode die Rede ist, dann beziehen sich die Ausführungen immer nur auf die genannten Umgebungen, in denen der Macro Level nicht mehr existiert.

Zur Umstellung auf den ESA-Mode müssen CPG-Programme mit einem neuen Command Level/ESA-Interface umgewandelt werden. Mit dieser Umstellung, (die erst bei Einsatz der oben genannten Umgebungen notwendig wird), kann ab dem CPG-Release 1.6 begonnen werden.

ESA-Mode-fähige Programme sind unter heutigen CICS-Releases bereits ablauffähig.

Vorgehensweise:
1. alle eigenen Programme im ESA-Mode umwandeln.
2. CPG2*LNE und CPG3*LNE umwandeln.
3. jetzt können CPG-Programme mit PARM='AMODE=31, RMODE=ANY' gelinkt werden.

Nur bei ESA-Mode unterstützt:

SCS-Printer mit QLF direktes Drucken.

Standard-H-Karte 2974

Die ESA-Fähigkeit wird über die H-Karte gesteuert. Da es sich um einen unternehmensweiten Standard handelt, wenn ESA-Mode eingeführt werden soll, sollte auch die entsprechende Eintragung (E in Spalte 47 des Copy Books CPGUCSTH) in der Standard-H-Karte vorgenommen werden.

ESA erzeugt ein Command Level-Programm, das auf das Command Level/ESA-Interface zugreift.

Macro Level-Programme sowie Nicht-ESA-Command Level Programme werden in den genannten Umgebungen nicht mehr ablauffähig sein.

Da aber bis zur tatsächlichen Umstellung noch eine lange Übergangsphase den Parallelbetrieb von ESA-Mode-fähigen Programmen und Nicht-ESA-Mode-fähigen Programmen ermöglicht, ist dies auch mit CPG möglich.

Ein Eintrag 'O' in Spalte 47 generiert auch bei einem Parameter E(SA) in der Standard-Header-Karte ein Command-Level-Programm, das auf das bisherige Command Level Interface (CPGCLI) zugreift.

Ein Eintrag 'M' in Spalte 47 generiert ein Macro Level Programm.

Beachte:

Grundsätzlich überschreiben Eintragungen in Spalte 47 der H-Karte, die im Programm gesetzt werden, den Eintrag in der Standard Header Karte. Hiervon ausgenommen ist der Parameter 'L' für Command Level Programme. 'L' überschreibt ESA nicht. Um 'alte' Command Level-Programme zu generieren, steht deshalb die Eintragung 'O' zur Verfügung.

Wird in Spalte 97 der Standard Header-Karte ein 'C' wie CANCEL eingetragen, dann brechen Umwandlungen ab, die nicht zu ESA-Mode-fähigen Programmen führen. Ansonsten bekommen solche Programme eine Warning CPG0070 (siehe Fehlermeldungen).

Linken 2976

Bei der Umwandlung ESA-Mode-fähiger Programme muss beachtet werden, dass das Linken mit INCLUDE DFHEAI und CPGSCEIC durchgeführt wird.

Fehlermeldungen 2977

Im ESA-Mode brechen Macro Level-Programme und nicht ESA Mode-fähige Command Level Programme mit ASRD ab.

Ab Release 1.6 werden Umwandlungen, die nicht ESA-Mode-fähige Programme generiert haben, mit einer WARNING CPG0070 abgeschlossen.

Veränderungen infolge des ESA-Mode 2978

- Einzelsatzverarbeitung von Temporary Storage-Bereichen wird nicht mehr unterstützt sein.

  Abhilfe:

  Durch die Eintragung S in der FILE Description (bzw. im Data Dictionary) erzeugt CPG eine Queue, die aus einem Satz besteht. Sonstige Eingriffe in die Verarbeitung der Queue brauchen vom Programmierer nicht vorgenommen zu werden.

  Hierbei ist zu beachten, dass alle Programme, die den entsprechenden TS-Bereich verarbeiten, gleichzeitig konvertiert werden müssen. Die gemischte Verarbeitung eines Bereichs (einmal als Queue, einmal als Einzelsatz) führt zu einem "ILLOGIC"-Fehler.

- Um aus ESA-Mode-fähigen Programmen HL1-Module aufrufen zu können, müssen diese Module mit Release 1.6 des CPG oder höher umgewandelt sein.

- Um Programme in der CICS-Version zu generieren (das sind Programme, die ohne Methodenbank ablaufen) ist anstelle von 'C' ein 'D' für CICSESA einzutragen.

- Register 13 adressiert nicht mehr die CSA. In Programmen mit assemblergeschriebenen Sequenzen können zwischen BEGAS und ENDAS die Felder CPGCSA und CPGCSACI nicht mehr angesprochen werden.

- 'DI' für Data Dictionary Independent ist für ESA nicht unterstützt.

Operationen 3000

```
ADD   Addieren
AFOOT Mittelwert einer Feldgruppe rechnen
BEGAS Beginn Assembler
BEGDT Beginn Entscheidungstabelle
BEGSR Beginn Unterprogramm
BITON Bitschalter setzen
BITOF Bitschalter löschen
BREAK Beenden einer DO-Schleife
CABXX Vergleichen und Verzweigen bei ...
CALL  Link Unterprogramm
CALLD Dataset Benutzerzugriff
CALLM Unterprogramm aus der Methodenbank abrufen.
CASxx Vergleichen und Subroutine Aufruf
CBSxx Vergleichen und Subroutine Aufruf
CHAIN Satz ketten
CHANG Feldinhalte austauschen
CHECK Dateistatus prüfen
CLEAR TWA initialisieren
CLOSE Datei abschließen
COMP  Vergleichen
COMRG Communication Region
CONT  Unterbrechen einer DO-Schleife
CONVT Konvertieren eines alphanumerischen Feldes
COPY  Assembler Unterprogramm einfügen
CLRIN Bezugszahl mit Index löschen
DEBUG Test-Hilfe
DELC  Zeichen entfernen
DELET Satz löschen
DEQ   Programmteil entriegeln
DIV   Dividieren
DLI   Call für DL/I aufrufen
DO    Schleife
DOU   Ausführen...bis
DOW   Ausführen...solange
DSPLY Ausgabe auf der Konsole
DUMP  Dump ausgeben
EDIT  Feld aufbereiten
ELIM  Zeichen entfernen
ELSE  Sonst...ausführen
END   End
ENDAS Ende Assembler
ENDDO DOxxx Ende
ENDDT Ende Entscheidungstabelle
ENDIF IFxx Ende
ENDSR Ende Unterprogramm
ENQ   Programmteil sperren
EREAD Erweitertes Bildschirm-Read
EXCPT Ausgabe
EXITD Direkt verzweigen mit Daten
EXITI Direkt verzweigen
EXITP In externes Programm verzweigen
EXITT Verzweigen für Readfreie Programme
EXPR  Programm ausführen
EXSR  Unterprogramm ausführen
FILL  Alphafeld füllen
FIND  Durchsuchen einer Datenview
GOTO  Verzweigen nach
```

IF Wenn-Abfrage
IFC Interface-Call
INDOF Bezugszahlen löschen
INDON Bezugszahlen setzen
JRB Rechtsbündig verschieben mit Blank
JRC Rechtsbündig verschieben mit Zeichen
JRZ Rechtsbündig verschieben mit Null
LIST Ausgabe extern beschriebener Listen
LOADT Wiedererstellen Bildschirminhalt
LOKUP Tabellen suchen
MACRO Assembler Macro einfügen
MAP Lesen QSF-Map transaktionsorientiert
MAPD Dialog QSF-Map
MAPI Eingabe QSF-Map
MAPO Ausgabe QSF-Map
MAPP Ausgabe QSF-Map auf Drucker
MLLZO Move Low to Low Zone
MOVE Feld rechtsbündig übertragen
MOVEA Bereich übertragen
MOVEL Feld linksbündig übertragen
MOVEN alphanumerisches in numerisches Feld übertragen
MOVEV variable MOVE-Operation
MULT Multiplizieren
MVR Rest übertragen
OPEN Datei eröffnen
PARM Parameter definieren
PRNT Drucksteuerung Assemblerliste
PROG QPG Programm ausführen
PROT Nur für CPG-TOP Anwender
PURGE Temporary Storage Queue löschen
QSSA Aufbereiten Qualifiziertes SSA für DL/I
READ Satz lesen
READB Rückwärts lesen
READI Lesen Bildschirm transaktionsorientiert/FWA lesen
READP Seite lesen
REPLC Blank durch Zeichen ersetzen
ROLL Feldgruppe verschieben
ROLLB Feldgruppe rückwärts verschieben
RNDOM Wahlweise Verarbeitung
SAVET Retten Bildschirm Inhalt
SCAN Alphafeld nach einer Zeichenfolge durchsuchen
SDUMP Special Terminaldump
SELCT Feldauswahl
SETIN Bezugszahl mit Index setzen
SETIX Index setzen
SETLL Set lower Limit
SETOF Bezugszahl löschen
SETON Bezugszahl setzen
SORTA Feldgruppen sortieren
SQRT Wurzel ziehen
SUB Subtrahieren
SYNCP Sync. Point setzen für Recovery
TAG Merkmal setzen
TESTB Bitschalter prüfen
TESTN Auf numerische Zeichen prüfen
TESTT Terminal abfragen
TIME Zeit setzen
TWALD TWA von Temporary Storage einlesen
TWASV TWA auf Temporary Storage retten
UCTRN CICS Parameter für Groß-/Kleinschreibung ändern

```
UPDAT Satz zurückschreiben
USSA  Aufbereiten Unqualifiziertes SSA für DL/I
VBOMP VBOMP-Zugriff
VSLCT Lesen eines VBOMP-Bereichs
WAIT  Warten
WRITE Satz hinzufügen
XFOOT Summe einer Feldgruppe rechnen
Z-ADD Löschen und addieren
Z-SUB Löschen und subtrahieren.
+     Addieren
-     Subtrahieren
*     Multiplizieren
/     Dividieren
=     Löschen und Addieren
----- Nach Bedingungsteil Entscheidungstabelle
```

Eine Übersicht nach Funktionen siehe Seite 7005.

Für das Distributed Transaction Processing werden die Schnittstellen LU61 und LU62 (mapped und unmapped Conversation) mit folgenden Operationen unterstützt:

```
ALLOC Verbindungsaufbau ( Allocate )
CNVRS Konversation zwischen zwei Anwendungen ( Converse )
CONCT Verknüpfen zweier Transaktionen (verschiedene Systeme)
                                        ( Connect )
EXCPT (mit Ergebnisfeld) Ausgabe LU61/LU62
EXTRT Lesen Conversation-Daten ( Extract )
FREE  Freigeben einer Session / Verbindung
READ  ( mit Ergebnisfeld ) Lesen der Daten einer anderen Task
```

Für das Distributed Transaction Processing steht eine separate Dokumentation zur Verfügung.

ADD Addieren 3101

Nach Ausführung der Instruktion enthält das Ergebnisfeld die Summe von Faktor 1 und Faktor 2.

Faktor 1, Faktor 2 und Ergebnisfeld können auch den Namen einer Feldgruppe enthalten. In diesem Fall werden alle Elemente dieser Feldgruppe zu den entsprechenden Elementen der anderen Feldgruppe addiert.

Ist die Anzahl der Elemente unterschiedlich, so werden nur so viele Elemente addiert, wie für die kleinste Feldgruppe definiert wurden.

Wird der Name einer Feldgruppe eingetragen, so kann mit einem festen (FG,3 FG,7) oder variablen (FG,I FG,KX) Index bestimmt werden, welche Einzelelemente der Feldgruppe zu addieren sind.

Beispiel: FG1,1 ADD FELDA FG1,IN

Wobei FELDA der Name eines numerisch definierten Feldes, FG1 der Name einer numerisch definierten Feldgruppe, 1 der feste Index 1 und IN der Name des numerisch definierten Indexfeldes ist, das den variablen Index enthält.

Wenn die Eintragung von Faktor 1 und Ergebnisfeld gleich ist, so kann die Eintragung in Faktor 1 entfallen.

Folgende Aufstellung bietet eine Übersicht der möglichen Eintragungen bei numerischen Operationen:

C	BEDING.	FAKTOR1	OPCD	FAKTOR2	ERG.	LG	DR	BEZ	KOMMENTAR
C		A	ADD	B	A				A+B=A
C		A	ADD	B	A	52			LG=5 davon 2 Dez
C		A	ADD	B	A		12		Prüfe auf +
C	10	A	ADD	B	C				Wenn 10: A+B=C
C		A	ADD	1	A				A+1=A
C		A	ADD	-15	A				A-15=A
C		A	ADD	5,67	B		H		A+5,67=B,Runden
C		A	ADD	.20	A				Dezimalpunkt
C		A	ADD	0,20	C				Dezimalkomma
C		FG1	ADD	FG2	FG1				ADD Feldgruppen
C		FG1	ADD	1000	FG1				Alle + 1000
C		FG1	ADD	FG2	FG3				ADD Feldgruppen
C		FG1,1	ADD	FG2,3	FG1,1				ADD Elemente
C		FG1,I	ADD	FG2,J	FG3,K				ADD Indiziert
C		FG1,4	ADD	1000	FG1,4				4.Elem + 1000
C			ADD	B	A				A+B=A
C			ADD	1	A				A+1=A
C			ADD	1000	FG1,4				4.ELEM + 1000

AFOOT Mittelwert einer Feldgruppe rechnen 3102

Mit dieser Operation kann der arithmetische Mittelwert einer numerischen Feldgruppe errechnet werden.

Faktor 2 enthält den Namen der Feldgruppe. Das Ergebnisfeld enthält den Namen des Feldes, in das der Mittelwert gespeichert werden soll.

Felder mit dem Inhalt 0 werden nicht in die Mittelwertrechnung einbezogen.

Beispiel: AFOOTFG1 MW

Inhalt der Feldgruppe 'FG1': FELD 1 125,00
FELD 2 75,00
FELD 3 85,00
FELD 4 0,00
FELD 5 0,00
FELD 6 0,00

Inhalt des Feldes 'MW'
Nach Ausführung der Instruktion: 95,00

```
   BEDING.  FAKTOR1   OPCD FAKTOR2  ERG.  LG DR BEZ  KOMMENTAR
C.-.--.--.--.---------.----.--------.-----.--...-.-.-.--------------.-

C                     AFOOTFG1      MW                MW = Mittel FG1
C   10                AFOOTFG1      MW                Wenn 10 ...
C                     AFOOTFG1      MW      70        LG=7 davon 0 Dez
C                     AFOOTFG1      MW           12   Prüfe auf >
```

BEGAS Beginn Assembler 3121

Diese Operation besagt, dass ab der nächsten Operation das Programm im Assemblermodus fortgeführt wird, d.h. dass der Programmierer sein Programm an beliebiger Stelle durch Assembler-Passagen nterbrechen kann.

Für den Assemblerteil erfolgt jedoch im CPG-Programm keine Diagnostik. Der Programmierer sollte darauf achten, dass Spalte 11 der CPG-Steuerkarte ein A enthält.

Beispiel: BEGAS

```
          MVC   A,B   (in Assemblerformat)

                ENDAS
```

BEGDT Beginn Entscheidungstabelle 3122

Siehe Entscheidungstabellen (2410)

BEGSR Beginn Unterprogramm 3123

Beginn eines Unterprogramms. In Faktor 1 muss der Name des Unterprogramms eingetragen werden. Unterprogramme müssen am Ende der Rechenbestimmungen liegen. Die Eintragung 'SR' in Spalte 7-8 ist nicht erforderlich.

```
   BEDING.  FAKTOR1   OPCD FAKTOR2  ERG.  LG DR BEZ  KOMMENTAR
C.-.--.--.---------.----.--------.-----.--.--.-.--------------.-
C                     BEGAS                          BEGINN ASSEMBLER
C           ETAB1     BEGDT                          ENTSCH.TAB. ETAB1
CSR         UPRO      BEGSR                          UNTERPROGR. UPRO
```

BITOF Bitschalter löschen 3124

Mit dieser Operation können 1 bis 8 Bits in einem Byte gelöscht werden. Das Ergebnisfeld enthält den Namen des zu verändernden einstelligen Feldes. Faktor 2 enthält in Hochkommata eingeschlossen die zu verändernden Bits in Form von Ziffern von 0 bis 7 in Zählrichtung von links nach rechts.

Diese Operation gilt nur für alphanumerische Felder.

Beispiel: BITOF'567' BYTE 1

Inhalt des Feldes Byte	vorher	nachher
In Bit-Schreibweise:	11110111	11110000
In hexadezimaler Schreibweise:	F 7	F 0
In Characterform:	7	0

BITON Bitschalter setzen 3125

Mit dieser Operation können 1 bis 8 Bits in einem Byte gesetzt werden. Das Ergebnisfeld enthält den Namen des zu verändernden einstelligen Feldes. Faktor 2 enthält in Hochkommata eingeschlossen die zu verändernden Bits in Form von Ziffern von 0 bis 7 (in Zählrichtung) von links nach rechts.

Diese Operation gilt nur für alphanumerische Felder.

Beispiel: BITON'23' BYTE 1

Inhalt des Feldes BYTE	vorher	nachher
In BIT-Schreibweise:	11000001	11110001
In hexadezimaler Schreibweise:	C 1	F 1
In Characterformat:	A	1

BREAK Eine Schleife beenden 3126

Eine DO-, DO UNTIL- oder DO WHILE-Schleife soll beendet werden.

Durch den Befehl BREAK wird eine DO-Schleife unabhängig von der Schleifenbedingung sofort beendet.

BREAK verzweigt hinter das ENDDO. Die Statements zwischen BREAK und ENDDO werden nicht mehr ausgeführt.

Wird in Spalte 53 ein 'A' eingetragen, so bedeutet dies, dass bei verschachtelten DO-Schleifen die Verarbeitung bis zur äußersten Ebene (bei Subroutinen bis zur äußersten Ebene innerhalb der Subroutine) beendet wird.

CABxx Vergleichen und Verzweigen 3127

Die CABxx-Operation vergleicht Faktor 1 mit Faktor 2. Stimmt das Ergebnis der Operation mit dem xx-Teil der Operation überein, verzweigt das Programm zu der TAG-Operation, die dem Label im Ergebnisfeld entspricht. Stimmt das Ergebnis der Operation nicht mit dem xx-Teil der Operation überein, wird das Programm mit der nächst folgenden Operation fortgeführt.

Bedingende Bezugszahlen können spezifiziert werden. Faktor 1 und Faktor 2 müssen ein alphanumerisches oder numerisches Literal oder einen Feldnamen enthalten. Die Eintragungen in Faktor 1 und Faktor 2 müssen beide alphanumerisch oder beide numerisch sein.

Der im Ergebnisfeld spezifizierte Name der TAG-Marke muss einer eindeutigen TAG-Operation zugeordnet sein. Ergebnisbezugszahlen können wahlweise spezifiziert werden, es sei denn, xx ist blank. Ist xx blank, muss zumindest eine Ergebnisbezugszahl eingetragen werden.

Sind Ergebnisbezugszahlen spezifiziert, werden sie gesetzt, um das Ergebnis des Vergleichs anzuzeigen. Die Regeln für die CABxx-Operation sind bei den Vergleichsoperationen in diesem Kapitel beschrieben.

xx	Bedeutung
GT	Faktor 1 ist größer Faktor 2
LT	Faktor 1 ist kleiner Faktor 2
EQ	Faktor 1 ist gleich Faktor 2
NE	Faktor 1 ist ungleich Faktor 2

GE Faktor 1 ist größer oder gleich Faktor 2
LE Faktor 1 ist kleiner oder gleich Faktor 2
Blanks Ein GOTO zu dem im Ergebnisfeld eingetragenen TAG wird unabhängig vom Vergleich ausgeführt

Beispiel:

```
C            NF        CABNE4711      LABEL
```

Sind Ergebnisbezugszahlen spezifiziert, werden sie gesetzt, um das Ergebnis des Vergleichs anzuzeigen, und zwar unabhängig vom verwendeten Operationscode.
Das heisst im einzelnen :

Bezugszahl 1 wird gesetzt, wenn F1 > F2 ist.
Bezugszahl 2 wird gesetzt, wenn F1 < F2 ist.
Bezugszahl 3 wird gesetzt, wenn F1 = F2 ist.

CALL Unterprogramm ausführen 3130

Die CALL-Operation erlaubt den Zugriff auf praktisch alle Datenbanken, sowohl online als auch im Batch. Eigene Unterprogramme (Object Code) können mit CALL zum Programm gelinkt werden.

Mit der CALL-Operation wird die Steuerung auf ein in Faktor 2 angegebenes Unterprogramm übertragen. Faktor 1 bleibt leer.

Im Ergebnisfeld kann ein numerisches Feld ohne Dezimalstellen eingetragen werden. In diesem Feld wird bei der Rückkehr aus dem Unterprogramm ein Returncode übertragen.

Das Ergebnisfeld darf nur benutzt werden, wenn die Unterroutine einen Returncode setzt, da sonst unvorhersehbare Returncodes gesetzt werden können.

```
 C              CALL 'PROGA'       RC
```

Nach der Verarbeitung des Programms 'PROGA' kehrt die Steuerung zur nächsten zu verarbeitenden Anweisung zurück. Zur Datenübergabe siehe Operation PARM.

Das Unterprogramm wird zur Linkage Editor-Zeit mit dem CPG-Programm verbunden.
Parameter können unmittelbar hinter der CALL-Operation mit PARM-Anweisung(en) übergeben werden.

Beachte: Bei CALL-Aufrufen von COBOL-Programmen setzt man zuerst ein CALL ILBDSET0 ab (damit das COBOL-Programm weiß, dass es ein Unterprogramm ist)

Die bisherige Operation wurde aus Gründen der Kompatibilität zu RPG in 'CALLM' umbenannt. Die alte Bezeichnung ist ab Release 2.1. in der Form nicht mehr verfügbar.

CALLD Dataset Zugriff **(veraltet !!!)** 3131

Mit dieser Operation kann ein benutzerspezifischer Dataset-Zugriff ausgeführt werden. In Faktor 1 muss ein Schlüsselfeld eingetragen werden.

Der maximal 8-stellige Dataset-Name muss in Faktor 2 eingetragen werden. Eintragungen in Spalte 53 werden an das Dataset übergeben. Diese Operation setzt den Datenbank-Prozessor voraus.

CALLM Unterprogramm aus der Methodenbank abrufen 3132

Mit dieser Operation kann ein benutzerspezifisches Unterprogramm aus der Methodenbank abgerufen werden.

Faktor 2 enthält den maximal 8-stelligen Namen des Unterprogramms.

Das Ergebnisfeld kann den Namen eines Feldes, einer Feldgruppe oder einer Parameterliste enthalten, die im Unterprogramm verwendet werden. Eine Parameterliste kann beispielsweise durch Überlagerung von Feldern in der TWA erzeugt werden.

Beispiel:

```
E                    PARLST      0 14      (1+4+4+5)
E                    ME             1
E                    MENGE          7 0    (GEPACKT=4)
E                    PREIS          7 2    (GEPACKT=4)
E                    WERT           9 2    (GEPACKT=5)

C                     CALLMUPWERT    PARLST
```

Im Unterprogramm 'UPWERT' soll der Wert einer Rechnungsposition errechnet werden. Dazu müssen dem Unterprogramm die Felder ME, MENGE und PREIS übergeben werden.

Das Ergebnis soll nach Ablauf des Unterprogramms im Feld WERT stehen. Das Feld WERT muss folglich dem Unterprogramm ebenfalls bekannt sein. Die erforderlichen Felder werden dem Unterprogramm in Form einer Parameterliste übergeben.

Voraussetzung ist, dass die Methodenbank eine Assembler-Routine mit dem Namen UPWERT enthält. Einzelheiten siehe (9500).

CASxx CASxx Gruppe vergleichen und Aufruf Subroutinen 3139

Das Ergebnisfeld muss den Namen einer gültigen Subroutine enthalten. Wenn die Bedingung, festgelegt durch xx,

zwischen Faktor1 und Faktor2 besteht, wird die im Ergebnisfeld definierte Subroutine ausgeführt. Besteht die Bedingung nicht, so wird die nächste CASxx-Operation in der CAS-Gruppe durchgeführt.

Eine CAS-Gruppe kann nur CASxx-Operationen enthalten. Eine END-Anweisung muss dem letzten CASxx Befehl folgen. Nach Durchführen der Subroutine verzweigt das Programm zur nächsten Operation, die dem END Statement dieser CAS-Gruppe folgt.

Die unbedingte 'CAS ' Operation ohne Bezugszahlen entspricht funktionell einer EXSR-Operation.

Alle CASxx-Operationen, die einer unbedingten CAS-Operation folgen, werden nicht ausgeführt. Somit ist eine solche Operation nur sinnvoll als letzte Anweisung vor dem END-Statement einer CAS-Gruppe.

xx kann bei CASxx folgenden Inhalt haben:

```
'GT'      Faktor1 ist größer Faktor2
'LT'      Faktor1 ist kleiner  Faktor2
'EQ'      Faktor1 ist gleich   Faktor2
'NE'      Faktor1 ist ungleich Faktor2
'GE'      Faktor1 ist größer oder gleich Faktor 2
'LE'      Faktor1 ist kleiner oder gleich Faktor 2
'  '      kein Vergleich
```

Beispiel:

```
C           FELDA     CASGTFELDB     UP01
C           FELDA     CASEQFELDC     UP02
C                     CAS            UP03
C                     END
```

Die CASGT-Operation vergleicht den Inhalt von FELDA mit FELDB und führt, wenn FELDA größer ist, die Subroutine UP01 durch. Ist FELDA gleich mit FELDC, so wird die Subroutine UP02 durchgeführt, ansonsten UP03.

Sind Ergebnisbezugszahlen spezifiziert, werden sie gesetzt, um das Ergebnis des Vergleichs anzuzeigen und zwar unabhängig vom verwendeten Operationscode.
Das heisst im einzelnen :

```
Bezugszahl 1  wird gesetzt, wenn  F1 > F2  ist.
Bezugszahl 2  wird gesetzt, wenn  F1 < F2  ist.
Bezugszahl 3  wird gesetzt, wenn  F1 = F2  ist.
```

CBSxx CBSxx Vergleich und Aufruf Subroutine 3140

Das Ergebnisfeld muss den Namen einer gültigen Subroutine enthalten. Wenn die Bedingung, festgelegt durch xx, zwischen Faktor1 und Faktor2 besteht, wird die im Ergebnisfeld definierte Subroutine ausgeführt. Besteht die Bedingung nicht, so wird die nächste Operation, die der CBS-Anweisung folgt, durchgeführt.

Nach Durchführen der Subroutine verzweigt das Programm zur nächsten Operation, die dem CBS-Statement folgt. Der xx-Anzeiger darf nicht auf Blank gesetzt sein. In diesem Fall ist statt der CBS-Operation EXSR zu verwenden.

xx kann bei CBSxx folgenden Inhalt haben:

```
'GT'      Faktor1 ist größer Faktor2
'LT'      Faktor1 ist kleiner Faktor2
'EQ'      Faktor1 ist gleich Faktor2
'NE'      Faktor1 ist ungleich Faktor2
'GE'      Faktor1 ist größer oder gleich Faktor2
'LE'      Faktor1 ist kleiner oder gleich Faktor2
```

Beispiel:

```
C           FELDA     CBSEQFELDB     UP01
```

Entspricht der Inhalt von FELDA dem Inhalt von FELDB so wird die Subroutine UP01 durchgeführt.

Sind Ergebnisbezugszahlen spezifiziert, werden sie gesetzt, um das Ergebnis des Vergleichs anzuzeigen, und zwar unabhängig vom verwendeten Operationscode.

Das heisst im einzelnen :

```
Bezugszahl 1  wird gesetzt, wenn  F1 > F2  ist.
Bezugszahl 2  wird gesetzt, wenn  F1 < F2  ist.
Bezugszahl 3  wird gesetzt, wenn  F1 = F2  ist.
```

CHAIN Satz wahlfrei lesen 3141

Mit dieser Operation wird ein Satz, der in Faktor 2 genannten RANDOM-Datei mit dem in Faktor 1 genannten Schlüssel gelesen. Die Datei muss eine Platten-Datei sein. Die Schlüssellänge muss in der zugehörigen Dateizuordnung definiert werden. In Spalte 54 kann eine Bezugszahl eingetragen werden. Diese Bezugszahl wird vom Programm gesetzt, wenn der gesuchte Satz nicht gefunden wurde.
Bei HL1-Datasets ist die Bezugszahl in Spalte 54 nicht erforderlich.
Ein 'C' in Spalte 53 bedeutet, dass der Satz nur auf Verfügbarkeit geprüft wird. In diesem Fall werden keine Daten gelesen. Die Bezugszahl in Spalte 54-55 wird gesetzt, wenn der Satz nicht gefunden wurde.
Es braucht kein Schalter angegeben zu werden. Die Information 'NF' für not found wird im internen Feld CPGFRC übergeben.
Ein 'P' in Spalte 53 schließt C und U ein.
Ein 'U' in Spalte 53 bewirkt, dass der Satz solange für ein weiteres CHAIN mit 'U' in 53 gesperrt wird, bis ein Update durchgeführt oder der Satz durch ein RNDOM wieder freigegeben wird. Es wird lediglich auf Satz-Ebene gesperrt. Bei Share-Option 4 oder Journaling wird das gesamte VSAM-CI gesperrt.
Im Ergebnisfeld kann ein Label (einer TAG-Operation) eingetragen werden.

Wenn der Schlüssel nicht gefunden wurde, wird die Bezugszahl gesetzt und zu diesem Label verzweigt.

Ein '**' in Spalte 54-55 ist in Verbindung mit einem Eintrag im Ergebnisfeld unterstützt. Wenn der Schlüssel nicht gefunden wird, so wird zu diesem Label verzweigt.

Bei HL1-Dataset-Verarbeitung benötigt die CHAIN-Operation immer Input-Bestimmungen. Bei VSAM-Dateiverarbeitung ist CHAIN auch ohne Input-Bestimmungen unterstützt. Bei QPG-Datasets werden ebenfalls keine Input-Bestimmungen benötigt.

CHANG Feldinhalte austauschen 3142

Mit dieser Operation können die Inhalte zweier Datenfelder ausgetauscht werden.

Die Feldnamen stehen in Faktor 1 und Faktor 2, wie bei der COMP-Operation.

Beispiel: FELD1 CHANGFELD2

Nach Ausführung der Operation enthält Feld1 den Wert von Feld2 und Feld2 den Wert von Feld1.

Sonderfall: FELD1 CHANGFELD1

Nach Ausführung der Operation enthält Feld1 den Wert X'00'.

CHECK Dateistatus prüfen 3143

Im Faktor2 wird der Name der zu prüfenden Datei eingetragen. Eine Bezugszahl muss in Spalte 54-55 eingetragen sein. Dieser Schalter wird gesetzt, falls die Datei vom System her eröffnet ist. Ist der Schalter nicht gesetzt, so war die Datei geclosed. In Spalte 53 kann wie bei der OPEN Operation ein "V" gesetzt sein, so dass im Faktor2 ein 8-stelliger Feldname eingetragen wird, der den variablen Dateinamen enthält.

Über eine Bezugszahl in Spalte 56-57 kann geprüft werden, ob die Datei in der CICS-FCT eingetragen ist. Ist der Schalter gesetzt, so wurde die Datei nicht gefunden.

Will man auf Schalter verzichten, so kann man die Informationen 'NO' (not open) und 'NF' (not found) im internen Feld CPGFRC abfragen.

In Batch-Anwendungen wird 'NF' gesetzt, wenn im Hauptprogramm die Datei nicht als File definiert ist und wenn bei der Ausführung diese Datei noch nicht eröffnet wurde.

Für den Zustand OPEN DISABLED kann unterschieden werden, ob dafür OD oder Blank in CPGFRC zurückgemeldet wird (siehe CPGURSI2 im Handbuch CPG3-Installation). Der Default ist ab Release 2.5 OD.

In den Spalten 54-55 und 56-57 darf nicht die gleiche Bezugszahl eingetragen werden.

Beispiel:

```
C                     CHECKCPGWRK               5051
```

Ist nach Durchführen der Operation Schalter 50 gesetzt, so ist diese Datei zur Zeit eröffnet.
Ist nach Durchführen der Operation Schalter 51 gesetzt, so wurde die Datei in der CICS-FCT nicht gefunden.

CLEAR Feldinhalte löschen 3144

Diese Operation löscht alle alphanumerischen Felder auf Blank und alle numerischen Felder auf Null.

Wird in der H-Karte Spalte 27-30 (TWA des Vorprogramms) eine Eintragung vorgenommen, so wird dieser Bereich bei der CLEAR-Operation nicht gelöscht.

Die CLEAR-Operation darf nicht in einer Subroutine, nicht im Batch und nicht in HL1-Modulen aufgerufen werden.

Achtung: Wir empfehlen, vorher alle Dateien mit dem Befehl RNDOM*ALL freizugeben.

CLOSE Datei abschließen 3145

Die explizite CLOSE-Operation schließt eine Datei ab. Faktor 2 bezeichnet die abzuschließende Datei. In den Stellen 56 bis 57 kann eine Ergebnisbezugszahl angegeben werden, die bei nicht erfolgreichem Abschluss der CLOSE-Operation auf Ein gesetzt wird.

Soll auf den Schalter verzichtet werden, so kann im internen Feld CPGFRC 'NC' für 'not closed' und 'NF' für 'nicht in der FCT gefunden' bzw 'Nicht gefunden' abgefragt werden.

Dies erfolgt bis CICS Release 1.7, auch wenn ein CLOSE für Dateien benutzt wird, die nicht eröffnet sind.

```
Beispiel:           CLOSEDATEI                18
```

Ein 'V' in Spalte 53 bedeutet, dass in Faktor 2 ein Feldname eingetragen werden kann. Das Feld muss acht Stellen alpha definiert sein und kann mit variablem Datei-Namen aufbereitet werden.

```
Beispiel:           MOVEL'DATEI'     FELD  8
                    CLOSEFELD              V  18
```

CLRIN Bezugszahl mit Index löschen 3146

Diese Instruktion ist die Umkehrung der SETIN-Instruktion.

Das Ergebnisfeld enthält den Namen eines numerisch mit 0 Dezimalstellen definierten Indexfeldes und die Spalten 54 und 55 die erste Bezugszahl einer fortlaufenden Bezugszahlengruppe. Faktor 1 und 2 bleiben frei.

Nach Ausführung der Operation wird die Bezugszahl gelöscht, die sich aus der Addition der in den Spalten 54 und 55 eingetragenen Bezugszahl und dem Inhalt des Index

Feldes verringert um 1 ergibt.

Enthält das Ergebnisfeld keinen gültigen Index,so wird bei der Ausführung eine Fehlermeldung 'Index falsch' ausgegeben.

Ein I in Spalte 53 unterdrückt die Indexprüfung. Wenn das Ergebnisfeld gleich Null ist, wird die Fehlermeldung 'Index falsch' nicht angezeigt und die Verarbeitung fortgesetzt.

Beispiel: CLRIN I 25

Inhalt des Feldes 'I': 4

Nach Ausführung ist die Bezugszahl 28 gelöscht.

COMP Vergleichen 3147

Faktor1 und Faktor2 werden miteinander verglichen. Wenn Faktor1 größer, kleiner oder gleich Faktor2 ist, werden die Bezugszahlen in den Spalten 54-59 angesetzt (in der genannten Reihenfolge).

Vor Ausführung der Operation werden diese Bezugszahlen gelöscht.

Die zu vergleichenden Felder müssen entweder beide numerisch oder beide alphamerisch definiert sein.

Bei Alphafeldern wird in der Länge des kürzeren Operanden verglichen.

Die Operation COMP ist im Faktor 1 und im Faktor 2 voll indizierbar.

Unterschiedlich zu RPG kann bei der COMP-Operation im Ergebnisfeld der Name einer TAG-Operation eingetragen werden. In diesem Fall wird zu diesem TAG verzweigt, wenn eine der in den Spalten 54 bis 59 eingetragenen Bedingungen erfüllt ist.

Dabei können für diese Operation ausnahmsweise auch an Stelle einer Bezugszahl zwei Sterne eingetragen werden. Enthalten z.B. die Spalten 58,59 zwei Sterne, so wird unmittelbar zu dem im Ergebnisfeld eingetragenen TAG verzweigt, wenn Faktor 1 und 2 gleich sind.

Zur besseren Übersicht können statt der Bezugszahlen 01-99 zusätzlich in Spalte 54 - 55 '>>' oder 'GT', in Spalte 56-57 '<<' oder 'LT' und in Spalte 58 - 59 '==' oder 'EQ' eingetragen werden.

Beim COMP-Befehl sind die Einräge 'D', 'I' und 'K' für Datumsvergleiche in der Spalte 53 unterstützt.

Diese Funktionen sind bei der IF-Operation beschrieben.

COMRG Communication Region 3148

Die Operation COMRG stellt eine Communication Region zur Verfügung, die dem Programmierer den Zugriff auf interne Daten ermöglicht. Die Communication Region ist ein 32-stelliges Alphafeld, das wie folgt belegt ist:

Stelle	1 - 3	Operator Identification (Sign On Table)
	4 - 6	Cursor Position numerisch gepackt (170=Zeile 3 Position 11)
	7 - 10	TransId
	11 - 12	Anzahl Bildschirmz. Default (gepackt)
	13 - 14	Anzahl Bildschirmsp.Default (gepackt)
	15 - 16	Anzahl Bildschirmz. Alternate (gepackt)
	17 - 18	Anzahl Bildschirmsp.Alternate (gepackt)
	19 - 21	Tasknummer (gepackt)
	22 - 22	'U' = UCTRAN ein Übers.in Großbuchst. 'N' = NOUCTRAN aus Übers.erfolgt nicht
	23 - 26	CPG-intern
	27 - 27	UCTRAN Informationen. 'N' = NOUCTRAN aus Übers.erfolgt nicht 'T' = UCTRAN Transaktion (ab CICS 2.2) 'U' = UCTRAN ein Übers.in Großbuchst.
	28 - 32	Reserve

Das Ergebnisfeld muss den Namen eines 32-stelligen Feldes enthalten, das die Communication Area aufnimmt. Die einzelnen Felder können über eine SELCT-Operation ausgewählt werden bzw. durch Unterdefinition in der D- oder E-Karte.

Beispiel:

```
          D    FELD     0 32      GESAMT
          D    OPID        3      OPERATOR ID
          D    CURSOR      5 0    CURSOR POSITION
          D    TRANID      4      TRANSID
          D    DZEILE      3 0    ANZ. BILDSCHZ. DEF.
          D    DSPALT      3 0    ANZ. BILDSCHSP.DEF.
          D    AZEILE      3 0    ANZ. BILDSCHZ. ALTERN.
          D    ASPALT      3 0    ANZ. BILDSCHSP.ALTERN.
          D    TSKNO       5 0    TASK NUMMER
          D    UCTRN       1      UCTRAN BYTE
          D    FILLER      4      CPG-intern
          D    UCTRNE      1      UCTRAN BYTE erweitert
          D    REST        5      noch nicht belegt

          C          COMRG          FELD
```

2. COMRG mit dem internen Feld CPGSIN

Mit COMRG CPGSIN können erweiterte Systeminformationen abgerufen werden.

Ein grundsätzlicher Unterschied besteht in der Verarbeitungslogik: CPGSIN wird in der Eingabe als Feld wie für eine SELECT-Eingabe beschrieben. COMRG füllt automatisch die in der Eingabe beschriebenen Felder. Ein SELCT CPGSIN kann nicht codiert werden.

Einschränkung: COMRG mit CPGSIN führt nur in Command-Level-Programmen zu sinnvollen Ergebnissen.

```
ICPGSIN  F
I                                  1   8 APPLID
C              COMRG          CPGSIN
```

Folgende Parameter können abgerufen werden:

gep.	von	bis	dec.	Bezeichnung
	1	8		CICS Application Id
	9	12		CICS SYSID
	13	20		VTAM Net Name
	21	28		achtstellige User Id
	29	31		dreistellige Operator Id
	32	34		Hex: OPCLASS, Security-Klasse
	35	37		Hex: OPSECURITY, alter Security Key
	38	45		Hex: OPERKEYS, neuer Security-Key
P	46	48	0	Länge der Common Work Area
P	49	51	0	Länge der Transaction Work A.
P	52	54	0	Länge der Terminal User Area
	55	58		Terminal-Features COLOR,EXTDS, HILIGHT und PS
P	59	61	0	akt. Anzahl Bildschirmzeilen
P	62	64	0	akt. Anzahl Bildschirmspalten
	65	68		letzter ABEND Code
	69	70		QD = Start von Transient Data S = Start mit EXITI SD = Start mit EXITD TD = Start über Bildschirm-eingabe oder EXITT oder MAPD 'T'
	71	80		Wochentag im Klartext
P	81	81	0	Wochentag als Ziffer (Sonntag = 0, Samstag = 6)
P	82	84	0	Länge der Common Area
	85	86		Programmfunktionstaste
P	87	90	0	CICS Tasknummer
	91	94		Transaction Identification
P	95	102	0	Anzahl Sekunden seit 1.1.1900
	103	108		Datum JJ.TTT
	109	116		Datum JJ.TT.MM
	117	124		Datum JJ.MM.TT
	125	132		Datum TT.MM.JJ
	133	140		Datum MM.TT.YY
P	141	148	0	Anzahl Tage seit 1.1.1900
P	149	151	0	aktuelles Jahr
P	152	153	0	Zeile der Cursorposition
P	154	155	0	Spalte der Cursorposition

CONT Unterbrechen einer Schleife 3149

Eine DO-, DO UNTIL- oder DO WHILE-Schleife soll unterbrochen, aber entsprechend der Schleifenbedingung weiter durchlaufen werden.

Die Schleife wird entsprechend der Schleifenbedingung durchlaufen.

CONT verzweigt vor das END-Statement der Schleife. Die Statements zwischen CONT und ENDDO werden nicht ausgeführt, die Schleife wird aber entsprechend der programmierten Schleifenbedingung weiter durchlaufen.

CONVT Konvertieren eines alphanumerischen Feldes 3150

Beispiele:

```
C                     CONVT          A1
C                     CONVT A1       A2        X
```

Zweck:

Der Inhalt eines alphanumerischen Feldes soll konvertiert werden.

Beschreibung:

CONVT A1 übersetzt in A1 Kleinbuchstaben in Großbuchstaben.

Die Spalte 53 hat folgende Auswirkung:

' ' Übersetzt Klein- in Großbuchstaben.

S (Lower Case Translation) übersetzt Großbuchstaben in Kleinbuchstaben.

X übersetzt Character in ihre Halbbytes (EBCDIC-Darstellung), z. B. den Buchstaben A in C1.

C fasst je zwei Hexadezimalwerte zu einem Character zusammen, z. B. C1 zu A.

Wenn in Spalte 53 ein X oder ein C eingetragen wird, muss in Faktor 2 ein alphanumerisches Feld eingetragen sein.

Wird Faktor 2 und Ergebnisfeld eingetragen, so bleibt das Ursprungsfeld in Faktor 2 unverändert. Die Konvertierung wird linksbündig in der Länge des kürzeren Operanden durchgeführt.

Datumskonvertierung (Date und Year): (nur dt.Schreibw.)

Bei Date und Year müssen die beiden Operanden von gleicher Länge sein. Folgende Längen sind unterstützt:

```
alpha   6     num.   6,0 oder 7,0   Format:    (0)TTMMJJ
alpha   8     num.   8,0 oder 9,0   Format:  (0)TTMMJJJJ
alpha   8                           Format:     TT.MM.JJ
alpha  10                           Format:   TT.MM.JJJJ
```

D vertauscht Jahr und Tag, so dass die Jahreszahl nach der Operation vorne im Feld steht.

Y für Year wird benötigt, falls die Jahreszahl in den ersten vier Stellen des Feldes steht. 'Y' ist die Umkehrfunktion von 'D'. (Bei zweistelligen Jahreszahlen genügt die Servicefunktion 'D' für beide Konvertierungsrichtungen).

Datumskonvertierung von ISO-Format ins UDATE-Format.

V vertauscht Jahr und Tag, so dass die Jahreszahl nach der Operation hinten im Feld steht und bereitet das Feld mit '.' auf.

C	CONVTFAK2	ERGEB	V
Faktor 2:			
numerisch	8,0 und 9,0	Inhalt:	(0)JJJJMMTT
alphanumerisch	8	Inhalt:	JJJJMMTT
Ergebnis:			
alphanumerisch	8	Inhalt:	TT.MM.JJ
alphanumerisch	10	Inhalt:	TT.MM.JJJJ

Konvertierung: Uhrzeit <-> Sekunden ('W', 'Z')

Die Uhrzeit steht in einem Feld, das numerisch von 5,0 bis maximal 9,0 definiert ist. Die beiden letzten Stellen werden als Sekunden interpretiert, die vorletzten als Minuten und alle weiteren als Stunden. Die maximale Anzahl Sekunden, die bei der Konvertierung unterstützt ist, ist 359.999.999 in einem mit 9,0 definierten numerischen Feld.

W übersetzt eine Uhrzeit bzw. einen Wert (in Stunden, Minuten und Sekunden) in die Sekunden-Anzahl. Ist die Sekunden- oder Minutenanzahl größer als 59, so wird das Ergebnisfeld auf Null gesetzt.

Z übersetzt eine Sekunden-Anzahl in einen Zeit-Wert.

Faktor 2 ist optional. Er muss jedoch bei den Services 'X', 'C', 'W' und 'Z' vorhanden sein.

COPY Assembler-Unterprogramm einfügen 3151

Die Instruktion entspricht der COPY-Instruktion des Assembler. Sie erlaubt dem Programmierer, katalogisierte Assembler-Quellenmodule in das Programm einzufügen. Faktor 2 enthält den Namen des Quellenmoduls und darf maximal 8 Stellen lang sein.

Beispiel: COPY ASSBOOK

Das in der Source-Library katalogisierte Assembler-Book wird an dieser Stelle in das CPG-Programm eingefügt.

DEBUG Test-Hilfe 3160

Dieser Operationsschlüssel erzeugt auf dem Bildschirm ein Testbild, das dem Programmierer den Status des Programms an dieser Stelle mitteilt. Faktor1, Faktor2 und Ergebnisfeld sind blank. (Siehe 2810).

Die DEBUG-Funktion wird auch von der interaktiven Testhilfe QDF (Quick Debugging Facility) erfüllt. DEBUG verliert somit als CPG-Operation an Bedeutung.

DEBUG ON/OFF ist eine Compiler-Anweisung, die bewirkt, dass Programmteile vom interaktiven Test mit QDF ausgeschlossen werden können.

Diese Funktion ist zum Beispiel dann vorteilhaft, wenn große Programme getestet werden sollen, die infolge der Option DEBUG so groß werden, dass die Assemblierung Addressability Errors bringt. Für den verriegelten Programmteil wird kein Mehr-Code generiert.
Nur in Verbindung mit 'S' in Spalte 46 der H-Karte.

DELC Zeichen entfernen 3161

Mit dieser Operation kann ein Zeichen aus einem alphanumerischen Feld oder einem Feldgruppenelement entfernt werden.Dabei werden alle folgenden Zeichen um eine Stelle nach links verschoben und in der letzten Stelle wird ein Blank eingefügt.

Faktor 2 enthält das zu entfernende Zeichen in Hochkommata und das Ergebnisfeld den Namen des Feldes. Faktor 1 bleibt frei.

```
Beispiel:          DELC '*'      FELD
Feldinhalt vorher:                (A*B*C)
Feldinhalt nachher:               (ABC  )
```

Faktor 2 kann auch einen hexadezimalen Direktwert in der

Form X'00' enthalten.

Im Ergebnisfeld kann auch eine Feldgruppe ohne Index eingetragen werden.

DELET Satz löschen 3162

Mit der DELET-Operation wird ein Satz in einem Datenbestand gelöscht. Der gelöschte Satz kann nicht wieder aufgefunden werden.

In Faktor 1 kann ein Schlüsselfeld eingetragen werden, dessen Inhalt den zu löschenden Satz bestimmt.
Enthält Faktor 1 keinen Eintrag, so löscht die DELET-Operation den gerade verarbeiteten Satz. In Faktor 2 muss der Dateiname enthalten sein, von der der Satz gelöscht werden soll.
Es ist auch möglich, einen Satz, der durch eine CHAIN-U Operation gelesen wurde, mit der DELET-Operation zu löschen.

DEQ Programmteil entriegeln 3163

Bei dieser Operation wird ein mit ENQ gesperrter Programmteil wieder entriegelt. (Siehe ENQ).

Beispiel: X DEQ

Ein 'E' in Spalte 53 bedeutet, dass eine mit ENQ gesperrte externe Speicheradresse wieder entriegelt wird.

DIV Dividieren 3164

Faktor1 wird durch Faktor 2 dividiert. Nach Ausführung der Instruktion steht der Quotient im Ergebnisfeld.

Diese Operation ist indizierbar. Beispiel siehe 'ADD'.

Bei der Division ist zu beachten, dass die Dezimalstellenanpassung zu fehlerhaften Ergebnissen führen kann, wenn das Ergebnisfeld nicht die erforderliche Größe aufweist.

Bei der Durchführung der Operation erfolgt eine automatische Dezimalstellenanpassung. Für diese Anpassung gelten dieselben Restriktionen für die Feldlänge wie beim RPG, das heisst:

L1 + (D2 - D1 + DE) muss kleiner oder gleich 15 sein
L2 + (D2 - D1 + DE) muss kleiner oder gleich 15 sein

wobei L1 = Länge Faktor 1
L2 = Länge Faktor 2

D1 = Dezimalstellen Faktor 1
D2 = Dezimalstellen Faktor 2
DE = Dezimalstellen Ergebnisfeld

Die Stellenzahl des Ergebnisfeldes muss errechnet werden um sicher zu stellen, dass dieses Feld auch das gesamte Ergebnis aufnehmen kann. Dazu wird sinnvollerweise zuerst die Anzahl der gewünschten Dezimalstellen (DE) bestimmt. Die Länge des Ergebnisfeldes errechnet sich dann nach folgender Formel:

LE = DE + (L1 - D1) + D2

Beispiel: Das Ergebnis soll 2 Dezimalstellen haben

```
111,2    DIV  222,11    ERG     72
LE = 2 + (4 - 1) + 2 = 7
```

Wenn bei der Division gerundet werden soll, so gilt für Faktor 1 die Formel:

L1 + (D2 - D1 - DE) muss kleiner oder gleich 14 sein.

Wenn im Anschluss an die Division die Operation 'MVR' benutzt wird, darf nicht gerundet werden.
Wird eine Division mit Null durchgeführt, so erscheint am Bildschirm die Fehlermeldung: Division durch 0.
Wurde gegen die Regeln bezüglich Feldlängen verstoßen, so dass kein richtiges Ergebnis errechnet werden kann, erscheint die Fehlermeldung: Divisionsfehler.

DLI DL/I-Call aufrufen 3165

Diese Operation führt den Call zur DL/I-Datenbank durch.
In Faktor1 wird ein gültiger DL/I-Call eingetragen, z.B. GU,GN,GHU,ISRT,...). Faktor 2 bleibt normalerweise frei oder enthält in Hochkommata den PSB-Namen, wenn in einer Task/Programm auf verschiedene PSBs zugegriffen wird.
Im Ergebnisfeld wird die Datenbeschreibung der DL/I eingetragen. Das Feldlängendefinitionsfeld (Spalte 49-51) enthält die PCB-Nummer. Unter Bezugszahl größer muss ein Schalter eingetragen sein, der gesetzt wird, wenn der Call nicht ordnungsgemäß verlaufen ist. Der Schalter wird gesetzt, wenn der Returncode von DL/I nicht gleich " ", oder "GA" oder "GK" gesetzt wird. Dieser Returncode kann über das interne Feld CPGDRC abgefragt werden.

Durch die Operation DL/I werden alle aufgebauten SSAs gelöscht. Für jeden weiteren DL/I-Aufruf sind die SSAs mit den Operationen QSSA oder USSA neu zu setzen.

```
Beispiel:  GHU        DLI           ARTIRT  3  10
          10CPGDRC    CABNE"  "     DLIERR
```

Wird in Spalte 53 ein 'V' eingetragen, so werden sämtliche Einträge dem Feld CPGDLV entnommen. (Vgl. 2335)

DO Schleife 3166

Die Operation DO erlaubt es, eine Gruppe von Rechenbestimmungen (eine DO-Gruppe) mehrere Male auszuführen. Faktor 1 kann ein numerisches Feld ohne Dezimalstellen sein, das den Anfangswert enthält.

Faktor 2 kann ein numerisches Literal, Feld oder Feldgruppenelement ohne Dezimalstellen sein, das den Begrenzungswert enthält.

Im Ergebnisfeld kann ein Name für ein numerisches Feld eingetragen werden. Dieses Feld enthält dann den laufenden Index.

Ein Ergebnisfeld muss nicht spezifiziert werden. Fehlt das Ergebnisfeld, so steht der Index im Feld CPGDxx. Ist Faktor 2 nicht spezifiziert, ist der Begrenzungswert 1.

Ein 'L' in Spalte 53 bedeutet, dass die DO-Schleife immer weiter durchlaufen wird. (Loop).

Es steht die Spalte 53 analog zur IF-Operation zur Verfügung.

Siehe auch Kapitel Strukturierte Programmierung (2450).

(beim ENDDO ist I = 6 wenn DO 5 I)

DOU Ausführen ... bis 3167

Die DOU-Operation erlaubt es, dass eine Gruppe von Rechenbestimmungen einmal oder mehrfach ausgeführt wird, bis eine bestimmte Relation zwischen Faktor1 und Faktor2 besteht.

Eine DOU-Operation mit einer zugeordneten END-Operation bildet eine DO-Gruppe. Bedingende Bezugszahlen können verwendet werden. Faktor1 und Faktor2 müssen entweder ein alphanumerisches oder ein numerisches Feld oder einen Feldnamen enthalten.

Die Eintragungen in Faktor1 und Faktor2 müssen vom gleichen Typ sein, entweder beide alphanumerisch oder beide numerisch.

Der Vergleich von Faktor1 zu Faktor2 folgt den gleichen Regeln wie für die Vergleichsoperationen.

Die DOU-Operation ist bei alphanumerischem Vergleich in einem Faktor, sonst in beiden Faktoren indizierbar.

Eine '1' in Spalte 53 bedeutet, dass die DOU-Schleife mindestens einmal durchlaufen wird.

Siehe auch Kapitel Strukturierte Programmierung (2450).

DOW Ausführen ... während 3168

Die DOW-Operation erlaubt es, eine Gruppe von Rechenbestimmungen solange auszuführen, wie eine bestimmte Beziehung zwischen Faktor 1 und Faktor 2 besteht.

Eine DOW-Operation mit einer zugeordneten END-Operation bildet eine DO-Gruppe.

Bedingende Anzeiger/Bezugszahlen können spezifiziert werden. Faktor 1 und Faktor 2 müssen entweder ein alphanumerisches oder ein numerisches Feld oder einen Feldnamen enthalten.

Die Eintragungen in Faktor 1 und Faktor 2 müssen gleich sein, entweder beide alphanumerisch oder beide numerisch. Der Vergleich von Faktor 1 und Faktor 2 folgt den gleichen Regeln wie für die Vergleichsoperationen.

Die DOW-Operation ist bei alphanumerischem Vergleich in einem Faktor, sonst in beiden Faktoren indizierbar.

Siehe auch Kapitel Strukturierte Programmierung (2450).

DSPLY Ausgabe auf der Konsole 3170

Die DSPLY-Operation ermöglicht dem Programm, mit der Systemkonsole zu kommunizieren. In Faktor 1 und Ergebnisfeld ist wahlweise ein alphanumerisches Feld einzutragen. In der CICS-Version muss Faktor 1 eine Konstante enthalten, eine Eintragung im Ergebnisfeld ist nicht unterstützt.

Enthält Faktor 1 einen Feldnamen und das Ergebnisfeld ist blank, wird der Inhalt des Feldes in Faktor 1 auf der Konsole angezeigt.

Ist Faktor 1 blank und das Ergebnisfeld enthält einen Eintrag, wird der Inhalt des Ergebnisfeldes angezeigt.

Enthält sowohl Faktor 1 als auch das Ergebnisfeld einen Eintrag, so werden die Inhalte beider Eintragungen nacheinander angezeigt. In HL1-Batchprogrammen wird das Programm an dieser Stelle unterbrochen und das Programm erwartet eine Antwort. Die Antwort wird im Ergebnisfeld abgestellt.

Wird keine Antwort eingegeben, bleibt das Ergebnisfeld unverändert.

```
     C           NACHR     DSPLY
     C                     DSPLY        ANTW
     C           NACHR     DSPLY        ANTW
```

DUMP Dump ausgeben 3171

Im CICS erzeugt diese Operation einen Transaction Dump auf der Platten-Einheit, die den 'CICS-Dump-File'-Bereich enthält. Der DUMP muss mit einem separaten Programm gedruckt werden. Die Operation kann mehrmals im Programm vorkommen. Zur Unterscheidung verschiedener Dumps wird in Faktor 2 ein vierstelliger Dumpcode eingetragen.

Beispiel: DUMP NR11

Im Batch wird ein formatierter Dump gedruckt. Durch die DUMP-Operation wird das Programm nicht beendet.

EDIT Feld aufbereiten 3180

Mit dieser Operation kann ein Feld über die Ausgabebestimmungen aufbereitet werden. Das Ergebnisfeld enthält den Namen des aufzubereitenden Feldes. Faktor 1 ist blank.

Das Ergebnisfeld kann auch den Namen einer Feldgruppe mit festem oder variablem Index enthalten.

Diese Operation gilt nur für alphanumerische Felder.

Beispiel: EDIT FG,I

Dabei ist FG der Name einer alphanumerisch definierten Feldgruppe und I der Name des numerisch mit 0 Dezimalstellen definierten Indexfeldes, das den variablen Index enthält.

Wird eine EDIT-Operation mit einem Namen in Faktor 2 ausgeführt, so kann eine bestimmte Ausgabe-Satzbestimmung gezielt angesprochen werden.

```
C         EDIT FELD      ALPHA

OALPHA  F     FELD
```

In Verbindung mit einem 'V' in Spalte 53 kann in Faktor2 ein Feldname (6-stellig-alpha) eingetragen werden.

```
C         MOVEL'FELD'    NAME
C         EDIT NAME      ALPHA     V

OALPHA  F     FELD
```

Zu beachten ist, dass alle Ausgabe-Satzbestimmungen eines Feldes in den O-Karten nacheinander beschrieben werden.

In Spalte 53 kann ein 'I' eingetragen werden. Dies bewirkt, dass keine Prüfung des benutzten Indexes durchgeführt wird. Der Programmierer trägt dann die Verantwortung dafür, dass kein Speicherbereich irrtümlich überschrieben wird.

In den Spalten 54 bis 59 können bis zu drei Bezugszahlen eingetragen werden. Diese Bezugszahlen werden jeweils vor Ausführung der EDIT-Operation wie mit einer SETON-Operation gesetzt und nach Ausführung der EDIT-Operation wie mit einer SETOF-Operation wieder gelöscht.
Siehe auch (2420).

ELIM Zeichen löschen 3181

Mit dieser Operation kann ein beliebiges Zeichen in einem alphanumerischen Feld durch Blank ersetzt werden.

Faktor 2 enthält das zu löschende Zeichen in Hochkommata und das Ergebnisfeld den Namen des Feldes. Faktor 1 bleibt frei.

Faktor 2 kann auch einen Direktwert in hexadezimaler Darstellung (X'00') enthalten.

Das Ergebnisfeld kann den Namen eines Alphafeldes, einer alphanumerischen Feldgruppe oder eines Feldgruppenelements enthalten.

```
Beispiel:            ELIM '-'      FELD

Feldinhalt vorher:    (A-B-C)     (Hugo--------)
Feldinhalt nachher:   (A B C)     (Hugo        )
```

ELSE Sonst ... ausführen 3182

Die ELSE-Operation kann wahlweise mit der IF-Operation verwendet werden.

Die ELSE-Operation wird unmittelbar nach den Rechenbestimmungen eingetragen, die ausgeführt werden, wenn der Vergleich in der IF-Operation positiv ist.

Nach der ELSE-Anweisung folgen die Rechenbestimmungen, die ausgeführt werden sollen, wenn der Vergleich in der IF-Operation negativ ist.

Siehe auch Kapitel Strukturierte Programmierung. (2463).

END Ende 3183

Die END-Anweisung zeigt das Ende einer CAS-, DO-, DOW-, DOU- oder IF-Gruppe an. Es kann wahlweise auch die Operation ENDDO bzw. ENDIF eingesetzt werden.

Faktor 2 ist der Erhöhungswert der DO-Operation und kann ein numerisches Literal (Ziffer 1-9) oder ein numerisches Feld mit 0 Dezimalstellen enthalten. (Ein numeri-

sches Feld kann Minus-Werte enthalten).

Ist bei einer DO-Operation in Faktor 2 keine Eintragung vorgenommen worden, so ist der Erhöhungsfaktor 1.

Bezugszahlen in Spalte 10 - 18 können nicht eingetragen werden, wenn die END-Operation zu einer IF-Operation gehört.

Bezugszahlen beim END-Statement einer DO-Schleife bewirken, dass nur zum Start verzweigt wird, falls die Bedingung noch erfüllt ist.

ENDDO DOxxx Ende 3184

Die ENDDO-Anweisung zeigt das Ende einer DO-, DOW- oder DOU-Gruppe an. Sie kann anstelle der END-Anweisung benutzt werden, um das Programm besser zu dokumentieren. In Faktor 2 kann auch ein numerisches Feld mit 0 Dezimalstellen eingetragen werden, das den Erhöhungswert beinhaltet. Das numerische Feld unterstützt positive und negative Werte.

ENDAS Ende Assembler 3185

Diese Operation besagt, dass ab hier der Assembler-Modus endet und das Programm in CPG fortgeführt wird. (Siehe BEGAS).

ENDDT Ende Entscheidungstabelle 3186

Siehe auch Entscheidungstabellen (2410).

ENDIF IFxx Ende 3187

Die ENDIF-Anweisung zeigt das Ende einer IF-Gruppe an. Sie kann anstelle der END-Anweisung benutzt werden, um das Programm besser zu dokumentieren.

ENDSR Ende Unterprogramm 3188

Das Programm verzweigt ab hier zum Aufrufbefehl (EXSR) zurück. Faktor 1 darf einen Namen enthalten.

ENQ Programmteil sperren 3189

CPG-Programme sind reentrant, d.h. ein Programm kann gleichzeitig von verschiedenen Terminals benutzt werden. Dies bedeutet, dass ein Plattensatz gleichzeitig von verschiedenen Stellen verändert werden kann.

Die Operation ENQ erlaubt dem Programmierer, zwischen dem Lesen und Zurückschreiben des Satzes die Benutzung eines Programmabschnitts solange zu sperren, bis der Update-Vorgang abgeschlossen ist.

Die Entriegelung erfolgt durch den Befehl 'DEQ'.

Beispiel: ENQ X

Von dieser Stelle ab wird das Programm für andere Benutzer solange gesperrt, bis der laufende Benutzer die Stelle X erreicht hat.

Ein 'E' in Spalte 53 bewirkt, dass eine externe Speicheradresse gesperrt werden soll, z.B. eine CWA-Adresse für verschiedene Benutzerprogramme.

Werden z.B. im User Copy-Buch A.CPGUCCUA die Felder

```
C1        DS      C
C2        DS      C
C3        DS      C
```

definiert, kann im Programm ein Schutz auf Systemebene erreicht werden.

Programm 1

```
C           ENQ  C1              E
            .
C     C1    DEQ                  E
```

Programm 2

```
C           ENQ  C1              E
            .
C     C1    DEQ                  E
```

EREAD Erweitertes Read 3190

Diese Operation ist eine Kombination der Operationen 'SETON', 'EXCPT', 'READ ' und 'SETOF' . Sie gilt nur für Bildschirmdateien. Sie bewirkt das Vorformatisieren und Lesen der Bildschirmnachricht in einer Operation.

Beispiel: EREADBILD 10

ist gleichbedeutend mit:

```
                    SETON                10
                    EXCPT
                    READ BILD
                    SETOF                10
```

Ein 'I' in Spalte 53 bewirkt, dass nach der READ-Operation der Schalter CL abgefragt werden kann. Andernfalls bewirkt die Clear-Taste die Beendigung des Programms. Die Clear-Taste löscht den Bildschirm (Terminal Erase).

Ein 'L' in Spalte 53 bewirkt, dass bei der Read Operation Kleinbuchstaben nicht automatisch in Großbuchstaben übersetzt werden.

Ein 'S' in Spalte 53 schließt I und L ein.

Für 'quasi-transaktionsorientiertes' Programmieren ste-

hen vier weitere Einträge in Spalte 53 zur Verfügung:

T für die transaktionsorientierte Programmierweise
A impliziert T und L
C impliziert T und I
K Impliziert T und S
A und K setzen UCTRAN OFF voraus.

Hinweis: Diese 'quasi-transaktionsorientierte' Programmierweise unterliegt einigen Einschränkungen. Sie sollte nur nach ausreichender Schulung angewandt werden.

Tabelle der Einträge in Spalte 53

Spalte 53		A	C	I	K	L	S	T
Task		x	x		x			x
Lower Case		x			x	x	x	
CL-Abfrage			x	x	x		x	

EXCPT Ausgabe 3191

Diese Operation entspricht einer Verzweigung in die Ausgabe-Bestimmungen. Die Ausgaben für alle Dateien werden ausgeführt.

Faktor 1 und Ergebnisfeld bleiben frei.

Beispiel: EXCPT

In den Spalten 54 bis 59 können Ergebnisbezugszahlen gesetzt werden.

Beispiel: EXCPT 25

Diese Instruktion erzielt den gleichen Effekt wie die drei Instruktionen:

```
                    SETON                   25
                    EXCPT
                    SETOF                   25
```

Wird eine EXCPT-Operation mit einem Namen in Faktor 2 ausgeführt, so werden die gesteuerten Ausgabezeilen mit dem gleichen EXCPT-Namen geprüft und ausgegeben, wenn die Prüfung der bedingenden Bezugszahlen positiv ist.

```
C                   EXCPTAUSG1
ODATEI      E           AUSG1
```

In Verbindung mit einem 'V' in Spalte 53 kann in Faktor2 ein Feldname (6-stellig-alpha) eingetragen werden).

```
C                   MOVEL'AUSG1'    FELD    6
C                   EXCPTFELD                 V
ODATEI      E           AUSG1
```

EXHM EXECUTE HL1-Modul 3192

Mit dieser Operation wird ein HL1-Baustein oder eine Baugruppe aufgerufen. Faktor 2 muss den Namen des Bausteins oder der Baugruppe enthalten. Dieser Name muss in der HL1-Tabelle enthalten sein. Ist 'PROG' der Name des Bausteins, so muss eine Programmphase mit dem Namen 'HMPROG' in der Core Image Library enthalten sein. Ist '$PROG' der Name des Bausteins, so muss eine Programmphase mit dem Namen 'HMXPROG' in der CIL enthalten sein, wenn das Programm von einem Bildschirm aufgerufen wird.

Beispiel:

```
C                       EXHM $PROG
```

Das Ergebnisfeld kann den Namen eines Datenkanals enthalten. In diesem Falle muss der Datenkanal in den Eingabebestimmungen beschrieben sein. Beim Aufruf des Bausteins werden dann alle unter dem Datenkanal beschriebenen Felder in die private Transaction Work Area des Bausteins übertragen und nach Ausführung des Bausteins werden die veränderten Feldinhalte wieder in die Ausgangsfelder zurück übertragen.

Beispiel:

```
C                       EXHM PROG       KANAL
```

Bei variabler Verarbeitung (Spalte 53=V) ist diese Form der Datenübergabe nicht unterstützt. Sollen bei variablem EXHM Daten zwischen rufendem und gerufenem Modul ausgetauscht werden, so muss sich das aufgerufene Modul die Daten beim rufenden abholen. Dazu muss ein Eintrag in der H-Karte Spalte 34 A,E oder T des gerufenen Programms gesetzt werden. Dann werden Daten zwischen Feldern mit gleichem Namen und gleichen Feldeigenschaften ausgetauscht.

Achtung: Ist das rufende Programm mit AMODE=31 (ESA) gelinkt,so muss auch das gerufene Modul mit AMODE=31 katalogisiert sein.

In Faktor 2 wird der Name eines achtstelligen Feldes angegeben. Dieses enthält in den ersten sechs Stellen den Namen des auszuführenden Bausteins und in Stelle 7 gegebenenfalls die private HL1-Library. Stelle 8 ist frei.

Die Servicefunktion 'V' beinhaltet die Servicefunktion 'T', die im folgenden erklärt ist.

Enthält die Spalte 53 ein 'I', so erhält das aufrufende Programm für die so initialisierte Baugruppe den Status eines Hauptprogramms, das heisst, dass die Operationen GETMI und PUTMI und die Operationen EDIT und SELCT CPGTWA sich für alle Bausteine auf einer der folgenden Stufen auf die Bezugszahlen oder die TWA des Programms beziehen, in dem die Baugruppe initialisiert wurde. Bei Betätigung der Löschtaste einer Bildschirmtastatur wird in diesem Fall die Verarbeitung mit der Operation fortgesetzt, die der EXHM-Operation folgt.

Ein 'T' in Spalte 53 arbeitet wie der Eintrag 'I' mit dem Unterschied, dass die Programmfunktionstasten nach der EXHM-Operation abgefragt werden können.

Beispiel:

```
C                   EXHM PROG      KANAL     I
C                   EXHM PROG      KANAL     T
C                   EXHM PROG      KANAL     V
```

EXITD Direkt verzweigen mit Daten 3193

Die Operation ist eine Erweiterung der Operation EXITI. Im Ergebnisfeld wird der Name einer Datenstruktur eingetragen, die folgenden Aufbau hat:

In den Stellen 1-4 die TransId der Folgetask.

In den Stellen 5-8 den Namen des Terminals, an dem die Task gestartet wird. Ist in den Stellen 5-8 Blank eingetragen, so wird die Folgetask am gleichen Terminal ausgeführt.

In den Stellen 9-12 entweder ein Intervall gepackt oder die Uhrzeit, zu der die Task gestartet werden soll. Das Format der Zeit ist 0HHMMSSC.

In den Stellen 13-20 einen frei wählbaren TS-Namen, der falls blank vom TP-Monitor selbst vergeben wird.

In den Stellen 25-nn die Daten, die übergeben werden.

Mit der Operation EXITD können gleichzeitig Daten an eine Folgetask übergeben werden. Die Folgetask kann diese Daten über eine READ-Operation auf "$CPG", einer CPG-internen TS Queue, lesen. Sind keine Daten vorhanden, so wird der Schalter EF gesetzt. Die Daten werden beim READ $CPG, spätestens beim Ende der Task gelöscht.

Ein 'T' in Spalte 53 sagt aus, dass in der Datenstruktur eine Uhrzeit und kein Intervall gesetzt ist. Beispiel:

```
IFELDX      DS
I                                        1   4 TRANID
I                                        5   8 TERMID
I                                    P   9  120INTRVL
I                                    P   9  120TIME
I                                       13  20 REQID
I*   wird intern von CPG gefüllt       21  24 LLBB
I                                       25  78 DATA
C                    MOVEL'MENU'    TRANID
C                    MOVEL'RZ01'    TERMID
C                    Z-ADD12500     INTRVL
C                    MOVEL'z/VSE'   DATA
C                    EXITD          FELDX
```

Die Operation EXITD startet eine Task MENU am Terminal RZ01 in 1 Stunde und 25 Minuten und übergibt die Mel-

dung "z/VSE".

Die Folgetask kann diese Meldung mit folgenden Befehlen lesen:

```
I$CPG    KF
I                                        1  54 DATA
C                     READ $CPG
C   EF                GOTO NOTFND
C           DATA      CABEQ'z/VSE'   OK
```

Unter $CPG wird eine Pseudo-TS-Queue gelesen, wenn nicht vorhanden, so wird der Schalter EF gesetzt. Ansonsten prüft das Programm, ob die Daten mit "z/VSE" beginnen und verzweigt nach OK.

Beachte: Nach dem EXITD wird das aufrufende Programm nicht verlassen, sondern bis zum letzten Statement abgearbeitet.

EF wird gesetzt, wenn entweder die aufgerufene Task oder der angesprochene Bildschirm nicht in der entsprechenden CICS-Tabelle liegt.

EXITI Direkt verzweigen 3194

Diese Operation arbeitet wie Fall 2 der EXITP Operation, das heisst, Faktor 2 muss den Transaktions-Schlüssel in Hochkommata enthalten. Die Operation EXITI verzweigt dann direkt ins Folgeprogramm, während bei der Operation EXITP die Verzweigung erst durch Betätigung einer Programmfunktionstaste erfolgt.

Ein durch EXITI aufgerufenes Programm liest beim Start keine Daten vom Bildschirm ein.

Beispiel: EXITI'TRID'

Die Verarbeitung wird mit dem Programm fortgesetzt, das mit dem Transaktionsschlüssel TRID in der Programmtabelle des Steuerprogramms enthalten ist. Eine Rückkehr zum aufrufenden Programm erfolgt nicht.

Faktor 2 kann ebenfalls den Namen einer 4-stelligen Variablen enthalten, wenn der Programmierer sicherstellt, dass dieses Feld zur Ausführungszeit den Namen einer gültigen Trans-Id enthält. Die Verarbeitung wird dann mit diesem Programm fortgesetzt.

Beispiel: EXITITRID

Faktor 1 kann eine 4-stellige Konstante oder den Namen eines 4-stelligen alphanumerisch definierten Feldes enthalten, wenn der Programmierer sicherstellt, dass das Feld einen gültigen Terminal-Namen enthält. In diesem Fall wird die Verarbeitung an dem Terminal mit diesem Namen fortgesetzt.

Beispiel: 'BS01' EXITITRID

Das Ergebnisfeld kann eine vierstellige Konstante oder den Namen eines siebenstelligen numerisch mit null Dezimalstellen definierten Feldes enthalten. Der Inhalt der Konstanten wird dann als Zeitverzögerung in Stunden und Minuten und der Inhalt des Feldes in der Form 0SSMMSS ebenfalls als Zeitverzögerung in Stunden, Minuten und Sekunden angenommen. Eine Eintragung '0005' bedeutet z. B., dass das Programm in 5 Minuten gestartet wird.

Beispiel: TERM EXITITRID '0005'

Enthält die Spalte 53 ein 'T', so wird der Inhalt des Ergebnisfeldes als Uhrzeit angenommen. Der Programmierer muss dabei sicherstellen, dass das Feld eine gültige Uhrzeit enthält.

Beispiel: TERM EXITITRID '1230' T

Eine Eintragung 'N' in Spalte 53 bedeutet, dass das laufende Programm durch die EXITI-Operation nicht verlassen wird, sondern dass das aufgerufene Programm erst dann ausgeführt wird, wenn das laufende Programm beendet ist.

Ein 'S' in Spalte 53 schließt T und N ein.

Termid = 'NONE' bedeutet Nonterminal Task. In diesem Fall läuft das Programm im Hintergrund ab und benötigt zur Ausführung keinen Bildschirm.
Die CPGTCT kann bei einer Nonterminal-Task nicht verwendet werden.
EF wird gesetzt, wenn entweder die aufgerufene Task oder der angesprochene Bildschirm nicht in der entsprechenden CICS-Tabelle liegt.
In einer Nonterminal-Task wird bei EXITI die Verarbeitung mit dem nächsten Statement fortgesetzt. Spalte 53 hat dann keine Bedeutung.

EXITP Verzweigen in externes Programm 3195

Diese Operation ruft ein anderes in der Programm-Tabelle des TP-Monitors definiertes Programm auf. Der Phasenname des Programms wird in Faktor 2 eingetragen.

In Faktor 2 steht der Phasenname des Programms, in welches verzweigt werden soll. In diesem Fall bleibt die Task erhalten, lediglich das Programm wird ausgetauscht. Sollen Daten übernommen werden, so sind diese in der D- oder E- Karte deckungsgleich in beiden Programmen zu beschreiben. Siehe dazu das Kapitel Programmverbindungen.

Es ist darauf zu achten, dass keine Schleifen entstehen, da diese zu erheblichen Performance-Verlusten führen können.

Diese Operation entspricht dem EXEC CICS XCTL.

Wird in Spalte 53 ein "S" eingetragen, so wird die TWA auf eine CPG-interne Temporary Storage Queue zwischengespeichert und im Folgeprogramm in der definierten Länge

(H-Karte Stelle 27-30) übernommen. Das ist dann wichtig, wenn in einem Taskzyklus mehrere Programme mit EXPR sowie EXITP mit Programmname verknüpft sind (siehe EXPR).

Beispiel: EXITPPROG02 S

Die Operation EXITP mit Programmnamen schließt eine RNDOM*ALL Operation ein. (Siehe H-Karte Spalte 33).

Hinweis:

Wird aus einem Nicht-CPG-Programm mit XCTL oder LINK in ein CPG-Programm verzweigt, so können sich Probleme beim Programmstart ergeben. Deshalb sollte im Nicht-CPG-Programm die TIOA mit einem Macro DFHSC TYPE=FREEMAIN, RELEASE=ALL freigegeben werden. Ist dies nicht möglich, dann sollte im CPG-Programm eine Dummy-Bildschirmdatei (F-Karte) definiert werden und eine Eingabe-Satzbestimmung für diese Datei unmittelbar vor der ersten Rechenbestimmung codiert werden.

Bei Programmverbindungen (EXITP, EXPR) in ESA-Umgebungen größer 16MB müssen alle verbundenen Programme auf der gleichen Seite der 16MB Linie liegen.

Im Faktor2 kann auch in Hochkommata eingeschlossen die Trans-Id einer Folgetask gesetzt werden. Nach Betätigen einer Programmfunktionstaste wird dann diese Task gestartet. Im wesentlichen entspricht diese Funktion der EXITT-Operation, mit dem Unterschied, dass MDT-Bits am Bildschirm gelöscht werden.

Beispiel: EXITP'TRID'

Im Ergebnisfeld kann der Name einer 4-stelligen (bei TransId) oder 8-stelligen (bei Programmname) Variablen eingetragen werden, wenn der Programmierer sicher stellt, dass dieses Feld zur Ausführungszeit eine gültige TransId bzw. einen gültigen Programmnamen enthält.

In einem solchen Fall muss Faktor 2 Blank sein. Ansonsten entsprechen die Regeln einer Eintragung im Faktor 2.

EF wird gesetzt, wenn das aufgerufene Programm nicht in der PPT liegt.

EXITT Verzweigen für READfreie Programme 3196

Diese Operation arbeitet wie die EXITP-Operation, jedoch muss Faktor 2 einen Transaktionsschlüssel enthalten.

Der Unterschied zur EXITP-Operation besteht darin, dass die Tastatur nach Ausführung der Operation nicht entriegelt und vom Programm gesetzte Modified Data Tags nicht gelöscht werden.

Diese Operation ist immer dann zu verwenden, wenn bei READ-freier (taskorientierter) Programmierung Daten auf dem Bildschirm zwischengespeichert werden bzw. gelesen werden sollen.

Im Ergebnisfeld kann der Name einer 4-stelligen Variablen eingetragen werden, wenn der Programmierer sicher stellt, dass dieses Feld zur Ausführungszeit eine gültige Trans-Id enthält.

In einem solchen Fall muss Faktor 2 Blank sein. Ansonsten entsprechen die Regeln einer Eintragung im Faktor 2.

EF wird gesetzt, wenn das aufgerufene Programm nicht in der PPT liegt.

EXPR Programm ausführen 3197

Mit diesem Befehl kann an dieser Stelle des Programms ein anderes CPG- (oder CICS-) Programm ausgeführt werden. Nach Beendigung der Ausführung wird die Verarbeitung mit der nächsten Instruktion fortgesetzt.

Voraussetzung ist, dass die TWAs des aufrufenden und des aufgerufenen Programms übereinstimmen. Faktor 2 enthält den Phasennamen des auszuführenden Programms.

Es ist darauf zu achten, dass keine Schleifen entstehen, da diese zu erheblichen Performance-Verlusten führen können. Diese Operation entspricht dem EXEC CICS LINK.

Beispiel: EXPR PROGRM

Ein "S" in Spalte 53 der Operation EXPR rettet die TWA des aufrufenden Programms auf Temporary Storage in eine CPG-interne Queue. Diese TS-Queue ("TERM$CPG") sollte vom Anwendungsprogrammierer nicht verwendet werden. Die H-Karte des aufgerufenen Programms bestimmt, wieviele Stellen aus der Transaction Work Area übernommen werden.

Kehrt die Kontrolle in das rufende Programm zurück, so wird die gleiche Anzahl in die TWA des aufrufenden Programms zurückübertragen,die zuvor übernommen wurde.

Die Bezugszahlen werden ebenfalls aus dem aufgerufenen Programm übernommen. Der Rest der TWA hat den alten Inhalt, also genau so wie vor dem Aufruf des Unterprogramms.

Ein "I" in Spalte 53 arbeitet wie der Eintrag "S" mit dem Unterschied, dass die Bezugszahlen nicht aus dem Unterprogramm übernommen, sondern vom Temporary Storage zurückgeladen werden.

Beispiel : EXPR PROG2 S

Die Operation EXPR mit Programmnamen schließt eine RNDOM*ALL Operation ein. (Siehe H-Karte Spalte 33).

Im Ergebnisfeld kann der Name einer 8-stelligen Variablen eingetragen werden, wenn der Programmierer sicher stellt, dass dieses Feld zur Ausführungszeit einen gültigen Programmnamen enthält.
In einem solchen Fall muss Faktor 2 Blank sein. Ansonsten entsprechen die Regeln einer Eintragung im Faktor 2.

EF wird gesetzt, wenn das aufgerufene Programm nicht in der PPT liegt.

Optional kann im Faktor 1 ein numerisches Feld ohne Dezimalstellen angegeben werden. Der Inhalt dieses Feldes bestimmt bei der Ausführung des EXPR die Länge, in der die Common Area verarbeitet wird (maximal 4080 Bytes).

Bei Programmverbindungen (EXITP, EXPR) in ESA-Umgebungen größer 16 MB müssen alle verbundenen Programme auf der gleichen Seite der 16 MB-Linie liegen !

EXSR Unterprogramm ausführen 3198

Das Programm verzweigt in die unter Faktor 2 angegebene Unterroutine. In den Spalten 54 bis 59 können Ergebnisbezugszahlen gesetzt werden.

Beispiel:
```
EXSR SUBR                25
```

Diese Instruktion erzielt den gleichen Effekt wie die drei Instruktionen:
```
SETON                    25
EXSR SUBR
SETOF                    25
```

FILL Alphafeld füllen 3200

Mit diesem Befehl kann ein alphanumerisches Feld mit jedem beliebigen Zeichen (auch Blank) gefüllt werden. Das Ergebnisfeld enthält den Namen es zu füllenden Feldes. Faktor 2 enthält das Füllzeichen in Hochkommata, Faktor 1 bleibt frei.

Das Ergebnisfeld kann den Namen eines Alphafeldes, einer alphanumerischen Feldgruppe oder eines Feldgruppenelements enthalten.

Beispiel:
```
FILL '*'        FELD    6

Feldinhalt vorher:          (123456)
Feldinhalt nachher:         (******)
```

Faktor 2 kann auch einen hexadezimalen Direktwert in der Form X'00' enthalten.

FIND Durchsuchen einer Datenview 3210

Eine (extern erstellte) Datenview wird nach einem Feldinhalt durchsucht. Ist die Suche erfolgreich, wird auch der 'primary key' einer zugehörigen Datei gefunden.

Voraussetzung ist, dass eine Tabelle angelegt wurde, die mit FIND verarbeitet werden kann. Zum Erstellen einer solchen Tabelle und Regeln, die bei der Verarbeitung von Datenviews zu beachten sind, vergleiche Kapitel 2340.

Anwendungsbeispiele der FIND-Operation finden Sie in Kapitel 8000, Beispiele 33 und 34.

Faktor 1 enthält den Namen eines Feldes, das die zu durchsuchende Spalte der Tabelle angibt und nach dessen Inhalt diese Spalte durchsucht wird.

In Faktor2 steht der vierstellige Name der Tabelle.

Es kann eine Bezugszahl für den Vergleich auf Gleichheit gesetzt werden, also in den Spalten 58/59. Diese Bezugszahl wird gesetzt, wenn das Suchargument gefunden wurde.

Wird kein Element gefunden, wird der Schalter EF gesetzt und das CPG-interne Feld CPGFRC mit 'EF' gefüllt.

Wenn bei der Ausführung der Operation FIND 'AICA' oder 'SYSTEM UNDER STRESS' auftritt, so ist eventuell die Satzlängeneintragung in dieser Tabelle falsch.

GETHS Kanal erneut übertragen 3211

In einem HL1-Dataset soll der Datenkanal so wiederhergestellt werden, wie er beim Aufruf des Datasetmoduls gefüllt war.

Beschreibung:

GETHS wird z.B. in HL1-Datasets eingesetzt. Bei der Programmierung einer logischen Datei in einem Dataset kann die Schwierigkeit auftreten, dass Kanalfelder durch Lese-Operationen überschrieben wurden, z.B beim CHAIN vor einem Update.

Damit ist es möglich, mit einem HL1-Dataset die Update-Funktion auch dann zu realisieren, wenn nur ein Teil der Felder im Datenkanal übertragen wird. (Siehe Beispiel 1 : 1 Dataset).

Die Operation setzt jeweils alle Kanal-Felder auf den Wert zurück, den sie bei Aufruf des Datasets hatten.

GETIN GET Indicator 3212

Mit dieser Operation kann eine Bezugszahl von der nächst höheren Stufe ins Programm geholt werden. Dabei wird der Inhalt der Bezugszahl von der höheren Stufe in die gleichlautende Bezugszahl des ausführenden Programms übertragen.

Die Spalten 54 bis 59 können eine, zwei oder drei Bezugszahlen enthalten.

Beispiel:

```
C                   GETIN          15  20
```

Die Inhalte der Bezugszahlen 15 und 20 auf der nächst höheren Stufe werden in die Bezugszahlen 15 uund 20 des verarbeitenden Programms übertragen, das heisst, ist die Bezugszahl 15 auf der nächst höheren Stufe gesetzt,und

die Bezugszahl 20 gelöscht, so ist nach Ausführung der Operation auch im laufenden Programm die Bezugszahl 15 gesetzt und die Bezugszahl 20 gelöscht, unabhängig vom vorherigen Zustand.

GETMI GET MAIN Indicator 3213

Mit dieser Operation kann eine Bezugszahl von der nächst höheren Hauptprogrammstufe ins Programm geholt werden. Dabei wird der Inhalt der Bezugszahl des Hauptprogramms in die gleichlautende Bezugszahl des ausführenden Programms übertragen.

Die Spalten 54 bis 59 können eine, zwei oder drei Bezugszahlen enthalten.

Beispiel:

```
C                     GETMI           15  20
```

Die Inhalte der Bezugszahlen 15 und 20 des nächst höheren Hauptprogramms werden in die Bezugszahlen 15 und 20 des verarbeitenden Programms übertragen, das heisst, ist die Bezugszahl 15 im nächst höheren Hauptprogramm gesetzt und die Bezugszahl 20 gelöscht, so ist nach Ausführung der Operation auch im laufenden Programm die Bezugszahl 15 gesetzt und die Bezugszahl 20 gelöscht, unabhängig vom vorherigen Zustand.

GOTO Verzweigen nach 3220

Diese Operation verzweigt zu dem in Faktor 2 genannten Merkmal. Dieses Merkmal muss mit Hilfe einer TAG-Operation definiert werden.

IF Wenn-Abfrage 3250

Die IF-Operation ermöglicht die Ausführung einer Gruppe von Rechenbestimmungen unter der Bedingung, dass eine Beziehung zwischen Faktor 1 und Faktor 2 besteht.

IF ist in beiden Faktoren indizierbar.
Siehe auch Kapitel Strukturierte Programmierung. (2450).

Mit IF können auch Schalter abgefragt werden. Bei dieser Verarbeitungsart bleiben Faktor 1 und 2 leer.

In Spalte 53 kann ein 'D' eingetragen werden.

Beispiele:

```
C           LOEDAT    IFGT UDATE                 D
C           ALPHA8    IFLT NUM20                 D
-----------------------------------------------------
```

Zweck:

Vergleich von Datumsfeldern, die in der Form ttmmjj (Tag,Monat,Jahr) gefüllt sind. Durch den Eintrag von 'D' spart man die Konvertierung der Felder vor dem Vergleich.

Beschreibung:

Intern werden die Felder CPGWD1 und CPGWD2 verglichen, die das Datum achtstellig alphanumerisch im ISO-Format enthalten.

Man kann alphanumerische und numerische Felder miteinander vergleichen, auch wenn Länge und/oder Typ unterschiedlich sind. Die Vergleichsfelder werden intern zunächst in achtstellige CPGWD-Felder konvertiert. Lässt die Feldlänge der Vergleichsfelder keine Werte für Monat und/oder Tag zu, werden diese intern durch '01' ersetzt.

Beispiel: Aus dem zweistelligen Feld, das den Wert 97 enthält, wird CPGWDx mit dem Wert 19970101.

Beachte: Die Vergleichsfelder werden nicht geprüft.Der Programmierer ist dafür verantwortlich, dass die Felder im richtigen Format zur Verfügung stehen.

Unterstützte Formate für den Eintrag 'D':

Länge	Alpha	Numerisch mit 0 Dezimalen	intern
2	97	97	19970101
3		097	19970101
		197	19970101
4	1297	1297	19971201
5	12.97	01297	19971201
	12/97	11297	19971201
		21297	20971201
6	311297	311297	19971231
7		0311297	19971231
		1311297	19971231
		0120502	20020512
		1120502	19020512
		2120502	20020512
8	31121997	31121997	19971231
	31.12.97		19971231
	31/12/97		19971231
9		031121997	19971231
10	31.12.1997		19971231
	31/12/1997		19971231

Grundsätzlich arbeitet die interne Routine nach der Regel, dass übergebene Daten nicht verändert werden. Ist ein Vergleichsfeld z.B. 4-stellig alpha mit dem Wert '0097', so wird intern ein Wert '01001997' verarbeitet. Der (nicht vorhandene) Tag wird durch '01' ersetzt, der (falsche) Monat '00' wird nicht verändert.

Bei ungeradstelligen numerischen Feldern steht die erste Stelle für das Jahrtausend: 1 für 1900, 2 für 2000. Für alle anderen Werte wird das 'Fenster' angenommen.

Die IF-'D'-Operation arbeitet nach der gleitenden Fenstertechnik mit einem Defaultwert von 30. Das heisst, dass Jahreszahlen größer als 30 dem Jahrhundert 19xx zugeordnet werden, Jahre kleiner und gleich 30 dem Jahrhundert 20xx. Ein anderes Fenster kann in der Kundenkonfiguration (CPGURSI2) vorgegeben werden.

In Spalte 53 kann ein 'I' eingetragen werden.

Beispiele:

```
C           LOEDAT    IFGT CPGDAI                I
C           ALPHA8    IFLT NUM20                 I
```

Zweck:

Vergleich von Datumsfeldern, die im ISO-Format vorliegen, unabhängig von der Länge. Insbesondere bedeutet das, dass Datumsabfragen mit bis zu sechsstelligen Datumsfeldern auch über den Jahrtausendwechsel hinaus funktionieren.

Beschreibung:

Es werden intern die Felder CPGWD1 und CPGWD2, die das Datum achtstellig alphanumerisch im ISO-Format enthalten, verglichen.

Es können alphanumerische und numerische Felder miteinander verglichen werden, auch wenn Länge und/oder Typ unterschiedlich sind. Die Vergleichsfelder werden intern zunächst in achtstellige CPGWD-Felder konvertiert. Lässt die Feldlänge der Vergleichsfelder keine Werte für Monat und/oder Tag zu, werden diese intern durch '01' ersetzt.

Beispiel: Aus dem vierstelligen Feld, das den Wert 9707 enthält, wird CPGWDx mit dem Wert 19970701.

Beachte:
- Die Vergleichsfelder werden nicht geprüft. Der Programmierer ist dafür verantwortlich, dass die Felder im richtigen Format zur Verfügung stehen.
- Vorsicht bei Vergleichen mit unterschiedlichen Feldlängen !
- Beim Ersetzen funktionierender Vergleiche (mit IF oder COMP) durch den Eintrag 'D' muss man genau untersuchen, ob das Ergebnis gleich sein wird.

Unterstützte Formate für den Eintrag 'I':

Länge	Alpha	Numerisch mit 0 Dezimalen	intern
2	97	97	19970101
3		097	19970101
		197	19970101
4	9712	9712	19971201
5	97.12	09712	19971201
	97/12	19712	19971201
6	971231	971231	19971231
7		0971231	19971231
		1971231	19971231
8	19971231	19971231	19971231
	97.12.31		19971231
	97/12/31		19971231
9		019971231	19971231
10	1997.12.31		19971231
	1997/12/31		19971231

Grundsätzlich arbeitet die interne Routine nach der Regel, dass übergebene Daten nicht verändert werden. Ist ein Vergleichsfeld z.B. 4-stellig alpha mit dem Wert '0097', so wird intern ein Wert '01001997' verarbeitet. Der (nicht vorhandene) Tag wird durch '01' ersetzt, der (falsche) Monat '00' wird nicht verändert.

Bei ungeradstelligen numerischen Feldern steht die

erste Stelle für das Jahrtausend: 1 für 1900, 2 für 2000. Für alle anderen Werte wird das 'Fenster' angenommen.

Die IF-'I'-Operation arbeitet nach der gleitenden Fenstertechnik mit einem Defaultwert von 30. Das heisst, dass Jahreszahlen größer als 30 dem Jahrhundert 19xx zugeordnet werden, Jahre kleiner und gleich 30 dem Jahrhundert 20xx. Ein anderes Fenster kann in der Kundenkonfiguration (CPGURSI2) vorgegeben werden.

Es kann ein 'K' in Spalte 53 eingetragen werden.

Beispiel:

```
C           KTAG1     IFLE KTAG2                  K
-----------------------------------------------------
```

Zweck:

Vergleich von Kalendertagen, die in der Form jjttt oder jjjjttt in einem fünf- oder siebenstelligen alphanumerischen oder numerischen Feld stehen.

Beachte:

Diese Operation ist in erster Linie eine Umstellungshilfe für den Jahrtausendwechsel. Nach dem Jahr 2000 wird die Operation IF zur Abfrage ausreichen.

Intern werden die Daten in der Form jjjjttt0 bereitgestellt und verglichen.

Die Operation steht im QPG nicht zur Verfügung.

IFC Interface-Call 3260

Mit Hilfe dieser Operation kann der Programmierer zu einer selbstgeschriebenen Routine im Steuerprogramm-Interface verzweigen. Voraussetzung für einen Interface-Call ist, dass die Interface-Communication-Area (CPGIFC) vorher über eine EDIT-Operation die erforderlichen Parameter übernommen hat.

Der Operationscode kann entweder allein oder in Verbindung mit einem gültigen Dateinamen in Faktor 2 verwendet werden. Im 2. Falle erfolgt nach Ausführung der Interface-Routine eine Verzweigung in die Eingabebestimmungen für die in Faktor 2 genannte Datei.

Wird im Ergebnisfeld als Feldname CPGIFC eingetragen, so wird vor dem Interface Call eine EDIT-Operation für dieses Feld durchgeführt.

In Spalte 54 bis 59 können bis zu drei Bezugszahlen eingetragen werden, die dann jeweils vor Ausführung der Operation gesetzt und nachher wieder gelöscht werden.

INDOF Bezugszahlen löschen 3261

Mit Hilfe dieser Operation können alle Bezugszahlen 01 bis 99 gelöscht werden.

Beispiel: INDOF

Sind vorher die Schalter 08, 10, 23, 17, T1, C3 und P1 gesetzt, so sind nach Ausführen der Operation die Schalter : T1, C3, P1 noch gesetzt.

Werden in den Spalten 54 bis 57 zwei gültige Bezugszahlen aufsteigend sortiert eingetragen, so wird der damit beschriebene Bezugszahlenbereich gelöscht.

Beispiel: INDOF 2139

Diese Operation löscht die Bezugszahlen von 21 bis 39.

INDON Bezugszahlen setzen 3262

Mit Hilfe dieser Operation können alle Bezugszahlen 01 bis 99 gesetzt werden.

Beispiel: INDON 0199
oder INDON

Sind vorher die Schalter 08, 10, 23, 37, T1, C3 und P1 an, so sind nach Ausführen der Operation die Schalter: T1, C3, P1 sowie alle Schalter 01 bis 99 gesetzt.

In den Spalten 54 bis 56 können zwei gültige Bezugszahlen aufsteigend sortiert eingetragen werden. Der entsprechende Bezugszahlenbereich wird damit auf EIN gesetzt.

Beispiel: INDON 2139

Diese Operation setzt die Bezugszahlen von 21 bis 39.

JRB Rechtsbündig verschieben mit Blank 3280

Mit diesem Befehl kann der Inhalt eines Alphafeldes so verschoben werden, dass das letzte Zeichen ungleich Blank in der rechtesten Stelle des Feldes steht. Von links wird das Feld bis zum 1. Byte des Inhalts mit Blanks aufgefüllt. Anwendung: Randausgleich für Textausgaben, ungepackte numerische Daten u.a.

Faktor 1 und 2 bleiben frei.

Beispiel: JRB FELD 6

Feldinhalt vorher: (123)
Feldinhalt nachher: (123)

Wird in Spalte 53 ein 'L' eingetragen, so wird das Feld linksbündig verschoben.

Beispiel: JRB FELD 6 L

Feldinhalt vorher: (123)
Feldinhalt nachher: (123)

JRC Rechtsbündig verschieben mit Zeichen 3281

wie JRB, jedoch Auffüllung von links mit einem beliebigen Zeichen, das in Faktor 2 in Hochkommata eingeschlossen angegeben wird.

```
Beispiel:            JRC  '*'        FELD    6

Feldinhalt vorher                   (123   )
Feldinhalt nachher                  (***123)
```

JRZ Rechtsbündig verschieben mit Null 3282

Wie JRB, jedoch Auffüllung von links mit Nullen.

```
Beispiel:            JRZ             FELD    6

Feldinhalt vorher:                  (123   )
Feldinhalt nachher:                 (000123)
```

LIST Ausgabe extern beschriebener Listen 3283

```
Faktor 1      Druckername ( nur online erforderlich )
Operation     LIST  muss eingetragen werden
Faktor 2      Name des zu druckenden QTF-Dokuments
Ergebnisfeld  Name einer Section im Druckdokument
```

Beispiele:

```
C   08      'L86C'   LIST KUNDE
C           DRID     LIST DOKUM5  KOPF
-----------------------------------------------------
```

Zweck:

Drucken einer Liste, die programmextern im QTF (Quick Text Facility) beschrieben ist.

Beschreibung:

In F1 muss (außer bei Batch-Programmierung) der Druckername als Konstante oder als Feld angegeben werden. In F2 wird der Name des QTF-Dokuments angegeben, das die Beschreibung der Liste enthält. Dieses Dokument muss im QTF in der Library LIST abgestellt sein.

Zusätzlich kann der Name einer Section angegeben werden. Eine Section ist ein Teil eines Dokuments.

Vorschübe vor und nach dem Drucken werden ebenfalls im Dokument beschrieben. Wird die definierte Seitenlänge überschritten, so wird der Schalter OF gesetzt.

In Spalte 53 kann 'P' oder 'I' eingetragen werden.

Der LIST-Befehl soll variabel programmiert werden. (Spalte 53=V). Die notwendigen Daten werden zur Ausführungszeit einem 32-stelligen Feld im Ergebnisfeld entnommen.

Aufbau des 32-stelligen Feldes (alle Teilfelder alpha)

```
Stelle  1 -  8  Dokument
Stelle  9 - 14  Section
Stelle 15 - 18  Drucker-Id
Stelle 19 - 22  Library
Stelle 23 - 23  Drucker Exit (siehe QTF-Handbuch)
Stelle 24 - 24  Übersetzen  ( N=nicht, 1=A-Z, 2=ÄÖÜ)
Stelle 25 - 25  Phasen-Verarbeitung ( P oder I )
Stelle 26 - 32  noch nicht belegt
```

Ein ausführliches Anwendungsbeispiel befindet sich im QTF-Handbuch. (Seite 5041).

LOADT Geretteten Bildschirminhalt zurückladen 3284

Diese Operation schreibt den mit der Operation SAVET geretteten Bildschirminhalt wieder zurück auf das Terminal. Im Faktor2 wird 4-stellig der Name der TS-Queue definiert, welcher schon bei der SAVET-Operation verwendet wurde. Diese Operation ist nur zusammen mit der SAVET-Operation sinnvoll. Dabei ist es nicht erforderlich, dass beide Operationen im gleichen Programm verwendet werden. Bei einem Fehler wird von CPG der Schalter EF gesetzt.

In Spalte 53 kann ein 'S' für Save eingetragen werden. Ein mit SAVET gerettetes Bild kann mit LOADT 'S' mehrfach angezeigt werden. Der Zwischenspeicherbereich bleibt so lange erhalten, bis LOADT ohne 'S' in Spalte 53 ausgeführt wird (bzw. bis zum CICS Shut Down).

Im Ergebnisfeld kann ein 4-stelliger Feldname für einen variablen Storage eingetragen werden.

Beispiel:

```
SAVETHUGO
EXPR PROGDESC              I
LOADTHUGO
MOVE 'EMIL'  AF        4
SAVET        AF
EXPR SUBPROG
LOADT        AF
```

Das Zurückladen der Farben ist nur dann gewährleistet, wenn sie per Farbattribut(B,G,P,R,T,W,Y) gesetzt werden.

LOKUP Feldgruppen suchen 3320

Dieser Befehl ermöglicht es, Feldgruppen auf einen bestimmten Inhalt hin zu untersuchen. Der zu suchende Inhalt wird entweder als alphanumerisches Literal oder als variables Feld angegeben.

Die angegebene Feldgruppe wird elementweise mit dem Suchargument auf Identität verglichen. Die Bezugszahlen werden in Abhängigkeit vom Vergleich gesetzt.

Beispiel : Ist die dritte Bezugszahl für den Vergleich auf Gleichheit angegeben, so wird diese Bezugszahl gesetzt, falls ein Feldgruppenelement mit dem Suchargument gleich ist. Ansonsten bleibt die Bezugszahl aus.

Sobald ein Schalter gesetzt ist, ist die LOKUP-Operation beendet.

Wird die LOKUP-Operation ohne Bezugszahl verwendet, so wird auf 'gleich' gesucht.

Anstatt Bezugszahlen kann >> << == oder GT LT EQ eingesetzt werden.

Das Durchsuchen einer Feldgruppe auf größer oder kleiner ist nur möglich, wenn die Feldgruppe sortiert ist.

Es besteht die Möglichkeit, einen Index zur Feldgruppe anzugeben. Dieser Index erfüllt dann zwei Funktionen:

1. Der Vergleich beginnt erst bei dem durch den Inhalt des Indexfeldes gekennzeichneten Feldgruppenelement.

2. Nach der LOKUP-Operation steht im Indexfeld der Index des Elements, das die Bedingung des Vergleichs erfüllt hat bzw. 0 (Null), wenn die Bedingung von keinem Element der Feldgruppe erfüllt wurde.

Beispiel : Die 5-elementige Feldgruppe FG3 habe folgenden Inhalt :

```
CPG
HL1
QFF
QSF
QTF
```

Der Befehl

```
C          'HL1'     LOKUPFG3,I                11
```

hat folgende Auswirkungen :

Ist vor dem LOKUP z.B. I=1, dann wird 'HL1' gefunden. Damit ist nach dem LOKUP I=2 und der Schalter 11 gesetzt.

Ist vor dem LOKUP z.B. I=3, dann werden nur die Elemente 3 bis 5 mit dem Argument 'HL1' verglichen; daraus folgt, dass der Schalter 11 nicht gesetzt und der Index I auf 0 gesetzt wird.

Zu beachten ist, dass das Suchargument und die Feldgruppe in der Länge übereinstimmen müssen. Im obigen Beispiel wäre etwa ein Suchargument 'Q' wegen der Länge 1 nicht zulässig.

MACRO Macro einfügen 3330

Mit der Operation MACRO kann eine Assembler-Instruktion ins Programm eingefügt werden. Faktor 2 enthält dabei von Spalte 33 bis 74 die Assembler-Instruktion.

In Faktor 1 kann ein Label eingetragen werden.

Beispiel:

```
      C              MACROMVC     AUSG(5),EING
      C     LABEL    MACROGETTIME TU
```

Das MACRO 'GETTIME' mit dem Parameter 'TU' wird an dieser Stelle ins Programm eingefügt.

MAP Lesen QSF-Map oder Löschen von Maskenfeldern im Programm 3335

Die Operation MAP liest transaktionsorientiert Daten vom Bildschirm ein. Dieser Befehl kann am Programmanfang gegeben werden. Im Faktor2 wird der Mapname eingetragen, zu der die Felder der Terminal I-O Area zugeordnet werden. In Spalte 53 kann ein 'L' gesetzt sein. In diesem Fall wird von QSF keine Übersetzung in Großbuchstaben durchgeführt. Dies setzt allerdings voraus, dass in der CICS TCT UCTRAN nicht gesetzt ist.

Beispiel:

```
      C              MAP  BUCHMENU
```

Die Map "BUCHMENU" wird als Programmteil interaktiv beim Befehl MAP nach Eingabefeldern dieses Programms durchsucht; Eingabefelder werden, falls vorhanden, in die TWA übertragen.

Wird in Spalte 53 ein 'B' gesetzt, so findet keine Eingabeübertragung statt, sondern die Felder, die in der Maske verwendet werden, werden im Programm auf Blank bzw Null gelöscht.

Wird in Spalte 53 ein 'C' gesetzt, so findet keine Eingabeübertragung statt, sondern die in der Maske verwendeten variablen Attributfelder werden im Programm auf Blank gelöscht.

Durch die Positionierung der Eintragungen kann man die Verwendung eines variablen Mapnamens ermöglichen.

Wird eine Eintragung in das Ergebnisfeld anstatt in Faktor 2 vorgenommen, so wird diese als Feld interpretiert, das unter anderem den Mapnamen variabel enthält.

Beispiel:

```
      C              MAP            VMAP01
```

Dieses Feld im Ergebnisfeld muss 16-stellig alphanumerisch definiert sein und per Programm gepflegt werden; es enthält die folgenden Informationen:

1-8 Mapname

9 'Y' bedeutet, dass der Bildschirm vorher gelöscht wird.

'N' bedeutet, dass kein Erase durchgeführt wird.

'Y' und 'N' haben Vorrang vor dem im QSF definierten Erase-Eintrag.

' ' der Erase-Eintrag aus der QSF-Definition wird verwendet.

10 Write Control Character. Siehe Tabelle Seite 7030.

11 'F' bedeutet, nur variable Felder ausgeben. Dies bewirkt eine Minimierung der Datenübertragung bei entfernt betriebenen Bildschirmen. (Alle var. Felder werden mit Attributen ausgegeben).

'S' Dieser Eintrag bewirkt das gleiche wie 'F',aber die Ausgabe der variablen Felder erfolgt ohne Attribute.

12 ' ' MAP-Funktion ausführen
'B' Maskenfelder im Programm löschen.
'C' Variable Attributfelder der Maske im Programm löschen
'O' Maskenausgabe (auch bei anderen Einträgen)

13-16 reserviert.

MAPD Dialog QSF-Map 3336

Die Operation MAPD ist eine Abkürzung für die Operationen MAPO und MAPI. Im Faktor2 wird ein Mapname eingetragen, der sowohl für die Ein- als auch für die Ausgabe verwendet wird. Einträge in Spalte 53 werden wie für die MAPI-Operation gesetzt.

Beispiel:

```
      C              MAPD BUCHMENU
```

Wird eine Eintragung in das Ergebnisfeld anstatt in Faktor 2 vorgenommen, so wird diese als Feld interpretiert, das (unter anderem) den Mapnamen variabel enthält.

Dieses Feld im Ergebnisfeld muss 16-stellig alphanume-

risch definiert sein und per Programm gepflegt werden; es ist unter dem Befehl MAP ausführlich erklärt.

Beispiel:

```
      C              MAPD           VMAP01
```

Bei dialogorientierter Programmierung ist für Kleinbuchstaben bei der Operation MAPD in Spalte 53 ein 'L' einzutragen.

MAPI Eingabe QSF-Map 3337

Die Operation MAPI ersetzt in einem Dialogprogramm die READ-Operation vom Bildschirm. Der in Faktor2 eingetragene Mapname definiert die Eingabemap, zu der die Bildschirmfelder über QSF zugeordnet werden.

Beispiel:

```
      C              MAPI BUCHMENU
```

Wird eine Eintragung in das Ergebnisfeld anstatt in Faktor 2 vorgenommen, so wird diese als Feld interpretiert, das (unter anderem) den Mapnamen variabel enthält.

Dieses Feld im Ergebnisfeld muss 16-stellig alphanumerisch definiert sein und per Programm gepflegt werden; es ist unter dem Befehl MAP ausführlich erklärt.

Beispiel:

```
      C              MAPI           VMAP01
```

Die Einträge in Spalte 53 entsprechen den Einträgen der READ-Operation vom Bildschirm.

Bei dialogorientierter Programmierung ist für Kleinbuchstaben bei der Operation MAPI in Spalte 53 ein 'L' einzutragen.

Im übertragenen Sinn entspricht diese Operation einer READ-Operation vom Bildschirm. Im Faktor2 wird jedoch der Mapname und nicht eine Eingabesatzbestimmung definiert.

Tabelle der Einträge in Spalte 53

Spalte 53		A	C	I	K	L	S	T
Task		x	x		x			x
Lower Case		x			x	x	x	
CL-Abfrage			x	x	x		x	

MAPO Ausgabe QSF-Map 3338

Die Operation MAPO schreibt die in Faktor2 angegebene MAP auf den Bildschirm. Ausgabebestimmungen für Bildschirme sind bei dieser Programmierart nicht erforderlich.

Wird eine Eintragung in das Ergebnisfeld anstatt in Faktor 2 vorgenommen, so wird diese als Feld interpretiert, das (unter anderem) den Mapnamen variabel enthält.

Dieses Feld im Ergebnisfeld muss 16-stellig alphanumerisch definiert sein und per Programm gepflegt werden; es ist unter dem Befehl MAP ausführlich erklärt.

Mapausgaben können auf bestimmte Zeilen begrenzt werden. (Siehe Abschnitt 2132).

```
CPGMFN = '$QV'
CPGMLC = '0205'   (von Zeile - bis Zeile)
MAPO Maske
```

MAPP Drucken einer QSF-Map 3339

Die Operation MAPP druckt die im Faktor 2 eingetragene MAP auf den im Faktor 1 definierten Online-Drucker. Faktor 1 kann ein 4-stelliges alphanumerisches Feld oder eine Konstante sein.

Wird eine Eintragung in das Ergebnisfeld anstatt in Faktor 2 vorgenommen, so wird diese als Feld interpretiert, das (unter anderem) den Mapnamen variabel enthält.

Dieses Feld im Ergebnisfeld muss 16-stellig alphanumerisch definiert sein und per Programm gepflegt werden; es ist unter dem Befehl MAP ausführlich erklärt.

Beispiel:

```
      C     DRID     MAPP           VMAP
```

Achtung: Es werden nur Zeilen gedruckt, in denen sich zumindest ein Zeichen befindet. Sollen also 24 Zeilen gedruckt werden, so muss in jeder Zeile der Map zumindest ein Blank (dargestellt durch ein '#') beschrieben werden.

In Spalte 53 sind folgende Einträge unterstützt:

'A' - Nach dem MAPP Command wird ein Vorschub auf Blattanfang durchgeführt.
'B' - Vor dem MAPP Command wird ein Vorschub auf Blattanfang durchgeführt.
'S' - Vor und nach dem MAPP Command wird ein Vorschub auf Blattanfang durchgeführt.

MLLZO Move Low to Low Zone 3340

Übertragen der Zone des rechtesten Bytes des in Faktor 2 angegebenen Feldes auf das rechteste Halbbyte des Ergebnisfeldes. Faktor 1 bleibt frei. Diese Operation ist für alphanumerische und numerische Felder unterstützt.

Für numerische Felder wird in Faktor 2 eine Ziffer eingetragen; es wird also grundsätzlich das Zonenhalbbyte 'F' in das Ergebnisfeld übertragen. Bei Alphafeldern kann Faktor 2 einen Feldnamen oder eine einstellige Konstante in Hochkommata enthalten.

Beispiel:	MLLZO'9'	ALPHA
Feldinhalt vorher:	(F4 F3 F2 C1)	(432A)
Feldinhalt nachher:	(F4 F3 F2 F1)	(4321)
	MLLZO1	NUM
Feldinhalt vorher:	(00 10 0C)	(0010)
oder	(00 10 0D)	(0010)
oder	(00 10 0F)	(00100)
Feldinhalt nachher:	(00 10 0F)	(00100)

MOVE Rechtsbündig übertragen 3341

Faktor 2 wird von rechts nach links in das Ergebnisfeld übertragen.

Die Felder können alphanumerisch, numerisch oder unterschiedlich definiert sein.

MOVE ist voll indizierbar.

Ein ' ' in Spalte 53 bewirkt, dass bei der MOVE-Operation von numerisch nach alpha bei positivem Feldinhalt das Vorzeichen F eingesetzt wird.

Bei einer MOVE-Operation von numerisch nach alpha kann die rechte Zone des Alphafelds verändert werden. In der erweiterten Standard H- Karte (Abschnitt 9310) kann eine der folgenden Eintragungen vorgenommen werden, die dann für alle MOVE-Operationen des Programms gültig ist.

Ein Eintrag in Spalte 53 der Operation selbst hat Vorrang vor dem H-Karteneintrag.

Ein 'C' bewirkt, dass bei der MOVE-Operation von numerisch nach alpha das Vorzeichen des numerischen Feldes beibehalten wird.

Ein 'F' entspricht dem Blank.

Ein 'I' bewirkt, dass ein benutzter Index nicht geprüft wird. Der Programmierer ist dann dafür verantwortlich, dass kein Bereich irrtümlich überschrieben wird.

MOVEA Bereich übertragen 3342

Mit dieser Operation kann eine Feldgruppe oder eine Konstante in eine andere Feldgruppe oder in ein Einzelfeld, oder ein Feld in eine Feldgruppe übertragen werden.

MOVEA gilt nur für alphanumerische Felder.

Maximale Feldlänge sind 256 Bytes.

Faktor 2 oder das Ergebnisfeld muss eine Feldgruppe enthalten. In Faktor 2 und im Ergebnisfeld darf nicht dieselbe Feldgruppe eingetragen werden, selbst wenn es sich um eine indizierte Feldgruppe handelt.

Mit der MOVEA-Operation ist es möglich,

. mehrere aufeinanderfolgende alphanumerische Feldgruppenelemente in ein einzelnes alphanumerisches Feld zu übertragen.

. ein einzelnes alphanumerisches Feld in mehrere aufeinanderfolgende alphanumerische Feldgruppenelemente zu übertragen.

. aufeinanderfolgende Feldgruppenelemente in aufeinanderfolgende Elemente einer anderen Feldgruppe zu übertragen.

Bei nicht indizierten Feldgruppen beginnt die Übertragung mit dem ersten Element der Feldgruppe, bei indizierten Feldgruppen mit dem angegebenen Element.

Die Übertragung ist abgeschlossen, wenn das letzte Element der Feldgruppe übertragen oder aufgefüllt ist.

Die Übertragung von Daten mit einer alphanumerischen MOVEA-Operation ist beendet, wenn die Anzahl der übertragenen Zeichen gleich der kürzeren Längendefinition der Felder in Faktor 2 bzw. im Ergebnisfeld ist.

Eine alphanumerische MOVEA-Operation kann daher auch mitten in einem Feldgruppenelement enden.

Beispiel: C MOVEA'ABC' FG

MOVEL Linksbündig übertragen 3343

Faktor 2 wird von links nach rechts in das Ergebnisfeld übertragen.

Die Felder können alphanumerisch, numerisch oder unterschiedlich definiert sein.

Diese Operation ist voll indizierbar.

Ein 'I' in Spalte 53 unterdrückt die Indexprüfung.

MOVEN Alphanumerisches in numerisches Feld übertragen 3344

Der Inhalt eines alphanumerischen Feldes soll in ein numerisches Feld übertragen werden, und zwar so, dass es der Bildschirmeingabe entspricht.

Faktor2 und das Ergebnisfeld können auch Feldgruppenelemente oder ganze Feldgruppen sein.

Das bedeutet, dass eine Dezimalstellenanpassung vorgenommen wird und gegebenenfalls die Stellen mit dem höchsten Wert vorne abgeschnitten werden. Das Komma wird als Separator der Dezimalstellen erkannt. Minuszeichen und die Zeichenfolge 'CR' im Alphafeld bewirken, dass das Feld als negativer Wert interpretiert wird. Ungültige Zeichen werden aus dem Feld eliminiert und das Feld wird vor dem MOVEN entsprechend komprimiert.

Es können drei Bezugszahlen angegeben werden:

Die erste wird gesetzt, wenn das Alphafeld ungültige Zeichen enthält.

Die zweite wird gesetzt, wenn das Alphafeld vor oder nach einem Komma zu viele Stellen enthält.

Die dritte wird gesetzt, wenn das Alphafeld Blank ist.

Beispiel : MOVENALPHA NUM 111213

Das numerische Feld sei siebenstellig mit 2 Dezimalstellen definiert.

ALPHA	NUM	intern	gesetzte Bezugszahl
'123 '	123,00	0012300C	
' 123 '	123,00	0012300C	
' 1 2 3 '	123,00	0012300C	
'-123 '	123,00-	0012300D	
'12-3 '	123,00-	0012300D	
' 987,65CR '	987,65-	0098765D	

```
',1        '      0,10   0000010C
'12.345,678'  12345,67   1234567C    12, zu viele Dezim.
'56,7D     '     56,70   0005670C    11, ung. Zeichen
'ELF DM    '      0,00   0000000F    11, ung. Zeichen
'          '      0,00   0000000F    13, Feld ist blank
'12A4567   '  24567,00   2456700C    11 und 12
```

Ist ein 'H' in Spalte 53 eingetragen, so gelten folgende Regeln:

```
' 1 2 3    '    123,00   0012300C    11, ung. Zeichen
'12-3      '    123,00-  0012300D    11, ung. Zeichen
```

Grundsätzlich gilt bei der Fehlerprüfung: Sobald der Schalter für die falsche Anzahl von Dezimalstellen gesetzt ist (Schalter 2) wird die Prüfung abgebrochen. Geprüft wird byteweise von links nach rechts.

Beispiel: (Sonderfall)

In ein 7-stelliges Feld ohne Dezimalstellen wird mit dem Service H die Zeichenkette ' 1-,5- ' geschoben. Es müsste der erste Schalter für 'ungültiges Zeichen' gesetzt werden (Minuszeichen mitten im Feld). Da aber nach dem Komma die 5 als falsche Dezimalstelle erkannt und somit der entsprechende Schalter 2 gesetzt wird, ist die bisher geprüfte Zeichenkette nicht fehlerhaft. Schalter 1 bleibt deshalb aus.

MOVEV variable MOVE-Operation 3345

```
Operation     MOVEV muss eingetragen werden
Faktor 2      10-stelliges alphanumerisches Feld
Ergebnisfeld  10-stelliges alphanumerisches Feld

SP53 A, L, R
```

Beispiele:

```
C                    MOVEVA         B
C                    MOVEVA         B         L
C   P1               MOVEVF1        F2        A
```

Zweck:

Der Inhalt des Feldes, dessen Name in F2 steht, soll in das Feld, dessen Name in EG steht, übertragen werden.

Beschreibung:

Diese Operation ermöglicht es, den Programmablauf von außen, z.B. über eine Tabelle, zu beeinflussen.

Alphanumerische Felder werden linksbündig, numerische rechtsbündig übertragen.

Die Servicefunktionen haben folgende Bedeutung:

A für ARRAY. MOVEV arbeitet in diesem Fall wie MOVEA.
L für Left. MOVEV arbeitet in diesem Fall wie MOVEL.
N für numerisch. MOVEV arbeitet dann wie MOVEN.
R für Right. MOVEV arbeitet in diesem Fall wie MOVE.

MOVEV ist nicht in den Einträgen indizierbar, kann aber wie folgt indiziert verarbeitet werden:

Beispiel:

```
C                   MOVE X          A
C                   MOVEL'FG    '   A
C                   MOVEL'RESULT'   B
C                   MOVEVA          B          R
```

Damit wird das X-te Element der Feldgruppe FG in das Feld RESULT rechtsbündig übertragen.

Damit diese indizierte Verarbeitungsform möglich ist, müssen die beiden Einträge immer 10 Stellen große Alphafelder sein. In den ersten sechs Stellen dieser Felder steht immer der Feldname. Bei indizierter Verarbeitung steht also in den Stellen 1 bis 6 der Name der Feldgruppe, in den Stellen 7 bis 10 aber der Wert des Indexfelds.

Wird bei der Operation MOVEV ein Fehler festgestellt, z. B. dass der Feldname fehlerhaft oder der Index ungültig ist, so findet keine Übertragung statt. Es wird stattdessen der Schalter EF gesetzt, um den Fehler anzuzeigen.

In Spalte 53 kann ein 'N' für eingetragen werden, allerdings nicht für Feldgruppen und Feldgruppenelemente.

1. Alpha nach numerisch: Übertragung wie bei MOVEN.

Beispiel:	Alpha (15)	Numerisch (9,3)
Feldinhalt:	123,999999	000123999C
Feldinhalt:	123,999999-	000123999D
Feldinhalt:	1234567,999999	234567999C
Feldinhalt:	-1234567,999999	234567999D

2. Numerisch nach alpha.

   - Die Übertragung erfolgt rechtsbündig.
   - Das empfangende Alphafeld muss groß genug sein.
   - Der Wert wird mit dem Edit-Code J aufbereitet.
   - Das Alphafeld sollte mit Blanks initialisiert werden.

Beispiel:	Numerisch (9,3)	Alpha (15)
Feldinhalt:	000123999C	123,999
Feldinhalt:	000123999D	123,999-
Feldinhalt:	234567999C	123.456,999
Feldinhalt:	234567999D	123.456,999-

MULT Multiplizieren 3347

Nach Ausführung der Operation enthält das Ergebnisfeld das Produkt von Faktor1 und Faktor2.

Diese Operation ist indizierbar. Beispiel siehe 'ADD'.

Bei der Multiplikation ist zu beachten, dass die Dezimalstellenanpassung zu fehlerhaften Ergebnissen führen kann, wenn das Ergebnisfeld nicht die erforderliche Größe aufweist. Als Regel gilt, dass das Ergebnisfeld mindestens die Summe der Stellen vor dem Komma und die Summe der Stellen nach dem Komma aufnehmen können muss.

Beispiel:

```
            333,1    MULT 333,22    ERG     93
            XXX,X         XXX,XX    XXXXXXX,XXX
```

MVR Rest übertragen 3348

Mit dieser Operation kann unmittelbar nach einer Division (DIV) der Rest in das im Ergebnisfeld eingetragene Feld übertragen werden.

MVR kann nur verwendet werden, wenn in der zugehörigen DIV-Operation nicht gerundet und im Ergebnisfeld keine Feldgruppe eingetragen wurde.

```
Beispiel:    50       DIV   3        QUOT
                      MVR            REST
```

Nach der Operation MVR enthält das Ergebnisfeld (REST) den Wert 2.

Vom CPG wird automatisch an das Ergebnisfeld eine Dezimalstellenanpassung durchgeführt.

NEWRC Siehe WRITE 3360

Diese Operation wurde aus Gründen der Kompatibilität zu RPG in WRITE umbenannt.

Bei neuen Programmen bzw. bei Programmänderungen sollte die neue Bezeichnung verwendet werden.

OPEN Datei eröffnen 3390

Die OPEN-Operation eröffnet die Datei, die in Faktor 2 eingetragen ist. In den Stellen 56/57 kann eine Ergebnisbezugszahl angegeben werden, die bei nicht erfolgreichem Abschluss der OPEN-Operation auf Ein gesetzt wird.

Will man auf den Schalter verzichten, so kann im internen Feld CPGFRC 'NF' für 'not found in FCT' bzw. 'Not found' (Batch) und 'NO' für 'not open' abgefragt werden.

```
Beispiel:       OPEN DATEI                      18
```

Ein 'V' in Spalte 53 bedeutet, dass in Faktor 2 ein Feldname eingetragen werden kann. Das Feld muss acht Stellen alpha definiert sein und kann mit variablen Dateinamen aufbereitet werden.

```
Beispiel:       MOVEL'DATEI'     FELD       8
                OPEN FELD                     V  18
```

In Spalte 53 sind für spezielle Anforderungen in HL1-Schnittstellen-Modulen zu CPG4-Produkten noch die Einträge 'A', 'I' und 'R' unterstützt. Diese Einträge sind jeweils bei dem entsprechenden HL1-Modul erklärt.

Wird ein OPEN auf eine leere VSAM-Datei im Batch durchgeführt, so wird der Schalter EF gesetzt. Weiterhin wird das Feld CPGFRC mit 'EF' gefüllt.

Ein OPEN für Input, Output,Update und Reuse für VSAM-Dateien im Batch ist jetzt möglich.

Beispiel:

```
             OPEN CPGWRK     INPUT
```

PARM Parameter definieren 3395

Diese Anweisung kann nur in Zusammenhang mit der Operation CALL durchgeführt werden und muss deshalb auch unmittelbar hinter dem CALL stehen.

Mit der PARM-Anweisung wird die Speicheradresse des Feldes aus Faktor 2 dem aufgerufenen Unterprogramm übergeben. Es können mehrere PARM-Operationen nach einem CALL angegeben werden.

```
Beispiel:  C          CALL 'PROGA'
           C          PARM              FELDA
           C          PARM              FELDB
```

Nach der Verarbeitung des Unterprogramms kehrt die Steuerung zur nächsten zu verarbeitenden Anweisung nach der PARM-Anweisung zurück. Eine Bezugszahl kann nicht benutzt werden.

PRNT Assembler Listausgabe 3400

In Faktor2 können folgende Operanden eingetragen werden: 'ON', 'OFF', 'GEN', 'NOGEN' .

Es hat sich gezeigt, dass der Programmierer normalerweise mit der Anlistung der TWA auskommt. Sollte sich jedoch die Notwendigkeit ergeben, Programmausschnitte anzulisten, so kann man in den Rechenbestimmungen die Assemblerliste steuern.

PRNT ON bedeutet: Ab diesem Statement soll die Assemblerliste angelistet werden.

PRNT OFF bedeutet: Ab hier soll keine Assemblerliste mehr gedruckt werden.

PRNT GEN bedeutet: Ab hier soll die Assemblerliste mit Makroauflösung angelistet werden.

PRNT NOGEN bedeutet: Ab hier wird die Assemblerliste ohne Makroauflösung gedruckt.

PROG QPG-Programm ausführen (siehe Handbuch QPG) 3402

Ein QPG-Programm soll ausgeführt werden.

Beispiel: PROG DOKU

In Faktor 2 steht der Programmname.

Das Ergebnisfeld enthält (optional) die QPG-Library.

Wird nur das Ergebnisfeld eingetragen, dann wird die PROG-Operation variabel ausgeführt.

QPG und der Befehl PROG sind im separaten QPG-Handbuch beschrieben.

PROT Protection-Code 3405

Die Operation PROT bezieht sich auf den Schutz von Programmen mit dem CPG3-Produkt Sign On.

In Faktor 2 wird der Protection Code entweder als 10-stellige hexadezimale Konstante oder als bis zu sechsstelliger symbolischer Name angegeben.

Aufbau der Protection-Karte und die Logik des CPG3..Sign On sind im Handbuch der CPG3-Serviceprogramme ausführlich beschrieben.

PURGE Queue löschen 3410

Mit dieser Operation wird eine Temporary Storage Queue gelöscht.

Beispiel: PURGESTOR

Die in Faktor 2 definierte Queue STOR wird gelöscht.

Wurde die Queue in der F-Karte variabel definiert, so muss das Feld CPGTSN vor der PURGE-Operation gefüllt sein.

Beachte: Der Schalter EF und das Feld CPGFRC werden durch die Operation PURGE gelöscht.

PUTIN PUT Indicator 3412

Mit dieser Operation kann eine Bezugszahl vom laufenden Programm zur nächst höheren Stufe übertragen werden.

Dabei wird der Inhalt der Bezugszahl des ausführenden Programms in die gleichlautende Bezugszahl der höheren Stufe übertragen.

Die Spalten 54 bis 59 können eine, zwei oder drei Bezugszahlen enthalten.

Beispiel:

```
C                   PUTIN           15  20
```

Die Inhalte der Bezugszahlen 15 und 20 des ausführenden Programms werden in die Bezugszahlen 15 und 20 der nächst höheren Stufe übertragen, das heisst, ist die Bezugszahl 15 im laufenden Programm gesetzt und die Bezugszahl 20 gelöscht, so ist nach Ausführung der Operation auch auf der höheren Stufe die Bezugszahl 15 gesetzt und die Bezugszahl 20 gelöscht, unabhängig vom vorherigen Zustand.

PUTMI PUT MAIN Indicator 3415

Mit dieser Operation kann eine Bezugszahl vom laufenden Programm zum nächst höheren Hauptprogramm übertragen werden. Dabei wird der Inhalt der Bezugszahl des ausführenden Programms in die gleichlautende Bezugszahl des Hauptprogramms übertragen.

Die Spalten 54 bis 59 können eine, zwei oder drei Bezugszahlen enthalten.

Beispiel:

```
C                     PUTMI           15  20
```

Die Inhalte der Bezugszahlen 15 und 20 des ausführenden Programms werden in die Bezugszahlen 15 und 20 des nächst höheren Hauptprogramms übertragen, das heisst, ist die Bezugszahl 15 im laufenden Programm gesetzt und die Bezugszahl 20 gelöscht, so ist nach Ausführung der Operation auch im Hauptprogramm die Bezugszahl 15 gesetzt und die Bezugszahl 20 gelöscht, unabhängig vom vorherigen Zustand.

QSSA Qualifiziertes SSA aufbauen. 3420

Diese Operation dient dem Aufbau von qualifizierten SSAs. Im Faktor 1 wird in Hochkommata der Segmentname 8-stellig eingetragen. Im Faktor 2 wird ebenfalls 8-stellig der Name des in DL/I definierten Sucharguments eingetragen. Im Ergebnisfeld wird das Schlüsselfeld des Programms definiert, aus dem die linken Stellen als Schlüssel übertragen werden.

Im Felddefinitionsfeld (Spalte 49-51) wird angegeben, in welcher Länge das Schlüsselfeld linksbündig verarbeitet wird. Insbesondere bedeutet dies, dass die Eintragung nicht als definierende Längenangabe für das Ergebnisfeld genutzt werden kann.

Unter der Bezugszahl größer (Spalte 54-55) muss eine Vergleichsoperation eingetragen werden, z. B. GE, LE, EQ, GT, LT, NE. Die Einträge in Faktor 1 und 2, im Ergebnisfeld, im Feldlängenfeld und für die Vergleichsoperation sind Muss-Eintragungen; sonst gibt CPG eine Fehlermeldung.

In Spalte 53 kann ein "A" oder ein "O" für boolsche Operationen AND bzw. OR bei der folgenden QSSA-Operation eingetragen werden. In diesem Fall bleibt Faktor 1 frei.

```
Beispiel:  "ARTIROOT"QSSA "ART001  "ATKEY    7  GT
                     QSSA "ART005  "ATBEZ    1 OEQ
```

Nachdem alle SSAs gefüllt sind, kann der DL/I-Call erfolgen.

Programmierung mit erweiterten Funktionen.

Command-Codes im SSA:

CPG bietet die Möglichkeit, das SSA mit bis zu drei Com-Mand-Codes zu erweitern. Hierzu wird beim SSA-Befehl auf Stelle 56 ein "*" eingetragen. Die Stellen 57-59 können bis zu 3 der folgenden Commands aufnehmen: L,F,D,N,Q,U,V. Die Command-Codes sind in der Broschüre SH-12-5411 DL/I Application Programming, CALL and RQDLI Interface im Abschnitt 4 genau beschrieben.

Wird in Spalte 53 ein 'V' eingetragen, so werden sämtliche Einträge dem Feld CPGDLV entnommen. (Vgl. 2335)

READ Lesen 3440

1. Bildschirmdatei.

Mit dieser Operation wird ein Satz der in Faktor 2 genannten Datei gelesen.

Faktor 1 bleibt frei.

Ein 'I' in Spalte 53 bewirkt, dass nach der READ-Operation der Schalter CL abgefragt werden kann. Andernfalls bewirkt die CLEAR-Taste die Beendigung des Programms.

Ein 'L' in Spalte 53 bewirkt, dass Kleinbuchstaben nicht in Großbuchstaben übersetzt werden.

Ein 'S' in Spalte 53 schließt I+L ein.

Tabelle der Einträge in Spalte 53

Spalte 53		A	C	I	K	L	S	T
Task		x	x		x			x
Lower Case		x			x	x	x	
CL-Abfrage			x	x	x		x	

Beachte: Der interne TS-Name beim taskorientierten Ablauf heisst:

'TERMCPGP' bei MACRO Level Programmen
'TERMPCPG' bei Com. Level Programmen

Die Parameter T, A, C und K sollten nur nach ausreichender Schulung oder Beratung angewandt werden.

2. Platten-Datei.

Plattensätze einer indexsequentiell organisierten Datei werden sequentiell gelesen. In Faktor 1 kann der Feldname des Schlüssels eingetragen werden. Das Schlüsselfeld muss bei der erstmaligen Ausführung der Instruktion den Schlüssel des Satzes enthalten, mit dem die sequentielle Verarbeitung beginnen soll.

Faktor 2 enthält den Namen der Datei.

Durch Kombination der Befehle SETLL und READ kann eine VSAM-Datei auch abschnittsweise verarbeitet werden. Dabei wird mit der Operation SETLL ein Pointer auf den ersten Satz des Abschnitts gesetzt, während 'READ' den Satz und die logischen Folgesätze liest.

Bei Dateiende wird der Schalter 'EF' gesetzt und das interne Feld CPGFRC mit 'EF' gefüllt.

Bei VSAM-Dateien muss 'EF' nach der READ-Operation abgefragt werden, da sonst beim erneuten Lesen das Programm abgebrochen wird.

Das Ergebnisfeld kann den Namen einer Feldgruppe enthalten. In diesem Falle werden so viele Sätze gelesen, wie die Feldgruppe Elemente enthält. Voraussetzung ist dabei, dass in den Ausgabebestimmungen beschrieben wird, wie die Feldgruppe aufbereitet werden soll.

Siehe hierzu (8000), Beispiel 3.

3. Storage **(folgende Hinweise sind veraltet!)**

Ein ' ' in Spalte 53 bewirkt, dass der Bereich nach dem Lesen freigegeben wird.

Ein 'S' in Spalte 53 bewirkt, dass der Bereich nach dem Lesen nicht freigegeben wird.

(Gilt beides nicht für Temporary Storage Queuing).

READB Satz rückwärts lesen. 3441

Diese Operation gilt nur für VSAM-Dateien. Sie arbeitet wie die READ-Operation, jedoch werden dabei die Sätze rückwärts gelesen, das heisst, der logisch nächste Satz ist der Satz mit dem nächst kleineren Schlüssel. Bei Datei-Anfang wird der Schalter 'EF' gesetzt und das interne Feld CPGFRC mit 'EF' gefüllt.

Unterschiedlich zur READ-Operation muss der erste mit READB gelesene Satz in der Datei vorhanden sein.

Wenn der gelesene Satz nicht vorhanden ist, so wird der Schalter 'EF' gesetzt und keine Eingabe durchgeführt.

Das Ergebnisfeld kann den Namen einer Feldgruppe enthalten. In diesem Falle werden so viele Sätze gelesen, wie die Feldgruppe Elemente enthält. Voraussetzung ist dabei, dass in den Ausgabebestimmungen beschrieben wird, wie die Feldgruppe aufbereitet werden soll.

READI Lesen Bildschirm transaktionsorientiert. 3442

CPG liest bei transaktionsorientierter Schreibweise des Programms automatisch bei Programmstart, d.h. vor der 1. Rechenbestimmung, die modifizierten Felder in die Eingabe des zuletzt definierten Bildes ein.

Die Operation READI erlaubt es nun, gezielt eine ganz bestimmte Eingabe anzusteuern. Wenn aus programminternen Gründen die Daten auf temporären Speicher gesichert wurden, kann nun von Temporary Storage gelesen und anschließend der Bildschirminhalt übertragen werden.

Die Spalte 53 kann ein 'L' enthalten, so dass CPG keine Übersetzung in Großbuchstaben durchführt.

A c h t u n g: Groß/Kleinschreibung funktioniert beim transaktionsorientierten Programm nur, wenn in der TCT der Parameter UCTRAN nicht definiert wurde. Außerdem muss man in der H-Karte in Spalte 39 ein 'L' eintragen.

Siehe hierzu auch die Befehle : COMRG und UCTRN.

1. Aus einer bereits gelesenen Datei soll eine bestimmte Struktur nochmals ins Programm übertragen werden.

2. In einem transaktionsorientierten Programm ohne QSF-Maps soll eine bestimmte Eingabe angesteuert werden.

Beschreibung:

1. Übertragen von Segmenten

Insbesondere bei VSAM-Dateien mit verschiedenen Satzarten ist es interessant, zunächst einen Teil des Datensatzes zu lesen. Entsprechend dem Inhalt der gelesenen Daten entscheidet man dann, in welche Struktur der Eingabesatz übertragen wird.

Diese zusätzliche Übertragung bereits gelesener Eingabedaten erreicht man mit dem Befehl READI.

1.1. READI ohne Angabe eines Segments
Entsprechend den Eingabebestimmungen für die Datei werden die Eingabedaten nochmals übertragen.

1.2. READI mit Angabe eines Segments
Entsprechend den Eingabebestimmungen für ein Segment werden die Eingabedaten des zuletzt gelesenen Satzes nochmals übertragen. Segmente müssen unmittelbar hinter der Eingabedatei beschrieben sein, auf die sie sich beziehen.

Beachte:

READI ist immer nur nach einem READ-, READB- und CHAIN-U-Befehl möglich. Wird diese Vorschrift nicht beachtet, so bricht das Programm bei der Ausführung ab.

Desweiteren ist die Operation READI für sequentielle Batch-Dateien, Tapes und Temporary Storage unterstützt.

READP Seite lesen 3443

Diese Operation wurde in die Operation READ integriert. Die bisherige Bezeichnung ist weiterhin verwendbar. Bei neuen Programmen sollte jedoch die Operation READ mit Feldgruppe im Ergebnisfeld verwendet werden.

REPLC Blank durch Zeichen ersetzen 3444

Mit dieser Operation kann ein beliebiges Zeichen in einem alphanumerischen Feld durch ein anderes beliebiges Zeichen ersetzt werden.

Faktor 1 kann das Zeichen, das ersetzt werden soll, als alphanumerische oder hexadezimale Konstante oder variabel in einem einstelligen Alphafeld enthalten. Wird Faktor 1 nicht angegeben, so wird Blank (X'40') ersetzt.

Faktor 2 enthält das ersetzende Zeichen als alphanumerische oder hexadezimale Konstante oder als Variable in einem einstelligen Alphafeld.

Das Ergebnisfeld kann den Namen eines Alphafeldes, einer alphanumerischen Feldgruppe oder eines Feldgruppenelements enthalten.

```
Beispiel:            REPLC'-'      FELD

Feldinhalt vorher:                (Hugo        )
Feldinhalt nachher:               (Hugo--------)

   C                 MOVE '-'      AF1     1
   C                 MOVE '.'      AF2     1
   C    AF1          REPLCAF2      FELD

Feldinhalt nachher:               (Hugo.......)
```

RNDOM Wahlweise Verarbeitung 3445

VSAM-Dateien können in einem Programm sowohl wahlweise als auch sequentiell verarbeitet werden.

Mit Hilfe der CHAIN-Operation wird die Datei wahlweise verarbeitet. Die wahlweise Verarbeitung wird jedoch unterbrochen, wenn für diese Datei ein READ-Befehl gegeben wurde. Von diesem Zeitpunkt an wird die Datei nur noch sequentiell verarbeitet und eine nachfolgende CHAIN-Operation bewirkt lediglich ein Wiederaufsetzen in der Datei an anderer Stelle, ohne dass dabei daten gelesen werden.

Mit der Operation RNDOM kann die sequentielle Verarbeitung wieder beendet werden. Die folgenden 'CHAIN'-Operationen lesen dann wieder wahlfreie Einzelsätze, solange bis ein neuer READ-Befehl gegeben wird.

Die RNDOM-Operation wird außerdem benutzt, um einen mit CHAIN (U) gesperrten Satz wieder zu entriegeln, wenn kein Update erfolgt. RNDOM gibt den VSAM-String frei.

RNDOM löscht den EF-Schalter und das Feld CPGFRC.

Vor dem Aufrufen eines anderen Programms mit der EXPR-Operation und vor dem Rücksprung in das aufrufende Pro-

gramm sollen die Programmdateizeiger zurückgesetzt (Reset) werden.

Sie sollen durch Angabe von RNDOM-Operationen für alle in dem Programm definierten DISK-Dateien zurückgesetzt werden.

Faktor 2 enthält den Namen der Datei. Faktor 1, Ergebnisfeld und Ergebnisbezugszahlen bleiben frei.

Beispiel: RNDOMDATEIX

Im Faktor2 kann der Eintrag *ALL erfolgen, um alle im Programm verwendeten Dateien freizugeben.

RNDOM*ALL kann in Batch-Programmen nicht benutzt werden.

Vor Verlassen eines Programms mit den Operationen EXPR und EXITP mit Programmnamen werden von CPG alle im Programm verwendeten Dateien mit einer RNDOM*ALL Operation freigegeben. Diese Operation steht auch dem Anwender zur Verfügung. Sie sollte immer dann verwendet werden, wenn alle Dateien auf Anfangsstatus zurückgesetzt werden sollen.

Der Eintrag *ALL in Faktor 2 gilt nur für VSAM-Dateien, die in die CICS FCT eingetragen sind.

Bei der Operation SYNCP ist die Verwendung von RNDOM*ALL zu empfehlen.

Beispiel: RNDOM*ALL

Die Operation RNDOM wird zudem dazu benutzt, in einer Datenview auf den Anfang zu positionieren. Diese Positionierung ist z.B. erforderlich, wenn in einem Programm das Suchargument der FIND-Operation wechselt und der interne Zeiger noch auf das zuletzt gefundene Tabellenelement zeigt.

ROLL Feldgruppe verschieben 3446

Mit dieser Operation werden die Felder einer Feldgruppe auf den nächst niedrigeren Index verschoben. Das zweite Feld der Feldgruppe wird in das erste übertragen, das dritte in das zweite usw. Der Inhalt des letzten Feldes der Feldgruppe bleibt unverändert erhalten.

Beispiel: ROLL FG

Inhalt der Feldgruppe 'FG'

	vor	nach	Ausführen der Operation
FG,1	AAA	BBB	
FG,2	BBB	CCC	
FG,3	CCC	CCC	

Die Operation wurde erweitert. Wird der Name der Feldgruppe mit einem variablen Index versehen, so gibt dieser Index die Stelle innerhalb der Feldgruppe an, ab der gerollt werden soll.

ROLLB Feldgruppe rückwärts verschieben 3447

Mit dieser Operation werden die Felder einer Feldgruppe um ein Feld verschoben. Das erste Feld der Feldgruppe wird in das zweite Feld übertragen, das zweite Feld in das dritte usw. Der Inhalt des ersten Feldes der Feldgruppe bleibt erhalten.

Beispiel: ROLLB FG

Inhalt der Feldgruppe 'FG'

	vor	nach	Ausführen der Operation:
FG,1	AAA	AAA	
FG,2	BBB	AAA	
FG,3	CCC	BBB	

Die Operation wurde erweitert. Wird der Name der Feldgruppe mit einem variablen Index versehen, so gibt dieser Index die Stelle innerhalb der Feldgruppe an, ab der gerollt werden soll.

Beispiel: ROLLB FG,I
(Indexwert = 3)

Vorher:	Nachher:
AAAAAA	AAAAAA
BBBBBB	BBBBBB
CCCCCC	CCCCCC
DDDDDD	CCCCCC
EEEEEE	DDDDDD

SAVET Retten Bildschirminhalt 3448

Diese Operation rettet das aktuell angezeigte Bild in einen Temporary Storage Bereich. Im Faktor 2 wird vierstellig der Name einer TS-Queue definiert, den CPG intern vorne um die Terminal-ID erweitert.

Mit der Operation LOADT kann eine gerettete Bildschirmseite wieder angezeigt werden. Diese Operation kann benutzt werden, um aus einem Programm heraus in ein anderes zu verzweigen und hinterher den alten Bildschirminhalt mit Hilfe der LOADT-Operation zurückzusetzen.

Im Ergebnisfeld kann ein 4-stelliger Feldname für einen variablen Storage eingetragen werden.

Die Operation sichert auch die sieben Farben (falls sie per Farbattribut gesetzt sind) sowie die erweiterte Helligkeit, falls sie für das Terminal zugelassen sind.

Beispiel: SAVETHUGO

Der interne TS- Name heisst: "TERMHUGO".

SCAN Alphafeld nach einer Zeichenfolge durchsuchen 3450

Faktor1	Suchargument (Zeichenfolge)
Operation	SCAN muss eingetragen werde
Faktor 2	Name des Feldes, das durchsucht werden soll
Ergebnis	Numerisches Feld für den Startwert und die gefundene Position.
Bezugszahl	kann in Spalte 58/59 (==) angegeben sein

Beispiele:

```
C           FELD      SCAN FG,I      POS            99
C           FG        SCAN SATZ                     01
C           F1        SCAN F2        I     V
```

Zweck:

Ein alphanumerisches Feld wird nach einer Zeichenfolge durchsucht.

Beschreibung:

Die Operation SCAN durchsucht das Feld in F2 nach dem Inhalt des Feldes in F1. Wird die Zeichenfolge gefunden, so wird im Ergebnisfeld die Position des ersten Zeichens von F1 im Feld F2 angegeben.

Als Ergebnisfeld kann ein numerisches Feld mit 0 Dezimalstellen angegeben werden. Ist dieses Feld vor dem Befehl größer 0, so wird erst bei der entsprechenden Position im zu durchsuchenden Feld mit der Suche begonnen. Nach der Operation enthält das Feld die Position, bei der die Zeichenfolge (d.h. das erste Zeichen der Folge) gefunden wurde. Wird die Zeichenfolge nicht gefunden, so wird das Ergebnisfeld wieder auf 0 gesetzt.

Bei einer Eintragung im Ergebnisfeld ist die Bezugszahl in Spalte 58/59 nicht zwingend erforderlich.

Die Operation SCAN ist in beiden Faktoren indizierbar.

Die Feldlänge von Faktor 2 muss größer als die Feldlänge von Faktor 1 sein. Bei Feldgruppen in Faktor 2 ist zu beachten, dass die Elementlänge größer ist als die Feldlänge von Faktor 1.

In Spalte 53 kann hinter dem Index noch ein V eingetragen werden. Das zu vergleichende Feld wird dann nur mit den im F2 angegebenen Zeichen und nicht in der ganzen Feldlänge verglichen.

Eine vorgegebene Startposition wird berücksichtigt.

Mit dem Service 'V' kann das Suchen in variabler Länge ausgeführt werden. Die Länge ergibt sich durch die Anzahl Stellen, die im Suchargument gefüllt sind. Das Ende des Arguments ist dabei Blank oder x'00'. Wenn nach einem Argument gesucht wird, das z.B. Leerstellen enthält, dann

kann auch ein beliebiges Sonderzeichen benutzt werden,um das Suchargument einzuschließen (am Anfang und am Ende das gleiche Sonderzeichen). Das Sonderzeichen wird dabei nicht als Suchargument benutzt.

SDUMP Special Terminaldump 3460

Diese Operation ist eine Testhilfe-Operation, die die Operation DEBUG beinhaltet.

In Faktor 2 kann ein vierstelliger Dump-Code eingetragen werden. Wird in Faktor 2 keine Eintragung vorgenommen,so wird die Statementnummer als Dump-Code angezeigt.

Beispiel: SDUMPCODE

Ausführliche Beschreibung unter Testhilfen (2830).

SELCT Feldauswahl 3461

Mit dieser Operation können über die Eingabebestimmungen Feldteile in andere Felder übertragen werden. SELCT ist die Umkehrung der EDIT-Operation.

Siehe auch Abschnitt 1: Feldaufbereitung.

Das Ergebnisfeld enthält den Namen des aufzubereitenden Feldes. Faktor 1 ist blank.

Das Ergebnisfeld kann auch den Namen einer Feldgruppe mit festem oder variablem Index enthalten.

Beispiel: SELCT FG,I

Dabei ist FG der Name einer alphanumerisch definierten Feldgruppe und I der Name des numerisch mit 0 Dezimal-stellen definierten Indexfeldes, das den variablen Index enthält.

Wird eine SELCT-Operation mit einem Namen in Faktor 2 ausgeführt, so kann die Eingabebestimmung eines Feldes spezifiziert werden.

```
          IALPHA  F     FELD  (Spalte: I=6  F=15  F.=32)

          C         SELCTFELD      ALPHA
```

In Verbindung mit einem 'V' in Spalte 53 kann in Faktor2 ein Feldname (6-stellig-alpha) eingetragen werden.

```
          IALPHA  F     FELD

          C         MOVEL'FELD'    NAME
          C         SELCTNAME      ALPHA     V
```

Zu beachten ist, dass alle Eingabebestimmungen eines Feldes nacheinander beschrieben werden.

Ein 'I' in Spalte 53 unterdrückt die Indexprüfung.
Siehe EDIT bzw. Seite 2420.

SETIN Bezugszahl mit Index setzen 3462

Diese Instruktion ist die Umkehrung des SETIX.

Das Ergebnisfeld enthält den Namen eines numerisch mit 0 Dezimalstellen definierten Indexfeldes und die Spalten 54 und 55 die erste Bezugszahl einer fortlaufenden Bezugszahlengruppe. Faktor 1 und 2 bleiben frei.

Nach Ausführung der Operation wird die Bezugszahl gesetzt, die sich aus der Addition der in den Spalten 54 und 55 eingetragenen Bezugszahl und dem Inhalt des Index Feldes verringert um 1 ergibt.

Enthält das Ergebnisfeld keinen gültigen Index,so wird bei der Ausführung eine Fehlermeldung 'Index falsch' ausgegeben.

Ein I in Spalte 53 unterdrückt die Indexprüfung. Wenn das Ergebnisfeld gleich Null ist, wird die Fehlermeldung 'Index falsch' nicht angezeigt und die Verarbeitung fortgesetzt.

Beispiel: SETIN I 25

Inhalt des Feldes 'I': 4

Nach der Ausführung ist die Bezugszahl 28 gesetzt.

SETIX Index setzen 3463

Das Ergebnisfeld enthält dabei den Namen eines numerisch mit 0 Dezimalstellen definierten Indexfeldes.

1. Index setzen in Verbindung mit einer Bezugszahl.

Mit dieser Instruktion kann ermittelt werden, welches Feld einer Feldgruppe auf dem Bildschirm mit dem Lichtstift ausgewählt wurde.

Die Spalte 54 und 55 enthält die erste Bezugszahl einer in den Eingabebestimmungen Spalte 61-62 definierten Bezugszahlengruppe. Faktor 1 und 2 und die Spalten 58-59 bleiben frei.

In den Spalten 56-57 kann eine Bezugszahl eingetragen werden, die eine Begrenzung der SETIX-Operation angibt.

In Spalte 53 kann ein 'I' eingetragen werden. Dies bewirkt, dass bei einem Indexfehler keine Fehlermeldung ausgegeben, das Ergebnisfeld auf '0' gelöscht und die Verarbeitung fortgesetzt wird.

Nach Ausführung der Operation enthält das Indexfeld die Positionsnummer des Feldes in der Feldgruppe, z.B. 7, wenn das 7. Feld ausgewählt wurde. Sind keine ent-

sprechenden Bezugszahlen gesetzt, so wird bei der Ausführung eine Fehlermeldung 'Index falsch' ausgegeben.

Beispiel: SETIX I 25

Gesetzte Bezugszahl: 31

Inhalt des Feldes 'I' nach Ausführung: 7

2. Ermittlung der Dimension einer Feldgruppe oder einer Feldlänge

Es kann ermittelt werden, wieviele Elemente bei einer Feldgruppe definiert wurden. Im Faktor 2 muss ein Feldgruppenname eingetragen werden.

```
          D      FG        10  5
          C                  SETIXFG     I
```

Inhalt des Feldes 'I' nach der Ausführung: 10

Die Servicefunktion 'F' in Spalte 53 bewirkt, dass in einem numerischen Feld (im Beispiel Ergebnisfeld I) die Länge eines im Faktor 2 angegebenen Felds (im Beispiel FELD) ermittelt werden kann.

Beispiel: C SETIXFELD I F

SETLL Set Lower Limit 3464

Mit dieser Operation kann bei sequentieller Verarbeitung die Reihenfolge unterbrochen und an beliebiger Stelle wieder aufgesetzt werden. Die SETLL-Operation liest keine Daten, sondern bestimmt durch den Inhalt des in Faktor 1 eingetragenen Schlüsselfeldes lediglich den Satz, der mit der nächsten READ-Operation gelesen werden soll.

Faktor 2 enthält den Namen der Datei.

Ist bei der Ausführung einer SETLL-Operation das Ende einer Datei erreicht worden, so wird der Schalter 'EF' gesetzt und das interne Feld CPGFRC mit 'EF' gefüllt. Befindet sich die Datei nicht im sequentiellen Modus, so wird das interne Feld CPGFRC nach den Regeln des Befehls CHAIN gefüllt !

In Spalte 54 braucht keine Bezugszahl eingetragen zu werden. Sollte eine Bezugszahl eingetragen sein, so wird diese nicht verändert.

SETOF Bezugszahl löschen. 3465

Mit dieser Operation können Bezugszahlen von 01 bis 99 sowie T- und C-Schalter gelöscht werden.

Die zu verändernden Bezugszahlen werden in den Spalten 54-55, 56-57 oder 58-59 eingetragen. Die Reihenfolge ist beliebig. Es können 1, 2 oder 3 Bezugszahlen mit einer SETOF-Operation gelöscht werden.

SETON Bezugszahl setzen 3466

Mit dieser Operation können Bezugszahlen von 01 bis 99 sowie T- und C-Schalter gesetzt werden.

Die zu verändernden Bezugszahlen werden in den Spalten 54-55, 56-57 oder 58-59 eingetragen. Die Reihenfolge ist beliebig. Es können 1, 2 oder 3 Bezugszahlen mit einer Operation gesetzt werden.

SORTA Feldgruppensortierung 3467

In Faktor 2 wird der Name der zu sortierenden Feldgruppe eingetragen. Die Feldgruppe wird in auf- oder absteigender Folge sortiert, abhängig von der in Stelle 53 festgelegten Folge.

Ist keine Folge festgelegt, so wird die Feldgruppe aufsteigend sortiert.

Ein 'B' in Spalte 53 bedeutet, dass die Feldgruppe aufsteigend sortiert wird, nur mit der Besonderheit, dass 'Blank' bei alphanumerischen Feldgruppen bzw. 'Null'-Inhalte bei numerischen Feldgruppen nach hinten sortiert werden.

Ein 'D' in Spalte 53 bedeutet, dass die Feldgruppe in absteigender Folge sortiert wird.

```
D         FG1     8    1
D         FG2     8    1
D         FG3     8    1
C              SORTAFG1
C              SORTAFG2              D
C              SORTAFG3              B

Feldinhalt vorher:        G/T/1/B/ /2/L/0
FG1       nachher:         /B/G/L/T/0/1/2
FG2       nachher:        2/1/0/T/L/G/B/
FG3       nachher:        B/G/L/T/0/1/2/
```

SQRT Quadratwurzel ziehen 3468

Mit dieser Operation kann aus einer Zahl die Quadratwurzel gezogen werden. Die Zahl steht dabei in Faktor2, die Wurzel im Ergebnisfeld.

Beispiel: SQRT ZAHL WURZEL

Ein 'H' in Spalte 53 bewirkt, dass das Ergebnis gerundet wird.

SUB Subtrahieren 3469

Nach Ausführung der Instruktion enthält das Ergebnisfeld die Differenz von Faktor 1 minus Faktor 2.

Ist Faktor 1 leer, so wird die Differenz vom Ergebnis minus Faktor 2 nach Ergebnis gebracht.

SUB ist indizierbar. Beispiel siehe 'ADD'.

SYNCP Synchronization Point spezifizieren. 3470

Die Operation SYNCP gibt bei der System Recovery des TP-Steuerprogramms an, bis zu welchem Punkt die Verarbeitung abgeschlossen war. Alle nach einem SYNCP erfolgten Veränderungen, Hinzufügen und Löschen von Sätzen in Dateien werden nach einem abnormalen Programm /Task Ende über Dynamic Transaction Backout wieder rückgängig gemacht.

Voraussetzung für die Funktion der SYNCP-Operation ist, dass für den TP-Monitor ein Log-File angelegt wird und dass die File Control Table und die Program Control Table entsprechende Eintragungen erhalten.

Für CICS lauten diese Eintragungen z.B. wie folgt:

In der FCT:	JID=System,LOG=YES	Für alle im Programm benutzten Update-Dateien.
In der PCT:	DTB=YES	Für alle Programme die SYNCP enthalten

Durch Eintragung eines 'R' in Spalte 53 kann ein irrtümlich gegebener SYNCP-Befehl wieder aufgehoben werden. (ROLLBACK). Vor der SYNCP-Operation muss ein RNDOM für alle im Programm verwendeten Dateien durchgeführt werden. Dazu kann bei der Operation RNDOM im Faktor 2 die Konstante *ALL eingetragen werden.

Beispiel:
```
RNDOM*ALL
SYNCP
```

TAG Merkmal setzen 3480

Diese Operation setzt ein Merkmal, zu dem mit Hilfe einer GOTO-Operation verzweigt werden kann. Der Name des Merkmals steht in Faktor1.

Der Name des Merkmals darf nicht mit einem bereits verwendeten Feldnamen übereinstimmen.

TESTB Bitschalter prüfen. 3482

Mit dieser Operation können einzelne Bits in einem einstelligen alphanumerischen Feld abgefragt werden. Das Ergebnisfeld enthält den Namen des zu prüfenden Feldes. Faktor2 enthält in Hochkommata die zu prüfenden Bits in Form von Ziffern von 0 bis 7 in Zaehlrichtung von links nach rechts.

Nach Ausführung der Operation werden die Ergebnisbezugszahlen wie folgt gesetzt:

Wenn alle zu prüfenden Bits auf 'Aus' gesetzt sind, wird die Bezugszahl auf größer (Spalte 54/55) gesetzt.

Sind die zu prüfenden Bits gemischt, d.h. teils auf 'Ein' und zum anderen Teil auf 'Aus' gesetzt, so wird die Bezugszahl auf kleiner (Spalte 56/57) gesetzt, aber nur, wenn die Anzahl der zu prüfenden Bits größer als 1 ist.

Sind alle Bits, die geprüft werden, auf 'Ein' gesetzt, so wird die Bezugszahl gleich (Spalte 58/59) gesetzt.

Beispiel: TESTB'23' BYTE 1 101112

Inhalt des Feldes BYTE 1.
In Bitschreibweise 11110011 (Hex: F3)
Es wird die Bezugszahl 12 gesetzt.
Inhalt des Feldes BYTE 2.
In Bitschreibweise 11100011 (Hex: E3)
Es wird die Bezugszahl 11 gesetzt.
Inhalt des Feldes BYTE 3.
In Bitschreibweise 11000011 (Hex: C3)
Es wird die Bezugszahl 10 gesetzt.

TESTN Feld auf numerische Zeichen prüfen. 3483

Mit dieser Operation können Alphafelder auf numerischen Feldinhalt geprüft werden. Das Ergebnisfeld enthält den Namen des zu prüfenden Feldes. Faktor1 und Faktor2 werden nicht benötigt.

Die Ergebnisbezugszahlen werden wie folgt gesetzt:

Enthalten alle Bytes des zu prüfenden Alphafeldes Ziffern, so wird die Bezugszahl auf größer gesetzt.

Enthält das zu prüfende Feld außer Ziffern noch führende Blanks, so wird die Bezugszahl auf kleiner gesetzt (falls die Feldlänge größer als 1 ist).

Die letzte Stelle kann auch ein Buchstabe mit Zone C oder D sein.

Ist der Feldinhalt gleich Blank, so wird die Bezugszahl auf gleich gesetzt.

Beispiel:

```
            TESTN          FELD1  3   101112
            TESTN          FELD2  3   111012
            TESTN          FELD3  3   111210

Feldinhalte  :    FELD1    FELD2    FELD3
                  (999)    ( 99)    (   )
```

In allen drei Fällen wird die Bezugszahl '10' gesetzt.

Ein 'L' in Spalte 53 bewirkt, dass auch das letzte Byte auf Ziffer (Zone F) geprüft wird.

Ist ein Zeichen des zu prüfenden Felds kleiner als Blank, so wird kein Schalter gesetzt.

Auf die Schalter kann verzichtet werden, indem man das zu prüfende Feld im Faktor 2 einträgt und im Ergebnisfeld ein einstelliges Alphafeld, in das das Ergebnis des TESTN abgestellt wird.

Beispiel:

```
C           TESTNFELD2     A1
```

A1 = 'N' : alle Zeichen in Feld1 sind Ziffern
A1 = 'M' = Feld1 enthält Ziffern und führende Blanks
A1 = 'B' = Feld1 enthält nur Blanks (hexadezimal 40)
A1 = ' ' = bei allen anderen Fällen

Im Normalfall wird die Servicefunktion Spalte 53 L eingetragen, damit auch das letzte Byte auf Ziffer geprüft wird.

(In speziellen Fällen, in denen numerisch verarbeitete Felder ungepackt auf Dateien abgestellt wurden, arbeitet man ohne den Service L. In diesen Fällen kann die PACK-/UNPACK-Funktion des Assemblers dazu geführt haben, dass im letzten Byte des Feldes die Zone im ersten Halbbyte steht. Ist das eine Zone C oder D, dann stehen hier Buchstaben A bis I oder J bis R, die beim Packen wieder richtig in einen Wert umgesetzt würden. Auch solche Felder können mit TESTN getestet werden: Ohne Service L werden auch Buchstaben A bis R in der letzten Stelle wie die entsprechenden Ziffern erkannt und behandelt. Siehe Tabelle des EBCDI-Codes.)

TESTT Terminal abfragen 3484

Mit dieser Operation kann der Name eines Terminals oder einer Terminalgruppe abgefragt werden. Stimmen Name oder ein Teil des Namens überein, so werden die Bezugszahlen in den Spalten 54 bis 59 wie bei einer Vergleichsoperation (COMP) gesetzt.

Der Name oder Namensteil steht in Hochkommata eingeschlossen in Faktor2. Faktor1 und Ergebnisfeld bleiben frei.

Die Konstante in Faktor2 wird mit der bei der Generierung der TCT vergebenen 4-stelligen Terminal-Identnummer verglichen. Enthält die Konstante nur ein Zeichen, z.B. 'A', so wird die Bezugszahl in den Spalten 58,59 gesetzt, wenn der Name des Terminals mit 'A' beginnt.

Beispiel: TESTT'KA' 101112

Erfolgt der Aufruf von Terminal 'KA07'
so wird die Bezugszahl 12 gesetzt (KA = KA).

Erfolgt der Aufruf von Terminal 'HUGO'
so wird die Bezugszahl 11 gesetzt (HU < KA).

TIME Zeit setzen 3486

Mit dieser Operation kann der Programmierer in den Rechenbestimmungen die Felder 'UTIME' und 'CPGTIM' aktualisieren. Die Felder enthalten dann die Zeit dieser TIME-Operation. Faktor 1 und Faktor 2 bleiben frei.

Mit TIME wird 'UDATE' beim Tageswechsel aktualisiert.

Im Ergebnisfeld kann ein numerisches Feld eingetragen werden. In den Fällen wird der Inhalt aus 'CPGTIM' in dieses Feld rechtsbündig übertragen.

Beispiel:
```
TIME
TIME          SEKUND    20
```

TWALD TWA von Temporary Storage einlesen 3490

Eine mit TWASV auf Temporary Storage gerettete TWA wird wieder eingelesen.

In Faktor 2 wird vierstellig der Name des Temporary Storage angegeben, von dem die TWA gelesen werden soll. Für diesen Temporary Storage Bereich braucht keine F-Karte angelegt zu werden.

Im Ergebnisfeld kann ein 4-stelliger Feldname für einen variablen Storage eingetragen werden.

Der Schalter EF wird gesetzt, wenn die TWA nicht gefunden wurde, außerdem wird das Feld CPGFRC mit 'EF' gefüllt. Die Operation darf nicht in einer Subroutine benutzt werden.

C TWALDNAME

TWALD darf nicht in HL1-Modulen eingesetzt werden.

TWASV TWA auf Temporary Storage retten 3491

Eine TWA wird auf Temporary Storage gerettet.

In Faktor 2 wird vierstellig der Name des Temporary Storage angegeben, auf den die TWA ausgegeben wird. Für diesen Temporary Storage Bereich braucht keine F-Karte angelegt zu werden. Intern wird eine Storage Queue 'NAMETERM' angelegt.

Im Ergebnisfeld kann ein 4-stelliger Feldname für einen variablen Storage eingetragen werden.

C TWASVNAME

TWASV darf nicht in HL1-Modulen eingesetzt werden.

UCTRN Groß- Kleinschreibung an oder aus 3498

Die Operation UCTRN kann im Faktor 2 den Eintrag 'ON' oder 'OFF' enthalten. Der Eintrag 'ON' setzt für den Bildschirm, an dem das Programm ausgeführt wird, das interne Feature UCTRAN auf ein. Damit übersetzt der TP-

Monitor alle Eingaben von Kleinbuchstaben auf Großbuchstaben. Der Eintrag 'OFF' im Faktor2 setzt das Feature in der TCT auf Nicht-übersetzen. Somit ist es möglich, transaktionsorientiert auch Texteingaben zu erfassen, die in Groß- /Kleinschreibung erfolgt. Die Operation COMRG bietet die Möglichkeit, den Status von UCTRAN abzufragen. Ist eine Anwendung beendet, sollte der Urzustand wieder hergestellt werden.

In Spalte 53 kann ein 'T' eingetragen werden, das bedeutet, dass bei Eingabe einer TransId eine Übersetzung erfolgt.

CPG führt auch eine Übersetzung in Großbuchstaben durch. Über die H-Karte kann programmabhängig über die Spalte 39 das Einlesen von Text gesteuert werden. Dies gilt aber nur für die interne Eingabeübertragung bei transaktionsorientierter Ausführung. Wenn die Operation READI verwendet wird, sollte Spalte 53 den Eintrag 'L' enthalten, wenn Kleinbuchstaben erwünscht sind.

Beispiel:

```
ITEST    DS                  32
I                         22  22 UCASE
C                     COMRG          TEST
C           UCASE     CABEQ'U'       OK
C                     UCTRNON
C           OK        TAG
```

COMRG überprüft, ob an diesem Bildschirm UCTRAN gesetzt ist. Wenn ja, verzweigt das Programm nach OK, sonst setzt die Operation UCTRN das TCT-Feature UCTRAN auf an.

'UCTRN ON' und 'UCTRN OFF' sind auch im HL1-Batch unterstützt. Nach UCTRN ON wird die Druckausgabe des Batchprogramms in Großbuchstaben übersetzt, mit UCTRN OFF wird die Übersetzung ausgeschaltet.

UCTRN gilt sowohl für alle Printer im Programm und den untergeordneten HL1-Modulen als auch bei LIST-Verarbeitung.

UPDAT Satz zurückschreiben 3500

Mit dieser Operation kann ein Satz einer Datei verändert werden, ohne dass dabei Ausgabebestimmungen erforderlich sind.

Faktor1 enthält den Namen des Schlüsselfeldes, Faktor2 enthält einen gültigen Dateinamen und das Ergebnisfeld den Namen des zu verändernden Satzes. Voraussetzung für die Durchführung der Operation ist, dass das Ergebnisfeld in den Eingabebestimmungen unter der in Faktor2 angegebenen Datei eingelesen wurde.

Beispiel: KEY UPDATDATEI SATZ

Beispiel 5 (8050) zeigt ein ausführliches Anwendungsbeispiel für diese Operation.

Das Ergebnisfeld ist ein alphanumerisches Feld und darf maximal 256 Stellen lang sein.

Für Dateien mit variabler Satzlänge ist diese Operation nicht unterstützt.

Es wird grundsätzlich nur der komplette Satz verarbeitet.

USSA Unqualifiziertes SSA aufbauen. 3510

Diese Operation dient dem Aufbau von unqualifizierten SSAs. In Faktor1 wird in Hochkommata der Segmentname 8-stellig eingetragen. Faktor2, Ergebnisfeld, Feldlängenfeld sowie die Ergebnisbezugszahlen bleiben frei.

Beispiel: 'ARTIROOT'USSA

Nachdem alle SSAs gefüllt sind, kann der DL/I-Call erfolgen.

Wird in Spalte 53 ein 'V' eingetragen, so werden sämtliche Einträge dem Feld CPGDLV entnommen. Der Aufbau des Feldes CPGDLV ist im Abschnitt 2335 beschrieben.

VBOMP VBOMP Zugriff 3520

Mit dieser Operation kann direkt auf eine VBOMP-Datenbank zugegriffen werden. Faktor1 kann den Namen eines Schlüsselfeldes enthalten. Faktor2 kann den Namen der Datei enthalten und das Ergebnisfeld muss den VBOMP-Operationscode enthalten.

Bei VBOMP sind maximal 7-stellige Dateinamen unterstützt.

Beispiel:

KEY VBOMPMASTER MRAN

Nähere Informationen entnehmen Sie bitte dem VBOMP-Handbuch.

VSLCT VBOMP Daten in TWA übertragen. 3530

Mit dieser Operation werden die Daten aus der File Work Area in die TWA-Felder übertragen. Anders als bei anderen Dateizugriffen bleibt diese Area pro Datei neben der TWA als Datenbereich erhalten. Der Anwender ist für die Übertragung der Daten in seine TWA selbst zuständig. Bei VSLCT sind max. 7-stellige Dateinamen unterstützt.

Beispiel:

VSLCTMASTER

Nähere Informationen entnehmen Sie bitte dem VBOMP-Handbuch.

WAIT Warten 3540

Diese Operation wird verwendet, wenn auf ein äußeres Ereignis gewartet werden soll. Das Programm wird dann jeweils für eine Sekunde unterbrochen, damit die anderen Arbeiten nicht blockiert werden. WAIT ist auch im Batch unterstützt.

Beispiele:

```
WAIT
WAIT AMINIT
```

Im ersten Beispiel wird 1 Sekunde gewartet.

Im zweiten Beispiel steht eine Uhrzeit im siebenstelligen numerischen Feld AMINIT wie folgt zur Verfügung: '0HHMMSS'.

Im Ergebnisfeld kann ein bis zu 7-stelliges, numerisch definiertes Feld eingetragen werden, das die Dauer des Wartens angibt. (Format bei 7 Stellen : '0HHMMSS').

WRITE Satz hinzufügen 3550

Diese Operation entspricht der Operation UPDAT mit dem Unterschied, dass der Satz nicht zurückgeschrieben, sondern zu einer VSAM-Datei hinzugefügt wird.

Faktor2 enthält den Namen einer Datei und das Ergebnisfeld den Namen des hinzuzufügenden Satzes.

Beispiel: KEY WRITEDATEI SATZ

Das im Ergebnisfeld spezifizierte Feld muss in den Eingabebestimmungen unter der in Faktor2 angegebenen Datei eingelesen werden.

Für Dateien mit variabler Satzlänge ist diese Operation nicht unterstützt.

Ist der Schlüssel bereits vorhanden, so wird der Schalter 'DR' (duplicate record) und das Feld CPGFRC auf 'DR' gesetzt.

Es kann ein Satz einer Datei oder einer Storage Queue hinzugefügt werden , ohne dass dabei die Ausgabe in der Ausgabebestimmung beschrieben werden muss.

Für Storage Queues gilt:

In der Eingabebestimmung für die Storage Queue dürfen sich nur alphanumerische und numerisch gepackte Felder befinden.

XFOOT Summe einer Feldgruppe rechnen 3560

Mit dieser Operation kann die Summe einer numerischen Feldgruppe errechnet werden.

Faktor2 enthält den Namen der Feldgruppe. Das Ergebnisfeld enthält den Namen des Feldes, in das die Summe gespeichert werden soll.

Beispiel: XFOOTFG1 SUMME

```
Inhalt der Feldgruppe 'FG1':  Feld1    125,00
                              Feld2     75,00
                              Feld3     85,00
Inhalt des Feldes 'SUMME'
nach Ausführung der Instruktion:       285,00
```

Z-ADD Löschen und Addieren 3580

Das Ergebnisfeld wird gelöscht und Faktor2 wird in das Ergebnisfeld addiert.

Faktor 2 und Ergebnisfeld können auch den Namen einer Feldgruppe mit festem oder variablem Index enthalten. Beispiel siehe 'ADD'-Operation.

Z-SUB Löschen und Subtrahieren 3581

Das Ergebnisfeld wird gelöscht, und Faktor2 wird mit entgegengesetztem Vorzeichen in das Ergebnisfeld

addiert.

Faktor 2 und Ergebnisfeld können auch den Namen einer Feldgruppe mit festem oder variablem Index enthalten. Beispiel siehe 'ADD'-Operation.

+ Addieren 3582

Die Regeln entsprechen denen der Operation ADD.

Beispiel:

```
A + B C 50
```

- Subtrahieren 3583

Die Regeln entsprechen denen der Operation SUB.

Beispiel:

```
A - B C 50
```

* Multiplizieren 3584

Die Regeln entsprechen denen der Operation MULT.

Beispiel:

```
A * B C 50
```

/ Dividieren 3585

Die Regeln entsprechen denen der Operation DIV.

Beispiel:

```
A / B C 50
```

= Löschen und Addieren 3586

Die Regeln entsprechen denen der Operation Z-ADD.

Beispiel:

```
= 5 C 50
```

Abschnitt 4 Formularbeschreibung 4000

CPG kann als Generatorsprache formatgebunden oder in freier Schreibweise codiert werden. Bei formatierter Schreibweise ist für die verschiedenen Programm-Anweisungen jeweils ein fest vorgegebenes Eingabeformat erforderlich. Die Formate werden über eine Satzart in Spalte 6 des jeweiligen Formulars definiert.

Wir unterscheiden folgende Satzarten:

H	Header	CPG-Steuerkarte
F	File Description	Dateizuordnung
E	Extended File Description	Erweiterte Dateizuordnung
D	Data Description	Datenbeschreibung
L	Line Specifications	Druckzeilen Steuerung
I	Input Description	Eingabebestimmung
C	Calculation Description	Rechenbestimmung
O	Output Description	Ausgabebestimmung

Bei der Satzart 'C' ist dabei beispielsweise vorgeschrieben, dass der 5-stellige Operationscode immer in den Spalten 28 bis 32 der C-Karte steht. Beginnt der Operationscode einmal irrtümlich eine Spalte früher so wird er vom Compiler nicht erkannt und der Compiler meldet eine ungültige Operation. Wie der Operationscode, so müssen auch alle anderen Angaben spaltengenau eingetragen werden.

Die verschiedenen Formate werden in den folgenden Kapiteln beschrieben.

Das spaltengebundene Format entspricht dem RPG-Formalismus.

Formatfreie Codierung im CPG (ab der Ausbaustufe CPG2) 4010

Für das CPG-Format steht ein separates Handbuch zur Verfügung (CPG2-Programmierung).

Ein Beispiel für ein Programm im CPG-Format finden Sie z.B. am Ende des Kapitels HL1-Datasets.

CPG-Steuerkarte 4100

Die CPG-Steuerkarte enthält die nachstehend beschriebenen Parameter zur Steuerung des Programm-Ablaufs. CPG bietet die Möglichkeit, Parameter, die immer wieder vorkommen, in Form einer Standard-Steuerkarte dem System einzugeben. In diesem Falle werden bei fehlender Eintragung in der 'H'-Karte die Standard-Werte übernommen. Eine 'H'-Karte ist daher nur dann erforderlich, wenn mindestens ein Wert von den Standardwerten abweicht.

Die Generierung einer Standard-Steuerkarte kann jederzeit unabhängig von der CPG-Katalogisierung erfolgen. Die Generierung ist in Kapitel 9310 beschrieben.

Zu beachten ist, dass bei fehlender 'H' Karte die erste Programmkarte in den Spalten 75 bis 80 den Phasennamen des Programms enthalten muss, wenn dieses katalogisiert werden soll.

Formular-Beschreibung

Spalte 6 CPG-Kartenart

'H' muss eingetragen werden

Spalte 7-9 Partition-Size (VS)

Eine numerische Eintragung in diesen Stellen entspricht dem Size-Parameter in der EXEC-Karte für die Assembler-Umwandlung.
Der Eintrag darf nicht größer sein, als die VSIZE minus 64 K der Umwandlungspartition.

Spalte 8-9 1. OS/390- oder z/OS-Benutzer

Die Eintragung 'OS' in diesen Spalten generiert ein OS-Assembler-Programm.(Es wird keine // OPTION CATAL,PHASE, ASSGN SYSIN-Karte generiert).

Wenn in der CPGURSIT Spalte 21 ein 'M' eingetragen ist, so wird 'OS' als Default Wert angenommen.

2. 'A'
Suffix

Durch diese Einträge ist der Aufruf // EXEC ASSEMBLY variabel unterstützt. Soll der CPG-Source-Code nicht mit // EXEC ASSEMBLY in Maschinensprache übersetzt werden, sondern mit einer anderen Assemblier-Prozedur, so wird hier der Suffix der Prozedur in Spalte 9 angegeben.

Es muss dann sichergestellt sein, dass es eine Assemblierprozedur mit dem Namen CPGUASSx gibt, die dann wie folgt generiert wird: // EXEC PROC=CPGUASSx.

Spalte 10 Stanzen Assembler-Deck

Bei der Eintragung 'C' wird eine // OPTION DECK Karte generiert und die Eintragung 'D' mit eingeschlossen.

Ein 'D' in dieser Spalte bewirkt, dass das generierte Assembler-Quellenprogramm auf einer mit SYSPCH adressierbaren Einheit ausgegeben wird.

Die Eintragung 'O' schließt 'C' und 'D' ein. Es wird am Programmanfang das Copy CPG*CPRQ und am Ende das Copy CPG*CPRE generiert. Diese Copy-Books enthalten JCL-Karten für die Power Reader Queue.

P ermöglicht die CPG-Umwandlung über die Punch Queue, so dass IJSYS04 nicht benötigt wird. Für diese Art der Umwandlung wird der Job CPGZPUN benötigt, der unter Zubehör in Kapitel 7540 beschrieben ist.

Spalte 11 Generierungsliste

Folgende Eintragungen sind möglich:

'A' Assemblerliste ohne Makro-Auflösung.

Der für die Verarbeitung wesentliche Teil des Assembler-Programms wird gedruckt.
TP-Dummy-Sections, Makros und CPG-Unterprogramme werden nicht gedruckt.

'C' Komplettes Assemblerprogramm.

Das vom CPG erzeugte Assemblerprogramm wird komplett angelistet.

'D' Assemblerliste mit allen Dummy-Sections.

Wie 'A' jedoch werden alle Dummy-Sections mit angelistet.

'M' Assemblerliste mit Makroauflösung.

Wie 'A', jedoch werden zu jedem Makro die generierten Statements mit angelistet.

'N' Keine Assemblerliste

Das vom CPG erzeugte Assembler-Programm wird nicht aufgelistet. Es wird nur die CPG-Umwandlungsliste einschließlich CPG- und Assembler-Diagnostik gedruckt.

'S' Komplettes Assemblerprogramm mit Short-Cross-Reference.

Wie 'X', jedoch mit Anlistung der Assembler-Short-Cross-Reference (nur VSE).

'T' Transaction Work Area.

Von der Assemblerliste wird nur die Transaction-Work-Area und die Assembler-Diagnostik angelistet.

'X' Komplettes Assemblerprogramm mit Cross-Reference.

Wie 'C', jedoch zusätzlich mit Anlistung der Assembler-Cross-Reference.

Spalte 12-14 Leser Adresse

In diesen Spalten ist die physische Adresse des Lesers Einzutragen, über den das CPG-Programm eingelesen wird. Bei fehlender Eintragung wird die Eintragung der Standard H-Karte (00C) angenommen. Bei Eintragung blank in der Standard H-Karte wird die Einheit READER angenommen.

Spalte 15 Checkliste, Blockschaltbild oder Programmanalyse.

Ein 'E' in dieser Spalte bewirkt, dass die Fehlermeldung rechts neben der CPG-Liste und kein Blockdiagramm gedruckt wird.

Ein 'F' in dieser Spalte bewirkt, dass rechts neben dem CPG-Quellenprogramm ein Blockschaltbild des Programms angelistet wird. Nähere Erklärung siehe (2242).

Ein 'L' in dieser Spalte bewirkt, dass der Ausdruck der Zeilennummer,wenn blank oder 00000, unterdrückt wird. Sonst treffen die gleichen Bedingungen wie bei 'O' zu.

Ein 'N' in dieser Spalte bewirkt, dass die Programmierer-Checkliste am Ende des Programms nicht gedruckt wird.

Ein 'O' in dieser Spalte bewirkt, dass alle CPG Fehlernoten innerhalb der ersten 80 Stellen der Umwandlungsliste ausgegeben werden. Damit wird die Darstellung auf dem Bildschirm bei Verwendung von Online Programmier-Systemen erleichtert. Die Fehlermeldung wird jeweils vor dem fehlerhaften Statement ausgegeben.
Bei vielen CPG-Fehlermeldungen wird die fehlerhafte Eintragung mit Dollar gekennzeichnet. Für Dollar kann auch ein anderes Zeichen festgelegt werden, siehe CPGURSIT Stelle 22.
Sonst treffen die gleichen Bedingungen wie bei F zu.

Ein 'P' in dieser Spalte bewirkt, dass je CPG-Statement die erforderlichen Bytes am rechten Rand der Umwandlungsliste angelistet werden. Diese Programmanalyse kann dem Programmierer Hinweise zur Optimierung des Programms geben. Siehe Abschnitt 2248.

Spalte 16 CPG-Umwandlungsliste, Optimierungsfunktion

CPG-Programme werden grundsätzlich so optimiert, dass alle in den Eingabebestimmungen spezifizierten Felder, die im weiteren Programmverlauf nicht mehr benutzt werden, vom Compiler ignoriert werden. Dies wird dem Programmierer angezeigt durch die fehlende Statementnummer am linken Rand der CPG-Umwandlungsliste.

Ein 'D' in Spalte 16 bewirkt, dass die Optimierungsfunktion des CPG aufgehoben wird und alle in den Eingabebestimmungen aufgeführten Felder unabhängig von ihrer Verwendung ins Programm übernommen werden. Dies ist unter Umständen erforderlich bei der Verwendung von Eingabefeldern in Assembler-Unterroutinen.

Ein 'S' in Spalte 16 bewirkt, dass die derart ignorierten Felder in der CPG-Umwandlungsliste auch nicht gedruckt werden. Dies ist vor allem sinvoll bei der Verwendung der Copy-Funktion für die Eingabe.

Ein 'X' in Spalte 16 bewirkt, dass im Anschluss an die Umwandlung eine CPG-Cross-Reference-Liste gedruckt wird. Die Eintragung 'X' schließt jetzt die Optimierung mit ein, das heisst nicht benutzte Felder werden ignoriert. Die Eintragung Partition Size (Sp. 7-9) muss mindestens 192 betragen.

Ein 'Y' in Spalte 16 schließt 'D' und 'X' ein.

Ein 'Z' in Spalte 16 schließt 'S' und 'X' ein.

Spalte 21 Sprache bei einer CPG-Umwandlung

Ein 'D', 'E', 'I', 'J' in dieser Spalte bewirkt, dass die Ausgabe sämtlicher Texte in englischer Sprache erfolgt.

Ein ' ' oder ein 'G' in dieser Spalte bewirkt, dass die Ausgabe sämtlicher Texte in deutscher Sprache erfolgt.

Jedes andere Zeichen in dieser Spalte bewirkt, dass eine vom Programmierer selbst unter CPGSX* katalogisierte Textphase angezogen wird, wobei der * durch das in Spalte 21 einzutragende Zeichen ersetzt wird.
Spalte 21 hat keine Auswirkungen mehr auf die Darstellung der Dezimalzeichen.

Spalte 22 Privat-Bibliothek

Bei einer CPG-Umwandlung muss diese Spalte immer blank sein. Bei einer HL1-Umwandlung kann hier der Buchstabe einer privaten HL1-Library eingetragen werden.

Spalte 27-30 TWA-Größe des Vorprogramms

Wird von einem anderen Programm in dieses Programm verzweigt, so wird hier rechtsbündig die TWA-Größe des aufrufenden Programms eingetragen. Kann das Programm von mehreren Programmen aufgerufen werden, so ist die TWA-Größe des größten aufrufenden Programms einzutragen.
Siehe auch Programmverbindungen.

Wenn nur ein Teil der TWA übernommen werden soll, so ist der dezimale Wert aus der TWA-Liste einzutragen, der hinter dem entsprechenden Feld aufgeführt wird. Wird in der D-Karte die Eintragung ORG verwendet oder eine Felddefi-

nition auf Halb-, Voll- bzw. Doppelwortgrenze vorgenommen, so ist der Wert anhand der Assemblerdefinition selbständig zu ermitteln. Der Wert errechnet sich wie folgt: 16 Bytes interne Information + 100 Byte Bezugszahlen + Summe der Längen der übernommenen Felder.

Numerische Felder, die hinter dem übernommenen TWA-Bereich beginnen, werden auf X'0C' initialisiert.

Eine Eintragung TWA-Größe ist nur dann erforderlich, wenn dieses Programm über die Operationen EXPR oder EXITP Programmname aufgerufen wird. In allen anderen Fällen darf keine Eintragung erfolgen.

Spalte 31 Länge des Phasennamens

Die Eintragung '8' in dieser Spalte bewirkt, dass der Phasenname aus Stelle 73 bis 80 der H-Karte eingelesen wird. Der Kommentar, siehe Spalte 52-74, wird dadurch um zwei Stellen verkürzt.

Ein 'O' in dieser Spalte bedeutet, dass keine // OPTION CATAL und keine PHASE-Karte erzeugt wird. (Bei Online-Programmierung erfolgt eine Numerierung von 75-80).

Ein 'P' in dieser Spalte hat die Funktion von'8'und 'O'.

Alle anderen Eintragungen lesen den Phasennamen 6-stellig aus Spalte 75-80.

Spalte 32 Programmgröße (siehe auch Kapitel 6930)

Durch den Eintrag 'S' oder 'T' in dieser Spalte können CPG-Programme mit einer Assembler-Programmgröße größer 24 K generiert werden.

Die TWA-Size ist bei 'S' 4 K bzw. maximal 8 K, bei 'T' können maximal 12 K verwendet werden.

Es wird für jede Eingabe-Datei, für jede Feldaufbereitung, für das Hauptprogramm, für jede Subroutine und für die Ausgabe automatisch eine eigene CSECT angelegt (maximal 200 CSECTS pro Programm).

Bei jeder Feldaufbereitung können maximal 400 Felder übertragen werden. Die CSECT für die Feldaufbereitung darf maximal 4K umfassen.

Die anderen CSECTS dürfen maximal 12 K, bei 'T' maximal 8 K groß sein. Der allgemeine Programmteil, d.h. das Hauptprogramm, darf nicht größer werden als 8 K.

Wird einer dieser Werte überschritten, erscheint keine CPG-Fehlermeldung, nur Assembler Errors. Siehe Abschnitt 6930. Die Operationen UPDATE und WRITE sind nicht unterstützt.

Spalte 33 Automatisches RNDOM *ALL

Ein ' ' in dieser Spalte bewirkt, dass bei EXPR und EXITP automatisch alle File Workareas freigegeben werden.

Ein 'N' in dieser Spalte bewirkt, dass die implizit generierte RNDOM*ALL-Operation bei den Operationen EXPR und EXITP nicht automatisch erzeugt wird.

Spalte 34 HL1-Modul-Ausführung (CPG3-Benutzer)

'A' ---> Das Modul tauscht mit dem rufenden Programm automatisch Daten aus zwischen Feldern, die in beiden Programmen den gleichen Namen und die gleichen Feldeigenschaften haben.
Die Datenübergabe erfolgt wie bei QPG nur bis zum Feld CPGEDS, wenn dies im Modul definiert ist.
'D' ---> Daten in der PWA werden innerhalb der Task erhalten.
'E' ---> Kombination von 'A' und 'D'
'S' ---> PWA bleibt bestehen, aber PWA wird initialisiert.
'T' ---> Kombination von 'A' und 'S'
Die Datenübergabe erfolgt wie bei QPG nur bis zum Feld CPGEDS, wenn dies im Modul definiert ist.

Bei 'D' und 'S' Modulen brauchen die F-Karten nicht mehr im Hauptprogramm definiert zu werden. Diese Module sind besonders für HL1-Datasets geeignet. Die Performance steigt sowohl online als auch im Batch. Bei 'D' Modulen können feste Werte, z. B. Zaehler im Modul gespeichert werden, ohne dass diese z. B. im Kanal gerettet werden müssen.
Datasets für Query und für QFF lassen sich wesentlich einfacher realisieren.
Achtung: Dateien bei 'D' und 'S' Modulen belegen eigene VSAM-Strings.

Spalte 37 Dezimalzeichen für CPG-Umwandlung

Wird ein ' ' in dieser Spalte eingetragen, so wird als Dezimalzeichen die Eintragung CPGURSIT (Stelle 10) für die Umwandlung angenommen.
Wird ein ',' in dieser Spalte eingetragen, so werden Dezimalstellen mit Komma abgetrennt.
Wird ein '.' in dieser Spalte eingetragen, so werden Dezimalstellen mit Punkt abgetrennt.

Spalte 38 Referenzliste der QSF-Maps und der LIST-Dokumente

Wird ein 'M' in dieser Spalte eingetragen, so werden am Anschluss an die CPG-Umwandlungsliste alle im Programm verwendeten QSF-Maps und LIST-Dokumente aufgelistet.

Wird ein 'N' in dieser Spalte eingetragen, so werden zu den Maps die darin definierten Feldnamen und Konstanten aufgelistet.

Spalte 39 Kleinbuchstaben bei Bildschirm.

Ein 'L' in dieser Spalte bewirkt, dass bei einem transaktionsorientierten Programm vom Bildschirm Groß- und Kleinschreibung eingelesen wird. Die Daten werden nicht von CPG übersetzt. In der Terminal Control Tabelle darf der Parameter UCTRAN nicht eingetragen sein, bzw. im Programm muss UCTRN OFF verwendet werden.

Spalte 40 Numerische Felder

Wird ein 'S' in dieser Spalte eingetragen, so werden numerische Felder, die ungepackt eingelesen werden, mit Vorzeichen C statt F versehen. (Vgl. Kapitel 2145)

Spalte 41 8-Zoll Listausgabe

In diese Spalte ist eine 8 einzutragen, wenn die Umwandlungsliste des CPG-Programms auf 8-Zoll-Papier ausgegeben werden soll. Bei allen anderen Eintragungen wird 12-Zoll angenommen.

Spalte 42 Freies Format Listausgabe

' ' Es werden die programmierten Statements aufgelistet, einschließlich der Data Dictionary Eintragungen als I und O Karten.

'C' Es werden nur die generierten Statements im festen Format aufgelistet.

'D' Es werden die programmierten Statements aufgelistet, einschließlich der Data Dictionary Eintragungen als '-' Karten.

'F' Es werden nur die programmierten Statements aufgelistet.

'M' Es werden die programmierten und die generierten Statements aufgelistet.

Spalte 45 Release Parameter

Spalte 46 Generierung

Ein 'I' in dieser Spalte bewirkt bei einem Batch-Programm, dass bei einem Programmfehler (Cancel) eine Programmunterbrechung erfolgt.

Ein 'O' in dieser Spalte bewirkt eine Optimierung für numerische Operationen. Dieser Eintrag bezieht sich auf die numerischen Operationen ADD, SUB und Z-ADD.

Optimierung bedeutet, dass zu diesen Operationen ein direkter Assembler-Quellencode generiert wird. Somit wird bei der Ausführung nicht in die CPG-Methodenbank verzweigt. Insbesondere in CPG-Batchprogrammen bewirkt dies eine erhebliche Verbesserung der Laufzeit.

Ein 'S' in dieser Spalte bewirkt,dass das Debug Facility genutzt werden kann. Zu beachten ist, dass sich durch die Eintragung 'S' der Programmcode um 8 Bytes pro Rechenbestimmung vergrößert. (Nur CPG2-Benutzer).

Ein 'T' in dieser Spalte bewirkt, dass nur die numerischen Operationen im interaktiven Test mit QDF verfolgt werden können. Der Vorteil liegt darin, dass sich der

Programmcode im Gegensatz zu der Eintragung 'S' nicht vergrößert. (Nur CPG2-Benutzer).

Spalte 47 Methodenbank

Ein 'M' in dieser Spalte erzeugt systemunabhängige Macro-Level-Programme und verlegt alle Unterroutinen in die CPG-Methodenbank. (**Default, aber heute veraltet!**)

Ein 'C' in dieser Spalte erzeugt Programme, die ohne Methodenbank ausgeführt werden können.
Das generierte Assembler Programm enthält alle benötigten CICS-Makros. (Daher erhöhte Umwandlungszeiten und größerer Programmcode). Außerdem sind die Einschränkungen der CICS-Version Abschnitt 2905 zu beachten.

Ein 'E' in dieser Spalte generiert ein Command-Level-Programm, welches voll ESA-lauffähig ist. **! Empfehlung !**

Ein 'O' in dieser Spalte generiert ein Command-Level-Programm, welches nicht ESA-lauffähig ist.

'L' Command Level

Es wird ein Programm generiert, das im Command Level ausgeführt wird. Sonst treffen die gleichen Bedingungen wie bei 'M' zu.

Ein 'B' in dieser Spalte bedeutet, dass es sich um ein HL1-Batchprogramm handelt. Spalte 48 muss leer sein.

Ein 'D' in dieser Spalte bedeutet, dass ein Command-Level-Programm ohne die Methodenbank des CPG arbeitet.

Spalte 48 Eigene Adressierung

Die Zuteilung der Register erfolgt im allgemeinen dynamisch (siehe Abschnitt 1). Der Programmierer kann jedoch für spezielle Anwendungen eine eigene Adressierungsroutine ins Programm einfügen, die vorher unter dem Namen 'CPG*CADX' in die Source Library katalogisiert wird, wobei für * der Release Suffix (z.B. 4 für 2.6) und für X ein beliebiges Zeichen eingesetzt wird.

Wird beispielsweise in Spalte 48 eine '1' eingetragen, so wird das Copy 'CPG*CAD1' ins Programm eingefügt.

Damit kann der Programmierer beispielsweise die TWA von 4 K auf 8 K erweitern oder bestimmte z.B. in eigenen Unterprogrammen benutzte Register von der Adressierung ausnehmen.

Für folgende Eintragungen werden standardmäßig Copy-Books mitgeliefert:

'B' CPG-intern.
'C' CPG-intern.
'D' DL/I-Datasets.
'E' ETC Anwendungsprogramme.
'G' CPG/HL1 intern.
'I' CPG-intern.

'J' CPG-intern.
'K' CPG-intern.
'L' CPG-intern.
'O' OS Assembler.
'P' CPG-intern.
'Q' CPG-intern.
'S' Shadow Anwender.
'T' TCSS Anwender.
'U' PLT-Programm mit PWA-Verwendung.
'V' Com.-Level und DL/I-Dataset.
'W' siehe 'V' und TWA Size 8K.
'X' maximal 12 K TWA (Makro Level, intern gesetzt)
'Y' maximal 12 K TWA (Command Level, intern gesetzt)
'#' Command Level Anwendungsprogramme (#=X'7B').
'$' TOP intern
'§' TOP intern
'0' Com. Level TWA SIZE 8K (intern gesetzt)
'1' max. ProgrammGröße 20K und TWA Size 8K.
'1' bei HL1 max. Programmgröße von 16K und 8K PWA.
'2' CPG-intern.
'3' Batch-Programme und 12 K TWA. (mit Einschränkungen)
'4' CPG-intern.
'5' CL-ESA intern gesetzt. Batch 4K TWA.
'6' maximal 12 K TWA (Command Level, intern gesetzt)
'8' PLT-Programm mit PWA-Verwendung (Command Level).
'9' DL/I intern

Bei Batchprogrammen muss diese Spalte ' ' oder '3' sein.
Bei HL1-Modulen muss diese Spalte ' ' oder '1' sein.

Hinweis für HL1-Benutzer:

Werden in einem HL1-Hauptprogramm mehr als 4K TWA verwendet, so müssen die Keyfelder der VSAM-Dateien am Anfang der TWA mit 'D' oder 'E' Karten definiert werden.

Bei 'T' in Spalte 32 muss dieser Eintrag leer bleiben.

Spalte 49 Hardware-Attribute

Ein 'C' in dieser Spalte bewirkt, dass die Eintragungen in den Spalten 38 der Bildschirm-Feld-Ausgaben als Hardware-Attribute interpretiert werden. Damit steht dem Programmierer der gesamte Hardware-Attribut-Satz des Bildschirms zur Verfügung, wenn aus irgend einem Grund die CPG-Attribute nicht ausreichen.

Ein 'E' in dieser Spalte bewirkt, dass ein Aufbereitungsschlüssel in Spalte 38 der Ausgabe-Feldbestimmung bei numerischen Feldern eingetragen werden kann. Das Bildschirm-Attribut-Byte muss dann in Spalte 16 eingetragen werden.
Wird in Spalte 16 der Ausgabe-Feldbestimmung mindestens ein Attribut-Byte eingetragen,so kann auf die Eintragung 'E' verzichtet werden.

Spalte 50 Umwandlung

Ein ' ' in dieser Spalte bedeutet, dass die Umwandlung bei CPG-Fehler abgebrochen wird.

Ein 'A' in dieser Spalte bewirkt, dass die Umwandlung auch bei CPG-Fehler ausgeführt wird.

Spalte 51 Compiler

Diese Spalte muss für CPG-Umwandlungen 'Blank' oder 'C' sein.

Ein 'C' in dieser Spalte bewirkt, das bei einer HL1-Umwandlung ein CPG-Hauptprogramm generiert wird.

Ein 'H' (Default bei HL1-Umwandlungen) in dieser Spalte bewirkt, dass bei einer HL1-Umwandlung ein HL1-Baustein generiert wird.

Ein 'S' in dieser Spalte bewirkt, dass die Ausführung bei Programmende an das übergeordnete Programm (z. B. DLI) zurückgegeben wird. Dieser Eintrag gilt nur für Batchprogramme.

Spalte 52-74 Überschrift

Der hier eingetragene Text, z.B. Titel des Programms oder Name des Programmierers wird auf jeder Seite der Umwandlungsliste in der Kopfzeile angedruckt.

Bei achtstelligem Phasennamen (siehe Spalte 31) wird die Überschrift aus Spalte 52-72 eingelesen.

Spalte 75-80 Programmname

Enthält den Namen, unter dem die Programmphase katalogisiert wird. Entspricht der Eintragung in der PHASE-Karte beim Linkage Editor Lauf.

Bei Eintragung einer '8' in Spalte 31 wird der Phasenname aus Spalte 73-80 gelesen.

Die Eintragung aus Spalte 75-80 wird in alle Folgekarten des Programms übernommen. Bei fehlender Eintragung wird 'CPGOBJ' eingesetzt.

Die Eintragung '-' kann bei allen Parametern vorgenommen werden, um die Werte aus der Standard-H-Karte, die von uns mitgeliefert wird, zu neutralisieren. Individuell modifizierte Standard-H-Karten müssen in der Regel mit einem gültigen Wert explizit überschrieben werden.

Alle anderen Eintragungen werden nicht unterstützt.

Dateizuordnung 4200

Standard-Dateizuordnung

Eine Standard-Dateizuordnung kann erstellt werden. Abschnitt 5320 Zeigt ein Beispiel für eine solche Tabelle. Die Dateizuordnung ent-Hält für in der Standard-File-Table aufgenommene Dateien nur noch den Dateinamen.

Die Dateizuordnungskarte kann entfallen, wenn die Standard-File-Table eine entsprechende Eintragung enthält und die Eingabe-Satzbestimmung in Spalte 15,16 die Konstante 'SF' enthält.

Dateizuordnung

Es können maximal 50 Dateien (F-Karten) in einem Programm definiert werden. Bei Überschreitung erfolgt eine Fehlermeldung.

Formularbeschreibung

Spalte 6 CPG-Kartenart

'F' muss eingetragen werden

Spalte 7-14 Dateiname

Name der in der TP-File-Table definierten Datei. Der Name darf für VSE 7 Stellen und für z/OS 8 Stellen lang sein.

Wird eine Plattendatei zum ersten Mal in einem Online-programm benutzt, so muss sie vorher in der TP-File-Table angelegt werden.

Spalte 15 Ein-/Ausgabe-Art

Mögliche Eintragungen sind:

'I' Eingabe-Datei

'O' Ausgabe-Datei

'U' Update-Datei

'C' Combinierte Datei
Bildschirm-Einheiten im Dialogbetrieb

Spalte 16 Ein-/Ausgabe-Modus

Druckerdatei:

' ' Der Drucker wird wie jeder Systemdrucker program-miert, d.h. für jede Zeile ist eine Zeilenbestimmung auszufüllen. Die Zeilentransporte erfolgen über die

Eintragungen in den Spalten 17 und 18 der Ausgabebestimmungen. Für die Druckersteuerung muss eine L-Karte (4400) angelegt werden.

'B' der Drucker arbeitet im Buffermode wie ein Bildschirm-Terminal, d.h. mit einer 'Zeilenbestimmung' wird der gesamte Puffer (bis zu 24 Zeilen) gedruckt. Der Zeilentransport erfolgt über die Zeilennummer in den Spalten 40 und 41 der Ausgabebestimmungen.

Temporary Storage:

'Q' in dieser Spalte bewirkt, dass Daten in einem Datenbestand (Queue) hinzugefügt werden können. (Siehe Abschnitt 2190).

Die Verarbeitung von Queues und TS-Einzelsätzen ist in der Syntax unterschiedlich. TS-Einzelsätze sind in manchen Umgebungen (ESA) nicht unterstützt. Soll mit den Befehlen der Einzelsatzlogik eine Queue erstellt und verarbeitet werden, so steht dafür der Eintrag 'S' zur Verfügung.

Spalte 17-18 Nicht unterstützt

Spalte 19 Satzformat

Mögliche Eintragungen sind:

'F' Feste Satzlänge.

'V' Variable Satzlänge (nur für Bildschirme und VSAM-Dateien).

Bei fehlender Eintragung wird bei Plattendateien 'F', bei Bildschirmdateien 'V' angenommen.

Spalte 20-23 Blocklänge

Platten-Datei:

In diesem Feld wird bei geblockten Sätzen die Blocklänge, bei ungeblockten Sätzen die Satzlänge eingetragen.

Bei einer VSAM-Datei wird die doppelte Satzlänge bzw. Blank eingetragen (in der CICS-FCT Recform=Blocked).

Dataset:

Für Datasets bleibt dieses Feld blank.

Bildschirm:

In diesem Feld wird die Anzahl der Bildschirmzeilen eingetragen. (z.B. 12 , 24 , 27 , 32 , 43, 50 oder 62). Das Feld Satzlänge muss dabei jedoch die Länge einer Bildschirmzeile enthalten.

Spalte 24-27 Satzlänge

Bei der Platten-Datei m u s s die logische Satzlänge eingetragen werden.

Bei Dateien mit variabler Satzlänge muss die Länge eingetragen werden, die dem längsten Satz entspricht, der verarbeitet werden kann.

Bei Datasets m u s s die Länge des Datasets eingetragen werden.

Beim Bildschirm wird die Länge einer Bildschirmzeile eingetragen. Zum Beispiel '40', '80', '132'; für 3290 Bildschirme ist ebenfalls '106' und '160' erlaubt.
Der CPG-Übersetzer überprüft aufgrund dieser Längenangabe die erste und letzte Stelle in den Ein- und Ausgabebestimmungen für die Datei.

Spalte 28 Nicht unterstützt

Spalte 29-30 Länge des Schlüssels

ISAM, VSAM, DAM, Dataset.
Nur DISK Dateien

Fehlt die Länge des Schlüssels, so wird bei Batchprogrammen und HL1-Modulen die Datei als sequentielle Plattendatei verarbeitet.

Ansonsten muss die Länge des Schlüssels in Bytes eingetragen werden. Der Schlüssel kann ungepackt oder gepackt sein.
Für VSAM-ESDS- oder RRDS-Dateien ist 4 Bytes anzugeben.

Spalte 31 Satzadressierung - Dateiart

Für ISAM- und VSAM-Dateien:

' '	KEY	ISAM und VSAM-KSDS und RRDS
'K'	KEY	ISAM und VSAM-KSDS und RRDS
'R'	RBA	VSAM-ESDS

Bei fehlender Eintragung wird 'K' angenommen.

Spalte 32 Dateiorganisation

Plattendateien:

' '	ISAM-, VSAM-, DA oder sequentielle Batchdateien
'I'	ISAM-, VSAM- oder DA-Datei
'V'	VSAM-Datei
'L'	VSAM-Datei Locate Mode
'R'	REUSE, VSAM-Datei wird neu geladen (nur Batch) bei Ausgabe-Dateien.

Temporary Storage:

' '	Die Daten können nur vom eigenen Terminal gelesen werden und werden im Hauptspeicher gespeichert.

'I' Die Daten können von allen Terminals gelesen werden und werden im Hauptspeicher gespeichert.

'A' Die Daten können nur vom eigenen Terminal gelesen und im Hilfsspeicher gespeichert werden.

Spalte 33-38 Keine Eintragung erforderlich

Die Daten werden den Tabellen des TP-Steuerprogramms entnommen.

Spalte 39 Verarbeitungsform

'V' variabler Dateiname

Ein 'V' kennzeichnet einen variablen Dateinamen für Drucker, Transient Data oder Temporary Storage.

Ist ein 'V' eingetragen, so kann der Dateiname während der Programmdurchführung geändert werden. Der Standardwert wird den Spalten 7 bis 10 bzw. 7 bis 14 (der F-Karte) entnommen.

Die CPG-internen Felder werden während der Ausführung mit den gewünschten Dateinamen gefüllt.

CPGDID	4 Stellen	für	Einheit PRINTER
CPGTDI	4 Stellen	für	Einheit TRANSDT
CPGTSN	8 Stellen	für	Einheit STORAGE

Es folgt ein Anwendungsbeispiel für eine Druckausgabe.

Soll ein Satz mit einem anderen Namen an eine Druckerdatei ausgegeben werden, muss im Feld CPGDID der neue 4 Zeichen lange Dateiname abgesetzt werden.

Zum Beispiel kann ein Satz an die Druckerdatei PRT1 ausgegeben werden, indem das 4 Zeichen Literal 'PRT1' mit der MOVE-Operation in das Feld CPGDID übertragen wird.

Der Dateiname kann so oft wie erforderlich geändert werden.

'C' Farbbildschirm

Ein 'C' in dieser Spalte bedeutet für eine Bildschirmdatei, dass die Funktionen für 7- farbige Bildschirme von CPG unterstützt werden. Siehe Abschnitt 4910.

Spalte 40-46 Ein/Ausgabe-Einheit

Mögliche Eintragungen sind:

DISPLAY Bildschirm

PRINTER Drucker

PUNCHER Stanzer (nur im Batch)

READER Leser (nur im Batch)

DISK Platteneinheit (für alle Plattentypen) (für alle Dateiorganisationen)

KSDS VSAM-KSDS-Datei

VBOMP VBOMP-Datenbank

RRDS VSAM-RRDS-Datei

DL1 DL/1-Datenbank

ESDS VSAM-ESDS-Datei

DATASET Datensatz (Siehe Abschnitt 1 : Datenbankunabhängige Programmierung).

HL1 HL1-Dataset (für HL1-Benutzer).

HL1DS HL1-Dataset (für HL1-Benutzer), bei dem die PWA erhalten bleibt.

LU62 Distributed Data Processing

TABLE Tabelle

TAPE Banddatei (nur Batch)

TRANSDT Übergangsspeicher (Transient Data)

STORAGE Zwischenspeicher (Temporary Storage)

Bildschirm und Drucker können sowohl für Lokal- als auch für Remoteanschluss verwendet werden.

Beim HL1-Batch alle Einheiten zugelassen außer:

DISPLAY, VBOMP, DATASET und TRANSDT.

Spalte 45-46 Satzart

Es kann eine Satzart eingetragen werden, die einer Data-Dictionary-Struktur entspricht.

Spalte 47-74 Keine Eintragung erforderlich

Spalte 47-52 SYSNR bei Bändern (Nur für HL1-Batch-Benutzer).

Spalte 53 'S' (Standard-Kennsatz, nur für HL1-Batch-Benutzer)

'S' muss eingetragen werden, wenn sequentielle Banddateien mit Kennsatz verarbeitet werden. Eine TLBL-Karte ist erforderlich.

Ist kein 'S' eingetragen, entfaellt die Prüfung der TLBL-Parameter durch das System.

Spalte 66 'U' (Unsortiert hinzufügen, nur für Batch-Benutzer)

Eine VSAM-KSDS Datei kann somit unsortiert geladen wer-
Den, d.h. unsortiertes Hinzufügen wird ermöglicht.

Es ist zu beachten, dass diese Verarbeitungsart die Performance in hohem Maße beeinträchtigen kann. Größere Datenmengen werden sinnvollerweise sortiert in eine VSAM-KSDS-Datei eingespielt.

Spalte 70 Band zurückspulen (Nur für HL1-Batch-Benutzer)

Wird hier keine Eintragung vorgenommen, dann wird bei der Band-Verarbeitung das Band zu Beginn und zum Ende des Programms zurückgespult.

'N' für No Rewind bewirkt, dass das Band jeweils nicht zurückgespult wird.

'U' für Unload. Das Band wird beim Programmanfang zurückgespult und am Programmende aus der Bandstation entladen.

Spalte 71-72 'NO' (Datei nicht eröffnen, mit Ausnahme von Drucker, Leser und Stanzer). (Nur für HL1-Batch Benutzer)

Die Angabe 'NO' gilt nicht für die Einheiten DL1, TABLE, STORAGE, READER, sowie PRINTER und PUNCHER, wenn keine SYSNR angegeben wurde.

Dateien, die wegen der Eintragung 'NO' nicht eröffnet sind, müssen zur Verarbeitung mit dem CPG-Befehl OPEN explizit eröffnet werden.

Spalte 75-80 Programmname

Bei fehlender Eintragung wird der Phasenname aus der H-Karte eingesetzt.

Beispiele:

```
FDATEI                                                  Standard FCT
FDATEI                                                  CPG.DD
FDATEI    I                                             CPG.DD mit Eingabe
FISAM     U   F 200 100  5 I       DISK                 ISAM-Datei
FVSAM     U   F     100  5KV       DISK                 VSAM-KSDS Datei
FSAM      I   F2000 100            DISK                 Seq. Batchdatei
FKSDS     U   F     100  5KV       KSDS                 VSAM-KSDS Datei
FRRDS     U   F     100  4         RRDS                 VSAM-RRDS Datei
FESDS     U   F     100  4R        ESDS                 VSAM-ESDS Datei
FRELHEX   U   F     100  3         DISK                 DA(Rel.Spur+Satz)
FRELDEC   U   F     100  8         DISK                 DA(REL.Spur+Satz)
FPHYSDAM  U   F     100  8         DISK                 DA Phys. Address
FDATASET  U   F     100  5         DATASET              Datenbank EXIT
FHL1DAT   U   F     100  4         HL1                  HL1-Dataset
FHL1DS    U   F     370 12         HL1DS                HL1-Dataset
FPSBNAME  U   F       7            DL1                  DL/1 Datenbank
FBILD     C   V  24  80            DISPLAY              Bildschirm
FBILDD    C                        DISPLAY              Bildschirm Default
FBILD7F   C                       CDISPLAY              Siebenf.Bildschirm
FDRUC     O   F     132             PRINTER             Drucker Line Mode
FDRUC     O   F     132            VPRINTER             Variabler Name
FDRUC     OB  F  24  80             PRINTER             Drucker Buffer m.
FDATA     O   F     100             STORAGE             Zwischenspeicher
FDATANAMEO    F     100            VSTORAGE             var. Zwischensp.
FSTOR     UQ  F     100             STORAGE             Temp. St. Queuing
FSTORNAMEUQ   F     100    A       VSTORAGE             var.TS Queue, Aux.
FDEST     O   F     100             TRANSDT             Transient Data
FDEST     O   F     100            VTRANSDT             var.Transient Data
FB3290A   C   V  62 160             DISPLAY             3290 Bildschirm
FB3290B   C   V  62 132             DISPLAY             3290 Bildschirm
FB3290C   C   V  50 106             DISPLAY             3290 Bildschirm
FREADER   I   F     80             READER               HL1-Batch Leser
FPUNCHER  O   F     80             PUNCHER              HL1-Batch Stanzer
FTAPO300  O   F 900 300            TAPE   SYS010S       HL1-Batch Band
FETAB     I   F     19  9          TABLE                Datenview
F$OGSTA$  I   F     100 14         DISK                 Dateiname mit $
FOG$STA   I   F     100 14         DISK                 Dateiname mit $
FLISTE2   O   F     132            PRINTER SYS011       alternative Ausg.
FLISTE3   O   F     150            PRINTER SYS011       z.B. " für Laser
```

Erweiterte Dateizuordnung 4300

Formular-Beschreibung

Spalte 6 CPG-Kartenart

'E' muss eingetragen werden

Programmverbindungen:

Spalte 27-32 Feldname

Spalte 40-42 Feldlänge

Spalte 44 Anzahl Dezimalstellen

Feldgruppen:

Eine Feldgruppe wird wie folgt definiert:

Spalte 27-32 Name der Feldgruppe

Spalte 37-39 Anzahl der Felder

Ein Eintrag von 1 ist nicht zulässig.

Spalte 40-42 Feldlänge

Spalte 44 Dezimalstellen

Spalte 75-80 Programmname

Bei fehlender Eintragung wird der Phasenname aus der H-Karte eingesetzt.

Es können max. 80 Feldgruppen definiert werden.

Überlagerung von Feldern in der TWA

Es können auch Überlagerungsfelder (Overlay-Felder) definiert werden. Ein Überlagerungsfeld ist ein Feld, das in weitere Felder unterteilt ist. Es wird durch Angabe einer Null in Spalte 39 definiert. Überlagerungsfelder können numerisch oder alphanumerisch sein. Auf jede Spezifikation eines Überlagerungsfeldes müssen Bestimmungen für die Felder folgen, in die der Bereich unterteilt werden soll. Diese werden als gewöhnliche Felder oder Feldgruppen bei CPG definiert.

Ein Überlagerungsfeld kann auch weitere Überlagerungs-

felder enthalten. Es ist darauf zu achten, dass genügend Felder definiert werden, um den gesamten für ein Überlagerungsfeld definierten Bereich zu füllen.

Beispiel:

```
ANSCHR      0 60
NAME          20
ORT           20
STR           20
```

Die drei Felder NAME, ORT, STR können dann auch unter dem gemeinsamen Namen 'ANSCHR' angesprochen werden, bzw. beim Einlesen des Feldes ANSCHR werden auch die Felder NAME, ORT und STR gefüllt.

Bei einer Überlagerung von einem Datensatz können keine ungepackten Felder definiert werden.

Eigene Copy-Books in der TWA.

Um die Kommunikation mit bestehenden Programmen zu ermöglichen, die nicht in CPG geschrieben sind, erhält der Benutzer die Möglichkeit, eine bereits bestehende Transaction Work Area in Form eines Copy-Books in die TWA des CPG-Programms zu übernehmen.

Spalte 11-18 Enthält dabei den Namen des in der SLB katalogisierten Copy-Buchs.

Im Copy-Book verwendete Feldnamen können nur aus einer Assembler-Routine angesprochen werden. Sie dürfen im CPG-Teil nicht verwendet werden.

Bei Programm-Verbindungen muss das Copy-Book in alle miteinander verbundenen Programme eingefügt werden.

Beispiel:

```
FBILD                              L3270
E    TWACOPY          ANSCHR     0 60
E                     NAME         20
E                     ORT          20
E                     STR          20
```

TWA-Copy-Books dürfen nicht in Batch-Programmen benutzt werden.

Datenbeschreibung 4400

Zur besseren Programmdokumentation kann die E-Karte durch eine D-Karte ersetzt werden. Die Datenbeschreibungs-Karte entspricht in ihrer Einteilung der E-Karte, wobei die Felder lediglich um 12 Stellen nach links verschoben sind. Dadurch bietet die rechte Hälfte der Karte Platz für eine ausführliche Beschreibung des Feldnamens. Werden beide Kartenarten verwendet, so sind die D-Karten hinter den E-Karten einzuordnen.

Formular-Beschreibung

Spalte 6 CPG-Kartenart

'D' muss eingetragen werden.

Spalte 7-14 Dateiname

Nur in Verbindung mit Data Dictionary. S. Spalte 21-22.

Spalte 15-20 Feldname oder Name der Feldgruppe

Spalte 15 und 16 kann außerdem eine Bezugszahl enthalten, deren Bedeutung im nachfolgenden Kommentar beschrieben wird.

Spalte 21-22 Data Dictionary

Die Eintragung 'DD' bewirkt, dass die Dateifelddefinitionen aus dem CPG2..Data Dictionary hierhinter eingefügt werden.

Außerdem ist ein FILLER in den D-Karten (Define DD) unterstützt. (Nur bei Dateien mit Directory Field Check).

Beispiel:

```
DEFINE DEMO
  F1              10    * FELD1
   CPGFIL         10    * CPGFILLER
  F2              20    * FELD2
  F3              15    * FELD3
   CPGFIL         65    * CPGFILLER
  F4              20    * FELD4
```

Spalte 21-23 'ORG' Überdefinition von Feldern (Beispiel s.u.)

Spalte 21-24 'ORGE' Überlagerungen

Um Redefinitionen noch flexibler gestalten zu können, kann jetzt auch das Ende einer Redefinition mit ORGE explizit angegeben werden.

Spalte 23-24 Data Dictionary Satzart

In diesen Spalten kann von der ausgewählten Datei eine Satzart aus dem Data-Dictionary eingetragen werden.

Spalte 25-27 Anzahl der Felder

Ein Eintrag von 1 ist nicht zulässig.

Spalte 28-30 Feldlänge

Spalte 32 Dezimalstellen

Bei numerischen Feldern wird eine Eintragung zwischen 0 und 9 vorgenommen, womit die Anzahl der Dezimalstellen definiert werden. Die Definition eines alphanumerischen Feldes auf Halb-, Voll- oder Doppelwortgrenze kann mit folgenden Eintragungen erreicht werden:

'D' ausrichten auf Doppelwortgrenze
'F' ausrichten auf Vollwortgrenze
'H' ausrichten auf Halbwortgrenze

Spalte 33 Externe Felder

Ein 'E' in dieser Spalte bewirkt, dass das definierte Feld nicht in die TWA des Programms übernommen wird. Der Programmierer kann damit auf externe Felder z. B. in der CSA zugreifen, wobei er jedoch sicherstellen muss, dass das angegebene Feld auch unter gleichem Namen und mit gleichen Eigenschaften außerhalb der TWA vorhanden ist.

Wenn das gleiche Feld in der D-Karte mehrfach definiert wird, kommt es zu der Fehlermeldung 'doppelt definiert'. Arbeitet man in der D-Karte mit Define Dictionary-Struktur, dann tritt dieser Fall auf, wenn ein Feld in mehreren Strukturen liegt. Durch ein 'M' in dieser Spalte ist es möglich, ein Feld mehrfach zu definieren.

Beachte:

Diese mehrfach definierten Felder werden intern auf Kommentar gesetzt, dürfen also nicht Teil einer Redefinition (Überlagerung) sein.

Spalte 40-74 Dokumentation

Spalte 75-80 Programmname

Bei fehlender Eintragung wird der Phasenname aus der H-Karte eingesetzt.

Es können maximal 80 Feldgruppen definiert werden.

Überlagerungen von Feldern in der TWA sind wie bei der E-Karte möglich. Eigene Copy-Books sind bei der Datenbeschreibungs-Karte nicht unterstützt.

Beispiel:

```
D        LFDBSL       1 0   Lieferbedingungs-Schlüssel
D        LFDBTX      11     Lieferbedingungen in Klartext
```

```
        D        20                 Ungültiger Schlüssel
        D        AFELD        5     Alphafeld 5 Stellen
        D        NFELD        3 1   Num. Feld 3 Stellen davon 1 Dez.
        D        FELDG     5  3     Feldgruppe 5 * 3 Stellen
        DFNAME   SATZ DD01 0100     DD Eintr. einfügen mit Feld def.
        DFNAME        DD02          DD Eintr. einfügen ohne Feld def.
        D        AFELDF       4 F   Alpha Feld 4 Stellen auf Vollwort-
                                    grenze ausgerichtet.
```

Beispiel einer doppelten Überlagerung:

```
        D        AFELD1    0 10     Byte  1 - 10
        D        AFELD2       4     Byte  1 -  4

        D        AFELD3    0  6     Byte  5 - 10
        D        AFELD4       2     Byte  5 -  6
        D        AFELD5       4     Byte  7 - 10
```

Beispiel einer ORG-Anwendung:

```
        D        FELD1       10     Byte  1 - 10
        D        FELD2       10     Byte 11 - 20
        D        FELD1 ORG          Auf Byte  1 positionieren
        D        FELD3        5     Byte  1 -  5
        D        FELD4       10     Byte  6 - 15
        D              ORG          Auf Byte 21 positionieren
        D        FELD5        3     Byte 21 - 23
```

Druckersteuerung 4500

Die einfachste Art, eine Druckausgabe zu erstellen, bietet der Befehl LIST, wenn QTF installiert ist. Wird LIST nicht verwendet, so kann der Drucker auch explizit beschrieben werden.

Drucker 3286 können auf zwei Arten programmiert werden, einmal wie ein Bildschirm im 'Buffermode', d. h. mit einer Ausgabezeilenbestimmung wird der gesamte Puffer gefüllt. Die Zeilentransporte und Vorschübe ergeben sich aus der Positionsangabe in den Spalten 40 - 41 der Ausgabebestimmungen. Zu beachten ist hierbei, dass Leerzeilen nur dann gedruckt werden, wenn mindestens 1 Blank für diese Zeilen angegeben wurde. Im Buffermode sind nur die Zeilen von 1 - 24 und die Spalten von 1 - 80 unterstützt. Soll zum Beispiel auf ein 36-zeiliges Formular gedruckt werden, so ist eine zweite Ausgabe erforderlich, in der zum Beispiel von Zeile 1-12 jeweils ein Leerzeichen ausgegeben wird.

Für diese Programmierart enthält die Spalte 16 der zugehörigen Dateizuordnung ein 'B'.

Fehlt diese Eintragung, so wird 'Linemode' angenommen. In diesem Fall wird der Drucker wie jeder Systemdrucker programmiert. Je Ausgabezeilenbestimmung wird jeweils eine Zeile gedruckt. Die Zeilentransporte und Vorschübe erfolgen gemäß den Eintragungen in den Spalten 17 bis 22 der Ausgabezeilenbestimmungen.

In einem Programm darf nur ein 'Line Mode'-Drucker definiert werden.

Voraussetzung für diese Programmierart ist die folgende Druckzeilen-Bestimmung.

Im Batch können beliebig viele zusätzliche Drucker definiert werden. Hierzu ist in der File-Karte die SYSNR anzugeben. Bei diesen Druckern erfolgt die Vorschubsteuerung nicht über die L-Karte, sondern über den FCB des Druckers.

Im Batch ist nur die Ausgabe im Linemode unterstützt.

QTF-Benutzer können eine Druckausgabe am einfachsten mit dem Befehl LIST realisieren. F-, L- und O-Statements für den Drucker sind dabei nicht erforderlich.

Formular-Beschreibung

Spalte 6 CPG-Kartenart

'L' muss eingetragen werden.

Spalte 7-14 Dateiname

Name des Druckers (wie Spalte 7-14 der zugehörigen Dateizuordnung).

Spalte 15-17 Formularlänge

Länge des Formulars in Anzahl Zeilen.

Spalte 18-19 Kennzeichen für Formularlänge

Diese Spalten müssen die Konstante 'FL' enthalten.

Spalte 20-22 Druckzeile
25-27
30-32
35-37
40-42
45-47
50-52
55-57
60-62
65-67
70-72

In diesen Spalten wird die Druckzeile eingetragen, die dem Kanal in den beiden Folgespalten entspricht. Diese Eintragung darf nicht größer als die Formularlänge sein.

Spalte 23-24 Lochbandkanal
28-29
33-34
38-39
43-44
48-49
53-54
58-59
63-64
68-69 In diesen Spalten wird ein Kanal zwischen '01' und '12'
73-74 eingetragen.

Spalte 75-80 Programmname

Bei fehlender Eintragung wird der Phasenname aus der H-Karte eingesetzt.

Enthält die Spalte 39 der Drucker-Dateizuordnung ein 'V', so wird der Name des Druckers dem CPG-internen vierstelligen Alphafeld CPGDID entnommen. Der Programmierer muss dann sicherstellen, dass dieses Feld zur Ausführungszeit einen gültigen Drucker-Namen (Destination-Id) enthält. Bei Line Mode-Druckern wird der in der L-Karte angegebene Name als Defaultwert eingesetzt.

Wird ein Drucker im Linemode definiert und die zugehörige L-Karte nicht codiert, so erscheint keine Fehlermeldung. Es wird als Default eine Formularlänge von 72 Zeilen angenommen, Zeile 3 wird Kanal 01 zugeordnet, Kanal 12 wird mit Zeile 68 belegt.

Bei Batchprogrammen kennzeichnet Kanal 12 die Zeile, ab der der Overflow-Schalter OF gesetzt wird.

Eingabebestimmung 4600

Formular-Beschreibung

Spalte 6 CPG-Kartenart

Satzbestimmung: 'I' muss eingetragen werden.

Spalte 7-14 Dateiname

Name der in der Dateizuordnung definierten Datei.
Der Name darf nur einmal in den Eingabebestimmungen vorhanden sein.
Für SELCT-Operationen wird der Name des entsprechenden Feldes eingetragen.

Spalte 15-16 Reihenfolge

Eine beliebige gültige Eintragung ungleich Blank kennzeichnet diese Eingabe-Bestimmung als Satzbestimmung.

Die Eintragung 'OR' in den Spalten 14,15 bewirkt eine Oder-Verknüpfung der Eingabebestimmungen.

Die Eintragung 'AND' in den Spalten 14-16 bewirkt eine Und-Verknüpfung der Eingabebestimmungen.

Die Eintragungen 'OR' und 'AND' gelten nicht für Bildschirmdateien und Feldaufbereitung.

Bei Feldaufbereitung (SELCT) müssen diese Spalten die Konstante 'F ' enthalten.

Bei Eintragung der Konstanten 'SF' (Standard File Table) kann die Dateizuordnungskarte entfallen, wenn die Datei in der Standard File Table enthalten ist.

Bei HL1-Programmen muss 'HS' eingetragen werden, wenn ein Datenkanal definiert wird.

Zur Eintragung 'DD ' siehe Spalte 21-22.

Die Eintragung 'DI' bewirkt das gleiche wie 'DD', jedoch werden die Feldpositionen dem aktuellen Data Dictionary entnommen. Das Programm wird also unabhängig von Verschiebungen von Feldern innerhalb einer Struktur. Das heisst, eine Umwandlung ist nicht erforderlich, wenn Felder innerhalb der Struktur verschoben werden. (Diese Eintragung **ist unter ESA und im Batch nicht unterstützt**)

Achtung: 'DI' sollte wegen ESA auf 'DD' umgestellt werden.

Bei der Eintragung 'FS' siehe CPG2 Segment.

Spalte 17-18 Satzart

Siehe Spalte 23-24.

Spalte 19-42 Für Bildschirmdateien nicht unterstützt.

Spalte 19-20 Satzbezugszahl (bei Platte und Band), Datenstruktur.

Diese Bezugszahl wird gesetzt, wenn ein Satz gelesen wird, der den Bedingungen in den Spalten 21 bis 41 entspricht.

Die Bezugszahl bleibt gesetzt bis der nächste Satz gelesen wird, wenn sie nicht vorher vom Programmierer gelöscht wird.

'DS' Bei einer Datenstruktur muss DS eingetragen werden.

Spalte 21-22 Data Dictionary (neue Position der Eintragung, die alte bleibt unterstützt)

Die Eintragung 'DD ' bewirkt, dass die Dateifeldbestimmungen vom Data-Dictionary hierhinter in alphabetischer Reihenfolge oder nach Linienbelegung aufsteigend eingefügt werden.

Die Eintragung 'DE' für 'DD' plus Feld definieren wurde unterstützt. Es können Felder für Dateiverarbeitung mit WRITE bzw. UPDATE definiert werden.

Spalte 23-24 Satzart (neue Position der Eintragung, die alte bleibt unterstützt)

In diesen Spalten kann von der ausgewählten Datei eine Satzart aus dem Data-Dictionary eingetragen werden.

Spalte 21-24 Stelle
28-31
35-38

Hier wird rechtsbündig die Position der zu prüfenden Stelle im Satz eingetragen, z.B. 1, wenn das 1. Byte des Plattensatzes geprüft werden soll.

Werden verschiedene Satzarten eingelesen, so muss in Spalte 19-20 eine Bezugszahl eingetragen werden.

Spalte 25 Negation
32
39

Ein 'N' in diesen Spalten setzt die Bezugszahl in 19-20, wenn die Bedingung n i c h t erfüllt ist.

Spalte 26 Prüfung
33
40

Ein 'C' in dieser Spalte bewirkt, dass das gesamte Byte mit dem Zeichen in Spalte 27 (34,41) verglichen wird. Die Eintragungen 'D' und 'Z' werden nicht unterstützt.

Spalte 27 Zeichen
34
41

In diesen Spalten wird das zu prüfende Zeichen (z.B. Satzart) eingetragen.

Spalte 32-37 Feldtype bei Selektion

Es können zu einem Feld mehrere Typen angegeben werden. Dadurch kann bei der SELCT-Anweisung gesteuert werden, welche Belegung des Feldes eingelesen wird. Alle zu einem Feld gehörigen Typen müssen direkt nach dem Feld stehen.

Spalte 42 Ein 'T' in Verbindung mit Data Dictionary erzeugt zum jeweiligen Eingabe-Statement eine Textzeile, in der die Feldbezeichnungen angelistet werden.

Ein 'A' in dieser Spalte bewirkt, dass die Textzeile bei Data Dictionary hinter der I-Felddefinition aufgelistet wird. Wird ein nicht benutztes Feld nicht aufgelistet, so wird die dazugehörende Textzeile auch unterdrückt. Für die Eingabedatei wird mit der Dateibeschreibung auch eine I*-Zeile generiert.

Feldbestimmung: O (Overflow)

Dieser Eintrag kann nur bei einer Bildschirmdatei eingetragen werden. Wenn die Eintragungen Spalte 44-45 mit der Spalte 48-49 nicht gleich sind, wird eine Fehlermeldung gedruckt. Die Eintragung 'O' unterdrückt diese Fehlermeldung. Diese Möglichkeit wird benutzt, um ein Feld über mehrere Zeilen einzulesen. Bei 132-stelligen Bildschirmen muss ein Overflow-Feld in den E-oder D-Karten definiert werden.

Das Feld, welches über mehrere Zeilen gelesen wird, sollte schon vorher definiert worden sein.

Spalte 43 Gepackte Eingabe

Ein 'P' in dieser Spalte bedeutet, dass numerische Daten gelesen werden sollen.

Ein 'O' in dieser Spalte bedeutet, dass numerische Daten mit gerader Stellenzahl eingelesen werden können.

Ein 'B' in dieser Spalte bedeutet, dass numerische Daten in binärer Form eingelesen werden. Das Feld muss mit 2 oder 4 Bytes definiert werden.

Ein 'L' in dieser Spalte bedeutet, dass numerische Daten in logisch gepackter Form gelesen werden sollen. Siehe logisch gepackte Felder, Abschnitt 2144.

Gepackte oder binäre Eingabe von Bildschirm-Dateien ist nicht möglich.

'V' Verschiebung bei Satzbestimmung

In Verbindung mit dem Feld CPGFIS kann die Input-Area

verschoben werden. Hiermit besteht die Möglichkeit, Dateien mit Satzlängen größer 8000 Bytes zu verarbeiten. Diese Verarbeitungsart ist nur bei Datei-Eingaben möglich.

Spalte 44-47 Von Stelle

Platten-Datei:
Erste Stelle des Feldes im Plattensatz.

Bildschirm:
In Spalte 44 bis 45 wird die Bildschirmzeile eingetragen. In Spalte 46 bis 47 die erste Stelle des Datenfeldes (ausschließlich Attributbyte) in der Zeile.

Wenn die Bildschirmzeile größer als 99 Zeichen ist, wird in der Zehner-Stelle hexadezimal weiter gezählt (z.B. A0 für 100, B2 für 112, C0 für 120).

Spalte 48-51 Bis Stelle

Wie Spalte 44-47, jedoch letzte Stelle des Feldes.

Bei einer Datenstruktur kann hier die Länge der Datenstruktur rechtsbündig eingetragen werden.

Spalte 52 Davon Dezimalstellen

Bei Alpha-Feldern bleibt dieses Feld blank. Die Eintragung einer Ziffer von 0 bis 9 definiert dieses Feld als numerisches Feld, wobei die Ziffer die Anzahl der Dezimalstellen angibt. Die Zahl darf nicht größer als die Feldlänge des Feldes sein.

Spalte 53-58 Feldname

Regeln siehe Abschnitt 1, Feldnamen.

In diesen Spalten kann der Name eines Feldes oder einer Feldgruppe stehen.

Bildschirm:

Bei einer Feldgruppe gelten die angegebenen Positionen für das erste Feld der Gruppe. Die weiteren Felder werden aus den gleichen Positionen der folgenden Zeilen gelesen.

Spalte 53-60 Referenzdatei für DD (Data Dictionary)

Spalte 59-60 Gruppenstufen

Die Eintragung L0 bis L9 bewirkt, dass der Inhalt dieses Feldes auf Gruppenwechsel geprüft wird und bei erfolgtem Gruppenwechsel der entsprechende Gruppenschalter gesetzt und vor dem Einlesen der Daten die Total-Rechenbestimmungen durchlaufen werden.

Für Bildschirmdateien, Transient Data und Temporary Storage Queuing werden Gruppenstufen nicht unterstützt.

Spalte 61-62 Bezugszahl für Lichtstift-Auswahl

Nur für Bildschirmdateien. Eine Bezugszahl, die in diese Spalten eingetragen wird, wird dann gesetzt, wenn das zugehörige Feld mit dem Lichtstift angetippt wird. Das Feld muss jedoch mit dem Aufbereitungsschlüssel 'L' oder 'I' oder entsprechend (siehe 4903) auf den Bildschirm gebracht worden sein und in der ersten Stelle ein Blank enthalten.

Achtung: Diese Bezugszahl darf nur verwendet werden, wenn der Bildschirm keine Felder enthält, deren Aufbereitungsschlüssel die Feldeigenschaft 'Modified Data Tag gesetzt' enthält (siehe Ausgabe,Spalte 38).

Wird diese Bezugszahl bei einer Feldgruppe verwendet, so werden soviele Bezugszahlen belegt, wie die Feldgruppe Elemente hat.

Beispiel:

Eine Feldgruppe 'PAGE' wurde definiert mit 10 Elementen. Spalte 53-58 enthält den Namen PAGE, Spalte 61 - 62 enthält die Bezugszahl 31.

Die Bezugszahlen 31 bis 40 werden der Feldgruppe so zugeordnet, dass die Bezugszahl 31 dann gesetzt wird, wenn das erste Feld mit dem Lichtstift angetippt wurde, die Bezugszahl 36 wenn das 6. und die Bezugszahl 39 wenn das 9. Feld angetippt wurde.

Die gesetzten Bezugszahlen werden bei der nächsten Eingabeoperation für diese Datei vom CPG wieder gelöscht.

Die Lichtstiftbezugszahl kann auch gesetzt werden, um zu prüfen, ob ein Feld per Tastatur verändert wurde.

Wird ein Feld per Tastatur überschrieben, wird die Bezugszahl auf ein gesetzt. Diese Abfragemöglichkeit besteht nur, wenn eine Programmfunktionstaste PF1-PF24 oder Datenfreigabe betätigt wurde.

Spalte 63-64 Nicht unterstützt

Spalte 65-66 Setzen Bezugszahl größer als...

Für Alphafelder wird die hier eingetragene Bezugszahl gesetzt, wenn der Inhalt des Feldes größer als blank ist.

Für numerische Felder wird die hier eingetragene Bezugszahl gesetzt, wenn der Inhalt des Feldes größer als 0 ist. Feldgruppen sind nicht unterstützt.

Spalte 67-68 Setzen Bezugszahl kleiner als...

Für Alphafelder wird die hier eingetragene Bezugszahl gesetzt, wenn der Inhalt des Feldes kleiner als blank ist.

Für numerische Felder wird die hier eingetragene Be-

zugszahl gesetzt, wenn der Inhalt des Feldes kleiner als 0 ist. Feldgruppen sind nicht unterstützt.

Spalte 69-70 Setzen Bezugszahl gleich...

Für Alphafelder wird die hier eingetragene Bezugszahl gesetzt, wenn der Inhalt des Feldes gleich blank ist.

Für numerische Felder wird die hier eingetragene Bezugszahl gesetzt, wenn der Inhalt des Feldes gleich null ist. Feldgruppen sind nicht unterstützt.
Die Spalten 65 - 70 sind auch für Feldgruppenelemente unterstützt.

Spalte 71-74 Nicht unterstützt

Spalte 75-80 Programmname

Bei fehlender Eintragung wird der Phasenname aus der H-Karte eingesetzt.

Rechenbestimmung 4700

Formular-Beschreibung

Spalte 6 CPG-Kartenart

'C' muss eingetragen werden

Spalte 7 - 8 Gruppenstufe, Unterprogramm

Die Eintragung 'L' in Spalte 7 kennzeichnet die Rechenbestimmung als Gruppenberechnung. Eine Ziffer 0 bis 9 in Spalte 8 gibt dabei die jeweilige Gruppenstufe an.

Gruppenberechnungen müssen nach Gruppenstufen aufwärts sortiert jeweils am Ende der Rechenbestimmungen liegen.

Bei CPG-Unterprogrammen (BEGSR - ENDSR) kann in diesen Spalten die Konstante 'SR' eingetragen werden.

Die Eintragung 'AN' in diesen Spalten bedeutet, dass die in diesem Statement angegebenen Bezugszahlen mit denen des vorherigen Statements und-verknüpft werden.

Die Eintragung 'OR' in diesen Spalten bedeutet, dass die in diesem Statement angegebenen Bezugszahlen mit denen des vorherigen Statements oder-verknüpft werden.

In beiden Fällen, die auch gemischt verwendet werden können, steht die Operation im letzten Statement.

```
Beispiel:    10 11 12
          AN 13
          OR 14 15          SETON           20
```

Die Bezugszahl 20 wird gesetzt, wenn die Indicator 10 und 11 und 12 und 13 oder 14 und 15 gesetzt sind.

Unterschiedlich gegenüber RPG werden bei einer OR-/AN-Verknüpfung die Bezugszahlen der AN-Anweisung immer geprüft.

```
Beispiel:    10
          OR 11
          AN 12             SETON           20
```

Die Bezugszahl 20 wird gesetzt, wenn die Indicator 10 und 12 oder 11 und 12 gesetzt sind.

Bei der direkten Codierung von SQL-Statements (ab dem Service-Level CPG3) wird 'SQ' eingetragen.

Spalte 9,12,15 Negation der nachstehenden Bezugszahl

Ein 'N' in einer dieser Spalten besagt, dass die Operation durchgeführt wird, wenn die nachfolgende Bezugszahl nicht ein ist.

Spalte 10-11 Bedingung
13-14
16-17 Die Operation in Spalte 28-32 dieser Zeile wird durchgeführt, wenn die Bedingungen (Bezugszahlen ein) von Spalte 10 bis 17 erfüllt sind. Sind mehrere Bedingungen angegeben,so müssen alle erfüllt sein. (Und-Verknüpfung)

Spalte 18-27 Faktor 1

Eintragung erfolgt je nach Art der Operation (siehe auch Spalte 28-32).

Spalte 28-32 Operationscode

Siehe Operationen (3000).

Spalte 33-42 Faktor 2

Eintragung erfolgt je nach Art der Operation (siehe auch Beschreibung der einzelnen Operationen.

Spalte 43-48 Ergebnisfeld

Eintragung erfolgt je nach Art der Operation (siehe auch Operationen).

Spalte 49-51 Feldlänge

Stellenzahl des Ergebnisfeldes. Alphafelder dürfen maximal 256 Stellen lang sein, numerische Felder höchstens 15 Stellen. Eine Eintragung in diesen Spalten ist nur erforderlich, wenn ein Feld mit gleichem Namen vorher noch nicht definiert wurde. Ein Feld mit gleichem Namen (im Ergebnisfeld oder in den Eingabebestimmungen) darf nicht mit unterschiedlicher Stellenzahl definiert werden.

Spalte 52 Dezimalstellen

Ein Blank in dieser Spalte definiert das Feld als Alphafeld, eine Ziffer von 0 bis 9 als numerisches Feld, wobei die Ziffer die Anzahl der Dezimalstellen angibt.

Spalte 53 Runden (CPG-Service-Funktionen)

' ' Ein Blank in dieser Spalte bewirkt bei einer MOVE-Operation numerisch nach alpha, dass das Vorzeichen F eingesetzt wird, bzw. dass der Eintrag aus der Standard-H-Karte durchgeführt wird.

'A' Ein 'A' bei MAP-Operationen oder einem (E)READ für eine Bildschirmdatei kombiniert die Einträge 'T' und 'L'.

Ein 'A' beim MAPP bewirkt nach Ausgabe der Map einen

Vorschub auf den Blattanfang.

Ein 'A' bei MOVEV bewirkt, dass die Operation wie MOVEA arbeitet.

Ein 'A' bei BREAK bewirkt, dass bei verschachtelten DO-Schleifen die Verarbeitung bis zur äußersten Ebene beendet wird.

'B' Ein 'B' führt bei der SORTA-Operation zu aufsteigender Sortierfolge mit der Besonderheit, dass 'Blank'- bzw. 'Null'-Elemente hinten angeführt werden.

Ein 'B' beim MAPP bewirkt vor Ausgabe der Map einen Vorschub auf den Blattanfang.

'C' Ein 'C' in dieser Spalte bewirkt bei einer CHAIN-Operation, dass der Satz nicht eingelesen wird. Es wird nur geprüft, ob der Schlüssel des Satzes in der Datei vorhanden ist. Die Eingabebestimmungen werden hierbei nicht durchgeführt.

Ein 'C' in dieser Spalte bewirkt bei einer MOVE-Operation numerisch nach alpha, dass das Vorzeichen des numerischen Feldes beibehalten wird.

Ein 'C' bei MAP-Operationen oder einem (E)READ für eine Bildschirmdatei kombiniert die Einträge 'T' und 'I'.

Ein 'C' bei CONVT bewirkt, dass ein Feld vom Hexformat in das Character-Format konvertiert wird.

'D' Ein 'D' beim SORTA sortiert in absteigender Reihenfolge.

'E' Ein 'E' bewirkt bei den Operationen ENQ und DEQ, dass externe Adressen gesperrt werden können.

'F' Ein 'F' in dieser Spalte erfüllt bei einer MOVE-Operation numerisch nach alpha, dass das Vorzeichen F eingesetzt wird.

'H' Ein 'H' in dieser Spalte bewirkt bei Rechen-Operationen, dass das Ergebnis in der letzten Stelle gerundet wird.

Ein 'H' beim MOVEN bewirkt eine erweiterte Prüfung ungültiger Zeichen.

'I' Ein 'I' in dieser Spalte bewirkt bei der READ-Operation für ein Terminal oder einer MAP-Operation, dass die Betätigung der Clear-Taste nicht zum Programmabschluss führt, sondern dass das Programm normal weiter arbeitet und der Programmierer anschließend den Schalter CL abfragen kann.

Bei einer indizierten Feldgruppenoperation wird keine Indexprüfung durchgeführt. Dies gilt für MOVE, MOVEL, EDIT und SELCT.

Ein 'I' unterbindet auch die Indexprüfung bei den Bezugszahlen-Operationen SETIX, SETIN und CLRIN.

Ein 'I' rettet die Bezugszahlen bei EXPR des rufenden Programms, wenn aus einer unteren Stufe zurückgekehrt wird. Sonst hat 'I' die gleiche Arbeitsweise wie 'S'.

'K' Ein 'K' bei MAP-Operationen oder einem (E)READ für eine Bildschirmdatei kombiniert die Einträge 'T', 'L', 'I'.

'L' Ein 'L' bewirkt bei einer MAP-Operation oder einem READ für eine Bildschirm-Datei, dass Kleinbuchstaben beim Einlesen nicht in Großbuchstaben übersetzt werden.

Ein 'L' in dieser Spalte bewirkt bei einer TESTN-Operation, dass auch das letzte Byte auf Ziffer geprüft wird (Zone 'F').

Ein 'L' beim unbedingten DO bewirkt eine Endlosschleife.

Ein 'L' bei MOVEV bewirkt, dass die Operation wie MOVEL arbeitet.

'N' Ein 'N' in dieser Spalte bewirkt bei einer EXITI-Operation, dass das laufende Programm durch die EXITI-Operation nicht verlassen wird, sondern dass das aufzurufende Programm erst dann aufgerufen wird, wenn das laufende Programm beendet ist.

'P' Ein 'P' in dieser Spalte bewirkt bei einer CHAIN-Operation die Funktion der Einträge 'C' + 'U'.

'R' Ein 'R' in dieser Spalte bewirkt bei einer SYNCP-Operation, dass ein irrtümlich gegebener SYNCP-Befehl wieder aufgehoben werden kann.

Ein 'R' bei MOVEV bewirkt, dass die Operation wie MOVE arbeitet (rechtsbündig).

'S' Ein 'S' sichert bei EXPR und EXITP-Programmen die TWA auf Temporary Storage.

Ein 'S' in dieser Spalte bewirkt bei der READ-Operation für Temporary Storage, dass der gelesene Satz nicht gelöscht wird, sondern für weitere READ-Operationen erhalten bleibt. Dieser Eintrag ist für Temporary Storage Queuing nicht erforderlich.

Ein 'S' bei einem READ vom Bildschirm hat die gleiche Funktion wie 'I' und 'L'.

Ein 'S' in dieser Spalte schließt 'T' und 'N' bei der EXITI-Operation ein.

Ein 'S' bei CONVT bewirkt, dass ein Feldinhalt in Kleinbuchstaben konvertiert wird.

Ein 'S' beim MAPP bewirkt vor und nach Ausgabe der Map einen Vorschub auf den Blattanfang.

'T' Ein 'T' in dieser Spalte bewirkt bei einer EXITI-Operation, dass der Inhalt des Ergebnisfeldes als Uhrzeit angenommen wird. Es ist dabei sicherzustellen, dass das Ergebnisfeld eine gültige Uhrzeit enthält.(auch EXITD)

Ein 'T' bei MAP-Operationen oder einem (E)READ für eine Bildschirmdatei steht für transaktionsorientierte Programmierweise.

'U' Ein 'U' in dieser Spalte bewirkt bei einer CHAIN-Operation, dass der Satz nicht mehr mit einer anderen CHAIN-Operation für ein UPDATE gelesen werden kann, bis in diesem Programm ein UPDATE erfolgt, der Satz durch ein RNDOM freigegeben wird oder das Programm beendet wird.

'V' Ein 'V' bei den Datei-Operationen CHECK, CLOSE und OPEN zeigt an, dass der Dateiname variabel verarbeitet wird, d.h. er steht in einem achtstelligen Alphafeld zur Verfügung.

Ein 'V' bei DLI-Operationen bewirkt, dass die einzelnen Einträge variabel gehandhabt werden und dem internen Feld CPGDLV entnommen werden.

Ein 'V' bei der Operation EXCPT bewirkt, dass der EXCPT-Name einem Feld entnommen wird, das in Faktor 2 der Operation angegeben ist.

'X' Ein 'X' bei der Operation CONVT bewirkt, dass ein Feldinhalt vom Character- ins Hexformat konvertiert wird.

'1' Eine '1' bewirkt bei einer DOU.. - Operation, dass die Schleife mindestens einmal durchlaufen wird.

Spalte 54-59 Ergebnisbezugszahlen

Diese Bezugszahlen können bei verschiedenen Operationen eingetragen werden (siehe Übersicht auf Seite 7000).

In Abhängigkeit vom jeweiligen Ergebnis werden diese Bezugszahlen dann gesetzt oder gelöscht.

Bei den Operationen EXCPT, EREAD, EDIT, IFC, EXSR bewirkt die Eintragung einer Bezugszahl in diesen Spalten, dass diese Bezugszahl vor der Operation gesetzt und nach der Operation wieder gelöscht wird.

Spalte 60-74 Kommentar

Eine Eintragung in diesem Feld wird in der CPG-Umwandlungsliste als Kommentar angelistet.

Spalte 75-80 Programmname

Bei fehlender Eintragung wird der Phasenname aus der H-Karte eingesetzt.

Ausgabebestimmung 4800

Formular-Beschreibung

Spalte 6 CPG-Kartenart

'O' muss eingetragen werden.

Spalte 7-14 Datei

Bei der ersten Satzbestimmung einer Datei m u s s hier der Name eingetragen werden. Die Datei muss in der Dateizuordnung als Ausgabe, Update oder Combined Datei definiert sein ('O','U' oder 'C' in Spalte 15 der Dateizuordnung). Für die EDIT-Operationen wird hier der Name der aufzubereitenden Felder eingetragen.

Spalte 14-15 Oder-Zeile

Die Eintragung 'OR' in dieser Spalte bewirkt eine Oder-Verknüpfung mit den vorher abgefragten Bezugszahlen.

Oder-Zeilen sind nicht erlaubt für Feldaufbereitung, Feldbestimmung sowie Konstanten.

Spalte 14-16 And-Zeile

Die Eintragung 'AND' in diesen Spalten bewirkt eine Und-Verknüpfung mit den vorher abgefragten Bezugszahlen.

AND-Zeilen sind nicht erlaubt für Feldaufbereitung, Feldbestimmung sowie Konstanten.

Spalte 15 Satzbestimmung

Ein 'E' in dieser Spalte wertet die Karte als Satzbestimmung. Für alle anderen Eintragungen ungleich blank außer 'F' oder 'C' wird vom CPG 'E' eingesetzt.

Ein 'F' in dieser Spalte bedeutet Feldaufbereitung. In Spalte 7 bis 14 muss in diesem Fall ein gültiger Feldname stehen. Siehe auch Feldaufbereitung.

Ein 'C' in dieser Spalte bedeutet für Bildschirm- oder Drucker-Dateien 'Diagramm-Ausgabe' (Chart). Siehe auch Diagramm-Ausgabe.

'D','T' und 'H' Zeilen in den Ausgabebestimmungen sind ungültig.

Spalte 16 Unformatisierte Ausgabe oder Bildschirmattribute

Ein 'U' in dieser Spalte gibt an, dass die Bildschirmausgabe unformatisiert erfolgen soll, das heisst es werden keine Command-Codes und keine Attribut-Bytes auf den

Bildschirm ausgegeben. Das Standard-Attribut ist ungeschützt, normal hell und modified. Die Positions-Angabe erfolgt relativ zur ersten Bildschirm-Position, das heisst eine Ausgabe in Zeile 2 Position 4 entspricht der Position 84 (80 + 4). Die Daten können bei der nächsten Transaktion mit einer Operation SELCT CPGTIO wieder gelesen werden.
Zu beachten ist, dass der Programmierer bei einer unformatierten Ausgabe ohne Erase (auf einen unformatierten Bildschirm) selbst für den Aufbau der Ausgabe verantwortlich ist. An den Stellen, die in den O-Karten nicht beschrieben sind, werden Blanks ausgegeben (die ein bestehendes Bild überschreiben).

Wenn in der H-Karte in Spalte 49 ein E eingetragen wird, so muss hier bei Feldbestimmungen das Bildschirm-Attribut-Byte eingetragen werden. Eine Aufstellung der Attribute befindet sich auf Seite 4900.

Bei einer Feldbestimmung für siebenfarbige Bildschirme können Eintragungen vorgenommen werden. Siehe Abschnitt 4910.

Spalte 16-18 Hinzufügen, Ändern oder Löschen von Sätzen

Die Konstante 'ADD' in diesen Spalten bedeutet, dass zu einer Datei ein neuer Satz hinzugefügt wird und ist nur zulässig, wenn Spalte 15 ein 'E' enthält.

Die Eintragung ADD ist nur für ISAM- oder VSAM-Dateien, Temporary Storage Queuing oder Datasets zulässig.

Die Eintragung 'ALG' in diesen Spalten bewirkt, dass beim Zurückschreiben von Datensätzen aus VSAM-Dateien mit variabler Satzlänge die Satzlänge geändert wird. Die höchste Ausgabeposition beim UPDATE wird als neue Satzlänge angenommen.

Die Konstante 'DEL' in diesen Spalten bedeutet, dass aus einer VSAM-Plattendatei ein Satz gelöscht wird.

Für andere Dateien gilt:

Spalte 17 Erase

Ein 'E' in dieser Spalte bewirkt, dass der Bildschirm vor der Ausgabe gelöscht wird.

Ein 'U' in dieser Spalte bewirkt bei der Bildschirmdatei, dass alle ungeschützten Felder vor der Ausgabe gelöscht werden.

Zeilentransport vor dem Drucken.

Bei einer Druckerdatei bewirkt eine Ziffer von 1 bis 9 einen entsprechenden Zeilentransport, wenn Spalte 16 der zugehörigen Dateizuordnung kein 'B' enthält.

Bei einer Feldbestimmung für siebenfarbige Bildschirme können Eintragungen vorgenommen werden. Siehe Abschnitt 4910.

Eintragungen bei Temporary Storage

Blank Die Daten können nur vom eigenen Terminal gelesen werden und werden im Hauptspeicher gespeichert.

I Die Daten können von allen Terminals gelesen werden und werden im Hauptspeicher gespeichert.

A Die Daten können nur vom eigenen Terminal gelesen werden und werden im Hilfsspeicher (Auxiliary) gespeichert.

Spalte 18 Zeilentransport nach dem Drucken

Eine Ziffer von 0 bis 9 muss eingetragen werden.
Bei einer Druckerdatei bewirkt eine Ziffer von 0 bis 9 einen entsprechenden Zeilentransport, wenn Spalte 16 der zugehörigen Dateizuordnung kein 'B' enthält.
Bei Bildschirmdateien kann hier ' ','H','K','L','M','N', 'O' oder 'S' eingetragen werden.

Diese Eintragung steuert:

. Dass die Hupe ertönt.
. Dass bei allen mit Modified Data Tag ausgegebenen Feldern die MDT-Bits gesetzt bleiben.
. Dass die Tastatur entriegelt wird.

Die Bedeutung der einzelnen Eintragungen siehe Abschnitt 7030.

Bei einer Feldbestimmung für siebenfarbige Bildschirme können Eintragungen vorgenommen werden. Siehe Abschnitt 4910.

Spalte 19-20 Vorschub vor dem Drucken/Data Dictionary

Bei einer Druckerdatei bewirkt eine Eintragung von 01-12 einen Vorschub bis zu der Zeile, die in der Druckzeilenbestimmung diesem Lochbandkanal zugeordnet wurde. Siehe auch Druckzeilenbestimmung.

Die Eintragung 'DD' bewirkt, dass die Dateifeldbestimmungen vom CPG2..Data-Dictionary hierhinter eingefügt werden. Siehe auch Abschnitt 2730.

Spalte 21-22 Satzart

In diesen Spalten kann von der ausgewählten Datei eine Satzart aus dem Data-Dictionary eingetragen werden.

Spalte 23-31 Bedingungen

Bei der Satzbestimmung wird ein Satz, bei der Feldbestimmung ein Feld, nur ausgegeben, wenn die Bedingungen in diesen Spalten erfüllt sind.

Bei einer Map-Feldausgabe sind diese Bedingungen nicht unterstützt.

Achtung: Für CPG-CICS-Subset-Programme gilt folgende Einschränkung. (H-Karte Spalte 47=C oder D).

Bei Verwendung von Feldbezugszahlen für eine Terminal-Datei (Bildschirm oder Drucker) muss der Programmierer folgendes beachten: Wird ein Feld unter Bedingungen ausgegeben, so wird bei Nicht-Ausgabe des Feldes das in den Ausgabebestimmungen voranliegende Feld um eine Blank-Konstante von der Länge des nicht ausgegebenen Feldes plus 8 Bytes verlängert. Dadurch kann möglicherweise ein Folgefeld durch Blanks überschrieben und damit gelöscht werden. Der Programmierer muss daher darauf achten, dass Feldbezugszahlen nur dort verwendet werden, wo Folgefelder nicht überschrieben werden können, z.B. am Ende der Ausgabebestimmungen, oder nach Feldern, welchen genügend Freiraum (z.B. eine Leerzeile) folgt.

Diese Einschränkung gilt nicht für Methodenbank-Programme, jedoch dürfen hierbei in Feldbestimmungen nur die Schalter 01 bis 99, die Programmfunktionsschalter P1 bis PC, T1 bis T9 und A1 bis A3 eingetragen werden.
Alle anderen Schalter sind in der Feldbestimmung nicht erlaubt.

Spalte 32-37 Ausgabefeld/EXCPT-Name

Name des Ausgabefeldes. Das Feld muss vorher in den Eingabe oder Rechenbestimmungen definiert sein.

Bildschirmdatei:

Bei einer Bildschirm-Datei kann auch der Name einer Ausgabe-Map eingetragen werden,wenn über CPG..TOP eine solche Map angelegt wurde.

Bei einer Satzbestimmung kann ein EXCPT-Name eingetragen werden. Es wird nur in den Fällen eine Ausgabe durchgeführt, wenn bei einer EXCPT-Operation der gleiche Name im Faktor2 eingetragen wurde.

Spalte 38 Feldattribut oder Aufbereitungsschlüssel-Verarbeitung variabler Satzlänge

In dieser Spalte wird bei einer Bildschirmdatei das jeweilige Feldattribut eingetragen.

Wenn in Spalte 49 der H-Karte ein E eingetragen wurde, so wird in dieser Spalte der Aufbereitungsschlüssel eingetragen. Das Attribut-Byte muss dann in Spalte 16 eingetragen werden.

Für eine Aufstellung der Attribut-Schlüssel siehe Seite 4900, für eine Aufstellung der Aufbereitungsschlüssel siehe Seite 4950.

Ein 'V' in dieser Spalte in Verbindung mit ADD/ALG bewirkt, dass der Wert aus CPGVRL bei VSAM-Dateien mit variabler Satzlänge die Länge des Ausgabesatzes bestimmt

Spalte 39 Löschen nach Ausgabe

Ein 'B' in dieser Spalte löscht das Feld nach der Ausgabe.

Ein 'C' in dieser Spalte setzt bei der Bildschirmdatei den Positionsanzeiger in dieses Feld.

Ein 'V' in dieser Spalte unterstützt eine variable Cursorpositionierung. In den Rechenbestimmungen kann mit einer MOVE- oder MOVEL-Operation das Feld CPGCUR mit der gewünschten Zeile und Cursor-Position gefüllt werden. CPGCUR ist 4 Stellen alpha und hat den Inhalt 'ZZSS'. 'ZZ' gleich Zeile und 'SS' gleich Spalte. Als Defaultwert wird '0101' angenommen.

Eine Ziffer in Verbindung mit DIAG in Spalte 40-43 gibt an, wie breit jeder Balken eines Diagramms dargestellt werden soll.

Spalte 40-43 Letzte Stelle im Ausgabebereich

Bildschirm-Datei:

Bei der Bildschirm-Datei wird in den Spalten 40 und 41 die Bildschirmzeile und in den Spalten 42 und 43 die Bildschirm-Spalte (1 bis 80) eingetragen, in welchen die letzte Stelle des Ausgabe-Feldes angezeigt werden soll.

Wenn die Bildschirmspalte größer als 99 Zeichen ist, wird in der Zehner-Stelle hexadezimal weiter gezählt (z.B. A0 für 100, B2 für 112, C0 für 120).

Eine Fehlermeldung 'ungültige Position' erfolgt immer dann, wenn eine Position eingetragen wird, die größer ist als die in der Dateizuordnung angegebene Zeilenzahl oder Zeilenlänge.

'DIAG' für die Diagrammausgabe (siehe Kapitel 2440)

Spalte 44 Gepackte Ausgabe

Ein 'P' in dieser Spalte bedeutet, dass das Feld in gepacktem Format ausgegeben wird. (Nicht für Bildschirm oder Drucker).

Ein 'B' in dieser Spalte bedeutet,d ass das Feld in binärem Format ausgegeben wird. (Nicht für Bildschirm oder Drucker).

Ein 'L' in dieser Spalte bedeutet, dass das Feld in logisch gepacktem Format ausgegeben wird. (Nicht für Bildschirm oder Drucker).

Ein 'X' in dieser Spalte bedeutet, dass die folgende Konstante einen Text in hexadezimaler Schreibweise enthält.

Ein 'A' in dieser Spalte bewirkt bei der Feldaufbereitung, dass für dieses Feld eine zwei Byte große S-Kon-

stante ausgegeben wird. Dies erleichtert die Erstellung von Parameterlisten für Unterprogramme. (Nicht für Bildschirm oder Drucker).

Ein 'O' in dieser Spalte unterstützt einen Overflow für die Aufbereitung einer ganzen Feldgruppe.

Spalte 45-47 Schutzstern '*', fließendes Währungszeichen '$', führendes Minuszeichen '-'

Zeichen in Verbindung mit DIAG ersetzt die Standarddarstellung der Balken ('*') durch dieses Zeichen, z.B. '$'

Spalte 45-52 Referenzdatei für DD

Spalte 45-70 Konstante oder Schablone

Spalte 75-80 Programmname

Bei fehlender Eintragung wird der Phasenname aus der H-Karte eingesetzt.

Datenformatierung - Attributbytes für 3270 4850

Ein Feld auf einem Bildschirm vom Typ 3270 ist definiert als der Bereich zwischen zwei Attributbytes.

Vor und hinter jedem Feld und jeder Konstanten, welche auf dem Bildschirm ausgegeben werden, steht ein Attributbyte.

Das ausgegebene Feld bzw. die Konstante endet stets auf der angegebenen Position. Das abschließende Attributbyte des Feldes befindet sich auf der Position hinter der angegebenen Position. Das standardmäßige abschließende Attributbyte hat die Eigenschaft 'geschützt, überspringen'. Durch die Angabe eines nachfolgenden Feldes bzw. einer Konstanten kann dieses Attributbyte durch das Anfangsattribut eines anderen Feldes ersetzt werden.

Das vor einem Feld stehende Attribut definiert die Eigenschaften des Feldes. Ist in Spalte 38 kein Attributcode angegeben, so wird standardmäßig entweder das CPG-Attribut 'S' (geschützt, überspringen) oder das Hardware-Attribut X'40' angenommen - bei Eintrag C in Spalte 49 der H-Karte.

Wird in Spalte 32-37 bzw. 45-70 kein Eintrag vorgenommen, so gibt CPG ein Attribut auf die Bildschirmposition aus, die in Spalte 40-43 angegeben ist. Ist in Spalte 38 kein Attributcode spezifiziert, so wird das standardmäßige Attributbyte ausgegeben.

Die Möglichkeit, einzelne Attributbytes auf dem Bildschirm auszugeben, kann für folgende Zwecke eingesetzt werden:

a) Felder auf dem Bildschirm definieren, ohne Felddaten ausgeben zu müssen.

b) Das derzeitige Attribut für ein Feld auf dem Bildschirm ändern.

c) Den Cursor auf eine andere Bildschirmposition stellen (hierzu wird ein Attributbyte in Spalte 38 mit einem 'C' in Spalte 39 spezifiziert).

Wird das Attribut 'V' für ein Feld oder eine Konstante angegeben, so wird das Anfangsattributbyte für das Feld aus dem einstelligen Feld CPGATR entnommen. Dieses muss im Programm definiert und auf den entsprechenden Hardware-Attributcode für das Feld gesetzt werden, bevor die Ausgabeoperation erfolgt.

Ist 'V' angegeben und enthält Spalte 32-37 keinen Feld- oder Feldgruppennamen und Spalte 45-70 keine Konstante, so wird der Inhalt von CPGATR als Attributbyte auf die angegebene Position ausgegeben.

Der CPG-Attributcode 'F' bedeutet, dass das erste Byte des auszugebenden Feldes bzw. der Konstanten als Attributbyte gelten soll. Es muss sich dabei um eine gültige Bitkombination für Attribute für 3270 handeln.

Es ist zu beachten, dass für diesen Code die Cursorpositionierung innerhalb des Feldes oder der Konstanten auch behandelt werden muss.

Alphanumerische Daten in der TWA 4855

Die Ausgabe von alphanumerischen Daten auf dem Bildschirm erfolgt entweder aus der in der TWA definierten Feldern oder als in Spalte 45-70 der Ausgabebestimmungen spezifizierte Konstanten.

Die Länge der ausgegebenen Daten entspricht der Länge des Feldes in der TWA bzw. der Länge der angegebenen Konstanten.

Werden in einer Konstanten & - Zeichen bzw. Hochkommata ausgegeben, so sind diese doppelt anzugeben.

Es können Konstanten und Felder ausgegeben werden, die sich über mehrere Zeilen auf dem Bildschirm erstrecken.

Die maximale Länge eines alphanumerischen Feldes beträgt 256 Bytes.

Numerische Daten 4860

In der TWA gespeicherte Felder können im Zeichenformat auf den Bildschirm ausgegeben werden. Bevor die Daten ausgegeben werden, wird das Feld entpackt. Dies führt in der Regel dazu, dass die letzte Ziffer des Feldes als Alphazeichen im Bereich A-R ausgegeben wird. Die Länge der Daten auf dem Bildschirm entspricht der erforderlichen Anzahl Bytes für das Feld in der TWA multipliziert mit 2 minus 1, d. h. CPG reserviert Platz für alle Ziffern, die das Feld enthalten kann.

Wenn ein Feld mit einer geraden Anzahl Ziffern definiert ist, folgt daraus, dass bei der Ausgabe auf dem Bildschirm eine zusätzliche Ziffer vorgesehen werden muss.

CPG unterdrückt nicht die Nullen in numerischen Feldern, die auf diese Weise ausgegeben werden.

Nullenunterdrückung und die Korrektur des abschließenden Alphazeichens sind möglich durch Spezifikation einer Schablone für das Feld oder durch den Eintrag eines Aufbereitungsschlüssels in Spalte 38.

Schablonen 4865

Eine Schablone wird in Spalte 45-70 der Ausgabebestimmungen spezifiziert und besteht aus einem in Hochkommata eingeschlossenen Literal. Schablonen werden für die folgenden Zwecke eingesetzt:

a) Unterdrückung führender Nullen bis einschließlich zu der angegebenen Position in einem Feld.

b) Die Ausgabe von einem oder mehreren Zeichen rechts von einem Feld, wenn dessen Inhalt negativ ist.

c) Einfügung von Leerzeichen links von dem Feld.

Die Länge der an dem Bildschirm ausgegebenen Daten entspricht der Länge der Schablone (d.h. der Anzahl Zeichen zwischen den Hochkommata) plus 1. Das zusätzliche Zeichen geht den Daten voraus und wird auf dem Bildschirm als führendes Leerzeichen angezeigt.

Aufbereitete Felder werden von links nach rechts abgearbeitet. Die einzelnen Ziffern des Feldes aus der TWA werden auf die in der Schablone als Leerzeichen oder Null angegebenen Positionen gestellt. Enthält das Feld in der TWA mehr Ziffern, als die Schablone vorsieht, so werden die Daten rechts abgeschnitten. Ungültige Ergebnisse können sich einstellen, wenn die Schablone mehr Ziffern vorsieht als das Feld enthält.

Nullenunterdrückung 4867

Durch Nullen in der Schablone wird die Unterdrückung führender Nullen gesteuert. Alle auftretenden führenden Nullen werden bei der Aufbereitung des Feldes unterdrückt und durch Leerzeichen ersetzt und zwar bis einschließlich zu der Stelle, auf der die Null in der Schablone steht.

Tritt in dem Feld vor der Null eine signifikante Ziffer auf, so endet die Nullenunterdrückung automatisch.

Neben der Unterdrückung führender Nullen unterdrückt diese Funktion auch alle Einfügungszeichen, die als Bestandteil der Schablone vor der Null spezifiziert sind. Wird eine signifikante Ziffer erkannt, so endet die Unterdrückung der Füllzeichen an dieser Stelle.

Ist in der Schablone keine Null angegeben, so gilt Nullenunterdrückung bis zum Ende der Schablone.

Einfügungszeichen 4868

Es folgen einige typische Beispiele für Einfügungszeichen:

/ Schrägstrich
, Komma
. Punkt

Diese werden, wie in der Schablone spezifiziert, ausgegeben, wenn sie nicht wegen führender Nullen in den Eingabedaten unterdrückt werden.

Ein & in der Schablone definiert eine Leerstelle (Blank) bei der Ausgabe. Innerhalb einer Schablone muss das '&' nur einmal pro Blank gesetzt werden.

Kennzeichnung negativer Beträge 4869

Die Zeichen, die rechts von der Schablone vor dem abschließenden Hochkomma angegeben sind, werden nur dann ausgegeben, wenn der Inhalt des aufzubereitenden Feldes negativ ist.

Ist der Inhalt des Feldes positiv, werden statt dessen ein oder mehrere Leerzeichen ausgegeben.

Typische Kennzeichnungen für negative Beträge sind '-' und 'CR'.

Einfügung führender Leerzeichen 4870

Alle links von der Schablone angegebenen Einfügungszeichen werden bei der Aufbereitung unterdrückt und durch Leerzeichen ersetzt.

Mit dieser Funktion können bei der Ausgabe auf dem Bildschirm links von einem Feld Leerzeichen eingefügt werden.

Länge der verarbeiteten Daten 4875

Die Anzahl der Ziffern aus dem Feld in der TWA, die in die Schablone übernommen werden, entspricht der erforderlichen Anzahl Bytes für das Feld multipliziert mit 2 minus 1.

Bei Feldern, die mit einer geraden Anzahl Ziffern definiert sind, muss eine zusätzliche linksbündige Zifferposition vorgesehen werden, wenn eine Schablone dafür definiert wird.

Wenn ein numerisches Feld mit Hilfe einer Schablone ausgegeben wird, wird die rechtsbündige Ziffer richtig als Zahl von 0 bis 9 angezeigt.

Feldgruppen 4876

Aus einer Feldgruppe können sowohl alphanumerische als auch numerische Daten auf den Bildschirm ausgegeben werden.

Soll die gesamte Feldgruppe ausgegeben werden, so muss der Name der Feldgruppe in Spalte 32-37 und die letzte Stelle des ersten Elements der Feldgruppe auf dem Bildschirm in Spalte 40-43 angegeben werden. Die nachfolgenden Elemente werden auf den darunter liegenden Zeilen auf denselben Zeichenpositionen ausgegeben, bis sämtliche Elemente auf dem Bildschirm stehen.

Wird eine Ausgabeposition so angegeben, dass auf dem Bildschirm nicht genügend Zeilen für alle Feldgruppenelemente vorhanden sind, so führt dies zu Fehlern bei der Verarbeitung.

Bei numerischen Feldgruppen kann eine Schablone in Spalte 45-70 angegeben werden. Aufgrund dieser Schablone werden die einzelnen Elemente der Feldgruppe formatiert. Ein Element mit dem Wert Null bewirkt jedoch, dass das gesamte Feld bei der Ausgabe nur Leerzeichen enthält.

Einzelne Elemente einer Feldgruppe werden dadurch ausgegeben, dass der Feldgruppenname in Spalte 32-35, anschließend ein Komma und eine Dezimalzahl (relativ zu 1) angegeben wird, wobei letztere das anzuzeigende Element bezeichnet. Wenn einzelne Elemente numerischer Feldgruppen anzeigt werden sollen, kann auch mit Schablonen gearbeitet werden.

Der von einem Element belegte Platz ist genauso groß wie bei jedem anderen alphanumerischen oder numerischen Feld mit demselben Modus und derselben Länge.

Überlappte Felder 4877

Die Ausgabebestimmungen für eine Bildschirmdatei werden sequentiell von oben nach unten entsprechend ihrer Reihenfolge im Programm aufgearbeitet.

Daher können Daten so ausgegeben werden, dass sie von einem nachfolgenden Feld bzw. einer Konstante überschrieben werden. Dies führt nicht zu Verarbeitungsfehlern und kann sinnvoll sein, wenn komplexe Bildschirmformate verarbeitet werden.

Balkendiagramme 4878

CPG enthält eine Funktion, aufgrund der eine Feldgruppe schnell und bequem als Balkendiagramm auf dem Bildschirm dargestellt werden kann.

Diese Funktion wird wie folgt spezifiziert:

Als erstes auf dem Bildschirm auszugebendes Feld den Namen der Feldgruppe in Spalte 32-37 und die Konstante DIAG in Spalte 40-43 angeben.

CPG baut die Balken für das Diagramm aus Sternen (*) auf. Soll hierfür ein anderes Zeichen verwendet werden, so muss dies in Hochkommata eingeschlossen in Spalte 45-47 angegeben werden.

Jeder Balken ist standardmäßig eine Spalte breit. Sollen die Balken breiter sein, so muss eine Zahl in Spalte 39 eingetragen werden. Dies gilt dann anstelle des Standardwertes.

Die zu formatierende Feldgruppe darf maximal 69 Ziffern enthalten. Diese wird errechnet, indem die Anzahl Ziffern pro Element in der TWA mit der Anzahl Elemente multipliziert wird.

Das Balkendiagramm wird in Zeile 2-23 des Bildschirms ausgegeben. Es wird so formatiert, dass es auf einen Bildschirm mit 1920 Zeichen passt.

Die Werte in der Feldgruppe werden als positive oder negative dezimale Ganzzahlen behandelt. Die verwendeten Werte (ohne Vorzeichen) werden in Zeile 23 des Bildschirms ausgegeben.

Die Feldgruppenwerte werden von der Diagrammroutine automatisch maßstabgerecht dargestellt, und am linken Rand des Bildschirms wird ein Maßstab ausgegeben.

Die positiven Werte des Diagramms werden oberhalb der X-Achse, die negativen Werte unterhalb der X-Achse angezeigt. Nachdem die Feldgruppe in den Ausgabebestimmungen spezifiziert ist, können Felder oder Werte angegeben werden, die die Feldgruppe auf dem Bildschirm überschreiben sollen.

Alle vor der Feldgruppe spezifizierten Felder oder Werte werden überschrieben, wenn diese formatiert wird.

Lichtstift-Auswahl 4880

Es gibt zwei unterschiedliche Anwendungsverfahren lichtstiftauswählbarer Felder in den Ausgabespezifikationen.

Bei Auswahl eines Feldes bewirkt die erste Methode eine Unterbrechung, die zweite nicht.

Bei Verwendung der ersten Methode muss jedes lichtstiftauswählbare Feld ein Leerzeichen als erstes Ausgabezeichen enthalten. Der Rest des Feldes soll mindestens ein alphanumerisches Zeichen enthalten.

Bei Benutzung der zweiten Methode muss jedes lichtstiftauswählbare Feld mit einem '?' oder einem '>' als erstem Zeichen ausgegeben werden. Der Rest des Feldes muss mindestens ein alphanumerisches Zeichen enthalten.

Beim Aufsuchen eines Feldes wird ein '?' als erstes Zeichen in '>' geändert und ein '>' als erstes Zeichen wird in '?' geändert.

Es tritt keine Unterbrechung auf.

Durch Drücken der Eingabetaste werden alle lichtstiftauswählbaren Felder mit einem '>' als erstem Zeichen ausgewählt. Die Bezugszahl wird gesetzt, wenn irgendwelche anderen Felder zusammen mit anderen spezifizierten Bezugszahlen ausgewählt werden. SP wird nicht gesetzt, jedoch die Lichtstift-Bezugszahlen. (61-62).

Für beide Methoden finden folgende Regeln Anwendung:

Bei Benutzung von CICS, TCP, SII, ETC usw. dürfen sich keine modifizierten Felder und lichtstiftauswählbare Felder auf der gleichen Bildschirm-Maske befinden.

Lichtstiftauswählbare Felder müssen mit ihren Attributbytes auf derselben Bildschirmzeile wie der Rest des Feldes angegeben werden.
Das Attribut muss eine Auswahl zulassen.

Feldaufbereitung 4885

Mit Hilfe von Ausgabebestimmungen für die Feldaufbereitung werden Daten entweder in ein Feld (bzw. eine Feldgruppe) in der TWA oder in einem vom jeweiligen TP-Monitor bereitgestellten Standardbereich übertragen.

Diese Funktion wird durch die EDIT-Operation in den Rechenbestimmungen aufgerufen. Wie diese Operation verwendet wird, ist in Abschnitt 2420 im einzelnen beschrieben.

Eine Ausgabebestimmung für die Feldaufbereitung besteht aus einer Satzbestimmung, die das aufzubereitende Feld kennzeichnet, und anschließend aus den Spezifikationen für die Daten, die in das Feld

übertragen werden sollen.

Die Ausgabebestimmungen für ein Feld werden entweder durch eine weitere Satzbestimmung im Rahmen der Ausgabebestimmungen oder durch das Datenende des CPG-Anwenderprogramms beschlossen.

Die Spezifikationen für die Feldaufbereitung werden als eigene Unterprogramme innerhalb der Ausgabebestimmungen behandelt. Diese werden bei der durch einen EXCPT-Befehl eingeleiteten Verarbeitung der Ausgabebestimmungen ignoriert.

Wenn die Verarbeitung einer Ausgabebestimmung für die Feldaufbereitung mit Hilfe eines EDIT eingeleitet wird, wird zu den Ausgabebestimmungen für das betreffende Feld verzweigt. Diese werden sequentiell von oben nach unten abgearbeitet, bis entweder die nächste Satzbestimmung im Rahmen der Ausgabebestimmungen oder das Datenende-Kennzeichen für den Quellencode des Programms erreicht wird. An dieser Stelle wird zu der Anweisung in den Rechenbestimmungen zurück verzweigt, die unmittelbar auf den EDIT-Befehl folgt.

Schutzsternschreibung 4887

Schutzsternschreibung wird beispielsweise beim Ausstellen von Schecks eingesetzt, um ein nachträgliches Ändern des gedruckten Betrages unmöglich zu machen. Dazu wird das aufbereitete numerische Betragsfeld linksbündig mit Sternen (*) aufgefüllt.

Im CPG ist die Schutzsternschreibung zusätzlich zum Aufbereitungsschlüssel unterstützt. Es braucht nur die Konstante '*' in Spalte 45 bis 47 der O-Karten angegeben zu werden.

Beispiel: Das Feld SUMME mit dem Inhalt 98765,55 wird mit Editcode K (Nullenunterdrückung, Komma, Tausenderpunkt, Minuszeichen) und Schutzsternschreibung ausgegeben. SUMME ist als elfstelliges numerisches Feld, davon zwei Dezimalstellen, vereinbart.

Ausgabe: *****98.765,55

Syntax : C SUMME K 1020 '*'

Besonderheit: Der Schutzstern in Kombination mit dem Edit-Code Y für Datumsaufbereitung bewirkt, dass bei leeren Datumsfeldern Blank angezeigt wird und dass führende Nullen unterdrückt werden.

Fließendes Währungszeichen 4888

Eine andere Möglichkeit des Schutzes bietet das fließende Währungszeichen. Dabei wird direkt vor einen Betrag ein Dollarzeichen ($) gesetzt, so dass eine mögliche spätere Manipulation des Betrages verhindert wird.

Das fließende Währungszeichen ist im CPG zusätzlich zu den Aufbereitungsschlüsseln unterstützt. Es braucht nur die Konstante '$' in

den Spalten 45-47 der O-Karte angegeben zu werden.

Beachte: Beim fließenden Währungszeichen wird intern das Ausgabefeld um ein Byte (für das Dollarzeichen) erweitert.

Beispiel: Das Feld SUMME mit dem Inhalt 98765,55 wird mit Editcode K (Nullenunterdrückung, Komma, Tausenderpunkt, Minuszeichen) und fließendem Währungszeichen ausgegeben. SUMME ist als elfstelliges numerisches Feld, davon zwei Dezimalstellen, vereinbart.

Ausgabe: $98.765,55

```
Syntax :  O                        SUMME K 1020 '$'
```

Führendes Minuszeichen 4889

In Verbindung mit den Edit-Codes J, K, L und M ist in den Spalten 45-47 der Eintrag '-' (Minus) unterstützt. Minus sorgt dafür, dass bei der Aufbereitung das Minuszeichen vor die erste Ziffer des Feldes gesetzt wird. Diese Aufbereitung wird z.B. bei der Ausgabe in Microsoft Excel benötigt.

Ausgabe: -3,75

```
Syntax :  O                        SUMME K   10 '-'
```

Bildschirmattribute 4900

Das Attribut-Byte der Bildschirmdatei.

Jedem Ausgabefeld auf dem Bildschirm geht ein Attribut-Byte voran, das die Eigenschaften des folgenden Datenfeldes beschreibt. Dieses Byte belegt eine Anzeige-Position auf dem Bildschirm und muss vom Programmierer beim Entwurf der Bildschirmmaske berücksichtigt werden.

Beispiel: Feld A ist 4 Stellen groß, die letzte Stelle im Ausgabebereich wird mit 126 angegeben. Das Ausgabefeld belegt die Positionen 123 bis 126, die Position 122 muss dann für das Attribut-Byte reserviert bleiben. Ein Überschreiben dieses Bytes würde die Eigenständigkeit des folgenden Datenfeldes aufheben und dieses Feld dem vorangehenden zuschlagen.

Attribut-Schlüssel.

Folgende Eintragungen sind unterstützt:

A ,B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, 0, 1, 2, 3, 4, 5, 6, 7, 8, * als Attribut oder Leerstelle.

Die Bedeutung der einzelnen Eintragungen siehe Abschnitt 7020.

'*' Attributfeld bzw. Attributfeldgruppe.
Das Ausgabefeld enthält in 4 Bytes den CPG-Attributschlüssel, den Extended Highlight Wert, sowie den Farbschlüssel. Diese Werte überschreiben dann die Attribute auf der angegebenen Position.
Handelt es sich beim Ausgabefeld um eine Feldgruppe, so werden die Attribute der nächsten Zeile auf der gleichen Position, jedoch vom zweiten Element der Attributfeldgruppe, überschrieben.
Bei monochromen Bildschirmen werden Farb- und Extended Highlight Werte automatisch ignoriert.
Bei Farbbildschirmen ist ein 'C' in Spalte 39 der F-Karte für den Bildschirm einzutragen.

Beispiel:

```
C   51              MOVEL'ARR '     F1
.
O         U B                   C 117 '.......'
O         *       51      F1      110
```

Bedeutet :
Eingabefeld auf Zeile 1 Position 17 ausgeben. Normalerweise eine Kann-Eingabe; daher Farbe blau. Wenn aber Schalter 51 gesetzt ist, wird das Attribut auf reversiv und rot sowie erhöhte Helligkeit geändert. Damit soll eine Muss-Eingabe gekennzeichnet werden.

'F' Feldattribut

Bei diesem Aufbereitungsschlüssel wird das 1. Byte des Feldes als Feldattribut interpretiert. Als Feldattribute gelten die für den Bildschirm vorgeschriebenen Bitkombinationen. Das Feldattribut wird auf dem Bildschirm als Blank angezeigt. Eine Cursor-Positionierung ist nicht möglich.

'S' Skip Feld

Ein Skip-Feld ist ein Feld, das vom Positionsanzeiger übersprungen wird. Wenn der Aufbereitungsschlüssel blank ist, wird Skip angenommen.

'V' Variables Attribut

Dieser Aufbereitungsschlüssel ermöglicht eine Veränderung des Attributbytes aufgrund von äußeren Bedingungen z.B doppelte Helligkeit bei fälligen Posten. Das Attributbyte kann dabei im Plattensatz mit abgespeichert werden.
Der Programmierer muss bei Verwendung des Aufbereitungsschlüssels 'V' ein 1 Byte großes Feld mit dem Namen 'CPGATR' definieren. Der Inhalt dieses Feldes wird bei der Ausgabe als Attribut eingesetzt.

Achtung: Als Attribute gelten in diesem Falle die für den Bildschirm vorgeschriebenen Bitkombinationen, nicht die CPG-Aufbereitungsschlüssel.

Der Programmierer kann durch eine einfache Eintragung in der CPG-Steuerkarte (C in Spalte 49) auf den Attributsatz der Hardware umschalten. Die Umschaltung gilt für das gesamte Programm, d.h. ein 'A' in Spalte 38 wird nicht mehr als CPG-Aufbereitungsschlüssel 'A', sondern als Hardware-Attribut 'C1' interpretiert.

Bildschirme 3279 Modell 2A (vierfarbige Bildschirme) werden von CPG in der Form unterstützt, wie dies hardwareseitig vom Hersteller vorgesehen ist und zwar aus der Kombination von geschützt, ungeschützt und einfache oder doppelte Helligkeit.

Dabei ist:

```
Geschützt,   einfach hell  =  Blau
Ungeschützt, einfach hell  =  Grün
Geschützt,   doppelt hell  =  Weiss
Ungeschützt, doppelt hell  =  Rot
```

Bei fehlender Eintragung (Blank) wird 'S' angenommen.

Attributbytes können auch einzeln auf den Bildschirm ausgegeben werden. Dadurch ist es möglich, die Feldeigenschaften während der Verarbeitung zu verändern, ohne dabei den Feldinhalt mehrfach zu übertragen. Die Ausgabebestimmung enthält in diesem Falle nur den Aufbereitungsschlüssel und die Position des Attributs.

```
Beispiel:    O             FELD  A  444
             O       20          P  440
```

Ein 4-stelliges Feld wird im Normalfall (N20) unge-

schützt auf den Bildschirm ausgegeben. Ist die Bezugszahl 20 an, so wird der Feldinhalt geschützt.

Attributbytes können auch z. B. zur Feldteilung in ein bestehendes Feld hineingesetzt werden.

Für andere Dateien wird die Spalte 16 bzw. 38 (Attribut-Byte) nicht unterstützt.

Siebenfarbiger Bildschirm 4910

Ab dem Release 1.0 ist der siebenfarbige Bildschirm voll unterstützt. Somit ist es möglich, jede beliebige Stelle auf dem Bildschirm in einer beliebigen Farbe darzustellen.

Die Farbattribute, erhöhte Helligkeit usw. werden nur generiert, wenn in Spalte 39 der F-Karte ein 'C' eingetragen ist, z.B.:

```
      FBILD                         CL3270
```

Spalte 16 Set Attribut Order

Bei einer Feldbestimmung kann ein '$' für 'Set Attribut Order' eingetragen werden. Diese Attribute gelten dann für die folgenden Felder.
Spalte 32 - 70 muss blank sein; Spalte 17 kann einen EH-Wert (erhöhte Helligkeit) enthalten, Spalte 18 eine der 7 Farbkürzel. Sollen Default-Werte gesetzt werden, wird in Spalte 17 und 18 einfach '00' eingetragen.

Spalte 17 Anzeigeart

Es können folgende Eintragungen vorgenommen werden:

'R' Reversiv

'B' Blinkend

'U' Unterstrichen

'0' Normale Anzeige (vorheriger Wert wird aufgehoben)

' ' Anzeige bleibt unverändert

Spalte 18 Farbe

In Spalte 18 stehen die Eintragungen für die entsprechenden Farben.

'B' Blau

'W' Weiss

'G' Grün

'R' Rot

'Y' Gelb

'T' Tuerkis

'P' Pink

'0' Normale Anzeige nur bei Set Attribut Order

' ' Farbe unverändert (siehe Bildschirmattribut)

Die mit Set Attribut Order definierte Farbe und Helligkeit gilt für alle anschließend angezeigten Felder bis zum nächsten Set Attribut Order bzw. bis zum Ende der Bildschirmausgabe.

Beispiel:

```
O           $RY
O                     124 'DIESER TEXT WIRD REVERSIV'
O                     148 'UND GELB (Y) ANGEZEIGT'
O           $00
O                     172 'AB HIER WIEDER NORMAL'
```

Die Set Attribut Order gelten nur für die Ausgabe; bei einer Bildschirmeingabe erscheinen die eingegebenen Zeichen in normaler Anzeige und in weisser bzw. grüner Farbe. (Siehe Bildschirmattribute). Bezugszahlen sind bei Set Attribut Order erlaubt.

Unabhängig von der Set Attribut Order können einem Feld durch Eintragung in Spalte 17 und 18 der Feldausgabe eigene Farb- und Helligkeitsattribute zugeordnet werden. Diese gelten speziell beim Eingeben des Feldes, jedoch auch, solange kein anderes Set Attribut Order vor dem Feld definiert ist, bei der Ausgabe des Feldes.

Beispiel:

```
O            RR        AC 324 'EINGABE REVERSIV UND ROT'
O           $BP
O                         424 'AUSGABE BLINKEND UND PINK'
O            UY  HUGO  A  448
```

Dieses Feld 'HUGO' wird blinkend und pink ausgegeben, jedoch bei der Eingabe erscheint es unterstrichen und gelb (Y).

Ein Beispielprogramm zum 7-farbigen Bildschirm siehe Seite 8910.

Aufbereitungsschlüssel 4950

Man unterteilt Aufbereitungsschlüssel in zwei Kategorien.
Einfache (X, Y, Z) und kombinierte Aufbereitungsschlüssel (1-4, A-D, J-M).

Wenn mit Aufbereitungsschlüsseln gearbeitet wird, muss in Spalte 49 der H-Karte ein 'E' eingetragen werden.
Um ein Feld aufzubereiten, wird der Aufbereitungsschlüssel in Stelle 38 der Ausgabebestimmungen eingetragen.
Das Bildschirm-Attribute-Byte wird dann in Spalte 16 der jeweiligen Feldbestimmung eingetragen.

Einfache Aufbereitungsschlüssel

Der Aufbereitungsschlüssel 'X' gewährleistet ein hexadezimales F als Vorzeichen für positive Felder. 'X' verändert keine negativen Zahlen und führende Nullen werden unterdrückt. (Anders als bei RPG).

Mit dem Aufbereitungsschlüssel 'Y' wird normalerweise ein 3 bis 6 oder 8 Stellen langes Datumsfeld aufbereitet.

Bei Eintrag eines 'Z' in Spalte 20 der Kundenkonfiguration wird die äußerste linke Null angezeigt, ansonsten nicht. Dies gilt auch für das Feld UDATE.

Der Aufbereitungsschlüssel 'Z' entfernt das Vorzeichen (plus und minus) bei einem numerischen Feld und führende Nullen werden unterdrückt. Der Dezimalpunkt wird nicht aufbereitet und nicht gedruckt.

Kombinierte Aufbereitungsschlüssel:

Die kombinierten Aufbereitungsschlüssel (1-4, A-D, J-M) bestimmen die Interpunktion von numerischen Feldern.

Alle Aufbereitungsschlüssel unterdrücken führende Nullen links des Dezimalzeichens.
Wurden für das Feld Dezimalstellen definiert und eine Nullenunterdrückung vorgesehen, wird das Dezimalkomma nur gedruckt, wenn das Feld nicht Null ist.

Soll ein Null-Saldo nicht gedruckt werden, werden Leerstellen gedruckt. Soll es jedoch angedruckt werden und das Feld ist gleich Null, wird folgendes gedruckt:

0	Für ein Feld ohne Dezimalstellen
0,00	Für ein Feld mit 2 Dezimalstellen

Wird ein Aufbereitungsschlüssel zum Drucken einer ganzen Feldgruppe angegeben, erscheinen vor jedem Element außer dem ersten zwei Leerstellen.

Eine Nullenunterdrückung wird erst ab der zweiten Stelle links vom Dezimalkomma wirksam, dass heisst alle Null-Salden und alle Salden mit Null-erten links vom Komma werden mit einer führenden Null gedruckt. (0,00 oder 0,04).

Die Aufbereitungsschlüssel 1-4, A-D, J-M unterstützen fließendes Währungszeichen und Schutzsternschreibung. Wird im Konstantenfeld ab Stelle 45 eine 1-stellige Konstante '*' eingetragen, so wird für dieses Feld ein Schutzstern statt eines Blanks für alle nicht besetzten Ziffern gedruckt bzw. angezeigt.

Ein '$' bewirkt, dass unmittelbar vor der letzten Ziffer ein fließendes Währungszeichen gesetzt wird. Die Angabe eines '$' vergrößert die Feldlänge intern um eine Stelle. Dies ist bei der Feldausrichtung zu berücksichtigen.

Zusammenfassung der kombinierten Aufbereitungsschlüssel:

Aufbereitetes Format	Darstellung negativer Werte		
	ohne Vorzeichen	CR	_
Drucken mit Kolonnentrennzeichen und führenden Nullen	1	A	J
Drucken mit Kolonnentrennzeichen, führende Nullen unterdrückt	2	B	K
Drucken ohne Kolonnentrennzeichen Drucken Nullbeträge	3	C	L
Drucken ohne Kolonnentrennzeichen, führende Nullen unterdrückt	4	D	M

Beispiele für die Benutzung von Aufbereitungsschlüsseln:

Aufbereitungs-schlüssel	Positive Zahl mit 2 Dezimal-stellen	Positive Zahl ohne Dezimal-stellen	Negative Zahl mit 3 Dezimal-stellen
Nicht aufbereitet	1234567	1234567	00120
1	12.345,67	1.234.567	0,120
2	12.345,67	1.234.567	120
3	12345,67	1234567	0,120
4	12345,67	1234567	120
A	12.345,67	1.234.567	0,120CR
B	12.345,67	1.234.567	120CR
C	12345,67	1234567	0,120CR
D	12345,67	1234567	120CR
J	12.345,67	1.234.567	0,120-
K	12.345,67	1.234.567	120-
L	12345,67	1234567	0,120-
M	12345,67	1234567	120-
X	1234567	1234567	12?
Y			0/01/20
Z	1234567	1234567	120

Fortsetzung der Aufbereitungsschlüssel siehe folgende Seite.

Aufbereitungs-schlüssel	Negative Zahl ohne Dezimalstellen	Nullbetrag mit 2 Dezimalstellen	Nullbetrag ohne Dezimalstellen
Nicht aufbereitet	000120	000000	000000
1	120	0,00	0
2	120		
3	120	0,00	0
4	120		
A	120CR	0,00	0
B	120CR		
C	120CR	0,00	0
D	120CR		
J	120-	0,00	0
K	120-		
L	120-	0,00	0
M	120-		
X	12?		
Y	0.01.20	0.00.00	0.00.00
Z	120		

Bei der amerikanischen Schreibweise sind Dezimalzeichen und Tausenderpunkt vertauscht und beim Datum wird / statt . verwendet. Um diese Darstellung zu erreichen, muss in der CPGURSIT ein entsprechender Eintrag vorgenommen werden, siehe Installationshandbuch.

Bei der Datumsaufbereitung mit Y können die führenden Nullen unterdrückt werden. Ist ein Datumsfeld gleich Null, so wird Blank angezeigt. Diese Aufbereitung erreicht man, indem man zusätzlich zum Edit-Code Y einen Schutzstern angibt.

H L 1 5000

HL1-Programmierung. 5010

HL1-Programme werden in der Programmiersprache CPG geschrieben.

HL1-Programme können so geschrieben werden, dass sie sich im äußeren Aufbau und in der Programmierung nicht von CPG-Programmen unterscheiden.

Zur Erstellung von HL1-Programmen ist jedoch ein HL1-Compiler erforderlich, der mit EXEC HL1 aufgerufen wird. Dieser Compiler schließt CPG ein und kann über eine Eintragung in der Header-Karte auch normale CPG-Programme generieren.

HL1-Programmbausteine können aus jedem CPG-Programm und aus jedem HL1-Baustein mit der Operation 'EXECUTE HL1-Modul'

```
C                       EXHM NAME
```

aufgerufen werden. Faktor 2 enthält dabei den Namen des HL1-Bausteins der verarbeitet werden soll.

Siehe Anwendungsbeispiel 1 (HL1) Abschnitt 8

HL1-Bausteine sind object-kompatibel, das heisst sie können ohne jede Änderung und ohne Neu-Umwandlung unter jedem TP-Monitor verarbeitet werden, für den ein CPG-Interface verfügbar ist.

Phasennamen für HL1-Programme sind bis zu 6 Stellen lang und frei wählbar.

Die Phasennamen für HL1-Bausteine werden vom Compiler links um die Konstante 'HM' erweitert, d.h. das Programm XZEIL1 wird unter dem Namen HMXZEIL1 katalogisiert. In der Core-Image-Library sind folglich alle HL1-Bausteine unter 'HM' zu finden, wodurch eine Unterscheidung von allen anderen Programmen gegeben ist.

HL1-Bausteine sind nicht selbständig ausführbar.

Der Aufruf erfolgt ausschließlich aus einem Hauptprogramm oder einem anderen HL1-Baustein mit Hilfe der Operation EXHM. Hauptprogramme werden ebenfalls durch HL1 erstellt.

Online-Hauptprogramme werden generiert,wenn Spalte 51 der Header-Karte ein 'C' enthält.

Hauptprogramme müssen die Dateizuordnungskarten aller Dateien enthalten, die auf einer beliebigen nachfolgenden Stufe vom Programm angesprochen werden.

HL1-Programme sind reentrant geschrieben, das generierte Assembler-Quellenprogramm enthält nur reine Assembler-Instruktionen und ist frei von allen Ein-Ausgabe-Makroinstruktionen und frei von Datenfeldern.

Alle Makro-Instruktionen werden in einem zentralen Interface zusammengefasst, das im Bedarfsfalle eine schnelle Anpassungsmöglichkeit bei einem Wechsel der Basis-Software bietet und dadurch den Anwendungen einen hohen Grad von Flexibilität gibt.

Der Datenkanal 5020

Alle Datenfelder werden in einer eigenen Transaction Work Area zusammengefasst, die bei der Online-Verarbeitung jedem Terminal zugeordnet werden kann, das auf das Programm zugreift.

Jeder HL1-Baustein hat dabei seine eigene private Transaction Work Area (PWA - Private Work Area) mit einem eigenen, von anderen Bausteinen unabhängigen Bezugszahlensatz von 99 Schaltern.

HL1-Bausteine sind unabhängig voneinander und können untereinander Daten austauschen.

Der Programmierer kann beim Aufruf eines HL1-Bausteins einen Datenkanal in die private TWA dieses Bausteins übertragen. Der Name dieses Datenkanals wird in diesem Falle im Ergebnisfeld der Operation eingetragen.

```
Beispiel      C                    EXHM MODUL     DATEN
```

Der Datenkanal muss in diesem Falle in den Eingabebestimmungen wie folgt beschrieben werden:

```
             IDATEN   HS
             I                                      1   6 KDNR
             I                                      7  24 NAME
             I                                  P 25  260NUMMER
```

Bei numerischen Datenfeldern muss in Spalte 43 ein 'P' eingetragen werden.

Außerdem müssen die Felder KDNR, NAME und NUMMER im Baustein 'MODUL' in der E-Karte oder Datenbeschreibungskarte (siehe folgende Seite) als erste Felder mit 6, 17 und 3,0 Stellen definiert werden.

Die Felder KDNR, NAME und NUMMER werden dann beim Aufruf des Programms

(EXHM) in die private TWA des Bausteins 'PROG' übertragen. Nach Ausführung des Bausteins 'MODUL'werden die eventuell veränderten Feldinhalte wieder zurückübertragen und der für die private TWA benötigte Speicher wird wieder frei gegeben.

Die Eintragung 'HS' in Spalte 15 und 16 der Satzbestimmung bewirkt für die Beschreibung eines Datenkanals, dass für diese Eingabebestimmungen keine Optimierung durchgeführt wird.

Siehe Anwendungsbeispiel 3 (HL1).

Datenbeschreibung. 5030

Der Datenkanal wird im jeweiligen Baustein in der Datenbeschreibung beschrieben. Die Datenbeschreibung erfolgt durch die Datenbeschreibungskarte. Diese Karte hat folgenden Aufbau:

Formularbeschreibung:

Spalte 06		Kartenart (muss 'D' sein)
	15 - 20	Feldname (oder 15 - 16 Bezugszahl)
	24 - 27	Anzahl Felder
	28 - 30	Feldlänge
	32	Anzahl Dezimalstellen
	40 - 75	Text (zur Dokumentation)
	75 - 80	Phasenname

Der Datenkanal, der aus einer Kontonummer und einem Abrechnungskreis besteht,wird wie folgt beschrieben:

```
D        KONTO          7         KONTONUMMER DER DEBITORENDATEI
D        AK             2         ABRECHNUNGSKREIS
D*-------------------------------------------------------------
```

Die Beschreibung des Datenkanals muss grundsätzlich am Anfang des Programms liegen. Die Datenbeschreibungskarten liegen unmittelbar nach den Dateizuordnungskarten. Die Beschreibung des Datenkanals kann auch über die E-Karte (erweiterte Dateizuordnung) erfolgen, jedoch dürfen beide Karten nicht gemischt verwendet werden.

Ein Beispiel für die Übertragung eines Datenkanals wird im Anwendungsbeispiel 3 gezeigt. In diesem Beispiel wird der Datenkanal im Baustein 'FKAT' mit Hilfe von E-Karten beschrieben.

```
E                   NOTE          3 0
E                   FEHLER       24
E*-------------------------------------------------------------
```

Bei Verwendung von Datenbeschreibungskarten würde dieses Beispiel wie folgt abgeändert.

```
D        NOTE          3 0       FEHLERSCHLÜSSEL (LFD. NR)
D        FEHLER       24         KLARTEXT DER FEHLERMELDUNG
D*-------------------------------------------------------------
```

Bezugszahlen. 5040

Jeder HL1-Baustein hat seinen eigenen Satz von 99 Bezugszahlen. Allgemeine Schalter (T- und C-Schalter und PF-Tasten) gelten immer für die gesamte Anwendung. Dabei gelten die Regeln des CPG.

Der Programmierer hat bei HL1 die Möglichkeit, Bezugszahlen der nächst höheren Verarbeitungsstufe abzufragen und zu verändern. Dies erfolgt durch die Operationen 'GETIN' (GET Indicator) und 'PUTIN' (PUT Indicator).

Soll beispielsweise zum Punkt X verzweigt werden, wenn in der nächst höheren Stufe die Bezugszahl 15 gesetzt ist, so sind folgende Anweisungen erforderlich:

```
C                    GETIN                    15
C   15               GOTO X
```

Ebenso kann eine Bezugszahl auf der nächst höheren Stufe wie folgt gesetzt werden:

```
C                    SETON                    15
C                    PUTIN                    15
```

Bezugszahlen können zwischen den Bausteinen ausgetauscht werden.

Bezugszahlen können außerdem zwischen dem Hauptprogramm und einem HL1-Baustein auf einer beliebigen Ebene ausgetauscht werden. Dies erfolgt im HL1-Baustein durch die Operationen 'GETMI' (GET MAIN Indicator) und 'PUTMI' (PUT MAIN Indicator).

Soll beispielsweise zum Punkt Y verzweigt werden,wenn im Hauptprogramm die Bezugszahl 20 gesetzt ist, so sind folgende Anweisungen erforderlich.

```
C                    GETMI                    20
C   20               GOTO Y
```

Ebenso kann eine Bezugszahl im Hauptprogramm wie folgt gesetzt werden:

```
C                    SETON                    20
C                    PUTMI                    20
```

Die Operation 'PUTMI' bewirkt dabei, dass der Inhalt des Schalters 20 im Hauptprogramm dem Inhalt des Schalters im Baustein gleichgesetzt wird, d.h. ist die Bezugszahl gelöscht, so wird auch die Bezugszahl im Hauptprogramm gelöscht, ist die Bezugszahl gesetzt, so wird auch die Bezugszahl im Hauptprogramm gesetzt.

Verarbeitung. 5050

HL1-Bausteine können von jedem CPG-Online-Programm und von jedem HL1-Batchprogramm aufgerufen werden.

HL1-Bausteine sind keine selbständig ausführbaren Programme. Sie sind in etwa vergleichbar mit Unterprogrammen, weil sie nur unter Steuerung eines übergeordneten Programms ablaufen können. Der Unterschied zu herkömmlichen Unterprogrammen besteht jedoch darin, dass sie nicht fest mit dem aufrufenden Programm verbunden sind, sondern je nach Bedarf erst zur Ausführungszeit zum Programm hinzu geladen werden.

HL1-Bausteine liegen daher nicht wie Unterprogramme als Bestandteile vieler Programme mehrfach in der Bibliothek für ausführbare Programme, sondern nur jeweils einmal.

HL1-Programme brauchen sehr viel weniger Platz in der Bibliothek als herkömmliche Programme.

Bei Online-Verarbeitung werden HL1-Bausteine wahlweise unter Steuerung des TP-Monitors oder in einen Programmpool der HL1-Methodenbank geladen. Der Programmpool wird selbsttätig initialisiert beim ersten Aufruf eines HL1-Bausteins. Alle weiteren Bausteine werden beim jeweils ersten Aufruf in die Methodenbank geladen.

Die Methodenbank ist eine zentrale Bibliothek für ausführbare Unterprogramme, auf die alle in der TP-Partition ablaufenden Programme zugreifen können.

HL1-Bausteine müssen daher nicht in die Programmtabellen des TP-Monitor-Programms eingetragen werden, jedoch ist eine Eintragung in einer HL1-Tabelle erforderlich. Bei dieser Eintragung handelt es sich jedoch nur um den Namen des Bausteins. Alle bei herkömmlichen Programmtabellen erforderlichen Parameter werden von HL1 selbst ermittelt.

VSE-Benutzer können ihre HL1-Bausteine von CICS verwalten lassen. Hieraus ergeben sich entscheidende Vorteile während der Test-Phase, aber auch wenn die Bausteine relativ groß sind. CICS versucht bei einem Newcopy diesen Platz wieder zu verwenden, falls das HL1-Modul nicht über eine 2K-Grenze verändert wurde. Der HL1-Aufruf und die Ausführung eines Bausteins werden hiervon nicht beeinflusst.

Baugruppen. 5060

HL1-Bausteine können vom Programmierer zu Baugruppen zusammengefasst werden. Eine Baugruppe kann dann von einem Programm wie ein HL1-Baustein aufgerufen werden.

Baugruppen sind wie HL1-Programme stufenweise gegliedert, das heisst eine Baugruppe kann sowohl aus Einzelbausteinen, als auch aus anderen

Baugruppen bestehen. Die Anzahl der Stufen ist unbegrenzt. Extrem gesehen könnten alle Anwendungen zu einem einzigen HL1-Programm zusammengefasst werden.

Alle HL1-Bausteine oder Baugruppen münden dabei in der höchsten Stufe in einem Hauptprogramm, das die Verbindung zum übergeordneten Steuerprogramm darstellt. Nur dieses Hauptprogramm in der obersten Stufe gilt für den TP-Monitor oder für das Job-Control-Programm als Anwendungsprogramm im herkömmlichen Sinne. Nur dieses Programm muss daher bei einem Release-Wechsel des übergeordneten Steuerprogramms im Bedarfsfalle neu umgewandelt werden. Alle anderen Bausteine sind unabhängig von der verwendeten Basissoftware und sind dabei objectkompatibel, das heisst sie müssen nicht einmal umgewandelt werden, wenn die Steuerung über einen anderen TP-Monitor erfolgt.

Wird in einem Baustein eine Datei angesprochen, so muss im Header in der Spalte 34 ein Eintrag für Datasetverarbeitung vorgenommen werden.

Wie im Kapitel Bezugszahlen bereits beschrieben, kann der Programmierer Bezugszahlen im Hauptprogramm abfragen oder verändern.

Bei der Online-Verarbeitung wird bei Betätigung der Löschtaste der Bildschirmtastatur die Steuerung an das Hauptprogramm und damit unmittelbar an das übergeordnete Steuerprogramm zurückgegeben.

Um eine optimale Gliederung der Struktur eines Programms zu erreichen, kann der Programmierer auch einen HL1-Baustein auf einer beliebigen Stufe in den Rang eines Hauptprogramms erheben, wobei dieser Baustein alle oben genannten Eigenschaften des Hauptprogramms erhält, mit Ausnahme der Eigenständigkeit und damit der Notwendigkeit einer Eintragung in den Programm-Steuertabellen des übergeordneten Steuerprogramms.

Dies wird dadurch erreicht, dass in einem Baustein 'A' für eine Baugruppe 'B' eine neue Transaction-Work-Area initialisiert wird.

Der Baustein 'A' wird damit zum Hauptprogramm für alle über den Baustein 'B' aufgerufenen Bausteine oder Baugruppen. Im Programm 'A' wird dazu lediglich beim Aufruf des Bausteins 'B' ein 'I' in Spalte 53 der Rechenbestimmungen eingetragen. Beispiel:

```
    C                   EXHM B                  I
```

In diesem Fall beziehen sich die Operationen 'GETMI' und 'PUTMI' in Baustein 'B' oder einem nachgeordneten Baustein auf die Bezugszahlen des Programms 'A'. Die Operationen 'EDIT' oder 'SELCT' CPGTWA im Baustein 'B' oder einem nachgeordneten Baustein beziehen sich auf die TWA des Programms 'A'. Bei Online-Verarbeitung wird bei Betätigung der Löschtaste der Bildschirmtastatur im Baustein 'B' oder einem nachgeordneten Baustein die Verarbeitung mit der Anweisung fortgesetzt, die der Operation

```
    C                   EXHM B                  I
```

folgt.

Zu beachten ist jedoch, dass das Programm A damit nicht zum selbstständigen Programm wird, das von einem TP-Monitor oder vom Job-Control-Programm ausgeführt werden kann. Selbständig ist nach wie vor

nur das Programm, das mit 'C' in Spalte 51 der H-Karte umgewandelt wurde.

Besonderheiten bei der Datei-Verarbeitung mit HL1 5065

Grundsätzlich besteht bei der Dateiverarbeitung innerhalb des HL1-Hauptprogramms und innerhalb von HL1-Bausteinen kein Unterschied zur Dateiverarbeitung im CPG.

Bei der Dateiverarbeitung in Bausteinen ist aber zu beachten, dass die Dateibeschreibung (F-Karte) einer Datei, die im Baustein verarbeitet wird, unter bestimmten Bedingungen auch ins zugehörige Hauptprogramm aufgenommen werden muss.

Es gibt zwei Typen von Bausteinen: Standardbausteine (Spalte 34 = ' ') und Dataset-Bausteine (Spalte 34 der H-Karte = 'D' oder 'S')

Standardbausteine sollten dann eingesetzt werden, wenn sie Funktionen haben, die ohne Dateizugriffe auskommen. Standardbausteine geben den benutzten Arbeitsbereich (PWA) nach jedem Aufruf wieder frei.

Wenn Standardbausteine Dateien oder FIND-Tabellen verarbeiten, müssen die F-Karten für diese Dateien auch in allen Hauptprogrammen liegen, die mit dem Standardbaustein arbeiten.

Wenn ein HL1-Hauptprogramm, das Standardmodule anspricht, mit einer TWA arbeitet, die größer als 4K ist, so müssen die CPG-internen Keyfelder CPGKxx innerhalb der ersten 4K der TWA vom Programmierer selbst definiert werden (z.B. mit einer D-Karte).

Dataset-Bausteine werden immer dann eingesetzt, wenn Dateien oder Datenbanken zu verarbeiten sind. Bei solchen Bausteinen bleibt die PWA (private Work Area) innerhalb der Task erhalten, d.h. die Inhalte aller Benutzerfelder stehen zur Verfügung (Eintrag 'D' in Spalte 34 der H-Karte). Bei 'S' werden die Benutzerfelder in der PWA initialisiert, nur die CPG-internen Dateirettfelder und Keyfelder bleiben erhalten.

Bei Einsatz von Dataset-Modulen brauchen die F-Karten des Moduls nicht in die zugehörigen Hauptprogramme aufgenommen zu werden. **Dateiverarbeitung sollte grundsätzlich nur mit Dataset-Modulen gemacht werden**.

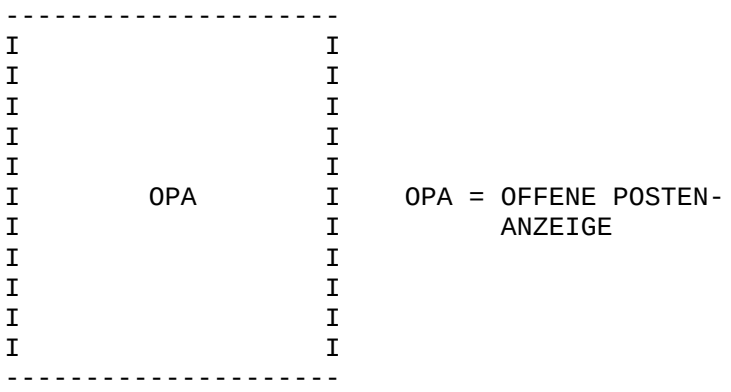
Bei sequentieller Verarbeitung mit Dataset-Modulen verbessert sich die Performance sowohl online als auch im Batch. Es ist allerdings zu beachten, dass die PWA, die in diesen Modulen erhalten bleibt, im System Platz benötigt. Dies wird insbesondere bei Dialogprogrammierung bemerkbar sein. Außerdem belegen Dataset-Module bei READ oder CHAIN(U) jeweils eigene VSAM-Strings.

Programmstruktur 5070

HL1 gibt dem Programmierer die Möglichkeit, Programme so zu strukturieren, dass die Aufgabenstellung in ihre Einzelteile zerlegt wird und dadurch sowohl bei der Erstellung, als auch bei der Wartung der Programme die Aufgabenstellung eindeutig auf die konkrete Anwendung beschränkt bleibt.

Dies soll nachfolgend am Beispiel eines Programms für die Anzeige der offenen Posten einer Buchhaltung aufgezeigt werden. Bei herkömmlicher Programmierweise wurde in diesem Falle ein Programm 'OFFENE POSTEN ANZEIGE' geschrieben.

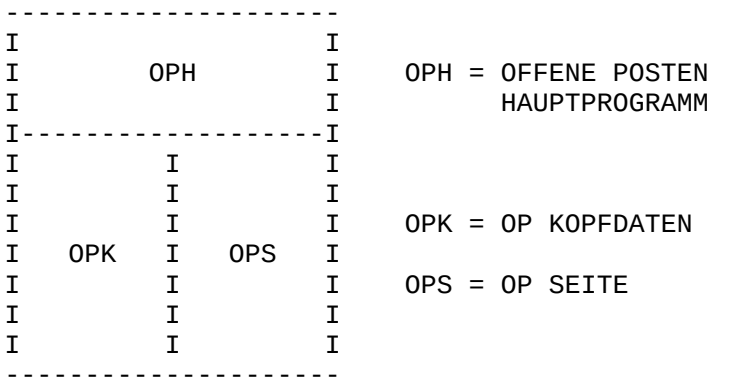
Bild 1



In diesem Programm werden Daten aus einem Kontenstammsatz gelesen. In diesem Kontenstammsatz ist unter anderem vermerkt, ob zu diesem Konto 'Offene Posten' vorhanden sind und wo diese in der OP-Datei zu finden sind.

Daraus ergibt sich zunächst eine funktionale Teilung der Aufgabenstellung in die Unterfunktionen 'Stammdaten lesen und anzeigen' und 'offene Posten lesen und anzeigen'. Für beide Funktionen gelten unterschiedliche Bedingungen. Wir können daher Bild 1 wie folgt erweitern.

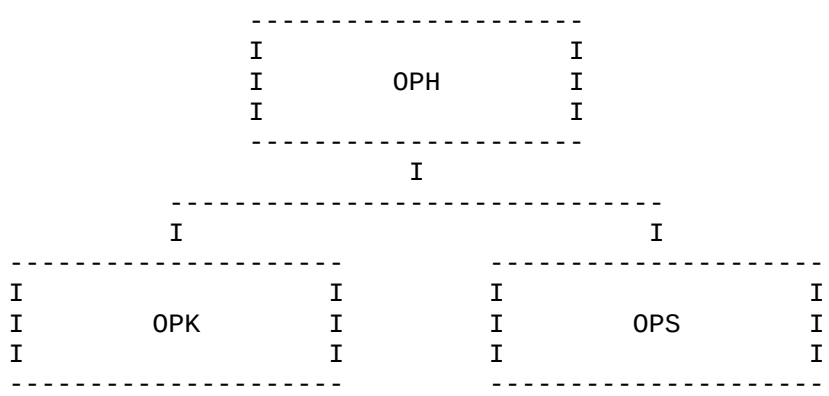
Bild 2



Kopfdaten sind die Kontenstammdaten, die im Kopfteil des Bildschirms angezeigt werden. Unterhalb der Kopfdaten wird dann jeweils eine Seite offene Posten angezeigt, wobei die Daten eines offenen Postens jeweils eine Zeile belegen.

Wir können für Bild 2 auch die folgende Darstellungsform wählen:

Bild 3

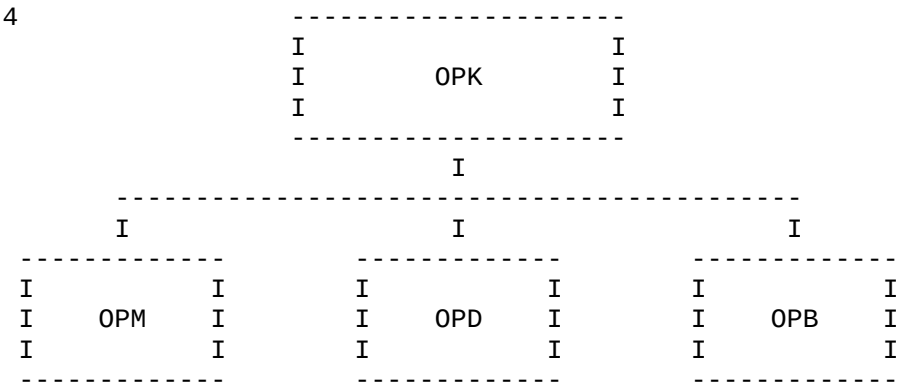


Dieses Bild verdeutlicht, dass aus dem einen Programm drei unabhängige Bausteine entstanden sind, die unabhängig voneinander erstellt oder später gewartet werden können und die jeweils für sich wieder eine eigene Programmeinheit darstellen.

Wir können folglich den Baustein 'OPK' als ein eigenes Programm ansehen und überlegen, ob hier eine weitere funktionale Gliederung möglich ist.

In diesem Programm muss zunächst eine Maske auf den Bildschirm ausgegeben werden, die die Eingabe der Kontonummer und des Abrechnungskreises ermöglicht. Danach werden die Kontenstammdaten gelesen und auf den Bildschirm ausgegeben. Wir unterteilen den Baustein 'OPK' in drei Unterfunktionen:

Bild 4



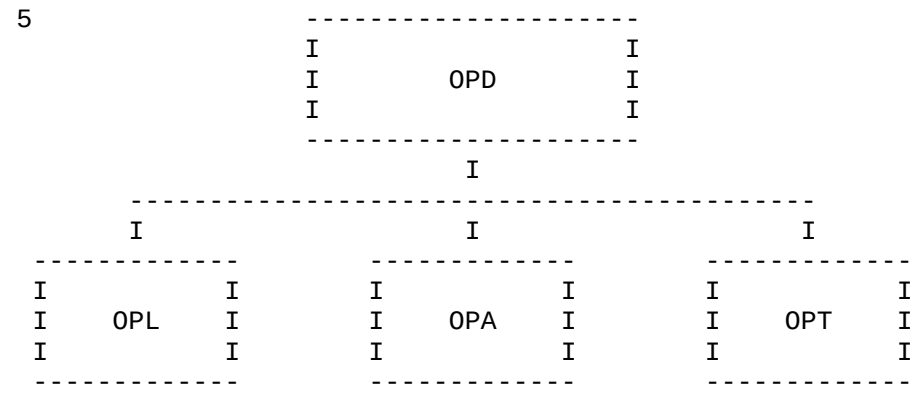
'OPM' ist ein Baustein, der eine Maske für den Bildschirmkopf erstellt und die Eingabefelder vorformatisiert. 'OPD' ist ein Baustein, der die erforderlichen Kontenstammdaten liest und soweit erforderlich für die Verarbeitung aufbereitet. 'OPB' ist ein Baustein, der die Kontenstammdaten auf den Bildschirm ausgibt.

Wir untersuchen jetzt, ob für die drei neuen Bausteine eine weitere Unterteilung möglich ist. Für die Bausteine 'OPM' und 'OPB' erscheint dabei eine weitere Unterteilung nicht sinnvoll. Der Baustein 'OPD' jedoch soll zunächst ein Datensegment aus der Kontenstammdatei lesen und dabei die Daten bereits so aufbereiten, dass sie dem aufrufenden Programm in der Form zur Verfügung stehen, wie sie gebraucht werden.

Dabei soll der Schlüssel für die Lieferbedingungen bereits in Klartext übersetzt werden und die in der Datei verstreut liegenden abrechnungskreisbezogenen Daten zu einer Feldgruppe vereint werden.

Der Baustein 'OPD' wird also wie folgt weiter untergliedert:

Bild 5



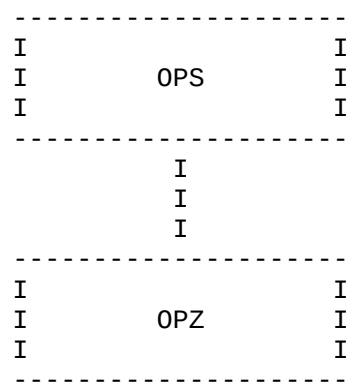
Der Baustein 'OPL' liest dabei aus dem Kontenstammsatz den Teil der Daten, die für die Anzeige und weitere Verarbeitung der offenen Posten erforderlich sind. Der Baustein 'OPA' fasst die in der Datei verstreut liegenden abrechnungskreisbezogenen Daten zu einer Feldgruppe zusammen, die die Weiterverarbeitung entsprechend vereinfacht. Der Baustein 'OPT' schließlich übersetzt den numerischen Schlüssel für die Lieferbedingungen in Klartext.

Da für die drei neuen Bausteine eine weitere Untergliederung nicht mehr sinnvoll erscheint, wenden wir uns dem Baustein 'OPS' aus Bild 3 zu.

Hierin sollen die Daten der offenen Posten zeilenweise zu einer Bildschirmseite zusammengefasst werden. Wir werden daher die OP-Seite in diesem Baustein aufbauen, wobei wir die immer wieder gleiche Aufbereitung der Einzelteile in einen anderen Baustein verlegen.

Es entsteht also folgende Gliederung:

Bild 6

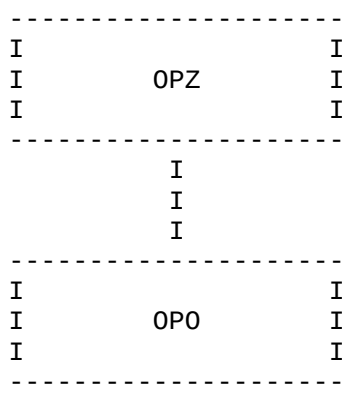


Der Baustein 'OPZ' bereitet eine Bildschirmzeile für die Ausgabe auf. Da Daten aus der OP-Datei auch für andere Zwecke gelesen werden können, werden wir für das Lesen der Daten einen eigenen Baustein vorse-

hen, der für alle Anwendungen den Zugriff zu dieser Datei auf einen Programmbaustein konzentriert.

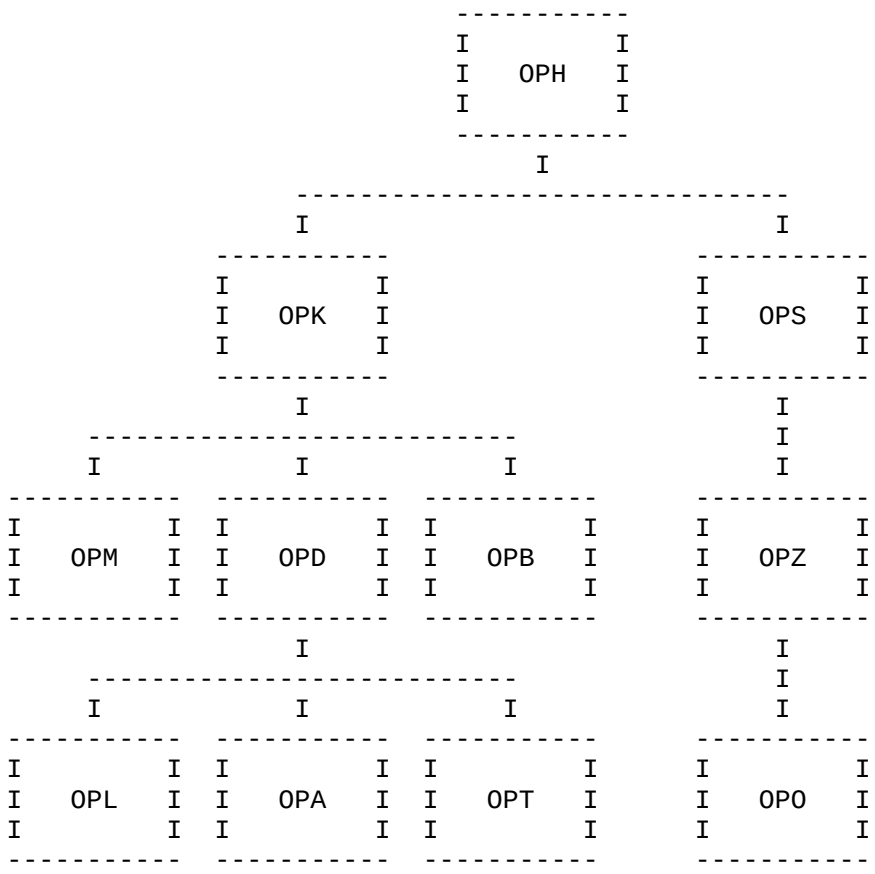
Wir erhalten damit ein ähnliches Bild wie Bild 6 für den Baustein 'OPZ'.

Bild 7



Im Baustein 'OPO' werden die OP-Daten gelesen und dem Anwendungsprogramm in aufbereiteter Form zur Verfügung gestellt. Damit ist die Strukturierung des Programms zunächst abgeschlossen und wir können die Bilder 3 bis 7 zum Strukturdiagramm für diese Anwendung zusammenfassen.

Bild 8



Aus Bild 8 ist zwar die Programmstruktur ersichtlich, das Bild gibt jedoch noch keine Aussage über den Datenfluss zwischen den einzelnen Bausteinen, denn wie oben bereits beschrieben, können Daten und Bezugszahlen zwischen den Bausteinen ausgetauscht werden.

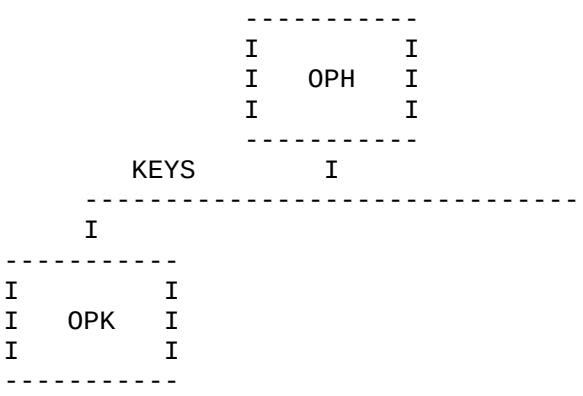
Daher wählen wir für das Strukturdiagramm eine andere Darstellungsform, die uns sowohl eine Auskunft über den Datenfluss gibt, als auch die maschinelle Verknüpfung der Einzel-Bausteine zu einer Gesamtübersicht über die Anwendung ermöglicht.

Dazu fassen wir die Daten und Bezugszahlen, die zwischen zwei Bausteinen ausgetauscht werden zu einer Einheit zusammen, die wir den 'Datenkanal' nennen.

Werden beispielsweise zwischen den Bausteinen 'OPK' und 'OPH' die Felder 'Kontonummer' und 'Abrechnungskreis' ausgetauscht, so fassen wir diese Felder zum Datenkanal 'KEYS' zusammen.

In Bild 8 würde dies wie folgt dargestellt werden:

Bild 9



Die von HL1 benutzte Darstellungsweise überträgt das Strukturdiagramm in ein ablochfähiges Format mit folgendem Aufbau:

Formularbeschreibung:

```
Spalte 01 - 02   Blatt
       03 - 05   Zeile
       06 - 06   Kartenart (muss 'S' sein)
       07 - 22   Programmstruktur
       23 - 28   Name des Datenkanals
       32 - 37   Name des HL1-Bausteins
       40 - 74   Text
       75 - 80   Name des Programms
```

Übertragen in dieses Format wird Bild 9 dann wie folgt dargestellt:

```
    SI--------------          OPH     OFFENE POSTEN HAUPTPROGRAMM
    S I------------- KEYS     OPK     OFFENE POSTEN KOPFDATEN
```

Die Eintragung 'I' in Spalte 7 der ersten Karte gibt an, dass der Baustein 'OPH' die höchste Hierarchiestufe der Baugruppe belegt. Die folgenden Minuszeichen dienen der graphischen Darstellung des Strukturdiagramms. Die Eintragung 'I' in Spalte 8 der zweiten Karte gibt an , dass der Baustein 'OPK' eine Stufe unterhalb von 'OPH' ansetzt.

Die Eintragung 'KEYS' bedeutet, dass zwischen 'OPK' und der nächsthöheren Stufe die im Kanal 'KEYS' beschriebenen Daten ausgetauscht werden.

Für das in Bild 8 beschriebene komplette Strukturdiagramm ergibt sich damit folgendes Bild:

```
     Struktur        Kanal    Phase   Text
_____________________________________________________________

    SI--------------          OPH     offene Posten Hauptprogramm
    S I------------- KEYS     OPK     OP Kopfdaten
    S II------------ KEYS     OPM     OP Bildschirmmaske
    S II------------ STDAT    OPD     OP Stammdaten
    S III----------- STSGM    OPL     OP Stammdaten lesen
    S III----------- AKDAT    OPA     OP Abrechnungskreis
    S III----------- LIEFBD   OPT     OP Text Lieferbedingungen
    S II------------ STDAT    OPB     OP Bildschirmausgabe Stammdaten
    S I------------- KEYS     OPS     OP Seite anzeigen
    S  I------------ OPZEIL   OPZ     OP Zeile aufbereiten
    S   I----------- OPDAT    OPO     OP Daten lesen
```

Das Diagramm ist wie folgt zu lesen:

Der Baustein 'OPH' bildet die oberste Hierarchiestufe und steht daher für die gesamte Baugruppe 'offene Posten-Anzeige'. Der Anwendungsprogrammierer kann diesen Baustein aus einer Anwendung aufrufen, ohne dabei die interne Struktur der Baugruppe zu kennen.

Die Bausteine 'OPK' und 'OPS' liegen in der zweiten Hierarchiestufe und sind unmittelbar mit dem Baustein 'OPH' verbunden. Dies wird ausgedrückt durch den Schnittpunkt der waagerechten Linie bei 'OPH' mit der senkrechten Linie (I in Spalte 8), die die zweite Hierarchie-Ebene symbolisiert.

Zwischen den Bausteinen 'OPK' und 'OPS' und dem Baustein 'OPH' werden dabei die im Kanal 'KEYS' beschriebenen Daten ausgetauscht.

Die nächste senkrechte Linie (Spalte 9) symbolisiert die dritte Hierarchie-Ebene. Dabei werden die Bausteine 'OPM', 'OPD' und 'OPB' dem Baustein 'OPK' in der zweiten Ebene und der Baustein 'OPZ' dem Baustein 'OPS' in der zweiten Ebene zugeordnet. Dies wird ebenfalls symbolisiert durch die Schnittpunkte mit den beiden waagerechten Linien bei 'OPK' und 'OPS'.

Die weitere Untergliederung erfolgt sinngemäß. Ist die Struktur der Baugruppe 'OPH' einmal beschrieben, so kann die Baugruppe in einer uebergeordneten Struktur wie ein Einzelbaustein 'OPH' eingesetzt werden.

Private HL1-Bibliotheken. 5080

HL1-Bausteine werden nicht in die Programmtabellen der TP-Steuerprogramme eingetragen, weil sie von der HL1-Methodenbank gesteuert und verwaltet werden.

VSE-Benutzer können dennoch ihre HL1-Bausteine von CICS verwalten lassen. Hieraus ergeben sich Vorteile während der Testphase oder wenn es sich um besonders große Bausteine handelt. CICS versucht bei einem Newcopy, diesen Platz wieder zu verwenden, falls das HL1-Modul nicht über eine 2K-Grenze hinaus verändert wurde.

Jedoch ist eine Eintragung dieser Bausteine in einer HL1-Programm-Tabelle erforderlich. Die Eintragung erfordert je Baustein 4 Bytes und wird über den HL1-Tabellen-Generator (HL1HTG) generiert.

Die Größe der Programmtabelle ist auf 4K Bytes begrenzt. Eine HL1-Programmtabelle kann folglich nicht mehr als 1024 Eintragungen enthalten.

Die Reihenfolge der Eintragungen ist zwingend für die Verarbeitung. Eine spätere Änderung der Reihenfolge erfordert die Neu-Umwandlung aller Programme, die Bausteine aus dieser Tabelle aufrufen. Es ist daher sinnvoll, bereits vor der Installation eine Schaetzung des spaeteren Bedarfs vorzunehmen. Werden insgesamt mehr als 1000 Bausteine für den End-Ausbau erwartet, so wird vorgeschlagen, bereits bei der Installation für die einzelnen Anwendungen private Bibliotheken anzulegen. Diese Bibliotheken bieten außerdem eine erhöhte Sicherheit bei Programmtests und bei der Programm-Wartung, da eventuelle Fehler auf die eigene Anwendung beschränkt bleiben.

Bei der Einrichtung privater Bibliotheken werden die Programm-Bausteine nach ihrer Verwendung sortiert. Sind die Bausteine eindeutig anwendungsbezogen, z.B. Skonto rechnen, Artikeldaten lesen usw., so werden diese Bausteine in die private Bibliothek für diese Anwendung, z. B. Buchhaltung, Auftragsabwicklung usw., eingetragen. Sind die Bausteine anwendungsneutral, z.B. Programmbeschreibung, Bildschirmkopf usw., so werden diese Bausteine in die Tabelle für die allgemeine Bibliothek eingetragen, die zu jeder privaten Bibliothek hinzugefügt wird.

Private Bibliotheken werden durch einen einstelligen Suffix gekennzeichnet. Eine Muster-Tabelle für eine allgemeine Bibliothek und eine private Bibliothek mit dem Namen 'A' gehört zum Lieferumfang von HL1. Die beschriebenen HL1-Bibliotheken sind für eine allgemeine und maximal 23 private Bibliotheken ausgelegt.

Bei der Verwendung von Programmtabellen für private Bibliotheken muss bei einer HL1-Umwandlung die Spalte 22 der H-Karte unter Umständen eine entsprechende Eintragung enthalten. Siehe Abschnitt 9760.

Module, die zum Lieferumfang des HL1 gehören (z.B. die standardisierten Schnittstellen zu anderen Lattwein-Produkten) liegen in der privaten Library H.

HL1-Datasets 5085

HL1-Datasets sind eine Teilmenge der HL1-Bausteine. Grundsätzlich unterscheiden sie sich nicht von anderen HL1-Modulen.

HL1-Datasets eignen sich insbesondere zur dateiunabhängigen Programmierung, da sie mit gewohnten Dateioperationen ohne zusätzliche Programmierung zur Datei- oder Datenbankverarbeitung angezogen werden.

Bei HL1-Datasets sind folgende Punkte zu beachten:

- HL1-Datasets müssen nach der Dataset-Logik arbeiten, also in Spalte 34 der H-Karte ein 'D' oder 'S' haben.

- Die F-Karte für ein HL1-Dataset erhält für die Einheit (Spalten 40 bis 46) ein 'HL1' oder 'HL1DS'.

- Die Datenübergabe vom rufenden zum gerufenen Modul und zurück kann über den HL1-Datenkanal sehr einfach abgewickelt werden. HL1-Datenkanäle enthalten nur alpanumerische und gepackte numerische Werte. Datenkanäle sollten keine Lücken enthalten und im Data Dictionary beschrieben werden. Sie werden im rufenden Programm im Input beschrieben, im gerufenen in den ersten Stellen der D-Karten.

- Der Datenkanal muss das vierstellige Alphafeld CPGHIC (HL1 Interface Control) enthalten, das die Dataset-Verarbeitung steuert.

- Alle Dateioperationen sind für HL1-Datasets unterstützt.

- Der Programmierer, der mit einem HL1-Dataset arbeitet, benötigt lediglich I-Karten und Rechenbestimmungen. Dateien werden mit den Operationen UPDAT, WRITE und DELET geändert.

- Der Eintrag eines Keys in Faktor 1 der Dateioperation ist nicht erforderlich, da er über den Datenkanal definiert wird. Aus Kompatibilitätsgründen kann jedoch in Faktor 1 ein Keyfeld eingetragen werden.In diesem Fall muss jedoch das CPG-interne Keyfeld für den Dateizugriff im Datenkanal definiert werden.

Eine ausführliche Beschreibung mit Beispielprogrammen finden Sie auf den nächsten Seiten.

Programmierung mit HL1-Datasets 5090

Das folgende Beispiel zeigt ein HL1-Hauptprogramm mit den möglichen Dateizugriffen über ein HL1-Dataset.

```
    FSAMPLE  U   F     118 14         HL1DS

    ISAMPLE  HS
    I                                      15  93 FELD
    I                                     101 114 KEY
    I                                     115 118 CPGHIC

    C                    READ SAMPLE
    C                    CHAINSAMPLE
```

```
C           CPGFRC    IF   'NF'
C                     :
C                     END
C                     RNDOMSAMPLE
C                     SETLLSAMPLE
C                     OPEN SAMPLE
C                     CLOSESAMPLE
C                     CHECKSAMPLE
C                     DELETSAMPLE
C                     CHAINSAMPLE             C
C                     CHAINSAMPLE             P
C                     CHAINSAMPLE             U
C                     READBSAMPLE
C                     READBSAMPLE
C                     WRITESAMPLE
C                     UPDATSAMPLE
```

Das Hauptprogramm enthält in der F-Karte für das Dataset bei Einheit den Eintrag HL1 (Spalte 40-43).

Für die Übertragung der Felder ist ein Datenkanal erforderlich. Dieser entspricht den I-Karten bei Datei-Verarbeitung. Sein Name muss gleich dem Phasennamen des HL1-Dataset-Moduls sein. In den Spalten 15-16 der Input-Satzbestimmung wird HS (für HL1-Struktur) angegeben.

Der Datenkanal beschreibt die Felder des Datasets. Zusätzlich muss er das 4-stellige HL1-Interface Controlfeld CPGHIC enthalten.

Datenkanäle sollten im Data Dictionary beschrieben werden !

Nicht unterstützt ist der Eintrag eines Labels im Ergebnisfeld bei einer CHAIN-Operation, sowie ** bei der Ergebnisbezugszahl bei einer CHAIN-Operation.

Schalter bei CHAIN, SETLL, OPEN und CLOSE sind ebenfalls über das HL1-Dataset unterstützt. Die entsprechenden Return-Codes werden aber auch in dem Feld CPGFRC zurückgegeben.

Das folgende Beispiel zeigt das HL1-Dataset-Modul für einen 1:1-Datei-zugriff:

```
H                           D                          SAMPLE

FCPGWRK  U   F     100 14         KSDS

D        SATZ         100         DATENSATZ
D        KEY           14         SCHLÜSSEL
D        CPGHIC      0  4         HL1 INTERFACE CONTROLFELD
D        OC             2         HL1 INTERFACE OPCODE
D        RC             2         HL1 INTERFACE RTCODE

ICPGWRK  KF
I                                       1 100 SATZ

C*------------------------------------------ OPEN, CLOSE + CHECK
C           OC        IF   'O'
C                     OPEN CPGWRK                98
C                     END
C           OC        IF   'C'
C                     CLOSECPGWRK                98
```

```
C                   END
C           OC      IF   'Z'
C                   CHECKCPGWRK              99
C                   END
C*------------------------------------------ READ, READB + RNDOM
C           OC      IF   'R'
C           KEY     READ CPGWRK
C                   MOVEL'S'       RC
C                   END
C           OC      IF   'B'
C           KEY     READBCPGWRK
C                   MOVEL'S'       RC
C                   END
C           OC      IF   'D'
C                   RNDOMCPGWRK
C                   MOVEL' '       RC
C                   END
C*------------------------------------------------ CHAIN + SETLL
C           OC      IF   'G '
C           KEY     CHAINCPGWRK    EXIT       99
C                   END
C           OC      IF   'GC'
C           KEY     CHAINCPGWRK    EXIT      C99
C                   END
C           OC      IF   'GP'
C           KEY     CHAINCPGWRK    EXIT      P99
C                   MOVEL'U'       RC
C                   END

C           OC      IF   'GU'
C           KEY     CHAINCPGWRK    EXIT      U99
C                   MOVEL'U'       RC
C                   END
C           OC      IF   'S '

C           KEY     SETLLCPGWRK              99
C                   END
C*------------------------------------------ UPDAT, WRITE + DELET
C           OC      IF   'U'
C           KEY     CHAINCPGWRK    EXIT      U**
C                   GETHS
C                   MOVELKEY       SATZ
C           KEY     UPDATCPGWRK    SATZ
C                   END
C           OC      IF   'N'
C                   MOVELKEY       SATZ
C           KEY     WRITECPGWRK    SATZ
C                   END
C           OC      IF   'L'
C           KEY     DELETCPGWRK
C                   END
C           EXIT    TAG
C*------------------------------------------------------ SCHALTER
C   DR              MOVE 'D'       RC             DUPL RECORD
C   EF              MOVE 'E'       RC             END OF FILE
C   98              MOVE 'F'       RC             FILE  ERROR
C   99              MOVE 'G'       RC             GET N FOUND
```

Die F-Karten enthalten nur die benötigten Dateien.

Die D-Karten müssen mit dem Datenkanal aus dem übergeordneten Programm übereinstimmen. Sie müssen zusammenhängend am Anfang der D-Karten liegen. Es dürfen jedoch Felder anders benannt sein, ent-

scheidend ist die Position im Datenkanal. CPGHIC, das vierstellige Interface-Control-Feld, muss im Datenkanal definiert sein (im Beispiel spiel durch die Felder OC und RC).

Die Art des Dateizugriffs wird durch die ersten 2 Stellen des HL1-Interface-Controlfeldes übermittelt. Die 3. Stelle wird von CPG intern benutzt.

Die 4. Stelle wird für den Returncode benutzt, z.B. EF, DR, NOT FOUND oder Dateifehler OPEN-CLOSE.

Die Benutzung des HL1-Interface-Controlfeldes CPGHIC ist in der folgenden Tabelle beschrieben.

CPGHIC Byte 1: Operationscode:	'B' READB 'C' CLOSE 'D' RNDOM 'E' EXCPT 'G' CHAIN 'L' DELET 'N' WRITE 'O' OPEN 'R' READ 'S' SETLL 'U' UPDAT 'Z' CHECK
CPGHIC Byte 2: Opcode Erweiterung:	wie Spalte 53 bei Calculations ' ' bei UPDATE in Ausgabe 'A' bei EADD in Ausgabe 'D' bei EDEL in Ausgabe
CPGHIC Byte 3: intern benutzt	
CPGHIC Byte 4: Returncode/Schalter:	ist im HL1-Dataset zu definieren, dabei gilt: 'D' Duplicate Record, DR wird gesetzt. 'E' End of File, EF wird gesetzt und Schalter EQ (58-59),wenn definiert. 'F' File Error bei Open und Close, Schalter LT (56-57) wird gesetzt. 'G' Satz nicht gefunden bei Get Schalter GT (54-55) wird gesetzt.

C GETHS

GETHS erlaubt ein nochmaliges Übertragen des Datenkanals.

Dies ist jedoch nur erforderlich, wenn eine UPDAT-Operation durch das übergeordnete Programm durchgeführt wird, und nicht alle Felder des Satzes im Datenkanal definiert wurden.

Das folgende Beispiel enthält ein Dataset im CPG-Format, das strukturiert programmiert wurde:

```
- OPTIONS DATASET PHASE SAMPLE.
-  FILE CPGWRK.
- -D.
-     DEFINE CPGWRK TYPE DS.      * Datenkanal, enthält KEY,
-                                 *             SATZ und CPGHIC
-     ORG CPGHIC.                 * Redefinition von CPGHIC
-        OC  2.                   * HL1 Interface Opcode
-        RC  2.                   * HL1 Interface Return-Code
-        ORG.                     * Ende der Redefinition
- -I.
-     FILE CPGWRK
-        1 100 SATZ
- -C.
-     EVALUATE
-       WHEN OC = 'B '.           * READ-BACK -------------------*
-         KEY READ-BACK CPGWRK
-         IF CPGFRC = 'EF'.       * End of File
-            MOVE 'E' TO RC
-         ENDIF
-       WHEN OC = 'C '.           * CLOSE -----------------------*
-         CLOSE CPGWRK
-         IF CPGFRC >< '  '.      * Not found oder not closed
-            MOVE 'F' TO RC
-         ENDIF
-       WHEN OC = 'D '.           * RANDOM ----------------------*
-         RANDOM CPGWRK
-       WHEN OC = 'G'.            * CHAIN -----------------------*
-         IF OC = 'G '.           * CHAIN
-           KEY CHAIN CPGWRK
-         ELSE
-           IF OC = 'GU'.         * CHAIN für Update
-             KEY CHAIN CPGWRK UPDATE
-           ELSE
-             IF OC = 'GC'.       * CHAIN Check (ohne Einlesen)
-               KEY CHAIN CPGWRK CHECK
-             ELSE.               * CHAIN Check für Update
-               KEY CHAIN CPGWRK P
-             END
-           END
-         END
-         IF CPGFRC = 'NF'.       * Not found
-           MOVE 'G' TO RC
-         END
-       WHEN OC = 'L '.           * DELETE (Löschen) ------------*
-         KEY DELET CPGWRK
-       WHEN OC = 'N '.           * WRITE (New Record) ----------*
-         SATZ = KEY
-         KEY WRITE CPGWRK SATZ
-         IF CPGFRC >< '  '.      * Duplicate Record
-           MOVE 'D' TO RC
-         END
-       WHEN OC = 'O '.           * OPEN ------------------------*
-         OPEN CPGWRK
-         IF CPGFRC >< '  '.      * Not found oder not open
-           MOVE 'F' TO RC
-         END
-       WHEN OC = 'R '.           * READ ------------------------*
-         KEY READ CPGWRK
-         IF CPGFRC = 'EF'.       * End of File
```

```
-           MOVE 'E' TO RC
-         END
-       WHEN OC = 'S '.           * SET LOWER LIMIT -------------*
-         KEY SETLL CPGWRK
-       WHEN OC = 'U '.           * UPDATE ----------------------*
-         SATZ = KEY.
-         KEY UPDAT CPGWRK SATZ.
-       WHEN OC = 'Z '.           * CHECK -----------------------*
-         CHECK CPGWRK
-         IF CPGFRC =  '  '.      * Not found
-           MOVE 'G' TO RC
-         END
-     END-EVALUATE
```

Hinweise zur Programmierung

. Das Beispiel arbeitet auf einer VSAM-Datei mit fester Satzlänge von weniger als 256 Bytes Länge. Deshalb ist hier die Dateiausgabe mit den Befehlen UPDAT, WRITE und DELET unterstützt. Bei größeren Satzlängen oder bei VSAM-Dateien mit variabler Satzlänge müßten die entsprechenden Statements durch EXCPT-Ausgaben ersetzt werden.

  Hier ist natürlich der gesamte CPG-Funktionsumfang unterstützt, also auch der Zugriff auf eine beliebige andere Datenbasis. Der Programmierer des Anwendungsprogramms kann also auch bei anderer Datenbasis mit den gewohnte Lese- und Schreibbefehlen arbeiten.

. Datenkanäle sollten im Data Dictionary beschrieben werden, weil sie bei der Programmierung zweimal benötigt werden: Im Input des rufenden Programms und in der Datendefinition des gerufenen Programms.

  Bei 1:1-Datasets ist es oft der Fall, dass die Dataset-Struktur mit der Dateistruktur identisch ist, dass sie aber zusätzlich das CPGHIC-Feld enthalten muss. In diesem Fall kann an allen drei Stellen (Dataset-Kanal-Input im rufenden Programm, Kanalbeschreibung im gerufenen Modul und Datei-Inputbestimmungen im gerufenen Programm) mit der gleichen Data-Dictionary-Struktur gearbeitet werden. CPG generiert das CPGHIC-Feld dabei nicht im Input für die VSAM-Datei.

. Die maximale Satzlänge eines Datasets beträgt knapp 4000 Bytes.

Ausführung 6000

CPG-Methodenbank 6010

Die CPG-Methodenbank ist ein Programm, das alle häufig benutzten Verarbeitungsroutinen, darunter auch alle TP- und IOCS-Makros in sich vereint und daduch verhindert, dass gleichartige Programmteile mehrfach in der TP-Partition liegen und damit das System unnötig belasten.

Dies bedeutet, dass CPG-Programme der Standardversion nur arbeiten können, wenn vorher die Methodenbank geladen worden ist.

Das Laden der Methodenbank erfolgt daher sinnvollerweise unmittelbar nach dem Laden des TP-Steuerprogramms, soweit die Methodenbank nicht bereits in diesem enthalten ist.

Einheiten-Unabhängigkeit 6020

Die Ein- und Ausgabe des CPG über die Einheiten Leser, Drucker und Stanzer ist einheitenunabhängig, d.h. die logischen Einheiten können über eine ASSGN-Karte zur Job-Controlzeit einer beliebigen physischen Einheit zugeordnet (assigned) werden.

Job-Control-Steuerkarten 6200

1. CPG-Programme

```
// JOB XXXXXXXX
// EXEC CPG

   ...  CPG-PROGRAMM

/*
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT (VS,VSE)oder IJSYSIN // EXEC DFHLINK (nicht VS)
/&
```

Eine eventuelle SIZE-Angabe in der EXEC-Karte sollte bei VSE mindestens 256K enthalten.

Die I/O-Bereiche werden bei der Umwandlung wie folgt benötigt

```
CPGRDR(SYSIPT)  ---> CPGWRKA  (CPG-DECK)
IJSYS03(CPGWRKA)---> IJSYS04  (ASS-DECK)
IJSYSIN(IJSYS04)
```

1a. Beispiel VSE

```
*-------------------------------------------------------------*
*  Beispiel für eine CPG-Umwandlung mit der Prozedur $SYSIN   *
*  ---------------------------------------------------------*
// EXEC CPG
      H                                  CPG-UMWANDLUNG   XXXXX
      C*
      C*         CPG-PROGRAMM
      C*
/*
// EXEC PROC=$$SYSIN
// EXEC LNKEDT
/*
/&


// JOB CATAL   PROCEDURES
*-------------------------------------------------------------*
*  Beispiel Arbeitsbereiche zum Umwandeln von CPG-Programmen  *
*  in mehreren Partitions in der VSE-Umgebung.                *
*  Die EXTENT Anweisungen sind ihrer Konstellation anzupassen.*
*  IJSYSIN kann nicht durch den VSAM Space Manager verwaltet  *
*  werden.                                                    *
*-------------------------------------------------------------*
// LIBDEF PHASE,SEARCH=(PRD1.BASE,IJSYSRES.SYSLIB)
// EXEC LIBR,PARM='ACC S=IJSYSRS.SYSLIB'
CATALOG $F4LBL.PROC     REPLACE=YES    DATA=YES
// OPTION PARSTD
// DLBL IJSYS03,'F4.WORK.THREE',0,SD,,CISIZE=8192
// EXTENT SYS003,PRD002,1,0,780000,10000
// DLBL CPGWRKA,'F4.WORK.THREE',0,SD,,CISIZE=8192
// EXTENT SYS003,PRD002,1,0,780000,10000
// DLBL IJSYS04,'F4.WORK.FOUR',0,SD,,CISIZE=8192
// EXTENT SYS004,PRD002,1,0,770000,10000
// DLBL IJSYSIN,'F4.WORK.FOUR',0,SD,,CISIZE=8192
// EXTENT SYSIN,PRD002,1,0,770000,10000
*
*  USW.
*
/+
CATALOG $BGLBL.PROC     R=YES     DATA=YES
// OPTION PARSTD=BG
// DLBL IJSYS03,'BG.WORK.THREE',0,SD,,CISIZE=8192
// EXTENT SYS003,SYSWK1,1,0,538836,011076
// DLBL IJSYS04,'BG.WORK.FOUR',0,SD,,CISIZE=8192
// EXTENT SYS004,SYSWK1,1,0,527760,011076
// DLBL CPGWRKA,'BG.WORK.THREE',0,SD,,CISIZE=8192
// EXTENT SYS003,SYSWK1,1,0,538836,011076
// DLBL IJSYSIN,'BG.WORK.FOUR',0,SD,,CISIZE=8192
// EXTENT SYSIN,SYSWK1,1,0,527760,011076
*
*  USW.
*
/+
```

```
CATALOG $0SYSIN.PROC R=YES
ASSGN SYSIN,C01
/+
CATALOG $4SYSIN.PROC R=YES
ASSGN SYSIN,C02
/+
/*
/&
```

2. CPG-Programme mit Autoreport-Funktionen (z.B. */Copy)

Für CPG-Programme, die /Copy-Anweisungen - ähnlich dem RPG-Autoreport - enthalten, ist ein Vorlauf mit dem CPG-Prepro-Zessor CPGPP erforderlich. Außerdem muss die Übergabe an CPG über eine Procedure erfolgen.

Folgende Job-Control-Karten sind erforderlich:

```
// JOB XXXXXXXX
// EXEC CPGPP                 CPG Deck -> SYS003

   ... CPG-PROGRAMM           Punch    -> SYS004

/*
// EXEC PROC=CPG
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT
ASSGN SYSIPT,X'CUU'           Sysin Adresse
/&
CLOSE SYSIPT,X'01C'           Leser Adresse
/&
```

Die Procedure CPG ist für VSE- Anwender wie folgt zu katalogisieren:

```
   CATALP CPG,EOP=/+
x  // DLBL IJSYSPH,'F4.WORK.TWO',0,SD
x  // EXTENT SYSPCH,VOLLBL,1,0,81840,7440        (Beispiel)
   ASSGN SYSPCH,224
   // DLBL IJSYSIN,'F4.WORK.FOUR',0,SD
   // EXTENT SYSIPT,VOLLBL,1,0
   ASSGN SYSIPT,224
   // EXEC LIBR,PARM='ACC S=SP2USR.USRLBL'
   CLOSE SYSIPT,READER
   CLOSE SYSPCH,PUNCH
   /*
   // EXEC CPGP2
   // DLBL IJSYS02,'F4.WORK.TWO',0,SD
   // EXTENT SYS002,VOLLBL,1,0,81840,7440
   // DLBL IJSYSIN,'F4.WORK.FOUR',0,SD
   // EXTENT SYSIPT,VOL224,1,0
   ASSGN SYSIPT,224
   // EXEC CPG
   CLOSE SYSIPT,READER
   /+
```

3. CPG3-Programme mit SQL-Befehlen (Beispiel)

```
// JOB CPGSQL
// LIBDEF PROC,SEARCH=(PRD2.SQL220)
// EXEC PROC=ARISLIBP   *-- SQL/DS PRODUCTION LIBRARY ID PROC
// ON $ABEND GOTO REASS
// EXEC CPGPREP,PARM='USERID=SQLDBA/SQLDBAPW'
   :
   :   CPG-Programm mit SQL-Befehlen
   :
/*
// IF $RC NE 0 THEN
// GOTO ENDE
*  STEP HL1
// LIBDEF PHASE,CATALOG=SP4U.ULIBL
// DLBL IJSYSIN,'F4.WORK.04',0,SD,,CISIZE=8192
// EXTENT SYSIPT,PRD201,1,0,46000,4000
ASSGN SYSIPT,122
// EXEC HL1
/*
// IF $RC NE 0 THEN
// GOTO REASS
*  STEP ASSEMBLER
CLOSE SYSIPT,READER
ASSGN SYSIN,122
*  STEP LNKEDT
// EXEC LNKEDT
/*
// IF $RC EQ 0 THEN
// GOTO ENDE
/. REASS
CLOSE SYSIPT,READER
/. ENDE
/&
```

Programmtest und Fehlerbeseitigung 6250

Alle Programme sollten gründlich getestet werden, bevor sie produktiv eingesetzt werden. CPG-Beinhaltet mehrere Funktionen, die das Testen von Online-Transaktionen erleichtern.

Es können Referenz-Karten zur Überprüfung der spaltengebundenen Eintragungen im CPG bezogen werden. Diese Referenzkarten enthalten zudem Tabellen der Operationscodes, Attribute und Aufbereitungsschlüssel.

Übersetzung 6260

Bevor ein Programm getestet werden kann, muss es vom CPG-Übersetzer übersetzt worden sein. Der Ausdruck des Übersetzers darf keine Syntaxfehler aufweisen.

Aus der Übersetzung des CPG-Quellencodes wird ein Assemblerprogramm erstellt. Es ist zu prüfen,dass am Ende der Assembler-Liste die Nachricht 'NO ERROR FOUND' ausgedruckt ist.

Werden in dem Assemblerprogramm Fehler festgestellt, so können diese auch von Programmierern, die Assembler nicht kennen, problemlos korrigiert werden. Wird eine vollständige Assembler-Liste vom Programm erstellt (X in Spalte 11 der H-Karte), so werden die fehlerhaften Anweisungen in der Umwandlungsliste deutlich markiert. Der CPG-Quellencode erscheint als Kommentar im Assemblerprogramm. Die Kommentarzeilen werden rechts über dem daraus entstehenden Assemblercode angedruckt.

Somit kann ohne weiteres festgestellt werden, welche CPG-Quellenanweisung zu dem ungültigen Assemblercode geführt hat. Der Fehler wird durch Korrektur des CPG-Quellencodes korrigiert.

Sobald eine völlig fehlerfreie Kompilierung erreicht ist, kann der Programmierer das Programm austesten.

Programmaufruf am Bildschirm 6270

Bevor ein Onlineprogramm ausgeführt werden kann, muss ein Eintrag in der Programmtabelle und in der Transaktionstabelle vorgenommen werden. Folgendes Beispiel zeigt die Eintragung für CPG-Programme in der CICS PPT/PCT:

```
DFHPPT TYPE=ENTRY,PROGRAM=PROG01

DFHPCT TYPE=ENTRY,TRANSID=PR01,TWASIZE=3840,          +
      SPURGE=YES,TPURGE=YES,PROGRAM=PROG01
```

Werden in einem Onlineprogramm Dateien verarbeitet, so ist ein Eintrag in der Dateitabelle erforderlich.
Folgendes Beispiel zeigt eine Eintragung in der CICSFCT für eine VSAM-KSDS-Datei:

```
DFHFCT TYPE=DATASET,DATASET=DNAME,                    +
```

```
ACCMETH=(VSAM,KSDS),                              +
SERVREQ=(GET,PUT,UPDATE,NEWRC,DELETE,BROWSE),     +
RECFORM=(FIXED,BLOCKED),STRNO=3,BUFND=3,          +
BUFNI=2,OPEN=INITIAL
```

Initialisierung 6280

Für jede von einem oder mehreren CPG-Programmen zu verarbeitende Transaktion ist ein Bereich für Variablen usw. erforderlich, die in dem/den CPG-Programm(en) verwendet werden. Dieser Bereich ist die Tranasction Work Area. In der TWA werden die 100 Programmbezugszahlen, sämtliche von den CPG-Programmen benutzten Felder, und die von CPG benötigten Steuerinformationen gespeichert.

Bei CICS-Systemen wird dieser Bereich für eine Transaktion innerhalb der PCT definiert. Die zu definierende Größe kann der CPG-Umwandlungsliste entnommen werden.

Beim Aufruf einer Transaktion sind alle 100 Bezugszahlen 'AUS' gesetzt. Wenn in einem Programm nichts anderes angegeben ist, initialisieren alle innerhalb der Transaktion aufgerufenen CPG-Programme die alphanumerischen Felder in der TWA auf Leerzeichen (Blank) und die numerischen Felder auf Null.

Wenn ein Online-Programm aufgerufen wird, kann der TP-Monitor dem Programm einen Datenbereich übergeben, der die zuletzt vom Bildschirm gelesenen Daten enthält (bei taskorientierter Programmierweise).

Wird dem CPG-Programm ein solcher Bereich übergeben, so versucht es, den Datenstrom von dem Terminal unter Verwendung einer seiner Eingabebestimmungen für den Bildschirm zu interpretieren. Enthält das Programm mehrere Eingabebestimmungsgruppen für Bildschirme, so wird die zuletzt angegebene Gruppe herangezogen,um die Daten zu interpretieren. Stimmen die Daten im Eingabebereich mit den in den Eingabebestimmungen angegebenen Positionen überein, so werden die entsprechenden Felder in der TWA geändert.

Sind keine modifizierten Felder vom Bildschirm verfügbar, so wird dieser Schritt übergangen. Die automatische Verarbeitung von Bildschirmeingaben, die einem Programm bei Aufruf übergeben werden, wird im Zusammenhang mit den Operationen EXITP und EXITT für die Transaktionsverknüpfung angewandt. Nach diesen Schritten beginnt der Programmablauf mit den ersten Anweisungen in den Rechenbestimmungen.

Beenden eines Onlineprogramms 6290

Ein CPG-Onlineprogramm endet, wenn keine weiteren Rechenbestimmungen vorhanden sind oder wenn der Terminalbediener während des Programmablaufs die Clear-Taste drückt. (Es sei denn ein 'I' wurde in Spalte 53 der READ-Operation oder in Stelle 12 der CPGURSIT eingetragen).

Bei Beendigung eines CPG-Programms werden die folgenden Operationen ausgeführt:

a) Eine Ausgabeoperation für ein Terminal wird ausgeführt. Dadurch wird die Tastatur freigegeben, sofern sie zuvor gesperrt war, und es werden alle Anzeiger für modifizierte Daten (MDTs) auf dem Bildschirm gelöscht.

b) Die Steuerung wird an das nächst höhere Programm bzw. zum TP Monitor zurückgegeben.

Beenden eines Batchprogramms 6293

Ein Batchprogramm endet, wenn keine weiteren Rechenbestimmungen vorhanden sind. Die Steuerung wird an das Betriebssystem zurückgegeben, es sei denn, in Spalte 51 der H-Karte wurde ein 'S' eingetragen. Der Eintrag 'S' bewirkt, dass das Programm als Unterprogramm, z.B. zu DL1 abläuft und bei Programmende die Steuerung an das übergeordnete Programm zurückgegeben wird.

Übergehen der normalen Programmbeendigung 6295

Der obengenannte Schritt a) bei der Beendigung eines CPG-Programms kann übergangen werden, wenn die folgende Anweisung als letzte Operation für das CPG-Programm codiert wird:

```
C                     GOTO CPGEND
```

CPGEND ist eine CPG-interne Marke, die einen sofortigen Rücksprung zur Nächst höheren Stufe bewirkt. Dabei wird keine Ausgabeoperation für den Bildschirm ausgeführt.

Dies geschieht automatisch, wenn keine READ-Operation für ein Terminal im Programm ist.

In HL1-Modulen, in denen eine Ausgabe auf einen Online-Drucker durchgeführt wurde, kann mit

```
C                     GOTO CPGEMD
```

zur nächst höheren Stufe zurück verzweigt werden, ohne dass automatisch ein Vorschub erzeugt wird. Der Programmierer hat hierbei allerdings dafür zu sorgen, dass der Line-Counter (CPGLCT) ordnungsgemäß zwischen den Programmen übergeben wird.

HL1-Batch-Programme 6300

Mit HL1 können neben Online-Programmen auch Batch-Programme erstellt werden.

Die einzige grundsätzliche Änderung hierfür besteht darin, dass in Spalte 47 der H-Karte ein 'B' (für Batch-Version) eingetragen wird.

Standardmäßig sind Batch-Programme für eine Programmgröße von bis zu 20K und eine TCA-Größe bis zu 8K vorgesehen. Durch 'S' oder 'T' in Spalte 32 der H-Karte können jedoch erheblich größere Programme erstellt werden.

Die Batch-Programme arbeiten genauso wie die meisten CPG- und HL1-Onlineprogramme mit einer Methodenbank. Die Batch-Methodenbank ist nicht Bestandteil des Programms, sondern wird zur Ausführungszeit zum Programm hinzugeladen.

Ebenso werden die Datenbereiche CSA, TCA und PIW (Interface Bereiche) beim Programmstart angefordert. Hiermit werden die HL1-Batch-Programme erheblich kleiner als vergleichbare RPG- oder COBOL-Programme. Dadurch wird der Platzbedarf in der Core Image Library wesentlich reduziert.

Zur Zeit sind im Batch folgende Dateiarten unterstützt:

READER für die Karteneingabe von SYSIPT.

PRINTER für die Druckausgabe auf SYSLST. Im Interface sind für die Druckausgabe zwei unterschiedliche DTFs verfügbar:

a) Standardmäßig DTFDI (Device independant), welches jedoch nur die Ausgabe bis zu 120 Stellen zulässt. Die Vorschubsteuerung erfolgt in diesem Fall über die L-Karte.

b) Außerdem DTFPR, welches eine Ausgabe bis zu 132 Stellen zulässt. Dieses DTF wird dann benutzt, wenn der Dateiname mit 'PRIN' beginnt.

Es können beliebig viele Drucker angegeben werden, die über die SYS-Nummer unterschieden werden.

PUNCHER für die Stanzausgabe auf SYSPCH.

Ein 'PUNCHER' wird jetzt erst bei der ersten Ausgabe eröffnet. Damit ist das Problem der 'leeren' Punch-Queue Einträge bei einigen VSE Systemen mit Query Batch gelöst.

KSDS für VSAM-Indexdateien mit fester oder variabler Satzlänge.

Diese Angabe kann auch für VSAM-PATH-Dateien benutzt werden, wenn der Zugriff über einen Alternativindex erfolgt. Hierbei ist in der F-Karte der Pfadname einzutragen und die Keylänge vom Alternativindex anzugeben.

ESDS für sequentielle VSAM-Dateien mit fester oder variabler Satzlänge.

RRDS für VSAM-Satzadressdateien (haben immer feste Satzlänge)

DISK für sequentielle Plattendateien. Die Schlüssellänge in der F-Karte muss leer bleiben.

TAPE für Banddateien.

DL1 für Datenbankzugriffe zu DL1.

HL1 für die Verarbeitung von HL1-Datasets.

STORAGE für die Verarbeitung von Temporary Storage. Es sind sowohl Einzelsätze als auch Queues unterstützt.

TABLE für die Verarbeitung von Tabellen.

Außerdem kann jetzt EDIT und SELCT CPGTCT aus Kompatibilitätsgründen zu CICS Programmen und Modulen verwendet werden.

Die Dateien werden bei Programmstart automatisch eröffnet und bei Programmende geschlossen, sofern nicht in der F-Karte 'NO' angegeben ist und im Programm die Befehle 'OPEN' und 'CLOSE' verwendet werden.

Bei dem Leser ist zu beachten, dass der End-of-File Schalter 'EF' geprüft wird, ansonsten würde ein weiteres Read hinter End-of-File,also hinter der '/* ' oder '/& ' Karte zu einem Systemfehler führen.

Ist ein Drucker angegeben, so kann durch den Druckernamen das benutzte DTF ausgewählt werden. Kanalvorschübe werden bei DTFDI über L-Karte, bei DTFPR über FCB gesteuert. Ist ein Kanal nicht definiert, dann bricht das Programm mit einem I/O-Error ab. Zur Abfrage des Seitenüberlaufs kann der Overflow-Schalter 'OF' benutzt werden.
Der Schalter 'OF' wird gesetzt, wenn die Überlaufzeile (definiert durch Kanal 12 der L-Karte) überschritten wurde. Ist kein Kanal 12 angegeben, so wird der 'OF'-Schalter gesetzt, wenn die Formularlänge überschritten wurde.
Es wird empfohlen, dass die erste Druckausgabe einen Vorschub nach Kanal 01 enthält, da sonst die Steuerung mit dem 'OF'-Schalter fehlerhaft sein kann.

Bei VSAM-Dateien muss in Spalte 15 der F-Karte 'I', 'U' oder 'O' eingetragen werden. Bei 'I' wird die Datei nur für Eingabe eröffnet, bei 'U' und 'O' für Ausgabe. Dies ist wichtig bei VSAM-Dateien mit Shareoption 2. Dateien, die z. B. im CICS für Ausgabe eröffnet sind, können im Batch noch für Eingabe eröffnet werden, wenn sie mit Shareoption (2) definiert sind. Ein Eröffnen für Ausgabe im Batch würde hierbei zu einem VSAM-Open-Fehler und damit zu einem Programmabbruch führen.

Für VSAM-Dateien ist die Option REUSE unterstützt. Damit können Arbeitsdateien erstellt werden, ohne dass hierzu immer ein DELETE/DEFINE erfolgen muss.

Die Dateiverarbeitungslogik ist im Batch die gleiche wie bei der Online-Verarbeitung (siehe Abschnitt 2306 bis 2308).

Anders als bei Onlineverarbeitung können Sätze im Batch auch sequentiell upgedatet oder gelöscht werden.

Beim Hinzufügen von Sätzen ist folgendes zu beachten:

1. Sätze in KSDS Dateien können direkt und sequentiell in Schluesselfolge hinzugefügt werden. Bei sequentiellem Hinzufügen müssen die

Sätze in aufsteigender Schlüsselfolge vorliegen. Sätze können zwischen bestehende Datensätze eingefügt werden. Existiert bereits ein Satz mit dem angegebenen Schlüssel, so wird der Schalter Duplicate Record 'DR' gesetzt.

2. Sätze in ESDS-Dateien werden sequentiell an das Ende der Datei angefügt. Es können keine Sätze zwischen bestehende Sätze eingefügt werden.

3. Sätze in RRDS-Dateien können sequentiell und direkt hinzugefügt werden. Bei sequentiellem Hinzufügen werden die Sätze an das Ende der Datei angefügt. Bei direktem Hinzufügen können Sätze nur eingefügt werden, wenn zu der angegebenen Satznummer kein Satz existiert, z. B. wenn der Satz vorher gelöscht wurde. Existiert bereits ein Satz mit der angegebenen Satznummer, so wird der Schalter Duplicate Record 'DR' gesetzt.

Beim Hinzufügen ist in der Ausgabe immer 'ADD' einzutragen. Dies gilt auch für das erstmalige Laden von Dateien.

Löschen von Sätzen ist nur in KSDS- und RRDS-Dateien möglich. Die Sätze können sowohl sequentiell als auch direkt gelöscht werden.

Ab Release 1.5 sind sequentielle Platten und Banddateien und alternative Drucker in HL1-Modulen unterstützt. Die Datei muss immer in dem Programm eröffnet werden, in dem sie verarbeitet wird, ansonsten erScheint bei der Ausführung eine Fehlermeldung.

Einschränkungen und Hinweise zur Batch-Version 6310

Grundsätzlich sind im Batch alle CPG-Operationen unterstützt, bis auf folgende Ausnahmen:

1. Operationen, die speziell für Online-Verarbeitung konzipiert sind. Hierzu gehören:

   COMRG DEBUG DEQ ENQ EREAD EXITD EXITI EXITT EXPR
   LOADT MAP MAPD MAPI MAPO MAPP PROT READI SAVET
   SDUMP SYNCP TESTT TWALD TWASV

   Ebenfalls ist die Operation RNDOM*ALL,die speziell für Online-Programmverbindungen konzipiert ist, nicht unterstützt.

2. Sonstige Operationen, die nicht unterstützt sind:

   CALLD CLEAR EXITP IFC

3. Die Operation CHAIN mit U oder P in Spalte 53 sperrt den gelesenen Satz nicht für nachfolgende CHAIN-Operationen.

   Achtung: CHAIN mit U oder P sollte im Batch nicht verwendet werden. In Datasets kann z.B. durch Abfrage von CPGTID auf ' ' für den Betrieb im Batch auf CHAIN oder CHAIN C umgeschaltet werden.

Folgende Schalter sind nicht unterstützt:

1. die Bildschirm-Schalter A1-A3 CL DE P1-PC Q1-QC SP und NI

In der H-Karte sind folgende Einträge nicht unterstützt bzw. erforderlich.

Spalte 27-30 TWA Größe des Vorprogramms
33 Verarbeitung bei EXPR und EXITP
39 Kleinbuchstaben am Bildschirm
45 Release Parameter
46 Generieren Dataset Modul
47 Methodenbank muss 'B' sein.
48 Eigene Adressierung muss ' ' sein.

Die Felder CPGIFC und CPGTIO können nicht mit EDIT- und SELCT-Operationen angesprochen werden.

User-Copy-Books in der TWA sind nicht unterstützt.

Die Diagrammausgabe über DIAG ist unterstützt.

Werden nicht erlaubte Operationen in einem Batchprogramm ausgeführt, so bricht das Programm mit einem formatierten Dump ab.

Dateiverarbeitung: Der Eingabebereich ist mit X'00' (online mit X'40') formatiert.

Kompatibilität 6320

Die Methodenbank wird zur Ausführungszeit geladen. Bestehende Batch-Programme können damit nach einem HL1-Releasewechsel auch ohne erneute Umwandlung mit der neuesten Batch-Methodenbank arbeiten. Makros, die für die Ausführung benötigt werden, sind in der Batch-Methodenbank (Phase CPGMBKB) enthalten. Lediglich die Makros GETVIS und CDLOAD werden bei Programmstart zum Laden der Methodenbank benötigt.

Die Routine zum Laden der Batch-Methodenbank wird zum Batchprogramm hinzugelinkt. Bei einem Wechsel der Betriebssystem-Software brauchen die HL1-Batchprogramme nur dann umgewandelt zu werden, wenn sich am Objekt-Code zum Laden der Batch-Methodenbank (bei den Makros GETVIS und CDLOAD) etwas ändern sollte.

Module für Batch- und Online-Verarbeitung 6325

Ein Modul kann grundsätzlich sowohl online als auch im Batch eingesetzt werden. Sollte ein unterschiedlicher Ablauf zwischen Batch- und Onlineausführung erforderlich sein, so kann dies mit dem Feld CPGTID gesteuert werden, z.B.:

```
C           CPGTID    IF   '    '
 *                     .                              BATCH
C                      ELSE
 *                     .                              ONLINE
C                      END
```

Ausführen eines HL1-Batchprogramms 6330

Nach folgendem Muster kann der Batch-Job aufgebaut sein.

```
// JOB HL1BATCH
   ... hier ggfs. ASSGN, DLBL und EXTENT Karten einfügen
// EXEC BATCHPRG,SIZE=AUTO
   ... evtl. Datenkarten
/*
/&
```

Batchprogramme benötigen den GETVIS-Bereich der Partition. Daher ist in der EXEC-Anweisung immer der SIZE-Parameter anzugeben. Die Partition sollte ausreichend groß sein (z.B. 256K + evtl. benötigter Speicher bei Verwendung von Temporary Storage).

Sollten bei einer Programmausführung Fehler auftreten, so kann dies wie folgt verursacht sein:

1. Durch einen Fehler im Anwendungsprogramm, z. B. verursacht durch einen Datenfehler oder durch eine Operation die im Batch nicht unterstützt ist. Hierdurch bricht das Programm ab und es wird ein formatierter DUMP erzeugt. Die folgende Register- und Speicherbelegung hilft bei der Fehlersuche. In der Regel wird der Fehler im Datenbereich (Register 12) zu suchen sein.

2. Durch einen Fehler im Interface, z.B. einem VSAM-Fehler. Hierbei wird ebenfalls ein formatierter Dump gedruckt. Die Abbruchursache ist dabei am Anfang des Dumpbereichs im Klartext ausgedruckt. Gegebenenfalls sind VSAM-Error- und -Returncodes in der Broschüre 'VSAM Messages and Codes' nachzuschlagen. Die Angabe dieser Codes erfolgt dezimal. Nach einem VSAM-Fehler erfolgt ein Programmabbruch. Die folgenden Job-Steps werden dann nicht mehr ausgeführt.

UPSI-Schalter 6335

Bei VSE-Systemen können mit dem JCL-Statement // UPSI xxxxxxxx Schalter gesetzt werden. Diese Schalter können im Programm mit U1 bis U8 abgefragt werden.

Für HL1-Bausteine ist aus Kompatibilitätsgründen zum CICS die Benutzung von T1 bis T8 freigestellt (T1 ist identisch mit U1, T2 mit U2 usw).

UPSI-Schalter können mit SETON eingeschaltet und mit SETOF ausgeschaltet werden, dies gilt jedoch nur für die Dauer der Programmausführung.
Später laufende Job-Steps sind hiervon nicht betroffen, diese müssen gegebenenfalls durch weitere JCL-Statements gesetzt werden.

UPSI-Schalter können gesetzt werden, um z.B. bestimmte Dateien von der Verarbeitung auszuwählen oder auszuschließen. In diesem Fall sind hierbei die Operationen OPEN und CLOSE zu verwenden.

Beispiel:

```
C   U1              OPEN DATEI1
                    .
                    .                       Verarbeitung
                    .
C   U1              CLOSE DATEI1
```

Batch Methodenbank 6340

Die Batch-Methodenbank besteht aus folgenden Teilen:

1. Aus den allgemeinen Methodenbankroutinen wie z.B. SORTA. Diese sind in der Phase CPGMBKB enthalten.

2. Aus dem File-Interface, welches die Dateizugriffe enthält.

3. Aus dem Batch-Interface, welches die Programmsteuerung enthält.

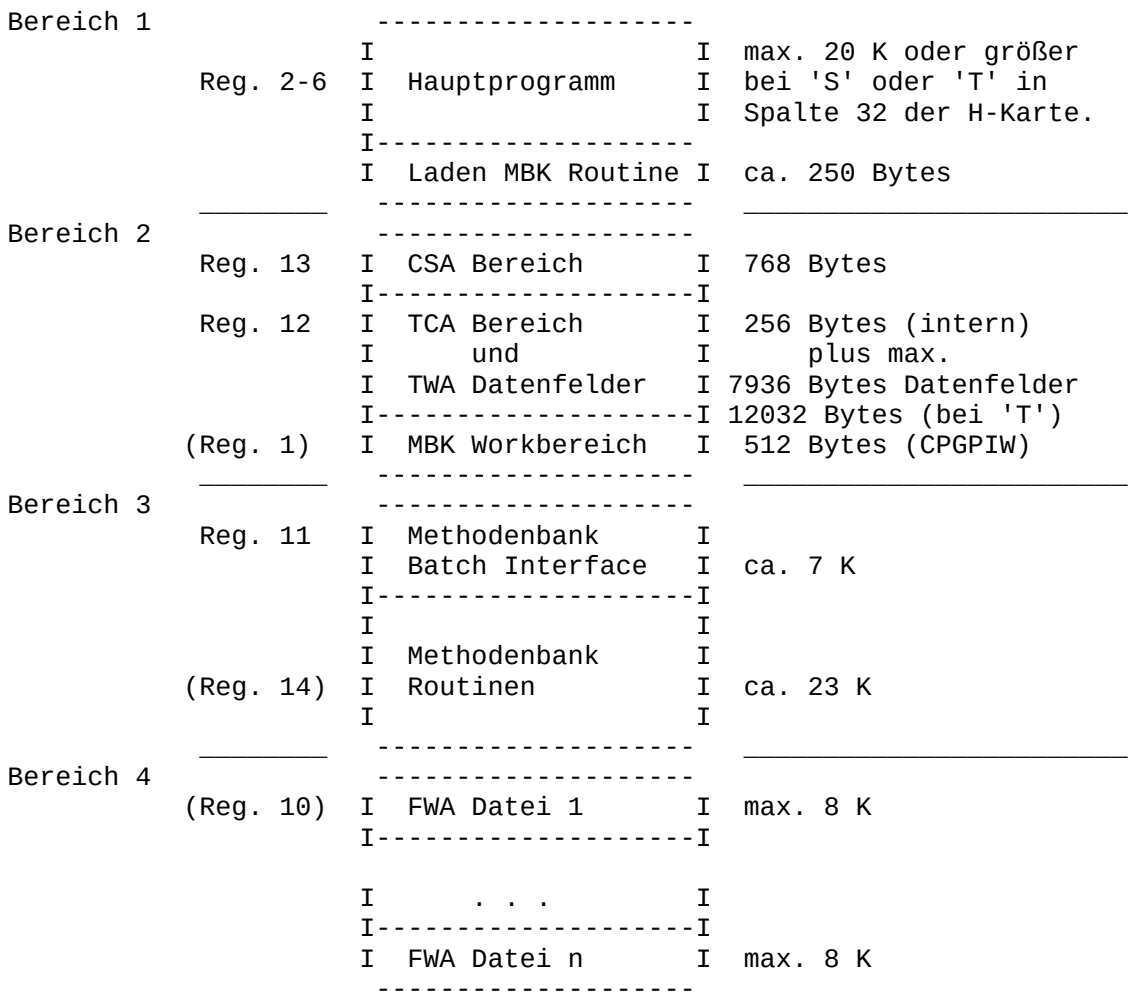
Die Register werden vom Anwendungsprogramm und von der Methodenbank folgendermaßen benutzt:

Register	0	unbenutzt
	1	Arbeitsregister (WORKREG)
	2-6	Basisregister (sowohl Anwendungsprogramm als auch MBK)
	7	erweiterter TCA Bereich (PWAREG)
	8	Unterprogrammregister (SUBREG)
	9	Ein-/Ausgaberegister (PUTREG) (GETREG)
	10	Ein-/Ausgabebereich (CIOABAR)
	11	Methodenbank-Basisregister (METHBAR)
	12	TCA-Bereich, Datenfelder Anwendungsprogramm
	13	CSA-Register, VSAM und Interface-Save-Bereich
	14	Verbindungsregister zu Methodenbankroutinen (LNKREG)
	15	Arbeitsregister (COUNTREG)

Die Register 2-6, 7, 11, 12 und 13 dürfen vom Anwender in Assembler Instruktionen nicht verändert werden.

Speicherbelegung 6350

Ein Speicherauszug, der z.B. durch einen Programmabbruch erzeugt wird, lässt sich in 4 Bereiche unterteilen:



Die Register 1 und 14 werden jeweils vor Einsprung in das Interface neu geladen. Das Register 10 wird jeweils vor Dateizugriffen auf die jeweilige File Work Area (FWA) positioniert.

Der Aufbau der Bereiche CSA, TCA und FWA ist wie folgt festgelegt:

1. Belegung des CSA Bereiches

Von	1 - 72	reserviert für VSAM und IO-Makros.
	73 - 96	reserviert.
	97 - 128	Savebereich Reg. 1-8 für MBK.
	129 - 192	Savebereich Reg. 0-15 für Batch Interface Control.
	193 - 224	Savebereich Reg. 1-8 für Batch Interface.
	225 - 255	reserviert.
	256	Schalter für Dump.
	257 - 320	Register 0-15 File Interface.
	321 - 448	HL1-Library

2. Belegung des TCA Bereichs

Von	1 - 4	CSA Adresse (Reg. 13 wird hier von MBK gerettet).
	5 - 8	MBK Adresse (Reg. 11 wird bei Init. gerettet).

9 -	12	FCT Adresse des Programms (CPGFCT).
13 -	256	reserviert.

3. Belegung des FWA Bereichs

Von	1 -	4	ACB Adresse der VSAM-Datei
	5 -	8	RPL Adresse für den VSAM Zugriff
	9 -	10	max. Satzlänge binär
	11 -	11	reserviert.
	12 -	12	Datei Statusanzeiger: Bit 1 an bei direkter, aus bei seq. Verarbeitung. Bit 2 an nach GET, PUT oder ERASE, aus nach ENDREQ. Bit 3 an nach GET, aus nach PUT. Bit 4 an bei RPL für ADD, sonst aus. Bit 5 reserviert. Bit 6 reserviert. Bit 7 an bei RRDS Dateien, sonst aus. Bit 8 an wenn die Datei eröffnet ist, sonst aus.
	13 -	14	Schlüssellänge binär.
	15 -	16	relative Schlüsselposition im Satz.

zusätzlich bei Dateien mit variabler Satzlänge:

17 -	18	variable Satzlänge einschließlich 4 Bytes Prefix. Vor Hinzufügen in KSDS oder ESDS Datei oder vor Ändern der Satzlänge in einer KSDS Datei.
19 -	20	reserviert.

Ab Stelle 17 bei Dateien mit fester Satzlänge, bzw. ab Stelle 21 bei Dateien mit variabler Satzlänge folgt der Datensatz, der maximal 8172 Bytes lang sein kann.

HL1-Umwandlung. 6550

Die Umwandlung von HL1-Bausteinen erfolgt wie eine CPG-Umwandlung, mit dem einzigen Unterschied, dass in der // EXEC Karte statt CPG jetzt HL1 eingetragen wird.

Beispiel:

```
// JOB XXXXXXXX
// EXEC HL1

   ... HL1-BAUSTEIN

/*
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT
/&
```

Das Strukturdiagramm 6600

Bei dieser Umwandlung wird immer nur der jeweilige Baustein umgewandelt. Die Umwandlungszeiten werden dadurch auf die Umwandlung des Unterprogramms beschränkt und sind daher sehr viel kürzer als bei der herkömmlichen Programmierweise. Andererseits gibt die Umwandlungsliste auch nur einen Überblick über einen Programmteil. Um dennoch eine Umwandlungsliste über das volle Programm oder die jeweilige Baugruppe zu erhalten, bietet HL1 eine zusätzliche Strukturliste an, die neben den Einzelbausteinen deren Verknüpfung und einen Überblick über den Datenfluss zeigt.

Voraussetzung für die Generierung der Strukturliste ist, dass alle in dieser Liste angezogenen Bausteine im CPG-Format geschrieben sind und vorher in die Source Statement Library des Systems katalogisiert wurden. Dies erfolgt z.B. im VSE mit folgenden JCL-Befehlen:

```
// JOB CATAL
// EXEC MAINT
   CATALS R.HMPROG       (wenn PROG der Name des Bausteins ist)
        BKEND

   ... HL1-BAUSTEIN

        BKEND
/*
/&
```

Für VSE SP 2/3 sieht dieser Job wie folgt aus:

```
// JOB CATAL
// EXEC LIBR,PARM='ACC S=LIB.SLIB'
   CATALOG HMPROG.R      (WENN PROG DER NAME DES BAUSTEINS IST)

   ... HL1-BAUSTEIN

/+
/*
/&
```

Das heisst HL1-Bausteine werden in die (RPG) R-Library katalogisiert und erhalten dort zur Unterscheidung von anderen Copy-Books vor dem Namen grundsätzlich die Konstante 'HM'.

Nachdem die einzelnen Bausteine so katalogisiert worden sind, kann die Strukturliste mit folgendem Kartensatz generiert werden:

```
// JOB HL1 STRUKTUR
// EXEC HL1STR

   ...  HL1 STRUKTURKARTEN    (Siehe folgendes Beispiel)

/*
// EXEC PROC=HL1STR
/&
```

JOB FÜR VSE/SP

```
// JOB HL1 STRUKTUR VSE/SP
// EXEC HL1STR

     H     FEC ACCESS SUBLIB=LIB.SLIB              STRUKTUR BEISPIEL

     H*    FEC --------------------------------: READER RETURN ADRESSE
     H*        ACCESS SUBLIB=LIB.SLIB ---------: ACCESS FÜR LIBR JOB

     S*               KANAL    MODUL    TEXT
     SI-------------           RZHPT    HAUPTPROGRAMM
     S I------------           RZ001    PAGE AUFBEREITEN
     S  I-----------           RZ002    KOPF AUSGEBEN
     S  I-----------  KANAL    RZ003    SATZ LESEN
/*
// EXEC PROC=HL1STR
ASSGN SYSIN,C02
// EXEC HL1ST2
/*
CLOSE SYSPCH,FED
/*
/&
```

Die Procedure HL1STR muss vorher katalogisiert werden. Dazu kann das folgende Beispiel für das jeweilige System modifiziert werden.

```
CATALOG HL1STR.PROC                          REPLACE=YES
// DLBL IJSYSPH,'F4.WORK.TWO',0,SD
// EXTENT SYSPCH,PRD001,1,0,20000,2000
ASSGN SYSPCH,X'200'
ASSGN SYSIN,X'200'
/*
/+


// EXEC HL1STR
    H                                                HAUPTPROGRAMM OP
    S*               KANAL    MODUL   TEXT
    SI--------------          OPH     Offene Posten Hauptprogramm
    S I------------- KEYS     OPK     OP Kopfdaten
    S II------------ KEYS     OPM     OP Bildschirmmaske
    S II------------ STDAT    OPD     OP Stammdaten
    S III----------- STSGM    OPL     OP Stammdaten lesen
    S III----------- AKDAT    OPA     OP Abrechnungskreis
    S III----------- LIEFBD   OPT     OP Text Lieferbedingungen
    S II------------ STDAT    OPB     OP Bildschirmausgabe Stammdaten
    S I------------- KEYS     OPS     OP Seite anzeigen
    S  I------------ OPZEIL   OPZ     OP Zeile aufbereiten
    S   I----------- OPDAT    OPO     OP Daten lesen
/*
```

Das Programm HL1STR ist ein reines Listprogramm, das lediglich eine Übersicht über die gesamte Baugruppe liefert. Dieses Programm muss daher nicht bei jeder Änderung in einem Baustein ausgeführt werden.

Das Programm listet zunächst die Strukturkarten auf und ergänzt jede einzelne Karte am rechten Rand der Liste um eine laufende Nummer. Diese laufende Nummer dient zum schnelleren Auffinden des Bausteins in der folgenden Liste und wird in der ersten Zeile einer jeden Seite zum jeweiligen Baustein angelistet.

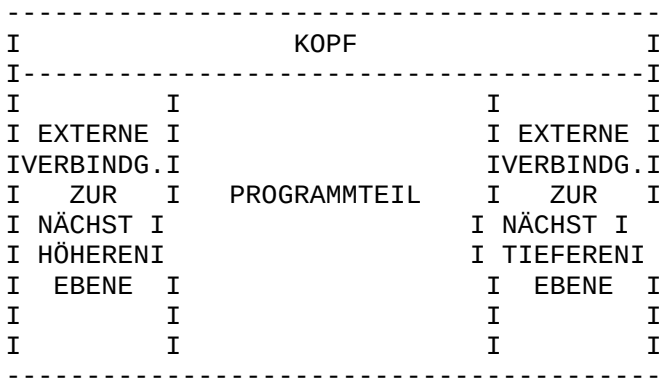
Das Strukturdiagramm wird in der Liste gespreizt, um die Übersicht zu erleichtern und hat folgenden Aufbau.

```
HAUPTPROGRAMM OP                        STRUKTURDIAGRAMM     20.12.93
_____________________________________________________________________

STRUKTUR                                                  KANAL    MODUL
_____________________________________________________________________

I-----------------------------------------                         OPH
   I--------------------------------------                KEYS     OPK
   I  I-----------------------------------                KEYS     OPM
   I  I-----------------------------------                STDAT    OPD
   I  I  I--------------------------------                STSGM    OPL
   I  I  I--------------------------------                AKDAT    OPA
   I  I  I--------------------------------                LIEFBD   OPT
   I  I-----------------------------------                STDAT    OPB
   I--------------------------------------                KEYS     OPS
      I-----------------------------------                OPZEIL   OPZ
         I--------------------------------                OPDAT    OPO
```

Dem Strukturdiagramm folgen die Programmlisten der einzelnen Bausteine. Diese Listen haben grundsätzlich folgenden Aufbau:



Im Kopfteil werden der Phasenname und die Beschreibung (Text aus der Strukturkarte) des Bausteins, Datum, Uhrzeit und die laufende Nummer des Bausteins im Strukturdiagramm angelistet. Außerdem kann wahlweise eine Verknüpfungsleiste angelistet werden, die den Pfad vom Baustein in der höchsten Hierarchiestufe bis zum angelisteten Baustein beschreibt.

Für den Baustein 'OPT' zum Beispiel enthält der Listkopf folgende Daten:

```
OPT     TEXT LIEFERBEDINGUNGEN          STRUKTURDIAGRAMM     21.12.93
```

```
     1       2       4       7
           KEYS   STDAT  LIEFBD
   OPH      OPK     OPD     OPT
```

Die Verknüpfungsleiste zeigt an, dass der Pfad von 'OPH' nach 'OPT' über 'OPK' und 'OPD' führt. Über den Namen der Bausteine werden die jeweils zugehörigen Datenkanaele und darüber die zugehörigen laufenden Nummern im Strukturdiagramm angezeigt.

Der Programmteil der Programmliste entspricht im Aufbau der Umwandlungsliste. Das folgende Bild zeigt einen Auszug aus der Programmliste für den Baustein 'OPT'.

```
LIEFBD  LF     <--->  001D       LF             1 0       SCHLÜSSEL F
        LFBED         002D       LFBED         11         LIEFERBEDING

OPD   ------------->
                      003C                    SETIN          LF
                      004C                    EDIT           LFBED  11
OPD   <-------------
                      005OLFBED   F
                      006O                 61                11 'NACHN
                      007O                 62                11 'VORAU
                      008O                 63                11 'ZUG U
                      009O                 64                11 'LIEFE
                      010O                 65                11 'AKKRE
                      011O                 66                11 'DOKUM
                      012O                 67                11 'ZAHLU
                      013O                 68                11 'VERGL
                      014O                 69                11 'KONKU
```

Die Pfeile im linken Teil der Liste geben die Richtung des Ablaufs an. In Zeile 1 wird gezeigt, dass die Daten des Kanals 'LIEFBD' sowohl von der höheren Ebene geholt, als auch dorthin zurückgebracht werden (Pfeil in beide Richtungen). Zu Beginn der Rechenbestimmungen wird angezeigt, dass die Verarbeitung vom Baustein 'OPD' an diesen Baustein übergeben wird und am Ende der Rechenbestimmungen wird angezeigt, dass die Verarbeitung im Baustein 'OPD' wieder fortgesetzt wird.

Das folgende Bild zeigt den entsprechenden Auszug des rechten Teils der Programmliste für den Baustein 'OPD'.

```
                                        D         LF        1    1
                                        D         LFBED     2   12
                                        I <--->    LIEFBD   ------
            P   1   10LF                I                    LF
                2  12 LFBED              I                    LFBED

                                          ----------------------
EXHM OPT       LIEFBD                   C  --->    LIEFBD   OPT
```

Bei der Operation EXHM wird angezeigt, dass die Verarbeitung auf der nächst tieferen Stufe fortgesetzt wird. Es wird in den Baustein OPT verzweigt, wobei der Datenkanal LIEFBD übertragen wird. Der Datenkanal wird in den Eingabebestimmungen beschrieben. Die Pfeile zeigen an, dass die Daten in beide Richtungen übertragen werden.

Fehlermeldungen in der Umwandlungsliste 6900

Die Fehlerdiagnostik in der Umwandlungsliste erfolgt unmittelbar bei der fehlerhaften Instruktion mit einem Kurztext.

Enthält die Instruktion mehrere Fehler, so werden die zugehörigen Texte in der Umwandlungsliste vor der Instruktion angelistet.
CPG bietet verschiedene Möglichkeiten, die Position des Kurztextes zu bestimmen. Siehe Spalte 15 der H-Karte, Abschnitt 5100.

Die Diagnostik erfolgt normalerweise in deutscher Sprache. Die Eintragung eines 'E' in Spalte 21 der H-Karte bewirkt eine Ausgabe sämtlicher Texte in englischer Sprache. Alle Fehlertexte sind in einer eigenen Textphase zusammengefasst, die es dem Anwender sehr leicht ermöglicht, die Fehlernoten in jede andere Sprache zu übersetzen.

Folgende Fehlermeldungen können während der Umwandlung des Programms bei fehlerhaften Instruktionen angelistet werden.

ALPHAFELD IN RECHENOPERATION

ASSEMBLER ERRORS

In seltenen Ausnahmefällen kann es dazu kommen, dass bei der CPG-Umwandlung keine Fehler erkannt werden, aber der Assembler Fehlermeldungen ausdruckt. Siehe folgende Möglichkeiten.

BEZUGSZAHL SPALTE 54 FEHLT

Bei einer CHAIN-Operation muss in Spalte 54-55 eine Bezugszahl angegeben werden. Diese Bezugszahl wird gesetzt, wenn der gesuchte Satz nicht gefunden wurde.

B FÜR ALPHAFELD

In Spalte 44 wurde ein 'B' (binäre Ausgabe) eingetragen. Das Feld In Spalte 32 bis 37 wurde jedoch alphanumerisch definiert. Binäre Ausgabe ist nur für numerische Felder möglich.

BINÄRFELD ZU LANG

Binärfelder dürfen 2 oder 4 Bytes groß sein. Die zugehörigen numerischen Felder dürfen dabei nicht größer als 4 bzw. 9 Stellen sein.

CPG FEHLER

Alle CPG-Funktionen werden bei jedem Releasewechsel einzeln und in den in der Praxis häufig vorkommenden Kombinationen sorgfältig getestet. Bei aller Sorgfalt ist jedoch nicht auszuschließen, dass bei diesen Tests einmal ein Fehler nicht erfasst wird.

Aus diesem Grunde führt der CPG eine Selbstdiagnostik durch, die dem Anwender die Fehlersuche ersparen soll, wenn ein solcher Ausnahmefall eintritt.

Die Meldung CPG FEHLER darf im Normalfall nie auftreten. Sollte dennoch bei einer Umwandlung diese Meldung erscheinen, so bitten wir, uns die Umwandlungsliste zur Analyse zuzuschicken.

CPGSXn PHASE NOT FOUND

In Spalte 21 der H-Karte wurde eine Eintragung für Sprache, wozu keine Phase verfügbar ist, vorgenommen.

DATEINAME FEHLT

Die erste Ausgabebestimmung enthält keinen Dateinamen. Alle Folgekarten bis zur nächsten Satzbestimmung mit gültigem Dateinamen werden ignoriert.

DATEI DOPPELT DEFINIERT

Die gleiche Datei wurde in der F-Karte zweimal eingetragen. Der Dateiname und der Phasenname sind gleich.

DATEI NICHT DEFINIERT

Für die in Spalte 7-14 angegebene Datei fehlt eine gültige Dateizuordnung.

Für die angegebene Datei fehlen die I-Karten

Wenn bei der Operation UPDAT und WRITE für die entsprechende Datei keine Eingabe (I-Karten) definiert wurde.

Datenstruktur Tabelle voll

Copy CPG*CDT0 vergrößern und Compiler neu linken.

EINTRAGUNG SPALTE 16 - 26

Ein Eintrag in diesen Spalten ist falsch.

EINTRAGUNG SPALTE 31 - 45

Ein Eintrag in diesen Spalten ist falsch.

EINTRAGUNG SPALTE 47 - 51

Ein Eintrag in diesen Spalten ist falsch.

END STATEMENT FEHLT

Nach einer DO- oder IF-Operation wurde kein gültiges END-Statement definiert.

ERGEBNISFELD FALSCH

Es wurde in Spalte 43-48 keine Eintragung vorgenommen. Im Ergebnisfeld ist bei einer solchen Fehlermeldung eine Eintragung erforderlich.

ERGEBNISFELD FALSCH/FEHLT

Bei der CAB-Operation wurde kein Ergebnisfeld eingetragen.

ERGEBNISFELD FEHLT

Das Ergebnisfeld muss bei dieser Operation einen gültigen Feldnamen enthalten.

ERGEBNISFELD NICHT BLANK

Es wurde eine Eintragung in Spalte 43-48 vorgenommen. Das Ergebnisfeld ist bei einer solchen Fehlermeldung nicht unterstützt.

FAKTOR 1 IST FALSCH

Beim CHAIN ist die Feldlänge von Faktor 1 ungleich der Keylänge aus der F-Karte.

Beim READ eines Storage wurde Faktor 1 nicht mit 0 Dezimalstellen definiert.

FAKTOR 1 NICHT BLANK

Es wurde eine Eintragung in Spalte 18 - 27 vorgenommen. Faktor1 ist bei einer solchen Fehlermeldung nicht unterstützt.

FAKTOR 2 IST FALSCH

Ein Mapname wurde in Hochkommata eingetragen.

FAKTOR 2 IST FALSCH/FEHLT

Bei der LOKUP-Operation wurde ein fester Index eingetragen.

FAKTOR 2 NICHT BLANK

Es wurde eine Eintragung in Spalte 33 - 42 vorgenommen. Faktor2 ist bei einer solchen Fehlermeldung nicht unterstützt.

FAKTOR 2 KEINE FELDGRUPPE

Der Feldname in Faktor 2 muss als Feldgruppe definiert sein.

FALSCHER AUFBEREITUNGSSCHLÜSSEL

Spalte 38 enthält einen ungültigen Aufbereitungsschlüssel.

FALSCHE EINTRAGUNG SCHLÜSSELFELD

Spalte 29-30 der Dateizuordnung enthält keine gültige Schlüssellänge.

FALSCHE EINTRAGUNG SPALTE 11

Spalte 11 der CPG-Steuerkarte enthält eine falsche Eintragung. Gültige Eintragungen sind: A,C,D,M,T oder Blank.

FALSCHE EINTRAGUNG SPALTE 15

Spalte 15 der CPG-Steuerkarte enthält nicht N, P oder Blank.

FALSCHE FELDPOSITIONEN

Bei der Eingabe-Feldbestimmung ist die Von- oder Bis-Position = 0.

FELD NICHT DEFINIERT

In Spalte 18-23,33-38 oder 43-48 der Rechen-Bestimmungen oder in Spalte 32-37 der Ausgabebestimmungen ist ein Feldname eingetragen,

der weder in den Eingabe- noch in den Rechenbestimmungen definiert wurde. Die Definition erfolgt in der Eingabe durch Angabe der ersten und letzten Position in einem Datensatz, oder in den Rechenbestimmungen durch Angabe der Feldlänge in den Spalten 49-51, wenn das Ergebnisfeld den Namen des zu definierenden Feldes enthält.

Bei der Dataset-Verarbeitung wurde das CPGHIC-Feld nicht definiert.

Wenn CPGFIl in I, C oder O-Statements vorkommt.

FELD ZU GROß

Ein alphanumerisches Feld ist größer als 256 Bytes, oder ein numerisches Feld wurde mit mehr als 15 Stellen definiert.

FELD ZU LANG FÜR BINÄR

Bei binärer Ausgabe darf das auszugebende numerische Feld maximal 9 Stellen groß sein. Bei größeren Feldern muss das Ergebnis vorher in ein 9-stelliges Feld zwischengespeichert werden.

FELDAUFBEREITUNG FEHLT

Es fehlt in der Eingabe bzw. Ausgabe die Feldaufbereitung.

FELDGRUPPE ZU GROß

Bei einer Diagrammausgabe ('C' in der voranliegenden Satzbestimmung) wurde eine Feldgruppe angegeben, die insgesamt größer als 69 Stellen ist.

FELDNAME DOPPELT DEFINIERT

Der gleiche Feldname wurde bereits für ein Feld mit anderen Felddaten (Feldlänge,Feldart,Dezimalstellen) vergeben.

Es wurden Feldgruppen in den Eingabebestimmungen unterschiedlich gegenüber den E-/D-Karten definiert.

Die Feldaufbereitungen mit TYPE für ein Feld in der Eingabe bzw. Ausgabe sind nicht zusammenhängend codiert.

FELDNAME FEHLT

Es wurde kein Feldname eingetragen.

FILE QSFMAP CLOSE ERROR X'90' (144)

Bei der // EXEC CPG-Anweisung wurde der Eintrag SIZE=XXX nicht groß genug eingetragen. (192K Minimum).

GRUPPENWECHSEL FALSCH DEFINIERT

Gruppenberechnungen liegen nicht in aufsteigender Reihenfolge (L1, L2, L3, usw.)

GRUPPENBEGRIFF FALSCH DEFINIERT

Es werden L-Schalter in den C-Bestimmungen abgefragt, aber in den I-Karten nicht definiert.

HL1-MODUL NICHT GEFUNDEN

Bei einer EXHM-Operation wurde ein Baustein eingetragen, der in der HL1-Tabelle nicht vorgefunden wird.
In der H-Karte Spalte 22 wurde die Eintragung für eine private HL1-Library nicht vorgenommen.

Es wurde ein QPG-Dataset in der Ausgabe definiert.

HOCHKOMMA FEHLT

Eine Konstante wird nicht durch ein Hochkomma abgeschlossen.

INDEX FALSCH

Bei einer indizierten Operation wurde ein alphanumerisches Feld als Index eingetragen.

KEINE UPDATE DATEI

Das Programm enthält Ausgabebestimmungen für eine Datei, die in der Dateizuordnung kein 'U' in Spalte 15 enthält.

Eine Datei hat in der F-Karte Spalte 15 ein 'I' eingetragen und in den Rechenbestimmungen wird versucht die Operation UPDAT, DELET oder WRITE einzusetzen.

KONSTANTE FÜR BINÄR UNGÜLTIG

B in Spalte 44 der Ausgabebestimmungen darf nicht in Verbindung mit einer Konstanten verwendet werden.

LABEL IN OPERATION

Es wurde ein Anspringpunkt anstelle eines gültigen Feldnamens eingetragen.

NAMENTABELLE VOLL

Es wurden zu viele Felder bzw. Anspringpunkte definiert (siehe Seite 7400).

NICHT UNTERSTÜTZT

Es wurde eine MVR-Operation nicht unmittelbar nach einer DIV-Operation durchgeführt.

Vor einer MVR-Operation wurde eine DIV-Operation mit Runden durchgeführt.

Es wurde eine RPG-Funktion angegeben, von CPG nicht unterstützt wird.

Nach einer BEGAS-Operation folgt kein gültiges ENDAS.

Bei der Subset-Version wurden die Einschränkungen nicht beachtet.

Es wurde versucht, eine Batch-Umwandlung durchzuführen. Dies ist nur mit HL1 möglich.

Es wurde ein Buffer-Mode-Drucker im Batch-Programm definiert.

Es wurde versucht, ein EDIT CPGTIO durchzuführen.

Bei der Bildschirmausgabe wurden C-Schalter verwendet.

Wenn bei der Operation DELC, ELIM, FILL oder REPLC eine Datenstruktur im Ergebnisfeld eingetragen wird, mit einer Feldlänge größer 256 Bytes.

Die Operation DEBUG wurde in Command-Level-Programmen verwendet.

In der CICS-Version in Spalte 46 der H-Karte wurde ein 'S' oder ein 'T' eingetragen.

COMRG mit CPGSIN wurde nicht in einem Command-Level-Programm benutzt.

Wenn MAPO mit einem Eintrag in Spalte 53 codiert wird.

Wenn DO mit Faktor 2 und LOOP codiert wird.

Wenn mehr als 99 Subroutinen codiert werden.

Bei einer Ausgabe-Datei wurde keine Feldausgabe codiert.

Siehe auch Einschränkungen Seite 2900.

OPEN ERROR X'80' DATEI CPGWRK

Dieser Fehler tritt auf, wenn in Spalte 17 der CPGURSIT ein 'A' oder 'T' angegeben wurde und das Data Dictionary nicht installiert wurde und außerdem:

1. Eine F-Karte im Programm unvollständig ausgefüllt wurde, oder
2. Bei einer I-Dateibeschreibung in Sp. 15-16 'DD' eingetragen ist.

oder wenn das Data Dictionary installiert ist und:

1. Bei der CPG-Umwandlung die DLBL-Anw. der Datei CPGWRK fehlt oder
2. Bei der // EXEC CPG-Anweisung der Eintrag SIZE=XXXK nicht oder zu klein angegeben wurde.

In allen anderen Fällen bitte den zentralen Wartungsdienst der Lattwein GmbH befragen.

OPEN ERROR: RC=08 EC=110 (for empty 'ESD' file)

Diesen Fehler kann man abfragen, wenn die Datei in der F-Karte mit NO (no open) definiert wird und mit der Operation OPEN eröffnet.

P FÜR ALPHAFELD

Gepackte Ein- oder Ausgabe für ein Alphafeld ist nicht möglich.

PHASENNAME FEHLT

Spalte 75-80 der CPG-Steuerkarte enthält keinen Phasennamen. Soll das Programm trotzdem katalogisiert werden, so muss vor der '// EXEC CPG'-Karte eine '// OPTION CATAL' und eine 'PHASE'-Karte liegen, sowie Spalte 31 der H-Karte den Eintrag 'O' enthalten.

SATZLÄNGE FEHLT

In der Dateizuordnung ist eine Satzlängen-Eintragung erforderlich.

SCHABLONE FÜR ALPHAFELD

Ein Alphafeld kann nicht mit Schablone ausgegeben werden.

SCHABLONE FÜR PACKFELD

Ein numerisches Feld, welches gepackt ausgegeben wurde (P in Spalte 44), kann nicht mit Schablone definiert werden.

SPALTE 6 IST UNGÜLTIG

Spalte 6 enthält keine gültige CPG Kartenart. Gültige Eintragungen sind: H,F,E,D,L,I,C,O,-.

SPALTE 15 IST NICHT E

Spalte 15 einer Ausgabebestimmung enthält eine ungültige Eintragung. Gültige Eintragungen sind: E, F, C, oder 'OR' bzw. 'AND' in Spalte 14-16.

UNGLEICHE FELDARTEN

Bei der COMP-Operation wurden ungleiche Feldarten verwendet.

UNGÜLTIGE BEZUGSZAHL

Gültige Eintragungen sind: 01 bis 99, P1 bis P9 und PA,PB,PC,SP DE, EF, DR, IC, NI, A1, A2, A3, L1-L9, Q1 bis Q9, QA, QB, und QC.

UNGÜLTIGE DATEIART

Spalte 15 der Dateizuordnung enthält eine Dateiart, die für die angegebene Einheit ungültig ist (z.B.'I' für Drucker).

UNGÜLTIGE DATEI

Es wurde kein gültiger Dateiname eingetragen.

UNGÜLTIGE EINHEIT

Spalte 40-46 der Dateizuordnung enthält eine ungültige Ein- Ausgabeeinheit (siehe Abschnitt 3, Dateizuordnung).

UNGÜLTIGE EINTRAGUNG

Die Karte enthält eine ungültige Eintragung. Die Meldung erfolgt wenn

1. Felder,die für diese Operation nicht benutzt oder nicht unterstützt werden, ungleich Blank sind.
2. numerische Felder Alphazeichen enthalten.
3. numerische Felder nicht rechtsbündig eingetragen sind.
4. Alphafelder nicht linksbündig eingetragen sind.
5. Felder Sonderzeichen oder Blanks im Feldnamen haben.
6. CPGTCT bzw CPGCSA in einer anderen Operation als EDIT oder SELCT verwendet wird.
7. in der Ausgabe ein Kanalvorschub > 12 eingetragen wird.
8. der Feldname im Ergebnisfeld ab Spalte 42 eingetragen wird
9. Datei = RRDS und Satzformat variable 'V'.
10. bei einer Konstanten nach dem Schluss-Hochkomma noch ein

Text folgt.
11. bei einem HL1-Baustein in Spalte 47 der H-Karte ein L wie Command Level eingetragen wird.
12. in der D-Karte als Feldlänge eine '0' eingetragen ist.
13. eine ungültige numerische Konstante eingetragen wurde.
14. READER in einem Online-Programm codiert wird.
15. Datenstruktur (DS) ohne Feldname (SP.7) codiert wird.

UNGÜLTIGE EINTRAGUNG SPALTE 8 - 9

UNGÜLTIGE EINTRAGUNG SPALTE 10

UNGÜLTIGE EINTRAGUNG SPALTE 27 - 30

UNGÜLTIGER FELDNAME

Ein Feldname darf 1 bis 6 Stellen lang sein. Das erste Byte muss ein Buchstabe (A bis Z) sein. Die folgenden Bytes können Buchstaben oder Ziffern sein. Sonderzeichen sind nicht erlaubt.

UNGÜLTIGE KONSTANTE

UNGÜLTIGE NEGATION, N ANGENOMMEN

In einer Negationsspalte wurde ein Zeichen ungleich N oder Blank gefunden. N wird angenommen. Möglicherweise wurde eine Bezugszahl falsch gelocht.

UNGÜLTIGE OPERATION

Der Operationsschlüssel in Spalte 28-32 ist ungültig oder nicht unterstützt.

Bei einer Subroutine wurde kein 'ENDSR' eingetragen.

UNGÜLTIGE POSITION

1. Die Positionsangabe enthält nichtnumerische Zeichen.
2. Bei Platten- oder Druckerdateien (Linemode) wurde eine Position angegeben, die größer ist als die in der zugehörigen Dateizuordnung angegebene Satzlänge.
3. Bei Bildschirm- oder Druckerdateien (Buffermode) wurde eine Position angegeben, die in den beiden ersten Spalten eine Eintragung kleiner als 1 oder größer als die in der Dateizuordnung unter Blocklänge angegebene Zeilenzahl enthält, oder in den beiden letzten Spalten eine Eintragung kleiner als 01 oder größer als die in der Dateizuordnung unter Satzlänge angegebene Länge der Bildschirmzeile enthält.

UNGÜLTIGE PROGRAMMFUNKTION

Gültige Programm-Funktionen sind: P1 bis P9, PA, PB, PC, Q1 bis Q9, QA, QB, QC, A1 bis A3, DE, SP

UNGÜLTIGES END STATEMENT

Es wurde vorher keine gültige DO- bzw. IF-Operation definiert.

UNG. BREAK, CONTINUE, ELSE

Bei der Operation ELSE wurde vorher keine gültige IF-Operation definiert. Bei der Operation BREAK oder CONTINUE wurde vorher keine gültige DO-, DOU-, DOW-Operation definiert.

UNGÜLTIGES FARBATTRIBUT

Es wurde kein gültiges Farbattribut eingetragen.

UNG. EINTRAGUNG SPALTE 16-18

Es wurde keine gültige Eintragung vorgenommen.

UNG. SCHABLONENKÜRZEL

Es wurde kein gültiges Schablonenkürzel eingetragen, bzw. es wurde ein Schablonenkürzel bei einem gepackten Feld eingetragen. Bei Ausgabe eines gepackten Feldes ist die Ausgabe mit Schablone nicht zulässig.

UNG. FELDLÄNGE BEI CODE Y

Der Aufbereitungsschlüssel 'Y' ist nur bei einer Feldlänge zwischen 3 und 6 mit null Dezimalstellen unterstützt.

UNG. SCHABLONE/KONSTANTE

Es wurde zu einem Schablonenkürzel auch noch eine Aufbereitungs-Schablone ab Spalte 45 eingetragen.

ZUVIELE DO/IF STUFEN

Es sind maximal 40 DO / IF Verschachtelungen möglich. Diese Anzahl wurde überschritten.

ZU VIELE DATEIEN

Es sind maximal 100 Dateizuordnungen bzw. unterschiedliche LIST-Befehle in einem Programm möglich. Diese Anzahl wurde überschritten.

ZU VIELE FELDGRUPPEN

Es wurden mehr als 100 Feldgruppen in einem Programm definiert.

Warnings 6920

Bei fehlerfreier CPG-Umwandlung ist es möglich, dass im Anschluss an die Umwandlung eine Warning ausgegeben wird. Es sollte dann anhand der unten aufgeführten Erklärungen des Warning-Codes überprüft werden, ob die Gefahr besteht, dass das Programm unerwünschte Ergebnisse liefert.

Warning: Ursache

CPG0020 - Es wurden über 100 Listdokumente oder Datei-Einträge im Programm verwendet. Die Feldoptimierung sowie die Listreferenz kann nicht ordnungsgemäß ausgeführt werden.

CPG0030 - Diese Warning wird angelistet, wenn in einem Programm Datasets in der F-Karte definiert wurden und die Operation

EXPR mit 'S' oder 'I' in Spalte 53 verwendet wird. Wenn in dem aufgerufenen Unterprogramm auch ein Dataset definiert wurde, so kann diese Warning ignoriert werden. Wurde kein Dataset definiert, so kann die Funktion SAVE nicht ordnungsgemäß durchgeführt werden.

CPG0040 - Diese Warning wird angelistet, wenn im Programm eine Datei bzw. Satzart mit 'DD' eingetragen ist, die im Data Dictionary File nicht angelegt wurde. Das Programm wird mit Cancel abgebrochen.

CPG0050 - Ein Feld wurde mehrfach definiert. Die Definition in den I-Karten kann aber in der Umwandlungsliste nicht gefunden werden, weil das Listing per H-Karte unterdrückt wurde oder es wurde ein Dateiname als Feldname benutzt.

CPG0060 - In den D-Karten wurde eine Überlagerung nicht voll definiert.

CPG0061 - Diese Warning wird gesetzt, wenn ein Feld nicht definiert wurde.

CPG0070 - Storage Einzelsatz (nicht Queuing) und DATASET sind unter CICS-ESA nicht lauffähig.

CPG0080 - Diese Warning wird gesetzt, wenn für ein Label (TAG) ein CP-Name generiert wurde aus einem Wort, das in den ersten 5 Stellen mit einem Opcode identisch ist, z. B. PURGETPTC.

CPG0090 - Diese Warning wird gesetzt, wenn ein OPTIONS-Parameter ohne Bedeutung gefunden wurde.

CPG0100 - Diese Warning wird ausgegeben, wenn bei EXCPT der Name in Faktor 2 mit ADD oder DEL beginnt.

CPG0110 - Diese Warning wird ausgegeben, wenn bei einem HL1-Modul mit Dateien in den Options DAT oder PWA fehlt.

Fehlermeldungen in der Assembly 6930

Werden bei der Umwandlung Assemblerfehler ausgewiesen, so handelt es sich in der Regel um Adressierungsfehler (Addressability Errors), die bei der CPG-Ausführung nicht festgestellt werden können. Das heisst, entweder ist das Programm oder die TWA zu groß.

ADRESSABILITY ERRORS

Die Seite hinter der Programmierer-Check-Liste gibt Aufschluss über die fehlerhafte Programmroutine. (Siehe folgenden Listauszug).

```
                              EXTERNAL SYMBOL DICTIONARY        PAGE 1

SYMBOL    TYPE           ID  ADDR  LENGTH LD-ID

PROGN     SD (CSECT)    001 000000 005FFF

                              DUMMY SECTION DICTIONARY
```

```
SYMBOL     ID LENGTH

CPGTCADS  1FB 000FFF
```

Bei Standard-Programmen darf bei PROGN (Programmname) die Länge nicht größer X'5FFF' (24K) und CPGTCADS nicht größer X'FFF' (4K) sein.

Bei Programmen mit 8K TWA (H-Karte Spalte 48=1) darf die Programmgroesse nicht größer X'4FFF' (20K) und CPGTCADS nicht größer X'1FFF' (8K) sein.

Wurde in der H-Karte Spalte 32 ein 'S' oder ein 'T' eingetragen, so werden mehrere CSECTs generiert.

```
SYMBOL    TYPE           ID  ADDR  LENGTH LD-ID

PROGN     SD (CSECT)    001 000000 001FFF
BILD      SD (CSECT)    002 000000 002FFF
CPGF002   ER (EXTRN)    003
CPGF002   SD (CSECT)    004 000000 002FFF
SSSELCT   SD (CSECT)    005 000000 000FFF
BILD      ER (EXTRN)    006
SUBRN     SD (CSECT)    007 000000 002FFF
CPGAUSG   SD (CSECT)    008 000000 002FFF
PPEDIT    SD (CSECT)    009 000000 000FFF
```

Wurde ein 'S' eingetragen, so darf bei PROGN die Länge nicht größer X'1FFF' (8K), bei Feldaufbereitung die Länge nicht größer X'FFF'(4K) und jede weitere CSECT (Subroutine) nicht größer als X'2FFF' (12K) sein.

Wurde ein 'T' eingetragen, so darf die Programmlänge sowie Subroutine-Länge nicht größer X'1FFF' (8K) und CPGTCADS nicht größer X'2FFF' (12K) sein.

```
CPGSnn  PREVIOUSLY DEFINED
```

Es wurden mehr als 99 Subroutinen (BEGSR) in einem Programm codiert.

Fehlermeldungen bei der Ausführung 6950

Bei der Ausführung von CPG-Programmen kann am Bildschirm folgende Fehlermeldung auftreten:

```
**************************************************************************
SYSTEM-FEHLER                CPG 2.1 CLE        TT01  TST001    0002A0
**************************************************************************

DATEI xxxxxx  NICHT ERÖFFNET

VERSTÄNDIGEN SIE BITTE IHR RECHENZENTRUM

**************************************************************************
PF1 ===> ABEND   +   PF3 ===> IGNORE   +   SONST ===> ENDE
**************************************************************************
```

Durch Betätigen der PF1-Taste wird das Programm abgebrochen und ein CICS-Dump erzeugt. Durch Betätigen der PF3-Taste erfolgt ein Rücksprung ins Anwendungsprogramm.

Diese Auswahl kann bei der CPG-Installation eingeschränkt werden. Beim Betätigen einer anderen Funktionstaste wird das Programm ohne Dump beendet. Über die Kundenkonfiguration kann gesteuert werden, ob jedesmal ein Dump und abnormales Ende des Programms gesetzt wird. (Wegen DTB). Der Dump-Code und die Fehlermeldung am Bildschirm geben Auskunft über die Fehlerursache. Folgende Fehlermeldungen können am Bildschirm auftreten.:

DATEI XXXXX NICHT ERÖFFNET:

Die Datei wurde für die Online-Verarbeitung nicht eröffnet. Ein CICS-Dump ist bei dieser Fehlermeldung nicht möglich.

DATEI XXXXX NICHT IN DER FCT:

Es wurde eine I/O-Operation mit einer Datei durchgeführt, die in der CICS-FCT nicht definiert ist.

DATEI XXXXX NICHT VERFÜGBAR:

Es wurde ein Update versucht, ohne vorher ein CHAIN für diesen Satz durchgeführt zu haben. (Put no Get).
Es wurde versucht, einen nicht existierenden Satz mit Update zu verändern.
Bei CICS 1.6 kann diese Meldung auch erscheinen, wenn bei einem NEWRC kein vorheriges CHAIN durchgeführt wurde.
Die Dateikomponente zwischen Define Cluster und CICSFCT stimmt nicht überein.
Der Service für diesen Zugriff ist im CICS FCP nicht generiert. (Reg 9 = X'00') VSAM und ISAM im FCP nicht unterstützt.

Der Service für diesen Zugriff ist in der CICS FCT nicht generiert. (Reg 9 ungleich X'00' VSAM Delete und Delete nicht als SERVREQ aufgeführt).

Es wurde ein SYNCP gegeben, ohne vorher die Dateien mit RNDOM freizugeben.

DATEI XXXXX EINGABEFEHLER:

Das System kann auf die entsprechende Einheit nicht zugreifen. (Hardware-Defekt).
Bei einer Druckausgabe wurde versucht, einen Satz auszugeben, dessen Länge größer ist als die Satzlänge des VSAM Clusters DFHNTRA.

DATEI XXXXX BEREICH ZU KLEIN:

Beim Hinzufügen von Datensätzen wurde kein freier Platten-speicher mehr gefunden.
Es kann keine Druckausgabe mehr durchgeführt werden, es ist kein freier TRANSDT-Bereich mehr verfügbar.

DATEI XXXXX SATZ NICHT VORHANDEN:

Bei dem Service DELET wurde der angegeben Schlüssel nicht mehr gefunden.

DATEI XXXXX VSAM ERROR ILLOGIC:

Diese Fehlermeldung erscheint, wenn ein VSAM-Fehler auftritt, der keinem anderen Fehlercode entspricht.
Der VSAM-Error und Return-Code wird am Bildschirm mit angezeigt oder kann im CICS-Transaction-Dump im Transaction Storage File (Länge = D0) auf Displacement 2C nachgesehen werden. (z.B. 0008006C = Returncode 8 und Errorcode 6c).
Für Alternativindex-Verarbeitung traten zuviele gleiche Keys auf, so dass die Satzlänge des VSAM-AIX zu klein war.
VSAM Error/Return-Codes können in der IBM Broschüre VSE/VSAM Messages und Codes (SC 24-5146) nachgesehen werden.

ESA Error:

Aus einem ESA-Programm wurde eine EXHM-Operation für ein HL1-Modul ausgeführt, welches nicht mit HL1 größer 1.5 umgewandelt wurde.

Ein HL1-Modul, das in der CICS-PPT eingetragen ist und zufällig den Status DISABLE hat, wurde mit EXHM aufgerufen.

HL1 Fehler Datei: XXXXXXX HMModul

Beim EXHM wird festgestellt, dass ein Modul aufgerufen wird, das nicht nach der Dataset-Logik arbeitet und das Dateien enthält, die im übergeordneten Hauptprogramm nicht definiert sind.

Der Fehler wird behoben, indem man in der H-Karte in Spalte 34 ein Kennzeichen für Dataset-Verarbeitung einträgt (z.B. D, S).

HL1-Pool Fehler:

Eine EXHM-Operation konnte nicht ordnungsgemäß ausgeführt werden.

Bei einer EXHM-Operation für ein HL1-Modul wurde festgestellt, dass der HL1-Pool voll ist oder die HL1 Library nicht vorhanden ist.

Bei einer EXHM-Operation ist ein anderes Modul schon auf dem

vorgesehenen Tabellenplatz gespeichert.

INDEX FALSCH:

Es wurde eine Operation ausgeführt mit variablem Index.
Der Inhalt des Indexfeldes ist entweder Null oder größer als die Anzahl der definierten Elemente der entsprechenden Feldgruppe, bzw. Bei SETIX oder SETIN ist der Index zu groß.

SQRT NEGATIV:

Bei der Operation SQRT wurde ein negativer Feldinhalt vorgefunden.

READI FEHLER:

Es wurde ein READI ohne vorhergehendes READ codiert.

DIVISIONS FEHLER:

Bei der Operation DIV wurden nicht die Regeln für die Feldgröße eingehalten.

DIVISION DURCH 0:

Bei der Operation DIV hatte der Divisor (Faktor 2) den Feldinhalt Null.

DATEI XXXXXXXX PROGRAMMFEHLER ERROR CODE

Es soll eine Operationsfolge (Dateizugriff) durchgeführt werden, die der TP-Monitor untersagt. Folgende Operationen dürfen nicht hintereinander für die gleiche Datei ausgeführt werden:

Error Code	Operation		Auszuführende Operation	
SEQ EF	Nach End-of-File	darf kein	READ	durchgef. werden
SEQ FU	Nach einem READ	"	CHAIN U	"
SEQ FU	" " READ	"	UPDATE	"
SEQ FA	" " READ	"	ADD	"
SEQ FL	" " READ	"	DELET	"
SEQ EF	" " READ	END-OF-FILE	READ	"
UPD FC	" " CHAIN U	"	CHAIN	"
UPD FR	" " CHAIN U	"	READ	"
UPD FB	" " CHAIN U	"	READB	"
UPD FU	" " CHAIN U	"	CHAIN U	"
UPD FA	" " CHAIN U	(nicht NF)	ADD	"

Sollten in Ihrem Hause diese Regeln nicht eingehalten worden sein, so kann für eine Übergangszeit diese Fehlermeldung unterdrückt werden. Siehe Abschnitt 5125.
In diesem Fall wird eine entsprechende Fehlermeldung auf der Systemkonsole ausgegeben und die Programme sollten schnellstens umgestellt werden.

Beispiel einer solchen Fehlermeldung auf der Konsole:

```
-------------------------------------------------------------
F2 002 *** FILE  CPGWRK  PROGRAM-ERROR     UPD FU
F2 002 *** TRANSID TT02  PROGRAM TST002   TERM RZR1    ****
-------------------------------------------------------------
```

TSQUE XXXXXXXX NNN TCA RCODE:

Bei der Ausgabe einer Temporary Storage Queue traten Fehler auf.
XXXXXXXX gibt den Namen des Storage an, auf den die Ausgabe Ausgeführt werden sollte.

NNN gibt den Temporary-Storage-Return-Code (dezimal) an.
(Nähere Einzelheiten siehe CICS-Broschüre DFHTS TYPE=PUTQ).

TOTAL FILE ERROR

Unter TOTAL-Rechenbestimmungen wurde ein Dateizugriff durchgeführt zu einer Datei, in der Gruppenstufenschalter eingetragen sind.

QLF LIST FEHLER XXXXXXXX

Bei der Operation LIST wurde ein Dokument angegeben, das nicht in der LIST-Library im QTF gespeichert ist.

Datei CPGWRK nicht eröffnet/nicht in der FCT

Es wurde versucht, eine Map-Ausgabe durchzuführen, und die Datei CPGWRK ist nicht verfügbar.

Entscheidungstabellen-Verarbeitung

Wenn immer nur die erste Aktion durchgeführt wird, ist eventuell eine Nummerierung im CPG-Programm ab Spalte 73 vorgenommen worden.

TWALENGTH: xxx/ yyy

Beim quasi-transaktionsorientierten Programmieren wurde festgestellt, dass die TWA-Länge yyy nicht der Länge der geretteten TWA xxx entspricht. Gesteuert über CPGURSI2, Stelle 7.

Der Dump-Code aus dem CICS-Dump gibt über die Fehlerart Auskunft. Folgende Codes können auftreten:

'ASRB' Es wurde versucht, mit EXHM einen HL1-Baustein aufzurufen, der als Phase nicht existiert. Es können Speicherverletzungen auf-auftreten.

'CPGL' Das Command-Level-Interface ist nicht geladen.

'ECPF' variables GOTO und Label nicht gefunden.

'ECPG' Allgemeine CPG-Fehlermeldungen

'ECPL' variables GOTO und falscher Subroutinen-Level.

'ECPR' Fehler bei QPG (im Service Level CPG5)

'ECPS' Storage Control. Beim Initialisieren einer Task war zu wenig Speicher verfügbar.

'ECPT' Terminal Control. Bei einer Bildschirmein- oder -ausgabe war kein Speicher verfügbar.

'ECPW' Die Eintragung TWA-Size in der CICS-PCT wurde zu klein eingetragen.

Der Datenkanal im HL1-Modul wurde zu klein definiert.

'ECPX' EXHM length error. Die PWA eines Dataset-S Moduls wurde durch ein HL1-Newcopy vergrößert, während das Modul online im Zugriff war.

'NDL1' DL/I-Anwender. In Spalte 48 der H-Karte ist 'D' nicht eingetragen.

'NAME' Falscher Modulname gefunden.

'NESA' Das HL1-Modul ist nicht ESA-fähig.

'NFND' Das HL1-Modul wurde beim EXHM nicht gefunden.

'NLIB' Die HL1-Library wurde nicht gefunden.

'POOL' Der HL1-Pool ist voll.

'QSFF' Die Datei QSFLIB ist nicht eröffnet.

'QSFI' Eine QSF-Map wurde nicht gefunden.

'QSFM' Fehler in der Directory oder Phase nicht gefunden.

'QSFS' Es liegt ein Size-Error vor.

'QSFT' In einer Non-Terminal-Task wurde ein MAP-Befehl abgesetzt. Der Abend-Code ist in der Statistik zu sehen.

'QSFV' Die Directory ist voll.

'QSFZ' Es sind zu viele Felder in einer Map.

Fehlermeldungen bei der Ausführung im Batch 6960

Durch die Eintragung 'I' in Spalte 46 der H-Karte haben Sie die Möglichkeit, bei einem Programmfehler (Cancel) eine Programmunterbrechung zu erreichen. Auf der Systemkonsole wird z. B. folgende Meldung angezeigt:

```
F4 004 *** DIVIDE BY ZERO    IN STATEMENT     4
F4 004 ===> C=CANCEL, E=EOJ
```

Der Operator hat nun die Möglichkeit, nachfolgende Jobs zu stoppen. Die Programmunterbrechung wird durch die Eingabe 'C' oder 'E' aufgehoben.

Bei der Eingabe 'E' wird das Programm an der entsprechenden Stelle beendet, bei 'C' wird das Programm durch Cancel abgebrochen. In beiden Fällen wird ein Dump aufgelistet.

Bei einem Programmabbruch wird im Batch ein formatierter Dump ausgedruckt. Hierdurch wird die Fehlerdiagnostik wesentlich erleichtert. Der Dump ist im wesentlichen so aufgebaut wie ein CICS-Dump.

*** VSAM ERROR: RC=XX EC=XXX ***

Der Returncode RC und der Errorcode EC geben Aufschluss über den aufgetretenen Fehler. Return- und Errorcodes sind in der Broschüre 'VSAM Messages and Codes' aufgeführt.

*** ERROR GET/FREEVIS: EX=XX ***

In diesem Fall ist entweder die Partition zu klein, d.h. das System kann keinen Speicher mehr zur Verfügung stellen, oder der Parameter SIZE=AUTO fehlt bei der Ausführung.

*** FEHLER BEI CDLOAD: EC=XX ***

Ein Programm konnte nicht geladen werden, z.B.weil es nicht gefunden wurde oder nicht verfügbar war. Unter Umständen ist auch die Partition zu klein.

DELET OHNE VORH. GET

Es wurde versucht, einen Satz zu löschen, ohne dass vorher ein CHAIN oder ein READ durchgeführt wurde. Der Fehler kann natuerlich auch auftreten, wenn der Satz beim vorhergehenden CHAIN nicht gefunden wurde.

ZU VIELE DATEIEN OPEN

Es können maximal 62 Dateien gleichzeitig geöffnet sein. Dabei sind auch interne Dateien zu berücksichtigen, die z.B. bei der LIST-Operation benötigt werden.

PROGRAM CHECK INTERRUPTION

Dies ist ein allgemeiner Programmabbruch. Die Cancel-Adresse und der Abbruchcode werden im Dump aufgelistet. Die Analyse entspricht der Analyse eines CICS-Dumps. Die Abbruchcodes, z.B. 0007 für Data Exception, sind im 'System Reference Summary' aufgelistet.

INVALID OPCODE

Mit diesem Fehler werden Operationen angegeben, die im Batch nicht unterstützt sind.

STORAGE ERROR

Dieser interne Fehler tritt auf, wenn kein Speicher mehr zur Verfügung steht, z. B. wenn zu viele Module aufgerufen wurden. Es sind max. 255 Storages unterstützt.

TEMP STORAGE ERROR

Es sind maximal 63 Temporary-Storage-Einzelsätze unterstützt.

TS QUEUEING ERROR

Es sind maximal 63 Temporary-Storage-Queues unterstützt. Dabei sind auch die internen Queues zu berücksichtigen, die z.B. bei der LIST-Operation benötigt werden. Pro Queue sind maximal 32000 Sätze zu-

lässig, insgesamt darf die Queue nicht mehr als 2 MB Daten beinhalten. Der Fehler tritt ebenfalls auf, wenn versucht wurde, mit einer Satznummer kleiner oder gleich 0 auf die TS-Queue zuzugreifen.

INDEX ERROR

Es wurde eine Operation mit variablem Index ausgeführt. Dabei war der Inhalt des Indexfeldes entweder kleiner oder gleich Null oder größer als die Anzahl der definierten Elemente der entsprechenden Feldgruppe.

DIVIDE BY ZERO

Bei einer Division hatte der Divisor (Faktor2) den Feldinhalt Null.

DIVISION ERROR

Bei einer Division wurden die Regeln für die Feldgröße nicht eingehalten.

SQUAREROOT MINUS

Bei der Operation SQRT wurde ein negativer Feldinhalt vorgefunden.

ERROR IN TABLE XXXX

Dieser Fehler tritt auf, wenn die genannte Tabelle nicht verhanden oder nicht verfügbar (z. B. nicht geladen) ist.

ERROR IN LIST DOKUMENT: XXXXXXXX

Das angegebene Dokument ist nicht verfügbar, z. B. da es mit einem Passwort versehen wurde. Eventuell liegt das Dokument nicht in der LIST-Library, jede andere Library muss den Benutzer *LA zugewiesen bekommen. Wurde eine LIST-Phase angesprochen, so wurde u. U. die entsprechende Phase nicht gefunden. Ursache für diese Fehlermeldung kann ebenfalls sein, dass die Printer Tabelle voll ist. Folgende Maximalwerte sind einzuhalten: 50 List-Dokumente oder 200 Printer-Einträge im CICS, 20 im Batch (siehe QTF Handbuch S. 5300).

ERROR 2ND FILE ACCCESS TOTAL CALCS

Dieser Fehler tritt bei der Gruppenstufenverarbeitung auf, wenn versucht wurde, bei den Total-Bestimmungen einen weiteren Lesezugriff auf die gleiche Datei durchzuführen.

FEHLER BEI OPEN/CLOSE

Die Datei, auf die das OPEN bzw. CLOSE durchgeführt werden sollte, ist nicht verfügbar.

Fehlermeldungen bei der Ausführung 6970

Bei der Ausführung von Online-Programmen unter dem TP-Monitor CICS können folgende Fehlermeldungen auftreten:

CICS-Absturz

Ursache:
1. TWA-Size Angabe in der PCT ist kleiner als die TWA - Größe des Anwendungsprogramms.
   Diese Ursache wird ab diesem Release vom CPG abgeprüft. Mit Release 1.2 umgewandelten Programmen ist diese Absturzursache ausgeschaltet. (Bei Methodenbankversion Macro Level oder Command Level).
2. Storage Violation (siehe Speicherverletzung).

Behebung:
1. Dump ausdrucken. Die Transaction mit mehr als 20 Seiten im Dump ist der Verursacher. Nach dem Restart muss das Programm sofort mit DISAB abgehängt werden.

Speicherverletzung

Ursache:
1. TWA-Size zu klein (siehe CICS-Absturz).
   (Wird bei Programmen die ab Release 1.2 umgewandelt wurden, abgeprüft).
2. Die Operationen BEGAS und ENDAS wurden benutzt und ein Assembler-Statement überträgt Daten zu einer Adresse außerhalb des Programms.
3. Die Operation COPY wurde benutzt und ein Assembler-Statement überträgt Daten zu einer Adresse außerhalb des Programms.
4. Satzlänge in der Dateizuordnung für ein Dataset ist zu klein.
5. Bei Temporary Storage haben zwei Segmente verschiedener Länge den gleichen Namen.
6. Assembler-Fehlermeldungen im CPG-Programm wurden nicht beachtet.
7. Vor einem SYNCP-Befehl wurden Dateien nicht ordnungsgemäß mit RNDOM freigegeben.
8. Bei der Satzlänge für VBOMP wurde der Praefix nicht mitgerechnet.
9. Bei der ADD-Funktion wurde zweimal die gleiche Datei mit EXCPT angesprochen.
10. In der Ausgabe sind unter einer EXCPT Zeile 2 mal DIAG mit Bezugszahlen verwendet worden.

Behebung: Im CICS-DUMP 'DFHCSA' aufsuchen. Ab 'Storage +1' plus 4C steht die Adresse der letzten aktiven Transaktion.

CICS-Loop

Ursache:
1. Die Druckausgabe im CPG-Programm hat keine positive Ausgabezeile.
2. Zwei EXCPT-Zeilen sind ohne Ausgabefeld hintereinander codiert worden.
3. VSAM-Datei-Fehler (Task wurde beim CA-Split hart abgebrochen.
4. Nach einem READ für eine VSAM-Datei wurde die Bedingung 'EF' nicht abgefragt.

Behebung: CICS canceln und Dump ausdrucken. Bei Dateifehler muss Forward-Recovery gefahren werden.
Bei Task-Fehler muss nach dem Restart das Programm mit DISAB sofort abgehängt werden.

CICS Data Base Loop

Ursache: Eine VSAM-Datei mit SHR 3 wurde im Batch bzw. online zerstört.

Behebung: CICS canceln. Im Register 1, in der Partition SAVE -AREA, steht die Adresse der defekten VSAM-Datei (ACB).
Online-VSAM-Dateien mit SHR 3 ermitteln:

In der FCT haben die Dateien den Eintrag 'ACCMETH= (VSAM, KSDS,KEY)'. Auf die entsprechende Datei Verify fahren.

Bei erfolglosem Verify muss Delete/Define Cluster gefahren werden.

CICS Terminal Dataset Loop

Ursache: Lokale Steuereinheiten waren bei CICS-Start nicht ready.

Behebung: CICS canceln. IPL für VSE durchführen. Lokale Steuereinheiten einschalten. CICS neu starten.

Transaktionen blockieren

Ursache: Eine Update-Datei ist für andere Tasks gesperrt.

Für ein dialogorientiertes Programm mit DTB = YES in der PCT, ist der Befehl 'SYNCP' falsch bzw. nicht codiert worden.

Behebung:
1. CSMT Term,Outserv,all
2. Dump-Bereich switchen
3. CSMT Term,inserv,all
4. Dump Bereich ausdrucken

Für die blockierten Tasks die entsprechenden Dumps dem Anwendungsprogrammierer übergeben.
Der entsprechende Dateiname befindet sich im Task Dump 'TCA USER' ab Stelle X'84'.

VSAM Illogic Error

Ursache:
1. Pseudo-Satz fehlte bei ADD-Datei.
2. Delete/Define war fehlerhaft.
3. Bei einer VSAM-Datei mit variabler Satzlänge wurden die vier Byte Verschiebung vergessen.
4. Bei einer ESDS- oder RRDS-Datei wurde nicht die richtige RBA gefunden.

Behebung: Der genaue VSAM-Error-Code wird am Bildschirm angezeigt.

CICS arbeitet ungewöhnlich langsam

Ursache:
1. VSIZE der CICS Partition ist zu klein.
2. In der PCT wurde DTB = YES angegeben und im CPG-Programm wurde kein 'SYNCP' gegeben.
3. AMXT in der SIT ist zu klein angelegt.
4. Im CPG-Programm werden auf eine oder mehrere Dateien (VBOMP) sehr viele Zugriffe gestartet.
5. Priorität von CICS ist falsch vergeben worden.

Behebung:
1. VSIZE vergrößern.
2. CICS canceln und Dump ausdrucken. CICS-Dump auf belegte Speicherplaetze im DTB-Bereich untersuchen. Nach dem Restart das ermittelte Programm mit DISAB abhängen.
3. Mit 'CSMT AMX' die Anzahl der aktiven Tasks erhöhen. Anschließend DFHSIT mit neuer AMX umwandeln.
4. CICS-Statistik ausdrucken. Im Abschnitt File Statistik die aufgeführten Dateien überprüfen.
5. Die Priorität für CICS muss vom Operator hochgesetzt werden.

Task-Abbruch mit ASRA

Ursache:
1. Falsche Ausgabe-Schablone.
2. Gepackte Daten ohne 'P' in 43 eingelesen.
3. Ungepackte Daten mit 'P' in 43 eingelesen.
4. Ein numerisches Feld enthält keine gepackten Daten.
5. Fehler in einer Assembler-Unterroutine.
6. Durch Speicherverletzung ist die Task zerstört worden
7. Ein CPG-Programm wurde mit XCTL aus einem Assembler-Programm aufgerufen, dabei wurde vom Assembler-Programm

vor dem XCTL die TWA nicht initialisiert.

8. Wenn die Ursache bei einem EXPR liegt und eine CSA-Verschiebung vorliegt, so ist die Verschiebung nicht im CPGUCCBA und somit nicht in den CICS-Interfaces oder nicht in der CPGURSIT und somit nicht in CPGWRK.

Behebung:

1 bis 5. Dump ausdrucken, fehlerhaftes Feld suchen und Fehlerursache beheben.

6. Dump ausdrucken und Speicherverletzer feststellen. Den Speicherverletzter mit DISAB abhängen. Task mit 'CSMT NEW,PGRMID= ' laden.

7. Vor XCTL ein XC 256(116,12),256(12) legen (bei normaler CPG-TWA), sonst Verschiebung (z. B. bei Datasets) entsprechend berücksichtigen.

Task-Abbruch mit AICA

Ursache:

1. Programmloop. Eine GOTO-Operation verzweigt zu einem vorher liegenden TAG und die Schleife wird nicht durch ein READ Bildschirm oder ein WAIT unterbrochen.

2. Im CPG-Programm wurde in einer Subroutine auf BEGSR ENDSR mit GOTO gesprungen.

3. Eine Rechenroutine wird ohne Ein-/Ausgabe-Unterbrechung so oft durchlaufen, dass die vom CICS festgelegte Maximalzeit erreicht wird.

4. Innerhalb einer DO-Schleife wurde die Operation CLEAR verwendet.

5. Bei einer DOW-Operation wurde versehentlich eine Bezugszahl in Spalte 54-55 eingetragen.

Behebung:

1. Schleife entfernen oder unterbrechen.

2. Für GOTO in der Subroutine muss eine TAG-Zeile definiert werden.

3. Zeit im CICS höher setzen oder Rechenroutine durch ein 'WAIT' unterbrechen.

Task-Abbruch mit ASCF

1. In der Eingabe für Temp.Storage wurden Gruppenstufenschalter eingesetzt.

Task-Abbruch mit ATNI

1. Es wurde versucht, nicht darstellbare Zeichen auf dem Bildschirm auszugeben.

Task-Abbruch mit APCT

1. Ein HL1-Modul wurde nicht gefunden.

2. Das Programm ist disabled (z.B. wegen eines Newcopys, während das Programm aktiv war).

Task-Abbruch mit ECPW

1. Wenn der Datenkanal zu lang ist, erfolgt ein Abbruch im CICS mit dem Code ECPW.

Tabellen 7000

Übersicht der Rechenbestimmungen 7000

Die folgende Tabelle gibt eine Übersicht über die möglichen Eintragungen bei den einzelnen Rechenoperationen. Dabei bedeutet ein *, dass in diesem Feld eine Eintragung sein muss, ein O, dass in diesem Feld eine Eintragung sein kann. Ist die Spalte blank, so bedeutet dies, dass dieses Feld keine Eintragung enthalten darf.

Bedeutung der Abkürzungen: BED=Bedingungen, F1=Faktor 1, OP=Operationscode, F2=Faktor 2, ERG=Ergebnis, LG=Feldlänge, D=Dezimalstellen, H=Runden, EB=Ergebnisbezugszahlen.

Formulareintragungen:

BED	F1	OP	F2	ERG	LG	D	H	EB
OOO	O	ADD	*	*	O	O	O	OOO
OOO	O	+	*	*	O	O	O	OOO
OOO		AFOOT	*	*	O	O	O	OOO
		BEGAS						
	*	BEGDT						
	*	BEGSR						
OOO		BITON	*	*	O			
OOO		BITOF	*	*	O			
OOO		BREAK				O		
OOO	*	CABxx	*	*			OOO	
		CALL	*	O				
OOO	*	CALLD	*			O		
OOO		CALLM	*	O				
OOO	O	CASxx	O	*			OOO	
OOO	*	CBSxx	*	*			OOO	
OOO	*	CHAIN	*	O		O	O	
OOO	*	CHANG	*					
OOO		CHECK	*			O	OO	
OOO		CLEAR						
OOO		CLOSE	*			O	O	
OOO		CLRIN		*	O	O	O	*
OOO	*	COMP	*	O		O	OOO	
OOO		COMRG		*	O			
OOO		CONT						
OOO		CONVT	O	*		O		
		COPY	*					
OOO		DEBUG						
OOO		DELC	*	*	O			
OOO	O	DELET	*					
	*	DEQ				O		
OOO	O	DIV	*	*	O	O	OOO	
OOO	O	/	*	*	O	O	OOO	
OOO	*	DLI	O	*	*	O	*	
OOO	O	DO	O	O			OOO	
OOO	*	DOU	*			O	OOO	

BED	F1	OP	F2	ERG	LG	D	H	EB
OOO	*	DOW	*					OOO
OOO	O	DSPLY		O				
OOO		DUMP	O					
OOO		EDIT	O	*	O		O	OOO
OOO		ELIM	*	*	O			
		ELSE						
OOO		END	O					
		ENDAS						
OOO		ENDDO	O					
		ENDDT						
		ENDIF						
	O	ENDSR						
		ENQ	*			O		
OOO		EREAD	*			O		OOO
OOO		EXCPT	O			O		OOO
OOO		EXITD		*				
OOO		EXITP	O	O		O		
OOO	O	EXITI	*	O		O		
OOO		EXITT	O	O				
OOO	O	EXPR	O	O		O		
OOO		EXSR	*					OOO
OOO		FILL	*	*	O			
OOO	*	FIND	*					O
OOO		GOTO	*					
OOO	*	IF	*			O		
OOO		IFC	O	O				OOO
OOO		INDOF						OO
OOO		INDON						OO
OOO		JRB		*	O		O	
OOO		JRC	*	*	O			
OOO		JRZ		*	O			
OOO	O	LIST	*	O				
OOO		LOADT	*	O		O		
OOO	*	LOKUP	*					OOO
OOO	O	MACRO	*					
OOO		MAP	O	O		O		
OOO		MAPD	O	O		O		
OOO		MAPI	O	O		O		
OOO		MAPO	O	O				
OOO	*	MAPP	O	O		O		
OOO		MLLZO	*	*	O			
OOO		MOVE	*	*	O			
OOO		MOVEA	*	*	O			
OOO		MOVEL	*	*	O	O		
OOO		MOVEN	*	*	O	O	O	OOO
OOO		MOVEV	*	*		O		
OOO	O	MULT	*	*	O	O	O	OOO
OOO	O	*	*	*	O	O	O	OOO
OOO		MVR		*	O	O		
OOO	*	NEWRC	*	*				
OOO		OPEN	*			O	O	
		PARM	*					
OOO		PRNT	*					
OOO		PROG	O	O				
OOO		PROT	*					
OOO		PURGE	*					
OOO	*	QSSA	*	*	*		O	*

BED	F1	OP	F2	ERG	LG	D	H	EB
OOO	O	READ	*	O			O	
OOO	O	READB	*	O				
OOO		READI	*				O	
OOO	O	READP	*	*				
OOO	O	REPLC	*	*	O			
OOO		RNDOM	*					
OOO		ROLL		*				
OOO		ROLLB		*				
OOO		SAVET	*	O				
OOO	*	SCAN	*	O			O	O
OOO		SDUMP	O					
OOO		SELCT	O	*	O		O	
OOO		SETIN		*	O	O	O	*
OOO		SETIX	O	*	O	O		O
OOO	*	SETLL	*					
OOO		SETOF						*OO
OOO		SETON						*OO
OOO		SORTA	*				O	
OOO		SQRT	*	*	O	O	O	
OOO	O	SUB	*	*	O	O	O	OOO
OOO	O	-	*	*	O	O	O	OOO
OOO		SYNCP					O	
	*	TAG						
OOO		TESTB	*	*	O			*OO
OOO		TESTN	O	*	O		O	*OO
OOO		TESTT	*					*OO
OOO		TIME		O	O	O		OOO
OOO		TWALD	*	O				
OOO		TWASV	*	O				
OOO		UCTRN	*				O	
OOO	*	UPDAT	*	*				
OOO	*	USSA					O	
OOO	O	VBOMP	*	*				
OOO		VSLCT	*					
OOO		WAIT		O				
OOO	*	WRITE	*	*				
OOO		XFOOT	*	*	O	O	O	OOO
OOO		Z-ADD	*	*	O	O	O	OOO
OOO		=	*	*	O	O	O	OOO
OOO		Z-SUB	*	*	O	O	O	OOO

Die Operationscodes von CPG im Überblick 7005

Arithmetische Operationen

ADD	+
DIV	/
MULT	*
MVR	
SQRT	
SUB	-
Z-ADD	=
Z-SUB	

Übertragungsoperationen

CONVT		
MOVE	MOVEL	
MOVEN		
MOVEV		
MLLZO		
CHANG		
CLEAR		
DELC	ELIM	REPLC
EDIT		
FILL		
JRB	JRC	JRZ
SELCT		

Vergleichsoperationen und Bitabfrageoperationen

COMP		
TESTB	TESTN	TESTT
CABxx	CASxx	CBSxx

Verzweigungs- und Programmverknüpfungsoperationen

GOTO	TAG		
EXITD	EXITI	EXITP	EXITT
EXPR	PROG		
EXSR	BEGSR	ENDSR	
IF	ELSE	ENDIF	END
DO	DOUxx	DOWxx	END
	BREAK	CONT	ENDDO
IFC			
CALLD	CALLM	PARM	
CASxx	CABxx	CBSxx	

Feldgruppenoperationen

AFOOT	XFOOT
LOKUP	
MOVEA	
ROLL	ROLLB
SCAN	
SORTA	

Operationen für Setzen von Bezugszahlen/Anzeigern, Indizes und Bits

BITOF	BITON
CLRIN	
INDOF	INDON
SETOF	SETON
SETIN	SETIX
TESTN	

Entscheidungstabellenoperationen

BEGDT	ENDDT
HIGH	LOW
EQUAL	=
>	<
EXCPT	EXSR
GOTO	

Datei- / Ein-Ausgabe-Operationen

CHAIN				
CHECK				
CLOSE	OPEN			
DELET	NEWRC	UPDAT	WRITE	
DLI	QSSA	USSA		
EXCPT	FIND			
LIST	LOADT			
MAP	MAPD	MAPI	MAPO	MAPP
READ	READB	READI	READP	
RNDOM				
SAVET				
SETLL				
VBOMP	VSLCT			

Diagnoseoperationen

DEBUG
DUMP
SDUMP

TP-bezogene Operationen

COMRG	
ENQ	DEQ
SYNCP	

Verschiedene Operationen

BEGAS	ENDAS
COPY	
DSPLY	
FIND	
MACRO	
PRNT	
TIME	
TWALD	TWASV
UCTRN	
WAIT	

Distributed Transaction Processing

ALLOC	
CNVAS	
CONCT	
EXTRT	
FREE	

CPG-Bildschirmattribute 7020

Die folgende Tabelle zeigt die Feldeigenschaften in Abhängigkeit von den Aufbereitungsschlüsseln.

CPG Attribut	CICS Hex	Geschuetzt	Anzeiger D:Doppelhell :Normalhell	Nur numerisch	Lichtstift	MDT gesetzt	Farbe Ter	Farbe Dru
U	40	N					G	
K	C1	N				J		Sch
Q	C4	N			J		r	
B	C5	N			J	J		hwa
							u	
W	50	N		J				rz
E	D1	N		J		J	e	
J	D4	N		J	J			
G	D5	N		J	J	J	n	
A	C8	N	D		J			
C	C9	N	D		J	J	R	
N	D8	N	D	J	J		o	
H	D9	N	D	J	J	J	t	
X	4C	N	Keine					
D	4D	N	Keine			J		
Y	5C	N	Keine	J				
Z	5D	N	Keine	J		J		
0	60	J						
1	61	J				J	B	
2	E4	J			J			
3	E5	J			J	J	l	
	F0	J					a	
R	F1	J				J		
I	F4	J			J		u	
O	F5	J			J	J		
4	E8	J	D		J		W	Gru
5	E9	J	D		J	J	e	r
P	F8	J	D		J		i	u
L	F8	J	D		J		s	e
M	F9	J	D		J	J	s	n
6	6C	J	Keine					
7	6D	J	Keine			J		
8	7C	J	Keine					
T	7D	J	Keine			J		

CPG-Ausgabe Write Control Character 7030

Folgende Tabelle zeigt die WCC (Write Control Character) Eintragungen, die bei einer Bildschirmausgabe in Spalte 18 der Satzbestimmung eingetragen werden können.

CPG Eintragung	WCC Hex	Hupe	MDT-On	Reset Keybord
K	C0		Y	
L	C1			
M	C2		Y	Y
	C3			Y
N	C4	Y	Y	
O	C5	Y		
S	C6	Y	Y	Y
H	C7	Y		Y

Maximalwerte bei einem CPG-Programm 7040

Standardmäßige maximale Programmgröße	24K
HL1-Baustein maximale Programmgröße	20K
HL1-Batch standardmäßige maximale Programmgröße	20K
HL1-Batch standardmäßige maximale TWA-Größe	8K
Standardmäßige maximale Größe von CPGTWA (wovon ca. 400 Bytes für CPG-intern und den TP-Monitor belegt werden)	4K
Standardäßige maximale Länge eines Eingabe- oder Ausgabesatzes (einschließlich eines ggf. vorhandenen Monitor-Präfixes)	8K
Maximale Größe der TIOA (Terminal-I/O-Area)	4080 Bytes
Maximale Länge eines alphanumerischen Feldes	256 Bytes
Maximale Länge eines numerischen Feldes (15 Ziffern)	8 Bytes
Anzahl Eintragungen in der Namentabelle	2000
Anzahl Eintragungen in der Langnamentabelle	500
maximale Anzahl Feldgruppen	100
maximale Anzahl Dateien und LIST-Operationen	100
maximale Anzahl Eintragungen in Datenstrukturen	1000
maximale Anzahl Strukturen	50
maximale Anz. Dateien in der Standard-File-Tabelle	100
maximale Anz. DO- oder IF-Operationen im Programm	999
maximale Anz. Subroutinen (BEGSR) im Programm	999
max. Verschachtelung von DO- oder IF-Operationen	40
maximale Anzahl von QSF-Maps	64

Eine maximale Programmgröße von 20K mit einer maximalen TWA-Größe von 8K kann alternativ angewendet werden. Das wird durch die Verwendung eines anderen Adressblocks erreicht. (Siehe H-Karte Spalte 48).

Die hier aufgeführte Programm- sowie TWA-Größe (z.B.12K) kann durch einen Eintrag in der H-Karte Spalte 32 beeinflusst werden.

Die maximale Eingabesatzlänge von 8K kann durch das Feld CPGFIS erweitert werden.

Zwischenspeicher von Daten für CICS-Anwender 7050

1. C W A Common Work Area 7050

Maximale Länge	:	3584
Speichermedium	:	Hauptspeicher
Gültigkeit	:	Systemweit
Definition	:	Systeminitialisierung
Update ?	:	Ja
Adressierung	:	EDIT/SELCT CPGCSA
Eignung	:	Für allgemeine Informationen, welche systemweit zur Verfügung stehen müssen.
Wichtig	:	Es sollten Belegungskonventionen angelegt werden. Die Daten werden nicht automatisch gelöscht.

2. T C T User Area 7055

Maximale Länge	:	255 Bytes, von CPG-Programm 245 Bytes.
Speichermedium	:	Hauptspeicher
Gültigkeit	:	Der Datenstation zugeordnet
Definition	:	In der TCT
Update ?	:	Ja
Adressierung	:	EDIT/SELCT CPGTCT
Eignung	:	Für Anwendungsinformationen, die terminalbezogen zwischengespeichert werden müssen und so auch von einer Task zu einer zu einem beliebigen Zeitpunkt später laufenden anderen Task übergeben werden können.
Wichtig	:	Daten werden nicht automatisch gelöscht.

3. Temporary Storage 7060

Maximale Länge : 8 K

Speichermedium	: Hauptspeicher oder 'Auxiliary' Subpool 5 oder VSAM-ESDS
Gültigkeit	: Abhängig von Namenskonventionen
Definition	: Vom ersten Benutzer, der Daten unter einem Namen speichert.
Update ?	: Ja
Eignung	: Zwischenzuspeichernde Informationen
Wichtig	: Ein bestehender TS-Satz darf bei einer Änderung nicht verlängert werden. Aus diesem Grund sollte ein Verzeichnis der einzelnen TS-Queues geführt werden.

Von CPG erzeugte Temporary Storage Bereiche 7070

TRAN TERM	bei TWASV, TWALD
TERM PCPG	bei MAPD taskorientiert, bei EXPR mit TWA SAVE

Erweiterung der Namen-Tabelle 7400

In einem CPG-Programm können maximal 2000 Felder, Anspringpunkte, Programmverbindungen usw. definiert werden.
Sollte die Anzahl 2000 überschritten werden, so wird die Fehlermeldung 'Namen Tabelle voll' angelistet. Im Bedarfsfall kann die Namen-Tabelle wie folgt erweitert werden:

1. Das Copy CPG*CNTB muss aus der SL abgestanzt werden und um einen Eintrag DS CL4080 (entspricht weitere 340 Einträge) ergänzt werden.

2. Copy in SL neu katalogisieren.

3.
```
// JOB CPGNTAB
// OPTION DECK
// EXEC ASSEMBLY
     COPY   CPG*CNTB
     END
/*
/&
```

Die bei der Umwandlung gestanzten Karten in die RL katalogisieren.

4.
```
// JOB CPGLINK
// OPTION CATAL
   PHASE CPG,*
   INCLUDE CPG*             * = Release Suffix
/*
// EXEC LNKEDT
/&
```

Erweiterung der Schlüsselwort-Tabelle (CPG2) 7450

Unabhängig von den übrigen Schlüsselworten kann der Benutzer jederzeit eigene Schlüsselworte generieren (Copy CPGUCKWT). Bei der Erstinstallation werden Mustereintragungen mitgeliefert, die nach Bedarf geändert bzw. ergänzt werden können.

Im Bedarfsfalle kann die Schlüsselwort-Tabelle wie folgt erweitert werden:

1. Das Copy CPGUCKWT muss aus der SL abgestanzt und geändert werden.

   Aufbau einer Eintragung:
   z.B.
```
DC    C'SETZE      SETON,X'11'
        .          .       .
        .          .       ...Hex Wert der jeweiligen CPG-Oper-
        .          .              ation Copy CPG*RKWT entnehmen
        .          ...CPG Operation
        ...frei wählbares Schlüsselwort max. 10 Stellen
```

2. Copy in SL neu katalogisieren.

3.
```
// JOB CPGKWT
// OPTION DECK
// EXEC ASSEMBLY
        COPY  CPG*RKWT              * = Release Suffix
        END
/*
/&
```

Die bei der Umwandlung gestanzten Karten in die RL katalogisieren.

4.
```
// JOB CPGLINK
// OPTION CATAL
   PHASE  CPG*NX,*
   INCLUDE  CPG*NX                  * = Release Suffix
/*
// EXEC LNKEDT
/&
```

Wenn eigene Schlüsselworte generiert werden, so ist bei einem CPG-Releasewechsel nach der Bandinstallation Punkt 3 und 4 durchzuführen.

Übersetzungstabelle CPGUCUTR 7460

Bei der Bildschirmeingabe sowie bei der Operation CONVT kann eine Übersetzung von Klein- in Großbuchstaben aktiviert werden. Das Copy-Buch CPGUCUTR enthält eine Übersetzungstabelle, aus der abgeleitet werden kann, wie die Übersetzung durchgeführt wird. Sollte es erforderlich sein, so kann die Tabelle wie folgt geändert werden:

1. Das Copy CPGUCUTR aus der SL abstanzen und ändern.

2. Copy in SL neu katalogisieren.
3.
```
// JOB CPGUCUTR
// OPTION DECK
// EXEC ASSEMBLY
     COPY   CPGUCUTR
     END
/*
/&
```

   Die bei der Umwandlung gestanzten Karten in die RL katalogisieren.

4. Die Phase CPGMBK neu linken.

Job-Control für Option-Deck-Lauf 7510

Wenn ein Option-Deck-Lauf mit anschließendem MAINT durchgeführt werden soll, so kann folgender Job verwendet werden. Nach der Assembly werden die generierten Object Statements in die Power Reader Queue abgestellt.

Ein Musterprogramm wird in dem SL-Copy-Buch A.CPG*ZCCI mitgeliefert.

```
   * $$ JOB CPGDECK,,,A
   * $$ PRT CLASS=X
   * $$ PUN DISP=I,CLASS=A
   // JOB CPGDECK
   // OPTION DECK
   // EXEC ASSEMBLY
            GBLC  &CPGTYPE,&CPGCICS
            CPGGEN CICS=17              <---- z.B. für CICS 1.7
            PUNCH '* $$ JOB CPGMAINT,,,A'
            PUNCH '* $$ PRT CLASS=X'
            PUNCH '// JOB CPGMAINT'
            PUNCH '// EXEC PROC=CPGINST1'
            COPY  CPG*MCCI              <---- z.B. für CCI
            END
   /*
   // EXEC ASSEMBLY
            PUNCH '/&&'
            PUNCH '* $$ EOJ'
            END
   /&
   * $$ EOJ
```

Online-Programm zum Löschen von Temporary-Storage-Bereichen 7510

Programmgesteuert können Gruppen von temporären Speicherbereichen mit dem HL1-Modul HQDELS gelöscht werden. Dazu wird über einen Datenkanal mit dem Modul kommuniziert.

```
IHQDELS  HS    DD
I*   1  2  oc.              * Operationscode
I*   3  4  rc.              * Returncode
I*   5  8  p1bis4.          * Position 1 bis Position 4 des TS-Namens

C
C*                         * alle TS mit Namen TESTxxxx löschen
C                    MOVEL'L '     OC           * Op-Code 'Löschen'
C                    MOVEL'TEST'   P1BIS4.
C                    EXHM HQDELS   HQDELS    C  * Alle Storages, deren
C*                                              * Name mit TEST be-
C*                                              * ginnt,werden gelöscht
```

Anlegen einer Datenview 7530

In relationalen Datenbanken wird eine Datenview online erstellt und bleibt für die Zeit ihrer Verarbeitung im Hauptspeicher.

Um diese Verarbeitungsform zu simulieren, wird eine View per Programm, CPG3..Query-Programm oder mittels QTS erstellt und auf einer Datei abgestellt. Die View wird erst dann von dieser Datei in den Hauptspeicher geladen, wenn sie in einem Programm mit dem Befehl FIND angesprochen wird. Die geladene View bleibt dann bis zum Shut Down des TP-Monitors im Hauptspeicher. Eine View kann auch in einem Batchprogramm verarbeitet werden, sofern der verfügbare GETVIS-Bereich der Partition für die Verarbeitung ausreicht.

Vorgehensweise für Nur-CPG-Anwender 7531

Die View wird in diesem Fall in zwei Schritten angelegt: Zunächst wird die Struktur der Tabelle mit einem CPG-Programm beschrieben und ihre Elemente mit Werten gefüllt, dann wird mit dem Serviceprogramm CPGZCTB die View dokumentiert und auf die Datei CPGWKV abgestellt.

1. Schritt: Füllen der View mit einem CPG-Programm

Das folgende CPG-Programm zeigt beispielhaft das Füllen einer View. Vereinfachend wird hier unterstellt, dass die gesamte Tabelle über den Bildschirm (über die Map EINGABE) eingegeben wird.

```
FCPGWKV  U   V    4020 20KV       KSDS                    PROGNAME
D        SATZ        0 22
D        KDNR           5 0
D        FIRMA         13
D        PLZ            6
C           ANFANG    TAG
C                     MAPD EINGABE
C                     +    1         LNR     50
C                     EDIT           KEY    20
C           KEY       CHAINCPGWKV              C01
C                     EXCPT
C                     GOTO ANFANG
OCPGWKV  EADD     01
O                         KEY       20
O                         SATZ      42
OCPGWKV  E       N01
O                         SATZ      42
OKEY     F
O                                    2 '22'
O                         LNR       20P
```

Das erstellte Programm dient als Input für das in Schritt 2 beschriebene Serviceprogramm. Deshalb müssen der Dateizugriff und die Schlüsselpositionen unbedingt mit dem Beispiel übereinstimmen.

Das Beispiel ist gewählt worden, um die wesentlichen Karten übersichtlich darzustellen. In der Regel wird in der Praxis ein Batchprogramm zu schreiben sein, das die Daten der View aus einer Datei oder mehreren Dateien zusammenstellt.

2.Schritt: Dokumentieren und Abstellen der View

Zum Dokumentieren und Abstellen der View steht die Phase CPGZCTB zur Verfügung. Es ist ein Job wie folgt zu erstellen:

```
:
// JOB JCPGZCTB
// PAUSE          Datei CPGWKV closen
// EXEC CPGZCTB,SIZE=AUTO
Vorlaufkarte
/*
// PAUSE          Datei CPGWKV eröffnen
/&
```

Die Vorlaufkarte hat folgenden Aufbau:

Stellen 1 - 4	Name der Tabelle, mit dem sie im Befehl FIND angesprochen wird.
Stellen 5 - 7	Länge der Tabellenelemente
Stelle 8	L für 'Löschen einer Tabelle' (optional) C für nur 'Create einer Tabelle'
Stellen 10 - 12	Personenkurzzeichen des Sachbearbeiters (optional)
Stellen 13 - 40	Dokumentation, Teil 1 (optional)
Stellen 41 - 80	Dokumentation, Teil 2 (optional)

3.Schritt: Kopieren FIND-Tabellen

Mit folgendem Programm (CPGZCTC) können FIND-Tabellen von einer Datei on eine andere kopiert werden.

```
// JOB CPGZCTC
// PAUSE          DATEI CPGWKV.OUT CLOSEN
// DLBL CPGWKV,'CPGWKV.INP',,VSAM,CAT...
// DLBL CPGWKVO,'CPGWKV.OUT',,VSAM,CAT...
// EXEC CPGZCTC,SIZE=(AUTO,64K)
TAB1                    .
TAB2                    Kommentar ab 25
TAB3                    .
/*
/&
```

Vorgehensweise für CPG3-Anwender 7532

CPG3-Anwendern wird das Handling von Datenviews erleichtert. Zum Erstellen einer View kann CPG3..Query benutzt werden und das Serviceprogramm QTS (Quick Table Service) ermöglicht online das Löschen einer View aus der Datei oder im Hauptspeicher, das Laden einer neuen Version aus der Datei und das Anzeigen aller angelegten Tabellen.

Diese Serviceprogramme sind im CPG3-Handbuch beschrieben.

Tabellen im QTF 7534

CPG4-Benutzer können Tabellen einfach mit QTF verwalten. Siehe QTF-Handbuch.

CPG-Umwandlung ohne IJSYS04 7540

Mit folgendem JCL Job wird ein Beispiel aufgezeigt, wie CPG-Umwandlungen ohne ASSGN-SYSIN-Anweisung ermöglicht werden. Bei dieser Ausführung wird der IJSYS04-File nicht benötigt.

Diese Art der CPG-Umwandlung bringt Laufzeitverbesserungen, wenn 3380-Platten installiert sind oder wenn die Umwandlungsbereiche über VSAM Space Manager verwaltet werden.

```
 * $$ JOB JNM=CPGUMW,DISP=D,CLASS=A
 * $$ PUN DISP=I,CLASS=A
 // JOB CPGUMW
 // EXEC CPGZPUN,SIZE=AUTO
        * $$ JOB JNM=CPGASM,DISP=D,CLASS=A
        // JOB CPGASM
 /*
 // EXEC PROC=CPGPROC                    (evt. wenn nicht STD-Label)
 // EXEC CPG
      H   P
      :
      :    CPG-PROGRAMM
      :
 /*
 // EXEC CPGZPUN,SIZE=AUTO
       // LIBDEF PHASE,CATALOG=USR1T.BG   (evt. wenn nicht PERM
       // EXEC LNKEDT                      zugeordnet)
       /&
       * $$ EOJ
 /&
 * $$ EOJ
```

Erläuterungen zu den JCL-Karten:

Die Power-Punch-Anweisung mit dem Parameter DISP=I stellt die gestanzten Karten in die Power Reader Queue ab.

In der CPG-H-Karte muss in Spalte 10 ein 'P' eingetragen werden. Die Phase CPGZPUN ist im Lieferumfang von CPG enthalten.

Führende Leerstellen bei einem auszugebenden Statement müssen mit '#' gekennzeichnet werden. Siehe folgendes Beispiel:

```
 // OPTION CATAL
 ###PHASE NAME,*
```

CPG-Umwandlung Command Level ohne Methodenbank 7560

```
* $$ JOB JNM=CPGUMW,CLASS=A
* $$ LST CLASS=L,FNO=0112
* $$ PUN CLASS=A,DISP=I
// JOB CPGUMW
// EXEC CPGZPUN
   * $$ JOB JNM=CPGPREP,CLASS=A
   * $$ LST CLASS=L,FNO=0112
   * $$ PUN CLASS=A,DISP=I
   // JOB CPGPREP          JOB DFHEAP1$
   // EXEC CPGZPUN
   ##* $$ JOB JNM=CPGASM,CLASS=A
   ##* $$ LST CLASS=L,FNO=0112
   ##// JOB CPGASM         JOB ASSEMBLY
/*
// EXEC HL1
     H   P                              D   C TITEL           TST026
-INC TCPG390                                                       %
/*
// EXEC CPGZPUN
   // EXEC CPGZPUN
   ##/*
   ##// EXEC LNKEDT
   ##/&                    JOB ASSEMBLY
   ##* $$ EOJ
   /&                      JOB DFHEAP1$
   * $$ EOJ
/&
* $$ EOJ
```

Ein Beispiel für die Job Control Statements bei - SQL-Verarbeitung mit CPG3:

```
 // JOB CPGSQL
 // LIBDEF PROC,SEARCH=(PRD2.SQL220)
 // EXEC PROC=ARISLIBP   *-- SQL/DS PRODUCTION LIBRARY ID PROC
 // ON $ABEND GOTO REASS
 // EXEC CPGPREP,PARM='USERID=SQLDBA/SQLDBAPW'
      H                                  L  CSQL TESTPROGRAMM   TST026
      FBILD
      DSQ      BEGIN DECLARE SECTION
      D        USER           8
      D        PASSW          8
      D        KDNR           5 0
      D        FIRMA         30
      DSQ      END DECLARE SECTION
      C                     MOVEL'SQLDBA  'USER
      C                     MOVEL'SQLDBAPW'PASSW
      CSQ CONNECT :USER IDENTIFIED BY :PASSW
      CSQ DECLARE C2 CURSOR FOR
      CSQ     SELECT KKDNR,KFIRMA
      CSQ     FROM KUNDENA
      CSQ     WHERE KKDNR  < 3000
      CSQ OPEN C2
      C           A100      TAG
      CSQ FETCH C2
      CSQ     INTO  :KDNR,:FIRMA
      C           SQCODE    COMP 0                    6060
      C                     EREADBILD
      C  NP1                GOTO A100
      CSQ CLOSE C2
      CSQ COMMIT WORK
      OBILD    E E
      O                                  250 'TEST SQL'
      O                         KDNR     340
      O                         FIRMA    440
      O                60       SQCODE   640
 /*
 // IF $RC NE 0 THEN
 // GOTO ENDE
 *  STEP HL1
 // LIBDEF PHASE,CATALOG=SP4U.ULIBL
 // DLBL IJSYSIN,'F4.WORK.04',0,SD,,CISIZE=8192
 // EXTENT SYSIPT,PRD201,1,0,46000,4000
 ASSGN SYSIPT,122
 // EXEC HL1
 /*
 // IF $RC NE 0 THEN
 // GOTO REASS
 *  STEP ASSEMBLER
 CLOSE SYSIPT,READER
 ASSGN SYSIN,122
 *  STEP LNKEDT
 // EXEC LNKEDT
 /*
```

```
// IF $RC EQ 0 THEN
// GOTO ENDE
/. REASS
CLOSE SYSIPT,READER
/. ENDE
/&
```

Beispiele 8000

Beispiel 1. Echo

```
BEISPIEL 1, ECHO                                                01.07.81
_____________________________________________________________________

 1     01 001  H                                        BEISPIEL 1,ECHO

 2     01 002  FBILD    C                    DISPLAY

 3     02 001  IBILD    KF
 4     02 002  I                                      209 212 FELDA

 5     03 001  C                    SETON                      01
 6     03 002  C                    EXCPT
 7     03 003  C                    READ BILD
 8     03 004  C                    SETON                      02
 9     03 005  C                    EXCPT

10     04 001  OBILD    E E      01N02
11     04 002  O                                  207 'EINGABE'
12     04 003  O                                AC 212 '    '
13     04 004  O        E        02
14     04 005  O                                  307 'AUSGABE'
15     04 006  O                         FELDA A  312
```

Dieses Beispiel zeigt die einfachste Form eines Dialogs zwischen Bildschirm und Zentraleinheit, nämlich: Ausgabe einer Maske auf den Bildschirm(der Bildschirm wird zur Aufnahme der Nachricht vorformatisiert) Lesen der über die Bildschirmtastatur eingegebenen Nachricht, Ausgabe der Nachricht auf den Bildschirm.

Erläuterungen:

1 CPG-Steuerkarte.

2 Dateizuordnung: Eine Datei mit dem Namen BILD wird definiert. Die Dateiart ist 'C' (=kombinierte) oder in diesem Falle Dialogdatei. Die Einheit ist am Bildschirm DISPLAY. Standardmäßig wird 24 Zeilen mit je 80 Zeichen angenommen.

3 Eine Eingabenachricht soll auf Anforderung des Programms von der Datei BILD gelesen werden. In den Spalten 15 und 16 muss dazu eine beliebige Eintragung ungleich blank enthalten sein (KF).

4 Die Daten werden aus dem Feld 'FELDA' gelesen. Das Feld beginnt in Zeile 2, Stelle 9 (209), und endet in Zeile 2, Stelle 12 (212).Das Feld ist alphanumerisch (blank in Spalte 52)

5 Schalter 01 wird angesetzt.

6 Durchführung sämtlicher Ausgabe-Bestimmungen.

7 Satz der Datei BILD lesen.

8 Schalter 02 wird angesetzt.

9 Durchführung sämtlicher Ausgabe-Bestimmungen.

10 Ein Satz soll auf Anforderung des Programms (EXCPT) zur Datei BILD ausgegeben werden. 'E' in Spalte 15 besagt, dass diese Ausgabe-Bestimmung durch EXCPT ausgeführt wird. 'E' in Spalte 17 löscht den Bildschirm vor der Ausgabe. Die Ausgabe erfolgt, wenn Schalter 01 ein (01), und Schalter 02 nicht ein (N02) ist.

11 Die Konstante 'Eingabe' wird in Zeile 2 ausgegeben. Die letzte Position der Konstanten steht in Spalte 7 der Zeile 2 (207).

12 Die Konstante ' ' wird ausgegeben. Das letzte Blank der Konstanten steht in Zeile 2, Position 12 (212). Das vierstellige Feld ist ungeschützt, d.h.formatisiert für die Aufnahme von Daten über die Bildschirm-Tastatur. Die Daten werden mit doppelter Helligkeit angezeigt und können alphanumerisch sein (A) (siehe auch Attribut). Der Positionsanzeiger steht in der 1. Stelle des Feldes (C).

13 Ein zweiter Satz soll auf Anforderung des Programms zur Datei BILD ausgegeben werden. Bei fehlendem Dateinamen gilt der Name der vorherigen Satzbestimmung. Der Bildschirm wird diesmal nicht gelöscht (Blank in Spalte 17). Die Ausgabe erfolgt nur, wenn der Schalter 02 ein ist (02).

14 In Zeile 3 wird die Konstante 'AUSGABE' von Position 1 bis 7 angezeigt (307).

15 In Zeile 3 wird der Inhalt des Feldes mit dem Namen 'FELDA' angezeigt. Die letzte Stelle des Feldes steht auf Position 12.
Die Feldeigenschaften sind wie bei Position 12 beschrieben (A).

Die Rechenbestimmungen zu diesem Beispiel wurden in der dem Anfänger geläufigeren RPG-Schreibweise geschrieben. Mit Hilfe der CPG-Erweiterungen können die 5 Operationen wie folgt auf 2 Operationen reduziert werden:

```
 5     03 001  C                   EREADBILD                 01
 6     03 002  C                   EXCPT                     02
```

Beispiel 2: Platten-Änderung

Die Datei 'PLATTE' ist eine indexsequentiell organisierte Plattendatei mit einer Satzlänge von 85 Bytes. Die Daten stehen in Spalte 1 bis 80. Spalte 81 enthält ein Änderungskontrollbyte (A) und Spalte 82 bis 85 den Schlüssel als fortlaufende Satznummer in gepacktem Format.

```
STANDARDTEXT (Z.B. FIRMA)             KDNR KURZNAME           01.07.81
_____________________________________________________________________

 1     01 001  FBILD    C                          DISPLAY
 2     01 002  FPLATTE  U   F      85 04           KSDS

 3     02 001  IBILD    KF
 4     02 002  I                                      227 2330SATZNR
 5     02 003  I                                      601 680 BKARTE
 6     02 004  IPLATTE  KF
 7     02 005  I                                        1  80 BKARTE

 8     03 001  C                    EDIT           SCALA  80
 9     03 002  C           OBEN     TAG
10     03 003  C                    EREADBILD                  01
11     03 004  C                    Z-ADDSATZNR    SATZNR
12     03 005  C           SATZNR   CHAINPLATTE                10
13     03 006  C                    EREADBILD                  02
14     03 007  C   P1               EXCPT                      03
15     03 008  C                    GOTO OBEN

16     04 001  OBILD    E E      01
17     04 002  O                            216 'BITTE SATZNUMMER'
18     04 003  O                            225 'EINGEBEN'
19     04 004  O                         AC 233 '       '
20     04 005  O        E        02
21     04 006  O                   BKARTEAC 680
22     04 007  O                   SCALA    880
23     04 008  OPLATTE  E        03
24     04 009  O                   BKARTE    80
25     04 010  O                              81 'A'
26     04 011  OSCALA   F
27     04 012  O                              20 '....5...10....5...20'
28     04 013  O                              40 '....5...30....5...40'
29     04 014  O                              60 '....5...50....5...60'
30     04 015  O                              80 '....5...70....5...80'
```

Erläuterung:

Ablauf: Nach Eingabe des Transaktionsschlüssels (Programm-Identifikation) über die Bildschirm-Tastatur erscheint auf dem Bildschirm die Meldung: 'Bitte Satznummer eingeben'. Der Positionsanzeiger steht dabei in Zeile 2 Stelle 27. Der Bediener gibt in das vorformatisierte 7-stellige Feld eine gültige Satznummer ein. Nach Betätigung einer beliebigen Programmfunktionstaste durch den Bediener wird der gewünschte Satz in Zeile 6 des Bildschirms angezeigt. In Zeile 8 zeigt eine Skala von 1 - 80 die Spalten des Satzes an. Der Positionsanzeiger steht in Spalte 1 des Satzes. Der Bediener kann jetzt den angezeigten Satz durch einfaches Überschreiben beliebig verändern. Durch Betätigung der Programmfunktionstaste 1 wird der geänderte Satz auf die Platte zurück geschrieben. Zur späteren Kontrolle wird in Spalte 81 die Konstante 'A' ausgegeben. Die Betätigung irgendeiner anderen Funktionstaste als PF1 setzt bei der Meldung 'Bitte Satznummer eingeben' wieder neu auf, ohne den Plattensatz zu verändern.

Die Bezugszahl P1 ist immer dann 'EIN', wenn bei der vorangehenden 'READ'-Operation für ein Bildschirm-Terminal vom Bildschirmbediener die Programmfunktionstaste 'PF1' betätigt wurde und wird bei der nächsten 'READ'-Operation vom Programm wieder gelöscht.

Zu Position 11: Der Z-ADD-Befehl ist erforderlich, wenn der Schlüssel des Plattensatzes über eine Rechenroutine erstellt wurde. Der Schlüssel enthält dann das positive Vorzeichen 'C', während die am Bildschirm eingegebene Zahl das Vorzeichen 'F' enthält. Der Satz würde folglich ohne den Z-ADD-Befehl nicht gefunden.

Über den Eintrag 'S' in Spalte 40 der H-Karte kann ein Einlesen numerischer Felder mit Vorzeichen 'C' wenn positiv erzeugt werden.

Beispiel 3: Dateianzeige mit READ PAGE

Die Datei Kunden ist eine indexsequentiell organisierte Plattendatei

```
DATEIANZEIGEPROGRAMM                  KDNR KURZNAME        01.07.81
_____________________________________________________________________

 1     01 001 H                                  DATEIANZEIGEPROGRAMM

 2     01 002 FBILD    C      24  80              DISPLAY
 3     01 003 FKUNDEN  I   F     500  7           DISK

 4     02 001 E                    PAGE       20 79

 5     03 001 IBILD    KF
 6     03 002 I                                     227 233 KDNR
 7     03 003 IKUNDEN  KF
 8     03 004 I                                       3   9 KDNR
 9     03 005 I                                      21  44 NAME
10     03 006 I                                      73  92 ORT
11     03 007 I                                    P 135 1392UMSATZ

12     04 001 C                      SETON                    02
13     04 002 C                      EXCPT
14     04 003 C                      SETOF                    02
15     04 004 C            LESEN     TAG
16     04 005 C                      READ BILD
17     04 006 C            KDNR      READ KUNDEN    PAGE
18     04 007 C   EF                 SETON                    03
19     05 008 C                      RNDOMKUNDEN
20     04 009 C                      EXCPT                    04
21     04 010 C   P1N03              GOTO LESEN

22     05 001 OBILD    E E      02
23     05 002 O                                        214 'BITTE KUNDENNR'
24     05 003 O                                        224 'EINGEBEN'
25     05 004 O                                     AC 233 '       '
26     05 005 O        E E      04
27     05 006 O                                        305 'KDNR'
28     05 007 O                                        314 'NAME'
29     05 008 O                                        339 'ORT'
30     05 009 O                                        363 'UMSATZ'
31     05 010 O                           PAGE         580
32     05 011 OPAGE    F
33     05 012 O                           KDNR          7
34     05 013 O                           NAME         33
35     06 014 O                           ORT          55
36     07 015 O                           UMSATZ       70 ' .   . 0 ,  -'
```

Erläuterungen:

1 CPG-Steuerkarte.

2 Dateizuordnung: Eine Datei mit dem Namen BILD wird definiert. Die Dateiart ist 'C' (=kombinierte) oder in diesem Falle Dialogdatei. Die Satzlänge ist variabel (V), der Bildschirm hat 24 Zeilen a 80 Zeichen und der Bildschirm-Arbeitsbereich wurde in der Terminal-Tabelle mit 300 Bytes definiert. Die physische Einheit ist ein 3270-Bildschirm (DISPLAY).

3 Eine Datei mit dem Namen KUNDEN wird definiert. Die Dateiart ist 'I' Eingabedatei. Die Satzlänge ist fest (F) 500 Bytes. Die Schlüssellänge ist 7 Bytes. Die physische Einheit ist eine Platte. DISK gilt für ISAM und VSAM.

4 Erweiterte Dateizuordnung: Eine Feldgruppe mit dem Namen PAGE wird definiert. Die Feldgruppe besteht aus 20 Feldern mit einer Länge von je 79 Bytes. Die Feldgruppe soll eine Bildschirmseite mit 20 Zeilen (Zeilenlänge 79) aufnehmen.

5 Eingabebestimmungen: Von der Datei BILD sollen Daten gelesen werden.

6 In Zeile 2 steht von Pos. 27 bis Pos. 33 die Kundennummer in alphanumerischem Format.

7 Von der Datei KUNDEN soll ein Satz gelesen werden.

8 Im Kundensatz steht die Kundennr (KDNR) von Pos.3 bis Pos.9.

9 Der Name steht von Pos. 21 bis Pos. 44.

10 Der Ort steht von Pos. 73 bis Pos. 92.

11 Der Umsatz des Kunden steht von Pos. 135 bis Pos. 139 in gepacktem Format (P).Das Feld wird numerisch definiert mit 2 Dezimalstellen.

12 Rechenbestimmungen: Schalter 02 wird gesetzt.

13 Die Ausgabebestimmungen werden durchgeführt.

14 Schalter 02 wird gelöscht.

15 Ein Merkmal 'LESEN' wird gesetzt.

16 Von der Datei BILD wird ein Satz gelesen.

17 Von der Datei KUNDEN werden so viele Sätze gelesen, wie die im Ergebnisfeld eingetragene Feldgruppe Elemente enthält. Der erste gelesene Satz ist der, dessen Schlüssel gleich oder größer als der Inhalt des Feldes 'KDNR' ist. Die Seite wird in der Feldgruppe PAGE gespeichert. Die Anzahl der Zeilen ergibt sich aus der unter Nr.4 definierten Feldanzahl. Zur Aufbereitung der Zeilen siehe Nr.31 bis 35.

18 Bei Dateiende (EF) wird Schalter 03 gesetzt. Der EF-Schalter muss unmittelbar nach der READ-Operation für eine Datei abgefragt werden.

20 Die Ausgabebestimmungen werden selektiert durchgeführt, d. h. der

Schalter 04 wird gesetzt, die Ausgabebestimmungen werden durchgeführt, der Schalter 04 wird wieder gelöscht.

21 Der Schalter P1 ist immer dann gesetzt,wenn bei der vorangehenden READ-Operation für ein Terminal vom Bediener die Programmfunktionstaste PF1 betätigt wurde. Sie wird bei der nächsten READ-Operation für diese Datei vom Programm wieder gelöscht.

Bei P1 wird ein neues READ PAGE durchgeführt, d.h. die Funktionstaste PF1 hat für dieses Programm die Funktion 'Blättern' (oder 'Lies nächste Seite'). Die Wiederholung erfolgt jedoch nur, wenn der Schalter 03 (Dateiende) nicht gesetzt ist.

22 Ausgabebestimmungen: Zur Datei BILD soll ein Satz ausgegeben werden, wenn der Befehl EXCPT gegeben wird (E in Spalte 15) und der Schalter 02 gesetzt ist. Der Bildschirm soll vor Ausgabe gelöscht werden (E in Spalte 17).

23 In Zeile 2 wird die Konstante 'BITTE KUNDENNR' ausgegeben. Das letzte Byte steht auf Pos. 15

25 In Zeile 2 wird von Pos. 27 bis 33 ein Feld für die Eingabe der Kundennummer vorformatisiert (siehe Nr.6). Das Feld kann alphanumerische Zeichen aufnehmen (A). Der Positionsanzeiger oder Cursor wird in dieses Feld gesetzt (C).

26 Wie Nr. 22, jedoch bei Schalter 04.

27 In Zeile 3 werden die folgenden Konstanten ausgegeben:
'KDNR' letztes Byte in Pos. 5

28 'NAME' letztes Byte in Pos. 14

29 'ORT' letztes Byte in Pos. 39

30 'UMSATZ' letztes Byte in Pos. 63

31 Die Feldgruppe 'PAGE' wird ausgegeben. Das erste Feld steht in Zeile 5 von Pos. 2 bis Pos. 80(580). Die folgenden Felder werden in den gleichen Positionen der jeweils folgenden Zeilen ausgegeben z.B. Feld 2 in Zeile 6, Feld 3 in Zeile 7 usw.

32 Das Feld bzw. die Feldgruppe PAGE soll aufbereitet werden (F in Spalte 15). Die Feldaufbereitung wird ausgelöst durch den Befehl 'READP'.

33 Das Feld KDNR wird in Pos. 1 bis 7 aller Felder der Feldgruppe ausgegeben.

34 Wie 32, Feld NAME, letztes Byte auf Pos. 33.

35 Wie 32, Feld ORT, letztes Byte auf Pos. 55.

36 Wie 32, Feld UMSATZ, jedoch aufbereitet mit Schablone.

Beispiel 3b: Dateianzeige mit READ PAGE transaktionsorientiert.

Die Datei KUNDEN ist eine indexsequentiell organisierte Plattendatei

```
DATEIANZEIGEPROGRAMM                  KDNR KURZNAME        18.12.87
_____________________________________________________________________

 1           H                                  DATEIANZEIGEPROGRAMM

 2           FBILD    C      24  80              DISPLAY
 3           FKUNDEN  I   F     500  7           DISK

 4           E                    PAGE       20 79

 5           IBILD    KF
 6           I                                      227 233 KDNR
 7           IKUNDEN  KF
 8           I                                        3   9 KDNR
 9           I                                       21  44 NAME
10           I                                       73  92 ORT
11           I                                     P 135 1392UMSATZ

12           C   CL               GOTO ENDE
13           C                    TWALDTT12
14           C   EF               GOTO ANFANG
15           C                    READIBILD
16           C   P1               GOTO ANFANG
17           C           KDNR     READ CPGKDN    PAGE
18           C   EF               GOTO ENDE
19           C           KDNR     READ CPGKDN
20           C                    RNDOMCPGKDN
21           C                    EXCPT                    02
22           C                    GOTO LESEN
23           C           ANFANG   TAG
24           C                    EXCPT                    01
25           C           LESEN    TAG
26           C                    TWASVTT12
27           C                    EXITT'TT12'
28           C           ENDE     TAG
29           C                    EXCPT                    09

30           OBILD    E E      01
31           O                                   214 'BITTE KUNDENNR'
32           O                                   224 'EINGEBEN'
33           O         C                       C 233 '       '
34           O        E E      02
35           O                                   305 'KDNR'
36           O                                   314 'NAME'
37           O                                   339 'ORT'
38           O                                   363 'UMSATZ'
39           O                          PAGE     580
40           O        E E      09
41           O                                  2413 'PROGRAMMENDE'
42           O         A                       C2480
43           OPAGE    F
44           O                          KDNR       7
45           O                          NAME      33
46           O                          ORT       55
47           O                          UMSATZ    70 ' .   . 0 ,  -'
```

Erläuterungen:

Das Programm hat etwa die gleichen Funktionen wie das Programmbeispiel 3. Es weist jedoch bei transaktionsorientierter Verarbeitungsform folgende Besonderheiten auf:

12 Die CL-Taste muss am Programmanfang abgefragt werden, um das Programm zu beenden.

13 Mit der Operation TWALD wird die komplette TWA von Temporary Storage zurückgeladen. Der Programmierer wird dadurch vom Erstellen der Eingaberoutinen zum Zurückspielen der noch benötigten Felder entlastet.

14 Beim ersten Aufruf des Programms ist der Schalter EF gesetzt, da noch keine TWA auf TS gerettet wurde. Hierbei wird zweckmäßigerweise zu der Anfangsmaske verzweigt.

15 Mit der Operation READIBILD werden die modifizierten Felder vom Bildschirm eingelesen.

26 Vor Verlassen der Transaktion wird die komplette TWA auf Temporary Storage gerettet. Der Programmierer wird dadurch vom Erstellen der Ausgaberoutinen zum Speichern der noch benötigten Felder entlastet.

27 Mit der Operation EXITT wird die Folgetransaktion gestartet.

29 Bei Programmende sollte mit einem EXCPT die Tastatur entriegelt.

42 Durch das Attribut 'A' können Daten am Bildschirm eingegeben werden, um die nächste CICS-Transaktion zu starten. Der Cursor befindet sich am linken oberen Bildschirmrand.

Beispiel 4: Feldauswahl mit Lichtstift.

Erläuterungen:

Beispiel 3 soll so abgewandelt werden, dass aus der angezeigten Datei durch Antippen mit dem Lichtstift ein Satz ausgewählt und die zugehörige Kundennummer zur Verwendung in einem Folgeprogramm in die TCT übertragen wird.

Erweiterung des Beispiels 3.

```
STANDARDTEXT Z.B. FIRMA                KDNR KURZNAME        01.07.81
_____________________________________________________________________

 1     01 001 FBILD                                                PROG01
 2     01 002 FKUNDEN

 3     02 001 E                    PAGE       20 71

 4     03 001 IBILD    KF
 5     03 002 I                                      227 233 KDNR
 6     03 003 I                                      502 572 PAGE   20
 7     03 004 IKUNDEN  KF
 8     03 005 I                                        3   9 KDNR
 9     03 006 I                                       21  44 NAME
10     03 007 I                                       73  92 ORT
11     03 008 I                                     P 135 1392UMSATZ
12     03 009 IPAGE    F
13     03 010 I                                        2   8 KDNR

14     04 001 C                    EREADBILD                 01
15     04 002 C           READ     TAG
16     04 003 C           KDNR     READPKUNDEN    PAGE
17     04 004 C           KDNR     READ KUNDEN
18     04 005 C   EF               GOTO ANFANG
19     04 006 C                    RNDOMKUNDEN
20     04 007 C                    EREADBILD                 02
21     04 008 C  NSP               GOTO READ
22     04 009 C                    SETIX          I       30 20
23     04 010 C           I        COMP 20                   50
24     04 011 C  N50               SELCT          PAGE,I
25     04 012 C  N50               EDIT           CPGTCT
26     04 013 C  N50               EXIT 'PR02'
27     05 001 OBILD    E E      01
28     05 002 O                                     214 'BITTE KUNDENNR'
29     05 003 O                                     224 ' EINGEBEN'
30     05 004 O                                  AC 233 '       '
31     05 005 O        E E      02
32     05 006 O                                     306 'KDNR'
33     05 007 O                                     315 'NAME'
34     05 008 O                                     340 'ORT'
35     05 009 O                                     364 'UMSATZ'
36     05 010 O                       PAGE    L     572
37     05 011 OPAGE    F
38     05 012 O                       KDNR          8
39     05 013 O                       NAME         34
40     05 014 O                       ORT          56
41     05 015 O                       UMSATZ       71 ' .   . 0 ,  -'
42     05 016 OCPGTCT  F
43     05 017 0                       KDNR          7
```

Die Abweichungen gegenüber Beispiel 3 sind:

1 Die CPG-Steuerkarte ist nicht erforderlich, wenn Standard-Werte angenommen werden. Die Daten der Datei werden der CPG-File-Table entnommen, soweit diese mit dem CPG-File-Table-Generator generiert wurde.

2 Die Daten der Datei KUNDEN werden der CPG-File-Table entnommen (Siehe 1).

6 Die Feldgruppe PAGE muss vom Bildschirm wieder eingelesen werden, um festzustellen, welches Feld ausgewählt wurde. Die Bezugszahl 20 in den Spalten 61-62 besagt, dass jedem Feld der Feldgruppe fortlaufend, beginnend mit der Bezugszahl 20, eine Bezugszahl zugeordnet wird, also bei 20 Feldern die Bezugszahlen 20 bis 39. Die Bezugszahl 21 wird dann gesetzt, wenn das 2. Feld angetippt wird usw.

Es ist darauf zu achten, dass die Bezugszahlen 20 bis 39 nicht an Anderer Stelle im Programm gesetzt werden.

12 Aus der Feldgruppe PAGE soll mit Hilfe einer SELCT-Operation ein Feld ausgewählt werden; daher muss die Feldgruppe in den Eingabebestimmungen mit 'F' in Spalte 15 angeführt werden.

13 Aus den Stellen 2 bis 8 eines Feldes der Feldgruppe soll das Feld KDNR gelesen werden.

14 Die Operationen: SETON 01, EXCPT, READ BILD, SETOF 01 werden zur Operation EREADBILD 01 zusammengefasst.

20 Wie 14.

22 Wenn der Lichtstift ein Feld ausgewählt hat (SP), soll im Feld I, das mit 3 Stellen, davon 0 Dezimalstellen definiert wurde, der Index ermittelt werden, d.h. die wievielte Bezugszahl beginnend mit 20 gesetzt wurde.

23 Bei Programmfehlern kann der Index nach der SETIX-Operation einen unsinnigen Wert enthalten, daher sollte der Index vor der weiteren Verwendung auf Gültigkeit geprüft werden.

24 Aus der Feldgruppe soll das angetippte Feld ausgewählt werden. Nach Ausführung dieser Instruktion steht im Feld KDNR die Kundennummer des ausgewählten Feldes.

25 Die Kundennummer soll zur weiteren Verarbeitung in einem oder mehreren Folgeprogrammen in die User-Area der Terminal-Control-Table gestellt werden.

26 Die weitere Verarbeitung erfolgt in einem Programm mit dem Transaktionsschlüssel 'PRO2'.

36 Die Feldgruppe muss mit dem Lichtstift ansprechbar sein, daher 'L' in Spalte 38.

38 Das Feld KDNR ist 7 Stellen lang. Der Lichtstift erfordert neben dem Aufbereitungsschlüssel 'L' ein Blank in der 1. Stelle des Feldes, daher Ausgabe der Kundennummer in Position 2 bis 8. Eine weitere Voraussetzung ist, dass das Feld nicht in Spalte 1 der Zeile beginnen darf.

42 CPGTCT ist die User-Area der TCT. Dieses Feld braucht im Programm nicht definiert zu werden. Es muss hier mit 'F' in Spalte 15 aufgeführt werden, weil es in den Rechenbestimmungen durch eine EDIT-Operation angesprochen wird.

43 Die Kundennummer kann in späteren Programmen mit einer SELCT-Operation den Stellen 1 bis 7 von CPGTCT wieder entnommen werden.

Beispiel 5: Dateiänderungsprogramm mit UPDATE, WRITE

```
STANDARDTEXT (Z.B. FIRMA)              KDNR KURZNAME        01.07.81
_____________________________________________________________________

 1     01 001 FBILD                                              PROG05
 2     01 002 FKUNDEN

 3     02 001 E                    SATZ         0140
 4     02 002 E                    D1              2
 5     02 003 E                    KDNR            7
 6     02 004 E                    D2             11
 7     02 005 E                    NAME           24
 8     02 006 E                    D3             28
 9     02 007 E                    ORT            20
10     02 008 E                    D4             43
11     02 009 E                    UMSATZ          9 2

12     03 001 IBILD    KF
13     03 002 I                                       227 233 KDNR
14     03 003 I                                       327 350 NAME
15     03 004 I                                       427 446 ORT
16     03 005 I                                       527 5352UMSATZ
17     03 006 IKUNDEN  KF
18     03 007 I                                         1 140 SATZ

19     04 001 C           ANFANG    TAG
20     04 002 C                     EREADBILD                01
21     04 003 C           KDNR      CHAINKUNDEN             U20
22     04 004 C                     EREADBILD                02
23     04 005 C  N20      KDNR      UPDATKUNDEN    SATZ
24     04 006 C   20      KDNR      WRITEKUNDEN    SATZ
25     04 007 C                     GOTO ANFANG

26     05 001 OBILD    E E      01
27     05 002 O                                    214 'BITTE KUNDENNR'
28     05 003 O                                    224 'EINGEBEN'
29     05 004 O                                 AC 233 '       '
30     05 005 O        E E      02
31     05 006 O                          KDNR     233
32     05 007 O                          NAME  AC 350
33     05 008 O                          ORT   A  446
34     05 009 O                          UMSATZA  540 ' .   . 0 ,  -'
```

Erläuterungen:

Beispiel 5 zeigt ein Änderungsprogramm für eine Kundenstammdatei mit den CPG-Operationen UPDAT und WRITE, die den Programmier- und Speicheraufwand je nach Anwendung reduzieren können.

1 Die CPG-Steuerkarte ist nicht erforderlich, wenn Standard-Werte angenommen werden. Die erste Karte des Quellenprogramms enthält dann in den Spalten 75-80 den Phasennamen des Programms. Die Daten der Datei BILD werden der CPG-File-Table entnommen (siehe Abschnitt 8: Standard-Dateizuordnung).

2 Die Daten der Datei KUNDEN werden der CPG-File-Table entnommen (siehe 1).

3 Die folgenden Felder (Statement 4 bis 11) werden zu einem Feld mit dem Namen 'SATZ' zusammengefasst. Das Feld ist 140 Stellen groß und alphanumerisch (Blank in Spalte 44). Für das Feld wird kein Speicherplatz in der TWA reserviert (Null in Spalte 39), sondern das Feld wird nacheinander von links nach rechts durch die unmittelbar im Anschluss daran definierten Felder belegt. Der Programmierer muss beachten, dass die Summe der Einzelfeldlängen mit der Feldlänge des überlagerten Feldes übereinstimmt. Bei numerischen Feldern ist dabei die Länge des gepackten Feldes in Bytes (nicht die Stellenzahl) für die Berechnung einzusetzen.

4 Bis 11: Die Felder 'D1' bis 'UMSATZ' werden entsprechend ihrer Lage im Plattensatz definiert. Nicht benutzte Plattenfelder müssen dabei als Dummy-Felder (D1 bis D4) ebenfalls definiert werden, damit das Feld 'Satz' ein echtes Abbild des Plattensatzes beschreibt. Das Feld Umsatz ist dabei in 5 Bytes gepackt.

12 Von der Datei BILD soll ein Satz gelesen werden.

13 In Zeile 2 des Bildschirms steht von Pos. 27 bis Pos. 33 die Kundennummer in alphanumerischem Format.

14 bis 16: Wie bei 13 werden die Felder Name, Ort und Umsatz eingelesen.

17 Von der Datei Kunden soll ein Satz gelesen werden.

18 Der Satz wird von Stelle 1 bis 140 in das Feld 'Satz' eingelesen. Über die Definition in der erweiterten Dateizuordnung werden dabei gleichzeitig die Felder KDNR, NAME, ORT und UMSATZ gefüllt.

19 Ein Merkmal 'ANFANG' wird gesetzt.

20 Die Operationen: SETON 01, EXCPT, READ BILD, SETOF 01 werden zur Operation EREAD Bild 01 zusammengefasst.

21 Ein Plattensatz wird von der Datei KUNDEN mit dem Schlüssel KDNR gelesen. Der Plattensatz soll bis zur Veränderung gegen Zugriffe von außen gesperrt werden (U in Spalte 53). Wenn der Satz nicht vorhanden ist, soll der Schalter 20 (Spalte 54-55) gesetzt werden.

22 Wie 20.

23 Wenn 20 nicht gesetzt ist, soll das Feld 'SATZ' in die Datei KUNDEN,in den Satz mit dem Schlüssel KDNR,zurückgeschrieben werden. Ausgabebestimmungen sind dabei nicht erforderlich, jedoch muss dass Feld Satz in den Eingabebestimmungen unter der Datei KUNDEN beschrieben sein.

24 Wie 23, jedoch wird hier,wenn der Schalter 20 gesetzt ist ein neuer Satz zur Datei KUNDEN hinzugefügt.

25 Das Programm wird bei Anfang (Statement 19) fortgesetzt.

26 Bis 34. Die Bildschirmausgabe wurde unter Beispiel 3 ausführlich beschrieben.

Beispiel 5A: Dateiänderungsprogramm, mit EXCPT

```
STANDARDTEXT (z.B. FIRMA)              KDNR KURZNAME        26.11.85
_____________________________________________________________________

 1     01 001 FBILD                                              PROG05
 2     01 002 FKUNDEN

 3     02 001 IBILD    KF
 4     02 002 I                                          227 233 KDNR
 5     02 003 I                                          327 350 NAME
 6     02 004 I                                          427 446 ORT
 7     02 005 I                                          527 5352UMSATZ
 8     02 006 IKUNDEN  KF
 9     02 007 I                                            3   9 KDNR
10     02 008 I                                           21  44 NAME
11     02 009 I                                           73  92 ORT
12     02 010 I                                         P 136 1402UMSATZ

13     03 001 C           ANFANG    TAG
14     03 002 C                     EREADBILD                  01
15     03 003 C           KDNR      CHAINKUNDEN               U20
16     03 004 C                     EREADBILD                  02
17     03 005 C                     EXCPT                      06
18     03 006 C                     GOTO ANFANG

19     04 001 OKUNDEN  E        06N20NP1
20     04 002 O                          KDNR       9
21     04 003 O                          NAME      44
22     04 004 O                          ORT       92
23     04 005 O                          UMSATZ   140P
24     04 006 O        EADD     06 20NP1
25     04 007 O                          KDNR       9
26     04 008 O                          NAME      44
27     04 009 O                          ORT       92
28     04 010 O                          UMSATZ   140P
29     04 011 O        EDEL     06N20 P1
```

Beispiel 5B: Dateiänderungsprogramm mit EXCPT und Überlagerung in der E-Karte

```
STANDARDTEXT (z.B. FIRMA)              KDNR KURZNAME        01.07.81
_____________________________________________________________________

 1     01 001 FBILD                                                PROG05
 2     01 002 FKUNDEN

 3     02 001 E                    SATZ         0140
 4     02 002 E                    D1              2
 5     02 003 E                    KDNR            7
 6     02 004 E                    D2             11
 7     02 005 E                    NAME           24
 8     02 006 E                    D3             28
 9     02 007 E                    ORT            20
10     02 008 E                    D4             43
11     02 009 E                    UMSATZ          9 2

12     03 001 IBILD    KF
13     03 002 I                                        227 233 KDNR
14     03 003 I                                        327 350 NAME
15     03 004 I                                        427 446 ORT
16     03 005 I                                        527 5352UMSATZ
17     03 006 IKUNDEN  KF
18     03 007 I                                          1 140 SATZ

19     04 001 C           ANFANG    TAG
20     04 002 C                     EREADBILD                 01
21     04 003 C           KDNR      CHAINKUNDEN              U20
22     04 004 C                     EREADBILD                 02
23     04 005 C                     EXCPT                     06
24     04 006 C                     GOTO ANFANG

25     05 001 OKUNDEN  E         06N20NP1
26     05 002 O                            SATZ     140
27     05 003 O        EADD      06 20NP1
28     05 004 O                            SATZ     140
29     05 005 O        EDEL      06N20 P1
```

Beispiel 6: Eingabe - Ausgabe

```
STANDARDTEXT (Z.B. FIRMA)              KDNR KURZNAME        01.07.81
_____________________________________________________________________

 1     01 001 FBILD                                                PROG06

 2     02 001 IBILD    KF
 3     02 002 I                                          151 155 A1
 4     02 003 I                                          251 255 A2
 5     02 004 I                                          350 355 A3
 6     02 005 I                                          450 455 A4
 7     02 006 I                                          551 5550N5
 8     02 007 I                                          650 6540N6
 9     02 008 I                                          749 7530N7
10     02 009 I                                          849 8542N8
11     02 010 I                                          947 9522N9
12     02 011 I                                         104610512N10
13     02 012 I                                         114811542N11
14     02 013 I                                         124712532N12
15     02 014 I                                         134613522N13

16     03 001 C           ANFANG    TAG
17     03 002 C                     EREADBILD                 01
18     03 003 C                     EREADBILD                 02
19     03 004 C                     GOTO ANFANG

20     04 001 OBILD    E E      01
21     04 002 O                                NC 155 '     '
22     04 003 O                          A2    N  255
23     04 005 O                                N  355 '      '
24     04 006 O                          A4    N  455
25     04 008 O                                N  555 '     '
26     04 009 O                          N6    N  655 '     '
27     04 010 O                          N7    N  755 '  .   '
28     04 012 O                                N  855 '    ,  '
29     04 013 O                          N9    N  955 '     ,  '
30     04 014 O                          N10   N 1055 '  .   ,  '
31     04 016 O                                N 1155 '     ,  '
32     04 017 O                          N12   N 1255 '     ,  '
33     04 018 O                          N13   N 1355 '  .   ,  '
34     04 019 O                                  1455 'TEST EINGABE'
35     04 020 O        E E      02
36     04 021 O                          A1     135
37     04 022 O                          A2     235
38     04 023 O                          A3     335
39     04 024 O                          A4     435
40     04 025 O                          N5     535 '     '
41     04 026 O                          N6     635 '     '
42     04 027 O                          N7     735 '  .   '
43     04 028 O                          N8     835 '     ,  '
44     04 029 O                          N9     935 '     ,  '
45     04 030 O                          N10   1035 '  .   ,  '
46     04 031 O                          N11   1135 '     ,  '
47     04 032 O                          N12   1235 '     ,  '
48     04 033 O                          N13   1335 '  .   ,  '
49     04 034 O                                  1435 'ANZEIGE'
```

Beispiel 7: RRDS Datei mit numerischem Keyfeld

Mit CPG können RRDS-Dateien analog zu KSDS-Dateien verarbeitet werden. Lesen ist sowohl sequentiell als auch im direkten Zugriff möglich. Veränderungen sind in der Form Update, Hinzufügen und Löschen eines Satzes unterstützt, d.h. es kann beim CHAIN auch festgestellt werden, ob ein Satz vorhanden ist oder nicht.

Die einzelnen Funktionen sind im Folgenden anhand von Programmfragmenten erläutert:

```
H                                              RRDS BEISPIEL
FBILD    C                        DISPLAY
FRRDS    U   F     100 04KV       RRDS
IRRDS    KF
I                                       1 100 SATZ
```

Sequentielles Lesen:

```
C                    Z-ADD1         RRN     90
C           READ     TAG
C           RRN      READ RRDS                        READ
C   EF               GOTO EOF
C                    GOTO READ
C           EOF      TAG
C                    RNDOMRRDS
```

Sätze hinzufügen

```
C           ADD      TAG
C           RRN      ADD  1         RRN
C           RRN      CHAINRRDS                50
C   50               EXCPT                    90
C  N50               GOTO ADD
```

Direkter Zugriff:

```
C           CHAIN    TAG
C                    Z-ADD200       RRN
C           RRN      CHAINRRDS                50
C   50               GOTO NOTFND
```

Update:

```
C           UPDATE   TAG
C           RRN      CHAINRRDS               U50
C  N50               EXCPT                    91
C           NOTFND   TAG
```

Löschen eines Datensatzes:

```
C           LÖSCH   TAG
C           RRN      CHAINRRDS               U50
C  N50               DELETRRDS
```

Zugehörige Output-Karten:

```
ORRDS    EADD     90
O                        SATZ     100
ORRDS    E        91
O                        SATZ     100
```

Beispiel 8: ESDS-Datei

```
H                                              SAMPLE ESDS PROGRAM
FBILD    C                        DISPLAY
FESDS    U   F     100 04RV       ESDS
IESDS    KF
I                                       1 100 SATZ
```

Hinzufügen eines Satzes. Nach dem Hinzufügen wird die aktuelle RBA (relative Byte-Adresse) im Feld CPGKxx (xx = lfd. Nr. der Datei im Programm) zur Verfügung gestellt:

```
C           ADD       TAG
C                     EXCPTNEU                          ADD NEW.REC
C                     MOVE CPGK02    LRBA    4          SAVE RBA
```

Direkter Zugriff. Ist das Schlüsselfeld alphanumerisch, so muss es mit einem numerischen Feld binär gefüllt werden (EDIT). Bei numerischem Schlüssel wird die Aufbereitung intern vorgenommen. Mit dem numerischen Feld RBAN könnte im Beispiel direkt zugegriffen werden.

```
C           CHAIN     TAG
C                     EDIT           KEY
C           KEY       CHAINESDS                50
C   50                GOTO NOTFND
```

Update:

```
C           UPDATE    TAG
C                     EDIT           KEY
C           KEY       CHAINESDS              U50
C                     EXCPT                   91
```

Sequentiell lesen:

```
C                     Z-ADD0         RBAN    90         ERSTE RBA
C                     FILL X'00'     KEY     4          ERSTE RBA
C           READ      TAG                               READ----------
C           KEY       READ ESDS
C   EF                GOTO EOF
C                     GOTO READ
C           EOF       TAG                               EOF-----------
C                     RNDOMESDS                         ENDE SEQ.
C           NOTFND    TAG
```

Zugehörige Output-Karten:

```
OESDS    EADD             NEU
O                         SATZ     100
OESDS    E        91
O                         SATZ     100
OKEY     F
O                         RBAN       4B
O* ODER VORHERIGER SATZ   LRBA       4
O* ODER 2. SATZ                      4X'00000064'
O* ODER 41. SATZ BEI 4K CI SIZE      4X'00001000'
```

Beispiel 8A: ESDS-Datei mit numerischem Keyfeld

```
H                                                SAMPLE ESDS PROGRAM
FBILD    C                       DISPLAY
FESDS    U   F     100 04RV      ESDS
IESDS    KF
I                                       1 100 SATZ
I*-------------------------------------------------------------------
C           ADD       TAG
C                     EXCPT                      90     ADD NEW.REC
C                     MOVE CPGK02    LRBA    4          SAVE RBA
C           LRBA      CHAINESDS                     50
C*------------------------------------------------------------------
C           CHAIN     TAG
C           RBAN      CHAINESDS                  50
C   50                GOTO NOTFND
C*------------------------------------------------------------------
C           UPDATE    TAG
C           RBAN      CHAINESDS                U50
C                     EXCPT                     91
C*-------------------------------------------------------------------
C                     Z-ADD0         RBAN    90        ERSTE RBA
C           READ      TAG                              READ----------
C           RBAN      READ ESDS
C   EF                GOTO EOF
C                     GOTO READ
C           EOF       TAG                              EOF-----------
C                     RNDOMESDS                        ENDE SEQ.
C           NOTFND    TAG
C*-------------------------------------------------------------------
OESDS    EADD     90
O                         SATZ     100
OESDS    E        91
O                         SATZ     100
```

Beispiel 8B: Hinzufügen in einer ESDS-Datei. Es wird die Möglichkeit geboten, die zehn zuletzt hinzugefügten Sätze zu durchblättern.

```
    H*-------------------------------------------------------------*
    H*            ESDS DATEI  10 SATZE BLÄTTERN                    *
    H*-------------------------------------------------------------*
 1  FBILD    C                        L3270
 2  FCPGESD  U   V      10  4RV       ESDS
 3  D        FG         10  4
 4  IBILD    KF
 5  I                                    101 105 SATZ
 6  ICPGESD  KF
 7  I                                      6  10 SATZ
 8  C           A100      TAG
 9  C                     EREADBILD                01
10  C   DE                GOTO A200                     NORM. EINGABE
    C*--BLÄTTERN--------------------------------------------------*
11  C   P1      I         ADD  1         I       30       RÜCKW.
12  C   P2      I         SUB  1         I                VORW.
13  C           I         COMP 0                    5050
14  C  N50      I         COMP 10                   50
15  C   50                GOTO A100                     FEHLERMELDUNG
16  C                     MOVE FG,I      KEY     4
17  C           KEY       COMP '    '                   50
18  C   50                GOTO A100                     FEHLERMELDUNG
19  C           KEY       CHAINCPGESD               50
20  C                     GOTO A100
    C*--BLÄTTERN ENDE---------------------------------------------*
21  C           A200      TAG
22  C                     EXCPT                     02
23  C           CPGK02    CHAINCPGESD              C50
24  C                     ROLLB          FG
25  C                     MOVE CPGK02    FG,1
26  C                     Z-ADD0         I
27  C                     GOTO A100
28  OBILD    E E      01
29  O                                   550 'EINGABE ESDS DATEI  '
30  O                                   750 'PF1 = 1 SATZ RÜCKW.'
31  O                                   850 'PF2 = 1 SATZ VORW.  '
32  O                50            L 1051 'FEHLER BEIM BLÄTTERN'
33  O                        SATZ  AC 105
34  OCPGESD  EADD     02
35  O                                     2 'BZ'
36  O                        SATZ        10
```

Erklärung des Programmteils 'Satz hinzufügen' (Karten 21-27,34-36)

Der Satz, der aus der Map eingelesen wurde, wird der ESDS-Datei hinten hinzugefügt.

Das CHAIN in Statement 23 wird durchgeführt, damit der Satz tatsächlich auf die Platte geschrieben wird; ohne das CHAIN wäre der neue Satz zu diesem Zeitpunkt nur im Speicher hinzugefügt. Mit dem 'C' in Spalte 53 des CHAIN-Befehls erreicht man, dass keine Daten eingelesen werden.

Die Feldgruppe FG nimmt die RBAs der letzten zehn hinzugefügten Saetze auf. Mit dem ROLL-BACK werden die bereits geretteten RBAs in der Feldgruppe nach hinten geschoben (Statement 24). Die aktuelle RBA wird im ersten Element der RBA-Feldgruppe abgestellt.

Beispiel 9: Beispielprogramm mit SYNCP

```
H                                                BEISPIELPROGRAMM SYNCP

FBILD    C                       DISPLAY
FARTIKEL U   F     500 08        DISK
FAUFTRAG U   F     200 04        DISK
FKUND    I   F     100           STORAGE

E                   PAGE       15 78

IKUND    KF
I                                       1   7 KDNR
I                                      11  30 NAME
I                                      31  50 ORT
I                                      51  70 STR
I                                      71  74 COND
I                                      75  77 VTR
I                                      78  83 AUFNR
IARTIKEL KF
I                                       1   8 ARTNR
I                                    P  41  452MENGE
I                                    P  46  502ZUGANG
I                                    P  51  552ABGANG
IAUFTRAG KF
I                                       1   6 AUFNRA
I                                       7  13 ARTNRA
I                                    P  14  182MENGV
I                                    P  19  232BESTM
IBILD    KF
I                                   23212328 ARTNR
I                                   233123392BESTM
ICPGCSA  F
I                                      50  50 TBYTE
I                                    P  51  540NR
C*-------------------------------------------------------------------*
C* EIN VORPROGRAMM ERSTELLT DEN AUFTRAGSDATENSATZ ZU EINEM AUF-     *
C* TRAG.   IN TEMPORARY STORAGE WERDEN DIE DATEN DES KUNDEN ZUR     *
C* VERFÜGUNG GESTELLT.                                              *
C*-------------------------------------------------------------------*
C                   READ KUND                        KUNDEN-DATEN
C           ANFANG  TAG                              ANFANG
C                   EREADBILD                 01     MENGE EINGEBEN
C           ANF001  TAG                              FEHLERKORREKTUR
C           ARTNR   CHAINARTIKEL             U50     ARTIKEL LESEN
C   50              GOTO ERROR                       NICHT GEFUNDEN
C           ABGANG  ADD  BESTM     ABG     92        ABGANG RECHNEN
C*-------------------------------------------------EINGABE PRÜFEN
C           BESTM   COMP 0         ERROR        5151BESTELLMENGE
C           MENGE   ADD  ZUGANG    SMENG   92       ZUGANG
C           SMENG   SUB  ABGANG    SMENG            ABGANG
C           BESTM   COMP SMENG     ERROR      52    BESTAND PRÜFEN
C*-----------------------------------------------------OK-------------
C                   SELCT          CPGCSA           SCHLÜSSEL
C           NR      ADD  1         NKEY    70       + 1
C                   EDIT           CPGCSA           ZURÜCK
C           NR      CHAINAUFTRAG             U54     LESEN
C   54              GOTO ERROR                       NICHT GEFUNDEN
C*-----------------------------------------------------UPDATE---------
C                   EXCPT                     9091   UPDATE
```

```
C                   SYNCP                           SYNC POINT
C                   GOTO ANFANG                     NÄCHSTER
C           ERROR   TAG                             FEHLER---------
C                   EDIT          FEHLER 24         FEHLERMELDUNG
C                   RNDOMAUFTRAG                    RESET AUFTRAG
C                   RNDOMARTIKEL                    RESET ARTIKEL
C                   EREADBILD               02      FEHLERMELDUNG
C                   GOTO ANF001                     NOCHMAL
C*-------------------------------------------------------------------
OBILD    E E    01
O                       UDATE    180
O                       UTIME    170
O                       CPGTID   160
O                                155 'Z . . '
O                                148 'A N G   F I R M A   X Y '
O                                124 'A U F T R A G S E I N G '
O                                204X'3CC25F5C'
O                                316 'KDNR...........:'
O                                416 'NAME...........:'
O                                516 'ORT............:'
O                                356 'KONDITIONEN....:'
O                                456 'VERTRETER......:'
O                                556 'STRASSE/POSTF..:'
O                       KDNR  P  324
O                       NAME  P  437
O                       ORT   P  537
O                       COND  P  361
O                       VTR   P  460
O                       STR   P  577
O                                604X'3CC75F5C'
O                       PAGE  L  779
O                                2104X'3C5A4F5C'
O                       FEHLERPB2280
O                       AUFNR P 2316
O                       ARTNR A 2328
O                       BESTM A 2340
O       E  S    02
O                       FEHLERPB2280
O                51             C2331
O                50             C2321
O*-------------------------------------------------------------------
OARTIKEL E      90
O                       ABG    B  55P
OAUFTRAG E      91
O                       AUFNR      6
O                       ARTNR  B  13
O                       BESTM  B  19P
O                       KDNR      26
O                       SMENG  B 100P
O*-------------------------------------------------------------------
OFEHLER  F
O                                  24 'PROGRAMM FEHLER         '
O                50                24 'ARTIKEL NICHT VORHANDEN '
O                51                24 'BESTELLMENGE < = 0 ???  '
O                52                24 'BESTAND NICHT VORHANDEN '
O                54                24 'AUFTRAGS DATEI VOLL     '
O*-------------------------------------------------------------------
OCPGCSA  F
O                       NKEY      54P
```

Beispiel 10: Drucken im Linemode

```
H                         O

FBILD    C                          DISPLAY
FL86C    O   F     132              PRINTER

LL86C    072FL010010350207012

C                     EXCPT

OBILD    E  E
O                                    550 'BEISPIEL DRUCKER'
O                                    650 'L I N E M O D E '
OL86C    E 3501
O                                     30 'VORSCHUB VOR DEM DRUCKEN'
O                                     60 'NACH KANAL 01 = ZEILE 10'
O                                     90 '3 ZEILEN VOR DEM DRUCKEN'
O                                    120 '5 ZEILEN NACH DEM DRUCK.'
O        E 2102
O                                     30 'VORSCHUB VOR DEM DRUCKEN'
O                                     60 'NACH KANAL 02 = ZEILE 35'
O                                     90 '2 ZEILEN VOR DEM DRUCKEN'
O                                    120 '1 ZEILEN NACH DEM DRUCK.'
O        E  2
O                                    120 '2 ZEILEN NACH DEM DRUCK.'
O        E  112
O                                     30 'VORSCHUB VOR DEM DRUCKEN'
O                                     60 'NACH KANAL 12 = ZEILE 70'
O                                    120 '1 ZEILEN NACH DEM DRUCK.'
```

Beispiel 11: Drucken im Buffermode

```
H                         O

FBILD    C                          DISPLAY
FL86C    OB  F     132              PRINTER

C                     EXCPT

OBILD    E  E
O                                    550 'BEISPIEL DRUCKER'
O                                    650 'B U F F E R-MODE'
OL86C    E
O                                    130 'DRUCKEN ZEILE 1'
O                                    260 'DRUCKEN ZEILE 2'
O                                    301 ' '
O                                    420 'DRUCKEN ZEILE 4'
O                                    630 'DRUCKEN ZEILE 5'
O                                    740 'DRUCKEN ZEILE 6'
```

Beispiel 12: Cursor-Stop

```
H                                                BEISPIELPROGRAMM
H*---------------------------------------------------------------------*
H*             BEISPIELPROGRAMM MIT CURSOR-STOP                        *
H*            NACH DER LETZTEN BILDSCHIRMEINGABE                       *
H*---------------------------------------------------------------------*
FBILD    C                          DISPLAY
IBILD    KF
I                                        114 118 EING1
I                                        214 218 EING2
I                                        314 318 EING3
C           OBEN       TAG
C                      EREADBILD
C                      GOTO OBEN
OBILD    E E
O                                   110 '1 EINGABE:'
O                          EING1 AC 118
O                                   210 '2 EINGABE:'
O                          EING2 A  218
O                                   310 '3 EINGABE:'
O                          EING3 A  318
O                                 0  319
```

Erläuterungen:

STMT 21 Eine Stelle hinter der letzten Eingabe ist das Bildschirmattribut '0' auszugeben.

Der Cursor springt nach der dritten Eingabe eine Stelle nach rechts und die Tastatur wird für weitere Eingaben blockiert.

Mit der Sprungtaste kann der Cursor zu einem Eingabefeld bewegt werden, um eine fehlerhafte Eingabe zu korrigieren.

Durch den Cursor-Stop wird verhindert, dass durch irrtümliches Betätigen einer Taste die ersten Bildschirmfelder überschrieben werden.

Beispiel 13: COMRG

Die Operation COMRG stellt eine Communication Region zur Verfügung die dem Programmierer den Zugriff auf interne Daten ermöglicht. Das folgende Beispiel zeigt, wie über diese Operation die Operator-Id abgefragt werden kann.

```
H                                              OPERATOR ID.

FBILD    C                       DISPLAY

IFELD    F
I                                        1   3 OPID

C                    COMRG          FELD   32        COMMUN.REGION
C                    SELCT          FELD             AUSWAHL
C           OPID     COMP 'XYZ'     ENDE       ****  OPERATOR ID.
C*
C*    P R O G R A M M
C*
C           ENDE     TAG                             ENDE
```

Beispiel 13a: ELIM

Die Operation ELIM soll den Wert Hex-Null durch Blanks ersetzen:

```
    C                   ELIM X'00'     FELD
```

Erläuterung:

Wenn 'FELD' vorher den Wert X'C1C2C3000000' enthielt, wird der Inhalt in X'C1C2C3404040' geändert.

Beispiel 14: EXITI

```
C                    MOVE 'TT01'    TRID    4
C                    MOVE 'RZ01'    TERM    4
C                    Z-ADD0123000   TIME    70
C                    Z-ADD10101     INTERV  70
C*-------------------------------------------------------------------
C           'RZ01'   EXITI'TT01'                      AN TERM. RZ01
C*-------------------------------------------------------------------
C                    EXITI'TT01'    '1230'    T      UM 12.30 UHR
C*-------------------------------------------------------------------
C           TERM     EXITITRID      TIME      T      TERM X UM X UHR
C*-------------------------------------------------------------------
C           TERM     EXITITRID      INTERV           IN X STD,MIN,S
C*-------------------------------------------------------------------
C           TERM     EXITITRID      INTERV    N      WEITER IM PROGR
```

Erläuterung:

Das Beispiel zeigt fünf mögliche Codierungen einer EXITI-Operation.

Die erste Version

```
C           'RZ01'    EXITI'TT01'                    AN TERM. RZ01
```

'sendet' das Programm TT01 zum Terminal RZ01 und beendet danach die Arbeit.

Die zweite Version

```
C                     EXITI'TT01'    '1230'    T      UM 12.30 UHR
```

setzt um 12.30 Uhr die Verarbeitung am eigenen Terminal mit dem Programm TT01 fort. Die Eintragung 'T' in Spalte 53 besagt, dass der Wert im Ergebnisfeld als Uhrzeit und nicht als Zeitintervall (12 Minuten 30 Sekunden) interpretiert wird.

Die dritte Version

```
C           TERM      EXITITRID      TIME      T       TERM X UM X UHR
```

setzt zu der im Feld TIME angegebenen Uhrzeit (T in 53) die Verarbeitung an dem im Feld TERM angegebenen Terminal die Verarbeitung mit dem Programm fort, dessen Transaction Identification im Feld TRID steht. Der Programmierer muss dabei sicherstellen, dass das Feld TERM eine gültige Terminal-Id, das Feld TIME eine gültige Uhrzeit und das Feld TRID eine gültige Trans-Id enthält.

Die vierte Version

```
C           TERM      EXITITRID      INTERV             IN X STD,MIN,S
```

entspricht der vorherigen mit der Ausnahme, dass das Ergebnisfeld nicht eine Uhrzeit, sondern ein Zeitintervall in der Form XX.XX.XX (Stunden. Minuten. Sekunden) enthält (Blank statt 'T' in Spalte 53).

Die fünfte Version

```
C           TERM      EXITITRID      INTERV    N       WEITER IM PROGR
```

entspricht der vorherigen, mit der Ausnahme, dass nach der EXITI-Operation die Verarbeitung im sendenden Programm nicht abgebrochen, sondern mit der nächsten Operation fortgesetzt wird (N in Spalte 53).

Beispiel 15: LOKUP 8088

Das folgende Beispiel zeigt die Verarbeitung einer programminternen Tabelle mit Hilfe der Lokup-Operation. Im Beispiel wird die Tabelle über eine EDIT-Operation erstellt.

```
    H                                              TEST LOKUP
    FBILD    C                      DISPLAY
    D        FGA0        0 40       TABELLE ALPHA
    D        FGA        10  4       FELDGRUPPE ALPHA
    D        FGN0        0 30       TABELLE NUMERISCH
    D        FN         10  5 2     FELDGRUPPE NUMERISCH
```

```
D        FGN        10  5 2       FELDGRUPPE NUMERISCH
C                     EDIT           FGA0
C                     EDIT           FGN0
C                     MOVE 'FFFFFFFF'FA        4
C           'CCCC'    LOKUPFGA                      20
C* ODER
C*          000,09    LOKUPFGN                   21  22
C*          FA        LOKUPPG,I                      23
C*          NUM       LOKUPFGN,I                     24
C*          FA        LOKUPFGA                       25
C*          FN,1      LOKUPFGN                       26
C*          FN,3      LOKUPFGN,1                     27
C*          FN,I      LOKUPFGN,5                     28
C*          FN,X      LOKUPFGN,I                     29
C*          PG,I      LOKUPFGA,5                     30
C*          FA        LOKUPFGA,19                    31
C*          'UUUU'    LOKUPFGA,I                     32
C*          FA        LOKUPFGA,I                     33
C*          FA        LOKUPFGA,I                     34
C*          'UUUU'    LOKUPFGA,I                     35
C*          000,10    LOKUPFGN,I                     36
C*          XX        LOKUPFGN,I                     37
C*          000,10    LOKUPFGN,I                     38
C                     EREADBILD                   39

OBILD    E E      39
O                 20              544 'GEFUNDEN'
OFGA0    F
O                                   20 'AAAABBBBCCCCDDDDEEEE'
O                                   40 'FFFFGGGGHHHHIIIIJJJJ'
OFGN0    F
O                                    9X'00005C00006C00007C'
O                                   18X'00008C00009C00010C'
O                                   27X'00011C00012C00013C'
O                                   30X'00014C'
```

Nach Ausführung dieses Programms sind die Bezugszahlen 20 und 22 bis 38 gesetzt.

Beispiel 16: MLLZO 8089

Die Operation MLLZO soll für ein numerisch definiertes Feld ein Vorzeichen C in F umwandeln.

```
              C                    MLLZO1         FELD
```

Erläuterung:

```
Feldinhalt vorher:   X'0000500C'
           nachher   X'0000500F'
```

Beispiel 17: READ

```
STANDARDTEXT (Z.B. FIRMA)            KDNR KURZNAME        01.06.81
```

```
 1     01 001 E                    PAGE       20 70

 2     02 001 IKUNDEN  SF
 3     02 002 I                                        3   9 KDNR
 4     02 003 I                                       21  44 NAME
 5     02 004 I                                       73  92 ORT
 6     02 005 I                                    P 135 1392UMSATZ

 7     03 001 C           READ      TAG
 8     03 002 C           KDNR      READ KUNDEN    PAGE
 9     03 003 C                     EREADBILD
10     03 004 C                     GOTO READ

11     04 001 OBILD    E E
12     04 002 O                        PAGE      571
13     04 003 OPAGE    F
14     04 004 O                        KDNR        7
15     04 005 O                        NAME       33
16     04 006 O                        ORT        55
17     04 007 O                        UMSATZ     70 ' .   . 0 ,  -'
```

Erläuterungen:

1 Erweiterte Dateizuordnung: Eine Feldgruppe mit dem Namen PAGE wird definiert. Die Feldgruppe besteht aus 20 Feldern mit einer Länge von je 70 Bytes. Die Feldgruppe soll eine Bildschirmseite mit 20 Zeilen aufnehmen.

2 Von der Datei KUNDEN soll ein Satz gelesen werden

3 Im Kundensatz steht die Kundennr (KDNR) von Pos. 3 bis Pos. 9.

4 Der Name steht von Pos. 21 bis Pos. 44

5 Der Ort steht von Pos. 73 bis Pos. 92

6 Der Umsatz des Kunden steht von Pos. 135 bis Pos. 139 in gepacktem Format (P). Das Feld wird numerisch definiert mit 2 Dezimalstellen.

8 Von der Datei KUNDEN werden so viele Sätze gelesen, wie die im Ergebnisfeld eingtragene Feldgruppe Elemente enthält. Der erste gelesene Satz ist der, dessen Schlüssel gleich oder größer als der Inhalt des Feldes 'KDNR' ist. Die Seite wird in der Feldgruppe 'PAGE' gespeichert. Die Anzahl der Zeilen ergibt sich aus der unter Nr.1 definierten Feldanzahl. Zur Aufbereitung der Zeilen siehe Nr.13 bis 17.

12 Die Feldgruppe PAGE wird ausgegeben. Das erste Feld steht in Zeile 5 von Pos. 2 bis Pos. 71 (571). Die folgenden Felder werden in den gleichen Positionen der jeweils folgenden Zeilen ausgegeben z.B. Feld 2 in Zeile 6, Feld 3 in Zeile 7 usw.

13 Das Feld bzw. die Feldgruppe PAGE soll aufbereitet werden (F in Spalte 15). Die Feldaufbereitung wird ausgelöst durch den Befehl

READ.

14 Das Feld KDNR wird in Pos.1 bis 7 aller Felder der Feldgruppe ausgegeben

15 Wie 14, Feld NAME, letztes Byte auf Pos. 33

16 Wie 14, Feld ORT, letztes Byte auf Pos. 55.

17 Wie 14, Feld UMSATZ, jedoch aufbereitet mit Schablone.

Beispiel 18: READB

```
STANDARDTEXT (Z.B. FIRMA)              KDNR KURZNAME        01.06.81
```

```
 1     01 001 E                    PAGE       20 70

 2     02 001 IKUNDEN  SF
 3     02 002 I                                          3   9 KDNR
 4     02 003 I                                         21  44 NAME
 5     02 004 I                                         73  92 ORT

 6     03 001 C                     MOVE '9999999' KDNR
 7     03 002 C           READ      TAG
 8     03 003 C           KDNR      READBKUNDEN    PAGE
 9     03 004 C                     EREADBILD
10     03 005 C                     GOTO READ

11     04 001 OBILD    E E
12     04 002 O                        PAGE      571
13     04 003 OPAGE    F
14     04 004 O                        KDNR        7
15     04 005 O                        NAME       33
16     04 006 O                        ORT        55
```

Erläuterungen:

1 Erweiterte Dateizuordnung: Eine Feldgruppe mit dem Namen PAGE wird definiert. Die Feldgruppe besteht aus 20 Feldern mit einer Länge von je 70 Bytes. Die Feldgruppe soll eine Bildschirmseite mit 20 Zeilen aufnehmen.

2 Von der Datei KUNDEN soll ein Satz gelesen werden

3 Im Kundensatz steht die Kundennr (KDNR) von Pos. 3 bis Pos. 9.

4 Der Name steht von Pos. 21 bis Pos. 44

5 Der Ort steht von Pos. 73 bis Pos. 92

6 Das Schlüsselfeld muss einen gültigen Schlüssel enthalten, von dem aus die Datei rückwärts gelesen wird.

8 Von der Datei KUNDEN werden so viele Sätze rückwärts gelesen, wie die im Ergebnisfeld eingetragene Feldgruppe Elemente enthält. Der erste gelesene Satz ist der, dessen Schlüssel gleich

dem Inhalt des Feldes KDNR ist. Die Seite wird in der Feldgruppe PAGE gespeichert. Die Anzahl der Zeilen ergibt sich aus der unter Nr.1 definierten Feldanzahl. Zur Aufbereitung der Zeilen siehe Nr.13 bis 16.

12 Die Feldgruppe 'PAGE' wird ausgegeben. Das erste Feld steht in Zeile 5 von Pos. 2 bis Pos. 80(571). Die folgenden Felder werden in den gleichen Positionen der jeweils folgenden Zeilen ausgegeben z.b. Feld 2 in Zeile 6, Feld 3 in Zeile 7 usw.

13 Das Feld bzw. die Feldgruppe PAGE soll aufbereitet werden (F in Spalte 15). Die Feldaufbereitung wird ausgelöst durch den Befehl READB.

14 Das Feld KDNR wird in Pos.1 bis 7 aller Felder der Feldgruppe ausgegeben

15 Wie 14, Feld Name, letztes Byte auf Pos. 33

16 Wie 14, Feld Ort, letztes Byte auf Pos. 55.

Beispiel 19: REPLC 8092

Die Operation REPLC soll den Wert Blank durch HEX-Null ersetzen:

```
              C                    REPLCX'00'     FELD
```

Erläuterung:

```
Feldinhalt vorher:   X'C1C2C3404040'
           nachher   X'C1C2C3000000'
```

Beispiel 20: SYNCP

Die Operation SYNCP gibt bei der System Recovery an, bis zu welchem Punkt die Verarbeitung abgeschlossen war.

```
C           ANFANG    TAG                              ANFANG
C                     EREADBILD                 01     MENGE EINGEBEN
C           ARTNR     CHAINARTIKEL             U50     ARTIKEL LESEN
C           MENGE     ADD  ZUGANG    SMENG   92        ZUGANG
C           NR        CHAINAUFTRAG               U54   LESEN
C                     EXCPT                       9091 UPDATE
C                     SYNCP                            SYNC POINT
C                     GOTO ANFANG                      NÄCHSTER
```

Beispiel 21: Unformatisierte Ausgabe

In Zeile 1 Position 4 soll das Feld 'TRID' und in Zeile 2 Position 4 die Konstante 'TEST' ausgegeben werden.

```
OBILD    EU
O                        TRID       4
O                                  84 'TEST'
```

Beispiel 22: Update VSAM variable Satzlänge, wenn die Satzlänge verändert werden soll.

Erste Möglichkeit:

```
FDATEI   U   V     100 10         DISK

IDATEI   KF
I                                      1  10 KEY

C           KEY       CHAINDATEI    NOTFND      50
C   P1                EXCPT                     01
C   P2                EXCPT                     02
C           NOTFND    TAG

ODATEI   EALG     01
O                                     50 'SATZLÄNGE 50'
O        EALG     02
O                                    100 'SATZLÄNGE 100'
```

Zweite Möglichkeit: Die tatsächliche Länge des Ausgabesatzes wird dem internen Feld CPGVRL entnommen.

```
FDATEI   U   V     100 10         DISK

IDATEI   KF
I                                      1  10 KEY

C           KEY       CHAINDATEI     NOTFND     50
C   P1                =    50        CPGVRL
C   P2                =    100       CPGVRL
C           NOTFND    TAG

ODATEI   EALG                   V
O                         CPGVRLZ   50
O                                   46 'Satzlänge'
```

Beispiel 23: DO

Die Operation DO erlaubt es, eine Gruppe von Rechenbestimmungen mehrere Male auszuführen. Das folgende Beispiel zeigt, wie eine Schleife programmiert werden kann.

```
C                   DO   10
C           X       ADD  1         X      50
C                   END
C           6       DO   10
C           XX      ADD  1         XX     50
C                   END  2
```

Beispiel 24: DOU

Die Operation DOU erlaubt es, eine Gruppe von Rechenbestimmungen einmal oder mehrmals auszuführen, bis eine bestimmte Relation zwischen Faktor 1 und Faktor 2 besteht.

```
C                   Z-ADD0         SALDO 112
C                   MOVEL'5000'    KEY
C           KEY     DOUGT'5999'
C           KEY     READ OPODAT
C           SALDO   ADD  BETRAG    SALDO
C                   END
```

Beispiel 25: DOW

Die Operation DOW erlaubt es, eine Gruppe von Rechenbestimmungen solange auszuführen, wie eine bestimmte Beziehung zwischen Faktor 1 und Faktor 2 besteht.

```
C           KEY     READ AUFTRAG
C           KDNR    DOWEQ'4711'
C           SUMME   ADD  BETR      SUMME
C           KEY     READ AUFTRAG
C                   END
```

In dem oben gezeigten Beispiel wird die Summe aller Aufträge des Kunden 4711 gebildet.

Beispiel 26: IF

Die Operation IF ermöglicht die Ausführung einer Gruppe von Rechenbestimmungen unter der Bedingung, dass eine gewisse Beziehung zwischen Faktor 1 und Faktor 2 besteht.

```
C           F1        IFEQ F2
C                     .
C                     .
C                     .
C                     END
```

Es ist zu beachten, dass die Operation nur dann ausgeführt wird, wenn Faktor 1 und Faktor 2 gleich sind.
Ansonsten wird zur ersten Rechenbestimmung nach der zugehörigen Endbestimmung verzweigt.

Beispiel 27: ELSE

```
C           F1        IFEQ F2
C                     .
C                     .
C                     .
C                     ELSE
C                     .
C                     .
C                     .
C                     END
C                     .
C                     .
C                     .
```

Die ELSE-Anweisung wird unmittelbar nach den Rechenbestimmungen eingetragen, die ausgeführt werden, wenn der Vergleich in der IF-Operation positiv ist.

Sofort nach der ELSE-Anweisung folgen die Rechenbestimmungen, die ausgeführt werden sollen, wenn der Vergleich in der IF-Operation negativ ist.

Beispiel 28: Kundenanzeige mit 7-farbigem Bildschirm

```
 1      H                                                KUNDENANZEIGE

 2      FBILD
 3      FKUNDEN

 4      IBILD    KF
 5      I                                    319 325 KDNR
 6      IKUNDEN  DD

 7      C           ANFANG    TAG
 8      C                     EREADBILD
 9      C           KDNR      CHAINKUNDEN              71
10      C           LIEFSP    COMP 'X'                     72
11      C                     GOTO ANFANG

12      OBILD    E E
13      O                         KDANZ    MAP 'FIB'
14      O          U              KDNR  A  325
15      O         $RY
16      O                         KNAME LB 425
17      O                         KANSCHLB 525
18      O                         KSTRA LB 625
19      O                         KPLTZ LB 704
20      O                         KORT  LB 725
21      O         $BR
22      O                 72              2480 'KUNDE HAT LIEFERSPERRE'
23      O                 71              2480 'KUNDENNR IST FALSCH'
```

Erläuterungen:

6 Die für die Kundenanzeige benutzten Felder (KNAME, KANSCH, KSTRA, KPLTZ und KORT) werden aus dem Data Dictionary in das Programm eingefügt (nur möglich für TOP Benutzer).

13 Ausgabe der Map KDANZ **(veraltet)**
An dieser Stelle wird auf dem Bildschirm die Kundenanzeigemaske ausgegeben. Diese Maske wurde durch die 'FIB' erstellt und kann jederzeit mit dem CPG..Top Utility 'TPMP' geändert werden.

14 Das Feld KDNR wird zur Eingabe unterstrichen dargestellt. Hiermit wird das Formatisieren des Feldes (z.B. mit Punkten erspart). Die Farbe ist weiss (Default-Wert).

15 Set Attribut Order: Hiermit werden die folgenden Felder (KNAME, KANSCH, KSTRA, KPLTZ, KORT) reversiv und gelb am Bildschirm ausgegeben.

16 Das Feld KNAME wird mit dem Attribut L ausgegeben, hierdurch wird erreicht, das die durch Set Attribut Order definierte Ausgabe (reversiv und gelb) nur in der Länge des Feldes am Bildschirm angezeigt wird.
Bei ungeschützten Feldern geht die Set Attribut Order über die Feldlänge hinaus, und zwar bis zum nächsten Bildschirmattribut.

21 Für Fehlermeldungen wird hier Set Attribut Order als blinkend und rot definiert. Die Meldungen werden somit gleich erkannt.

Ein weiteres Attribut ist hier nicht erforderlich, da die blinkende und reversive Anzeige durch das Ende des Bildschirms begrenzt ist.

Beispiel 29: Temporary Storage Queuing

```
 1      H                                                TEXTSYSTEM

 2      FBILD
 3      FSTOR    UQ  F      80            STORAGE

 4      D        PAGE       16 78

 5      ISTOR    KF
 6      I                                     1  78 ZEILE
 7      IBILD    KF
 8      I                                   157 160 TERM
 9      I                                   302 379 PAGE

10      C                     MOVE CPGTID    TERM    4
11      C                     FILL ' '       PAGE
12      C                     DO   16        I       30
13      C           I         READ STOR
14      C   EF                GOTO WEITER
15      C                     MOVE ZEILE     PAGE,I
16      C                     END
17      C           WEITER    TAG
18      C                     EREADBILD                01
19      C                     PURGESTOR
20      C                     DO   16        I       30
21      C                     MOVE PAGE,I    ZEILE
22      C                     Z-ADD0         CPGQ02
23      C                     EXCPT                    91
24      C                     END
25      C           TERM      EXITI'TT20'

26      OSTOR    E I      91
27      O                       ZEILE     78
28      OBILD    E E      01
29      O                                150 'NACHRICHT AN BILDSCHIRM:'
30      O                       TERM  AC 160
31      O                       PAGE  A  379
```

Erläuterungen:

Das Beispiel zeigt ein Anwendungsprogramm für Textübertragung von einem Bildschirm zu einem beliebigen anderen, durch Verwendung der Funktionen von Temporary Storage Queuing.

3 Dateizuordnung für Temporary Storage Queing. U in Spalte 16 bedeutet, dass der Storage Bereich sowohl für Eingabe als auch für Ausgabe benutzt wird. Die Eintragung Q in Spalte 16 ist für Temporary Storage Queuing erforderlich. Die Satzlänge ist fest und 80 Stellen groß. Der Eintrag Storage in Spalte 40 - 46 ist erforderlich.

5 Eingabebestimmungen für den Storage-Bereich.

10 In das Feld TERM wird hier das Terminalkennzeichen aus der CICSTCT übertragen.

12 Mit der DO-Operation wird eine Schleife begonnen, mit der ein Index I jeweils um 1 erhöht wird. Die Schleife wird 16 mal durchlaufen.

13 Mit dem Index I wird der Temporary-Storage-Queing-Bereich STOR gelesen, hierbei erfolgt der Zugriff (anders als bei sonstigen Dateizugriffen) direkt auf den Satz mit der laufenden Nummer I.
Um beim nächsten Read den folgenden Satz zu lesen, muss der Index jeweils um 1 erhöht werden (wozu hier die DO- Schleife verwendet wird).
Es ist jedoch auch möglich, die folgenden READs ohne eine Eintragung in Faktor 1 der Rechenbestimmungen durchzuführen - hierdurch wird jeweils der nächste Satz gelesen.

14 Mit dem Schalter EF wird abgefragt, ob das Ende des Storage-Bereichs erreicht ist. Der Schalter EF ist ebenfalls an, wenn bei der READ-Operation ein Fehler festgestellt wurde.

15 Übertrage das Feld ZEILE in das i-te Element der PAGE.

16 Die Operation END kennzeichnet das Ende der DO-Schleife, es erfolgt so lange der Rücksprung zur DO-Operation, bis die Schleife 16 mal durchlaufen ist.

19 Mit der Operation PURGE wird der Queuing-Bereich STOR gelöscht.

20 Diese DO-Schleife wird ebenfalls 16 mal durchlaufen, der Index wird in I gespeichert.

22 Löschen der internen Queue-Nummer. Dies bewirkt, dass die Sätze bei den nachfolgenden Ausgaben nicht verändert, sondern hinzugefügt werden.

23 Ausgabe eines Satzes.

24 Ende der zweiten DO-Schleife

25 Mit der Operation EXITI wird das gleiche Programm (TT20) an einem anderen Bildschirm gestartet. Die Terminal-Id des anderen Bildschirms ist im Feld TERM gespeichert.

26 Ausgabebestimmungen für den Queuing-Bereich. Die Eintragung I in Spalte 17 bewirkt, dass der Bereich STOR für alle Bildschirme verfügbar ist.

Beispiel 30: Edit-Codes

```
 1      H                                          E   OP ANZEIGE

 2      FBILD
 3      FOPODAT

 4      D        PAGE       20 78
 5      D        REDAT          6 0

 6      IOPODAT  DD
 7      IBILD    KF
 8      I                                     121 130 KEY

 9      C           ANFANG    TAG
10      C                     FILL ' '       PAGE
11      C   P1      KEY       SETLLOPODAT
12      C           KEY       READPOPODAT    PAGE
13      C           SEITE     ADD  1         SEITE   30
14      C                     EREADBILD
15      C   EF                RNDOMOPODAT
16      C                     GOTO ANFANG

17      OPAGE    F
18      O                         KDNR        7
19      O                         KNAME      40
20      O                         RENR  2    50
21      O                         REDAT Y    65
22      O                         RBETR A    78
23      OBILD    E E
24      O                         OPANZ    MAP 'FIB'
25      O                         SEITE Z  180
26      O         L               PAGE     379
```

Erläuterungen:

Das Beispiel zeigt die Benutzung der Edit-Codes.

1 In Spalte 49 der H-Karte muss ein E eingetragen werden. Dies ermöglicht die Codierung des Aufbereitungsschlüssels in Spalte 38 der Ausgabe. Das Bildschirmattribut wird in Spalte 16 der Ausgabe eingetragen.

5 Mit dieser D-Spezifikation wird die Länge des Feldes REDAT mit 6 Stellen und null Dezimalen definiert, da sonst bei der Aufbereitung des Datums mit Schlüssel Y eine Fehlermeldung generiert würde.

6 Die Felder der Datei OPDAT werden über die Data-Dictionary-Funktion in das Programm eingefügt. Voraussetzung ist mindestens Service-Level CPG2.

15 Wenn eine Datei EF erreicht, muss sie mit der Operation RNDOM aus der sequentiellen Berarbeitung freigegeben werden.

17 Feldaufbereitung einer PAGE mit Edit-Codes.

18 Aufbereitung der Kundennummer. Bei der Ausgabe wird auf die rechte Stelle der Kundennr (soll immer positiv sein) die Zone F gebracht. Somit erscheint nachher eine gültige Ziffer. Führende Nullen werden unterdrückt.

20 Die Rechnungsnummer wird mit dem Schlüssel '2' so aufbereitet, dass jeweils drei Stellen durch einen Punkt getrennt sind. Es wird kein Vorzeichen gedruckt. Führende Nullen werden unterdrückt.

21 Das Rechnungsdatum wird mit dem Schlüssel Y aufbereitet. Somit erfolgt die Ausgabe in der Form TT.MM.JJ

22 Der Rechnungsbetrag wird mit dem Schlüssel A aufbereitet. Hiermit werden Tausenderpunkte gedruckt und ein negativer Betrag wird mit 'CR' gekennzeichnet. Führende Nullen werden unterdrückt.

24 Durch die Eintragung Map wird die Maske OPANZ des Bereichs 'FIB' in die Bildschirmausgabe eingefügt. **Diese CPG-TOP-Funktion ist veraltet.**

25 Die Seitennummer wird mit dem Schlüssel Z aufbereitet. Es erfolgt lediglich eine Nullenunterdrückung, es wird kein Vorzeichen gedruckt.

26 Das Bildschirmattribut (L = lichtstiftempfindlich) für die PAGE wird in Spalte 16 der O-Karte codiert.

Beispiel 30a: DL1-Anwendungsbeispiel

```
H    X    X              8                    TESTPROGRAMM M 2  PSB
F*********************************************************************
F* TITLE: TEST -XXXX- = 2 PSB                                        *
F*********************************************************************
F*                                                                   *
F*             - C P G T O 8 6 -                                     *
F*             TESTPROGRAMM MIT 2 PSB.                               *
F*                                                  18.09.87/AM.     *
F*                                                                   *
F*********************************************************************
F*********************************************************************
F*                                                                   *
F*      DATENBANKEN:    DBDGEPA                                      *
F*      ------------    DBDTEXT                                      *
F*                                                                   *
F*      P S B      :    PSBGEPA                                      *
F*      ------------    PSBTEXT                                      *
F*                                                                   *
F*********************************************************************
F*********************************************************************
F*                                                                   *
F*     SUBROUTINEN-INHALTSVERZEICHNIS                                *
F*     ------------------------------                                *
F*     SR-NR      INHALT                                             *
F*     -----      ------                                             *
F*     SR010  =   ........................                           *
F*                                                                   *
F*     BEZUGSZAHLEN-NACHWEIS                                         *
F*     ---------------------                                         *
F*     BZ.-Z      VERWENDUNG                                         *
F*     -----      ----------                                         *
F*        01  =   ........................                           *
F*        90  =   ........................                           *
F*     91-99  =   ........................                           *
F*                                                                   *
F*.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.*
F*********************************************************************
```

```
F*****************************************************************
F*****************************************************************
F*                                                               *
FTERMI   C   V24000080            L3270
FPSBGEPA U   F       1            DLI
F*                                                               *
F*****************************************************************
D*
D*       KEYFIELD FÜR DBD-GEPA
D*       ---------------------
D        ZKEYG         10
D*
D*       KEYFIELD FÜR DBD-TEXT
D*       ---------------------
D        ZKEYT1         6
D        ZKEYT2         2
D*
D*---------------------------------------------------------------*
D*****************************************************************
```

```
I*****************************************************************
I*****************************************************************
I*                    E I N G A B E                              *
I*****************************************************************
I*     T E R M I N A L  -  A N Z E I G E                         *
I*****************************************************************
I*
ITERMI   KF
I                                      020402100TENR
I*
I*****************************************************************
I*      DATENBANKSEGMENTE FÜR 1. PSB  (SIEHE -F-KARTE-)          *
I*****************************************************************
I*--- C U S A D R ------------------------------------------------
I*-------------   GESCHAEFTSPARTNER-DB     KUNDENADRESSE
IKDADR       DS
I                                       1   2 CAADSQ
I                                       3   8 CAADUC
I                                       9  16 CAZIPC
I                                       9  12 CAZIP4
I                                      17 136 CAADDR
I                                      17  46 CANAM1
I                                      47  76 CANAM2
I                                      77 106 CASTR
I                                     107 136 CAORT
I                                     137 139 CACNTR
I                                     140 142 CASHPT
I                                     143 150 CUFILA
I*
I*****************************************************************
I*      DATENBANKSEGMENTE FÜR 2. PSB  (OHNE -F-KARTE-)           *
I*****************************************************************
I*--- T E D A T A ------------------------------------------------
I*-------------   TEXTDATENBANK            TEXTDATEN
ITXXTA       DS
I                                       1   2 TDDASQ
I                                       3  62 TDTEXT
I                                      63  80 TDFIL
I*
I*****************************************************************
```

```
C*****************************************************************
C*****************************************************************
C*        START-BILD AUSGEBEN UND SUCHBEGRIFF EINLESEN           *
C*****************************************************************
C*
C           START     TAG
C*          -------------
C                     FILL X'00'     ZKEYG            LÖSCHEN
C                     =    0         TENR             LÖSCHEN
C                     GOTO STAR20
C*
C*   ***********************************************
C*   *                                             *
C*   *  LESEN -DBD-GEPA- MIT DEFINIERTEM -PSBGEPA-   *
C*   *  ========================================   *
C*   *                                             *
C*   ***********************************************
C           STAR10    TAG
C*          -------------
C                     MOVELTENR      ZKEYG            KEYFIELD
C           'CUSTID  'QSSA 'CUSUST  'ZKEYG  10  EQ    ROOT
C           'CUSADR  'USSA                            TWIN
C           GU        DLI            KSADR   1  13    CALL
C           CPGDRC    COMP '  '      STAR20         ** GEFUNDEN
C*                                                     ========
C*
C*  *******************************************************
C*  *                                                     *
C*  *  DAS SEGMENT WURDE NICHT GEFUNDEN, LESEN -DBD-TEXT-    *
C*  *  (FEHLERTEXT) MIT  N I C H T  DEFINIERTEM -PSBTEXT-    *
C*  *  ================================================    *
C*  *                                                     *
C*  *******************************************************
C*     -OSSK7D- = SUCHBEGRIFF FÜR -TEXT-DB-
C*     -91-     = FOLGE-NR. IM TWIN-SEGMENT
C*     ------------------------------------
C                     MOVE 'OSSK6D'  ZKEYT1           KEYFIELD
C                     MOVE '91'      ZKEYT2           KEYFIELD
C           'TEROOT  'QSSA 'TRKEYT  'ZKEYT1  6  EQ    ROOT
C           'TEDATA  'QSSA 'TDDASQ  'ZKEYT2  2  EQ    TWIN
C           GU        DLI  'PSBTEXT 'TXXTA   1  13    NEUES PSB
C*                          =======                    =========
C           CPGDRC    COMP '  '      STAR20         ** GEFUNDEN
C                     MOVELCPGDRC    TDTEXT           NICHT GEF
C*
C           STAR20    TAG
C*          -------------
C                     EREADTERMI                I10
C                     FILL ' '       CANAM1
C                     FILL ' '       TDTEXT
C   P5                GOTO START                      NEUE NR.
C   P8                EXCPT                    99     P-ENDE
C   P8                GOTO CPGEND                     P-ENDE
C                     GOTO STAR10
C*
C*****************************************************************
```

```
O*****************************************************************
O*****************************************************************
O*                                                               *
O*                     A U S G A B E                             *
O*                     =============                             *
O*****************************************************************
O*****************************************************************
O*       B I L D S C H I R M  -  A U S G A B E                   *
O*****************************************************************
O*
OTERMI   E E      10
O                         TENR  GC0210
O                         CANAM1  0455
O                         TDTEXTP 0670
O*
O*****************************************************************
O*         PSEUDO-AUSGABE ZUM ENTRIEGELN TASTATUR                *
O*****************************************************************
O*
OTERMI   E E      99
O                               C 2050
O*
O*---------------------------------------------------------------
O*****************************************************************
```

Beispiel 31: Variable Cursorposition, z.B. bei Fehlermeldungen

```
 1      FBILD

 2      C                    EREADBILD
 3      C                    MOVE '0420'    CPGCUR
 4      C                    EREADBILD
 5      C                    MOVEL'07'      CPGCUR
 6      C                    EREADBILD
 7      C                    MOVE '50'      CPGCUR
 8      C                    EREADBILD

 9      OBILD    E E
10      O                                V 550 'TEST CURSOR'
```

Erläuterungen:

1. Das Beispiel zeigt die Benutzung der variablen Cursor-Positionierung.

2. Es wird eine Bildschirmausgabe durchgeführt, ohne dass das Feld CPGCUR gefüllt wurde. Die Position des Anzeigers ist Zeile 1 Position 1 bei dieser Ausgabe.

3. Mit der MOVE-Operation wird in das Feld CPGCUR eine neue Position, Zeile '4' und Spalte '20' übertragen.

4. Bei dieser Ausgabe wird der Inhalt von CPGCUR aktiv. Die Position des Cursors ist Zeile 4 und Spalte 20.

5. Die Zeile wird nun geändert und die Spalte wird beibehalten.

6. Der Inhalt von CPGCUR wird für die Cursorposition aktiv. Die Position ist Zeile 7 und Spalte 20.

7. Die Spalte wird nun geändert und die Zeile wird beibehalten.

8. Der Inhalt von CPGCUR ist für die Cursorposition maßgeblich, und zwar Zeile 7 und Spalte 50.

10. Ein 'V' in Spalte 39 unterstützt die varariable Cursorpositionierung. Der Feldinhalt von CPGCUR gibt dann die Position für den Cursor an.

Beispiel 32: Sortierte Verarbeitung einer Temporary Storage Queue

```
 1      FTSQ     UQ  F     128            STORAGE
 2      D        FG        100 13
 5      ITSQ     KF
 6      I                                      1  10 SF
 7      I                                     11 128 REST


10      C                    +    1          Z       30
11      C                    MOVE Z          FG,Z
12      C                    MOVELSF         FG,Z
13      C                    EXCPTTS

20      C                    SORTAFG                   B

30      C                    DO   100        I
31      C                    MOVE FG,I       Z
32      C           Z        READ TSQ
        :
39      C                    ENDDO

90      OTSQ     EADD            TS
91      O                        SATZ     128
92      O                        SF        10
```

Erläuterungen:

2. Es wird eine Feldgruppe vereinbart, deren Anzahl Elemente der geschätzten maximalen Anzahl der Sätze der Temporary Storage Queue entspricht. Jedes Element der Feldgruppe nimmt den Schlüsselbegriff der TS-Sätze (im Bsp. die ersten 10 Stellen) und die laufende Nummer des TS-Satzes (die letzten 3 Stellen) auf.

1. Feldgruppe pflegen - *

10. Zähler Z für die laufende Nummer des TS-Satzes inkrementieren.

11. Laufende Nummer Z und Sortierfeld der Queue SF in die Feldgruppe
12. ausgeben.

13. Satz auf TS ausgeben.

2. Feldgruppe sortieren - *

20. Die Blankfelder werden nach hinten sortiert.

3. Temporary Storage Queue sortiert wieder einlesen - - - - - - *

30. Schleife, um bis zu 100 Sätze sortiert einzulesen.

31. Elementweise die Feldgruppe in das Feld für die laufende Nummer übertragen.

32. Mit der laufenden Nummer direkt auf die Temporary Storage Queue zum Lesen zugreifen.

Beispiele zur Operation FIND:

Die Anwendung der Operation FIND setzt voraus, dass eine Tabelle angelegt wurde, die durch die Operation FIND zur Verarbeitung in den Hauptspeicher geladen wird.

In den folgenden Beispielen zur FIND-Operation wird vorausgesetzt,dass eine solche Tabelle generiert ist; sie habe folgenden Aufbau:

Spalte 1 - 6	Auftragsnummer
Spalte 7 - 12	Auftragsdatum
Spalte 13 - 19	Kundennummer
Spalte 20 - 29	Artikelnummer
Spalte 30 - 31	Vertreternummer
Spalte 32 - 34	Warengruppe

Die Tabellengröße ist also 34, die 'Schlüssellänge' 10, weil jedes Element Schlüssel sein kann und die Artikelnummer mit 10 Bytes der längste mögliche Schlüssel ist.

Die Tabelle ist nach Auftragsnummern geordnet.

Beispiel 33: Anzeige einer Tabelle nach Auswahlkriterium. Es sollen genau die Aufträge angezeigt werden, die einem bestimmten Vertreter zugeordnet sind.

```
 1      FVIEW    I   F      34 10         TABLE
 2      D        PAGE       20 78

 5      IVIEW    XX
 6      I                                    1   6 AUFTNR
 7      I                                    7  12 DATUM
 8      I                                   13  19 KDNR
 9      I                                   20  29 ARTNR
10      I                                   30  31 VNR
11      I                                   32  34 WGRP

20      C           ANFANG    TAG
21      C                     Z-ADD1         I
22      C                     MAPD EINLESEN
23      C           SEITE     TAG
24      C           VNR       FIND VIEW                    80
25      C  N80                GOTO ANZEIG
26      C                     EDIT           PAGE,I
27      C                     +    1         I
28      C           I         COMP 20        SEITE        <<==
29      C           ANZEIG    TAG
30      C                     MAPD ANZEIGE
31      C                     GOTO ANFANG

90      OPAGE    F
91      O                         AUFTNR     6
92      O                         DATUM Y   16
93      O                         KDNR      30
94      O                         ARTNR     76
```

Erläuterungen:

5 Beschreibung der Datenview

22 Einlesen einer Vertreternummer (VNR)
24 Die Tabelle wird nach der eingegebenen Vertreternummer durchsucht. Wird die Nummer gefunden, wird Schalter 80 auf ein gesetzt und die zugehörigen Tabellenelemente werden eingelesen. Beim nächsten Durchlaufen der Schleife wird beginnend mit dem folgenden Tabellenelement der Rest der Tabelle durchsucht. Mit FIND kann die Tabelle vollständig sequentiell durchlaufen werden.

26 Die Elemente der Page werden nur mit Feldern aufbereitet, die in der Tabelle gefunden wurden.

Beispiel 34: Aufträge anzeigen, dabei zusätzliche Informationen wie Klartexte von Kunden und Artikeln aus Dateien anziehen.

```
 1      FVIEW    I   F      34 10         TABLE
 2      FKUNDEN
 3      FARTSTA
 4      D        PAGE       20 78

 5      IVIEW    XX
 6      I                                      1   6 AUFTNR
 7      I                                      7  12 DATUM
 8      I                                     13  19 KDNR
 9      I                                     20  29 ARTNR
10      I                                     30  31 VNR
11      I                                     32  34 WGRP

12      IKUNDEN  KF
13      I                                     37  61 KUNDE

14      IARTSTA  KF
15      I                                     21  45 ARTIKL

        :
30      C           AUFTNR    FIND VIEW                    80
31      C  N80                EXSR FEHLER
32      C  N80                GOTO ERROR
33      C           KDNR      CHAINKUNDEN              50
34      C   50                EXSR FEHLER
35      C           ARTNR     CHAINARTSTA              51
36      C   51                EXSR FEHLER
        :
50      C                     EDIT           PAGE,I
        :

90      OPAGE    F
91      O                         AUFTNR     6
92      O                         DATUM Y   16
93      O                         ARTIKL    44
94      O                         KUNDE     72
```

Erläuterungen:

5 Beschreibung der Datenview

30 Die Tabelle ist nach Auftragsnummern geordnet. Mit dem Befehl FIND kann in der Tabelle direkt auf einen Auftrag zugegriffen werden. Liegt der FIND-Befehl in einer Schleife, so werden (von einer bestimmten Auftragsnummer an) alle Tabellenelemente eingelesen.

33 Mit der in der Tabelle gefundenen Kundennummer wird auf die Kundendatei zugegriffen, um den Namen des Kunden als Klartext einzulesen. Voraussetzung ist allerdings, dass das Tabellenelement KDNR dem Schlüsselfeld der Datei KUNDEN entspricht. Dies ist bei der Generierung der Datenview zu berücksichtigen.

35 Mit der in der Tabelle gefundenen Artikelnummer wird auf die Artikelstammdatei zugegriffen, um die Artikelbezeichnung als Klartext einzulesen. Voraussetzung ist wiederum, dass das Tabellenelement

ARTNR dem Schlüsselfeld der Datei ARTSTA entspricht.

90 Die Page wird sowohl mit Werten aus der Tabelle als auch mit zusätzlichen Informationen, die durch Dateizugriffe bereitgestellt wurden, aufbereitet.

Beispiel 35: RNDOM bei der Tabellenverarbeitung

```
        :
20      C           AUFTNR    FIND VIEW                    80
21      C   80                EXSR UP01
22      C                     RNDOMVIEW
23      C           ARTNR     FIND VIEW                    80
        :
```

20 Die Tabelle wird nach einer Auftragsnummer durchsucht. Wird diese nicht gefunden, so wird der Schalter EF gesetzt. Das folgende FIND beginnt wieder beim ersten Element der View mit der Suche. Wird die Auftragsnummer aber in der Tabelle gefunden, so wird beim folgenden FIND an dieser Stelle der Tabelle wieder aufgesetzt.

22 In diesem Fall ist ein RANDOM VIEW erforderlich, um den internen Zeiger auf das erste Element der View zu setzen.

Beispiel 36: Variabler Mapname

Zur Flexibilisierung der MAP-Befehle steht der variable Mapname zur Verfügung. In diesem Zusammenhang kann nicht nur der Name der Map variabel gehandhabt werden, sondern auch das Löschen des Bildschirms vor der Ausgabe, der Write Control Character und die Minimierung der Datenübertragung bei entfernt betriebenen Bildschirmen.

Die Syntax und einige Verarbeitungsbeispiele mit variablem Mapnamen sind im folgenden beschrieben:

Die Schnittstelle zum QSF bildet ein 16-stelliges Alphafeld; in den ersten acht Stellen enthält es den Mapnamen, in der neunten die Information über das Löschen des Bildschirms; die zehnte Stelle enthält den WCC und in der elften kann die Übertragung der Konstanten unterdrückt werden; die restlichen fünf Stellen sind derzeit noch Reservebytes. Ein solches Feld kann zum Beispiel in den D-Karten als Überlagerung beschrieben werden:

```
D          MAPCTL     0 16
D          MAP           8
D          ERASE         1
D          WCC           1
D          OCTL          1
D          REST          5
D          DRUC          4
```

Beispiel:

Es soll die Map BUCH01 ausgegeben und eingelesen werden. Vor der Ausgabe soll der Bildschirm gelöscht werden, bei der Ausgabe soll die Hupe ertönen. Anschließend soll wahlweise (gesteuert durch die Taste PF11) diese Map aud dem Drucker DR01 ausgegeben werden:

```
C                    MOVEL'BUCH01  'MAP
C                    MOVE 'Y'      ERASE  * Auf jeden Fall Erase
C                    MOVE 'H'      WCC    * Ausgabe mit Hupe
C                    MOVE 'DR01'   DRUC   * Setzen Druckername
C                    MAPD          MAPCTL
C   PB      DRUC     MAPP          MAPCTL
```

Der variable Mapname ist natürlich für alle MAP-Operationen unterstützt. Beispiele:

```
C                    MAPO          MAPCTL  * Ausgabe Map BUCH01
C                    MAP           MAPCTL  * Taskorientiert
C                    MAPI          MAPCTL  * Einlesen der Map
C           'DR01'   MAPP          MAPCTL  * Drucken Map BUCH01
```

Bei der Verwendung des variablen Mapnamens ist der Programmierer dafür verantwortlich, dass das 16-stellige Feld mit sinnvollen Werten gefüllt ist. Ein Mapname muss immer angegeben werden; die restlichen Informationsbytes sind optional zu füllen; bleiben sie leer, so wird die im QSF angegebene Information übernommen, ansonsten wird sie überschrieben.

Beispiel 37: TWA-SAVE und TWA-LOAD

Die beiden Operationen TWASV und TWALD erleichtern das Zwischenspeichern von Daten. Eine typische Anwendung: Bei transaktionsorientiert geschriebenen Programmen können mit Hilfe dieser Operationen auch solche Daten einfach an eine folgende Transaktion übergeben werden, die nicht auf dem Bildschirm zwischengespeichert werden.

```
    :
    C                TWALDBSP1
    C                MAP  BEISPIEL
    :
    :   *  Verarbeitung der eingelesenen Daten. Nicht alle Daten,
    :   *  die  für die  weitere  Verarbeitung  benötigt werden,
    :   *  sind in der Map BEISPIEL enthalten.
    :
    C                MAPO BEISPIEL
    C                TWASVBSP1
    C                EXITT'BSP1'
```

Es handelt sich im Beispiel um eine Transaktion, die sich immer wieder selbst aufruft. Zu Beginn des Programms wird die gesamte TWA der vorherigen Transaktion geladen. Somit sind dem Programm auch die Daten bekannt, die nicht über den Bildschirm übergeben wurden. Voraussetzung ist dabei, dass die TWA vor dem Verlassen des Programms auch mit TWASV unter dem gleichen Namen (hier: BSP1) gerettet wurde.

Beispiel 38a) Lagerprogramm nur in CPG erstellt:

```
FBILD    C   F  24  80            DISPLAY
FFNAME   U   V     100 10         KSDS

E                    PAGE       20 55

IBILD    KF
I                                    715 724 ANR
I                                    915 9210MENGE
I                                   11151134 ARTBEZ
I                                   131513230SUMME
IFNAME   KF
I                                      1  10 ANR
I                                     11  30 ARTBEZ
I                                   P  31  350SUMME
I                                     36  36 KZ

C           A100      TAG
C                     SETON                    01
C                     EXCPT
C                     SETOF                    01
C                     READ BILD
C   P1                GOTO A300
C   P2                GOTO A300
C           A200      TAG
C           ANR       READ FNAME     PAGE
C   EF                MOVEL'     '   ANR
C   EF                MOVE '     '   ANR
C  NEF      ANR       READ FNAME
C                     RNDOMFNAME
C                     SETON                    03
C                     EXCPT
C                     SETOF                    03
C                     READ BILD
C   PA                GOTO A100
C                     GOTO A200
C           A300      TAG
C   P2                SETON                    52
C           ANR       CHAINFNAME               51
C                     SETON                    02
C                     EXCPT
C                     SETOF                    02
C                     READ BILD
C           SUMME     ADD  MENGE     SUMME
C  N52                SETON                    11    UPDATE
C   52                SETON                    12    DELETE
C                     EXCPT
C                     SETOF                    521112
C                     GOTO A100
```

```
OBILD    E E      01
O         L                        115 'LAGERVERWALTUNG'
O                         CPGTID   159
O                         UDATE    169
O                         UTIME    179
O                                  279 '-------'
O                                  272 '------------------------'
O                                  248 '------------------------'
O                                  224 '------------------------'
O                                  710 'ARTIKEL NR'
O         A               ANR    C 724
O                                 2379 '-------'
O                                 2372 '------------------------'
O                                 2348 '------------------------'
O                                 2324 '------------------------'
O                                 2418 'DATEN EINGEBEN UND'
O                                 2437 'FUNKTION AUSWAHLEN'
O                                 2452 'DE=ANZEIGE'
O                                 2465 'PF1=ANEDERN'
O                                 2479 'PF2=LÖSCHEN'
O        E        02
O         L               ANR      724
O         L      N51               733 'ÄNDERN'
O         L       51               732 'ZUGANG'
O         L       52               734 'LÖSCHEN'
O                                  909 'MENGE +/-'
O         N                      C 921 '       '
O                                 1111 'BEZEICHNUNG'
O         A               ARTBEZ  1134
O                                 1307 'BESTAND'
O         P               SUMME J 1334
O        E        03
O         P               PAGE   B 355
OFNAME   EADD     51 11
O                         ANR       10
O                         ARTBEZ    30
O                         SUMME     35P
O                                   36 'Z'
O        E       N51 11
O                         ARTBEZ    30
O                         SUMME     35P
O                                   36 ' '
O        EDEL    N51 12
OPAGE    F
O                         ANR       10
O                         ARTBEZ    35
O                         SUMME 1   50
O                         KZ        55
```

Beispiel 38b) Lagerprogramm in Verbindung mit CPG3 erstellt:

```
FFNAME

D        PAGE       20 55

IFNAME   DD

C                     FILL '-'       STRICH 79
C           A100      TAG
C                     SETOF                     52
C                     MAPD MASKE1
C   P1                GOTO A300
C   P2                GOTO A300
C           A200      TAG
C           ANR       READ FNAME     PAGE
C   EF                FILL ' '       ANR
C  NEF      ANR       READ FNAME
C                     RNDOMFNAME
C                     MAPD MASKE3
C   PA                GOTO A100
C                     GOTO A200

C           A300      TAG
C   P2                SETON                     52
C           ANR       CHAINFNAME                51
C                     MAPD MASKE2
C   52                DELETFNAME
C   52                GOTO A100
C           SUMME     ADD  MENGE     SUMME
C                     EXCPTDATEI
C                     GOTO A100

OFNAME   EADDDD   51      DATEI
O                                    36 'Z'
O        E   DD  N51      DATEI
O                                    36 ' '
OPAGE    F
O                         ANR        10
O                         ARTBEZ     35
O                         SUMME 1    50
O                         KZ         55
```

Beispiel 39: Lesen von Dateien mit Satzlänge größer 8 K

Dateien mit Satzlängen größer 8 K können über variable Eingabepositionen in Verbindung mit dem CPG-internen Feld CPGFIS eingelesen werden.

Die variable Verarbeitung der Eingabepositionen wird in der Satzbestimmung der Input Division durch den Parameter VAR erreicht, der Verschiebefaktor wird in der Procedure Division im internen Feld CPGFIS (5-stellig numerisch) gepflegt.

```
H         D              O          C      E

FBIGFILE I   F    9999 10KI       DISK
FBIGFILE2I   F    9999 10KI       DISK

D        I             5 0

IBIGFILE AA
I                                     15  24 F0
I                                   10011010 F1000
IBIGFILE AA                         V
I                                       1  35 F10000
IBIGFILE2KF        3NC*
I                                       1   2 SA
IBIGFILE2KF        3NC*             V
I                                    19992000 LAST

C            LESEN1    TAG
C                      Z-ADD10000    CPGFIS
C            ' '       READ BIGFILE
C   EF                 GOTO ENDE
C            LESEN2    TAG
C                      Z-ADD8000     CPGFIS
C            ' '       READ BIGFILE2
C   EF                 GOTO ENDE
C            ENDE      TAG
```

Angenommen wird hier, dass die Satzlänge der Datei BIGFILE 12000 Byte groß ist, die der Datei BIGFIL2 10000 Byte.

Im Programmabschnitt hinter dem Label LESEN1 wird BIGFILE eingelesen. Die Felder F0 und F1000 werden auf herkömmliche Art gelesen. Zum Lesen der Information von Stelle 10001 bis 100035 wird der Verschiebefaktor benötigt; im Programm wird er durch CPGFIS = 10000 gesetzt. Die Verschiebung der Eingabepositionen ist nur möglich, weil an die Satzbestimmung der Input Division das Schlüsselwort VAR angehängt wurde.

Analog wird hinter dem Label LESEN2 die Datei BIGFIL2 gelesen. Der Unterschied ist, dass in diesem Fall eine Prüfung der dritten Stelle im Satz vor dem Einlesen vorgenommen wird. Zu beachten ist bei dieser Art der Verarbeitung:

- der Verschiebefaktor in CPGFIS bezieht sich nur auf die Eingabepositionen, also nicht etwa auf die zu prüfende Stelle im Satz;

- es darf für die Zeichenprüfung keine Bezugszahl angegeben werden, da die zweite Satzbestimmung mit gleichem Namen bei gesetzter Bezugszahl nicht mehr verarbeitet würde.

Beispiel 40: Programm zur Dateipflege, dialogorientiert

```
 1  H         S              -                 E DATEIPFLEGE IM DIALOG

 2  FCPGWRK  U   F     100 14KV       DISK

 3  D        PAGE       20 78

 4  D        I             3 0

 5  ICPGWRK  AA    DD
 6  I                                          1 100 SATZ
 7  I                                          1  14 KEY
 8  I                                         15  64 SATZ1
 9  I                                         51 100 SATZ2
10  C           A100      TAG
11  C                     MAPD MASKE1
12  C   P1                GOTO A300
13  C           A200      TAG
14  C                     DO   20        I
15  C           KEY       READ CPGWRK
16  C   EF                GOTO A250
17  C                     EDIT           PAGE,I
18  C                     END
19  C           KEY       READ CPGWRK
20  C           A250      TAG
21  C                     RNDOMCPGWRK
22  C                     MAPD MASKE2
23  C   P2                IF
24  C                     GOTO A100
25  C                     ELSE
26  C                     GOTO A200
27  C                     END
28  C           A300      TAG
29  C           KEY       CHAINCPGWRK               50
30  C                     MAPD MASKE3
31  C                     EXCPT
32  C                     GOTO A100

33  OCPGWRK  E   DD   DEN50
34  O                         SATZ     100
35  O                         KEY       14
36  O                         SATZ1     64
37  O                         SATZ2    100
38  OCPGWRK  EADDDD   DE 50
39  O                         SATZ     100
40  O                         KEY       14
41  O                         SATZ1     64
42  O                         SATZ2    100
43  OCPGWRK  EDEL     P1N50
44  OPAGE    F
45  O                         KEY       14
46  O                         SATZ1     78
```

Das Programm startet mit einem Bildschirmdialog mit Maske 1. Es kann der Schlüssel einer Datei eingegeben werden. Bei der Funktionstaste PF1 wird zum Programmabschnitt 'Datei pflegen' verzweigt, der beim Label A300 beginnt; ansonsten wird die Datei sequentiell gelesen und je 20 Sätze werden seitenweise angezeigt.

Beachte (Statement 15 und 17):

Vor dem Mapdialog mit Maske 2 (Anzeige einer Seite mit 20 Datensaetzen) sollte der Zugriff auf die Datei, genauer die VSAM-Strings, freigegeben werden. Deshalb wird ein weiterer Satz gelesen und ein RANDOM für die Datei CPGWRK gegeben.

Im Detail und im Zusammenhang werden solche Programmiertechniken in unseren Seminaren erläutert.

Beispiel 40a: Programm zur Dateipflege, transaktionsorientiert

Die beiden folgenden Programme sind für den Benutzer identisch mit dem Programm in Teil a. Der Vorteil liegt darin, dass in der Zeit, in der der Benutzer einen Bildschirm bearbeitet, keine Task aktiv ist. Nach einem MAPO wird jeweils die Task verlassen. Erst beim erneuten Betätigen einer Programmfunktionstaste wird die Folgetask gestartet.

```
 1  H         S              -          C      E  DATEIPFLEGE TASKORIE
    H*            * DER DATENAUSTAUSCH ERFOLGT HIER ÜBER DIE CPGTCT
    H*            * ( TERMINAL CONTROL TABLE USER AREA )

 2  FCPGWRK  U   F     100 14KV       DISK

 3  D        PAGE       20 78
 4  D        I              3 0

 5  ICPGWRK  AA    DD
 6  I                                          1  100 SATZ
 7  I                                          1   14 KEY
 8  I                                         15   64 SATZ1
 9  I                                         51  100 SATZ2
10  ICPGTCT  F
11  I                                          1    1 KZ
12  I                                          2   15 KEY

13  C   CL                IF
14  C                     MOVE ' '       KZ
15  C                     FILL ' '       KEY
16  C                     EDIT           CPGTCT
17  C                     MAPO ENDE
18  C                     GOTO CPGEND
19  C                     END
20  C                     SELCT          CPGTCT
21  C   P2                MOVE ' '       KZ
22  C           KZ        IFEQ ' '
23  C                     MAPO MASKE1
24  C                     MOVE '1'       KZ
25  C                     GOTO ENDE
26  C                     END
27  C   DE                IF
28  C           KZ        IFEQ '1'
29  C                     MAP  MASKE1
30  C                     DO   20        I
31  C           KEY       READ CPGWRK
32  C   EF                MOVE ' '       KZ
33  C   EF                GOTO A250
34  C                     EDIT           PAGE,I
35  C                     END
36  C           KEY       READ CPGWRK
37  C           A250      TAG
38  C                     MAPO MASKE2
38  C                     GOTO ENDE
39  C                     END
40  C                     END
41  C           KZ        IFEQ '1'
42  C                     MAP  MASKE1
```

```
43  C           KEY       CHAINCPGWRK               50
44  C                     MAPO MASKE3
45  C                     MOVE '2'       KZ
46  C                     GOTO ENDE
47  C                     END
48  C           KZ        IFEQ '2'
49  C           KEY       CHAINCPGWRK               50
50  C                     MAP  MASKE3
51  C                     EXCPT
52  C                     MAPO MASKE1
53  C                     MOVE '1'       KZ
54  C                     END
55  C           ENDE      TAG
56  C                     EDIT           CPGTCT
57  C                     EXITT'TEST'

58  OCPGWRK  E   DD   DEN50
59  O                         SATZ     100
60  O                         KEY       14
61  O                         SATZ1     64
62  O                         SATZ2    100
63  OCPGWRK  EADDDD   DE 50
64  O                         SATZ     100
65  O                         KEY       14
66  O                         SATZ1     64
67  O                         SATZ2    100
68  OCPGWRK  EDEL     P1N50
69  OPAGE    F
70  O                         KEY       14
71  O                         SATZ1     78
72  OCPGTCT  F
73  O                         KZ         1
74  O                         KEY       15
```

Da der Bereich CPGTCT gewissen Einschränkungen unterliegt, empfiehlt es sich das Programm nach Beispiel 40b aufzubauen.

Beispiel 40b: Programm zur Dateipflege, transaktionsorientiert

```
 1  H         S              -          C      E  DATEIPFLEGE TASKORIE
    H*            * DER DATENAUSTAUSCH ERFOLGT  MIT DEN OPERATIONEN
    H*            * TWA-SAVE UND TWA-LOAD

 2  FCPGWRK  U   F     100 14KV       DISK

 3  D        PAGE       20 78
 4  D        I             3 0
 5  D        KZ            1

 6  ICPGWRK  AA    DD
 7  I                                      1  100 SATZ
 8  I                                      1   14 KEY
 9  I                                     15   64 SATZ1
10  I                                     51  100 SATZ2

11  C   CL               IF
12  C                    TWALDXXXX
13  C                    MAPO ENDE
14  C                    GOTO CPGEND
15  C                    END
16  C                    TWALDXXXX
17  C   P2               MOVE ' '       KZ
18  C           KZ       IFEQ ' '
19  C                    MAPO MASKE1
20  C                    MOVE '1'       KZ
21  C                    GOTO ENDE
22  C                    END
23  C   DE               IF
24  C           KZ       IFEQ '1'
25  C                    MAP  MASKE1
26  C                    DO   20        I
27  C           KEY      READ CPGWRK
28  C   EF               MOVE ' '       KZ
29  C   EF               GOTO A250
30  C                    EDIT           PAGE,I
31  C                    END
32  C           KEY      READ CPGWRK
33  C           A250     TAG
34  C                    MAPO MASKE2
35  C                    GOTO ENDE
36  C                    END
37  C                    END
38  C           KZ       IFEQ '1'
39  C                    MAP  MASKE1
40  C           KEY      CHAINCPGWRK              50
41  C                    MAPO MASKE3
42  C                    MOVE '2'       KZ
43  C                    GOTO ENDE
44  C                    END
45  C           KZ       IFEQ '2'
46  C           KEY      CHAINCPGWRK              50
47  C                    MAP  MASKE3
48  C                    EXCPT
49  C                    MAPO MASKE1
50  C                    MOVE '1'       KZ
51  C                    END
```

```
52  C           ENDE      TAG
53  C                     TWASVXXXX
54  C                     EXITT'TEST'

55  OCPGWRK  E   DD   DEN50
56  O                         SATZ     100
57  O                         KEY       14
58  O                         SATZ1     64
59  O                         SATZ2    100
60  OCPGWRK  EADDDD   DE 50
61  O                         SATZ     100
62  O                         KEY       14
63  O                         SATZ1     64
64  O                         SATZ2    100
65  OCPGWRK  EDEL     P1N50
66  OPAGE    F
67  O                         KEY       14
68  O                         SATZ1     78
```

Beispiel 41: Konvertieren von Feldinhalten mit CONVT

Im folgenden ist ein Programmauszug dargestellt, der die unterschiedlichen Möglichkeiten der Operation CONVT beispielhaft erläutert:

```
                          Statement          Feldinhalt F1     F2
                          -------------      ---------- ----   --------

 1  C                     MOVE 'Test'    F1             Test

 2  C                     CONVT          F1         S   test

 3  C                     CONVT          F1             TEST

 4  C                     CONVTF1        F2         X   TEST   E3C5E2E3

 5  C                     FILL '*'       F1             ****   E3C5E2E3

 6  C                     CONVTF2        F1         C   TEST   E3C5E2E3

 7  C                     MOVE 'F0F0'    F1             F0F0   E3C5E2E3

 8  C                     CONVTF1        F2         C   F0F0   00C5E2E3
```

Zu 2: Grundsätzlich gilt: Zeichen, die nicht im Sinne der Operation konvertiert werden können, bleiben unverändert.

Im Beispiel: Es sollen alle Bytes des Feldes in Kleinbuchstaben geändert werden. Der einzige Großbuchstabe ist 'T', alle anderen Zeichen bleiben unverändert; wären z.B. Ziffern im Feld enthalten, dann blieben auch sie unverändert.

Bei der Konvertierung vom Hex- ins Characterformat muss der Programmierer aber sicherstellen, dass im Ursprungsfeld sinnvolle Werte, also Werte zwischen '00' und 'FF' stehen.

Zu 8: Grundsätzlich gilt: Das Ergebnisfeld wird vor der Operation nicht gelöscht.

Das Ergebnis der Konvertierung wird linksbündig im Ergebnisfeld abgestellt.

Beispiel 42: Sequentielle Verarbeitung von Satzarten (Segmente) 8165

Sowohl VSAM-Dateien als auch HL1-Strukturen können aus mehreren Satzarten aufgebaut sein.

Im folgenden Beispiel wird gezeigt, wie eine solche Datei bei der Eingabe verarbeitet werden kann.

```
OPTIONS PHASE TST000  TITEL Segment-Verarbeitung.
*
FILE AUFTRAG
*
INPUT DIVISION
   FILE AUFTRAG
       1  2  SA
       SEGMENT  KOPF  DD  TYPE 01  REFERENCE  AUFTRAG
       SEGMENT POSTEN DD  TYPE 02  REFERENCE  AUFTRAG
       SEGMENT  TEXT  DD  TYPE 03  REFERENCE  AUFTRAG
*
PROCEDURE DIVISION
   DO WHILE CPGFRC = '  '
      READ AUFTRAG
      IF CPGFRC >< 'EF'
        EVALUATE
          WHEN SA = '01'
            READI AUFTRAG  SEGMENT KOPF
          WHEN SA = '02'
            READI AUFTRAG  SEGMENT POSTEN
          WHEN SA = '03'
            READI AUFTRAG  SEGMENT TEXT
        END-EVALUATE
         :
         :  Verarbeitung
         :
      ENDIF
   ENDDO
```

Beachte:

- Die Segmente müssen in der Input Division unmittelbar im Anschluss an die zugehörige Datei beschrieben werden.

Beschreibung:

Beim sequentiellen Lesen wird zunächst immer nur die Satzart eingelesen. In Abhängigkeit von der gelesenen Satzart wird danach der gleiche Datensatz nochmals eingelesen.

Im Data Dictionary ist die Datei AUFTRAG mit den Satzarten 01, 02 und 03 beschrieben. Um diese unterschiedlichen Strukturen nun im Programm gezielt ansprechen zu können, erhalten sie Segmentnamen, im vorliegenden Fall KOPF, POSTEN und TEXT.

Über die Eintragung REFerence wird die Verbindung vom Segmentnamen zur Satzart der Datei AUFTRAG im Data Dictionary hergestellt.

In der Procedure Division wird dann die gewünschte Satzart der Datei angesprochen, indem man bei der READI-Operation zusätzlich zum Dateinamen das Schlüsselwort SEGMent und einen Segmentnamen angibt.

Beispiel 43: Variable Cursorposition, z.B. bei Fehlermeldungen 8170

```
C           ERROR     IFEQ '**'
C                     MOVEL'KDNR  '  CPGMCU
C                     ENDIF
C                     MAPO MASKE
```

Erläuterungen:

Das Beispiel zeigt das Prinzip der variablen Positionierung des Cursors in einer mit QSF erstellten Map.

Zur Cursor-Positionierung steht das CPG-interne Feld CPGMCU zur Verfügung. Es ist 6 Bytes groß und alphanumerisch.

Mit der Operation MOVEL wird in das Feld CPGMCU der Name des Feldes übertragen, auf dem der Cursor bei der nächsten Ausgabe stehen soll. Der Feldname wird dabei als Textkonstante in Hochkommata übertragen.

Ist das Feld CPGMCU nicht gefüllt, so wird der Cursor dort positioniert, wo er in der Beschreibung der Map angegeben wurde.

Zu beachten ist, dass das Feld CPGMCU nach jeder Bildschirmausgabe gelöscht wird; sind also in einem Programm mehrere Bildschirmausgaben vorhanden, so muss der Programmierer sicherstellen, dass zur Zeit einer MAP-Ausgabe-Operation das Feld CPGMCU 'richtig' gefüllt ist.

Handelt es sich bei dem Feld, in das der Cursor positioniert werden soll, um eine Feldgruppe, dann kann der Index des gewünschten Elements dem Feld CPGMCI (dreistellig numerisch) zugewiesen werden

Beispiel 44: Groß-/Kleinschreibung in taskorientierter Anwendung 8180

Das folgende Beipiel soll zeigen, wie die Übersetzung in Großbuchstaben deaktiviert, der ursprüngliche UCTRN-Status gesichert und am Programmende wieder hergestellt wird. (UCTRN steht für Upper Case Translation und ist ein Parameter der Terminal Control Table des CICS).

```
 1            H
 2            FSTOR    UQ  F       1            STORAGE
 3            IKANAL   HS    DD
 4            I                                      1   1 STATUS
 5            ISTOR    AA    DD
 6            I                                      1   1 STATUS
 7            C           1         READ STOR
 8            C           CPGMPF    IFEQ 'CL'
 9            C                     EXHM PENDE     KANAL     T
10            C                     PURGESTOR
11            C                     MAPO ENDE
12            C                     ELSE
13            C           CPGFRC    IFEQ 'EF'
14            C                     EXHM PSTART    KANAL
15            C                     ENDIF
16            C                     MAP  TEST                L
              :
              :
17            C                     MAPO TEST
18            C                     UPDATSTOR
19            C                     EXITT'TT31'
20            C                     ENDIF
```

Zu 2: Zwischenspeichern des UCTRAN-Status
Zu 6: UCTRN-Status bei Programm-Start
Zu 8: Lösch-Taste
Zu 9: UCTRN zurücksetzen
Zu 10: Zwischenspeicher löschen
Zu 13: Wenn erster Programmaufruf, dann...
Zu 14: UCTRN-Status ermitteln/retten
Zu 15: Spalte 53 = 'L' damit der MAP - Befehl nicht in Großbuchstaben übersetzt.
Zu 18: UCTRN-Status zwischenspeichern

vom Programmierer bestimmt, die automatische Programmbeendung ist deshalb nicht sinnvoll.

Die Operation EXHM prüft beim Rücksprung ins rufende Programm,

Erläuterungen:

Zu 9: Spalte 53 = 'T' wie transaktionsorientiert beim Operationscode EXHM:

Wird in einem dialogorientierten Programm die Lösch-Taste betätigt, so wird automatisch zum Programmende verzweigt. In transaktionsorientierten Programmen wird der Weg zum Programmende

ob die Lösch-Taste gedrückt ist und verzweigt dann ans Ende des Programms. Diese Automatik m u s s in transaktionsorientierten Programmen durch den Eintrag ' T ' in Spalte 53 ausgeschaltet werden.

Zu 16: Nicht nur CICS kennt die Übersetzung in Großbuchstaben beim Bildschirm-Input. Auch CPG übersetzt standardmäßig in Großbuchstaben. Diese Übersetzung vom CPG wird mit einem 'L' in Spalte 53 des MAP-Befehls ausgeschaltet.

Das folgende HL1-Modul erhält über die Operation COMRG den UCTRAN-Status des CICS.

```
 1            H

 2            D        STATUS         1
 3            D        SYSINF        32
 4            D        SYSINFORG
 5            D        DUMMY1        21
 6            D        UCTALT         1
 7            D        DUMMY2         4
 8            D        UCTR           1
 9            D              ORG

10            C                    COMRG          SYSINF
11            C                    UCTRNOFF
12            C          UCTR      IFNE UCTALT
13            C          UCTR      IFNE 'T'
14            C                    MOVELUCTALT    UCTR
15            C                    ENDIF
16            C                    ENDIF
17            C                    MOVELUCTR      STATUS
```

Zu 2: Datenkanal
Zu 3: COMRG Bereich muss 32-stellig definiert sein
Zu 6: Stelle 22, UCTRN bis CPG 2.0
Zu 8: Stelle 27, erweitertes UCTRN
Zu 9: Ende der Redefinition
Zu 10: Ermittelt den UCTRAN-Status
Zu 11: Grundsätzlich: UCTRN OFF
Zu 12 bis 16:
Stelle 27 beim COMRG wird erst ab CICS 2.2 gefüllt. Deshalb sollte sicherheitshalber die Stelle 22 noch berücksichtigt werden.
Zu 17: UCTRN-Status an das Kanalfeld STATUS übergeben.

Erläuterungen:

Seit CPG-Release 2.1 steht der UCTRN-Status an zwei Stellen im COMRG-Bereich: In Stelle 22 (U oder N) und in Stelle 27 (U, N oder T). Die Eintragung T steht für UCTRN Transaction. Stelle 27 wird vom CPG gefüllt, falls ein CICS-Release 2.2 oder höher im Einsatz ist.

Das folgende HL1-Modul setzt den UCTRN-Status auf den Stand vor Aufruf des Programms zurück.

```
  1             H

  2             D        STATUS         1

  3             C           STATUS    IFEQ 'U'
  4             C                     UCTRNON
  5             C                     ELSE
  6             C           STATUS    IFEQ 'T'
  7             C                     UCTRNON                  T
  8             C                     ENDIF
  9             C                     ENDIF
```

Beispiel 45: Massen-Insert im CICS für VSAM-Dateien 8190

Massen-Insert ist eine Funktion im CICS zum schnellen Hinzufügen von Sätzen in ESDS-, RRDS- und KSDS-Dateien.

Besonderheiten:

- Für KSDS-Dateien müssen die hinzuzufügenden Sätze nach ihrem Schlüssel aufsteigend sortiert sein.
- Während des Massen-Inserts sind keine anderen Dateioperationen möglich !
- Die Verarbeitungsart Massen-Insert wird **„eingeschaltet“**, wenn in der F-Karte die Verarbeitungsart O gesetzt ist und Sätze mit EXCPT und dem Eintrag ADD in der Satzbestimmung der O-Karten ausgegeben werden.

  Die Verarbeitungsart Massen-Insert wird **„ausgeschaltet“**, wenn die Transaktion beendet wird oder wenn ein RANDOM für die Ausgabedatei ausgeführt wird.

Für die Programmierung bedeutet das, dass neben dem Massen-Insert in einem Programm oder Modul keine weitere Dateiverarbeitung möglich ist.

Wird ein Modul geschrieben, das aus anderen Programmen aufgerufen wird, dann sollte es sicherheitshalber die Massen-Insert-Verarbeitung mit einem RANDOM beenden.

```
H    A                   8               E   C
H*-------------------------------------------------------------*
H*            TEST CPGESD ADD ONLINE MASSINSERT                *
H*-------------------------------------------------------------*

FCPGESD  O   V4000 200 04R        ESDS
D        RECORD        80
D        COUNT          7 0
D        I              5 0
D        J              5 0
D        CPGVRL         5 0
D        STIME          7 0
C                     FILL 'B'       RECORD
C                     Z-ADD200       CPGVRL
C                     Z-ADDCPGTIM    STIME
C                     DO   10        J
C                     TIME
C                     MAPO TT2501
C                     FILL 'B'       RECORD
C                     DO   1000      I
C                     EXCPTSATZ
C                     ENDDO
C                     ENDDO
C           CPGTIM    SUB  STIME     STIME
C                     MAPO TT2501
C                     RNDOMCPGESD
OCPGESD  EADD             SATZ
O                         RECORD    80
O                                  200 ' '
```

Anwendungsbeispiel 1 (HL1): 8501

Zeile 1 der Bildschirmausgabe:

Die erste Bildschirmzeile soll für alle Anwendungen gleich gestaltet werden. Sie soll den Firmennamen, das Sachbearbeiter-Kurzzeichen,den Terminalnamen, sowie Datum und Uhrzeit enthalten.

Wir erstellen dazu folgendes HL1-Modul, wobei wir davon ausgehen, dass das Sachbearbeiter-Kurzzeichen zu Beginn der Arbeit durch ein eigenes Anmeldeprogramm in Position 1 bis 3 der CPGTCT abgestellt wurde.

BILDSCHIRM ZEILE 1 UMWANDLUNGSLISTE VOM 01.02.79

```
 1     01 001  H                              BILDSCHIRM ZEILE 1  XZEIL1

 2     01 002  FBILD

 3     02 001  ICPGTCT F
 4     02 002  I                                        1   3 TKZ

 5     03 001  C                    SELCT          CPGTCT
 6     03 002  C                    EXCPT

 7     04 001  OBILD    E E
 8     04 002  O                                   113 'LATTWEIN GMBH'
 9     04 003  O                        CPGTID     154
10     04 004  O                        TKZ        160
11     04 005  O                        UDATE      170
12     04 006  O                        UTIME      180
```

Einmal katalogisiert kann dieses Modul dann aus jedem Anwendungsprogramm mit dem Befehl:

```
               C                    EXHM XZEIL1
```

aufgerufen werden.

Anwendungsbeispiel 2 (HL1): 8502

Fehlerkatalog

Bestimmte Fehlermeldungen kommen bei verschiedenen Anwendungen immer wieder vor. Es erscheint daher sinnvoll, einen Fehlerkatalog als Baustein anzulegen, der einen vom Anwendungsprogramm bereitgestellten Fehlerschlüssel in Klartext übersetzt.

```
FEHLERKATALOG                          UMWANDLUNGSLISTE VOM 01.02.79

 1     01 001  H                                FEHLERKATALOG       FKAT

 2     02 001  E                     NOTE        3 0
 3     02 002  E                     FEHLER     24

 4     03 001  C                      SETIN          NOTE       11
 5     03 002  C                      EDIT           FEHLER

 6     04 001  OFEHLER  F
 7     04 002  O                11          21 'FALSCHE PROGRAMMTASTE'
 8     04 003  O                12          13 'KEINE EINGABE'
 9     04 004  O                13          15 'FALSCHE EINGABE'
10     04 005  O                14          21 'UNBERECHTIGTE ABFRAGE'
```

Dieses Programm kann von jedem HL1-Baustein aufgerufen werden. Das folgende Beispiel zeigt einen Auszug aus einem solchen Baustein.

```
KUNDENÄNDERUNG                         UMWANDLUNGSLISTE VOM 01.02.79

                ...

15     03 023  IFM     HS
16     03 024  I                                 P   1   20NOTE
17     03 025  I                                     3  26 FEHLER

                ...

38     04 037  C                     TESTT'VT'                        50
39     04 038  C   50                GOTO ABFR
40     04 039  C                     Z-ADD 4         NOTE
41     04 040  C                     EXHM FKAT       FM

                ...

87     07 018  O                  FEHLER  124

                ...
```

In diesem Beispiel wird angenommen, dass Kundenänderungen nur von einem Bildschirm im Vertrieb vorgenommen werden dürfen. Die Terminals im Vertrieb heissen VT01, VT02 und VT03. Das Kundenänderungsprogramm fragt ab, ob die beiden ersten Stellen des Terminalnamens gleich 'VT' sind. Wenn ja, wird zur normalen Abfrage verzweigt. Wird das Programm von einem anderen Terminal aufgerufen, so wird die Zahl 4 in das Feld 'NOTE' gebracht und mit dem Feld 'NOTE' und dem Feld 'FEHLER', das vorerst Blanks enthält, in den HL1-Baustein 'FKAT' verzweigt. Dort wird auf Grund des Inhalts von 'NOTE' zunächst die Bezugszahl 14 gesetzt und anschließend über eine EDIT-Operation der Text 'UNBERECHTIGTE ABFRAGE' in das Feld 'FEHLER' gesetzt. Die Felder 'NOTE' und 'FEHLER' werden darauf wieder an das aufrufende Programm zurück gegeben.

Wird der Platz für die Fehlermeldung für alle Anwendungen genormt, z.B. 1.Zeile Stelle 21 bis 44, so kann das Beispiel wie folgt abgeändert werden: Statement Nr. 17 im Programm Kundenänderung entfällt.

Das Programm 'FKAT' wird um folgende Statements erweitert:

```
 6     03 003  C                   EXCPT

11     04 006  OBILD    E
12     04 007  O                FEHLER  144
```

Beispiel 3: CPG-Hauptprogramm mit HL1-Dataset

```
 1  H               D        8                 E CKUNDENSTAMM / MAHNEN

 2  FBILD
 3  FKDVSAM  U   V     100 10          KSDS
 4  FKUNDEN  U   F     100  6          HL1

 5  IBILD    KF
 6  I                                      521 526 KDNR
 7  I                                      721 744 FIRMA
 8  I                                      921 926 PLZ
 9  I                                      928 951 ORT
10  I                                     11211138 STR
11  I                                     132113292NETTO
12  I                                     15171517 US
13  IKUNDEN  HS
14  I                                        1   4 CPGHIC
15  I                                        5  10 KDNR
16  I                                       11  34 FIRMA
17  I                                       35  40 PLZ
18  I                                       41  64 ORT
19  I                                       65  65 US
20  I                                     P  66  702STEUER
21  I                                     P  71  752NETTO
22  I                                     P  76  802BRUTTO
23  I                                     P  81  821MWST
24  I                                        83 100 STR

25  C           ANFANG    TAG
26  C                     EREADBILD               01
27  C                     CHAINKUNDEN             11
28  C                     EREADBILD               02
29  C   DE 11             NEWRCKUNDEN
30  C   DEN11             UPDATKUNDEN
31  C   P1N11             DELETKUNDEN
32  C                     GOTO ANFANG

33  OBILD    E E      01
34  O                         KDVSAM   MAP 'PRI'
35  O         N               KDNR   C 526
36  O        E E      02
37  O                         KDVSAM   MAP 'PRI'
38  O         P               KDNR     526
39  O                                  544 '** KORREKTUR **'
40  O                11               544 '** NEUANLAGE **'
41  O               N11               570 'PF 1 = L O E S C H E N !'
42  O         A               FIRMA  C 744
43  O         A               PLZ      926
44  O         A               ORT      951
45  O         A               STR     1138
46  O         N               NETTO K 1333
47  O         A               US      1517
48  O         P               STEUERK 1533
49  O         P               MWST  K 1540
50  O         P               BRUTTOK 1733
```

Erläuterungen zu:

1 In der H-Karte wird die HL1-Library (SP. 22) zugeordnet.
2 Der Bildschirm ist in der Standard-File-Table definiert.
3 Die Datei KDVSAM muss definiert werden, da sie im HL1-Baustein benutzt wird (HL1-Konvention). Dies ist nicht erforderlich, wenn der HL1-Baustein (hier das Dataset KUNDEN) mit einem Eintrag 'D' oder 'S' in Spalte 34 der H-Karte umgewandelt wird.
4 das HL1-Dataset wird genauso wie eine normale Datei definiert, die Einheit ist 'HL1'. Feste Satzlänge wird vorausgesetzt.
5 bis
12 Enthalten die Bildschirmeingabefelder.
13 Ist die Eingabedefinition für das HL1-Dataset, in den Spalten 15-16 ist 'HS' für HL1-Storage einzutragen.
14 Das HL1-Interface-Controlfeld CPGHIC (4 Stellen)muss im Datenkanal definiert werden. Durch dieses Feld wird der Operationscode an den HL1-Baustein übergeben und der Returncode zurückübertragen.
Das Feld CPGHIC wird intern durch CPG gefüllt und interpretiert.
15 Das Feld KDNR enthält den Schlüssel, für das HL1-Dataset.
16 bis
24 Hier sind die Datenfelder des HL1-Datasets beschrieben. Wird mit dem HL1-Dataset eine Ausgabe durchgeführt sollten hier alle Felder definiert sein.
26 Am Bildschirm wird der Schlüssel (KDNR) angefordert.
27 Mit CHAIN werden die Daten vom Baustein angefordert. An den Bausten wird der Operationscode 'G' übergeben.Übergibt der Baustein den Returncode'G'(nicht gefunden),so wird der Schalter 11 gesetzt. In Faktor 1 braucht kein Key angegeben zu werden,da dieser bereits in HL1-Datenkanal definiert ist (Feld KDNR).
28 Die Daten werden am Bildschirm verarbeitet.
29 Ein neuer Satz wird mit NEWRC über den Baustein hinzugefügt(Operationscode 'N',bei Returncode 'D' wird der Schalter 'DR'gesetzt).
30 Ein bestehender Satz wird mit der Operation UPDAT verändert (Operationscode 'U').
31 Mit DELET werden Sätze durch den Baustein gelöscht (Operationscode 'L').
33 bis
35 Bildschirmbeschreibung zur Eingabe der Kundennr. Die MAP 'KDVSAM' enthält das konstante Bildschirmformat.(Nur für TOP oder AF Ben)
36 bis
50 Enthält die Bildschirmanzeige der Kundendaten. Die Felder FIRMA, PLZ, ORT, STR, NETTO und US können modifiziert werden. Die Felder STEUER, MWST und BRUTTO werden errechnet und vom Dataset zur Verfügung gestellt.

Beispiel 4: HL1-DATASET zu Programmbeispiel 3

```
 1  H                                                DATASET KUNDEN/UMSATZ

 2  FKDVSAM  U   V     100 10KV       KSDS

 3  D        OC             2         HL1-DATASET OPERATION CODE
 4  D        RC             2         HL1-DATASET RETURNCODE
 5  D        KDNR           6         KUNDENNUMMER
 6  D        SATZ         0 90        DATENSATZ
 7  D        FIRMA         24         FIRMENBEZEICHNUNG
 8  D        PLZ            6         POSTLEITZAHL
 9  D        ORT           24         ORT
10  D        US             1         UMSATZSTEUERSCHLÜSSEL
11  D        STEUER         9 2       MEHRWERTSTEUERBETRAG IN D-MARK
12  D        NETTO          9 2       BETRAG OHNE MWST IN D-MARK
13  D        BRUTTO         9 2       BETRAG MIT MWST IN D-MARK
14  D        MWST           3 1       MEHRWERTSTEUER IN PROZENT
15  D        STR           18         STRASSE

16  IKDVSAM  KF
17  I                                     1  10 KEY
18  I                                    11 100 SATZ

19  C                    MOVEL'HSOA'    KEY
20  C                    MOVE KDNR      KEY
21  C           OC       IF   'G'                        CHAIN
22  C           KEY      CHAINKDVSAM    EXIT       99
23  C                    EXSR CALC
24  C                    END
25  C           OC       IF   'N'                        NEWRC
26  C                    EXSR CALC
27  C                    EXCPT                   91
28  C                    END
29  C           OC       IF   'U'                        UPDAT
30  C                    EXSR CALC
31  C                    EXCPT                   92
32  C                    END
33  C           OC       IF   'L'                        DELET
34  C           KEY      DELETKDVSAM
35  C                    END
36  C           EXIT     TAG
37  C   DR               MOVE 'D'       RC               DUPL RECORD
38  C   99               MOVE 'G'       RC               GET N FOUND
39  CSR         CALC     BEGSR
40  CSR                  Z-ADD14,0      MWST
41  CSR         US       IF   'A'
42  CSR                  Z-ADD7,0       MWST
43  CSR                  END
44  CSR         NETTO    *    MWST      BRUTTO
45  CSR         BRUTTO   /    100       STEUER
46  CSR         NETTO    ADD  STEUER    BRUTTO
47  CSR                  ENDSR

48  OKDVSAM  EADD     91
49  O                        SATZ     100
50  O                        KEY       10
51  O        E        92
52  O                        SATZ     100
```

Erläuterungen zu:

```
 2  Die Datei KDVSAM wird benutzt, um die Kundeninformationen zu spei-
    chern.
 3  Das Feld OC (Stellen 1-2 des HL1-Interface-Controlfeldes CPGHIC)
    wird benutzt um die Operation zu kennzeichnen.
 4  Das Feld RC (Stellen 3-4 des HL1-Interface-Controlfeldes CPGHIC)
    wird benutzt um den Return-Code zu übermitteln.
 5  Das Feld KDNR ist Bestandteil des Schlüssels.
 6  bis
15  Enthält die Beschreibung der Datenfelder.
16  bis
18  Enthält die Eingabe der physischen Datei.
19  Der Schlüssel enthält einen konstanten Praefix 'HSOA'.
20  Die KDNR wird rechtsbündig in den Schlussel übertragen.
21  bis
24  Mit Operationscode 'G' wird die CHAIN-Operation aufgerufen.
25  bis
28  Mit Operationscode 'N' wird die NEWRC Operation aufgerufen.
29  bis
32  Mit Operationscode 'U' wird die UPDAT Operation aufgerufen.
33  bis
35  Mit Operationscode 'L' wird die DELET Operation aufgerufen.
36  bis
38  Vor Rückkehr in das aufrufende Programm werden die Returncodes
    gesetzt: 'D' bei DUPLICATE RECORD und 'G' bei GET NOT FOUND.
39  bis
47  Zeigt eine Rechenroutine mit der das HL1-Dataset eine Berechnung
    durchführt (hier Mehrwertsteuer).
48  bis
52  Beschreibt die Ausgabe auf die physische Datei KDVSAM.
```

Beispiel 5: Aufruf HL1-Module in RPG-Programm

```
// OPTION CATAL
   PHASE TESTPR,*
// EXEC CPGTDTR,SIZE=AUTO
// EXEC RPGII

     H             J                   S

     FKARTE   IP  F      80            READ05 SYSIPT
     FLISTE   O   F     132            PRINTERSYSLST

     IKARTE   KF  01
     I                                       1  64 SATZ
     IKANRPG  HS    DD
     IKAN002  HS
     I                                       1  80 XXX

     C                    .
     C                    .    VERARBEITUNG
     C                    .
     C                   EXHM AL030     KANRPG
     C                    .
     C                    .    VERARBEITUNG
     C                    .
     C                   EXHM AL030     KAN002
     C                    .
     C                    .    VERARBEITUNG
     C                    .
     C                   EXHM HHT02
     C                    .
     C                    .    VERARBEITUNG
     C                    .

     OLISTE   E  201   01
     O                         VSATZ     64
/*
// EXEC PROG=$SYSIN
/*
// EXEC LNKEDT
/*
/&
* $$ EOJ
```

Beispiel B01 : Leser, Drucker und Überlaufsteuerung.

```
 1  H    A                   O               B E C20  DRUCKER OF

 2  FREADER  I   F      80            READER
 3  FPRINTER O   F     132            PRINTER

 4  LPRINTER 072FL0010106612

 5  IREADER  KF
 6  I                                      1  80 SATZ

 7  C                     EXCPT                    01
 8  C           ANFANG    TAG
 9  C                     READ READER
10  C   EF                GOTO ENDE
11  C   OF                EXCPT                    02
12  C                     EXCPT                    03
13  C                     GOTO ANFANG
14  C           ENDE      TAG

15  OPRINTER E  201   01
16  O                                    24 'ERSTE SEITE'
17  O                          UDATE     70
18  O                          UTIME     80
19  O        E  201   02
20  O                                    24 'FOLGESEITE '
21  O        E  1     03
22  O                          SATZ      80
23  O        E  1     OF
24  O                                    24 '*** OVERFLOW ***'
```

Erläuterungen:

1 Steuerkarte. Ein 'B' in Spalte 47 muss eingetragen werden.

2 Es wird eine Datei 'READER' definiert. Die Dateiart ist 'I' Eingabedatei. Die Satzlänge ist fest (F) 80 Bytes. Die Eingabeeinheit ist READER (SYSIPT).

3 Es wird eine Druckerdatei PRINTER definiert. Die Dateiart ist 'O', Ausgabedatei. Die Satzlänge ist fest (F) 132 Bytes. Die Eingabe-/Ausgabeeinheit ist PRINTER (SYSLST).

4 Die Formularlänge wird mit 72 Zeilen definiert. Die erste Zeile wird mit Kanal 01 und Zeile 66 mit Kanal 12 definiert.(Kanal 12 beschreibt die Zeile, nach der der Überlaufschalter 'OF' gesetzt wird).

5 Das Eingabefeld der Datei 'READER' wird definiert.'KF' kennzeichnet die Eingabebestimmung als Satzbestimmung.

7 Ausführung der Ausgabe mit Bezugszahl 01.

8 Ein Merkmal 'ANFANG' wird gesetzt.

9 Die Eingabedatei 'READER' wird gelesen.

10 Der 'End of File' Schalter wird abgefragt.

11 Der Überlaufschalter wird abgefragt.

12 Ausführung der Ausgabe mit Bezugszahl 03.

13 Es wird wieder zum Merkmal 'ANFANG' verzweigt.

14 Das Merkmal ENDE wird gesetzt. Da keine weiteren Rechenbestimmungen folgen, wird hier bei der Ausführung das Programm beendet.

15 Vor dem Drucken nach Kanal 01; nach dem Drucken 2 Zeilentransporte, wenn Bezugszahl 01 gesetzt ist.

16 bis
18 Ausgabe der Überschrift, Datum und Uhrzeit.

19 Vor dem Drucken nach Kanal 01, nach dem Drucken 2 Zeilentransporte.

20 Ausgabe einer Konstanten.

21 Ein Zeilentransport, wenn Bezugszahl 03 gesetzt ist.

22 Ausgabe des Satzes.

23 Ein Zeilentransport.

24 Wenn der OF-Schalter gesetzt ist, Ausgabe der Konstanten.

Beispiel B02 : Programmkatalog CPGWRK.

```
 1  H    A                   O              B E C02  PROGRAMMKAT.

 2  FPRINTER O   F     132            PRINTER
 3  FCPGWRK  I   F     100 14         KSDS

 4  LPRINTER 072FL00101

 5  ICPGWRK  KF
 6  I                                      1   2 SA
 7  I                                      3  10 PNAME
 8  I                                     11  14 TRANID
 9  I                                     15  24 PROT
10  I                                     25  290TWA
11  I                                     30  49 TEXT
12  I                                     50  52 PKZ

13  C                     FILL '-'       STRICH 80
14  C                     EXCPT                    01
15  C                     MOVEL'01 '     KEY    14
16  C           KEY       SETLLCPGWRK
17  C           ANFANG    TAG
18  C           KEY       READ CPGWRK
19  C   EF                GOTO ENDE
20  C           SA        COMP KEY       ENDE       **
21  C                     EXCPT                    02
22  C                     GOTO ANFANG
23  C           ENDE      TAG

24  OPRINTER E  201   01
25  O                                    15 'PROGRAMMKATALOG'
26  O                         UDATE      70
27  O                         UTIME      80
28  O        E  2     01
29  O                         STRICH     80
30  O        E  2     01
31  O                                     8 'PROGRAMM'
32  O                                    15 'TRID'
33  O                                    22 'TEXT'
34  O                                    46 'PKZ'
35  O                                    54 'TWA'
36  O                                    67 'PROTECTION'
37  O        E  1     02
38  O                         PNAME       8
39  O                         TRANID     15
40  O                         TEXT       38
41  O                         PKZ        46
42  O                         TWA   J    54
43  O                         PROT       67
```

Erläuterungen:

In diesem Beispiel wird mit einem Key in der Datei 'CPGWRK' positioniert und eine Liste (Programmkatalog) auf dem Drucker ausgegeben.

Bei Ende der Datei 'EF' oder einer neuen Satzart wird das Programm beendet.

Beispiel B03 : Sätze vom Kartenleser in eine KSDS-Datei laden.

```
H    A                   O               B E C03  LOAD CPGKSD

FREADER  I   F      80            READER
FPRINTER O   F     132            PRINTER
FCPGKSD  O   V    8100 20         KSDS

LPRINTER 072FL00101

IREADER  KF
I                                      1  20 KEY
I                                     21  80 DATEN

C                     FILL '-'       STRICH 80
C                     EXCPT                     01
C           ANFANG    TAG
C                     READ READER
C   EF                GOTO ENDE
C                     EXCPT                     02
C                     GOTO ANFANG
C           ENDE      TAG

OPRINTER E  201   01
O                                     18 'CPGKSD NEUE SÄTZE'
O                           UDATE     70
O                           UTIME     80
O        E  2     01
O                           STRICH    80
O        E  2     01
O                                      3 'KEY'
O                                     26 'DATEN'
O        E  1     02
O                           KEY       20
O                           DATEN     81
OCPGKSD  EADD     02
O                           DATEN     80
O                           KEY       20
```

Erläuterungen:

Im obenstehenden Beispiel werden Karten von einem Kartenleser gelesen und in die KSDS-Datei CPGKSD übertragen. Die gelesenen Sätze werden mit 'EADD' sequentiell hinzugefügt, d. h. wenn bereits Sätze in der Datei existieren, werden neue Sätze in die Datei eingefügt.

Das 'O' in Spalte 15 der F-Karte kennzeichnet die Datei 'CPGKSD' als Ausgabedatei. Die Ausgabe erfolgt sequentiell. Beim sequentiellen Hinzufügen müssen die Sätze in aufsteigender Schlüsselfolge vorliegen. Beim Laden der Datei wird gleichzeitig auf der Datei 'PRINTER' ein Protokoll gedruckt.

Beispiel B04 : Sätze von der CPGWRK in eine KSDS-Datei kopieren.

```
H    A                   O               B E C04  COPY CPGWRK

FREADER  I   F      80            READER
FPRINTER O   F     132            PRINTER
FCPGWRK  I   F     100 14         KSDS
FCPGKSD  O   V    8100 20         KSDS

LPRINTER 072FL00101

IREADER  KF
I                                        1   2 SA
I                                        1  14 KEYVL
ICPGWRK  KF
I                                        1  14 KEY
I                                       15 100 SATZ

C                    FILL '-'       STRICH 80
C           ANFANG   TAG
C                    READ READER
C   EF               GOTO ENDE
C                    EXCPT                    01
C           KEYVL    SETLLCPGWRK
C                    DO   999
C           KEYVL    READ CPGWRK
C   EF               GOTO ENDE
C           KEY      COMP SA        ANFANG    **
C                    EXCPT                    02
C                    END
C                    GOTO ANFANG
C           ENDE     TAG

OPRINTER E  201   01
O                                       15 'SÄTZE KOPIEREN'
O                                       38 'VON CPGWRK NACH CPGKSD'
O                                       50 'SATZART'
O                        SA             53
O                        UDATE          70
O                        UTIME          80
O        E  2     01
O                        STRICH         80
O        E  2     01
O                                        3 'KEY'
O                                       25 'SATZ'
O        E  1     02
O                        KEY            14
O                        SATZ          107
OCPGKSD  EADD     02
O                        KEY            14
O                        SATZ          106
```

Erläuterungen:

Im obenstehenden Beispiel werden Sätze aus der 'CPGWRK' sequentiell in eine KSDS-Datei kopiert. Die Sätze werden mit 'EADD' in in die KSDS-Datei übertragen.
Über die Datei 'READER' werden Vorlaufkarten eingelesen, die die zu kopierenden Satzarten angibt.
Auf der Datei 'PRINTER' wird ein Protokoll ausgedruckt.

Beispiel B05 : Sätze in einer KSDS-Datei direkt updaten.

```
H    A                   O              B E C06  UPDATE DIREKT

FREADER  I   F      80            READER
FPRINTER O   F     132            PRINTER
FCPGKSD  U   V    8100 20         KSDS

LPRINTER 072FL00101

IREADER  KF
I                                       1  20 KEY
I                                      21  21 UPDKZ
ICPGKSD  KF
I                                      21  39 NAME

C                    EXCPT                    01
C           ANFANG   TAG
C                    READ READER
C   EF               GOTO ENDE
C           KEY      CHAINCPGKSD              99
C                    EXCPT                    02
C                    GOTO ANFANG
C           ENDE     TAG

OPRINTER E  201   01
O                                      15 'SÄTZE UPDATEN '
O                                      25 'IN CPGKSD'
O                          UDATE       70
O                          UTIME       80
O        E  2     01
O                                       3 'KEY'
O                                      25 'NAME'
O                                      42 'X'
O        E  1     02
O                          KEY         20
O                          NAME        40
O                          UPDKZ       42
O                  99                  43 '*** NICHT GEFUNDEN ***'
OCPGKSD  E         02N99
O                          NAME        59
O                          UPDKZ      106
```

Erläuterungen:

In diesem Beispiel werden Sätze, die von einem Kartenleser eingelesen wurden, direkt in einer KSDS-Datei upgedated. Ein Protokoll wird auf dem Drucker ausgedruckt.

Beispiel B06 : Sätze in einer KSDS-Datei sequentiell updaten.

```
H    A                   O              B E C12  UPDATE VORW.

FLISTE   O   F     132            PRINTER
FCPGKSD  U   V    8100 20         KSDS

LLISTE   072FL00101

ICPGKSD  KF
I                                       1  20 KEY
I                                       1  40 SATZ

C                     EXCPT                    01
C           ANFANG    TAG
C           '11'      READ CPGKSD
C   EF                GOTO ENDE
C           KEY       CABGT'11'      ENDE
C                     EXCPT                    02
C                     GOTO ANFANG
C           ENDE      TAG

OLISTE   E  201   01
O                                    15 'SÄTZE UPDATEN '
O                                    36 'IN CPGKSD SEQUENTIELL'
O                         UDATE      70
O                         UTIME      80
O        E  1     02
O                         SATZ       40
O                                    40 '*'
OCPGKSD  E        02N99
O                                    40 '*'
```

Erläuterungen:

In diesem Beispiel werden Sätze sequentiell in einer KSDS - Datei upgedated und gleichzeitig als Liste auf dem Drucker ausgegeben. Es werden nur die Sätze upgedated, die den Schlüssel '11' haben.

Beispiel B07 : Sätze in einer KSDS-Datei sequentiell löschen.

```
H    A                   O              B E C13  SEQ DELETE

FREADER  I   F      80            READER
FPRINTER O   F     132            PRINTER
FCPGKSD  U   V    8100 20         KSDS

LPRINTER 072FL00101

IREADER  KF
I                                      1   2 SA
ICPGKSD  KF
I                                      1  20 KEY
I                                     21 100 SATZ

C           ANFANG    TAG
C                     READ READER
C   EF                GOTO ENDE
C                     EXCPT                    01
C           LESEN     TAG
C           SA        READ CPGKSD
C   EF                GOTO RNDOM
C           KEY       CABGTSA        RNDOM
C                     EXCPT                    02
C                     GOTO LESEN
C                     GOTO ANFANG
C           RNDOM     TAG
C                     RNDOMCPGKSD
C                     GOTO ANFANG
C           ENDE      TAG

OPRINTER E  201   01
O                                     24 'DELETE RECORDS IN CPGKSD'
O                                     34 'SATZART'
O                         SA          37
O                         UDATE       70
O                         UTIME       80
O        E  2     01
O                                      3 'KEY'
O                                     25 'SATZ'
O        E  1     02
O                         KEY         20
O                         SATZ       101
OCPGKSD  EDEL     02
```

Erläuterungen:

In diesem Beispiel werden Sätze sequentiell in einer KSDS-Datei gelöscht und ein Protokoll wird auf dem Drucker ausgegeben.

Beispiel B08 : Sätze in einer KSDS-Datei direkt löschen.

```
H    A                   O              B E C14  DELETE DIRECT

FREADER  I   F      80            READER
FPRINTER O   F     132            PRINTER
FCPGKSD  U   V    8100 20         KSDS

LPRINTER 072FL00101

IREADER  KF
I                                      1  20 KEY
ICPGKSD  KF
I                                     21  39 NAME

C                    EXCPT                    01
C           ANFANG   TAG
C                    READ READER
C   EF               GOTO ENDE
C           KEY      CHAINCPGKSD              99
C                    EXCPT                    02
C                    GOTO ANFANG
C           ENDE     TAG

OPRINTER E  201   01
O                                     15 'SÄTZE LÖSCHEN'
O                                     33 'IN CPGKSD DIREKT'
O                         UDATE       70
O                         UTIME       80
O        E  2     01
O                                      3 'KEY'
O                                     25 'NAME'
O        E  1     02
O                         KEY         20
O                         NAME        40
O                  99                 43 '*** NICHT GEFUNDEN ***'
OCPGKSD  EDEL     02N99
```

Erläuterungen:

Es werden Sätze in einer KSDS-Datei direkt gelöscht. Ein Protokoll wird auf dem Drucker ausgegeben.

Beispiel B09 : Programmkatalog rückwärts auflisten.

```
H    A                   O              B E C15  PKAT RÜCKW.

FPRINTER O   F     132           PRINTER
FCPGWRK  I   F     100 14        KSDS

LPRINTER 072FL00101

ICPGWRK  KF
I                                       1   2 SA
I                                       3  10 PNAME
I                                      11  14 TRANID
I                                      15  24 PROT
I                                      25  290TWA
I                                      30  49 TEXT
I                                      50  52 PKZ

C                    FILL '-'       STRICH 80
C                    EXCPT                     01
C           '02'     READBCPGWRK
C           ANFANG   TAG
C           '01'     READBCPGWRK
C   EF               GOTO ENDE
C           SA       COMP '01'      ENDE       ****
C                    EXCPT                     02
C                    GOTO ANFANG
C           ENDE     TAG

OPRINTER E  201   01
O                                      15 'PROGRAMMKATALOG'
O                                      28 'RÜCKWÄRTS'
O                         UDATE        70
O                         UTIME        80
O        E  2     01
O                         STRICH       80
O        E  2     01
O                                       8 'PROGRAMM'
O                                      15 'TRID'
O                                      22 'TEXT'
O                                      46 'PKZ'
O                                      54 'TWA'
O                                      67 'PROTECTION'
O        E  1     02
O                         PNAME         8
O                         TRANID       15
O                         TEXT         38
O                         PKZ          46
O                         TWA    J     54
O                         PROT         67
```

Erläuterungen:

Es wird in einem Programmkatalog rückwärts gelesen (READB) und ein Protokoll auf dem Drucker ausgegeben.

Beispiel B10 : Sätze vom Kartenleser in eine ESDS-Datei kopieren.

```
H    A                   O               B E C23  COPY > CPGESD

FREADER  I   F      80            READER
FCPGESD  O   V    8100  4R        ESDS

IREADER  KF
I                                        1  80 SATZ

C           ANFANG    TAG
C                     READ READER
C   EF                GOTO ENDE
C                     EXCPT                    01
C                     GOTO ANFANG
C           ENDE      TAG

OCPGESD  EADD     01
O                       SATZ      80
```

Erläuterungen:

In diesem Beispiel werden Sätze von einem Kartenleser in eine ESDS-Datei kopiert. Die Sätze werden mit 'EADD' in die ESDS-Datei hinzugefügt.

Bestehen bereits Sätze in der ESDS-Datei, so werden die neuen Sätze an das Ende der Datei angefügt.

BeispieL B11 : Drucken einer ESDS-Datei.

```
H    A                   O               B E C24

FCPGESD  I   V    8100  4R        ESDS
FPRINTER O   F     132            PRINTER

LPRINTER 072FL0010106612

ICPGESD  KF
I                                       1  80 SATZ

C                     EXCPT                     01
C                     FILL X'00'     KEY     4
C                     DO   999
C           KEY       READ CPGESD
C   EF                GOTO ENDE
C                     ADD  1         COUNT   50
C                     EXCPT                     02
C                     END
C           ENDE      TAG
C                     EXCPT                     03

OPRINTER E  201   01
O                                     17 'SÄTZE VON CPGESD'
O                          UDATE      70
O                          UTIME      80
O        E  1     02
O                          SATZ       80
O        E 21     03
O                                     17 '*** ANZAHL SÄTZE'
O                          COUNT Z    23
```

Erläuterungen:

Im obigen Beispiel wird eine ESDS-Datei gelesen und auf dem Drukker ausgegeben.

Die Verarbeitung beginnt mit dem 1. Satz der Datei.
(FILL X'00' nach KEY).

Beispiel B12 : Sequentielles Update einer ESDS-Datei.

```
H    A                   O               B E C31

FCPGESD  U   V    8100  4R        ESDS
FPRINTER O   F     132            PRINTER

LPRINTER 072FL0010106612

ICPGESD  KF
I                                       1  80 SATZ

C                     EXCPT                     01
C                     FILL X'00'     KEY     4
C                     DO   999
C           KEY       READ CPGESD
C   EF                GOTO ENDE
C                     ADD  1         COUNT   50
C                     EXCPT                     02
C                     END
C           ENDE      TAG
C                     EXCPT                     03

OPRINTER E  201   01
O                                     17 'SÄTZE VON CPGESD'
O                          UDATE      70
O                          UTIME      80
O        E  1     02
O                          SATZ       80
O        E 21     03
O                                     17 '*** ANZAHL SÄTZE'
O                          COUNT Z    23
OCPGESD  E        02
O                                     40 '<--- SEQ.UPD --->'
```

Erläuterungen:

In obigem Beispiel werden Sätze sequentiell in einer ESDS-Datei upgedated und gleichzeitig als Liste auf dem Drucker ausgegeben.

Die F-Karte muss in Spalte 40 den Eintrag 'ESDS' enthalten.

Die Verarbeitung beginnt mit dem 1. Satz der Datei.

Beispiel B13 : Direkte und sequentielle Verarbeitung einer ESDS-Datei.

```
H    A                   O              B E C26

FCPGESD  U   V    8100  4R        ESDS
FPRINTER O   F     132            PRINTER

D        KEYS       99  4         SCHLÜSSEL (RBA)

LPRINTER 072FL0010106612

ICPGESD  KF
I                                       1  80 SATZ

C                    EXCPT                    01
C                    FILL X'00'     KEY     4
C                    DO   99        X       30
C           KEY      READ CPGESD
C   EF               GOTO ENDE
C                    MOVELCPGK01    KEYS,X
C                    ADD  1         COUNT   50
C                    EXCPT                    02
C                    END
C           ENDE     TAG
C                    EXCPT                    03
C                    RNDOMCPGESD
C                    DO   5         Y       30
C                    MOVELKEYS,Y    KEY
C           X        CHAINCPGESD              11
C                    EXCPT                    04
C                    END

OPRINTER E  201   01
O                                    17 'SÄTZE VON CPGESD'
O                         UDATE      70
O                         UTIME      80
O        E  1     02
O                         SATZ       80
O                                    90 'SEQUENT.'
O        E 21     03
O                                    17 '*** ANZAHL SÄTZE'
O                         COUNT Z    23
O        E  1     04
O                         SATZ       80
O                                    90 'DIREKT '
O                 11                110 'NICHT GEFUNDEN'
OCPGESD  E        04
O                                    20 '<<< DIR-UPD >>>'
```

Erläuterungen:

Hier wird eine ESDS-Datei sowohl direkt ('CHAIN'), als auch sequentiell ('READ') verarbeitet. Durch den Befehl 'RNDOM' wird von der sequentiellen Verarbeitung umgeschaltet auf die direkte Verarbeitung, gleichzeitig wird ein Protokoll auf der Datei 'PRINTER' gedruckt.
Der direkte Zugriff darf nur mit einer gültigen relativen Byte-Adresse (RBA) erfolgen. In diesem Beispiel wurden die relativen Byte-Adressen beim sequentiellen Lesen in der Feldgruppe KEYS gespeichert.

Beispiel B14 : Drucken einer RRDS-Datei.

```
H    A                   O              B E C25

FCPGRRT  I   F    8100  4K        RRDS
FPRINTER O   F     132            PRINTER

LPRINTER 072FL0010106612

ICPGRRT  KF
I                                       1  80 SATZ

C                    EXCPT                    01
C                    Z-ADD1         KEYN    40
C           ANFANG   TAG
C           KEYN     READ CPGRRT
C   EF               GOTO ENDE
C                    ADD  1         COUNT   50
C                    EXCPT                    02
C                    GOTO ANFANG
C           ENDE     TAG
C                    EXCPT                    03

OPRINTER E  201   01
O                                    17 'SÄTZE VON CPGRRT'
O                         UDATE      70
O                         UTIME      80
O        E  1     02
O                         SATZ       80
O        E 21     03
O                                    17 '*** ANZAHL SÄTZE'
O                         COUNT Z    23
```

Erläuterungen:

In obigem Beispiel wird eine RRDS-Datei gelesen und auf dem Drukker ausgegeben.

Die Verarbeitung beginnt mit dem 1. Satz der Datei.

Beispiel B15 : RRDS-Datei direkt verarbeiten.

```
 1  H    A                   O              B E C28

 2  FREADER  I   F      80            READER
 3  FCPGRRT  U   F    8100  4K        RRDS
 4  FPRINTER O   F     132            PRINTER

 5  LPRINTER 072FL0010106612

 6  IREADER  KF
 7  I                                      1   50X
 8  I                                      6   6 OP
 9  I                                      1  20 FELD
10  ICPGRRT  KF
11  I                                      1  80 SATZ

12  C                     EXCPT                    01
13  C           ANFANG    TAG
14  C                     READ READER
15  C   EF                GOTO ENDE
16  C           X         CHAINCPGRRT              11
17  C                     ADD  1         COUNT   50
18  C           OP        COMP '*'                      12
19  C           OP        COMP '+'                      13
20  C           OP        COMP '-'                      14
21  C                     MOVELX         SATZ
22  C                     MOVE FELD      SATZ
23  C                     EXCPT                    02
24  C   DR                MOVE 'DUP.REC' INFO    8
25  C                     EXCPT                    03
26  C                     GOTO ANFANG
27  C           ENDE      TAG
28  C                     EXCPT                    04

29  OPRINTER E  201   01
30  O                                        21 'CPGRRT DIREKT CHAIN, '
31  O                                        42 'UPDAT, NEWRC, DELET  '
32  O                          UDATE         70
33  O                          UTIME         80
34  O        E  1     03
35  O                          SATZ          80
36  O                11                      90 'NICHT GEFUNDEN'
37  O                12                     100 'UPDATE'
38  O                13                     100 'NEWREC'
39  O                14                     100 'DELETE'
40  O                          INFO   B      120
41  O                          X      Z       125
42  O        E 21     04
43  O                                        17 '*** ANZAHL SÄTZE'
44  O                          COUNT Z        23
45  OCPGRRT  E        02 12N11
46  O                          SATZ          80
47  O        EADD     02 13
48  O                          SATZ          80
49  O        EDEL     02 14N11
```

Erläuterungen:

1 CPG-Steuerkarte.

2 Es wird eine Datei 'READER' definiert. Die Dateiart ist 'I' Eingabedatei. Die Satzlänge ist fest (F) 80 Bytes. Die Eingabe- / Ausgabeeinheit ist 'READER' (SYSIPT).

3 Es wird eine Datei 'CPGRRT' definiert. Die Dateiart ist 'U' Updatedatei. Die Satzlänge ist fest (F) 8100 Bytes. Schlüssel ist die 4 Byte lange relative Satznummer. Die Satzadressierung ist 'K' (VSAM-KSDS und RRDS). Die Eingabe- / Ausgabeeinheit ist RRDS (VSAM RRDS - Datei).

4 Es wird eine Druckerdatei 'PRINTER' definiert. Die Dateiart ist 'O' Ausgabedatei. Die Satzlänge ist fest (F) 132 Bytes. Die Eingabe- / Ausgabeeinheit ist 'PRINTER' (SYSLST).

5 Der Drucker 'PRINTER' wird definiert. Die Formularlänge ist 72 Zeilen. 'FL' ist das Kennzeichen der Formularlänge. Zeile 1 entspricht Kanal 1 und Zeile 66 entspricht Kanal 12. Wird die Zeile, bei der Kanal 12 gesetzt ist überschritten,so wird der Überlaufanzeiger ('OF') gesetzt.

6 - 9 Die Eingabefelder der Datei 'READER' werden bestimmt. 'KF' kennzeichnet die Eingabe-Bestimmung als Satzbestimmung.

10 - 11 Die Eingabefelder der Datei 'CPGRRT' werden bestimmt. 'KF' kennzeichnet die Eingabe-Bestimmung als Satzbestimmung.

12 Die Ausgabebestimmungen werden ausgeführt.

14 - 15 Es wird ein Satz der Datei 'READER' gelesen. Bei Dateiende (EF) wird zum Merkmal 'ENDE' verzweigt.

16 Der Satz mit der im Feld 'X' enthaltenen Satznummer der Datei 'CPGRRT' wird gelesen. Ist der Satz nicht vorhanden, wird die Bezugszahl 11 gesetzt.

21 Das Feld 'X' wird linksbündig in das Feld 'SATZ' übertragen.

22 Das Feld 'FELD' wird rechtsbündig in das Feld 'SATZ' übertragen.

23 - 24 Die Ausgabebestimmungen werden ausgeführt. Bei 'DR' (Duplicate Record) wird die Konstante 'DUP.REC' nach 'INFO' übertragen.

29 In den Zeilen 30 - 44 werden die Ausgabebestimmungen für die Datei 'PRINTER' definiert.

45 In den Zeilen 46 - 49 werden die Ausgabebestimmungen für die Datei 'CPGRRT' definiert.

47 Sind die Bezugszahlen 02 und 13 gesetzt, wird der Datei ein Satz hinzugefügt.

49 Sind die Bezugszahlen 02 und 14 gesetzt und 11 nicht, wird ein Satz gelöscht.

Beispiel B16 : Laden einer RRDS-Datei vom Kartenleser.

```
H    A                   O              B E C29

FREADER  I   F      80            READER
FCPGRRT  O   F    8100  4K        RRDS
FPRINTER O   F     132            PRINTER

LPRINTER 072FL0010106612

IREADER  KF
I                                       1  80 SATZ

C                     EXCPT                    01
C           ANFANG    TAG
C                     READ READER
C   EF                GOTO ENDE
C                     EXCPT                    02
C                     GOTO ANFANG
C           ENDE      TAG
C                     EXCPT                    03

OPRINTER E  201   01
O                                     24 'FROM READER '
O                                     12 'LOAD CPGRRT '
O                         UDATE       70
O                         UTIME       80
O        E  1     02
O                         SATZ        80
O        E 21     03
O                                     17 '*** ENDE ***     '
OCPGRRT  EADD     02
O                         SATZ        80
```

Erläuterungen:

Mit oben abgebildetem Beispiel werden Karten von einem Kartenleser gelesen und in die RRDS-Datei 'CPGRRT' übertragen. Die gelesenen Sätze werden mit 'EADD' hinzugefügt, d. h. wenn bereits Sätze auf der Datei existieren, werden neue Sätze am Ende der Datei angefügt.

'O' in Spalte 15 der F-Karte kennzeichnet die Datei 'CPGRRT' als Ausgabedatei. Die Ausgabe erfolgt sequentiell.

Beim Laden der Datei wird gleichzeitig auf der Datei 'PRINTER' ein Protokoll gedruckt.

Beispiel B17 : Sequentielles Update einer RRDS-Datei.

```
H    A                   O               B E C30

FCPGRRT  U   F    8100  4K        RRDS
FPRINTER O   F     132            PRINTER

LPRINTER 072FL0010106612

ICPGRRT  KF
I                                       1  80 SATZ

C                     EXCPT                    01
C                     Z-ADD1         RECN    30
C           ANFANG    TAG
C           RECN      READ CPGRRT
C   EF                GOTO ENDE
C                     ADD  1         COUNT   50
C                     EXCPT                    02
C                     GOTO ANFANG
C           ENDE      TAG
C                     EXCPT                    03

OPRINTER E  201   01
O                                     17 'SÄTZE VON CPGRRT'
O                          UDATE      70
O                          UTIME      80
O        E  1     02
O                          SATZ       80
O        E 21     03
O                                     17 '*** ANZAHL SÄTZE'
O                          COUNT Z    23
OCPGRRT  E        02
O                                     40 '<--- SEQ.UPD --->'
```

Erläuterungen:

In obigem Beispiel werden Sätze sequentiell in einer RRDS-Datei upgedated und gleichzeitig als Liste auf dem Drucker ausgegeben.

Die Verarbeitung beginnt mit dem 1. Satz der Datei.

Beispiel B18 : Datei mit OPEN im Programm

```
H    A                   8              B E C46  PRINT CPGESD

FCPGESD  I   V    8100  4R        ESDS                          NO
FPRINTER O   F     132            PRINTER

LPRINTER 072FL0010106612

ICPGESD  KF
I                                       1  80 SATZ

C                    EXCPT                     01
C   U1               OPEN CPGESD                 99
C   99 EF            GOTO ENDE                       EMPTY FILE
C   99               SDUMP                           CANCEL PROG
C                    FILL X'00'     KEY     4
C                    DO   999
C           KEY      READ CPGESD
C   EF               GOTO ENDE
C                    ADD  1         COUNT   50
C                    EXCPT                     02
C                    END
C           ENDE     TAG
C   U1               CLOSECPGESD
C                    EXCPT                     03

OPRINTER E  201   01
O                                       17 'SÄTZE VON CPGESD'
O                         UDATE         70
O                         UTIME         80
O        E  1     02
O                         SATZ          80
O        E 21     03
O                                       17 '*** ANZAHL SÄTZE'
O                         COUNT Z       23
O                99                     44 'DATEI IST LEER'
```

Erläuterungen:

Die Datei CPGESD ist mit 'NO' in der F-Karte spezifiziert. Hierdurch wird die Datei nicht automatisch eröffnet, sondern kann gezielt mit dem Befehl OPEN eröffnet werden. In diesem Beispiel wird das OPEN nur ausgeführt, wenn der Schalter U1 (Upsi 1) gesetzt ist. Beim OPEN kann eine Bezugszahl angegeben werden, um zu prüfen, ob ein Fehler aufgetreten ist. Die Bezugszahl wird auch dann gesetzt, wenn die Datei leer ist. Bei einer leeren Datei wird zusätzlich zu der Fehlerbezugszahl noch der Schalter EF gesetzt. Hier kann in der Regel das Programm weiter fortgesetzt werden. Ist ef nicht gesetzt, so liegt ein genereller Fehler vor, und die Ausführung des Programms sollte abgebrochen werden (hier mit dem Befehl SDUMP). Auf der Systemkonsole kann der Error- und Returncode nachgesehen werden, um dann die Abbruchursache festzustellen.

Mit dem Befel CLOSE sollte die Datei bei Ende der Verarbeitung abgeschlossen werden.

Beispiel B19 : Ausgabe auf Stanzer

```
H    A                   O               B E C21  PUNCH OUTPUT
FREADER  I   F      80            READER
FPUNCHER O   F      80            PUNCHER
IREADER  KF
I                                        1  80 SATZ
C           ANFANG    TAG
C                     READ READER
C   EF                GOTO ENDE
C                     EXCPT                    01
C                     GOTO ANFANG
C           ENDE      TAG
OPUNCHER E        01
O                         SATZ      80
```

Beispiel B20 : Kopieren Platte Band

```
H    A                   8                    KOPIEREN PLATTE BAND
FIJSYS04 I   F      80            DISK
FTAPE    O   F1600  80            TAPE   SYS010
IIJSYS04 KF
I                                        1  80 SATZ
C           ANFANG    TAG
C                     READ IJSYS04
C   EF                GOTO ENDE
C                     EXCPT                    91
C                     GOTO ANFANG
C           ENDE      TAG
OTAPE    E        91
O                         SATZ      80
```

Beispiel B21 : Es kann nach mehreren Kriterien sortiert werden

```
  OPTIONS BATCH MAIN PHASE TESTPR3
          HLI H
          END
  FILE READER
  FILE SORTWORK
  FILE HSORT.                    * DATASET ZUM SORTIEREN
-D
  TITEL              20
  MSG                20.         * ERROR MESSAGE
  PAGE                3 0
  X                   5 0
  S                   3 0.        * SORTIERPARAMETER ANZAHL
  PARM          10 *  8.          * SORTIERPARAMETER SSSSLLTA
-I
  FILE READER
                 1   80   SATZ
  FILE SORTWORK
                 1   80   SATZ
  FILE HSORT HS
                 1    8   CPGTSN
          PAC    9   10 0 RC.    * RETURNCODE
                11   14   CPGHIC. * HL1 INTERFACE CONTROL
          PAC   15   17 0 X.     * RECORD NR. IN QUEUE
          PAC   18   19 0 S.     * NR OF SORT PARMS
                21  100   PARM.  * SORTPARMS
-C
  TITEL = 'BATCH TEST'
  CPGTSN = 'SORTWORK'
  DO LOOP
     READ READER
     ON EF BREAK
     EXCPT SORTWORK
  END
  PARM(1) = '000302CA'.          * VON ST. 3 LEN 2 CHAR AUFSTG.
  PARM(2) = '000101CD'.          * VON ST. 1 LEN 1 CHAR ABSTG.
  S = 2.                         * 2 SORTPARAMATER WERDEN BENUTZT
  SETLL HSORT.                   * SORTIEREN NACH SORTPARAMETERN
  IF RC = 1
     MSG = 'SORTAREA IS FULL'.
     MSG DSPLY.                  * INFO AN OPERATOR
     SDUMP.                      * CANCEL
  END
  LIST PRINTER TITEL
  DO WHILE RC = 0.               * SOLANGE SÄTZE DA SIND
     READ HSORT.                 * WELCHER SATZ IST AN DER REIHE
     ON EF BREAK.
     X READ SORTWORK
     ON EF BREAK
     LIST PRINTER SATZ
  END
  LIST PRINTER ENDE
  RNDOM HSORT.                   * SORTAREA WIRD N.MEHR BENÖTIGT
-O
  FILE SORTWORK ADD SORTWORK
            SATZ     80
```

Die Beschreibung der CPG3-Installation wurde aus diesem Handbuch ausgelagert. Bei der Erstinstallation und beim Releasewechsel wird stattdessen eine separates Installationshandbuch entsprechend Ihrer Systemumgebung ausgeliefert.

Verweise im Inhaltsverzeichnis auf Seitenzahlen von 9000 bis 9750 beziehen sich auf dieses separate Installationshandbuch.

Generieren der HL1-Programm-Tabellen 9760

HL1-Programme können wie herkömmliche Programme geschrieben werden.
Bei Online-Verarbeitung ist in Spalte 51 der H-Karte ein 'C' einzutragen.
In diesem Fall ist die Generierung der in diesem Abschnitt beschriebenen Tabellen nicht erforderlich.

Diese Tabellen und die im nächsten Abschnitt beschriebenen Bibliotheken sind erst erforderlich, wenn in einem dieser Programme die Operation 'EXHM' verwendet wird.

Die Operation:

```
    C                     EXHM XMOD
```

setzt voraus, dass eine HL1-Tabelle generiert wurde und dass diese eine Eintragung 'XMOD' enthält.

Generieren der HL1-Tabellen

Die Generierung der Tabellen erfolgt mit Hilfe des HL1 Tabellengenerators, der unter dem Phasennamen 'HL1HTG' mitgeliefert wird.
Der Generator wird aufgerufen mit

```
                          // EXEC HL1HTG
```

Eine Steuerkarte (H in Spalte 6) enthält die Parameter für die Generierung:

```
Spalte  7 -  9  Size-Angabe für die EXEC-Karte der Assembly
       10 - 10  'D' Assembler-Deck stanzen
                'P' Assembler-Deck stanzen ohne IJSYSIN
       11 - 11  'A' für Ausdruck der Assembler-Liste
       12 - 14  Adresse von SYSRDR. Bei fehlender Eintragung
                wird die Einheit 'READER' angenommen.
       22 - 22  Kurzbezeichnung der privaten Library
                Siehe unten: private HL1-Bibliotheken.
       52 - 74  Kommentar

                Bei fehlender Eintragung werden diese Daten aus
                der CPG-Standard-H-Karte übernommen.
```

Für jeden HL1-Baustein ist eine Modul-Beschreibungs-Karte erforderlich. Diese Karte hat folgenden Aufbau:

```
Spalte  6 -  6  Kartenart (muss 'M' sein)
        7 - 12  Name des Bausteins. Der Name der in Faktor2 der Operation 'EXHM' eingetragen wird.
       52 - 74  Beschreibender Text
```

Beispiel:

Für das Programm 'XMOD' wird die Karte wie folgt ausgefüllt:

```
    MXMOD                                          HL1-Baustein 'X'
```

Die folgenden Tabellen werden als Start-Set mit HL1 mitgeliefert. Die private Bibliothek 'H' (HL1) wird von Lattwein gepflegt und ist zu reservieren. Die anderen Tabellen können vom Anwender beliebig erweitert werden. Hinter der H-Karte muss immer die Karte M$ERROR eingetragen werden.

```
// JOB HL1HTG
// EXEC HL1HTG

     H                                          Allgemeine Tabelle
     M$ERROR                                    Fehler Routine
     MTEST0                                     Test 0
     MTEST1                                     Test 1
     MTEST2                                     Test 2
     MTEST3                                     Test 3
     MTEST4                                     Test 4
     MTEST5                                     Test 5
     MTEST6                                     Test 6
     MTEST7                                     Test 7
     MTEST8                                     Test 8
     MTEST9                                     Test 9

/*
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT
/&


// JOB HL1HTG
// EXEC HL1HTG

     H              A                           Tabelle A-Library
     M$ERROR                                    Fehler Routine
     MTEST0                                     Test 0
     MTEST1                                     Test 1
     MTEST2                                     Test 2
     MTEST3                                     Test 3
     MTEST4                                     Test 4
     MTEST5                                     Test 5
     MTEST6                                     Test 6
     MTEST7                                     Test 7
     MTEST8                                     Test 8
     MTEST9                                     Test 9

     DCPGHLIH                                   HL1 - Lattwein
     DCPGHLIA                                   Anwendung A

     MA0001                                     Test 1
     MA0002                                     Test 2
     MA0003                                     Test 3
     MA0004                                     Test 4
     MA0005                                     Test 5
     MA0006                                     Test 6
     MA0007                                     Test 7
     MA0008                                     Test 8
     MA0009                                     Test 9
     MA0010                                     Test 10

/*
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT
```

```
/&
```

Beschreibung der A-Library.

Mit diesem Kartensatz wird eine private HL1-Bibliothek mit dem Namen 'A' generiert (A in Spalte 22 der H-Karte). Der Name CPGHLIA ist der Phasenname dieser Tabelle in der Core Image Library.

Die allgemeine HL1 Tabelle enthält die Bausteine:

$ERROR, TEST0, TEST1, TEST2, TEST3, TEST4, TEST5, TEST6, TEST7, TEST8, TEST9.

Für jeden dieser Bausteine ist eine M-Karte ('M' in Spalte 6) erforderlich.

Die folgende D-Karte (Directory-Karte, 'D' in Spalte 6) zeigt an, dass die allgemeine Bibliothek hier endet und die erste private Bibliothek mit dem Namen 'H' folgt. M-Karten für diese Bibliothek werden nicht benötigt. Die zweite D-Karte zeigt an, dass die private Bibliothek mit dem Namen 'A' folgt.

Die private Bibliothek enthält die Bausteine:

A0001, A0002, A0003, A0004, A0005, A0006, A0007, A0008, A0009, A0010.

Für jeden dieser Bausteine ist wieder eine M-Karte erforderlich. Die Reihenfolge der M-Karten ist zwingend und darf bei späteren Änderungen der Tabelle nicht verändert werden. Eine Änderung der Reihenfolge in einer Tabelle erfordert die Umwandlung sämtlicher Programme, die auf Bausteine dieser Bibliothek zugreifen.

Beim Einsatz mehrerer privater Bibliotheken ist die Reihenfolge der D-Karten ebenfalls zwingend. Siehe folgendes Beispiel:

Es wird angenommen, dass der Benutzer nach der Installation von HL1 neben der allgemeinen Bibliothek drei private Bibliotheken für die Anwendungen 'Rechnungswesen', 'Datenerfassung' und 'Vertrieb' einrichten will. Die Bibliotheken erhalten die Namen 'R', 'D' und 'V'.

Die HL1-Programm-Tabellen können wie folgt generiert werden:

1. Eine gemeinsame Tabelle für alle Bibliotheken.

```
// JOB HL1HTG
// EXEC HL1HTG

     H                                          Universal Tabelle
     M$ERROR                                    Fehler Routine
     MTEST0                                     Test 0
     MTEST1                                     Test 1
     MTEST2                                     Test 2
     MTEST3                                     Test 3
     MTEST4                                     Test 4
     MTEST5                                     Test 5
     MTEST6                                     Test 6
     MTEST7                                     Test 7
     MTEST8                                     Test 8
```

```
      MTEST9                                    Test 9

      DCPGHLIH                                  HL1-Lattwein

      DCPGHLIR                                  Rechnungswesen

      MR0001                                    Test 1
      MR0002                                    Test 2
      MR0003                                    Test 3
      MR0004                                    Test 4
      MR0005                                    Test 5
      MR0006                                    Test 6
      MR0007                                    Test 7
      MR0008                                    Test 8
      MR0009                                    Test 9
      MR0010                                    Test 10

      DCPGHLID                                  Datenerfassung

      MD0001                                    Test 1
      MD0002                                    Test 2
      MD0003                                    Test 3
      MD0004                                    Test 4
      MD0005                                    Test 5
      MD0006                                    Test 6
      MD0007                                    Test 7
      MD0008                                    Test 8
      MD0009                                    Test 9
      MD0010                                    Test 10

      DCPGHLIV                                  Vertrieb

      MTESTV1                                   Test 1
      MTESTV2                                   Test 2
      MTESTV3                                   Test 3
      MTESTV4                                   Test 4
      MTESTV5                                   Test 5
      MTESTV6                                   Test 6
      MTESTV7                                   Test 7
      MTESTV8                                   Test 8
      MTESTV9                                   Test 9
      MANSCHR                                   Kundenanschrift

/*
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT
/&
```

Die HL1-H-Karte für die Umwandlung eines Programms, das auf einen HL1-Baustein dieser Tabelle zugreift, enthält in Spalte 22 ein Blank.

Diese Form der HL1-Tabelle ist möglich, solange die Gesamtzahl der eingetragenen Bausteine 2000 nicht übersteigt. Bei mehr als 2000 Bausteinen müssen die Tabellen nach dem folgenden Beispiel erstellt werden.

2. Getrennte Tabellen für jede Anwendung.

A. Allgemeine Bibliothek

```
// JOB HL1HTG
// EXEC HL1HTG

     H                                          Allgemeine Tabelle
     M$ERROR                                    Fehler Routine
     MTEST0                                     Test 0
     MTEST1                                     Test 1
     MTEST2                                     Test 2
     MTEST3                                     Test 3
     MTEST4                                     Test 4
     MTEST5                                     Test 5
     MTEST6                                     Test 6
     MTEST7                                     Test 7
     MTEST8                                     Test 8
     MTEST9                                     Test 9

/*
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT
/&
```

Die HL1-H-Karte für die Umwandlung eines Programms, das auf einen HL1-Baustein dieser Tabelle zugreift, enthält in Spalte 22 ein Blank.

B. Private Bibliothek Rechnungswesen

```
// JOB HL1HTG
// EXEC HL1HTG

     H              R                           Rechnungswesen
     M$ERROR                                    Fehler Routine
     MTEST0                                     Test 0
     MTEST1                                     Test 1
     MTEST2                                     Test 2
     MTEST3                                     Test 3
     MTEST4                                     Test 4
     MTEST5                                     Test 5
     MTEST6                                     Test 6
     MTEST7                                     Test 7
     MTEST8                                     Test 8
     MTEST9                                     Test 9

     DCPGHLIH                                   HL1 - Lattwein
     DCPGHLIR                                   Rechnungswesen

     MR0001                                     Test 1
     MR0002                                     Test 2
     MR0003                                     Test 3
     MR0004                                     Test 4
     MR0005                                     Test 5
     MR0006                                     Test 6
     MR0007                                     Test 7
     MR0008                                     Test 8
     MR0009                                     Test 9
     MR0010                                     Test 10

/*
```

```
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT
/&
```

Die HL1-H-Karte für die Umwandlung eines Programms, das auf einen HL1-Baustein dieser Tabelle zugreift, enthält in Spalte 22 ein 'R'.

C. Private Bibliothek Datenerfassung

```
// JOB HL1HTG
// EXEC HL1HTG

     H              D                          Datenerfassung
     M$ERROR                                   Fehler Routine
     MTEST0                                    Test 0
     MTEST1                                    Test 1
     MTEST2                                    Test 2
     MTEST3                                    Test 3
     MTEST4                                    Test 4
     MTEST5                                    Test 5
     MTEST6                                    Test 6
     MTEST7                                    Test 7
     MTEST8                                    Test 8
     MTEST9                                    Test 9

     DCPGHLIH                                  HL1 - Lattwein
     DCPGHLIR                                  Rechnungswesen
     DCPGHLID                                  Datenerfassung

     MD0001                                    Test 1
     MD0002                                    Test 2
     MD0003                                    Test 3
     MD0004                                    Test 4
     MD0005                                    Test 5
     MD0006                                    Test 6
     MD0007                                    Test 7
     MD0008                                    Test 8
     MD0009                                    Test 9
     MD0010                                    Test 10

/*
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT
/&
```

Die HL1-H-Karte für die Umwandlung eines Programms, das auf einen HL1-Baustein dieser Tabelle zugreift, enthält in Spalte 22 ein 'D'.
Es ist bei einer privaten Tabelle besonders darauf zu achten, dass die D-Eintragungen richtig vorgenommen werden.

D. Private Bibliothek Vertrieb

```
// JOB HL1HTG
// EXEC HL1HTG

     H              V                          Vertrieb
     M$ERROR                                   Fehler Routine
     MTEST0                                    Test 0
     MTEST1                                    Test 1
```

```
     MTEST2                                    Test 2
     MTEST3                                    Test 3
     MTEST4                                    Test 4
     MTEST5                                    Test 5
     MTEST6                                    Test 6
     MTEST7                                    Test 7
     MTEST8                                    Test 8
     MTEST9                                    Test 9

     DCPGHL1H                                  HL1 - Lattwein
     DCPGHL1R                                  Rechnungswesen
     DCPGHLID                                  Datenerfassung
     DCPGHLIV                                  Vertrieb

     MTESTV1                                   Test 1
     MTESTV2                                   Test 2
     MTESTV3                                   Test 3
     MTESTV4                                   Test 4
     MTESTV5                                   Test 5
     MTESTV6                                   Test 6
     MTESTV7                                   Test 7
     MTESTV8                                   Test 8
     MTESTV9                                   Test 9
     MANSCHR                                   Kundenanschrift

/*
ASSGN SYSIN,X'CUU'
// EXEC LNKEDT
/&
```

Die HL1-H-Karte für die Umwandlung eines Programms, das auf einen HL1-Baustein dieser Tabelle zugreift, enthält in Spalte 22 ein 'V'.

HL1 LIBRARY ONLINE 9770

Folgendes Programm generiert das Initialisierungsprogramm für die HL1-Online-Library.

Wenn eine neue private Library angelegt wird, oder der HL1-Pool (Standard 65520 Bytes) vergrößert werden soll, muss folgendes Programm neu katalogisiert werden.

```
// JOB CPGHLB         HL1 Library Online
// OPTION CATAL
   PHASE CPGHLB,*
// EXEC ASSEMBLY
       COPY  CPG*HHLO         * = HL1-Release-Suffix
       END
/*
// EXEC LNKEDT
/&
```

Das Initialisierungs-Programm enthält zwei Copy-Books.
Das erste Book enthält das jeweilige Initialisierungs-Programm, das

zweite Book enthält die benutzerspezifischen Daten.

Das Copy Book CPGUHHOU kann vom Benutzer entsprechend modifiziert werden und wird wie folgt mitgeliefert:

```
   CATALS A.CPGUHHOU
          BKEND
*-------------------------------------------------------------------*
*           PRIVATE DIRECTORYS + LIBRARYS                           *
*-------------------------------------------------------------------*
CPGHDH   DC    500F'0'                 500 DIR. EINTR. HL1
CPGHDA   DC    100F'0'                 100 DIR. EINTR. PRIV. ANWENDUNG
*--------------------------------------PRIV. LIBRARYS---------------*
         ORG   CPGHHLIB
         DC    C'H',AL3(CPGHDH-CPGHLB)
         DC    C'A',AL3(CPGHDA-CPGHLB)
*--------------------------------------POOL GRÖßE-----------------*
         ORG
         DC    CL8'POOLANF'
POOLANF  DC    16380F'0'               65.520 BYTES (16380*4)
        BKEND
```

Beim Einsatz mehrerer privater Bibliotheken ist die Reihenfolge der CPGHD*-Karten zwingend. Siehe folgendes Beispiel:

Es wird angenommen, dass der Benutzer nach der Installation von HL1 neben der allgemeinen Bibliothek drei private Bibliotheken für die Anwendungen 'Rechnungswesen', 'Datenerfassung' und 'Vertrieb' einrichten will.

Das Copy CPGUHHOU muss wie folgt abgeändert und mit // EXEC MAINT in die SL katalogisiert werden:

```
   CATALS A.CPGUHHOU
          BKEND
*-------------------------------------------------------------------*
*           PRIVATE DIRECTORYS + LIBRARYS                           *
*-------------------------------------------------------------------*
CPGHDH   DC    500F'0'                 500 DIR. EINTR. HL1
CPGHDR   DC    200F'0'                 200 DIR. RECHNUNGSWESEN
CPGHDD   DC    500F'0'                 500 DIR. DATENERFASSUNG
CPGHDV   DC     30F'0'                  30 DIR. VERTRIEB
*--------------------------------------PRIV. LIBRARYS---------------*
         ORG   CPGHHLIB
         DC    C'H',AL3(CPGHDH-CPGHLB)
         DC    C'R',AL3(CPGHDR-CPGHLB)
         DC    C'D',AL3(CPGHDD-CPGHLB)
         DC    C'V',AL3(CPGHDV-CPGHLB)
*--------------------------------------POOL GRÖßE-----------------*
         ORG
         DC    CL8'POOLANF'
POOLANF  DC    16380F'0'               65.520 BYTES (16380*4)
        BKEND
```

Achtung: Um ein HL1-Programm ausführen zu können, muss die HL1-Library online geladen sein. Siehe Seite 9616, den Abschnitt für HL1-Anwender.

Zweiter HL1-Pool 9780

Wenn in Ihrem Hause ein zweiter HL1-Pool eingerichtet werden soll, so Muss ein zweiter HL1-Pool katalogisiert werden:

```
// JOB CPGHL2
// OPTION CATAL
   PHASE CPGHL2,*
// EXEC ASSEMBLY
       COPY  CPG*HHL2         * = HL1-Release-Suffix
       END
/*
// EXEC LNKEDT
/&
```

Das Initialisierungs-Programm enthält zwei Copy-Books.

Das erste Book enthält das jeweilige Initialisierungs-Programm, das zweite Book enthält die benutzerspezifischen Daten. (Größe des 2. HL1-Pools).

Das Copy Book CPGUHHO2 kann vom Benutzer entsprechend modifiziert werden und wird wie folgt mitgeliefert:

```
   CATALS A.CPGUHHO2
         BKEND
*-------------------------------------------------------------------*
*              HL1  POOL 2  (GRÖßE)                                 *
*-------------------------------------------------------------------*
         DC    2000F'0'                 8.000 BYTES (2000*4)
         BKEND
```

Stichwortverzeichnis CPG 9900