Befehle	Anschlußcode an Kanal 1	Komplement
Data	17 hexadezimal	C2 hexadezimal
Transmit	16 "	entfällt
Receive	14 "	entfällt
Test	15 "	entfällt
Phone	17 "	C1 hexadezimal

2.10.3. Bedienung

2.10.3.1. Steuerpult

Das Steuerpult des DATANET 51 weist nachstehende Tasten und Leuchtanzeigen auf:

- Tasten

ON	Stromversorgung "Ein"	Im Schalter leuchtet eine Kontroll-Lampe auf, wenn die Einheit eingeschaltet ist.
OFF	Stromversorgung "Aus"	Im Schalter leuchtet eine Kontroll-Lampe auf, wenn die Einheit ausgeschaltet ist, aber die Netzspannung noch vorhanden ist.
TPH	Telefon	Die Betätigung des Schalters versetzt das DATANET 51 in den Betriebszustand "TALK" (nur, wenn die Kontroll-Lampe MAN leuchtet). Kontroll-Lampe leuchtet: Fernsprechapparat mit der Übertragungsleitung verbunden. Kontroll-Lampe erloschen: DATANET 51 mit der Übertragungsleitung verbunden.

600	600 Bit pro Sekunde	Diese beiden Schalter ermöglichen die Auswahl der erforderlichen Schrittgeschwindigkeit für die Datenfernübertragung.
1200	1200 "	

Hinweis:

Statt 600 und 1200 können die Ziffern 1200, 2400 vorhanden sein.

MAN	Manual	Manuelle Anschaltung der Übertragungsleitung (durch Taste TPH) *
RIN	Ring Indicator	Automatische Verbindungsannahme *

*) Die entsprechende Kontroll-Lampe leuchtet bei Betätigung auf.

Leuchtanzeigen

DSR	Diese Kontroll-Lampe leuchtet auf, wenn der Modem mit der Übertragungsleitung verbunden ist.
SD	Diese Kontroll-Lampe leuchtet auf, wenn das DATANET 51 Daten sendet
WBR	Diese Kontroll-Lampe leuchtet auf, wenn das DATANET 51 im Empfangszustand ist und keine Daten empfangen werden.
RD	Diese Kontroll-Lampe leuchtet auf, wenn das DATANET 51 Daten empfängt.

2.10.3.2. Programmierhinweise

Die Daten für den Sendevorgang müssen bereitgestellt und können in einem beliebigen Puffer des Kernspeichers empfangen werden; zum Senden und Empfangen können getrennte oder auch gemeinsame Puffer benutzt werden.

Die Steuerzeichen werden beim Senden und Empfangen in den vom Benutzerprogramm vorgegebenen Puffer geschrieben oder aus ihm gelesen.

Es ist nicht möglich, das DATANET 51 einzusetzen, ohne vorher einen geeigneten Dialogverkehr zu vereinbaren, der die Probleme des Anrufens, Quittierens, Wiederholens, usw.....regelt. Eine Hilfe hierfür ist GECOR (General Electric Communication Routin).

3. Die Basisbefehle des GE-55

3.1. Einführung

Dem Benutzer des GE-55 stehen zur Programmierung der Arbeiten zwei symbolische Programmiersprachen zur Verfügung:

- die "Grundsprache" und
- der "Assembler Trommel", der die Grundsprache bei Verwendung der Magnettrommel ergänzt.

Diese Sprachen werden durch Standardprogramme interpretiert, die nach dem Laden die Programme in eine für den Rechner verständliche Maschinensprache umsetzen. Der "Assembler Trommel" ist in dem Handbuch "Programmiersystem GE-55 - Trommel" beschrieben.

Die "Grundsprache" ist Gegenstand des vorliegenden Kapitels.

Die Programme werden auf Kodierformularen (Best.-Nr. BULL 1009) geschrieben, die gleichzeitig Belege zum Ablochen der Programmkarten sind. Ein Musterformular wird in Kapitel 5 beschrieben, in dem ebenfalls die Richtlinien zur allgemeinen Programmierung gegeben werden.

3.1.1. Darstellung der Grundsprache

3.1.1.1. Allgemeine Definitionen

Diese Sprache heißt "Grundsprache", weil sie erstens für alle Konfigurationen des GE-55 anwendbar ist und zweitens der internen Maschinensprache am nächsten kommt.

Sie besteht aus Elementarbefehlen und symbolischen Pseudobefehlen.

Jedem Elementarbefehl entspricht ein Befehl in Maschinensprache, der von der Zentraleinheit verstanden wird. Er besteht aus einem symbolischen Schlüssel, der den Operationstyp identifiziert und einer variablen Anzahl Parameter, die

- Ergänzungsschlüssel (z.B. absolute Werte, Längenangaben, Codes der Randeinheiten etc.) oder
- Adressen von Daten oder Zonen sein können.

Einem Pseudobefehl dagegen entspricht kein Maschinenbefehl. Er dient lediglich dazu, Steuerimpulse an das "Programmsystem" zu geben, das nach dem Laden ein Programm in Maschinensprache umwandelt, z.B. Bestimmen der reellen Adresse bei einem Programmsprung.

Die Pseudobefehle entsprechen in ihrem Aufbau den Elementarbefehlen und können darum jeden beliebigen Platz in einem vom Programmierer erstellten Programm belegen.

Ein in der Grundsprache geschriebenes Programm kann an beliebiger Stelle im Zentralspeicher stehen, natürlich im Rahmen seiner Gesamtkapazität. So kann man auch, unter den gleichen Bedingungen, Verschiebungen vornehmen, um Änderungen einzusetzen.

Die Grundsprache ermöglicht es auch, Programme in "Sektionen"aufzuteilen, die in gleicher Weise in beliebiger Reihenfolge eingespeichert und verschoben werden können. Eine "Sektion" wird als eine Folge von Befehlen verstanden, die immer nacheinander ablaufen. Um sie zu identifizieren genügt es,die reelle Adresse durch eine Verweisung zu symbolisieren. Durch einen solchen Programmaufbau kann man innerhalb einer Sektion Änderungen vornehmen, die im allgemeinen in einem Hinzufügen oder Wegnehmen von Befehlen bestehen, ohne die nicht betroffenen Sektionen zu berühren.

Um ein Grundprogramm in ein von der Maschine auswertbares Programm zu übertragen, übersetzt das "Programmsystem" die symbolischen Operationstypen in Binärcodes oder Operationstypen in Maschinensprache, teilt jedem Befehl eine Adresse im Kernspeicher zu und setzt aufgrund der symbolischen Verweisungen jeder Sektion die Adresse des ersten Befehls dieser Sektion ein.

3.1.1.2. Die einzelnen Befehlstypen

Die in den folgenden Abschnitten beschriebenen Befehle lassen sich in folgende große Gruppen mit gemeinsamen Merkmalen einteilen:

(3.2.) Befehle zur Programmverknüpfung: Programmsprünge.

(3.3.) Registeroperationen: Rechenoperationen, numerische Vergleiche, Verschiebungen und Überträge.

(3.4.) Operationen auf Zeichenbasis: Logische Operationen und Überträge im Normalspeicherbereich.

(3.5.) Mehrfachüberträge: Von einer Zone nach mehreren anderen.

(3.6.) Ein-/Ausgabeoperationen: Überträge zwischen dem Zentralspeicher und den Randeinheiten.

(3.7.) Umschlüsselungsbefehle: Von externen Codes in den internen Code und umgekehrt.

(3.8.) Hilfsbefehle: Zum Testen und Laden der Programme.

Eine achte Gruppe von Befehlen, die sich auf das Multiprogramming beziehen, sind in Kapitel 4 beschrieben.

3.1.2. Befehlsformat

3.1.2.1. Die Maschinenbefehle

Das Befehlsformat hängt von der Art des Befehles ab. Von ihrer Struktur her kann man zwei Hauptklassen unterscheiden:

- einfache Befehle; Es erfolgt eine Operation, entweder ein Übertrag, eine logische Analyse oder eine Umwandlung. Sie geben eine große Flexibilität bei der Datenverarbeitung.

- komplexe Befehle; Es erfolgen mehrere Operationen, entweder Lesen oder Schreiben auf externe Randeinheiten oder gleichzeitiger Übertrag und Umwandlung. Diese Befehle entsprechen ganz besonders den Erfordernissen bei der Eingabe von Daten und bei der Ausgabe von Resultaten und optimieren die Leistung eines Programmes, indem sie seinen Umfang und die Zeit der Entschlüsselung der Operationstypen verringern. Sie geben eine große Beweglichkeit bei Operationen mit den Randeinheiten und ihrer Simultaneität.

Die einfachen Befehle

Die einfachen Befehle erfordern einen oder zwei Operanden. Sie bestehen aus:

- einem Operationstyp
- evtl. einem Ergänzungsschlüssel, der
  - . entweder eine Zahl N ist, die die Länge des (oder der) Operanden angibt
  - . oder ein Wert K, der entweder ein Bezugszeichen oder ein Operand ist.
- einer oder zwei Adressen, die die Operanden bezeichnen (A1 und A2).

An dieser Stelle sei darauf hingewiesen, daß die allgemeinen Adressen A ebenfalls die Nummer R eines numerischen Registers oder die Nummer B eines Basisregisters, ergänzt durch eine Verschiebung xxx, sein können (s. 2.1.4.).

Die komplexen Befehle

Diese Befehle setzen sich zusammen aus:

- einem Operationstyp
- einem Nebenschlüssel N, der die Anzahl der durchzuführenden Operationen angibt
- den Adressen der entsprechenden Ein-/Ausgabemedien.

Zu diesen Befehlen gehören:

- die Befehle der Mehrfachüberträge, bei denen die Adressen durch die Verschiebungen D (die Basisregister sind einbegriffen) oder durch die Nummer R des numerischen Registers dargestellt werden.
- die Ein-/Ausgabebefehle, bei denen die Randeinheiten durch Spezialcodes C und die Ein-/Ausgabezonen im Zentralspeicher durch die Nummern R der numerischen Register identifiziert werden.

Die speziellen Formate dieser Befehle werden in Abschnitt 3.5 und 3.6 beschrieben.

Format der Parameter

. Der Operationstyp ist ein Binärschlüssel und belegt ein Byte.

. Der Nebenschlüssel N ist eine zweistellige Dezimalzahl in gepackter Form auf einem Byte.

. Der Nebenschlüssel K ist ein Binärwert oder eine Kombination des IS$\emptyset$-Codes auf einem Byte.

. Die Adressen belegen in der Form von zwei-, drei- oder vierstelligen gepackten Dezimalzahlen ein oder zwei Bytes.

Die Art der Adressierung ist dabei gleichgültig:

- absolute Adressierung: 4 Ziffern (2 Bytes) bezeichnen die Nummer eines Bytes.
- Adressierung der Register: 2 Ziffern (1 Byte) bezeichnen die Nummer eines numerischen Registers oder eines Basisregisters, das die Adresse einer Speicherzone enthält.
- Adressierung einer Sektion in einer Speicherzone:
  . mit ausführlicher Basis: 4 Ziffern (2 Bytes), von denen die 1. die Nummer des Basisregisters und die drei anderen die Verschiebung darstellen.

mit einbezogener Basis: - 2 Ziffern (1 Byte) stellen eine Verschiebung ⩽ 99 dar,

- 3 Ziffern mit vorangestelltem Buchstaben A (2 Bytes) stellen eine Verschiebung zwischen 100 und 999 dar.

. Der Anschlußcode C ist ein Binärcode und belegt 1 Byte.

Die Länge der Befehle ist variabel, aber immer gleich einer Zahl ganzer Bytes. Sie wird bestimmt durch die Analyse des Operationstyps und des Nebenschlüssels bei komplexen Befehlen.

3.1.2.2. Die Befehle der Grundsprache

Die Elementarbefehle

Die Elementarbefehle, aus denen ein Programm besteht, werden vom Programmierer im Format der Maschinenbefehle geschrieben, die ihnen entsprechen. Wie die Maschinenbefehle besitzt auch der Elementarbefehl einen Operationstyp, evtl. einen Nebenschlüssel und eine oder mehrere Adressen.

. Der Operationstyp wird symbolisch durch einen numerischen Schlüssel dargestellt, der sich aus Buchstaben und Ziffern für bestimmte Befehle zusammensetzt.

Beispiel:

	OT symb.	Binärcode
- Addition in einfacher Länge	ADD	1010 0001

Der Binärcode des OT in Maschinensprache erscheint bei einem Kernspeicherausdruck oder bei Auflisten eines Programms in Form von zwei hexadezimalen Zeichen nach den Normen, die in Abschnitt 2.1.2.1 beschrieben wurden. (z.B. 1010 0001 = A1)

. Der Nebenschlüssel N, dargestellt durch Dezimalziffern, kann eine Zahl von 00 bis 99 sein. Bei bestimmten Befehlen kann dieser Wert jedoch niedriger sein.

. Der Nebenschlüssel K und die Anschlußcodes C werden dargestellt durch zwei hexadezimale Zeichen (s. 2.1.2.1). Die Tafel des internen Codes in Kapitel 6 zeigt den Zusammenhang zwischen den Binär- und hexadezimalen Kombinationen.

Beispiele:

1) K = A4; K = 31; K = BC
2) Anschlußcode Drucker MB = CA

. Die Adressen werden durch Dezimalziffern nach den gleichen Bedingungen wie bei den Maschinenbefehlen dargestellt.

Beispiele:

- absolute Adresse: 4021 bezeichnet das Byte 4021 im Zentralspeicher
- Adresse der Register: 04 bezeichnet das 4. numerische Register.
- Basisadresse: 4021 bezeichnet als Basisregister: Register 4 und als Verschiebung 21. Enthält Reg. 4 z.B. 2320, so lautet die absolute Adresse: 2341 (2320 + 21).
- Adresse mit einbezogener Basis: 62 bezeichnet eine Verschiebung von 62 Bytes; A 105 eine von 105 Bytes. Die einbezogene Basis ist eins der Register 6-9. Enthält Reg. 8 z.B. 2114, so ist die absolute Adresse im 1. Fall: 2176 und im 2. Fall: 2219.

Der Anfang der Sektionen eines Programms wird durch ein symbolisches Bezugszeichen gekennzeichnet, das sich aus zwei hexadezimalen Zeichen (*) zusammensetzt. Dieses Symbol bezeichnet bei Verwendung in einem Pseudobefehl die Adresse des ersten Befehls einer Sektion.

(*) Anmerkung: Die Kombinationen F1 - FF sind verboten. Sollte der Programmierer eines dieser Zeichen unbedingt verwenden müssen, ist es erforderlich, innerhalb des Programms eine Serie zu programmieren, in der diese Zeichen durch logische Operationen zusammengesetzt werden. Dieser Vorgang ist nicht anwendbar bei den symbolischen Bezugszeichen. Es bleiben jedoch 240 Kombinationen zur freien Verfügung.

Die Pseudobefehle

Die Pseudobefehle werden in der gleichen Weise geschrieben wie die Elementarbefehle und enthalten einen symbolischen Operationstyp und evtl. symbolische Parameter.

3.1.3. Darstellung der Befehle

Die in den folgenden Abschnitten beschriebenen Befehle sind in folgende Gruppen eingeteilt:

- Befehle zur Programmverknüpfung (3.2)
- Registeroperationen (3.3)
- Operationen auf Zeichenbasis (3.4)
- Mehrfachüberträge (3.5)
- Ein/Ausgabeoperationen (3.6)

- Umschlüsselungsbefehle (3.7.)
- Hilfsbefehle (3.8.)
- Pseudobefehle (3.9.)

Für jede Befehlsgruppe enthält ein Einführungsparagraph die gemeinsamen Charakteristika.

Die besonderen Eigenschaften jedes Befehls werden in folgender Weise dargestellt:

- symbolischer Operationstyp und englische Bezeichnung,
- Befehlsformat in Maschinensprache: Operationscode und Ergänzungsparameter,
- Länge des Befehls in Bytes,
- besondere Normen der Anwendung und der Parametrierung im gegebenem Fall,
- Wirkung, Anwendung und Besonderheiten,
- Beispiele.

Falls mehrere Befehle identische Eigenschaften haben, werden sie in einer gemeinsamen Beschreibung zusammengefaßt. Am Anfang des Abschnittes sind in einer Tabelle die verschiedenen symbolischen Operationstypen und die entsprechenden OT im Maschinencode angegeben.

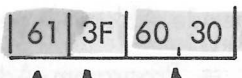
Innerhalb der Beispiele sind die Werte, die im Speicher enthalten sind, wie folgt dargestellt:

- der größte Teil in hexadezimaler Form (1 Zeichen je Byte),
- einige in Binärdarstellung (acht Ziffern (0/1) je Byte),
- einige in Klartext (2 Zeichen je Byte).

Eine Information oder ein Befehl besteht aus einer Folge von Zeichen, die immer in ein Schema gezeichnet sind, das die Speicherstellen (oder Bytes) darstellt, das diesen Wert enthält. Die interessanten Zonen (Register, Information) sind mit Markierungen versehen.

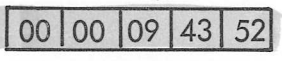
Beispiel:

Ein Befehl:



Byte 1 : OT
Byte 2 : K
Byte 3 + 4 : Adresse

Ein Register: (5 Bytes)



Am Ende des Handbuches geben Tabellen Auskunft über die Ablaufzeit eines Befehles und verweisen auf die entsprechenden Abschnitte , in denen der Befehl beschrieben wurde.

Definitionen

Die verschiedenen Elemente eines Befehls werden durch folgende Zeichen dargestellt:

OT = Operationstyp

N = Nebenschlüssel dezimal, oder zwei gepackte Dezimalziffern.

K = Nebenschlüssel hexadezimal (Binärwert) oder zwei gepackte hexadezimale Zeichen.

C = Anschlußcode (hexadezimal) einer Randeinheit.

R = Nummer eines Registers (2 Dezimalziffern)

A = 1 alphanumerisches ungepacktes Zeichen

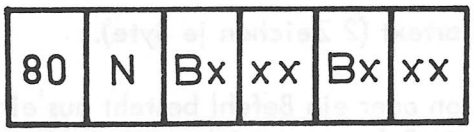
xxxx = absolute Adresse (dezimal)

Bxxx = Basisadresse (dezimal) oder:
B = Nummer des Basisregisters, das die Basisadresse enthält
xxx = Verschiebung oder Ergänzung der Adresse. Dieser Wert wird zur Basis addiert, um die absolute Adresse zu erhalten.

Dn = Verschiebung (dezimal) in bezug auf eine Basis n.

Die Befehle werden so eingerahmt, daß ihr Format und die Anzahl der Bytes sichtbar werden.

Beispiel:



Binärcodes des Operationstypes

Folgende Abkürzungen werden ebenfalls verwendet:

PAR = Programm-Adreß-Register

ANR = Adresse nach Rücksprung

3.2. Die Befehle zur Programmverknüpfung

3.2.1. Adressierung der Sprünge

Der Programmierer hat die Wahl zwischen zwei Formulierungen: Er kann die Sprünge mit reellen Adressen oder mit Symbolen programmieren.

a) reelle Adressen

Der Programmierer prüft die effektive Einspeicherung des Programmes im Zentralspeicher, um die Adresse jedes Sprunges zu bestimmen. Die anzugebende Adresse ist die Stelle, die der Operationstyp des Befehls belegt, mit dem nach dem Sprung fortgefahren werden soll.

b) Symbole

Der Programmierer springt aus dem Programm auf Sprungniveaus, die durch den Pseudobefehl LEVEL (Abschnitt 3.9.) dargestellt werden. Jedes Niveau erhält ein besonderes Etikett, das die Sprungadresse ersetzt.

Der Programmierer setzt dieses Symbol in den Sprungbefehl in folgender Weise ein:

| F1 | K |

F1 = festes Zeichen, das besagt, daß es sich um ein Bezugszeichen handelt.

K = ein Zeichen in hexadezimaler Form, das dieses Bezugszeichen darstellt. Die Kombinationen F1 bis FF sind verboten.

Um von der Maschine verarbeitet werden zu können, werden die Symbole durch reelle Adressen des Sprungniveaus ersetzt. Dieser Austausch findet beim Laden des Programmes in den Kernspeicher durch das Standardladeprogramm statt.

Wahl der Methode

Die erste Methode setzt voraus, daß das Programm völlig fertiggestellt ist, damit die reellen Sprungadressen festgelegt werden können. Weiterhin ist die Veränderung eines Programmteiles mit einer Revision auch der Sprungbefehle, die nicht direkt berührt werden, verbunden. Das ist eine ständige Quelle des Ärgers und der Schwierigkeiten.

Die zweite Methode enthält nicht die vorgenannten Unbequemlichkeiten.

Die symbolische Adressierung kann in dem Augenblick festgelegt werden, in dem das Programm entsteht. Der Programmierer setzt die Sprungniveaus in das Flußdiagramm ein und ordnet die Bezugszeichen zu. Nach Maßgabe

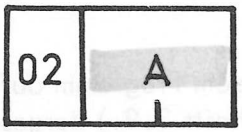
des Programmes schiebt er die Pseudobefehle LEVEL ein und vermerkt die Symbole in den Sprungbefehlen.

Spätere Veränderungen des Programmes bleiben ohne Einfluß auf die Sprungbefehle, wenn sie nicht gerade selbst benutzt werden. In diesen Fällen werden die neuen reellen Adressen automatisch während des Ladens des veränderten Programmes in den Kernspeicher festgelegt.

3.2.2. Sprungbefehle

3.2.2.1. Unbedingter Sprung

J R T	Jump; save adress for Return



A (Sprungadresse)	reelle Adresse	: | xx | xx |
	symbolische Bezugszeichen	: | F1 | K |

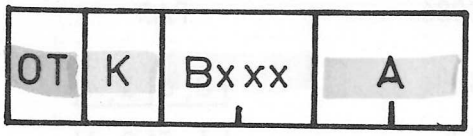
Der Inhalt des Programm-Adress-Registers wird im Register der Adresse bei Rücksprung (ANR) gespeichert (Adresse des dem JRT folgenden Befehls). Die im JRT festgelegte Sprungadresse wird in das PAR übertragen. Das Programm fährt mit dieser Adresse fort.

Beispiele:

Adresse	Programm		PAR	ANR
734	JRT | 0 2 | 1 2 | 8 7 |	vorher:	0734	0000
737	ADD | A 1 | 3 2 | 1 6 |	nachher:	1287	0737

3.2.2.2. Bedingte Sprünge

JIERT	Jump if Immediate octet Equal to store octet; save adresse for ReTurn
JIURT	Uniqual



OT = 04 JIERT
05 JIURT

K = Bezugszeichen

Bxxx = Adresse der zu untersuchenden Stellen

A = Sprungadresse { reelle Adresse |xx |xx|
Etikett des Niveaus |F1 |xx|

Der Programmsprung ist abhängig vom Vergleichsergebnis zwischen dem Zeichen K und dem Zeichen, das sich an der Adresse Bxxx befindet.

- bei JIERT wird der Sprung ausgeführt bei Gleichheit zwischen den beiden Zeichen.

- bei JIURT wird der Sprung durchgeführt, wenn Ungleichheit zwischen den Zeichen besteht.

a) Bei Sprungausführung

Der Inhalt des PAR wird im ANR gespeichert (Adresse des Befehles, der auf den Befehl JIERT oder JIURT folgt)

Die Sprungadresse des JIERT oder JIURT wird in das PAR übertragen und das Programm fährt an der Stelle fort, die dieser Adresse entspricht.

b) Bei Nichtausführung des Sprunges

Der Inhalt des PAR bleibt unverändert und das Programm fährt mit dem Befehl, der dem JIERT oder JIURT folgt, fort.

Beispiele:

a) JIERT (Sprung wird ausgeführt)

Adresse 654 |0,4|3,E|0,0|8,6|1,5|0,1| Basis 0 = 0000

	Stelle 0086 (VG)	PAR	ANR
vorher	: 3 E	|0,6|5,4|	|x,x|x,x|
nachher	: 3 E	|1,5|0,1|	|0,6|6,0|

b) JIURT (Sprung wird nicht ausgeführt)

Adresse 654 |0,5|0,0|0,0|9,1|1,2|1,8| Basis 0 = 0000

	Stelle 0091 (Kap)	PAR	ANR
vorher	: 0 0	|0,6|5,4|	|x,x|x,x|
nachher	: 0 0	|0,6|6,0|	|x,x|x,x|

3.3. Registeroperationen

3.3.1. Allgemeines

Die Operationen, die in den Registern durchgeführt werden, beziehen sich im wesentlichen auf die Verarbeitung von numerischen Daten, die in gepackter, algebraischer Form dargestellt werden. (s. 2.1.2.2.) Die Verarbeitung erfolgt in einfacher oder doppelter fester Länge (s. 2.1.3.3.).

Die Befehle dieser Gruppe enthalten:

- Rechen- und algebraische Vergleichsbefehle,
- Versetzungs- und Rundungsbefehle,
- Übertragungsbefehle.

Ihre Beschreibung und die angeführten Beispiele beziehen sich nur auf Dezimalzahlen.

Die Benutzung dieser Befehle setzt die folgenden allgemeinen Richtlinien voraus.

Löschen der Register

Es besteht kein besonderer Löschbefehl.
Ein Löschen der Register besteht darin, jedes Halbbyte auf Null zu setzen.

Man erreicht dies, indem man das Register mit dem Zeichen für "Nichts" (00) belegt, entweder durch den Befehl "Einsetzen eines Zeichens" -INC- (s. 3.4.3.2.) oder durch den Übertrag des Inhaltes eines anderen Registers, das nichts enthält.

Da die Zeichen "0" (30) und "Leerzeichen" (20) ein Halbbyte ungleich Null enthalten, können diese Zeichen nicht benutzt werden.

Versetzung

Die Rechenbefehle und der Vergleich erfordern Operanden, die in den numerischen Registern rechtsbündig stehen. Wird ein Operand nach links verschoben, so daß die rechten Stellen mit Null aufgefüllt werden, wird der Wert so behandelt, als ob er mit 10^n multipliziert worden wäre.
n entspricht der Anzahl der Nullen rechts.

Die Rechenergebnisse stehen immer rechtsbündig in den Registern.

Kapazitätsüberschreitung

Je nachdem es sich um ein Einfach- oder Doppelregister handelt, kann ein numerisches Register 9 oder 19 Ziffern und ein Vorzeichen (C oder 9) enthalten.

Es ist Aufgabe des Programmierers darauf zu achten, daß die Rechenoperanden die besonderen Normen jedes Befehls beachten, so daß sie niemals die erlaubte Maximalkapazität überschreiten. Die Kapazität, die man für einen Operanden in Betracht ziehen muß, ist gleich der Maximalzahl von Ziffern, die er enthält, zuzüglich eventueller Nullen rechts im Register.

Beispiel:	Register	zu beachtende Kapazität
Operand rechtsbündig:	0,0,0,0,0,x,x,x,x,x	5 Ziffern
Operand versetzt:	0,0,x,x,x,x,x,0,0,0	8 Ziffern

Am Ende der Rechenoperationen prüft die Maschine das erhaltene Ergebnis. Diese Kontrolle besteht darin, den Wert der Vorzeichenstelle des Resultats zu analysieren. Ein Wert ungleich 0 oder 9 zeigt an, daß es sich um eine Kapazitätsüberschreitung handelt und daß dadurch das Resultat unauswertbar ist (falsches Vorzeichen). In diesem Fall wird ein Testzeichen in das Register "Kapazitätsüberschreitung" gesetzt.

Bei der Division wird dieses Testzeichen dargestellt durch AA. Bei den anderen Operationen ist es ein unbestimmbarer Wert; nämlich der Inhalt des äußersten linken Bytes des Resultats. Dieser Inhalt ist immer ungleich 00 und variabel.

Um sich von der Art des Resultats zu überzeugen, geht der Programmierer nach folgender Methode vor:

- Vor der Rechenoperation setzt er das Register "Kap" auf Null (00).

- nach der Operation analysiert er diese Stelle durch einen Vergleich gegen 00, um festzustellen, ob sich der Inhalt geändert hat.

Diese Methode ist unabhängig vom Wert des Testzeichens gültig.

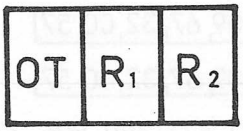
Nicht dezimale Werte

Das Vorhandensein von nicht dezimalen Werten (größer 9) in den Registern wird von den Befehlen nicht festgestellt. Sie laufen ohne einen eventuellen Fehler anzuzeigen ab. Bei den Rechen- und Rundungsbefehlen erhält man falsche Resultate. Bei Versetzungsbefehlen dagegen kann es geduldet werden.

3.3.2. Rechenbefehle und algebraischer Vergleich

3.3.2.1. Additionen und Subtraktionen

ADD	ADd Decimal single length	A 1
ADDD	ADd Decimal Double length	A 2
SBD	SuBtract Decimal single length	B 1
SBDD	SuBtract Decimal Double length	B 2



OT : siehe Maschinencode in obenstehender Tabelle

R1 und R2 : Einfach- oder Doppelregister je nach OT

Der Inhalt des Registers 1 wird auf den Inhalt des Registers 2 addiert oder von ihm subtrahiert. Das Resultat steht in Register 2. Wenn Register 1 ein anderes als Register 2 ist, wird sein Inhalt durch die Operation nicht verändert.

Der Befehl wird algebraisch durchgeführt.

Wenn R1 und R2 das gleiche Register bezeichnen, wird der Inhalt dieses Registers verdoppelt oder gelöscht, je nachdem, ob eine Addition oder Subtraktion vorlag.

Die Operation bezieht sich auf Register von

- 5 Bytes (9 Ziffern und Vorzeichen) bei ADD und SBD (einfache Länge)

- 10 Bytes (19 Ziffern und Vorzeichen) bei ADDD und SBDD (dopp. Länge)

Überschreitet das Ergebnis 9 oder 19 Ziffern, liegt Kapazitätsüberschreitung vor. Das Resultat bleibt erhalten, das höchste Byte wird aber in das Register Kap (0091) übertragen.

Beispiele:

A) Addition, einfache Länge: ADD | A1 | 27 | 35 |

	Reg. Kap.	Reg. 27	Reg. 35
a) vorher:	00	00,76,93,21,69	99,99,83,29,67
nachher:	00	00,76,93,21,69	00,76,76,51,36

(+ 76.932.169) + (- 167.033) = (+ 76.765.136)

Es fand keine Kapazitätsüberschreitung statt.

b) vorher:	00	09,67,32,00,57	03,99,50,63,39
nachher:	13	09,67,32,00,57	13,66,82,63,96

(+ 967.320.057) + (+ 399.506.339) = (+ 1.366.826.396)

Es fand eine positive Kapazitätsüberschreitung statt.

B) Substraktion, einfache Länge: SBD | B1 | 50 | 97 |

	Reg. Kap.	Reg. 50	Reg. 97
a) vorher:	00	00,76,93,21,69	00,00,16,70,33
nachher:	00	00,76,93,21,69	99,23,23,48,64

(+ 167.033) - (+ 76.932.169) = (- 76.765.136)

Es fand keine Kapazitätsüberschreitung statt.

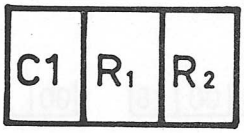
b) vorher:	00	07,36,29,92,02	95,21,07,63,22
nachher:	87	07,36,29,92,02	87,84,77,71,20

(- 478.923.678) - (+ 736.299.202) = (- 1.215.222.880)

Es fand eine negative Kapazitätsüberschreitung statt.

3.3.2.2. Multiplikation

MPD — Multi Ply Decimal



R1 = Einfachregister Multiplikator

R2 = Doppelregister Multiplikand

Der Inhalt des Doppelregisters R2 wird multipliziert mit dem Inhalt des Einfachregisters R1. Das Produkt befindet sich am Ende der Operation im Register 2. Ist das Register 1 ein anderes als Doppelregister 2, so wird sein Inhalt durch die Operation nicht verändert.

Die Operation erfolgt algebraisch.

Größe der Faktoren:
- Multiplikator : 1 - 9 Ziffern + VZ
- Multiplikand : 1 - 19 Ziffern + VZ
- Produkt : 1 - 19 Ziffern + VZ

Die Maximalkapazität der Operanden errechnet sich nach der Formel:

$M + m \leqslant 19$

M = Anzahl Stellen Multiplikand

m = Anzahl Stellen Multiplikator

Ansonsten gelten die Regeln des Abschnittes 3.3.1.

Überschreitet das Produkt 19 Ziffern, liegt Kapazitätsüberschreitung vor. Man kann es jedoch nur erkennen, wenn die 20. Ziffer ungleich 0 oder 9 ist. Das Resultat bleibt erhalten, das äußerste linke Byte des Registers 2 wird jedoch in das Register "Kap" (0091) übertragen. Überschreitet das Produkt 20 Stellen, gehen die überzähligen Stellen verloren.

Ist R1 = R2 wird die Zahl, die sich in R1 befindet, quadriert. Diese Zahl kann 9 Stellen nicht übersteigen. R1 kann auch gleich R2 - 1 sein.

Beispiele:

MPD | C1 | 59 | 32 |

Reg. 59 Reg. 31 und 32

a) +x+

vorher : |00|00|00|00|18| |00| ... |00|08|34|79|

nachher : |00|00|00|00|18| |00| ... |01|50|26|22|

0251

(+ 83.479) x (+ 18) = (+ 1.502.622)

b) +x-

vorher : |99|99|99|99|82| |00| .... |00|08|34|79|

nachher : |99|99|99|99|82| |99| .... |98|49|73|78|

(+ 83.479) x (- 18) = (- 1.502.622)

3.3.2.3. Division DVD

Für die Division existiert ein Unterprogramm, das vom Benutzer frei in den Kernspeicher eingesetzt werden kann. Es muß in den Programmkartensatz zwischen KA17 und KA31/32 eingefügt werden.
Die UP-Karten sind aufsteigend von 0001 bis 0014 numeriert, die Reihenfolge darf nicht geändert werden.
Das UP benötigt 249 Kernspeicherstellen. Die Adresse der UP-Divission wird vom Ladeprogramm auf 2 Bytes gespeichert; und zwar bei einer Kernspeichergröße von

2500 Bytes auf die Stellen 2300 und 2301,
5000 " " " " 4800 und 4801,
10000 " " " " 9800 und 9801.

Der Dividend muß im Doppelregister (13) 14, der Divisor im Register 15 gespeichert sein. Das Register 00 muß stets Null sein.

Der Anruf dieses Unterprogramms erfolt automatisch durch den Divisionsbefehl:

DVD DiVide Decimal

| D1 | R_1 | R_2 |

R1 = Einfachregister (Divisor) = R 15
R2 = Doppelregister (Dividend) = RR 14

Der Inhalt des Doppelregisters R2 (RR14) wird durch den Inhalt des Einfachregisters R1 (R15) dividiert.

Ergebnis:
- Der Quotient steht in den 5 linken Bytes (Einfachregister mit der Nummer R2 - 1 = R13)
- Der Rest steht in den 5 rechten Bytes (Einfachregister mit der Nummer R2 = R14)

Nach der Operation ist der Dividend (RR14) verändert; der Divisor (R15) bleibt erhalten.

Wenn das Register R1 ein anderes als das Doppelregister R2 ist, wird sein Inhalt durch die Operation nicht verändert.

Die Division erfolgt nicht algebraisch. Dividend und Divisor müssen positiv sein.

Größe der Faktoren:
- Dividend : 1 - 17 Ziffern + VZ
- Divisor : 1 - 8 Ziffern + VZ
- Quotient : 1 - 9 Ziffern + VZ

Die Maximalgröße der Operanden errechnet sich nach folgender Formel:

$$D - d + 1 \leqslant 9$$

- D = Anzahl Stellen Dividend
- d = Anzahl Stellen Divisor

Ansonsten gelten die Regeln des Abschnittes 3.3.1.

Überschreitet der Quotient 9 Stellen oder ist der Divisor Null, empfängt das Register "Kap" (0091) das Zeichen AA.

Beispiel:

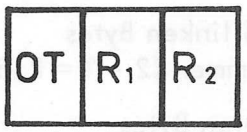
DVD | D1 | 15 | 14 |

	Reg. 15	Reg. 13	Reg. 14
vorher :	00 00 00 05 24	00 00 00 00 00	00 00 20 19 87
nachher:	00 00 00 05 24	00 00 00 03 85	00 00 00 02 47
		Quotient	Rest

(+ 201.987) : (+ 524) = (+ 385)

3.3.2.4. Algebraischer Vergleich

CMD	CoMpare Decimal single length	91
CMDD	CoMpare Decimal Double length	92



OT = 91 oder 92

R1 + R2 = Einfach- oder Doppelregister, nach OT.

Der Inhalt des Registers 1 wird mit dem Inhalt des Registers 2 verglichen. Der Inhalt beider Register wird durch die Operation nicht verändert.

Der Vergleich erfolgt algebraisch. Das Ergebnis wird in dem Vergleichsregister (0086) in Form eines Zeichens gespeichert:

3 C bei R1 < R2
3 D bei R1 = R2
3 E bei R1 > R2

Anmerkung:

Der Inhalt des Vergleichsregisters bleibt bis zum nächsten Vergleich, der den Inhalt verändert, erhalten.

Vor einer Veränderung muß der Inhalt ausgewertet worden sein.

Größe der Operanden:

- CMD : 9 Ziffern + VZ (Register mit 5 Bytes)
- CMDD : 19 Ziffern + VZ (Register mit 10 Bytes)

Beispiele:

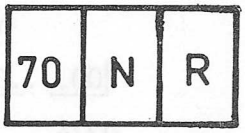
CMD |91|67|59|

	R1 = Reg. 67	R2 = Reg. 59	VG (0086)
a) vorher :	...|00|46|38|	...|00|46|38|	|xx|
nachher :	...|00|46|38|	...|00|46|38|	|3D|
			R1 = R2
b) vorher :	...|00|03|78|	...|99|86|54|	|3D|
nachher :	...|00|03|78|	...|99|86|54|	|3E|
	(+ 378)	(- 1.346)	R1 > R2
c) vorher :	...|99|76|34|	...|99|94|12|	|3E|
nachher :	...|99|76|34|	...|99|94|12|	|3C|
	(- 2.366)	(- 588)	R1 < R2

3.3.3. Versetzungs- und Rundungsbefehle

3.3.3.1. Versetzung nach links

SLD — Shift Left Decimal



N = kann variieren von 00 bis 19

R = Doppelregister

Der Inhalt des Doppelregisters R wird um N Dezimalstellen (Halbbytes) nach links verschoben. Die N ersten Stellen rechts im Register werden nach der Versetzung auf Null gesetzt. Das Vorzeichen des Operanden, der sich im Register R befindet, wird unter der Voraussetzung nicht verändert, daß der Operand beim Versetzen nicht die dafür vorgesehene Stelle erreicht.

Kapazitätsüberschreitung wird nicht angezeigt. Ist die Zahl N der Versetzungen größer als die Zahl N' der zur Verfügung stehenden Stellen links des Operanden, gehen N' - N Ziffern einschließlich des Vorzeichens verloren.

Bei N = 00 findet keine Versetzung statt, der Befehl wird nicht ausgeführt.

Bei N > 19 ist die Zahl der durchgeführten Versetzungen gleich dem Rest der Division von N durch 20.

Beispiel:
N = 77; Zahl der Versetzungen 17.

Das Register kann auch nicht numerische Zeichen enthalten.

Sie werden in der gleichen Weise, Halbbyte für Halbbyte, versetzt.

Anmerkung:

Dieser Befehl ermöglicht die Multiplikation einer Zahl mit 10^n.

Beispiele:

SLD | 70 | 03 | 37 |

vorher:

| 00 | ... | 00 | 00 | 23 | 75 |

0276 0285

nachher:

| 00 ... | 02 | 37 | 50 | 00 |

0276 0285

2.375 ⟶ 2.375.000

vorher:

| 99 | ... | 99 | 99 | 96 | 25 |

0276 0285

nachher:

| 99 ... | 99 | 62 | 50 | 00 |

0276 0285

- 375 ⟶ - 375.000

3.3.3.2. Versetzung nach rechts

SRD — Shift Right Decimal

72	N	R

N = 00 - 19

R = Doppelregister

Der Inhalt des Doppelregisters R wird um N Dezimalstellen (Halbbytes) nach rechts verschoben.

Die Ziffern, die sich an den N ersten Stellen rechts befinden, verschwinden und gehen verloren.

Die Ziffer, die sich an der ersten Stelle von links befindet, bleibt erhalten und wird auf den N folgenden Stellen eingesetzt. Dadurch bleibt die algebraische Form des Registerinhaltes erhalten.

Bei N = 00 findet keine Versetzung statt; der Befehl wird nicht ausgeführt.

Bei N > 19 ist die Zahl der durchgeführten Versetzungen gleich dem Rest der Division von N durch 20.

Beispiel: N = 35; Zahl der Versetzungen 15

Das Register kann auch nicht numerische Zeichen enthalten. Sie werden in der gleichen Weise, Halbbyte für Halbbyte, versetzt.

Anmerkung:

Dieser Befehl ermöglicht eine Division durch 10 unter Vernachlässigung des Restes. Ebenso kann eine Abrundung erfolgen.

Beispiele:

a) SRD |72|02|21|

vorher: nachher:

|00...03|75|45| |00...00|03|75|

0196 0205 0196 0205

375.45 → 375

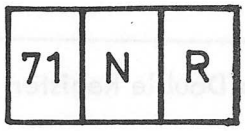
b) SRD |72|14|21|

|99|24|50.....00|00|00| |99|99|99.....99|24|50|

0196 0205 0196 0205

3.3.3.3. Rundungen

ROUND Round



N = 00 - 99

R = Doppelregister

Die Zahl N wird ohne Berücksichtigung des Vorzeichens auf den Inhalt des Doppelregisters R addiert. Anschließend wird das Resultat, das sich in R befindet, um zwei Dezimalstellen (1 Byte) nach rechts versetzt, wobei die zwei rechten Stellen verloren gehen.

Die Operation erfolgt ohne Berücksichtigung des Vorzeichens.

Anmerkung:

Bei N = 00 erhält man eine Division durch 100 unter Vernachlässigung der Dezimalstellen.

Enthält das Register keine numerischen Werte, so wird das nicht angezeigt. Das Resultat ist jedoch falsch.

Das Resultat der Addition darf 19 Stellen nicht überschreiten, damit das Vorzeichen nicht zerstört wird. Kapazitätsüberschreitung wird jedoch nicht angezeigt.

Beispiele:

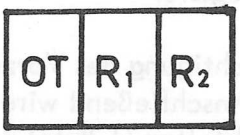
Round |71|50|85|

vorher:	nachher:
a) ...00 07 25	00 00 07 = + 7.25 → 7
b) ...00 07 85	00 00 08 = + 7.85 → 8
c) ...99 92 75	99 99 93 = - 7.25 → 7
d) ...99 92 15	99 99 92 = - 7.85 → 8
0525	0525

3.3.4. Übertragungsbefehle

3.3.4.1. Überträge in einfacher und doppelter Länge

MRR	Move the contents of a single Register into another one	E 1
MDRR	Move the contents of a Double Register into another one	E 2



OT = siehe oben
R 1 = Abgebendes Register (einfach oder doppelt je nach OT)
R 2 = Empfangendes Register (einfach oder doppelt je nach OT)

Der Inhalt des Registers R 1 wird in das Register R 2 übertragen, nachdem dessen Inhalt gelöscht wurde. Der übertragene Wert erfährt keine Veränderung. Handelt es sich um verschiedene Register, wird der Inhalt von R 1 nicht verändert.

Länge:

- MRR : 5 Bytes oder 10 Dezimalziffern (R1 + R2 einfach)
- MDRR : 10 Bytes oder 20 Dezimalziffern (R1 + R2 doppelt)

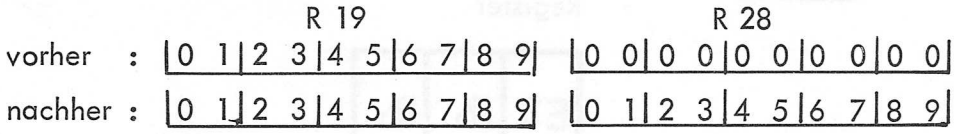
Bei MDRR:

- Wenn $R_1 = R_2 - 1$ erfolgt der Übertrag normal. Das kommt auf das gleiche hinaus, als wenn man den Inhalt des Registers 1 um 10 Dezimalstellen nach rechts verschoben hätte.

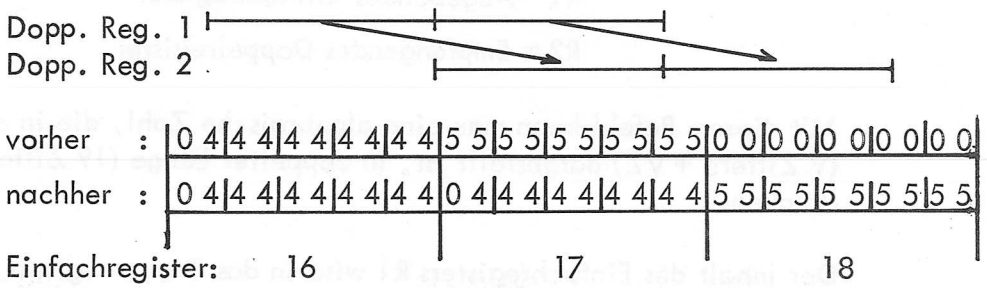
- Wenn $R_1 = R_2 + 1$ erhält man dreimal den Inhalt der 10 Dezimalstellen von rechts des Registers 1 in den Einfachregistern: R1, R2 und R2 - 1.

Beispiele:

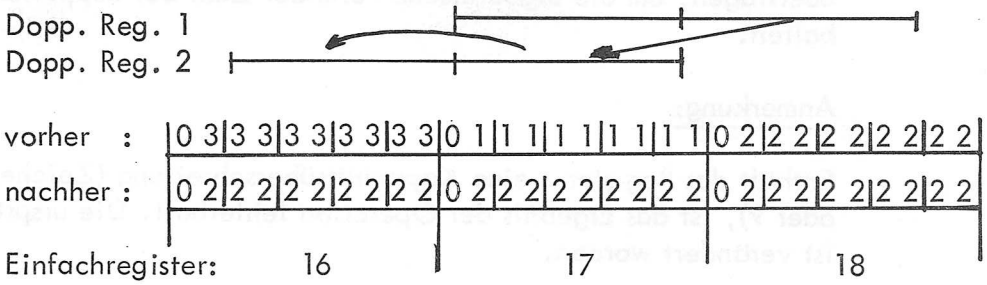
a) MRR |E1|19|28|



b) MDRR |E2|17|18| (R1 = R2 - 1)

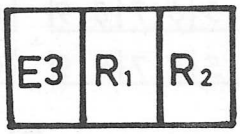


c) MDRR |E2|18|17| (R1 = R2 + 1)



3.3.4.2. Überträge von einfachem noch doppeltem Register

MSRDR — Move the contents of a Single Register into a Double Register



R1 = Abgebendes Einfachregister

R2 = Empfangendes Doppelregister

Mit diesem Befehl kann man eine algebraische Zahl, die in einfacher Länge (9 Ziffern + VZ) dargestellt ist, in doppelter Länge (19 Ziffern + VZ) ausdrücken.

Der Inhalt des Einfachregisters R1 wird in das Doppelregister R2 übertragen, nachdem dessen Inhalt vorher gelöscht wurde. Die Ziffer (0 oder 9), die in R1 das Vorzeichen darstellt, wird 10 mal in das Register 2 nach links übertragen, um die algebraische Form der Zahl bei doppelter Länge zu erhalten.

Anmerkung:

Enthält das Register 1 eine Kapazitätsüberschreitung (Zeichen ungleich 0 oder 9), ist das Ergebnis der Operation fehlerhaft. Die ursprüngliche Zahl ist verändert worden.

Die Operation ist möglich mit R1 = R2 oder mit R1 = R2 - 1. Diese zwei Methoden ermöglichen die "Erweiterung" einer Zahl bei Wiederverwertung des Registers, das sie enthält.

Der Inhalt des Registers R1 wird durch die Operation nicht verändert, außer in dem Fall, wo R1 = R2 - 1 ist.

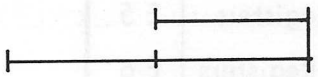
Beispiele:

a) MSRDR |E3|17|83|

	R 17	R 83
vorher :	|0 1|2 3|4 5|6 7|8 9	|9 9|9 9|9 9|9 9|9 9|9 9|9 9|9 9|9 9|9 9|9 9|9 9|
nachher:	|0 1|2 3|4 5|6 7|8 9	|0 0|0 0|0 0|0 0|0 0|0 1|2 3|4 5|6 7|8 9|

Zahl positiv

b) MSRDR | E3 | 17 | 18 |

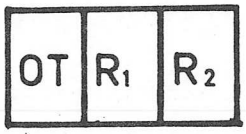


vorher :	0	0	0	0	0	0	0	0	0	0	9	8	7	6	5	4	3	2	1	0
nachher :	9	9	9	9	9	9	9	9	9	9	9	8	7	6	5	4	3	2	1	0

Zahl negativ

3.3.4.3. Registeraustausch

EXR	EXchange single Registers	E 5
EXDR	EXchange Double Registers	E 6



OT = siehe oben
R1, R2 = Einfach- oder Doppelregister je nach OT

Dieser Befehl vollzieht den Austausch des Inhaltes der Register R1 und R2. Am Ende der Operation befindet sich im Register 1 der vorige Inhalt von R2 und in R2 der vorige Inhalt von R1.

Größen:

- bei EXR : 5 Bytes oder 10 Dezimalstellen
- bei EXDR : 10 Bytes oder 20 Dezimalstellen

Bei EXDR ist R1 = R2 - 1 möglich. In diesem Fall erhält man einen rechtsläufigen Ringtausch des Inhaltes der drei Einfachregister.

Beispiele:

a) EXR |E5|17|45|

	R 17									R 45									
vorher :	0	1	2	3	4	5	6	7	8	9	9	8	7	6	5	4	3	2	1	0
nachher:	9	8	7	6	5	4	3	2	1	0	0	1	2	3	4	5	6	7	8	9

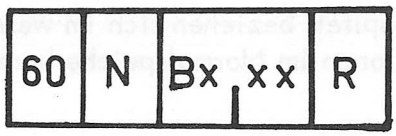
b) EXDR |E6|87|88|

vorher :	0	0	0	0	0	0	0	0	1	4	0	0	0	0	0	0	0	0	2	4	0	0	0	0	0	0	0	0	3	4
nachher :	0	0	0	0	0	0	0	0	3	4	0	0	0	0	0	0	0	0	1	4	0	0	0	0	0	0	0	0	2	4

Einfachregister: 86 87 88

3.3.4.4. Laden eines Registers

LRN	Load Register N



N = 01 - 10
Bxxx = Adresse des zu übertragenden Wertes
R = empfangendes Einfach- oder Doppelregister

Mit diesem Befehl kann man in ein Register R eine Zahl von N Bytes (2N Dezimalziffern) übertragen, die in algebraischer Form gepackt im Normalspeicherbereich an der Adresse Bxxx steht.

Der Übertrag erfolgt in ein Einfachregister, wenn N≤ 5 ist, und in ein Doppelregister, wenn N> 5 ist. Der vorherige Inhalt des Registers wird gelöscht.

Die Zahl wird rechtsbündig in das Register eingesetzt. Wenn nicht alle Stellen des Registers belegt werden (N< 5 oder 10), wird die äußerste linke Ziffer bis zum linken Ende des Registers wiederholt. Diese Ziffer bestimmt das Vorzeichen: Es muß unbedingt 0 oder 9 sein. Anderenfalls ist die Zahl im Register ungleich der ursprünglichen Zahl .

Anmerkung:

Es werden nur die Ziffern der Einer bei N berücksichtigt; die Ziffer 0 wird wie 10 behandelt.

Beispiel:

LRN	60 03 71 52 14	Basis 7 = 2000
	Normalspeicher	Reg. 14
vorher :	2 0 9 5 4 2 1 3 2 0	0 0 0 0 0 0 0 0 0 0
nachher :	2 0 9 5 4 2 1 3 2 0	9 9 9 9 9 5 4 2 1 3

3.4. Operationen auf Zeichenbasis

3.4.1. Allgemeines

Die Befehle dieses Kapitels beziehen sich im wesentlichen auf die Bearbeitung von Informationen im Normalspeicherbereich des Zentralspeichers und zwar sind es:

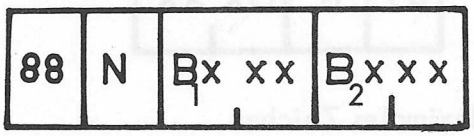
- logische Befehle
- Übertragungsbefehle.

Sie arbeiten in variabler Länge auf 8 Bit-Zeichen. Die Länge der bearbeiteten Worte wird durch die Zahl des Bytes spezifiziert, außer, ein Befehl bearbeitet nur ein Zeichen. Das trifft bei einigen logischen Befehlen zu. Die benutzte Adressierung ist die Basisadressierung.

3.4.2. Logische Befehle

3.4.2.1. Alphanumerischer Vergleich

CMC — CoMpare Characters



N = Die Zahl der zu vergleichenden Bytes von 00 - 99
Bxxx = Adresse der rechten Stelle der zu vergleichenden Worte

Mit diesem Befehl können zwei Worte gleicher Länge N verglichen werden: Das Wort, das sich an B_1xxx befindet, wird verglichen mit dem Wort an B_2xxx. Die beiden Operanden werden durch die Operation nicht verändert. Der Vergleich verarbeitet Zeichen, die sich aus 8 Bits zusammensetzen. Die Worte können daher beliebige Zeichen des internen Codes enthalten.

Das Vergleichergebnis beruht auf der Binärdarstellung des internen Codes und wird im Vergleichregister in Form eines hexadezimalen Zeichens gespeichert.

3 C bei $B_1xxx < B_2xxx$

3 D bei $B_1xxx = B_2xxx$

3 E bei $B_1xxx > B_2xxx$

Bei N = 0 werden beide Worte wie "Nichts" behandelt. Das Register enthält das Ergebnis "Gleich" (3 D).

Beispiel:

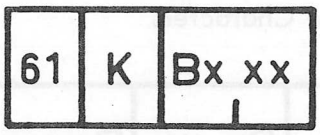
CMC |88|06|6006|7006| Basis 6 = 1500, Basis 7 = 2000

	Zone 6						Zone 7						VG
vorher :	3 0	4 5	3 1	3 8	5 4	4 F	3 4	3 0	4 C	3 0	3 3	4 2	5 6	3 9	00
nachher :	3 0	4 5	3 1	3 8	5 4	4 F	3 4	3 0	4 C	3 0	3 3	4 2	5 6	3 9	3C
	1500				1505			2000					2005		

$B_1xxx < B_2xxx$

3.4.2.2. Logisches UND

NI	aNd	Immediat



K = hexadezimales Zeichen
Bxxx = Adresse des zu bearbeitenden Zeichens

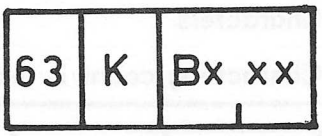
Dieser Befehl setzt nach den Regeln des logischen UND das Zeichen K auf das Zeichen, das sich an Adresse Bxxx befindet.

Beispiel:

NI	61 | 77 | 6004	Basis = 1800
K (Binärdarstellung)	0 1 1 1,0 1 1 1	(77)
Stelle 1804 vorher:	1 1 1 1,1 0 1 1	(FB)
nachher:	0 1 1 1,0 0 1 1	(73)

3.4.2.3. Logisches ODER

ØI	Or	Immediat



K = hexadezimales Zeichen
Bxxx = Adresse des zu bearbeitenden Zeichens

Dieser Befehl setzt nach den Regeln des logischen ODER das Zeichen K auf das Zeichen, das sich an Adresse Bxxx befindet.

Das Ergebnis steht an dieser Adresse zur Verfügung. Das Zeichen, das ursprünglich dort stand, ist verändert worden.

Beispiel:

ØI |63|34|6081| Basis 6 = 2300

K (Binärdarstellung)	0 0 1 1,0 1 0 0	(34)
Stelle 2381 vorher:	1 1 1 0,0 0 0 0	(D0)
nachher:	1 1 1 1,0 1 0 0	(F4)

3.4.3. Übertragungsbefehle im Normalspeicherbereich

3.4.3.1. Einfachüberträge

MVC	MoVe Characters
MVIC	MoVe Characters; count is Indirect

80	N	Bx xx	Bx xx
8A	R	Bx xx	Bx xx

N = Länge des Übertrages in Anzahl Bytes
R = Nummer des Registers, das die Länge N enthält.
Die Länge N kann 00 - 99 sein.
B_1xxx = rechte Adresse der abgebenden Zone
B_2xxx = rechte Adresse der empfangenden Zone

Die Adresse B_2xxx muß ⩾ als Adresse B_1xxx oder ⩽ Adresse B_2xxx - N sein.

Das Wort mit der Länge von N Bytes, das an Adresse B_1xxx steht, wird nach Adresse B_2xxx übertragen.

Der alte Inhalt der empfangenden Zone wird gelöscht. Der Inhalt der abgebenden Zone bleibt unverändert, wenn beide Felder unterschiedliche Adressen haben.

Die Länge des Wortes wird durch den Wert von N bestimmt:

- bei MVC innerhalb des Befehls
- bei MVIC durch den Inhalt des rechten Bytes des Registers R.

Der Übertrag erfolgt Stelle für Stelle von rechts nach links.

Anmerkung:

Befindet sich die Adresse B_2xxx innerhalb der Adressen B_1xxx - N und B_1xxx, werden nur die Stellen zwischen den Adressen B_2xxx und B_1xxx richtig übertragen; sie wiederholen sich aber innerhalb des Empfangsfeldes (vgl. Beispiel -b)).

Beispiele:

a) Normaler Übertrag: MVC |80|08|1008|1010|

Basis 1 = 2000

vorher : |3| |P|A|R|I|S| |2|4|5|3|

nachher: |3| |P| |P|A|R|I|S| |5|3|

2000 2005 2010

b) Unkorrekter Übertrag: MVIC |8A|06|1008|1006|

Basis 1 = 2000; Register 06 enthält: 6

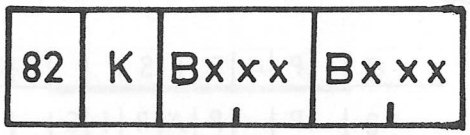
vorher : 8 |4|3| |P|A|R|I|S| | |2|4|5|3|

nachher: 8 |I|S|I|S|I|S|I|S| | |2|4|5|3|

2000 2005 2010

3.4.3.2. Einsetzen eines Zeichens

INC IN sert Character



K = einzusetzendes Zeichen
B_1xxx = Adresse der 1. Stelle, die belegt werden soll (rechte Adresse)
B_2xxx = Adresse der letzten Stelle - 1, die belegt werden soll (linke Adresse)

Die Adresse B_1xxx muß > B_2xxx sein.

Das Zeichen K wird auf jede Stelle einer Speicherzone eingesetzt, die durch die beiden Adressen begrenzt wird. Das Einsetzen erfolgt von rechts nach links von Stelle B_1xxx einschließlich bis Stelle B_2xxx ausschließlich.

Wichtig:

Ist die Adresse B_1xxx ≤ Adresse B_2xxx, erfolgt die Belegung bis Stelle 0000 des Speichers und vom Ende des Speichers bis Adresse B_2xxx. Der Inhalt der speziellen Zonen, der Register und des Programmes werden zerstört.

Beispiel:

INC |82|20|7117|7108| Basis 7 = 1200

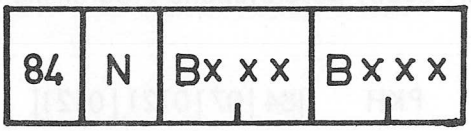
vorher : |39|39|37|36|35|34|33|32|31|30|20|50|5D|4D|

nachher : |39|38|37|20|20|20|20|20|20|20|20|20|5D|4D|

1306 1310 1315

3.4.3.3. Packen hexadezimal

PKH Pack Hexadezimal



N = Zahl der zu packenden Bytes von 00-99
B_1xxx = rechte Adresse der abgebenden Zone
B_2xxx = rechte Adresse der empfangenden Zone

Die Adresse B_2xxx muß ≥ als B_1xxx oder ≤ B_1xxx - N sein.

Dieser Befehl packt auf N Halbbytes ein Wort von N Bytes, das im ISO-Code ausgedrückt ist und nur die folgenden Werte enthält: Leerstelle, 0 - 9 und A - F. Die Umwandlung von Byte auf Halbbyte ist in der Tabelle dargestellt:

Zeichen	Leer-stelle	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ISO-Code (Byte)	20	30	31	32	33	34	35	36	37	38	39	41	42	43	44	45	46
gepackt (Halb-byte)	0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Jede andere Kombination des ISO-Codes in den ursprünglichen Bytes gibt ein nicht auswertbares, sinnloses Ergebnis.

Das Wort von N Bytes, das gepackt werden soll, steht an Adresse B_1xxx. Es wird in gepackter Form nach Adresse B_2xxx übertragen und belegt dort $\frac{N}{2}$ Bytes. Ist N ungerade, wird auf volle Bytes aufgerundet. In diesem Fall enthält das äußerste linke Halbbyte Null.

Der alte Inhalt der empfangenden Zone wird gelöscht. Bei unterschiedlichen Feldern bleibt der Inhalt des abgebenden Feldes unverändert. Die Operation packt Stelle für Stelle von rechts nach links.

Anmerkung:

Befindet sich die Adresse B_2xxx innerhalb der Adressen B_1xxx - N und B_1xxx, ist das Ergebnis nicht voraussehbar und hat keinerlei Wert.

Der Befehl PKH dient nicht dazu, Zahlen, mit denen gerechnet werden soll, von ungepackter in gepackte Form zu bringen. Diese Aufgabe wird von den Mehrfachübertragungsbefehlen übernommen. (s. 3.5.) Im wesentlichen dient PKH dazu, geschriebene Programme durch das Ladeprogramm zu verdichten.

Beispiel: PKH |84|07|0121|0121| Basis 0 = 2500

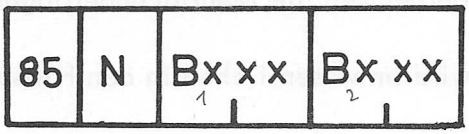
vorher : |45|42|30|46|41|32|31|20|20|

nachher: |45|42|30|0E|B0|FA|21|20|20|

2621

3.4.3.4. Entpacken hexadezimal

UPH Un Pack Hexadezimal



N = Zahl der zu erziehlenden Bytes von 00 - 99
B_1xxx = rechte Adresse der abgebenden Zone
B_2xxx = rechte Adresse der empfangenden Zone

Die Adresse B_2xxx muß:

$\geqslant B_1xxx + \frac{N}{2}$ oder $\leqslant B_1xxx - \frac{N}{2}$ sein.

Ist N ungerade, wird $\frac{N}{2}$ auf volle Bytes aufgerundet.

Dieser Befehl drückt im internen ISØ-Code ein Wort auf N Bytes aus, das auf N Halbbytes gespeichert ist. Die 16 hexadezimalen Werte, die jedes Halbbyte darstellen kann, ergeben folgende Zeichen ISØ-Codes:

Inhalt des Halbbytes	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F		
ISØ-Code (1 Byte)	30	31	32	33	34	35	36	37	38	39	41	42	43	44	45	46		
Zeichen	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F		

Das an Adresse B_1xxx stehende, gepackte Wort wird in den ISØ-Code übersetzt und auf N Bytes an Adresse B_2xxx übertragen.

Die abgebende Zone enthält $\frac{N}{2}$ Stellen, aufgerundet auf volle Stellen, wenn N ungerade ist. In diesem Fall wird das äußerste linke Halbbyte dieser Zone nicht bearbeitet.

Der alte Inhalt der empfangenden Zone wird gelöcht. Bei verschiedenen Feldern bleibt der Inhalt der abgebenden Zone unverändert. Die Operation entpackt Stelle für Stelle von rechts nach links.

Anmerkung:

Befindet sich die Adresse B_2xxx zwischen den Adressen $B_1xxx + \frac{N}{2}$ und $B_1xxx - \frac{N}{2}$ ist das Ergebnis nicht vorhersehbar und ohne Bedeutung.

Dieser Befehl dient nicht dazu, algebraische Zahlen aus gepackter in ungepackte Darstellung zu bringen.

Diese Aufgabe übernehmen die Mehrfachübertragungsbefehle (s. 3.5.).

Der Befehl UPH wird im wesentlichen in den Hilfsprogrammen benutzt.

Beispiel:

UPH	|85|05| 0150| 0153|	Basis 0 = 1000
vorher :	|AB|4C|0B|1E|20|20|20|A1|	
nachher:	|AB|4C|43|30|42|31|45|A1|	
	1150	

3.5. Mehrfachüberträge

3.5.1. Darstellung

Die Mehrfachübertragungsbefehle gestatten es, ein Wort variabler Länge von einer Zone nach 1,2 oder 3 anderen Zonen gleichzeitig zu übertragen.

Die abgebende Stelle ist entweder ein numerisches Register oder eine von zwei Normalspeicherzonen; im allgemeinen Eingabezonen von Randeinheiten.

Die empfangenden Stellen sind: ein numerisches Register und/oder eine oder 2 Normalspeicherzonen; im allgemeinen Ausgabezonen für Randeinheiten.

Im Befehl kann nur ein Register entweder als abgebend oder als empfangend angegeben werden. Der Befehl erkennt die Normalspeicherzonen an den Adressen, die in den Basisregistern 6 - 9 stehen. Die Basisregister 6 und 7 entsprechen immer abgebenden Zonen; die Basisregister 8 und 9 immer empfangenden Zonen. Durch Vereinbarung und zur Unterscheidung bezeichnet man die Zonen mit der Nummer des ihnen zugeordneten Basisregisters.

Die verschiedenen Kombinationen der zu erzielenden Überträge, die durch Variation der abgebenden und empfangenden Zonen entstehen, ergeben eine Anzahl Befehle, die in Abschnitt 3.5.4 zusammengestellt sind.

Die Abschnitte 3.5.2 und 3.5.3 beschreiben die gemeinsamen Eigenschaften.

Drei Befehle geben die Möglichkeit, ein Zeichen in eine Zahl, die nach Zone 8 übertragen wurde, einzusetzen. Sie sind beschrieben in Abschnitt 3.5.5.

3.5.2. Allgemeiner Aufbau und Darstellung

Symbolischer Operationstyp

Jedem Mehrfachübertragungsbefehl entspricht ein symbolischer Schlüssel der sich zusammensetzt aus dem Buchstaben M und 2, 3 oder 4 Zeichen, die von links nach rechts bedeuten:

die abgebende Stelle und die empfangende(n) Stelle(n). Die empfangenden Zonen stehen in folgender Reihenfolge:

numerisches Register, Zone 8, Zone 9.

Der Buchstabe M (Move) symbolisiert die Übertragungsoperation. Die abgebenden und empfangenden Zonen werden bezeichnet:

- durch den Buchstaben R für ein numerisches Register
- durch die Zahl (6, 7, 8 oder 9) des Basisregisters der entsprechenden Normalspeicherzone.

Beispiel:

Abgebend	Empfangend	Symbolischer OT
Zone 6	Reg., Zonen 8 + 9	M 6 R 8 9
Register	Zonen 8 + 9	M R 8 9
Zone 7	Zone 9	M 7 9

Das so gebildete Symbol kann noch ergänzt werden durch den Buchstaben G (Graphic), der anzeigt, daß es sich um einen Befehl handelt, der ein Zeichen in das zu übertragende Wort einsetzt.

Beispiel: M R 8 G

Format der Befehle

Die Länge eines Befehles hängt von der Anzahl der empfangenden Felder ab. Der Befehl enthält:

- den Operationstyp (OT)
- die Zahl N der zu übertragenden Stellen
- die Adresse des abgebenden Feldes
- die Adressen der empfangenden Felder in folgender Reihenfolge: Numerisches Register, Zone 8, Zone 9.

Theoretischer Aufbau:

O T	N	Adresse d. abgeb. Feldes	Adresse d. 1. empf. Feld. (Register)	Adresse d. 2. empf. Feld. (Zone 8)	Adresse d. 3. empf. Feldes (Zone 9)

Betrifft der Befehl nur ein oder zwei empfangende Felder, werden die entsprechenden Adressen nach links verschoben.

Operationsschlüssel (OT)

Der Operationsschlüssel in Maschinensprache hängt von den abgebenden und den empfangenden Feldern ab. Er berücksichtigt auch das eventuelle Einsetzen eines Zeichens. Die verschiedenen Operationstypen, die möglich sind, stehen in den Abschnitten 3.5.4 und 3.5.5.

Länge N des Übertrages

Wenn der Befehl kein Register benutzt, kann die Länge N von 01 bis 99 zu übertragende Stellen betragen.

Wenn ein numerisches Register benutzt wird, wird die Länge N durch die Kapazität des Registers begrenzt.
Sie wird in Dezimalziffern dargestellt und beträgt:

- 01 - 09 bei einem Einfachregister
- 01 - 19 bei einem Doppelregister.

Wenn das Register als abgebende Stelle dient, wird die Länge N von der ersten Dezimalstelle rechts an gezählt.

Adressierung

Die Register werden normal mit ihrer Nummer adressiert. Die Adressierung der Normalspeicherzonen wurde vereinfacht:

Der Programmierer braucht nur die Verschiebungen der in Betracht kommenden Worte innerhalb der interessierenden Zonen in Betracht zu ziehen. Die zu jeder Zone gehörenden Basisregister werden automatisch durch die Entschlüsselung des Operationstyps bestimmt.

Dieses System, vereinfachte Basisadressierung genannt, funktioniert nur korrekt, wenn die Adressen (Registernummer und/oder Verschiebungen) im Befehl in der durch den Befehlsaufbau vorgesehenen Form von links nach rechts stehen. An diese Reihenfolge wird man durch den symbolischen Operationstyp erinnert.

Die Verschiebungen werden in folgender Form dargestellt:

- zwei Ziffern, wenn $\leqslant$ 99 : xx
- drei Ziffern, denen der Buchstabe A vorangestellt ist, wenn > 99 : Axxx.

Sie belegen 1 oder 2 Bytes und werden im Befehl mit dem Buchstaben D bezeichnet, dem die Nummer der betreffenden Zone zugefügt wird.

Beispiele:

M6R89

27	09	38	16	A114	24
OT	N	D6	R	D8	D9

MR89

16	12	42	A108	72
OT	N	R	D8	D9

3.5.3.3. Wirkung der Befehle

Die Befehle der Mehrfachüberträge beachten die allgemeinen Regeln der Überträge im Zentralspeicher.

Jeder Übertrag vom abgebenden Feld nach dem entsprechenden empfangenden Feld erfolgt Zeichen für Zeichen von rechts nach links. Der alte Inhalt des empfangenden Feldes wird gelöscht. Der Inhalt des abgebenden Feldes bleibt erhalten.

Der Übertrag bringt das übertragene Wort in die richtige Form, wenn ein Register abgibt oder aufnimmt.

Übertrag einer Zone nach einem Register

Die Zahl, die ungepackt in dem abgebenden Feld steht, wird in das Register gepackt übertragen. Der Übertrag erfolgt in algebraischer Form, abhängig vom Inhalt des Vorzeichenspeichers, der unter Umständen vor dem Übertrag verändert werden muß. Durch die Operation wird der Vorzeichenspeicher gelöscht.

Übertrag Register nach Zone

Die algebraische Zahl, die gepackt im Register steht, wird komplementiert, falls sie negativ ist und anschließend in ungepackter Form in die empfangende Zone übertragen. Nullen links werden durch Leerzeichen (20) bis zur Länge N ersetzt. Der Vorzeichenspeicher wird abhängig vom Vorzeichen der Zahl geladen: 20 bei positivem oder keinem, 2D bei negativem Inhalt.

Überträge Zone nach Zone

Das übertragene Wort erfährt keinerlei Veränderung. Handelt es sich um eine ungepackte Zahl, die durch Nullen links ergänzt ist, bleiben diese erhalten.

Anmerkung:

Ist N = 00, findet keine Übertrag statt. Ist die abgebende Stelle ein Register, wird das Vorzeichen der Zahl, die in ihm enthalten ist, in den Vorzeichenspeicher übertragen.

3.5.4. Mehrfachüberträge (ohne Einsetzen eines Zeichens)

3.5.4.1. Die Befehle

von	nach R	nach 8	nach 9	OT symb.	Befehle	Länge
R		X		MR8	12 N R D8	4/5
			X	MR9	14 N R D9	4/5
		X	X	MR89	16 N R D8 D9	5/7
Zone 6	X			M6R	21 N D6 R	4/5
	X	X		M6R8	23 N D6 R D8	5/7
	X		X	M6R9	25 N D6 R D9	5/7
	X	X	X	M6R89	27 N D6 R D8 D9	6/9
		X		M68	22 N D6 D8	4/6
			X	M69	24 N D6 D9	4/6
		X	X	M689	26 N D6 D8 D9	5/8
Zone 7	X			M7R	31 N D7 R	4/5
	X	X		M7R8	33 N D7 R D8	5/7
	X		X	M7R9	35 N D7 R D9	5/7
	X	X	X	M7R89	37 N D7 R D8 D9	6/9
		X		M78	32 N D7 D8	4/6
			X	M79	34 N D7 D9	4/6
		X	X	M789	36 N D7 D8 D9	5/8

L = Länge des Befehles in Bytes (min. und max. je nachdem, ob die Verschiebungen 1 oder 2 Bytes belegen).

3.5.4.2. Beispiele:

A Überträge von einem Register nach einer Zone (MR8, MR9, MR89)

a) MR 8 |12|06|10|12| Basis 8 = 2100

	Reg. 10	VZ	Zone 8
vorher :	00 00 00 21 14	00	42 34 33 46 54 20 59 54
nachher :	00 00 00 21 14	20	42 20 20 32 31 31 34 54

b) MR 9 |14|05|13|24| Basis 9 = 2175

	Reg. 13	VZ	Zone 9
vorher :	99 99 99 94 37	00	43 58 37 38 20 46 45 52
nachher :	99 99 99 94 37	2D	43 58 20 20 35 36 33 52

c) MR 89 |16|07|12|A112| 58 Basis 8 = 2038
Basis 9 = 2175

	vorher:	nachher:
VZ	20	20
Reg. 12	00 00 05 48 32	00 00 05 48 32
Zone 8	20 20 20 20 20 20 20	20 20 35 34 38 33 32
Zone 9	41 42 43 44 45 46 47	20 20 35 34 38 33 32

B Überträge einer Zone nach Register (M6R, M7R)

a) M6R |21|05|32|15| Basis 6 = 2257

	Zone 6	VZ	Reg. 15
vorher :	20 20 32 34 39	20	99 99 99 83 14
nachher :	20 20 32 34 39	00	00 00 00 02 49
			+ 249

b) M7R | 31 | 10 | 34 | 14 | Basis 7 = 1780

Vorzeichenspeicher	vorher:	nachher:
	2D	00

Zone 7	20	20	20	34	33	38	35	32	31	39
Reg. 13/14	00	00	09	43	83	00	00	00	28	34
	99	99	99	99	99	95	61	47	81	

(- 4.385.219)

C Überträge von einer Zone nach einem Register und einer Zone (M6R8, M6R9, M6R89, M7R8, M7R9, M7R89)

M6R8 | 23 | 06 | 12 | 44 | A1 | 24 | Basis 6 = 2257
Basis 8 = 2038

	vorher:							nachher:						
VZ	2D							00						
Zone 6	42	20	30	38	33	30	30	46	42	20	30	38	33	30	30	46
Zone 8	20	20	20	20	20	20	20	20	20	20	30	38	33	30	30	20
Reg. 44	00	98	04	00	17				99	99	99	17	00			

(- 8.300)

D Überträge von Zone nach Zone (M68, M69, M689, M78, M79, M789)

M789 | 36 | 05 | 32 | 12 | 80 | Basis 7 = 2257
Basis 8 = 2038
Basis 9 = 2175

	vorher:							nachher:						
Zone 7	44	41	39	34	54	30	20	44	41	39	34	54	30	20
Zone 8	20	20	20	20	20	20	20	20	41	39	34	54	30	20
Zone 9	39	46	30	36	48	32	45	39	41	39	34	54	30	45

3.5.5. Überträge mit Einsetzen eines Zeichens

Dieser Befehl ist nur in Verbindung mit einer empfangenden Zone anwendbar: nämlich mit der Zone 8. Die abgebende Zone dagegen kann verschieden sein. Daraus ergeben sich 3 Befehle:

Abgebendes Feld	OT symb.	Befehle in Maschinensprache	
Reg.	MR8G	1E N R D8 KN'	(5/6 Bytes)
Zone 6	M68G	2E N D6 D8 KN'	(5/7 Bytes)
Zone 7	M78G	3E N D7 D8 KN'	(5/7 Bytes)

K = eine Ziffer von 0 - 9 oder ein Buchstabe A - E. Der Gebrauch von F ist verboten.

N' = Zahl der Stellen rechts von dem eingesetzten Zeichen. N' kann variieren von 1 - 9.

Diese Befehle dienen grundsätzlich zur Druckaufbereitung von Zahlen, indem sie die ganzen Zahlen von den Dezimalstellen durch eine Interpunktion (Komma oder Punkt beispielsweise) trennen.

Sie haben die gleichen Aufgaben wir die Befehle MR8, M68, M78 mit folgenden Besonderheiten:

- ein Zeichen wird in die Zahl eingesetzt, die nach Zone 8 übertragen wird.
- die Nullen, an der ersten Stelle links dieses Zeichens und alle Nullen rechts bleiben erhalten.
- die Zahl der Stellen rechts des Zeichens wird durch die Zahl N' bestimmt. Der Rest der Zahl wird um eine Stelle nach links versetzt. Am Ende der Operation belegt die Zahl N + 1 Stelle von Basisadresse D8 an.
- das einzusetzende Zeichen wird aus den Werten 20 - 2E genommen. Sein linkes Halbbyte, immer 2, wird durch die Maschine ergänzt. Sein rechtes Halbbyte wird durch die Zahl K bestimmt.

Beispiele:

a) M68G |2E|05|34|58|C2| Basis 6 = 2257, Basis 8 = 2038

	Zone 6	Zone 8
vorher :	|45|30|34|38|32|30|39|	|20|32|35|33|30|32|36|20|
nachher :	|45|30|34|38|32|30|39|	|20|30|34|38|2C|32|30|20|
	4820 ⟶	48,20

b) MR8G |1E|06|24|A1|12|C2| Basis 8 = 2038

	Reg. 24	VZ	Zone 8
vorher :	|00|00|00|00|05|	|00|	|20|44|52|43|56|58|41|49|
nachher:	|00|00|00|00|05|	|20|	|20|20|20|20|30|2C|30|35|

0000000005 ⟶ 0,05

3.6. Ein-/Ausgabeoperationen

3.6.0. Die Zone IØC

Sie zerfällt in 2 große Bereiche:

- Eingabe - Ausgabe
- Kennzeichnung der Verbindung

Der Bereich "Eingabe - Ausgabe" umfaßt 6 Gruppen zu je 4 Bytes. Jede der Gruppen kann einer Randeinheit zugeordnet sein. Ein IØC-Befehl belegt soviele Gruppen, wie er Randeinheiten verbindet.

Die Belegung dieses Bereiches wird im Augenblick der Aufnahme eines IØC durchgeführt, und zwar von links nach rechts.

Die Anzahl der gleichzeitig in diesem Bereich verbundenen Randeinheiten ist auf 6 begrenzt. Der Rechner berücksichtigt diesen Umstand.

Der Bereich wird von links nach rechts abgetastet. Während dieses Vorganges werden die einzelnen Randeinheiten entsprechend ihrer Verfügbarkeit in Arbeit gesetzt oder nicht. Wenn eine Gruppe entweder nichts oder die Elemente eines durchgeführten IØC enthält, wird sie nicht mehr abgetastet.

Der Bereich "Kennzeichnung der Verbindung" umfaßt 5 Bytes, die den Programmen 1 - 5 entsprechen.

Jeweils das rechte Halbbyte enthält die Anzahl der Randeinheiten, deren Arbeit noch nicht beendet ist. Bei gleitenden IØC-Befehlen können weitere IØC desselben Programms folgen. In diesem Fall wird eine Addition der Anzahl der durch einen jeden IOC angesprochenen Randeinheiten in diesem Halbbyte vorgenommen.

Das jeweils linke Halbbyte enthält:

- 8 bei einem blockierenden IØC
- 9 bei einem gleitenden IØC

3.6.1. Rolle und Aufgabe des Ein-/Ausgabebefehls

Jeder Informationsaustausch zwischen der Zentraleinheit und den verschiedenen Randeinheiten, die angeschlossen sind, wird durch einen einzigen Befehl vorgenommen: den Ein-/Ausgabebefehl.

Dieser Befehl, der in zwei Versionen existiert, steuert den Ablauf der Randeinheiten, indem er:

- entweder in den Zentralspeicher die gelesenen Daten zur Verarbeitung eingibt, die vom Kartenleser, dem Pufferspeicher oder der Trommel herstammen;

- oder Resultate nach durchgeführten Bearbeitungen durch die Zentraleinheit zum "Schreiben" auf externe Randeinheiten (z.B. Karten, Listen, Trommel etc.) bereitstellt;

- oder bestimmte Funktionen der Randeinheiten, wie z.B. Auswahl der Bahn auf der Trommel oder Papiervorschub auf dem Drucker etc. steuert.

Diese Funktionen sind in den Kapiteln beschrieben, die sich auf die Randeinheiten beziehen. Die allgemeinen Richtlinien, um diese Funktionen anzuwenden, befinden sich in Abschnitt 3.6.5.

Ein einziger Befehl kann mehrere Operationen (max. 6) unter den Bedingungen, die in Abschnitt 3.6.3 erwähnt sind, durchführen.

Der Ein-/Ausgabebefehl enthält:

- einen Operationstyp (ØT),
- die Zahl N der angerufenen Operationen,
- die entsprechenden Parameter für jede angerufene Operation.

Eine Operation, die sich auf eine Randeinheit bezieht, wird definiert durch zwei Parameter: den Anschlußcode und die Nummer des Basisregisters.

Der Anschlußcode

- identifiziert die angesprochene Randeinheit:
  - langsame oder schnelle Einheit;
  - Nummer des Kanals
  - Nummer der Einheit

- legt die durchzuführende Operation fest:

  - Lesen,
  - Schreiben,
  - besondere Anweisungen (bei Trommel).

Er setzt sich aus 8 Bits zusammen und wird in der Form von zwei hexadezimalen Zeichen dargestellt.

Die Basisregister

geben die linke Adresse - 1 der Ein- oder Ausgabezone im Zentralspeicher an.

3.6.2. Organisation einer Ein-/Ausgabeoperation

Jeder angeschlossenen Randeinheit ist im Zentralspeicher eine Zone für die Ein- oder Ausgabe zugeordnet. Ihre Länge wird bestimmt durch die maximale Anzahl der im Laufe der Operation zu übertragenen Zeichen zuzüglich eventueller Hilfsschlüssel (s. 3.6.5.) und einer Trennmarke.

Der Übertrag einer Information erfolgt in reeller Länge.

Er beginnt:

- bei der Randeinheit an der ersten angesprochenen Stelle (normalerweise Stelle 1)
- im Zentralspeicher an der Stelle + 1, die als Basisadresse im angesprochenen Register steht.

Die Operation läuft im Speicher von links nach rechts bis zu einer Trennmarke ab, die anzeigt, daß das letzte Zeichen übertragen wurde.

Diese Trennmarke, dargestellt durch das Zeichen "F4" begrenzt eine Information nach folgenden Richtlinien:

a) Informationsausgabe aus dem Zentralspeicher

Die Trennmarke wird in der Ausgabezone des Zentralspeichers rechts vom letzten zu übertragenden Zeichen gesetzt. Bezieht sich die Ausgabe auf die Trommel, wird diese Trennmarke mitübertragen. In allen anderen Fällen zeigt sie lediglich das Ende des Übertrages an, ohne übertragen zu werden.

b) Informationsausgabe von der Trommel

Die Trennmarke, die vorher mit der Information auf die Trommel geschrieben wurde, zeigt das Ende des Übertrages an. Sie wird mit der Information in den Zentralspeicher übertragen.

c) Informationsausgabe von einer anderen Randeinheit

Da die Trennmarke von den Randeinheiten nicht übertragen wird, muß sie vorher im Zentralspeicher in der betreffenden Zone gesetzt werden. Sie belegt die erste Stelle rechts nach den für die Information vorgesehenen Stellen. Die Zentraleinheit analysiert den Inhalt der Stellen, die im Laufe des Übertrages einen Wert erhalten haben. Die Analyse der Trennmarke beendet die Operation.

Es ist Aufgabe des Programmierers, darauf zu achten, daß die Trennmarken an der richtigen Stelle stehen.

Die Trennmarke darf nicht an der Stelle + 1 stehen, die durch das Basisregister angegeben ist, weil sie dort nicht erkannt werden kann. Das bedeutet, daß die zu übertragende Information wenigstens eine Stelle umfassen muß.

Übertrag der Information

Die Randeinheiten stehen mit der Zentraleinheit über Kanäle in Verbindung. Jeder Kanal mündet in der Zentraleinheit in ein einstelliges Pufferregister, durch das die Information übertragen wird. Bevor die Operation gestartet wird, untersucht die Zentraleinheit, ob die angesprochene Randeinheit frei ist. Ist das der Fall, wird die Operation gestartet und läuft in folgender Weise ab:

a) Eingabe

Die Randeinheit überträgt die gelesenen Zeichen nacheinander in das Pufferregister. Anschließend erfolgt der Übertrag in die Eingabezone.

b) Ausgabe

Die Zentraleinheit entnimmt der Ausgabezone Zeichen und überträgt sie nacheinander in das Pufferregister unter der Voraussetzung, daß dieses frei ist.

Das erfolgt jedesmal, wenn die Randeinheit ein Zeichen übernommen hat, das vorher zur Ausgabe bereitgestellt wurde.

Bei jedem Übertrag eines Zeichens rechnet die Zentraleinheit die laufende Adresse der Ein- oder Ausgabezone weiter.

Anmerkung:

Die Dauer der Operation wird durch die langsamste Randeinheit bestimmt.

Ist eine Randeinheit nicht frei, wartet die Operation solange, bis der Anschluß möglich ist.

3.6.3. Simultanbefehle mehrerer Operationen (Horizontale Simultaneität)

Ein einziger Befehl kann mehrere Randeinheiten ansprechen unter der Bedingung, daß diese sich an verschiedenen Kanälen befinden. So kann z.B. der Kartenstanzer (Kanal 2) und der Drucker (Kanal 3) gemeinsam arbeiten; dagegen müssen der Kartenleser und die numerische Tastatur durch verschiedene Befehle angesprochen werden, weil sie sich an einem gemeinsamen Kanal (1) befinden.

Gegenseitige Blockierung von Randeinheiten am gleichen Kanal

Zwischen Randeinheiten, die an denselben Kanal angeschlossen sind, ist keine Simultaneität möglich.

Beispiel: IØC Leser - num. Tastatur - Alphatastatur

Während des Abtastens der IØC-Zone wird in diesem Fall nur die erste erkannte langsame Randeinheit bedient. Die Schaltkreise des GE-55 halten die Tatsache fest, daß diese eine Randeinheit verbunden ist und alle anderen langsamen Randeinheiten am gleichen Kanal noch zu bedienen sind.

Beispiel: 53 11 94 R
simultanes Programm
53 01 9A 12

Die Alphatastatur ist während der Kernspeicherarbeit simultan verbunden. Die numerische Tastatur wird erst entriegelt, wenn die Zone für die Alphatastatur gefüllt worden ist.

Ist beim Multiprogramming ein Programm in der Zone PRC in Wartestellung, führt der IØC für die numerische Tastatur einen Programmwechsel herbei. Dieses Programm läuft simultan mit der Verbindung der Alphatastatur ab. Ist die Arbeit an der Alphatastatur beendet, kommt es zu einer Simultaneität mit der numerischen Tastatur.

Ein Befehl ist abgeschlossen, wenn alle befohlenen Operationen durchgeführt worden sind.

Die Operationen werden in der Reihenfolge gestartet, in der die Einheiten zur Verfügung stehen. Die langsamen Randeinheiten können simultan arbeiten (s. 2.1.1.). Die Zentraleinheit löst, soweit es möglich ist, die Operationen parallel aus und kontrolliert den Ablauf Zeichen für Zeichen.

Dagegen erlaubt der Anruf der Trommel der Zentraleinheit nicht, mehrere Operationen simultan auszuführen, von denen eine diese Einheit betrifft. Diese wird vorrangig gestartet, sobald die Trommel zur Verfügung steht und läuft allein ab; die anderen Operationen werden erst ausgeführt, wenn diese vollständig durchgeführt wurde.

Kontrolle der Operationen

Die Zentraleinheit speichert die Ein-/Ausgabebefehle und führt die Operationen in der IØC-Zone aus, die im einzelnen enthält:

- die erforderlichen Angaben, spez. die Anschlußcodes, die durch den Befehl gegeben werden.
- die Informationen für den Ablauf der Operationen (lfd. Adressen der Zonen, Zahl der angeschalteten Randeinheiten etc.)

Nach jeder Beendigung einer Operation werden die entsprechenden Angaben in der IØC-Zone gelöscht und die Zahl der angeschalteten Randeinheiten wird um eine vermindert. Sobald diese Zahl Null erreicht, ist der Befehl beendet.

Wenn einer der Ein-/Ausgabebefehle, die gleichzeitig in der IØC-Zone gespeichert sind, den Kartenleser betrifft, dürfen es nur maximal 5 Befehle sein und zwar:

- höchstens 3 bei normalen Randeinheiten (einschl. Kartenleser),
- höchstens 2 bei schnellen Randeinheiten (z.B. Trommel).

Daraus ergibt sich in diesen Fällen:

- ein einziger Ein-/Ausgabebefehl kann nicht mehr als 5 Operationen in der oben beschriebenen Weise bearbeiten.
- mehrere Ein-/Ausgabebefehle, die diese Konditionen nicht beachten können in Verbindung mit Multiprogramming nicht in der IØC-Zone eingespeichert werden. Es ist erforderlich, die notwendigen Schutzmaßnahmen vorzusehen (s. Kap. 4).

3.6.4. Verhalten bei Störungen

Jede normale (leeres Fach bei Leser und Stanzer, Papierende am Drucker etc.) oder anormale (gestörte Schreib- oder Leseoperation etc.) Störung an einer Randeinheit bewirkt, daß diese Einheit nicht ansprechbar ist. Eine entsprechende Operation kann nicht gestartet werden. Ein Ein-/Ausgabebefehl, der diese Einheit anspricht, wird in der IØC-Zone solange in Wartestellung gesetzt, bis die Störung beseitigt ist.

Aus diesem Grunde testet die Zentraleinheit vor jeder Ein-/Ausgabeoperation die angesprochene Randeinheit auf Bereitschaft. Ist das nicht der Fall, wird die Operation nicht gestartet und das Programm blockiert auf dem Befehl, weil es nicht beendet werden kann.

Der Bedienung wird das angezeigt durch Aufleuchten der entsprechenden Lampe am Bedienungspult. Sie hat daraufhin den Fehler zu beseitigen und die Randeinheit wieder betriebsbereit zu machen.

Wenn die Störungsanzeige aufgrund eines Lese- oder Schreibfehlers bei Trommelüberträgen erfolgte, oder wenn ein Stanzfehler durch die Kontrolleinrichtung des Stanzers angezeigt wurde, kann die Bedienung das Programm unterbrechen und ein "Fehlerprogramm" starten, das eingespeichert ist, um diese Störung zu beseitigen. Sie drückt hierzu auf die Taste "PI" (Program Interrupt) bevor sie die Randeinheit wieder einschaltet.

Nach Durchführung dieser Manipulationen und wiederhergestellter Arbeitsbereitschaft der Randeinheit startet die Zentraleinheit die in der IØC-Zone wartende Operation, führt sie aus und beendet den Ein-/Ausgabebefehl, der blockiert worden war. Infolgedessen und je nachdem das Fehlerprogramm gestartet wurde oder nicht, springt das laufende Programm auf das Fehlerprogramm oder fährt in seinem normalen Ablauf fort.

Schlußfolgerungen

Der oben beschriebene Vorgang erfordert die Beachtung folgender Bemerkungen:

a) Eine Störung, die durch eine Einheit im Laufe einer Operation X hervorgerufen wurde, wird durch die Zentraleinheit nur festgestellt und angezeigt, wenn die nächste Operation Y diese Randeinheit anspricht.

b) Nach Beseitigung der Störung wird die in Wartestellung befindliche Operation Y durchgeführt.

c) Eine Programmunterbrechung kann nur am Ende des Befehls erfolgen, der die Operation Y enthält.

Am Ende eines fehlerhaften Lesens oder Schreibens auf der Trommel (Operation X) hat man die Möglichkeit, die Operation zu wiederholen, bevor in der Bearbeitung fortgefahren wird. Dazu muß das Programm am Ende der Operation X angehalten werden, damit die fehlerhaften Daten nicht verarbeitet oder die Daten erneut zum Schreiben bereitgestellt werden können. Dazu muß man unmittelbar nach der Operation X eine Spezialoperation Y programmieren, deren einzige Aufgabe darin besteht, festzustellen, ob die Randeinheit zur Verfügung steht.
Diese besondere Operation ist in Abschnitt 3.6.5 beschrieben.

Das gleiche Verfahren ist auch beim Kartenstanzer anzuwenden, um sich die Möglichkeit zu lassen, bei fehlerhafter Stanzung den Stanzvorgang zu wiederholen.

3.6.5. Besonderheiten des Ein-/Ausgabebefehles

Anschlußfunktionen

Die Anschlußfunktionen der Randeinheiten werden entweder durch spezielle Anschlußcodes oder durch Hilfsschlüssel angerufen. Diese besonderen Fälle der Anwendung des Ein-/Ausgabebefehls sind in den in Abschnitt 3.6.6 angeführten Beispielen beschrieben.

1. Trommel

Zwei spezielle Anschlußcodes ermöglichen es, die Bahn auf der Trommel auszuwählen, auf die geschrieben oder von der gelesen werden soll und den Zustand der Randeinheiten zu testen.

Für jede dieser Operationen drückt der Befehl, zusätzlich zum jeweiligen Anschlußcode, die Nummer des Basisregisters aus, dessen Aufgabe im folgenden für jeden Fall beschrieben wird.

Auswahl der Bahn

Die Operation besteht darin, die Nummer der gewünschten Bahn auszuwählen. Das Basisregister enthält die Adresse - 1 einer Zone von 4 Bytes, die die Nummer der Bahn auf drei Bytes enthält. Auf dem vierten Byte steht eine Trennmarke F4.

Die Auswahl einer Bahn kann nicht durch den gleichen Befehl kontrolliert werden, der auch die Lese- oder Schreiboperation kontrolliert.

Darum sollte die Auswahl der Bahn und das Lesen-Schreiben unabhängig voneinander programmiert werden , damit jede Operation gesondert getestet werden kann.

Test der Einheit

Hier wird der Ein-/Ausgabebefehl dazu benutzt, zu testen, ob die letzte auf der Trommel durchgeführte Operation richtig durchgeführt wurde (s. 3.6.4.). Ist im Laufe dieser Operation ein Fehler aufgetreten, kann der Testbefehl nicht durchgeführt werden (Einheit ist nicht ansprechbar) und blockiert so das Programm.

Das spezifizierte Basisregister enthält die Adresse (nicht die Adresse -1) einer Stelle, die keinen Wert enthalten darf. Um nicht ein besonderes Testregister zu belegen, kann man die Stelle vor der Ein/Ausgabezone für die Trommel verwenden. Bei vorher richtig durchgeführter Operation wird das hexadezimale Zeichen "10" an der Basisadresse zur Verfügung gestellt. Durch Analyse des Testzeichens kann gegebenenfalls die falsch durchgeführte Operation wiederholt werden.

Der Test wird durch einen besonderen Ein-/Ausgabebefehl programmiert, der unmittelbar der Operation folgt, die man testen will: Auswahl, Lesen oder Schreiben.

Anschluß von zwei Trommeln

Der Festspeicher MMS9 ermöglicht den Anschluß von zwei Magnettrommeln. (vgl. 2.8.3.2.)

2. Pufferspeicher, Kartenstanzer, Drucker

Diese Randeinheiten benutzen Hilfsschlüssel, deren Aufgabe in den Kapiteln beschrieben wurde, die sich auf sie beziehen.

Ein Hilfsschlüssel ist eine besondere Anweisung, die unter der Form eines 8 Bit-Zeichens dargestellt wird und aus der im internen Code zur Verfügung stehenden Kombinationen genommen worden ist (Spalten 0 und 1). Er wird im Zentralspeicher wie ein normaler Wert gespeichert und wird an die entsprechende Randeinheit durch einen Ein-/Ausgabebefehl entweder in Verbindung mit Daten oder allein übertragen. Seine Wirkung ist unmittelbar: Sobald eine Randeinheit einen Hilfsschlüssel empfangen hat, führt sie die entsprechende Funktion aus.

Hilfsschlüssel, die eine Information begleiten

Diese Schlüssel dienen dazu, am Ende einer Stanzoperation die Karte auszuwerfen oder am Ende einer Zeile den Papiervorschub zu ermöglichen.

Der Hilfsschlüssel wird in die Ausgabezone, die die auszugebende Information enthält, übertragen; und zwar an die Stelle, wo man die gewünschte Anschlußfunktion ausüben will unter Berücksichtigung, daß die Randeinheit Stelle für Stelle verarbeitet: Handelt es sich um ein gewöhnliches Zeichen, wird es geschrieben, ist es ein Hilfsschlüssel, wird die befohlene Funktion ausgeübt. Will man zum Beispiel einen Papiervorschub nach dem Drucken einer Zeile haben, setzt man den entsprechenden Hilfsschlüssel zwischen die zu druckende Information und die Trennmarke.

Mehrere Hilfsschlüssel können im Laufe einer Operation übertragen werden. Sie werden in der Reihenfolge, in der sie stehen, ausgeführt.

Hilfsschlüssel, die allein übertragen werden.

Diese Schlüssel werden speziell zum Einschalten der numerischen Tastatur und der Leuchtanzeige benutzt.

Der Übertrag dieser Schlüssel besteht aus einer besonderen Operation, die definiert wird durch:

- einen Anschlußcode, der die Randeinheit auswählt;
- die Nummer eines Registers, das die Adresse - 1 der Stelle enthält, an der der Hilfsschlüssel steht.

Diesem Schlüssel muß die Trennmarke F4 folgen.

Die Operation kann mit anderen Operationen der Ein-/Ausgabe im gleichen Befehl programmiert sein.

Anmerkung:

Der Pufferspeicher wird durch zwei verschiedene Anschlußcodes bezeichnet, und zwar je nachdem er empfangendes oder abgebendes Organ ist.

3. Kartenstanzer PS 40

Test der Einheit

Ein besonderer Anschlußcode ermöglicht die Feststellung, ob die zuletzt durchgeführte Operation zufriedenstellend abgelaufen ist.

Dieser Test benötigt einen speziellen Ein-/Ausgabebefehl, der unmittelbar dem Stanzbefehl folgt. Der spezielle Anschlußcode benutzt ein Basisregister, das die rechte Adresse - 1 einer Zone von 2 Bytes enthält, in der irgendein Zeichen und eine Trennmarke steht.

3.6.6. Die Befehle

IØC	Input Output Control
IØIC	Input Output Indirect Control

53	N	C_1	R_1	C_2	R_2	C_3	R_3	C_4	R_4	C_5	R_5	C_6	R_6
54	N	RC_1	R_1	RC_2	R_2	RC_3	R_3	RC_4	R_4	RC_5	R_5	RC_6	R_6

N = Zahl der Randeinheiten von 1 - 6

Für jede Einheit bedeutet:

C = Anschlußcode (s. Tafel)
RC = Nummer des Registers, das den Anschlußcode enthält
R = Nummer des Basisregisters

Die folgenden Angaben werden in die spezielle IØC-Zone übertragen:

- die Zahl N der Randeinheiten;
- je Einheit: - der entschlüsselte Anschlußcode,
  - die Basisadresse des Registers R.

Der Anschlußcode ist enthalten:

- im IØC : im Befehl selbst;
- im IØIC : an der ersten, rechten Stelle des Registers RC.

Anmerkung:

Die Abschnitte 3.6.3 und 3.6.5 legten die Bedingungen zur Anwendung der Befehle IØC und IØIC fest und zeigten, welche Operationen simultan nicht vereinbar sind. Es ist darum erforderlich, daß man sich gut die Operationen merkt, die in einem Befehl angerufen werden können.

BULL

GENERAL ELECTRIC

GE 55 Codetabelle

ANSCHLUSSCODES FÜR RANDEINHEITEN

Kanal		Ohne Halt auf d. Zeich.	Mit Halt auf dem Zeichen	Nummer des Registers	Funktion
normal	1	95	92	10	Puffer → Zentralspeicher
		9D	9A	12	Freigabe num. Tastatur
		96	93	06	Kartenleser
		97	94	R	Freigabe α-Tastatur
	2				
	3				
	1	9D	9A	10	Zentralspeicher → Puffer
		9D	9A	11	Freigabe Sichtanzeige
	2	AD	AA	09	Kartenstanzer
		AF	AC	12	Test
	3	CD	CA	08	Drucker MB 50
schnell	0	10		R	1. Trommel - Lesen
		11		R	- Test
	1	14		R	2. Trommel - Lesen
		15		R	- Test
	2	18		R	Ext. → Zentralspeicher
		19		R	Test
	3	1D		08	Drucker I41 Test
	0	12		R	1. Trommel - Schreiben
		13		R	- Auswahl d. Spur
	1	16		R	2. Trommel - Schreiben
		17		R	- Auswahl d. Spur
	2	1A		R	Zentralspeicher → Ext.
		1B		R	Berechnen Schlüssel
	3	1E		08	Drucker I41 drucken

Ref.-Nr.: 23.57.001

DISPLAY für falsche Karte lesen

9D10 9D12 95-10 ✓ HLB

9D10 9A12 95-10 ✓ HLB

9D10 9D11 9210 ✓ CLB

9D10 9D11 9510 ✓ CLB

hält an,
weiter lesen nach ↑

KKFBBV 151277

17.12.77

Beispiele:

A Kartenleser

IØC |53|01|96|06| Reg. 6 = |00|00|00|22|56|

Zone 6 vorher : |20|20|20| .... |20|20|F4|

nachher : |32|31|30| .... |38|34|F4|

2257 2337

B Pufferspeicher

a) Freigabe numerische Tastatur

IØC |53|01|9A|12| Hilfsschl. Adresse - 1 des Codes

Reg. 12 |0 F|F 4|0 0|0 1|5 5|

b) Übertrag Pufferspeicher an Zentralspeicher

IØC |53|01|92|10| Reg. 10 |00|00|00|23|47|

Zentralspeicher Pufferspeicher

vorher :|20|20|20|20|20|20|F4| 104:

nachher :|20|20|31|30|34|3A|F4|

2348

c) Übertrag Zentralspeicher an Pufferspeicher

IØC |53|01|9A|10| Reg. 10 |00|00|00|23|47|

Zentralspeicher Pufferspeicher

vorher : |20|20|20|39|38|32|F4|

nachher : |20|20|20|39|38|32|F4| 982

2348

d) Freigabe Sichtanzeige Hilfsschlüssel Adresse - 1

IØC |53|01|9A|11| Reg. 11 |0E|F4|00|01|50|

0151

C Stanzen einer Karte und gleichzeitiges Drucken

Befehl der Operationen: IØC |53|02|AA|09|CA|08|

Stanzer
Drucker

Test des Stanzers IØC |53|01|AC|12|

Reg. 8	Reg. 9	Reg. 12
|00|00|00|20|37|	|00|00|00|21|74|	|0F|F4|00|01|55|

Zone 9 |41|42| .. |39|0D|F4|38|F4| Stanzen von 49 Zeichen
2175 2223
Schlüssel "Kartenauswurf"

Zone 8 |44|20|32| ... |38|0D|0C|F4|20 Drucken von 64 Stellen
2038 2097
Schlüssel "Sprung"
Schlüssel "Wagenrücklauf"

Register und Zonen sind nach der Operation unverändert.

D Trommel

a) Lesen von einer Trommel, die an Kanal R0 angeschlossen ist, evtl. unabhängig von anderen.

- Auswahl der zu lesenden Bahn IØC 5301 1318
- Test der Auswahloperation IØC 5301 1105
- Lesebefehl IØC 5301 1005
- Test 2 der Leseoperation IØC 5301 1105

Register 18 |00|00|00|06|00|
Register 05 |00|00|00|13|00|
Nummer der Bahn |30|32|34|F4|
0601

Für die Trommel belegte Zone:

vorher : |00|44|52| .. |54|52|30|48|43|F4|32|
1300 1533

nachher: |30|33|34| .. |37|30|F4|48|43|F4|32|
1300 1533

Daten von Bahn 24 übertragen
(233 Zeichen + Trennmarke F4)

b) Die Benutzung von zwei Trommeln als logische Einheit

Die Programmierung erfolgt hier mit dem IØIC-Befehl. Die Anschlußcodes der Trommel, auf die man schreiben will, werden vor dem entsprechenden Trommelbefehl in den Registern 14, 15 und 16 gespeichert.

In dem folgenden Beispiel ist gerade die Trommel an Kanal R1 in Arbeit.

- Auswahl der Schreibbahn	IØIC	54 01 14 19
- Test 1 der Auswahloperation	IØIC	54 01 16 18
- Schreibbefehl	IØIC	54 01 15 10
- Test 2 des Schreibbefehls	IØIC	54 01 16 18

Reg. 14	Reg. 15	Reg. 16
|00|00|00|00|17|	|00|00|00|00|16|	|00|00|00|00|15|

Reg. 18	Reg. 19	Nr.d. Bahn
|00|00|00|12|00|	|00|00|00|06|00|	|31|32|36|F4|
		0601

Für die Trommel belegte Zone:

vorher: |00|32|48 .. 42|46|F4|38|35|

1300

Daten auf Bahn 126 geschrieben
(157 Zeichen + Trennmarke F4)

nachher: |30|32|48 .. 42|46|F4|38|35|

1300 1457

3.6.7. Die vertikale Simultaneität

Es handelt sich herbei um eine Simultaneität zwischen der Arbeit einer oder mehrerer Randeinheiten auf der einen und Bearbeitungsbefehlen auf der anderen Seite innerhalb desselben Programms. Hierfür werden die IØC-Befehle in 3 Kategorien aufgeteilt.

3.6.7.1. Gleitende IØC-Befehle

Das "N" in der Mitte dieses Befehls, das die Anzahl der zu verbindenden Randeinheiten angibt, wird in der Form 1N geschrieben.

Hier gibt es wiederum 3 Fälle:

- Der IØC spricht eine langsame Randeinheit mit Halt auf dem Zeichen an. Es kommt zu einer Simultaneität Randeinheit - Bearbeitung. Der Befehl, der diesem IØC folgt, wird sofort nach dem Start der Randeinheit durchgeführt.

- Es handelt sich um eine langsame Randeinheit ohne Halt auf dem Zeichen. In diesem Fall wird der Befehl, der dem IØC folgt, erst durchgeführt, wenn dieser beendet ist.

  Beispiel: 53 11 AA 09 (Stanzen)
  Programmfolge
  53 11 96 06 (Lesen)

  Das Stanzen wird gestartet, und das Programm läuft simultan hierzu ab bis zum Auftreten des Lesebefehls. Das Lesen erfolgt simultan mit dem Stanzen, wenn dieses noch nicht beendet ist. Der Befehl, der dem Lesebefehl folgt, wird jedoch erst ausgeführt, wenn das Lesen beendet ist, allerdings auch dann, wenn das Stanzen noch nicht beendet ist.

  Beispiel: 53 11 AD 09 (Stanzen)

  Es handelt sich um einen gleitenden IØC mit einem Verbindungscode ohne Halt auf dem Zeichen. Der folgende Befehl wird erst ausgeführt, wenn das Stanzen beendet ist.

- Der IØC spricht eine schnelle Randeinheit an. Der folgende Befehl wird erst durchgeführt, wenn der IØC beendet ist.

  Im Multiprogramming löst der gleitende IØC weder einen Programmwechsel (Bit 2 im PRC) noch einen Programmhalt (PO im PRC) aus. Ebenso wird dieses Programm nicht wieder "gestartet" (einsetzen der Programmnummer in die PRC-Zone), wenn alle durch den IØC verbundenen Randeinheiten ihre Arbeit beendet haben.

3.6.7.2 Blockierende IØC-Befehle

Die Anzahl der zu verbindenden Randeinheiten wird in der Form ON ausgedrückt.

Erkennen die Schaltkreise des GE-55 einen solchen Befehl, wird ein Programmhalt ausgelöst (PO im PRC). Das Programm wird wieder gestartet, wenn alle durch dieses Programm verbundenen Randeinheiten ihre Arbeit beendet haben (Trennmarke am Ende einer jeden Zone erkannt). Es ist zu beachten, daß es sich hierbei um sämtliche Randeinheiten handelt, verbunden durch:

- den blockierenden Befehl selbst und evtl. durch einen oder mehrere IØC, die vorher vom Programm durchlaufen wurden (gleitende IØC).

Beispiel: 53 11 AA 09
Programmfolge
53 01 96 06

Der Befehl, der dem Lesebefehl folgt, wird erst ausgeführt, wenn sowohl das Lesen als auch das Stanzen beendet ist.

3.6.7.3. Wartender IØC-Befehl

Dieser Befehl ermöglicht es, die Verarbeitung, die simultan zur Durchführung eines gleitenden IØC-Befehls abläuft, zu unterbrechen, ohne daß durch ihn neue Randeinheiten verbunden werden.

Die Anzahl "N" der zu verbindenden Randeinheiten ist 0: 53 00

Er hat die gleichen Charakteristiken wie der blockierende IØC. Wird dieser Befehl zu einem Zeitpunkt durch die Schaltkreise des GE-55 erkannt, zu dem bereits alle durch das Programm verbundenen Randeinheiten ihre Arbeit beendet haben, verhält er sich wie ein Befehl NØP.

Beispiel: 53 12 AA 09 CA 08
simultane Verarbeitung
53 00
keine simultane Verarbeitung

Beispiel: Spezieller Fall des Sichtkartenlesers
53 11 93 06
simultane Verarbeitung in der Wartezeit von 110 Millisekunden
53 00

110 Millisekunden benötigt eine Karte, um vor die Lesezellen zu gelangen. Es ist möglich, in dieser Zeit Verarbeitungsprogramm ablaufen zu lassen.

3.7. Die Umschlüsselungsbefehle

3.7.1. Allgemeines

Einige Randeinheiten können den internen ISØ-Code nicht verarbeiten. Sie erhalten oder geben die Informationen in einem Zwischencode aus.

Der Kartenleser und der Stanzer PS 40 haben zwei Zwischencodes, die von den beiden Lochcodes abhängig sind und zwar:

- den Zwischencode T 121
- den Zwischencode H 14.012

Die Umschlüsselungsbefehle bewirken eine Umwandlung des Zwischencodes in den internen Code und umgekehrt.

Diese Umschlüsselungen finden im Kernspeicher statt und zwar in den Ein- oder Ausgabezonen und gelten jeweils für die gesamte angerufene Zone.

Bearbeitung der Überlochungen im Code T 121 und H 14.012

Die Kombination einer Überlochung mit einer numerischen Lochung kann weder vom Code T 121 noch vom Code H 14.012 in den internen ISØ-Code übersetzt werden:

- da einige Kombinationen im ISØ-Code nicht existieren.
  Beispiel: im Code T 121: Ziffern von 1 - 6 mit 11 oder 12

- da die Kombinationen, die ein Zeichen geben, nach der Übersetzung nicht mehr in ihrem ursprünglichen binären Aufbau als numerisches Zeichen und Überloch auseinandergeführt werden können.
  Beispiel: im Code T 121: Ziffern 9, 8, 7 mit 11 oder 12
  im Code H 14.012: alle Kombinationen.

Dagegen können Überlochungen und die numerischen Werte im Zwischencode leicht unterschieden und getrennt werden; man stellt also vor der Übersetzung in den internen ISØ-Code den ursprünglichen binären Aufbau des numerischen Wertes her, indem man die Bits, die die Überlochung darstellen, eliminiert. Wenn das Überloch für die weitere Bearbeitung noch benötigt wird, werden diese eliminierten Bits in eine vorläufige Bearbeitungszone übertragen.

Im umgekehrten Fall, wenn man in ein und dieselbe Spalte einer Karte ein Überloch und numerische Werte stanzen will, müssen die binären Kombinationen im Zwischencode hergestellt werden; diese Operation wird ausgeführt, nachdem die numerischen Werte, die getrennt im ISØ-Code dargestellt waren, in den Zwischencode übersetzt wurden.

Diese Bearbeitung wird meistens durch logisches UND und logisches ODER je nach den Gegebenheiten durchgeführt.

Zwischencode T 121

Wenn man die Bits von rechts nach links mit 0-7 numeriert, werden die Lochungen 11, 12 und 0 bis 9 wie folgt dargestellt:

Lochungen	Bits	hexadezimaler Code
11	7 und 3	88
12	7	80
0 bis 9	6, 5, 4 und 2, 1, 0	variable

Die Kombination von zwei Lochungen, numerisch und Überloch 11 oder 12, ergibt sich aus der Zusammensetzung der oben angegebenen Bits.

Die Trennung der numerischen Werte von den Überlochungen erfolgt durch Löschen der Bits 7 und 3 mittels logischer Addition folgender hexadezimaler Schlüssel:

- 88 isoliert die Lochung "11";
- 80 isoliert die Lochung "12";
- 77 isoliert die numerischen Lochungen.

Das Einsetzen einer Überlochung über einen numerischen Wert erfolgt dementsprechend durch Belegung der Bits 7 und 3 und zwar wird mittels des logischen ODER mit den hexadezimalen Schlüsseln:

- 88 Wert "11" eingesetzt,
- 80 Wert "12" eingesetzt.

Zwischencode H 14.012

Die Lochungen 0 bis 9 ergeben mit oder ohne Lochungen 11 und 12 kombiniert folgende hexadezimale Schlüssel:

Lochungen	0	1	2	3	4	5	6	7	8	9	nichts
nichts	30	31	32	33	34	35	36	37	38	39	
11	70	51	52	53	54	55	56	57	58	59	50
12	60	41	42	43	44	45	46	47	48	49	40

Die sich ergebenden Schlüssel auf zwei Lochungen, numerisch und Überloch 11 oder 12 (Zeile 2 und 3 der Tafel) zeigen, daß:

a) die Trennung der Überlochung durch Isolierung der Bits 4 und 6 mittels einer logischen Addition mit folgenden hexadezimalen Schlüsseln erlangt werden kann:

- 50 isoliert die Lochung "11",
- 40 isoliert die Lochung "12".

b) die entsprechenden Schlüssel für numerische Lochungen allein (Zeile 1 der Tafel) durch Löschen des linken Halbbytes des kombinierten Schlüssels und durch Belegung der Bits 4 und 5 mittels folgender Befehle, erlangt werden können:

- logisches UND mit dem hexadezimalen Schlüssel OF,
- logisches ODER mit dem hexadezimalen Schlüssel 30.

Das Einsetzen einer Überlochung über einen numerischen Wert im Code H 14.012 (Zeile 1 der Tafel) erfordert das Löschen des linken Halbbytes des numerischen Wertes und die entsprechende Belegung der Bits 4 und 6 mittels folgender Befehle:

- logisches UND mit dem hexadezimalen Schlüssel OF,
- logisches ODER mit dem hexadezimalen Schlüssel:

·50 zum Einsetzen des Wertes "11" (Zeile 2 der Tafel),
·40 zum Einsetzen des Wertes "12" (Zeile 3 der Tafel).

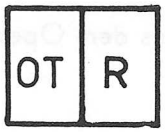
Anmerkung:

Es ist nicht möglich, nach dieser Methode in die gleiche Spalte ein Überloch mit einer Ziffer "0" zu stanzen; die "0" wird nicht erkannt (siehe 1. Spalte der Tafel). Um die Ziffer "0" zu erhalten, muß ein zusätzliches logisches ODER mit dem hexadezimalen Schlüssel 20 durchgeführt werden.

3.7.2. Die Umschlüsselungsbefehle

TRX	TRanslate X

X = siehe nachstehende Tafel.



OT = siehe nachstehende Tafel.
R = Basisregister der zu übersetzenden Zone.

Die Zeichen, die in der durch das Basisregister angegebenen Zone enthalten sind, werden nacheinander in aufsteigender Reihenfolge übersetzt. Auf die Basisadresse wird solange "1" addiert, bis die Trennmarke F4 erkannt wird.

Das Resultat steht in der gleichen Zone, jedes übersetzte Zeichen an der gleichen Stelle, an der es vorher stand.

Die Trennmarke wird nie verändert und nie gelöscht.

Die Übersetzungen:

Symbolischer OT	Maschinen OT	durchgeführte Übersetzung
T R 0	50	Zwischencode T 121 in ISØ-Code
T R 1	51	Zwischencode H 14.021 in ISØ-Code
T R A	5A	ISØ-Code in Zwischencode T 121
T R B	5B	ISØ-Code in Zwischencode H 14.012
T R D	5D	I 41 64 Zeichen

3.8. Die Hilfsbefehle

Die Hilfsbefehle werden im wesentlichen nur bei den Programmen der Software verwendet. Da der Benutzer aber eventuell einige von ihnen gebrauchen muß, werden sie hier beschrieben.

Einige dieser Befehle bestehen nur aus dem Operationstyp.

3.8.1. Kernspeicherausdruck

PRSTØ PRint STOre

OT = 78 für PRSTØ

Dieser Befehl bewirkt den Kernspeicherausdruck über den Drucker und zwar durch Anruf eines fest eingespeicherten Unterprogrammes (ca. 204 Bytes).

Zu Beginn erfolgt systematisch ein Ausdruck bis Kernspeicherstelle 0200. Anschließend wird die numerische Tastatur freigegeben; durch Eintasten der Anfangs- und Endhunderteradresse werden dann die Zeichen Byte für Byte in aufsteigender Reihenfolge ausgedruckt. Bei Eingabe von z. B. 0306 werden die Kernspeicherstellen 0301 bis 0600 ausgedruckt.

Anmerkung:

Der Inhalt der ersten 25 Stellen des Kernspeichers wird nicht gedruckt.

Druckaufteilung

Der Kernspeicherausdruck beginnt mit einem Wagenrücklauf auf Stelle 1 und einem Zeilenvorschub. Anschließend werden je Zeile 25 Bytes zu je 5 gedruckt. Ein Leerzeichen trennt jede 5er-Gruppe.

Die Bytes werden in hexadezimaler Form ausgegeben. Jedes Byte wird durch zwei Zeichen dargestellt.

Die Adresse des ersten Bytes jeder Zeile ist am Anfang der Zeile angegeben. Sie ist durch eine Leerstelle von der Zeile getrennt.

Am Ende des Ausdruckes bleibt der Schreibkopf am Ende der Zeile stehen. Will man mit einer Arbeit fortfahren muß man den Wagenrücklauf und einen Sprung durch die Taste SCR herbeiführen.

Anmerkung:

Diese Befehle können beim Test eines Programmes gebraucht werden, um den Inhalt bestimmter Kernspeicherzonen nach bestimmten Bearbeitungsfolgen auszudrucken. Sie werden an der Stelle in das Programm eingeschoben, die vom Programmierer gewählt wurde. Nach dem Test läuft die normale Bearbeitung weiter.

Der Ausdruck des Kernspeichers kann auch mittels der Drucktaste PRS am Bedienungspult hervorgerufen werden. → Z 28/29

Beispiel:

```
1026  47515A6B57  4E52 .....
1051  5263616272  7121 .....
1076  6022233132  3538 .....
1101  3500003728  2931 .....
```

3.8.2. Laden des Kernspeichers

LØAD LOAD into store

7D

Dieser Befehl lädt den Inhalt einer oder mehrerer Karten in den Kernspeicher, normalerweise ein Programm oder eine Konstante.

Die zu speichernden Informationen sind:

- in der ersten Karte von Spalte 11 bis 80 gelocht,
- in den Folgekarten von Spalte 1 bis 80 gelocht.

Das Ende der Information wird durch das Zeichen "FF" angezeigt, das in der letzten Karte hinter das letzte zu speichernde Zeichen in zwei Spalten gestanzt wird. Die Information wird in hexadezimaler Form ausgedrückt: jede Spalte der Karte stellt ein Halbbyte dar und enthält ein hexadezimales Zeichen (0 bis F). Zwei aneinandergrenzende Spalten, ungerade und gerade, entsprechen einem Byte.

Die erste Karte enthält unter anderem die notwendigen Angaben für das Laden, und zwar:

- in Spalte 1 und 2 einen Schlüssel, der den benutzten Lochcode angibt:

  "00" für Code T 121,
  "04" für Code H 14.012.

- in Spalte 3 bis 6 die Speicheradresse der in Spalte 11 und 12 gelesenen Zeichen,
- in Spalte 7 bis 10 die Startadresse.

Das Laden des Kernspeichers erfolgt in aufsteigender Reihenfolge ab der 1. Speicheradresse. Die gelesenen Spalten werden übersetzt und anschliessend je zwei auf eine Kernspeicherstelle verdichtet. Das Laden wird beendet, wenn das Zeichen FF erkannt wird, das nicht eingespeichert wird.

Wenn der Befehl "LØAD" ausgeführt ist, übernimmt das Programm den Befehl, der sich an der oben erwähnten Startadresse befindet, und der in das Programm-Adreß-Register übertragen worden ist.

Anmerkung:

a) Es können zwei Ladebefehle nacheinander ausgeführt werden unter der Voraussetzung, daß zwischen dem Ende des ersten und dem Anfang des zweiten Befehls 3 ms verstreichen.

b) Die oben beschriebenen Funktionen des Ladens können auch von der Bedienung mittels einer am Bedienungspult befindlichen Taste "LØAD" durchgeführt werden.

3.8.3. Adressierung der Sprungstufen

R L A Replace Levels by Adresse

6A	00	Bxxx

Bxxx = Speicheradresse im Programm

Dieser Befehl ersetzt in einem Programm die symbolischen Sprungadressen (siehe 3.2.) durch ihre wirkliche Adresse.

Die reelle Adresse jeder Sprungstufe, die durch einen Pseudobefehl LEVEL angegeben ist, wird ab der Anfangsadresse Bxxx des Programmes im Kernspeicher berechnet. Dann wird sie an die Stelle der symbolischen Adresse gesetzt, und zwar bei allen Sprungbefehlen, die sich im Programm befinden.

Am Ende der Operation sind alle Pseudobefehle herausgefallen, und das Programm ist zur Adresse Bxxx hin verschoben worden. Diese Verschiebung wird bei der Kernspeicherbelegung mit gerechnet.

Beispiel:

R L A 6A|00|08, 00| Basis 0 = 0000

|04|3D|00|86|F1|A1|80|08|60|60|90|38|F2|A1|80|18|60|

900 905 910 915

|04|3D|00|86|09|12|80|08|60|60|90|38|80|18|60|32|90|

900 905 910 915

3.8.4. Fakultatives Halt

KHLT Key HalT

Dieser Befehl ermöglicht es, an einer bestimmten Stelle der Bearbeitung ein fakultatives Halt in das Programm einzuschieben, z.B. am Anfang des Druckens auf ein Blatt.

Der Halt wird durch die Bedienung mit der Drucktaste KHLT freigegeben.

Wenn die Drucktaste KHLT eingelegt ist:

- hält das Programm an dem Befehl KHLT an.
- ein Druck auf die Taste START hebt den Halt auf; das Programm fährt mit dem folgenden Befehl fort. Wenn ein neuer KHLT-Befehl erkannt wird, hält das Programm aufs neue an, aber nur, wenn die Taste KHLT eingelegt bleibt.

Wenn die Drucktaste KHLT nicht eingelegt ist:

- ist der KHLT-Befehl ohne Wirkung. Das Programm läuft ohne Halt weiter.

3.9. Die Pseudobefehle

Die Pseudobefehle der Grundsprache dienen teils der Steuerung des Arbeitssystems und beziehen sich auf das Laden eines Programms in den Kernspeicher, teils sind es feste Werte, die an bestimmten Stellen des Zentralspeichers gespeichert werden.

Da sie keine Beziehung zur Maschinensprache haben, entfällt bei ihnen auch der entsprechende Binärwert des Operationstypes. Sie setzen sich lediglich aus dem symbolischen OT und evtl. einigen Parametern oder festen Werten zusammen. Diese werden symbolisch dargestellt, so wie sie in den folgenden Abschnitten beschrieben werden.

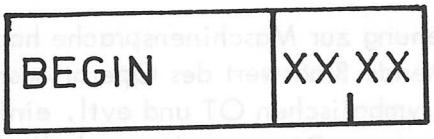
Man unterscheidet zwei Arten von Pseudobefehlen:

- organisatorische Pseudobefehle, die den Anfang und das Ende eines Programms festlegen sowie die vorläufigen Niveaus (Anfang der Sektionen).
- Pseudobefehle zur Erstellung von "Konstanten" (festen Werten), die feststehende und immer wieder beim Ablauf der Programme benutzten Daten eingeben.

3.9.1. Organisation eines Programmes

3.9.1.1. Programmanfang

BEGIN	BEGINing adress where the program is loading



xxxx = absolute Adresse

Dieser Pseudobefehl steht immer am Anfang eines Programmes und gibt dem Ladeprogramm die Adresse im Zentralspeicher an, von der ab die Befehle eingespeichert werden, die diesem Befehl folgen.

Beispiel:

BEGIN	06 01
ADD	A1 23 34
CMD	91 34 78

Zentralspeicher

A1	23	34	91	34	78

0601 0604

3.9.1.2. Programmende

END

END of program

END		F3

F3 = fester Wert (Trennmarke)

Dieser Pseudobefehl steht immer am Ende eines Programmes und zeigt an, daß der vorhergehende Befehl der letzte des betreffenden Programmes ist.

Er bewirkt das Einsetzen der Trennmarke F3 an die Stelle, die diesem Befehl folgt.

Beispiel:

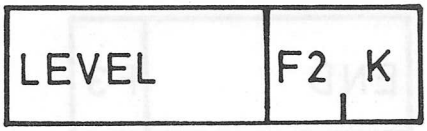
IØC	53	01	CA	08	
END	F3				

Zentralspeicher

53	01	CA	08	F3

3.9.1.3. Anfang einer Sektion

LEVEL	LEVEL at which to stop jump.



F2 = festes Zeichen, das den Pseudobefehl darstellt.
K = 2 hexadezimale Zeichen, die das Etikett des Niveaus darstellen (die Kombinationen F1 bis FF sind verboten).

Der Pseudobefehl LEVEL erfüllt in einem in die Maschinensprache umgewandelten Programm keine Aufgabe und erzeugt darum auch keinen Befehl.

Während des Ladens eines Programmes in den Kernspeicher wird das Bezugszeichen jedes Niveaus in allen Sprungbefehlen durch die reelle Adresse des Befehls ersetzt, der dem betreffenden Pseudobefehl unmittelbar folgt.

Die Pseudobefehle werden anschließend unterdrückt; das Programm entsprechend zusammengeschoben, wobei bei der Bestimmung der reellen Adresse dieser Vorgang berücksichtigt wird.

Beispiel:

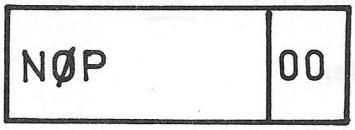
Anfangsprogramm		Programm in Maschinensprache	
Adr. A	Befehle	Adr. B	Befehle
0800	JRT 02 F1 1A	0800	02 09 54
0954	LEVEL F2 1A	0954	80 12 60 60 90 40
0956	MVC 80 12 60 60 90 40	0956	Fortsetzung des Programmes
	Fortsetzung des Programmes		

Adressen A: Adressen bei Laden vor dem Zusammenschieben

Adressen B: endgültige reelle Adressen.

3.9.1.4. Nullbefehl

NØP — No OPeration



Dieser Befehl erzeugt zwei gepackte Nullen an der Stelle, an der er im Programm steht.

Wenn das Programm auf diese Stelle stößt, wird keine Funktion ausgelöst. Das PAR erhöht sich aber um 1. Dadurch wird der Programmablauf nicht unterbrochen und der nächste Befehl kommt zur Durchführung.

Anmerkung:

Wenn dieser Pseudobefehl innerhalb einer Befehlsfolge steht, darf diese Stelle kein anderes Zeichen erhalten, da dieses von der Zentraleinheit als Operationstyp gewertet wird und ein nicht vorherzusehendes Resultat ergibt.

Beispiel:

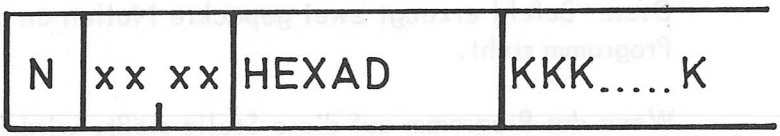
			PAR
800	NØP	vorher	0800
		nachher	0801
801	MVC		
	etc.		

3.9.2. Erstellung von Konstanten

Die Pseudobefehle zur Erstellung von Konstanten haben einen besonderen Aufbau, so daß sie nicht innerhalb eines Programmes auftreten dürfen. Ihr Platz ist entweder unmittelbar vor dem BEGIN oder nach dem END.

3.9.2.1. Hexadezimale Konstanten

HEXAD — HEXADezimal data to load.



N = Zahl der Bytes, die von den einzelnen Zeichen belegt werden (max. 32);
xxxx = absolute Adresse, an der das 1. Zeichen von links eingespeichert werden soll;
KK...K = hexadezimale Zeichen, die eingesetzt werden sollen (2-64 Zeichen, K = 2 Zeichen).

Dieser Pseudobefehl bewirkt das Erstellen von Zeichen K an den Stellen, die der absoluten Adresse xxxx folgen.

Die Zeichen K werden von links nach rechts eingespeichert, so daß je 2 ein Byte in aufsteigender Reihenfolge belegen.

Beispiel:

Erstellen einer Basisadresse im Register 8

```
|02|01,39|HEXAD |21|14|
```

Register 8

```
vorher   |00|00|00|00|00|
nachher  |00|00|00|21|14|
                  0140
```

3.9.2.2. Alphanumerische Konstanten

AN AlphaNumeric data to load.

N = Zahl der Bytes, die von den einzelnen Zeichen belegt werden (max. 64);
xxxx = absolute Adresse, an der das 1. Zeichen von links eingespeichert werden soll;
AA...A = alphanumerische Zeichen, die eingesetzt werden sollen (1-64; A = 1 Zeichen).

Dieser Pseudobefehl bewirkt das Erstellen von Zeichen A an den Stellen, die der absoluten Adresse xxxx folgen.

Die Zeichen A werden von links nach rechts eingespeichert, so daß 1 Zeichen ein Byte in aufsteigender Reihenfolge belegt.

Beispiel:

Erstellen von konstanten Daten

| 06 | 08 | 30 | AN | 1 | 0 | 8 | % |

vorher: | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |

nachher: | 00 | 20 | 31 | 30 | 38 | 25 | 20 | 00 |

0830

4. Simultaneität und Multiprogramming beim GE-55

4.0. Vorbemerkungen

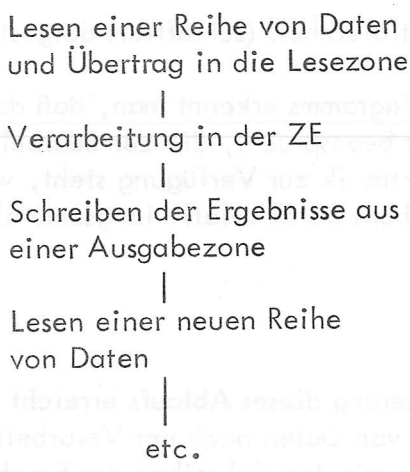
Der erste Abschnitt dieses Kapitels hat es sich zur Aufgabe gestellt, anhand eines ganz allgemeinen Plans eine Vorstellung der Simultaneität und des Multiprogramming, die wiederum nur eine erweiterte Form der Simultaneität ist, zu geben. Die verschiedenen Stufen des Simultanarbeitens werden dargelegt, mit denen eine Optimierung des Nutzen eines Rechners, der nur mit einem Processor ausgestattet ist, wie der GE-55 erreicht werden kann. Es wird gezeigt:

- wo jede Stufe gerechtfertigt ist und was sie für Folgen hat;
- wie man sie anwendet;
- wo die kritische Grenze ist, nach der eine Optimierung illusorisch ist.

Dieser erste Abschnitt nimmt darum auch keinen Bezug auf die Art und Weise, mit der diese Stufen der Simultaneität des GE-55 realisiert werden. Das ist Aufgabe der Abschnitte 4.2. und 4.3. Im besonderen soll man keine Schlüsse in bezug auf den GE-55 aus den Diagrammen ziehen, die ganz allgemein das in 4.1. gesagte illustrieren sollen.

4.1. Allgemeine Einführung in die Simultaneität

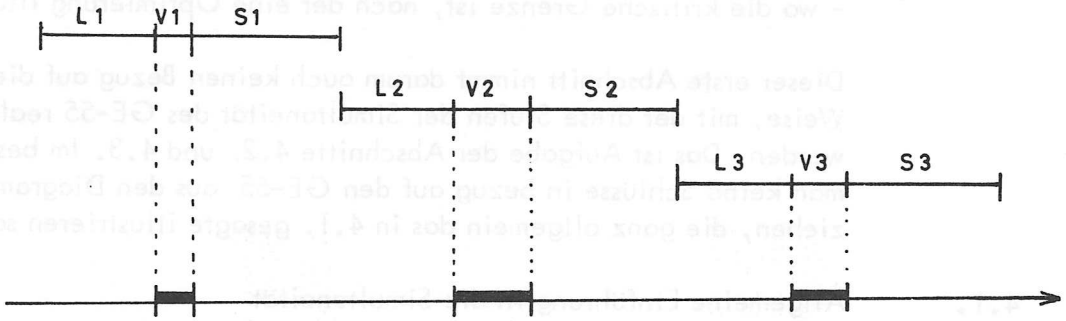
Im Normalfall hat ein Programm folgenden Aufbau:



Bezeichnet man mit:

L 1 die Dauer des Lesens der Daten 1,
V 1 die Dauer der Verarbeitung dieser Daten,
S 1 die Dauer des Schreibens der Ergebnisse,
L 2, V 2, S 2, die Dauer dieser Arbeiten für die Daten 2 etc.

So entsteht folgendes Diagramm der Operationen für eine Maschine, die keine Möglichkeit der Simultanarbeit hat:



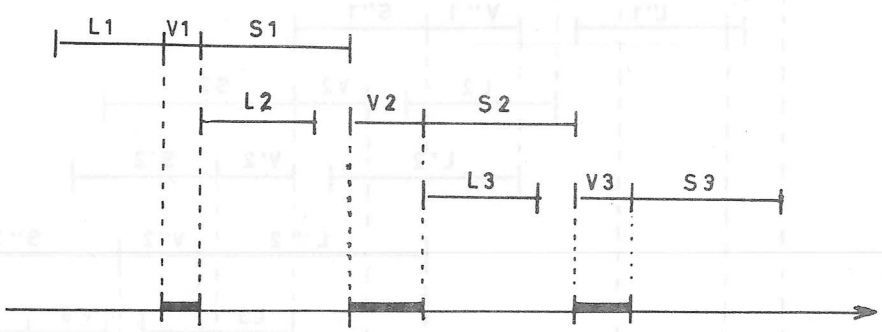
Belegung der Zentraleinheit (schraffiert dargestellt).

Aufgrund dieses Diagramms erkennt man, daß das Programm nur einen kleinen Bruchteil der Zeit beansprucht, die zur Bearbeitung durch die Zentraleinheit mit ihrer Elektronik zur Verfügung steht, während die Funktionen des Lesens und Schreibens durch relativ langsame elektromechanische Geräte erfolgen.

1. Abschnitt

Eine erste Verbesserung dieses Ablaufs erreicht man, wenn man das Lesen der zweiten Serie von Daten nach der Verarbeitung der Daten 1 vornimmt, d.h., gleichzeitig mit dem Schreiben der Ergebnisse 1. Da man nach der Verarbeitung V1 die Daten 1 nicht mehr benötigt, kann man neue Daten in die Lesezone übertragen.

Das Diagramm sieht dann folgendermaßen aus:



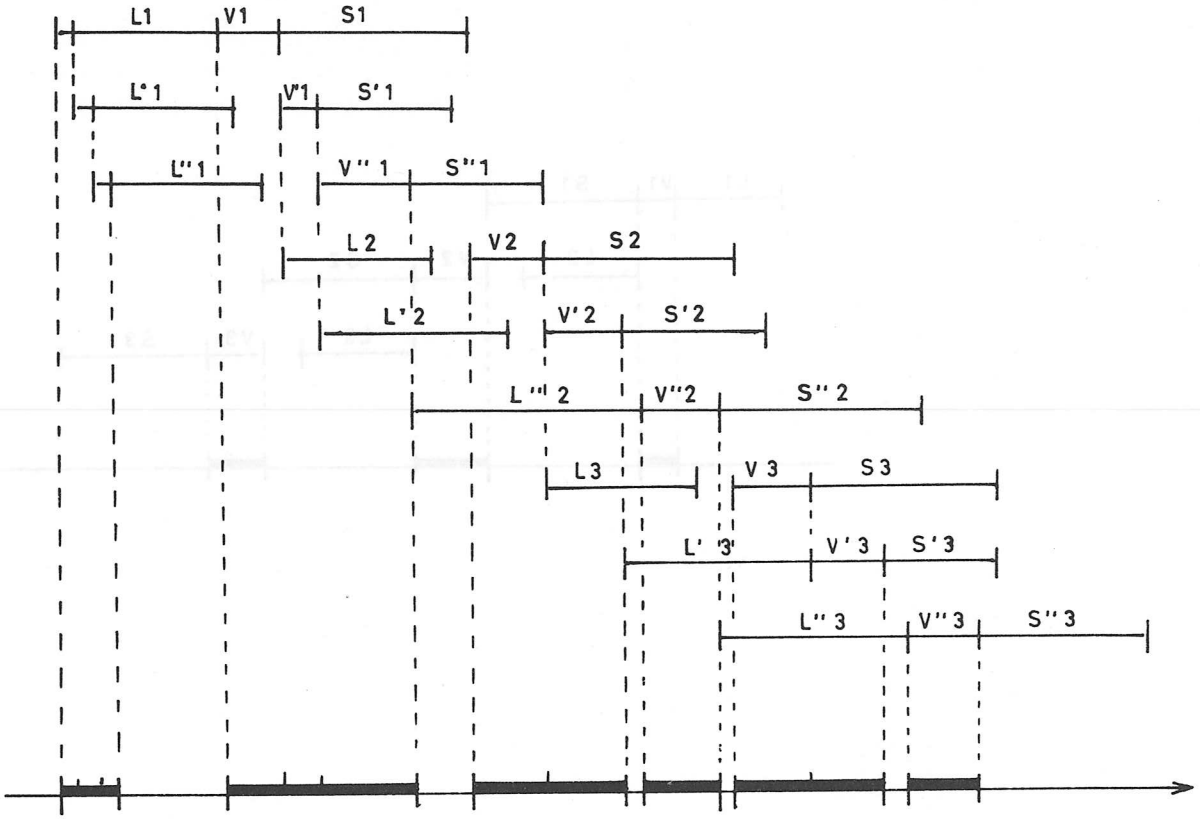
In diesem angenommenen Beispiel ist der Schreibvorgang länger als der Lesevorgang. Die Verarbeitung V 2 kann also nicht vor dem Ende des Schreibens S 1 beginnen, weil sonst die Ausgabezone zerstört würde.

Diese Simultaneität der Ein/Ausgaben entspricht der Hardware des GE-55 . Die Gesamtdauer eines Programms ist also gleich der Summe der eigentlichen Verarbeitungszeit plus der Zeit für die längste Ein/Ausgabeoperation.

2. Abschnitt

Trotzdem ist man weit davon entfernt, alle Möglichkeiten der Verarbeitung innerhalb der Maschine voll auszuschöpfen. Um in dieser Richtung Fortschritte zu erzielen, öffnet das Multiprogramming einen Weg, der, gut angewendet, es erlaubt, fast das Optimum zu erzielen.

Das Multiprogramming besteht daraus, parallel mehrere Programme, die sich zur gleichen Zeit im Zentralspeicher befinden, auszuführen. Das folgende Diagramm zeigt den Ablauf von drei Programmen P, P und P" im Zusammenhang mit Multiprogramming. L 1 bezeichnet die Leseoperation der ersten Daten des Programms P; V3" die Verarbeitung der dritten Daten des Programms P" etc.



Anfang der Arbeit

Dieses Schema zeigt, daß bei einer solchen Organisation die drei Programme im ganzen genommen simultan, praktisch in der gleichen Zeit arbeiten, die sie auch allein brauchen würden.

Das Ergebnis ist wie folgt:

1) Die Möglichkeiten der Simultanarbeit der Ein/Ausgabeeinheiten werden auf breiter Basis angewendet.

2) Die elektronischen Organe werden zu einer Verarbeitung benutzt, die wechselweise den verschiedenen Programmen zur Verfügung steht.

Untersucht man das Diagramm, so ist folgendes festgehalten:

a) in einem bestimmten Augenblick können die simultan ablaufenden Ein- und Ausgaben verschiedene Programme betreffen.

b) Die Ein- und Ausgaben eines Programms werden simultan zur Verarbeitung der anderen Programme ausgeführt.

c) Es erfolgt keinerlei Simultaneität in bezug auf die eigentliche Verarbeitung. Diese erfolgte durch die gleichen Schaltkreise der Zentraleinheit und zwar in der Reihenfolge, in der diese Schaltkreise zur Verfügung stehen. Das hat gewisse Wartezeiten im Ablauf der Programme zur Folge.

Zusammenfassung:

Es ist also letztlich nicht exact, zu sagen, daß die Programme simultan immer in der gleichen Zeit arbeiten, in der sie auch allein laufen würden. Dafür sind aber die Schaltkreise fast ständig in Arbeit. Je näher sie der Vollbelegung kommen, um so weniger treten Wartezeiten auf und die oben gezeigte Formel erhält ihren vollen Wert. Aber in dem Augenblick, in dem die Verarbeitungsmöglichkeiten der Maschine erschöpft sind, ist es evident, daß das Multiprogramming eine Täuschung sein muß. Für eine gegebene Maschine bestimmt das Verhältnis von internen Verarbeitungsgeschwindigkeiten zu der Geschwindigkeit der Ein/Ausgaben die maximale Zahl von Programmen, die man tatsächlich simultan und genau so schnell, als wenn sie allein wären, ablaufen lassen kann. Für den GE-55 ist diese Zahl 5. Das ist der Grund dafür, daß - wie man im Abschnitt über das Multiprogramming sehen wird - die logische Organisation der Maschine die mögliche Zahl der simultan arbeitenden Programme auf 5 begrenzt.

4.2. Die Simultaneität der Ein/Ausgaben des GE-55

4.2.1. Darstellung

Es ist bekannt, daß mehrere Randeinheiten zur gleichen Zeit im gleichen Ein/Ausgabebefehl (IØC oder IØIC - s.3.6.) unter der Voraussetzung, daß sie an verschiedenen Kanälen angeschlossen sind, angerufen werden können. Sie können aber auch durch mehrere Befehle in verschiedenen Programmen unter dem gleichen Vorbehalt angerufen werden. Diese Möglichkeit wird im Abschnitt über Multiprogramming beschrieben.

Die Aufgabe dieses Kapitels ist es, festzulegen, unter welchen Bedingungen die Randeinheiten arbeiten, wenn die Befehle wie folgt gegeben werden. Vorher müssen jedoch drei Typen von Ein/Ausgabeeinheiten festgelegt werden:

1) Normale Randeinheiten, die auf jedem Zeichen halten können, wie Drucker MB 50 und Stanzer PS 40.
   Der Übertrag des Inhalts einer Speicherzone nach einer solchen Einheit kann zu jedem Zeitpunkt unterbrochen und später fortgesetzt werden.
2) Normale Randeinheiten, die nicht auf jedem Zeichen halten können, wie der Kartenleser.
   Der Übertrag der Zeichen nach oder von einer solchen Einheit kann nicht unterbrochen werden.
3) Schnelle Randeinheiten wie die Trommel und der Drucker I 41.

Der Pufferspeicher, der im gleichen Rythmus mit der Zentraleinheit selbst arbeitet, wird nicht berücksichtigt.

4.2.2. Anwendung

Wie bekannt belegt ein Ein/Ausgabebefehl eine bestimmte Zone im Zentralspeicher, die IØC-Zone von Stelle 31 - 59. Infolgedessen steuert der verdrahtete Supervisor der Zentraleinheit aufgrund des Inhalts dieser Zone vollständig den Anschluß der Randeinheiten ohne weiteren Befehl durch das Programm.

4.2.2.1. Die IØC - Zone

Sie besteht aus zwei Teilen:

- 24 Bytes (31 - 54) sind unterteilt in 6 Gruppen zu je 4 Bytes, die "IØC-Register" genannt werden. Jede dieser Gruppen enthält alle notwendigen Angaben zur Steuerung einer Einheit.
- 5 Bytes (55 - 59) werden entsprechend den 5 Programmen belegt, die simultan arbeiten können (Zusammenhang mit Multiprogramming).

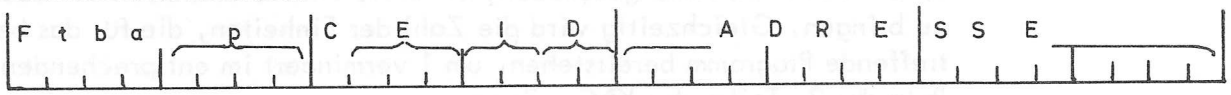
Sie enthalten zu jedem Zeitpunkt die Anzahl der Randeinheiten, die für jedes Programm effektiv zur Verfügung stehen.

4.2.2.2. Die IØC - Register

Sie enthalten für jede in Betrieb befindliche Einheit:

- in den ersten zwei Bytes:
  - die Programmnummer für das sie arbeitet,
  - die Nummer des Kanals, an den sie angeschlossen ist,
  - die Nummer der Einheit an diesem Kanal,
  - den Typ der Einheit (langsam, schnell, ohne oder mit Halt auf dem Zeichen),
  - die durchzuführende Operation (Lesen, Schreiben etc.);
- in den anderen zwei Bytes:
  - die Kernspeicherstelle des nächsten zu übertragenden Zeichens. Am Anfang ist diese Adresse gleich dem Inhalt des im IØC angegebenen Registers.

IØC - Register



F = 1 Freie Gruppe
t = 0 Einheit ist angeschaltet; = 1 Einheit ist abgeschaltet
b = 0 Einheit kann auf der Stelle halten oder (1) nicht
a = 0 Einheit ist anzuschliessen; = 1 ist oder war angeschlossen
p = Nummer des Programms
C = 0 schnelle und (1) langsame Randeinheit
E = Nummer der Einheit des Kanals
I = Nummer des Kanals
D = Aufgabe (Lesen, Schreiben etc.)

Zu einem gegebenen Zeitpunkt kann man also bis zu 6 Randeinheiten in dieser Zone haben, die simultan arbeiten und durch einen oder mehrere IØC (oder IØIC) Befehle angerufen wurden.

Vorstehendes ist die Beschreibung der Steuerung der Randeinheiten nach Belegung der IØC - Zone.

4.2.3. Ablauf der Organisationen

Man hat drei Möglichkeiten zu unterscheiden:

- Alle Einheiten, die in der IØC - Zone gespeichert sind, sind langsame Einheiten mit Halt auf jedem Zeichen;
- Es sind langsame Einheiten ohne Halt auf jedem Zeichen;
- Es sind schnelle Einheiten.

4.2.3.1. Alle Einheiten sind langsam mit Halt auf jedem Zeichen.

Die Zentraleinheit tastet alle Steuergruppen ab, die effektiv belegt sind. Im Verlauf dieser Abtastung gibt sie einen Übertragungsbefehl für ein Zeichen an alle Einheiten, die, mechanisch gesehen, bereit sind. Am Ende der Abtastung wird ein Programm gestartet, das sich in der Bearbeitungsphase befindet (soweit eins vorhanden ist. Das ist nur in Verbindung mit Multiprogramming möglich) um einen Befehl durchzuführen, bevor eine neue Abtastung erfolgt und evtl. neue Befehle zur Übertragung eines Zeichens gegeben werden. Man sieht also, daß im Zusammenhang mit dem Multiprogramming eine effektive Simultaneität von mehreren Ein/Ausgaben und Verarbeitung erzielt werden kann. Ist eine Bearbeitung etwas länger, kann es sein, daß eine Einheit etwas auf den Befehl zum Übertrag eines Zeichens warten muß. Das hat jedoch keine Bedeutung, weil die Einheit zwischen zwei Zeichen halten kann. Die einzige Folge ist nur ein kurzer Aufenthalt.

Bei jedem von oder nach einer Randeinheit übertragenen Zeichen wird die in den Steuergruppen enthaltene Adresse um 1 weitergerechnet. Am Ende des Übertrags, der durch das Erkennen einer Trennmarke festgestellt wird, wird die Steuergruppe dieser Einheit gelöscht und die IØC - Zone zusammengeschoben, um evtl. andere Einheiten in Arbeit zu bringen. Gleichzeitig wird die Zahl der Einheiten, die für das betreffende Programm bereitstehen, um 1 vermindert im entsprechenden Byte des 2. Teiles der IØC - Zone.

4.2.3.2. Die IØC - Zone enthält Steuergruppen langsamer Randeinheiten ohne Halt auf jedem Zeichen

Die langsamen Randeinheiten, mit oder ohne Halt an jeder Stelle, können wegen der Abtastung der Steuergruppen der IØC - Zone simultan arbeiten. Aber im Gegensatz zu dem vorher Gesagten wird durch die Zentraleinheit kein evtl. in der Bearbeitungsphase befindliches Programm gestartet, weil Einheiten ohne Halt auf jeder Stelle sich in Arbeit befinden. Es erfolgt eine ständige Abtastung der IØC - Zone, um in der erforderlichen Zeit die Befehle zur Übertragung eines Zeichens an diese Randeinheit geben zu können. So könnte z.B. das Lesen einer Karte nicht unterbrochen werden, um einen Befehl abzuwarten, ohne daß Zeichen verlorengingen.

Die unterbrochene Abtastung der IØC - Zone bewirkt also, daß ein Simultanarbeiten in Verbindung mit einer Randeinheit ohne Halt auf dem Zeichen nicht möglich ist.

Wie im vorigen Fall erfolgt auch hier nach Beendigung eines Übertrags ein Zusammenschieben der IØC - Zone. Es ist übrigens möglich, daß bei dieser Gelegenheit auf den vorhergehenden Fall genau zurückgekommen wird. (Durch die Möglichkeit der Simultaneität der Verarbeitung und der Ein/Ausgaben).

4.2.3.3. Die IØC - Zone enthält Steuergruppen für schnelle Randeinheiten (wie die Magnettrommel oder der Drucker I 41)

Die Natur dieser Randeinheiten erfordert es, daß Überträge nur ohne Unterbrechung zwischen den Zeichen erfolgen können. Weiterhin erlaubt es der Rhythmus des Übertrags der Zentraleinheit nicht, wie in den vorigen Fällen, mehrere Randeinheiten zu steuern. Wenn im Laufe der Abtastung der IØC - Zone die Zentraleinheit gleichzeitig auf folgende drei Bedingungen trifft:

- die IØC - Zone enthält Steuergruppen schneller Einheiten,
- diese schnellen Einheiten sind bereit, Überträge auszuführen,
- keine langsame Einheit ohne Halt auf dem Zeichen ist in Arbeit,

erfolgt der Übertrag in einem Zug. Die Abtastung der IØC - Zone wird unterbrochen. Das bedeutet, daß der Anschluß einer schnellen Randeinheit jede Simultanarbeit ausschließt, sowohl mit anderen Ein/Ausgaben als auch mit Bearbeitungen. Sobald der Übertrag beendet ist, beginnt die Abtastung erneut nach dem Zusammenschieben und man kommt auf einen der drei vorher genannten Fälle zurück.

4.2.3.4. Der Pufferspeicher

Wird im Laufe der Abtastung der IØC - Zone eine Steuergruppe angetroffen, die sich auf den Pufferspeicher bezieht, erfolgt der Übertrag von 6 Zeichen in einem Zug, wie bei einer schnellen Randeinheit. Man kann aber nicht von einer Unterbrechung der Simultaneität sprechen, weil der Übertrag im Rhythmus des Zentralspeichers erfolgt und auf dieser Stufe keinerlei Simultaneität möglich ist.

4.2.4. Simultaneität des I 41

4.2.4.1. Simultaneität des I 41 mit einer langsamen Randeinheit.

Die Schaltkreise des GE-55 erlauben eine Simultaneität des I 41 mit einer einzelnen langsamen Randeinheit, gleich, ob diese Randeinheit auf dem Zeichen hält oder nicht.

4.2.4.2. I 41 und eine Randeinheit mit Halt auf dem Zeichen. Die Einheiten arbeiten voll simultan.

Beispiel: IØC mit I 41 und Stanzer

Die Placierung der Einheiten innerhalb des IØC-Befehls (I 41 - Stanzer oder umgekehrt) ist ohne Einfluß auf die Simultanarbeit.

4.2.4.3. I 41 und eine langsame Randeinheit ohne Halt auf dem Zeichen

Entsprechend dem unter 2.7.1. aufgezeigten Verfahren ist eine Simultanarbeit nur dann möglich, wenn das Register AK im Augenblick der Verbindung des I 41 Null enthält.

Das bedeutet, daß der I 41 zuerst gestartet werden muß, wenn man eine Simultaneität erreichen will.

Beispiele: IØC mit Leser und I 41
IØC mit I 41 und Leser

In dem ersten Beispiel wird der Leser zuerst gestartet. Das Register enthält dann "1", und die Verbindung des I 41 ist nicht möglich. Er wird erst dann gestartet, wenn das Lesen beendet ist.

In dem zweiten Beispiel ist eine volle Simultaneität gegeben. In dem Augenblick, in dem der I 41 zu verbinden ist, enthält das Register AK Null. Der I 41 wird gestartet. Die Schaltkreise des GE-55 erlauben eine Simultanarbeit des I 41 mit einer langsamen Randeinheit und somit wird auch der Leser gestartet.

4.2.4.4. I 41 und mehrere langsame Randeinheiten

Das kann sich in den folgenden 3 Fällen ergeben:

- ein IØC spricht mehrere Randeinheiten an (maximal 6)
- in der vertikalen Simultaneität
- im Multiprogramming

Grundsätzlich tasten die Schaltkreise des GE-55, wenn der I 41 einmal gestartet worden ist, den ersten Bereich der IØC - Zone ab und ermitteln von links nach rechts die erste langsame Randeinheit, die schon verbunden ist oder noch zu verbinden ist. Das führt zu einer Simultaneität zwischen dem I 41 und dieser langsamen Einheit.

Beispiel: IØC - Stanzer - I 41 - Leser

Der Stanzer wird zuerst verbunden, dann der I 41 (AK = 0), und die erste langsame Randeinheit wird ermittelt. Die erste Einheit, die von links gefunden wird, ist der Stanzer. PS 40 und I 41 arbeiten also simultan. Wenn der I 41 seine Arbeit beendet, kommt es zu einer Simultaneität zwischen Stanzer und Leser. Wenn das Stanzen vor dem Drucken beendet ist, wird der Ausdruck allein weitergeführt, und erst danach wird der Leser verbunden.

Beispiel: (vertikale Simultaneität)
53 11 AA 09

simultanes Programm
53 12 1E 08 96 06

Durch den ersten IØC wird das Stanzen gestartet, und das anschließende Verarbeitungsprogramm läuft simultan dazu ab. In dem Fall, daß das Stanzen bei der Aufnahme des doppelten IØC I 41 - Leser noch nicht beendet ist, befinden sich zu diesem Zeitpunkt in der IØC - Zone:

PS 40 - I 41 - Leser

Der Stanzer ist verbunden.
Das Register AK ist Null.

Der I 41 wird verbunden, und das Abtasten im ersten Bereich der IØC - Zone wird aufgenommen. PS 40 und I 41 arbeiten simultan, und wir kommen zurück auf den vorangegangenen Fall.

Bei Beendigung des Stanzens vor der Aufnahme des IØC I 41 - Leser kommt es zu einer Simultanarbeit I 41 - Leser.

4.2.4.5. Im Multiprogramming ist das Problem im Prinzip genau so wie bei der vertikalen Simultaneität; die einzelnen IØC in der IØC-Zone kommen hier lediglich aus verschiedenen Programmen.

4.3. Multiprogramming beim GE-55

4.3.1. Darstellung

Es ist beim GE-55 möglich, maximal 5 Programme, die sich gleichzeitig im Zentralspeicher befinden, nach den Regeln des Multiprogramming ablaufen zu lassen.

Wie am Anfang dieses Kapitels gezeigt wurde, bestehen diese Programme aus Verarbeitungsserien, die durch Ein/Ausgabebefehle (IØC) getrennt werden. Trifft eines von ihnen im Laufe der Arbeit einen IØC - Befehl, setzt dieser Befehl eine oder mehrere Randeinheiten nach den im vorigen Abschnitt beschriebenen Regeln in Arbeit. Solange die Ein/Ausgabeoperationen dieses Programms nicht beendet sind, wird mit der Weiterverarbeitung dieses Programms ausgesetzt. Man sagt, daß ein solches Programm "erstarrt" ist. Das ist der Fall, wenn der Inhalt des PAR konstant bleibt.

Ein anderes Programm, das sich in der Verarbeitungsphase befindet, wird also zur Ausführung gebracht. Es läuft simultan mit den Ein/Ausgabeoperationen des "erstarrten" Programms ab, die Befehle schieben sich in die Abtastung der IØC - Zone unter den vorher gesagten Bedingungen ein. Sobald das zweite Programm in seinem Ablauf auf einen Ein/Ausgabebefehl trifft, "erstarrt" es ebenfalls und es wird zu einem weiteren Programm übergewechselt. Kann allerdings der Ein/Ausgabebefehl nicht in die IØC - Zone eingespeichert werden, weil die Zahl der zur Verfügung stehenden Steuergruppen nicht ausreicht, wird der Programmwechsel solange aufgeschoben, bis der Befehl effektiv zur Ausführung gelangt.

Nachdem die Ein/Ausgabeoperationen eines "erstarrten" Programms durchgeführt wurden, kann dieses seinen Ablauf wieder aufnehmen. Aber genau wie zu einem bestimmten Zeitpunkt mehrere Programme "erstarrt" sein können, gibt es auch mehrere Programme, die ablaufen können. Das freiwerdende Programm wird darum in eine Warteschleife eingereiht, um zum Laufen zu kommen, wenn es an der Reihe ist. Man sagt, ein solches Programm ist "gestartet".

Das Multiprogramming des GE-55 besteht also in einer Folge von Abläufen "gestarteter" Programme simultan mit den Ein/Ausgaben "erstarrter" Programme.

Außer den "gestarteten" und "erstarrten" Programmen gibt es noch ein oder mehrere "anhaltende" Programme. Das sind Programme, bei denen keine Operationen erfolgen, sondern die darauf warten, von einem anderen Programm gestartet zu werden. Der Halt eines solchen Programms kann durch das gleiche Programm oder durch die Einrichtung einer "Sicherung" (Schutz) erfolgt sein, der dazu bestimmt ist, zu vermeiden, daß Sequenzen, die unvereinbar mit den verschiedenen Programmen sind, gleichzeitig mit ihnen ablaufen. Das ist besonders der Fall, wenn mehrere

Programme gleiche Zonen im Zentralspeicher oder gleiche Randeinheiten, oder zwei am gleichen Kanal angeschlossene Einheiten benutzen. Es ist bekannt aus Abschnitt 4.2.1., daß zwei Einheiten, die an dem gleichen Kanal angeschlossen sind, nicht gleichzeitig von der Zentraleinheit gesteuert werden können. Das bedeutet also, daß nur IØC - Befehle übernommen werden können, die den gleichen Kanal benutzen, wenn sie einzeln in die IØC - Zone eingehen.

Das Zwischenspiel zwischen den Programmen wird durch Spezialbefehle gesteuert, die in Abschnitt 4.3.3. beschrieben werden.

Anmerkung:

Die Gesamtheit der simultan durchgeführten Programme ergibt eine "Verarbeitungseinheit". Die Programme werden numeriert von 1 - 5. Am Anfang der Arbeit sind alle Programme, außer einem, "angehalten". Nur das Programm 1 ist gestartet und kommt unmittelbar zur Durchführung. Dieses Programm muß die anderen im gewünschten Augenblick in der vom Programmierer gewünschten Reihenfolge starten.

4.3.2. Anwendungsweise

Es handelt sich hierbei im wesentlichen um verschiedene Zonen im Zentralspeicher, die von der Zentraleinheit ausgewertet werden, um den reibungslosen Ablauf der gemeinsam im Speicher befindlichen Programme zu steuern.

4.3.2.1. Die Registerzonen

Von den in der Standardregisterzone (Bytes 85-595) befindlichen Registern ist eine gewisse Zahl mit genau definierten Aufgaben für das gerade ablaufende Programm belegt.

Das sind:

- die spezialisierten Register (85-95), die notwendige Angaben zur Steuerung des ablaufenden Programms enthalten: PAR, Vergleichsergebnisse etc..
- die numerischen Register 0-9, in denen die Basisadressen, speziell die für die Mehrfachüberträge, gespeichert sind.

Bei Multiprogramming spielen diese Stellen die gleiche Rolle für das gerade "in Ausführung" befindliche Programm, wie bei Monoprogramming. Wenn aber in der Folge dieses Programm auf einen IØC - Befehl stößt und "erstarrt" und ein anderes Programm "gestartet" wird, nimmt es den Platz des vorherigen ein. Man muß also sowohl den Inhalt dieser Register fortspeichern als auch sie mit den Angaben für dieses neue Programm belegen. Daher braucht man 5 Registerzonen, von denen eine bei Stelle 85 beginnt und die anderen nach Angabe des Programmierers. Jedesmal, wenn ein Programmwechsel erfolgt, wird durch die Zentraleinheit automatisch ein Austausch der zwei entsprechenden Zonen durchgeführt.

Was die numerischen Register betrifft, so können zwei Möglichkeiten vorliegen: Entweder übersteigt die Gesamtzahl aller von den verschiedenen Programmen benutzten Register die Zahl 100 nicht, dann werden die Register einfach auf die einzelnen Programme aufgeteilt, wie auch der Normalspeicherbereich; oder sie übersteigt die Zahl 100, dann muß eine Anzahl Register verschiedenen Programmen dienen und es muß bei jedem Programmwechsel der gleiche Austausch vorgenommen werden, wie auch bei den spezialisierten Registern.

Zusammenfassend dienen also die Registerzonen nicht nur zur Aufnahme der spezialisierten Register, sondern auch für eine gewisse Zahl von numerischen nach Angabe des Programmierers. Diese letzteren bestimmen also die Größe der Registerzonen und es ist klar, daß demnach die Länge der Registerzone die gleiche ist für alle im Speicher befindlichen Programme. Die Registerzone für das in Arbeit befindliche Programm beginnt also an Stelle 85 bis zu einer Stelle, an der sich eine Trennmarke FE befindet, die vom Programmierer gesetzt wird.

Diese Trennmarke begrenzt die Länge des Austausches im Augenblick eines Programmwechsels. Es versteht sich, daß diese Zone so klein wie möglich sein soll, damit die Dauer des Austausches verringert wird.

4.3.2.2. Die Zonen zur Steuerung der Randeinheiten und der Programme

Das Multiprogramming benutzt folgende drei Zonen:

1) Die Zone der Ein/Ausgaben (IØC - Zone - Input - Output-Control von Stelle 31 - 59)

Diese Zone wurde bereits in Abschnitt 4.2.2. beschrieben. Sie steuert im wesentlichen die Simultaneität der Ein- und Ausgaben, und zwar sowohl bei Mono - als auch bei Multiprogramming.

Im folgenden soll daher nur noch eine Beschreibung der letzten 5 Stellen (55 - 59) dieser Zone gegeben werden, deren Aufgabe sich bei Multiprogramming zeigt.

Diese Stellen werden in bezug auf die Programme 1 - 5 belegt. Jedes von Ihnen enthält in seinem rechten Halbbyte die Zahl der im Laufe des Programms verwendeten Randeinheiten zum Abzählen durch das Programm, dessen Byte belegt ist.

In dem Augenblick, wo ein IØC - Befehl in einem Programm "P" zum Zuge kommt, wird die Zahl der durch den Befehl angeschalteten Randeinheiten an der Stelle (54 + P) gespeichert, und zwar im rechten Halbbyte. Jedesmal wenn der Rechner die auf eine dieser Einheiten bezogenen Überträge beendet, wird der Inhalt dieses Halbbytes um 1 vermindert. Wenn der Inhalt "0" erreicht, wird das entsprechende Programm "wieder gestartet". (s. Zone PRC unten).

2) Die Zone der "Registerzonenadressen" (RFA-Register Field Adresses Stelle 60 - 69)

Diese Zone enthält 5 Register zu je 2 Bytes, die entsprechend von rechts nach links für die Programme 1 - 5 belegt werden. Diese Register zeigen in jedem Moment die Adressen an, wohin die besonderen Register jedes Programms übertragen werden.

Nachdem vorher bereits Gesagten, ist eine dieser Adressen 85 und entspricht dem laufenden Programm. Jeder Wechsel der Registerzonen ruft daher die entsprechenden Adressen in der RFA - Zone an.

3) Die Zone der "Programmsteuerung" (PRC-PRograms control, 70-84)

Diese Zone enthält zwei Teile, die durch eine Trennmarke "00" getrennt werden.

a) Der linke Teile der PRC - Zone enthält die Warteschleife der "gestarteten" Programme (max.5). Jedesmal wenn ein Programm gestartet wird, empfängt das erste freie Byte von links:

- in seinem rechten Halbbyte die Nummer des Programms,
- in seinem linken Halbbyte den Wert 1, der anzeigt, daß es sich um das letzte "gestartete" Programm handelt. Das linke Halbbyte des vorhergehenden Bytes wird auf Null gesetzt.

Das Programm, dessen Nummer an Stelle 70 gespeichert ist, ist in Arbeit. Wenn dieses Programm auf einen IØC, STØP oder PRØ - Befehl (s.4.3.3.) trifft und ihm in der Warteschleife ein "gestartetes" Programm folgt, erfolgt Programmwechsel. Die Warteschleife wird um eine Stelle nach links versetzt und das Programm, dessen Nummer sich an Stelle 71 befand, kommt zur Ausführung (Stelle 70). Das "erstarrte" oder "angehaltene" Programm verschwindet aus diesem Teil der PRC - Zone. Die Programme werden so in der Reihenfolge, in der sie "gestartet" wurden, zur Ausführung gebracht.

Anmerkung:

Befindet sich ein arbeitendes Programm allein in der Warteschleife in dem Augenblick, in dem es auf einem IØC-Befehl "erstarrt" (Wert 1 im linken Halbbyte der Stelle 70), wird die Versetzung ausgesetzt bis dieses eine Programm erneut "gestartet" wird, das dann wiederum das Programm an Stelle 70 ist. Diese Besonderheit wird verwendet bei Monoprogramming.

b) Der rechte Teil der PRC - Zone dient der Steuerung des Zusammenwirkens der Programme.

Er nimmt die Zeichen auf, die die Bedingungen dieses Zusammenwirkens darstellen. Diese Zeichen werden nacheinander von rechts nach links in der Reihenfolge, in der sie von den "Halt" oder "Schutz" - Befehlen erstellt werden, eingesetzt (s.4.3.3.)

Wenn man mit "n" das linke Halbbyte und mit "k" das rechte Halbbyte jeder Stelle dieses Teiles der PRC - Zone bezeichnet, werden die Bedingungen nach folgendem Schema dargestellt:

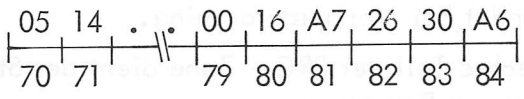
	n	k
Ein Programm "N" wurde durch einen STOP-Befehl angehalten.	Nummer des Programms	0
Ein Programm wurde durch den Schutz "k" geschlossen.	A	Nummer des Schutzes K
Ein Programm "N" wurde durch einen geschlossenen Schutz "k" angehalten	Nummer des Programms N	Nummer des Schutzes K

n = gepackte Dezimalziffer; k = hexadezimales Zeichen

Die Zentraleinheit untersucht diese Zone durch Abtasten von rechts nach links. Die durch den gleichen Schutz "k" angehaltenen Programme werden in der Reihenfolge, in der sie blockiert wurden, wieder gestartet.

Diese Folge von Zeichen wird links durch eine Trennmarke 00 begrenzt. Die Grenze zwischen dem rechten und linken Teil dieser Zone ist jedoch fließend und zwar in Abhängigkeit von den plötzlich eintretenden Bedingungen des Zusammenspiels der Programme. Es kann sogar vorkommen, daß beide Teile dazu neigen, sich zu überschneiden, da die Gesamtlänge der Zone nicht ausreicht. In diesem Fall setzt der letzte Vorfall, der in dieser Zone ein neues Zeichen erzeugt, es an die Stelle der Trennmarke 00, und die Zentraleinheit zeigt Programmunterbrechung an. Das ist immer ein Zeichen für eine schlechte logische Konzeption in der Organisation der Simultaneität der Programme. Der Programmierer muss eingreifen, um die sich daraus ergebenden Änderungen vorzunehmen.

Beispiel:



- das Programm 5 ist in Arbeit,
- das Programm 4 ist gestartet,
- das Programm 3 wurde angehalten,
- der Schutz 6 und 7 ist geschlossen,
- die Programme 1 und 2 sind durch den Schutz 6 blockiert.

4.3.3. Die Befehle

4.3.3.1. Haltbefehle

STØP

STOP program execution

40

Der Befehl STØP, der in einem arbeitendem Programm mit der Nummer "N" steht, bewirkt den Halt des Programmablaufs und einen Programmwechsel.

Ein Zeichen "n0" wird an der ersten freien Stelle von rechts in der PRC-Zone eingesetzt (n - Nr. des Programms). Der linke Teil der PRC-Zone wird um eine Stelle nach links versetzt, wodurch die Nummer "n", die an Stelle 70 stand, verschwindet.

Anmerkungen:

1. Ein durch den Befehl STØP angehaltenes Programm kann nur durch einen Befehl START, der sich in einem anderen Programm befindet, das später in Aktion tritt, wiedergestartet werden (s.4.3.3.2).
2. Der Befehl STØP dient nicht dazu, ein Programm endgültig anzuhalten.
3. Er kann keine Kapazitätsüberschreitung der PRC-Zone hervorrufen.

4.3.3.2. Starten

START

reSTART a program halted by a stop.

41	N

N = Nummer des Programmes, das gestartet wird (01-05)

Der Befehl START, der in einem Programm, dessen Nummer ungleich "N" (im Befehl) ist, startet ein Programm "N", das vorher durch den Befehl STØP angehalten wurde.

Dazu wird der rechte Teil der PRC-Zone untersucht, ob das Zeichen "n0" eingespeichert wurde:

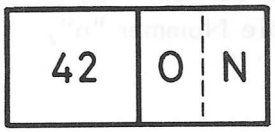
a) wenn Ja : wird dieses Zeichen gelöscht, der rechte Teil der PRC-Zone wird zusammengeschoben und das Zeichen "1n" wird an der ersten freien Stelle von links in diese Zone eingesetzt. Das linke Halbbyte des vorhergehenden Bytes wird auf Null gesetzt.

b) wenn Nein: ist der Befehl START ohne jede Wirkung.

In jedem Fall geht das laufende Programm zum nächsten Befehl über. Dieser Befehl kann keine Kapazitätsüberschreitung der PRC-Zone hervorrufen.

4.3.3.3. Schutz

PRØ PROtect



N = Schutz-Nr. von 1 - F

Trifft ein Programm "N" auf diesen Befehl, wird der rechte Teil der PRC-Zone untersucht, ob ein dem Befehl entsprechendes Zeichen "Ak" eingespeichert wurde.

a) bei Nein: ist dieser Schutz offen .

- das Zeichen "Ak" wird an die erste freie Stelle rechts in der PRC-Zone eingesetzt (k = Nummer des im Befehl genannten Schutzes).
- das Programm "N" geht zum nächsten Befehl über.

b) bei Ja : ist dieser Schutz geschlossen .

- das Zeichen "nk" wird an die erste freie Stelle von rechts in die PRC-Zone eingesetzt (n = Nummer des Programms an Stelle 70; k = Nummer des im Befehl gegebenen Schutzes).
- das Programm "N" wird angehalten : der linke Teil der PRC-Zone wird um eine Stelle nach links versetzt, wodurch die Nummer "n" an Stelle 70 verschwindet und ein Wechsel des Programms erfolgt.

Anmerkungen:

1. Ein durch den Befehl PRØ angehaltenes Programm kann nur durch den Befehl FREE gestartet werden, der sich in einem anderen Programm befindet, das später zur Ausführung gelangt (s.4.3.3.4).
2. Der Befehl PRØ kann im Fall (a) eine Kapazitätsüberschreitung der PRC-Zone und damit eine Programmunterbrechung nach sich ziehen.

4.3.3.4. Freimachung

FREE

FREE

44	O ¦ N

N = Schutz-Nr. von 1 - F

Trifft ein Programm einen Befehl FREE, wird der rechte Teil der PRC-Zone untersucht, ob ein Zeichen "Ak" und ein oder mehrere Zeichen "nk", die dem angegebenen Schutz entsprechen, vorhanden sind. Drei Fälle können auftreten:

a) Keines dieser Zeichen ist vorhanden: Der Schutz mit der Nummer "k" ist offen. Der Befehl FREE hat keine Auswirkung.

b) Nur das Zeichen "Ak" befindet sich in der Zone : Der Schutz "K" ist geschlossen, aber es wird durch ihn kein Programm blockiert. Das Zeichen "Ak" wird gelöscht, wodurch der Schutz "K" geöffnet wird.

c) Das Zeichen "Ak" und ein oder mehrere Zeichen "nk" befinden sich in der Zone: Der Schutz "K" ist geschlossen und blockiert ein oder mehrere Programme.

   - das erste Zeichen "nk" (k = im Befehl angegebene Nummer), das von rechts an angetroffen wird, wird gelöscht und das Zeichen "1n" (n = Nummer, die sich im gelöschten Zeichen "nk" befand) wird an die erste freie Stelle von links in die PRC-Zone eingesetzt (das linke Halbbyte der vorhergehenden Stelle wird auf Null gesetzt). Das drückt sich durch den Start des Programms "N" aus.
   - Das Zeichen "Ak" und die anderen evtl. Zeichen "nk" bleiben erhalten.

In allen Fällen geht das Programm zum nächsten Befehl über.

Anmerkung:

Die Fälle (b) und (c) ziehen ein Verschieben des rechten Teils der PRC-Zone nach sich. Der Befehl ruft keine Kapazitätsüberschreitung hervor.

Die beiden Befehle PRØ und FREE arbeiten analog dem Blocksystem der Eisenbahn. Der Befehl PRØ erzeugt Zeichen "Ak" oder "nk" (rotes Licht) bei jeder Ausführung und der Befehl FREE löscht jedesmal nur ein Zeichen (grünes Licht). Das bedeutet, daß jede Überschreitung eines PRØ, der einen Schutz "K" betrifft ein Übergang auf einen FREE entspricht, der den gleichen Schutz betrifft, wenn man nicht eine allgemeine Blockade des Programms riskieren will.

4.3.4. Beispiele

Um die Funktion des Multiprogramming zusammenzufassen, wird ein aussagefähiges Beispiel gegeben. Ausgangspunkt der angenommenen Situation ist das Ende des Abschnitts 4.3.2. Die PRC-Zone enthält:

05	14	. . . .	00	16	A7	26	30	A6
70	71		79	80	81	82	83	84

1. Das Programm 5 trifft den Befehl FREE 6

   Das erste durch den Schutz 6 angehaltene Programm (Programm 2) wird gestartet.

05	04	12	. . . .	00	16	A7	30	A6
70	71	72		80	81	82	83	84

2. Das Programm 5 trifft den Befehl START 3

   Das Programm 3 wird gestartet.

05	04	02	13	. . . .	00	16	A7	A6
70	71	72	73		81	82	83	84

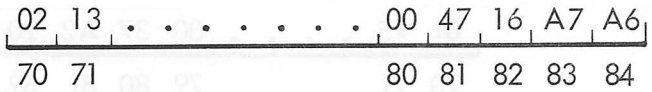
3. Das Programm 5 trifft einen IØC-Befehl

   Es erstarrt während der Durchführung dieses Befehls. Das Programm 4 wird zur Ausführung gebracht.

04	02	13	. . . . .	00	16	A7	A6
70	71	72		81	82	83	84

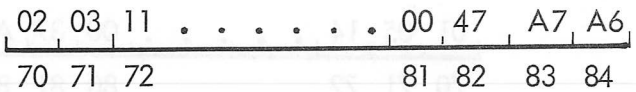
4. Das Programm 4 trifft einen Befehl PRØ 7

   Der Schutz 7 ist geschlossen. Man geht zu Programm 2 über.



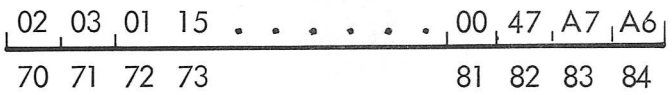
5. Das Programm 2 trifft einen Befehl FREE 6

   Das durch den Schutz 6 blockierte Programm1 wird gestartet.



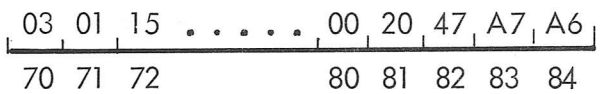
6. Der IØC-Befehl des Programms 5 ist beendet.

   Das Programm 5 wird wiedergestartet.



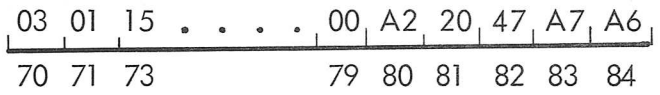
7. Das Programm 2 trifft einen Befehl STØP

   Es wird angehalten und Programm 3 fährt fort.



8. Das Programm 3 trifft einen Befehl PRØ 2

   Der Schutz 2 war "offen", es wird fortgefahren, nachdem er geschlossen wurde.



9. Das Programm 3 trifft einen Befehl FREE 6

10. Das Programm 3 trifft einen Befehl PRØ 7

Es wird blockiert. Es wird mit Programm 1 fortgefahren.

01	15	. . . .	00	37	A2	20	47	A7
70	71		79	80	81	82	83	84

11. Das Programm 1 trifft einen Befehl FREE 7

Das Programm 4 wird entblockiert und gestartet.

01	05	14	. . . . .	00	37	A2	20	A7
70	71	72		80	81	82	83	84

Diese Folge von Vorfällen in einem aussagefähigen Beispiel erläutert alle Fälle, die bei Anwendung des Multiprogramming beim GAMMA 55 auftreten können.

5. Die Bedienung des GE-55

5.1. Einführung

Um den GE-55 in Betrieb zu nehmen, ist eine Reihe von Programmen erforderlich, die in den verschiedenen Phasen zur Anwendung kommen. Diese Programme bilden "Systeme", die in den folgenden Abschnitten beschrieben werden sollen. Außerdem werden die Richtlinien zur Erstellung und Anwendung der Arbeitsprogramme gegeben.

5.2. Die Software des GE-55

Die Methoden zur Benutzung eines GE-55 richten sich nach den angeschlossenen Randeinheiten. Man unterscheidet darum zwei Methoden, je nachdem, ob der GE-55 mit oder ohne Hilfsspeicher (Magnettrommel) ausgestattet ist.

Außerdem unterscheidet man noch zwei Elemente, ohne die der Rechner nicht arbeiten kann:

- die Programme, die seine Arbeit steuern;
- die Dateien, die verarbeitet werden sollen.

5.2.1. Lochkartenanlage GE-55 (ohne Trommel)

Bei dieser Konfiguration werden die durchzuführenden Programme und zu verarbeitenden Dateien obligatorisch durch Karten in den Kernspeicher eingegeben. Daher rührt auch der Name dieser Art der Konfiguration.

Um eine Arbeit zu beginnen, müssen die Parameterkarten und anschließend die Datenkarten in den Kartenleser eingelegt werden. Das Programm wird nun geladen, gestartet und die Daten in der Reihenfolge verarbeitet, wie sie gelesen werden. Um mehrere Arbeiten aneinanderzureihen, gibt man in der gleichen Art die entsprechenden Programme und Dateien ein.

Während die Dateien, die Karte für Karte gelesen werden, einen unbegrenzten Umfang haben können, ist der Umfang eines Programmes, das in Zentralspeicher gespeichert wird, begrenzt durch die Kapazität dieses Speichers. Das gleiche gilt für bestimmte Daten, Tabellen etc. die am Anfang der Arbeit eingespeichert werden, und auf die man ständig Zugriff hat.

5.2.2. GE-55 mit Trommel

Die Magnettrommel bietet eine zusätzliche Speicherkapazität, die bei Anschluß von zwei Trommeln verdoppelt werden kann. Es ist durchaus möglich, Programme auf die Trommel zu übertragen, anstatt sie im Kernspeicher zu speichern und jeweils nur den Teil in den Zentralspeicher abzurufen, der gerade benötigt wird.

Man kann also ein Programm in mehrere "Segmente" unterteilen, die nacheinander in derselben Zone des Zentralspeichers stehen, und zwar in der erforderlichen Reihenfolge. Diesen Vorgang nennt man "Segmentierung".

Das ist aber nicht der ganze Vorteil der Trommel. So kann man gewisse Hilfsoperationen übergehen und ihre Durchführung erleichtern, indem man sie am Anfang einer Arbeitsperiode, z.B. täglich, auf der Trommel speichert gemeinsam mit den durchzuführenden Programmen und den permanenten Dateien einschließlich der Dateien mit Direktzugriff, die sehr wichtig sein können. Man erreicht so eine Beschleunigung der Ladezeit für Programme und Daten, die man mehrmals innerhalb einer Arbeitsperiode benötigt.

Um diese Arbeiten durchzuführen, ist es nur noch erforderlich, die durchzuführenden Programme dem Rechner anzuzeigen und sie in den Zentralspeicher zu übertragen, wo sie dann zur Ausführung gelangen. Der Bedienungskraft obliegt es also nur noch, neue Werte einzugeben und die Resultate in Empfang zu nehmen.

Zusammenfassung:

Aufgrund dieser beiden schematisch dargestellten Anwendungsmethoden muß auch die Software den Besonderheiten einer jeden von ihnen Rechnung tragen, um die Forderungen des Benutzers zu erfüllen. Sie muß daher:

- sich nach den beiden Konfigurationen der Maschine richten;
- bei jeder Konfiguration eine Hilfe bieten, die es ermöglicht, jede Stufe erreichen zu können und dadurch den Programmierer zu entlasten.

5.2.2.1. Die verschiedenen Systeme

Den zwei oben beschriebenen Konfigurationen entsprechen auch zwei Arten der Software, das "Kartensystem" und das "Trommelsystem". Von der Funktion her gesehen unterteilen sich diese wiederum in zwei Kategorien:

- Die "Programmiersysteme" dienen zur Vereinfachung der Erstellung von spezifischen Programmen, und zwar vom Schreiben des Programms ab bis zur endgültigen Fertigstellung in Maschinensprache.

- Die "Steuerungssysteme" steuern entweder die Durchführung der Programme in Maschinensprache oder - im Rahmen der zusätzlichen Speicher - die Organisation und die Auswertung der Dateien nach den vorher festgelegten Methoden.

Diese Systeme werden außerdem noch durch eine Bibliothek von Hilfs- und Standardprogrammen ergänzt, die dem Programmierer das Schreiben von Hilfsprogrammen zur Steuerung der Programme, der Dateien und deren Auswertung abnehmen.

Die Karten- und Trommelsysteme haben gemeinsame Berührungspunkte, die im folgenden zusammen beschrieben werden. Abweichungen werden jeweils angezeigt.

5.2.2.2. Die Programmiersysteme

Die Programmiersysteme bieten dem Benutzer

1. die Programmiersprachen, und zwar die Grundsprachen und die Umwandlungssprache für die Trommel (siehe Kapitel 3). Diese beiden Sprachen sind leichter anzuwenden als die Maschinensprache, bieten eine größere Sicherheit und enthalten Makrobefehle.

2. ein Übersetz- und Kontrollsystem, daß
   - die in der Grundsprache abgefaßten Programme übersetzt,
   - Aufbau und Wahrscheinlichkeit der kodierten Befehle prüft,
   - die Programme in die Maschinensprache bringt und sie listet.

Beim GE-55 mit Trommel kommt zu diesem Übersetz- und Kontrollsystem noch ein Umwandlungssystem hinzu, das
- die zusätzliche Trommelsprache übersetzt,
- die verschiedenen Teile des Source-Programms (Sektionen, Unterprogramme usw.) umwandelt, indem es ihnen die effektiven Adressen im Speicher und auf der Trommel zuordnet und die notwendigen Verbindungen herstellt,
- die erhaltenen Programme in Segmente für die Verarbeitung aufteilt,
- die für die Programme erforderlichen Felder im Kernspeicher und auf der Trommel reserviert.

3. ein Listprogramm, das
   - beim Testen von Programmen Programmlisten erstellt, um Fehlersuche und -korrekturen zu erleichtern;
   - eine endgültige Programmliste nach dem Test, die zu den Programmunterlagen gehört, erstellt;
   - das Erstellen von Parameterkarten für das Programm "Verdichten" ermöglicht.

4. ein Programm "Verdichten", das
   die getesteten Programme "verdichtet", um die Anzahl von Programmkarten zu verringern und so die Ladezeiten jeder Arbeit zu optimieren.

5.2.2.3. Die Betriebssysteme

Dem Benutzer stehen zur Verfügung:

Zum Fahren der Programme

Bei einer GE-55 Kartenanlage

- ein Ladeprogramm Kernspeicher, das
  . die Verbindungszonen des verdrahteten Supervisors initialisiert (diese Zonen befinden sich am Anfang des Speichers) und den Kernspeicher evtl. ab Byte 161 löscht,
  . das Programm "Unterbrechung" am Ende des Kernspeichers lädt,
  . entweder das zu testende Source-Programm oder das auszuführende Teilprogramm in verdichteter Form in den Kernspeicher lädt,
  . bei einem Source-Programm die symbolischen Adressen in absolute Adressen umwandelt (siehe 3.2.),
  . das Source-Programm oder Teilprogramm startet an der Anfangsadresse, die in einer Befehlskarte angegeben ist.

bei einem GE-55 mit Trommel

- ein Monitor-Programm, das von der Trommel abgerufen wird und
  . das Programm "Supervisor TB" in den Kernspeicher lädt,
  . die Verbindungszone (erweitert) des verdrahteten Supervisors und des Supervisors TB initialisiert,
  . entweder das zu testende Programm oder das auszuführende Teilprogramm in den Kernspeicher lädt aufgrund des Kernspeicherladeprogramms, das zu einem Programm-Modul des Monitor-Programms wird.
- das Programm startet.

Anmerkung:
Das Kernspeicher-Ladeprogramm und der Monitor sorgen für die Betriebsfunktionen zwischen zwei Arbeiten, um sie aufeinander folgen zu lassen. Zu diesem Zweck werden sie in den Speicher geladen, sobald eine Arbeit beendet ist, und verschwinden, wenn die nächste Arbeit fertig zur Ausführung ist.

Bei einer GE-55 Kartenanlage

- ein "Unterbrechungsprogramm" P.G.I., das dem Bediener Standardverfahren zur Behebung von Störungen oder zum Informationsaustausch mit dem ablaufenden Programm liefert.

  Dieses Programm wird vom Bediener wie unter 2.1.6. beschrieben in Betrieb gebracht. Es kann aufgrund angegebener Anweisungen folgende Funktionen ausführen:

  . das Lochen von Informationen wiederholen, die sich noch in der Stanzzone befinden,

  . eine Karte noch einmal lesen,

  . über die numerische Tastatur eine verschlüsselte Nachricht für das laufende Programm eingeben,

  . ein spezifisches Wiederanlaufprogramm zu starten, das evtl. unter mehreren solcher Programme ausgewählt wird,

  . vor allem das laufende Programm dort zum Wiederanlauf bringen, wo es unterbrochen worden ist.

  Das P.G.I. kann ebenfalls auf der Sichtanzeige eine verschlüsselte Nachricht vom laufenden Programm anzeigen.

GE-55 mit Trommel

- ein Programm "Supervisor TB" (TB-Trommel)

  . verarbeitet die Eingabe und Ausgabe der Trommel physikalisch, d.h. steuert (kontrolliert) die Lese/Schreiboperationen,

  . verarbeitet Störungen auf den Randeinheiten und Programmunterbrechungen mittels eines P.G.I.-Moduls,

  . verarbeitet die Anrufe von Segmenten eines gleichen Programms: Suche auf der Trommel, Laden in den Kernspeicher.

  . Ruft schließlich das Monitor-Programm in den Kernspeicher, wenn eine Arbeit beendet ist, um die nächste anschließen zu können.

Anmerkung:

Das P.G.I. und der Subvisor TB befinden sich während des Arbeitsablaufes im Kernspeicher, um den Arbeitsablauf zu steuern.

GE-55 mit Trommel

- ein Ladeprogramm Trommel, das das oder die auszuführenden Programme in verdichteter Form auf die Trommel lädt und evtl. Trommelzonen für die Dateien reserviert. Diese Zonen werden anschließend vom Supervisor TB verarbeitet.

Zur Datenverarbeitung auf der Trommel

1. ein Programm "Dateienspeicherung", das die Dateien auf die Trommel speichert. Es gibt zwei Speicherungsverfahren:
   - sequentiell verkettet, d.h. jeder Satz auf der Trommel enthält die Adresse des nächstfolgenden Satzes,
   - indiziert sequentiell, d.h. die Sätze werden der Reihe nach nach ihren Kennzeichen gespeichert und können anhand einer Tabelle gesucht werden.
2. ein Programm "Dateifortschreibung", das es ermöglicht, in einer auf der Trommel gespeicherten Datei Sätze zu löschen, zu ersetzen oder hinzuzufügen. Die Datei kann wie oben angegeben gespeichert sein.
3. ein Programm "Eingabe und Ausgabe ausgewählter Informationen" das es ermöglicht, Teile einer Datei oder Sätze, die sich auf der Trommel befinden, in Karten zu lochen oder auszudrucken, um sie extern zu bearbeiten oder aus Sicherheitsgründen später wieder einzugeben.

5.2.3. Bibliothek der Service- und Standardverarbeitungsprogramme

Die Service-Programme, die die Programmier- und Betriebssysteme vervollständigen, sind in den vorangehenden Punkten beschrieben worden. Folgende Programme können noch dazugezählt werden:

- Erstellen Kartendateien
- Standardprogramm doppeln, Listen, Lochen
- Listen
- Bearbeitung von Kartendateien

Bei den Standard-Bearbeitungsprogrammen handelt es sich im wesentlichen um spezifische Unterprogramme:

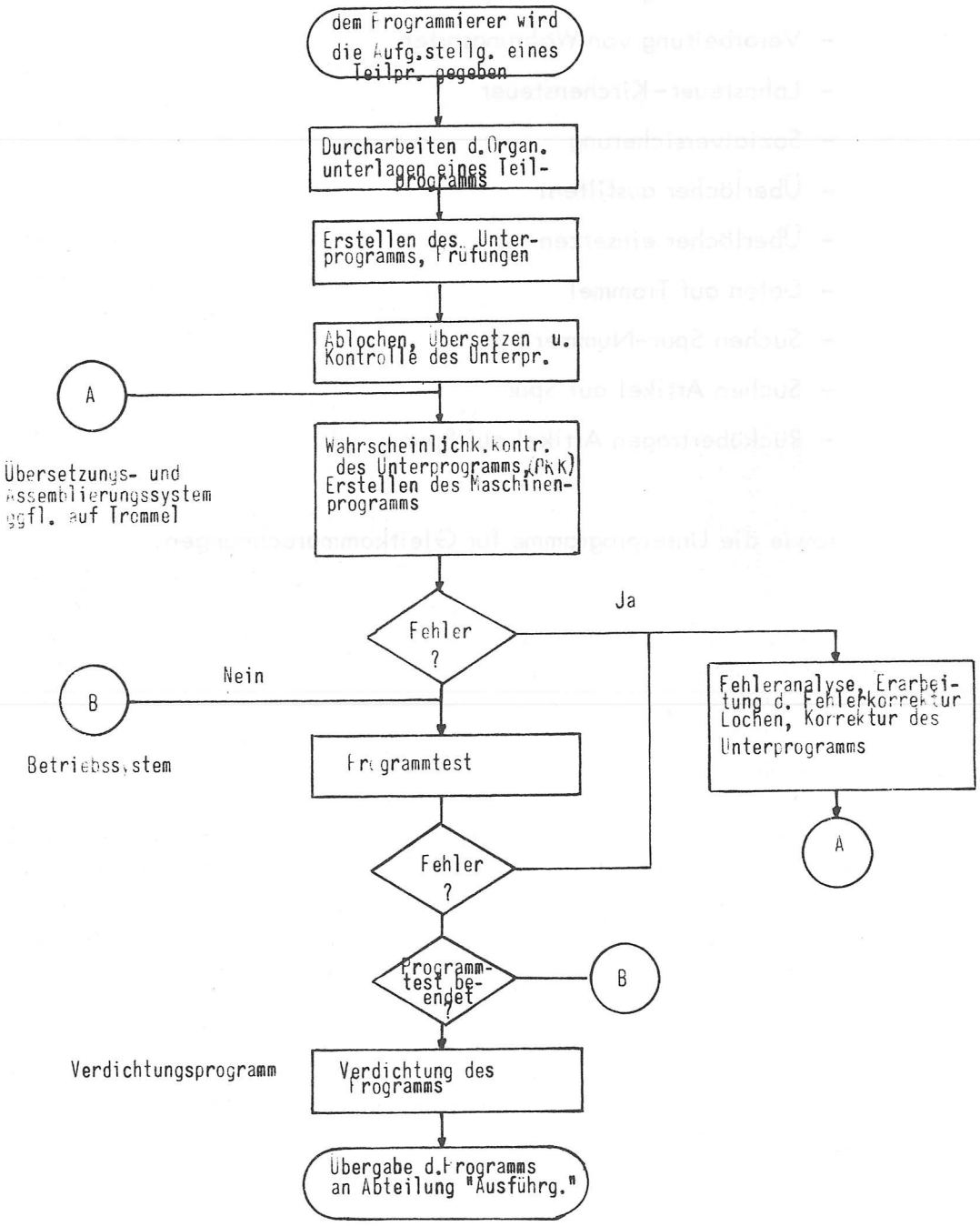
- Division
- Kennzeichenkontrolle
- Verarbeitung von Überweisungen (franz. System)
- Verarbeitung von Daten (Datum)
- Verarbeitung von Währungsarten
- Lohnsteuer-Kirchensteuer
- Sozialversicherung
- Überlöcher ausfiltern
- Überlöcher einsetzen
- Daten auf Trommel
- Suchen Spur-Nummer
- Suchen Artikel auf Spur
- Rückübertragen Artikel auf Spur

sowie die Unterprogramme für Gleitkommarechnungen.

5.3. Erstellen eines Programms

5.3.1. Verfahren

Die verschiedenen Operationen eines funktionsfähigen Programms gehen folgendermaßen ineinander über:



5.3.2. Erstellen der Unterprogramme

5.3.2.1. Untersuchung und Organisation eines Teilprogramms

Vor dem Erstellen eines Programms untersucht der Programmierer die Spezifikationen, die in der Aufgabenstellung des jeweiligen Teilprogramms enthalten sind. Diese Spezifikationen beziehen sich auf die zu verarbeitenden Daten, die erwarteten Ergebnisse und die auszuführenden Verarbeitungen. Die Spezifikationen stellen den ersten Teil der Programmunterlagen dar.

Der Programmierer untersucht dann die Organisation dieser Verarbeitungen auf der Maschine und erstellt dabei folgende Programmunterlagen:

- Kernspeicherbelegung

- In diesem Schema sind die zu reservierenden Speicherzonen (Ein-/Ausgabe, Arbeitsfelder usw.), die Zuordnung der numerischen Register, d.h. also die Kernspeicherbelegung der Daten und Ergebnisse und die verbleibende Kapazität für das Programm angegeben.

- Ein detailliertes Flußdiagramm.

Dieses Flußdiagramm ermöglicht:

- Das Programm evtl. in mehrere Teilprogramme zu unterteilen, die in Multiprogramming laufen, und diese Programme in Sektionen aufzuteilen, die mit symbolischen Kennzeichen versehen sind. Diese Symbole werden vom Programmierer sofort festgelegt.

- Die Sektionen zu definieren, die als Unterprogramme aufgebaut werden können.

- Im Rahmen der Trommelprogrammierung diese Sektionen herauszulösen, die unabhängige Segmente darstellen können (z.B. die zu Beginn des Programms benutzten Sektionen können nach ihrer Ausführung durch Wiederholungssektionen im Kernspeicher überlagert werden), oder solche Sektionen, die über zwei Segmente gelegt werden können, die im Kernspeicher überlagert werden können.

  Vor der Erstellung dieser Unterlagen hat der Programmierer die Normen und Einsatzbedingungen des betreffenden Betriebssystems zur Kenntnis genommen, um:

- die erforderliche Speicherkapazität für dieses System zu berücksichtigen (PGI. Supervisor, TB),

- die Standard-Zuordnung für Ein-/Ausgabefelder und Register zu berücksichtigen,

- zu berücksichtigen, welche Initialisationen beim Laden des Programms in den Kernspeicher vorgenommen werden, die nicht programmiert werden (Initialisation von Registern und Ein-/Ausgabefeldern),und diejenigen, die er vorsehen muß.

- Die Standard-Wiederanlaufverfahren nach Fehlern, um vor allem die spezifischen Wiederanlaufverfahren zu verkürzen, die aufgrund der Art der Verarbeitung und der Dauer programmiert werden müssen.

- Das Ladeverfahren von Segmenten, die von der Trommel kommen, um die bestmögliche Aufteilung festzulegen unter Berücksichtigung des Verhältnisses verfügbare Kernspeicherkapazität / voraussichtliche Ladezeit.

- Die Aufbaunormen dieser Segmente.

Der Programmierer muß außerdem die Standard-Unterprogramme kennen, um sie sinnvoll einzusetzen, und deren Einsatzbedingungen, die reservierten Symbole usw. kennen.

Symbolische Kennzeichnung der Sektionen

In der Basis-Programmiersprache stehen dem Programmierer 240 verschiedene Kennzeichnungen zur Verfügung, von denen er ggfl. die für die benutzten Software-Programme reservierten Bezeichnungen abziehen muß.

Was die Anwendung der Kennzeichnung betrifft, ist es gemäß den besonderen Kriterien in jedem Programm vorzuziehen, einheitliche Normen für alle Programme festzulegen, und die symbolischen Kennbezeichnungen anzusehen als ein praktisches Mittel zur Untersuchung und zur Verbindung zwischen dem Flußdiagramm und dem zugehörigen Programm. Zu diesem Zweck ordnet man die Kennzeichnung aufsteigend in der gleichen Ordnung wie im Flußdiagramm. Die Kennzeichnung wird aus zwei Zeichen gebildet: Das erste kann zum Beispiel die Seitennummer des Flußdiagramms sein und das zweite die laufende Nummer der Sektion auf der Seite. Auf diese Weise kann man 15 Seiten und 16 Sektionen pro Programm unterscheiden.

5.3.2.2. Ausarbeitung des Programms

Ist das Flußdiagramm detailliert aufgestellt, kann der Programmierer die Ausarbeitung des Programms (oder der Programme) in Angriff nehmen. Dazu benutzt er Programmformulare, die sowohl für die Basissprache als auch für die Assembler-Sprache in der Trommel gültig sind.
Die ausgefüllten Programmformulare dienen als schriftliche Grundlage für die Ablochung der Programme. Jede Zeile entspricht einer Karte von 80 Spalten.

Das Programmformular

In 8.4. wird ein Beispiel für ein Programmformular gezeigt. Bei der Programmierung ist dem Aufbau dieses Formulars genau zu entsprechen. Die einzelnen Angaben sind nur in die zugehörigen Spalten einzusetzen.

Die Rubriken

Der Programmierer hat die Möglichkeit, in die einzelnen Zeilen neben den Befehlen zusätzliche Bemerkungen einzusetzen, um den Verlauf des Programms zu verdeutlichen.

Rubrik Kartenart (Spalte 2 und 3).
Diese Kartenart gibt an, wie der in der Zeile enthaltene Befehl verarbeitet werden muß.

Das erste Zeichen (Spalte 2) zeigt an, wo der Befehl gespeichert werden muß.

Beispiele:

Der Schlüssel 1 zeigt an, daß die Befehle der Reihe nach von der Anfangsadresse an gespeichert werden. Die Adresse wird durch den Pseudo-Befehl BEGIN definiert (Elementarbefehle der Basissprache etc.).

Der Schlüssel 3 zeigt an, daß der Befehl an einer absoluten Adresse zu speichern ist. Diese wird vom Programmierer angegeben.

Das zweite Zeichen (Spalte 3) gibt Art und Größe des Befehls an.

Beispiele:

Der Schlüssel 1 gibt an, daß der Befehl in entpackter Form anzuordnen ist, d.h. daß jedes Zeichen ein ganzes Byte belegt (alphanumerische Konstante).

Der Schlüssel 2 gibt an, daß der Befehl in gepackter Form gespeichert wird, d.h. daß jedes Zeichen ein Halbbyte belegt (Befehlsparameter, hexadezimale Konstanten).

Rubrik Zahl der Bytes (Spalte 4 und 5)
Der Programmierer gibt in dieser Rubrik die Anzahl der Bytes an, die durch die Parameter der Befehle ab Spalte 17 belegt werden (Befehle, Konstanten). Eventuelle Kommentare sind in dieser Anzahl nicht enthalten.
Diese Zahl begründet einen der Parameter der Pseudo-Befehle der Konstantenerstellung (siehe 3.9.2.).

Rubrik Ordnungsnummer (Spalte 6 bis 9)

Diese Nummer setzt sich aus 4 Dezimalstellen zusammen und legt die Reihenfolge der Programmkarten fest.

Während der Anfangsausarbeitung eines Programms ist es empfehlenswert, die äußerste rechte Ziffer nicht zu benutzen und ihr systematisch den Wert 0 zu geben. Dieses erleichtert die späteren Modifikationen des Programms und erlaubt das Einsetzen von 1 bis 9 Zeilen zwischen zwei ursprünglich auf dem Formular vorhandenen Zeilen. Es ist nicht erforderlich, die bereits vorhandenen Nummern zu ändern; die neuen Linien werden dann beziffert von xxx1 bis xxx9.

Bemerkung

Der Programmierer beziffert nicht die Pseudo-Befehle der Konstantenerstellung, die sich in einer beliebigen Reihenfolge befinden können. Er vermerkt vielmehr in den Spalten 6 bis 9 einen der Parameter dieser Pseudo-Befehle, nämlich die Adresse der durch die spezifischen Konstanten belegten Zone in der betreffenden Linie.

Der Befehl

Rubrik Symbolischer Operationstyp (Spalte 10 bis 16)

Die Buchstaben und Ziffern, die zum symbolischen Operationstyp zusammengefügt sind, werden von Spalte 10 an ohne Zwischenraum geschrieben. Die rechts nicht gebrauchten Spalten bleiben frei.

Rubrik Befehl (Spalte 17 bis 44)

Die Parameter werden von Spalte 17 an nacheinander ohne Zwischenraum geschrieben auf der Grundlage eines alphanumerischen oder hexadezimalen Zeichens pro Spalte. Die rechts unbenutzten Spalten bleiben frei. Beim Schreiben der Parameter sind die Formvorschriften und Kodierungsregeln, die in den Kapiteln 3 und 4 angegeben sind, genau einzuhalten.

Was indessen die Grundbefehle betrifft, sind die Spalten 17 und 18 reserviert. Der Programmierer kann hier dem hexadezimalen Operationstyp angeben.

Die Kommentare (Spalten 34 oder 46 bis 80)

Die Kommentare sind Informationen in Klartext, um die Sektionen, Sequenzen oder Befehle zu erläutern.

Das Programmiersystem führt keine Bearbeitung des Kommentartextes durch. Die Kommentare werden nur auf den Programmlisten wiedergegeben.

Der Programmierer kann sowohl Befehlslinien durch Kommentare vervollständigen als auch zwischen den Befehlslinien besondere Kommentarzeilen einfügen. Im ersten Falle können die Kommentare in Spalte 46, im zweiten Falle in Spalte 34 begonnen werden.

Der Programmierer kann zu diesem Zweck alle druckbaren Zeichen gebrauchen und so viele Abstände hinzufügen, wie er benötigt. Die Kommentare können eine beliebige Stelle innerhalb der zugeordneten Zonen belegen.

Bemerkungen:

Wird ein Kommentar in eine Zeile gesetzt, in der auch ein Befehl enthalten ist, muß die Spalte 45 frei bleiben. In einer Zeile, die nur einen Kommentar enthält, müssen die Spalten 10 bis 33 frei bleiben.

Kommentare können nach den absoluten spezifischen Werten eines Pseudo-Befehls der Konstantenerstellung geschrieben werden. Es ist indessen dringend erforderlich, sie durch Abstände zu trennen.

Kennzeichen (Spalte x bis 80)

Es ist empfehlenswert, die letzten Spalten jeder Linie zu reservieren, um hier das Kennzeichen des Programms zu vermerken (Nummer, Code, etc.). Dieses Kennzeichen wird in alle Karten eines bestimmten Programms wiederholt.

Es erlaubt auf einfache Art, eine falsch liegende Karte einzuordnen. Diese Rubrik ist auf dem Programm nicht festgelegt, damit der Programmierer die Zahl der Spalten reservieren kann, die dafür unbedingt nötig sind. Er benutzt sie, als ob es sich um einen Kommentar handelte.

Art der Zeilen

	Auszufüllende Rubriken					
Art der Zeile	K A 2/3	Zahl Bytes 4/5 D	Ord.N 6-9 D	Symb.OT 10-16 H oder AN	Befehle 17-44 H oder AN	Kommentare und Kennz. AN
Elementar-Befehle	12	ja	ja	ja (a)	ja (a) hexad.OT wahlweise (b)	evtl. Sp.46-80 (d)
Pseudo-Befehle						
BEGIN	15	nein	ja	ja (a)	ja gem.Befehl (a)	d
END	17	01				
LEVEL	14	02				
NØP	12	01				
HEXAD	32	ja	(c)			
AN	31	ja	(c)			
Kommentar	12	nein	ja	nein	nein	ja von Sp.34 an (d)

Bemerkungen:

a) Linksbündig nicht benutzte Spalten bleiben rechts frei
b) Die Spalten 17 und 18 bleiben frei, wenn der hexadezimale O.T. nicht angegeben wird.
c) Enthält eine Rubrik einen Parameter, ist die Adresse dieses Zeichens in Spalte 17 angegeben.
d) Beliebige Form

Abkürzungen:

D = dezimal
AN = alphanumerisch
H = hexadezimal

Ratschläge für die Ausarbeitung von Programmen

Um ein Programm auszuarbeiten, muß der Programmierer die Programmiersprache beherrschen. Unklarheiten können mit der vorliegenden Unterlage oder mit der Unterlage über die Programmiersprache für die Magnettrommel beseitigt werden. Die Ausarbeitung wird enorm vereinfacht, wenn der Programmierer sich die Mühe macht, sich ein zusammenfassendes Merkblatt anzulegen, das seinen Bedürfnissen angepaßt ist und das die Formate und besonderen Regeln für die Kodierung in eine gedrängte und schematisierte Form bringt.

Er muß außerdem die Normen kennen, die für die Benutzung der Software erforderlich sind, z.B., Anrufsnormen der Unterprogramme, maximale Kapazität der Segmente etc.

Im allgemeinen enthält ein Programm einen ersten Teil, dessen Aufgabe es ist, die Zentraleinheit zu "initialisieren".

Diese Initialisierung besteht darin, vor der Arbeit die benötigten Informationen in den Kernspeicher einzulesen und evtl. noch erforderliche Programmteile anzurufen

Zur Erstellung eines Programms ist in "umgekehrter" Reihenfolge vorzugehen, d.h. am besten wird zunächst der Programmablauf fixiert. Dabei werden Anmerkungen vorgenommen, die Reservierungen von Feldern bzw. das Anlegen von Konstanten betreffen. Ist der Programmablauf festgelegt, können dann aufgrund der besseren Übersicht die entsprechenden Definitionen einfach und leicht erstellt werden.

Ratschläge für die Ausarbeitung

- In jedem Programm wird die erste Linie für den Pseude-Befehl "BEGIN" reserviert. Die in jedem Befehl enthaltene Adresse kann im Augenblick des Durchlaufs durch die Maschine festgelegt werden.

- Die großen Unterteilungen des Programms (Programme, Segmente, Unterprogramme, wichtige Sektionen, Vorprogramme) werden auf getrennte Formulare geschrieben.

- Jedem Programmteil ist eine Serie von laufenden Nummern zuzuordnen, wobei darauf zu achten ist, daß möglichst weite Abstände zwischen den einzelnen Nummern bestehen bleiben. Dadurch wird vermieden, daß die gesamte Numerierung neu vorgenommen werden muß, wenn eventuelle Modifikationen in bereits bestehende Serien eingefügt werden.

- Die Programme sind mit Hilfe zahlreicher Kommentare zu erläutern.

- Der Pseudo-Befehl "END" am Ende eines jeden Programms darf nicht vergessen werden.

Technische Ratschläge

- Übertragungen von Operanden von Rechenoperationen:
  Vorzugsweise sind Mehrfachbefehle zu verwenden. Mehr zusammenfassende Funktionen, weniger platzraubende Befehle, automatische Verarbeitung der in den Ziffern enthaltenen Vorzeichen, richtiges Löschen der Register.

- Löschen der Register:
  Um ein Einfach- oder Doppelregister zu löschen, wird eine Registeroperation verwendet; um mehr als 4 Register zu löschen, benutze man den Befehl "INC", der leistungsfähiger ist.

- Initialisationen:
  Man benutze vorzugsweise die Pseudo-Befehle der Konstantenerstellung, was den Vorteil hat, daß sie während des Programmladens ausgeführt werden und dann nicht mehr vorhanden sind. Dadurch wird eine unnötige Speicherbelegung vermieden.

Kontrolle

Bevor die Lochung auf der Karte vorgenommen ist, empfielt es sich, die folgenden Punkte zu prüfen, die Gelegenheit zu häufigen Fehlern bieten:

- Symbolische Kennzeichnungen
  In Sprungbefehlen geht "F1" voran, in den Pseudo-Befehlen LEVEL "F2".

- Pseudo-Befehle
  Kein hexadezimaler Operations-Typ. Die Parameter beginnen in Spalte 17.

- Verschiedene Übertragungen
  Mehrfach-Übertragungen von 3 Ziffern, denen der Buchstabe A vorangeht.

- Befehl INC
  Die zuerst angegebene Adresse > als die andere

- Multiplikation
  Der Multiplikand, der weniger als 10 Ziffern enthält, ist in doppelter Länge zu übertragen, um das linke Register zu löschen.

- Verschieben
  in einem einfachen Register, wobei das vorhergehende (linke) mit benutzt wird.

- Löschen der Register
  Einsetzen von 0.

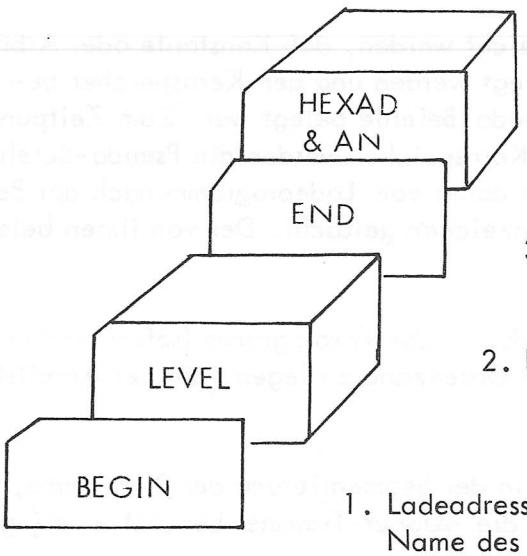
5.3.3. Erstellen und Test von Maschinenprogrammen

Die einmal geschriebenen Programme sind auf Programmkarten abgelocht, deren Bild dem Programmformular entspricht. Die gelochten Karten werden im Verhältnis von einer Karte pro Zeile interpretiert und kontrolliert. Danach kann der Programmierer die Aufstellung und das Testen des Maschinenprogramms vornehmen.

5.3.3.1. Kontrolle, Übersetzung, Assemblierung der Sourceprogramme

Die erste Operation in der Maschine besteht aus der Kontrolle des Programmkarten-Pakets, wobei der formelle und logische Aufbau der durch den Programmierer geschriebenen Befehle überprüft und mittels der Standardprogramme, die in dem Programmiersystem enthalten sind.

Die zu überprüfenden Sourceprogramme werden auf folgende Weise zusammengesetzt:



4. Sektion für die Initialisierung (Konstanten)

3. Ende des eigentlichen Programms

2. Programmsektion (Befehle)

1. Ladeadresse der Programmsektionen und Name des Programms (als Kommentar)

Enthält ein Programm mehrere Einzelprogramme, so wird jedes einzelne von ihnen wie oben gezeigt, zusammengesetzt. Danach werden sie getrennt kontrolliert, und zwar in der Reihenfolge, in der sie in den Kernspeicher geladen werden. Vor jedem Durchlauf muß der Programmierer die Anfangsadresse des Einzelprogramms in die Karte BEGIN lochen. Es ist dabei darauf zu achten, daß evtl. reservierte Zonen bzw. Zonen für vorhergehende Programme frei bleiben.

Das Kontrollprogramm erstellt eine Liste der Programmkarten, wobei die tatsächlichen Kernspeicheradressen der Elementarbefehle mit ausgedruckt werden. Evtl. entdeckte Form- oder Kodierungsfehler werden ebenfalls auf der Liste angezeigt.

Der Programmierer korrigiert anhand dieser "Kontroll-Liste" evtl. vorhandene Fehler.

Bemerkung:

Bei der Prüfung des Kernspeicherbedarfs für ein Programm ist darauf zu achten, daß die Pseudo-Befehle LEVEL 2N Bytes belegen (N = Anzahl der Pseudo-Befehle). Diese Belegung erscheint nicht auf der Druckliste des Kontrollprogramms. Die Pseudo-Befehle werden nur beim Laden des Programms mit Hilfe des Ladeprogramms gelöscht. Übersteigt das Ursprungsprogramm bei der Eingabe der Programmkarten die Kernspeicherkapazität, kann das Programmladen nicht fortgesetzt werden.

In diesem Fall ist zu prüfen, ob die Kernspeicherbelegung reduziert werden kann.

Dies kann zunächst dadurch erreicht werden, daß Konstante oder Arbeitsdaten an das Programmende verlegt werden und den Kernspeicher benutzen, der davor durch die Pseudo-Befehle belegt war. Zum Zeitpunkt der Konstanten-Eingabe in den Kernspeicher werden die Pseudo-Befehle nicht mehr benötigt. Sie werden daher vom Ladeprogramm nach der Bearbeitung der symbolischen Kennzeichen gelöscht. Der von ihnen belegte Kernspeicher wird dadurch frei.

Eine weitere Möglichkeit liegt darin, das Vorprogramm (sofern vorhanden) in eine Arbeitszone oder in eine Datenzone zu legen, die bei Arbeitsbeginn nicht benutzt wird.

Eine dritte Möglichkeit besteht in der Segmentierung der Programme, wenn das Programmiersystem für die Magnet-Trommel benutzt wird (vgl. Unterlagen für Programmiersystem TB).

Welche Lösung auch immer angewendet wird, es ist empfehlenswert, den Kernspeicher nie vollständig zu belegen, da dadurch spätere Modifikationen u.U. unmöglich gemacht werden. Es könnte in diesem Fall vorkommen, daß das Programm vollständig neu erstellt werden muß, wenn eine unumgängliche Modifikation vorgenommen werden soll.

Nach der Korrektur wird das Ursprungsprogramm einer erneuten Kontrolle durch das Kontrollprogramm unterworfen. Dabei werden die symbolischen OT's übersetzt und das übersetzte Programm auf Karten ausgegeben. Eine Karte enthält in diesem Fall nur einen Befehl. Dieses übersetzte Programm wird nun für Testläufe innerhalb des Operating-Systems für Karten benutzt. Das Ursprungsprogramm wird archviert. Durch diesen zweiten Kontrollauf wird überdies eine Kontrolliste des korrigierten Programms erstellt.

Vor Beginn der Testarbeiten wird das Programm innerhalb des Programmsystems TB durch den Assembler bearbeitet. Dabei erfolgt eine Assemblierung des Programms im Hinblick auf die Verarbeitung des Programms von der Trommel her. Das auf diese Art umgewandelte und auf der Trommel gespeicherte Programm kann direkt für Testläufe geladen und gestartet werden.

5.3.3.2. Testläufe des Programms

Nach der ersten Formalkorrektur und der Übersetzung und Assemblierung des Programms können die Testläufe begonnen werden. Durch die Testläufe werden die logischen und technischen Fehler des Programmierers festgestellt. Die Testläufe sollen daher dem tatsächlichen Arbeitsablauf des Programms soweit als möglich entsprechen. Es sind so viele Testläufe durchzuführen wie Programmverzweigungen vorhanden sind, wobei möglichst vollständige Testdaten als Grundlage dienen sollen.

Der Programmtest selbst besteht in einem Start des Programms, wobei das entsprechende Operating-System benutzt wird. Es ist dabei so vorzugehen, als ob das Programm für einen tatsächlichen Arbeitsablauf gestartet würde. Beim Operating-System Lochkarten wird das Programm mit Hilfe des Ladeprogramms, beim Operating-System für Magnet-Trommeln mit Hilfe des Monitors geladen.

Beim Test sind nicht nur die Bearbeitungsfunktionen des Programms zu prüfen, sondern auch die spezifischen Wiederanlaufroutinen im Programm zu testen. Darüber hinaus sind die Standardprogramme für die Programmunterbrechung (OS Karten) bzw. die Supervisor-Funktionen (OS Trommel) zu überprüfen. Dies wird durch Simulation der verschiedenen Fehlerfälle erreicht.

Die Korrektur der beim Test erkannten Fehler wird im Ursprungsprogramm vorgenommen, um es auf dem neuesten Stand zu halten. Dadurch wird auch vorausgesetzt, daß nach den Korrekturen sämtliche bis zu diesem Punkt beschriebenen Arbeiten wiederholt werden müssen. Um eine zu häufige Wiederholung dieser Arbeitsgänge zu vermeiden, empfielt es sich, mehrere Testläufe auf einmal durchzuführen und erst dann die erforderlichen Korrekturen im Ursprungsprogramm vorzunehmen.

5.3.3.3. Arbeitsaufnahme mit dem Programm

Nach der Durchführung aller Testläufe und Korrekturen erfolgt eine letztmalige Redaktion des Programms für die endgültige Arbeitsaufnahme. Zu diesem Zweck wird das Programm in den Kernspeicher geladen und mit Hilfe des Programms "auf Programmkarten verdichten" in verdichtete Programmkarten gestanzt. Eine verdichtete Programmkarte enthält in der Regel mehrere Programmbefehle.

5.4. Betrieb der Programme

In diesen Abschnitten wurden nur die Arbeitsgänge innerhalb der einzelnen Operating-Systeme kurz schematisiert. Eine genaue Beschreibung der Operating-Systeme befinden sich im Abschnitt 5.2.

6. Tabellen

6.1. Die Codes

Für jeden Code steht der gleiche Zeichenvorrat zur Verfügung. Das erleichert den Übergang von einem zum anderen Code. Die verwendeten Zeichen entsprechen den Zeichen des Druckers.

6.1.1. Die externen Codes

6.1.1.1. Die Lochcodes

Diese Codes werden nicht automatisch in den internen Code (oder umgekehrt) umgeschlüsselt. Für jeden gibt es einen spziellen Zwischencode.

Die Tafeln auf den folgenden zwei Seiten zeigen für jeden Code:

- das dargestellte Symbol
- den Zwischencode
- den Lochcode
- die hexadezimale Darstellung im ISØ-Code.

H 14.012

Zeichen	Zwischen code	Lochcode			ISO - code = HEX	Zeichen	Zwischen code	Lochcode			ISO - code = HEX
↑	20				20	P	57	11	7		50
0	30		0		30	Q	58	11	8		51
1	31		1		31	R	59	11	9		52
2	32		2		32	S	22	0	2		53
3	33		3		33	T	23	0	3		54
4	34		4		34	U	24	0	4		55
5	35		5		35	V	25	0	5		56
6	36		6		36	W	26	0	6		57
7	37		7		37	X	27	0	7		58
8	38		8		38	Y	28	0	8		59
9	39		9		39	Z	29	0	9		5A
.	4B	12	8	3	2E	$	5B	11	8	3	24
/	21	0	1		2F	] <	4C	12	8	4	5D
#	3B		8	3	23	@ @	3C		8	4	40
%	2C	0	8	4	25	& &	40	12			26
-	50	11			2D	,	2B	0	8	3	2C
*	5C	11	8	4	2A	' ^	5F	11	8	7	27
A	41	12	1		41	((	4D	12	8	5	28
B	42	12	2		42	< +	4E	12	8	6	3C
C	43	12	3		43	\ !	4F	12	8	7	5C
D	44	12	4		44	↑	70	11	0		5E
E	45	12	5		45	;	5E	11	8	6	3B
F	46	12	6		46	=	2D	0	8	5	3D
G	47	12	7		47	" >	2E	0	8	6	22
H	48	12	8		48	! ?	2F	0	8	7	21
I	49	12	9		49	> =	3E		8	6	3E
J	51	11	1		4A	? "	3F		8	7	3F
K	52	11	2		4B	← \	2A	0	8	2	5F
L	53	11	3		4C	: '	3D		8	5	3A
M	54	11	4		4D	+	60	12	0		2B
N	55	11	5		4E	[:	3A		8	2	5B
O	56	11	6		4F	))	5D	11	8	5	29

^ 5E
¬ 5F

Ref.-Nr.: 23.20.001 D
Juli 1968

T 121

Zeichen	Zwischen code	Lochcode			ISO - code	Zeichen	Zwischen code	Lochcode			ISO - code
↛	70				20	P	55	8		4	50
0	71			0	30	Q	56	8		5	51
1	72			1	31	R	57	8		6	52
2	73			2	32	S	E8		9	11	53
3	74			3	33	T	61		9	0	54
4	75			4	34	U	62		9	1	55
5	76			5	35	V	63		9	2	56
6	77			6	36	W	64		9	3	57
7	30			7	37	X	65		9	4	58
8	50			8	38	Y	66		9	5	59
9	60			9	39	Z	67		9	6	5A
.	F8			11	2E	$	A0	7	9	12	24
/	20	7	9		2F	]	45	8	9	4	5D
#	21	7	9	0	23	@	47	8	9	6	40
%	25	7	9	4	25	&	26	7	9	5	26
-	43	8	9	2	2D	,	C8	8	9	11	2C
*	F0			12	2A	'	E0		9	12	27
A	B8	7		11	41	(	41	8	9	0	28
B	31	7		0	42	<	F2		1	12	3C
C	32	7		1	43	\	A8	7	9	11	5C
D	33	7		2	44	↑	F9		0	11	5E
E	34	7		3	45	;	27	7	9	6	3B
F	35	7		4	46	=	40	8	9		3D
G	36	7		5	47	"	24	7	9	3	22
H	37	7		6	48	!	F1		0	12	21
I	D0	8		12	49	>	C0	8	9	12	3E
J	D8	8		11	4A	?	46	8	9	5	3F
K	51	8		0	4B	←	23	7	9	2	5F
L	52	8		1	4C	:	22	7	9	1	3A
M	53	8		2	4D	+	44	8	9	3	2B
N	54	8		3	4E	[	FA		1	11	5B
O	B0	7		12	4F	)	42	8	9	1	29

Ref.-Nr.: 23.20.001 D

Juli 1968

6.1.2. Der (interne) ISØ-Code

Die folgende Tafel gibt eine Übersicht über den ISØ-Code.

Die mit Spalte bezeichneten Ziffern von 0 - 7 geben die Bitbelegung des linken Halbbytes an und die mit Zeile bezeichneten Ziffern von 0 - F die Belegung des rechten Halbbytes

Beispiel: Buchstabe G = 47 im ISØ-Code. Spalte 4 und Zeile 7 ergeben im Schnittpunkt den Buchstaben G

SPALTE	0	1	2	3	4	5	6	7	ISO CODE
Binär darstell.	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	ZEILE
0000			ß	0	@	P			0
0001			!	1	A	Q			1
0010			"	2	B	R			2
0011			#	3	C	S			3
0100			$	4	D	T			4
0101			%	5	E	U			5
0110			&	6	F	V			6
0111			'	7	G	W			7
1000			(	8	H	X			8
1001			)	9	I	Y			9
1010	Z		*	:	J	Z			A
1011	WRZ		+	;	K	Ä			B
1100	S		,	<	L	\			C
1101	WR		-	=	M	Ü			D
1110	V		.	>	N	↑			E
1111	N		/	?	O	←			F

6.2. Befehlskatalog

6.2.1. Ablaufzeiten der Befehle (in Millisekunden)

SPRÜNGE (3.2.)		
	durchgeführt	nicht durchgeführt
JRT	0.8	-
JIERT	1.5	1.3
JIURT	1.5	1.3
NOP	-	0.35
REGISTEROPERATIONEN (3.3) Arithmetische Operationen (3.3.2.)		
ADD	2.0	
ADDD	2.7	
SBD	2.0	
SBDD	2.7	
MPD	14.42 + 6.28 Z + 2.42 wenn m > 0 und M < 0 + 1.60 " m < 0 " M > 0 Z = Zahl der Wertstellen des Multiplikators m = Multiplikator; M = Multiplikand	
DVD	133,6 + (12,4 X 9) 9 = Anzahl der Ziffern des Quotienten	
Vergleichsbefehle (3.3.2)		
	gleich	ungleich
CMD	1.7	2.0
CMDD	2.0	2.6
Versetzungen und Rundung (3.3.3)		
SLD	1.0 + 0.9 V	V = Zahl der
SRD	1.0 + 0.9 V	Versetzungen
RØUND	5.0	

	Registerüberträge (3.3.4.)
MRR MDRR MSRDR EXR EXDR LRN	1.5 1.7 1.7 1.6 2.0 1.4 + 0.08 N (= N im Befehl)
	OPERATION AUF ZEICHENBASIS (3.4.) Logische Operationen
CMC NI ØI	bei Gleichheit : 2.0 + 0.07 N bei Ungleichheit : 2.3 + 0.10 N 1.6 1.1
	Überträge im Normalspeicherbereich (3.4.3.)
MVC MVIC INC PKH UPH	1.9 + 0.05 N 2.2 + 0.05 N 2.0 + 0.20 N 2.0 + 0.20 N 2.0 + 0.17 N
	MEHRFACHÜBERTRÄGE (3.5.4 und 3.5.5)
MR^{8}_{9} MR8G MR89	4.7 + 0.19 (a) (b) 4.7 + 0.18 N + 0.05 N (a) (b) 5.6 + 0.23 N (a) (b)
$M^{6}_{7}R89$	6.6 + 0.28 N (a)
$M^{6}_{7}R^{8}_{9}$	5.4 + 0.23 N (a)
$M^{6}_{7}R$	4.4 + 0.19 N (a)
$M^{6}_{7}89$	3.8 + 0.09 N
$M^{6}_{7}8G$	3.3 + 0.05 N + 0.05 N
$M^{6}_{7}{}^{8}_{9}$	3.3 + 0.05 N (a) zuzüglich : + 1.0, wenn N > 9 (b) " : + 0.4, " negativ

EIN/AUSGABE (3.6)	
IØC IØIC	Übernahme des Befehls: 2.0 + 1. E Durchführung: variabel (E = Zahl der Einheiten)
UMSCHLÜSSELUNGEN (3.7)	
TRx	1.1 + 0.10 N
MULTIPROGRAMMING (5.3.3)	
STØP START PRØ FREE	1.0 + 0.1 n 0.7 bis 1.0 offen: 1.0 geschlossen: 1.0 + 0.1n 0.7 bis 1.0 n = Zahl der Bytes der zu übertragenden Registerzonen.

6.2.2. Befehlskatalog in alphabetischer Ordnung

OT	Funktion	Abschn.
ADD	Addition in einfacher Länge	3.3.2.1.
ADDD	Addition in doppelter Länge	3.3.2.1.
CMC	Alphanumerischer Vergleich	3.4.2.1.
CMD	Algebr. Dezimalvergleich (einfacher Länge)	3.3.2.4.
CMDD	Algebr. Dezimalvergleich (doppelter Länge)	3.3.2.4.
DVD	Algebr. dezimale Division	3.3.2.3.
EXDR	Austausch des Inhalts von 2 Doppelregistern	3.3.4.3.
EXR	Austausch des Inhalts von 2 Einfachregistern	3.3.4.3.
FREE	Freigabe eines durch einen geschlossenen Schutz angehaltenen Programms	4.3.3.4.
INC	Einsetzen eines Zeichens	3.4.3.2.
IØC	Ein/Ausgabebefehl	3.6.6.
IØIC	Indirekter Ein/Ausgabebefehl	3.6.6.
JIERT	Sprung bei Gleichheit	3.2.2.2.
JIURT	Sprung bei Ungleichheit	3.2.2.2.
JRT	Systematischer Sprung	3.2.2.1.
KHLT	Fakultativer Halt	3.8.4.
LEVEL	Sprungniveau	~~3.2.3.~~ 39.1.3
LØAD	Laden eines Programms	3.8.2.
LRN	Laden eines numerischen Registers	3.3.4.4.
MDRR	Übertrag eines Doppelregister	3.3.4.1.
MPD	Algebr. dezimale Multiplikation	3.3.2.2.
MRR	Übertrag eines Einfachregisters	3.3.4.1.
MSRDR	Übertrag eines Einfachregisters in ein doppeltes	3.3.4.2.
MVC	Übertragungsbefehl	3.4.3.1.
MVIC	Indirekter Übertragungsbefehl	3.4.3.1.
Mxy(y)(y)	Mehrfachübertragungsbefehle	3.5.4.1.
Mx8G	Druckaufbereitungsbefehl	3.5.5.
NI	Logisches UND	3.4.2.2.
NØP	Nullbefehle (keine Operation)	3.2.4.
ØI	Logisches ODER	3.4.2.3.
PKH	Packen	3.4.3.3.
PRØ	Schutz	4.3.3.3.
PRSTØ	Kernspeicherausdruck	3.8.1.
RLA	Adressierung (reell) der Sprünge	3.8.3.
RØUND	Rundungsbefehl	3.3.3.3.
SBD	Subtraktion in einfacher Länge	3.3.2.1.
SBDD	Subtraktion in doppelter Länge	3.3.2.1.
SLD	Verschiebung nach links	3.3.3.2.
SRD	Verschiebung nach rechts	3.3.3.2.
START	Start eines Programms (durch ein anderes)	4.3.3.2.

< 3C
= 3D
> 3E

6.2.2. Befehlskatalog in alphabetischer Ordnung

OT Seite	Funktion	Abschn.
ADD 85	Addition in einfacher Länge	3.3.2.1.
ADDD 86	Addition in doppelter Länge	3.3.2.1.
CMC 103	Alphanumerischer Vergleich	3.4.2.1.
CMD 90	Algebr. Dezimalvergleich (einfacher Länge)	3.3.2.4.
CMDD 90	Algebr. Dezimalvergleich (doppelter Länge)	3.3.2.4.
DVD 88	Algebr. dezimale Division	3.3.2.3.
EXDR 100	Austausch des Inhalts von 2 Doppelregistern	3.3.4.3.
EXR 100	Austausch des Inhalts von 2 Einfachregistern	3.3.4.3.
FREE 135	Freigabe eines durch einen geschlossenen Schutz angehaltenen Programms	4.3.3.4.
INC 108	Einsetzen eines Zeichens	3.4.3.2.
IØC 133	Ein/Ausgabebefehl	3.6.6.
IØIC 132	Indirekter Ein/Ausgabebefehl	3.6.6.
JIERT 81	Sprung bei Gleichheit	3.2.2.2.
JIURT 81	Sprung bei Ungleichheit	3.2.2.2.
JRT 80	Systematischer Sprung	3.2.2.1.
KHLT 142	Fakultativer Halt	3.8.4.
LEVEL 152	Sprungniveau	~~3.2.3.~~ 3.9.1.3
LØAD 146	Laden eines Programms	3.8.2.
LRN 101	Laden eines numerischen Registers	3.3.4.4.
MDRR 96	Übertrag eines Doppelregister	3.3.4.1.
MPD 87	Algebr. dezimale Multiplikation	3.3.2.2.
MRR 96	Übertrag eines Einfachregisters	3.3.4.1.
MSRDR 98	Übertrag eines Einfachregisters in ein doppeltes	3.3.4.2.
MVC 106	Übertragungsbefehl	3.4.3.1.
MVIC 106	Indirekter Übertragungsbefehl	3.4.3.1.
Mxy(y)(y) 113	Mehrfachübertragungsbefehle	3.5.4.1.
Mx8G 116	Druckaufbereitungsbefehl	3.5.5.
NI 104	Logisches UND	3.4.2.2.
NØP 133	Nullbefehle (keine Operation)	~~3.2.4.~~ 3.9.1.4
ØI 105	Logisches ODER	3.4.2.3.
PKH 109	Packen	3.4.3.3.
PRØ 174	Schutz	4.3.3.3.
PRSTØ 144	Kernspeicherausdruck	3.8.1.
RLA 142	Adressierung (reell) der Sprünge	3.8.3.
RØUND 95	Rundungsbefehl	3.3.3.3.
SBD 85	Subtraktion in einfacher Länge	3.3.2.1.
SBDD 85	Subtraktion in doppelter Länge	3.3.2.1.
SLD 91	Verschiebung nach links	3.3.3.2.
SRD 93	Verschiebung nach rechts	3.3.3.2.
START 132	Start eines Programms (durch ein anderes)	4.3.3.2.

OT	Funktion	Abschn.
STØP 173	Halt eines Programms	4.3.3.1.
TRx 143	Umschlüsselung des Inhalts einer Zone	3.7.2.
UPH 111	Entpacken	3.4.3.4.
BEGIN 150		
END 151		
HEXAD 154		
AN 155		
Kartenmarken	192	
Ueberloch	142	
Zeilenschaltung	61	
Acceptability	213	

OT	Funktion	Abschn.
STØP	Halt eines Programms	4.3.3.1.
TRx	Umschlüsselung des Inhalts einer Zone	3.7.2.
UPH	Entpacken	3.4.3.4.

6.2.3. Befehlskatalog in hexadezimaler Reihenfolge

OT hexad.	L	OT symb.	Abschnitt	OT hexad.	L	OT symb.	Abschnitt
0 0	1	NØP	3.2.4.	6 0	5	LRN	3.3.4.4.
0 2	3	JRT	3.2.2.1.	6 1	4	NI	3.4.2.2.
0 4	6	JIERT	3.2.2.2.	6 3	4	ØI	3.4.2.3.
0 5	6	JIURT	3.2.2.2.	6 A	4	RLA	3.8.3.
1 2	4/5	MR8	3.5.4.1.	7 0	3	SLD	3.3.3.1.
1 4	4/5	MR9	3.5.4.1.	7 1	3	RØUND	3.3.3.3.
1 6	5/7	MR89	3.5.4.1.	7 2	3	SRD	3.3.3.2.
1 E	5/6	MR8G	3.5.5.	7 8	1	PRSTØ	3.8.1.
				7 D	1	LØAD	3.8.2.
2 1	4/5	M6R	3.5.4.1.				
2 2	4/6	M68	3.5.4.1.				
2 3	5/7	M6R8	3.5.4.1.				
2 4	4/6	M69	3.5.4.1.	8 0	6	MVC	3.4.3.1.
2 5	5/7	M6R9	3.5.4.1.	8 2	6	INC	3.4.3.2.
2 6	5/8	M689	3.5.4.1.	8 4	6	PKH	3.4.3.3.
2 7	6/9	M6R89	3.5.4.1.	8 5	6	UPH	3.4.3.4.
2 E	5/7	M68G	3.5.5.	8 8	6	CMC	3.4.2.1.
				8 A	6	MVIC	3.4.3.1.
3 1	4/5	M7R	3.5.4.1.				
3 2	4/6	M78	3.5.4.1.				
3 3	5/7	M7R8	3.5.4.1.	9 1	3	CMD	3.3.2.4.
3 4	4/6	M79	3.5.4.1.	9 2	3	CMDD	3.3.2.4.
3 5	5/7	M7R9	3.5.4.1.				
3 6	5/8	M789	3.5.4.1.				
3 7	6/9	M7R89	3.5.4.1.	A 1	3	ADD	3.3.2.1.
3 E	5/7	M78G	3.5.5.	A 2	3	ADDD	3.3.2.1.
				B 1	3	SBD	3.3.2.1.
4 0	1	STØP	4.3.3.1.	B 2	3	SBDD	3.3.2.1.
4 1	2	START	4.3.3.2.				
4 2	2	PRØ	4.3.3.3.	C 1	3	MPD	3.3.2.2.
4 4	2	FREE	4.3.3.4.				
				D 1	3	DVD	3.3.2.3.
5 0	2	TRØ	3.7.2.				
5 1	2	TR1	3.7.2.	E 1	3	MRR	3.3.4.1.
5 3	4/14	IØC	3.6.6.	E 2	3	MDRR	3.3.4.1.
5 4	4/14	IØIC	3.6.6.	E 3	3	MSRDR	3.3.4.2.
5 7	1	KHLT	3.8.4.	E 5	3	EXR	3.3.4.3.
5 A	2	TRA	3.7.2.	E 6	3	EXDR	3.3.4.3.
5 B	2	TRB	3.7.2.				
				F 2	2	LEVEL	3.2.3.

7. Die Software des GE-55

Wie jeder Computer ist auch der GE-55 mit einer Software ausgestattet. Sie besteht aus einer Reihe von Standardprogrammen, um dem Benutzer der Anlage bei dem Anlauf seiner Arbeitsprogramme zu helfen. Es sind mehrere Gruppen der Software vorhanden, die den einzelnen Konfigurationen der Anlage entsprechend für die verschiedene Methoden angewendet werden müssen: so gibt es zum Beispiel verschiedene Software für den GE-55 mit oder ohne Trommel.

7.1. Die Basissoftware

Die Anwendung der Basissoftware setzt einen GE-55 mit folgender Mindestausstattung voraus:

- einer Zentraleinheit mit:
  Festspeicher MMS9
  Zentralspeicher 2500, 5000 oder 10 000 Bytes
  Pufferspeicher mit Sichtanzeige
- einem Kartenleser
- einer numerischen Tastatur
- einem Stanzer PS 40
- einem Drucker MB 50.

Die Arbeiten auf einer solchen Anlage werden aufgrund von Programmen durchgeführt, die in Karten gelocht und in den Zentralspeicher vor der Anwendung eingegeben wurden.

Die Basissoftware besteht im einzelnen aus folgenden Programmen:

a) zum Einstellen und Testen der Arbeitsprogramme:

- einem Kontrollprogramm
- einem Listprogramm
- einem Verdichtungsprogramm

b) zur Ausführung dieser Arbeitsprogramme:

- einem Ladeprogramm
- einem Löschprogramm
- einem Unterbrechungsprogramm.

7.1.1. Aufbau der Basis-Softwareprogramme

Jedes Programm ist in Karten gelocht.

Es gibt mehrere Karten für jedes Programm. Die ersten 68 Spalten jeder Karte sind für das eigentliche Programm reserviert und die 12 letzten sind den Angaben, die zum Erkennen des Programms dienen, reserviert. Das sind:

- in der Spalte 69 und 70 der Schlüssel 23, der besagt, daß es sich um den GE-55 handelt (s. Ref.-Nr. der Dokumentation),

- in den Spalten 71 bis 74 die Programmnummer,

- in Spalte 75 den Lochcode, und zwar:

  1. wenn es sich um den Code H14.012 handelt,
  2. wenn es sich um den Code T121 handelt,

- in Spalte 76 steht die letzte Stelle des Jahresdatums der Erstellung dieses Programms,

- in Spalten 77 bis 79 die Nummer des lfd. Tages der Erstellung des Programmes,

- in der Spalte 80 eine 1, wenn es die erste Karte, eine 9, wenn es die letzte Karte und nichts, wenn es andere Karten sind. Man muß unbedingt die Karten, die eine 1 in Spalte 80 hat, an den Anfang legen und die Karte, die eine 9 in Spalte 80 hat, an den Schluß. Die Reihenfolge der anderen Karten ist ohne Bedeutung.

Die Anzahl der Karten der Software-Programme wird später angegeben.

7.2. Die Basis-Softwareprogramme

7.2.1. Die Programmkartenkontrolle (PKK)

Mit diesem Programm werden:

- die Lochungen in den Programmkarten geprüft
- die Konstantenkarten in Format 1 und 2 kontrolliert
- die Maschinen-OT aufgrund des symbolischen Operationstyps gestanzt
- die Befehle aufgelistet.

Die Programmnummer lautet für:

2500 Bytes : 23.30.12 (91 Karten) und 23.30 14 (77 Karten)
5000 Bytes : 23.30 15 (168 Karten)
10000 Bytes : 23.30 18 (168 Karten)

Die Kontrolle der Programm- und Konstantenkarten erfolgt Karte für Karte. Eine Liste der angezeigten Fehler befindet sich in Abschnitt 7.2.1.3.

7.2.1.1. Handhabung

- Eingabe der Programm-Kartenkontrolle.
- Eingabe einer Parameterkarte, die die gewünschte Funktion anzeigt (siehe 7.2.1.2.).
- Eingabe des zu kontrollierenden Kartenpaketes: (unverdichtetes Programm).

Das Programm muß von normalem Aufbau sein, d.h. eine Programm-Leitkarte, die Karte F3 und die Startkarte enthalten.

Ergebnisse:

Jede gelesene Karte wird gelistet mit der reellen Adresse und evtl. aufgetretenen Fehlern.

Wenn die Startkarte gelesen ist, werden sämtliche fehlenden LEVEL gelistet. Sobald die Programmkartenkontrolle beendet ist, wird auf der Leuchtanzeige 6 x die 7 angezeigt. Zu diesem Zeitpunkt wird die numerische Tastatur freigegeben und man kann mit Druck auf die Taste CLB den Puffer löschen.

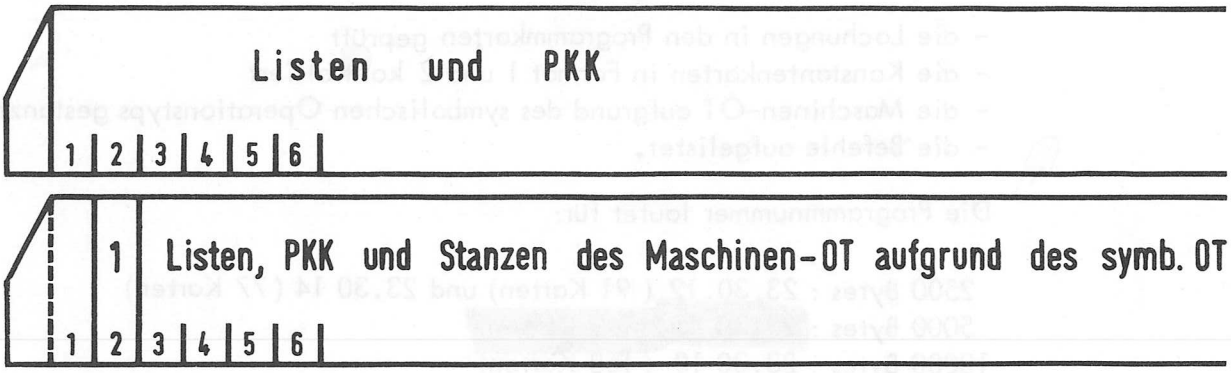
Wiederanlauf nach Fehler

Siehe Abschnitt 7.5.

Begrenzung

Für das zu kontrollierende Kartenpaket ist keinerlei Einschränkung vorgesehen.

7.2.1.2. Parameter



7.2.1.3. Fehlerarten

Fehler 1: Der Schlüssel in Spalte 2 + 3 stimmt nicht.

Fehler 2: Der symbolische Operationstyp existiert nicht.

Fehler 3: Der symbolische Operationstyp und der Maschinencode stimmen nicht überein (wenn der Maschinencode bereits gelocht ist).

Fehler 4: Übertragungsbefehle stimmen nicht überein (die Verschiebungen der Basen sind nicht numerisch oder unterschiedlich von Axxx).

Fehler 5: Anschlußcode des IØC oder des IØIC existieren nicht oder die angegebenen Register sind nicht numerisch.

Fehler: 6: Fehler in den Parametern (der Parametertyp entspricht nicht 0, 1, 2, 3, 4, 5, 6).
- Die Nummer der Karte ist entweder nicht numerisch oder größer 999.
- Die Feldnummer ist unterschiedlich von 1 - 9.

Fehler 7: Die Karte enthält Buchstaben oder Sonderzeichen, die verschieden sind von 1 - 9, A - F oder Leerzeichen (eine Ausnahme besteht für die Kartenart 31).

Fehler 8: Der Operationstyp existiert nicht.

Fehler 9: Die in den Spalten 4 + 5 angegebene Länge entspricht nicht der Länge, die dem Operationstyp entspricht.

Fehler 10: Die Anzahl der in der Karte gelochten Zeichen ist verschieden von der Zahl, die in den Spalten 4 - 5 gelocht ist.

Fehler 11: Die Länge ist nicht numerisch und entspricht auch nicht einer freien Stelle.

Fehler 12: Die Register sind nicht numerisch.

Fehler 13: Bxxx ist nicht numerisch.

Fehler 14: Ein Sprung ohne F1 mit Buchstaben in der Rücksprungadresse.

Fehler 15: Die Rücksprungadresse ist größer 2499 (4999, 9999).

Fehler 16: Sprung ohne F2.

Fehler 17: 2 Sprungstufen sind einander sehr ähnlich.

Fehler 18: Fehler in der Sortierfolge der Ordnungsnummer (leere Felder werden nicht berücksichtigt).

Fehler 19: Fehler in der Anzahl der Karten, d.h. die Anzahl der gelesenen Karten ist nicht identisch mit der gelochten Anzahl). Die Anzahl der gelesenen Karten wird rechts der Ziffer 19 angeschrieben.

Fehler 20: Die Startadresse ist nicht korrekt.

Fehler 21: In der Sprungangabe befindet sich eine Trennmarke.

Fehler # : Die Adresse in der Programm-Leitkarte ist nicht numerisch (In diesem Fall blockiert der GAMMA 55).

Fehler ! : Die Anzahl der gelochten Zeichen, die in Spalten 4 + 5 gelocht sind, ist nicht numerisch (In diesem Fall blockiert der GAMMA 55).

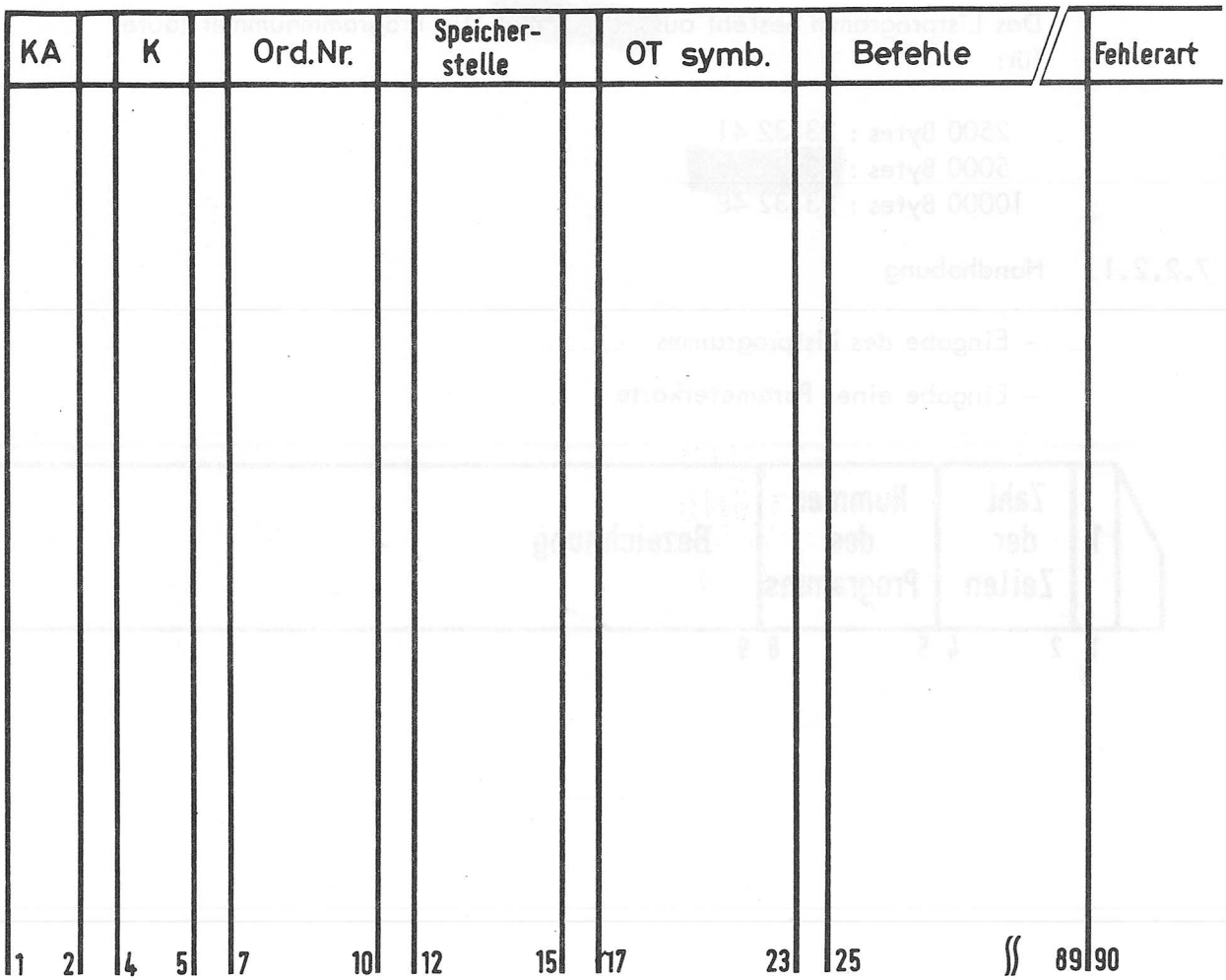
Fehler * : Eine LEVEL-Karte hat in den Spalten 4 + 5 eine Zahl, die unterschiedlich von 02 ist.

Fehler ? : Die Konstantenkarte befindet sich vor der Programmendkarte F3. Anschließend liegen noch Befehlskarten.

Fehler < : Die Verschiebung 1 ist kleiner als Verschiebung 2 in einem MVC-Befehl, falls die gleiche Basis benutzt wurde.

Außerdem werden bei der PKK, sobald die Startkarte gelesen ist, die Sprungbefehle angeschrieben, zu denen kein entsprechender LEVEL gefunden wurde.

Die Programmkarten-Kontrolle prüft die ersten 206 LEVEL-Nummern. Werden in einem Programm mehr als 206 Nummern belegt, oder ein LEVEL mit einer größeren Nummer wie 206 belegt, und die Sprungbefehle stimmen nicht mit den dazu gehörigen LEVEL überein, so schreibt die PKK statt der Fehlerliste: Fehler VV.


KA
K
Ord.Nr.
Speicher-stelle
OT symb.
Befehle
Fehlerart
1 2 4 5 7 10 12 15 17 23 25 89 90

7.2.2. Das Listprogramm (PK-Listen)

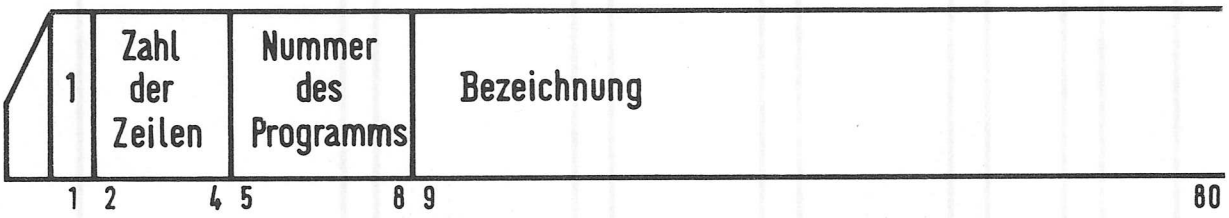
Mit diesem Programm werden die Programmkarten Karte für Karte aufgelistet. Gleichzeitig werden die Parameter für das Verdichtungsprogramm gestanzt.

Das Listprogramm besteht aus 45 Karten. Die Programmnummer lautet für:

2500 Bytes : 23.32 41
5000 Bytes : 23.32 45
10000 Bytes : 23.32 48

7.2.2.1. Handhabung

- Eingabe des Listprogramms
- Eingabe einer Parameterkarte



In den Spalten 2 - 4 der Parameterkarte werden die gewünschten Zeilen je Blatt gelocht. Ist diese Lochung vorhanden, erfolgt ein entsprechender Papiersprung bei der gewünschten Zeile.

Die angegebene Anzahl der Zeilen umfaßt die Kopfzeile, die Leerzeile zwischen der Kopfzeile und der 1. zu druckenden Zeile und den zu druckenden Zeilen. Die Programmbezeichnung wird auf der Höhe der Kopfzeile angeschrieben.

Das Vorhandensein einer Lochung in den Spalten 5 - 8 der Parameterkarte (Programmnummer) ermöglicht das Stanzen der für das Verdichtungsprogramm erforderlichen Parameterkarten (siehe Abschn. 7.2.3.). Diese Karten enthalten eine oder mehrere Parameter, die Anfangsadresse und die Endadresse der oder des zu verdichteten Programmes und eine Startkarte. Diese Startkarte dient nur dazu, um im Verdichtungsprogramm dupliziert und ergänzt zu werden mit der Anzahl der Karten des verdichteten Programms, die in die Spalten 17 - 20 gelocht wird.

Die Parameterkarte ist obligatorisch. Fehlt diese Parameterkarte, so wird die erste zu listende Karte wie eine Parameterkarte behandelt. Da im Prinzip die erste zu listende Karte die Programm-Leitkarte ist, die die Anfangsadresse enthält, wird diese nicht beachtet. Wird keine der gebotenen Möglichkeiten des Listprogramms gewünscht, so ist es möglich, eine Leerkarte als Parameterkarte zu nehmen.

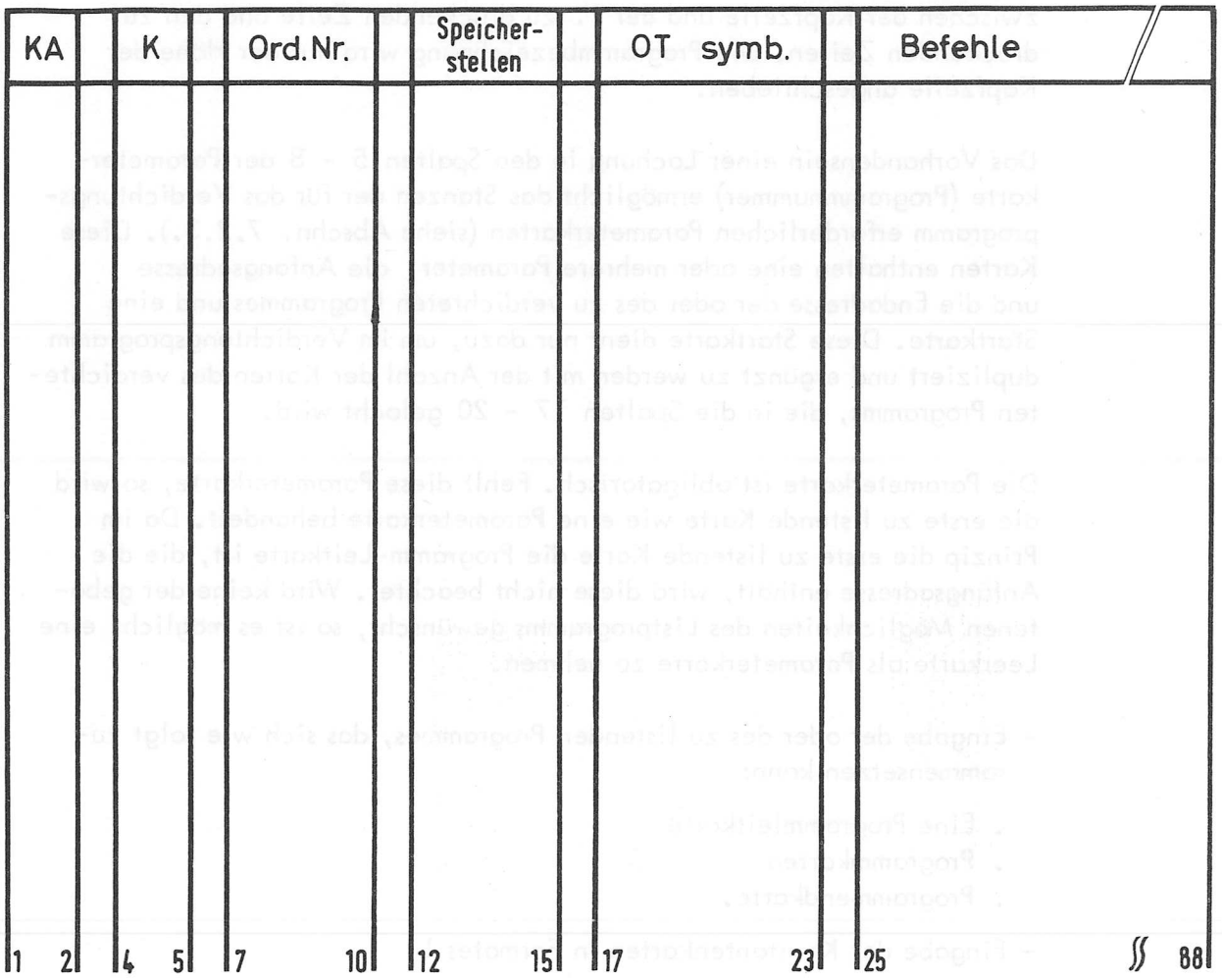
- Eingabe der oder des zu listenden Programmes, das sich wie folgt zusammensetzen kann:
  - . Eine Programmleitkarte
  - . Programmkarten
  - . Programmendkarte.
- Eingabe der Konstantenkarten in Formates 1
- Eingabe der Konstantenkarten in Formates 2
- Eingabe der Programmstartkarte.

Wiederanlauf nach Fehlerhalt

Wiederanlauf nach Halt des Druckers (Papierende).

Leuchtet die Lampe PR und bleibt die Maschine stehen, muß das Papier gewechselt werden, wobei darauf zu achten ist, daß das erste Blatt auf der ersten zu druckenden Zeile steht. Anschließend ist der Sprungstreifen einzustellen und das letzte Blatt des Papierstapels in den Kontakt für Papierende einzuschieben. Ein Druck auf die Taste RES des Druckers läßt die Arbeit normal weiterlaufen.

Wiederanlauf nach Fehlerhalt des Stanzers (siehe Abschn.7.5)


KA
K
Ord.Nr.
Speicher-
stellen
OT symb.
Befehle
1
2
4
5
7
10
12
15
17
23
25
88

7.2.3. Das Verdichtungsprogramm (MEFE)

Mit diesem Programm werden getestete Arbeitsprogramme verdichtet. In einem nicht fertig getesteten Programm enthält jede Programmkarte nur einen Befehl. In dem verdichteten Programm sind mehrere Befehle je Karte gelocht.

Das Verdichtungsprogramm besteht aus 19 Karten. Die Programmnummer lautet für:

2500 Bytes	:	23.32 43
5000 Bytes	:	23.32 46
10000 Bytes	:	23.32 49

Ein verdichtetes Programm bietet zwei wesentliche Vorteile:

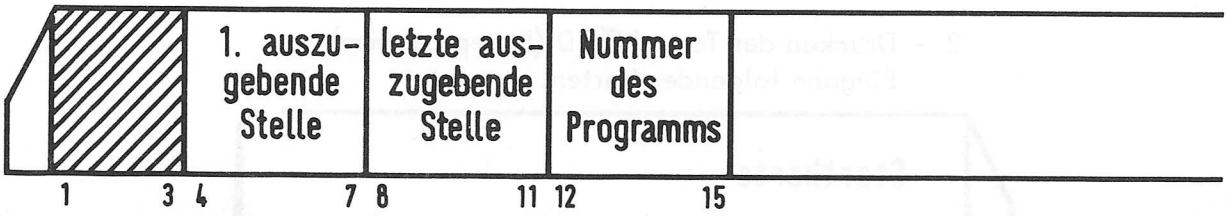
- Es gibt eine wesentliche Verringerung der Kartenanzahl.
- Die gelochte Adresse in den Spalten 6 - 9 gestattet es, die Karten in beliebiger Reihenfolge einzugeben.

7.2.3.1. Handhabung

Das Programm "Listen der Programmkarten" liefert die für das Verdichtungsprogramm erforderlichen Parameter, die folgendes enthalten:

- Eine oder mehrere Parameterkarten des Types, der unten beschrieben ist und eine Startkarte.

Es ist ratsam, sich von jedem Programm eine Liste anzufertigen, bevor es verdichtet wird, da man dann automatisch die gewünschten Parameterkarten für das Verdichtungsprogramm erhält.



- Eventuell ist das Löschprogramm einzugeben.
- Anschließend wird das getestete Programm mit dem Ladeprogramm (ICARE) eingegeben. Es ist unbedingt erforderlich, daß die Startkarte vor dem Eingeben herausgenommen wird.
- Danach wird das Verdichtungsprogramm (MEFE) eingegeben.

- Eingeben der Parameterkarten und der Startkarte, die beim Listen des Programmes gestanzt wurden. Vorher ist das Zufuhrmagazin des Stanzers mit Leerkarten zu füllen.

Anlauf nach Fehlerhalt

Der einzige, vorgesehene Fehlerhalt liegt bei Stanzfehler vor.

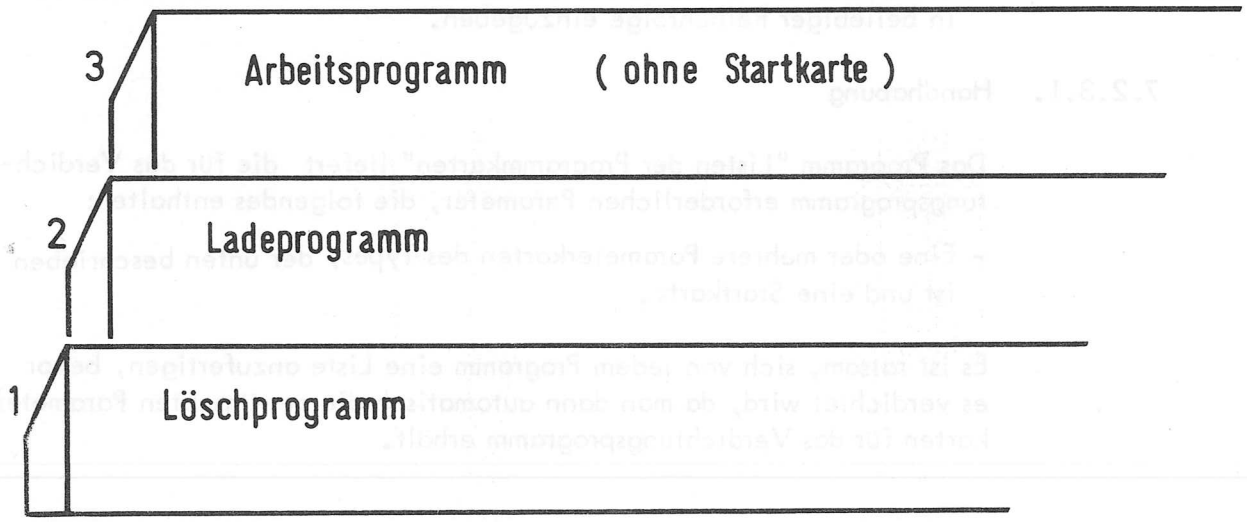
Begrenzungen

Es ist nur möglich, Programme von Stelle 0161 bis zur Stelle
1732 bei 2.500 Bytes
4040 bei 5.000 Bytes
9040 bei 10.000 Bytes

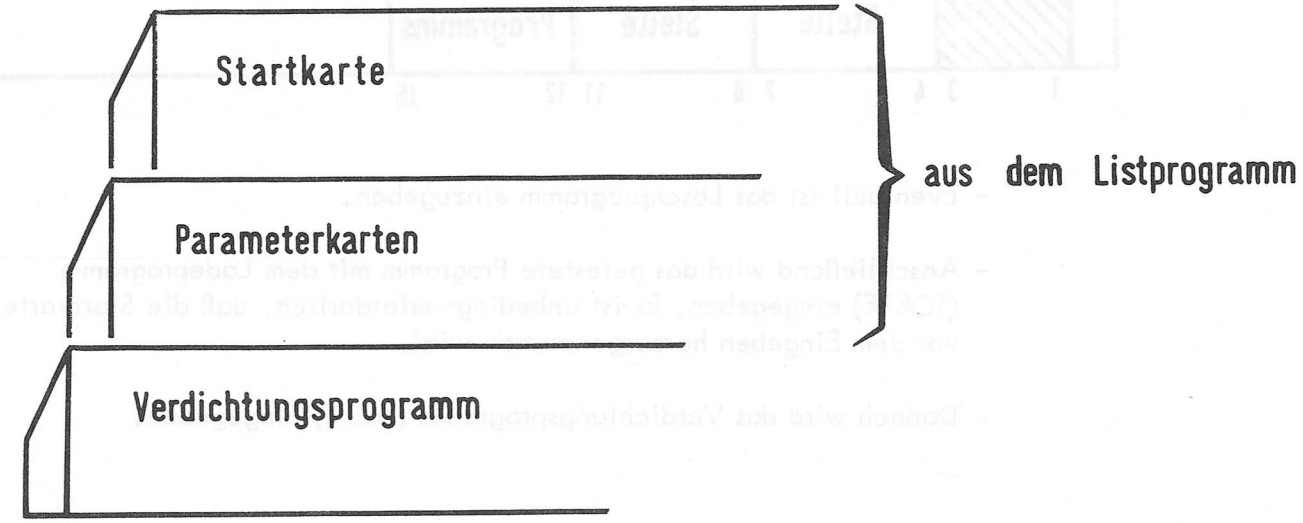
einschließlich einzugeben.

7.2.3.2. Arbeitsablauf

1 - Drücken der Taste LØAD
Eingabe folgender Karten:



2 - Drücken der Taste LØAD (Laaeprogramm)
Eingabe folgender Karten:



7.2.4. Das Ladeprogramm (ICARE)

Das Ladeprogramm gibt ein Arbeitsprogramm in den Kernspeicher ein, ohne es zu starten. Zusätzlich wird das Unterbrechungsprogramm eingespeichert (s. 7.2.6.).

Die Programmnummer lautet für:

2500 Bytes	:	23.33 32	(32 Karten)
5000 Bytes	:	23.33 33	(37 Karten)
10000 Bytes	:	23.33 34	(37 Karten)

Durch das Ladeprogramm erfolgt außerdem:

- das Einsetzen der reellen Adresse anstelle der Symbole in den Sprungbefehlen.
- eine Kontrolle der Zahl von Programm- und Konstantenkarten.
- ein Freimachen der von den LEVEL belegten Stellen (Zahl der LEVEL x 2) und ein Zusammenschieben des Programms danach.
- ein Belegen des Kernspeichers mit bestimmten Werten (s. Formular "Kernspeicherbelegung").
- der Start des eingegebenen Programms.

Das gesamte System besteht aus:

- dem eigentlichen Ladeprogramm für unverdichtete oder verdichtete Programme.
- dem Unterbrechungsprogramm (s. 7.2.6.).

7.2.4.1. Die Handhabung

Laden eines unverdichteten Programms

- Laden des Löschprogramms.
- Laden des Ladeprogramms und des Unterbrechungsprogramms.
- Eingabe des zu ladenden Programms, das sich wie folgt zusammensetzen muß:
  - Eine Programmbeginnkarte
  - Die Programmkarten
  - Eine Endkarte
- Eingabe der Konstantenkarten, Format 1,
- Eingabe der Konstantenkarten, Format 2,
- Eingabe der Startkarte.

Laden eines verdichteten Programms:

- Laden des Löschprogramms
- Laden des Ladeprogramms und des Fehlerprogramms
- Eingabe des verdichteten Programms, das sich wie folgt zusammensetzt:
  - . Konstantenkarten oder Programmkarten, Format 2,
  - . eine Programm-Startkarte.

Wiederanlauf nach Unterbrechung

Es gibt eine Unterbrechung, wenn die Anzahl der eingegebenen Karten nicht mit der in der Programmstartkarte angegebenen Anzahl der Karten übereinstimmt. In diesem Fall wird an der Leuchtanzeige dreimal die 1 sichtbar. Man kann diese Unterbrechung übergehen, indem man den Wiederanlauf mit einer fixen Adresse, die im Fehlerprogramm vorgesehen ist, bewerkstelligt:

- Löschen des Pufferspeichers (Taste CLB)
- Eintasten (auf der numerischen Tastatur): 88
- Übertrag Puffer an Kernspeicher (Taste MVB).

Die Begrenzungen

Es ist möglich, ein Programm einzugeben, das von Stelle 0161 bis zur Stelle 1732 bei 2.500 Bytes
4040 bei 5.000 Bytes
9040 bei 10.000 Bytes
einschließlich eingespeichert wird.

Die maximale Anzahl der benutzbaren LEVEL ist 256.

Ref.-Nr.: 23.20.001 D

Juli 1968

7.2.5. Das Löschprogramm

Das Löschprogramm löscht den Kernspeicher von Stelle 161 an bis zum Ende. Das ist je nach Kernspeicherkapazität die Stelle 2499, 4999 oder 9999.

Es besteht aus 3 Karten. Die Programmnummer (Sp. 71 - 74) lautet für:

2500 Bytes	:	23.32 42
5000 Bytes	:	23.32 44
10000 Bytes	:	23.32 47

7.2.6. Das Unterbrechungsprogramm (PGI)

Ziel des Unterbrechungsprogramms ist es, die Arbeit des Programmierers zu erleichtern, indem sie ihm eine Standardlösung für die Wiederanläufe nach Unterbrechungen gibt, die er in seinem Programm vorsehen muß. Außerdem soll es in gleichem Maße die Arbeit des Bedieners erleichtern durch die Normalisierung der zu bewerkstelligenden Handgriffe im Falle einer Unterbrechung.

Es besteht darin:

- die Art der Unterbrechung sichtbar zu machen
- einen automatischen Wiederanlauf für die nichtprogrammierten Unterbrechungen zu gewährleisten
- ein spezielles Wiederanlaufprogramm für die Unterbrechungen zu starten.

7.2.6.1. Beschreibung des Programms

Anruf des Programms

Der Anruf des Unterbrechungsprogramms kann auf zwei Arten erfolgen:

- durch Druck auf die Taste PI (Programmunterbrechung). Der Druck auf die Taste PI bewirkt:
  - daß die Programmadresse von PAR nach ANU übertragen wird,
  - daß die Adresse von AFP nach PAR übertragen wird.
- durch Belegung der Pufferzone mit den Fehlercodes und durch einen Sprung auf die Adresse 4676 (bei 5000 Bytes). Der Programmierer hat die Möglichkeit, die drei linken Stellen der Pufferzone mit Signalcodes für Unterbrechungen zu belegen. Durch das PGI werden an der Leuchtanzeige die drei rechten Stellen mit 111 belegt.

Ablauf des Programms

Nacheinander erfolgt:

- die Speicherung des Inhalts der Pufferzone
- die Speicherung der Rücksprungadresse
- die Speicherung der drei ersten Bytes der PRC-Zone
- Setzen eines Testes für Monoprogramming
- Sichtbarmachung von 3 x einer '1' auf der Leuchtanzeige und evtl. auch der spezielle Code für Unterbrechungen.

Jetzt kann durch Druck auf die Taste CLB der Inhalt des Pufferspeichers gelöscht und über die numerische Tastatur die Art des Wiederanlaufs angegeben werden:

- Lesen einer Karte
- Stanzen einer Karte von der Stanzzone her
- Setzen eines Tests im Zentralspeicher auf der Adresse 4661 (bei 5000 Bytes)

Anschließend erfolgt:

Beim Wiederanlauf nach einer Unterbrechung des Lesers, Stanzers oder nach dem Test wird der ursprüngliche Zustand der logischen Zone wieder hergestellt. Dieses Herstellen des ursprünglichen Zustandes umfaßt:

- den Start des Programmes, das gestartet war im Augenblick der Unterbrechung,
- die Annullierung des Tests für Monoprogramming, falls ein Test gesetzt wurde,
- die Wiederherstellung des ursprünglichen Zustandes in der Pufferzone,
- den Rücksprung ins Hauptprogramm,
- den Start eines Programmes mit einem spezifischen Wiederanlauf, dessen Adresse sich an den Stellen 4659 und 4660 befinden muß (bzw. 2355 und 2356)
- den Start eines spezifischen Programms, dessen Anfangsadresse über die Tastatur eingegeben wird.

Möglichkeiten des Wiederanlaufs

Es ist möglich, alle diese Starts durch das Unterbrechungsprogramms durchzuführen, wenn das Programm durch die Taste PI oder durch Programm angerufen wird.

Es ist aber zu bemerken, daß der Anruf des Unterbrechungsprogramms (PGI) durch Druck der Taste PI automatisch die Rücksprungadresse nach Unterbrechung in die Programmadresse setzt.

Dadurch wird dem PGI ermöglicht, das unterbrochene Programm auf der Adresse neu zu starten, die der unterbrochenen Adresse folgt.

Daraus ergibt sich, daß, wenn das PGI durch ein Programm angerufen wurde, um ein Lesen oder Stanzen oder das Setzen eines Testes zu bewirken, ein Einsetzen der Adresse nach Unterbrechung nicht erfolgt. Daher muß der Programmierer wie folgt verfahren:

Anstatt sofort in das Fehlerprogramm zu springen, erfolgt ein Sprung in ein Unterprogramm:

```
LEVEL   F 2  ..
MVC     80   02   0088   0093
MVC     80   02   0095   0090
```

Vorsichtsmaßnahmen

Monoprogramming_

Der Wiederanlauf nach einer Unterbrechung, die das PGI anruft, schließt den Gebrauch der numerischen Tastatur und des Pufferspeichers ein.

Der Programmierer muß daher beachten, daß er die numerische Tastatur nicht benutzt, wenn das PGI angesprochen werden kann.

Beispiel:

Dieses Programmierbeispiel ist zu vermeiden:

```
IØC   Freigabe numerischer Tastatur
IØC   Stanzen
IØC   Übertrag Puffer an Zentralspeicher.
```

Wird ein Fehler im Stanzer festgestellt, so wird die Freigabe für die numerische Tastatur durch den Sprung ins Fehlerprogramm (PGI) annulliert.

Multiprogramming_

Es existieren zwei Arten des Multiprogramming:

Die eine Methode besteht darin, ein Hauptprogramm und mehrere Unterprogramme (1 - 4)dazu zu schreiben. In diesem Falle genügt es,als Wiederanlauf die Arbeit von Beginn einer Gruppe an neu aufzunehmen. Dafür kann ein spezifisches Programm angewendet werden, das unter 7.2.6.2. erläutert ist. Mit ihm können 2 - 5 Unterprogramme bearbeitet werden.

Ein Wiederanlauf nach einem Fehler an den Randeinheiten kann durch das PGI nach der alten klassischen Methode erfolgen, ohne daß man mit Beginn einer neuen Gruppe neu anfangen muß.

Die zweite Art von Multiprogramming besteht darin, daß man mehrere Programme hat, die in keiner Verbindung zueinander stehen und die verschiedene Probleme behandeln. In diesem Falle wird jedes Programm als ein lineares Programm, wie bei Monoprogramming, betrachtet.

Jedenfalls muß der Programmierer beachten, daß der gleiche Kanal nicht im selben Moment von mehreren Programmen beansprucht wird. Vor jedem IØC muß ein Schutz (PRØ) vorgesehen werden. Es wird empfohlen, für einen Schutz die Nummer anzugeben, die mit der Kanalnummer übereinstimmt, z.B. für einen IØC, der den Kanal 1 benutzt, wird PRØ 01 gesetzt.

Beispiel:

```
PRØ     02
IØC     Stanzen
FREE    02      Programm 1
PRØ     02
IØC     Stanzen
FREE    02      Programm 2
```

Der Wiederanlauf nach einer Unterbrechung durch die Taste PI läuft normal weiter. Der Wiederanlauf nach einer programmierten Unterbrechung wird mit einem Sprung auf die Adresse 4726 (2422) gestartet, so daß die anderen Programme nicht blockiert werden. Der Programmierer kann daher in dem Pufferspeicher einen Code eingeben, der die Art der Unterbrechung kennzeichnet.

7.2.6.2. Anwendung

Anruf eines Lesebefehls

- Drücken der Taste P.I.
- 111 wird an der Leuchtanzeige sichtbar.
- Löschen des Pufferspeichers (Druck auf die Taste CLB).
- Eintasten der Zahl 55 (51).
- Übertrag Puffer an Zentralspeicher (Druck auf die Taste MVB).

Wiederanlauf nach einem Halt des Stanzers

- Drücken der Taste P.I.
- Drücken der Taste Auswurf (EJ), um die letzte richtig gestanzte Karte auszusteuern.
- Drücken der Taste RES. Zu diesem Zeitpunkt wird das Ende der fehlerhaften Karte in eine Leerkarte gestanzt und dreimal die '1' auf der Leuchtanzeige sichtbar.

- Drücken zweimal auf die Taste EJ.
- Herausnehmen der fehlerhaften Karten, die sich im Ablagefach befinden.
- Löschen des Pufferspeichers mit Taste CLB.
- Eintasten auf der numerischen Tastatur den Code 48 (44).
- Drücken der Taste MVB (Übertrag Pufferspeicher an Zentralspeicher).

Die Arbeit läuft normal weiter. In jedem Falle liegt eine Leerkarte im Ablagefach.

Setzen eines Tests

- Drücken der Taste P1.
- Es muß dreimal die '1' auf der Leuchtanzeige sichtbar gemacht werden, sofern deren Ablauf nicht gestört und kein Fehlerfall angezeigt wurde.
- Löschen des Pufferspeichers durch Drücken der Taste CLB.
- Eintasten auf der numerischen Tastatur des einzugebenden Testwert und anschließend den Code 62 (58)
- Übertrag Puffer an Zentralspeicher durch Drücken der Taste MVB.

Dadurch wird erreicht, daß der Test im Zentralspeicher die Stelle 4661 (2357) besetzt und daß man auf das Hauptprogramm zurückkommen kann.

- Starten eines Programms an einer fixen Adresse.

Bei Multiprogramming muß der Programmierer das spezifische Wiederanlaufprogramm vorsehen.

Um dies zu starten, muß man nach Auftreten des Fehlercodes und der angezeigten '111' wie folgt verfahren:

- Löschen des Pufferspeichers.
- Eintasten der Zahl 88
- Übertragen dieses Wertes in den Kernspeicher mit der Taste MVB.

Dadurch wird das Programm mit der Adresse gestartet, die im Kernspeicher an Stelle 4659 und 4660 (2355/56) steht. Es ist zu beachten, daß das Ladeprogramm in der Programmadresse die Startadresse, die in der Startkarte gelocht ist, speichert. Der Programmierer hat die Möglichkeit, an einer anderen Adresse als der Startadresse zu starten. Die Stellen 4659 und 4660 (2355/56) sind dann fixe Adressen.

- Starten eines Programms an eine Adresse, die mit der Tastatur eingegeben wird.

Bei Multiprogramming muß der Programmierer das spezifische Wiederanlaufprogramm vorsehen.

Um das Programm starten, wird nach Erscheinen des Fehlercodes und der '111' auf der Leuchtanzeige wie folgt verfahren:

- Löschen des Pufferspeichers (Taste CLB).
- Eingeben der reellen Adresse des Programms, auf welches man springen will, anschließend Eingabe des Codes 98 (94).
- Übertrag des Pufferspeichers an den Zentralspeicher (Taste MVB).

Dadurch wird das Programm an der angezeigten Adresse gestartet.

Bemerkungen:

Der Programmierer muß sich immer die Tatsache vor Augen halten, daß das Unterbrechungsprogramm sich der numerischen Tastatur bedient und daß jede Freigabe der numerischen Tastatur nicht in dem Moment erfolgen darf, wo ein Anruf des PGI erfolgt.

- Platzbelegung

Das Unterbrechungsprogramm (PGI) belegt 142 (145) Bytes, wovon 128 für das Programm und 14 (17) zum Speichern benutzt werden.

Einschränkungen

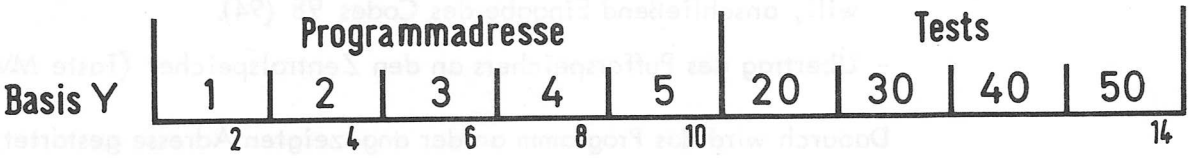
Die Programme, die sich des Unterbrechungsprogrammes bedienen, müssen zwangsläufig in jeder ihrer Registerzonen, die dem Programmwechsel unterliegen, folgendes berücksichtigen:

- Das Register 00 muß immer leer bleiben.
- Die Register 6, 8 und 9 müssen immer die Standardbasen für das Lesen, den Druck und für das Stanzen beinhalten.
- Das Register 10 muß immer die Standardbasis für den Pufferspeicher enthalten.
- Das Register 11 muß immer den Code für die Leuchtanzeige, die Trennmarke F4 und die Adresse 0150 enthalten.
- Das Register 12 muß immer den Code für den Übertrag des Puffers an den Zentralspeicher, die Trennmarke F4 und die Adresse 0155 enthalten.

7.2.6.2. Das spezifische Wiederanlaufprogramm

Dieses Programm benötigt im Kernspeicher eine Arbeitszone von 14 Bytes, die folgendes enthält:

Die Adresse des Unterprogramms des spezifischen Wiederanlaufs, die Anfangsadressen der Untergruppenprogramme und die Tests "Halt-Programm" der Programme 2 - 5. Beispiel:



- es benutzt einen LEVEL (AA)
- es belegt ein Basisregister, das durch den Programmierer zu bestimmen ist, der dann die Programm-Linien, die dieses Register benutzen, ergänzen muß.

In der Basis Y finden wir folgendes:

1 : Adresse des Unterprogramms des spezifischen Wiederanlaufs

2 - 5 : Anfangsadressen der Untergruppen-Programme

20 - 50: Die wirklichen Werte, die in der PRC-Zone zu untersuchen sind.

Plazierung

Das Programm belegt 94 Bytes, wenn 4 Unterprogramme benutzt sind.

Es belegt 2 Basisregister X + Y und eine Arbeitszone von 14 Bytes.

Einschränkungen

Dieses Programm wird den normalen Bedingungen des Anwendungsbereichs der Maschine angepaßt, d.h. es benutzt die normale Basis-Software.
Das Register 0 muß immer vor einem Sprung ins PGI auf 0 stehen.

Beispiel:

siehe Seite 233

BULL GENERAL ELECTRIC

Name des Programms :

Nr.:

PROGRAMMFORMULAR GE 55

Erläuterung	Sprünge F1	Sprünge F2	KA 2 3	Zahl der Bytes 4 5	Ord.-Nr. 6 9	Symbolischer Operationstyp 10 16	Befehl 17 20	21 24	25 28	29 32
			1206		0000	JIURT	05110070	F1AA		
Übertrag für P2			1206		0100	MVC	80020067	Adresse eines Basisregisters (x)		
			1206		0200	MVC	8002Y004	X005		
			120		0300					
Übertrag für P3			1206		0400	MVC	80020065	Adresse eines Basisregisters (x)		
			1206		0500	MVC	8002Y006	X005		
			120		0600					
Übertrag für P4			1206		0700	MVC	80020063	Adresse eines Basisregisters (x)		
			1206		0800	MVC	8002Y008	X005		
			120		0900					
Übertrag für P5			1206		1000	MVC	80020061	Adresse eines Basisregisters (x)		
			1206		1100	MVC	8002Y010	X005		
			120		1200					
Löschen PRC			1206		1300	INC	82000084	0070		
Übersetzen "HALT" (PRC)			1206		1400	MVC	8004Y014	0084		
			120		1500					
			1206		1600	JRT	02 Anfangadresse des Programms			1
			120		1700					
			1402		1800	LEVEL	F2AA			
			120		1900					
Übertrag für P1			1206		2000	MVC	80020069	Adresse eines Basisregisters (x)		
			1206		2100	MVC	8002Y002	X005		
			120		2200					
			1204		2300	NI	610F0070			
PRC 11			1206		2400	INC	82110071	0070		
			120		2500					
Programmwechsel			1201		2600	STOP	40			
			120		2700					
			120		2800					
			120		2900					
			120		3000					
			120		3100					

12.66 / 1009

Buchstabe O = Ø, Ziffer Null = 0

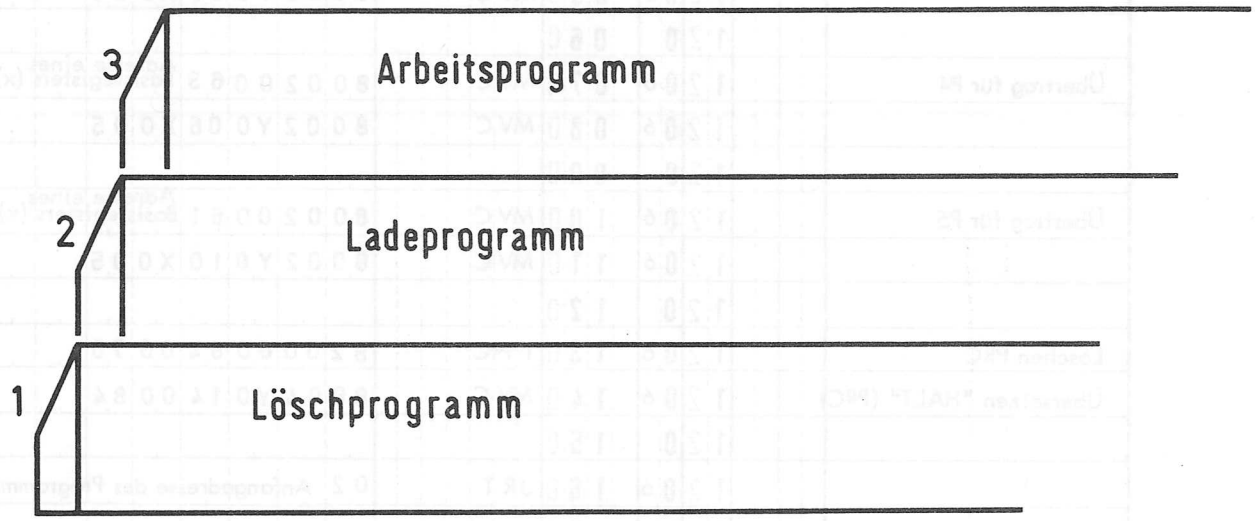
7.3. Anwendung der Basis-Software-Programme

7.3.1. Das Laden eines Arbeitsprogrammes

Um ein Arbeitsprogramm zu laden, muß man folgende Software-Programme benutzen:

- das Löschprogramm. In gewissen Fällen wird dieses Programm nicht benutzt. Die genaue Handhabung muß auf der Bedienungsanweisung eingetragen sein.
- das Ladeprogramm.

Die verschiedenen Programme müssen in folgender Reihenfolge eingegeben werden:



7.3.1.1. Handhabung

- Taste LØAD drücken.
- Die Karten in den Kartenleser in oben angezeigter Reihenfolge einlegen.
- Taste RES drücken.

Jetzt werden die Karten gelesen. Nach dem Lesen der letzten Karte, die immer eine Startkarte mit einem Schlüssel 16 oder 36 in Spalte 2 und 3 sein muß, können sich zwei Möglichkeiten ergeben.

a) Auf der Sichtanzeige wird 3 x eine 1 (111) angezeigt. In diesem Fall stimmt die Anzahl der Programmkarten nicht mit der abgelochten Zahl überein. Es ist folglich notwendig, das Programm in die richtige Ordnung zu bringen und von vorne anzufangen.

b) Es erscheint auf der Leuchtanzeige keine 1.

- Die Datenkarten in das Zufuhrmagazin legen.
- Auf RES drücken.

7.3.1.2. Wiederaufnahme bei Fall a)

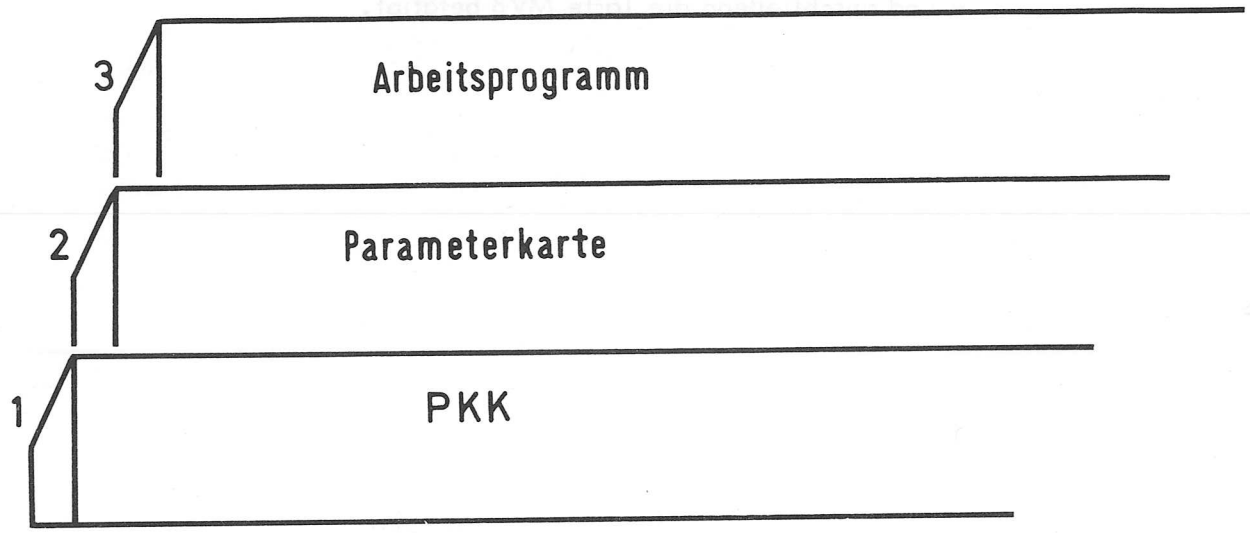
Vorher wurde gesagt, daß ein Fehler vorliegt, wenn die Anzahl der eingelesenen Karten unterschiedlich ist von der Zahl, die auf der Startkarte gestanzt ist. Man kann diesen Fehler übergehen, wenn man:

- Den Puffer löscht, indem man auf die Taste CLB drückt,
- auf der numerischen Tastatur 88 eingibt,
- und anschließend die Taste MVB betätigt.

7.3.2. Die Programmkartenkontrolle

Um ein Arbeitsprogramm zu testen, ist es notwendig, das Programm Programmkartenkontrolle anzuwenden (s. 7.2.1.).

Am Anfang liegt die PKK, gefolgt von einer Parameterkarte und dem zu testenden Programm.



7.3.2.1. Handhabung

- Taste LØAD drücken.
- Die Karten, wie oben, in den Kartenleser einlegen.
- Taste RES drücken.

Jetzt werden die Karten gelesen.

7.3.2.2. Die Wiederaufnahme bei Fehlerfällen

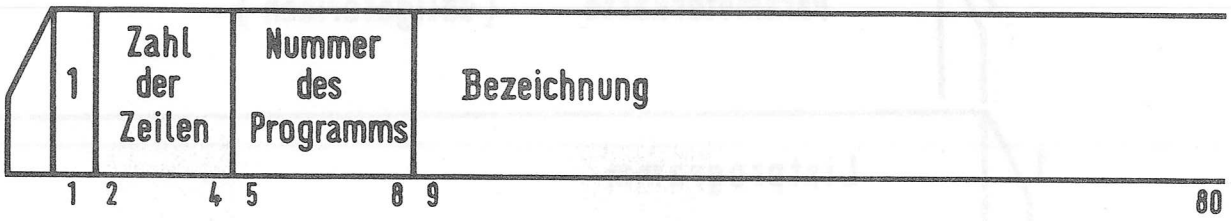
s. 7.5.

7.3.3. Das Listen eines Arbeitsprogrammes

Um ein Arbeitsprogramm zu listen, wird das Programm 'Listen' benötigt. Hinter dem Programm 'Listen' muß eine Parameterkarte liegen und dahinter das zu listende Programm.

Die Parameterkarte

Eine Karte mit 1 in Spalte 1 bewirkt einen Papiersprung und die Erstellung einer oder mehrerer Parameterkarten, die die Anfangsadresse und die Endadresse des eingespeicherten Programmes angeben. Diese Karten werden für das Verdichten des Arbeitsprogrammes benötigt.



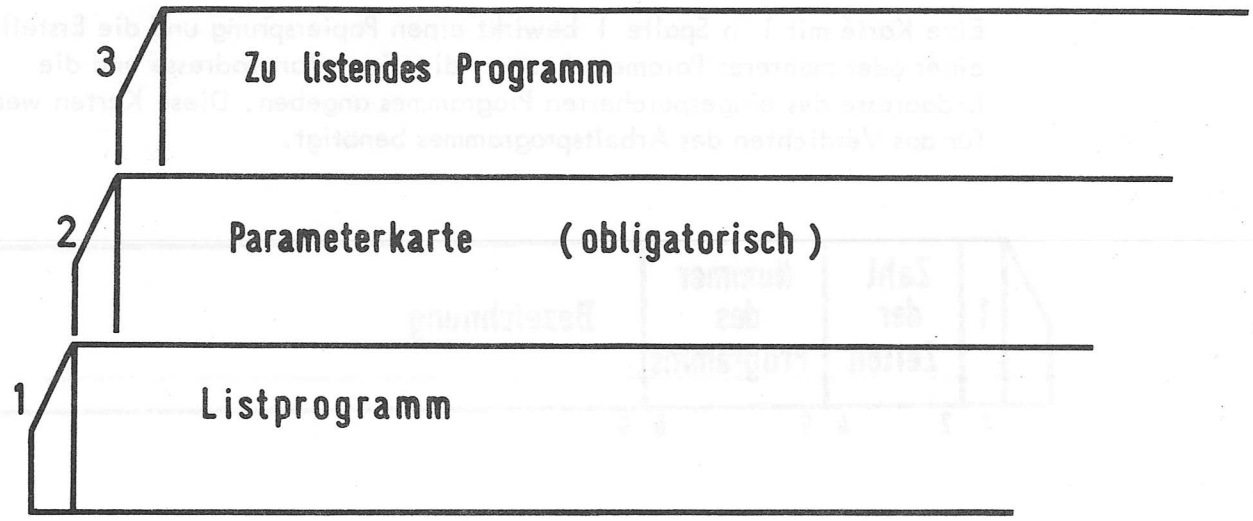
Sind die Spalten 2 bis 4 gelocht, erfolgt ein Papiersprung, der in dem Moment ausgeführt wird, wo die Anzahl der bedruckten Zeilen mit dieser gelochten Zahl übereinstimmt. In dieser Zahl ist eine Zeile für die Überschrift, eine Leerzeile zwischen dieser Titelzeile und der ersten Listzeile des zu listenden Programms und die erforderlichen Zeilen für das Programm enthalten.

Eine Lochung in den Spalten 5 bis 8 bewirkt das Stanzen der Parameterkarten für das Verdichtungsprogramm (MEFE). Am Ende der Liste erhält man:

- eine oder mehrere Parameterkarten MEFE,
- eine Startkarte.

Die Bezeichnung des Programmes wird auf der Höhe der Titelzeile angeschrieben. Wenn keine der angebotenen Möglichkeiten eingehalten werden, ist es möglich, eine Leerkarte als Parameterkarte zu verwenden, da diese Parameterkarte obligatorisch ist.

Am Anfang liegt das Programm 'Listen', anschließend eine Parameterkarte und danach das zu listende Programm.



7.3.3.1. Die Handhabung

- Taste LØAD drücken.
- Die Karten sind in der obengenannten Reihenfolge in den Kartenleser zu legen.
- Drücken der Taste RES.

Anschließend werden die Karten gelesen und das Programm gelistet.

Nach dem Lesen der letzten zu listenden Karte werden die Parameterkarten für das Verdichten (MEFE) gestanzt, wenn eine Parameterkarte mit 1 in Spalte 1 und der Programmnummer vorhanden ist.

Die Wiederanläufe nach Unterbrechungen 'Papier Ende' und 'Ende des Stanzens' sind vorgesehen.

1) Wiederanlauf nach Papier Ende

Der Drucker blockiert auf der Höhe der ersten zu druckenden Zeile des Blattes nach Abfühlen des ersten Sprungloches, wenn der erste Sprung ausgeführt ist.

Die Lampe 'PR' leuchtet.

Folgende Handgriffe sind erforderlich:

- Wechseln des Papiers und richtige Einstellung des ersten Blattes auf die Höhe der ersten zu druckenden Zeile des Blattes.
- Einklemmen des letzten Blattes zwischen die Kontakte für Papierende.
- Druck auf die Taste RES des Druckers.

Bei einer Liste ohne Sprung bei vollem Blatt ist es erforderlich, das Papier zu wechseln, nachdem Papierende festgestellt wurde.

2) Wiederanlauf nach Unterbrechung des Stanzers.

s. 7.5.

7.3.4. Das Verdichten eines Programmes

Um ein Programm zu verdichten, ist es erforderlich, das Programm MEFE (Verdichtungsprogramm), hinter dem die Parameterkarten liegen, zu verwenden, wenn das Programm einmal in den Kernspeicher mit Hilfe des Ladeprogramms eingespeichert wurde.

Die Parameterkarten sind die Karten, die bei der Liste des Programms erstellt wurde.

Die Karten müssen in der unten angegebenen Reihenfolge liegen: (s.S. 220)

1) Einspeichern des Programms

- Löschprogramm
- Ladeprogramm
- Arbeitsprogramm (ohne Startkarte).

2) Das Verdichten des Programms

- Verdichtungsprogramm
- Parameterkarten
- Startkarte.

Diese Arbeit besteht aus zwei Arbeitsgängen:
- dem Einspeichern des Programmes mit Hilfe des Ladeprogrammes und und anschließend dem Verdichten dieses Programmes.

7.3.4.1. Die Handhabung

1) - Drücken der Taste LOAD.
   - Einlegen der Karten in den Leser in der richtigen Reihenfolge.
   - Drücken der Taste RES.

Nachdem alle Karten gelesen sind:

2) - Druck auf die Taste LOAD
   - Eingabe der Karten in den Leser in der richtigen Reihenfolge.
   - Drücken der Taste RES.

   (Prüfen, ob Leerkarten im Stanzer liegen)

Wiederanlauf nach Unterbrechung

s. 7.5.

7.4. Wiederanlauf nach Unterbrechungen bei Kartenstau auf dem Leser

Stau am Schlitz

- Herausnehmen der Karten.
- Evtl. Erneuern der Karte, die den Stau verursacht hat und diese Karte neu an den Kopf des Kartenpaketes legen.
- Neues Füllen des Zufuhrmagazins.
- Drücken der Taste RES.

Stau auf der Kartenbahn oder am Ablagefach

Wenn es erforderlich ist, das Kartenpaket aus dem Zufuhrmagazin nehmen.

- Öffnen der Verkleidung der Kartenbahn.
- Evtl. das Plastikführungsband herausnehmen. Um dies zu bewerkstelligen, ist der bewegliche Block der Bahn zu öffnen, indem man die Verriegelung öffnet, die sich über dem Block befindet und den beweglichen Block mittels eines Knopfes in der Mitte des Blocks herauszieht.
- Entfernen der zerstörten Karten.

Erneutes Einlegen des Plastikbandes. Um das Band einzulegen, nimmt man das Band mit beiden Händen und führt es um den beweglichen Block, so daß die Lochungen für die Synchronisation nach unten liegen. Nun läßt man das Band herabgleiten, bis es auf den Führungsrollen liegt.

Den Block neu verriegeln

Verschieben des Führungsbandes mittels Daumen und Zeigefinger nach rechts, bis die eine der drei Richtschlitze mit dem Führungsloch am beweglichen Teil des Blocks genau übereinstimmt.

Es ist darauf zu achten, daß das Führungsband nicht in umgekehrter Richtung verschoben wird, da man dabei die Führungsschlitze beschädigt.

Anschließend läßt man eine Leerkarte durch den Leser laufen, indem man auf die Taste EJ des Lesers drückt, um zu sehen, ob noch ein Rest aus einer zerstörten Karte in der Lesebahn steckt und ob das Führungsband genau abläuft.

Danach den Leser verschließen.

Drücken der Taste RES.

Bei vollem Fach:

Herausnehmen des Kartenpaketes und Drücken auf die Taste RES.

7.5. Wiederanlauf nach Unterbrechung beim Stanzen in den Software-Programmen

Die Lampe PU leuchtet.

- Drücken auf die Taste PI.
- Drücken der Taste EJ, um die letzte, korrekt gelochte Karte auszusteuern.
- Herausnehmen der Karten aus dem Ablagefach.
- Drücken auf die Taste RES. Nun wird das Ende der fehlerhaften Karte in eine Leerkarte gestanzt und am Ende dieses Lochvorgangs wird auf der Sichttafel 3 x 1 sichtbar.
- Zweimal Taste EJ drücken.
- Herausnehmen der fehlerhaften Karten aus dem Ablagefach.
- Drücken der Taste CLB.
- Die Arbeit wird normal wieder aufgenommen. In jedem Falle ist die erste Karte im Ablagefach eine Leerkarte.

8. Programmunterlagen

8.1. Allgemeines

Die Programmunterlagen werden vom Programmierer erstellt.

Die in den folgenden Abschnitten beschriebenen Formulare und Listen sind zwar für die Programmierung erforderlich, gehören aber nicht alle zu den obligatorischen Unterlagen. Diese sind in Abschnitt 8.4. besonders erwähnt.

8.2. Die verwendeten Formulare

8.2.1. Block- und Flußdiagramm

Auf diesem Formular wird für jede Arbeit der detaillierte Ablauf aufgezeichnet. Für eine Arbeit können mehrere Blätter verwendet werden.

Bestell-Nr.: 218 (Abb. siehe Seite 247)

8.2.2. Karteneinteilung

Alle Karten, die in einem Programm Verwendung finden, werden hier mit ihrer Einteilung aufgezeichnet. Es müssen sowohl die Eingabekarten wie auch die Summenkarten berücksichtigt werden.

Bestell-Nr.: 202 (Abb. siehe Seite 248)

8.2.3. Druckbild

Auf diesem Formular wird das gewünschte Druckbild schreibstellengerecht eingetragen. Die senkrechte Zeilennumerierung erleichtert das spätere Lochen des Vorschubstreifens.

Bestell-Nr.: 375 (Abb. siehe Seite 249)

8.2.4. Programmliste

Während des Testens entsteht durch das Softwareprogramm "Programmkartenlisten" auf Blankopapier eine Liste der Programmbefehle. Eine Beschreibung dieser Liste ist in dem entsprechenden Absatz in Kapitel 7 gegeben.

Diese Liste ersetzt nach Fertigstellung des Programmes das "Programmformular" (s. 8.2.8.).

(Abb. siehe Seite 255)

8.2.5. Kernspeicherausdruck

Während des Testens wird man häufig einen Kernspeicherausdruck vornehmen.

Nach dem Test, wenn das Programm fertiggestellt ist, gehört ein Kernspeicherausdruck zu den Programmunterlagen. Dieser Ausdruck erfolgt unmittelbar nach dem Laden des Programmes.

Der techn. Kundendienst hat mit dieser Unterlage die Möglichkeit, bei evtl. Störungen die korrekte Eingabe zu kontrollieren.
(Abb. siehe Seite 256)

8.2.6. Bedienungsanweisung

Auf diesem Formular müssen alle notwendigen Angaben enthalten sein, die der Maschinenbediener am Anfang und während der Arbeit benötigt. Ebenfalls müssen die Anweisungen für sein Verhalten bei Störungen gegeben werden.

Bestell-Nr.: 1003 (Abb. siehe Seite 250)

8.2.7. Kernspeicherbelegung

Dieses Blatt gibt einen Überblick über die Kernspeicherbelegung mit der Belegung durch das Ladeprogramm.

Dieses Blatt, evtl. in Verbindung mit dem neutralen Speicherbelegungsformular (Best.-Nr. 233) dient dazu, eine Übersicht über die Belegung der Speicherstellen durch Programm, Tabellen und Konstanten zu geben.

Dieses Blatt soll die Belegung nach dem Laden eines Programmes am Anfang der Arbeit wiederspiegeln.

Bestell-Nr.: 1010 (Abb. siehe Seite 251)

1016 für 2500 Bytes
1014 für 5000 Bytes
1015 für 10000 Bytes

8.2.8. Programmformular

Dieses Formular ist das Grundformular der Kodierung eines Programmes. Es ist gleichzeitig Lochbeleg. Nach Fertigstellung des Programmes werden diese Formulare durch die Programmliste ersetzt.

Bestell-Nr. 1009 (Abb. siehe Seite 252)

8.2.9. Allgemeine Beschreibung

Es ist empfehlenswert eine genaue Beschreibung der Arbeit bei den Programmunterlagen zu haben. Diese Beschreibung kann durch die Aufgabenstellung ersetzt werden.

8.2.10. Symbole und Register

Dieses Formular kann als Ergänzung zu dem Formular "Kernspeicherbelegung" dienen, indem es eine Übersicht über die belegten numerischen Register gibt.

Gleichzeitig können auf ihm die verwendeten LEVEL notiert werden, um bei umfangreichen Programmen die Gefahr einer Doppelbelegung zu vermeiden.

Bestell-Nr.: 1001 (Abb. siehe Seite 253)

8.3. Testunterlagen

Unabhängig von den Programmunterlagen gehört zu jeder Arbeit ein Satz Probe- oder Testkarten und ein mit diesen Karten geschriebenes Druckformular. Mit diesen Unterlagen kann der Operateur vor jeder Programmeingabe die korrekte Arbeit der Maschine prüfen.

8.3.1. Probekarten und 80/80-Liste

Nach dem Laden eines Programmes werden zuerst diese Karten in die Maschine eingelesen. Der Kartensatz wurde vom Prommierer erstellt und berücksichtigt alle in der wirklichen Arbeit vorkommenden Möglichkeiten. Jede Programmserie (Verzweigung) muß von ihm angesprochen werden.

Die 80/80-Liste dieser Karten, wird als Unterlage für den Fall aufbewahrt, daß die Probekarten neu gelocht werden müssen.

8.3.2. Druckformular und Summenkarten

Nach dem Durchlauf dieser Karten durch die Maschine ist das erhaltene Druckbild und die Summenkarten mit der Originalvorlage auf Übereinstimmung zu vergleichen.

8.4. Formularmuster

Auf den folgenden Seiten sind die in Abschnitt 8.2. beschriebenen Formulare in verkleinerter Form abgebildet. Von diesen Formularen gehören die folgenden obligatorisch zu den Programmunterlagen:

- Block- und Flußdiagramm
- Karteneinteilung
- Kernspeicherbelegung
- Druckbild
- Programmliste
- Kernspeicherausdruck
- Bedienungsanweisung
- Allgem. Beschreibung
- Probekarten und 80/80-Auflistung
- Druckformulare und Summenkarten als Muster einer Arbeit mit den Probekarten.

9.65 / 218

Darstellung		Text	Verweisung
0	1	2	3

Geschäftsstelle :
Firma :

Name des Programms :
Nr.:

Erstellt am :
durch :
berichtigt am:
durch :

BLOCKDIAGRAMM
FLUSSDIAGRAMM

BULL GENERAL ELECTRIC

Symbole: Funktion - 1 Ausgang, Entscheidung - 2 Ausgänge (linker Ausgang - Bedingung erfüllt / rechter Ausgang - Bedingung nicht erfüllt)

Geschäftsstelle:
Firma:

Name des Programms:
Nr.

Erstellt am:
durch:
berichtigt am:
durch:

KARTENEINTEILUNG

BULL GENERAL ELECTRIC

8.66 / 202

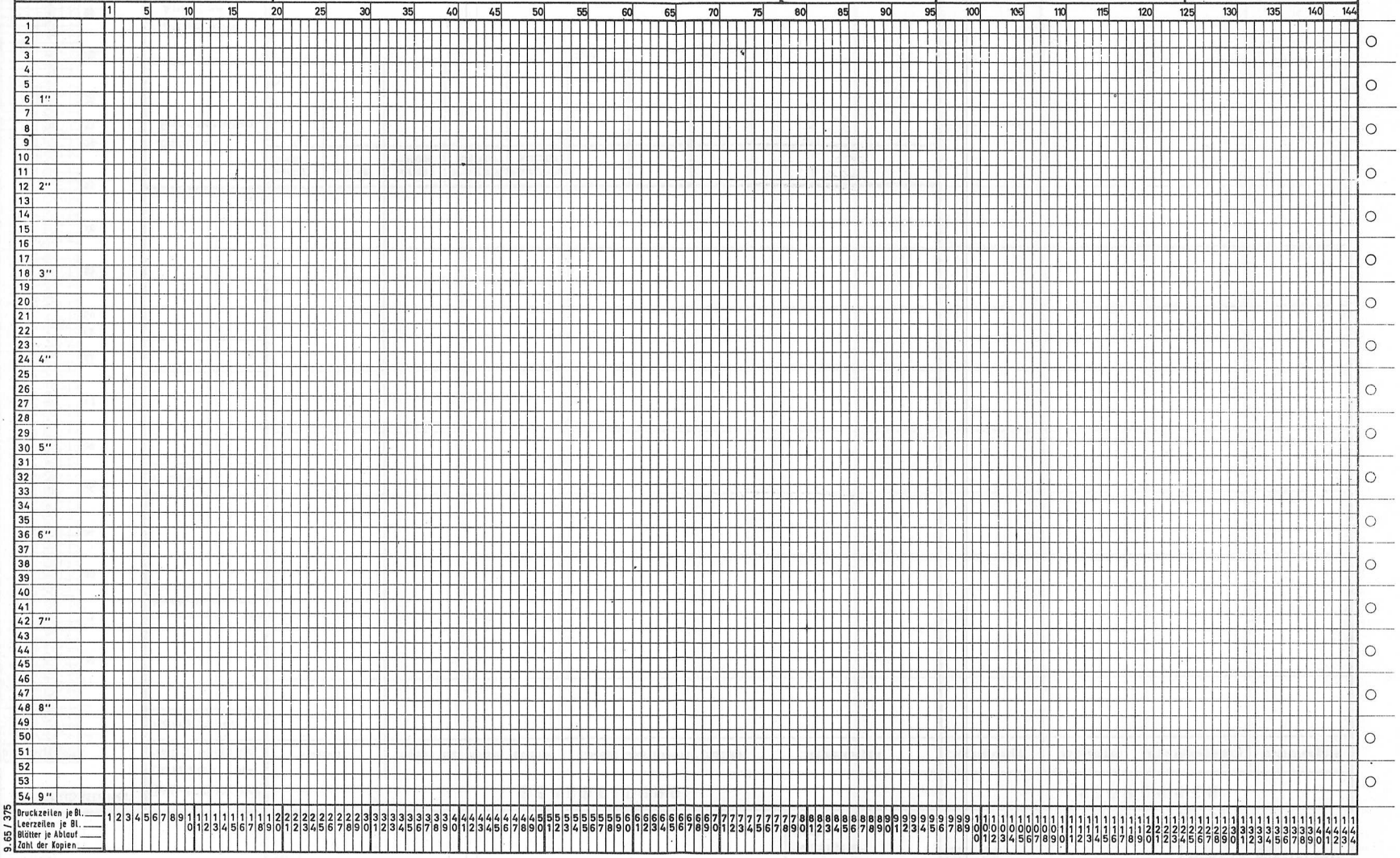
Geschäftsstelle:
Firma:

Name des Programms:
Nr.:

Erstellt am:
durch:
berichtigt am:
durch:

DRUCKBILD 10 Stellen je Zoll
Vorschub 1/2
Druckbild Nr.:

BULL
GENERAL ELECTRIC



Bei Druckvorlage gelten nicht die Abmessungen dieses Formulars, sondern folgende Maße für 1 Druckstelle: Höhe 1/6 Zoll
Breite 1/10 Zoll

Geschäftsstelle:
Firma:

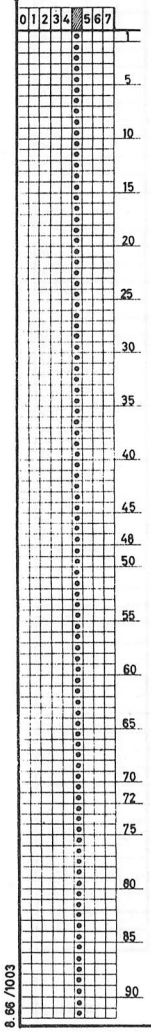
Name des Programms:
Nr.:

Erstellt am:
durch:
berichtigt am:
durch:

BEDIENUNGSANWEISUNG
GE 55

BULL GENERAL ELECTRIC

Vorschubstreifen



8.66/1003

Bezeichnung der benutzten Formul. u. Karteien	BEARBEITUNG			STÖRUNGEN			
	Kartenart	Eingabe über numer. Tastatur	Eingabe über alpha-numer. Tastatur	Bemerkungen	Leuchtanzeige	Randeinheit	Art d. Wiederanlaufs	Tastatureingabe
Kartenleser								
Kartenstanzer								
Drucker								

benutzte Randeinheiten	Kartenleser	Drucker	Kartenstanzer	numerische Tastatur	alpha-numerische Tastatur							

Allgemeine Angaben zum Programm												

Geschäftsstelle:

Firma:

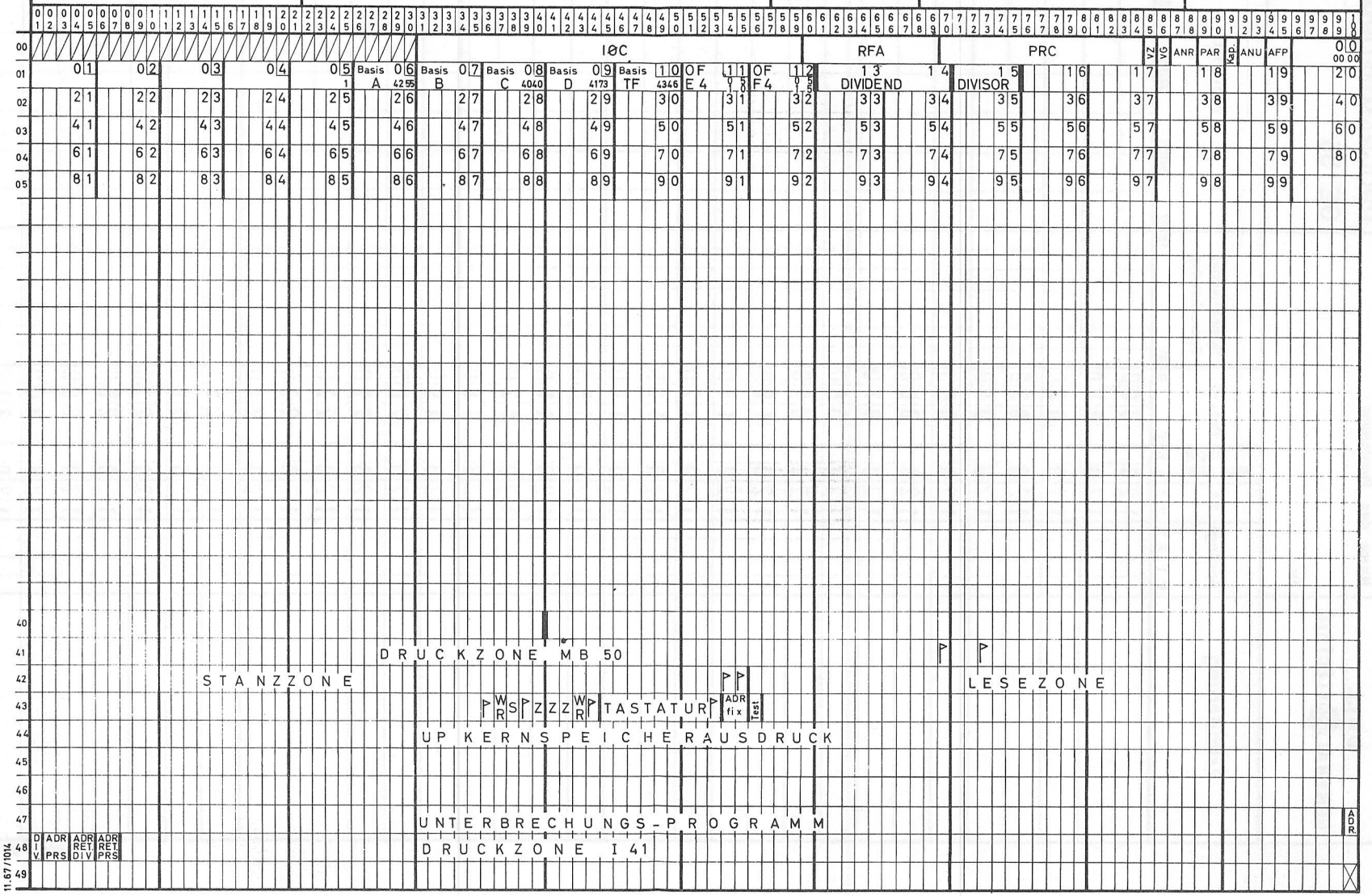
Name des Programms:

Nr.:

Erstellt am:
durch:
berichtigt am:
durch:

KERNSPEICHERBELEGUNG
GE 55 5000 Bytes

BULL GENERAL ELECTRIC



11.67/1014

BULL GENERAL ELECTRIC

Name des Programms :

Nr.:

PROGRAMMFORMULAR GE 55

Erläuterung	Sprünge F 1	Sprünge F 2	KA 2 3	Zahl der Bytes 4 5	Ord.-Nr. 6 9	Symbolischer Operationstyp 10 16	Befehl 17 20	21 24	25 28	29 32
			12	0	000					
			12	0	010					
			12	0	020					
			12	0	030					
			12	0	040					
			12	0	050					
			12	0	060					
			12	0	070					
			12	0	080					
			12	0	090					
			12	0	100					
			12	0	110					
			12	0	120					
			12	0	130					
			12	0	140					
			12	0	150					
			12	0	160					
			12	0	170					
			12	0	180					
			12	0	190					
			12	0	200					
			12	0	210					
			12	0	220					
			12	0	230					
			12	0	240					
			12	0	250					
			12	0	260					
			12	0	270					
			12	0	280					
			12	0	290					
			12	0	300					
			12	0	310					

12.66 / 1009

Buchstabe O = Ø, Ziffer Null = 0

Ref.-Nr.: 23.20.001 D
Juli 1968

Geschäftsstelle:
Firma:

Name des Programms:
Nr.:
Phase:

Version:

Erstellt am:
durch:
berichtigt am:
durch:

SYMBOLE und REGISTER
GE 55

BULL GENERAL ELECTRIC

Feldbezeichnung	Symbol	Register	Feldbezeichnung	Symbol	Register	Feldbezeichnung	Symbol	Register

8.66 / 1001

Geschäftsstelle :
Firma :

Organisationsgebiet :
Programmkreis :
Datei :

Erstellt am :
durch :
berichtigt am :
durch :

Lochstreifen
Magnetband
Magnetplatte
Kernspeicher

SPEICHERBELEGUNG

BULL
GENERAL ELECTRIC

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

3.66 / 233

```
C.NK.  NUM.  IMP.TO.      CODES

15        5  1101         1101
14 02    10  1101  LEVEL  F201
12 03    15  1101  MRR    E12406
12 06    20  1104  INC    8200|0440|0220
12 04    25  1110  IOC    53|01|9A|12
12 04    30  1114  IOC    53|01|92|10
12 04    35  1118  IOC    53|01|9A|12
12 04    40  1122  MBR    21|02|97|25
12 04    45  1126  IOC    53019210
12 04    50  1130  MBR    21029726
12 03    55  1134  MRR    E11902
14 02    60  1137  LEVEL  F202
12 03    65  1137  MSRDR  E30014
12 03    70  1140  MRR    E11501
12 03    75  1143  ADD    A12501
12 06    80  1146  MVC    800110000235
12 03    85  1152  MRR    E11501
12 03    90  1155  ADD    A12601
12 06    95  1158  MVC    800110000240
12 03   100  1164  MSRDR  E30030
12 03   105  1167  MRR    E11501
14 02   110  1170  LEVEL  F203
12 06   115  1170  MVC    800110010165
12 03   120  1176  ADD    A11329
12 03   125  1179  ADD    A10514
12 03   130  1182  ADD    A10501
12 03   135  1185  CML    911425
12 06   140  1188  JIBT   043C0086F103
12 03   145  1194  MRR    E11501
12 03   150  1197  MSRDR  E30014
14 02   155  1200  LEVEL  F204
12 06   160  1200  MVC    800110010165
12 03   165  1206  ADD    A11330
12 03   170  1209  ADD    A10514
12 03   175  1212  ADD    A10501
12 03   180  1215  CML    911426
12 06   185  1218  JIBT   043C0086F104
12 03   190  1224  MRR    E11501
12 03   195  1227  MRR    E10906
12 06   200  1230  INC    8200141210 88
12 04   205  1236  IOC    53019606
12 02   210  1240  TR1    5106
14 02   215  1242  LEVEL  F205
12 06   220  1242  MVC    800260826162
12 06   225  1248  INC    822060826080
12 03   230  1254  ADD    A11706
12 04   235  1257  IOC    53019606
12 02   240  1261  TR1    5106
12 03   245  1263  MSRDR  E30014
12 03   250  1266  MRR    E10906
12 03   255  1269  ADD    A12906
```

Programmliste des GE-55

```
0026 2020321100 2191934680 0000008000 0000800000 0080000000
0051 8000000081 0000000000 0000000000 0000008531 0000000000
0076 0000000010 5040302000 3C92459738 0000009672 0000000000
0101 0000000801 0000000270 0000000005 0000000334 0000000001
0126 0000009255 0000000000 0000009040 0000009173 0000009346
0151 800EF40151 110FF40156 0000000000 0000000000 0000000000
0176 0000000000 0000000000 0000000000 0000000000 0000000000
0201 0000000000 0000000000 0000000000 0000000000 0000000000
0226 0000000000 0000000000 0000000000 0000000000 0000000000
0251 0000000060 0000000600 0000000750 0000000006 0000000000
0276 0000000000 0000000000 0000000000 0000000000 0000000000
0301 0000000000 0000000000 0000000000 0000000000 0000000000
0326 0000000000 0000000000 0000000000 0000000000 0000000000
0351 0000000000 0000000000 0000000000 0000000000 0000000000
0376 0000000000 0000000000 0000000000 0000000000 0000000000
0401 0000000000 0000000000 0000000000 0000000000 0000000000
0426 0000000000 0000000000 0000000000 0000000000 0000000000
0451 0000000000 0000000000 0000000000 0000000000 0000000000
0476 0000000000 0000000000 0000000000 0000000000 0000000000
0501 0000000000 0000000000 0000000000 0000000000 0000000000
0526 0000000000 0000000000 0000000000 0000000000 0000000000
0551 0000000000 0000000000 0000000000 0000000000 0000000000
0576 0000000000 0000000000 0000000000 0000000000 0000000000
0601 534045475A 414C202D20 4252415545 5245495341 43484B4F4E
0626 54454E2020 202A202020 4B52454449 544F52454E 3139363848
0651 544F2E4E52 2E44415455 4D20472E4B 542E42454C 2E234A2D53
0676 5445585420 2F2053594D 424F4C534B 4F4E544F53 4F4C4C2020
0701 4D57535448 4142454E4B 5753545355 4D4D452020 534F4C4C20
```

UMSCHLÜSSELUNGSTABELLE

Zeichen	Zwischencode	Lochcode			ISO code	Zeichen	Zwischencode	Lochcode			ISO code
↛	20				20	P	57	11	7		50
0	30	°	0		30	Q	58	11	8		51
1	31		1		31	R	59	11	9		52
2	32		2		32	S	22	0	2		53
3	33		3		33	T	23	0	3		54
4	34		4		34	U	24	0	4		55
5	35		5		35	V	25	0	5		56
6	36		6		36	W	26	0	6		57
7	37		7		37	X	27	0	7		58
8	38		8		38	Y	28	0	8		59
9	39		9		39	Z	29	0	9		5A
•	4B	12	8	3	2E	$	5B	11	8	3	24
/	21	0		1	2F	⌋	4C	12	8	4	5D
#	3B		8	3	23	@	3C		8	4	40
%	2C	0	8	4	25	&	40	12			26
–	50	11			2D	,	2B	0	8	3	2C
*	5C	11	8	4	2A	'	5F	11	8	7	27
A	41	12	1		41	(	4D	12	8	5	28
B	42	12	2		42	<	4E	12	8	6	3C
C	43	12	3		43	\	4F	12	8	7	5C
D	44	12	4		44	↑	70	11	0		5E
E	45	12	5		45	;	5E	11	8	6	3B
F	46	12	6		46	=	2D	0	8	5	3D
G	47	12	7		47	"	2E	0	8	6	22
H	48	12	8		48	!	2F	0	8	7	21
I	49	12	9		49	>	3E		8	6	3E
J	51	11	1		4A	?	3F		8	7	3F
K	52	11	2		4B	←	2A	0	8	2	5F
L	53	11	3		4C	:	3D		8	5	3A
M	54	11	4		4D	+	60	12	0		2B
N	55	11	5		4E	⊏	3A		8	2	5B
O	56	11	6		4F	)	5D	11	8	5	29

I 41 halbe Druckgeschwindigkeit

SPALTE	0	1	2	3	4	5	6	7	ISO CODE
Binär darstell.	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1	ZEILE
0000	⧹		↛	0	@	P			0
0001			!	1	A	Q			1
0010			"	2	B	R			2
0011			#	3	C	S			3
0100			$	4	D	T			4
0101			%	5	E	U			5
0110			&	6	F	V			6
0111			'	7	G	W			7
1000			(	8	H	X			8
1001			)	9	I	Y			9
1010	Z		*	:	J	Z			A
1011	WRZ		+	;	K	⊏			B
1100	S		,	<	L	\			C
1101	WR		–	=	M	⌋			D
1110	V		.	>	N	↑			E
1111	N		/	?	O	←			F

ANSCHLUSSCODES FÜR RANDEINHEITEN

CC	R	Funktion
96	06	Kartenleser
AA	09	Kartenstanzer
CA	08	Drucker MB 50
9A	12	Freigabe num. Tastatur N
9A	11	Freigabe Sichtanzeige V
94	R	Alphatastatur A
9A	10	Übertrag Zentralspeicher Puffer
92	10	Übertrag Puffer Zentralspeicher
10	R	Trommel Lesen
12	R	Trommel Schreiben
13	R	Trommel Auswahl der Bahn
11	R	Trommel Test
1E	08	Drucker I 41 Druck
1D	08	Drucker I 41 Test

FUNKTIONSSCHLÜSSEL I 41

48 Z	64 Z	Funktion
3C	4C	keine Zwischenzeile
3D	4D	Einfacher Zeilenvorschub
3E	4E	Doppelter Zeilenvorschub
3F	4F	Dreifacher Zeilenvorschub
21	51	Papiervorschub und Stop auf Kanal A
22	52	Papiervorschub und Stop auf Kanal B

Testzeichen an Basisadresse: 10 = nicht volles Blatt
30 = volles Blatt

GE 55 Befehle

Sprünge

NØP	00			
JRT	02	F1	xx	
JIERT	04	K	Bxxx	F1xx
JIURT	05	K	Bxxx	F1xx
LEVEL	F2	xx		

Überträge

MRR	E1	R1	R2	
MDRR	E2	RR1	RR2	
MSRDR	E3	R1	RR2	
EXR	E5	R1	R2	
EXDR	E6	RR1	RR2	
LRN	60	N	Bxxx	R(R)
MVC	80	N	Bxxx	Bxxx
MVIC	8A	R	Bxxx	Bxxx
PKH	84	N	Bxxx	Bxxx
UPH	85	N	Bxxx	Bxxx
INC	82	K	Bxxx	Bxxx

Logische Operationen

CMD	91	R1	R2	
CMDD	92	RR1	RR2	
CMC	88	N	Bxxx	Bxxx
NI	61	K	Bxxx	
ØI	63	K	Bxxx	

Arithmetische Operationen

ADD	A1	R1	R2
ADDD	A2	RR1	RR2
SBD	B1	R1	R2
SBDD	B2	RR1	RR2
RØUND	71	K	RR
MPD	C1	R1	RR2
DVD	D1	R1	RR2

Verschiebungen

SLD	70	N	RR
SRD	72	N	RR

EIN-/AUSGABE - Überträge

MR8	12	N	R	Dc		M7R	31	N Db...
MR8G	1E	N	R	Dc	KN'	M78	32	N Db...
MR9	14	N	R	Dd		M78G	3E	N Db...
MR89	16	N	R	Dc	Dd	M7R8	33	N Db...
						M79	34	N Db...
M6R	21	N	Da	R		M7R9	35	N Db...
M68	22	N	Da	Dc		M789	36	N Db...
M68G	2E	N	Da	Dc	KN'	M7R89	37	N Db...
M6R8	23	N	Da	R	Dc			
M69	24	N	Da	Dd				
M6R9	25	N	Da	R	Dd			
M689	26	N	Da	Dc	Dd			
M6R89	27	N	Da	R	Dc	Dd		

wie M6...

EIN-/AUSGABE

IØC	53	N	CC1	R1--CCn	Rn
IØIC	54	N	RCC1	R1--CCn	
TR1	51	R	H14.012 → ISO		
TRB	5B	R	ISO → H14.012		
TRD	5D	R	(141 → 64 Zeichen)		

Hilfsbefehle

PRSTØ	78		
LØAD	7D		
RLA	6A	00	Bxxx
KHLT	57		

Multiprogramming

PRØ	42	N
FREE	44	N
START	41	N
STØP	40	

Reservierte Speicherstellen

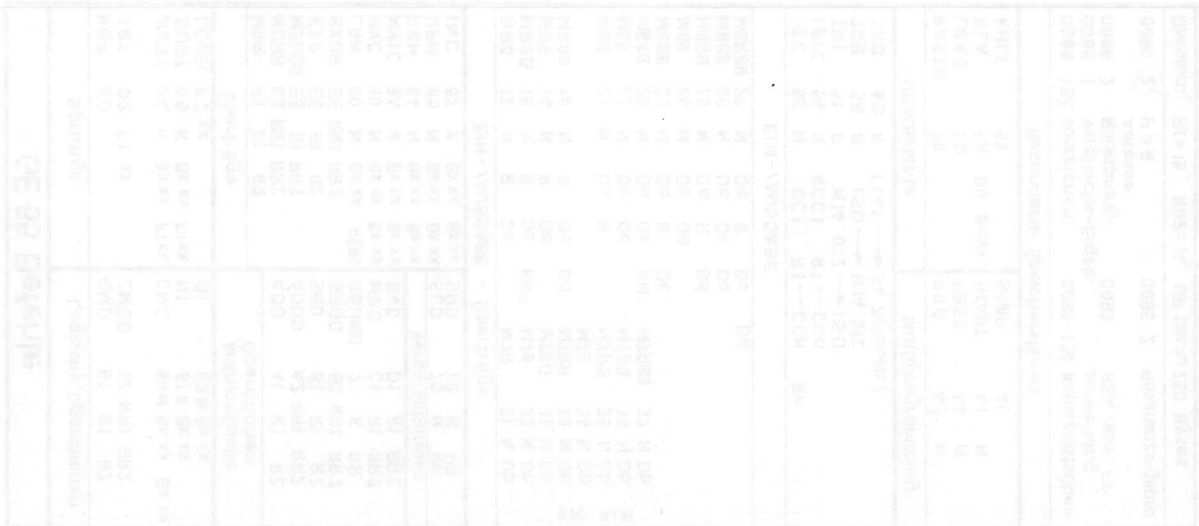
0085	1 St.	Vorzeichen	0091	1 St.	Kapazitätsüberschreitung
0086	1	Vergleichs-Ergeb.			
0088	2	Rücksprung-Adresse	0093		RSA nach FP
0090	(2)	PAR	0095	2	Fehleranzeigepro.

Division: R1 = 15; RR2 = 14; UP Div. ≈ 250 Bytes

2 Byte, daher Adressen 89, 90

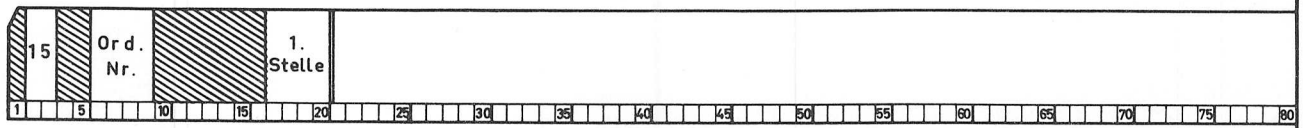
* Halt: Nur wirksam, wenn Taste gedrückt ist.

Ref.-Nr.: 23.20.001 D
Juli 1968

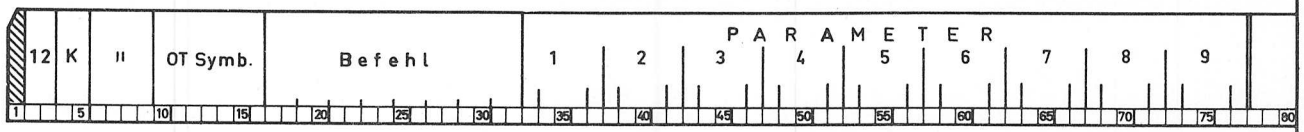


Geschäftsstelle: Firma:	Name des Programms: Nr.	Erstellt am: durch: berichtigt am: durch:	KARTENEINTEILUNG	BULL GENERAL ELECTRIC

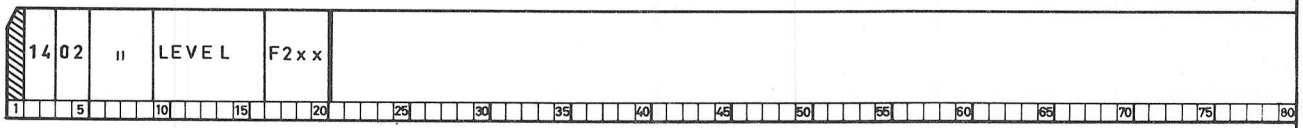
KA 15 PROGRAMMLEITKARTE
Sp. 6 - 9 Ordnungsnummer
Sp. 17 - 20 Speicherstelle des ersten einzugebenden Befehls



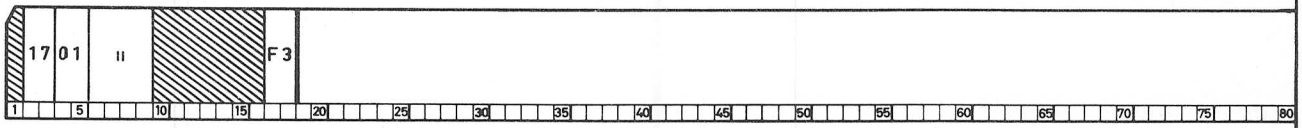
KA 12 BEFEHLSKARTE
Sp. 4 - 5 Anzahl der Stellen
Sp. 6 - 9 Ordnungsnummer
Sp. 10 - 16 symbolischer OT
Sp. 17 - 32 Befehlscode



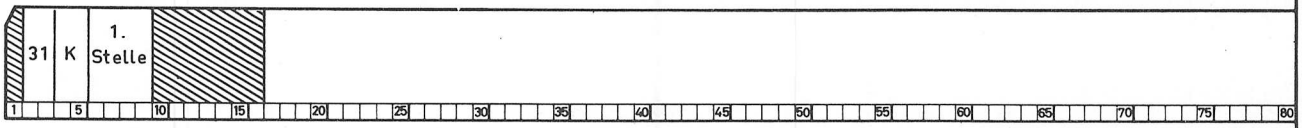
KA 14 NIVEAUKARTE
Sp. 6 - 9 Ordnungsnummer
Sp. 17 - 2o F2 und lfd. Nr. des Niveaus (von 00 - 99)



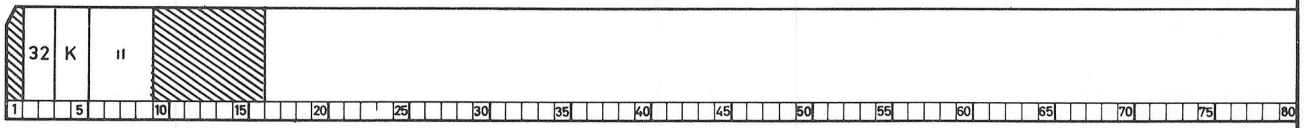
KA 17 PROGRAMMENDKARTE
Sp. 6 - 9 Ordnungsnummer



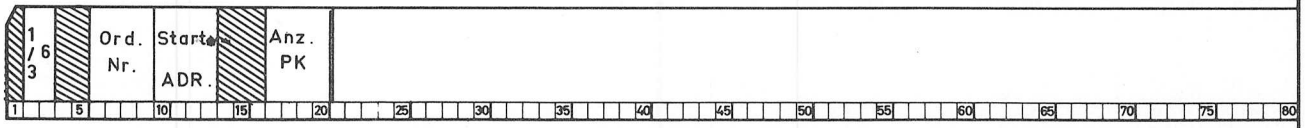
KA 31 KONSTANTENKARTE
Sp. 4 - 5 Anzahl der Stellen
Sp. 6 - 9 1. Stelle im Speicher
Sp. 17 - 80 Konstanten im Format 1 (1 Spalte je Byte)



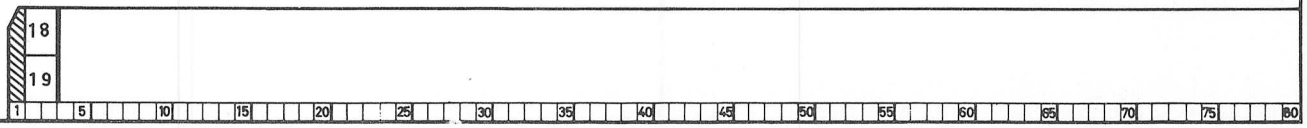
KA 32 KONSTANTENKARTE
wie KA 31, jedoch Konstanten im Format 2 (2 Spalten je Byte)



KA 16/36 PROGRAMMSTARTKARTE
Sp. 6 - 9 Ordnungsnummer
Sp. 10 - 13 Stelle der ersten Adresse des Befehls, mit dem angefangen werden soll.



KA 18/19 PROGRAMMLISTKARTE
Ka 18 Startkarte
KA 19 Stopkarte



8.69/202