ZUSE Z 31

Programmierungsanleitung

ZUSE KG · BAD HERSFELD

Elektronische Rechenanlagen

Programmierungsanleitung ZUSE Z 31

Ausgabe April 1964

Inhaltsübersicht

1. Einführung

Die Grundaufgabe der Datenverarbeitung besteht darin, aus vorhandenen Datenfolgen (Zahlen, Text) nach bestimmten, von Fall zu Fall verschiedenen Regeln neue Datenfolgen zu gewinnen. Die Übersetzung dieser Regeln in eine Folge von Instruktionen, sogenannten Befehlen , die die Datenverarbeitungsanlage lesen und ausführen kann, bezeichnet man als Programmieren.

Diese Befehle und die zu behandelnden Daten müssen für die Eingabe in die Maschine in einer für sie verständlichen Sprache, der sogenannten Programmierungssprache oder Externcode, auf einem geeigneten Medium (z.B. Lochstreifen oder Lochkarte) dargestellt werden. Die Maschine übersetzt die gelesenen Informationen in ihre eigene Sprache (Interncode), speichert sie und kann dann mit der Verarbeitung beginnen.

Der Externcode kann grundsätzlich beliebig aus den bestehenden Systemen gewählt werden; Voraussetzung ist nur, daß die Rechenanlage mit einem entsprechenden Programm zur Umschlüsselung in die maschineninterne Sprache ausgestattet wird.

Für die Standardausführung der ZUSE Z 31 wurde ein Externcode gewählt, der dem bewährten Freiburger Code für die ZUSE Z 22 und ZUSE Z 23 verwandt ist. Ein Leseprogramm übersetzt diesen Externcode in den ziffernverschlüsselten Interncode der Anlage und speichert gelesene Informationen fortlaufend auf Schnellspeicher. Das Programm wird für jede Maschine mitgeliefert.

Eingabemedien sind Lochstreifen oder Lochkarten. Programme und Daten werden auf einem besonderen Herstellgerät gelocht, das unabhängig von der Rechenanlage arbeitet. Für die Standardform der Anlage ZUSE Z 31 ist ein Lochstreifengerät vorgesehen, das den "ZUSE-Code" verwendet, einen 8-Kanal-Code, der auf den maschineninternen Code der ZUSE Z 31 aufgebaut ist und daher hohe Ein- und Ausgabegeschwindigkeiten gestattet.

1.1 Informationsdarstellung in der Zentraleinheit ZUSE Z 31

Die Informationseinheit der ZUSE Z 31 ist das "Wort", das aus 44 Bits besteht, eingeteilt in 11 Gruppen zu je 4 Bits. Die Information wird im Wort binär-dezimal verschlüsselt, d.h. in jeder Gruppe kann durch eine Kombination von 4 Ja-Nein-Werten (L und 0) eine Dezimalziffer dargestellt werden. Die sich somit ergebenden 11 Dezimalstellen werden mit D1 - D11 bezeichnet, wobei (z.B. bei Zahlen) D1 den niedrigsten und D10 den höchsten Stellenwert besitzt. D11 dient als Kennzeichenstelle und gibt an, ob die Dezimalen D10 ... D1 als Zahlen-, Befehls- oder Textwort von der Maschine gedeutet werden sollen.

Zur Verschlüsselung der einzelnen Dezimalstellen wird der sogenannte 3-Exzesscode verwendet. Dieser Code heißt 3-Exzesscode, weil die binäre Wertigkeit einer Bitgruppe den Wert der zugeordneten Dezimalziffer um 3 übertrifft. Die Zuordnung ist:

Dezimalziffer-	Bitgruppe	bin.Wert der Bitgruppe
0	00LL	3
1	0L00	4
2	0L0L	5
3	0LL0	6
4	0LLL	7
5	L000	8
6	L00L	9
7	L0L0	10
8	L0LL	11
9	LL00	12

Sechs der 16 möglichen 4-Bit-Gruppen, nämlich

0000
000L
00L0
LL0L
LLL0
LLLL

sind ungültige Ziffern (sogenannte Pseudotetraden); sie werden zur Kontrolle verwendet.

1.1.1 Darstellung von Zahlen

Jedes Wort, das in der Kennzeichenstelle eine 0 oder eine 9 aufweist, wird von der Anlage als Zahl gedeutet, und zwar gekennzeichnet

0 in D11 eine positive Zahl und

9 in D11 eine negative Zahl,

die in der Maschine durch ihr 10er-Komplement dargestellt wird; d.h. zum Beispiel 1723835- als 99998276165.

1.1.2 Darstellung von Text

Jedes Wort, das in der Kennzeichenstelle eine 4 oder eine 5 hat, wird von der Rechenanlage als Text gedeutet.

Ist das Kennzeichen 5, so müssen die Dezimalen D10 ... D1 gültige Ziffern enthalten. Jeweils zwei Ziffern stellen dann ein Textzeichen dar; ein Wort kann also 5 Textzeichen aufnehmen. Mit dem so dargestellten Text können arithmetische Operationen durchgeführt werden, da jedes Textwort eine gültige Zahl darstellt. Es können z.B. bei Vergleichsoperationen Additionen und Subtraktionen durchgeführt werden, wodurch sich erhebliche Vorteile ergeben.

Ein weiterer Vorzug des ziffernverschlüsselten Textes besteht darin, daß jedes Textwort wie eine Zahl auf die Gültigkeit seiner einzelnen Ziffern kontrolliert wird.

Der ZUSE-Code (siehe Anhang I) ist ziffernverschlüsselt; der Lochstreifencode entspricht bis auf ein Quersummenbit und die Umschlüsselung der Zeichen auf Großumschaltung der internen Verschlüsselung, daher ist bei der Lochstreifen-Ein- und -Ausgabe keine komplizierte Umschlüsselung nötig.

Ist das Kennzeichen eines Textwortes 4, so können in den 40 zu den Dezimalstellen D10 ... D1 gehörigen Binärstellen Informationen in einem beliebigen Code vorkommen; in diesem Fall sind also auch ungültige Ziffern (Pseudotetraden) erlaubt.

1.1.3 Darstellung von Befehlen

Jedes Wort, das in der Kennzeichenstelle D11 einer der Ziffern 1, 2, 3, 6, 7 oder 8 aufweist, wird von der Rechenanlage als Befehl gedeutet. Die Stellen D10 bis D1 müssen gültige Ziffern enthalten. Ein Befehlswort hat folgende Struktur:

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1
Befehls-kennzeichen	Bedingungs-zeichen		Operations-zeichen				Adresse			

1.2 Aufbau der Zentraleinheit ZUSE Z 31

Die wesentlichen Bauteile der Zentraleinheit ZUSE Z 31 sind:

Leitwerk, Operationswerk, Bedienungspult.
Speicher, Rechenregister,

Soweit es für die Programmierung notwendig erscheint, sollen die Bauteile in ihrer Wirkungsweise kurz beschrieben werden.

1.2.1 Leitwerk

Das Leitwerk besteht aus drei Registern. Von diesen sorgt das erste für die Aufnahme eines Befehls (Befehlsregister), das zweite für die Entschlüsselung der im Befehl enthaltenen Adresse (Steuerregister), das dritte für den Aufruf des nächsten Befehls (Befehlszählregister). Das Leitwerk wählt gemäß den im Adressenteil der Befehle angegebenen Adressen die betreffenden Speicherzellen an und öffnet bzw. schließt gemäß dem Operationsteil der Befehle entsprechende Wege in der Rechenanlage.

1.2.2 Speicher

Zur Aufnahme von Informationen dienen Speicher, die bei der ZUSE Z 31 in 11-stellige Speicherzellen (10 Dezimalen und eine Kennzeichenstelle) unterteilt sind. Jeder Speicherzelle ist eine Adresse zugeordnet. Der Inhalt einer Speicherzelle wird als "Wort" definiert. Die ZUSE Z 31 verfügt über drei Speichertypen.

1.2.2.1 Programmspeicher (fest verdrahteter Ferritkernspeicher)

Der Programmspeicher dient zur Aufnahme von Programmen und Daten, die ständig verfügbar sein müssen, wie z.B. die sogenannten Grundprogramme, d.h. Programme für die Ein- und Ausgabe sowie für die arithmetischen Operationen; ferner spezielle Programme, die vom Einsatz der Rechenanlage abhängen.

Dieser Speicher besteht aus zwei steckbaren Einheiten mit je 1300 Worten, die gegen andere derartige Einheiten beliebig ausgetauscht werden können. Die Informationen des Programmspeichers sind fest verdrahtet, d.h. sie können in einen beliebigen Programmablauf gerufen, aber nicht durch Speicherbefehle verändert werden. Jede einzelne Programmspeicherzelle ist adressierbar; die Adressen sind

0 bis 1299 in der ersten,
1300 bis 2599 in der zweiten steckbaren Einheit.

1.2.2.2 Schnellspeicher (Ferritkernspeicher)

Der Schnellspeicher ist der eigentliche Arbeitsspeicher der ZUSE Z 31. Er hat eine Kapazität von 200 bis zu 9000 Worten. Jedes Wort belegt eine Schnellspeicherzelle. Jede Schnellspeicherzelle ist einzeln adressierbar, die Adressen sind

1000 bis 9999.

Die Eigenschaft der Adressierbarkeit haben auch einige Register. Sie lassen sich unter Sonderadressen (Adressen kleiner als 1000) wie Schnellspeicherzellen anwählen und neu belegen oder ins Operationswerk rugen. Die für die Programmierung wichtigsten Register haben folgende Sonderadressen:

Rechenregister X	Sonderadresse	4
Rechenregister Y	"	6
Zählregister 1	"	7
Zählregister 2	"	8
Rückkehradressenspeicher	"	5

Die beiden Zählregister sind einstellig, der Rückkehradressenspeicher ist 5-stellig und die beiden Rechenregister sind 12-stellig.

Die Sonderadresse 0 liefert beim Aufruf dezimale Nullen.

1.2.2.3 Großraumspeicher

Hierzu zählen die Magnettrommeln (je 6400 Worte) und Magnetbänder (je 750 000 Worte), die dem ZUSE Z 31-System angegliedert werden können. Großraumspeicher dienen als Nachschubspeicher für große Mengen von Daten und Befehlen. Diese sind in Blöcke variabler Länge eingeteilt, jeder Block ist durch eine Adresse anwählbar. Zur Verarbeitung müssen Informationen vom Großraumspeicher zunächst auf den Schnellspeicher übertragen werden.

Dieses geschieht durch "Blocktransfer" zwischen Schnellspeicher und angewähltem Großraumspeicher.

1.2.3 Operationswerk

Seine Hauptbausteine sind das Addierwerk, das in Zusammenarbeit mit einem Komplementwerk positive und negative Zahlen addieren kann, ferner das logische Operationswerk, in dem sich die logischen Verknüpfungen, wie Intersektion und Kongruenzvergleich, vollziehen, und ein $\pm$ 1 Addierwerk zum Zählen in Schnellspeicherzellen und in den Rechenregistern.

1.2.4 Rechenregister

Das Operationswerk ist mit den beiden Rechenregistern X und Y sowohl direkt als auch über ein Verschiebewerk verbunden. Dadurch können die Inhalte der Rechenregister einzeln oder gekoppelt links- oder rechtsverschoben werden, bevor sie ins Operationswerk gelangen.

Beide Rechenregister sind 12-stellig; die zwölfte Dezimalstelle erhält beim Belegen eines Rechenregisters eine Kopie der Vorzeichenstelle D11. Sie soll in Rechenvorgängen bei eventuellem Überlauf auf die Stelle 11 das Vorzeichen erhalten.

1.2.5 Bedienungspult

Durch Tasten und Schalter am Bedienungspult kann der Programmablauf in der ZUSE Z 31 kontrolliert und beeinflußt werden. Ein große Anzahl von Leuchtanzeigen gibt Aufschluß über den Stand von Speichern der Anlage sowie über den Zustand von Kontrollstellen. Die Kontrollampen sind in ein Blockschaltbild der ZUSE Z 31 eingefügt.

Die für die Programmierung wichtigsten Bedienungselemente sind:

die Adressentastatur,
die Steuertasten P-Programm,
E-Programm,
Start,
Stop,
Weiter,
Adressenstop,

Tasten zum Setzen, Löschen und Blockieren von Bedingungsspeichern,

ein Zusatzbedienungspult mit
Zehnertastatur und den Tasten
Übernahme,
Irrung,
Löschen,

sowie eine 11-stellige optische Ziffernanzeigevorrichtung, die den Inhalt des Befehlsregisters anzeigt (Befehlsregisteranzeige).

Ein Feld mit Anzeigelampen läßt erkennen, in welchem Zustand sich die Rechenanlage befindet; es enthält die Anzeigen

Programmablauf
Stop vor Befehlsausführung
Stop nach Befehlsausführung
Adressenstop
Bedingungsspeicher blockiert und
Wartezeit.

1.2.5.1 Adressentastatur

Die Adressentastatur wird stets in Zusammenarbeit mit anderen Tasten des Bedienungspultes wirksam, und zwar:

a) mit den Tasten E-Programm bzw. P-Programm und Start bzw. Weiter zum Starten eines linearen Programmablaufs (siehe 2.2):

b) mit der rastenden Taste Adressenstop.

Ist die Taste Adressenstop eingerastet, so wird ein ablaufendes Programm unterbrochen sowie ein Befehl ins Befehlsregister gelangt, dessen Adressenteil (die untersten 4 Dezimalen) mit der auf der Adressentastatur eingestellten Adresse übereinstimmt. Die Maschine stoppt mit den Anzeigen Stop vor Befehlsausführung und Adressenstop.

Der Adressenstop ist ein wertvolles Hilfsmittel bei der Programmprüfung.

c) mit der Übernahme-Taste des Hilfsbedienungspultes.

Ist eine Schnellspeicheradresse eingetastet, so werden bei Betätigung der Übernahmetaste die Inhalte des Befehlsregisters und der angewählten Speicherzelle vertauscht, d.h. man kann vom Bedienungspult aus den Inhalt des Befehlsregisters in jede beliebige Schnellspeicherzelle und umgekehrt den Inhalt jeder Schnellspeicherzelle ins Befehlsregister übertragen und damit zur Anzeige bringen.

Ist eine Registeradresse (Sonderadresse) eingetastet, so kann wegen der unterschiedlichen Stellenzahlen der einzelnen Register nicht ausgetauscht werden.
Hier wird
bei der ersten Betätigung der Übernahmetaste der Inhalt des Befehlsregisters in ein besonderes Register, das Konstantenregister, übertragen; der Inhalt der letzten elf Stellen des von der Tastatur angewählten Registers wird ins Befehlsregister gebracht und angezeigt. Gleichzeitig wird das Register regeneriert, d.h. sein Inhalt bleibt erhalten.

Bei der zweiten Betätigung der Übernahmetaste wird der Inhalt des Konstantenregisters mit dem Befehlsregister vertauscht (d.h. das Befehlsregister erhält wieder seinen ursprünglichen Inhalt), wenn nicht zwischen der ersten und zweiten Übernahme eine der Tasten Start, Stop, Weiter, Zehnertastatur, Irrung oder Löschen gedrückt wurde. In diesem Fall ist die Wirkung wie bei der ersten Betätigung.

1.2.5.2 Zusatzbedienungspult

Mit Hilfe der Zehnertastatur können bei gestoppter Maschine Zahlen direkt ins Befehlsregister eingegeben werden.

Die Betätigung einer Zifferntaste bewirkt, daß der Inhalt des Befehlsregisters um eine Stelle nach links verschoben wird; in der frei gewordenen Stelle erscheint die eingetastete Ziffer. Das Befehlsregister kann so ziffernweise mit jeder beliebigen 11-stelligen Zahl aufgefüllt werden; die Befehlsregisteranzeige gestattet eine sofortige optische Kontrolle der eingetasteten Ziffern.

Ein Druck auf die Irrungstaste hat eine Rechtsverschiebung des Befehlsregisterinhaltes zur Folge, d.h. die zuletzt eingetastete Ziffer wird wieder herausgeschoben und kann korrigiert werden.

Die Löschtaste setzt alle Stellen des Befehlsregisters auf Null.

Die Funktion der Übernahmetaste ist in 1.2.5.1 beschrieben.

Zusatzbedienungspult und Adressentastatur gestatten es, vom Bedienungspult aus jedes beliebige ziffernverschlüsseltes Zahl-, Befehls- oder Textwort in jede gewünschte Speicherzelle zu bringen.

1.2.5.3 Weitere Bedienungselemente

Durch Niederdrücken der Stop-Taste wird ein ablaufendes Programm unterbrochen; die Anzeige Stop vor Befehlsausführung (d.h. vor Ausführung des angezeigten Befehls) leuchtet auf.

Nach Betätigung der Weiter-Taste wird der im Befehlsregister stehende (angezeigte) Befehl ausgeführt und der nächste Befehl ins Befehlsregister geholt. Dann stoppt die Anlage wieder mit der Anzeige Stop vor Befehlsausführung. So wird ein schrittweises Arbeiten möglich, das für Programm- und Maschinenprüfungen nützlich ist. Mit Hilfe von Adressentastatur und Übernahmetaste kann sich der Programmierer nach jedem Programmschritt den Inhalt von Registern und Speicherzellen ansehen und so überprüfen, ob die bislang ausgeführten Programmschritte die erwartete Wirkung hatten.

Die Start-Taste dient dazu, einen linearen Programmablauf (siehe 2.2) mit dem im Befehlsregister stehenden Befehl beginnen zu lassen. Nach Betätigung der Start-Taste leuchtet die Anzeige Programmablauf, wenn nicht etwa vorhandene Alarme einen Programmstart verhindern.

Die Funktion von Tasten und Anzeigen, die mit den Bedingungsspeichern im Zusammenhang stehen, wird in 2.8.6 beschrieben.

Zum An- und Abschalten der Maschine von der Stromversorgung dienen die in der obenstehenden Zusammenstellung nicht erwähnten Tasten Maschine ein, Maschine aus und Nacht.

Beim Einschalten der Maschine werden alle Register mit Ausnahme des Befehlszählregisters gelöscht.(d.h. mit Null belegt); der Inhalt des Befehlszählregisters nach dem Einschalten ist der Befehl E1111 (s. 2.2).

Ist die Taste Nacht eingerastet, so trennt sich die Anlage automatisch vom Netz, sowie ein Alarm autritt oder der Programmablauf durch einen Stop (Stopbefehl, Adressenstop, Bedingungsspeicher blockiert s.u.) unterbrochen wird. Die Nachttaste gestattet es, längere Programme ohne Aufsicht rechnen zu lassen.

2. Befehle der Zentraleinheit

2.1 Symbolik zur Erläuterung der Befehle

Zur Erläuterung der Befehle und ihrer Wirkungsweise dienen die folgenden Symbole:

X	1. Rechenregister (X-Register)
Y	2. Rechenregister (Y-Register)
b	Befehlsregister
c	Befehlszählregister
m	maximal vierstellige Zahl
n	" " "
sn	Schnellspeicherzelle mit der Adresse n
pn	Programmspeicherzelle mit der Adresse n
RAS	Rückkehradressenspeicher
ZR1	1. Zählregister
ZR2	2. Zählregister
D_i	Dezimale i eines Wortes
$\ldots_i$	Dezimale i von ...
$\ldots_{i-j}$	Dezimalen i bis j von ...
$\langle \ldots \rangle$	Inhalt von ...
... → ...	"nach" : Die links vom Pfeil angegebene Information wird nach der rechts angegebenen Stelle transportiert.
... ⇒ ...	"ergibt": Der links vom Doppelpfeil stehende Ausdruck ergibt den rechtsstehenden.
≠	"ungleich"
>	"größer"
<	"kleiner"
≅	Kongruenzvergleich
∧	Intersektion
␣	Zwischenraum (blank)

2.2 Linearer Programmablauf, Sprungbefehle, Stopbefehl

Grundsymbol	Zusatzsymbole	Adresse
E	F	n
P	F	n
Z0		

Ein linearer Programmablauf liegt dann vor, wenn die einzelnen Befehle einer Befehlsfolge, die in (Schnell- oder Programm-) Speicherzellen mit aufeinanderfolgenden Adressen gespeichert ist, in der durch die Speicherplatzadressen gegebenen Reihenfolge ins Befehlsregister gelangen und ausgeführt werden.

Der Programmablauf in der ZUSE Z 31 ist linear:
außerdem bestehen Möglichkeiten, einen linearen Programmablauf zu stoppen, zu starten oder zu unterbrechen.

Den Stop eines linearen Programmablauf bewirkt der Befehl

Z0 Wirkung: Die Maschine stoppt; am Bedienungspult leuchtet die Anzeige "Stop vor Befehlsausführung" auf.

Durch Niederdrücken der Starttaste wird der Programmablauf fortgesetzt, und zwar wird

falls Z0 mit einem Sprungbefehl kombiniert war, dieser ausgeführt;

andernfalls wird der Programmablauf, in der auf den Speicherplatz von Z0 folgenden Speicherzelle beginnend, linear fortgesetzt.

Die Maschine stoppt ebenfalls, wenn (im allgemeinen durch einen Programmierungsfehler) eine Zahl oder ein Textwort statt eines Befehls ins Befehlsregister gelangt.

Gelangt im Zuge eines linearen Programmablaufs ein Sprungbefehl ins Befehlsregister, so wird dieser Ablauf unterbrochen und ein neuer linearer Programmablauf eingeleitet. Es gibt zwei verschiedene Sprungbefehle:

En Wirkung: $\langle sn \rangle \rightarrow b$
Der lineare Programmablauf wird unterbrochen und ein neuer linearer Programmablauf eingeleitet, der mit dem in der Schnellspeicherzelle sn stehenden Befehl beginnt.

Pn Wirkung: ⟨pn⟩ → b

Der lineare Programmablauf wird unterbrochen und ein neuer linearer Programmablauf eingeleitet, der mit dem in der Programmspeicherzelle pn stehenden Befehl beginnt.

Vom Bedienungspult aus kann ein linearer Programmablauf gestartet werden. Dazu ist folgendes zu unternehmen:

1. Mit der Adressentastatur wird die Anfangsadresse n des gewünschten Programmes eingetastet, sodann
2. die Taste E-Programm (wenn sich das Programm im Schnellspeicher befindet) oder die Taste P-Programm (wenn das Programm im Programmspeicher steht) niedergedrückt und schließlich
3. die Starttaste betätigt.

Mit Schritt 2 wird der Sprungbefehl En bzw. Pn ins Befehlsregister gebracht, Schritt 3 veranlaßt die Ausführung dieses Befehls, d.h. ein linearer Programmablauf, beginnend in sn bzw. pn, wird eingeleitet.

Zur Durchführung eines linearen Programmablaufs muß die Maschine feststellen können, aus welcher Speicherzelle der Befehl gekommen ist, der im Befehlsregister steht, damit sie nach seiner Ausführung den Befehl aus der folgenden Speicherzelle, den "Folgebefehl" ins Befehlsregister rufen kann.

Diesem Zweck dient das Befehlszählregister c . In Form eines Sprungbefehls En bzw. Pn ist dort die Angabe gespeichert, daß der im Befehlsregister b stehende Befehl aus sn bzw. pn dorthin gelangt ist. Während der Ausführung dieses Befehls wird dann (ausgenommen bei Sprung-, Wiederholung- und Stopbefehlen) in c um 1 vorgezählt, d.h. der Sprungbefehl zum Folgebefehl gelangt nach c. Mit Hilfe des neuen Inhaltes von c wird dann der Folgebefehl nach b gerufen.

Beim Stopbefehl Z0 wird erst nach Betätigung der Starttaste in c vorgezählt; die Sprungbefehle veranlassen eine Neubelegung von c entsprechend den von ihnen eingeleiteten linearen Programmabläufen.

Wird in einem Programm ein Unterprogramm (durch einen Sprungbefehl) aufgerufen, so soll nach der Durchführung des Unterprogramms das übergeordnete Programm in der Regel mit dem Befehl fortgesetzt werden, der auf den Sprungbefehl in das Unterprogramm folgt. Dazu muß das Unterprogramm feststellen können, wo dieser "Folgebefehl" zu finden ist.

Das kann auf einfache Weise ermöglicht werden: dem Sprungbefehl En bzw. Pn in das gewünschte Unterprogramm wird das Symbol F hinzugefügt. Dadurch wird die Wirkung der Sprungbefehle folgendermaßen erweitert:

FEn Wirkung: ⟨c⟩ + 1 → RAS, ⟨sn⟩ → b

1. Der Sprungbefehl zum Folgebefehl von FEn gelangt in den Rückkehradressenspeicher RAS. (D.h. wenn FEn in sm stand, der Befehl Em+1, wenn FEn in pm stand, der Befehl Pm+1).

2. Der lineare Programmablauf wird unterbrochen und, mit dem in sn stehenden Befehl beginnend, ein neuer linearer Programmablauf eingeleitet

FPn Wirkung: ⟨c⟩ + 1 → RAS, ⟨pn⟩ → b

1. Der Sprungbefehl zum Folgebefehl von FPn gelangt in den Rückkehradressenspeicher.
2. Der lineare Programmablauf wird unterbrochen und, mit dem in pn gespeicherten Befehl beginnend, ein linearer Programmablauf eingeleitet.

Der Rückkehradressenspeicher hat die Adresse 5.
Steht dann am Ende eines mit FEn oder FPn aufgerufenen Unterprogrammes der Befehl

E 5,

so wird durch ihn der Inhalt des Rückkehradressenspeichers ins Befehlsregister gerufen. Der Rückkehradressenspeicher enthält wiederum einen Sprungbefehl, der dann den Folgebefehl des Befehls FEn bzw.FPn ins Befehlsregister ruft, d.h. der durch Aufruf des Unterprogramms unterbrochene lineare Ablauf des Hauptprogramms wird fortgesetzt.

2.3 Transportbefehle

Der Transport von Informationen (Zahlen, Befehlen und Text) innerhalb der Maschine von Speicherzelle zu Rechenregister und umgekehrt wird durch die Transportbefehle veranlaßt.

Grundsymbol	Zusatzsymbole	Adresse
B T	FP, Y, C, CY Y	n n

Dabei kommt die Richtung des Transportes bereits in der Symbolbezeichnung der Befehle zum Ausdruck:

B: Bringen von Speicherzelle nach Register und

T: Transportieren von Register nach Speicherzelle.

Bn Wirkung: ⟨sn⟩ → X, ⟨sn⟩ → sn

1. Der Inhalt von sn wird ins X-Register gebracht: der bisherige Inhalt von X wird gelöscht.
2. Der Inhalt von sn bleibt unverändert.

Der Programmspeicher der ZUSE Z 31 wird stets mit dem Sprungbefehl Pn angesprochen, auch wenn - wie bei den in 2.3 bis 2.6 beschriebenen Operationen - der Inhalt einer Programmspeicherzelle ins X-Register oder ins Operationswerk gerufen werden soll. In diesen Fällen verhindert das Zusatzsymbol F, daß nach Durchführung der jeweiligen Operation ein linearer Programmablauf aus dem Programmspeicher begonnen wird.

Es ist zu beachten, daß vor der Ausführung eines jeden Befehls mit dem Zusatzsymbol F der Rückkehradressenspeicher RAS mit dem Sprungbefehl auf die Folgebefehlsadresse (⟨c⟩ + 1) belegt wird.

BFPn Wirkung: ⟨c⟩ + 1 → RAS, ⟨pn⟩ → X
⟨pn⟩ konstant.

1. Der Sprungbefehl zum Folgebefehl von BFPn gelangt in den Rückkehradressenspeicher
2. Der Inhalt von pn wird ins X-Register gebracht; der bisherige Inhalt von X geht verloren.
3. Der Inhalt von pn bleibt konstant.

BYn Wirkung: ⟨sn⟩ → Y, ⟨sn⟩ → sn

1. Der Inhalt von sn wird ins Y-Register gebracht; der bisherige Inhalt von Y wird gelöscht.
2. Der Inhalt von sn bleibt erhalten.

Zwei Befehle ermöglichen es, die Rechenregister mit maximal vierstelligen Konstanten zu belegen:

BCn Wirkung: n → X

Die Zahl n wird ins X-Register gebracht; der vorherige Inhalt von X geht verloren.

BCYn Wirkung: n → Y

Einen Informationstransport in entgegengesetzter Richtung bewirken die Befehle:

Tn Wirkung: ⟨X⟩ → sn, ⟨X⟩ → X

1. Der Inhalt des X-Registers wird nach sn gebracht; der bisherige Inhalt von sn wird gelöscht.
2. Der Inhalt der X-Registers bleibt erhalten.

TYn Wirkung: ⟨Y⟩ → sn, ⟨Y⟩ → Y

2.4 Verschiebebefehle

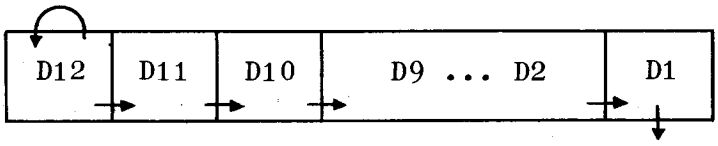
Grundsymbol	Zusatzsymbole	Adresse
R L	Y, R Y, L	

Die Verschiebebefehle haben die Aufgabe, entweder den Inhalt eines Rechenregisters oder die Inhalte von beiden Rechenregistern gekoppelt links- oder rechtszuverschieben. Gekennzeichnet werden sie mit R (Rechtsverschiebung) und L (Linksverschiebung), wenn sie auf das 1. Rechenregister, und durch Hinzufügen von Y (RY, LY), wenn sie auf das 2. Rechenregister wirken sollen; für gekoppelte Verschiebungen werden die Operationszeichen R bzw. L verdoppelt zu RR bzw. LL.

Verschoben wird jeweils um eine Dezimalstelle, das bedeutet praktisch eine Multiplikation mit 10^{+1} bzw. 10^{-1} Anhand der dezimalen Wortaufteilung sollen die Verschiebebefehle im einzelnen erläutert werden.

Wirkung: $\langle X \rangle \cdot 10^{-1} \rightarrow X$

Rechtsverschiebung im X-Register



D12 bleibt erhalten,

D1 geht verloren.

Analog:

RY Wirkung: $\langle Y \rangle \cdot 10^{-1} \rightarrow Y$

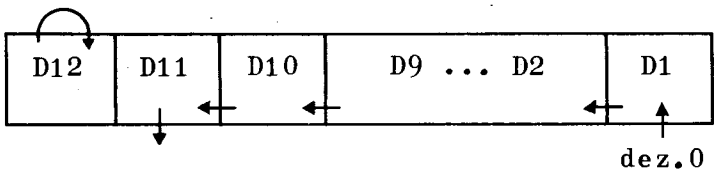
Rechtsverschiebung im Y-Register

Beispiel: $\langle X \rangle$ = 000987654321

nach R $\langle X \rangle$ = 000098765432

L Wirkung: $\langle X \rangle \cdot 10 \rightarrow X$

Linksverschiebung im X-Register.



D12 bleibt erhalten,

D11 geht verloren,

dez. 0 wird nach D1 geschoben.

Wirkung: $\langle Y \rangle \cdot 10 \rightarrow Y$

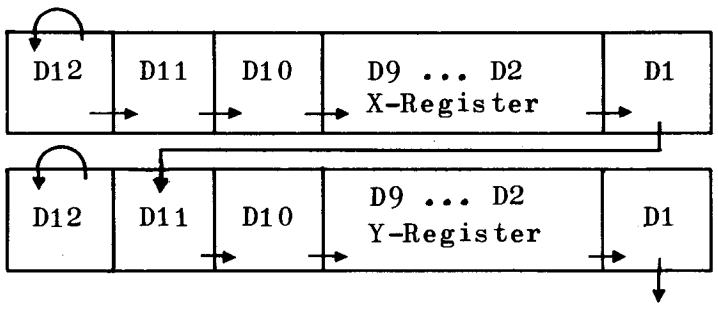
Linksverschiebung im Y-Register

Beispiel: $\langle Y \rangle$ = 000987654321

nach LY $\langle Y \rangle$ = 009876543210

RR Wirkung: $\langle X,Y \rangle \cdot 10^{-1} \rightarrow X,Y$

Gekoppelte Rechtsverschiebung im X- und Y-Register.

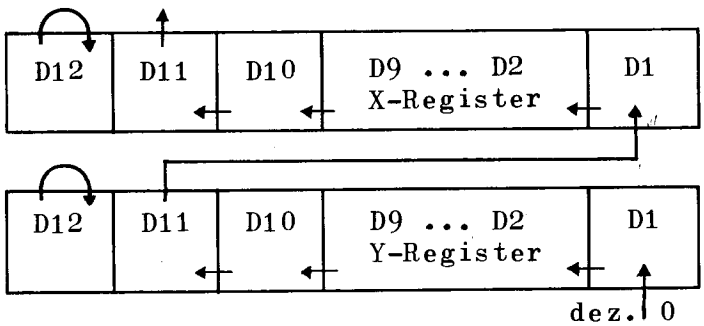


D12 bleibt in X und Y erhalten,
D1 von X gelangt nach D11 von Y,
D1 von Y geht verloren.

Beispiel: ⟨X⟩ = 000987609876
⟨Y⟩ = 995432154321
nach RR ⟨X⟩ = 000098760987
⟨Y⟩ = 969543215432

LL Wirkung:

⟨X, Y⟩ · 10 → X, Y
Gekoppelte Linksverschiebung
im X- und Y-Register.



D12 bleibt in X und Y erhalten,
D11 von X geht verloren,
D11 von Y gelangt nach D1 von X,
dez. 0 nach D1 von Y geschoben.

Beispiel: ⟨X⟩ = 000987609876
⟨Y⟩ = 995432154321
nach LL ⟨X⟩ = 009876098769
⟨Y⟩ = 954321543210

2.5 Arithmetische Befehle

Das Rechenwerk der Zentraleinheit ZUSE Z 31 führt die arithmetischen Operationen Addition und Subtraktion durch; die höheren Operationen Multiplikation und Division werden durch Unterprogramme ausgeführt, die als Teil der Grundprogramme der ZUSE Z 31 im 2. Teil des Programmspeichers, dem sogenannten Standardprogrammspeicher, fest verdrahtet sind.

Daher wird zwischen arithmetischen Befehlen mit und ohne Aufruf eines Unterprogrammes unterschieden.

2.5.1 Arithmetische Befehle ohne Aufruf eines Unterprogrammes

Grundsymbol	Zusatzsymbole	Adresse
A	FP, T, Y, TY, C, CY	n
S	FP, T, Y, TY, C, CY	n

Für die Additions- und Subtraktionsbefehle werden die Befehlssymbole A und S verwendet.

Bei jedem Additions- bzw. Subtraktionsbefehl werden die beiden Operanden als ganze Zahlen betrachtet. Dennoch können auch Dezimalzahlen addiert werden; Voraussetzung ist nur, daß die Lage des Dezimalpunktes bei beiden Operanden übereinstimmt. Der Dezimalpunkt wird nicht wirklich in der Zahl gesetzt; der Programmierer behandelt Dezimalzahlen nur so, als stünde zwischen zwei bestimmten Dezimalen ein Dezimalpunkt. Die Lage des Dezimalpunktes merkt er sich bei jeder Zahl als

Skalenfaktor q (z).

Der Skalenfaktor q (z) ist eine Zahl, die angibt, vor welcher Stelle des Maschinenwortes, das die Zahl z darstellt, der Dezimalpunkt angenommen wird.

Beispiel: Die Zahl 7256.43 wird in der Maschine als

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1
0	0	0	0	0	7	2	5	6	4	3

gespeichert; im Programm muß jedoch berücksichtigt werden, daß diese Zahl den Skalenfaktor q = 2 hat.

Voraussetzung für die Anwendbarkeit der Additions- und Subtraktionsbefehle ist, daß beide Operanden den gleichen Skalenfaktor haben.

Bei Operanden mit verschiedenen Skalenfaktoren müssen diese erst durch Verschieben eines oder beider Operanden in Übereinstimmung gebracht werden.

An Wirkung: ⟨X⟩ + ⟨sn⟩ → X, ⟨sn⟩ → sn

1. Zum Inhalt des X-Registers wird der Inhalt von sn addiert und die Summe ins X-Register gebracht.
2. Der Inhalt von sn bleibt unverändert

AFPn Wirkung: ⟨c⟩ + 1 → RAS, ⟨X⟩ + ⟨pn⟩ → X, ⟨pn⟩ konstant

1. Der um 1 erhöhte Inhalt des Befehlszählregister c wird im Rückkehradressenspeicher RAS notiert.
2. Zum Inhalt von X wird der Inhalt von pn addiert und die Summe ins X-Register gebracht.
3. Der Inhalt von pn bleibt konstant.

ATn Wirkung: ⟨sn⟩ + ⟨X⟩ → sn, ⟨X⟩ → X

1. Zum Inhalt von sn wird der Inhalt des X-Registers addiert und die Summe nach sn transportiert.
2. Der Inhalt des X-Registers bleibt unverändert.

Die analogen Befehle für das zweite Rechenregister sind:

AYn Wirkung: ⟨Y⟩ + ⟨sn⟩ → Y, ⟨sn⟩ → sn

ATYn Wirkung: ⟨sn⟩ + ⟨Y⟩ → sn, ⟨Y⟩ → Y

Die Addition von (maximal vierstelligen) Konstanten erlauben die folgenden Befehle:

ACn Wirkung: ⟨X⟩ +n → X
Zum Inhalt des X-Registers wird die Zahl n addiert und das Ergebnis im X-Register bereitgestellt.

ACYn Wirkung: ⟨Y⟩ +n → Y

Für die Subtraktion gibt es die Befehle:

Sn Wirkung: ⟨X⟩ - ⟨sn⟩ → X, ⟨sn⟩ → sn

1. Vom Inhalt des X-Registers wird der Inhalt von sn subtrahiert und die Differenz nach X gebracht.
2. Der Inhalt von sn bleibt unverändert

SFPn Wirkung: ⟨c⟩ +1 → RAS, ⟨X⟩-⟨pn⟩ → X, ⟨pn⟩ konstant

1. Der um 1 erhöhte Inhalt des Befehlszählregister c wird im Rückkehradressenspeicher RAS notiert.
2. Vom Inhalt des X-Registers wird der Inhalt von pn subtrahiert und das Ergebnis nach X gebracht.
3. Der Inhalt von pn bleibt konstant.

STn Wirkung: ⟨sn⟩ - ⟨X⟩ → sn, ⟨X⟩ → X

1. Vom Inhalt von sn wird der Inhalt von X subtrahiert und das Ergebnis nach sn transportiert.
2. Der Inhalt von X bleibt erhalten.

Die analogen Befehle für das Y-Register lauten:

SYn Wirkung: ⟨Y⟩ - ⟨sn⟩ → Y, ⟨sn⟩ → sn

STYn Wirkung: ⟨sn⟩ - ⟨Y⟩ → sn, ⟨Y⟩ → Y

Die Subtraktion von (maximal 4-stelligen) Konstanten gestatten die folgenden Befehle:

SCn Wirkung: ⟨X⟩ -n → X
Vom Inhalt des X-Registers wird die Zahl n subtrahiert und das Ergebnis nach X gebracht.

SCYn Wirkung: ⟨Y⟩ -n → Y

2.5.2 Arithmetische Befehle mit Aufruf eines Unterprogrammes

Grundsymbol	Zusatzsymbole	Adresse
*	R	
/	/, R, /R	

Als Operationszeichen für Multiplikation und Division werden die in Formelsprachen üblichen mathematischen Zeichen * und / verwendet.

Vom Leseprogramm (siehe Abschnitt 3) werden diese Zeichen in Rufbefehle (FP-Befehle) für die entsprechenden verdrahteten Unterprogramme übersetzt. Daher ist nach

Ausführung eines Multiplikations- bzw. Divisionsbefehls der bisherige Inhalt des Rückkehradressenspeichers zerstört.

Die Speicher 1010 und 1017 werden von den arithmetischen Unterprogrammen als Hilfsspeicher benutzt, ihr Inhalt darf für das Hauptprogramm, das arithmetische Befehle verwendet, keine Bedeutung haben.

Die arithmetischen Befehle sind im einzelnen:

* Wirkung: ⟨c⟩ +1 → RAS; ⟨X⟩ * ⟨Y⟩ → X, Y
Die Inhalte der beiden Rechenregister werden als 10-stellige ganze Zahlen betrachtet und miteinander multipliziert; das 20-stellige Ergebnis, also eine Zahl von doppelter Wortlänge, wird in den Rechenregistern bereitgestellt und zwar enthält

das X-Register die oberen 10 Stellen und
das Y-Register die unteren 10 Stellen

des Resultates.

Die Inhalte beider Register tragen das Vorzeichen des Resultates, ein negatives Resultat wird durch sein 10-er Komplement dargestellt.

Beispiel: 1. Operand ⟨X⟩ = 1723830-
2. Operand ⟨Y⟩ = 42021

Das Resultat 72437060430- steht dann in den Rechenregistern als
⟨X⟩ = 999999999992 obere 10 Stellen
⟨Y⟩ = 997562939570 untere 10 Stellen

Werden die Operanden vom Programm als Dezimalzahlen betrachtet, so haben ihre Skalenfaktoren (siehe 2.5.1) Einfluß auf den Skalenfaktor des Resultates.

Es gilt die Gleichung:

q (Produkt) = q (Multiplikand) + q (Multiplikator)

Wird im angeführten Beispiel

der 1. Operand als 172.3830- (q=4) und
der 2. Operand als 42.021 (q=3) gedeutet, so ist das
Resultat als 7243.7060430- (q=7) aufzufassen.

Zur Erläuterung des nächsten Befehls muß ein wenig auf den Abschnitt 2.7.1 vorgegriffen werden. Bei der Rundung hat der Inhalt des schon in 1.2.2.2 erwähnten einstelligen Zählregisters 2 (ZR2) eine Bedeutung. Dieses Zählregister kann durch den Befehl

I2n

mit der Einerstelle der Adressen belegt werden.

* R Wirkung: ⟨c⟩ +1 → RAS, $\frac{\langle X\rangle * \langle Y\rangle}{\langle ZR2\rangle}$ gerundet → X und Y
10

⟨ZR2⟩ bleibt erhalten.

Der Befehl zur Multiplikation mit Rundung wird in zwei Schritten ausgeführt:

1. Schritt: ⟨c⟩ +1 → RAS, ⟨X⟩*⟨Y⟩ → X, Y
D.h. zunächst werden die beiden Operanden mit dem Befehl * multipliziert. Das Ergebnis wird dann im

2. Schritt: $\frac{\langle X, Y\rangle}{\langle ZR2\rangle}$ gerundet → X und Y
10

durch die gewünschte Zehnerpotenz geteilt (Rechtsverschiebung) und die letzte Stelle wird gerundet.

Bei der Multiplikation mit Rundung gilt für den Skalenfaktor des Resultates

q (Produkt) = q (Multiplikand) + q (Multiplikator) - ⟨ZR2⟩

Beispiel: Die Operanden seien wie oben 1723830- und 42021, nur werde diesmal durch den Befehl

*R mit ⟨ZR2⟩ = 3

Multipliaktion mit Rundung verlangt. Das Ergebnis ist

⟨X⟩ = ⟨Y⟩ = 72437060-

Nimmt man wiederum q (Multiplikand) = 4 und q (Multiplikator) = 3 an, so ist q (Produkt) = 4, d.h. das Ergebnis ist als

7243.7060-

zu deuten.

In der Praxis sind häufig Zahlen d aus drei anderen Zahlen a, b und c nach der Formel

$$d = \frac{a * b}{c}$$

zu berechnen. Sind die drei Ausgangswerte 10-stellige Zahlen, so ist es die Genauigkeit des Ergebnisses am größten, wenn man zunächst n = a * b berechnet und das 20-stellige Ergebnis dieser Multiplikation durch c dividiert. Dieses Vorgehen erlaubt der Befehl

// Wirkung: ⟨c⟩ +1 → RAS, ⟨1011,Y⟩ / ⟨X⟩ → Y
Divisionsrest → X

Die Inhalte von 1011 und Y werden als eine ganze Zahl von doppelter Länge aufgefaßt; 1011 enthält die oberen, das Y-Register die unteren 10 Stellen. Es wird der Absolutbetrag des Dividenden durch den Absolutbetrag des Divisors geteilt; das Resultat ist negativ, wenn die beiden Operanden verschiedenes Vorzeichen haben; der Rest bleibt positiv stehen.

Das Resultat dieser Division darf höchstens 10 Stellen haben; ist der Divisor Null oder so klein, daß die

Stellenzahl des Resultates größer als 10 würde, so stoppt die Anlage mit dem Befehl

Z0E1050

im Befehlsregister, nachdem die Protokollschreibmaschine mit

BUEB 1n oder
BUEB 2n

die Ursache der Programmunterbrechung (BEREICHS-UEBERSCHREITUNG) und die Adresse n derjenigen Schnell-(1) oder Programm-(2) -speicherzelle bekanntgegeben hat, in der der Divisionsbefehl steht.

Den Skalenfaktor des Quotienten berechnet man bei der Division von Dezimalzahlen nach der Formel

q (Quotient) = q (Dividend) - q (Divisor)

Beispiel:	Dividend 14173602013840	⟨1011⟩	=	00000001417
		⟨ Y ⟩	=	003602013840
	Divisor 52143-	⟨ X ⟩	=	999999947857
	Resultat 271821759-		=	⟨Y⟩
	Rest 34303		=	⟨X⟩

Die Berechnung der Formel $d = \frac{a * b}{c}$ würde man wie folgt programmieren:

Es sei	⟨1150⟩ = a	
	⟨1151⟩ = b	
	⟨1152⟩ = c	
B1150		a → X
BY1151		b → Y
*		a * b → X, Y
T1011		obere Resultatstellen 1011
B1152		c → X
//		⟨1011,Y⟩ / ⟨X⟩ → Y

Das Resultat steht im Y-Register zur weiteren Verwendung bereit.

Neben der Division mit 10-stelligen Dividenden ist auch eine Division möglich, bei der beide Operanden nur einfache Länge besitzen:

/ Wirkung: ⟨c⟩ +1 → RAS, 0 → 1011,
⟨1011,Y⟩ / ⟨X⟩ → Y
Divisionsrest → X

Das für diese Division zuständige Unterprogramm belegt zunächst 1011 mit Null (d.h. die oberen 10 Stellen eines 20-stelligen Dividenden werden zu Null gemacht) und geht dann zur Division // über.

BUEB-Stop kann hier nur dann auftreten, wenn der Divisor Null ist.

Der Skalenfaktor des Quotienten wird wie bei der Division // nach der Formel

q (Quotient) = q (Dividend) - q (Divisor)

berechnet.

Ein charakteristisches Anwendungsbeispiel für die Division mit Rundung, die - analog zur Multiplikation mit Rundung - ebenfalls zu den Grundprogrammen der ZUSE Z 31 gehört, ist die Berechnung von Prozent- bzw. Promillesätzen. Dabei sind zwei Zahlen mit gleicher Dezimalpunktlage (gleichem Skalenfaktor) so zu dividieren, daß das Resultat auf zwei bzw. drei Dezimal hinter dem Dezimalpunkt genau ist. Die normale Division / oder // liefert jedoch als Quotient zweier Zahlen mit gleichem Skalenfaktor eine Zahl mit dem Skalenfaktor 0.

Die Division mit Rundung gestattet es, den Skalenfaktor des Resultates zu vergrößern. Beim Aufruf einer Division mit Rundung ist im Zählregister 2 anzugeben, um wieviel der Skalenfaktor zu vergrößern ist.

//R Wirkung: ⟨c⟩ +1 → RAS
⟨1011,Y⟩ · $10^{\langle ZR2 \rangle}$ / ⟨X⟩ → X und Y, gerundet
⟨ZR2⟩ bleibt erhalten

Bei der Division //R wird zunächst der Dividend mit $10^{\langle ZR2 \rangle}$ multipliziert (durch Linksverschieben), anschließend wird die Division // ausgeführt. Aus dem Divisionsrest wird dann noch eine Resultatstelle ermittelt, mit deren Hilfe das Resultat gerundet wird. Das gerundete Resultat wird in beiden Rechenregistern bereitgestellt.

/R Wirkung: $\langle c \rangle + 1 \rightarrow RAS$, $0 \rightarrow 1011$
$\langle 1011, Y \rangle \cdot 10^{\langle ZR2 \rangle} / \langle X \rangle \rightarrow X$ und Y, gerundet
$\langle ZR2 \rangle$ bleibt erhalten.

Das Programm zur Division mit Rundung bei einfach langem Dividenden belegt zunächst 1011 (d.h. die oberen Dividendenstellen) mit Null und geht dann zur Division //R über.

BUEB-Stop tritt bei der Division mit Rundung dann ein, wenn bei der aufgerufenen Division // das Resultat mehr als 10 Stellen haben würde, weil der Divisor zu klein oder Null ist. Für den Skalenfaktor des Quotienten gilt bei der Division mit Rundung die Beziehung

q (Quotient) = q (Dividend) - q (Divisor) + $\langle$ ZR2 $\rangle$

Beispiel: $\langle X \rangle$ = 3572890 wird als 35728.90 und
$\langle Y \rangle$ = 135265 als 1352.65 gedeudet.

Nach Ausführung des Befehls /R bei $\langle ZR2 \rangle = 3$ enthalten die beiden Rechenregister die Zahl

38

Sie kann dahingehend gedeutet werden, daß 1352.65 38 % von 35728.90 ausmacht.

2.6 Logische Befehle

Grundsymbol	Zusatzsymbole	Adresse
U	FP,Y,C,CY	n
K	FP,Y,C,CY	n

Die ZUSE Z 31 ist in der Lage, verschiedene logische Operationen auszuführen. In den Externcode der ZUSE Z 31 wurden Befehle für die Intersektion und den Kongruenzvergleich aufgenommen.

2.6.1 Intersektion

Intersektion ist die logische Operation der Konjunktion ("Und"-Operation), sie wird mit dem Befehlssymbol U gekennzeichnet.

Bei der Intersektion werden die Inhalte eines Rechenregisters und einer Schnellspeicherzelle nach folgendem Schema verknüpft:

Stelle im Register (beliebiger Inhalt)	entsprechende Dezimalstelle in sn	Resultatstelle im Register
Z	0	0
Z	≠ 0	Z

Aufgabe der Intersektion ist es, aus dem X-Register oder Y-Register bestimmte Teile eines Wortes (Zahl oder Text) herauszuschneiden. Das im Register stehende Ergebnis hat an allen denjenigen Dezimalstellen Nullen, wo auch in der angesprochenen Schnellspeicherzelle Nullen stehen; auf den übrige Dezimalen bleibt der vorherige Inhalt des Registers stehen.

Beispiel: Im X-Register sollen folgende Daten eines Lohnempfängers stehen:

Zahl der Kinder auf den beiden letzten Stellen, Geburtsdatum (Tag, Monat, Jahr) auf den restlichen acht Stellen. Durch Intersektion soll das Geburtsjahr herausgeschnitten werden.

⟨ X ⟩ = 000511192302
⟨sn ⟩ = 00000ZZZZ00 (Z = 1, ..9)

Resultat ⟨ X ⟩ = 000000192300

Zur symbolischen Darstellung der Befehlswirkung wird das logistische Zeichen für "und" verwendet. Die Intersektionsbefehle lauten im einzelnen:

Un Wirkung: $\langle X \rangle \wedge \langle sn \rangle \rightarrow X$, $\langle sn \rangle \rightarrow sn$

Will man den Inhalt einer Programmspeicherzelle zur Intersektion heranziehen, fügt man dem Intersektionsbefehl Un das Symbol FP hinzu.

UFPn Wirkung: $\langle c \rangle + 1 \rightarrow RAS$, $\langle X \rangle \wedge \langle pn \rangle \rightarrow X$
$\langle pn \rangle$ konstant

Die Intersektion der Inhalte einer Schnellspeicherzelle und des Y-Registers bewirkt

UYn Wirkung: $\langle Y \rangle \wedge \langle sn \rangle \rightarrow Y$, $\langle sn \rangle \rightarrow sn$

Intersektion mit (maximal vierstelligen) Konstanten.

UCn Wirkung: $\langle X \rangle \wedge n \rightarrow X$

Der Inhalt von X wird mit der Zahl n, die durch unwesentliche Nullen auf die volle Wortlänge gebracht wird, interseziert und das Ergebnis im X-Register bereitgestellt.

Analog für das Y-Register

UCYn Wirkung: $\langle Y \rangle \wedge n \rightarrow Y$

2.6.2 Kongruenzvergleich

Der Inhalt eines Rechenregisters kann mit dem Inhalt einer beliebigen Speicherzelle verglichen werden.
Als Ergebnis des Kongruenzvergleichs werden in der Maschine einer oder zwei der folgenden Indikatoren gesetzt oder gelöscht:

1. NG: "Nicht gleich", wenn $\langle sn \rangle \neq \langle X \rangle$ bzw. $\langle sn \rangle \neq \langle Y \rangle$
2. GR: "Größer" , wenn $\langle sn \rangle > \langle X \rangle$ bzw. $\langle sn \rangle > \langle Y \rangle$
3. KL: "Kleiner" , wenn $\langle sn \rangle < \langle X \rangle$ bzw. $\langle sn \rangle < \langle Y \rangle$

Diese Indikatoren können mit dem Bedingungsteil eines Befehls abgefragt werden (siehe auch unter Bedingte Befehle). Befehle, die auf einen Kongruenzvergleich folgen, können unter diese Bedingungen gestellt und davon ihre Ausführung anhängig gemacht werden.

Sind die verglichenen Worte gleich, so ist keine der drei Bedingungen erfüllt.

Die Inhalte des zum Vergleich herangezogenen Rechenregisters und der Speicherzelle bleiben unverändert.

Im einzelnen sind folgende Kongruenzbefehle möglich:

Kn Wirkung: ⟨sn⟩ $\cong$ ⟨X,⟩ , ⟨sn⟩ → sn, ⟨X⟩ → X

KFPn Wirkung: ⟨c⟩ +1 → RAS
⟨pn⟩ $\cong$ ⟨X,⟩ , ⟨pn⟩ konstant, ⟨X⟩ → X

KYn Wirkung: ⟨sn⟩ $\cong$ ⟨Y⟩ , ⟨Y⟩ → Y, ⟨sn⟩ → sn
und die Vergleichsoperationen mit Konstanten

KCn Wirkung: ⟨n⟩ $\cong$ X, ⟨X⟩ → X

KCYn Wirkung: ⟨n⟩ $\cong$ Y, ⟨Y⟩ → Y

Einige Beispiele sollen die Durchführung des Kongruenzvergleiches verdeutlichen.

1.

	D11	D10	D9	D8	.....
⟨n⟩	00LL**)	0L0L	LL00	L0L0	beliebige Stellen
⟨X⟩	00LL**)	0L0L	LL00	0LLL	

ergibt GR, weil ⟨n⟩ $_8>$ ⟨X⟩ $_8$ ist und die Dezimalen D10 bis D9 gleich sind.

2.

	D11	D10	D9	D8	.....
⟨n⟩	00LL**)	L0L0	LL00	L00L	beliebige Stellen
⟨X⟩	LL00***)	0LL0	0LL0	LL00	

ergibt GR, weil die Zahl in n positiv, die im X-Register negativ ist.

3.

	D11	D10	D9	D8	.....
⟨n⟩	LL00***)	0LL0	0LLL	LL00	beliebige Stellen
⟨X⟩	LL00***)	L00L	L0L0	0L0L	

ergibt KL, weil im X-Register die dem Betrage nach kleinere negative Zahl steht.

**) Null im Dreiexcesscode

***) Neun im Dreiexcesscode,

Kennzeichen der negativen Zahl, die in der Maschine durch ihr 10er-Komplement dargestellt wird.

2.7 Zählbefehle

Grundsymbol	Zusatzsymbole	Adresse
V	1,2,X,Y	
MVX		n
Z	1,2,X,Y	
MZX		n
I	1,2	n

Die Zählbefehle bewirken ein Vor- bzw. Zurückzählen um eine Einheit und werden entsprechend mit V oder Z gekennzeichnet. Soweit es sich um ein Zählen in den beiden Zähl- oder Rechenregistern handelt, sind an V bzw. Z die Symbole 1, 2, X oder Y hinzuzufügen.

Außerdem kann noch in jeder Schnellspeicherzelle gezählt werden, die mit der Adresse nach den Symbolen MVX bzw. MZX näher bestimmt wird. Zu beachten ist, daß die Zählregister, da sie nur 1 Dezimalstelle fassen, zyklisch von 0 bis 9 zählen.

V1 Wirkung: ⟨ZR1⟩ +1 → ZR1
Vorzählen im Zählregister 1

Z1 Wirkung: ⟨ZR1⟩ -1 → ZR1
Zurückzählen im Zählregister 1

V2 Wirkung: ⟨ZR2⟩ +1 → ZR2
Vorzählen im Zählregister 2

Z2 Wirkung: ⟨ZR2⟩ -1 → ZR2
Zurückzählen im Zählregister 2

VX Wirkung: ⟨X⟩ +1 → X
Vorzählen im X-Register

ZX Wirkung: ⟨X⟩ -1 → X
Zurückzählen im X-Register

VY Wirkung: ⟨Y⟩ +1 → Y
Vorzählen im Y-Register

ZY Wirkung: ⟨Y⟩ -1 → Y
Zurückzählen im Y-Register

MVXn Wirkung: ⟨sn⟩ +1 → sn
Vorzählen in der Schnellspeicherzelle sn

MZXn Wirkung: ⟨sn⟩ -1 → sn
Zurückzählen in der Schnellspeicherzelle sn

Die Zählbefehle spielen in zyklischen Programmen eine wesentliche Rolle. Man verändert durch Vor- und Zurückzählen laufend den betreffenden Zählerstand und verwendet diesen z.B. als Bedingung dafür, ob ein Sprung in einen anderen Programmteil erfolgen soll oder nicht.

2.7 Belegung der Zählregister

Die Zählregister ZR1 und ZR2 können durch entsprechende Befehle mit Zahlen von 0 bis 9 belegt werden.

I1n Wirkung: n_1 → ZR1
Das Zählregister 1 wird mit der letzten Dezimalstelle von n belegt.

I2n Wirkung: n_1 → ZR2
Das Zählregister 2 wird mit der letzten Dezimalstelle von n belegt.

2.8 Bedingte Befehle

Erst wenn man die Möglichkeit hat, die Ausführung eines Befehls von dem Erfülltsein einer Bedingung abhängig zu machen, kann man ein nichtlineares Programm aufstellen. Der Interncode der ZUSE Z 31 kennt 20 verschiedene Bedingungen, von denen jede einzeln oder, nach gewissen Regeln (siehe 2.11) kombiniert, bis zu drei Bedingungen jedem Befehl vorangestellt werden können.

Diese Vielfalt von Entscheidungsmöglichkeiten erlaubt ein sehr elegantes Programmieren.

Ein bedingter Befehl wird nur dann ausgeführt, wenn alle Bedingungen, unter die er gestellt ist, erfüllt sind; sonst wird der Befehl überlaufen.

Kennzeichen der bedingten Befehle sind die Bedingungssymbole, die dem Befehl in runden Klammern vorangestellt werden.

Gegenstand dieses Abschnittes ist es, die einzelnen Bedingungszeichen zu erläutern.

Grundsymbol	Zusatzsymbole			Adresse
		(J1)	(NE)	
		(J2)	(PO)	
	(W)	(J3)	(NU)	
		(N4)	(SU)	
		(N5)	(NG)	
		(N6)	(GR)	
	(Y0)	(YZ)	(KL)	
	(10)	(1Z)	(Y5)	
		(20)	(2Z)	

2.8.1 Bedingungen, die sich nur auf das X-Register beziehen

(NE)...: Wenn $\langle X \rangle < 0$ (NEgativ), dann ...
d.h. wenn der Inhalt des X-Registers negativ ist, dann führe den Befehl ... aus.

(PO)...: Wenn $\langle X \rangle \geq 0$ (POsitiv) ist, dann ...
(0 gilt als positive Zahl)

(NU)...: Wenn $\langle X \rangle = 0$ (NUll) ist, dann ...

2.8.2 Bedingungen, die sich nur auf das Y-Register beziehen

(Y0)...: Wenn $\langle Y \rangle_1 = 0$, dann ...
d.h. wenn die letzte Dezimale D1 im Y-Register gleich Null ist, dann führe den Befehl ... aus.

(YZ)...: Wenn $\langle Y \rangle_1 \neq 0$, dann ...
d.h. wenn die letzte Dezimale D1 im Y-Register eine Ziffer ungleich Null ist, dann führe den Befehl ... aus.

(Y5)...: Wenn $\langle Y \rangle_1 \geq 5$, dann ...

2.8.3 Eine Bedingung, die sich auf beide Rechenregister bezieht

(SU)...: Wenn sgn ⟨X⟩ ≠ sgn ⟨Y⟩ , dann ...
d.h. wenn das Vorzeichen ("Signum") vom Inhalt des X-Registers "Ungleich" ist dem Vorzeichen des Inhaltes vom Y-Register, dann wird der Befehl ... ausgeführt.

2.8.4 Bedingungen, die sich auf den letzten Kongruenzvergleich beziehen

(NG)...: Wenn der letzte Kongruenzvergleich "Nicht Gleich" ergab, dann ...

(GR)...: Wenn der letzte Kongruenzvergleich "GRößer" ergab, dann ...

(KL)...: Wenn der letzte Kongruenzvergleich "KLeiner" ergab, dann ...

(siehe auch 2.6 Logische Befehle)

2.8.5 Bedingungen, die sich auf den Stand eines Zählregisters beziehen

(10)...: Wenn ⟨ZR1⟩ = 0, dann ...

(1Z)...: Wenn ⟨ZR1⟩ ≠ 0, dann ...

(20)...: Wenn ⟨ZR2⟩ = 0, dann ...

(2Z)...: Wenn ⟨ZR2⟩ ≠ 0, dann ...

2.8.6 Abfrage der Bedingungsspeicher B1 ... B6

Die Zentraleinheit ZUSE Z 31 verfügt über 6 sogenannte Bedingungsspeicher. Jeder von ihnen vermag eine Binärstelle, d.h. einen Ja-Nein-Wert zu speichern. Im Verlauf eines Programms oder mit den entsprechenden Tasten am Bedienungspult kann man jeden dieser Bedingungsspeicher in seine "Ja"- oder "Nein"- Stellung setzen.

Ist jedoch ein Bedingungsspeicher mittels der betreffenden Taste am Bedienungspult blockiert, so bewirkt jeder Versuch,

den Inhalt dieses Bedingungsspeichers vom Programm her zu ändern, einen Stop der Anlage vor Ausführung des Änderungsbefehls mit den Anzeigen "Stop vor Befehlsausführung" und "Bedingungsspeicher blockiert".

Grundsymbol	Zusatzsymbole	Adresse
J N	1,2,3,4,5,6 1,2,3,4,5,6	

Die Befehle sind im einzelnen:

J1 . . J6 Wirkung: Der betreffende Bedingungsspeicher B1 ... B6 wird in Ja-Stellung gebracht

N1 . . N6 Wirkung: Der betreffende Bedingungsspeicher B1 ... B6 wird in Nein-Stellung gebracht.

Zur Abfrage der Stellung der einzelnen Bedingungsspeicher dienen die folgenden Bedingungszeichen:

(J1) ... B1
(J2)...: Wenn der Bedingungsspeicher B2 in Ja-Stellung
(J3)...: B3
ist, das führe den Befehl ... aus.

(N4)... B4
(N5)...: Wenn der Bedingungsspeicher B5 in Neinstellung
(N6)... B6
ist, dann ...

Wie alle anderen Befehle der ZUSE Z 31 können auch die Befehle zum Setzen der Bedingungsspeicher bedingt gegeben werden; über gewisse Einschränkungen, die in der Liste der Befehlskombinationen angegeben sind, setzt man sich mit Hilfe eines kleines Kunstgriffes hinweg. Dies soll ein Beispiel erläutern.

In einem Programm soll, falls der Bedingungsspeicher 2 in Ja-Stellung steht, der Bedingungsspeicher 3 in Nein-Stellung gebracht werden; steht B2 in Nein-, so soll B3 die Ja-Stellung erhalten.

Man programmiert:

```
3050    N3
3051    (J2) E3053
3052    J3
3053    ..... (nächster Befehl des Programms)
```

Zunächst wird B3 in Nein-Stellung gesetzt. Wenn die Speicherzelle 3051 zum Aufruf kommt, so erfolgt die Befehlsausführung. d.h. der Programmsprung auf 3053 nur dann, wenn der Bedingungsspeicher 2 in Ja-Stellung steht. Damit würde automatisch der Befehl 3052 übergangen werden. Steht jedoch B2 in Nein-Stellung, so wird der Befehl in 3051 überlaufen und 3052 und folgende werden ausgeführt, d.h. der Bedingungsspeicher B3 wird in Ja-Stellung gesetzt.

2.8.7 Sonderfunktion des Bedingungsspeichers B6

Der Bedingungsspeicher B6 gibt Aufschluß über den Zahlenbereich einer Zahl im X-Register. Eine Zahl liegt im (Zahlen-) Bereich, wenn $\langle X_{11} \rangle = \langle X_{12} \rangle$ ist. Der Bereich wird überschritten, sobald die Zahl im X-Register 11-stellig, d.h. $\langle X_{11} \rangle \neq \langle X_{12} \rangle$ geworden ist, z.B. nach einer Rechenoperati oder Linksverschiebung im X-Register. Sobald der Bereich überschritten ist, wird nun automatisch der Bedingungsspeicher B6 in Ja-Stellung gebracht; die abfragbare Bedingung (N6) ist dann nicht mehr erfüllt.

<u>Beispiel:</u> $\langle X \rangle = 007482653084$

$\langle 1927 \rangle = 06219337180$

Nach dem Befehl

A 1927

ist $\langle X \rangle = 013701990264$

und B6 steht in Ja-Stellung, da durch Überlauf bei der Addition $\langle X_{12} \rangle \neq \langle X_{11} \rangle$ geworden ist.

2.9 Sonder-Transportbefehle

Die Bedingungsspeicher und die Zählregister werden in fast allen Programmen zur Steuerung des Programmablaufs eingesetzt, auch in den meisten Unterprogrammen, die von beliebigen Programmen aufgerufen werden können. Der für den Rücksprung ins Hauptprogramm wichtige Inhalt des Rückkehradressenspeichers wird in Unterprogrammen häufig verändert; z.B. durch den Aufruf weiterer Unterprogramme oder das Operieren mit Konstanten aus dem Programmspeicher.

Durch die Anwendung zweier Sonderbefehle innerhalb eines Unterprogrammes kann der Programmierer auf einfache Weise erreichen, daß die Stellung der Bedingungsspeicher, der Zählregister und des Rückkehradressenspeichers nach dem Rücksprung aus dem Unterprogramm ins Hauptprogramm (der durch den Befehl E5 bewirkt wird) die gleiche ist wie beim Sprung (Befehl EFn oder PFn) in dieses Unterprogramm. Diese Befehle lauten:

TFn Wirkung: ⟨RAS, ZR1, ZR2, B1,.....,B6⟩ → sn
Die Inhalte von Rückkehradressenspeicher, beiden Zählregistern und den 6 Bedingungsspeichern werden gemeinsam in die Schnellspeicherzelle n transportiert; die jeweiligen Inhalte von RAS, ZR1, ZR2, und B1,....,B6 bleiben erhalten.

BFn Wirkung: ⟨sn⟩ → RAS, ZR1, ZR2, B1,....,B6
Der Inhalt von sn wird auf den Rückkehradressenspeicher, die beiden Zählregister und die Bedingungsspeicher verteilt; der Inhalt von sn bleibt erhalten.

Bei gleichem n stellt dieser Befehl den Zustand wieder her, den die angesprochenen Register und Speicher (ausgenommen sn) hatten, als der Befehl TFn ausgeführt wurde.

Die in der ZUSE Z 31 fest verdrahteten Unterprogramme beginnen in der Regel mit einem Befehl

TFn

und enden mit

BFn
E5

2.10 Adressensubstitution und Adressenmodifikation

Grundsymbol	Zusatzsymbole	Adresse
...	G	n
...	Xi+,Xi#,Xi+G,Xi#G	n

2.10.1 Adressensubstitution

Setzt man das Symbol G vor die Adresse eines Befehles mit Adressenteil, so wird vor Ausführung der verlangten Operation ... erst eine neue Adresse gebildet:

...Gn Wirkung: $\langle sn \rangle_{4-1} \Rightarrow \tilde{n}$: neuer Adressenteil des Befehls ...

1.Schritt: Im Befehlsregister b wird der Adressenteil n des Befehls ... durch die letzten vier Dezimalstellen des Inhalts von sn ersetzt und das Symbol G im Befehl gelöscht.

2.Schritt: Der Befehl ... wird mit der neuen (substituierten) Adresse $\tilde{n}$ ausgeführt.

Beispiel: Der Inhalt der Schnellspeicherzelle 3010 sei $\langle s3010 \rangle$ = 03624567321

Der Befehl

BG3010

wird in zwei Schritten ausgeführt:

1. Die Adresse 3010 wird durch 7321 ersetzt und das Symbol G im Befehl gelöscht. Der Befehl lautet dann

   B7321

2. Auf Grund dieses Befehles wird der Inhalt von s7321 in das X-Register gebracht.

2.10.2 Adressenmodifikation

Wie bei der Adressensubstitution wird auch bei der Adressenmodifikation vor Ausführung der Operation eine neue Adresse aufgebaut. Dies geschieht durch Umrechnung

der Adresse in Zusammenwirkung mit einem der 10 zur Verfügung stehenden Indexregister s1000, ..., s1009, welches man gleichzeitig mit dem Modifikationsbefehl aufruft.

Dazu wird im Modifikationsbefehl mit i = 0,...,9 die letzte Dezimalstelle der Indexregisteradresse angegeben.

...Xi+n Wirkung: $(\langle s1000+i \rangle + n)_{4-1} \Rightarrow \tilde{n}$: neuer Adressenteil des Befehls ...

$\langle s1000+i \rangle \rightarrow s1000+i$

1.Schritt: Der Inhalt des aufgerufenen Indexregisters wird zu n addiert. Im Befehlsregister b wird der Adressenteil n durch die letzten vier Dezimalstellen der Summe ersetzt und das Symbol Xi+ im Befehl gelöscht.

2.Schritt: Der Befehl ... wird mit der neuen (modifizierten)Adresse $\tilde{n}$ ausgeführt.

Der Inhalt des Indexregisters bleibt unverändert.

...Xi#n Wirkung: $(\langle s1000+i \rangle + n)_{4-1} \Rightarrow \tilde{n}$

$\tilde{n} \rightarrow s1000+i_{4-1}$

1.Schritt: Wie bei Xi+n; außerdem gelangt die neue Adresse in die letzten vier Stelle des aufgerufenen Indexregisters. Die Dezimalen D11-D5 des Indexregisterinhaltes bleiben unverändert.

2.Schritt: Wie bei Xi+n.

<u>Beispiel:</u> Der Inhalt des Indexregisters s1004 sei

$\langle s1004 \rangle$ = 00000004220

Der Befehl

AX4+9999

wird in zwei Schritten ausgeführt:

1. Der Inhalt von s 1004 wird zu 9999 addiert. Die Adresse 9999 wird durch die letzten vier Dezimalstellen der Summe 14219 ersetzt und das Symbol X4+ im Befehl gelöscht. Der Befehl lautet dann

   A4219

2. Auf Grund dieses Befehls wird der Inhalt von s4219 zum Inhalt des X-Registers addiert und das Ergebnis nach X gebracht.

In einem Befehl können gleichzeitig Adressensubstitution und Adressenmodifikation verlangt werden. In diesem Falle hat die Adressensubstitution Vorrang.

Beispiel: Es sei

⟨s2034⟩ = 00000000026
⟨s1006⟩ = 00000003440

Der Befehl

TX6#G2034

wird in drei Schritten ausgeführt:

1. Die Adresse 2034 wird durch 0026 ersetzt und das Symbol G im Befehl gelöscht. Dieser lautet dann

   TX6#26.

2. Auf Grund dieses Befehls werden 26 und der Inhalt von s1006 addiert; die letzten vier Stellen des Ergebnisses werden nach $s1006_{4-1}$ gebracht und treten an die Stelle der Adresse 26; außerdem wird das Symbol X6 # im Befehl gelöscht; damit steht im Befehlsregister b der Befehl

   T3466

   und es ist

   s1006 = 00000003466

3. Der Befehl T3466 bewirkt sodann, daß der Inhalt des X-Registers nach s3466 gebracht wird.

2.11 Befehlskombinationen, Wiederholungsbefehle

Der analytische Aufbau des Interncode der ZUSE Z 31 ermöglicht es, auch im Externcode eine Reihe von Befehlen zu kombinieren, d.h. mehrere Befehle in einem Befehlswort zusammenzufassen. Kombinierte Externbefehle werden in einer Speicherzelle abgelegt und bis auf zwei Ausnahmen parallel ausgeführt:

I1 hat Vorrang vor V1 und Z1

Z0 hat Vorrang vor den Sprungbefehlen.

Es können auch mehrere Bedingungszeichen kombiniert werden. Hierbei spielt das bisher nicht erwähnte Bedingungszeichen

(W): Wiederholungsbefehl

eine besondere Rolle.

(W), mit einem oder zwei weiteren Bedingungszeichen gekoppelt vor einen Befehl oder eine Befehlskombination gesetzt, bewirkt, daß der Befehl bzw. die Befehlskombination so lange wiederholt ausgeführt wird, wie die Bedingungen erfüllt sind.
(W) muß mit mindestens einem Bedingungszeichen gekoppelt sein, und der Operationsteil der Befehlskombination muß eine Operation enthalten, die bewirkt, daß die gekoppelten Bedingungen nicht immer erfüllt bleiben. (Andernfalls würde sich der Wiederholungsbefehl "totlaufen", d.h. ständig ausgeführt werden.)

Beispiel: (W.N6)L

Im X-Register muß eine von Null verschiedene Zahl stehen. Diese wird dann so lange nach links verschoben, bis ein Übertrag auf die Vorzeichenstelle D11 gelangt und damit die Bedingung N6 nicht mehr erfüllt ist.

Bei Wiederholungsbefehlen mit Adressenmodifikation ist zu beachten, daß die neu gebildete Adresse des Operationsteiles im folgenden Wiederholungsschritt als Ausgangsadresse für die nächste Modifikation gilt. Damit ist in der Regel nur eine wiederholte Adressenmodifikation ohne Änderung des Indexregisterinhaltes sinnvoll.

Beispiel: (W.NE)AX5+1050

Der Inhalt des Indexregisters 1005 sei die "Schrittweite" 1.

Die Adressen der Schnellspeicherzellen, deren Inhalte zum Inhalt von X addiert werden, solange $\langle X \rangle < 0$ ist, sind dann 1051, 1052, 1053 usw.

Über die möglichen Kombinationen von Befehlen im Externcode der ZUSE Z 31 gibt die folgende Tabelle Aufschluß.

1 **)	2 *)	3	4	5	6	7 *)	8 **)	9
(W) (Y0) (10)	(J1) (J2) (J3) (N4) (N5) (N6) (YZ) (1Z) (20)	(NE) (P0) (NU) (SU) (NG) (GR) (KL) (Y5) (2Z)	B T A AT S ST U K	V1 Z1 Ji*) Ni*)	I1 I2 V2 Z2 VY ZY	Xi+ Xi#	G	n
			BC AC SC UC KC		I1 I2 V2 Z2 VY ZY	Xi+ Xi#	G	n
			BY TY AY ATY SY STY UY KY	V1 Z1 VX ZX Ji*) Ni*)		Xi+ Xi#	G	n
			E EF P PF	Z0*)		Xi+ Xi#	G	n

Es können nur Befehle aus verschiedenen Spalten kombiniert werden; waagerechte Striche schließen eine Kombination aus.

*) Die Befehle Xi+, Xi#, Z0, Ji und Ni dürfen weder miteinander noch mit Bedingungen aus Spalte 2 kombiniert werden. werden.

**)Bedingungen aus Spalte 1 und Adressensubstitution dürfen nicht in einem Befehl gleichzeitig auftreten.

Alle hier nicht aufgeführten Befehle können unter je eine Bedingung aus den Spalten 1 bis 3 gestellt werden; bei Befehlen mit Adressenteil sind außerdem Adressensubstitution und -modifikation zulässig; dabei sind *) und **) zu beachten.

Die Beschränkungen der Kombinierbarkeit haben ihre Ursache in der maschineninternen Darstellung der einzelnen Befehle. Die folgende Tabelle gibt an, welche Dezimalwerte in welchen Dezimalstellen die einzelnen Grundbefehlszeichen haben.

Beim Programmtest muß der Programmierer mitunter Befehle in intern-ziffernverschlüsselter Form lesen können; dabei hilft ihm diese Tabelle. Sie gibt ferner einen Ausblick auf die Vielzahl der Kombinationsmöglichkeiten des Intercode der ZUSE Z 31; ein Studium der Druckschrift "ZUSE Z 31 / Intercode" wird jedem Programmierer empfohlen.

Kennzeichenstelle

		Bedingung		Operation				Adresse			
	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1
0	+	-	-	-	-	-	-				
1	Befehl	(J1)	(NE)	A	R	V1	E				
2	G	(J2)	(P0)	S	L	Z1	P				
3	(W)	(J3)	(NU)	AT	RR	Ji*)	V2				
4	TE	(N4)	(SU)	ST	LL	Ni*)	Z2				
5	Text	(N5)	(NG)	U	LD	VX	I1				
6	VB	(N6)	(GR)	K	Xi+*)	ZX	I2				
7	(Y0)	(YZ)	(KL)	0	Xi#*)	C	VY				
8	(10)	(1Z)	(Y5)	B	Z0	Q	ZY				
9	-	(20)	(2Z)	T	M	F	Y				

*) i wird in D10 markiert.
Einige der aufgeführten Befehlszeichen sind dem Leser noch unbekannt; diese Symbole werden im Interncode der ZUSE Z 31 erläutert.

3. Lochstreifeneingabe

3.1 Lochstreifenherstellung/Lochstreifencode

Programme und Daten für die Verarbeitung in der ZUSE Z 31 werden auf einem besonderen Herstellgerät abgelocht, das unabhängig von der Rechenanlage arbeitet.

Die Hauptbestandteile dieses Gerätes sind

1. eine elektrische Schreibmaschine
2. ein Streifenlocher und
3. ein Lochstreifenleser.

Mit Hilfe dieses Gerätes können Lochstreifen erstellt, korrigiert, kopiert und ausgeschrieben werden. Als Lochstreifencode wird der "ZUSE-Code" verwendet, ein 8-Kanal-Code, der der internen Informationsdarstellung der ZUSE Z 31 angepaßt ist.

Anlage I gibt über die Symbole des ZUSE-Code Aufschluß. Neben den Ziffern, dem Alphabet (groß und klein) und den Funktionstasten (Zwischenraum ZW, Tabulatorsprung TAB, Wagenrücklauf WR, Groß-, Klein-, Schwarz- und Rotschaltung) enthält der ZUSE-Code noch 26 weitere Symbole.

Sie werden verwendet

als Satzzeichen	'-():"/,.;!?
bei kommerziellen Aufgaben	+-.# ◊*%&
in Formelsprachen	[ ] ( ) :='; +-*/.,→
im Externcode der ZUSE Z 31	
zur Zahlendarstellung	+-.
als Textkennzeichen	"
zur Befehlsdarstellung	( )+ # */.:
als Endezeichen	,;
als Steuerzeichen	⊤∈

Für die Benutzung des (Standard-) Leseprogramms der ZUSE Z 31 ist Voraussetzung, daß die einzulesenden Informationen im ZUSE-Code auf dem Lochstreifen stehen.

3.2 Das Leseprogramm der ZUSE Z 31

Ein Leseprogramm hat die Aufgabe, extern dargestellte Informationen (Befehle, Daten) vom jeweiligen Eingabemedium zu lesen, in den internen Code der Datenverarbeitungsanlage umzuschlüsseln und die Speicherung der internen Informationen zu veranlassen.

Diese Aufgabe läßt sich lösen, wenn folgende Voraussetzungen erfüllt sind:

1. Dem Leseprogramm sind Angaben über das zu benutzende Eingabegerät (falls mehrere Möglichkeiten der Informationseingabe vorgesehen sind) und über die Verschlüsselung der einzulesenden Informationen (falls mehrere Verschlüsselungen möglich sind) zugänglich.

2. Die externen Informationen sind auf dem jeweiligen Eingabemedium so dargestellt, daß die Maschine in der Lage ist, die externen Informationen zu identifizieren (d.h. z.B. für die ZUSE Z 31, daß Befehle, Zahlen und Text unterscheidbar sein müssen) und festzustellen, wann eine Information beendet ist.

3. Das Leseprogramm kann feststellen, wohin die eingelesenen Informationen zu speichern sind.

4. Das Ende des Einlesevorganges ist (abhängig oder unabhängig von den einzulesenden Informationen) festgelegt.

Das mit der ZUSE Z 31 gelieferte, im Programmspeicher verdrahtete Leseprogramm kann externe Informationen von Lochstreifen oder Lochkarten lesen und übersetzen, sowie dem Schnellspeicher zur Verarbeitung zuführen.

Im Folgenden wird beschrieben, wie die oben aufgeführten Voraussetzungen 1. bis 4. speziell für die Lochstreifeneingabe der ZUSE Z 31 geartet sind; das Lesen von Lochkarten wird im entsprechenden Abschnitt der Programmierungsanleitung behandelt.

3.3 Lochstreifenleser

An die Standardausführung der ZUSE Z 31 können zwei Lochstreifenleser angeschlossen werden; jeder von ihnen kann 300 Zeichen/s verarbeiten. Mit diesen Lesegeräten können auch Lochstreifen mit geringerer Kanalzahl als 8 (also z.B. Fernschreiblochstreifen) in die Rechenanlage eingelesen werden; ein Umbau der Leser ist dazu nicht erforderlich. Das Einlesen eines Lochstreifens, dessen Informationen nicht im ZUSE-Code verschlüsselt sind, setzt nur ein besonderes Leseprogramm (zur Umschlüsselung in den internen Code) voraus.

Die Anschlußelektronik der Lochstreifenleser führt automatisch die Umschlüsselung des Lochstreifencode, in dem noch beide Zeichen einer Taste des Herstellgerätes (z.B. a und A, 1 und [ ) durch die gleiche Lochkombination dargestellt sind, in den eindeutigen internen Code der Anlage durch; die Zeichen für Groß- und Kleinschaltung sind dadurch innerhalb der Maschine überflüssig und werden nicht eingelesen.

Durch den Umschlüsselungsvorgang entsteht jedoch kein Zeitverlust.

Dem Leseprogramm muß angegeben werden, mit welchem Gerät Lochstreifen gelesen werden sollen. Diese Angabe ist in der Speicherzelle 1033 zu machen. Es bedeutet

$\langle 1033 \rangle = 11$, daß Abtaster 1
$\langle 1033 \rangle = 13$, daß Abtaster 2

zum Einlesen benutzt werden soll.

3.4 Informationsverschlüsselung

Zur Eingabe in die ZUSE Z 31 über Lochstreifen müssen Zahlen, Befehle und Text nach bestimmten Regeln abgelocht sein; diese Regeln sind sehr allgemein gehalten, so daß dem Programmierer viel Freiheit in der externen Darstellung von Informationen bleibt.

Es werden vier Informationstypen unterschieden:

Befehle, Text und
Zahlen, TE-Konstanten

Auf dem Lochstreifen muß hinter jeder Information ein Informationsendezeichen stehen. Als Informationsendezeichen gelten:

;	bei allen vier Informationstypen
,	bei Befehlen, Zahlen und TE-Konstanten
ZW	
TAB	
WR	

3.4.1 Darstellung von Befehlen

Für die Verschlüsselung von Befehlen gelten vier Regeln:

1. Bei einem bedingten Befehl muß das bzw. müssen die Bedingungszeichen am Anfang des Befehls stehen und in runde Klammern gesetzt werden. Dabei können mehrere Bedingungszeichen in einer Klammer stehen.
2. Der Adresse eines Befehls dürfen nur noch Bandbefehlszeichen (siehe 3.5.1.1) folgen.
3. Jeder Befehl und jedes Bedingungszeichen muß zusammenhängend geschrieben werden; verschiedene Befehle einer Befehlskombination oder verschiedene, einem Befehl vorangestellte Bedingungen können durch Punkte voneinander getrennt werden, die Reihenfolge der Einzelbefehle einer Befehlskombination ist bedeutungslos. Zwischen Operationsteil und Adresse eines Befehls kann ebenfalls ein Punkt gesetzt werden.
4. Alphabetische Befehlszeichen können mit Groß- oder mit Kleinbuchstaben geschrieben werden; Farbumschaltungen innerhalb von Befehlen sind bedeutungslos.

Beispiel: (W.PO)S.ZY.1086

(jlne)i1.tgl728

v1.byx8 # 127

3.4.2 Darstellung von Zahlen

In Anlehnung an die internationalen Maschinensprachen ALGOL und COBOL wird zur Darstellung von Dezimalzahlen der Dezimalpunkt verwendet.

Es gelten folgende Regeln:

1. Neben den Ziffern von 0 bis 9 dürfen zur Darstellung von Zahlen nur die Zeichen + - und . verwendet werden.

2. Ein positives Vorzeichen kann, ein negatives Vorzeichen muß angegeben werden. Das Vorzeichen kann vor oder hinter der Zahl stehen.

3. Der Punkt kann auch zur Trennung von Zifferngruppen Verwendung finden. Farbumschaltungen innerhalb einer Zahl sind bedeutungslos.

Eine Kontrolle der Ziffernanzahl findet nicht statt; es können also Zahlen mit bis zu 11 Ziffern eingelesen werden (von längeren Zahlen werden nur die letzten 11 Stellen gespeichert). Durch diese Regelung wird es möglich, auch Befehle und Text intern-ziffernverschlüsselt in die Rechenanlage einzulesen. Von der Möglichkeit, ziffernverschlüsselten Interncode einzulesen, wird bei häufig benutzten Programmen Gebrauch gemacht. Nachdem das im Externcode erstellte Programm geprüft ist, läßt man es sich von der Maschine ziffernverschlüsselt ausgeben; wenn das Programm später wieder benutzt werden soll, kann es mit der Maximalgeschwindigkeit des Lesers von 300 Zeichen/s wieder eingelesen werden. (Ein Programm mit 1000 Befehlen hat man so in 40 Sekunden in der Maschine).

3.4.3 Darstellung von Text

Der Anfang eines Textes ist mit dem Zeichen " zu kennzeichnen. Im Text dürfen mit Ausnahme des Textendezeichens; alle Zeichen des ZUSE-Code vorkommen; das Textanfangszeichen " wird in der Maschine nicht gespeichert, wohl aber das Textendezeichen;. Es belegt wie jedes andere Textzeichen zwei Dezimalstellen.

Beispiel: "Externcode der ZUSE Z 31; wird gespeichert (beispielsweise nach s 1526 und den folgenden Schnellspeicherzellen) in der Form:

```
1526  Exter
1527  ncode
1528  ß der ß
1529  ZUSE ß
1530  Z ß 31;
```

3.4.4 Darstellung von TE-Konstanten

TE-Konstanten entsprechen denjenigen Maschinenwörtern, die in der Kennzeichenstelle D11 eine 4 aufweisen und in den restlichen Dezimalen Informationen in beliebigen Code, also auch Pseudotetraden, enthalten können.
Da man mitunter TE-Konstanten benötigt, die beliebige Tetraden enthalten können (z.B. Lochprogramm für Fernschreibcode), wurde eine Möglichkeit geschaffen, auch Pseudotetraden extern darzustellen und einzulesen.

Nach dem Kennzeichen TE werden vom Leseprogramm die ersten 6 Buchstaben des Alphabets umgedeutet und zwar

a	in	binär	0	0000
b	"	"	1	000L
c	"	"	2	00L0
d	"	"	13	LL0L
e	"	"	14	LLL0
f	"	"	15	LLLL

Binär 3 bis binär 12 entsprechen ja den Ziffern 0 bis 9 im Dreiexzesscode.

Auf das Kennzeichen TE (oder te) müssen die Zeichen für genau 10 Tetraden folgen, anschließend ist ein Endezeichen zu geben.

Beispiel: Eine TE-Konstante, die auf der Dezimalen Di (i = 1... 10) gerade die Binärzahl i enthalten soll, muß folgendermaßen abgelocht werden:

te76543210cb

3.5 Aufruf des Leseprogramms

Im Folgenden werden zwei Möglichkeiten zum Aufruf des Leseprogramms besprochen: der Sprungbefehl P2000,

der in erster Linie zum Start des Leseprogramms vom Bedienungspult aus benutzt wird, und die Lesebefehle des Externcode der ZUSE Z 31, die den Aufruf des Leseprogramms als Unterprogramm zu einem beliebigen Hauptprogramm bewirken.

In beiden Fällen ist Voraussetzung, daß

1. die Speicherzelle 1033 mit 11 oder 13 belegt ist, abhängig davon, ob mit Leser 1 oder Leser 2 gearbeitet werden soll und daß
2. die einzulesenden Informationen im ZUSE-Code und nach den in 3.4 beschriebenen Regeln verschlüsselt auf dem Lochstreifen stehen.

Sonderzeichen des ZUSE-Code, die zur Darstellung von Informationen im Externcode der ZUSE Z 31 nicht verwendet werden, lassen sich in zwei Gruppen einteilen.

Gruppe 1 enthält die beiden Zeichen ⊤ und ⊆ , die bei der Ausführung von Lesebefehlen als Endezeichen wirken (siehe unten).

Gruppe 2 enthält alle übrigen Sonderzeichen. Wird eins dieser Zeichen gelesen, so wird der Einlesevorgang unterbrochen und das Programm mit dem in einer bestimmten Schnellspeicherzelle stehenden Befehl fortgesetzt. Verschiedenen Sonderzeichen sind auch verschiedene Speicherzellen zugeordnet (siehe Anhang Indem er diese Zellen mit Sprungbefehlen belegt, kann der Programmierer vom Datenstreifen her den Programmablauf steuern: ein bestimmtes Sonderzeichen ruft ein bestimmtes Programm auf, das die soeben eingelesenen Daten verarbeitet.

3.5.1 Programmeinlesen mit dem Sprungbefehl P2000

Nachdem s1033 mit 11 oder 13 belegt ist, wird mit Hilfe von Adressentastatur und P-Programm-Taste der Sprungbefehl P2000 ins Befehlsregister gebracht. Ein Druck auf die Starttaste läßt das Leseprogramm anlaufen.-

Die Informationen von dem in den gewählten Leser eingelegten Lochstreifen werden gelesen, in die maschineninterne Darstellung übersetzt und fortlaufend gespeichert. Ein Zeichen τ oder ε auf dem Lochstreifen beendet den Einlesevorgang: die Anlage stoppt mit dem Befehl

Z0P2000

im Befehlsregister; nach Betätigung der Starttaste liest sie vom gleichen Leser weiter ein.

Beim Speichern dient das Indexregister 1009 als Zähler: wird nach einer Information ein Endezeichen gelesen, so wird diese mit dem Indexbefehl tx9 # 1 im Schnellspeicher abgelegt. D.h. beim Aufruf des Leseprogramms mit P2000 beginnt die Speicherung in $s\ \langle 1009\rangle_{4-1}+1$.

Das Indexregister 1009 kann auf einfache Weise vom einzulesenden Lochstreifen her belegt werden: mit Hilfe eines sogenannten Bandbefehls.

3.5.1.1 Bandbefehle

Im Gegensatz zu gewöhnlichen Befehlen, die vom Lochstreifen gelesen, in der Maschine gespeichert und erst nach dem Einlesen des gesamten Programms aufgerufen werden können, werden Bandbefehle sofort ausgeführt, nachdem sie gelesen und übersetzt worden ist.

Bandbefehle werden in erster Linie dazu benutzt, der Rechenanlage Anweisungen zu geben, die das Leseprogramm steuern. So gibt es z.B. einen Bandbefehl, der den Anfangsspeicherplatz der einzulesenden Informationen festlegt, einen anderen, der den Einlesevorgang beendet und der Maschine angibt, in welcher Speicherzelle der erste nunmehr auszuführende Befehl steht; zwei weitere Bandbefehle, die im Abschnitt 5 beschrieben werden, dienen dazu, die Adresse eines soeben eingelesenen Befehls umzurechnen.

Das Indexregister 1009, der Speicherzähler des Leseprogramms, wird belegt durch

TnT Wirkung: n-1 → 1009
Der Befehl gibt dem Leseprogramm die Anweisung, die folgenden Informationen vom Lochstreifen (soweit es sich nicht um Bandbefehle handelt) in die Schnellspeicherzellen sn, sn+1 usw. zu speichern.

Der Befehl TnT selbst wird nicht gespeichert.

Der Einlesevorgang wird unterbrochen durch den Bandbefehl

EnE Wirkung: En → b
Die Maschine beginnt einen linearen Programmablauf mit dem in sn stehenden Befehl,

oder durch den Bandbefehl

ZOEnE Wirkung: ZOEn → b, Stop vor Befehlsausführung
Die Maschine stoppt mit dem Befehl ZOEn im Befehlsregister; ein Druck auf die Starttaste veranlaßt, daß ein linearer Programmablauf mit dem in sn stehenden Befehl begonnen wird.

Der erste Befehl wird am Schluß eines Programms gelocht und veranlaßt die Maschine, sofort nach dem Einlesen eines Programms mit dem Programmablauf zu beginnen.

Der zweite gibt - analog zum ersten angewendet - den Programmierer Gelegenheit, zu überprüfen, ab alle Vorbereitungen zum Programmablauf getroffen sind (Bereitschaft der Anschlußgeräte, Setzen von Bedingungsspeichern usw.), bevor durch Niederdrücken der Starttaste der Programmablauf eingeleitet wird.

Am Anfang des Abschnitts 3.5 wurde schon erwähnt, daß das Leseprogramm nach der Entschlüsselung bestimmter Sonderzeichen (Anhang 4) auf bestimmte Speicherzellen springt. Unter Berücksichtigung des soeben Gesagten kann man diese Sonderzeichen auch als Bandbefehle EnE mit festen Adressen in abgekürzter Schreibweise auffassen.

Wie die Sprungbefehle En und ZOEn kann auch jeder andere Befehl des Externcode durch Anfügen des Symbols E als Bandbefehl gekennzeichnet werden; Voraussetzung ist nur,

daß er eine Adresse besitzt. (Ausnahme: arithmetische Befehle und Druckbefehle, die aber, da sie in FP-Befehle übersetzt werden, im Grunde doch eine Adresse haben).

...nE Wirkung: Der Befehl ...n wird sofort ausgeführt.

Dabei übernehmen die Schnellspeicherzellen 1034 und 1036 die Funktionen von X- und Y-Register, da die beiden Rechenregister beim Einlesen ständig benötigt werden. Die beiden Speicherzellen werden bei jedem Aufruf des Leseprogramms mit den derzeitigen Inhalten der Rechenregister belegt; in s1035 wird (mittels TF-Befehls) der Stand von Zählregistern und Bedingungsspeichern vor Aufruf des Leseprogramms festgehalten.

Die Forderung nach einer Adresse im Bandbefehl verbietet es nicht, auch solche Befehle als Bandbefehle ausführen zu lassen, die an sich keinen Adressenteil haben: der Befehl v1 beispielsweise wird als Bandbefehl

v1.0e

geschrieben. (Befehle ohne Adresse werden vom Leseprogramm ohnehin mit der Adresse Null versehen).

Bandbefehle, die auf Zählregister oder Bedingungsspeicher einwirken sollen, verändern das bei Aufruf des Leseprogramms in s1035 konservierte Niveau, auf das das Leseprogramm sonst keinen Einfluß hat.

Drei Beispiele sollen die unterschiedliche Wirkung von Bandbefehlen und gewöhnlichen Befehlen verdeutlichen:

Bandbefehle		gewöhnliche Befehle	
bc1065e	1065 → 1034	bc1065	1065 → X
ay1750e	⟨1036⟩ +⟨1750⟩ → 1036	ay1750	⟨Y⟩ + ⟨1750⟩ → Y
t2343e	⟨1034⟩ → 2343	t2343	⟨X⟩ → 2343

Mit Hilfe von Bandbefehlen kann auch der andere Abtaster in den Lesevorgang eingeschaltet werden. So erfolgt etwa die Umschal-

tung von Leser 1 auf Leser 2, nachdem die Anlage die beiden Bandbefehle

```
bc13e
t1033e
```

übersetzt und ausgeführt hat. (Neubelegung von 1033 mit der Adresse von Leser 2).

Den Einsatz von Bandbefehlen zur Speicherplatzdefinition zeigt das folgende Beispiel:

<u>Aufgabe:</u> Das Resultat der Formel $\frac{a * b}{c} + d$ soll für

$a = 316.10$
$b = 48$
$c = 7.58$
$d = 4.31$

ermittelt und gedruckt werden; das Programm und die Eingangswerte sind auf die Zellen 2050 ff einzulesen.

Nach dem Einlesen des folgenden Lochstreifen sind die Eingangswerte in den Speicherzellen 2050 bis 2053 gespeichert:

```
t2050t
316.10
48
7.58
4.31
T
```

Vor Aufruf des Leseprogramms mit P2000 wird die Speicherzelle 1033 vom Bedienungspult aus mit 11 belegt; der Lochstreifen kann dann über Abtaster 1 gelesen werden. Das Leseprogramm übersetzt den Lochstreifen folgendermaßen:

t2050t — 2049 → 1009, Vorbereitung für die Speicherung der folgenden Informationen

316.10
.
.
4.31 } Speicherung nach { 2050
.
.
2053

T — Stop des Leseprogramms mit Z0P2000

Die Berechnung der Formel leistet das folgende kleine Programm:

b2050	a → X
by2051	b → Y
*	a * b → X,Y
	Da das Resultat nicht mehr als 10 Stellen hat, ist es nicht erforderlich, die folgende Division mit doppelter Zahlenlänge durchzuführen, d.h. die oberen 10 Resultatstellen brauchen nicht nach 1011 umgespeichert zu werden.
bi2.1052	c → X, 2 → ZR2 Das Produkt a * b und c haben den Skalenfaktor q = 2. Das Resultat einer normalen Division hätte folglich den Skalenfaktor q = 0. Das Resultat der Division muß aber noch mit einer Zahl addiert werden, die zwei Stellen hinter dem Dezimalpunkt aufweist (q (d) = 2). Da nur Zahlen mit gleichem Skalenfaktor addiert werden können, muß durch eine Division mit Rundung dafür gesorgt werden, daß auch $\frac{a * b}{c}$ den Skalenfaktor 2 besitzt. Der Skalenfaktor des Resultats einer Division mit Rest ist um zwei zu erhöhen; diese Zahl wird durch den Befehl i2.1052 nach ZR2 gebracht.
/R	$\langle Y \rangle \cdot 10^2 / \langle X \rangle$ → X und Y, gerundet
a1053	$\langle X \rangle$ +d → X
dx	Dieser Befehl wird erst in Abschnitt 4 erläutert, er veranlaßt den Druck einer Zahl aus dem X-Register. Die Druckanordnung wird vorerst noch außer Acht gelassen.
zop2000	Stop des Programmablaufs, die Maschine steht zum weiteren Einlesen bereit.

Es ist zweckmäßig, Daten und Programm auf dem gleichen Streifen abzulochen; auf diese Weise kann man den unnötigen Stop des Leseprogramms nach dem Einlesen der Eingangswerte vermeiden. Der Streifen sieht dann so aus:

t2050t	Speicherplatzdefinition, 2049 → 1009
316.10	
48	
7.58	
4.31	
b2050	
by2051	Speicherung nach
*	2050 bis 2061
bi2.1052	
/R	
a1053	
dx	
zop2000	
zoe2054e	Stop des Einlesens, Vorbereitung des Programmstarts

Der abschließende Bandbefehl stoppt das Einlesen, nach Betätigung der Starttaste beginnt die Rechenanlage einen linearen Programmablauf mit dem in s2054 stehenden Befehl. Dieser Befehl (b2050) ist gerade der erste Befehl unseres kleinen Programms.

Locht man zum Schluß statt zoe2054e den Bandbefehl e2054e, so beginnt die Maschine sofort nach dem Einlesen zu rechnen, sie stoppt in diesem Fall erst, nachdem sie das Resultat ausgegeben hat.

3.5.2 Lesebefehle des Externcode der ZUSE Z 31

Die Lesebefehle haben die Aufgabe, den Transport von Daten (Zahlen und Text) in den Arbeitsspeicher der ZUSE Z 31 zum Zwecke ihrer Verarbeitung zu veranlassen. Der Lochstreifen dient als Nachschubspeicher, aus dem der jeweilige Bedarf an Daten in die Maschine gelesen werden kann. Das heißt, auch bei geringerer Schnellspeicherkapazität kann eine größere Datenmenge verarbeitet werden. Darüberhinaus können durch Lesebefehle noch weitere Befehle, einzeln oder in Gruppen, in die Maschine eingelesen werden. Das können Bandbefehle sein oder gewöhnliche Befehle, die gespeichert werden sollen. Hiermit ist die Möglichkeit gegeben, vom Lochstreifen her das vorher eingelesene Programm wahlweise zu steuern, zu ändern oder zu ergänzen.

In der Speicherzelle 1033 muß stets angegeben sein, welcher Leser durch einen Lesebefehl angesprochen werden soll: bei ⟨1033⟩=11 wird über Leser 1, bei ⟨1033⟩=13 über Leser 2 eingelesen.

Die Lesebefehle beginnen mit dem Symbol H (Hole...); ein zweites Symbol unterscheidet die verschiedenen Lesebefehle und gibt einen Hinweis auf ihre Wirkung. Die Lesebefehle sind im einzelnen:

HX Wirkung: Eine Information - d.h. hier eine Zahl, ein Befehl oder eine TE-Konstante - wird vom Lochstreifen gelesen, in ihre maschineninterne Darstellung übersetzt und im X-Register zur Verarbeitung bereitgestellt.

Mit dem Befehl HX können Texte nur dann eingelesen werden, wenn sie nicht mehr als fünf Zeichen umfassen.

Trifft das Leseprogramm bei der Ausführung des Befehls HX einen Bandbefehl, so wird zunächst der Bandbefehl ausgeführt, anschließend wird weiter eingelesen. Nach Sprung-Band-Befehlen oder Sonderzeichen, die ebenfalls zur Programmfortzetzung im Schnellspeicher führen, muß das jeweils aufgerufene Programm gegebenenfalls eine Rückkehr ins Leseprogramm veranlassen.

Die Sonderzeichen τ und ε, die ja in der Regel das Ende einer Datenfolge anzeigen, werden beim Lesen mit HX wie der Bandbefehl e1042e behandelt, d.h. nach dem Einlesen eines dieser Zeichen springt das Leseprogramm zur Fortsetzung des Hauptprogramm nach s1042.

In der Praxis will man häufig eine Folge von Informationen nacheinander einlesen, sie in aufeinanderfolgenden Schnellspeicherzellen ablegen und anschließend verarbeiten. Damit das Leseprogramm diese Aufgabe erfüllen kann, müssen ihm zwei Angaben gemacht werden:

1. Die Adresse derjenigen Schnellspeicherzelle, in der mit dem Speichern begonnen werden soll und
2. eine Angabe darüber, wann der Einlesevorgang beendet ist und der Fortgang im Hauptprogramm erfolgen soll.

Beim Programmeinlesen mit dem Sprungbefehl P2000 werden Anfangsspeicherplatz und Ende des Lesevorgangs durch Bandbefehle (TnT, EnE) auf dem einzulesenden Lochstreifen definiert; auch die Sonderzeichen τ und ε beenden das Einlesen.

Beim Einlesen mit Lesebefehlen ist es zweckmäßiger, wenn die Adresse des ersten Speicherplatzes vom Programm her angegeben wird, schon wegen der sich so ergebenden Möglichkeiten der Substitution und Modifikation. Das Ende des Einlesevorgangs wird jedoch auch hier besser vom Lochstreifen bestimmt, da so das Arbeiten mit Datenfolgen von variabler Länge erleichtert wird.

Die Adresse der ersten zu belegenden Speicherzelle ist im Y-Register anzugeben; ein τ oder ein ε auf dem Lochstreifen beendet das Einlesen.

HY Wirkung: Informationen werden von Lochstreifen gelesen und nach s⟨Y⟩, s⟨Y⟩+1, ... gespeichert, bis das Zeichen τ oder ε gelesen wird. Danach wird im Hauptprogramm fortgefahren. Das Zeichen τ bzw. ε wird nicht mehr gespeichert.

Bandbefehle und bestimmte Sonderzeichen (Anhang 4) unterbrechen auch hier den Einlesevorgang. Durch die ständige Bereitschaft des Leseprogramms, Bandbefehle auszuführen, wird eine äußert flexible Datenorganisation möglich. Ein ausführliches Anwendungsbeispiel, das den Rahmen dieser Druckschrift sprengen würde, wird gesondert veröffentlicht.

Auch beim Lesebefehl HY dient das Indexregister 1009 als Speicherzähler; es enthält nach Ausführung dieses Befehls (wie auch nach P2000) die Adresse der zuletzt beschriebenen Speicherzelle.

3.6 Fehleranzeigen des Leseprogramms

Das Leseprogramm der ZUSE Z 31 überprüft verschiedene Eingabewerte (z.B. TE-Konstanten und Druckbefehle) auf formale Richtigkeit.

Findet das Leseprogramm eine derartige Information auf dem Lochstreifen, die abweichend von den in 3.4 erläuterten Regeln dargestellt ist, so speichert es diese Information nicht, sondern läßt die zu ihrer Speicherung vorgesehene Zelle frei. Nachdem die Rechenanlage auf der Protokollschreibmaschine in der Form

n Darstellungsfehler

den Fehler und die Adresse n der freigelassenen Speicherzelle bekanntgegeben hat, stoppt sie mit dem Befehl

zop2011;

nach Betätigung der Starttaste wird der Programmablauf fortgesetzt,

Die Fehlerkontrolle ist beim ersten Einlesen eines Programms besonders nützlich. Mit Hilfe von Adressen- und Zehnertastatur kann der Programmierer die angezeigten Fehler sofort oder nach Abschluß des Programmeinlesens auf einfache Weise vom Bedienungspult aus korrigieren.

Auch der forgeschrittene Programmierer der ZUSE Z 31 wird mitunter eine nicht erlaubte Befehlskombination wählen, die -maschinenintern- eine Doppelbelegung einer Dezimalen des Befehlswortes bedeutet. Das Leseprogramm übersetzt so einen Befehl nicht; wie beim Darstellungsfehler wird eine Speicherzelle freigelaasen; mit dem Text

n Doppelbelegung

zeigt die Maschine die Art des Fehlers und den vorhergesehenen Speicherplatz des Befehls an, anschließend stoppt sie wie bei "Darstellungsfehler". Der gleiche Stop tritt ein, wenn ein Befehlszeichen hinter einer Adresse gelesen wird, das kein Bandbefehlszeichen ist.

3.7 Zusammenfassung in Stichworten

⟨1033⟩ bestimmt das Lesegerät

⟨1033⟩ = 11 : Leser 1

⟨1033⟩ = 13 : Leser 2

Die einzelnen Lesebefehle sind

Grundsymbol	Zusatzsymbole	Adresse
P H	X, Y	2000

Das Einlesen kann durch Bandbefehle und bestimmte Sonderzeichen unterbrochen werden (Steuerung des Programmablaufs vom Datenstreifen).

Bei den beiden Befehlen zum Einlesen in den Arbeitsspeicher dient Indexregister 1009 als Speicherzähler.

1009 wird belegt durch Bandbefehl TnT bzw. durch eine Angabe im Y-Register (bei HY).

Nach jedem Einlesen in den Speicher enthält 1009 die Adresse der zuletzt belegten Speicherzelle.

Bei Fehleranzeigen

n Darstellungsfehler

n Doppelbelegung

wird die fehlerhafte Information überlesen und die Speicherzelle sn für ihre Korrektur freigelassen; anschließend Stop mit dem Befehl

zop2011

im Befehlsregister. Weiterlesen nach START.

4. Lochstreifenausgabe/Schreibmaschinenausgabe

4.1 Das Ausgabeprogramm der ZUSE Z 31

Ein Ausgabeprogramm hat die Aufgabe, die Übertragung von Informationen aus dem Arbeitsspeicher der Datenverarbeitungsanlage auf das jeweilige Ausgabemedium zu steuern. Dabei muß es, soweit das nicht automatisch durch die Anschlußelektronik

der Zusatzgeräte übernommen wird, eine Umschlüsselung der internen Informationen in die für das entsprechende Ausgabemedium gewünschte oder erforderliche Darstellungsform durchführen. Das Ausgabeprogramm muß ferner in der Lage sein, auf Grund von Masken oder von Angaben über die Druckstellenzahl pro Information oder pro Druckzeile oder gemäß besonderer Tabellierungsbefehle jede gewünschte Druckanordnung zu erzeugen (z.B. bei der Ausgabe über Schreibmaschine oder Zeilendrucker).

Das mit der ZUSE Z 31 gelieferte, im Programmspeicher verdrahtete Ausgabeprogramm steuert die Ausgabe interne Informationen auf der Protokollschreibmaschine, auf Lochstreifen und Lochkarten.

Im Folgenden wird die Organisation des Ausgabeprogramms der ZUSE Z 31 für die Lochstreifen- und Schreibmaschinenausgabe erläutert; die Funktion des Ausgabeprogramms beim Verkehr der Zentraleinheit mit anderen Anschlußgeräten wird in den entsprechenden Abschnitten der Programmierungsanleitung behandelt.

4.2 Streifenlocher, Ausgabeschreibmaschine

An die Standardausführung der ZUSE Z 31 können zwei Streifenlocher angeschlossen werden; über jeden Locher können bis zu 150 Zeichen/s ausgegeben werden. Der ZUSE-Code verwendet 8-Kanal-Lochstreifen; mit standardmäßigen Streifenlochern können jedoch auch Lochstreifen mit geringerer Kanalzahl (z.B. 5-Kanal-Fernschreiblochstreifen) erstellt werden. Zur Ausgabe von Lochstreifen mit einem anderen als ZUSE-Code ist nur ein besonderes Ausgabeprogramm erforderlich; ein Umbau der Streifenlocher ist nicht nötig.

Das Leseprogramm für den Externcode der ZUSE Z 31 setzt voraus, daß eine Schreibmaschine an die Zentraleinheit angeschlossen ist, die zur Programmprüfung (Fehlermeldung, Protokolle, Speicherlisten) ohnehin kaum entbehrlich ist. Diese Schreibmaschine ist für ZUSE-Code eingerichtet; sie kann 10 Zeichen/s verarbeiten.

Das Ausgabeprogramm und die Anschlußelektronik der Ausgabegeräte übernehmen automatisch die Umschlüsselung des eindeu-

tigen internen ZUSE-Code in den ZUSE-Lochstreifencode, d.h. (siehe 3.3) das Einschieben der Zeichen für Groß- und Kleinumschaltung und die Zuordnung der gleichen Lochkombination zu je zwei entsprechenden Zeichen auf Groß- und Kleinumschaltung (z.B. aA, 1 [ ).

In der Speicherzelle 1022 muß dem Ausgabeprogramm angegeben werden, auf welchem Gerät Informationen ausgegeben werden sollen.

Es bedeutet

⟨1022⟩ = 17, daß Streifenlocher 1,
⟨1022⟩ = 20, daß Streifenlocher 2 und
⟨1022⟩ = 26, daß die Schreibmaschine

zur Ausgabe benutzt werden soll.

4.3 Tabellierung

Während eine Textinformation die zu ihrer Darstellung erforderlichen Tabellierungszeichen (d.h. Zeichen für Zwischenraum, Wagenrücklauf und Tabulatorsprung) enthält, muß bei der Ausgabe anderer Informationen (d.h. in erster Linie bei der Ausgabe von Zahlen) angegeben werden, wieviele Zwischenräume hinter jeder Information erscheinen und wieviele Informationen in einer Zeile ausgegeben werden sollen.

Diese Angaben sind sowohl bei der Schreibmaschinen- als auch bei der Lochstreifenausgabe erforderlich, da die erstellten Lochstreifen, soweit sie nicht ausschließlich zum Wiedereinlesen bestimmte Zwischenergebnisse enthalten, später auf einer lochstreifengesteuerten Schreibmaschine (Programmierungstisch) ausgeschrieben werden müssen.

Die Angaben zur Tabellierung nimmt die Speicherzelle 1023 auf, vom Ausgabeprogramm wird

⟨1023⟩ $_{10-2}$ als Anzahl der Informationen pro Zeile und

⟨1023⟩ $_{1}$ als Zahl der Zwischenräume, die hinter jeder Information auszugeben sind, gedeutet.

Das Ausgabeprogramm muß zu jedem Zeitpunkt wissen, wieviele Informationen in der Zeile, in der es sich gerade befindet, noch ausgegeben werden müssen. Als Zähler für diese Aufgabe dient ihm die Speicherzelle 1025. Der Inhalt von s1025 wird nach der Ausgabe einer Zahl, eines Befehls oder einer TE-Konstanten um eines vermindert; ist er negativ geworden, so erscheint hinter der zuletzt ausgegebenen Information nicht $\langle 1023 \rangle_1$-mal Zwischenraum, sondern Wagenrücklauf als Endezeichen; gleichzeitig wird 1025 mit dem um eins verminderten Inhalt von $s1023_{10-2}$neu belegt.

Der Programmierer kann das von ihm gewünschte Druckbild auch auf andere Weise erzeugen, und zwar mit Hilfe der

4.3.1 Tabellierbefehle

Mit Hilfe von sechs verschiedenen Befehlen kann der Programmierer jede gewünschte Tabellierung erreichen.

DZW1 Wirkung: Es wird einmal Zwischenraum ausgegeben
DTB1 Wirkung: " " " Tabulatorsprung ausgegeben
DWR1 Wirkung: " " " Wagenrücklauf/Zeilenvorschub ausgegeben.

Gleichzeitig wird s1025 mit $\langle s1023 \rangle_{10-2}-1$ neu belegt, d.h. dem Ausgabeprogramm wird angegeben, daß es sich am Anfang einer neuen Zeile befindet.

Zur mehrmaligen Ausgabe der Tabellierungszeichen werden die folgenden drei Befehle benutzt; die gewünschte Anzahl ist in jedem Fall im X-Register anzugebe. Während der Ausgabe wird $\langle X \rangle$ auf Null heruntergezählt.

DZW Wirkung:
DTB Wirkung:
DWR Wirkung:
} Es wird $\langle X \rangle$ -mal { Zwischenraum / Tabulatorsprung / Wagenrücklauf } ausgeg.

Nach DWR wird s1025 wie nach DWR1 neu belegt.

Ist $\langle X \rangle$ keine von Null verschiedene positive Zahl, so erfolgt keine Ausgabe; der Inhalt des X-Registers bleibt in diesem Fall erhalten.

Alle Tabellierbefehle bewirken eine Ausgabe auf dem durch ⟨1022⟩ bestimmten Gerät.

4.4 Ausgabebefehle

Alle Ausgabebefehle beginnen wie die Tabellierbefehle mit dem Symbol D (Drucke ...); weitere Symbole unterscheiden die einzelnen Druckbefehle und geben einen Hinweis auf ihre Wirkung.

Die Ausgabe eines Maschinenwortes gestattet der Befehl

DX Wirkung: Die im X-Register stehende Information wird ausgegeben. Handelt es sich dabei um eine Zahl, einen Befehl oder eine TE-Konstante, so wird außerdem ⟨1025⟩ um eins vermindert. Hinter jeder dieser Informationen erscheint ein Endezeichen, und zwar

$\langle 1023 \rangle_1$-mal Zwischenraum, wenn ⟨1025⟩>0, und Wagenrücklauf, wenn ⟨1025⟩ = 0 ist.

Im letzten Falle wird 1025 mit $\langle 1023 \rangle_{10-2^{-1}}$ neu belegt (siehe 4.3).
Hinter einem Textwort, das ohne Textanfangs- und -endekennzeichen ausgegeben wird, erscheint kein Endezeichen; auch wird nach der Ausgabe eines Textes mit DX der Inhalt von s1025 nicht verändert.

Analog zur Informationseingabe benötigt das Druckprogramm, wenn eine Folge von Informationen aus aufeinanderfolgenden Schnellspeicherzellen ausgegeben werden soll, Angaben über die Adresse derjenigen Speicherzelle, bei der mit der Ausgabe begonnen werden soll, und über das Ende des Ausgabevorgangs.

Wie bei den Lesebefehlen ist die Adresse der ersten anzusprechenden Speicherzelle im Y-Register anzugeben; das Ende des Ausgabevorgangs wird durch den Inhalt des X-Registers bestimmt.

DY Wirkung: Die im X-Register angegebene Anzahl von Zahlen, Befehlen oder TE-Konstanten wird aus den Speicherzellen s ⟨Y⟩ , s ⟨Y⟩ +1,... ausgegeben.

Trifft das Ausgabeprogramm bei Ausführung dieses Befehls Klartext im Speicher an, so wird dieser ebenfalls ausgegeben, aber nicht mitgezählt; d.h. zwischen den ⟨X⟩ Zahlen, Befehlen und TE-Konstanten kann beliebig viel mitauszugebender Text stehen.

Die Tabellierung erfolgt nach den Angaben in s1023 und s1025 (siehe 4.3).

Ein besonderer Befehl dient der Ausgabe von Text; auch hier ist Voraussetzung, daß die Adresse des Anfangsspeicherplatzes im Y-Register steht.

DT Wirkung: Text wird aus s ⟨Y⟩ , s ⟨Y⟩ +1,... ausgegeben, bis das Ausgabeprogramm ein Textendezeichen; oder eine Speicherzelle antrifft, die keinen Text enthält, Textanfangs- und -endekennzeichen werden nicht ausgegeben; durch DT werden die Inhalte von s1023 und s1025 nicht verändert.

Besonders bei kommerziellen Aufgaben muß häufig das Datum gedruckt werden. Dies erleichtert ein Sonderbefehl:

DAT Wirkung: Der Inhalt der letzten 6 Stellen von s1040 wird in durch Punkte getrennten Zweiergruppen ausgegeben.

Z.B. ⟨1040⟩ = 00000280563 wird ausgegeben als 28.05.63 (d.h. 28. Mai 1963).

4.5 Ausgabeform

Im Abschnitt 1.1 wurde die Darstellung von Informationen in der ZUSE Z 31 beschrieben. Danach können am Inhalt der Vorzeichenstelle D11 in der Maschine Befehle (1,2,3,6,7,8), Zahlen (0,9), zifferverschlüsselter Text (5) und TE-Konstanten

(4) unterschieden werden. Die Vorzeichenstelle wird auch vom Druckprogramm zur Unterscheidung der verschiedenen Informationstypen herangezogen.

4.5.1 Ausgabeform von Befehlen

Bei der Ausgabe von Befehlen wird eine bestimmte Reihenfolge der einzelnen Befehlssymbole eingehalten, die nur an wenigen Stellen zur Verbesserung des Lesbarkeit von der internen Befehlsstruktur abweicht.

Alle alphabetischen Befehlszeichen werden mit Kleinbuchstaben geschrieben; dadurch brauchen nur wenige Bereichsumschaltungen ausgegeben zu werden (d.h. die Druckgeschwindigkeit wird erhöht).

Jeder Befehl besteht aus einem oder mehreren der nachstehend erklärten Teile, die in der aufgeführten Reihenfolge ausgegeben werden.

Bedingungsteil: alle Bedingungszeichen werden in eine Klammer gesetzt. Beispiele: (jlpo)
(wyzkl)

Operationsteil: dazu gehören alle Operationszeichen, die nicht der Adressenumrechnung dienen. Hinter allen Operationszeichen, deren zweites Symbol eine Ziffer ist, wird zur Verbesserung des Lesbarkeit ein Punkt ausgegeben. Beispiele: bvxil.
azl.y

Adressenteil: An dieser Stelle werden auch die Befehlszeichen für Adressensubstitution und Adressenmodifikation ausgegeben; führende Nullen der (intern vierstelligen) Adresse werden unterdrückt, die Adresse Null wird nicht geschrieben. Beispiele: bx6+g1395
kv1.x7 # 3
rrv1.

Befehle des Externcode, die ein Unterprogramm aufrufen, werden als Ruf- (fp-)befehle für das entsprechende Unterprogramm ausgeschrieben. Eine Aufstellung der entsprechenden Befehle enthält Anhang 5.

4.5.2 Ausgabeform von Zahlen

Das Druckbild der mit einem Druckbefehl auszugebenden Zahlen kann der Programmierer selbst bestimmen. Er gibt dazu in

1024_{4-3} die Zahl p der Stellen vor und in
1024_{2-1} die Zahl q der Stellen nach dem Dezimalpunkt an.

Die letzten p+q (≤ 10) Stellen einer Zahl werden dann als Dezimalzahl aufgefaßt und entsprechend ausgegeben.

Wird durch q = 0 die Ausgabe ganzer Zahlen verlangt, so erscheint hinter der Zahl kein Dezimalpunkt; die Angabe p = 0 hat zur Folge, daß die Zahl mit einem Dezimalpunkt beginnt.

Die p Stellen vor dem Dezimalpunkt werden mit Nullenunterdrückung ausgegeben. D.h. unwesentliche (führende) Nullen werden durch Zwischenräume ersetzt; stehen nur Nullen vor dem Dezimalpunkt, so wird die letzte ausgegeben.

Das Vorzeichen erscheint, wie im kommerziellen Bereich üblich, hinter der Zahl, und zwar ein positives Vorzeichen als Zwischenraum und ein negatives als -.

Beispiele: (␣ bedeutet Zwischenraum)

p=4, p=2, d.h. $\langle 1024 \rangle_{4-1}$ = 0402

Maschinenwort	Druckbild
00000609493	6094.93␣
99999981220	␣187.80-
01200002618	␣␣26.18␣
99999999999	␣␣␣0.01-

4.5.3 Ausgabeform von Text

Klartext wird grundsätzlich von der Maschine so ausgegeben, wie er eingelesen wurde, d.h. unter Berücksichtigung aller Satz-

zeichen, Tabellierungszeichen und der Groß- und Kleinumschaltung. Das Textanfangszeichen " und das Textendezeichen ; werden nicht mit ausgegeben.

4.5.4 Ausgabeform von TE-Konstanten

Nach dem Kennzeichen te wird für jede Tetrade ein Zeichen ausgegeben: Ziffern für die gültigen Zeichen und Buchstaben für Pseudotetraden gemäß folgender Tabelle:

bin. Wert	dez. Wert	Druckzeichen
0000	-	a
000L	-	b
00L0	-	c
00LL	0	0
0L00	1	1
0L0L	2	2
0LL0	3	3
0LLL	4	4
L000	5	5
L00L	6	6
L0L0	7	7
L0LL	8	8
LL00	9	9
LL0L	-	d
LLL0	-	e
LLLL	-	f

Beispiel: Ein Maschinenwort, das auf den Dezimalen D1 ... D10 jeweils die entsprechende Binärzahl enthält, wird in der Form

te76543210cb

gedruckt.

4.6 Ausgabe in intern-ziffernverschlüsselter Form

In Abschnitt 3.4.2 wurde gesagt, daß das Leseprogramm auch intern-ziffernverschlüsselte Informationen einlesen kann. Das Druckprogramm ist in der Lage, Informationen in dieser Form auszugeben.

Die Ausgabe ziffernverschlüsselter Informationen veranlaßt der schon in 4.4 besprochene Befehl DY, wenn im X-Register eine negative Zahl steht. Die Adresse der ersten auszugebenden Speicherzelle ist im Y-Register anzugeben.

DY Wirkung: (bei ⟨X⟩ < 0) Der Inhalt von - ⟨X⟩ Speicherzellen wird, beginnend bei s ⟨Y⟩ , in ziffernverschlüsselter Form ausgegeben. D.h. jedes Zahlen-, Befehls- oder Textwort erscheint als 11-stellige Zahl. TE-Konstanten werden bei Ausführung dieses Befehls in der in 4.5.4 beschriebenen Form ausgegeben.

Hinter dem Inhalt jeder Speicherzelle erscheint Wagenrücklauf als Endezeichen.

Ein ausgetestetes Programm wird sich der Programmierer in der Regel in dieser Form ausgeben lassen, da es so mit Maximalgeschwindigkeit wieder eingelesen werden kann.

4.7 Ausgabe einer Speicherliste

Die Prüfung eines in die ZUSE Z 31 eingelesenen Programms wird dem Programmierer dadurch wesentlich erleichtert, daß er mit einem Befehl den Druck einer Liste bewirken kann, die, in zwei Spalten nebeneinander angeordnet, Adressen und Inhalte der ihn interessierenden Schnellspeicherzellen enthält.

Anfangs- und Endadresse der Liste werden in s1034 und s1036 vorausgesetzt; die beiden Speicherzellen können durch Bandbefehle (siehe 3.5.1.1) leicht mit den gewünschten Adressen belegt werden.

⟨1022⟩ bestimmt auch hier das Ausgabegerät; die Angabe zum Zahlendruck in 1024 wird nicht beachtet, sondern zerstört.

DS Wirkung: Druck einer Speicherliste von s (⟨1034⟩ ∧ 1110) bis s (⟨1036⟩ ∧ 1110 +9) in externer Form (d.h. Befehle, Zahlen (10-stellig, ganz), TE-Konstanten und Text werden gemäß 4.5 dargestellt) mit Seiteneinteilung.

Die Art der Seiteneinteilung wird durch den Bedingungsspeicher 1 bestimmt:

bei ⟨B1⟩ = 0 wird nach jeweils fünfzig Zeilen 10 x WR ausgegeben,

bei ⟨B1⟩ = L jedoch 5 x WR, 100 x Minuszeichen (Perforation), weitere 5 x WR; außerdem vor jeder Zellennummer ein Tabulatorsprung.

Die letzte Seite der Speicherliste wird, falls erforderlich, durch zusätzliche Ausgabe von Wagenrückläufen auf das Format einer vollen Seite gebracht.

Ist die Liste vollständig gedruckt, so stoppt das Programm mit dem Befehl

Z0P2000

im Befehlsregister, d.h. die Rechenanlage ist lesebereit.

Beispiel:

```
1180  (gr)e1164
1181  zoe1160
1182            23
1183          121-
1184  Ausga
1185  ng
1186  i2.8
1187  11
1188  teaa03adf4cb
1189  kcy1000
```

4.8 Zusammenfassung in Stichworten

⟨1022⟩ bestimmt das Ausgabegerät:

⟨1022⟩ = 17 : Locher 1
⟨1022⟩ = 20 : Locher 2
⟨1022⟩ = 26 : Schreibmaschine

⟨1023⟩ und ⟨1025⟩ regeln die Druckanordnung:

⟨1023⟩ $_{10-2}$ = Anzahl der Informationen/Zeile
⟨1023⟩ $_{1}$ = Anzahl der Zwischenräume nach jeder Information
1025 = Merkspeicher für den Stand in der Zeile (enthält die um 1 verminderte Anzahl der in der augenblicklich zu druckenden Zeile noch auszugebenden Informationen).

Die Inhalte von 1023 und 1025 bleiben beim Druck von Klartext ohne Einfluß.

⟨1024⟩ ist die Druckanordnung für Zahlen

⟨1024⟩ $_{4-3}$ = Anzahl p der Stellen vor dem Dezimalpunkt
⟨1024⟩ $_{2-1}$ = Anzahl q der Stellen hinter dem Dezimalpunkt

Zahlendruck mit Nullenunterdrückung und Vorzeichen (- oder Zwischenraum) hinter der Zahl.

Die einzelnen Ausgabebefehle sind

Grundsymbol	Zusatzsymbol	Adresse
D	ZW1, TB1, WR1, ZW, TB, WR X, Y, T AT, S	

Bei den Befehlen DX, DY und DT wird der Inhalt des Indexregisters 1008 zerstört; es dient bei Befehlen zur Ausgabe mehrerer Informationen als Speicherzähler.

Nach jeder Ausgabe aus dem Speicher enthält 1008 die Adresse der zuletzt ausgegebenen Speicherzelle.

Druck in ziffernverschlüsselter Form (zum schnellen Wiedereinlesen) bewirkt DY mit negativer Anzahlangabe in X.

5. Programmierung der ZUSE Z 31 mit relativen und symbolischen Adressen

5.1 Relative Adressierung

Beim Erstellen eines Programmes weiß der Programmierer in der Regel noch nicht, an welcher Stelle des Schnellspeichers dieses Programm später einmal stehen soll. Er wird daher eine unbestimmte Anfangsadresse n annehmen, auf die er sich in einer vorläufigen Speicherliste ähnlich wie im folgenden Beispiel bezieht:

```
n+0   tf14+n
  1   jlil.0
  2   byl3+n
  3   bll
  :   :
```

Er kennzeichnet also alle Adressen, die sich auf die Anfangsadresse n beziehen, durch die Symbole +n als Relativadressen, die bei der endgültigen Adressierung des Programms noch um seine Anfangsadresse vermehrt werden müssen. Der Programmierer könnte genauso das Zeichen r (relativ) zur Kennzeichnung dieser Adressen verwenden, also etwa

```
0   tl4r
1   jl.il.0
2   byl3r
3   bll
:   :
```

in seine Speicherliste schreiben.

Auf Befehlszeichen hinter der Adresse eines Befehls reagiert das Leseprogramm (siehe 3.6) im allgemeinen mit der Fehleranzeige "n Doppelbelegung "; zwei Ausnahmen von dieser Regel hat der Leser bereits im Abschnitt 3.5.1.1 kennengelernt, nämlich die "Bandbefehlszeichen" t und e in den Befehlen tnt und ene. Jedes dieser Zeichen veranlaßt das Leseprogramm, wenn es hinter einer Adresse auftritt, etwas vom normalen Einlesevorgang Abweichendes zu tun: einmal wird 1009 belegt, im anderen Fall wird der Sprungbefehl sofort ausgeführt.

Das Zeichen r hinter einer Adresse wird vom Leseprogramm ebenfalls als Bandbefehlszeichen gedeutet: es veranlaßt das

Leseprogramm, auf die bereits eingelesene Adresse des Befehls den Inhalt von s1034 zu addieren und mit dem Lesen fortzufahren.

Die Umadressierung eines relativ adressierten Programms führt das Leseprogramm der ZUSE Z 31 folglich automatisch durch, wenn

1. wenn Relativadressen des Programms durch den Buchstaben r hinter der Adresse gekennzeichnet sind und wenn
2. s1034 die Bezugsadresse, d.h. die Anfangsadresse des Programms enthält.

s1034 kann auf einfache Weise mit der Bezugsadresse n belegt werden: durch den Bandbefehl

bcne

der ja, wie in 3.5.1.1 beschrieben, eine Belegung des "Hilfs-X-Registers" s1034 mit der Zahl n bewirkt.

Den Bandbefehl, der die Relativadresse definiert, wird man häufig zusammen mit dem tnt-Befehl zur Speicherplatzdefinition auf einem besonderen Streifen ablochen und diesen sogenannten Vorausstreifen dann vor dem eigentlichen Programmstreifen einlesen.

Das nach dem Abschnitt Bandbefehle aufgeführte kleine Programmbeispiel würde relativ adressiert so aussehen:

```
316.10
48
7.58
4.31
b0r
bylr
*
i2.2      *)
b2r       *)
/R
a3r
dx
zop2000
zoe4re **)
```

Beim Vergleich mit dem ursprünglichen Beispiel fällt zweierlei auf.

1. Die beiden mit *) gekennzeichneten Befehle waren dort zu einer Befehlskombination zusammengefaßt. Das ist bei einem relativ adressierten Programm, das an jede Stelle des Speichers eingelesen werden kann, nicht mehr möglich, da ja nicht gesichert ist, daß die Adresse 2r die Endziffer 2 hat.

2. Der abschließende Bandbefehl enthält ebenfalls eine Relativadresse und damit 2 Bandbefehlszeichen. Das ist erlaubt: bei Bandbefehlen tnt und ... ne kann die Adresse absolut, relativ oder (siehe 5.2) symbolisch sein.

Zur Speicherung des Programms ab 2873 wird folgender Vorausstreifen abgelocht und mit P2000 eingelesen:

```
t2873t
bc2873e
т
```

Nach dem Einlesen stoppt die Maschine mit Z0P2000; der Programmstreifen kann eingelesen werden. Er wird übersetzt und gespeichert

nach	als
2873	31610
4	48
5	758
6	431
7	b2873
8	by2874
9	*
80	i2.2
1	b2875
2	/R
3	a2876
4	dx
5	zop2000

Viele Programme verwenden eine große Zahl von Unterprogrammen. Beim Einlesen einer solchen Programmfolge - alle Programme seien relativ adressiert - ist es unangenehm, wenn man vor jedem Programm einen Vorausstreifen einlesen muß.

Diese Umständlichkeit läßt sich jedoch vermeiden. Eine Möglichkeit dazu gibt der Bandbefehl

bcx9+1e

der eine Belegung von 1034 mit dem um 1 vermehrten Inhalt des Leseprogramm-Speicherzählers 1009 veranlaßt.

Vor dem ersten Befehl eines Unterprogramms abgelocht, bewirkt dieser Befehl, daß alle Relativadressen innerhalb dieses Unterprogramms auf dessen Anfangsadresse bezogen werden. (Bem.: tnt hat die Wirkung n-1 → 1009, also sind die Befehlsfolgen tnt, bcne und tnt, bcx9+1e äquivalent).

Eine elegantere Möglichkeit, die erwähnte Umständlichkeit zu vermeiden, bietet ein weiteres Bandbefehlszeichen, das ein Arbeiten ohne den "Bezugsadressenspeicher" 1034 gestattet.

Das Symbol s hinter der Adresse eines Befehls veranlaßt das Leseprogramm, zu der bereits eingelesenen Adresse den um 1 vermehrten Inhalt des Indexregisters 1009 zu addieren, d.h. die Adresse, unter der der eingelesene Befehl (mittels Indexbefehls tx9# 1) abgelegt wird.

Beispiel: Wird der Befehl

e14s

vom Leseprogramm nach s1794 gespeichert, so erscheint er dort als

e1808

ens heißt also: "Springe n Zellen weiter".

Diese "Adressierung relativ zum eigenen Speicherplatz" ist nicht nur in der Vorwärtsrichtung möglich, unter den im Beispiel gemachten Voraussetzungen wird

e-12s

vom Leseprogramm als

e1782

nach s1794 gespeichert.

e-ns heißt also: "Springe n Zellen zurück".

Das schon angeführte Programmbeispiel kann bei dieser Adressierungsform so geschrieben werden:

```
b9s
by9s
*
i2.2
b7s
/R
a6s
dx
zop2000
316.10
48
7.58
4.31
zoe-13se        *)
```

*) Auch Bandbefehle, die nicht gespeichert werden, können "relativ zum eigenen Speicherplatz" adressiert werden; sie sind dabei wie normale Befehle zu behandeln.

Relative Adressierung mit den Bandbefehlszeichen r und s, absolute und (s.u.) symbolische Adressierung können beliebig gemischt in einem Programm auftreten.

5.2 Symbolische Adressierung

Das Führen einer Speicherliste ist bei relativer oder absoluter Programmierung unerläßlich, da schon bei Programmen mit mehr als 20 Befehlen das Verfahren, die richtige Adresse beispielsweise für einen Sprungbefehl durch Abzählen zu ermitteln, sehr fehleranfällig ist. Bei diesen Arten der Adressierung ist es sehr unangenehm, wenn einmal irgendwo ein Befehl vergessen wurde oder wenn durch geschicktere Programmierung mitten in einem umfangreichen Programm Befehle eingespart werden können. Da jeder Programmierer bestrebt ist, den für ein Programm verfügbaren Speicherraum optimal auszunutzen, wird er in jedem dieser Fälle den auf die Änderungsstelle folgenden

Programmteil - in erster Linie hinsichtlich der Sprungadressen - ebenso wie den der Korrektur vorangehenden Teil durchsehen und an mehreren Stellen ändern müssen.

Dabei kann leicht eine notwendige Änderung übersehen werden; ein einziger falscher Sprungbefehl macht das ganze Programm falsch und ist beim Testen nicht immer leicht zu ermitteln.

Der Programmierer hilft sich damit, daß er beim Erstellen eines Programmes zunächst einmal alle Befehle, die aus größerer 'Entfernung' angesprungen werden, mit symbolischen Markierungen versieht; das jeweilige Symbol schreibt er auch in den Adressenteil eines jeden Befehls, der auf eine der markierten Stellen Bezug nimmt.

In unserem Beispiel könnte das so aussehen:

```
FORMEL b      A
       by     B
       *
       i2.2
       b      C
       /R
       a      D
       dx
       zop2000
A      316.10
B      48
C      7.58
D      4.31
       zoe FORMEL e
```

Man wird in der Regel Symbole wählen, die auf Art und Ziel der an den betreffenden Stellen eingeleiteten Operationen Bezug nehmen. Auch Hilfsspeicher und Konstanten wird man sich häufig durch Symbole kennzeichnen, da beispielsweise die Befehle

```
t      SUMME
b      LOHN
e      STEUER
```

leichter den Zweck der jeweiligen Operation erkennen lassen als

```
t25r,
b-12s   oder
e3516.
```

Außerdem kann man so auf einfache Weise die Fehlerquelle des Abzählens ausschalten.

Nachdem ein Programm auf diese Weise entstanden ist, müßte sich - gäbe es keine Möglichkeit zur symbolischen Adressierung - der Programmierer daran machen, es auf absolute oder besser auf relative Adressen umzuschreiben - eine wenig interessante, mehr oder weniger mechanische Arbeit, die ihm die ZUSE Z 31 ohne weiteres abnehmen kann.

Die Zuordnung der Adressen in einem Programm (seine 'Adressierung') ist dem Leseprogramm möglich, wenn es die Symbole als solche erkennen kann, d.h. wenn die Symbole geeignet gekennzeichnet sind.

Das Leseprogramm der ZUSE Z 31 wertet den Doppelpunkt als Anfangs- bzw. Endkennzeichen eines Symbols.

Unser Beispiel wäre dann wie folgt zu schreiben:

```
:FORMEL:b :A:
        by :B:
        *
        i2.2
        b :C:
        /R
        a :D:
        dx
        zop2000
    :A :316.10
    :B :48
    :C :7.58
    :D :4.31
       zoe :FORMEL :e
```

Beim Übersetzen eines symbolisch adressierten Programms muß das Leseprogramm im Arbeitsspeicher ein sogenanntes Adressbuch führen, das im Laufe der Übersetzung mit einer Tabelle aller Symbole und der ihnen zugeordneten Adressen (s.u) gefüllt wird. Ferner wird ein sogenanntes Vormerkbuch benötigt; seine Funktion wird unten erläutert.

Mit Hilfe eines besonderen Bandbefehls wird die Lage des Adressbuches im Speicher festgelegt. Der Befehl

mab

(m = vierstellige Zahl) bewirkt, daß der Speicherbereich von sm bis s ⟨1009⟩ zur Aufnahme von Adressbuch und Vormerkbuch reserviert wird.

Trifft das Leseprogramm beim Übersetzen eines symbolisch adressierten Programms auf ein Symbol, so kann es feststellen, ob das Symbol im Adressenteil eines Befehls steht.
Ist das nicht der Fall - wie in unserem Beispiel bei FORMEL - so soll das Symbol den Speicherplatz der nächsten Information bezeichnen. Das Symbol und die Adresse, unter der die nächste Information zu speichern ist, werden in zwei aufeinanderfolgende Zellen des Adressbuchs eingetragen. Man sagt, das Symbol sei definiert worden.

Eine Speicherzelle kann 5 Zeichen des ZUSE-Code aufnehmen; daher werden Symbole von der Maschine anhand ihrer ersten 5 Zeichen unterschieden. Verschiedene Symbole müssen sich in den ersten fünf Zeichen unterscheiden.

Beachtet der Programmierer diese Vorschrift nicht, so kann es vorkommen, daß ein Symbol - für die Maschine - zum zweiten Male definiert wird. Dann wird es nicht noch einmal ins Adressbuch eingetragen, sondern die Rechenanlage schreibt das Symbol (z.B. LESEN) in der Form

LESEN doppelt

auf der Schreibmaschine aus; anschließend stoppt sie mit

zop1763.

Nach START liest die Maschine so weiter, als ob sie das Symbol nicht zweimal gelesen hätte.

Steht das Symbol im Adressenteil eines Befehls, so wird durch Absuchen des Adressbuches festgestellt, ob dieses Symbol schon definiert ist und somit die ihm zugeordnete Adresse feststeht. Ist dies der Fall, so wird diese Adresse dem gelesenen Befehl hinzugefügt. Andernfalls muß der Adressenteil des Befehls noch unvollständig bleiben. In diesem Fall notiert das Leseprogramm das Symbol und die Adresse, unter der der unvollständig adressierte Befehls abgelegt werden muß, im Vormerkbuch.

Jedesmal, wenn ein Symbol definiert wird, wird das Vormerkbuch nach Vormerkungen zu diesem Symbol abgesucht. Etwa vorhandene Vormerkungen werden ausgeführt, d.h. die nunmehr bekannte zugeordnete Adresse wird in die vorgemerkten Befehle eingesetzt und die Vormerkung wird gelöscht.

Zur Programmprüfung ist es für den Programmierer wichtig zu wissen, welche Adressen das Leseprogramm den einzelnen Symbolen zugeordnet hat, d.h. er möchte den Inhalt des Adressbuchs kennen. Es gibt zwei Möglichkeiten, das Leseprogramm zum Druck des Adressbuchs zu veranlassen.

1. Druck während des Programmeinlesens

   Der Programmierer belegt z.B. mit Hilfe des Bandbefehls

   bcle

   die Speicherzelle 1034 mit einer von Null verschiedenen Zahl. Sowie im Verlauf des Einlesens ein Symbol definiert wird, schreibt die Anlage auf der Protokollschreibmaschine das Symbol und die zugeordnete Adresse in einer Zeile aus.

2. Druck nach dem Programmeinlesen

   Der Programmierer sorgt dafür, daß während des Programmeinlesens der Inhalt von s1034 Null ist; anschließend gibt er den Befehl

   dab.

   Daraufhin werden Adressbuch und Vormerkbuch ausgeschrieben,

in unserem Beispiel - bei Speicherung des Programms ab 2830 -

```
FORME 2830
A     2839
B     2840
C     2841
D     2842
```

Das Vormerkbuch, im Beispiel leer, ist durch eine Leerzeile vom Adressbuch getrennt.

Durch den Befehl

dvb

erreicht man, daß nur das Vormerkbuch ausgedruckt wird.

Das Vormerkbuch muß nach dem Einlesen eines symbolisch adressierten Programms leer sein, sonst ist das Programm unvollständig adressiert. Durch Schreibfehler beim Ablochen wird jedoch mitunter die eine oder andere Vormerkung stehen bleiben. Der Programmierer sollte sich daher nach dem Einlesen eines symbolisch adressierten Programms stets den Inhalt des Vormerkbuches ausdrucken lassen, bevor er mit dem Programmtest beginnt.

Zur Schreibweise der Symbole ist noch etwas zu sagen: in einem Symbol dürfen alle schreibenden Zeichen des ZUSE-Code auftreten, die keine Endezeichen sind; von den nichtschreibenden Betriebszeichen sind nur

rot, schwarz und ZW

zulässig, die alle drei überlesen werden. Die Endezeichen

,;WR und TAB sind verboten.

Dieses Verbot ist eine Sicherung gegen Schreibfehler: wird beim Ablochen eines symbolischen Programms einmal ein Doppelpunkt vergessen, so stößt das Adressierprogramm bald auf eins dieser verbotenen Zeichen (meist Wagenrücklauf). Es stoppt dann mit der Fehleranzeige

n Darstellungsfehler
zop2011

und läßt (siehe 3.6) die Speicherzelle n frei, um den Programmierer Gelegenheit zur nachträglichen Korrektur zu geben.

Beim Einlesen eines symbolisch adressierten Programms ist noch ein weiterer Fehlerstop möglich:

```
Adressbuch voll
zop1763.
```

Dieser Stop tritt ein, wenn das Adressbuch/Vormerkbuch keine weiteren Eintragungen mehr aufnehmen kann. Dann muß fast immer das Programm noch einmal mit größerem Adressbuch eingelesen werden; nur wenn keine Notierung im Adressbuch mehr nötig ist, darf durchgestartet werden.

Ein besonderer Befehl gestattet es, das Adressbuch von einem bestimmten Symbol ab zu löschen:

ABL:...: Wirkung: Das Adressbuch wird vom Symbol ... ab gelöscht. Steht ... nicht im Adressbuch, so wird auch nicht gelöscht.

Die gelöschten Symbole können anschließend mit neuer Bedeutung wieder verwendet werden.

Das Vormerkbuch kann nur durch Neudefinition des Adressbuchs gelöscht werden.

Die Möglichkeit, relativ zu einem Symbol zu adressieren, gestattet es dem Programmierer, die Zahl der Symbole einzuschränken und so für das Adressbuch weniger Speicherplatz zu verbrauchen. So wird z.B. durch den Befehl

```
b5:LOHN:
```

die 5.Speicherzelle nach der mit LOHN bezeichneten angesprochen, der Befehl

```
e-6:STEUER:
```

wird in einem Sprungbefehl auf die 6.Zelle vor der Speicherzelle mit dem Namen STEUER übersetzt.
Unser Programmbeispiel läßt sich bei Adressierung relativ zum Symbol so schreiben:

```
:FORMEL:   b:DATEN:
           byl:DATEN:
           *
           i2.2
           b2:DATEN:
           /R
           a3:DATEN:
           dx
           zop2000
:DATEN:    316.10
           48
           7.58
           4.31
           zoe:FORMEL:e
```

5.3 Zusammenfassung in Stichworten

1. Relative Adressierung

⟨1034⟩ = Bezugsadresse m, wird durch Bandbefehl

bcme oder

bcx9+1e belegt

Das Bandbefehlszeichen r hinter der Adresse kennzeichnet diese als Relativadresse:

Adresse + ⟨1034⟩ ⇒ Adresse.

Das Bandbefehlszeichen s hinter der positiven oder negativen Adresse bewirkt eine Adressierung relativ zum eigenen Speicherplatz:

Adresse + $\langle 1009 \rangle_{4-1}$+1 ⇒ Adresse.

2. Symbolische Adressierung

Adressbuchdefinition durch Bandbefehl

mAB;

das Adressbuch erstreckt sich dann von sm bis s $\langle 1009 \rangle_{4-1}$.

Ein Symbol darf aus beliebig vielen Zeichen des ZUSE-Code

bestehen, zur Unterscheidung werden nur die ersten fünf Zeichen herangezogen.

Verbotene Zeichen: ,; WR und TAB
überlesen werden: schwarz, rot, ZW

Anfangs- und Endkennzeichnung eines Symbols durch je einen Doppelpunkt.

Ein Befehl mit Symbol im Adressenteil wird gemäß

Adresse + zugeordnete Adresse ⇒ Adresse

übersetzt, d.h. Adressierung relativ zu einem Symbol in der Form (Beispiel)

b5:LOHN:
e-6:STEUER:

ist möglich.

Druck des Adressbuchs während des Einlesens, wenn

⟨1034⟩ ≠ 0

Druck des Vormerkbuchs durch

dvb

Druck von Adressbuch und Vormerkbuch durch

dab.

Fehleranzeigen:

Adressbuch voll zop1763	Neudefinition des Adressbuchs nötig, bei der nächsten Eintragung wird Adressbuch oder Programm zerstört.
...doppelt zop1763	Symbol ... tritt doppelt auf; bei START bleibt die zweite Definition unberücksichtigt
n Darstellungs-fehler zop2011	Verbotenes Zeichen im Symbol, Zelle sn wird freigelassen.

Beim Einlesen eines symbolisch adressierten Programms wird der Inhalt des Indexregisters 1008 zerstört.

Anhang 1

ZUSE-Code

Kanal 8	7	6	5	4	3	2	1	Dez. Wert	Zeichen kl.	gr.	In der ZUSE Z 31 1. Zchn.	2. Zchn.
o		o	o			o	o	0	0	/	0	90
		o	o		o			1	1	[	1	91
o		o	o		o		o	2	2	]	2	92
o		o	o		o	o		3	3	(	3	93
		o	o		o	o	o	4	4	)	4	94
		o	o	o				5	5	:	5	95
o		o	o	o			o	6	6	=	6	96
o		o	o	o		o		7	7	'	7	97
		o	o	o		o	o	8	8	"	8	98
o		o	o	o	o			9	9	→	9	99
	o					o	o	10	10	%	10	50
o	o				o			11	.	!	11	51
	o				o		o	12	+	■	12	52
	o				o	o		13	-	?	13	53
o	o				o	o	o	14	*	&	14	54
o	o			o				15	ZW	ZW	15	55
	o			o			o	16	,	;	16	56
	o			o		o		17	TAB	TAB	17	57
o	o			o		o	o	18	WR	WR	18	58
	o			o	o			19	τ	ε	19	59
o	o		o			o	o	20	a	A	20	60
	o		o		o			21	b	B	21	61
o	o		o		o		o	22	c	C	22	62
o	o		o		o	o		23	d	D	23	63
	o		o		o	o	o	24	e	E	24	64
	o		o	o				25	f	F	25	65
o	o		o	o			o	26	g	G	26	66
o	o		o	o		o		27	h	H	27	67
	o		o	o		o	o	28	i	I	28	68
o	o		o	o	o			29	j	J	29	69
o	o	o				o	o	30	k	K	30	70
	o	o			o			31	l	L	31	71
o	o	o			o		o	32	m	M	32	72
o	o	o			o	o		33	n	N	33	73
	o	o			o	o	o	34	o	O	34	74
	o	o		o				35	p	P	35	75
o	o	o		o			o	36	q	Q	36	76
o	o	o		o		o		37	r	R	37	77
	o	o		o		o	o	38	s	S	38	78
o	o	o		o	o			39	t	T	39	79
	o	o	o			o	o	40	u	U	40	80
o	o	o	o		o			41	v	V	41	81
	o	o	o		o		o	42	w	W	42	82
	o	o	o		o	o		43	x	X	43	83
o	o	o	o		o	o	o	44	y	Y	44	84
o	o	o	o	o				45	z	Z	45	85
	o	o	o	o			o	46	#	◊	46	86
	o	o	o	o		o		47	frei		47	87
o	o	o	o	o		o	o	48	schw.	schw.	48	88
	o	o	o	o	o			49	rot	rot	49	89
			o	o	o	o	o	-	KL	KL	Diese Zeichen gelangen nicht in die ZUSE Z 31	
		o		o	o	o	o	-	GR	GR		
	o	o	o	o	o	o	o	-	STOP-CODE			
o	o	o	o	o	o	o	o	-	IRRUNG			

Anhang 2

Befehle der Zentraleinheit (Übersicht)

Grundsymbol	Zusatzsymbole	Adresse
Sprungbefehle, Stopbefehl		
E P ZO	F F	n n
Transportbefehle		
B T	FP, Y, C, CY Y	n n
BF TF		n n
Verschiebebefehle		
R L	Y, R Y, L	
Arithmetische Befehle		
A S	FP, T, Y, TY, C, CY FP, T, Y, TY, C, CY	n n
* /	R /, R, /R	
Logische Befehle		
U K	FP, Y, C, CY FP, Y, C, CY	n n
Zählbefehle		
V MVX Z MZX	1, 2, X, Y 1, 2, X, Y	n n
Setzen der Bedingungsspeicher		
J N	1, 2, 3, 4, 5, 6 1, 2, 3, 4, 5, 6	

Grundsymbol	Zusatzsymbol	Adresse
Bedingungen		
	(W) (Y0) (10) (J1) (J2) (J3) (N4) (N5) (N6) (YZ) (1Z) (20) (NE) (P0) (NU) (SU) (NG) (GR) (KL) (Y5) (2Z)	
Adressensubstitution und -modifikation		
	G ...	n
	Xi+, Xi #, Xi+G, Xi # G	n

Anhang 3

Befehle zur Lochstreifenein- und -ausgabe sowie zur Schreibmaschinenausgabe (Übersicht)

Grundsymbol	Zusatzsymbole	Adresse
	Lesebefehle	
P H	X,Y	2000
	Ausgabebefehle	
D	ZW1, TB1, WR1 ZW, TB, WR X, Y, T AT, S	

Anhang 4

Aussprünge des Leseprogramms auf Sonderzeichen

Nach Einlesen von ... wird nach s... gesprungen.

⌜, ≤	1042	nur bei HX
10	1043	
*	1044	*)
#	1045	
%	1046	
!	1047	
■	1048	
?	1049	
◊	1051	
/	1052	*)
[	1053	
]	1054	
=	1055	
'	1056	
→	1057	
Dez.Werte 48,88	1058	
&	1059	

*) Der Sprung auf die Angegebene Adresse erfolgt nur dann, wenn unmittelbar vorher eine Zahl eingelesen wurde. Der Absolutbetrag dieser Zahl steht dann im Y-Register, bei ⟨ B2 ⟩ = L wurde ein Minuszeichen gelesen.

Anhang 5

Die Befehle für Multiplikation und Division sowie für die Ein- und Ausgabe werden vom Leseprogramm in Ruf-(FP-)Befehle für das entsprechende Unterprogramm übersetzt, und zwar

*	in	FP1856
* R	"	FP1872
/	"	FP1884
//	"	FP1886
/R	"	FP1924
//R	"	FP1926
HX	"	FP2003
HY	"	FP2006
DX	"	FP1348
DY	"	FP1338
DT	"	FP1577
DS	"	FP1951
DZW1	"	FP1307
DTB1	"	FP1309
DWR1	"	FP1311
DZW	"	FP1326
DTB	"	FP1329
DWR	"	FP1332
DAT	"	FP1938
DAB	"	FP1669
DVB	"	FP1672