

LGP-30

Programmierungsanleitung

Eurocomp GmbH · Elektronische Rechenanlagen
495 Minden 2 · Schillerstraße 72

I n h a l t

Einleitung	Seite 2
A) Aufbau des LGP-30	3
B) Zahlen- und Befehlsdarstellung	4
1. Das Dualsystem	4
2. Zahlendarstellung im LGP-30	6
3. Maschinenkomma und gedachtes Komma	7
4. Rechenregeln	8
5. Hexadezimalzahlen	9
6. Befehlsdarstellung	10
7. Die vier Phasen der Befehlsausführung	11
C) Codierung	11
1. Befehlsliste des LGP-30	12
2. Einfache Befehlskombinationen	15
3. Lineares Programm	21
4. Programmverzweigungen	22
5. Zyklisches Programm	23
6. Eingabeplan 10.4	24
7. Optimalisierung	25
D) Bedienungsanweisungen	27
1. Eingabe des Ladeprogramms 10.4	27
2. Eingabe von Programm-Lochstreifen	28
3. Ein- und Ausschaltvorschrift	29
E) Programmieren im Gleitkomma	30
1. Zahlendarstellung und Register	30
2. Befehlsliste	31
3. Ein- und Ausgabeformat	33
4. Programmbeispiel	34
5. Unterpläne	35
6. Programmierte Stops	36
F) Zusammenfassung der technischen Daten des LGP-30	37
G) Schlußbemerkung	38

E I N L E I T U N G

Der LGP-30 ist ein elektronischer, programmgesteuerter Universal-Ziffernrechenautomat mit gespeichertem Programm. Die wesentlichen Merkmale solcher Rechenautomaten sind Speicherfähigkeit, hohe Rechengeschwindigkeit, Entscheidungsfähigkeit und Flexibilität. Die Lösung komplizierter Probleme kann damit automatisch und ohne Eingriff eines Menschen erfolgen.

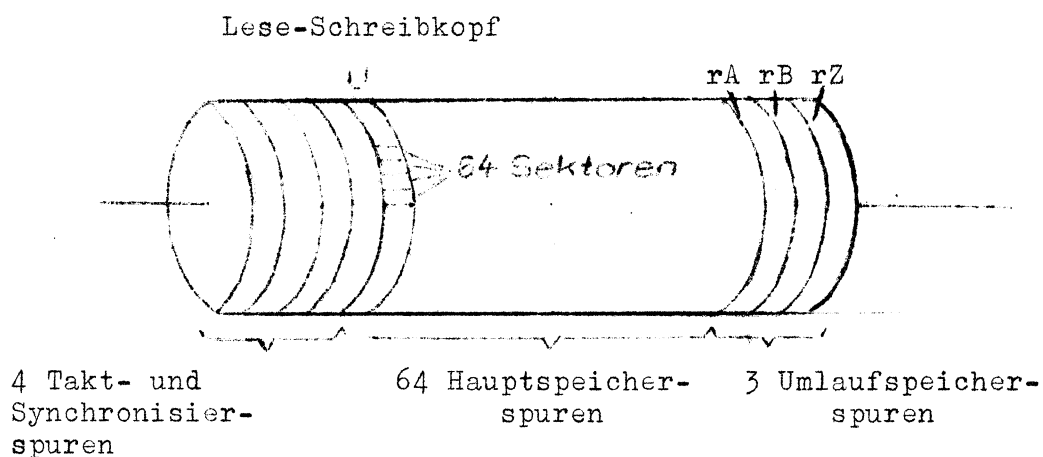
Einem elektronischen Ziffernrechenautomaten gibt man die Arbeitsanweisung zur Lösung eines Problems in Form eines Rechenplans ein. Er enthält in Maschinensprache die in logische Einzelschritte zerlegte Lösungsvorschrift, das sogenannte Programm. Es ist Aufgabe des Programmierers, dieses Programm aufzustellen. Es muß beliebig oft für neue Zahlenwerte verwendbar sein.

Die folgende Darstellung soll eine Anleitung zur Benutzung des LGP-30 sein, wobei sowohl die Programmierung als auch die Bedienung der Anlage erklärt werden. Es wird dabei nicht auf Fragen der Problemanalyse, der mathematischen Lösungsmethoden oder der Aufstellung von Ablaufprogrammen eingegangen. Die dazu notwendigen Kenntnisse und Anregungen können - da sie in allgemeingültiger Form darstellbar sind - den einschlägigen Lehrbüchern entnommen werden. Hier werden lediglich die den LGP-30 betreffenden speziellen Dinge behandelt.

A) Aufbau des LGP-30

1. Hauptspeicher

Als Speichermedium dient eine Magnettrommel mit 4000 U/Min. 64 Spuren dieser Trommel dienen als Hauptspeicher. Außerdem sind noch 4 Spuren mit Takt- und Synchronisierungssignalen vorhanden, 3 weitere Spuren sind als Umlaufspeicher (Register) ausgebildet.



Jede der 64 Hauptspeicherspuren umfaßt 64 Sektoren (Zellen) von je einer Wortlänge. Die Speicherkapazität beträgt also $64 \cdot 64 = 4096$ Worte. Zu jeder Spur gehört ein Kopf, der entsprechend einem Signal vom Kommandowerk liest oder schreibt. Die Kopfauswahl erfolgt elektronisch. Ein Wort besteht aus 32 Bits (Binärstellen).

2. Umlaufspeicher

Ein Umlaufspeicher hat je einen Lese- und Schreibkopf im Abstand von einer Wortlänge. Der Inhalt eines Umlaufspeichers wird ständig vom Lesekopf aufgenommen und über einen Verstärker und das Kommandowerk dem Schreibkopf zugeführt, der die gleiche oder abgeänderte Information um eine Wortlänge versetzt auf die Trommel schreibt. Dadurch ist der Inhalt eines Umlaufspeichers zu jeder Wortzeit ohne Zeitverlust verfügbar.

Es gibt 3 Umlaufspeicher, auch Register genannt: Akkumulator, Befehlsregister und Befehlszähler. Der Akkumulator nimmt einen Operanden und nach jeder Rechenoperation deren Ergebnis auf. Er hat einen zweiten Lesekopf, der vom ersten einen Abstand von einer Wortlänge weniger 1 Bit hat. Dadurch kann der Akkumulator um 31 Bit erweitert werden, um während der Multiplikation das sechzigstellige Produkt zu bilden. Nach der Multiplikation läuft dann die vom Programmierer gewählte Hälfte im Akkumulator um.

Das Befehlsregister enthält den gerade ausgeführten Befehl. Im Befehlszähler steht die Adresse der Speicherzelle, aus der der nächste Befehl geholt werden soll. Ist der Befehl gefunden, so wird der Zähler um "1" erhöht.

B) Zahlen- und Befehlsdarstellung

1. Das Dualsystem

Ebenso wie das Zehnersystem hat das Dualsystem bewertete Stellen. Die Ziffern sind aber hier Koeffizienten von Zweierpotenzen anstelle der Zehnerpotenzen des Dezimalsystems. Der Exponent der jeweiligen Basis (10 bzw. 2) wird durch die Stellenlage des Koeffizienten bestimmt.

Beispiel für Zehner-(Dezimal-)System:

$$\begin{array}{rcccc} 8647_{10} & = & 8 \times 10^3 & + & 6 \times 10^2 & + & 4 \times 10^1 & + & 7 \times 10^0 \\ & & 8 & & 6 & & 4 & & 7 \end{array}$$

Oben ausführliche, unten abgekürzte Schreibweise.

Beispiel für Zweier-(Dual-)System: $8647_{10} = 10000111000111_2$

$$\begin{array}{cccccccccccccccc} 1x2^{13} & + & 0x2^{12} & + & 0x2^{11} & + & 0x2^{10} & + & 0x2^9 & + & 1x2^8 & + & 1x2^7 & + & 1x2^6 & + & 0x2^5 & + & 0x2^4 & + & 0x2^3 & + & 1x2^2 & + & 1x2^1 & + & 1x2^0 \\ 1 & & 0 & & 0 & & 0 & & 0 & & 1 & & 1 & & 1 & & 0 & & 0 & & 0 & & 1 & & 1 & & 1 \end{array}$$

Oben ausführliche, unten abgekürzte, normale Schreibweise.

Die Umrechnung vom Dezimalsystem in das Dualsystem ist leicht zu bewerkstelligen und kann durch sukzessives Abziehen der Zweierpotenzen oder durch Feststellung, ob die jeweils halbierte Zahl gerade oder ungerade ist, erfolgen.

Beispiele

1386_{10} soll binärisiert werden.

1386	:	2	=	693	Rest	0	0	
693	:	2	=	346	"	1	1	
346	:	2	=	173	"	0	0	
173	:	2	=	86	"	1	1	
86	:	2	=	43	"	0	0	
43	:	2	=	21	"	1	1	$1386_{10} = 101\ 0110\ 1010_2$
21	:	2	=	10	"	1	1	
10	:	2	=	5	"	0	0	
5	:	2	=	2	"	1	1	
2	:	2	=	1	"	0	0	
1	:	2	=	0	"	1	1	Meistbedeutendes Bit

Anmerkung: Im Dualsystem steht wie im Dezimalsystem die "meistbedeutende Stelle" (MB-Stelle) links, die "wenigstbedeutende Stelle" (WB-Stelle) rechts.

Zahlen ≤ 1 haben entsprechend negative Exponenten. Das duale Komma wird hier im Gegensatz zum dezimalen Komma (,) als Dreieck (Δ) dargestellt.

$$0,625_{10} = 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} = \Delta 101_2$$

1 0 1

oder $0,625 \times 2 = \underline{1},25$ 1 MB-Stelle

$$0,25 \times 2 = \underline{0},5 \quad 0$$

$$0,5 \times 2 = \underline{1},0 \quad 1$$

$$47,25_{10} = 10\ 1111 \Delta 01_2$$

$$3,125_{10} = 11 \Delta 001_2$$

Die Addition erfolgt nach besonders einfachen Regeln:

$$1 + 0 = 1$$

$$0 + 1 = 1$$

$$0 + 0 = 0$$

$$1 + 1 = 0 \text{ plus Übertrag nach links}$$

$$= 10 \text{ (also } 10_2 = 2_{10})$$

Beispiele

5	101	6	110
+ 2	+ 10	+ 3	+ 11
<hr/>	<hr/>	<hr/>	<hr/>
7	111	9	1001

Subtraktion: Der LGP-30 subtrahiert, indem er das Komplement des Subtrahenden addiert und den letzten Übertrag vernachlässigt. Das Komplement wird gebildet, indem man Einsen durch Nullen und Nullen durch Einsen ersetzt und dann zur letzten Dualstelle ein Bit hinzuaddiert:

Dualzahl	1001	11000

Vertauschung	0110	00111
Addition einer Einheit +	1	1
	<hr/>	<hr/>
Komplement	0111	01000

Beispiel

$$\begin{array}{rcl}
 30 - 27 = 3 & 30 & 11110 \\
 & \text{Kompl.}(-27) & + 00101 \\
 & \hline
 & 3 & 1\ 00011 \quad \text{Ergebnis: } \underline{00011}
 \end{array}$$

Multiplikation und Division werden analog zum Dezimalsystem durch fortgesetzte Addition bzw. Subtraktion unter gleichzeitiger Stellenverschiebung ausgeführt.

Die Eingabe von Zahlen in den LGP-30 erfolgt in dezimaler Form. Durch ein entsprechendes Unterprogramm kann der LGP-30 sie selbst binärisieren.

2. Zahlendarstellung im LGP-30

Den Inhalt einer Speicherzelle des Hauptspeichers oder eines Registers bildet ein Wort von 32 Bit Länge, das entweder eine Zahl oder einen Befehl bedeuten kann. Die Zahlen sind rein dual dargestellt; das vorderste Bit (MB-Stelle) ist das Vorzeichen, das letzte (WB-Stelle) ist eine Leerstelle. Die Verarbeitung der Zahlen erfolgt serienmäßig, zuerst kommt die Leerstelle, zuletzt das Vorzeichen. Das Komma steht hinter dem Vorzeichen und wird als Maschinenkomma bezeichnet.

Bei einer positiven Zahl ist das Vorzeichenbit = 0,
bei einer negativen Zahl ist das Vorzeichenbit = 1,
und dahinter steht ihr Komplement.

Beispiele 0 Δ 110 1000

$$0 + 0,5 + 0,25 + 0,0625 = 0,8125$$

1 Δ 011 1000

$$-1 + 0 + 0,25 + 0,125 + 0,0625 = -1 + 0,4375 = -0,5625$$

Das Vorzeichen wird als -1 interpretiert, während alle übrigen Stellen die übliche Bedeutung haben.

Alle Zahlen im LGP-30 liegen demnach im Bereich zwischen -1 und $1 - 2^{-30}$, weil die WB-Stelle (ohne Leerstelle) die Wertigkeit 2^{-30} hat.

Die Grenzwerte sind:

	Zahl	Darstellung in der Maschine
größte pos.	$1 - 2^{-30}$	0 111 1111 1111 1111 1111 1111 1111 111
größte neg.	-1	1 000 0000 0000 0000 0000 0000 0000 000

Die kleinsten von Null verschiedenen Werte sind:

positiv	2^{-30}	0 000 0000 0000 0000 0000 0000 0000 001
negativ	-2^{-30}	1 111 1111 1111 1111 1111 1111 1111 111

Eine Dualzahl von 30 Stellen entspricht etwa einer Dezimalzahl von 9 Stellen.

3. Maschinenkomma und gedachtes Komma

Durch Multiplikation von Zahlen innerhalb dieses Bereiches mit Zweierpotenzen kann man den Zahlenbereich bedeutend erweitern; in der Praxis braucht man diese Multiplikation jedoch nicht wirklich auszuführen, da jede Multiplikation einer Dualzahl mit 2 einer Verschiebung des Kommas um eine Dualstelle nach rechts gleichkommt. Es genügt, sich die entsprechende Verschiebung der Kommastelle zu merken. Zu diesem Zweck bezeichnet man das neue Komma im Gegensatz zum Maschinenkomma (MK), welches hinter dem Vorzeichenbit steht, als "gedachtes Komma" (GK). Die Stellen zwischen GK und MK heißen Zwischenstellen. Ihre Anzahl wird mit q bezeichnet. Dieses q ist positiv, wenn das GK rechts von MK liegt und negativ, wenn das GK links von MK liegt. Fallen GK und MK zusammen, so ist $q = 0$.

Beispiel Hat man die Zahl 0,5 in der Maschine gespeichert ($0 \Delta 10$) und denkt sie sich mit $4 = 2^2$ multipliziert, so stellt sie die Zahl $0,5 \times 4 = 2$ dar ($010 \Delta 0$). Das GK ist dann um zwei Stellen gegenüber dem MK nach rechts verschoben, $q = 2$.

Zur eindeutigen Bestimmung einer Zahl in der Maschine gehört also nicht nur die Angabe ihrer Größe und ihres Vorzeichens, sondern auch die Angabe ihrer Zwischenstellen. Zur besseren Unterscheidung wird im folgenden MK durch Δ und GK durch ∇ dargestellt.

Beispiele

Masch. Darst.	Dual	Dezimal	Zwischenstellen q
0 Δ 110 1 ∇ 010	1101,01	13,25	4
∇ 00 Δ 110 1000	0,001101	0,203125	-2
1 Δ 001 1011 ∇	-1100101	-101	7

Bei gegebenem q ist $2^q - 2^{q-30}$ eine obere Schranke für die darstellbaren positiven Zahlen.

Masch. Darst.	Dual	Dezimal	Schranke	q_{\min}
0 Δ 111 1 ∇ 000	1111	15	$< 2^4$	4
∇ 00 Δ 110 0000	0,0011	0,1875	$< 2^{-2}$	-2

Es ist üblich, die Anzahl der Zwischenstellen wie folgt anzugeben:

Dezimale ∇ q etwa
 (lies: bei einem q von) 0,1875 ∇ -2

4. Rechenregeln

- a) Alle Operanden bei Addition bzw. Subtraktion müssen gleiches q haben, das Ergebnis steht selbstverständlich ebenfalls mit diesem q im Akkumulator.

Beispiel

$$\begin{array}{r}
 0 \Delta 001 1 \nabla 000 \quad 3 \nabla 4 \\
 + 0 \Delta 000 1 \nabla 000 \quad 1 \nabla 4 \\
 \hline
 0 \Delta 010 0 \nabla 000 \quad 4 \nabla 4
 \end{array}$$

- b) Bei der Multiplikation addieren sich die q's der beiden Operanden.

Gegeben: $a \nabla q_a$ und $b \nabla q_b$

Ergebnis: $c = (a \cdot b) \nabla q_c$ mit $q_a + q_b = q_c$

c) Bei der Division ergibt die Differenz zwischen dem q des Dividenden und dem q des Divisors das q des Quotienten

Bei $c = (a : b) @ q_c$ gilt $q_c = q_a - q_b$.

5. Hexadezimalzahlen

Will man eine Zahl so in den LGP-30 eingeben, daß sie ohne Umwandlung unmittelbar in ihrer dualen Form im Akkumulator steht, so muß man sie hexadezimal verschlüsseln, weil die Eingabe in Tetraden (Gruppen zu 4 Bit) erfolgt.

Um eine vorgegebene Dualzahl hexadezimal darzustellen, teilt man sie, rechts beginnend, in 8 Vierer-Gruppen ein und ordnet jeder Gruppe das entsprechende hexadezimale Schlüsselssymbol zu. Linksseitige Nullen können ausgelassen werden. Es gibt 16 hexadezimale Symbole, entsprechend den 16 Möglichkeiten der Darstellung einer Zahl durch 4 Bit.

Im folgenden sind als Symbole benutzt worden:

Dez.: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

Hex.: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, f, g, j, k, q, w

Beispiele

$$8647_{10} = 0010\ 0001\ 1100\ 0111_2 = 21j7_{16}$$

$$1386_{10} = \quad\quad\quad 0101\ 0110\ 1010_2 = 56f_{16}$$

hex

$$00000004 \hat{=} 1 @ 29$$

$$40000000 \hat{=} 1 @ 1$$

$$80000000 \hat{=} -1 @ 0$$

$$01q00000 \hat{=} 15 @ 10$$

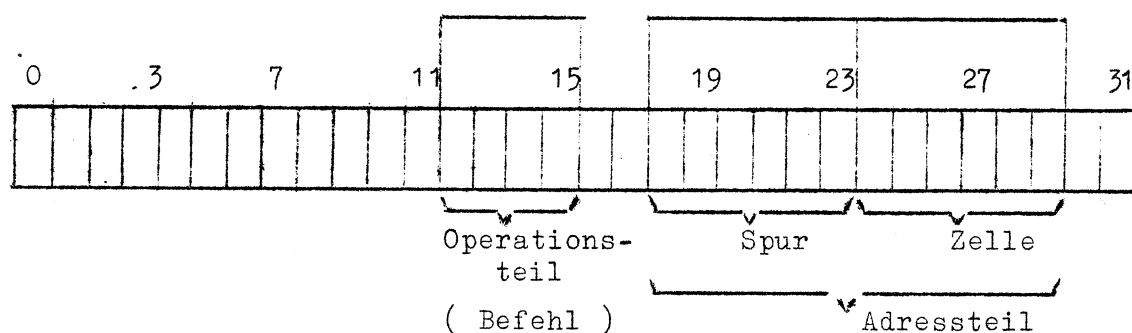
$$k0000000 \hat{=} -6 @ 4$$

Die Hexadezimalzahlen erlauben eine handliche Darstellung der langen Dualzahlen.

6. Befehlsdarstellung

Der LGP-30 hat 16 verschiedene Befehle. Er ist eine Ein-Adress-Maschine. Jeder Befehl besteht aus Operationsteil und Adressteil. Intern werden die Operationsteile durch vierstellige Dualzahlen verschlüsselt, die auf den festen Bitpositionen 12 - 15 eines Wortes stehen, während die Position 0 vom Vorzeichenbit eingenommen wird. Der Programmierer benutzt anstelle der dualen Schlüsselzahlen aber stets einen Buchstaben.

Die Adresse hat zwei Teile zu je 6 Dualstellen. Die eine Gruppe (Spurteil) benutzt die Bits 18 - 23, die zweite Gruppe (Zellenteil) die Bits 24 - 29. Die übrigen Stellen sind Leerstellen, die bei der Entschlüsselung nicht beachtet werden. Eine Ausnahme bildet der später zu besprechende T-Befehl. Es gibt 64 Spuren (0 bis 63) und in jeder Spur 64 Zellen (0 bis 63). Die Zählung der Zellennummern erfolgt Modulo 64, so daß ein



Übertrag im Zellenteil die Spurnummer um 1 erhöht. Erhöht man beispielsweise eine Adresse mit der Sektornummer 63 um 1, so erhält man einen Übertrag vom Zellenteil in den Spurteil, und der Zellenteil wird Null.

Beispiel	Adresse	2563		011001 111111
	Addition von	1	+	1
	Adresse	2600		011010 000000

Auf diese Weise wird der Übergang von einer Spur zur nächsten bewerkstelligt, soweit er nicht durch einen Sprungbefehl erfolgt.

Anmerkung: Die dezimal einzugebenden Adressen und die Befehlssymbole werden unmittelbar nach der Eingabe durch ein entsprechendes Unterprogramm in ihre dualen Äquivalente umgerechnet (s. Seite 24).

7. Die vier Phasen der Befehlsausführung

Die Ausführung eines Befehls erfolgt in vier Arbeitsgängen, Phasen genannt.

Phase 1: Es wird die Zelle im Hauptspeicher gesucht, deren Adresse im Befehlszähler (rZ) steht.

Phase 2: Der Inhalt der gefundenen Zelle wird als Befehl interpretiert und ins Befehlsregister (rB) geschrieben. Gleichzeitig wird der Inhalt des Zählers um 1 erhöht.

Phase 3: Es wird die Zelle des Hauptspeichers gesucht, deren Adresse im Adressteil des Befehlsregisters steht.

Phase 4: Der Operationsteil von rB wird entschlüsselt und der Befehl ausgeführt.

Nach Ausführung der Phase 4 beginnt die Maschine wieder in Phase 1, d.h. die Maschine sucht jetzt den Befehl in der folgenden Speicherzelle, da der Inhalt von rZ in der Phase 2 um 1 erhöht wurde. So fragt die Maschine eine Zelle nach der anderen ab. Kommt ein Stopbefehl, so hält sie in Phase 1 an; kommt ein Sprungbefehl, so wird durch die Ausführung des Sprungbefehls in Phase 4 der Inhalt von rZ abgeändert. Der nächste Zyklus unterbricht daher die Reihenfolge.

c) Codierung

Ein Elektronenrechner muß wie jede andere Maschine seine Arbeitsanweisung vom Menschen erhalten. Das geschieht durch Folgen von Einzelanweisungen, die jeweils ganz einfache Operationen des Automaten auslösen und Befehle heißen. Das Zusammenstellen einer Befehlsreihe, Programm genannt, erfordert folgende Arbeitsgänge:

I. Die Problemanalyse

- a) Aufstellen des Formelplans für das gegebene Problem nach Festlegung der Lösungsmethode.
- b) Zerlegung des Formelplans in einfache logische Schritte (Algorithmen).
- c) Aufstellen eines Ablaufdiagrammes.

II. Das Codieren

Umwandlung des Ablaufdiagramms in das Maschinenprogramm unter Benutzung der im LGP-30 verdrahteten Befehle.

Die folgende Darstellung beschränkt sich im wesentlichen auf das Codieren.

1. Befehlsliste des LGP-30

Im LGP-30 enthält jeder Befehl nur eine Adresse. Bei den arithmetischen Befehlen gibt diese Adresse den Speicherplatz des einen Operanden an. Der andere Operand muß zuvor in den Akkumulator gebracht werden. Das Ergebnis erscheint ebenfalls im Akkumulator. Bei der Beschreibung der Befehle des LGP-30 werden folgende Abkürzungen benutzt:

- m Spurnummer $0 \leq m \leq 63$
- n Zellennummer $0 \leq n \leq 63$
- mn Adresse einer Speicherzelle bzw. eines Befehls; durch mn ist eine Speicherzelle eindeutig definiert.
- (mn) Inhalt der Zelle, die durch mn definiert ist
- rA Akkumulator
- rZ Befehlszähler
- (rA) Inhalt des Akkumulators
- * Bei gesternten Symbolen ist der entsprechende Adressteil gemeint, etwa:
(rA)* Adressteil des Akkumulatorinhaltes.
- wird zum Inhalt von
- ⇒ springe
- (n) Zellennummer ist für die Befehlsfunktion ohne Bedeutung

Die Bedeutung der einzelnen Befehle ist im einzelnen die folgende:

- B mn Der Inhalt der Zelle mn wird in den Akkumulator gebracht. Der Inhalt der Zelle mn bleibt unverändert.
- A mn Der Inhalt der Zelle mn wird zum Akkumulatorinhalt addiert. Der Inhalt der Zelle mn bleibt unverändert.
- S mn Der Inhalt der Zelle mn wird vom Akkumulatorinhalt subtrahiert. Der Inhalt der Zelle mn bleibt unverändert.

Befehl

Änderung von

Operations- teil	Adress- teil	Dezimal- Code	Hex. Äquiv.	Dual- Code	Symbolische Gleichung	Bedeutung in Kurzform	RA Speicher	Aussehen auf der Anzeige	
B	mn	1	1	0001	$(mn) \rightarrow RA$	Bringen	ja nein		Arithme- tische Befehle
A	mn	14	q	1110	$(RA) + (mn) \rightarrow RA$	Addieren	ja nein		
S	mn	15	w	1111	$(RA) - (mn) \rightarrow RA$	Subtrahieren	ja nein		
M	mn	7	7	0111	$(RA) \cdot (mn) \rightarrow RA$	Mult.l.Hälfte	ja nein		
N	mn	6	6	0110	$(RA) : (mn) \rightarrow RA$	M-Multipl. r. Hälfte	ja nein		
D	mn	5	5	0101	$(RA) : (mn) \rightarrow RA$	Dividieren	ja nein		
E	mn	9	9	1001	$(RA) \wedge (mn) \rightarrow RA$ 2)	Extrahieren	ja nein		Speicher- befehle
H	mn	12	j	1100	$(RA) \rightarrow mn$	Speichern u. Behalten	nein ja		
O	mn	13	k	1101	$(RA) \rightarrow mn; 0 \rightarrow RA$	Speichern u. Löschen	ja ja		
U	mn	10	f	1010	$\Rightarrow mn$	Unbed.Sprung	nein nein		Organisa- tionsbe- fehle
T	mn	11	g	1011	$(RA) 0 \Rightarrow mn$	Bed. Sprung	nein nein		
Y	mn	2	2	0010	$(RA) * \rightarrow mn *$	Adr. ändern	nein ja		
R	mn	3	3	0011	$(RZ) * + 1 \rightarrow mn *$	Rückkehradres- se speichern	nein ja		
I	0000	4	4	0100		Eingabe	ja ¹⁾ nein		
P	m(n)	8	8	1000		Drucken	nein nein		
Z	m(n)	0	0	0000		Halt	nein nein		

1) Der Akkumulatorinhalt wird nur geändert, wenn wirklich eingegeben wurde.

2) Logisches Produkt

- M mn Der Inhalt des Akkumulators wird mit dem Inhalt der Zelle mn multipliziert. Das Vorzeichen und die ersten 30 Dualstellen des Ergebnisses werden im Akkumulator gespeichert. Der Inhalt von mn bleibt unverändert.
- N mn Wie M, jedoch enthält der Akkumulator die letzten 31 Stellen des Ergebnisses, wobei in der Vorzeichenstelle nicht das Vorzeichen, sondern die erste Wertstelle der zweiten Ergebnishälfte steht.
- D mn Der Inhalt des Akkumulators wird durch den Inhalt der Zelle mn dividiert und der Quotient im Akkumulator gespeichert. Der Inhalt der Zelle mn bleibt unverändert.
- H mn Der Inhalt des Akkumulators wird in die Zelle mn gespeichert. Der Inhalt des Akkumulators bleibt unverändert.
- C mn Der Inhalt des Akkumulators wird in die Zelle mn gespeichert. Der Inhalt des Akkumulators wird durch Nullen ersetzt.
- Y mn Der Adressteil des Akkumulators wird in den Adressteil der Zelle mn schreiben. Der Inhalt des Akkumulators und alle anderen Stellen der Zelle mn bleiben unverändert.
- E mn Bilde das logische Produkt aus dem Inhalt des Akkumulators und dem Inhalt der Zelle mn; alle Stellen des Akkumulators, denen eine Null in der Zelle mn entspricht, werden durch Nullen ersetzt. Die übrigen Stellen des Akkumulators und der Inhalt der Zelle mn bleiben unverändert.
- U mn Ersetze den Adressteil des Befehlszählers durch mn. Der nächste Befehl, den die Maschine ausführt, wird aus Zelle mn genommen. *an Befehlzeiger springen*
- T mn Ist der Akkumulatorinhalt negativ, so springe nach mn (ersetze den Adressteil des Befehlszählers durch mn). Ist der Akkumulatorinhalt positiv, so führe den nächsten Befehl aus. *an Befehlzeiger springen*
- R mn Der um 1 vergrößerte Inhalt des Adressteils des Befehlszählers wird zum Adressteil der Zelle mn. Alle übrigen Stellen der Zelle mn und der Inhalt des Zählregisters bleiben unverändert. *an Befehlzeiger springen*

Z mn Die Maschine stoppt in Phase 1 des nächsten Befehls.
Ist die Spurnummer eine der Zahlen 4, 8, 16 oder 32,
und ist der entsprechende Schalter "Kein Halt" des
Bedienungsfeldes gedrückt, so stoppt die Maschine nicht,
sondern geht zum nächsten Befehl über.

TO000 Dieser Befehl macht die Maschine bereit, den Akkumulator
von außen zu füllen. Das Auffüllen kann von Hand oder
von Lochstreifen aus erfolgen. Jeder Typenanschlag bzw.
jede Lochkombination des Streifens bringt 4 - oder wenn
6 Bit-Eingabe gedrückt ist - 6 Bits in die letzten Stellen
des Akkumulators. Zuvor wird der Akkumulator um 4 bzw. 6
Stellen nach links geschiftet.

P mn Die Schreibmaschine des LGP-30 schreibt ein Zeichen oder
führt eine Funktion aus (Wagenrücklauf, Tabulator, Umschal-
ten usw.). Die Spurnummer m bestimmt, welches Zeichen ge-
schrieben oder welche Funktion ausgeführt wird.

2. Einfache Befehlskombinationen

a) Umspeichern von Worten: Die Zahl $a = 4@7$ sei in 3214 und
 $b = 6@7$ in 1701 gespeichert. Die Summe soll in 5000 gespeichert
werden.

Die Befehlsreihe für diese Aufgabe lautet:

Befehl	Plangleichung	Duale Darstellung des Akkumulatorinhalts	
B 3214	$(3214) \rightarrow rA$	$0_{\Delta}0000100$	$4@7$
A1701	$(1701) + (rA) \rightarrow rA$	$0_{\Delta}0000110$	$6@7$
C5000	$(rA) \rightarrow 5000$	$0_{\Delta}0001010$	$10@7$
	$0 \rightarrow rA$		

b) Extrahieren: Der E-Befehl erlaubt, beliebige Teile aus einem
Wort herauszuschneiden. Der Adressteil des E-Befehls bezieht sich
auf die Zelle, in welcher der sogenannte Extraktor steht. Dieser
Extraktor wirkt auf den Akkumulatorinhalt wie eine Maske.

Beispiel In Zelle 2163 steht die Maske $0_{\Delta}00111110000$, die in der
Lage ist, die Bits 3 - 7 aus einem Wort - hier in Zelle
2207 - herauszuschneiden.

B2207	$0_{\Delta}101\ 1011\ 0011$	(rA)	} nach Befehlsausführung
E2163	$0_{\Delta}001\ 1011\ 0000$	(rA)	

- c) **Schiften** durch M- und D-Befehl: In Zelle 4028 sei eine $1@3$ und in 4154 die Zahl $a = 11@8$ gespeichert. In Zelle 1000 soll $a@11$, in 1001 soll $a@5$ gespeichert werden.

Befehl	Speicherinhalt	
B4154	0 _Δ 000 0101 1 _▲ 000 0	$11@8$
M4028		$(11@8) \cdot (1@3) = 11@11$
C1000	0 _Δ 000 0000 1011 _▲ 0	$11@11$
B4154		$11@8$
D4028		$(11@8) : (1@3) = 11@5$
C1001	0 _Δ 010 11 _▲ 00 0000 0	$11@5$

Anmerkung: Bei Verwendung des D-Befehls ist auf Überlauf zu achten. Weiterhin erlaubt der **D**-Befehl nicht, eine Ziffer in die Vorzeichenstelle zu schiften.

- d) **Linksschiften** durch N-Multiplikation: Multipliziert man zwei Zahlen a und b mit dem N-Befehl, so gilt für die Anzahl der Zwischenstellen des Produktes: $q_c = q_a + q_b - 31$. Ist $b = 1$ bei $q_b = n$, so bedeutet die N-Multiplikation eine Verschiebung des Kommas um $31 - n$ Stellen nach links.

Beispiele

Die Zelle 5324 enthalte $1@28$. In 3607 sei $a = 9@8$ gespeichert.

B3607	0 _Δ 00001001 _▲ 0	$9@8$
N5324	0 _Δ 01001 _▲ 0000	$9@5$

In 3608 sei $b = 13@5$ gespeichert.

B3608	0 _Δ 01101 _▲ 0000	$13@5$
N5324	1 _Δ 01 _▲ 0000000	Dieses Beispiel zeigt, daß Stellen des Ergebnisses verschwinden können. Ein Überlaufhalt erfolgt nicht. Eine 1 kann in die Vorzeichenstelle geschifft werden.

- e) **Programmverzweigungen**; U- und T-Befehl: Der Programmierer hat die Möglichkeit, mit einem Sprungbefehl die Reihenfolge der Befehle zu unterbrechen und an einer beliebigen anderen Stelle im Programm fortzufahren. Bei Verwendung des T-Befehls kann man den Sprung noch vom Ergebnis der vorangegangenen Rechenoperation abhängig machen.

Beispiel

Gegeben seien die Zahlen a und b . Ist $a \geq b$, so soll $a = x$ gesetzt werden; ist $a < b$, so soll $b = x$ gesetzt werden. In beiden Fällen ist $y = x^2$ zu berechnen.

Speicherliste:

6000	a	(a) 8
6001	b	(a) 8
6002	x	(a) 8
6003	y	(a) 16

Zelle	Befehl	Plangleichung	
0500	B6000	$a \rightarrow rA$	(a) 8
0501	S6001	$a - b \rightarrow rA$	(a) 8
0502	T0505	$(rA) < 0 \Rightarrow 0505$	
0503	B6000	$(rA) \geq 0; a \rightarrow rA$	(a) 8
0504	U0506	$\Rightarrow 0506$	
0505	B0601		
0506	H6002	$(rA) \rightarrow 6002; (6002) = x$	
0507	M6002	$x \cdot x \rightarrow rA$	(a) 16
0508	C6003	$x^2 \rightarrow 6003$	
0509	Z0000	Stop	

Ist die Differenz von a und b kleiner Null, so erfolgt in Zelle 0502 ein Sprung nach Zelle 0505, und die Befehlsreihe ab dieser Zelle wird ausgeführt. Ist die Differenz aber größer oder gleich Null, so wird nach dem T-Befehl der Befehl in Zelle 0503 ausgeführt. Der sich daran anschließende Befehl in 0504 bewirkt immer einen Sprung nach 0506. Der H-Befehl in 0506 dient dazu, den gleichen Operanden sowohl im Hauptspeicher als auch im rA zu haben.

- f) Der Test auf Null: Gegeben seien zwei Zahlen a und b . Wenn ihre Differenz gleich Null ist, soll die Maschine anhalten. Sonst soll vorher das Ergebnis nach 0405 gespeichert werden.

Speicherliste:

1500	a	(a) 5
1501	b	(a) 5
1502	1	(a) 30
1503	a - b	

Zelle	Befehl	Plangleichung
0300	B1500	$a \rightarrow rA$
0301	S1501	$a - b \rightarrow rA$
0302	H1503	$a - b \rightarrow 1503$
0303	T0306	$(rA) < 0 \Rightarrow 0306$
0304	S1502	$(rA) - 1030 \rightarrow rA$
0305	T0308	$(rA) < 0 \Rightarrow 0308$
0306	B1503	$a - b \rightarrow rA$
0307	C0405	$a - b \rightarrow 0405$
0308	Z0000	Stop

Die Befehlskombination TST bewirkt hier den Test auf Null. Ist der erste T-Befehl erfolgreich (ein T-Befehl wird erfolgreich genannt, wenn ein Sprung erfolgt), so ist die Differenz $a - b$ kleiner als Null, also $a < b$. Im anderen Fall weiß man, daß die Differenz größer oder gleich Null ist. Zieht man von der Differenz jetzt 1030 ab (die kleinste in der Maschine darstellbare positive Zahl) und der T-Befehl in 0305 ist erfolgreich, so ist die Differenz $a - b$ sicher Null; denn wenn der Akkumulatorinhalt nach dem S-Befehl in 0304 Null ist, wird der T-Befehl in 0305 nicht erfolgreich.

- g) Der Y-Befehl, Schleifen: Mit Hilfe des Y-Befehls kann man ein Programm von der Maschine selbst abändern lassen. Wie das geschieht, zeigt folgendes Beispiel:

Ab Zelle 4000 seien Werte a_i , ab Zelle 4020 Werte b_i fortlaufend gespeichert. Die Summen der gleichindizierten a_i und b_i sollen ab 4040 fortlaufend gespeichert werden. Der Index i möge von 1 - 10 laufen.

Zelle	Befehl	erster Durchlauf	letzter Durchlauf
1000	B4000	$a_1 \rightarrow rA$	$a_{10} \rightarrow rA$
1001	A4020	$a_1 + b_1 \rightarrow rA$	$a_{10} + b_{10} \rightarrow rA$
1002	C4040	$a_1 + b_1 \rightarrow 4040$	$a_{10} + b_{10} \rightarrow 4049$
1003	B1000	$B4000 \rightarrow rA$	$B4009 \rightarrow rA$
1004	A1013	$B4001 \rightarrow rA$	$B4010 \rightarrow rA$
1005	Y1000	$4001 \rightarrow 1000*$	$4010 \rightarrow 1000*$
1006	A1014	$B4021 \rightarrow rA$	$B4030 \rightarrow rA$
1007	Y1001	$4021 \rightarrow 1001*$	$4030 \rightarrow 1001*$
1008	A1014	$B4041 \rightarrow rA$	$B4050 \rightarrow rA$
1009	Y1002	$4041 \rightarrow 1002*$	$4050 \rightarrow 1002*$
1010	S1015	$B4041 - B4050 = -9 \rightarrow rA$	$B4050 - B4050 = 0$

Zelle	Befehl	erster Durchlauf	letzter Durchlauf
1011	T1000	$\Rightarrow 1000$	$\Rightarrow 1012$
1012	Z0000		Stop
1013	1@29		
1014	20@29		
1015	B4050		

Im obigen Programm wird indiziert durch Addition von entsprechenden Konstanten zum Adressteil des entsprechenden Befehls. Konstanten, die zum Adressteil eines Befehls hinzuaddiert oder davon subtrahiert werden, müssen @29 stehen, da die Adresse @29 steht.

Der Y-Befehl in Zelle 1005 speichert den Adressteil des Akkumulators (beim ersten Durchlauf 4001) in den Adressteil des B-Befehls in Zelle 1000. Analoges geschieht in den Zellen 1007 und 1009, so daß die ersten 3 Befehle nach jedem Durchlauf eine um 1 erhöhte Adresse haben. Der Programmteil von 1000 bis 1011 wird 10mal durchlaufen, er bildet eine Schleife. Der Befehl in 1015 ist am Programmablauf nicht aktiv beteiligt. Solche Befehle werden "konstant" genannt, da sie nur zum Adressenrechnen bzw. -prüfen dienen.

Anmerkung: Bei der Subtraktion von konstanten Befehlen zu Testzwecken ist darauf zu achten, daß der Operationsteil in der Differenz Null wird.

- h) Speichern der Rückkehradresse: In den Zellen von 0300 bis 0314 stehe ein Teilprogramm; in Zelle 0314 ein U-Befehl. An einer bestimmten Stelle eines anderen Programms soll das Programm in Spur 3 benutzt werden. Als Beispiel diene folgende Befehlsfolge:

1000	B1105	(1105) \rightarrow rA
1001	C0544	(rA) \rightarrow 0544; 0 \rightarrow rA
1002	R0314	1003 + 1 \rightarrow 0314*
1003	U0300	\Rightarrow 0300
1004	B0631	

Der R-Befehl in 1002 setzt den um 1 erhöhten Inhalt des Adressteils des Befehlszählers (d.h. die um 1 erhöhte Adresse der nächsten Zelle) in den Adressteil des U-Befehls in Zelle 0314. Dann erfolgt durch den U-Befehl in 1003 ein Sprung an den Anfang des Programms in Spur 3. Kommt die Maschine zum U-Befehl in Zelle 0314, so steht jetzt in dessen Adressteil 1004, und es erfolgt ein Rücksprung zum B-Befehl in 1004, wo das Hauptprogramm fortgesetzt wird.

Anmerkung: Die R/U-Kombination benutzt man, um sogenannte Unterprogramme aufzurufen. Solche Unterprogramme gibt es für viele Zwecke; sie können an jeder beliebigen Stelle eines Hauptprogramms eingebaut werden und ersparen das wiederholte Codieren häufig wiederkehrender Teilprobleme (Funktionen o. ä.).

i) Der Eingabe-Befehl: Zur Eingabe eines Wortes über den Flexowriter (von Hand oder vom Band) sind zwei Befehle nötig.

- I. P0000 mache den Eingabeteil des Flexowriters funktionsfähig.
a) Ist "Eingabe von Hand" am Flexowriter gedrückt, so leuchtet eine Lampe auf, und es kann über das Tastenfeld eingegeben werden.
b) Im anderen Falle wird der Leser in Funktion gesetzt.
- II. I0000 übernehme die gelesenen Zeichen in den Akkumulator.

Beispiel

Es ist eine Zahl einzulesen und nach 6300 zu speichern. Die entsprechende Befehlsreihe lautet:

0300	C6300	0 → rA
0301	P0000	} eine Zahl einlesen
0302	I0000	
0303	C6300	Zahl nach 6300 speichern
0304	Z0000	Stop

Der Befehl in Zelle 0300 löscht den rA, bevor eingelesen wird. Besteht die einzulesende Zahl nämlich aus weniger als 8 Ziffern, so füllt sie den Akkumulator nicht ganz aus, und in den ersten Stellen könnte noch ein Teil des vorherigen Inhalts stehen.

j) Besonderheiten von P- und I-Befehl: Die Funktionen von P- und I-Befehl sind abhängig vom Spurteil ihrer Adressen. Der P-Befehl führt entsprechend dem Zahlensymbol in seinem Spurteil eine Flexowriter-Funktion aus und dient daher der Ausgabe. Die Drucksymbole sind aus einem Kärtchen ersichtlich. Während der Ausführung eines P-Befehls kann der LGP-30 weiterrechnen. Durch einen sogenannten Zwischenstop (ZW) ist nur dafür zu sorgen, daß vor Beendigung der Flexowriter-Operation kein neuer P-Befehl aufgerufen wird. Nach Beendigung der Druckoperation erhält der LGP-30 vom Flexowriter automatisch ein Startsignal.

Beispiel

Es ist LGP-30 zu drucken.

0000	P0800	Umschalten auf Großbuchstaben
0001	Z0000	ZW
0002	P0600	Drucke L
0003	Z0000	ZW

0004	P4600	Drucke G
0005	Z0000	ZW
0006	P3300	Drucke P
0007	Z0000	ZW
0008	P0400	Umschalten auf Kleinbuchstaben
0009	Z0000	ZW
0010	P0700	Drucke "-"
0011	Z0000	ZW
0012	P1400	Drucke 3
0013	Z0000	ZW
0014	P0200	Drucke 0
0015	Z0000	ZW
0016	Z0000	Stop

Anmerkung: Zur Ausgabe von Zahlen und alphanumerischen Symbolen in größerem Umfang benutzt man vorhandene Unterprogramme.

Der Spurteil des I-Befehls

Beim Aufruf des I-Befehls wird der Akkumulatorinhalt um 4 bzw. 6 Stellen nach links verschoben, bevor ein Symbol vom Flexowriter eingelesen wird. In diese Lücke werden die ersten 4 bzw. alle 6 Bits des Spurteils des I-Befehls eingeschoben.

Beispiel

Der Akkumulator enthalte das hexadezimale Wort 42Q37806.

Durch I0000 werde 214 eingelesen. Dann enthält der Akkumulator nach dem Einlesen (4 Bit-Eingabe) das Wort 78060214. Hätte man mit I4800 eingelesen, so enthielte der Akkumulator das Wort 7806J214. Das J ist das hexadezimale Äquivalent der ersten 4 Bits der Spurnummer 48.

3. Lineares Programm

Zu berechnen ist P nach der Formel:

$$P = a_2 x^2 + a_1 x + a_0$$

Umgeschriebene Formel (Horner-Schema):

$$P = (a_2 x + a_1) x + a_0$$

Speicherliste:

Zelle	Dualzahl	
5000	a_0 111 Δ	7 @ 10
5001	a_1 11 Δ	3 @ 10
5002	a_2 10 Δ	2 @ 10
5003	x 101 Δ 1	5,5 @ 3
5004	1	1 @ 3

Programm:

Zelle	Befehl	Plangleichung	q
0300	B5002	$a_2 \rightarrow rA$	10
0301	M5003	$a_2 x \rightarrow rA$	13
0302	D5004	$a_2 x \rightarrow rA$	10
0303	A5001	$a_2 x + a_1 \rightarrow rA$	10
0304	M5003	$(a_2 x + a_1) x \rightarrow rA$	13
0305	D5004	$(a_2 x + a_1) x \rightarrow rA$	10
0306	A5000	$P \rightarrow rA$	10
0307	C5005	$P \rightarrow 5005$	10
0308	Z0000	Stop	

4. Programmverzweigungen

Es sind die drei Zahlen a, b, c in den Zellen 5907, 5908 und 5909 gespeichert. Die größte ist auszusortieren und in 5910 zu speichern.

Zelle	Befehl	Plangleichung
1500	B5907	$a \rightarrow rA$
1501	S5908	$a - b \rightarrow rA$
1502	T1509	$a - b < 0 \Rightarrow (1)$
1503	B5907	$a \rightarrow rA$
1504	S5909	$a - c \rightarrow rA$
1505	T1515	$a - c < 0 \Rightarrow (2)$
1506	B5907	$a \rightarrow rA$
1507	H5910	$a \rightarrow 5910$
1508	U....	Ausgang
1509	B5908 (1)	$b \rightarrow rA$
1510	S5909	$b - c \rightarrow rA$
1511	T1515	$b - c < 0 \Rightarrow (2)$
1512	B5908	$b \rightarrow rA$
1513	H5910	$b \rightarrow 5910$
1514	U....	Ausgang

Zelle	Befehl	Plangleichung
1515	B5909 (2)	$c \rightarrow rA$
1516	H5910	$c \rightarrow 5910$
1517	U....	Ausgang

5. Zyklisches Programm

Aus 30 Zahlen a_i , die ab Zelle 4000 fortlaufend gespeichert sind, sollen 30 Zahlen b_i nach der Vorschrift

$$b_i = \frac{a_i^2 + 2}{a_i(a_i + 4)}$$

berechnet und ab Zelle 6300 fortlaufend gespeichert werden. Die a_i seien positive Zahlen @ 10.

Zelle	Befehl	Plangleichung	Bemerkungen
0300	B0323	$1 \rightarrow i$	Zu Beginn stets alle veränderl. Adressen (=Indexzähler) auf Anfangswerte, damit Programm immer wieder benutzt werden kann.
0301	Y0304		
0302	A0325		
0303	Y0313		
0304	B4000 (1)	$a_i \rightarrow rA$	@ 10
0305	H4539	$a_i \rightarrow 4539$	4539 ist ein Arbeitsspeicher
0306	A0326	$a_i + 4 \rightarrow rA$	10
0307	M4539	$(a_i + 4)a_i \rightarrow rA$	20
0308	C4540	$(a_i + 4)a_i \rightarrow Z$	20
0309	B4539	$a_i \rightarrow rA$	10
0310	M4539	$a_i \rightarrow rA$	20
0311	A0327	$a_i + 2 \rightarrow rA$	20
0312	D4540	$b_i \rightarrow rA$	0
0313	C6300	$b_i \rightarrow 6300$	
0314	B0304	$i + 1 \rightarrow i$	Der Index i wird um 1 erhöht; die Adressen der Befehle in 0304 und 0313 dienen als Zähler
0315	A0324		
0316	Y0304		
0317	A0325		
0318	Y0313	$i < 30 \Rightarrow (1)$	Prüfung, ob 30 Werte verarbeitet sind.
0319	S0322		
0320	T0304		
0321	Z0000		
		Stop	

Zelle	Befehl	Plangleichung
0322	B6330	} Vergleichskonstanten
0323	B4000	
0324	4	1 @ 29
0325	1700	23 @ 23
0326	800000	4 @ 10
0327	1000	2 @ 20

6) Eingabeplan 10.4

Ein Programmeingabeplan muß zwischen Befehlen und Konstanten unterscheiden können, da er die Adresse der Befehle dualisieren muß, die hexadezimalen Konstanten jedoch nicht. Mittels des Eingabeplans 10.4 kann man weiterhin den Speicherplatz eines Programmes willkürlich vorschreiben, indem man die Anfangsadresse des einzulesenden Programmes vorgibt. Ferner werden alle Adressen, die sich auf Befehle und Konstanten dieses Programmes beziehen, umgerechnet (relative Adressen werden zu absoluten Adressen). Adressen, die sich auf feste Speicherplätze beziehen, werden nicht umgerechnet, weil sie bereits absolute Adressen darstellen. Die entsprechenden Anweisungen werden dem Eingabeplan durch die folgenden Schlüsselwörter angegeben:

000Bxxxx (Befehl). Das Wort wird als Befehl gedeutet. Die Adresse xxxx wird dezimal eingegeben und durch 10.4 dualisiert. B steht für eines der 16 Befehlssymbole. Steht an der ersten Stelle des Wortes eine 8 so enthält die Vorzeichenstelle des Befehls nach der Umwandlung eine 1.

,00000NN (hex.Eingabe). Dieses Wort ist ein reines Schlüsselwort und wird nicht mitgespeichert. Es bewirkt, daß die folgenden NN Wörter ($0 \leq NN \leq 63$), die in hexadezimaler Form eingegeben werden müssen, unverändert als Dualzahlen gespeichert werden.

;000xxxx (Anfangsadresse zum Speichern). Die folgenden Befehle oder hexadezimalen Zahlen werden ab Zelle xxxx fortlaufend gespeichert. Diese Folge kann nur durch ein neues ;000xxxx-Symbol unterbrochen werden.

/000xxxx (Adressenmodifikator). xxxx wird zu allen abzuändernden, relativen Adressen addiert. Befehle mit absoluten Adressen müssen durch ein x vor dem Befehlssymbol gekennzeichnet werden. Der Modifikator xxxx kann nur durch ein neues Codewort verändert werden, da er von 10.4 gespeichert wird.

.000xxxx (Zwischenstop und Sprung). Der LGP-30 hält auf KH 32 an. Ein Startsignal bewirkt einen Sprung zur Speicherzelle xxxx.

00Hxxxx (Befehl ausführen). Der Befehl Hxxxx wird in das Befehlsregister gebracht, der LGP-30 hält an. Bei Start wird der Befehl ausgeführt; vorher kann der Akkumulator von Hand gefüllt werden.

10.4 erlaubt es, relativ zu programmieren. So kann man etwa die Anfangsadresse eines Programmes als 0000 annehmen und das Programm später durch ;000xxxx und /000xxxx auf geeignete Spuren schreiben.

Anmerkung: Beim Lochen eines Programmes ist darauf zu achten, daß nach jedem Befehl, jedem Codewort und jeder Konstanten ein Stopcode steht.

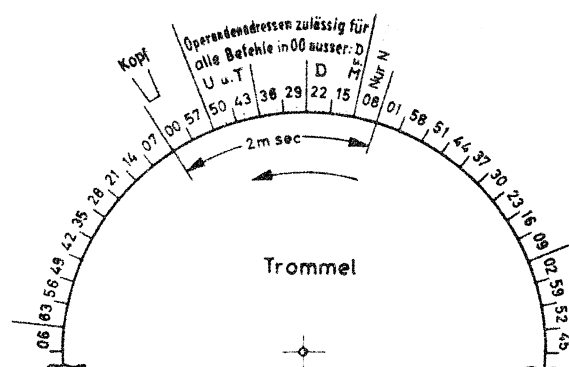
Beispiel

Das Programm auf Seite 18 ist folgendermaßen zu lochen:

```
;0001000'  
/0001000'  
xB4000'xA4020'xC4040'A0013'Y0000'  
A0014'Y0001'A0014'Y0002'S0015'T0000'xZ0000'  
,0000002'4'50'  
xB4050'  
.0001000'
```

7. Optimalisierung

Die Zeit, welche zwischen dem Lesen und Schreiben von Operanden und dem Kommando dazu verstreicht, ist variabel. Diese sogenannte Zugriffszeit, die abhängig ist von der Stellung der Trommel zum Lese-Schreibkopf, läßt sich auf ein Minimum reduzieren. Zu diesem Zweck sind die Sektoren auf einer Spur in besonderer Weise angeordnet. Zwischen zwei Stellen, deren Nummern sich um 1 unterscheiden, befinden sich 8 andere Zellen (s. Skizze).



Zellen-Numerierung einer Spur zum Zwecke der Optimalisierung

Ist etwa in Zelle 00 der Befehl B0015 gespeichert, so wird der Operand in 0015 sechs Wortzeiten nach dem Lesen des Befehls gefunden. Nach einer weiteren Wortzeit kann der Befehl in Zelle 01 gelesen werden. Zwischen dem Aufruf des B-Befehls in 00 und dem nächsten Befehl in 01 vergehen also 9 Wortzeiten. Das ist auch der Fall, wenn die Adresse des B-Befehls einen der Sektoren 50, 43, 36, 29, 22 enthält. In diesen 9 Wortzeiten (= 2 ms) wird ein Befehl komplett ausgeführt. Es sind beide Zugriffszeiten (Suchen des Befehls und des Operanden) enthalten. Liegt die Operandenadresse nicht im zulässigen Bereich, wird stets eine ganze Trommelumdrehung mehr benötigt. Die zulässigen Operandenadressen hängen vom Befehl ab (s. Skizze). Die Spurnummern spielen bei der Optimalisierung keine Rolle.

Praxis der Optimalisierung

Will man wissen, welche Adressen für einen B-Befehl in Sektor 37 optimal sind, so suche man in der folgenden Tabelle die Zahl 37. Die zweite bis siebente Zahl nach 37, von oben nach unten gelesen, gibt die optimalen Sektoren an. Es sind 23, 16, 09, 02, 59 und 52.

00	01	02	03	04	05	06	07
↓ 57	58	59	60	61	62	63	00
↘ 50	51	52	53	54	55	56	57
43	44	45	46	47	48	49	.
36	37	38	39	40	41	42	.
29	30	31	32	33	34	35	.
22	23	24	25	26	27	28	.
15	16	17	18	19	20	21	
08	09	10	11	12	13	14	

Die Anzahl der optimalen Sektoren ist nicht bei allen Befehlen die gleiche. Die folgende Tabelle gibt zu jedem Befehl die optimalen Sektoren an.

D	2 - 5
M	2 - 6
B, H, C, E, A, S, R, Y	2 - 7
Z, P, I, N	2 - 8
U, T	4 ff

Beispiel

Für einen D-Befehl in Zelle 24 sind die Sektoren 10, 3, 60, 53 optimal.

Durch die Optimalisierung wächst der Arbeitsaufwand für ein Programm stark an. Wirtschaftlich ist die Optimalisierung daher nur für Programme, die sehr oft benutzt werden. Insbesondere sollte man nur häufig zu durchlaufende Schleifen eines Programmes optimieren. Unterprogramme sind in den meisten Fällen optimal.

D. Bedienungsanweisungen

1. Eingabe des Ladeprogramms und des Programms 10.4

Das Ladeprogramm (Bootstrap) ist ein Hilfsprogramm zum Einlesen der Programm-Eingabe-Routine 10.4. Beide Programme befinden sich auf einem Lochstreifen.

1. Lochstreifen in Lesevorrichtung des Flexowriters einlegen.
2. Schalter VERBINDUNG des Flexowriters auf "AUS" stellen.
3. Taste LESEN START des Flexowriters betätigen.
(Es wird das Kennwort "manual" ausgedruckt)
4. Schalter EINGABE VON HAND des LGP-30 drücken.
5. Taste LESEN START des Flexowriters betätigen.
6. Taste BEFEHL EINGEBEN des LGP-30 betätigen.
7. Taste LESEN START des Flexowriters betätigen.
8. Schalter EINZELOPERATION des LGP-30 drücken.
9. Taste BEFEHL AUSFÜHREN des LGP-30 betätigen.
10. Anweisung 4 - 9 noch fünf Mal wiederholen.
11. Schalter EINGABE VON HAND des LGP-30 drücken.
12. Taste LESEN START des Flexowriters betätigen.

Auf dem Flexowriter wird "normal" geschrieben.

13. Für das Einlesen von 10.4 mittels Flexowriter gelten folgende Anweisungen:

14. Schalter EINZELOPERATION des LGP-30 drücken.
15. Schalter NORMAL des LGP-30 drücken.
16. Schalter VERBINDUNG des Flexowriters auf "EIN" stellen.
17. Taste START des LGP-30 betätigen.
(Der Lochstreifen 10.4 wird eingelesen)
18. Nach dem Einlesen hält der Flexowriter automatisch an.
19. Schalter EINZELOPERATION des LGP-30 drücken.
20. Schalter EINGABE VON HAND des LGP-30 drücken.

13. Für das Einlesen von 10.4 mittels Fotoleser gelten folgende Anweisungen

14. Lochstreifen aus der Lesevorrichtung des Flexowriters herausnehmen - Lochstreifen-Halteklappe am Flexowriter nicht vergessen zurückklappen - und mit der bedruckten Lochstreifen-seite nach unten so in den Fotoleser einlegen, daß nur 10.4 eingelesen werden kann.
15. Schalter EINZELOPERATION des LGP-30 drücken.
16. Schalter NORMAL des LGP-30 drücken.
17. Schalter EINGABE des Fotolesers auf "ABTASTER" stellen.
18. Taste START des LGP-30 betätigen.
(Der Lochstreifen 10.4 wird eingelesen)
19. Schalter EINZELOPERATION des LGP-30 drücken.
20. Schalter EINGABE VON HAND des LGP-30 drücken.
21. Schalter EINGABE des Fotolesers auf "SCHREIBMASCHINE" stellen.

2. Eingabe von Programm-Lochstreifen

Eingabe dezimaler Lochstreifen.

a.) Eingabe über Flexowriter.

1. Lochstreifen in Lesevorrichtung des Flexowriters einlegen.
 2. Schalter VERBINDUNG des Flexowriters auf "EIN" stellen.
 3. Schalter EINGABE VON HAND des Flexowriters drücken.
 4. Schalter EINZELOPERATION des LGP-30 drücken.
 5. Taste ZÄHLER LÖSCHEN des LGP-30 drücken.
 6. Schalter NORMAL des LGP-30 drücken
 7. Taste START des LGP-30 drücken.
(Lichtfeld am Flexowriter leuchtet auf)
 8. Anfangsadresse (Startfill) ;000XXXX , z.B. ;0004711 mit Flexowriter schreiben.
 9. Taste RECHNER START des Flexowriters drücken.
(Anfangsadresse ist eingegeben)
 10. Adressenmodifikator (Modifier) /000XXXX , z.B. /0004711 mit Flexowriter schreiben.
 11. Taste RECHNER START des Flexowriters drücken.
 12. Schalter EINGABE VON HAND des Flexowriters hochdrücken.
(Programm wird eingelesen)
 13. Schalter EINZELOPERATION des LGP-30 drücken.
 14. Schalter EINGABE VON HAND des LGP-30 drücken.
- } Aufruf von 10.4

b.) Eingabe über Fotoleser.

1. Lochstreifen in Lesevorrichtung des Fotolesers einlegen.
Punkt 2. bis 10. wie unter a.)
11. Schalter EINGABE des Fotolesers auf "ABTASTER" stellen.
12. Taste RECHNER START des Flexowriters drücken.
(Programm wird eingelesen)
13. Schalter EINZELOPERATION des LGP-30 drücken.
14. Schalter EINGABE VON HAND des LGP-30 drücken.
15. Schalter EINGABE des Fotolesers auf "SCHREIBMASCHINE" stellen.

WICHTIG: Punkt 11 muß unbedingt vor Punkt 12 erfolgen.

Eingabe hexadezimaler Lochstreifen.

Bei Eingabe hexadezimaler Lochstreifen entfällt unter

- a.) Punkt 8. bis 11. (incl.)
- b.) Punkt 8. bis 10. (incl.)

3. Ein- und Ausschaltvorschrift für LGP-30, Flexowriter, Fotoleser und Schnellstanzer

Die Einschaltung der Ziffern-Rechenanlage LGP-30 ist nach folgender Anweisung durchzuführen:

1. Druckschalter EINGABE VON HAND des LGP-30 drücken.
2. Druckschalter BETRIEB des LGP-30 drücken.
3. Hauptschalter EIN des LGP-30 drücken.

Folgende Lampen leuchten nach Betätigung des Hauptschalters EIN am Tastenfeld des LGP-30 auf:

- a.) unmittelbar: PAUSE
EINGABE VON HAND, ZÄHLER LÖSCHEN, BEFEHL EINGEBEN.
- b.) nach ca. 50 s.: BETRIEB
EINGABE VON HAND, ZÄHLER LÖSCHEN, BEFEHL EINGEBEN.
- c.) nach ca. 100 s.: BEREIT
EINGABE VON HAND, ZÄHLER LÖSCHEN, BEFEHL EINGEBEN.

Wenn die Lampe BEREIT aufleuchtet, ist der Rechner betriebsbereit.

4. Netzschalter des Flexowriters auf "EIN" stellen.
5. Bei Bedarf Schalter ABTASTER EIN des Fotolesers drücken.
(Taste leuchtet in eingeschaltetem Zustand auf)
6. Nach etwa 1 Minute ABTASTER HALT drücken.
7. Bei Bedarf Schalter LOCHER EIN des Schnellstanzers drücken.
(Taste leuchtet in eingeschaltetem Zustand auf)

Die Abschaltung der Anlage geschieht nach folgender Anweisung:

1. Druckschalter LOCHER EIN des Schnellstanzers drücken.
(Taste leuchtet in ausgeschaltetem Zustand nicht auf)
2. Druckschalter ABTASTER EIN des Fotolesers drücken.
(Taste leuchtet in ausgeschaltetem Zustand nicht auf)
3. Netzschalter des Flexowriters auf "AUS" stellen.
4. Druckschalter EINZELOPERATION des LGP-30 drücken.
5. Druckschalter EINGABE VON HAND des LGP-30 drücken.
6. Hauptschalter AUS des LGP-30 drücken.

WICHTIG: Netzschalter des Flexowriters muß unbedingt auf AUS (Punkt 3), bevor der LGP-30 abgeschaltet wird.

E. Programmieren im Gleitkomma

Programmiert man ein Problem im Festkomma, so hat man eine Reihe Vorüberlegungen über die möglichen Zahlenbereiche und eine Anzahl Prüfungen sowie Skalierungen im Programm einzubauen. Das ist umständlich und zeitraubend. Viel bequemer ist es daher, Zahlen in halblogarithmischer Darstellung, das sogenannte Gleitkomma, zu verwenden. Da der LGP-30 an sich aber nur im Festkomma arbeitet, müssen interpretierende Programme vorgesehen werden, die das Arbeiten im Gleitkomma in gleicher Weise wie bei einer Anlage mit festverdrahtetem Gleitkomma ermöglichen. Es gibt eine ganze Reihe solcher Programmiersysteme, die sich im wesentlichen durch die Art der Zahlendarstellung unterscheiden.

Das Gleitkomma-Unterprogramm 24.0 ist eins davon und zeichnet sich durch besondere Flexibilität aus, da es mehr und noch wirkungsvollere Befehle als der Elementarcode des LGP-30 und zusätzliche Register aufweist.

11. Zahlendarstellung und Register

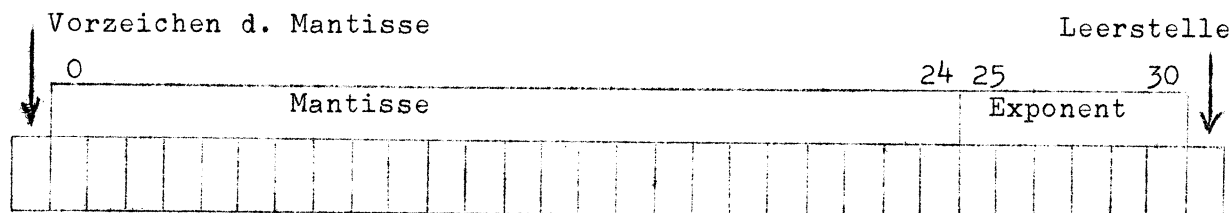
Jede Zahl A kann man schreiben als $A = a \cdot 2^x$

Ist $A \neq 0$ und verlangt man $0,5 \leq |a| < 1$, so ist die Zerlegung eindeutig. Die Zahl a heißt Mantisse und x heißt Exponent.

Beispiele

Dez.	Mant.	Exp.	hex
8	= 0,500	$\cdot 2^4$	40000008
-10	= -0,625	$\cdot 2^4$	G0000008
0,25	= 0,500	$\cdot 2^{-1}$	4000007Q
-0,1	= -0,800	$\cdot 2^{-3}$	99999F7F

Im Interpretiersystem 24.0 werden a und x in einem Wort gespeichert; die Mantisse a mit 24 Bit bei q = 0 und der Exponent x mit 5 Bit und Vorzeichen bei q = 30. Negative Mantissen und Exponenten werden durch ihr Komplement dargestellt.



Die zusätzlichen Register bestehen jeweils aus 2 Wörtern des Hauptspeichers für Mantisse und Exponent:

- a) Akkumulator
- b) Multiplikationsregister für den Multiplikator
- c) Adressenregister für die Umrechnung von Adressen

Der Inhalt eines Registers wird nur durch solche Befehle zerstört, die dieses Register neu belegen.

2. Befehlsliste

a) Arithmetische Befehle

- Bxxxx Die Zahl in Zelle xxxx wird im Akkumulator gespeichert.
- Axxxx Die Zahl in Zelle xxxx wird zum Inhalt von rA addiert, und die Summe wird in rA gespeichert.
- Sxxxx Die Zahl in Zelle xxxx wird von rA subtrahiert und die Differenz in rA gespeichert.
- Dxxxx Der Inhalt von rA wird durch die Zahl in Zelle xxxx dividiert und der Quotient in rA gespeichert.
- Pxxxx Die Zahl in Zelle xxxx wird im Multiplikationsregister (rM) gespeichert.
- Mxxxx Die Zahl in rM wird mit der Zahl in Zelle xxxx multipliziert und das Produkt in rA gespeichert.
- Nxxxx Die Zahl in rM wird mit der Zahl in Zelle xxxx multipliziert und zum Inhalt von rA addiert. Das Ergebnis wird in rA gespeichert.
- D000y Der Inhalt von rA wird durch 2^y dividiert und der Quotient in rA gespeichert ($0 \leq y \leq 9$).
- M000y Der Inhalt von rA wird mit 2^y multipliziert und das Ergebnis in rA gespeichert ($0 \leq y \leq 9$).
- Hxxxx Der Inhalt von rA wird in Zelle xxxx gespeichert; rA bleibt unverändert.
- Cxxxx Der Inhalt von rA wird in Zelle xxxx gespeichert; rA wird mit Nullen gefüllt.

b) Sprungbefehle

- Uxxxx Der nächste Pseudobefehl wird aus Zelle xxxx entnommen (\Rightarrow nach xxxx).
- Txxxx Ist die Zahl in rA negativ, so erfolgt ein Sprung nach xxxx, anderenfalls wird der nächste Befehl interpretiert.
- 800Txxxx Es erfolgt ein Sprung nach xxxx, wenn der Inhalt von rA negativ ist oder die Sprungtaste des LGP-30 gedrückt ist. Sonst wird der nächste Befehl interpretiert.

Die Pseudobefehle können nicht zum Verlassen des Interpretiersystems benutzt werden.

c) Adressenumrechnung

- Exxxx** Der Adressteil des Adressenregisters wird durch den Adressteil der Zelle xxxx ersetzt.
- Ixxxx** Der Inhalt des Adressenregisters wird um die Zahl xxxx erhöht. Man kann die Adresse im Adressenregister um uuuu vermindern, wenn man anstelle von xxxx das Komplement von uuuu einsetzt.
- Yxxxx** Der Adressteil der Zelle xxxx wird durch den Adressteil des Adressenregisters ersetzt.
- Zxxxx** Die Adresse xxxx wird vom Inhalt des Adressenregisters subtrahiert. Ist die Differenz ungleich Null, so wird als nächster der Befehl in der folgenden Zelle interpretiert. Ist die Differenz gleich Null, so wird der folgende Befehl übersprungen.

d) Hilfsbefehle

- Rxxxx** Die Adresse der Zelle, die den R-Befehl enthält, wird um 2 vergrößert und im Adressteil der Zelle xxxx gespeichert.
- U0000** Die Inhalte des Registers rM und des Adressenregisters werden ausgetauscht.
- B0000** Das Vorzeichen der Zahl im Akkumulator wird zu + gemacht.
- T0000** Das Vorzeichen der Zahl im Akkumulator wird zu - gemacht.
- Y0000** Das Vorzeichen der Zahl im Akkumulator wird umgekehrt.
- Z0000** Stop, wenn Schalter "Kein Halt 16" nicht gedrückt ist.
- E0000** Das Interpretiersystem wird verlassen. Der folgende Befehl wird als Elementarbefehl interpretiert.

e) Ein- und Ausgabebefehle

- I0000** Es werden dezimale Zahlen von Band gelesen, entsprechend einem Code-Wort auf dem Lochstreifen (s. Abschn.3.S.33).
- P0000** Der Inhalt von rA wird ausgedruckt. Während des Druckes wird er nicht zerstört (Format s. Abschn.3. S.34).

f) Aufruf von Funktionen

RO000 Die Quadratwurzel der Zahl in rA wird berechnet und in rA gespeichert.
SO000 Der Sinus der Zahl in rA wird berechnet und in rA gespeichert. Das Argument muß im Bogenmaß gegeben sein.
CO000 Der Cosinus der Zahl in rA wird berechnet und in rA gespeichert. Das Argument muß im Bogenmaß gegeben sein.
AO000 Der Arcustangens der Zahl in rA wird berechnet und in rA gespeichert. Das Ergebnis wird im Bogenmaß gespeichert.
NO000 Der natürliche Logarithmus der Zahl in rA wird berechnet und in rA gespeichert.
HO000 Die e-Funktion für die Zahl im rA als Argument wird berechnet und in rA gespeichert.

3. Ein- und Ausgabenformat

a) Eingabe

Einzugebende Daten werden im allgemeinen vom Lochstreifen gelesen. Das Band besteht aus drei Gruppen: Schlüsselwort, Daten und Schlußzeichen.

Das Schlüsselwort hat die Form \pm pp AAAA. Dabei bedeutet AAAA die Anfangsadresse der Zellen, in denen die folgenden Zahlen gespeichert werden sollen. Die Zahl pp gibt die Kommalage an und muß der Bedingung genügen:

$$- 3 \leq pp \leq 15 .$$

Die Daten (Zahlen) bestehen aus Vorzeichen und 7 dezimalen Ziffern. Alle Zahlen, die zu einem Schlüssel gehören, haben die gleiche Kommalage. Sie werden dualisiert und mit dem Exponent in Aufeinanderfolgenden Zellen beginnend bei AAAA gespeichert. Das Schlüsselwort wird nicht gespeichert.

Das Schlußzeichen heißt - 000 0000 und beschließt eine Datengruppe. Es wird nicht gespeichert. Dem Schlußzeichen kann ein neues Schlüsselwort zur Einleitung einer neuen Gruppe oder ein Nullwort zur Beendigung des Einlesens folgen. Das Nullwort hinter dem letzten Schlußzeichen bewirkt das Verlassen des Einlese-Unterprogramms. Danach wird der auf IO000 folgende Befehl ausgeführt.

Beispiel

Die Zahlen 1073,247 - 0,601 3,040
sollen in den Zellen 2400, 2401,.... und
die Zahlen 2,13 4,24 1,00 -26,01 14,26
in den Zellen 5000 , 5001, gespeichert werden.
Der Datenstreifen hat dann folgendes Aussehen:

+032400'	1073247'
	-0000601'
	3040'
	-00000000'
+025000'	213'
	424'
	100'
	-0002601'
	1426'
	-00000000'
	00000000'

Anstelle des Nullwortes genügt ein Stopcode.

b) Ausgabe

Die Zahlen werden in zwei Gruppen geschrieben: Mantisse mit 7 Dezimalziffern und Zehnerexponent mit 2 Dezimalziffern. Die Vorzeichen stehen jeweils hinter der Zahl. Nach jedem Exponent erfolgt ein Tabulatorsprung.

Beispiel

Ausgabeformat		Zahlengröße
.5060000-	02	-50,600000
.1472000	01-	0,014720
.3141592	01	3,141592
.6300000-	00	-0,630000

4. Programmbeispiel

Die Summe der Zahlen $a = 3$ und $b = 5$ soll gedruckt werden. Danach ist ein Wagenrücklauf auszuführen und zu stoppen. Der Plan 24.0 beginne bei 0300.

00	xR0300		}	Elementarbefehle zum Aufruf des Systems 24.0
01	xU0300			
02	B0009	} a + b → rA	}	Pseudobefehle im System 24.0
03	A0010			
04	xP0000	(rA) drucken		
05	xE0000	Ausgang		
06	xP1600	WR	}	Elementarbefehle
07	xZ0000	ZH		
08	xZ0000	Stop		
09	60000004	a = 3 = 0,75 · 2 ²		
10	50000006	b = 5 = 0,625 · 2 ³		

5. Unterpläne

Die Programme für die arithmetischen und logischen Befehle, für die Hilfsbefehle und für die Befehle zur Adressenmodifikation sowie für die Wurzel sind im Grundsystem enthalten. Werden die übrigen Befehle benutzt, so muß der entsprechende Unterplan gespeichert sein.

Die Adressen der Unterpläne sind durch die Adresse des Grundsystems bestimmt.

Programm	Eingabewort = Anfangsadresse	Modifikator	Spuren
24.0 Grundsystem einschl. Wurzel	Lo	Lo	10
11.6/12.6 Ein- u. Ausgabe	Lo + 1000	Lo + 1000	6
14.1.Sin.-Cos.	Lo + 1600	Lo	2 1/2
16.2.Arcustangens	Lo + 1832	Lo	1 1/2
18.1.Logarithmus	Lo + 2000	Lo	1
17.1.Exponential- funktion	Lo + 2100	Lo	2

Die Spur 63 wird als Zwischenspeicher benutzt. Daher ist Lo so zu wählen, daß kein Unterplan diese Spur belegt. Als Anfangsadresse von 24.0 wählt man im allgemeinen 4000.

Folgende Zellen von Spur 63 werden nicht benutzt:

10, 15, 16, 23, 27, 29, 34, 36, 40, 47,
48, 49, 50, 52, 56, 57, 58, 60, 63.

6) Programmierte Stops (Fehler)

Das System enthält verschiedene Stopbefehle, die die Maschine bei fehlerhaften Operationen anhalten. Die folgende Tabelle gibt die Stelle, den letzten Befehl und den Grund für den Stop an.

Stelle	Befehl	Grund
Lo + 0654	Z0000	Normaler Halt. Durch Betätigen des Schalters "START" wird die Rechnung fortgesetzt.
Lo + 0557	$\left\{ \begin{array}{l} \text{Hxxxx} \\ \text{Cxxxx} \end{array} \right\}$ oder	Der Exponent ist zu groß. Die Adresse des H- oder C-Befehls steht im Elementar-Akkumulator.
Lo + 0557	R0000	Der Akkumulatorinhalt ist negativ. Die Adresse des R-Befehls befindet sich im Elementar-Akkumulator.
Lo + 1152	I0000	Eines der Eingabedaten hat einen zu großen Exponenten. Ein Start bewirkt die Speicherung einer Null anstelle dieser Zahl. Danach wird das nächste Wort gelesen.
Lo + 0612	Dxxxx	Division durch Null oder eine nichtnormalisierte Zahl. Die Rechnung muß abgebrochen werden.
Lo + 2005	N0000	Der Akkumulatorinhalt ist negativ oder Null. Bei Start wird die Rechnung mit Null im ra fortgesetzt.
Lo + 2028 oder Lo + 2030	$\left\{ \begin{array}{l} \\ \\ \end{array} \right\}$ N0000	$\left\{ \begin{array}{l} \\ \\ \end{array} \right\}$ Der Exponent des Akkumulator ist zu groß oder zu klein. Die Rechnung muß abgebrochen werden.

Bemerkungen: Das System kann verlassen oder wieder aufgerufen werden, ohne die Inhalte der 3 Register zu zerstören.

Der Exponent einer Zahl in ra, die gespeichert werden soll, muß kleiner als +32 sein. Ist er kleiner als -31, so wird die Zahl durch eine Null ersetzt. Die Adresse Lo muß den Sektor 00 enthalten. Anderenfalls sind viele Befehle des Systems nicht optimal.

Alle Pseudobefehle mit der Adresse 0000 haben spezielle Bedeutungen. Keine von ihnen bezieht sich auf die Zelle 0000. Ähnliches gilt für die Schifftbefehle D000y und M000y; bei normaler Division oder Multiplikation dürfen die Zellen 0000 bis 0009 nicht benutzt werden.

F. Zusammenfassung der technischen Daten des LGP-30

Arbeitsweise

Einadressen-Serienmaschine mit festem
Komma, innerem Programm und Dualsystem.
Zahlenbereich: -1 bis $1 - 2^{-30}$

Operationszeiten

Zugriffszeit:	2 msec. minimal	
	15 msec. maximal	
Additionszeit:	0,23 msec	} ohne Zugriffs- zeit
Subtraktionszeit:	0,23 msec.	
Multiplikationszeit:	15 msec.	
Divisionszeit:	15 msec.	

Maße

Breite 66 cm, Höhe 84 cm, Länge 118 cm
(ohne Schreibmaschine), Gewicht 365 kg

Trommel

Kapazität: 4096 Worte
Wortlänge: 31 Dualstellen einschl. Vorzeichen
Taktfrequenz: 140 kHz
Trommeldrehzahl: 4000 U/Min.
Anzahl der Spuren im Hauptspeicher: 64
Taktspuren: 4
Umlaufspeicher: 3
Schreibdichte: 4 Bit je mm
Kopfabstand zur Trommel: ca. 25 μ
Rotor: 165 mm \varnothing , 178 mm lang

Anschlußwert

220 V \pm 10 % - 20 %, 50 Hz, einphasig mit
Schutzerde, 1,5 KW
Der LGP-30 kann ohne weiteres an jede Schuko-
Steckdose angeschlossen werden, die den VDE-
Vorschriften entspricht. Sonstige Installations-
vorrichtungen sind nicht erforderlich.

G. Schlußbemerkung

Die vorliegende Programmieranleitung stellt eine kurzgefaßte Einführung in die Programmierung des LGP-30 dar. Sie erhebt keinen Anspruch auf Vollständigkeit, sollte aber im Zusammenhang mit den ausführlichen Druckschriften SF 014 und SF 015 zu verstehen sein, so daß jedes als Ablaufdiagramm vorliegende Problem aufgrund des Studiums dieser Anleitung ohne weiteres programmiert werden kann.

Auf andere Programmiersysteme, wie z.B. erweitertes Gleitkomma in 2 oder $1\frac{1}{3}$ -Wort-Darstellung, Gleitkomma mit doppelter Genauigkeit oder den Formelübersetzer ACT I (Compiler) sei hier nur kurz hingewiesen. Für sie existieren gesonderte spezielle Benutzungsanweisungen und Beschreibungen, die leicht zu verstehen sind. Ebenso ist jedes einzelne Unterprogramm aus der ungewöhnlich reichhaltigen Programmbibliothek mit einer speziellen Beschreibung versehen, aus der Zweck, Funktion, Eingabe- und Ausgabeformat, Aufruffolge, Zeitbedarf, Speicherbelegung usw. eindeutig hervorgehen.

Die Unterprogramm-Handbücher stellen für den LGP-30-Benutzer zwar ein unentbehrliches Hilfsmittel und Nachschlagewerk dar, sind aber zum Erlernen der Programmierung des LGP-30 nicht erforderlich.

Eurocomp GmbH · 495 Minden 2 · Schillerstraße 72

EC 13-022

Printed in Germany

4/63