Standard-Compiler ACT V +K

I. Speicherbelegung während der Compilierphase

Der Standard-Compiler ACT V liegt als hexadezimaler Programmstreifen vor. Dieser Streifen enthält auch alle zum Compilieren notwendigen Tabellen und belegt die Spuren 3200 bis 6163. Ferner werden die Spur 62 für die Konstanten des Objektprogramms und die Spur 63 als Zwischenspeicher benutzt. Da das Objektprogramm immer in 0332 beginnt, kann die Programmeingabe mit J1-10.0 (von 0000 bis 0328) erfolgen.

II. Compilieren mit ACT V

1.) Programmstreifen des Standard-Compilers ACT V mit J1-10.0 nach Vorschrift Nr. 9 der Bedienungsanleitung für den LGP-21 einlesen.

2.) Streifen des Quellenprogramms in Lesevorrichtung des Flexowriters legen.

3.) "Eingabe von Hand" am Flexowriter drücken.

4.) Aufruf J1-10.0 :

   a) Schalter am Rechner auf "Step" stellen
   b) "Füllen/Löschen" am Rechner drücken
   c) Schalter am Rechner auf "Normal" stellen
   d) "Start" am Rechner drücken.

5.) Eingabe des Schlüsselwortes .0004000

6.) "Rechner Start" am Flexowriter drücken

7.) "Eingabe von Hand" am Flexowriter lösen

8.) "Start" am Rechner drücken.

Das Quellenprogramm wird in die Maschinensprache übersetzt und im Rechner gespeichert. Fehler im Quellenprogramm werden während des Compilierens angezeigt; ausführliche Fehlerbesprechung in Kapitel IV. Anschließend an das Compilieren werden die Anweisungs- und Variablenlisten ausgegeben, wenn PS 8 nicht gedrückt ist. Bei gedrückter PS 8-Taste hält der Rechner nach dem Übersetzen an und erst nach Drücken von "Start" am Rechner werden die Anweisungs- und Variablenlisten ausgeschrieben. Außerdem wenn PS 4 gedrückt ist: Halt nach jeder Anweisung beim Compilieren. Ist PS 4 gelöst: Kein Stop.

Jedes Objektprogramm beginnt immer in 0332 und seine Endadresse wird immer in der Anweisungsliste durch S000 angezeigt.

III. Ausstanzen von Objektprogrammen

Zum Ausstanzen sind 3 Programme vorhanden, die das Objektprogramm als hexadezimalen Streifen mit Prüfsumme nach jeder Spur ausgeben.

1.) "Normal Punch" belegt die Speicherplätze von 2900 bis 3163. Damit können Objektprogramme ausgestanzt werden, die maximal bis 2863 gehen.

Der Programmstreifen liegt in 2 Fassungen vor. Bei der 1. Version erfolgt die Ausgabe über Flexowriter, bei der 2. Version über Schnellstanzer.

Bedienung:

a) Hex. Programmstreifen "Normal Punch" für Ausgabe über Flexowriter oder Schnellstanzer mit J1-10.0 nach Vorschrift Nr. 9 der Bedienungsanleitung für den LGP-21 einlesen.

b) "Lochen Ein" am Flexowriter oder "Stanzer Ein" am Zusatzgerät drücken.

c) Ein Stück "Bandlauf" ausführen.

d) "Start" am Rechner drücken.

Das Objektprogramm und alle notwendigen Konstanten werden hexadezimal ausgestanzt. Ist das "Normal Punch" Programm noch im Rechner gespeichert, dann entfällt a).
b) und c) werden genauso ausgeführt und für d) sind folgende Operationen auszuführen:

"Eingabe von Hand" am Flexowriter drücken

Aufruf J1-10.0

.0002900 über Flexowriter eingeben

"Rechner Start" am Flexowriter drücken

"Start" am Rechner drücken

Objektprogramm wird ausgestanzt.

2.) "Emergency Punch" belegt die Speicherplätze 3200 - 3463. Damit können Objektprogramme ausgestanzt werden, die auch die Spuren 2900 bis 3163 benutzen. Durch dieses Stanzprogramm wird allerdings ein Teil des Compilerprogrammes zerstört.

"Emergency Punch" liegt ebenfalls in 2 Versionen vor (Ausgabe über Flexowriter; Ausgabe über Schnellstanzer).

Bedienung:

a) Einlesen von "Emergency Punch" für Ausgabe über Flexowriter oder Schnellstanzer mit J1-10.0. Siehe Vorschrift Nr. 9 in Bedienungsanleitung LGP-21.

b) "Lochen Ein" am Flexowriter oder "Stanzer Ein" am Zusatzgerät drücken.

c) Ein Stück "Bandlauf" ausführen.

d) "Start" am Rechner drücken.

Das Objektprogramm wird ausgestanzt.

3.) "Partial Punch" belegt die Speicherplätze 2800 bis 3163 und dient zum Ausstanzen von Teilprogrammen (Prozeduren). Die Bedienung bei "Partial Punch" entspricht der Bedienung von "Normal Punch". Nur bei einem erneuten Aufruf muß der Sprung nach .0002800 erfolgen.

IV. Fehleranzeigen beim Compilieren und ihre Korrekturen

Der ACT V - Compiler besitzt eine umfangreiche Fehlerdiagnostik. Es sind 3 Fehlergruppen zu unterscheiden, die zu verschiedenen Zeiten während der Compilierphase angezeigt werden und dementsprechend unterschiedlich zu korrigieren sind:

Gruppe 1:

Error - 10	Bei der Marke Snnn ist nnn > 255.
" - 11	Die Marke wurde bereits für eine vorhergehende Anweisung benutzt.
" - 12	Eine Variable tritt als Index auf, die vorher im Quellenprogramm nicht vorkommt.
" - 13	Bei einer Gleitkomma-Konstanten ist der zweite Teil keine Zahl bzw. der Exponent ist zu groß, oder in einer DIM-Anweisung steht hinter der Feldvariablen keine Zahl.
" - 15	Die Anzahl der öffnenden Klammern ist nicht gleich der Anzahl der schließenden.
" - 16	Bei einer indizierten Variablen stehen 3 Indizes bzw. es fehlt ein Operator zwischen zwei Operanden.
" - 21	Unzulässiger Operator, z.B. cr, tab, index usw.
" - 24	Hinter einer schließenden Klammer steht kein Operator.
" - 25	Es fehlt ein rechter Operand.
" - 26	Es fehlt ein linker Operand.
" - 27	Es fehlt ein linker Operand. Fehlende Operanden werden nicht in allen Fällen erkannt!
" - 31	In einer MACHN-Anweisung steht ein falscher Maschinenbefehl.

Die obigen Fehler werden während des Einlesens und der dabei erfolgenden sequentiellen Übersetzung der fehlerhaften Anweisung entdeckt. Wenn auch ein in diesem Falle falsches Objektprogrammstück entwickelt wurde, so können die Fehler doch sofort korrigiert werden. Dazu ist nach dem Fehlerstop die fehlerhafte Anweisung zu verbessern und nach dem Drücken der Starttaste erneut einzulesen. Die laufende Objektprogrammadresse wird auf den Wert zu Beginn der Anweisung zurückgesetzt, so daß das falsche Objektprogrammstück jetzt überschrieben wird. Beginnt die Anweisung mit einer (korrekten!) Marke, so ist diese bereits in der Tabelle der markierten Anweisungen enthalten. Die Marke darf daher nicht erneut eingelesen werden, da sonst sofort Error - 11 angezeigt wird. Tritt in einer fehlerhaften Anweisung irrtümlich eine falsche, bislang noch nicht definierte Variable auf, so wird diese während des sequentiellen Übersetzens in die Variablentabelle aufgenommen. Wird die korrigierte Anweisung dann ohne diese Variable nochmals compiliert, so bleibt die falsche Variable natürlich in der Variablentabelle und wird auch später beim Ausdruck der Speicherbelegungstabelle ausgewiesen. Das wird im allgemeinen aber nicht stören.

Gruppe 2:

Error - 22	Das Objektprogramm überschneidet sich mit dem Variablenbereich.
" - 23	Die Tabelle der Variablenbezeichnungen bzw. die Tabelle der Konstanten ist voll.

Diese beiden Fehler werden beim Compilieren angezeigt, sobald die betreffende Situation eintritt. Abhilfe wird im allgemeinen nur durch Abänderung des Quellenprogrammes und Neu-Compilierung möglich sein.

Gruppe 3:

Error - 17	Im Quellenprogramm existiert ein Sprung auf eine Marke, die nicht definiert ist. D.h. diese Marke tritt bei keiner Anweisung als Kennzeichen auf.

Dieser Fehler kann erst nach Abschluß der Compilierung des gesamten Quellenprogrammes entdeckt werden.

Wenn die fehlerhafte Sprunganweisung z.B. lautet: USE' Smmm'', wobei Smmm nicht existiert, so erfolgt die Fehleranzeige nach Beendigung der Compilierung vor dem Ausdruck der Speicherbelegungsliste in folgender Form:

Error - 17 Smmm ttss

Anschließend wird die Speicherbelegungsliste in der üblichen Form ausgegeben.
In diesem Fall ist das Objektprogramm richtig entwickelt, lediglich in Zelle ttss steht ein Sprungbefehl mit falscher Adresse.

Der Fehler läßt sich verbessern, wenn man vor dem Ausstanzen des Objektprogrammes von Hand den Sprung mit der richtigen Adresse in die Zelle ttss einfügt.

V. Speicherbelegung während der Rechenphase

Das Objektprogramm beginnt immer in 0332 und kann sich maximal bis 3163 erstrecken. Zum Rechnen müssen außerdem unbedingt die Standard-Unterprogramme gespeichert werden. Die Standard-Unterprogramme belegen die Speicherplätze 4200 bis 6163 und enthalten die Unterprogramme für Ein- und Ausgabe und alle Gleit- und Festkomma-Operationen.

Bereiche und Variable werden von 4163 abwärts reserviert und gespeichert, in der Reihenfolge wie es nach dem Compilieren in der Variablenliste ausgedruckt worden ist. Dabei muß bei Objektprogrammen mit Prozeduren beachtet werden, daß für die lokalen Variablen der Prozeduren zwar Speicherplätze von 4163 abwärts reserviert werden, aber nicht in der Variablenliste erscheinen.

Werden im Objektprogramm außerdem Funktionsunterprogramme benötigt, so müssen diese auf die freien Plätze zwischen Objektprogramm und Variablenspeicher eingegeben werden.

Die Funktionsunterprogramme liegen einzeln als relativ hexadezimale Streifen vor und können auf beliebige Spuren zwischen Programm und Variablen gelegt werden. Es ist allerdings auch möglich, durch dim' subs' n'' als erste Anweisung im Quellenprogramm, die Funktionsunterprogramme von 4163 abwärts zu legen. Dabei ist n die Anzahl der von den Funktionsunterprogrammen benötigten Speicherplätze (Platzbedarf siehe Liste der Funktionsunterprogramme). In diesem Fall werden die Bereiche und Variablen abwärts an die Funktionsunterprogramme gelegt.

Die Verknüpfungen (linkages) zwischen Funktionsunterprogrammen und Objektprogramm liegen in verschiedenen Spuren der Standardunterprogramme, und zwar auf folgenden Plätzen:

Funktion	Speicherplatz	Befehl (nicht modifiziert)
sqrt	4203	U0000
ln	4210	U0137
log	4303	U0107
exp	4503	U0200
pwr	4513	U0021
artan	4810	U0000
sin	4710	U0000
cos	4603	U0100

Die Speicherplätze für die linkages bleiben beim Einlesen der Standard-Unterprogramme frei. Erst beim Einlesen des jeweiligen Funktionsunterprogrammes wird der modifizierte Sprungbefehl auf den festgelegten Platz gespeichert. Soll z.B. das Wurzelunterprogramm (sqrt) ab Spur 3900 gespeichert werden, dann muß vor der Programmeingabe der Modifikator /0003900 eingegeben werden. Da am Ende des Lochstreifens für das Wurzelunterprogramm ;0004203' U0000' steht, wird bedingt durch den Modifikator nach 4203 automatisch der richtige Sprungbefehl, nämlich U3900, gespeichert.

Die Länge der einzelnen Funktionsunterprogramme ergibt sich aus folgender Liste:

sqrt	64 Sektoren	= 1 Spur
ln, log, exp, pwr	256 "	= 4 Spuren
artan	128 "	= 2 Spuren
sin, cos	192 "	= 3 Spuren

Wie aus der Liste zu ersehen ist, sind sin, cos und ln, log, exp, pwr jeweils zu einem Unterprogramm zusammengefaßt.

VI. Rechnen mit dem Objektprogramm

1.) Einlesen des hex. Objektprogramms mit J1-10.0 nach Vorschrift Nr. 9 der Bedienungsanleitung für den LGP-21.

Ist das Objektprogramm gerade compiliert worden, entfällt 1.

2.) Einlesen des hex. Programmstreifens der Standard-Unterprogramme mit J1-10.0 nach Vorschrift Nr. 9 der Bedienungsanleitung für den LGP-21.

3.) Nur wenn im Objektprogramm Funktionsunterprogramme benötigt werden, müssen als erstes die Speicherplätze für die jeweiligen Funktionsunterprogramme festgelegt werden. Daraus ergeben sich dann die Anfangsadressen der entsprechenden Funktionsunterprogramme.

Die Funktionsunterprogramme selbst müssen, da sie als relativ hexadezimaler Streifen vorliegen, mit J1-10.0 nach Vorschrift Nr. 12 der Bedienungsanleitung für den LGP-21 eingelesen werden. Der dabei einzugebende Adressenmodifikator entspricht der jeweils festgelegten Anfangsadresse der einzelnen Funktionsunterprogramme.

4.) Sind Objektprogramm und alle notwendigen Unterprogramme gespeichert, kann mit der eigentlichen Rechnung begonnen werden. Dazu sind folgende Operationen nötig:

a) Datenstreifen in Lesestation des Flexowriters einlegen.

b) "Eingabe von Hand" am Flexowriter drücken.

c) Aufruf J1-10.0.

d) Schreiben des Schlüsselwortes .000XXXX (XXXX = Anfangsadresse des Objektprogramms, meistens ist es 0332).

e) "Rechner Start" am Flexowriter drücken.

f) "Eingabe von Hand" am Flexowriter lösen.

g) "Start" am Rechner drücken.

Die Daten werden über Flexowriter eingelesen und die Rechnung ausgeführt. Die Ausgabe erfolgt normalerweise über Flexowriter. Soll die Dateneingabe über Schnelleser oder die Datenausgabe über Schnellstanzer erfolgen, dann muß zusätzlich zu den Standard-Unterprogrammen ein Korrekturstreifen eingelesen werden, der die I- und P-Befehle für Schnelleser oder Schnellstanzer abwandelt. Danach werden die gleichen Operationen ausgeführt wie unter 4 a)- 4 g)beschrieben. Nur bei 4 a) wird der Datenstreifen nicht in den Flexowriter, sondern in den Schnelleser gelegt.

VII. Fehleranzeige während der Rechenphase

Wenn während der Rechnung Überläufe oder unzulässige Zahlenwerte vorkommen, wird das durch eine entsprechende Fehlernummer, deren Bedeutung aus der untenstehenden Liste entnommen werden kann, angezeigt.

Die Fehlerausschrift sieht folgendermaßen aus:

Error - Nr. des Fehlers

Rückkehradresse des jeweiligen Unterprogramms ins Hauptprogramm	Zahlenwert, der beim Sprung ins Unterprogramm im Akkumulator stand, als Ganzzahl und als Gleitkomma-Zahl.

Nach jeder Fehlerausschrift erfolgt ein Stop.

Beispiel einer Fehleranzeige:

Bei einer Ganzzahlmultiplikation von +9999999 mal +9999999 tritt ein Überlauf auf. Dieser Überlauf wird in folgender Weise angezeigt:

Error - 3204

0344 9999999 .01862639 e 30

Liste der Fehler:

Error	- 3204 - ix	Überlauf
"	- 3304 - fix	zu groß
"	- 3404 - i/	i/Null
"	- 3504 - unflo	zu groß

Error - 3608 - x10p		Überlauf
"	- 3708 - +	Überlauf
"	- 3808 - -	Überlauf
"	- 3908 - x	Überlauf
"	- 4008 - /	Überlauf
"	- 4016 - /	Division durch Null
"	- 4108 - flo	Überlauf
"	- 4216 - sqrt	Argument negativ
"	- 4316 - sin	Argument zu groß
"	- 4416 - cos	Argument zu groß
"	- 4516 - ln	Argument ≤ 0
"	- 4616 - log	Argument ≤ 0
"	- 5008 - exp	Überlauf
"	- 5104 - ipwr	Überlauf
"	- 5204 - pwr	Argument negativ
"	- 5208 - pwr	Überlauf
"	- 5216 - pwr	0^{-p}
"	- 5304 - ipch	zu groß
"	- 5408 - read	Exponent zu groß
"	- 5416 - read	rdxit nicht gesetzt
"	- 5516 - iread	rdxit nicht gesetzt
"	- 6208 - go to s0	s0 nicht gesetzt
"	- 6300 - fehlendes	Funktionsunterprogramm

VIII. Compilieren von Prozeduren

Wenn Prozeduren benutzt werden, müssen diese immer vor dem eigentlichen Hauptprogramm übersetzt werden. Die Bedienung beim Compilieren von den Quellenprogrammen der Prozeduren ist genauso wie bei anderen Quellenprogrammen auch (siehe unter II.) . Der einzige Unterschied zum Compilieren eines normalen Programmes besteht darin, daß nach dem Compilieren einer Prozedur keine Variablen- und Anweisungsliste ausgeschrieben wird.

Durch den Operator "end" am Ende der Prozedur werden alle Anweisungskennzeichen und alle lokalen Variablen der jeweiligen Prozedur gelöscht. Die Speicherreservierung für die Anweisungskennzeichen und Variablen bleibt allerdings erhalten, nur können sie nicht mehr ausgegeben werden.

Möchte man sie aber doch noch ausgeschrieben haben, muß vor dem Operator "end" der Operator "wait" geschrieben werden. Nach "wait" stoppt der Rechner. Durch Sprung nach 5917 (.0005917)

erhält man die Ausschrift der Anweisungs- und lokalen Variablenliste. Danach stoppt der Rechner wieder. Es muß jetzt von Hand das Sprung-Codewort .0004211 eingegeben werden, damit noch "end" eingelesen werden kann (siehe auch Ablaufdiagramm zum Compilieren von Prozeduren).

Gehören zu einem Programm noch weitere Prozeduren, so brauchen nur die Quellenprogramme hintereinander in die Lesestation des Flexowriters gelegt und die "Start"-Taste am Rechner gedrückt zu werden. Nach dem Compilieren einer Prozedur erfolgt jeweils ein Stop. Wenn alle benötigten Prozeduren übersetzt sind, wird zum Schluß das Quellenprogramm des Hauptprogramms compiliert. Daran anschließend kann man sich die Objektprogramme aller Prozeduren und des Hauptprogramms auf einen Streifen mit "Normal Punch" oder "Emergency Punch" ausstanzen lassen.

Es ist allerdings auch möglich, Prozeduren einzeln übersetzen zu lassen und das jeweilige Objektprogramm mit "Partial Punch" ausstanzen zu lassen. Ein mit "Partial Punch" ausgestanztes Objektprogramm einer Prozedur kann später vor dem Compilieren eines Hauptprogrammes eingelesen und gespeichert werden.

IX. Trace-Programm für ACT V

1.) Verwendungszweck:

Das Trace-Programm ist ein Hilfsprogramm zum Austesten von Objektprogrammen, die mit ACT V compiliert wurden. Während des Compilierens braucht man auf ein eventuelles späteres "Tracing" keine Rücksicht zu nehmen.

Die Ergebnisse aller Anweisungen in Gleitkomma-Arithmetik und der Anweisungen in Ganzzahl-Arithmetik, die mit "ix" oder "i/" enden, werden während des Programmablaufs ausgeschrieben.

2.) Bedienung:

Das Objektprogramm muß mit den Standard-Unterprogrammen von ACT V und etwa erforderlichen Funktionsunterprogrammen gespeichert sein. Danach wird das Trace-Programm auf zwei beliebige freie Spuren eingelesen. Da das Trace-Programm als relativ hexadezimaler Streifen vorliegt, muß es mit J1-10.0 nach Vorschrift Nr. 12 der Bedienungsanleitung für den LGP-21 eingelesen werden. Der dabei einzugebende Adressenmodifikator entspricht der jeweils festgelegten Anfangsadresse für das Trace-Programm. Die Verknüpfungen (linkages) zwischen dem Trace-Programm und den Standard-Unterprogrammen werden automatisch hergestellt.

Das Objektprogramm kann nun gestartet werden. Ob ein "Trace"-Ausdruck erfolgt, ist abhängig von der Sprungtaste. Ist die Sprungtaste (PST) gelöst, wird nicht ausgeschrieben; nur das Objektprogramm läuft etwas langsamer.

Nur bei gedrückter Sprungtaste arbeitet das Trace-Programm und druckt Ergebnisse aus.

Verfahren:

Bei gedrückter Sprungtaste untersucht das Trace-Programm jeden Befehl des Objektprogramms, der unmittelbar nach einem arithmetischen oder Eingabe-Unterprogramm ausgeführt wird.
Wird ein H-Befehl mit einer Adresse, die zwischen 0000 und 4163 liegt, gefunden, wird daraus der Schluß gezogen, daß es sich um ein brauchbares Ergebnis und um das Ende einer Anweisung handelt.Dann erfolgt ein Wagenrücklauf und es wird folgendes ausgedruckt:

1.) Der Speicherplatz des H-Befehls selbst.

2.) Der zu speichernde Wert als Gleitkomma oder als Ganzzahl, je nachdem, ob es sich um eine Gleitkomma- oder Ganzzahlvariable handelt.

3.) Im Anschluß an ein Semikolon die Operandenadresse, die in dem H-Befehl steht.

Wenn auf der rechten Seite einer Anweisung eine einfach oder doppelt indizierte Variable (z.B. a'i' oder a'j'i') steht, so steht im Objektprogramm folgende Befehlsfolge:

α	800R6100
α +1	U6063
α +2	H(a_o)
α +3	[(i)] oder [(j) (i)]

Dabei bedeutet a_o die Adresse von a'o' (Feldadresse der Speicherbelegungsliste).

In α+3 steht die Adresse des Index i bzw. die Adressen der Indizes j und i. In diesem Fall erkennt das Trace-Programm den 800R6100-Befehl und untersucht den Befehl in α+2. Handelt es sich dabei um einen H-Befehl mit einer Operandenadresse zwischen 0000 und 4163, so erfolgt ebenfalls eine Ausgabe, und zwar in folgender Form:

1.) Speicherplatz des 800R6100-Befehls.

2.) Der zu speichernde Wert als Gleitkomma oder Ganzzahl.

3.) Die Variablen-Adresse, die im H-Befehl in α+2 steht.

4.) Der jeweilige Wert des Index i bzw. der Indizes j und i.

Wenn kein "Tracing" mehr erforderlich scheint, kann das Objektprogramm wieder mit voller Geschwindigkeit laufen, wenn die Verbindung mit dem Trace-Programm gelöst wird. Dazu wird das Objektprogramm angehalten und folgende Operationen ausgeführt:

1.) Schalter am Rechner auf "step" stellen

2.) "Eingabe von Hand" am Flexowriter drücken

3.) Aufruf J1-10.0

4.) .000 L_o eingeben, dabei ist L_o die Anfangsadresse des Trace-Programms

5.) "Rechner Start" am Flexowriter drücken

6.) "Start" am Rechner drücken.

Wenn der Rechner stoppt, ist die Verbindung zwischen Trace-Programm und Objektprogramm gelöst und das Objektprogramm kann wieder mit voller Geschwindigkeit laufen.

Die Verbindung kann auch folgendermaßen gelöst werden:

1.) Schalter am Rechner auf "Step" stellen

2.) "Eingabe von Hand" am Flexowriter drücken

3.) Aufruf J1-10.0

4.) ;0004813' über Flexowriter eingeben

5.) "Rechner Start" am Flexowriter drücken

6.) xU4814' eingeben

7.) "Rechner Start" am Flexowriter drücken.

Die Verbindung ist damit gelöst. Es kann normal mit dem Objektprogramm weitergerechnet werden.

Durch neues Einlesen des Trace-Programm können die Verknüpfungen wiederhergestellt werden. Einfacher und schneller erfolgt es allerdings so:

1.) "Eingabe von Hand" am Flexowriter drücken

2.) Aufruf J1-10.0

3.) ;0004813' über Flexowriter eingeben

4.) "Rechner Start" am Flexowriter drücken

5.) xU (L_o + 0131) über Flexowriter eingeben
L_o = Anfangsadresse des Trace-Programms

6.) "Rechner Start" am Flexowriter drücken.

Die Verbindung zwischen Trace-Programm und dem Objektprogramm ist wiederhergestellt. Bei der anschließenden Rechnung mit dem Objektprogramm wird wieder das Trace-Programm benutzt.

Eingabe- und Ausgabeeinheit

Das Trace-Programm benötigt keine Eingabe. Zur Ausgabe benutzt das Trace-Programm die in den Standard-Unterprogrammen ausgewählte Einheit.

Wenn als Ausgabeeinheit in den Standard-Unterprogrammen der Flexowriter benutzt wird, muß der Programmstreifen "Trace", Ausgabe über Flexowriter, eingelesen werden.
Die gesamte Ausgabe erfolgt dann über Flexowriter.

Ist in den Standard-Unterprogrammen der Schnellstanzer ausgewählt worden, muß der Programmstreifen "Trace", Ausgabe über Schnellstanzer, eingelesen werden. Die gesamte Ausgabe erfolgt dann über Schnellstanzer.

Speicherbedarf

2 Spuren und Zwischenspeicher der Spur 63.

X. ACT V "CHECKOUT" - Programm

Verwendungszweck

Das "CHECKOUT"-Programm ist ein Hilfsprogramm zur Überprüfung von Objektprogrammen, die mit ACT V compiliert wurden.

Mit dem "CHECKOUT"-Programm können die verschiedenen Speicherinhalte ausgedruckt werden, und zwar: Dezimale Befehle, hexadezimale Konstanten sowie Daten im Ganzzahl- oder Gleitkomma-Format.

Ferner können Daten im Ganzzahl oder Gleitkomma-Format eingelesen und in beliebig wählbaren Zellen gespeichert werden.

Bedienung

Das Objektprogramm muß mit den Standard-Unterprogrammen und etwa erforderlichen Funktions-Unterprogrammen gespeichert sein. Danach wird das "CHECKOUT"-Programm auf zwei beliebige freie Spuren eingelesen. Da das "CHECKOUT"-Programm als relativ hexadezimaler Streifen vorliegt, muß es mit Hilfe von J1-10:0 nach Vorschrift Nr. 12 der Bedienungsanleitung für den LGP-21 eingelesen werden. Der dabei einzugebende Adressenmodifikator entspricht der jeweils festgelegten Anfangsadresse für das "CHECKOUT"-Programm.

Zwischen dem Objektprogramm und dem"CHECKOUT"-Programm besteht kein Zusammenhang. Jedes Programm läuft unabhängig von dem anderen. Der Sprung an den Anfang des "CHECKOUT"-Programms geschieht folgendermaßen:

1.) "Eingabe von Hand" am Flexowriter drücken

2.) Aufruf J1-10.0

3.) .000 L_o über Flexowriter eingeben;
dabei bedeutet L_o die Anfangsadresse des "CHECKOUT"-Programms

4.) "Rechner Start" am Flexowriter drücken

5.) "Start" am Rechner drücken.

Nach dem "Start" am Rechner erfolgt zunächst eine Verzögerung von ca. 1 Sekunde. Dann vollführt der Flexowriter einen Wagenrücklauf und einen Tabulator-Sprung. Dann kommt die Eingabelampe als Zeichen, daß ein Schlüsselwort erwartet wird.

Je nach der Art des Schlüsselwortes kann das "CHECKOUT"-Programm vier grundsätzliche Funktionen erfüllen:

a) Ausdrucken von Befehlen

Eingabe des Schlüsselwortes

pittss

nachdem die Eingabelampe aufleuchtet. ttss ist die dezimale Anfangsadresse, von der ab Befehle ausgedruckt werden sollen. Reine Befehle werden mit Operationsteil und dezimaler Adresse ausgedruckt. (Negative Befehle mit - Vorzeichen).

Alles übrige wird als hexadezimales Wort ausgegeben.

Nach jeweils 8 Befehlen erfolgt ein Wagenrücklauf. Ähnlich wie bei dem Programm K2-10.1 wird am Beginn jeder Zeile die dezimale Speicheradresse ausgedruckt.

Das Ausdrucken erfolgt so lange, bis die Taste PS 4 gedrückt wird. Dann hört der Ausdruck mit dem Ende der letzten Zeile auf und der Flexowriter führt Wagenrückläufe aus in Abständen von etwa 1 Sekunde. Wenn die Taste PS 4 gelöst wird, erfolgt ein Wagenrücklauf, ein Tabulator-Sprung und die Eingabelampe leuchtet auf zur Eingabe eines neuen Schlüsselwortes.

b) Ausdrucken von hexadezimalen Konstanten

Eingabe des Schlüsselwortes

phttss

nachdem die Eingabelampe aufleuchtet. Der Speicherinhalt wird in hexadezimaler Form von der dezimalen Anfangsadresse ttss an ausgegeben. Es werden ebenfalls 8 Worte pro Zeile und am Beginn der Zeile die jeweilige dezimale Speicheradresse gedruckt. Der Ausdruck erfolgt so lange, bis PS 4 gedrückt wird. Dann weiter wie unter a).

c) Ausdrucken von Daten vom Ganzzahl- und Gleitkomma-Format

Eingabe des Schlüsselwortes

pdttss

ttss ist die dezimale Speicheradresse des ersten Datenwortes. Nach einem Wagenrücklauf wird zunächst die dezimale Speicheradresse und anschließend das Datenwort im Ganzzahl- und Gleitkomma-Format ausgedruckt.

In der nächsten Zeile wird dann in gleicher Weise das Datenwort aus der um 1 verminderten Speicheradresse ausgedruckt. Diese absteigende Folge der Adressen entspricht der aufsteigenden Reihe der Variablen des Objektprogramms.

Datenfelder werden also mit steigenden Indexwerten ausgedruckt. Der Index selbst wird nicht angegeben. Es ist ferner zu beachten, daß doppelt-indizierte Felder spaltenweise gespeichert sind. Der Ausdruck erfolgt wiederum so lange, bis PS 4 gedrückt wird. Dann geht es nach dem Ausdruck des letzten Datenwortes weiter wie unter a).

d) Eingabe von Daten im Ganzzahl- oder Gleitkomma-Format

Eingabe des Schlüsselwortes

rdttss

ttss ist die dezimale Speicheradresse, wohin das erste Datenwort gespeichert werden soll. Nach einem Wagenrücklauf wird die dezimale Speicheradresse nochmals ausgedruckt, dann erfolgt ein Tabulatorsprung und es leuchtet die Eingabelampe auf zur Eingabe der Daten.

Wenn PS-T gelöst ist, muß die Eingabe im Ganzzahl-Format erfolgen. (Vorzeichen und 1 bis 7 Dezimalziffern.) Das "CHECKOUT" -Programm vollführt in diesem Falle die Befehlsfolge für IREAD und liest mit Hilfe der Standard-Unterprogramme ein.

Wenn Daten im Gleitkomma-Format eingegeben werden sollen,ist nach dem Aufruf des "CHECKOUT"-Programms, wenn die Eingabelampe aufleuchtet und noch vor der Eingabe des Schlüsselwortes rdttss,die Taste PS-T zu drücken. Es wird wiederum ein Wagenrücklauf ausgeführt, die dezimale Speicheradresse ausgedruckt, dann ein Tabulatorsprung ausgeführt und dann leuchtet die Eingabelampe auf zur Eingabe der Daten.
Jetzt ist aber die Befehlsfolge für READ ausgeführt worden und mit Hilfe der Standard-Unterprogramme kann im Gleitkomma-Format eingelesen werden. (Das erste Datenwort enthält Vorzeichen und 1 bis 7 Dezimalziffern für die Mantisse, das zweite Datenwort enthält Vorzeichen und 1 bis 2 Dezimalziffern für den Exponenten.)

Nach der Eingabe in dem ausgewählten Format erfolgt ein Wagenrücklauf, dann wird die um 1 verminderte Speicheradresse ausgedruckt und nach einem Tabulatorsprung leuchtet die Eingabelampe auf für den nächsten Datenwert im gleichen Format.

Auf diese Weise können Daten in eine absteigende Folge von Adressen gespeichert werden (siehe auch unter c).

Soll von einer bestimmten Adresse an das Eingabeformat gewechselt werden, so kann folgendermaßen verfahren werden:

Wenn die betreffende Speicheradresse ausgedruckt, der Tabulatorsprung ausgeführt ist und die Eingabelampe aufleuchtet, wird zunächst die Stellung von PS-T gewechselt (d.h. EIN bei Übergang von Ganzzahl zu Gleitkomma und AUS im entgegengesetzten Fall). Es wird kein Datenwort eingegeben, sondern ohne Eingabe nur "Rechnen Start" am Flexowriter gedrückt.

Jetzt tritt der durch das "CHECKOUT"-Programm gesetzte Schalter RDXIT in Aktion. Der Flexowriter vollführt einen Wagenrücklauf, druckt nochmals die jeweilige Speicheradresse und dann leuchtet wiederum die Eingabelampe auf. Jetzt ist aber die Befehlsfolge für das neue Format ausgeführt und über Standard-Unterprogramme kann entsprechend Wert für Wert eingelesen und mit dem "CHECKOUT"-Programm gespeichert werden.

Zur Beendigung der Dateneingabe wird folgendermaßen verfahren:

Nach Eingabe des letzten Wertes wird abgewartet, bis in einer neuen Zeile die nächst niedrige Adresse ausgedruckt, der Tabulatorsprung ausgeführt ist und die Eingabelampe aufleuchtet: Jetzt wird PS 4 gedrückt und die Taste PS-T gelöst, falls sie gedrückt war. Dann wird ohne Eingabe nur "Rechnen Start" am Flexowriter gedrückt. Über RDXIT wird dann dieser Teil des "CHECKOUT"-Programms verlassen. Der Flexowriter beginnt wieder Wagenrückläufe in Abständen von etwa 1 Sekunde auszuführen, solange PS 4 gedrückt ist. Nach dem Lösen von PS 4 erfolgt wiederum Wagenrücklauf, Tabulatorsprung und die Eingabelampe leuchtet auf zur Eingabe eines neuen Schlüsselwortes.

Anmerkungen

Wenn das "CHECKOUT"-Programm ganz verlassen werden soll, so ist nach Beendigung der zuletzt benötigten Funktion außer PS 4 gleichzeitig auch PS-T zu drücken. Dann erfolgt ein Sprung in die Programmeingabe, d.h. nach einem Wagenrücklauf kommt kein Tabulatorsprung, sondern es leuchtet die Eingabelampe auf. Jetzt befindet sich der Rechner aber in der Programmeingabe J1-10.0. Es wird also ein Codewort des Eingabeprogramms erwartet!

Wenn der Abschnitt d), Eingabe von Daten des "CHECKOUT"-Programms , benutzt wurde, ist der Schalter RDXIT in den Standard-Unterprogrammen nicht mehr in seiner früheren Stellung. Wenn er im Objektprogramm benutzt wird, so ist bei der Rückkehr in das Objektprogramm darauf zu achten, daß er zunächst wieder neu gesetzt wird.

Eingabe- und Ausgabe-Einheit

Das "CHECKOUT"-Programm benutzt zur Ein- und Ausgabe den Flexowriter. Die Standard-Unterprogramme müssen vor Benutzung des "CHECKOUT"-Programms auf Eingabe und Ausgabe über Flexowriter eingestellt werden.

Speicherbedarf

2 Spuren, Ein- und Ausgabe UP in den Standard-Unterprogrammen, Adressenbinärisierung in J1-10.0, Zwischenspeicher in Spur 63.

"Load and Go" - Compiler ACT V

I. Speicherbelegung beim Compilieren und Rechnen mit dem "Load and Go" - Compiler

Der "Load and Go"-Compiler belegt die Spuren 1200 bis 4163. Die Standard-Unterprogramme liegen wie beim Standard-Compiler von 4200 bis 6163.

Das Objektprogramm einschließlich der Speicherplätze für Variable kann sich also nur von 0300 bis maximal 1163 erstrecken. Wenn Funktionsunterprogramme gebraucht werden, müssen diese auch noch in dem Objektprogrammbereich mit untergebracht werden.

II. Compilieren und Rechnen mit dem "Load and Go" -Compiler

1.) Compiler "Load and Go" und Standard-Unterprogramme mit J1-10.0 nach Vorschrift Nr. 9 der Bedienungsanleitung für den LGP-21 speichern.

2.) "Eingabe von Hand" am Flexowriter.

3.) Lochstreifen des Quellenprogrammes in Lesestation des Flexowriters legen.

4.) Aufruf J1-10.0

   a) Schalter auf "Step" am Rechner

   b) Taste "Füllen/Löschen" am Rechner drücken

   c) Schalter auf "Normal" am Rechner

   d) Taste "Start" am Rechner drücken

5.) Schreiben des Schlüsselwortes .0002000' auf dem Flexowriter

6.) "Rechner Start" am Flexowriter drücken

7.) "Eingabe von Hand" am Flexowriter lösen

8.) "Start" am Rechner drücken; Quellenprogramm wird übersetzt.

   Nach dem Übersetzen kann sofort mit der Rechnung begonnen werden, falls keine Funktionsunterprogramme gebraucht werden.

9.) Datenstreifen in Lesestation vom Flexowriter legen

10.) "Eingabe von Hand" am Flexowriter drücken

11.) Aufruf J1-10.0

12.) Schreiben des Schlüsselwortes .000xxxx (XXXX = Anfangsadresse des Objektprogramms, meistens ist es 0300)

13.) "Rechner Start" am Flexowriter drücken

14.) "Eingabe von Hand" am Flexowriter lösen

15.) "Start" am Rechner drücken

Die Daten werden eingelesen und die Rechnung wird ausgeführt. Nach Beendigung der Rechnung kann sofort ein neues Quellenprogramm compiliert werden, denn der Compiler ist ja durch das Objektprogramm nicht zerstört worden.

"Load and Go" - Compiler ACT V für den LGP-21

mit doppelter Speicherkapazität

I. Speicherbelegung während der Compilierphase

Die Erweiterung der Speicherkapazität auf 8192 Worte ermöglicht es, immer den "Load and Go"-Compiler zu benutzen. Das heißt, Compiler und Standard-Unterprogramm können immer gleichzeitig gespeichert sein. Der "Load and Go"-Compiler belegt, wie bisher, die Speicherplätze 1200 - 4163, das Standard-Unterprogramm die Speicherplätze 4200 - 6163. Die Spur 63 wird weiterhin als Zwischenspeicher benutzt. Für die Speicherung der Konstanten des Objektprogramms liegt der Compiler in zwei Versionen vor. Einmal die bisherige Form mit maximal 63 Konstanten in Spur 62 und einmal mit maximal 127 Konstanten in den Spuren 126 und 127.

Das Objektprogramm beginnt immer in 6400. Die Variablen werden, wenn für die Konstanten die Spur 62 benutzt ist, von 12762 abwärts oder, wenn für die Konstanten die Spuren 126 und 127 benutzt sind, von 12563 abwärts gespeichert.

Zusätzlicher Operator " var "

Wenn die Konstanten des Quellenprogramms in den Spuren 126 und 127 gespeichert werden, besteht die Möglichkeit, den Compiler um den Operator " var " zu erweitern.

Der Operator " var " ordnet den Variablen feste Speicheradressen zu. Damit können bei langen Programmen bzw. großen Datenmengen auch Speicherplätze unter 4163 für Variable reserviert werden. Der Compiler wird dann allerdings während der Rechenphase zerstört. Der Operator " var " ist wie " dim " anzuwenden, es wird nur statt der Feldgröße die Anfangsadresse angegeben. Die Feldgröße muß bei der Festlegung der nächsten Variablenadresse vom Programmierer berücksichtigt werden.

Beispiel :

Die Variable a hat nur 1 Speicherplatz, Variable b soll indiziert und ein Feld von 20 Speicherplätzen belegen, Variable c hat auch nur einen Speicherplatz. Die Variablen sollen ab 2963 gespeichert

werden. Dann muß die Anweisung heißen:

var'a'2963'b'2962'c'2942''

Im Quellenprogramm können mehrere " var "-Anweisungen vorkommen. Eine Anweisung kann beliebig lang sein. Wichtig ist nur, daß sie vor der ersten Anweisung steht, in der die Variable verwendet wird.

Diese Programmerweiterung liegt als gesonderter Streifen vor und ist zusätzlich zum Compiler einzulesen. Sie belegt einen Teil der Spur 62.

II. Compilieren mit dem ACT V

Am Compilieren hat sich nichts geändert.
Der Aufruf ist .0002000'.
Die Eingabe des Quellenprogramms erfolgt über den Flexowriter.
Die Fehleranzeige ist ebenfalls unverändert.

III. Ausstanzen des Objektprogrammes

Zum Ausstanzen des Objektprogrammes sind 2 Programme vorhanden.

1) Normal-Punch zum Ausstanzen von vollständigen Programmen. Es liegt in zwei Fassungen vor, rel. dezimal und hexadezimal fest in 0900 - 1163. Der Aufruf ist Lo bzw. 0900. Das Programm kann über Flexowriter oder Tally-Stanzer ausgegeben werden. Auf dem hex. Programmstreifen ist .0000900' abgelocht.

2) Partial-Punch zum Ausstanzen von Teilprogrammen und Proceduren. Es belegt eine Spur zusätzlich zum Normal-Punch, und zwar vor dem Normal-Punch. Das Programm liegt ebenfalls in 2 Fassungen vor, rel. dezimal und hexadezimal fest in 0800 - 1163. Der Aufruf ist Lo bzw. 0800. Auf dem hexadezimalen Streifen ist .0000800' abgelocht. Bei erneutem Aufruf muß von Hand nach Lo gesprungen werden, vom Programm erfolgt ein Sprung nach Normal-Punch.

   Mit Partial-Punch ausgestanzte Programme können später nach der Eingabe des Compilers mit HVI oder einem anderen Eingabeprogramm gespeichert serden. Das . -Codewort am Streifende: .0002624', führt automatisch an die Stelle des Compilers, wo weitercompiliert

werden kann.

Wird der Compiler nicht neu eingelesen, muß, bevor das Teilprogramm gespeichert wird, ein Sprung nach 2000 erfolgen, damit die alten Statement- und Variablenlisten gelöscht werden. Das Löschen der Listen ist beendet, wenn der Compiler die erste Eingabe verlangt (am besten am Flexowriter "Eingabe von Hand" drücken).

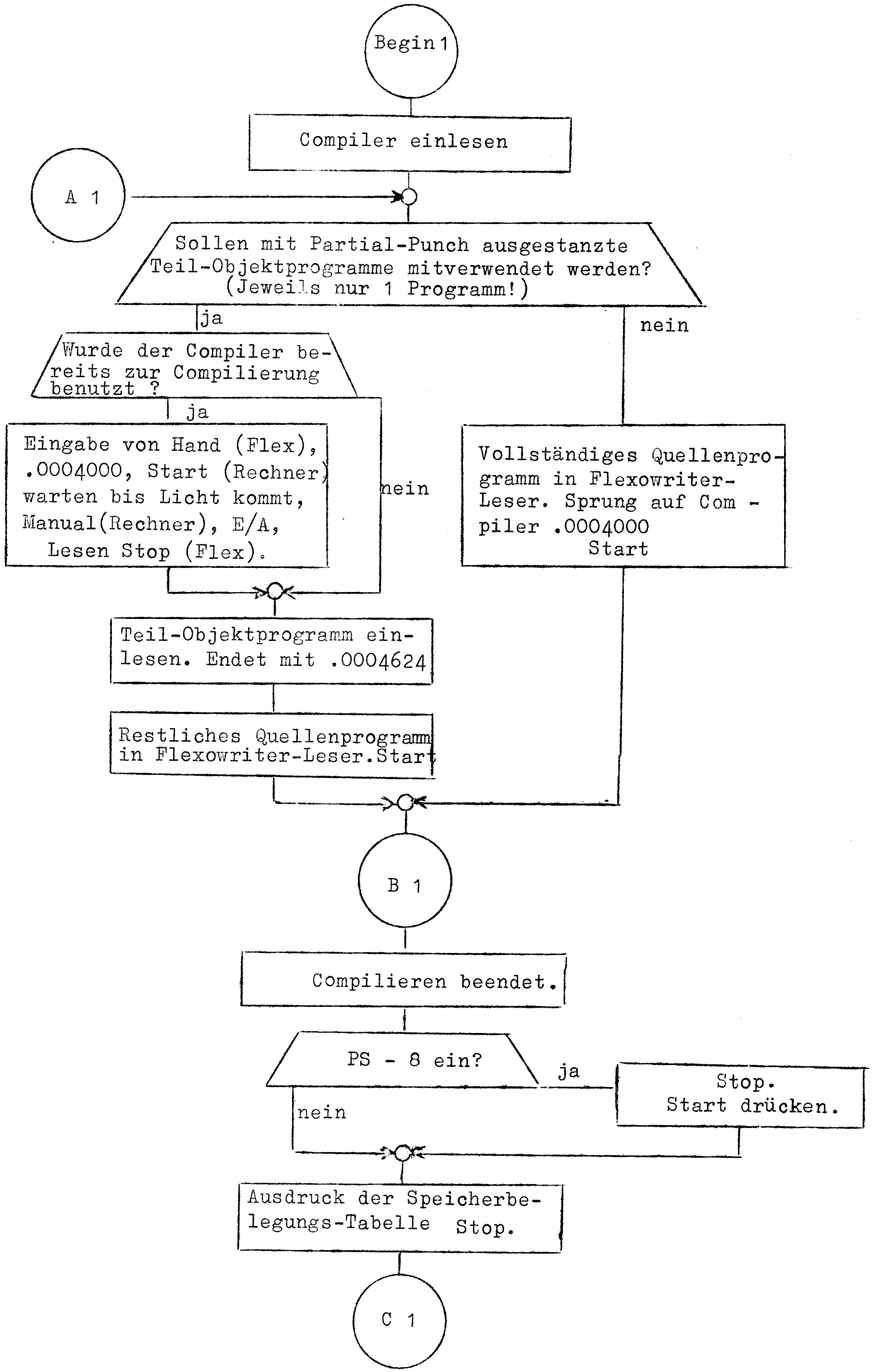
IV. Arbeiten mit dem Objektprogramm

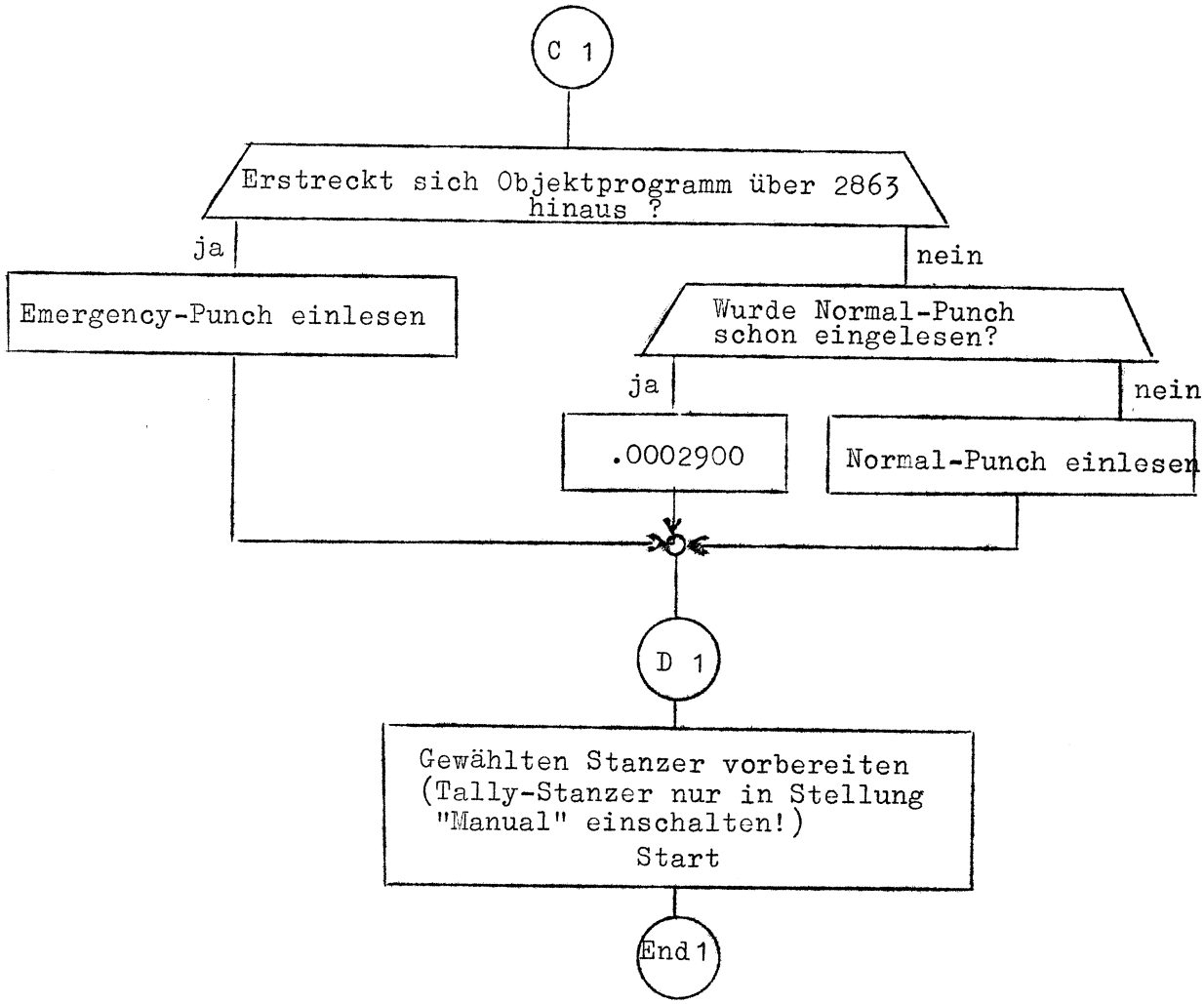
Das Objektprogramm beginnt in 6400 und kann maximal bis 12763 gehen. An der Arbeitsweise hat sich nicht verändert. Die Standard-Unterprogramme müssen immer gespeichert sein. Sie belegen wieder die Speicherplätze 4200 - 6163. Die Funktions-Unterprogramme liegen in rel. hexadezimaler Form vor.

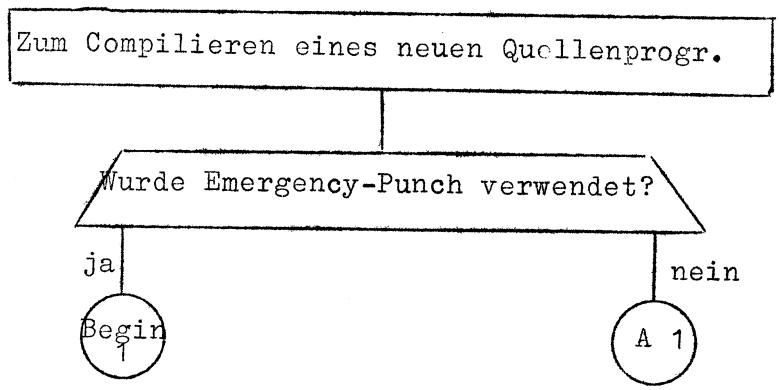
Zum Austesten liegen Trace- und Checkout-Programm ebenfalls in rel. hexadezimaler Form vor. Beim Trace-Programm werden nicht mehr nur die Ergebnisse einer Anweisung sondern auch die Zwischenergebnisse (H6300 usw) ausgeschrieben. Sonst ist die Arbeitsweise dieser Programme unverändert.
An der Fehleranzeige während der Rechenphase hat sich ebenfalls nichts geändert.

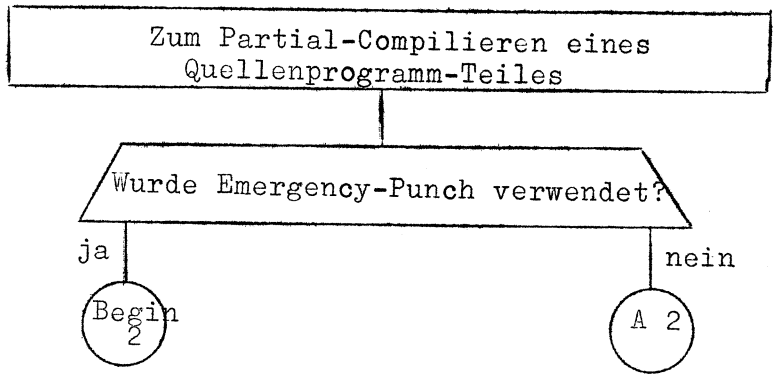
A C T V - LGP - 21

Compilieren mit Standard-Compiler



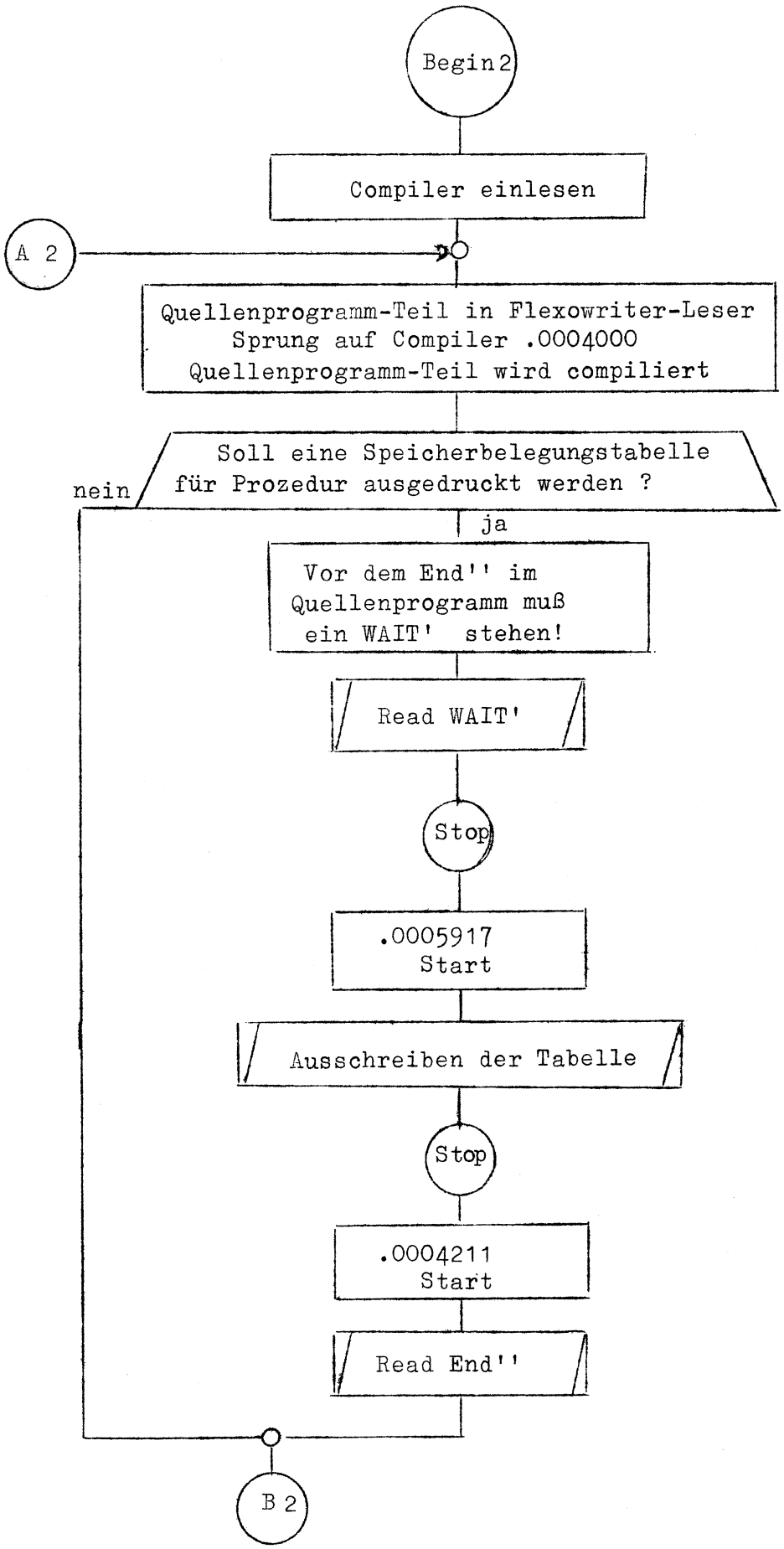

C 1
Erstreckt sich Objektprogramm über 2863 hinaus ?
ja
nein
Emergency-Punch einlesen
Wurde Normal-Punch schon eingelesen?
ja
nein
.0002900
Normal-Punch einlesen
D 1
Gewählten Stanzer vorbereiten
(Tally-Stanzer nur in Stellung "Manual" einschalten!)
Start
End 1

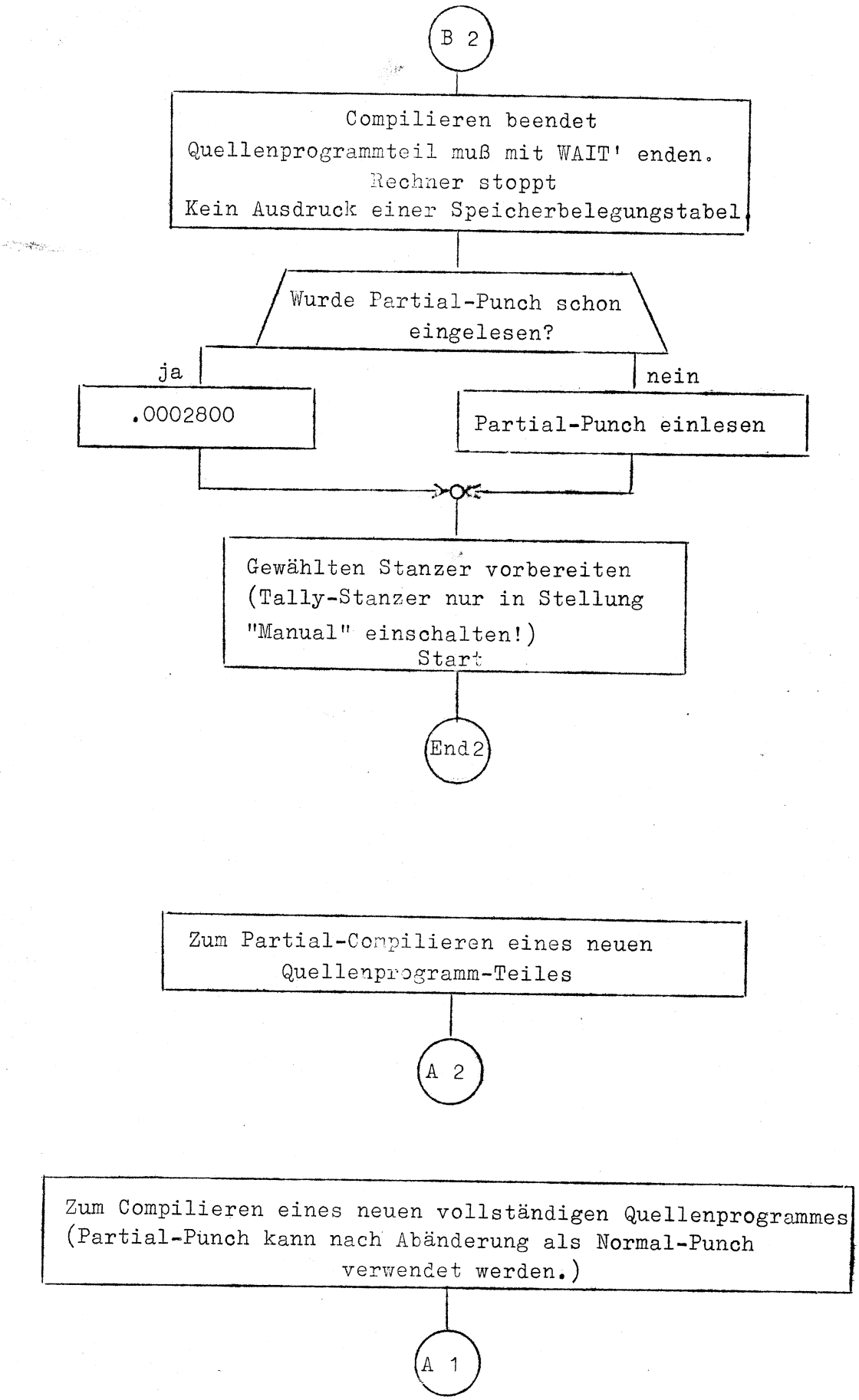

Zum Compilieren eines neuen Quellenprogr.
Wurde Emergency-Punch verwendet?
ja
nein
Begin 1
A 1


Zum Partial-Compilieren eines Quellenprogramm-Teiles
Wurde Emergency-Punch verwendet?
ja
nein
Begin 2
A 2

A C T V - LGP-21

Compilierung von Programmteilen.
(Prozeduren usw.)




B 2
Compilieren beendet
Quellenprogrammteil muß mit WAIT' enden.
Rechner stoppt
Kein Ausdruck einer Speicherbelegungstabel
Wurde Partial-Punch schon eingelesen?
ja
nein
.0002800
Partial-Punch einlesen
Gewählten Stanzer vorbereiten
(Tally-Stanzer nur in Stellung "Manual" einschalten!)
Start
End2
Zum Partial-Compilieren eines neuen Quellenprogramm-Teiles
A 2
Zum Compilieren eines neuen vollständigen Quellenprogrammes
(Partial-Punch kann nach Abänderung als Normal-Punch verwendet werden.)
A 1