Heike und Manfred Fillinger

Ein Tiny-Pascal-Compiler

Teil 2

Der erste Teil des Beitrages brachte die Syntax des Compilers und ein bißchen Einführung in das „Einfach-Pascal", das hier verwendet wird. Jetzt geht es um die Arbeitsweise des Compilers selbst.

Beim Tiny-Pascal-Compiler handelt es sich um ein Basic-Programm mit 8080-Maschinen-Routinen (siehe Listing). Die REM-Zeilen ab 1460 können bei der Eingabe fortgelassen werden.

Die Programm-Zeile 1 dient als Platzhalter-Zeile für Maschinen-Routinen. Die Programm-Zeile 9 dient als Übergabezeile von der Maschinen-Ebene (nach Eingabe des Befehls CMD) in das Dialog-Modul des Compilers.

Anwender, welche über einen Hauptspeicherausbau von mehr als 16 KByte verfügen, seien folgende Vergrößerungen der STRING-Felder empfohlen:
1470 CLEAR 2000
1480 DIM IP $(30), IB$(30), RP$(30), RB$(30), SP$(30), SB$(30),QS$(30),

Benutzerdialog

Nach dem Laden des Compilers wird, mit Eingabe des Befehles, RUN, der Lader für die Maschinen-Routinen gestartet. Die Maschinen-Routinen werden im Hauptspeicher von hex 403E bis hex 4071 und im Textbereich der ersten REM-Zeile geladen. Die Datenfelder, welche Maschinen-Routinen enthalten, werden anschließend automatisch gelöscht. Hierdurch wird die Länge des Compilers erheblich gekürzt. Nach Beendigung dieser Aktion meldet sich der Compilers mit:

```
>RUN
READY
```

Jetzt ist erneut der Befehl RUN zu geben. Es werden automatisch die aktuelle Länge des Compilers bestimmt, der Compilers verdeckt und die entsprechenden Zeiger für den Basic-Interpreter eingerichtet. Der Compiler meldet sich anschließend mit:

```
>RUN
PASCAL: 'EDIT, CSAVE, CLOAD'
READY
```

Der Benutzer kann nun ein Tiny-Pascal-Programm unter Ausnutzung aller Editor-Hilfen des Basic erstellen und aufzeichnen (CSAVE'File-Name'). Oder ein vorhandenes Tiny-Pascal-Programm laden (CLOAD'File-Name'). Bei der Neu-

```
10000 "PROGRAM VERZINSUNG;
10010 "(*BERECHNET AUS ANF.KAPITAL,ZINSEN,JAHRE DAS ENDKAPITAL*)
10020 "
10030 "VAR
10040 "    KAPITAL,ZINSSATZ : REAL;
10050 "    ENDKAPITAL,FAKTOR : REAL;
10055 "    I,JAHRE : INTEGER;
10060 "
10070 "PROCEDURE EINGABE;
10080 "(*EINGABE-WERTE IM DIALOG HOLEN*)
10090 "    BEGIN
10100 "        WRITE('ANFANGS-KAPITAL IM VOLLEN DM :');
10110 "        READ(KAPITAL);
10115 "        WRITELN;
10120 "        WRITE('JAHRE :');
10130 "        READ(JAHRE);
10135 "        WRITELN;
10140 "        WRITE('ZINSSATZ IN PROZENT :');
10150 "        READ(ZINSSATZ);
10155 "        WRITELN
10160 "    END; (*EINGABE*)
10170 "
10180 "PROCEDURE RECHNEN;
10190 "(*BERECHNUNG UND AUSGABE DES ENDKAPITALS*)
10200 "    BEGIN
10210 "        FAKTOR := 1.0 + ZINSSATZ / 100.0;
10220 "        FOR I := 1 TO JAHRE DO
10230 "            KAPITAL := KAPITAL * FAKTOR;
10235 "        ENDKAPITAL := TRUNC(KAPITAL + 0.5);
10240 "        WRITELN('ENDKAPITAL IN DM :',ENDKAPITAL)
10250 "    END; (*RECHNEN*)
10260 "
10270 "BEGIN (*HAUPTPROGRAMM*)
10280 "    WRITELN('VERZINSUNG');
10290 "    EINGABE;
10300 "    RECHNEN
10310 "END. (*VERZINSUNG*)
```

Bild 24. Das Programm Verzinsung muß so eingegeben werden

erstellung empfiehlt es sich, Zeilennummern ab 10 000 zu verwenden. Nach Eingabe der Zeilennummer müssen ein Leerzeichen (Blank) und ein Anführungszeichen folgen. Das Anführungszeichen verhindert, daß die eingegebene Programm-Zeile vom Basic-Interpreter im Platz-Spar-Code abgelegt wird. Dieser Compiler arbeitet in weiten Grenzen formatfrei. Gegenüber Standard-Pascal ergeben sich jedoch, wegen des knappen Speicherplatzes und der zeilenweisen Übersetzung einige Einschränkungen. Die Schlüsselwörter CONST und VAR müssen jeweils eine eigene Programmzeile bilden. Innerhalb eines Kommentares (*.....*) darf kein Zeilenende auftreten. Strukturierte Anweisungen, welche die Schlüsselwörter D0 bzw. 0F enthalten, müssen vom Erscheinen des ersten Schlüsselwortes der strukturierten Anweisung bis D0 bzw. 0F in einer Programmzeile geschrieben werden. Diese Einschränkungen bezüglich des Programmaufbaus dürften jedoch, wie die Programm-Beispiele zeigen, kein Hindernis beim Programmieren darstellen.
Als Beispiel soll folgendes Pascal-Programm erstellt, übersetzt und ausgeführt werden (Bild 24).

Ist ein übersetzungsfähiges Pascal-Programm im Hauptspeicher vorhanden, kann mit dem Übersetzen begonnen werden. Dazu ist der Befehl CMD zu geben. Dieser Befehl ruft durch Ausnutzung der Maschinen-Routinen die Dialog-Routinen des Compilers auf.

Er meldet sich mit folgendem Menü:
>CMD
1: PASCAL-PROG. LISTEN?
2: COMPILER-OUTPUT?
3: PASCAL-PROG. MIT BASIC-CODE ÜBERSCHREIBEN?
4: NEUERSTELLUNG,PASCAL: 'EDIT,CSAVE,CLOAD' ?

1: Listet das Pascal-Programm ohne Zeilennummern.

   Dieses Listing kann wahlweise über Drucker ausgegeben werden.

2: Erzeugt einen Compiler-Output. Der generierte Basic-Code wird dabei nicht im Hauptspeicher abgelegt. Trifft der Compiler auf einen Fehler, so wird der Übersetzungsvorgang abgebrochen. Sollte sich der Basic-Interpreter mit einem Fehler melden, kann durch Eingabe des Befehles RUN 5070 der Compiler verdeckt, und der Pascal-Quell-Code aufgedeckt werden.

```
>CMD
1: PASCAL-PROG. LISTEN ?
2: COMPILER-OUTPUT     ?
3: PASCAL-PROG. MIT BASIC-CODE UEBERSCHREIBEN ?
4: NEUERSTELLUNG , PASCAL''EDIT,CSAVE,CLOAD'  ?

? 2
OUTPUT AUF DRUCKER (J,N) ? N
TINY-PASCAL-COMPILER REAL VERSION 1.0 STARTET


PROGRAM VERZINSUNG;
--- 10000 REM PROGRAM VERZINSUNG;
--- 10001 CLEAR 1000:DEFINT G,B,J

(*BERECHNET AUS ANF.KAPITAL,ZINSEN,JAHRE DAS ENDKAPITAL*)
--- 11099 REM  (*BERECHNET AUS ANF.KAPITAL,ZINSEN,JAHRE DAS ENDKAPITAL*)

VAR
    KAPITAL,ZINSSATZ : REAL;
    ENDKAPITAL,FAKTOR : REAL;
    I,JAHRE : INTEGER;

PROCEDURE EINGABE;
(*EINGABE-WERTE IM DIALOG HOLEN*)
    BEGIN
        WRITE('ANFANGS-KAPITAL IM VOLLEN DM :');
--- 20000 PRINT"ANFANGS-KAPITAL IM VOLLEN DM :";

        READ(KAPITAL);
--- 20001 INPUTRA

        WRITELN;
--- 20002 PRINT

        WRITE('JAHRE :');
--- 20003 PRINT"JAHRE :";
        READ(JAHRE);
--- 20004 INPUTGB

        WRITELN;
--- 20005 PRINT

        WRITE('ZINSSATZ IN PROZENT :');
--- 20006 PRINT"ZINSSATZ IN PROZENT :";
        READ(ZINSSATZ);
--- 20007 INPUTRB

        WRITELN
--- 20008 PRINT

    END; (*EINGABE*)
--- 20009 RETURN

PROCEDURE RECHNEN;
(*BERECHNUNG UND AUSGABE DES ENDKAPITALS*)
    BEGIN
        FAKTOR := 1.0 + ZINSSATZ / 100.0;
--- 20100 RD=1.0+RB/100.0

        FOR I := 1 TO JAHRE DO
--- 20101 FORGA=1TOGB

            KAPITAL := KAPITAL * FAKTOR;
--- 20103 RA=RA*RD
--- 20104 NEXT

        ENDKAPITAL := TRUNC(KAPITAL + 0.5);
--- 20105 RC=FIX(RA+0.5)

        WRITELN('ENDKAPITAL IN DM :',ENDKAPITAL)
--- 20106 PRINT"ENDKAPITAL IN DM :";RC

    END; (*RECHNEN*)
--- 20107 RETURN


BEGIN (*HAUPTPROGRAMM*)
    WRITELN('VERZINSUNG');
--- 12000 PRINT"VERZINSUNG"

    EINGABE;
--- 12001 GOSUB 20000

    RECHNEN
--- 12002 GOSUB 20100

END. (*VERZINSUNG*)
--- 12003 END
*****PROGRAMMENDE*****
READY
```

Bild 25. Das macht der Compiler beim Übersetzen daraus

```
*****PROGRAMMENDE*****
BASIC 'LIST,CSAVE,RUN'
READY


>LIST
10000 REM PROGRAM VERZINSUNG
10001 CLEAR 1000:DEFINT G,B,J :DEFDBL R,P
11099 REM  (*BERECHNET AUS ANF.KAPITAL,ZINSEN,JAHRE DAS ENDKAPITAL*)
12000 PRINT"VERZINSUNG"
12001 GOSUB 20000
12002 GOSUB 20100
12003 END
20000 PRINT"ANFANGS-KAPITAL IM VOLLEN DM :";
20001 INPUTRA
20002 PRINT
20003 PRINT"JAHRE :";
20004 INPUTGB
20005 PRINT
20006 PRINT"ZINSSATZ IN PROZENT :";
20007 INPUTRB
20008 PRINT
20009 RETURN
20100 RD=1.0+RB/100.0
20101 FORGA=1TOGB
20103 RA=RA*RD
20104 NEXT
20105 RC=FIX(RA+0.5)
20106 PRINT"ENDKAPITAL IN DM :";RC
20107 RETURN
READY
```

Bild 26. Das generierte Basic-Programm

```
1 REM TINY-PASCAL-COMPILER REAL-VERSION : V1.0
  COPYRIGHT 1982 BY H.UND M.FILLINGER 5000 KOELN 71 JOH. PRASSEL-STR. 17
2 CLS:POKE16756,64:POKE16757,64:X=16448:FOR N=0 TO 28:READ B:POKE (X+N),B:NEXT
 N
3 DATA 33,233,66,34,164,64,33,82,85,34,232,65,33,78,57,34,234,65,62,0,50,236,65,
33,231,65,195,126,26
4 X=17136:FOR N=0 TO 112:READ B:POKE(X+N),B:NEXT N:DELETE2-7
5 DATA 62,195,50,184,65,62,52,50,185,65,62,67,50,186,65,175,175,175,87,30,2,42,6
2,64,34,164,64,25,34,251,64,34,249,64,34,253,64,17,232,65
6 DATA 126,254,255,202,62,67,235,119,235,183,202,42,67,19,35,195,24,67,34,112,64
,33,231,65,209,195,129,26,17,232,65,42,112,64,35,195,24,67
7 DATA 62,201,50,184,65,195,204,6,42,164,64,175,119,35,119,35,34,249,64,34,251,6
4,34,253,64,42,62,64,175,119,35,119,195,204,6
8 GOTO5000
9 REM
1460 CLS:DEFINT A-Z
1470 CLEAR 400
1481 REM DIALOG
1482 PRINT"1: PASCAL-PROG. LISTEN ?":PRINT"2: COMPILER-OUTPUT     ?"
1483 PRINT"3: PASCAL-PROG. MIT BASIC-CODE UEBERSCHREIBEN ?":PRINT"4: NEUERSTELLU
NG , PASCAL:'EDIT,CSAVE,CLOAD'  ?"
```

Bild 27. Der Compiler ist lang

```
1484 X9=0:INPUTX7:IF X7=4 THEN5030 ELSE IF NOT((X7=1)OR(X7=2)OR(X7=3))THEN1482 E
LSE INPUT"OUTPUT AUF DRUCKER (J,N) :";X7$:IF  NOT((X7$="J")OR(X7$="N"))THEN1482
1486 IF (X7=2)AND(X7$="J")THEN X8=0 ELSE IF X7=2 THEN X8=1 ELSE IF X7=3 THEN X8=
2 ELSE IF(X7=1)AND(X7$="J")THEN X8=0:X9=1:GOTO1520 ELSE IF X7=1 THEN X8=1:X9=1:G
OTO1520
1488 IF X7=3 THEN INPUT"DATENTYP 'REAL' MIT DOPPELTER GENAUIGKEIT (J,N) :";X8$:I
F NOT((X8$="J")OR(X8$="N"))THEN1482
1490 PRINT"TINY-PASCAL-COMPILER REAL VERSION 1.0 STARTET "
1520 PRINT
1530 REM ANF.SRC
1540 SR=PEEK(16446)+256*PEEK(16447)+256:LS=SR
1550 IF X9=1 THEN GOSUB2610:IF X9=2 THEN5070 ELSE1550
1560 BC=PEEK(16446)+256*PEEK(16447)+2:BL=BC
1580 PRINT
1590 BZ =11000:GZ=20000:DZ=10010
1600 REM 1.SRC-ZEILE LESEN UND ABLEGEN
1610 GOSUB 2610
1620 BC$=" 10000 REM "+SR$
1630 GOSUB 2710
1650 REM STANDARD-ZUWEISUNGEN
1660 BC$=" 10001 CLEAR 1000:DEFINT G,B,J":IF X8$="J"THEN BC$=BC$+":DEFDBL R,P"
1670 GOSUB 2710 :BZ=BZ+99
1680 REM SRC LESEN UND INTERPRETIEREN
1690 VT=0 :VN=0
1700 BC$="":GOSUB 2610
1710 Y$="CONST":GOSUB2840 :IF I <> 0 THEN GOTO1820
1730 Y$="VAR":GOSUB2840 :IF I <> 0 THEN GOTO2080
1750 Y$="PROCEDURE":GOSUB2840 :IF I <> 0 THEN GOTO2360
1760 Y$="BEGIN":GOSUB2840 :IF I <> 0 THEN GOTO3700
1780 Y$="(*":GOSUB2840 :IF I <> 0 THEN GOTO2060
1790 ON VT GOTO 2920 , 1840 ,2920 ,2100 ,
1800 GOTO2890
1810 REM CONST
1820 VT=2 :GOSUB2910
1830 IF I+LEN(Y$)>LEN(SR$) THEN GOTO1700
1840 CO$="":CN$="":CG$="":TY=0
1850 GOSUB2910
1860 IF I=0 THEN I=1:VN=2
1870 FOR I=I TO LEN(SR$)
1880 CO$=MID$(SR$,I,1)
1890 IF CO$=" " THEN NEXT I
1900 IF CO$="=" THEN GOTO1940
1910 CN$=CN$+CO$
1920 NEXT I
1930 GOTO 2920
1940 FOR I=I+1 TO LEN(SR$) :CO$=MID$(SR$,I,1)
1950 IF CO$=" " AND(TY <> 5) THEN NEXT I
1960 IF CO$="'" THEN CO$=CHR$(34)
1970 IF CO$=";" THEN VN=0:GOTO2020
1980 IF CO$=CHR$(34) THEN CG$=CG$+CO$ :TY=5 :IF I=LEN(SR$) THEN2930  ELSE NEXT I
1990 CG$=CG$+CO$
2000 NEXT I
2010 GOTO 2920
2020 I=0 :  IF TY=5 THEN SP$(SP)=CN$:GOSUB 2580 :BC$=STR$(BZ)+" "+SB$(SP)+"="+CG$
:SP=SP+1
2030 I=0 :  IF TY <> 5 THEN RP$(RP)=CN$:GOSUB2490  :BC$=STR$(BZ)+" "+RB$(RP)+"="+
CG$:RP=RP+1
2040 GOSUB2710 :BZ=BZ+1:GOTO1700
2050 REM (*.......*)
2060 BC$=STR$(BZ)+" REM  "+SR$ :GOSUB 2710 : BZ=BZ+1: GOTO1700
2070 REM VAR
2080 VT=4:GOSUB2910  0=0
2090 IF I+LEN(Y$) > LEN(SR$) THEN 1700
2100 CO$="":CN$=""
2110 GOSUB2910
2120 IF I=0 THEN I=1 :VN=4
2130 FOR I=I TO LEN(SR$)
2140 CO$=MID$(SR$,I,1)
2150 IF CO$=" " THEN NEXT I
```

```
2160 IF CO$="," THEN 0$(Q)=CN$:Q=Q+1:CN$="":VN=0:NEXT I
2170 IF CO$=")" THEN 0$(Q)=CN$:GOTO 2200
2180 CN$=CN$+CO$:IF I>=LEN(SR$) THEN1700  ELSE VN=4:NEXT I
2190 GOTO 1700
2200 Y$="ARRAY":GOSUB2800 :IF I <> 0 THEN CN$="":GOTO 2250
2210 Y$="INTEGER":GOSUB2800 :IF I <> 0 THEN FOR I=0 TO Q:IP$(IP+I)=0$(I):GOSUB24
60 :NEXT I:VN=0:IP=IP+1+Q:Q=0:GOTO1700  ELSE 2220
2220 Y$="REAL":GOSUB2800 :IF I <> 0 THEN FOR I=0 TO Q:RP$(RP+I)=0$(I):GOSUB2490
:NEXT I:VN=0:RP=RP+1+Q:Q=0:GOTO1700  ELSE 2230
2230 Y$="STRING":GOSUB2800 :IF I <> 0 THEN FOR I=0 TO Q:SP$(SP+I)=0$(I):GOSUB258
0 :NEXT I:VN=0:SP=SP+1+Q:Q=0:GOTO1700  ELSE2890
2240 REM VAR (ARRAY)
2250 FOR I=I+5 TO LEN(SR$) :CO$=MID$(SR$,I,1)
2260 IF CO$=" "THEN NEXT I
2270 CN$=CN$+CO$
2280 IF CO$=")"THEN DI$=CN$:GOTO2310
2290 NEXT I
2300 GOTO2920
2310 Y$="INTEGER":GOSUB2800 :IF I <> 0 THEN3190
2320 Y$="REAL":GOSUB2800 :IF I <> 0 THEN3230
2330 Y$="STRING":GOSUB2800 :IF I <> 0 THEN3270
2340 GOTO 2890
2350 REM PROCEDURE
2360 I1=I+LEN(Y$):GOSUB2910 :GOSUB2950 :VT=6:VN=6
2370 YP$(YP)=CN$:YB$(YP)="GOSUB"+STR$(GZ):BZ=GZ:GZ=GZ+100:YP=YP+1:BC$=""
2380 GOSUB2950
2390 IF CN$="BEGIN"THEN3730
2400 IF CN$<>""THEN PRINT"NUR PROC.-RUMPF ZUL.":GOTO5070
2410 IF I1>=LEN(SR$) THEN GOSUB2610 :I1=1:GOTO2380
2420 BZ=BZ-1:BC$=STR$(BZ)+" RETURN":GOSUB2710 :BC$=""
2430 VN=0:BZ=12000:GOTO1700
2440 REM QUERVERWEIS PASCAL-BEZ. ZU BASIC-VAR.
2460 IB$(IP+I)="G"+CHR$(IP+I+65)
2470 RETURN
2490 RB$(RP+I)="R"+CHR$(RP+I+65)
2500 RETURN
2580 SB$(SP+I)="S"+CHR$(SP+I+65)+"$"
2590 RETURN
2600 REM LADE SR$ MIT AKT.SRC-ZEILE
2610 FOR I2=1 TO 5
2611 LS=LS+1 :IF LS=32768 THEN LS=-LS
2612 IF I2=4 THEN IF CHR$(PEEK(LS)) <> CHR$(34) THEN IF X9=1 THEN X9=2:RETURN EL
SE PRINT"";CHR$(34);" FEHLT":GOTO5070
2613 NEXT I2
2630 SR$=""
2635 IF LS=32768 THEN LS=-LS
2640 IF PEEK(LS) <> 0 THEN SR$=SR$+CHR$(PEEK(LS)):LS=LS+1:GOTO 2635
2650 LS=LS+1:IF LS=32768 THEN LS=-LS
2660 IF WE=1 THEN PRINT:IF X8=0 THENLPRINT
2670 PRINT SR$:WE=0
2680 IF X8=0 THEN LPRINT SR$
2685 IF SR$=""THEN2610
2690 RETURN
2750 NEXT I
2760 POKE BL,0:BL=BL+1:IF BL=32768 THEN BL=-BL
2770 PRINT"---";BC$:WE=1:IF X8=0 THEN LPRINT"---";BC$
2780 I=A :RETURN
2790 REM ZEICHEN-KETTEN-VERGLEICH
2800 FOR I=1 TO LEN(SR$)-LEN(Y$)+1
2810 IF Y$=MID$(SR$,I,LEN(Y$)) THEN  :RETURN
2820 NEXT I: I=0 :RETURN
2830 REM SUCHE 1.SCHLUESSELWORT IN SR$
2840 FOR I=1 TO LEN(SR$) : CO$=MID$(SR$,I,1)
2850 IF CO$=" " THEN NEXT I
2860 IF MID$(SR$,I,LEN(Y$))=Y$ THEN RETURN
2870 I=0 : RETURN
2880 REM FEHLER
2890 PRINT " NICHT VORGESEHEN":GOTO5070
2910 IF VN>0 THEN PRINT" : FEHLT":GOTO5070 ELSE RETURN
2920 PRINT"--FEHLER--":GOTO5070
2930 PRINT" ) FEHLT":GOTO5070
2940 REM WORT ISOLIEREN UND (*,...*) AUSBLENDEN
2950 CN$="":SE=0:CO$="":HK=0
2955 IF I1 > LEN(SR$) THEN RETURN
2960 FOR I1=I1 TO LEN(SR$) :CO$=MID$(SR$,I1,1):A=ASC(CO$):IF A=39 THEN A=34:CO$=
CHR$(A)
2970 IF(KO=1)AND(CO$=")")THEN IF MID$(SR$,(I1-1),1)="*"THEN KO=0:SE=1:RETURN
2980 IF((A=34)OR(A=39))AND(HK=0) THEN HK=1 :GOTO2990 :ELSE IF ((A=34)OR(A=39))A
ND(HK=1) THEN HK=0
2990 IF KO=1 THEN NEXT I1
3000 IF A=32 THEN IF CN$="" THEN IF TY <> 5 THEN NEXT I1
3010 IF HK=1 THEN CN$=CN$+CO$:NEXT
3020 IF CO$="(" AND(MID$(SR$,I1+1,1)="*") THEN KO=1:NEXT
3030 IF CO$=" " THEN CO$=""
3040 IF CO$="." THEN SE=1
3050 IF (A >64)AND(A <91) OR (A >47)AND(A <58) OR (A=39) OR (A=34) OR (A=46) OR
(A=35) THEN TY=5 :CN$=CN$+CO$ :NEXT ELSE IF CN$="" THEN NEXT ELSE TY=0 :RETURN
3060 RETURN
3070 REM ZUWEISUNG/OPERATOR :(),.: SUCHEN UND IN BC$ ABLEGEN
3080 IF I1 > LEN(SR$)THEN RETURN ELSE FOR I1=I1 TO LEN(SR$) :CO$=MID$(SR$,I1,1)
:A=ASC(CO$)
3090 IF I1 > LEN(SR$) THEN 2930
3100 IF CO$=" " THEN NEXT I1
3110 IF (CO$ >= "A")AND(CO$ <= "Z") OR (CO$ >="0")AND(CO$<="9") OR (A=34) OR (A=
39) THEN RETURN
3120 IF CO$="." THEN NEXT I1
3130 IF CO$=")" THEN SE=1 :RETURN
3140 IF (A=61) OR (A=60) OR (A=62) OR (A=40) OR (A=41) OR (A=44) OR (A=43) OR (A
=45) OR (A=42) OR (A=47) THEN SE=0
3145 IF (A=44)AND(AK=1)THEN CO$=","
3150 IF I1 =LEN(SR$) THEN 3170  ELSE A=ASC(MID$(SR$,I1+1,1))
3160 IF (A >59)AND(A <63) THEN CO$=CO$+CHR$(A) :I1=I1+1
3170 BC$=BC$+CO$ :GOSUB3580 :NEXT I1:RETURN
3180 REM BASIC-FELDER BILDEN UND DIM.
3190 FOR I=0 TO Q
3200 JP$(JP+I)=Q$(I):JB$(JP+I)="J"+CHR$(65+JP+I)
3210 BC$=STR$(DZ)+" DIM "+JB$(JP+I)+DI$:DZ=DZ+1:GOSUB2710
3220 NEXT I :JP=JP+1+Q:Q=0:VN=0:GOTO1700
3230 FOR I=0 TO Q
3240 PP$(PP+I)=Q$(I):PB$(PP+I)="P"+CHR$(65+PP+I)
3250 BC$=STR$(DZ)+" DIM "+PB$(PP+I)+DI$:DZ=DZ+1:GOSUB2710
3260 NEXT I :PP=PP+1+Q:Q=0:VN=0:GOTO1700
3270 FOR I=0 TO Q
3280 TP$(TP+I)=Q$(I):TB$(TP+I)="T"+CHR$(65+TP+I)+"%"
3290 BC$=STR$(DZ)+" DIM "+TB$(TP+I)+DI$:DZ=DZ+1:GOSUB2710
3300 NEXT I :TP=TP+1+Q:Q=0:VN=0:GOTO1700
3370 REM PASCALBEZEICHNER
3380 FOR I=0 TO RP-1:IF RP$(I)=CN$ THEN X$=RB$(I):RETURN ELSE NEXT I
3390 FOR I=0 TO IP-1:IF IP$(I)=CN$ THEN X$=IB$(I):RETURN ELSE NEXT I
3400 FOR I=0 TO SP-1:IF SP$(I)=CN$ THEN X$=SB$(I):RETURN ELSE NEXT I
3410 X$="":RETURN
3420 REM PASCAL-ARRAY-BEZEICHNER
3430 FOR I=0 TO PP-1:IF PP$(I)=CN$ THEN X$=PB$(I):RETURN ELSE NEXT I
3440 FOR I=0 TO JP-1:IF JP$(I)=CN$ THEN X$=JB$(I):RETURN ELSE NEXT I
3450 FOR I=0 TO TP-1:IF TP$(I)=CN$ THEN X$=TB$(I):RETURN ELSE NEXT I
3460 X$="":RETURN
3470 REM PASCAL-FUNKTIONS-BEZEICHNER
3480 RESTORE:FOR I=0 TO 29:READ FP$:IF FP$=CN$ THEN3485 ELSE NEXT I
3482 X$="":RETURN
3485 FOR I=I TO I+29:READ X$:NEXT I
3490 RETURN
3500 REM PROCEDURE-BEZEICHNER
3510 FOR I=0 TO YP-1:IF YP$(I)=CN$ THEN X$=YB$(I):RETURN ELSE NEXT I
3520 X$="":RETURN
3530 REM BEGINNT CN$ MIT EINER ZIFFER ?
3540 IF CN$ <> "" THEN A=ASC(LEFT$(CN$,1)) ELSE TY=10 :RETURN
3550 IF((A > 47)AND(A < 58)) OR (A=34) OR (A=39) THEN BC$=BC$+CN$:TY=0 ELSE TY=1
0
```

```
3560 RETURN
3570 REM QUADRAT (SQR)
3580 IF A1=0 THEN 3620
3590 IF CO$="(" THEN KL=KL+1 ELSE IF CO$=")" THEN KL=KL-1 : IF KL=0 THEN BC$=BC$
+"↑2" : A1=0
3600 RETURN
3610 REM PRINT(...);
3620 IF A2=0 THEN RETURN
3630 IF CO$="(" THEN KM=KM+1 :GOSUB3670 ELSE IF CO$=")" THEN KM=KM-1 ELSE RETURN

3640 IF KM=0 THEN GOSUB3680 :A2=0:AK=0:IF A3=0 THEN BC$=BC$+";" ELSE A3=0
3650 RETURN
3660 REM UP: "(....")" AUS BC$ ENTFERNEN
3670 IF KM=1 THEN 3680 ELSE RETURN
3680 BC$=MID$(BC$,1,LEN(BC$)-1) :RETURN
3690 REM --PROGRAMMRUMPF--
3700 VT=7 : GOSUB2910
3710 I1=I+LEN(Y$)
3720 REM --HAUPTSCHLEIFE--
3730 IF I1 >= LEN(SR$) THEN IF(SE=0)AND(BC$ <> "")AND(SS=0)THEN GOSUB2950:SE=1:G
OTO4310 ELSE GOSUB2610 :I1=1
3740 GOSUB2950
3750 IF BC$="" THEN BC$=STR$(BZ)+" " :BZ=BZ+1
3760 REM READ/WRITE
3770 IF (CN$="READ") OR (CN$="READLN") THEN BC$=BC$+"INPUT":A3=1:A2=1:GOTO4420
3775 IF CN$="READ#"THEN BC$=BC$+"INPUT#-1,":A3=1:A2=1:GOTO4420
3780 IF CN$="WRITE" THEN A2=1:A3=0:KM=0:AK=1: IF X7$="J"THEN BC$=BC$+"LPRINT" GO
TO4420ELSE BC$=BC$+"PRINT":GOTO4420
3790 IF CN$="WRITELN" THEN A2=1:A3=1:KM=0:AK=1: IF X7$="J"THEN BC$=BC$+"LPRINT"
GOTO4420 ELSE BC$=BC$+"PRINT":GOTO4420
3795 IF CN$="WRITE#"THEN BC$=BC$+"PRINT#-1,":A2=1:A3=0:KM=0:GOTO4420
3800 REM BEGIN/END
3810 IF CN$="BEGIN" THEN BE(ST)=BE(ST)+1 :GOTO3730
3820 IF CN$ <> "END" THEN 3910
3825 IF (ST(=0)AND(BE(0))=1) THEN3730
3830 IF (BE(0)(=0)AND(VT=6)AND(ST(=0) THEN2420
3840 IF (ST)0)AND(ST(ST-1)=6)THEN ST=ST-1:GOTO3730
3850 IF (BE(ST) (= 0) OR (ST <= 0) THEN PRINT"BEGIN-END-FEHLER":GOTO5070
3860 BE(ST)=BE(ST)-1
3870 IF BE(ST) <> 0 THEN 3730
3875 I1=LEN(SR$)
3880 ON ST(ST-1) GOTO 4495 ,3885 ,3730 ,4535 ,4535
3885 BZ=BZ-1
3890 BC$=STR$(BZ)+" NEXT":GOSUB 2710 :BC$="":BZ=BZ+1:ST=ST-1:GOTO3730
3900 REM WHILE/DO
3910 IF CN$="WHILE" THEN BC$=BC$+"IF" :ST(ST)=1 :GOTO4420
3920 IF CN$ <> "DO" THEN 3980
3930 IF ST(ST) <> 1 THEN3980
3940 BC$=BC$+"THEN"+STR$(GZ)
3950 SZ(ST)=BZ :ST=ST+1 :BZ=GZ :GZ=GZ+100
3960 GOSUB2710 :BC$="" :GOTO 3730
3970 REM FOR/TO,DOWNTO/DO
3980 IF CN$="FOR"THEN BC$=BC$+"FOR":ST(ST)=2:GOTO3730
3990 IF CN$="TO"THEN FT=1:BC$=BC$+"TO":GOTO3730
4000 IF CN$="DOWNTO"THEN FT=2:BC$=BC$+"TO":GOTO3730
4010 IF CN$ <> "DO"THEN4050
4020 IF (ST(ST)=2)AND(FT=2)THEN BC$=BC$+"STEP-1"
4030 ST=ST+1:GOSUB2710 :BC$="":BZ=BZ+1:GOTO3730
4040 REM REPEAT/UNTIL
4050 IF CN$="REPEAT"THEN ST(ST)=3:SZ(ST)=BZ:ST=ST+1:GOTO3730
4060 IF CN$="UNTIL"THEN BC$=BC$+"IF NOT(":UE=1:GOTO4420
4070 REM IF/THEN/ELSE
4080 IF CN$="IF"THEN BC$=BC$+"IF":ST(ST)=4:SS=1:GOTO4420
4090 IF CN$="THEN"THEN SW$(ST)=MID$(BC$,10,LEN(BC$)):BC$=BC$+"THEN"+STR$(GZ):SS=
0:GOTO3950
4095 IF (CN$="ELSE")AND(LEN(BC$) > 8)THEN GOSUB2710:BC$=STR$(BZ)+" GOTO"+STR$(SZ
(ST-1)):GOSUB2710:ST=ST-1:BZ=SZ(ST):BC$=STR$(BZ)+" ":BZ=BZ+1
4100 IF CN$="ELSE"THEN BC$=BC$+"IF NOT("+SW$(ST)+")THEN"+STR$(GZ):GOTO3950
4110 REM CASE/OF
4120 IF CN$ <> "CASE"THEN4190
4125 SS=1
4130 GOSUB2950 :GOSUB3390
4140 IF X$ <> ""THEN4160
4150 GOSUB3440 :IF X$ <> ""THEN4160 ELSE4400
4160 SW$(ST)="("+X$+"=":GOSUB2950
4170 IF CN$ <> "OF"THEN4400
4180 ST(ST)=6:ST=ST+1:GOTO3730
4190 IF ST <= 0 THEN 4310
4200 IF ST(ST-1) <> 6 THEN4310
4210 A=ASC(LEFT$(CN$,1))
4220 IF NOT((A=34)OR(A=39)OR((A)47)AND(A(58))) THEN4310
4225 SS=0
4230 BC$=BC$+"IF"
4240 BC$=BC$+SW$(ST-1)+CN$+")"
4250 FOR I1=I1 TO LEN(SR$):CO$=MID$(SR$,I1,1)
4260 IF CO$=" "THEN NEXT I1
4270 IF CO$=","THEN BC$=BC$+"OR":GOSUB2950 :GOTO4240
4280 IF CO$=":"THEN BC$=BC$+"THEN"+STR$(GZ):ST(ST)=5:GOTO3950
4290 PRINT"CASE-MARKE-FEHLER":GOTO5070
4300 REM SEQUENZ (EINFACHE ANWEISUNGEN)
4310 IF CN$="" THEN4440 ELSE GOSUB3540:IF TY=0 THEN4420
4320 GOSUB3380
4330 IF X$ <> "" THEN BC$=BC$+X$ :GOTO4420
4340 GOSUB3430
4350 IF X$ <> "" THEN BC$=BC$+X$ :GOTO4420
4360 GOSUB3510 :IF X$ <> ""THEN BC$=BC$+X$:GOTO4420
4370 IF CN$="E" THEN BC$=BC$+CN$ : GOTO4420
4380 IF CN$="SQR" THEN A1=1 :GOTO 4420
4390 GOSUB3480
4395 IF (X$="")AND(SE=1) THEN BC$="":GOTO3730
4400 IF X$="" THEN IF CN$="END," THEN BC$=STR$(BZ-1)+" END":GOSUB2710 : PRINT"**
***PROGRAMMENDE*****":POKE BL,255 :GOTO5040 ELSE PRINT"--",CN$,"--UNKLAR" GOTO50
70
4410 BC$=BC$+X$
4420 GOSUB3080:IF I1=LEN(SR$)THEN3740
4430 REM SEMIKOLON
4440 IF SE=0 THEN 3730
4445 A2=0:A3=0
4450 SE=1:IF UE=1 THEN BC$=BC$+")"THEN"+STR$(SZ(ST-1)-1):UE=0:GOSUB2710 :BC$="":S
T=ST-1:BZ=BZ+1:GOTO3730
4460 IF ST <= 0 THEN GOSUB2710 :BC$="" :GOTO3730
4470 IF BE(ST) <> 0 THEN GOSUB 2710 :BC$="" :GOTO3730
4480 GOSUB2710
4490 ON ST(ST-1) GOTO 4500 ,3890 ,4530 ,4540 ,4540
4495 BZ=BZ-1
4500 BC$=STR$(BZ)+" GOTO"+STR$(SZ(ST-1)-1) :GOSUB2710 :BC$=""
4510 ST=ST-1:BZ=SZ(ST)
4520 GOTO3730
4530 BC$="":GOTO3730
4535 BZ=BZ-1
4540 BC$=STR$(BZ)+" GOTO"+STR$(SZ(ST-1)):GOSUB2710 :BC$="":ST=ST-1:BZ=SZ(ST):GOT
03730
4549 REM TABELLEN UND MASCHINEN-EBENE
4550 DATA"SQRT","ARCTAN","LN","TRUNC","ROUND","ABS","SIN","COS","EXP","TAN","NOT
","AND","OR","PEEK","POKE","INP","OUT","LENGTH"
4560 DATA"SUBSTR","SET","RESET","CLS","POINT","RND","TAB","STR","POS","USING","O
RD","CHR"
4570 DATA"SQR","ATN","LOG","FIX","INT","ABS","SIN","COS","EXP","TAN","NOT","AND"
,"OR","PEEK","POKE","INP","OUT","LEN"
4580 DATA"MID$","SET","RESET","CLS","POINT","RND","TAB","STR$","STRING$","USING",
"ASC","CHR$"
4590 END
5000 PRINT"PASCAL:'EDIT,CSAVE,CLOAD'"
5010 B=PEEK(16633)+256*PEEK(16634)-2:BH=B/256:BL=B-INT(BH)*256
5020 POKE16446,BL:POKE16447,BH
5030 POKE16548,PEEK(16446):POKE16549,PEEK(16447)+1:GOTO5060
5040 IF X8 <> 2 THENPOKE16548,PEEK(16446):POKE16549,PEEK(16447)+1 END
5050 PRINT"BASIC:'/LIST,CSAVE,RUN'":POKE16526,240:POKE16527,66:PRINT USR(0):END
5060 POKE16526,70:POKE16527,67:PRINT USR(0):END
5070 X8=7:GOTO5040
```

Der Quell-Code (das Pascal-Programm) kann nun korrigiert und erneut übersetzt werden. Der Compiler-Output kann ebenfalls über Drucker ausgegeben werden. Dieser Output macht den Zusammenhang zwischen Pascal-Programm und dem erzeugten Basic-Code deutlich. Er stellt somit ein sehr nützliches Hilfsmittel zur Fehlersuche während der Laufzeit dar.

Beispiel: *Bild 25*; die erzeugten Basic-Zeilen werden durch '---' angezeigt.

3: Übersetzt das Pascal-Programm in den Basic-Code und legt diesen im Hauptspeicher ab. Dabei wird der Pascal-Quell-Code überschrieben. Vor der Übersetzung kann entschieden werden:

```
OUTPUT AUF DRUCKER (J,N):? N
DATENTYP 'REAL' MIT DOPPELTER
GENAUIGKEIT (J,N):? J
```

Nach Beendigung des Übersetzungsvorgangs und der Sortierung des Basic-Codes nach Zeilennummern meldet sich der Compiler wie in *Bild 26* oben.
Der generierte Basic-Code kann durch Eingabe des Befehls RUN ausgeführt und mit den bekannten Befehlen des Basic-Level II getestet werden.

```
>RUN
VERZINSUNG
ANFANGS-KAPITAL IM VOLLEN
DM:? 150 000
JAHRE:? 5
ZINSSATZ IN PROZENT:? 9.8
ENDKAPITAL IN DM: 239388
READY
```

Der Bezug zum Pascal-Quell-Programm kann über den Compiler-Output hergestellt werden.
Ist der erzeugte Basic-Code fehlerfrei, kann er aufgezeichnet werden. Bei Bedarf (auch ohne Vorhandensein des Compilers) kann er erneut eingelesen und ausgeführt werden.

4: Löscht den freien Arbeitspeicher und schaltet in die Betriebsart Pascal: 'EDIT, CSAVE, CLOAD' (Neuerstellung)

Literatur

[1] *Doberkat, E.-E., Rath, P., Rupietta, W.:* Pascal-Programmier-Kursus der Fernuniversität Hagen.
[2] *Enger, E.:* Programmieren mit Pascal, Elektronik 1980, Hefte 4...8.
[3] *Radio Shack:* Handbuch für Basic Level II.
[4] *Baumann, R.:* Programmieren mit Pascal. Vogel-Verlag, Würzburg, 1981.
[5] Roeckrath, L.: Der geknackte TRS-80. mc 1981, Heft 2.

Übertragbarkeit auf andere Rechnertypen

Für die Übertragung des Pascal-Compilers auf einen anderen Rechnertyp sind genaue Kentnisse über die Erweiterbarkeit des Basic-Interpreters sowie dessen Speicherverwaltung erforderlich.
Neben den interpreterspezifischen RAM-Adressen (siehe [5]), die man anspricht über PEEK und POKE, werden zwei Maschinen-Programme angerufen.
Das erste Maschinen-Programm, unter DATA-Zeile 3, wird zwischen hex 4040 und hex 405C geladen. Es setzt den Programmzeiger auf Compileranfang, lädt den Befehl „RUN 9" in den I/O-Puffer und verzweigt zum Interpreter mit der Ausführung des Befehls.
Das zweite Maschinen-Programm nimmt mit einem Programmsegment, aufgerufen über Programmzeile 5060, das Setzen der Zeiger für Pascal-Quell-Code, Basic-Code und dessen Variablen-Tabelle vor. Das andere Programmsegment wird über Zeile 5050 aufgerufen. Dieser Aufruf erfolgt nach Beendigung der Übersetzungsphase (in Basic-Code-ASCII-Format). Dieses Modul verdeckt den Compiler, trennt die Interpreter-Hauptschleife auf, lädt schrittweise jede vom Compiler generierte Code-Zeile in den I/O-Puffer. Anschließend verzweigt das Programm zur Interpreter-Routine „I/O-Puffer in Zwischen-Code umwandeln und im Programmtext ablegen".
Ist das Ende des generierten Codes erreicht, wird die Hauptschleife wieder geschlossen und zum Basic-Warm-Start verzweigt. Der Interpreter meldet sich dann mit READY.

H. u. M. Fillinger

Was ist „Simon's Basic"?

Eine neue Programmiersprache für den C-64 macht von sich reden, genauer gesagt, ein gegenüber dem ansonsten eher spartanischen Commodore-Basic ein leistungsfähiger Basic-Dialekt, der auch Grafik- und Sound-Befehle besitzt.
CALL wirkt wie GOTO, läßt aber Sprünge auf Namen statt auf Zeilennummern zu. AUTO hilft bei der Programmeingabe durch automatisches Erzeugen von Zeilennummern, z. B. in Zehnerschritten. CGOGO gestattet als Sprungziel einen arithmetischen Ausdruck, z. B. A*12. ON ERROR springt zu vordefinierbaren Zeilennummern abhängig von der Fehlerart. LOCAL und GLOBAL gestatten lokale und globale Variablendefinitionen wie z. B. in Algol oder Pascal.
Für die Erstellung von Grafiken braucht man keine POKE-Befehle mehr: HIRES schaltet auf hochauflösende Grafik um, MULTI auf mehrfarbige HIRES-Grafik. ARC zeichnet einen Bogen, CIRCLE einen Kreis, REC ein Rechteck. Die Befehle DRAW und ROT ähneln dem Zeichen von Shapes beim Apple-II. Mit TEXT kann man Buchstaben auf den Grafik-Schirm zaubern.
Auch Musik kann man ohne POKEs ertönen lassen. ENVELOPE setzt die Parameter für ein bestimmtes Instrument, MUSIC speichert eine Melodie im RAM, PLAY spielt dann diese Melodie. VOL stellt die gewünschte Lautstärke ein, WAVE die Wellenform. Simon's Basic besitzt eine Reihe weiterer Befehle, wie RENUMBER, EXOR, FRAC, MERGE und viele andere. Der neue Interpreter wird als Diskette oder Steckkassette (Cartridge) auch in Deutschland von Commodore-Händlern verkauft; die Entwicklung wurde von Commodore in England durchgeführt. Insgesamt stehen dem Benutzer mit Simon's Basic 114 neue Befehle zur Verfügung – und aus dem C-64 wird damit ein Gerät, dessen grafische und musikalische Fähigkeiten sich auch dem Basic-Programmierer erschließen, der mit Assembler und POKE-Befehlen weniger im Sinn hat.

Fe.