

```

100 REM *** ANALOGUHR *** <038>
110 REM MARC MICHAELIS <188>
120 REM PFARRHOEHE 11 <132>
130 REM 8507 OBERASBACH <035>
140 REM <202>
150 IF A=0 THEN A=1:LOAD"UHR PRG.",8,1 <165>
160 FOR J=0 TO 83 <161>
170 IF J=0 OR J=60 THEN X1=0:Y1=20 <164>
180 IF J=15 OR J=66 THEN X1=0:Y1=0 <079>
190 IF J=30 OR J=72 THEN X1=20:Y1=0 <050>
200 IF J=45 OR J=78 THEN X1=20:Y1=20 <101>
210 GOSUB 460 <018>
220 READ X2,Y2 <211>
230 GOSUB 370 <046>
240 W=40960+64*J <165>
250 GOSUB 480 <090>
260 NEXT <016>
270 FOR J=46336 TO 46591 <000>
280 READ A:POKE J,A <192>
290 NEXT <046>
300 FOR J=832 TO 880:READ A:POKE J,A:NEXT <234>
310 SYS 832 <236>
320 END <068>
330 REM PLOT <206>
340 Q=Y*3+INT(X/8)+832:Q1=2+(7-(X-INT(X/8) <156>
*8)) <108>
350 POKE Q,PEEK(Q)OR Q1 <164>
360 RETURN <066>
370 REM LINIE ZIEHEN (X1,Y1,X2,Y2) <231>
380 IF ABS(X2-X1)<5 THEN 420 <253>
390 FOR I=X1 TO X2 STEP(X2-X1)/20 <251>
400 Y=INT((Y2-Y1)/(X2-X1)*(I-X1)+Y1) <039>
410 X=INT(I):GOSUB 330:NEXT:RETURN <204>
420 FOR I=Y1 TO Y2 STEP(Y2-Y1)/20 <081>
430 X=INT(((X2-X1)/(Y2-Y1)*(I-Y1)+X1)) <073>
440 Y=INT(I):GOSUB 330:NEXT:RETURN <042>
450 REM LOESCHEN SPRITE <066>
460 FOR I=832 TO 896:POKE I,0:NEXT:RETURN <133>
470 REM UEBERTRAGEN SPRITE
480 FOR I=0 TO 63:POKE W+I,PEEK(832+I):NEXT:RETURN <004>
490 REM DATA FUER ZEIGER ( X UND Y ) <074>
500 DATA 0,0,2,0,4,1,6,1,8,2,10,3,12,4,13, <123>
5,15,7,16,8,17,10,18,12,19,14,19,16 <143>
510 DATA 20,18
520 DATA 20,0,20,2,19,4,19,6,18,8,17,10,16 <243>
,12,15,13,13,15,12,16,10,17,8,18,6,19 <228>
530 DATA 4,19,2,20
540 DATA 20,20,18,20,16,19,14,19,12,18,10, <142>
17,8,16,7,15,5,13,4,12,3,10,2,8,1,6 <188>
550 DATA 1,4,0,2
560 DATA 0,20,0,18,1,16,1,14,2,12,3,10,4,8 <141>
,5,7,7,5,8,4,10,3,12,2,14,1,16,1,18,0 <078>
570 DATA 0,10,3,10,5,11,7,13,9,15,10,17 <209>
580 DATA 10,0,10,3,9,5,7,7,5,9,3,10 <214>
590 DATA 20,10,17,10,15,9,13,7,11,5,10,3
600 DATA 10,20,10,17,11,15,13,13,15,11,17, <154>
10 <117>
610 REM ZIFFERBLATT SPRITES
620 DATA 0,0,255,0,7,8,0,24,8,0,32,0,0,224 <075>
,85,1,16,37,2,0,85,4,84,0,4,36,0,8
630 DATA 84,0,8,0,0,24,0,0,37,64,0,32,128, <142>
0,65,64,0,64,0,0,64,0,0,128,0,0,128
640 DATA 0,0,128,0,0,138,128,0,239,128,0,0, <050>
,112,0,0,12,0,0,2,0,0,3,128,0,4,64
650 DATA 0,0,32,0,16,16,0,16,8,0,16,4,0,0, <074>
4,0,0,6,0,0,169,0,0,161,0,0,160,128
660 DATA 0,0,128,0,0,128,0,0,64,0,0,64,0,0, <230>
,64,0,84,64,200,233,0,8,138,128,0
670 DATA 128,0,0,128,0,0,128,0,0,85,80,0,8 <198>
5,80,0,73,80,0,32,0,0,36,0,0,24,0
680 DATA 0,8,170,0,8,170,0,4,74,0,2,0,42,1 <035>
,16,42,0,224,18,0,32,0,0,24,8,0,7
690 DATA 8,0,0,255,141,0,85,192,0,84,64,0, <068>
0,64,0,0,64,0,0,64,2,160,128,2,160
700 DATA 128,2,64,128,0,1,0,0,9,0,0,6,0,40 <104>
,4,0,40,4,0,16,24,0,0,32,0,4,64,0
710 DATA 3,128,0,2,0,0,12,0,0,112,0,0,128, <048>
0,0,0 <032>
720 REM DATEN FUER ABSPEICHERROUTINE
730 DATA 169,54,133,1,162,8,32,186,255,169 <168>
,9,162,104,160,3,32,189,255,169,64
740 DATA 133,251,169,156,133,252,169,251,1 <008>
62,0,160,182,32,216,255,169,55,133 <088>
750 DATA 1,96,65,78,65,76,79,71,85,72,82

```

© 64'er

Listing 2. Das Basic-Programm »UHR SPRITEGEN« erzeugt die nötigen Sprites der Uhr, verbindet diese mit dem Maschinenprogramm und speichert beides als »ANALOGUHR« auf Diskette

Das eigentliche Steuerprogramm ist eine neue Interrupt-routine. Zuerst wird die Stunde gelesen, von BCD in binär umgewandelt, und daraus die vorläufige Spritenummer für den Stundenzeiger berechnet. Außerdem wird die Spriteposition des Stundenzeigers berechnet (Stunde < 3 $\triangleq$ rechtes oberes Viertel, Stunde < 6 $\triangleq$ rechtes unteres Viertel etc.). Dann werden die Minuten gelesen, umgerechnet und die Position des Minutenzeigers berechnet. Wenn die Minuten größer oder gleich 30 sind, wird der Stundenzeiger um eine halbe Stunde vorgerückt. Schließlich wird noch der Sekundenzeiger nach der gleichen Methode gesetzt. Bevor zur IRQ-Routine des Betriebssystems gesprungen wird, muß noch das Zehntelsekunden-Register gelesen werden, um die Uhr wieder freizugeben.

(M. Michaelis/og)

Löschen ohne Verluste

Ein dimensioniertes Feld läßt sich normalerweise nicht ohne den Verlust aller Variablenwerte aus dem Speicher entfernen. Dieses Programm macht's möglich.

Ein einmal angelegtes Variablenfeld läßt sich beim Commodore 64 nicht wieder löschen; es sei denn, man benutzt den Befehl CLR, der dann aber sämtliche Variablen löscht. Mit dem Programm »CLEAR« (Listing 1) können Sie ein einzelnes Variablenfeld löschen. Somit ist es möglich, diese Variable neu zu dimensionieren. Mit SYS 49152 wird der CLR-Befehl erweitert. Folgt nun nach CLR ein Variablenname (zum Beispiel CLR A\$), so wird diese Variable (falls vorhanden) aus dem Array-Bereich entfernt. Die Funktionsweise des Programms kann dem Assemblerlisting entnommen werden (Listing 2).

Wie verwaltet der Computer nun ein Variablenfeld? Im Bild rechts sehen Sie, wie die drei möglichen Variablentypen im Speicher abgelegt werden. In den ersten beiden Bytes steht der Variablenname. Besteht der Name aus nur einem Buchstaben, so wird für den zweiten Namen der Wert 0 eingesetzt. Die drei Variablentypen erkennt der Computer daran, das bei einer Stringvariablen im zweiten Byte das Bit 7 gesetzt ist. Zu dem ASCII-Wert des Variablennamens wird also der Wert 128 dazugezählt. Bei einer Intervariablen ist im ersten und zweiten Byte das Bit 7 gesetzt. In den nächsten beiden Bytes steht die Anzahl der Bytes, die das Feld benötigt. Danach folgt die Anzahl der Dimensionen und die Anzahl der einzelnen Felder. Zum Schluß finden wir noch die Variablen selbst. Wie im Bild zu sehen ist, belegt eine Stringvariable 3 Byte, eine Intervariable 2 Byte und eine Fließkommavariablen 5 Byte. Man kann also Speicherplatz sparen, wenn nach Möglichkeit Intervariablen benutzt werden. Dies ist aber nur bei Variablenfeldern der Fall, weil bei nichtdimensionierten Variablen immer 7 Byte pro Variable belegt werden.

(H. Kunz/og)

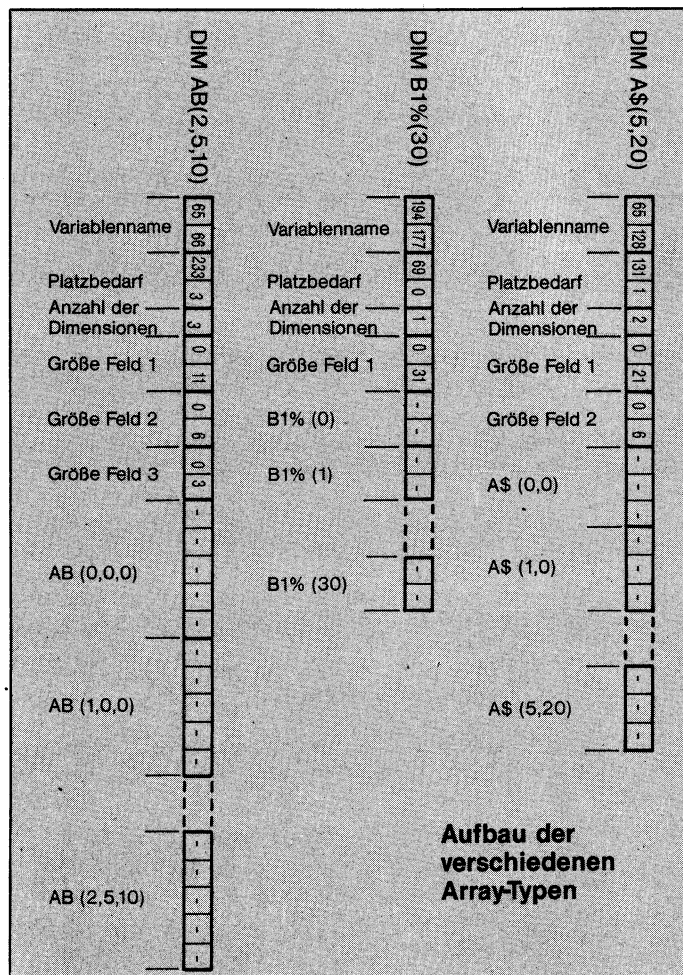
programm : clear-dim c000 c0de

```

c000 : a9 0b 8d 08 03 a9 c0 8d 2f
c008 : 09 03 60 20 73 00 c9 9c 47
c010 : f0 0f 20 79 00 4c e7 a7 10
c018 : 4c 08 af 20 60 a6 4c ae 22
c020 : a7 a9 00 85 f7 85 f8 20 1c
c028 : 73 00 f0 ef 20 13 b1 90 58
c030 : e7 20 79 00 85 f7 20 73 05
c038 : 00 f0 2b c9 24 f0 1e c9 8a
c040 : 25 f0 0f 85 f8 20 73 00 b0
c048 : f0 1c c9 24 f0 0f c9 25 36
c050 : d0 f3 20 73 00 a9 80 05 ea
c058 : f7 85 f7 d0 03 20 73 00 29
c060 : a9 80 05 f8 85 f8 a5 2f bf
c068 : a6 30 85 f9 86 fa a5 fa 94
c070 : c5 32 d0 06 a5 f9 c5 31 e7
c078 : f0 34 a0 00 b1 f9 c5 f7 9c
c080 : d0 07 c8 b1 f9 c5 f8 f0 d0
c088 : 06 20 c6 c0 4c 6e c0 a5 ee
c090 : f9 a6 fa 85 f7 86 f8 20 23
c098 : c6 c0 a5 fa c5 32 d0 11 da
c0a0 : a5 f9 c5 31 d0 0b a5 f7 c5
c0a8 : 85 31 a5 f8 85 32 4c ae c7
c0b0 : a7 a0 00 b1 f9 91 f7 e6 b7
c0b8 : f7 d0 02 e6 f8 e6 f9 d0 c5
c0c0 : d9 e6 fa 4c 9a c0 a0 03 8d
c0c8 : b1 f9 aa 88 b1 f9 18 65 48
c0d0 : f9 90 02 e6 fa 85 f9 8a 48
c0d8 : 18 65 fa 85 fa 60 ff fc bf

```

Listing 1. »CLEAR-DIM« bitte mit dem MSE eingeben



```

10 " ; =====
20 " ;
30 " ; PROGRAMM: CLEAR
40 " ;
50 " ; LOESCHEN EINES EINZELNEN
60 " ; VARIABLENFELDES MIT :
70 " ; CLR X
80 " ;
90 " ; =====
100 " ; .BA 49152
105 " ; LDA #0B ; VEKTOREN AUF
110 " ; STA #0308 ;
115 " ; LDA #C0 ; NEUE ROUTINE
120 " ; STA #0309 ;
125 " ; RTS ;
130 " ; =====
135 " ; JSR #0073 ; NAECHSTES ZEICHEN
140 " ; CMP #9C ; IST ES CLR
145 " ; BEQ JA ;
150 " ; JSR #0079 ; LETZTES ZEICHEN HOLEN
155 " ; JMP #A7E7 ; BEFEHL AUSFUEHREN
160 " ; ERROR JMP #AF08 ; SYNTAX ERROR AUSGEBEN
165 " ; NCLR JSR #A660 ; NORMALEN CLR BEFEHL AUSFUEHREN
170 " ; JMP #A7AE ; ZUR INTERPRETERSCHLEIFE
175 " ; JA LDA #00 ;
180 " ; STA #F7 ; VARIABLENNAME AUF
185 " ; STA #F8 ; NULL SETZEN
190 " ; JSR #0073 ; NAECHSTES ZEICHEN HOLEN
195 " ; BEQ NCLR ; NICHTS, DANN NORMALEN CLR AUSFUEHREN
200 " ; JSR #B113 ; PRUEFT AUF BUCHSTABE
205 " ; BCC ERROR ; KEIN BUCHSTABE, DANN FEHLER
210 " ; JSR #0079 ; LETZTES ZEICHEN NACH ACCU
215 " ; STA #F7 ; UND NACH #F7
220 " ; JSR #0073 ; NAECHSTES ZEICHEN
225 " ; BEQ X2 ; NICHTS, DANN NUMERISCHE VARIABLE
230 " ; CMP #* ; STRINGVARIABLE ?
235 " ; BEQ STRING
240 " ; CMP #* ; INTEGERVARIABLE
245 " ; BEQ INTEGER
250 " ; STA #F8 ; ZEICHEN NACH #F8
255 " ; JSR #0073 ; NAECHSTES ZEICHEN
260 " ; BEQ X2 ; NICHTS, DANN NUMERISCHE VARIABLE
265 " ; CMP #* ; STRINGVARIABLE ?
270 " ; BEQ STRING
275 " ; CMP #* ; INTEGERVARIABLE ?
280 " ; BNE LLX ;
285 " ; JSR #0073 ; NAECHSTES ZEICHEN
290 " ; LDA #80 ; BIT 7 IM 1. BYTE
295 " ; ORA #F7 ; DES VARIABLENNAMEN
300 " ; STA #F7 ; SETZEN
305 " ; BNE SPRUNG
310 " ; JSR #0073 ; NAECHSTES ZEICHEN
315 " ; LDA #80 ; BIT 7 IM 2. BYTE
320 " ; ORA #F8 ; DES VARIABLENNAMEN
325 " ; STA #F8 ; SETZEN
330 " ; X2 LDA #2F ;
335 " ; LDX #30 ; ARRAYANFANG NACH
340 " ; STA #F9 ;
345 " ; STX #FA ;
350 " ; LDA #FA ;
355 " ; CMP #32 ; #F9/#FA MIT ARRAY ENDE
360 " ; BNE N01 ;
365 " ; LDA #F9 ; VERGLEICHEN
370 " ; CMP #31 ;
375 " ; BEQ NOFOUND ;
380 " ; LDY #400 ; ERSTES BYTE DES
385 " ; LDA (#F9),Y ; VARIABLENNAMENS
390 " ; CMP #F7 ; MIT GESUCHTER VARIABLE
395 " ; BNE NO ; VERGLEICHEN
400 " ; INY ;
405 " ; LDA (#F9),Y ; ZWEITES BYTE
410 " ; CMP #F8 ; VERGLEICHEN
415 " ; BEQ FOUND ;
420 " ; JSR PLUS ; NAECHSTE VARIABLE
425 " ; JMP LOOP ;
430 " ; LDA #F9 ; VARIABLE GEFUNDEN
435 " ; LDX #FA ; ANFANG DER GESUCHTEN VARIABLE
440 " ; STA #F7 ; NACH #F7/#F8 SPEICHERN
445 " ; STX #F8 ;
450 " ; JSR PLUS ; ANFANG DER NAECHSTEN VARIABLEN
455 " ; LDA #FA ;
460 " ; CMP #32 ;
465 " ; BNE VERSCH ; ENDE DES ARRAYBEREICHS ?
470 " ; LDA #F9 ;
475 " ; CMP #31 ;
480 " ; BNE VERSCH ;
485 " ; LDA #F7 ; ZEIGER ARRAY ENDE
490 " ; STA #31 ;
495 " ; LDA #F8 ; SETZEN
500 " ; STA #32 ;
505 " ; NOFOUND JMP #A7AE ; ZUR INTERPRETERSCHLEIFE
510 " ; VERSCH LDY #400 ;
515 " ; LDA (#F9),Y ;
520 " ; STA (#F7),Y ; ANFANG DER NAECHSTEN VARIABLEN
525 " ; INC #F7 ; NACH ANFANG DER GESUCHTEN VARIABLEN
530 " ; BNE N1 ; VERSCHIEBEN, BIS ARRAY ENDE
535 " ; INC #F8 ; ERREICHT IST.
540 " ; INC #F9 ;
545 " ; BNE N2 ;
550 " ; INC #FA ;
555 " ; JMP N2 ;
560 " ; PLUS LDY #403 ;
565 " ; LDA (#F9),Y ;
570 " ; TAX ;
575 " ; DEY ;
580 " ; LDA (#F9),Y ;
585 " ; CLC ; ANFANG DER NAECHSTEN
590 " ; ADC #F9 ;
595 " ; BCC N3 ; VARIABLEN ERRECHNEN
600 " ; INC #FA ;
605 " ; STA #F9 ; UND NACH #F9/#FA SPEICHERN
610 " ; TAX ;
615 " ; CLC ;
620 " ; ADC #FA ;
625 " ; STA #FA ;
630 " ; RTS ;

```

Listing 2. Assembler-Listing zu »CLEAR-DIM«