BASIC/DEBUG für ES4.0

BASIC/DEBUG ist ein kleines (nur 2048 Byte!) und extrem leistungsfähiges TINY-BASIC. Es umfasst ein Basic mit Editor/Debugger, Hex, peek/deek und poke/doke (via @ und ^), Rechnen mit Punkt- vor Strichrechnung.

Das Original ist im ROM des Zilog Z8671 enthalten, Details siehe BASIC/DEBUG.

angepasst an Tiny ES4.0, V. Pohlers Nov. 2021:

- Speicherbereich 8000-87FF, Start auf 8000h
- USR-Funktionen zum Laden/Speichern
- neue Funktion RND(n)

Download

- z8671basic_es40.zip

Binary, Quellen, Basic-Converter (Perl)

Beschreibung

siehe das sehr gute Handbuch von Zilog

z8671_basic_debug.pdf

(der Hochpfeil in der Doku ist das Zeichen „^“) Das meiste daraus auf deutsch steht hier Basic/Debug-Handbuch.

Der abw. Adressbereich und abw. Fehlernummern (s.u.) sind zu beachten!

Nach dem Start (auf Adr. 8000h) wir ein Autostart-Programm gesucht. Wird kein Programm auf Adresse 9020h gefunden, sendet das BASIC/DEBUG des Z8671 einen ':' als Bereitschaftszeichen (Prompt) für eine mögliche Kommunikation. Im BASIC/DEBUG-Modus können beliebige BASIC/DEBUG-Befehle an den Z8671 gesendet werden, z.B.

```
:10 PRINT "HALLO TINY"
:LIST
:RUN
```

Erweiterungen/Änderungen

Laden und Speichern erfolgt über USR-Funktionen:

```
LOAD:   L=USR(%8003)    L,S sind hier Dummy-Variablen wg. Funktionsaufruf
```

```
SAVE:   S=USR(%8006)
```

Die Funktion **RND**(n) liefert eine Zufallszahl zwischen 0 und n-1.

```
A=RND(88)
```

Achtung: Es wird die RND-Funktion des ES4.0 genutzt. Diese überschreibt leider die Register %70..%77, genau den Bereich, den BASIC/DEBUG intern für den expression stack benötigt. RND darf deshalb nicht in zu tief verschachtelten Funktionen genutzt werden, möglichst nur per Direktzuweisung an Variablen.

```
10 A=1+2*(0+0*(0+0*(RND(88)+1)))
```

ist daher nicht zulässig und produziert fehlerhafte Ergebnisse.

Speicherbereiche

```
Programm 8000-87FF
Autostart-ROM-Basic-Programm ab 9020h
Basic-Programm  E000..EFFF (am oberen Ende Arbeitszellen)
```

Der **Basic-Beginn** kann geändert werden (Reg8+9), z.B.

```
PRINT HEX(^8)   : REM Anzeige Inhalt Doppelregister %8,9
LET ^8=%A000    : neuer Basic-Beginn auf A000h
```

Hinweis: Das Ende sollte dann auch geändert werden: ^4=Endadr+20h (oder größer), sonst kein insert/delete vernünftig möglich. Wird das Programm nur ausgeführt (z.B. ROM-BASIC-Programme), ist das nicht nötig. Wichtig ist, dass ^4 auf jeden Fall auf freien RAM zeigt (GOSUB-Stack).

RENEW, stellt nach Neustart das Programm wieder her:

```
^^8=1       (setzt erste Zeilennummer von FFFF auf 1)
```

Beenden mit Reset oder

```
GO@0
```

USR-Funktionen zur **Einzelzeichen-Ein- und -Ausgabe**

```
input     %8054  (statt %54)
output  %805E  (statt %61)
```

CLS

```
LET X=USR(%805E,12): REM CLS, X IST DUMMY
```

JTC-Emu-Hinweis:

jtc-Dateien laden, dann Endadr+20h (oder größer) nach %4,5, sonst kein insert/delete vernünftig

möglich. z.B.

```
^4=%89FF
```

Beispiele

LIST innerhalb Programm zur Textausgabe (aus Z8671 BASIC/DEBUG Manual)

```
100 REM THIS PROGRAM PRINTS A MESSAGE N TIMES
110 IF N>0 THEN 200
120 : PRINT "HOW MANY TIMES";
130 : INPUT N
200 REM BEGIN LOOP
210 : LET N=N-1
220 : LIST 1000, 1070
230 : IF N>0 THEN 210
240 STOP
1000|  This is a message saved in memory.  It will be
1010|printed when the program is RUN.  If you tried to
1020|execute lines 1000 to 1070 you would get an error
1030|message.  But in this program, lines 1000+ are not
1040|executed, just LISTed.
1050|
1060| (Signed)
1070|
```

MASTERMIND, aus Zilog Z8671-7-chip-computer, leicht modifiziert.

- Einsatz der zusätzlichen Funktion RND
- ^10+2 = YY22 Pointer auf Variable A. Durch Direktzugriff im Speicher kann auf die Variablen A..D,E..H indiziert zugegriffen werden, es sind Schleifen möglich. (als Array-Ersatz)
- THEN, LET, PRINT vor „“, GOTO vor Zahl nach IF, Leerzeichen können entfallen, Für Tempo und Lesbarkeit sollten sie stehen.
- „Y/N“:INPUT X - Eingabe von Variablen ist zulässig, „Y“ wird als Variable ausgewertet. Die nachfolgende Zeile vergleicht X mit dem Wert Y. Gute Idee!

```
10 REM MASTERMIND
40 A=RND(10):B=RND(10):C=RND(10):D=RND(10)
50 "":I=0
100 "GUESS ",:IN E,F,G,H
110 I=I+1
300 J=^10+2:K=J+8
301 L=0
302 R=0:P=0
310 IF ^J=^K LET P=P+1
320 J=J+2:K=K+2:L=L+1:IF 4>L GOTO 310
330 J=^10+2:K=J+8
331 L=0
```

```
340 IF ^J=^K LET R=R+1:^J=^J+10:L=3
341 J=J+2
350 L=L+1: IF 4>L GOTO 340
351 J=^10+2
352 L=0
360 K=K+2:IF ^10+17>K GOTO 340
363 J=^10+2:K=J+8
366 IF ^J>9 LET ^J=^J-10
367 J=J+2
368 IF ^10+9>J GOTO 366
369 Y=1:N=0
370 "RIGHT ";R;" PLACE ";P
380 IF 4>P GOTO 100
390 X=0:Y=1
400 "RIGHT IN ";I;" GUESSES;";"PLAY ANOTHER Y/N":INPUT X
410 IF X=Y GOTO 10
```

Fehlermeldungen

Durch die erweiterte IL verschieben sich die Fehlernummern gegenüber dem Original:

```
11  Program line has a line number 0 or greater than 32768.
17  Memory full; new line not inserted.
27  No program to RUN.
38  GOTO is not at the end of program line.
42  Cannot GOTO negative or zero line number.
45  Line number in GOTO does not exist.
67  GOSUB is not at the end of the line.
72  Unrecognizable statement type beginning with GO.
82  Unrecognizable statement type, or '=' missing from LET statement.
100 LET is missing its '='.
142 Quote missing in PRINT statement.
173 RETURN is not at the end of the line.
174 GOSUB stack underflow.
177 The GOSUB for this RETURN no longer exists.
183 STOP is not at the end of the line.
209 INPUT variable name is missing.
212 IN or INPUT expects variable name.
250 LIST is not at end of line.
313 Unrecognizable relation in IF statement.
349 Out of memory on GOSUB or expression evaluation.
384 Divide by zero.
394 Missing parenthesis in AND, RND or USR call.
441 Syntax error in expression, or unrecognizable statement type.
445 Missing right parenthesis in expression.
```

From:
https://hc-ddr.hucki.net/wiki/ - **Homecomputer DDR**

Permanent link:
https://hc-ddr.hucki.net/wiki/doku.php/tiny/es40/basic_debug

Last update: **2022/08/04 06:37**