# HC-Basic

## Downloads

- z1013-hcbasic.zip alle HC-BASIC-Varianten incl. komplettem kommentieren Quellcode!

## HC-BASIC 85/1 H

auf der Kassette M0191 zur ersten Z9001-Tagung gab es einen 10K-BASIC-Interpreter. Dieses BASIC basiert im Kern auf dem RAM-BASIC des Z9001 (BASIC-84).

Dieses BASIC enthält viel Code, der auch im orignal Riesa-HC-BASIC (nächster Absatz) enthalten ist, aber an teilweise völlig anderen Stellen. Der Code sieht anfangs recht strukturiert aus, aber offenbar wurde das BASIC nachträglich gepatcht, um die Befehle HLOAD und HSAVE unterzubringen.

Könnte das Basic ein frühe Version aus der Riesaer Schmiede sein?

## orignal Riesa HC-BASIC

Auf der Kassette M0111 gab es die wohl offizielle Version eines großen, zum Z9001 (und KC85/3) kompatiblen Basic, das HC-BASIC.

Dieses BASIC basiert im Kern auf dem BASIC-85 des Z9001. Die Basicerweiterung enthält M511-Code, der weiterentwickelt wurde, aber keinen BM608-Code. Die Entwickler hatten wohl Zugang zu den Quellcodes des 10K-Basics aus Dresden…

Die Weiterentwicklungen (s.u.) basieren alle auf diesem BASIC und sind offenbar durch Patchen entstanden.

Adressbereich : 100H - 2AFFH
zusätzlich: 0B0h-0FFh
Startadresse : 300H od. @B
Restart : 302H od. @R

Für den A2-Monitor muss das BASIC angepasst werden. Details dazu siehe Bedienungsanleitung Z1013.64.

## KC-BASIC+

**KC-BASIC+** von V.Lühne/IG-HC TU Dresden

KC-BASIC+ ist das 10-K-BASIC, welches gegenüber dem HC-BASIC von Riesa 16-K alle Direktsprünge in den MONITOR vermeidet.

**KC-BASIC+m** von M.Duchrow/Berlin

**KC-BASIC+m1** von Dr.Trog-CC URANIA Berlin

Die letzte Version mit Bearbeitungen ist mit implementiertem 'main-tape' (m) (Duchow/Berlin) und verbesserter Funktion des 'LIST#', wo jetzt ein anschließendes 'verify' möglich ist (Dr.Trog-CC URANIA Berlin). Ergebnisse sind ein verbessertes Einladen von BASIC-Programmen im KC-Modus und bei Rundfunk-Mitschnitten.

# ROM-BASIC

**KC-ROM-BASIC+** von V.Lühne/IG-HC TU Dresden

Dieses 10-K-Basic kann ab C000H in einem ROM (auch RAM-lauffähig) stehen. Es ist zum KC-BASIC+ voll kompatibel. Seine Programme stehen zwar in einem anderen Speicherbereich, nutzen aber intern die gleichen Adressen. Der KC-ROM-BASIC+ rechnet sich diese dann jeweils um. Der KC-BASIC+ ab 100H lädt sich diese Programme richtig in den Speicher, umgekehrt ebenfalls.

Außerdem gibt es ein externes Programm zur Umwandlung von HC-Basic-Programmen vom oberen zum unteren Standort: „wandlg hc-basic“ (vp).

# Bedienung

TODO

## Kommandos

AUTO /ZN ab/,ZN

```
      Automatische Zeilennummerierung.Ohne
      Parameter 10,10
      Parameter : ab Zeilennummer,Schrittweite
      Abbruch   : BREAK
```

CLEAR /I,J

```
      Loeschen aller Variablen
      Parameter : I Neureservierung Stringbereich
                  J neues Speicherende
```

DELETE ZN/,ZN bis

```
      Loeschen Programmzeile bzw. von,bis
```

LIST /ZN

```
      Auflisten des Programmes ab niedrigster
      Zeilennummer bzw. ab ZN.Es werden 10 Zeilen
      aufgelistet.Veraenderbar durch LINES N.
      Fortsetzung mit ENTER.
```

LINES N

```
      Neuvereinbarung N-Zeilen bei LIST.
      Bleibt bis auf Wiederruf erhalten.
```

RENUMBER /ZN ab/,ZN bis/,ZN 1/,SW

```
      Neunummerierung des Programmes.
      Zeilennr. ab,Zeilennr. bis,neue niedrigste Schrw.
      Standard: Niedrigste,hoechste,niedrigste =10
```

EDIT ZN

```
      Editieren des Programmes ab ZN.Der Kursor
      steht am Zeilenende und kann zur fehler-
      haften Stelle gefuehrt werden.
      Abbruch : BREAK
```

RUN ZN

```
      Loeschen aller Variablen und Starten des
      Programmes ab niedrigster oder ab ZN.
      Sollen vereinbarte Variablen erhalten bleiben,
      dann Start mit GOTO ZN.
```

CSAVE„Filename“

```
      Abspeichern des Programmes.Filename max. 8 Bytes.
```

CLOAD„Filename“

```
      Laden eines Programmes.
```

BYE

```
      Verlassen des Interpreters.
```

TRON

```
      Trace-Modus.Bei Abarbeitung des Programmes
      werden die Zeilennummern mit ausgegeben.
```

TROFF

```
      Ausschalten des Trace-Modus.
```

Jedes laufende Programm kann mit BREAK (03H) abgebr. werden. In diesem Fall erfolgt Meldung: BREAK AT Zeilennummer.

- max. Zeilenlaenge 70 Zeichen.
- Anweisungen durch Doppelpunkt getrennt.
- Zahlenbereich : Festpunktzahlen sechsstellig + Vorzeichen

Gleitpunktzahlen Mantisse wie Festpunktz.
Exponent max. +/- 38

- Variablen : Namen = 2 Stellen,erstes Zeichen ein Buchstabe

IF,OR,ON als Namen verboten.
num.Variable z.B. A1, AA ...
String z.B. B$, U$ ...

- Anweisungen ohne Zeilennummer werden sofort ausgefuehrt.
- Basic-Programme sind ab 2C00H abgelegt. Ende durch 00 00 00.

## Anweisungen

DATA N1/,N2/,N3 ...

```
      Bereitstellen von Daten fuer die READ-Anw.
      Zahlenwerte oder Strings(Strings ohne "......")
```

READ N/,N1...

```
      Die in DATA bereitgestellten Daten werden der
      Reihe nach ausgelesen und dem entsprechenden Namen
      zugeordnet.Es koennen nicht
      mehr Daten gelesen werden als bereitstehen.
```

RESTORE /ZN

```
      Ruecksetzen des DATA-Zeigers auf erste
      DATA-Zeile bzw. Zeilennummer ZN.
```

DIM N(I)/,N1(I,J,...)

```
      Dimensionieren von Indexvariablen.Menge
      beliebig.Auch Strings(dann Stringzeichen)
      Theoretisch 255 Dimensionen moeglich.
```

PRINT N, N$ ,„N“

```
      Statt PRINT ist auch " ? " moeglich.
      Ausdruck der folgenden Werte ab letzter Cursor-
      position. Variablen,Strings ("...")
      PRINT ohne Ausdruck ergibt Leerzeile.
```

PRINT AT (Z,S);...

```
      Ausdruck auf ZEILE 0-31,SPALTE 0-31.
      Kursor bleibt aber auf Spaltenposition
      wird intern weitergezaehlt.Aufzaehlungen
      mit Kommas trennen!
```

LET N=...

```
      Wertzuweisung,kann entfallen.
      Additive Verknuepfung von Strings moeglich.
```

DEF FNA(I)=...

```
      Definieren einer Funktion mit dem Namen A
      und einer Variablen I.
```

OUT P,I

```
      Ausgabe eines Wertes I auf dem Port P.
```

A=INP(P)

```
      Port P wird gelesen und der Variablen A zugewiesen
```

WAIT P,U/,X

```
     Stop bis ein Wert vom Port P kommt.Wert wird
      mit U UND-verknuepft.Wenn =0 dann weiter.
      X=Erstverknuepfungswert XOR (normal =0)
```

WIDTH N

```
      Breite einer Ausgabezeile N Zeichen.
```

A=POS(X)

```
      Die aktuelle Cursorposition wird A
      zugewiesen.
```

PAUSE/N

```
      Ohne Variable Unterbrechung bis BREAK
      mit Variable Unterbrechung fuer N/10 Sekunden.
```

GOSUB ZN

```
      Sprung in ein Unterprogramm ab ZN.
      Abschluss des Unterprogramms mit RETURN
```

GOTO ZN

```
      Unbedingter Sprung zu Zeile ZN.
```

ON A GOTO ZN1/,ZN2...

```
      Verteilersprung.Es wird zur A-ten
      Zeile gesprungen.Ist A gleich oder grosser
      als ZN angegeben sind,so wird zur
      folgenden ZN gesprungen.
```

IF ... THEN ... /ELSE ...

```
      Bedingte Anweisung.Statt THEN kann auch
      GOTO stehen.ELSE kann entfallen.
```

FOR I=A TO B/STEP C

```
      Laufanweisung.A=Anfangswert,B=Endwert
      C=Schrittweite.(Standard SW=1)
```

NEXT/I/J

```
      Ende einer Laufanweisung.Mehrere Schleifen
      koennen verschachtelt sein,duerfen sich
      aber nicht ueberlappen.
```

WINDOW Z1,Z2,S1,S2

```
      Fenster von Z1-Z2,S1-S2
      WINDOW ohne Werte = volles Fenster
```

CALL N

```
      Aufruf Maschinenunterprogramm (Ende C9H)
      Ist N groesser 32767,dann N=N-65536
      Auch HEX-Werte moeglich.Bsp. CALL*37A0
```

POKE A,N

```
      Auf Adresse A wird der Wert N abgelegt.
```

N=PEEK(A)

```
      Der Wert von Speicherplatz A wird N.
```

DOKE A,N

```
      Auf Adresse A u. A+1 wird N abgelegt.
```

N=DEEK(A)

```
      Ab A werden 2 Bytes gelesen (NWT,HWT)
```

CLS

```
      Bildschirm loeschen
```

REM

```
      Kommentarzeile
      (Statt REM kann auch ! geschrieben werden)
```

END

```
      Programmende.(Notwendig wenn Unterprogr.folgen)
```

STOP

```
      Programmierter Abbruch (BREAK IN ZN)
```

CONT

```
      Fortsetzung des Programms nach STOP.
```

N=FRE(A)

```
      N erhaelt Anzahl der noch freien Speicher-
      plaetze. Klammerausdruck definiert
      Variablentyp z.B. PRINT FRE(A$) gibt Platz
      fuer Strings an.
```

SPC(N)

```
      Nach PRINT Ausgabe von N Leerzeichen.
```

TAB(N)

```
      Nach PRINT Ausgabe ab N-ter Spalte.
```

INPUT„Text“;N,N$,T6

```
      Eingabe eines Wetes bzw.einer Zeichen-
```

```
      kette wird erwartet. (Abschluss mit ENTER)
      Es werden so viele Werte abgefragt, wie
      angegeben sind.
      Bei zu wenig Werten  : ??
          zu vielen        : EXTRA IGNORED
          unzulaessig      : REDO FROM START
```

N$=INKEY$

```
      Abfrage Tastatur bis Taste gedrueckt und
      danach N$ zugewiesen.Programm haelt nicht an.
```

## Stringverarbeitung

ASC (X$)

```
      liefert ASCII-Code des 1. Zeichens in X$
```

CHR$ (I)

```
      wandelt I in ASCII-Code um
```

LEFT$ (X$,I)

```
      liefert I Zeichen von links aus X$
```

RIGHT (X$,I)

```
      liefert I Zeichen von rechts aus X$
```

MID$ (X$,I)

```
      liefert die Zeichen ab der i-ten Position ab X$
```

STR$ (A)

```
      wandelt den Zahlenwert von A in ein String
```

VAL (X$)

```
      wandelt X$ in eine Zahl (Umkehr von STR$ (A))
```

LEN (X$)

```
      liefert die Laenge von X$
```

STRING$ (I,X$)

```
      setzt I-mal X$ zu einem neuen String zusammen.
```

INSTR(A$,B$)

```
      Posit. des ersten Auftretens von A$ in B$
```

## Mathematische Funktionen

ABS (N)

```
      Absolutbetrag von N
```

INT (N)

```
      Wert vor dem Komma (Integer)
```

SGN (N)

```
      Signum-Funktion (Vorzeichen)
```

SQR (N)

```
      Wurzel aus N
```

SIN (N)

```
      Sinus N in Bogenmaß
```

COS (N)

```
      Cosinus N in Bogenmaß
```

TAN (N)

```
      Tangens N in Bogenmaß
```

ATN (N)

```
      Arcus-Tangens in Bogenmaß
```

LN (N)

```
      Natuerlicher Logarithmus von N
```

EXP (N)

```
      Exponentialfunktion N
```

RND (N)

```
      Zufallszahl groesser 0 kleiner 1
      N=0 dann Wert ZZ normal
      N>0 dann naechster Wert der ZZ-Reihe
      N<0 dann neue Folge von ZZ (neu initialisieren)
```

PI

```
      Konstante PI=3.14159
```

## Logische Operationen

AND

```
      logisches UND
```

OR

```
      logisches ODER
```

NOT

```
      logische Negation
```

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

Permanent link:
**https://hc-ddr.hucki.net/wiki/doku.php/z1013/software/hc-basic**

Last update: **2016/11/09 10:38**