Programmieren

Für den Z9001 stehen diverse Programmiersprachen bereit:

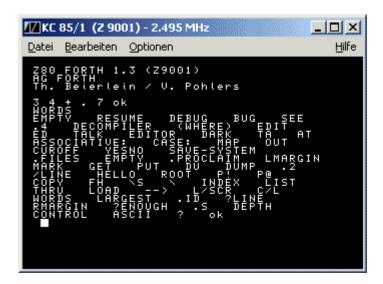
- BASIC
- FORTH
- PASCAL
- Assembler

BASIC

Der Einsteiger sollte mit BASIC beginnen. Mit einem BASIC-ROM-Modul oder mit dem eingebauten BASIC des KC 87 2.x kann man gleich nach dem Einschalten mit der Arbeit beginnen. Das BASIC wird ausführlich im **Programmierhandbuch** und im **Anhang zum Programmierhandbuch** beschrieben (s. Handbücher und Downloads der Handbücher bei http://www.sax.de/~zander/.

FORTH

Es gibt verschiedene FORTH-Versionen. Bekannt ist das f.i.g.-FORTH **FORTH** und das FG FORTH83 d. DDR **F83**. Beide sind im Mega-Flash-Modul enthalten. F83 ist deutlich umfangreicher und schneller.



Beim KC-Club Treffen 2012 gab es eine kleine Einführung ins FG-FORTH FORTH83 z9001 f83.pdf.

PASCAL

Für den KC gibt es das KC-PASCAL, eine Variante des bekannten Hisoft-PASCAL, sowie als 32K-Modul ein Turbo-Pascal-ähnliches Pascal von der TH Leipzig (s. weitere Module, das Handbuch findet man bei http://www.sax.de/~zander/).

Last update: 2014/07/30 07:55

Assembler

Zur vollständigen und systemnahen Programmierung eignet sich Assembler. robotron bietet mit EDAS und IDAS gleich zwei Assembler an. Mit **ZSID** und **R80** stehen außerdem Debugger und Reassembler im Mega-Flash-Modul zur Verfügung.

Allerdings ist das Programmieren in Assembler um einiges komplexer und schwerer als in den "höheren" Programmiersprachen.

Als Basis sollten unbedingt die Beschreibung des Betriebssystems incl. Betriebssystemlisting **robotron Betriebssystem KC 85/1 (Z9001)** studiert werden. Das Handbuch findet man bei http://www.sax.de/~zander/.

Systemfunktionen

Zur systemunabhängigen Programmierung werden vom Betriebssystem 33 Systemrufe bereitgestellt. Diese werden analog CP/M über CALL 0005 aufgerufen. Die Auswahl des gewünschten Systemrufes erfolgt über das C-Register, dessen Inhalt den Systemruf adressiert. Verschiedene Systemrufe erwarten Eingabeparameter bzw. liefern Parameter zurück.

Eingabeparameter:

- Bytewerte im E -Register
- Wortwerte im DE-Register

Ausgabeparameter:

- Bytewerte im A -Register
- Wortwerte im BC-Register

Wichtige Systemrufe:

Rufnr.	Name	Funktion
01	CONSI	Eingabe eines Zeichens von CONST
02	CONSO	Ausgabe eines Zeichens zu CONST
09	PRNST	Ausgabe einer Zeichenkette zu CONST
10	RCONB	Eingabe einer Zeichenkette von CONST
11	CSTS	Abfrage Status CONST
17	GETCU	Abfrage logische und pyhsische Cursoradresse
18	SETCU	Setzen logische Cursoradresse

Der OS-Rahmen

Damit eigene Programme vom OS aus gestartet werden können, wird ein spezieller Code benötigt, der sogenannte OS-Rahmen. Damit erscheinen Programme als transiente Kommandos im OS und können über den Programmnamen aufgerufen werden. Außerdem können Parameter übergeben werden (s. z.B. Code von OS-SAVE).

Ein OS-Rahmen muss auf einer integralen 100H-Grenze (300h ... 0BF00h) beginnen. Es können beliebig viele Kommandos in einem OS-Rahmen angegeben werden.

```
ORG
          xx00h
   JP
         AUSF
                      ;Sprung zur Kommandoausführung1
          'NAME ',0 ;Kommandoname1 (im OS-Mode einzugeben)
   DB
               ;8 Zeichen, ggf. mit Leerzeichen auffüllen, Null-Byte
   JP
                       ;Sprung zur Kommandoausführung2
          AUSF2
                    ',0 ;Kommandoname2 (im OS-Mode einzugeben)
          'NAME2
   DB
   DB
         0
                   :Kennzeichen OS-Rahmen Ende
AUSF:
```

Beispiele

Folgendes Programm gibt den Text "Hallo User!" auf den Bildschirm aus. Das Programm wird mit dem Kommando TEST gestartet.

```
z80
   cpu
           300h
   org
Beispiel:
;Löschen Bildschirm in Hintergrundfarbe blau
;Ausgabe einer Kopfzeile in der Farbe rot
; Start im OS mit TEST
   jр
         main
          "TEST
                   ",0
                        ; 8 Zeichen; Ende der Zeichenkette
   db
   db
                     ; Ende des Headers
      ld
main:
             de, text
   ld
         c,9
   call
           5
   jp
         0
; Zeichenkettendefinition
TEXT:
       DB 15H
                           ;Farbsteuercode Hintergrund
   DB 4
                     ; Farbe BLAU
                     ;Code für CLEAR SCREEN
   DB OCH
   DB 14H
                       ; Farbsteuercode Vordergrund
                     ; Farbe ROT
   DB 1
   DB "Hallo User!"
   DA OAODH
                     ; CRLF
                     ;Ende der Zeichenkette
   DB 0
   end
```

Beispiel2: Tastaturabfrage

```
cpu
             z80
       orq
             300h
; Ausgabe Taste hexadezimal
; Start im OS mit TEST
       iр
            main
       db "TEST ",0 ; 8 Zeichen; Ende der Zeichenkette
       db
                       ; Ende des Headers
main:
      ld c,11 ; CSTS
       call 5
       push af
       call out a
       pop af
; jr main ; variante A: der Tastcode bleibt erhalten
       or
           а
       jr z,main ; keine Taste gedrückt
ld c,1 ; CONSI
call 5 ; sonst Taste aus Puffer holen
       jr main
; Ausgabe A hexadezimal ASCII 2 Stellen
out a: push af
       and 0F0h
       rlca
       rlca
       rlca
       rlca
       call out_a1
       pop af
       and OFh
           add a, 30h ; '0'
out_a1: add a, 30h

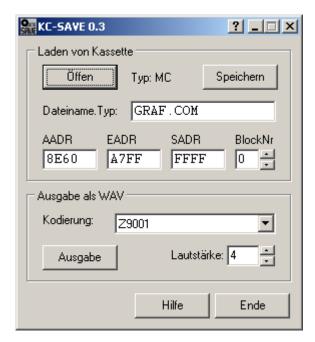
cp 3Ah ; '9'+1

jr c, out_a2
       add a, 7
; Zeichenausgabe A
out_a2: ld e, a
       ld c,2 ; CONSO
       call
       ret
       end
```

https://hc-ddr.hucki.net/wiki/ Printed on 2025/12/03 21:21

Programmerstellung am PC

Bei großen Programmen ist es leichter diese am PC zu schreiben und zu assemblieren. Ich nutze dafür den arnold-assembler. Kleine in Perl geschriebene Hilfstools unterstützen den Prozess und erzeugen z.B. gleich tap-Dateien, die im Emulator geladen werden können oder mit KCSAVE kcsave.rar als Audiosignal am realen KC geladen werden können.





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

Permanent link:

https://hc-ddr.hucki.net/wiki/doku.php/z9001/programmieren?rev=1406706904

