EDAS

ein Editor/Assembler für den Z9001

Mit dem EDAS können Programme in Assemblersprache geschrieben und in Maschinencode übersetzt werden. Der Editor ist recht einfach aufgebaut und arbeitet zeilenorientiert. Der Assembler benutzt ebenso wie der IDAS die SYPS-K-1520-Syntax. Der Assembler ist ein Absolutassembler, d.h. er kann nur direkt Maschinencode erzeugen, aber keine REL-Dateien. Er ist leider auch nicht makrofähig.

Speicherbelegung

RAM-Version 1000H-37FFh

ROM-Version C000h-E7FFh

Der Speicher von 0300h-0FFFh wird als Arbeitspeicher genutzt.

Der Speicher von 3800H bis RAM-Ende (RAM-Version) bzw. 1000H-RAM-Ende (ROM-Version) wird als Textspeicher genutzt. Mit dem Kommando #A des Editors kann dieser Bereich geändert werden, um Speicherplatz für den übersetzten Maschinencode freizuhalten.

Editor

Der Editor wird vom OS durch Eingabe von **EDIT** gestartet.

Nach dem Start meldet sich der Editor mit „EDIT:“. Wird hier „N“ (Neustart) eingegeben, wir der Editor initialisiert; vorher eingegebener Text ist verloren. Jede andere Taste bewirken einen Restart des Editors. Ein im Textpuffer stehender Text bleibt erhalten.

Durch die Ausgabe von „+“ auf dem Bildschirm wird die nächste Bedienereingabe angefordert.

Der Editor arbeitet zeilenorientiert, d.h., es werden stets vollständige Zeilen verarbeitet. Die Spezifizierung der Zeilen kann sowohl durch die Zeilennummer als auch durch eine zu Beginn der Zeile stehende Zeichenkette erfolgen.

Korrekturen während der Eingabe einer Zeile sind durch Betätigen der Kursortaste ← und anschließende Eingabe der richtigen Zeichen möglich. Beginnt die Bedienereingabe mit dem Zeichen # (Kommandokennzeichen), so wird die nachfolgend eingegebene Zeile als Kommentar interpretiert. Andernfalls wird sie als neue Zeile in den Quelltext übernommen. Die maximale Länge einer Zeile beträgt 72 Zeichen.

EDIT realisiert folgende Funktionen:

- Streichen von Zeilen,
- Einfügen von Zeilen,
- Substituieren von Zeichenketten,

- Anfügen von Kommentar,
- Streichen von Kommentar,
- Anzeigen des Textpufferinhaltes auf dem Bildschirm,
- Ausgeben des Textpufferinhaltes auf Magnetbandkassette,
- Eingeben von Quelltexten von der Magnetbandkassette in den Textpuffer.

Kommando	**Syntax**	**Erläuterung**
assign	#A	Anzeigen und Modifizieren des Textpufferbereiches Eingabe hexadezimal als „neue Textpufferanfangsadresse,Endeadresse[N]“ in der Form XXXXX,XXXXX[N]. Das „N“ (Neustart) bewirkt ein Leeren des Textpuffers.
begin	#B	Neustart des Aufbereitungsvorganges
end	#E	Beenden der Arbeit mit dem Editor
read	#R„Name“	Eingabe von der Kassette in den Textpuffer
write	#W„Name“	Ausgabe des Textpufferinhaltes auf Magnetband
text	#Tn,m	Ausschreiben des Textpufferinhaltes
insert	#In	Einfügen von Zeilen
kill	#Kn,m	Streichen von Zeilen
comment	#Cn,m	Anfügen von Kommentar
no comment	#Nn,m	Streichen von Kommentar
substitute	#Sn,m /ZKa1/ZKn1/,/ZKa2/ZKn2/,…,/ZKan/ZKnn/	Substituieren von Teilzeichenketten
uppercase	#UN oder #U[Y]	Umwandlung von Kleinbuchstaben in Großbuchstaben
???	#D	Kommando ist nicht dokumentiert
???	#L	Kommando ist nicht dokumentiert

Fehlermeldungen

+++CE (command error)	Das eingegebene Kommando ist syntaktisch fehlerhaft oder nicht sinnfällig.
+++LE (length error)	Die maximale Zeilenlänge (72) wurde überschritten.
+++NF (not found)	Eine spezifizierte Zeile wurde im Textpuffer nicht gefunden.
+++TB (textbuffer)	Das Ende des Textpuffers ist erreicht.
+++SE (substitution error)	Bei einem S-Kommando konnte keine der angegebenen Substitutionen ausgeführt werden

Assembler

Der Assembler wird vom OS durch Eingabe von **ASM** gestartet. Er meldet sich nach dem Start mit der Ausschrift „ASM–HC:“ und erwartet die Eingabe eines Pflichtkommandos. Beim ersten Start muß ein „N“ eingegeben werden.

ASM realisiert folgende Funktionen:

- Quellprogrammeingabe aus dem Textpuffer des Editors EDIT,
- Maschinencodeausgabe in den Speicher,

- Maschinencodeausgabe auf das Magnetband,
- Ausgabe der Assemblerprotokolle auf dem Listgerät (Bildschirm oder Drucker). Es kann zwischen Übersetzungsliste, Fehlerliste, Crossreferenztabelle und Symbolreferenztabelle gewählt werden.

ASM übersetzt den Inhalt des Textpuffers des Editors EDIT in den Maschinencode. Die Ausgabe erfolgt auf das Magnetband oder in den Arbeitsspeicher (RAM) des Heimcomputers. Das Assemblerprotokoll kann über den Bildschirm oder einen Drucker, falls ein Drucker und das entsprechende Druckerbedienungsprogramm vorhanden sind, ausgegeben werden.

ASM benötigt 2 Durchläufe (Pässe) zur Übersetzung. Im 1. Durchlauf erzeugt er die Symboltabelle. Im 2. Durchlauf kann gewählt werden zwischen:

- Ausgabe der Übersetzungsliste,
- Ausgabe des Maschinencodes in den Arbeitsspeicher oder auf Kassette,
- Ausgabe der Fehlerliste der Assemblerquelle,
- Ausgabe der Crossreferenztabelle,
- Ausgabe der Symbolreferenztabelle.

Mit der Ausschrift „PASS:“ fordert ASM die Spezifikation des aktuellen Passes an. Es sind folgende Eingaben möglich:

1	Ausführung des 1. Durchlaufs. Von dem Quelltext, der im Textpuffer des EDIT steht, wird eine Symboltabelle im RAM erzeugt. Dieser Paß ist für die Übersetzung eines Programmes unbedingt notwendig.
P (print)	P[Zeilennummer1,Zeilennummer2[,Programmende]] Ausgabe der Übersetzungsliste auf dem Listgerät.
M (MC-Output)	Ausgabe des Maschinencodes auf Magnetbandkassette oder in den Arbeitsspeicher entsprechend der Angabe „T“ oder „R“.
F (fault)	Syntaktische Prüfung des Quellcodes und Ausgabe der Fehlermeldungen bei syntaktisch falschem Programm.
X (cross-referenze)	Ausgabe der Crossreferenztabelle
A (adress-dictionary)	Ausgabe der Symbolreferenztabelle
V (variation)	Änderung des Datenflusses für die Ausgabe des Maschinencodes und der Assemblerprotokolle. Nach der Ausschrift „MO:“ wird durch die Eingabe von „T“ oder „R“ festgelegt, ob die MC-Ausgabe bei der Ausführung des Kommandos M auf die Magnetbandkassette oder in den RAM oder erfolgen soll. Nach der Ausgabe von „LO:“ wird durch die Eingabe von „C“ der Bildschirm (Console) oder „F“ der Drucker als Listgerät festgelegt.
E (end)	Rücksprung in das Steuerprogramm (Monitor).

Der 1. Paß ist Voraussetzung für die Übersetzungsläufe P, F, M, X und A. Diese können in beliebiger Reihenfolge durchgeführt werden.

Syntax

ASM versteht nur Befehle (Mnemoniks) mit Großbuchstaben. Mnemoniks mit Kleinbuchstaben weist ASM als syntaktisch falsch aus.

Gegenüberstellung der Schreibweise einiger Beispielbefehle:

```
     ROBOTRON      Zilog             ROBOTRON    Zilog

M1:  LD M,A   M1:  LD (HL),A         OUT 80H     OUT (80H),A
     JR M1-#       JR M1             IN  3CH     IN A,(3CH)
     JMP M1        JP M1             CMP 0       CP 0
     JRC M1-#      JR C,M1           ADD B       ADD A,B
     JPZ M1        JP Z,M1           JMP M       JP (HL)
     CANZ M1       CALL NZ,M1        RNC         RET NC
     DJNZ M1-#     DJNZ M1
```

Steueranweisungen für den Assembler

ORG adresse	Anfangsadresse ; muss im MC-Bereich liegen
konstante:EQU wert	Wertzuweisung zu einem Symbolnamen
BER anzahl	Bereich freihalten (wird nicht gelöscht)
DB byte	Byte definieren
DB 'text'	Text definieren. Max. 35 Zeichen.
DA adr	Adresse (Doppelbyte) definieren
END	Ende des Quelltextes ; danach dürfen keine Befehle mehr stehen, sonst Absturz möglich

Zahlenwerte können dezimal (ohne Kennzeichnung) oder hexadezimal (mit nachgestelltem 'H') angegeben werden. Zahlen müssen mit einer Ziffer beginnen, d.h. Hexadezimalzahlen größer 9FH benötigen eine Vornull (z.B.: 0CBH). Die Verwendung von ASCII-Zeichen ist möglich, wenn diese in Hochkommas eingeschlossen werden.

Bsp.: LD A,75 LD A,4BH LD A,'K' sind identisch.

Fehlermeldungen

Folgende Fehlerausschriften können auftreten:

OVERFLOW	Überlauf der Symbol- oder Crossreferenztabelle, oder Textpuffer enthält kein vollständiges Programm (PN ... END)
NO PN:	Programm beginnt nicht mit der PN-Anweisung. Durch Eingabe eines „C“ wird PN weiter im Quellcode gesucht.
+A+	Adressierungsfehler
+C+	Falsche Mnemonik
+F+	Formatfehler
+M+	mehrfach definiertes Symbol
+N+	nicht definiertes Symbol
+O+	Operandenfehler

Hinweise

Versionitis:

RAM-Version von Kassette R0121
1. Block CRC = FF06, SUM = F6D4
2. Block CRC = D060, SUM = A690
3. Block CRC = 58BB, SUM = E640
4. Block CRC = 2BD6, SUM = FB4F
5. Block CRC = 2868, SUM = F82A

ROM-Version V. Pohlers ASM.COM
1. Block CRC = A06F, SUM = 750A
2. Block CRC = 62E2, SUM = E05D
3. Block CRC = CCE5, SUM = 542C
4. Block CRC = 9249, SUM = 6ABC
5. Block CRC = 1397, SUM = 6FF4

ROM-Version U. Zander edas__c0.851
1. Block CRC = A06F, SUM = 750A
2. Block CRC = 62E2, SUM = E05D
3. Block CRC = CCE5, SUM = 542C
4. Block CRC = B883, SUM = 718E
5. Block CRC = 1397, SUM = 6FF4

ROM-Version W. Hasche asm_whasche.rom
1. Block CRC = A06F, SUM = 750A
2. Block CRC = 62E2, SUM = E05D
3. Block CRC = CCE5, SUM = 542C
4. Block CRC = B883, SUM = 718E
5. Block CRC = E14F, SUM = 7DC4

ROM-Version Torsten Paul asm_tpaul.rom
1. Block CRC = 1253, SUM = 6CD3
2. Block CRC = D3D1, SUM = EBE3
3. Block CRC = 9467, SUM = 51DC
4. Block CRC = 3F16, SUM = 8D99
5. Block CRC = 64BF, SUM = 8434

ASM-Modul von AS
1. Block CRC = 1253
2. Block CRC = 0E96
3. Block CRC = 9467
4. Block CRC = 9D54
5. Block CRC = C6FD

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

Permanent link:
https://hc-ddr.hucki.net/wiki/doku.php/z9001/software/edas?rev=1296731893

Last update: **2011/02/03 10:18**