Ein Maschinencode-Editor für den LC-80ex

V.1.0

WeRo, Stand: 5. Juli 2016

Inhaltsverzeichnis

Einleitung....2
- Einschränkungen....2
- Programmstart und -Ende....2
- Hardware....3
  - *Speicherausbau*....3
  - *Terminal*....3
  - *Drucker*....3
  - *Dateiarbeit*....3
- Bedienung....4
- Ansichten....5

Laden & Speichern per USB....6
- Inhaltsverzeichnis USB anzeigen....6
- MC-Programm laden....6
- MC-Programm abspeichern....6
- Disassembler-Quelltext abspeichern....6

Grundlegende Bedienbeispiele....7
- Speicheradresse (MP) setzen....7
- Speicherinhalt als HEX-/ASCII-Dump listen....7
- MC-Programm oder Daten ändern/eingeben....7
- Ansicht Hauptanzeige ändern....7
- Text als String eingeben....7
- Speicherbereich füllen....7
- Bytefolge finden....8
- Kopieren/Einfügen von Bytes....8
- Prüfsumme berechnen....8
- MC-Programm starten....8
- Programm schrittweise ausführen....9
- Haltepunkt setzen....9

Disassembler....10
- Funktionen....10
- (Dis-)Assembler-Quelltext....11

Kommandoübersicht....12

Einleitung

Das Eintippen und Testen von Maschinencodeprogrammen am LC-80 "wie früher" per Hex-Tastatur ist mühsam. Hat man aber ein Terminal angeschlossen, so kann das komfortabler erfolgen...

Der nachfolgend vorgestellte Maschinencode-Editor (im folgenden "MCE" genannt) wurde auf den LC-80ex zugeschnitten. Er basiert auf einem Z1013-Programm, welches eine Portierung des Programms "**MONS3**" (ZX Spectrum, Schneider CPC64,...) ist.

Er bietet folgende Möglichkeiten:

- Speicheransicht/ -Modifikation (Hex- & ASCII-Dump, Kopieren, Füllen, Suchen),
- Registeransicht/ -Modifikation,
- Disassembler,
- Debugger (Haltepunkte, Einzelschritt)
- Starten von Programmen,
- Ermittlung von Prüfsummen,
- Filearbeit per USB (Maschinencode laden/speichern, Disassemblat speichern)
- Druckausgabe und Kommunikation per RS232.

Einschränkungen

- Der RAM-Bereich #2000....#23FF wird für Stack und Arbeitszellen benötigt, sodass "alte" LC-80-Programme ab #2000 mit "MCE" (in den originalen Adressbereich) nicht geladen bzw. dort nicht bearbeitet werden können.
- Der verfügbare RAM ist ebenfalls der Grund dafür, dass es Beschränkungen bei der Ausgabe des Disassemblers in ein File gibt (siehe hier).
- "MCE" ist langsam im Bildaufbau. Es wird zwar eine Vielzahl von Informationen ausgegeben, aber die Hauptursache ist die relativ geringe Geschwindigkeit der Zeichen-ausgabe von 2400 Baud über die serielle Schnittstelle.
- "MCE" ist kein Editor/Assembler. Ein solcher ist für den LC-80ex als separates Programm verfügbar, ebenfalls basierend auf einer Z1013-Portierung.

Programmstart und -Ende

MCE-Start	RES ADR B000 EX Bei Neustart bleibt das letzte MC-Anwenderprogramm im RAM erhalten.
ext. EPROM	#Y Startet Programm im (E)PROM ab #C000 z.B. ASSEMBLER, 8k-BASIC,...
Beenden	#Q Die Steuerung wird wieder an den LC-80ex-Monitor übergeben. Da „MCE“ zahlreiche Systemzellen im Bereich #2000...#23FF benutzt, erfolgt beim Verlassen ein „Kaltstart“ des LC-80ex (also Abspiel der Startmusik). Damit werden die Monitorarbeitszellen wieder ordnungsgemäß belegt.

Hardware

Die Benutzung von "MCE" am LC-80ex erfordert nachfolgende Hardware-Voraussetzungen:

Speicherausbau

"MCE" ist Bestandteil von LCTOOLS3 und läuft (einschließlich Treiber und Hilfsroutinen) auf einem (E)EPROM im Adressbereich #A000...#BFFF.

Zu empfehlen ist ein durchgehender RAM im Bereich #2000...#7FF0. Dabei wird der Bereich #2000...#23FF für Arbeitszellen und Stack benutzt. #2400...#7FF0 stehen für ladbare/bearbeitbare Maschinenprogramme zur Verfügung. Unabhängig davon kann der gesamte Adressraum #0000...#FFFF inspiziert/disassembliert werden.

Terminal

Für die Bildschirmausgabe und Eingabe per QWERTZ-Tastatur kommt die serielle Schnittstelle (SIO-B) zum Einsatz. Sie wird in der Standardvariante initialisiert:

2400 Baud, 8 Datenbits, 1 Stopbit, keine Parität, kein Handshake

Das gilt für einen Systemtakt des LC-80ex von 1,8432 MHz. Die LC-80ex-Hardware ist dazu wie folgt einzustellen: **JP9: Systemtakt 1/2, JP12: SIO-Port B0.** Die Verwendung des Standardtaktes 921,6 kHz ist ebenso möglich, allerdings halbiert sich dann die Baudrate auf 1200.

"MCE" ist für den Betrieb am "TV-Terminal" eingerichtet. PC-Terminal-Emulationen können auch benutzt werden, wenn sich deren Tastatur konfigurieren lässt (Kursortasten: #08...#0B, Backspace: #7F). Ggf. müssen anstelle der Kursortasten die Steuerzeichen Strg+H... Strg+K benutzt werden.

Drucker

Für einen seriellen Drucker wird das gleiche o.a. Protokoll benutzt, jedoch nur für Ausgabe an SIO Kanal A. Ein Hardware-Handshake ist realisiert (der LC-80ex wartet, bis Drucker fertig).

"MCE" ermöglicht das Ausdrucken

- eines HEX-/ASCII-Dumps Strg+P
- sowie des Disassembler-Listungs OUTPUT **(P**)rinter

Vor der Auslösung dieser druckenden Funktionen ist der Drucker bereit zu machen. Ansonsten hängt der LC-80ex fest!

Dateiarbeit

Anstelle der Routinen für die Kassettenarbeit des Z1013 wurde in "MCE" eine USB-Unterstützung implementiert. Dazu ist die entsprechende USB-Hardware erforderlich (siehe Anleitung LCTOOLS).

Bedienung

Nach dem Programmstart erscheint die Hauptansicht mit einer zusammengefassten Darstellung der aktuellen Register- und Speicherinhalte sowie einer Kommandoaufforderung # . Nach diesem "Prompt"-Zeichen können sowohl Steuercodes (Strg-Taste +...), Ziffern, Buchstaben, einige Sonderzeichen und Kombinationen daraus eingegeben werden. Beginnt eine Eingabe mit einer Ziffer 0...9 bzw. einem Buchstaben A...F, so wird das als Hexadezimalangabe interpretiert.

Erwartet ein Kommando weitere Eingaben (z.B. Adressangaben), so bleibt das bislang Eingegebene stehen, ggf. unter Angabe der Art des Parameters (z.B. "FIRST"). Ansonsten erfolgt sofortige Ausführung, ohne dass <ENTER> zu drücken ist.

Die meisten Kommandos sind sinnfällige (englische) Abkürzungen, z.B. M=**M**emorypointer setzen. Das H-Kommando zeigt eine Kurzhilfe mit den wichtigsten Kommmandos an. Eine vollständige Übersicht ist im Anhang aufgeführt.

Einige Kommandos halten in der Schirmausgabe an, nachdem eine bestimmte Menge abgearbeitet wurde. Dann erscheint <ET/^C>. Mit <ET> wird fortgesetzt. Strg+C ist auch das allgemeine Abbruchkommando für Kommandoeingaben (Adressen, Filenamen etc.). Es wird dann zur Hauptansicht zurückgekehrt, sofern nicht weitere Eingaben erforderlich waren.

Sondertasten:
Zur Navigation in der Speicheranzeige werden die Kursortasten benutzt:

Taste	**alternativ**	**Wirkung**
<Kursor links>	Strg+H	4 Adressen zurück
<Kursor rechts>	Strg+I	4 Adressen vor
<Kursor hoch>	Strg+K	1 Adresse zurück
<Kursor runter>	Strg+J	1 Adresse vor

<Backspace> löscht das zuletzt eingegebene Zeichen.

Speicherzeiger:
Für das Verständnis der Wirkung der einzelnen Kommandos ist es wichtig, die Bedeutung zweier Adress-Angaben auseinander zu halten:

MP: "Memory pointer"
Die aktuell im Zugriff befindliche RAM-Adresse ist in der Speicheranzeige mit > xxxx < markiert. Man kann sie mit den Kursortasten (+/-1, +/- 4) verändern oder auf eine einzugebende Adresse setzen: #M:nnnn <ET>
In der untersten (Assembler-)Zeile wird der MP ganz links angezeigt.

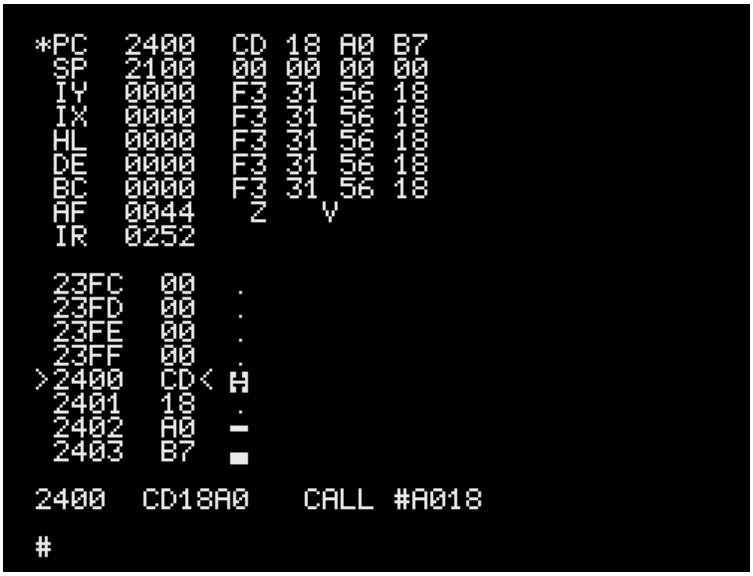
PC: "Program counter"
Diese Adresse weist den Stand des Z80-Befehlszählers aus, d.h. diejenige Adresse, ab welcher eine Abarbeitung z.B. im Einzelschrittbetrieb beginnt.
Einstellen auf Adresse pppp:
1. PC mit * markieren (per Punkt-Taste, sofern nicht schon so gegeben)
2. #pppp. neue Adresse eingeben, gefolgt von einem Punkt!

Nach Erststart von "MCE" (und dem K-Kommoando) weisen beide Adressen auf #2400, d.h. den Beginn des Anwenderbereiches. Je nach abgearbeitetem Kommando bzw. Aufgabe weicht der Befehlszähler aber im Laufe der Bedienung von der aktuellen Adresse ab. Das Laden einer Datei mit dem R-Kommando setzt MP und PC automatisch auf die Anfangsadresse der Datei.

Ansichten

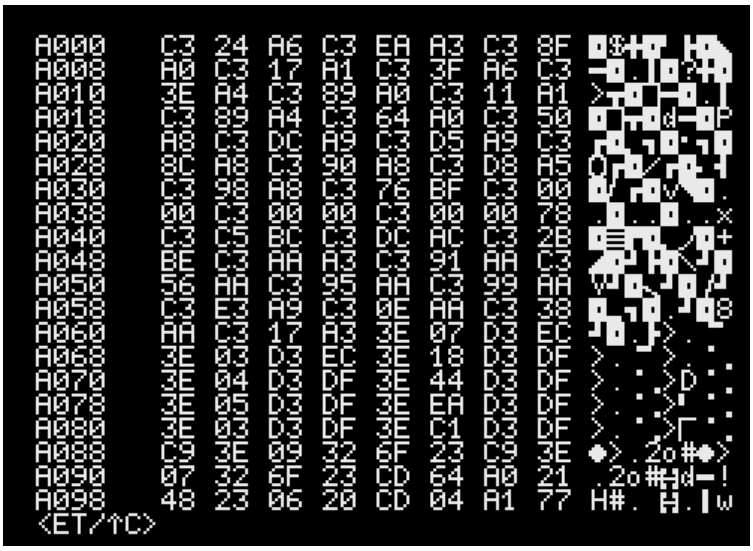
"MCE" hat drei verschiedene Ansichten:

Hauptansicht



- PC=Befehlszähler stand
- Registeranzeige jeweils mit 4 Bytes Inhalt ab Registeradresse
  * zeigt auf bearbeitbares Register
  V schaltet Registeranzeige aus/ein
- Speicheranzeige 8 Bytes
  akt. Adresse - 4/+3 Bytes
  >xxxx xx< markiert (MP), Anzeige Inhalt als HEX und ASCII-Zeichen
- Befehl auf aktueller Adresse (MP)
- Kommandoeingabe

HEX-/ASCII-Dump



nach L-Kommando: 20 Zeilen je 8 Byte ab akt. Adresse MP

Grafikdarstellung entsprechend aktuellem Terminal-Zeichensatz

Steuerzeichen (<20h) werden im ASCII-Bereich zu "."

Rücksprung zur Hauptansicht mit Strg+C

mit beliebiger Taste fortlaufend weiter

nach Strg+P (Drucken): Aufforderung zur Eingabe der Druck-Endeadresse

(Dis-)Assembler-Ansicht

nach Strg+A:
jeweils immer ca. 25 Bytes ab aktueller Adresse (MP), dann Stopp

Abbruch mit Strg+C,
mit anderer Taste weiter

nach Strg+D:
Anfang und Ende wählbar, Listing kann mit Strg+C abgebrochen werden

Laden & Speichern per USB

Maschinencode-Dateien müssen im Headersave-Format (*.z80) und Dateityp "C" (ausführbares Programm, ohne Autostart) vorliegen bzw. werden als solche erzeugt. Der Dateiname darf nicht länger als 8 Zeichen sein, die Endung "z80" ist nicht mit anzugeben.

Inhaltsverzeichnis USB anzeigen	**(U)**SB

Das U-Kommando zeigt den Inhalt des USB-Sticks. Die Dateien werden unsortiert aufgelistet (Anzeigereihenfolge ist i.A. die Schreibreihenfolge). Bei mehr als 20 Dateien erfolgt ein Stopp ("!"), weiter mit beliebiger Taste oder Abbruch mit Strg+C.

MC-Programm laden	**(R)**ead

Nach dem R-Kommando (READ) ergeht eine Aufforderung zur Eingabe eines Filenamens. Der ist mit <ET> abzuschließen. Anschließend wird die Datei in "MCE" geladen. PC und MP werden auf die Anfangsadresse des geladenen Programms gesetzt.

Anmerkungen:

- Der Adressbereich, in welchen geladen wird, ist fest im z80-File verankert. Es ist hier nicht möglich, die Datei auf eine andere Adresse zu laden! Bis auf frei verschiebliche Programme wäre das auch nicht sinnvoll.
- Es dürfen nur z80-Files geladen werden, deren Anfangsadresse >=#2400 und deren Ende <#7FF0 beträgt. Ansonsten werden die Arbeitszellen überschrieben und "MCE" ist nicht mehr funktionsfähig! Also: Keine "alten" z80-Files (ab #2000) laden!
- Ein Abbruch des R-Kommandos kann durch <ET> bei leerem Filenamen erfolgen oder auch mitten in der Eingabe per Strg+C.

MC-Programm abspeichern	**(W)**rite

Nach dem W-Kommando (WRITE) werden ein Dateiname sowie Anfangsadresse (First) und Endeadresse (Last) des abzuspeichernden Bereiches abgefragt.

Ein Abspeichern empfiehlt sich immer vor einem auszuführenden Test. So kann ein fehlerhaftes Programm (was sich beim Test ggf. selbst zerstört) schnell wieder geladen und bearbeitet werden.

Anmerkungen:

- Es wird immer ein z80-File vom Typ "C" geschrieben (ausführbares MC-Programm, kein Autostart). Als Startadresse wird die Anfangsadresse eingetragen.
- Existiert ein gleichnamiges File bereits, so erfolgt Rückfrage "Overwrite (Y)". Nur wenn Y gedrückt wurde, erfolgt ein Überschreiben, ansonsten Abbruch des Speichervorganges.
- Fehlt ein USB-Stick oder ist die Hardware fehlerhaft, so erfolgt eine Meldung "USB Error".
- Ein Abbruch des W-Kommandos kann durch <ET> bei leerem Filenamen erfolgen oder auch mitten in der Namenseingabe per Strg+C.

Disassembler-Quelltext abspeichern

Bei Angabe "OUTPUT: (P)rinter (F)ile ? " = **F** erfragt der Disassembler einen Dateinamen und speichert den Quelltext in einer Datei. Das erfolgt nicht im *.z80-Format sondern ohne Header und mit der Erweiterung ".SRC".

Grundlegende Bedienbeispiele

Nachfolgend sind einige grundlegende Bedienbeispiele aufgeführt, die den Umgang mit "MCE" erklären.

Speicheradresse (MP) setzen — **(M)**emory

Kommando	#M:3000	<ET>

Anmerkung:

- Die Hauptansicht wird aktualisiert, der Code auf der Adresse sofort disassembliert.

Speicherinhalt als HEX-/ASCII-Dump listen — **(L)**ist

1. Anfangsadresse setzen:	#M:2400	<ET>
2. Kommando	#L	
bzw. für ein Ausdrucken:	#	<Strg+P>

Anmerkung:

- Es werden immer 20 Zeilen zu je 8 Bytes ausgegeben, dann HALT oder Abbruch.
- Beim Druck ist nach LAST die letzte auszudruckende Adresse anzugeben.

MC-Programm oder Daten ändern/eingeben — Hex-Byte

1. Anfangsadresse MP setzen:	#**M**:4000	<ET>
2. erstes Datenbyte eingeben:	#AF	
<ET> oder Leertaste	übernimmt Byte und bleibt auf aktueller Adresse	
<Kursor runter>	übernimmt Byte und schaltet zur nächsten Adresse	
3. nächstes Datenbyte eingeben...		

Anmerkungen:

- Es wird immer nur 1 Byte geschrieben, bei mehreren nur das zuletzt eingegebene Byte übernommen.
- Dier letzten eingegebenen Bytes sind in der Speicheranzeige sichtbar. Hat man sich vertippt, einfach mit den Kursortasten an die gewünschte Stelle gehen und neu eingeben.
- Das eingegebene Byte wird sofort als Befehl interpretiert und in der unteren Zeile als Assembleranweisung (ggf. abhängig von Folgebytes) angezeigt.

Ansicht Hauptanzeige ändern — (**V**)iew Register

Wird keine Registeranzeige benötigt (z.B. bei fortlaufender Byte-Eingabe) so kann zur Beschleunigung der sich aktualisierenden Anzeige damit (zeitweilig) die Registeranzeige abgeschaltet werden. Das Setzen von Registerinhalten ist ist diesem Zustand nicht möglich.

Text als String eingeben — **(T)**ext

1. Anfangsadresse (MP) setzen:	#M:6000	<ET>
2. Kommando "TEXT"	#**T**	
3. ASCII-Zeichen eingeben:	QWERTZUIOP... (auch ein <ET> wird übernommen!)	
4. Ende der Eingabe:	<Strg+C>	

Anmerkungen:

- Abbruch der Texteingabe ist nur möglich, wenn nach dem T-Kommando noch nichts eingegeben wurden.
- Die Speicheranzeige wird erst nach Abschluss der Texteingabe aktualisiert.
- Nach dem letzten Zeichen wird MP um 1 erhöht, sodass das auf den String folgende Byte sofort im Zugriff ist.

Speicherbereich füllen — **(P)**attern

1. Kommando "PATTERN"	#**P**	
2. Anfangsadresse eintragen:	FIRST:2400	<ET>
3. Endeadresse eintragen:	LAST: 2450	<ET>
4. Füllbyte eintragen:	WITH: 55	<ET>

Anmerkung: Bereich #2000...#23FF nicht befüllen! (Systemzellen)

Bytefolge finden — **(G)**et Byte

1. Adresse MP=Suchanfang setzen: #**M**:4000 <ET>
2. Kommando "GET BYTE": #**G**
3. erstes Byte hexadezimal eingeben :11 <ET>
4. nächstes Byte eingeben :22 <ET>
5. Ende der Eingabe: leer : <ET>

Anmerkungen:

- Man kann nach mehreren aufeinanderfolgenden Bytes suchen. Suche beginnt immer nach dem leeren <ET>.
- Wird die Bytefolge gefunden, so wird die Speicheranzeige auf die erste Adresse der Bytefolge eingestellt.
- Wurde die Bytefolge am MP bis FFFF nicht gefunden, so wird beginnend bei 0000 weiter gesucht. Da die Bytefolge in den Arbeitszellen temporär gespeichert ist, wird sie dort immer gefunden!
- Mit dem Kommando "N" kann ein weiteres Vorkommen gesucht werden.
- Eine Eingabe der Suchzeichen als als ASCII-Werte ist nicht möglich.

Kopieren/Einfügen von Bytes — **(I)**ntell. Copy

Das I-Kommando "Intelligente Kopierfunktion" kopiert den Speicherinhalt ab der angegebenen ersten (FIRST-)Adresse bis zu letzten (LAST-)Adresse an die neue (TO-)Adresse. Eine Überlappung ist zulässig.

Mitunter steht man vor der Aufgabe, ein einziges Byte an einer bestimmten Stelle einfügen zu müssen, ohne alle nachfolgenden nochmals einzutippen (vergessen beim Eintippen...). Auch das kann sehr leicht bewerkstelligt werden. Angenommen, zwischen den Adressen 2410 und 2411 soll noch ein Byte eingefügt werden. Das Programm reiche von 2400...24FF. Dann wären die Parameter wie folgt anzugeben:

1. Kommando #**I**
2. ab hier FIRST: 2411 <ET>
3. bis hier LAST: 24FF <ET>
4. nach hier TO: 2412 <ET>

Eventuelle feste Adressbezüge in diesem Bereich werden jedoch nicht automatisch aktualisiert!

Prüfsumme berechnen — **Strg+N**

1. Kommando #Strg+**N**
2. Anfangsadresse eintragen: FIRST:1000 <ET>
3. Endeadresse eintragen: LAST: 2000 <ET>

Anmerkungen:

- Es wird eine 16bit-Prüfsumme CRC (SDLC) für den angegebenen Bereich (Anfang+Ende eingeschlossen) berechnet. Der Algorithmus entspricht dem im Monitor AC1/LLC2 bzw. dem AC1-Viewer.
- Das Ergebnis bleibt solange auf dem Schirm stehen, bis eine beliebige Taste gedrückt wird.

MC-Programm starten — **(J)**ump

Ein im RAM befindliches MC-Programm kann aus "MCE" heraus wie folgt (testweise) gestartet werden (aaaa steht für dessen Startadresse):

a) Programm normal ausführen, keine Rückkehr zu "MCE":
#**J**:aaaa

b) Programm ausführen, anschließend Rückkehr zu "MCE". Dazu ist nötig:
Adresse des Programmendes (-ausstieg) bestimmen: xxxx

#M:xxxx <ET>	Adresse aufsuchen (z.B. ein Sprung zu 0000 = RESET)
# <Strg+B>	Haltepunkt dort setzen
#J:aaaa <ET>	Starten

Programm schrittweise ausführen | + !

Für das *Debugging* ("Entwanzen") eines entworfenen Programms ist es oft nützlich, dieses schrittweise abarbeiten zu können. Dabei wird immer eine einzelne Z80-Anweisung abgearbeitet, das Programm hält an und man kann den Inhalt der einzelnen Register und Flags prüfen. Voraussetzung ist, dass die Registeranzeige eingeschaltet ist (V-Kommando).

Zu Beginn sind sowohl MP als auch PC auf die Anfangsadresse zu setzen, ab welcher man starten möchte:

#M:aaaa <ET>	MP auf Startadresse setzen
#aaaa.	PC auf Startadresse setzen (* muss vor PC stehen, dann wird mit dem Punkt der Wert übernommen)

Es stehen zwei verschiedene Kommandos zur Verfügung:

! Funktioniert nur im RAM-Bereich, aber Interrupt möglich (Haltepunkt durch Einfügen eines CALLs).

\+ Funktioniert auch im ROM-Bereich, allerdings darf dort kein Interrupt verwendet werden, da die Step-Routine die Schattenregister benutzt

Haltepunkt setzen | **(Strg+B)**reakpoint

Möchte man einen größeren Programmbereich abarbeiten und erst dann im Einzelschritt weitergehen, so kann an der Stelle ein Haltpunkt gesetzt werden. Dazu ist MP auf das Byte zu setzen, bei dem angehalten werden soll. Der Debugger trägt beim Setzen mit ^B an dieser Stelle zeitweilig einen CALL ein. Das ist auch der Grund dafür, dass die Methode nur im RAM funktioniert.

Es empfiehlt sich, den Haltpunkt möglichst auf eine 3-Byte-Anweisung zu setzen. Dies ist immer dann von Bedeutung, wenn z.B. das Programm je nach Verzweigungen nicht unbedingt mit dem 1. Byte am Haltepunkt fortfährt sondern ggf. dieses überspringt.

Hat man den Haltepunkt eingetragen, so wählt man durch entsprechendes Beschreiben von PC die Startadresse. Mit dem Kommando Strg+E (Execute) wird das Programm dann ab dort gestartet und es stoppt am definierten Haltepunkt. Der CALL-Eintrag wird nach Erreichen automatisch gelöscht und der ursprüngliche Inhalt der drei Bytes wieder hergestellt.

Es lassen sich auch mehrere Haltpunkte definieren. Jedes Strg+E geht einen Haltepunkt weiter.

Achtung:
Wird ein Haltpunkt (infolge eines Programmfehlers oder ungünstig gewählter Startadresse) gar nicht erreicht, so wird er nicht abgearbeitet und der CALL-Eintrag im Programm bleibt bestehen!

Ein direktes Anspringen der Haltepunktadresse mit J:xxxx beseitigt ihn dann und stellt den originalen Programminhalt wieder her.

Disassembler

Funktionen

Der Disassembler übersetzt den Maschinencode in festgelegte "Mnemonics" als ASCII-Text.
Grundfunktionen:

- disassembliert automatisch ständig den Befehl an akt. Adresse in der Hauptansicht und
- Schnellansicht Assemblerlisting ab akt. Adresse ca. 25 Bytes mit dem Strg+A-Kommando

Mit dem Strg+D-Kommando und diversen Parametern kann variabel gearbeitet werden:

Ausschrift	**Bedeutung**	**Beispieleingabe, Hinweise,**
DISASM RANGE FIRST: LAST:	Anfangsadresse angeben Endeadresse angeben	2000 <ET> 2FFF <ET>
OUTPUT (P)rinter (F)ile ?	Ausgabe zusätzlich auf (P)rinter oder in ein (F)ile	<ET> => nur auf Schirm P => zusätzlich an Drucker oder F => zusätzlich in ein File
DATA FIRST: LAST:	 Datenbereichsanfang manuell Datenbereichsende manuell	 <ET> keiner <ET> keiner

Der Disassembler kann selbst keine Datenbereiche (z.B. Texte) identifizieren. Er würde sie stur in Mnemonics umsetzen. Man kann jedoch Datenbereiche manuell festlegen (erstes und letztes Datenbyte). Dann werden diese Bereiche nicht übersetzt sondern als "DEFB xx" ausgewiesen

Wurden bei der Abfrage "DATA FIRST/LAST" Werte für einen Datenblock angegeben, so erfolgt eine erneute DATA-Abfrage für einen eventuellen weiteren Bereich. Das wiederholt sich solange, bis nur <ET> bei FIRST/LAST eingegeben wird. Es lassen sich auf diese Weise mehrere DATA-Bereiche angeben.

Bei der Ausgabe in ein File wird ein Dateiname erfragt und automatisch um die Erweiterung ".SRC" ergänzt. Ein eventuell vorhandenes gleichnamiges File wird ohne Rückfrage gelöscht!

Die Disassemblierung kann vorzeitig mit Strg+C abgebrochen werden.

Sprungziele werden entweder mit ihrer Adresse (z.B. JR Z,#0100) oder einer Markenbezeichnung (z.B. CALL L1000) angegeben. Welche Form benutzt wird hängt davon ab, wo sich die Zieladresse im Vergleich zur Adresse des aktuellen Befehls befindet:

- unterhalb der akt. Adresse => Lxxxx
- oberhalb der akt. Adresse oder außerhalb des disassemblierten Bereiches => #xxxx

(Dis-)Assembler-Quelltext

Der beim Disassemblieren in ein File ausgegebene Quelltext hat spezielles Format, um einen Import im Assembler zu ermöglichen. Beispiel einer Zeile:

06 00	4C	30	30	30	42	20	43	41	4C	4C	20	4C	30	31	31	44	0D
	L	**0**	**0**	**0**	**B**		**C**	**A**	**L**	**L**		**L**	**0**	**1**	**1**	**D**	
Zeilen-Nr. 6 als HEX-Zahl	Markenname					n LZ	Befehls-Mnemonic				n LZ	Operand(en)-Mnemonic					Zeilen-Ende

Enthält eine Quelltextzeile keine Marke, so entfällt dieses Feld. Vor der Befehls-Mnemonic steht immer ein Leerzeichen. Die Anzahl der Leerzeichen (n) ist abhängig von der Länge des vorherigen Feldes.

Infolge der hexadezimalen Darstellung der Zeilennummer ist es keine reine ASCII-Datei; sie kann in einem Texteditor regulär nicht gelesen werden (Zeilennummer wird verstümmelt).

Für den beim Disassemblieren erzeugten und in eine Datei geschriebenen Quelltext (*.SRC-Datei) gibt es theoretisch keine Grenze für die Größe. In der Praxis tritt jedoch eine Beschränkung ein: Da der Quelltext anstelle von absoluten Adressen nun Zeilennummern und Marken enthält, benötigt man eine Markentabelle. Diese wird vor der eigentlichen Disassemblierung erstellt, und ihre Größe ist von der Anzahl der verwendeten Marken abhängig. Sie darf den Systemzellenbereich (max. bis #23FF) nicht überschreiten. Passiert das jedoch, so kann der Quelltext nicht in eine Datei geschrieben werden und die Disassemblierung wird abgebrochen.

Soll der Quelltext in den Assembler des LC-80ex importiert und ggf. weiter bearbeitet werden, so darf seine Länge wegen der RAM-Grenzen ca. 16 kB nicht überschreiten! Andernfalls reagiert der Assembler mit einer "OUT OF SPACE"-Meldung.

Kommandoübersicht

Taste	Bedeutung	Hinweise, Beispiele
BACKSP	Eingabekorrektur	letztes Eingabezeichen wird gelöscht
→	4 Adressen vor	Mit den Kursortasten ändert man die aktuelle Adresse "MP" in der Hauptansicht schrittweise um +/-1, +/-4). Für direkte Angaben siehe M-Kommando.
↓	1 Adresse vor	
←	4 Adressen zurück	
↑	1 Adresse zurück	
Strg+A	(A)ssembleransicht	Schnell-Disassembler, Anzeige ab MP, zurück mit Strg+C, andere Taste setzt fort
Strg+B	(B)reakpoint	Haltepunkt auf MP setzen nur im RAM, nicht im ROM-Bereich möglich!
Strg+C	allgemeines Abbruchkommando	Rückkehr zur Hauptansicht aus Befehlsanzeige oder HEX-Dump. Abbruch eines Kommandos oder einer Eingabe (z.B. Filename, Adresse)
Strg+D	(D)isassembler	siehe extra Kapitel
Strg+E	(E)xecute	Programm ab PC fortsetzen (bis Haltepunkt) ggf. PC vorher auf gewünschte Adresse setzen!
Strg+N	Prüfsumme bestimmen	Methode wie am AC1/LLC2
Strg+O	Undo zu O	Offset aufsuchen rückgängig
Strg+P	(P)rint Hex-/ASCII-dump	ab MP (ggf. auf "runde" Adresse stellen) LAST: Abfrage letzte zu druckende Adresse ohne Halt, Bit7 (Grafikzeichen) immer =0
Strg+X	Undo zu X	JP/CALL aufsuchen rückgängig
+	Einzelschritt	ggf. PC vorher auf gewünschte Adresse setzen! auch im ROM möglich, aber Bedingung: kein Interrupt!
!	Einzelschritt	fortlaufendes Setzen eines Haltepunktes nach der akt. Anweisung, dann Ausführung aktuelle Anweisung, nur im RAM möglich
.	Register auswählen/ modifizieren	. wählt gewünschtes Register akt. Register ist mit * markiert (SP und IR nicht möglich) 1000. setzt akt. Register auf 1000
0...9	Dezimalziffern	z.B. Speicher beschreiben: #FF <ET> trägt FF auf (MP) ein #55 <down> trägt 55 auf (MP) ein und schaltet zur nächsten Adresse
A...F	Hexadezimalziffern	
G	(G)et Byte	Bytefolge ab MP suchen Abbruch Eingabe: Strg+C, nächstes finden: "N"
H	(H)elp	Kurzhilfe wichtigste Kommandos
I	(I)ntell. Kopieren Überlappung zulässig	COPY FIRST: Bereichsanfang, LAST: Bereichsende TO: Anfang Ziel Korrektur mit BACKSP, Eingabeabbruch: Strg+C

J	(J)ump Starte ab angegeb. Adresse	J:4000 um danach wieder in "MCE" zurückzukehren: am Ende des auszuführenden Programms einen Haltepunkt setzen
K	(K)ill	Speicher löschen 2400...(7FF0) muss mit "Y" bestätigt werden
L	(L)ist Hexdump ab akt. Adresse	Steuerzeichen (<20h) werden als "." im ASCII-Bereich dargestellt zurück mit Strg+C, andere Taste setzt fort
M	(M)emorypointer aktuelle Adresse setzen	nur Tasten 0...9/A...F zulässig negative Angaben: "-" dahinter (1- => FFFF) alle anderen Tasten brechen ab (Adresse=0) Korrektur mit BACKSP
N	(N)ext	nach G das nächstes Byte finden
O	(O)ffset zu akt. Adresse aufsuchen	MP auf Offset eines Relativsprungs setzen, dann O => springt dorthin Strg+O: springt zurück
P	(P)attern füllt Bereich mit Byte	PATTERN FILL FIRST: 4000 LAST: 4FFF WITH: 55
Q	(Q)uit	zurück zum Monitor/Betriebssystem (RESET) LED-Anzeige leuchtet wieder
R	(R)ead file Laden MC-Programm auf USB als Z80	Filename:TEST ;Dateiname ohne *.z80!
S	(S)tackadresse in MP	akt. Adresse auf die im Stackpointer setzen
T	(T)ext ab aktueller Adresse einen String eingeben	String-Ende mit Strg+C Korrektur letztes Zeichen mit BACKSP kein Abbruch möglich
U	(U)SB-Inhaltsverzeichnis	20 Dateien, dann Stopp. Weiter mit beliebiger Taste oder Abbruch mit Strg+C
V	(V)iew	Registeranzeige aus-/einschalten
W	(W)rite file Abspeichern MC-Programm auf USB als z80	Filename:TEST ;Dateiname ohne *.z80 First:2400 ;Anfangsadresse Last :2500 ;Endeadresse
X	gehe zu JP/CALL Adresse	erst MP auf low-Byte Ziel setzen, dann X => springt dorthin Strg+X: springt zurück
Y	ext. Programm aufrufen	(E)EPROM auf C000, z.B.: ASSEMBLER 8K-BASIC (Kaltstart)
Z	(Z)ap Registersatzanzeige	Wechsel zwischen Normal-/Schattenregister, letztere sind markiert mit '