caz

COLLABORATORS			
	TITLE : caz		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		September 19, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

caz

1.1 Inhalt 'caz' Dokumentation

Januar
,94

C.Rose

Z80 Cross-Assembler für den Amiga

Version: 1.26ß Beta

Inhaltsverzeichnis:

1.2 copyright

11.Januar.1993

Dieses Programm ist Freeware und soll verteilt werden. Fuer die kommerzielle

Nutzung bedarf es der ausdruecklichen Genehmigung des Autors. PD-Disketten
auf denen sich 'caz' befindet duerfen nicht mehr als 10 DM kosten.

Autor: Carsten Rose EMail: caro@uni-paderborn.de Tel.:05251/27323
Leostr.35
33098 Paderborn

Deutschland

1.3 einführung

Beschreibung zum Z80 Assembler 'caz' V1.26 Beta

Der Assembler ist komplett in Standard ANSI 'C' geschrieben und sollte mit
jedem ANSI C Compiler zu übersetzen sein. 'caz' ist der erste Assembler den ich
geschrieben habe - und genau so war die Entwicklung: Ich hab viel
gelernt wie man es nicht machen sollten :-)

Generell: Ich bin fuer jeden Bugfix dankbar. Wer Verbesserungsvorschlaege hat
soll sich melden - ich warte.

Was kann/versteht 'caz':
- Die meisten Z80 Direktiven (ich hab die Auswirkung/Funktion einiger
  Direktiven nicht ganz verstanden !)
- Erzeugung von IntelHex Dateien (zum Brennen von Eproms).
- Optional die Ausgabe der Taktzyklen die jeder Befehl benötigt.

Was kann 'caz' noch nicht:
- Macros ;-(
- Folgende Kommandos versteht 'caz' nicht:
  HEADING S,MACLIST ON,MACLIST OFF <- kann mir einer sagen wofuer die gut sind ?

Bekannte Fehler: siehe
Bekannte Fehler

1.4 aufruf

Aufruf:

caz DATEINAME/A
OPTIONEN
DATEINAME: Der Dateiname des Assembler Quelltextes muß komplett ↩
angegeben
werden (also z.B. 'test.asm').

Wird 'caz' ohne oder mit falschen Optionen aufgerufen, so gibt 'caz'
einen kurzen Hilfstext über die Optionen aus.

1.5 optionen

```
OPTIONEN: Je nach Option muß evtl. ein weiteres Argument angegeben werden.
          Für jede Option existiert eine Voreinstellung.

-o [Output Filename]
 Dateiname unter dem der erzeugte Objektcode gespeichert wird.
 Voreinstellung: 'a.out'.

-d/D
 Nur für 'caz' Testzwecke, Interner Debug Modus.
 Voreinstellung: AUS

-v/V
 Verbose Modus: Gibt den Source Text zusammen mit dem erzeugten Objektcode auf
 STDOUT aus.
 'v' - Leerzeilen und reine Komentarzeilen werden nicht angzeigt.
 'V' - Alle Zeilen werden angezeigt.
 Voreinstellung: AUS

-c/C
 Clock Cycle Modus: Gibt den Source Text zusammen mit dem Taktzyklen eines
 jeden Mnemomics aus.
 'c' - Leerzeilen und reine Komentarzeilen werden nicht angzeigt.
 'C' - Alle Zeilen werden angezeigt.
 Voreinstellung: AUS

-w [Filename]
 Schreibt die Symbol Tabelle im Z80 Format in das File [Filename]
 Damit ist ein eingeschränktes 'linken' von mehreren Programmfiles möglich.
 Alle Konstanten werden als 'EQU' Anweisung mit ihrem Wert ausgegeben.

-s
 Symbol Table Modus: Gibt die 'caz'-interne Symbol Tabelle aus. Dies sind
 Konstanten und Label.
 Voreinstellung: AUS

-i [IntelHex Filename]
 Dateiname unter dem die erzeugte 'IntelHex'-Datei gespeichert wird.
 Voreinstellung: AUS (keine Erzeugung der 'IntelHex'-Datei)

Bemerkungen zu der Benutzung der Optionen:
- Die Optionen 'c' und 'v' schließen sich aus. Bei einer Assemblierung darf nur
  immer eine Option angegeben sein.
```

1.6 syntax

```
Beschreibung der Syntax

- 'caz' arbeitet zeilenorientiert. D.h. Jede Anweisung muß innerhalb einer
  Zeile abgeschlossen sein. Pro Zeile darf nur eine Anweisung (Mnemomic
  oder Direktive) und optional ein Label und Kommentar stehen.
  Eine Zeile darf maximal 1024 Zeichen lang sein.

- Groß/Klein-schreibung wird nur für Label beachtet, also nicht für Mnemomics,
  Register und Direktiven.
```

- Auf welcher Zeilenposition Mnemomics,Direktive oder Labels beginnen
  ist freigestellt.

- Label werden durch einen ':' abgeschlossen.
  - Zwischen dem Label und dem ':' darf kein Freiplatz sein !
  - Label müssen immer die erste Anweisung in einer Zeile sein.
  - Label sollten nicht zusammen in einer Zeile mit einer 'ORG' Anweisung
    stehen - das Label bezieht sich dann auf eine falsche Adresse (die der
    vorausgehenden Zeile) !

- '$' ist das Zeichen für den Addresszähler. Wo dieses Zeichen vorkommt, wird
  es durch den Inhalt des Adresszählers ersetzt.

- Kommentare: Zeichen ';'
  Alles was nach dem Semikolon kommt, ist Kommentar - wird also nicht
  mehr beachtet.
  Ausnahme: Falls ein Semikolon in einem Textstring (DEFM Anweisung) steht,
            ist es natürlich kein Kommentarzeichen mehr !

- Texte müssen entweder von einfachen Anführungszeichen ' (Tastenkombination
  ALT-ä), doppelten Anführungszeichen " oder von Schrägstrichen / einge-
  klammert sein.
  Ein 'geklammertes' ASCII Zeichen wird als Konstante interpretiert:
     LD C,'A' entspricht LD C,41h.

- Mathematische Ausdrücke:
  Unterstützt werden die 4 Grundrechenarten +,-,*,/ sowie Klammerung.
  Gerechnet wird Punkt vor Strich.
  Die Länge des Ausdruks, die Anzahl der Klammer und die Klammerungstiefe sind
  nicht beschränkt.

- Konstanten:
  Die Kontsante muß nach einem Durchlauf (Pass 1) vollständig auflösbar sein.

  Ist keine Basis angegeben, so wird die DEFAULT Basis benutzt. Diese ist
  voreingestellt auf 10 und kann mit der Derektive DEFBASE verändert werden.

  - Binär   : % vor der Zahl oder b/B am Ende   %1010011,1010011b
  - Octal   : o/O am Ende   123o, 123
  - Decimal   : Nichts oder d/D am Ende   83 , 83d
  - Hexadecimal : x/X/0x/0X/# am Anfang
                  oder h/H am Ende   x53,0x53,#53,53h

- Folgen dem Mnemomic Argumente, so muß zwischen Mnemomic und dem 1. Argument
  mindesten 1 Freiplatz (Tabulator oder Space) sein.

- Bei DEFB/DEFW/DEFM können mehrere Argumente, durch Kommata getrennt,
  angegeben werden.

- Bei DEFM dürfen Text (natürlich in Anführungszeichen) und Zahlen gemischt
  werden.

1.7 direktiven

- Direktiven:

INCLUDE	Lädt Header/Programm Dateien in den Assembler Quellcode. Die Schachtelungstiefe der INCLUDE-Anweisungen ist nicht beschränkt
ORG nn	Setzt den Befehlszähler auf 'nn'
name EQU nn	Weist dem Symbol 'name' den Wert 'nn' zu.
name DEFL nn	Wie EQU, darf aber für ein und das selbe Symbol mehrmals benutzt werden.
DEFB n	Weist der Speicherzelle, auf die der Befehlszähler zeigt, den Wert 'n' zu. Mehrere Bytes können durch Komma getrennt, angegeben werden.
DEFW nn	Wie DEFB, aber Zweibyte Werte 'nn' werden zugewiesen.
DEFS nn	Reserviert einen Speicherblock der Länge 'nn'
DEFM 'string'	Speichert die Zeichenkette 'string'.
END	Beendet Assemblierung, falls noch weiterer Sourcecode folgt, wird dieser ignoriert.
LIST ON	Schaltet den 'Verbose' Modus an.
LIST OFF	Schaltet den 'Verbose' Modus aus. Ist der 'Verbose' Modus mittels der 'caz' Option -v oder -V schon gesetzt, so wird LIST ON/OFF ignoriert
COND nn	Bedingte Assemblierung. Ist 'nn'==0 (FALSE) wird bis 'ENDC' gefunden wurde, nichts assembliert. Ist 'nn'<>0 (TRUE), wird der folgende Quelltext ganz normal assembliert.
ENDC	Endezeichen für 'COND'.
EJECT	Führt einen Seitenvorschub aus, falls 'Verbose' oder 'Clockcykle' Modus aktiv sind.
DEFBASE name	Setzt die Default Zahlenbasis. Werden Zahlen gelesen bei denen keine explizite Basis angegeben ist, so wird die Default Zahlenbasis benutzt. Ist nichts definiert, so ist dies 10. Folgende Basen sind möglich:

```
DEFBASE Hex       16-Basis
DEFBASE Decimal   10-Basis
DEFBASE Octal      8-Basis
DEFBASE Binär      2-Basis
```

1.8 hinweise

Hinweise:

- Bei Mnemomics mit den Registern IX/IY und Offset gilt: Auch wenn der Offset 0 ist, so muß er trotzdem angegeben werden, z.B.: LD A,(IX+0) .

- Label sollten nicht zusammen in einer Zeile mit einer 'ORG' Anweisung stehen - das Label bezieht sich dann auf eine falsche Adresse !

- Im Objekt File sind nicht initialisierte Speicherbereiche mit 0x0 vorbesetzt (DEFS jedoch mit 0xFF).

1.9 fehlermeldungen

Fehlermeldungen:

Label too long (Max 32 chars)
- Der Name des Labels ist länger als 32 Zeichen.

Label declared twice
- Ein Label oder Konstante wurde zweimal definiert.

Symbol declared with EQU and DEFL
- Ein Symbol wurde einmal mit EQU und einmal mit DEFL deklariert. Entweder immer mit DEFL oder nur einmal mit EQU deklarieren.

No start adress specified (use 'ORG')
- Eine Zuweisung in den Objektspeicher (Objektcode eines Mnemomics oder eine Direktive) sollte gemacht werden, aber es ist noch keine Adresse festgelegt worden, wohin dies erfolgen soll.

No closing bracket found
- Bei einer geklammerten Anweisung (Mathematischer Ausdruck oder indirekte Adressierung) fehlt die schließende Klammer.

Expect any expression inside
- In einer geklammerten Anweisung fehlt der Ausdruck.

Expect any expression behind '+'
- Nach dem '+' folgt kein weiterer Ausdruck.

Offset too big (must be <256)
- Der angegebene Offset muß kleiner als 256 sein (indirekte Adressierung).

Syntax error
- Mit der Anweisung kann 'caz' nichts anfangen.

Unknown Argument (Label,Number,Branch)
- Das angegebene Argument ist falsch, oder nicht zu interpretieren.

Jump destination not in Range
- Das Ziel des Sprunges ist zu weit entfernt - absoluten Sprung benutzen.

RST only with 0/8/10/18/20/28/30/38h allowed
- Falsche RST Nummer angegeben.

Data overwrites previous assembled data
- Der erzeugte Objektcode überschreibt sich selbst. ORG Anweisungen anpassen.

Missing end quote: \' or /
- Start/Ende-kennzeichen für den Textstring fehlt.

Need a symbol name at first
- Bei einer EQU oder DEFL Anweisung fehlt der Symbolname.

Unknown Mnemomic
- Unbekannter Mnemomic.

Unknown Command
- Unbekanntes Kommando.

```
Expect any constant value
- Eine Konstante wurde in einem Ausdruck erwartet, aber nicht gefunden.

Symbol Name redefined
- Der Symbolname wurde redefiniert. Einmal benutzte Namen dürfen nicht neu
  definiert werden.

Can't resolve definition
- Die Definition konnte in 'Pass 2' nicht aufgelöst werden.

No Argument allowed here
- Das Mnemomic hat keine Argumente.

Need an Argument
- Das Mnemomic benötigt ein weiteres Argument.

Need a second Argument
- Das Mnemomic benötigt ein zweites Argument.

Need two Arguments
- Das Mnemomic benötigt zwei Argumente.

Wrong type of Argument
- Der Typ des Arguments ist für diesen Mnemomic nicht möglich,
  z.B. ADC B,C - es kann nicht in das B Register addiert werden.

Wrong Argument combination
- Die Argumente sind in dieser Kombination für das Mnemomic nicht erlaubt.

Need an Argument (ON or OFF)
- Die Direktive 'LIST' benötigt ein Argument: ON oder OFF, je nachdem ob der
  'Verbose' Modus eingeschaltet werden soll oder nicht.

Directive not implemented
- Die angegebene Direktive ist nicht in 'caz' implementiert.

Expect comma
- Die angegebenen Argumente müssen durch Kommas getrennt sein.

Unknown default number base
- Die definierte DEFBASE ist unbekannt (h/d/o/b)

Assembler need more memory (malloc failed)
- Der Hauptspeicher des Computers ist voll. Nicht benötigte Programme entfernen
  und nochmal assemblieren.

Can't open file:YYY
- 'caz' konnte die Datei YYY nicht öffnen - Datei nicht vorhanden, oder
  screibgeschützt ?

Can't get position:XX of File:YYY
- 'caz' konnte die Position XX in dem File YYY nicht anspringen - DOS Fehler.

Only 1 mode allowed: 'Clock Cycle' or 'Verbose'
- Zu einer Zeit ist immer nur einer der beiden Modi erlaubt.
  Werden die Ausgaben beider Modi benötigt, muß 2 mal Assembliert werden.
```

Unknown option
 - Eine, beim Aufruf von 'caz' angegebene, Option ist unbekannt.

1.10 knownbugs

Bekannte Fehler:

- Fehlermeldungen erscheinen teilweise doppelt