Top-Der erste für den

Endlich ist er da, der erste Wir haben ihn für Sie

Entstanden ist der Assembler aus dem in Ausgabe 7/85 beziehungsweise Sonderheft 8/85 veröffentlichten Hypra-Ass. Jedoch ist Top-Ass, wie der neue Assembler heißt, um viele Funktionen erweitert worden. Außerdem befinden sich auf der Diskette ein zusätzlicher Monitor und Programme, die das Arbeiten mit Top-Ass zum Genuß werden lassen.

Der Editor

Die Geschwindigkeit, mit der Top-Ass einen Quelltext in Maschinensprache übersetzt, liegt um ein Vielfaches höher als beim Hypra-Ass. Erreicht wurde dieser Geschwindigkeitszuwachs, weil der Quelltext nicht im ASCII-Format im Speicher abgelegt wird, sondern als Token. Daraus resultiert ein weiterer Vorteil. Top-Ass überprüft nämlich schon während der Erstellung des Quelltextes auf eventuell vorhandene Syntax-Fehler und zeigt sie an. Pseudo-Opcodes, also Steueranweisungen an den Assembler, können wie ganz normale Basic-Befehle abgekürzt eingegeben werden. Sobald RETURN gedrückt wird, erscheinen sie voll ausgeschrieben auf dem Bildschirm. Außerdem wird die jeweils zuvor bearbeitete Quelltextzeile genauso wie bei Hypra-Ass formatiert ausgegeben. Die Editorbefehle sind weitgehendst identisch zu denen von Hypra-Ass. Die einzigen Befehle, die neu hinzugekommen sind, beziehen sich auf die formatierte Ausgabe des Quelltextes. Unter Top-Ass läßt sich mit Hilfe der Editorbefehle ».-« beziehungsweise ».+« die formatierte Ausgabe aus- beziehungsweise einschalten.

Der Assembler

Die eigentlichen Vorteile von Top-Ass gegenüber jedem anderen Assembler ist der Top-Assembler selbst. In ihm wurden alle Besonderheiten der zur Zeit erhältlichen Assembler für den C 64 eingebaut. So liegt zum Beispiel die maximale Verschachtelungstiefe für Makros bei über 80 Makros, bei denen es sich bekanntlich um kurze, häufig auftretende Befehlsfolgen handelt. Dies gehört heute zum Standard eines guten Assemblers. Was aber die wenigsten beherrschen, ist die Definition eines Blocks. Innerhalb eines Blocks sind alle Label und Variablen lokal. Dies erleichtert das Einfügen von Quelltext ganz erheblich und erhöht die Übersichtlichkeit des Programms. Denn man braucht sich keine Gedanken mehr darüber machen, ob einige Label vielleicht doppelt verwendet wurden. Definiert man einen Quelltextteil als Block, so ist es gleichgültig, ob zum Beispiel das Label »LOOP« schon einmal außerhalb des Blocks benutzt wurde. Der Assembler erkennt automatisch, daß es sich bei den beiden Label um unterschiedliche Label handelt.

Neben den normalen, frei definierbaren Makros sind im Top-Ass schon sogenannte Minimakros integriert. Sie enthalten Befehle für 16-Bit-Operationen und zur bedingten Verzweigung über den gesamten 64-KByte-Bereich. Das Besondere an den Minimakros sind die eingebauten Befehle zur strukturierten Programmierung. Durch sie wird die Programmierung in Assembler fast so leicht wie in Basic. Im einzelnen enthalten die Minimakros folgende Strukturen:
REPEAT / UNTIL; DO / WHILE; IF / ENDIF; CASE OF / CASEEND.

Alle Schleifentypen haben noch zusätzliche Abbruchbedingungen.

Einige Befehle zur bedingten Assemblierung sind bei Top-Ass leider unter den Tisch gefallen. Übrig geblieben ist noch die bedingte IF / ELSE / ENDIF- und eine Art CASE OF-Konstruktion. Dafür hat sich der Autor jedoch einiges zur Verkettung von Quelltextteilen einfallen lassen.

Bei Top-Ass existieren zwei unterschiedliche Methoden zur Verkettung von Quelltexten. Die erste, wohl bekannteste Methode ist die, jeden Quelltextteil zweimal in den Speicher zu laden — einmal im ersten und einmal im zweiten Paß. Jeder Paß erstreckt sich bei dieser Verkettungsart, der sogenannten »chain«-Verkettung, über alle Quelltextteile. Dadurch werden alle Label und Variablen des Gesamtquelltextes verfügbar gemacht, indem alle Label- und Variablennamen in die Namenstabelle beziehungsweise in die »Symboltabelle« aufgenommen werden. Die zweite Verkettungsart funktioniert vollkommen anders. Hier wird jeder Quelltextteil vor dem Nachladen des nächsten vollständig, also in Paß 1 und 2, assembliert. Erst nach erfolgreicher Assemblierung wird der nächste Quelltextteil nachgeladen. Dabei werden nach der Assemblierung alle Label und Variablen in der Symboltabelle gelöscht. Das heißt, daß alle Einträge in der Symboltabelle für jeden Quelltextteil lokal sind. Von anderen Quelltextteilen kann auf die zuvor definierten Variablen nicht zurückgegriffen werden. Daraus folgt natürlich, daß eine solche Verkettung ziemlich witzlos wäre, könnte man sich in keinster Weise im nachgeladenen Qelltextteil auf vorangegangene beziehen. Zu diesem Zweck existiert ein Befehl, der einzelne Quelltextbereiche vor dem Löschen beim Nachladen schützt. Der Befehl heißt »common« und hat zwei Aufgaben.

1. Der nachzuladende Quelltextteil wird hinter der letzten »Common«-Zeile angefügt. Bei der anschließenden Assemblierung werden die Zeilen im Common-Bereich noch einmal durchlaufen. Liegen in dem Bereich Label- oder Variablen-Definitionen, so sind diese auch für die neue Assemblierung gültig.

2. Der Common-Befehl selbst darf Labelnamen enthalten. Solche Label sind dann automatisch von der Löschung der Symboltabelle ausgenommen.

Natürlich lassen sich mit Top-Ass nicht nur Quelltextteile verketten, sondern auch einbinden. Dieses geschieht mit dem Befehl »source«. Ei-

Ass: Assembler C 128

Assembler für den C 128. ausführlich getestet.

ne Einbindung von Quelltext liegt dann vor, wenn der nächste Teil des Gesamttextes nicht am Ende, sondern innerhalb des aufrufenden Textes angesprochen wird. Daraus folgt, daß der eingebundene Quelltextteil direkt von der Diskette bearbeitet wird. Nach erfolgreicher Bearbeitung wird die Assemblierung hinter dem »source«-Befehl im RAM fortgesetzt.

Top-Ass ist einer der wenigen Assembler, die in der Lage sind, relokatible Module zu erzeugen. Zuerst einmal ein paar Worte dazu, was ein relokatibles Modul ist. Bei Top-Ass handelt es sich hierbei um ein File auf Diskette. Dieses File stellt kein lauffähiges Maschinenprogramm dar, sondern eine Art Zwischencode, der den Relativlader und den eingebauten Linker in die Lage versetzt, aus diesem File ein lauffähiges Maschinenprogramm zu erzeugen, das an einer frei wählbaren Startadresse liegen darf.

Das Besondere nun ist der eingebaute Linker. Durch ihn lassen sich größere Programme in mehrere Teile splitten, deren relokatible Module man nach völlig getrennter Assemblierung mit Hilfe des Relativladers zusammenbinden kann. Dies ist für die Entwicklung größerer Programme sehr nützlich, denn Änderungen des Programms ziehen dann nicht eine Assemblierung des gesamten Quelltextes nach sich, sondern es muß nur der Quelltextteil assembliert werden, in dem eine Änderung stattgefunden hat. Alle anderen Teile liegen ja in Form von Modulen vor, die im Speicher beliebig hin- und hergeschoben werden können. Die Assemblierung einzelner Quelltextteile wäre aber sinnlos, wenn man aus einem Modul heraus nicht auf andere Module zurückgreifen könnte; zum Beispiel Unterprogrammaufrufe oder gemeinsame Variablen. Dazu existieren bei Top-Ass zwei zusätzliche Pseudo-Opcodes »extern« und »public«. Diese beiden Pseudos könnte man auch als Kopf eines Moduls bezeichnen. Sollen zum Beispiel von einem Modul Unterprogramme aus einem anderen Modul aufgerufen werden, so ist der Labelname im aufrufenden Quelltextteil und zwar in der ersten Zeile, als extern zu deklarieren. Das setzt natürlich voraus, daß in dem Modul beziehungsweise in dem dazugehörigen Quelltext, der das Unterprogramm enthält, ebenfalls in der ersten Zeile der Labelname als »public« deklariert wurde.

Der Monitor

Die Top-Ass-Diskette enthält zusätzlich noch einen »Splitscreen-Monitor« in zwei Versionen, als Maschinenprogramm und als Relativlader. Die Bildschirmsteuerung kann zwischen verschiedenen Bildschirmtypen umschalten. Man hat einmal die Möglichkeit, Dumps wie gewohnt auf dem 40-Zeichen- beziehungsweise 80-Zeichen-Bildschirm auszugeben. Zum anderen kann der Bildschirm in zwei Fenster zu je 20 beziehungsweise 40 Zeichen gesplittet werden. Beide Fenster liegen parallel nebeneinander, lassen sich getrennt bearbeiten und, was nicht zu unterschätzen ist, nach oben und unten verschieben (scrollen). So läßt sich auf der einen Bildschirmhälfte zum Beispiel ein Hexdump und auf der anderen ein Disassemblerlisting ausgeben. In beiden Bildschirmhälften kann durch einfaches Überschreiben editiert werden. Die Befehle des Monitors entsprechen dem Standard. Allerdings wurde ein besonderer Wert auf den Suchbefehl gelegt, der nicht nur hexadezimale Zahlen sucht, sondern auch ASCII-Zeichenketten. Der interessanteste Suchbefehl ist das Suchen von Befehlen innerhalb eines Programms. Hier wurde ein Konzept gewählt, das es erlaubt Befehle, nach denen gesucht werden soll so einzugeben, wie sie im Disassemblerlisting erscheinen würden. Auch Joker, die entweder durch ».« oder »*« gekennzeichnet werden, sind erlaubt.

Das Aufspüren und Entfernen von Programmfehlern wird vom Monitor durch eine sehr leistungsfähige Breakpoint-Behandlung unterstützt.

Dabei werden drei Arten von Breakpoints unterschieden:

1. Der unbedingte Breakpoint. Dieser führt auf jeden Fall zum Abbruch des Testprogramms.
2. Der bedingte Breakpoint. Er wird nur dann ausgelöst, wenn ein Breakpoint n-mal durchlaufen wird. Das »n« muß natürlich zuvor definiert werden.
3. Der Userbreakpoint. Dieser führt nicht direkt zum Abbruch des zu testenden Programms, sondern verzweigt in eine vom User geschriebene Routine. In ihr wird erst entschieden, ob das Programm fortgesetzt oder beendet werden soll.

Insgesamt lassen sich zehn bedingte beziehungsweise unbedingte und fünf User-Breakpoints gleichzeitig setzen. Zusätzlich zu der Anzeige der Registerinhalte, die auch von anderen Monitoren vorgenommen wird, erlaubt dieser Monitor als Leckerbissen die Anzeige ausgesuchter Speicherbereiche — sogenannter Hot Spots — während der Breakpoint-Behandlung. Ein »Hot Spot« wird in Form einer Hexdump-Zeile angezeigt. Auf diese Weise läßt sich auch der Inhalt flüchtiger Speicherzellen festhalten.

Fazit

Für 89 Mark erhält man ein komplettes Maschinensprachepaket für den C 128, das neben dem Assembler einen Monitor enthält und die Möglichkeit relokatible Module zu erzeugen. Das Programmpaket, das nicht nur für den Profi entwickelt wurde, läßt keine Wünsche offen und ist jedem zu empfehlen, der in Assembler auf dem C 128 programmieren will. Es ist alles vorhanden, selbst Funktionen, von denen der verwöhnte C 64-Nutzer bisher nur geträumt hat. (ah)

Info: Markt & Technik Verlag AG, Hans-Pinsel-Str. 2, 8013 Haar bei München, Tel.: 089/46 130