Darf es etwas mehr sein?

Steckmodule verbrauchen fast immer Speicherplatz. Business Basic erweitert ihn auf 61 KByte RAM und bietet zudem über 50 neue Befehle.

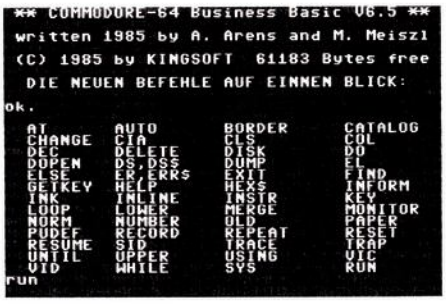
Der Prozessor des C 64 (MOS 6510) ist als 8-Bit-CPU in der Lage, genau 65536 Speicherzellen zu adressieren. Beim C 64 sind, im Gegensatz zum VC 20, alle diese Speicherzellen mit RAM-Bausteinen bestückt. Trotzdem bleiben nur 38911 Byte zum Programmieren in Basic übrig, denn 24 KByte ROM, 2 KByte für Zeropage und Bildschirmspeicher fordern ihren Tribut. Der Rest ist nur für Programme in Maschinensprache zugänglich. Diesem Manko hilft Business Basic von Kingsoft gründlich ab. Es ist schon beeindruckend, wenn man in der neuen Einschaltmeldung »61183 Bytes free« (Bild) lesen und über dieses Speichervolumen in Basic auch frei verfügen kann.

Insgesamt verfügt der C 64 mit Business Basic über 96 KByte (64 KByte RAM + 28 KByte ROM). Das Geheimnis, das hinter diesem Konzept steckt, ist die konsequente Weiterführung eines schon im Standard-C-64 vorhandenen Prinzips. Es wird, je nachdem welche Funktion benötigt wird, zwischen verschiedenen Speicherbereichen, ROM und RAM umgeschaltet. Für den Computer ist also immer nur der Teil des Speichers »sichtbar«, den er gerade benötigt. Dieses sogenannte »banking« hat aber auch seine Nachteile, denn es treten Geschwindigkeitseinbußen von zirka 18 bis 25 Prozent auf. Dafür wurden aber einige leistungsfähige Befehle und Programmiertricks implementiert, die diesen Nachteil relativieren. So dauert die gefürchtete Garbage Collection jetzt maximal 1,5 Sekunden, statt wie bisher bis zu 20 Minuten. Man kann jetzt also ohne weiteres mit sehr großen Stringfeldern arbeiten.

Basic-Befehle wie beim C 16

Wem die Ablaufgeschwindigkeit der unter Business Basic geschriebenen Programme trotz der Beseitigung der Garbage Collection zu gering ist, kann sie mit einem gleichfalls von Kingsoft angebotenen Spezialcompiler noch um einiges vervielfachen.

Das ist aber nur ein Teil der Vorteile, die das Modul zur Verfügung stellt. Denn neben den genannten Leistungen wird noch eine Basic-Erweiterung geboten, die allerdings speziell die Programmierung von Anwenderprogrammen unterstützt (daher auch der Name des Moduls). Bei der Namenswahl der



Die neuen Befehle

neuen, zusätzlichen Befehle (Bild) wurde darauf geachtet, daß die Syntax und die Tokencodierung der des Basic 3.5 entspricht. Die Programme des C 16 und Plus/4 (beide mit Basic 3.5 ausgestattet) funktionieren aber nur sehr eingeschränkt. Eine Anpassung dürfte aber nicht allzu schwer sein.

Zu den zusätzlichen Befehlen von Business Basic gehört zunächst eine sinnvolle Erweiterung der Fähigkeiten des Basic-Editors. Es gibt Befehle wie AUTO zur automatischen Zeilennummervorgabe, DUMP zum Anzeigen der Variableninhalte, NUMBER zum Umnumerieren eines Programmes mit Anpassung aller GOTO- und GOSUB-Befehle, KEY zur Funktionstastenbelegung und FIND zum Auffinden bestimmter Programmteile. Interessant ist der CHANGE-Befehl, der ähnlich der FIND-Anweisung bestimmte Stellen des Programmes sucht und gegen neue Ausdrücke ersetzt.

Bei Fehlern im Programmablauf wird die fehlerhafte Zeile angezeigt und der Cursor an die betreffende Stelle gesetzt. Diese Eigenschaften lassen sich durch HELP ON/OFF ein- beziehungsweise ausschalten. Es gibt auch den Befehl TRACE, mit dem man den Programmablauf anhand der ausgedruckten Zeilennummern verfolgen kann. Dieser Befehl läßt sich auch programmieren, so daß man auch Teile des Programmes gezielt beobachten kann. Das ist besonders dann wichtig, wenn eine lange Leseschleife am Anfang des Programms steht. Außerdem kann man jetzt, wie bei Exbasic Level II, den Ausdruck des Listings auf dem Bildschirm mit den Cursortasten vor- und rückwärts steuern. Ferner ist der LIST-Befehl ebenfalls programmierbar, so daß kein Abbruch mehr nach der Ausführung dieses Befehls unter Programmkontrolle erfolgt.

Formatierte Ausgabe

Sehr nützlich bei der Arbeit mit Bildschirmmasken sind die Möglichkeiten zur formatierten Ausgabe von Zahlen und Zeichenketten mit PRINT USING, das auch auf dem Drucker funktioniert. Erwähnenswert ist hierbei die Möglichkeit mit PUDEF die Verwendung von Kommas und Punkten bei der formatierten Ausgabe von Zahlen an deutsche Verhältnisse anzugleichen (Dezimalkomma und Tausenderpunkt). Weiterhin kann man den Ort der Bildschirmausgabe mit PRINT AT und der Angabe der gewünschten Zeile und Spalte bestimmen. Diese gesteuerte Ausgabe läßt sich auch in Zeichenketten einbauen, indem man den Zeichencode, gefolgt von der entsprechenden Zeichencodierung für Zeile und Spalte, in einen String einbaut (CHR$(1)+CHR$(Zeile)+CHR$(Spalte)).

Auch Kommandos zur gesteuerten Eingabe von Zeichen fehlen nicht. Der Befehl INLINE etwa gestattet die Eingabe aller Zeichen (also auch Kommas und Doppelpunkte), während der INFORM-Befehl die Cursortasten bei der Eingabe blockiert. Ferner gibt es noch den Befehl GETKEY, der auf ein Zeichen von der Tastatur wartet und die lästige Abfrage mit einer IF-THEN-Schleife erspart.

Selbstverständlich wurden einige Befehle zur Diskettensteuerung eingebaut. Da gibt es zum Beispiel CATALOG zum Anzeigen eines Inhaltsverzeichnisses oder DISK zum Senden von Floppy-Kommandos. Der Fehlerkanal kann über die Variablen DS und DS$ abgefragt werden. Weiterhin wird die Verwendung relativer Dateien mit Befehlen wie RECORD und DOPEN unterstützt. Die Befehle LOAD, SAVE und VERIFY beziehen sich ohne Parameteranga-

be immer auf die Floppy, wobei man in einem mit CATALOG gelisteten Inhaltsverzeichnis vor das zu ladende Programm nur noch LOAD schreiben muß und kein Doppelpunkt mehr hinter den Namen notwendig wird. Auch wird jetzt mit der RUN-Taste das erste Programm von Diskette geladen und gestartet. Man kann aber auch jedes beliebige Programm von Diskette laden und starten, indem man RUN und dahinter den Namen des Programmes eintippt. Sehr interessant ist der Befehl MERGE, der nicht nur ein weiteres Programm an das im Speicher vorhandene anhängt, sondern zeilenweise einsortiert.

Auch die Anhänger der Kassettenspeicherung wurden nicht vergessen. Die Aufzeichnungsgeschwindigkeit auf Band wurde um ein Vielfaches erhöht. Trotzdem können Programme im normalen Aufzeichnungsformat geladen werden.

Zur übersichtlichen Programmierung wurden sowohl bestehende Befehle erweitert, als auch neue definiert. Das altbekannte IF-THEN wurde um die Möglichkeit der Verwendung eines ELSE-Falles bereichert. Auch kann man den Befehl RESTORE in Zusammenhang mit Zeilennummern benutzen. Ferner liefert die Abfrage ASC(CHR$(0)) keinen Fehler mehr, sondern den Wert Null.

Neu hinzugekommen sind Schleifenbefehle wie DO-LOOP-WHILE,

Fortsetzung auf Seite 132

Mathemat nicht im Test

An dieser Stelle sollte eigentlich ein Testbericht des Mathemat von Data Becker stehen. Wir ließen dieses an sich sehr interessante Programm von zwei Lehrern unabhängig voneinander testen. Beide kamen jedoch zu dem Urteil »Nicht empfehlenswert«. Zu viele Details sind unausgereift, zu viele Schwachstellen müßten noch bearbeitet werden. Selbst im Vorwort zum Mathemat liest man: »... Sollte also Ihr Examen, der Nobelpreis oder eine komplette Autobahnbrücke vom Rechenergebnis abhängen, so empfehlen wir, mit herkömmlichen Methoden noch einmal das Ergebnis zu kontrollieren.«

Aufgrund der vorliegenden Ergebnisse verzichten wir deshalb auf einen ausführlichen Bericht.

Was leistet Pilot?

Pilot ist eine wenig bekannte Programmiersprache, die speziell für Lehrzwecke entworfen wurde.

Für den C 64 sind mittlerweile zahlreiche Sprachen für jeden Geschmack erhältlich. Von Pascal über C bis zu Forth sind heute fast alle Computer-Idiome in einer auf dem C 64 lauffähigen Version vorhanden. Eine der weniger bekannten Sprachen ist Pilot, die für den C 64 von Commodore selbst angeboten wird.

Pilot ist eine sogenannte Interpretersprache. Sie verfügt über einen »Immediate-Mode« in dem alle eingegebenen Befehle sofort ausgeführt werden, einen »Edit-Mode« zur Erstellung von Programmen, den »Run-Mode« zur Benutzung der erstellten oder geladenen Programme, sowie den »Command-Mode«, von dem aus Disketten- und Drukkeroperationen gehandhabt werden. Einmal von der mitgelieferten Diskette geladen, unternimmt man unter der Anleitung der — leider in Englisch verfaßten — Dokumentation die ersten Versuche, sich mit der neuen Sprache vertraut zu machen.

Bald schon wird klar, daß einer der Schwerpunkte von Pilot die Grafik ist. Mittels verschiedener Befehle kann man Punkte setzen, Linien ziehen und Flächen ausfüllen. Der Ausgangspunkt der Koordinaten, ursprünglich in der linken unteren Bildschirmecke angesiedelt, ist dabei nach Belieben veränderbar. Der Bildschirm ist durch den Split-Befehl in bis zu fünf Fenster unterteilbar, durch die Eingabe eines einfachen Programmes können Sprites erstellt und die vorgegebenen Buchstaben der einzelnen Tasten verändert werden. Dazu gibt man das neue Zeichen oder das Sprite in Punkten und »x« ein, wobei die Punkte für Hintergrundfarbe, die »x« für hervorgehobene »Pixels« stehen, wie das auch von anderen Sprite-Editoren her bekannt ist. Das fertige Sprite kann dann auf dem Bildschirm bewegt werden und mit dem Rand oder anderen Sprites kollidieren. Auch zur Einstellung der verschiedenen Sound-Parameter hält Pilot eine Reihe von Befehlen bereit, mit denen sich eine Tonkurve genau vordefinieren läßt.

Die bisher doch recht konventionell geratene Sprache hält bei der Stringbehandlung einige Überraschungen bereit. Der Befehl »Match« ermöglicht beispielsweise das Durchsuchen eines Antwortsatzes nach einem bestimmten Begriff. Dabei läßt sich auch die Möglichkeit berücksichtigen, daß das eingegebene Wort vom Programm-Anwender falsch geschrieben werden kann: Modifiziert man den »Match«-Befehl, so akzeptiert der Computer auch ein als »commadore« geschriebenes »commodore«. Der »Jump«-Befehl nimmt, statt der in Pilot nicht vorhandenen Zeilennummern, Namen als Sprungziele. Diese Möglichkeiten erlauben die einfache Programmierung von Vokabel-Trainingsprogrammen, Grammatikprogrammen oder auch Text-Adventures. Der interessanteste Befehl ist aber sicherlich »Execute«. Dabei wird die Antwort des Benutzers als Pilot-Befehl behandelt und erlaubt so die Veränderung eines Programmes, während es läuft. Dies ist beispielsweise bei der Eingabe von mathematischen Funktionen von Vorteil, was in Basic nicht so ohne weiteres zu bewerkstelligen ist. Eines der Haupthindernisse für eine umfangreichere oder professionellere Nutzung von Pilot ist, wie auch in vielen anderen Sprachen, der zur Verfügung stehende Speicherplatz, der mit 12 KByte nicht gerade exzessiv dimensioniert ist. Zwar kann mit dem »Link«-Befehl vom Programm aus ein weiteres Programm nachgeladen werden, aber diese Möglichkeit kann sicher nicht alle Speicherprobleme lösen, da das vorhergehende Programm dabei natürlich aus dem Arbeitsspeicher geworfen wird. Auch der für Maschinenroutinen zur Verfügung stehende Bereich, der 1024 Byte umfaßt, wird für größere Anwendungen sicher nicht genügen. So wird Pilot, obschon es eine Sprache mit einigen interessanten Details ist, die teilweise recht interessante Programmiermöglichkeiten eröffnen, doch wohl eher eine Exotenrolle beschieden bleiben. Aufgrund der Einfachheit der Sprache dürfte sie aber für Anfänger, die sich den Zugriff zu den Grafik-Möglichkeiten ihres C 64 erleichtern wollen, einer Überlegung wert sein.

(Christof Bachmair/ev)

Info: Pilot gibt es nur auf Diskette für den C 64 bei Ihrem Commodore-Händler. Preis: 59 Mark

Eine **negative Zahl**
zum Beispiel RND(—1) oder RND(—95) bringt als erstes das Argument selbst (in meinem Beispiel —1 oder —95) als Gleitkommazahl in die Speicherzellen 139 bis 143, von wo sie als Samen den schon beschriebenen Manipulationen unterworfen wird. Nur — mit einem bestimmten negativen Argument erhalten Sie immer dieselbe Zufallszahl. Probieren Sie es aus:
PRINT RND(—2) ergibt immer dieselbe Zahl.

Es mag Fälle geben, wo die Vorgabe des allerersten Samens wünschenswert ist. Ich will aber von zufälligen Zahlen sprechen. Wir können diese Methode des negativen Arguments dadurch verbessern, daß wir als Argument selbst eine Zufallszahl nehmen.

Als derartige Zahl bietet sich der Wert der inneren Uhr TI an, die beim Einschalten des Computers losläuft und 60mal in der Sekunde weitergestellt wird. Da kein Mensch wissen kann, welchen Wert die Uhr TI gerade hat, kommt der Befehl RND(—TI) dem absoluten Zufall schon sehr nahe.

Das **Argument (0)**
verwendet eine andere Methode. Als Samen nimmt er eine sich ständig ändernde Zahl, die beim VC 20 aus vier Registerinhalten des VIC-Interface-Bausteins genommen werden. Beim C 64 wird es ähnlich gemacht, nur ist der VIC-Baustein ein anderer Typ.

Mit derselben Methode nach dem Einschalten wie im ersten Fall oben, können Sie das leicht überprüfen.

Ich habe eingangs zitiert, daß RND(X) eine Zahl zwischen 0 und 1 erzeugt; das gilt aber nur für ein positives Argument. Wenn Sie hingegen eine Zufallszahl innerhalb eines ganz bestimmten Bereiches brauchen, müssen Sie anders vorgehen.

Kochrezept Nr. 1

Mit folgender Formel ist der Zahlenbereich beliebig vorgebbar:
X=(RND(1)*A)+B

Die Zahl A gibt einen Bereich von 0 bis A vor.

Die Zahl B legt den untersten Wert des Bereiches fest.

Beispiele:
10 PRINT (RND(1)*6)+1:GOTO 10 erzeugt Zahlen von 1 bis 6
10 PRINT (RND(1)*52)+1:GOTO 10 erzeugt Zahlen von 1 bis 52
10 PRINT (RND(1)*6)+10:GOTO 10 erzeugt Zahlen von 10 bis 16

Mit dem Vorschalten der Funktion INT vor den Befehl RND werden die Zufallszahlen auf ganze Zahlen beschränkt.
10 PRINT INT (RND1)*6)+10:GOTO 10

Noch einmal: Zufallszahlen innerhalb bestimmter Zahlenbereiche sind gekoppelt mit einem positiven Argument. Wir haben aber gesehen, daß gerade so keine echten Zufallszahlen erzeugt werden. Deshalb brauchen wir noch ein zweites Kochrezept.

Kochrezept Nr. 2

Wenn Sie in einem Programm nach dem Einschalten des Computers immer neue Zufallszahlen brauchen, ist es empfehlenswert, für die allererste Zufallszahl RND(—TI) oder RND(0) zu verwenden, dann aber mit RND(1) fortzufahren.

Dasselbe gilt, wenn ein Programm wegen INPUT oder WAIT eine Pause hat. Nach der Pause sollte zuerst ein neuer Ausgangswert genommen werden.

Als letztes will ich noch beschreiben, wie man Zufallszahlen innerhalb von Maschinenprogrammen erzeugen kann.

Im Betriebssystem steht natürlich eine Routine für den Befehl RND. Im C 64 beginnt sie ab 57495 ($E097), im VC 20 ab 57492 ($E094).

Der Ausgangswert (Samen) wird dabei aus dem Gleitkomma-Akkumulator Nr. 1 geholt, dessen Vorzeichen oder Wert 0 das weitere Vorgehen der Routine bestimmt.

Sie müssen also den Samen in den Akkumulator Nr. 1 laden und dann mit JSR auf die RND-Routine springen. Als Resultat können Sie einen oder mehrere Werte der Zellen 140 bis 143 verwenden und nach Belieben weiterverarbeiten.

Fortsetzung von Seite 121

mit denen allgemein Schleifen wesentlich flexibler und übersichtlicher aufgebaut werden können. Dabei können 64 solcher Schleifen ineinander verschachtelt werden.

Weiterhin können Fehler im Programm mit dem Befehl TRAP abgefangen und mit den dazu vorhandenen Variablen EL, ER und ERR$ lokalisiert werden. Anschließend kann mit RESUME zu der Programmstelle zurückgesprungen werden, an der der Fehler auftrat.

Einige Kommandos zur Stringverarbeitung wurden ebenfalls eingebaut, wie etwa INSTR zur Stringsuche. Interessant ist hier die Erweiterung des Befehls MID$, der jetzt bei Zuweisungen auch links vom Gleichheitszeichen stehen kann und somit ein kontrolliertes Einsetzen von Zeichenketten in andere ermöglicht.

Die gesamte Speicherverwaltung wurde, wie am Anfang erwähnt, umgekrempelt. Deshalb wurden Befehle wie POKE und PEEK mit geändert und greifen jetzt nur noch auf den 64-KByte-RAM-Speicher zu. Um trotzdem alle Ein-/Ausgabebausteine zu erreichen, wurden einige fest installierte Variablenfelder eingerichtet. Man kann zum Beispiel mit den Variablen VIC(x) auf alle Register des Grafikbausteins direkt zugreifen, ohne einen einzigen POKE-Befehl verwenden zu müssen. Ähnliches gilt für den Soundchip und die beiden CIAs. Ferner läßt sich der Bildschirm- und Farbspeicher mit den Variablen VID(x) und COL(x) beeinflussen, wobei x von 0 bis 999 reicht. Daneben gibt es die Variablen BORDER, PAPER und INK, mit denen man die Bildschirmfarben direkt beeinflussen kann. Der Nachteil dieses Konzeptes ist es, daß Programme, die in Basic V2.0 geschrieben wurden und viele POKEs verwenden, oft nicht funktionieren.

Abgerundet wird die Palette der neuen Befehle durch Kommandos wie zum Beispiel UPPER und LOWER zum Festlegen des Groß- oder Kleinschriftmodus, CLS zum Löschen des Bildschirmes, RESET um den Einschaltzustand zu erreichen und MONITOR zum direkten Aufruf eines vorher eingeladenen Monitors, wenn dieser über einen BREAK-Einsprungspunkt verfügt. Auch der OLD-Befehl ist vorhanden, mit dem man ein NEW oder das oben erwähnte RESET wieder aufheben und damit ein Programm im Speicher wieder restaurieren kann. Ferner läßt sich mit dem Befehl STOP ON/OFF die Stop-Taste blockieren oder freigeben.

Das Handbuch erläutert all diese Befehle in verständlicher Art mit vielen Programmbeispielen.

Bemerkenswert ist in diesem Zusammenhang, daß Erweiterungen wie TurboAccess oder Hypra-Load (ROM-Version) weiterhin funktionieren.

Business Basic ist sowohl wegen der Fähigkeit mehr als 61 KByte direkt in Basic zur Verfügung zu stellen, als auch wegen seiner Qualitäten als Basic-Erweiterung sehr interessant. Nur einige Befehle zur Steuerung der hochauflösenden Grafik, die ja durchaus im Sinne dieser Erweiterung liegen würden, fehlen etwas. Der Name dieser Erweiterung ist trotzdem durchaus berechtigt, denn Business Basic erlaubt professionelles Programmieren zum Heimanwender-Preis (198 Mark).

(K. Hinsch/A. Wängler/rg)

Info: Kingsoft, Schnakebusch 4, 5106 Roetgen, Tel. 02408/8319, Preis: 198 Mark