Was ist Comal

Die Programmiersprache Comal für den C 64 soll eine Herausforderung an Pascal, Modula-2, PL/1 und C sein. Betrachtet man den Befehlsumfang und den Komfort dieser Sprache, so könnte dies zutreffen. Auch der Preis ist eine Sensation. Diese Sprache wird umsonst abgegeben.

Viele C 64-Besitzer werden beim Programmieren festgestellt haben, daß das Commodore-Basic auf ihrem Computer nicht gerade ideal ist. In diesem Fall gibt es zwei Möglichkeiten. Entweder man kauft sich eine Basicerweiterung oder man sieht sich nach einer komfortableren Programmiersprache um. Inzwischen werden für den C 64 einige Sprachen angeboten. So auch Comal.

Doch wo liegen bei dieser Sprache die Vorteile und Unterschiede gegenüber Basic und den anderen Programmiersprachen? Zuerst wollen wir den Befehlsumfang von Comal betrachten.

Comal verfügt über 69 Befehle (Tabelle 1). Um die speziellen Möglichkeiten des C 64 zu nutzen, sind

Comal Schlüsselbefehle	Bedeutung	Befehlsform
//	Kommentarzeile	//(Kommentar)
ABS	ergibt den Absolutwert einer Zahl	ABS(<numerischer Ausdruck>)
AND	logisches »und«	<Ausdruck> AND <Ausdruck>
APPEND	Start und Ende eines sequentiellen Files	OPEN(FILE)<filenum>, <filename>,APPEND
ATN	Arcustangens (rad)	ATN(<numerischer Ausdruck>)
AUTO	automatische Zeilennumerierung	(<Zeilenanfang>),(<Schrittweite>)
BASIC	zurück in Basic-Modus	BASIC
CASE	Mehrfachbedingung	CASE<Kontrollausdruck>(OF)
CAT	Directory	CAT(<Laufwerknummer>)
CHAIN	Laden und Starten von Diskettenprogrammen	CHAIN<filename>
CHR$	Characterstring	CHR$(<numerischer Ausdruck>)
CLOSE	schließt Files	CLOSE((FILE)<filenummer>)
CLOSED	Geltungsbereich von Prozedur oder Funktionsvariablen begrenzen	PROC<procname>((Parameter)) (CLOSED) FUNC<funcname>((Parameter)) (CLOSED)
CON	Programm fortsetzen	CON
COS	Cosinus (rad)	COS(<numerischer Ausdruck>)
DATA	Datazeilen	DATA<Zahl>,...
DEL	löscht Zeilen	DEL<Bereich>
DELETE	löscht File auf Disk	DELETE<filename>
DIM	Dimensionieren eines Feldes	DIM<string var>OF<max.char> DIM<string array>(<array index>) OF <max.char> DIM<arrayname>(<array index>)
DIV	Division	<Divident>DIV<Divisor>
DO	führe die folgenden Befehle aus	DO<Befehl>
EDIT	listet Programme ohne Zeileneinrückung	EDIT(<Bereich>)
ELIF	ELSE IF	ELIF<Ausdruck>THEN
ELSE	alternative Befehle in IF-Konstrukten	ELSE
END	beendet Programm	END
ENDCASE	beendet CASE-Konstrukt	ENDCASE
ENDFOR	beendet FOR-Konstrukt	ENDFOR
ENDFUNC	beendet Funktion	ENDFUNC(<function name>)
ENDIF	beendet IF-Konstrukt	ENDIF
ENDPROC	beendet Prozedur	ENDPROC(<procname>)
ENDWHILE	beendet WHILE-Konstrukt	ENDWHILE
ENTER	ein Programmsegment von Diskette »mergen«	ENTER <filename>
EOD	End Of Data flag	EOD
EOF	End Of File flag	EOF(<filenum>)
ESC	stop key pressed flag	ESC TRAP ESC<type>
EXEC	führe eine Prozedur aus	(EXEC)<procname>((<Übergabeparameter>))
EXP	Exponent	EXP(<numerischer Ausdruck>)
FALSE	vordefinieren: Value = 0	False
FILE	definiert welches File benutzt wird	INPUT FILE<filenum> (,<recnum>):<varlist> PRINT FILE<filenum> (,<recnum>):<varlist> READ FILE<filenum> (,<recnum>):<varlist> WRITE FILE<filenum> (,<recnum>):<varlist> OPEN (FILE)<filenum>, <filename>(,<type>) CLOSE ((FILE)<filenum>)
FOR	beginnt FOR Schleife	FOR<var>=<start>TO<end> (STEP) (DO)
FUNC	beginnt Mehrzeilenfunktion	FUNK<name>((<params>)) (CLOSED)
GOTO	Sprungbefehl	GOTO<label name>
IF	Beginn IF-Konstrukt	IF<Bedingung>(THEN) IF<bedingung>THEN<Befehl>
IN	lokalisiert String1 in String2	<string1>IN<string2>
INPUT	Eingabe von Tastatur oder File	INPUT(<promt>:)<varLIST> INPUT FILE (s.o.)
INT	gibt die nächste ganze Zahl kleiner oder gleich	INT<numerischer Ausdruck>
KEY$	prüft Tastatur	KEY$
LABEL	vergibt Namen an Zeilennummer	<label name>

Tabelle 1. Befehlsübersicht von Comal

diese Befehle um drei weitere Befehlsblöcke erweitert worden. Diese Erweiterungen umfassen die Sprite, die HiRes- und die Turtle-Grafik sowie die Musik. Zwölf spezielle Befehle stehen für die Sprites zur Verfügung. Vergleicht man diese Erweiterung mit der Programmiersprache Logo, so stehen immer noch fünf zusätzliche Befehle zur Auswahl (Tabelle 1). Auch bei den HiRes- und Turtle-Grafik-Befehlen hat man gegenüber Logo fünf zusätzliche Befehle (Tabelle 2). Die Befehle sind sehr leicht zu handhaben und prägen sich schnell ein.

Nun zu der Schnelligkeit

Comal ist mit einem »Three pass interpreter — run time compiler« ausgerüstet. Der erste »pass« setzt gleich beim Programmieren ein. Alle Zeilen werden, sobald sie abgeschlossen worden sind, auf die Syntax überprüft. Dies ist ein extremer Vorteil gegenüber Basic, denn 57 verschiedene Fehlermeldungen stehen Comal zur Verfügung. So ist ein Fehler leicht zu lokalisieren und eine langwierige Fehlersuche bleibt dem Programmierer erspart. Laut Hersteller sind Fehler drei bis zehn mal schneller zu beseitigen als in Basic.

Im zweiten »pass« werden die Befehlsstrukturen auf Korrektheit überprüft und die Sprungadressen in Absolutadressen umgerechnet. Dieser Durchlauf wird durch RUN ausgelöst und nimmt normalerweise weniger als eine Sekunde in Anspruch.

Comal Schlüsselbefehle	Bedeutung	Befehlsform
LEN	definiert Länge eines Strings	LEN(<string Ausdr.>) Strings
LET	besetzt Variablen	:=
LIST	listet Programm	LIST(<Bereich>)(<filename>)
LOAD	lädt Programm von Disk	Load<filename>
LOG	Logarithmus von n	LOG(<numerischer Ausdruck>)
MOD	ergibt REST der Division	<Divident>MOD<Divisor>
NEW	löscht Arbeitsspeicher	NEW
NOT	logisches NEIN	NOT<Bedingung>
NULL	bewirkt nichts	NULL
OF	Teil von DIM- und CASE-Konstrukten	CASE<Ausdruck>OF DIM<stringvar>OF<max.char> DIM<stringarray>(array index> OF<max.char>
OPEN	öffnet ein File	siehe »FILE«
OR	logisches ODER	<Bedingung>OR<Bedingung>
ORD	ergibt Zahl die einem Charakter entspricht	ORD(<string-Ausdruck>)
OTHERWISE	entspricht bei CASE dem ELSE bei IF	OTHERWISE <Bedingung>
OUTPUT	wählt Ausgabegerät	SELECT (OUTPUT)<type>
PASS	übergibt einen String an den Kommandokanal der Disk	PASS<disc command>
PEEK	ergibt den Inhalt einer Speicheradresse	PEEK<Speicheradresse>
POKE	besetzt Speicheradresse	POKE<Speicheradresse>,<Bedingung>
PRINT	schreibt auf Drucker, Bildschirm oder File	PRINT(FILE<filenum>:)(<items>) PRINT(FILE<filenum>:)USING<FORMAT>:<vars>(RANDOM file use:(file<filnum>,<recnum> :))
PROC	startet Mehrzeilen Prozeduren	PROC<name>((<params>))(CLOSED)
RANDOM	wahlfreier Zugriff auf Disk-File	OPEN FILE<filenum>,<filename),RANDOM<recln>
READ	Datas oder File lesen	READ<var list> READ FILE<filenum>(,<rec num>):<varlist> OPEN(FILE)<filenum>,<file name>,READ
REF	Parametervariable wird in einer Prozedur verwendet	REF<var>
RENUM	Renumber Programm	RENUM (<targetstart>)(,<Schrittweite>)
REPEAT	Beginn des REPEAT-Konstruktes	REPEAT
RESTORE	Datenzeiger auf ersten Data-Wert zurücksetzen	RESTORE
RND	Zufallszahl	RND(<num>)
RUN	Programmstart	RUN
SAVE	Programm auf Disk speichern	SAVE<filename>
SELECT	Ausgabegerät wählen	SELECT(OUTPUT)<type>
SGN	-1 wenn negativ 0 wenn 0 +1 wenn positiv	SGN(numerischer Ausdruck)
SIN	Sinus (rad)	SIN(numerischer Ausdruck)
SIZE	gibt benötigten Speicherplatz an	SIZE
SQR	Quadratwurzel	SQR(<numerischer Ausdruck>)
STATUS$	Status vom Disk-Kanal	STATUS$
STEP	Angabe für die Schrittweite in Schleifen	STEP<numerischer Ausdruck>
STOP	Ende des Programms	STOP
SYS	startet Maschienenprogramme	SYS(<Speicheradresse>)
TAB	Tabulator	TAB(<Spaltennummer>)
TAN	Tangens (rad)	TAN(<numerischer Ausdruck>)
THEN	Teil des IF-Konstruktes	THEN
TO	Endangabe für Schleifen	<startnum>TO<endnum>
TRAP	Ausschalten des Stopkey	TRAPESC<type>
TRUE	Variable mit eins vorbesetzen	TRUE
UNIT	Gerät definieren	OPEN FILE<#>,<nm>,UNIT<def>(,<sec>)(,<type>)

Tabelle 1. Befehlsübersicht von Comal (Fortsetzung)

UNTIL	Ende der REPEAT-Schleife	UNTIL<Ausdruck>
USING	erlaubt formatierten Output	PRINT USING<format>:<varlist>
WHEN	Auswahl in CASE-Konstrukt	WHEN<list of values>
WHILE	Beginn WHILE-Konstrukt	WHILE<Ausdruck>(DO)(<Befehl>)
WRITE	Schreiben in ein File	WRITE FILE<filenum>(,<recnum>):<var list> OPEN (FILE)<filenum>,<file name>,WRITE
ZONE	Tabellenbereich	ZONE<tab interval> ZONE

Spezielle Spritebefehle

DATACOLLISION	Kollisionsabfrage mit Data	DATACOLLISION<sprite#>,<reset collisn flg?>
DEFINE	Sprite für späteren Gebrauch definieren	DEFINE<sprite definition num>,<64 byte def$>
HIDESPRITE	bestimmtes Sprite ausschalten	HIDESPRITE<sprite number>
IDENTIFY	einem Sprite eine Nummer zuordnen	IDENTIFY<sprite number>,<definition number>
(Bemerkung:Sprite 7 wird für TURTLE benutzt)		
PRIORITY	gibt Sprite Priorität über Data	PRIOITY<sprite num>,<data priority?>
SPRITEBACK	setzt zwei Multicolor-Sprite Farben	SPRITEBACK<color1>,<color2>
SPRITECOLLISION	testet Spritekollision	SPRITECOLLISION<sprite#>,<reset collisn flg?>
SPRITECOLOR	setzt Sprite-Farbe	SPRITECOLOR<sprite number>,<color number>
SPRITEPOS	positioniert Sprite auf x,y Position	SPRITEPOS<sprite#>,<x coord>,<y coord>
SPRITESIZE	setzt Sprite-Größe	SPRITESIZE<sprite#>,<x expand?>,<y expand?>

Tabelle 2. TURTLE GRAFIK von Comal im Vergleich zu Logo

TURTLE GRAPHICS CHART Items	C 64 LOGO WORD	C 64 LOGO EXAMPLE	C 64 COMAL WORD	C 64 COMAL EXAMPLE
TURTLE CONTROL:				
Move forward length	FORWARD	FORWARD 100	FORWARD	FORWARD 100
Move to a point	SETXY	SETXY 45	SETXY	SETXY 4,5
Move backward length	BACK	BACK THISFAR	BACK	BACK THISFAR
Home turtle	HOME	HOME	HOME	HOME
Turn turtle left	LEFT	LEFT 90	LEFT	LEFT 90
Turn turtle right	RIGHT	RIGHT MY-TURN	RIGHT	RIGHT MY-TURN
Turn to specific heading	SETHEADING	SETHEADING 0	SETHEADING	SETHEADING 0
Make turtle visible	SHOWTURTLE	SHOWTURTLE	SHOWTURTLE	SHOWTURTLE
Make turtle Invisible	HIDETURTLE	HIDETURTLE	HIDETURTLE	HIDETURTLE
Pen up off paper	PENUP	PENUP	PENUP	PENUP
Pen down on paper	PENDOWN	PENDOWN	PENDOWN	PENDOWN
Set pen color	PENCOLOR	PENCOLOR 4	PENCOLOR	PENCOLOR 4
Number of colors	16		16	
Set size of turtle	—	—	TURTLESIZE	TURTLESIZE 6
Plot a point	—	—	PLOT	PLOT 20,35
Print text in graphics	?	?	PLOTTEXT	PLOTTEXT 1,1,»TEXT«
SCREEN AND COLOR CONTROL:				
Clear graphics screen	CLEARSCREEN	CLEARSCREEN	CLEAR	CLEAR
Set to graphics mode	DRAW	DRAW	SETGRAPHIC	SETGRAPHIC
Set to text screen	NODRAW	NODRAW	SETTEXT	SETTEXT
Set background color	BACKGROUND	BACKGROUND 2	BACKGROUND	BACKGROUND 2
Set border color	—	—	BORDER	BORDER 4
Fill in an area	—	—	FILL	FILL 25,20
Full screen mode	FULLSCREEN	FULLSCREEN	FULLSCREEN	FULLSCREEN
Split screen mode	SPLITSCREEN	SPLITSCREEN	SPLITSCREEN	SPLITSCREEN
FUNCTION KEYS RESULTS:				
F1	TEXT SCREEN		TEXT SCREEN	
F3	SPLITSCREEN		SPLITSCREEN	
F5	FULLSCREEN		FULLSCREEN	
F7	GRAPHICS		GRAPHICS	

Der dritte »pass« ist der normale Programmlauf.

Im direkten Zeitvergleich ist Comal sechsmal schneller als Basic. Die Suche nach Strings soll nach Herstellerangabe sogar 79mal schneller als in Basic sein. Weiterhin soll man seine Programme drei- bis zehnmal schneller erstellen können.

Auch ist zu erwähnen, daß Comal strukturiertes Programmieren voll unterstützt.

Besonders muß darauf hingewiesen werden, daß die Diskette mit dieser Programmiersprache nicht kopiergeschützt ist, und man sich so beliebig viele Backups herstellen kann.

Diese Sprache gibt es fast kostenlos

Wie im Vorspann schon erwähnt, wird diese Sprache umsonst oder zum Selbstkostenpreis der Datenträger abgegeben. In der Anleitung selbst wird dazu aufgefordert, Comal zu kopieren und die Sprache dann an seine Freunde oder an seine Schule weiterzugeben. In Schleswig-Holstein ist Comal bereits an den Schulen eingeführt. Niedersachsens Schulen sollen bald folgen.

Comal ist über folgende Bezugsquelle zu erhalten:

Fa. INSTRUTEK
Christian Holmsgarde, 8700 Horsens, Dänemark

Ausdrücklich sei erwähnt, daß die 64'er Redaktion keine Kopien weitergeben kann. (rg)