

C64- Programme für C16 und VC20

In vielen Zeitschriften und Büchern findet man Programme für den C64, während es um Software für C16 und VC20 schon schlechter bestellt ist. Hier sind einige Tips, wie man C64-Programme für den eigenen Computer umschreiben kann.

Leider wird man nur sehr selten Programme für den C64 finden, die ohne Änderungen auf dem VC20 oder C16 laufen. Zwar sind Betriebssystem und Basic des VC20 und des C64 praktisch identisch, aber selbst hier stimmen die sonstigen Hardware-Eigenschaften überhaupt nicht überein. Noch schlimmer wird es beim C16, bei dem es verschiedene Speicher-Bänke gibt für den dem Anwender zur Verfügung stehenden Speicher sowie für Betriebssystem und Basic. Diese Speicherbänke werden durch ein recht kompliziertes Bank-Switching-Verfahren, ähnlich wie beim C128, verwaltet.

Immer dann, wenn zum Beispiel die grafischen oder musikalischen Fähigkeiten des C64 angesprochen werden, geschieht dies durch POKE-Befehle, mit denen Video- und Sound-Chip direkt gesteuert werden. Aufgrund der unterschiedlichen Hardware von C64, VC20 und C16, die zudem auch noch völlig unterschiedliche Adreßbereiche belegt, ist ein C64-Programm, das irgendwelche POKE-Befehle enthält, in der Regel weder auf dem VC20 noch auf dem C16 lauffähig. Tabelle 1 gibt einen ersten Überblick über wichtige Speicheradressen bei allen drei Computern.

Wir wollen uns im folgenden damit beschäftigen, inwiefern man dennoch C64-Programme auf den VC20 und den C16 übertragen kann. Dabei beschränken wir uns auf Basic-Programme, denn das Übertragen von Maschinencode-Programmen ist noch um einiges schwieriger und erfordert gute Kenntnisse und Erfahrung in Maschinensprache.

Generell kann man sagen, daß alle C64-Programme, die die fünf Befehle POKE, PEEK, SYS, USR und WAIT nicht enthalten, ohne jegliche Änderung sowohl auf dem VC20 als auch auf dem C16 lauffähig sind – sofern sie natürlich nicht die Kapazität des bei beiden Computern in der Grundversion doch arg beschränkten Speichers überschreiten.

Beim VC20 tritt noch das spezielle Problem des anderen Bildschirmformats auf. Während C64 und C16 auf dem Bildschirm 25 Zeilen zu je 40 Zeichen darstellen, sind es beim VC20 nur 23 Zeilen zu je 22 Zeichen. Dadurch gerät bei einfacher Übernahme der C64-Vorlage beim VC20 das ganze Bildschirm-Layout aus den Fugen. Eine Anpassung der PRINT-Anweisungen im Programm an die veränderten Verhältnisse ist also in fast allen Fällen erforderlich, dürfte aber in der Regel auch kein großes Problem sein. Bei der Anpassung von C64-Programmen an den VC20 wird man des öfteren zusätzliche Print-Befehle einfügen, da eine Bildschirmzeile beim VC20 nur etwa halb soviel Zeichen aufnehmen kann, wie das beim C64 der Fall ist.

Ernste Schwierigkeiten kann es möglicherweise bei der Darstellung von Tabellen geben. Eine sechsspaltige Zahlen-

tabelle zum Beispiel läßt sich beim C64 und C16 ganz gut darstellen, beim VC20 wird man bei derartigen Versuchen unangenehm an die arg begrenzte Zeilenbreite erinnert. In solchen Fällen kann man versuchen, weniger interessante Spalten der Tabelle einfach wegzulassen oder die Tabelle in zwei Teilen auszugeben. Wenn das nicht erwünscht ist, hilft nur noch der CMD-Befehl, um die Ausgabe der entsprechenden Tabelle auf den Drucker umzuleiten.

Was nicht geht

Leider gibt es viele Programme für den C64, die man nicht durch Änderung einiger POKE-Adressen und kleinerer Korrekturen am Bildschirmaufbau übernehmen kann. Dazu gehören beispielsweise die meisten Spiele, jedenfalls soweit es sich um Video-Spiele handelt. Der Grund hierfür sind die beim C64 vorhandenen acht Hardware-Sprites, die sich weder auf dem VC20 noch auf dem C16 reproduzieren lassen. Sprites sind kleine, bewegliche Grafikobjekte, die vom Video-Chip des C64 in den normalen Bildschirm eingeblendet werden können. Sprites sind nicht zu verwechseln mit den beim C16 vorhandenen Shapes. Shapes sind einfach rechteckige Ausschnitte aus der hochauflösenden Grafik des C16, die mittels spezieller Basic-Befehle vom Bildschirm gelesen und auf ihn geschrieben werden können. Shapes werden also rein softwaremäßig erzeugt, während die Sprites des C64 von der Hardware, nämlich vom Video-Chip, kontrolliert werden. Der Programmierer muß, nachdem er die Form eines Sprites einmal festgelegt hat, nur noch die Sprite-Position in spezielle dafür bestimmte Register des Video-Chips POKEn, um Bewegungsabläufe zu erzeugen. In der Regel werden Sprites beim C64 auch nicht im Grafik-Modus verwendet, sondern im normalen Textmodus, was bei den Shapes des C16 zum Beispiel auch nicht möglich ist.

Ebenfalls nicht verwenden kann man Musik-Programme für den C64. Im Gegensatz zu VC20 und C16 verfügt der C64 über einen echten Synthesizer-Baustein, mit dem sich eine ungeheure Fülle von Klangeffekten realisieren läßt. Dies alles führt dazu, daß sich leider ein Großteil aller C64-Programme, darunter die meisten Spiele, praktisch überhaupt nicht an den VC20/C16 anpassen lassen – es sei denn, man würde diese Programme völlig neu entwickeln. Nach diesen eher etwas ernüchternden Feststellungen wollen wir uns jetzt den Programmen zuwenden, die sich mit vertretbarem Aufwand umschreiben lassen. Das sind jene Basic-Programme, die keine Maschinensprache-Routinen enthalten und nur normale Textausgabe (allenfalls noch die Blockgrafik) verwenden, also keine »Tips & Tricks-Routinen«

Eine Grundregel

Beim Umschreiben eines C64-Programms auf den C16 oder VC20 sollte man eine ganz grundlegende Regel beachten: Niemals einen der bereits genannten »gefährlichen« Befehle POKE, SYS oder USR beim Abtippen des Listings so ohne weiteres übernehmen, sondern diese Befehle grundsätzlich nur übernehmen (meist in sinnvoll abgeänderter Form), wenn man sich über ihre Wirkung im klaren ist. Diese drei Befehle sind als einzige imstande, Speicherinhalte ungewollt zu verändern oder unkontrollierte Maschinensprache-Aufrufe zu bewirken. Hat man Bedenken, weil man sich nicht genau darüber im klaren ist, ob einer dieser Befehle schädlich ist oder nicht, dann empfiehlt es sich, den Befehl in eine PRINT-Anweisung umzuwandeln. Damit ist gemeint, den Befehl einfach in Anführungszeichen zu setzen und eine PRINT-Anweisung davor zu schreiben. Beispiel: Aus der Zeile

```
10 FOR I=49152 TO 49233 : READ X: POKE I,X : NEXT
  wird durch diese Methode die folgende Zeile
10 FOR I=49152 TO 49233 : READ X: PRINT "POKE"
; I; X :NEXT
```

Dadurch werden die POKE-Befehle innerhalb der FOR-NEXT-Schleife, die beim C64 durchaus sinnvoll sind, bei anderen Computern aber bestenfalls nutzlos, nicht ausgeführt, sondern nur am Bildschirm protokolliert. Bei einem Probelauf des Programms bekommt man so eher einen Eindruck, was da eigentlich vor sich geht, und kann möglicherweise entsprechende Schlüsse für die Anpassung einer solchen Routine an den eigenen Computer ziehen.

Natürlich kann ein solches Programm nicht richtig laufen, aber es kann – und das ist die positive Seite – auf gar keinen Fall durch unkontrollierte POKE- oder SYS-Befehle »abstürzen«. Weniger problematisch sind die Befehle PEEK und WAIT, da hierdurch nichts unkontrolliert verändert werden kann. Gleichwohl sind auch diese beiden Basic-Befehle nicht unproblematisch. In den seltensten Fällen können die beim C64 verwendeten Adressen für diese Befehle auch beim VC 20/C16 benutzt werden. Daher empfiehlt sich hier der Einbau einer zusätzlichen PRINT-Anweisung unmittelbar vor dem WAIT-Befehl (etwa in der Art »PRINT"Zeile 255, WAIT-BEFEHL !"«) oder unmittelbar nach PEEK. Zum Beispiel sollte man die Zeile

```
10 IF PEEK(X) <> 32 THEN 10
  ändern in
10 IF PEEK(X) <> 32 THEN PRINT "X=",X,"PEEK(X) ="
,PEEK(X):GOTO 10
```

Hierdurch hat man die Gewähr, beim Programmlauf ständig den Überblick zu behalten, was sich da gerade abspielt. Natürlich muß man sich nach dem Probelauf überlegen, wie man die POKE-Befehle (und andere) sinnvoll für seinen Computer abändert. Hierfür kann es naturgemäß kein Patentrezept geben, einige häufig in C64-Programmen vorkommende POKE-Adressen sollen aber im folgenden Teil etwas ausführlicher besprochen werden.

POKE-Befehle für Farbe und Bildschirm

Selbst die einfachsten Programme enthalten in der Regel Befehle, um die Rahmen- und Hintergrundfarbe des Bildschirms einzustellen. Während es beim C16 zu diesem Zweck den COLOR-Befehl gibt, arbeiten C64 und VC 20 hier mit POKE-Befehlen. Beim VC 20 werden beide Einstellungen (Rahmenfarbe und Hintergrundfarbe) gleichzeitig mit einem einzigen POKE-Befehl in Register 36879 durchgeführt (Tabelle 2). Der C64 verwendet dafür zwei getrennte Register, nämlich 53280 für die Rahmenfarbe und 53281 für die Hintergrundfarbe. Beim C16 schließlich geht es etwas komfortabler zu. Tabelle 3 zeigt die Anwendung des COLOR-Befehls bei diesem Computer. Bei allen drei Geräten sind jeweils 16 Farben möglich (Tabelle 4). Zum Beispiel erzeugen die Befehle »POKE 53280,7:POKE 53281,5« beim C64 einen gelben Bildschirmrahmen und einen grünen Hintergrund. Mit »POKE 36879,95« erreicht man dasselbe beim VC 20, mit »COLOR 4, 8 : COLOR 0,6« beim C16. Die Farbnummern sind beim C16 immer um Eins erhöht gegenüber den entsprechenden Nummern beim C64.

Der Bildschirmspeicher des C64 belegt die Adressen 1024 bis 2023 (1000 Zeichen), der zugehörige Farbspeicher geht von 55296 bis 56295. Mit »POKE 1024, 1: POKE 55296,2« erscheint zum Beispiel beim C64 ein rotes »A« in der linken oberen Bildschirmcke. Beim C16 ist der Videospeicher etwas höher angesiedelt. Er belegt die Adressen 3072 bis 4071, das Farb-RAM liegt von Adresse 2048 bis 3047. Der für den Anwender verfügbare Basic-Speicher-

bereich beginnt beim C64 bei Adresse 2048, beim C16 bei Adresse 4096. Tabelle 5 zeigt die Speicheraufteilung von C64, C16 und VC 20.

Beim VC 20 liegen die Verhältnisse noch etwas komplizierter: Die Anfangsadressen von Bildschirm- und Farbspeicher sind nämlich je nach Speicherausbau unterschiedlich. In der Grundversion und mit der 3-KByte Erweiterung beginnt das Video-RAM bei Adresse 7680 und geht bis zur Adresse 8185. Das Farb-RAM belegt dann den Bereich von 38400 bis 38905. In dieser Konfiguration liegt der Bildschirmspeicher oberhalb des für den Anwender verfügbaren RAM-Bereichs, der ab Adresse 4096 (Grundversion) beziehungsweise 1024 (3-KByte-Erweiterung) beginnt. Sobald jedoch beim VC 20 eine Speichererweiterung von mindestens 8 KByte eingesteckt ist, wandert das Video-RAM nach unten und beginnt dann bei Adresse 4096. Dies geschieht, um für Basic-Programme einen zusammenhängenden Speicherbereich von der Adresse 4608 an aufwärts zu schaffen. Eine eventuell vorhandene zusätzliche 3-KByte-Erweiterung kann in diesem Fall nicht für Basic-Programme genutzt werden. Natürlich verändert auch das Farb-RAM noch seine Lage und startet jetzt bei Adresse 37888.

Die musikalische Seite

Einige beim C64 ebenfalls häufig vorkommende POKE-Adressen wurden bisher noch gar nicht erwähnt. Gemeint sind die zur Programmierung von Musik und Geräuscheffekten benutzten Register. Leider ist die Art der Tonerzeugung beim C64 auf der einen und VC 20/C16 auf der anderen Seite völlig unterschiedlich, so daß sich keine äquivalenten Befehle angeben lassen. Wie bereits zu Anfang erwähnt, verfügt der C64 über einen vollwertigen, dreistimmigen Synthesizer-Baustein, während der C16 und VC 20 sich mit einfachen Tongeneratoren zufrieden geben müssen. Tabelle 6 enthält zur Referenz eine Übersicht über die beim C64 zur Tonerzeugung benutzten Register. Diese Tabelle dient allerdings wirklich nur zur Orientierung, denn eine Simulation des C64-Synthesizers ist weder mit dem VC 20 noch mit dem C16 möglich. Es empfiehlt sich daher, bei der Programmanpassung zunächst einmal alle derartigen POKE-Befehle fortzulassen und später eigene Sound-Routinen einzufügen. C64-Besitzer haben es umgekehrt natürlich etwas einfacher. Mit etwas Geschick und einem guten Handbuch zur Musik-Programmierung können Sie dem C64 natürlich auch die VC 20/C16-Töne entlocken. Für alle C64- und C16-Besitzer, die vielleicht doch einmal in die Verlegenheit kommen, das eine oder andere VC 20-Programm auf ihre Geräte umzuschreiben, sind in Tabelle 7 die Notenwerte und die Tongenerator-Adressen des VC 20 angegeben.

Joysticks und Paddles

Der C64 verfügt ebenso wie der C16 über zwei Joystick-Ports, während dem VC 20 nur einer vergönnt ist. Die Abfrage ist beim C16 wieder am komfortabelsten, nämlich per Basic-Funktion. »PRINT JOY(1)« ergibt den Richtungswert von Joystick 1, entsprechend wird mit JOY(2) die Richtung des zweiten Joysticks abgefragt. Beim C64 entsprechen diesen Abfragen PEEKs in die Speicherstellen 56320 (Joystick 1) und 56321 (Joystick 2). Ziemlich merkwürdig gelöst ist die Joystick-Abfrage beim VC 20, wo ein recht kompliziertes Abfrageprogramm erforderlich ist. Der Feuerknopf und die Schalter 0, 1 und 2 des Joysticks werden nämlich beim VC 20 über die VIA 1 gelesen, während der Zustand von

Wo sich das Umschreiben lohnt

An dieser Stelle muß noch einmal deutlich darauf hingewiesen werden, daß die Zahl der C 64-Programme, die sich mit vertretbarem Aufwand zum VC 20 oder C 16 übertragen lassen, doch verhältnismäßig gering ist. VC20-Besitzer sind hier etwas besser dran. Sie finden in diesem Sonderheft ein Programm, das Maschinenprogramme für den C 64 in entsprechende Routinen für den VC20 übersetzt. Es funktioniert in etwa 70-80% aller Fälle. Im allgemeinen ist jetzt speziell bei Spielprogrammen in der Regel äußerste Vorsicht geboten, da hier aus Geschwindigkeitsgründen zumeist mit verschiedenen Routinen in Maschinensprache gearbeitet wird. Von Sprites und hochauflösender Grafik einmal ganz zu schweigen. Dagegen gibt es viele Anwendungsprogramme, die den C 64 nicht als hochspezialisierte Spielmaschine (die er ohne Zweifel ist) ausnutzen, sondern ganz einfach Problemlösungen in Basic anbieten. Dabei kann es sich beispielsweise um eine Dateiverwaltung, einen Vokabeltrainer oder ganz einfach ein Programm zum Ausdrucken eines Jahreskalenders handeln. Für fast alle derartigen Programme sollte es möglich sein, eine Anpassung mit Hilfe der hier abgedruckten Tabellen und Hinweise vorzunehmen. (ev)

Tabelle 5. Speicherbelegung bei C64, C16 und VC 20

	C 64	VC 20		C16/C116/Plus 4
		Grundversion	ab 8 KByte	
Video-RAM	1024	7680	4096	3072
freier RAM-Bereich	2048	4096	4608	4096
Video-Chip (VIC/TED)	53248	36864	36864	65280
Synthesizer-Chip	54272	—	—	—
Farb-RAM	55296	38400	37888	2048
Basic-ROM	40960	49152	49152	32768 (in ROM-Bank)

Tabelle 1. Wichtige Speicheradressen bei C64, C16 und VC 20 zur ersten Orientierung

COLOR-Farbzonen-Nr., Farb-Nr., [, Helligkeitswert]

Mit der COLOR-Anweisung wird eine der fünf Farbzonen bestimmt:

Farbzonen-Nr.	Bezeichnung
0	Bildschirm-Hintergrund
1	Vordergrund (Buchstaben, Zeichen)
2	Mehrfarben 1
3	Mehrfarben 2
4	Bildschirmrand

Die zweite Zahl in der COLOR-Anweisung selektiert die Hintergrundfarbe des gewählten Bildschirmbereichs. Diese Zahl ist mit den Farbtasten der Tastatur identisch.

Farb-Nr.	Farbe	Farb-Nr.	Farbe
1	Schwarz	9	Orange
2	Weiß	10	Braun
3	Rot	11	Gelb/Grün
4	Cyan	12	Rosa
5	Purpur	13	Blau/Grün
6	Grün	14	Hellblau
7	Blau	15	Dunkelblau
8	Gelb	16	Hellgrün

Jede Farbe ist auch noch in ihrer Helligkeit (Luminanz) veränderbar. Im Anhang an die Farb-Nr. kann diese Luminanz von Null (0= dunkel) bis sieben (7= hell) variieren. Der Standardwert für die Helligkeit ist 7. Die acht Helligkeitsstufen gelten für alle Farben, mit Ausnahme von Schwarz.

Tabelle 3. Der COLOR-Befehl des C16 kann bis zu drei Parameter enthalten (vergleiche Tabelle 4)

VC 20 und C64		nur C64	
0 schwarz	4 violett	8 orange	12 grau 2
1 weiß	5 grün	9 braun	13 hellgrün
2 rot	6 blau	10 hellrot	14 hellblau
3 türkis	7 gelb	11 grau 1	15 grau 3

Tabelle 4. Codierung einzelner Farben bei C64 und VC 20

Adresse			Inhalt
Stimme 1	Stimme 2	Stimme 3	
54272	54279	54286	Tonfrequenz (Low-Byte)
54273	54280	54287	Tonfrequenz (High-Byte)
54274	54281	54288	Tastaturverhältnis Rechteckgenerator (Low)
54275	54282	54289	Tastaturverhältnis Rechteckgenerator (High)
54276	54283	54290	Wellenform
54277	54284	54291	Anschlag/Abschwellen
54278	54285	54292	Halten/Ausklingen
54296			Lautstärke
Mögliche Wellenformen: Dreieck (17), Sägezahn (33), Rechteck (65), Rauschen (129).			

Tabelle 6. Die wichtigsten Register des C64-Soundchips

Rahmen ►	BLK	WHT	RED	CYAN	PUR	GRN	BLU	YEL
Hintergrund ▼	SCHW	WEISS	ROT	TÜRKIS	VIOLETT	GRÜN	BLAU	GELB
SCHWARZ	8	9	10	11	12	13	14	15
WEISS	24	25	26	27	28	29	30	31
ROT	40	41	42	43	44	45	46	47
TÜRKIS	56	57	58	59	60	61	62	63
VIOLETT	72	73	74	75	76	77	78	79
GRÜN	88	89	90	91	92	93	94	95
BLAU	104	105	106	107	108	109	110	111
GELB	120	121	122	123	124	125	126	127
ORANGE	136	137	138	139	140	141	142	143
HELLORANGE	152	153	154	155	156	157	158	159
ROSA	168	169	170	171	172	173	174	175
HELLTÜRKIS	184	185	186	187	188	189	190	191
HELLVIOLETT	200	201	202	203	204	205	206	207
HELLGRÜN	216	217	218	219	220	221	222	223
HELLBLAU	232	233	234	235	236	237	238	239
HELLGELB	248	249	250	251	252	253	254	255

Tabelle 2. Farbkombinationen für Bildschirmrahmen und Hintergrund beim VC 20. Die einzelnen Zahlenwerte errechnen sich nach der Formel »POKE-Wert = Hintergrund x 16 + Rahmen + 8« (vergleiche Tabelle 4).

NOTE	WERT	NOTE	WERT
C	135	G	215
C #	143	A ^b	217
D	147	A	219
E ^b	151	B	221
E	159	H	223
F	163	C	225
F #	167	C #	227
G	175	D	228
A ^b	179	E ^b	229
A	183	E	231
B	187	F	232
H	191	F #	233
C	195	G	235
C #	199	A ^b	236
D	201	A	237
E ^b	203	B	238
E	207	H	239
F	209	C	240
F #	212	C #	241

Stimmlagen-Befehle	X =	Funktion
POKE 36878,X	0 bis 15	setzt die Lautstärke
POKE 36874,X	128 bis 255	spielt Note
POKE 36875,X	128 bis 255	spielt Note
POKE 36876,X	128 bis 255	spielt Note
POKE 36877,X	128 bis 255	Geräuscheffekte

Tabelle 7. Notenwerte und Tongenerator-Adressen beim VC 20. Diese Tabelle dient zur Orientierung für C 16-Besitzer, die das eine oder andere VC 20-Programm für ihren Computer umsetzen möchten.