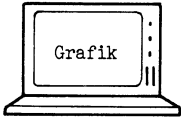


Grafikprogrammierung auf dem C 64



Einfach zeigen wir Ihnen, was sich mit dem Videocontroller des C 64 alles anstellen läßt. Gestützt durch viele Basic-Beispiele lernen Sie mit Sprites, HiRes-Bildschirmen und allem was mit Grafik zu tun hat, umzugehen.

Zunächst einige allgemeine Informationen. Der Bildschirm des C 64 verfügt über 1000 Positionen. Normalerweise beginnt der Bildschirm ab Adresse 1024 (\$0400) und endet bei Adresse 2023. Jede dieser Adressen kann 8 Bit speichern. Das entspricht einer beliebigen ganzen Zahl zwischen 0 und 255. Der Bildschirmspeicher entspricht einer Gruppe von 1000 Adressen, der Farbspeicher oder Farb-RAM genannt wird. Dieses Farb-RAM belegt den Speicher von 55296 (\$D800) bis 56295 (\$DBE7). Jede Farb-RAM-Adresse speichert 4 Bit und kann daher eine beliebige ganze Zahl zwischen 0 und 15 aufnehmen.

Die verschiedenen Grafikmodi werden über 47 Steuerregister im Videocontroller-Baustein ausgewählt. Viele Grafikfunktionen lassen sich steuern, indem der richtige Wert über die POKE-Anweisung in eines der Register geschrieben wird, das im Adressbereich von 53248 (\$D000) bis 53294 (\$D02E) liegt.

Wahl der Video-Bank

Der Videocontroller-Baustein kann immer nur auf einen Speicherbereich von 16 KByte zugreifen. Da im C 64 64 KByte RAM untergebracht sind, soll der Videocontroller natürlich auch

den ganzen Speicher »sehen« können. Dies ist möglich. Es gibt vier verschiedene Banks (oder Abschnitte), die für jeweils 16 KByte gelten. Nun muß lediglich noch geregelt werden, auf welche dieser Abschnitte der Videocontroller-Baustein zugreift. Auf diese Weise kann der Baustein die gesamte Speicherkapazität von 64 KByte ausnutzen. Die Bankanwahl-Bits, die Ihnen einen Zugriff auf die verschiedenen Speicherabschnitte ermöglichen, befinden sich im Complex-Interface-Adapter #2 (CIA #2) 6526. Über die Basic-Anweisung POKE und PEEK wird eine Bank durch Steuerung der Bits 0 und 1 von Port A des CIA #2 (oder hex 56576 \$DD00) gewählt. Zur Änderung der Speicherabschnitte müssen diese zwei Bit auf Ausgabe gesetzt sein. Dies wird anhand nachstehender Befehle deutlich: POKE 56578, PEEK(56578) OR 3: REM BITS 0 UND 1 SETZEN POKE 56576, (PEEK(56576) AND 252) OR A: REM VIDEO-BANK WECHSELN

»A« muß einen der folgenden Werte haben (Tabelle 1). Dieses Konzept der 16-KByte-Abschnitte spielt bei allen Anwendungen des VIC eine Rolle. Sie sollten stets wissen, auf welche Bank der Videocontroller zugreift, denn dadurch wird festgelegt, woher die Zeichenda-

Wert von A	Bits	Bank	Start-Platz	Bereich des Videocontroller-Bausteins
0	00	3	49152	(\$C000-\$FFFF)
1	01	2	32768	(\$8000-\$BFFF)
2	10	1	16384	(\$4000-\$7FFF)
3	11	0	0	(\$0000-\$3FFF) (Standardwert)

Tabelle 1. Alle möglichen 16-KByte-Bänke des Videocontrollers

tenmuster kommen, wo sich der Bildschirm und der Speicherbereich für die Sprites befinden. Nach dem Einschalten des C 64 wird automatisch Bank 0 (\$0000-\$3FFF) aktiviert.

Anmerkung: Der Zeichensatz des Commodore 64 ist in den Banks 1 und 3 für den Videocontroller-Baustein nicht verfügbar.

Bildschirmspeicher

Durch POKE in das Kontrollregister 53272 (\$D018 HEX) kann die Adresse des Bildschirmspeichers geändert werden. Dieses Register wird jedoch auch zur Steuerung des jeweils benutzten Zeichensatzes verwendet. Achten Sie daher besonders darauf, diesen Teil des Steuerregisters nicht zu stören. Die oberen 4 Bit steuern den Bildschirmspeicher. Um den Bildschirm in einen anderen Bereich zu legen, ist folgende Anweisung erforderlich:

```
POKE 53272, (PEEK(53272) AND 15) OR A
```

Hierbei hat A einen der folgenden Werte (Tabelle 2).

Farbspeicher

Der Farbspeicher kann nicht verschoben werden. Er befindet sich stets im Adressbereich von 55296 (\$D800) bis 56295 (\$DBE7). Bild-

schirmspeicher und Farbspeicher werden in den verschiedenen Grafikmodi unterschiedlich benutzt. Ein in einem Modus erstelltes Bild sieht in einem anderen Grafikmodus häufig anders aus.

Zeichenspeicher

Für die Programmierung von Grafiken ist es wesentlich, von wo genau der Videocontroller die Zeicheninformation bekommt. Normalerweise erhält der Baustein die Konturen der anzuzeigenden Zeichen vom Character-Generator-ROM. In diesem Baustein sind die Muster gespeichert, die die verschiedenen Buchstaben, Zeichen, Interpunktionsymbole und alle anderen Zeichen der Tastatur bilden.

Eines der Merkmale des Commodore 64 ist seine Fähigkeit, im RAM-Speicher befindliche Muster zu benutzen. Diese RAM-Muster werden von Ihnen erstellt, so daß ein nahezu unbegrenzter Satz an Symbolen für Spiele und so weiter zur Verfügung steht.

Ein normaler Zeichensatz enthält 256 Zeichen, bei dem jedes Zeichen durch 8 Byte bestimmt wird. Da jedes Zeichen also 8 Byte beansprucht, benötigt der komplette Zeichensatz $256 \times 8 = 2$

A	Bits	Lage*	
		dezimal	hexadezimal
0	0000XXXX	0	\$0000
16	0001XXXX	1024	\$0400 (Standards)
32	0010XXXX	2048	\$0800
48	0011XXXX	3072	\$0C00
64	0100XXXX	4096	\$1000
80	0101XXXX	5120	\$1400
96	0110XXXX	6144	\$1800
112	0111XXXX	7168	\$1C00
128	1000XXXX	8192	\$2000
144	1001XXXX	9216	\$2400
160	1010XXXX	10240	\$2800
176	1011XXXX	11264	\$2C00
192	1100XXXX	12288	\$3000
208	1101XXXX	13312	\$3400
224	1110XXXX	14336	\$3800
240	1111XXXX	15360	\$3C00

* Bitte denken Sie daran, daß die Startadresse der jeweiligen Bank des Videocontroller-Bausteins addiert werden muß.

Tabelle 2. Die oberen vier Bit des Kontrollregisters bestimmen die Lage des Bildschirmspeichers

KByte. Da der Videocontroller-Baustein gleichzeitig auf 16 KByte zugreift, gibt es acht verschiedene Speicherplatzmöglichkeiten für einen vollständigen Zeichensatz. Sie brauchen natürlich nicht immer einen ganzen Zeichensatz zu verwenden. Er muß jedoch stets an einem der acht möglichen Startplätze beginnen. Die Lage des Zeichenspeichers wird durch 3 Bit vom Videocontroller kontrolliert (53272 (\$D018 HEX)). Die Bits 3, 2 und 1 steuern die Lage des Zeichensatzes, Bit 0 wird überlesen. Um die Lage des Zeichensatzes zu ändern, benutzen Sie folgende Basic-Anweisung:

```
POKE 53272, (PEEK(53272)
AND240) OR A
```

Hierbei hat A einen der folgenden Werte (Tabelle 3).

Greift der VIC auf die Zeichendaten zu, so wird das ROM eingeschaltet. Ansonsten wird dieser Bereich von den Ein-/Ausgaberegistern beansprucht, und das Zeichen-ROM kann nur vom VIC erreicht werden.

Es kann jedoch passieren, daß Sie das Zeichen-ROM benötigen, und zwar dann, wenn Sie programmierbare Zeichen benutzen wollen und eine Kopie eines Teils vom Zeichen-ROM für die Zeichendefinition brauchen. In diesem Fall müssen Sie das Ein-/Ausgaberegister aus- und das Zeichen-ROM einschalten. Dann können

Unterbrechungen erlaubt. Für Unterbrechungen werden nämlich die Ein-/Ausgaberegister benötigt. Wenn Sie dies vergessen und eine Unterbrechung vornehmen, passiert Unvorhersehbares. Die Tasteneingabe darf während des Kopierens nicht gelesen werden. Um die Tastatur und weitere normale Unterbrechungen abzuschalten, die mit dem C 64 möglich sind, benutzen Sie folgende POKE-Anweisung:

```
POKE 56334, PEEK(56334)
AND254 (Interrupt AUS)
```

Wenn Sie den Zugriff auf das Zeichen-ROM beendet haben, wird die Tastatur durch folgende POKE-Anweisung eingeschaltet:

```
POKE 56334, PEEK(56334) OR 1
(Interrupt EIN)
```

Durch folgende POKE-Anweisung wird die Ein-/Ausgabe ausgeschaltet und das Zeichen-ROM eingeschaltet:

```
POKE 1, PEEK(1) AND251
```

Das Zeichen-ROM befindet sich nun im Bereich von 53248 bis 57343 (\$D000 bis \$DFFF). Um die Ein-/Ausgabe für den normalen Betrieb zu aktivieren, ist folgende POKE-Anweisung erforderlich:

```
POKE 1, PEEK(1) OR 4
```

Standardzeichenmodus

Beim Einschalten des C 64 befindet sich dieser im Standardzeichenmodus, in dem Sie normalerweise programmieren.

spezielle Grafikzeichen, so sind lediglich die neuen Zeichenmuster im RAM zu definieren und der VIC ist anzuweisen, die Zeicheninformationen aus dem RAM und nicht aus dem Zeichen-ROM zu nehmen.

Um Zeichen auf dem Bildschirm in Farbe anzuzeigen, greift der Videocontroller-Baustein auf den Bildschirmspeicher zu, um den Zeichen-Code für diesen Bildschirmplatz zu bestimmen. Gleichzeitig greift er auf den Farbspeicher zu, um die Farbe für die Zeichenanzeige festzulegen. Der Zeichen-Code wird vom Videocontroller in die Startadresse des 8-KByte-Satzes mit Ihrem Zeichenmuster umgesetzt.

Die Umsetzung ist nicht zu kompliziert, zur Erstellung der gewünschten Adresse werden jedoch verschiedene Punkte kombiniert. Zunächst ist der von Ihnen bei der POKE-Anweisung für den Bildschirmspeicher benutzte Zeichencode mit 8 zu multiplizieren. Danach wird

Zeichen und jedes Bit für einen Punkt. Der Zeichenspeicher im ROM beginnt ab 53248 (bei ausgeschalteter Ein-/Ausgabe). Die ersten 8 Byte ab 53248 (\$D000) bis 53255 (\$D007) enthalten das Muster für das Zeichen @, dessen Zeichencodewert im Bildschirmspeicher 0 ist. Die nächsten 8 Byte ab 53256 (\$D008) bis 53263 (\$D00F) enthalten die Information zur Bildung des Buchstabens A (Bild 1).

Jeder vollständige Zeichensatz beansprucht eine Speicherkapazität von 2 KByte (2048 Bit). Insgesamt sind 256 Zeichen vorhanden. Da es insgesamt zwei Zeichensätze gibt, enthält der ROM-Zeichenspeicher insgesamt 4 KByte Speicherplatz.

Programmierbare Zeichen

Da die Zeichen im ROM gespeichert sind, sieht es so aus, als ob sie für frei programmierbare Zeichen nicht geändert werden könnten. Der Speicherplatz, der dem VIC mitteilt, wo die Zeichen zu finden sind, ist jedoch ein programmierbares Register. Dieses kann so geändert werden, daß es auf viele Speicherbereiche zeigt, indem der Zeichenspeicherzeiger so geändert wird, daß er auf das RAM zeigt. Alle Buchstaben, Zahlen oder Standardgrafikzeichen vom C 64 müssen zuerst in den RAM-Speicher kopiert werden, damit Sie ihn in Ihrem Programm benutzen können. Achtung: Achten Sie darauf, daß Ihr Zeichensatz nicht vom Basic-Programm, das auch das RAM benutzt, überschrieben wird.

Zwei Adressen im C 64 dürfen nicht als Beginn des Zeichensatzes gewählt werden: Adresse 0 und Adresse 2048. Der erste darf nicht benutzt werden, weil das System auf Seite 0 (0-Page) wichtige Daten speichert. Adresse 2048 ist der Beginn Ihres Basic-Programms!

Für Ihren Zeichensatz stehen jedoch noch sechs weitere Anfangspositionen zur Verfügung. Am besten wählen Sie hierzu den Bereich ab 12288 (\$3000). Dies erfolgt, indem zu den unteren 4 Bit in Speicherzelle 53272 der Wert 12 addiert wird. Probieren

Belegung	binär	PEEK
**	00011000	24
****	00111100	60
** **	01100110	102
*****	01111110	126
** **	01100110	102
** **	01100110	102
** **	01100110	102
** **	00000000	0

Bild 1. Der Buchstabe »A« steht im Zeichen-ROM in den Speicherzellen von 53256 bis 53263

der Anfang vom Zeichenspeicher addiert. Nun werden die Bankwahl-Bits berücksichtigt. Hierzu wird die Basisadresse addiert. Anhand der folgenden einfachen Gleichung können Sie sehen, wie dies gemeint ist: Zeichenadresse = Bildschirmcode x 8 + (Zeichensatz x 2048) + (Bank x 16348)

Zeichendefinitionen

Jedes Zeichen wird aus einer Matrix von 8 x 8 Punkten gebildet. Hierbei können die einzelnen Punkte entweder ein- oder ausgeschaltet sein. Beim Commodore 64 sind die Zeichenbits im Zeichengenerator-ROM abgelegt. Jedes Zeichen ist hierbei als Satz von 8 Byte gespeichert. Jedes Byte steht für das Punktmuster einer Reihe im

Wert	Bits	Lage des Zwischenspeichers*	
		dezimal	hexadezimal
0	XXXX000X	0	\$0000-\$07FF
2	XXXX001X	2048	\$0800-\$0FFF
4	XXXX010X	4096	\$1000-\$17FF
6	XXXX011X	6144	\$1800-\$1FFF
8	XXXX100X	8192	\$2000-\$27FF
10	XXXX101X	10240	\$2800-\$2FFF
12	XXXX110X	12288	\$3000-\$37FF
14	XXXX111X	14336	\$3800-\$3FFF

*Bitte denken Sie daran, die Startadresse der Bank zu addieren.

Tabelle 3. Die Bits 1, 2 und 3 des Kontrollregisters bestimmen die Lage des Zeichensatzspeichers

Sie kopieren. Danach muß das Ein-/Ausgaberegister erneut eingeschaltet werden. Während des Kopierens (bei ausgeschalteter Ein-/Ausgabe) sind keine Un-

Zeichen können aus dem ROM oder dem RAM gelesen werden. Normalerweise wird jedoch auf die Zeichen im ROM zugegriffen. Benötigen Sie für ein Programm

Bit-Paar	Farbregister	Speicherplatz
00	Hintergrundfarbe #0 (Bildschirmfarbe)	53281 (\$D021)
01	Hintergrundfarbe #1	53282 (\$D022)
10	Hintergrundfarbe #2	53283 (\$D023)
11	Durch die unteren 3 Bits im Farbspeicher bestimmte Farbe	Farbspeicher

Tabelle 4. Mögliche Farbkombinationen für Multicolor-Zeichen

bens A und das entsprechende Bit-Muster (Bild 2).

Im normalen oder HiRes-Modus wird die Bildschirmfarbe bei jedem 0-Bit und die Zeichenfarbe stets da angezeigt, wo das Bit 1 ist. Beim Mehrfarbenmodus werden die Bits paarweise benutzt (Bild 3).

Im Bildbereich werden die durch AA gekennzeichneten Stellen in der Hintergrundfarbe #1, die durch BB gekennzeichneten Stellen in der Hintergrundfarbe #2 und die durch CC gekennzeichneten Stellen in der Zeichenfarbe dargestellt. Dies wird entsprechend nachstehender Tabelle 4 durch die Bit-Paare bestimmt.

Zeichencode			Hintergrundfarbregister	
Bereich	BIT 7	BIT 6	Nummer	Adresse
0 - 63	0	0	0	53281 (\$D021)
64 - 127	0	1	1	53282 (\$D022)
128 - 191	1	0	2	53283 (\$D023)
192 - 255	1	1	3	53284 (\$D024)

Tabelle 5. Im Hintergrundfarbmodus können die ersten 64 Zeichen mit unterschiedlichen Hintergrundfarben dargestellt werden

Hintergrundfarbmodus

In diesem Modus können Sie für jedes einzelne Zeichen sowohl die Hintergrund- als auch die Vordergrundfarbe steuern. So ist es zum Beispiel möglich, auf einem weißen Bildschirm ein blaues Zeichen mit gelbem Hintergrund anzuzeigen.

Für den erweiterten Hintergrundfarbmodus stehen vier Register zur Verfügung. Für jedes Register kann eine der 16 Farben gewählt werden.

In diesem Modus wird über den Farbspeicher die Vordergrundfarbe festgelegt. Die Anwendung ist die gleiche wie beim Standard-Zeichenmodus.

Bei erweitertem Modus ist die Anzahl der verschiedenen anzeigbaren Zeichen jedoch eingeschränkt. Ist der erweiterte Farbmodus eingeschaltet, können nur die ersten 64 Zeichen des Zeichen-ROM (oder die ersten 64 in Ihrem programmierba-

ren Zeichensatz) benutzt werden. Zwei Bit des Zeichen-Codes werden nämlich für die Wahl der Hintergrundfarbe benutzt.

Der Zeichen-Code (die auf dem Bildschirm gePOKEte Zahl) vom Buchstaben A ist eine 1. Im erweiterten Farbmodus erscheint nach dem POKEN einer 1 ein A. Normalerweise muß nach dem POKEN von 65 das Zeichen mit dem Zeichen-Code (CHR\$) 129, also ein invertiertes A, erscheinen. Dies passiert nicht im erweiterten Farbmodus. Es erscheint genau das gleiche A wie vorher, jedoch eine andere Hintergrundfarbe. Entnehmen Sie die Codes der Tabelle 5.

Zum Einschalten des erweiterten Farbmodus wird Bit 6 des VIC-II-Registers mit der Adresse 53265 (\$D011) auf 1 gesetzt. Dies geschieht durch folgende POKE-Anweisung:

POKE 53265, PEEK(53265) OR 64

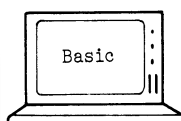
Zum Ausschalten des erweiterten Farbmodus wird Bit 6 des VIC-II-Registers mit der Adresse 53265 (\$D011) auf 0 gesetzt. Hierzu dient folgende Anweisung:

POKE 53265, PEEK(53265) AND 191

Damit sind wir am Schluß des ersten Teils angelangt. Beim nächsten Mal werden wir uns mit dem Bit-Map-Modus beschäftigen. Sie werden dann lernen, wie man HiRes- und Multicolor-Bilder auf den Bildschirm zaubert. (ah)

Gekürzter Auszug aus dem Buch »Alles über den C 64« aus der Commodore Sachbuchreihe, Markt & Technik Verlag, Hans-Pinsel-Str. 2, 8013 Haar, Bestnr.: ISBN 3-89090-379-7, Preis 59 Mark

Variablen — Schwerarbeiter in der Datenverarbeitung



Sind Sie Computerneuling? Dann sind Sie in dieser Rubrik genau richtig. An dieser Stelle beginnt ein Kurs, der die Funktionen und Eigenschaften des Commodore-Basic Version 2.0, das im C 64 eingebaut ist, erklärt. Im ersten Teil erfahren Sie alles Wissenswerte zu den bisher vielleicht noch unbekannten Variablen.

Legen Sie zunächst das Handbuch des C 64 neben sich, denn hier soll das dort Geschriebene weiter vertieft und ausführlich erklärt werden. Lesen Sie zunächst die Kapitel über die Variablen durch (S. 35,98 bis 100). Wir werden im Laufe des Kurses immer wieder auf das Handbuch verweisen und darauf aufbauen.

Vielleicht haben Sie sich schon des öfteren gefragt, wie es möglich ist, daß der Computer Umengen von Daten in relativ kurzer Zeit verarbeiten kann. Des Rätsels Lösung liegt in den sogenannten Variablen verborgen, die im folgenden ihren vielleicht noch etwas rätselhaften Charakter verlieren werden.

Vielleicht erinnert sich der eine oder andere noch entfernt an den Mathematikunterricht in der Schule und möchte jetzt beim Gedanken an diese ungeliebten Wesen aus der Algebra am liebsten nicht mehr weiter lesen. Doch keine Angst, schließlich soll hier nicht höhere Mathematik betrieben werden, vielmehr erfolgt eine leicht verständliche Erklärung der für die Programmierung so wichtigen Variablen.

Sicher haben Sie schon des öfteren mit den im Handbuch des C 64 erklärten Befehlen gearbeitet. Als Bei-

spiel soll der PRINT-Befehl herangezogen werden. Geben Sie doch einmal die Anweisung

10 PRINT "HALLO"
ein. Wenn dieses kleine Programm mit RUN gestartet wird, erhalten Sie auf dem Bildschirm das Wort »HALLO« angezeigt. Leider kann diese Ausgabe jetzt nicht mehr verändert werden, ohne das Programm zu ändern. Man spricht deshalb von einer Textkonstanten (ständig in derselben Weise vorhanden). Soll nun beispielsweise eine ganz einfache Berechnung angestellt werden, bei der ständig neue Werte erforderlich sind, reicht diese Möglichkeit der Datenverarbeitung nicht mehr aus.

Erste Gehversuche

Für diese komplexeren Aufgaben werden Variablen benötigt. Diese bezeichnen ein Datenfeld (Speicherbereich für bestimmte Daten), dem Werte oder Texte zugewiesen werden können, die sich ständig durch Ihre Eingabe oder den Programmablauf ändern. Um dieses Phänomen näher zu betrachten, sollte die Funktion der beiden Basic-Befehle INPUT und LET aus dem Handbuch entnommen werden. Man kann also einer Variablen einen beliebigen Wert zuweisen und mit dieser Variablen