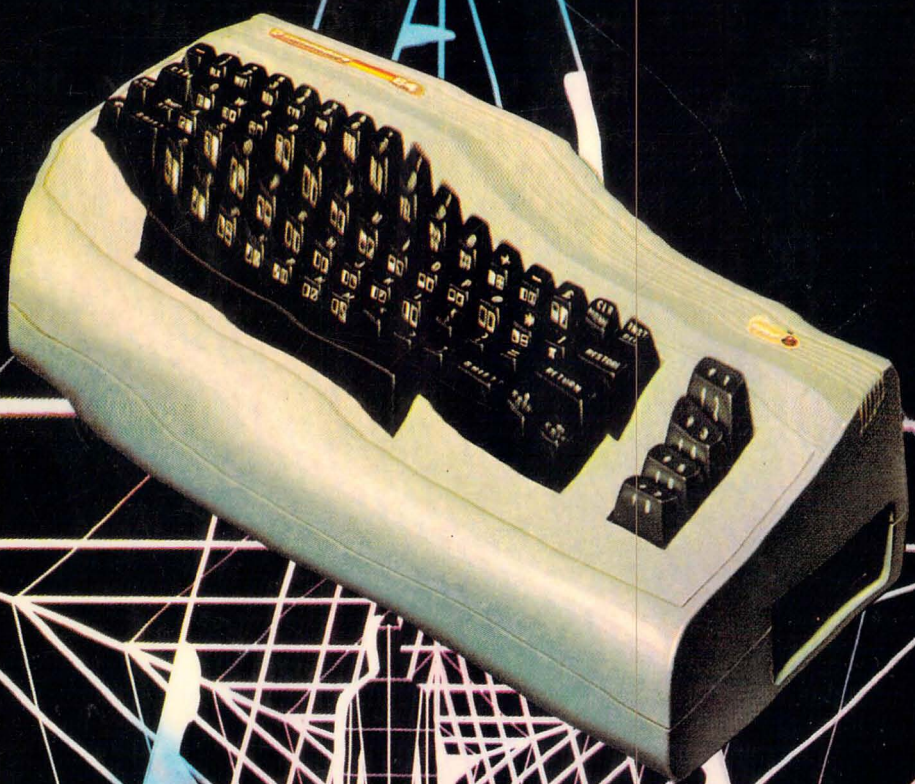
Barrett/Colwill

DAS GROSSE Commodore 64 SPIELE-BUCH



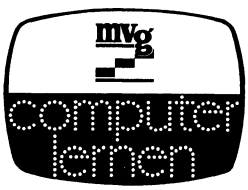
mvg
computer lernen

Terry Barrett – Stephen Colwill
Das große Commodore 64 Spiele-Buch

Terry Barret – Stephen Colwill

Das große Commodore 64 Spiele-Buch

Deutsche Bearbeitung: Dirk und Stefan Petry



moderne verlags gesellschaft

CIP-Kurztitelaufnahme der Deutschen Bibliothek

Barrett, Terry:
[Das große Commodore-vierundsechzig-Spiele-Buch]
Das große Commodore 64 Spiele-Buch/ Terry Barrett;
Stephen Colwill. [Dt. Bearb.: Petry, Dirk u. Stefan].
– Landsberg am Lech: Moderne Verlags-Gesellschaft, 1984.
(Computer lernen)
Einheitssacht.: Winning games on the Commodore 64 <dt.>
ISBN 3-478-09040-7
NE: Colwill , Steven; Petry, Dirk [Bearb.]

Titel der Originalausgabe: Winning games on the Commodore 64

This work was originally published as Barrett & Colwill: WINNING GAMES ON THE COMMODORE 64 (book) by Ellis Horwood Ltd.,
Publishers, Market Cross House, Cooper Street, Chichester, England

Deutsche Bearbeitung: Petry, Dirk und Stefan.

© Alle deutschen Rechte bei mvg – moderne verlags gesellschaft mbh
8910 Landsberg am Lech
Umschlaggestaltung: Hendrik van Gemert
Umschlagfoto: Artreference, Frankfurt
Satz: ABC, Fotosatz & Repro GmbH, München
Druck und Bindung: Druckerei Mühlthaler, München
Printed in Germany 090 040/9 84 402
ISBN 3-478-09040-7

Danksagung
Wir wollen allen danken, die uns geholfen haben, dieses Buch zusammenzustellen. Unser besonderer Dank gilt Antonia Jones, deren hilfreiche Begleitung alles erst möglich machte. Steve möchte auch Christine für ihre heimischen Bemühungen und seinen Eltern für deren ausdauernde Unterstützung danken. Auch Liz Coley gebührt Dank für ihre Hilfe.

STEVE COLWILL
TERRY BARRETT
Wokingham, Berkshire, England

Inhalt

Vorwort

Dieses Buch haben wir für drei Gruppen von Commodore 64-Benutzern geschrieben. Wenn Sie nur ein wenig oder überhaupt keine Programmiererfahrung besitzen, aber gern Computerspiele spielen, ist Abschnitt 2 das Richtige für Sie. Hier finden Sie die Listings der Spiele. Wenn Sie schon über ein Grundwissen von BASIC verfügen und anfangen wollen, Ihre eigenen Spiele auf dem 64er zu schreiben, dann wird Ihnen Abschnitt 1 in einfachen Etappen über die Schwierigkeiten der Zeichen- und Sprite-Definition und der Verwendung des Joysticks hinweghelfen. Wenn Sie schon einen hohen Programmierstandard erreicht haben, hoffen wir, daß Sie in Kapitel 9 und in den Listings selbst noch einige neue Tricks und Anregungen finden. Zusätzlich sind zwei Maschinenprogramme abgedruckt; das eine kopiert den Zeichensatz vom ROM ins RAM, das andere fragt den Joystick ab. Sie sind in einem BASIC-Lader abgelegt, so daß Sie sie direkt in Ihre BASIC-Programme einbauen können. Gleichgültig, welchen Grad von Erfahrung Sie erreicht haben, wir hoffen, daß für jeden etwas dabei ist. Da einige Programme dieses Buches Maschinenroutinen enthalten, ist es nötig, SYS64738 einzugeben oder den Computer aus- und wieder einzuschalten, bevor Sie ein neues Programm laden.

Wir haben auf die schwer zu identifizierenden Steuerzeichen, die sich häufig in Listings zum C-64 oder VC-20 finden, weitestgehend verzichtet. Das macht die Eingabe der Programme aus Abschnitt 2 vielleicht weniger nervenaufreibend. Wenn Sie feststellen, daß Ihre Finger schmerzen, erinnern Sie sich daran, daß alle Programme sowohl auf Diskette als auch auf Kassette beim Verlag erhältlich sind.

Der Commodore 64 bietet viele Möglichkeiten für den zukünftigen BASIC-Spiele-Programmierer. Sitzen Sie also nicht bloß herum... nutzen Sie sie!

1

Grundkenntnisse

In diesem Kapitel wollen wir die Grundregeln eines guten Programmierstils zusammenfassen. Anhand dieser Regeln ist es dem Leser möglich, gut strukturierte und leicht lesbare Programme zu schreiben. Es besteht natürlich kein Zwang, diesen Regeln zu folgen; es ist aber auch dem einzelnen überlassen, gute Programme zu schreiben oder nicht.

SCHLEIFEN

Wohl jeder Leser kennt die FOR ... NEXT-Schleife. Sie ermöglicht es uns, eine Gruppe von Befehlen beliebig oft zu wiederholen. Ein Beispiel:

```
10 PRINT"DIESES PROGRAMM IST";
20 FOR I=1TO10
30 PRINT"EXTREM"
40 NEXT
50 PRINT"LANGWEILIG!"
60 END
```

Wenn Sie dieses Programm starten, werden Sie feststellen, daß das Wort EXTREM zehnmal ausgegeben wird. Das bewirkt die Schleife, die in Zeile 20 beginnt.

```
20 FOR I=1TO10
 :
 :
40 NEXT
```

Der Computer beginnt die Schleife in Zeile 20, indem er der Variablen I den Wert 1 zuweist. Dann werden die folgenden Befehle ausgeführt, bis der Computer Zeile 40 erreicht. Dort sagt ihm der Befehl NEXT, daß er zu Zeile 20 zurückkehren und der Variablen I den Wert 2 zuweisen muß; wieder werden die folgenden Befehle abgearbeitet; wieder erreicht der Computer Zeile 40, die ihm befiehlt, zu Zeile 20 zurückzukehren, der Variablen I den Wert 3 zuzuweisen, usw.

Schließlich hat I den Wert 10, und die Schleife wird zum letzten Mal durchlaufen. Der Computer setzt dann in Zeile 50 den Programmablauf fort.

Der Wert der Variablen I kann innerhalb der Schleife benutzt werden:

```
10 FORI=1TO10
20 PRINT I
30 NEXT
40 PRINT"HIER IST DIE SCHLEIFE ZU ENDE"
50 END
```

Dieses Programm gibt nacheinander die Zahlen von 1 bis 10 aus, gefolgt von einer Meldung, daß das Schleifenende erreicht ist.

Das Programm läßt sich einfach so abändern, daß nur die geraden Zahlen ausgegeben werden. Wir müssen dem Computer nur mitteilen, daß er die Schleife mit I gleich 2 beginnen und dieses I nach jedem Durchlauf um 2 hochzählen soll. Wir erreichen dieses Ziel, indem wir Zeile 10 durch die folgende ersetzen:

```
10 FORI=2TO10STEP2
```

Wir können den Computer auch rückwärts zählen lassen. Ändern Sie dazu die Zeile 10 folgendermaßen:

```
10 FORI=10TO1STEP-1
```

Oder wir geben wie zuvor nur die geraden Zahlen aus, jedoch in umgekehrter Reihenfolge. Ersetzen Sie Zeile 10 durch:

```
10 FORI=10TO1STEP-2
```

Wir können sogar eine zweite Schleife innerhalb der ersten einrichten:

```
10 FORI=1TO10
20 FORJ=1TO3
30 PRINTI,J
40 NEXTJ
50 NEXTI
60 END
```

Dieses Programm gibt einfach Zahlenpaare aus. Die Ausgabe beginnt:

```
I       J

1       1
1       2
1       3
2       1
2       2 usw.
```

Beachten Sie, daß die 'J'-Schleife sich völlig innerhalb der 'I'-Schleife befindet. Man sagt, die Schleifen sind „geschachtelt". Hier ein Beispiel für falsch geschachtelte Schleifen:

```
10 FORI=1TO10
20 FORJ=1TO3
30 PRINTI,J
40 NEXTI
50 NEXTJ
60 END
```

Auf den ersten Blick scheint sich nicht viel geändert zu haben; nach genauerer Betrachtung werden Sie aber feststellen, daß die Zeilen 40 und 50 ausgetauscht wurden. Die „J"-Schleife ist nun nicht mehr völlig in der „I"-Schleife enthalten. Auf diese Weise geschachtelte FOR...NEXT-Befehle kann der Computer nicht mehr ausführen, das Ergebnis ist eine Fehlermeldung. Um falsches Schachteln zu vermeiden, erlaubt der Computer, das „I" im Befehl NEXTI und das „J" im Befehl NEXTJ, also die Variable, wegzulassen und die Befehle zu NEXT zu kürzen. Der Computer interpretiert dann die Schleifen automatisch als korrekt geschachtelt.

GOTO UND UNTERPROGRAMME

Der GOTO-Befehl ist ein sehr wirkungsvolles Mittel, um den Programmablauf zu steuern. Er ist außerdem der am häufigsten mißbrauchte Befehl in BASIC! GOTO sollte nur sparsam und mit Bedacht eingesetzt werden. Das folgende Programm zeigt, wie man GOTO nicht gebrauchen sollte:

```
10 INPUTA$
20 IFA$="JA"THENGOTO40
30 GOTO10
40 PRINT"WELCH EINE VERZWEIGUNG"
50 END
```

Vergleichen Sie dieses Programm mit dem folgenden, das den gleichen Zweck erfüllt.

```
10 INPUTA$
20 IFA$<>"JA"THEN10
30 PRINT"SCHON BESSER"
40 END
```

Benutzen Sie niemals GOTO, wenn Sie es vermeiden können.

Unterprogramme sind um einiges intelligenter als GOTOs, denn sie können sich merken, von wo aus sie gestartet wurden. Ein Unterprogramm ist eine Folge von Befehlen, die innerhalb eines Programms an mehreren Stellen durchlaufen werden muß. Unterprogramme werden an das Ende des Hauptprogramms, hinter den END-Befehl, gelegt. Um ein Unterprogramm von einem anderen Programmteil aus aufzurufen, benutzt man den Befehl GOSUB, gefolgt von der Zeilennummer, bei der das Unterprogramm beginnt. Am Ende des Unterprogramms bewirkt der Befehl RETURN, daß der Computer zum Hauptprogramm, d.h. direkt zu dem Punkt zurückkehrt, von dem das Unterprogramm aufgerufen wurde. Er arbeitet dann mit dem Befehl weiter, der dem GOSUB-Befehl folgt. Hier ein einfaches Beispiel, wie ein Unterprogramm arbeitet:

```
 10 PRINT"HALLO!"
 20 GOSUB1000
 30 PRINT"NA, WIE GEHT'S?"
 40 GOSUB1000
 50 PRINT"AUF EINE SOLCHE UNTERHALTUNG HABE ICH
    KEINE LUST."
 60 GOSUB1000
 70 PRINT"ICH MUSS JETZT WIRKLICH GEHEN."
 80 END
1000 REM HIER BEGINNT DAS UNTERPROGRAMM
1010 PRINT"HALLO, ICH BIN DAS UNTERPROGRAMM!"
1020 RETURN
```

DIE PROGRAMM-STRUKTURIERUNG

Das allgegenwärtige GOTO erhebt wieder sein häßliches Haupt! Eine der goldenen Regeln des „guten" Programmierens ist, sicherzustellen, daß nach Möglichkeit jeder Programmabschnitt EINEN Eingang und EINEN Ausgang hat. Nach diesem Prinzip aufgebaut, sind Ihre Programme nicht nur für andere leichter zu lesen, sondern auch Sie haben weniger Schwierigkeiten, das Programm bei der Fehlersuche zu überblicken.

Beim Ausführen eines GOSUB-Befehls merkt sich der Computer die Zeilennummer und die Position des GOSUB innerhalb der Zeile im „Stack". Der Stack, zu deutsch Stapel, ist ein spezieller Speicherbereich, mit dem der Prozessor des Computers arbeitet. Jedesmal, wenn ein GOSUB-Befehl abgearbeitet wird, werden im Stack fünf Bytes an Informationen über die Position des GOSUB abgelegt. Diese fünf Bytes werden erst gelöscht, wenn ein RETURN-Befehl erreicht ist. Das bedeutet: Wenn Sie vor einem RETURN aus einem Unterprogramm herausspringen, wird der Stack nicht gelöscht. Wie man leicht erkennt, kann dieses Herausspringen nur begrenzt oft durchgeführt werden, nämlich bis der gesamte Stack mit GOSUB-Positionen gefüllt ist. Schreiben Sie also keine Unterprogramme wie dieses:

```
1000 REM HIER BEGINNT EIN UNTERPROGRAMM
1010 . . .
1020 . . .
1030 IFX=30THEN200
1040 . . .
1050 . . .
1060 RETURN
```

Wir sehen, daß, wenn in Zeile 1030 die Bedingung X=30 erfüllt ist, der Computer niemals Zeile 1060 erreicht und den Stack-Eintrag löscht.

Das bedeutet, daß Sie sich beim Entwurf Ihres Programms vielleicht etwas mehr Zeit lassen sollten, um solche Fehler zu vermeiden; Sie gehen so aber auch sicher, daß Ihre Programme immer fehlerfrei laufen.

Dieselbe Regel sollte auch bei FOR...NEXT-Schleifen Anwendung finden. Legen Sie das Programm immer so an, daß jedes NEXT ausgeführt wird, bis die Laufvariable ihren Endwert erreicht hat. Mit anderen Worten, programmieren Sie auf keinen Fall so:

```
10 FORI=1TO10
20 . . .
30 . . .
40 IFX>120THEN200
50 . . .
60 . . .
70 NEXT
```

Wenn Sie diese Regeln befolgen, werden Ihnen später einige Kopfschmerzen erspart bleiben!

2

PEEK und POKE

SPEICHERORGANISATION

Der Speicher des Commodore 64 ist in kleine Einheiten eingeteilt, die man sich wie Briefkästen vorstellen kann. Um einen Briefkasten vom anderen zu unterscheiden, trägt jeder eine Nummer. Diese Nummer nennt man die „Adresse". Der eigentliche Name eines Briefkastens lautet „Speicherstelle".

Der Speicher des Computers ist also in Speicherstellen aufgeteilt, die jede eine bestimmte Adresse haben. Was ist daran weiter interessant?

Da im Commodore-BASIC nur sehr wenig Grafik-Befehle existieren, müssen wir den Speicher des Computers direkt beeinflussen. Das erreichen wir mit den Befehlen PEEK und POKE.

PEEK UND POKE

Die Funktion PEEK greift auf eine angegebene Speicherstelle zu und stellt fest, welchen Wert diese enthält. Löschen Sie den Bildschirm Ihrer Anlage, indem Sie gleichzeitig SHIFT und CLR/HOME drücken. Geben Sie dann folgende Zeile ein:

```
PRINT PEEK(2023)
```

Der Computer sollte daraufhin die Zahl 32 ausgeben. Was ist geschehen?

Die Zahl 2023 ist die Adresse einer bestimmten Speicherstelle. Diese Speicherstelle ist der Briefkasten, der das Zeichen enthält, das in der unteren rechten Ecke des Bildschirms steht.

Die Zeile, die wir eingegeben haben, sagt dem Computer, daß er die Speicherstelle 2023 PEEKen und das Ergebnis ausgeben soll.

Die Zahl 32 ist der Bildschirmcode des Leerzeichens. In der rechten unteren Ecke des Bildschirms befindet sich also ein Leerzeichen, denn Sie haben zuvor den Bildschirm gelöscht.

Probieren Sie diese Zeile aus:

```
PRINT PEEK(56295)
```

Das Ergebnis dieses Befehls sollte die Zahl 6 sein. Die Speicherstelle mit der Adresse 56295 enthält nämlich die Farbe des Zeichens in der rechten unteren Ecke des Bildschirms.

Löschen Sie wieder den Bildschirm und schreiben Sie:

```
POKE2023,1
```

Es scheint nichts geschehen zu sein; schreiben Sie, ohne den Bildschirm zu löschen:

```
POKE56295,0
```

Haben Sie alles richtig gemacht, dann ist jetzt der Buchstabe „A" in der rechten unteren Ecke des Bildschirms zu sehen. Wie konnte ein solches „Wunder" geschehen?

Der erste POKE-Befehl schreibt den Wert 1 in die Speicherstelle 2023. 1 ist der Bildschirmcode des Buchstabens „A". Es schien nichts geschehen zu sein, doch der Buchstabe „A" war da, nur war er blau gefärbt, wie der Hintergrund. Das zweite POKE war notwendig, um dem „A" eine andere Farbe zu geben; 0 ist der Farbcode für Schwarz.

Fassen wir zusammen:

Jede Position auf dem Bildschirm hat zwei zugehörige Speicherstellen. Eine enthält den Bildschirmcode des Zeichens, das gerade an der betreffenden Position steht; die andere Speicherstelle enthält den Farbcode dieses Zeichens.

Der Bildschirm des Commodore 64 besteht aus 25 Zeilen mit Platz für je 40 Zeichen. Das heißt, es existieren 1000 Positionen, auf denen sich Zeichen befinden können.

BILDSCHIRM- UND FARBSPEICHER

Die Speicherstellen des Bildschirmspeichers laufen von 1024 (linke obere Ecke) bis 2023 (rechte untere Ecke). Wie schon gesagt, enthält der Bildschirmspeicher die Codes der Zeichen, die auf dem Bildschirm erscheinen.

Die korrespondierenden Speicherstellen des Farbspeichers liegen auf 55296 bis 56295.

Wenn Sie ein Zeichen auf dem Bildschirm erscheinen lassen wollen, müssen Sie sowohl den Bildschirm- als auch den Farbspeicher POKEn, sonst bleibt das Zeichen, das Sie gePOKEt haben, unsichtbar. Eine nützliche Beziehung zwischen beiden ist die folgende:

```
FARBSPEICHERSTELLE=BILDSCHIRMSPEICHERSTELLE+54272
```

Um zu sehen, wie man diese Beziehung ausnutzen kann, geben Sie einmal dieses Programm ein:

```
10 PRINT CHR$(147):REM BILDSCHIRM LOESCHEN
20 SC=1504
30 FORI=0TO39
40 POKESC+I,26
50 POKESC+54272+I,2
60 NEXT
70 END
```

Zeile 20 initialisiert SC (für „screen" (engl.: Bildschirm)) mit einer Bildschirmadresse, die etwa in der mittleren Zeile am linken Rand liegt.

Die Schleife in den Zeilen 30 bis 60 POKEt in die Speicherstellen von 1504 bis 1543 den Wert 26, der den Bildschirmcode des Buchstaben „Z" darstellt. In Zeile 50 wird die korrespondierende Speicherstelle des Farbspeichers errechnet und in diese der Wert 2 gePOKEt; 2 ist der Farbcode für Rot.

Mit welcher Bildschirmadresse würden Sie SC initialisieren, wenn die Zeile „Z"s am unteren Bildschirmrand erscheinen sollte?

Wie würden Sie Zeile 50 ändern, wenn grüne „Z"s gewünscht wären?

Schreiben Sie ein Programm, das mehrere Zeilen gelber „E"s erzeugt (der Bildschirmcode von „E" ist 5)!

Eine vollständige Liste aller Bildschirmcodes befindet sich in Ihrem Commodore-64-Handbuch, Anhang E.

Diagramme der Adreßbereiche des Bildschirm- und des Farbspeichers sind zusammen mit den 16 Farbcodes im Anhang G des Handbuchs abgedruckt.

WAS IST SONST NOCH INTERESSANT?

Wollen Sie die Hintergrundfarbe ändern, so können Sie dies durch den folgenden POKE-Befehl erreichen:

```
POKE53281,F
```

F ist einer der Farbcodes, die in Anhang G beschrieben werden.

Die Randfarbe kann man verändern, indem man den gewünschten Farbcode in die Speicherstelle 53280 POKEt.

WIE GEBRAUCHEN WIR UNSER NEUES WISSEN?

Wir wollen nun ein kleines Programm schreiben, das Bewegung und „Animation" auf den Bildschirm bringt – der erste Schritt auf dem Weg zum Grafik-Spiel.

Wir wollen ein rotes „X" über drei Zeilen wandern lassen; wir beginnen damit, daß wir das erste „X" an Adresse 1504 POKEn:

```
10 PRINTCHR$(147):REM BILDSCHIRM LOESCHEN
20 SC=1504
30 POKESC,24:REM"X"
40 POKESC+54272,2:REM ROT FAERBEN
```

Löschen Sie den Bildschirm und schreiben Sie weiter:

```
 50 FORI=0TO118
 60 POKESC+I,32:REM LEERZEICHEN
 70 POKESC+I+1,24:REM "X"
 80 POKESC+I+54272,6:REM BLAU FAERBEN
 90 POKESC+I+54272+1,2:REM ROT FAERBEN
100 NEXT
110 END
```

Betrachten Sie sorgfältig, was die Schleife in den Zeilen 50 bis 100 bewirkt. Sie POKEt nicht einfach ein Zeichen auf den Bildschirm, sondern zwei nacheinander.

Die Schleife POKEt „_X". Hierbei ist das Leerzeichen vor dem „X" durchaus beabsichtigt. Wenn die Schleife einmal durchlaufen wird, bewegt sich das „X" um eins nach rechts. Dabei wird das Leerzeichen an die Stelle des alten „X" gePOKEt; das alte „X" wird also gelöscht. Das neue „X" wird dann rechts neben das alte gePOKEt.

Wenn Sie dieses kleine Programm nun laufen lassen, werden Sie feststellen, daß sich das „X" wie der Blitz über den Bildschirm bewegt. Um die Sache etwas abzubremsen, können wir eine sogenannte Warteschleife in die Hauptschleife einbauen. Eine Warteschleife dient keinem anderen Zweck als den Programmablauf zu bremsen. Geben Sie ein:

```
55 FORJ=1TO500
56 NEXT
```

Starten Sie das Programm nun wieder und achten Sie auf den Unterschied. Mit dem Endwert der Warteschleife können Sie die Geschwindigkeit des gesamten Programms bestimmen. Sie können das Programm praktisch unbegrenzt verlangsamen; die maximale Geschwindigkeit ist allerdings durch die Zeit festgelegt, die die Hauptschleife für einen Durchlauf benötigt.

Hierbei entdecken wir ein wichtiges Prinzip der Computeranimation: Die Geschwindigkeit der Bewegung auf dem Bildschirm hängt davon ab, wieviel Zeit der Computer für einen Durchlauf der Hauptprogrammschleife benötigt. Aber damit wollen wir uns später näher befassen.

INVADERS

Bis jetzt haben wir nur kurze Programme betrachtet. Das vorliegende Programm ist das erste etwas längere; es soll einige Techniken der Grafikanimation demonstrieren. Das Programm basiert auf dem klassischen „Space Invaders"-Thema, wurde allerdings für unsere Zwecke etwas vereinfacht.

Mit den Tasten „Z" und „X" bewegen Sie den Laser nach links und rechts. Taste „M" löst den Schuß aus.

Das Programm ist eigentlich mehr dazu gedacht, leicht verstanden zu werden, statt dem Benutzer ein aufregendes Spiel zu bieten. Dem Listing des Programms folgt eine detaillierte Beschreibung jeder einzelnen Zeile.

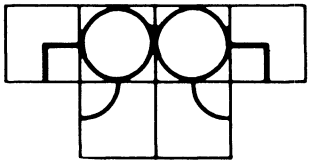
```
1 REM  ***************
2 REM  ***************
3 REM  **             **
4 REM  ** INVADERS **
5 REM  **   DEMO    **
6 REM  ** PROGRAMM **
7 REM  **             **
8 REM  ***************
9 REM  ***************
10 PRINTCHR$(147)
20 POKE53281,0
30 S=1024:A=1999:C=55296:SC=1024:CC=5529
6
40 FORL=1TO2
50 FORI=1TO10
60 POKESC,112:POKESC+1,87:POKESC+2,87
70 POKESC+3,110:POKESC+41,75
80 POKESC+42,74
90 FORJ=0TO3
100 POKECC+J,5:NEXT
110 POKECC+41,5:POKECC+42,5
120 SC=SC+4:CC=CC+4
130 NEXT
140 SC=S+80:CC=C+80
150 NEXT
200 REM *** HAUPTPROGRAMMSCHLEIFE ***
210 REM *** 'ALIEN' LOESCHEN ***
220 FORI=0TO3
230 POKES+I,32
240 NEXT
250 POKES+41,32:POKES+42,32
```

```
260 GOSUB1000:REM UNTERPROGRAMM 'LASER'
270 REM *** NEUES 'ALIEN' AUSGEBEN ***
280 SC=S+160:CC=SC+54272
290 POKESC,112:POKESC+1,87:POKESC+2,87
300 POKESC+3,110:POKESC+41,75::POKESC+42
,74
310 FORJ=0TO3
320 POKECC+J,5:NEXT
330 POKECC+41,5:POKECC+42,5
340 GOSUB1000:REM UNTERPROGRAMM 'LASER'
350 GETJ$:IFJ$<>""THEN350
360 S=S+4
370 T=(S-984)/40:IFT=INT(T)THENS=S+40
500 IFPEEK(1744)=32THENGOTO200:REM SCHLE
IFE WIEDERHOLEN
600 REM *** PROGRAMMENDE ***
610 REM *** BILDSCHIRM BLINKEN LASSEN **
*
620 PRINTCHR$(147)
630 FORI=1TO5
640 POKE53281,5:FORJ=1TO300:NEXT
650 POKE53281,2:FORJ=1TO300:NEXT
660 NEXT
670 REM *** MELDUNG AUSGEBEN ***
680 FORI=1TO12:PRINT:NEXT
690 PRINTTAB(11)"DU BIST GESCHLAGEN"
700 PRINT:PRINT
710 PRINTTAB(9)"<<LEERTASTE DRUECKEN>>"
720 GETJ$:IFJ$<>""THEN720
730 GETB$:IFB$<>" "THEN730
740 RUN
1000 REM *** UNTERPROGRAMM 'LASER' ***
1010 B=54272+A
1020 POKEA,160:POKEA+1,160:POKEA-40,108
1030 POKEA-39,123
1040 POKEB,2:POKEB+1,2:POKEB-40,2
1050 POKEB-39,2
1060 POKEA-2,32:POKEA-1,32:POKEA-42,32
1070 POKEA-41,32:POKEA+2,32:POKEA+3,32
1080 POKEA-38,32:POKEA-37,32
1090 REM *** LASER BEWEGEN ***
1100 GETA$
1110 IFA$="Z"THENA=A-2:IFA<1984THENA=198
4
1120 IFA$="X"THENA=A+2:IFA>2022THENA=202
2
```

```
1130 IFA$="M"THENGOSUB2000
1140 RETURN
2000 REM *** UNTERPROGRAMM 'FEUERN' ***
2010 P=A-119
2020 FORI=1TO22
2030 POKEP,117:POKEP+54272,1
2040 POKEP+40,32
2050 P=P-40
2060 IFPEEK(P)<>32THENGOSUB3000
2070 NEXT
2080 POKEP+40,32
2090 RETURN
3000 REM *** UNTERPROGRAMM 'TREFFER' ***
3010 POKEP+40,32
3020 IFPEEK(P)=112THENH=P+40
3030 IFPEEK(P)=110THENH=P+37
3040 IFPEEK(P)=75THENH=P-1
3050 IFPEEK(P)=74THENH=P-2
3060 FORI=0TO3
3070 POKEH+I,32:NEXT
3080 FORI=40TO37STEP-1
3090 POKEH-I,32:NEXT
3100 RETURN
```

(1) Aufbau des Bildschirms

Zeile 10:	Bildschirm löschen.
Zeile 20:	Bildschirm schwarz färben.
Zeile 30:	Variablen initialisieren, die später noch gebraucht werden. S ist der Anfang des Bildschirmspeichers. SC ist die Bildschirmspeicherstelle, in die gerade ein „Alien" gePOKEt wird. A ist die Bildschirmspeicherstelle, an der gerade der Laser steht. C (für „colour") ist der Anfang des Farbspeichers. CC ist die mit SC korrespondierende Speicherstelle im Farbspeicher.
Die Schleife von Zeilen 50–130:	Läßt eine Zeile von „Aliens" auf dem Bildschirm erscheinen.
Zeilen 60–110:	Einzelnes „Alien" in den Bildschirmspeicher POKEn. Das „Alien" wird aus sechs Zeichen aufgebaut.



Nimmt man an, daß sich das linke obere Zeichen an der Bildschirmspeicherstelle SC befindet, dann kann man die Positionen der anderen fünf Zeichen wie folgt errechnen:

SC	SC+1	SC+2	SC+3
	SC+41	SC+42	

Die Bildschirmcodes der einzelnen Zeichen sind:

112	87	87	110
	75	74	

Der Farbcode, der entsprechend in den Farbspeicher gePOKEt wird, ist 5, also Grün.

Zeile 120: Nachdem ein Alien gePOKEt ist, wird die Bildschirmposition SC um vier erhöht, damit beim nächsten Schleifendurchlauf das folgende „Alien" direkt neben das vorhergehende gesetzt wird.

Zeile 150: Ist eine Zeile von „Aliens" ausgegeben, so muß die Bildschirmpositon SC um 40 erhöht werden. Hierdurch wird eine Bildschirmzeile übersprungen, bevor die nächste Zeile „Aliens" gePOKEt wird.

(2) Hauptprogrammschleife

(a) Altes „Alien" löschen

Zeilen 220–250: Wie Sie wissen, war S der Anfang des Bildschirmspeichers, der in seinem ursprünglichen Wert 1024 während des Aufbaus des Bildschirms nicht verändert wurde. S wird nun als Position für das Löschen des ersten „Aliens" verwendet.
32 ist der Bildschirmcode des Leerzeichens.

Zeile 260: An diesem Punkt wird das Unterprogramm in Zeile 1000 ausgeführt, um den Laser zu bewegen.

(b) Neues „Alien" POKEn

Zeile 280: Die Bildschirmposition SC nimmt nun einen Wert an, der vier Zeilen tiefer liegt als die Position S. Die Farbspeicherposition wird SC angepaßt.

Zeilen 290–330: Wie in den Zeilen 80–120 wird ein neues „Alien" an der durch SC definierten Position aufgebaut.

Zeile 340: Die Routine für die Steuerung des Lasers wird nochmals aufgerufen.

Zeile 350: Tastaturpuffer löschen.
Um zu vermeiden, daß eingegebene Zeichen verlorengehen, wenn Tasten sehr schnell hintereinander gedrückt werden, werden die Zeichen zuerst in einem sogenannten Puffer gespeichert, bis der Computer in der Lage ist, sie zu verarbeiten. Der Tastaturpuffer kann bis zu zehn Zeichen aufnehmen, die dann nach dem System „FIFO" („First-In-First-Out", das zuerst in den Puffer gelangte Zeichen wird auch zuerst wieder ausgelesen) durch den Computer verarbeitet werden. Um sicherzustellen, daß in Zeile 1100 nur ein Zeichen gelesen werden kann, springt sich diese Zeile immer wieder selbst an, bis der Tastaturpuffer völlig leer ist.

Zeile 360: Zur Vorbereitung des nächsten Hauptschleifendurchlaufs wird S um vier erhöht.

Zeile 370: Ergibt es sich, daß S eine Position hat, die am Anfang einer Zeile liegt, so wird S nochmals um 40 erhöht, um eine Bildschirmzeile zu überspringen.

Zeile 500: Normalerweise enthält die Bildschirmspeicherstelle 1744 den Wert 32, also ein Leerzeichen. Zeile 500 überprüft, ob dies immer noch der Fall ist, oder ob ein „Alien" schon diese Stelle erreicht hat. Nur wenn diese Speicherstelle einen anderen Wert als 32 enthält, darf das Programm mit der „Spiel-Ende"-Routine fortfahren.

(3) Spiel-Ende

Zeile 650: Bildschirm löschen.

Zeilen 630–660: Schleife, die den Bildschirm zuerst grün, dann rot färbt. Beachten Sie hierbei die Verwendung von Warteschleifen, um das Aufblitzen der Farben abzubremsen.

Zeilen 670–710: Um eine Meldung auszugeben, wird der Ausgabecursor in der Mitte des Bildschirms positioniert. Beachten Sie, daß der PRINT-Befehl bewirkt, daß der Cursor sich zuvor eine Zeile nach unten bewegt. TAB(X) bewegt den Cursor auf eine Position, die X Zeichen vom linken Bildschirmrand entfernt ist.

Zeile 720: Tastaturpuffer löschen (Erklärung siehe oben).

Zeile 730: Auf einen Druck der Leertaste warten.

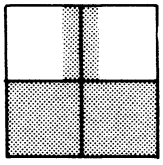
Zeile 740: Startet das Programm von vorn.

(4) Die Unterprogramme

(a) Das „Laser"-Unterprogramm

Zeile 1010: B ist der Farbcode des Lasers.
A ist die Position des Lasers.
Zeile 1010 paßt B an A an.

Zeilen 1010–1050: Die Zeichen und Farben des Lasers an die entsprechenden Stellen POKEn.
Der Laser besteht aus 4 Zeichen:



Hat das linke untere Zeichen die Bildschirmposition A, so errechnen sich daraus die Positionen der restlichen Zeichen wie folgt:

A−40	A−39
A	A+1

Die zugehörigen Bildschirmcodes sind:

108	123
160	160

Beachten Sie, daß ein inverses Zeichen einen Bildschirmcode von 128 + (Code des entsprechenden nichtinversen Zeichens) hat. Ein inverses Leerzeichen hat also den Bildschirmcode 128+32=160.
Der Farbcode des Lasers ist 2; das entspricht Rot.

Zeilen 1060–1080: Einen Block aus vier Leerzeichen an jede Seite der aktuellen Laserposition POKEn. Wenn der Laser nach rechts oder nach links bewegt wird, löscht diese Zeile den alten Laser.

Zeile 1100: Der GET-Befehl liest ein auf der Tastatur eingegebenes Zeichen und speichert es in der Stringvariablen A$.

Zeile 1110: Ist das eingegebene Zeichen ein „Z", so wird die Position des Lasers um zwei verringert, d.h., wenn der Laser von neuem gezeichnet wird, entsteht er zwei Zeichen weiter links.
Der zweite Teil dieser Zeile legt den unteren Grenzwert für A fest. Wenn A einen kleineren Wert als 1984 annehmen würde, dann erschiene der Laser eine Zeile weiter oben auf der rechten Seite des Bildschirms an Position 1982.

Zeile 1120: Entsprechend wird bei der Eingabe von „X" die Position des Lasers um 2 erhöht. Weiterhin wird auch der obere Grenzwert für A festgelegt. (Vorsicht! Es können eigenartige Dinge geschehen, wenn Sie unabsichtlich irgendwelche Werte in Adressen POKEn, die größer als 2023 sind.)

Zeile 1130: Wurde die Taste „M" gedrückt, so wird das Unterprogramm in in Zeile 2000 aufgerufen.

Zeile 1140: Rückkehr zum Hauptprogramm.

(b) „Feuer"-Unterprogramm

Zeile 2010: P wird an A angepaßt. P ist die Anfangsposition des Geschosses.

Zeile 2020: Ab hier bildet das Unterprogramm eine Schleife. Nach jedem Schleifendurchlauf wird I um Eins erhöht.
Nimmt I den Wert 22 an, so hat das Geschoß den oberen Rand des Bildschirms erreicht. In diesem Fall wird das Geschoß gelöscht, und das Programm kehrt wieder zur Hauptschleife zurück.

Zeile 2030: Zeichen und Farbe des Geschosses an Position P des Bildschirms POKEn.

Zeile 2040: Geschoß an vorhergehender Position löschen.

Zeile 2050: P wird um 40 erniedrigt; das bedeutet, P enthält jetzt eine Position, die um Eins weiter oben auf dem Bildschirm liegt.

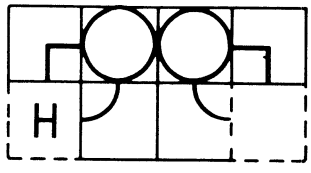
Zeile 2060: Das PEEK greift auf die Speicherstelle zu, die direkt über dem aktuellen Wert von P liegt. Enthält diese Speicherstelle einen anderen Wert als 32 (also kein Leerzeichen), so wird ein Unterprogramm in Zeile 3000 aufgerufen. Dieses Unterprogramm löscht das Geschoß und das getroffene „Alien".
Merke: Es ist möglich, von einem Unterprogramm aus ein anderes aufzurufen; es existiert allerdings ein Höchstwert für die Anzahl der Unterprogrammebenen, die man bei einer solchen Gelegenheit schaffen kann.

Zeile 2090: Rückkehr zu Unterprogramm 1000.

(c) „Treffer"-Unterprogramm

Zeile 3010: Altes Geschoß löschen.

Zeilen 3020–3050: PEEK der Position P. Die Position kann eins von vier möglichen Zeichen des „Alien" enthalten. Der Variablen H wird die Adresse des folgenden Bildschirmspeicherplatzes zugewiesen:



Zeilen 3070–3090: Löschen des getroffenen Aliens durch POKEn eines Blocks von 4*2 Leerzeichen in folgende Speicherstellen:

H–40	H–39	H–38	H–37
H	H+1	H+2	H+3

Zeile 3100: Rückkehr zu Unterprogramm 2000.

PROGRAMMSTRUKTUR

Beachten Sie, daß das Programm aus drei getrennten Abschnitten besteht:

1. Die „Aufbau"-Routine.
2. Die Hauptprogrammschleife.
3. Unterprogramme, die nach Bedarf aufgerufen werden.

Durch den Einbau der „Laser"-Steuerung in ein Unterprogramm erreicht man, daß sie mehrmals während eines Durchlaufs der Hauptprogrammschleife angesprochen werden kann. Durch diese Struktur werden viele Probleme der benutzerkontrollierten Bewegung behoben, die in BASIC-Spielen dieser Art meistens nur unbefriedigend gelöst sind. Die Feinfühligkeit der Steuerung kann um vieles verbessert werden, wenn in der Hauptschleife mehrmals Steuerungsunterprogramme aufgerufen werden.

3

Ansteuerung des Bildschirms mit PRINT

Im letzten Kapitel haben wir untersucht, wie man Zeichen durch POKE-Befehle auf dem Bildschirm erscheinen lassen kann. Eine zweite Möglichkeit, dies zu bewerkstelligen, ist der PRINT-Befehl. Der PRINT-Befehl des Commodore-BASIC ist ein sehr vielseitiges Werkzeug, das mehrere verschiedene Wirkungen haben kann. In diesem Kapitel wollen wir die unterschiedlichen Funktionen des PRINT-Befehls in Ihrem Commodore 64 behandeln.

PRINT UND DIE INTERPUNKTION

Mit einigen wichtigen Ausnahmen, die wir aber später betrachten wollen, arbeitet der PRINT-Befehl nach zwei Mustern. Erstens bewirkt der Befehl

```
PRINT X
```

eine Ausgabe des aktuellen Wertes der Variablen X auf dem Bildschirm. Im zweiten Arbeitsmuster kopiert PRINT eine Zeichenkette auf den Bildschirm, wobei diese Zeichenkette in Anführungszeichen dem PRINT-Befehl folgen muß. Z.B.

```
PRINT"GUTEN MORGEN"
```

Es ist auch möglich, mehrere Variablen und/oder Zeichenketten innerhalb eines PRINT-Befehls auszugeben, indem man Interpunktionszeichen setzt.

Wenn zwei auszugebende Elemente durch ein SEMIKOLON (;) getrennt werden, so erscheint das zweite Element direkt hinter dem ersten:

```
10 A$="HAL":B$="LO"
20 PRINTA$;B$
```

Dieses Programm PRINTet das Wort HALLO zusammenhängend auf den Bildschirm. Zu dieser Regel existieren zwei Ausnahmen:

(1) Dem Ausdruck einer numerischen Variablen folgt immer ein zusätzliches Leerzeichen.
(2) Haben diese numerischen Variablen einen positiven Wert, so geht dem Ausdruck des eigentlichen Werts eine Leerstelle voraus. (Falls der Wert negativ ist, wird diese Leerstelle vom Minus-Zeichen eingenommen.)

Der Bildschirm ist in der Vertikalen in vier je zehn Zeichen breite Zonen eingeteilt. Werden zwei Elemente in einer PRINT-Zeile mit einem KOMMA (,) getrennt, so wird das zweite Element am Anfang der nächsten Zone ausgegeben. Diese Möglichkeit ist besonders nützlich, wenn Sie auf dem Bildschirm Tabellen mit irgendwelchen Werten o.ä. erstellen wollen. Ändern wir im obigen Programmbeispiel Zeile 20 ab, indem wir das Semikolon durch ein Komma ersetzen, so wird die Ausgabe folgendermaßen aussehen:

```
HAL       LO
```

Endet ein PRINT-Befehl mit einem Semikolon oder Komma, dann hat dies zur Folge, daß der nächste PRINT-Befehl seine Ausgabe nach den soeben beschriebenen Regeln der Interpunktion dort fortsetzt, wo der erste PRINT-Befehl aufgehört hat.

Endet der Befehl nicht mit einem Interpunktionszeichen, so beginnt der folgende seine Ausgabe am Anfang einer neuen Zeile.

AUSGABEFORMATIERUNG

(1) Die TAB-Funktion

Die TAB-Funktion erlaubt es Ihnen, innerhalb eines PRINT-Befehls den Anfang der Ausgabe eines Elementes an eine bestimmte Position der Zeile zu legen. Die Form des TAB-Befehls ist folgende:

```
PRINTTAB(17)"MITTE"
```

Diese Zeile bewirkt, daß das Wort Mitte ab der siebzehnten Spalte (die erste Spalte zählt als Spalte 0) ausgegeben wird. Die vertikale Position des Wortes MITTE hängt davon ab, in welcher Position sich der Cursor zuvor befand. Der Bildschirm ist 40 Zeichen breit, aber in der TAB-Funktion kann ein ganzzahliger Wert zwischen 0 und 255 angegeben werden. TAB(80) würde also bewirken, daß das nächste Element des PRINT-Befehls zwei Zeilen unter der bisherigen Cursor-Position ausgegeben wird.

(2) Die Steuerzeichen

Sie werden wahrscheinlich schon einige seltsame Erlebnisse beim Verbessern von Zeichenketten in Anführungszeichen gehabt haben. Tippen Sie nämlich ein Anführungszeichen ("), dann arbeiten die Cursortasten nicht mehr wie sonst. Man sagt, die Cursortasten befinden sich im Anführungszeichen-Modus. Die sogenannten Steuerzeichen, die sie nun erzeugen, erscheinen zwar als inverse Zeichen innerhalb der in Anführungszeichen gesetzten Zeichenkette; sie haben aber nur die Funktion, den Cursor während des Programmablaufs zu steuern. Als Beispiel sollten Sie das folgende Programm eingeben und starten. Es demonstriert auch die eingebaute Uhr des CBM 64, die Variable TI$.

```
5 REM*** KAPITEL 3 PROGRAMM 1 ***
10 REM  ************
20 REM  *          *
30 REM  * STOPPUHR *
40 REM  *          *
50 REM  ************
60 PRINT"□":REM BILDSCHIRM LOESCHEN
70 TI$="000000"
80 PRINT TAB(16)"□";TI$
90 GOTO 80
100 REM PROGRAMMENDE: RUN/STOP DRUECKEN
READY.
```

Bei jedem Durchlauf der Schleife führt das Programm einen PRINT-Befehl aus. Durch das „Cursor-nach-oben"-Steuerzeichen wird der Cursor aber immer wieder eine Zeile zurückgesetzt, so daß die alte Ausgabe andauernd mit dem neuen Wert der Variablen TI$ überdeckt wird. Übrigens wird die Uhrzeit in den sechs Stellen von TI$ in folgendem Format angezeigt: HHMMSS.

Will man Cursor-Steuerzeichen anwenden, so muß man beachten, welche Position der Cursor hat, bevor das Programm zu dem Print-Befehl kommt, der die Steuerzeichen enthält. Auch die HOME- und die CLR/HOME-Funktionen erzeugen Steuerzeichen im Anführungszeichenmodus. Diese können im Programm gut dazu verwendet werden, immer eine gleiche Cursor-Ausgangsposition für weitere Steuerzeichen zu schaffen.

Bemerkung: Die in diesem Buch enthaltenen Listings enthalten keine Steuerzeichen. Aus reproduktionstechnischen Gründen haben wir uns an die ebenfalls verwendbare CHR$-Schreibweise gehalten. Zum Vergleich zeigen wir daher das obige Listing ein zweites Mal, aber in der CHR$-Schreibweise. „Cursor-nach-oben" bewirkt nun der Ausdruck CHR$(145). Eine vollständige Liste aller CHR$-Codes enthält das Handbuch zum Commodore 64 im Anhang F.

```
5 REM*** KAPITEL 3 PROGRAMM 2 ***
10 REM  ****************
20 REM  *               *
30 REM  * STOPPUHR II *
40 REM  *               *
50 REM  ****************
60 PRINTCHR$(147):REM BILDSCHIRM LOESCHE
N
70 TI$="000000"
80 PRINT TAB(16)CHR$(145);TI$
90 GOTO 80
100 REM PROGRAMMENDE: RUN/STOP DRUECKEN
READY.
```

Die Cursor-Steuerzeichen sind:

Taste	Zeichen
CLR/HOME	S
SHIFT CLR/HOME	♥
⇑ CRSR ⇓	Q
SHIFT ⇑ CRSR ⇓	●
⇐ CRSR ⇒	]
SHIFT ⇐ CRSR ⇒	▌

Im Anführungszeichenmodus kann ebenfalls die Farbe der auszugebenden Zeichen festgelegt werden. Die Farbsteuerzeichen können an jeder Stelle innerhalb des PRINT-Befehls eingebaut werden. Alle folgenden Zeichen in diesem Befehl, aber auch die in allen späteren PRINT-Befehlen, erscheinen dann in der so definierten Farbe, bis ein weiteres Farbsteuerzeichen auftritt. Die Farbsteuerzeichen werden im Anführungszeichenmodus mit den Ziffer-Tasten und der gleichzeitig gedrückten CRTL- bzw. Commodore-Taste (C=) erzeugt. Eine vollständige Tabelle der Farbsteuerzeichen enthält das Handbuch zum Commodore 64 auf Seite 57.

Hier ist ein kleines Programm, das ihre Wirkung demonstriert:

```
5 REM*** KAPITEL 3 PROGRAMM 3 ***
10 REM  ***********
20 REM  *         *
30 REM  * RAINBOW *
40 REM  *         *
50 REM  ***********
60 PRINT"■"
70 POKE53281,11:REM GRAUER HINTERGRUND
80 PRINT"■■■■■■■■■■■■■■■■■■■■■■■■■":REM CURSOR NAC
H UNTEN (15)
90 FORI=0TO10STEP2:REM  RAUF-
100 PRINT TAB(I)"■■■R■A■I■N■B■O■W"
110 NEXT
120 PRINT TAB(15)"■■■R■A■I■N■B■O■W"
130 FORI=20TO30STEP2:REM -UND RUNTER
140 PRINT TAB(I)"■R■A■I■N■B■O■W"
150 NEXT
160 END
```

(4) Der Commodore-Zeichensatz

Auf der Vorderseite der meisten Tasten der Commodore-Tastatur sind zwei kleine Quadrate abgebildet, die jedes ein spezielles Zeichen enthalten. Wenn man sie in Anführungszeichen setzt, können diese Zeichen auch gePRINTet werden. Das jeweils rechte der beiden Zeichen wird durch Druck der Taste selbst und SHIFT erzeugt. Das linke erhält man, wenn man die Taste zusammen mit C=, der Commodore-Taste, betätigt. Um z.B. unseren Freund „Alien" aus Kapitel wieder auf dem Bildschirm erscheinen zu lassen, müssen Sie diese Tasten drücken:

C=/A SHIFT/W SHIFT/W C=/S (SHIFT/RETURN)
SPACE SHIFT/K SHIFT/J

(5) Invers-Modus

Drückt man CRTL zusammen mit 9 im Anführungszeichenmodus, so erscheint ein inverses „R". Dieses Steuerzeichen schaltet den Inversmodus an (im Englischen „reverse mode", daher die Tastenaufschrift „RVS ON"). Alle Zeichen erscheinen nun als ihre eigenen Negative auf dem Bildschirm. Um den Zeichen ihr normales Aussehen zurückzugeben, betätigt man CRTL/0 bzw. erzeugt das entsprechende Steuerzeichen.

Durch den Einsatz dieses Modus kann man eine beliebige Gruppe von Zeichen innerhalb einer Zeichenkette isolieren und sie als Negativ ausgeben.

Das folgende Programm zeigt, wie man den Inversmodus benutzen kann, um blinkende Zeichen zu erzeugen.

```
5 REM*** KAPITEL 3 PROGRAMM 4 ***
10 REM  ***********
20 REM  *         *
30 REM  * NEGATIV *
40 REM  *         *
50 REM  ***********
60 PRINT"■"
70 PRINT"■■■■■■■■■":REM CURSOR NACH UNTEN
 (8)
80 PRINT TAB(10)"ER HAT EINEN ZIEMLICH"
90 PRINT TAB(15)"■■■NEGATIVEN"
100 FORI=1TO500:NEXT
110 PRINT TAB(15)"■■NEGATIVEN"
120 PRINT TAB(15)"■■CHARAKTER."
130 PRINT"■■■■■":REM CURSOR NACH OBEN (5
)
140 FORI=1TO500:NEXT
150 GOTO90
160 REM PROGRAMMENDE: RUN/STOP DRUECKEN
```

Wir haben nun zwei Methoden für das Verändern und Färben von Zeichen auf dem Bildschirm zur Verfügung. Beide Methoden haben Ihre Vor- und Nachteile, und es gibt keinen Grund, sie nicht beide in einem Programm zur Lösung verschiedener Probleme einzusetzen. Für welche Methode man sich nun im einzelnen Fall entscheidet, hängt stark von den persönlichen Erfahrungen und Vorlieben ab.

4

Die logischen Operatoren AND, OR und NOT

DAS DEZIMALSYSTEM

Bevor wir das Binärsystem betrachten, wollen wir zuerst unser eigenes Zahlensystem, das Dezimalsystem, unter die Lupe nehmen.

In einer Zahl, wie z.B. 1375, hat jede Stelle einen bestimmten Wert. In diesem Beispiel steht die „1" für „einmal Eintausend"; die „3" bedeutet „dreimal Einhundert" usw. Wir können die Zahl 1375 in vier Spalten anordnen, wobei jede Spalte eine bestimmte Überschrift hat:

T	H	Z	E
1	3	7	5

Die höchste Ziffer, die in jeder Spalte erscheinen kann, ist die „9".

Die Spaltenüberschriften basieren auf der Zahl 10.

EINER	1
ZEHNER	10
HUNDERTER	10*10 = 100
TAUSENDER	10*10*10 = 1000

DAS BINÄRSYSTEM

Basiert das Dezimalsystem (decem Lat.: zehn) auf der 10, so basiert das Binärsystem (bini Lat.: je zwei) auf der Zahl 2. Die Überschriften zu einer entsprechenden Tabelle würden folgendermaßen lauten:

EINER	1
ZWEIER	2
VIERER	2*2 = 4
ACHTER	2*2*2 = 8
SECHZEHNER	2*2*2*2 = 16

Die höchste Ziffer, die in einer Spalte erscheinen kann, ist die „1". Das Binärsystem benutzt also nur zwei Ziffern, die „0" und die „1".

Warum benutzen Computer dieses primitive Zahlensystem? Das ist ganz einfach: Der Computer kann in seinem Inneren nicht mehr als zwei Ziffern darstellen, „1„ wird durch eine elektrische Spannung wiedergegeben, „0" durch die Abwesenheit einer solchen.

UMRECHNUNG AUS DEM BINÄR- INS DEZIMALSYSTEM

Als Beispiel wollen wir die Binärzahl 11001011 in das Dezimalsystem umrechnen.

Als erstes müssen wir die Zahl in eine Tabelle mit den Werten der einzelnen Stellen eintragen.

128	64	32	16	8	4	2	1
1	1	0	0	1	0	1	1

Jetzt müssen wir nur noch die einzelnen Werte addieren:

1*128	=	128
1*64	=	64
1*8	=	8
1*2	=	2
1*1	=	1
SUMME	=	203

Die Binärzahl 11001011 lautet also im Dezimalsystem 203.

UMRECHNUNG AUS DEM DEZIMAL- INS BINÄRSYSTEM

Die einfachste Methode hierfür ist das mehrfache Dividieren durch Zwei. Als Beispiel nehmen wir das Ergebnis des vorhergehenden Abschnitts und versuchen die binäre Ausgangszahl zu errechnen.

203/2 = 101 Rest: 1
101/2 = 50 Rest: 1
50/2 = 25 Rest: 0
25/2 = 12 Rest: 1
12/2 = 6 Rest: 0
6/2 = 3 Rest: 0
3/2 = 1 Rest: 1
1/2 = 0 Rest: 1
0/2 = 0

Das Binäräquivalent von 203 erhält man, indem man die „Rest"-Spalte von unten nach oben liest.

DARSTELLUNGSWEISE VON LOGISCH VERKNÜPFBAREN ZAHLEN INNERHALB DES COMPUTERS

Jede dezimale Zahl wird intern als eine Reihe von 16 Bits (BIT = BInary digiT, engl.: binäre Stelle (einer Zahl)). Z.B. ist das Binärequivalent von 241 gleich 11110001. Für die Darstellung dieser Zahl werden zwar nur acht Bits benötigt, der Computer speichert 241 jedoch als 0000000011110001, wobei die ersten acht Bits mit Nullen gefüllt werden.

Auf diese Weise können Zahlen zwischen -32768 und +32767 gespeichert werden. Wird eine Zahl logisch verknüpft, die diese Grenzen überschreitet, dann erscheint eine Fehlermeldung.

Wie werden negative Zahlen gespeichert? Jetzt wird die Sache schon etwas komplizierter. Negative Zahlen werden im „Zweierkomplement" dargestellt. Hier eine kurze Erklärung der Arbeitsweise:

Nehmen wir die Zahl -241.

Wir wissen schon, daß 241 im 16-Bit-Format 0000000011110001 lautet. Um -241 zu erhalten, ändern wir erstens jede Ziffer in ihr Gegenteil (Komplement), d.h. wir schreiben für jede „1" eine „0" und für jede „0" eine „1". Zweitens addieren wir zum Ergebnis 1.

	Binär	Dezimal
	0000000011110001	+241
Komplement bilden	1111111100001110	
1 addieren	1111111100001111	−241

Die Zahl −241 wird also durch das Zweierkomplement der Zahl + 241 dargestellt. Wir müssen diese Prozesse verstehen, damit wir auch die Funktionen der logischen Operatoren verstehen können.

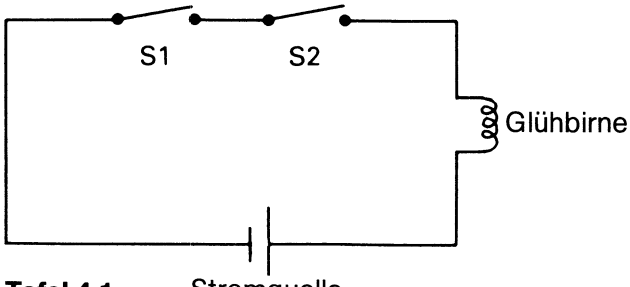
DER OPERATOR „AND"

Vielleicht haben Sie in einem Programm schon einmal einen Befehl wie diesen gesehen:

X=YAND18

Nehmen wir an, daß die Variable Y den Wert 35 besitzt, dann entspricht diese Zeile dem Befehl X=35AND18. Das ist nicht etwa gleichwertig mit X=35+18(=54). Das Wort AND bedeutet eine „logische" Verknüpfung. Sehen Sie sich nun den Schaltkreis auf Tafel 4.1 an. Er besteht aus einer Stromquelle (Batterie), einer Glühbirne, und zwei Schaltern, S1 und S2. In welchem Zustand müssen die Schalter sein, damit die Birne glüht? Die Wahrheitstafel zu diesem Schaltkreis hat folgendes Aussehen:

S1	S2	Birne
aus	aus	aus
aus	ein	aus
ein	aus	aus
ein	ein	ein



Tafel 4.1

Die Birne leuchtet nur, wenn S1 UND S2 eingeschaltet sind. Benutzen wir in der Wahrheitstafel statt „aus" die Ziffer „0" und statt „ein" die Ziffer „1", dann sieht sie so aus:

S1	S2	Birne
0	0	0
0	1	0
1	0	0
1	1	1

Entsprechend hat die Birne nur dann den Zustand „1", wenn auch S1 UND S2 sich in diesem Zustand befinden. Daher leitet sich der Name der AND-Verknüpfung ab.

Wir können die letzte Wahrheitstafel dazu verwenden, unser Beispiel zu berechnen:

35AND18

Zuerst müssen wir die Dezimalzahlen 35 und 18 in ihre Binäräquivalente umrechnen.

	Binär		Dezimal
	0000000000100011		35
AND	0000000000010010	AND	18
	0000000000000010		2

Betrachten wir die beiden Binärzahlen Spalte für Spalte und schreiben wir nur dann eine 1, wenn beide Bits im Zustand „1" sind, dann erhalten wir das oben gezeigte Ergebnis. Rechnen wir es wieder ins Dezimalsystem um, so erhalten wir 2.

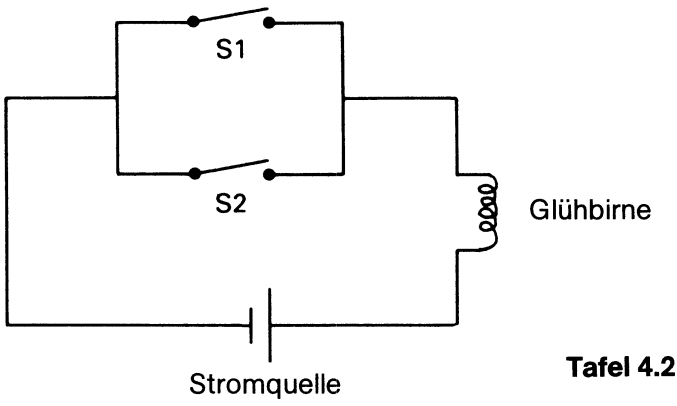
Der Befehl X=YAND18 mit Y=35 entspricht also X=2.

DER OPERATOR „OR"

Betrachten wir nun den Schaltkreis in Tafel 4.2. Die zugehörige Wahrheitstafel sieht so aus:

S1	S2	Birne
0	0	0
0	1	1
1	0	1
1	1	1

Wie leicht zu erkennen ist, leuchtet die Glühbirne, wenn entweder S1 oder S2 oder beide eingeschaltet sind.



Tafel 4.2

Hier ein Beispiel für eine logische OR (ODER) -Verknüpfung:
124OR73

	Binär		Dezimal
	0000000001111100		124
OR	0000000001001001	OR	73
	0000000001111101		125

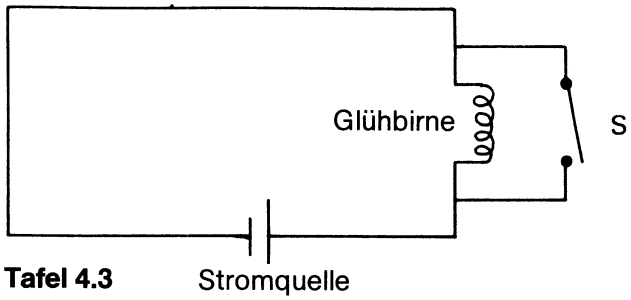
Um dieses Ergebnis zu erhalten, schreiben wir nur dann eine „0", wenn in einer Spalte jeweils beide Ziffern „0" sind; sonst schreiben wir eine „1".

Rechnen wir das Ergebnis in das Dezimalsystem um, so erhalten wir 125, also ist 124 OR 73 = 125.

DER OPERATOR „NOT"

Als letzte logische Verknüpfung wollen wir den NOT-Operator betrachten. Wir können uns seine Funktionsweise wieder an einem Schaltkreis klarmachen (siehe Tafel 4.3). Die Glühbirne leuchtet, wenn der Schalter offen ist, und bleibt dunkel, wenn er geschlossen ist. Die Wahrheitstabelle ist daher sehr kurz:

S	Birne
0	1
1	0



Tafel 4.3

Auf diese Weise bildet der NOT-Operator das „Komplement" des Operanden (um das Zweierkomplement zu erhalten, müssen wir noch 1 addieren; siehe auch den zugehörigen Abschnitt in diesem Kapitel).

Um aus einem ganzzahligen X sein Komplement zu errechnen, kann man folgenden Ausdruck verwenden:

```
X=-X-1
```

Beispiele für die Wirkung des Not-Operators:

```
NOT253=-254
NOT16 =-17
NOT-89=88
```

usw.

WEITERE ANWENDUNGEN VON „AND" und "OR"

Eine den meisten Programmierern besser bekannte Anwendung von „AND" und „OR" ist wohl die Verknüpfung von Bedingungen im IF-THEN-Befehl. Betrachten Sie das folgende Programm und versuchen Sie, bevor Sie es laufen lassen, die Ausgabe vorherzusagen.

```
10 CHR$(147):REM BILDSCHIRM LOESCHEN
20 FORI=1TO8
30 FORJ=1TO5
40 IFI=3ANDJ=2THENPRINTI,J
50 NEXT:NEXT
60 END
```

Das Programm führt die beiden geschachtelten Schleifen aus, PRINTet aber nur die Werte der Laufvariablen I und J, wenn I=3 und J=2. Die Ausgabe des Programms sieht also so aus:

```
3       2
```

Dies scheint eine natürliche Verwendung der AND-Verknüpfung zu sein. OR kann auf ähnliche Weise eingesetzt werden, alleinstehend oder in Verbindung mit anderen Operatoren. Verändern Sie Zeile 40 zu.

```
40 IFI=3ANDJ=2ORJ=4THEN PRINTI,J
```

Mit dieser Zeile lautet die Ausgabe:

```
1       4
2       4
3       2
3       4
4       4
5       4
6       4
7       4
8       4
```

Der Computer führt als erstes die AND-Operation aus. I und J werden jedoch nur dann ausgedruckt, wenn I=3 UND J=2 oder wenn J=4.

Wir können die Prioritätsreihenfolge durch die Verwendung von Klammern ändern. Verändern Sie Zeile 40 zu:

```
40 IFI=3AND(J=2ORJ=4)THEN PRINTI,J
```

Die Ausgabe lautet diesmal:

```
3       2
3       4
```

Durch das Einklammern des OR-Ausdrucks erreichen wir, daß diese Operation zuerst ausgeführt wird.

Die Prioritätsreihenfolge des CBM 64 sieht im Normalfall vor, daß bei einer Kombination von AND und OR die AND-Verknüpfung zuerst vorgenommen wird. Diese Reihenfolge kann nur durch Klammern geändert werden.

In diesem Kapitel haben wir einige wichtige Aspekte der internen Arbeitsweise des Computers kennengelernt. Dieses Wissen wird in verschiedenen Abschnitten des Buches weiterverwendet.

5

Selbstdefinierte Zeichen

Obwohl sich auch mit dem in Ihrem Commodore 64 eingebauten Zeichensatz passable Ergebnisse erzielen lassen, werden Sie eines Tages doch den Wunsch haben, eigene Zeichen zu definieren und zu benutzen. Dieses Unterfangen ist nicht so schwierig, wie es sich im ersten Moment anhört. Dieses Kapitel zeigt Ihnen alle Schritte, die für die Konstruktion von selbstdefinierten Zeichen wichtig sind.

DAS ZEICHENSATZ-KOPIERPROGRAMM

Unglücklicherweise ist der Standard-Zeichensatz im ROM des Computers gespeichert, das heißt, wir können die Zeichen nicht ohne weiteres verändern. Bevor wir also mit dem eigentlichen Definieren beginnen, müssen wir den Zeichensatz vom ROM ins RAM kopieren. Betrachten Sie hierzu den Anhang E des Commodore-64-Handbuchs. Für unsere späteren Zwecke werden wir wahrscheinlich die Buchstaben, die Ziffern von 0 bis 9 und einige Interpunktionszeichen benötigen. Es wäre aber überflüssig, den gesamten Zeichensatz mit allen Grafikzeichen zu kopieren, da wir dann begännen, den Speicherplatz zu verbrauchen, den wir eigentlich dem Programm zugedacht hatten, das die selbstdefinierten Zeichen benutzt. Letztlich reichen die ersten 64 Zeichen (also die Zeichen 0 bis 63) völlig aus.

Das folgende Zeichensatz-Kopierprogramm arbeitet genau gemäß diesen Anforderungen. Es ist ein BASIC-Lader für ein MASCHINENSPRACHE-Programm. Wollen Sie selbstdefinierte Zeichen in Ihren Programmen verwenden, dann ist es sinnvoll, dieses Kopierprogramm als Unterprogramm einzubauen. Da es wie gesagt eine Maschinenroutine für den Kopiervorgang enthält, führt es seine Aufgabe um einiges schneller aus als jedes entsprechende BASIC-Programm.

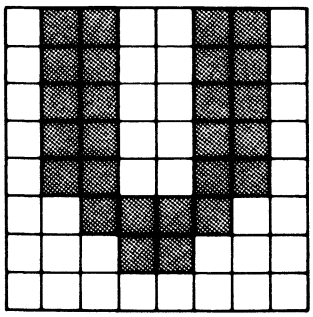
Geben Sie den Zeichensatz-Kopierer ein und SAVEn Sie ihn.

```
1000 REM  ******************
1010 REM  **                **
1020 REM  ** ZEICHENSATZ-  **
1040 REM  **   KOPIERER     **
1050 REM  **                **
1060 REM  ** KOPIERT VON    **
1070 REM  ** ROM NACH RAM   **
1080 REM  **                **
1090 REM  ******************
1100 POKE52,48:POKE56,48:REM SPEICHEROBE
RGRENZE HERABSETZEN
1110 FORI=828TO950:REM KASSETTENPUFFER
1120 READJ          :REM --------
1130 AA=AA+J        :REM BASIC-
1140 POKEI,J        :REM LADER
1150 NEXT           :REM --------
1160 READJ          :REM PRUEFSUMME
1170 IFAA<>JTHENPRINT"FEHLERIN DEN DATEN
":END
1180 SYS(828)       :REM AUFRUFEN
1190 POKE53272,28 :REM POINTER AUF RAM S
ETZEN
1200 REM DATEN FUER DAS MASCHINENPROGRAM
M
1210 DATA 169,0,141,14,220,169,51,141
1220 DATA 1,0,162,0,189,0,208,157,0,48
1230 DATA 189,0,209,157,0,49,189,0,210
1240 DATA 157,0,50,189,0,211,157,0,51
1250 DATA 189,0,212,157,0,52,189,0,213
1260 DATA 157,0,53,189,0,214,157,0,54
1270 DATA 189,0,215,157,0,55,232,224
1280 DATA 255,208,203,173,255,208,141
1290 DATA 255,48,173,255,209,141,255
1300 DATA 49,173,255,210,141,255,50
1310 DATA 173,255,211,141,255,51,173
1320 DATA 255,212,141,255,52,173,255
1330 DATA 213,141,255,58,173,255,214
1340 DATA 141,255,54,169,55,141,1,0
1350 DATA 169,1,141,14,220,169,28,141
1360 DATA 24,208,96
1370 REM PRUEFSUMME
1380 DATA 16246
1400 RETURN
```

DEFINITION VON EIGENEN ZEICHEN

Wenn Sie sich ein Zeichen auf dem Bildschirm genauer ansehen, werden Sie feststellen, daß es aus einem Muster von winzigen Punkten besteht. Diese Punkte nennt man PIXELS (von picture element, engl.: kleinste Einheit eines Bildes). Jedes Zeichen besteht aus einem Muster von acht mal acht Pixels. Einige Pixels in diesem Feld sind eingeschaltet und für den Betrachter sichtbar, die übrigen, ausgeschalteten, haben Hintergrundfarbe.

Tafel 5.1 zeigt, wie ein großes „V" aufgebaut ist. Beachten Sie, daß jede Reihe von acht Pixels auch als binäre Zahl betrachtet werden kann. Ein Bit im Zustand „1" steht für ein eingeschaltetes Pixel, ein Bit im Zustand „0" steht für ein ausgeschaltetes. Jede binäre Zahl besteht aus acht Bits. Also einem BYTE. Sie kann auch ins Dezimalsystem umgerechnet werden.



BINÄR								DEZIMAL		
128	**64**	**32**	**16**	**8**	**4**	**2**	**1**			
0	1	1	0	0	1	1	0	1	0	2
0	1	1	0	0	1	1	0	1	0	2
0	1	1	0	0	1	1	0	1	0	2
0	1	1	0	0	1	1	0	1	0	2
0	1	1	0	0	1	1	0	1	0	2
0	0	1	1	1	1	0	0		6	0
0	0	0	1	1	0	0	0		2	4
0	0	0	0	0	0	0	0			0

Tafel 5.1

Nachdem das Kopierprogramm gelaufen ist, befindet sich der aktuelle Zeichensatz im RAM ab der Speicherstelle 12288. Jede Speicherstelle enthält ein Byte. Wir benötigen daher acht Bytes, um die Daten für ein Zeichen zu speichern. Die acht Bytes, die das „ @ " definieren, stehen also auf folgenden Speicherstellen:

12288
12289
12290
12291
12292
12293
12294
12295

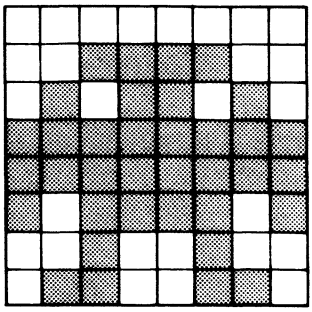
Die Daten zum nächsten Zeichen, „A", befinden sich dann auf 12296–12303 usw. Die Beziehung zwischen dem Bildschirmcode eines Zeichens (siehe Anhang E des Commodore-64-Handbuchs) und der zugehörigen ersten Speicherstelle dieses Zeichens im RAM ist sehr einfach:

Erste Speicherstelle im RAM = 12288+8*Bildschirmcode

Doch jetzt wollen wir endlich ein eigenes Zeichen definieren. Als erstes müssen wir das Kopierprogramm eingeben oder laden. Dann können wir den Rest des Programms darum herum bauen. „Zeichensatzkopierer" sollte als Unterprogramm auf Zeile 1000 ff. verwendet werden. Die erste Zeile unseres Programms muß also lauten:

```
10 GOSUB 1000:REM KOPIERPROGRAMM
```

Jetzt müssen wir unser Zeichen entwerfen. Nehmen wir an, wir wollen ein kleines Weltraumwesen auf dem Bildschirm erscheinen lassen. Zuerst zeichnen wir es auf einem acht mal acht Kästchen großen Feld, z.B. auf kariertem Papier. Diesen Entwurf übertragen wir in die binäre Schreibweise und dann in dezimale Zahlen (siehe Tafel 5.2). Sie können den in Tafel 5.2 gezeigten Vorschlag übernehmen oder selbst ein schöneres Zeichen entwerfen. Das Ergebnis besteht in jedem Fall aus acht dezimalen Zahlen, die die Definition unseres Zeichens darstellen. Wir müssen diese Definition jetzt nur noch in bestimmte Speicherstellen unserer Zeichensatzkopie POKEn. Damit überschreiben wir die Definition eines Zeichens, das wir aus dem ROM abgeschrieben haben. Es ist daher am besten, wenn wir ein Zeichen überschreiben, das im weiteren Verlauf des Programms nicht verwendet wird. Das Kaufmanns-Und „&" erfüllt in unserem Fall diese Anforderung.



BINÄR								DEZIMAL
128	**64**	**32**	**16**	**8**	**4**	**2**	**1**	
0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0	60
0	1	0	1	1	0	1	0	90
1	1	1	1	1	1	1	1	255
1	1	1	1	1	1	1	1	255
1	0	1	1	1	1	0	1	189
0	0	1	0	0	1	0	0	36
0	1	1	0	0	1	1	0	102

Tafel 5.2

Der Bildschirmcode des Kaufmanns-Und ist 38. Um die Anfangsadresse seiner Definition zu errechnen, benutzen wir die oben beschriebene Beziehung:

Erste Speicherzelle im RAM = 12288+8*38
= 12592

Es empfiehlt sich, die acht Werte der Definition in einer DATA-Zeile abzulegen, sie dann im Programm per READ auszulesen und in die Speicherstellen von 12592 bis 12599 zu POKEn. Die folgende Programmzeile führt dies aus:

```
20 FORI=12592TO12599:READA:POKEI,A:NEXT
```

Die Data-Zeile legen wir ans Ende des Programms, z.B. auf Zeile 10000. Das gesamte Listing sieht dann so aus:

```
1 REM **********************
2 REM *                      *
3 REM * ZEICHENDEFINITION *
4 REM *                      *
5 REM **********************
6 REM
10 GOSUB1000:REM ZEICHENSATZKOPIERER
20 FORI=12592TO12599:READ A:POKE I,A:NEX
T
30 END
1000 REM ZEICHSATZKOPIERER
1200 :
1400 RETURN
10000 REM DATEN FUER ZEICHENDEFINITION
10010 DATA 0,60,90,255,255,189,36,102
```

Lassen Sie das Programm laufen und versuchen Sie dann, ein Kaufmanns-Und zu erzeugen (SHIFT/6). Wie das ursprüngliche Zeichen können Sie auch unser neues Zeichen in PRINT-Befehlen verwenden oder es auf den Bildschirm POKEn und färben; der Bildschirmcode ist immer noch 38. Sie können natürlich beliebig viele der 64 kopierten Zeichen umdefinieren; achten Sie aber darauf, daß Sie das Zeichen später nicht in seiner normalen Form benötigen.

Wie sie im Beispiel gesehen haben, ist ein einzelnes Zeichen recht klein. Es ist aber möglich, mehrere Zeichen zu einem großen zusammenzusetzen (siehe auch „Invaders", Kapitel 2).

DER MEHRFARBENMODUS

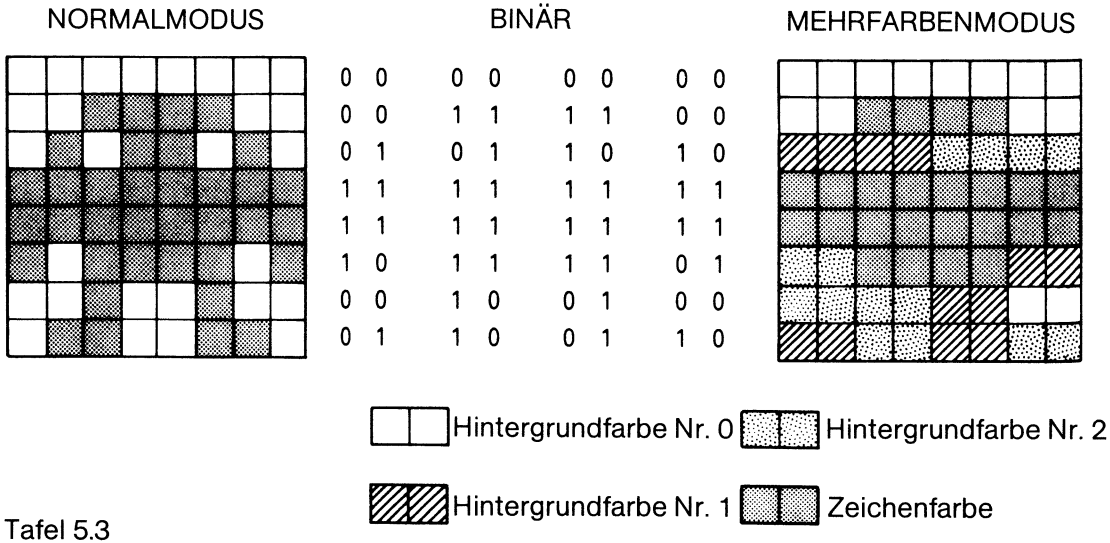
Der Mehrfarbenmodus des Commodore 64 erlaubt Ihnen, bis zu vier Farben innerhalb eines Zeichens (eines 8*8-Feldes) zu benutzen. Dieser Modus muß jedoch angeschaltet werden, bevor er genutzt werden kann. Das erreicht man, indem man das fünfte Bit der Speicherstelle 53270 auf 1 setzt. Man könnte, um dieses Bit zu setzen, einfach 16 in die Speicherstelle POKEn; die übrigen Bits dieses Registers haben aber ebenfalls bestimmte Funktionen und dürfen nicht beeinflußt werden. Mit dem POKEn von 16 würden alle Bits außer dem fünften gelöscht. Um Bit Nr. 5 zu setzen, ohne die anderen zu beeinflussen, kann der folgende POKE-Befehl verwendet werden (das System, nach dem hier vorgegangen wird, erklärt ausführlich Kapitel 6):

```
POKE53270,PEEK(53270)OR16
```

Im Mehrfarbenmodus interpretiert der Computer die Bitmuster der Zeichen auf eine andere Weise. Anstelle der getrennten Betrachtung jedes einzelnen Bits teilt er die Bits horizontal in Paare ein. Da jedes Bit zwei Zustände hat („0" und „1"), ergeben sich vier verschiedene Möglichkeiten für das Aussehen eines Paars: 00, 01, 10, 11. Dieses Bitmuster korrespondieren mit vier Farben, die festgelegt werden können, indem ein Farbcode in die entsprechende Speicherstelle gePOKEt wird. Beide Bits erscheinen auf dem Bildschirm in der Farbe, die sie durch Ihr Muster anzeigen. Tabelle 5.1 zeigt die mit den Bitmustern korrespondierenden Farben und die Speicherstellen, in die man den gewünschten Farbcode POKEn muß. Da nun nur noch jeweils zwei Pixels auf einmal angesteuert werden können, reduziert sich die horizontale Auflösung der Grafik auf ihren halben Wert. Um die Funktion des Mehrfarbenmodus zu verdeutlichen, betrachten wir nochmals unser Weltraumwesen. Wenn der Mehrfarbenmodus eingeschaltet ist, wird das Bitmuster, das das Wesen definiert, anders interpretiert (siehe Tafel 5.3). Sie sehen, daß dasselbe Bitmuster im Mehrfarbenmodus ein Bild erzeugt, das mit dem im Normalmodus erzeugten kaum Ähnlichkeit hat. Es ist daher wichtig, bei der Definition von Zeichen darauf zu achten, ob das jeweilige Zeichen im Normal- oder im Mehrfarbenmodus erscheinen soll.

Tabelle 5.1

Bitpaar	Farbe	Speicherstelle
00	Hintergrundfarbe Nr. 0 (Bildschirmfarbe)	53281
01	Hintergrundfarbe Nr. 1	53282
10	Hintergrundfarbe Nr. 2	53283
11	Normale Zeichenfarbe	Farbspeicher


NORMALMODUS
BINÄR
MEHRFARBENMODUS
0 0 0 0 0 0 0 0
0 0 1 1 1 1 0 0
0 1 0 1 1 0 1 0
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1
1 0 1 1 1 1 0 1
0 0 1 0 0 1 0 0
0 1 1 0 0 1 1 0
Hintergrundfarbe Nr. 0
Hintergrundfarbe Nr. 2
Hintergrundfarbe Nr. 1
Zeichenfarbe

Tafel 5.3

Im Normalmodus, wenn also der Mehrfarbenmodus ausgeschaltet ist, können Sie jeden der 16 Farbcodes mittels der Zahlen 0 bis 15 in den Farbspeicher POKEn. Im Mehrfarbenmodus entscheidet diese Farbinformation aber auch darüber, ob das Zeichen mehrfarbig oder normal erscheint. Der Schalter hierfür ist jeweils das vierte Bit der Farbspeicherstelle. Ist das Bit gesetzt, so erscheint das Zeichen mehrfarbig; ist es nicht gesetzt, so erscheint es normal auf dem Bildschirm. Da der Farbcode jeweils vier Bits (ein sogenanntes Nibble) einnimmt, bedeutet das eben Gesagte, daß, wenn der Mehrfarbenmodus EINgeschaltet ist, die ersten drei Bits des Farbcodes die Farbe des Zeichens kontrollieren und das vierte Bit entscheidet, ob das Zeichen normal oder mehrfarbig erscheint. Tabelle 5.2 verdeutlicht diesen Sachverhalt. Wie aus dieser Tabelle zu erkennen ist, bewirken die Farbcodes 0 bis 7 bei eingeschaltetem Mehrfarbenmodus, daß das Zeichen normal in einer der acht Farben auf dem Bildschirm erscheint. Die Farbcodes von 8 bis 15 entsprechen den Farben 0 bis 7, mit dem Unterschied, daß das Zeichen mehrfarbig ausgegeben wird.

Wenn Sie immer noch das Zeichendefinitionsprogramm im Speicher haben, dann testen Sie doch diese Veränderungen aus:

Ändern Sie Zeile 30 zu:

```
30 POKE53270,PEEK(53270)OR16
```

Hiermit schalten Sie den Mehrfarbenmodus ein. Fügen Sie folgende Zeilen hinzu:

```
 40 POKE53281,0:REM HINTERGRUNDFARBE NR.0 AUF SCHWARZ
 50 POKE53282,1:REM HINTERGRUNDFARBE NR.1 AUF WEISS
 60 POKE53283,5:REM HINTERGRUNDFARBE NR.2 AUF GRUEN
 70 PRINTCHR$(147):REM BILDSCHIRM LOESCHEN
 80 FORI=1024TO2023STEP4
 90 POKEI,38:POKEI+2,38:REM "&" AUF BILDSCHIRM POKEN
100 POKE54272+I,10:POKE54272+I+2,2
110 NEXT
120 END
```

Die Schleife in den Zeilen 80 bis 110 POKEt unser Weltraumwesen auf jeden zweiten Speicherplatz des Bildschirmspeichers. Beachten Sie, daß beim Einfärben der Zeichen in Zeile 100 jedes erste mehrfarbig mit roter Zeichenfarbe erscheint. Das zweite Wesen wird aber normal und rot gefärbt ausgegeben.

Tabelle 5.2

Farb-code	Nibble	Mehrfarbenmodus EIN		Mehrfarbenmodus AUS
		Ausgabe	Farbe	
0	0000	normal	Schwarz	Schwarz
1	0001	normal	Weiß	Weiß
2	0010	normal	Rot	Rot
3	0011	normal	Türkis	Türkis
4	0100	normal	Violett	Violett
5	0101	normal	Grün	Grün
6	0110	normal	Blau	Blau
7	0111	normal	Gelb	Gelb
8	1000	mehrfarbig	Schwarz	Orange
9	1001	mehrfarbig	Weiß	Braun
10	1010	mehrfarbig	Rot	Hellrot
11	1011	mehrfarbig	Türkis	Grau 1
12	1100	mehrfarbig	Violett	Grau 2
13	1101	mehrfarbig	Grün	Hellgrün
14	1110	mehrfarbig	Blau	Hellblau
15	1111	mehrfarbig	Gelb	Grau 3

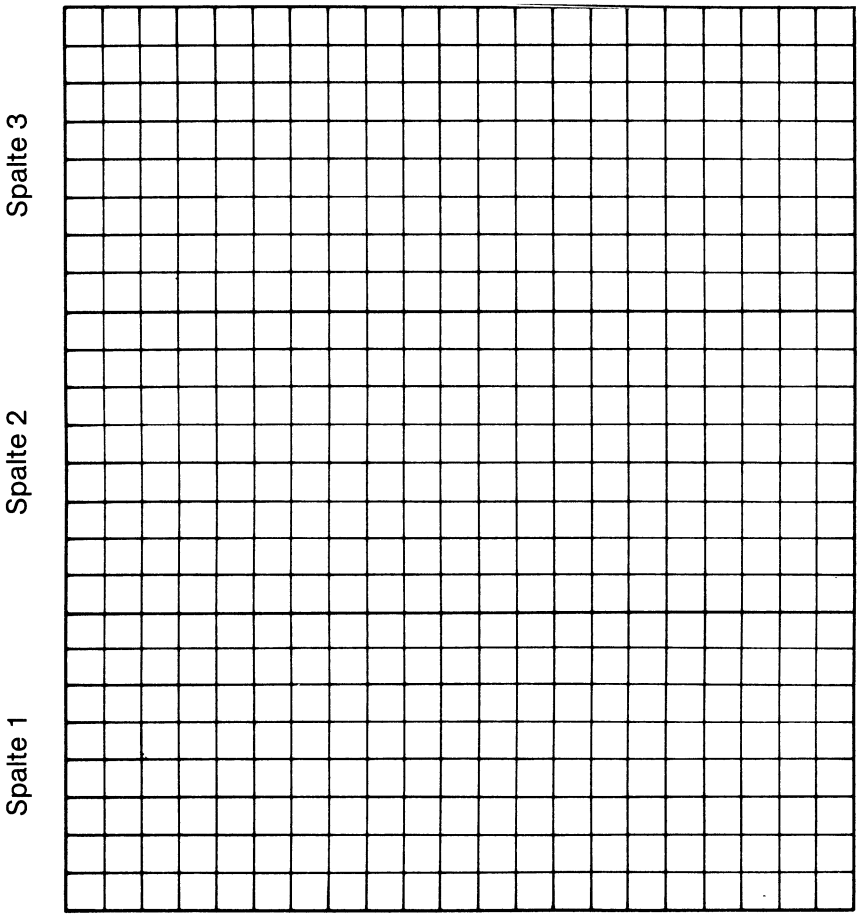
6

Sprites

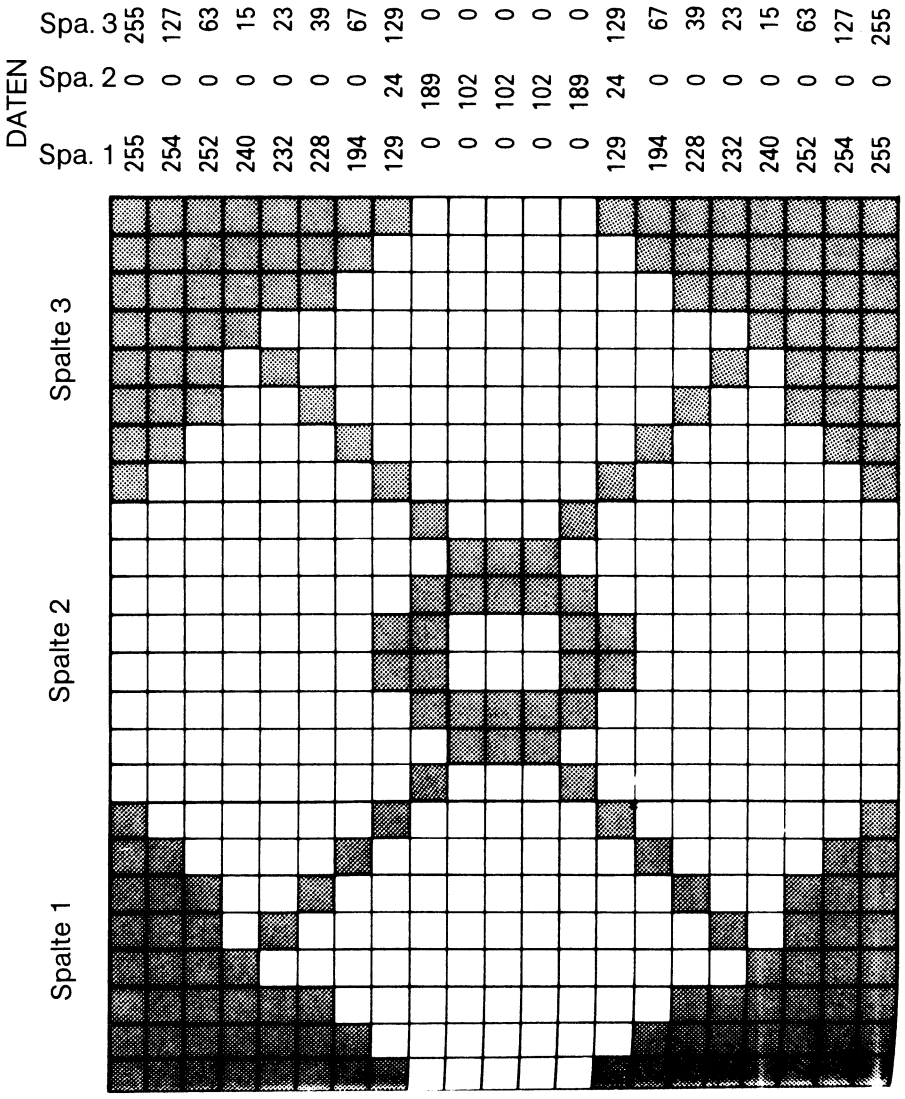
Die Sprites stellen eine besondere Grafikfähigkeit des Commodore 64 dar. Sie vereinfachen die Programmierung von Action-Spielen enorm. Sprites bestehen aus 504 Pixels, die in einem Rechteck mit 24 Pixels Breite und 21 Pixels Höhe angeordnet sind. Die Bewegung einzelner Zeichen sieht auf dem Bildschirm sehr ruckartig aus, da sie jeweils einen Block von acht Pixels überspringen, wenn sie von einer Bildschirmposition zur anderen bewegt werden. Bei der Verwendung von Sprites entfällt dieses Problem, weil sie um je ein Pixel bewegt werden können. Weitere Eigenschaften der Sprites sind ihre Prioritäten und die Kollisionserkennung. Das erste bedeutet, daß sich Sprites vor oder hinter anderen Sprites bewegen können. Das zweite heißt, daß man auf einfache Weise feststellen kann, ob ein Sprite mit einem anderen Sprite oder mit Hintergrundzeichen kollidiert. Doch dazu später mehr. Beginnen wir erst einmal mit der Definition eines Sprites.

DIE DEFINITION EINES SPRITES

Der Entwurf eines Sprites ähnelt dem eines selbstdefinierten Zeichens, nur daß wir nicht mehr an das Acht-mal-acht-Gitter gebunden sind. Wie schon erwähnt, ist ein Sprite, wie in Tafel 6.1 zu sehen, aus dem rechteckigen Gitter, 24 mal 21 Pixels, aufgebaut. Beachten Sie, daß das Rechteck in drei Spalten, die je acht Pixels breit sind, eingeteilt ist. Als erstes müssen Sie die Punkte einfärben, die im fertigen Sprite sichtbar sein sollen. Jeder nicht eingefärbte Punkt wird später transparent sein und den Hintergrund durchscheinen lassen. Tafel 6.2 zeigt einen meiner Entwürfe.



Tafel 6.1


DATEN
Spa. 1
Spa. 2
Spa. 3
255 0 255
254 0 127
252 0 63
240 0 15
232 0 23
228 0 39
194 0 67
129 24 129
0 189 0
0 102 0
0 102 0
0 102 0
0 189 0
129 24 129
194 0 67
228 0 39
232 0 23
240 0 15
252 0 63
254 0 127
255 0 255
Spalte 1
Spalte 2
Spalte 3

Tafel 6.2

Die Informationen des Sprite-Entwurfs müssen in Zahlen umgewandelt werden, die vom Computer eingelesen werden können. Das geschieht auf ähnliche Weise wie bei den selbstdefinierten Zeichen. Nehmen wir z.B. die oberste Zeile des Sprites: Sie entsteht aus 24 Pixels, die in drei Gruppen zu je acht Pixels eingeteilt ist. Da jedes Pixels durch ein Bit repräsentiert werden kann, besteht eine Gruppe aus genau einem Byte. Jedes Byte kann in eine Dezimalzahl im Bereich 0 bis 255 umgewandelt werden. Bei meinem Entwurf ist das erste Byte der obersten Zeile 11111111 oder 255 im Dezimalsystem. Das zweite Byte ist 00000000; das ist dezimal 0. Das dritte Byte ist wieder 11111111, also 255 dezimal. Beginnen wir jetzt mit der zweiten Zeile: Das erste Byte ist 11111110. Im Dezimalsystem ist das 254. Wenn Sie Ihren Weg auf diese Weise durch alle 21 Zeilen fortsetzen, haben Sie am Ende 21 mal 3 Dezimalzahlen. Insgesamt sind das 63 Daten. Sehen wir uns jetzt an, wie aus diesen Daten ein kleines Bild auf dem Bildschirm aufgebaut werden kann.

EIN EINFACHES SPRITE-PROGRAMM

Wenn Sie Ihr Sprite entworfen und die Sprite-Daten in Dezimalzahlen umgewandelt haben, müssen noch einige andere wichtige Schritte durchgeführt werden, bevor Ihr Sprite auf dem Schirm erscheint. Das sind:

1. Speicherzeiger setzen.
2. Daten einlesen und in den speziellen Speicherbereich POKEn.
3. Sprite färben.
4. Sprite auf dem Schirm positionieren.
5. Sprite anschalten.

Jeder Schritt wird später ganz detailliert erklärt, also „Ruhe bewahren".

Es gibt acht Sprites auf dem Commodore 64; die Sprites sind von 0 bis 7 durchnumeriert. Für unsere Beispiele wollen wir Sprite 0 nehmen.

Die Speicherstelle 2040 ist der Speicherzeiger für Sprite 0. Dieser Zeiger ist auf die Speicherstelle gerichtet, mit der die Daten des Entwurfs beginnen. In unserem Fall soll der Speicherzeiger den Wert 192 bekommen. Um Schritt Eins durchzuführen, müssen wir auf 2040 den Wert 192 POKEn.

Wir wollen die 63 Sprite-Daten im Speicherbereich von 12288 bis 12350 ablegen. Alles, was wir dazu tun müssen, ist die Daten der Reihe nach einlesen und auf die entsprechenden Adressen POKEn.

Die Farbe eines jeden Sprites wird in einer gesonderten Adresse abgelegt (53287 bis 53294 für die Sprites von 0 bis 7). Ein Sprite wird durch POKEn einer Zahl zwischen 0 bis 15 in eine dieser Adressen gefärbt. Oftmals ist es einfacher, sich die Adressen des VIC-Chips zu merken, wenn man sie zu seiner Basisadresse 53248 in Beziehung setzt. Wenn wir dieser Zahl den Namen „V" geben, dann liegt die Farbe für Sprite 0 auf V+39.

Die Position des Sprites 0 wird durch seine horizontale X- und durch seine vertikale Y-Koordinate bestimmt. Diese beiden Koordinaten legen die Position der linken oberen Ecke des Sprites fest. Die X-Koordinate liegt auf V (53248), die Y-Koordinate auf V+1 (53249).

Wenn alle diese Parameter eingestellt sind, bleibt nur noch das Anschalten des Sprites übrig. Das geschieht mit POKE V+21,1.

Fassen wir alle Schritte in einem Programm zusammen. Hinzugefügt wurden die DATA-Zeilen und ein Befehl zum Bildschirmlöschen. Das Programm zeigt unser Sprite in schönem Schwarz. Beachten Sie, daß V zu Beginn des Programms auf 53248 gesetzt werden muß, wenn sich Adressen auf „V+Zahl" beziehen.

```
1 REM *** KAPITEL 6 PROGRAMM 1 ***
2 REM  ************
3 REM  ************
4 REM  **          **
5 REM  ** SPRITE **
6 REM  **  DEMO  **
7 REM  **  NR.1  **
8 REM  **          **
9 REM  ************
10 REM ************
11 PRINTCHR$(147):REM BILDSCHIRM LOESCHE
N
20 REM SPRITE-POINTER AUF SPEICHERBEREIC
H SETZEN:V IST DIE BASISADRESSE DES VIC
30 POKE2040,192:V=53248
40 REM SPRITE-DATEN LESEN
50 FORI=12288TO12288+62:READA:POKEI,A
55 NEXT
60 REM SPRITE SCHWARZ FAERBEN
70 POKEV+39,0
80 REM SPRITE POSITIONIEREN
90 POKEV,100:POKEV+1,100
100 REM SPRITE ANSCHALTEN
110 POKEV+21,1
190 REM SPRITE-DATEN
200 DATA255,0,255,254,0,127,252,0,63
210 DATA240,0,15,232,0,23,228,0,39
220 DATA194,0,67,129,24,129,0,189,0
230 DATA0,102,0
240 DATA0,102,0
250 DATA0,102,0
260 DATA0,189,0,129,24,129,194,0,67
270 DATA228,0,39,232,0,23,240,0,15
280 DATA252,0,63,254,0,127,255,0,255
```

Geben Sie das Programm ein und starten Sie es. Wenn das Sprite anders aussieht als erwartet, prüfen Sie die DATAs sorgfältig, da hier leicht Fehler gemacht werden. Da Sie das Sprite schon auf dem Schirm haben, wollen wir einige weitere Möglichkeiten der Sprites ausprobieren.

Geben Sie diesen Befehl im Direktmodus ein:

```
POKEV+29,1
```

Wir können unser Sprite mit Hilfe eines POKEs auf seine doppelte Breite vergrößern. POKEV+29,0 macht diese Vergrößerung rückgängig. Probieren Sie danach folgendes aus:

```
POKEV+23,1
```

Dieser Befehl vergrößert das Sprite in vertikaler Richtung. Natürlich können Sie es auch in beide Richtungen gleichzeitig ausdehnen. POKEV+29,1 zeigt Ihnen das.

Wir können dem Sprite auch eine neue Farbe geben, indem wir einen anderen Wert zwischen 0 und 15 auf V+39 POKEn. Versuchen Sie es mit POKEV+39,7: Das Sprite wird gelb.

Auch das Bewegen des Sprites ist einfach. POKEV,200 bewirkt, daß daß Sprite 100 Pixels nach rechts springt. Um eine gleichmäßige Bewegung über den Bildschirm zu erreichen, geben Sie diese Zeile im Direktmodus ein:

```
FORX=50TO250:POKEV,X:NEXT
```

Sehen Sie genau hin, denn sobald Sie RETURN drücken, wird sich das Sprite gleichmäßig aber schnell über den Schirm bewegen. Sollten Sie ein geringeres Tempo bevorzugen, bauen Sie in die Hauptschleife eine Verzögerungsschleife ein:

```
FORJ=1TO500:NEXT
```

Die Geschwindigkeit wird jetzt durch die Länge der Verzögerungsschleife bestimmt. Ersetzen Sie die 500 durch einige andere Zahlen.

Das gleiche können wir natürlich auch in der Vertikalen machen. Versuchen Sie:

```
FORY=50TO150:POKEV+1,Y:FORJ=1TO500:NEXT:NEXT
```

Wenn Sie die 150 der „Y"-Schleife zu 250 ändern und erneut RETURN drücken, können Sie einen interessanten Effekt beobachten: Wenn das Sprite den unteren Rand des Schirms erreicht, verschwindet es. Das Sprite ist nicht etwa verloren, es ist nur aus dem sichtbaren Teil des Bildschirms verschwunden. Es wird wieder erscheinen, wenn Sie

```
FORY=250TO50STEP-1:POKEV+1,Y:FORJ=1TO500:NEXT:NEXT
```

eingeben. Dieser Effekt kann in vielen Spielen sehr nützlich sein.

Wir können mit POKEV+21,0 das Sprite ausschalten, ohne dabei seine Definition zu verlieren. Durch POKEV+21,1 wird es wieder angeschaltet.

Es gibt in der Tat nur zwei Wege, sich von einem Sprite zu befreien: entweder den Computer ausschalten oder gleichzeitig RUN/STOP und RESTORE drücken.

Wir haben uns bis jetzt einen Überblick über die Sprites verschafft. Nun wollen wir uns die einzelnen Funktionen detailliert ansehen.

DIE SPEICHERZEIGER

Wie bereits erwähnt verfügt der Commodore 64 über acht Sprites. Jedes von ihnen hat eine Adresse, die als Speicherzeiger dient. Die folgende Tafel gibt eine Übersicht:

Sprite Nr.	0	1	2	3	4	5	6	7
Adresse	2040	2041	2042	2043	2044	2045	2046	2047
Speicher-zeiger	192	193	194	195	196	197	198	199

Jeder Speicherzeiger ist auf den Speicherbereich gerichtet, in dem die Daten für die Form des Sprites liegen. Man kann mit diesen Zeigern einige interessante Effekte erzielen. Wenn auf dem Speicherzeiger 0 mit der Adresse 2040 der Wert 192 liegt, verweist er auf die 63 Bytes, die an der Stelle 12288 im Speicher beginnen. Ist der Zeiger jedoch auf 193 eingestellt, dann ist er auf die 63 Bytes gerichtet, die bei 12352 beginnen. Wenn also auf Adresse 2040 der Wert 192 liegt, bezieht Sprite 0 seine Daten aus dem Bereich von 12288 aufwärts. Sollte es innerhalb eines Programms erforderlich sein, die Form von Sprite 0 zu ändern (z.B. nach einer Kollision), so kann dies einfach dadurch erreicht werden, daß der Speicherzeiger mit der Adresse 2040 auf 193 umgeschaltet wird. Das Sprite 0 bezieht jetzt die Daten, die Sie zuvor dort abgelegt haben, aus dem Bereich 12352 bis 12414.

Die Bereiche, die normalerweise zum Speichern von Sprite-Daten verwendet werden, sind:

Speicher-bereich	12288 bis 12350	12352 bis 12414	12416 bis 12478	12480 bis 12542	12544 bis 12606	12608 bis 12670	12672 bis 12734	12736 bis 12798

Neben diesen Bereichen können Sie auch den Kasettenpuffer verwenden, um einige Sprite-Daten zu speichern. Die entsprechenden Zeiger und Bereiche sind dann:

Alternativer Speicherbereich

Sprite Nr.	0	1	2
Speicher-zeiger	13	14	15
Speicher-bereich	832 bis 894	896 bis 958	960 bis 1022

DIE FARBE DER SPRITES

Einem Sprite eine einzelne Farbe zu geben, ist sehr einfach. Alles, was Sie tun müssen, ist den Farbwert in die entsprechende Adresse POKEn. Wenn V=53248 ist, dann sind die Farbadressen der Sprites:

Sprite Nr.	0	1	2	3	4	5	6	7
Adresse	V+39	V+40	V+41	V+42	V+43	V+44	V+45	V+46

Es gibt jedoch auch einen Weg, Ihrem Sprite mehr als nur eine Farbe zu geben. Dazu müssen Sie das Sprite in den Mehrfarbenmodus schalten. Es gibt für alle acht Sprites nur ein Mehrfarben-Register. Das Register besteht aus acht Bits (einem Byte), und jedes Bit korrespondiert mit einem Sprite. Wenn ein Bit den Zustand 1 hat, dann befindet sich das entsprechende Sprite im Mehrfarbenmodus. Ist das Bit 0, dann befindet sich das Sprite im Normalmodus. Der einfachste Weg, ein Bit eines Bytes zu setzen, besteht darin, eine Dezimalzahl zu POKEn. Wenn wir z.B. Sprite 4 in den Mehrfarbenmodus schalten wollen, POKEn wir V+28 auf 16. 16 deshalb, weil 16 binär 10000 ist und ein POKEn dieses Wertes in das Mehrfarben-Register Bit 4 auf 1 setzt. Das läßt sich mit Hilfe eines Diagramms leichter verstehen. Vorausgesetzt, das Register wird anfangs auf 0 gesetzt:

Sprite Nr.	7	6	5	4	3	2	1	0
	128	64	32	16	8	4	2	1
Mehrfarben-Register	0	0	0	0	0	0	0	0
POKE mit 16	0	0	0	1	0	0	0	0

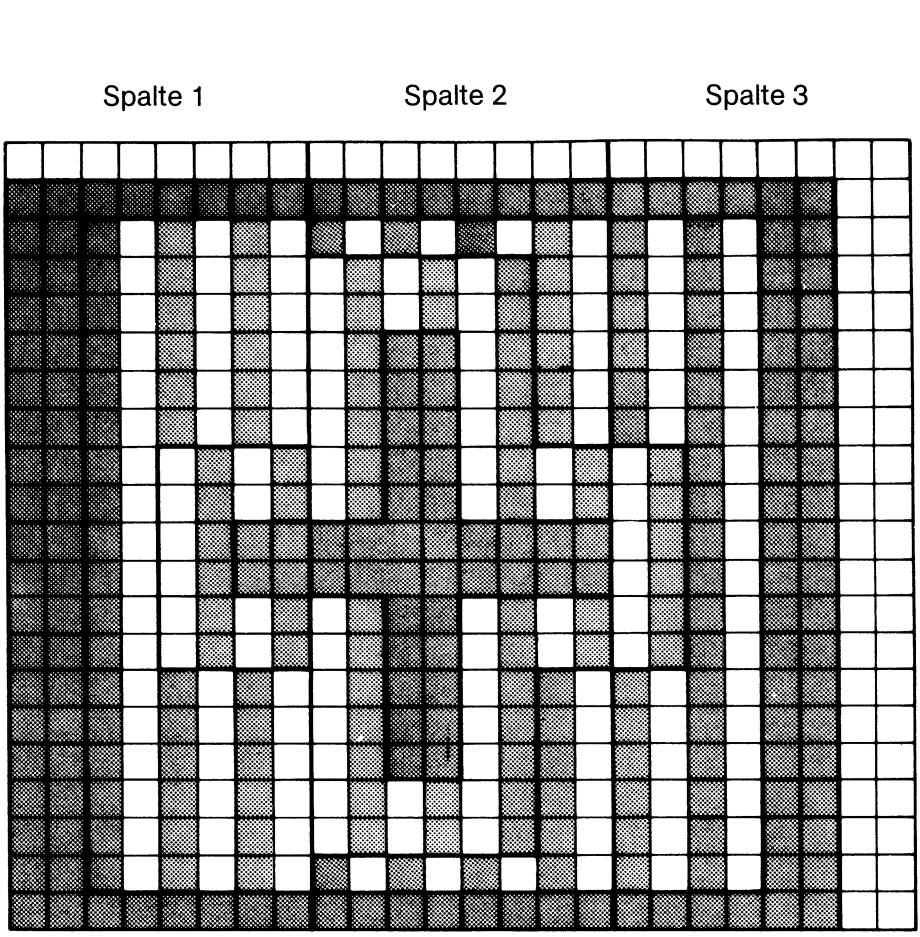
Wenn der Mehrfarbenmodus für ein Sprite gesetzt ist, dann werden die Daten für dieses Sprite anders verarbeitet. Der Computer paart nun zwei nebeneinanderliegende Pixels und verwendet diese Information, um über die Farbe des Paars zu entscheiden. Es gibt vier mögliche Kombinationen, die ein Pixelpaar annehmen kann:

Paar	Kombination	Farbe
0 0	beide Pixels aus	transparent, Hintergrund sichtbar
0 1	rechtes Pixel an	Pixelpaar hat Mehrfarbe 1
1 0	linkes Pixel an	Pixelpaar hat normale Sprite-Farbe
1 1	beide Pixels an	Pixelpaar hat Mehrfarbe 2

Mehrfarbe 1 wird festgelegt durch POKEV+37, Farbcode
Mehrfarbe 2 wird festgelegt durch POKEV+38, Farbcode

Leider müssen sich alle Sprites die beiden Mehrfarben-Register teilen. Das hat zur Folge, daß Sie nicht verschiedene Mehrfarben für verschiedene Sprites gleichzeitig auf dem Schirm erscheinen lassen können.

Um den Mehrfarbenmodus anschaulicher zu machen, folgt ein kurzes Programm, das ein Mehrfarbensprite (Tafel 6.3) zeigt. Beachten Sie, daß die Dezimalzahlen ganz normal errechnet werden, obwohl es sich um ein Mehrfarbensprite handelt. Das Programm stellt das Sprite horizontal und vertikal vergrößert dar.



DATEN Spa. 1	Spa. 2	Spa. 3
0	0	0
255	255	252
234	170	172
234	86	172
234	86	172
234	118	172
234	118	172
234	118	172
229	117	108
229	117	108
231	255	108
231	255	108
229	117	108
229	117	108
234	118	172
234	118	172
234	118	172
234	86	172
234	86	172
234	170	172
255	255	252

Mehrfarbe 1 – Rot

Sprite-Farbe – Grün

Mehrfarbe 2 – Schwarz

Hintergrundfarbe

Tafel 6.3

```
0 REM*** KAPITEL 6 PROGRAMM 2 ***
1 REM  **********************
2 REM  **********************
3 REM  **                    **
4 REM  ** MEHRFARBENMODUS **
5 REM  **      SPRITE        **
6 REM  **        DEMO        **
7 REM  **                    **
8 REM  **********************
9 REM  **********************
10 PRINTCHR$(147):REM BILDSCHIRM LOESCHE
N
20 V=53248
30 POKE2040,192:REM SPRITE-POINTER SETZE
N
40 FORI=12288TO12350:READA:POKEI,A
50 NEXT
60 POKEV+28,1:REM MEHRFARBENMODUS SETZEN
70 POKEV+37,2:REM ZUSATZFARBE 1
80 POKEV+38,0:REM ZUSATZFARBE 2
90 POKEV+39,5:REM NORMALE SPRITE-FARBE
100 POKEV,100:POKEV+1,100:REM SPRITE POS
ITIONIEREN
110 POKEV+23,1:POKEV+29,1:REM SPRITE VER
GROESSERN
120 POKEV+21,1:REM SPRITE ANSCHALTEN
190 REM  SPRITE-DATEN
200 DATA0,0,0,255,255,252,234,170,172
210 DATA234,86,172,234,86,172,234,118
220 DATA172,234,118,172,234,118,172,229
230 DATA117,108,229,117,108,231,255,108
240 DATA231,255,108,229,117,108,229,117
250 DATA108,234,118,172,234,118,172,234
260 DATA118,172,234,86,172,234,86,172
270 DATA234,170,172,255,255,252
```

Um bei der Flagge zu bleiben, zeigt das folgende Programm, wie man mittels POKE Zeichen auf den Bildschirm bringen kann und sie mit einem Sprite kombiniert. Es wird aus inversen Leerzeichen (POKE-Code 160) ein Flaggenmast gezeichnet, an dem dann die Flagge gehißt wird. Dieses Programm ist dem vorangegangenen sehr ähnlich; zwei Routinen wurden noch angefügt: Die eine zeichnet den Flaggenmast, die andere bewegt das Sprite den Mast hoch. Drücken Sie RUN/STOP und RESTORE, bevor Sie das Programm eingeben.

```
0 REM*** KAPITEL 6 PROGRAMM 3 ***
1 REM   ************
2 REM   ************
3 REM   **          **
4 REM   ** FLAGGE **
5 REM   ** HISSEN **
6 REM   **          **
7 REM   ************
8 REM   ************
10 PRINTCHR$(147):REM BILDSCHIRM LOESCHE
N
20 V=53248
30 POKE2040,192:REM SPRITE-POINTER SETZE
N
40 FORI=12288TO12350:READA:POKEI,A
50 NEXT
60 POKEV+28,1:REM MEHRFARBENMODUS SETZEN
70 POKEV+37,2:REM ZUSATZFARBE 1
80 POKEV+38,0:REM ZUSATZFARBE 2
90 POKEV+39,5:REM NORMALE SPRITE FARBE
100 REM FLAGGENMAST ZEICHNEN
110 POKE1231,160:POKE55503,1
120 POKE1232,160:POKE55504,1
130 POKE1233,160:POKE55505,1
140 FORI=1272TO1992STEP40
150 POKEI,160:POKE54272+I,1
160 NEXT
170 POKEV+23,1:POKEV+29,1:REM SPRITE VER
GROESSERN
180 POKEV+21,1:REM SPRITE ANSCHALTEN
190 POKEV,100
200 FORY=250TO100STEP-1
210 POKEV+1,Y:NEXT
1999 REM  SPRITE-DATEN
2000 DATA0,0,0,255,255,252,234,170,172
2010 DATA234,86,172,234,86,172,234,118
2020 DATA172,234,118,172,234,118,172,229
2030 DATA117,108,229,117,108,231,255,108
2040 DATA231,255,108,229,117,108,229,117
2050 DATA108,234,118,172,234,118,172,234
2060 DATA118,172,234,86,172,234,86,172
2070 DATA234,170,172,255,255,252
```

DIE POSITIONIERUNG DES SPRITES

Die Position eines Sprites wird durch zwei Koordinaten bestimmt. Die X-Koordinate bestimmt die horizontale Position des Sprites und die Y-Koordinate die vertikale. Beide Koordinaten können Werte von 0 bis 255 haben. Die Register, auf denen die X- und die Y-Koordinaten liegen, sind:

Sprite	0		1		2		3	
Koord.	X	Y	X	Y	X	Y	X	Y
Register	V	V+1	V+2	V+3	V+4	V+5	V+6	V+7

Sprite	4		5		6		7	
Koord.	X	Y	X	Y	X	Y	X	Y
Register	V+8	V+9	V+10	V+11	V+12	V+13	V+14	V+15

Der normale 40-mal-25-Zeichen-Bildschirm ist in 320 mal 200 Pixels eingeteilt. Die Y-Koordinate bewegt sich zwischen 0 und 255; es gibt in vertikaler Richtung also mehr als genug Platz. Ein Problem taucht auf, wenn wir uns die X-Koordinate betrachten: Die größte Zahl, die in einem der X-Register gespeichert werden kann, ist 255 (11111111 binär); der Schirm ist aber 320 Pixel breit. Es gibt jedoch eine Möglichkeit, auch die rechte Seite des Schirms zu erreichen. Wenn wir noch ein Bit mehr hätten, könnten wir einen Bereich von 0 bis 511 abdecken, mehr als genug also, um die X-Koordinate festzulegen. Dieses Zusatzbit befindet sich auf Adresse V+16. Da jedes Sprite nur ein zusätzliches Bit braucht, um die X-Koordinate zu verlängern, wurden alle Bits zu einem Byte zusammengefaßt. Jedes Bit in der Adresse V+16 repräsentiert also das neunte Bit der entsprechenden X-Koordinate. Um das verständlich zu machen, stellen Sie sich Sprite 0 vor, das sich von X=254 nach X=257 bewegen soll.

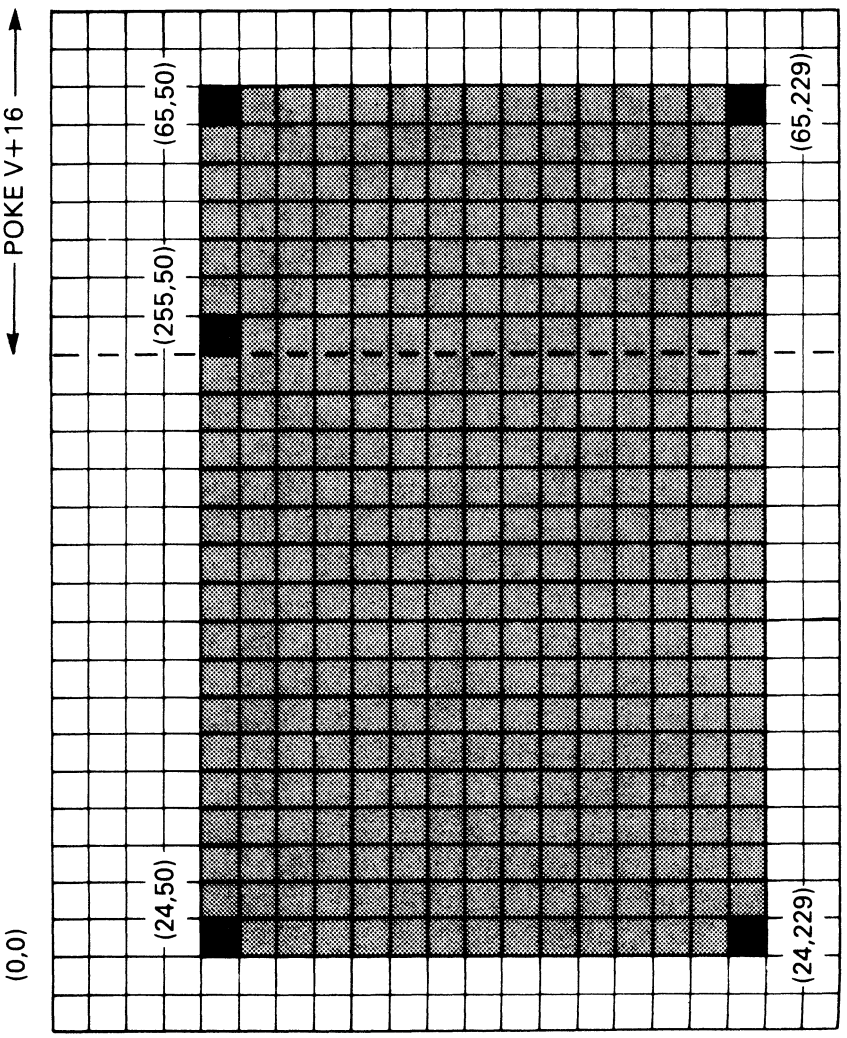
X-Koordinate	Adresse V+16								Adresse V							
254	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
255	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
256	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
257	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Da die X-Koordinate größer als 255 werden muß, wird bei der 256. Position das Zusatzbit für Sprite 0 gesetzt und das normale Register der X-Koordinate auf 0 gesetzt. Das ähnelt dem Wechsel, der auftritt, wenn zu 999 eins addiert wird, um 1000 zu erhalten. Die ersten drei Stellen wurden auf 0 gesetzt und die nächste auf 1.

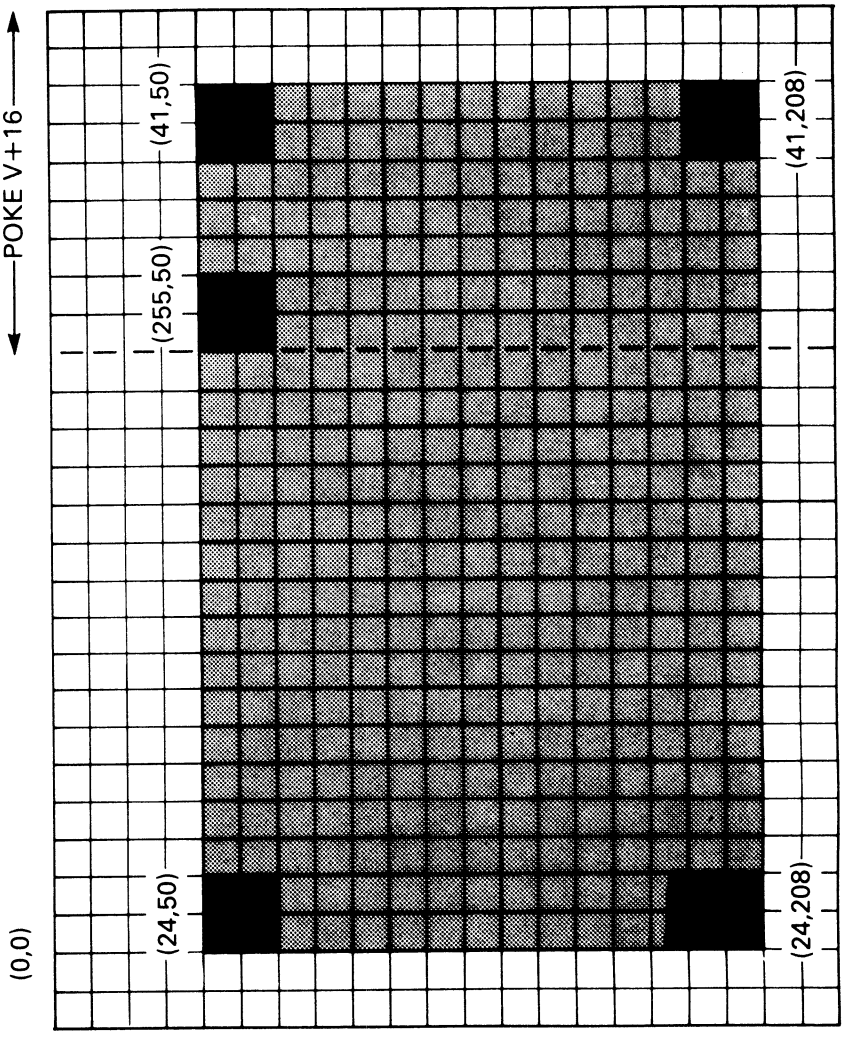
Vereinfacht gesagt: Um über die 255. Position hinauszukommen, POKEn Sie Register V+16 mit der entsprechenden Zahl und beginnen mit dem X-Register wieder von 0. Die Zahlen, die Sie in V+16 POKEn müssen, sind:

Sprite	0	1	2	3	4	5	6	7
POKE	1	2	4	8	16	32	64	128

Es ist möglich, wie schon zuvor gezeigt, das Sprite aus dem Bildschirmfenster zu bewegen. Der Ursprung der Koordinaten beginnt nicht in der linken oberen Ecke des Bildschirmfensters, sondern an einer Position außerhalb des Bildschirms. Als Positionshilfe zeigen Tafel 6.4 und 6.5 einige Schlüsselpunkte, zuerst für gewöhnliche Sprites, dann für vergrößerte Sprites.



Tafel 6.4



Tafel 6.5

ANDERE SPRITE-FUNKTIONEN

Alle weiteren Sprite-Funktionen werden durch je ein Register kontrolliert. Innerhalb dieser Register steht für jedes Sprite je ein Bit. Das am weitesten rechts gelegene Bit schaltet die jeweilige Funktion für Sprite 0 ein oder aus. Das Bit links davon steuert die Funktion für Sprite 1. usw. bis zum achten Bit, das die Funktion von Sprite 7 kontrolliert. Funktionen, die auf diese Weise arbeiten und noch nicht erwähnt wurden, sind:

Funktion	Register
Sprite ein-, ausschalten	V+21
In X-Richtung vergrößern	V+29
In Y-Richtung vergrößern	V+23

DIE VERWENDUNG VON „AND" UND „OR" ZUR ISOLATION VON BITS

Bei der Verwendung dieser Mehrfunktionsregister innerhalb von Programmen kommt es oft zu Problemen, wenn mehr als ein Sprite verwendet wird. Das auffälligste Beispiel findet sich bei der Verwendung des Registers V+21, das die Sprites an- und ausschaltet. Wir haben zuvor Sprite 0 angeschaltet und wollen jetzt noch Sprite 2 anschalten. Für das angeschaltete Sprite 0 sieht das Register V+21 so aus:

0	0	0	0	0	0	0	1

Um Sprite 2 anzuschalten, müssen wir eine 4 (00000100) POKEn. Leider schalten wir mit diesem Befehl nicht nur Sprite 2 an, sondern auch Sprite 0 wieder aus. Um beide Sprites auf dem Bildschirm erscheinen zu lassen, müssen wir das dezimale Äquivalent zu

0	0	0	0	0	1	0	1

POKEn, nämlich 5. 1 und 4 sind die Zahlen, die die Sprites 0 und 2 anschalten. Um beide anzuschalten, müssen wir nur die beiden Zahlen addieren. Zum Beispiel: Sprite 1 wird durch eine 2 aktiviert, Sprite 0 durch eine 1 und Sprite 2 durch eine 4. Also müssen wir auf V+21 den Wert 7 POKEn (1+2+4,00000111).

Diese Methode versagt allerdings, wenn Sie nicht wissen, welche Sprites momentan eingeschaltet sind. In Spielen tritt dieser Fall leider sehr häufig auf. Wir können auch diese Hürde mit den logischen Operatoren AND und OR (siehe Kapitel 4) nehmen.

Stellen Sie sich vor, wir wollen innerhalb eines Programms Sprite 4 aktivieren, ohne die übrigen Sprites zu beeinflussen. Das erreichen wir, indem wir den Wert von V+21 PEEKen und ihn mit der Zahl, die Sprite 4 einschaltet, mit Hilfe des logischen Operators OR verknüpfen. Im einzelnen sieht das so aus:

Der momentane Status des Registers, Sprites 0,1,5 und 6 sind eingeschaltet:

0	1	1	0	0	0	1	1

Wenn wir das Register PEEKen und mit OR 16 verknüpfen, erhalten wir:

0	1	1	0	0	0	1	1
OR 0	0	0	1	0	0	0	0
0	1	1	1	0	0	1	1

POKEV+21,PEEK(V+21)OR16
Daraufhin sieht das Register so aus:

0	1	1	1	0	0	1	1

Jetzt sind Sprites 0,1,4,5 und 6 eingeschaltet.

Um nur Sprite 4 auszuschalten, müssen wir den AND Operator so verwenden, daß nur das fünfte Bit des Registers auf Null gesetzt wird.

Wenn wir das Register PEEKen und mit AND239 verknüpfen, erhalten wir:

0	1	1	1	0	0	1	1
AND 1	1	1	0	1	1	1	1
0	1	1	0	0	0	1	1

POKEV+21,PEEK(V+21)AND239 stellt den ursprünglichen Zustand des Registers wieder her:

0	1	1	0	0	0	1	1

Die Zahl 239 erhalten wir, indem wir rechnen: 255—16.

Fassen wir zusammen:

Wenn „S" die Sprite-Nummer ist, schaltet folgender Befehl Sprite S an:

```
POKEV+21,PEEK(V+21)OR(2↑S)
```

Dieser Befehl schaltet Sprite S wieder aus:

```
POKEV+21,PEEK(V+21)AND(255—2↑S)
```

Diese Methode kann selbstverständlich bei allen Mehrfunktionsregistern verwendet werden.

DIE PRIORITÄTEN

(1) Sprite/Sprite-Priorität
Wenn sich zwei Sprites an derselben Bildschirmposition befinden, scheint das eine Sprite vor oder über dem anderen zu schweben. Das vordere Sprite hat eine höhere Priorität als das hintere. Diese Reihenfolge der Priorität hängt von der Sprite-Nummer ab und kann vom Programmierer nicht geändert werden. Sprites mit kleinerer Nummer haben eine höhere Priorität als Sprites mit größerer Nummer. Wenn sich also die Wege von Sprite 1 und 2 kreuzen, dann sieht es so aus, als ob sich Sprite 1 vor Sprite 2 vorbeibewegt. Wenn Sprite 3 im gleichen Moment kreuzen würde, würde es hinter Sprite 1 und 2 liegen. Falls Sie ein Programm schreiben, in dem Sie auf die Priorität Wert legen, überlegen Sie sich vorher, welches Sprite welche Priorität haben soll, und weisen Sie ihm die entsprechende Nummer zu. Die Sprite-Priorität kann auch bei Figuren, die aus mehreren Sprites bestehen, verwendet werden. Wenn das vordere Sprite „Löcher" hat, dann scheint das hintere an diesen Stellen durch. Durch das Übereinanderlegen von Sprites lassen sich interessante Formen- und Farbeffekte erzielen.

(2) Sprite/Hintergrund-Priorität
Die Sprite/Hintergrund-Priorität ist mit Hilfe des V+27-Registers programmierbar. Diese Funktion arbeitet genauso wie die anderen Mehrfunktionsregister; jedem Bit ist ein Sprite zugeordnet. Wenn ein bestimmtes Bit den Zustand 1 hat, dann erscheint das entsprechende Sprite hinter den Zeichen des Bildschirms. Beim Einschalten ist dieses Register auf 0. Solange Sie hier also keine Veränderungen vornehmen, erscheinen alle Sprites vor dem Hintergrund.

DIE KOLLISION

(1) Sprite/Sprite-Kollision
Sprite/Sprite-Kollisionen werden im Register V+30 abgelegt, wenn sie auftreten. Jedes Sprite hat in diesem Register ein Bit, das auf 1 steht, sobald dieses Sprite eine Kollision mit einem anderen Sprite hat. Wenn zwei Sprites zusammenstoßen, werden ihre beiden Bits auf 1 gesetzt. Um diesen Effekt in Ihren Programmen zu verwenden, müssen Sie das Register PEEKen, um nachzusehen, was es enthält. Das geschieht normalerweise innerhalb eines IF...THEN-Befehls:

```
IF(PEEK(V+30)AND2↑S)=2↑STHENGOSUBxxx
```

S ist in diesem Fall die Sprite-Nummer. Das AND isoliert das betreffende Bit des Registers. Sie wollen wahrscheinlich aufgrund der Kollision, wie im Beispiel, einige Maßnahmen ergreifen. Das Unterprogramm, das auf xxx beginnt, kann diese Programmschritte enthalten. Wenn Sie die Erkennung einer speziellen Kollision zwischen Sprite S1 und S2 wünschen, können Sie das mit diesem Befehl erreichen:

```
IF(PEEK)V+30)AND(2↑S1+2↑S2))=2↑S1+2↑S2THENGOSUBxxx
```

Anmerkung: Nachdem das Kollisions-Register gePEEKt wurde, wird es automatisch auf 0 gesetzt. Wenn Sie seinen Wert in einem anderen Programmteil weiterverwenden wollen, speichern Sie ihn in einer Variablen.

Wir stellten bei der Verwendung dieses Registers einige eigenartige Nebeneffekte fest. Wir kamen zu dem Schluß, daß das Register wahrscheinlich nicht schnell genug auf 0 gesetzt wird, nachdem es gePEEKt worden ist. Dieses Problem läßt sich umgehen, wenn wir das Register durch ein POKE „manuell" zurücksetzen, bevor wir es auslesen.

(2) Sprite/Hintergrund-Kollision
Das Sprite/Hintergrund-Kollisions-Register liegt auf V+31. Es arbeitet ähnlich wie das Sprite/Sprite-Register. Jedes Sprite hat in diesem Register ein Bit, das bei einer Kollision seines Sprites mit Daten des Hintergrundes auf 1 gesetzt wird. Auch dieses Register wird automatisch gelöscht, nachdem es gePEEKt worden ist.

Sie können bestimmen, mit welchen Teilen des Hintergrundes ein Sprite kollidieren kann. Wenn sich der Hintergrund im Mehrfarbenmodus befindet, sind Teile, die die Hintergrundfarbe 1 haben, zwar sichtbar, erscheinen dem Sprite aber nicht als „fest". Mit anderen Worten: Wenn ein Sprite mit einem Teil des Hintergrunds kollidiert, der die Hintergrundfarbe 1 besitzt, wird diese Kollision nicht im Kollisionsregister verzeichnet (siehe Kapitel 2 für die Erklärung des Mehrfarben-Hintergrundmodus).

„Rundflug"-Programm
Das folgende Programm zeigt sehr anschaulich die Verwendung der Prioritäten, der Speicherzeiger und der Kollisionsregister:

```
1 REM  ******************
2 REM  ******************
3 REM  **                **
4 REM  **  RUNDFLUG      **
5 REM  **                **
6 REM  ******************
7 REM  ******************
10 PRINTCHR$(147)
100 REM *** SPRITE-DATEN LESEN ***
110 REM SPRITE-0A-DATEN NACH 12288-12350
120 FORI=12288TO12350:READA:POKEI,A:NEXT
130 REM SPRITE-0B-DATEN NACH 832-894
140 FORI=832TO894:READA:POKEI,A:NEXT
150 REM SPRITE-0C-DATEN NACH 896-958
160 FORI=896TO958:READA:POKEI,A:NEXT
170 REM SPRITE-0D-DATEN NACH 960-1022
180 FORI=960TO1022:READA:POKEI,A:NEXT
190 REM SPRITE-0E-DATEN NACH 12352-12414
200 FORI=12352TO12414:READA:POKEI,A:NEXT
210 V=53248:GOSUB1000:REM BILDSCHIRM AUF
BAUEN
220 REM *** SPRITE FAERBEN ***
230 POKEV+39,0
240 REM *** Y-KOORDINATE SETZEN ***
250 POKEV+1,120
260 REM *** SPRITE 0 ANSCHALTEN ***
270 POKEV+21,1
280 REM *** SPRITE VERGROESSERN ***
290 POKEV+23,1:POKEV+29,1
300 REM *** HAUPTSCHLEIFE FUER RUNDFLUG
***
310 REM *** SPRITE-POINTER AUF 0A ***
320 POKE2040,192
330 FORX=70TO250:POKEV,X
340 IFL=5ANDPEEK(V+31)AND1=1THEN600
350 NEXT
360 REM *** SPRITE-POINTER AUF 0D ***
370 POKE2040,15:FORJ=1TO500:NEXT
380 REM *** SPRITE-POINTER AUF 0B ***
390 POKE2040,13
```

```
400 REM *** HINTERGRUND HAT PRIORITAET G
EGENUEBER DEN SPRITES ***
410 POKEV+27,1
420 FORX=250TO70STEP-1:POKEV,X
430 FORJ=1TO10:NEXT:NEXT
440 REM *** SPRITE-POINTER AUF 0C ***
450 POKE2040,14:FORJ=1TO500:NEXT
460 REM *** SPRITES HABEN WIEDER PRIORIT
AET GEGENUEBER DEM HINTERGRUND ***
470 POKEV+27,0
480 REM *** SCHLEIFENDURCHLAEUFE ZAEHLEN
 ***
490 L=L+1
500 REM *** KOLLISIONSREGISTER LOESCHEN
***
510 POKEV+31,0
520 REM *** NAECHSTER DURCHLAUF ***
530 GOTO300
600 REM *** ABSTURZ ***
610 REM *** SPRITE-POINTER AUF 0E ***
620 POKE2040,193
630 REM *** GEBAEUDE HERUNTERRUTSCHEN **
*
640 FORY=120TO180:POKEV+1,Y
650 FORJ=1TO10:NEXT:NEXT
660 END
1000 REM *** BILDSCHIRM AUFBAUEN ***
1010 FORI=1083TO1163STEP40:POKEI,103:POK
EI+1,101
1020 POKE54272+I,7:POKE54272+I+1,7
1030 NEXT
1035 REM *** GRASS ***
1040 FORI=1744TO1983:POKEI,160
1050 POKE54272+I,5:NEXT
1055 REM *** DACH ***
1060 POKE1203,160:POKE1204,160
1070 FORJ=0TO3:POKE1242+J,160:NEXT
1080 POKE1281,233
1090 FORJ=0TO3:POKE1282+J,160:NEXT
1100 POKE1286,223
1110 FORI=1201TO1281STEP40:FORJ=0TO5
1120 POKE54272+I+J,7:NEXT:NEXT
1125 REM *** MAUERN ***
1130 FORI=1321TO1881STEP40:FORJ=0TO5
1140 POKEI+J,160:POKE54272+I+J,7:NEXT:NE
XT
```

```
1145 REM *** FENSTER ***
1150 FORI=1321TO1801STEP80
1160 POKEI+1,32:POKEI+4,32:NEXT
1165 REM *** STRASSEN ***
1170 FORI=1984TO2023:POKEI,160
1180 POKE54272+I,11:NEXT
1190 FORI=1923TO1963STEP40
1200 POKEI,160:POKEI+1,160
1210 POKE54272+I,11:POKE54272+I+1,11
1220 NEXT
1230 FORI=1797TO1807STEP10
1240 FORJ=0TO3:POKEI+J,160
1250 POKE54272+I+J,11:NEXT:NEXT
1260 FORJ=1837TO1849STEP12
1270 FORI=JTOJ+120STEP40
1280 POKEI,160:POKEI+1,160
1290 POKE54272+I,11:POKE54272+I+1,11
1300 NEXT:NEXT
1999 RETURN
10000 REM *** DATEN FUER SPRITE 0A ***
10010 DATA0,0,0,0,0,0,0,0,0
10020 DATA30,0,0,15,0,0,7,128,0
10030 DATA3,192,0,193,224,1,224,240
10040 DATA1,224,236,1,255,233,252
10050 DATA127,255,255,7,255,86,0,255
10060 DATA253,0,15,129,0,7,193,0
10070 DATA3,224,0,1,240,0,0,248
10080 DATA0,0,124,0,0,62
10100 REM *** DATEN FUER SPRITE 0B ***
10110 DATA0,0,0,0,0,0,0,0,0
10120 DATA240,0,0,120,0,0,60,0,0
10130 DATA158,0,3,143,0,7,135,176
10140 DATA7,63,151,255,255,255,254
10150 DATA106,255,224,191,255,0,128
10160 DATA248,0,128,124,0,0,62,0
10170 DATA0,31,0,0,15,128,0,7,192
10180 DATA0,3,224,0,0,0
10200 REM *** DATEN FUER SPRITE 0C ***
10210 DATA0,0,0,0,0,0,0,0,0,0,0,0,0
10220 DATA8,0,0,8,0,0
10230 DATA0,0,0,42,0,1,200,224,0
10240 DATA136,128,127,34,127,1,8,64
10250 DATA1,34,64,0,73,0,0,128
10260 DATA128,0,34,0,0,8
10270 DATA0,0,0,0,0,0,0,0,0,0,0,0,0
10300 REM *** DATEN FUER SPRITE 0D ***
```

```
10310 DATA0,0,0,0,0,0,0,0,0
10320 DATA0,0,0,0,0,0,0,0,0,0
10330 DATA16,0,0,84,0,0,16,0,1
10340 DATA57,0,1,254,0,255,255,255
10350 DATA0,56,0,0,130,0,0,40,0
10360 DATA0,0,0,0,0,0,0,0,0
10370 DATA0,0,0,0,0,0,0,0,0
10400 REM *** DATEN FUER SPRITE 0E ***
10410 DATA0,0,0,8,0,0,28,0,0
10420 DATA56,0,0,120,0,0,240,240
10430 DATA0,126,120,2,15,176,1,3
10440 DATA230,137,12,245,220,30,127
10450 DATA255,0,63,214,0,31,253,0
10460 DATA7,201,0,3,226,0,1,240,0
10470 DATA7,224,0,31,0,0,124
10480 DATA0,0,0,0,0,0,0
```

Tabelle der VIC-Chip Adressen

Adresse	Adresse relativ zu 53248 (=V)	Funktion
53248	V	X-Koordinate von Sprite 0
53249	V+1	Y-Koordinate von Sprite 0
53250	V+2	X-Koordinate von Sprite 1
53251	V+3	Y-Koordinate von Sprite 1
53252	V+4	X-Koordinate von Sprite 2
53253	V+5	Y-Koordinate von Sprite 2
53254	V+6	X-Koordinate von Sprite 3
53255	V+7	Y-Koordinate von Sprite 3
53256	V+8	X-Koordinate von Sprite 4
53257	V+9	Y-Koordinate von Sprite 4
53258	V+10	X-Koordinate von Sprite 5
53259	V+11	Y-Koordinate von Sprite 5
53260	V+12	X-Koordinate von Sprite 6
53261	V+13	Y-Koordinate von Sprite 6
53262	V+14	X-Koordinate von Sprite 7
53263	V+15	Y-Koordinate von Sprite 7

Adresse	Adresse relativ zu 53248 (=V)	Funktion
53264	V+16	Zusatzbit für die X-Koordinate
53269	V+21	schaltet Sprites 0–7 AN/AUS
53271	V+23	vergrößert Sprites 0–7 in Y-Richtung
53275	V+27	bestimmt Sprite/Hintergrund-Priorität
53276	V+28	schaltet Mehrfarbenmodus AN/AUS
53277	V+29	vergrößert Sprites 0–7 in X-Richtung
53278	V+30	Sprite/Sprite-Kollisions-Register
53279	V+31	Sprite/Hintergrund-Kollisions-Register
53285	V+37	Mehrfarbe 1
53286	V+38	Mehrfarbe 2
53287	V+39	Farbe von Sprite 0
53288	V+40	Farbe von Sprite 1
53289	V+41	Farbe von Sprite 2
53290	V+42	Farbe von Sprite 3
53291	V+43	Farbe von Sprite 4
53292	V+44	Farbe von Sprite 5
53293	V+45	Farbe von Sprite 6
53294	V+46	Farbe von Sprite 7

7

Der Joystick

EINLEITUNG

An der rechten Seitenwand Ihres Commodore 64, neben dem ON/OFF-Schalter, finden Sie zwei Joystickbuchsen. Sie werden mit CONTROL PORT 1 und CONTROL PORT 2 bezeichnet. Stecken Sie Ihren Joystick in den PORT 2 und geben Sie das folgende Programm ein:

```
10 PRINT CHR$(145);PEEK(56320)
20 GOTO 10
```

Starten Sie das Programm. Auf dem Bildschirm wird eine Zahl angezeigt. Verwenden Sie jetzt Ihren Joystick; bewegen Sie ihn in alle Richtungen und drücken Sie den Feuerknopf. Beachten Sie, wie sich die Zahl verändert, während Sie den Joystick bewegen. (Wenn sich die Zahl nicht verändert, prüfen Sie, ob Sie den Joystick am richtigen Port angeschlossen haben!)

Was macht nun dieses kleine Programm? Zeile 10 zeigt den Inhalt des Speicherplatzes 56320. In Ihrem Joystick sind fünf Schalter eingebaut, vier für die Steuerrichtungen und einer für den Feuerknopf. Jeder dieser Schalter beeinflußt ein Bit der Speicherstelle 56320. Während das Programm läuft, vergleichen Sie die untenstehende Tabelle mit der Zahl auf dem Bildschirm.

Control Port 2

Joystick	Dezimal	Binär
keine	127	01111111
oben	126	01111110
unten	125	01111101
links	123	01111011
rechts	119	01110111
Feuer	111	01101111

Beachten Sie, daß ein Bit auf Null gesetzt wird, wenn sein korrespondierender Schalter betätigt wird. Diagonale Bewegungen lassen sich aus der Kombination von zwei der vier Richtungen erreichen. Dadurch erhält man vier weitere Codes:

oben & links	122	01111010
oben & rechts	118	01110110
unten & links	121	01111001
unten & rechts	117	01110101

Unterbrechen Sie das Programm und verändern Sie mit dem Cursor Zeile 10 so, daß der zu PEEKende Speicherplatz in 56321 geändert wird. Schließen Sie den Joystick jetzt an Port 1 an und starten Sie das Programm erneut. Die Zahl, die anfangs angezeigt wird, ist 255 (binär 11111111). Auch diese Zahl wird sich verändern, wenn Sie den Joystick bewegen. Vergleichen Sie die Zahlen mit der folgenden Tabelle:

Control Port 1

Joystick	Dezimal	Binär
keine	255	11111111
oben	254	11111110
unten	253	11111101
links	251	11111011
rechts	247	11110111
Feuer	239	11101111
oben & links	250	11111010
oben & rechts	246	11110110
unten & links	249	11111001
unten & rechts	245	11110101

Unterbrechen Sie das Programm wieder. Wenn Sie jetzt den Joystick bewegen, erscheinen einige sinnlose Zeichen auf dem Schirm. Dies geschieht nur, wenn der Joystick an Port 1 angeschlossen ist. Deshalb ist bei Spielen, die nur einen Joystick verwenden, Port 2 vorzuziehen.

Fassen wir noch einmal zusammen, was wir über die Funktionsweise des Joysticks herausgefunden haben:

Der Joystick hat fünf Schalter, vier, die die Bewegungsrichtung steuern und einen für den Feuerknopf. Der Commodore 64 verfügt über zwei Joystick-Ports. Jeder Port ist mit einer Speicherstelle innerhalb des Computers verbunden. Port 2 beeinflußt 56320, Port 1 beeinflußt 56321. Die vier Schalter für die Richtungen korrespondieren mit den ersten vier Bits der Speicherstelle. Das fünfte Bit ist mit dem Feuerknopf ver-

bunden. Wenn ein Schalter des Joysticks geschlossen wird, wird das entsprechende Bit der Speicherstelle auf Null gesetzt. Sobald der Kontakt unterbrochen ist, wird der Zustand des Bits wieder 1. Diagonale Bewegungen werden durch Betätigen zweier Richtungsschalter gleichzeitig, also durch einen diagonalen Ausschlag des Joysticks, erzielt.

EINE JOYSTICKABFRAGE IN BASIC

Es ist sehr unpraktisch, bei der Joystickabfrage große Dezimalzahlen zu verarbeiten. Da alle Steuerrichtungen nur die ersten vier Bits beeinflussen, wollen wir sie mittels AND isolieren. Zuerst legen wir den Wert der Speicherstelle auf einer Variablen ab, der wir den Namen PJ (für PEEK JOYSTICK) geben wollen. Unser Unterprogramm beginnt in Zeile 2000:

```
2000 REM UNTERPROGRAMM ZUR JOYSTICKABFRAGE
2010 PJ=PEEK(56320)
2020 DJ=15-(PJAND15)
2030 FB=PJAND16
2040 RETURN
```

Die Operation PJAND15 isoliert die ersten vier Bits von PJ. Wenn wir diesen Wert von 15 abziehen, erhalten wir eine Zahl zwischen 0 und 10, die einer der acht möglichen Richtungen entspricht.

Zeile 2030 isoliert das fünfte Bit von PJ. Dieses Bit bezieht sich auf den Feuerknopf (FB). Wenn der Feuerknopf nicht gedrückt ist, hat dieses Bit den Wert 1 und FB=16. Ist er gedrückt, dann ist FB=0.

Die Werte von DJ entsprechen diesen Ausschlägen am Joystick:

DJ	Richtung
0	–
1	oben
2	unten
3	–
4	links
5	oben & links
6	unten & links
8	–
9	oben & rechts
10	unten & rechts

Innerhalb eines Programmes könnte dieses Unterprogramm wie folgt verwendet werden:

```
1 REM  ***************
2 REM  ***************
3 REM  **             **
4 REM  **   BASIC     **
5 REM  ** JOYSTICK    **
6 REM  **   DEMO      **
7 REM  **             **
8 REM  ***************
9 REM  ***************
10 PRINTCHR$(147)
100 REM *** SPRITE INITIALISIEREN ***
110 V=53248:X=150:Y=140:REM VARIABLEN
120 POKE2040,192:REM SPRITE-POINTER SETZ
EN
130 FOR I=12288TO12350:POKEI,255:NEXT
140 POKEV+39,7:REM SPRITE GELB FAERBEN
150 POKEV+21,1:REM SPRITE ANSCHALTEN
200 REM *** HAUPTSCHLEIFE FUER BEWEGUNG
***
210 GOSUB2000:REM PEEK JOYSTICK
220 IFDJ=1THENY=Y-1:IFY<50THENY=50
230 IFDJ=2THENY=Y+1:IFY>229THENY=229
240 IFDJ=4THENX=X-1:IFX<24THENX=24
250 IFDJ=8THENX=X+1:IFX>320THENX=320
260 HX=INT(X/256):LX=X-256*HX
270 POKEV+16,HX
280 POKEV,LX:POKEV+1,Y:REM NEUE KOORDINA
TEN
290 IFFB=16THEN200:REM NAECHSTER SCHLEIF
ENDURCHLAUF
300 PRINTCHR$(147);TAB(13)"FEUER FREI!!"
310 FORI=1TO1000:NEXT:REM VERZOEGERUNG
320 PRINTCHR$(147)
330 GOTO200:REM NAECHSTER SCHLEIFENDURCH
LAUF
999 END
2000 REM JOYSTICK-UNTERPROGRAMM
2010 PJ=PEEK(56320)
2020 DJ=15-(PJAND15)
2030 FB=PJAND16
2040 RETURN
```

Die Hauptbewegungsschleife beginnt in Zeile 200. Sie enthält eine Reihe von Abfragen und die den Abfragen entsprechenden Programmteile. In Zeile 220 wird z.B. überprüft, ob der Wert von DJ gleich 1 ist. Dieser Wert von DJ entspricht dem Steuerausschlag „Oben" am Joystick. Falls die Antwort auf diese Abfrage „Ja" lautet, DJ also gleich 1 ist, wird der momentane Wert der Y-Koordinate um Eins vermindert. Das Sprite bewegt sich um ein Pixel nach oben, wenn ihm in den Zeilen 270–280 die neuen X- und Y-Koordinaten übergeben werden. Beachten Sie, daß in den Zeilen 220–250 hinter der ersten Bedingung noch eine zweite folgt. In ihr werden die Maximal- und Minimalwerte der X- und Y-Koordinaten festgelegt. Es ist sinnvoll, die Abfrage an diese Stelle zu legen, da der Computer diese Bedingung nur abarbeitet, wenn die erste Abfrage wahr ist. Ist die Bedingung falsch, überspringt der Computer die zweite Abfrage und fährt gleich mit der folgenden Programmzeile fort. Das spart Ausführungszeit, weil der Computer überflüssige Abfragen nicht durchführen muß. Die Begrenzungen wurden so gesetzt, daß das Sprite nie aus dem Blickfeld geraten kann. Wenn X und Y zwischen 0 und 255 liegen dürfen, können Sie das Sprite aus dem sichtbaren Bereich hinausbewegen. Probieren Sie es aus, indem Sie die untere Grenze von X und Y auf 0 und die obere Grenze von Y auf 255 setzen. Starten Sie das Programm erneut; Sie können das Sprite jetzt aus dem Bildschirmfenster hinausmanövrieren. Sie können es sogar auf der einen Seite verschwinden und auf der anderen Seite wieder auftauchen lassen. Ändern Sie den zweiten Teil der Zeile 220 zu:

```
IFY<29THENY=250
```

und den zweiten Teil der Zeile 230 zu:

```
IFY>250THENY=29
```

Das Programm bewegt das Sprite um jeweils ein Pixel in jede Richtung. Das bedeutet, daß die Bewegung zwar sehr gleichmäßig, aber ein bißchen langsam ist. Sie können den Ablauf beschleunigen, indem Sie das Sprite mehr als ein Pixel je Schleifendurchlauf bewegen. Ändern Sie in den Zeilen 220–250 Y=Y−1 zu Y=Y−2 usw. Es besteht selbstverständlich ein direkter Zusammenhang zwischen der Geschwindigkeit und der Gleichmäßigkeit, mit der sich ein Sprite bewegt. Experimentieren Sie mit der Zahl der Pixels, mit der ein Sprite bewegt wird.

Das Programm ist so aufgebaut, daß die Hauptschleife erneut durchlaufen wird, wenn in Zeile 290 FB=16 ist (FB ist dann 16, wenn der Feuerknopf nicht gedrückt ist). Der Computer kann die auf 290 folgenden Zeilen nur durchlaufen, wenn der Feuerknopf gedrückt, FB also ungleich 16 ist. Das Programm gibt dann nur eine kleine Meldung aus, die nach einiger Zeit wieder gelöscht wird. Während die Meldung auf dem Schirm erscheint, kann das Sprite nicht bewegt werden, da der Computer die Verzögerungsschleife in Zeile 310 abarbeitet. Die Moral: Machen Sie Ihre Feuer- und Schußroutinen so kurz wie möglich, damit diese den Programmablauf nicht aufhalten. So viel zu BASIC-Unterprogrammen. Sie sind jetzt in der Lage, Programme zu schrei-

ben, deren Ablauf durch einen Joystick kontrolliert wird. Sie werden jedoch feststellen, daß Sie in Zeitprobleme kommen, wenn Ihre Programme umfangreicher werden. Um die Joystickabfrage zu beschleunigen, müssen wir ein Maschinenprogramm einsetzen.

EINE JOYSTICKABFRAGE IN MASCHINENSPRACHE

Das folgende Programm ist der BASIC-Lader für die Maschinenroutine. Da es in BASIC geschrieben ist, können Sie es einfach in Ihren Commodore 64 eingeben. Um zu gewährleisten, daß die DATAs richtig eingegeben wurden, befindet sich am Ende des Programms eine Prüfsumme. Alle Werte werden addiert, während sie eingelesen werden. Am Ende des Programms wird die errechnete Summe mit der Prüfsumme verglichen. Wenn die Meldung „FEHLER IN DEN DATEN" erscheint, haben Sie die DATAs falsch eingegeben.

Die Joystickabfrage ist von 2000 an aufwärts durchnumeriert. Nachdem Sie das Unterprogramm einmal eingegeben haben, sollten Sie es auf Diskette oder Kassette speichern. Wenn Sie ein Programm mit Joystickabfrage schreiben wollen, laden Sie zuerst das Unterprogramm und konstruieren das restliche Programm darum herum.

```
2000 REM  ******************
2010 REM  **                **
2020 REM  ** BASIC-LADER    **
2030 REM  **  FUER MS-      **
2040 REM  ** ABFRAGE DES    **
2050 REM  **  JOYSTICKS     **
2060 REM  **                **
2070 REM  ******************
2100 DATA120,173,20,3,141,29,192,173,21
2110 DATA3,141,30,192,169,71,141,20,3
2120 DATA169,192,141,21,3,88,96,120,173
2130 DATA29,192,141,20,3,173,30,192,141
2140 DATA21,3,88,96,173,27,192,205,16
2150 DATA192,240,3,76,65,193,169,255
2160 DATA141,27,192,162,0,160,0,185,2
2170 DATA220,141,28,192,169,224,153,2
2180 DATA220,185,0,220,72,173,28,192
2190 DATA153,2,220,189,8,192,221,21,192
2200 DATA189,9,192,253,22,192,144,18
2210 DATA189,8,192,157,21,192,189,9,192
2220 DATA157,22,192,104,74,72,76,162
2230 DATA192,104,74,72,176,13,222,21
2240 DATA192,189,21,192,201,255,208,3
2250 DATA222,22,192,189,21,192,221,12
```

```
2260 DATA192,189,22,192,253,13,192,144
2270 DATA18,189,12,192,157,21,192,189
2280 DATA13,192,157,22,192,104,74,72,76
2290 DATA207,192,104,74,72,176,8,254,21
2300 DATA192,208,3,254,22,192,189,0,192
2310 DATA221,17,192,189,1,192,253,18
2320 DATA192,144,18,189,0,192,157,17
2330 DATA192,189,1,192,157,18,192,104
2340 DATA74,72,76,1,193,104,74,72,176
2350 DATA13,222,17,192,189,17,192,201
2360 DATA255,208,3,222,18,192,189,17
2370 DATA192,221,4,192,189,18,192,253,5
2380 DATA192,144,18,189,4,192,157,17
2390 DATA192,189,5,192,157,18,192,104
2400 DATA74,72,76,46,193,104,74,72,176
2410 DATA8,254,17,192,208,3,254,18,192
2420 DATA104,74,176,5,169,1,153,25,192
2430 DATA200,232,232,224,4,240,3,76,91
2440 DATA192,238,27,192,108,29,192
2450 DATA35994:REM PRUEFSUMME
2460 FORI=49183TO49478
2470 READX:POKEI,X:CC=CC+X
2480 NEXT
2490 READX:IFX<>CCTHEN PRINT"FEHLER IN D
EN DATEN."
2500 RETURN
```

Nachdem das Unterprogramm gestartet ist, müssen wir noch einige andere Dinge erledigen, bevor wir den Joystick verwenden können. Zunächst müssen die Ober- und Untergrenzen der X- und Y-Werte festgelegt werden. Das wird durch POKEn von Zahlen in diese Adressen erreicht:

Grenze	Port	Lo-Byte	Hi-Byte
X MIN	2	49152	49153
X MIN	1	49154	49155
X MAX	2	49156	49157
X MAX	1	49158	49159
Y MIN	2	49160	49161
Y MIN	1	49162	49163
Y MAX	2	49164	49165
Y MAX	1	49166	49167

Wie Sie sehen können, hat jede Begrenzung zwei Adressen. Die eine wird als „Lo-Byte" (niederwertiges Byte), die andere als „Hi-Byte" (höherwertiges Byte) bezeichnet. Die größte Zahl, die in einer Adresse gespeichert werden kann, ist 255 (Binär 11111111). Die sichtbaren X-Koordinaten bewegen sich jedoch zwischen 24 und 343. Es wird also eine zweite Adresse benötigt, die X-Werte größer 255 enthält.

Hi- und Lo-Byte werden so kombiniert:

Wert von X	Hi-Byte	Lo-Byte
254	0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 0
255	0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1
256	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 0
257	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 1

Die obere X-Begrenzung für Joystick 2 kann so auf 321 gePOKEt werden:

```
POKE49156,65:POKE49157,1
```

Als nächstes muß die Abfragehäufigkeit festgelegt werden. Sie bestimmt, wie oft der Joystick abgefragt wird. Durch POKEn einer Zahl zwischen 0 und 255 in die Adresse 49168 wird eine entsprechende Abfragehäufigkeit zwischen 1/60 und 4,3 Sekunden eingestellt.

Bevor Sie das Hauptprogramm beginnen lassen, müssen Sie noch das Maschinenprogramm aktivieren. Das geschieht mit dem folgenden Befehl:

```
SYS49183
```

Diese Adressen müssen Sie PEEKen, um die X- und Y-Werte zu erhalten:

Koord.	Lo-Byte	Hi-Byte
X2	49169	49170
X1	49171	49172
Y2	49173	49174
Y1	49175	49176

Den Zustand der Feuerknöpfe können Sie für Feuerknopf 2 aus Adresse 49177 und für Feuerknopf 1 aus Adresse 49178 lesen. Wenn Sie diese Adressen innerhalb einer Schleife PEEKen, verwenden Sie die folgende Routine:

```
100 POKE49177,0:POKE49178,0
110 K2=PEEK(49177):K1=PEEK(49178)
120 IFK1=1THEN(Aktion)
130 IFK2=2THEN(Aktion)
140 GOTO100
```

Beide Feuerknopfadressen sollten innerhalb der Schleife auf 0 gesetzt werden; das geschieht am besten direkt vor dem Auslesen.

Am Ende eines Programms sollten Sie das Maschinenprogramm mit SYS49208 desaktivieren.

Wenn Sie ein Programm, das die Joystickabfrage verwendet, unterbrechen und erneut starten, kann es passieren, daß der Joystick fehlerhaft arbeitet. Um dieses Problem zu beheben, drücken Sie RUN/STOP und RESTORE oder desaktivieren das Maschinenprogramm mit SYS49208.

Dieses Programm vereint alle Schritte in sich:

```
1 REM  ****************
2 REM  ****************
3 REM  **              **
4 REM  ** M-SPRACHE   **
5 REM  ** JOYSTICK    **
6 REM  **    DEMO      **
7 REM  **              **
8 REM  ****************
9 REM  ****************
10 PRINTCHR$(147):REM BILDSCHIRM LOESCHE
N
20 V=53248
30 GOSUB2000:REM M-S JOYSTICK
100 REM *** BEGRENZUNGEN SETZEN ***
110 POKE49152,24:POKE49153,0: REM X2 MIN
IMUM
120 POKE49156,65:POKE49157,1: REM X2 MAX
IMUM
130 POKE49160,50:POKE49161,0: REM Y2 MIN
IMUM
140 POKE49164,229:POKE49165,0:REM Y2 MAX
IMUM
150 REM *** ABFRAGEHAEUFIGKEIT SETZEN **
*
```

```
160 POKE49168,0
200 REM *** SPRITE INITIALISIEREN ***
210 FORI=12288TO12350:POKEI,255:NEXT
220 POKE2040,192:REM SPRITE-ZEIGER
230 POKEV+39,7:REM FARBE
240 POKEV+21,1:REM ANSCHALTEN
250 REM *** ANFANGSPOSITION SETZEN ***
260 POKE49169,160:POKE49170,0:REM X2
270 POKE49173,140:POKE49174,0:REM Y2
280 REM *** MS-PROGRAMM AUFRUFEN ***
290 SYS49183
300 REM *** HAUPTPROGRAMM ***
310 X2=PEEK(49169)+256*PEEK(49170)
320 Y2=PEEK(49173)+256*PEEK(49174)
330 REM *** MSB BEI BEDARF SETZEN ***
340 H2=INT(X2/256):L2=X2-256*H2
350 POKEV+16,H2
360 REM *** SPRITE POSITIONIEREN ***
370 POKEV,L2:POKEV+1,Y2
380 REM *** FEUERKNOPF ABFRAGEN ***
390 POKE49177,0:REM FEUERKNOPF-REGISTER
LOESCHEN
400 FB2=PEEK(49177)
410 IFFB2<>1THEN300
420 PRINTCHR$(145):REM CURSOR NACH OBEN
430 PRINTTAB(13)"FEUER FREI"
440 FORI=1TO500:NEXT
450 PRINTCHR$(147):REM BILDSCHIRM LOESCH
EN
460 GOTO300
2000 REM *** MS-PROGRAMM FUER BASIC-LADE
R ***
2010 DATA120,173,20,3,141,29,192,173,21
2020 DATA3,141,30,192,169,71,141,20,3
2030 DATA169,192,141,21,3,88,96,120,173
2040 DATA29,192,141,20,3,173,30,192,141
2050 DATA21,3,88,96,173,27,192,205,16
2060 DATA192,240,3,76,65,193,169,255
2070 DATA141,27,192,162,0,160,0,185,2
2080 DATA220,141,28,192,169,224,153,2
2090 DATA220,185,0,220,72,173,28,192
2100 DATA153,2,220,189,8,192,221,21,192
2110 DATA189,9,192,253,22,192,144,18
2120 DATA189,8,192,157,21,192,189,9,192
2130 DATA157,22,192,104,74,72,76,162
```

```
2140 DATA192,104,74,72,176,13,222,21
2150 DATA192,189,21,192,201,255,208,3
2160 DATA222,22,192,189,21,192,221,12
2170 DATA192,189,22,192,253,13,192,144
2180 DATA18,189,12,192,157,21,192,189
2190 DATA13,192,157,22,192,104,74,72,76
2200 DATA207,192,104,74,72,176,8,254,21
2210 DATA192,208,3,254,22,192,189,0,192
2220 DATA221,17,192,189,1,192,253,18
2230 DATA192,144,18,189,0,192,157,17
2240 DATA192,189,1,192,157,18,192,104
2250 DATA74,72,76,1,193,104,74,72,176
2260 DATA13,222,17,192,189,17,192,201
2270 DATA255,208,3,222,18,192,189,17
2280 DATA192,221,4,192,189,18,192,253,5
2290 DATA192,144,18,189,4,192,157,17
2300 DATA192,189,5,192,157,18,192,104
2310 DATA74,72,76,46,193,104,74,72,176
2320 DATA8,254,17,192,208,3,254,18,192
2330 DATA104,74,176,5,169,1,153,25,192
2340 DATA200,232,232,224,4,240,3,76,91
2350 DATA192,238,27,192,108,29,192
2360 DATA35994:REM PRUEFSUMME
2370 FORI=49183TO49478
2380 READX:POKEI,X:CC=CC+X
2390 NEXT
2400 READX:IFX<>CCTHEN PRINT"FEHLER IN D
ER PRUEFSUMME."
2410 RETURN
```

Eine vollständige Erklärung des Maschinenprogramms finden Sie in „Mastering the CBM 64" von A.J. Jones und G. Carpenter, ebenso veröffentlicht von Ellis Horwood.

8

Tonerzeugung

EINFÜHRUNG

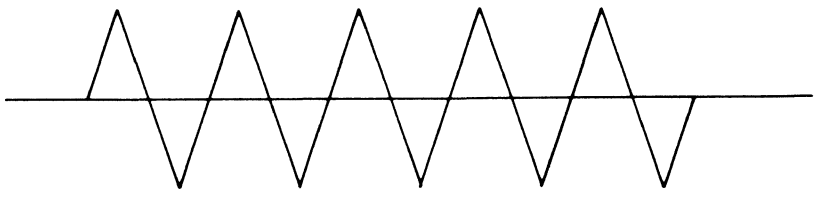
Das „Sound Interface Device" oder auch „SID"-Chip, der Tongenerator des Commodore 64, ist einer der höchstentwickelten Synthesizer, die in Homecomputern eingebaut werden. Mit viel Mühe kann man beinahe jeden beliebigen Ton erzeugen. Leider ist die Bedienung nicht gerade benutzerfreundlich. Die große Anzahl der notwendigen POKEs führt leicht zu einer völligen Verwirrung.

In diesem Kapitel wollen wir Ihnen einen kleinen Einblick in die Ansteuerung des „SID" geben.

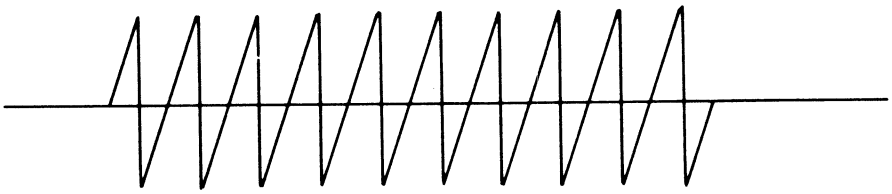
Der „SID" hat drei Oszillatoren mit einem Tonumfang von je sieben Oktaven (eine Notentabelle befindet sich in Anhang P des Handbuchs zum Commodore 64); ebenso besitzt jede Stimme eine Hüllkurve, die durch die ADSR-Register angesteuert werden kann: (ADSR: A=ATTACK (Anschlag), D=DECAY (Abschwingen), S=SUSTAIN (Halten), R=RELEASE (Ausklingen)); weiterhin sind vier verschiedene Wellenformen möglich, nämlich Dreieck, Sägezahn, Rechteck (. oder auch Puls) und Rauschen.

Wir wollen mit der Regelung der Lautstärke beginnen. Die zugehörige Speicherstelle ist 54296; die Lautstärke belegt jedoch nur die ersten vier Bits und kann daher Werte zwischen 0 und 15 annehmen.

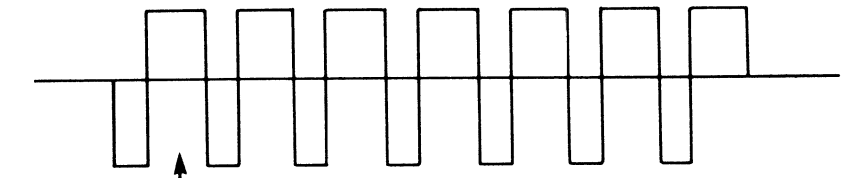
Als zweites haben wir die Wellenform, die über das Timbre oder auch die Qualität des Klangs entscheidet. Tafel 8.1 beschreibt die Unterschiede der einzelnen Formen wohl am besten.



Dreieckswelle



Sägezahnwelle



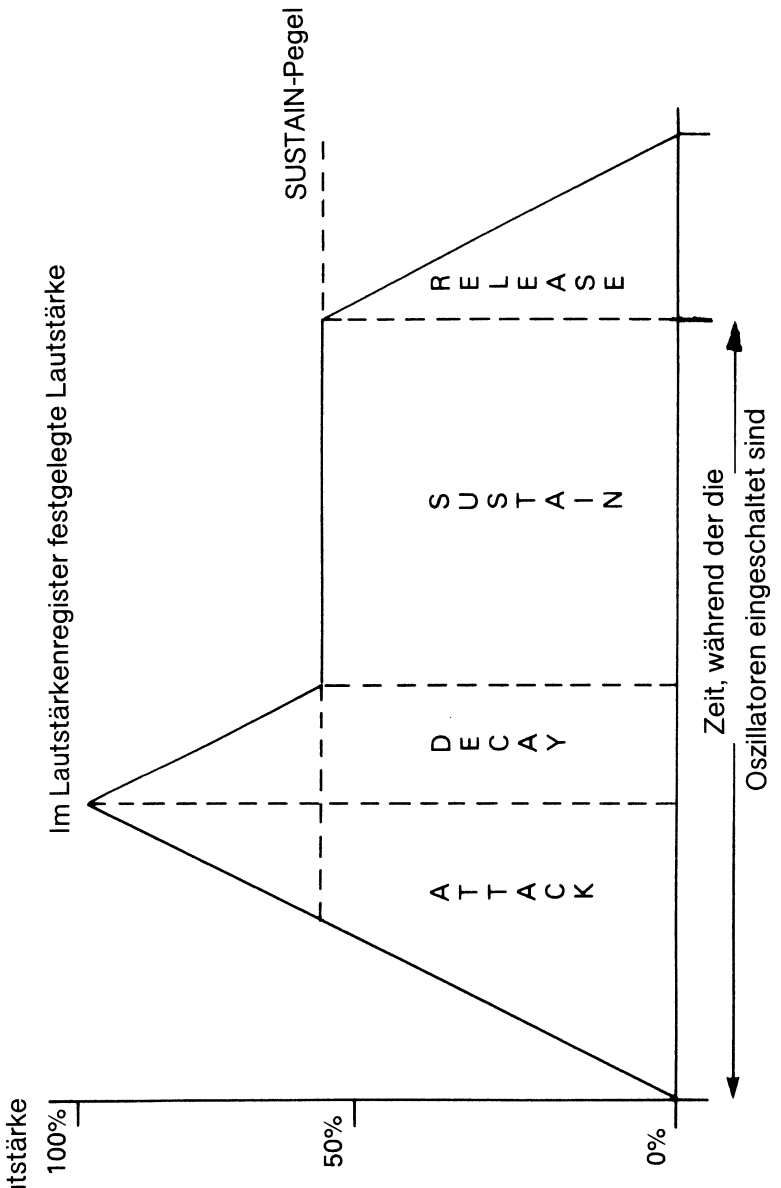
Variable Pulswelle
(Rechteckwelle)

Tafel 8.1

Drittens die sogenannten ADSR- oder auch Hüllkurvenregister: „ATTACK" bezeichnet die Geschwindigkeit, mit der der Ton zu seiner maximalen Lautstärke anschwillt; „DECAY" ist die Geschwindigkeit, mit der die Lautstärke auf die „SUSTAIN"-Lautstärke abfällt; „SUSTAIN" ist die Lautstärke, auf der der Ton gehalten wird; „RELEASE" schließlich ist die Geschwindigkeit, mit der die Lautstärke nach dem Ausschalten der Oszillatoren auf Null abfällt (siehe Tafel 8.2).

Als letztes und vielleicht wichtigstes existieren noch die Oszillatoren selbst. Zu diesen muß in jeweils zwei Speicherstellen die Frequenz (Tonhöhe) angegeben werden. Die Frequenzwerte müssen in folgende Speicherstellen gePOKEt werden:

Frequenz Oszillator 1:	Lo-Byte 54272
	Hi-Byte 54273
Frequenz Oszillator 2:	Lo-Byte 54279
	Hi-Byte 54280
Frequenz Oszillator 3:	Lo-Byte 54286
	Hi-Byte 54287


Lautstärke
100%
50%
0%
Im Lautstärkenregister festgelegte Lautstärke
SUSTAIN-Pegel
ATTACK
DECAY
SUSTAIN
RELEASE
Zeit, während der die Oszillatoren eingeschaltet sind

Tafel 8.2

Die Hauptschritte der Tonerzeugung sind also:

(a)	LAUTSTÄRKE POKEN	(1–15)
(b)	WELLENFORM POKEN	(17,33,65,129)
(c)	ATTACK/DECAY POKEN	(0–255)
(d)	SUSTAIN/RELEASE POKEN	(0–255)
(e)	HI-BYTE DER PULSWELLE (wenn Pulswelle verwendet)	(0–15)
(f)	LO-BYTE DER PULSWELLE (wenn Pulswelle verwendet)	(0–255)
(g)	HI-BYTE DER FREQUENZ	(0–255)
(h)	LO-BYTE DER FREQUENZ	(0–255)

Sobald eine Note ausgeklungen ist, müssen die ADSR-Register gelöscht werden, da andernfalls eigenartige Geräuscheffekte auftreten. Z.B. klingt eine eigentlich ausgeschaltete Note weiter, oder die Note klingt, wenn sie das nächste Mal gespielt wird, nicht besonders laut. Aber das werden Sie vielleicht schon selbst erlebt und die Ursachen erkannt haben.

Nun, da wir Ihnen den Mund wässerig gemacht haben, wollen wir etwas von dem verraten, was wir in der Praxis gelernt haben. Das erste Demo-Programm erzeugt einen Ton, dessen Länge durch eine Warteschleife festgelegt wird. Sie sehen, daß auch für einen einfachen Ton viel getan werden muß. Geben Sie das folgende Programm ein:

```
1000 S=54272        :REM BASIS-ADRESSE DES SID
1010 FORI=0TO24     :REM ————————————————————
1020 POKES+I,0      :REM SID IN AUSGANGSZUSTAND
1030 NEXTI          :REM ————————————————————
1040 POKES+24,15    :REM LAUTSTAERKE
1050 POKES+4,33     :REM WELLENFORM=SAEGEZAHN
1060 POKES+5,31     :REM ATTACK/DECAY
1070 POKES+6,15     :REM SUSTAIN/RELEASE
1080 POKES+1,51     :REM FREQUENZ HI-BYTE
1090 POKES+0,97     :REM FREQUENZ LO-BYTE
1100 FORI=1TO500    :REM ————————————————————
1110 NEXTI          :REM WARTESCHLEIFE
1120 POKES+4,0      :REM WELLENFORM LOESCHEN
1130 POKES+5,0      :REM ATTACK/DECAY LOESCHEN
1140 POKES+6,0      :REM SUSTAIN/RELEASE LOESCHEN
```

Lesen Sie sich das Programm, nachdem Sie es eingegeben haben, noch einmal genau durch und starten Sie es. Sie sollten dann einen Ton hören.

Wollen Sie Klang-Effekte in ein Programm einbauen, dann sollten Sie einen Großteil der Parameter setzen, bevor Sie die Hauptschleife beginnen. Gehen Sie vor wie in diesem Beispiel:

```
1000 S=54272        :REM BASISADRESSE DES SID
1010 FORI=0TO24 :REM ----------
1020 POKES+I,0      :REM SID IN AUSGANGSZUSTAND
1030 NEXTI          :REM ----------
1040 POKES+5,31     :REM ATTACK/DECAY
1050 POKES+6,15     :REM SUSTAIN/RELEASE
2000 REM ----------
2010 REM HAUPTSCHLEIFE DES PROGRAMMS
2020 REM ----------
3000 REM TONERZEUGUNGS-UNTERPROGRAMM
3010 POKES+4,33     :REM WELLENFORM=SAEGEZAHN
3020 POKES+1,25     :REM FREQUENZ HI-BYTE
3030 POKES+0,90     :REM FREQUENZ LO-BYTE
3040 FORI=1TO500:REM WARTE-
3050 NEXTI          :REM SCHLEIFE
3060 POKES+4,0      :REM WELLENFORM LOESCHEN
```

Wollte man näher auf die Einzelheiten der Tonerzeugung eingehen, dann sollte man besser ein eigenes Buch dafür schreiben; über die Grenzen dieses Buches geht das Thema auf jeden Fall hinaus. Zum Abschluß hier noch einige Beispielprogramme, die Sie in Programmen direkt verwenden oder nur analysieren können. Ihre so gewonnenen Kenntnisse können Sie zur Schöpfung eigener Effekte einsetzen. Anhang O des Handbuchs zum Commodore 64 enthält einige Schlüsselwerte zur Erzeugung von brillanten Klängen.

PROGRAMMBEISPIELE

```
1000 REM *** BEISPIEL1: EXPLOSION ***
1010 S=54272
1020 FORI=0TO24
1030 POKES+I,0
1040 NEXTI
1050 POKES+5,31
1060 POKES+6,240
1070 POKES+4,129
1080 FORV=15TO0STEP-.2
1090 POKES+24,V
1100 POKES+1,2
1110 POKES+0,90
1120 NEXTV
1130 POKES+4,0
```

```
1000 REM *** BEISPIEL 2: UFO ***
1010 S=54272
1020 FORI=0TO24
1030 POKES+I,0
1040 NEXTI
1050 POKES+24,15
1060 A=1
1070 POKES+5,240
1080 POKES+6,190
1090 POKES+4,17
1100 FORT=1TO100
1110 FORI=0TO50STEPA
1120 POKES+1,16+I+A
1130 POKES,200
1140 NEXTI
1150 A=A+.1
1160 IFA>49THENA=1
1170 NEXTT
1180 POKES+4,0
```

```
1000 REM *** BEISPIEL 3: ALARM ***
1010 S=54272
1020 FORI=1TO24
1030 POKES+I,0
1040 NEXTI
1050 POKES+24,15
1060 POKES+5,240
1070 POKES+6,190
1080 POKES+4,33
1090 FORT=1TO20
1100 FORI=0TO50
1110 POKES+1,16+I
1120 POKES+0,200
1130 NEXTI,T
1140 POKES+4,0
```

```
1000 REM *** BEISPIEL 4: FROSCH ***
1010 S=54272
1020 FORI=1TO24
1030 POKES+I,0
1040 NEXTI
1050 POKES+24,15
1060 POKES+5,47
1070 POKES+6,31
1080 FORI=0TO80STEP9
1090 POKES+4,0
1100 POKES+1,I
1110 POKES+0,20
1120 POKES+4,17
1130 NEXTI
1140 POKES+4,0
```

9

Tips und Tricks für Fortgeschrittene

EINIGE LEITSÄTZE

- Die Hauptschleife Ihres Programms sollte so kurz wie möglich sein; für Programme mit schnellen Bewegungsabläufen nicht länger als 20 Zeilen.
- Kennzeichnen Sie jeden Programmteil mit REM, ebenso jedes GOSUB.
- Beschleunigen Sie Ihr Programm, indem Sie Variablen und Konstanten anstelle von Zahlen verwenden.
- Bevor Sie mit der Programmierarbeit beginnen, erstellen Sie einen groben Ablaufplan und teilen Sie Ihr Programm in Blöcke ein.
- Verwenden Sie häufig, aber sinnvoll Unterprogramme.
- Speichern Sie Ihr Programm, bevor Sie es zum ersten Mal starten; das gilt besonders für noch unvollständige Programme.
- Bewegen Sie Zeichen auf dem Bildschirm mit PRINT und nicht mit POKE. Das ist zwar nicht so elegant, geht aber schneller.
- Erzählen Sie Ihren Freunden von diesem Buch.

- Verlassen Sie Ihre Unterprogramme immer mit RETURN!
- Springen Sie nie aus FOR...NEXT-Schleifen heraus, ohne das letzte NEXT passiert zu haben.
- Entscheiden Sie sich nicht immer für die erste Lösung, die Ihnen zu einem Problem einfällt. Durchdenken Sie das ganze Problem; es gibt vielleicht noch eine schnellere und/oder geschicktere Lösung.
- Vergessen Sie nicht, Ihren Freunden von diesem Buch zu erzählen.

INTERESSANTE POKES

Das LIST-Kommando wird folgendermaßen desaktiviert:

```
POKE 775,200
```

Dieses POKE aktiviert das LIST-Kommando wieder:

```
POKE 775,167
```

Um die RUN/STOP-Taste abzuschalten, tippen Sie:

```
POKE 808,239
```

Sie wird wieder eingeschaltet durch:

```
POKE 808,237
```

RUN/STOP- und RESTORE-Taste werden durch folgendes POKE außer Funktion gesetzt:

```
POKE 808,225
```

Reaktiviert werden sie durch:

```
POKE 808,235
```

Die Tastatur wird abgeschaltet durch:

```
POKE 649,0
```

Und wieder eingeschaltet durch:

```
POKE 649,10
```

Ein Wegblenden des Bildschirms wird erreicht durch:

```
POKE 53265,PEEK(53265)AND239
```

Der Bildschirm strahlt wieder in seinem alten Glanz durch:

```
POKE 53265,PEEK(53265)OR16
```

Ein Zeichen kann in Spalte X und Zeile Y mit PRINT ausgegeben werden:

```
POKE 782,X
POKE 781,Y
SYS 65520
```

Der Wert für X sollte zwischen 0 und 39, der für Y zwischen 0 und 24 liegen.

WEITERE PRAKTISCHE HINWEISE

Nach SYS 64738 führt der Commodore 64 einen Kaltstart durch, das heißt, der Computer wird in seinen Einschaltzustand zurückversetzt.

PRINT FRE(0)–(FRE(0)<0)*65536 gibt die momentane Zahl an freien BASIC-Bytes aus.

Der Bildschirm kann mit dieser Befehlsfolge abwärts gerollt werden:

```
PRINT CHR$(19)CHR$(17)CHR$(157)CHR$(148)
POKE 218,160
```

Ändern Sie einmal das POKE zu POKE 218, 158 und sehen Sie, was passiert.

DIE KOLLISIONSREGISTER

Löschen Sie ein Kollisionsregister immer, bevor Sie seinen Wert mit PEEK auslesen. Das geschieht mit POKE V+30 (oder V+31),0. Die Kollisionsregister werden zwar automatisch gelöscht; das geschieht häufig aber nicht schnell genug (siehe auch Kapitel 6). Denken Sie daran, daß V=53248.

Das folgende POKE schaltet zwei beliebige Sprites, die an einer Kollision beteiligt sind, aus:

```
POKE V+21,PEEK(V+21)AND(255–PEEK(V+30))
```

DIE RETURN-TASTE BEIM EDITIEREN VON PROGRAMMEN

Im Direkt-Modus kann man eine Zeile schnell verlassen, wenn man die Tasten SHIFT und RETURN gleichzeitig drückt. Die Zeile wird dadurch weder verändert noch ausgeführt.

Wenn Sie beim Editieren innerhalb von Anführungszeichen in Schwierigkeiten geraten, drücken Sie einfach SHIFT/RETURN, positionieren den Cursor wieder und fahren mit dem Schreiben fort.

PROBLEMFÄLLE

Alle Programme in diesem Buch wurden sorgfältig ausgetestet. Sie werden einwandfrei laufen, wenn sie so eingegeben werden, wie sie geLISTet sind. Wenn Sie ein Spiel nicht zum Laufen bringen, prüfen Sie das Listing genau. Das geht am einfachsten zu zweit: Bitten Sie einen Freund, Ihnen das gedruckte Listing vorzulesen, während Sie es mit dem auf dem Bildschirm vergleichen. Der häufigste Fehler ist „SYNTAX ERROR IN LINE XXX". Sie haben vermutlich einen simplen Tippfehler gemacht

(der am weitesten verbreitete ist eine fehlende Klammer). Wenn keine SYNTAX ERRORs mehr auftreten, das Programm aber immer noch nicht richtig arbeitet, haben Sie vielleicht eine DATA-Zeile vergessen. Die Fehlermeldung hierzu lautet „OUT OF DATA ERROR". Alle längeren Blocks von DATA-Zeilen wurden mit Prüfsummen versehen. Die Prüfsumme erhält man, wenn man alle Werte in den DATAs addiert. Bevor Sie Ihr Programm weiter austesten, sollten Sie hier alle Fehler beheben.

Es mag selbstverständlich klingen, dennoch sollte es hier gesagt sein: Nachdem Sie ein Programm eingegeben haben, speichern Sie es immer zuerst ab, dann starten Sie es. Das gilt besonders für Programme, die Maschinenspracheroutinen verwenden.

HILFS- UND DEMONSTRATIONSPROGRAMME

Hier sind noch einige Programme, die hilfreich sein können, wenn Sie Ihre eigenen Spiele auf dem Commodore 64 programmieren wollen. Besonders nützlich ist der „Sprite-Editor". Die anderen Programme beinhalten Möglichkeiten der String-Behandlung.

SPRITE-EDITOR

Dieser Sprite-Editor hilft Ihnen beim Entwerfen und Ablegen von Sprites in DATA-Zeilen.

Geben Sie das Programm ein und starten Sie es. Auf der linken Seite des Bildschirms sehen Sie ein Feld, in dem ein Cursor blinkt. Er kann mit Hilfe des Joysticks in dem Feld bewegt werden. Gelangt er an den Rand des Feldes, springt er automatisch zur anderen Seite.

Es gibt zwei Arbeitsmodi: einen zum Einfügen und einen zum Löschen. Im Einfügungsmodus wird jedesmal, wenn Sie den Feuerknopf drücken, unter dem Cursor ein Punkt gesetzt. Umgekehrt wird im Löschmodus der Punkt, der sich unter dem Cursor befindet, gelöscht. Die Rahmenfarbe zeigt an, in welchem Modus Sie sich gerade befinden: „Grün" steht für „Einfügen" und „Rot" für „Löschen".

Während Sie im großen Feld ein Muster des Sprites entwerfen, entsteht parallel dazu in dem kleinen Feld auf der rechten Seite das tatsächliche Sprite. Wenn Sie einen Punkt löschen, wird das entsprechende Pixel (engl. Abkürzung für „picture element") des Sprites ebenfalls gelöscht. Das Sprite ist in horizontaler und vertikaler Richtung vergrößert. Wenn Ihnen Ihr Entwurf ganz und gar nicht gefällt, drücken Sie einfach die CLR/HOME-Taste, und beide Felder werden gelöscht.

Nachdem Sie Ihr Sprite entworfen haben, drücken Sie die „F1"-Taste, und die Daten des Sprites werden ausgegeben. (Vergewissern Sie sich, daß Sie keine ROM-Erweiterung eingesteckt haben, die diese Taste anders belegt.) Schreiben Sie sich die Daten ab und drücken Sie den Feuerknopf, um das nächste Sprite zu entwerfen.

Achtung! Lassen Sie sich die Sprite-Daten erst anzeigen, wenn Sie mit dem Entwurf zufrieden sind. Das Programm sieht keine Möglichkeit vor, einmal umgerechnete Sprites zu verändern.

Der Sprite-Editor hat uns unschätzbare Dienste geleistet, als wir dieses Buch geschrieben haben; wir hoffen, daß Sie mit ihm genauso viel Freude haben werden wie wir.

```
1000 REM *** CBM 64  SPRITE-EDITOR ***
1010 PRINTCHR$(147)
1020 POKE53280,5:POKE53281,0
1030 S$="▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
1040 D$="▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌"
1050 FORI=0TO7:C%(I)=2^(7-I):NEXT
1060 SC=1105:CO=55377
1070 GOSUB1450:REM BILDSCHIRM AUFBAUEN
1080 REM *** HAUPTSCHLEIFE ***
1090 K=PEEK(197):IFK=51THENGOSUB1450
1100 IFK=4THENGOSUB1270
1110 IFK=14THENF=0:REM 'EINFUEGEN'
1120 IFK=42THENF=1:REM 'LOESCHEN'
1130 POKESC+40*Y+X,M1
1140 P=PEEK(56320):REM PORT 2
1150 IFP=126THENY=Y-1:IFY<0THENY=20
1160 IFP=125THENY=Y+1:IFY>20THENY=0
1170 IFP=123THENX=X-1:IFX<0THENX=23
1180 IFP=119THENX=X+1:IFX>23THENX=0
1190 IFF=0ANDP=111THENPOKESC+X+40*Y,160:
GOSUB1840
1200 IFF=1ANDP=111THENPOKESC+X+40*Y,79:G
OSUB1890
1210 IFF=0THENPOKE53280,5:REM GRUENER RA
HMEN
1220 IFF=1THENPOKE53280,2:REM ROTER RAHM
EN
1230 M2=PEEK(SC+X+40*Y):M1=M2
1240 POKECO+X+40*Y,14:POKESC+X+40*Y,81
1250 FORL=1TO50:NEXT
1260 GOTO1090
1270 REM *** DATEN ERRECHNEN ***
1280 PRINTCHR$(147):FORL=1TO3
1290 PRINTCHR$(17):NEXT
1300 PRINT"     ▁▁▁▁▁SPRITE.▁▁▁▁▁";
1310 PRINTCHR$(18)"◥"CHR$(146)
1320 PRINT"     ────────────────────◤"
1330 PRINT:PRINT:PRINT
1340 FORD=0TO8
1350 PRINT" DATA";
```

```
1360 FORI=832+(D*7)TO838+(D*7):P=PEEK(I)
1370 PRINTP;CHR$(157)CHR$(44)" ";
1380 PRINTCHR$(157);:NEXT:PRINTCHR$(157)
" ":NEXT
1390 PRINT
1400 PRINTCHR$(18)TAB(3)"◤FEUERKNOPF";
1410 PRINT" FUER NAECHSTES SPRITE◥"
1420 IFPEEK(56320)<>111THEN1420
1430 GOSUB1450:RETURN
1440 REM *** AUSGABE ***
1450 M1=79
1460 PRINTCHR$(147)CHR$(30)CHR$(18)"    *
**  CBM ";
1470 PRINT"64   -   SPRITE-EDITOR   ***    "
;
1480 PRINTCHR$(146)CHR$(154)" ▬▬▬▬▬▬▬▬";
1490 PRINT"▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬ "
1500 FORT=1TO21
1510 PRINTCHR$(18)"▌"CHR$(146)"┌┬┬┬┬┬┐";
1520 PRINT"└┴┴┴┴┴┴┴┴┴┴┴┴┴┴┴┴┴┴▌"
1530 NEXT
1540 PRINT" ▪"CHR$(18)"▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬";
1550 PRINT"▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬"CHR$(146)"▪ "
1560 PRINTCHR$(30)CHR$(18)"          Ä F1
FUER";
1570 PRINT" DATENAUSGABE Ü          ";
1580 PRINTCHR$(19)
1590 X=0:Y=0:POKE56295,5:POKE2023,160
1600 PRINTS$;D$;CHR$(18)"E"CHR$(146)"=EI
NFUEGEN"
1610 PRINTS$;D$;RIGHT$(S$,1)" (GRUEN)"
1620 PRINTS$;D$;RIGHT$(S$,2);CHR$(18)"L"
CHR$(146)"=LOESCHEN"
1630 PRINTS$;D$;RIGHT$(S$,3)" (ROT)"
1640 PRINTS$;D$;RIGHT$(S$,4);CHR$(18)"CL
R"CHR$(146)"=NEU "
1650 PRINTS$;D$;RIGHT$(S$,5)"FEUERKNOPF"
1660 PRINTS$;D$;RIGHT$(S$,6)"FUER E UND
Z"
1670 K$=LEFT$(S$,3)+LEFT$(D$,26)
1680 PRINTK$;CHR$(145)"* SPRITE *":PRINT
K$;
1690 PRINT" ┌──────────┐"
1700 FORK=1TO11:PRINTLEFT$(D$,26);
1710 PRINT"|          |":NEXT
1720 K$=LEFT$(S$,14)+LEFT$(D$,26)
```

```
1730 PRINTK$"└──────────┘"
1740 VC=53248
1750 POKEVC+21,1:REM SPRITE 0 ANSCHALTEN
1760 POKE2040,13:REM POINTER AUF 13*64=8
32 SETZEN
1770 POKEVC,5:REM X-KOORDINATE
1780 POKEVC+16,1:REM MSB
1790 POKEVC+1,95:REM Y-KOORDINATE
1800 POKEVC+39,5:REM GRUEN
1810 POKEVC+29,1:POKEVC+23,1:REM VERGROE
SSERN
1820 FORI=0TO62:POKE832+I,0:NEXT
1830 RETURN
1840 REM *** PIXEL IM SPRITE SETZEN ***
1850 TH=INT(X/8):BI=X-8*TH
1860 BY=832+3*Y+TH
1870 POKEBY,PEEK(BY)ORC%(BI)
1880 RETURN
1890 REM *** PIXEL IM SPRITE LOESCHEN **
*
1900 TH=INT(X/8):BI=X-8*TH
1910 BY=832+3*Y+TH
1920 POKEBY,PEEK(BY)AND(255-C%(BI))
1930 RETURN
```

Dieses Programm wandelt den normalen Zeichensatz in einen des Raumfahrerzeitalters um.

```
1000 REM *** ZEICHENSATZ ***
1010 REM *** SPEICHEROBERGRENZE HERABSET
ZEN ***
1020 POKE52,48:POKE56,48:CLR
1030 FORI=0TO511
1040 POKEI+12288,PEEK(I+53248):NEXT
1050 REM *** ALPHABET ***
1060 FORALPHA=12288TO12503:READX:AA=AA+X
:POKEALPHA,X:NEXTALPHA
1070 READX:IFX<>AATHENPRINT"FEHLER IM AL
PHABET":END
1080 REM *** ZAHLEN ***
1090 FORNUM=12672TO12751:READZ:BB=BB+Z:P
OKENUM,Z:NEXTNUM
1100 READZ:IFZ<>BBTHENPRINT"FEHLER IN DE
N ZAHLEN":END
1110 REM *** POINTER SETZEN ***
1120 POKE53272,(PEEK(53272)AND240)+12
1130 REM *** ALPHABET-DATEN ***
1140 DATA0,0,0,0,0,0,0,0
1150 DATA252,132,132,254,134,134,134,0
1160 DATA252,132,132,254,134,134,254,0
1170 DATA252,128,128,128,128,128,254,0
1180 DATA240,136,132,134,134,134,254,0
1190 DATA248,128,128,252,128,128,254,0
1200 DATA254,128,128,252,128,128,128,0
1210 DATA252,128,128,134,134,134,254,0
1220 DATA132,132,132,254,134,134,134,0
1230 DATA16,16,16,56,56,56,56,0
1240 DATA254,16,16,24,24,24,248,0
1250 DATA128,132,136,240,140,140,140,0
1260 DATA128,128,128,128,128,128,254,0
1270 DATA204,180,132,134,134,134,134,0
1280 DATA132,132,228,158,134,134,134,0
1290 DATA252,132,132,134,134,134,254,0
1300 DATA254,130,254,128,128,128,128,0
1310 DATA252,132,132,150,142,134,254,1
1320 DATA254,130,252,134,134,134,134,0
1330 DATA248,128,252,6,6,6,254,0
1340 DATA254,16,16,24,24,24,24,0
1350 DATA132,132,132,134,134,134,254,0
```

```
1360 DATA132,132,196,70,102,38,62,0
1370 DATA132,132,132,150,150,150,254,0
1380 DATA68,68,40,16,40,68,68,0
1390 DATA132,68,52,14,6,6,6,0
1400 DATA126,2,30,32,64,128,254,0
1410 REM *** PRUEFSUMME ***
1420 DATA24607
1430 REM *** ZAHLEN-DATEN ***
1440 DATA124,68,76,86,102,70,126,0
1450 DATA16,16,16,24,24,24,24,0
1460 DATA254,2,2,2,126,64,126,0
1470 DATA28,4,4,62,6,6,254,0
1480 DATA132,132,132,254,6,6,6,0
1490 DATA252,128,128,128,248,8,248,0
1500 DATA128,128,128,252,132,132,252,0
1510 DATA252,4,4,6,6,6,6,0
1520 DATA120,72,252,134,134,134,254,0
1530 DATA252,132,252,6,6,6,6,0
1540 REM *** PRUEFSUMME ***
1550 DATA6740
```

```
1000 REM *** STRING DEMO 1 ***
1010 PRINTCHR$(147)CHR$(30)
1020 POKE53280,0:POKE53281,0
1030 A$(1)="ATTACK COMPUTER ON"
1040 A$(2)="PHASERS LOCKED ON"
1050 A$(3)="TARGET...."
1060 A$(4)="SHIELDS ON MAXIMUM"
1070 A$(5)="ENERGY SUPPLY 10.547"
1080 A$(6)="ENGINES AT WARP .003"
1090 A$(7)="ENEMY IN SECTOR"
1100 A$(8)="CURRENT STATUS NOMINAL"
1110 A$(9)="DIRECT HIT..."
1120 A$(10)="AWAIT COMMAND INPUT"
1130 FORL=1TO10
1140 FORT=1TO22
1150 FORA=1TO30:NEXT
1160 PRINTLEFT$(A$(L),T)
1170 PRINTCHR$(145)CHR$(145):NEXT
1180 PRINT:PRINT:NEXT
```

```
1000 REM *** STRING DEMO 2 ***
1010 PRINTCHR$(147)
1020 FORX=18TO1STEP-1
1030 READA$
1040 IFA$="*"THEN1100
1050 IFA$="§"THENGOSUB1180:END
1060 FORL=0TOX
1070 PRINTTAB(L+9)" "A$
1080 PRINTCHR$(145)CHR$(145)
1090 NEXTL,X
1100 PRINT:PRINT:GOSUB1180
1110 GOTO1020
1120 DATA, ,T,S,I, ,T,R,O,W, ,S,E,D,E,J,
*
1130 DATA, ,N,E,L,I,E,Z,-,A,T,A,D, ,N,I,
*
1140 DATA, , ,D,N,U, ,T,G,E,L,E,G,B,A, ,
*
1150 DATA, ,E,S,E,I,D, ,F,U,A, ,N,N,A,K,
*
1160 DATA ,N,E,B,E,G,E,G,S,U,A, ,E,S,I,E
,W,*
1170 DATA, , , , , ,N,E,D,R,E,W,§
1180 FORT=1TO50:NEXT:RETURN
```

```
1000 REM *** STRING DEMO 3 ***
1010 PRINTCHR$(147)
1020 PRINTTAB(10)"┌──────────────────────────┐"
1030 PRINTTAB(10)"│                          │"
1040 PRINTTAB(10)"└──────────────────────────┘"
1050 A$="  COMMODORE * 64  "
1060 PRINTCHR$(19)CHR$(17)
1070 C$=RIGHT$(A$,17)+LEFT$(A$,1)
1080 A$=C$
1090 PRINTTAB(11)A$
1100 FORI=1TO50:NEXT
1110 GOTO1060
```

Abschnitt 2

Listings der Spiele

So viel zur Theorie; jetzt kommt das, worauf Sie gewartet haben. Wir hoffen, daß Sie beim Spielen mit den Programmen, die wir für Sie geschrieben haben, viel Spaß haben werden. Wir jedenfalls hatten viel Spaß beim Schreiben. Wenn Sie die Programme selbst eingeben, halten Sie sich exakt an die Listings. Speichern Sie die Programme immer erst ab, bevor Sie sie starten.

Wir haben uns bemüht, die Programme übersichtlich und gut strukturiert zu gestalten, so daß keine Verständnisprobleme auftreten. Die Verwendung der Commodore-Steuerzeichen haben wir vermieden und sie durch die entsprechenden CHR$-Codes ersetzt. Die einzige Ausnahme von dieser Regel ist der Gebrauch der RVS ON- und RVS OFF-Steuerzeichen innerhalb von PRINT-Befehlen: RVS ON wird als inverses „R" dargestellt, RVS OFF als Quadrat, durch das ein diagonaler Balken läuft. Sie werden im Anführungszeichenmodus durch gleichzeitiges Drücken der Tasten CTRL und 9 bzw. CTRL und 0 erzeugt. Alle Grafikzeichen sind auf der Vorderseite der Tasten dargestellt. Das rechte Zeichen erreichen Sie, indem Sie die entsprechende Taste und gleichzeitig SHIFT drücken. Das linke Zeichen wird durch gleichzeitiges Drücken der entsprechenden Taste und der Taste mit dem Commodore-Symbol erzeugt.

Das Eingeben langer Programme kann ermüdend sein; vergessen Sie also nicht, daß Sie alle Programme dieses Buches, sowohl die Spiele als auch die Demonstrationsprogramme des ersten Abschnitts, auf Diskette oder Kassette beim Verlag bestellen können.

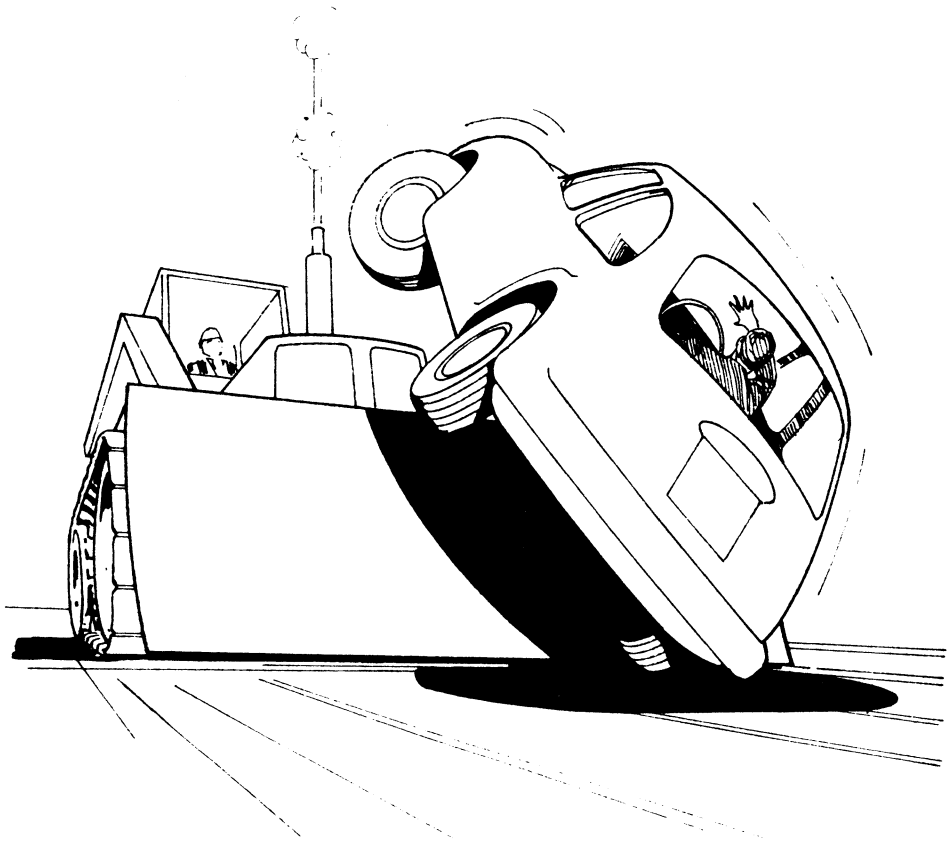
Die meisten unserer Programme benötigen einen Joystick, der an den CONTROL PORT 2 angeschlossen ist. Dieser befindet sich an der rechten Seitenwand des Computers neben dem ON/OFF-Schalter.

Und jetzt

V I E L S P A S S!

Demon Dozer

(Geisterfahrender Bulldozer)



Der Demon Dozer kriegt Sie am Ende doch; und wenn er es nicht schafft, dann schafft's die Leitplanke bestimmt.

Für dieses Spiel brauchen Sie ein schnelles Auge und viel Geschicklichkeit. Nachdem Sie das Programm gestartet haben, erscheint eine Straße, in deren Mitte ein blaues Auto fährt, in dem Sie sitzen. Plötzlich taucht aus dem Nichts ein Bulldozer auf, der versucht, Sie zu rammen. Sie müssen ihm möglichst lange ausweichen. Wenn Sie einmal angefangen haben, Ihr Auto zu bewegen, bewegt es sich selbständig weiter – das erschwert Ihre Aufgabe zusätzlich.

Versuchen Sie, nachdem Sie das Spiel einige Male gespielt haben, die Geschwindigkeit des Bulldozers oder die Ihres Autos zu verändern.

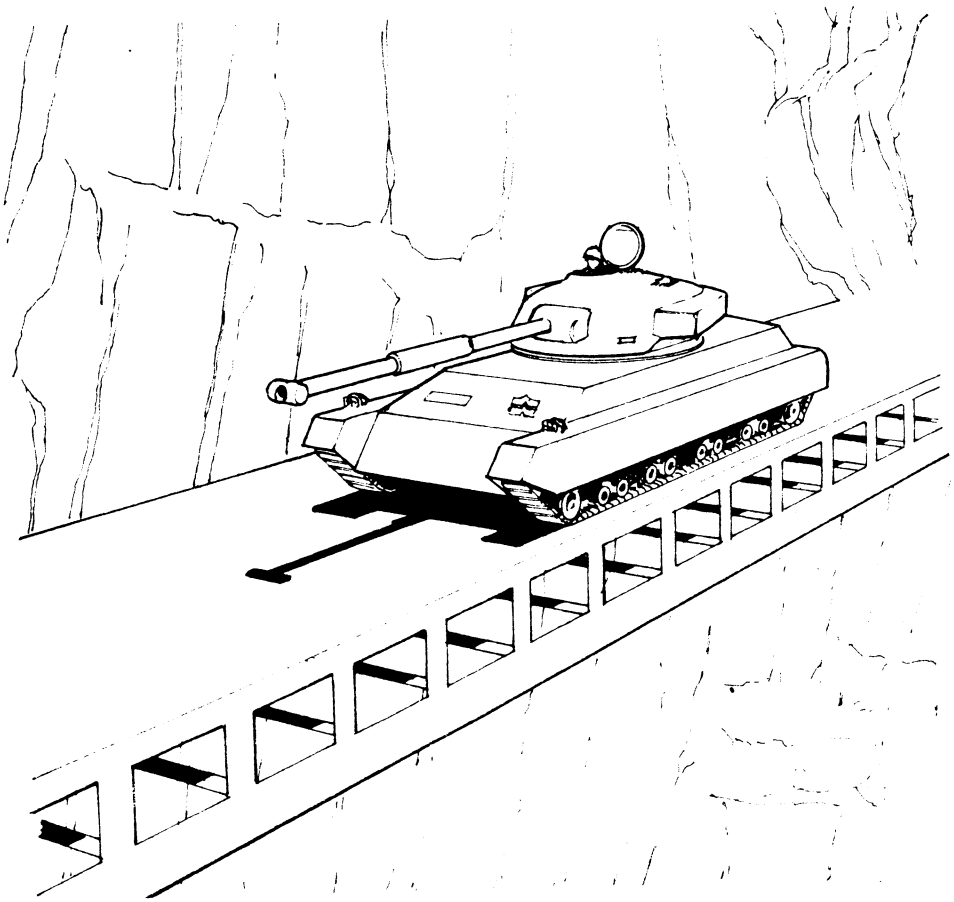
```
1000 REM *** DEMON DOZER ***
1010 PRINTCHR$(147)
1020 L$="":FORI=1TO4:L$=L$+CHR$(157):NEX
T
1030 POKE53280,0:POKE53281,0
1040 GOSUB1820:REM TITELBILD
1050 REM *** SPRITE-DATEN LESEN ***
1060 FORI=0TO62:READJ:POKE832+I,J:NEXT
1070 FORI=0TO62:READJ:POKE896+I,J:NEXT
1080 FORI=0TO62:READJ:POKE960+I,J:NEXT
1090 PRINTCHR$(147)
1100 REM *** SPRITES INITIALISIEREN ***
1110 POKE2040,13:VC=53248:POKEVC+21,5
1120 POKE2041,14:POKE2042,15:POKEVC+40,8
1130 POKEVC+39,14:POKEVC+23,7:POKEVC+29,
7
1140 POKEVC+0,150:POKEVC+1,180
1150 POKEVC+28,2:POKEVC+41,10
1160 REM *** TONREGISTER SETZEN ***
1170 S=54272:FORI=0TO24:POKES+I,0:NEXT
1180 S1=54286:POKES+5,31:POKES+6,240
1190 POKES+24,15:POKES1+5,31:POKES1+6,24
0
1200 L=12:X=150:W=12:Z=0:SE=0
1210 POKE646,14
1220 PRINTCHR$(19)CHR$(5)
1230 FORI=1TO21
1240 PRINTTAB(9)"▗ "SPC(16)" ▖"
1250 NEXT:POKEVC+31,0:POKEVC+30,0
1260 POKES+4,33:POKES,100:POKES+1,4
1270 POKES1+4,33:POKES1,100:POKES1+1,5
1280 PRINTCHR$(19)CHR$(30)
1290 FORI=1TO21
1300 PRINTCHR$(18)"          "SPC(20);
1310 PRINT"          ":REM JE 9 LEERZEICH
EN
1320 NEXTI
1330 PRINTCHR$(19)CHR$(5)"PUNKTE"
1340 PRINTCHR$(19)TAB(31)"MAX"
1350 REM *** HAUPTSCHLEIFE ***
1360 IFCR=0THENGOSUB1580
1370 SE=SE+1
1380 CY=CY+16:IFCY>210THENCY=1:CR=0
1390 P=PEEK(56320)
1400 IFP=123THENZ=1
```

```
1410 IFP=119THENZ=2
1420 ONZGOSUB1530,1550
1430 POKEVC+0,X
1440 POKEVC+4,R:POKEVC+5,CY
1450 PRINTCHR$(19)TAB(7)"    "L$SE
1460 IFSE>HITHENHI=SE
1470 PRINTCHR$(19)TAB(34);HI
1480 IFPEEK(VC+31)AND1=1THEN1600
1490 IFPEEK(VC+30)AND1=1THEN1600
1500 POKEVC+31,0:POKEVC+30,0
1510 GOTO1360
1520 REM *** BEWEGUNGSRICHTUNG DES AUTOS
 ***
1530 X=X-5:IFX<10THENX=10
1540 RETURN
1550 X=X+5:IFX>250THENX=249
1560 RETURN
1570 REM *** NEUE X-POSITION DES BULLDOZ
ERS ***
1580 R=INT(RND(TI)*84+110):CR=1:RETURN
1590 REM *** ZUSAMMENSTOSS MIT LEITPLANK
E ***
1600 POKEVC+21,6:POKES1+4,0
1610 POKEVC+2,X:POKEVC+3,180
1620 FORV=15TO0STEP-1:POKES+1,2+V
1630 POKE53270,INT(RND(1)*8)
1640 POKES+4,129:POKES+24,V
1650 POKEVC+37,INT(RND(TI)*15)
1660 POKEVC+38,INT(RND(TI)*15)
1670 NEXT:POKES+4,0
1680 POKEVC+21,0
1690 REM *** NEUES SPIEL ***
1700 PRINTCHR$(19):CY=1:CR=0
1710 POKE646,INT(RND(TI)*15)
1720 FORI=1TO4:PRINTCHR$(17):NEXT
1730 POKE53270,200
1740 PRINTTAB(14)"FEUERKNOPF"
1750 PRINTTAB(15)"DRUECKEN"
1760 POKEVC+4,R:POKEVC+5,CY
1770 IFPEEK(56320)<>111THEN1700
1780 PRINTCHR$(145)CHR$(145)CHR$(145)
1790 PRINTTAB(13)"              ":REM 11
1800 PRINTTAB(12)"                ":REM 13
1810 GOTO1110
1820 REM *** TITELBILD ***
1830 PRINTCHR$(19)CHR$(5)
```

```
1840 PRINTTAB(7)"< D E M O N * ";
1850 PRINT"D O Z E R >"
1860 PRINTCHR$(17)CHR$(17)
1870 PRINT"  STEUERE DEIN AUTO ";
1880 PRINT"MIT DEM JOYSTICK"
1890 PRINT"  UND WEICHE DEM ";
1900 PRINT"BULLDOZER AUS!"
1910 PRINT"  HUETE DICH VOR ";
1920 PRINT"DER LEITPLANKE..."
1930 PRINTCHR$(17)CHR$(17)CHR$(17)
1940 PRINTTAB(4)"ZUM SPIELBEGINN TASTE D
RUECKEN"
1950 GETA$:IFA$=""THEN1950
1960 RETURN:REM SPIELBEGINN
1970 REM *** SPRITE 0 (AUTO) ***
1980 DATA0,0,0,7,129,224,7,189,224,7
1990 DATA126,224,6,255,96,6,255,96,7
2000 DATA255,224,4,0,32,6,0,96,5,255
2010 DATA160,5,255,160,5,255,160,7,255
2020 DATA224,5,255,160,5,255,160,6,0,96
2030 DATA4,0,32,7,255,224,7,255,224,7
2040 DATA255,224,6,255,96
2050 REM *** SPRITE 1 (EXPLOSION) ***
2060 DATA0,32,0,2,73,52,6,157,193,104
2070 DATA188,29,131,54,124,44,118,249
2080 DATA65,231,242,31,227,178,78,8,57
2090 DATA39,63,156,47,159,30,73,222,124
2100 DATA36,194,240,73,248,198,19,253
2110 DATA216,71,205,144,23,15,36,16,103
2120 DATA72,13,146,80,0,8,144,3,87,32
2130 REM *** SPRITE 2 (BULLDOZER) ***
2140 DATA0,10,0,12,10,48,13,255,176,15
2150 DATA255,240,12,255,48,13,189,176,1
2160 DATA219,128,1,255,128,5,0,160,7,36
2170 DATA224,7,129,224,5,255,160,1,255
2180 DATA128,0,129,0,0,129,0,7,255,224
2190 DATA8,0,16,0,0,0,0,0,0,0,0,0,0,0,0
```

3D Tank Assault

(Panzerschlacht)



In diesem Spiel liegen Sie in einem Hinterhalt und versuchen, so viele feindliche Panzer wie möglich zu zerstören. Halten Sie sie mit den zur Verfügung stehenden Granaten auf und verhindern Sie, daß sie sich am anderen Ende der Brücke in Sicherheit bringen.

Die Landschaft wird in diesem Spiel sehr effektvoll durch Grafikzeichen dargestellt.

Der Panzer und das Fadenkreuz sind Sprites. Wie bei den meisten Spielen in diesem Buch wird auch hier das Maschinenprogramm zur Joystickabfrage verwendet.

```
1000 REM *** 3D TANK ASSAULT ***
1010 PRINTCHR$(147):POKE53280,0:POKE5328
1,0
1020 POKE54296,15
1030 POKE53269,3:POKE2040,13:POKE2041,14
1040 FORI=0TO62:READJ:POKE832+I,J:NEXT:R
EM FADENKREUZ
1050 FORI=0TO62:READJ:POKE896+I,J:NEXT:R
EM PANZER
1060 FORI=49183TO49478:READJ:POKEI,J
1070 CC=CC+J:NEXTI:READJ
1080 IFJ<>CCTHENPRINT"FEHLER IN DEN DATE
N":END
1090 SYS49183
1100 POKE53277,PEEK(53277)OR(2^1):REM PA
NZER VERGROESSERN
1110 POKE53250,250:POKE53251,150
1120 POKE53287,7:POKE53288,0:S=150:J=150
1130 SE=0:SH=99:R$=CHR$(18):S$=CHR$(19)
1140 SP$="                              ":REM 20 L
EERZEICHEN
1150 GOSUB1630:REM BILDSCHIRM AUFBAUEN
1160 REM JOYSTICK-PARAMETER SETZEN
1170 POKE49152,85:POKE49153,0
1180 POKE49156,250:POKE49157,0
1190 POKE49160,100:POKE49161,0
1200 POKE49164,200:POKE49165,0
1210 POKE49177,0:POKE49168,0
1220 POKE49169,150:POKE49173,100
1230 POKE53281,4
1240 REM HAUPTSCHLEIFE
1250 T=T-4:IFT<40THENT=255
1260 POKE53250,T
1270 PRINTS$"PUNKTE:"SE:PRINTS$;SPC(15)"
MAX:"HI
1280 PRINTS$;SPC(27)"GRANATEN:"SH;CHR$(1
57)" "
1290 IFSE>HITHENHI=SE
1300 X=PEEK(49169):Y=PEEK(49173)
1310 IFSH<1THEN1570
1320 POKE53248,X:POKE53249,Y
1330 F=PEEK(49177)
1340 POKE49177,0
1350 IFF=1THENSH=SH-1
1360 IFF=1ANDX=T+14ANDY=150THENGOSUB1390
```

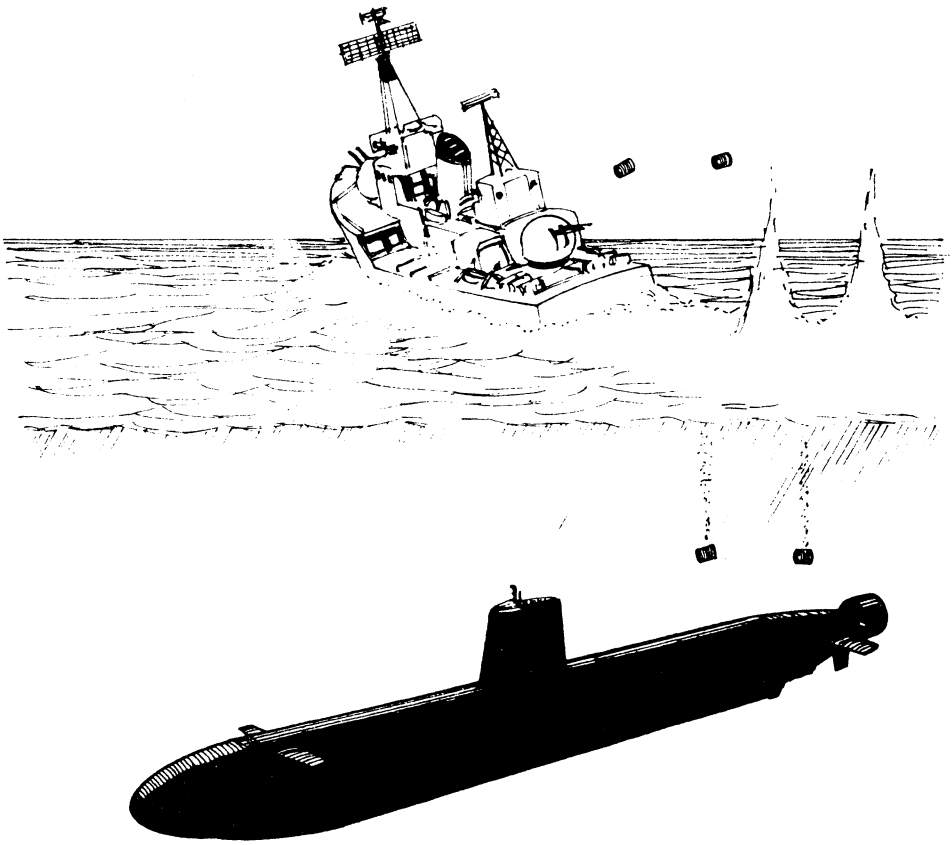
```
1370 POKE49177,0
1380 GOTO1250
1390 REM PANZER EXPLODIERT
1400 POKE54296,15:SE=SE+1*SH:G=1
1410 SH=SH-1
1420 POKE54276,0:POKE54290,0
1430 POKE54276,33:POKE54290,33
1440 POKE54277,31:POKE54291,31
1450 POKE54278,240:POKE54292,240
1460 FORL=255TO18STEP-3
1470 POKE54273,L-7:POKE54287,L
1480 POKE54272,200:POKE54286,200
1490 POKE53288,L:NEXTL
1500 POKE54276,129:POKE54290,129
1510 FORV=15TO0STEP-1:POKE53296,3
1520 POKE54273,2:POKE54287,200+V
1530 POKE54272,90:POKE54286,90
1540 POKE54296,V:POKE53269,1:NEXT:T=255
1550 POKE54276,0:POKE54290,0:POKE53250,T
1560 POKE53288,0:POKE53269,3:RETURN
1570 REM *** SPIELENDE ***
1580 PRINTCHR$(19)CHR$(17)CHR$(17)CHR$(5
)
1590 PRINT"  TASTE DRUECKEN FUER EIN NEU
ES SPIEL"
1600 GETA$:IFA$=""THEN1600
1610 PRINTCHR$(147):POKE53281,0:SH=99:SE
=0
1620 GOTO1150
1630 REM BILDSCHIRM AUFBAUEN (JE 1 LEERZ
EICHEN)
1640 PRINTCHR$(147);:FORT=1TO5:PRINTCHR$
(17):NEXT:PRINTCHR$(144)
1650 PRINTSPC(39);R$" ";
1660 PRINTSPC(37);R$;LEFT$(SP$,3);
1670 PRINTR$" "SPC(33);R$;LEFT$(SP$,6);
1680 PRINTR$;LEFT$(SP$,2);CHR$(127);SPC(
29);R$;CHR$(190);LEFT$(SP$,7);
1690 PRINTR$;LEFT$(SP$,4);SPC(27);R$;LEF
T$(SP$,9);
1700 PRINTR$;LEFT$(SP$,4);CHR$(127);SPC(
25);R$;CHR$(169);LEFT$(SP$,9);
1710 PRINTR$;LEFT$(SP$,6);CHR$(127);SPC(
22)R$;CHR$(191);LEFT$(SP$,10);
1720 PRINTR$;LEFT$(SP$,7);CHR$(172);SPC(
20)R$;CHR$(169)LEFT$(SP$,11);
```

```
1730 PRINTR$;LEFT$(SP$,9);:FORT=1TO9
1740 PRINTCHR$(162);R$" ";:NEXT:PRINTR$;
LEFT$(SP$,12);
1750 PRINTR$;SP$;SP$;
1760 PRINTR$;LEFT$(SP$,10);CHR$(127);SPC
(14);R$;CHR$(169);LEFT$(SP$,14);
1770 PRINTR$;LEFT$(SP$,12);CHR$(188);SPC
(9);R$;CHR$(187);LEFT$(SP$,17);
1780 PRINTR$;LEFT$(SP$,12);CHR$(172);SPC
(7)R$;SP$;
1790 PRINTR$;LEFT$(SP$,12);SPC(10);R$;CH
R$(187);LEFT$(SP$,17);
1800 PRINTR$;LEFT$(SP$,13);CHR$(172);SPC
(5);R$;SP$;" ";
1810 PRINTR$;LEFT$(SP$,15);CHR$(127);SPC
(2);R$;CHR$(169);SP$;" ";
1820 PRINTR$;SP$;SP$;
1830 PRINTR$;SP$;LEFT$(SP$,19);
1840 FORT=1TO10:R=INT(RND(TI)*280):POKE5
5296+R,1:POKE1024+R,46:NEXT
1850 POKE2022,160:POKE56295,0
1860 POKE2023,160:POKE56294,0
1870 POKE55463,1:POKE1191,81
1880 RETURN
1890 REM SPRITE-DATEN FUER FADENKREUZ
1900 DATA255,255,255,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0
1910 DATA0,24,0,0,24,0,0,24,0,0,0,0,7,23
1,224,0,0,0,0,24,0,0,24,0,0,24,0
1920 DATA0,0,0,0,0,0,0,0,0,0,0,0,255,255
,255
1930 REM SPRITE-DATEN FUER PANZER
1940 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0
1950 DATA0,8,0,0,31,0,0,63,128,63,255,19
2,0,127,192,0,127,192
1960 DATA0,63,0,3,255,192,6,172,160,11,2
55,240
1970 DATA15,255,208,5,85,96,3,255,192
1980 REM MS-PGR-DATEN
1990 DATA120,173,20,3,141,29,192,173,21
2000 DATA3,141,30,192,169,71,141,20,3
2010 DATA169,192,141,21,3,88,96,120,173
2020 DATA29,192,141,20,3,173,30,192,141
2030 DATA21,3,88,96,173,27,192,205,16
```

```
2040 DATA192,240,3,76,65,193,169,255
2050 DATA141,27,192,162,0,160,0,185,2
2060 DATA220,141,28,192,169,224,153,2
2070 DATA220,185,0,220,72,173,28,192
2080 DATA153,2,220,189,8,192,221,21,192
2090 DATA189,9,192,253,22,192,144,18
2100 DATA189,8,192,157,21,192,189,9,192
2110 DATA157,22,192,104,74,72,76,162
2120 DATA192,104,74,72,176,13,222,21
2130 DATA192,189,21,192,201,255,208,3
2140 DATA222,22,192,189,21,192,221,12
2150 DATA192,189,22,192,253,13,192,144
2160 DATA18,189,12,192,157,21,192,189
2170 DATA13,192,157,22,192,104,74,72,76
2180 DATA207,192,104,74,72,176,8,254,21
2190 DATA192,208,3,254,22,192,189,0,192
2200 DATA221,17,192,189,1,192,253,18
2210 DATA192,144,18,189,0,192,157,17
2220 DATA192,189,1,192,157,18,192,104
2230 DATA74,72,76,1,193,104,74,72,176
2240 DATA13,222,17,192,189,17,192,201
2250 DATA255,208,3,222,18,192,189,17
2260 DATA192,221,4,192,189,18,192,253,5
2270 DATA192,144,18,189,4,192,157,17
2280 DATA192,189,5,192,157,18,192,104
2290 DATA74,72,76,46,193,104,74,72,176
2300 DATA8,254,17,192,208,3,254,18,192
2310 DATA104,74,176,5,169,1,153,25,192
2320 DATA200,232,232,224,4,240,3,76,91
2330 DATA192,238,27,192,108,29,192
2340 DATA35994:REM*PRUEFSUMME*
```

Sub Hunt

(U-Boot-Jagd)



„Sonar-Raum an Brücke... feindliche U-Boote dringen durch Unterwasserhöhlen ein... werft Wasserbomben!!!"

Die U-Boote passieren in unterschiedlicher Tiefe mit unterschiedlicher Geschwindigkeit. Passen Sie also den richtigen Zeitpunkt des Abwurfs ab, oder die U-Boote entkommen. Wenn Sie schon sicher sind, eines zu treffen, driftet dieses im letzten Moment noch ein Stück nach unten weg. Und es kommen ständig neue.

Lassen Sie nicht mehr als zehn U-Boote entkommen, oder das Spiel ist zu Ende.

Das Schiff wird mit Hilfe des Joysticks über den Schirm bewegt, die Wasserbomben über den Feuerknopf abgeworfen.

Gute Jagd!

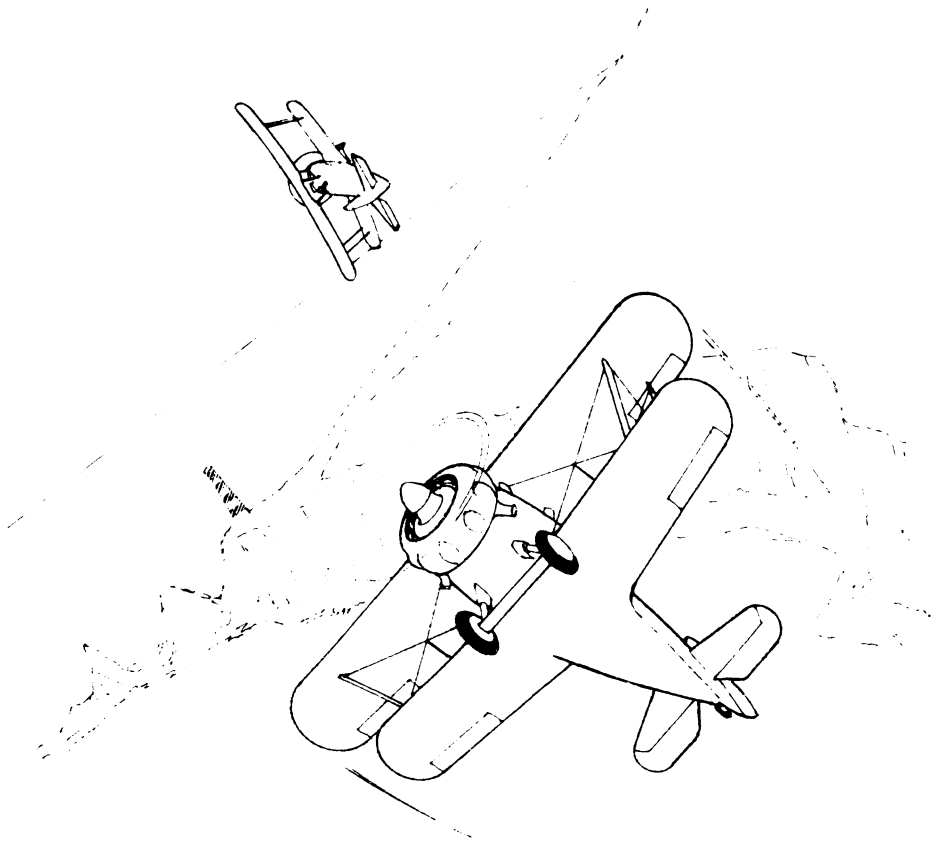
```
1000 REM *** SUB HUNT ***
1010 PRINTCHR$(147)
1020 POKE53280,0:POKE53281,6
1030 POKE54276,0:POKE54290,0
1040 FORI=49183TO49478:READJ
1050 POKEI,J:CC=CC+J:NEXTI
1060 READJ:IFCC<>JTHENPRINT"FEHLER IN DE
N DATEN":END
1070 FORI=0TO62:READJ:POKEI+832,J:NEXT
1080 FORI=0TO62:READJ:POKEI+896,J:NEXT
1090 FORI=0TO62:READJ:POKEI+960,J:NEXT
1100 SYS49183:REM JOYSTICK-ABFRAGE IN MS
1110 SC=1024:CO=55296
1120 FORT=0TO239:POKECO+40+T,14
1130 POKESC+40+T,160:NEXTT
1140 FORT=0TO39:POKE56256+T,5
1150 POKE1984+T,102:NEXTT
1160 PRINTCHR$(19)CHR$(5)"PUNKTE:"SE
1170 PRINTCHR$(19)TAB(13)CHR$(18)CHR$(15
4)"<< SUB HUNT >>"CHR$(146)
1180 PRINTCHR$(19)CHR$(5)TAB(30)"MAX:"HI
1190 REM *** JOYSTICK-PARAMETER SETZEN *
**
1200 POKE49152,60:POKE49153,0
1210 POKE49156,255:POKE49157,0
1220 POKE49160,92:POKE49161,0
1230 POKE49164,92:POKE49165,0
1240 POKE49168,0:POKE49177,0
1250 POKE49169,150:POKE49173,100
1260 REM *** SPRITES INITIALISIEREN ***
1270 VC=53248:PO=2040
1280 POKEPO,13:POKEPO+1,14
1290 POKEPO+2,15
1300 POKEVC+21,3:REM SPRITES 0 UND 1 ANS
CHALTEN
1310 POKEVC+39,0:POKEVC+40,0:POKEVC+41,3
1320 POKEVC+29,3:REM 0 UND 1 IN X-RICHTU
NG VERGROESSERN
1330 POKEVC,50+INT(RND(TI)*150)
1340 POKEVC+2,150:POKEVC+3,92:REM SCHIFF
1350 POKEVC+4,120:POKEVC+5,170:REM WASSE
RBOMBE
1360 POKEVC+39,0
1370 BY=110
1380 GOSUB1740
```

```
1390 REM *** HAUPTSCHLEIFE ***
1400 X=PEEK(49169)
1410 IFSE>HITHENHI=SE
1420 PRINTCHR$(19)TAB(7);SE:PRINTCHR$(19
)TAB(35);HI
1430 POKEVC+2,X
1440 IFDR=0THENBX=X
1450 F=PEEK(49177)
1460 POKE49177,0
1470 IFF=1THENDR=1
1480 IFDR=1THENPOKEVC+21,7
1490 IFDR=1THENBY=BY+2
1500 IFBY>242THENBY=110:POKEVC+21,3:DR=0
1510 POKEVC+4,BX:POKEVC+5,BY
1520 POKEVC+30,0
1530 IFPEEK(VC+30)AND2=2THEN1580
1540 SX=SX+D:IFSX>254THENGOSUB1740:ESC=E
SC+1:IFESC>9THEN1790
1550 IFRND(1)<.03THENSY=SY+D:IFSY>220THE
NSY=220
1560 POKEVC+0,SX:POKEVC+1,SY
1570 GOTO1400
1580 REM *** VOLLTREFFER ***
1590 SE=SE+50+INT(SY/5)+D
1600 POKE54296,15:POKE54276,129:POKE5429
0,33
1610 POKE54277,31:POKE54291,31
1620 POKE54278,240:POKE54292,240
1630 POKEVC+21,3:BY=110:DR=0
1640 FORT=1TO50:R=INT(RND(TI)*15)
1650 POKEVC+39,R
1660 POKE54273,INT(RND(TI)*5)
1670 POKE54287,INT(RND(TI)*100)
1680 POKE54272,100:POKE54286,100
1690 NEXTT
1700 POKE54276,0:POKE54290,0
1710 POKEVC+21,2:FORT=1TO500:NEXT
1720 GOTO1360
1730 REM *** NEUES U-BOOT ***
1740 SY=150+INT(RND(TI)*75):SX=0
1750 POKEVC,SX:POKEVC+1,SX
1760 D=INT(RND(TI)*8+1)
1770 POKEVC+21,3:BY=110:DR=0
1780 RETURN
1790 REM *** SPIELENDE ***
```

```
1800 POKE54276,33:POKE54277,31
1810 POKE54278,15:FORK=1TO4
1820 FORT=50TO120
1830 POKE54273,T:POKE54272,100
1840 NEXT:NEXT
1850 POKE54276,0:SE=0:ESC=0
1860 FORT=1TO3:PRINTCHR$(17):NEXT
1870 PRINTCHR$(5)TAB(5)"TASTE DRUECKEN F
UER NEUES SPIEL"
1880 GETA$:IFA$=""THEN1880
1890 PRINTCHR$(147):GOTO1110
1900 REM *** DATEN FUER JOYSTICKABFRAGE
***
1910 DATA120,173,20,3,141,29,192,173,21
1920 DATA3,141,30,192,169,71,141,20,3
1930 DATA169,192,141,21,3,88,96,120,173
1940 DATA29,192,141,20,3,173,30,192,141
1950 DATA21,3,88,96,173,27,192,205,16
1960 DATA192,240,3,76,65,193,169,255
1970 DATA141,27,192,162,0,160,0,185,2
1980 DATA220,141,28,192,169,224,153,2
1990 DATA220,185,0,220,72,173,28,192
2000 DATA153,2,220,189,8,192,221,21,192
2010 DATA189,9,192,253,22,192,144,18
2020 DATA189,8,192,157,21,192,189,9,192
2030 DATA157,22,192,104,74,72,76,162
2040 DATA192,104,74,72,176,13,222,21
2050 DATA192,189,21,192,201,255,208,3
2060 DATA222,22,192,189,21,192,221,12
2070 DATA192,189,22,192,253,13,192,144
2080 DATA18,189,12,192,157,21,192,189
2090 DATA13,192,157,22,192,104,74,72,76
2100 DATA207,192,104,74,72,176,8,254,21
2110 DATA192,208,3,254,22,192,189,0,192
2120 DATA221,17,192,189,1,192,253,18
2130 DATA192,144,18,189,0,192,157,17
2140 DATA192,189,1,192,157,18,192,104
2150 DATA74,72,76,1,193,104,74,72,176
2160 DATA13,222,17,192,189,17,192,201
2170 DATA255,208,3,222,18,192,189,17
2180 DATA192,221,4,192,189,18,192,253,5
2190 DATA192,144,18,189,4,192,157,17
2200 DATA192,189,5,192,157,18,192,104
2210 DATA74,72,76,46,193,104,74,72,176
2220 DATA8,254,17,192,208,3,254,18,192
```

```
2230 DATA104,74,176,5,169,1,153,25,192
2240 DATA200,232,232,224,4,240,3,76,91
2250 DATA192,238,27,192,108,29,192
2260 DATA35994:REM*PRUEFSUMME*
2270 REM *** SPRITE 0 (U-BOOT) ***
2280 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,8,0
2290 DATA0,12,0,0,12,0,0,12,0,192,28,0
2300 DATA231,255,240,159,255,248,127
2310 DATA109,188,95,255,248,231,255,240
2320 DATA192,0,0,0,0,0,0,0,0,0,0,0,0,0
2330 DATA0,0,0,0,0,0,0
2340 REM *** SPRITE 1 (SCHIFF) ***
2350 DATA0,0,0,0,0,0,0,0,0,0,128,0,0
2360 DATA192,0,0,224,0,5,193,32,3,225
2370 DATA200,231,251,242,255,255,255
2380 DATA255,255,255,127,255,254,63,255
2390 DATA254,31,255,252,0,0,0,0,0,0,0,0
2400 DATA0,0,0,0,0,0,0,0,0,0,0,0,0
2410 REM *** SPRITE 2 (WASSERBOMBE) ***
2420 DATA0,34,0,0,0,0,0,84,0,0,0,1,16,0
2430 DATA0,4,0,0,1,32,0,0,0,0,0,34,0,0
2440 DATA112,0,0,248,0,0,120,0,0,48,0,0
2450 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2460 DATA0,0,0,0,0,0,0,0
```

Dog Fight
(Luftkampf)



Dog Fight versetzt Sie in die Zeit des Ersten Weltkriegs in das Cockpit eines Doppeldeckers zurück.

Die Luftschlacht findet über dem Englischen Kanal statt. Ihr Auftrag ist es, so viele gegnerische Flugzeuge wie möglich abzuschießen. Aufgrund des heftigen Windes müssen Sie die Ziele sehr sorgfältig auffassen. Bringen Sie Ihr Fadenkreuz mit dem Flugzeug zur Deckung und schießen Sie, daß die Rohre glühen!!!

```
1000 REM *** DOG FIGHT ***
1010 REM *** DATEN DES MS-PROGRAMMS ZUR
JOYSTICKABFRAGE ***
1020 DATA120,173,20,3,141,29,192,173,21
1030 DATA3,141,30,192,169,71,141,20,3
1040 DATA169,192,141,21,3,88,96,120,173
1050 DATA29,192,141,20,3,173,30,192,141
1060 DATA21,3,88,96,173,27,192,205,16
1070 DATA192,240,3,76,65,193,169,255
1080 DATA141,27,192,162,0,160,0,185,2
1090 DATA220,141,28,192,169,224,153,2
1100 DATA220,185,0,220,72,173,28,192
1110 DATA153,2,220,189,8,192,221,21,192
1120 DATA189,9,192,253,22,192,144,18
1130 DATA189,8,192,157,21,192,189,9,192
1140 DATA157,22,192,104,74,72,76,162
1150 DATA192,104,74,72,176,13,222,21
1160 DATA192,189,21,192,201,255,208,3
1170 DATA222,22,192,189,21,192,221,12
1180 DATA192,189,22,192,253,13,192,144
1190 DATA18,189,12,192,157,21,192,189
1200 DATA13,192,157,22,192,104,74,72,76
1210 DATA207,192,104,74,72,176,8,254,21
1220 DATA192,208,3,254,22,192,189,0,192
1230 DATA221,17,192,189,1,192,253,18
1240 DATA192,144,18,189,0,192,157,17
1250 DATA192,189,1,192,157,18,192,104
1260 DATA74,72,76,1,193,104,74,72,176
1270 DATA13,222,17,192,189,17,192,201
1280 DATA255,208,3,222,18,192,189,17
1290 DATA192,221,4,192,189,18,192,253,5
1300 DATA192,144,18,189,4,192,157,17
1310 DATA192,189,5,192,157,18,192,104
1320 DATA74,72,76,46,193,104,74,72,176
1330 DATA8,254,17,192,208,3,254,18,192
1340 DATA104,74,176,5,169,1,153,25,192
1350 DATA200,232,232,224,4,240,3,76,91
1360 DATA192,238,27,192,108,29,192
1370 REM *** PRUEFSUMME ***
1380 DATA35994
1390 REM *** DATEN DES MS-PROGRAMMS LESE
N ***
1400 FORI=49183TO49478
1410 READJ
1420 POKEI,J
```

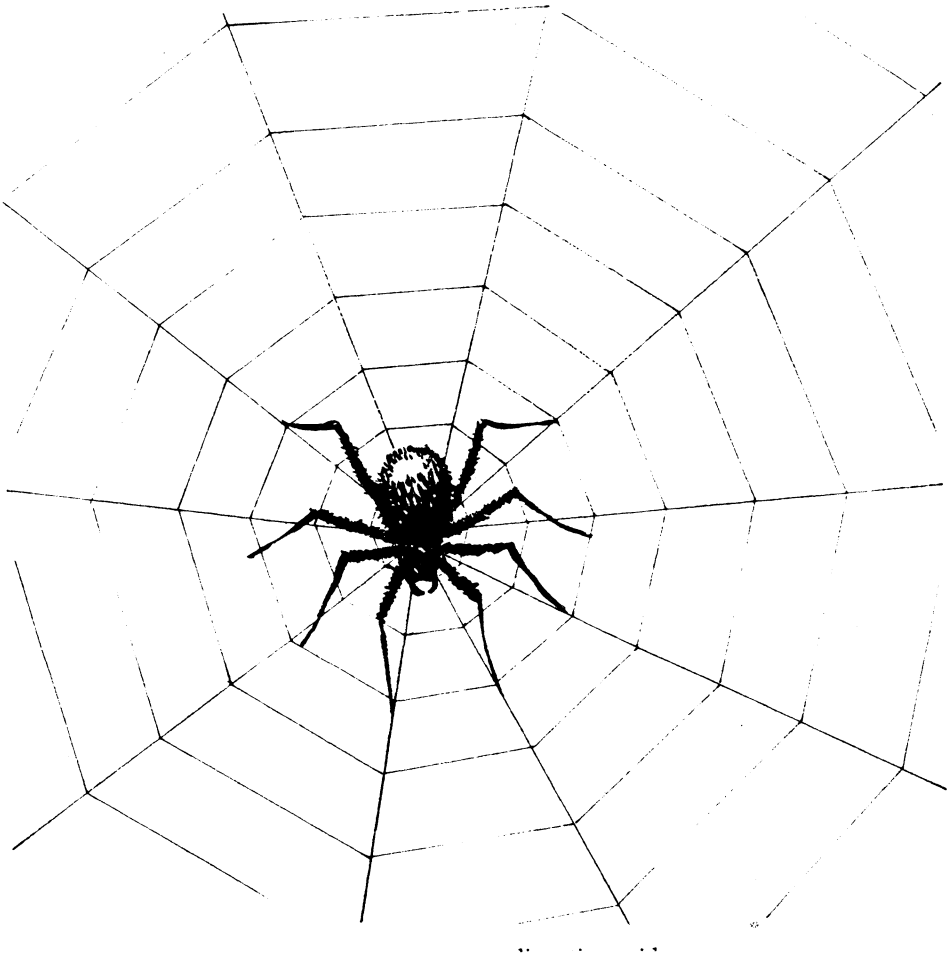
```
1430 CC=CC+J
1440 NEXTI
1450 READJ
1460 IFCC=JTHEN1480
1470 PRINT"FEHLER IN DEN DATEN":END
1480 SYS(49183):REM MS-PGR AUFRUFEN
1490 REM *** SPRITE 0 ***
1500 FORI=0TO62:READJ
1510 POKE832+I,J:NEXTI
1520 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1530 DATA0,0,0,195,0,0,66,0,0,66,0,127
1540 DATA195,254,0,66,0,0,66,0,0,195,0,0
1550 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1560 DATA0,0,0,0,0,0,0
1570 REM *** SPRITE 1 ***
1580 FORI=0TO62:READJ
1590 POKE896+I,J:NEXTI
1600 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1610 DATA24,0,255,255,255,97,24,134,81
1620 DATA90,138,73,60,146,69,126,162,67
1630 DATA126,194,255,255,255,0,60,0,0,90
1640 DATA0,0,129,0,1,129,128,0,0,0,0,0,0
1650 DATA0,0,0,0,0,0
1660 REM *** SPRITE 2 ***
1670 FORI=0TO62:POKEI+960,INT(RND(TI)*25
5)
1680 NEXTI
1690 REM *** JOYSTICK-PARAMETER SETZEN *
**
1700 Y1=122:X1=155
1710 POKE49152,30:POKE49153,0
1720 POKE49156,250:POKE49157,0
1730 POKE49160,29:POKE49161,0
1740 POKE49164,229:POKE49165,0
1750 POKE49177,0:POKE49168,0
1760 REM *** VARIABLEN & KONSTANTEN ***
1770 VC=53248
1780 S=54272
1790 FORI=0TO24:POKES+I,0:NEXTI
1800 POKES+24,15:POKES+5,31
1810 POKES+6,15
1820 REM *** SPRITES INITIALISIEREN ***
1830 POKE2040,13:POKE2041,14
1840 POKEVC+21,3:REM 0 UND 1 ANSCHALTEN
1850 POKEVC+39,1:REM FARBE SPRITE 0
```

```
1860 POKEVC+40,0:REM FARBE SPRITE 1
1870 POKEVC+29,3:REM SPRITE 0 IN X-RICHT
UNG VERGROESSERN
1880 POKEVC+23,1:REM SPRITE 1 IN Y-RICHT
UNG VERGROESSERN
1890 POKE49169,X1:POKE49173,Y1
1900 POKE49169,X1:POKE49173,Y1
1910 PRINTCHR$(147)
1920 POKE53280,9:POKE53281,9
1930 PRINTCHR$(17)CHR$(17)
1940 PRINT"   ┌──────────────────────────────
──────┐"
1950 PRINT"   │  ┌─┐  ┌─┐  ┌─┐   ┌──┬─┐  ┌─┐  │
  │ ──┬──  │"
1960 PRINT"   │  │ │  │ │  │ │     │    │ │  │
  │  │  │    │"
1970 PRINT"   │  │ │  │ │  │ │     │    │ │  │
  │  │  │    │"
1980 PRINT"   │  │ │  │ │  │ │     ├─   │ │  ├
─┤  │  │    │"
1990 PRINT"   │  │ │  │ │  │ ┬   │    │ │ ┬ │
  │  │  │    │"
2000 PRINT"   │  └─┘  └─┘  └─┘   │   ┴  └─┘  │
  │  │  │    │"
2010 PRINT"   └──────────────────────────────
──────┘"
2020 PRINTCHR$(17)CHR$(17)CHR$(17)CHR$(5
)
2030 PRINTTAB(12)"-TASTE DRUECKEN-"
2040 GETA$:IFA$=""THEN2040
2050 PRINTCHR$(147)
2060 PRINTCHR$(147)
2070 POKE53280,9:POKE53281,14
2080 FORI=0TO400:POKE1624+I,160
2090 POKE55896+I,6:NEXT
2100 X2=50:Y2=50:DX=1:DY=1
2110 POKE2041,14:POKEVC+40,0
2120 POKEVC+2,X2:POKEVC+3,Y2
2130 POKEVC+21,3
2140 REM *** SCHLEIFE ***
2150 X1=PEEK(49169):Y1=PEEK(49173)
2160 F=PEEK(49177):POKE49177,0
2170 IFRND(1)<.05THENPOKEVC+23,1:POKEVC+
29,1
2180 PRINTCHR$(19)CHR$(5)"PUNKTE:"SE
```

```
2190 IFSE>HITHENHI=SE
2200 PRINTCHR$(19)TAB(25)"MAX:"HI
2210 IFRND(1)<.05THENPOKEVC+23,3:POKEVC+
29,3
2220 IFF<>1THEN2260
2230 POKES+4,129:POKES+1,20:POKES,140
2240 FORK=1TO2:NEXT:POKES+4,0:B=B+1
2250 IFB=500THEN2430
2260 POKEVC+0,X1:POKEVC+1,Y1
2270 POKEVC+2,X2:POKEVC+3,Y2
2280 X2=X2+DX:IFX2>240THENX2=240:DX=-DX
2290 IFX2<50THENX2=50:DX=-DX
2300 Y2=Y2+DY:IFY2>200THENY2=200:DY=-DY
2310 IFY2<50THENY2=50:DY=-DY
2320 IFRND(1)<.1THENDX=DX+1:IFDX>10THEND
X=1
2330 IFRND(1)<.1THENDY=DY+1:IFDY>10THEND
Y=1
2340 IFF=1ANDPEEK(VC+30)AND2=2THENHT=HT+
1
2350 IFHT=30THEN2380
2360 GOTO2150
2370 REM *** TREFFER ***
2380 POKE2041,15:POKES+4,129:HT=0
2390 POKEVC+40,8:SE=SE+71
2400 POKES+1,7:POKES,90:POKEVC+21,2
2410 FORK=1TO1000:NEXT:POKES+4,0
2420 POKEVC+21,1:F=0:GOTO2100
2430 REM *** MUNITION VERSCHOSSEN ***
2440 PRINTCHR$(19)CHR$(17)CHR$(17)CHR$(1
7)
2450 PRINTTAB(10)"MUNITION VERSCHOSSEN"
2460 SE=0:B=0:FORK=1TO2000:NEXT
2470 POKEVC+21,0
2480 GOTO1910
```

Spider Hunt

(Spinnenjagd)



In „Spider Hunt" müssen Sie möglichst viele radioaktive Spinnen mit Ihrem Laser zerstören. Dirigieren Sie das Fadenkreuz über die Spinne und betätigen Sie den Feuerknopf – die Spinne vergeht in gleißendem Licht zu Nichts.

```
1000 REM *** SPIDER HUNT ***
1010 PRINTCHR$(147):GOSUB1930:REM TITELB
ILD
1020 POKE53280,0:POKE53281,0
1030 REM *** JOYSTICK-DATEN LESEN ***
1040 FORI=49183TO49478:READJ
1050 POKEI,J:CC=CC+J:NEXTI
1060 READJ:IFCC<>JTHENPRINT"FEHLER IN DE
N DATEN":END
1070 SYS(49183):REM MS-PGR AUFRUFEN
1080 REM *** SPRITE-DATEN LESEN ***
1090 FORI=0TO62:READJ:POKEI+832,J:NEXTI
1100 FORI=0TO62:READJ:POKEI+896,J:NEXTI
1110 REM *** JOYSTICK-PARAMETER SETZEN *
**
1120 POKE49152,40:POKE49153,0
1130 POKE49156,255:POKE49157,0
1140 POKE49160,62:POKE49161,0
1150 POKE49164,213:POKE49165,0
1160 POKE49168,0:POKE49177,0
1170 POKE49169,100:POKE49170,0
1180 POKE49173,150:POKE49174,0
1190 REM *** SPRITES INITIALISIEREN ***
1200 POKE2040,13:POKE2041,14:VC=53248
1210 POKEVC+23,2:REM IN Y-RICHTUNG VERGR
OESSERN
1220 POKEVC+29,2:REM IN X-RICHTUNG VERGR
OESSERN
1230 POKEVC+2,150:POKEVC+3,150
1240 REM *** TONREGISTER SETZEN ***
1250 S=54272:FORI=0TO24:POKES+L,0:NEXTI
1260 POKES+24,15:REM LAUTSTAERKE
1270 POKES+5,41:POKES+6,89
1280 POKES+19,31:POKES+20,240
1290 POKE782,3:POKE781,18:SYS65520
1300 PRINTCHR$(30)"* ZUM SPIELBEGINN TAS
TE DRUECKEN *"
1310 GETA$:IFA$=""THEN1310
1320 PRINTCHR$(147)
1330 REM *** BILDSCHIRM AUFBAUEN ***
1340 PRINTCHR$(19)CHR$(28)CHR$(17)
1350 FORK=1TO22
1360 PRINT"XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXX";
1370 NEXTK
```

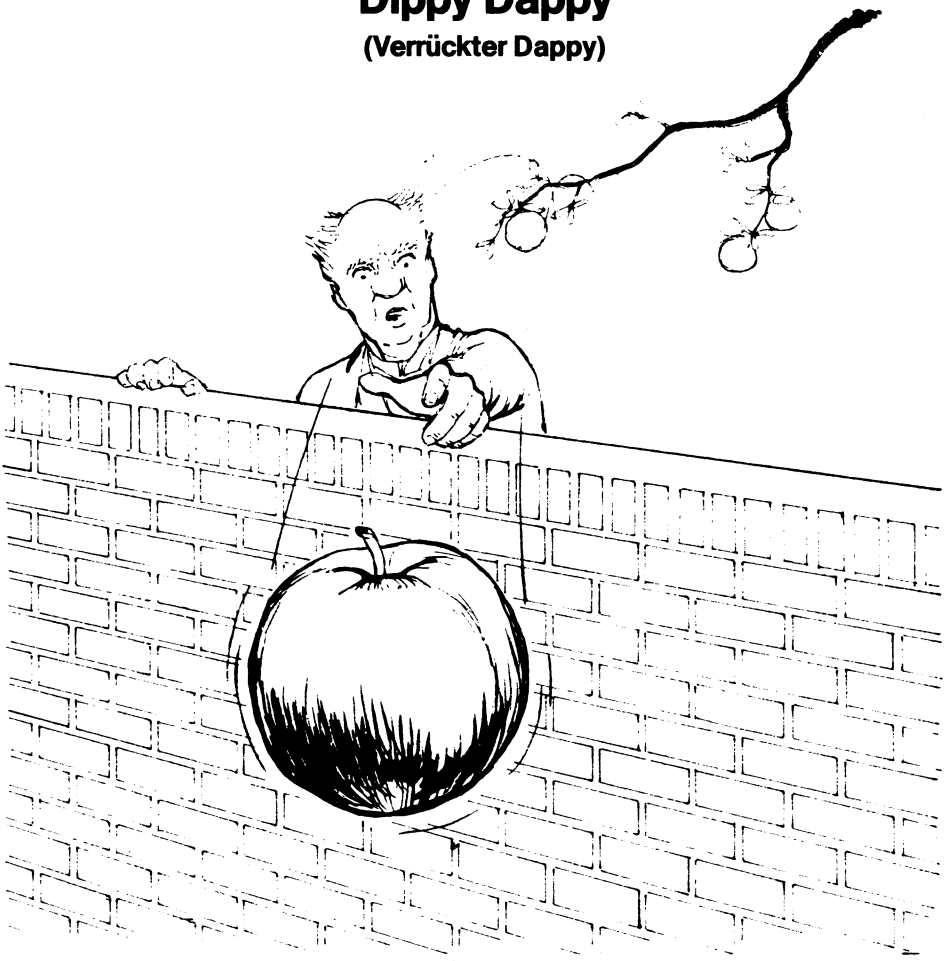
```
1380 FORI=0TO39:POKE55336+I,5
1390 POKE1064+I,160:NEXTI
1400 FORI=0TO39:POKE56256+I,5
1410 POKE1984+I,160:NEXTI
1420 FORI=0TO22*40STEP40
1430 POKE55376+I,5:POKE1104+I,160
1440 POKE55415+I,5:POKE1143+I,160
1450 NEXTI
1460 POKEVC+21,3:REM SPRITES ANSCHALTEN
1470 POKEVC+39,1:POKEVC+40,7:REM FARBE
1480 DX=4:DY=4:GOSUB1660
1490 LI=50
1500 REM *** HAUPTSCHLEIFE ***
1510 PX=PEEK(49169):PY=PEEK(49173)
1520 POKEVC+0,PX:POKEVC+1,PY
1530 NP=NP+1:F=PEEK(49177)
1540 IFNP>LITHENGOSUB1660
1550 POKEVC+2,TX:POKEVC+3,TY
1560 POKEVC+30,0:POKE49177,0
1570 IFF=1ANDPEEK(VC+30)AND1=1THEN1710
1580 IFED=3THEN1850
1590 PRINTCHR$(19)CHR$(5)"PUNKTE:"SE
1600 IFSE>HITHENHI=SE
1610 PRINTCHR$(19)TAB(25)"MAX:"HI
1620 TX=TX+DX:IFTX<40ORTX>250THENDX=-DX
1630 TY=TY+DY:IFTY<62ORTY>213THENDY=-DY
1640 GOTO1510
1650 REM *** NEUE STARTPOSITION DER SPIN
NE ***
1660 TX=INT(RND(TI)*210+40)
1670 TY=INT(RND(TI)*151+62)
1680 POKEVC+2,TX:POKEVC+3,TY
1690 NP=0:LI=LI-1:IFLI<1THENED=ED+1:LI=5
0
1700 RETURN
1710 REM *** TREFFER ***
1720 POKEVC+21,2:SE=SE+INT(500/LI)+NP
1730 FORT=128TO1STEP-7:POKEVC+40,T
1740 POKEVC+29,0
1750 POKES+4,33:POKES+0,T:POKES+1,T+127
1760 POKES+18,33:POKES+15,T+7:POKES+14,T
+127
1770 POKEVC+29,2
1780 NEXT:POKEVC+21,1
1790 POKES+4,0:POKES+18,0
```

```
1800 FORK=1TO1000:NEXT
1810 GOSUB1660
1820 POKE49177,0:POKEVC+40,7
1830 POKEVC+21,3:NP=0
1840 GOTO1510
1850 REM *** SPIELENDE ***
1860 PRINTCHR$(19)CHR$(17)CHR$(17)CHR$(1
7)
1870 PRINTTAB(14)"-SPIELENDE-"CHR$(17)
1880 PRINT"  * TASTE DRUECKEN FUER NEUES
 SPIEL *"
1890 GETA$:IFA$=""THENPOKE646,INT(RND(TI
)*15):GOTO1860
1900 PRINTCHR$(147):POKEVC+21,0
1910 NP=0:ED=0:SE=0
1920 PRINTCHR$(147):GOTO1330
1930 REM *** TITELBILD ***
1940 PRINTCHR$(30)
1950 PRINT TAB(7)"┌──────────────────────────
─┐"
1960 PRINT TAB(7)"│ ●●● ●●  ●  ●●  ●●● ●●
 │"
1970 PRINT TAB(7)"│●   ● ●  ● ● ●  ●
●│"
1980 PRINT TAB(7)"│ ●●  ●●  ● ● ● ●●  ●●
 │"
1990 PRINT TAB(7)"│  ● ●  ● ● ● ●  ●
●│"
2000 PRINT TAB(7)"│●●●  ●  ● ●●  ●●● ●
●│"
2010 PRINT TAB(7)"└──────────────────────────
─┘"
2020 PRINTTAB(10)"┌──────────────────┐"
2030 PRINTTAB(10)"│● ● ● ● ●  ● ●●●│"
2040 PRINTTAB(10)"│● ● ● ● ●● ●  ●  │"
2050 PRINTTAB(10)"│●●● ● ● ●●●●  ●  │"
2060 PRINTTAB(10)"│● ● ● ● ● ●●  ●  │"
2070 PRINTTAB(10)"│● ● ●●● ●  ●  ●  │"
2080 PRINTTAB(10)"└──────────────────┘"
2090 RETURN
2100 DATA120,173,20,3,141,29,192,173,21
2110 DATA3,141,30,192,169,71,141,20,3
2120 DATA169,192,141,21,3,88,96,120,173
2130 DATA29,192,141,20,3,173,30,192,141
2140 DATA21,3,88,96,173,27,192,205,16
```

```
2150 DATA192,240,3,76,65,193,169,255
2160 DATA141,27,192,162,0,160,0,185,2
2170 DATA220,141,28,192,169,224,153,2
2180 DATA220,185,0,220,72,173,28,192
2190 DATA153,2,220,189,8,192,221,21,192
2200 DATA189,9,192,253,22,192,144,18
2210 DATA189,8,192,157,21,192,189,9,192
2220 DATA157,22,192,104,74,72,76,162
2230 DATA192,104,74,72,176,13,222,21
2240 DATA192,189,21,192,201,255,208,3
2250 DATA222,22,192,189,21,192,221,12
2260 DATA192,189,22,192,253,13,192,144
2270 DATA18,189,12,192,157,21,192,189
2280 DATA13,192,157,22,192,104,74,72,76
2290 DATA207,192,104,74,72,176,8,254,21
2300 DATA192,208,3,254,22,192,189,0,192
2310 DATA221,17,192,189,1,192,253,18
2320 DATA192,144,18,189,0,192,157,17
2330 DATA192,189,1,192,157,18,192,104
2340 DATA74,72,76,1,193,104,74,72,176
2350 DATA13,222,17,192,189,17,192,201
2360 DATA255,208,3,222,18,192,189,17
2370 DATA192,221,4,192,189,18,192,253,5
2380 DATA192,144,18,189,4,192,157,17
2390 DATA192,189,5,192,157,18,192,104
2400 DATA74,72,76,46,193,104,74,72,176
2410 DATA8,254,17,192,208,3,254,18,192
2420 DATA104,74,176,5,169,1,153,25,192
2430 DATA200,232,232,224,4,240,3,76,91
2440 DATA192,238,27,192,108,29,192
2450 DATA35994:REM*PRUEFSUMME*
2460 REM *** SPRITE 0 (FADENKREUZ) ***
2470 DATA0,0,0,31,1,240,24,0,48,20,0,80
2480 DATA18,0,144,17,41,16,0,170,0,0
2490 DATA108,0,1,239,0,0,0,0,1,239,0,0
2500 DATA108,0,0,170,0,17,41,16,18,0
2510 DATA144,20,0,80,24,0,48,31,1,240
2520 DATA0,0,0,0,0,0,0,0,0
2530 REM *** SPRITE 1 (SPINNE) ***
2540 DATA0,0,0,0,0,0,0,0,0,24,97,128
2550 DATA102,246,96,129,216,16,15,127
2560 DATA0,51,236,192,199,190,48,30,247
2570 DATA128,99,252,96,130,100,16,15
2580 DATA111,0,48,240,192,65,104,32,130
2590 DATA4,16,128,0,16,0,0,0,0,0,0,0,0
2600 DATA0,0,0,0
```

Dippy Dappy

(Verrückter Dappy)



„O.K. Du wirfst die Äpfel 'rüber, Dappy, und ich fang' sie."

„Verstanden, Boss!"

Also, er hat gesagt, ich soll die Äpfel 'rüberwerfen, oder? Und er hat mir gesagt, wohin ich sie werfen soll, oder?

Hier ist die ganze Geschichte: Dappy brach auf Befehl seines Bosses Ralf Raffke in den Obstgarten von Bauer Emsig ein, um dessen rotbackige Äpfel zu stehlen. Dappy jedoch, mit dessen Schulbildung es nicht weit her ist, wußte nicht, wie ein Apfel aussieht. Sie verstehen also, daß es nicht seine Schuld war, als er plötzlich begann, mit Eiern zu werfen.

„Aber woher hat er die Eier?", werden Sie fragen. Nun ja, das ist eine andere Geschichte.

Steuern Sie Ralf Raffke mit Hilfe des Joysticks so, daß er alle Eier fängt. Wenn Sie zu viele fallen lassen, ist das Spiel beendet!

Ich wünsche Ihnen mit „Dippy Dappy" eiweißreiche Kost.

```
1000 REM *** DIPPY DAPPY ***
1010 REM *** JOYSTICK-DATEN LESEN ***
1020 FORI=49183TO49478:READJ
1030 POKEI,J:CC=CC+J:NEXTI
1040 READJ:IFJ<>CCTHENPRINT"DATA ERROR":
END
1050 REM *** DATEN FUER SPRITE 0,1,2 ***
1060 FORI=0TO62:READJ:POKEI+832,J:NEXT
1070 FORI=0TO62:READJ:POKEI+896,J:NEXT
1080 FORI=0TO62:READJ:POKEI+960,J:NEXT
1090 REM *** JOYSTICK-PARAMETER SETZEN *
**
1100 POKE49152,30:POKE49153,0
1110 POKE49156,255:POKE49157,0
1120 POKE49160,180:POKE49161,0
1130 POKE49164,197:POKE49165,0
1140 POKE49168,0:POKE49169,255:POKE49173
,180
1150 REM *** SPRITES 0,1,2 INITIALISIERE
N ***
1160 POKE2040,13:POKE2041,14:POKE2042,15
1170 VC=53248:SYS49183
1180 POKEVC+0,120:POKEVC+1,68:REM POSITI
ON SPRITE 0
1190 POKEVC+2,120:POKEVC+3,160:REM POSIT
ION SPRITE 1
1200 POKEVC+29,3:REM 0,1 IN X-RICHTUNG V
ERGROESSERN
1210 POKEVC+23,1:REM 0,1 IN Y-RICHTUNG V
ERGROESSERN
1220 POKEVC+39,5:REM FARBE SPRITE 0
1230 POKEVC+40,3:REM FARBE SPRITE 1
1240 POKEVC+41,1:REM FARBE SPRITE 2
1250 X=120
1260 REM *** BILDSCHIRM AUFBAUEN ***
1270 PRINTCHR$(147):POKE53280,0:POKE5328
1,0
1280 REM POKE53265,PEEK(53265)AND239
1290 SC=1064:CO=55336
1300 FORT=0TO160:POKECO+T,6
1310 POKESC+T,160:NEXTT
1320 FORT=0TO559STEP3:POKECO+T+160,2
1330 POKECO+T+160+1,2
1340 POKECO+T+160+2,2:POKESC+T+160,239
1350 POKESC+T+160+1,239
```

```
1360 POKESC+T+160+2,250:NEXTT
1370 FORT=0TO240:POKECO+T+720,5
1380 POKESC+T+720,160:NEXT
1390 PRINTCHR$(19)CHR$(30)
1400 PRINTTAB(30)CHR$(18)"   "CHR$(149);
1410 PRINT" "CHR$(28)" "CHR$(30)" ";
1420 PRINTCHR$(149)" "CHR$(30)" "
1430 PRINTCHR$(18);
1440 PRINTTAB(29)" "CHR$(28)" "CHR$(30);
1450 PRINT"   "CHR$(28)" "CHR$(30);
1460 PRINTCHR$(18);
1470 PRINT"   "CHR$(149)" "CHR$(30)" "
1480 PRINTCHR$(18);
1490 PRINTTAB(29)CHR$(28)" "CHR$(30)" ";
1500 PRINTCHR$(149)" "CHR$(30)" "CHR$(14
9);
1510 PRINT" "CHR$(28)" "CHR$(30)" ";
1520 PRINTCHR$(149)" "CHR$(28)" "CHR$(30
);
1530 PRINT" "
1540 PRINTCHR$(18);
1550 PRINTTAB(28)CHR$(28)" "CHR$(30)"   "
;
1560 PRINTCHR$(28)" "CHR$(149)" "CHR$(30
);
1570 PRINT"   "CHR$(149)" "CHR$(30)"   ";
1580 PRINTCHR$(149)" "
1590 REM POKE53265,PEEK(53265)OR16
1600 POKEVC+21,3:REM SPRITES 0,1,2 ANSCH
ALTEN
1610 POKE782,28:POKE781,10:SYS65520
1620 PRINTCHR$(5)CHR$(18)"┌──────────┐"
1630 POKE782,28:POKE781,11:SYS65520
1640 PRINTCHR$(5)CHR$(18)"|EINTRITT|"
1650 POKE782,28:POKE781,12:SYS65520
1660 PRINTCHR$(5)CHR$(18)"|VERBOTEN|"
1670 POKE782,28:POKE781,13:SYS65520
1680 PRINTCHR$(5)CHR$(18)"└──────────┘"CHR
$(146)
1690 PRINTCHR$(19)CHR$(158)"PUNKTE:"
1700 PRINTCHR$(19)CHR$(158)TAB(17)"MAX.P
UNKTE:"
1710 D=8:GOSUB1930:GOSUB1970
1720 REM *** HAUPTSCHLEIFE ***
1730 PRINTCHR$(19)CHR$(5)TAB(9);SE;" "
```

```
1740 IFSE>HITHENHI=SE
1750 PRINTCHR$(19)CHR$(5)TAB(30);HI
1760 POKEVC+0,X
1770 IFDR=0THENAX=INT(RND(TI)*210+40)
1780 IFDR=0THENF=F-1:IFF<1THENF=1
1790 X=X+D:IFX<30ORX>246THEND=-D
1800 POKEVC+39,INT(RND(TI)*2+4)
1810 POKEVC+30,0
1820 PX=PEEK(49169):PY=PEEK(49173)
1830 IFPEEK(VC+30)AND2=2THENGOSUB1970
1840 IFF=1THENDR=1
1850 IFDR=1THENPOKEVC+21,7
1860 IFDR=1THENAY=AY+6+INT(SE/100)
1870 IFAY>190THENGOSUB1930
1880 POKEVC+2,PX:POKEVC+3,PY
1890 POKEVC+4,AX:POKEVC+5,AY
1900 IFCA=1THENSE=SE+10:CA=0
1910 GOTO1730
1920 REM *** EI FALLENLASSEN ***
1930 AY=95:POKEVC+21,3:DR=0
1940 IFCA=0THENMI=MI+1:IFMI>5THEN2000
1950 RETURN
1960 REM *** GEFANGEN ***
1970 CA=1:AY=95:DR=0
1980 POKEVC+21,3:F=2
1990 RETURN
2000 REM *** EINES ZU VIEL VERFEHLT ***
2010 POKE781,20:POKE782,2:SYS65520
2020 PRINTCHR$(5)"DU HAST EIN EI ZU VIEL
 FALLENGELASSEN"
2030 POKE781,22:POKE782,2:SYS65520
2040 PRINTCHR$(5)"TASTE DRUECKEN FUER EI
N NEUES SPIEL! "
2050 GETA$:IFA$=""THEN2050
2060 SE=0:DR=0:F=5:AY=85:CA=0:MI=0
2070 GOTO1370
2080 DATA120,173,20,3,141,29,192,173,21
2090 DATA3,141,30,192,169,71,141,20,3
2100 DATA169,192,141,21,3,88,96,120,173
2110 DATA29,192,141,20,3,173,30,192,141
2120 DATA21,3,88,96,173,27,192,205,16
2130 DATA192,240,3,76,65,193,169,255
2140 DATA141,27,192,162,0,160,0,185,2
2150 DATA220,141,28,192,169,224,153,2
2160 DATA220,185,0,220,72,173,28,192
```

```
2170 DATA153,2,220,189,8,192,221,21,192
2180 DATA189,9,192,253,22,192,144,18
2190 DATA189,8,192,157,21,192,189,9,192
2200 DATA157,22,192,104,74,72,76,162
2210 DATA192,104,74,72,176,13,222,21
2220 DATA192,189,21,192,201,255,208,3
2230 DATA222,22,192,189,21,192,221,12
2240 DATA192,189,22,192,253,13,192,144
2250 DATA18,189,12,192,157,21,192,189
2260 DATA13,192,157,22,192,104,74,72,76
2270 DATA207,192,104,74,72,176,8,254,21
2280 DATA192,208,3,254,22,192,189,0,192
2290 DATA221,17,192,189,1,192,253,18
2300 DATA192,144,18,189,0,192,157,17
2310 DATA192,189,1,192,157,18,192,104
2320 DATA74,72,76,1,193,104,74,72,176
2330 DATA13,222,17,192,189,17,192,201
2340 DATA255,208,3,222,18,192,189,17
2350 DATA192,221,4,192,189,18,192,253,5
2360 DATA192,144,18,189,4,192,157,17
2370 DATA192,189,5,192,157,18,192,104
2380 DATA74,72,76,46,193,104,74,72,176
2390 DATA8,254,17,192,208,3,254,18,192
2400 DATA104,74,176,5,169,1,153,25,192
2410 DATA200,232,232,224,4,240,3,76,91
2420 DATA192,238,27,192,108,29,192
2430 DATA35994:REM*PRUEFSUMME*
2440 REM *** SPRITE 0 (DAPPY) ***
2450 DATA12,156,152,98,127,35,17,255
2460 DATA196,11,255,232,103,255,243,31
2470 DATA127,124,14,62,56,30,62,60,28
2480 DATA255,156,28,190,156,31,255,252
2490 DATA0,62,0,0,127,0,0,99,0,0,193
2500 DATA128,0,128,128,0,65,0,0,62,0
2510 DATA0,0,0,0,0,0,0,0,0
2520 REM *** SPRITE 1 (FAENGER) ***
2530 DATA0,0,0,15,255,224,18,170,144,19
2540 DATA255,144,13,85,96,15,255,224,24
2550 DATA254,48,48,0,24,96,254,12,99
2560 DATA255,140,103,17,204,62,68,248
2570 DATA30,0,240,3,57,128,3,255,128,3
2580 DATA199,128,1,255,0,0,108,0,0,108
2590 DATA0,1,239,0,3,239,128
2600 REM *** SPRITE 2 (EI) ***
2610 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

```
2620 DATA0,0,0,0,32,0,0,32,0,0,32,0,0
2630 DATA32,0,1,36,0,1,36,0,1,4,0,1,56
2640 DATA128,1,124,128,0,254,128,0,254
2650 DATA0,0,124,0,0,56,0,0,0,0,0,0,0
```

Fun Match

(Wort-Kaleidoskop)

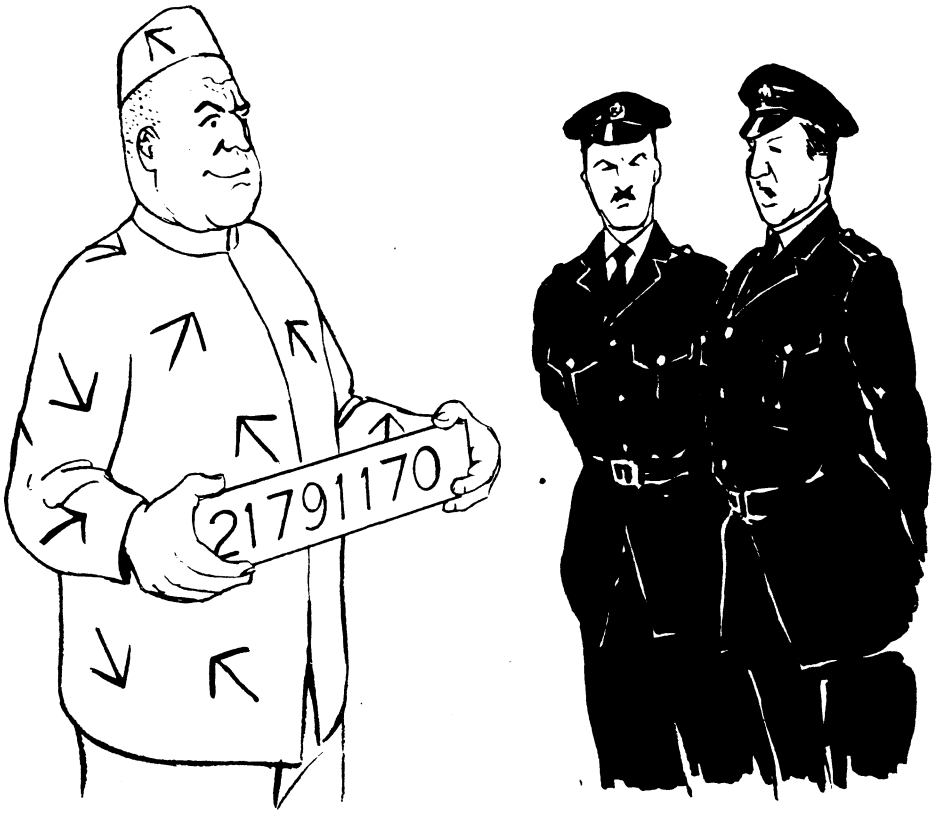
Fun Match ist ein kleines Programm, das aus einem bestimmten Wortvorrat immer neue Wortkombinationen entwickelt. Dabei kommen sehr lustige Ergebnisse zustande.

Sie können in den DATA-Zeilen auch Worte Ihrer Wahl unterbringen. Die von uns vorgeschlagenen Wörter sollen nur Beispiele sein. Sie sind natürlich nicht auf zehn Wortsätze beschränkt. Dieses Programm kann als Grundstock für längere Programme, wie z.B. Anagramm oder Galgenmännchen, verwendet werden.

```
1000 REM *** FUN MATCH ***
1010 DIMF$(10),S$(10),T$(10):REM FELDER
DIMENSIONIEREN
1020 FORI=1TO10
1030 READF$(I),S$(I),T$(I)
1040 NEXTI
1050 REM *** HAUPTPROGRAMMSCHLEIFE ***
1060 PRINTCHR$(147)
1070 N1=INT(RND(TI)*10+1)
1080 N2=INT(RND(TI)*10+1)
1090 N3=INT(RND(TI)*10+1)
1100 PRINT"DER ";F$(N1);
1110 PRINT" ";S$(N2);" ";T$(N3)
1120 PRINT:PRINT:PRINT
1130 PRINT"TASTE DRUECKEN"
1140 GETA$:IFA$=""THEN1140
1150 GOTO1060
1160 REM *** WOERTER ***
1170 DATA"GROSSE","BRAUNE","HUND"
1180 DATA"KLEINE","RUNDE","BALL"
1190 DATA"TRAURIGE","WEINENDE","CLOWN"
1200 DATA"LUSTIGE","FETTE","MANN"
1210 DATA"STILLE","BLAUE","SEE"
1220 DATA"PELZIGE","ROTE","PFIRSICH"
1230 DATA"EINSAME","HUNGRIGE","BAER"
1240 DATA"JUNGE","GUTAUSSEHENDE","KELLNE
R"
1250 DATA"SCHNELLE","ZERLEGBARE","TRETRO
LLER"
1260 DATA"SCHLAFFE","NASSE","WASCHLAPPEN
"
```

Guess my number

(Zahlenraten)



Es ist sinnvoll, mit einem einfacheren Programm wie „Guess my number" zu beginnen, mit dem Sie nicht nur spielen, sondern auch die Verwendung von Variablen und IF...THEN-Befehlen lernen können.

Die Idee des Spiels ist, daß der Computer eine Zufallszahl bildet, die Sie erraten müssen. Die Zahl ist eine ganze Zahl zwischen 0 und 100. Diese Spanne können Sie natürlich Ihren Bedürfnissen anpassen. Jedesmal, wenn Sie einen Rateversuch eingegeben haben, antwortet der Computer entweder mit „Zu groß" oder mit „Zu klein". Sie können Ihre Schätzungen dann diesen Hinweisen anpassen.

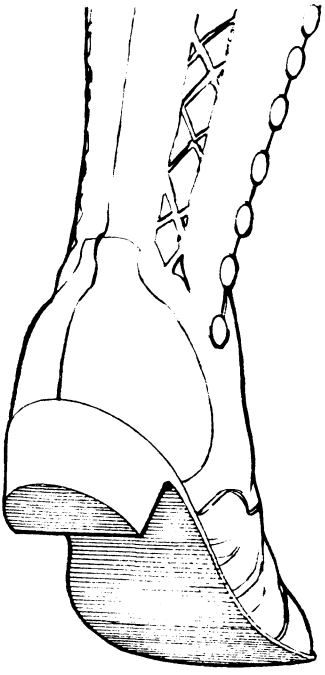
Nachdem Sie die richtige Zahl erraten haben, teilt Ihnen der Computer mit, wieviele Versuche Sie gebraucht haben.

Das eigentliche Listing ist gerade eine Bildschirmseite lang und beweist, daß man nicht viel Speicherplatz braucht, um ein interessantes Programm zu schreiben.

```
1000 REM *** GUESS MY NUMBER ***
1010 PRINTCHR$(147):GG=0:POKE53280,6
1020 PRINTCHR$(5)"RATE MEINE ZAHL (ZWISC
HEN 1 UND 100)
1030 X=INT(RND(1)*100)+1
1040 POKE19,1
1050 PRINT:PRINTCHR$(154)"WELCHE ZAHL RA
ETST DU >";
1060 INPUTG$:G=INT(VAL(G$))
1070 IFG<XTHENPRINT:PRINTCHR$(155)"ZU KL
EIN. VERSUCH'S NOCH MAL!":GG=GG+1
1080 IFG>XTHENPRINT:PRINTCHR$(155)"ZU GR
OSS. VERSUCH'S NOCH MAL!":GG=GG+1
1090 IFG=XTHENPRINT:PRINTCHR$(5)"GUT, DU
 HAST"GG"VERSUCHE GEBRAUCHT."
1100 IFG=XTHENPRINT:PRINT"NOCH EINMAL? (
J/N)"
1110 IFG<>XTHEN1040
1120 GETA$:IFA$<>"J"ANDA$<>"N"THEN1120
1130 IFA$="J"THEN1010
1140 PRINTCHR$(147)CHR$(154)
1150 POKE53280,14:END
```

Bug Bovver

(Wanzenstampfer)



Diejenigen unter uns, die unter regelmäßigen Termitenüberfällen leiden oder die durch Insekten an den Rand der Verzweiflung getrieben worden sind, können sich in diesem Spiel revanchieren.

Nachdem das Spiel gestartet ist, bewegt sich ein Stiefel bedrohlich am oberen Rand des Schirms auf und ab. Wenn Sie die Leertaste drücken, senkt sich der Stiefel. Beachten Sie das knirschende Geräusch, das entsteht, wenn Sie den Käfer plätten.

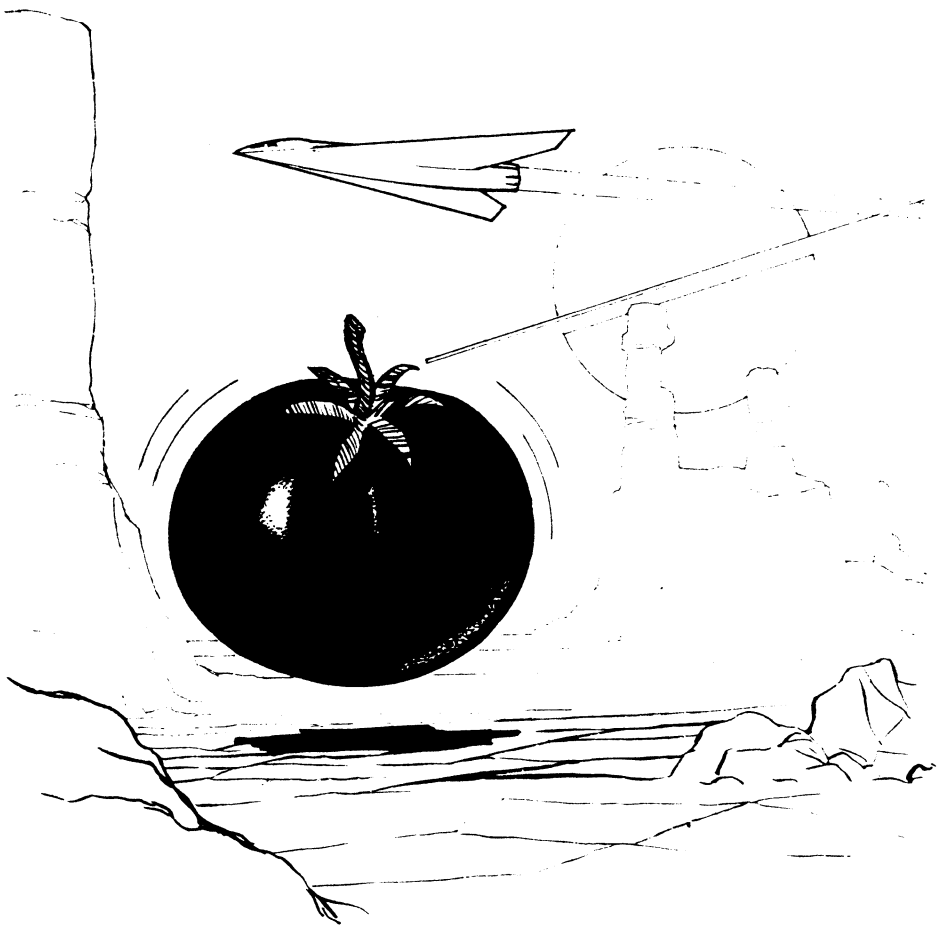
```
1000 REM *** BUG BOVVER ***
1010 PRINTCHR$(147)
1020 POKE53280,0:POKE53281,0
1030 GOSUB1720
1040 PRINTCHR$(152)
1050 TI$="000000"
1060 FORI=0TO62:READJ:POKEI+832,J:NEXT
1070 FORI=0TO62:READJ:POKEI+896,J:NEXT
1080 FORI=0TO62:READJ:POKEI+960,J:NEXT
1090 POKE2040,13:POKE2041,14
1100 VC=53248
1110 POKEVC+23,6:POKEVC+29,6
1120 POKEVC+39,9
1130 POKEVC+40,5
1140 POKEVC+41,5
1150 POKEVC+37,2:POKEVC+38,7
1160 BD=8:DY=8:BY=60:POKEVC+1,BY
1170 GD=4:POKEVC+3,200
1180 BX=INT(RND(1)*140+56)
1190 GX=INT(RND(1)*115+66)
1200 REM***BUG BOVVER***
1210 S=54272:D=-1:SO=38
1220 POKES+24,14
1230 POKES+5,31:POKES+6,240
1240 POKE781,22:POKE782,0:SYS65520
1250 FORI=1TO40:PRINT"▂";:NEXT
1260 PRINTCHR$(19)"PUNKTE:"SPC(21)"MAX.:
"
1270 PRINTCHR$(19):FORI=1TO120
1280 PRINTCHR$(18)" ";:NEXT
1290 POKEVC+21,3
1300 REM *** HAUPTSCHLEIFE ***
1310 BX=BX+BD:IFBX<55ORBX>246THENBD=-BD
1320 POKES+4,0:POKEVC+30,0
1330 POKEVC,BX:POKEVC+1,BY
1340 GX=GX+GD:IFGX<45ORGX>240THENGD=-GD
1350 POKEVC+2,GX
1360 IFSC>HITHENHI=SC
1370 PRINTCHR$(19)TAB(7);SC
1380 PRINTCHR$(19)TAB(32);HI
1390 POKE781,23:POKE782,17:SYS65520
1400 PRINTMID$(TI$,4,1)":"RIGHT$(TI$,2)C
HR$(145)
1410 GETA$:IFA$=""THEN1310
1420 POKES+4,17
```

```
1430 BY=BY+DY:IFBY>210THENDY=-DY:K=1:Z=1
1440 IFBY<60THENBY=60:DY=-DY:D=-D:K=0:GO
TO1310
1450 GX=GX+GD:IFGX<45ORGX>240THENGD=-GD
1460 IFZ=1THENIFK=1THEND=-D:Z=0
1470 SO=SO+D
1480 POKEVC+2,GX
1490 POKEVC+1,BY
1500 IFK=1THEN1530
1510 IF(PEEK(VC+30)AND3)=3THENPOKEVC+30,
0:GOTO1590
1520 POKEVC+30,0
1530 POKES,200:POKES+1,SO
1540 POKE781,23:POKE782,17:SYS65520
1550 PRINTMID$(TI$,4,1)":"RIGHT$(TI$,2)C
HR$(145)CHR$(145)
1560 IFTI$>"000200"THEN1810
1570 GOTO1430
1580 REM *** PLATSCH! ***
1590 POKE2041,15:SC=SC+250-(VAL(TI$))
1600 GOSUB1690
1610 POKES+4,0:POKEVC+30,0:SO=38
1620 GX=GX-2:IFGX<4THEN1650
1630 POKEVC+2,GX
1640 GOTO1620
1650 POKEVC+21,0
1660 FORT=1TO200:NEXT
1670 GOTO1090
1680 REM *** PLATSCH-GERAEUSCH ***
1690 FORI=1TO25:POKEVC+28,2
1700 POKES+4,0:POKES+4,129:NEXT
1710 POKEVC+28,0:RETURN
1720 REM *** TITELBILD ***
1730 PRINTCHR$(17)CHR$(17)CHR$(30)
1740 PRINTTAB(8)"
"
1750 PRINTTAB(8)"
"
1760 PRINTTAB(8)"
"
1770 PRINTCHR$(17)CHR$(17)CHR$(17)
1780 PRINTTAB(12)"TASTE DRUECKEN"
1790 GETB$:IFB$=""THEN1790
1800 PRINTCHR$(147):RETURN
1810 REM *** ZEIT ABGELAUFEN ***
```

```
1820 POKES+4,0
1830 PRINTCHR$(19)CHR$(17)CHR$(17)CHR$(1
7)
1840 PRINTCHR$(17)
1850 PRINTTAB(12)"-ZEIT ABGELAUFEN!-"
1860 PRINTCHR$(17)CHR$(17)CHR$(17)
1870 PRINTTAB(4)"TASTE DRUECKEN FUER NEU
ES SPIEL"
1880 GETB$:IFB$=""THEN1880
1890 TI$="000000"
1900 SC=0:PRINTCHR$(147):GOTO1090
1910 REM *** SPRITE 0 (STIEFEL) ***
1920 DATA0,0,0,124,0,0,63,248,0,63,248
1930 DATA0,63,240,0,31,192,0,31,240,0
1940 DATA31,192,0,31,240,0,15,192,0,15
1950 DATA248,0,15,228,0,31,254,0,31,249
1960 DATA176,63,255,184,0,15,188,127
1970 DATA240,188,127,255,0,127,143,254
1980 DATA127,128,254,0,0,0
1990 REM *** SPRITE 1 (KAEFER) ***
2000 DATA0,0,0,0,124,0,0,254,0,1,187,0
2010 DATA3,17,128,3,1,128,7,69,192,7,17
2020 DATA192,7,255,192,1,199,0,0,124,0
2030 DATA0,108,0,0,108,0,0,108,0,3,239
2040 DATA128,0,0,0,0,0,0,0,0,0,0,0,0,0
2050 DATA0,0,0,0,0
2060 REM *** SPRITE 2 (PLATSCH) ***
2070 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2080 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2090 DATA0,0,0,7,255,224,62,90,124,127
2100 DATA255,254,0,195,0,7,195,224,0,0
2110 DATA0,0,0,0,0,0,0,0,0,0,0,0
```

Attack of the Tomato

(Angriff der Killertomaten)



Ihre Heimatstadt wird von überdüngten Tomaten angegriffen. Dieses gemeine Gemüse ist tödlich! Als Kommandant eines Front-Vorpostens lautet Ihr Auftrag

VERWANDELN SIE DIE TOMATEN IN RATATOUILLE!!!

Wenn Ihnen mehr als fünf entkommen sind, ertrinkt Ihre Stadt im KETCHUP.

```
1000 REM *** ATTACK OF THE TOMATO ***
1010 PRINTCHR$(147):POKE53281,0:POKE5328
0,0
1020 PRINTTAB(45)CHR$(28)"
1030 PRINTTAB(4)"
1040 PRINTTAB(4)"
1050 PRINTTAB(4)"
1060 PRINTTAB(4)"
1070 PRINT:PRINT
1080 PRINTTAB(6)"
1090 PRINTTAB(6)"
1100 PRINTTAB(6)"
1110 PRINTTAB(6)"
1120 PRINTTAB(6)"
1130 PRINT:PRINT
1140 PRINTTAB(4)"
1150 PRINTTAB(4)"
1160 PRINTTAB(6)"
1170 PRINTTAB(6)"
1180 PRINTTAB(6)"
1190 PRINT:PRINT
1200 PRINTTAB(12)"<<TASTE DRUECKEN>>"
1210 GETJ$:IFJ$<>""THENGOTO1210
1220 GETA$:IF A$=""THEN1220
1230 V=53248:N=4:LX2=100:MX=0:Y2=100
1240 GOSUB1940:REM BASICLADER FUER 'JOYS
TICK' AUFRUFEN
1250 PRINTCHR$(147):PRINTCHR$(5):REM BIL
DSCHIRM LOESCHEN
1260 GOSUB1780:REM BILDSCHIRM AUFBAUEN
```

```
1270 REM *** SPRITES ***
1280 REM *** SPRITE-POINTER SETZEN ***
1290 POKE2040,192:POKE2041,193
1300 POKE2042,195:POKE2044,196
1310 REM *** SPRITE-DATEN LESEN ***
1320 FORI=12288TO12350:READA:POKEI,A:NEX
T
1330 FORI=12352TO12414:READA:POKEI,A:NEX
T
1340 FORI=12416TO12478:READA:POKEI,A:NEX
T
1350 FORI=12480TO12542:READA:POKEI,A:NEX
T
1360 FORI=12544TO12606:READA:POKEI,A:NEX
T
1370 FORI=12736TO12798:READA:POKEI,A:NEX
T
1380 REM *** SPRITES FAERBEN ***
1390 POKEV+41,1:POKEV+43,7:POKEV+39,5
1400 REM *** SPRITES VERGROESSERN ***
1410 POKEV+23,3:POKEV+29,11
1420 REM *** SPRITES EINSCHALTEN ***
1430 POKEV+21,7
1440 REM *** JOYSTICK-PARAMETER SETZEN *
**
1450 POKE49152,0:POKE49153,0:REM  X2 MIN
1460 POKE49156,65:POKE49157,1:REM X2 MAX
1470 POKE49160,47:POKE49161,0:REM Y2 MIN
1480 POKE49164,232:POKE49165,0:REM Y2 MI
N
1490 POKE49168,0:REM ABFRAGEHAEUFIGKEIT
1500 SYS49183:REM MS-PGR AKTIVIEREN
1510 REM *** HAUPTROUTINE FUER BEWEGUNG
***
1520 REM *** NEUE TOMATE ***
1530 X0=345:POKEV+16,PEEK(V+16)OR3
1540 POKE49169,LX2:POKE49170,MX
1550 POKE49173,Y2:POKE49174,0
1560 POKEV,90:POKEV+2,90
1570 POKE2041,193:POKEV+40,2:REM SPRITE-
POINTER UND FARBE ZURUECKSETZEN
1580 POKEV+21,PEEK(V+21)OR3:REM TOMATE W
IEDER EINSCHALTEN
1590 REM *** BEWEGUNGSSCHLEIFE ***
1600 X0=X0-N:Y0=140-40*SIN(X0/N)
```

```
1610 PRINTTAB(3)CHR$(145)"DURCHGELASSEN"
;T,"PUNKTZAHL";YS
1620 GOSUB2690:REM STEUERUNG DES JAEGERS
1630 Y1=Y0+14
1640 HX=INT(X0/256):LX=X0-256*HX
1650 IF HX=0THEN POKEV+16,PEEK(V+16)AND2
52
1660 POKEV,LX:POKEV+1,Y0:REM SPR 0 POS
1670 POKEV+2,LX:POKEV+3,Y1:REM SPR 1 POS
1680 GOSUB2690:REM STEUERUNG DES JAEGERS
1690 IFC>=5THENC=0:YS=YS+100:N=N+2:GOTO1
530
1700 REM *** TOMATE ENTKOMMEN ***
1710 IFX0<10THENT=T+1:C=0:GOTO1530
1720 IFT>5THEN1740:REM ANZAHL DURCHGEKOM
MENER TOMATEN>5?
1730 GOTO1600
1740 PRINTCHR$(147):GOSUB2880:REM ENDE D
ES SPIELS
1750 SYS49208:REM MS-PRG DESAKTIVIEREN
1760 RUN1230
1770 END
1780 REM *** BILDSCHIRM AUFBAUEN ***
1790 POKE53280,0:REM RAND FAERBEN
1800 POKE53281,6:REM HINTERGRUND FAERBEN
1810 REM *** DATEN FUER HINTERGRUND (BER
GE) LESEN ***
1820 FORI=1464TO1663:READA:POKEI,A
1830 POKE54272+I,4:NEXT
1840 FORI=1664TO1743:POKEI,160
1850 POKE54272+I,4:NEXT
1860 FORI=1744TO2023:POKEI,160
1870 POKE54272+I,14:NEXT
1880 REM *** STERNE SETZEN ***
1890 FORI=1TO40
1900 P=1024+INT(RND(1)*440)
1910 POKEP,46:POKEP+54272,1
1920 NEXT
1930 RETURN
1940 REM *** DATEN FUER MSPRG 'JOYSTICK'
 ***
1950 DATA120,173,20,3,141,29,192,173,21
1960 DATA3,141,30,192,169,71,141,20,3
1970 DATA169,192,141,21,3,88,96,120,173
1980 DATA29,192,141,20,3,173,30,192,141
```

```
1990 DATA21,3,88,96,173,27,192,205,16
2000 DATA192,240,3,76,65,193,169,255
2010 DATA141,27,192,162,0,160,0,185,2
2020 DATA220,141,28,192,169,224,153,2
2030 DATA220,185,0,220,72,173,28,192
2040 DATA153,2,220,189,8,192,221,21,192
2050 DATA189,9,192,253,22,192,144,18
2060 DATA189,8,192,157,21,192,189,9,192
2070 DATA157,22,192,104,74,72,76,162
2080 DATA192,104,74,72,176,13,222,21
2090 DATA192,189,21,192,201,255,208,3
2100 DATA222,22,192,189,21,192,221,12
2110 DATA192,189,22,192,253,13,192,144
2120 DATA18,189,12,192,157,21,192,189
2130 DATA13,192,157,22,192,104,74,72,76
2140 DATA207,192,104,74,72,176,8,254,21
2150 DATA192,208,3,254,22,192,189,0,192
2160 DATA221,17,192,189,1,192,253,18
2170 DATA192,144,18,189,0,192,157,17
2180 DATA192,189,1,192,157,18,192,104
2190 DATA74,72,76,1,193,104,74,72,176
2200 DATA13,222,17,192,189,17,192,201
2210 DATA255,208,3,222,18,192,189,17
2220 DATA192,221,4,192,189,18,192,253,5
2230 DATA192,144,18,189,4,192,157,17
2240 DATA192,189,5,192,157,18,192,104
2250 DATA74,72,76,46,193,104,74,72,176
2260 DATA8,254,17,192,208,3,254,18,192
2270 DATA104,74,176,5,169,1,153,25,192
2280 DATA200,232,232,224,4,240,3,76,91
2290 DATA192,238,27,192,108,29,192
2300 DATA35994:REM*PRUEFSUMME*
2310 FORI=49183TO49478
2320 READX:POKEI,X:CC=CC+X
2330 NEXT
2340 READX:IFX<>CCTHEN PRINT"FEHLER IN D
EN DATEN"
2350 RETURN
2360 REM *** UNTERPRG FUER SCHUSS ***
2370 X4=X2+12
2380 POKEV+21,PEEK(V+21)OR16:REM SCHUSS
EINSCHALTEN
2390 YS=YS-10
2400 IFPEEK(2042)=194THEN DX=-30
2410 L=0
```

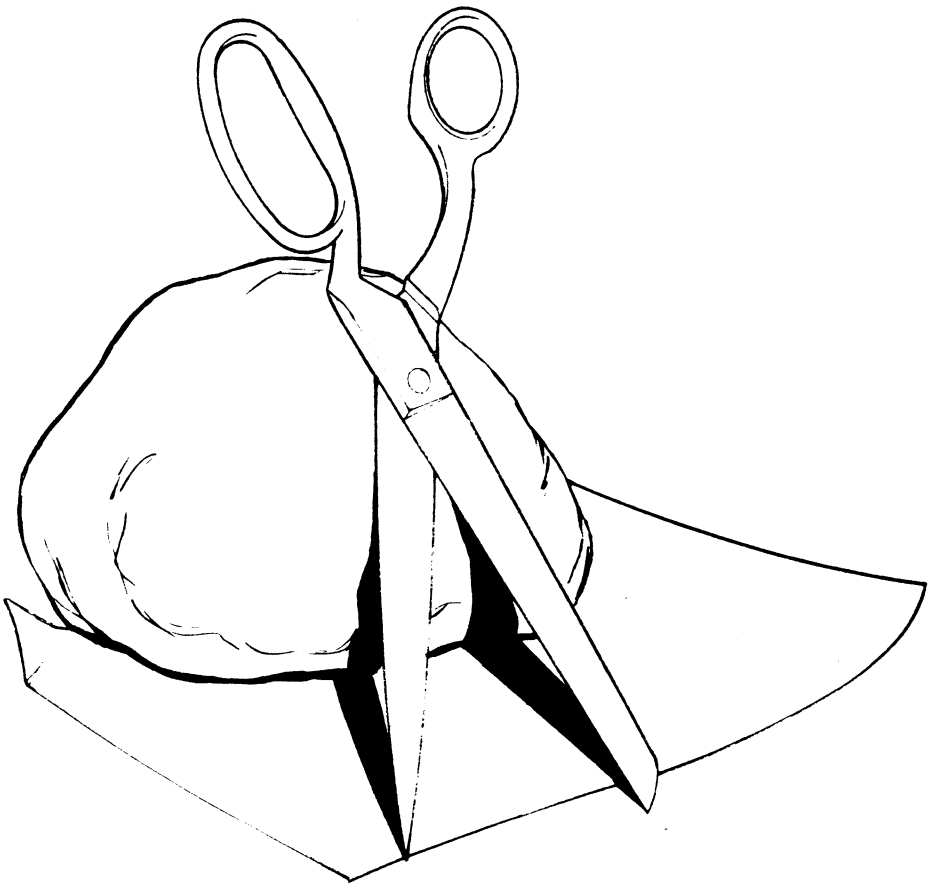
```
2420 IFPEEK(2042)=195THEN DX=30
2430 X4=X4+DX
2440 L=L+1:IFL>4THENGOTO2540
2450 H4=INT(X4/256):L4=X4-256*H4
2460 IFH4=1THENPOKEV+16,PEEK(V+16)OR16
2470 IFH4=0THENPOKEV+16,PEEK(V+16)AND239
2480 POKEV+8,L4
2490 REM JOYSTICK-PARAMETER SETZEN
2500 IFX4<0ORX4>320THENPOKEV+21,PEEK(V+2
1)AND239:GOTO2540
2510 PC=PEEK(V+30):REM KOLLISION REGISTR
IEREN
2520 IFPC=19THENPOKEV+21,PEEK(V+21)AND23
9:GOSUB2550
2530 GOTO2430
2540 RETURN
2550 REM *** KOLLISIONSROUTINE ***
2560 C=C+0.5
2570 REM *** TOMATENFARBE AENDERN ***
2580 ONC+1GOTO2620,2630,2640,2650,2660
2590 REM *** TOMATE EXPLODIEREN LASSEN U
ND ABSCHALTEN ***
2600 POKE2041,199:FORI=1TO2000:NEXT
2610 POKEV+21,PEEK(V+21)AND252:GOTO2670
2620 POKEV+40,8:GOTO2670
2630 POKEV+40,10:GOTO2670
2640 POKEV+40,9:GOTO2670
2650 POKEV+40,11:GOTO2670
2660 POKEV+40,0:GOTO2670
2670 YS=YS+20
2680 RETURN
2690 REM ***  STEUERUNG  ***
2700 REM *** DES JAEGERS ***
2710 XL=X2:REM LETZTES X2 MERKEN
2720 IFPEEK(56320)AND16<>16THENGOSUB2360
2730 X2=PEEK(49169)+256*PEEK(49170)
2740 Y2=PEEK(49173)+256*PEEK(49174)
2750 REM SPRITE-POINTER NEU SETZEN
2760 IFX2>XLTHENPOKE2042,195
2770 REM SPRITE-POINTER NEU SETZEN
2780 IFX2<XLTHENPOKE2042,194
2790 MX=INT(X2/256):LX2=X2-256*MX
2800 IFMX=1THENPOKEV+16,PEEK(V+16)OR20
2810 IFMX=0THENPOKEV+16,PEEK(V+16)AND235
2820 POKEV+4,LX2:POKEV+5,Y2:REM SPR 2 PO
S
```

```
2830 POKEV+8,LX2:POKEV+9,Y2:REM SPR 4 PO
S
2840 REM *** FEUERKNOPF GEDRUECKT? ***
2850 FR=PEEK(56320)AND16
2860 IFFR<>16THENGOSUB2360
2870 RETURN
2880 REM *** ENDE DES SPIELS ***
2890 POKE53281,0
2900 POKEV+21,0
2910 PRINTCHR$(28):PRINTCHR$(147)
2920 PRINT:PRINT:PRINT
2930 PRINTTAB(3)"
2940 PRINTTAB(3)"
2950 PRINTTAB(3)"
2960 PRINTTAB(3)"
2970 PRINTTAB(3)"
2980 PRINT:PRINTTAB(15)"(UEBERRANNT)":PR
INTCHR$(5)
2990 PRINT:PRINT:PRINT:PRINTTAB(10)"ERRE
ICHTE PUNKTZAHL:";YS
3000 PRINTCHR$(28)
3010 PRINT:PRINT:PRINTTAB(12)"<<TASTE DR
UECKEN>>"
3020 GETJ$:IFJ$<>""THEN3020
3030 GETB$:IFB$=""THEN3030
3040 RETURN
3050 REM *** DATEN FUER HINTERGRUND (BER
GE) ***
3060 DATA32,32,32,233,223,32,32,32,32
3070 DATA32,32,32,32,32,32,32,32,32,32
3080 DATA32,32,32,32,32,32,32,32,32,32
3090 DATA32,32,32,32,32,32,32,32,32,32
3100 DATA32,32,32,233,160,160,160,223
3110 DATA32,32,32,32,32,32,32,32,32,32
3120 DATA32,32,32,32,32,32,32,32,32,32
3130 DATA32,32,32,233,223,32,32,32,32
3140 DATA32,32,32,32,160,160,160,160
3150 DATA160,160,160,223,32,32,32,32,32
3160 DATA32,32,32,233,223,32,32,32,32
3170 DATA32,32,32,32,32,32,32,233,160
3180 DATA160,160,160,223,32,32,32,32
```

```
3190 DATA233,160,160,160,160,160,160
3200 DATA160,160,223,32,32,32,32,32,32
3210 DATA233,160,160,32,32,32,32,32,32
3220 DATA32,32,233,160,160,160,160,160
3230 DATA160,160,160,223,32,32,233,160
3240 DATA160,160,160,160,160,160,160
3250 DATA160,160,223,32,32,32,233,160
3260 DATA160,160,160,223,32,32,32,32
3270 DATA233,160,160,160,160,160,160
3280 DATA160,160,160,160,160,160,160
3290 DATA160,160,160
3300 REM *** DATEN FUER SPRITE 0 ***
3310 DATA0,0,0,0,0,0,0,0,0,0
3320 DATA192,0,0,192,0,0,192,0,0,224
3330 DATA0,0,96,0,60,115,192,39,54,32
3340 DATA1,252,0,3,159,0,6,3,128,4,0
3350 DATA128,4,0,128
3360 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
3370 DATA0,0,0
3380 REM *** DATEN FUER SPRITE 1 ***
3390 DATA0,0,0,15,131,192,31,199,224
3400 DATA63,199,240,63,239,248,127,255
3410 DATA252,127,255,252,255,255,254
3420 DATA255,255,254,255,255,254,255
3430 DATA255,254,255,255,254,255,255
3440 DATA252,127,255,252,127,255,248
3450 DATA63,255,248,63,255,240,31,255
3460 DATA224,15,255,192,7,255,0,1,252,0
3470 REM *** DATEN FUER SPRITE 2 ***
3480 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0
3490 DATA7,0,0,63,0,0,237,0,7,152,0,28
3500 DATA48,0,125,224,7,255,252,3,255
3510 DATA252,0,125,224,0,28,48,0
3520 DATA7,152,0,0,237,0,0,63,0,0,7
3530 DATA0,0,0,0,0,0,0,0,0
3540 REM *** DATEN FUER SPRITE 3 ***
3550 DATA0,0,0,0,0,0,0,0,0
3560 DATA224,0,0,252,0,0,183,0,0,25,224
3570 DATA0,12,56,0,7,190,0,63,255,224
3580 DATA63,255,192,7,190,0,12,56,0,25
3590 DATA224,0,183,0,0,252,0,0,224
3600 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
3610 REM *** DATEN FUER SPRITE 4 ***
3620 DATA0,0,0,0,0,0,0,0,0
3630 DATA0,0,0,0,0,0,0,0,0
```

```
3640 DATA0,0,0,0,0,0,0,0,0
3650 DATA195,12,48,195,12,48
3660 DATA0,0,0,0,0,0,0
3670 DATA0,0,0,0,0,0,0,0
3680 DATA0,0,0,0,0,0,0,0
3690 DATA0,0,0,0,0,0,0
3700 REM *** DATEN FUER SPRITE 1B ***
3710 DATA1,16,8,64,0,16,32,0
3720 DATA0,16,16,64,8,0 ,128,0,65
3730 DATA0,66,16,4,1,16,16,4,128
3740 DATA64,0,19,0,73,56,0,0
3750 DATA108,105,4,56,0,128,16,0
3760 DATA0,130,16,1,17,1,64,0,128
3770 DATA0,4,0,8,0,32,16,82,16,32,16,0
```

Schere, Stein, Papier



Schlagen Sie den Computer in diesem alten Kinderspiel. Die Regeln sind einfach:

Schere schneidet Papier,
Stein schleift Schere,
Papier wickelt Stein ein.

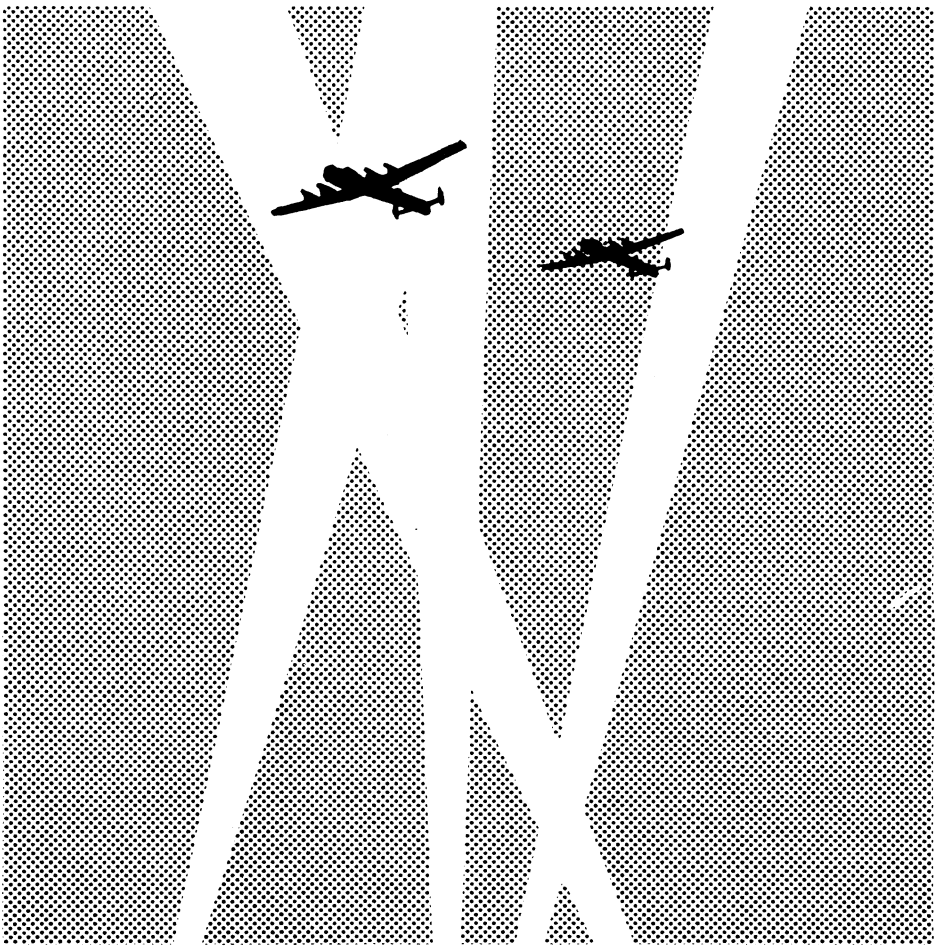
Wenn Sie entweder Schere, Stein oder Papier wählen – indem Sie E, S oder P drükken –, trifft der Computer per Zufall ebenfalls eine Wahl. Sie wählen z.B. Schere, und der Computer entscheidet sich für das Papier; dann gewinnen Sie. Wenn der Computer aber Stein gewählt hätte, dann hätte er gewonnen – weil der Stein die Schere schleift.

```
1000 REM *** SCHERE, STEIN, PAPIER ***
1010 PRINTCHR$(147)
1020 P$(1)="PAPIER":P$(2)="STEIN "
1030 P$(3)="SCHERE"
1040 C$(1)=P$(1):C$(2)=P$(2):C$(3)=P$(3)
1050 REM *** ANLEITUNG ***
1060 PRINTTAB(9)"-SCHERE/STEIN/PAPIER-"
1070 PRINTCHR$(17)CHR$(17)
1080 PRINT"DREI DINGE SIND WICHTIG:"
1090 PRINT"1) DIE SCHERE SCHNEIDET DAS P
APIER."
1100 PRINT"2) DER STEIN SCHLEIFT DIE SCH
ERE."
1110 PRINT"3) DAS PAPIER WICKELT DEN STE
IN EIN.
1120 PRINTCHR$(17)CHR$(17)
1130 PRINTTAB(9)"DRUECKE E,S ODER P"
1140 PRINTTAB(4)"FUER SCHERE, STEIN ODER
 PAPIER"
1150 PRINTCHR$(17)CHR$(17)
1160 PRINTTAB(12)"TASTE DRUECKEN"
1170 GETA$:IFA$=""THEN1170
1180 PRINTCHR$(147)
1190 REM *** WAHL DES SPIELERS ***
1200 PRINTCHR$(19)"COMPUTER:"CS
1210 PRINTCHR$(19)TAB(20)"SPIELER:"PS
1220 PRINTCHR$(17)CHR$(17)
1230 PRINT"E-(SCHERE),S-(STEIN)";
1240 PRINT"ODER P-(PAPIER)"
1250 PRINTCHR$(17)CHR$(17)
1260 GETA$
1270 IFA$<>"P"ANDA$<>"S"ANDA$<>"E"THEN12
60
1280 IFA$="P"THENGOSUB1430
1290 IFA$="S"THENGOSUB1440
1300 IFA$="E"THENGOSUB1450
1310 REM *** WAHL DES COMPUTERS ***
1320 R=INT(RND(TI)*3+1)
1330 ONRGOSUB1470,1480,1490
1340 REM *** WER HAT GEWONNEN? ***
1350 IFPC=1ANDCC=2THENPS=PS+1
1360 IFPC=1ANDCC=3THENCS=CS+1
1370 IFPC=2ANDCC=1THENCS=CS+1
1380 IFPC=2ANDCC=3THENPS=PS+1
1390 IFPC=3ANDCC=1THENPS=PS+1
```

```
1400 IFPC=3ANDCC=2THENCS=CS+1
1410 GOTO1200
1420 REM *** SPIELER ***
1430 PRINTTAB(20);P$(1):PC=1:RETURN
1440 PRINTTAB(20);P$(2):PC=2:RETURN
1450 PRINTTAB(20);P$(3):PC=3:RETURN
1460 REM *** COMPUTER ***
1470 PRINTCHR$(145);C$(1):CC=1:RETURN
1480 PRINTCHR$(145);C$(2):CC=2:RETURN
1490 PRINTCHR$(145);C$(3):CC=3:RETURN
```

Searchlight

(Nachtangriff)



Ihre Bodenstation liegt im Feuer eines Nachtangriffs. Orten Sie mit Ihrem Suchscheinwerfer die feindlichen Bomber und holen Sie sie dann vom Himmel! Der Mond spendet nicht genug Licht; Sie haben aber drei Leuchtraketen, die Sie einsetzen können, wenn Sie kein Ziel finden. Schießen Sie möglichst viele Flugzeuge ab, bevor Ihre Station ausgebombt ist.

Mit Hilfe des Joysticks dirigieren Sie den Suchscheinwerfer. Mit dem Feuerknopf eröffnen Sie das Feuer. Die Leuchtraketen werden über die Leertaste abgefeuert.

```
1000 REM *** SEARCHLIGHT ***
1010 POKE53281,0:POKE53280,0:PRINTCHR$(1
47):PRINTCHR$(158)
1020 PRINT"
"
1030 PRINT"
"
1040 PRINT"
"
1050 PRINT"
  |"
1060 PRINT"
  |"
1070 PRINT"
  |"
1080 PRINT"
┘ ├"
1090 PRINT"
 ||"
1100 PRINT"
 ||"
1110 PRINT"
 ||"
1120 PRINT"
 ||"
1130 PRINT"
 | └"
1140 PRINT"                                |"
1150 PRINT"                                |"
1160 PRINT"                              | |"
1170 PRINT"                              | |"
1180 PRINT"                              | |"
1190 PRINT"                              └─┘"
1200 PRINT:PRINTTAB(11)"<<TASTE DRUECKEN
>>"
1210 DIMX(6),D(6),P(6)
1220 V=53248:FORI=1TO5:D(I)=1:P(I)=14:NE
XT
1230 FL=3:S=54272
1240 GOSUB2520:REM MS-JOYSTICK-ABFRAGE
1250 GETA$:IFA$=""THEN1250
1260 GETJ$:IFJ$<>""THEN1260
1270 GOSUB2160:REM BILDSCHIRM AUFBAUEN
1280 REM *** SPRITE-DATEN LESEN ***
1290 FORI=12288TO12350:READA:POKEI,A:NEX
T
```

```
1300 FORI=12352TO12414:READA:POKEI,A:NEX
T
1310 FORI=896TO958:READA:POKEI,A:NEXT
1320 FORI=12672TO12734:READA:POKEI,A:NEX
T
1330 FORI=12736TO12798:READA:POKEI,A:NEX
T
1340 REM *** SPRITES FAERBEN ***
1350 REM *** SPRITE 0 ***
1360 POKEV+39,2:POKEV+37,7:POKEV+38,1
1370 REM *** SPRITE 1-5 ***
1380 FORI=V+40TOV+44:POKEI,0:NEXT
1390 REM *** SPRITE 6 ***
1400 POKEV+45,7
1410 REM *** SPRITE 7 ***
1420 POKEV+46,2
1430 POKEV+28,128:REM MEHRFARBENMODUS SE
TEN
1440 REM *** POINTER SETZEN ***
1450 POKE2040,192:REM LUFTEXPLOSION
1460 POKE2046,198:REM SUCHSCHEINWERFER
1470 POKE2047,199:REM BODENEXPLOSION
1480 REM *** JOYSTICK-PARAMETER SETZEN *
**
1490 POKE49152,0:POKE49153,0:REM X2 MIN
1500 POKE49156,113:POKE49157,1:REM X2 MA
X
1510 POKE49160,0:POKE49161,0:REM Y2 MIN
1520 POKE49164,192:POKE49165,0:REM Y2 MA
X
1530 POKE49168,0:REM ABFRAGEHAEUFIGKEIT
1540 SYS49183:REM JOYSTICK-ABFRAGE
1550 REM *** HINTERGRUND HAT PRIORITAET
UEBER SUCHSCHEINWERFER ***
1560 POKEV+27,64
1570 REM *** SPRITES 6,7 VERGROESSERN **
*
1580 POKEV+23,192:POKEV+29,192
1590 REM *** POSITION DES SCHEINWERFERS
SETZEN ***
1600 POKE49169,41:POKE49170,1
1610 POKE49173,129:POKE49174,0
1620 REM *** SPRITE 6 ANSCHALTEN ***
1630 POKEV+21,64
1640 REM *** BOMBER-FORMATION ***
1650 F=INT(RND(1)*3)
```

```
1660 IFF=0THENGOSUB2940:REM FORMATION 1
1670 IFF=1THENGOSUB2990:REM FORMATION 2
1680 IFF=2THENGOSUB3040:REM FORMATION 3
1690 FORB=1TO5:POKEV+2*B,X(B):NEXT
1700 POKEV+3,Y1:POKEV+5,Y2:POKEV+7,Y3:PO
KEV+9,Y4:POKEV+11,Y5
1710 POKEV+9,Y4:POKEV+11,Y5
1720 POKEV+21,PEEK(V+21)OR62:REM FORMATI
ON ANSCHALTEN
1730 REM *** HAUPTSCHLEIFE ***
1740 C=C+1
1750 GOSUB3090:REM SCHEINWERFER BEWEGEN
1760 FORB=1TO5
1770 IFX(B)>254THEND(B)=-1:P(B)=193
1780 IFX(B)<1THEND(B)=1:P(B)=14
1790 POKEV+2*B,X(B)
1800 POKE2040+B,P(B)
1810 GOSUB3090:REM SCHEINWERFER BEWEGEN
1820 X(B)=X(B)+D(B):NEXT
1830 IFC/20=INT(C/20)THEN GOSUB3440
1840 IFEN=1THEN1890:REM AUF ENDE PRUEFEN
1850 IF(PEEK(V+21)AND62)=0THEN1640
1860 REM *** LEUCHTKUGEL ***
1870 GETF$:IFF$=" "ANDFL<>0THENGOSUB4090
1880 GOTO1730:REM SCHLEIFE WIEDER STARTE
N
1890 REM *** SPIELENDE ***
1900 PRINTCHR$(147):REM BILDSCHIRM LOESC
HEN
1910 POKEV+21,0:REM SPRITES AUSSCHALTEN
1920 REM *** AUSGEBOMBT! ***
1930 POKE53281,0:POKE53280,0
1940 PRINT"   ├─┐                    ";:PRINT
"   ┌─┘ "
1950 PRINT"   | |          |         |";:PRINT
"   | |           |"
1960 PRINT"   | |          |         |";:PRINT
"   | |           |"
1970 PRINT"   | |          |         |";:PRINT
"   | |     |     |"
1980 PRINT"   | |          |         |";:PRINT
"   | |     |     |"
1990 PRINT"   | |          |         |";:PRINT
"   | |     |     |"
2000 PRINT"   | |          |         |";:PRINT
"   | |     |     |"
```

```
2010 PRINT"   ├─┴┐ ┌┐ ┌┐ ├┐ ┌┐ ┌┐ ┌┤";:PRINT
"   | || | +     |"
2020 PRINT"   |   || ||||| || || |";:PRINT
"   | || | |     |"
2030 PRINT"   |   || ||||| || || |";:PRINT
"   | || | |     |"
2040 PRINT"   |   || ||||| ||└─| |";:PRINT
"   | || | |     |"
2050 PRINT"   |   || || || ||   | |";:PRINT
"   | || | |     |"
2060 PRINT"   |   || || || || || |";:PRINT
"   | || | | |   |"
2070 PRINT"   |   || || || || || |";:PRINT
"   | || | | |  "
2080 PRINT"   └──┘ └─| |└─┘ └─┘ └─┤";:PRINT
"   └─┘ └─┘ └─┘   o"
2090 PRINT:PRINTTAB(15)"(AUSGEBOMBT)":PR
INT:PRINTTAB(11)"PUNKTZAHL:";YS
2100 PRINT:PRINTTAB(11)CHR$(129);"<<TAST
E DRUECKEN>>"
2110 SYS49208:REM MS-PGR DESAKTIVIEREN
2120 REM *** SCHEINWERFER POSITIONIEREN
***
2130 POKEV+12,41:POKEV+16,1:POKEV+13,129
2140 RUN1210
2150 END
2160 REM *** BILDSCHIRM AUFBAUEN ***
2170 PRINTCHR$(147):REM BILDSCHIRM LOESC
HEN
2180 POKE53280,6:POKE53281,0
2190 PRINTCHR$(145);CHR$(158)"PUNKTZAHL:
 0"
2200 PRINTTAB(18)CHR$(145)"LEUCHTRAKETEN
";FL
2210 REM *** ERDBODEN ***
2220 FORI=1944TO2023:POKEI,160
2230 POKEI+54272,11:NEXT
2240 REM *** TURM ***
2250 FORI=1908TO1708STEP-40
2260 POKEI,106:POKEI+1,86:POKEI+2,116
2270 POKEI+54272,11:POKEI+1+54272,11
2280 POKEI+2+54272,11:NEXT
2290 POKE1629,66:POKE1629+54272,11
2300 POKE1669,66:POKE1669+54272,11
2310 REM *** 1. GEBAEUDE ***
2320 FORI=1920TO1923:POKEI,160
```

```
2330 POKEI-40,160:POKEI+54272,11:POKEI-4
0+54272,11:NEXT
2340 POKE1879,233:POKE1884,223
2350 POKE1879+54272,11:POKE1884+54272,11
2360 REM *** 2. GEBAEUDE ***
2370 FORI=1929TO1933:POKEI,160
2380 POKEI+54272,11:NEXT
2390 POKE1889,160:POKE1889+54272,11
2400 POKE1890,160:POKE1890+54272,11
2410 REM *** OELTANKS ***
2420 FORI=1939TO1943:POKEI,81
2430 POKEI+54272,11:NEXT
2440 FORI=1900TO1902:POKEI,81
2450 POKEI+54272,11:NEXT
2460 REM *** STERNE ***
2470 FORI=1TO30
2480 P=INT(RND(1)*560)
2490 POKE1064+P,46:POKE1064+54272+P,1
2500 NEXT
2510 RETURN
2520 REM *** DATEN DES MS-PGR ***
2530 DATA120,173,20,3,141,29,192,173,21
2540 DATA3,141,30,192,169,71,141,20,3
2550 DATA169,192,141,21,3,88,96,120,173
2560 DATA29,192,141,20,3,173,30,192,141
2570 DATA21,3,88,96,173,27,192,205,16
2580 DATA192,240,3,76,65,193,169,255
2590 DATA141,27,192,162,0,160,0,185,2
2600 DATA220,141,28,192,169,224,153,2
2610 DATA220,185,0,220,72,173,28,192
2620 DATA153,2,220,189,8,192,221,21,192
2630 DATA189,9,192,253,22,192,144,18
2640 DATA189,8,192,157,21,192,189,9,192
2650 DATA157,22,192,104,74,72,76,162
2660 DATA192,104,74,72,176,13,222,21
2670 DATA192,189,21,192,201,255,208,3
2680 DATA222,22,192,189,21,192,221,12
2690 DATA192,189,22,192,253,13,192,144
2700 DATA18,189,12,192,157,21,192,189
2710 DATA13,192,157,22,192,104,74,72,76
2720 DATA207,192,104,74,72,176,8,254,21
2730 DATA192,208,3,254,22,192,189,0,192
2740 DATA221,17,192,189,1,192,253,18
2750 DATA192,144,18,189,0,192,157,17
2760 DATA192,189,1,192,157,18,192,104
2770 DATA74,72,76,1,193,104,74,72,176
```

```
2780 DATA13,222,17,192,189,17,192,201
2790 DATA255,208,3,222,18,192,189,17
2800 DATA192,221,4,192,189,18,192,253,5
2810 DATA192,144,18,189,4,192,157,17
2820 DATA192,189,5,192,157,18,192,104
2830 DATA74,72,76,46,193,104,74,72,176
2840 DATA8,254,17,192,208,3,254,18,192
2850 DATA104,74,176,5,169,1,153,25,192
2860 DATA200,232,232,224,4,240,3,76,91
2870 DATA192,238,27,192,108,29,192
2880 DATA35994:REM*PRUEFSUMME*
2890 FORI=49183TO49478
2900 READX:POKEI,X:CC=CC+X
2910 NEXT
2920 READX:IFX<>CCTHENPRINT"FEHLER IN DE
N DATEN"
2930 RETURN
2940 REM *** FORMATION 1 ***
2950 X(1)=0:Y1=115:X(2)=96:Y2=90
2960 X(3)=24:Y3=75:X(4)=56:Y4=60
2970 X(5)=144:Y5=110
2980 RETURN
2990 REM *** FORMATION 2 ***
3000 X(1)=0:Y1=58:X(2)=48:Y2=75
3010 X(3)=36:Y3=90:X(4)=70:Y4=60
3020 X(5)=96:Y5=90
3030 RETURN
3040 REM *** FORMATION 3 ***
3050 X(1)=0:Y1=60:X(2)=48:Y2=90
3060 X(3)=96:Y3=120:X(4)=144:Y4=156
3070 X(5)=192:Y5=170
3080 RETURN
3090 REM *** SCHEINWERFER BEWEGEN ***
3100 X6=PEEK(49169)+256*PEEK(49170)
3110 Y6=PEEK(49173)+256*PEEK(49174)
3120 H6=INT(X6/256):L6=X6-256*H6
3130 IFH6=1THENPOKEV+16,PEEK(V+16)OR65:G
OTO3150
3140 POKEV+16,PEEK(V+16)AND190
3150 POKEV+12,L6:POKEV+13,Y6
3160 REM *** FEUERKNOPF ***
3170 POKE49177,0
3180 FB2= PEEK(49177)
3190 IFFB2=1THENGOSUB3210:REM FIRE
3200 RETURN
3210 REM *** FEUERN ***
```

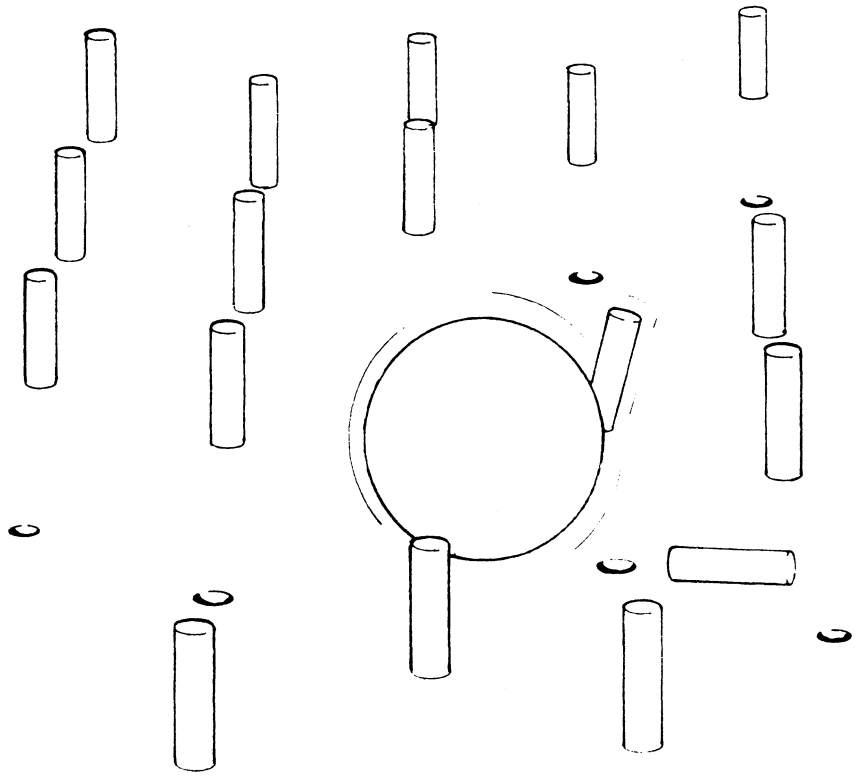
```
3220 POKE(V+30),0:REM KOLLISIONSREGISTER
 LOESCHEN
3230 L0=L6+12
3240 IFL0>255THENPOKEV+16,PEEK(V+16)OR1:
L0=L0-255
3250 PV=PEEK(V+21)
3260 POKEV+21,PVOR1
3270 POKEV,L0:POKEV+1,Y6+11:FORI=1TO20:N
EXT
3280 GOSUB3330
3290 POKE(V+21),PVAND(255-(PEEK(V+30)AND
63))
3300 IFPEEK(V+21)<PVTHENYS=YS+100:PRINTC
HR$(145);"PUNKTZAHL:";YS:C=C-5
3310 IFC<0THENC=1
3320 RETURN
3330 REM *** GERAEUSCHEFFEKT NR.2 ***
3340 S=54272
3350 POKES+24,15
3360 POKES+6,128
3370 POKES+4,129
3380 POKES,75:POKES+1,3
3390 POKES+5,131
3400 FORI=15TO0STEP-.4
3410 POKES+24,I
3420 NEXT
3430 RETURN
3440 REM *** BODENEXPLOSION ***
3450 POKEV+15,192:REM Y-KOORDINATE SETZE
N
3460 POKEV+16,PEEK(V+16)AND127
3470 A=A+1:ONAGOTO3480,3520,3560,3610,36
50,3700,3740,3790,3850
3480 GOSUB3920
3490 POKEV+14,20:POKEV+21,PEEK(V+21)OR12
8:FORI=1TO70:NEXT
3500 POKEV+21,PEEK(V+21)AND127
3510 GOSUB4020:GOTO3910
3520 GOSUB3920
3530 POKEV+14,92:POKEV+21,PEEK(V+21)OR12
8:FORI=1TO70:NEXT
3540 POKEV+21,PEEK(V+21)AND127
3550 GOSUB4020:GOTO3910
3560 GOSUB3920
3570 POKEV+14,132:POKEV+21,PEEK(V+21)OR1
28:FORI=1TO70:NEXT
```

```
3580 FORI=1919TO1921:POKEI,32:POKEI-40,3
2:NEXT
3590 POKEV+21,PEEK(V+21)AND127
3600 GOSUB4020:GOTO3910
3610 GOSUB3920
3620 POKEV+14,252:POKEV+21,PEEK(V+21)OR1
28:FORI=1TO70:NEXT
3630 POKEV+21,PEEK(V+21)AND127
3640 GOSUB4020:GOTO3910
3650 GOSUB3920
3660 POKEV+14,44:POKEV+21,PEEK(V+21)OR12
8:FORI=1TO70:NEXT
3670 FORI=1629TO1909STEP40:POKEI,32:POKE
I-1,32:POKEI+1,32:NEXT
3680 POKEV+21,PEEK(V+21)AND127
3690 GOSUB4020:GOTO3910
3700 GOSUB3920
3710 POKEV+14,188:POKEV+21,PEEK(V+21)OR1
28:FORI=1TO70:NEXT
3720 POKEV+21,PEEK(V+21)AND127
3730 GOSUB4020:GOTO3910
3740 GOSUB3920
3750 POKEV+14,220:POKEV+21,PEEK(V+21)OR1
28:FORI=1TO70:NEXT
3760 FORI=1929TO1933:POKEI,32:POKEI-40,3
2:NEXT
3770 POKEV+21,PEEK(V+21)AND127
3780 GOSUB4020:GOTO3910
3790 GOSUB3920
3800 POKEV+14,37:POKEV+16,PEEK(V+16)OR12
8
3810 POKEV+21,PEEK(V+21)OR128:FORI=1TO70
:NEXT
3820 FORI=1939TO1943:POKEI,32:POKEI-40,3
2:NEXT
3830 POKEV+21,PEEK(V+21)AND127
3840 GOSUB4020:GOTO3910
3850 GOSUB3920
3860 POKEV+14,156:POKEV+21,PEEK(V+21)OR1
28:FORI=1TO200:NEXT
3870 FORI=1922TO1924:POKEI,32:POKEI-40,3
2:NEXT
3880 POKEV+21,PEEK(V+21)AND127
3890 GOSUB4020
3900 EN=1
3910 RETURN
```

```
3920 REM *** GERAEUSCHEFFEKT ***
3930 S=54272
3940 POKES+5,96:POKES+4,17:POKES+24,15
3950 POKES+6,128
3960 FORI=50000TO14000STEP-500
3970 HF=INT(I/256):LF=I-256*HF
3980 POKES,LF:POKES+1,HF
3990 NEXT
4000 POKES+24,0
4010 RETURN
4020 REM *** EXPLOSION ***
4030 POKES+4,129:POKES,127:POKES+1,5
4040 FORI=15TO0STEP-.05
4050 POKES+24,I
4060 NEXT
4070 RETURN
4080 END
4090 REM *** LEUCHTKUGEL ***
4100 GOSUB3920:REM SOUND
4110 FL=FL-1:IFFL<0THENFL=0
4120 FORI=1TO10
4130 POKE53281,6:FORJ=1TO50:NEXT
4140 POKE53281,1:FORJ=1TO50:NEXT
4150 NEXT
4160 POKE53281,0
4170 PRINTTAB(18)CHR$(145)"LEUCHTRAKETEN
";FL
4180 RETURN
4190 REM *** DATEN FUER LUFTEXPLOSION **
*
4200 DATA0,0,0,0,0,0,0,16,0,0,8,0,4,16
4210 DATA0,3,2,64,1,56,128,12,255,144
4220 DATA1,238,40,5,151,0,11,121,0,1
4230 DATA183,0,25,214,96,0,236,48,6,24
4240 DATA152,3,98,0,8,51,0,0,96,128,0
4250 DATA64,0,0,0,0,0,0,0
4260 REM *** DATEN FUER LINKEN BOMBER **
*
4270 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0
4280 DATA14,0,0,14,0,0,30,15,255,254
4290 DATA241,255,126,254,219,247,127
4300 DATA255,128,3,252,0,0,254,0,0,63
4310 DATA0,0,0,0,0,0,0,0,0,0,0
4320 DATA0,0,0,0,0,0,0,0,0,0,0
4330 REM *** DATEN FUER RECHTEN BOMBER *
**
```

```
4340 DATA0,0,0,0,0,0,0,0,0,0,0,0
4350 DATA112,0,0,112,0,0,120,0,0,127
4360 DATA255,240,126,255,143,239,219
4370 DATA127,1,255,254,0,63,192,0,127
4380 DATA0,0,252
4390 DATA0,0,0,0,0,0,0,0,0,0,0
4400 DATA0,0,0,0,0,0,0,0,0,0,0
4410 REM *** DATEN FUER SCHEINWERFER ***
4420 DATA0,0,0,0,126,0,1,255,128,7,255
4430 DATA224,15,255,240,31,255,248,31
4440 DATA255,248,63,255,252,63,255,252
4450 DATA127,255,254,127,255,254,63,255
4460 DATA252,63,255,252,31,255,248,31
4470 DATA255,248,15,255,240,7,255,224,1
4480 DATA255,128,0,126,0,0,0,0,0,0,0
4490 REM *** DATEN FUER BODENEXPLOSION *
**
4500 DATA0,0,0,0,0,0,0,8,0,0,24,0,0,24
4510 DATA0,0,48,0,0,170,0,0,68,0,8,110
4520 DATA0,4,176,192,0,166,128,3,105
4530 DATA128,0,51,0,5,60,0,1,154,128,2
4540 DATA124,0,0,186,0,0,92,0,0,56
4550 DATA0,0,60,0,0,24,0,0
```

Flip Flap



Die Grundidee dieses Spiels ist schon alt. Können Sie sich an die Spiele erinnern, die Sie spielten, bevor Space Invaders alles eroberte? Nein???... Da gab es ein Spiel, in dem Sie eine Münze von oben in einen Automaten werfen mußten. Die Münze bahnte sich ihren Weg schrittweise nach unten und wurde dabei zufällig von Stiften abgelenkt, die gitterförmig angeordnet waren... Es tut mir leid, daß ich völlig vergessen habe, um was es ging; es schien mir jedoch eine gute Idee für ein Spiel. So entstand Flip Flap.

Das Ziel dieses Spiels ist offenkundig. Am oberen Rand des Bildschirms bewegt sich ein Ball hin und her. Sie können jederzeit durch Druck auf eine Taste den Ball fallenlassen.

Während der Ball fällt, kollidiert er mit den Stiften, die über den Schirm verteilt sind. Jedesmal, wenn er einen Stift trifft, wird dieser eingefahren, und Sie bekommen einige Punkte. Einen Bonus gibt es, wenn Sie mit einem Ball zehn Stifte treffen (nicht unmöglich). Im Laufe des Spiels wird Ihre Aufgabe immer schwieriger. Für Stifte, die schon einmal getroffen worden sind, gibt es natürlich keine Punkte.

Das Spiel ist beendet, wenn Sie 25 Bälle fallengelassen haben (vielleicht auch schon früher, wenn Sie ein schlechter Verlierer sind!).

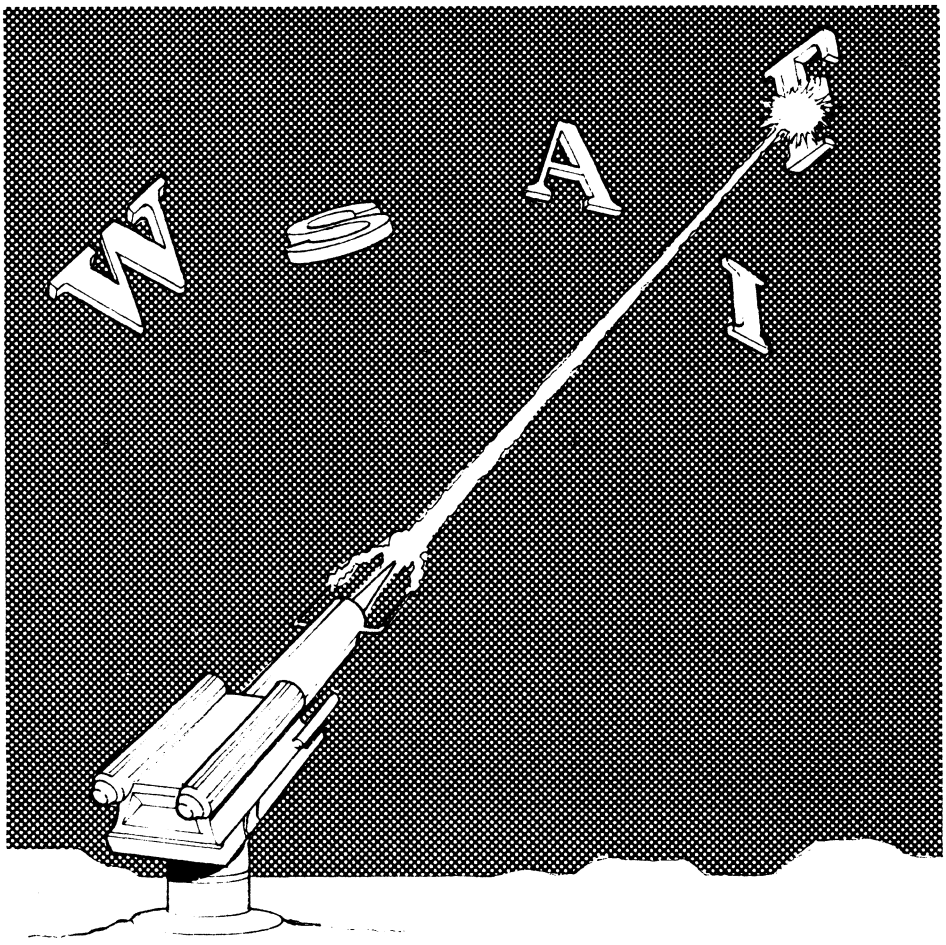
```
1000 REM *** FLIP FLAP ***
1010 GOSUB1760:REM TITELBILD
1020 GOSUB1620:REM BILDSCHIRM AUFBAUEN
1030 S=54272:REM BASISADR. DES SID-CHIP
1040 FORI=0TO24:POKES+I,0:NEXT:REM SID-R
EGISTER LOESCHEN
1050 POKES+5,8:POKES+22,74:POKES+23,1
1060 POKES+24,79
1070 REM *** VARIABLEN INITIALISIEREN **
*
1080 X=1:Y=1:D=1:SC=1024:CO=55296
1090 REM *** HAUPTSCHLEIFE ***
1100 REM *** BALL FALLENLASSEN ***
1110 OX=X:OY=Y:REM ALTE X-, Y-POSITION
1120 HT=0:IFB=25THEN1460
1130 X=X+D:IFX=0ORX>=39THEND=-D
1140 POKESC+OX+40*OY,32
1150 POKECO+X+40*Y,1:POKESC+X+40*Y,81
1160 GETA$:IFA$=""THEN1110
1170 REM *** HAUPTROUTINE ***
1180 B=B+1
1190 OX=X:OY=Y:Y=Y+1
1200 IFY>23THENGOSUB1740:GOTO1110
1210 PRINTCHR$(19)CHR$(5)"PUNKTE:"SE
1220 PRINTCHR$(19)TAB(15)"BAELLE:"B
1230 IFSE>HITHENHI=SE
1240 PRINTCHR$(19)TAB(30)"MAX.:"HI
1250 P=PEEK(SC+X+40*Y)
1260 IFP=93ORP=46THENZ=RND(1):GOSUB1320
1270 IFP=93THENSE=SE+1
1280 POKESC+OX+40*OY,32
1290 POKECO+X+40*Y,1:POKESC+X+40*Y,81
1300 GOTO1190:REM ENDE DER SCHLEIFE
1310 REM *** RICHTUNG WECHSELN ***
1320 POKECO+X+40*Y,1:POKESC+X+40*Y,46
1330 X=OX:Y=OY
1340 IFZ<.5THENX=X-1:IFX=0THENX=39
1350 IFZ>.5THENX=X+1:IFX=39THENX=0
1360 IFP=46THENGOSUB1430:RETURN
1370 POKES+0,33:POKES+1,INT(RND(1)*50+9)
1380 POKES+4,128:FORT=1TO20:NEXT
1390 HT=HT+1
1400 IFHT=10THENSE=SE+INT(RND(1)*9):HT=0
1410 POKES+4,129:RETURN
1420 REM *** TONERZEUGUNG ***
1430 POKES+0,90:POKES+1,50
```

```
1440 POKES+4,32:FORT=1TO20:NEXT
1450 POKES+4,33:RETURN
1460 REM *** SPIELENDE ***
1470 FORT=1TO25:PRINTCHR$(19)CHR$(17);
1480 PRINTCHR$(157)CHR$(148):POKE218,160
1490 NEXTT
1500 POKES+4,0:POKES+5,31:POKES+6,240
1510 PRINTCHR$(17)CHR$(17)CHR$(17)
1520 PRINTTAB(12)"-SPIELENDE!-"
1530 POKES+4,33:FORT=1TO100
1540 R=INT(RND(TI)*10+1):POKE53280,R
1550 POKES+1,1+T+R:POKES,200
1560 FORK=1TO20:NEXT
1570 NEXTT:POKES+4,0:POKE53280,0
1580 PRINTCHR$(17)TAB(4);
1590 PRINT"-TASTE DRUECKEN FUER NEUES SP
IEL-"
1600 GETA$:IFA$=""THEN1600
1610 SE=0:B=0:GOTO1010:REM NOCH EINMAL S
PIELEN
1620 REM *** BILDSCHIRM AUFBAUEN ***
1630 PRINTCHR$(147)
1640 POKE53280,0:POKE53281,0
1650 PRINTCHR$(17)
1660 PRINTCHR$(158):FORT=1TO5
1670 PRINT"|  |  |  |  |  |  |  |  |  |  ";
1680 PRINT"|  |  |  |  |  |  |  |  |  |  "
1690 PRINT"  |  |  |  |  |  |  |  |  |  |";
1700 PRINT"  |  |  |  |  |  |  |  |  |  |"
1710 NEXT
1720 RETURN
1730 REM *** LEERSTELLE POKEN ***
1740 POKESC+OX+40*OY,32
1750 Y=1:X=1:RETURN
1760 REM *** TITELBILD ***
1770 PRINTCHR$(147)CHR$(156)
1780 POKE53280,0:POKE53281,0
1790 PRINTTAB(5)"  ____________________  "
1800 PRINTTAB(5)" /                    \ "
1810 PRINTTAB(5)" |  ┌──────────────────┐  |"
1820 PRINTTAB(5)" |  |●●● o   o  ooo    |  |"
1830 PRINTTAB(5)" |  |●   o   o  o  o   |  |"
1840 PRINTTAB(5)" |  |●●  o   o  ooo    |  |"
1850 PRINTTAB(5)" |  |●   o   o  o      |  |"
1860 PRINTTAB(5)" |  |●   ooo o  o      |  |"
1870 PRINTTAB(5)" |  └──────────────────┘  |"
```

```
1880 PRINTTAB(17)"  _____                    "
1890 PRINTTAB(17)" |\                     \"
1900 PRINTTAB(17)" | ┌─────────────────────┐"
1910 PRINTTAB(17)" | | ●●● o     o   ooo   |"
1920 PRINTTAB(17)" | | ●   o    o o o  o  |"
1930 PRINTTAB(17)" | | ●● o    ooo ooo    |"
1940 PRINTTAB(17)" | | ●  o    o o o      |"
1950 PRINTTAB(17)" | | ●  ooo  o o o      |"
1960 PRINTTAB(17)" \└─────────────────────┘|"
1970 PRINTCHR$(17)CHR$(17)
1980 PRINTTAB(11)"-TASTE DRUECKEN-"
1990 GETA$:IFA$=""THEN1990
2000 RETURN
```

Laser Spell

(Buchstabieren im Jahre 2000)



Buchstabieren Sie das Wort, das am oberen Bildschirmrand eingeblendet wird, indem Sie die passenden Buchstaben abschießen, wenn sie erscheinen. Die Laserkanone wird mit dem Joystick bewegt und abgefeuert. Das Programm wählt zufällig ein Wort aus den DATA-Zeilen am Ende aus. Wenn Sie das Spiel leichter oder schwieriger machen wollen, können Sie diese Wörter natürlich ändern.

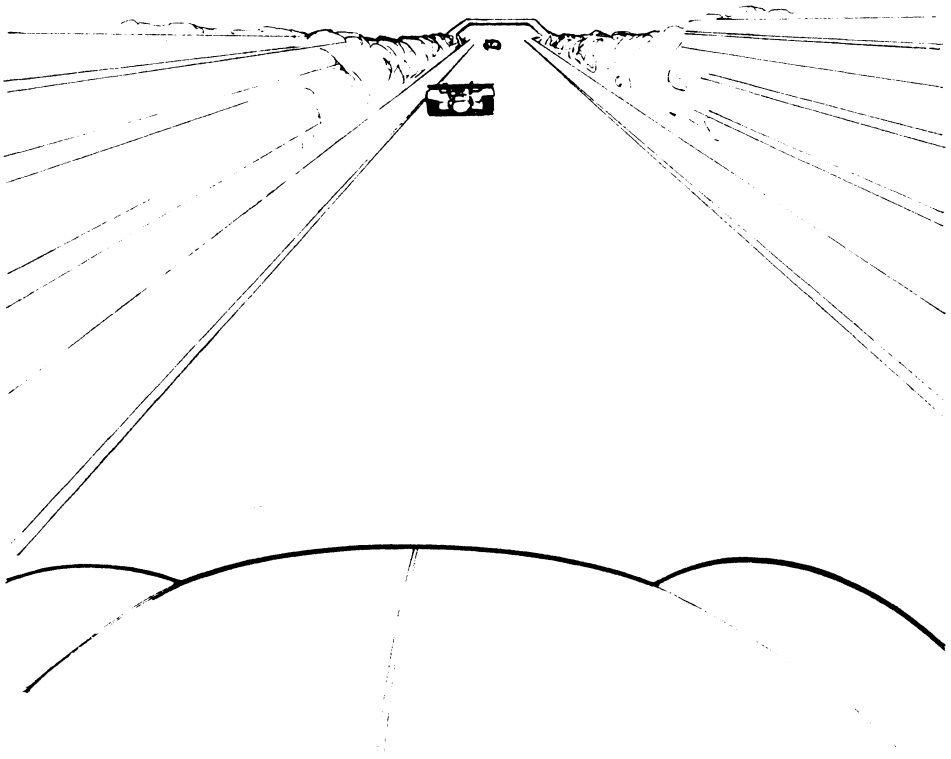
(Space Invaders mit erzieherischem Wert??!)

```
1000 REM *** LASER SPELL ***
1010 POKE53280,0:POKE53281,0
1020 DIML$(20)
1030 FORI=1TO20:READL$(I):NEXT
1040 S=54272:REM TON-REGISTER SETZEN
1050 FORI=0TO24:POKES+I,0:NEXT
1060 POKES+24,14
1070 POKES+5,31:POKES+6,240:GOSUB1670
1080 PRINTCHR$(147)
1090 POKE53280,0:POKE53281,0
1100 REM *** BILDSCHIRM AUFBAUEN ***
1110 SC=1024:CO=55296
1120 X=19:Q=-1:J=0
1130 P=INT(RND(1)*20)
1140 W$=L$(P)
1150 B$="┴":REM LASER
1160 W=LEN(W$)
1170 PRINTCHR$(19)CHR$(8);W$
1180 POKE781,23:POKE782,0:SYS65520
1190 FORI=1TO40:PRINT"━";:NEXT
1200 REM *** HAUPTSCHLEIFE ***
1210 P=PEEK(56320):REM PORT 2
1220 IFP=123THENX=X-1:IFX<0THENX=0
1230 IFP=119THENX=X+1:IFX>35THENX=35
1240 IFP=111THENGOSUB1330
1250 IFJ=WTHENGOSUB1670:GOTO1080:REM SPI
ELENDE
1260 POKE781,22:POKE782,X:SYS65520
1270 PRINT" ";B$;" "
1280 Q=Q+1
1290 IFQ=0THENGOSUB1430:REM BUCHSTABEN A
USGEBEN
1300 IFQ>50THENGOSUB1480:REM BUCHSTABEN
LOESCHEN
1310 GOTO1210
1320 REM *** LASERSTRAHL ABFEUERN ***
1330 BO=X+2:FORI=1TO20:Q=Q+1
1340 IFQ>50THENGOSUB1480:R=0
1350 POKE781,22-I:POKE782,BO:SYS65520
1360 PRINT"|"
1370 IFPEEK(SC+R)=93THENGOSUB1510:I=20
1380 POKE781,22-I:POKE782,BO:SYS65520
1390 PRINT" ":NEXT
1400 IFR=0THENQ=-1
1410 RETURN
1420 REM *** BUCHSTABEN ZUFAELLIG AUSGEB
```

```
EN ***
1430 R=INT(RND(1)*520+120)
1440 L=INT(RND(1)*26+1)
1450 POKECO+R,7:POKESC+R,L
1460 Q=0:RETURN
1470 REM *** BUCHSTABEN LOESCHEN ***
1480 POKESC+R,32
1490 Q=-1:RETURN
1500 REM *** BUCHSTABE GETROFFEN ***
1510 POKES+4,33
1520 POKESC+R,46:POKESC+R,86
1530 POKES,200:POKES+1,100
1540 FORT=1TO50:NEXT:POKESC+R,32
1550 POKES+4,0
1560 FORT=1TOLEN(W$)
1570 IFL=PEEK(SC+T-1)THENGOSUB1610
1580 NEXT
1590 RETURN
1600 REM *** BUCHSTABEN INVERTIEREN ***
1610 REM
1620 POKESC+T-1,PEEK(SC+T-1)+128
1630 T=LEN(W$)
1640 J=J+1
1650 RETURN
1660 REM *** ENDE DES SPIELS ***
1670 PRINTCHR$(147)
1680 POKES+4,33
1690 FORT=10TO50
1700 POKES,100:POKES+1,T
1710 POKE646,INT(RND(1)*15)
1720 PRINTTAB(5)"NOCH EIN SPIEL? TASTE D
RUECKEN!
1730 POKES+4,0
1740 GETA$:IFA$=""THEN1740
1750 RETURN
1760 REM *** WORT-DATEN ***
1770 DATA"MONITOR","GEMUESE"
1780 DATA"COMMODORE","COMPUTER"
1790 DATA"ALUMINIUM","WOERTERBUCH"
1800 DATA"BLEISTIFT","NACHTTISCH"
1810 DATA"JOYSTICK","SAXOPHON"
1820 DATA"KLAVIER","SPAGHETTI"
1830 DATA"TRANSISTOR","FLUSSDAMPFER"
1840 DATA"WEIHNACHT","GEBURTSTAG"
1850 DATA"STACHELBEERE","SONNENBLUME"
1860 DATA"COUCH","FAHRRAD"
```

Track Racer

(Gelände-Rennen)



In „Track Racer" müssen Sie eine Geländestrecke fahren, die stufenweise schmaler wird. Jedesmal, wenn Sie 100 Punkte gewinnen, verengt sich die Fahrbahn um eine Zeichenbreite. Wenn Sie gegen einen der Pfosten am Fahrbahnrand rasen, explodiert Ihr Wagen und löst sich in eine schillernde Wolke auf.

Gesteuert wird der Wagen mit dem Joystick; Vorsicht: Wenn Sie nicht in die Gegenrichtung steuern, driftet der Wagen weiter in Richtung Straßenrand!

Dieses Programm benutzt eine kurze Routine, die den Bildschirminhalt nach unten bewegt (scrollt). Diesen Effekt zu erreichen, ist etwas schwieriger als das normale Aufwärts-Scrollen, aber die zusätzliche Mühe lohnt sich.

```
1000 REM *** TRACK RACER ***
1010 PRINTCHR$(147)
1020 POKE53280,0:POKE53281,0
1030 GOSUB1720:REM TITELBILD
1040 REM *** SPRITE-DATEN LESEN ***
1050 FORI=0TO62:READJ:POKE832+I,J:NEXT
1060 FORI=0TO62:READJ:POKE896+I,J:NEXT
1070 REM *** SPRITES INITIALISIEREN ***
1080 POKE2040,13:VC=53248:POKEVC+21,1
1090 POKE2041,14:POKEVC+40,8
1100 POKEVC+39,14:POKEVC+23,3:POKEVC+29,
3
1110 POKEVC+0,150:POKEVC+1,180
1120 POKEVC+28,2
1130 REM *** SID-CHIP INITIALISIEREN ***
1140 S=54272:FORI=0TO24:POKES+I,0:NEXT
1150 POKES+5,31:POKES+6,240
1160 POKES+24,15
1170 L=12:X=150:W=12
1180 POKE646,14
1190 PRINTCHR$(147):SE=0
1200 FORI=1TO20
1210 PRINTTAB(L)"▒"SPC(W)"▒"
1220 NEXT:POKEVC+31,0
1230 POKES+4,33:POKES,100:POKES+1,4
1240 PRINTCHR$(19)CHR$(158)"PUNKTE"
1250 PRINTCHR$(19)TAB(28)"MAX."
1260 REM *** HAUPTSCHLEIFE ***
1270 DIR=1:IFRND(1)<.5THENDIR=2
1280 SE=SE+1:Q=Q+1
1290 IFDIR=1THENL=L-1
1300 IFL<10THENL=L+1
1310 IFDIR=2THENL=L+1
1320 IFL>20THENL=L-1
1330 PRINTCHR$(19)CHR$(17);
1340 PRINTCHR$(157)CHR$(148)
1350 POKE218,160
1360 PRINTTAB(L)"^"SPC(W)"^"
1370 P=PEEK(56320)
1380 IFP=123THENZ=1
1390 IFP=119THENZ=2
1400 ONZGOSUB1510,1530
1410 POKEVC+0,X
1420 PRINTCHR$(19)TAB(6);SE
1430 IFSE>HITHENHI=SE
```

```
1440 PRINTCHR$(19)TAB(34);HI
1450 IFQ=100THENW=W-1:Q=0
1460 IFPEEK(VC+31)AND1=1THEN1560
1470 POKEVC+31,0
1480 IFW=8THENW=12
1490 GOTO1270
1500 REM *** RICHTUNG DES AUTOS ***
1510 X=X-4:IFX<10THENX=10
1520 RETURN
1530 X=X+4:IFX>250THENX=250
1540 RETURN
1550 REM *** KOLLISION MIT LEITPLANKE **
*
1560 POKEVC+21,2
1570 POKEVC+2,X:POKEVC+3,180
1580 FORV=15TO0STEP-.2:POKES+1,2+V
1590 POKES+4,129:POKES+24,V
1600 POKEVC+37,INT(RND(TI)*15)
1610 POKEVC+38,INT(RND(TI)*15)
1620 NEXT:POKES+4,0
1630 POKEVC+21,0
1640 REM *** NEUES SPIEL ***
1650 PRINTCHR$(19)
1660 POKE646,INT(RND(TI)*15)
1670 FORI=1TO4:PRINTCHR$(17):NEXT
1680 PRINTTAB(7)"FEUERKNOPF FUER ";
1690 PRINT"NEUES SPIEL"
1700 IFPEEK(56320)<>111THEN1650
1710 GOTO1080
1720 REM *** TITELBILD ***
1730 PRINTCHR$(19)CHR$(30)
1740 PRINTTAB(7)"< T R A C K * ";
1750 PRINT"R A C E R >"
1760 PRINTCHR$(17)CHR$(17)
1770 PRINTTAB(2)"STEUERE DEIN AUTO ";
1780 PRINT"MIT DEM JOYSTICK"
1790 PRINTTAB(2)"DIE GLATTE FAHRBAHN ";
1800 PRINT"ENTLANG."
1810 PRINTTAB(2)"ABER PASS AUF! ";
1820 PRINT"DIE STRASSE WIRD"
1830 PRINTTAB(2)"SCHMALER..."
1840 PRINTCHR$(17)CHR$(17)CHR$(17)
1850 PRINTTAB(12)"TASTE DRUECKEN"
1860 GETA$:IFA$=""THEN1860
1870 RETURN:REM SPIELBEGINN
```

```
1880 REM *** SPRITE 0 (AUTO) ***
1890 DATA0,248,0,0,248,0,0,248,0,5,141
1900 DATA0,7,239,0,5,221,0,1,220,0,1
1910 DATA252,0,1,140,0,1,36,0,1,116,0,1
1920 DATA252,0,1,252,0,3,86,0,54,219,96
1930 DATA55,223,96,63,87,224,54,251,96
1940 DATA50,2,96,7,255,0,7,255,0
1950 REM *** SPRITE 1 (EXPLOSION) ***
1960 DATA0,32,0,2,73,52,6,157,193,104
1970 DATA188,29,131,54,124,44,118,249
1980 DATA65,231,242,31,227,178,78,8,57
1990 DATA39,63,156,47,159,30,73,222,124
2000 DATA36,194,240,73,248,198,19,253
2010 DATA216,71,205,144,23,15,36,16,103
2020 DATA72,13,146,80,0,8,144,3,87,32
```

Mine Field

(Minenfeld)

Mine Field ist ein Spiel, das Logik und Strategie verlangt. Sie sind im Besitz eines Minen-Suchgerätes und haben den Auftrag, in drei Minuten so viele Minen wie möglich zu zerstören. Unglücklicherweise geht Ihr Assistent Norbert Nachmach genau seitenverkehrt zu Ihnen vor. Passen Sie also auf, daß er nicht auf eine Mine tritt, oder Sie werden zusammen mit ihm den nächsten Tag nicht erleben!

Um eine Mine unschädlich zu machen, bringen Sie das Suchgerät über die Mine und drücken dann den Feuerknopf. Jede erfolgreich beseitigte Mine erbringt 100 Punkte.

BEHUTSAM AUFTRETEN!

```
1000 REM *** MINE FIELD ***
1010 PRINTCHR$(147):GOSUB2570
1020 GOSUB1850:REM ZEICHENSATZ KOPIEREN
MIT MS-PGR
1030 GOSUB2150:REM MS-PGR FUER JOYSTCK-A
BFRAGE
1040 REM SPRITE 0 UND 1
1050 REM IM KASSETTENPUFFER SPEICHERN
1060 POKE2040,13:POKE2041,14
1070 POKE53281,0
1080 REM *** VERTEILUNG DER MINEN ***
1090 REM *** SPRITE-DATEN LESEN ***
1100 FOR I=832 TO 894:READA
1110 POKEI,A:NEXT
1120 FOR I=896TO958:READA
1130 POKEI,A:NEXT
1140 V=53248
1150 POKEV+39,2:REM SPR 0 ROT FAERBEN
1160 POKEV+40,7:REM SPR 1 WEISS FAERBEN
1170 REM *** MINEN-DATEN LESEN ***
1180 FORI=12504TO12519:READA
1190 POKEI,A:NEXT
1200 GETA$:IFA$=""THEN1200
1210 GETJ$:IFJ$<>""THEN1210
1220 PRINTCHR$(147)
1230 REM *** ZUFALLSVERTEILUNG DER MINEN
 ***
1240 FORI=1TO40:SC=INT(RND(1)*840)
1250 IFSC=38ORSC=39THENI=I-1:GOTO1290
1260 IFSC/2<>INT(SC/2)THENI=I-1:GOTO1290
1270 POKE1104+SC,27:POKE1104+1+SC,28
1280 POKE55376+SC,5:POKE55376+1+SC,5
1290 NEXT
1300 REM *** JOYSTICK-PARAMETER SETZEN *
**
1310 POKE49152,24:POKE49153,0:REM X MIN
1320 POKE49156,70:POKE49157,1:REM X MAX
1330 POKE49160,60:POKE49161,0:REM Y MIN
1340 POKE49164,225:POKE49165,0:REM Y MAX
1350 POKE49168,1:REM ABFRAGEHAEUFIGKEIT
1360 SYS49183:REM JOYSTICK-ABFRAGE AKTIV
IEREN
1370 TI$="000000":REM ZEITNEHMER
1380 PRINTTAB(6)CHR$(145)CHR$(158)"PUNKT
ZAHL: 000"
```

```
1390 POKE49169,24:POKE49170,0:REM X-KOOR
DINATE SETZEN
1400 POKE49173,229:POKE49174,0:REM Y-KOO
RDINATE SETZEN
1410 POKEV,24:POKEV+1,229:POKEV+2,66:POK
EV+16,2
1420 POKEV+3,50
1430 POKEV+21,3:REM SPRITES 0 UND 1 ANSC
HALTEN
1440 REM *** BEWEGUNGSHAUPTROUTINE ***
1450 POKEV+31,0:REM KOLLISIONSREGISTER L
OESCHEN
1460 PRINTCHR$(145);TAB(22)"ZEIT: ";MID$
(TI$,4,1);":";RIGHT$(TI$,2)
1470 IFVAL(TI$)>=300THENGOSUB3070:REM EN
DE
1480 X=PEEK(49169)+256*PEEK(49170)
1490 Y=PEEK(49173)+256*PEEK(49174)
1500 N=345-X:M=279-Y
1510 HN=INT(N/256):LN=N-256*HN
1520 HX=INT(X/256):LX=X-256*HX
1530 IFX<90THEN POKEV+16,2:GOTO1550
1540 POKEV+16,HX
1550 POKEV,LX:POKEV+1,Y
1560 POKEV+2,LN:POKEV+3,M
1570 HC=PEEK(V+31):REM KOLLISION ERKANNT
1580 REM *** MANN/MINEN-KOLLISION? ***
1590 IFHC=2ORHC=3THENGOTO1680
1600 REM *** FEUERKNOPF GEDRUECKT? ***
1610 POKE49177,0
1620 FB2=PEEK(49177)
1630 IFFB2<>1THEN1450
1640 REM *** DETEKTOR/MINEN-KOLLISION? *
**
1650 IFHCAND1=1THENGOSUB2870
1660 GOTO1450
1670 END
1680 REM *** MINE/MANN KOLLISION ***
1690 PRINTCHR$(147)
1700 REM *** BILDSCHIRM BLINKT ***
1710 FORI=1TO5
1720 GOSUB2760:REM LETZTE EXPLOSION
1730 NEXT
1740 REM *** SPRITES AUSSCHALTEN ***
1750 POKEV+21,0
```

```
1760 FORI=1TO9:PRINTCHR$(17);:NEXT
1770 PRINTTAB(10)"DEIN ASSISTENT IST TOT
!"
1780 FORI=1TO3:PRINTCHR$(17);:NEXT
1790 PRINTTAB(14)"PUNKTZAHL:";YS
1800 FORI=1TO3:PRINTCHR$(17);:NEXT
1810 PRINTTAB(11)"<<<TASTE DRUECKEN>>>"
1820 SYS49200:REM JOYSTICK-ABFRAGE DESAK
TIVIEREN
1830 RUN1020
1840 END
1850 REM *** DATEN FUER MS-PGR ZUM ZEICH
ENSATZ KOPIEREN ***
1860 POKE52,48:POKE56,48:REM SPEICHEROBE
RGREZE HERABSETZEN
1870 FORI=828TO950:REM PGR IM
1880 READJ:REM------------
1890 AA=AA+J:REM KASSETTEN-
1900 POKEI,J:REM PUFFER ABLEGEN
1910 NEXT:REM------------
1920 READJ:REM PRUEFSUMME
1930 IFAA<>JTHENPRINT"FEHLER IN DEN DATE
N":END
1940 SYS(828):REM AUFRUFEN
1950 POKE53272,28:REM AUF RAM ZEIGEN
1960 DATA169,0,141,14,220,169,51,141
1970 DATA1,0,162,0,189,0,208,157,0,48
1980 DATA189,0,209,157,0,49,189,0,210
1990 DATA157,0,50,189,0,211,157,0,51
2000 DATA189,0,212,157,0,52,189,0,213
2010 DATA157,0,53,189,0,214,157,0,54
2020 DATA189,0,215,157,0,55,232,224
2030 DATA255,208,203,173,255,208,141
2040 DATA255,48,173,255,209,141,255
2050 DATA49,173,255,210,141,255,50
2060 DATA173,255,211,141,255,51,173
2070 DATA255,212,141,255,52,173,255
2080 DATA213,141,255,58,173,255,214
2090 DATA141,255,54,169,55,141,1,0
2100 DATA169,1,141,14,220,169,28,141
2110 DATA24,208,96
2120 REM PRUEFSUMME
2130 DATA16246
2140 RETURN
2150 REM *** DATEN FUER MS-PGR ZUR JOYST
ICK-ABFRAGE ***
```

```
2160 DATA120,173,20,3,141,29,192,173,21
2170 DATA3,141,30,192,169,71,141,20,3
2180 DATA169,192,141,21,3,88,96,120,173
2190 DATA29,192,141,20,3,173,30,192,141
2200 DATA21,3,88,96,173,27,192,205,16
2210 DATA192,240,3,76,65,193,169,255
2220 DATA141,27,192,162,0,160,0,185,2
2230 DATA220,141,28,192,169,224,153,2
2240 DATA220,185,0,220,72,173,28,192
2250 DATA153,2,220,189,8,192,221,21,192
2260 DATA189,9,192,253,22,192,144,18
2270 DATA189,8,192,157,21,192,189,9,192
2280 DATA157,22,192,104,74,72,76,162
2290 DATA192,104,74,72,176,13,222,21
2300 DATA192,189,21,192,201,255,208,3
2310 DATA222,22,192,189,21,192,221,12
2320 DATA192,189,22,192,253,13,192,144
2330 DATA18,189,12,192,157,21,192,189
2340 DATA13,192,157,22,192,104,74,72,76
2350 DATA207,192,104,74,72,176,8,254,21
2360 DATA192,208,3,254,22,192,189,0,192
2370 DATA221,17,192,189,1,192,253,18
2380 DATA192,144,18,189,0,192,157,17
2390 DATA192,189,1,192,157,18,192,104
2400 DATA74,72,76,1,193,104,74,72,176
2410 DATA13,222,17,192,189,17,192,201
2420 DATA255,208,3,222,18,192,189,17
2430 DATA192,221,4,192,189,18,192,253,5
2440 DATA192,144,18,189,4,192,157,17
2450 DATA192,189,5,192,157,18,192,104
2460 DATA74,72,76,46,193,104,74,72,176
2470 DATA8,254,17,192,208,3,254,18,192
2480 DATA104,74,176,5,169,1,153,25,192
2490 DATA200,232,232,224,4,240,3,76,91
2500 DATA192,238,27,192,108,29,192
2510 DATA35994:REM*PRUEFSUMME*
2520 FORI=49183TO49478
2530 READX:POKEI,X:CC=CC+X
2540 NEXT
2550 READX:IFX<>CCTHENPRINT"FEHLER IN DE
N DATEN"
2560 RETURN
2570 REM *** TITELBILD ***
2580 POKE53281,0:PRINTCHR$(30)
2590 PRINT:PRINT:PRINT
2600 PRINTTAB(18)"I"
```

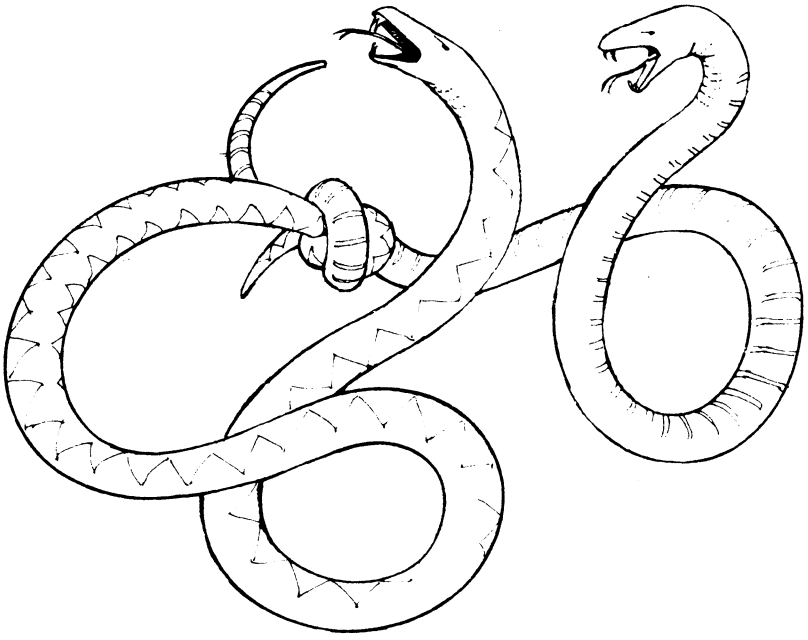
```
2610 PRINTTAB(6)"M         F           D"
2620 PRINTTAB(7)"I                E"
2630 PRINTTAB(11)"E"
2640 PRINTTAB(9)"N"
2650 PRINTTAB(22)"L"
2660 PRINT:PRINT:PRINT
2670 PRINTTAB(30)"C"
2680 PRINTTAB(24)"U   E      E"
2690 PRINTTAB(9)"A    T        D
    N"
2700 PRINTTAB(12)"S  E        R           K
"
2710 PRINTTAB(7)"T"
2720 PRINT
2730 PRINT
2740 PRINT
2750 RETURN
2760 REM *** LETZTE EXPLOSION ***
2770 IFI/2=INT(I/2)THENPOKE53281,2:GOTO2
790
2780 POKE53281,7
2790 S=54296:W=54276:A=54277
2800 H=54273:L=54272
2810 FORB=20TO0STEP-1:POKES,B:POKEW,129
2820 POKEA,15:POKEH,7:POKEL,12:NEXT
2830 POKEW,0:POKEA,0
2840 POKE53281,8:FORJ=1TO20:NEXT
2850 POKE53281,0
2860 RETURN
2870 REM *** MINE/DETEKTOR-KOLLISION ***
2880 CODE=1024+INT((Y-50)/8+1)*40+INT((X
-18)/8):REM X,Y IN BS-CODE UMWANDELN
2890 IFPEEK(CODE)=32THEN3060
2900 IFPEEK(CODE+1)=32THENCODE=CODE-1
2910 REM *** GERAEUSCHEFFEKT ***
2920 A=54277:S=54296:W=54276:H=54273:L=5
4272
2930 FORB=15TO0STEP-1
2940 POKE54272+CODE,B
2950 POKE54272+CODE+1,B:POKE53280,B+1
2960 POKES,B:POKEW,129
2970 POKEA,15:POKEH,7:POKEL,12:NEXT
2980 POKEW,0:POKEA,0
2990 REM *** MINE LOESCHEN ***
3000 POKECODE,32:POKECODE+1,32
```

```
3010 REM *** RAHMENFARBE SETZEN ***
3020 POKE53280,14
3030 REM *** PUNKTZAHL ***
3040 YS=YS+100
3050 PRINTTAB(6)CHR$(145)"YOUR SCORE";YS
3060 RETURN
3070 REM *** ZEIT ABGELAUFEN ***
3080 PRINTCHR$(147)
3090 PRINT:PRINT:PRINT:PRINT:PRINT
3100 PRINTTAB(12)CHR$(30)"■     ■ ■ ■ ■
▼ ■ ■   ■"
3110 PRINTTAB(12)"◤ ■ ■ ◥ ■ ■ ■    ■ ■ ■
"
3120 PRINTTAB(12)"  ■ ■   ■ ■ ■ ■▼■ ■ ■
  ■"
3130 PRINTTAB(12)"  ■ ■   ■ ■ ■ ■  ■ ■ ■
 ■"
3140 PRINTTAB(12)"  ■ ■   ■ ■ ■ ■  ■ ■ ■
   ■"
3150 PRINT:PRINT:PRINT
3160 PRINTTAB(13)"■◤  ◥■ ■ ■  ■ ■ ■
■"
3170 PRINTTAB(13)"■ ■  ■ ■ ■ ■  ■ ■ ◤ ■
■ ◥"
3180 PRINTTAB(13)"■ ■  ■ ■ ■ ■  ■ ■   ■
■"
3190 PRINTTAB(13)"■ ■  ■ ■ ■ ■  ■ ■   ■
■"
3200 PRINTTAB(13)"◥■  ■◤ ◥■  ■◤   ■ ■"
3210 PRINT:PRINT:PRINT
3220 PRINTTAB(13)"PUNKTZAHL:";YS
3230 PRINT:PRINT
3240 PRINTTAB(11)"<<TASTE DRUECKEN>>"
3250 POKEV+21,0
3260 REM *** GERAEUSCHEFFEKT ***
3270 S=54272
3280 FORL=0TO24:POKES+L,0
3290 POKES+1,64:POKES+5,9
3300 POKES+15,30:POKES+24,15
3310 FORI=1TO12:POKES+4,21
3320 FORJ=1TO200:NEXT:POKES+4,20
3330 FORJ=1TO200:NEXT:NEXT
3340 POKES+24,0
3350 RUN1020
3360 REM:
```

```
3370 REM *** DATEN FUER SPRITE 0 ***
3380 DATA252,63,0,248,31,0,240,15,0
3390 DATA232,23,0,199,227,0,132,33,0
3400 DATA4,32,0,4,32,0,4,32,0
3410 DATA132,33,0,199,227,0,232,23,0
3420 DATA240,15,0,248,31,0,252,63,0
3430 DATA0,0,0,0,0,0,0,0,0
3440 DATA0,0,0,0,0,0,0,0,0
3450 REM:
3460 REM ***DATEN FUER SPRITE 1 ***
3470 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
3480 DATA0,0,224,0,1,80,0,1,240,0,0,224
3490 DATA0,0,64,0,3,248,0,7,188,0,13
3500 DATA246,0,25,179,0,25,243,0,1,240
3510 DATA0,1,176,0,3,24,0,3,24,0,7,28,0
3520 DATA15,30
3530 REM:
3540 REM *** DATEN FUER MINEN ***
3550 DATA0,0,3,63,122,63,15,0
3560 DATA0,0,192,252,94,252,240,0
```

Look Out!

(Paß auf!)



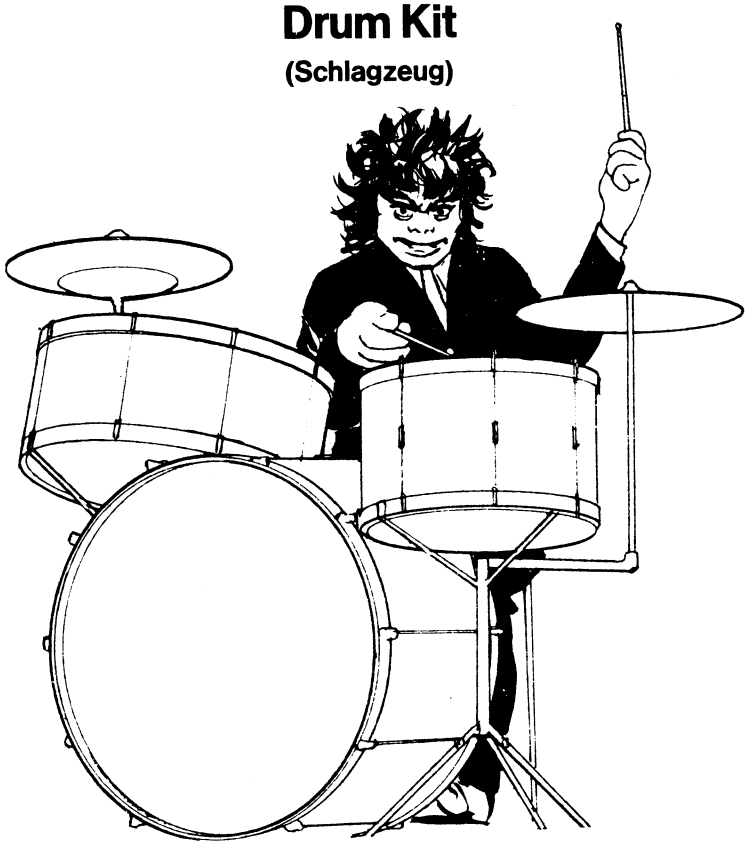
Dies ist ein Spiel für zwei Personen mit je einem Joystick. Als erstes müssen Sie die „target-score", die Ziel-Punktanzahl, bestimmen – wer diese zuerst erreicht hat, ist Sieger. Dann kann jeder Spieler mit seinem Joystick auf dem Bildschirm herumsteuern. Kreuzen Sie nicht die Bahn des anderen Spielers und berühren Sie sich auch nicht selbst, es sei denn, Sie wollen verlieren!

```
1000 REM *** LOOK OUT! ***
1010 S=54272
1020 FORI=0TO24:POKES+I,0:NEXT
1030 POKES+24,14
1040 POKES+5,31:POKES+6,15
1050 POKES,200
1060 PRINTCHR$(147)
1070 PRINT"ZIELPUNKTZAHL EINGEBEN: ";
1080 INPUTQ$:Q=VAL(Q$)
1090 IFQ<1THEN1060
1100 PRINTCHR$(147)
1110 POKE53281,0:POKE53280,0
1120 SC=1024:CO=55296:Z1=0:Z2=0
1130 X1=19:Y1=12:X2=21:Y2=12
1140 PRINTCHR$(19)CHR$(158)"SPIELER 1:"S
1
1150 PRINTCHR$(19)TAB(24)"SPIELER 2:"S2
1160 REM *** HAUPTSCHLEIFE ***
1170 P1=PEEK(56320):P2=PEEK(56321)
1180 XO=X1:YO=Y1:OX=X2:OY=Y2
1190 IFP1=126THENZ1=1
1200 IFP2=254THENZ2=1
1210 IFP1=125THENZ1=2
1220 IFP2=253THENZ2=2
1230 IFP1=123THENZ1=3
1240 IFP2=251THENZ2=3
1250 IFP1=119THENZ1=4
1260 IFP2=247THENZ2=4
1270 IFRND(1)>.5THEN1330
1280 ONZ1GOSUB1450,1470,1490,1510
1290 ONZ2GOSUB1530,1550,1570,1590
1300 IFPEEK(SC+X1+40*Y1)=160THEN1620
1310 IFPEEK(SC+X2+40*Y2)=160THEN1670
1320 GOTO1370
1330 ONZ2GOSUB1530,1550,1570,1590
1340 ONZ1GOSUB1450,1470,1490,1510
1350 IFPEEK(SC+X1+40*Y1)=160THEN1620
1360 IFPEEK(SC+X2+40*Y2)=160THEN1670
1370 POKESC+XO+40*YO,160
1380 POKESC+OX+40*OY,160
1390 POKECO+X1+40*Y1,14
1400 POKESC+X1+40*Y1,177
1410 POKECO+X2+40*Y2,5
1420 POKESC+X2+40*Y2,178
1430 GOTO1170
```

```
1440 REM *** BEWEGUNG ***
1450 Y1=Y1-1:IFY1<1THENY1=24
1460 RETURN
1470 Y1=Y1+1:IFY1>24THENY1=1
1480 RETURN
1490 X1=X1-1:IFX1<0THENX1=39
1500 RETURN
1510 X1=X1+1:IFX1>39THENX1=0
1520 RETURN
1530 Y2=Y2-1:IFY2<1THENY2=24
1540 RETURN
1550 Y2=Y2+1:IFY2>24THENY2=1
1560 RETURN
1570 X2=X2-1:IFX2<0THENX2=39
1580 RETURN
1590 X2=X2+1:IFX2>39THENX2=0
1600 RETURN
1610 REM *** PUNKT FUER SPIELER 2 ***
1620 S2=S2+1:POKES+4,17
1630 POKES+1,30:FORT=1TO120:NEXT
1640 IFS2=QTHEN1790
1650 POKES+4,0:GOTO1100
1660 REM *** PUNKT FUER SPIELER 1 ***
1670 S1=S1+1:POKES+4,17
1680 POKES+1,30:FORT=1TO120:NEXT
1690 IFS1=QTHEN1720
1700 POKES+4,0:GOTO1100
1710 REM *** SPIELER 1 GEWINNT ***
1720 PRINTCHR$(147):POKES+4,0
1730 PRINT"SPIELER 1 HAT GEWONNEN"
1740 PRINT:PRINT:PRINT"NOCH EINMAL?"
1750 GETA$:IFA$<>"J"ANDA$<>"N"THEN1750
1760 IFA$="J"THENRUN
1770 POKE53280,14:POKE53281,6:END
1780 REM *** SPIELER 2 GEWINNT ***
1790 PRINTCHR$(147):POKES+4,0
1800 PRINT"SPIELER 2 HAT GEWONNEN"
1810 PRINT:PRINT:PRINT"NOCH EINMAL?"
1820 GETA$:IFA$<>"J"ANDA$<>"N"THEN1750
1830 IFA$="J"THENRUN
1840 POKE53280,14:POKE53281,6:END
```

Drum Kit

(Schlagzeug)



Der Commodore 64 hat außerordentliche Tonerzeugungsmöglichkeiten, aber wie bei den meisten hochentwickelten Geräten ist es schwer, diese zu nutzen. Dieses synthetische Schlagzeug vermittelt zumindest eine gewisse Ahnung, wie man schlagzeugähnliche Klänge erzeugt.

Zur Verfügung stehen eine BASS- und eine SNARE-DRUM (große und kleine Trommel), ein HI- und ein LO-TOM-TOM (Hänge- und Stand-TOM-TOM) und ein Cymbal (Becken), wobei jeder Klangeffekt getrennt als Modul programmiert wurde und somit auch getrennt weiterverwendet werden kann.

Trotzdem ist Drum Kit ein abgeschlossenes Programm. Während es läuft, können Sie die einzelnen Drums von der Tastatur aus ansteuern; dabei schlagen Sie mit „F" und „G" auf das HI- bzw. auf das LO-TOM-TOM, mit „B" auf die BASS- und mit „J" auf die SNARE-DRUM. Wenn Sie auf die Leertaste drücken, ertönt das Cymbal. Diese Tastaturbelegung wurde gewählt, damit jede Drum leicht erreicht werden kann.

Bleibt eine Taste gedrückt, dann klingt die zugehörige Drum weiter; mit etwas Übung kann man auch kompliziertere Rhythmen spielen.

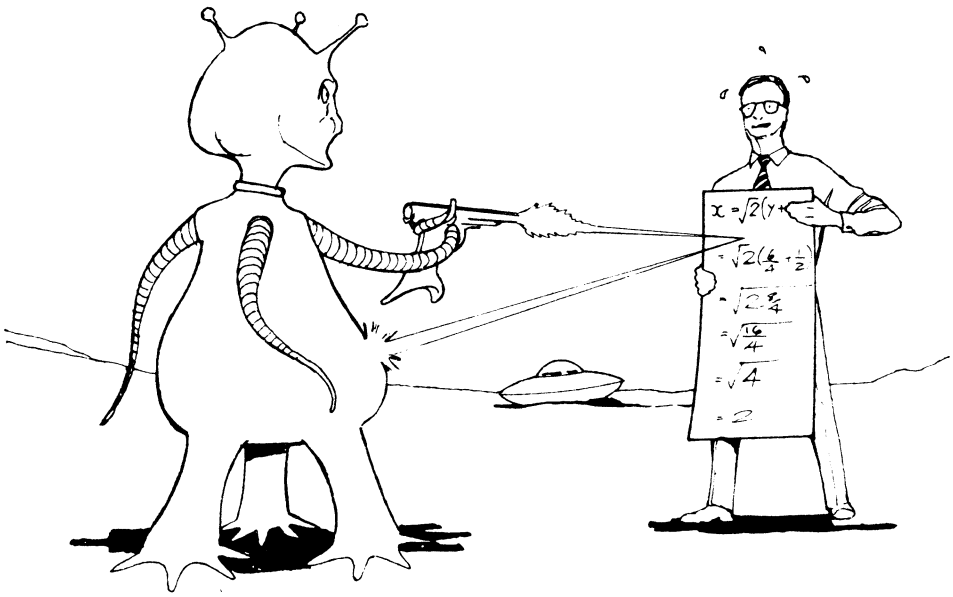
Das Programm hat noch viel Freiraum für Erweiterungen. Versuchen Sie doch mal, noch ein Crash Cymbal (Crash-Becken) oder ein Hi-Hat einzubauen. Fröhliches Trommeln!

```
1000 REM *** DRUM KIT ***
1010 PRINTCHR$(147)
1020 POKE53280,0:POKE53281,0
1030 PRINTTAB(4)"▬▬  ▬▬ ▬ ▬ ▬ ▬  ▬
▬ ▬▬ ▬▬▬▬"
1040 PRINTTAB(4)"▬ ▬  ▬ ▬ ▬ ▬ ▬▬▬  ▬ ▬
  ▬   ▬"
1050 PRINTTAB(4)"▬  ▬ ▬▬  ▬ ▬ ▬  ▬  ▬▬
  ▬   ▬"
1060 PRINTTAB(4)"▬ ▬  ▬ ▬ ▬ ▬ ▬  ▬  ▬ ▬
  ▬   ▬"
1070 PRINTTAB(4)"▬▬  ▬ ▬ ▬▬  ▬  ▬  ▬
▬ ▬▬  ▬"
1080 PRINT:PRINT:PRINT
1090 PRINTTAB(7)"(B) = BASS DRUM
1100 PRINTTAB(7)"(J) = SNARE DRUM
1110 PRINTTAB(7)"(F) = HI-TOM-TOM
1120 PRINTTAB(7)"(G) = LO-TOM-TOM
1130 PRINTTAB(7)"( ) = CRASH CYMBAL
1140 PRINTTAB(7)"(Z) = ZUFALLS-SCHLAGZEU
GSOLO
1150 PRINTTAB(7)"(T) = TASTATUR WIEDER E
INSCHALTEN
1160 REM *** TON-REGISTER SETZEN ***
1170 S=54272:S1=54286:S2=54279
1180 FORI=0TO24:POKEI+S,0:NEXT
1190 POKES+24,14:REM LAUTSTAERKE
1200 REM *** TASTATURABFRAGE ***
1210 GETA$
1220 IFA$="Z"THEN1720:REM SOLO
1230 IFA$="J"THENGOSUB1300:REM SNARE
1240 IFA$="B"THENGOSUB1380:REM BASS
1250 IFA$="F"THENGOSUB1470:REM TOM 1
1260 IFA$="G"THENGOSUB1560:REM TOM 2
1270 IFA$=" "THENGOSUB1650:REM CYMBAL
1280 GOTO1210
1290 REM *** SNARE DRUM ***
1300 POKES2+4,0
1310 POKES2+5,8
1320 POKES2+6,0
1330 POKES2+4,129
1340 POKES2,200
1350 POKES2+1,90
1360 RETURN
1370 REM *** BASS DRUM ***
```

```
1380 POKES1+4,0
1390 POKES2+4,0
1400 POKES1+5,4
1410 POKES1+6,2
1420 POKES1+4,17
1430 POKES1,2
1440 POKES1+1,10
1450 RETURN
1460 REM *** TOM 1 ***
1470 POKES1+4,0
1480 POKES2+4,0
1490 POKES1+5,4
1500 POKES1+6,2
1510 POKES1+4,17
1520 POKES1,2
1530 POKES1+1,20
1540 RETURN
1550 REM *** TOM 2 ***
1560 POKES1+4,0
1570 POKES2+4,0
1580 POKES1+5,4
1590 POKES1+6,2
1600 POKES1+4,17
1610 POKES1,2
1620 POKES1+1,15
1630 RETURN
1640 REM *** CRASH CYMBAL ***
1650 POKES+4,0
1660 POKES+5,9
1670 POKES+6,0
1680 POKES+4,129
1690 POKES,2
1700 POKES+1,150
1710 RETURN
1720 REM *** ZUFAELLIGES SCHLAGZEUGSOLO
***
1730 R=INT(RND(TI)*5+1)
1740 D=INT(RND(1)*100)
1750 FORT=1TO50+D:NEXT
1760 ONRGOSUB1300,1380,1470,1560,1650
1770 GETB$:IFB$<>"T"THEN1730
1780 GOTO1210
```

Math Encounters

(Mathematische Begegnung der dritten Art)



In diesem Spiel müssen Sie Ihre mathematischen Fähigkeiten einsetzen, um einen Humanoiden zu retten. Berechnen Sie die Aufgabe richtig, dann schützt ihn die „Macht des Wissens". Antworten Sie falsch, dann steht seiner Liquidation durch das Raumwesen nichts mehr im Wege.

Das „+"-Zeichen kann in den Zeilen 1720 und 1740 zu „—", „*", oder „/" verändert werden. Auch die Größenordnung der Zahlen ist in den Zeilen 1690 und 1700 manipulierbar.

Da in diesem Programm Kleinbuchstaben verwendet werden, wurde es auch im Groß-/Kleinschreibungsmodus ausgedruckt. Drücken Sie also, bevor Sie das Programm eingeben, die SHIFT- und die C=-Taste gleichzeitig, um in diesen Modus umzuschalten.

```
1000 rem *** math  encounters ***
1010 printchr$(147)chr$(14)
1020 poke53280,0:poke53281,0
1030 printtab(5)"Bei diesem Spiel muss m
an die"
1040 printtab(5)"richtige  Loesung  eine
r  Re-"
1050 printtab(5)"chenaufgabe  eingeben u
nd die"
1060 printtab(5)"RETURN-Taste druecken."
1070 print
1080 printtab(5)"Daraufhin schiesst  das
 Raum-"
1090 printtab(5)"wesen mit seiner Laserk
anone."
1100 printtab(5)"Bei  falscher Loesung wi
rd der"
1110 printtab(5)"arme Humanoide getroffe
n."
1120 printtab(5)"Wenn aber  die  richtig
e Loe-"
1130 printtab(5)"sung  eingegeben  wurde
, kann"
1140 printtab(5)"der  Humanoide mit  ein
em Ab-"
1150 printtab(5)"wehrschild den Strahl a
uf das"
1160 printtab(5)"Raumwesen  lenken  und
  sich"
1170 printtab(5)"retten."
1180 print
1190 printtab(5)"Was geschieht  als naec
hstes?"
1200 printtab(5)"Es wird eine neue Aufga
be ge-"
1210 printtab(5)"stellt. LOS GEHT'S!!!"
1220 print
1230 printtab(12)"TASTE DRUECKEN"
1240 getc$:ifc$=""then1240
1250 printchr$(147):poke53280,4
1260 rem *** sprites und ton initialisie
ren ***
1270 fori=0to62:readj:pokei+832,j:next
1280 fori=0to62:readj:pokei+896,j:next
1290 poke981,3:rem 'geschoss'-sprite
```

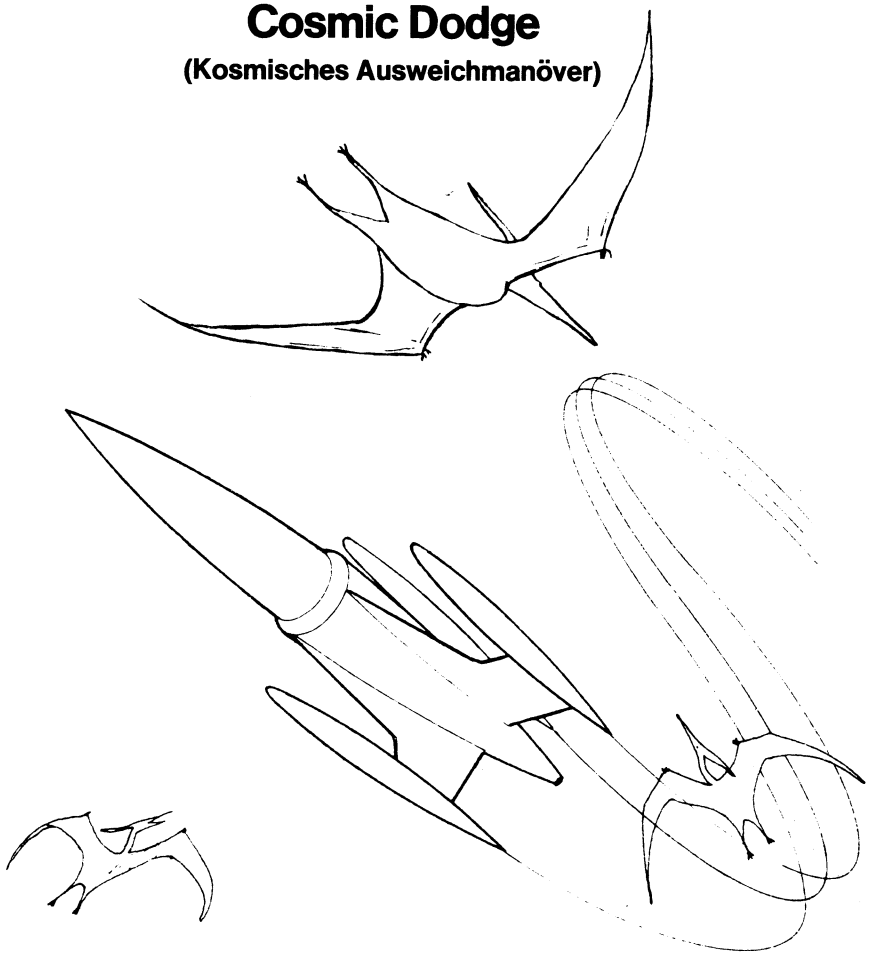
```
1300 vc=53248:s=54272:s1=54286
1310 poke2040,13:poke2041,14:poke2042,15
1320 pokevc+21,0:poke930,170
1330 pokevc+28,3:pokevc+37,10
1340 pokevc+38,9:pokevc+39,5
1350 pokevc+40,5:pokevc+41,1
1360 pokevc,50:pokevc+1,100
1370 pokevc+2,255:pokevc+3,100
1380 pokevc+4,245:pokevc+5,100
1390 pokevc+23,7:pokevc+29,7
1400 fori=0to24:pokei+s,0:next
1410 pokes+24,14
1420 pokes+5,31:pokes+6,15
1430 pokes1+5,31:pokes1+6,15
1440 pokes,200:pokes1,200
1450 rem *** bildschirm aufbauen ***
1460 printchr$(19)chr$(158)
1470 fort=1to8:print:next
1480 fort=1to40:print"_";:next
1490 printchr$(18);
1500 printchr$(156);
1510 fort=1to80:print" ";:next
1520 printchr$(158)chr$(146);
1530 fort=1to40:print"¯";:next
1540 fort=1to20:r=int(rnd(1)*360)
1550 poke55296+r,7:poke1024+r,46:next
1560 print:printchr$(30);
1570 printtab(5)"Gib' die richtige Loesu
ng ein"
1580 print
1590 printtab(7)"und rette den Humanoide
n!"
1600 printchr$(158)
1610 fort=1to40:print"_";:next
1620 printchr$(19);
1630 printchr$(18)"         *** Math  ";
1640 print"Encounters ***         "
1650 rem *** hauptprogramm ***
1660 fort=1to2000:next:pokevc+21,3
1670 b=245:d=-2
1680 fori=1864to1903:pokei,32:next
1690 x=int(rnd(ti)*20):rem hier zahlen-
1700 y=int(rnd(ti)*20):rem bereich festl
egen
1710 poke781,21:poke782,13:sys65520
1720 printx"+"y"=";
```

```
1730 inputq$:q=val(q$):m=0
1740 ifq<>(x+y)thenm=1
1750 ifm=1thengosub1790
1760 ifm=0thengosub1920
1770 goto1660
1780 rem *** falsch ***
1790 pokevc+21,7
1800 gosub2040
1810 b=b-2:ifb>50thenpokevc+4,b:goto1810
1820 pokes+4,33:pokes1+4,33
1830 fort=1to100:pokevc+21,3
1840 pokes+1,110-t:pokes1+1,101-t
1850 pokevc+21,2:nextt
1860 pokes+4,0:pokes1+4,0
1870 poke781,21:poke782,7:sys65520
1880 print"Die Loesung lautet"x+y
1890 ford=1to8000:next
1900 return
1910 rem *** richtig ***
1920 pokevc+21,7
1930 gosub2040
1940 b=b+d:ifb<84thend=-d:poke930,138
1950 pokevc+4,b
1960 ifb<250then1940
1970 pokes+4,33:pokes1+4,33
1980 fort=1to100:pokevc+21,3
1990 pokes+1,10+t:pokes1+1,1+t
2000 pokevc+21,1:next:poke930,170
2010 pokes+4,0:pokes1+4,0
2020 return
2030 rem *** laser abfeuern ***
2040 pokes+4,33:pokes1+4,33
2050 forg=40to10step-5
2060 pokes+1,g:pokes1+1,g+28
2070 nextg
2080 pokes+4,0:pokes1+4,0
2090 return
2100 rem *** sprite-daten ***
2110 data0,0,0,0,60,0,0,255,0,0,215,15
2120 data0,215,60,0,20,48,0,191,192,2
2130 data191,128,10,170,160,40,186,40
2140 data96,170,9,16,186,4,0,255,0,0
2150 data255,0,0,255,0,0,195,0,0,195,0
2160 data0,195,0,0,195,0,0,130,0,3,195,1
92
```

```
2170 data0,0,0,0,0,0,8,0,32,2,0,128,2
2180 data40,128,0,170,0,0,170,0,242,238
2190 data128,58,170,160,2,170,160,0,170
2200 data32,0,138,0,0,170,0,0,170,0,0
2210 data168,0,0,136,0,0,136,0,0,136,0
2220 data2,138,0,0,0,0,0,0,0
ready.
```

Cosmic Dodge

(Kosmisches Ausweichmanöver)



Dieses Programm versetzt Sie an Bord eines Raumfrachters, der keinerlei Verteidigungssysteme besitzt. Vor Ihnen liegt ein Schwarm von vogelähnlichen Kreaturen, die Ihr Schiff zwar nicht angreifen, aber doch mit ihm zusammenstoßen können. Sie haben also folgende Möglichkeiten:

(1) Sie weichen ihnen aus.
(2) Sie fliegen im Hyperraum.

Wenn Sie im Hyperraum fliegen, gewinnen Sie keine Punkte; außerdem könnte es katastrophal sein, aus dem Hyperraum zurückzukehren und direkt eine dieser Kreaturen zu rammen. Merken Sie sich daher, wo Ihr Schiff war, als Sie es das letzte Mal gesehen haben!

Die F1-Taste bringt Sie in den Hyperraum, F7 bringt Sie wieder in das normale Raum-Zeit-Kontinuum zurück.

Wir wünschen einen angenehmen Flug!

```
1000 REM *** COSMIC DODGE ***
1010 PRINTCHR$(147):GOSUB1700:REM TITELB
ILD
1020 POKE53280,0:POKE53281,0
1030 REM *** JOYSTICK-DATEN LESEN ***
1040 FORI=49183TO49478:READJ
1050 POKEI,J:CC=CC+J:NEXTI:READJ
1060 IFJ<>CCTHENPRINT"FEHLER IN DEN DATE
N":END
1070 SYS49183
1080 REM *** SPRITE-DATEN LESEN ***
1090 FORI=0TO62:READJ:POKE832+I,J:NEXT
1100 FORI=0TO62:READJ:POKE896+I,J:NEXT
1110 REM *** JOYSTICK-PARAMETER SETZEN *
**
1120 POKE49152,50:POKE49153,0
1130 POKE49156,255:POKE49157,0
1140 POKE49160,180:POKE49161,0
1150 POKE49164,207:POKE49165,0
1160 POKE49168,0
1170 REM *** SPRITES INITIALISIEREN ***
1180 POKE2040,13:VC=53248
1190 POKEVC+39,8:POKEVC+31,0
1200 POKEVC+23,1:POKEVC+29,1
1210 X=170:Y=207:POKEVC+1,Y:POKEVC,X
1220 POKE49169,X:POKE49173,Y
1230 S=54272
1240 FORI=0TO24:POKES+I,0:NEXT
1250 POKES+5,31:POKES+6,240
1260 PRINTCHR$(17)CHR$(17)
1270 PRINTTAB(12)"TASTE DRUECKEN"
1280 GETJ$:IFJ$<>""THEN1280
1290 GETA$:IFA$=""THEN1290
1300 PRINTCHR$(147):POKEVC+21,1
1310 REM *** HAUPTSCHLEIFE ***
1320 X=PEEK(49169):Y=PEEK(49173)
1330 PRINTCHR$(19)CHR$(17);
1340 PRINTCHR$(157)CHR$(148)
1350 POKE218,160
1360 POKE646,INT(RND(1)*14+1)
1370 PRINTTAB(INT(RND(TI)*35))"↓"
1380 PRINTCHR$(19)CHR$(158)"PUNKTE:"SE
1390 IFSE>HITHENHI=SE
1400 PRINTCHR$(19)TAB(23)"MAX.PUNKTE:"HI
1410 GETA$:IFA$=CHR$(133)THENGOSUB1500
```

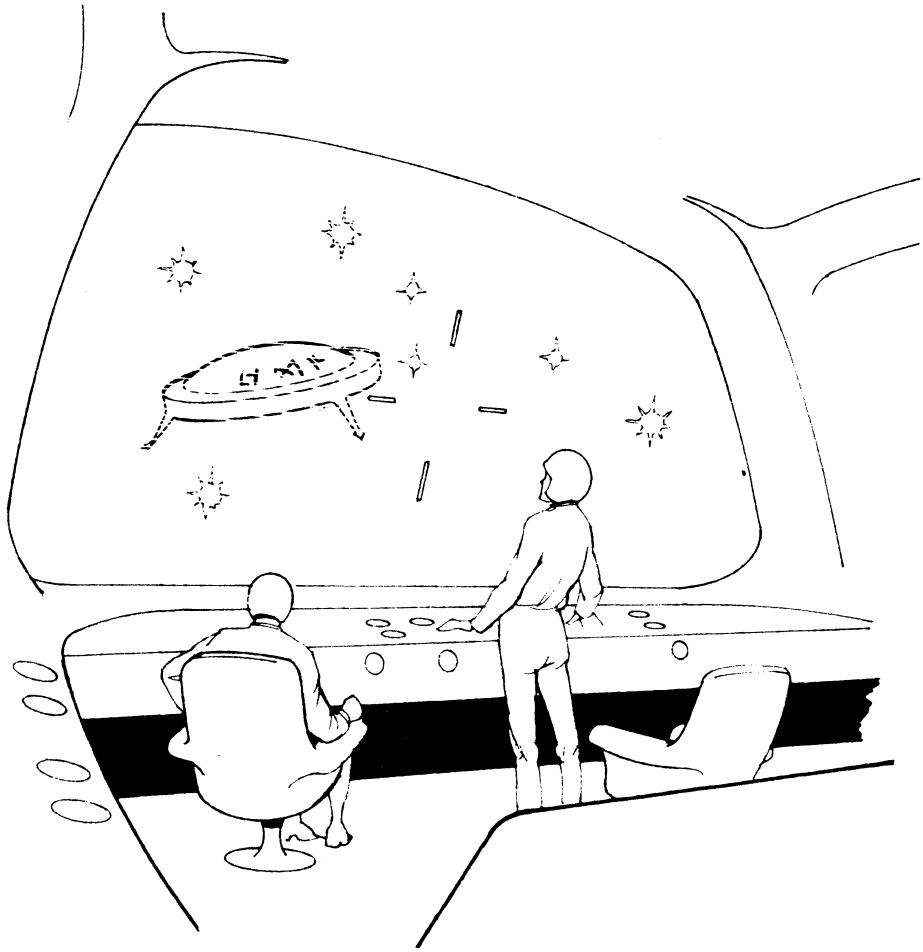
```
1420 IFA$=CHR$(136)THENGOSUB1550
1430 IFHY=1THEN1330
1440 POKEVC+31,0
1450 IFPEEK(VC+31)AND1=1THEN1590
1460 IFHY=0THENSE=SE+1
1470 POKEVC,X:POKEVC+1,Y
1480 GOTO1320
1490 REM *** HYPERRAUM ***
1500 PRINTCHR$(30)
1510 PRINTCHR$(19)TAB(11)"-HYPERRAUM-"
1520 POKEVC+21,0:HY=1
1530 RETURN
1540 REM *** RUECKKEHR AUS HYPERRAUM ***
1550 PRINTCHR$(19)TAB(11)"           "
1560 POKEVC+21,1:HY=0
1570 RETURN
1580 REM *** KOLLISION ***
1590 POKE2040,14
1600 FORV=15TO0STEP-.3:POKEVC+21,1
1610 POKES+4,129:POKES,100:POKES+1,3+V
1620 POKES+24,V:POKEVC+21,0:NEXT
1630 FORK=1TO1000:NEXT
1640 POKE2040,13
1650 PRINTCHR$(19):FORT=1TO4:PRINTCHR$(1
7)
1660 NEXT
1670 PRINTTAB(14)"-SPIELENDE-"CHR$(17)
1680 FORK=1TO1000:NEXT
1690 SE=0:GOSUB1700:GOTO1180
1700 REM *** TITELBILD ***
1710 PRINTCHR$(147)CHR$(158)
1720 PRINTCHR$(17)
1730 PRINTTAB(11)"┌─ ┌─┐ ┌─┐ ┌┐┌┐┬ ┌─"
1740 PRINTTAB(11)"│  │ │ ││  │└┘│└┐│ "
1750 PRINTTAB(11)"│  │ │ ├┤  │  ├┤ │ "
1760 PRINTTAB(11)"│  │ │ ├┤  │  ├┤ │ "
1770 PRINTTAB(11)"│  │ │ ├┤  │  ├┤ │ "
1780 PRINTTAB(11)"│  │ │ ├┤  │  ├┤ │ "
1790 PRINTTAB(11)"│  │ │ └┘  │  ├┤ │ "
1800 PRINTTAB(11)"│  │ │ │   ┌┤  ├┤ │ "
1810 PRINTTAB(11)"│  │ │ │   ┌┤  ├┤ │ "
1820 PRINTTAB(11)"│  │ │ │   ┌┤  ├┤ │ "
1830 PRINTTAB(11)"│  │ │ │   ┌┤  ├┤ │ "
1840 PRINTTAB(11)"│  │ │ │   ┌┤  ├┤ │ "
1850 PRINTTAB(11)"└─ └─┘ └─┘  └┘  └┴ └─"
```

```
1860 PRINTCHR$(17)
1870 PRINTTAB(11)"*** D O D G E ***
1880 RETURN
1890 DATA120,173,20,3,141,29,192,173,21
1900 DATA3,141,30,192,169,71,141,20,3
1910 DATA169,192,141,21,3,88,96,120,173
1920 DATA29,192,141,20,3,173,30,192,141
1930 DATA21,3,88,96,173,27,192,205,16
1940 DATA192,240,3,76,65,193,169,255
1950 DATA141,27,192,162,0,160,0,185,2
1960 DATA220,141,28,192,169,224,153,2
1970 DATA220,185,0,220,72,173,28,192
1980 DATA153,2,220,189,8,192,221,21,192
1990 DATA189,9,192,253,22,192,144,18
2000 DATA189,8,192,157,21,192,189,9,192
2010 DATA157,22,192,104,74,72,76,162
2020 DATA192,104,74,72,176,13,222,21
2030 DATA192,189,21,192,201,255,208,3
2040 DATA222,22,192,189,21,192,221,12
2050 DATA192,189,22,192,253,13,192,144
2060 DATA18,189,12,192,157,21,192,189
2070 DATA13,192,157,22,192,104,74,72,76
2080 DATA207,192,104,74,72,176,8,254,21
2090 DATA192,208,3,254,22,192,189,0,192
2100 DATA221,17,192,189,1,192,253,18
2110 DATA192,144,18,189,0,192,157,17
2120 DATA192,189,1,192,157,18,192,104
2130 DATA74,72,76,1,193,104,74,72,176
2140 DATA13,222,17,192,189,17,192,201
2150 DATA255,208,3,222,18,192,189,17
2160 DATA192,221,4,192,189,18,192,253,5
2170 DATA192,144,18,189,4,192,157,17
2180 DATA192,189,5,192,157,18,192,104
2190 DATA74,72,76,46,193,104,74,72,176
2200 DATA8,254,17,192,208,3,254,18,192
2210 DATA104,74,176,5,169,1,153,25,192
2220 DATA200,232,232,224,4,240,3,76,91
2230 DATA192,238,27,192,108,29,192
2240 DATA35994:REM*PRUEFSUMME*
2250 REM *** SPRITE 0 (SCHIFF) ***
2260 DATA0,32,0,0,32,0,0,32,0,0,112,0,0
2270 DATA80,0,0,112,0,0,112,0,0,112,0,0
2280 DATA248,0,0,248,0,0,248,0,0,248,0
2290 DATA0,248,0,2,250,0,2,170,0,6,171
2300 DATA0,6,171,0,6,171,0,6,171,0,7,39
```

```
2310 DATA0,7,39,0
2320 REM *** SPRITE 0 (EXPLOSION) ***
2330 DATA0,16,64,34,16,128,19,9,16,8
2340 DATA200,96,4,68,64,2,32,128,48,9,0
2350 DATA12,172,28,2,68,224,0,17,0,1
2360 DATA168,48,24,37,8,97,64,128,2,8
2370 DATA96,4,136,16,1,36,140,2,36,64,4
2380 DATA68,32,8,66,0,16,2,0,0,0,0
```

Galactic Battle

(Raumschlacht)



Dies ist nur ein kleines Spiel für Liebhaber der „Schieß sie runter"-Spiele. Galactic Battle hat als Schauplatz eine weit, weit entfernte Galaxie. Unter Ihrem Befehl steht ein Kundschafter-Schiff, das einst Teil eines ganzen Konvois war. Alle Schiffe außer Ihrem sind jedoch durch eine fremde Macht zerstört worden. Sie sind völlig allein. Ihr Hyper-Com-Funksprechgerät ist defekt, und Ihre Energievorräte neigen sich dem Ende zu. Dann, Sie dachten schon, Sie wären den Fängen der bösen Macht entkommen, werden Sie wieder angegriffen. Wie lange werden Sie mit Ihren begrenzten Energiereserven durchhalten?

Nachdem Sie dieses Spiel gestartet haben, befinden Sie sich auf der Brücke des Kundschafterschiffs und blicken in die Weite des Raums. Unter dem Sichtfenster befindet sich ein Armaturenbrett, das Ihnen einige wichtige Daten mitteilt. In der Mitte des Armaturenbretts sehen Sie eine laufende Uhr, die Ihnen anzeigt, wie lange Sie bis jetzt durchhalten konnten. In der Mitte des Sichtfensters befindet sich ein Fadenkreuz, das sich mit dem feindlichen Raumschiff decken muß, wenn Sie es abschießen wollen. Wenn Sie den Joystick bewegen, hat der Feuerknopf keine Wirkung, um Sie davon abzuhalten, ununterbrochen zu schießen. Sind Ihre Energievorräte erschöpft, dann haben Ihre Schutzschilde keine Wirkung mehr, und sobald die Beschädigungen einen Wert von 500 überschreiten, tritt die völlige Zerstörung ein. In diesem Moment wird Ihnen die Möglichkeit gegeben zu beweisen, daß Sie kein Feigling sind!

Die in diesem Programm benutzten Variablen sind:

SC = Basisadresse des Bildschirms
CO = Basisadresse des Farbspeichers
SE = Bildschirmspeicherstelle, an der Explosion erscheint
CE = Farbspeicherstelle, an der Explosion erscheint
X = Horizontale Bewegung des feindlichen Raumschiffs
Y = Vertikale Bewegung des feindlichen Raumschiffs
A() = Feld, das die Bewegungsrichtung der Trümmer enthält
EN = Aktueller Energievorrat
KI = Anzahl der zerstörten feindlichen Raumschiffe
DA = Beschädigungsgrad
PH = Verbleibende Photonengeschosse
S = Basisadresse des SID
VC = Basisadresse des VIC

Achtung: Speichern Sie dieses Programm unbedingt ab, bevor Sie es starten. Es enthält nämlich den Befehl SYS 64738 (Basic-Kaltstart), der das Programm löscht. Der Befehl wird ausgeführt, wenn kein weiteres Spiel mehr erwünscht ist. Um den Kaltstart zu umgehen, drücken Sie einfach RUN/STOP und RESTORE, wenn der Computer fragt, ob Sie nochmals spielen wollen.

```
1000 REM *** GALACTIC BATTLE ***
1010 REM "■"=CURSOR NACH LINKS
1020 :
1030 PRINTCHR$(147)CHR$(152)
1040 POKE53280,11:POKE53281,0
1050 :
1060 REM *** SPRITE-DATEN LESEN ***
1070 :
1080 FORI=0TO62:READJ:POKEI+832,J:NEXT
1090 FORI=0TO62:READJ:POKEI+896,J:NEXT
1100 :
1110 REM *** SPRITES INITIALISIEREN ***
1120 :
1130 VC=53248:POKEVC+21,3
1140 POKE2040,13:POKE2041,14
1150 POKEVC+23,0:POKEVC+29,3
1160 POKEVC+39,7:POKEVC+40,12
1170 POKEVC,161:POKEVC+1,107
1180 :
1190 REM *** VARIABLEN INITIALISIEREN **
*
1200 :
1210 SC=1024:CO=55296
1220 SE=SC+380:CE=CO+380
1230 X=INT(RND(1)*200+30)
1240 Y=INT(RND(1)*150+10)
1250 A(1)=1:A(2)=39:A(3)=40:A(4)=41
1260 A(5)=-1:A(6)=-39:A(7)=-40:A(8)=-41
1270 EN=999:KI=0:DA=0:PH=99
1280 TI$="000000"
1290 :
1300 REM *** TONREGISTER SETZEN ***
1310 :
1320 S=54272
1330 FORI=0TO24:POKES+I,0:NEXT
1340 POKES+24,14
1350 POKES+5,31:POKES+6,15
1360 POKES,150:POKES+1,1
1370 :
1380 REM *** ARMATURENBRETT ***
1390 :
1400 PRINTCHR$(151)
1410 POKE781,17:POKE782,0:SYS65520
1420 PRINTCHR$(18)"◤ ";
1430 FORT=1TO36:PRINT"▁";:NEXT
1440 PRINT" ◥";
```

```
1450 PRINT"  | ┌──────────────┐    ";
1460 PRINT"    ┌──────────────┐ |  ";
1470 PRINT"  | |ENERGIE       | ┌──";
1480 PRINT"┌──┐|TORPEDO      | |  ";
1490 PRINT"  | ├──────────────┤|   ";
1500 PRINT"  | ├──────────────┤ |  ";
1510 PRINT"  | |SCHADEN       | └──";
1520 PRINT"└──┘|TREFFER      | |  ";
1530 PRINT"  | └──────────────┘    ";
1540 PRINT"    └──────────────┘ |  ";
1550 PRINT" └";
1560 FORT=1TO36:PRINT"─";:NEXT
1570 PRINT"┘ ";CHR$(158);
1580 FORI=0TO39:POKE1984+I,160
1590 POKE56256+I,11:NEXT
1600 FORT=1TO25:Q=INT(RND(1)*680)
1610 POKECO+Q,7:POKESC+Q,46:NEXT
1620 :
1630 REM *** HAUPTSCHLEIFE ***
1640 :
1650 P=PEEK(56320):Z=0:R=0
1660 IFP=126THENZ=1
1670 IFP=125THENZ=2
1680 IFP=123THENZ=4
1690 IFP=119THENZ=3
1700 IFP=111THENPH=PH-1:GOSUB2380
1710 ONZGOSUB1950,1970,1990,2010
1720 POKEVC+30,0
1730 IFP=111AND(PEEK(VC+30)AND3)=3THENGO
SUB2150
1740 EN=EN-.1:IFEN<0THEN2430
1750 IFRND(1)<.025THENGOSUB2060
1760 IFRND(1)<.2THENR=INT(RND(TI)*4+1)
1770 ONRGOSUB1950,1970,1990,2010
1780 IFDA>500THEN2430
1790 POKE781,19:POKE782,11:SYS65520
1800 PRINT;INT(EN);"■■ "
1810 POKE781,21:POKE782,11:SYS65520
1820 PRINT;DA;
1830 POKE781,19:POKE782,32:SYS65520
1840 PRINT;PH;"■■ "
1850 POKE781,21:POKE782,32:SYS65520
1860 PRINT;KI
1870 POKE781,20:POKE782,18:SYS65520
1880 PRINTMID$(TI$,3,1)MID$(TI$,4,1);
1890 PRINT":";RIGHT$(TI$,2)
```

```
1900 POKEVC+2,X:POKEVC+3,Y
1910 GOTO1650
1920 :
1930 REM *** BEWEGUNG DES ANGREIFERS ***
1940 :
1950 Y=Y-8:IFY<10THENY=10
1960 RETURN
1970 Y=Y+8:IFY>165THENY=165
1980 RETURN
1990 X=X-8:IFX<10THENX=10
2000 RETURN
2010 X=X+8:IFX>246THENX=246
2020 RETURN
2030 :
2040 REM *** GEGENFEUER DES ANGREIFERS *
**
2050 :
2060 POKE53281,2:POKEVC+40,1
2070 POKES+4,129:DA=DA+INT(RND(1)*30)
2080 EN=EN-10:IFEN<0THEN2430
2090 FORD=1TO100:NEXT:POKEVC+40,12
2100 POKES+4,0:POKE53281,0
2110 RETURN
2120 :
2130 REM *** EXPLOSION DES ANGREIFERS **
*
2140 :
2150 POKEVC+21,1:POKES+4,129
2160 POKE53281,7:POKE53280,12
2170 FORD=1TO50:NEXTD:POKE53281,0
2180 POKE53280,11
2190 FORT=1TO7:FORD=1TO25:NEXTD
2200 G=INT(RND(TI)*8+1):W(T)=G
2210 POKECE+T*A(G),1:POKESE+T*A(G),127
2220 NEXTT:KI=KI+1
2230 FORT=1TO7STEP.5
2240 POKESE+T*A(W(T)),32
2250 NEXT:POKES+4,0
2260 :
2270 REM *** NAECHSTER ANGREIFER ***
2280 :
2290 FORJ=1TO2000:NEXT
2300 X=INT(RND(1)*200+30)
2310 Y=INT(RND(1)*150+10)
2320 POKEVC+2,X:POKEVC+3,Y
2330 POKEVC+21,3
```

```
2340 RETURN
2350 :
2360 REM *** ZAHL DER PHOTONENTORPEDOS P
RUEFEN ***
2370 :
2380 IFPH>0THENRETURN
2390 PH=99:EN=EN-200:RETURN
2400 :
2410 REM *** EXPLOSION DES EIGENEN SCHIF
FES ***
2420 :
2430 POKEVC+21,0:FORI=15TO0STEP-1
2440 FORC=1TO15
2450 POKE53281,C:POKE53280,C
2460 POKES+4,129:POKES+1,20+I
2470 POKES+24,I:NEXTC,I
2480 POKES+4,0:POKE53281,0:POKE53280,0
2490 PRINTCHR$(158)CHR$(147)
2500 PRINT"NOCH EIN KAMPF, ";
2510 PRINT"ODER SIND SIE ZU FEIGE?"
2520 PRINT:PRINT:PRINT
2530 PRINT"JA ODER NEIN?"
2540 GETA$:IFA$<>"J"ANDA$<>"N"THEN2540
2550 IFA$="J"THENRUN
2560 SYS64738:REM ACHTUNG! SPEICHER WIRD
 GELOESCHT
2570 :
2580 REM *** SPRITE-DATEN ***
2590 :
2600 DATA0,16,0,0,16,0,0,16,0,0,16,0,0
2610 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2620 DATA0,0,254,0,254,0,0,0,0,0,0,0,0
2630 DATA0,0,0,0,0,0,0,0,0,0,0,16,0,0
2640 DATA16,0,0,16,0,0,16,0
2650 :
2660 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2670 DATA0,0,0,0,120,0,7,207,128,60
2680 DATA120,240,96,0,24,71,255,136
2690 DATA124,120,248,88,48,104,76,48
2700 DATA200,71,255,136,64,0,8,224,0
2710 DATA28,160,0,20,160,0,20,224,0,28
2720 DATA0,0,0
2730 :
2740 REM *** PROGRAMMENDE ***
```

Das große Commodore 64 Spiele-Buch: Lehrbuch und Spielebuch zugleich!

Probleme zu erkennen, zu wissen, wie man sie löst, auf welche Informationen und Hilfsmittel man zurückgreifen kann, ist Ziel jedes Lernprogramms. Die Entwicklung dieser Fähigkeit ist Voraussetzung für Fortschritt, auch beim Programmieren.

Welchen anregenderen Weg gibt es, als mit Ihrem Commodore 64 spielend programmieren zu lernen?

In anschaulich-verständlicher Form vermitteln die Programmierer Terry B. Barrett und St. W. Colwill dem Anfänger fortgeschrittene Programmierkenntnisse: Sie gehen ein auf die grafischen und akustischen Möglichkeiten, die Verfeinerung von Programmiertechniken, führen Maschinencode-Routinen vor sowie den Joystick-Einsatz.

Im Spiele-Teil werden diese theoretischen Ausführungen in die Praxis umgesetzt. Spielanleitung, Programmschreibung und Listing verschaffen dem Leser Einblick in das Spiel, so daß es ihm nicht schwerfallen wird, auftretende Schwierigkeiten zu identifizieren und sie unter Rückgriff auf das entsprechende Einführungskapitel zu lösen.

Zeichensatzkopierprogramm
Joystick-Abfrage in Maschinensprache
Sprite-Editor
Geräusch-Effekte
Demon Dozer
3-D Tank Assault
Sub Hunt
Dog Fight
Spider Hunt
Dippy Dappy
Fun Match
Guess my Number
Bug Bovver
Attack of the Tomato
Papier, Schere, Stein
Search Light
Flip-Flap
Laser Spell
Track Racer
Mine Field
Look Out!
Drum Kit
Math. Begegnungen der Dritten Art
Cosmic Dodge
Galactic Battle

Dieses Buch eignet sich für jeden Commodore 64-Anwender: Einsteigern vermittelt es das nötige Programmier-Know-How, Fortgeschrittenen bietet es eine Fülle anspruchsvoller Spiele, Könner entdecken die Kreativität beim Erstellen eigener Programme.

Die Bücher der Reihe „mvg computer lernen“ bieten ein[e]rn-konzept. In unterhaltsamer Weise wird der Leser in di[e] BASIC eingeführt. Er lernt, welche Befehle er der Ma[schine] er grafische Zeichen, Farben und Bewegung auf [dem] kann, wie er Programme schneller macht oder wie e[r] umfangreiche Programmschreibungen einspart.

Alle Listings sind auch als Software erhältlich. Sie ersparen sich damit das Eingeben der Programme.

mvg computer lernen

mvg computer lernen

ISBN 3-478-09040-7