SPRÜHENDE IDEEN

mit ATARI GRAPHIK

Tom Rowley

te-wi

IMPRESSUM

Dieses Buch ist die Übersetzung der amerikanischen Originalausgabe "DESIGNS FROM YOUR MIND with ATARI® GRAPHICS" by Tom Rowley

English edition Copyright © 1983
by Reston Publishing Company, Inc.,
A Prentice-Hall Company, Reston,
Virginia 22090
All rights reserved.

Übersetzung Copyright © 1984
by te-wi Verlag GmbH.

Alle Rechte vorbehalten. Ohne ausdrückliche schriftliche Genehmigung der Herausgeber ist es nicht gestattet, das Buch oder Teile daraus in irgendeiner Form durch Fotokopie, Mikrofilm oder ein anderes Verfahren zu vervielfältigen oder zu verbreiten. Dasselbe gilt für das Recht der öffentlichen Wiedergabe.

LIZENZNEHMER und HERAUSGEBER:
te-wi Verlag GmbH, Theo-Prosel-Weg 1
8000 München 40

Der Herausgeber übernimmt keine Gewähr dafür, daß die beschriebenen Schaltungen, Baugruppen, Verfahren, Programme usw. funktionsfähig und frei von Schutzrechten Dritter sind.

GESAMTHERSTELLUNG:
technik marketing, München
tm4003/1284, 1. Auflage
Printed in Austria

ISBN 3-921803-39-X

INHALTSVERZEICHNIS

TEIL ZWEI

ANHÄNGE

EINFÜHRUNG

Würden Sie gerne die Geheimnisse eines Künstlers erlernen? Jetzt ist Ihre Chance gekommen. Trotzdem werden Sie durch die Geheimnisse, die Sie lernen werden, kein Künstler im üblichen Sinne dieses Wortes. Sie werden ein Graphik-Designer, ein Video-Komponist, ganz allgemein ein Computer-Künstler.

Dieses Buch beschäftigt sich primär mit der Erstellung von Computergraphiken. Graphik-Design auf diese Art ist mehr, als nur die Darstellung eines Computerbildes auf dem Bildschirm. Es ist die Kunst der elektronischen Gestaltung eines Bildwerks. Es ist die Anwendung und Erstellung von Objekten, Farben, Ausgewogenheit, Stimmung, und Bewegung.

Dieses Buch handelt auch von der Kommunikation. Es bespricht die Befehle, die den Computer veranlassen Ihre Ideen auf dem Bildschirm darzustellen, genauso, wie ein Künstler seine Ideen auf dem Zeichenbrett zum Ausdruck bringt. Jeder Künstler braucht sein eigenes spezifisches Werkzeug. Das wesentlichste Werkzeug, das für die Verwendung dieses Buches benötigt wird, ist ein ATARI 800 oder ein ATARI 400 Computer. Er ist die Farbe und der Pinsel des Video-Künstlers in einem. Da viele der in diesem Buch gegebenen Informationen mit Farbe arbeiten, ist ein Farbfernseher oder Farb-Monitor unerläßlich. Er ist das Zeichenbrett des Video-Künstlers.

Ihre kreativen Ideen werden mittels der ATARI BASIC Computersprache über den Computer auf den Bildschirm übermittelt. Im entsprechenden Modulschacht des Computers muß sich also eine ATARI BASIC Cartridge befinden.

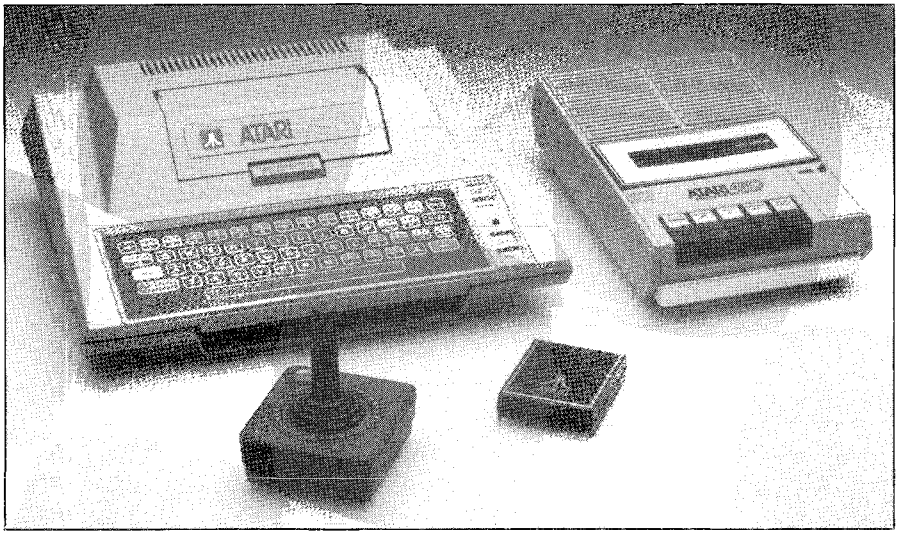
Eine Diskettenstation oder ein Programmrekorder sind nicht unbedingt erforderlich, wenn Sie keine Programme speichern wollen.

Dieses Buch ist ein Lehrbuch, das in zwei Hauptteile unterteilt ist. Teil 1 gibt eine Einführung in die Erstellung von Figuren (Shapes), Farben, und in die Bildschirmgestaltung. Dieser Teil erfordert keine sonderlichen Computerkenntnisse. Er ist für Anfänger geschrieben. Eine gewisse Vertrautheit mit dem ATARI Computer und seiner Funktionsweise wird trotzdem vorausgesetzt. Diese minimalen Kenntnisse können Sie sich z.B. im ATARI Handbuch aneignen. Ein Basiswissen über die Programmiersprache BASIC wäre selbstverständlich nützlich.

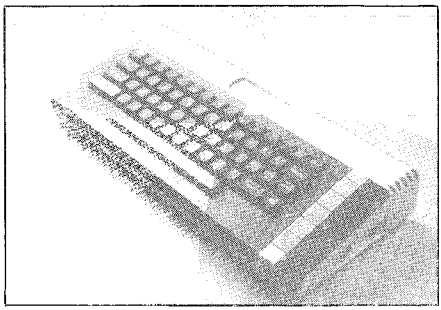
Eine erweiterte Anwendung der Graphiktechniken wird in Teil 2 erklärt. Diese Techniken sind speziell auf die graphischen Fähigkeiten des ATARI Computers und seine interne Architektur abgestimmt. Weiterhin stellt Teil 2 eine Fortsetzung von Teil 1 dar. Kenntnisse in der BASIC Programmierung sind unbedingt erforderlich. Außerdem wäre ein Grundwissen in Maschinensprache von Vorteil.

Wenn Sie beginnen dieses Buch zu lesen, und den Computer mit seiner Programmiersprache verwenden, um die Möglichkeiten der Computergraphik zu erforschen, bedenken Sie stets:

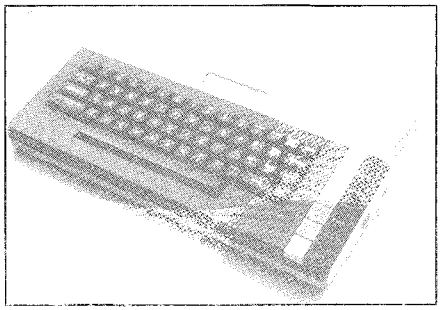
- Dieses Buch beschreibt ein aufregendes Gefühl aus dem Inneren.
- Dieses Buch handelt von einer Stimulanz Ihrer visuellen Gefühle.
- Dieses Buch beschreibt den Sinn einer Befriedigung.
- Dieses Buch gibt Ihnen die Gelegenheit, Ihre Kreativität frei zu entfalten.
- Dieses Buch fördert Ihre graphische Kreativität - Entwürfe aus dem Geist.



Atari 400



Atari 600XL



Atari 800XL

TEIL EINS

1. Das Video Zeichenbrett
2. Figuren, Farben, und Graphik-Modi
3. Farben und Kontraste
4. Drei Dimensionen
5. Zeichensätze
6. Eine Video-Komposition

KAPITEL 1

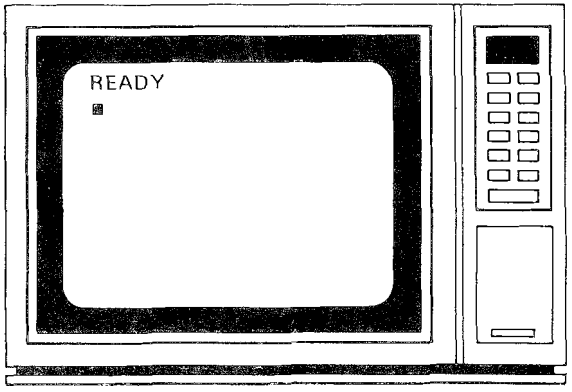
Der Bildschirm

Sind Sie bereit?

Dieses Kapitel ist eine Einführung in die Graphikmöglichkeiten des ATARI. Es ist unsere erste Exkursion in das Reich der Bildschirmkünstler. In diesem Kapitel werden Sie lernen:

- Die Bildschirmfarben zu ändern
- farbigen Text auszudrucken
- auf dem Fernsehbildschirm zu malen

Wir wollen sofort anfangen. Schalten Sie den Computer ein. In der linken oberen Ecke des Bildschirm sollte das Wort READY erscheinen. Darunter müßten Sie den Cursor (das kleine weiße Quadrat) sehen. Wenn irgendetwas nicht funktionieren sollte, lesen Sie bitte im Bedienungshandbuch Ihres ATARI nach.



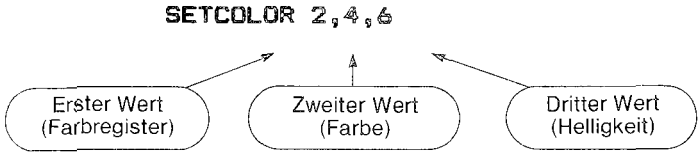
Ändern der Farben

Lassen Sie uns versuchen mit Ihrem Computer in Kommunikation zu treten. Geben Sie die folgende Anweisung ein.

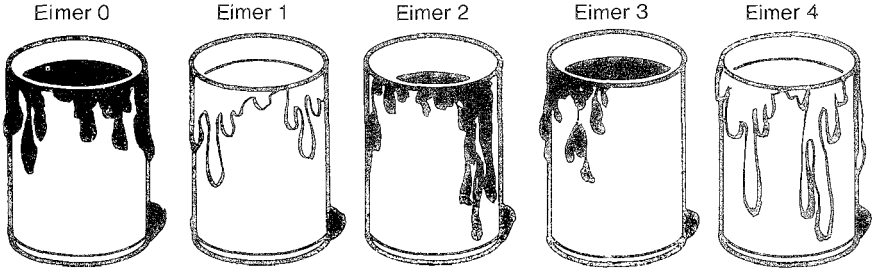
```
SETCOLOR 2,4,6 <RETURN>
```


Hat sich die Farbe des Bildschirms in lila geändert? Wenn dies der Fall ist, so klappt alles bestens. (Hinweis: Die Farben mancher Fernseher können sich unterscheiden oder bedürfen auch einer Einstellung.) Wenn Sie einen Fehler gemacht haben, wird der Computer mit ERROR (einer Fehlermeldung) antworten. Das ist aber überhaupt nicht schlimm. Geben Sie die Anweisung einfach noch einmal neu ein.
Die SETCOLOR-Anweisung wird zur Änderung der Farben auf dem Bildschirm verwendet. Es handelt sich dabei um eine Programmieranweisung der ATARI-BASIC Programmiersprache. Die

SETCOLOR-Anweisung verwendet drei Zahlenwerte zur Änderung der Farben.



Der erste Zahlenwert repräsentiert das Farbregister. Ihr Computer hat insgesamt fünf Farbregister, die Sie mit der SETCOLOR-Anweisung ansprechen können. Diese Farbregister werden verwendet um die Farben des Hintergrunds, der Ränder, Buchstaben und Bilder auf dem Fernsehschirm wiederzugeben. Sie können sich jedes dieser Farbregister als einen Platz, ähnlich wie einen Farbeimer, vorstellen, um dort eine bestimmte Farbe aufzuheben.



Wir können jede beliebige Farbe in die einzelnen Farbeimer (also die Farbregister) füllen. Der zweite und dritte Zahlenwert der SETCOLOR-Anweisung bestimmt die Farbe.
Der zweite Zahlenwert entspricht der Farbnummer. Jede Farbe ist mit einem Wert von 0 bis 15 bezeichnet. Die Farben und ihre entsprechenden Werte sind in Tabelle 1-1 aufgeführt.
Der dritte Zahlenwert stellt die Helligkeit einer Farbe dar. Der Bereich dieses Zahlenwertes kann in Zweierschritten zwischen 0 und 14 liegen. Der Wert 0 repräsentiert den dunkelstens, 14 den hellsten Farbwert.

TABELLE 1-1 Farbwerte

Farbe	entsprechender Zahlenwert (zweite Zahl)
Grau	0
Hellorange	1
Orange	2
Rot-Orange	3
Rosa	4
Lila-Blau	6
Blau	7
Blau	8
Hellblau	9
Türkis	10
Grün-Blau	11
Grün	12
Gelb-Grün	13
Orange-Grün	14
Hellorange	15

Haben Sie alles verstanden? Gut, dann lassen Sie uns eine Testfrage stellen.

Welche Hintergrundfarbe erhält der Fernsehschirm durch die folgende SETCOLOR-Anweisung?

```
SETCOLOR 2,13,4
```

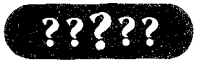
(Hinweis: Welche Farbe wird durch den Wert 13 in Tabelle 1-1 repräsentiert?)

Antwort #1 _____

(Die Antworten zu den Fragen sind jeweils am unteren Ende einer Seite auf dem Kopf dargestellt.)

Ändern Sie die Hintergrundfarbe. Geben Sie ein:

```
SETCOLOR 2,13,4  <RETURN>
```



Welche Anweisung kann zur Änderung der Hintergrundfarbe in ein helles Blau mit einem Helligkeitswert von 4 verwendet werden? Sehen Sie noch einmal in Tabelle 1-1 nach.

Antwort #2 _____

#1. Gelb-Grün
#2. SETCOLOR 2,9,4

Lassen Sie uns ein anderes Farbregister betrachten. Hierzu werden wir die Zahl 4 als ersten Wert in der SETCOLOR-Anweisung verwenden. Geben Sie ein:

SETCOLOR 4,2,6 <RETURN>

Die Randfarbe müßte nun orange sein. Und richtig, das Farbregister 4 ändert die Randfarbe.
Um sie wieder in schwarz zu ändern, geben Sie ein:

SETCOLOR 4,0,0 <RETURN>

Können Sie die Randfarbe in weiß ändern? Welche Anweisung müssen Sie eingeben? (HINWEIS: Verwenden Sie ein helles grau mit einem großen Helligkeitswert.)

Antwort #3 _____

Wollen Sie noch mehr erfahren? Dann lassen Sie uns einmal Farbregister 1 untersuchen. Geben Sie ein:

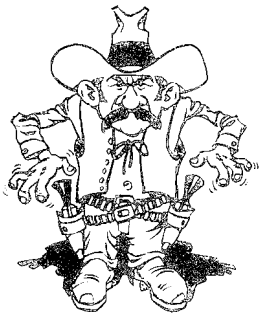
SETCOLOR 1,9,14 <RETURN>

Jetzt müßten die einzelnen Buchstaben stärker leuchten. Dieses Farbregister bestimmt also die Leuchtkraft (Helligkeit) der Buchstaben und Worte auf dem Bildschirm. Die zweite Zahl in der SETCOLOR-Anweisung hat allerdings keine Änderung bewirkt. Die Worte haben immer noch dieselbe Farbe, wie der Hintergrund. Im Augenblick können wir nur ihre Helligkeit und nicht ihre Farbe verändern. Doch keine Angst, zu einem späteren Zeitpunkt werden wir auch die Worte einfärben.
Jetzt wollen wir doch einmal versuchen, die Zeichen auf dem Bildschirm verschwinden zu lassen. Natürlich wollen wir sie nicht wirklich löschen, sondern nur den Anschein erwecken. Wie? Versuchen Sie den Buchstaben dieselbe Farbe zu geben, wie dem Hintergrund. Diese Technik werden wir später zum Verstecken von Objekten auf dem Bildschirm verwenden.
ACHTUNG! Farbregister 2 beeinflusst nicht immer die Hintergrundfarbe. Ähnlich verhält sich Farbregister 4, das nicht immer für die Randfarbe zuständig ist, und Register 1 ändert nicht immer die Helligkeit der Buchstaben. Doch lassen Sie sich jetzt nicht verwirren. Wenn wir in Kapitel 2 näher auf die Farbregister eingehen, werden wir dieses Problem eingehend erläutern.
Jetzt noch etwas zum Aufbau dieses Buches. Nach jedem Abschnitt stellen wir Ihnen eine Aufgabe. Einige werden einfach sein,

#3. SETCOLOR 4,0,14

andere wiederum schwer. Sicherlich gibt es viele Möglichkeiten, die einzelnen Aufgaben zu lösen. Eine mögliche Antwort zu jeder Frage finden Sie in Anhang B. Viel Erfolg!

Hier ist unsere erste Aufgabe.



Aufgabe #1

Ändern Sie die Hindergrundfarbe in orange.
Ändern Sie die Randfarbe in blau.
Machen Sie die Worte so dunkel, wie möglich.

Der Graphik-Bildschirm

Farben sind schon etwas Tolles. Aber wie können wir auf dem Bildschirm Text schreiben und gleichzeitig farbige Bilder malen? FOLGENDES AUF DEM KOPF: #2. SETCOLOR 2,9,4 #3. SETCOLOR 4,0,14 Ganz einfach, lesen Sie weiter, und wir werden es Ihnen zeigen. Drücken Sie zuerst einmal die <SYSTEM RESET> Taste, damit der Bildschirm wieder so aussieht, wie nach dem Einschalten des Computers. Das können Sie übrigens immer machen, wenn Sie neu starten wollen.

Malen und Schreiben auf dem Fernsehschirm ist gar nicht so schwierig. Doch zuvor müssen wir einen GRAPHIK-Modus (Stufe) wählen. Der ATARI Computer verfügt über eine ganze Reihe verschiedener GRAPHIK-Modi. Sie dienen dazu, dem Computer mitzuteilen, in welcher Größe er Schrift und in welcher Auflösung er Zeichnungen darstellen soll. Jeder dieser Modi verwendet ein anderes Format für die Graphik. Tabelle 1-2 zeigt die einzelnen Graphik-Modi und das entsprechende Bildschirmformat.

Um die Tabelle besser zu verstehen, wollen wir uns GRAPHIK-Modus 0 anschauen. GRAPHIK-Modus 0 ist ein sogenannter Text-Modus. In einem Text-Modus kann Text (also Buchstaben) auf dem Bildschirm geschrieben werden. Der gesamte Bildschirm ist in diesem Fall 40 Spalten breit und 24 Zeilen hoch. (Geteilte Bildschirme besprechen wir später.) In diesem Modus können zwei Farben verwendet werden. Eine detailliertere Erklärung dieses GRAPHIK-Modus folgt in den Kapiteln 2, 3, 4, und 5.

Tabelle 1-2. GRAPHIK-Modi und Bildschirmformate

GRAPHIK-Modus	Art	Bildschirmformat: Spalten	Bildschirmformat: Reihen gesamter/geteilter Bildschirm	Zahl der Farben
0	Text	40	24/--	2
1	Text	20	24/20	5
2	Text	20	12/10	5
3	Graphik	40	24/20	4
4	Graphik	80	48/40	2
5	Graphik	80	48/40	4
6	Graphik	160	96/80	2
7	Graphik	160	96/80	4
8	Graphik	320	192/160	2
9	Graphik	80	192/--	16
10	Graphik	80	192/--	9
11	Graphik	80	192/--	16

Anmerkung: Die Modi 9, 10, und 11 funktionieren nur, wenn der Computer einen GTIA Video-Chip hat.

Schreiben von Text

Ganz allgemein gesagt, gibt es nur zwei mögliche Darstellungsformen auf dem Bildschirm - Text und Graphik. In den Text-Modi (GRAPHICS 0, 1, und 2) sind drei verschiedene Größen zur Darstellung der Zeichen möglich. Die Schriftgröße in GRAPHIK-Modus 0 haben Sie ja schon gesehen. Obwohl sich der Bildschirm vermutlich noch im GRAPHIK-Modus 0 befindet, wollen wir dem Computer mitteilen, weiterhin GRAPHIK-Modus 0 zu verwenden. Geben Sie bitte Folgendes ein:

```
GRAPHICS 0  <RETURN>
```

Beachten Sie, daß die GRAPHICS-Anweisung auch das Löschen des Bildschirmes bewirkt. Um nun Text auf dem Bildschirm zu schreiben, können Sie die PRINT-Anweisung verwenden. Geben Sie bitte ein:

```
PRINT "GRAPHICS MODUS 0"  <RETURN>
```

Der in Anführungszeichen stehende Satz wird sofort direkt unter der Anweisung auf den Bildschirm geschrieben. Aber nehmen wir an, Sie wollten ihn an irgendeine andere Stelle des Bildschirmes

setzen. Um diesen Satz an einer bestimmten Position zu schreiben, geben Sie bitte folgendes ein:

```
POSITION 10,15:PRINT "GRAPHIK-MODUS 0"  <RETURN>
```

Wie Sie sehen, haben wir zwei Anweisungen in eine Zeile geschrieben. Zuerst haben wir die Position bestimmt und dann die Anweisung zum Schreiben des Textes gegeben. Die beiden Anweisungen sind durch einen Doppelpunkt getrennt. Die erste Schreibposition befindet sich jetzt 10 Zeichen von der linken Seite des Bildschirmes und 15 Zeilen vom oberen Rand des Bildschirmes. Auch dies werden wir in Kapitel 5 noch einmal ausführlicher besprechen.
Jetzt wollen wir einen GRAPHIK-Modus 1 Bildschirm betrachten. Geben Sie die folgende Anweisung ein:

```
GRAPHICS 1  <RETURN>
```

Na sowas!? Was ist passiert? Der Bildschirm hat sich geteilt. Der obere Teil des Bildschirmes befindet sich im GRAPHIK-Modus 1, wogegen sich der untere nach wie vor blaue Teil immer noch im GRAPHIK-Modus 0 befindet. Dieser untere Teil wird auch oft als Textfenster bezeichnet. Geben Sie jetzt die folgende Anweisung ein:

```
PRINT#6;"GRAPHICS 1"  <RETURN>
```

Und da haben wir es - zwei verschiedene Schriftgrößen auf einem Bildschirm. Mit PRINT#6; wird dem Computer gesagt, in der oberen Hälfte eines geteilten Bildschirmes zu schreiben. Der Satz innerhalb der Anführungszeichen wird in bekannter Form auf den Bildschirm geschrieben. Beachten Sie, daß im GRAPHIK-Modus 1 die Buchstaben doppelt so breit sind, wie in GRAPHIK-Modus 0. Ob Sie es glauben oder nicht, die Höhe ist gleich geblieben. Beachten Sie weiterhin, daß sich die Hintergrund- und die Zeichenfarbe geändert hat.
So weit, so gut. Lassen Sie uns jetzt auch die Schriftgröße im GRAPHIK-Modus 2 betrachten. Geben Sie die folgende Anweisung ein:

```
GRAPHICS 2  <RETURN>
```

Der Bildschirm sollte gelöscht worden sein. Geben Sie jetzt ein:

```
PRINT#6;"GRAPHICS 2"  <RETURN>
```

Und da sehen wir es! Im GRAPHIK-Modus 2 werden die Buchstaben doppelt so breit und doppelt so hoch wie im GRAPHIK-Modus 0 dargestellt.

Welche zwei Anweisungen würden Sie verwenden, um ENDE in doppeltbreiten und einfach-hohen Buchstaben auf den Bildschirm zu schreiben?

Antwort #4 _____
_____

Zu einem späteren Zeitpunkt werden wir Ihnen zeigen, wie Sie alle drei eben besprochenen GRAPHIK-Modi auf einem einzigen Bildschirm mischen können. Doch zuvor wollen wir noch einige andere GRAPHIK-Modi betrachten.

Zeichnen auf dem Bildschirm

Wir wollen mit GRAPHIK-Modus 3 beginnen. Und Sie haben es sich vermutlich schon gedacht; geben Sie ein:

FOLGENDES BITTE AUF DEM KOPF: #4.GRAPHICS 1 PRINT#6;"ENDE"

```
GRAPHICS 3  <RETURN>
```

Und wieder wird der Bildschirm gelöscht.

Normalerweise ist es nicht möglich auf einem GRAPHIK-Bildschirm Text zu schreiben. Trotzdem kann man einzelne Punkte auf den Bildschirm plotten (setzen). Dabei muß man allerdings zwei Dinge beachten. Erstens müssen Sie dem Computer mitteilen welche Farbe Sie verwenden wollen. Zweitens, müssen Sie ihm sagen, an welchen Bildschirmkoordinaten er den Punkt plotten soll.
Die Farbe teilen wir dem Computer mit der COLOR-Anweisung mit. In unserem Beispiel wollen wir die Farbe mit der Nummer 1 verwenden. Geben Sie jetzt bitte die folgende Anweisung ein:

```
COLOR 1  <RETURN>
```

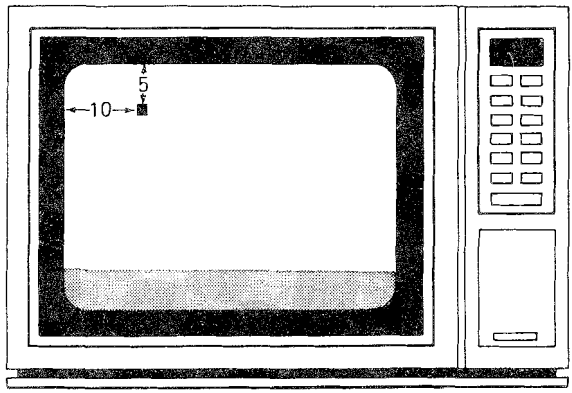
Wie Sie sehen hat sich nichts geändert. Doch warten Sie ab, bis etwas auf den Bildschirm geplottet wird. Geben Sie ein:

```
PLOT 10,5  <RETURN>
```

Auf dem Bildschirm erscheint ein oranges Quadrat. Orange, da das durch die COLOR 1-Anweisung verwendete Farbregister normalerweise diese Farbe enthält.

#4.GRAPHICS 1
PRINT#6;"ENDE"

Lassen Sie uns die COLOR-Anweisung etwas näher untersuchen. Die obere Hälfte des geteilten GRAPHIK-Modus 3 Bildschirmes ist in 40 horizontale und 20 vertikale Quadrate unterteilt. Lesen Sie noch einmal Tabelle 1-2 und betrachten Sie das GRAPHIK-Modus 3 Arbeitsblatt in Anhang A. Die Koordinate 10,5 befindet sich in der oberen linken Ecke des Bildschirmes. Das bedeutet, daß 10 leere Quadrate auf der linken Seite, und 5 leere Quadrate oberhalb der Koordinate 10,5 frei bleiben. In gewisser Hinsicht kann man die PLOT-Anweisung mit der POSITION-Anweisung vergleichen. Die POSITION-Anweisung bestimmt eine Position auf dem Bildschirm, an dem dann ein Text geschrieben wird, wogegen die PLOT-Anweisung die entsprechende Position mit einem farbigen Punkt markiert.



Wo befindet sich auf dem Bildschirm die Position 30,5? (Oben rechts, oben links, unten rechts, oder unten links?)

Antwort #5 _____

Mit der folgenden Anweisung können wir eine Linie zeichnen (eine sehr dicke Linie). Versuchen Sie es.

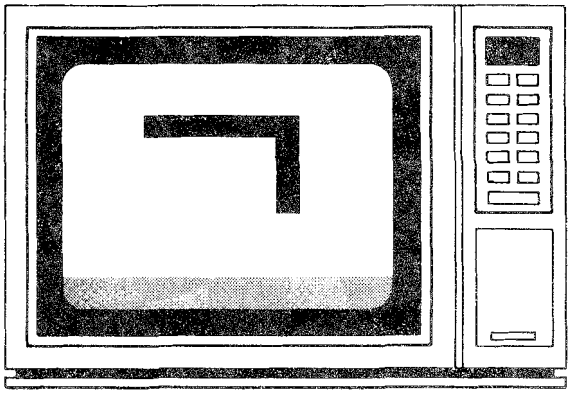
```
DRAWTO 30,5  <RETURN>
```

Toll, nicht wahr? Und jetzt geben Sie bitte ein:

```
DRAWTO 30,15  <RETURN>
```

Was Sie auf Ihrem Bildschirm sehen, sollte jetzt mit unserem Beispiel übereinstimmen.

#5. Oben rechts



Wir wollen das Rechteck fertig zeichnen. Dazu brauchen wir zwei weitere DRAWTO-Anweisungen.

```
DRAWTO 10,15  <RETURN>

DRAWTO 10,5  <RETURN>
```

Die gesamte Befehlsfolge sieht so aus:

```
GRAPHICS 3
COLOR 1
PLOT 10,5
DRAWTO 30,5
DRAWTO 30,15
DRAWTO 10,15
DRAWTO 10,5
```

Diese Folge von Anweisungen ergibt ein in der Programmiersprache ATARI BASIC geschriebenes Computer Programm, zum Zeichnen eines Rechtecks in GRAPHIK-Modus 3. In der oben aufgeführten Form wird jede Anweisung, die dem Computer eingegeben wird, sofort ausgeführt. Der Computer kann sich die einzelnen Anweisungen zum Zeichnen des Rechtecks allerdings nicht merken.
Führen Sie die folgenden Anweisungen aus, damit der Computer sich die gesamte Folge der Anweisungen merken kann:

1. Drücken Sie <SYSTEM RESET>

2. Tippen Sie NEW <RETURN>. Dieser Befehl löscht jedes Programm (Folge von Anweisungen), das eventuell im Computer gespeichert war.

3. Geben Sie jetzt die einzelnen Programm-Anweisungen noch einmal ein, mit dem Unterschied, daß Sie vor jeder Anweisung eine Zeilennummer eingeben. Der Computer verwendet diese Zeilen-

nummern, um sich die richtige Reihenfolge, in der er die Anweisungen ausführen muß,zu merken. Die Werte der Zeilennummern haben keine Bedeutung. Verwenden Sie Zahlen in aufsteigender Ordnung. Wenn Sie einen Fehler machen, geben Sie die entsprechende Zeile neu ein. Um eine Zeile zu löschen, geben Sie einfach die entsprechende Zeilennummer ein und drücken dann direkt RETURN. Sollten irgendwelche Schwierigkeiten auftreten, lesen Sie noch einmal im Bedienungshandbuch nach.

```
10 GRAPHICS 3
20 COLOR 1
30 PLOT 10,5
40 DRAWTO 30,5
50 DRAWTO 30,15
60 DRAWTO 10,15
70 DRAWTO 10,5
```

Denken Sie daran am Ende jeder Zeile RETURN zu drücken.

4. Um zu überprüfen, ob Sie alles richtig eingegeben haben, tippen Sie

   LIST <RETURN>

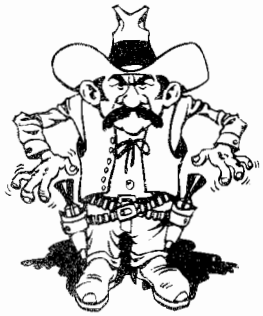
Auf dem Bildschirm sollte jetzt eine Liste der eingegebenen Zeilen erscheinen. Wenn Sie nicht exakt mit den oben aufgeführten Anweisungen übereinstimmt, korrigieren Sie entsprechend den Fehler.

5. Ok, sind Sie bereit für den großen Augenblick? Gut, dann geben Sie ein:

   RUN <RETURN>

Und sofort wird ein Rechteck auf dem Bildschirm gezeichnet. Der RUN-Befehl weist den Computer an, alle Programmzeilen in der Reihenfolge der Zeilennummern abzuarbeiten. Der Computer hat sich bisher zwar alle Anweisungen gemerkt, doch erst ausgeführt, als wir den Befehl RUN verwendeten.

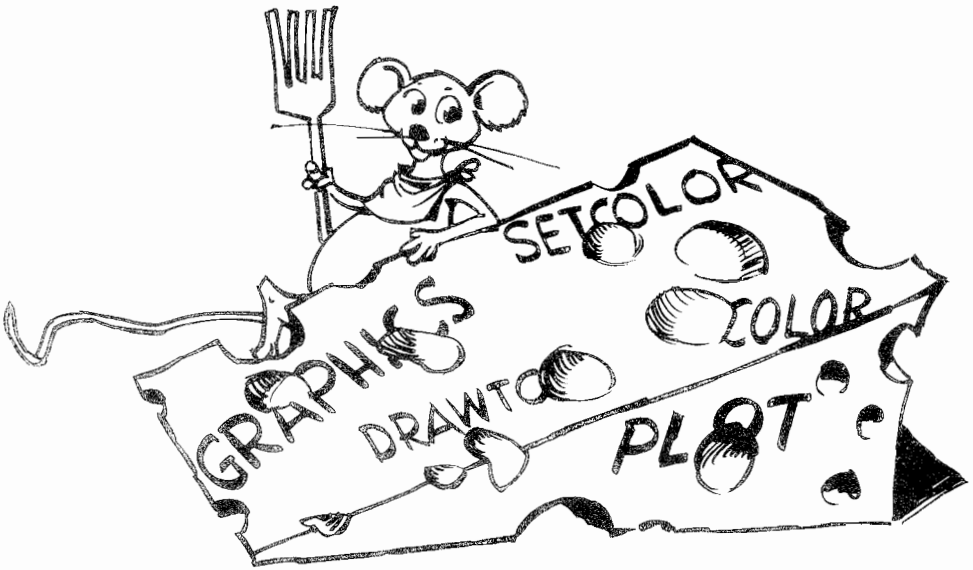
Ein ähnliches Rechteck könnten wir auch in den GRAPHIK-Modi 4 bis 11 erzeugen. Die Art der Anweisungen bleibt gleich. In Kapitel 2 werden wir näher auf dieses Thema eingehen.



Aufgabe #2

Schreiben Sie ein Programm zum Zeichen eines Quadrates in GRAPHIK-Modus 3. Jede Seite sollte 10 Einheiten lang sein. Die obere linke Ecke des Quadrates soll auf Position 10,5 liegen.

Sie haben in diesem ersten Kapitel zwar eine ganze Menge gelernt, doch werden Sie Ihr Wissen sicher als sehr löchrig empfinden - fast wie einen Schweizer Käse. Es scheinen doch noch recht viele "Löcher" in dem erlernten Lehrstoff zu sein.



In den nächsten fünf Kapiteln werden wir die einzelnen hier in Kapitel 1 nur angedeuteten Themen ausführlich erklären. Doch zuvor wollen wir uns noch die Zusammenfassung des ersten Kapitels anschauen.

Zusammenfassung von Kapitel 1

In diesem Kapitel haben Sie die folgenden ATARI BASIC Programmanweisungen verwendet:

SETCOLOR GRAPHICS COLOR PRINT
PLOT DRAWTO POSITION

Können Sie sich daran erinnern, wozu jede Anweisung verwendet wurde? Vergleichen Sie jede der oben aufgeführten GRAPHIK-Anweisungen mit den folgenden Beschreibungen.

1. _____ löscht den Bildschirm und setzt ein neues Bildschirmformat.

2. _____ ändert die Werte der Farbregister. Kann zum Ändern der Hintergrund-, Rand-, und Textfarben benutzt werden.

3. _____ stellt einen Punkt (Quadrat) auf dem Bildschirm dar.

4. _____ wird verwendet, um die Farbe von Punkten und Linien zu bestimmen, in der diese auf dem Bildschirm gezeichnet werden sollen.

5. _____ wird verwendet, um zwei Punkte mit einer Linie zu verbinden.

6. _____ dient zum Festlegen einer Position, ab der die durch eine folgende PRINT-Anweisung auszugebenden Zeichen auf den Bildschirm geschrieben werden sollen.

7. _____ bewirkt, daß Text in den GRAPHIK-Modi 0, 1, und 2 auf dem Bildschirm gezeigt werden kann.

Die folgenden Anweisungen ändern die Hintergrundfarbe des Bildschirmes im GRAPHIK-Modus 0. Welche Farben erhält der Hintergrund durch welche Anweisung?

8. SETCOLOR 2,4,8 _____

9. SETCOLOR 2,12,8 _____

10. SETCOLOR 2,2,8 _____

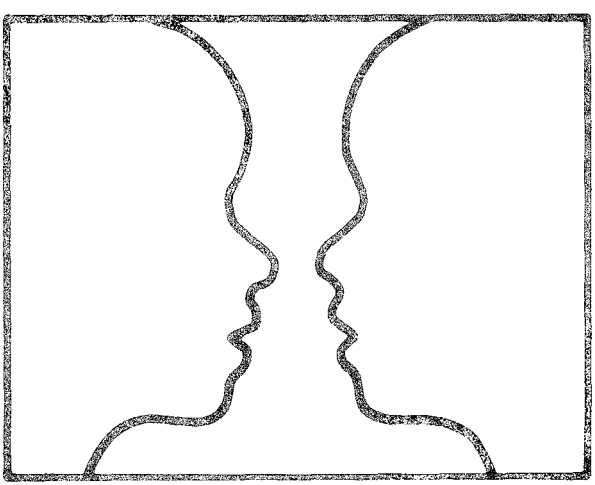
11. Das folgende Programm zeichnet eine Figur auf den Bildschirm. Skizzieren Sie die Form der Figur auf einem GRAPHIK-Modus 3 Arbeitsblatt.

```
10 GRAPHICS 3
20 COLOR 1
30 PLOT 2,10
40 DRAWTO 2,15
50 DRAWTO 37,15
60 DRAWTO 37,10
```

KAPITEL 2

Formen, Farben und GRAPHIK-Modi

Die Form ist eines der Grundelemente zum Erfassen optischer Reize. Wir identifizieren Figuren auf Grund ihrer Form. Sie können die Buchstaben auf dieser Seite anhand ihrer Form erkennen. Außerdem ordnen wir bestimmten Formen bestimmte Bedeutungen zu. Was würden Sie zum Beispiel mit dieser Form in Verbindung bringen?



Es könnte eine Vase sein, oder aber auch zwei miteinander sprechende Personen.
Die Form und Gestaltung der Darstellung auf dem Bildschirm trägt den meisten Informationsgehalt. Er wird durch Farben noch erweitert, doch am wichtigsten für den Video-Künstler ist die Möglichkeit zu zeichnen.

In diesem Kapitel werden wir die ATARI BASIC GRAPHIK-Modi untersuchen. Sie werden lernen:

- Figuren unterschiedlicher Größe zu zeichnen
- die Farbe dieser Figuren zu ändern
- die GRAPHIK-Modi 3 bis 8 zu verwenden

Figuren und der Graphik Bildschirm

In Kapitel 1 haben wir eine Figur gezeichnet. Es war ein Rechteck in GRAPHIK-Modus 3. Das Programm zum Zeichnen des Rechtecks war wirklich sehr einfach.
In Bild 2-1 sehen Sie den Umriß des Rechtecks auf einem GRAPHIK-Modus 3 Arbeitsblatt. Sie sehen, so können die Arbeitsblätter in Anhang A sinnvoll verwendet werden und Ihnen bei der Gestaltung von Zeichnungen und Figuren helfen.

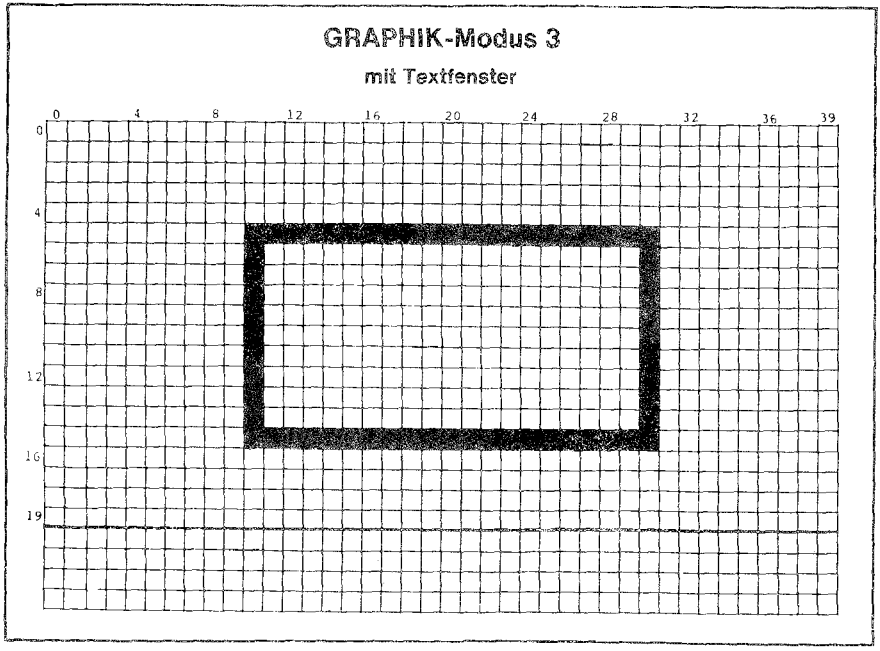


BILD 2-1: Umriß eines Rechtecks auf einem Modus 3 Arbeitsblatt.

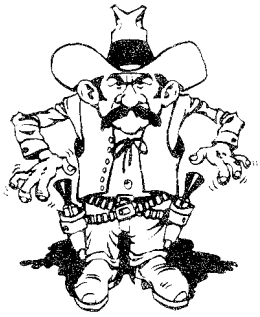
Erinnern Sie sich an die Anweisungen zum Zeichnen des Rechtecks aus Kapitel 1.

```
10 GRAPHICS 3
20 COLOR 1
30 PLOT 10,5
40 DRAWTO 30,5
50 DRAWTO 30,15
60 DRAWTO 10,15
70 DRAWTO 10,5
```

Die obere linke Ecke des Bildschirms hat die Koordinate 0,0. Die untere rechte Ecke wird durch die Koordinate 39,0 gekennzeichnet. Das bedeutet, daß die horizontalen Koordinaten sich von 0 bis 39 erstrecken, und somit ein Bildschirmformat von 40 Positionen Breite ergeben.
Ähnlich verhält es sich mit dem vertikalen Format. Obwohl der Bildschirm insgesamt 24 Positionen hat, ist die unterste Position auf der linken Seite des Bildschirmes mit der Koordinate 0,23 gekennzeichnet. Das kommt daher, daß auch hier die 0 als eigenständige Position mitgezählt wird.
Merken Sie sich also: Der Minimalwert einer Koordinate ist 0. Der Maximalwert ist dann einen Wert kleiner, als die gesamte Anzahl der möglichen Positionen.

In Bezug auf das Arbeitsblatt wollen wir auch noch eine Anmerkung machen. Der GRAPHIK-Modus 3 kann, wie die meisten anderen GRAPHIK-Modi auch, entweder in einem geteilten oder einem kompletten Bildschirmformat verwendet werden. Bisher haben wir nur das geteilte Format benutzt. Später in diesem Kapitel werden wir auch das komplette Format anwenden.
Im geteilten Bildschirmformat ist der Bildschirm in zwei Teile unterteilt. Im oberen Teil des Bildschirmes kann sich jeder beliebige GRAPHIK-Modus zwischen 1 und 8 befinden. Der untere Teil des Bildschirmes befindet sich im GRAPHIK-Modus 0. In diesem unteren Teil , auch Textfenster genannt, finden 4 Zeilen GRAPHIK-Modus 0 Text Platz. Zu einem späteren Zeitpunkt werden wir lernen, wie wir dieses Textfenster an verschiedenen Positionen des GRAPHIK Bildschirmes plazieren können.

In Aufgabe #2 sollten wir ein Quadrat zeichnen. Wenn man die Koordinaten anpasst, kann man Quadrate verschiedener Größe an beliebigen Positionen auf dem Bildschirm zeichnen. Dies ist Ihre nächste Aufgabe.



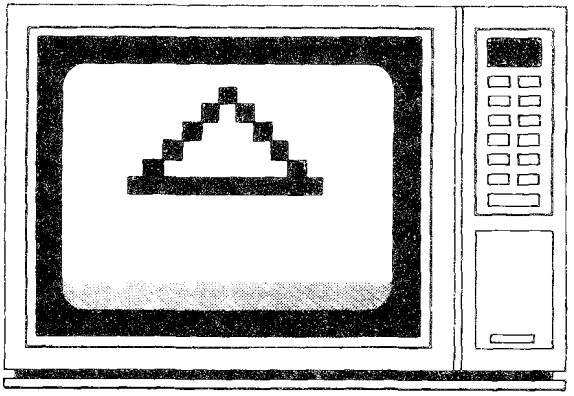
Aufgabe #3

Schreiben Sie eine Liste der Anweisungen, das ein Quadrat mit einer Seitenlänge von sechs Einheiten zeichnet. Seine Position sollte sich in der Mitte der oberen Hälfte eines geteilten Bildschirmformats im GRAPHIK-Modus 3 befinden.

Um einmal eine andere Figur darzustellen, hier ein Programm zum Zeichnen eines Dreiecks:

```
10 GRAPHICS 3
20 COLOR 1
30 PLOT 20,5
40 DRAWTO 30,15
50 DRAWTO 10,15
60 DRAWTO 20,5
```

Tippen Sie NEW, um eventuell im Speicher befindliche alte Programme zu löschen. Geben Sie dann die einzelnen Programmzeilen ein. Wir setzen voraus, daß wir für den Rest des Buches nicht mehr erwähnen müssen, daß nach der Eingabe jeder Programmzeile die RETURN-Taste gedrückt werden muß.
Starten Sie das Programm. Der Bildschirm sollte aussehen wie dieser:



Haben Sie den GRAPHIK-Modus 3 Bildschirm verstanden?

Welche Programmzeile könnten Sie ändern, um die Seitenlänge des Dreiecks zu verändern?

Antwort #1 _____

Lassen Sie uns die Größe des Dreiecks verändern. Ändern Sie die folgenden zwei Zeilen.

```
40 DRAWTO 35,15

50 DRAWTO 5,15
```

Starten Sie das Programm erneut. Sieht das Dreieck anders aus? Beantworten Sie jetzt eine weitere Frage.

Welche zwei Anweisungen würden Sie verwenden, um von der oberen linken Ecke des Bildschirms eine Diagonale zur unteren rechten Ecke des Bildschirms zu zeichnen? Der Bildschirm sollte sich im GRAPHIK-Modus 3 befinden.

Antwort #2 _____
_____

Die GRAPHIK-Modi

Mit dem ATARI Computer stehen Ihnen eine Anzahl verschiedener GRAPHIK-Modi zur Verfügung. In diesem Kapitel werden wir uns mit den Modi 3 bis 8 beschäftigen. Die anderen GRAPHIK-Stufen besprechen wir dann in späteren Kapiteln. Der Unterschied zwischen den einzelnen Modi liegt primär in den verschiedenen Auflösungen und in den vom Programmierer verwendeten Farben.
Lassen Sie uns einige einfache Experimente durchführen. Damit wir anfangen können, löschen wir den Bildschirm,indem wir die <SYSTEM RESET> Taste drücken. Löschen Sie auch alle alten im Speicher befindlichen Programme mit Hilfe des NEW-Befehls. Erinnern Sie sich, daß das immer notwendig ist, wenn Sie ein neues Programm eingeben wollen.

#1. Zeilen 40 und 50 #2. PLOT 39,0
DRAWTO 0,19

Geben Sie jetzt das folgende Programm ein.

```
10 GRAPHICS 3
20 COLOR 1
30 PLOT 20,10
```

Starten Sie das Programm. Das Quadrat, das auf dem Bildschirm erscheint, bezeichnet man als ein Pixel. Die Größe dieses Pixels ist vom verwendeten GRAPHIK-Modus abhängig.
Ändern Sie in dem Programm Zeile 10 in GRAPHICS 4 und starten Sie es. Sie brauchen Zeile 10 lediglich neu zu schreiben. Ihr Programm sollte dann wie unten gezeigt aussehen. Listen Sie es mit der LIST-Anweisung, um zu überprüfen, ob es identisch ist.

```
10 GRAPHICS 4
20 COLOR 1
30 PLOT 20,10
```

Starten Sie das Programm. Die Größe des Pixels ist kleiner, als im GRAPHIK-Modus 3. Probieren Sie das Programm auch mit den GRAPHIK-Modi 5, 6, 7, und 8 aus, und beobachten Sie die Größe des Pixels.

Welcher GRAPHIK-Modus hat die höchste Auflösung (kleinste Pixelgröße)?

Antwort #3 _____

Sie haben sicher bemerkt, daß die GRAPHIK-Modi 4 und 5 sowie die Modi 6 und 7 jeweils dieselbe Pixelgröße haben. Wir werden zu einem späteren Zeitpunkt sehen, daß sie nicht ganz gleich sind, und sich in der Anzahl an darstellbaren Farben unterscheiden. In Modus 8 änderte sich die Hintergrundfarbe und in der linken oberen Ecke des Bildschirmes befand sich ein sehr kleines Pixel. Doch jetzt wollen wir uns wieder mit unserem Dreieck beschäftigen. Was glauben Sie, wie es im GRAPHIK-Modus 8 aussieht? Geben Sie das folgende Programm ein:

```
10 GRAPHICS 8
20 COLOR 1
30 PLOT 20,5
40 DRAWTO 30,15
50 DRAWTO 10,15
60 DRAWTO 20,5
```

Das Dreieck erscheint erheblich klarer und geradliniger als im GRAPHIK-Modus 3. Es hat sich allerdings auch seine Position auf dem Bildschirm geändert und es ist wesentlich kleiner geworden. Das kommt daher, daß im GRAPHIK-Modus 8 die Pixel kleiner sind, und so ist das ganze Dreieck kleiner geworden und hat seine Position jetzt in der oberen linken Ecke des Bildschirmes.

Ein Haus bauen

Gut, jetzt ist es Zeit für ein Projekt. Wir wollen ein einfaches kleines Haus darstellen. Der obere Teil des Hauses besteht aus einem Dreieck, und der untere Teil aus einem Quadrat. Praktischerweise haben wir diese beiden Formen ja schon gezeichnet.
Wir wollen GRAPHIK-Modus 5 für unsere Zeichnung verwenden. Verwenden Sie ein GRAPHIK-Modus 5 Arbeitsblatt aus Anhang A. Es wäre übrigens eine gute Idee, wenn Sie die verschiedenen Arbeitsblätter fotokopieren würden. Beginnen Sie mit dem Entwurf in der Mitte des Arbeitsblattes. Das Ganze sollte dann in etwas so aussehen, wie in Bild 2-2.
Das Programm zum Zeichnen des Hauses enthält die Anweisungen zum Zeichnen eines Dreiecks und eines Quadrates.

```
210 GRAPHICS 5
220 COLOR 1
310 PLOT 40,5
320 DRAWTO 50,15
330 DRAWTO 30,15
340 DRAWTO 40,5
```

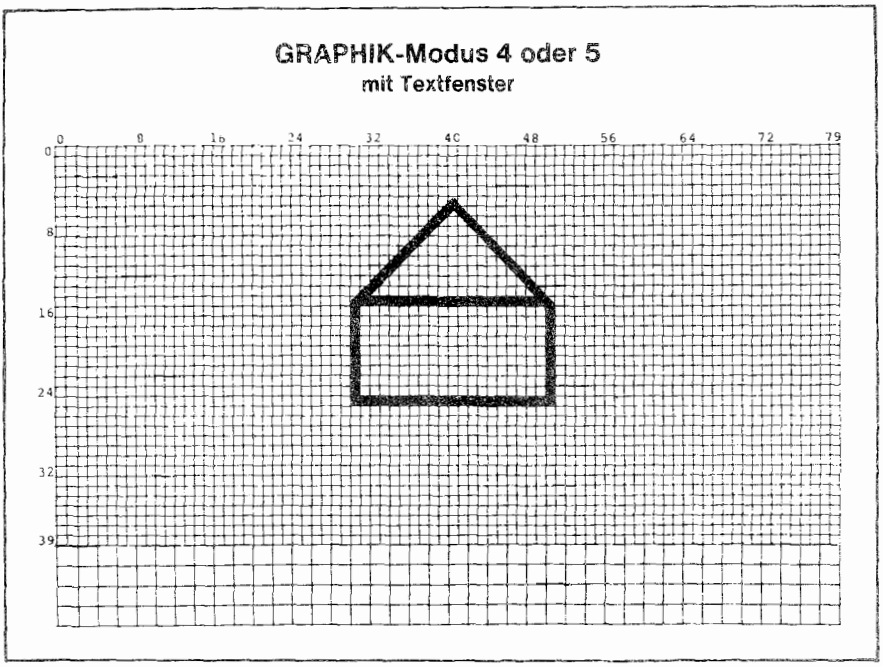


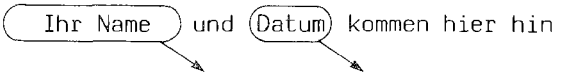
BILD 2-2: Entwurf eines Hauses auf einem Modus 5 Arbeitsblatt.

```
410 PLOT 30,15
420 DRAWTO 50,15
430 DRAWTO 50,25
440 DRAWTO 30,25
450 DRAWTO 30,15
```

Geben Sie diese Zeilen ein, und starten Sie das Programm. Sieht das Haus so aus, wie Sie es erwartet haben?

Bevor wir fortfahren, wollen wir uns doch die Zeit nehmen, um zu untersuchen was wir eigentlich gemacht haben.
Ein sehr wichtiger Faktor in der Computerprogrammierung ist die Organisation und Dokumentation der Arbeit. Sie können dann Ihre Arbeit speichern und zu einem späteren Zeitpunkt anhand der Dokumentation problemlos alle Schritte und Gedanken nachvollziehen. Außerdem wird es für andere einfacher sein, sich in Ihre Arbeit hineinzudenken.
Wir könnten das oben dargestellte Programm dokumentieren, indem wir vier REM-Anweisungen (remark) einfügen würden. Diese REM-Anweisungen haben keinen Einfluß auf den Programmablauf. Sie können in separaten Zeilennummern, aber auch zusammen mit anderen Anweisungen in einer Zeile abgelegt werden. Wenn Sie sie in Zeilen mit anderen Anweisungen zusammen schreiben wollen,

bedenken Sie, daß sie nur am Ende der Zeilen als letzte Anweisungen geschrieben werden können. Außerdem müssen sie durch einen Doppelpunkt von den vorhergehenden Anweisungen getrennt werden. Fügen Sie die folgenden Zeilen zu unserem Programm hinzu. Geben Sie sie genau so ein, wie wir sie hier zeigen.




```
100 REM EIN KLEINES HAUS VON DIRK AM 25 MAI 1984
200 REM ERSTELLT DEN GRAPHIK BILDSCHIRM
300 REM ZEICHNET DAS DACH DES HAUSES
400 REM ZEICHNET DEN UNTEREN TEIL DES HAUSES
```

Außerdem sollte jedes Programm eine sogenannte END-Anweisung beinhalten. Diese Anweisung sagt dem Computer, daß das Programm jetzt fertig ist. Fügen Sie also eine END-Anweisung in Ihr Programm ein:

```
500 END
```

Starten Sie jetzt das Programm noch einmal. Es wird genauso funktionieren, wie zuvor.

Ändern der Hausfarbe

Kommen wir wieder zu unserem Haus. Nehmen wir einmal an, Sie möchten kein gelbes Haus. Nehmen wir weiterhin an, Sie möchten Ihr Haus rosa streichen. Lassen Sie es uns versuchen. Fügen Sie die folgende Programmzeile zu Ihrem Programm:

```
230 SETCOLOR 0,4,10
```

Starten Sie das Programm. Jetzt haben Sie ein rosa gestrichenes Haus.
Das war jetzt etwas verwirrend, da Sie nicht wissen konnten, daß die Anweisung SETCOLOR 0,4,10 die Farbe des Hauses ändert! Hier ist die Erklärung: Die COLOR- und SETCOLOR-Anweisungen arbeiten sozusagen zusammen, wenn Sie etwas auf dem Bildschirm plotten oder zeichnen. Die COLOR-Anweisung spezifiziert ein Farbregister, das dann beim Zeichnen verwendet wird. Die SETCOLOR-Anweisung wird dann verwendet, um die Farbe dieses Registers zu ändern.

Das erscheint doch etwas kompliziert. Betrachten wir einmal die Farben, die wir in unserem Haus-Programm verwendet haben. Wenn wir das Programm listen, sollte es so aussehen:

```
100 REM EIN KLEINES HAUS VON DIRK AM 25 MAI 1984
200 REM ERSTELLT DEN GRAPHIK BILDSCHIRM
210 GRAPHICS 5
220 COLOR 1
230 SETCOLOR 0,4,10
300 REM ZEICHNET DAS DACH DES HAUSES
310 PLOT 40,5
320 DRAWTO 50,15
330 DRAWTO 30,15
340 DRAWTO 40,5
400 REM ZEICHNET DEN UNTEREN TEIL DES HAUSES
410 PLOT 30,15
420 DRAWTO 50,15
430 DRAWTO 50,25
440 DRAWTO 30,25
450 DRAWTO 30,15
500 END
```

Beachten Sie, daß insgesamt drei Zeilen die Farbe des Hauses beeinflussen. Es sind die Zeilen:

```
210 GRAPHICS 5
220 COLOR 1
230 SETCOLOR 0,4,10
```

Tabelle 2-1 beschreibt die Verbindungen zwischen den GRAPHIK-, COLOR- und SETCOLOR-Anweisungen.
Schauen Sie in die Tabelle und suchen Sie GRAPHIK-Modus 5. Da wir die Anweisung COLOR 1 in unserem Programm verwendet haben, suchen Sie die Zahl 1 in der Spalte mit dem Titel COLOR und in derselben Reihe wie GRAPHIK-Modus 5. Beachten Sie, daß die gesamte Zeile für unser Beispiel in Tabelle 2-1 schattiert gedruckt ist. Wenn Sie in der entsprechenden Reihe nachlesen, sollte Ihnen auffallen, daß für den GRAPHIK-Modus 5, COLOR 1 zusammen mit SETCOLOR für Register 0 zu verwenden ist. In unserem Haus Programm haben wir die Farbe mit der Anweisung SETCOLOR 0,4,10 in rosa umgewandelt.
Jetzt wollen wir das Haus blau einfärben. Ändern Sie Zeile 230 in:

```
230 SETCOLOR 0,8,6
```

Starten Sie das Programm erneut. Ist das Haus blau? Und wieder haben wir zum Ändern von COLOR 1 im GRAPHIK-Modus 5 die Anweisung SETCOLOR für das Register 0 verwendet.

TABELLE 2-1. GRAPHIK-MODI, SETCOLOR, COLOR

GRAPHIK-Modus	COLOR	SETCOLOR Register #	Bemerkungen
0	nicht verwendet	0	- nicht verwendet
		1	Zeichenhelligkeit
		2	Hintergrund
		3	- nicht verwendet
		4	Rand
1 und 2	nicht verwendet	0	Zeichen
		1	Zeichen (Textfenster Zeichenhelligkeit)
		2	Zeichen (Textfenster Hintergrund)
		3	Zeichen
		4	Hintergrund, Rand
3, 5, und 7	1	0	Graphikpunkt
	2	1	Graphikpunkt (Textf. Zeichenhelligkeit)
	3	2	Graphikpunkt (Textf. Hintergrund)
	-	3	- nicht verwendet
	0	4	Graphikpunkt (Hintergrund, Rand)
4 und 6	1	0	Graphikpunkt
	-	1	(Textfenster Zeichenhelligkeit)
	-	2	(Textf. Hintergrund)
	-	3	nicht verwendet
	0	4	Graphikpunkt (Hintergrund, Rand)
8	-	0	- nicht verwendet
	1	1	Graphikpunkt Helligkeit
	0	2	Graphikpunkt (Hintergrund)
	-	3	- nicht verwendet
	-	4	Rand

Mit Genehmigung der ATARI, Inc. 1980 aus dem BASIC REFERENCE MANUAL übernommen.

Aber warum war das Haus zu Beginn orange? Nun, das Farbregister 0 enthält normalerweise nach dem Einschalten des Computers die Farbe orange. Jede Farbregister enthält nach dem Einschalten des Computers einen Ursprungswert. Diese Farben sind in der Tabelle 2-2 aufgeführt.

TABELLE 2-2. URSPRUNGSWERTE DER FARBREGISTER

SETCOLOR Register #	Ursprungswerte für Farbwert	Helligkeit	Aktuelle Farbe
0	2	8	Orange
1	12	10	Grün
2	9	4	Dunkelblau
3	4	6	Rosa
4	0	0	Schwarz

Kommen wir noch einmal zurück zu unserem Haus Programm um zu sehen, ob Sie alles verstanden haben. Ändern Sie Zeile 220 wie folgt:

```
220 COLOR 2
```

Starten Sie das Programm. Das Haus ist grün, da dies die Ursprungsfarbe ist, wenn wir mit COLOR 2 im GRAPHIK-Modus 5 zeichnen.
Um jetzt die Hausfarbe zu ändern, müssen wir Farbregister 1 verwenden. Schlagen Sie in Tabelle 2-1 nach. Ändern Sie Zeile 230 wie folgt:

```
230 SETCOLOR 1,0,14
```

Starten Sie das Programm. Ihr Haus sollte jetzt weiß sein.

Wenn wir Zeile 220 in 220 COLOR 3 ändern, welche Farbe hat dann das Haus?

Antwort #4 _____

#4. Dunkelblau

Welche SETCOLOR-Anweisung muß verwendet werden, um das Haus weiß zu machen, wenn wir mit COLOR 3 gezeichnet haben?

Antwort #5 _____

Lassen Sie uns zwei verschiedene Farben verwenden, um den unteren Teil des Hauses in einer anderen Farbe darzustellen, als den oberen. Ändern Sie Ihr Programm entsprechend unserem folgenden Beispiel:

```
100 REM EIN KLEINES HAUS VON DIRK AM 25 MAI 1984
200 REM ERSTELLE GRAPHIK BILDSCHIRM
210 GRAPHICS 5
300 REM ZEICHNE DAS DACH DES HAUSES
305 COLOR 1
310 PLOT 40,5
320 DRAWTO 50,15
330 DRAWTO 30,15
340 DRAWTO 40,5
400 REM ZEICHNE DEN UNTEREN TEIL DES HAUSES
405 COLOR 2
410 PLOT 30,15
420 DRAWTO 50,15
430 DRAWTO 50,25
440 DRAWTO 30,25
450 DRAWTO 30,15
500 END
```

Starten Sie das Programm. Das Hausdach sollte orange sein (Ursprungswert des Registers 0), und der untere Teil des Hauses sollte grün sein (Ursprungswert des Registers 1). Die COLOR 1 Anweisung gilt für die Zeilen 310 bis 340. Die COLOR 2 Anweisung hat in den Zeilen 410 bis 450 Gültigkeit.
Lassen Sie uns ein anderes Farbregister ändern. Fügen Sie die folgende Zeile zu unserem Programm hinzu:

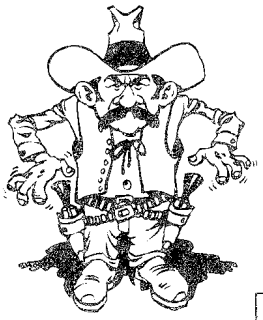
```
240 SETCOLOR 4,4,4
```

Können Sie vorhersagen, was passieren wird? Starten Sie das Programm, und Sie werden sehen, ob Sie richtig getippt haben.

#5. SETCOLOR 2,0,14

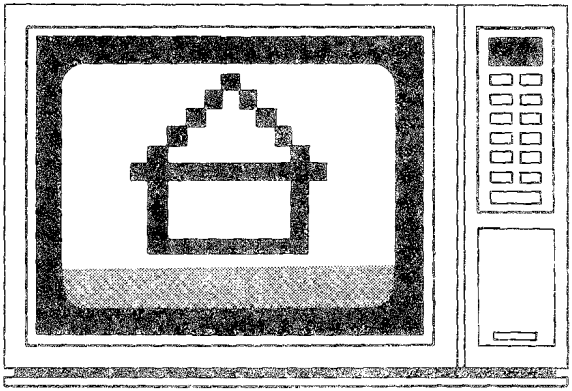
Welche Anweisung würden Sie verwenden, um die Hintergrundfarbe in blau zu ändern?

Antwort #6 _____



Aufgabe #4

Zeichnen Sie ein Haus im GRAPHIK-Modus 7. Es sollte so ähnlich aussehen, wie auf unserem Bild. Die Größe können Sie selbst bestimmen. Machen Sie den Hintergrund schwarz, das Dach rot, und den unteren Teil blau.



Farbflächen

Es gibt verschiedene Wege, wie man Flächen auf dem Bildschirm mit einer Farbe ausfüllen kann. Ein Weg ist, kontinuierlich Linien zu zeichnen, bis die gewünschte Fläche ausgefüllt ist. Wir können das anhand eines Programmbeispiels ausprobieren. Löschen Sie alle alten Programme aus dem Speicher des Computers und geben Sie die folgenden Programmzeilen ein.

#6. SETCOLOR 4,8,4

```
100 REM FARBFLAECHEN PROGRAMM
200 REM ERSTELLE GRAPHIK BILDSCHIRM
210 GRAPHICS 7
220 COLOR 1
300 REM ZEICHNE DREI LINIEN
310 PLOT 10,20
320 DRAWTO 150,20
330 PLOT 10,21
340 DRAWTO 150,21
350 PLOT 10,22
360 DRAWTO 150,22
400 END
```

Starten Sie es. Das scheint doch ein wenig viel Aufwand zu sein, nur um drei Linien auf den Bildschirm zu zeichnen?.
Verwenden wir doch lieber eine FOR...TO, NEXT Schleife. Wenn Sie im Umgang mit Schleifen keine Erfahrung haben, so lesen Sie bitte im ATARI BASIC Reference Manual den entsprechenden Abschnitt durch. Ändern Sie unser Programm so, daß es wie folgt aussieht:

```
100 REM FARBFLAECHEN PROGRAMM
200 REM ERSTELLE GRAPHIK BILDSCHIRM
210 GRAPHICS 7
220 COLOR 1
300 REM ZEICHNE DREI LINIEN
310 FOR X=20 TO 22
320 PLOT 10,X
330 DRAWTO 150,X
340 NEXT X
400 END
```

Starten Sie es. Das Ergebnis sollte dasselbe sein, wie zuvor.

In diesem Programm wird eine Variable X verwendet, in der als erster Wert die Zahl 20 abgelegt wird. Sobald die Anweisung NEXT X erreicht wird, verzweigt das Programm zurück zur FOR-Anweisung und der Wert von X wird um 1 auf 21 erhöht. Dies wird solange fortgesetzt, bis X den Wert 22 erreicht. Dann endet die Schleife. Beachten Sie, daß der Wert von X auch in den PLOT- und DRAWTO-Anweisungen verwendet wird.
Auf diese Weise ist es sehr einfach viele Linien zu zeichnen. Ändern Sie Zeile 310 in:

```
310 FOR X=20 TO 60
```

Starten Sie das Programm. Es werden 41 Linien gezeichnet. Zu einem späteren Zeitpunkt werden wir noch eine andere Möglichkeit zum Füllen von Flächen mit Farbe besprechen.

Graphik über den gesamten Bildschirm

Jetzt ist die Zeit gekommen, daß wir uns endlich des Textfensters entledigen, um den gesamten Bildschirm für GRAPHIK zur Verfügung zu haben. Dies kann ganz einfach durch Addieren der Zahl 16 zur entsprechenden GRAPHIK-Modus Nummer durchgeführt werden.
Verwenden wir noch einmal unser letztes Programm. Zeile 210 könnte wie folgt geändert werden:

```
210 GRAPHICS 7+16
```

oder

```
210 GRAPHICS 23
```

Ändern Sie die Zeile und starten Sie das Programm. Oh! Die Zeichnung verschwand direkt, nachdem Sie fertig war.
Das liegt daran, daß Programme, wenn sie fertig bearbeitet sind, wieder den GRAPHIK-Modus 0 aufrufen. Ein Weg, um dies zu verhindern ist, das Programm in eine Endlosschleife zu schicken. Wenn wir Zeile 390 hinzufügen, haben wir eine solche Schleife.

```
390 GOTO 390
```

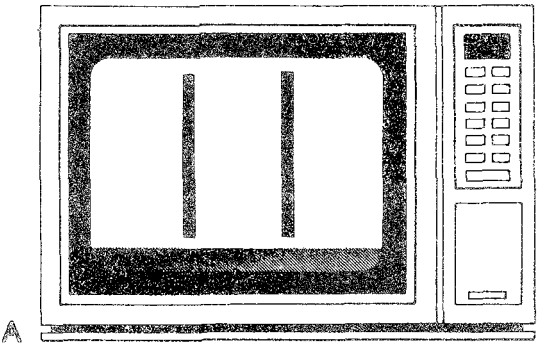
Wenn Zeile 390 erreicht wird, wird der Computer angewiesen immer wieder zu Zeile 390 zu gehen (obwohl er da ja schon die ganze Zeit ist). Starten Sie das Programm. Und tatsächlich, wir haben einen Bildschirm ohne Textfenster. Wenn Sie das Programm beenden wollen, können Sie die <BREAK>-Taste oder die <SYSTEM RESET> Taste drücken.
Und da wären wir auch schon am Ende des zweiten Kapitels angelangt. Wir haben eine Menge neuen Stoff gelernt. Um zu sehen, ob Sie auch alles verstanden haben, wollen wir den Inhalt des Kapitels 2 noch einmal zusammenfassen.

Zusammenfassung von Kapitel 2

In diesem Kapitel haben Sie mit den GRAPHIK-Modi 3 bis 8 experimentiert. Hier folgen jetzt einige Fragen, um zu sehen, wie gut Sie den erlernten Stoff verstanden haben.

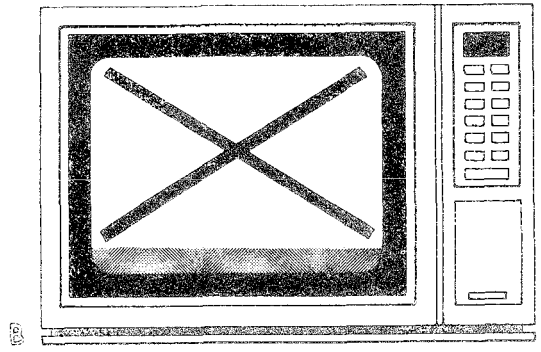
ZEICHNUNGEN - Vergleichen Sie jede der Zeichnungen mit dem richtigen Programm.

1.
```
10 GRAPHICS 7
20 COLOR 1
30 PLOT 0,0
40 DRAWTO 159,79
50 PLOT 0,79
60 DRAWTO 159,0
```



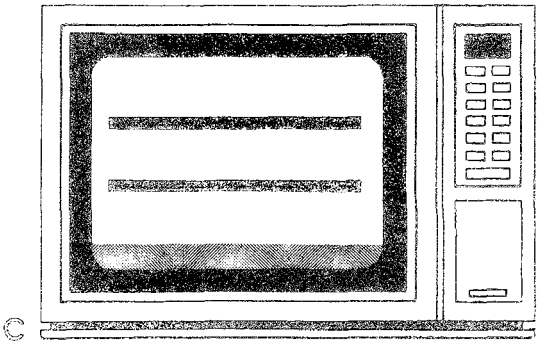
A

2.
```
10 GRAPHICS 7
20 COLOR 1
30 PLOT 53,0
40 DRAWTO 53,79
50 PLOT 106,0
60 DRAWTO 106,79
```



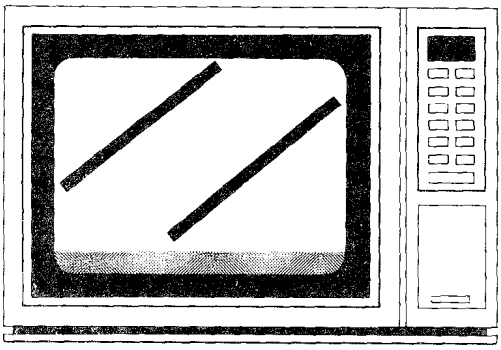
B

3.
```
10 GRAPHICS 7
20 COLOR 1
30 PLOT 0,39
40 DRAWTO 79,79
50 PLOT 79,0
60 DRAWTO 159,39
```



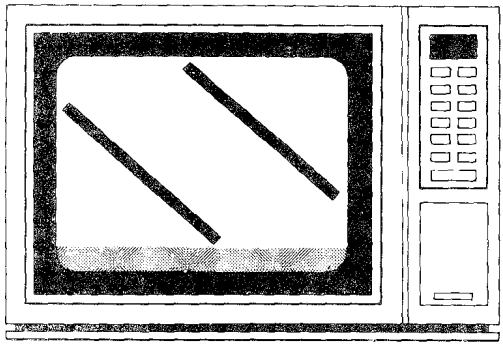
C

4.
```
10 GRAPHICS 7
20 COLOR 1
30 PLOT 79,0
40 DRAWTO 0,39
50 PLOT 159,39
60 DRAWTO 79,79
```



D

5.
```
10 GRAPHICS 7
20 COLOR 1
30 PLOT 0,27
40 DRAWTO 159,27
50 PLOT 0,54
60 DRAWTO 159,54
```



E

FARBEN - Geben Sie für jede der aufgeführten Situationen an, welches Farbregister verwendet werden muß.

	GRAPHIK-Modus	zu setzende Farbkondition	Farbregister (SETCOLOR)
6.	Modus 5	Hintergrundfarbe	_____
7.	Modus 0	Randfarbe	_____
8.	Modus 4	COLOR 1 Zeichnungen	_____
9.	Modus 1	Hintergrundfarbe	_____
10.	Modus 7	COLOR 3 Zeichnungen	_____
11.	Modus 8	Hintergrundfarbe	_____
12.	Modus 3	COLOR 0 Zeichnungen	_____

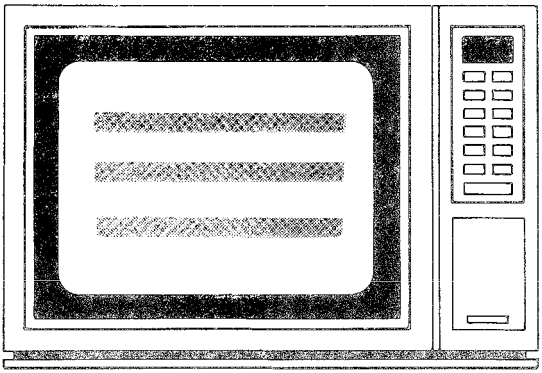
Das folgende Programm wird verwendet, um drei Linien auf den Bildschirm zu zeichen, genau wie auf unserem Bild. Welche Farbe hat welche Zeile?

```
100 REM ZEICHNE DREI FARBIGE LINIEN
200 REM ERSTELLE GRAPHIK BILDSCHIRM
210 GRAPHICS 3
300 REM ZEICHNE ERSTE LINIE
310 COLOR 1
320 PLOT 5,5
330 DRAWTO 35,5
400 REM ZEICHNE ZWEITE LINIE
410 SETCOLOR 1,8,4
420 COLOR 2
430 PLOT 5,10
440 DRAWTO 35,10
500 REM ZEICHNE DRITTE LINIE
510 SETCOLOR 2,4,10
520 COLOR 3
530 PLOT 5,15
540 DRAWTO 35,15
600 END
```

Erste Linie

Zweite Linie

Dritte Linie



13. Farbe der ersten Linie _____

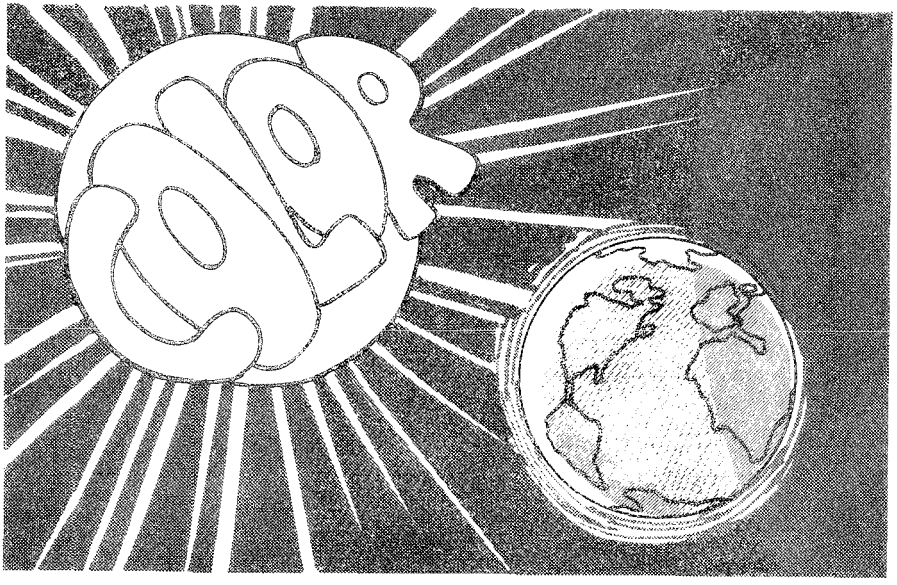
14. Farbe der zweiten Linie _____

15. Farbe der dritten Linie _____

KAPITEL 3

Farben und Kontraste

Farbe! Sie ist der Sonnenschein unserer Welt.



In Kapitel 2 lernten wir die auf dem Bildschirm dargestellten Farben zu kontrollieren. Aber farbige Videographiken sind viel mehr, als eine bloße Darstellung von Farben. Gerade so wie der Künstler in der Lage ist sich auf einer Leinwand aus zudrücken, muß der Video Komponist in der Lage sein, seine Gefühle mit den Farben des Bildschirms darzulegen.
In diesem Kapitel werden wir noch mehr über die GRAPHIK-Modi erfahren. Hauptsächlich werden wir uns jedoch mit den Farben und ihrer Beziehung zu Fernsehgraphiken beschäftigen.

In diesem Kapitel werden Sie lernen, wie Sie:

- Farben mit den passenden Kontrasten wählen
- Zeichnungen mit unterschiedlichen Kontrasten erstellen
- die GRAPHIK-Modi 9, 10 und 11 verwenden

Zuerst brauchen wir jedoch einmal ein grundsätzliches Verständnis für Farben.
Die Farben, die wir mit unserem Auge wahrnehmen, sind das direkte Resultat des Lichts, das uns ständig umgibt. Ohne Licht (dunkler Raum) erscheint alles schwarz. Bei Licht erkennen wir eine Vielzahl von Farben. Licht (besonders weißes Licht) ist eine Kombination aller Lichtfarben. Ein Objekt reflektiert immer nur einen Teil dieses Spektrums. Diesen Teil erkennen wir dann als Farbe. So erscheint uns der rote Wagen an einem sonnigen Tag nur deshalb so rot, da die Lackierung des Wagens nur die roten Lichtstrahlen reflektiert. Die anderen Komponenten des weißen Lichts (gelb, blau, grün) werden absorbiert.
In diesem Kapitel werden wir mit den Farben arbeiten, als hätten wir Pinsel und Farbe zur Hand. Der einzige Unterschied besteht darin, daß wir unseren Bildschirm anmalen und keine Leinwand. Bei unseren Experimenten werden wir den Unterschied zwischen primären und sekundären Farben studieren. In einem kurzen Exkurs erfahren Sie alles Wissenswerte über diese Farben. Obwohl wir uns dabei hauptsächlich auf die Pigmente konzentrieren, können die Informationen auch für Fernsehgraphiken nützlich sein. Dieser Exkurs hat jedoch nichts mit den natürlichen Lichtfarben und ihrer Verwendung zu tun. Die primären und sekundären Farben werden im Bild 3-1 dargestellt.
Das Ergebnis zeigt das Resultat, wenn verschiedene Farben miteinander gemischt werden. So entseht aus der Mixtur der primären Farben gelb und blau die sekundäre Farbe grün. Eine sekundäre Farbe ist immer das Ergebnis der Vermischung mindestens zweier primärer Farben.

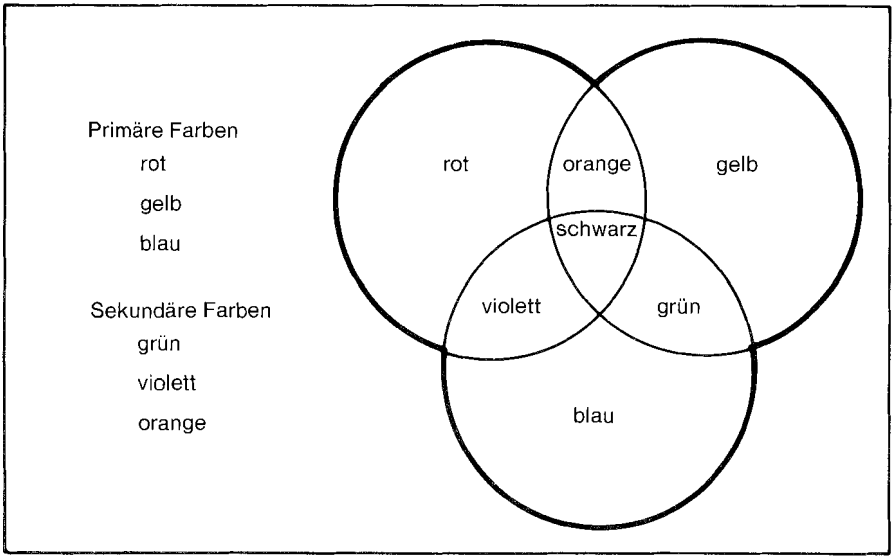


BILD 3-1: Primäre und sekundäre Farben.

Schwarz ensteht aus der Kombination aller drei primären oder auch sekundären Farben.

Kontraste

Es gibt viele Arten von Kontrasten oder Vergleichen: jung-alt, groß-klein, schwer-leicht, usw. Auch Farben können kontrastieren. Sie können hell oder dunkel sein. Aber Farben können auch dazu verwendet werden, um Beziehungen und Gefühle auszudrücken, wie z. B. Kalt-warm, nah-fern und naß-trocken. Ein paar Farben verblaßen, wenn sie in Kontrast gesetzt werden, andere wiederum erscheinen klar und deutlich. Mit Farbkontrasten lassen sich wirklich alle Gefühle ausdrücken.
Sind Sie bereit ein paar Farbkontraste zu probieren? Lassen Sie uns ein Programm zur Darstellung von Farben schreiben, welches wir in unseren Experimenten verwenden wollen.

Wir werden dieses Programm während des ganzen Kapitels verwenden, um daran Farbkontraste zu testen. Geben Sie ein:

```
100 REM DARSTELLUNG VON FARBKONTRASTEN
200 REM SETZE GRAPHIKBILDSCHIRM
210 GRAPHICS 3+16
220 SETCOLOR 0,4,4
230 SETCOLOR 1.1.14
240 SETCOLOR 2,8,4
250 SETCOLOR 4,0,14
300 REM ZEICHNE DREI BLOECKE IN DEN FARBEN 1,2,3
310 FOR C=0 TO 2
320 COLOR C+1
330 REM ZEICHNE 7 HORIZONTALE LINIEN
340 FOR X=1 TO 7
350 PLOT 2,C*7+X
360 DRAWTO 38,C*7+X
370 NEXT X
380 NEXT C
390 GOTO 390
400 END
```

Dieses Programm wird drei horizontale Farbblöcke auf den Bildschirm zeichnen. Jeder Block enthält sieben GRAPHIK-Modus 3 Zeilen. Jede der SETCOLOR-Anweisungen der Zeilen 220 bis 250 legt eine bestimmte Farbe fest. Wir werden diese drei Zeilen verändern, um mit den Farben zu experimentieren.

Werfen Sie einen Blick auf die Tabelle 1-1 und versuchen Sie vorherzusagen welche Farben dargestellt werden. Schreiben Sie die Farben in die Leerzeilen.

SETCOLOR 0,4,4 Antwort #1_____

SETCOLOR 1,1,14 Antwort #2_____

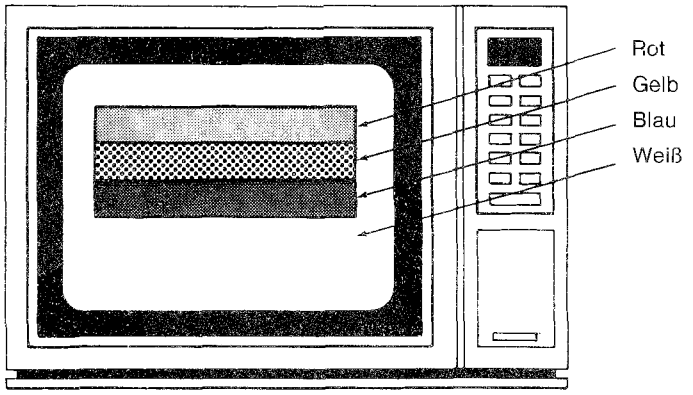
SETCOLOR 2,8,4 Antwort #3_____

SETCOLOR 4,0,14 Antwort #4_____

Beachten Sie bitte, daß das Farbregister 3 im GRAPHIK-Modus 3 nicht verwendet wird.

#1. Rot #2. Gelb #3. Blau #4. Weiß

Starten Sie das Programm. Die Bildschirmdarstellung sollte ungefähr folgendermaßen ausehen:



Wenn die Farben auf Ihrem Fernseher nicht den hier aufgeführten Farben entsprechen, haben Sie jetzt die ideale Gelegenheit Ihren Fernseher einzustellen.

Farbkontrast

Die primären Farben rot, gelb und blau kontrastieren am stärksten. Die Kontraste werden verstärkt, wenn die Farben durch eine schwarze oder weiße Linie getrennt werden.
Versuchen wir's. Verändern Sie Zeile 340 in

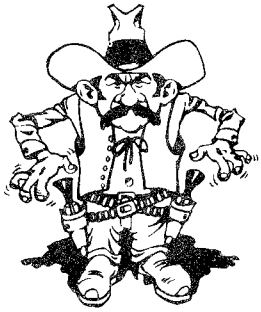
```
340 FOR X=1 TO 6
```

Starten Sie es. Sehen Sie einen Unterschied im Kontrast?
Machen Sie den Hintergrund schwarz. Ändern Sie Zeile 250 in

```
SETCOLOR 4,0,0
```

Starten Sie es. Können Sie jetzt einen Unterschied erkennen?

Jetzt kommt's, Aufgabe #5.



Aufgabe #5

Verändern Sie die SETCOLOR Werte des Programms dahingehend, daß die sekundären Farben produziert werden.

Hell-Dunkel Kontrast

Hell und dunkel sind die grundlegensten Kontraste. Den stärksten Kontrast bilden dabei die Farben weiß und schwarz. Hell-dunkel Kontraste sind allerdings relativ. Grau kann verglichen mit schwarz als hell erscheinen, verglichen mit weiß aber als dunkel. Selbst Farben mit der gleichen Helligkeit können hell-dunkel Kontraste haben. Die Farben gelb und violett erzeugen den stärksten hell-dunkel Kontrast. Lassen Sie uns die SETCOLOR-Anweisungen des Farbdarstellungsprogramms so verändern, daß die Farben gelb und violett auf den Bildschirm gebracht werden. Versuchen Sie die folgenden Änderungen.

```
220 SETCOLOR 0,6,14
240 SETCOLOR 2,6,14
```

Ihr Programm sollte jetzt folgendermaßen aussehen:

```
100 REM DARSTELLUNG VON FARBKONTRASTEN
200 REM SETZE GRAPHIKBILDSCHIRM
210 GRAPHICS 3+16
220 SETCOLOR 0,6,14
230 SETCOLOR 1,1,14
240 SETCOLOR 2,6,14
250 SETCOLOR 4,0,0
300 REM ZEICHNE 3 BLÖCKE IN DEN FARBEN 1,2,3
310 FOR C=0 TO 2
320 COLOR C+1
330 REM ZEICHNE 7 HORIZONTALE LINIEN
340 FOR X=1 TO 6
```

```
350 PLOT 2,C*7+X
360 DRAWTO 38,C*7+X
370 NEXT X
380 NEXT C
390 GOTO 390
400 END
```

Starten Sie es. Beachten Sie den Kontrast zwischen den Farben mit gleicher Helligkeit.

Kalt-Warm Kontrast

Farbkontraste können dazu benutzt werden, um ein Gefühl von Kälte und Wärme zu vermitteln. Dieselben Kontraste können auch Gefühle für andere Verbindungen hervorrufen. Einige davon sind hier aufgezählt.

kalt-warm
schattig-sonnig
durchsichtig-massiv
beruhigend-anregend
öde-strotzend
Luft-Boden
nah-fern
leicht-schwer
naß-trocken

Diese Verbindungen werden uns in Kapitel 6 nützlich, wenn wir an der Gestaltung des Bildschirms arbeiten.
Die Farben, die die stärksten kalt-warm Kontraste vermitteln, sind blau-grün--rot-orange. Alle anderen Farben liegen dazwischen.
Wir wollen unser Programm so ändern, daß es einen kalt-warm Effekt erzeugt. Ändern Sie diese Zeilen:

```
220 SETCOLOR 0,3,8
230 SETCOLOR 1,6,8
240 SETCOLOR 2,11,8
340 FOR X=1 TO 7
```

Starten Sie das Programm. Beachten Sie wie warm violett gegen blau-grün wirkt, aber wie kalt gegen rot-orange. Das blau-grün wirkt fern, das rot-orange nah.

Stellen Sie sich vor Sie würden die Farben braun und grün darstellen. Welche dieser Farben erweckt bei einem Kontrast schwer-leicht, den schwereren Eindruck?

Antwort #5_____

Versuchen wir eine andere Frage. Stellen Sie sich vor Sie wollen Näße und Trockenheit darstellen. Welche Farben würden Sie wählen?

Antwort #6_____ - _____

Komplementär-Kontrast

Zwei Farben sind ergänzend (komplementär), wenn ihre Pigmente bei einer Vermischung ein neutrales grau-schwarz erzeugen. Vergleichen Sie mit Bild 3-1. Sie werden feststellen, daß überall dort wo eine primäre Farbe mit einer gegensätzlichen sekundären Farbe gemischt wird, schwarz entsteht. Solche Paare nennt man komplementär. Beispiele hierfür sind:

gelb, violett

blau, orange

rot, grün

Gelb/violett stellen nicht nur einen komplementären-sondern auch einen extremen hell-dunkel Kontrast dar. Dies wurde durch das letzte Programm dokumentiert.
Lassen Sie uns einmal sehen, wie das komplementäre Paar blau-orange aussieht. Ändern Sie die Zeilen 220, 230 und 240 unseres Farbdarstellungsprogramms.

(Antworten können variieren) #5. Braun #6. Blau-Orange

```
220 SETCOLOR 0,8,6
230 SETCOLOR 1,2,6
240 SETCOLOR 2,8,6
```

Starten Sie das Programm. Sehen Sie diesen Kontrast?

Gleichzeitiger Kontrast

Ist das menschliche Auge für längere Zeit einer Farbe ausgesetzt, wird es blind für diese Farbe. Wenn Sie zum Beispiel für längere Zeit die Farbe gelb ansehen, reagieren Ihre Augen nicht mehr so stark auf gelb wie anfangs. Sie würden gelb nicht mehr als so grell empfinden.
Sollten Sie dann plötzlich auf ein neutrales grau sehen, reagieren Ihre Augen stärker auf die Komplementärfarbe zu gelb (violett). Sie hätten das Gefühl die graue Farbe hätte einen violetten Stich. Da Ihre Augen sich an das gelb gewöhnt haben, sehen Sie weniger davon, dafür aber mehr von den anderen Farben des Spektrums , die von grau ausgestrahlt werden. Diese Farben ergeben zusammen violett. Vergleichen Sie mit Bild 3-1.
Glauben Sie es? Machen wir einen Test. Ändern Sie diese Zeilen:

```
220 SETCOLOR 0,1,14
230 SETCOLOR 1,0,10
240 SETCOLOR 2,1,14
250 SETCOLOR 4,1,14
```

Starten Sie das Programm. Blicken Sie für ungefähr 30 Sekunden auf die gelbe Fläche des Bildschirms. Während Sie weiterhin auf den Bildschirm sehen, können Sie feststellen, daß das grau langsam violette Töne annimmt. Sehen heißt glauben!

In welche Farbe würde sich grau verwandeln, wenn der Hintergrund blau wäre?

Antwort #7_____

Versuchen Sie es mit einem blauen, orangem oder grünem Hintergrund.

#7. Orange

Verbindung zwischen Größen- und Warm-Kalt Kontrast

Um z. B. ein Bild zu erzeugen, das weder das Gefühl von besonderer Kälte noch das von Wärme vermittelt, müssen neben dem Warm-Kalt Kontrast auch die zu verwendenden Flächengrößen der jeweiligen Farben beachtet werden.
Damit diese Ausgewogenheit entstehen kann , machen wir die Flächengröße von der Farbe abhängig. Warme Töne wirken kräftiger als kalte Farben. Um eine ausgeglichene Wirkung zu erzielen, benötigen wir daher mehr Fläche mit kalter Farbe, als mit warmen Tönen.
Den Eindruck von Wärme, den die einzelnen Farben erzeugen, kann man in Zahlenwerten ausdrücken. z. B.:

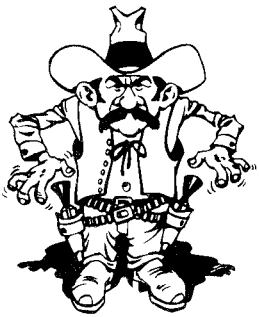
gelb-9
orange-8
rot-6
violett-3
blau-4
grün-6

Ein ausgewogener Bildschirm mit den Farben gelb und violett, hätte demnach dreimal soviel violette Fläche als gelbe. Es liegt dann an dem "Künstler", seine Gefühle und Empfindungen auszudrücken, indem er unausgewogene Bilder entwirft. Wenn der Bildschirm zum Beispiel Wärme ausdrücken soll, dann wird ein Übergewicht an warmen Farben gewählt werden. Das bedeutet aber trotzdem nicht, daß diese warmen Farben dann eine größere Fläche einnehmen müssen, als die kalten.
Versuchen Sie ein Beispiel. Ändern Sie diese Zeilen:

```
220 SETCOLOR 0,8,4
230 SETCOLOR 1,8,4
240 SETCOLOR 2,2,8
250 SETCOLOR 4,2,8
```

Starten Sie das Programm. Wie Sie sehen füllen blau und orange in etwa die selbe Fläche aus, wodurch ein Gefühl der Wärme vermittelt wird.

Gut, es ist Zeit für Aufgabe #6.



Aufgabe #6

Dies ist eine zweifache Aufgabe. Setzen Sie die Farben in unserem Farbdarstellungsprogramm, so daß:

1. mit den Farben rot und violett ein Gefühl der Wärme erzeugt wird. Machen Sie den Hintergrund schwarz. (Rot kann eine Färbung und eine Helligkeit von 4 haben; violett eine Färbung von 6 und eine Helligkeit von 4.)
2. mit den Farben rot und violett ein Gefühl der Kälte erzeugt wird. Gestalten Sie dabei den Hintergrund entweder in violett oder rot.

TIP: Sie werden sich mit den von den Farben beanspruchten Flächen beschäftigen müssen.

Farbige GRAPHIK-Modi 9, 10, 11

Die graphischen Fähigkeiten des ATARI Computers hängen von einem speziellen Videodarstellungschip innerhalb des Computers ab. Die ersten Ausführungen der ATARI Computer haben ein Chip namens CTIA verwendet. Die neueren aktuellen Maschinen verwenden ein Chip namens GTIA. Das Originalchip arbeitete mit den BASIC GRAPHIK-Modi 0 bis 8. Das neue Chip bietet zusätzlich die Modi 9, 10 und 11. Mit diesen zusätzlichen Modi können noch mehr Farben erzeugt werden.
Die nun folgenden Programme funktionieren nur, wenn Ihr Computer über ein GTIA Chip verfügt. Wenn Sie nicht wissen, ob Ihr Computer ein solches Chip eingebaut hat, probieren Sie es einfach aus.

Modus 11

Beginnen wir mit einem Blick auf die 16 Farben des GRAPHIK-Modus 11. Hier ist ein Testprogramm. Vergessen Sie nicht die älteren Programme aus dem Computer zu löschen.

```
100 REM 16 FARBEN
200 REM SETZE GRAPHIKBIKDSCHIRM
210 GRAPHICS 11
220 SETCOLOR 4,0,6
300 REM ZEICHNE 16 VERTIKALE LINIEN
310 FOR C=0 TO 15
320 COLOR C
330 PLOT 5*C,10
340 DRAWTO 5*C,150
350 NEXT C
390 GOTO 390
400 END
```

Lassen Sie das Programm laufen. Die 16 Farben (Färbungen) sind dieselben, die bereits in der Tabelle 1-1 aufgeführt wurden. Der Hintergrund ist schwarz und die Helligkeit der Farben wurde in Zeile 220 festgelegt (hier der Wert 6). Der Wert der COLOR-Anweisung in Zeile 320 reicht von 0 bis 15.
Sie können auch sehen, daß die Linien vertikal gezeichnet wurden. Die horizontale Auflösung des GRAPHIK-Modus 11 ist dieselbe, wie in GRAPHIK-Modus 5 (80 Spalten), aber die vertikale Auflösung ist die des Modus 8 (192 Reihen). Jedes Pixel ist damit viermal so breit wie hoch.

Modus 9

Modus 9 ähnelt in seiner Anwendung dem Modus 11. Der Unterschied besteht darin, daß anstatt 16 verschiedener Farben, 16 verschiedene Helligkeiten dargestellt werden können.
Wir wollen das vorige Programm derart verändern, daß nun 16 Helligkeiten dargestellt werden. Die Zeilen 100, 210 und 220 werden geändert, damit Ihr Programm folgendermaßen aussieht:

```
100 REM 16 HELLIGKEITEN
200 REM SETZE GRAPHIKBILDSCHIRM
```

```
210 GRAPHICS 9
220 SETCOLOR 4,8,0
300 REM ZEICHNE 16 VERTIKALE LINIEN
310 FOR C=0 TI 15
320 COLOR C
330 PLOT 5*C,10
340 DRAWTO 5*C,150
350 NEXT C
390 GOTO 390
400 END
```

Starten Sie es. Sie können feststellen, daß die Zeilenfarben andere Helligkeiten haben, als der Hintergrund. Die Hintergrundfarbe wurde in diesem Beispiel in Zeile 220 (zu 8-blau) festgelegt.

Modus 10

Wir haben uns Modus 10 für den Schluß aufgehoben. Mit diesem Modus können 9 verschiedene Farben in 16 unterschiedlichen Helligkeiten dargestellt werden. Wir behandeln ihn erst jetzt am Ende, da er sich in der Programmierung unterscheidet.
Bis jetzt haben wir die fünf Farbregister 0 bis 4 zusammen mit der SETCOLOR-Anweisung verwendet. In Wahrheit verfügt der ATARI Computer aber über neun Farbregister. Die verbleibenden vier Farbregister können jedoch nicht zusammen mit dem SETCOLOR-Befehl verwendet werden. Nur mit der POKE-Anweisung können sie Farben zuordnen. POKE ist eine allgemeine BASIC-Anweisung, die es auch dem Anfänger ermöglichen soll Register, die er auf dem üblichen Programmierweg noch nicht erreichen kann, zu verändern. Ein Beispiel ist der Befehl POKE 704,70. Der Computer wird angewiesen die Farbregister Position 704 mit der Farbe 70 zu füllen.
Die vier zusätzlichen Farbregister Positionen, die nicht mit der SETCOLOR-Anweisung angsteuert werden können, sind 704, 705, 706 und 707. Die fünf mit der SETCOLOR-Anweisung zu verwendenden Register sind 708 bis 712. Farbwerte können auch in diese Positionen gePOKEt werden.
Die Farbwerte können von 0 bis 255 reichen. Um einen Farbwert zu berechnen, nehmen Sie die gewünschte Farbzahl aus der Tabelle 1-1, und multiplizieren sie mit 16. Dazu müssen Sie noch den gewünschten Helligkeitswert addieren.

So zum Beispiel rosa mit einem Farbwert von 4 und einer Helligkeit von 6, hätte den Wert 70.

```
(4 X 16) + 6 = 70
```

Sehen wir uns ein Programmbeispiel an.

```
100 REM 9 FARBEN UND 16 HELLIGKEITEN
200 REM SETZE GRAPHIKBILDSCHIRM
210 GRAPHICS 10
220 FOR C=704 TO 712
230 POKE C,RND(0)*256
240 NEXT C
300 REM ZEICHNE 9 VERTIKALE LINIEN
310 FOR C=0 TO 8
320 COLOR C
330 PLOT 5*C,10
340 DRAWTO 5*C,150
350 NEXT C
390 GOTO 390
400 END
```

Geben Sie diese Zeilen ein und starten Sie das Programm. In diesem Beispiel wurden zufällige Farbwerte (Zeile 230-RND) in die Farbregister gePOKEt. Da die Farben zufällig gewählt werden, bietet der Bildschirm bei jedem neuen Programmdurchlauf einen neuen Anblick. Sehen Sie in einem BASIC Programmierlehrbuch für weitere Details über die BASIC RND Anweisung nach. In diesem Modus steuern wir die neun Farbregister an, indem wir für die COLOR-Anweisung Werte wählen, die von 0 bis 8 reichen.

Zusammenfassung von Kapitel 3

Beantworten Sie die folgenden Fragen.

1. Welche Farben besitzen den größten Kontrast in der Färbung: rot-blau oder grün-blau?

2. Im Vergleich zu rosa. Welche Farben erscheinen wärmer: blau, gelb, violett, weiß oder orange?

3. Mit welchen Farben läßt sich am besten ein Gefühl der Distanz erzeugen, blau oder gelb?

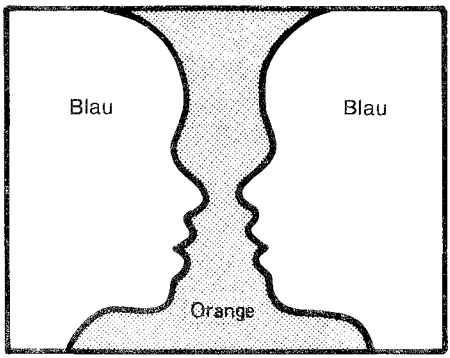
4. Ein Bildschirm ist zu gleichen Teilen blau und orange. Vermittelt der Bildschirm ein Gefühl der Wärme oder der Kälte?

5. Welcher der Modi 9,10 oder 11 sollte verwendet werden, um möglichst viele Variationen einer Farbe zu erzeugen?

6. Welchen Farbwert würden Sie verwenden, um eine orange Färbung mit der Helligkeit 6 zu produzieren?

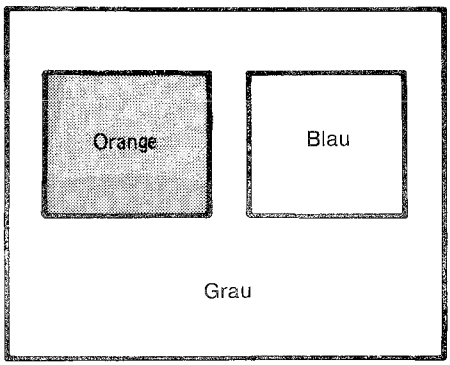
7. A. Welche der beiden Farben erscheint in größerer Entfernung?

   B. Was stellt dieses Bild eher dar? Zwei Gesichter oder eine Vase?



8. A. Würde dieser Bildschirm Wärme oder Kälte vermitteln?

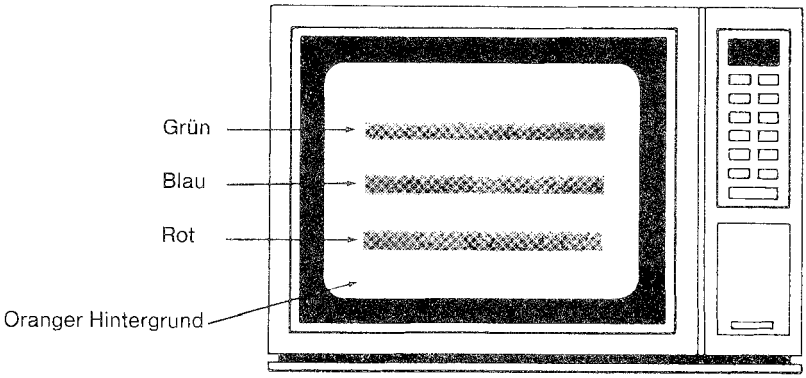
   B. Welches der beiden Quadrate (blau oder orange) macht den gewichtigeren Eindruck?



Mit dem folgenden Programm werden drei Linien auf dem Bildschirm gezeichnet:

```
100 REM ZEICHNE DREI LINIEN
200 REM SETZE GRAPHIKBILDSCHIRM
210 GRAPHICS 3+16
220 SETCOLOR 4,2,0
```

```
300 REM ZEICHNE ERSTE LINIE
305 SETCOLOR 0,13,12
310 COLOR 1
320 PLOT 5,5
330 DRAWTO 35,5
400 REM ZEICHNE ZWEITE LINIE
410 SETCOLOR 1,8,0
420 COLOR 2
430 PLOT 5,10
440 DRAWTO 35,10
500 REM ZEICHNE DRITTE LINIE
510 SETCOLOR
520 COLOR 3
530 PLOT 5,15
540 DRAWTO 35,15
600 GOTO 600
```



9. Welche beiden Farben haben den stärksten hell-dunkel Kontrast?

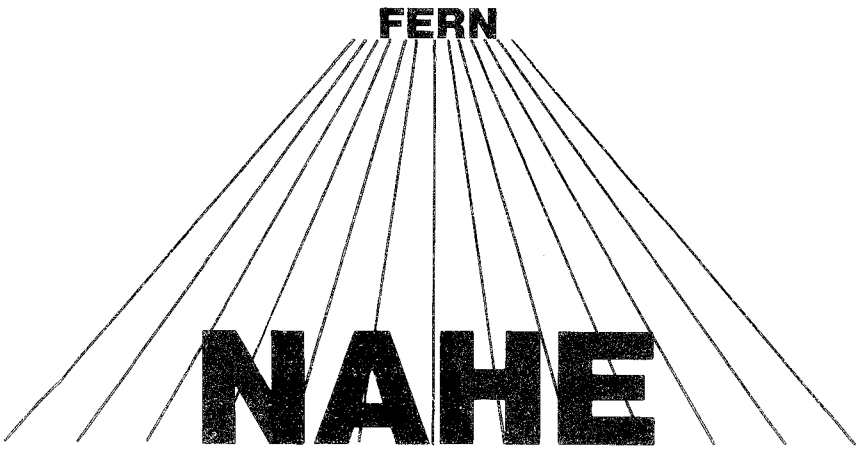
10. Welche beiden Farben haben den stärksten Farbkontrast?

11. Welche Farbe ist die wärmste?

KAPITEL 4

Drei Dimensionen

Ist es nah oder fern?



Mit welchen graphischen Techniken können Sie ein Gefühl der Tiefe hervorrufen?

In diesem Kapitel werden Sie lernen, wie Sie:

- Figuren von einer Punktperspektive zeichnen können
- Schattierungen anzuwenden um Tiefe zu zeigen
- Figuren von einer Zweipunktperspektive zeichnen können

Die Bildschirme von Fernsehern können nicht von alleine Tiefe darstellen. Sie sind zweidimensional. Trotzdem kann man den TV-Firguren einen dreidimensionalen Charakter verleihen. Um eine dreidimensional wirkende Figur zu erzeugen, müssen wir unseren Sehsinn täuschen. Sehen Sie sich die Zeichnung zu Beginn des Kapitels einmal genau an. Obwohl sie auf einer zweidimensionalen Ebene erzeugt wurde, wird durch die Linien und die unterschiedlichen Wortgrößen ein Gefühl der Tiefe erzeugt.

Ein-Punktperspektive

Erfahrung hat uns gelehrt, daß nahe Objekte größer wirken, als ferne Objekte der gleichen Größe. Objekte, die sich sehr weit weg befinden, erscheinen nicht größer, als ein kleiner Punkt am Horizont. Es ist als wenn Sie beinahe außer Sicht geraten sind. Dies ist der Fluchtpunkt. Wir werden das Prinzip des Fluchtpunkte verwenden, um unseren Zeichnungen Tiefe zu geben.
Lassen Sie uns mit einem einzelnen Fluchtpunkt oder einer Punktperspektive beginnen. Stellen Sie sich einen, in gleichmäßige Quadrate unterteilten, Gehweg vor. Ein Gehweg wird je nach Blickpunkt unterschiedlich erscheinen. Bild 4-1 stellt zwei Ansichten eines Gehwegs dar. Ansicht A ist von oben, vielleicht von einem Flugzeug. Ansicht B ist von vorn, vielleicht aus einer Standperspektive.
Die von Ansicht B angefertigte Zeichnung vermittelt ein Gefühl der Tiefe. Wir werden die in Bild 4-1 (B) gezeigte Zeichnung auf den Bildschirm bringen, doch werfen wir zuerst einen Blick auf den Aufbau einer Zeichnung aus der Punktperspektive.
Als erstes wählen Sie einen Fluchtpunkt. Dann ziehen Sie zwei diagonale Linien von dem Punkt (Bild 4-2 (B)). Die Länge der Linien richtet sich nach der Zeichnung in Bild 4-2 (B). Verbinden Sie die beiden Enden des ersten Gehwegsquadranten.
Skizzieren Sie dann eine weitere Konstruktionslinie parallel dazu (Bild 4-2 (C)).

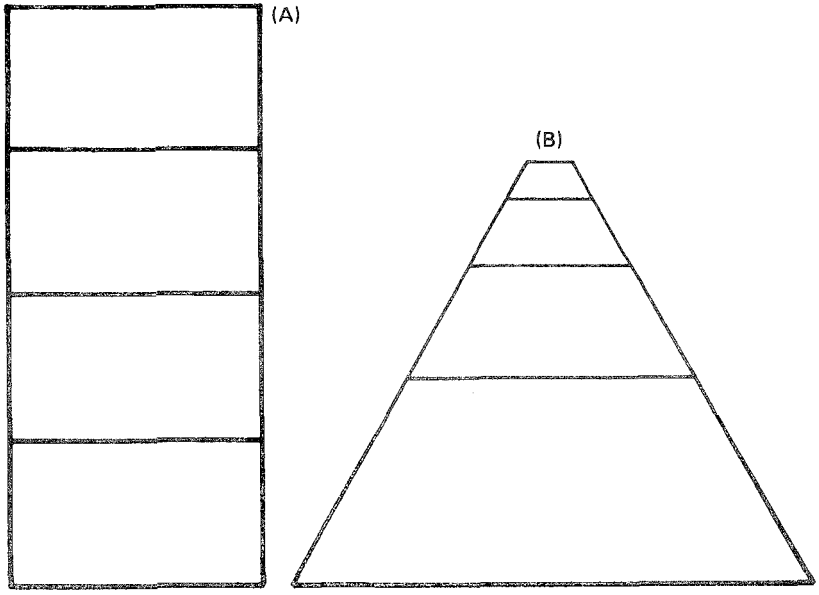
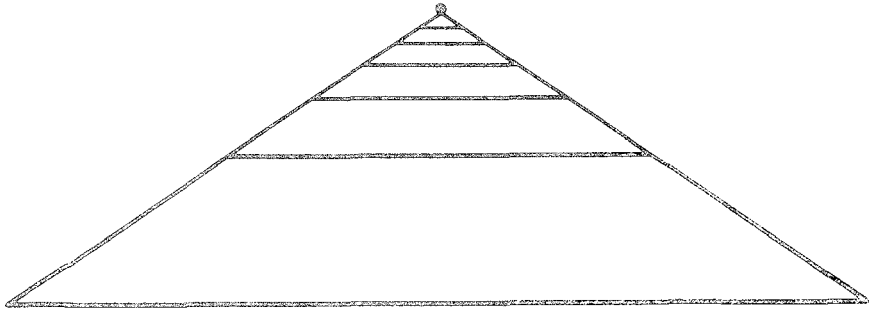


BILD 4-1: Zwei Ansichten eines Gehwegs.

Ziehen Sie eine horizontale Linie, um das nächste Quadrat des Gehwegs darzustellen. Fügen Sie so weitere Konstruktionslinien hinzu, bis Sie die gewünschte Anzahl Quadrate haben (Bild 4-2 (D)). Sie können das Aussehen des Gehwegs auf unterschiedliche Art und Weise verändern.

1. Sie können die Breite des Gehwegs verändern und den Fluchtpunkt näher am Horizont erscheinen lassen, indem Sie die Basis des Gehwegs verbreitern.



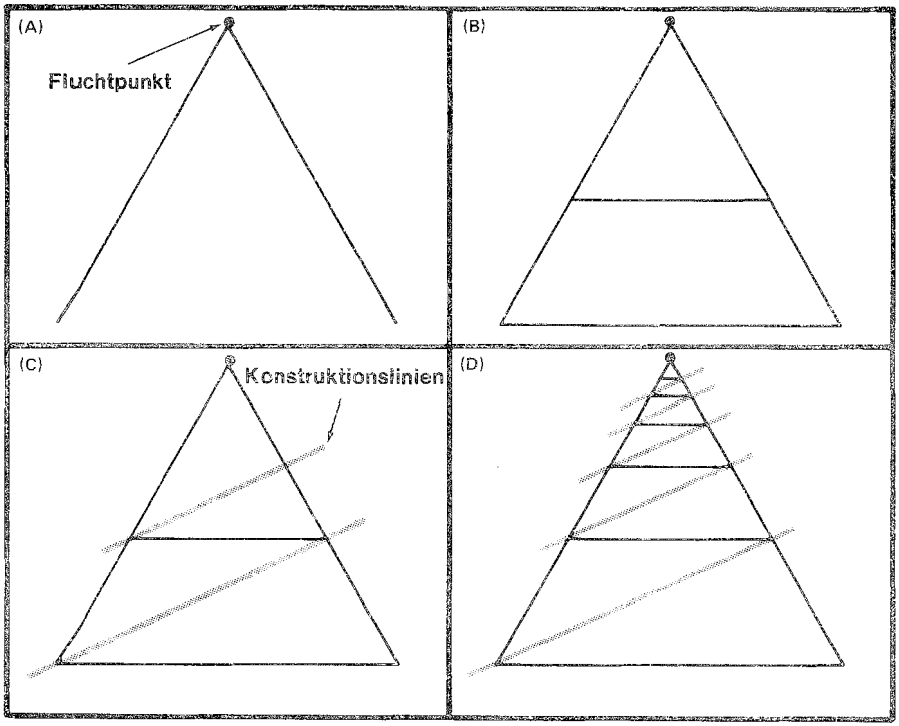
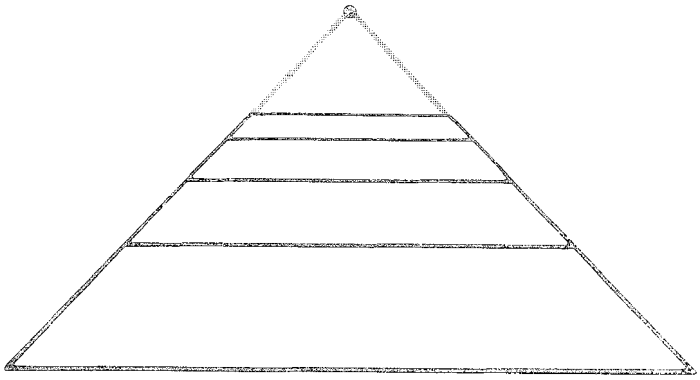


BILD 4-2: Konstruktion von einer Punktperspektive.

2. Ihr Gehweg kann kurz vor dem Fluchtpunkt enden.



Eine Computerskizze

Lassen Sie uns die Zeichnung des Gehwegs auf den Bildschirm bringen. Zuerst wird der Gehweg auf einem Bildschirmarbeitsblatt skizziert. Wir wollen dazu GRAPHIK-Modus 7 benutzen. Es könnte dann aussehen, wie in Bild 4-3.

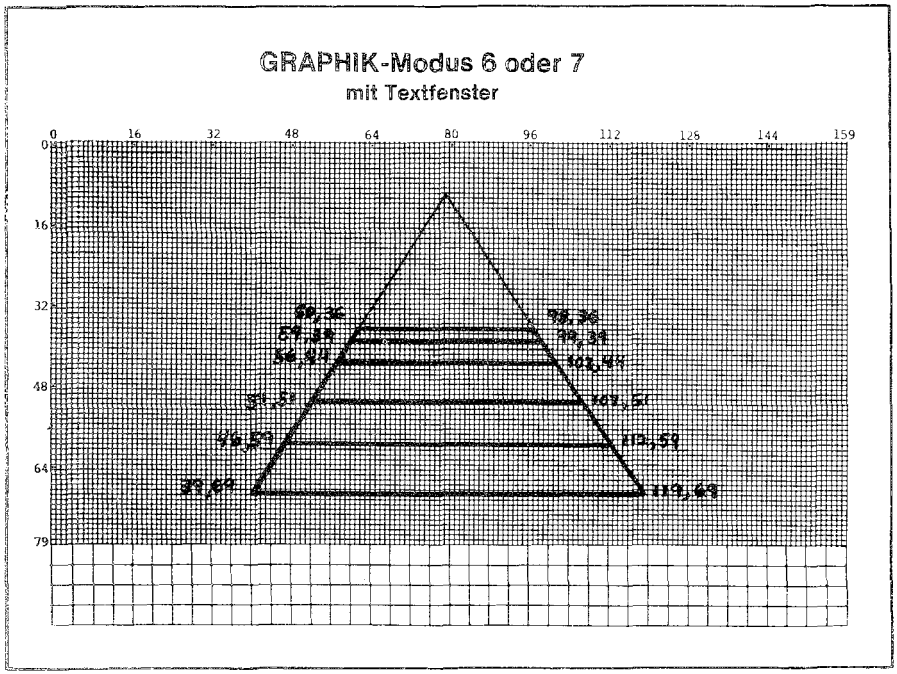


BILD 4-3: Entwurf eines Gehwegs auf einem Arbeitsblatt im Modus 7.

Beachten Sie, daß wir die Linien nicht bis ganz zum Fluchtpunkt gezeichnet haben. Tatsächlich soll der Fluchtpunkt auf dem Bildschirm gar nicht zu erkennen sein.
Wir sind jetzt bereit ein Programm zu schreiben mit dem unsere Skizze erstellt wird. Eine einfache Möglichkeit ist es, die Endpunkte jeder Linie

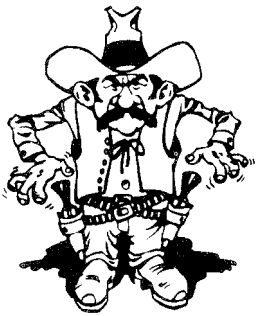
zu definieren, dann einen der Endpunkte zu PLOTten und mittels DRAWTO mit dem anderen zu verbinden. Beachten Sie, daß die Endpunkte in dem vorstehendem Diagramm gekennzeichnet sind. Ihr Programm könnte folgendermaßen aussehen:

```
100 REM GEHWEG - PUNKTPERSPEKTIVE
200 REM SETZT GRAPHIKMODUS
210 GRAPHICS 7
220 COLOR 1
300 REM LESE DATEN UND ZIEHE LINIEN
310 READ LINES
320 FOR L=1 TO LINES
330 READ P1,P2,D1,D2
340 PLOT P1,P2:DRAWTO D1,D2
350 NEXT L
360 DATA 8
370 DATA 60,36,98,36,59,39,99,39,56,44,103,44
380 DATA 51,51,107,51,46,59,112,59,39,69,119,69
390 DATA 60,36,39,69,98,36,119,69
800 END
```

Geben Sie die Zeilen ein und starten Sie das Programm.
Die Werte der Endpunkte sind in den DATA-Anweisungen der Zeilen 370, 380, 390 enthalten. Wenn die READ-Anweisung der Zeile 330 ausgeführt wird, werden vier Datenwerte (zwei Graphikpunkte) gelesen. Diese Werte werden dann in Zeile 340 verwendet, um die Linie zu ziehen. Der Arbeitsgang wird entsprechend der in der Zeile 360 gegebenen DATA-Anweisung für die festgelegte Anzahl Linien wiederholt. Wenn Sie mit den READ und DATA Anweisungen noch nicht vertraut sind, sollten Sie ein BASIC-Programmierlehrbuch oder das ATARI BASIC Handbuch zu Hilfe nehmen.

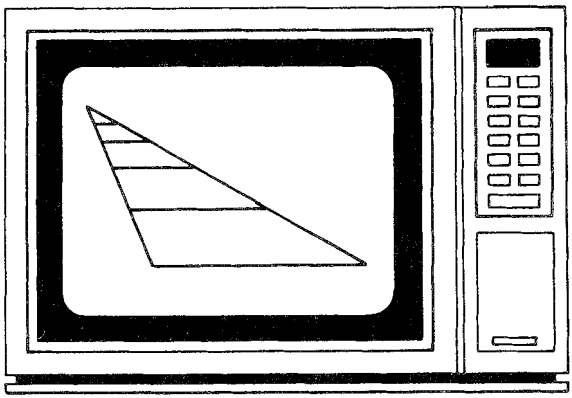
Tiefenwirkung durch Schattierung

Je näher ein Objekt ist, desto besser können wir es sehen. Verschiedene Helligkeiten der selben Farbe können verwendet werden, um den Eindruck von Tiefe zu wecken. Die näheren Teile eines Objektes erscheinen heller, als die entfernteren Abschnitte desselben Objektes.
Wir wollen die Gehwegzeichnung aus dem vorherigen Programm verwenden und die einzelnen Rechtecke wie in Bild 4-4 illustriert schattieren.



Aufgabe #7

Zeichnen Sie einen Gehweg im GRAPHIK-Modus 7, der dem hier abgebildeten gleicht. (TIP: Skizzieren Sie die Zeichnung auf einem Arbeitsblatt. Definieren Sie die einzelnen Endpunkte jeder Linie und ändern Sie entsprechend die Werte in den DATA-Anweisungen des vorherigen Programms.)



Wir könnten, wie wir das in Kapitel 3 gemacht haben, angrenzende Linien zeichnen, um die gewünschten Abschnitte zu färben. Vielleicht wäre es aber in diesem Fall, die einfachere Methode den Farbbefehl auszunutzen. Bevor wir den Gehweg schattieren, sollten wir uns genau ansehen, wie der Farbbefehl funktioniert. Die folgenden Schritte werden Ihnen helfen die Farbbefehle zu verstehen. Probieren Sie sie aus.

1. Zeichnen Sie eine vertikale Linie, um den rechten Rand der zu färbenden Fläche zu markieren. Es muß keine gerade Linie sein. Tippen Sie diese drei Zeilen.

```
GRAPHICS 7  <RETURN>

COLOR 1  <RETURN>

PLOT 90,20:DRAWTO 110,60  <RETURN>
```

Jetzt müssten Sie sehen, wie eine Linie auf dem Bildschirm gezeichnet wird.

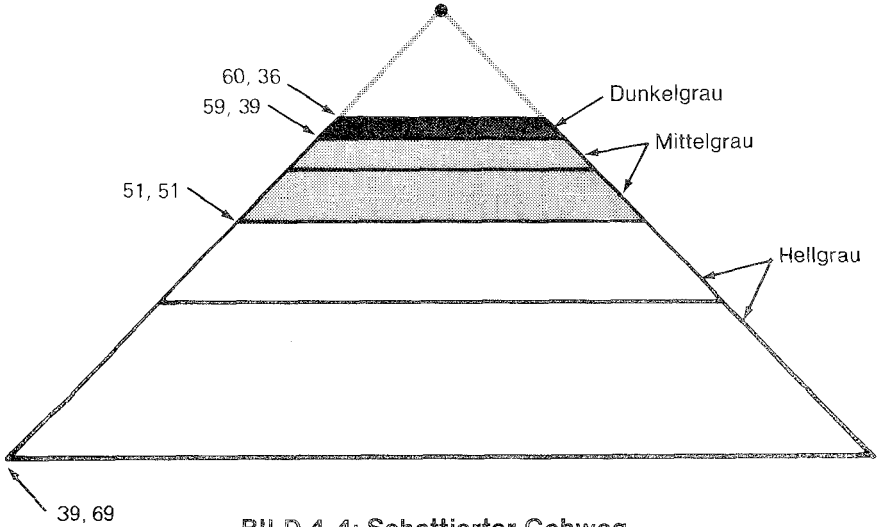


BILD 4-4: Schattierter Gehweg.

2. Verwenden Sie die POKE-Anweisung, um den Farbton festzusetzen. Tippen Sie

```
POKE 765,1  <RETURN>
```

Die Farbe des Abschnitts wird COLOR 1 sein.

3. Verwenden Sie die PLOT-Anweisung, um die obere linke Ecke des Abschnitts festzulegen. Vergleichen Sie mit Bild 4-5. Tippen Sie

```
PLOT 65,40  <RETURN>
```

Jetzt sollte ein einzelnes Pixel geplottet werden.

4. Verwenden Sie die POSITION-Anweisung, um die untere linke Ecke des Abschnittts festzulegen. Tippen Sie

```
POSITION 60,50  <RETURN>
```

5. Ausführen des Farbbefehls. Tippen Sie

```
XIO 18,#6,0,0,"S:"  <RETURN>
```

Der Bildschirm sollte wie Bild 4-5 aussehen.

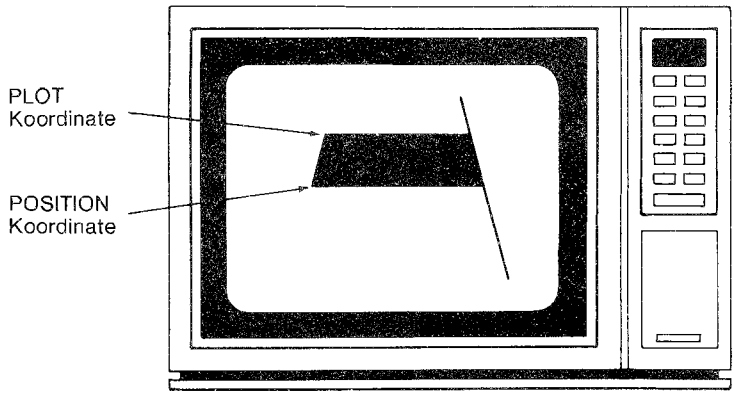


BILD 4-5: Farbbeispiel.

Gut, schattieren wir den Gehweg. Fügen Sie diese Zeilen zu dem Gehwegprogramm hinzu:

```
400 REM SCHATTIERE GEHWEG
500 REM SETZE FARBEN
510 SETCOLOR 0,0,4: SETCOLOR 1,0,8:SETCOLOR 2,0,12
600 REM FAERBE GEHWEG MIT DREI FARBEN
630 FOR C=1 TO 3
640 READ P1,P2,F1,F2
650 POKE 765,C
660 PLOT P1,P2:POSITION F1,F2
670 XIO 18,#6,0,0,"S:"
680 NEXT C
690 DATA 60,36,59,39,59,39,51,51,51,51,39,69
```

Starten Sie es.
Die Punkte die geplottet und positioniert werden sollen, sind in der DATA-Zeile 690 enthalten. Diese Punkte werden in Zeile 640 gelesen und in Zeile 660 auf den Bildschirm gebracht.
Nur die Farbbegrenzung des GRAPHIK-Modus 7 beschränkt die Klarheit mit der Sie die Zeichnung anfertigen können. Wenn Sie einen GTIA-Chip in Ihrem Computer haben, können Sie die Zeichnung im GRAPHIK-Modus 9 erstellen. Im Modus 9 können Sie bis zu 16 verschiedene Schattierungen wählen.

Wenn Sie einen GTIA-Chip in Ihrem Computer haben, sollten Sie folgendes Beispiel ausprobieren:

```
100 REM SCHATTIERTER GEHWEG IN MODUS 9
200 REM SETZE GRAPHIKMODUS
210 GRAPHICS 9
220 SETCOLOR 4,0,0
300 REM ZEICHNE 16 HORIZONTALE LINIEN
310 FOR C=15 TO 0 STEP -1
315 COLOR 15-C
320 FOR Y=0 TO 2
330 PLOT 69-2*C,100-3*C+Y:DRAWTO 2*C+10,100-3*C+Y
340 NEXT C
490 GOTO 490
```

Zwei-Punktperspektive

Bis jetzt wurden unsere Zeichnungen anhand eines einzelnen Fluchtpunktes erstellt. Durch Zeichnen nach einer Zweipunktperspektive, kann eine weitere Dimension hinzugefügt werden. Lassen Sie uns einen Kasten zeichnen. Wie wir es schon bei der Zeichnung aus der Punktperspektive getan haben, wollen wir uns auch hier erst einmal mit dem Aufbau beschäftigen.
Zu Beginn wählen Sie zwei Fluchtpunkte und ziehen eine vertikale Linie, um die vordere Eckseite des Kastens darzustellen (Bild 4-6 (A)). Als nächstes verbinden Sie die Fluchtpunkte jeweils mit dem Anfang und Ende der Linie (Bild 4-6 (B)). Jetzt ziehen Sie zwei weitere vertikale Linien, um die beiden Frontseiten des Kastens zu markieren (Bild 4-6 (C)). Vollenden Sie den Deckel des Kastens mit zwei weiteren Konstruktionslinien (Bild 4-6 (D)).
Jetzt wissen Sie, wie Sie eine Kiste darstellen können. Wir sind bereit unser Programm zu starten. Als erstes müssen wir unseren Kasten auf einem Arbeitsblatt skizzieren. Dazu wählen wir den GRAPHIK-Modus 7. Ihre Zeichnung auf dem Bildschirm kann der Figur in Bild 4-7 ähnlich sehen. Beachten Sie, daß die Enden jeder Linie gekennzeichnet wurden. Das Programm mit dem der Kasten gezeichnet wird, unterscheidet sich nicht besonders von dem Programm mit dem der Gehweg erstellt wurde. Die einzige Differenz liegt in den DATA-Anweisungen.
Das Programm sieht folgendermaßen aus. Geben Sie es ein und testen Sie es.

```
100 REM KASTEN - ZWEIPUNKTPERSPEKTIVE
200 REM SETZE GRAPHIK-MODUS
210 GRAPHICS 7
```

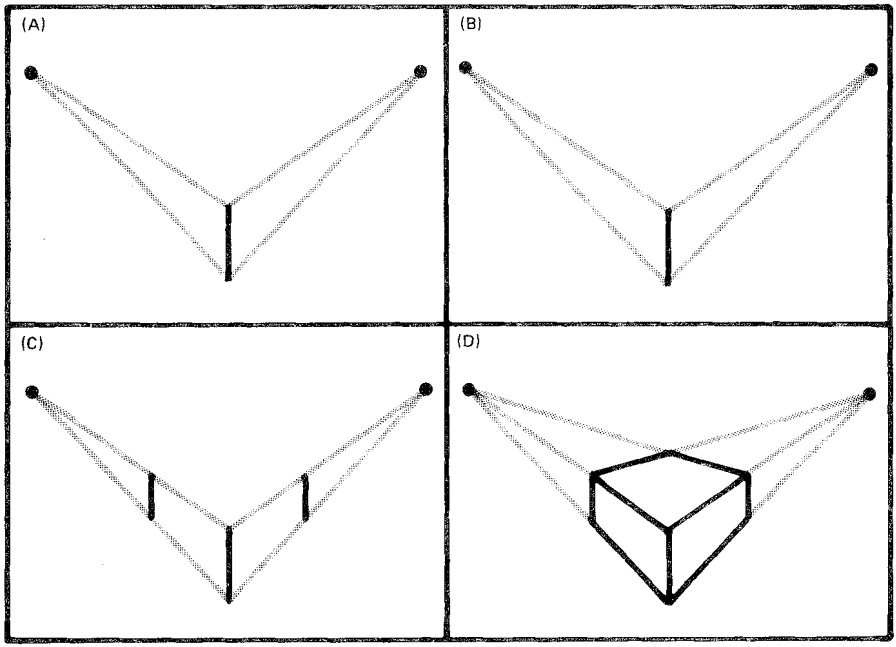


BILD 4-6: Konstruktion mit einer Zwei-Punktperspektive.

```
220 COLOR 1
300 REM LESE DATA UND ZEICHNE LINIEN
310 READ LINES
320 FOR L=1 TO LINES
330 READ P1,P2,D1,D2
340 PLOT P1,P2:DRAWTO D1,D2
350 NEXT L
360 DATA 9
370 DATA 49,36,49,47,49,47,79,69,79,69,109,47
380 DATA 109,47,109,36,109,36,79,31,79,31,49,36
390 DATA 49,36,79,49,79,49,109,36,79,49,79,69
800 END
```

Lassen Sie uns mittels des Farbbefehls eine Seite färben. Fügen Sie diese Zeilen hinzu:

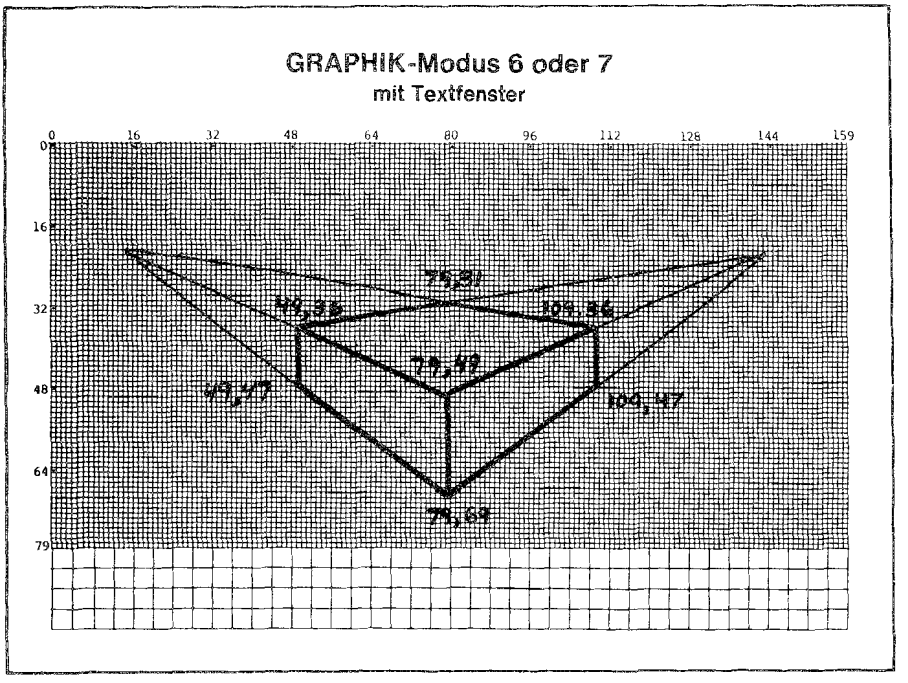


BILD 4-7: Skizze eines Kastens auf einem Arbeitsblatt im Modus 7.

```
400 REM SCHATTIERE DIE LINKE SEITE
500 REM SETZE FARBEN
510 SETCOLOR 0,0,14
600 REM FAERBE LINKE SEITE MIT FARBE
610 POKE 765,1
620 PLOT 49,36:POSITION 49,47
630 XIO 18,#6,0,0,"S:"
640 PLOT 49,47:POSITION 79,68
650 XIO 18,#6,0,0,"S:"
```

Da der Farbbefehl nur horizontal färbt, mußten wir ihn zweimal durchführen lassen. Bild 4-8 zeigt die beiden Abschnitte, die jeweils getrennt gefärbt werden mußten.

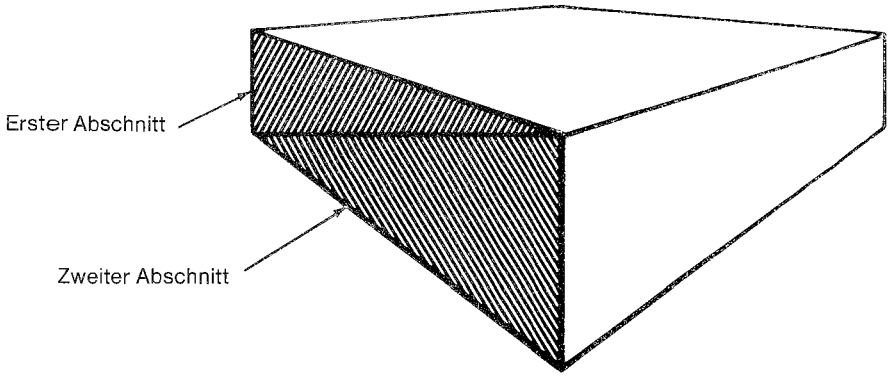
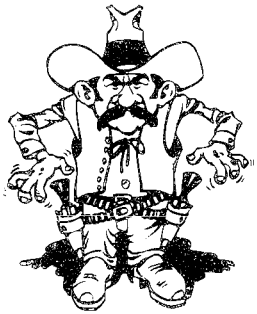
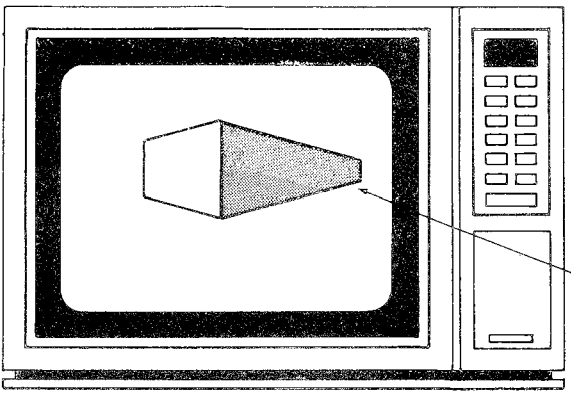


BILD 4-8: Farbbefehl für unregelmäßige Formen.



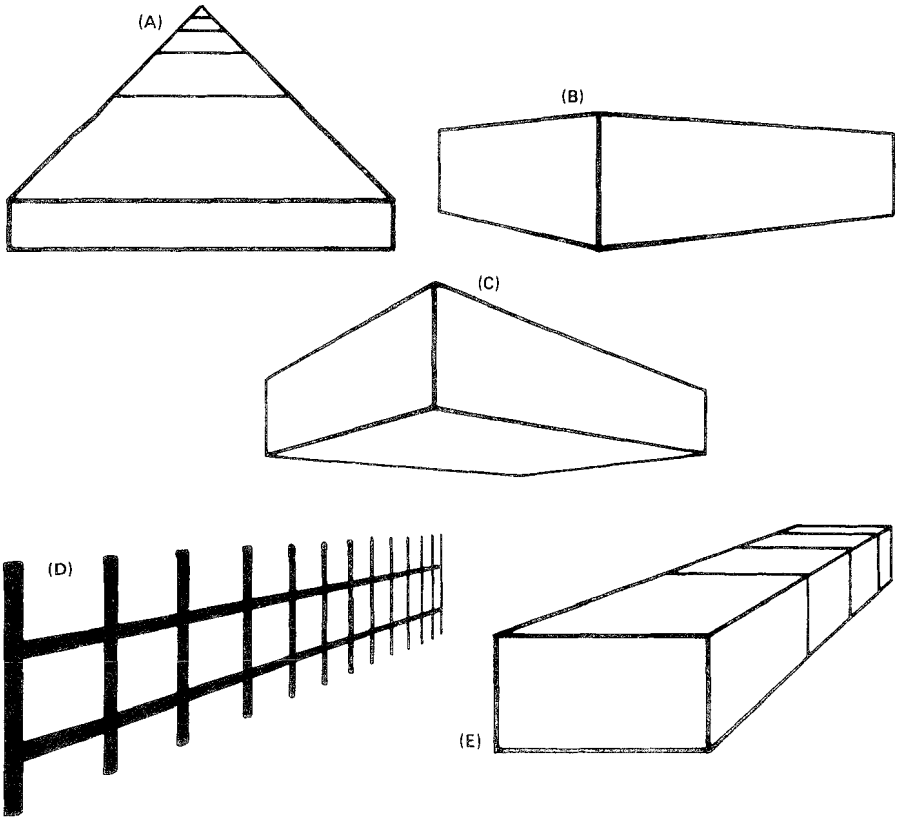
Aufgabe #8

Zeichnen Sie die folgende Figur auf den Bildschirm. Der Hintergrund soll schwarz und die Randlinien in einem mittleren Grau sein.



Zusammenfassung von Kapitel 4

Verwenden Sie die folgenden Diagramme für die Fragen 1, 2 und 3.



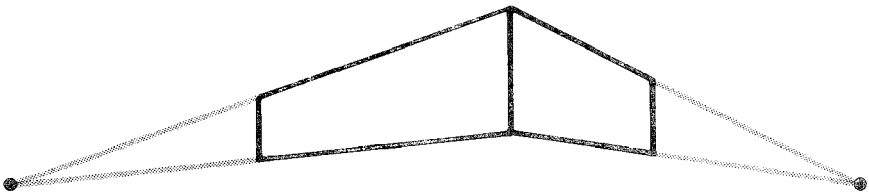
1. Lokalisieren Sie den/die Fluchtpunkt(e) von jeder Skizze.

2. Welche dieser Skizzen wurden mit der Ein-Punktperspektive gezeichnet?

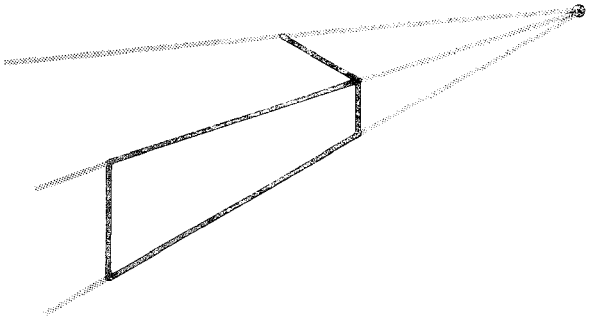
3. Welche der Skizzen mit der Zwei-Punktperspektive?

Vervollständigen Sie die folgenden Kastenkonstruktionen in den Fragen 4,5 und 6.

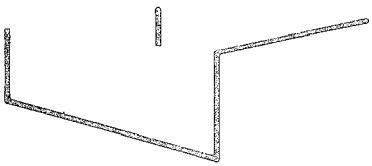
4.



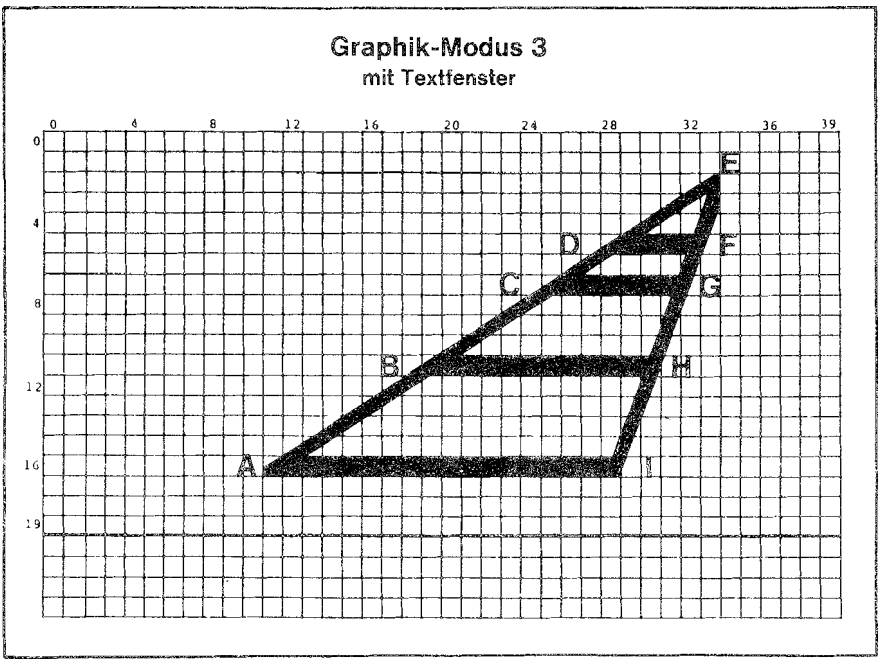
5.



6.



7. Wie lauten die Endpunkte für jede der Linien in der folgenden Skizze? Die Skizze wurde auf einem Arbeitsblatt im Modus 3 gezeichnet.



A. _____ B. _____

C. _____ D. _____

E. _____ F. _____

G. _____ H. _____

I. _____

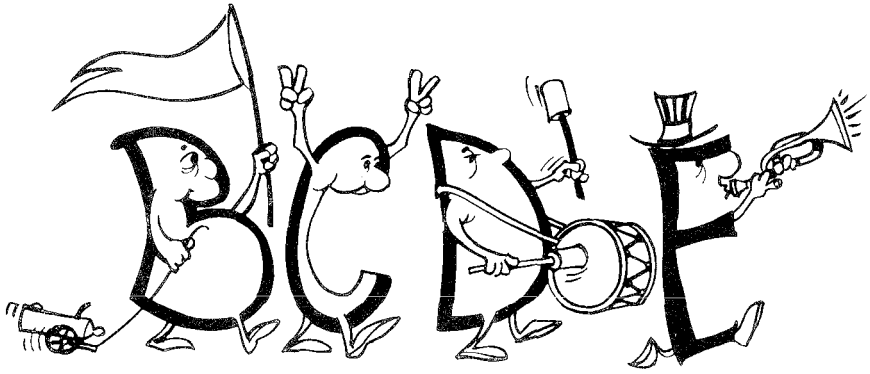
8. Skizzieren Sie die Figur die durch das folgende farbgebende Programm erstellt werden würde.

```
100 REM FARBBEFEHL PROBLEM
200 SETZE GRAPHIK-MODUS
210 GRAPHICS 7
300 REM ZEICHNE RECHTEN RAND
310 COLOR 1
320 PLOT 100,50:DRAWTO 140,10
330 POKE 765,1
340 PLOT 70,10:POSITION 30,50
350 XIO 18,#6,0,0,"S:"
```

KAPITEL 5

Zeichen

Lassen Sie uns einen Blick auf ein paar Zeichen werfen.



Manchmal können Informationen nicht einfach nur bildhaft dargestellt werden, wie dies in den Kapiteln 2, 3 und 4 der Fall war. Durch die geschickte Verwendung von Zeichen und Text kann einem Bildschirm oft mehr Klarheit und Informationsgehalt verliehen werden, als dies mit Bildern möglich wäre.
In diesem Kapitel werden Sie lernen, wie Sie:

- die GRAPHIK-Modi 0, 1 und 2 verwenden
- die Farbe einzelner Zeichen verändern
- nur aus Zeichen bestehende Graphiken erzeugen

ATARI-Zeichen

Was ist ein Zeichen? Im allgemeinen versteht man unter Zeichen Buchstaben und Zahlen. Aber es können auch Interpunktationszeichen oder Graphikzeichen sein. Wir werden später in diesem Buch unsere eigenen Zeichen entwerfen, aber bis dahin wollen wir die bereits im Computer gespeicherten Zeichen verwenden. Die Tabelle 5-1 zeigt den vom ATARI Computer normalerweise verwendeten Zeichensatz. Jedem der Zeichen ist ein eigener Dezimalwert zugeordnet. Auf die Nützlichkeit der Werte kommen wir später noch zurück.
Die drei Text-Modi die im ATARI-BASIC verwendet werden können sind die GRAPHIK-Modi 0, 1 und 2. Ein GRAPHIK-Modus 0 Textfenster kann auch in den Modi 1 bis 8 verwendet werden. Generell können alle der in Tabelle 5-1 gezeigten Zeichen in jedem GRAPHIK-Modus auf dem Bildschirm dargestellt werden.
Um die auf der Tastatur abgebildeten Zeichen zu erhalten, brauchen Sie jeweils nur die gewünschte Taste zu drücken. Spezielle Tasten und Tastenkombinationen werden benötigt um Kleinbuchstaben und Graphikzeichen zu produzieren. In Bild 5-1 wird das Diagramm einer Tastatur dargestellt.
Eine genaue Beschreibung der ATARI Tastatur finden Sie in Ihrem ATARI-Bedienungshandbuch. Hier nur ein kurzer Überblick.

Graphikzeichen

Wir wollen ein paar Zeichen ausprobieren. Beachten Sie die Position der <CTRL>-Taste auf der linken Seite der Tastatur. Halten Sie die <CTRL>-Taste gedrückt während Sie ein paar Buchstaben von A bis Z drücken. Die Zeichen, die Sie auf dem Bildschirm sehen, werden Graphikzeichen genannt.

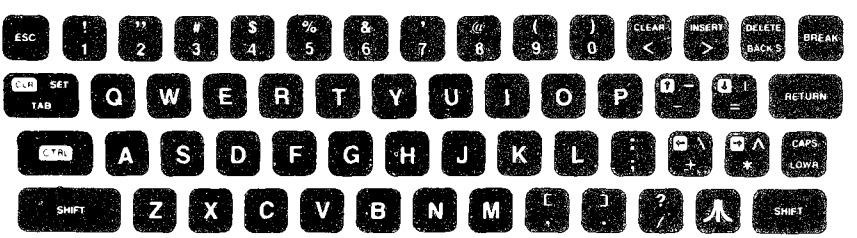


BILD 5-1: ATARI TASTATUR.

Mit Genehmigung der Atari, Inc. 1980 abgedruckt aus dem Atari 400/800 BASIC REFERENCE MANUAL.

Tabelle 5-1: ATARI-Zeichensatz

Dezimalwert	Zeichen	Dezimalwert	Zeichen	Dezimalwert	Zeichen	Dezimalwert	Zeichen	Dezimalwert	Zeichen	Dezimalwert	Zeichen	Dezimalwert	Zeichen	Dezimalwert	Zeichen	Dezimalwert	Zeichen	Dezimalwert	Zeichen
0		13		26		39	'	52	4	65	A	78	N	91	[	104	h	117	u
1		14		27		40	(	53	5	66	B	79	O	92	\	105	i	118	v
2		15		28		41	)	54	6	67	C	80	P	93	]	106	j	119	w
3		16		29		42	*	55	7	68	D	81	Q	94	^	107	k	120	x
4		17		30		43	+	56	8	69	E	82	R	95	_	108	l	121	y
5		18		31		44	,	57	9	70	F	83	S	96		109	m	122	z
6		19		32	Space	45	-	58	:	71	G	84	T	97	a	110	n	123	
7		20		33	!	46	.	59	;	72	H	85	U	98	b	111	o	124	|
8		21		34	"	47	/	60	<	73	I	86	V	99	c	112	p	125	
9		22		35	#	48	0	61	=	74	J	87	W	100	d	113	q	126	
10		23		36	$	49	1	62	>	75	K	88	X	101	e	114	r	127	
11		24		37	%	50	2	63	?	76	L	89	Y	102	f	115	s		
12		25		38	&	51	3	64	@	77	M	90	Z	103	g	116	t		

Adapted from the ATARI® 400/800 BASIC REFERENCE MANUAL by permission of Atari, Inc. © 1980.

Kleinbuchstaben

Für Kleinbuchstaben drücken Sie die mit <CAPS LOWR> bezeichnete Taste. Sie brauchen Sie nicht gedrückt zu halten wie die <CTRL>-Taste. Probieren Sie verschiedene Tasten von A bis Z.
Um wieder Großbuchstaben schreiben zu können, drücken Sie die <SHIFT>-Taste und gleichzeitig die <CAPS LOWR> Taste. Auch mit <SYSTEM RESET> können Sie wieder in die gewohnte Schreibweise zurückkehren.

Umkehrschrift (Inverses Video)

Alle vom ATARI Computer verwendeten Zeichen können auch in inversem Video dargestellt werden. Dies ist eine Umkehr zwischen Hintergrund- und Zeichenunterlegung. Bild 5-2 gibt den Buchstaben H einmal in normalen und einmal in inversem Video wieder.

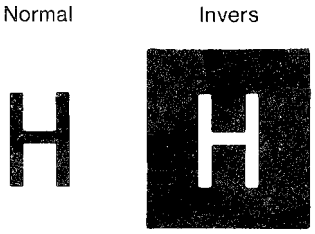


BILD 5-2: Inverses Videozeichen.

Um inverses Video zu produzieren, drücken Sie die (Atari)-Taste. Versuchen Sie es und drücken Sie dann einige der Zeichen von A bis Z. Um wieder in die normale Schreibweise zu gelangen drücken Sie einfach erneut die (Atari)-Taste.

GRAPHIK-Modus 0

Lassen Sie uns ein Programm schreiben mit dem ein paar Zeichen auf dem Bildschirm dargestellt werden. Geben Sie die folgenden Zeilen ein. Vergessen Sie nicht das Semikolon (;) in Zeile 30.

```
10 GRAPHICS 0
20 FOR X=1 TO 760
30 PRINT "A";
40 NEXT X
```

Starten Sie es. Sie sollten jetzt 20 Reihen voller A's sehen. Jede Reihe enthält 38 Zeichen. 20 Reihen a 38 Zeichen macht insgesamt 760 Zeichen. Beachten Sie, daß die Zeile 20 unseres Programms die FOR...TO, NEXT Schleife 760 Mal ausführt. Weiter können Sie feststellen, daß die beiden Zeichenpositionen auf der linken Seite nicht verwendet wurden. Sie werden generell nicht verwendet, da Sie auf einigen Monitoren nicht besonders gut dargestellt werden. Wir werden Ihnen gleich zeigen, wie Sie auch diese beiden Positionen verwenden können, aber lassen Sie uns vorher noch einmal zu unserem Programm zurückkehren und ein anderes Zeichen als den Buchstaben A produzieren.
Verändern Sie Zeile 30 in

```
30 PRINT "+";
```

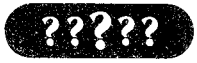
(Drücken Sie die <CTRL>-Taste und gleichzeitig die S-Taste, um dieses Zeichen zu erhalten.)

Lassen Sie es laufen. Der Bildschirm sollte nun wie ein Stück Graphikpapier aussehen. Sie können auch andere Graphikzeichen ausprobieren, um verschiedene Designs zu entwerfen.

Sie können die Zeichen auch indirekt produzieren, ohne daß Sie die Tasten drücken müssen, die die gewünschten Zeichen repräsentieren. Jedes Zeichen wird im Computer durch eine Zahl initialisiert. Für diese Initialisierung gibt es eine Norm. Sie wird der ASCII Code (American Standard Code for Information Interchange) genannt. Die Dezimalwerte (ASCII Werte) für jedes ATARI Zeichen wurden bereits in Tabelle 5-1 wiedergegeben. Da die Graphikzeichen keiner Norm unterliegen, wurden die Werte in dieser Tabelle ATARI ASCII Code (ATASCII) genannt.
Lassen Sie uns ein Beispiel probieren. Der ATASCII Code für den Buchstaben A ist 65 (siehe Tabelle 5-1). Um den ATASCII Code der ATARI BASIC zu verwenden, können wir die CHR$-Funktion verwenden. Tippen Sie die folgende Zeile.

```
PRINT CHR$(65)  <RETURN>
```

Wurde der Buchstabe A dargestellt? Er sollte. Mittels der CHR$-Funktion können wir jeden Buchstaben mit einem Dezimalwert wiedergeben.



Welcher ATASCII Code wird verwendet, um den Buchstaben Z zu produzieren? Sehen Sie in Tabelle 5-1 nach.

Antwort #1_____

Wir wollen das vorherige Programm derart ändern, daß wir den Bildschirm voller A's schreiben, dabei aber nur die CHR$-Funktion verwenden. Ihr Programm sollte folgendermaßen aussehen:

```
10 GRAPHICS 0
20 FOR X=1 TO 760
30 PRINT CHR$(65);
40 NEXT X
```

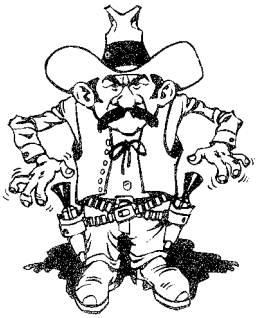
Testen Sie es. Sie sollten wie zuvor 20 Zeilen voller A's sehen. Indem Sie zu dem jeweiligen Dezimalwert der Tabelle 5-1 128 hinzuaddieren, sind Sie in der Lage jedes der dort aufgeführten Zeichen in inversem Video darzustellen. So würde ein inverses A den Dezimalwert 65 plus 128 oder 193 haben. Probieren wir das mal aus. Verändern Sie Zeile 30 in

```
30 PRINT CHR$(193);
```

Starten Sie es. Zeigt der Bildschirm jetzt lauter inverse A?
Lassen Sie uns verschiedene Zeichen produzieren. Machen Sie folgende Änderungen an dem Programm:

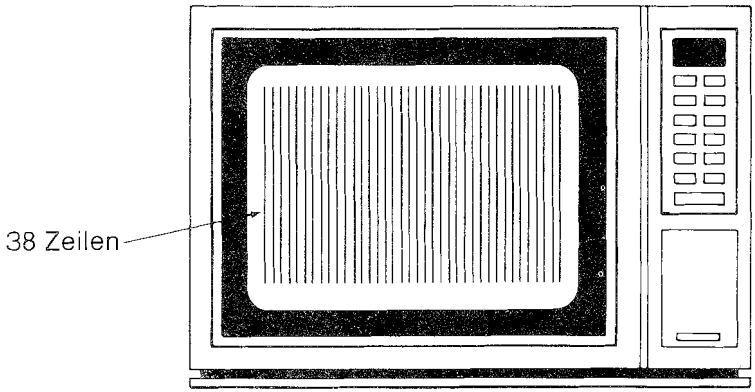
```
20 FOR X=65 TO 90

30 PRINT CHR$(X);
```

Starten Sie es. Sie sollten das ganze Alphabet in Großbuchstaben auf der oberen Hälfte des Bildschirms wiederfinden. Erkennen Sie die Möglichkeiten, die Sie durch die Wiedergabe von Zeichen in Dezimalwerten, oder wie in diesem Fall durch eine Variable, erhalten?
Nun zur Aufgabe #9.



Aufgabe #9

Schreiben Sie ein Programm, um folgende Bildschirmdarstellung mit Zeichen des Modus 0 zu produzieren.



Zeichenposition

Die Position der Zeichen auf dem Bildschirm kann auf verschiedene Weisen kontrolliert werden. Wir wollen ein paar Möglichkeiten ausprobieren. Klären Sie den Bildschirm. Halten Sie die SHIFT Taste gedrückt und drücken Sie die CLEAR Taste. Jetzt tippen Sie

```
PRINT "TEST";"WOERTER"  <RETURN>
```

Wurden die beiden Wörter nebeneinander und ohne eine Leerstelle dazwischen abgebildet? Sie sollten. Versuchen Sie jetzt diese Anweisung:

```
PRINT "TEST","WOERTER"  <RETURN>
```

Durch die Verwendung des Kommas wurde das zweite Wort nach rechts gerückt. Tatsächlich veranlaßt das Komma das eine Druckposition übersprungen wird.

Die Druckzonen sind jeweils um 10 Zeichen getrennt.
Das Komma und das Semikolon kontrollieren jeweils nur die horizontale Positionierung. Mit der POSITION-Anweisung können wir sowohl die horizontale Positionierung, als auch die vertikale Positionierung von Zeichen kontrollieren. Wir wollen das ausprobieren. Löschen Sie alle älteren Programme aus dem Computer und programmieren Sie dieses ein:

```
10 GRAPHICS 0
20 FOR P=20 TO 10 STEP -1
30 POSITION P,P
40 PRINT "TEST"
50 NEXT P
```

Starten Sie es. Der Druck wurde an der Koordinate 20,20 begonnen. Für jede neue Zeile wurde die Druckposition um eins nach oben und eins nach links verschoben, bis die Position 10,10 erreicht war. Mit der POSITION-Anweisung können wir überall auf dem Bildschirm schreiben, auch in den ersten beiden Spalten. Wir wollen es probieren. Klären Sie den Bildschirm und tippen Sie

```
POSITION 0,5:PRINT "TEST"  <RETURN>
```

Wir können auch die linke Randbegrenzung verändern, um die beiden Spalten auf der linken Seite zu verwenden. Tippen Sie

```
POKE 82,0  <RETURN>
```

Die linke Randbegrenzung für alle Druckvorgänge ist nun in Spalte 0. Die Nummer 82 in der POKE-Anweisung ist die Stelle, in der der Computer die Spaltennummer für die linke Randbegrenzung festlegt. Wenn wir diese Zahl mit der POKE-Anweisung auf 0 ändern, beginnt der Text in Spalte 0 des Bildschirms. Wir können so an jeder beliebigen Spaltennummer beginnen. Wir können auch die rechte Randbegrenzung verändern. Versuchen Sie es

```
POKE 83,6  <RETURN>
```

Drücken Sie jetzt eine Taste und halten Sie sie gedrückt, bis eine ganze Reihe Zeichen dargestellt sind. Sind die Grenzen bei 0 und 6? Sie können alles wieder normalisieren, wenn Sie die Taste <SYSTEM RESET> drücken.

?????

Mit welcher Anweisung können Sie die rechte Randbegrenzung in die Spalte 3 setzen?

Antwort #2_____

GRAPHIK-Modi 1 und 2

Wie Sie sich vielleicht aus Kapitel 1 erinnern werden, benutzt man die GRAPHIK-Modi 1 und 2, um farbige Zeichen zu produzieren, die größer sind als die in GRAPHIK-Modus 0 erzeugten Zeichen. Lassen Sie uns ein paar Graphikzeichen im Modus 2 auf den Bildschirm produzieren. Starten Sie dieses Programm:

```
10 GRAPHICS 2
20 FOR X=1 TO 200
30 PRINT#6;"*";
40 NEXT X
```

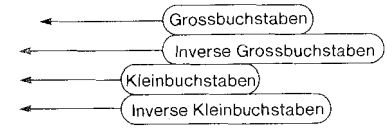
Wieviele Zeilen und Spalten haben Sie im Vergleich zu GRAPHIK-Modus 0? Da jedes Zeichen doppelt so hoch und doppelt so breit ist, wie in GRAPHIK-Modus 0 haben Sie auch nur halb soviele Zeilen und Spalten. Versuchen Sie den GRAPHIK-Modus 1. Verändern Sie Zeile 10 in

```
10 GRAPHICS 1
```

Starten Sie das Programm erneut. Wie verhalten sich die Zeichengrößen von Modus 1 und Modus 2 zueinander?
Bis jetzt waren alle Zeichen die wir in den Modi 1 und 2 produziert haben orange. Diese beiden GRAPHIK-Modi sind aber fünffarbig. Das heißt, daß Sie bis zu fünf Farben auf dem Bildschirm darstellen können. Der Hintergrund besteht aus einer dieser fünf Farben. Es bleiben also vier Farben für die Zeichen übrig. Das folgende Programm wird das Wort "FARBE" in vier verschiedenen Farben drucken. Geben Sie die Zeilen ein und starten Sie es:

#2. POKE 83,3

```
10 GRAPHICS 2
20 PRINT #6;"FARBE"
30 PRINT #6;"FARBE"
40 PRINT #6;"farbe"
50 PRINT #6;"farbe"
```



Beachten Sie, das nur Großbuchstaben produziert wurden. Wir werden bald auch Kleinbuchstaben darstellen.
Die Farben der Zeichen werden durch die Farbregister 0 bis 3 festgelegt. Die Tabelle 5-2 zeigt die verschiedenen Zeichenarten und ihre dazugehörigen Farbregister.

Tabelle 5-2: Zeichenarten und Farbregister

Zeichenart	SETCOLOR Register#	ursprüngliche Farbe
Großbuchstaben	0	Orange
Kleinbuchstaben	1	Hellgrün
Inverse Großbuchstaben	2	Dunkelblau
Inverse Kleinbuchstaben	3	Rot
Zahlen	0	Orange
Inverse Zahlen	2	Dunkelblau

Mit Genehmigung der Atari, Inc. 1980 aus dem ATARI 400/800 BASIC REFERENCE MANUAL übernommen.

Sie können die Farben der Zeichen wechseln, indem Sie die SETCOLOR-Anweisung verwenden, um die Werte der Farbregister aus Tabelle 5-2 zu verändern. Versuchen Sie es. Tippen Sie:

```
SETCOLOR 0,8,2  <RETURN>
```

Das Farbregister 0 enthält den Farbwert für die Farben der Großbuchstaben. Die ehemals orangen Zeichen sind jetzt dunkelblau.

Mit welcher Anweisung würden Sie Kleinbuchstaben in dunkelblau darstellen?

Antwort #3_____

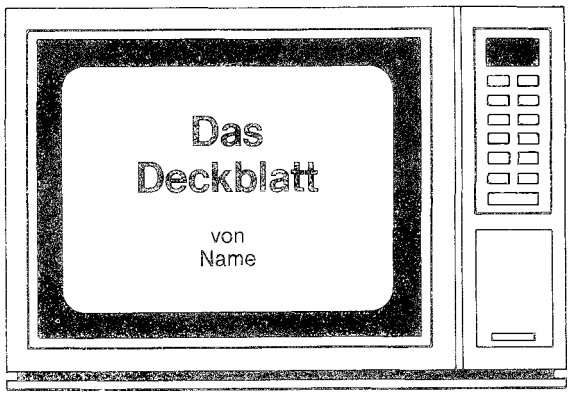
Bis jetzt sind immer nur Großbuchstaben auf dem Bildschirm erschienen. Normalerweise können Groß- und Kleinbuchstaben nicht gleichzeitig auf dem Bildschirm dargestellt werden. Wir können aber die POKE-Anweisung verwenden, um alle Großbuchstaben in Kleinbuchstaben zu verwandeln. Versuchen Sie es und tippen Sie

```
POKE 756,226  <RETURN>
```

Alle Buchstaben auf dem Bildschirm sollten nun kleingeschrieben sein. Sie haben vielleicht schon all diese Herzen entdeckt, die Leerstellen ausgefüllt haben. Momentan können wir sie nur loswerden, wenn wir ein anderes Zeichen darüberschreiben. In Teil 2 werden Sie sehen, wie Sie die Herzen in Leerstellen umwandeln können, indem Sie den Zeichensatz ändern.

Titelseite

Lassen Sie uns ein kleines Projekt versuchen. Stellen Sie sich vor wir wollen ein hübsches Titelbild für ein von uns geschriebenes Programm entwerfen. Es könnte ungefähr so aussehen:



#3. SETCOLOR 1,8,2

Wir werden einen geteilten Bildschirm im GRAPHIK-Modus 2 benutzen. Den Hintergrund und das Textfenster wollen wir schwarz gestalten. Die Wörter "DAS" und "DECKBLATT" sollen hellblau sein. Tippen Sie Ihren Namen in das Textfenster. Beantworten Sie bitte die folgenden Fragen, um zu überprüfen ob Sie alles begriffen haben.

Welche GRAPHIK-Anweisung werden Sie in diesem Programm verwenden?

Antwort #4_____

Welche SETCOLOR-Anweisung werden Sie verwenden, um das Textfenster schwarz zu gestalten?

Antwort #5----------

Das Programm könnte folgendermaßen aussehen:

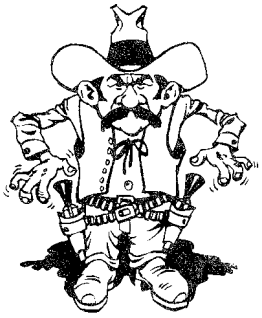
```
10  REM TITELBILD
20  GRAPHICS 2
30  SETCOLOR 2,0,0:SETCOLOR 0,9,10
40  POSITION 8,2:PRINT #6;"DAS"
50  POSITION 7,4:PRINT #6;"DECKBLATT"
70  PRINT "                  VON"
80  PRINT "                OSWIN"
100 GOTO 100
```

Versuchen Sie es. Erfüllt das Titelbild die Erwartungen?
Lassen Sie uns nun noch eine kleine Veränderung an dem Programm vornehmen. Fügen Sie diese Zeile hinzu:

```
65 POKE 752,1
```

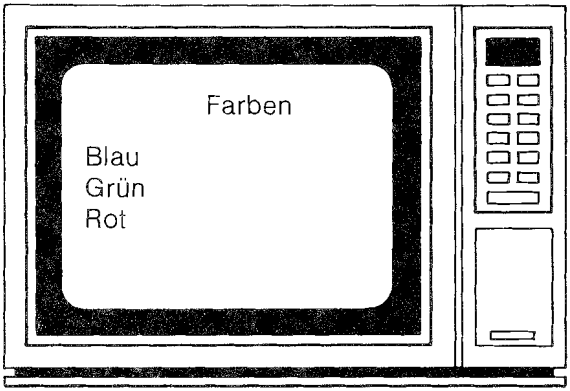
Starten Sie es jetzt. Haben Sie festgestellt, daß der Cursor unsichtbar wurde und nicht mehr auf dem Bildschirm zu sehen ist. Gut, jetzt Aufgabe #10.

#4. GRAPHIK 2 #5. SETCOLOR 2,0,0



Aufgabe #10

Schreiben Sie ein Programm, um die folgende Bildschirmdarstellung im GRAPHIK-Modus 2 zu realisieren. Das B ist blau, das G ist grün und das R ist rot. Alle anderen Buchstaben sollen orange und der Hintergrund schwarz sein.



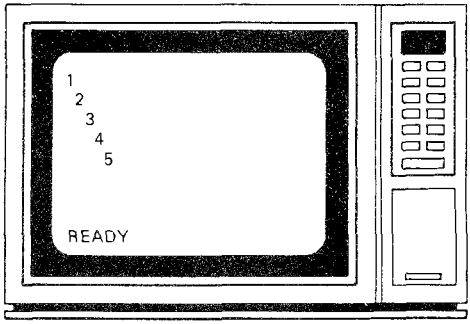
Zusammenfassung von Kapitel 5

Erstellen Sie die folgenden Bilddarstellungen mit den entsprechenden Programmen.

1. --------

```
10 GRAPHICS 2
20 FOR X=5 TO 1 STEP -1
30 POSITION 5-X,X:PRINT #6;X
40 NEXT X
```

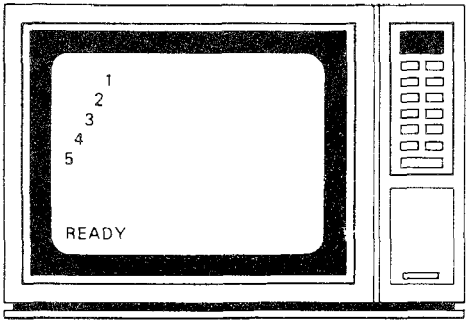
(A)



2. --------

```
10 GRAPHICS 2
20 FOR X=54 TO 49 STEP -1
30 PRINT #6;CHR$(X)
40 NEXT X
```

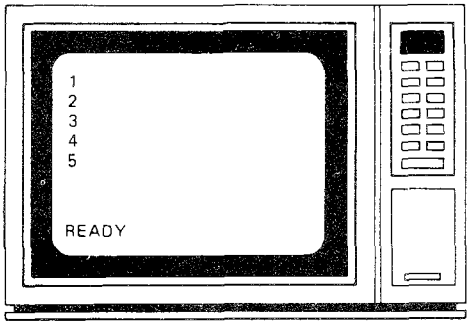
(B)



3. --------

```
10 GRAPHICS 2
20 FOR X=1 TO 5
30 POSITION X,X:PRINT #6;X
40 NEXT X
```

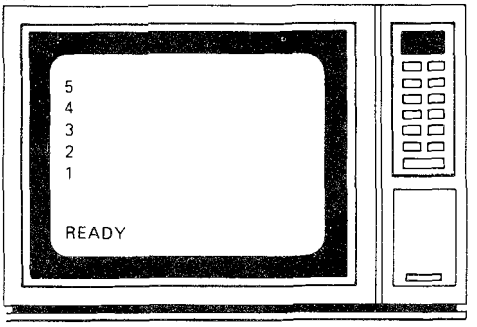
(C)



4. --------

```
10 GRAPHICS 2
20 FOR X=49 TO 54
30 PRINT #6;CHR$(X)
40 NEXT X
```

(D)



5. Welche Zeichen würden durch die folgenden Anweisungen produziert werden?

PRINT CHR$(72) _____

PRINT CHR$(6) _____

PRINT CHR$(198) _____

Wenn die folgenden Zeichen im GRAPHIK-Modus 1 geschrieben werden würden, in welcher Farbe würden Sie dann erscheinen und mit welchem SETCOLOR Register könnte die Farbe geändert werden?

	Ursprüngliche Farbe	SETCOLOR Register	
6. C	a)_____	b)_____	(Inverses Video)
o	a)_____	b)_____	
L	a)_____	b)_____	
o	a)_____	b)_____	(Inverses Video)
r	a)_____	b)_____	

7. Mit welcher Anweisung wird die rechte Randbegrenzung im GRAPHIK-Modus 0 auf Spalte 22 gesetzt?

8. Mit welcher Anweisung wird der Cursor abgeschaltet?

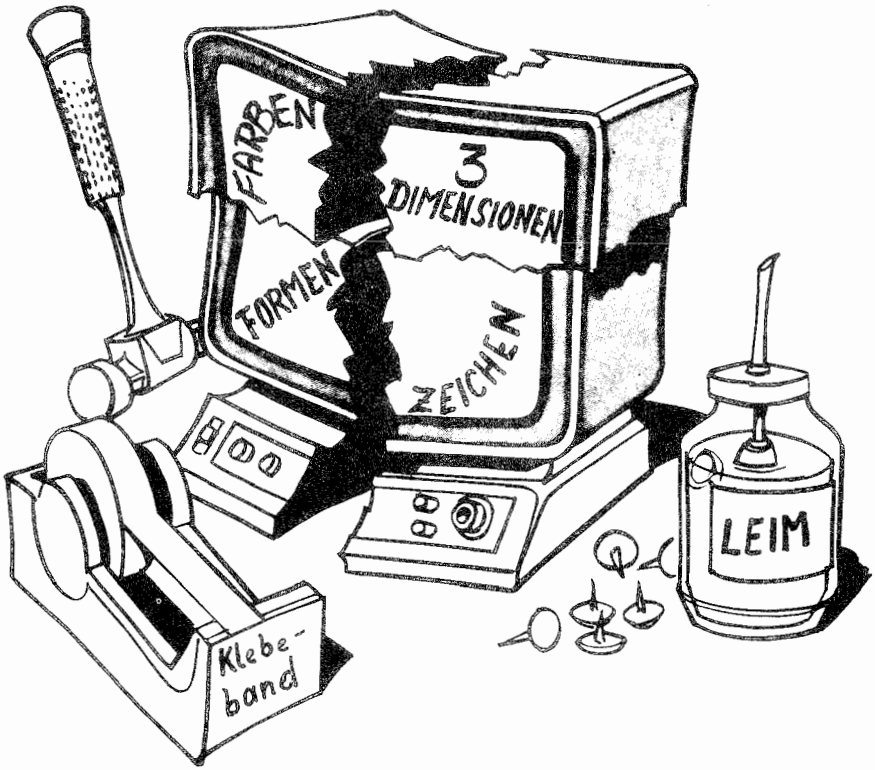
9. Was für ein Bild wird das folgende Programm produzieren?

```
10 PRINT "DIES";" IST"
20 PRINT "EIN";
30 PRINT " TEST"
```

KAPITEL 6

Ein fertiges Videobild

Es wird Zeit die einzelnen Teile zusammenzusetzen.



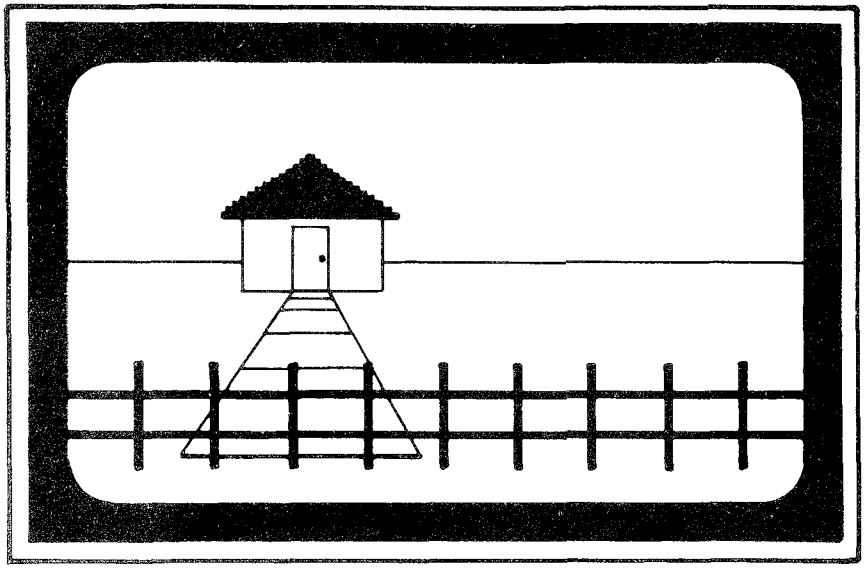
In diesem Kapitel werden Sie lernen, wie Sie:

- eine Geschichte in ein Bild übertragen
- verschiedene optische Effekte erzeugen
- ein Videokunstwerk entwerfen

In den ersten fünf Kapiteln haben wir uns mit dem Entwurf und der BASIC Programmierung von Graphiken auf dem ATARI beschäftigt. Sie sollten jetzt ein Gefühl für die Anwendung von Farben und Zeichen, sowie den Entwurf von dreidimensionalen Figuren entwickelt haben. In diesem Kapitel werden wir versuchen, alle in den Kapiteln 1 bis 5 präsentierten Ideen zu einem einzigen fertigen Videobild zu kombinieren. Beginnen wir mit einer Geschichte.

Sie fühlen sich vollkommen entspannt, während Sie sich in Ihren bequemen Sessel zurücklehnen. Während Ihre Gedanken so umherirren, sind Sie auf einmal weit, weit weg von zu Hause. Sie spazieren auf einem offenen Feld. Es ist ein später Herbstnachmittag und die frische Brise erinnert Sie daran, daß der Sommer schon lange vorbei ist. Der Boden sieht karg aus nach der Ernte. In einiger Entfernung sehen Sie etwas, daß Sie an ein Gebäude erinnert. Während Sie darauf zugehen, nimmt der Umriß des Gebäudes langsam Konturen an. Es scheint ein Haus zu sein. Die Vorderseite ist plan und besitzt keine Fenster. Die einzige Eintrittsmögichkeit scheint die große Tür im Mittelpunkt zu bieten.
Das Haus scheint schon vor Jahren verlassen worden zu sein. Sollten Sie zu der Tür gehen? Etwas scheint nicht zu stimmen. Der breite Gehweg, der zu dem Haus führt, ist in unglaublich perfekte Rechtecke unterteilt. Aber der Zaun zwischen Ihnen und diesem Weg hat kein Tor. Es ist, als wenn eine Einladung ausgesprochen wurde, Sie aber Angst haben sie anzunehmen.

Gut, kehren wir zur Realität zurück. Dies war nur eine Geschichte, aber es war gleichzeitig eine Beschreibung des Videobildes, das wir nun gemeinsam erstellen wollen.
Die Vorstellung ist das erste und wichtigste Glied einer Videokomposition. Diese Geschichte soll Ihnen helfen, sich ein einsames Haus auf einem Feld an einem Hebstnachmittag vorzustellen.
Es wird Zeit, daß wir unsere Gedanken in Richtung Bild drehen. Es folgt eine visuelle Interpretation der kurzen Geschichte.



Es muß nicht dieselbe Interpretation sein, die Sie vielleicht gezeichnet hätten, aber lassen Sie uns diese Zeichnung verwenden, um uns bei der Erstellung des Programmes zu inspirieren.
Wir müssen den einzelnen Teilen der Zeichnung Farben zuordnen. Wir wollen annehmen, daß uns kein GTIA-Chip zur Verfügung steht und, daß wir deshalb immer nur vier Farben zur Verfügung haben.
Der Himmel ist klar und wirkt sehr fern. Lassen Sie uns dafür eine kalte Farbe, wie hellblau, wählen. Da es Herbst und Spätnachmittags ist wollen wir für den Boden ein helles braun oder orange nehmen. Wir haben jetzt zwei Farben verwendet, es bleiben also noch zwei weitere. Wir könnten das Haus und den Zaun in derselben Farbe darstellen (vielleicht weiß) und den Gehweg und das Dach mit der letzten Farbe (vielleicht grau) färben. Nun haben wir genug Informationen, um mit unserem Programm zu beginnen. Wir werden später zurückkehren, um die Farben zu verändern und die Ergebnisse zu beobachten.

Programmabschnitte und ihre Anordnung

Bevor wir mit dem Programmieren beginnen, wollen wir unsere Zeichnung auf einem Arbeitsblatt skizzieren. Welchen GRAPHIK-Modus sollen wir verwenden?

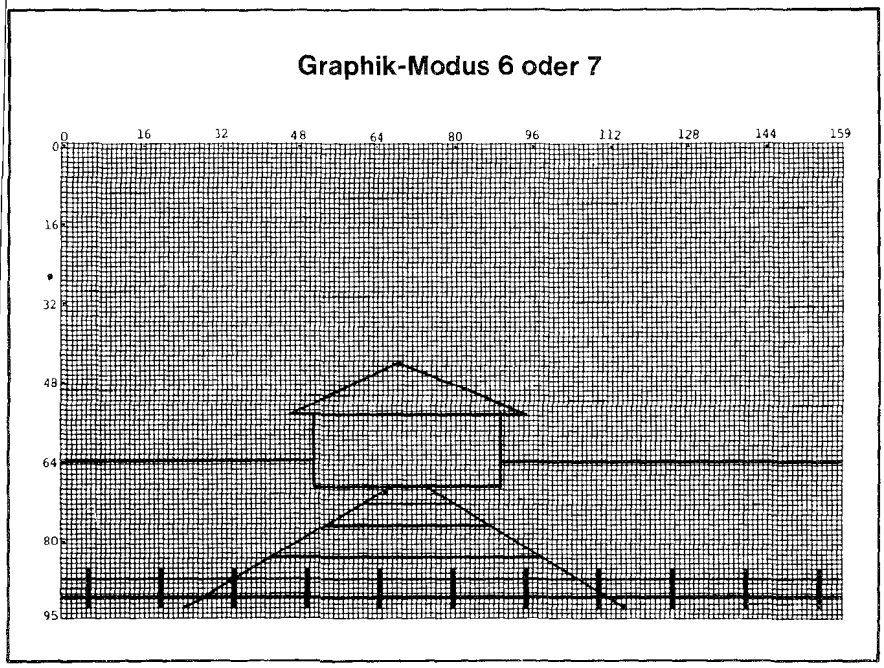


BILD 6-1: Entwurf einer Hausszene auf einem Arbeitsblatt im Modus 7.

Da wir vier Farben verwenden wollen, müssen wir einen vierfarbigen GRAPHIK-Modus wie 3, 5 oder 7 verwenden. Modus 7 bietet die höchste Auflösung. Die auf einem Arbeitsblatt im Modus 7 angefertigte Skizze könnte, wie in Bild 6-1 aussehen.
Lassen Sie uns das Programm stückchenweise zusammensetzen, und dabei immer derartige Dinge wie Farbe, Größe und Ausgewogenheit im Blick behalten. Damit das Gesamtwerk leichter zu verstehen ist, werden wir es in einzelne Abschnitte unterteilen. So wird zum Beispiel die Routine, mit der das Haus gezeichnet wird, ein Abschnitt sein, die Routine für den Hintergrund ein anderer. Diese Abschnitte werden untereinander so unabhängig wie möglich sein, und unserem Programm eine Art Gliederung verleihen.
Bevor wir nun mit unserem Bild beginnen, wollen wir das ganze Programm gliedern. Wir können dazu das Format eines Programms verwenden.

Zeilen	Anmerkungen
100-190	Titel und Kommentare
200-290	Setze GRAPHIK-Modi, initialisiere Farben
1000-1900	Zeichne Hintergrund
2000-2900	Zeichne Haus
3000-3900	Zeichne Gehweg
4000-4900	Zeichne Zaun

Jeder Programmabschnitt wird in diesen entsprechenden Zeilen geschrieben werden.

Titel

Wir wollen das Programm mit ein paar REM-Anweisungen dokumentieren. Geben Sie diese Zeilen ein:

```
100 REM DAS HAUS - EINE VIDEO KOMPOSITION
110 REM GESCHRIEBEN AM VON
```

Setzen von Modus und Farbe

Im GRAPHIK-Modus 7 werden die Farbregister 0, 1, 2 und 4 verwendet. Wir können den Himmel mit dem Farbregister 4, für die Hintergrundfarbe, darstellen. Es macht keinen Unterschied, welche Farben den anderen Registern zugeordnet werden. Fügen Sie diese Zeilen zu unserem Programm hinzu:

```
200 REM SETZE GRAPHIKSCHIRM
210 GRAPHICS 7+16
220 SETCOLOR 0,1,10
230 SETCOLOR 1,0,14
240 SETCOLOR 2,0,6
250 SETCOLOR 4,9,10
```

Das Farbregister 0 wird für den Boden verwendet, Register 1 für das Haus und den Zaun, Register 2 für das Dach und den Gehweg und Register 4 für den Himmel.

Der Hintergrund

Wir können mit dem Zeichnen beginnen. Unsere erste Aufgabe wird es sein den Hintergrund darzustellen. Die Farbe des Himmels wurde in Zeile 250 festgelegt. Die Grundfarbe wird durch das Register 0 bestimmt, so daß die Linien, die wir für den Hintergrund ziehen, die COLOR 1-Anweisung verwenden müssen. Wir können die Fläche ausfüllen, indem wir aneinanderliegende horizontale Linien ziehen, wie wir das bereits in Kapitel 3 getan haben. Der Abschnitt könnte so aussehen. Geben Sie die Zeilen ein:

```
1000 REM HINTERGRUNDABSCHNITT
1010 COLOR 1
1020 FOR Y=65 TO 95
1030 PLOT 0,Y:DRAWTO 159,Y
1040 NEXT Y
```

Fügen Sie diese Zeile ans Ende des Programms, so daß wir es laufen lassen können, ohne das die Darstellung wieder in den GRAPHIK-Modus 0 zurückfällt.

```
9000 GOTO 9000
```

Starten Sie das Programm. Haben Sie einen blauen Himmel und einen orangen Boden?

Das Haus

Wir können jetzt das Haus zeichnen. Als erstes das Dach. Wir können dazu den Farbbefehl, wie in Kapitel 4 verwenden. Geben Sie diese Zeilen ein:

```
2000 REM HAUSABSCHNITT
2100 REM HAUSDACH
2110 COLOR 3
2115 POKE 765,3
2120 PLOT 95,55:DRAWTO 70,45
2130 POSITION 45,55
2140 XIO 18,#6,0,0,"S:"
```

Wir können das Haus mit derselben Technik zeichnen, die wir schon für den Boden verwendet haben. Geben Sie diese Zeilen ein:

```
2200 REM HAUS
2210 COLOR 2
2220 FOR Y=56 TO 70
2230 PLOT 50,Y:DRAWTO 90,Y
2240 NEXT Y
```

Starten Sie das Programm. Hat Ihr Haus nun ein graues Dach und ein weißes Fundament? Nimmt Ihre Zeichnung langsam Form an?

Der Gehweg

Wir wissen bereits alles über das Zeichnen von Gehwegen. Wir können die Endpunkte jeder Gehweglinie in DATA-Anweisungen angeben, wie wir das bereits in Kapitel 4 getan haben.Der Gehwegabschnitt sieht folgendermaßen aus.

```
3000 REM GEHWEGABSCHNITT
3010 COLOR 3
3020 READ LINES
3030 FOR L=1 TO LINES
3040 READ P1,P2,D1,D2
3050 PLOT P1,P2:DRAWTO D1,D2
3060 NEXT L
3090 DATA 5
3100 DATA 65,71,25,93,75,71,115,93,61,73,79,73
3110 DATA 53,78,86,78,41,85,99,85
```

Fügen Sie die Zeilen zu Ihrem Programm hinzu und starten Sie es. Erfüllt es Ihre Erwartungen?

Der Zaun

Der Zaun besteht aus zwei horizontalen und mehreren vertikalen Linien. Lassen Sie uns zuerst die beiden horizontalen Linien zeichnen und dann die vertikalen Linien dazwischenfügen, indem wir eine Schleife verwenden. Hier ist der Programmabschnitt.

```
4000 REM ZAUNABSCHNITT
4010 COLOR 2
4015 REM HORIZONTALE ZAUNLINIEN
4020 PLOT 0,89:DRAWTO 159,89
4030 PLOT 0,91:DRAWTO 159,91
4035 REM VERTIKALE ZAUNLINIEN
4040 FOR X=5 TO 155 STEP 15
4050 PLOT X,87:DRAWTO X,93
4060 NEXT X
```

Fügen Sie diese Zeilen zu Ihrem Programm. Starten Sie es. Sieht der Bildschirm jetzt genauso aus, wie der Entwurf auf dem Arbeitsblatt im Modus 7?

Einige Extras

Die Tür

Die Zeichnung ist vollständig, aber irgendetwas scheint zu fehlen. Ach, wir haben die Tür vergessen. Lassen Sie uns wieder in den Hausabschnitt des Programmes gehen und die folgenden Zeilen eingeben.

```
2300 REM HAUSTUER
2310 COLOR 3
2320 PLOT 65,70:DRAWTO 65,60:DRAWTO 75,60:DRAWTO 75,70
```

Starten Sie es. Macht es jetzt einen besseren Eindruck?

Ein Vogel

Der Himmel sieht so leer aus. Lassen Sie uns einen Vogel in den Himmel setzen. Fügen Sie diese Zeilen hinzu:

```
5000 REM VOEGEL
5010 COLOR 2
5020 PLOT 110,30:DRAWTO 114,27:DRAWTO 118,31
5025 DRAWTO 122,27:DRAWTO 126,30
```

Probieren Sie das Programm erneut. Erscheint es jetzt etwas ausgewogener?

Ein Rahmen

Die Zeichnung sieht hübsch genug aus, um sie einzurahmen. Fügen Sie diese Zeilen hinzu:

```
6000 REM EIN RAHMEN
6005 COLOR 3
6010 FOR X=0 TO 1
6020 PLOT X,X:DRAWTO 159-X,X:DRAWTO 159-X,95-X
6025 DRAWTO X,95:DRAWTO X,X
6030 NEXT X
```

Starten Sie es. Sind Sie zufrieden? Wenn ja, Gratulation. Sie haben soeben Ihr erstes Videokunstwerk vollendet. Hier ist ein Listing des vollständigen Programmes:

```
100 REM DAS HAUS - EINE VIDEOKOMPOSITION
110 REM GESCHRIEBEN AM VON
200 REM SETZE GRAPHIKSCHIRM
210 GRAPHICS 7+16
220 SETCOLOR 0,1,10
230 SETCOLOR 1,0,14
240 SETCOLOR 2,0,6
250 SETCOLOR 4,9,10
1000 REM HINTERGRUNDABSCHNITT
1010 COLOR1
1020 FOR Y=65 TO 95
1030 PLOT 0,Y:DRAWTO 159,Y
1040 NEXT Y
2000 REM HAUSABSCHNITT
2100 REM HAUSDACH
2110 COLOR 3
2115 POKE 765,3
2120 PLOT 95,55:DRAWTO 70,45
2130 POSITION 45,55
2140 XIO 18,#6,0,0,"S:"
2200 REM HAUS
2210 COLOR 2
2220 FOR Y=56 TO 70
2230 PLOT 50,Y:DRAWTO 90,Y
2240 NEXT Y
2300 REM HAUSTUER
2310 COLOR 3
```

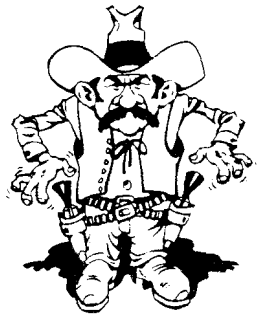
```
2320 PLOT 65,70:DRAWTO 65,60:DRAWTO 75,60:DRAWTO 75,70
3000 REM GEHWEGABSCHNITT
3010 COLOR 3
3020 READ LINES
3030 FOR L=1 TO LINES
3040 READ P1,P2,D1,D2
3050 PLOT P1,P2:DRAWTO D1,D2
3060 NEXT L
3090 DATA 5
3100 DATA 65,71,25,93,75,71,115,93,61,73,79,73
3110 DATA 53,78,86,78,41,85,99,85
4000 REM ZAUNABSCHNITT
4010 COLOR 2
4015 REM HORIZONTALE ZAUNLINIEN
4020 PLOT 0,89:DRAWTO 159,89
4030 PLOT 0,91:DRAWTO 159,91
4035 REM VERTIKALE ZAUNLINIEN
4040 FOR X=5 TO 155 STEP 15
4050 PLOT X,87:DRAWTO X,93
4060 NEXT X
5000 REM VOEGEL
5010 COLOR 2
5020 PLOT 110,30:DRAWTO 114,27:DRAWTO 118,31
5025 DRAWTO 122,27:DRAWTO 126,30
6000 REM EIN RAHMEN
6005 COLOR 3
6010 FOR X=0 TO 1
6020 PLOT X,X:DRAWTO 159-X,X:DRAWTO 159-X,95-X
6025 DRAWTO X,95-X:DRAWTO X,X
6030 NEXT X
9000 GOTO 9000
```

Experiment

Vielleicht möchten Sie jetzt beginnen ein bißchen mit den Farben herumzuexperimentieren. Wir werden ein Beispiel zeigen und den Rest der Experimente Ihnen überlassen. Verändern Sie diese Zeilen:

```
220 SETCOLOR 0,0,10
250 SETCOLOR 4,0,0
```

Beachten Sie das alle Farben neutral sind. Ein interessanter Effekt, finden Sie nicht auch?



Aufgabe #11

Schreiben Sie eine Routine mit der Sie eine Wolke auf den oberen linken Teil Ihrer Videokomposition zaubern. Zeile 7000 wäre der ideale Platz für diesen Programmabschnitt.

Eine Zusammenfassung

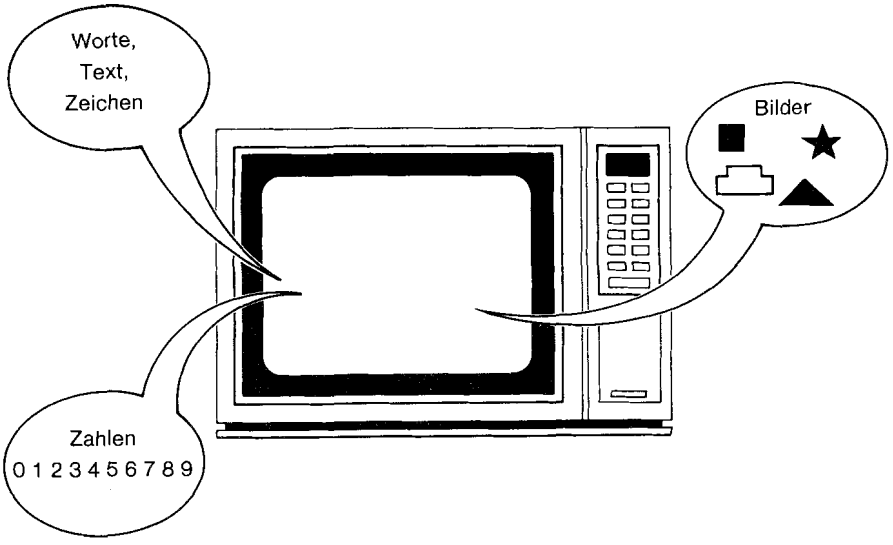
Die Kapitel 1 bis 6 waren eine einfache Einführung in die Erstellung und Programmierung von Graphiken in der BASIC Sprache. Sie waren für den Anfänger mit nur geringer Programmiererfahrung gedacht. Schon mit sehr einfachen BASIC Programmen lassen sich viele graphische Darstellungen erstellen.
Die Fähigkeit die Programme zu schreiben ist nur ein Teil, der den guten Videokünstler ausmacht. Auch das Gefühl für Farben, Formen, Ausgewogenheit und Stimmungen spielt eine große Rolle.
Obwohl schon sehr schöne Graphiken mit grundlegenden BASIC Programmen erstellt werden können, werden die wirklich tollen Graphiken auf dem ATARI Computer erst durch aufwendige Programmierung möglich. In Teil Zwei werden Sie die Graphik Hardware des ATARI Computers kennenlernen. Die Ideen sind allerdings wesentlich schwerer zu verstehen, als die in Teil Eins. Und wenn Sie nicht einigermaßen mit der Programmierung vertraut sind, könnten Verständnisprobleme auftauchen. Aber das sollen Sie selbst beurteilen. Wenn Sie sich nicht sicher sind, ob Sie es schaffen können, probieren Sie es einfach aus. Wir wünschen Ihnen viel Glück!

TEIL ZWEI

KAPITEL 7

Bilder und Worte – verknüpfen von Modi

Bereiten Sie sich darauf vor Bilder mit Worten zu verbinden, indem Sie verschiedene GRAPHIK-Modi miteinander verknüpfen.



In diesem Kapitel werden Sie lernen, wie Sie:

- die Funktionsweise eines Fernsehers erläutern
- verschiedene Modi miteinander verknüpfen
- ein Display List erstellen
- verschiedene Darstellungstechniken mit gemischten Modi verwenden.

Worum geht es bei der Verknüpfung von Modi? Hier ist die Antwort. Verknüpfen der Modi bedeutet, daß mehr als ein GRAPHIK-Modus auf einem Bildschirm dargestellt wird. In den Kapiteln 1 bis 6 haben wir eine Anzahl verschiedener GRAPHIK-Modi verwendet. Aber immer nur einen Modus zur gleichen Zeit. Die einzige Ausnahme hierbei war, als wir mit geteilten Bildschirm-Modi arbeiteten. In diesen Fällen benutzten wir ein Textfenster im GRAPHIK-Modus 0 zusammen mit einem anderen GRAPHIK-Modus.
Aber stellen Sie sich vor, Sie wollen auf der oberen Hälfte des Bildschirms etwas im GRAPHIK-Modus 2 schreiben, auf dem unteren Teil desselben Bildes aber eine Zeichnung im GRAPHIK-Modus 7 anfertigen. Das würde die Verknüpfung der GRAPHIK-Modi 2 und 7 erfordern.
Lassen Sie uns dies an einem kleinen Programm ausprobieren. Geben Sie die Zeilen in den Computer ein und starten Sie das Programm:

```
10  REM DEMONSTRATION VERKNUEPFTER MODI
20  GRAPHICS 7+16
30  DLIST=PEEK(560)+PEEK(561)*256
40  POKE DLIST+3,71:POKE DLIST+6,7
60  POKE DLIST+92,65
70  POKE DLIST+93,PEEK(560)
80  POKE DLIST+94,PEEK(561)
90  POKE 87,2:POSITION 8,0:PRINT #6;"TEST"
100 POKE 87,7:COLOR 2
110 FOR Y=10 TO 60 STEP 3:PLOT 5,Y:DRAWTO 155,Y:NEXT Y
200 GOTO 200
```

Sie sollten jetzt das Wort "TEST" im GRAPHIK-Modus 2 auf dem oberen Teil des Bildschirms sehen. Der Rest des Bildschirms ist mit horizontalen Streifen im GRAPHIK-Modus 7 gefüllt. Ein Blick auf die Technik wird Ihnen die Vorgänge erläutern.

Entstehung eines Fernsehbildes

Damit Sie dieses Kapitel wirklich verstehen können, ist es notwendig einen kurzen Blick auf die Funktionsweise Ihres Fernsehers und Ihres ATARI Computers zu werfen. Als erstes müssen wir verstehen, wie ein Fernseher die Figuren auf dem Bildschirm darstellt. Das folgende ist eine vereinfachte Erklärung.

Die Bildröhre Ihres Fernsehers setzt sich aus verschiedenen Komponenten zusammen. Am Ende dieser Röhre befindet sich ein Kathodenstrahler. Dieser Strahler "schießt" einen feinen Elektronenstrahl auf die Vorderseite der Bildröhre. Im weiteren wollen wir diese Vorderseite Bildschirm nennen. Der Bildschirm besitzt eine Phosphorummantelung. Diese Hülle leuchtet an den Stellen auf an denen sie der Kathodenstrahl trifft. Bild 7-1 gibt eine einfache Skizze einer typischen Bildröhre wieder.
Da der Kathodenstrahl nur einen winzigen Durchmesser besitzt, würde nur ein kleiner Punkt auf dem Bildschirm aufleuchten. Aber durch Magnetspulen die an der Seite der Röhre angebracht sind, wird der Strahl so abgelenkt, daß er den ganzen Bildschirm streifen kann.

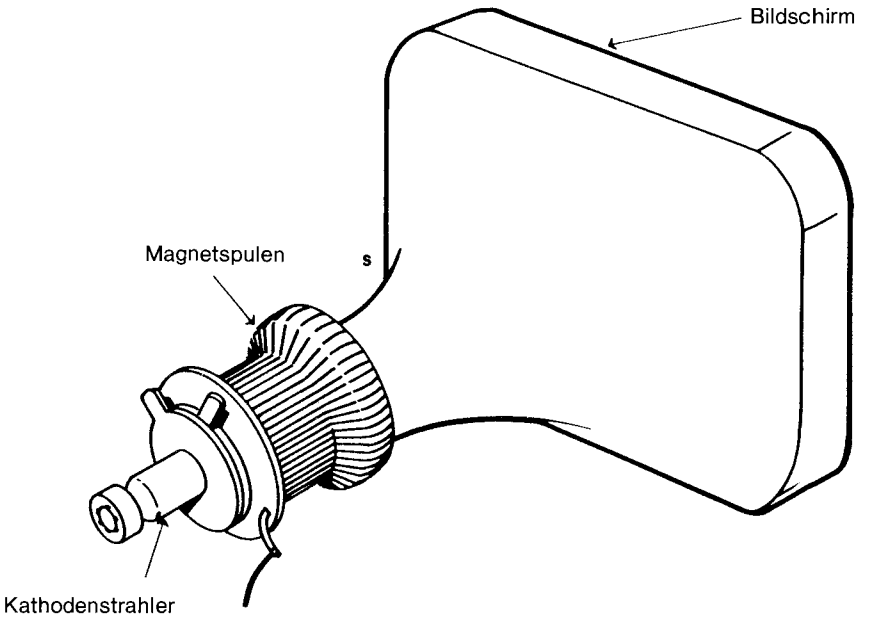


BILD 7-1: Typische Fernsehbildröhre.

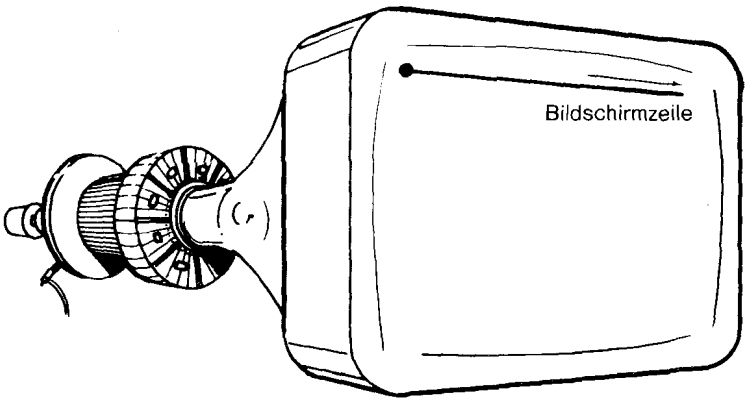
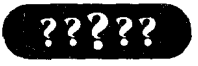


BILD 7-2: Scan Line (Bildschirmzeile).

Der Kathodenstrahl streift den Bildschirm in einer sehr systematischen Art und Weise. Er beginnt in der oberen linken Ecke des Bildschirms und fährt dann rasend schnell in die rechte obere Ecke des Bildschirms. Durch diesen Vorgang entsteht eine sogenannte Scan Line (Bildschirmzeile, siehe auch Bild 7-1). Der Strahl wird dann abgeschaltet und kehrt auf die linke Seite des Bildschirms zurück, allerdings eine Zeile tiefer. Der Kathodenstrahl wird wieder eingeschaltet und rast nun in der zweiten Zeile nach rechts. Dieser Prozess wird solange wiederholt, bis der ganze Bildschirm bestrahlt worden ist, Für unsere Zwecke gibt es 262 Scan Lines. Nachdem der Strahl die untere rechte Ecke des Bildschirms erreicht hat, wird er wieder abgeschaltet und kehrt in die obere linke Ecke des Bildschirms zurück. Der ganze Prozess wiederholt sich 60 Mal in einer Sekunde. Auf Grund dieser ungeheuren Geschwindigkeit, kann unser Auge keine Veränderung in der Leuchtintensität des Phosphormantels feststellen. Für uns sieht es aus, als wenn der Kathodenstrahl immer den ganzen Bildschirm gleichzeitg bestrahlt. Diese Technik der Bilddarstellund auf einem Fernsehschirm wird Rastertechnik genannt.

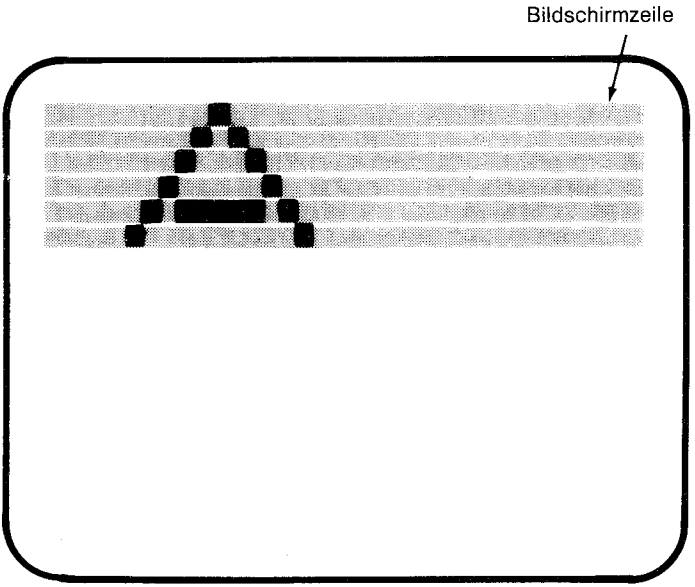


Es ist wieder Zeit für eine Frage. Wie lange braucht der Kathodenstrahl, um einmal den ganzen Bildschirm zu bestrahlen?

Antwort #1_____

#1. 1/60 Sekunden

Nur durch das Bestrahlen würden allerdings noch keine Figuren auf dem Bildschirm entstehen. Die Figuren werden erzeugt, indem die Intensität des Strahls verändert wird, während er über die Zeilen huscht. Der Buchstabe A würde zum Beispiel dadurch Zeilenweise erzeugt, daß der Strahl immer nur an bestimmten Stellen in einer Zeile aufflackert.



So können umfassende Figuren mit verschiedenen Helligkeiten geformt werden. Bedenken Sie bitte, daß der Prozess in Wirklichkeit etwas komplizierter ist, als hier beschrieben. Dies gilt besonders für Farbbilder.

ATARI Video Chip

Die auf dem Bildschirm dargestellten Informationen müssen vorher aber auf irgendeine Weise erzeugt werden. Lassen Sie uns einen Blick auf den ATARI Videochip und seine Arbeitsweise werfen.

Video Hardware

Für die Darstellung eines vom Computer erzeugten Videobildes sind verschiedene Hardwarekomponenten des ATARI Computers entweder direkt oder indirekt zuständig. Eine dieser Komponenten ist der bereits in Kapitel 1 erwähnte CTIA oder GTIA Chip. Die Hauptaufgabe dieses Chips ist es, die Digitalimpulse des Computers derart aufzubereiten, daß sie als Signale zum Fernseher gesendet werden können.
Ein anderes Chip namens ANTIC wiederum ist zuständig für die Informationen die dem CTIA oder GTIA zugesendet werden. ANTIC ist eigentlich ein Mikroprozessor. Es kann dazu programmiert werden, die auf dem Bildschirm verwendeten GRAPHIK-Modi zu wechseln. Um dieses Chip geht es in diesem Kapitel.

Video Software (Display List)

Rufen Sie sich ins Gedächtnis zurück, daß Fernsehbilder durch die Verwendung von Scan Lines entstehen und ein ganzer Bildschirm normalerweise aus 262 solcher Zeilen besteht. Die Scan Lines beginnen über dem oberen Ende des Bildschirms und enden unterhalb des unteren Endes. Dies wird Overscan (Überschreiben) genannt. Dies ist normal für Fernseher und verhindert das die häßlichen Bildränder gesehen werden. Wenn Sie einen Fernseher mit Ihrem Computer verwenden, ist es wichtig, daß keine Informationen bei der Darstellung verloren gehen. Daher verwendet das ANTIC nur 192 der 262 zur Verfügung stehenden Zeilen und deshalb erscheint auch der Rand auf dem Bildschirm. Trotzdem können Sie mehr Zeilen verwenden, wie wir später noch feststellen werden. Im GRAPHIK-Modus 0 werden 192 Zeilen benutzt. Die Ränder über und unter des Darstellungsrechtecks beherbergen die fehlenden 70 Zeilen.

Es können immer mehrere Scan Lines verwendet werden, um ein Pixel bzw. ein Zeichen auf dem Bildschirm zu produzieren. So benötigt ein Pixel im GRAPHIK-Modus 7 zum Beispiel 2 Scan Lines, dasselbe Pixel im GRAPHIK-Modus 3 allerdings 8. Eine Gruppe Scan Lines wird eine Mode Line genannt. Eine GRAPHIK 3 Mode Line besteht daher aus 8 Scan Lines.
Wie bereits erwähnt kann ANTIC programmiert werden. Das Programm mit dem ANTIC programmiert wird, nennt man Display List. Die Display List besteht aus einer Reihe von Anweisungen, die bestimmen welche GRAPHIK-Modi auf dem Bildschirm dargestellt werden.

Die Display List ist im RAM des Computers gespeichert und kann verändert werden.
Die ATARI BASIC Sprache ist so konzipiert, daß sie automatisch eine Display List erzeugt. Dies wird durch die Ausführung einer Graphikanweisung, wie GRAPHICS 0, erreicht. Bevor wir nun aber verschiedene Modi miteinander verknüpfen können, müssen wir eine eigene Display List erstellen oder aber eine bereits existierende BASIC Display List entsprechend ändern.
Bevor wir ein derartiges Programm angehen, wollen wir erst noch einen Blick auf ein Programm werfen, das von dem Computer auf die BASIC Anweisung GRAPHICS 0 hin erstellt wurde. Programmieren und starten Sie das folgende Programm:

```
10 REM BLICK AUF GRAPHICS 0 DISPLAY LIST
15 GRAPHICS 0
20 DLIST=PEEK(560)+PEEK(561)*256
30 FOR M=DLIST TO DLIST+31
40 PRINT PEEK(M)
50 NEXT M
```

Ich weiß, es ging alles so schnell, daß der größte Teil wieder nach oben aus dem Bildschirm gewandert ist. Da die Liste leichter verständlich wird, wenn sie vertikal gedruckt ist, wollen wir uns das unten aufgeführte Listing einmal näher betrachten. Das Listing auf dem Bildschirm sieht ungefähr gleich aus, die Kommentare wurden allerdings erst später als Verständnishilfen beigefügt.

```
112
112   ---> Acht Leerzeilen (dreimal)
112
66    ---> LMS-Anweisung
64    ---> Adresse der Bilddarstellungs RAM
156          (Diese Werte werden allerdings anders sein, wenn Sie
2            keine 48K RAM in Ihrem Computer haben)
2
2
2
2
2
2    ---> Modus Bytes für GRAPHICS 0
2
2
2
2
2
```

```
2  ┐
2  │
2  │
2  │
2  │
2  │
2  ├──Modus Bytes für GRAPHICS 0
2  │
2  │
2  │
2  │
2  │
2  ┘
65 ──Sprunganweisung
32  ┐ Adresse der Display List RAM
15  ├─(Diese Werte werden anders sein, wenn Sie keine 48K RAM in
    │  Ihrem Computer haben.)
```

Lassen Sie uns einmal erforschen, was das alles zu bedeuten hat. Jeder der hier aufgeführten Werte, repräsentiert eine Anweisung in der Maschinensprache oder eine Speicheradresse. ANTIC erzeugt das Bildschirmformat, indem er diese Werte verwendet, um die entsprechenden Mode Lines zu erzeugen. Beachten Sie, daß die gezeigten Werte Dezimaldarstellungen sind. Beachten Sie auch, daß ANTIC, als Mikroprozessor, seinen eigenen Befehlssatz hat.

Jede der ersten drei Anweisungen, veranlassen ANTIC acht leere Scan Lines zu schreiben. Sie werden Leeranweisungen genannt. Damit wird dem Overscan vorgebeugt und das Computerbild kommt in lesbare Bereiche.

Die nächste Anweisung ist die LMS (Load Memory Scan) Anweisung. Es veranlaßt ANTIC die nächsten beiden Werte (Bytes) einzuladen. Die folgenden zwei Werte sind dann die Adresse, an denen das Bilddarstellungs-RAM beginnt. Die LMS-Anweisung teilt ANTIC auch mit, welcher Modus für die erste Mode Line verwendet werden soll. Mit dieser Anweisung haben wir noch weitere Möglichkeiten, die wir später erläutern werden.

Nach den Werten, die die Bildschirm RAM Position angaben, folgen die Werte, die den jeweils auf dem Bildschirm darzustellenden Mode Lines entsprechen. Hier vertritt der Wert 2 die GRAPHICS 0 Mode Lines. Da es im GRAPHIK-Modus 0 genau 24 Mode Lines gibt, werden zusätzlich zu der LMS-Anweisung, die die erste Mode Line spezifiziert, weitere 23 dieser Werte gebraucht.

Die letzte Anweisung schließlich, jagt unser Display List Programm in eine Endlosschleife. Sie wird die Sprunganweisung (JMP) genannt. Sie veranlaßt ANTIC an den Beginn der Display List zurückzukehren.

Dieser letzte Schritt schließt das Display List Programm ab. Da ein Bildschirm 60 Mal in der Sekunde bestrahlt wird, wird dieses Programm auch 60 Mal in der Sekunde ausgeführt. ANTIC wird fortfahren, die in diesem Programm gegebenen Werte zu verwenden bis sie verändert werden. Die Werte können verändert werden, indem man entweder eine Graphikanweisung in BASIC ausführen läßt oder aber einzelne Teile der Display List entweder durch BASIC oder aber durch Maschinenroutinen ändert.

Erstellen einer eigenen Display List

Bevor Sie Ihre eigene Display List erstellen, müssen Sie sich erst einmal darüber im klaren sein, wie der Bildschirm aussehen soll. Das Display List Arbeitsblatt in Anhang A sollte hierbei verwendet werden.
Lassen Sie uns als Beispiel einen Bildschirm entwerfen, dessen oberste Zeilen im GRAPHIK-Modus 2 sind, worauf dann vier Zeilen im GRAPHIK-Modus 0 folgen (siehe Bild 7-3). Der Rest des Bildschirms soll im GRAPHIK-Modus 3 sein. Wir werden dieses Beispiel während der gesamten Erstellung der Display List verwenden.
Die in der Tabelle 7-1 enthaltenen Informationen werden benötigt, sobald die Display List erstellt wird. Die Tabelle wird hier nicht näher erläutert werden, wird aber für den Rest des Kapitels als Referenz verwendet.
Wir werden die Display List in der BASIC Sprache programmieren.

1. Unsere erste Aufgabe besteht darin, die gewünschte Anzahl der Mode Lines mit der erforderlichen Anzahl der Scan Lines zu koordinieren. Sehen Sie in Tabelle 7-1 nach. Dies wurde in unserem Beispiel bereits getan. Bei diesem Arbeitsgang sollten Sie darauf achten, daß die Anzahl der Mode Lines genau 192 Scan Lines erfordern sollte. Die folgende Tabelle gibt die in unserem Beispiel benötigten Mode Lines und Scan Lines an.

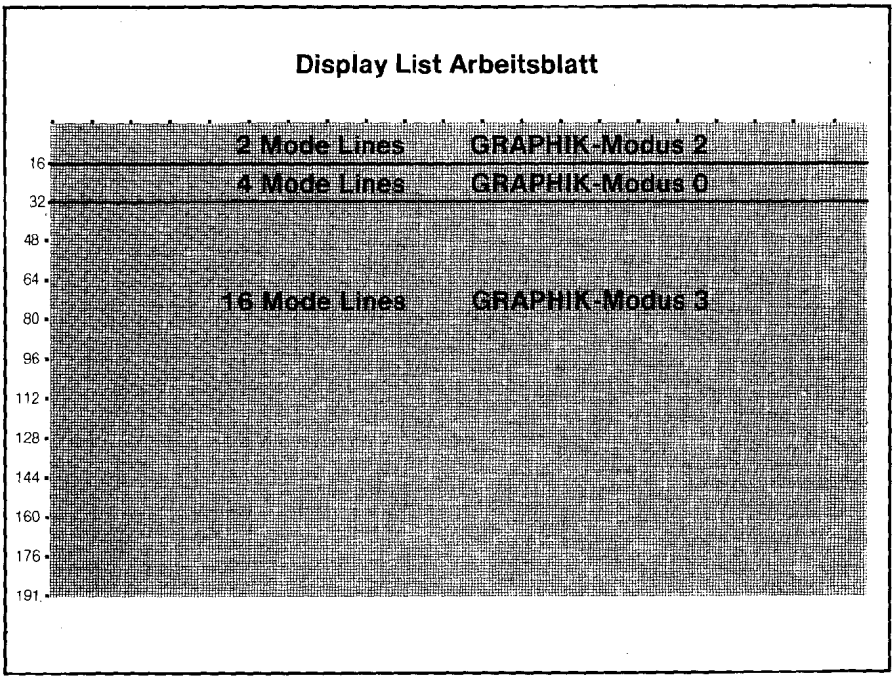


BILD 7-3: Entwurf der Mode Lines.

TABELLE 7-1: ANTIC Mode Lines Erfordernisse

ANTIC Mode	BASIC Mode	No. of Colors	Scan Lines/ Mode Line	Pixels/ Mode Line	Bytes/ Line	Bytes/ Screen
2	0	2	8	40	40	960
3	ohne	2	10	40	40	760
4	ohne	4	8	40	40	960
5	ohne	4	16	40	40	480
6	1	5	8	20	20	480
7	2	5	16	20	20	240
8	3	4	8	40	10	240
9	4	2	4	80	10	480
10	5	4	4	80	20	960
11	6	2	2	160	20	1920
12	ohne	2	1	160	20	3840
13	7	4	2	160	40	3840
14	ohne	4	1	160	40	7680
15	8	2	1	320	40	7680

Mit Genehmigung der Atari, Inc. nachgedruckt aus dem De Re Atari Guide

BASIC MODE	ZAHL DER MODE LINES		ZAHL DER SCAN LINES PRO MODE LINE		SUMME DER SCAN LINES
2	2	X	16	=	32
0	4	X	8	=	32
3	16	X	8	=	128

					192

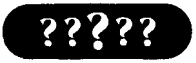
Testen Sie ob Sie den Unterschied zwischen Mode Line und Scan Line verstanden haben.

Wieviele Scan Lines würden für drei GRAPHICS % Mode Lines benötigt werden?

Antwort #2_____

2. Der nächste Schritt besteht darin, für jeden der gewünschten BASIC Modi, das entsprechenden ANTIC Mode Byte zu bestimmen. Sehen Sie in Tabelle 7-1 nach. Achten Sie darauf nicht den ANTIC Modus mit dem BASIC Modus durcheinander zu bringen.

BASIC Mode	ANTIC Mode Byte
2	7
0	2
3	8



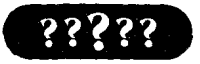
Wie lautet das ANTIC Mode Byte für den BASIC Mode 7?

Antwort #3_____

#2. 12 Scanlinien #3. 13

5. Legen Sie die LMS-Anweisung fest. Diese Anweisung wird gewählt, indem Bit 6 des Modus Anweisungsbytes gesetzt wird. Da Bit 6 in einen Dezimalwert von 64 versetzt wird, nehmen wir den ANTIC- Modus-Byte-Wert für die oberste Mode Line des Bildschirms und addieren 64 hinzu. Das Byte für unsere erste BASIC Modus 2 Zeile, das die LMS-Anweisung beinhaltet ist daher

LMS option(64) + ANTIC Modus Byte (7) = 71



Welchen Wert hätte das LMS Byte, wenn die erste Mode Line des Bildschirms den GRAPHIK-Modus 7 hätte?

Antwort #4_____

Werfen wir einen Blick darauf, wie unsere Speicherkarte der Display List aussehen sollte.

```
CODE   ANMERKUNGEN
112    8 Leerzeilen
112    8 Leerzeilen
112    8 Leerzeilen
 71    Setzt LMS-Option und stellt ANTIC Modus 7 (BASIC  Modus2)
       dar.)
  ?    PEEK(88)   Speicheradresse
  ?    PEEK(89)   des Bildschirm RAM
 07    ANTIC Modus 7 (BASIC Modus 2)
 02
 02
 02    4 Zeilen im BASIC Modus 0
 02
 08
 08
 08
 08
 08
 08    16 Zeilen im BASIC Modus 3
 08
 08
 08
 08
 08
 08
 08
 08
 08
 08
 65    Springe zum Anfang der Display List
  ?    PEEK(560)  Speicheradresse des
  ?    PEEK(561)  Anfangs der Display List
```

#4.77

Beachten Sie, daß wir nicht in der Lage waren die Speicheradressen des Bildschirm-RAM und der Display List einzufügen. Das Arbeitssystem wird jedoch selbstständig diese Adressen mitverfolgen.
Die dezimalen Speicheradressen 88 und 89 repräsentieren das nieder- und höchstwertigste Byte der Bildschirm RAM Adresse. Die dezimalen Speicherpositionen 560 und 561 repräsentieren das nieder- und höchstwertige Byte der Display List-Adresse.

4. Von jetzt an brauchen Sie nur noch die POKE-Anweisung und ein BASIC-Programm, um Ihre eigene Display List zu erstellen. Die Routine könnte folgendermaßen aussehen:

```
10 REM BEISPIEL VERKNUEPFTER MODI
15 GRAPHICS 0
20 DLIST=PEEK(560)+PEEK(561)*256
30 REM  3*8 LEERZEILEN
40 POKE DLIST,112
50 POKE DLIST+1,112
60 POKE DLIST+2,112
70 REM SETZE LMS-OPTION
80 POKE DLIST+3,71
90 REM BILDSCHIRMSPEICHERPOSITION
100 POKE DLIST+4,PEEK(88)
110 POKE DLIST+5,PEEK(89)
120 REM EINE WEITERE MODUS 2 ZEILE
130 POKE DLIST+6,7
140 REM MODUS 0 ZEILEN
150 FOR X=7 TO 10:POKE DLIST+X,2:NEXT X
160 REM MODUS 3 ZEILEN
170 FOR X=11 TO 28:POKE DLIST+X,8:NEXT X
180 REM SETZE SPRINGANWEISUNG
190 POKE DLIST+29,65
200 REM ANFANG DER DISPLAY LIST
210 POKE DLIST+30,PEEK(560)
229 POKE DLIST+31,PEEK(561)
```

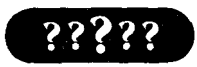
Beachten Sie, daß wir eine GRAPHICS 0 Anweisung in Zeile 15 des Programms verwendet haben. Dies war notwendig um dem BASIC mitzuteilen, das der Bildschirm geöffnet werden sollte und etwas Bildschirm RAM zur Verfügung sein mußte. GRAPHIK-Modus 0 wurde deshalb verwendet, da es von allen Modi den größten Speicherplatz des RAM reserviert.
Geben Sie die Zeilen schon einmal ein, starten Sie das Programm aber noch nicht. Warten Sie bis wir etwas auf den Bildschirm drucken können.

Schreiben auf einem Bildschirm mit verknüpften Modi

Das Schreiben auf einen Bildschirm mit verknüpften Modi (Mixed Modi) kann unter Umständen Schwierigkeiten bereiten. Versuche bestimmte Teile des Bildschirms zu PRINTen oder zu PLOTen, werden von BASIC falsch interpretiert. Da das BASIC nicht direkt, die Erstellung einer eigenen Display List ermöglicht hat, ist es auch nicht möglich die normalen Schreibvorgänge auf dem Bildschirm zu vollziehen. Es ist allerdings möglich ein paar der Speicheradressen zu ändern, um dieses Problem zu beheben. Zwei Dinge müssen erledigt werden, damit ohne Schwierigkeiten auf einem Bildschirm mit verknüpften Modi geschrieben werden kann.

1. Zuerst müssen Sie dem Computer angeben in welchem Modus Sie zu schreiben wünschen. Dies kann erledigt werden, indem Sie die Adresse 87 zusammen mit dem BASIC GRAPHIK-Modus POKEn. Bevor wir also auf den BASIC Modus 0 Teil des Bildschirms zu schreiben beginnen, verwenden wir zuerst die folgende Anweisung.

```
POKE 87,0
```



Welche Anweisung würden Sie verwenden, wenn Sie auf einen Bildschirmteil mit dem GRAPHIK-Modus 3 schreiben wollten?

Antwort #5_____

2. Der nächste Schritt ist, daß Sie dem Computer mitteilen müssen, wo der Anfang des Bildschirmspeichers ist, für das Textfenster in das Sie zu schreiben gedenken. Als erstes müssen wir die Bildschirmspeicheradresse der oberen linken Ecke des Bildschirms finden. Unter Verwendung der Bildschirm RAM Punkte, kann das errechnet werden.

```
SCREENMEM=PEEK(88) + PEEK(89)*256
```

Berechnen Sie jetzt den Betrag des Bildschirmspeichers zwischen der oberen linken Ecke des Bildschirms und der oberen linken Ecke Ihres Textfensters.

#5. POKE 87,3

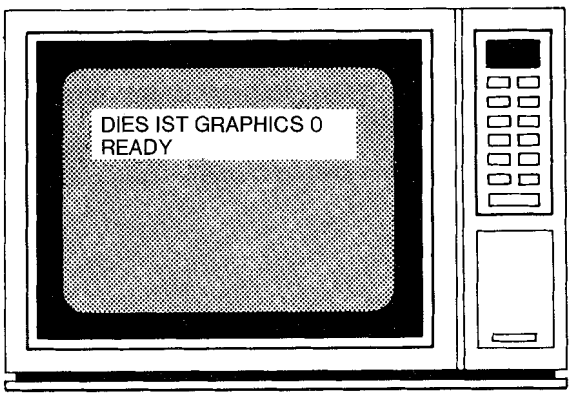
In diesem Beispiel existieren 2 Zeilen im BASIC Modus 2 über dem BASIC Modus 0 Fenster. Nach der Tabelle 7-1, beansprucht jede dieser Zeilen 20 Bytes RAM. Unser BASIC Modus 0 Fenster wird also bei SCREENMEM + 40 beginnen.
Sie müssen jetzt diesen neuen Wert in die Adressen 88 und 89 zurückPOKEn. Aber vorher müssen Sie ihn noch in niedrigst- und höchstwertige Bytes unterteilen. Sie könnten diese Anweisungen dazu verwenden die neuen niedrigst- und höchstwertigen Bytes der Bildschirmspeicheradressen zu berechnen und sie dann wieder zu den Speicheradressen der Bildschirmpunkte einzufügen.

```
HI=INT(SCREENMEM +40)/256
LO=(SCREENMEM + 40)-(HI * 256)
POKE 89, HI
POKE 88, LO
```

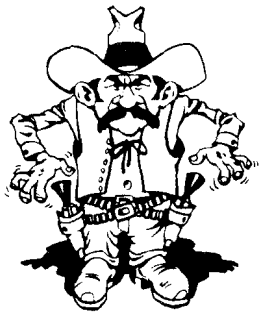
Fügen Sie diese Zeilen zu Ihrem Programm hinzu:

```
300 REM SCHREIBE AUF DEN BILDSCHIRM
310 SCREENMEM=PEEK(88)+PEEK(89)*256
320 HI=INT((SCREENME+40)/256)
330 LO=(SCREENMEM+40)-(HI*256)
340 POKE 87,0
350 POKE 88,L0
360 POKE 89,HI
370 PRINT CHR$(125):REM LOESCHE BILDSCHIRM
380 POSITION 0,0:PRINT "DIES IST GRAPHICS 0"
```

Drücken Sie <SYSTEM RESET> um den Bildschirm zu klären. Ihr Bildschirm sollte so aussehen:

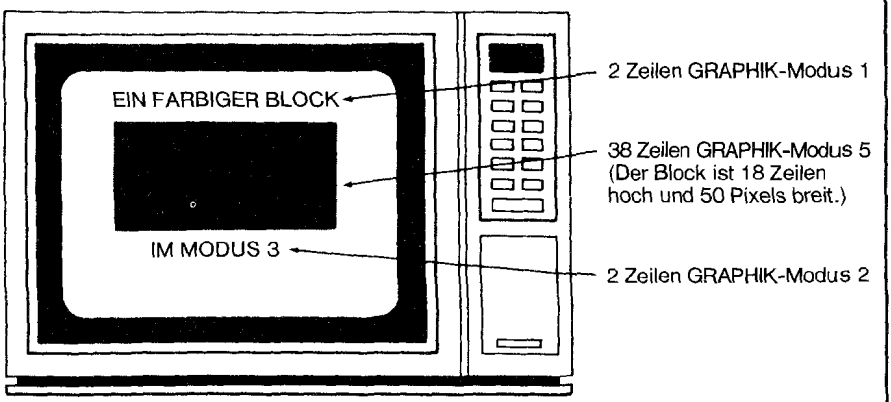


Machen wir eine Pause und eine neue Aufgabe.



Aufgabe #12

Schreiben Sie ein Programm, das den Bildschirm wie folgt aussehen läßt.



Beispiele und Vorschläge

Wann werden Bildschirme mit verknüpften Modi verwendet? Die offensichtliche Antwort lautet, daß dies immer dann der Fall ist, wenn Sie einen Bildschirm erstellen wollen, der mit normalem BASIC nicht zu programmieren ist. Hier sind ein paar Beispiele.

1. Eine Textseite oder ein Graphikdiagramm, kann sehr gut mit einer Titelzeile im BASIC Modus 1 oder 2 überschrieben werden.

2. BASIC GRAPHIK-Modus 0 Fenster können passend auf dem Bildschirm angeordnet werden.

3. Ein einzelnes Graphikdiagramm kann aus verschiedenen GRAPHIK-Modi kombiniert werden.

4. Die ANTIC Modi 3, 4, 5, 12 und 14 können normalerweise nicht im BASIC verwendet werden, aber in eigenen Display Lists. Vierfarbige Zeichen können in den ANTIC Modi 4 und 5 verwendet werden (mehr davon in Kapitel 10). ANTIC Modus 14 verwendet auch vier Farben mit derselben vertikalen Auflösung, wie BASIC Modus 8.

5. Die Bildschirmgröße kann derart verändert werden, daß der Schirm mehr als 192 Scan Lines umfasst.

6. Leerzeilen können verwendet werden, um Bildschirmabschnitte zu trennen.

7. Sie können vielfältige Bildschirme und Display Lists haben.

Lassen Sie uns ein paar dieser Beispiele testen. Hier ist ein Programm, das die Anzahl der Leerzeilen an der Spitze eines Bildschirms im GRAPHIK-Modus 0 verändert:

```
10 GRAPHICS 0
20 DLIST=PEEK(560)+PEEK(561)*256
30 POKE DLIST,0:POKE DLIST+1,0:POKE DLIST+2,0
```

Starten Sie es. Haben Sie bemerkt, das der Bildschirm nach oben gerückt ist? Die Codes der Leerzeilen wurden in der Display List verändert. Die Codes sind jetzt: 0-1 Zeile, 16-2 Zeilen, 32-3 Zeilen, 48-4 Zeilen, 64-5 Zeilen, 80-6 Zeilen, 96-7 Zeilen, 112-8 Zeilen.
Wir wollen noch etwas anderes ausprobieren. Geben Sie das folgende Programm ein und starten Sie es:

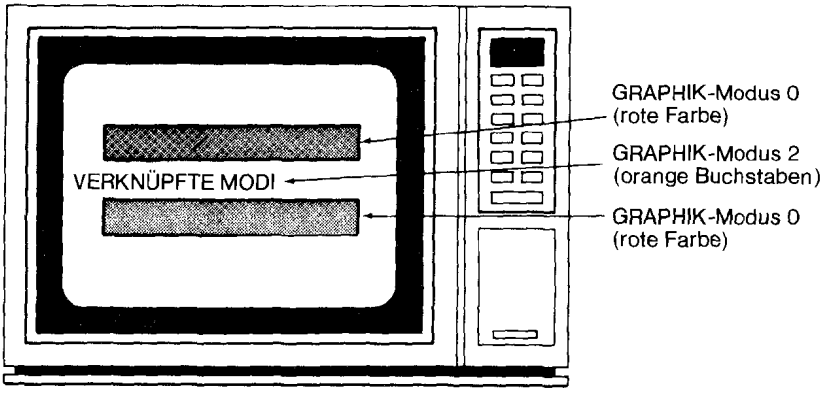
```
10 REM VIELFAELTIGE BILDSCHIRME
200 REM BILDSCHIRM NR. 1
210 MEMTOP=PEEK(106):REM SPITZE DES SPEICHERS
220 GRAPHICS 5
230 DLIST1=PEEK(560)+PEEK(561)*256:REM DISPLAY LIST 1
240 REM ZEICHNE ETWAS AUF BILDSCHIRM 1
250 COLOR 1
260 FOR X=1 TO 79 STEP 2
270 PLOT X,0:DRAWTO X,39
280 NEXT X
290 PRINT "WELCHE DARSTELLUNG (1 ODER 2)"
300 REM BILDSCHIRM 2
310 POKE 106, MEMTOP-8:REM BEWEGE SPITZE DES SPEICHERS NACH UNTEN
320 GRAPHICS 3
330 DLIST2=PEEK(560)+PEEK(561)*256:REM DISPLAY LIST 2
340 REM ZEICHNE ETWAS AUF BILDSCHIRM 2
350 COLOR 2
360 FOR Y=1 TO 19 STEP 2
370 PLOT 0,Y:DRAWTO 39,Y
380 NEXT Y
```

```
400 REM WECHSLE ZWISCHEN BILDSCHIRMEN
410 PRINT "WELCHE DARSTELLUNG (1 ODER 2)":INPUT D
420 IF D=1 THEN DLIST=DLIST1
430 IF D=2 THEN DLIST=DLIST2
500 REM BERECHNE HI UND LO BYTES DER DISPLAY LIST
510 HI=INT(DLIST/256):LO=DLIST-(HI*256)
520 POKE 561,HI:POKE 560,LO
530 GOTO 400
```

Das Programm hat zwei Display Lists und zwei Bildschirm-RAM Flächen erzeugt. Da wir die Bild 1 RAM sozusagen lahmgelegt haben, können wir unsere Eingabe einer 1 oder 2 gar nicht in dem Textfenster beobachten, wenn der Bildschirm 1 dargestellt wird. Das Programm wird trotzdem auf Ihre Eingabe reagieren.
Diese Technik gibt uns die Möglichkeit zwischen zusammengefaßten Bildschirmdarstellungen zu wechseln, ohne immer abwarten zu müssen, bis die Zeichnung wiederhergestellt ist. Die Anzahl der verfügbaren Bildschirme wird nur durch die vorhandene RAM-Kapazität eingeschränkt.

Aufgabe #13

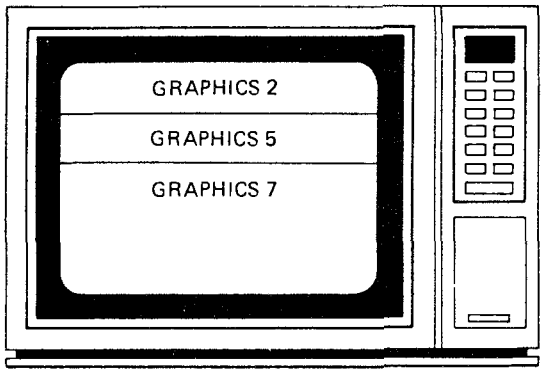
Schreiben Sie ein Programm für die folgende Bilddarstellung.



Zusammenfassung von Kapitel 7

Beantworten Sie bitte die folgenden Fragen.

Die nachfolgende Abbildung bezieht sich auf die Fragen 1, 2 und 3.



1. Wie viele Bildschirmzeilen werden benötigt, um vier GRAPHICS 2 Mode Lines zu erzeugen?

2. Ein Bildschirm mit 192 Zeilen wird belegt von den BASIC GRAFIK-Modi 2,5 und 7. Wieviele Mode Lines verbleiben für GRAPHICS 7, wenn Sie vier GRAPHICS 2 Mode Lines und zehn GRAPHICS 5 Mode Lines verwenden?

3. Wieviele Mode Lines würden für jedes GRAPHIK-Modus Fenster beansprucht, wenn jedes der drei GRAPHIK-Fenster den selben Anteil der Bildschirmfläche belegt?

4. Welche Speicheradresse wird verwendet, um, vor der Ausgabe auf den Bildschirm, in den GRAPHIK-Modus umzuschalten?

5. Nennen Sie die Werte des nieder- und höchstwertigen Bytes, wenn die Bildschirmadresse einen Wert von 14296 (dezimal) hat.

6. Wieviele ANTIC Mode Bytes werden in der Display List benötigt, um 80 Bildschirmzeilen im GRAPHIK-Modus 5 darzustellen?

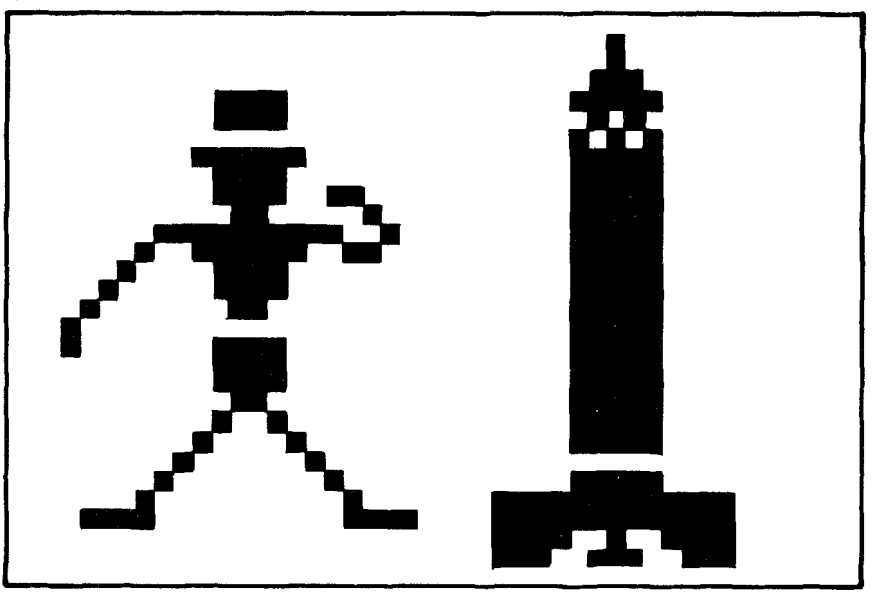
7. In welcher Speicherzelle steht die Anfangsadresse der Display List?

8. Schreiben Sie eine Display List mit Dezimalwerten, die ein Schirmbild erzeugt mit zwei GRAPHICS 0 Mode Lines und dem restlichen Bildschirm in GRAPHICS 2 Modus.

KAPITEL 8

Spielen mit Playern und Missiles

Player - Spieler? Missiles - Raketen? Das klingt gefährlich.



In diesem Kapitel werden Sie lernen:

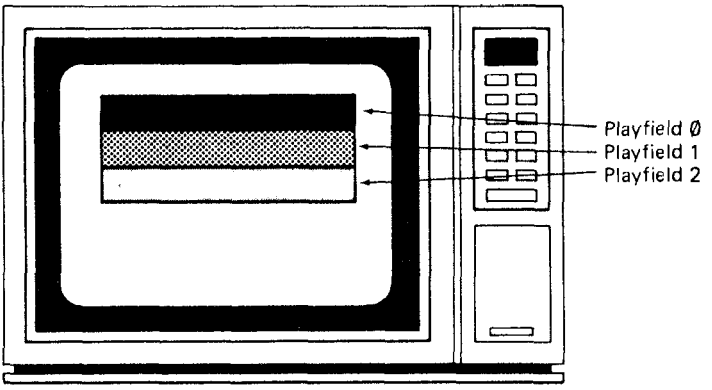
- Player und Missiles zu erstellen
- Player und Missiles zu bewegen
- etwas über die Zusammenhänge zwischen Playern/Missiles/ und Playfields

Der ATARI Computer hat eine spezielle Handhabung für die Verwendung von Playern und Missiles. Aber was sind Player und Missiles? Übersetzt heißt das ja Spieler und Raketen. Da es sich in Verbindung mit dem ATARI Computer um feststehende Ausdrücke handelt, belassen wir es bei Playern und Missiles. Zurück zu unserer Frage. Die beste Antwort auf diese Frage erhalten wir durch ein Beispiel. Geben Sie das folgende Programm ein und starten Sie es:

```
100 REM PLAYER/MISSILE BEISPIELPROGRAMM
110 GRAPHICS 0
200 REM SETZT DIE BASIS ADRESSE FÜR PLAYER
210 PMBASE=PEEK(106)-8
220 REM LOESCHT SPEICHERBEREICH UND LEGT DORT OBJEKTDATEN AB
230 FOR X=0 TO 255:POKE PMBASE*256+X,0:NEXT X
240 FOR X=40 TO 45:READ D:POKE PMBASE*256+512+X,D:NEXT X
250 DATA 24,24,60,60,126,126
300 REM SETZT PLAYER PARAMETER
310 POKE 559,46
320 POKE 53248,100
330 POKE 53256,3
340 POKE 54279,PMBASE
350 POKE 53277,3
```

Die Figur, die Sie auf dem Bildschirm sehen, ist ein Player. Allgemein gesagt sind Player und Missiles Figuren, die vom Programmierer entworfen werden und dann einfach über den Bildschirm bewegt werden können. Diese Figuren sind sogar unabhängig von den auf dem Bildschirm befindlichen Texten oder Zeichnungen. Listen Sie einmal das Programm, damit Sie sehen was wir meinen. Die Programmzeilen werden über dem Player dargestellt.

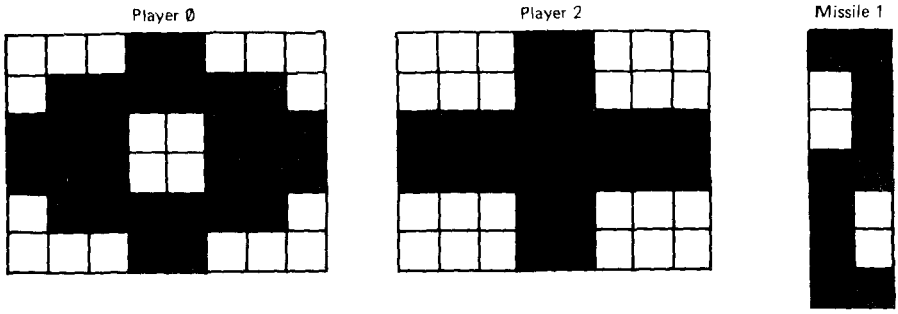
Um diese spezielle Art von Graphik zu verstehen, müssen wir etwas zurückdenken, und wiederholen, was wir über die Graphikmöglichkeiten des ATARI wissen. Im Teil Eins dieses Buches haben wir gelernt auf dem Bildschirm zu zeichnen. Dabei haben wir die verschiedenen Farbregister verwendet. Jedes mal, wenn wir ein bestimmtes Register verwendeten, haben wir ein sogenanntes Playfield gezeichnet. Das Farbdarstellungsprogramm in Kapitel 3 zum Beispiel verwendet drei Playfields, da drei verschiedene Farbregister verwendet wurden, um die Figuren auf dem Bildschirm zu zeichnen.



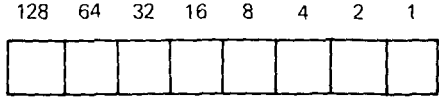
Jedes Pixel in einem Playfield wird durch Daten repräsentiert, die im Bildschirmspeicher des RAM abgelegt sind. Die Daten für die linke obere Ecke des Bildschirmes starten an einer bestimmten Speicheradresse. Die weiteren Bildschirmdaten sind dann in aufeinanderfolgenden Adressen abgelegt, bis die rechte untere Ecke des Bildschirmes erreicht ist. Zwei verschiedene Playfields können nicht den selben Bereich im Bildschirmspeicher belegen, *und Playfields können sich nicht* überschneiden. Die verschiedenen Playfields werden im Bildschirmspeicher einfach durch unterschiedliche Datenwerte gekennzeichnet, entsprechend dem zum Zeichnen des Playfields verwendeten Farbregister.
Players und Missiles dagegen beanspruchen einen separaten Speicherbereich und können unabhängig von den Playfield-Daten dargestellt werden.

Erstellen von Playern und Missiles

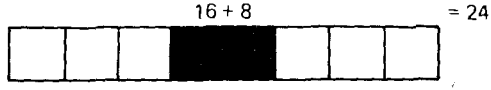
Am besten lernen wir das Erstellen von Playern und Missiles mit einem Beispiel. Ein Player ist eine Figur, die 8 Bits (Pixel) breit ist und bis zu 256 Scan Lines (Bildschirmzeilen) hoch sein kann. Zur selben Zeit können bis zu vier verschiedene Player gleichzeitig verwendet werden. Ein Missile ist eine Figur, die 2 Bits (Pixel) breit ist und ebenfalls bis zu 256 Scan Lines (Bildschirmzeilen) hoch sein kann. Genau wie bei den Playern, können vier Missiles zur gleichen Zeit verwendet werden. Lassen Sie uns die Form von zwei Playern und einem Missile entwerfen. Nennen wir sie Player 0, Player 2, und Missile 1.



Jede Zeile jedes Players ist acht Bit breit und kann so durch ein Byte dargestellt werden. Da wir diese Daten eventuell mit Hilfe der BASIC-Anweisung POKE in den Speicher des Computers bringen wollen, repräsentieren wir jede dieser Zeilen (Bytes) mit einem Dezimalwert.
Betrachten wir Zeile für Zeile Player 0. Die acht Bits in jeder Zeile können, wie unten gezeigt, jeweils durch einen Dezimalwert dargestellt werden.



Wenn die beiden mittleren Bits, wie es bei Player 0 der Fall ist, EINgeschaltet werden, muß unser Dezimalwert lauten:

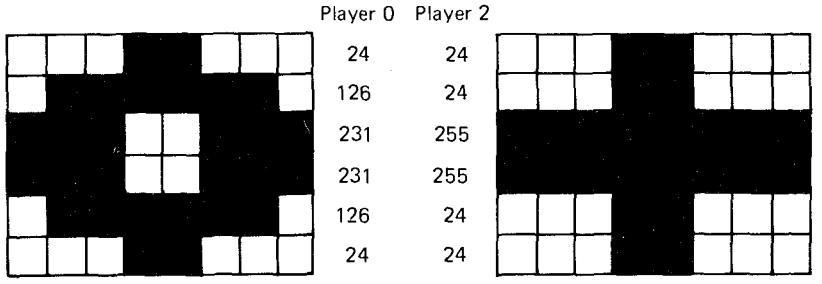


Wie lautet der entsprechende Dezimalwert für diese Zeile?

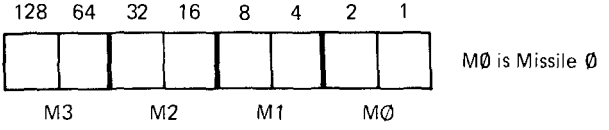
Antwort #1 _____

#1.90

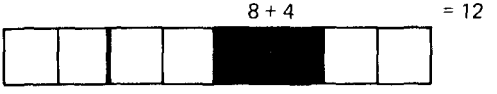
Jeder dieser Player kann durch die folgenden Werte dargestellt werden:



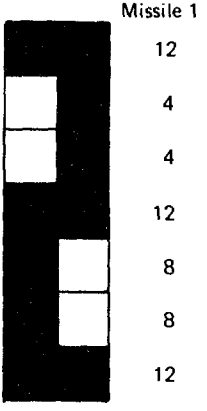
Jetzt wollen wir die Werte für Missile 1 herausfinden. Jede Zeile des Missiles ist nur zwei Bits breit. Trotzdem werden die Daten für alle vier Missiles in einem Byte abgelegt. Deshalb unterscheidet sich die Berechnung etwas von der bei den Playern verwendeten Methode.



Die erste Zeile von Missile 1 wäre dann



Der gesamte Missile wird dann wie folgt dargestellt



?????

Nehmen wir an, wir würden für Missile 0 dieselbe Figur wie für Missile 1 verwenden (Missile 1 nicht verwendet). Wie lautete der Wert für das Byte, das die erste Zeile vertritt?

Antwort #2 _____

?????

Nehmen wir an, wir würden dieselbe Figur von eben für Missile 1 und Missile 0 verwenden. Wie lautete der Wert für das Byte, das die erste Zeile repräsentiert?

Antwort #3 _____

Jetzt, nachdem wir die Player und das Missile entworfen haben, stellt sich die Frage, was wir mit ihnen anfangen können. Lassen Sie sie uns zuerst einmal im Speicher ablegen. Selbstverständlich können Sie nicht irgendwo abgelegt werden. Jeder Player benötigt entweder 128 oder 256 Bytes, je nachdem ob einfache oder doppelte Scan Line-Auflösung gewünscht wird. Die vier Missiles benötigen ebenso entweder 128 oder 256 Bytes Speicherplatz.

#2.3 #3.15

Betrachten wir die Player/Missile Speicherkarte in Bild 8-1. Diese Karte zeigt die Zusammenstellung der Player und Missiles unter Bezug auf eine bestimmte Player/Missile Basis ADRESSE (PMBASE). Wir werden diese Karte verwenden, um die Speicherstellen für die von uns entworfenen Player und das Missile zu bestimmen.

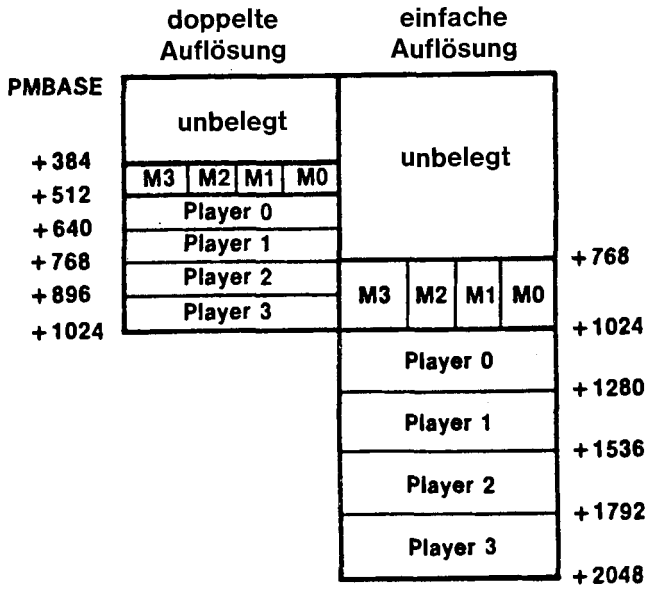


BILD 8-1: Player-Missile Speicherkarte. (Abgedruckt aus dem De Re ATARI mit Genehmigung von ATARI, Inc.)

Wir wollen jetzt gleichzeitig Player und Missiles erstellen und ein Programm schreiben. Wir werden die Player und Missiles unter Verwendung von doppelter Auflösung erstellen. Das bedeutet, daß sich jeder Player oder jedes Missile auf dem Bildschirm über zwei Scan Lines erstreckt.

1. Starten wir mit der folgenden Zeile:

```
100 REM PLAYER 0, 2 UND MISSILE 1
```

2. Als nächstes müssen wir einen Speicherbereich finden, den wir, ohne Schwierigkeiten befürchten zu müssen, verwenden können. Dazu müssen wir die höchste Adresse des Speicherbereichs finden und dann vier 256 Byte Seiten (Pages), bzw. 1024 Bytes, abziehen. Für einfache Auflösung müßten acht Seiten abgezogen werden. Geben Sie die folgenden Programmzeilen ein:

```
200 REM SETZE PLAYER/MISSILE BASIS ADRESSE
210 PMBASE=PEEK(106)-4
```

Damit sich der Bildschirmspeicher, der sich normalerweise direkt ab der höchsten Adresse des Speicherplatzes befindet, nicht mit dem Player/Missile Speicherbereich überschneidet, müssen wir den Computer täuschen, indem wir den Zeiger für die höchste ADRESSE des Speicherbereichs verändern. Fügen Sie zwei weitere Zeilen zu unserem Programm hinzu:

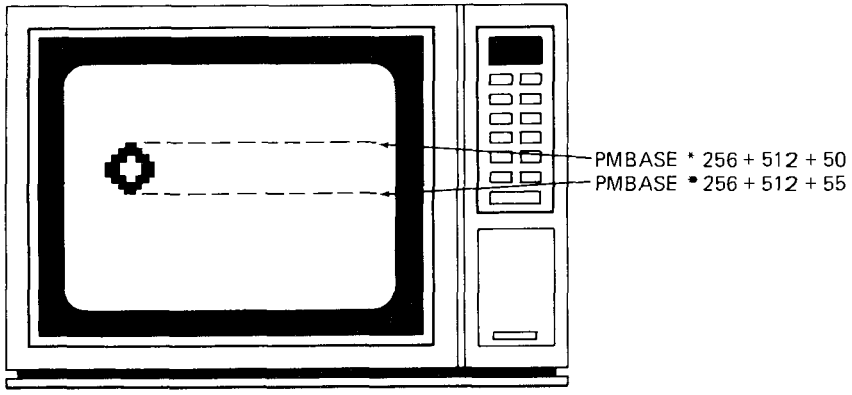
```
220 POKE 106,PMBASE-1
230 GRAPHICS 0
```

Zeile 220 setzt den Zeiger für die höchste Adresse des Speicherbereichs direkt eine Seite unter den Player/Missile-Speicherbereich. Zeile 230 initialisiert die Position des Bildschirmspeichers neu.

3. Um sicher zu gehen, daß dieser obere Bereich des Speichers leer ist, können wir ihn mit Nullen füllen. Die folgende Routine führt diesen Vorgang aus. Geben Sie die Zeilen ein:

```
300 REM LOESCHT DEN PLAYER/MISSILE SPEICHERBEREICH
310 FOR X=384 TO 1024
320 POKE PMBASE*256+X,0
330 NEXT X
```

4. Jetzt können wir unsere Player und Missile-Daten in diesem Speicherbereich ablegen. Die vertikale Position eines Players hängt von seiner Position im Speicher ab. Jeder Player kann irgendwo innerhalb der 128 vertikalen Positionen auf dem Bildschirm plaziert werden, indem er irgendwo innerhalb der für ihn reservierten 128 Bytes abgelegt wird. Zum Beispiel kann Player 0 irgendwo innerhalb des Bereiches von PMBASE*256+512 und PMBASE*256+640 abgelegt werden. Wenn wir wollen, daß Player 0 in einem Abstand von 50 doppelten Scan Lines vom oberen Rand des Bildschirmes dargestellt wird, wird er im Speicher ab ADRESSE PMBASE*256+512+50 abgelegt.



Zu diesem Zweck kann die folgende Routine verwendet werden. Geben Sie diese Programmzeilen ein:

```
400 REM LESE PLAYER MISSILE DATEN
410 FOR TIMES=1 TO 3
420 READ ADDR1,ADDR2
430 FOR X=ADDR1 TO ADDR2
440 READ N:POKE PMBASE*256+X,N
450 NEXT X
455 NEXT TIMES
460 REM PLAYER 0
465 DATA 562,567,24,126,231,231,126,24
470 REM PLAYER 2
475 DATA 818,823,24,24,255,255,24,24
480 REM MISSILE 1
485 DATA 434,440,12,4,4,12,8,8,12
```

Als nächstes müssen wir ANTIC mit den notwendigen Informationen wie Farbe, Position des Players usw. versorgen. Eine komplette Liste aller Player/Missile SpeicherADRESSEn ist in Tabelle 8-1 aufgeführt. Sie wird bei der Bestimmung der spezifischen SpeicherADRESSEn jedes Players hilfreich sein.

5. Um die Farbe der Player und Missiles zu setzen, müssen wir mit dem Befehl POKE den entsprechenden Farbwert in das Player/Missile Farbregister legen. Fügen Sie die folgenden Zeilen zu unserem Programm hinzu:

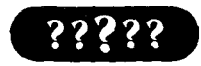
```
500 REM SETZE FARBEN
505 POKE 704,0
510 POKE 705,200
520 POKE 706,30
```

Die Werte 704, 705, und 706 sind die Farbregister ADRESSEn von Player 0, Player 1, und Player 2. Missile 1 erhält dieselbe Farbe, wie Player 1. Die Farbwerte können Tabelle 1-1 entnommen werden, doch müssen sie mit folgender Formel umgerechnet werden:

Farbe * 16 + Helligkeit = Farbwert

In diesem Fall errechnen sich die Farbwerte zu:

0 * 16 + 0 = 0	**Eine Farbe mit Wert 0 ist schwarz**
12 * 16 + 8 = 200	**Eine Farbe mit Wert 12 ist grün**
1 * 16 + 14 = 30	**Eine Farbe mit Wert 1 ist orange**



Was wäre der Farbwert für die Farbe rosa (4) mit einer Helligkeit von 10?

Antwort #4 _____

6. Die horizontalen Positionen der Player und Missiles werden mit einem Horizontal-Positions Register bestimmt. Geben Sie die folgenden Programmzeilen ein:

```
600 REM SETZE DIE HORIZONTALE POSITION
610 POKE 53248,100
620 POKE 53250,160
630 POKE 53253,120
```

Die Werte 53248, 53250, und 53253 sind die Horizontal-Positions Register für Player 0, Player 2, und Missile 1. Jedes Register kann einen Wert zwischen 0 und 228 enthalten. Einige dieser Positionswerte befinden sich allerdings außerhalb der rechten und linken Seite des Bildschirmes.

7. Die vertikale Auflösung wollen wir auf doppelte Scan Lines setzen. Geben Sie die folgenden Zeilen ein:

```
700 REM SETZE VERTIKALE AUFLOESUNG
710 POKE 559,46
```

Der Wert 559 ist das Schattenregister für die Direkte Speicherzugriffs-Kontrolle (DMA = direct memory access control). Bit 4 ist für die vertikale Auflösung zuständig. Für doppelte Zeilenauflösung wird der Wert 46 und für einfache Zeilenauflösung wird der Wert 64 in diesem Register abgelegt.

#4.74

8. Zum Schluß müssen wir die Player aktivieren. Geben Sie die folgenden Zeilen ein:

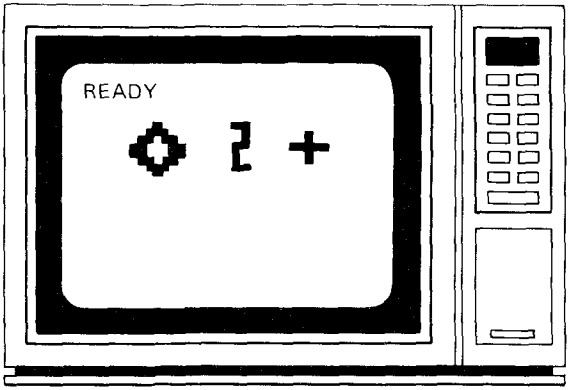
```
800 REM AKTIVIERT DIE PLAYER
810 POKE 54279,PMBASE
820 POKE 53277,3
```

Der Wert 4279 ist das Register, in dem der Wert der Basis-Adresse für die Player und Missiles abgelegt wird. Adresse 53277 ist das Player/Missile-Aktivierungsregister.

Ihr fertiges Programm sollte jetzt so aussehen:

```
100 REM PLAYER 0, 2 UND MISSILE 1
200 REM SETZE PLAYER/MISSILE BASIS ADRESSE
210 PMBASE=PEEK(106)-4
220 POKE 106,PMBASE-1
230 GRAPHICS 0
300 REM LOESCHE PLAYER/MISSILE SPEICHERBEREICH
310 FOR X=384 TO 1024
320 POKE PMBASE*256+X,0
330 NEXT X
400 REM LESE PLAYER MISSILE DATEN
410 FOR TIMES=1 TO 3
420 READ ADDR1,ADDR2
430 FOR X=ADDR1 TO ADDR2
440 READ N:POKE PMBASE*256+X,N
450 NEXT X
455 NEXT TIMES
460 REM PLAYER 0
465 DATA 562,567,24,126,231,231,126,24
470 REM PLAYER 2
475 DATA 818,823,24,24,255,255,24,24
480 REM MISSILE 1
485 DATA 434,440,12,4,4,12,8,8,12
500 REM SETZE FARBEN
505 POKE 704,0
510 POKE 705,200
520 POKE 706,30
600 REM SETZE HORIZONTALE POSITION
610 POKE 53248,100
620 POKE 53250,160
630 POKE 53253,120
700 REM SETZE VERTIKALE AUFLOESUNG
710 POKE 559,46
800 REM AKTIVIERE DIE PLAYER
810 POKE 54279,PMBASE
820 POKE 53277,3
```

Starten Sie das Programm. Haben Sie bitte etwas Geduld, da der Programmlauf einige Sekunden dauert. Sieht Ihr Bildschirm ungefähr so aus, wie der Folgende?



Etwas über Player und Missiles

Lassen Sie uns mehr über Player und Missiles erfahren, indem wir einige Änderungen an unserem Programm vornehmen. Schlagen Sie jeweils in Tabelle 8-1 nach, wenn Sie die folgenden Änderungen durchführen.

Vertikale Position

Verschieben Sie Missile 1 auf dem Bildschirm nach unten, indem Sie seine Position im Speicherbereich ändern. Sie können seine Position verschieben, indem Sie die Werte in der DATA-Anweisung in Zeile 485 ändern. Ändern Sie Zeile 485 wie folgt:

```
485 DATA 464,470,12,4,4,12,8,8,12
```

Drücken Sie <SYSTEM RESET>, um den Zeiger für die höchste SpeicherADRESSE wieder auf den Normalwert zu setzen. Starten Sie dann das Programm. Hat sich Missile 1 auf dem Bildschirm nach unten bewegt?
Im nächsten Kapitel werden wir einen Player animieren, indem wir verschiedene Daten innerhalb des Speichers verschieben.

TABELLE 8-1. PLAYER/MISSILE SPEICHERADRESSEN

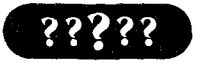
Beschreibung	ADRESSE
Direkte Speicherzugriffs-Kontrolle (DMA)	**559**
Graphik Kontrolle	53277
Graphik für alle Missiles	53265
Graphik für Player 0	53261
Graphik für Player 1	53262
Graphik für Player 2	53263
Graphik für Player 3	53264
Horizontale Position von Missile 0	53252
Horizontale Position von Missile 1	53253
Horizontale Position von Missile 2	53254
Horizontale Position von Missile 3	53255
Horizontale Position von Player 0	53248
Horizontale Position von Player 1	53249
Horizontale Position von Player 2	53250
Horizontale Position von Player 3	53251
Farbhelligkeit vom Hintergrund	712
Farbhelligkeit von Playfield 0	708
Farbhelligkeit von Playfield 1	709
Farbhellogkeit von Playfield 2	710
Farbhelligkeit von Playfield 3	711
Farbhelligkeit von Player-Missile 0	704
Farbhelligkeit von Player-Missile 1	705
Farbhelligkeit von Player-Missile 2	706
Farbhelligkeit von Player-Missile 3	707
Größe für alle Missiles	53260
Größe von Player 0	53256
Größe von Player 1	53257
Größe von Player 2	53258
Größe von Player 3	53259
Prioritäten-Auswahl	623
Player Missile Basis Adresse	54279

Horizontale Position

Ändern Sie die horizontale Position von Missile 1. Dies kann ganz einfach durchgeführt werden, indem man den Wert des Horizontal-Positions Registers ändert. Das können Sie sogar im Direkt-Modus machen. Tippen Sie:

POKE 53253,160 <RETURN>

Hat sich Missile 1 unter Player 2 geschoben? Selbstverständlich können Sie diese Positionen auch im Programm-Modus ändern. Der Wert dieses Registers kann zwischen 0 und 228 liegen. Dennoch liegen einige dieser Positionswerte außerhalb des Bildschirmes. Wenn Sie einen Player oder Missile verschwinden lassen wollen, so brauchen Sie nur das entsprechende Horizontal-Positions Register auf 0 zu setzen.



Welchen Wert würden Sie verwenden, um einen Player auf der horizontalen Mitte des Bildschirmes zu positionieren? Dabei gehen wir davon aus, daß der Bildschirm 228 Einheiten breit ist.

Antwort #5 _____

Breite

Die Breite von Playern oder Missiles kann ganz einfach geändert werden. Auch das wollen wir sofort einmal im Direkt-Modus ausprobieren. Tippen Sie:

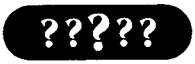
POKE 53260,1 <RETURN>

Missile 1 sollte nun doppelt so breit sein, wie zuvor. Lassen Sie uns die Breite vervierfachen. Tippen Sie:

POKE 53260,3 <RETURN>

Schlagen Sie in Tabelle 8-1 für die ADRESSEn der anderen Player nach. Wenn wir in die ADRESSE 53260 den Wert 0 legen, so erhalten wir wieder unsere normale Breite.

#5. 114



Welche Anweisung würden Sie verwenden, um die Breite von Player 0 zu vervierfachen?

Antwort #6 _____

Priorität

Nehmen wir einmal an, die horizontalen Positionen von Player 0 und Player 1 hätten beide denselben Wert. Welcher der beiden Player würde auf welchem erscheinen? Probieren wir es doch einfach aus. Dazu setzen wir zuerst die horizontale Position der beiden Player auf 160. Tippen Sie:

POKE 53248,160 <RETURN>

Können Sie Player 2 durch die Mitte von Player 0 durchscheinen sehen? Ändern Sie jetzt die Priorität, indem Sie die folgende Anweisung eingeben:

POKE 623,2 <RETURN>

Beachten Sie, daß Player 2 verschwunden ist.
Zwischen Playern oder zwischen Playern und Playfields können also verschiedene Prioritäten (Vorränge) gesetzt werden. Bild 8-2 zeigt die einzelnen Prioritäten und die entsprechenden Bits des Prioritätenregisters (PRIOR), die gesetzt werden müssen.

PRIOR

Bit 4	Bit 3	Bit 2	Bit 1	Bit Ø
Fifth Player Enable	PF0-PF1 P0-P3 PF2-PF3	PF0-PF3 P0-P3 ———	P0-P1 PF0-PF3 P2-P3	P0-P3 PF0-PF3 ———

BILD 8-2: Player/Playfield Prioritäten.

#6 POKE 53256,3

In unserem Beispiel hat die Anweisung POKE 623,2 (Bit 1 gesetzt) Player 0 und Player 1 (P0-P1) Priorität über alle Playfields (PF0-PF3) gegeben. Die Playfields wiederum haben Priorität über Player 2 und Player 3 (P2-P3). Da der Hintergrund eine höhere Priorität als Player 2 hat, verschwindet dieser hinter dem Hintergrund.

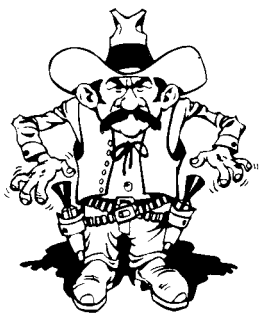
?????

Welche Anweisung würden Sie verwenden, um Player 0 (P0) Priorität über Playfield 3 (PF3) zu geben?

Antwort #7 _____

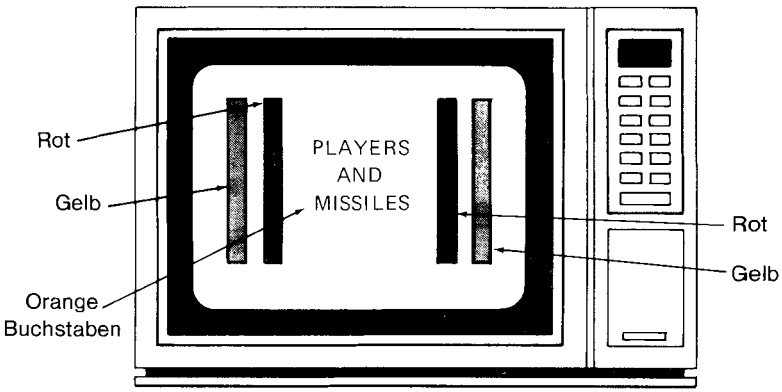
Die vier Missiles können auch zu einem fünften Player zusammengefasst werden. Wenn in PRIOR Bit 4 gesetzt ist, hat der fünfte Player den Farbwert des Playfield Farbregisters 3. Die Position kann nach wie vor unabhängig bestimmt werden.

Aufgabe #14



Schreiben Sie ein Programm, das Ihren Bildschirm so aussehen läßt, wie den unten abgebildeten. Verwenden Sie für den Text GRAPHIK-Modus 1.

Hinweis: Die roten und gelben Balken sind Player.



#7. POKE 623,1

Kollisionen

Es gibt auch Hardware Register, die das Überlappen bzw. eine Kollision (Zusammenstoß) von Playern mit Playern oder Missiles und Playern mit Playfields erkennen lassen. Tabelle 8-2 zeigt alle Kollisionsregister.

Das wollen wir am besten gleich einmal ausprobieren. Drücken Sie SYSTEM RESET und starten Sie erneut unser Programm. Um festzustellen, ob Player 0 mit einem anderen Player kollidierte, überprüfen wir den Inhalt seines Kollisionsregisters. Geben Sie bitte ein:

```
PRINT PEEK(53260)  <RETURN>
```

Als Ergebnis sollte der Wert 0 auf dem Bildschirm gezeigt werden, da keine Kollision stattgefunden hat. Jetzt lassen wir ihn kollidieren. Tippen Sie:

```
POKE 53248,160  <RETURN>
```

TABELLE 8-2. KOLLISIONS REGISTER

Beschreibung	ADRESSE
Missile 0 - Playfield Kollision	53248
Missile 0 - Player Kollision	53256
Missile 1 - Playfield Kollision	53249
Missile 1 - Player Kollision	53257
Missile 2 - Playfield Kollision	53250
Missile 2 - Player Kollision	53258
Missile 3 - Playfield Kollision	53251
Missile 3 - Player Kollision	53259
Player 0 - Playfield Kollision	53252
Player 0 - Player Kollision	53260
Player 1 - Playfield Kollision	53253
Player 1 - Player Kollision	53261
Player 2 - Playfield Kollision	53254
Player 2 - Player Kollision	53262
Player 3 - Playfield Kollision	53255
Player 3 - Player Kollision	53263
Löschen der Kollisionsregister	53278

Überprüfen Sie jetzt noch einmal das Kollisionsregister. Tippen Sie:

```
PRINT PEEK(53260) <RETURN>
```

Auf dem Bildschirm wird ein Wert ungleich Null ausgegeben. Jeder Wert ungleich Null zeigt an, daß eine Kollision stattgefunden hat. Um das Kollisionsregister einer beliebigen Kollision zu löschen, braucht nur ein beliebiger Wert in das Register 53278 gePOKEt zu werden (Kollisions-Lösch-Register).
Das Erkennen von Kollisionen ist besonders bei animierter Graphik von großer Bedeutung. In Kapitel 9 werden wir die Technik der Animation näher untersuchen.

Anwendungen

Die Möglichkeiten, Player und Missiles kreativ einzusetzen, sind nur durch Ihre Phantasie begrenzt. Hier sind einige Ideen:

1. Player können gebraucht werden, um vier zusätzliche Farben auf einem Bild verwenden zu können, und zwar unabhängig vom angesprochenen GRAPHIK-Modus.

2. Sie können Player verwenden, um in jedem beliebigen GRAPHIK-Modus Zeichen vertikal auf den Bildschirm zu schreiben.

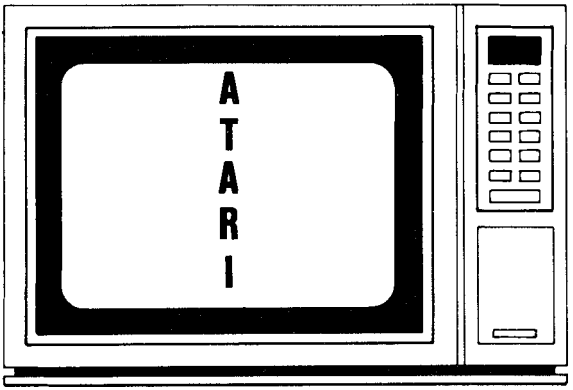
3. Player können einfach animiert werden (siehe Kapitel 9).

4. Player können durchsichtig gemacht werden, dadurch kann man sie zum Hervorheben bestimmter Wörter oder von Teilen eines Diagrammes verwenden.

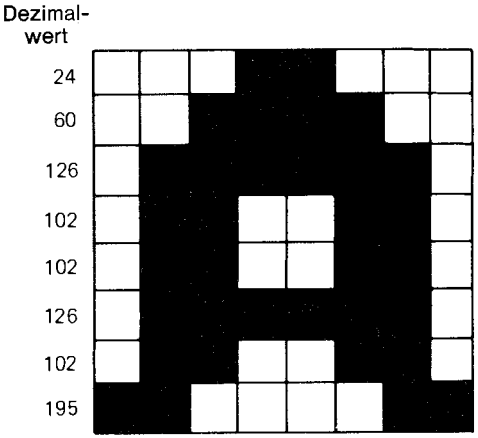
5. Player können in Auswahlmenüs zur Kennzeichnung der ausgewählten Möglichkeit verwendet werden.

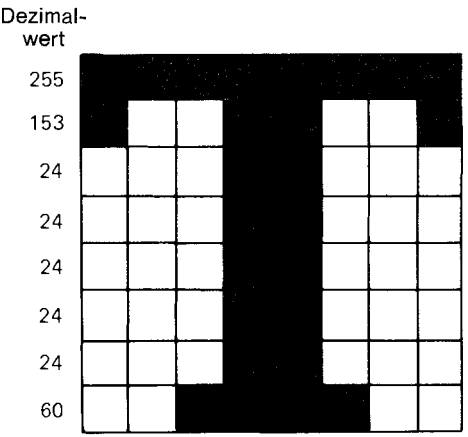
6. Es können dreidimensionale Effekte erzeugt werden, indem man die entsprechenden Prioritäten und die Breite der Player umschaltet.

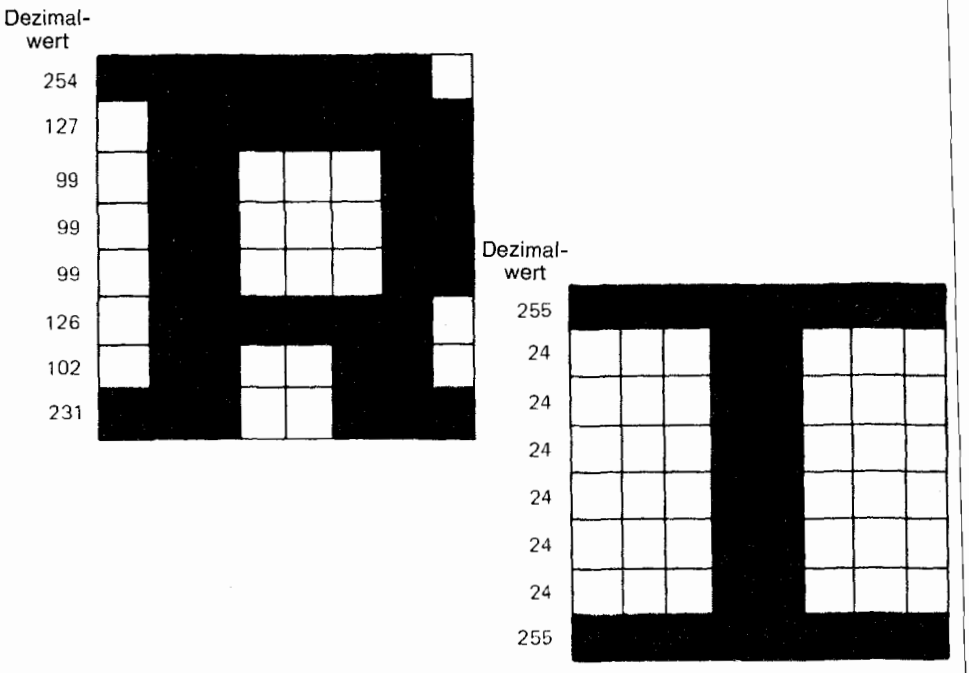
Schauen wir uns noch ein Beispiel an. Wir werden einen Player zur Erstellung von Buchstaben verwenden, und diese dann, wie auf unserem Beispiel, vertikal auf den Bildschirm schreiben.



Als erstes müssen wir die benötigten Zeichen erstellen. Es sind die Buchstaben A, T, R, und I. Sie können eigene Formen entwerfen, oder einfach die folgenden verwenden. Als Hilfsmittel können Sie das Arbeitsblatt zum Entwerfen von Zeichen in Anhang A benutzen.



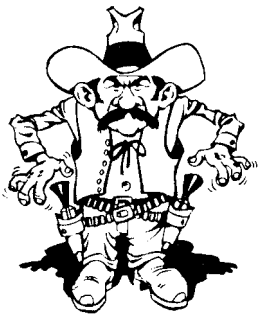




Die Werte können aus DATA-Anweisungen gelesen werden und dann in den Player/Missile Speicherbereich gePOKEt werden.
Für dieses Programm wird nur ein einziger Player benötigt. Geben Sie das Programm ein und probieren Sie es aus.

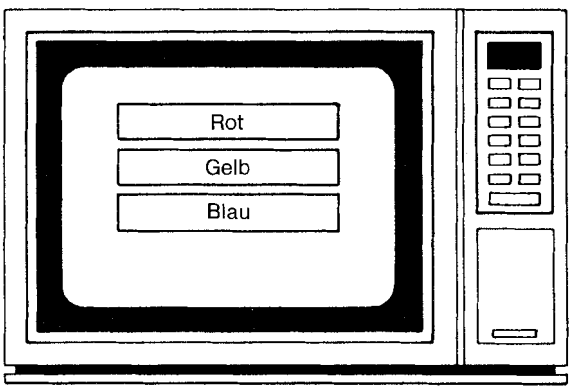
```
100 REM VERWENDEN EINES PLAYERS ALS ZEICHEN
200 REM SETZE PLAYER/MISSILE BASIS ADRESSE
210 PMBASE=PEEK(106)-4
220 POKE 106,PMBASE-1
230 GRAPHICS 0
300 REM LOESCHT PLAYER/MISSILE SPEICHERBEREICH
310 FOR X=384 TO 1024
320 POKE PMBASE*256+X,0
330 NEXT X
400 REM LESE PLAYER MISSILE DATEN
410 FOR X=540 TO 588
420 READ N:POKE PMBASE*256+X,N
430 NEXT X
450 REM DATEN FÜR PLAYER 0
455 REM BUCHSTABE A
```

```
460 DATA 24,60,126,102,102,126,102,195,0,0
465 REM BUCHSTABE T
470 DATA 255,153,24,24,24,24,24,60,0,0
475 REM BUCHSTABE A
480 DATA 24,60,126,102,102,126,102,195,0,0
485 REM BUCHSTABE R
490 DATA 254,127,99,99,99,126,102,231,0,0
495 REM BUCHSTABE I
497 DATA 255,24,24,24,24,24,24,255,0,0
500 REM SETZE FARBEN
510 POKE 704,31
600 REM SETZE HORIZONTALE POSITION
610 POKE 53248,114
700 REM SETZE VERTIKALE AUFLOESUNG
710 POKE 559,46
800 REM AKTIVIERE PLAYER 0
810 POKE 54279,PMBASE
820 POKE 53277,3
```



Aufgabe #15

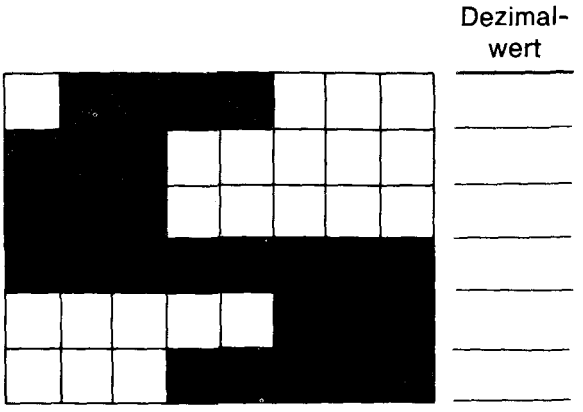
Schreiben Sie ein Programm, um folgende Darstellung im GRAPHIK-Modus 0 zu erhalten. Drei Player (rot, gelb und blau) in vierfacher Breite sind über dem Text in GRAPHIK-Modus 0 plaziert.



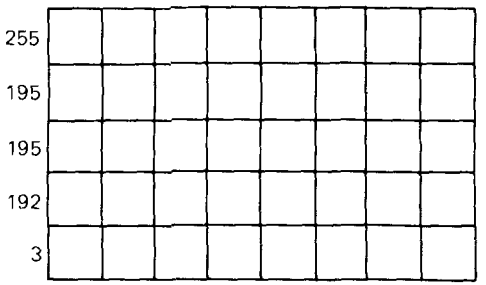
Zusammenfassung von Kapitel 8

Beantworten Sie bitte die folgenden Fragen.

1. Wie lauten die Dezimalwerte für jede Zeile des folgenden Diagrammes?



2. Zeichnen Sie die Form des Players, die durch die fünf aufgeführten Bytes repräsentiert wird.

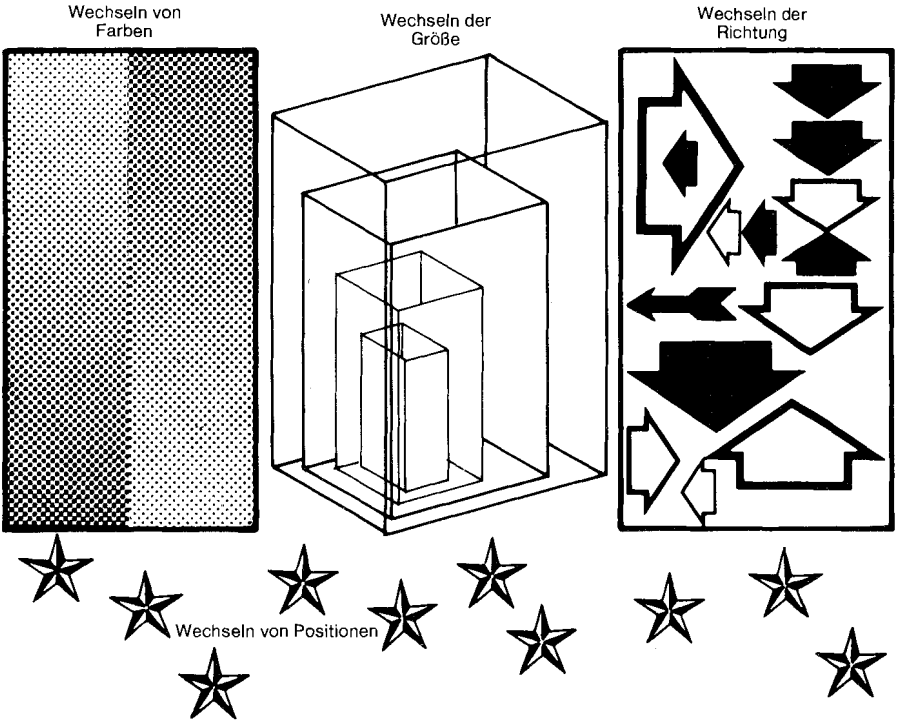


3. Welche Anweisung muß verwendet werden, um Player 2 die Farbe blau (8) mit einer Helligkeit von 4 zu geben?
4. Welche Anweisung müßte verwendet werden, um die horizontale Position von Player 2 auf die horizontale Mitte des Bildschirmes zu setzen?
5. Welche Anweisung wird verwendet, um die Breite von Player 2 zu verdoppeln?
6. Wieviele Seiten (Pages) des Speicherbereichs müssen für Player Missile Graphik in einfacher Auflösung reserviert werden?
7. Welches Bit des PRIOR-Registers muß gesetzt werden, damit Playfield 0 Priorität über Player 3 hat?

KAPITEL 9

Animation

Animation bedeutet ständigen Wechsel, Belebung.



In diesem Kapitel werden Sie lernen:

- Zeichen zu animieren
- Playfields zu animieren
- Player und Missiles zu animieren
- animierte Graphiken zu erstellen

Wenn Sie einmal einen Zeichentrickfilm gesehen haben, haben Sie Animation aus erster Hand miterlebt. Animation bedeutet also auch Bewegung. Für uns ist es jedoch mehr als das. Wir werden unter anderem auch anhand einer Figur auf dem Bildschirm alle Bewegungen durchführen, die etwas mit Animation zu tun haben. Im Zusammenhang mit dem Computer zählen zur Animation blinkende Zeichen, der Wechsel von Farben oder die Bewegung von Figuren oder Zeichen auf dem Bildschirm.

Animation mit Farben

Um die Aufmerksamkeit auf den Bildschirm oder auf Figuren auf dem Bildschirm zu lenken, kann man sehr effektiv Farben einsetzen. Wechselnde Farben können den Programmierer auf Fehler oder wichtige Informationen aufmerksam machen. Zum Beispiel kann man die Randfarben ändern, um in einem Programm auf einen neuen Operationsabschnitt hinzuweisen. Ein blinkender Hintergrund oder ein blinkendes Zeichen kann als Warnung dienen. Die Farben können sogar durch verschiedene Playfields rotiert werden, um eine Bewegung zu vorzutäuschen. Lassen Sie uns einige Beispiele ausprobieren.
In Kapitel 5 haben wir gelernt, wie man die Farbe von Zeichen verändern kann. So kann man auch ein Zeichen blinken lassen, indem man seine Farbe immer wieder umschaltet. Hier ist ein Programmbeispiel. Geben Sie die Zeilen ein und starten Sie es:

```
10 REM BLINKENDER TEXT
20 GRAPHICS 2+16
30 POSITION 8,5:PRINT #6;"ES BLINKT"
40 REM FARBAENDERUNG
50 SETCOLOR 0,0,14
60 FOR WAIT=1 TO 100:NEXT WAIT
70 SETCOLOR 0,0,0
80 FOR WAIT=1 TO 100:NEXT WAIT
90 GOTO 50
```

Die Farbe der Zeichen wird immer wieder zwischen weiß und schwarz (der Hintergrundfarbe) umgeschaltet.

?????

Können Sie die Farbe der Zeichen auch rosa und blau blinken lassen? Welche zwei Anweisungen würden Sie ändern?

Antwort #1 _____

Antwort #2 _____

Das Wechseln der Farben kann auch verwendet werden, um eine Bewegung vorzutäuschen. Betrachten Sie dieses Programmbeispiel:

```
10  REM VERSCHIEBEN DER FARBEN
20  GRAPHICS 3+16
30  REM ZEICHNE DREI PLAYFIELDS
40  FOR X=1 TO 3
50  COLOR X
60  FOR J=1 TO 5
70  PLOT 5*X+5+J,5:DRAWTO 5*X+5+J,20
80  NEXT J
90  NEXT X
100 REM AENDERE FARBEN IN SEQUENZEN
110 FOR X=708 TO 710
120 POKE X,14
130 FOR WAIT=1 TO 100:NEXT WAIT
140 POKE X,0
150 NEXT X
160 GOTO 100
```

In diesem Beispiel haben wir drei Playfields auf den Bildschirm gezeichnet. Zwei dieser Playfields haben die Farbe schwarz (die Hintergrundfarbe) und eines hat die Farbe weiß. Die Farben der Playfields werden dann zwischen schwarz und weiß umgeschaltet, so daß der Eindruck entsteht, ein weißes Rechteck würde sich über den Bildschirm bewegen. Selbstverständlich hat sich keines der Playfields wirklich bewegt. Wir haben nur die Farben geändert.
Wenn Sie in Ihrem Computer einen GTIA Chip haben, kann man das Ergebnis noch verbessern. Ändern Sie in unserem Programm die folgenden Zeilen und betrachten Sie sich die Auswirkungen:

```
20 GRAPHICS 10
40 FOR X=0 TO 8
60 FOR J=1 TO 8
70 PLOT 8*X+J,5:DRAWTO 8*X+J,170
110 FOR X=705 TO 713
```

#1. 50 SETCOLOR 0,8,6 #2. 30 SETCOLOR 0,4,6

Animation von Zeichen

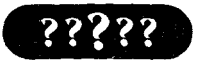
Bis jetzt haben wir eine Animation nur durch Wechseln von Farben erzeugt. Ebenso kann man aber auch durch das Verschieben eines Zeichens eine Animation erreichen. Als erstes wollen wir ein Zeichen auswählen und über den Bildschirm bewegen. Geben Sie das folgende Programm ein und starten Sie es:

```
10 REM BEWEGE EIN ZEICHEN
20 GRAPHICS 0
30 FOR P=0 TO 37
40 PRINT CHR$(0);
50 FOR WAIT=1 TO 100:NEXT WAIT
60 NEXT P
```

Sie sollten jetzt sehen, wie das Graphikzeichen für ein Herz langsam über den Bildschirm geschrieben wird. Aber nehmen wir an, wir wollen ein einzelnes Zeichen über den Bildschirm bewegen. Zu diesem Zweck müßen wir jeweils das vorhergehende Zeichen löschen, wenn ein neues auf den Bildschirm geschrieben wird. Betrachten Sie das folgende Programmbeispiel:

```
10 REM BEWEGE EIN ZEICHEN
20 GRAPHICS 0
30 FOR P=0 TO 37
40 POSITION P+1,0:PRINT CHR$(0)
50 POSITION P,0:PRINT CHR$(32)
60 FOR WAIT=1 TO 100:NEXT WAIT
70 NEXT P
```

In dieser Routine wird jedesmal, wenn die Druckposition um eine Stelle nach rechts verschoben wird, an der vorhergehenden Position ein Leerzeichen CHR$(32) geschrieben. Als Ergebnis sehen wir ein einzelnes Zeichen, das sich über den Bildschirm bewegt.



Welche zwei Zeilen würden Sie ändern, um das Zeichen vertikal zu verschieben?

Antwort #3 _____

Antwort #4 _____

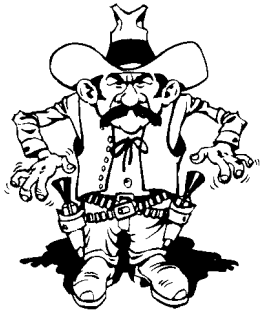
#3. 40 POSITION 0,P+1:PRINT CHR$(0) #4. 50 POSITION 0,P:PRINT CHR$(32)

Zeichenanimation kann auch durch Verschieben eines Zeichen-Codes durch den Bildschirmspeicher erzeugt werden. Betrachten Sie das folgende Beispiel:

```
10 REM VERSCHIEBE ZEICHEN IM BILDSCHIRMSPEICHER
20 GRAPHICS 0
30 SCREENMEM=PEEK(88)+PEEK(89)*256
40 FOR P=0 TO 37
50 POKE SCREENMEM+P+1,64
60 POKE SCREENMEM+P,0
70 FOR WAIT=1 TO 100:NEXT WAIT
80 NEXT P
```

Zuerst wird die Startadresse des Bildschirmspeichers berechnet und dann ein Zeichenwert in den Speicher gePOKEt. Jedes mal, wenn das Zeichen im Speicher um einen Schritt weitergeschoben wird, wird die vorhergehende Speicherstelle gelöscht. Beachten Sie, daß in diesem Fall nicht der ASCII Code die auf dem Bildschirm dargestellten Zeichen repräsentiert. In den Zeilen 50 und 60 unseres Programmes wird ein interner Code mit den Werten 64 (für das Herz) und 0 (für die Leerstelle) verwendet. In Kapitel 10 werden wir diesen internen Code eingehend erklären.
Ein Zeichen kann auch vertikal über den Bildschirm bewegt werden. Der Speicher beginnt in der oberen linken Ecke des Bildschirms und ist so organisiert, daß die Speicherstellen sich fortlaufend erhöhen, wenn Sie etwas Zeichen für Zeichen über den Bildschirm bewegen. Nach 40 Bildschirmzeichen (40 Bytes des Bildschirmspeichers), werden die Zeichen in die nächste Zeile verschoben. Um ein Zeichen vertikal über den Bildschirm zu bewegen, müssen wir das Zeichen in einer Schrittweite von 40 Bytes durch den Bildschirmspeicher verschieben. Lassen Sie es uns versuchen. Ändern Sie in unserem Programm die folgenden Zeilen und starten Sie es erneut:

```
40 FOR P=0 TO 960 STEP 40
50 POKE SCREENMEM+P,128
60 POKE SCREENMEM+P-40,0
```



Aufgabe #16

Schreiben Sie ein Programm, das mit Hilfe der POSITION-Anweisung die Zahl 0 im GRAPHIK-Modus 2 über den Bildschirm bewegt.

Playfield Animation

Die Animation eines Playfields ist ähnlich der eines Zeichens. Lassen Sie uns zur Demonstration im GRAPHIK-Modus 3 ein einzelnes Pixel animierem, genauso, wie wir das eben im GRAPHIK-Modus 0 getan haben. Geben Sie das folgende Programm ein und starten Sie es:

```
10 REM BEWEGE EIN PIXEL IM MODUS 3
20 GRAPHICS 3
30 FOR P=0 TO 37
40 COLOR 1:PLOT P+1,0
50 COLOR 0:PLOT P,0
60 FOR WAIT=1 TO 100:NEXT WAIT
70 NEXT P
```

Für jedes geplottete COLOR 1 Pixel wird ein COLOR 0 (Hintergrundfarbe) -Pixel auf der vorhergehenden Position geplottet.

Welche Anweisung unseres Programmbeispiels würden Sie ändern, um die horizontale Distanz, mit der das Pixel sich zu bewegen scheint, zu verkürzen?

Antwort #5 _____

#5. 30 for P=0 to 20

Lassen Sie uns einen großen Playfieldbereich animieren. Zu diesem Zweck zeichnen wir zuerst einmal ein farbiges Quadrat im GRAPHIK-Modus 7. Danach werden wir es von der rechten Seite des Bildschirmes auf die linke Seite verschieben. Geben Sie das folgende Programm ein und starten Sie es:

```
10 REM PLAYFIELD ANIMATION IN GRAPHIK-MODUS 7
20 GRAPHICS 7
30 REM ZEICHNE EINE FIGUR
40 FOR X=5 TO 10
50 PLOT X,45:DRAWTO X,50
60 NEXT X
100 REM BEWEGE DIE FIGUR
110 FOR X=5 TO 154
120 COLOR 0:PLOT X,45:DRAWTO X,50
130 COLOR 1:PLOT X+5,45:DRAWTO X+5,50
140 NEXT X
```

Um die Figur zu bewegen, haben wir dieselbe Technik verwendet, wie eben für die Bewegung eines einzelnen Pixels. Für jede vertikale Linie, die wir auf der rechten Seite der Figur zeichnen, wird die entsprechende Linie auf der linken Seite der Figur gelöscht. Auch hier wird dies ermöglicht, indem die Linien auf der rechten Seite mit COLOR 1, und die auf der linken Seite mit COLOR 0 (Hintergrundfarbe) gezeichnet werden.
Beachten Sie, daß wir in diesem Programm keine Warteschleife zur Zeitverzögerung eingebaut haben. Je höher die Anzahl der zu verschiebenden Pixel, desto langsamer wird die Animation. Jetzt wollen wir die Figur einmal vergrößern und beobachten, wie sich das auf die Geschwindigkeit der Animation auswirkt. Ändern Sie in unserem Programm die folgenden Zeilen und starten Sie es erneut:

```
50 PLOT X,5:DRAWTO X,75
120 COLOR 0:PLOT X,5: DRAWTO X,75
130 COLOR 1:PLOT X+5,5:DRAWTO X+5,75
```

Die bisher durchgeführten Animationen waren äußerst einfach. Die Verschiebung der Pixel einer komplizierteren Figur wird erheblich schwieriger und benötigt auch mehr Rechenzeit des Computers. Zweidimensionale Animation erweitert auch die Zusammenfassung der Programmroutine. Aufgrund der eingeschränkten Geschwindigkeit von BASIC, müssen gute Playfield Animationen oft in Maschinensprache geschrieben werden.

Player Animation

Die Animation von Figuren auf dem Bildschirm kann durch die Verwendung der Player/Missile Graphik erheblich vereinfacht werden. Zum Beispiel kann eine horizontale Bewegung ganz einfach durch Ändern des Wertes im entsprechenden Horizontal-Positions-Register erreicht werden. Wir wollen jetzt einen Player erstellen um Ihnen das einmal zu zeigen. Geben Sie das folgende Programm ein:

```
100 REM PLAYER/MISSILE ANIMATION
200 REM ERSTELLE PLAYER UND SETZE REGISTER
210 PMBASE=PEEK(106)-4
220 POKE 106,PMBASE-1
230 GRAPHICS 3+16
240 FOR X=512 TO 640
250 POKE PMBASE*256+X,0
260 NEXT X
270 FOR X=562 TO 570
280 POKE PMBASE*256+X,255
290 NEXT X
300 POKE 704,30
310 POKE 559,46
320 POKE 54279,PMBASE
330 POKE 53277,3
```

Starten Sie das Programm. In der Mitte des Bildschirmes sollte ein kleines gelbes Quadrat erscheinen.
Der Player kann jetzt animiert werden, indem man die Werte im Horizontal-Positions Register ändert. Fügen Sie die folgende Routine zu unserem Programm. Sie bewirkt, daß sich das Quadrat ständig von einer Seite zur anderen bewegt.

```
400 REM ANIMIERE PLAYER
405 REM BEWEGE IHN NACH RECHTS
410 FOR X=40 TO 208
420 POKE 53248,X
430 NEXT X
435 REM BEWEGE IHN NACH LINKS
440 FOR X=208 TO 40 STEP -1
450 POKE 53248,X
460 NEXT X
470 GOTO 400
```

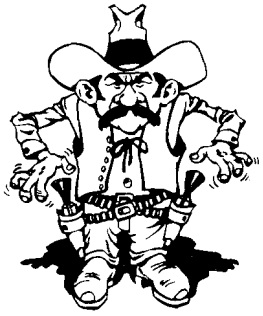
Starten Sie das Programm. Die Bewegung ist nicht nur fließend, sondern auch noch schnell. Wir können das Tempo der Bewegung sogar noch erhöhen, indem wir die Werte des Horizontal-Positions-Registers in einer Schrittweite von drei ändern. Ändern Sie die folgenden Zeilen:

```
410 FOR X=40 TO 208 STEP 3
440 FOR X=208 TO 40 STEP -3
```

Starten Sie das Programm erneut. Das Quadrat bewegt sich schneller, aber die Bewegung ist nicht mehr ganz so fließend.
Jetzt wollen wir den Player einmal in vertikaler Richtung bewegen. Zu diesem Zweck müssen wir dieselbe Technik verwenden, die wir bei der Animation der Playfields bereits kennengelernt haben. Wir müssen die Daten der Player im Speicher hoch bzw. runter schieben. Wenn wir jedes Byte des Players, das ja eine Zeile der Form des Players repräsentiert, um eine Adresse im Speicher nach oben verschieben, müssen wir das jeweils letzte Byte löschen. Das tun wir, indem wir eine 0 in der entsprechenden Adresse ablegen, desgleichen wenn wir die Daten des Players nach unten verschieben. Wir wollen das anhand eines Beispiels verdeutlichen. Fügen Sie die folgenden Zeilen zu unserem Programm hinzu. Beachten Sie, daß nur einige Zeilen geändert werden müssen.

```
400 REM ANIMIERE PLAYER
405 REM BEWEGE IHN NACH UNTEN
410 POKE 53248,114
420 FOR Y=503 TO 631
430 POKE PMBASE*256+Y,0
435 POKE PMBASE*256+Y+9,255
440 NEXT Y
450 REM BEWEGE IHN NACH OBEN
460 FOR Y=631 TO 503 STEP -1
470 POKE PMBASE*256+Y+9,0
480 POKE PMBASE*256+Y,255
490 NEXT Y
500 GOTO 400
```

Starten Sie das Programm. Bewegt sich der Player ständig vom oberen Rand des Bildschirmes zum unteren und umgekehrt?



Aufgabe #17

Schreiben Sie ein Programm, das zwei Player (sie sollten die Form eines Quadrates haben) in unterschiedlicher Geschwindigkeit horizontal zwischen den beiden Seiten des Bildschirms hin und her bewegt.

Anwendungen

Die Möglichkeiten der Animation eröffnen uns völlig neue Dimensionen in der Gestaltung. Dynamische Darstellungen erregen die Aufmerksamkeit des Beobachters. Es gibt viele Wege, wie man Animation verwenden kann. Hier haben wir einige aufgeführt:

1. Blinkende Farben fesseln die Aufmerksamkeit des Programmierers und können so auf wichtige Punkte auf dem Bildschirm hinweisen.

2. Das Bewegen von Figuren hat eine große Bedeutung bei Spielen. Die Bewegung dieser Figuren kann zum Beispiel mit einem Steuerknüppel oder mit Drehreglern kontrolliert werden.

3. Die Animation kann verwendet werden, um innerhalb einer Darstellung zusätzliche Informationen zu vermitteln. Wenn man zum Beispiel einen Ball auf dem Bildschirm zeigt, kann man diesen rollen, hüpfen, oder ruhig liegen lassen.

4. Die Animation kann verwendet werden, um einen Vorgang zu verdeutlichen. Zum Beispiel kann man einen Behälter mit einer Flüssigkeit füllen.

Die Anwendungen der Animation sind zahllos. Jedesmal, wenn Sie eine Bildschirmdarstellung entwerfen, sollten Sie sich fragen, ob nicht eventuell mit einer Animation zusätzliche wichtige Informationen gegeben werden könnten.

Betrachten wir uns zum Schluß noch ein weiteres Programmbeispiel. Wir haben es "Sich schließende Gardinen" genannt. Geben Sie das folgende Programm ein und starten Sie es:

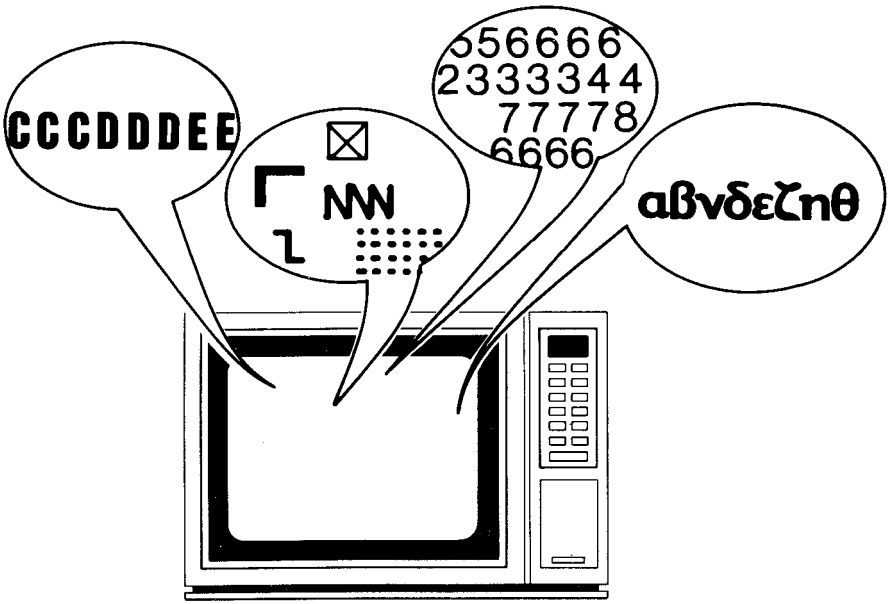
```
100 REM SICH SCHLIESSENDE GARDINEN
200 REM ERSTELLE GRAPHIK BILDSCHIRM
210 GRAPHICS 7+16
220 SETCOLOR 0,0,14
230 COLOR 1
300 REM ZEICHNE VERTIKALE LINIEN BIS ZUR MITTE
310 FOR X=0 TO 79
320 PLOT X,0:DRAWTO X,91:PLOT 159-X,0:DRAWTO 159-X,91
330 SETCOLOR 0,0,(90-X)/6
350 NEXT X
400 GOTO 400
```

Vielleicht könnten mit etwas mehr Aufwand die Gardinen etwas echter dargestellt werden.

KAPITEL 10

Neuer Zeichensatz

Würden Sie gerne Ihren eigenen Zeichensatz erstellen und auf dem Bildschirm darstellen?



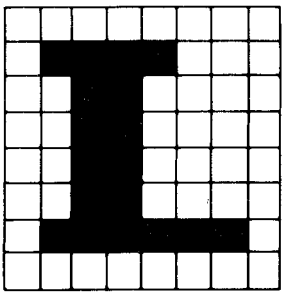
In diesem Kapitel werden Sie lernen:

- neue Zeichensätze zu erstellen
- Mehrfarbenzeichen zu erstellen
- in jedem GRAPHIK-Modus Zeichen darzustellen
- mehrere Zeichensätze auf einmal zu verwenden

Die graphischen Möglichkeiten eines jeden Computers werden wesentlich durch die Fähigkeit, eine Anzahl verschiedener Zeichen darstellen zu können, erweitert. Dies können spezielle Graphikzeichen, Sonderzeichen einer Fremdsprache oder sonstige völlig neue Zeichen sein.
Der ATARI Computer enthält einen vorgefertigten Satz von Zeichen, der sich in einem ROM, dem sogenannten Zeichen-Generator, befindet. Diese Zeichen sind ein fester Bestandteil der Hardware und können nicht geändert werden. Trotzdem ist es nicht unbedingt notwendig, nur diese Zeichen zu verwenden. Stattdessen können neue Zeichen bestimmt, und im Speicher abgelegt werden. Anschließend muß dem Computer gesagt werden, daß er den neuen Zeichensatz verwenden soll.

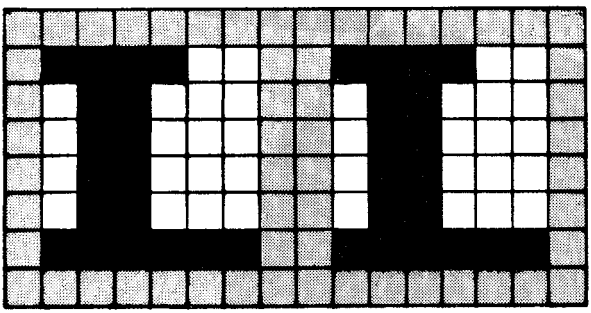
Bestimmen neuer Zeichen

Wir wollen mit Hilfe des Arbeitsblattes für die Zeichenerstellung aus Anhang A ein neues Zeichen entwerfen. Wir werden den Buchstaben L verwenden, jedoch seine Form etwas ändern, so daß er nicht wie ein normales L des ATARI-Zeichensatzes aussieht.
Jedes Zeichen setzt sich aus einer 8x8 Matrix zusammen. Jede Zeile des Zeichens kann durch einen Dezimalwert vertreten werden. Erinnern Sie sich an die Player in Kapitel 8? Richtig, da war es genauso. Da ein Zeichen aus acht Zeilen besteht, benötigt man acht Bytes zur Definition. Unser neuer Buchstabe L würde durch die folgenden Werte bestimmt.

Zeichen-form	Binär-code	Dezimal-wert
	0 0 0 0 0 0 0 0 =	0
	0 1 1 1 1 0 0 0 =	120
	0 0 1 1 0 0 0 0 =	48
	0 0 1 1 0 0 0 0 =	48
	0 0 1 1 0 0 0 0 =	48
	0 0 1 1 0 0 0 0 =	48
	0 1 1 1 1 1 1 0 =	126
	0 0 0 0 0 0 0 0 =	0

Beachten Sie in diesem Beispiel, daß die Ränder der Matrix nicht verwendet werden. Würde man sie doch benutzen, so würden sich zwei nebeneinander dargestellte Zeichen berühren. Das kann von Nutzen sein, aber auch nicht, je nachdem, wofür man den Zeichensatz verwenden will.

Der schattierte Teil sollte nicht verwendet werden, wenn man nicht will, daß sich die einzelnen Zeichen berühren.



Zeichensätze

Ein kompletter Zeichensatz umfasst 128 Zeichen. Jedes dieser Zeichen kann normal oder in inversem Video dargestellt werden. Es ist offensichtlich, daß es eine ganze Menge Arbeit bedeutet, wenn man einen ganzen Zeichensatz neu definieren will.

Betrachten wir ein einfaches Beispiel. Wir werden einen neuen Zeichensatz mit 128 Zeichen erstellen, doch alle 128 Zeichen werden gleich sein. Dazu können wir das eben definierten Zeichen L verwenden.

1. Der erste Schritt ist, einen Bereich im Speicher zu finden, in dem man den neuen Zeichensatz ablegen kann. Wir können ihn in der Nähe der höchsten Speicheradresse ablegen, genau so, wie wir es bereits mit den Playern und Missiles in Kapitel 8 gemacht haben. Da die Bestimmung eines Zeichens acht Bytes beansprucht, benötigen 128 Zeichen 1024 Bytes des Speichers. Dies entspricht genau vier 256 Byte Seiten (Pages).
   Wieder müssen wir den Computer täuschen, damit er den Bildschirmspeicher unterhalb des Bereiches, in dem wir unsere Zeichen ablegen, plaziert. Hierzu muß der Zeiger für die höchste Speicheradresse neu initialisiert werden. Um das Programm zu starten, geben Sie die folgenden Zeilen ein:

```
100 REM ZEICHENSATZ DEMONSTRATION
200 REM RESERVIERE SPEICHERPLATZ FÜR ZEICHENSATZ
210 CHARSET=PEEK(106)-4
220 POKE 106,CHARSET-1
230 GRAPHICS 0
```

CHARSET ist die Basis-Adresse des Zeichensatzes, ausgedrückt in 256 Bytes Seiten.

2. Der nächste Schritt ist, unseren neuen Zeichensatz in den Speicher zu schreiben. Wir können die Zeichen-Bestimmungs-Bytes in DATA-Anweisungen ablegen. Fügen Sie die folgenden Zeilen zu unserem Programm hinzu:

```
300 REM PLAZIERE NEUEN ZEICHENSATZ IM SPEICHER
310 FOR C=0 TO 127
320 RESTORE
330 FOR B=0 TO 7
340 READ N:POKE CHARSET*256+(C*8)+B,N
350 NEXT B
360 NEXT C
380 DATA 0,120,48,48,48,48,126,0
```

Beachten Sie, daß CHARSET mit 256 Bytes/Seite multipliziert wird, um die Speicheradresse des neuen Zeichensatzes zu berechnen.

3. Und schließlich brauchen wir nur noch ANTIC mitzuteilen, wo sich die Basis-Adresse unseres neuen Zeichensatzes befindet. Fügen Sie die folgenden Zeilen ein:

```
400 REM SETZT ZEICHENSATZ ZEIGER
410 POKE 756,CHARSET
```

Starten Sie das Programm. (Es dauert etwas 30 Sekunden.) Sie sollten jetzt einen Bildschirm voll mit L's sehen. Wir haben sogar das Leerzeichen in ein L verwandelt. Lassen Sie uns das Leerzeichen wieder zurückverwandeln. Drücken Sie <SYSTEM RESET> und fügen Sie dann die folgenden Zeilen in unser Programm ein:

```
301 FOR X=0 TO 7
302 POKE CHARSET*256+X,0
303 NEXT X

310 FOR C=1 TO 127
```

Starten Sie jetzt das Programm von neuem. Drücken Sie jede beliebige Taste der Tastatur. Wir haben tatsächlich einen Zeichensatz erstellt, der nur aus L's besteht. Die auf den Tasten aufgedruckten Zeichen, haben nicht mehr die gleiche Bedeutung.

Ändern von Zeichensätzen

In unserem letzten Beispiel haben wir die Zeichen-Bestimmungs-Bytes für das Leerzeichen in unseren Zeichensatzspeicher geschrieben. Doch wie konnten wir wissen, an welcher Position innerhalb des Zeichensatzes das Leerzeichen liegt?
Jede der Tasten auf der Tastatur hat einen Tastatur-Code (interner Code). Wenn Sie eine Taste drücken, lokalisiert der Computer das dem entsprechenden Tastatur-Code zugehörige Zeichen innerhalb des Zeichensatzes und stellt es dann auf dem Bildschirm dar. Wenn Sie die Form des Zeichens für diesen bestimmten Tastatur-Code ändern, wird entsprechend das geänderte Zeichen dargestellt.
Die Werte des Tastatur-Codes sind identisch mit der Position eines Zeichens im Zeichensatz. Ein Tastatur-Code von 0 bedeutet, daß das Zeichen an der ersten Position im Zeichensatz liegt, entsprechend ist das dem Tastatur-Code 127 zugehörige Zeichen an letzter Stelle zu finden.

Dieser Tastatur-Code kann aus den ASCII-Werten (Tabelle 5-1) berechnet werden. Die folgende Tabelle zeigt die Berechnung, die durchgeführt werden muß, um die ASCII-Werte in die entsprechenden Tastatur-Werte umzuwandeln.

ATARI ASCII Wert	Berechnung
0 bis 31	Wert + 64
32 bis 95	Wert - 32
96 bis 127	keine

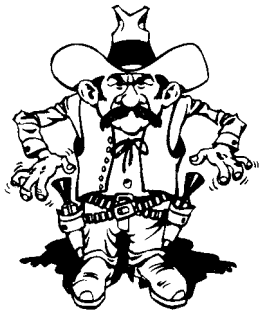
Da das Leerzeichen einen ASCII-Wert von 32 hat, ist der Tastatur-Code 32-32 = 0.
Jetzt ist es aber Zeit für ein Beispiel. Wir wollen den ATARI Zeichensatz aus dem ROM in den Speicher kopieren, so daß wir eines der Zeichen ändern können. Löschen Sie alle alten eventuell noch im Speicher befindlichen Programme und geben Sie die folgenden Zeilen ein:

```
100 REM MODIFIZIERUNG DES ATARI ZEICHENSATZES
200 REM RESERVIERE SPEICHERPLATZ FÜR ZEICHENSATZ
210 CHARSET=PEEK(106)-4
220 POKE 106,CHARSET-1
230 GRAPHICS 0
300 REM KOPIERE ATARI ZEICHENSATZ
310 FOR X=0 TO 1023
320 POKE CHARSET*256+X,PEEK(224*256+X)
330 NEXT X
```

Ein bestimmter Speicherbereich wird wie eben reserviert. Das ATARI - Zeichensatz-ROM befindet sich bei Seite (Page) 224. Es werden alle 1024 Zeichen aus dem ROM gelesen und in den reservierten Speicherbereich gePOKEt.
Jetzt wollen wir das große T in ein Leerzeichen verwandeln. Die acht Zeichen-Definitions-Bytes für ein Leerzeichen wären alle 0. Um den Tastatur-Code für das T zu finden, müssen wir zuerst den ASCII-Wert suchen (siehe Tabelle 5-1). Der ASCII-Wert für das T ist 84. Daraus ergibt sich ein Tastatur-Code von 84-32=52. Da jedes Zeichen des Zeichensatzes acht Bytes benötigt, startet die Daten für das Zeichen T 8x52=416 Bytes vom ersten Byte des Zeichensatz Speicherbereichs. Fügen Sie die folgenden Zeilen zu unserem Programm hinzu, um das Zeichen T in ein Leerzeichen zu verwandeln:

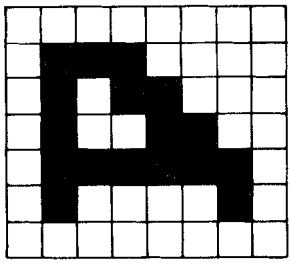
```
400 REM MODIFIZIERE DEN BUCHSTABEN T
410 FOR X=0 TO 7
420 POKE CHARSET*256+416+X,0
430 NEXT X
500 REM SETZE ZEICHENSATZ BASIS ZEIGER
510 POKE 756,CHARSET
```

Starten Sie das Programm. Drücken Sie jetzt die Taste für den Buchstaben T. Ist ein Leerzeichen ausgedruckt worden? Alle anderen Zeichen sollten normal vorhanden sein. Listen Sie Ihre Programm. Beachten Sie, daß alles beim Alten geblieben ist, mit Ausnahme des Buchstaben T.



Aufgabe #18

Ändern Sie den ATARI Zeichensatz derart, daß dieses Zeichen anstelle des großen Buchstaben A ausgedruckt wird.



Drucken von Zeichen auf dem Graphik Bildschirm

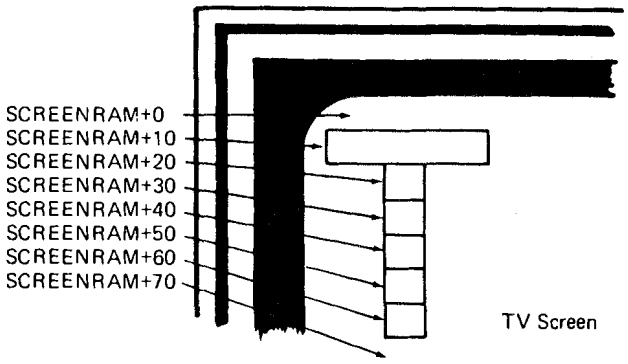
Die GRAPHIK-Modi 0, 1, und 2 werden als die sogenannten Textmodi bezeichnet. Trotzdem besteht manchmal die Notwendigkeit auch in den GRAPHIK-Modi 3 bis 8 Text zu schreiben. Betrachten wir hierzu zuerst einmal die Zweifarben-Modi 4, 6, und 8.

GRAPHIK-Modi 4, 6, und 8

In den GRAPHIK-Modi 4, 6, und 8 können jeweils zwei Farben auf einmal dargestellt werden. Die Pixel einer jeden in diesen Modi dargestellten Figur sind entweder ein- oder ausgeschaltet, genau wie in GRAPHIK-Modus 0. Jedes Pixel eines Zeichens benötigt ein Bit innerhalb des Bildschirmspeichers. In diesen Zweifarben-Modi ist es ganz einfach Zeichen darzustellen, da das Zeichensatz ROM direkt in den Bildschirmspeicher übertragen werden kann.
Betrachten Sie unser Beispiel. Geben Sie die folgenden Zeilen ein und starten Sie das Programm:

```
100 REM ZEICHEN IN DEN MODI 4, 6, UND 8
200 REM SETZE GRAPHIK-MODUS
210 GRAPHICS 4
300 REM FINDE BILDSCHIRMSPEICHER POSITION
310 SCREENRAM=PEEK(88)+PEEK(89)*256
400 REM SCHREIBE DEN BUCHSTABEN T
420 FOR C=0 TO 7
430 POKE SCREENRAM+C*10,PEEK(224*256+416+C)
440 NEXT C
```

Der Buchstabe T sollte in der linken oberen Ecke des Bildschirmes ausgedruckt worden sein. Zeile 430 bedarf einer Erklärung. Da jeweils 10 Bytes des Bildschirmspeichers für jede horizontale Mode-Line in GRAPHIK-Modus 4 verwendet werden (siehe auch Tabelle 7-1), muß jede Zeile des Buchstaben T nach jeweils 10 Bytes im Bildschirmspeicher plaziert werden.



Die zum Festlegen des Zeichens notwendigen Daten werden aus dem ATARI - Zeichensatz-ROM, das bei Adresse 224*256 beginnt, ausgelesen. Zu dieser Adresse haben wir noch 416 Bytes addiert, um, wie zuvor, den Buchstaben T zu erhalten.
Ändern Sie die folgenden Zeilen, um den Buchstaben T in GRAPHIK-Modus 6 zu sehen.

```
210 GRAPHICS 6

430 POKE SCREENRAM+C*20,PEEK(224*256+416+C)
```

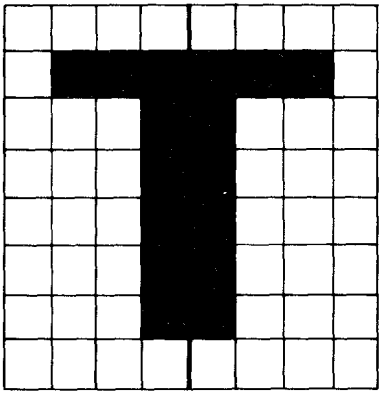
Beachten Sie, daß in Modus 6 20 Bytes für jede Mode Line benötigt werden.

GRAPHIK-Modi 3, 5, und 7

Genauso können wir auch in den GRAPHIK-Modi 3, 5, und 7 Text ausdrucken. Diese Modi sind Vierfarben-Modi. Daher benötigt jedes in diesen Modi dargestellte Pixel zwei Bits im Bildschirmspeicher. Die einzelnen Bit-Paare und die zugehörigen Farbregister sind nachfolgend aufgeführt:

Bit-Paar	Farbregister
00	Hintergrund
01	0
10	1
11	2

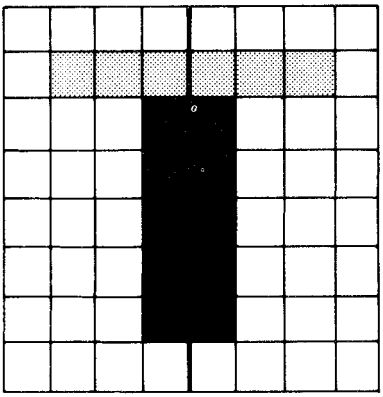
Zur Darstellung von Zeichen in den GRAPHIK-Modi 3, 5, und 7, werden für jede Zeile der Zeichenmatrix acht Bit-Paare bzw. zwei Bytes benötigt. Nehmen wir zum Beispiel an, unser Buchstabe T könnte mit einer Farbe von Register 0 wie gezeigt dargestellt werden. Beachten Sie, daß das Bit-Paar 01 für jeden Zeichenpunkt verwendet wird, der mit der Farbe des Farbregisters 0 dargestellt werden soll. Das Bit-Paar 00 entspricht der Hintergrundfarbe.

Zeichen-form	linkes Byte	rechtes Byte
	00 00 00 00 = 0	00 00 00 00 = 0
	00 01 01 01 = 21	01 01 01 00 = 84
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 00 = 0	00 00 00 00 = 0

Insgesamt sind 16 Bytes notwendig, um das Zeichen darzustellen. Jetzt wollen wir die Zeichendaten in einen GRAPHIK-Modus 5 Bildschirm POKEn, um zu sehen, was passiert. Geben Sie das folgende Programm ein und starten Sie es:

```
100 REM ZEICHEN IN DEN MODI 3, 5, UND 7
200 REM SETZE GRAPHIK BILDSCHIRM
210 GRAPHICS 5
220 SCREENRAM=PEEK(88)+PEEK(89)*256
300 REM STELLE BUCHSTABEN T AUF DEM BILDSCHIRM DAR
310 FOR X=0 TO 7
320 FOR I=0 TO 1
330 READ N:POKE SCREENRAM+X*20+I,N
340 NEXT I
350 NEXT X
390 DATA 0,0,21,84,1,64,1,64,1,64,1,64,1,64,0,0
```

In diesem Programm haben wir keinen Zeichensatz erstellt oder geändert. Die Daten für den Buchstaben T haben wir in der DATA-Anweisung abgelegt. Selbstverständlich könnten wir diese Daten wie zuvor auch im Speicher ablegen.
Und jetzt kommt der spannende Moment. Es ist eine einfache Aufgabe, einige der Bit-Paare so zu ändern, daß sie ein anderes Farbregister vertreten. Lassen Sie uns den oberen Teil des Buchstaben T derart ändern, daß Farbregister 2 statt 0 verwendet wird. Das neue Diagramm mit den Werten sieht dann entsprechend unserem Bild so aus:

Zeichen-form	linkes Byte	rechtes Byte
	00 00 00 00 = 0	00 00 00 00 = 0
	00 11 11 11 = 63	11 11 11 00 =252
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 01 = 1	01 00 00 00 = 64
	00 00 00 00 = 0	00 00 00 00 = 0

Ändern Sie die Werte in Zeile 390.

```
390 DATA 0,0,63,252,1,64,1,64,1,64,1,64,1,64,0,0
```

Starten Sie jetzt das Programm. Ist der obere Teil des T blau? Mehrfarbenzeichen können auch im ANTIC-Modus 4 produziert werden. Für diesen Modus müssen Sie allerdings die Display List ändern.

Anwendung

Sicherlich haben Sie schon ein dutzend gute Ideen, wie Sie selbst erstellte Zeichensätze einsetzen können. Hier sind noch einige zusätzliche Ideen:

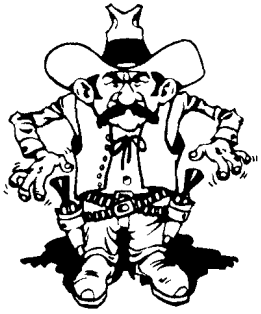
1. Entwerfen und erstellen Sie einen griechischen Zeichensatz, oder einen speziellen Satz von Buchstaben und Zahlen.

2. Entwerfen Sie Ihre eigenen Graphikzeichen und stellen Sie diese dann zu einem Bild zusammen.
3. Entwerfen Sie Mehrfarbenzeichen - sowohl Buchstaben und Zahlen, als auch Graphikzeichen.

4. Speichern Sie mehrere Zeichensätze im Speicher und schalten Sie dann je nach Bedarf hin und her.

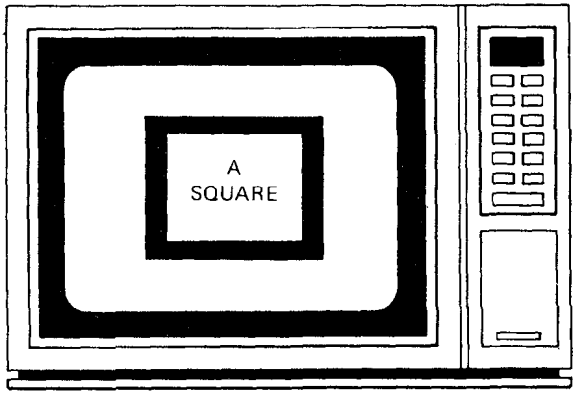
5. Entwerfen Sie einen Satz spezieller Zeichen, die Sie zum Zeichnen elektronischer Diagramme verwenden können.

6. Verwenden Sie den Zeichensatz zur Animation.



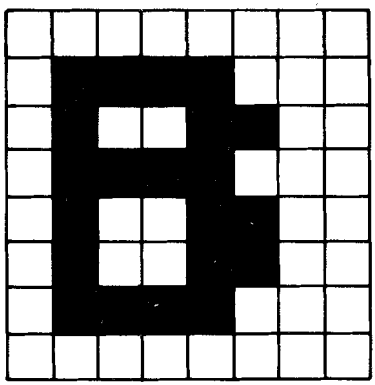
Aufgabe #19

Schreiben Sie ein Programm, das die folgende Darstellung im GRAPHIK-Modus 6 erzeugt.



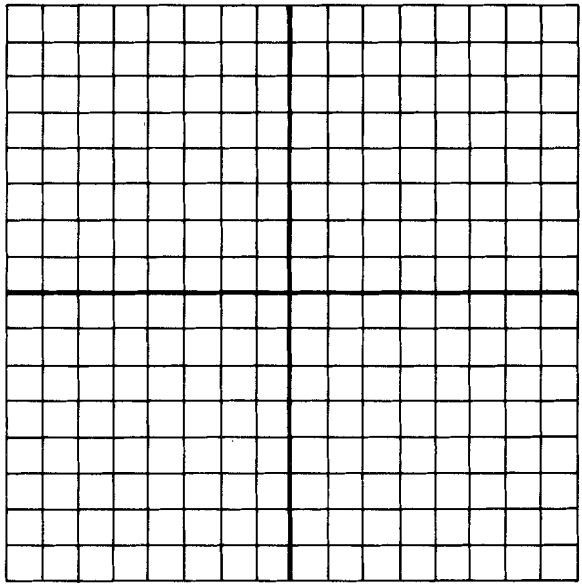
Zusammenfassung von Kapitel 10

1. Welche Dezimalwerte können verwendet werden, um jede Zeile dieses Zeichens zu repräsentieren?



128	64	32	16	8	4	2	1	Decimal Value

2. Zeichnen Sie ein großes T, indem Sie vier 8x8 Zeichen verwenden



3. Wieviele Bytes Speicherplatz werden für ein einfarbiges Zeichen benötigt?

4. Wieviele Bytes Speicherplatz werden für einen Satz von 128 einfarbigen Zeichen gebraucht?

5. Wie lauten die Tastatur-Codes für die folgenden Zeichen?

   a. 4

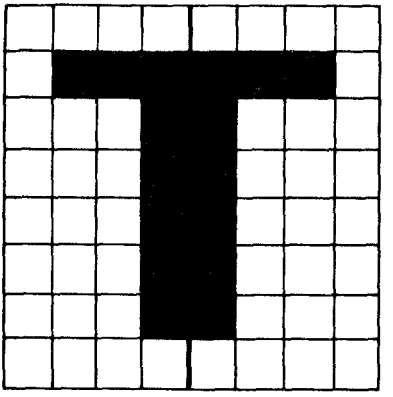
   b. S

   c. W

6. Der normale Zeichensatz sollte bei Adresse 56344 beginnen. Wie lautet die Startadresse der Daten für den Buchstaben A?

7. Eine Mode Line in GRAPHIK-Modus 6 benötigt 20 Bytes.. Wenn die linke Seite der ersten Mode Line direkt bei der Adresse SCREENRAM beginnt, wie lautet dann die Adresse zu Beginn der zweiten Mode Line?

8. Berechnen Sie die Dezimalwerte der 16 zur Darstellung des Buchstaben T benötigten Bytes unter Verwendung von Farbregister 2.

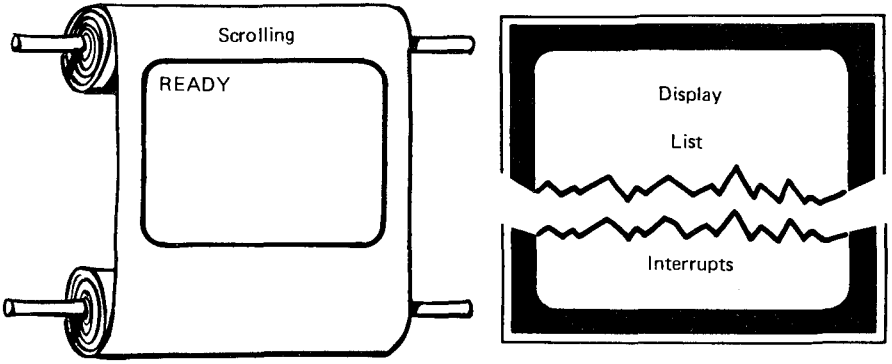


linkes Byte	rechtes Byte

KAPITEL 11

Die Display List

Mit der Display List haben wir uns bereits in Kapitel 7 beschäftigt, als wir mehrere Modi auf einem Bildschirm verknüpft haben. Doch Display Lists können auch für einige andere interessante Effekte eingesetzt werden.

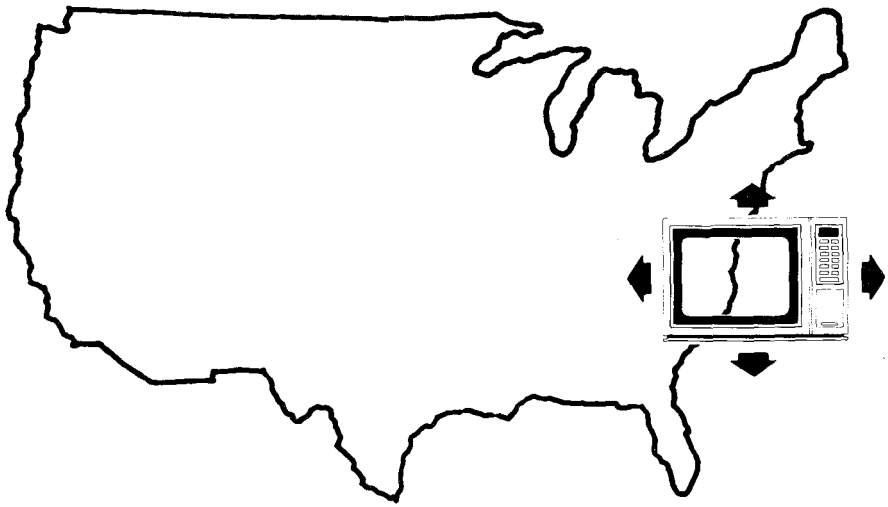


In diesem Kapitel werden Sie lernen:

- Figuren vertikal und horizontal zu scrollen
- Display List Interrupts zu verwenden

Scrolling

Manchmal kann es vorkommen, daß wir gerne mehr Informationen darstellen möchten, als auf einen Bildschirm passen. Nehmen wir einmal an, Sie wollten eine detailierte Karte der Vereinigten Staaten erstellen. Es gibt keinen GRAPHIK-Modus, in dem sich eine solch große Karte richtig darstellen ließe. Trotzdem könnten wir die Kartendaten in den Bildschirmspeicher legen, sie durch diesen scrollen, um dann die verschiedenen Teile der darstellen zu können.



Man kann sich das so vorstellen, als würde man den Fernsehschirm verschieben, um die einzelnen Teile sehen zu können.

Vertikales Scrollen

Diejenigen unter uns, die bereits viel mit dem Computer arbeiten, werden mit dem vertikalen Scrollen bereits vertraut sein. Wenn wir zum Beispiel ein Programm listen, so werden die einzelnen Programmzeilen vertikal, Zeile für Zeile über den Bildschirm gescrollt.

Allgemein gesagt kann man feststellen, daß Daten durch den Bildschirmspeicher geschoben werden können, um ein Scrolling zu erzeugen. Das kann ein langer und komplexer Prozess sein, je nachdem, wieviele Daten verschoben werden müssen. Der ATARI Computer verfügt jedoch über eine einfache Möglichkeit, den Bildschirmzeiger zu ändern und die gesamten Daten an ihrem Platz zu lassen. So ist ein einfacheres und schnelleres Scrolling möglich. Betrachten Sie das folgende Beispielprogramm, damit Sie besser verstehen, was wir meinen. Geben Sie das Programm ein:

```
100 REM BEWIRKT VERTIKALES SCROLLING IN MODUS 0
200 REM FUELLT BILDSCHIRMSPEICHER BEREICH
210 SCREENRAM=PEEK(88)+PEEK(89)*256
220 FOR X=0 TO 959
230 POKE SCREENRAM+X,34
240 NEXT X
300 REM ERHOEHT BILDSCHIRMZEIGER
310 DLIST=PEEK(560)+PEEK(561)*256
320 FOR X=1 TO 20
330 SCREENRAM=SCREENRAM+40
340 HI=INT(SCREENRAM/256)
350 LO=SCREENRAM-HI*256
355 REM POKE BILDSCHIRMADRESSE IN DIE DISPLAY LIST
360 POKE DLIST+4,LO:POKE DLIST+5,HI
370 FOR WAIT=1 TO 100:NEXT WAIT
380 NEXT X
```

Starten Sie das Programm. Die Bildschirmdaten werden über den Bildschirm gescrollt.
Erinnern Sie sich an Kapitel 7. Die Display List für einen GRAPHIK-Modus 0 Bildschirm sieht so aus:

```
112 ┐
112 ├──→ acht leere Zeilen (dreimal)
112 ┘
66  ──→ LMS-Anweisung
64  ┐   Adresse des Bildschirmspeichers
156 ┘   (Diese Werte können sich unterscheiden, wenn Sie nicht
2   ┐    48 K RAM in Ihrem Computer haben)
2   │
2   │
2   │
2   │
2   │
2   ├──→ Mode-Bytes für GRAPHIK-Modus 0
2   │
2   │
2   │
2   │
2   │
2   ┘
```

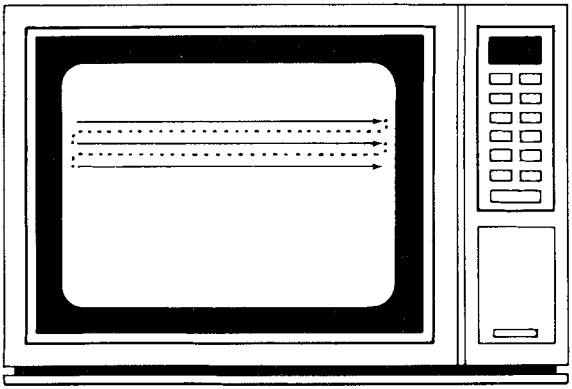
```
2 ┐
2 │
2 │
2 │
2 │
2 ├─► Mode-Bytes für GRAPHIK-Modus 0
2 │
2 │
2 │
2 ┘
65  ──► Sprungadresse
32  ├─► Adresse des Display List Speichers
156 │   (Diese Werte können sich unterscheiden, wenn Sie nicht
          48 k RAM in Ihrem Computer haben)
```

Die LMS-Anweisung in der Display List zeigt auf die Bildschirmspeicher-Adresse. Wenn man diesen Wert nun ändert, verschieben wir auf einfachste Weise die obere linke Ecke des Bildschirms zu einer neuen Position im Bildschirmspeicher. Da der GRAPHIK-Modus 0 pro Zeile 40 Bytes benötigt, verursacht eine Verschiebung um jeweils 40 Bytes das Scrollen einer Bildschirmzeile. Der Bildschirm scheint in Schritten von je einer Mode Line bzw. 8 Scan Lines verschoben zu werden.

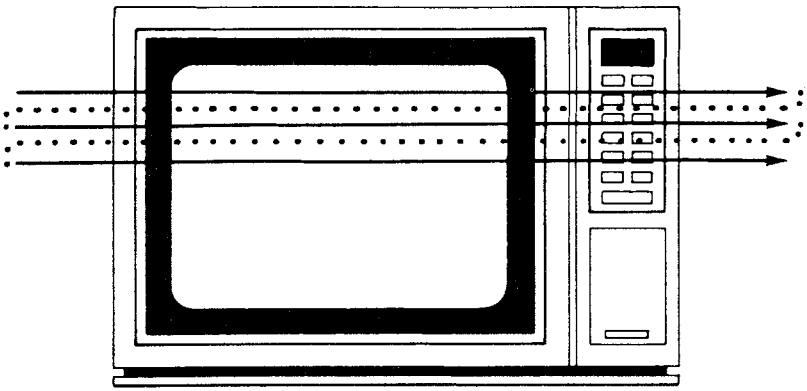
Horizontales Scrollen

Horizontales Scrollen ist nicht so einfach, wie das vertikale Scrollen. Lassen Sie uns einen näheren Blick auf die Organisation des Bildschirmspeichers werfen.

Der Bildschirmspeicher ist abschnittweise von der linken Seite des Bildschirmes zur rechten Seite organisiert. Das Byte, das den Beginn der zweiten Mode Line vertritt folgt direkt dem Byte, das das rechte Ende der ersten Mode Line kennzeichnet. Versucht man den Zeiger des Bildschirmspeichers nur ein Byte vorzuschieben, würde das Zeichen auf der linken Seite der ersten Zeile in die Position des letzten Zeichens in der ersten Zeile wandern. Das folgende Bild zeigt die Anordnung der Daten im Speicher:



Die Adresse der einzelnen Bildschirmpositionen erhöhen sich von links nach rechts und von oben nach unten.
Auf Grund dieser Organisation des Bildschirmspeichers, kann horizontales Scrollen nicht auf die selbe Art wie vertikales Scrollen ermöglicht werden. Das Problem kann übergangen werden, indem man jede horizontale Zeile größer macht als der Bildschirm ist. Das bedeutet, daß zusätzlicher Bildschirmspeicher benötigt wird. Das folgende Bild zeigt, wie die Anordnung im Bildschirmspeicher dann aussehen müßte:



Normalerweise gibt es in den Display Lists der einzelnen GRAPHIK-Modi keine Möglichkeit den Speicherbereich des Bildschirmspeichers zu erweitern. Dieses Problem läßt sich lösen, indem man seine eigene Display List schreibt. Jede Mode Line erfordert eine Display List Anweisung, um die richtige Adresse im Bildschirmspeicher zu setzen. Das ist gar nicht so schwierig, wie es vielleicht klingt.
Das folgende Programm produziert ein horizontales Scrolling. Geben Sie die Zeilen ein und starten Sie:

```
100 REM HORIZONTALES SCROLLEN IN MODUS 0
200 REM ERSTELLE NEUE DISPLAY LIST AB PAGE SIX
210 POKE 1536,112:POKE 1537,112:POKE 1538,112
220 FOR X=1 TO 24
230 POKE 1536+3*X,66
240 POKE 1536+3*X+1,0
250 POKE 1536+3*X+2,X
260 NEXT X
270 POKE 1611,65:POKE 1612,0:POKE 1613,6
280 POKE 560,0:POKE 561,6
300 REM SCROLLE SPEICHERDATEN
310 FOR X=0 TO 215
320 FOR Y=1 TO 24
330 POKE 1536+3*Y+1,X
340 NEXT Y
350 NEXT X
360 GOTO 300
```

Die von uns erstellte Display List unterscheidet sich etwas von einer normalen Display List. Anstatt eines einzelnen Mode Bytes für jede Mode Line, haben wir ein LMS-Byte gefolgt von der Bildschirmspeicheradresse jeder Mode Line verwendet. Wir haben die Display List auf Seite 6 des Speichers abgelegt. Seite 6 startet bei der Dezimaladresse 1536.
Um Daten zu scrollen, müssen die Bildschirmspeicheradressen jeder Mode Line geändert werden. Dies ist auch der Grund, weshalb die obersten Zeilen des Bildschirmes vor den unteren bewegt wurden.

Fine Scrolling

Der ATARI Computer ist sogar in der Lage eine einzelne Scan Line vertikal zu verschieben. Betrachten Sie das folgende Programm um zu sehen, was das bedeutet:

```
100 REM VERTIKALES FINE SCROLLING
200 REM AKTIVIERE VERTIKALES FINE SCROLLING BIT IN DISPLAY LIST
210 DLIST=PEEK(560)+PEEK(561)*256
220 FOR X=10 TO 20:POKE DLIST+X,34:NEXT X
300 REM FUELLE BILDSCHIRMSPEICHER MIT DATEN
310 SCREENRAM=PEEK(88)+PEEK(89)*256
320 FOR X=0 TO 959:POKE SCREENRAM+X,34:NEXT X
400 REM BEGINNE FINE SCROLLING
410 FOR Y=0 TO 7
420 POKE 54277,Y
425 FOR WAIT=1 TO 50:NEXT WAIT
430 NEXT Y
440 GOTO 400
```

Als erstes wird das vertikale Fine-Scrolling-Bit (Bit 5) für jede Mode Line, die gescrollt werden soll, gesetzt. Die Gesamtzahl der vertikal zu bewegenden Scan Lines wird dann in das Vertikal-Fine-Scrolling Register gePOKEt. Es können nicht mehr als 16 Scan Lines "fine gescrollt" werden. Um Figuren über den gesamten Bildschirm zu bewegen, muß man normales und Fine Scrolling kombinieren.
Vielleicht haben Sie bemerkt, daß der Bildschirm manchmal aufblinkt. Dies liegt daran, daß sich die Werte des Fine-Scrolling Registers ändern. Normalerweise sollte dieses Register nur in der Zeit geändert werden, in der der Elektronenstrahl vom unteren Bildschirmrand wieder zum oberen Rand zurückkehrt. Dies wird vertical blank genannt. Zugriff zu diesem vertical blank erfordert die Verwendung von Maschinensprache-Programmen. Überhaupt ist ein qualitativ hochwertiges Scrollen nur in Maschinensprache möglich.

Display List Interrupts

Eine der tollsten Möglichkeiten haben wir uns für den Schluß aufgehoben. Die Verwendung von Display List-Interrupts ist normalerweise für den BASIC-Programmierer nicht möglich, doch können wir verschiedene Beispiele zeigen.
Lassen Sie uns herausfinden, was man mit einem Display List Interrupt machen kann. Betrachten Sie das folgende Programm:

```
100 REM DISPLAY LIST INTERRUPT DEMONSTRATION
200 REM SETZE DLI IN DISPLAY LIST
210 DLIST=PEEK(560)+PEEK(561)*256
220 POKE DLIST+10,130
300 REM LEGE DLI SERVICE ROUTINE IM SPEICHER AB
```

```
310 FOR X=0 TO 10
320 READ N:POKE 1536+X,N
330 NEXT X
360 DATA 72,169,22,141,10,212,141,24,208,104,64
500 REM AKTIVIERE DLI
510 POKE 512,0:POKE 513,6
520 POKE 54286,192
```

Ist der untere Teil des Bildschirmes jetzt orange?
Das Programm arbeitet ungefähr wie folgt: Jedes Mode Byte in der Display List kann so gesetzt werden, daß es eine Interrupt Service-Routine aktiviert. Bit 7 des jeweiligen Mode Bytes setzt den Display List-Interrupt. Das entspricht dem Addieren des Werts 128 zum normalen Mode Byte des gerade verwendeten GRAPHIK-Modus. Wenn dieser Interrupt während der normalen Ausführung der Display List entdeckt wird, überprüft der Computer anhand des Inhaltes der Adresse 54286, ob der Interrupt aktiviert wurde. Ist der Inhalt der Adresse 54286 gleich 192, so ist er aktiviert. Dann überprüft der Computer die Adressen 512 und 513, um festzustellen, wo sich die Interrupt Service-Routine innerhalb des Speichers befindet. In unserem Beispiel befindet sich die Routine auf Seite 6 bzw. bei der Dezimaladresse 1536.
Die Interrupt Service-Routine ist in Maschinensprache geschrieben. In unserem Beispiel dient sie zur Änderung des Farbwertes von Farbregister 2. Wenn die Mode Line mit dem gesetzen Display List Interrupt Bit erreicht wird, wird die Hintergrundfarbe in orange geändert. Während des vertikal blank Prozesses wird sie wieder auf den Ursprungswert blau gesetzt. Daher bleibt der obere Teil des Bildschirmes blau. Für die Fortgeschritteneren wollen wir einen kurzen Blick auf die Maschinenspracheroutine werfen.

Assembler Code	Dezimalcode
PHA	72
LDA COLOR	169,22
STA REG2	141,10,212
STA WSYNC	141,24,208
PLA	104
RTI	64

Eine Erklärung der Routine wäre nicht sinnvo , bis Sie mit der Maschinensprache vertraut sind. Der Assembler Code kann sehr einfach verwendet werden, um einen Farbwert (hier 22) in das Farbregister 2 zu legen.

Wir wollen an unserem Programm einige Änderungen durchführen und einen zweiten Interrupt in einer anderen Mode Line einbauen. Außerdem fügen wir noch eine weitere Interrupt Service-Routine ein. Ändern, bzw. fügen Sie die folgenden Programmzeilen hinzu:

```
230 POKE DLIST+17,130

310 FOR X=0 TO 15

400 REM ZWEITE DLI SERVICE ROUTINE
410 FOR X=0 TO 15
420 READ N:POKE 1556+X,N
430 NEXT X
460 DATA 72,169,66,141,10,212,141,24,208,169,0,141,0,2,104,64
```

Starten Sie das Programm. Jetzt haben wir noch eine weitere Farbe auf den Bildschirm gebracht. In der Tat könnten wir jeder Mode Line des GRAPHIK-Modus 0 Bildschirmes eine andere Farbe geben. Auch hier ist wieder die Maschinensprache erforderlich. Vielleicht haben Sie jetzt Interesse bekommen die Maschinensprache zu erlernen.

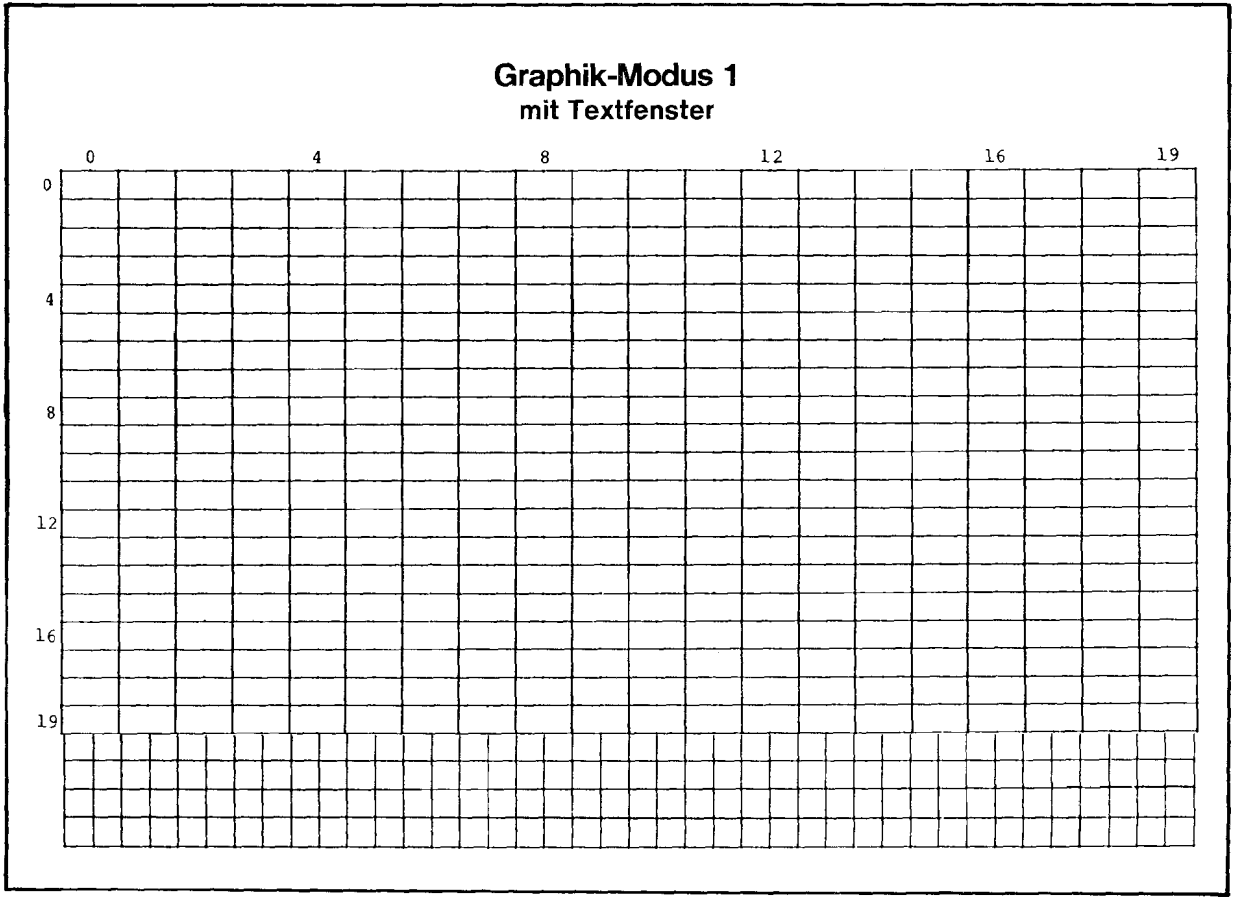
ANHANG A

Arbeitsblätter für Bildschirmentwürfe

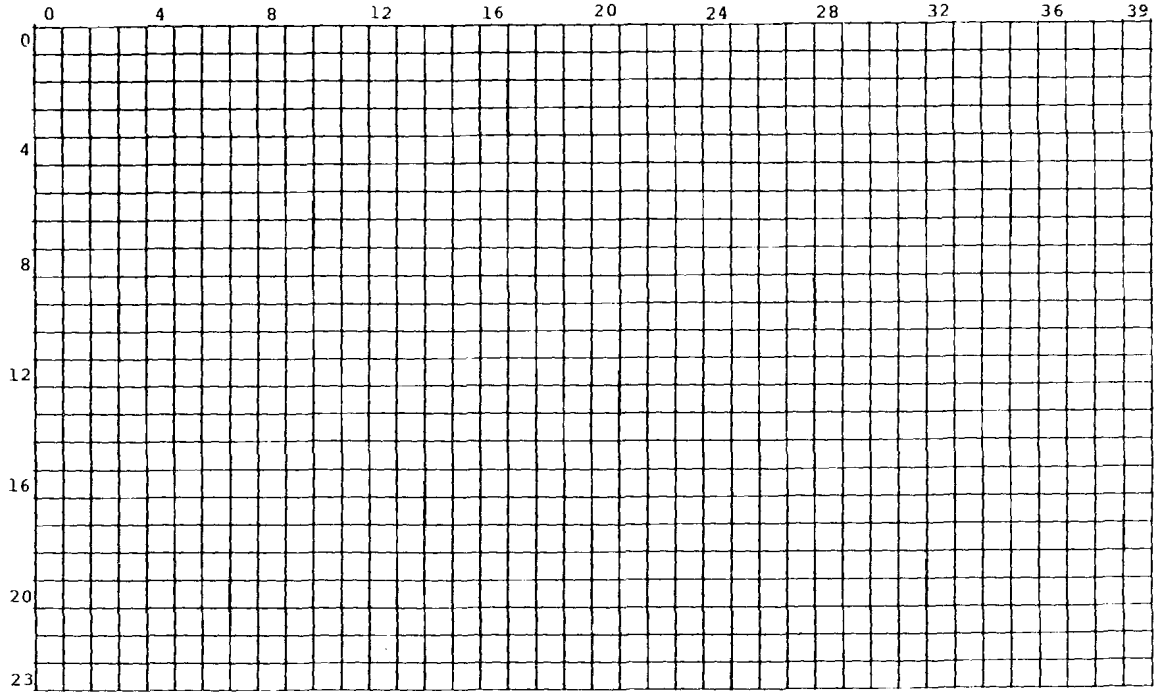
Der letzte Schliff

Jetzt, wo Sie diesen Punkt erreicht haben, werden Sie sich sicher nach einigen guten Abschlußworten sehnen. Ich kann nichts über Ihre Meinung zu diesem Buch sagen, doch will ich Ihnen einige Denkanstöße geben, über die Sie grübeln können.
Einer der großartigsten Aspekte bei der Verwendung von Computern ist, daß man die Kontrolle über den Vorgang und das Medium als solches hat. Sie können nicht nur graphische Figuren auf dem Bildschirm entwerfen, sondern diese ändern, manipulieren, oder völlig neu kreieren. Sie haben die Macht, ihnen Leben zu verleihen, ihre Bewegungen und Erscheinungsformen zu kontrollieren.
Dieses Buch hat sich mit den graphischen Fähigkeiten des ATARI Computer beschäftigt. Sie alleine haben die Möglichkeit animierte graphische Figuren zu entwerfen und zu entwickeln, die keine andere Person jemals sehen wird. Sie werden jetzt fragen, warum nur Sie diese Möglichkeit besitzen? Das ist ganz einfach, denn die Figuren, die Sie entwerfen sind ein Produkt Ihrer

"Sprühenden Ideen"

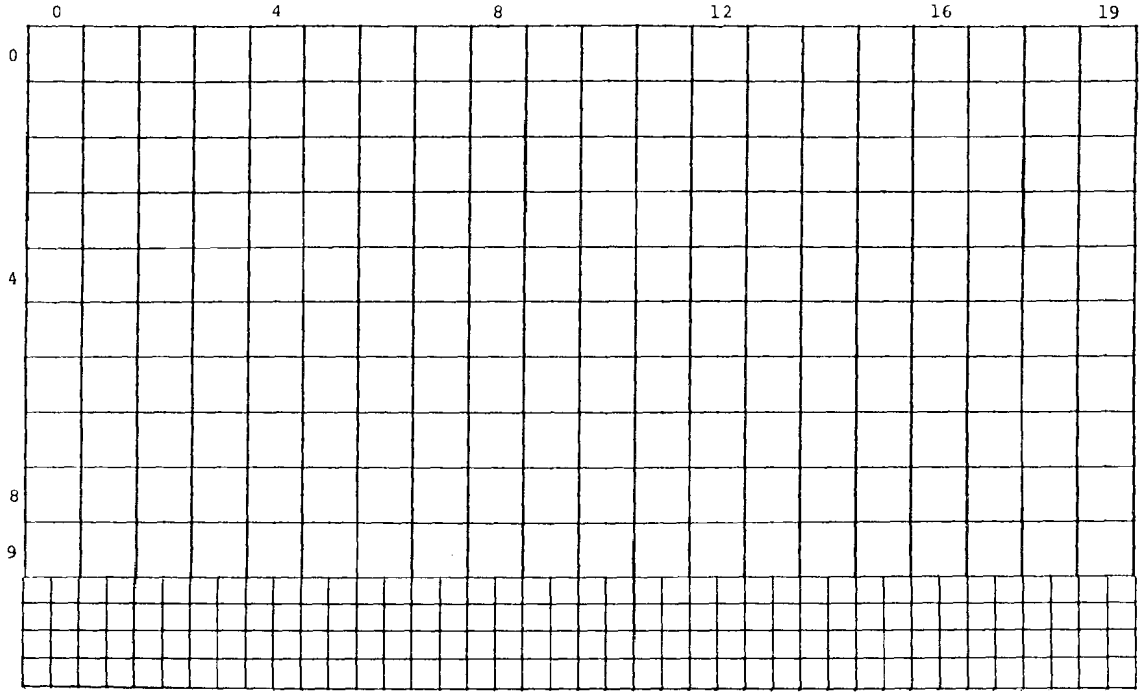

Graphik-Modus 1
mit Textfenster
0
4
8
12
16
19
0
4
8
12
16
19

Graphik-Modus 0

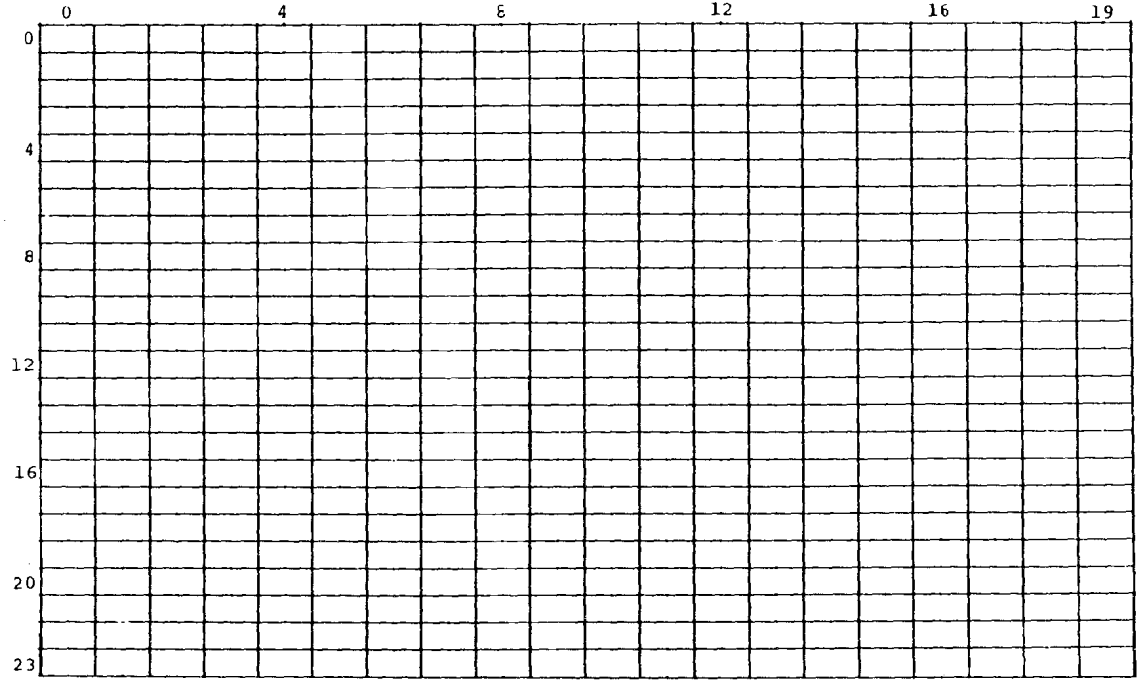


Bemerkungen: _____

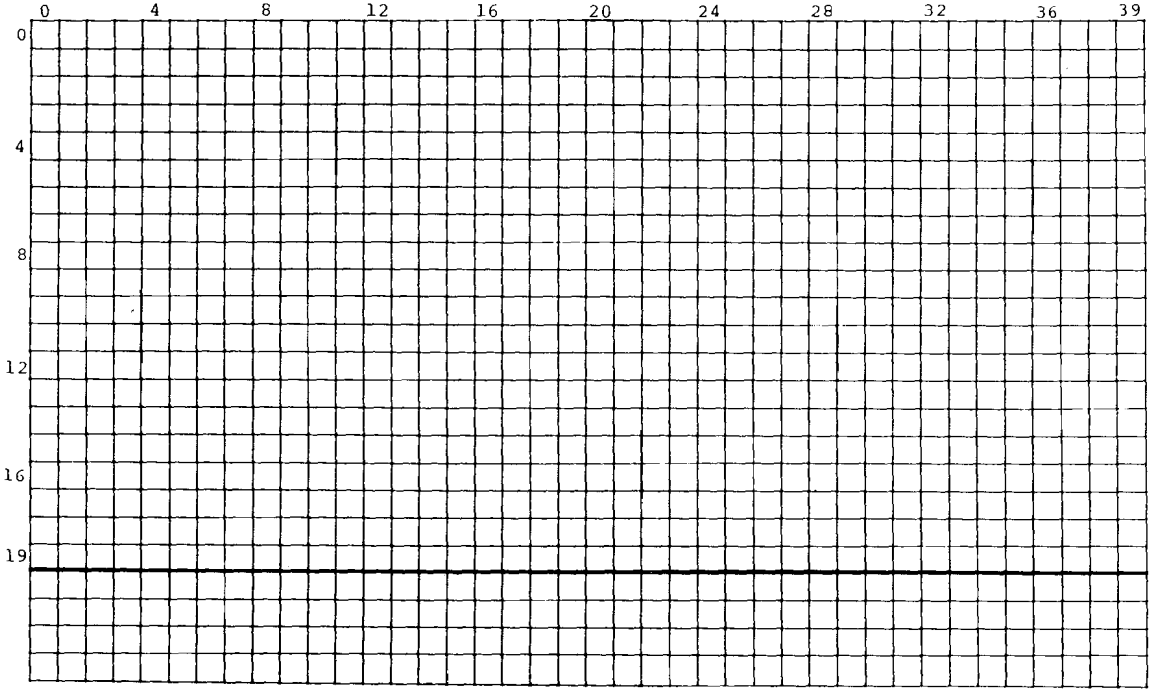
Graphik-Modus 2
mit Textfenster



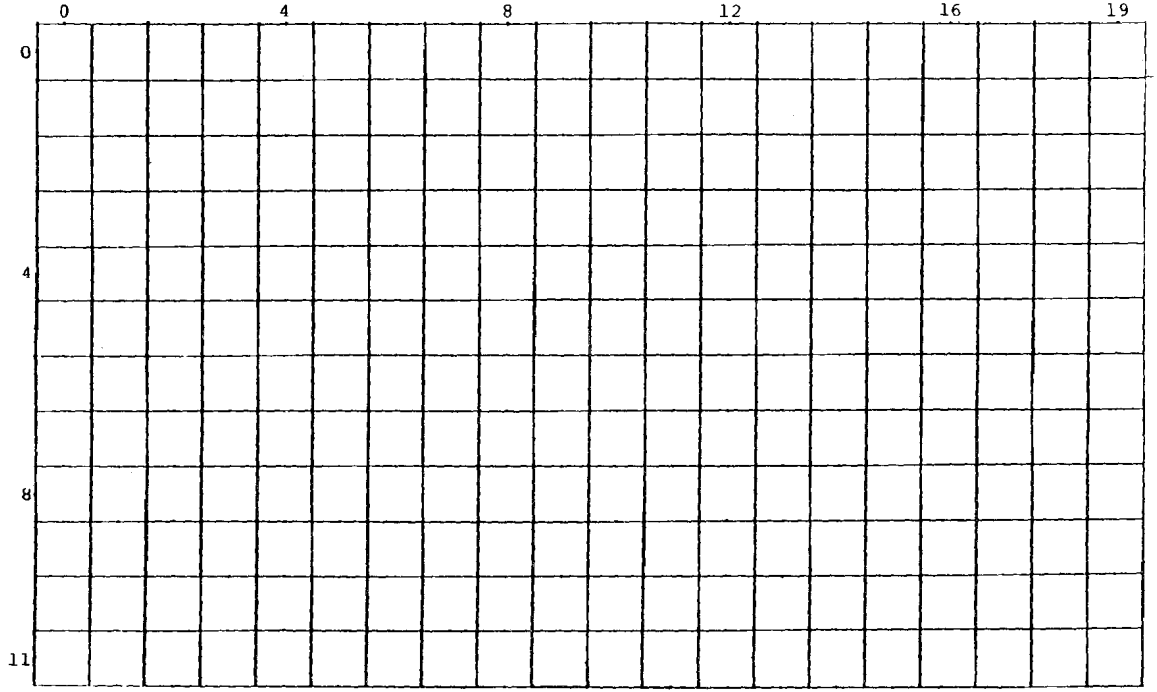
Graphik-Modus 1



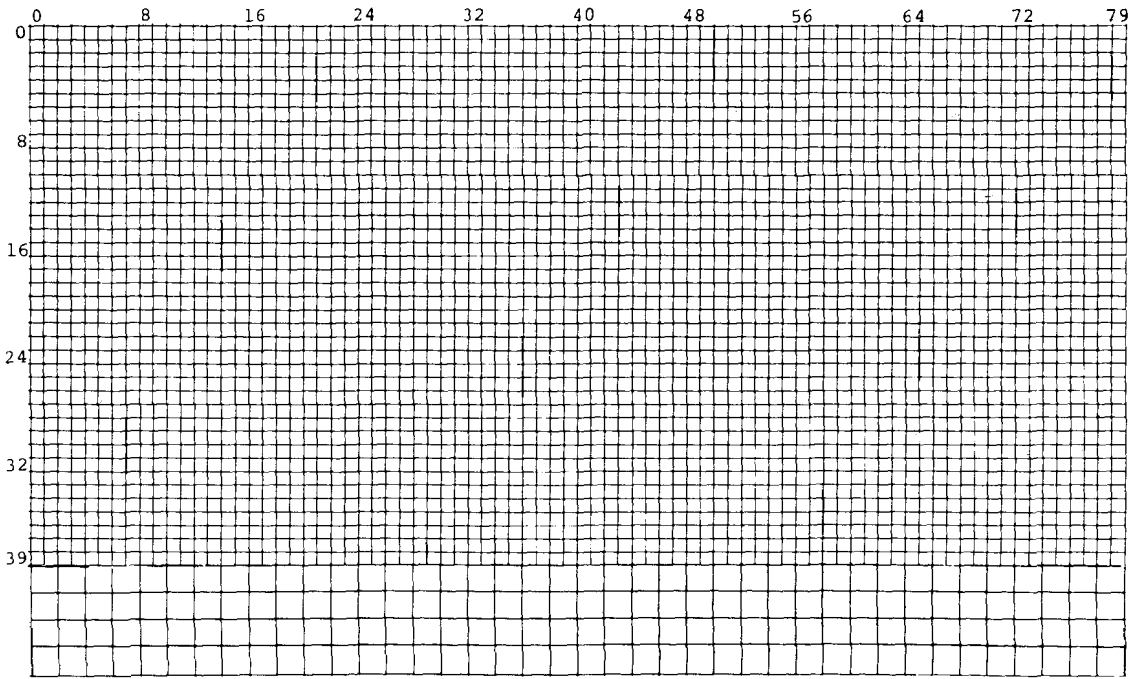
Graphik-Modus 3
mit Textfenster



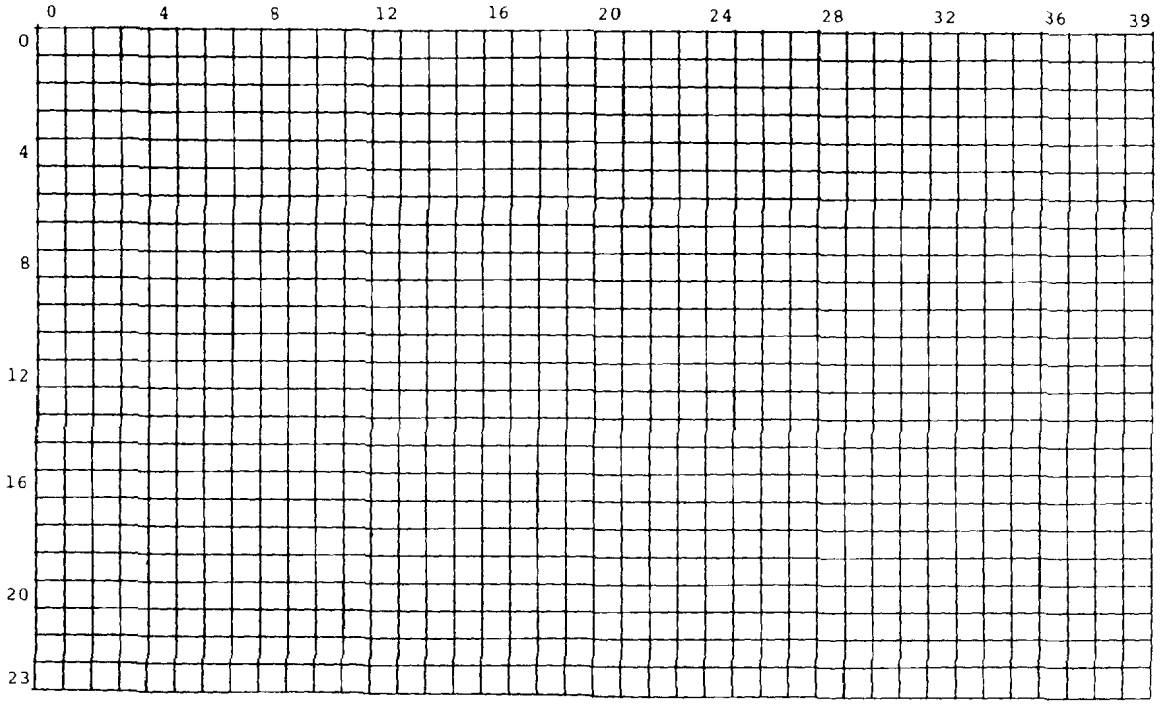
Graphik-Modus 2



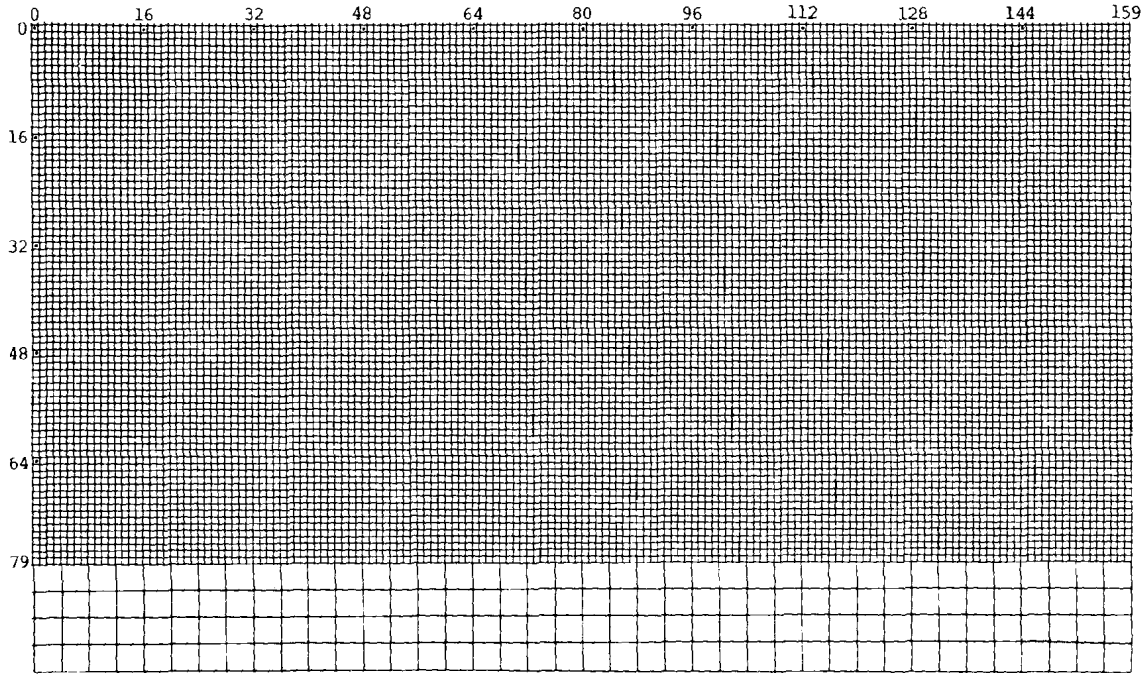
Graphik-Modus 4 oder 5
mit Textfenster



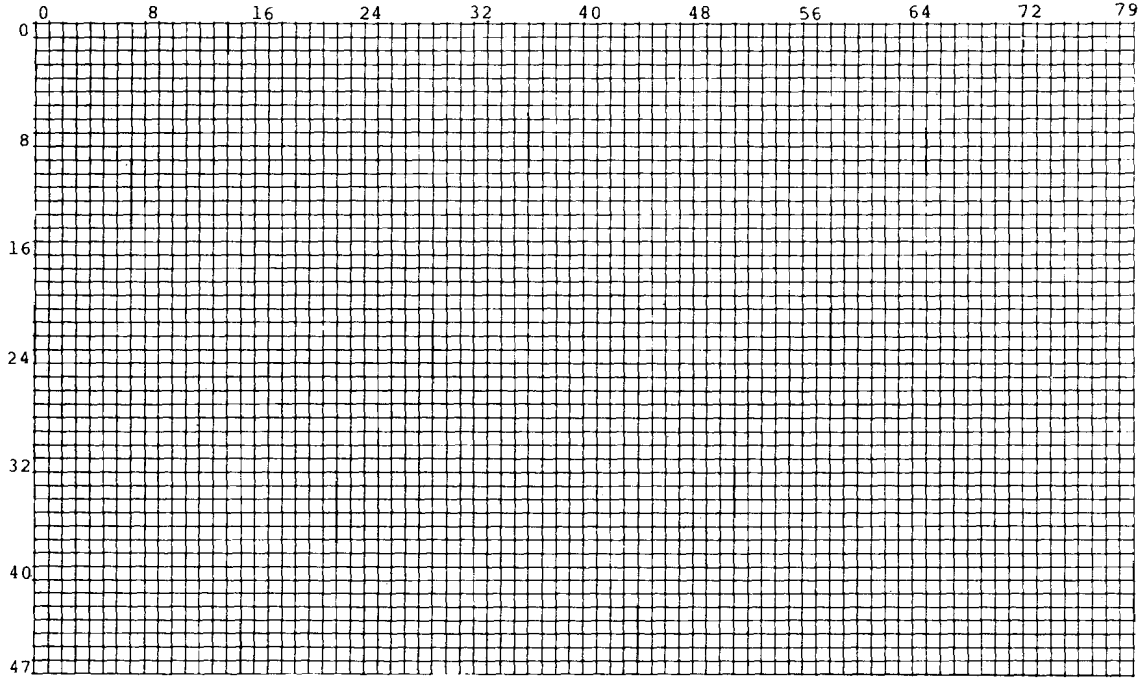
Graphik-Modus 3



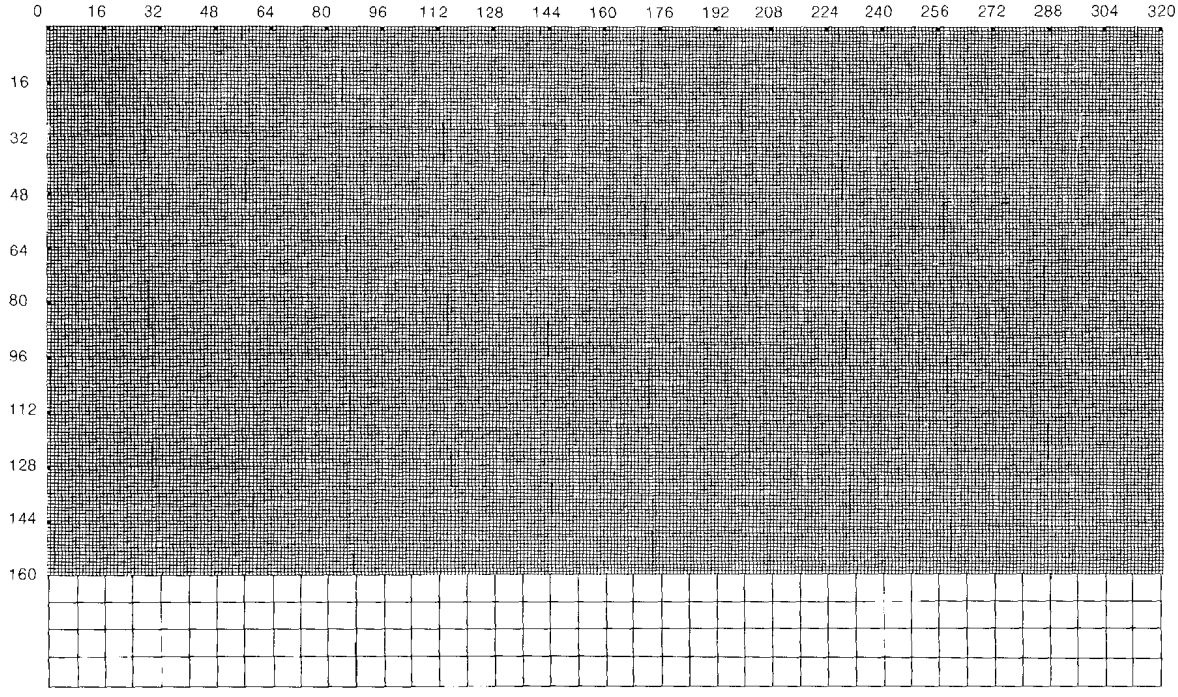
Graphik-Modus 6 oder 7
mit Textfenster



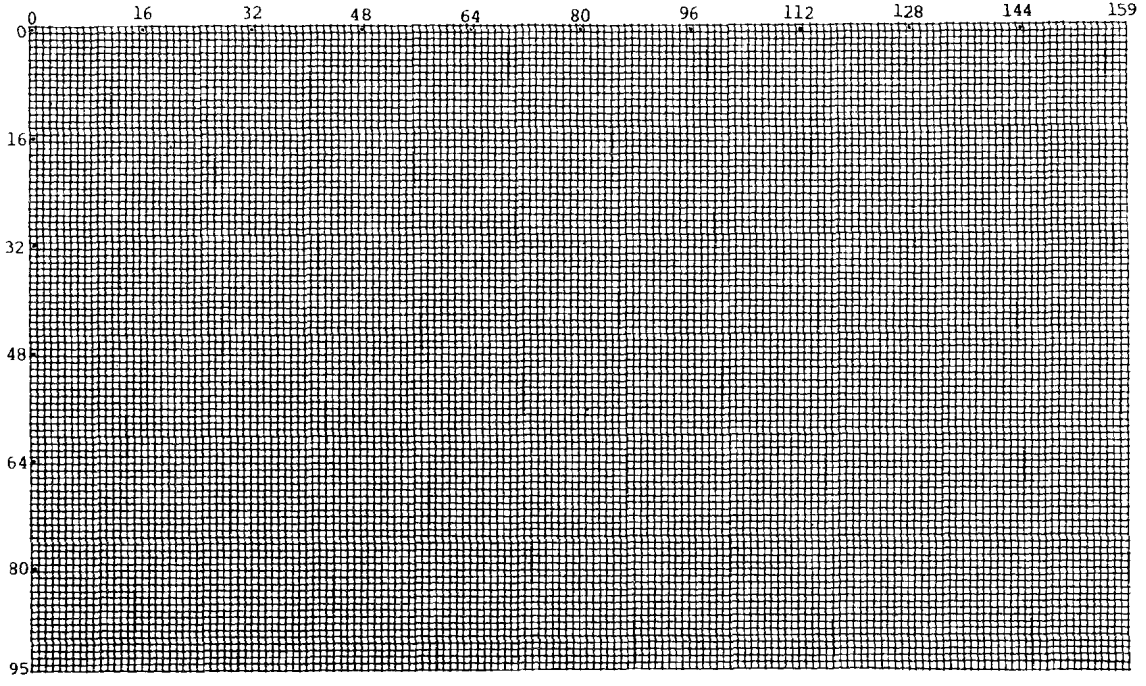
Graphik-Modus 4 oder 5



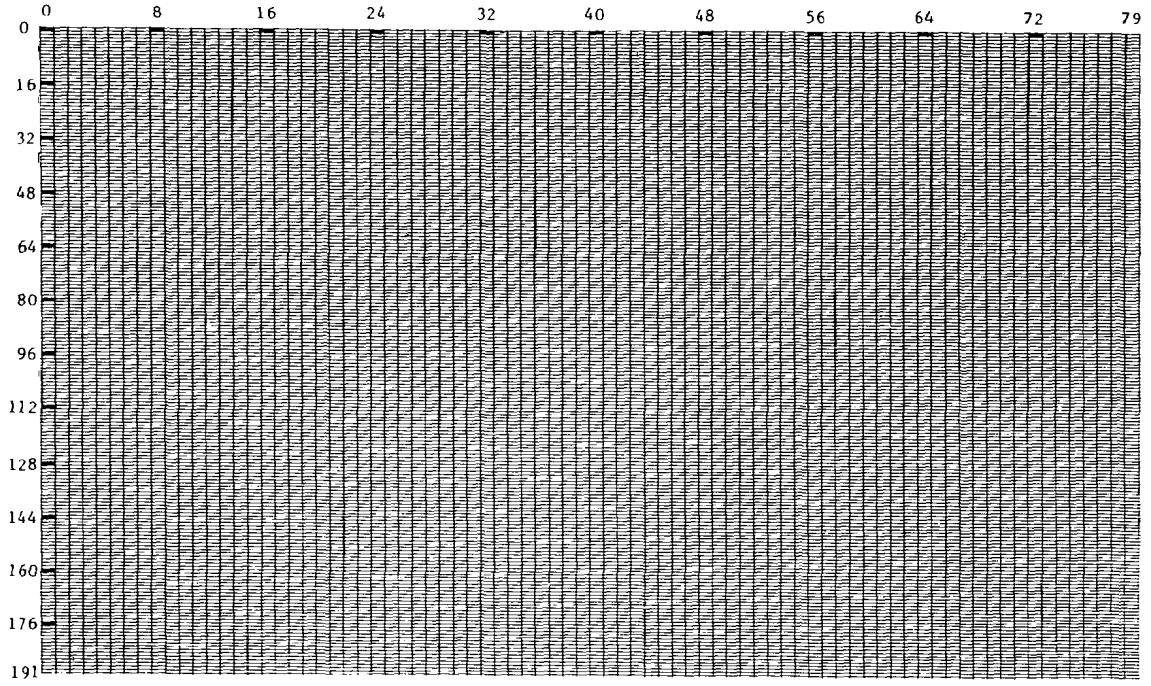
Graphik-Modus 8
mit Textfenster



Graphik-Modus 6 oder 7

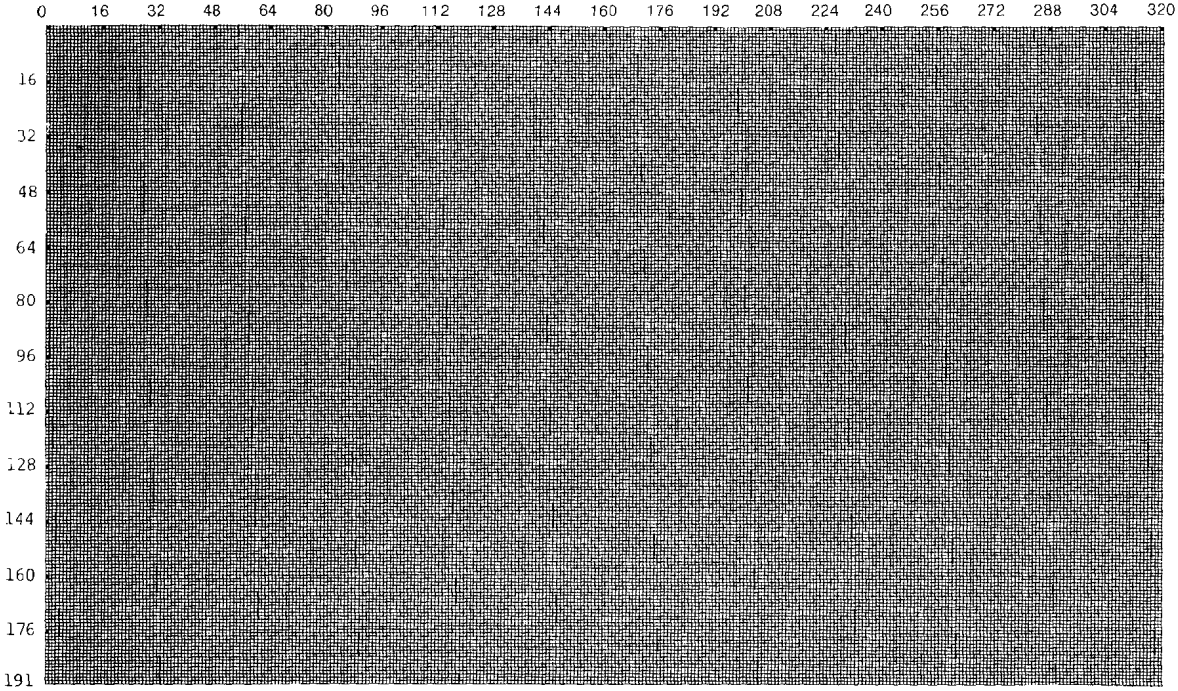


Graphik-Modus 9, 10 oder 11



GTIA

Graphik-Modus 8



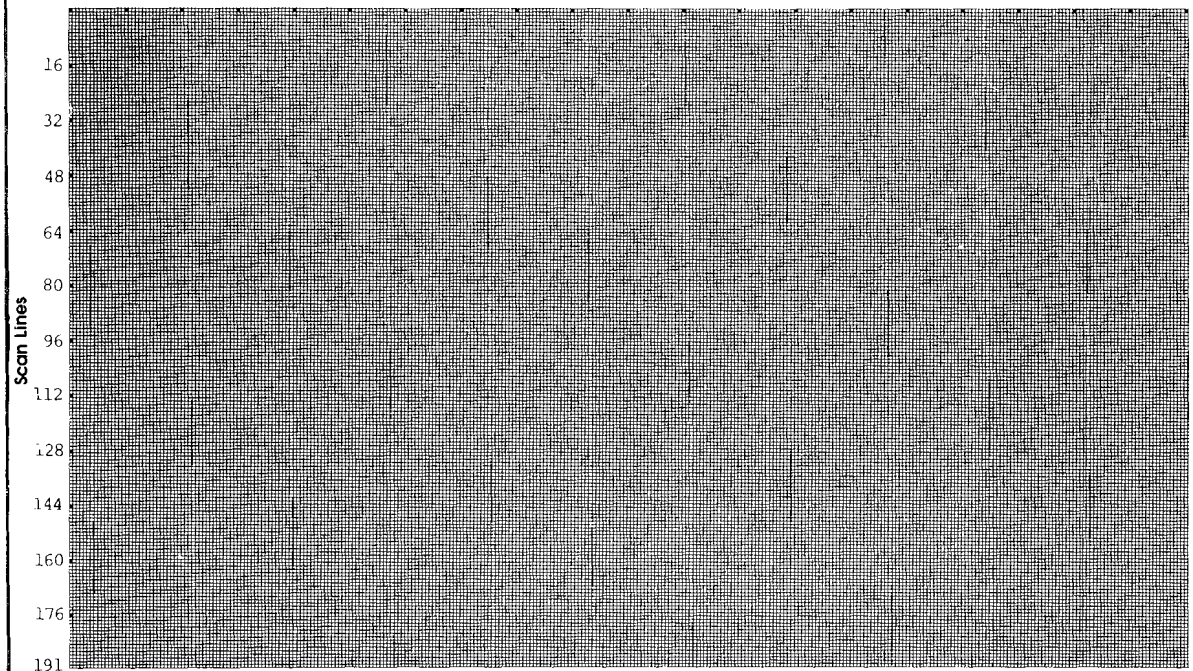
Arbeitsblatt für Zeichenentwürfe

128	64	32	16	8	4	2	1	Dezimalwert

128	64	32	16	8	4	2	1	Dezimalwert

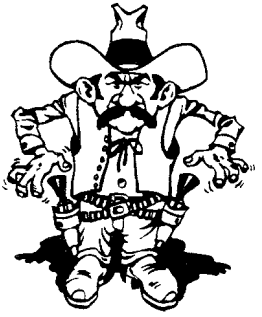
128	64	32	16	8	4	2	1	Dezimalwert

DISPLAY LIST ARBEITSBLATT



ANHANG B

Lösungen zu den Aufgaben



1. SETCOLOR 2,2,6 (Oranger Hintergrund)

   SETCOLOR 4,8,6 (Blauer Rand)

   SETCOLOR 1,2,0 (Dunkle Worte)

2.
```
10 GRAPHICS 3
20 COLOR 1
30 PLOT 10,5
40 DRAWTO 19,5
50 DRAWTO 19,14
60 DRAWTO 10,14
70 DRAWTO 10,5
```

3.
```
10 GRAPHICS 3
20 COLOR 1
30 PLOT 17,7
40 DRAWTO 22,7
50 DRAWTO 22,12
60 DRAWTO 17,12
70 DRAWTO 17,7
```

4.
```
100 REM HAUS IN GRAPHIK-MODUS 7
200 REM SETZE GRAPHIK-MODUS
210 GRAPHICS 7
220 SETCOLOR 0,4,4
230 SETCOLOR 1,8,4
300 REM ZEICHNE DACH DES HAUSES
310 COLOR 1
330 PLOT 79,30
340 DRAWTO 99,40
350 DRAWTO 59,40
360 DRAWTO 79,30
400 REM ZEICHNE FUNDAMENT DES HAUSES
410 COLOR 2
420 PLOT 64,41
430 DRAWTO 94,41
440 DRAWTO 94,61
450 DRAWTO 64,41
460 DRAWTO 64,41
500 END
```

```
   100 REM GEHWEG EIN-PUNKTPERSPEKTIVE
   200 REM SETZE GRAPHIK-MODUS
   210 GRAPHICS 7
   220 COLOR 1
   300 REM LESE DATA UND ZEICHNE LINIEN
   310 READ LINES
   320 FOR L=1 TO LINES
   330 READ P1,P2,D1,D1
   340 PLOT P1,P2:DRAWTO D1,D2
   350 NEXT L
   360 DATA 7
   370 DATA 10,10,39,69,10,10,119,69,39,69,119,69
   380 DATA 28,47,80,47,22,32,50,32,17,23,33,23
   390 DATA 14,16,20,16
   800 END
```

8.
```
   100 REM GEHWEG EIN-PUNKTPERSPEKTIVE
   200 REM SETZE GRAPHIK-MODUS
   210 GRAPHICS 7
   220 COLOR 2
   230 SETCOLOR 0,0,8
   300 REM LESE DATA UND ZEICHNE LINIEN
   320 FOR L=1 TO LINES
   330 READ P1,P2,D1,D2
   340 PLOT P1,P2:DRAWTO D1,D2
   350 NEXT L
   360 DATA 7
   370 DATA 60,20,20,25,20,25,20,45,20,45,60,55
   380 DATA 60,55,110,43,110,43,110,27,110,27,60,20
   390 DATA 60,20,60,55
   400 REM FUELLE RECHTE SEITE MIT FARBE
   410 POKE 765,1
   420 PLOT 60,20
   430 POSITION 60,55
   440 XIO 18,#6,0,0,"S:"
   800 END
```

9.
```
   10 REM VERTIKALE LINIEN IN GRAPHIK-MODUS 0
   20 GRAPHICS 0
   30 FOR X=1 TO 760
   40 PRINT CHR$(22);
   50 NEXT X
```

10.
```
   10 REM AUFGABE #10
   20 GRAPHICS 2+16
   40 POSITION 7,1:PRINT #6;"FARBEN"
   50 POSITION 0,4
   60 PRINT #6;"  BLAU"
   70 PRINT #6;"  GRUEN"
   80 PRINT #6;"  ROT"
   100 GOTO 100
```

5.
```
100 REM DARSTELLUNG VON FARBKONTRAST
200 REM SETZE GRAPHIK-MODUS
210 GRAPHICS 3+16
220 SETCOLOR 0,2,8
230 SETCOLOR 1,12,8
240 SETCOLOR 2,6,8
250 SETCOLOR 4,0,0
300 REM ZEICHNE 3 BLOECKE IN DEN FARBEN 1,2,3
310 FOR C=0 TO 2
320 COLOR C+1
330 REM ZEICHNE 7 HORIZONTALE LINIEN
340 FOR X=1 TO 7
350 PLOT 2,C*7+X
370 NEXT X
380 NEXT C
390 GOTO 390
400 END
```

6. (1)

```
100 REM DARSTELLUNG VON FARBKONTRAST
200 REM SETZE GRAPHIK-MODUS
210 GRAPHICS 3+16
220 SETCOLOR 0,4,4
230 SETCOLOR 1,6,4
240 SETCOLOR 2,4,4
250 SETCOLOR 4,0,0
300 REM ZEICHNE 3 BLOECKE IN DEN FARBEN 1,2,3
310 FOR C=0 TO2
320 COLOR C+1
330 REM ZEICHNE 7 HORIZONTALE LINIEN
340 FOR X=1 TO 7
350 PLOT 2,C*7+X
360 DRAWTO 38,C*7+X
370 NEXT X
380 NEXT C
390 GOTO 390
400 END
```

(2)

```
100 REM DARSTELLUNG VON FARBKONTRAST
200 REM SETZE GRAPHIK-MODUS
210 GRAPHICS 3+16
220 SETCOLOR 0,6,4
230 SETCOLOR 1,6,4
240 SETCOLOR 2,6,4
250 SETCOLOR 4,4,4
300 REM ZEICHNE 3 BLOECKE IN DEN FARBEN 1,2,3
310 FOR C=0 TO 2
320 COLOR C+1
330 REM  ZEICHNE 7 HORIZONTALE LINIEN
340 FOR X=1 TO 7
350 PLOT 2,C*7+X
360 DRAWTO 38,C*7+X
370 NEXT X
380 NEXT C
390 GOTO 390
400 END
```

13.
```
 10 REM AUFGABE #13
 15 GRAPHICS 0
 17 SETCOLOR 2,4,2
 20 DLIST=PEEK(560)+PEEK(561)*256
 30 REM 3*8 LEERZEILEN
 40 POKE DLIST,112
 50 POKE DLIST+1,112
 60 POKE DLIST+2,112
 70 REM SETZE LMS ANWEISUNG
 80 POKE DLIST+3,71
 90 REM BILDSCHIRMSPEICHERADRESSEN
100 POKE DLIST+4,PEEK(88)
110 POKE DLIST+5,PEEK(89)
120 REM MODUS 2 ZEILEN
130 FOR X=6 TO 9
140 POKE DLIST+X,7
150 NEXT X
160 REM MODUS 0 ZEILEN
170 POKE DLIST+10,2
180 REM MODUS 2 ZEILEN
190 POKE DLIST+11,7
200 REM MODUS 0 ZEILEN
210 POKE DLIST+12,2
220 REM MODUS 2 ZEILEN
230 FOR X=13 TO 17
240 POKE DLIST+X,7
250 NEXT X
260 REM SETZE SPRINGANWEISUNG
270 POKE DLIST+18,65
280 REM ANFANG DER DISPLAY LIST
290 POKE DLIST+19,PEEK(560)
300 POKE DLIST+20,PEEK(561)
305 REM SCHREIBE AN BILDSCHIRM
310 SCREENMEM=PEEK(88)+PEEK(89)*256
320 HI=INT((SCREENMEM+140)/256)
330 LO=(SCREENMEM+140)-(HI*256)
340 POKE 88,LO:POKE 89,HI
370 POKE 87,2
380 POSITION 0,0
390 PRINT #6;"   VERKNUEPFTE MODI"
500 GOTO 500
```

11.
```
7000 REM EINE WOLKE
7010 FOR T=1 TO 100
7015 COLOR 2
7020 PLOT RND(0)*15+25,RND(0)*10+15
7030 PLOT RND(0)*25+20,RND(0)*20+10
7050 NEXT T
```

12.
```
10 REM AUFGABE #12
15 GRAPHICS 5+16
20 DLIST=PEEK(560)+PEEK(561)*256
30 REM 3*8 LEERZEILEN
40 POKE DLIST,112
50 POKE DLIST+1,112
60 POKE DLIST+2,112
70 REM SETZE LMS-ANWEISUNG
80 POKE DLIST+3,70
90 REM BILDSCHIRMSPEICHERADRESSE
100 POKE DLIST+4,PEEK(88)
110 POKE DLIST+5,PEEK(89)
120 REM EINE ANDERE MODUS 1 ZEILE
130 POKE DLIST+6,6
140 REM MODUS 5 ZEILEN
150 FOR X=7 TO 44:POKE DLIST+X,10:NEXT X
160 REM MODUS 2 ZEILEN
170 FOR X=45 TO 46:POKE DLIST+X,7:NEXT X
180 REM SETZE SPRINGANWEISUNG
190 POKE DLIST+47,65
200 REM ANFANG DER DISPLAY LIST
210 POKE DLIST+48,PEEK(560)
220 POKE DLIST+49,PEEK(561)
300 REM SCHREIBE AUF BILDSCHIRM
310 SCREENMEM=PEEK(88)+PEEK(89)*256
320 POKE 87,1
325 POSITION 0,0
330 PRINT #6;"   EIN FARBIGER BLOCK"
335 REM ZEICHNE FARBIGEN BLOCK
340 HI=INT((SCREENMEM+40)/256)
350 LO=(SCREENMEM+40)-(HI*256)
360 POKE 87,5
370 POKE 88,LO:POKE 89,HI
390 FOR Y=9 TO 26
395 COLOR 1
400 PLOT 20,Y:DRAWTO 60,Y
410 NEXT Y
420 REM DRUCKE IM MODUS 2
430 HI=INT((SCREENMEM+800)/256)
440 LO=(SCREENMEM+800)-(HI*256)
450 POKE 87,2
460 POKE 88,LO:POKE 89,HI
465 POSITION 0,0
470 PRINT #6;"   IM MODUS 3"
500 GOTO 500
```

```
610 POKE 53248,105
620 POKE 53249,105
630 POKE 53258,105
650 REM SETZE PLAYER GROESSE
660 POKE 53256,3
670 POKE 53257,3
680 POKE 53258,3
700 REM SETZE VERTIKALE AUFLOESUNG
710 POKE 559,46
800 REM AKTIVIERE DIE PLAYER
820 POKE 54279,PMBASE
820 POKE 53277,3
850 POSITION 17,4:PRINT "ROT"
860 POSITION 15,9:PRINT "GELB"
870 POSITION 16,14:PRINT "BLAU"
900 GOTO 900
```

16.
```
10 REM BEWEGE ZEICHEN IM GRAPHIK-MODUS 2
20 GRAPHICS 2
30 FOR P=0 TO 18
40 POSITION P+1,0:PRINT #6;CHR$(48)
50 POSITION P,0:PRINT #6;CHR$(32)
60 FOR WAIT=1 TO 100:NEXT WAIT
70 NEXT P
```

17.
```
100 REM PLAYER/MISSILE ANIMATION
200 REM BILDE PLAYER UND SETZE REGISTER
210 PMBASE=PEEK(106)-4
220 POKE 106,PMBASE-1
230 GRAPHICS 3+16
240 FOR X=512 TO 768
250 POKE PMBASE*256+X,0
260 NEXT X
265 FOR PLAYER=0 TO 1
270 FOR X=20 TO 30
280 POKE PMBASE*256+PLAYER*128+PLAYER*30+512+X,255
290 NEXT X
295 NEXT PLAYER
300 POKE 704,30
305 POKE 705,66
310 POKE 559,46
320 POKE 54279,PMBASE
330 POKE 53277,3
400 REM BEWEGE PLAYER
410 J=40:K=40
415 I=1:H=2
420 POKE 53248,J
430 POKE 53249,K
440 J=J+I:K=K+H
450 IF J>208 THEN I=-1
460 IF K>208 THEN H=-1
470 IF J<40 THEN I=-1
480 IF K<40 THEN H=-1
490 GOTO 420
```

14.
```
100 REM PLAYER 0,1,2 UND 3
200 REM SETZE PLAYER/MISSILE BASISADRESSE
210 PMBASE=PEEK(106)-4
220 POKE 106,PMBASE-1
230 GRAPHICS 1+16
300 REM KLAERE PLAYER/MISSILE RAM
310 FOR X=512 TO 1023
320 POKE PMBASE*256+X,0
330 NEXT X
400 REM GEBE DATA IN PLAYERS
410 FOR PLAYERS=0 TO 3
420 FOR LINES=44 TO 84
430 POKE PMBASE*256+PLAYER*128+512+LINES,255
440 NEXT LINES
450 NEXT PLAYERS
500 REM SETZE FARBEN
510 POKE 704,30
520 POKE 705,66
530 POKE 706,66
540 POKE 707,30
600 REM SETZE HORIZONTALE POSITION
610 POKE 53249,60
620 POKE 53249,70
630 POKE 53250,170
640 POKE 53251,180
700 REM SETZE VERTIKALE AUFLOESUNG
710 POKE 559,46
800 REM AKTIVIERE DIE PLAYERS
810 POKE 54279,PMBASE
820 POKE 53277,3
850 POSITION 6,8:PRINT #6;"PLAYERS"
860 POSITION 8,11:PRINT #6;"UND"
870 POSITION 6,14:PRINT #6;"MISSILES"
900 GOTO 900
```

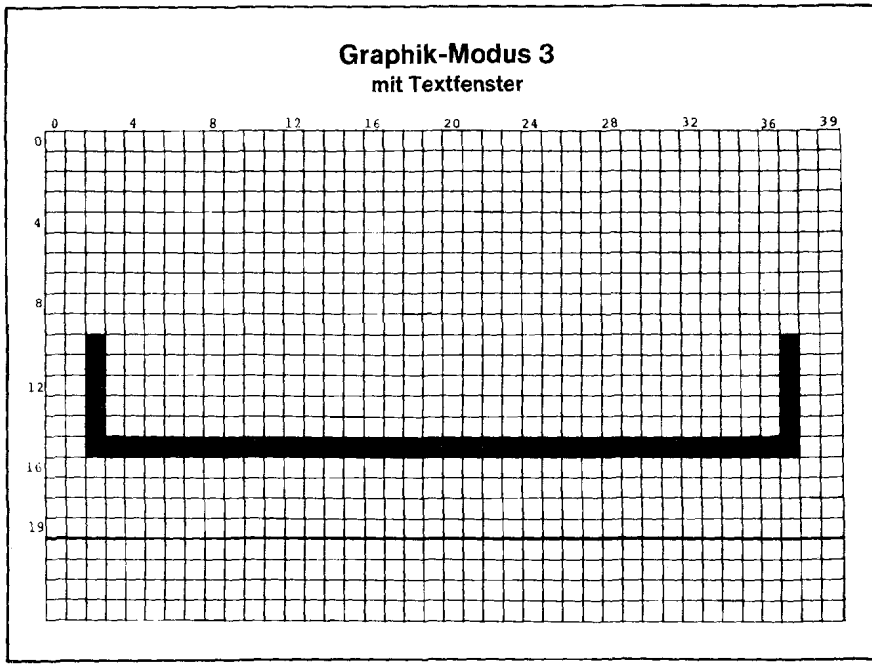
15.
```
100 REM PLAYER 0,1 UND 2
200 REM SETZE PLAYER/MISSILE BASISADRESSE
210 PMBASE=PEEK(106)-4
220 POKE 106,PMBASE-1
230 GRAPHICS 0
300 REM KLAERE PLAYER/MISSILE RAM
310 FOR X=512 TO 896
320 POKE PMBASE*256+X,0
330 NEXT X
400 REM GEBE DATA IN PLAYER
410 FOR PLAYER=0 TO 2
420 FOR LINES=30 TO 40
430 POKE PMBASE*256+PLAYER*128+PLAYER*20+512+LINES,255
440 NEXT LINES
450 NEXT PLAYER
500 REM SETZE FARBEN
510 POKE 704,66
520 POKE 705,30
530 POKE 706,158
600 REM SETZE HORIZONTALE POSITION
```

ANHANG C

Zusammenfassung der Antworten

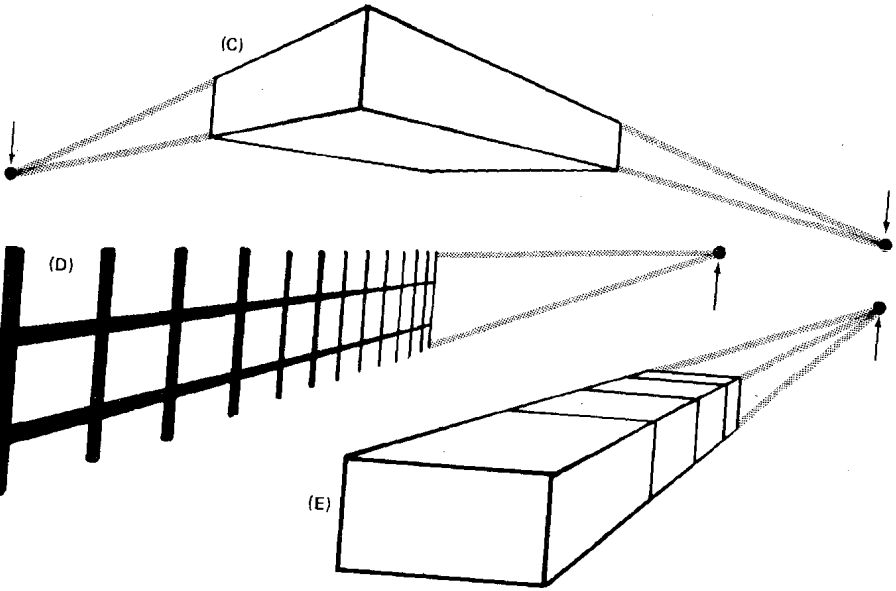
Kapitel 1

1. GRAPHICS
2. SETCOLOR
3. PLOT
4. COLOR
5. DRAWTO
6. POSITION
7. PRINT
8. Rosa
9. Grün
10. Orange
11.

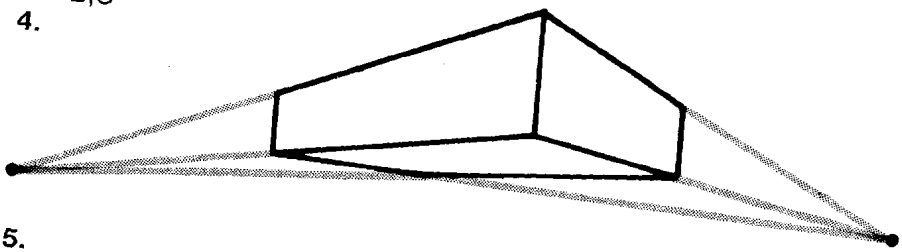


18.
```
100 REM AENDERE ATARI ZEICHENSATZ
200 REM NEHME ZEICHENSATZ RAM
210 CHARSET=PEEK(106)-4
220 POKE 106,CHARSET-1
230 GRAPHICS 0
300 REM KOPIERE ATARI ZEICHENSATZ
310 FOR X=0 TO 1023
320 POKE CHARSET*256+X,PEEK(224*256+X)
330 NEXT X
400 REM AENDERE DEN BUCHSTABEN A
410 FOR X=0 TO 7
420 READ N:POKE CHARSET*256+264+X,N
430 NEXT X
480 DATA 0,240,240,216,204,254,198,0
500 REM SETZE ZEICHENSATZ
510 POKE 756,CHARSET
```

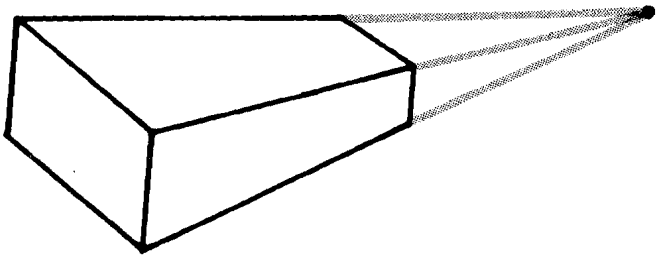
19.
```
100 REM ZEICHEN IM GRAPHIK-MODUS 6
200 REM SETZE GRAPHIK-MODUS
210 GRAPHICS 6+16
220 DIM CHAR$(10)
300 REM FINDE BILDSCHIRM RAM POSITION
310 SCREENRAM=PEEK(88)+PEEK(89)*256
400 REM SETZE ZEICHENREIHE
405 PX=9:PY=35
410 CHAR$="QUADRAT":GOSUB 500
460 REM ZEICHNE QUADRAT
460 COLOR 1
470 PLOT 40,20:DRAWTO 120,20:DRAWTO 120,80:DRAWTO 40,80
   :DRAWTO 40,20
490 GOTO 490
500 REM POKE ZEICHEN AUF DEN BILDSCHIRM
510 FOR U=1 TO LEN(CHAR$)
520 I=224*256+((ASC(CHAR$(U,U))-32)*8)
530 J=SCREENRAM+PY*20+PX+U-1
540 FOR Z=0 TO 7:POKE J+Z*20,PEEK(I+Z):NEXT Z
550 NEXT U
560 RETURN
```



2. A,D,E
3. B,C
4.



5.



Kapitel 2

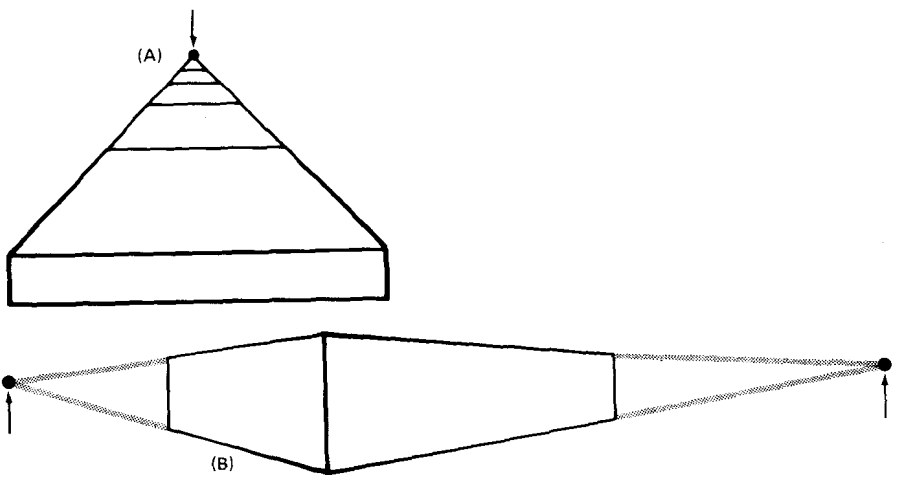
1. B
2. A
3. E
4. D
5. C
6. 4
7. 4
8. 0
9. 4
10. 2
11. 2
12. 4
13. Orange
14. Blau
15. Rosa

Kapitel 3

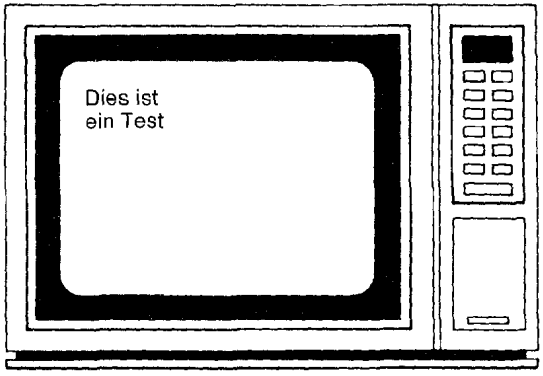
1. Rot-blau
2. Gelb und orange
3. Blau
4. Warm
5. Modus 9
6. 22
7. A) Blau
   B) Eine Vase
8. A) Warm
   B) Orange
9. Grün und blau
10. Blau und rot
11. Orange

Kapitel 4

1.



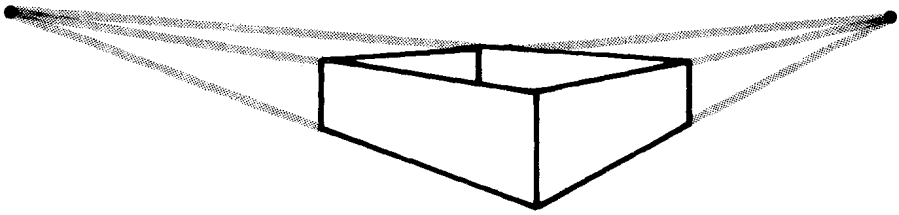
9.



Kapitel 7

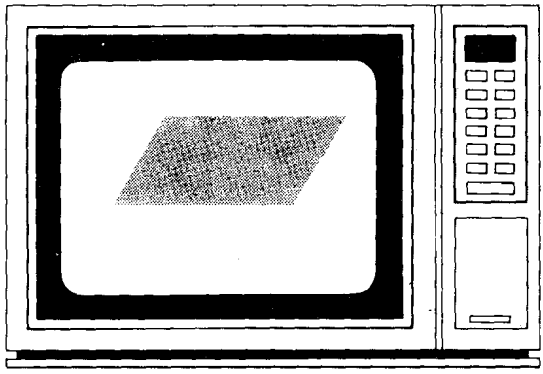
1. 32 Scanzeilen
2. 44 GRAPHIK-Modus 7 Zeilen
3. Vier GRAPHIK-Modus 2 Zeilen, sechszehn GRAPHIK-Modus 5 Zeilen, zweiunddreißig GRAPHIK-Modus 7 Zeilen
4. 87 (dezimal)
5. High Byte = 55 (dezimal)
   Low Byte = 216 (dezimal)
6. 20 Bytes
7. 560 enthält den Low Byte
   561 enthält den High Byte
8. 112 7
   112 7
   112 7
   66 7
   PEEK(88) 65
   PEEK(89) PEEK(560)
   2 PEEK(561)
   7
   7
   7
   7
   7
   7
   7

6.



7. a) 11,16 b) 19,11 c) 25,7 d) 28,5
e) 33,2 f) 32,5 g) 31,7 h) 30,11 i) 28,16

8.



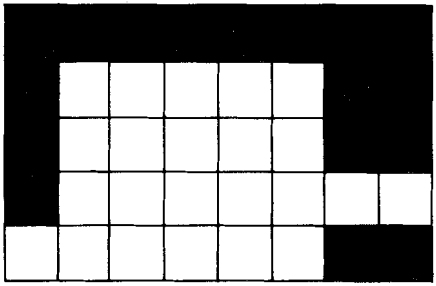
Kapitel 5

1. B
2. D
3. A
4. C
5. H
F
6. a) Dunkelblau
b) 2
c) Hellgrün
d) 1
e) Orange
f) 0
g) Rot
h) 3
i) Hellgrün
j) 1
7. POKE 83,22
8. POKE 752,1

Kapitel 8

1. 120
224
224
255
7
15

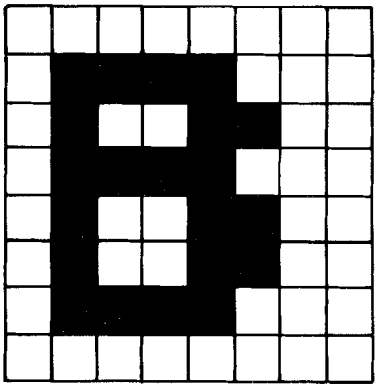
2.



3. POKE 706,132
4. POKE 53250,114
5. POKE 53258,1
6. Acht Seiten
7. Bit 2

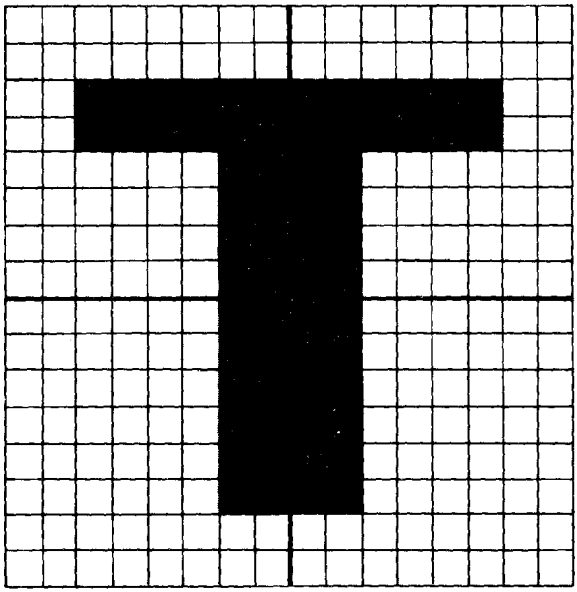
Kapitel 10

1.

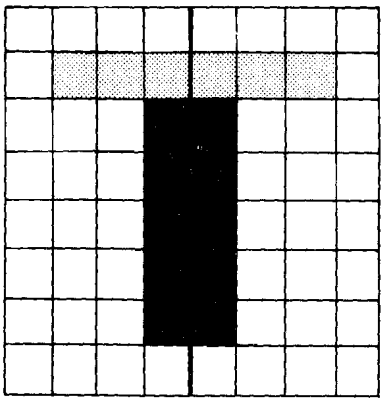


128	64	32	16	8	4	2	1	Dezimal-wert
0	0	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0	120
0	1	0	0	1	1	0	0	76
0	1	1	1	1	0	0	0	120
0	1	0	0	1	1	0	0	76
0	1	0	0	1	1	0	0	76
0	1	1	1	1	0	0	0	120
0	0	0	0	0	0	0	0	0

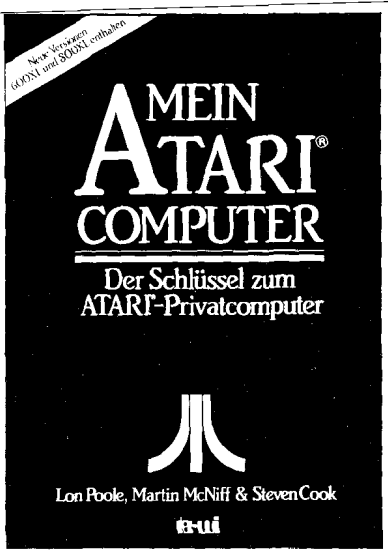
2.



3. 8 Bytes
4. 1024 Bytes
5. 4 – 20
   S – 51
   w – 119
6. 56864
7. SCREENRAM + 20
8.



linkes Byte	rechtes Byte
0	0
63	252
3	192
3	192
3	192
3	192
3	192
0	0



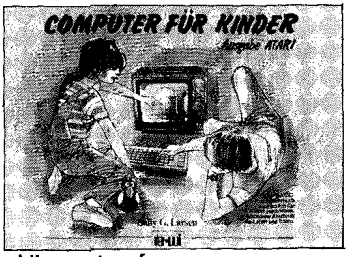
MEIN ATARI™-COMPUTER

(L. Poole, M.McNiff, S. Cook)

Dieses Buch macht die Möglichkeiten, die in Ihrem ATARI-Computer stecken, auf leicht verständliche Art transparent. In einfachen Schritten wird der Anwender mit der Bedienung der Geräte und mit der Software vertraut gemacht. Es enthält zahlreiche Tips zur Aufdeckung und Beseitigung von möglichen Fehlerquellen bei Hard- und Software. Scheinbar unlösbare Probleme werden beseitigt.
Dem fortgeschrittenen Programmierer bieten die vielen Übersichtstafeln, sowie die alphabetische Auflistung der BASIC-Befehle und Funktionen eine entscheidende Hilfestellung.

Ladenpreis: DM 59,–

COMPUTER FÜR KINDER, AUSGABE ATARI



Dieses Buch richtet sich an Kinder im Alter von 8 bis 13 Jahren, für deren Interesse an Computern keines der unzähligen Computer-Bücher geschrieben wurde.
„Computer für Kinder" ist ganz auf Kinder eingestellt und beschäftigt sich unterhaltsam und leicht verständlich mit folgenden Themen:

Wie arbeiten Computer
Wie funktioniert mein Computer
Wie programmiert man mit einfachen Flußdiagrammen
Wie kann ich BASIC leicht verstehen
Programme aufbauen mit Befehlen
Farbige Graphiken entwerfen
Erklärung von Computer-Begriffen

Sally Greenwood Larsen war Kindergärtnerin, ehe sie selbst Computern begegnete und zwischen den Welten von Kindern und Computern zu vermitteln begann.
Computer für Kinder, A4 quer, Fadenheftung, über 100 Seiten, DM 29,80

NEU!
LOGO Computersprache für Kinder und Eltern

Dieses Buch beweist: **Jeder kann programmieren.** LOGO ist die Computersprache für Eltern und Kinder. Nicht umsonst wurde dieser Titel zum **„Buch des Jahres 1983"** in den USA. LOGO ist das Ergebnis der Erforschung menschlicher Intelligenz; entwickelt von einem Pädagogen und Mathematikprofessor. LOGO ist die erste Computersprache, die bewußt Strategien menschlichen Denkens dient.

Daniel Watt, ca. 400 Seiten, Softcover, A4, DM 59,–

---

te-wi Verlag GmbH
Telefon 089/1292090

te-wi

Theo-Prosel-Weg 1
8000 München 40

MIKROCOMPUTER-GRUNDWISSEN

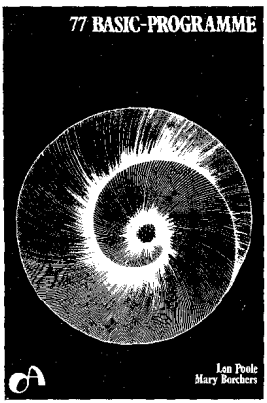
Eine allgemeinverständliche Einführung in die Mikrocomputer-Technik – vom Mikrocomputer-Papst Dr. Adam Osborne. Optimal als Einstieg für Elektronik-Laien; zur Kontaktaufnahme mit diesem die Technik und unsere Umwelt revolutionierenden Gebiet.

Von Adam Osborne
304 Seiten, DM 36,–

EINFÜHRUNG IN DIE MIKROCOMPUTER-TECHNIK

Dieses schon legendäre Standardwerk über die Mikrocomputertechnik von Erfolgsautor Osborne ist neu aufgelegt und völlig neu überarbeitet worden. Jetzt spiegelt sich darin der allerletzte Stand dieser faszinierenden Technologie wieder. In bewährter Manier ist es überaus reichhaltig bebildert. Bereits an mehr als 500 Universitäten weltweit ist es als reguläre Studiengrundlage eingeführt.

Von Adam Osborne
488 Seiten, DM 66,–



77 BASIC-PROGRAMME

Eine Sammlung von 77 praktischen Kurzprogrammen in BASIC, die mathematische, finanztechnische, statistische und verschiedene allgemeine Aufgaben behandeln. Die ausführlich erläuterten Befehle lassen sich leicht direkt anwenden oder sie dienen als Übungen.

Von Lon Poole und Mary Borchers
208 Seiten, DM 39,–

PROGRAMMIEREN IN ASSEMBLER

In diesem Buch wird die Programmierung des Mikroprozessors 6502 in Assemblersprache beschrieben. Der Titel enthält eine große Auswahl von praktischen Programmen in Standardformat einschließlich Flußdiagrammen, Quellprogrammen, Objektcodes und erläuternden Texten. Jeder Befehl der CPU wird detailliert erklärt.

Von Lance A. Leventhal
600 Seiten, DM 59,–

---

te-wi Verlag GmbH
Telefon 089/1292090

te-wi

Theo-Prosel-Weg 1
8000 München 40

LOGO

Jeder kann programmieren
Computersprache für Eltern und Kinder
DANIEL WATT

LOGO...Ergebnis der Erforschung menschlicher Intelligenz

Entwickelt von Seymour Papert, Pädagoge und Mathematikprofessor.
Erste Computersprache, die bewußt Strategien menschlichen Denkens dient – und in ihrer Logik der Realität gerecht wird. LOGO ersetzt BASIC, sagen Pädagogen und Mathematiker. LOGO kommt dem übergreifenden, assoziativen Denken entgegen. BASIC dagegen ist ein Setzkasten von Logik-Buchstaben.

DANIEL WATT... hat im Team von Seymour Papert gearbeitet und ein Buch geschrieben, das voller Bilder seine Erlebnisse mit Kindern am Computer wiedergibt. Ein hochwertiges Textbuch für LOGO-Kurse. Ein Buch für Eltern die mit ihren Kindern nicht "Computer", sondern "Lust am eigenen Denken" erleben wollen.

ca. 400 Seiten, DM 59,–

"Buch des Jahres 1983"
in den USA

te-wi
te-wi Verlag GmbH
Theo-Prosel-Weg 1
8000 München 40