Hardcopy für Ihren Drucker

... oder: von den Problemen, den Bildschirminhalt auf Papier zu bringen — und natürlich von den Lösungen dieser Probleme!

Das Problem dürfte sich jedem Benutzer des C 64 stellen, der einen Drucker hat: Auf dem Monitor oder Fernseher wird ein Bild dargestellt, doch leider ist es ganz unmöglich, dieses Bild auf dem Drucker auszugeben, denn es wird nicht der normale Bildschirm mit einer Auflösung von 25 Zeilen á 40 Zeichen dargestellt, sondern ein Bild in hochauflösender Grafik. Wie Sie trotzdem auch mit diesem Problem fertig werden, soll im folgenden beschrieben werden.

Aller Laster Anfang: Ihr Drucker

Schon beim Drucker stoßen wir auf schwerwiegende Probleme, denn manche Drucker sind wirklich nur sehr schwer oder gar nicht zum Drucken von Grafik zu bewegen. Folgende Kriterien sollten bei Ihrem Drucker erfüllt sein:

— der Drucker ist grafikfähig (natürlich die wichtigste Voraussetzung);
— der Drucker druckt mit mindestens acht Nadeln oder ähnlichem. Damit ist gemeint, daß der Druckkopf so konstruiert ist, daß er vertikal acht Punkte untereinander drucken kann;
— der Drucker beherrscht den sogenannten BI-Mode (Bit-Image-Mode). Das ist der Modus, in dem die hochauflösende Grafik ausgedruckt wird.

Wenn Ihr Drucker diese Voraussetzungen erfüllt, dann ist es Ihnen in jedem Fall möglich, Ihre Bildschirmgrafiken auf Papier zu bringen.

Aber auch Drucker, die keinen BI-Modus haben, sind mitunter grafikfähig. Da die Druckerpalette jedoch sehr groß ist, werden wir auf keinen speziellen Drucker eingehen, und es bleibt Ihnen letztlich überlassen, die Tauglichkeit Ihres Druckers zu überprüfen.

Auch bei Druckern der unteren Preisklasse findet man heute schon die sogenannten ESC-Sequenzen. Das sind Steuerbefehle für den Drucker, die ihn zum Beispiel dazu veranlassen, die Schriftart oder etwa den Zeilenvorschub zu verändern. Eine solche Steuersequenz erkennt der Drucker an einem speziellen Zeichencode, bei dem er nicht wie sonst üblich das empfangene Zeichen als solches ausgibt, sondern vielmehr auf weitere Zeichen wartet, die in ihrer Gesamtheit einen Steuerbefehl ergeben. Üblicherweise wird dabei als ESC-Code das Zeichen CHR$(27) an den Drucker gesendet.

ESC — (von escape, engl. für entkommen, flüchten)

Die ESC-Sequenzen sind leider in keiner Norm vereinheitlicht. Deswegen darf normalerweise die Kompatibilität zwischen Druckern gleicher Leistungsklasse nicht ohne Einschränkungen angenommen werden.

Nach Senden von ESC an den Drucker werden immer Daten nachgeschoben, die die Grundeinstellung des Druckers ändern, so etwa den Drucker dazu veranlassen, Grafik anstatt Klartext zu drucken. Wenn Sie einmal Ihr Druckerhandbuch aufschlagen und eine der Sequenzen ESC-K, ESC-L oder ESC-Z finden, dann werden Sie (hoffentlich) feststellen, daß Sie in dem Teil des Handbuchs lesen, der Sie interessiert — der Ausgabe von Grafik.

Jetzt sind noch ein paar Berechnungen anzustellen und schon kann das Bild auf dem Drucker ausgegeben werden. Doch diese Berechnungen haben es in sich!

Wie die Grafik im Computer steht

Der C 64 ist bekanntlich ein vielseitiger Computer. So hat er unter anderem in seinem Inneren einen kleinen Chip (6567 Video Interface Chip, kurz VIC II) eingebaut, der dafür verantwortlich ist, daß sich etwas auf Ihrem Bildschirm tut. Der VIC II ist sehr flexibel und hat vielseitige Verwendungsmöglichkeiten innerhalb des C 64; wir beschränken uns jedoch hier auf die Funktionen, die ausschließlich für die Erzeugung der Grafik verantwortlich sind.

Es können acht Grafikbildschirme vom VIC II verwaltet werden; ihre Basisadressen sind ein Vielfaches von 8192. So beginnt etwa der dritte Grafikbildschirm bei der Basisadresse 16384 (bei 0*8192 liegt der erste Grafikschirm, dieser wird aber im Regelfall nicht benutzt, da der Bereich anderweitig belegt ist, zum Beispiel durch Zero-Page und Stack.

Wenn Sie ein Bild drucken wollen, müssen Sie sich immer erst Klarheit darüber verschaffen, in welchem Bereich der Grafikbildschirm steht, den Sie ausgeben wollen. Zwar können Sie für die Berechnungen auch so lange zwischen den

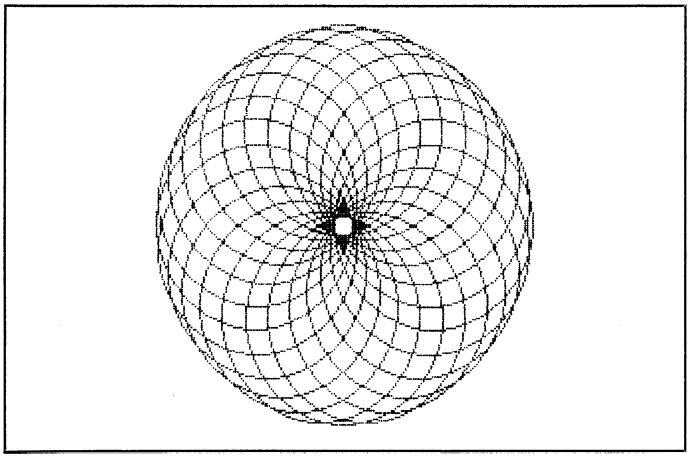

Bild 1. Beispiel einer HiRes-Darstellung am Commodore 64

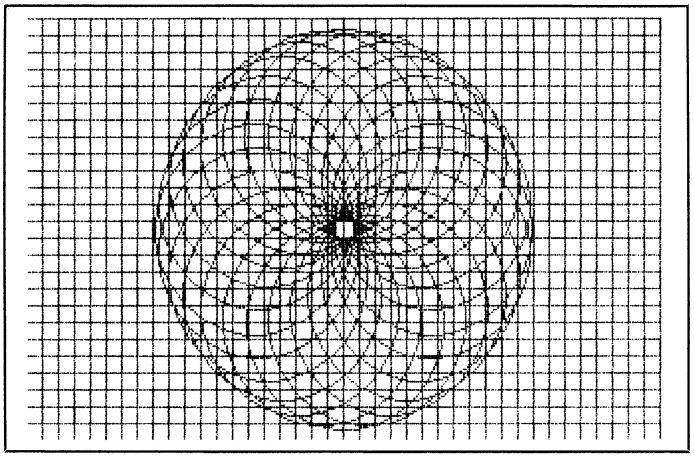

Bild 2. Die Rasterung verdeutlicht den Aufbau des HiRes-Bildschirms

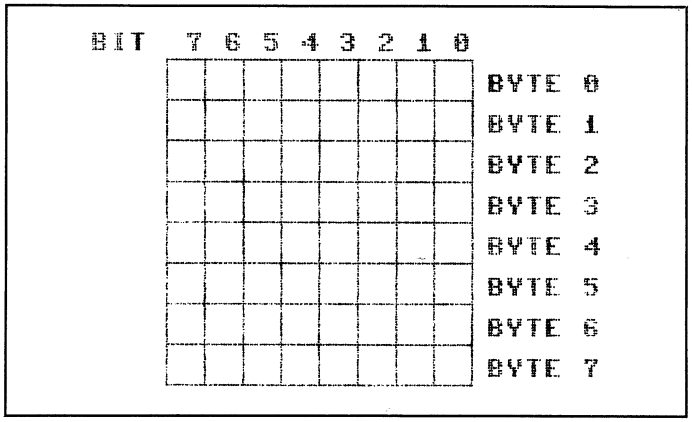

Bild 3. Aufbau einer einzelnen Rasterzelle aus Bild 2

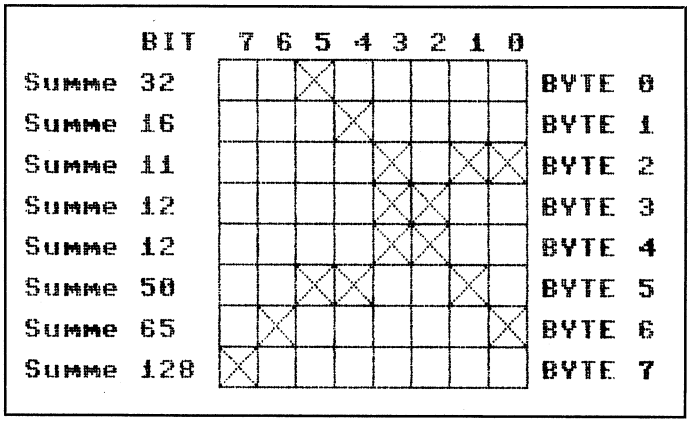

Bild 4. Beispiel zur Berechnung der Zeilen-Wertigkeiten innerhalb einer HiRes-Rasterzelle

acht Basisadressen wechseln, bis Sie den richtigen Bildschirm erwischt haben, effektiver dürfte da aber das kleine Basic-Programm sein, das in Listing 1 dargestellt ist. Mit Hilfe dieses kurzen Programms werden die Grafikbildschirme 3 (Startadresse 16384) bis 8 (Startadresse 57344) durch ein kurzes Maschinenprogramm auf die Startadresse des zweiten Grafikschirms verschoben und dort dargestellt. Dadurch wird es Ihnen möglich, auch die Grafikbildschirme 5 und 7 von Basic aus problemlos auszugeben, da diese Schirme normalerweise nicht erreichbar sind (sie liegen »unter« dem ROM). Weil sich dieser Artikel nur mit der Ausgabe von Grafik beschäftigt, verzichte ich an dieser Stelle auf Erläuterungen zu dem Programm, denn dabei würde zu stark auf die Besonderheiten des VIC II eingegangen werden müssen.

Nicht genug der acht Grafikschirme, verfügt der VIC II auch noch über zwei verschiedene Darstellungsarten: den sogenannten »High Resolution Mode« (HiRes, hochauflösende Grafik mit 64000 einzelnen Bildpunkten) und den »Multicolor-Mode«, in dem jeder Einzelpunkt in vier verschiedenen Farben dargestellt werden kann, bei dem aber dafür die Auflösung nur noch 32000 Bildpunkte beträgt. So mancher Anwender hat sich schon darüber gewundert, daß seine Hardcopy-Routine nur ein seltsames Strichmuster aufs Papier gebracht hat, aus dem man mit etwas Fantasie andeutungsweise das Originalbild entziffern konnte. Der Grund: im Multicolor-Modus werden in den Daten nicht nur die Bildinformation, sondern auch die Farbinformation verschlüsselt.

An dieser Stelle ein Tip: wenn Sie eine Grafik ausgeben wollen, jedoch keine Möglichkeiten haben, den Programmablauf zu unterbrechen, so können Sie durchaus einen Reset auslösen. Dabei wird das Bild normalerweise nicht gelöscht und Sie können dann fast immer von Basic aus weitere Schritte zur Ausgabe des Bildes einleiten.

High Resolution Mode

In Bild 1 ist zunächst einmal ein HiRes-Bild als solches dargestellt. In Bild 2 wurde über das Bild zusätzlich eine Schraffur gelegt, um die Struktur der Ablage des Bildes zu verdeutlichen. Durch das Rastermuster wird angedeutet, daß die Informationen nicht etwa Reihe für Reihe abgelegt werden, sondern in kleinen Blöcken, von denen in eine Zeile 40 passen, vertikal dagegen nur 25. Der Fachmann wird sich sofort an die Normaldarstellung von Zeichen erinnern. Tatsächlich ist diese Darstellung eng mit der normalen Zeichendarstellung verwandt.

Bild 3 zeigt eine Vergrößerung eines Blocks. Man erkennt, daß ein Block seinerseits aus einer Matrix von 8 x 8 Punkten besteht. Ergo besteht ein Block aus 64 Punkten, und wir wissen, daß 25 x 40 = 1000 dieser Blöcke auf dem Bildschirm Platz haben, wodurch also insgesamt eine Auflösung von 64000 Punkten erreicht wird. So weit, so gut. Aus Bild 3 geht aber noch weiter hervor, daß die Informationen innerhalb eines einzelnen Blocks zeilenweise abgelegt werden; ein Block setzt sich also aus 8 Bytes zusammen, die ihrerseits wieder in kleinste Informationseinheiten, die Bits, zerlegt werden können. Diese Blöcke werden bei der HiRes-Darstellung so lange aneinandergereiht, bis eine Zeile mit Blöcken (weiterhin auch als »Blockzeile« bezeichnet) gefüllt ist. Der nächste Block liegt dann naheliegenderweise unter dem ersten der vorangegangenen Zeile.

Damit es etwas anschaulicher wird, wollen wir uns an dieser Stelle mit einem kleinen Beispiel befassen.

Aus Bild 2 ist Block 27 in Zeile 13 vergrößert im Bild 4 wiedergegeben (Blockbereich 0-39, Zeilenbereich 0-24). Weiterhin wird für die Beispielrechnung angenommen, daß das Bild im Computer im Grafikschirm 2 (Basisadresse 8192) abgespeichert ist. Wenn Sie nun die Blöcke zeilenweise auszählen, werden Sie feststellen, daß der besagte Block die Nummer 547 (erster Block entspricht der Nummer 0) trägt.

Zunächst wollen wir klären, wie der Inhalt gerade dieses Blocks zu ermitteln ist. Dazu berechnen wir beispielsweise die Adresse von Byte 0 innerhalb dieses Blocks nach der Formel

```
Byte 0 = Basisadresse+Blockzeilen*320+Blocknummer*8
```

Wie läßt sich die Formel erklären?

Zunächst einmal ist es verständlich, daß die Basisadresse des Grafikschirms der Positionsberechnung zugrunde liegen muß. Weiterhin faßt eine Blockzeile 40 Blöcke zu je 8 Byte; eine Blockzeile besteht also aus 320 Byte. Nun muß man sich noch innerhalb der aktuellen Zeile zu dem Block bewegen, dessen Inhalt untersucht werden soll. Hierzu multipliziert man einfach die Anzahl der Blöcke mit 8, da sich ein Block aus 8 Byte zusammensetzt. Beachten Sie bitte bei der Berechnung, daß die Blockzeilen im Bereich 0 bis 24 liegen, die Blocknummern dagegen im Bereich 0 bis 39. So ergibt sich zum Beispiel als Startadresse für den 27. Block in der 13. Zeile:

```
Byte 0 = 8192 + 13 * 320 + 27 * 8 = 12568
```

Bei dem Block in unserem Beispiel ergibt also der Basic-Befehl

```
PRINT PEEK(12568)
```

den Inhalt von Byte 0 des Blocks 547. Wie groß wird nun dieser Wert sein?

Wie bereits erwähnt, besteht ein Byte aus kleinen Informationseinheiten (Bits). Das Bit kann entweder den Zustand »1« oder den Zustand »0« annehmen. Wenn Sie nun noch einmal Bild 3 betrachten, werden Sie erkennen, daß 8

solcher Bits ein Byte ergeben. Diese Bits werden mit Bit 0 bis Bit 7 von rechts nach links durchnumeriert. Da das Bit nur zwei Schaltzustände annehmen kann, wird zur Berechnung eines Bytes das Binärsystem zugrunde gelegt. Es ist

2^7=128
2^6= 64
2^5= 32
2^4= 16
2^3= 8
2^2= 4
2^1= 2

und schließlich

2^0= 1

Bereits mit diesen Vorüberlegungen läßt sich der Inhalt von Byte 0 berechnen. Wie Sie in Bild 4 sehen, ist nur Bit 5 gesetzt, also ergibt sich als Inhalt von Byte 0 der Wert 2^5 = 32. Diesen Wert würde auch die Abfrage »PRINT PEEK(12568)« ergeben.

Entsprechend ergeben sich zum Beispiel für Byte 2 als Inhalt

$2^3 + 2^1 + 2^0 = 8 + 2 + 1 = 11$.

Damit lassen sich auch alle anderen Bytes eines Blocks berechnen. Das Verfahren verläuft analog zum gezeigten Beispiel. Die Inhalte aller Bytes des Beispiels sind vollständig in Bild 4 wiedergegeben.

Der Drucker kommt ins Spiel

Zunächst muß der Drucker für die Datenausgabe vorbereitet werden. Hierfür müssen Sie sicherstellen, daß ein eventuell vorhandenes Interface keine falsche Interpretation von gesendeten Daten vornimmt. Dies geschieht im Normalfall durch Setzen einer bestimmten Sekundäradresse, bei einigen Herstellern auch als »Linearkanal« bezeichnet, weil gesendete Zeichen im Original an den Drucker weitergeleitet werden.

Ebenso muß der Zeilenvorschub korrigiert werden. Wenn man davon ausgeht, daß acht Punktreihen untereinander in einem Arbeitsgang gedruckt werden können, so ist der Zeilenvorschub so einzustellen, daß die nächste gedruckte Zeile sich nahtlos an die vorhergehende Zeile anschließt. Dazu ist der Zeilenvorschub im Regelfalle auf 24/216 Zoll zu stellen.

Damit sind die besonderen Voreinstellungen, die sowohl für den HiRes- als auch den Multicolor-Modus Gültigkeit besitzen, vorgenommen. Weitere Operationen beziehen sich nun auf den jeweiligen Modus, von dem ausgegangen wird.

Kommen wir schließlich zu dem schon anfangs erwähnten ESC-K und ESC-L. Mit diesen Sequenzen wird der BI-Mode eingeschaltet. Doch das reine Einschalten genügt nicht; man muß auch noch spezifizieren, wieviele Daten zum Drucker gesendet werden.

Schauen wir uns zunächst an, wie Grafik auf dem Drucker ausgegeben wird. Als Beispiel nehmen wir wieder Bezug auf Bild 4. Im ersten Teil haben wir überlegt, wie ein Block innerhalb des Computers abgelegt wird: ein Block besteht aus 8 Bytes, die horizontal untereinander stehen.

Beim Drucker ist das nicht üblich. Da der Druckkopf im allgemeinen nur aus einer Reihe von vertikal untereinander liegenden Dots (Nadeln) besteht, werden die Daten vertikal aufbereitet erwartet. Wird deshalb im Grafikmodus des Druckers ein Byte gesendet, so werden maximal 8 Punkte vertikal untereinander gedruckt. Wir können vom Computer her jedoch immer nur horizontale Punktreihen per PEEK abfragen. Was benötigt wird, ist eine Routine, die jeweils einen Block aus 8 x 8 Punkten entsprechend umrechnet. Bei unserem Beispiel muß der Block aus Bild 4 entsprechend Bild 5 umgerechnet werden. Bei der Berechnung gehen wir in diesem Fall von der Tatsache aus, daß das höchstwertige Bit der Druckerkopfdots innerhalb der Druckkopfmatrix oben liegt. In Ihrem Handbuch kann aber auch nachzulesen sein, daß das höchstwertige Bit in der Druckmatrix nach unten weist. In diesem Fall würden sich die berechneten Werte gemäß Bild 6 ändern. Wie Sie sehen, ist es durchaus wichtig zu wissen, wie die Dotmatrix Ihres Druckers aufgebaut ist.

Wir gehen davon aus, daß der zuerst beschriebene Fall zutrifft. Wie man relativ einfach die Matrix von Basic aus umrechnen kann, wird durch das Basic-Programm in Listing 2 illustriert. Mit diesem kleinen Programm können Sie jede beliebige Matrix umrechnen lassen. Dazu geben Sie nur die Startadresse Ihres Grafikschirms, die Zeile, in der sich der Block befindet und schließlich die Nummer des Blocks ähnlich wie oben in der Formel erwähnt an. Der Inhalt des betreffenden Blocks wird in dem Basic-Programm in Listing 2 in den Zeilen 190 bis 210 in das Feld BYTE eingelesen. Nun müssen die Daten gewandelt werden. Dazu folgende Überlegungen:

Um das erste Bit links oben des ersten vertikalen Bytes zu erhalten, muß von Byte 0 Bit 7 betrachtet werden (siehe Bild 4). Wenn es gesetzt ist, so wird es mit 2^7 = 128 multipliziert; man erhält so einen ersten Wert. Dann wird von Byte 1 Bit 7 betrachtet und untersucht, ob das Bit gesetzt ist. Falls dies der Fall ist, wird es mit 2^6 = 64 multipliziert und zum ersten Wert addiert. Dann wird jeweils nach dem gleichen Schema

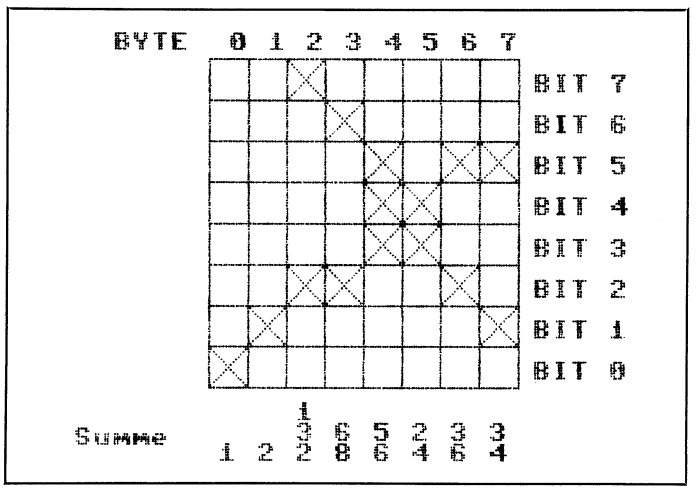

Bild 5. Das gleiche Zeichen wie in Bild 4, aber zur Druckeransteuerung um 90 Grad transformiert

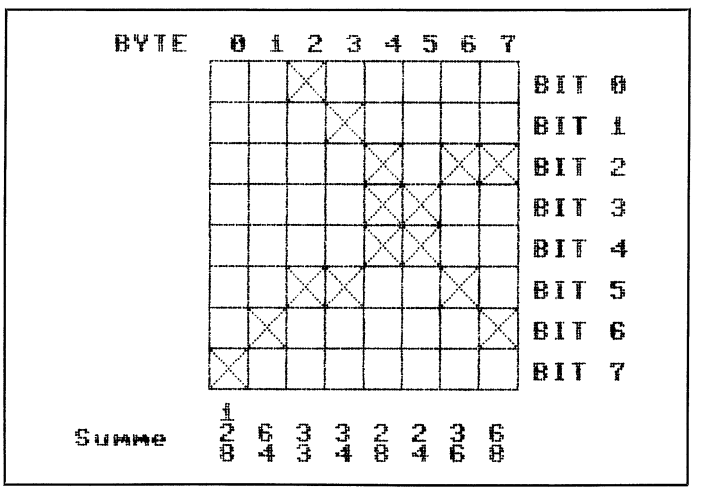

Bild 6. Wie Bild 5, aber für Drucker mit entgegengesetzter Nadelnumerierung (Bit 7 unten)

von Byte 2 bis 7 das 7. Bit untersucht und schließlich mit einer absteigenden Potenz von 2 multipliziert und zum jeweils schon erhaltenen Wert (im Listing 2 die Variable OUTPUT) hinzuaddiert. Hat man auf diese Art und Weise das erste vertikale Byte erhalten, kann selbiges an einen Drucker gesendet werden, denn jetzt ist die gewünschte Umwandlung vorgenommen worden, wovon Sie sich durch Nachrechnen überzeugen können.

Da der Block aber aus 8 Bytes besteht, wird dann mit der Aufbereitung des 2. vertikalen Bytes fortgefahren, also Bit 6 von Byte 0 untersucht. Ist es gesetzt, so muß es wieder mit 2^7 multipliziert werden, damit es an die korrekte Stelle innerhalb der Dotmatrix rutscht. Wieder erhält man einen ersten Zwischenwert. Dann wird Byte 1 auf Bit 6 hin untersucht. Ist es gesetzt, dann wird es mit 2^6 multipliziert, um dieses Bit an die korrekte Stelle der Dotmatrix des Druckers zu bringen. Nach diesem Verfahren wird fortgefahren, bis auch das letzte vertikale Byte berechnet worden ist.

Wenn Sie das Programm also starten und die Adresse eines Blocks eingeben, so erhalten Sie acht Werte, die jeweils (vergleiche hierzu nochmals den Zusammenhang zwischen den Bildern 4 und 5) die umgerechneten Werte wiedergeben.

Um die Effektivität und Berechnungszeit zu kürzen, wurde die Umrechnung in einen sehr kompakten Ausdruck eingebunden, über den Sie nicht gleich verzweifeln sollten (Zeile 240). Der Ausdruck kann nur verstanden werden, wenn Sie sich schon mal mit der »And/Or-Boolean-Wüste« des Computers beschäftigt haben, und ich möchte deswegen auch hier auf tiefergehende Erklärungen des Ausdrucks verzichten. Wenn Sie sich trotzdem für die Thematik interessieren, so empfehle ich Ihnen die im 64'er, Ausgabe 7/85 gestartete Serie »Logeleien«. Wichtig ist hier aber nur, daß Sie nachvollziehen können, wie die Umrechnung im Prinzip funktioniert.

Block an Block — Reihe an Reihe — fertig ist das HiRes-Bild

Da nun ein Programm erstellt ist, das einzelne Blöcke gemäß unseren Anforderungen aufbereitet, sind wir fast an unserem Ziel angelangt. Die Drucker, bei denen es nicht möglich ist, mehr als einen Block auf einmal zu senden, haben praktisch damit das endgültige Programm, mit dem der Druck von Grafik möglich gemacht wird. Besser dran sind da schon die Besitzer von Druckern, die es erlauben, Grafik zeilenweise auszugeben. Was jetzt nur noch fehlt, sind Schleifen, die die Blöcke hintereinander berechnen und eine korrekte Aufbereitung vornehmen.

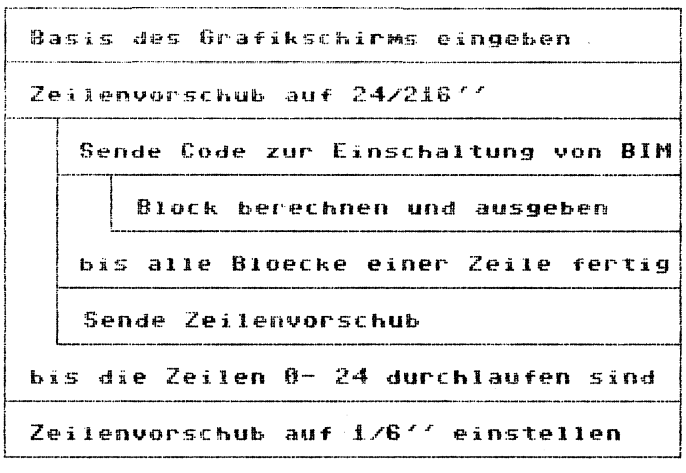

Bild 7. Struktogramm zur Erstellung einer HiRes-Hardcopy

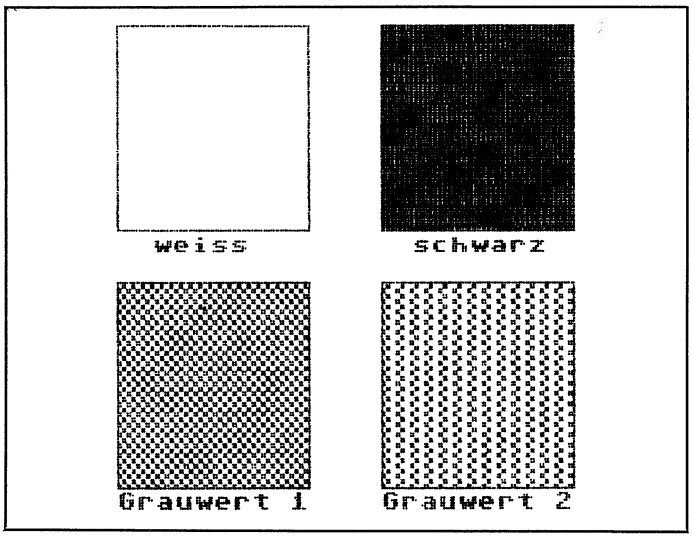

Bild 8. Verschiedene Graustufen für eine Multicolor-Hardcopy

Doch bevor die eigentliche Ausgabe beginnt: der Drucker muß bereit sein! Haben Sie also den Zeilenvorschub korrigiert?

Nun, dann dürfte Ihr Programm dem Struktogramm in Bild 7 entsprechen.

Das Struktogramm enthält einen Punkt, über den man sich noch Klarheit verschaffen sollte:

»Sende Code zur Einschaltung von HiRes«.

Hier stellt sich die Frage, was man dem Drucker mitteilen soll. Da die Daten einer Blockzeile hintereinandergereiht auch einer Blockreihe auf dem Drucker entsprechen sollen, wird deutlich, was dem Drucker mitgeteilt werden muß:

— BI-Mode einschalten, dabei geeignete Punktdichte wählen
— Anzahl der Daten definieren

Da 40 Blöcke zu je 8 Byte gesendet werden sollen, muß dem Drucker mitgeteilt werden, daß 320 Zeichendaten folgen. Üblicherweise wird dazu zur Einschaltung des BI-Mode ESC-L beziehungsweise ESC-K verwendet. Da die Vorgehensweise bei beiden Steuerbefehlen gleich ist und sie sich nur bezüglich Ihrer Punktdichten unterscheiden, betrachten wir nur ESC-L. Sie werden in Ihrem Druckerhandbuch finden, daß nach dem Einschalten von ESC-L noch definiert werden muß, wieviele Zeichendaten gesendet werden, bevor der Drucker wieder in seinen normalen Modus zurückschaltet. Würde nämlich keine solche Definition vorhanden sein, so würde der Drucker jedes gesendete Byte nach Einschaltung des BI-Mode als Grafikzeichen interpretieren und dann wäre zum Beispiel ein kontrollierter Zeilenvorschub unmöglich zu realisieren. Da Sie inzwischen wissen, daß sich eine Blockzeile im Computer und deshalb entsprechend auch später auf dem Drucker aus 320 Bytes zusammensetzt, müssen Sie vor der eigentlichen Ausgabe der Blockzeile dem Drucker die nachfolgende Anzahl von Zeichendaten mitteilen. Dazu werden üblicherweise zwei Zahlen n1 und n2 nach den Formeln

```
n1 = Zahl der Daten - INT(Zahl der Daten/256) * 256
```

und

```
n2 = INT(Zahl der Daten/256)
```

berechnet. In unserem Beispiel wären also n1 = 64 und n2 = 1. An den Drucker würde man also eine Steuersequenz senden, die folgendermaßen aussieht:

```
CHR$(27);"K";CHR$(64);CHR$(1);
```

Danach ist dann der Drucker im BI-Mode und erwartet 320 Zeichendaten, um dann wieder auf Normalbetrieb zurückzuschalten.

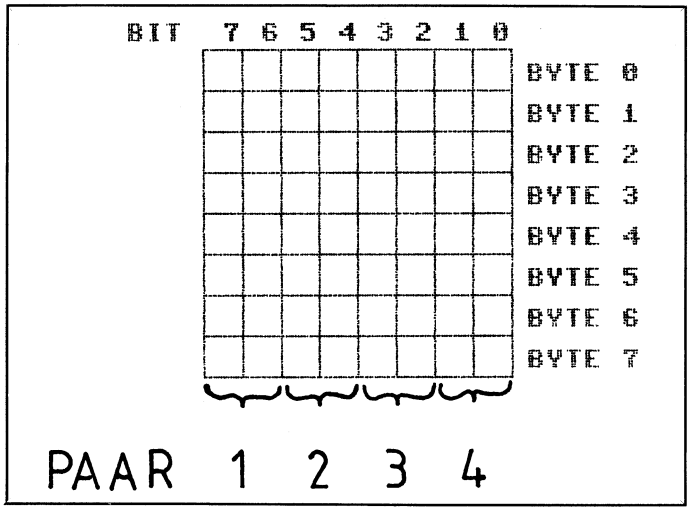

Bild 9. Die Aufteilung einer Rasterzelle in Bitpaare bei Multicolor

Das entsprechende Basic-Programm für die Ausgabe eines ganzen Grafikschirms im HiRes-Modus ist in Listing 3 dargestellt. Beachten Sie aber Ihre druckerspezifischen Einschränkungen, bevor Sie das Programm ausprobieren. Überprüfen Sie bitte, wie bei Ihrem Drucker die einzelnen Funktionen angesprochen werden. Hilfestellung sollten Ihnen dabei das Struktogramm aus Bild 7 und das eigentliche Basic-Programm mit den reichlich eingefügten REM-Statements in Listing 3 bieten. Wenn Sie das Listing 2 mit Listing 3 vergleichen, werden Sie weiterhin feststellen, daß der Algorithmus zur Umrechnung eines Blocks übernommen wurde. Mit diesen Abschlußbemerkungen wenden wir uns jetzt dem zweiten Problem zu: dem Druck von Grafik im zweiten Grafikmodus.

Der zweite Akt: Multicolor-Modus

Der Multicolor-Modus unterscheidet sich erheblich vom HiRes-Modus. Das liegt daran, daß mehrere Farben verwendet werden dürfen. Dabei werden immer zwei hintereinanderliegende Punkte zu einer Informationseinheit zusammengefaßt, aus der sich ableiten läßt, ob ein Punkt gesetzt ist, und wenn, in welcher Farbe. Dargestellt werden können in dieser Betriebsart des VIC II nur noch 32000 Bildpunkte.

Ein Punktpaar kann folgende Schaltzustände annehmen (1 = Bit gesetzt, 0 = Bit gelöscht):

00 Punkt nicht gesetzt, wird in Hintergrundfarbe dargestellt
01 Punkt gesetzt, Darstellung in Farbe 1
10 Punkt gesetzt, Darstellung in Farbe 2
11 Punkt gesetzt, Darstellung in Farbe 3

Wie man sich leicht selbst überlegt, muß eine Hardcopy einer Grafik dieses Modus also nicht nur zwischen gesetzten und nicht gesetzten Punkten differenzieren, sondern auch noch zusätzlich die Farbinformation weitergeben.

Drei Farben auf dem Papier — ist das möglich?

Mit einem normalen Drucker wird es selbstverständlich nicht möglich sein, Farben darzustellen. Der Drucker ist nur in der Lage, schwarze Farbe auf das Papier zu bringen. Es drängt sich hier der Vergleich mit einem Schwarzweiß-Fernsehgerät auf. Dort werden die unterschiedlichen Farben durch entsprechende Grauschattierungen wiedergegeben, die aus Mischprodukten von Weiß und Schwarz entstehen. Das gleiche Prinzip läßt sich auch auf dem Drucker verwirklichen. Dabei wählt man unterschiedliche Punktdichten, um so den Eindruck zu vermitteln, daß das Bild aus unterschiedlichen Farben besteht. Für uns reicht es wegen der maximal vier darstellbaren Farben aus, anzunehmen, daß die Farben Weiß (Hintergrundfarbe), Schwarz, und zwei Graustufen bereits das ganze Bild darstellen können. Bild 8 zeigt verschiedene Rastermuster.

So steht ein Multicolor-Bild im Speicher

Wieder wird über die Grafik ein Raster gelegt, welches das ganze Bild in einzelne Blöcke aufteilt. Eine Blockreihe besteht ebenfalls wieder aus 40 Blöcken. Ein Block ist in Bild 9 vergrößert. Zu erkennen ist die Struktur der Bitpaare, so daß im Prinzip eine Matrix vorliegt, mit der 4 Punkte in X-Richtung und 8 Punkte in Y-Richtung dargestellt werden können. Sie erhalten also eine 4 x 8-Matrix, die dann allerdings auch Informationen zur Farbe der einzelnen Punkte enthält. Zu finden ist nun ein Weg, der wieder Blöcke auswertet und zusätzlich noch Farbinformationen in Form eines Rastermusters mitliefert.

Wie kann man nun die Informationen, die der Block beinhaltet, verarbeiten?

Betrachten Sie zum Beispiel das fünfte Byte des Blocks in Bild 4. In Multicolor-Darstellung setzt sich das Byte aus 4 Bitpaaren zusammen, von denen das erste (Bitpaar 00) mit der Farbe des Hintergrundes, das zweite (Bitpaar 11) in Farbe 3, das dritte wieder in der Hintergrundfarbe und das vierte (Bitpaar 10) in Farbe 2 dargestellt werden würde. Ein Programm muß nun dem Inhalt eines Bitpaares verschiedene Punktdichten zuordnen, die dann als Graustufen auf dem Papier wiedergegeben werden.

Graustufen mit dem Drucker

Wenn davon ausgegangen wird, daß jedes Byte eines Blocks getrennt für sich ausgewertet wird, stellt sich einerseits der Sachverhalt relativ einfach dar, andererseits muß aber mit entsprechender Berechnungszeit kalkuliert werden.

Um ein gutes, kontrastreiches Bild entstehen zu lassen, muß die Punktdichte möglichst groß gewählt werden. Grafikfähige Drucker bieten hierzu meist einen BI-Mode »vierfache Dichte« an, mit dem eine genügend hohe Auflösung erzielt werden kann. Um das Bild nun möglichst im Maßstab 1:1 auf dem Drucker auszugeben, muß ein gesetzter Punkt auf dem Drucker durch ein kleines Quadrat wiedergegeben werden.

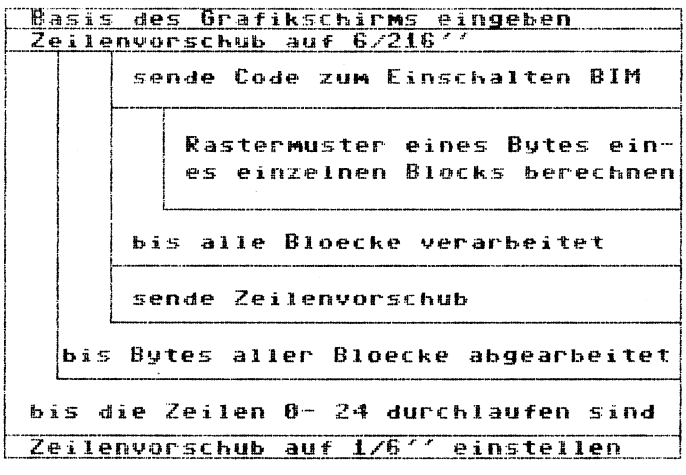

Bild 10. Struktogramm zur Erstellung einer Multicolor-Hardcopy

Bild 11. Beispielausdruck in »Multicolor«

Indem innerhalb eines solchen einzelnen Quadrates unterschiedliche Punkte gesetzt werden, erhält man unterschiedliche Grauwerte. Betrachten Sie hierzu einmal das Programm in Listing 4. Ähnlich wie im Fall einer HiRes-Grafik beschäftigen wir uns mit der Umwandlung eines einzelnen Blocks. Die Bestimmungsformeln zur Positionsbestimmung eines Blocks sind identisch mit denen zur Positionsbestimmung eines HiRes-Blocks, die Auswertung dagegen erfolgt anders. Die Rastermuster in den Zeilen 160 bis 190 erzeugen bei der gewählten vierfachen Punktdichte bei Ausgabe auf dem Drucker jeweils ein kleines Quadrat, das einen gesetzten Punkt in entsprechendem Grauwert wiedergibt (weiß wird hier auch mit einem Grauwert identifiziert). Diese Rastermuster werden zu Beginn des Programms in ein Feld DRUCKMATRIX eingelesen. Danach wird wie üblich definiert, welcher Block des Grafikbildschirms ausgegeben werden soll.

Während bei einer HiRes-Grafik die vertikalen Werte eines Blocks auf dem Bildschirm ausgegeben wurden, wird hier dagegen direkt der Drucker angesprochen, um das Prinzip der Umwandlung der Bitpaare in Rastermuster zu verdeutlichen. Da die Informationen innerhalb eines Blocks byteweise verarbeitet werden und da natürlich nach dem ersten berechneten Rastermuster für Byte 0 das zweite Rastermuster direkt unter dem ersten liegt, muß zusätzlich der Zeilenvorschub entsprechend korrigiert werden. Dazu wird der Zeilenvorschub auf zirka 6/216 Zoll gesetzt (siehe Zeile 270). Da weiterhin ein Byte des betrachteten Blocks aus vier Bitpaaren besteht, müssen selbstverständlich auch vier Rastermuster nebeneinander auf dem Drucker erzeugt werden.

Das Bit-Rastermuster

Weil das Rastermuster eines Bits aber schon aus acht Daten (siehe DATA-Zeilen) zusammengesetzt wird, werden also 32 Daten pro Byte an den Drucker gesendet. Daher erklärt sich die Sequenz CHR$(32);CHR$(0) in Zeile 300, durch die der BI-Mode eingeschaltet wird und 32 Daten als reine Grafikdaten vom Drucker erwartet werden. Dann wird in den Zeilen 330 der Inhalt eines Bytes eines Blocks ermittelt und daraufhin in den Zeilen 340 bis 380 in seine Bitpaare zerlegt.

Entsprechend dieser Zerlegung erhält die Variable RASTER einen Wert zwischen null und drei (wegen logischer Verknüpfungsoperationen). Durch die so zustande kommenden Inhalte der Variable RASTER kann an den Drucker die passende Druckmatrix gesendet werden. Mit Byte 1 bis 7 des gleichen Blocks wird genauso verfahren. Sie erhalten dadurch auf dem Drucker ein Rastermuster, dessen Aussehen von den Eingangsdaten abhängt.

Der nächste Schritt: Ausgabe eines Multicolor-Bildes

Ein Block besteht aus 8 untereinanderliegenden Bytes. Von diesen Blöcken stehen in einer Zeile 40 und ergeben so eine Blockreihe. Wenn wir nach dem oben beschriebenen Prinzip weiter vorgehen wollen, dann muß zunächst Byte 0 von Block 1 berechnet werden. Daraufhin darf man jedoch nicht das zweite Byte des Blocks auswerten, sondern muß Byte 0 des Blocks 2 berechnen, da die nächste Information neben dem Byte 0 des Blocks 1 liegt. Man fährt nach diesem Auswertungsprinzip fort, bis Byte 0 des Blocks 40 einer Blockreihe berechnet ist und führt erst dann einen Zeilenvorschub aus, der aber so bemessen sein muß, daß die folgenden Punkte direkt an die vorhergehenden anschließen.

Genauso wird jeweils Byte 1 bis 7 für alle Blöcke einer Reihe ausgewertet und daraufhin mit der nächsten Blockzeile fortgefahren. Es ergibt sich ein Algorithmus, der vollständig durch das Struktogramm in Bild 10 beschrieben wird. Dieses Struktogramm wurde in Listing 5 in ein Basic-Programm umgesetzt. Wieder werden Sie in wesentlichen Teilen die Übereinstimmungen zum Programm zur Berechnung eines Blocks im Multicolor-Modus erkennen.

Als Ergebnis Ihrer Bemühungen dürften Sie eine Hardcopy Ihres im Computer befindlichen Bildes erhalten, das etwa so wie das Beispiel in Bild 1 aussieht.

Auf ein Wort: Basic

Nun kennen Sie zwar die Geheimnisse, die es Ihnen ermöglichen, selbst hochauflösende Grafiken auf dem Drucker auszugeben, aber spätestens wenn Sie eines der beiden Programme aus Listing 3 oder 4 eintippen und anschließend starten, wird Ihre Freude gedämpft sein, denn der Computer macht seinem Namen alle Ehre: er rechnet. Die Berechnungen nehmen aber so viel Zeit in Anspruch, daß man mitunter die Geduld verliert. Was hier also wünschenswert erscheint, ist eine Programmiersprache, die die ganze Sache etwas beschleunigt. Das naheliegendste: Benutzung von Maschinensprache. Dadurch verkürzt sich zum Beispiel die Ausgabe einer HiRes-Grafik auf 30 Sekunden (etwa durch das in der 64'er-Ausgabe 8/84 auf Seite 83 veröffentlichte Listing). Das werden Sie zu schätzen wissen, wenn Sie einmal das entsprechende Basic-Programm haben laufen lassen und dann die Maschinenroutine dagegenhalten.

Wenn Sie also Maschinensprache beherrschen und nun das Prinzip der Grafikausgabe auf dem Drucker verstehen, empfehle ich Ihnen, eine Ihren Erfordernissen angepaßte Maschinenroutine zu schreiben, um Grafik auf dem Drucker auszugeben.

Deshalb gebe ich hier dem interessierten Maschinenspracheprogrammierer Informationen über eine möglichst einfache Ansteuerung des Druckers. Dazu zunächst eine Liste aller verwendbaren Systemroutinen und Zero-Page-Adressen:

BSOUT	$FFD2	Ausgabe eines Bytes auf aktivem Kanal
OPEN	$FFC0	Eröffnen eines Files auf dem Drucker
CKOUT	$FFC9	Ausgabe auf definiertem Kanal einleiten
CLRCH	$FFCC	Ausgabe wieder auf Standard zurücksetzen
CLOSE	$FFC3	Schließen eines Files auf dem Drucker
SETPAR	$FFBA	Setzen der Parameter für OPEN
LAENGE	$B7	für die Länge eines Strings bei OPEN

Bevor Sie ein File auf dem Drucker eröffnen, ist dem Computer erst mitzuteilen, welche Filenummer, welche Primäradresse und welche Sekundäradresse für das zu öffnende File zu verwenden sind. Da insbesondere beim Öffnen keine Zeichenkette an den Drucker zu senden ist, muß zunächst einmal das Flag LAENGE auf Null gesetzt werden:

```
LDA #0
STA LAENGE
```

Daraufhin wird in den Akku die Filenummer, in das X-Register die Primäradresse des Druckers (normalerweise 4) und in das Y-Register die Sekundäradresse eines Files geladen, das dann später durch OPEN eröffnet wird. Zuerst müssen aber mit der Systemroutine SETPAR die Werte für einen OPEN-Befehl gesetzt werden, bevor die OPEN-Routine angesprungen werden darf.

```
LDA #Filenummer
LDX #Primäradresse
LDY #Sekundäradresse
JSR SETPAR
JSR OPEN
```

Damit ist ein Kanal für die Ausgabe auf dem Drucker definiert. Die eigentliche Ausgabe auf dem Drucker wird durch das nochmalige Laden der Filenummer in das X-Register und das Anspringen der Systemroutine CHKOUT eingeleitet.

```
LDX #Filenummer
JSR CHKOUT
```

Danach kann man Zeichen, die im Akku stehen, mittels der Routine BSOUT auf dem Drucker ausgeben.

```
LDA #auszugebendes Zeichen
JSR BSOUT
```

Durch diese Routine wird die Ausgabe, die mittels der Routine BSOUT normalerweise auf dem Bildschirm erfolgen würde, auf den Drucker umgeleitet. Das heißt unter anderem, daß dann während dieser Umleitung keine Zeichenausgabe auf dem Bildschirm vorgenommen werden kann. Wollen Sie die Umleitung wieder aufheben, so können Sie dies mit Hilfe von CLRCH erreichen:

```
JSR CLRCH
```

Diese Systemroutine setzt die Ausgabe auf den Bildschirm (und die Eingabe auf die Tastatur) zurück. Beachten Sie bitte, daß das Druckerfile noch nicht geschlossen ist und Sie jederzeit wieder mit der beschriebenen Syntax bei der Systemroutine CHKOUT die Ausgabe erneut umleiten können.

Das Druckerfile schließen können Sie mit der Routine CLOSE. Dazu laden Sie die Filenummer in den Akku:

```
LDA #Filenummer
JSR CLOSE
```

Danach ist das Drucker-File ordnungsgemäß geschlossen. So, damit hätten wir unsere umfangreiche Expedition in das Gebiet der Grafikumsetzung für Druckausgaben abgeschlossen. Ich hoffe, daß Sie Ihnen ein wenig Spaß gemacht und Ihr Interesse an der Problematik geweckt hat. Ich wünsche Ihnen jedenfalls viel Erfolg bei der Programmierung einer eigenen Grafikroutine, sei es nun in Basic oder einer anderen Programmiersprache.

(Frank Lonczewski/ev)

```
100 REM ****************************      <112>
110 REM *                          *      <159>
120 REM *  HIRES-BILD VERSCHIEBUNG  *      <174>
130 REM *                          *      <179>
140 REM ****************************      <152>
150 DATA 165,001,072,169,000,133,251,133   <140>
160 DATA 253,169,032,133,254,120,169,052   <247>
170 DATA 133,001,162,032,160,000,177,251   <205>
180 DATA 145,253,200,208,249,230,252,230   <244>
190 DATA 254,202,208,240,104,133,001,088   <020>
200 DATA 096                               <176>
210 FOR I=4096 TO 4136:READ A:POKE I,A:PS=
    PS+A:NEXT I                            <023>
220 IF PS<>6115 THEN PRINT"PRUEFSUMMENFEHL
    ER":STOP                               <158>
230 FOR I=1 TO 7:PRINT"{CLR}GRAFIKBILDSCHI
    RM:";I                                 <230>
240 PRINT"{DOWN}STARTADRESSE:"I*8192:POKE
    252,I*32                               <253>
250 SYS 4096:GOSUB 280:PRINT"{CLR}"        <132>
260 POKE 53265,PEEK(53265)OR 32:POKE 53272
    ,PEEK(53272)OR 8:GOSUB 280             <002>
270 POKE 53265,PEEK(53265)AND 223:POKE 532
    72,PEEK(53272)AND 247:NEXT I:END       <208>
280 FOR J=1 TO 2000:NEXT J:RETURN          <108>
```

© 64'er

Listing 1. HiRes-Bildverschiebung. Bei der Eingabe bitte den Checksummer 64 in diesem Heft beachten.

```
100 REM ********************************    <238>
110 REM *                            *      <159>
120 REM * MATRIX- UMRECHNUNGSPROGRAMM *      <165>
130 REM *                            *      <179>
140 REM ********************************    <022>
150 INPUT"BASIS (X*8192) ";BASIS             <202>
160 INPUT"ZEILE (0-24){3SPACE}";ZEILE        <008>
170 INPUT"BLOCK (0-39){3SPACE}";BLOCK        <114>
180 DIM BYTE(7)                              <237>
190 FOR I=0 TO 7                             <003>
200 BYTE(I)=PEEK(BASIS+ZEILE*320+BLOCK*8+I
    )                                        <050>
210 NEXT I:PRINT:PRINT"AUSGABE DER BERECHN
    ETEN WERTE:":PRINT                       <078>
220 FOR I=0 TO 7:REM 7 BYTES VERTIKAL        <191>
230 OUTPUT=0:FOR I0=0 TO 7                   <017>
240 OUTPUT=OUTPUT-((BYTE(I0)AND 2↑(7-I))>0
    )*2↑(7-I0):NEXT I0                       <017>
250 PRINT OUTPUT:NEXT I                      <147>
```

© 64'er

Listing 2. Matrix-Umrechnungsprogramm. Bei der Eingabe bitte den Checksummer 64 in diesem Heft beachten.

```
100 REM********************************      <150>
110 REM*                            *        <159>
120 REM* HIRESBILDAUSGABE AUF DRUCKER *       <170>
130 REM*                            *        <179>
140 REM********************************      <190>
150 INPUT"BASIS (X*8192) ";BASIS:REM EINGA
    BE DER BASIS                             <074>
160 OPEN 1,4,4:REM ERDEFFNEN DES DRUCKERS
    MIT LINEARKANAL =>SEKUNDAERADRESSE 4     <148>
170 PRINT#1,CHR$(27);"3";CHR$(24);:REM ZEI
    LENVORSCHUB AUF 24/216"                  <091>
180 DIM BYTE(7)                              <237>
185 FOR J=0 TO 24:REM ZEILEN 1-25            <182>
```

Listing 3. HiRes-Bildausgabe auf Drucker. Bitte beachten Sie den Checksummer-Artikel.

```
186 PRINT#1,CHR$(27);"K";CHR$(64);CHR$(1);
    :REM SENDEN DES CODES FUER HIRES EIN   <160>
187 FOR K=0 TO 39:REM BLOECKE 1-39         <006>
190 FOR I=0 TO 7:REM AKTUELLER BLOCK UMREC
    HNEN                                   <150>
200 BYTE(I)=PEEK(BASIS+J*320+K*8+I)        <118>
210 NEXT I                                 <038>
220 FOR I=0 TO 7:REM 7 BYTES VERTIKAL      <191>
230 OUTPUT=0:FOR I0=0 TO 7                 <017>
240 OUTPUT=OUTPUT-((BYTE(I0)AND 2↑(7-I))>0
    )*2↑(7-I0):NEXT I0                     <017>
250 PRINT#1,CHR$(OUTPUT);:NEXT I:REM BLOCK
     AUSGEBEN, ENDE BLOCKUMRECHNUNG        <218>
260 NEXT K:REM ENDE BLOECKE EINER REIHE    <232>
270 PRINT#1,CHR$(10):REM ZEILENVORSCHUB    <018>
280 NEXT J:REM ENDE SCHLEIFE EINER REIHE   <083>
290 PRINT#1,CHR$(27)"2":REM ZEILENVORSCHUB
     WIEDER AUF 1/6" BRINGEN               <221>
300 CLOSE 1:REM SCHLIESSEN DRUCKERKANAL    <184>
```

© 64'er

Listing 3. HiRes-Bildausgabe auf Drucker. (Schluß) Bei der Eingabe bitte den Checksummer 64 in diesem Heft beachten.

```
100 REM ****************************       <112>
110 REM *                          *       <159>
120 REM *    MULTICOLORMATRIX      *       <226>
130 REM *  UMRECHNUNGSPROGRAMM     *       <135>
140 REM *                          *       <189>
150 REM ****************************       <162>
160 DATA 0,0,0,0,0,0,0,0:REM RASTERMUSTER
    BITPAAR (00)                           <189>
170 DATA 2,2,1,1,2,2,1,1:REM RASTERMUSTER
    BITPAAR (01)                           <109>
180 DATA 2,2,0,0,1,1,0,0:REM RASTERMUSTER
    BITPAAR (10)                           <074>
190 DATA 3,3,3,3,3,3,3,3:REM RASTERMUSTER
    BITPAAR (11)                           <220>
200 DIM DRUCKMATRIX(3,7):REM DREI RASTERMU
    STER MIT JEWEILS 7 DATEN               <128>
210 FOR I=0 TO 3                           <021>
220 FOR I0=0 TO 7                          <026>
230 READ DRUCKMATRIX(I,I0)                 <048>
240 NEXT I0,I                              <030>
250 INPUT"BASIS (X*8192) ";BASIS           <046>
251 INPUT"ZEILE (0-24){3SPACE}";ZEILE      <099>
252 INPUT"BLOCK (0-39){3SPACE}";BLOCK      <196>
260 OPEN 1,4,4:REM EROEFFNEN DES DRUCKERS
    MIT LINEARKANAL =>SEKUNDAERADRESSE 4   <250>
270 PRINT#1,CHR$(27);"3";CHR$(5);:REM ZEIL
    ENVORSCHUB AUF 6/216"                  <034>
290 FOR REIHE=0 TO 7 :REM REIHEN EINES BLO
    CKS                                    <025>
300 PRINT#1,CHR$(27);"Z";CHR$(32);CHR$(0); <033>
310 REM BIMODE 4-FACHE DICHTE EIN (MAX 192
    0 PUNKTE)                              <045>
330 BYTE=PEEK(BASIS+ZEILE*320+BLOCK*8+REIH
    E)                                     <031>
340 FOR I=6 TO 0 STEP-2:REM 4 BITPAARE AUS
    ZUWERTEN                               <166>
350 RASTER=-((BYTE AND 2↑(I+1))>0)*2-((BYT
    E AND 2↑I)>0)                          <169>
360 FOR I0=0 TO 7                          <168>
370 PRINT#1,CHR$(DRUCKMATRIX(RASTER,I0));:
    REM DRUCKMATRIX AUSGEBEN               <050>
380 NEXT I0,I:REM AUSWERTUNG ENDE          <153>
400 PRINT#1,CHR$(10):REM ZEILENVORSCHUB UM
     6/216"                                <057>
410 NEXT REIHE:REM NAECHSTE DES BLOCKS BEA
    RBEITEN                                <107>
420 PRINT#1,CHR$(27);"2":REM ZEILENVORSCHU
    B WIEDER AUF 1/6" BRINGEN              <198>
430 CLOSE 1                                <187>
```

© 64'er

Listing 4. Umrechnungsprogramm für Multicolor-Matrix. Bei der Eingabe bitte den Checksummer 64 in diesem Heft beachten.

```
100 REM ****************************       <112>
110 REM *                          *       <159>
120 REM *  MULTICOLORBILDAUSGABE    *       <076>
130 REM *       AUF DRUCKER        *       <013>
140 REM *                          *       <189>
150 REM ****************************       <162>
160 DATA 0,0,0,0,0,0,0,0:REM RASTERMUSTER
    BITPAAR (00)                           <189>
170 DATA 2,2,1,1,2,2,1,1:REM RASTERMUSTER
    BITPAAR (01)                           <109>
180 DATA 2,2,0,0,1,1,0,0:REM RASTERMUSTER
    BITPAAR (10)                           <074>
190 DATA 3,3,3,3,3,3,3,3:REM RASTERMUSTER
    BITPAAR (11)                           <220>
200 DIM DRUCKMATRIX(3,7):REM DREI RASTERMU
    STER MIT JEWEILS 7 DATEN               <128>
210 FOR I=0 TO 3                           <021>
220 FOR I0=0 TO 7                          <026>
230 READ DRUCKMATRIX(I,I0)                 <048>
240 NEXT I0,I                              <030>
250 INPUT"BASIS (X*8192) ";BASIS           <046>
260 OPEN 1,4,4:REM EROEFFNEN DES DRUCKERS
    MIT LINEARKANAL =>SEKUNDAERADRESSE 4   <250>
270 PRINT#1,CHR$(27);"3";CHR$(5);:REM ZEIL
    ENVORSCHUB AUF 5/216"                  <161>
280 FOR ZEILE=0 TO 24:REM ZEILEN           <009>
290 FOR REIHE=0 TO 7 :REM REIHEN EINES BLO
    CKS                                    <025>
300 PRINT#1,CHR$(27);"Z";CHR$(0);CHR$(5);  <184>
310 REM BIMODE 4-FACHE DICHTE EIN (MAX 192
    0 PUNKTE)                              <045>
320 FOR BLOCK=0 TO 39:REM FUER ALLE BLOECK
    E                                      <255>
330 BYTE=PEEK(BASIS+ZEILE*320+BLOCK*8+REIH
    E)                                     <031>
340 FOR I=6 TO 0 STEP-2:REM 4 BITPAARE AUS
    ZUWERTEN                               <166>
350 RASTER=-((BYTE AND 2↑(I+1))>0)*2-((BYT
    E AND 2↑I)>0)                          <169>
360 FOR I0=0 TO 7                          <168>
370 PRINT#1,CHR$(DRUCKMATRIX(RASTER,I0));:
    REM DRUCKMATRIX AUSGEBEN               <050>
380 NEXT I0,I:REM AUSWERTUNG ENDE          <153>
390 NEXT BLOCK                             <198>
400 PRINT#1,CHR$(10):REM ZEILENVORSCHUB    <148>
410 NEXT REIHE,ZEILE                       <020>
420 PRINT#1,CHR$(27);"2":REM ZEILENVORSCHU
    B WIEDER AUF 1/6" BRINGEN              <198>
430 CLOSE 1                                <187>
```

© 64'er

Listing 5. Multicolor-Bildausgabe auf Drucker. Bei der Eingabe bitte den Checksummer 64 in diesem Heft beachten.