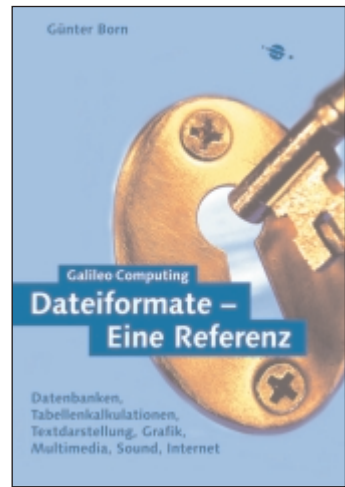


Dieses Kapitel stammt aus dem Buch
›Dateiformate – Eine Referenz‹
von Günter Born.

www.borncity.de

ISBN 3-934358-83-7
119,90 DM



Informationen zum Buch
mit Bestellmöglichkeit

www.galileocomputing.de

Galileo Computing

© Copyright 2001 by Galileo Press

Verlag und Autor schließen jede Haftung beim Gebrauch dieser Informationen aus.

33 ZSoft Paintbrush File Format (PCX)

Durch das Programm *Paintbrush* wurde das PCX-Format sehr populär. Dieses ursprünglich von der Firma ZSoft entwickelte Programm dient zur Speicherung von Grafiken und Bildern. Bedingt durch seine Popularität wird dieses Format von vielen Programmen unterstützt.

Leider besteht bei der Beschreibung der PCX-Dateiformate ein kleines Problem: Ausgehend von der ursprünglichen Definition für *PC Paintbrush* wurden zwischenzeitlich verschiedene Formatvariationen benutzt, was sich besonders bei der Kodierung der Farbdaten bemerkbar macht. Von diesen Änderungen waren insbesondere die ersten Versionen von *PC Paintbrush+* und *Publishers Paintbrush* betroffen. Eine Grafik wird gemäß Abbildung 33.1 durch Zeilen und Spalten in einzelne Bildpunkte aufgeteilt:

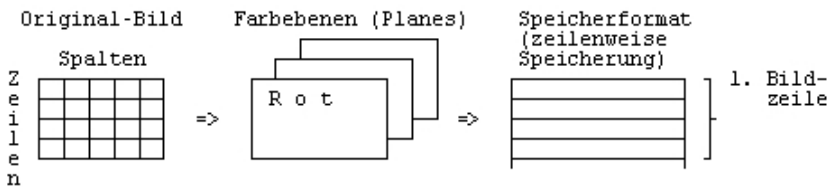


Abbildung 33.1 Umsetzung eines Bildes zur Speicherung im PCX-Format

Jeder durch eine Zeilen- und Spaltennummer adressierte Punkt des Originalbildes wird als *Pixel* (Picture Element) bezeichnet; das Gesamtbild entsteht durch zeilenweises Aneinanderfügen solcher Pixel. Bei Monochromdarstellung entspricht jedes Pixel einem Bit; ein gesetztes Bit erzeugt an dieser Stelle einen Punkt.

Aufwendiger wird die ganze Sache bei der Farbdarstellung: Hier lässt sich ein Pixel nicht mehr mit einem Bit darstellen; das Bild wird vielmehr in einzelne Farbebenen (Rot, Grün, Blau) zerlegt. Durch Addition dieser einzelnen Ebenen (*planes*) entsteht später das Bild in der Farbdarstellung. Die Kodierung dieser Farbinformationen ist zudem leider von der verwendeten Grafikkarte abhängig. Bei EGA- und VGA-Adaptoren werden die Farbebenen allgemein als *Bit Planes* bezeichnet. In Anlehnung an Abbildung 33.1 erhält man durch Zerlegen des Originalbildes in die Farbebenen eine dreidimensionale Bilddarstellung. In PCX-Dateien werden Bildinformationen zeilenweise abgespeichert, und zwar jeweils eine komplette Zeile mit Bildpunkten der betreffenden Ebene in der Reihenfolge »Ebene 0 = Blau, Ebene 1 = Grün, Ebene 2 = Rot, Ebene 3 = Intensität«. In der Datei ergibt sich damit folgendes Abbild:

Der Aufbau des PCX-Headers

Die PCX-Dateien besitzen – unabhängig von der jeweiligen Version – einen einheitlich aufgebauten Header von 128 Byte Länge. Dies gilt auch für Bildausschnitte, die in PCC-Dateien gespeichert werden. Tabelle 33.1 gibt das Format des PCX-Headers wieder.

Offset	Bytes	Bemerkungen
0	1	Identifikationsbyte: 0AH = PCX-Datei
1	1	PCX-Dateiversion 0 = Version 2.5 2 = Version 2.8 mit Palettendaten 3 = Version 2.8 ohne Palettendaten 5 = Version 3.0
2	1	Flag für komprimierte PCX-Dateien 0 = unkodiert, 1 = RLE-Kodierung
3	1	Bits pro Pixel (oder pro Ebene)
4	8	Koordinaten des Originalbildes als Worte: XMIN, YMIN, XMAX, YMAX
12	2	horizontale Auflösung eines Punktes in dpi (dots per inch)
14	2	vertikale Auflösung eines Punktes in dpi (dots per inch)
16	48	Color Map mit der Definition der Farbpalette; organisiert als 16 x 3 Byte-Feld
64	1	reserviert
65	1	Zahl der Farbebenen (maximal 4)
66	2	Bytes pro Bildzeile (gerade Zahl)
68	2	Palettendaten 1 = Farbe – S/W 2 = Graustufen
70	58	Leerbytes zum Auffüllen des Headers

Tabelle 33.1 Aufbau des PCX-Headers

Die Zahlen in den Spalten *Offset* und *Byte* sind in Tabelle 33.1 ausschließlich in dezimaler Schreibweise angegeben. Das erste Byte dient zur Identifikation einer gültigen PCX-Datei. ZSoft hat als Signatur den Wert 10 (0AH) festgelegt, und andere Hersteller richten sich in der Regel nach dieser Konvention.

Im zweiten Byte steht die Paintbrush-Versionsnummer, mit der die Datei erzeugt wurde. Der Wert 0 weist auf die Version 2.5 hin. Ab Version 2.8 besteht die Möglichkeit zur Aufnahme der Palettendaten in die PCX-Datei. Falls diese Information fehlt (z.B. bei s/w-Darstellung), wird das Byte mit dem Wert 03 belegt. Bei Farbdarstellungen sind ab Offset 68 die Palettendaten gespeichert. Das zweite Byte besitzt dann den Wert 02 zur Markierung der Version 2.8 mit Palettendaten. Ab Version 3.0 des Programms wird das Byte mit dem Wert 05H belegt.

Das dritte Byte enthält die Information darüber, ob die Daten der PCX-Datei kodiert sind. Bei einem Eintrag 0 liegt keine Kodierung vor. Mit dem Wert 1 wird signalisiert, daß die Daten im *Run Length Encoding*-Verfahren (RLE) komprimiert wurden; mehr dazu später.

Die Zahl der Bits pro Pixel (oder *Planes* pro Pixel) steht ab Offset 3. Dieser Wert berücksichtigt allerdings nicht die eventuell vorhandenen Farbebenen, weshalb sich dort in der Regel die Zahl 1 findet.

Ab Offset 04H beginnt eine Tabelle mit 4 Wörtern à 2 Byte, in denen die Dimension des Bildfensters definiert wird (siehe auch Bild 33.1). Die Koordinaten der linken oberen und rechten unteren Ecke des gültigen Bildabschnitts werden in der folgenden Reihenfolge abgelegt: XMIN, YMIN, XMAX, YMAX. Dies ist zum einen bei Bildausschnitten wichtig, wo die Koordinaten gesondert zu definieren sind. Weiterhin tritt häufig die Situation auf, daß sich die Zahl der gültigen Pixel pro Bild nicht ganz in Worten abbilden läßt. In diesem Fall befinden sich am rechten und unteren Bildrand Leerbits. Beim Lesen der PCX-Daten läßt sich dann der ungültige Bereich ausblenden.

In den Wörtern ab Offset 12 steht die Auflösung eines Bildpunktes in dpi (*dots per inch*). Die horizontale Auflösung (H_Res) findet sich ab Offset 12, während die Bildhöhe (vertikale Auflösung) ab Offset 14 gespeichert ist. Innerhalb der verschiedenen Paintbrush-Versionen ergeben sich hier allerdings einige Modifikationen, um die Darstellung geräteunabhängig zu gestalten, so daß die Werte mit Vorsicht zu genießen sind.

Ab Offset 16 ist ein 48 Byte langer Bereich für die Color Map reserviert. Bei Farbdarstellungen findet sich hier die Definition der einzelnen Farben; zum Verständnis des Aufbaus dieser Tabelle folgen nun einige Informationen zu den Möglichkeiten der Farbdarstellung:

Jede Farbe läßt sich grundsätzlich aus den folgenden drei Grundfarben mischen oder addieren:

Rot Grün Blau

Ein farbiger Bildpunkt ist im Grunde durch drei Bits darstellbar. Damit wird die Zahl der Farben aber auf $8 = 2^3$ beschränkt. Durch Hinzunahme eines weiteren Bits, des Intensitätsbits, sind 16 Farben möglich. Mittlerweile erlauben die Farbgrafikadapter aber eine wesentlich feinere Abstufung, die sich in der Abspeicherung der Farbinformationen niederschlägt. Dabei werden in der Regel die gleichen Verfahren benutzt, wie man sie bei Malern zur Erzeugung von Farbnuancen kennt: Verschiedene Farbtöne werden durch Variation der jeweiligen Anteile an den Grundfarben Rot, Grün und Blau erzeugt. Analog läßt sich auch bei Farbgrafikkarten verfahren. Wird der Anteil der Grundfarbe durch ein Byte angegeben, kann deren Anteil in 255 Stufen variiert werden. Bei drei Grundfarben sind nun drei Byte pro Farbton erforderlich. Damit wären aber auch drei Byte zur Darstellung der Farbe eines einzelnen Bildpunktes erforderlich, was einen enormen Spei-

cherbedarf erforderlich machen würde. Man greift hier deshalb zu einem Trick: Durch Abspeichern der Definition des Farbtons in einer Tabelle muß nur noch der Tabellenindex mit der Bildinformation gespeichert werden. Bei 16 Farben sind lediglich 4 Bit erforderlich, und bei 255 Stufen reichen immer noch 8 Bit. Die Tabelle, in der die Farbtöne definiert sind, wird in Anlehnung an das Werkzeug eines Malers als Palette bezeichnet – wie diese Palette gespeichert wird, ist jedoch abhängig von der verwendeten Grafikkarte.

CGA-Farbpaletteninformationen

Bei der standardmäßigen CGA-Karte von IBM (oder dazu kompatiblen Karten) ist die Abspeicherung der Color Map recht komplex. Die 48 Byte im Header werden in 16 Gruppen zu je drei Byte (Tripel) aufgeteilt. Das erste Byte der beiden ersten Tripel enthält die Daten.

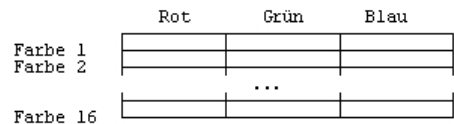


Abbildung 33.4 Speicherung der Color Map bei CGA-Karten

Daher sind nicht alle Byte eines Tripels belegt. Beim CGA-Adapter steht im ersten Byte (Offset 16) die Information über die Hintergrundfarbe, wobei nur die obersten vier Bits belegt sind; 16 verschiedene Farben sind darstellbar. Im ersten Byte (Offset 19) des folgenden Tripels findet sich in den oberen drei Bits die Information über die Vordergrundpalette. Es gilt dabei folgende Kodierung:

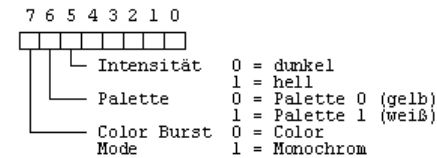


Abbildung 33.5 Kodierung des zweiten Bytes in der Color Map (CGA)

Im obersten Bit ist die Information über die Farbe (Monochrom/Farbe) gespeichert. Bit 5 legt die Intensität eines Bildpunktes fest. Mit Bit 6 kann eine der zwei CGA-Paletten ausgewählt werden. Hier gilt folgende Farbabstufung:

Palette	Farbabstufung
0	Grün, Rot, Braun (Gelb)
1	Cyan, Magenta, Weiß

Tabelle 33.2 CGA-Farbpaletten

Mit den drei Bit lassen sich insgesamt 8 verschiedene Kombinationen einstellen.

Anmerkung: Die Speicherung von PCX-Bildern mit einer CGA-Palette besitzt aber kaum noch Bedeutung, da sich Bilder mit minimal 256 Farben als Standard durchgesetzt haben.

EGA/VGA-Palette mit 16 Farben

Im Standard-EGA-/VGA-Modus stehen lediglich 16 Farben zur Verfügung. Hier wird die Color Map wieder in 16 Tripels à 3 Byte unterteilt. Jedem Byte wird eine Grundfarbe zugeordnet, in dem der jeweilige Farbtonanteil gespeichert ist. Die Werte eines Bytes können damit zwischen 0 und 255 liegen, so daß eine Umrechnung auf die RGB-Palette des Adapters erforderlich wird. Bei 16 Farben sind meist nur die obersten 2 Bits pro Byte definiert. Deshalb ergeben sich beim IBM-EGA-Adapter vier mögliche Stufen für die RGB-Farbpalette. Die in den drei Bytes gespeicherten Werte sind vor der Bearbeitung durch 64 zu dividieren und liegen dann bei 0, 1, 2 oder 3.

VGA-Palette mit 256 Farben

Pro Byte lassen sich im Grunde 256 verschiedene Farbstufen kodieren. Zur Unterstützung des VGA-Farbmodus hat ZSoft deshalb ab Version 3.0 eine Erweiterung der Color Map vorgenommen: Die Tabelle im Header faßt lediglich 16 x 3 Byte, weswegen eine eigene Tabelle mit 256 Einträgen à 3 Byte angelegt und an das Ende der Datei angehängt wurde. Hier finden sich dann die Informationen zur Farbpalette, die wie die Werte der (16 Farben der) EGA-Palette zu interpretieren sind (Division durch 64). Um die VGA-BIOS-Palette zu bestimmen, sind die Bytes der Tabelle durch 4 zu dividieren. Wird dieser Modus unterstützt, muß im Header ab Offset 2 (Version 3.0) der Wert 05H stehen – die letzten 769 Bytes der Datei sind dann in einem Puffer zu lesen. Findet sich im ersten Byte des Puffers die Signatur 0CH, liegen in den folgenden Bytes gültige Informationen über die Farbpalette vor. Damit ist die Beschreibung des *Color Map*-Feldes im Header abgeschlossen. Das Byte ab Offset 64 (40H) im Header der PCX-Datei ist für Erweiterungen reserviert.

Im Byte ab Offset 65 (41H) steht die Zahl der Color Planes, d.h., die Anzahl der Farbebenen pro Bildpunkt. Zur Zeit sind maximal vier Farbebenen (Rot, Grün, Blau, Intensität) möglich; bei Monochromdarstellung findet sich hier der Wert 1. Tabelle 33.3 enthält eine Übersicht über die möglichen Kombinationen.

Ebenen	Bit per Pixel	Bemerkung
1	1	Monochromes Bild mit 2 Farben
1	2	CGA-Bild mit 4 Farben
3	1	EGA-Bild mit 8 Farben (kaum benutzt)

Ebenen	Bit per Pixel	Bemerkung
4	1	EGA/VGA-Bild mit 16 Farben
1	8	VGA-Bild mit 256 Farben
3	8	True Color-Bild mit 16 Mio. Farben

Tabelle 33.3 Aufteilung der Ebenen und Bits per Pixel

Ab Offset 66 (42H) enthält der Header ein Wort mit der Anzahl der Bytes pro Bildzeile. Diese Zahl muß immer gerade sein, da die Bilddaten wortweise gespeichert werden; notfalls müssen beim Abspeichern also geeignete Füllbits benutzt werden. Beim Speichern farbiger Bilder entspricht dieser Wert auch der Zeilenlänge einer Farbebene.

Das Wort ab Offset 68 (44H) wird nicht immer benutzt. Hier liegt eine Information über die Interpretation der Palettenwerte in der Color Map. Der Eintrag 1 bedeutet, daß die Daten in der S/W- oder Farbdarstellung zu interpretieren sind. Beim Wert 2 sind alle Punkte als Graustufen gespeichert. Falls hier andere Werte eingetragen sind, ist das Feld unbenutzt und wird dem Füllbereich des Headers zugeordnet.

Am Schluß des Headers befinden sich ab Offset 70 weitere 58 Leerbytes, um die vorgeschriebene Länge von 128 Byte zu erreichen.

Anmerkung: Etwas verwirrend wird die Situation bei 24-Bit-True-Color-Bildern. Hier werden 24 Bit pro Bildpunkt verwendet. Zur Speicherung ist keine Palette erforderlich, die Farbinformation wird direkt im Bildpunkt gespeichert. Bei den 24-Bit-True-Color-Bildern wird ebenfalls die PCX-Version 3.0 im Header (Offset 01H) gesetzt. Diese Dateien lassen sich über die Einträge *Bits pro Pixel und Farbebene* (Offset 03H) und *Zahl der Farbenen* (Offset 41H) erkennen. Ergeben diese beiden Werte 24 Bit pro Pixel, liegt ein Echtfarbenbild vor.

Die Kodierung der PCX-Daten

An den Header schließen sich die Bilddaten an. Jedes Bild wird zeilenweise abgetastet und in seine Farbenen (Rot, Grün, Blau und Intensität) zerlegt. Dann werden alle Bits in der ersten Zeile der roten Farbebene wortweise zusammengefaßt und gespeichert. Unbelegte Bits im letzten Wort werden mit 0 aufgefüllt. Anschließend folgt die Abspeicherung der ersten Zeile der grünen Farbebene, der blauen Ebene und der Intensitätsinformationen. Erst danach kann mit der Bearbeitung der zweite Zeile der roten Farbebene begonnen werden. Dies wiederholt sich so lange, bis alle Zeilen gespeichert sind. Eventuell ungültige Bereiche am rechten und unteren Bildrand werden mit Nullbits aufgefüllt und müssen bei der Bearbeitung der Bilddaten ausgeblendet werden.

Die Informationen über den gültigen Bildausschnitt stehen im Header der PCX-Datei (XMIN, YMIN, XMAX, YMAX).

Text Marke

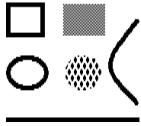


Abbildung 33.6 Original PC-Paintbrush-Bild

Bei den PCX-Dateien existieren nun zwei Methoden zur Ablage dieser Bilddaten in der Datei: Einmal lassen sich alle Bits in unkomprimierter Form ablegen – in diesem Fall steht im Header ab Offset 02 ein Wert ungleich 1. Leider ist bei diesem Verfahren wegen der unkomprimierten Dateien der Speicherbedarf recht hoch. Da jedoch bei Bildern mit mehreren einheitlichen Farbflächen viele Pixel identisch sind, stellt sich die Frage nach einer *Komprimierung* der Bilddaten. Beim *Run Length Encoding*-Verfahren steht im Header ab Offset 02H der Wert 1 – die Bilddaten liegen komprimiert vor.

Was verbirgt sich nun hinter diesem RLE-Verfahren? Man versucht einfach, gleiche Bits zusammenzufassen und deren Wiederholungsrate mit anzugeben. Hierzu gilt folgende Kodierung:

- Sind die beiden oberen Bits (6, 7) eines Bytes gesetzt (Bit = 1), liegt eine komprimierte Information vor. Die Bits 0 bis 5 werden dann als Zähler interpretiert, die den Wiederholungsfaktor für das im folgenden Byte gespeicherte Muster angeben.
- Sind die beiden oberen Bits eines Bytes *nicht* gesetzt (0), liegt ein reines Datenbyte vor. Die Bits 0 bis 7 lassen sich dann direkt als Bilddaten verarbeiten, d.h., bei der Speicherung unkomprimierter Bytes ist darauf zu achten, daß die beiden oberen Bits der Pixelfolge = 0 gesetzt sind.

Diese Komprimierung hat einige Konsequenzen: Mit dem RLE-Verfahren lassen sich mit 2 Bytes bis zu 63 Datenbytes oder 504 Bits kodieren, sofern alle Bits den gleichen Wert (0 oder 1) aufweisen. Bei Pixelreihen mit wechselnden Bits werden diese byteweise zusammengefaßt und (unkomprimiert) abgespeichert. Um beim Lesen Verwechslungen mit der RLE-Kennung auszuschließen, dürfen Datenbytes mit Werten größer 00H nicht direkt gespeichert werden. Vielmehr wird dann das Komprimierungsverfahren benutzt. Enthält eine Bildzeile zum Beispiel ein Bitmuster, das sich mit folgenden Hexzahlen angeben läßt:

```
FF FF FF FF C2 00 00 13 C9
```

ergibt sich folgende RLE-Kodierung:

```
C4 FF C1 C2 C2 00 13 C1 C9
```

Die zusammengehörenden Datenpaare sind dabei durch eine Unterstreichung markiert. Das Datenbyte 13 darf unkomprimiert gespeichert werden. Beim Wert C9 kommt es aber zu einem Konflikt mit der RLE-Kodierung. Das Datenbyte C9 wird deshalb als RLE-

Record mit den Codes C1 C9 gespeichert. Bei Bildern ohne gleiche Muster wird die Datei durch das Komprimierungsverfahren zwangsweise länger als beim reinen unkomprimierten Abspeichern der Bilddaten, da ja die Bits 6 und 7 eines Bytes nicht benutzt werden. Daher kann PC Paintbrush in diesem Fall Daten auch unkomprimiert speichern.

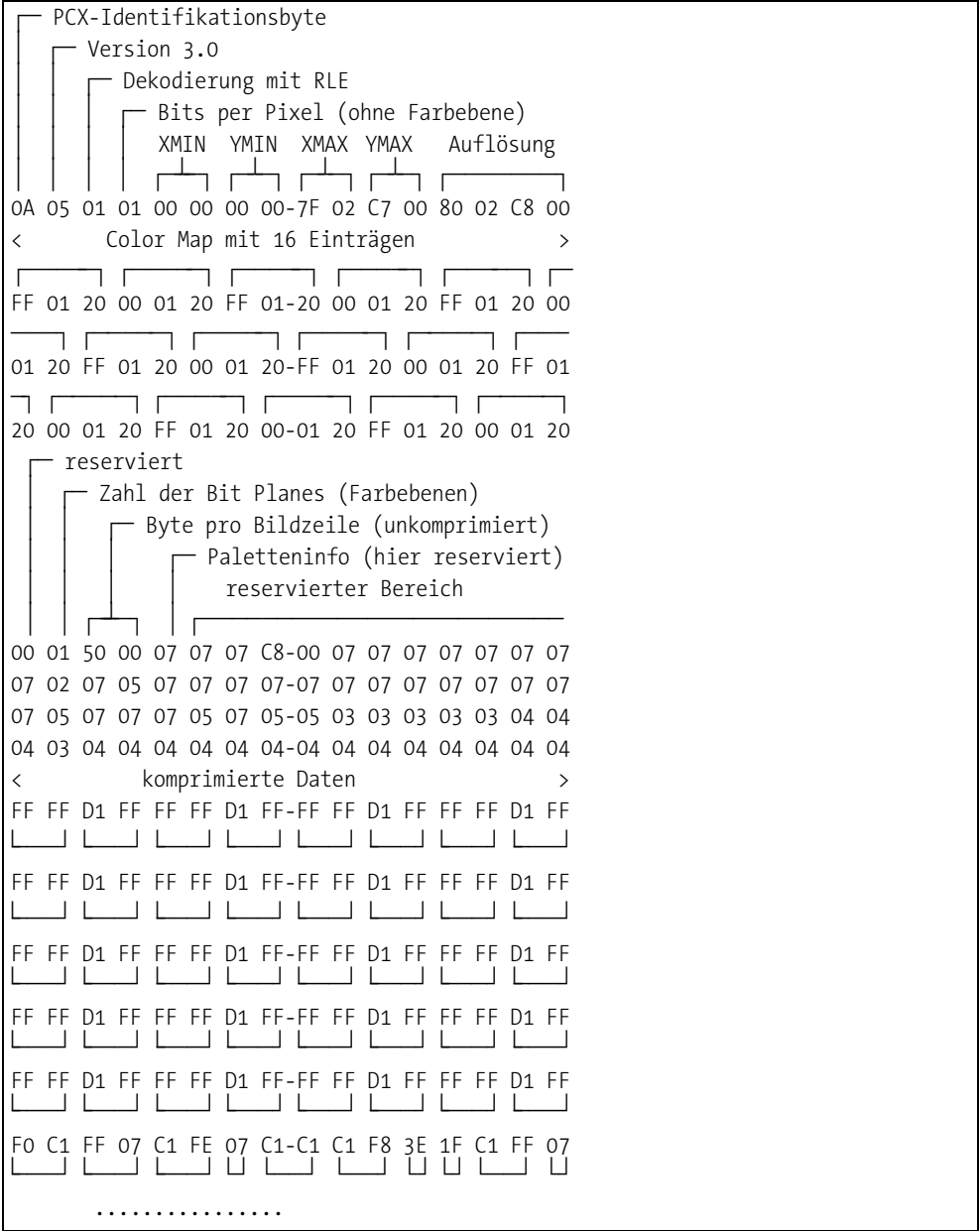


Abbildung 33.7 Auszug einer PCX-Datei als Hexdump

Anmerkung: Bei 24 Bit pro Pixel werden die Daten in der Regel auf drei Farbebenen (3 * 8) aufgeteilt. Dann ist ein 24-Bit-Farbwert aus den jeweiligen Bytes der drei Farbebenen zu kombinieren.

Abbildung 33.7 zeigt einen Ausschnitt aus einer PCX-Datei. Das Originalbild finden Sie in Abbildung 33.6. Mit den vorliegenden Informationen läßt sich der XE einer PCX-Datei leicht dekodieren.

Das Format der PC Paintbrush-Bitmap-Zeichen

Texte werden in PC Paintbrush als Bitmuster in das Bild eingefügt. Die Definition der Zeichen ist in eigenen Zeichensatzdateien mit einem recht einfachen Format abgelegt (siehe Tabelle 33.4).

Byte	Bedeutung
1	A0H + Fontbreite in Punkten
1	Fonthöhe in Punkten
256	256 * (Zeichenlänge + 1)
xxx	Pixel mit den Zeichendefinitionen

Tabelle 33.4 Aufbau der PCX-Zeichensatzdateien

Im *ersten Byte* ist die Signatur *A0H* in kodierter Form abgelegt. Das Feld wird gleichzeitig zur Definition der Zeichenbreite in Punkten benutzt. Zur Ermittlung der Zeichenbreite muß der Wert des ersten Bytes gelesen und um *A0H* erniedrigt werden.

Das *zweite Byte* enthält die Höhe einer Zeichenmatrix. Die Information ist erforderlich, da es verschiedene Zeichensatzdefinitionen mit 5 x 7 bzw. 9 x 14 Pixel pro Zeichen gibt. Ein Zeichen wird dabei in einzelne Zeilen zerlegt und abgespeichert. Bei der Matrix im Format 5 x 7 werden dann 7 Byte zur Speicherung gebraucht, während die 9 x 14-Pixel-Darstellung pro Zeile 2 Bytes belegt, so daß diese Zeichen 28 Bytes zur Abbildung benötigen. Die einzelnen Zeichen sind in Anlehnung an die ASCII-Tabelle definiert und ab Offset 258 abgelegt. Vor diesem Datenbereich befindet sich ein Feld mit 256 Byte, in dem die Länge eines jeden Zeichens + 1 gespeichert ist. Dies ist insbesondere für selbsterstellte Zeichensätze wichtig, da diese bis zu einem Maximalwert von 10 kByte jede beliebige Größe annehmen dürfen. Abbildung 33.8 zeigt den Speicherdump eines 5 x 7-Zeichensatzes.

Mit diesen Informationen lassen sich gegebenenfalls eigene Zeichensätze entwerfen und in PC Paintbrush einbringen.

		Zeichenbreite in Pixel			
		Zeichenhöhe in Pixel			
		256 * (Zeichenlänge+1)			
A5	07	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01 01	
01	01	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01 01	
01	01	06 02 04 06 06 06 06 06	03 03 03 06 06 03 06		
03	06	06 04 06 06 06 06 06 06	06 06 06 03 03 05 06		
05	06	06 06 06 06 06 06 06 06	06 06 04 06 06 06 06		
06	06	06 06 06 06 06 06 06 06	06 06 06 06 04 06 04		
06	06	03 06 05 05 05 05 05 05	05 05 04 05 05 05 06		
06	05	05 06 05 05 06 05 06 06	06 06 06 05 02 05		
06	06	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01	
01	01	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01	
01	01	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01	
01	01	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01	
01	01	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01	
01	01	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01	
01	01	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01	
01	01	01 01 01 01 01 01 01 01	01-01	01 01 01 01 01 01 01	
01	01	00 00 00 00 00 00 00 00	00-00	00 00 00 00 00 00 00	
		Definition der Zeichen			
00	00	00 00 00 00 00 00 00 00	00-00	00 00 00 00 00 00 00	
				
00	00	00 00 00 00 00 00 00 00	00-00	80 80 80 80 80 00 80	
A0	A0	00 00 00 00 00 00 00 00	50-50	F8 50 F8 50 50 20 78	
				

Abbildung 33.8 Dump einer 5 x 7-Zeichensatzdatei

Das CAPTURE File-Format (SCR)

Mit MS-Word für DOS wurde das Programm CAPTURE mitgeliefert. Aufgabe dieses Programms ist es, Bildschirmabzüge (Screen Shots) für Word unter DOS anzufertigen. Das Programm kann dabei die Bildschirmabzüge im Textmodus (Erweiterung LST) oder als Bitmap-Datei (Erweiterung SCR) speichern. Beim SCR-Dateiformat handelt es sich um eine modifizierte PCX-Variante. Der 128 Byte lange Header ist analog dem PCX-Format aufgebaut (siehe Tabelle 33.1). Es bestehen lediglich die folgenden Unterschiede:

- Das erste Byte (Offset 00) wird in PCX-Dateien auf 10 (0AH) als Markierung gesetzt. Eine SCR-Datei hat in diesem Byte den Wert 205 (CDH). Sofern Sie dieses Byte versuchsweise mit einem Debugger (z. B. DEBUG.COM) auf 0AH umsetzen, läßt sich die SCR-Datei mit dem Windows 3.1-Programm *Paintbrush* lesen und anzeigen.

- Die zweite Abweichung betrifft das Thema Bildschirmausschnitte. Bei PCX-Dateien wird jede Zeile mit einem Füllbyte abgeschlossen, um auf eine 16-Bit-Grenze zu kommen. Beim SCR-Format wird auf dieses Füllbyte verzichtet, d.h., eine Zeile weist die Zahl der im Header (Offset 67, 68) definierten Bytes pro Zeile auf.
- Die Daten werden mit RLE-Kodierung in der SCR-Datei gespeichert.

Mit diesem Wissen lassen sich SCR-Dateien recht einfach in das PCX-Format konvertieren und anzeigen.