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

17 Windows Bitmap-Format (BMP)

Windows 3.x, Windows NT und Windows 95 benutzen zum Speichern von Rasterbildern das BMP-Format. Dies ist ein geräteunabhängiges Format, welches Monochrom- und Farbbilder speichern kann. Das mit Windows 3.x gelieferte Programm PaintBrush sowie das in nachfolgenden Windows 9x- und Windows NT/2000-Versionen enthaltene Programm Paint sind wohl die populärsten Werkzeuge, die BMP-Dateien lesen und speichern können. Unter Windows 95 wird das gleiche Bitmap-Format durch das Programm MS Paint unterstützt. Eine BMP-Datei setzt sich aus mehreren Datenblöcken zusammen (Abbildung 17.1).

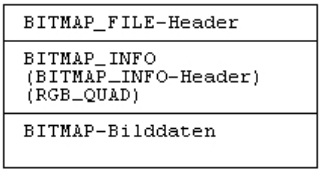

Abbildung 17.1 Aufbau des BMP-Formats (Windows 3.x)

Der BITMAP_FILE-Header ist eine Datenstruktur mit folgendem Aufbau:

Offset	Bytes	Name	Bedeutung
00H	2	bfType	File ID ('BM')
02H	4	bfSize	Filelänge in Byte
06H	2	-----	reserviert (muß 0 sein)
08H	2		reserviert (muß 0 sein)
0AH	4	bfOffs	Offset in Datenbereich

Tabelle 17.1 Aufbau des Windows BMP-Headers

Im ersten Word des Headers muß die Signatur 'BM' stehen, um eine gültige BMP-Datei zu identifizieren. Daran schließt sich ein Doppelwort mit der Dateilänge in Byte an. Der Header wird ab Offset 0AH mit einem 4-Byte-Zeiger abgeschlossen, der den Offset vom Dateianfang zum ersten Datenbyte des Bildes spezifiziert.

Anmerkungen: Dieser Header ist bei allen BMP-Versionen gleich. Die nachfolgend beschriebenen Strukturen weichen aber in den verschiedenen Windows-Versionen voneinander ab.

Erweiterter Header in Windows 3.x

Mit den Angaben im BITMAP_File-Header ist aber lediglich die Datei beschrieben. Was fehlt, sind Hinweise zum gespeicherten Bild. Deshalb schließt sich an den Datei-Header der BITMAP_INFO-Block an. Dieser enthält wiederum einen Header (BITMAP_INFO-

Header), der sich auf die Bilddaten bezieht. Der zweite Eintrag (RGB_QUAD) definiert die Farbpalette. Der BITMAP_INFO-Block für Windows 3.x-BMP-Dateien besitzt den in Tabelle 17.2 wiedergegebenen Aufbau.

Das Feld *biSize* (Offset 0EH) gibt die Länge des BITMAP_INFO-Headers in Byte an. An diesen Header schließt sich die Tabelle mit der Definition der Farbpalette (RGB_QUAD) an.

Offset	Bytes	Name	Bedeutung
			BITMAP_INFO-Header
0EH	4	biSize	Länge BITMAP_INFO-Header in Byte
12H	4	biWidth	Breite der Bitmap in Pixel
16H	4	biHeight	Höhe der Bitmap in Pixel
1AH	2	biPlanes	Farbebenen (muß 1 sein)
1CH	2	biBitCnt	Zahl der Bits pro Pixel
1EH	4	biCompr	Typ der Komprimierung
22H	4	biSizeIm	Bildgröße in Byte
26H	4	biXPels/m	horizontale Auflösung
2AH	4	BiYPels/m	vertikale Auflösung
2EH	4	biClrUsed	Zahl der benutzten Farben
32H	4	biClrImp.	Zahl der wichtigen Farben
			RGB_QUAD
36H	n*4		Definition für n Farben mit:
		rgbBlue	1 Byte Intensität Blau
		rgbgreen	1 Byte Intensität Grün
		rgbRed	1 Byte Intensität Rot
		rgbres	1 Byte reserviert

Tabelle 17.2 Aufbau des BITMAP_INFO-Blocks (Windows 3.x)

Die zwei Felder *biWidth* und *biHeight* geben Breite und Höhe des Bitmap-Bildes in Pixel an. Dies ist zur Erzeugung des späteren Bildes von Bedeutung.

Das Feld *biPlanes* (Offset 1AH) definiert die Zahl der Bitebenen (Planes) im Ausgabegerät. Der Eintrag muß zur Zeit auf 1 gesetzt werden.

Von besonderer Bedeutung ist das Feld ab Offset 1CH *biBitCount*. Dessen Wert definiert die Zahl der Bits pro Pixel und legt damit auch die Zahl der Farben im Bild fest. Hierbei sind folgende Werte erlaubt:

1: Definiert 1 Bit pro Pixel, was einem monochromen Bild entspricht. Ist das Bit gesetzt, wird die erste Farbe aus der Farbtabelle benutzt. Bei einem gelöschten Bit wird der Punkt mit der zweiten Farbe der Farbtabelle dargestellt.

4: Damit läßt sich ein Bild mit 16 Farben aufbauen. Die Farbpalette enthält dann 16 Einträge à 4 Byte. Jeder Bildpunkt besteht aus 4 Bit, d. h., ein Byte speichert die Farben zweier Bildpunkte. Der Wert 1FH steht dann für die 2. und 16. Farbe aus der Farbtabelle.

8: Diese Bitmap kann maximal 256 Farben enthalten, wobei jeder Bildpunkt durch 8 Bit dargestellt wird. Diese Werte dienen als Index in die 256 Elemente umfassende Farbtabelle. Der Wert 00H definiert dann einen Bildpunkt mit der ersten Farbe aus der Indextabelle.

24: Dieser Wert definiert ein Bild mit 2**24 (16 Millionen) Farben, d. h., jeder Bildpunkt wird durch 24 Bit dargestellt. Da die Farbtabelle in diesem Fall sehr groß würde, sind die Farben direkt im Bilddatenbereich kodiert. Die 24 Bit pro Pixel repräsentieren 3 Bytes, die die Intensitäten der Farben Rot, Grün und Blau im Bereich zwischen 0 und 255 definieren. Daraus wird von der Ausgabeeinheit dann die Mischfarbe generiert. Damit lassen sich Echtfarbenbilder erzeugen. Die Indextabelle mit den Farbdefinitionen im Header entfällt bei der 24-Bit-Darstellung.

Der Typ des Komprimierungsverfahrens der Bilddaten wird im Feld *biCompression* (Offset 1EH) als 4-Byte-Wert angegeben. Hierbei gibt es folgende Komprimierungsstufen:

BI_RGB: Die Bitmap-Daten liegen unkomprimiert vor.

BI_RLE8: Es wird ein Run-Length-Encoding-Verfahren (RLE) für Bitmaps mit 8 Bit pro Pixel verwendet.

BI_RLE4: Es wird ein Run-Length-Encoding-Verfahren (RLE) für Bitmaps mit 4 Bit pro Pixel verwendet.

Die Kodierung der Konstanten für die bisherigen Versionen von Windows sieht folgende Werte vor:

BI_RGB = 0
BI_RLE8 = 1
BI_RLE4 = 2

Eine Beschreibung dieser Komprimierungsverfahren findet sich weiter unten.

Das 4-Byte-Feld *biSizeImage* ab Offset 22H gibt die Länge des (komprimierten) Bitmap-Datenbereichs in Byte an. Die beiden folgenden Felder *biXPelsPerMeter* und *biYPelsPerMeter* definieren die Auflösung der Zieleinheit in Pels (Pels = Picture Elements) pro Meter für die x- und y-Achse. Damit kann eine Anwendung gegebenenfalls die für das Gerät am besten geeignete Bitmap aus einer Ressource auswählen.

Das Feld *biClrUsed* (Offset 2EH) spezifiziert, wie viele Farben das aktuelle Bild aus der Farbtabelle nutzt. Daher kann ein Bild weniger Farben enthalten, als nach der Farbtabelle möglich sind. Die wichtigsten Farben sollten deshalb zu Beginn der Indextabelle gespeichert werden. Ist der Wert des Feldes = 0, werden alle Farben benutzt. Die Ausführun-

gen gelten jedoch nur, falls weniger als 24 Bit pro Bildpunkt gespeichert werden. Ist das Feld *biCount* (Offset 1CH) auf den Wert 24 gesetzt, gibt der Inhalt von *biClrUsed* die Größe der Referenztabelle mit den Farben an. Normalerweise benötigt man bei der 24-Bit-Darstellung keine Farbtabelle. Aus Optimierungsgründen darf aber eine Farbtabelle mit angegeben werden. Folgt der Bereich mit den Bitmap-Daten direkt auf den BITMAP_INFO-Header, muß das Feld auf 0 gesetzt werden.

Das letzte Feld *biClrImportant* (Offset 32H) definiert die Zahl der für die Anzeige wichtigen Farben. Ist der Wert 0, sind alle Farben der Referenztabelle von Bedeutung.

Ab Offset 36H beginnt die Tabelle mit den Farbdefinitionen. Jede Farbe wird durch die Anteile der Farben Blau, Grün und Rot definiert. Die Definition umfaßt dabei 4 Bytes, wobei das letzte Byte unbelegt bleibt. Die Länge der Tabelle wird durch die Zahl der Farben (Bits pro Pixel) bestimmt.

Erweiterter Header in Windows NT

In Windows NT wird die gleiche Struktur für den BITMAP_INFO-Header benutzt (Tabelle 17.2). Allerdings kann das Feld ab Offset 1EH etwas geänderte Werte für die Komprimierung aufweisen.

0: unkomprimierte Daten

1: 8-Bit-RLE-Komprimierung

2: 4-Bit-RLE-Komprimierung

3: Bitfeld-Kodierung

Die Codes 0, 1 und 2 wurden auch unter Windows 3.x für BMP-Dateien definiert. Die neue Bitfeld-Kodierung wurde für Bilder eingeführt, die 16 oder 32 Bit pro Bildpunkt aufweisen. Dann schließt sich an Stelle der RGB_QUAD-Struktur die in Tabelle 17.3 gezeigte Struktur mit den drei 32-Bit-Werten für die Farbfiltermasken an den BITMAP_INFO-Header an.

Offset	Bytes	Name	Bedeutung
36H	4	redMask	Maske für roten Farbanteil
3AH	4	greenMask	Maske für grünen Farbanteil
3EH	4	blueMask	Maske für blauen Farbanteil

Tabelle 17.3 Erweiterte Maskenwerte im BITMAP-Header (Windows NT)

Zur Dekodierung der Bilddaten ist ein Wert für einen Bildpunkt zu lesen. Dieser Wert (16 oder 32 Bit) ist dann ein Bitfeld, welches die Rot-, Grün- und Blauanteile des Bildpunkts enthält. Die Farbanteile lassen sich dekodieren, indem der Wert des Bildpunkts mit der betreffenden Maske über eine AND-Funktion verglichen wird. Bei den 32-Bit-Maskenwerten befinden sich die gültigen Werte in den höherwertigen Bits. Bei einem Vergleich

mit 16-Bit-Bilddaten bleiben die niederwertigen Bits unberücksichtigt. Gültige Maskenwerte wären zum Beispiel:

redMask = F800 0000H
greenMask = 07E0 0000H
blueMask = 001F 0000H

Damit erhalten Sie 5 Bit für den roten Farbanteil, 6 Bit für den grünen Farbanteil und 5 Bit für den blauen Farbanteil. Bei 32 Bit werden in der Regel 10 Bit für jeden Farbanteil benutzt (die zwei niederwertigsten Bit der Maske bleiben frei).

Erweiterter BMP4-Header (Windows 95)

Mit Windows 95 wurde ein neuer BITMAPV4HEADER an Stelle des BITMAP_INFO-Headers eingeführt. Dieser besitzt die gleichen Felder wie der BITMAP_INFO-Header, verfügt aber über zusätzliche Felder (Tabelle 17.4).

Offset	Bytes	Bedeutung
		BITMAPV4HEADER
0EH	4	Länge Header in Byte
12H	4	Breite der Bitmap in Pixel
16H	4	Höhe der Bitmap in Pixel
1AH	2	Farbebenen (muß 1 sein)
1CH	2	Zahl der Bits pro Pixel
1EH	4	Typ der Komprimierung
22H	4	Bildgröße in Byte
26H	4	horizontale Auflösung (Pixel/Meter)
2AH	4	vertikale Auflösung (Pixel/Meter)
2EH	4	Zahl der benutzten Farben
32H	4	Zahl der wichtigen Farben zusätzlicher BMP4-Felder
36H	4	Maske roter Farbanteil
3AH	4	Maske grüner Farbanteil
3EH	4	Maske blauer Farbanteil
42H	4	Maske Alphakanal
46H	4	Color Space-Typ
4AH	4	X-Koordinate roter CIE-Endpunkt
4EH	4	Y-Koordinate roter CIE-Endpunkt
52H	4	Z-Koordinate roter CIE-Endpunkt
56H	4	X-Koordinate grüner CIE-Endpunkt

Offset	Bytes	Bedeutung
5AH	4	Y-Koordinate grüner CIE-Endpunkt
5EH	4	Z-Koordinate grüner CIE-Endpunkt
62H	4	X-Koordinate blauer CIE-Endpunkt
66H	4	Y-Koordinate blauer CIE-Endpunkt
5EH	4	Z-Koordinate blauer CIE-Endpunkt
62H	4	Gamma rote Koordinate
66H	4	Gamma grüne Koordinate
6AH	4	Gamma blaue Koordinate

Tabelle 17.4 Aufbau des BITMAPV4HEADER-Blocks (BMP4 Windows 95)

Die Maskenwerte werden wie bei Windows NT benutzt, um 16- oder 32-Bit-Bildpunkte in die einzelnen Farbanteile zu zerlegen. Windows 95 unterstützt bei 16-Bit-Bildwerten jedoch nur 5-5-5-oder 5-6-5-Masken. Bei 32-Bit-Bildwerten sind dagegen 8-8-8-Masken anzuwenden.

Das BMP4-Format unterstützt mehrere Farbsysteme (RGB, CIE). Der Typ des Farbsystems, in dem die Bilddaten kodiert sind, wird ab Offset 46H (Color Space-Typ) abgelegt. Dieses Feld kann dabei folgende Werte annehmen:

0: RGB-Farbsystem (kalibriert)

1: RGB-Farbsystem (geräteabhängig)

2: CMYK (geräteabhängig)

Standardmäßig werden die Bilddaten in einer geräteabhängigen RGB-Farbkodierung abgelegt (dies entspricht den BMP3-Bilddaten). Kalibrierte RGB-Bilddaten werden durch den 1931-CIE-XYZ-Standard festgelegt. Wird der Typ des Farbsystems auf 0 (RGB-Farbsystem kalibriert) gesetzt, folgen die X-, Y- und Z-Koordinaten der Endpunkte für die rote, grüne und blaue Farbe im CIE-Farbsystem. Die drei letzten 32-Bit-Felder sind als DWORD kodiert und enthalten die Skalierungswerte für die Gamma-Koordinaten der drei Farben Rot, Grün und Blau. Wird kein kalibriertes RGB-Farbsystem benutzt, bleiben die betreffenden Felder im BITMAPV4HEADER leer.

Der Datenbereich

An die beschriebenen Kopfdaten schließt sich der Datenbereich mit den Bitmap-Daten an. Das Bild wird Zeile für Zeile abgetastet und gespeichert. Dabei lassen sich die Daten gegebenenfalls komprimieren (was aber kaum genutzt wird). Hierbei wird zwischen der Komprimierung der Bilder mit:

- 4 Bit pro Pixel (RLE4)
- 8 Bit pro Pixel (RLE8)

unterschieden. Eine Bildzeile wird dabei immer zusammenhängend gespeichert. Gegebenenfalls werden die fehlenden Bildpunkte am rechten Rand bis zur Grenze von 32 Bit mit 0 ergänzt. Das Bild besitzt dabei seinen Ursprung in der linken unteren Ecke. Die (unkomprimierten) Bilddaten werden dabei Pixel für Pixel bearbeitet. Enthält ein Bild zum Beispiel 8 Bit pro Pixel, entspricht jedes Byte einem Bildpunkt. Bei 24 Bit pro Pixel müssen 3 Byte pro Bildpunkt gelesen werden. Bei 4 und 8 Bit pro Bildpunkt wird die Farbe eines Bildpunkts über eine Farbpalette ermittelt.

Auf NT-Rechnern werden 16-Bit-Bilddaten im Motorola-Format abgelegt — das Format wird auch als *big-endian* bezeichnet und bedeutet, das MSB wird auf der unteren Adresse gespeichert. Sie müssen auf Intel-Rechnern daher erst die Bytes tauschen, bevor Sie die Masken anwenden.

Im BMP3- und im BMP4-Format auf Windows 3.x- und Windows 95-Rechnern sind die Daten im Intel-Format (little endian) abgelegt. Sie können die Daten direkt lesen und ggf. über die Maskenwerte entschlüsseln.

8-Bit-RLE-Komprimierung

Sofern in der Bitmap pro Bildpunkt 8 Bit benutzt werden, verwendet Windows die 8-Bit-RLE-Kodierung. Hierbei können Bilddaten in zwei Varianten vorliegen.

Folgen gleicher Bits werden in 2 Bytes komprimiert:

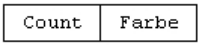

Abbildung 17.2 RLE-Kodierung (Typ 1)

Das erste Byte spezifiziert, wie oft das folgende Byte zu duplizieren ist. Dieses Byte definiert einen Index in die Farbtabelle, aus der dann die Farbe rekonstruiert wird. Aus der Sequenz:

```
03 04H
```

wird die Folge:

```
04 04 04H
```

generiert, die drei Bildpunkte mit der Farbe Nummer 5 (Zählung beginnt bei 0) erzeugt. Der Zähler darf dabei Werte zwischen 1 und 255 enthalten.

Ist das erste Byte einer Sequenz dagegen mit 00H belegt, wird die zweite Variante (Absolute Mode) zur Speicherung verwendet. Es gilt dann folgende Kodierung:

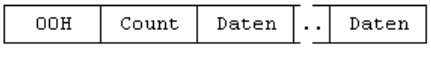

Abbildung 17.3 RLE-Kodierung (Typ 2)

Sofern im zweiten Byte ein Wert zwischen 3 und FFH steht, handelt es sich um einen Datenrecord. Der Wert gibt dann die Zahl der folgenden Datenbytes an. Diese Bytes enthalten jeweils einen Index in die Farbtabelle, der pro Byte einen Bildpunkt kodiert. Die Sequenz:

```
00 04 3F 66 01 07H
```

definiert dann vier Bildpunkte mit unterschiedlichen Farben.

Enthält das zweite Byte dagegen Werte zwischen 0 und 2, handelt es sich um einen ESCAPE-Record. Mit diesem Record läßt sich das Ende einer Zeile oder einer Bitmap markieren. Hierbei gilt folgende Belegung:

Byte 2	Bedeutung
0	Ende einer Zeile
1	Ende der Bitmap
2	Delta-Record

Tabelle 17.5 Kodierung des ESCAPE-Modus

Die Sequenz 00 00H markiert demnach den Abschluß einer Zeile. Das folgende Byte gehört dann bereits zum nächsten Record. Mit 00 01H wird das Ende der Bitmap markiert.

Der Delta-Record besteht aus 4 Bytes und besitzt folgenden Aufbau:

```
00 02 xx yy
```

Die Zahlen xx und yy definieren einen relativen Offset in Bildpunkten ab der aktuellen Position, an der der nächste Bildpunkt auszugeben ist. Damit lassen sich Bitmaps mit wenigen Punkten recht effizient speichern. Die Bytefolge:

```
┌─────┐  ┌──┐  ┌──────────────┐  ┌──────────┐
03 44  04 11  00 03 01 33 FF  00 02 05 01H
```

wird dann umgesetzt in:

```
44 44 44H
11 11 11 11H
01 33 FFH
5 Punkte rechts, 1 Punkt hoch
```

4-Bit-RLE-Komprimierung

Die Komprimierung entspricht im wesentlichen der 8-Bit-RLE-Kodierung. Steht im ersten Byte ein Wert zwischen 1 und 255, ist dies ein Wiederholungszähler, der angibt, wie viele Bildpunkte aus dem Folgebyte zu konstruieren sind. Das Folgebyte enthält bei 4 Bits pro Bildpunkt dann immer zwei Farbindizes (bzw. Bildpunkte). Aus den oberen 4 Bits

wird der erste Farbwert ermittelt und ausgegeben. Dann werden die unteren 4 Bits zur Ausgabe des nächsten Farbpunktes benutzt. Anschließend wird der dritte Punkt wieder aus den 4 oberen Bits konstruiert. Diese Methode wird so lange wiederholt, bis die angegebene Zahl von Bildpunkten erzeugt wurde.

Ist das erste Byte 00, wird die absolute Kodierung der 8-Bit-RLE-Kodierung verwendet, d. h., das zweite Byte muß einen Wert zwischen 3 und FFH aufweisen. Daran schließen sich n Datenbytes an, wobei jedes Byte zwei Bildpunkte repräsentiert. Im *Absolute Mode* muß die Bytefolge auf Word-Grenzen enden. Für die ESCAPE-Sequenzen (zweites Byte = 0, 1 oder 2) gelten die gleichen Bedingungen wie für die 8-Bit-RLE-Kodierung.

Das Windows RLE-Format (RLE)

Windows 3.x und 9x benutzen zum Beispiel zur Ausgabe des Startlogos ein weiteres Dateiformat mit der Erweiterung RLE. Eine Analyse des Windows-Startlogos als RLE-Datei ergab, dass die Daten in einer RLE-Datei mit dem RLE4-Verfahren komprimiert sind. Dies ergibt dann eine geringfügig kleinere Datei als ein unkomprimiertes BMP-Bild.