SpecialHost, ein Programm zum Auswerten von special-Strings für den AMIGA

β-Version 0.94

vom
25. August 1991

von
Georg Heßmann

Abstract

SpecialHost ist ein Hilfsprogramm für den Previewer ShowDVI und den Druckertreiber DVIprint. Es wertet die special-Strings aus, die die Treiber dem Programm übergeben. Es ist in der Lage IFF-Bilder in den TEX-Text einzubinden, sowie TPIC Specials auszuwerten. Der vollständige Source liegt dem Programm bei.

Inhaltsverzeichnis:

1. Copyright und ähnliches

Das Programm SpecialHost ist "freely distributable copyrighted software". Das heißt, jeder darf es sich kopieren und verwenden, das Copyright bleibt aber vollständig bei mir (Georg Heßmann).

Das Programm darf auch nur unverändert weitergegeben werden.

Das Programm darf ohne meine ausdrückliche Genehmigung nicht kommerziell verkauft werden. Als PD-Software vertrieben darf der Preis 5 DM pro Diskette nicht überschreiten.

Es darf auch nur in vollständiger und unveränderter Form weitergegeben werden.

Diese Anleitung "`spec.tex`" ist ebenfalls Copyright Georg Heßmann und darf nicht kommerziell vervielfältigt werden. Man darf sie sich ausdrucken und auch kopieren, man darf sie aber **nicht** für Geld weitergeben.

Die Sourcen dürfen verändert werden, solange diese deutlich gemacht wird und die Copyright-Notiz unverändert im Source verbleibt. Diese geänderten Sourcen dürfen aber nur mit dem explizieten Hinweis, daß sie geaendert wurden, weitergegeben werden. Und dies auch nicht im Rahmen von PasTEX, sondern nur auf einer eigenen Diskette.

Wenn Änderungen gemacht werden, sollten diese mir nach Möglichkeit mitgeteilt werden, damit sie in die allgemeine Version eingebaut werden können.

Zu beachten ist das Copyright der Files 'flext.c' und 'specialparse.c'.

Für weitere Erklärungen siehe das README-File.

2. Allgemeines über SpecialHost

Jeder, der auch nur ein wenig mit TEX gearbeitet hat, kennt dessen gravierendsten Nachteil: die fehlende Möglichkeit, Graphiken zu erstellen und in den Text einbinden zu können. Es gibt zwar Ansätze, dies TEX beizubringen (in LaTEX oder PICTEX), aber sie sind zum Scheitern verurteilt, wenn die Graphiken auch nur etwas komplizierter werden.

Nur mit einem Trick kann man diese Einschränkung umgehen: die Verwendung des \special-Kommandos. Glücklicherweise wurde im DVI-File-Format eine Möglichkeit vorgesehen, auch nicht Standardisiertes in das DVI-File aufzunehmen. Dazu dient das \special-Kommando. Alles was in geschweiften Klammern diesem Kommando folgt, wird unverändert in das DVI-File übernommen. So können treiberabhängige Anweisungen am TEX-Programm vorbei ins DVI-File gelangen. Der jeweilige Treiber kann den \special-String auswerten und zum Beispiel eine Graphik in die aktuelle Seite einbinden.

Der Nachteil dieser Methode soll aber nicht verschwiegen werden. Sobald man mit \special-Strings arbeitet, ist das DVI-File nicht mehr portabel. Das heißt, man kann dieses File nicht mehr auf beliebige Rechner übertragen und erwarten, daß die Ausgabe dort genau so aussieht, wie auf dem eigenen Rechner. So gravierend

ist diese Einschränkung allerdings auch nicht, da im schlimmsten Fall die \special-Strings einfach ignoriert werden und der Text eben ohne Graphik ausgedruckt wird.

Wie arbeitet SpecialHost?
SpecialHost nutzt die Möglichkeit des Multitasking und des Message-Transfers des AMIGA aus, um auf möglichst flexible Art und Weise die \special-Strings auszuwerten. SpecialHost wird parallel zu dem jeweiligen Treiber (ShowDVI oder DVIprint) gestartet und wartet auf eine hereinkommende Meldung des Treibers. Trifft der Treiber bei der Auswertung des DVI-Files auf einen \special-String, so übergibt er diesen unverändert dem SpecialHost-Programm zusammen mit ein paar weiteren Informationen wie der derzeitigen Auflösung des Bildes.

Die Aufgabe des SpecialHost-Programms ist es, die Message des Treibers abzuholen und den \special-String auszuwerten. Wird in dem String der Name eines IFF-Files angegeben, so lädt das Programm das angegebene Bild und paßt es der eingestellten Auflösung an (verkleinert oder vergrößert es). Danach übergibt es die so erzeugte Bitmap dem Treiber. Dieser muß sie nur noch in seine eigene Bitmap, in der er eine ganze TEX-Seite hält, einkopieren.

Warum aber ein eigenes Programm für die Auswertung des \special-Strings?
Diese Lösung ist meiner Meinung nach erheblich flexibler, als die gesamte Auswertung des \special-Strings in jeden Treiber einzubauen. Dadurch, daß eine fest definierte Schnittstelle zwischen den Treibern und dem SpecialHost-Programm existiert, kann man die Programme alle unabhängig voneinander weiterentwickeln, ohne auf größere Schwierigkeiten zu stoßen.

So kann man zum Beispiel das Format des \special-Strings umdefinieren oder neue Fähigkeiten in das SpecialHost-Programm einbauen wie die Auswertung von Farb- oder Vektorgraphiken, ohne auch nur eine Winzigkeit in den Treibern ändern zu müßen. Oder falls ich etwas in den Treibern ändere und jemand sich ein eigenes SpecialHost-Programm geschrieben hat, sollten keine Probleme bei der Zusammenarbeit dieser Programme auftreten.

Damit der Vorteil zum Tragen kommt, gebe ich den gesamten Source des SpecialHost-Programms heraus. Vielleicht findet sich jemand, der sich in der Bildverarbeitung gut auskennt und Lust hat, das Programm zu erweitern.

3. Der Aufruf von SpecialHost

Im derzeitigen Entwicklungsstadium besitzt SpecialHost noch keine einzige Option, die man ihm beim Aufruf angeben könnte.

Die einzige Besonderheit ist vielleicht, daß sich das Programm von selbst in den Hintergrund legt. Der CLI-Prompt erscheint wieder sofort nach dem Aufruf. Das Programm ignoriert auch 'stdin' und 'stdout'. 'stderr' wird dazu benutzt, im schweren Fehlerfall eine Meldung auszugeben. In solch einer Situation beendet sich das Programm.

4. Der Aufbau des Fensters von SpecialHost

Das Window von SpecialHost ist in drei Teile unterteilt. Der obere Teil dient der Eingabe. Dort sind alle Gadgets, mit denen das Programm gesteuert wird.

Der mittlere Teil besteht lediglich aus einer Statuszeile, in der angezeigt wird, auf welche Art das Programm die Bilder übergeben soll.

Der untere Teil ist als Kontrollfenster für die Aktivitäten des Programms gedacht. Dort wird mitprotokolliert, was das Programm gerade so macht. Es merkt sich auch die Zeilen, die nach oben herausgeschoben werden, wenn es viele Meldungen ausgibt. Diese können mit den Cursor-Tasten wieder hervorgeholt werden.

5. Die Bedienung von SpecialHost

Das Programm wird vollständig über Gadgets gesteuert. Dies sind:

`Memory` Wenn man dieses Gadget anklickt, so veranlaßt man das SpecialHost-Programm, einen Zeiger auf die Bitmap des einzubindenden Bildes an den Druckertreiber oder Previewer zu übergeben.

`File` Dieses Gadget bewirkt, daß dem Treiber nur ein Dateiname übergeben wird, in dem die Bitmap in einem besonderen (einfachen) Format abgespeichert ist. Das File hat als Endung die DPI-Auflösung des Treibers. Dabei wird allerdings erst geprüft, ob dieses File jüngeren Datums ist als das eigentliche IFF-File. Ist das IFF-File neuer – es wurde also geändert – oder existiert das Bitmap-File noch nicht, so wird die Bitmap im Speicher erstellt und das Bitmap-File neu angelegt. Da ja dann das Bild ohnehin schon im Speicher steht, wird nun die Bitmap direkt übergeben.

`Mem-B` Dieses Gadget ist verwandt mit dem `Memory`-Gadget. Der einzige Unterschied ist, daß der Treiber zusätzlich angewiesen wird, einen Rand um das Bild zu zeichnen.

`File-B` Dieses Gadget ähnelt dem `File`-Gadget, der Treiber zeichnet jedoch einen Rand um das Bild.

`Border` Hier weist das SpecialHost-Programm den Treiber an, lediglich einen Rand, der der Größe des Bildes entspricht, zu zeichnen. Die Größe ermittelt das SpecialHost-Programm, indem es den Header des IFF-Files einliest.

`Rect.` Hat die selbe Funktion wie das `Border`-Gadget. Nur wird nicht ein Rand, sondern ein ausgefülltes Rechteck gezeichnet.

`Base DPI` Mit diesem Gadget gibt man an, in welcher Auflösung die IFF-Bilder gezeichnet sind, die man in den TEX-Text einbindet. Es wird standardmäßig auf 100dpi gesetzt. Dies bedeutet, daß ein Bild im Previewer, eingestellt auf 100dpi, unverändert angezeigt wird. Bei einem Druckertreiber mit 300dpi Auflösung aber, wird das Bild um den Faktor 3 vergrößert. Wenn die `Base DPI` auf 100 eingestellt bleibt, so bedeutet dies, daß 100 Punkte im IFF-Bild immer ein Inch auf der Seite ausmachen – egal mit welcher Auflösung dieses Bild angeschaut oder ausgedruckt wird.

Invert Wenn dieses Gadget aktiviert ist, wird die Bitmap des einzubindenden Bildes invertiert.

Blitter SpecialHost kennt zwei verschiedene Methoden um schwarz-weiß Bilder zu vergrößern – verkleinern. Einmal mittels einer Software-Routine. Diese ist zawr nicht übermäßig schnell, liefert dafür aber recht gute Ergebnisse. Dann gibt es noch eine Routine, die den Blitter zu Hilfe nimmt. Diese ist **erheblich** schneller, ezeugt aber manchmal eine etwas schlechtere Ausgabe. Mit dem Gadget `Blitter` kann man entscheiden, welche Art der Skalierung verwendet werden soll.

`Icon` Durch Anklicken dieses Gadgets wird das Fenster in ein kleines Icon verwandelt. Öffnen kann man dieses Icon durch einen Doppelklick. Das Programm arbeitet auch als Icon ungestört weiter.

`Show` Diese Gadget holt den ShowDVI-Screen nach vorne.

6. Das Menü

Die Funktionen von SpecialHost können auch über ein Menü aufgerufen werden.

Es gibt drei Menüs: `Project`, `Draw Modus` und `Misc`

6.1. Das Untermenü Project

`About` Gibt ein paar Zeilen Copyright-Text aus.

`Read Config` Liest das Konfigurationsfile erneut ein. Setzt also alles wieder in den Urzustand zurück.

`Save Config` Speichert die derzeitige Konfiguration in ein File ab. Dazu gehören auch so Sachen wie die aktuelle Fensterposition.

`To ShowDVI` Enspricht dem `show` Button.

`Iconify` Entspricht dem `icon` Button.

`Use PubScr` Dies funktioniert nur unter 2.0. Wenn dies angewählt ist, versucht SpecialHost sein Fenster auf dem ShowDVI Public-Screen zu öffnen.

`Quit` Beendet das Programm.

6.2. Das Untermenü Draw Modus

Die sechst Menüpunkte dieses Untermenüs entsprechen genau den 6 Gadgets auf der rechten Seite des SpecialHost Fensters.

6.3. Das Untermenü Misc

`Invert` Enstpricht dem `Invert` Button.

`Use Blitter` Entspricht dem `blitter` Button.

7. Das Format des special-Strings

Damit die TEX-Treiber ShowDVI und DVIprint das Programm SpecialHost nach einzubindenden Bildern abfragen, müssen diese wissen, nach was sie fragen sollen. Dies steht im TEX-Text, die Anweisung ist der \special-String. Alles was bei

`\special`{beliebiger Text}

dem \special-Kommando zwischen den geschweiften Klammern steht, wird vom TEX-Programm ignoriert und unverändert zum Treiber durchgereicht. Trifft nun ShowDVI oder DVIprint auf solch einen String, so wird überprüft, ob ein SpecialHost-Programm im Hintergrund läuft. Ist dies nicht der Fall, so wird das \special-Kommando einfach ignoriert. Existiert das SpecialHost-Programm, so bekommt dieses den vollständigen String per Message übermittelt.

Es ist also das SpecialHost-Programm, welches den Special-String auswertet und die Bedeutung der einzelnen Wörter des Strings festlegt. Da dieser Special-String in keiner Weise standardisiert ist, kann sich das Format von Treiber zu Treiber ändern.

Das vorgestellte Format gilt also nur für diese Kollektion von Treibern und ist auch hier noch nicht endgültig festgelegt. In neueren Versionen werden sicherlich noch Funktionen hinzukommen. Es wird aber auf Abwärtskompatibilität Wert gelegt, so daß man ohne Probleme auf neuere Versionen umsteigen kann.

Das derzeitige Format der \special-Strings:

`\special{anw1 anw2 anw3}`

Die einzelnen Anweisungen innerhalb der Klammern werden mit ein oder mehreren Blanks voneinander getrennt.

Eine Anweisung wiederum besteht aus einem Schlüsselwort und einem Wert, so daß eine genauere Betrachtung des \special-Strings folgendes Format hervorbringt:

`\special{key1=val1 key2=val2 key3=val3}`

Zwischen Schlüsselwort (key) und Wert (value) muß immer ein Gleichheitszeichen stehen.

Als Schlüsselwörter gibt es bisher:

- `ifffile` definiert den Filenamen eines IFF-Files,
- `hoffset` gibt einen horizontalen Offset an,
- `voffset` gibt einen vertikalen Offset an,
- `hsize` legt die horizontale Größe des Bildes fest,
- `vsize` legt die vertikale Größe des Bildes fest.
- `mode` gibt den Type des IFF-Files an. Dies ist entweder '`bw`' für schwarz-weiß Bilder, '`color`' für Farbbilder oder '`ham`' für HAM-Bilder.
- `red` legt den Rot-Anteil des Farbbildes fest. Damit kann der Rot-Anteil des Bildes verstärt oder vermindert werden.
- `green` legt den Grün-Anteil des Farbbildes fest.
- `blue` legt den Blau-Anteil des Farbbildes fest.
- `gamma` definiert die "Gamma-Korrektion" für das Farbbild. Damit kann das gesamte Bild aufgehellt werden. Die Helligkeit der Farbe (r, g, b) wird bestimmt durch:

$$h = 255 * \left(\frac{(red * r + green * g + blue * b)}{15 * (red + green + blue)} \right)^{gamma}$$

Bei `ifffile` wird als Wert der Filename eines IFF-Files angegeben. Dazu ist zu beachten, daß das aktuelle Directory jenes ist, in welchem das SpecialHost-Programm gestartet wurde. Aus diesem Grund sollte der Filename eines IFF-Files immer den kompletten Pfad enthalten. In der jetzigen Version ist es leider noch nicht möglich, das aktuelle Directory innerhalb des SpecialHost Programms zu ändern.

Bei den nächsten vier Schlüsselwörtern wird als Wert eine Dimensionsangabe verlangt. Diese besteht aus einer reellen Zahl plus einer Einheitsbezeichnung. Folgende Einheiten stehen zur Verfügung:

`pt` point (wird von TEX verwendet)
`pc` pica $1pc = 12pt$
`in` inch $1in = 72.27pt$
`bp` big point $72bp = 1in$
`cm` centimeter
`mm` millimeter
`dd` didôt point $1157dd = 1238pt$
`cc` cicero $1cc = 12dd$

Der Parameter von `mode` ist, wie schon gesagt, aus: `bw`, `color` oder `ham`. Die Schlüsselwörter `red`, `green`, `blue` und `gamma` verlangen als Parameter eine reelle Zahl. Standardmäßig sind folgende Werte vordefiniert: `red=1.0`, `green=1.0`, `blue=1.0` und `gamma=0.5`.

Hier ein paar Beispiele für einen \special-Strings:

- \special{`ifffile=dh0:pict/graphic1`}
  Hier wird das IFF-File "graphic1" aus dem Directory "dh0:pict" an der aktuellen Textposition eingebunden.
- \special{`ifffile=graphic2 hoffset=1.5in`}
  Diesmal wird das Bild aus dem aktuellen Directory des SpecialHost Programms geholt und um 1,5 Zoll nach rechts verschoben.
- \special{`ifffile=graphic3 voffset=-1cm hsize=2in vsize=1in`}
  Das Bild "graphic3" wird bei diesem Beispiel um einen Zentimeter nach oben verschoben und auf die Größe 2 mal 1 Zoll vergrößert oder verkleinert.

Bei der Verwendung der Schlüsselwörter `hsize` und `vsize` muß übrigens nicht mehr auf die richtige Einstellung der "Base DPI" geachtet werden, da dann das Bild auf jeden Fall auf die geforderte Größe gebracht wird. Die Orginal- Größe des Bildes wird dann ignoriert.

Mit Ausnahme der TPIC Specials, die im Anschluß erläutert werden, muß derzeit immer ein IFF-File angegeben werden.

8. TPIC Unterstützung

SpecialHost versteht auch die Befehle der TPIC Sprache. In dieser Sprache gibt es Kommandos zum Zeichnen von Linien, Kreisen, Splines usw. Die Sprache TPIC ist genormt und wird auch von einigen anderen Programmen unterstützt. Zum Beipiel von dem Makropaket EEPIC, das Bilder im TPIC Format erzeugt.

Die TPIC Specials werden etwas anders verarbeitet, als die bisher besprochenen Special-Strings. Die TPIC Kommandos werden von den Treibern vorbereitet und SpecialHost bekommt nur noch komplette TPIC Kommandos übergeben. Dadurch wird etwas Kommunikation zwischen den Programmen eingespart.

9. FIG-Bilder in TEX-Texte

Das "FIG" Programm ist ein vektororientiertes Graphicprogramm, mit dem man Zeichnungen erstellen kann und im "fig" Format abspeichern kann. Diese Zeichnungen kann man nun auch auf sehr einfache Art und Weise in einen TEX-Text einbinden. Dazu muß man lediglich ein paar Schritte ausführen.

Als erstes wird das Bild mit dem `FIG` Programm zeichnen und abspeichern. Z.B. `pict.fig`. Nun muß man das Bild in ein anderes Format konvertieren. Dazu wird das Programm `Fig2Dev` verwendet. Der Aufruf:

```
Fig2Dev >pict.eepic -L eepic pict.fig
```

Dies konvertiert das Programm in das "EEPIC" Format. EEPIC ist ein Makropacket das zusammen mit LATEX verwendet werden kann. Nun schreibt man ein kleines TEX-File (z.B. `pict.tex`):

```
\documentstyle[11pt,epic,eepic]{article}
\begin{document}
\input pict.eepic
\end{document}
```

Dieses File muß nun mit LATEX übersetzt werden. Also z.B. mit:

```
virtex &lplain pict
```

Das entstehende DVI-File kann man wie üblich mit ShowDVI ansehen und mit DVIprint ausdrucken. Allerdings muß dabei natürlich SpecialHost im Hintergrund laufen, da das EEPIC Format TPIC Specials verwendet.

10. Die Software-Schnittstelle zu den Treibern

Hier ein kurzer Einblick in die Arbeitsweise des SpecialHost Programms.

10.1. Die Kommunikation über Ports

Die Kommunikation der Treiber mit dem SpecialHost-Programm läuft mittels eines Message-Ports ab. Sobald SpecialHost gestartet wird, öffnet es einen Port mit dem Namen "special_dvi". An diesem Port wartet das Programm auf Meldungen der Treiber.

Wenn ein Treiber im DVI-File auf einen \secial-String stößt, testet er zuerst, ob der Port des SpecialHost-Programms existiert. Wenn nicht, ignoriert er den String. Falls doch, so erzeugt der Treiber einen Port mit dem Namen "special_reply" und richtet ihn als Reply-Port ein.

Über den Port "special_dvi" schickt er dem SpecialHost-Programm eine Meldung, in der die derzeitige Auflösung des Druckers (oder Bildschirms) sowie der \special-String steht.

SpecialHost wertet diese Message aus und fügt die `special_map` Struktur hinzu. Diese enthält die Größe des Bildes, den Offset und auch den Namen des Bitmap-Files oder einen Pointer auf die Bitmap. Diese Meldung schickt es per "ReplyMsg" an den Treiber zurück.

Sobald der Treiber alle Informationen der "special_map" ausgewertet hat, schickt er dem SpecialHost-Programm noch eine Meldung, damit dieses weiß, daß der Speicherplatz der "special_map" wieder freigegeben werden kann. Diese Meldung wird von SpecialHost sofort bestätigt.

Danach löscht der Treiber seinen Reply-Port.

Während diese Kommunikation abläuft – zwischen dem Versenden der ersten Meldung durch dem Treiber und dem Reply der zweiten Message durch das SpecialHost Programm – dürfen die beiden Partner der Unterhaltung auf keinen Fall unterbrochen werden. Sobald einer der beiden seine Kommunikation einstellt, entsteht ein Deadlock. Das andere Programm wartet bis zum nächsten Reset. Da kein Timeout implementiert ist, hilft in dem Fall nur ein Reboot weiter. Wenn man das SpecialHost Programm ändert, dann muß man darauf achten, daß während der Kommunikation das Programm nie abgebrochen werden kann. Auch bei einem Fehlerfall muß die Kommunikation immer zu Ende geführt werden.

10.2. Das Bitmap-File-Format

Um nicht immer wieder das zeitaufwendige Vergrößern oder Verkleinern der Bilder durchführen zu müssen, gibt es die Möglichkeit, die schon auf die richtige Größe gebrachten Bilder abzuspeichern. Dies kann man mittels der zwei `File`-Gadgets einstellen.

Damit man die Bilder immer in den verschiedenen Auflösungen parat hat, wird beim Abspeichern die jeweilige Auflösung des Druckers (beziehungsweise des Bildschirmes) angehängt. Speichert man beim Previewer mit einer Auflösung von 91dpi das Bild `testbild` ab, so wird eine Datei `testbild.91x91` erzeugt.

Diese Datei hat ein ganz besonderes Format.
Die ersten vier Bytes des Files sind das 'Magic Word' "SPEC". An diesem erkennt man, daß es eine Bitmap-Datei des SpecialHost Programms ist.

Die nächsten vier Bytes ergeben sich, wenn man den Special-String, der zu diesem Bild gehört, Buchstabe für Buchstabe aufaddiert. Dies hat den Sinn, daß man daran erkennt, ob sich der Special-String geändert hat, seitdem man das File erzeugt hat. Ist dies der Fall, so wird das Bild neu berechnet und abgespeichert.

Nun folgen zwei Langwörter, die die Breite und Höhe des Bildes in Pixel angeben.

Zum Schluß kommt die Bitmap, Zeile für Zeile. Die Länge einer Zeile wird bis auf 16 Bit aufgerundet. Für eine Zeile mit 100 Bit benötigt man also sieben 16-Bit-Wörter.

Noch ein weiteres Datum wird indirekt mit dem File gespeichert. Das Erstellungsdatum. Es wird dazu verwendet zu überprüfen, ob das IFF-Bild jüngeren Datums ist als das Bitmap-File. Ist dies der Fall, so muß natürlich die Bitmap neu berechnet werden, da sich das IFF-Bild allem Anschein nach geändert hat.

Durch diesen Datumsvergleich und durch das Überprüfen des \special-Strings wird sichergestellt, daß kein veraltetes Bitmap-File verwendet wird. Man kann also ohne weiteres den Zeichenmodus `Transfer picture in a file` als Standard verwenden. So muß ein Bild für eine Auflösung nur einmal berechnet werden, und steht dann immer als File zur Verfügung. Das einzige, daß dabei zum Problem werden kann, ist der Speicherplatz, da die Bitmap-Files ungepackt abgespeichert werden.

10.3. Der Source

Um sich in dem Source schnell zurechtzufinden, wird hier zu jedem File eine kurze Beschreibung gemacht.

special.c	Source der Kommunikationsfunktionen für die Treiber.
special.h	Definitionen der Strukturen, die für die Kommunikation gebraucht werden.
flextr.c	Modul zum Vergrößern oder Verkleinern von einfarbigen Bitmaps. Algorithmus aus dem FBM-Packet entnommen.
specialhost.c	Modul mit allen Funktionen, die zur Kommunikation mit den Treibern gebraucht werden. Dazu noch Funktionen für die Auswertung des \special-Strings.
specialhost.h	Defines für Fehlermeldungen und Draw-Modus. Dazu noch Prototypen der modulübergreifenden Funktionen.
specialparse.c	Modul, das die Auswertung des \special-Strings übernimmt. Dieses Modul hat seinen Ursprung im File “dospecial.c” des PostScript-Treibers “dvips” von Thomas Rokicki.
specialparse.h	Definitionen von Strukturen für das File “specialparse.c”.
specialwin.c	In diesem Modul wird das Fenster- und Gadget-Handling angewickelt. Hier ist auch die “main”-Funktion mit der Hauptschleife, in der auf alle möglichen Signale gewartet wird.
specialwin.h	Hier werden die Gadgets definiert.
iff.h	Headerfile von Christian A. Webers “iff.library”. Allerdings habe ich es so geändert, daß nun auch mit dem Aztec-C Compiler die Pragmas verwendet werden.

11. Was wäre alles noch zu tun

Neben ein paar Unschönheiten bei der Bedienung des Programms, fehlt vor allem eine bessere Bildverarbeitung. Zum Beispiel Postscript-Support wäre sicherlich schön. Und mit der “`post.library`” ist es auch sicherlich nicht übermäßig schwierig, dies einzubauen.

Aber aus diesem Grund gebe ich den vollständigen Source des Programms frei. Vielleicht findet sich jemand, der sich in der Bildverarbeitung auskennt und ein wenig Zeit und Lust hat, sein Wissen in das Programm einzubringen.

12. Benötigte Soft- und Hardware

Zur Hardware ist nur zu sagen: ein Amiga, und zwar nach Möglichkeit einer mit etwas mehr Speicher. Sonst kann es bei großen Bildern und hoher Auflösung zu Speichermangel kommen.

Als Software werden die TEX-Treiber ShowDVI und DVIprint benötigt. Dazu noch die "iff.library" von Christian A. Weber.

13. Bekannte Fehler

Das File "specialwin.c" läßt sich nicht mit dem Optimierer von Lattice übersetzen. Dagegen erzeugt der Aztec-C-Compiler Version 5.0a inkorrekten Code für das Modul "specialparse.c".

Manchmal kommt es vor, daß Bilder bei der Option `transfer in file` nicht angezeigt werden. Dies kann man umgehen, in dem man auf `transfer in memory` umschaltet. Bis zur Version 1.0 wird der Fehler aber sicherlich noch behoben werden.