Paint Shop Pro (PSP)
File Format Specification

November, 1998

Paint Shop Pro (PSP) File Format Version 3.0
Paint Shop Pro Application Version 5.0
File Format Documentation Version 3.0

Copyright © 1998 Jasc Software, Inc..
All rights reserved.

The information contained within this document is confidential and is subject to change without notice

By downloading and / or using these Specifications, user certifies that they have read and agreed to the following disclaimers and warranties.

The Paint Shop Pro File Format Specification (the "Specification") is copyright 1998 by Jasc Software, Inc. Jasc grants you a nonexclusive license to use the Specification for the sole purposes of developing software products(s) incorporating the Specification. You are also granted the right to identify your software product(s) as incorporating the Paint Shop Pro Format (PSP) provided that your software in incorporating the Specification complies with the terms, definitions, constraints and specifications contained in the Specification and subject to the following:

DISCLAIMER OF WARRANTIES. THE SPECIFICATION IS PROVIDED "AS IS." JASC DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.

You are solely responsible for the selection, use, efficiency and suitability of the Specification for your software products.

OTHER WARRANTIES EXCLUDED. JASC SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, EXEMPLARY, PUNITIVE OR INCIDENTAL DAMAGES ARISING FROM ANY CAUSE EVEN IF JASC HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. CERTAIN JURISDICTIONS DO NOT PERMIT THE LIMITATION OR EXCLUSION OF INCIDENTAL DAMAGES, SO THIS LIMITATION MAY NOT APPLY TO YOU.

IN NO EVENT WILL JASC BE LIABLE FOR ANY AMOUNT GREATER THAN WHAT YOU ACTUALLY PAID FOR THE SPECIFICATION. Should any warranties be found to exist, such warranties shall be limited in duration to ninety (90) days following the date you receive the Specification.

INDEMNIFICATION. By your inclusion of the Paint Shop Pro File Format in your software product(s) you agree to indemnify and hold Jasc Software, Inc. harmless from any and all claims of any kind or nature made by any of your customers with respect to your software product(s).

EXPORT LAWS. You agree that you and your customers will not export your software or Specification except in compliance with the laws and regulations of the United States.

US GOVERNMENT RESTRICTED RIGHTS. The Specification and any accompanying materials are provided with Restricted Rights. Use, duplication or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of The Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, or subparagraphs (c)(1) and (2) of the Commercial Computer Software - Restricted Rights at 48 CFR 52.227-19, as applicable. Contractor/manufacturer is Jasc Software, Inc., PO Box 44997, Eden Prairie MN 55344.

Jasc reserves the right to amend, modify, change, revoke or withdraw the Specification at any time and from time to time. Jasc shall have no obligation to support or maintain the Specification.

Table of Contents

The information contained within this document is confidential and is subject to change without notice

1 Version History

Date	Author	Description
April 20, 1998	JMS	Initial draft, submitted for review.
April 23, 1998	JMS	Minor changes.
April 29, 1998	JMS	Minor changes (changed Layer flags, Layer transparency flags to Boolean Layer visible flag and Transparency protected flag).
August 7, 1998	JMS	Made additions for storing Picture Tube control information in PSP files.
November 16,1998	JCO	Added additional Copyright information, release notes, disclaimers, comments, and cleaned for public release.
November 23, 1998	JMS	Added restrictions for readers and writers. Changes based on latest review.
November 30, 1998	JCO	Restrictions clarified Requests email address added.
December 1, 1998	JMS	Added descriptions of compression types. Added descriptions of non-color channels.

The information contained within this document is confidential and is subject to change without notice

Introduction

The Paint Shop Pro (PSP) file format is an extremely rich graphics file format that is capable of describing a multi-layered image document in minute detail. The format facilitates the retention of a great deal of information, including layer bitmaps, layer masks, layer grouping information, a sample thumbnail, image dimension, image resolution, creator name, copyright owner, image description, masks, selections, alpha channels, etc.

The PSP format supports storing an image document in an uncompressed format or, to reduce the size of the file, in either Run Length Encoding (RLE) or LZ77 compressed format. As each of these is a lossless storage method, the PSP format always maintains the original quality of stored image documents.

Finally, the PSP format supports storing image bitmaps at a variety of bit depths, including 24 bit color (16.7 million colors), 8 bit color (256 color), 8 bit greyscale (256 greys), 4 bit color (16 colors) and 1 bit color (2 colors).

When developing an image document in Paint Shop Pro 5, the full-service graphics editing tool from Jasc Software, Inc, the PSP file format is the format of choice because an image document can be maintained in its entirety (saving image documents to less versatile formats results in loss of some image document data).

NOTICE: This document describes the PSP file format, not necessarily the technology or code required to fully read, interpret, or display all aspects of the data that may be contained within the file.

2 Resources

For more information about Jasc Software, Inc, Paint Shop Pro or the PSP file format, please visit the Jasc Software web site and the Paint Shop Pro newsgroup (both listed below). The latest published version of this document is available at the Jasc Software web site. If you have any questions or comments about this document or about the PSP file format, please send them to the Jasc Software PSP File Format email address, listed below.

The Jasc Software web site:

http://www.jasc.com

The Paint Shop Pro newsgroup:

comp.graphics.apps.paint-shop-pro

The Jasc Software PSP File Format email address:

PSPFileFormat@jasc.com

3 Format Overview

The Paint Shop Pro (PSP) file format is a block-oriented format that consists of a file header with a sequence of up to nine main blocks. The nine main blocks are the General Image Attributes Block, the Extended Data Block (optional), the Creator Data Block (optional), the Thumbnail Block (optional), the Color Palette Block (for paletted images), the Layer Bank Block, the Selection Block (optional) and the Alpha Channel Bank Block (optional). Each of these main blocks contains information about the image document and some of these blocks contain sub-blocks. See subsequent sections of this document for more detailed information about each of these main blocks.

The following table contains a high-level view of the PSP file format that includes only the header and the top-level blocks.

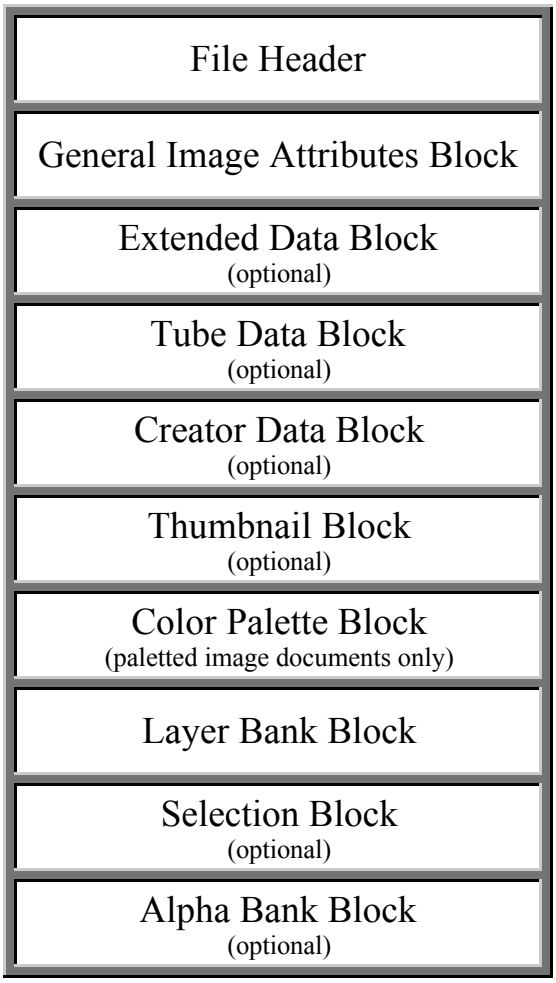

Table 1 : PSP Format Overview

Restrictions for PSP format readers:

- PSP format readers must be able to read and successfully process the PSP File Header, the General Image Attributes Block, the Color Palette Block (if present) and the Layer Bank Block. PSP readers may ignore all other main blocks.

- If a PSP reader encounters any unrecognized block (whether documented or not), the reader ***must*** ignore (skip over) that block and continue processing the rest of the file.

Restrictions for PSP format writers (Format Version 3.0):

- PSP format writers ***must*** write the PSP File Header, the General Image Attributes Block, the Color Palette Block (if appropriate) and the Layers Bank Block ***exactly*** as described in this specification. Additionally, PSP format writers may write the other blocks described in this document when necessary.
- PSP format writers ***must not*** write any blocks that are not documented in this specification, or otherwise stray from this specification. Additional block types or specification changes may only be created by Jasc Software. Requests for additional block types and other file format enhancements can be sent to PSPFileFormat@jasc.com.
- Color data of multi-layer image documents must be stored in either 24-bit truecolor format, or in 8-bit greyscale format.
- Color data of single-layer image documents can be stored in 1, 4 or 8-bit paletted format, in 24-bit truecolor format, or in 8-bit greyscale format.
- PSP format writers must not write an image document with more than 64 layers.

4 Common Structures and Definitions

This section contains information concerning structures and definitions that are used throughout the remainder of this document. They are presented here to give the user a hint as to what to expect in subsequent sections of the document, and to provide references that can be used in subsequent sections of the document to avoid too much repetition (though repetition is used when it provides needed clarification).

4.1 Parlance

With any documentation, it is important to ensure that the author and the audience have a common parlance. With that in mind, the following glossary describes the usage, throughout this document, of various words that otherwise might be ambiguous.

Word	Definition/Usage
Block	As described above, the PSP format is a block-oriented format. The term "block" will be used to refer to a part of a PSP file that consists of a Block Header (see section 5.4) ***and*** all the data following the header that is associated with the header. Since the Block Header indicates the size of the block, it is quickly apparent how much of the data immediately following the block header is associated with it.
Chunk	The term chunk will be used to represent an "interesting" piece of data within a PSP file that does not have a block header. Throughout the document, there is a need to refer to a piece of data within a block that is not a sub-block. The term "chunk" is used in these cases.
Main block	The PSP file format is a hierarchical format, with some blocks that are fully contained within other blocks. A main block is any block in a PSP file that is not contained within another block. The main blocks are illustrated in the table in section 4.
Sub-Block	A sub-block is a block that is fully contained within another block.

Table 2: Glossary of Terms

4.2 Type Definitions

The following table contains definitions for simple data types used throughout this document. The PSP file format is always written in Intel® (or little-endian) byte order.

Type	Description
BYTE	An 8-bit unsigned integer value.
B_ARRAY	An array of 8-bit unsigned integer values (i.e., an array of BYTE). The length of the array is either fixed, or can be determined from the discussion.
C_ARRAY	An array of 8-bit signed integer values (i.e., an array of char). The length of the array is either fixed, or can be determined from the discussion.
double	A 64-bit floating point value.

DWORD	A 32-bit unsigned integer value.
long	A 32-bit signed integer value.
RECT	An array of 4 longs that describe a rectangle (the first two longs are the x-coordinate and y-coordinate (respectively) of the upper-left corner of the rectangle, the third and fourth longs are the x-coordinate and y-coordinate (respectively) of the lower-right corner of the rectangle.
RGBQUAD	An array of 4 BYTEs representing a color palette entry, where the first BYTE is the entry's red component, the second BYTE is the entry's green component and the third BYTE is the entry's blue component and the fourth BYTE is always 0 (reserved).
WORD	A 16-bit unsigned integer value.

Table 3: Common type definitions

4.3 Limiting Field Values

The following enumerations are used throughout the document to limit the allowed values of various fields.

As an example, the Block Identifier field in a Block Header is a DWORD, but, as detailed in the Block Identifier field description, the DWORD must be in the range covered by the enumeration PSPBlockID.

```
/* Block identifiers.
 */
enum PSPBlockID {

      PSP_IMAGE_BLOCK = 0,    // General Image Attributes Block (main)
      PSP_CREATOR_BLOCK,      // Creator Data Block (main)
      PSP_COLOR_BLOCK,        // Color Palette Block (main and sub)
      PSP_LAYER_START_BLOCK,  // Layer Bank Block (main)
      PSP_LAYER_BLOCK,        // Layer Block (sub)
      PSP_CHANNEL_BLOCK,      // Channel Block (sub)
      PSP_SELECTION_BLOCK,    // Selection Block (main)
      PSP_ALPHA_BANK_BLOCK,   // Alpha Bank Block (main)
      PSP_ALPHA_CHANNEL_BLOCK,// Alpha Channel Block (sub)
      PSP_THUMBNAIL_BLOCK,    // Thumbnail Block (main)
      PSP_EXTENDED_DATA_BLOCK,// Extended Data Block (main)
PSP_TUBE_BLOCK,         // Picture Tube Data Block (main)
};

/* Bitmap type.
 */
enum PSPDIBType {

      PSP_DIB_IMAGE = 0,      // Layer color bitmap
      PSP_DIB_TRANS_MASK,     // Layer transparency mask bitmap
      PSP_DIB_USER_MASK,      // Layer user mask bitmap
      PSP_DIB_SELECTION,      // Selection mask bitmap
      PSP_DIB_ALPHA_MASK,     // Alpha channel mask bitmap
      PSP_DIB_THUMBNAIL       // Thumbnail bitmap
```

```
};

/* Channel types.
 */
enum PSPChannelType {

      PSP_CHANNEL_COMPOSITE = 0,    // Channel of single channel bitmap
      PSP_CHANNEL_RED,              // Red channel of 24 bit bitmap
      PSP_CHANNEL_GREEN,            // Green channel of 24 bit bitmap
      PSP_CHANNEL_BLUE,             // Blue channel of 24 bit bitmap

};

/* Possible metrics used to measure resolution.
 */
enum PSP_METRIC {

      PSP_METRIC_UNDEFINED = 0,     // Metric unknown
      PSP_METRIC_INCH,              // Resolution is in inches
      PSP_METRIC_CM,                // Resolution is in centimeters

};


/* Possible types of compression.
 */
enum PSPCompression {

      PSP_COMP_NONE = 0,            // No compression
      PSP_COMP_RLE,                 // RLE compression
      PSP_COMP_LZ77                 // LZ77 compression

};

/* Picture tube placement mode.
 */
enum TubePlacementMode {
      tpmRandom,        // Place tube images in random intervals
      tpmConstant,      // Place tube images in constant intervals
};

/* Picture tube selection mode.
 */
enum TubeSelectionMode {
      tsmRandom,        // Randomly select the next image in
                        // tube to display
      tsmIncremental,   // Select each tube image in turn
      tsmAngular,       // Select image based on cursor direction
      tsmPressure,      // Select image based on pressure
                        // (from pressure-sensitive pad)
      tsmVelocity,      // Select image based on cursor speed
};
```

```
/* Extended data field types.
 */
enum PSPExtendedDataID {

      PSP_XDATA_TRNS_INDEX = 0,           // Transparency index field
};

/* Creator field types.
 */
enum PSPCreatorFieldID {

      PSP_CRTR_FLD_TITLE = 0,       // Image document title field
      PSP_CRTR_FLD_CRT_DATE,        // Creation date field
      PSP_CRTR_FLD_MOD_DATE,        // Modification date field
      PSP_CRTR_FLD_ARTIST,          // Artist name field
      PSP_CRTR_FLD_CPYRGHT,         // Copyright holder name field
      PSP_CRTR_FLD_DESC,            // Image document description field
      PSP_CRTR_FLD_APP_ID,          // Creating app id field
      PSP_CRTR_FLD_APP_VER,         // Creating app version field

};

/* Creator application identifiers.
 */
enum PSPCreatorAppID {

      PSP_CREATOR_APP_UNKNOWN = 0,        // Creator application unknown
      PSP_CREATOR_APP_PAINT_SHOP_PRO,     // Creator is Paint Shop Pro

};

/* Layer types.
 */
enum PSPLayerType {

      PSP_LAYER_NORMAL = 0,               // Normal layer
      PSP_LAYER_FLOATING_SELECTION,       // Floating selection layer

};

/* Truth values.
 */
enum PSP_BOOLEAN {

      FALSE = 0,
      TRUE,

};
```


The information contained within this document is confidential and is subject to change without notice

4.4 Block Header

The block structure utilized by the PSP file format is quite simple. Each main block consists of a block header that indicates size and type information for the block, followed by the block data. The content of the block data depends on the block's type.

Moreover, the block structure used in the PSP format is hierarchical, so a block may consist of one or more sub-blocks. The sub-blocks take the same format as the main blocks (block header + block data).

As illustrated in the following table, the Block Header identifies the block's type, the length of the first "interesting piece of data" and the total block length.

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK" followed by a zero byte).
WORD	2 bytes	Block identifier	Identifies the type of the block (one of PSPBlockID).
DWORD	4 bytes	Initial data chunk length	Length of first "interesting piece of data" (this "interesting piece of data" may be the header of a sub-block or simply a data chunk containing relevant information).
DWORD	4 bytes	Total block length	Total length of block (including all sub-blocks), not including the header. Use this length to skip past unknown or unwanted blocks.

Table 3: Block Header

4.5 Color Palette Sub-Block

Color Palette Sub-Blocks are used throughout the PSP file format to store palettes associated with paletted bitmaps. For example, when the image document thumbnail is stored as a paletted bitmap, there is a Color Palette Sub-Block in the Thumbnail Block to store the thumbnail's palette.

As illustrated in the following table, the Color Palette Sub-Block consists of a Color Palette Block Header, a Color Palette Information Chunk, and a Color Palette Entries Chunk.

Color Palette Block Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_COLOR_BLOCK.
DWORD	4 bytes	Initial data chunk length	Length of following Color Palette Information Chunk.

DWORD	4 bytes	Total block length	Length of complete Color Palette Sub-Block, including Color Palette Information Chunk and Color Palette Entries Chunk.

The information contained within this document is confidential and is subject to change without notice

Color Palette Information Chunk:

DWORD	4 bytes	Palette entry count	Number of entries in the palette (for the purpose of determining the number of fields in the Color Palette Entries Chunk, let's call this i).

Color Palette Entries Chunk

RGBQUAD	4 bytes	Palette entry 1	The 1st palette entry.
…	…	…	…
RGBQUAD	4 bytes	Palette entry i	The ith palette entry

Table 4: Color Palette Sub-Block

4.6 Channel Sub-Block and channel compression

Channel Sub-Blocks are used throughout the PSP file format to store image document channel data. Each color channel is stored separately in the PSP format. For example, when the image document thumbnail is stored as an 24-bit bitmap, there are 3 Channel Sub-Blocks (a red Channel Sub-Block, a green Channel Sub-Block and a blue Channel Sub-Block) in the Thumbnail Block. However, when the image document thumbnail is stored as a paletted bitmap, there will only be one Channel Sub-Block in the Thumbnail Block (because a paletted bitmap contains only one channel).

As illustrated in the table below, the Channel Sub-Block consists of a Channel Block Header, a Channel Information Chunk and a Channel Content Chunk.

A Word About Compression

All channel data in a PSP file can be stored compressed in the LZ77 format, compressed in the Run Length Encoding (RLE) format, or uncompressed (in a "raw" format). This channel data can include thumbnail color channels, layer color channels, layer transparency mask channels, layer user mask channels, alpha channels or selection channels.

The thumbnail color channels are stored in the compression format indicated in the "Thumbnail compression type" field of the "Thumbnail Attributes Chunk," as described in section 5.7 of this document. All other channels are stored in the compression format indicated in the "Compression type" field of the "General Image Attributes Chunk," as described in section 5.3 of this document.

The LZ77 compression scheme used in the PSP file format is the LZ77 variant used in the PNG file format. There are a couple of important distinctions between the way this LZ77 variant is utilized in the PNG file format and the way it is utilized in the PSP file format. First, the PNG file format requires that data be sent through one of several filters before it can be compressed, whereas the PSP file format has no such filter. Second, the PNG file format mandates support for a discontiguous compression stream, whereas the PSP file format allows only a contiguous compression stream.

The Run Length Encoding (RLE) scheme used in the PSP file format is a relatively straightforward RLE variant. RLE is a simple compression method that strives to take advantage of contiguous runs of the same data. In the variant used in the PSP file format, the first byte encountered in the compressed stream is a byte that represents a "run count". While decompressing, if this run count is greater than 128, then 128 should be subtracted from the "run count." This calculated run count (run count – 128) indicates the number of times the next byte in the compressed stream should be repeated in the decompressed stream. If the "run count" read from compressed stream is less than 128, then it indicates the number of following

bytes (in the compressed stream) that should be copied as-is from the compressed stream to the decompressed stream. The compressed stream is comprised of these RLE "packets," each containing a "run count" and data associated with that run count.

In all compression methods used in the PSP format, each channel is a single compression stream which, when decompressed, represents bitmap data stored from left to right and from top to bottom. Each scanline in the image data is stored on a 4 byte boundary.

All channels that represent something other than color (including layer transparency mask channels, layer user mask channels, alpha channels and selection channels), when decompressed, are 8-bit greyscale bitmaps.

Channel Block Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_CHANNEL_BLOCK.
DWORD	4 bytes	Initial data chunk length	Length of following Channel Information Chunk.
DWORD	4 bytes	Total block length	Length of complete Color Palette Sub-Block, including Channel Information Chunk and Channel Content Chunk.

Channel Information Chunk:

DWORD	4 bytes	Compressed channel length	Size of the channel in compressed form (for the purpose of determining the length of the last field in this block, let's call this size j).
DWORD	4 bytes	Uncompressed channel length	Size of the channel in uncompressed form.
WORD	2 bytes	Bitmap type	Type of bitmap for which this channel is intended (one of `PSPDIBType`).
WORD	2 bytes	Channel type	Type of channel (one of `PSPChannelType`).

Channel Content Chunk:

B_ARRAY	j bytes	Channel content	Content of channel. This field contains (possibly compressed) data that defines the channel. Other blocks define whether the channel is compressed, and define the size of the bitmap associated with this channel (height and width).

Table 5: Channel Sub-Block

5 Paint Shop Pro (PSP) File Format Description

This section details the PSP file format.

5.1 Constraints

This section describes miscellaneous constraints that writers of the PSP file format must follow.

5.1.1 Main Block Ordering

With regard to the order of the nine main blocks, there are two hard and fast rules. First, the General Image Attributes block must come immediately after the File Header. Second, the Thumbnail Block, if present, must come before the Layer Bank Block.

The order of other blocks is not mandated. However, the order in which the main blocks are illustrated in Table 1 of section 4:Format Overview is the preferred order. This order allows programs to quickly obtain information about an image document (information such as bit depth, compression type, resolution, etc) without having to read layer information (such as color bitmap and mask data).

5.1.2 One Paletted Layer

The color bitmaps of images contained in multi-layered image documents must be either 24-bit bitmaps or 8-bit greyscale bitmaps. Color bitmaps of images contained in single-layered image documents can be 1-bit, 4-bit, 8-bit color, 8-bit greyscale or 24-bit bitmaps.

5.2 PSP File Header

Every PSP file begins with the following header, which should be used for initial validation. Any file that does not begin with this header should be considered an invalid PSP file.

As illustrated in the following table, the PSP file header consists of the PSP file signature, followed by a version number.

Type	Length	Name	Description
B_ARRAY	32 bytes	PSP file signature	Always "`50 61 69 6E 74 20 53 68 6F 70 20 50 72 6F 20 49 6D 61 67 65 20 46 69 6C 65 0A 1A 00 00 00 00`" (i.e., the string "Paint Shop Pro Image File\n\x1a", padded with zeroes to 32 bytes).
DWORD	2 bytes	PSP file major version number	Currently always 3.
DWORD	2 bytes	PSP file minor version	Currently always 0.

Table 6: PSP File Header

5.3 The General Image Attributes Block

The General Image Attributes Block, a required block that contains various information about the stored image document, immediately follows the PSP file header. Much of this information is needed to read other parts of the file (for example, the compression type is required to decode any channel blocks (with the

exception of the Thumbnail Block channel(s)) and the layer count is required to know how many layers need to be read from the Layers Block).

As illustrated in the following table, the General Image Attributes Block consists of the General Image Attributes Header and the General Image Attributes Chunk.

General Image Attributes Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "7E 42 4B 00" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_IMAGE_BLOCK.
DWORD	4 bytes	Initial data chunk length	Length of following General Image Attributes Chunk.
DWORD	4 bytes	Total block length	Length of complete General Image Attributes Block (which includes only the following General Image Attributes Chunk).

General Image Attributes Chunk:

long	4 bytes	Image width	Width of the image.
long	4 bytes	Image height	Height of the image.
double	8 bytes	Resolution value	Number of pixels per metric.
BYTE	1 byte	Resolution metric	Metric used for resolution (one of PSP_METRIC).
WORD	2 bytes	Compression type	Type of compression used to compress all image document channels, except those in the image thumbnail, which has its own compression type field (one of PSPCompression).
WORD	2 bytes	Bit depth	The bit depth of the color bitmap in each Layer of the image document (must be 1, 4, 8 or 24).
WORD	2 byte	Plane count	Number of planes in each layer of the image document (this value must be 1).
DWORD	4 bytes	Color count	Number of colors in each layer of the image document ($2^{\text{Bit depth}}$).
BYTE	1 byte	Greyscale flag	Indicates whether the color bitmap in each layer of image document is a greyscale (0 = not greyscale, 1 = greyscale).
DWORD	4 byte	Total image size	Sum of the sizes of all layer color bitmaps.
long	4 bytes	Active layer	Identifies the layer that was active when the image document was saved.

WORD	2 bytes	Layer count	Number of layers in the document.

Table 7: General Image Attributes Block

5.4 The Extended Data Block

The Extended Data Block is an optional block that contains miscellaneous information about the image document. This information is organized in the PSP field format, which consists of a series of keyword-value pairs.

Using keyword-value pairs in the Extended Data Block facilitates the storage of new, currently undefined, image document attributes. Applications can quickly be modified to make use of the new keywords, while old applications that do not know about the new keywords will simply ignore them (therefore, unknown keywords encountered in the Extended Data Block should be ignored). If the extended data block is present, it must contain at least one extended data field.

As illustrated in the following table, the Extended Data Block consists of a block header, followed by one or more Extended Data Fields (each Extended Data Field consists of a keyword chunk and a value chunk).

Extended Data Block Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_EXTENDED_DATA_BLOCK.
DWORD	4 bytes	Initial data chunk length	Undefined.
DWORD	4 bytes	Total block length	Length of following Extended Data Block (includes all following extended data fields – each of which consists of an Extended Data Keyword Chunk and an Extended Data Value Chunk).

Extended Data Keyword Chunk (first extended data field):

B_ARRAY	4 bytes	Field signature	Always "`7E 46 4C 00`" (i.e.,"~FL", followed by a zero byte).
WORD	2 bytes	Field keyword	Keyword identifying type of extended data that is contained in the Extended Data Value Chunk (must be one of PSPExtendedDataID).
DWORD	4 bytes	Extended Data Value Chunk length	Length of following Extended Data Value Chunk (lets call this k).

Extended Data Value Chunk (first extended data field):

B_ARRAY	k bytes	Field value	The contents of the Extended Data Value Chunk are dependent upon the keyword extracted from the Extended Data Keyword Chunk (currently there is only one defined keyword). Unknown keywords should be ignored:	
			Field keyword:	Field value description:
			PSP_XDATA_TRNS_INDEX	A WORD that identifies the palette index of the transparent color (this field is not valid for image documents whose color bitmaps are non-paletted, and should be ignored in this case).

• • •

Extended Data Keyword Chunk (last extended data field):

B_ARRAY	4 bytes	Field identifier	Always "`7E 46 4C 00`" (i.e.,"~FL", followed by a zero byte).
WORD	2 bytes	Field keyword	Keyword identifying type of extended data that is contained in the Extended Data Value Chunk (must be one of PSPExtendedDataID).
DWORD	4 bytes	Data length	Length of following Extended Data Value Chunk (lets call this k).

Extended Data Value Chunk (last extended data field):

B_ARRAY	k bytes	Field value	The contents of the Extended Data Value Chunk are dependent upon the keyword extracted from the previous Extended Data Keyword Chunk (currently there is only one defined keyword):	
			Field keyword:	Field value description:
			PSP_XDATA_TRNS_INDEX	A 2 byte WORD that identifies the palette index of the transparent color (this field is not valid for image documents whose color bitmaps are non-paletted, and should be ignored in this case).

Table 4: Extended Data Block

5.5 The Tube Data Block

The Tube Data Block is an optional block that contains Paint Shop Pro Picture Tube control data. This block controls how Picture Tube files (which are PSP files) act.

Tube Data Block Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "7E 42 4B 00" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_TUBE_BLOCK.
DWORD	4 bytes	Initial data chunk length	Length of following Tube Data chunk.
DWORD	4 bytes	Total block length	Length of following Tube Data chunk.

Tube Data Chunk:

WORD	2 bytes	Block version	Version of following Tube Data Block.
C_ARRAY	513 bytes	Tube name	User friendly name of the tube.
long	4 bytes	Tube step size	Tube step size.
long	4 byte	Tube column count	Number of columns in tube file.
long	4 bytes	Tube row count	Number of rows in tub.
long	4 bytes	Tube cell count	Number of cells in tube.
long	4 bytes	Tube placement mode	Controls tube placement mode (one of TubePlacementMode).
long	4 bytes	Tube selection mode	Controls tube selection mode (one of TubeSelectionMode).

5.6 The Creator Data Block

The Creator Data Block is an optional block that contains miscellaneous information about the creator of the image document. This information is organized in the PSP field format, which consists of a series of keyword-value pairs.

Using keyword-value pairs in the Creator Data Block facilitates the storage of new, currently undefined, creator attributes. Applications can quickly be modified to make use of the new keywords, while old applications that do not know about the new keywords will simply ignore them (therefore, unknown keywords encountered in the Creator Data Block should be ignored). If the Creator Data Block is present, it must contain at least one creator field.

As illustrated in the following table, the Creator Data Block consists of a block header, followed by one or more Creator Data Fields (each Creator Data Field consists of a keyword chunk and a value chunk).

Creator Block Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "7E 42 4B 00" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_CREATOR_BLOCK.
DWORD	4 bytes	Initial data chunk length	Undefined.
DWORD	4 bytes	Total block length	Length of following Creator Block (includes all following creator data fields, each of which consists of a Creator Keyword Chunk and a Creator Value Chunk).

Creator Keyword Chunk (first creator field):

B_ARRAY	4 bytes	Field identifier	Always "7E 46 4C 00" (i.e.,"~FL", followed by a zero byte).
WORD	2 bytes	Field keyword	Keyword identifying the type of data that is contained in the Creator Value Chunk (must be one of PSPCreatorFieldID).
DWORD	4 bytes	Data length	Length of the Creator Value Chunk (lets call this n).

Creator Value Chunk (first creator field):

B_ARRAY	n bytes	Field value	The contents of the field value are dependent upon the field keyword extracted from the previous Creator Keyword Chunk:	
			Field keyword:	Field value description
			PSP_CRTR_FLD_TITLE	Title of image document (in ASCII format).
			PSP_CRTR_FLD_CRT_DATE	DWORD indicating date of image document creation (in MSVC time_t format).
			PSP_CRTR_FLD_MOD_DATE	DWORD indicating date of last image document modification (in MSVC time_t format).
			PSP_CRTR_FLD_ARTIST	Name of artist (in ASCII text).
			PSP_CRTR_FLD_CPYRGHT	Name of copyright holder (in ASCII text).
			PSP_CRTR_FLD_DESC	Description of image document (in ASCII text).
			PSP_CRTR_FLD_APP_ID	DWORD identifying application that created image document (one of PSPCreatorAppID).
			PSP_CRTR_FLD_APP_VER	DWORD identifying version of application that created image document.

• • •

Creator Keyword Chunk (last creator field):

B_ARRAY	4 bytes	Field identifier	Always "`7E 46 4C 00`" (i.e.,"~FL", followed by a zero byte).
WORD	2 bytes	Field keyword	Keyword identifying the type of data that is contained Creator Value Chunk (must be one of PSPCreatorFieldID).
DWORD	4 bytes	Data length	Length of the Creator Value Chunk (lets call this p).

Creator Value Chunk (last creator field):

B_ARRAY	p bytes	Field value	The contents of the field value are dependent upon the field keyword extracted from the previous Creator Keyword Chunk:	
			Field keyword:	Field value description:
			PSP_CRTR_FLD_TITLE	Title of text (in ASCII format).
			PSP_CRTR_FLD_CRT_DATE	DWORD indicating date of image document creation (in MSVC time_t format).
			PSP_CRTR_FLD_MOD_DATE	DWORD indicating date of last image document modification (in MSVC time_t format).
			PSP_CRTR_FLD_ARTIST	Name of artist (in ASCII text).
			PSP_CRTR_FLD_CPYRGHT	Name of copyright holder (in ASCII text).
			PSP_CRTR_FLD_DESC	Description of image document (in ASCII text).
			PSP_CRTR_FLD_APP_ID	DWORD identifying application that created image document (one of PSPCreatorAppID).
			PSP_CRTR_FLD_APP_VER	DWORD identifying version of application that created image document.

Table 5: Creator Data Block

5.7 The Thumbnail Block

The thumbnail block is an optional block that contains a miniature bitmap of the merged document.

As illustrated in the following table, the Thumbnail Block consists of a Block Header, a Thumbnail Attributes Chunk, a Thumbnail Color Palette Sub-Block (containing the thumbnail's color palette, if any) and the Thumbnail Channel Sub-Blocks.

Thumbnail Block Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_THUMBNAIL_BLOCK.
DWORD	4 bytes	Initial data chunk length	Length of following Thumbnail Attributes chunk.
DWORD	4 bytes	Total block length	Length of complete Thumbnail Block, including Thumbnail Attributes Chunk, Thumbnail Color Palette Sub-Block (if any) and all Thumbnail Channel Sub-Blocks.

Thumbnail Attributes Chunk:

long	4 bytes	Thumbnail width	Width of the thumbnail.
long	4 bytes	Thumbnail height	Height of the thumbnail.
WORD	2 bytes	Thumbnail bit depth	Number of bits used to represent each color pixel of the thumbnail (must be 1, 4, 8 or 24).
WORD	2 bytes	Thumbnail compression type	Type of compression used to compress thumbnail channels (one of PSPCompression).
WORD	2 byte	Thumbnail plane count	Number of planes in thumbnail (this value must be 1).
DWORD	4 bytes	Thumbnail color count	Number of colors in the image ($2^{\text{Bit depth}}$).
DWORD	4 bytes	Thumbnail palette entry count	Number of entries in following color palette (0 for 24 bit bitmaps).
WORD	2 bytes	Thumbnail channel count	Number of channels following (1 for paletted bitmaps, 3 for 24-bit bitmaps).

Thumbnail Color Palette Sub-Block:

Color Palette Sub-Block containing the thumbnail's color palette (this Sub-Block is not present in 24-bit thumbnails). See section 5.5.

Thumbnail Channel Sub-Blocks:

One or Three Channel Sub-Blocks defining the thumbnail bitmap. If the thumbnail is a paletted bitmap, there is one channel, if the thumbnail is a 24-bit bitmap, there are three channels. See section 5.6.

Table 6: Thumbnail Block

5.8 The Color Palette Block

The Color Palette Block contains a global color palette to be used with the color bitmaps in all layers of the document. This block is required for image documents that contain paletted bitmaps, and should not be written for image documents that contain 24-bit bitmaps (the block can be ignored in the latter case).

As illustrated in the following table, the Color Palette Block consists of a Color Palette Sub-Block.

A Color Palette Block containing the color palette for color bitmaps in all of the image document's layers. This block should not be present in image documents whose color bitmaps are 24-bit bitmaps, and can be ignored in this case. See section 5.5.

Table 7: Color Palette Block

5.9 The Layer Bank Block

The Layer Bank Block is a required block that contains data describing all the layers in the image document.

As illustrated in the following table, the Layer Bank Block consists of a Block Header, and a separate Layer Sub-Block for each layer in the image document (the number of layers in the document is indicated in the General Image Attributes Block). Each Layer Sub-Block consists of a Layer Sub-Block Header, a Layer Information Sub-Block and one or more Channel Sub-Blocks.

Layer Bank Block Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_LAYER_START_BLOCK.
DWORD	4 bytes	Initial data chunk length	Always 0.
DWORD	4 bytes	Total block length	Length of complete Layers Block, including all Layer Sub-Blocks.

The information contained within this document is confidential and is subject to change without notice

Layer Sub-Block Header (first layer):

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "7E 42 4B 00" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_LAYER_BLOCK.
DWORD	4 bytes	Initial data chunk length	Size of following Layer Information Chunk.
DWORD	4 bytes	Total block length	Length of complete Layer Block, including Layer Information Chunk and all Channel Sub-Blocks.

Layer Information Chunk (first layer):

C_ARRAY	256 bytes	Layer name	Name of layer (in ASCII text).
BYTE	1 byte	Layer type	Type of layer (must be one of PSPLayerType).
RECT	16 bytes	Image rectangle	Rectangle defining image border.
RECT	16 bytes	Saved image rectangle	Rectangle within image rectangle that contains "significant" data (only the contents of this rectangle are saved to the file).
BYTE	1 byte	Layer opacity	Overall layer opacity.
BYTE	1 byte	Blending mode	Mode to use when blending layer.
BYTE	1 byte	Layer visibility flag	TRUE if layer was visible at time of save, FALSE otherwise.
BYTE	1 byte	Transparency protected flag	TRUE if transparency is protected.
BYTE	1 byte	Link group identifier	Identifies group to which this layer belongs.
RECT	16 bytes	Mask rectangle	Rectangle defining user mask border.
RECT	16 bytes	Saved mask rectangle	Rectangle within mask rectangle that contains "significant" data (only the contents of this rectangle are saved to the file).
BYTE	1 byte	Mask linked	TRUE if mask linked to layer (i.e., mask moves relative to layer), FALSE otherwise.
BYTE	1 byte	Mask disabled	TRUE if mask is disabled, false otherwise.
BYTE	1 byte	Invert mask on blend	TRUE if mask should be inverted when the layer is merged, FALSE otherwise.
WORD	2 bytes	Blend range count	Number of valid source-destination field pairs to follow (note, there are currently always 5 such pairs, but they are not necessarily all valid).

B_ARRAY	4 bytes	Source blend range #1	First source blend range value.
B_ARRAY	4 bytes	Destination blend range #1	First destination blend range value.
B_ARRAY	4 bytes	Source blend range #2	Second source blend range value.
B_ARRAY	4 bytes	Destination blend range #2	Second destination blend range value.
B_ARRAY	4 bytes	Source blend range #3	Third source blend range value.
B_ARRAY	4 bytes	Destination blend range #3	Third destination blend range value.
B_ARRAY	4 bytes	Source blend range #4	Fourth source blend range value.
B_ARRAY	4 bytes	Destination blend range #4	Fourth destination blend range value.
B_ARRAY	4 bytes	Source blend range #5	Fifth source blend range value.
B_ARRAY	4 bytes	Destination blend range #5	Fifth destination blend range value.
WORD	2 bytes	Count of bitmaps	Number of bitmaps to follow.
WORD	2 bytes	Channel count	Number of channels to follow.

Layer Channel Sub-Blocks (first layer):

All the Layer's channels. Possible channels include 1) one or three Channel Sub-Blocks defining the layer's color bitmap, 2) one channel defining the Layer's transparency mask and 3) one channel defining the Layer's user mask. Note that the type of each channel can be obtained from the Channel Sub-Block (see section 5.6) and the number of channels is specified in the previous Layer Information Chunk. Note also that the Layer's transparency mask and the Layer's user mask are stored as 8-bit greyscale bitmaps.

• • •

Layer Sub-Block Header (last layer):

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_LAYER_BLOCK.
DWORD	4 bytes	Initial data chunk length	Always 0.
DWORD	4 bytes	Total block length	Length of complete Layers Block, including all Layer Sub-Blocks.

Layer Information Chunk (last layer):

C_ARRAY	256 bytes	Layer name	Name of layer (in ASCII text).
BYTE	1 byte	Layer type	Type of layer (must be one of PSPLayerType).
RECT	16 bytes	Image rectangle	Rectangle defining image border.
RECT	16 bytes	Saved image rectangle	Rectangle within image rectangle that contains “significant” data (only the contents of this rectangle are saved to the file).
BYTE	1 byte	Layer opacity	Overall layer opacity.
BYTE	1 byte	Blending mode	Mode to use when blending layer.
BYTE	1 byte	Layer flags	PSPLayerFlags values, bitwise-ored together.
BYTE	1 byte	Layer transparency flag	PSPLayerTransparencyFlags values, bitwise-ored together.
BYTE	1 byte	Link group identifier	Identifies group to which this layer belongs.
RECT	16 bytes	Mask rectangle	Rectangle defining user mask border.
RECT	16 bytes	Saved mask rectangle	Rectangle within mask rectangle that contains “significant” data (only the contents of this rectangle are saved to the file).
BYTE	1 byte	Mask linked	TRUE if mask linked to layer (i.e., mask moves relative to layer), FALSE otherwise.
BYTE	1 byte	Mask disabled	TRUE if mask is disabled, false otherwise.
BYTE	1 byte	Invert mask on blend	TRUE if mask should be inverted when the layer is merged.
WORD	2 bytes	Blend range count	Number of valid source-destination field pairs to follow (note, there are currently always 5 such pairs, but they are not necessarily all valid).
B_ARRAY	4 bytes	Source blend range #1	First source blend range value.
B_ARRAY	4 bytes	Destination blend range #1	First destination blend range value.
B_ARRAY	4 bytes	Source blend range #2	Second source blend range value.
B_ARRAY	4 bytes	Destination blend range #2	Second destination blend range value.
B_ARRAY	4 bytes	Source blend range #3	Third source blend range value.
B_ARRAY	4 bytes	Destination blend range #3	Third destination blend range value.
B_ARRAY	4 bytes	Source blend range #4	Fourth source blend range value.
B_ARRAY	4 bytes	Destination blend range #4	Fourth destination blend range value.

B_ARRAY	4 bytes	Source blend range #5	Fifth source blend range value.
B_ARRAY	4 bytes	Destination blend range #5	Fifth destination blend range value.
WORD	2 bytes	Count of bitmaps	Number of bitmaps to follow.
WORD	2 bytes	Channel count	Number of channels to follow.

Layer Channel Sub-Blocks (last layer):

All the Layer's channels. Possible channels include 1) one or three Channel Sub-Blocks defining the layer's color bitmap, 2) one channel defining the Layer's transparency mask and 3) one channel defining the Layer's user mask. Note that the type of each channel can be obtained from the Channel Sub-Block (see section 5.6) and the number of channels is specified in the previous Layer Information Chunk. Note also that the Layer's transparency mask and the Layer's user mask are stored as 8-bit greyscale bitmaps.

Table 8: Layer Bank Block

5.10 The Selection Block

The Selection Block is an optional block that defines the area that was selected when the image document was saved.

As illustrated in the following table, the Selection Block consists of a block header, a Selection Information Chunk and a Selection Channel Sub-Block.

Selection Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_SELECTION_BLOCK.
DWORD	4 bytes	Initial data chunk length	Length of following Selection Information Chunk.
DWORD	4 bytes	Total block length	Length of complete Selection Block, including Selection Information Chunk and Selection Channel Sub-Block.

The information contained within this document is confidential and is subject to change without notice

Selection Information Chunk:

RECT	16 bytes	Selection rectangle	Rectangle containing selected data.
RECT	16 bytes	Saved selection rectangle	Rectangle within selection rectangle that contains "significant" data (only the contents of this rectangle are saved to the file).
WORD	2 bytes	Selection bitmap count	Count of following selection bitmaps. Currently always 1.
WORD	2 bytes	Channel count	Count of following channels. Currently always 1.

Selection Channel Sub-Block:

Selection Channel Sub-Block. See section 5.6. Note that selections are stored as 8-bit greyscale bitmaps.

Table 9: Selection Block

5.11 The Alpha Bank Block

The Alpha Bank Block is an optional block that defines alpha channels associated with the image document.

As is illustrated in the following table, the Alpha Bank Block consists of a block header, an Alpha Channel Information Chunk, and an Alpha Channel Sub-Block for each alpha channel in the image document (each Alpha Channel Sub-Block consists of a block header and a Channel Sub-Block).

Alpha Channel Bank Header:

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_ALPHA_BANK_BLOCK.
DWORD	4 bytes	Initial data chunk length	Length of following Alpha Bank Information Chunk.
DWORD	4 bytes	Total block length	Length of complete Alpha Bank Block, including Alpha Bank Information Chunk, and Alpha Channel Headers and Alpha Channel Information Chunks and Alpha Channel Sub-Blocks for all alpha channels.

Alpha Bank Information Chunk:

WORD	2 bytes	Alpha channel count	Number of Alpha channels.

Alpha Channel Header (first alpha channel):

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_ALPHA_CHANNEL_BLOCK.
DWORD	4 bytes	Initial data chunk length	Length of following Alpha Channel Information Chunk.
DWORD	4 bytes	Total block length	Length of complete Alpha Channel Block, including following Alpha Channel Information Chunk and Alpha Channel Sub-Block.

Alpha Channel Information Chunk (last alpha channel):

C_ARRAY	256 bytes	Alpha channel name	Name of alpha channel.
RECT	16 bytes	Alpha channel rectangle	Rectangle containing alpha channel.
RECT	16 bytes	Saved alpha channel rectangle	Rectangle within alpha channel rectangle that contains "significant" data (only the contents of this rectangle are saved to the file).
WORD	2 bytes	Alpha channel bitmap count	Number of following alpha channel bitmaps (currently always 1).
WORD	2 bytes	Channel count	Number of following channels (currently always 1).

Alpha Channel Sub-Block (first alpha channel):

Channel Sub-Block defining the alpha channel. See section 5.6. Note that alpha channels are stored as 8-bit greyscale bitmaps.

• • •

Alpha Channel Header (last alpha channel):

Type	Length	Name	Description
B_ARRAY	4 bytes	Header identifier	Always "`7E 42 4B 00`" (i.e.,"~BK", followed by a zero byte).
WORD	2 bytes	Block identifier	Always PSP_ALPHA_CHANNEL_BLOCK.
DWORD	4 bytes	Initial data chunk length	Length of following Alpha Channel Information Chunk.
DWORD	4 bytes	Total block length	Length of complete Alpha Channel Block, including Alpha Channel Information Chunk and Alpha Channel Sub-Block.

Alpha Channel Information Chunk (last alpha channel):

C_ARRAY	256 bytes	Alpha channel name	Name of alpha channel.
RECT	16 bytes	Alpha channel rectangle	Rectangle containing alpha channel.
RECT	16 bytes	Saved alpha channel rectangle	Rectangle within alpha channel rectangle that contains "significant" data (only the contents of this rectangle are saved to the file).
WORD	2 bytes	Alpha channel bitmap count	Number of following alpha channel bitmaps (currently always 1).
WORD	2 bytes	Channel count	Number following channels (currently always 1).

Alpha Channel Sub-Block (last alpha channel):

Channel Sub-Block defining the alpha channel. See section 5.6. Note that alpha channels are stored as 8-bit greyscale bitmaps.

Table 10: Alpha Bank Block