“FOVb” (.X3F) File Format External Specification

Spec Version history:

Date	Version	Changes
14 Jan 2003	1.0 (rfl)	from internal spec 0.32 (v2.2)
21 Oct 2003	1.1 (rfl)	details for public release
23 May 2004	1.2 (rfl) from 3.07 (rss)	Added JPEG image format, removed DIRP

Document note: this document describes the .X3F file format in enough detail that the preview and thumbnail images, and image properties, can be read. Image encodings other than types 3, 11, and 18 have been removed from this public document. Contact Foveon for more information.

1 Purpose

The FOVb (.X3F) file format is designed to hold the data associated with a single image. This can include image data, thumbnail(s), preview image(s), text properties, etc.

The file format is designed to allow changing the file data by appending subsequent information to the file. This allows the file format to be used on write-once media, where earlier data in the file cannot be changed.

2 Basic File Format

A FOVb format file always contains the following sections:

Section	Notes
Header	
Extended Header	header extension
Data	
Directory	Directory of subsections in the data section.
Directory Pointer	Offset from the start of the file to the start of the directory section, in bytes.

The directory is at the end of the file. This is necessary because it may not be possible to know the sizes of all the items in the data section before writing the data section. Putting the directory section last allows the entire file to be written sequentially. This is critical for write-once devices (for example, optical media) and desirable for media which is erased in large blocks (for example, solid-state memory such as Compact Flash cards).

It is possible that a file write might be terminated prematurely (for example, if the camera loses power while writing). If the last 4 bytes of a file do not point to the start of a directory, it may still be possible to recover the earlier portions of the file by searching the file for the magic numbers that should be present at the start of each section.

All multi-byte numbers are little-endian / LSB-first.

2.1 Version Numbers

Version numbers are 32-bit values. The high 16 bits of the version number are the major version number and the low 16 bits are the minor version number. For example, version 1.3 would be encoded as 0x00010003.

The major version number is incremented when a change to the FOVb file format prevents previous readers from properly parsing the file. The minor version number is incremented when the format is extended in a way which does not prevent older cam file readers with the same major version number from reading the file.

In the abstract form, if the cam file reader implements version A.b of this specification and the cam file is version C.d, here is a quick compatibility chart:

A != C	The reader cannot read the file.
A = C, b < d	The reader can read the file, but some elements of the file may not be parsed optimally / with all options.
A = C, b = d	The reader can read the file.
A = C, b > d	The reader can read the file.

2.2 Header Section

2.2.1 Versions 2.1 – 2.2

Offset	Length	Item	Notes
0	4	File type identifier	Contains "FOVb". Used to verify that this is an FOVb file.
4	4	File format version	Version of the file. Should be 2.2 right now.
8	16	Unique identifer	Guaranteed unique to each image. Formed from camera serial number / OUI, timestamp, and high-speed timer register. Can be used to identify images even if they are renamed. No, it's not UUID-compatible.
24	4	Mark bits	Can be used to denote that images are marked into one or more subsets. File interface functions allow setting these bits and searching for files based on these bits. This feature will not be usable on write-once media.
28	4	Image columns	Width of unrotated image in columns. This is the size output the user expects from this image, not the size of the raw image data. Not necessarily equal to the width of any image entry in the file; this supports images where the raw data has rectangular pixels.
32	4	Image rows	Height of unrotated image in rows. This is the size output the user expects from this image, not the

			size of the raw image data. Not necessarily equal to the width of any image entry in the file; this supports images where the raw data has rectangular pixels.
36	4	Rotation	Image rotation in degrees clockwise from normal camera orientation. Valid values are 0, 90, 180, 270.
40	32	White balance label string	Contains an ASCIIZ string label of the current white balance setting for this image.
72	32	Extended data types	Contains 32 8-bit values indicating the types of the following extended data.
104	128	Extended data	Contains 32 32-bit values of extended data.

2.2.2 Version 2.0 and older

This version of the FOVb header does not have the white balance label string or extended data.

Offset	Length	Item	Notes
0	4	File type identifier	Contains "FOVb". Used to verify that this is an FOVb file.
4	4	File format version	Version of the file. Should be 2.0 or lower.
8	16	Unique identifer	Guaranteed unique to each image. Formed from camera serial number / OUI, timestamp, and high-speed timer register. Can be used to identify images even if they are renamed. No, it's not UUID-compatible.
24	4	Mark bits	Can be used to denote that images are marked into one or more subsets. File interface functions allow setting these bits and searching for files based on these bits. This feature will not be usable on write-once media.
28	4	Image columns	Width of unrotated image in columns. This is the size output the user expects from this image, not the size of the raw image data. Not necessarily equal to the width of any image entry in the file; this supports images where the raw data has rectangular pixels.
32	4	Image rows	Height of unrotated image in rows. This is the size output the user expects from this image, not the size of the raw image data. Not necessarily equal to the width of any image entry in the file; this supports images where the raw data has rectangular pixels.
36	4	Rotation	Image rotation in degrees clockwise from normal camera orientation. Valid values: 0, 90, 180, 270.

2.2.3 Extended data types (version 2.1+)

The following types of data may be present in the extended data portion of the header. Readers should ignore types they do not recognize.

Value	Data type	Notes
0	None	Extended data slot is unused.
1	FLOAT32	Exposure adjust.
2	FLOAT32	Contrast adjust.
3	FLOAT32	Shadow adjust.
4	FLOAT32	Highlight adjust.
5	FLOAT32	Saturation adjust.
6	FLOAT32	Sharpness adjust.
7	FLOAT32	Color adjust red.
8	FLOAT32	Color adjust green.
9	FLOAT32	Color adjust blue.
10	FLOAT32	X3 Fill Light adjust.

2.3 Data Section

The data section consists of zero or more subsections. Each subsection should have a corresponding entry in the directory. Subsections must start on 32-bit boundaries.

2.4 Directory Section

The directory section always starts with the following data:

Offset	Length	Item	Notes
0	4	Section identifier	Contains "SECd".
4	4	Section version	Section version. Should be 2.0 for now.
4	4	Number of directory entries.	

This is followed by one or more directory entries in the following format:

Offset	Length	Item	Notes
0	4	Offset from start of file to start of entry's data, in bytes.	Offset must be a multiple of 4, so that the data starts on a 32-bit boundary.
4	4	Length of entry's data, in bytes.	
8	4	Type of entry.	See below for a list of valid types.

All subsections in the data section should have corresponding entries in the directory.

2.5 Directory Pointer Section

The directory pointer section consists of the following data:

Offset	Length	Item	Notes
0	4	Offset of start of directory section from start of file, in bytes.	

3 Data Subsection Types

Readers should ignore data subsections with unfamiliar types.
The following types of data are currently defined (others are RESERVED)

Type of Entry	Contents	Notes
"PROP"	Property list.	List of pairs of strings. Each pair is a name and its corresponding value.
"IMAG"	Image data	Image data. Has a header indicating dimensions, pixel type, compression, amount of processing done.
"IMA2"	Image data	Image data. Readers should treat this the same as IMAG. Writers should use this for image sections that contain processed-for-preview data in other than uncompressed RGB24 pixel format.

A file may have more than one subsection of each data type, as well as other subsections of types not specified here. For example, there could be several "PROP" subsections.

3.1 PROP – Property List

A property list is a list of name-value pairs, stored as 16-bit null-terminated Unicode character strings.

A property list always starts with the following header:

Offset	Length	Item	Notes
0	4	Section identifier	Should be "SECp"
4	4	Property list format version	Should be 2.0 for now.
8	4	Number of property entries	
12	4	Character format for all entries in this table.	0 = CHAR16 Unicode.
16	4	RESERVED	
20	4	Total length of name/value data in characters.	

The property list header is following by one or more property entries, in the following format:

Offset	Length	Item	Notes
0	4	Offset of name	Offset in characters of property name from start of character data. Not necessarily immediately following the previous property's value.
4	4	Offset of value	Offset in characters of property value from start of character data.

The property entries are followed by the name and value data for the entries.

The next name after a value does not necessarily immediately follow that value in the character data. This allows padding values to allow space for in-place editing on rewritable media.

If a property name occurs more than once, or in more than one property list, the returned value will be from the last match found while traversing all of the lists. This is needed to support updating properties for files which are stored on write-once media.

3.2 IMAG or IMA2 – Image Data

All image types begin with the following header:

Offset	Length	Item	Notes
0	4	Section identifier	Should be "SECi"
4	4	Image format version	Should be 2.0 for now.
8	4	Type of image data	2 = processed for preview (others RESERVED)
12	4	Data format	3 = uncompressed 24-bit 8/8/8 RGB 11 = Huffman-encoded DPCM 8/8/8 RGB 18 = JPEG-compressed 8/8/8 RGB (others RESERVED)
16	4	Image columns	Image width / row size in pixels
20	4	Image rows	Image height in pixels
24	4	Row size in bytes	Will always be a multiple of 4 (32-bit aligned). A value of zero here means that rows are variable-length (as in Huffman data).

This header is followed by the image data itself. Depending on the data format, this may include a data-format-specific header. For example, Huffman-encoded data includes the Huffman code table.

For compatibility with earlier readers, the directory that refers to an image section should use a section type of IMAG, with one exception noted below for the data format 11 (Huffman compressed 8/8/8 RGB). IMA2 was added in version 2.2 spec.

The data present in this header is sufficient to be able to decode the image data for a particular pixel.

Images of type 2 (processed for preview) are suitable for display as a thumbnail or preview on an sRGB display device. Various sizes may be available, so they should be compared to find a suitable size for the intended application.

3.2.1 Data format 3 – uncompressed 24-bit 8/8/8 RGB

Image data is a series of the following 48-bit-wide structures:

8-bit word 0	8-bit word 1	8-bit word 2
RRRRRRRR	GGGGGGGG	BBBBBBBB
00000000	00000000	00000000
76543210	76543210	76543210

Rows will be padded with 0's to a multiple of 4 bytes.

3.2.2 Data format 11 – Huffman-encoded DPCM 8/8/8 RGB

This format is intended for compressing 24-bit RGB preview data.

For compatibility with earlier readers, the directory that refers to an image section of this format should use a section type of IMA2, not IMAG.

Row size in bytes will always be zero for this format, because the rows are variable length.

Image data begins with a 256-entry table of UINT32 values which is the codeword table. The upper 5 bits of each entry are the length of the codeword in bits; the bottom 27 bits are the codeword, LSB-justified:

Codeword table entry
LLLLL CCCCCCCCCCCCCCCCCCCCCCCCCCC
00000 22222221111111110000000000
43210 65432109876543210987654321 0

This is followed by the encoded data. Each row of (3 * number of columns) 8-bit values is encoded separately, in the order $\{R_0, G_0, B_0, R_1, G_1, B_1, \ldots R_{N-1}, G_{N-1}, B_{N-1}\}$. Rows are always padded to the next 32-bit boundary.

This is followed by an array of (number of rows) 32-bit values, containing the byte offset of each row from the first byte of encoded data.

To decompress the pixel data, use the following algorithm for each row of each color channel, where C_i is the unsigned 8-bit value of a color channel in the i^{th} pixel on that row and D_i is the unsigned 8-bit decompressed value of that color channel for that pixel:

$$D_0 = C_0$$
$$D_{i+1} = D_i + C_{i+1}$$

Note that the addition should be done in 8-bit unsigned math, so that 200 + 57 = 1, not 257.

3.2.3 Data format 18 – JPEG-compressed 8/8/8 RGB

This format contains a complete lossy-JPEG-compressed 24-bit RGB image file.

The data for this section, if extracted to a separate file, should be readable by a standard JPEG reader (e.g. JFIF or DCF/EXIF format, readable by the code available at http://www.ijg.org/).

The JPEG file contained in this format may optionally contain EXIF information and/or an embedded thumbnail, if doing so is easier for a JPEG writer. JPEG readers are not expected to use this information.

This image type may be used to store a preview image corresponding to the X3F raw data; it can be stored at any resolution and any quality level. This image type is not supported by the reader in Sigma Photo Pro 2.0 and earlier.

3.3 DIRP – Directory Pointer

This previously-documented option for an added-on directory has been removed from the specification of FOVb versions 2.x because it has never been used and no readers support it.

4 Standard Property Names

Property Name	Example Value	Notes
AEMODE	8	Auto-exposure mode ("8" = 8-segment, "C" = center-weighted, "A" – average-weighted).
AFMODE	AF-S	Auto-focus mode ("AF-S" = single, "AF-C" = continuous, "MF" = manual focus)
AP_DESC	5.6	Aperture shown on LCD at time of shot.
APERTURE	5.65685	Exact aperture value.
BRACKET	3 of 9	If camera is shooting in auto-bracket mode, indicates the shot number in the bracket (starting with "1 of ___"
BURST	3	If camera is shooting in multishot mode, indicates the shot number within the burst; the first shot is "1".
CAMMANUF	OasisCo	Camera manufacturer
CAMMODEL	Fov2001	Camera model name.
CAMNAME	Bob's Camera	Camera name.
CAMSERIAL	FOV3-12345678	Camera serial number.
DRIVE	SINGLE	Drive dial position ("SINGLE" = single, "MULTI" = multishot, "2S" = 2-sec timer, "10S"

		= 10-sec timer, "UP" = mirror-up mode, "AB" = auto-bracket, "OFF" = off.
EXPCOMP	-1.00	User-set exposure compensation.
EXPNET	-1.50	Net exposure compensation for this shot (sum of exposure compensation and bracketing compensation).
EXPTIME	8000	Exposure length (integration time) in microseconds.
FIRMVERS	1.0.1.220 Eng	Firmware version/build.
FLASH	OFF	Flash setting ("ON", "OFF", "REDEYE")
FLENGTH	50	Focal length in mm.
FLEQ35MM	85	35mm equivalent focal length in mm.
FOCUS	AF	Focus status ("AF" = auto-focus locked, "NO LOCK" = auto-focus didn't lock, "M" = manual; user did focusing).
IMAGERBOARDID	001234-01 Rev3 Mod15 #10203	Imager board ID. Part number, board revision, hardware mod level, serial number.
IMAGERTEMP	I 27 28 D 28 29	Temperature (Celsius) of sensor / header board before and after image and before and after the last dark frame. If no dark frame is present, the D and following numbers will be absent.
ISO	100	ISO film speed equivalent.
LENSARANGE	2.8 to 32	Lens aperture range (in exact value numbers or LCD display numbers? at current zoom or ?)
LENSFRANGE	80 to 120	Lens focal length range in mm
LENSMODEL	123	Lens identifier byte or "NONE" if no lens present.
PMODE	P	Shooting mode ("P" = program, "A" = aperture-priority, "S" = shutter-priority, or "M" = manual).
RESOLUTION	HI	Image resolution shown on the LCD ("LOW", "MED", or "HI")
SENSORID	FOV19 Rev0	Sensor type and revision.
SH_DESC	1/125	Shutter speed shown on LCD at time of shot.
SHUTTER	0.007812	Exact shutter speed in seconds.
TIME	1000013247	Time in time()-style format – seconds since midnight 1/01/1970. The camera always uses UTC, so use gmtime() to parse time fields from this, not localtime().
WB_DESC	Sunlight	Original capture-time white balance setting. See WB_DESC under the Conditional Properties Section in the CAMb Content Specification for more information.

5 Addendum: recovering X3F files from corrupted media.

A suggested simple file recovery process:
File starts at "FOVb"
Then search for "SECd"
Retrieve the 32-bit number N which starts 4 bytes past "SECd".
File ends (4 + N * 12) bytes past the end of that number.