PS2 port of Half-life widely uses images in PSI format. There are two main types of PSI images: 8 bit and 32 bit. General PSI image file structure is shown on fig. 1.

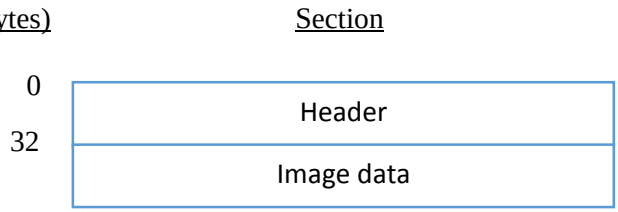

Fig. 1

Structure of PSI image header:

```
struct sPSIHeader
{
	char Name[16];		// Internal image name
	uchar Magic[3];		// Filled with zeroes in most cases
	uchar LODCount;		// Number of LODs that present in image file (used in decals only)
	ulong Type;		// 2 - 8 bit indexed bitmap, 5 - 32 bit RGBA bitmap
	ushort Width;		// Texture width (in pixels)
	ushort Height;		// Texture Height (in pixels)
	ushort UpWidth;		// Target image width for in-game upscale (in pixels)
	ushort UpHeight;	// Target image width for in-game upscale (in pixels)
}
```

1 8-bit PSI image

Structure of file is shown on fig. 1.

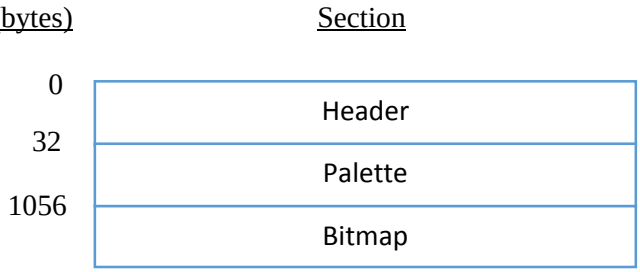

Fig. 1

Structure of 8-bit PSI images quite similar to windows 8-bit BMPs. However, PSIs are fully supporting transparency.

1.1 Palette

Palette consists of 256 colors represented in 4-byte RGBA format.

Red, green and blue bytes in some cases (sprites, GUI images) can have maximum value of 127 instead of normal 255.

The most significant bit in alpha byte tells if color is transparent or not. Other 7 bits represent level of color transparency. If the most significant bit is set to 1 (non transparent), value of other bits is ignored.

Palette addressing in PSI images is not linear. You can see it on fig. 2.

BMP

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F
80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
90	91	92	93	94	95	96	97	98	99	9A	9B	9C	9D	9E	9F
A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF
B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF
C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF
D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF
E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF
F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF

PSI

00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
08	09	0A	0B	0C	0D	0E	0F	18	19	1A	1B	1C	1D	1E	1F
20	21	22	23	24	25	26	27	30	31	32	33	34	35	36	37
28	29	2A	2B	2C	2D	2E	2F	38	39	3A	3B	3C	3D	3E	3F
40	41	42	43	44	45	46	47	50	51	52	53	54	55	56	57
48	49	4A	4B	4C	4D	4E	4F	58	59	5A	5B	5C	5D	5E	5F
60	61	62	63	64	65	66	67	70	71	72	73	74	75	76	77
68	69	6A	6B	6C	6D	6E	6F	78	79	7A	7B	7C	7D	7E	7F
80	81	82	83	84	85	86	87	90	91	92	93	94	95	96	97
88	89	8A	8B	8C	8D	8E	8F	98	99	9A	9B	9C	9D	9E	9F
A0	A1	A2	A3	A4	A5	A6	A7	B0	B1	B2	B3	B4	B5	B6	B7
A8	A9	AA	AB	AC	AD	AE	AF	B8	B9	BA	BB	BC	BD	BE	BF
C0	C1	C2	C3	C4	C5	C6	C7	D0	D1	D2	D3	D4	D5	D6	D7
C8	C9	CA	CB	CC	CD	CE	CF	D8	D9	DA	DB	DC	DD	DE	DF
E0	E1	E2	E3	E4	E5	E6	E7	F0	F1	F2	F3	F4	F5	F6	F7
E8	E9	EA	EB	EC	ED	EE	EF	F8	F9	FA	FB	FC	FD	FE	FF

Fig. 2 – Comparison of palette addressing in BMP (left) and PSI (right) images (red lines are added for better visibility)

My approach to converting palette to/from PSI format looks like that:

```
for (int PElement = 0; PElement < 256; PElement++)
{
	int Remainder = PElement % 32;
	if (16 <= Remainder) && (Remainder <= 23)
	{
		char Temp = Palette[PElement];
		Palette[PElement] = Palette[PElement - 8];
		Palette[PElement - 8] = Temp;
	}
}
```

1.2 Bitmap

Bitmap is generic 8-bit indexed one. Value of each byte of bitmap represents index of color in palette.

2 32-bit PSI image

Structure of 32-bit PSI image is shown on fig. 3.

Offset (bytes)	Section
0	Header
32	Bitmap

Fig. 3

2.1 Bitmap

Color of each pixel of image is stored in bitmap in 4-byte RGBA format.

Red, green, blue and alpha bytes can have maximum value of 127 instead of normal 255.

Alpha byte value can vary from 0 (fully transparent) to 127 (fully non-transparent).

3 LODs

PSI images inside decals have LODs. Each LOD is bitmap with half dimensions of previous LOD (half dimensions of original image for first LOD). Structure of 8-bit 64*64 PSI example image with 3 LODs is shown on fig. 4. Graphical representation is shown on fig. 5.

Offset (bytes)	Section
0	Header
32	Palette
1056	Image bitmap (64*64)
5152	LOD 1 bitmap (32*32)
6176	LOD 2 bitmap (16*16)
6432	LOD 3 bitmap (8*8)
6496	

Fig. 4 – Example image file structure

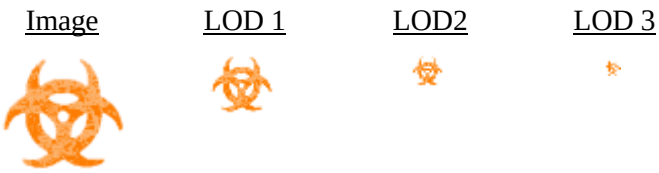

Fig. 5 – Graphical representation