HFE HxC Floppy Emulator file format

(Note : All data in this file are subject to changes)

Changes :

29 November 2010 – v1.0 : Initial version.
20 June 2012 – v1.1 : Add single_step, track0s0/1_altencoding and track0s0/1_encoding header fields.

Description :

The HFE file format is a simple floppy bitstream tracks container originally designed for the SD HxC Floppy Emulator hardware (PIC18F based).

HFE overall structure :

The HFE file format contains a file header with metadata like the number of tracks in the file, track format ID, floppy interface configuration…

The second part is a tracks offsets and sizes array.

And finally all the tracks bitstream buffers.

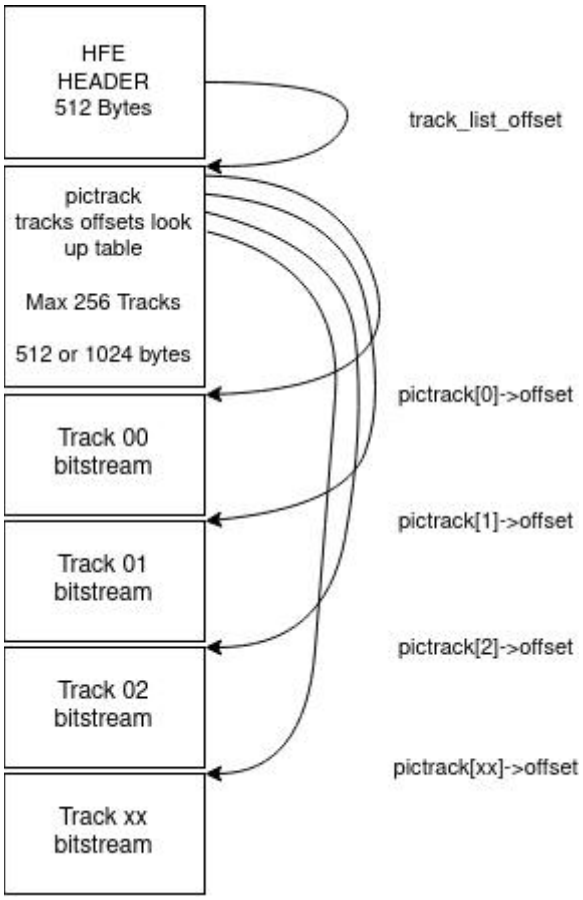

Figure 1: HFE format structure

First part : 0x0000-0x0200 (512 bytes) : HFE header

```
typedef struct picfileformatheader_
{
        unsigned char HEADERSIGNATURE[8];   // “HXCPICFE”
        unsigned char formatrevision;       // Revision 0
        unsigned char number_of_track;      // Number of track in the file
        unsigned char number_of_side;       // Number of valid side (Not used by the emulator)
        unsigned char track_encoding;       // Track Encoding mode
                                            // (Used for the write support - Please see the list above)
        unsigned short bitRate;             // Bitrate in Kbit/s. Ex : 250=250000bits/s
                                            // Max value : 500
        unsigned short floppyRPM;           // Rotation per minute  (Not used by the emulator)
        unsigned char floppyinterfacemode;  // Floppy interface mode. (Please see the list above.)
        unsigned char dnu;                  // Reserved
        unsigned short track_list_offset;   // Offset of the track list LUT in block of 512 bytes
                                            // (Ex: 1=0x200)
        unsigned char write_allowed;        // The Floppy image is write protected ?

        // v1.1 addition – Set them to 0xFF if unused.
        unsigned char single_step;          // 0xFF : Single Step – 0x00 Double Step mode
        unsigned char track0s0_altencoding; // 0x00 : Use an alternate track_encoding for track 0 Side 0
        unsigned char track0s0_encoding;    // alternate track_encoding for track 0 Side 0
        unsigned char track0s1_altencoding; // 0x00 : Use an alternate track_encoding for track 0 Side 1
        unsigned char track0s1_encoding;    // alternate track_encoding for track 0 Side 1
}picfileformatheader;
```

Note : short fields are in little endian format.

Note : Unused header bytes must be set to 0xFF.

floppyinterfacemode values :

```
#define IBMPC_DD_FLOPPYMODE                     0x00
#define IBMPC_HD_FLOPPYMODE                     0x01
#define ATARIST_DD_FLOPPYMODE                   0x02
#define ATARIST_HD_FLOPPYMODE                   0x03
#define AMIGA_DD_FLOPPYMODE                     0x04
#define AMIGA_HD_FLOPPYMODE                     0x05
#define CPC_DD_FLOPPYMODE                       0x06
#define GENERIC_SHUGGART_DD_FLOPPYMODE          0x07
#define IBMPC_ED_FLOPPYMODE                     0x08
#define MSX2_DD_FLOPPYMODE                      0x09
#define C64_DD_FLOPPYMODE                       0x0A
#define EMU_SHUGART_FLOPPYMODE                  0x0B
#define S950_DD_FLOPPYMODE                      0x0C
#define S950_HD_FLOPPYMODE                      0x0D
#define DISABLE_FLOPPYMODE                      0xFE
```

track_encoding / track0s0_encoding / track0s1_encoding values :

#define ISOIBM_MFM_ENCODING	0x00
#define AMIGA_MFM_ENCODING	0x01
#define ISOIBM_FM_ENCODING	0x02
#define EMU_FM_ENCODING	0x03
#define UNKNOWN_ENCODING	0xFF

Note :

If track0s0_altencoding is set to 0xFF, track0s0_encoding is ignored and track_encoding is used for track 0 side 0.

If track0s1_altencoding is set to 0xFF, track0s1_encoding is ignored and track_encoding is used for track 0 side 1.

Second part : (up to 1024 bytes) : Track offset LUT

```
typedef struct pictrack_
{
	unsigned short offset;		// Track data offset in block of 512 bytes (Ex: 2 = 0x400)
	unsigned short track_len;	// Length of the track data in byte.
}pictrack;
```

For a disk of 80 tracks there are a table of 80 pictrack structure.

Pictrack[80];

Note : short fields are in little-endian format.

Third part : Track data

A track data is a table containing the bit stream of a track of the floppy disk. A track can contain a MFM / FM / GCR or a custom encoding.

The track is divided in block of 512 Bytes and each block contains a part of the Side 0 track and a part of the Side 1 track:

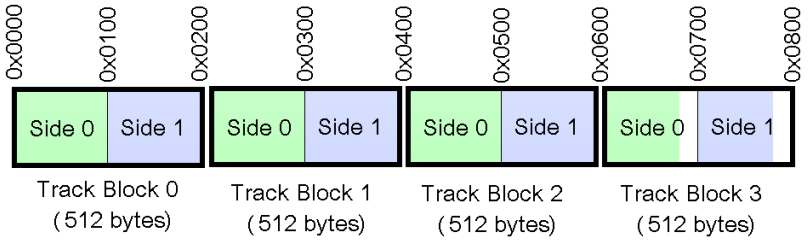

Figure 2 : A track data

The bits transmitting order to the FDC is LSB first :

Bit 0-> Bit 1-> Bit 2-> Bit 3-> Bit 4-> Bit 5-> Bit 6-> Bit 7->(next byte)

Note :

The bitstream content is specific to each targeted system and disk format ! The low level floppy disk controller track and sectors formats descriptions is not covered by this document ! For more floppy disks related informations, please have a look to these documentations : https://hxc2001.com/download/datasheet/floppy/thirdparty/