VMS File Header

All VMS files should contain a standard header which is used by the file managers in the VMS and in the DC boot ROM to display information about the file. (ICONDATA files use a somewhat simplified headers since they are not shown in the DC boot ROM file manager.) For data files, the header is stored at the very beginning of the file. For game files, it begins in the *second* block of the file (offset $200). This fact should be reflected by the header offset field in the VMS directory, which should contain 1 for game files, and 0 for data files.

This is the contents of the header:

Offset	Size (bytes)	Datatype	Contents
$00	16	Text	Description of file (shown in VMS file menu)
$10	32	Text	Description of file (shown in DC boot ROM file manager)
$30	16	String	Identifier of application that created the file
$40	2	Integer	Number of icons (>1 for animated icons)
$42	2	Integer	Icon animation speed
$44	2	Integer	Graphic eyecatch type (0 = none)
$46	2	Integer	CRC (Ignored for game files.)
$48	4	Integer	Number of bytes of actual file data following header, icon(s) and graphic eyecatch. (Ignored for game files.)
$4C	20		Reserved (fill with zeros)
$60	32	Integers	Icon palette (16 16-bit integers)
$80	512*n	Nybbles	Icon bitmaps
...	depends on type	...	Graphic eyecatch palette and bitmap

Text fields are padded with space ($20). String fileds are padded with NUL ($00). Integer fields are little endian.

CRC calculation

The CRC should be computed on the entire file, including header and data payload, but excluding any padding in the final block. When calcukating the CRC, set the CRC field itself to 0 to avoid miscalculating the CRC for the header. The CRC calculation algorithm is as follows (C code):

```
int calcCRC(const unsigned char *buf, int size)
{
  int i, c, n = 0;
  for (i = 0; i < size; i++)
  {
    n ^= (buf[i]<<8);
    for (c = 0; c < 8; c++)
      if (n & 0x8000)
        n = (n << 1) ^ 4129;
      else
        n = (n << 1);
  }
```

```
  return n & 0xffff;
}
```

Icon palette

The palette consists of 16 16-bit little endian integers, one for each colour in the palette. Each integer consists of four four-bit fields:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Alpha				Red				Green				Blue			

Alpha 15 is fully opaque, alpha 0 is fully transparent.

Icon bitmaps

The header should contain at least one icon bitmap for the file. If an animated icon is desired, up to three bitmaps can be used (set the field at offset $40 accordingly). The bitmaps are stored directly after each other, starting at offset $80. The bitmaps contain one nybble per pixel. Each byte thus represents two horizontally adjacent pixels, the high nybble being the left one and the low nybble being the right one. Each complete bitmap contains 1024 (32 * 32) nybbles, or 512 bytes.

Graphic eyecatch

The header can optionally contain a 72×56 pixel image shown as a graphic eyecatch for the file in the DC boot ROM file manager. The graphic data for the eyecatch is stored immediately after the last icon bitmap. There are four possible visual modes, selected with the field at offset $44:

Mode	Graphic data size (bytes)	Data format
0	0	None
1	8064	16-bit true colour, see Icon palette for pixel format.
2	4544 (512 bytes palette, 4032 bytes bitmap)	256 colour palette based. Begins with a palette in the same format as the Icon palette, but with 256 entries. Then the bitmap has one byte per pixel, giving the index into the palette.
3	2048 (32 bytes palette, 2016 bytes bitmap)	16 colour palette based. Format is just like the Icon palette and bitmap, except for the width and height of the bitmap of course.
