Animation Demonstration

This Help file demonstrates animation in an embedded pane and sounds.

It was created using EditHelp available from **Analogue Ltd., 1 Warrender Park Crescent, Edinburgh EH9 1DX, (031 228 6008).** Shareware registration is £25+VAT.

The animation is executed via the DLL:

AnimHelp.DLL

The source bitmaps and the playing instructions are kept in the HLP file as "Baggage". They are not compressed - which is why the HLP file is so big and takes so long to load. A compressed version should be available soon.

The implementation uses "dirty rectangle" animation: only the parts of the screen which have changed are updated.

This Help file contains three demonstrations:

Dinosaur
Disk Encoding
Playing Sounds

At present, there is no animation authoring tool. You have to calculate the "dirty rectangles" by hand. An authoring tool is planned.

FM Encoding

When data is stored on a low-density disk, FM Encoding is used.

Click the mouse on the picture or press the Enter key. Press Pg-Dn for more information:

{ewc HelpAnim.dll,AnimWnd,1/picpos.txt}

Notice that the width of the magnetisation domains is either 1 unit or 2 units.

MFM encoding can be used to store the data at twice this density.

MFM Encoding

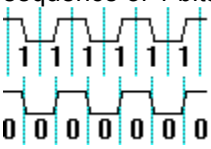
When data is stored on a high-density disk, MFM Encoding is used.

Click the mouse on the picture or press the Enter key. Press Pg-Dn for more information:

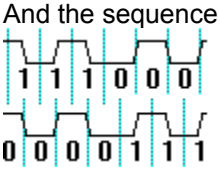
{ewc HelpAnim.dll,AnimWnd,1/picpos2.txt}

The width of the magnetisation domains is either 1 unit or 2 units or 1.5 units. The magnetic medium must be sufficiently good quality to record these fine differences in domain width.

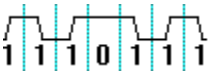
Note that, to decode the data, the disk controller must be synchronised with the bit frame. Otherwise, a sequence of 1 bits cannot be distinguished from a sequence of 0's:



And the sequence 111000 cannot be distinguished from 0000111:



The only sequence which can be distinguished uniquely is a 0 between two 1's. It translates to a domain of width 2 units:



So the header of each disk sector must contain this synchronisation pattern.

FM encoding can be used to store the data at half this density on poorer quality medium.

Dinosaur

{ewc HelpAnim.dll,AnimWnd,1/dino.txt}

See also:

FM Encoding
MFM Encoding

Playing Sounds

The PLAYSND DLL allows you to play a WAV file from within a help file.

There are two functions in PLAYSND.DLL:

Function	description
PLAYSND	plays a WAV file that has been stored in the HLP file as "baggage"
PLAYFILE	plays a WAV file that is held on disk

PLAYSND

The DLL call

{dll=PLAYSND.dll,PLAYSND(S:qchPath,S:"sine.wav",u:0),play sound}

plays the file SINE.WAV where SINE.WAV has been stored in this HLP file. Try it:

play sound

PLAYSND takes two string and one unsigned integer parameter:

Parameter	description
HelpFile	the first parameter is the name of the help file which contains the sound. You can use the Predefined Variable qchPath which specifies the help file containing the current topic
SoundFile	the second parameter is the name of the sound file contained in the help file
Flags	sum of: 1: play asynchronously 8: loop the sound until next PLAYSND 16: don't stop any currently playing sound

If "play asynchronously" is specified then the player will start the sound and immediately continue on with the next command. Otherwise, the player will wait until the sound has finished before continuing.

EditHelp will include a file inside a help file (HLP) if the directory containing the source of your help file (EDH) also contains the file

BAGGAGE.EPJ

BAGGAGE.EPJ is a text file. Each line is the name of a file to be copied into the HLP file as "Baggage". When this help file was compiled, the BAGGAGE.EPJ file contained the text:

sine.wav

PLAYFILE

The DLL call

{dll=PLAYSND.dll,PLAYFILE(S:"sine.wav",u:0),play file}

makes the same sound but it reads it from the file SINE.WAV:

play file

PLAYFILE takes one string and one unsigned integer parameter:

Parameter	description
SoundFile	the name of the sound file
Flags	sum of: 1: play asynchronously 8: loop the sound until next PLAYSND 16: don't stop any currently playing sound