AN001: Read Sample Points From .IWF Files

This application note describes IWF file format, for users to extract sample points from the files for future analysis.

IWF File Format

offset (hex)	size (dec)	type	name	description
0	16	CHAR[16]	signature, starting	"Ideofy LA-08 000", note that there are two space characters (ascii 0x20).
10	4	UINT32	iwf version	iwf file format index, currently 0x00010000
14	4	UINT32	sw_version	version number of the LA08.EXE which created the file. 0x01090201, means v1.9.2.1
144	4	UINT32	sample_rate	sample rate in KHz
14C	4	UINT32	n_channel	number of channels, valid values are 8, 4, and 2.
154	4	UINT32	n_sample	number of sample points
158	4	UINT32	trigger_pos	trigger position (%), valid value 0 to 100
160	8	UINT8[8]	trigger_set[8]	trigger settings, one byte per channel, possible values are 0 to 5, indicating none, high, low, rising edge, falling edge, and either edge, respectively. trigger_set[0] is for channel 1.
168	256	CHAR[8][32]	ch_text[8]	text for channel 1 to 8
4C8	4	UINT8[4]	signature, ending	55, AA, 55, AA (hex)
4CC			sample_data	sample data starts here

* UINT8 means 8-bit unsigned int, UINT32 is 32-bits unsigned int, CHAR is 8-bit.

* Multi-byte data are stored in the little endian order; that is, the least significant bye (LSB) is stored first..

* [n] denotes an array of *n* elements. Multi-dimensional arrays are stored in row-major format [2].

Mapping a sample point to corresponding sample memory location

If 8 channels are enabled, each sample point has 8 bits, and the channel 8 bit is the msb (most significant bit). In the sample memory each byte contains a sample point.

If 4 channels are enabled, each sample point has 4 bits. In the sample memory each byte contains two sample points: the preceding sample point is stored in the higher 4 bits (bit 7..4).

If 2 channels are enabled, each sample point has 2 bits. In the sample memory each byte contains four sample points: the preceding sample point is stored in the higher 2 bits (bit 7,6).

For example, when two channels are enabled, sample point 12345 is stored at the bit 5 and bit 4, respectively of the 3086th byte in the sample memory. (12345/4=3086; 12345%4=1).

Run-length Encoding

The sample memory is stored in the IWF file using Run-Length Encoding (RLE) [1]. A run consists of bytes of the same value and is stored in the IWF file by two bytes: *value* and *length*. For example, if the sample memory consists of the following data bytes: 55,55,55,aa,aa,aa,aa,aa,aa,77,77,77,77,77,77,… will be RLE encoded into 55,3,aa,6,77,6,.. in the IWF file.

A sample C program is provided to decode the RLE data. Varibles *inf*, *acq_data*, *acq_size* need to be initialized before executing the sample program. Variable *inf* is a file pointer pointing to an opened IWF file; in addition, the read/write access location of *inf* has to be set to the beginning location of the RLE data, (i.e., offset 0x4CC). *acq_size* is the total number of bytes in the sample memory, which can be calculated by *acq_size* = *n_sample*/(8/*n_channel*). *acq_data* is a pointer to the sample memory (unsigned char array) which is no smaller than *acq_size*.

```
// FILE * inf;    // input file
// int acq_size   // number of data bytes
// unsigned char * acq_data;   // pointer to sample memory

int i, b, c, e, bc;

i = 0; bc = 0;
while(!feof(inf)) {
     b = fgetc(inf);
     c = fgetc(inf);
     i = 0; e = bc+c+1;
     if(b==EOF||c==EOF) break;
     if(feof(inf)) break;
     acq_data[bc++] = b;

     while(bc<e) {
          acq_data[bc++] = b;
          if(bc>=acq_size) break;
     }
}
```

Reference

[1] Wikipedia, *Run-length encoding*, http://en.wikipedia.org/wiki/Run-length_encoding

[2] Wikipedia, *Row-major order*, http://en.wikipedia.org/wiki/Row-major_order