Tucker-Davis Technologies TTank format:

Data collected or used by TDT software is stored in *tanks* that are logically divided into *blocks*. These blocks contain all the data from a single recording session such as an experimental run. Within each block different stores can record different types of events at different rates. This document is intended to provide the user with the details necessary to directly access data stored in these tanks.

The disk format for a TDT tank consists of a single folder (the tank directory) containing per-block subfolders. Hidden *desktop.ini* files are used to set the icon on all of these folders and as a means for TDT software to recognize tank folders. Within each block folder are four additional files each named according to "TANKNAME_BLOCKNAME.XXX" where XXX is one of the following extensions: TBK, TDX, TSQ, TEV.

The low level details of the TBK and TDX files are beyond the scope of this document. Briefly, the TBK contains block-level headers and offsets to the beginning of each second of data collection in the TSQ file. The TDX file contains epoch indexing information for faster access or filtering when dealing with epoch data. The remaining two files (the TSQ and TEV) contain the actual event headers and data meaning that usable TBK and TDX files can be reconstructed from these files with tools provided by TDT technical support if the TBK or TDX becomes lost or corrupted. The TSQ holds the header information for all the individual events whose data are in turn stored in the TEV.

The format of the TSQ file is simply a list of headers for each data segment or snippet stored in the TEV. Fields in the headers are little-endian, aligned at 32-bit boundaries (except for Sort Code), and are interpreted as follows:

Field	**Type**	**# Bytes**	**Comment**
Size	`long`	4	Size of data, with header, in longs.
Type	`long`	4	Snip, stream, scalar, etc.
Code	`long`	4	4-character event name (cast as long)
Channel	`unsigned short`	2	
Sort Code	`unsigned short`	2	Sort code for snip data.
Time Stamp	`double`	8	Event start time in seconds.
Event Offset or Strobe	`__int64 or double`	8	Offset in TEV file or raw value.
Data Format	`long`	4	float, long, etc.
Frequency	`float`	4	Sampling frequency

This results in each header being 10 words, or 40 bytes, long. This constant size allows for faster random access within the TSQ files. Data in the TEV file can be variable length, for example if different types of data were being recorded into the same block.

The following C code block explains the above through an example:

```
// Demo program showing how to read tsq file and tev file
#include <stdio.h>

// tank file structure
//
// tbk file has block events information and on second time offsets
// to efficiently locate events if the data is queried by time .
//
// tsq file is a list of event headers, each 40 bytes long,
// ordered strictly by time .
//
// tev file contains event binary data
//
// tev and tsq files work together to get an event's data and
//     attributes
//
// tdx file contains just information about epoc stores,
// is optionally generated after recording for fast retrieval
// of epoc information


/* names and paths used in the code snippet to locate the tank files */
#define TANK_NAME "DEMOTANK2"
#define BLOCK_NAME "Block-3"
#define EV_NAME "LFPs"
#define CHANNEL 1
#define TANK_PATH "C:\\TDT\\OpenEx\\Tanks\\" TANK_NAME "\\" BLOCK_NAME "\\"

/* TTank event header structure */
struct tsqEventHeader
{
    long           size;
    long           type;      /* event type: snip, pdec, epoc etc */
    long           code;      /* event name: must be 4 chars, cast as a long */
    unsigned short channel;   /* data acquisition channel */
    unsigned short sortcode;  /* sort code for snip data */
    double         timestamp; /* time offset when even occurred */
    union
    {
        __int64    ev_offset; /* data offset in the TEV file */
        double     strobe;    /* raw data value */
    };
    long           format;    /* data format of event: byte, short, float
(usually), double */
    float          frequency; /* sampling frequency */
};

int main(int argc, char* argv[])
{
    FILE *tsq = NULL, *tev = NULL, *out = NULL;
    char bev[10000];
    tsqEventHeader bsq;

    tsq = fopen(TANK_PATH TANK_NAME "_" BLOCK_NAME ".tsq", "r+b");
    tev = fopen(TANK_PATH TANK_NAME "_" BLOCK_NAME ".tev", "r+b");
    out = fopen(TANK_PATH TANK_NAME "_" BLOCK_NAME ".csv", "w");

    if (tsq != NULL && tev != NULL && out != NULL)
        while(fread(&bsq, sizeof(bsq), 1, tsq) == 1)
```

```
            if ( bsq.code == *(long*)EV_NAME && bsq.channel == CHANNEL)
            {
                long nPts = bsq.size - 10;
                fsetpos(tev, &bsq.ev_offset);
                fread(bev, nPts <<2, 1, tev);
                for(long i = 0; i < nPts; i ++)
                    fprintf(out, "%E,", ((float*) bev)[i]);
                fprintf(out, "\r\n");
            }

    if (tsq)
        fclose(tsq);
    if (tev)
        fclose(tev);
    if (out)
        fclose(out);

    return 0;
}
```