RADAN DZT File Format

The RADAN DZT file is a binary file containing a header followed by scans of data. The header contains information concerning the settings used to collect the data and the header size.

Note: This information is provided to the User for informational use only. It is not supported by GSSI technical support and is provided for those Users who are proficient in working in a C programming environment.

A. Internal structures

```
struct tagRFDate // File header date/time structure

{

unsigned sec2 : 5;              // second/2 (0-29)

unsigned min : 6;                       // minute (0-59)

unsigned hour : 5;              // hour (0-23)

unsigned day : 5;                       // day (1-31)

unsigned month: 4;              // month (1=Jan, 2=Feb, etc.)

unsigned year : 7;              // year-1980 (0-127 = 1980-2107)

};

struct tagRFCoords // Start/End position

{

float rh_fstart;

float rh_fend;

};

struct RGPS // GPS record/system time SYNC

{

char RecordType[4];             // “GGA”

DWORD TickCount;                // CPU tick count

double PositionGPS[4];          // Latitude (positive if 'N'), Longitude (positive if ‘E’),

// Altitude, FIXUTC

};
```

B. Constants and macros

```
// constants
const int MINHEADSIZE = 1024;
const int PARAREASIZE = 128;
const int GPSAREASIZE = 2 * sizeof(RGPS);
const int INFOAREASIZE (MINHEADSIZE - PARAREASIZE- GPSAREASIZE) ;
// structure member alignment macros
#define TYPEBYTE(x,n) BYTE x##[n]
#define SHORTBYTE(x) TYPEBYTE(x,2) // short int (16 bit)
#define FLOATBYTE(x) TYPEBYTE(x,4) // float
#define RFDATEBYTE(x) TYPEBYTE(x,4) // tagRFDate
#define COORDBYTE(x) TYPEBYTE(x,8) // tagRFCoords
```

C. Radan Header structure

```
struct tagRFHeader
{                               // Offset in bytes
short rh_tag;                   // 0x00ff if header, 0xfnff for old file        00
short rh_data;                  // Offset to Data from beginning of file        02
                                        //    if rh_data < MINHEADSIZE then
                                        //    offset is MINHEADSIZE * rh_data
                                        //    else offset is MINHEADSIZE *rh_nchan
short rh_nsamp;                         // samples per scan                     04
short rh_bits;                  // bits per data word (8,16, 32) *              06
short rh_zero;                  // if rh_system is SIR-30 or UtilityScan DF     08
                                // then equals repeats/sample
// otherwise is 0x80 for 8 bit data and
// 0x8000 for 16 bit data
FLOATBYTE(rhf_sps);             // scans per second                             10
FLOATBYTE(rhf_spm);             // scans per meter                              14
FLOATBYTE(rhf_mpm);              // meters per mark                             18
FLOATBYTE(rhf_position);        // position (ns)                                22
FLOATBYTE(rhf_range);           // range (ns)                                   26
short rh_npass;                 // num of passes for 2-D files                  30
RFDATEBYTE(rhb_cdt);            // Creation date & time                         32
RFDATEBYTE(rhb_mdt);            // Last modification date & time                36
short rh_mapOffset;             // For internal use                             40
short rh_ mapSize;                    // For internal use
                    42
short rh_text;                        // offset to text                      44
short rh_ntext;                       // size of text                        46
short rh_proc;                         // offset to processing history       48
short rh_nproc;                       // size of processing history          50
short rh_nchan;                       // number of channels                  52
FLOATBYTE(rhf_epsr);                  // average dielectric constant         54
FLOATBYTE(rhf_top);                   // position in meters                  58
FLOATBYTE(rhf_depth);                 // range in meters                     62
COORDBYTE(rh_coordX);                 // X coordinates                       66
FLOATBYTE(rhf_servo_level);           // gain servo level                    74
char reserved[3];                     // reserved                            78
BYTE rh_accomp;                       // Ant Conf component                  81
short rh_sconfig;                     // setup config number                 82
short rh_spp;                         // scans per pass                      84
short rh_linenum;                     // line number                         86
COORDBYTE(rh_coordY);                 // Y coordinates                       88
BYTE rh_lineorder:4;                  //                                     96
BYTE rh_slicetype:4;                  //                                     96
char rh_dtype;                         //                                    97
char rh_antname[14];                  // Antenna name                        98
BYTE rh_pass0TX:4;                    // Activ Transmit mask                 112
BYTE rh_pass1TX:4;                    // Activ Transmit mask                 112
BYTE rh_version:3;                    // 1 – no GPS; 2 - GPS                 113
BYTE rh_system:5;                      // (see below for description)**      113
char rh_name[12];                     // Initial File Name                   114
short rh_chksum;                      // checksum for header                 126
char variable[INFOAREASIZE];          // Variable data                       128
RGPS rh_RGPS[2];                      // GPS info                            944
}; // End of tagRFHeader
```

*Data format is little-endian. Eight byte and sixteen byte samples are unsigned integers. Thirty-two bit samples are signed integers.

**rh_system values:

Control Unit	Number
SIR 2000	2
SIR 3000	3
TerraVision	4
SIR 20	6
StructureScan Mini	7
SIR 4000	8
SIR 30	9
UtilityScan DF	12
HS	13
StructureScan Mini XT	14