



SeaSonde Radial Site Release 6

Time Series File Format

Apr 19, 2009 ©Copyright CODAR Ocean Sensors

TimeSeries files are a collection consecutive time sweeps consisting of received signal power over time. The data is obtained by SeaSondeAcquisition by collecting a consecutive series of measurements from the receiver. The TimeSeries can later be imported by SeaSondeAcquisition to re-examine and further processing into range and cross spectra data.

File Name Format

"Lvl_XXXX_yyyy_mm_dd_hhmmss.ts"
where XXXX = four char code site name
where yyyy = created year ei 2009
where mm = created month 01 to 12
where dd = created day 01 to 31
where hh = created hour 00 to 23
where mm = created minute 00 to 59
where ss = created second 00 to 59

File Contents

Format is Resource Indexed File Format. The file is composed of keyed blocks of binary data where each block starts with a 4byte character type code followed by a 4byte long data size of how much data follows.

Big-Endian Byte ordering (MSB first)
IEEE floats & doubles
Two's complement integer values

The file is compose of multiple keys where each key consists of:
A 4 byte character key type code
A 4 byte integer of key data size (can be zero)
Followed by the key data, which is the data size length of bytes.

By convention, Keys with all CAPITALS have subkeys, meaning that the key's data is made up of more keys. When you read a subkey you should read the data in the key as more RIFF keys.

A key may have no data (zero size), in which case the key will contain only the type code and the zero value key size.

When Reading

If you do not recognize the key you should usually skip over it by doing a dummy read of the key's data size. Do not expect the keys to be in order unless implicitly stated.
Keys can be repeated as needed describing new or changed information.

If you read this file on an Intel Platform or other which uses Little Endian byte ordering the first four bytes will be '\VLQA'. In which case, you will need to swap the byte order on each value.

If the file has not finished writing or was closed improperly, the first riff key 'AQLV' will contain an invalid data size of 0xFFFFFFFF or ((UInt32)-1). You can then decide if you want to continue reading the partial file or skip it.

SeaSondeAcquisition writes the number of sweeps set (normally 512) of continuous time sweeps of data. If the sweep information changes or the next sample sweep is not continuous, the file will contain less than this number of sweeps.

When reading and processing consecutive files, you must verify that they are also consecutive in time.

Data Type Definitions

Fourcc	4bytes four character code (example 'xxxx')
Char	1byte char
LString	#bytes, string
Char[64]	64bytes, string, zero terminated
Char[]	[]bytes from key data size, zero terminated string
SInt8	1byte Signed -128 to +127 (2s Complement)
UInt8	1byte Unsigned 0 to 255
SInt16	2byte Signed -32768 to 32767(2s Complement)
UInt16	2byte Unsigned 0 to 65535
SInt24	3byte Signed (2s Complement)
SInt32	4byte Signed -2Giga to +2Giga (2s Complement)
UInt32	4byte Unsigned 0 to 4 Giga
Float	4byte IEEE single precision floating point
Double	8byte IEEE double precision floating point
Size32	4byte Unsigned 0 to 4 Gigabytes (tells how much data follows key)

Data Scaling

If the 'fbin' key data type is of 'fix4', 'fix3', or 'fix2' then the Range data is auto scaled to an integer value. The scalars used come from the 'scal' keys.

The default format used by SeaSondeAcquisition is 'fix2'.

If using fixed type 'fix4' then:

```
double IValue = (double)IntegerIValue / (double)0x7FFFFFFF * scalarOne;  
double QValue = (double)IntegerQValue / (double)0x7FFFFFFF * scalarTwo;
```

If using fixed type 'fix3' then:

```
double IValue = (double)IntegerIValue / (double)0x7FFFFFF * scalarOne;  
double QValue = (double)IntegerQValue / (double)0x7FFFFFF * scalarTwo;
```

If using fixed type 'fix2' then:

```
double IValue = (double)IntegerIValue / (double)0x7FFF * scalarOne;  
double QValue = (double)IntegerQValue / (double)0x7FFF * scalarTwo;
```

File Contents Layout

Each subkey contents is inside of {} brackets

Each key data content is indented in order after key.

```

// Begin File. The first 4bytes should read 'AQLV'
'AQLV' Size32 - This is the first key in the file. All data is inside this key.
{
  'HEAD' Size32 - Data Description Section
  {
    'sign' Size32 - File signature
      Fourcc   File version   '1.00'
      Fourcc   SiteCode      'XXXX'
      Fourcc   FileType      'ALVL'
      UInt32   UserFlags     0
      chr64    FileDescription "SeaSonde Acquisition Time Series"
      chr64    OwnerName     "CODAR Ocean Sensors Ltd"
      chr64    Comment       ""
    'mcda' Size32 - Mac Timestamp of first sweep
      UInt32   Seconds from Jan 1,1904
    'cnst' Size32 - Size information
      SInt32   <nChannels>   Number of Antennas/Channels (Normally 3)
      SInt32   <nSweeps>     Number of Sweeps Recorded (Normally 32)
      SInt32   <nSamples>    Number of Samples Per Sweep (Normally 2048)
      SInt32   IQ Indicator.  1 otherwise 2 if sample data is IQ
    'swep' Size32 - Sweep information
      SInt32   Number of Samples Per Sweep (Normally 2048)
      double   Sweep Start Frequency in Hz
      double   Sweep Bandwidth in Hz (maybe negative)
      double   Sweep Rate in Hz
      SInt32   RangeCell Offset (Not used)
    'fbin' Size32 - Sample Data Type
      Fourcc   format. Normally 'cviq' indicating complex voltage I & Q
      Fourcc   Type of 'alvl' data ('flt4', 'fix4', 'fix3', 'fix2')
  } // End of HEAD
  'BODY' Size32 - This key contains the repeated keys for each sweep recorded.
  {
    It normally contains a list of 'indx', 'scal', 'alvl' keys for each sweep.
    'indx' Size32 - This key helps to index the current sweep.
      SInt32   Sweep Index from zero to <nSweeps>-1
    'scal' Size32 - This key tells how to scale following 'alvl' sample data to get voltage.
      double   ScalarOne     I scale value.
      double   ScalarTwo     Q scale value.
    'alvl' Size32 - The sample data for a single sweep
      Array of SInt16 of IQ pairs <nSamples> long.
      <ISample#0><QSample#0><ISample#1><QSample#1>....
      <ISample#(nSamples-1)><QSample#(nSamples-1)>
  } // Repeat these keys for each sweep. End Of BODY
} // End of AQLV
'END ' Size32 - End of File key
// End Of File

```