Bathyswath File Formats

1 Introduction

1.1 Document Change Record

Version	Date	Changes
07.00	28/02/13	Bathyswath version, derived from “SWATHplus File Formats” rev 6E
07.01	22/08/13	Corrected description of ping position in SXP
07.02	03/09/13	Reserved a set of block IDs for client use

See section 8 for the full history of this document.

1.2 Scope

This document describes the format and interpretation of the data files written by the Bathyswath and SWATHplus sonar systems.
There are several file types written by the software, all using the same format. See “File Types” below.
These data files are written onto the PC’s hard disk by the sonar software and contain all the information recorded by the system during the survey.

1.3 Context

Bathyswath is a swath bathymetry sonar system. It is derived from the SWATHplus sonar system, and uses the same file formats. In turn, SWATHplus was derived from the Submetrix sonars, built by Submetrix Ltd.

1.4 Glossary

Bathyswath	A seabed mapping sonar system. Also the name of the organisation that builds and sells it
Swath	The Bathyswath Swath Processor software application
SWATHplus	The forerunner of the Bathyswath system

1.5 Table of Contents

2 File Types

The following file types are described in detail in this document. They all use the Bathyswath block-orientated format.

Type	Format	Suffix	Contains	Notes	See Section
Raw sonar data	binary	.sxr	All the data collected by the real-time system	Processed off-line to produce one or more of the processed data file types.	2
Coverage maps	binary	.swc	Coverage data used by the coverage plot view	These files are a summary of the data coverage, but contain no sonar data, so are not commonly read by third-party software.	
Processed sonar data	binary	.sxp	Processed data derived from the real-time software	These files have all corrections applied, including: attitude, position, tide, speed of sound. Includes down-sampled position, attitude and tide information. Processed data files can be created from raw data files or parsed data files.	3
Parsed data	binary	.sxi	Raw data, but parsed into a format that is easier for third-party code to interpret.	These files have none of the above corrections applied. Parsed data files can be created from raw data files, but not the other way around.	0

The following file types are used by the Bathyswath and SWATHplus software, but do not use the block-orientated format and are not easily read by third-party software:

Type	Format	Suffix	Contains
Swath processor session file	MFC “serialization” file	.sxs	The set-up of the Swath Processor program, including the configuration of the sonar, input port configurations, filter settings and view window parameters.
Grid file	Binary format, followed by MFC “serialization” data	.sxg	The grid data, in an open binary format, followed by the set-up of the Grid Processor, including filter settings and view window parameters.

3 Data Storage and Transmission

3.1 Disk Format

The data files are written using a Microsoft Windows operating system and therefore follow the conventions of that system in terms of file naming and low-level disk format.

3.2 File Names

The file name is specified by the user during the operation of the real-time sonar software. The extension is supplied by the software according to the file type (see the table above).

3.3 Endianness

All Bathyswath and SWATHplus data is little-endian, i.e. in the natural 80x86 format with the least significant byte at the lower address.

3.4 Real-Time TCP/IP Data Transmission

Bathyswath software can output data in real time, and be controlled, over a TCP/IP interface (for example, using an Ethernet wired or wireless LAN). This interface uses the same block-orientated structure as the data files.
The Bathyswath Swath Processor application can output the Parsed Data format (section 0) and the Raw Data format (section 2) by TCP/IP in real time.

1. Block-Orientated Data Format

1.1 General

All of the files listed in the first table of section 2 use the same block-orientated data format. They can be read using the same software code, and the blocks that they contain may be included in any of the files. The difference between these file types is therefore simply the types of data block that they tend to contain.
Each file contains a file header block, followed by a series of data blocks.
Every block contains a header that identifies the block, followed by the length of the block. Therefore, the reading software can identify the blocks that it wishes to read, and ignore and skip over any other block type that it encounters. In this way, new blocks can be added to a file without necessarily having to update the reading software.

1.2 File Header

A file header is used to identify each file type. It is formatted in the same way as data blocks, but with a "magic number" as the block type. Each file type uses a different magic number. See the table below.
However, the file header may not be present in some circumstances, and the file may start immediately with data blocks.
This magic number appears in the file as the sequence of bytes. The second 32-bit integer is the block length, which is currently set to a value of 8 bytes. The content of the header block is two 32-bit integers representing the software version number and the file format version number.
The software version number is encoded as an integer, as follows: (Major version- Minor version- Release- Build). For example, a version number of 3065601 means: Major version 3, Minor version 06, Release 56, Build 01.
The file format version number is now obsolete: use the data block identifiers as a way of checking file versions. For example, Swath version 3.7 writes "`SONAR_DATA3`" in its raw data files, and version 3.6 writes "`SONAR_DATA2`" blocks. To allow an application to read both kinds of file, add parsing code for both block types.

1.2.1 Magic Numbers

The file type magic numbers are:

File type	Magic number identifier	Magic number (hexadecimal)
Raw sonar data	SXR_HEADER_DATA	0xbad0bad0
Configuration data	SXC_HEADER_DATA	0xf1c0f1c0
Coverage maps	SWC_HEADER_DATA	0xc311c311
Processed sonar data	SXP_HEADER_DATA	0x01df01df
Grid data	SXG_HEADER_DATA	0xd1edede0
Parsed data	SXI_HEADER_DATA	0x521d52d1

1.3 Data Blocks

Data is stored in blocks; each block has a header consisting of type and length. Block types are 32-bit integer values encoded as follows.
These data blocks can occur in any Bathyswath block-encoded data files. They are also used in TCP/IP communications between applications. However, blocks of a certain type are most commonly found in particular files, and these file types are listed in the table below.

Type	Value	Notes	Usual File Type
SONAR_DATA	0x00	Written by version 2 code	Raw (sxr)
SONAR_DATA2	0x16	Replacement raw data type, used in version 3 code until May 08	sxr
SONAR_DATA3	0x17	Replacement raw data type, used in version 3 code from May 08	sxr
COMPASS_DATA	0x01	Obsolete	sxr
MRU_DATA	0x02	Obsolete	sxr
GPS_DATA	0x03	Obsolete	sxr
HWARE_DATA	0x04	Not used	sxr
FILE_DATA	0x05	Not used	sxr
GUI_DATA	0x06	Not used	sxr
NET_DATA	0x07	Not used	sxr
COMPASST_DATA	0x08	Timestamp + ASCII string	sxr
MRUT_DATA	0x09	Timestamp + data from instrument, in the instrument's native format: may be ASCII string or binary	sxr
GPST_DATA	0x0a	Timestamp + data from instrument, in the instrument's native format: may be ASCII string or binary	sxr
AUX1_DATA	0xb	Auxiliary port data (obsolete)	sxr
AUX2_DATA	0x0f	Auxiliary port data, channel 2 (obsolete)	sxr
AUX2_DATA	0x0f	Auxiliary port data	sxr
AUX1T_DATA	0xc	Auxiliary port data, Timestamp + ASCII string	sxr
AUX2T_DATA	0x10	Auxiliary port data, Timestamp + ASCII string	sxr
AUX3T_DATA	0x61	Auxiliary port data, Timestamp + ASCII string	sxr
AUX4T_DATA	0x62	Auxiliary port data, Timestamp + ASCII string	sxr
AUX5T_DATA	0x63	Auxiliary port data, Timestamp + ASCII string	sxr
AUX6T_DATA	0x64	Auxiliary port data, Timestamp + ASCII string	sxr
AUX7T_DATA	0x65	Auxiliary port data, Timestamp + ASCII string	sxr
PHCAL_DATA	0x0d	Phase calibration offsets	sxr
CNF_SENSOR_CORR	0x20	Sensor corrections	sxc
CNF_SENSOR_FILT	0x21	Sensor filters	sxc
CNF_SENSOR_INTERP	0x22	Sensor interpolation	sxc
CNF_DERIVE_ATT	0x23	Attitude derivation	sxc
CNF_ENVIR	0x24	Environment parameters	sxc

Type	Value	Notes	Usual File Type
CNF_DERIVE_POSN	0x25	Position derivation	sxc
CNF_TOW_POSN	0x26	Tow offsets: towed vehicles	sxc
CNF_POSN_OFFSETS	0x27	Offsets between sensors	sxc
SBP_XYZA_PING	0x28	Processed ping data. Written by version 3 code up to January 2010, version 3.6.N	sxp
SBP_XYZA_PING2	0x52	Replacement processed ping data type. Written by version 3.7 onwards.	
COVRG_DATA	0x0e	Coverage map data	swc
TEXT_DATA	0x11	Text message	TCP/IP
SYSTEM_COMMAND_DATA	0x12	Command from one system to another. See §	TCP/IP
TIME_SYNCH_DATA	0x13	Time synchronisation between systems	TCP/IP
PARSED_PING_DATA	0x29	Sonar data in parsed data	sxi & TCP/IP
PARSED_ATTITUDE	0x2b	Attitude data in parsed data	sxi & TCP/IP
PARSED_POSITION_LL	0x2C	Lat-long position data in parsed data	sxi & TCP/IP
PARSED_POSITION_EN	0x2d	Easting-Northing position in parsed data	sxi & TCP/IP
PARSED_SVP	0x2e	Speed of sound data in parsed data	sxi & TCP/IP
PARSED_ECHOSOUNDER	0x2f	Echosounder data in parsed data	sxi & TCP/IP
PARSED_TIDE	0x30	Tide data in parsed data	sxi & TCP/IP
PARSED_AGDS	0x31	AGDS data in parsed data	sxi & TCP/IP
CMS_CMD	0x40	Commands in	TCP/IP
CMS_STATUS	0x41	Status out	TCP/IP
AUX_ATTPOS	0X42	Attitude and position	TCP/IP
PARSED_FILTER_CMD	0x43	Controls for Parsed data filtering	TCP/IP
SBP_PROJECTION	0x50	Information on the projection used in processing (PLACEHOLDER)	sxp
SBP_PROCESS_INFO	0x51	Information on the processes used in processing (PLACEHOLDER)	sxp
Reserved	0x100 – 0x1ff	Reserved for client use; contact Bathyswath for details	

At present, only some of these block types are used. All other possible block types are reserved for future expansion. Types COMPASS_DATA, MRU_DATA and GPS_DATA are considered obsolete. Data blocks are concatenated with no further padding and in no particular order (the header record is, however, always the first record in the file).

1.3.2 Block Length

Immediately following the block type is the block length, again as a 32-bit integer. The block length is the number of bytes in the block, not including the header.

2. Raw Data File Blocks

2.1 General

Raw data files are written with the file extension “SXR”. They contain the following data blocks.
Note that some of the data block types are now obsolete. For example, the current distribution of Bathyswath software writes the sonar data in the “SONAR_DATA3” format. The “SONAR_DATA2” and “SONAR_DATA” block types only need to be decoded if data files written before 2009 need to be decoded.

2.2 Sonar Data Block, “SONAR_DATA3” (Code from May 2009)

This version is written by Bathyswath and SWATHplus code distributed after May 2009.
Data within the sonar data block consists of a header followed by the raw sonar samples. The header is always 49 bytes long.

The header information is:

Byte num	Num bytes	Item	Data Type	Code	Notes
0	4	Ping number	`long int`	`pingNum`	Unique in each survey session
3	1	Transducer channel	`unsigned char`	`tdrChannel`	Up to 4 channels active at any one instant (selected from any number of connected TEMs)
	1	FPGA code version	`unsigned char`	`fpgaIdent`	
	1	Transducer type	`unsigned char`	`tdrType`	See “Transducer type Identifiers” below
	1	Board type	`unsigned char`	`boardType`	See “Board type Identifiers” below
	8	Board identifier string	`char string`	`boardIdent`	In effect, the serial number. Often only the first two bytes populated
	4	Operating frequency	`float`	`operatingFreq`	See “Transducer Frequencies” below
	4	Hardware Gain	`float`	`analogueGain`	Not used on current boards
	1	Phase clock full scale	`unsigned char`	`noClocksIn360`	256 in most boards

Byte num	Num bytes	Item	Data Type	Code	Notes
	1	Error byte	unsigned char	error	0 = no error
	1	Calibration	unsigned char	calBit	Set 1 if board is in calibration mode
	1	Transmit power code	unsigned char	txPower	
	2	Transmit pulse length	short int	txCycles	In sonar cycles
	2	Samples in ping	short int	rxSamples	Number of samples in this ping
	1	Interval between samples	unsigned char	rxPeriod	In microseconds
	1	Code for which ADC channels are enabled	unsigned char	sidescanAdcEnable	Bit code for four possible channels, A, B, C, D
	4	Acquisition time: seconds	long int	timeSecPC	PC clock time
	2	Acquisition time: milliseconds	short int	timeMsecPC	PC clock time
	4	Sonar clock time: seconds	long int	timeSecSonar	TEM clock time
	2	Sonar clock time: milliseconds	short int	timeMsecSonar	TEM clock time
	1	Position in alternating and simultaneous scans	unsigned char	firstInScan	Used with alternating and simultaneous transmission: first = always 1; alternating or double-sided goes 1,0,1,0 …
	2	Spare bytes		spare	For expansion

The header is followed by a number of sample information items. Each item is 8 bytes long. The number of items is given by the "rxSamples" field of the sonar data header, or by reading the block size, subtracting the size of the sonar data header (49), and dividing by the size of the sample item (8).

The sample information is:

Byte num	Num bytes	Item
0	1	Phase AB
1	1	Phase AC
2	1	Phase AD
3	1	Transducer number
4	2	Sample number LSB
6	2	Amplitude

Note: the phase has the opposite sign in the new SONAR_DATA2 format to that in SONAR_DATA

2.2.3 Board type Identifiers

These codes are used to identify board types.

Value	Name	Description	Notes
1	BRD_TYPE_117_Q0	Quicklogic based 64 way DIN41612 interface (117kHz only)	Development only: shouldn't find in the field
2	BRD_TYPE_117	Quicklogic based 37 way D connector interface 117KHz	
3	BRD_TYPE_ISA	16-Bit ISA Board	Not a TEM. Not used in USB TEM systems.
4	BRD_TYPE_234	Quicklogic based 37 way D connector interface 234KHz	
5	BRD_TYPE_117_A	Altera based 37 way D connector interface 117KHz	
6	BRD_TYPE_234_A	Altera based 37 way D connector interface 234KHz	
7	BRD_TYPE_468_A	Altera based 37 way D connector interface 468KHz	
8	BRD_TYPE_USB_468	Altera based, USB interface TEM, 468KHz	

2.2.4 Transducer type Identifiers

These codes are used to identify the sonar frequency of the transducers. They correspond to a frequency code that is hard-wired into each transducer's connector.

Value	Name	Nominal Freq. /kHz	Actual Frequency /Hz	Bit Code	Notes
10	TXD_TYPE_117	117	117187.5	"1010"	
5	TXD_TYPE_234	234	234375	"0101"	
13	TXD_TYPE_468	468	468750	"1101"	
15	TXD_TYPE_NO_CONN	-	-	"1111"	No transducer connected to TEM

Note that the hard-wired transducer codes are being removed from the later Bathyswath hardware, so this field cannot be assumed to be provided in later systems.

2.2.5 Transducer Frequencies

These are binary divisions of 30 MHz.

Name	Value
TXD_FREQ_117	117.1875e3
TXD_FREQ_234	234.375e3
TXD_FREQ_468	468.750e3

2.2.6 FirstInScan Field

- In both simultaneous mode and alternating mode, with a two-TEM system (port and starboard), the file consists of a series of ping records. These will alternate port-starboard-port-starboard. In an ISA TEM system, channel 1, nominally the port TEM, will have firstInScan set to 1, and the starboard TEM firstInScan will be set 0. In the new USB system, the "first" TEM is not necessarily the port one, as there is no guarantee which one the USB driver sees first.
- If you have more than two TEMs fitted and working at the same time (alternating or simultaneous), then just one of the TEMs will have firstInScan set.
- In single-sided mode, firstInScan is always set.

2.3 Sonar Data Block, "SONAR_DATA2" (v3 code to May 08)

This version is written by version 3 SWATHplus code, distributed between September 2006 and May 2008.
Only the ping header is different from SONAR_DATA3: the sonar samples are the same.

Byte num	Num bytes	Item	Data Type	Code	Notes
0	2	Ping number	short int	pingNum	Unique in each survey session
3	1	Transducer channel	unsigned char	tdrChannel	Up to 4 channels active at any one instant (selected from any number of connected TEMs)
	1	FPGA code version	unsigned char	fpgaIdent	
	1	Transducer type	unsigned char	tdrType	See "Transducer type Identifiers" below
	1	Board type	unsigned char	boardType	See "Board type Identifiers" below
	8	Board identifier string	char string	boardIdent	In effect, the serial number. Often only the first two bytes populated
	4	Operating frequency	float	operatingFreq	See "Transducer Frequencies" below
	4	Hardware Gain	float	analogueGain	Not used on current boards
	1	Phase clock full scale	unsigned char	noClocksIn360	256 in most boards

Byte num	Num bytes	Item	Data Type	Code	Notes
	1	Error byte	unsigned char	error	0 = no error
	1	Calibration	unsigned char	calBit	Set 1 if board is in calibration mode
	1	Transmit power code	unsigned char	txPower	
	2	Transmit pulse length	short int	txCycles	In sonar cycles
	2	Samples in ping	short int	rxSamples	Number of samples in this ping
	1	Interval between samples	unsigned char	rxPeriod	In microseconds
	1	Code for which ADC channels are enabled	unsigned char	sidescanAdcEnable	Bit code for four possible channels, A, B, C, D
	4	Time in seconds	long int	timeSec	
	2	Millisecond component of time	short int	timeMsec	
	1	Position in alternating and simultaneous scans	unsigned char	firstInScan	Used with alternating and simultaneous transmission: first = always 1; alternating or double-sided goes 1,0,1,0 …
	2	Spare bytes		spare	For expansion

2.4 Sonar Data Block, "SONAR_DATA" (v2 code)

This version is written by version 2 SWATHplus code, distributed before September 2006. Data within the sonar data block consists of a header followed by the raw sonar samples. The header is always 160 bytes long – eight 32-bit integers followed by 16 8-byte structures. The following code defines the structure of a sonar header:

```
const int MAX_TX        = 15;
const int MAX_TX_SLOTS  = MAX_TX + 1;

class TxBoardInfo {
public:
   unsigned char tdcrtype;       // Register 0 bits [3:0]
   unsigned char ctrlreg;        // Register 5
   unsigned short int txcycles;  // Register 6 * 8
   unsigned short int rxsamps;   // Register 7 * 256
   unsigned char rxrate;         // Register 8
   unsigned char analch;         // Register 9
```

```
   TxBoardInfo ();
};

struct BathyIO {
   int version;
   int ping;
   int txok;
   int active;
   int sec;
   int usec;
   int spare1;
   int spare2;
   TxBoardInfo regs[MAX_TX_SLOTS];
};
```

The `BathyIO` structure contains a version number, ping number, transmit OK flag, active channel number and a timestamp. The version number is currently always set to one. The ping number increments as pings are generated. The transmit OK flag is currently always set to one. The active channel will be set to one for a port side ping and two for a starboard ping. The timestamp is represented as two integers: seconds since 1970 and microseconds since that second. The two spare fields are currently set to hexadecimal constants.

The `TxBoardInfo` class holds copies of the register values that were used to generate the ping.

Following the sonar header is the block of raw sonar samples. There are as many raw sonar samples as the sonar generated during the ping, as controlled by the 'Number of Receive Samples' register. Each sample has the following format:

```
class BathySample {
public:
   unsigned char ab;
   unsigned char ac;
   unsigned char ad;
   unsigned char txno;
   unsigned char samp0;
   unsigned char samp1;
   unsigned char anal0;
   unsigned char anal1;

   BathySample ();
};
```

The three bytes `ab`, `ac` and `ad` represent the phase differences between staves A-B, A-C and A-D. The `txno` byte represents the transducer channel number along with some flags. The pair of bytes `samp0` and `samp1` forms a 16-bit value for the sample number (zero-based). The pair of bytes `anal0` and `anal1` form a 12-bit signed value for the analogue signal amplitude.

The bytes `samp0` and `anal0` are the least-significant bytes, while `samp1` and `anal1` are the most-significant. To obtain the timestamp for a given sample, multiply the sample number by the sample rate, in microseconds. The sample rate is available as one of the register values mentioned earlier. This timestamp represents the round-trip time from sonar transmit to sample reception. The analogue value represents the received sonar signal amplitude at the moment of sampling. Minimum amplitude has a value of -4096 and maximum +4095.

2.5 Obsolete, non-timestamped Compass, MRU and GPS Data Block Format

All the non-timestamped string-type data blocks share the same basic format. The ASCII string simply follows the type/length header. The string is followed by a timestamp in ASCII format, consisting of seconds and microseconds since 1970.
This string-based timestamp was a temporary kludge and has been replaced by a binary-coded timestamp in the current version of this format. These records are no longer generated.

2.6 Timestamped Compass, MRU and GPS Data Block Format

All the timestamped string-type data blocks share the same basic format. The header is followed immediately by an eight-byte timestamp, organised as two four-byte integers. The first integer represents seconds since 1970, and the second integer represents microseconds since that second. The ASCII string follows immediately after the timestamp, but is not null-terminated. Therefore, the length of the data block is the length of the ASCII string plus eight.

2.7 Auxiliary Port Data

The Swath Processor program supports eight "auxiliary" inputs. These inputs can be configured to receive data from a range of serial input devices. The data from these channels are stored in the time-stamped data types AUX1T_DATA (for the first channel), AUX2T_DATA (for the second), etc. There is currently no way of knowing which data type has been stored in these auxiliary ports direct from the raw data files. The Swath Processor session file (sxs) stores the configuration of the ports.
Each data item consists of an 8-byte timestamp, followed an ASCII string. The data in the string will depend on the device that supplied the data.
Data that may be present includes:

Type	Sxs Code	Value
Raw ASCII data	FORMAT_AUX_RAW	1
Position data	AUX_INPUT_TYPE_POSITION	2
Heading data	AUX_INPUT_TYPE_HEADING	3
Motion sensor data	AUX_INPUT_TYPE_MRU	4
Speed of sound data	AUX_INPUT_TYPE_SVP	5
Echosounder data	FORMAT_AUX_ECHOSOUNDER	6
Tide data	AUX_INPUT_TYPE_TIDE	7
Acoustic Ground Discrimination System (e.g. ECHOplus)	AUX_INPUT_TYPE_AGDS	8
Commands between systems	AUX_INPUT_TYPE_COMMAND	9
Pressure sensor	AUX_INPUT_TYPE_PRESSURE	10
Cable out sensor	AUX_INPUT_TYPE_CABLE_OUT	11
CTD sensor	AUX_INPUT_TYPE_CTD	12
GPS	AUX_INPUT_TYPE_GPS	13
Magnetometer	AUX_INPUT_TYPE_MAGNETOMETER	14
Light Sensor	AUX_INPUT_TYPE_LIGHT_SENSOR	15

2.8 Phase Calibration offsets

The phase calibration block, PHCAL_DATA, contains a set of numerical offsets that need to be added to every phase value in the raw data file before phase to angle conversion is done.
The data area in the block consists of an array of C++ structures:

```
CPhaseOffsets m_phaseOffsets[MAX_TX_SLOTS];
```

Where `CphaseOffsets` is defined:

```
class CPhaseOffsets
{
```

```
public:
	CPhaseOffsets();
	~CPhaseOffsets();

	BOOL m_doOffset;		// Flag: apply offsets or
not
	// The offsets
	char m_AB;
	char m_AC;
	char m_AD;
};
```

and

MAX_TX_SLOTS = 16 in release 2 code
MAX_TX_SLOTS = 5 in release 3 code.

Therefore, the data area should contain 16 instances of at least four bytes each: offset (or not) flag, then three phase offsets. However, with word alignment and function pointers, this block has a data length of 40 in the R3 code. Most standard survey systems will have the port offsets in array[0] and the starboard offsets in array[1]. Note that transducers are numbered from '1', so transducer 'i' will need offsets applied from array[i-1].

Offsets should be added before conversions. Note that the phases should be stored as "`unsigned char`". This will make sure that the phase "wraps around" (past 255 = 360° phase) when the offset is added.

```
for (i = 0; i < number; i++)
{
	samp[i].m_phaseb_a += m_phaseCorrAB;
	samp[i].m_phasec_a += m_phaseCorrAC;
	samp[i].m_phased_a += m_phaseCorrAD;
}
```

2.9 Hardware Configuration Data Block Format

A block type number is reserved for hardware configuration data, but no records of this type are generated at present.

3. Processed Data File Blocks

3.1 General

Processed data files are written with the file extension "SXP". They contain the following data blocks.

Note that some of the data block types are now obsolete.

These items write out memory images of C++ classes, as created by Microsoft Visual Studio. These memory images may have padding between some objects, to align data objects to word boundaries, so caution may be needed when reading these objects with code created by other compilers, languages and operating systems.

3.2 XYZA Data Block Format, SBP_XYZA_PING2

This format is used in the processed data files written by SWATHplus code distributed after January 2010. It contains all the processed data for a single ping. It is similar to the formats used internally by the SEA SWATH software, but the structures are defined separately in order to keep control of file size.

Following the block header, there are three kinds of element in the data block:

- Ping data (class cXYZAPing)

- Transducer data header (class cXYZATxer)
- Bathymetric data samples (class cXYZAPoint)

Each block contains the data from one transducer. If the sonar is operated in “simultaneous” mode, with both transducers firing at the same time, then two separate “SBP_XYZA_PING2” blocks will be generated, with the same time stamp.
Each block therefore contains one each of the data types, in order:

- Ping data
- Transducer data header
- Bathy data array

The ping data is as follows:

```
class cXYZAPing
{
       char m_lineName[MAX_LINENAME_LEN];      // line name
       unsigned long m_pingNum;                // ping number
       double m_time;             // UNIX time of start of ping
       int m_noTxers;             // Number of transducers used (always 1)
       CPosn m_posn;              // position of transducer
       double m_roll;             // roll at start of ping
       double m_pitch;            // pitch at start of ping
       double m_heading;          // heading at start of ping
       double m_height;           // height of transducer at start of ping
       double m_tide;             // tide height applied
       double m_sos;              // speed of sound applied (mean value)
};
const int   MAX_LINENAME_LEN = 40;
```

The sign conventions are explained in section 6.1. Height is the height below datum (measured positive down), and combines the heave and datum offset, which could come from GPS height or tide, for example.
Tide is measured with the usual marine convention, positive up.
The member `m_noTxers` is used to determine the number of transducer blocks and data arrays that follow.

The transducer data is:

```
// Data for transducer
class cXYZATxer
{
public:
       unsigned char m_txNo;      // transducer identifier
       unsigned char m_txStat;    // tx status
       unsigned char m_txPower;
       short int m_analogGain;    // analog gain value

       unsigned char m_noStaves;  // no. of staves on tx
       unsigned char m_txInfo[MAX_TX_INFO];  // board type/revision/serial
number
       unsigned char m_freq;      // tx frequency (identifier code)
       double m_frequency;        // frequency in hertz

       short int m_trnsTime;      // transmit time/ number of cycles
       short int m_recvTime;      // receive time/ number of samples
       unsigned char m_sampRate;  // receive sample rate micro-seconds per
sample

       // sample data
       int m_noSampsOrig;         // no. of samples read in real time
       int m_noSampsFile;         // no. of samples in the processed file
       int m_noSampSlots;         // no. of sample slots

       Cposn m_posn;              // position of transducer (E,N)
```

```
        CposnOffset m_posoffset;   // position offset of this transducer from
survey centre
};
```

Sub-definitions:

```
// Position
class Cposn
{
        double E;           // easting
        double N;           // northing
};

// General offset from survey centre
class CposnOffset
{
        double height;
        double forward;
        double starboard;
        double azimuth;
        double elevation;
        double skew;
        double time;
        double water_depth;
        double pitch;
};

const int   MAX_TX_INFO = 4;
```

The number of samples stored in the array that follows is `m_noSampsFile`. The number of samples stored in the original array was `m_noSampsOrig`, but some of these samples may have been rejected by filters and not written to the file.
This data array contains the three-dimensional position (xyz) and amplitude derived from the sonar data. The data points are not ordered in any particular way, but they will usually be stored in the order of the time in which the underlying phase data was collected.
Each point is encoded as follows:

```
class cXYZAPoint
{
        int                 m_sampNum;    // sample number
        double              m_x;          // x position (northing)
        double              m_y;          // y position (easting)
        float               m_z;          // depth (positive down)
        unsigned short int  m_amp;        // raw amplitude (16 bits)
        unsigned short int  m_procAmp;    // processed amplitude (16 bits)
        unsigned char       m_status;     // extra information
        double              m_TPU;        // uncertainty
};
```

The `m_status` field gives information about the status of the data point. A value of zero indicates that the point has been rejected by a filter.

Note that the three-dimensional xyz axis set is:

x Northing
y Easting
z Depth, positive down

3.3 XYZA Data Block Format, "SBP_XYZA_PING"

This format is used in the processed data files written by SWATHplus code distributed before January 2010.
The ping data element is the same as in SBP_XYZA_PING2.
Within the transducer data the position offset sub-element contains one fewer fields:

```
// General offset from survey centre
```

```
class CposnOffset
{
	double height;
	double forward;
	double starboard;
	double azimuth;
	double elevation;
	double skew;
	double time;
	double water_depth;
};
```

The individual data points each contain one fewer fields:

```
class cXYZAPoint
{
	int                 m_sampNum;    // sample number
	double              m_x;          // x position (northing)
	double              m_y;          // y position (easting)
	float               m_z;          // depth (positive down)
	unsigned short int  m_amp;        // raw amplitude (16 bits)
	unsigned short int  m_procAmp;    // processed amplitude (16 bits)
	unsigned char       m_status;     // extra information
};
```

4. Parsed Data File Blocks

4.1 General

Raw data files are written with the file extension “SXI”. They contain the following data blocks.

4.2 Common Parsed Data Codes

The parsed data blocks encode time and data channel the same way, as follows.

4.2.7 Time Codes

Eight-byte timestamps are organised as two four-byte integers. The first integer represents seconds since 1970, and the second integer represents microseconds since that second.

4.2.8 Channel Number

Channel number identifies a transducer. The software that reads the data will need to store, and account for the location and pointing angle of each transducer, in three dimensions, relative to the attitude and position system data. There are usually, but not always, two channels. Channel 1 is usually port, and channel 2 is usually starboard, but that is not guaranteed. Transducers may fire alternately (port-starboard-port-starboard …), simultaneously, or singly (port-port-port … or starboard-starboard-starboard).

4.3 PARSED_PING_DATA

Byte num	Num bytes	Encoding	Item	Notes
0	8	See §4.2.7.	Time	Start of ping time code. All 8-byte time codes are encoded the same: see §4.2.7.
8	1	Unsigned char	Channel	Identifies the transducer
9	4	unsigned long int	Ping number	Starts from when program starts 1 and increments. Simultaneous pings are numbered separately.
13	4	Float	Sonar frequency	Frequency of the transducer, in Hz
17	4	Float	Sample period	Time period between sonar data samples, in seconds.
21	2	unsigned short int	Number of samples	Number of samples following
23	4	Float	Sound speed	Speed of sound used to calculate angles, m/s
27	2	Short int	Tx pulse	Transmit pulse length, in sonar cycles
29	1	Char bit field	Data options	Allows options in data encoding. See §4.3.9.
30	1	Unsigned char	Ping state	Records the status of pinging: single/alternating/simultaneous etc. See §4.3.11
31	2	Unsigned short int	Max count	Maximum data count before filtering
33	2	To be determined	Reserved	Reserved for other ping information
				The header information is followed by “Number of samples” examples of the following sample data

Byte num	Num bytes	Encoding	Item	Notes
35 +(n×7)	2	unsigned short int	Number	Sample number. Calculate range from this value: see §4.3.13. Guaranteed to increase in a ping, but may not be sequential: this allows for down-sampling
37 +(n×7)	2	signed short int	Angle	Angle coded +15 bits = 180° up, -15 bits = 180° down, relative to the transducer pointing angle. See §4.3.13.
39 +(n×7)	2	unsigned short int	Amplitude	Scaled so that 16 bits is the full scale of the ADC
41 +(n×7)	1	Unsigned char	Quality	As set by "Data options". See §4.3.9.

4.3.9 Sonar Data Options

The "Data Options" byte in the header part of the PARSED_PING_DATA block allows for options in the encoding of data, encoded as follows:
Bit codes:

7	6	5	4	3	2	1	0
Not currently used					Meaning of quality byte		

Bits 0-2 are used to encode the meaning of the quality byte. The remaining bits are currently not used.

Value	Name	Meaning The quality byte in each sample is …
0	PARSED_ QUALITY_MERGED_CALC	Generated from a combination of quality factors. A measure of relative quality. 255 = maximum quality, 0 is a rejected item.
1	PARSED_ QUALITY_PHASE_QUAL	Generated from phase decode quality. The SWATHplus transducers have several stave pairs to use: this value is a measure of how well the decodes for each pair agree
2	PARSED_ QUALITY_FILTER_ACCEPTANCE	A bit-wise set of accept-reject flags for a set of up to 8 filters. If a filter rejects a sample, then the bit corresponding to that filter is set to 1. If the data point is accepted, or that bit is not used (there isn't a filter assigned to the bit), then it is left zero. Accepted data points therefore have all bits set zero.

4.3.10 Quality Filters

If working in "PARSED_ QUALITY_FILTER_ACCEPTANCE" mode, the meaning of the bits in the quality flag is as follows:

Bit	Name	Meaning
0	FILTER_FLAG_PHASE	Phase coherence filter: checks that the angle decodes for various combinations of transducer stave pairs agree
1	FILTER_FLAG_ANGLE_CONFIDENCE	A moving-window filter in angle, checking that angles are within some set limit
2	FILTER_FLAG_AMPLITUDE	Checks that amplitude is greater than the base noise level
3	FILTER_FLAG_ZER0_ANGLE	Flags items that have an angle of exactly zero, or are within some user-defined angle of the transducer boresight
4	FILTER_FLAG_RANGE	Flags items that are outside user-defined minimum and maximum angles

Bit	Name	Meaning
5	FILTER_FLAG_WATERCOL	Flags items that either have a slant range less than an average nadir depth, or are within 30° of the nadir and are greater than the average nadir depth. In both cases, a percentage threshold margin is allowed either side of the nadir depth.
6	FILTER_FLAG_DEPTH	Flags items that are outside user-defined minimum and maximum depth

The user may choose not to output any points that have any filter flags set, in which case all filter bytes in the output will be zero.

4.3.11 Status of Pinging

This byte records the sonar activation options.
If the byte is set zero, then this byte has no meaning. This allows for successful decoding of previous versions of the data file.

7	6	5	4	3	2	1	0
Not currently used				Port-starboard (set 1 for stbd)	Tx on (otherwise receive only)	Ping mode	

Ping Mode:

Value	Name	Meaning
0	SONAR_SEL_OFF	Not used
1	SONAR_SEL_SINGLE	Single-sided pinging
2	SONAR_SEL_ALT	Alternating pinging
3	SONAR_SEL_SIM	Simultaneous pinging

Port-starboard flag is set using the transducer position data. There is no guarantee that the user has set this data, so the validity of this field is not guaranteed.

4.3.12 Sonar Data Angle

Sonar data angle is relative to the transducer pointing angle. For SWATHplus, the pointing angle is usually 30° down and -90° azimuth for port and +90° for starboard, but any configuration is theoretically possible. Angular measurement is only for the semicircle in front of the transducers.
The sign convention is positive up. On a flat seabed, samples will start negative and change to positive when the returns sweep through the boresight of the transducer.
The conversion factor into radians is (pi/ 32768), and (180/32768) for degrees.
If SWATHplus is not configured with the true speed of sound at the transducer head, then the angles need to be corrected for this speed of sound in the program that consumes this data.
The correction is as follows:

True angle = arcsin(sin(angle) * (true sound speed) / (nominal sound speed))

4.3.13 Sonar Data Range

Range is calculated by:

range = (sample number) * (sample period) * (speed of sound) / 2;

Range is always in front of the transducer, in a line along the transducer “boresight”.
Note that the speed of sound used is likely to be a default value, not taken from a measurement. Therefore, the range should be corrected by multiplying by the ratio between measured speed of sound and the speed of sound in the PARSED_PING_DATA sample.
A possible refinement is to modify range to the mid-point of the sonar pulse, using “Tx pulse” and the sonar frequency.

4.3.14 Sonar Data Time

The precise time of a sonar data sample is calculated by:

(PARSED_PING_DATA time) + ((PARSED_PING_DATA sample period) * (sample number)

4.3.15 Height

Height could be:

- An absolute height, from a combined attitude and position system that is capable of reading GPS height to sufficient accuracy
- A height relative to the water surface: in this case a tide table will be needed
- A heave value: that is, relative to a local average height

It is up to the processing application that uses this data to make the distinction for each case and deal with the data appropriately.

4.4 PARSED_ATTITUDE

Byte num	Num bytes	Encoding	Item	Notes
0	8	See below	Time	Start of ping time code. All 8-byte time codes are encoded the same: see §4.2.7.
8	1	Unsigned char	Channel	Identifies the data source
9	4	Float	Roll	Positive for starboard down
13	4	Float	Pitch	Positive for nose up
17	4	Float	Heading	Positive clockwise, looking down
21	4	Float	Height	Positive for down

Attitude information can come from a mix of sensors. The Swath system builds combined attitude and heading packets from whatever comes into the system, using user-entered "attitude derivation" selections. Data packets are provided at the same rate as whatever is providing roll (which needs to be at the highest frequency, and so should not be sub-sampled). If heading is at a lower rate, it would be repeated in attitude packets until a new item comes in.
See §6.1 for notes on sign conventions.

4.5 PARSED_POSITION_LL

Byte num	Num bytes	Encoding	Item	Notes
0	8	See §4.2.7.	Time	Start of ping time code. All 8-byte time codes are encoded the same.
8	1	Unsigned char	Channel	Identifies the data source
9	8	double	Latitude	Degrees
17	8	double	Longitude	Degrees

4.5.16 Position Latitude-Longitude and Easting-Northing

Position systems can provide position data either in latitude and longitude or easting and northing.

- If the positioning system provides latitude and longitude, then PARSED_POSITION_LL packets are provided. The system will also probably provide PARSED_POSITION_EN packets, using the conversion factors selected by the SWATHplus operator. However, this converted EN packet is not guaranteed in this case.
- If the positioning system only provides easting and northing, then PARSED_POSITION_EN packets only are supplied: no conversion to LL is provided.

4.6 PARSED_POSITION_EN

Byte num	Num bytes	Encoding	Item	Notes
0	8	See §4.2.7.	Time	Start of ping time code. All 8-byte time codes are encoded the same.
8	1	Unsigned char	Channel	Identifies the data source
9	8	double	Easting	Metres
17	8	double	Northing	Metres

4.7 PARSED_SVP

Byte num	Num bytes	Encoding	Item	Notes
0	8	See §4.2.7.	Time	Start of ping time code. All 8-byte time codes are encoded the same: see §4.2.7.
8	1	Unsigned char	Channel	Identifies the data source
9	4	float	Speed of sound	Metres per second

4.8 PARSED_ECHOSOUNDER

This is only sent out if a separate single-beam echosounder is fitted to the SWATHplus system.

Byte num	Num bytes	Encoding	Item	Notes
0	8	See §4.2.7.	Time	Start of ping time code. All 8-byte time codes are encoded the same: see §4.2.7.
8	1	Unsigned char	Channel	Identifies the data source
9	4	float	Altitude	Height above seabed; Metres

4.9 PARSED_TIDE

This is only sent out if tide data is provided to the SWATHplus system in real-time.

Byte num	Num bytes	Encoding	Item	Notes
0	8	See §4.2.7.	Time	Start of ping time code. All 8-byte time codes are encoded the same: see §4.2.7.
8	1	Unsigned char	Channel	Identifies the data source
9	4	float	Tide height	Metres

4.10 PARSED_AGDS

This is only sent out if a separate Acoustic Ground Discrimination System (e.g. SEA's ECHOplus) is connected to the SWATHplus system.

Byte num	Num bytes	Encoding	Item	Notes
0	8	See §4.2.7.	Time	Start of ping time code. All 8-byte time codes are encoded the same: see §4.2.7.
8	1	Unsigned char	Channel	Identifies the data source
9	4	float	Hardness	
13	4	float	Roughness	

5. Real-time Command and Status

5.1 Real-time Command and Status Data Formats

There are essentially two classes of TCP/IP command and status connections used with the Swath Processor program:

1. Connection between two Swath Processor programs. This might be used if data is collected in a remote location and processed and visualised on another computer. This connection uses the same data blocks as for the raw data file, plus the following command types: TEXT_DATA, SYSTEM_COMMAND_DATA and TIME_SYNCH_DATA.
2. Connection between SWATHplus and an external system. This uses the following data blocks: CMS_CMD, CMS_STATUS and AUX_ATTPOS.

5.2 Connection Between Swath Processors

5.2.17 TEXT_DATA

The data payload is a character string (the length is defined by the block header). The receiving program simply displays the text in the Status view window.

5.2.18 SYSTEM_COMMAND_DATA

A single integer follows the header, defined as follows:

Type	Value	Notes
SCOMMAND_SHUTDOWN	0	Sent from one process to the other, indicating that it is shutting down. It causes the other process to shut down its TCP/IP socket.
SCOMMAND_DATARESET	1	The idea is to cause the other process to reset its data buffers. It is disabled in software at the time of writing.
SCOMMAND_KILL_APP	2	Causes the receiving application to shut down.

5.2.19 TIME_SYNCH_DATA

The data payload is a Windows SYSTEMTIME structure. It causes the receiving application to set the computer's clock to the time transmitted to it.

5.3 External System Interface

Real-time TCP/IP data transfer uses the block-orientated structure that is used for other data transfer and file storage in the SWATHplus system.

Type	Value	Notes
CMS_CMD	0x40	Commands in
CMS_STATUS	0x41	Status out
AUX_ATTPOS	0X42	Attitude and position

Immediately following the block type is the block length, again as a 32-bit integer. The block length is the number of bytes in the block, not including the header.

5.3.20 CMS_CMD

The data section is as follows:

Item	Length /bytes	Values	Notes
Sonar on-off	1	0 = disable data acq	
		1 = enable data acq	
Transmit on	1	0 = disable sonar transmission	
		1 = enable sonar transmission	
Transmit power	1	0 to 100	Can be set to use old 0 to 15 range in swathprocconfig
		>100 : Ignore	
Range	2	>0: Nominal sonar range in metres	
		Limits 1 to 2000	
		0: Ignore	
Samples per ping	2	0 to 65535. Typically 1024, 2056 or 4068.	
		Limits 256 to 65535	
		0: Ignore	
Pulse length	2	>0: Pulse length in cycles. Typically 8 to 200	
		Limits 2 to 250 Normally set low; say 8. Then increase to get more range.	
		0: Ignore	
Pulse repetition frequency	2	>0: Pings per second. Only effective if slower than implied by “Range”.	
		Limits 0.01 to 40	
		0: Ignore	
Transducer	1	0 to 15.	
		0: alternate transducers (“ring round” if more than two transducers)	
		>0: use the transducer with this number	
		>15: Ignore	
Write to file	1	0 = don’t write	
		1 = write	
Close the file	1	0: Ignore	
		1: close the raw file	
Open a new file	1	0: Ignore	
		1: close the existing raw file and open a new one (file name must follow)	
		2: The File name field is taken as the base for auto-creating files	
File name	50	Name of new raw file or base name. See §0 below for name format. Null-terminated string.	
Generate	1	0: do not generate sxp files	

Item	Length /bytes	Values	Notes
processed file		1: generate sxp files, with the same names as the sxr files. All sxr commands also apply to sxp.	
Data controls valid	1	0: Ignore the following data controls 1: Apply the following data controls	The Swath code also responds to a control packet that ends at the above item.
Send data	1	0: Do not send data from the parsed data port 1: Send data	
Parsed data options	1	As per PARSED_PING_DATA bit 29.	Allows options in data encoding. See §4.3.9.
Parsed filter remove filtered data	1	0 = don't remove 1 = remove	For Parsed data Filtering
Parsed filter downsample	2	Unsigned short int 0 = don't downsample >0 = target value for downsampling	For Parsed data Filtering
Parsed filter enable/disable	1	Up to 8 filters, as per PARSED_PING_DATA Quality flag	For Parsed data Filtering
Parsed filter low amp	2	Unsigned short int If below 600, this number is interpreted as a percentage of the average amplitude background noise level. Above 600, it is interpreted as an absolute 16-bit amplitude level. 0 to 100 (%). Usually set ~110%. Min zero, max 65535.	Parsed data low amplitude filter
Parsed filter phase confidence	1	Unsigned char 0 to 100 (%). Usually set ~ 60%. Min zero, max 100.	Parsed phase confidence filter
Parsed filter angle window	2	Unsigned short int. Window count for angle filter. Usually set ~25. Min 3, max 250.	Parsed angle proximity filter
Parsed filter angle range	1	Unsigned char In hundredths of sine. Normally set to ~0.1, so 10 in this byte. Min 0, max 1.0.	Parsed angle proximity filter
Parsed filter angle threshold	2	Unsigned short int. Number of adjacent items needed. Normally ~10. Min 2, max 250.	Parsed angle proximity filter
Parsed filter boresight angle	2	Unsigned short int. Size of the angular segment around the boresight that is filtered. In tenths of a degree (0 => 0 degrees, 1 => 0.1°, 10 => 1°)	Parsed data Boresight Filter This and the following added issue 5M, May 08
Parsed filter minimum range	2	Unsigned short int. Minimum slant range, in metres	Parsed data range filter
Parsed filter maximum range	2	Unsigned short int. Maximum slant range, in metres	Parsed data range filter

Item	Length /bytes	Values	Notes
Parsed filter maximum horizontal range	2	Unsigned short int. Maximum horizontal range, in metres	Parsed data range filter
Parsed filter minimum depth	2	Short int Minimum depth, metres	Parsed data depth filter
Parsed filter maximum depth	2	Short int Maximum depth, metres	Parsed data depth filter
Parsed filter watercolumn threshold	1	Unsigned char. 0-100, as a percentage. E.g., if set to 10, then values shallower than 10% less than the nadir depth are removed, and values greater than 10% more than nadir and close in angle to nadir, are removed.	Parsed data watercolumn filter
Parsed filter soft maximum range	2	Unsigned short int. Maximum slant range, in metres	Only removes low amplitude data
Maximum Horizontal Range Filter	1	1 = Use filter 0 = Don't use filter	
Minimum Horizontal Range Filter	1	1 = Use filter 0 = Don't use filter	
Parsed filter minimum range	2	Unsigned short int. Minimum horizontal range	
Second parsed filter angle window	2	Unsigned short int. Window count for angle filter. Usually set ~25. Min 3, max 250.	Parsed angle proximity filter
Second parsed filter angle range	1	Unsigned char In hundredths of sine. Normally set to ~0.1, so 10 in this byte. Min 0, max 1.0.	Parsed angle proximity filter
Second parsed filter angle threshold	2	Unsigned short int. Number of adjacent items needed. Normally ~10. Min 2, max 250.	Parsed angle proximity filter

3.4.1.1 Data Output controls

The items from "Data controls valid" to "Parsed filter angle threshold" were added Aug 07. The SWATHplus code continues to support CMS_CMD without these items, and sets them all to zero if not present in the message.
If the sending process does not care about the settings of the data controls, then set "Data controls valid" to zero, and the rest of the data controls are ignored.

Auto-generated files

If "open a new file" is set to a value of 2, then the file is generated using the supplied file name (which could include the file path), plus the date and time. The auto-generated file name will be of the form:
<file path>yyyyMMMdd_hhmmss.

27yyyy: 4-digit year
MMM: three-letter short form of the month name
dd: day of the month
hhmmss: hours, minutes and seconds

E.g., if the file path is "C:\SW_files\test", and the time is 17:23:42 on 06 February 2006, then the file name generated is "C:\ SW_files\ test2006Feb06_172342.sxr".

5.3.21 CMS_STATUS

As per CMS_CMD, but filled in from the internal system status feedback where possible.

Item	Length /bytes	Values
System start	1	1 = the system has just started up. The other fields in the status message are uncontrolled
		0 = normal running. Other fields are valid
Error	2	0 = normal 1 = error detected (other bitwise error values may be defined later)
Pulse repetition frequency	2	Pings per second.

5.3.22 AUX_ATTPOS

Item	Length /bytes	Values
Latitude	4	Latitude, Double, degrees
Longitude	4	Longitude, Double, degrees
Depth	4	Depth below datum, Double, metres
Altitude	4	Vehicle altitude from “seabed”, metres
Roll	4	Euler 321, Double, degrees
Pitch	4	Euler 321, Double, degrees
Heading	4	Euler 321, Double, degrees

“Heading” and “yaw” are synonymous: rotations around a vertical axis.
Vehicle attitude is in the Euler 321 convention. This rotation from a fixed local origin frame to the body frame is represented by

i. Yaw Euler Angle (which equals heading for Euler 321) . A positive yaw from an attitude of (0,0,0) would cause the front of the vehicle to rotate to starboard.
ii. Pitch about the frame after the yaw. A positive pitch from an attitude of (0,0,0) would cause the front of the vehicle to move upwards.
iii. Roll about the frame after the pitch. A positive roll from an attitude of (0,0,0) would cause the starboard side of the vehicle to move downwards.

6. Notes

6.1 Axis Conventions

All angles obey the right-hand screw rule about the appropriate axis. Headings obey this rule about the z-axis, whilst at the same time maintaining the usual geographic convention, that is, positive going clockwise, measured from North.
Depth is positive for *down*.
This means that:

- Heading is positive clockwise, looking down.
- Roll is positive for starboard down.
- Pitch is positive for nose up.

The SWATHplus software uses the Euler angle convention for roll and pitch, rather than the horizontal-plane convention. This means that roll is measured about the body's own forward-aft axis, rather than relative to the horizontal plane.
Tide values are given in the usual marine convention, positive up.

7. Serial Port Connections

7.1 System Commands

Swath can be controlled by control codes passed to it over the RS232 “com” ports.
The codes are simple ASCII text messages. Each message must start with a ‘>’ character.

Available codes are:

>SNR ON	Sonar control: start pinging
>SNR OFF	Sonar control: stop pinging
>SNR PWR *power*	Sonar control: set power to specified number, 1 to 10. E.g. “>SNR PWR 8”
>SNR RNG *range*	Sonar control: set ping range in metres
>SNR PLS *pulselength*	Sonar control: set pulse length in cycles, 2 to 250. E.g. “>SNR PLS 100”
>SNR PRF *pingrate*	Sonar control: set ping repetition frequency in Hz, 1 to 30. E.g. “>SNR PRF 20”
>SNR SMP *samples per ping*	Sonar control: set the number of samples per ping, 512 to 8192. E.g. “>SNR SMP 4096”
>FILE ON	Start writing to raw file
>FILE OFF	Stop writing to raw file
>FILE CLOSE	Close raw file
>FILE PAUSE	Pause writing to raw file
>FILE OPEN *filename*	Create a new raw file, with name given by “filename”. E.g. “>FILE OPEN line3”.
>SESSION LOAD *filename*	Load a session file with the specified name
>SESSION SAVE *filename*	Save the session file with the specified name

8. Document Change Record

The previous change history of this document is as follows. See section 1.1 for recent change history.

Issue[1]	Date	Reason for Change and Issue
0.1		Original
2.0		Changes to the format to include timestamps on strings, as well as some minor typographical corrections
3.0	21 May 1998	Adds the file header record
3.2	18 August 1998	Adds the xyza format
3.3	25 August 1998	Additions to xyza format
3.4	26 August 1998	Config and processed data file information added
3.5	06 April 1999	`cXYZAPoint` corrected in line with released code
3.6	27 April 1999	Note added that file header may be absent
3.7	29 April 1999	Array sizes defined
3.8	05 May 1999	Clarification of easting and northing
3.9	07 May 1999	Name changed of samp counts in txer data
3.91	03 June 1999	File header length corrected
4.0	19 December 2003	Changed to SEA SWATHplus format
4.1	18 February 2004	Clarified some sign conventions
4.2	21 October 2005	Added phase offset notes
5 A	July 2006	Formal SEA format
5 B	November 2006	Version 3 software data format added
5 C	November 2006	Aux ports documented. Other data block types identified.
5 D	December 2006	TCP/IP and Com Port Command and Status command formats clarified.
5 E	February 2007	Added Parsed Data format
5 F	June 2007	Corrections to interpretation notes for angle in the parsed data format
5 G	July 2007	SXI filtering information
5 K	December 2007	Parsed Data Filter information updated
5L	March 2008	Format of MRUT_DATA clarified
5M	May 2008	CMS_CMD modified to add new filter settings
5N	September 2009	Updates to content of Processed ping data.
6A	January 2010	New format, updated appearance for clarity
6B	January 2010	Corrected error in description of old SXP format. Note on class memory images added.
6C	February 2010	Added extra filter controls in CMS_CMD

[1] Early versions used Submetrix issue format