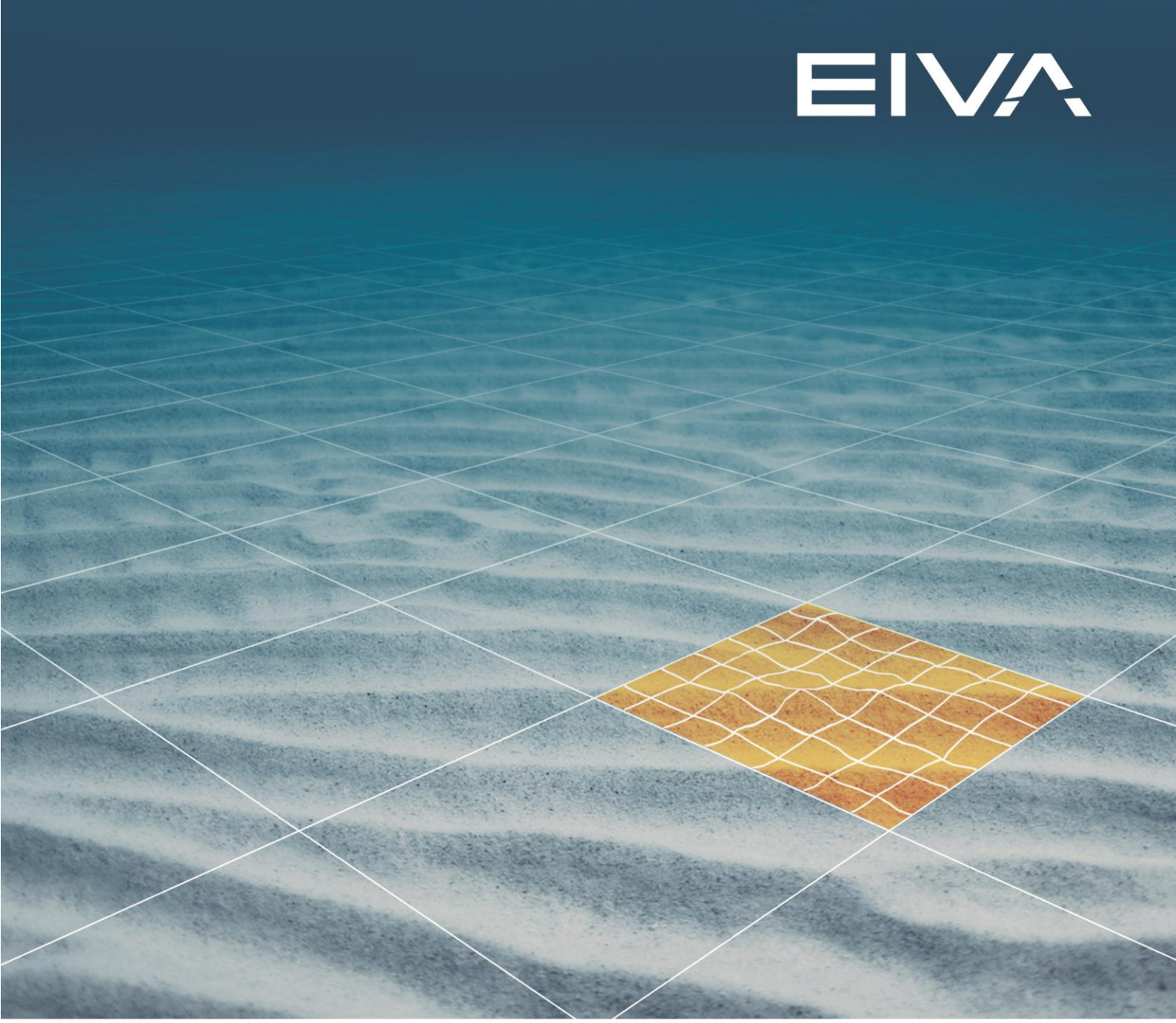


NAVISCAN

REFERENCE MANUAL

Last update: 25/03/2021
Version: 9.5

Contents

1 Introduction 4

1.1 Sensors 4

1.2 Serial and Network Interfacing 5

1.3 Tools and Offline Programs 5

1.3.1 NaviScan Online 5

2 Sensor Format 8

2.1 Gps time 8

2.2 Runline Control 9

2.3 Bathy 9

2.4 Gyro 12

2.5 Motion 14

2.6 Auxiliary 17

2.7 Rawdata 17

2.8 Navigation 17

2.9 Echosounder 19

2.10 SideScan 24

2.11 Pipetracker 27

2.12 Doppler log 28

2.13 Laser Scanner 29

2.14 Theoretical profile 30

3 NaviScan SBD logging format 30

3.1 File structure 31

3.1.1 Packet header 31

3.1.2 Sub Packet 33

3.2 FILE ID in SBD files 33

3.3 Header Record 34

3.3.1 UserNames and Dopplerlog offsets addition. 41

3.4 TimeBox interfaced sensors 42

3.5 Gyro Record 42
3.6 Motion Record 43
3.7 Bathy Record 43
3.8 Multiple position Record 44
3.9 Raw data Record 44
3.10 Auxiliary Record 44
3.11 Projection & Ellipsoid Record 45
3.12 Datum shift Record 46
3.13 Utility Record 46
3.14 Logcontrol Record 47
3.15 Multibeam echosounder Record 47
3.16 SideScan Record 47
3.17 Pipetracker 47
3.18 DopplerLog Record 48
3.19 Position Record 48
3.20 Filtered Position Record 48
4 Time stamping 48
4.1.1 TimeBox absolute time tagging 49

1 Introduction

The **NaviScan** data acquisition software is designed specifically for the multi-beam echo sounder and sidescan sonar technology. A wide range of echosounders, sidescans, pipetrackers and supporting positioning instruments are supported. **EIVA** offers the optimal solution for vessel and ROV surveys.

The basic purpose of **NaviScan** is to collect data from the Echosounder sensors and generate a file with exact positions of each scan. To do this navigation information is needed along with roll/pitch/heave, bathy sensor and gyro. To improve positions, NaviScan includes support for Doppler Velocity Log. Another feature is included; combine echosounder data and pipe-tracker to get the best pipeline survey.

The **NaviScan** package contains programs to convert the raw logged data to motion corrected data which can be used in processing software packets. During this conversion phase, data is corrected for pitch, roll, heading, C-O etc, and all low-quality data is filtered out.

All **NaviScan** programs can be executed on one PC, as the multitasking environment allows simultaneous execution of more programs without interfering with the on-line data time-tagging: this very important process is placed in a dedicated real-time queue. For larger set-up's, the system can be split on more computers.

The **NaviScan** system works on a PC platform which is simple, easy-to-handle and involves minimum mobilisation time. The system is flexible and user friendly. It offers precise time tagging and maximum up-time for continuous operation. The system configuration is based on the client/server philosophy and consists of one or more PC's (connected via a local area network (LAN)). The system is divided into two parts; one is dedicated to on-line data collection and logging (**NaviScan Online**) and the other is used for off-line data conversion, filtering, editing, and configuration set-up and data transportation (**NaviScan Configuration**).

The collected data is displayed on the screen so the operator can supervise the system; inspect the quality of incoming data and to make sure that none of the sensors are dropping out.

Data collected for each survey line will be stored in a file named by the operator. When the line is ended (on a signal from the navigation system or by operator) the file is closed.

1.1 Sensors

The echosounder sensors transfer information to NaviScan each scan. The scan information includes timing, depths and ranges.

An integrated navigation system is used for the identification of current system position. NaviScan can use data from any system giving Easting and Northing in a CR/LF terminated serial string. A special input is included for EIVA's NaviPac online navigation system, which furthermore allows remote control, DAL/DOL and KP information and on-line coverage information.

For ROV jobs, a Doppler Velocity log can be used to improve the positioning accuracy; a filter combining position long-term stability with Doppler short-term accuracy, gives a very accurate solution.

To compute the exact position of each beam, NaviScan needs heading (gyro) information. The system accepts any CR/LF terminated string giving the heading in decimal degrees.

Also the heave, roll and pitch data is needed for movement correction of each scan. NaviScan accepts data in any CR/LF terminated string giving Roll and Pitch in decimal degrees and Heave in cm.

When mounted on a ROV, the NaviScan system needs a bathy sensor giving the exact depth of the ROV. NaviScan accepts data in any CR/LF terminated string giving depth in decimal meters.

NaviScan supports integrated use of Pipe- or cable-tracker data; range to pipe can be combined with multi-beam data to give the ultimate pipe survey solution.

1.2 Serial and Network Interfacing

The online NaviScan program reads the incoming sensor data from a special intelligent serial communication board so that data I/O can be performed without interference from logging of the data, computation and presentation.

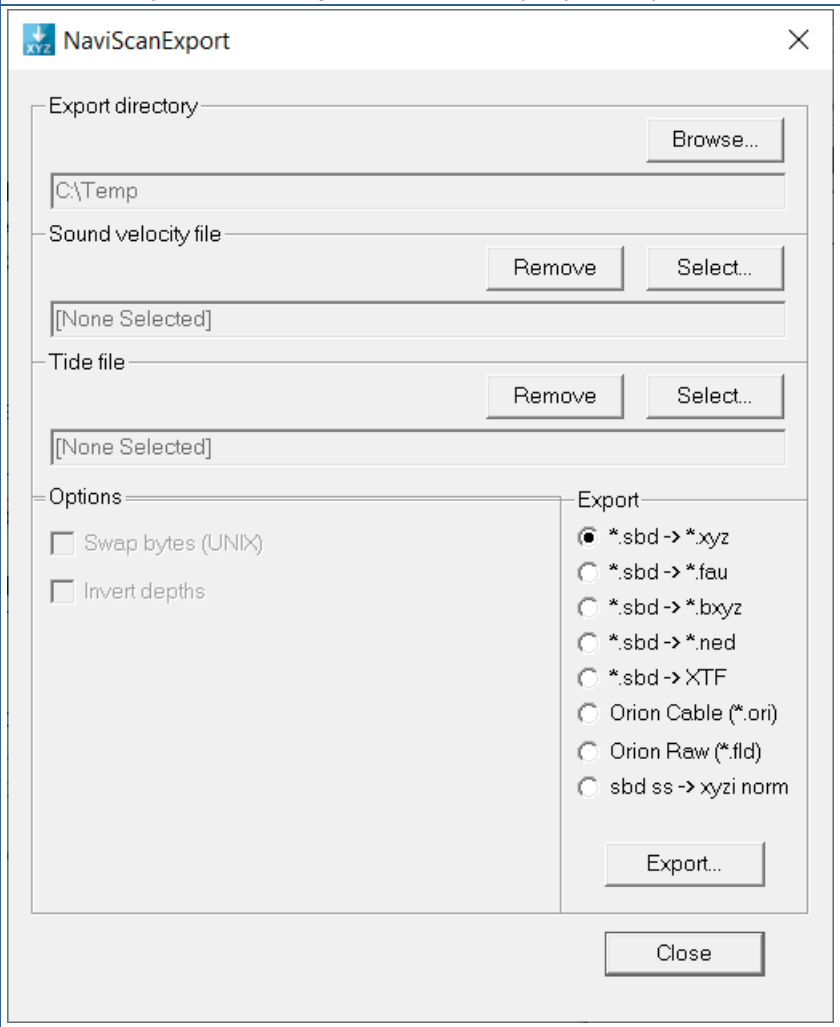
Network is also used for interfacing primarily sensors with heavy data loads or time tagged data. NaviScan supports both UDP and TCP communication.

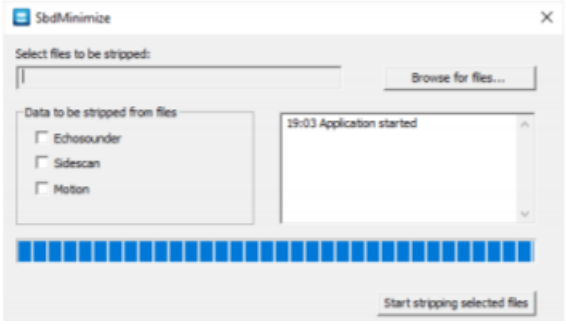
1.3 Tools and Offline Programs

1.3.1 NaviScan Online

The **NaviScan Online** program reads the binary file **NAVISCAN.BIN** to start-up the communication for all sensors. With the **NaviScan Configuration** program the operator can in an easy way create/modify this file.

Port identification and data format for all sensors, specification of serial communication, vessel or ROV offsets and Sensor correction values along with exact location of all sensors has to be defined in order to make NaviScan work.

Tools	Date
ATTU Monitor	Use for ATTU Time Synchronization program (AttuMon.exe). It was created for the purposes of monitoring the ATTUs and synchronization the system time of the computer with one of them.
Extract	Generates Windows EXCEL compatible files on basis of data extracted from the raw logged data files. Consequently, graphs and tables over selected data can easily be created. This feature is mainly of interest during the calibration and quality control phase
NaviScan Exporter	 The NaviScan Exporter has several options:

	• ***.sbd to *.xyz** Converts the raw logged data to motion corrected data stored in simple XYZ format (ASCII or binary), which can be used in some processing software packets. During conversion, the data is corrected for heading, roll, pitch, C-O etc and all low-quality data is filtered out. Furthermore the operator can remove specific beams; this is useful when the echosounder is operating at the limit of the specified range, which can cause low quality of the outer beams. At the end of the conversion, the NaviScan Converter creates a quality report, so the operator can inspect the amount of discarded data. A high number of discarded can indicate improper set-up of the echosounder or the NaviScan system. • ***.sbd to *.fau** • ***.sbd to *.bxyz** • ***.sbd to *.ned** ***.sbd to *.XTF** The XTF converter can take one or more SBD files (includes sidescan data) and convert them to the standard XTF (defined by Triton) file format • **Orion Cable (*.ori)** • **Orion Raw (*.fld)** • **sbd sss to xyzi norm**
SV Profiler	To read/show sound velocity profiles
SV Monitor	To monitor sound velocity read from ScanFish software
SVP2 Fansweep	For Hydrosweep inbound server, these setup items are used (here with default settings): hmsystemid=9 hmasddatatype=8 hmmbechannel=es_targets_port hmmotionchannel=fs20_motion_1 For the SVP to Fansweep program these items are used: XDR Port=3070 z-keel=0.050
SBD Minimizer	

SBD Splitter	To split an SBD file, by kp or size
Simulator	To simulate different sensors

Table 1 Tools

2 Sensor Format

NaviScan supports I/O for a number of predefined sensors and gives the opportunity to define new sensors as long as it outputs a fixed terminated string and the information is located with a given offset from the beginning of the received string.

Hereafter are listed the different possible sensor types in the SBD format. The enum id and the corresponding static char string identities the different sensors, and we try to keep these updated, after that follow a frame of the same and some extra information, but possibly new sensor may not be updated in the frames.

The field **SBD id** has been added, so that its possible to convert an id in the SBD file to the type. The order of these ids will never change; new sensors will always be added at the end. The definitions are per January 2011.

2.1 Gps time

NaviScan supports the following time synchronised data inputs:

Company	Instrument	Data string	Help	SBD id
	None		No data stored.	0
Applanix	Pos mv	Binary data...		6
Ashtech	Ashtech	$PASHR;PPS;dddddd<cr><lf>		2
NMEA	NMEA ZDA	PPS and Time string: $xxZDA;hhmmss.hh;dd;mm;yyyy;xx<cr><lf>		3
Novatel	ProPak6	Binary data...		5
QPS	Qinsy time	PPS and Time string: <stx><cr><lf>#T aaaa bbbbbbb<cr><lf>..		4
SBG Systems	Navsight Marine GPS	Binary data (sbgECom protocol); uses data from either external GPS or PPS input	Virtual Serial Port over RS232/RS422/UDP[/TCP]	8
Trimble	Trimble (PPS)	UTC yy.mm.dd hh:mm:ss ab<cr><lf>		1

Wassp	WasspS3 PPS time	Binary data... (PPSSTATS)		7

Table 2 Gps time

2.2 Runline Control

NaviScan supports the following data inputs for logging control:

Company	Instrument	Data string	Help	SBD id
FREEBina	Free	User defined ASCII		1
HDP	HDP4060 remote line control	$S;IIIYYDDMMMhhmmss_nnn<cr><lf> $E<cr><lf>		4
EIVA	Integrated Runline Control	Binary EIVA defined.		6
EIVA	NaviPac	$S;<filename><cr><lf> $E; <unused characters><cr><lf>	Also including geodesy info	3
EIVA	NaviPac (Passive)	$S;<filename><cr><lf> $E; <unused characters><cr><lf>	Not controlling – just used for recording of geodesy	5
EIVA	NaviPac Old	0 nnnnnn hh:mm:ss.ss eeeeeeee.ee nnnnnnnn.nn a.aaa dddddd.dd ddd.dd kkkk.kkk R FFFFFFFF 0/1 indicates logging on/off	Old driver – recommend that you use the NaviPac instrument instead	2
	None		No data stored.	0

Table 3 Runline Control

2.3 Bathy

NaviScan supports the following bathy/height data inputs:

Company	Instrument	Data string	Help	SBD id
	None		No data stored.	0
Aanderaa	AanderaaBaromet	01 Air pressure pppp.pp MB<cr><lf>	Pressure in psi	25
Applanix	GpsHeightPosmv	binary telegram		24
Applanix	GpsTidePosmv	Binary data		33
Balder	Balder ROV Bathy	": a.aa d.dd hhhh rrrrr - ppppp"		22
Benthos	Benthos PSA 900	Ddd.dd Ttt.tt Rrr.rr<cr><lf>		18
Benthos	Benthos Imux200	"***** ggg.g ppp.p rrr.r ttt.t ddd.d....."		21
EdgeTech	EdgeTech bathy	Binary data		27

EdgeTech	EdgeTech4600 B	Binary data		28
EIVA	NaviPacHeaveCor	"HC+hhh.hhh, Aaa.aaa, GPSHgg.ggg"		14
EIVA	GpsTideNP	$NPTide;<Date>;<Time>;<Tide>;<GPS H><cr><lf>		31
Eiva	EelumeNav	Protobuf data ...		45
FFI	FFI Depth Hugin	$PFFILLD;<time>;<latt>;<N/S>;<long>;<E/W>;<depth>;<C/S>...<lf>		20
Free	Free Bathy	Ex. "DDD.DD <CR><LF>"	Any Fixed terminated string (max. 70 characters) giving depth as decimal meters.	1
Hi-Target	iBeam Altitude	Binary Hi-Target iBeam (starts with ascii 'IB')		41
Honeywell	HoneyPPTR20Dyndraft	“#01CP=p.ppp<cr>“		23
iXblue	IXSE_HSPOS Depth	$PIXSE;HSPOS_;hhmmss.ss;llmm.mmmmm;H;LLmm.mmmmm;D;d.dd;a.aa;x.xx;y.yy;z.zz...		49
Ixsea	PhinsBthyBinNav	Binary data...		37
Ixsea	PhinsBthyLongBinNav	Binary data...		44
Klein	KleinFishBathy	Binary CKleinType3Header		43
Kongsberg	SIS3000 depth	*****ggg.g xxx.xx yyy.yy ppp.pp mmm.mm aaa.aa ...<lf>	Gives pressure not depth.	7
Kongsberg	Simrad Bathy	-Dddd.dd	Test have shown that Simrad UK90 could give problems caused by delay in output.	3
NMEA	GpsHeightGGA	$--GGA;hhmmss.ss;llll.ll;a;yyyyy.yy;a;x;xx;x.x;h.h;M;n.n;M;...<lf>		16
NMEA	GpsTideGGA	$--GGA;hhmmss.ss;llll.ll;a;yyyyy.yy;a;x;xx;x.x;h.h;M;n.n;M;...<lf>		32
Norbit	NORBIT_WBMS_UAV	Binary data…		46
OTC	ØTC Special	ggg.gg,rrr.rr,ppp.pp,eeeeeeeeee.ee,nnnnnnnnn.nn,zzz.zz,oo.oo<cr><lf>		10
OxTS	MCOM	Binary data		38
Paroscientific	DigiqzDptParos	*ccccppp.pppp<cr><lf>		13
Paroscientific	DigiqzBarometer	*ccccppp.pppp<cr><lf>		26
Paroscientific	DigiqzDynDraft	*ccccppp.pppp<cr><lf>		42

SAIV	SaivDynDraft	21:25:26.199;N57899 T+28.270 P 0006.230 (L43 H<EIVA+>)		34
SBG Systems	Navsight Marine Bathy	Binary data (sbgECom protocol)	Virtual Serial Port over RS232/RS422/UDP[/TCP]	50
SBG Systems	Navsight Marine GPS Height	Binary data (sbgECom protocol)	Virtual Serial Port over RS232/RS422/UDP[/TCP]	51
ScanSense	PS-30 Pressure	S<Mantisse>E<Exponent>T<Temp><cr><lf>		29
Solo	ROV Solo string	Gggg.gP+pp.ppR+rr.rrDddd.ddAaa.aa<cr><lf>		15
Sonardyne	PSONNAV Depth	$PSONNAV;hhmmss.sss;llll.llllll;a;yyyyy.yyyyyy;a;x.xxx;x.xxx;x.xx;a;d.ddd;..		39
Sonardyne	LnavUTC Depth	Binary Lodestar INS message		40
Teledyne	TSS340Altimeter	:SQsLLLL VVVV AAAAsCCCCsSSSs1111s2222s3333s4444s5555s6666 qq		17
Tritech	Tritech (old) B	dddddaaaa<lf>		4
Tritech	Tritech V3.16 B	ddddddaaaaa<lf>		5
Tritech	TritechOld 1s B	dddddaaaa<lf>	With 1 sec. delay	6
Tritech	Tritech V3.28 B	ddddddddaaaaaaa<lf>		8
Tritech	TritechV3.28 1s	ddddddddaaaaaaa<lf>	With 1 sec. delay	9
Tritech	Tritech SCU-3 B	%D0074041401tttttpppppppppp .... ddddddddddhhmmssss<cr><lf>		11
Tritech	TritechSCU-3 1s	%D0074041401tttttpppppppppp .... ddddddddddhhmmssss<cr><lf>		12
Tritech	TritechSCU3Alti	%D0073061410+000500000020000+00500000213564800019...<lf>		19
Ulvertech	Ulvertech Bathy	ddddd,aaaa<CR><LF>	Altitude is only included for test purpose	2
Vaisala	VaisalaBaromet	<pres><cr><lf> (hPa)		35
Vaisala	VaisalaWeather	P= <pres> hPa T= <temp> 'C RH=***** %RH <cr><lf>		36
Valeport	VpSVX2BthPackCombi	sv M/SEC press DBAR temp C cond MS/CM<cr><lf> OR dt<tb>Tm<tb>cond<tb>pres...	both bathy and altitude with same mounting offset	30
Valeport	VpSVX2BthPackBathy	sv M/SEC press DBAR temp C cond MS/CM<cr><lf> OR dt<tb>Tm<tb>cond<tb>pres...	only bathy data; no altitude; for separate bathy mounting offset	47

Valeport	VpSVX2BthPackAlti	sv M/SEC press DBAR temp C cond MS/CM<cr><lf> OR dt<tb>Tm<tb>cond<tb>pres...	only altitude data; no bathy; for separate altimeter mounting offset	48

Table 4 Bathy

2.4 Gyro

NaviScan supports the following heading data inputs: Type String Lay-out Note SBD id

Company	Instrument	Data string	Help	SBD id
	FREE	Ex. "ggg.g<CR><LF>"	Any fixed terminated serial string (max. 90 characters) giving the gyro as decimal degree.	0
ACZ	ACZ Gyro	"gggg<CR><LF>"	gyro value in 1/10 degree	4
Applanix	ApxPosmvGyro	Binary packet 102 or 103		21
Atlas	Stn Atlas Gyro	BINARY STNATLAS format (M-13)		14
Atlas	HMInboundServ Gyro	Binary Atlas Format		24
Balder	Balder ROV Gyro	: a.aa d.dd hhhh rrrrr -ppppp		20
Benthos	Benthos IMUX200Gyro	***** ggg.g ppp.p rrr.r.....		19
CDL	CDL Gyro	H123.45P+12.345R+123.456 M123W1234.56U12.3<cr><lf>		22
CDL	CDL Tokimec2 Gyro	$PTVF;Pitch;Roll;Heading;....		28
Datasonics	Datasonics specGyro	*****ggg.g xxx.x yyy.y ...<cr><lf>		8
Echologger	DASS710 Gyro	Simple point format 110; $dist; ha; va; hdg; pitch; roll; voltage	COM only; default baud 115200 8 N 1	44
Edgetech	EdgeTech Gyro	Binary data...		27
Edgetech	EdgeTech4600AttGyro	Binary data…		29

Edgetech	EdgeTech4600PosGyro	Binary... Edgetech4600 pos datagram		39
EIVA	NaviPacCalcGyro	#NBHDT;ggg.gg;T;a.aaa;S<cr><lf>	ONLY FOR TEST	17
EIVA	Eiva InterpretedGyro	Binary data...		37
Eiva	EelumeGyro	Protobuf data		46
FFI	FFIAttitHuginGyro	$PFFIAAV;<time>;<heading>;<course>;<pitch>...		18
Hi-Target	iBeam Gyro	Binary Hi-Target iBeam (starts with ascii 'IB')		38
Ixsea	IxseaGyroSCOMEXGyro	$PSXN;id;user;ppp;rrr;ggg;pspeed;rspeed;gspeed*hh<cr><lf>		25
Ixsea	PhinsRovinsTAHGyro	$HEHDT;90.30;/$PIXSE;HSATIT;h.hhh;/$PHOCT;01;hhmmss.sss;G;AA;HHH.HHH;<CR><LF>		31
Ixsea	PhinsGyroBinNavGyro	Binary data...		33
Ixsea	PhinsGyroLongBinNavGyro	Binary data...		41
Klein	KleinFisGyro	Binary CKleinType3Header		40
Kongsberg	SeapathEM3Gyro	Binary data...		13
Kongsberg	Seapath AsciiGyro	$PSXN;23;roll;pitch;head;heave*csum		15
Kongsberg	SeapathEM3BestQGyro	Binary EM format	Binary data...	26
KVH	KVH1000Gyro	$H?HDM;ggg.g <cr><lf>	Small low quality compass for demo purpose.	2
MDL	MDL TrimcubeGyro	HggggP....<cr><lf>		5
MDL	MDL Gyro	H1234P+1234R+1234<cr><lf>		23
NMEA	NMEA Gyro	$??HDT;ggg.g...<cr><lf>		6
NMEA	NMEA (jic)Gyro	$HCHDM;ggg.g...<cr><lf>		7
NMEA	NMEA ($HEHDT)	$HEHDT,ggg.g...<CR><LF>		11
NMEA	NMEA ($HEHDT)	$HEHDT,ggg.g...<CR><LF>		16

Ocean Tools	OceanTools1218Gyro	HaaaaP±bbbbR±ccccQ<cr><lf>/HyyyyP+ppppR+rrrrh+hhhhQ<cr><lf>		30
OTC	ØTC special Gyro	ggg.gg;rrr.rr;ppp.pp;eeeeeeeeee.ee;nnnnnnnnnn.nn;zzz.zz;oo.oo<cr><lf>		10
OxTS	MCOM Gyro	Binary data...		34
RDI	RDIGyro	$PRDID;P.PP;R.RR;HHH.HH;*CS<cr><lf>		32
Robertson	Robertson Gyro	4 Bytes binary	Gives gyro as ggg.g in four binary coded digits.	1
RovTech Gyro	RovTech Gyro	HDG:gggg.gDEPTH:dddd.d ALT:aaaa.a<cr><lf>		12
SBG Systems	Navsight Marine	Binary data (sbgECom protocol)	Virtual Serial Port over RS232/RS422/UDP[/TCP]	47
SG BRown	SG Brown Gyro	“abcd<CR><LF>”	a - 100 degrees b - 10 degrees c - 1 degrees d - 1/6 degrees	9
Solo	Rov Solo String	Gggg.gP+pp.ppR+rr.rrDddd.ddAaa.aa<cr><lf>		3
Sonardyne	PSONNAV Gyro	$PSONNAV;hhmmss.sss;llll.llllll;a;yyyyy.yyyyyy;a;x.xxx;x.xxx;x.xx;a;d.ddd;...		35
Sonardyne	LNavUTC Gyro	Binary Lodestar INS message		36
Sonardyne	PSON2Gyro	:HHMMSSmmm RRRRRR PPPPPP HHHHHH xxxU...		42
Sonardyne	SolsticeWaterfallGyro	Metadataforwaterfallline		43
Sound Metrics	ArisGyro	ArisFrameHeader		45

Table 5 Gyro

2.5 Motion

NaviScan supports the following attitude data inputs:

Company	Instrument	Data string	Help	SBD id
	None		No data stored.	0
Applanix	ApxPosmvRPH	Binary data – group 102 or 103	Only UDP/IP	19
Atlas	STN Atlas RPH	BINARY STNATLAS packet (M-13)	BINARY STNATLAS format (M-13)	11
Atlas	HM Inbound Srv RPH	Special Atlas binary..		23

Balder	Balder ROV RPH	: a.aa d.dd hhhh rrrrr -ppppp		18
Benthos	BenthosIMUX200RPH	***** ggg.g ppp.p rrr.r		16
CDL	CDL RPH	H123.45P+12.345R+123.456 M123W1234.56U12.3<cr><lf>		21
CLD	CDL Tokimec2 RP	$PTVF;Pitch;Roll;Heading;....		27
Dynabase	Dynabase CRU 2RPH	Binary..		14
Dynabase	DynabaseCRU RPH	Binary..		17
Echologger	DASS710Motion	Simple point format 110; $dist; ha; va; hdg; pitch; roll; voltage	COM only; default baud 115200 8 N 1	43
EdgeTech	EdgeTech RPH	Binary..	UDP/IP	26
EdgeTech	Edgetech4600 RPH	Binary data		28
Eiva	Eiva InterpretRPH	Binary data...		36
EIVA	EElume Motion	Protobuf data ...		45
FFI	FFIAttit HuginRPH	$PFFIAAV;<time>;<heading>;<course>;<pitch>...		15
Free	Free RPH	Ex. "rrr.r ppp.p hhhh <CR><LF>"	Any fixed terminated string (max. 80 characters) given roll and pitch as decimal degree and heave as cm. Signs: Roll : Positive starboard down, Pitch : Positive bow up and heave positive below datum.	1
Hi-Target	iBeam Motion	Binary Hi-Target iBeam (starts with ascii 'IB')		37
Ixsea	IxseaRPH B&A 1RPH (Octans RPH)	$PHTRH;<pitch>;M/P;<roll>;B/T;<heave>;U/O<cr><lf>		13
Ixsea	Ixsea RPH GC2 RPH	$PHTRH;<pitch>;M/P;<roll>;B/T;<heave>;U/O<cr><lf>		24
Ixsea	Ixsea RPH SCOMEX RPH	$PSXN;id;user;ppp;rrr;ggg;pspeed;rspeed;gspeed*hh<cr><lf>		25
Ixsea	PhinsRovinsRPH	$HEHDT;90.30;/$PIXSE;HSATIT;h.hhh;/$PHOCT;01;hhmmss.sss;G;AA;HHH.HHH;<CR><LF>		30
Ixsea	PhinsRphBinNavRPH	Binary data...		32
Ixsea	PhinsRphLongBinNavRPH	Binary data...		40
Klein	KleinFishMotionExt	KleinFishMotionExt		38
Klein	KleinFishMotionHdr	KleinFishMotionHdr		39

Kongsberg	Seapath EM3000RPH	BINARY EM3000 format		10
Kongsberg	Seapath RPH (Ascii)	$PSXN;23;roll;pitch;head;heave*csum		12
MDL	MDL TrimcubeRPH	HggggP+ppppR+rrrr<cr><lf>	Search for „P" for pitch and „R" for roll.	4
MDL	MDL RPH	H1234P+1234R+1234<cr><lf>		22
OceanTools	OceanTools1218RPH	HaaaaP±bbbbR±ccccQ<cr><lf>/HyyyyP+ppppR+rrrrh+hhhhQ<cr><lf>		29
ØTC	ØTC special RPH	"ggg.gg,rrr.rr,ppp.pp,eeeeeeeeee.ee,nnnnnnnnn.nn,zzz.zz,oo.oo<cr><lf>"		7
OxTS	MCOM RPH	Binary data...		33
RDI	RDIRPH	$PRDID;P.PP;R.RR;HHH.HH;*CS<cr><lf>		31
reserved	reserved			20
RovTech	RovTech RPH	PITCH:pppp.p ROLL:rrrr.r<cr><lf>	Search for „PITCH:" for pitch and „ROLL:" for roll.	9
SBG Systems	Navsight Marine	Binary data (sbgECom protocol)	Virtual Serial Port over RS232/RS422/UDP[/TCP]	46
Seatex	Seatex MRU-5 RPH	Coded information giving roll, pitch and heave as binary floats. "qltRRRRPPPPHHHH"	Set-up to give continuously output (minimum 10 times a second) in the given order (Roll, Pitch, Heave)	2
Seatex	SeatexMRU-5DatRPH (Datasonics)	BINARY FLOATS 'rrrrppppyyyyhhhh'		5
Solo	ROV Solo StringRPH	Gggg.gP+pp.ppR+rr.rrDddd.ddAaa.aa<cr><lf>		8
Sonardyne	PSONNAV Motion	$PSONNAV;hhmmss.sss;llll.llllll;a;yyyyy.yyyyyy;a;x.xxx;x.xxx;x.xx;a;d.ddd;...		34
Sonardyne	LNavUTCMotion	Binary Lodestar INS message		35
Sonardyne	PSON2Motion	:HHMMSSmmm RRRRRR PPPPPP HHHHHH xxxU...		41
Sonardyne	SolsticeWaterfallMotion	Metadataforwaterfallline	UDP 127.0.0.1 port 0; only used to match with related sensors	42
SoundMetrics	ArisMotion	ArisFrameHeader	UDP frame header. Setup must be equal to sidescan setup for Aris FLS	44
Teledyne	TSS335B RPH	:xxaaaasnhhhhqnrrrrsnpppp<cr><lf>	Preferable set-up	3
Teledyne	TSS DMS-xx RPH	:xxaaaasmhhhhqmrrrrsmpppp<cr><lf>		6

Table 6 Motion sensors

2.6 Auxiliary

NaviScan supports the following time synch data inputs:

Company	Instrument	Data string	Help	SBD id
	NONE			0
Franatec	Methane	M<ppmv>; R<ppmv>; T<degC>; H<transmission %>; L<humidity %> <cr><lf>		4
Reson	Reason (Seabird)	T<temp>; C<cond>; P<pres>; S<sal>; c<svel> <cr><lf>		2
Saab	Pressure Saab	... *0001<psi><cr><lf>		5
Saab	SVS Saab	... <m/s><cr><lf>		5
Saiv	SAIV204	N<id> C<cond> T<temp> P<pres> c<svel> S<sal>		1
Valeport	VpBathyPack	dd/mm/yyyy hh:mm:ss <cond> <pres> <temp> <alti> [...] <cr><lf>		3

Table 7 Auxiliary

2.7 Rawdata

NaviScan supports the following time synch data inputs:

Company	Instrument	Data string	Help	SBD id
	None			0
	FREE			1

Table 8 Rawdata

2.8 Navigation

NaviScan supports the following position data inputs:

Company	Instrument	Data string	Help	SBD id
Applanix	Applanix POSMV	Special UDP/IP message packet 1, 102 or 103		10
Axyle	Axyle	hh:mm:ss EEEEEEEE.EE NNNNNNNN.NN<CR><LF>		1
EdgeTech	EdgeTech4600 Position	Binary data		11
EIVA	FREE	Ex. "EEEEEEEE.EE NNNNNNNN.NN hh:mm:ss.ss AAAA<CR><LF>"	Any fixed terminated serial string including easting and	0

			northing as decimal meters (Max. 100 characters). May also include timestamp and age information (in ms).	
EIVA	NaviPac	“0 nnnnnn hh:mm:ss.ss eeeeeeee.ee nnnnnnnn.nn a.aaa dddddd.dd ddd.dd kkkk.kkk R FFFFFFFF”	See NaviScan / NaviPac specification	3
EIVA	NMEA GGA	“$GPGGA,hhmmss.ssss,ttttt.tttt,N,ggggg.gggg,F,...<CR><LF>”	Data stored in WGS84 with user defined projection	6
EIVA	NaviPac ext.	"$<objectname>,hh:mm:ss.ss,a.aaa,EEEEEE.EE,NNNNNNN.NN,ZZZZ.ZZ,kkkk.kkkk..."		9
EIVA	Eiva Interpreted Grid	Binary data...		18
EIVA	Eelume Position	Protobuf data ...		25
Hi-Target	iBeam Position	Binary Hi-Target iBeam (starts with ascii 'IB')		19
Ixsea	PhinsRovinsPos	$PIXSE;HSPOS_;hhmmss.ss;llmm.mmmmm;H;LLmm.mmmmm;D;d.dd;a.ad;x.xx;y.yy;z.zz;d.dd;nn;c;e.e;n.n;m.mmmm;s.ssss;vvvv.v<CR><LF>		12
Ixsea	PhinsPosBinNav	Binary data...		13
Ixsea	PhinsPosLongBinNav	Binary data...		20
Klein	Klein Ship Position	Binary data...		21
Klein	Klein Fish Position	Binary data...		22
Klein	Klein Layback Position	Binary data...		23
Norcom	Norcom	"08 ROV RV ttttt.tt,ENG EEEEEEEE.EEE,NNG NNNNNNNN.NNN <CR><LF>"	Used by RESON for demo purpose.	2
Novatel	ProPak6Pos	Binary data...		14
ØTC	ØTC special	“ggg.gg,rrr.rr,ppp.pp,eeeeeeeeee.ee,nnnnnnnnnn.nn,….		7
OxTS	MCOM Position	Binary data...		15
QPS	QPS	hh:mm:ss.sss ll.lll eeeeeeee.ee nnnnnnnn.nn<cr><lf>		8

SBG Systems	Navsight Marine	Binary data (sbgECom protocol)	Virtual Serial Port over RS232/RS422/UDP[/TCP]	26
Sonardyne	PSONNAV Position		$PSONNAV;hhmmss.sss;llll.llllll;a;yyyyy.yyyyyy;a;x.xxx;x.xxx;x.xx;a;d.ddd;...	16
Sonardyne	LNavUTC Position	Binary Lodestar INS message		17
Sonardyne	SolsticeWaterfallPosition	Metadataforwaterfallline	UDP 127.0.0.1 port 0; only used to match with related sensors.	24
Stolt	Stolt Comex	aaaaaaaa hh:mm:ss EEEEEEEE.EE NNNNNNNN.NN .. <CR><LF>		5
UDI	UDI Special	hh:mm:ss;dd mmm yyyy;EEEEEEEE.EE;NNNNNNNN.NN;aaaaaaaaaaaa.aa<CR><LF>		4

Table 9 Navigation

2.9 Echosounder

NaviScan supports the following MBE systems. The type number will be of interest if you want to decode the SBD files

Company	Instrument	Data string	Help	SBD id
Atlas	ATLAS HYDROSWEEP Sidescan	HMInboundServer Sidescan		22
	No echosounder system		No data stored.	0
Outdated	SIS-3000	former datasonic sis3000 profiler		4
2G Robotics	ULS500			46
3DatDEPTH	3D at DEPTH SL			63
Atlas	ATLAS FANSWEEP15			14
Atlas	ATLAS FANSWEEP20			17
Benthos	Benthos C3DMBE			18
Benthos	Benthos C3DMBEDual			19

BlueView	BlueView	Reson7k packet type 7006 7004	Reson7k packet type 7006 7004	28
BlueView	BlueViewDual	Reson7k packet type 7006 7004	TCP and UDP; port 7000 for automatic subscription of needed packets	29
BlueView	BV2250-360	BVTRangeProfile	Using BlueView SDK; dll network connected; but must be defined in setup as UDP; with the IP address of the sonar head; and port can be 0 as it is not used	68
CathX	CathX PCD	PCD v.7 FIELDS range bearing tilt rgb DATA binary		53
CathX	CATHX			44
Echologger	DASS710 3D profiler	Simple point format 110; $dist; ha; va; hdg; pitch; roll; voltage	COM only; default baud 115200 8 N 1	67
EdgeTech	Edgetech4600/6205		TCP port 1901 and IP address of the computer on which Discover is run typ. same as naviscan 127.0.0.1	36
Hi-Target	iBeam	Binary Hi-Target iBeam	datagram starts with ascii 'IB'	56
Hyspec	Hyspec modular scan MK-II			13
Imagenex	ImagenexDT360			47
Imagenex	Imagenex DeltaT			25
Imagenex	Imagenex DeltaTDual			26
Imagenex	Imagenex 831L	DD-MMM-YYYY;HH:MM:SS.SSS;ppp.ppp;rrr.rrr;ddd.ddd;RRR.RRR;BBB.BBBcrlf		52
Imagenex	ImagenexDT100	D1P PROFILE POINT DATA FORMAT (binary)	UDP; port 4040 127.0.0.1 typically. Supports external sensors via mbe port; that also can retransmit to NaviPac (enable in echosounder setup)	64
Imagenex	ImagenexDT100 Dual	D1P PROFILE POINT DATA FORMAT (binary)	UDP; port 4040 127.0.0.1 typically. Supports external sensors via mbe port; that also can retransmit to NaviPac (enable in echosounder setup)	65

ITER Systems	Bathyswath			49
Klein	Klein Mbe			48
Kongsberg	EM3000 (Raw Range&Beam only)			10
Kongsberg	EM3000 Dual head (Raw R&B)			15
Kongsberg	EM3000 Special (Hugin)			16
Kongsberg	EM710/2040/302/122/70/M3			42
Kongsberg	EM710/2040/302/122/70/M3 Dual			43
Kongsberg	KMALL format	KMALL mbe #MRZ datagram	UDP old broadcast protocol; or multicast via 224.1.20.240 port 6240; or TCP port 13132. Recommeded to use multiport split to get #MRZ datagram alone.	58
Kongsberg	KMALL format Dual	KMALL mbe #MRZ datagram	UDP old broadcast protocol; or multicast via 224.1.20.240 port 6240; or TCP port 13132. Recommeded to use multiport split to get #MRZ datagram alone.	59
Kongsberg	Geoswath	Geoswath range angle intensity	UDP 192.168.1.222 port 5001; GeoSwath contains backscatter for each mbe beam; and no extra sidescan data thus no ss driver is needed.	51
Kongsberg Mesotech	SM2000			12
Kongsberg Mesotech	MS1000 Profiler			27
Kongsberg Mesotech	MS1000 Profiler Dual			45
L-3	Elac Hydrostar			23
L-3	Elac Hydrostar Dual			24
NewtonLabs	Laser Scanner	xyz points via Las format 1.1 and higher		57

Norbit	Norbit WBMS Bathy	Norbit's UAV suitable protocol	Binary data stream reader via TCP	69
Norbit	Norbit RawRange 7027	Norbit via Reson packet type 7027. Can be used for Norbit Winghead 1024 beam support.	TCP and UDP; port 7000 for automatic subscription of needed packets.	60
Norbit	Norbit RawRange 7027 Dual	Norbit via Reson packet type 7027	TCP and UDP; port 7000 for automatic subscription of needed packets	61
Picotech	PicoMB	Picotech PicoMB angle and range	UDP; sonar resides on 10.0.100.120:9000; that must be set in Port. PicoMB sends data exclusively to a single port; which default is 13000 set in sep. recv port. Naviscan pc must have IP 10.0.100.70	62
R2Sonic	R2Sonic 2000 series	R2Sonic 2000 BTH0	UDP port 4000. Mbe bathy is sent from the sonar head ip addr (undocumented); enter that ip addr for the mbe port; or ip 0.0.0.0 for not using ip filter. Only sidescan is sent from GUI ip	32
R2Sonic	R2Sonic 2000 series Dual	R2Sonic 2000 BTH0	UDP port 4000. Mbe bathy is sent from the sonar head ip addr (undocumented); enter that ip addr for the mbe port; or ip 0.0.0.0 for not using ip filter. Only sidescan is sent from GUI ip	33
R2Sonic	R2Sonic old format	R2Sonic old format		31
Reson	Reson7k RawRange7027	Raw range packet type 7027		34
Reson	Reson7k RawRange7027 Dual	Raw range packet type 7027		35
Reson	Reson Seabat7K old	Range packet type 7006; 7004	TCP and UDP; port 7000 for automatic subscription of needed packets	20
Reson	Reson Seabat7KDual old	Range packet type 7006; 7004	TCP and UDP; port 7000 for automatic subscription of needed packets	21

Reson	Reson HydroSweep	Reson packet type 7047	TCP and UDP; port 7000 for automatic subscription of needed packets; for atlas Reson interface; after CM version 3.3.0.11	55
Reson	Reson7k Integrated Dual	Raw range packet type 7027		54
Reson	SeaBat 81xx			3
Reson	SeaBat 81xx(2-Heads)			11
Reson	SeaBat S9001 R-Theta			1
Reson	SeaBat S9002 R-Theta			2
Sonardyne	SolsticeWaterfall	Solstice Waterfallbathy packet	UDP 127.0.0.1 port 0 only used to match with related sensors; set directory to monitor live updates to swf8 file in Nsconfig/Options/Global Parameters/MBESS/Solstice	66
Teledyne Odom	Odom Echoscan			5
Teledyne Odom	Odom ES3			30
Teledyne Odom	Odom MB1			37
Teledyne Odom	Odom MB1 Dual			38
Tritech	SeaKing Profiler			6
Tritech	SeaKing Profiler Dual head			7
Tritech	ST1000 Profiler			8
Tritech	ST1000 Profiler Dual head			9
Tritech	Gemini Multibeam Profiler			39
Tritech	Gemini Multibeam Profiler Dual			40
Wassp	Wassp3250			41
Wassp	Wassp S3	Binary Wassp G3 packet		50

Table 10 Echosounder

2.10 SideScan

NaviScan supports the following SideScan systems. The type number will be of interest if you want to decode the SBD files

Company	Instrument	Data string	Help	SBD id
	None		No data stored.	0
Atlas	ATLAS FANSWEEP SS			2
Atlas	ATLAS HYDROSWEEP Sidescan	HMInboundServer Sidescan		10
Benthos	Benthos C3DSS	Benthos C3D data	TCP	7
Benthos	Benthos Sidescan	Benthos Telegrams	TCP	6
Blueprint	Oculus M series	Intesity Image	TCP port 52100; ip addr of sonar must reside on same subnet as naviscan; ip can be discovered by oculus ViewPoint SW or factory set	48
EdgeTech	EdgeTech Sidescan Dual	EdgeTech Sidescan	TCP port 1901 and IP address of the computer on which Discover is run typ. same as naviscan 127.0.0.1	15
EdgeTech	EdgeTech SS	EdgeTech Sidescan	TCP port 1901 and IP address of the computer on which Discover is run typ. same as naviscan 127.0.0.1	36
Elac L3	Elac Hydrostar Sidescan	Elac Hydrostar SS		11
Imagenex	Imagenex872	Imagenex872		26
Imagenex	Imagnex 881L FLS	Imagenex 881L FLS connection		47
ITER Systems	Bathyswath			31
Kongsberg	GeoSwath DEPRECATED	GeoSwath range angle intensity	UDP 192.168.1.222 port 5001; DEPRECATED as same backscatter values will be obtained from the GeoSwath echosounder driver; use only echosounder	33
Kongsberg	KongsbergEA440	EchogramDg	UDP from SIS	49
Kongsberg	EM3000 Compact	Telegram 53h		4

Kongsberg	SsEM710/2040/302/122/70	EM datagram 78 + 89	UDP	28
Kongsberg	SsEM710/2040/302/122/70 Dual	EM datagram 78 + 89	UDP	29
Kongsberg	EM Kmall Snippet	KMALL #MRZ packet	UDP old broadcast protocol; or multicast via 224.1.20.240 port 6240; or TCP port 13132. Recommeded to use multiport split to get #MRZ datagram alone	37
Kongsberg	EM Kmall Snippet Dual	KMALL #MRZ packet	UDP old broadcast protocol; or multicast via 224.1.20.240 port 6240; or TCP port 13132. Recommeded to use multiport split to get #MRZ datagram alone	38
Kongsberg Hugin	Hugin XTF	XTF headers and ping packet...		3
Kongsberg Mesotech	Flexview FLS	IMB beamformed data format	TCP 127.0.0.1 ; enter the used IMB data port - typical 20001; (the used command port is implicit one less than IMB port e.g. 20000)	51
L3Klein	Klein Sidescan	IMB beamformed data format	TCP port 0; ip addr of the sonar; using Klein dll interfacing	30
Norbit	Norbit WBMS FLS	Norbit UAV suitable protocol	Binary data stream reader via TCP	53
Norbit	Norbit Snippet 7028	Norbit via Reson Snippet packet 7028	TCP and UDP; port 7000 for automatic subscription of needed packets	41
Norbit	Norbit Snippet 7028 Dual	Norbit via Reson Snippet packet 7028	TCP and UDP; port 7000 for automatic subscription of needed packets	42
Norbit	Norbit7k Sidescan 7007	Norbit via Reson Sidescan packet 7007	TCP and UDP; port 7000 for automatic subscription of needed packets	39
Norbit	Norbit7k Sidescan 7007 Dual	Norbit via Reson Sidescan packet 7007	TCP and UDP; port 7000 for automatic subscription of needed packets	40
Norbit	Norbit FLS	Norbit FLS via Reson WC packet	TCP and UDP; port 7000 for automatic subscription of needed packets	45
R2Sonic	R2Sonic 2000 Snippet	R2Sonic 2000 Snippet SNI0	UDP port: baseport + 6 (typ. port 4006). Snippet is sent from the GUI ip; which must be entered for	16

			this port; or ip 0.0.0.0 for not using ip filter	
R2Sonic	R2Sonic 2000 Snippet Dual	R2Sonic 2000 Snippet SNI0	UDP port: baseport + 6 (typ. port 4006). Snippet is sent from the GUI ip; which must be entered for this port; or ip 0.0.0.0 for not using ip filter	17
R2Sonic	R2Sonic old format	R2Sonic old format		14
R2Sonic	R2Sonic 2000 Truepix	R2Sonic 2000 Truepix TPX0	UDP port: baseport + 1 (typ. port 4001). Truepix is sent from the GUI ip; which must be entered for this port; or ip 0.0.0.0 for not using ip filter	20
R2Sonic	R2Sonic 2000 Truepix Dual	R2Sonic 2000 Truepix TPX0	UDP port: baseport + 1 (typ. port 4001). Truepix is sent from the GUI ip; which must be entered for this port; or ip 0.0.0.0 for not using ip filter	21
Reson	Reson Snippet 7028	Reson Snippet packet type 7028	TCP and UDP; port 7000 for automatic subscription of needed packets	18
Reson	Reson Snippet 7028 Dual	Reson Snippet packet type 7028	TCP and UDP; port 7000 for automatic subscription of needed packets	19
Reson	Reson Snippet 7058	Reson Snippet packet type 7058	TCP and UDP; port 7000 for automatic subscription of needed packets	43
Reson	Reson Snippet 7058 Dual	Reson Snippet packet type 7058	TCP and UDP; port 7000 for automatic subscription of needed packets	44
Reson	Seabat7k Sidescan	Seabat7k Sidescan packet 7007	TCP and UDP; port 7000 for automatic subscription of needed packets	8
Reson	Seabat7k SnippetDual OBSOLETE	Seabat7k Snippet packet 7008 obsolete		12
Reson	Seabat7k SnippetDual OBSOLETE	Seabat7k Snippet packet 7008 obsolete		13
Reson	Seabat7k Snippet-OBSOLETE	Seabat7k Snippet packet 7008 obsolete		9

Reson	SeaBat 81xx Sidescan			1
Reson	Seabat8K Snippet Dual	Seabat 8000 Snippet packet	UDP	23
Reson	Seabat8K Sidescan Dual	Seabat 8000 Sidescan packet	UDP	22
Reson	Seabat8k Snippet	Snippets telegrams	UDP	5
Sonardyne	SolsticeWaterfall	Solstice WaterfallImage packet	UDP 127.0.0.1 port 0 only used to match with related sensors; set directory to monitor live updates to swf8 file in Nsconfig/Options/Global Parameters/MBESS/Solstice	50
Sound Metrics	Aris FLS	Aris FLS connection	UDP frame data; IP of sonar; user can define a recv port; e.g. 50000	46
Sound Metrics	Didson FLS			35
Teledyne Marine	Blueview M900 FLS	Blueveiw SDK DLL	UDP via Blueview DLL; must be set to UDP and IP address of the wanted sonar; port 0; as it is not used.	52
Teledyne Odom	OdomMB1 Snippet	OdomMB1 Snippet packet		24
Teledyne Odom	OdomMB1 Snippet Dual	OdomMB1 Snippet packet		25
Tritech	Gemini FLS 620/720i(s/k/d)	DLL interfaced	Set port selection to 'No Port' and enter the wanted sonars ID (written on the sonar head) in the field for that in the port.	34
Wassp	SsWassp3250	SsWassp3250		27
WASSPS3	Wassp S3			32

Table 11 SideScan

2.11 Pipetracker

NaviScan supports the following pipetracker systems

Company	Instrument	Data string	Help	SBD id
	None		No data stored.	0

DOF Subsea	Stabbing	zzz.zzz ddd.ddd	Special format designed for DOF Subsea.	4
Eiva	Eiva Pipe	binary		8
Innovatum	Innovatum	Dd Mmm YyyyHh:Mm:SsRh MS Str OhOdC HdisHerrVdisVerr Sktt Accur		1
Optimal Ranging	OrionCableTracker	COM: $IISOL;1;1;... IP: Message ID 1 Locate Solution		7
Reson	Seabat7KPipeDetect	Binary 7k 2004 packet		6
Teledyne	TSS 340/440/440 mm	:TQaaaa bbb ccc ddd1111112222223333334444444 XX		2
Teledyne	TSS 350	SQsLLLL VVVV AAAAsCCCCsSSSs1111s2222s3333s4444s5555s6666 qq OR :SmmQsLLLLL...		3
Tinsley	Tinsley Cable tracker	Tag;Date;Time;Lat;Long;Datum;Range;SigStrength %<cr><lf>		5

Table 12 Pipetracker

2.12 Doppler log

NaviScan supports the following DVL data inputs:

Company	Instrument	Data string	Help	SBD id
	None		No data stored.	0
WaterLinked	DVL A50	DVL A50	ASCII/Serial(115200 8-N-1) or JSON/TCP:16171	13
EDO	EDO 3050 287.5 kHz (low)	EDO 3050 287.5 kHz (low)		1
EDO	EDO 3050 596.5 kHz (med)	EDO 3050 596.5 kHz (med)		2
EDO	EDO 3050 894.5kHz (high)	EDO 3050 894.5kHz (high)		3
Eiva	Eiva Interpreted	Eiva Interpreted		9
Nortek	Nortek DF21-DF22	DF21 bottom track DF22 water track	COM, UDP and TCP supported; UDP recommended	10

SBG Systems	Navsight Marine DVL	Binary data (sbgECom protocol)	Virtual Serial Port over RS232/RS422/UDP[/TCP]	12
Teledyne Marine RD Instruments	Wayfinder DVL	Wayfinder DVL (Serial binary)	COM Binary DVL	11
Teledyne RDI	RDI DVL PD6 (Ascii)	DVL-S1 PD6 (Ascii)		5
Teledyne RDI	RDI DVL PD0 (Binary)	DVL-S1 PD0 (Binary)		6
Teledyne RDI	RDI DVL PD4 (Binary)	DVL-S1 PD4 (Binary)		7
Teledyne RDI	RDI DVL PD3 (Binary)	DVL-S1 PD3 (Binary)		8
Ulvertech	Ulvertech D.A.T.S	" XXXXYYYYZZZZ xxxxyyyyzzzz CLKK CHKS \r\n"		4

Table 13 Doppler log

2.13 Laser Scanner

NaviScan supports the following DVL data inputs:

Company	Instrument	Data string	Help	SBD id
EIVA	None		No data stored.	0
Carlson	Dynascan M250	@AABBCCDHHmmSSXXXXXXXXDDMMYYYY<LF>$RRSSGG<LF>		1
Velodyne	HDL-32E VLP-16 PuckLITE PuckHR	Binary 1206 bytes	UDP Port 2368	2
Quanergy	Quanergy-M8	Binary...	TCP Port 4141	3
Ouster	OS1 Family	Binary...	UDP:7502 + TCP:7501	4

Table 14 Laser Scanner

2.14 Theoretical profile

NaviScan supports the following data inputs for online presentation of theoretical profiles.

Type	String Lay-out	Note	SBD id
None		No data stored.	0
FREE	Binary packets – contact EIVA for details	30 points specifying a profile	1
HAM dredging	Binary packets – contact EIVA for details	30 points specifying a profile	2
Tideway	Binary packets – contact EIVA for details	Depth relative to pipe.	3

Table 15 Theoretical profile

3 NaviScan SBD logging format

This section describes the format of the raw data files and an example of the disk space required for logging.

1. Header Record
2. Multibeam echosounder Record
3. Doppler Velocity Record
4. Pipetracker Record
5. Position Record
6. Filtered position Record
7. Sound velocity profile Record
8. Theoretical profile Record
9. Sidescan image Record
10. Target Record
11. End of file Record

From version 7.0

12. Motion record
13. Gyro record
14. Bathy record

From version 7.1

15. Projection&Ellipsoid record
16. Datumshift record
17. Logcontrol record

From version 7.2

18. Multiple pos record
19. Raw data record
20. Sidescan head 2 record

From version 8.0

21. Additional timeoffset and usernames information for Multiple sensors

From version 8.2

22. Auxiliary sensor + pragma pack 1 and complete refactoring of setup structure

Note: the following structures are compiled with 8 bytes alignment.

3.1 File structure

SBD files all have the following structure (file with 3 records):

Size	Structure type
Xx bytes	LOG_HEADER
20 bytes	HEAD_LOGPACKET8_0
n * 40 bytes	SUBPACKET6_0
n bytes	Raw data
20 bytes	HEAD_LOGPACKET8_0
n * 40	bytes SUBPACKET6_0
n bytes	Raw data
20 bytes	HEAD_LOGPACKET8_0
n * 40 bytes	SUBPACKET6_0
n bytes	Raw data

The HEAD_LOGPACKET8_0 and SUBPACKET6_0is described in the following two chapters.

3.1.1 Packet header

Each data packet contains this structure. New header structure with absolute time included:

```
New header structure with abs time included: (from ver 7.7)
typedef struct PacketHead8_0
{
char spec[2];          // 2 bytes "? " Id for data record
unsigned short nSubPackets;// 2 bytes Number of subpackets before data - may be zero
long rel_time;         // 4 bytes Timestamp in ms (relative stamp) timeGetTime src
TIMEVAL abs_time;      // Timestamp of packet in absolute time - when is data valid
```

```
unsigned long size;    // 4 bytes Length of record data that follows
} HEAD_LOGPACKET8_0;    // 20 bytes
```

spec : The spec variable tells what kind of data packet is following.
“0 “ : Position
“1 “ : Multibeam echosounder head 1
“2 “ : Multibeam echosounder head 2
“3 “ : Pipetracker
“4 “ : Doppler log
“5 “ : Filtered position
“6 “ : Sound velocity profile
“7 “ : Theoretical profile
“8 “ : Sidescan image (head 1 in dual setup)
“F “ : Sidescan head 2 image
“9 “ : Target
“M “ : Motion
“G “ : Gyro
“B “ : Bathy
“A “ : Multiple pos
“X “ : Auxiliary
“Y “ : HuginMon sub bottom
“RD“ : Raw data
“PE“ : Projection & Ellipsoid
“DS“ : Datumshift
“LC“ : Logging control
“S “ : Start of file (LOG_HEADER)
“E “ : End of file
size : The size is the size of the raw data after the sub packets.
time : The time of the raw data. The time is relative to the LOG_HEADER StartTime.
nSubPackets : The number sub packets that follows the head packet.

From ver 7.7 the Huge packet disappears again as the new header has 4byte size

Old version huge packets:

For the “C” Huge Sidescan one must pay special attention to the change of nSubPacket and size fields. If the C specifier is used in the file (for the time being only for Benthos C3D sidescan), the file reader must first look at the specifier and the decide the packet size, for each packet in the file. For C spec the following header is valid:

Old version packethead for std and huge packets:

```
typedef struct PacketHead8_0B
{
```

```
        char spec[2];          // 2 bytes: "? " Id for data record
        unsigned short nSubPackets; // 2 bytes:length of record data
        DWORD time;            // 4 bytes: Timestamp in ms
        long size; // 4 bytes Number of subpackets before data
                               // ====================================
} HEAD_LOGPACKET8_0B; // 12 bytes
typedef struct PacketHead6_0
{
        char spec[2];                  // 2 bytes: "? " Id for data record
        unsigned short size;           // 2 bytes:length of record data
        DWORD time;                    // 4 bytes: Timestamp in ms
        unsigned long nSubPackets;     // 4 bytes Number of subpackets before data
                                       // ====================================
} HEAD_LOGPACKET6_0; // 12 bytes
```

3.1.2 Sub Packet

This structure contains all the data needed for correction of multibeam, pipetracker, etc. data.

```
typedef struct SubPacket
{
        long time;             // 4 bytes Time of values in packet
        float Depth;           // 4 bytes Depth in meters
        float Tide;            // 4 bytes Tide in meters
        float HeightGPS;       // 4 bytes GPS height in meters
        float Gyro;            // 4 bytes Heading in degrees
        float GyroC_O;         // 4 bytes Dynamic Gyro C-O value in degrees
        float Roll;            // 4 bytes Roll in degrees
        float Pitch;           // 4 bytes Pitch in degrees
        float Heave;           // 4 bytes Heave in meters
        float Aux1;            // 4 bytes multi purpose
                               // ====================================
} SUBPACKET;                                  // 40 bytes
```

3.2 FILE ID in SBD files

Important:

SBD files logged from ver 9 has initially a 12 byte FILE ID identifier as the very first place in a SBD file, this id is not included in a Naviscan.BIN setup file

This is used to inform the user that a 20 byte HEAD_LOGPACKET8_0 is used instead of a 12 byte HEAD_LOGPACKET6_0.

```
typedef struct SbdFile_Id //very first 12 bytes of newer sbd files
{
        char id[4]; //"_SBD"
        BYTE ver[4]; //8 0 8 0 = binary HEAD_LOGPACKET8_0main ,HEAD_LOGPACKET8_0sub,
NBLOGVERSIONMAIN, NBLOGVERSIONSUB
        char future[4]; //all NULLs
}SBDFILE_ID;
```

After those 12 bytes follow the standard definition of the file

3.3 Header Record

```
#define NBLOGVERSIONMAIN      8                     // Main version number
#define NBLOGVERSIONSUB       2                     // Sub version number

typedef struct NavibatLogHeader8_2
{
        SYSTEMTIME StartTime;          // Local time
        NBSETUP8_2 nbc;               // Naviscan Configuration
        char Version[16];             // "NaviBat V8.2"
        int Timezone;
        BOOL DST;
} LOG_HEADER8_2;

typedef struct NavibatLogHeader8_0
{
        SYSTEMTIME StartTime;         // Local time
        NBSETUP8_0 nbc;               // Navibat Configuration
        char Version[25];             // "NaviBat V8.0"
} LOG_HEADER8_0; /* ??? BYTES */

Previous version 7.7:
typedef struct NavibatLogHeader7_2
{
        SYSTEMTIME StartTime;         // Local time
        NBSETUP7_2 nbc;               // Navibat Configuration
        char Version[25];             // "NaviBat V7.2"
} LOG_HEADER7_2; /* ??? BYTES */
```

At start of line (beginning of a new file), a header record is written, eg <StartTime> <NB set-up> <Version>

- **Start time**
  Gives the start date and time for current line. Given in C notation SYSTEMTIME
- **NB setup**
  Specifies the current system set-up. Copy of NaviScan.bin
- **Version**
  Gives the current NaviScan version number in ASCII

The implementation of the newest header record (LOG_HEADER) is shown below:

```
#define MAX_MULTIPLE_SENSORS          10 // Maximum 10 additional sensors
#define MULTIPLE_MOTION               10
#define MULTIPLE_GYRO                 11
#define MULTIPLE_BATHY                12
#define MULTIPLE_BATHY_GPSHEIGHT      13
#define DOPPLERLOG                    14
#define MULTIPLE_AUXILIARY            15
#define MULTISEN_MAXNAMES             4
#define MULTISEN_NAMELEN              16
#define SENSORNAMELEN                 16
#define NAMELEN8_2                    32

#pragma pack(push,1)

typedef struct Positionsetup8_2
{
        PORTSETUP Port;         // Port settings
        short Kind,             // Identification of positioning system
           EastOffset,          // Offset to easting "EEEEEEEE.EE" in packet
           NorthOffset,         // Offset to northing "NNNNNNNN.NN" in packet
           TimeOffset,          // Offset to timestamp "hh:mm:ss" in packet
           AgeInPacket,         // Offset to age of data in packet (in ms) -1 if not used
           FixedAge,            // Age of data (in ms), -1 if not used
           UseGPSTim,           // Used absolute GPS time
           future;
        double PSX,PSY,PSZ;             // Distance from POS to CRP in m
        long future2;
        long timeoffsetms;
        char sensorname[NAMELEN8_2];
} POSITIONSETUP8_2;
```

```
typedef struct Gyrosetup8_2
```

```
{
        PORTSETUP Port;               // Port settings
        short Kind;                   // Identification of gyro system (or free)
        short GyroOffset;             // Offset to gyro info "ggg.g" in Packet
        BOOL UseGyroCorrection; // (bool) use dynamic gyro correction
        double C_O;                   // C_O value for this sensor
        long future;
        long timeoffsetms;
        char sensorname[NAMELEN8_2];
} GYROSETUP8_2;
```

```
typedef struct Motionsetup8_2
{
        PORTSETUP Port;         // Port settings
        short Kind;             // Identification of motion system (or free)
        short UseHeave,         // Enable(1)/Disable(0) use of heave
               RollOffset,      // Offset to Roll info "rrr.r" in packet
               PitchOffset,     // Offset to pitch info "ppp.p" in packet
               HeaveOffset,     // Offset to heave info "hhhh" in packet (in cm)
               future;
        long timeoffsetms;
        double RSX,RSY,RSZ;    // Distance from POS to RPH reference point in m
        double Roll_C_O,        // C_O value for this sensor
               Pitch_C_O,
               Heave_C_O;
        char sensorname[NAMELEN8_2];
} MOTIONSETUP8_2;
```

```
typedef struct Bathysetup8_2
{
        PORTSETUP Port;         // Port settings
        short Kind;             // Identification of motion system (or free)
        short UseBathy;         // Use bathy depth ?
        short BathyOffset;      // Offset to bathy "ddd.dd" in Packet (in m)
        short bathyAge;         // fixed age on bathy data in ms.
        double BSX,BSY,BSZ;     // Distance from POS to bathy ref point in m
        double C_O;             // C_O value for this sensor
        float densityWater;     // kg/dm3
        float gravity;          // m/s2
        float presureSurface;  // atm.
        long timeoffsetms;
        char sensorname[NAMELEN8_2];
} BATHYSETUP8_2;
```

```
typedef struct Auxiliarysetup8_2
{
        PORTSETUP Port;        // Port settings
        short Kind;            // Identification of system
        short future;
        long timeoffsetms;
        float customA,customB;
        double ASX,ASY,ASZ; // Distance from POS to ocean reference point in m
        char sensorname[NAMELEN8_2];
}AUXILIARYSETUP8_2;
```

```
typedef struct Rawdatasetup8_2
{
        PORTSETUP Port;        // Port settings
        short Kind;            // Identification of Rawdata type (or free)
        short future;
        long timeoffsetms;
        double RDSX,RDSY,RDSZ; // Distance from POS to Rawdata ref point in m
        char sensorname[NAMELEN8_2];
} RAWDATASETUP8_2;
```

```
typedef struct MultipleSensors8_2
{
        short Pnumber;                // Number of additional position sensors
        short Gnumber;                // Number of additional gyro sensors
        short Mnumber;                // Number of additional motion sensors
        short Bnumber;                // Number of additional bathy sensors
        short Anumber;                // Number of additional auxiliary sensors
        short Rnumber;                // Number of additional rawdata inputs
        short future;
        short future2;
        POSITIONSETUP8_2 PositionSetup[MAX_MULTIPLE_SENSORS];
                        // Array of additional position sensors setup structure
        GYROSETUP8_2 GyroSetup[MAX_MULTIPLE_SENSORS];
                        // Array of additional gyro sensors setup structure
        MOTIONSETUP8_2 MotionSetup[MAX_MULTIPLE_SENSORS];
                        // Array of additional motion sensors setup structure
        BATHYSETUP8_2 BathySetup[MAX_MULTIPLE_SENSORS];
                        // Array of additional bathy sensors setup structure
        AUXILIARYSETUP8_2AuxiliarySetup[MAX_MULTIPLE_SENSORS];
                        // Array of additional aux interpreted sensors setup structure
        RAWDATASETUP8_2 RawdataSetup[MAX_MULTIPLE_SENSORS];
                        // Array of rawdata sensors setup structure
} MULTIPLESENSORS8_2;
```

```
typedef struct NaviBatSetUpStructure8_2
{
        unsigned short size;           // Size of entire packet in bytes
        short versionMain;             // Main version number ( 8 )
        short versionSub;              // Sub version number ( 2 )
        char Date[20];                 // Date for last update of NAVIBAT.BIN
        char Time[10];                 // Time for last update of NAVIBAT.BIN
        char Note[100];                // Free notes
        // 17*8 = 136 bytes
// ====================================================================
// Setup of Sensors
// ====================================================================
// --- Echosounder Head 1 ---
        short SKind;                   // Identification of echosounder system
        BYTE SHasSepRx;                // indicator for use of separate rx antenna
                                       //(primary is then the tx antenna)
        BYTE Sfuture2;
        int beamsPrScan;

        PORTSETUP SPort;               // Port settings
        double PSX,PSY,PSZ;            // Dist. from POS to mbe ref. point in m
        double SBRolled,               // Roll mount Angle for 1st head in deg
               SBPitched,              // Pitch mount Angle for 1st head in deg
               SBHeading;              // Heading mount angle for 1st head in deg
        double PSXrx,PSYrx,PSZrx;// Dist. from POS to eventual separate mbe rx antenna ref.
point
        in m
        double SBRolledrx,             // Roll mount Angle for 1st rx head in deg
               SBPitchedrx,            // Pitch mount Angle for 1st rx head in deg
               SBHeadingrx;            // Heading mount angle for 1st rx head in deg
        char Sfuture[16];              // reserved for future use
// --- Echosounder Head 2 ---
        PORTSETUP S2Port;              // Port settings
        double PS2X,PS2Y,PS2Z;         // Offsets from POS to 2nd mbe head in m
        double SB2Rolled,              // Roll mount Angle for 2nd head deg
               SB2Pitched,             // Pitch mount Angle for 2nd head deg
               SB2Heading;             // Heading mount angle for 2nd head deg
        double PS2Xrx,PS2Yrx,PS2Zrx;// Dist. from POS to eventual separate mbe2
                                       //rx antenna ref. point in m
        double SB2Rolledrx,            // Roll mount Angle for 2nd rx head deg
               SB2Pitchedrx,           // Pitch mount Angle for 2nd rx head deg
               SB2Headingrx;           // Heading mount angle for 2nd rx head deg
        char SB2future[16];            // reserved for future use

// --- Sidescan ---
```

```
        short SDKind;                   // Identification of sidescan system
        short SDfuture1;
        float SDfuture2;
        PORTSETUP SDPort;               // Port settings
        double PSDX,PSDY,PSDZ;          // Offsets from POS to SideScan head in m
        double SDRolled,                // Roll mount Angle for head deg
               SDPitched,               // Pitch mount Angle for head deg
               SDHeading;               // Heading mount angle for head deg
        PORTSETUP SD2Port;              // Port2 settings
        double PSD2X,PSD2Y,PSD2Z;// Offsets from POS to SideScan head2 in m
        double SD2Rolled,               // Roll mount Angle for head deg
               SD2Pitched,              // Pitch mount Angle for head deg
               SD2Heading;              // Heading mount angles for head deg
        char SDfuture[16];              // reserved for future use

// --- Pipetracker ---
        PORTSETUP PTPort;               // Port settings
        short PTKind;                   // Identification of bathy system (or free)
        short PipeDiameter;             // Pipe diameter in cm
        float PTfuture2;
        double PTSX,PTSY,PTSZ;          // 1 Distance from POS to Pipetracker in m
        char PTfuture[16];              // reserved for future use

// --- Doppler Log ---
        PORTSETUP DLPort;               // Port settings
        short DLKind;                   // Identification of Doppler system
        short DLfuture2;
        float DLfuture3;
        double DXOffset,
               DYOffset,
               DZOffset;
        double DRollOffset,
               DPitchOffset,
               DYawOffset;
        char DLfuture[16];              // reserved for future use

// --- Logging control (start/stop, filename) ---
        PORTSETUP LPort;                // Port settings
        short LKind;                    // Identification of control input
        short LNameOffset;              // Offset to file name in string
        float Lfuture2;
        char LStartId[16];              // id for start logging
        char LStopId[16];               // id for stop logging
        char Lfuture[16];               // reserved for future use
```

```
// --- Theoretical Profile ---
        PORTSETUP TPPort;             // Port settings
        short TPKind;                 // Identification of the input
        short TPfuture2;
        float TPfuture3;
        char TPfuture[16];            // reserved for future use;

// --- GPS Clock ---
        PORTSETUP GPSPort;            // Port settings
        short GPSKind;                // Identification of the input
        short leapSeconds;            // C-O on GPS time
        short ppsEvent;               // type of event for pps
                                      //(outdated: time from string to PPS)
        BYTE bUsePPS;                 // if false delayMs is used
        BYTE GPSfuture2;
        char GPSfuture[16];           // reserved for future use

// --- Sound velocity profile ---
        PORTSETUP SVPort;             // Port settings
        short SVKind;                 // Identification of the input
        short SVfuture2;
        float SVfuture3;
        char SVfuture[16];            // reserved for future use;

// --- Online Filter ---
        PORTSETUP OFPort;             // Port settings
        short OFKind;                 // Online filter Off 0 HAS 1
        short TimeLim;                // Max time without accep. data
        float MaxAcc,                 // Max allowed acceleration m/s*s
               MaxOff,                // Max allowed position jump x,y
               Gain;                  // Filter Gain 0-1
        char OFfuture[16]; // reserved for future use

// --- C-O ---
        double GyroCorrection, // Primary sensor Gyro correction in degree
                RollCorrection,// Primary sensor Roll correction in degree
                PitchCorrection,// Primary sensor Pitch correction in degree
                HeaveCorrection,// Primary sensor Heave correction in m
                BathyCorrection;// Primary sensor Bathy correction in m
        double BSZ; // for mbe Draft calc when no bathy is present

MULTIPLESENSORS8_2 multiplesensors;

} NBSETUP8_2;
```

Note the byte alignment is default 8 byte in Eiva, and overruled locally to (eg.1) for securing backwards size compatibility when the char -future[16] has been replaced with something else, (eg. char Bfuture[16];).

3.3.1 UserNames and Dopplerlog offsets addition.

Note that NBSETUP7_2 now contains arrays for XXXUserNames at the end. These arrays reserves place for the names of the first 4 multiple sensor of the for types pos, gyro, motion and bathy. For more than 4 sensors of each type no name can be saved, for those the default system name are used implicit in NaviScan. It is limited to 4 places of each type, to be able to

contain the names into the former char AddFuture[256] array, and this way implement user names without changing the size and outlay of the NBSETUP7_2 structure, for maintaining backwards comp ability. Therefore it has not been necessary to change the name of the structure, while incorporating this latest change.

At the same time we have changed the Doppler log setup to include Doppler mounting offset angles and position. These offset can only be hold in the setupfile, but will for now not be used for any calculation and any view in NaviScan. Again we have reused half of the char Bfuture[16] and the char DLfuture[16], for this, thereby maintained the same size and outlay of the NBSETUP7_2 structure, with respect to backwards compatibility.

```
// --- Doppler Log ---
        #pragma pack(push,1)
        float DRollOffset;
        float DPitchOffset;
        #pragma pack(pop)
        short DLKind;          // Identification of Doppler system
        PORTSETUP DLPort;      // Port settings
// char DLfuture[16];          // reserved for future use
        #pragma pack(push,1)
        float DYawOffset;
        float DXOffset;
        float DYOffset;
        float DZOffset;
        #pragma pack(pop)
```

3.4 TimeBox interfaced sensors

Contrary to all other interface types, TimeBox sensors are preceded with 24 bytes NAVITAGHDR before the original data content from the comport:

```
// NaviTag.h
```

```
// NaviTag.h
// header file for the data structure in NaviTag
//always INTEL byte order
#pragma pack(push,1)
typedef struct tag_NAVITAGHDR
{
        char    Ident[4];               // Packet starter: always "EIVA"
        INT32   Length;                 // Length of packet - remaining of entire packet
        INT32   VersAndDomain; // First 16 bit version number (1)
                                        // Last 16 bit domain id - unique time box no.
        INT32   InsId;                  // Id of instrument - version 1 COM port number
        TIMEVAL time_stamp;             // Timestamp of reception of telegram (first bit) }
NAVITAGHDR;
typedef struct tag_NAVITAGDATA
{
        NAVITAGHDR hdr;
        char    data[1];                // In fact [length-16];
} NAVITAGDATA;
// The raw telegram - not really a part of
//structure - but follows right after the header
#pragma pack(pop)
```

3.5 Gyro Record

This record consists of a head packet, followed by the gyro record. The first (primary) sensor of this type is stored in this record and in the sub packages to ensure backwards compatibility.

Layout

```
typedef struct GyroLogPacket
{
        short Seq;              // Sequence number of sensor (1 - x)
        float Gyro;             // Interpreted value
        float Corr;             //speed or RTK corr without meridian corr
        float MeridianCorr;         //meridian corr
        float Spare;            //for future use
        short Raw_length;       // Length of raw package (Normally not used)
```

```
        char pRaw[1];  // Pointer to raw package (Normally not used)
}GYRO_PACKET;
```

3.6 Motion Record

This record consists of a head packet, followed by the motion record. The first (primary) sensor of this type is stored in this record and in the sub packages to ensure backwards compatibility.

Layout

```
typedef struct MotionLogPacket
{
        short Seq;              // Sequence number of sensor (1 - x)
        float Roll,
         Pitch,
         Heave;                 // Interpreted values
        short Raw_length;       // Length of raw package (Normally not used)
        char pRaw[1];           // Pointer to raw package (Normally not used)
}MOTION_PACKET;
```

3.7 Bathy Record

This record consists of a head packet, followed by the bathy record. The first (primary) sensor of this type is stored in this record and in the sub packages to ensure backwards compatibility.

Layout

```
typedef struct BathyLogPacket
{
        short Seq;                  // Sequence number of sensor (1 - x)
        float bathy;                // Interpreted depth (m)
        float altitude;             // Interpreted altitude
        float presure;              // Interpreted pressure (dBar)
        short Raw_length;           // Length of raw package (Normally not used)
        char pRaw[1];               // Pointer to raw package (Normally not used)
}BATHY_PACKET;
```

3.8 Multiple position Record

This record consists of a head packet, followed by the multiple position record. The first (primary) sensor of this type is stored in this record and in the old position record to ensure backwards compatibility.

Layout

```
typedef struct PositionLogPacket      // Only used with packettype "P "
{
        short Seq;                    // Sequence number of sensor (1 - x)
        double dEasting;              // Interpreted Easting pos
        double dNorthing;             // Interpreted Northing pos
        short Raw_length;             // Length of raw package
        char pRaw[1];                 // Pointer to raw position string
}POS_PACKET;
```

3.9 Raw data Record

This record consists of a head packet, followed by the raw data record. This data contains only the raw data as a binary block. This data has not been interpreted in any way.

Layout

```
typedef struct RawdataLogPacket       // Only used with packettype "R "
{
        short Seq;                    // Sequence number of sensor (1 - x)
        short Raw_length;             // Length of raw package
        char pRaw[1];                 // Pointer to raw data string
}RAWDATA_PACKET;
```

3.10 Auxiliary Record

This record consists of a head packet, followed by the auxiliary data record. This data contains only the raw data as a binary block. This data has not been interpreted in any way.

Layout

```
typedef struct AuxiliaryLogpacket{
        float Vals[8]; // interpreted custom data values, unused vals filled with FLT_MAX
        short Seq;             // Sequence number of sensor (1 - x)
        short Raw_length;      // Length of raw package (Normally not used)
```

```
        char pRaw[1];          // Pointer to raw package (Normally not used)
} AUXILIARY_PACKET;
```

3.11 Projection & Ellipsoid Record

This record consists of a head packet, followed by the Projection & Ellipsoid record.
This packet is only saved, when running NaviScan with **Loggingcontrol** and using NaviPac 3.3x or higher.

Layout

```
typedef struct ProjEllipLogPacket
{
        // Version control
        short Version_Major;          // 1
        short Version_Minor;          // 1

        // Primary parameters
        char sEllipsoidName[128];
        double SemiMajorAxis;         // [meters]
        double InverseFlattening;
        unsigned char ProjectionType;
        double OrgScale;       // Pointscale factor at the ellipsoid origin
        double FirstParallel;  // First parallel in Lamberts conical [degrees]
        double SecondParallel; // Second parallel in Lamberts conical [degrees]
        double OrgLambda;       // Origin lambda (lon) on the reference ellipsoid [degrees]
        double OrgPhi;         // Origin phi (lat) on the projection ellipsoid [degrees]
        double OrgEasting;     // Origin easting on the projection plane [meters]
        double OrgNorthing;    // Origin northing on the projection plane [meters]

        // Secondary parameters
        unsigned char UTMzone; // Derived from "OrgLambda" [1..60]

        // Raw string
        short Raw_length;      // Length of raw package
        char pRaw[1];          // Pointer to raw string

}PROJELLIP_PACKET;
```

3.12 Datum shift Record

This record consists of a head packet, followed by the Datumshift record.

This packet is only saved, when running NaviScan with “Logging control” and using NaviPac 3.3x or higher.

Layout

```
typedef struct DatumShiftLogPacket
{
        float   Tx;     // Transformation parameter X - meters
        Float   Ty;     // Transformation parameter Y - meters
        float   Tz;     // Transformation parameter Z - meters
        short   method; // How to perform the shift
        double  Rx;     // Rotation parameter X - [degrees]
        double  Ry;     // Rotation parameter Y - [degrees]
        double  Rz;     // Rotation parameter Z - [degrees]
        double  PPM;    // scaling factor transformation - 1/1000000

        // Raw string
        short Raw_length; // Length of raw package (Normally not used)
        char pRaw[1]; // Pointer to raw string

}DATUMSHIFT_PACKET;
```

3.13 Utility Record

This record consists of a head packet, followed by the Datumshift record.

This packet is only saved, when running NaviScan with “Logging control” and using NaviPac 3.3x or higher.

Layout

```
typedef struct UtilityPacket
{
        short Version_major;   // XX.
        short Version_minor;   // .XX
        short Type;                     // Type of data...(Not used yet)
        char Yaw_str[16];               // "YAW=[sign]XXX.XXXXXX" max resolution
        char spare[100];                // spare for future expanding..
}UTILITY_PACKET;
```

3.14 Logcontrol Record

This record consists of a head packet, followed by the logcontrol record.

This packet is only saved, when running NaviScan with “Logging control” of type "NaviPac (Passive)" and using NaviPac 4.

This record is mainly for use of other external importers, which need further information.

Layout

```
typedef struct LogcontrolPacket
{
        // Raw string
        short Raw_length;      // Length of raw package (Normally not used)
        char pRaw[1];          // Pointer to raw string

}LOGCONTROL_PACKET;
```

3.15 Multibeam echosounder Record

A multibeam echsounder record consists of a head packet, one or more subpackets and the raw multibeam data, as described in chapter 4.1. If head.spec[0] is „2" it means that the raw data is from the second head in a dual head setup. The multibeam type can be determined from the LOG_HEADER.

The raw data must be interpreted according to description from the echo sounder manufacturer, and Eiva can provide a set of header files, describing how Eiva interprets the raw data.

3.16 SideScan Record

The raw data contains the sidescan image packet from the sidescan system.

The raw data must be interpreted according to description from the echo sounder manufacturer, and EIVA can provide a set of header files, describing how EIVA interprets the raw data.

3.17 Pipetracker

```
typedef struct PipeTrackerLogPacket
```

```
{
                              // =============== Record data ==============
        float EditX, EditZ;   // 8 bytes: Edit value of pipe position,
                              // overrules packet
        short Flags;          // 2 bytes: TRUE if pipe is covered, FALSE
                              // otherwise
        float x, z;           // 8 bytes: Raw value of pipe position
        long quality;         // 4 bytes: Quality of raw values
        BYTE EditQ;           // 1 bytes: saved Q byte from Raw packet when Edited sensor
        BYTE NotUsed;         // 1 bytes:
        short FutureUse[7];    // 14 bytes:
        char Data[1];         // n bytes: Raw packet from pipetracker
                              // ==========================================
} PIPETRACKER_LOGPACKET; // n + 38 BYTES
```

3.18 DopplerLog Record

This packet contains the raw Doppler log data along with Gyro, Roll and Pitch values.

3.19 Position Record

In a position record the raw data contains the string received from the navigation system.

3.20 Filtered Position Record

In a filtered position record the raw data contains a string with the following format.
012345678901234567890123456789012345678901234567890123456789
ttttttttttt eeeeeeee.ee nnnnnnnn.nn s.ss d.dd<cr><lf>
t - string from navigation system containing the time (NaviPac : hh:mm:ss.ss)
e - easting
n - northing
s - standard deviation
d - distance from raw position

4 Time stamping

The design of **NaviScan** has been performed with special consideration to time tagging, since it is of highest importance that each scan is corrected with motion and heading values from exactely the same time the scan was collected.

The most trustworthy result is achieved when values from different sensors are collected at the same time. Since the single units supply data with different intervals this is not possible in practice. Consequently, as a minimum the system must calculate an average value, at the corresponding time of the controlling unit, eg SeaBat.

To obtain maximum accuracy, we have implemented I/O as a multi-thread process running on the real-time queue. Each sensor is supported by a dedicated thread, which allows exact time-tagging of data, even when data is arriving asynchronously.

Data is time tagged relative to the internal pc clock with an accuracy of a few milliseconds.

Sub packets are produced with time interpolated value, and logged but also used for online correction for the displayed scans.

Finding interpolated values:

Let echosounder supply data corresponding to the times T0 T1 T2 T3 etc. and the gyro corresponding to the times t0 t1 t2 t3 etc. What gyro value must be used for SeaBat value to the time Ti. If a j with Ti=tj exists, the gyro value corresponding to the time tj is used. If this is not the case the system must find a k so that tk<Ti<t(k+1) and a gyro value is calculated as follows:

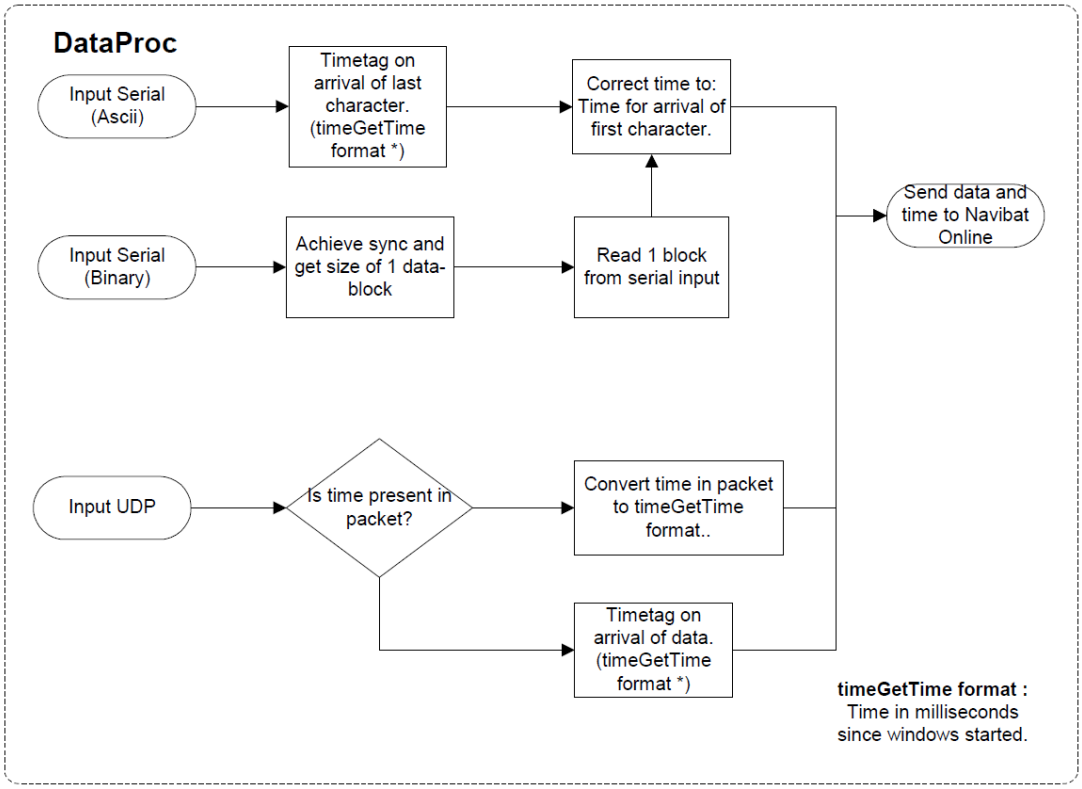
Gyro(ti) = Gyro(tk) * (ti-tk+1)/(tk-tk+1) + Gyro(tk+1) * (ti-tk)/(tk+1-tk)

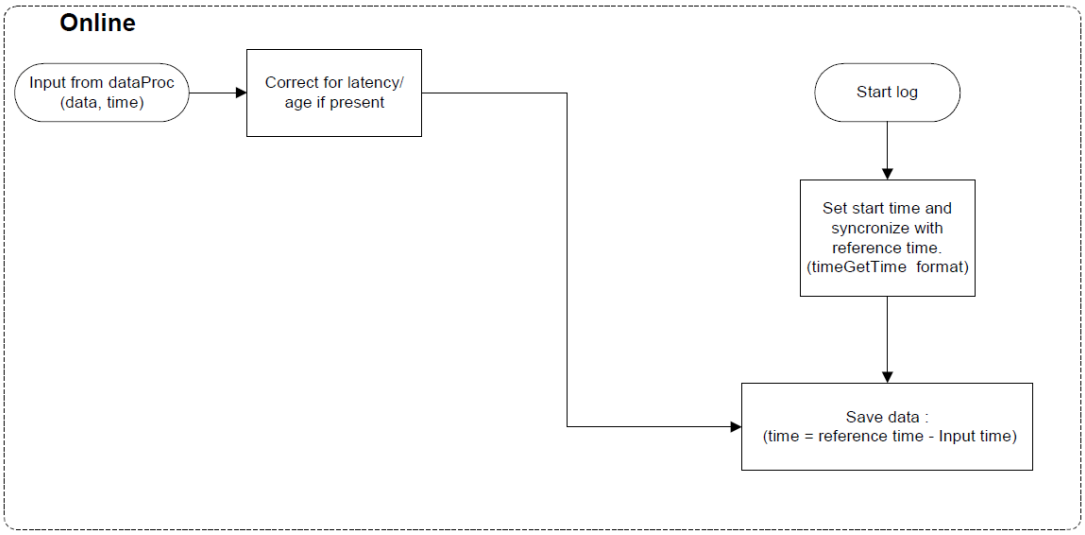
To make sure that the interpolated values are trustworthy it is required that all packages from the various units are tagged on the PC immediately upon receipt of the 1st. byte.

4.1.1 TimeBox absolute time tagging

If the most accurate time tagging is required then NaviScan should be combined with the EIVA TimeBox (See http://download.eiva.dk/online-training/Various%20Manuals/TimeBox%20description%201.pdf)

With this in use the sensor data can be time tagged with a accuracy better than 0.1 ms.


DataProc
Input Serial (Ascii)
Timetag on arrival of last character. (timeGetTime format *)
Correct time to: Time for arrival of first character.
Send data and time to Navibat Online
Input Serial (Binary)
Achieve sync and get size of 1 data-block
Read 1 block from serial input
Input UDP
Is time present in packet?
Convert time in packet to timeGetTime format..
Timetag on arrival of data. (timeGetTime format *)
timeGetTime format : Time in milliseconds since windows started.


Online
Input from dataProc (data, time)
Correct for latency/ age if present
Start log
Set start time and syncronize with reference time. (timeGetTime format)
Save data : (time = reference time - Input time)