Teledyne Odom/ Teledyne RDI TDY Data Format Definition

Contents

1. Introduction

The MB1 sonar was released in Jan, 2012 by TRDI/TOH. The initial release employed TDY format Version 0 as a QC and processing format. This and future versions of the TDY format contain descriptions for all data streams offered by the sonar. The sonar operator will select streams to be logged to media storage or passed, via UDP, to runtime logging and processing software. In both cases the data format is identical.

2. Data Format and Conventions

There are two main data types output in TDY streams (media storage or UDP) **TDYMB01** and **TDYRTA1**. TDYMB01 are sonar data and the TDYRTA1 are the sensor related data (GPS, SVP, HPR, etc).

The sonar data (TDYMB01) is split further into several separate data structures which related to different features of the sonar, like water column, snippets, side-scan, side-scan snippets, raw IQ data, bathymetry data, corrected bathymetry, etc. Each TDYMB01 packet starts with the packet header identified by string *TDYMB01*, which is always present in all TDY packets. Other data structured are **optional**. Which data type is included in the packet is indicated by the *offset* filed in the packet header If the *offset* is equal *0* it means the data structure is not included in the packet. For example the bathymetry data structure (AR) can be sent in a different packet that the snippet data, etc. The header structure should be used to consolidate the data in different packets relating to the same ping by either a *ping_number* filed in the packet header. This sonar operates differently from most beam forming sonars as it can, and in most cases will, produce a different number of range and angle pairs for each ping. Additionally, the individual beam launch angles can vary from ping to ping as well as the swath aperture.

The non-sonar data (TDYRTA1) packets contain position, attitude, heading, and sound velocity at the sonar face which are all time stamped in a Real Time Appliance (RTA) and are included in the TDY stream without any modification. Each data type is captured in its original format and is prepended with a timestamp. In the case of the GPS GGA string the original timestamp from the GPS receiver that is embedded in the GGA string must be used. For all other data types, attitude, heading and SV, the prepended time stamp must be used.

2.1. Data Types and Byte Ordering

Data types for parameters and values will be shown using the ANSI C99 specification as follows:

Type	***# bytes***	***Description***
float64	8	64 bit floating point value
float32	4	32 bit floating point value
uint32	4	32 Bit Unsigned Integer
uint16	2	16 Bit Unsigned Integer
uint8	1	8 Bit Unsigned Integer
int32	4	32 Bit Signed Integer
int16	2	16 Bit Signed Integer
int8	1	8 Bit Signed Integer
char	1	ASCII character

Computer architectures use two different methods for handling memory storage called big endian and little endian. They refer to the order in which the bytes are stored in memory. The MB1/RTA/TSPU sends and receives packets in the **little endian** format. It is the responsibility of the client applications to insure that the packets are encoded and decoded using the little endian byte order.

3. Data Format Definition

3.1. TDY File Header

When the TDY is saved on the media storage the special file header is placed in the beginning of each TDY file. For UDP communication the header is sent only once for each opening of the network port (e.g. at startup).

Offset	#Elements	Type	Value	Description
0	48	char	Teledyne Hydrographic direct logging version 3.0	TDY File header
This timestamp indicates the file creation time. It is not in any way associated with the acquisition time of data and should not be used to convert or process the data. It is included here to assist with book keeping and version control				
48	1	uint16	YYYY	Year
50	1	uint8	MM	Month
51	1	uint8	DD	Day
52	1	uint8	HH	hour
53	1	uint8	mm	minute
54	1	uint8	SS	second
55	1	uint16	sss	milliseconds
57	4	char	space	reserved- always 0

3.2. TDYMB01 Data Format Definition

Offset	*#Elements*	*Type*	*Value*	*Description*
0	7	char	TDYMB01	**Packet Header, Identifier – always constant value of TDYMB01**
7	1	uint32	Packet Size in Bytes	**Byte offset from start of current TDYMB01 packet to first byte following the TDY_MB01END string. This field will always be populated with a nonzero value equal to the total byes in the TDY_MB01 packet. Every valid TDY_MB01 packet will be terminated by a TDY_MB01END string.**
11	1	uint16	3	**version – Version number of this format description**
13	1	uint8	1…n	**Sonar Head**
14	8	char	MB1	**Model Number for this head**
22	1	uint16	xxxx	**Serial Number for this head**
24	1	uint16	xxxx	**Firmware version for this DSP**
26	1	uint16	xxxx	**Firmware version for this COMM**
28	1	uint16	xxxx	**Firmware version for this RTA**
30	1	uint16	xxxx	**Software version for this TSPU**
32	1	uint32	ping_number	**Ping Number for this ping**
36	1	uint8	data_structures+1	**Number of data structures defined for this TDY_MB01 packet. Data structure type and order are defined in note 1. The final offset is for the TDY_END token. There may be less than the 12 defined offsets. All non-zero offsets will be shown.**
37	1	uint32	TIME_Segment	**Byte offset from start of current TDYMB01 packet to TDY_TIME structure. Value of 0 indicates that there is no TDY_TIME data in current packet**
41	1	uint32	RAW_Segment	**Byte offset from start of current TDYMB01 packet to TDY_RAW structure. Value of 0 indicates that there is no TDY_RAW data in current packet**

Offset	#Elements	Type	Value	Description
45	1	uint32	IQ_Segment	**Byte offset from start of current TDYMB01 packet to TDY_IQ structure. Value of 0 indicates that there is no TDY_IQ data in current packet**
49	1	uint32	PROC_Segment	**Byte offset from start of current TDYMB01 packet to TDY_PROC structure. Value of 0 indicates that there is no TDY_PROC data in current packet**
53	1	uint32	AR_Segment	**Byte offset from start of current TDYMB01 packet to TDY_AR structure. Value of 0 indicates that there is no TDY_AR data in current packet**
57	1	uint32	Qual_Segment	**Byte offset from start of current TDYMB01 packet to TDY_QUAL structure. Value of 0 indicates that there is no TDY_QUAL data in current packet**
61	1	uint32	Snippet_Segment	**Byte offset from start of current TDYMB01 packet to TDY_SNIP structure. Value of 0 indicates that there is no TDY_SNIP data in current packet**
65	1	uint32	SnipSS_Segment	**Byte offset from start of current TDYMB01 packet to TDY_SNIPSS structure. Value of 0 indicates that there is no TDY_SNIPSS data in current packet**
69	1	uint32	SS_Segment	**Byte offset from start of current TDYMB01 packet to TDY_SS structure. Value of 0 indicates that there is no TDY_SS data in current packet**
73	1	uint32	WC_Segment	**Byte offset from start of current TDYMB01 packet to TDY_WC structure. Value of 0 indicates that there is no TDY_WC data in current packet**
77	1	uint32	RCAR_Segment	**Byte offset from start of current TDYMB01 packet to roll corrected Angles Range TDY_RCAR structure. Value of 0 indicates that there is no TDY_RCAR data in current packet**

Offset	*#Elements*	*Type*	*Value*	*Description*
81	1	uint32	OFFSET_Segment	**Physical Offsets stored in MB1 memory. Not Yet implemented**
85	1	uint32	Future1_Segment	**Byte offset from start of current TDYMB01 packet to a packet to be defined later. Packet can be defined by any user. Convertors unfamiliar with the packet will skip it.**
89	1	uint32	FutureN_Segment	**Byte offset from start of current TDYMB01 packet to a packet to be defined later. Packet can be defined by any user. Convertors unfamiliar with the packet will skip it.** **Multiple new packets may be defined**
Time Stamp section – This is the only time stamp for this ping. Time is for the middle of the transmission (first sonar data sample) and is readable as YYYYMMDD HH:MM:SS.ssss. Each element of the time stamp has its own field. Byte offsets are given from the start of this data structure.				
0	8	char	TDY_TIME	**Identifier—TDY_TIME**
8	1	uint16	YYYY	**Year**
10	1	uint8	MM	**Month**
11	1	uint8	DD	**Day**
12	1	uint8	HH	**Hour**
13	1	uint8	MM	**Minute**
14	1	uint8	SS	**Second**
15	1	uint64	sssss	**1/100th of milliseconds (10μs)**
16	1	uint8	Time_origin	**Time origin flag** **0 – RTA time** **1 – PC time**
Raw data section – Elements in this section relate to sonar settings and cannot be changed in processing. Byte offsets are given from the start of this data structure.				
0	7	char	TDY_RAW	**Identifier—TDY_RAW** **See Note 2.**
7	1	Float32	sample_rate	**Sample rate, Hz**
11	1	uint32	samples	**Backscatter samples per channel. True number of samples acquired in each ping**
15	1	uint32	range	**ping_range– Ping range in samples**

Offset	#Elements	Type	Value	Description
				Range as set by the user. May differ from samples
19	1	float 32	transmit_power	**ping_power – Ping transmit power in dB**
21	1	uint16	pulse_width	**ping_pulse_width- Ping pulse width in microseconds**
23	1	uint16	max_depth	**Max-depth Gate (samples) 0 if off**
25	1	uint16	min_depth	**Min-depth Gate (samples) 0 if off**
27	1	uint16	max_range	**Max-range Gate (samples) 0 if off**
29	1	uint16	min_range	**Min-range Gate (samples) 0 if off**
31	1	uint16	max_ping_rate	**Upper ping rate set by user [Hz]**
33	1	float32	start_frequency	**Start ping frequency in [Hz]**
37	1	float32	stop_frequency	**Stop ping frequency in [Hz]**
41	1	uint16	modulation_type	**Modulation type:** **0 – CW** **1 – LFM (linear FM)** **2 – HFM (Hyperbolic FM)**
IQ data section- down converted raw complex channel data. This data will not be generally required for processing				
0	6	char	TDY_IQ	**Start of IQ data (optional). IQ data will not typically be exported in the TDY format.**
6	1	uint16	n_channel	**Number of Channels. 24 in MB1**
8	1	uint32	n_samples	**Number of IQ pairs per channel (k)**
12	channel*samples*2	int32	IQ Data	**IQ Pairs, first channel first (n channels)** **(I1,Q1,(I2,Q2)…(In,Qn) – Sample 1 (I1,Q1,(I2,Q2)…(In,Qn) – Sample K** **Kx[i24x2xN] (k samples)**
Processed data section- Elements in this section are processed data products. Byte offsets are given from the start of this data structure.				
0	8	char	TDY_PROC	**Identifier—TDY_PROC** **Start of processed data**
8	1	float32	gain	**Gain – linear gain applied to display in dB**
12	1	float32	spreading	**Spreading – Spherical spreading**

Offset	*#Elements*	*Type*	*Value*	*Description*
16	1	float32	absorption	**Absorption [dB/km]**
20	1	uint8	stacking	**Number of pings stacked. Value of 1 indicates no stacking**
21	1	uint8	Stacking method	**Method used to stack data. Currently only one method is available. 1 in this field indicates that both the amplitude and phase are averaged after beamforming but before bottom detection.**
22	1	Float32	Head tilt angle	**The installation angle [deg]. Head tilt is counted clockwise direction. Sonar negative head tilt means the sonar broadside beam is pointing toward the PORT side, i.e. the broad side beam points to 30deg.**
Angle and Range Section- Angles and Ranges are only reported for valid soundings. The number of valid soundings may vary from ping to ping even if the requested number of beams remains constant. Byte offsets are given from the start of this data structure.				
0	6	char	TDY_AR	**Identifier—TDY_AR** **Start of sounding data. Angle and Range pairs**
6	1	Float32	SV	**Sound Velocity used to beam form this profile**
10	1	uint16	n_soundings	**Number of detections (angles and ranges)**
12	n_soundings	float32	R_beam_angle_sounding	**n angles in radians with 0 at sonar bore sight. These are across track angles.** **Beams are counted counterclockwise direction. The first beam is on the PORT side and has negative angle. The angles and beam count increases when going to STARBOARD side.**
	n_soundings	float32	P_beam_angle_sounding	**n angles in radians with 0 at sonar bore sight. These are along track angles. All will be zero unless pitch steering is enabled**
	n_soundings	float32	range_sounding	**n ranges in units of samples.** **Sample of 0 is no depth (not valid sounding).** **Sample of x means range of r = (x-1) ***

Offset	*#Elements*	*Type*	*Value*	*Description*
				SoS/(2*Fs) in meters.
Uncertainty values, measured in samples, for all valid soundings. Byte offsets are given from the start of this data structure.				
0	8	char	TDY_QUAL	**Identifier—TDY_QUAL** **Start of quality data**
8	n_soundings	float32	range_uncertainty	**n quality values give range uncertainty in samples**
Snippet backscatter data. One time series for each valid sounding.				
0	8	char	TDY_SNIP	**Identifier—TDY_SNIP** **Start of snippet data (optional)** **There will only be a snippet if there is a valid sounding on a beam. So, the number of snippets will always be n_soundings**
8	1	uint8	Snippet_gain_flag	**1= digital TVG Applied to WC data** **0= digital TVG not applied**
9	1	uint16	n_snippet	**Number of snippets.** **NOTE: There will be a one-to-one correspondence between soundings and snippets. There is no snippet unless there is a valid sounding. Snippet angles are redundant but can be used to clear up any ambiguity.**
11	n_snippet	float32	snippet_angle	**Radians.**
15	n_snippet	uint32	sample_one_snippet	**sample number of the first sample of the snippet for snippets 1 – n_snippets.**
15+n*4	n_snippet	uint32	n_samples_snippet	**n sample counts (number of samples) in each snippet**
	sum(n_samples_snippet)	float32	backscatter_snippet	**snippets for beams 1 – n_snippetss. Each snippet starts at sample_one and has n_samples.**
Sinppet sidescan section. Sidescan like record created from snippets				
0	10	char	TDY_SNIPSS	**Identifier—TDY_SNIPSS** **Start of side scan data constructed from snippets (optional)**
10	1	uint8	SnipSS_gain_flag	**1= digital TVG Applied to snippet data** **0= digital TVG not applied**

Offset	#Elements	Type	Value	Description
11	1	uint32	port_samples_snipss	**Number of samples in the port snippet _SS record**
15	1	uint32	stbd_samples_snipss	**Number of Samples in the Stbd snippet_SS**
19	1*port_samples_snipss + 1*stdb_samples_snipss	float32	backscatter_snippss	**Two packets of seafloor backscatter in side scan form. The first packet is port side data, the second packet is to starboard. Data starts at sample one and continues to sample port_samples_snipss or stbd_samples_snipss as required.**
Sidescan section. Sidescan record created from conventional sidescan beam forming				
0	6	char	TDY_SS	**Identifier—TDY_SS** **Start of side scan data constructed from side scan beam forming (optional)**
6	1	uint8	SideScan_gain_flag	**1= digital TVG Applied to sidescan data** **0= digital TVG not applied**
7	1	uint32	port_samples_ss	**Number of samples in the port SS record**
11	1	uint32	stbd_samples_ss	**Number of samples in the stbd SS record**
15	1*port_samples_ss + 1*stdb_samples_ss	float32	backscatter_SS	**Two packets of seafloor backscatter in side scan form. The first packet is port side data, the second packet is to starboard. Data starts at sample one and continues to sample port_samples_ss or stbd_samples_ss as required.**
Water column backscatter. Amplitude and phase data for all or a portion of the water column				
0	6	char	TDY_WC	**Identifier—TDY_WC** **Start of full water column backscatter (optional). There will be water column backscatter for all formed beams. Even if a valid sounding is not found on a beam, the will be water column backscatter for that beam.**
6		uint8	WC_gain_flag	**1= digital TVG Applied to WC data** **0= digital TVG not applied**

Offset	*#Elements*	*Type*	*Value*	*Description*
7	1	uint16	n_beams_wc	**Number of beams of water column data to follow**
9	n_beams_wc	float32	beam_angle_wc	**Launch angles of beams of water column data. Radians. 0 at sonar bore sight. N-beams_wc angles must be given.** **Beams are counted counterclockwise direction. The first beam is on the PORT side and has negative angle. The angles and beam count increases when going to STARBOARD side.**
	n_beams_wc	uint32	Sample1_WC	**First sample for each WC time series for beam n. Beach water column beam data can start at any sample**
	n_beams_wc	uint32	n_samples_WC	**Number of samples in water column time series for beam n.**
	n_beams_wc*n_samples	float32	backscatter_amp	**Water column backscatter magnitude for each beam. There will be (variable number of samples) backscatter values in each of n_beam_wc water column backscatter time series.**
	n_beams_wc* n_samples	float32	backscatter_phase	**Water column backscatter phase [rad] for each beam.**
Roll compensated bathymetry and applied corrections				
0	8	char	TDY_RCAR	**Identifier—TDY_RCAR** **This section holds roll compensated detections and applied corrections.**
7	1	uint16	Correction_method	**Correction method used for corrected detections:** **0 – basic subtraction of the Roll_correction from detection angles**
9	1	Float32	SV	**Sound Velocity used to beam form this profile**
13	1	uint16	n_soundings	**data – Ping data for this packet:** **n: Number of detections (angles and ranges)**
15	n_soundings	float32	Roll_stab_val	**Angular value (radians) per sounding used to adjust the beam angle the detection comes from to implement**

Offset	*#Elements*	*Type*	*Value*	*Description*
				gross roll stabilization
	n_soundings	float32	Roll_correction	**Angular correction (radians) applied to each detection's angle to correct for roll.**
	n_soundings	float32	R_beam_angle_sounding	**n corrected angles in radians with 0 at sonar bore sight. These are across track angles**
	n_soundings	float32	P_beam_angle_sounding	**n corrected angles in radians with 0 at sonar bore sight. These are along track angles. All will be zero unless pitch steering is enabled**
	n_soundings	float32	range_sounding	**n ranges in units of samples.** **Sample of 0 is no depth (not valid sounding).** **Sample of x means range of r = (x-1) * SoS/(2*Fs) in meters.**
	10	char	TDY_OFFSET	**Physical offsets measured from the sonar acoustic center to the given device. Units are meters or feet as indicated by units flag. Coordinate system is as follows: Z positive down (the face of the sonar points down), X is positive to the right, Y is positive forward (the receive array is forward of the transmit array)** **AC : Acoustic Center** **HPR : Heave/Pitch/Roll sensor** **Pos: Position** **COR: Center of rotation**
	1	uint8	n	**Units:** **0 = meters** **1= feet**
	1	float32	x	**X: AC to HPR**
	1	float32	y	**Y: AC to HPR**
	1	float32	z	**Z: AC to HPR**
	1	float32	x	**X: AC to POS**
	1	float32	y	**Y: AC to POS**

Offset	***#Elements***	***Type***	***Value***	***Description***
	1	float32	z	**Z: AC to POS**
	1	float32	x	**X: AT to COR**
	1	float32	y	**Y: AT to COR**
	1	float32	z	**Z: AT to COR**
	1	float32	p	**Pitch error (degrees) Positive bow up**
	1	float32	r	**Roll Error (degrees) Positive Starboard down**
	1	float32	y	**Yaw error (degrees) Positive clockwise**
Corrected Angle and Range Section- Corrected Angles and Ranges are only reported for valid soundings. The number of valid soundings may vary from ping to ping even if the requested number of beams remains constant. Byte offsets are given from the start of this data structure.				
0	7	char	TDY_Future1	**Only present if defined in offset section.**
0	TBD	char	TDY_FutureN	**Only present if defined in offset section.**
	11	char	TDYMB01_END	**Identifier—TDYMB01_END End of ping packet**

Note 2: The TDY_RAW section is required if any of the TDY_PROC fields are used.

3.3. TDYRTA1 Data Format Definition

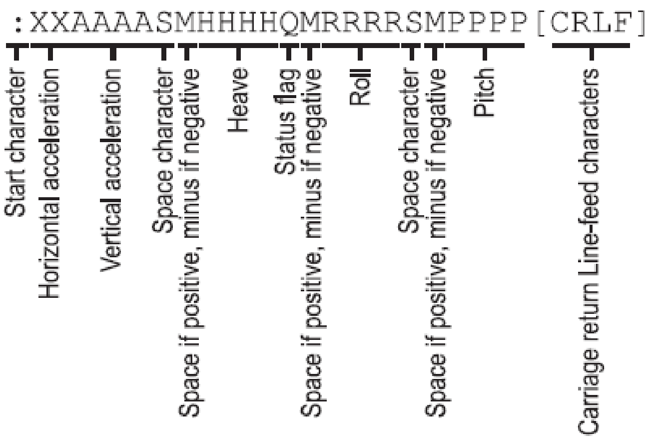
The TDYRTA1 records are sent via UDP or logged from TSPU as a separate packet for each sensor message like $GPGGA, $GPZDA, etc.

Attitude				
Offset	#Elements	Type	Value	Description
0	7	char	TDYRTA1	RTA designator
7	1	uint8	0	"0" indicates motion data
8	1	uint32	YYYY	year
10	1	uint16	MM	month
12	1	uint16	DD	day
16	1	uint16	HH	hour
18	1	uint16	MM	minute
20	1	uint16	SS	second
22	1	uint32	sssss	1/100th milliseconds (10 µs)
26	1	unit8	Time_origin	**Time origin flag** **0 – RTA time** 1 – PC time
27	1	uint32	27	TSS data length
31-58	27	unit8	:013D11 0000U-0051 0013<CR><LF>	TSS1 string (See Note 1)

Note 1: TSS1 data format

- The TSS1 data string contains 27 characters in five data fields.
- The acceleration fields contain ASCII-coded hexadecimal values. Horizontal acceleration uses units of 3.83cm/s² in the range zero to 9.81m/s². Vertical acceleration uses units of 0.0625cm/s² in the range –20.48 to +20.48m/s².
- The motion measurements contained in the data string will be in real time, valid for the instant when the Sensor begins to transmit the string.

- Motion measurements include ASCII-coded decimal values.
- Heave measurements are in cm in the range –99 to +99 meters. Positive heave is above datum.
- Roll and pitch measurements are in degrees in the range –90° to +90°. Positive roll is port-side up, starboard down. Positive pitch is bow up, stern down.

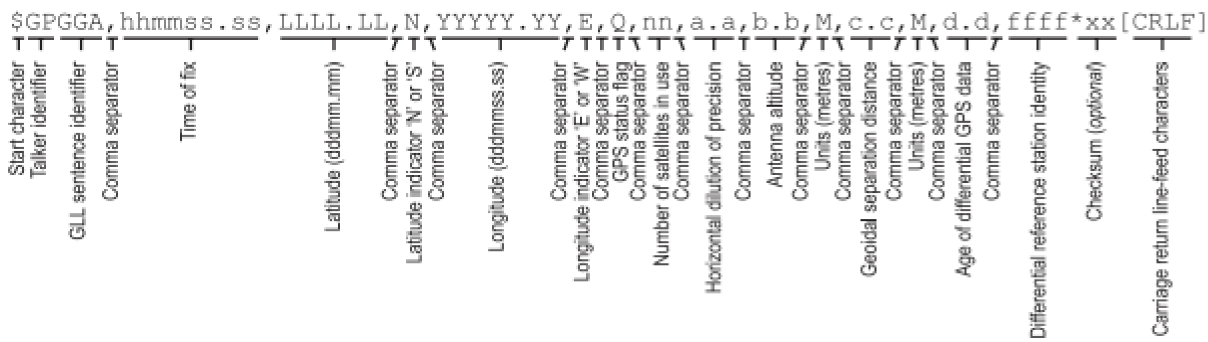


Position				
Offset	#Elements	Type	Value	Description
0	7	char	TDYRTA1	RTA designator
7	1	unit8	1	"1" indicates GGA
8	1	uint32	YYYY	year
10	1	uint16	MM	month
12	1	uint16	DD	day
16	1	uint16	HH	hour
18	1	uint16	MM	minute
20	1	uint16	SS	second
22	1	uint32	sssss	1/100th milliseconds (10 µs)

Position				
26	1	unit8	Time_origin	**Time origin flag** **0 – RTA time** 1 – PC time
27	1	uint32	80	GGA data length
31-111	80	char	$GPGGA,212318.00,3021.38185,N,09107.51921,W,2,09,,1,8.0,M,-25.8,M,6.6,0029*76<CR><LF>	GGA string (See Note 2)

Note 2

- Field lengths for Latitude and Longitude will be variable depending on the GPS receiver and correction method



Heading				
Offset	#Elements	Type	Value	Description
0	7	char	TDYRTA1	RTA designator
7	1	char	2	“2” indicates heading data
8	1	uint32	YYYY	year
10	1	uint16	MM	month
12	1	uint16	DD	day

Heading				
16	1	uint16	HH	hour
18	1	uint16	MM	minute
20	1	uint16	SS	second
22	1	uint32	sssss	1/100th milliseconds (10 µs)
26	1	unit8	Time_origin	**Time origin flag** **0 – RTA time** 1 – PC time
27	2	uint32	19	Heading data length
31-50	19	char	$GPHDT,215.33T,15<CR><LF>	HDT string

Sound Velocity				
Offset	#Elements	Type	Value	Description
0	7	char	TDYRTA1	RTA designator
7	1	char	4	"4" indicates velocity data
8	1	uint32	YYYY	year
10	1	uint16	MM	month
12	1	uint16	DD	day
16	1	uint16	HH	hour
18	1	uint16	MM	minute
20	1	uint16	SS	second
22	1	uint32	sssss	1/100th milliseconds (10 µs)

26	1	unit8	Time_origin	**Time origin flag** **0 – RTA time** 1 – PC time
27	1	uint32	9	Velocity data length
31-40	9	char	1457.48<CR><LF>	Velocity string, m/s

4. Dual head vs. single head

When two processors are used in dual head configuration, one for Master (1st head) and one for Slave (2nd) head, each processor generates its own TDY stream (UDP and files). The information about which processor sends which stream is included in the TDY header.

In the dual head configuration the TDY data from Slave processor contains exactly the same TDYRTA1 data as the Master processor. The head number and the head tilt for each head is included in the TDY for each stream.

The UDP streams from both processors are sent to the 3rd party software in asynchronous way, i.e. there is no assumption that the Master head ping will come before the Slave head ping data. This comes from the fact that the heads can ping simultaneously, interleaved or in a free run mode, so the 3rd party shall assume the TDY streams from Master and Slave processor are not synchronized to each other.

The logging of the TDY file is done is the same way as UDP streaming. Each processor generates its own TDY file.