Ethernet data protocol ScaLa 1403

Version History

Date	Version	Changes
24.04.2012	1.0	Initial Draft,
25.04.2012	1.1	Review / Corrections,
27.04.2012	1.2	Review / Corrections
25.05.2012	1.3	Add general Ethernet information Add scan data type 0x2202
31.08.2012	1.4	Review / Corrections
06.02.2013	1.5	Add object data type 0x2280. ScaLa object list sent by Ibeo ECU
29.07.2013	1.8	Add classes for under/over drivable to object classification

06.09.2013	1.9	• Added ECU scan data type 2205,
		• correction within classification flags, beam tilt
		• Added Information concerning:
		  • time stamping in B1-ECU systems,
		  • object flags for validation

11.10.2013	2.0	RaS: Add description for scan point for 0x2205
11.10.2013	2.1	RaS: Minor fixes
28.10.2013	2.2	Add device status data type 0x6301
07.11.2013	2.3	Update in 0x2280: Standard deviation values for object's orientation, for reference point and for velocity vector.
14.11.2013	2.4	Add descriptions for Scala-B2 sample, prepare for new scan data type 0x2208
09.12.2013	2.5	Added data type 0x2225 (LUX compatible object list format) to document the Object data output of B1-ECU systems build in downward compatible format
16.01.2014	3.0	Restructuring and completion of data type descriptions
03.02.2014	3.1	Clarify the definition of orientation of an object box.
13.06.2014	3.2	Added Object data type 0x2271 and some minor corrections
30.06.2014	3.3	Correct description of dynamic property and "HiddenStatusAge" of tracked properties. Correct description of property

03.07.2014	3.4	Correction in Object data type 0x2271 definition. The unit of orientation and f tracked and untracked object box is 1/100 deg
31.07.2014	3.5	Chapter 2.2 System Setup 2 – ScaLa B2: Fix data table listing the datatypes dependent on firmware version
14.10.2014	3.6	Add missing 4 bytes in object definition (0x2271). Page 22: Object definition offset 118+ Add missing description of TrackingPointLocation in object definition (0x2271)

Table of Content

1 Introduction

This document describes how data is transmitted by ScaLa sensors and ScaLa Sensor Systems via the Ethernet Interface.
The data protocol also describes the format of an *.idc file, because the same data types are directly stored within these files. An *.idc file is a proprietary Ibeo data container format for Ibeo/Valeo Laserscanners.

It covers all supported system setups using ScaLa-B1 and ScaLa-B2 sensors including data flows.

The document is structured as follows:

Section 2 describes possible system setups of ScaLa B1 and B2 samples. This includes systems consisting of a Scala samples of B1 status, which are always accompanied with an external ECU providing object tracking algorithms, B2 standalone systems with embedded object tracking and B2 samples with an (optional) external ECU, which can be provided in request for special needs.

As ScaLa systems are in sample status the data definition/protocol provided by these sensors and sensor systems have changed ad my change during development. The Scala data protocol over Etherent is defined in layered architecture such that changes in interfaces only relate to redefining a parser on the “datatype” level, not reconstruction the complete interface handling.

Scala and Scala systems provide data in form of “data types” on a TCP/IP stream. Datatypes related to ScaLa B1 and B2 systems are described in detail within **section** 3 **(Data Type Definition) and section** 4 **(Subtypes**).

Section 5 describes handling of the command interface of the external ECU to configure the data output of the device.

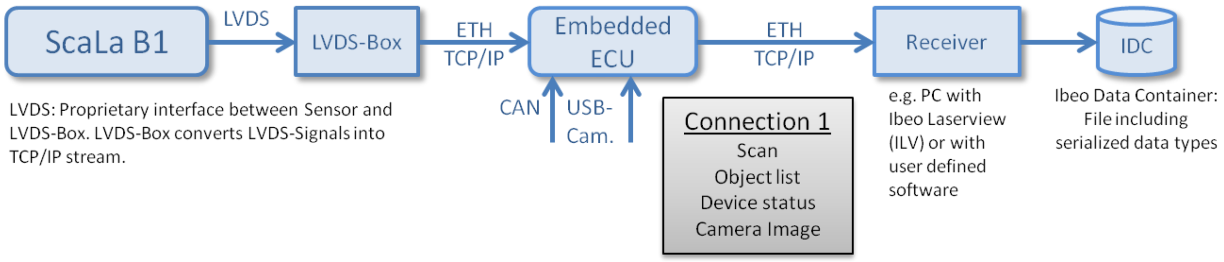
Section 6 provides background information with respect to the time stamping methodology implemented within Scala systems provided with external ECU.

Section 5 and 6 only refers to systems equipped with external ECUs.

2 System Setups

2.1 System Setup 1 – ScaLa B1 via ECU

The following diagrams show supported system setups with ScaLa-Sensors.



TCP/IP configuration

The receiver can connect to the embedded ECU using this configuration.

Default IP address: 192.168.0.100
Subnet mask: 255.255.0.0
Port Number: 12002

Note: In order to receive data from the embedded ECU, it is required to send the SetFilter-command to the ECU after a TCP/IP connection has been established. Please see chapter 5 ”Commands and replies”

The IP address can be changed using the ECU's configuration page.

Data type overview

The following table lists the data type being transferred through connection 1. The data types are described in detail in chapter 3 Data Type Definition.

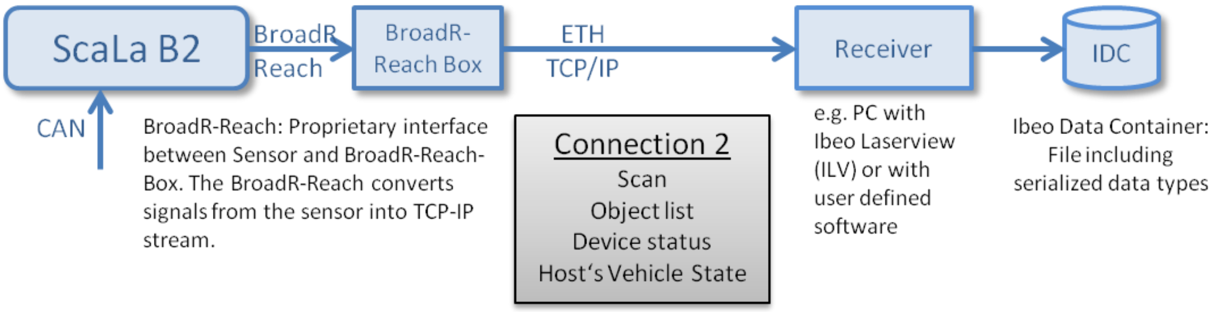
Data type is transmitted with a frequency of 25Hz.

		Provided datatypes by ECU/AppBase Version			
		Appbase ver < 5.3.x	**Appbase ver ≥ 5.3.x**		
Scan		0x2205	0x2205		
Object list		0x2280 or 0x2225 *(*)*	0x2280 or 0x2255 *(*)*		
Device status		n.a.	0x6301		
Camera Image		0x2403	0x2403		
Host Vehicle State		0x2806	0x2807		

(*) The object data type output depends on the ECU appbase package configuration. 0x2280 is the Scala default object list data type. On request a package configuration

supporting output of object data in 0x2225 is available (maintaining data type downward compatibility with existing ibeo LUX systems).

2.2 System Setup 2 – ScaLa B2



TCP/IP configuration

ScaLa 1403 B2 sensors with use the TCP/IP protocol over Ethernet with default configuration:

Default IP address: 192.168.1.52
Subnet mask: 255.255.0.0
Port Number: 12004

Data type overview

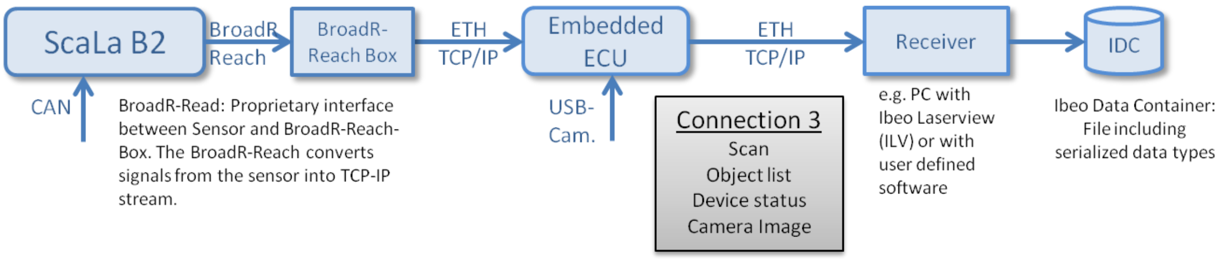
The following table lists the data type being transferred through connection 2. The data types are described in detail in chapter 3 Data Type Definition.

Data type is transmitted with a frequency of 25Hz.

		Provided datatypes by Scala B2 FW Version			
		Scala FW ver < X026	**Scala FW ver ≥ X026**	**Scala FW ver ≥ X027**	**Scala FW ver ≥ X031**
Scan		0x2202	0x2202	0x2208	0x2208
Object list		0x2270 or 0x2225 *(*)*	0x2270	0x2270	0x2271
Device status		0x6301	0x6301	0x6301	0x6301
Camera Image		n.a.	n.a.	n.a.	n.a.
Host Vehicle State		0x2805	0x2805	0x2805	0x2805

(*) The object data type output depends on the ECU-appbase package configuration. 0x2280 is the Scala default object list data type. On request a package configuration supporting output of object data in 0x2225 is available (maintaining data type downward compatibility with existing ibeo LUX systems).

2.3 System Setup 3 – ScaLa B2 via ECU



TCP/IP configuration

The receiver can connect to the embedded ECU using this configuration.

Default IP address:	192.168.0.100
Subnet mask:	255.255.0.0
Port Number:	12002

Note: In order to receive data from the embedded ECU, it is required to send the SetFilter-command to the ECU after a TCP/IP connection has been established. Please see chapter 5 "Commands and replies"

The IP address can be changed using the ECU's configuration page.

Data type overview

The following table lists the data type being transferred through connection 3. The data types are described in detail in chapter 3.

Data type is transmitted with a frequency of 25Hz.

		Provided datatypes by Scala B2 FW Version			
		Appbase ver < 5.3.x	**Appbase ver ≥ 5.3.x**		
Scan		0x2205	0x2205		
Object list		0x2280	0x2280		
Device status		0x6301	0x6301		
Camera Image		0x2403	0x2403		
Host Vehicle State		0x2806	0x2807		

3 Data Type Definition

In this chapter the data types are described in detail. Here you will find the data types which are mentioned in the data type overview tables for each system setup. A few data type description refer to special sub types. These are described within chapter 4 (Subtypes).

3.1 Ibeo Data Header

Each message always starts with an Ibeo data header. You can resynchronize with the data stream by searching for the Magic Word of this Ibeo data header.

The Ibeo data header is encoded in **network byte order / big endian format**.

Offset	Bytes	Ibeo data header	Data type	Description
0	4	Magic Word (0xAFFEC0C2)	UINT32	The “magic word” is used for searching Ibeo messages and to distinguish between different versions.
4	4	Size of previous messages	UINT32	Helps to navigate backwards through a file. Unused in live data.
8	4	Size of this message	UINT32	Helps to read the message data. Size of message content without this header.
12	1	Reserved	UINT8	-
13	1	DeviceID	UINT8	ID of the connected device. Unused in data received directly from the ScaLa sensors.
14	2	Data Type ID	UINT16	Specifies the data type within this message.
16	8	NTP time	NTP64	Time when this message was created.
24		Message data	-	Depending on data type.

The type of the message after this header is defined by the Data Type ID.

3.2 Data Type 0x2202: Scan Data

Each scan data block starts with a header followed by the scan point list.
The data is encoded in **little endian format**!

For angle information the unit angle ticks is used. A ScaLa uses 11520 ticks per rotation (see also Angle ticks per rotation below). Thus the angular resolution is 1/32°. This value is needed to convert angle ticks:

$$\text{angle} = 2\pi \frac{\text{angle ticks}}{\text{angle ticks per rotation}}$$

Angles are given in the ISO 8855 / DIN 70000 scanner coordinate system.

Coding in Little Endian Byte Order

Offset	Bytes	Scan header:	Data type	Description
0	2	Scan number	UINT16	The number of this scan. The number will be increased from scan to scan.
2	2	Scanner status	bit field 16 bits	Reserved (*) not available in ScaLa yet
4	2	Sync phase offset	UINT16	Phase difference between sync signal and scanner mirror crossing the synchronization angle. (not available in ScaLa yet)
6	8	Scan start time NTP	NTP64	time when the first measurement was done. (NTP format)
14	8	Scan end time NTP	NTP64	ime when the last measurement was done (NTP format)
22	2	Angle ticks per rotation	UINT16	Number of angle ticks per rotation.
24	2	Start angle	INT16	Start/end angle in angle ticks of this scan.
26	2	End angle	INT16	
28	2	Scan points	UINT16	Number of scan point transmitted in this scan.
30	2	Mounting yaw angle	INT16	Rotation of the scanner around the axes of the reference coordinate system. All angles are given in angle ticks. Order of translation and rotation is essential: Yaw->Pitch->Roll->Translation. Scan data is given in the scanner coordinate system without any transformation.
32	2	Mounting pitch angle	INT16	Mounting position of the scanner relative to the reference coordinate system (ISO 8855 / DIN 70000 coordinate system). The origin is located on flat ground under the center of the rear axle. X-axis faces to the vehicle front resp. straight driving direction. Y-axis faces left. The mounting position is needed for ego motion compensation (only available if scanner x-y-plane is almost
34	2	Mounting roll angle	INT16	
36	2	Mounting position x	INT16	
38	2	Mounting position y	INT16	
40	2	Mounting position z	INT16	

				parallel to the ground). All coordinates are given in centimeters. Order of translation and rotation is essential (Rotation -> Translation).
42	2	Flags	UINT16	reserved
44		Scan Point List	Scan Point	Array of scan points. See number of scan points above and point information below.

Offset	Bytes	Scan point:	Data type	Description
0	1	Layer	UINT4	Scan layer of this point (zero-based). Use the low nibble / bits 0…3 of this byte.
		Echo	UINT4	Echo number of this point (zero-based). Use the high nibble / bits 4…7 of this byte.
1	1	Flags	Bit field 8 bits	0x01: transparent point 0x02: clutter (atmospheric) 0x04: ground 0x08: dirt 0xF0: reserved
2	2	Horizontal angle	INT16	Angle of this point in angle ticks in the scanner coordinate system
4	2	Radial distance	UINT16	Distance of this point in the scanner coordinate system in cm
6	2	Echo pulse width	UINT16	Detected width of this echo pulse in cm
8	2	Reserved	UINT16	-
10				

3.3 Data Type 0x2205: ECU Scan Data

Each scan data block starts with a header followed by the scanner info list and the scan point list. Each scan point has a device ID that refers to a sensor in the sensor info list. By means of the sensor info list this data type is able to hold also a fused scan from a multi sensor fusion system consisting of multiple scanners.

The data is encoded in network byte order / big endian format.

Offset	Bytes	Scan header	Data type	Description
0	8	Scan start time	NTP64	NTP time when the first measurement was done.
8	4	Scan end time offset	UINT32	Time difference between last and first measurement in us.
12	4	Flags	Bit field: 32 bits	Bits 0…8: reserved Bit 9: fused scan

				Bit 10: mirror side (0 = front, 1 = rear) Bit 11: coordinate system (0 = scanner coordinates, 1 = vehicle coordinates)
16	2	Scan number	UINT16	The number of this scan. The number will be increased from scan to scan. Overflow occurs after 2^{16} scans.
18	2	Scan points	UINT16	Number of scan points transmitted in this scan.
20	1	Number of scanner infos	UINT8	Number of scanner infos transmitted in this scan.
21	3	Reserved	3 bytes	-
24	number of scanner infos * 148	Scanner info list	Scanner info	Array of scanner infos. See number of scanner infos above and scanner info below.
24 + number of scanner infos * 148	number of scan points * 28	Scan point List	Scan point	Array of scan points. See number of scan points above and point information below.

This is the scanner info from the **scanner info** list:

Offset	Bytes	Scanner info	Data type	Description
0	1	Device ID	UINT8	Device ID of this scanner.
1	1	Scanner type	UINT8	0x60 = ScaLa B1
2	2	Scan number	UINT16	The scan number coming from the scanner device. The number will be increased from scan to scan. Overflow occurs after 2^{16} scans.
4	4	Reserved	4 bytes	-
8	4	Start angle	FLOAT32	Field of view of this scanner given in its local coordinate system. In radians normalized to $[-\pi, +\pi]$.
12	4	End angle	FLOAT32	
16	8	Scan start time	NTP64	NTP time (based on computer time on which the Ibeo software runs) when the first measurement of this scanner was done.
24	8	Scan end time	NTP64	NTP time (based on computer time on which the Ibeo software runs) when the last measurement of this scanner was done.

32	8	Scan start time from device	NTP64	NTP time (as received from the sensor) when the first measurement of this scanner was done.
40	8	Scan end time from device	NTP64	NTP time (as received from the sensor) when the first measurement of this scanner was done.
48	4	Scan frequency	FLOAT32	Scan frequency of this scanner in Hz.
52	4	Beam tilt	FLOAT32	Vertical Angle between the measurements on mirror side front/rear. This value is valid for measuring in x-direction resp. 0° in the scanner coordinate system. In radians normalized to [−π, +π]. Beam is pitched downwards if values are positive and vice versa. **(*Not set for ScaLa B1)**
56	4	Scan flags	Bit field: 32 bits	reserved
60	24	Mounting Position	Mounting-PositionF	Mounting Position in [m] and [rad]. (see Subtype definition)
84	8	Resolution 1	Resolution Info	Scan resolution for different sectors of the scanner field of view. Resolutions can be the same for all sectors (constant angular resolution) or different (e.g. focused angular resolution). Please see resolution info description below.
92	8	Resolution 2	Resolution Info	
100	8	Resolution 3	Resolution Info	
108	8	Resolution 4	Resolution Info	
116	8	Resolution 5	Resolution Info	
124	8	Resolution 6	Resolution Info	
132	8	Resolution 7	Resolution Info	
140	8	Resolution 8	Resolution Info	
148				

Following block describes the data type "**Resolution Info**".

Offset	Bytes	Resolution Info:	Data type	Description
0	4	Resolution start angle	FLOAT32	Starting from this angle the given resolution is valid until the next resolution start angle or the scan end. In radians normalized to [−π, +π]. Valid only if resolution value is > 0.
4	4	Resolution	FLOAT32	Resolution for this sector. In radians normalized to [−π, +π]. Valid only if > 0.
8				

This is a **scan point** from the scan point list:

Offset	Bytes	Scan point:	Data type	Description
0	4	X position	FLOAT32	X position of this scan point in m.
4	4	Y position	FLOAT32	Y position of this scan point in m.
8	4	Z position	FLOAT32	Z position of this scan point in m.
12	4	Echo width	FLOAT32	Echo width of this scan point in m.
16	1	Device ID	UINT8	ID of the device measuring this point.
17	1	Layer	UINT8	Scan layer of this point (zero-based).
18	1	Echo	UINT8	Echo number of this point (zero-based).
19	1	Reserved	1 byte	-
20	4	Timestamp (µs)	UINT32	Time offset in µs when this scan point was measured based on the scan start time.
24	2	Flags	Bit field: 16 bits	0x0001: ground 0x0002: dirt 0x0004: rain/snow/spray/fog/… 0x1000: transparent Else : reserved
26	2	Reserved	2 bytes	-
28				

3.4 Data Type 0x2208: Scan Data

This data type can be received from Scala-B2 sensor with firmware version ≥ X027.

Coding in Big Endian Byte Order

For angle information the unit angle ticks is used. A ScaLa uses 11520 ticks per rotation (see also Angle ticks per rotation below). Thus the angular resolution is 1/32°. This value is needed to convert angle ticks:

$$\text{angle} = 2\pi \frac{\text{angle ticks}}{\text{angle ticks per rotation}}$$

Angles are given in the ISO 8855 / DIN 70000 scanner coordinate system.

Offset	Bytes	Scan header:	Data type	Description
0	2	Scan number	UINT16	The number of this scan. The number will be increased from scan to scan. Overflow occurs after 2^{16} scans.
2	2	Scanner type	UINT16	Scala-B2: 0x0062
4	2	Scanner status	bit field 16 bits	0x0001: motor on 0x0002: laser on 0x0004: reserved 0x0008: frequency locked 0x0010: reserved 0x0020: reserved 0x0040: reserved 0x0080: reserved 0x0100: motor rotating direction: 0: clockwise, 1: counter clockwise 0x0200 – 0x8000: reserved
6	2	Angle ticks per rotation	UINT16	ScaLa-B2: 11520
8	4	Scan Flags	Bit field 32 bit	Reserved
12	2	Mounting yaw angle	INT16	Rotation of the scanner around the axes of the reference coordinate system. All angles are given in angle ticks. Order of translation and rotation is essential: Yaw->Pitch->Roll->Translation. Scan data is given in the scanner coordinate system without any transformation.
14	2	Mounting pitch angle	INT16	
16	2	Mounting roll angle	INT16	
18	2	Mounting position x	INT16	Mounting position of the scanner relative to the reference coordinate system (ISO 8855 / DIN 70000 coordinate system). The origin is located on flat ground under the center of the rear axle. X-axis faces
20	2	Mounting position y	INT16	
22	2	Mounting position z	INT16	

				to the vehicle front resp. straight driving direction. Y-axis faces left. The mounting position is needed for ego motion compensation (only available if scanner x-y-plane is almost parallel to the ground). All coordinates are given in centimeters. Order of translation and rotation is essential (Rotation -> Translation). The mounting position is used for ego motion compensation, not to transform scan data but is available for further processing steps.
24	26	Reserved		Reserved
50	1	DeviceID	UINT8	Device ID of this scanner.
51	1		UINT8	Reserved
52	8	Scan start time NTP	NTP64	NTP time when the first measurement was done.
60	8	Scan end time NTP	NTP64	NTP time when the last measurement was done
68	2	Start angle	INT16	Start/end angle in angle ticks of this scan.
70	2	End angle	INT16	
72	1	Subflags	Bitfield 8 bit	Reserved
73	1	Mirror side	UINT8	For Scala: Bit 0: 0 = front mirror side, 1=rear mirror side
74	4	Reserved		
78	2	Mirror tilt	INT16	Mirror tilt of the current mirror plane relative to the rotation axis. Signed value: positive is upwards relative to sensor's coordinate system, negative values downwards. Unit 1/1000 deg
80	6	Reserved		
86	2	Number of scan points	UINT16	Number of scan point transmitted in this scan.
88		Scan Point List	Scan Point	Array of scan points. See number of scan points above and point information below.

Definition of a scan point:

Offset	Bytes	Scan point:	Data type	Description
0	0.5	Reserved	UINT4	
	0.25	Echo	UINT2	Echo number of this point (zero-based). 0..3
	0.25	Reserved	UINT2	
1	1	Layer	UINT8	Scan layer of this point (zero-based).

2	2	Scan point flags	Bit field 16bits	0x0001: transparent point 0x0002: clutter (atmospheric) 0x0004: ground 0x0008: dirt 0x0010-0x8000: reserved
4	2	Horizontal angle	INT16	Angle of this point in angle ticks in the scanner coordinate system
6	2	Radial distance	UINT16	Distance of this point in the scanner coordinate system in cm
8	2	Echo pulse width	UINT16	Detected width of this echo pulse in cm
10	1	Reserved	UINT8	-

3.5 Data Type 0x2225: ECU object data

Object data available from Ibeo API and Ibeo AppBase2 (ECU). Each data block starts with a header followed by the object list. Each object has a list of contour points.
The sigma values are calculated by Kalman filter by taking into account the object age.
All positions and angles are given in the vehicle / reference coordinate system.
Data is encoded in network byte order / big endian format.

Offset	Bytes	Object header	Data type	Description
0	8	Mid-scan timestamp	NTP64	Mid-scan timestamp is the absolute timestamp when the scanner mirror crossed the middle of the corresponding scan. Used for synchronization purpose.
8	2	Number of objects	UINT16	The number of objects transmitted in this message.
10		List of objects	Object	Array of objects.

Offset	Bytes	Object	Data type	Description
0	2	Object ID	UINT16	ID of this object from tracking.
2	2	Reserved	UINT16	-
4	4	Object age	UINT32	Number of scans this object has been tracked for.
8	8	Timestamp NTP	NTP64	Time when this object was observed. More precisely: the reference point of this object.
16	2	Object hidden status age	UINT16	Number of scans this object has only been predicted without measurement updates.
18	1	Classification	UINT8	Most likely class of this object: 0: unclassified 1: unknown small 2: unknown big 3: pedestrian 4: bike

				5: car 6: truck 7…: reserved
19	1	Classification certainty	UINT8	The higher this value is the more reliable is the assigned object class.
20	4	Classification age	UINT32	Number of scans this object has been classified as current class.
24	8	Bounding box center	Point2D	Center point of the bounding box of this object. See below for definition of Point2D.
32	8	Bounding box size	Point2D	Size of the bounding box (a rectangle parallel to vehicle coordinate system).
Following fields describe the object box. Please see 3.10 for detailed information.				
40	8	Object box center	Point2D	Center point (tracked) of this object.
48	8	Object box center sigma	Point2D	Standard deviation of the object box center point.
56	8	Object box size	Point2D	Size of the object box in the object coordinate system.
64	8	Reserved	8 bytes	-
72	4	Yaw angle	FLOAT32	Orientation or heading of the object in radians, [-PI; +PI [See 3.10. for further information
76	4	Reserved	4 bytes	-
80	8	Relative velocity	Point2D	Velocity of this object in m/s relative to the ego vehicle.
88	8	Relative velocity sigma	Point2D	Standard deviation of the relative velocity.
96	8	Absolute velocity	Point2D	Velocity of this object in m/s. Inform about the object velocity in the 'real world'.
104	8	Absolute velocity sigma	Point2D	Standard deviation of the absolute velocity.
112	18	Reserved	18 bytes	-
130	1	Number of contour points	UINT8	Number of contour points transmitted for this object.
131	1	Index of closest point	UINT8	Closes contour point of this object as index of the point list.
132		List of contour points	Point2D	Array of contour points (Point2D) in m.

Offset	Bytes	Point2D	Data type	Description
0	4	Position x	FLOAT32	X-part/coordinate of this value/point.
4	4	Position y	FLOAT32	Y-part/coordinate of this

				value/point.
8				

3.6 Data Type 0x2270: Object Data

Object data available from ScaLa B2 Laserscanners.
Each object list consists of a list header and concatenated objects.
Each object has a list of contour points. Subtypes are described below (chapter 4).

All positions and angles are given in the vehicle / reference coordinate system.

Coding in Little Endian Byte Order

In general positions, lengths, distances and sizes are coded in meters.
In general angles are coded in radians.

Object List Header:

Offset	Bytes	Object header	Data type	Description
0	8	Start Scan Timestamp	NTP64	64 bit timestamp at time of receive first measurement from the measurement core of the scan this object list is related to. Timer is started with power up of the sensor.
8	2	Object list number	UINT16	
10	2	Number of objects	UINT16	
12		Objects		

Object Data:

Offset	Bytes	Object	Data type	Description
0	2	Object ID	UINT16	A unique identifier which identifies the object. The value of the ID can be greater than the maximum number of objects tracked!
2	2	Object Age	UINT16	Number of scans for which the object has been tracked
4	2	Prediction Age	UINT16	Number of scans for which the object has only been predicted without measurement update.
6	2	Relative Moment of Measurement	UINT16	Time difference of scan start time and measurement time of reference point position [in ms]
8	1	reserved	UINT8	
9	1	Reference Point	UINT8	Location of reference point

		Location		relative to object box/bounding box 0 = Center of gravity 1 = Top/front left corner 2 = Top/front right corner 3 = Bottom/rear right corner 4 = Bottom/rear left corner 5 = Center of top/front edge 6 = Center of right edge 7 = Center of bottom/rear edge 8 = Center of left edge 9 = Box center 0xFF=Invalid Depending on tracking this is the tracked object reference point.
10	2	Reference Point Position X	INT16	X position of reference point in vehicle coordinate system, [cm]
12	2	Reference Point Position Y	INT16	Y position of reference point in vehicle coordinate system, [cm]
14	2	Reference Point Position Sigma X	UINT16	Statistical spread of the reference point position in X direction. Derived from tracking filter [cm]. Invalid value: 0xFFFF
16	2	Reference Point Position Sigma Y	UINT16	Statistical spread of the reference point position in Y direction. Derived from tracking filter [cm]. Invalid value: 0xFFFF
18	2	Reserved		
20	2	Reserved		
22	2	Reserved		
24	2	Reserved		
26	2	Reserved		
28	2	Reserved		
30	2	Reserved		
32	2	Reserved		
34	2	Reserved		
36	2	COG of all contour points, X-coordinate	INT16	
38	2	COG_Y	INT16	
Following fields describe the object box. Please see 3.10 for detailed information.				
40	2	Object Box Length	UINT16	Estimated length of the object [cm]

42	2	Object Box Width	UINT16	Estimated width of the object [cm]
44	2	Object Box Orientation Angle	INT16	Object box orientation angle in vehicle coordinate system [1/100 deg]
46	2	Reserved	UINT16	
48	2	Reserved	UINT16	
50	2	Object Box Orientation Angle Sigma	INT16	Statistical spread of object box orientation angle, derived from tracking filter [1/100deg]
Following fields describe the motion of tracked object. Absolute and relative velocities are represented by velocity vectors in vehicle's coordinate system.				
52	2	Absolute Velocity X coordinate	INT16	Estimated absolute velocity derived from tracking filter. host vehicle's motion is compensated [cm/s].
54	2	Absolute Velocity Y coordinate	INT16	
56	2	Absolute Velocity Sigma X coordinate	UINT16	statistical spread of the absolute velocity derived from tracking filter [cm/s]
58	2	Absolute Velocity Sigma Y coordinate	UINT16	
60	2	Relative Velocity X coordinate	INT16	Estimated relative velocity, which is defined by the difference between absolute and host vehicle's velocity [cm/s]
62	2	Relative Velocity Y coordinate	INT16	
64	2	Relative Velocity Sigma X coordinate	UINT16	Currently set to the same value as "Absolute Velocity Sigma"
66	2	Relative Velocity Sigma Y coordinate	UINT16	
68	1	Classification	UINT8	Most likely class of this object: 0: unclassified 1: unknown small 2: unknown big 3: pedestrian 4: bike 5: car 6: truck 7..9: reserved 10: over drivable (n.a. yet) 11: under drivable
69	1	Object Flags	UINT8	Bit 0: Tracking model 0 = dynamic model, 1 = static model Bit 1: 1 = mobile detected Bit 2: 1 = track valid For a detailed meaning of these flags and combinations please see section 0

				Definition of object flags
70	2	Classification age	UINT16	Number of same class assignments in a row.
72	2	Classification Confidence	UINT16	Statistical confidence / probability for this assignment
74	2	Number of Contour Points	UINT16	Number of 2D points forming the object's contour.
76	No. of Points * 4	Contour Point List [number of Contour Points]		Array of Contour points. Please see the definition of a contour point.

Contour Point

Offset	Bytes	Description	Data type	Description
0	2	Position x	INT16	X-part/coordinate of this value/point in cm.
2	2	Position y	INT16	Y-part/coordinate of this value/point in cm.
4				

3.7 Data Type 0x2271: Object Data

Object data available from ScaLa B2 Laserscanners.
Each object list consists of a list header and concatenated objects.
Each object has a list of contour points. Subtypes are described below.

All positions and angles are given in the vehicle / reference coordinate system.

Coding in Big Endian Byte Order

The Object data structure consists of a list header and body which is a list of objects.

Object List Header:

Offset	Bytes	Object header	Data type	Description
0	8	Start Scan Timestamp	NTP64	64 bit timestamp at time of receive first measurement from the measurement core of the scan this object list is related to. Timer is started with power up of the sensor.
8	2	Scan number	UINT16	The scan number this object list belongs to
10	8	internal		
18	2	Number of objects	UINT16	Number of Objects in the list
20		Objects		

Object Data:
Object data consists of general, untracked and tracked properties.

Tracked Properties come from the tracking model, these values are filtered. The untracked properties are raw properties they are not filtered. They will e.g. not be available if an object is 100% occluded, but the Object is still predicted. In this case only the Tracked Properties are available.

Offset	Bytes	Object header	Data type	Description
0	4	Object ID	UINT32	Unique Object ID in this list
4	2	internal		
6	1	PropertiesAvailable	UINT8	These Flags indicate what kind of Properties are available for this particular Object. Bit1: untracked Properties Bit3: tracked Properties Others: internal
7	35+	Untracked Properties		
42+	76+	Tracked Properties		
118+	4	Internal	UINT32	

Untracked Properties:

Offset	Bytes	Object header	Data type	Description
0	1	Internal		
1	2	Relative Time of Measure	UINT16	In [µs] since Start ScanTime Stamp
3	4	Position closest Point	Point2Di	In [cm] the closest x/y Point location of this object in sensor coordinate system
7	2	Internal		
9	4	Object box size	Point2Di	In [cm] length/width
13	4	Object Box Size Sigma	Point2Dui	In [cm] width/length
17	2	Object box orientation	INT16	In [1/100 deg]
19	2	Object box orientation sigma	UINT16	In [1/100 deg]
21	2	Internal		
23	4	Tracking Point Coordinate	Point2Di	In [cm] x/y Point location of the point the Ibeo tracking would track
27	4	Tracking Point Coordinate Sigma	Point2Dui	In [cm] x/y
31	3	Internal		
34	1	Number of contour points	UINT8	
35	N*8		N* ContourpointType	A polygon line that describes the outline of the current object measurement

Tracked Properties:

Offset	Bytes	Object header	Data type	Description

0	1	Internal		
1	2	Object Age	UINT16	Number of scans in which this object has been tracked
3	2	Hidden Status Age (= prediction age)	UINT16	Number of scan the object has been predicted
5	1	Dynamic Flags	UINT8	Indicating the dynamic state of an object: Bit 0: 0: obj. in initialization phase 1: obj. in tracking phase Bit 1..3: reserved Bit 4..6**:** dynamic property 1: dynamic and moving 2: dynamic and stopped 3: internal 4: a priori stationary 5..7: reserved Bit 7: reserved
6	2	Relative Time of Measure	UINT16	In [µs] since Start ScanTime Stamp
8	4	Position closest Point	Point2Di	In [cm] the closest x/y Point location of this object in sensor coordinate system
12	4	Relative velocity	Point2Di	In [cm/s] x/y
16	4	Relative velocity Sigma	Point2Dui	In [cm/s] velocity x/y
20	1	Object class	UINT8	The class assigned by the classifier: 0: unclassified 1: unknown small 2: unknown big 3: pedestrians 4: two wheeler 5: car 6: truck 10: over drivable 11: under drivable Others: internal
21	1	Internal		
22	2	Classification Age	UINT16	How long has this object been classified with this class
24	2	internal		
26	4	Object box size	Point2Di	In [cm] length/width
30	4	Object Box Size Sigma	Point2Dui	In [cm] width/length
34	2	Object box orientation	INT16	In [1/100 deg]
36	2	Object box	UINT16	In [1/100 deg]

		orientation sigma		
38	1	Internal		
39	1	TrackingPoint Location	UINT8 (Enum)	The tracking point (stated below) is located at the following position of the object box. 0: Center of gravity 1: Front/Left 2: Front/Right 3: Rear/Right 4: Rear/Left 5: Front/Center 6: Right/Center 7: Rear/Center 8: Left/Center 9: Object Center 0xF: unknown
40	4	Tracking Point Coordinate	Point2Di	Coordinates (x/y) of the tracking point in the reference coordinate system. In [cm]
44	4	Tracking Point Coordinate Sigma	Point2Dui	In [cm] x/y
48	3	Internal		
51	4	velocity	Point2Di	In [cm/s] x/y
55	4	velocity Sigma	Point2Dui	In [cm/s] x/y
59	2	internal		
61	4	Acceleration	Point2Di	In[cm/s^2] x/y
65	4	Acceleration Sigma	Point2Dui	In[cm/s^2] x/y
69	2	Internal		
71	2	Yaw rate	INT16	In [1/10000 rad]
73	2	Yaw rate sigma	UINT16	In [1/10000 rad]
75	1	Number of contour points	UINT8	
76	N*8		List of Contourpoint Type	A polygon line that describes the outline of the current object measurement

Contour Point Type:

Offset	Bytes	Object header	Data type	Description
0	2	X	INT16	In [cm]
2	2	Y	INT16	In [cm]
4	1	X sigma	UINT8	In [cm]
5	1	Y sigma	UINT8	In [cm]
6	2	Internal		

3.8 Data Type 0x2280: ECU Object Data

Object data available from Ibeo API and Ibeo ECU connected with ScaLa Laserscanners. Each data block starts with the IbeoDataHeader followed by the object list.
For each object list this header is preceded. The IbeoDataHeader is described section 3.1.

Each object has a list of contour points. Subtypes are described below (Section 4).

All positions and angles are given in the vehicle / reference coordinate system.
Data is encoded in network byte order / big endian format.

In general, positions, lengths, distances and sizes are coded in meters.
In general, angles are coded in radians.

ECU Object Data List:

Offset	Bytes	Object header	Data type	Description
0	8	Mid-scan timestamp	NTP64	Mid-scan timestamp is the absolute timestamp when the scanner mirror crossed the middle of the corresponding scan.
8	2	Number of objects	UINT16	The number of objects transmitted in this message. **(*) This number currently reflects all objects on the Interface including internal object (hypotheses) only put out for debug and evaluation. See** 0 **Definition of object flags**
10		List of objects	Object	Array of objects.

Offset	Bytes	Object	Data type	Description
0	2	Object ID	UINT16	ID of this object from tracking.
2	2	Flags	UINT16	Bit 6: 0 = tracked by dyn. model 1 = tracked by static model Bit 7: 0 = mobility of dynamic obj. not (yet) detected 1 = mobility of dynamic obj. successfully detected Bit 8: 0 = motion model not validated 1 = motion model validated

				For a detailed description of these flags and combinations please see section 0 Definition of object flags
4	4	Object age	UINT32	Number of scans this object has been tracked for.
8	8	Timestamp NTP	NTP64	Time when this object was observed. More precisely: the reference point of this object. **Not available in ScaLa yet refer to mid-scan timestamp above for object time**
16	2	Object Prediction Age	UINT16	Number of scans this object has only been predicted without measurement updates.
18	1	Classification	UINT8	Most likely class of this object: 0: unclassified 1: unknown small 2: unknown big 3: pedestrian 4: bike 5: car 6: truck 7..9: reserved 12: under drivable 13: over drivable (n.a. yet)
19	1	Classification certainty	UINT8	The higher this value is the more reliable is the assigned object class. **Not available in ScaLa yet**
20	4	Classification age	UINT32	Number of scans this object has been classified as current class.
24	8	Reserved	Point2DFloat	
32	8	Reserved	Point2DFloat	
Following fields describe the object box. Please see 3.10 for detailed information.				
40	8	Object box center	Point2DFloat	Center point of this object.
48	8	Object box center sigma	Point2DFloat	Standard deviation of the object box center point. **Not available in ScaLa yet**
56	8	Object box size	Point2DFloat	Size of the object box in the object coordinate system (vehicle coordinate system rotated around z axis by object orientation angle).
64	8	Reserved	8 bytes	-
72	4	Object Box	FLOAT32	Orientation angle of the

		Orientation Angle		object box in [radians].
76	4	Orientation Angle Sigma	FLOAT32	Standard deviation of the object box orientation angle, in [rad] ∈ [-PI, +PI[
80	8	Relative velocity	Point2DFloat	Velocity of this object in m/s relative to the ego vehicle. Ego motion information is not taken into account here.
88	8	Relative velocity sigma	Point2DFloat	Standard deviation of the relative velocity. *Currently equals the absolute velocity sigma.*
96	8	Absolute velocity	Point2DFloat	Velocity of this object in m/s with ego motion taken into account. Inform about the object velocity in the 'real world'.
104	8	Absolute velocity sigma	Point2DFloat	Standard deviation of the absolute velocity. *Currently equals the relative velocity sigma.*
112	18	Reserved	18 bytes	-
130	1	Number of contour points	UINT8	Number of contour points transmitted for this object.
131	1	Index of closest point	UINT8	Closest contour point of this object as index of the point list.
132	2	Reference point location	UINT16	The reference point can be located at the following points: 0: Center of gravity 1: Front/Left 2: Front/Right 3: Rear/Right 4: Rear/Left 5: Front/Center 6: Right/Center 7: Rear/Center 8: Left/Center 9: Object Center *0xFF: unknown*
134	8	Reference point coordinate	Point2DFloat	Depending on tracking this is the tracked object reference point, i.e. position of reference point [in m].
142	8	Reference point sigma	Point2DFloat	Standard deviation of the estimated reference point position [in m].
150	4	Reference point	FLOAT32	Pearson's product-moment

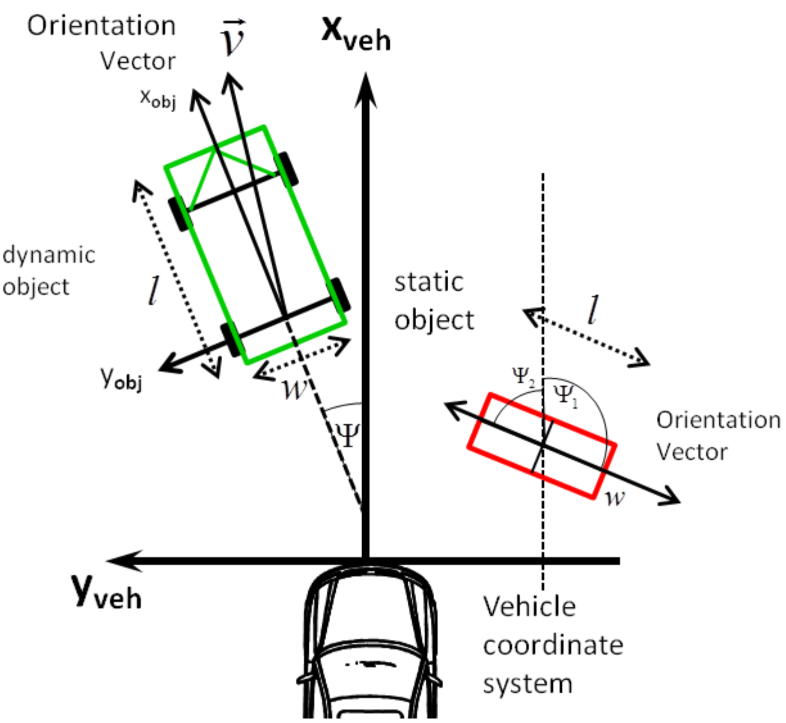
		position correction coefficient		coefficient. Scale: 10^{-3} **Not available in ScaLa yet**
154	8	Reserved		
162	2	Object priority	UINT16	Value determining priority. Value depends on performed algorithm for tracking processing. **Not available in ScaLa yet**
164	4	Object existence measurement	FLOAT32	**Not available in ScaLa yet**
168		List of contour points	Point2DFloat	Array of contour points (Point2DFloat).

3.9 Definition of object flags

Valid	Mobile detected	Static. Model/ Dynamic model	Description	Customer IF relevant	Put out for evaluation
0	0	0	Unvalidated object hypotheses	no	yes
0	0	1		no	yes
0	1	0	n.a.	No	No
0	1	1		No	No
1	0	0		no	no
1	**0**	**1**	**Validated stationary object (“a priori stationary”)**	**Yes**	**Yes**
1	**1**	**0**	**validated dynamic object with validated track (“moving”)**	**Yes**	**Yes**
1	**1**	**1**	**validated object, dynamic before, which is now stationary (“stopped”)**	**Yes**	**Yes**

3.10 Description of the object box

The object is represented by a rectangle (object box). It contains the object size and the object’s orientation.



The object box orientation angle Ψ is given in the vehicle's coordinate system. The orientation of a mobile-detected object is estimated based on the movement of the object. For a static (non-moving) object the orientation vector is defined being parallel to the long side of the object, therefore it could be either Ψ_1 or Ψ_2.

The object box size (Size_X, Size_Y) is given in the object coordinate system which is defined by the orientation vector and the vehicle's rear axis. That means the length l equals SizeX and the width w equals Size_Y.

3.11 Data Type 0x2403: Camera Image

Coding in Big Endian Byte Order!

Offset	Bytes	Image	Data type	Description
0	2	image format	UINT16	0 : JPEG , 1 : MJPEG, 2 : GRAY8, 3 : YUV420, 4 : YUV422
2	4	timestamp microseconds	UINT32	since power-on
6	8	NTP timestamp	NTP64	seconds; fractional seconds
14	1	device ID	UINT8	each IBEO device has a system wide unique id
15	24	Mounting position	Mounting-PositionF	Mounting position of the camera in vehicle coordinate system.
39	8	horizontal opening angle	DOUBLE64	radians
47	8	vertical opening angle	DOUBLE64	radians
55	2	image width	UINT16	pixel line count
57	2	image height	UINT16	pixel column count
59	4	compressed size	UINT32	size in bytes of the following image buffer
63	compressed size	Reserved	CAHR[]	image buffer
63 + compressed size				

3.12 Data Type 0x2805: Host's Vehicle State from ScaLa

The vehicle state is calculated by ScaLa-B2 from received CAN-Data.

Coding: Little Endian Byte Order.

All angles, position and distances are given in the ISO 8855 / DIN 70000 scanner coordinate system.

Bytes	Offset	Vehicle State:	Data type	Description
8	0	Timestamp	NTP64	
2	8	Scan number	UINT16	For synchronisation with Scan
2	10	Error flags	UINT16	***Currently not used in ScaLa-B1/B2***
2	12	Longitudinal velocity	INT16	Longitudinal Velocity [0.01m/s]
2	14	Steering wheel angle	INT16	Angle by which the steering wheel is rotated compared to its middle position. [0.001rad]
2	16	Front wheel angle	INT16	Wheel angle (calculated from steering wheel angle if available) [0.0001 rad]
2	18	Reserved		
4	20	X position	INT32	Distance from origin in X-Direction [0.01m]
4	24	Y position	INT32	Distance from origin in Y-Direction [0.01m]
2	28	Course angle	INT16	Orientation at time timestamp [0.0001 rad]
2	30	Time difference	UINT16	Time difference between this and last vehicle state message [ms]
2	32	X difference	INT16	Distance driven in X during time difference [0.001m]
2	34	Y difference	INT16	Distance driven in Y during time difference [0.001m]
2	36	Heading difference	INT16	Difference in Heading during time difference [0.0001 rad]
2	38	Reserved		
2	40	Current yaw rate	INT16	Yaw rate from latest CAN-Message received. Available since firmware version 2.5.00. [0.0001 rad/s]
4	42	Reserved		
	46			

3.13 Data Type 0x2806: Host's Vehicle State

It describes the current ego motion estimation that is used by the tracking and classification and all overlaying applications.

Offset	Bytes	Vehicle State:	Data type	Description
0	4	Reserved		
4	8	Timestamp	NTP64	Time stamp when this vehicle state was estimated
12	4	DistanceX	INT32	Distance from origin in x-direction [10^{-4}m]
16	4	DistanceY	INT32	Distance from origin in y-direction

				[10^{-4}m]
20	4	Course angle	FLOAT32	Course angle [rad]
24	4	Longitudinal velocity	FLOAT32	Longitudinal velocity [m/s]
28	4	Yaw rate	FLOAT32	Current yaw rate of vehicle [rad/s]
32	4	Steering wheel angle	FLOAT32	Angle by which the steering wheel is rotated compared to its middle position. [rad]
36	4	Cross Acceleration	FLOAT32	[m/s^2]
40	4	Front wheel angle	FLOAT32	Angle by which the front wheel is rotated compared to the vehicle's x-axis. [rad]
42	2	Reserved		
46	4	Vehicle Width	FLOAT32	Vehicle width in [m]
50	4	Reserved		
54	4	Distance: Vehicle's front to front axle	FLOAT32	Distance: front axle to vehicle's front [m]
58	4	Distance: rear axle to front axle	FLOAT32	Distance: vehicle's rear axle to vehicle's front axle [m]
62	4	Distance: rear axle to vehicle's rear.	FLOAT32	Distance: vehicle's rear axle to vehicle's rear [m]
66	4	Reserved		
70	4	SteerRatioPoly0 (s0)	FLOAT32	Coefficients for transfer function of steering wheel angle (x) $s_3x^3 + s_2x^2 + s_1x + s_0$
74	4	SteerRatioPoly1 (s1)	FLOAT32	
78	4	SteerRatioPoly2 (s2)	FLOAT32	
82	4	SteerRatioPoly3 (s3)	FLOAT32	
86				

All angles, position and distances are given in the ISO 8855 / DIN 70000 coordinate system.

3.14 Data Type 0x2807: Host's Vehicle State

It describes the current ego motion estimation that is used by the tracking and classification and all overlaying applications.

Data Type 0x2807 is based on 0x2806 with an additional datafield for longitudinal acceleration.

Offset	Bytes	Vehicle State:	Data type	Description
0	86	Vehicle State	0x2806	
86	4	Longitudinal Acceleration	FLOAT32	[m/s^2]
90				

3.15 Data Type 0x6301: Device Status

Coding in Little Endian Byte Order

Offset	Bytes	Scan header:	Data type	Description
0	6	Reserved	UINT16[3]	
6	1	Scanner Type	UINT8	Type of the scanner
7	29	Reserved		*for internal use*
36	4	Sensor Temperature	FLOAT32	Temperature of the sensor (near APD).
40	4	Frequency	FLOAT32	Frequency
44	124	Reserved		*for internal use*
168				

3.16 Reserved Data Types

Following data types are reserved for internal use and are not further specified. These data types can just be ignored on parsing the data stream.

Data type	Size
0x1002	*variable*
0x1100	32 bytes
0x4111	*variable*
0x6120	0 bytes
0x6130	*variable*
0x6430	*variable*
0x6940	*variable*

4 Subtypes

Coding: Byte order as described in data type description.

4.1 Primitive types

INT16	Signed 16 bit integer
UINT16	Unsigned 16 bit integer
INT32	Signed 32 bit integer
UINT32	Unsigned 32 bit integer
FLOAT32	Signed 32 bit floating point number (according IEEE)

4.2 NTP64

NTP64 timestamps represent the time encoded in 8 bytes. In order to decode NTP64 timestamps, the corresponding 8 bytes need to be interpreted as UINT64:

Offset	Bytes	Description	Data type	Description
0	8	Time stamp	UINT64	The higher 4 bytes are the number of seconds since 1.1.1900 - 0:00:00. The lower 4 bytes represent the fractional seconds with a resolution of $2^{(-32)}$ s

4.3 Point2D

Offset	Bytes	Description	Data type	Description
0	2	Position x	INT16	X-part/coordinate of this value/point in cm.
2	2	Position y	INT16	Y-part/coordinate of this value/point in cm.
4				

4.4 Point2DFloat

Offset	Bytes	Point2DFloat	Data type	Description
0	4	Position x	FLOAT32	X-part/coordinate of this value/point.
4	4	Position y	FLOAT32	Y-part/coordinate of this value/point.
8				

4.5 Size2D

Offset	Bytes	Description	Data type	Description
0	2	Size x	UINT16	X-value/size/width in cm.
2	2	Size y	UINT16	Y-value/size/length in cm.

4				

4.6 Sigma2D

Offset	Bytes	Description	Data type	Description
0	2	Sigma x	UINT16	Deviation in X direction.
2	2	Sigma y	UINT16	Deviation in Y direction.
4				

4.7 Velocity2D

Offset	Bytes	Description	Data type	Description
0	2	Velocity x	INT16	Velocity in X direction in cm/s.
2	2	Velocity y	INT16	Velocity in Y direction in cm/s.
4				

4.8 MountingPositionF

Offset	Bytes	Description	Data type	Description
0	4	Yaw Angle	FLOAT32	Mounting angles relative to vehicle coordinate system. [rad] in [−π, +π[.
4	4	Pitch Angle	FLOAT32	
8	4	Roll Angle	FLOAT32	
12	4	X Position	FLOAT32	Mounting position relative to vehicle coordinate system [m].
16	4	Y Position	FLOAT32	
20	4	Z Position	FLOAT32	
24				

5 Commands and replies

5.1 Command SetFilter to All Types: 0x2010

Before receiving any data from ECU this command needs to be sent to the ECU to configure the ECU according to the data which should be provided.

Data type to set in the Ibeo Data Header: 0x2010
Length: 8 Bytes

Offset	Bytes	Name	Data type	Value
0	2	Command Identifier	UINT16	0x0005
2	2	Version	UINT16	0x0002
4	2	Begin Filter Range	UINT16	0x0000
6	2	End Filter Range	UINT16	0xFFFF

After this command has been successfully executed by the ECU it will provide all data to the sender of this command.

5.1.1 Reply SetFilter: Success

After executing one command, the ECU replies the successful or unsuccessful operation, using this Reply-Data Type

Data type set in the Ibeo Data Header: 0x2020
Length: 2 Bytes

Offset	Bytes	Name	Data type	Value
0	2	Command Identifier	UINT16	0x0005

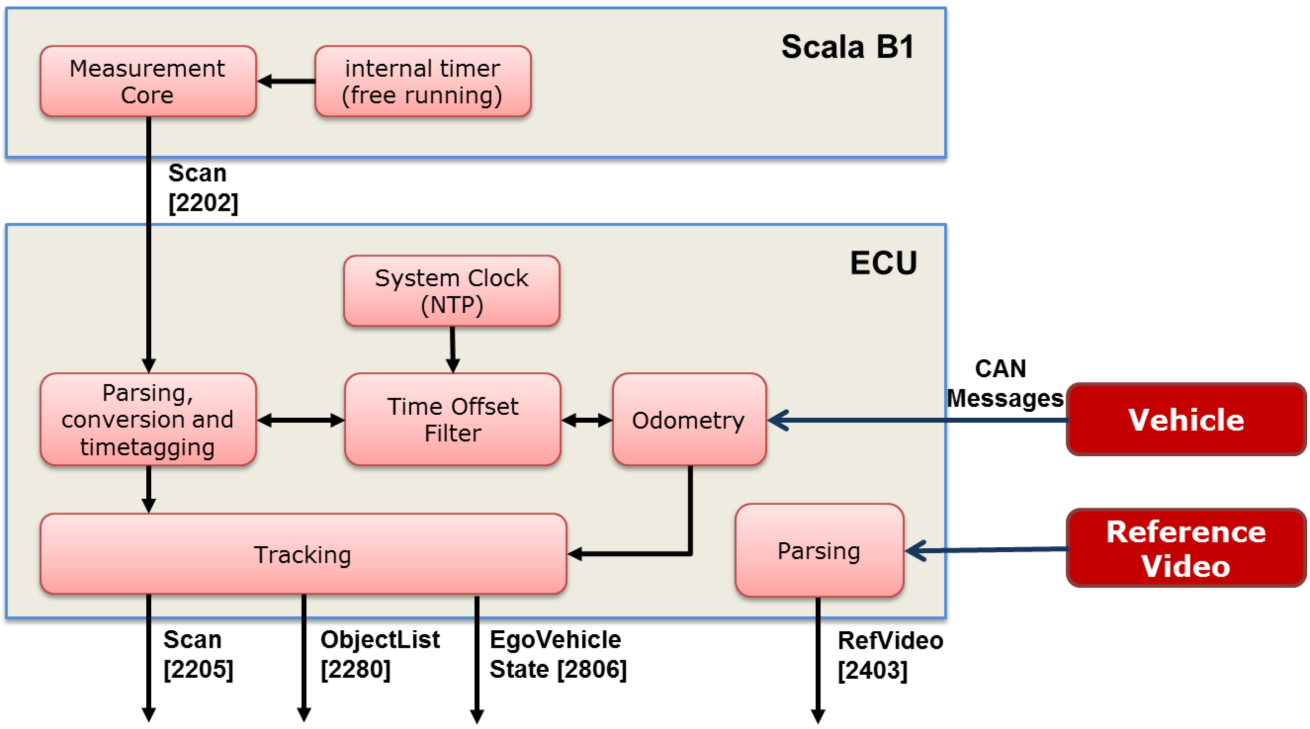
5.1.2 Reply SetFilter: Unsuccessful

Data type set in the Ibeo Data Header: 0x2020
Length: 2 Bytes

Offset	Bytes	Name	Data type	Value
0	2	Command Identifier	UINT16	0x8005

6 Dataflow and time stamping between ScaLa B1 and ECU

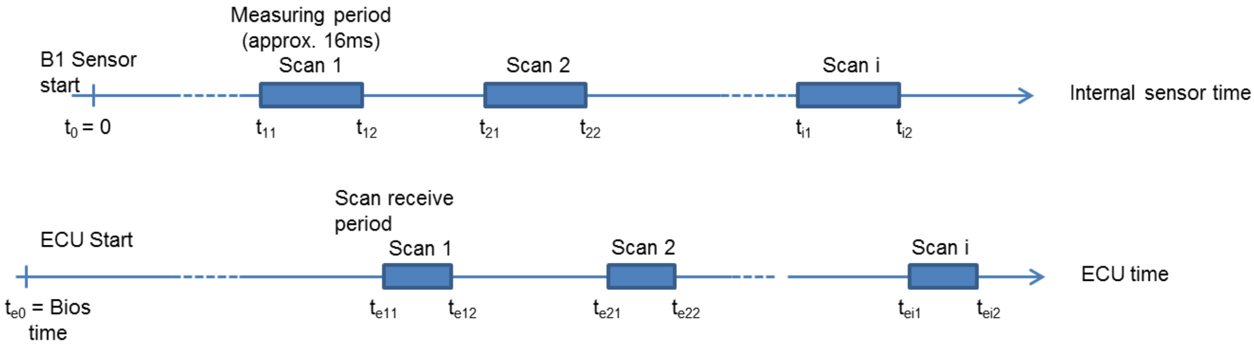
The following figure shows the dataflow of a system consisting of a ScaLa B1 and an ECU for object data processing.



The measured scan is transferred from the B1 sensor to the ECU as data type 2202. This data type contains a scan-start and scan-end timestamp in NTP format. As the ScaLa B1 only uses a free running internal clock without any external synchronization methodology (e.g. Set Timestamp Commands, synchronization by PTP), this time always start at a default time on startup of the sensor: (1.1.1900, 00:00:00).

The scan data received by the ECU is converted into a generic scan of data type 2205 before put into the tracking algorithm.
On receive the scan data it is time tagged with the current ECU time.

Within the time tagging, the offset between the clocks is filtered by a long term averaging, as described in the following figure:



Time Offset: $t_{of1} = t_{e12} - t_{11}$
$t_{of2} = t_{e22} - t_{21}$ …
$t_{ofi} = t_{ei2} - t_{i1}$

Time Offset filtered: t_{of} = sliding average over all t_{ofi}

The timestamps within the different headers are set according to the following table

Data Type	Field	Value
0x2202	Scan start time NTP	t_{i1}
0x2202	Scan end time NTP	t_{i2}
0x2205	Scan start time	$t_{i1} + t_{ofi}$
0x2205	Scan end time offset	$t_{i2} - t_{i1}$
0x2205	ScannerInfo: Scan start time	$t_{i1} + t_{ofi}$
0x2205	ScannerInfo: Scan end time	$t_{i2} + t_{ofi}$
0x2205	Scan start time from device	t_{i1}
0x2205	Scan end time from device	t_{i2}

According to this the timestamps are given in local time of the ECU. With respect to the time of measurement they show latency depending on

a) the duration of measuring period, resp. the time of the measurement within the measurement period
b) the data transfer time on the Ethernet interface

Data latency: $t_{ei2} - t_{i2}$

While the measurement period is a constant factor depending only on scan frequency and horizontal field of view, the data transfer time mainly depends on the Ethernet load and performance of data parsing routines.

The typical aggregated latency from the mid scan time to data receive in the ECU was measured to be approx. 20ms.