Ethernet data protocol
ibeo LUX and ibeo LUX systems

Version History

Date	Version	Changes
21.11.2008	1.0	Initial release
03.12.2008	1.1	Added section 9 with ibeo API trace message description
04.12.2008	1.2	Correction of description of data type 2224 (ObjectList), Field ScanNumber removed.
09.12.2008	1.3	New object data type 0x2225 (replaces 0x2224)
27.1.2009	1.4	• Corrected reply to GetStatus command – reserved word removed • Another reserved word in"SetNTPTimestampSec" and "SetNTPTimestampFracSec" command messages included • API and firmware version numbers integrated • FPGA and firmware version state are decimal coded
13.3.2009	1.5	Changed default IP-Address for LUX to 192.168.0.1
27.3.2009	1.6	Corrected sync phase offset in 0x2202.
6.4.2009	1.7	Comment added to set timestamp commands.
11.6.2009	1.8	Data type of parameter TCP/IP port fixed. Parameter name vehicle width fixed. Default IP configuration of Ibeo ECU changed.
16.6.2009	1.9	Corrected API/ECU data type trace message. Clarified reset default parameters command.
31.7.2009	1.10	Steering ratio types added. Default IP address of Ibeo ECU changed. Format and text changes.
01.10.2009	1.11	ECU Set filter command added.
05.10.2009	1.12	Fixed reserved values in set NTP time stamp commands.
14.10.2009	1.13	Clarified echo and layer encoding in 0x2202.
20.11.2009	1.14	Added new/future scan data type 0x2205.

Table of Content

1 Introduction

This document describes how data is received and transmitted from respectively to ibeo sensors and systems via the Ethernet connection.
Adressed systems are ibeo LUX sensors with the firmware version 1.2.x and ECUs or applications using the current Ibeo API/software version 1.5.
The data protocol also describes the file format of an *idc file, as the same data types as transferred via Ethernet are stored within these files.

2 General information

2.1 Ethernet configuration

ibeo LUX sensors and Ibeo ECUs use default ethernet configurations until changed by the user.

ibeo LUX sensors use the default IP address 192.168.0.1
with the subnet mask 255.255.255.0.
The default port is 12002.

Ibeo ECUs use the default IP address 192.168.0.10
with the subnet mask 255.255.255.0.
Default port for data connection is 12002.
Standard ports for telnet and FTP are used.

2.2 Data encoding

Attention! See the data type description if little or big endian byte order is used!

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:
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.

2.3 Ibeo data header

Each message always starts with an Ibeo data header. To resync just search for the “magic word”.
The Ibeo data header is encoded in network byte order / big endian format.

Bytes	Offset	Ibeo data header	Data type	Description
4	0	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.
4	8	Size of this message	UINT32	Helps to read the message data. Size of message content without this header.
1	12	Reserved	UINT8	-
1	13	DeviceID	UINT8	ID of the connected device. Unused in data received directly from ibeo LUX sensors.
2	14	Data type	UINT16	Specifies the data type within this message.
8	16	NTP time	NTP64	Time when this message was created.
	24	Message data	-	Depending on data type.

3 ibeo LUX scan data: Data type 0x2202

Scan data available from ibeo LUX laserscanners (not available for ibeo LUX prototypes). 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. An ibeo LUX typically 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.

Bytes	Offset	Scan header:	Data type	Description
2	0	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	0x0007: reserved 0x0008: set frequency reached 0x0010: external sync signal detected 0x0020: sync ok 0x0040: sync master (instead of slave) 0xFF80: reserved
2	4	Sync phase offset	UINT16	Phase difference (conversion factor 409.6 ns) between sync signal and scanner mirror crossing the synchronization angle.
8	6	Scan start time NTP	NTP64	NTP time when the first/last measurement was done.
8	14	Scan end time NTP	NTP64	
2	22	Angle ticks per rotation	UINT16	Number of angle ticks per rotation.
2	24	Start angle	INT16	Start/end angle in angle ticks of this scan.
2	26	End angle	INT16	
2	28	Scan points	UINT16	Number of scan point transmitted in this scan.
2	30	Mounting position 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.
2	32	Mounting position pitch angle	INT16	
2	34	Mounting position roll angle	INT16	

2	36	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 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.
2	38	Mounting position y	INT16	
2	40	Mounting position z	INT16	
2	42	Reserved	UINT16	-
	44	Scan Point List	Scan Point	Array of scan points. See number of scan points above and point information below.

Bytes	Offset	Scan point:	Data type	Description
1	0	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
2	4	Radial distance	UINT16	Distance of this point in the scanner coordinate system in cm
2	6	Echo pulse width	UINT16	Detected width of this echo pulse in cm
2	8	Reserved	UINT16	-
	10			

4 ibeo LUX object data: Data type 0x2221

Object data available from ibeo LUX laserscanners (not available for ibeo LUX prototypes).
Each data block starts with a header followed by the object list. Each object has a list of contour points.
The data is encoded in little endian format!

Bytes	Offset	Object header:	Data type	Description
8	0	Scan start timestamp	NTP64	Time stamp of the first measurement of the scan these objects are updated with.
2	8	Number of objects	UINT16	The number of objects transmitted in this message.
	10	List of objects	Object	Array of objects.

Bytes	Offset	Object: content	Data type	Description
2	0	Object ID	UINT16	ID of this object from tracking.
2	2	Object age	UINT16	Number of scans this object has been tracked for.
2	4	Object prediction age	UINT16	Number of scans this object has currently been predicted for without measurement update. Set to 0 as soon as a measurement update is available.
2	6	Relative timestamp	UINT16	Timestamp of this object relative to the scan start time in ms. The time is based on the object reference point.
4	8	Reference point	Point2D	Depending on tracking this is the tracked object reference point (e.g. center of gravity) in cm. See below for Point2D.
4	12	Reference point sigma	Point2D	Standard deviation of the estimated reference point position in cm.
4	16	Closest point	Point2D	Unfiltered position of the closest object point in cm.
4	20	Bounding box center	Point2D	Center and size in cm of a rectangle in the reference coordinate system containing all object points. See below for Size2D.
4	24	Bounding box size	Size2D	
4	28	Object box center	Point2D	Box center in the reference coordinate system in cm.
4	32	Object box size	Size2D	

2	36	Object box orientation	INT16	Box size in cm and orientation in 1/32° in the object coordinate system (box rotated by orientation in reference coordinate system).
4	38	Absolute velocity	Point2D	Velocity of this object in cm/s with ego motion taken into account. This velocity is based on the reference coordinate system which is compensated by the ego motion. Value set to 0x8000 if invalid.
4	42	Absolute velocity sigma	Size2D	Standard deviation of the estimated absolute velocity in cm/s.
4	46	Relative velocity	Point2D	Velocity of this object in cm/s without ego motion compensation (sensor/vehicle is seen as stationary).
2	50	Classification	UINT16	Most likely class of this object: 0: unclassified 1: unknown small 2: unknown big 3: pedestrian 4: bike 5: car 6: truck 7…: reserved
2	52	Classification age	UINT16	Number of scans this object has been classified as current class for.
2	54	Classification certainty	UINT16	The higher this value is the more reliable is the assigned object class.
2	56	Number of contour points	UINT16	The number of objects transmitted in this message.
	58	List of contour points	Point2D	Array of contour points in cm.

Bytes	Offset	Point2D:	Data type	Description
2	0	Position x	INT16	X-part/coordinate of this value/point.
2	2	Position y	INT16	Y-part/coordinate of this value/point.
	4			

Bytes	Offset	Size2D	Data type	Description
2	0	Size x	UINT16	X-value/size/width.
2	2	Size y	UINT16	Y-value/size/length.
	4			

5 ibeo LUX errors and warnings - Data type 0x2030

As soon as an ibeo LUX Laserscanner detects an error or wants to emit a warning, this message is sent. Errors and warning bits are reset after sending this message.
This message will be sent periodically as long as errors of warnings persist.
The data is encoded in little endian format!

Bytes	Offset	LUX error/warning	Data type	Description
2	0	Error register 1	bit field 16 bits	See below
2	2	Error register 2	bit field 16 bits	
2	4	Warning register 1	bit field 16 bits	
2	6	Warning register 2	bit field 16 bits	
2	8	reserved	bit field 16 bits	
2	10	reserved	bit field 16 bits	
2	12	reserved	bit field 16 bits	
2	14	reserved	bit field 16 bits	

5.1 Error register 1

Bytes	LUX error	Description	Comment
Bit 0	E-SP	internal error	contact support
Bit 1	E-Motor_1	motor fault	contact support
Bit 2	E-Buffer_1	scan buffer transmitted incompletely	decrease scan resolution/frequency/range; contact support
Bit 3	E-Buffer_2	Scan buffer overflow	decrease scan resolution/frequency/range; contact support
Bit 4	E-Meas_1	APD voltage failed	contact support
Bit 5		reserved	
Bit 6		reserved	
Bit 7		reserved	
Bit8…9	E-Temp	Bit 9: APD Over Temperature Bit 8: APD Under Temperature Bit 8 and 9: APD Temperature Sensor defect	provide cooling provide heating contact support
Bit 10	E-Motor_2	motor fault	contact support
Bit 11	E-Motor_3	motor fault	contact support
Bit 12	E-Motor_4	motor fault	contact support
Bit 13	E-Motor_5	motor fault	contact support
Bit14…15		reserved	

5.2 Error register 2

Bytes	LUX error	Description	Comment
Bit 0	E-IF_internal_1	no scan data received.	contact support
Bit 1	E-IF_internal_2	internal communication error	contact support
Bit 2	E-IF_internal_3	incorrect scan data	contact support
Bit 3	E-Configuration_1	FPGA not configurable	contact support
Bit 4	E-Configuartion_2	incorrect configuration data	load correct configuration values
Bit 5	E-Configuration_3	configuration contains incorrect parameters	load correct configuration values
Bit 6	E-Timeout_1	data processing timeout	decrease scan resolution or scan frequency
Bit 7	E-Timeout_2	reset the computation of the environmental model	contact support
Bit8…15	reserved		

5.3 Warning register 1

Bytes	LUX warning	Description	Comment
Bit0	W-CMD	internal communication error	
Bit1	W-Range_1	internal warning	
Bit2	W-Range_2	internal warning	
Bit3	W-low_temperature	temperature too low	warning of insufficient temperature
Bit4	W-high_temperature	temperature too high	warning of exceeding temperature
Bit5	W-Motor_1	internal warning	
Bit6	W-Motor_2	internal warning	
Bit 7	W-Sync	syncronisation error	check syncronisation- and scan frequency
Bit7…15	RES 7…15	reserved	

5.4 Warning register 2

Bytes	LUX warning	Description	Comment
Bit0	W-IF_CAN	CAN Interface blocked	check CAN bus and CAN connection
Bit1	E-IF_ETH	Ethernet Interface blocked	check Ethernet connection
Bit2	W-CANdata	incorrect CAN message received	check CAN data
Bit3	W-IF_internal_1	incorrect scan data	contact support
Bit4	W-ETHdata	unknown or incomplete data	check Ethernet data
Bit5	W-Command	incorrect or forbidden command received	check command
Bit6	W-Flash	memory access failure	restart ibeo LUX, contact support
Bit7	W-Overflow_1	internal overflow	contact support
Bit8	W-EgoMotion	vehicle data update missing	check CAN vehicle data
Bit9	W-Mounting Position	incorrect mounting parameters	correct mounting position according to OM
Bit10	W-CalcFrequency	no object computation due to scan frequency.	set the scan frequency to 12.5 Hz to receive objects
Bit11…15	reserved		

6 ibeo LUX command interface

For sending commands to the ibeo LUX the data type 0x2010 is used. The data is encoded in little endian format!

Bytes	Offset	LUX command	Data type	Description
2	0	Command ID	UINT16	See detailed list of commands and according options/parameters.
2	2	Reserved	UINT16	Unused, but these 2 bytes must be sent for all commands.
	4	Command Data	-	Depending on command. May be completely missing for some commands.

The ibeo LUX replies to a command with a dedicated reply message. The data type used is 0x2020. The data is encoded in little endian format!

Bytes	Offset	LUX reply:	Data type	Description
2	0	Reply ID	UINT16	If a command succeeded, the reply ID is equal to the corresponding command ID. If a command failed, the reply ID is the command ID + 0x8000. Thus, the most significant bit indicates a failed command.
	2	Reply data	-	Depending on the corresponding command this reply is related to. May be completely missing for some commands and if a command failed. See detailed command description below.

6.1 ibeo LUX commands and command replies – data types 0x2010/ 0x2020

6.1.1 Reset

Bytes	Offset	LUX command	Data type	Description
2	0	0x0000	UINT16	ID - Reset DSP
2	2	Reserved	UINT16	-

In case of command Reset no reply is sent.

6.1.2 Get Status

Bytes	Offset	LUX command	Data type	Description
2	0	0x0001	UINT16	Status request
2	2	Reserved	UINT16	-

Bytes	Offset	LUX reply	Data type	Description
2	0	0x0001	UINT16	Status request
2	2	Firmware version	UINT16	e. g. 0x1230 = version 1.2.3, 0x123B = version 1.2.3b
2	4	FPGA version	UINT16	e. g. 0x1230 = version 1.2.3, 0x123B = version 1.2.3b
2	6	Scanner status	UINT16	Bit field, with the following meaning for every bit: Bit 15 ...6: reserved / internal Bit 5: phase locked Bit 4: external sync signal available Bit 3: frequency locked Bit 2: reserved / internal Bit 1: laser on Bit 0: motor on
4	8		UINT32	reserved / internal
2	12	temperature	UINT16	T[°C] = - (temperature - 579.2364) / 3.63
2	14	serial number 0	UINT16	YYCW (z. B. YYCW = 0x0740 = year '07, calendar week 40)
2	16	serial number 1	UINT16	Counter of serial number
2	18		UINT16	reserved / internal
6	20	FPGA time stamp	[3] * UINT16	YYYY MMDD hhmm (FPGA version state decimal coded)
6	26	DSP time stamp	[3] * UINT16	YYYY MMDD hhmm (Firmware version state decimal coded)

6.1.3 SaveConfig

Bytes	Offset	LUX command	Data type	Description
2	0	0x0004	UINT16	Current sensor configuration will be saved permanently. Multiple SetParameter commands may be sent before saving the changes permanently.
2	2	Reserved	UINT16	-

The command SaveConfig will be acknowledged by the same command ID without command reply data.

6.1.4 Set Parameter

Bytes	Offset	LUX command	Data type	Description
2	0	0x0010	UINT16	Set a single Parameter by its index to the sensor memory. Parameter is set only temporarily until a SaveConfig command (see 6.1.3) is sent.
2	2	Reserved	UINT16	-
2	4	Parameter index	UINT16	Refer to ibeo LUX parameter list (see 6.2)
4	6	Parameter	UINT32	Set parameter accordingly to parameter list. If e.g. a 2 byte value is set, use the first 2 bytes. Fill the remaining 2 bytes with 0.

The command Set Parameter will be acknowledged by the same command ID without any command reply data.

6.1.5 Get Parameter

Bytes	Offset	LUX command	Data type	Description
2	0	0x0011	UINT16	Read a single Parameter with its index from the LUX.
2	2	Reserved	UINT16	-
2	4	Parameter index	UINT16	Refer to LUX parameter list (see 6.2)

Bytes	Offset	LUX reply	Data type	Description
2	0	0x0011	UINT16	Read a single Parameter by its index from the LUX.
2	2	Parameter index	UINT16	Refer to ibeo LUX parameter list (see 6.2)
4	4	Parameter	UINT32	

6.1.6 Reset Default Parameters

Bytes	Offset	LUX command	Data type	Description
2	0	0x001A	UINT16	Resets all parameters to the factory defaults.
2	2	Reserved	UINT16	-

The command Reset Default Parameters will be acknowledged by the same command ID without any command reply data.
Send SaveConfig command (see 6.1.3) to reset default parameters permanently after this command.

6.1.7 Start Measure

Bytes	Offset	LUX command	Data type	Description
2	0	0x0020	UINT16	Starts the measurement with the current settings.
2	2	Reserved	UINT16	-

The command Start Measure will be acknowledged by the same command ID without any command reply data.

6.1.8 Stop Measure

Bytes	Offset	LUX command	Data type	Description
2	0	0x0021	UINT16	Stops the measurement.
2	2	Reserved	UINT16	-

The command Stop Measure will be acknowledged by the same command ID without any command reply data.

6.1.9 SetNTPTimestampSec

Bytes	Offset	LUX command	Data type	Description
2	0	0x0030	UINT16	sets the second of NTPtimestamp.
2	2	Reserved	UINT16	-
2	4	Reserved	UNIT16	-
4	6	Timestamp	UINT32	Seconds (NTP format). The time will be set in the sensor when the fractional seconds command is received (see 0).

The command SetNTPTimestampSec will be acknowledged by the same command ID without any command reply data.
Timestamp will be used when the SetNTPTimestampFracSec command is received (see below).

6.1.10 SetNTPTimestampFracSec

Attention: Before this command can be executed, first command "SetNTPTimestampSec" (0x0030) must be sent (see 6.1.9)!

Bytes	Offset	LUX command	Data type	Description
2	0	0x0031	UINT16	sets the fractional second of NTPtimestamp.
2	2	Reserved	UINT16	-
2	4	Reserved	UINT16	-
4	6	Timestamp	UINT32	Fractional seconds (NTP format).

The command SetNTPTimestampFracSec will be acknowledged by the same command ID without any command reply data.

6.2 ibeo LUX parameter list

This table gives an overview of available ibeo LUX parameters. Please refer to 6.1.4 and 6.1.5 for details on getting and setting these parameters.

IP address, subnet mask and standard gateway encode the data as UINT32 value which is built like that: aa.bb.cc.dd = 0xaabbccdd. Due to little endian byte order this value must be sent as 0xddccbbaa.

Bytes	Parameter index	LUX parameter	Data type	Description
4	0x1000	IP address	UINT32	Valid: all
2	0x1001	TCP Port	UINT16	Valid: all
4	0x1002	Subnet Mask	UINT32	Valid: all
4	0x1003	Standard gateway	UINT32	Valid: all
4	0x1010	CAN Base ID	UINT32	Valid: value <= 0x7F0
2	0x1011	CAN Baud Rate	UINT16	in kBaud - next matching value (1000 kBaud, 500 kBaud, 250 kBaud, 125 kBaud) will be used.
2	0x1012	Data Output Flag	16 bit field	Bit true: disable output, false: enable output. 0xFFFF is invalid. bit0: ETH scan data bit1: reserved/internal bit2: ETH object data bit3: ETH vehicle data bit4: ETH errors/warnings bit5: CAN errors/warnings bit6: CAN object data bit7...15: reserved/internal
2	0x1013	maxObjectsViaCAN	UINT16	<= 65 (max. number of objects) limited by tracking and CAN bus capacity.
2	0x1014	ContourPointDensity	UINT16	Valid: < 3 0: closest point only 1: low density 2: high density
2	0x1015	ObjectPriorizationCriterion	UINT16	Valid: < 2 Used to reduce transmitted objects via CAN. Decision which objects are discarded is based on this criterion. 0: Radial 1: Look ahead

Bytes	Parameter index	LUX parameter	Data type	Description
2	0x1016	CAN object data options	16 bit field	Valid: all bit 0: 0 = absolute velocities, 1 = relative velocities bit 1: 0 = boxes are object boxes, 1 = boxes are bounding boxes bits 2…15: reserved
2	0x1017	Minimum Object Age	UINT16	Valid: all Minimum tracking age (number of scans) of an object to be transmitted.
2	0x1018	Maximum Prediction Age	UINT16	Valid: all Maximum prediction age (number of scans) of an object to be transmitted.
2	0x1100	Start angle	INT16	In 1/32°, in the sensor coordinate system. Valid: 1600…-1919. Start angle > end angle!
2	0x1101	End angle	INT16	In 1/32°, in the sensor coordinate system. Valid: 1599…-1920. Start angle > end angle!
2	0x1102	Scan frequency	UINT16	In 1/256 Hz. Valid: 3200 (12.5 Hz) 6400 (25.0 Hz) 12800 (50.0 Hz)
2	0x1103	Sync angle offset	INT14 (!) 16 bits used	In 1/32° in the sensor coordinate system. Valid: -5760…+5759 (-180°…+180°). Bits 14 and 15 are ignored!
2	0x1104	angular resolution type	UINT16	0: focused 1: constant 2: reserved
2	0x1105	angleTicksPerRotation	UINT16	11520 (read only), constant for ibeo LUX
2	0x1200	SensorMounting_X	INT16	In cm, related to vehicle reference point, rear axle. Order of translation and rotation is essential (Rotation -> Translation).

Bytes	Parameter index	LUX parameter	Data type	Description
2	0x1201	SensorMounting_Y	INT16	In cm, related to vehicle reference point, rear axle. Order of translation and rotation is essential (Rotation -> Translation).
2	0x1202	SensorMounting_Z	INT16	In cm, related to vehicle reference point, rear axle. Order of translation and rotation is essential (Rotation -> Translation).
2	0x1203	SensorMounting_Yaw	INT16	In 1/32°, order of translation and rotation is essential (Yaw->Pitch->Roll-> Translation).
2	0x1204	SensorMounting_Pitch	INT16	In 1/32°, order of translation and rotation is essential (Yaw->Pitch->Roll-> Translation).
2	0x1205	SensorMounting_Roll	INT16	In 1/32°, order of translation and rotation is essential (Yaw->Pitch->Roll-> Translation).
2	0x1206	VehicleFrontToFrontAxle	UINT16	valid: all; in cm
2	0x1207	FrontAxleToRearAxle	UINT16	valid: all; in cm
2	0x1208	RearAxleToVehicleRear	UINT16	valid: all; in cm
2	0x1209	VehicleWidth	UINT16	valid: all; in cm
2	0x120A	steerRatioType	UINT16	0: Transmission ratio Front wheel angle = $\frac{x}{1.095(s_3x^3 + s_2x^2 + s_1x + s_0)}$ 1: Transfer function Front wheel angle = $s_3x^3 + s_2x^2 + s_1x + s_0$ x = steering wheel angle
4	0x120C	SteerRatioPoly0 (s0)	Float32	valid: all
4	0x120D	SteerRatioPoly1 (s1)	Float32	valid: all
4	0x120E	SteerRatioPoly2 (s2)	Float32	valid: all
4	0x120F	SteerRatioPoly3 (s3)	Float32	valid: all
2	0x1210	Vehicle Motion Data Flags	16 bit field	Bit 0: Vehicle Motion data expected: 1=true, 0=false Bits 1 to 15: reserved

6.3 Example

This example shows how to set the IP address via Ethernet 192.168.0.200.

Bytes	Offset	Ibeo data header Big endian byte order!	Data type	Content
4	0	Magic word	UINT32	0xAFFEC0C2
4	4	Size of previous message	UINT32	Not mandatory. Set e.g. to 0: 0x00000000
4	8	Size of this message	UINT32	0x000000XX
1	12	Reserved	UINT8	0x00
1	13	Device ID	UINT8	Not mandatory. Set e.g. to 7: 0x07
2	14	Data type: ibeo LUX command	UINT16	0x2010
8	16	NTP timestamp	UINT64	Not mandatory. Set e.g. to 0: 0x0000000000000000
Bytes	Offset	Message data Little endian byte order!	Data type	Content
2	24	Command ID: Set parameter	UINT16	0x0010 (send encoded as 0x1000)
2	26	Reserved	UINT16	0x0000
2	28	Parameter index: IP address	UINT16	0x1000 (send encoded as 0x0010)
4	30	Parameter data (here: 192.168.0.200)	UINT32	0xC0A800C8 (send encoded as 0xC800A8C0)
	34			

7 Ibeo API/ECU scan data - Data type 0x2204

Scan data available from Ibeo API and Ibeo AppBase2 (ECU) are sent as data type 0x2204 until API version 2.1 and Ibeo Laserview 1.5. Please see data type 0x2205 for later versions.
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 which refers to a sensor in the sensor info list. The data is encoded in network byte order / big endian format.

Bytes	Offset	Scan header	Data type	Description
8	0	Scan start time	NTP64	NTP time when the first measurement was done.
4	8	Scan end time offset	UINT32	Time difference between last and first measurement in us.
4	12	Flags	Bit field: 32 bits	Bit 0: ground labeled Bit 1: dirt labeled Bit 2: rain labeled Bits 3…8: reserved Bit 9: fused scan Bit 10: reserved Bit 11: coordinate system (0 = scanner coordinates, 1 = vehicle coordinates)
2	16	Scan number	UINT16	The number of this scan. The number will be increased from scan to scan. Overflow occurs after 2^{16} scans.
2	18	Scan points	UINT16	Number of scan points transmitted in this scan.
1	20	Number of scanner infos	UINT8	Number of scanner infos transmitted in this scan.
3	21	Reserved	3 bytes	-
	24	Scanner info list	Scanner info	Array of scanner infos. See number of scanner infos above and scanner info below.
	24 + scanner infos * 40	Scan point List	Scan point	Array of scan points. See number of scan points above and point information below.
	24 + scanner infos * 40 + scan points * 28			

Bytes	Offset	Scanner info	Data type	Description
1	0	Device ID	UINT8	Device ID of this scanner.
1	1	Scanner type	UINT8	3 = Alasca XT 4 = ECU 5 = ibeo LUX prototype 6 = ibeo LUX
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	-
4	8	Start angle	FLOAT32	Field of view of this scanner given in its local coordinate system. In radians normalized to [−π, +π[.
4	12	End angle	FLOAT32	
4	16	Yaw angle	FLOAT32	Mounting angles relative to vehicle coordinate system. In radians normalized to [−π, +π[.
4	20	Pitch angle	FLOAT32	
4	24	Roll angle	FLOAT32	
4	28	Offset x	FLOAT32	Mounting position relative to vehicle coordinate system. In meters.
4	32	Offset y	FLOAT32	
4	36	Offset z	FLOAT32	
	40			

Bytes	Offset	Scan point:	Data type	Description
4	0	X position	FLOAT32	X position of this scan point in m.
4	4	Y position	FLOAT32	Y position of this scan point in m.
4	8	Z position	FLOAT32	Z position of this scan point in m.
4	12	Echo width	FLOAT32	Echo width of this scan point in m.
1	16	Device ID	UINT8	ID of the device measuring this point.
1	17	Layer	UINT8	Scan layer of this point (zero-based).
1	18	Echo	UINT8	Echo number of this point (zero-based).
1	19	Reserved	1 byte	-
4	20	Timestamp (µs)	UINT32	Time offset in µs when this scan point was measured based on the scan start time.
2	24	Flags	Bit field: 16 bits	0x0001: ground 0x0002: dirt 0x0004: rain/snow/spray/fog/… 0xFFF8: reserved
2	26	Reserved	2 bytes	-
	28			

8 Ibeo API/ECU scan data - Data type 0x2205

Scan data available from Ibeo API and Ibeo AppBase2 (ECU) are sent as data type 0x2205 using API version 2.2 and later and Ibeo Laserview 1.6 and later. Please see data type 0x2204 for earlier versions.
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 which refers to a sensor in the sensor info list. The data is encoded in network byte order / big endian format.

Bytes	Offset	Scan header	Data type	Description
8	0	Scan start time	NTP64	NTP time when the first measurement was done.
4	8	Scan end time offset	UINT32	Time difference between last and first measurement in us.
4	12	Flags	Bit field: 32 bits	Bit 0: ground labeled Bit 1: dirt labeled Bit 2: rain labeled Bits 3…8: reserved Bit 9: fused scan Bit 10: reserved Bit 11: coordinate system (0 = scanner coordinates, 1 = vehicle coordinates)
2	16	Scan number	UINT16	The number of this scan. The number will be increased from scan to scan. Overflow occurs after 2^{16} scans.
2	18	Scan points	UINT16	Number of scan points transmitted in this scan.
1	20	Number of scanner infos	UINT8	Number of scanner infos transmitted in this scan.
3	21	Reserved	3 bytes	-
	24	Scanner info list	Scanner info	Array of scanner infos. See number of scanner infos above and scanner info below.
	24 + scanner infos * 40	Scan point List	Scan point	Array of scan points. See number of scan points above and point information below.
	24 + scanner infos * 40 + scan points * 28			

Bytes	Offset	Scanner info	Data type	Description
1	0	Device ID	UINT8	Device ID of this scanner.
1	1	Scanner type	UINT8	3 = Alasca XT 4 = ECU 5 = ibeo LUX prototype 6 = ibeo LUX
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	-
4	8	Start angle	FLOAT32	Field of view of this scanner given in its local coordinate system. In radians normalized to [−π, +π[.
4	12	End angle	FLOAT32	
8	16	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.
8	24	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.
8	32	Scan start time from device	NTP64	NTP time (as received from the sensor) when the first measurement of this scanner was done.
8	40	Scan end time from device	NTP64	NTP time (as received from the sensor) when the first measurement of this scanner was done.
4	48	Scan frequency	FLOAT32	Scan frequency of this scanner in Hz.
4	52	Beam tilt	FLOAT32	Angle the scanner measurement is pitched relatively to sensor x-y plane. 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.

4	56	Scan flags	Bit field: 32 bits	Bit 0: Ground detection was performed Bit 1: Dirt detection was performed Bit 2: Clutter detection was performed Bits 3…8: Reserved Bit 9: Scan is result from scan data fusion Bit 10: Mirror side Bits 11..31: Reserved
4	60	Yaw angle	FLOAT32	Mounting angles relative to vehicle coordinate system. In radians normalized to [−π, +π[.
4	64	Pitch angle	FLOAT32	
4	68	Roll angle	FLOAT32	
4	72	Offset x	FLOAT32	Mounting position relative to vehicle coordinate system. In meters.
4	76	Offset y	FLOAT32	
4	80	Offset z	FLOAT32	
8	84	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.
8	92	Resolution 2	Resolution Info	
8	100	Resolution 3	Resolution Info	
8	108	Resolution 4	Resolution Info	
8	116	Resolution 5	Resolution Info	
8	124	Resolution 6	Resolution Info	
8	132	Resolution 7	Resolution Info	
8	140	Resolution 8	Resolution Info	
	148			

Bytes	Offset	Resolution Info:	Data type	Description
4	0	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			

Bytes	Offset	Scan point:	Data type	Description
4	0	X position	FLOAT32	X position of this scan point in m.
4	4	Y position	FLOAT32	Y position of this scan point in m.
4	8	Z position	FLOAT32	Z position of this scan point in m.
4	12	Echo width	FLOAT32	Echo width of this scan point in m.
1	16	Device ID	UINT8	ID of the device measuring this point.
1	17	Layer	UINT8	Scan layer of this point (zero-based).
1	18	Echo	UINT8	Echo number of this point (zero-based).
1	19	Reserved	1 byte	-
4	20	Timestamp (µs)	UINT32	Time offset in µs when this scan point was measured based on the scan start time.
2	24	Flags	Bit field: 16 bits	0x0001: ground 0x0002: dirt 0x0004: rain/snow/spray/fog/… 0xFFF8: reserved
2	26	Reserved	2 bytes	-
	28			

9 Ibeo API/ECU object data - Data type 0x2225

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

Bytes	Offset	Object header	Data type	Description
8	0	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.
2	8	Number of objects	UINT16	The number of objects transmitted in this message.
	10	List of objects	Object	Array of objects.

Bytes	Offset	Object	Data type	Description
2	0	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.
2	16	Object hidden status age	UINT16	Number of scans this object has only been predicted without measurement updates.
1	18	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
1	19	Classification certainty	UINT8	The higher this value is the more reliable is the assigned object class.
4	20	Classification age	UINT32	Number of scans this object has been classified as current class.
8	24	Bounding box center	Point2D	Center point of the bounding box of this object. See below for definition of Point2D.
8	32	Bounding box size	Point2D	Size of the bounding box (a rectangle parallel to vehicle

				coordinate system).
8	40	Object box center	Point2D	Center point (tracked) of this object.
8	48	Object box center sigma	Point2D	Standard deviation of the object box center point.
8	56	Object box size	Point2D	Size of the object box in the object coordinate system (vehicle coordinate system rotated around z axis by object course angle).
8	64	Reserved	8 bytes	-
4	72	Yaw angle	FLOAT32	Orientation or heading of the object in radians.
4	76	Reserved	8 bytes	-
8	80	Relative velocity	Point2D	Velocity of this object in m/s relative to the ego vehicle. Ego motions is not taken into account here.
8	88	Relative velocity sigma	Point2D	Standard deviation of the relative velocity.
8	96	Absolute velocity	Point2D	Velocity of this object in m/s with ego motion taken into account. Inform about the object velocity in the ‘real world’.
8	104	Absolute velocity sigma	Point2D	Standard deviation of the absolute velocity.
18	112	Reserved	18 bytes	-
1	130	Number of contour points	UINT8	Number of contour points transmitted for this object.
1	131	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.

Bytes	Offset	Point2D	Data type	Description
4	0	Position x	FLOAT32	X-part/coordinate of this value/point.
4	4	Position y	FLOAT32	Y-part/coordinate of this value/point.
	8			

10 Ibeo API/ECU Trace Messages - Data types 0x6400 .. 0x6440

Software modules which are deploying the Ibeo API for communication can sent trace messages consisting of a character string.
Trace Messages are distributed with 4 different data types dependent on their priority:

- Data type 0x6400 - Error
- Data type 0x6410 - Warning
- Data type 0x6420 - Note
- Data type 0x6430 – Debug Info

Bytes	Offset	Data	Data type	Description
1	0	Trace level	UINT8	Gives the trace level of this message. Currently this accords to the data type. 1=error, 2=warning, 3=note, 4=debug
Size of string	1	Trace message	String	Contains warnings and errors received from connected sensors. E.g. “*IbeoLUX3 "ibeoLUX": DSP warning: Invalid vehicle motion data. To avoid this warning please uncheck 'Vehicle Motion Data expected' in device configuration. Code: 0x0100*”
1	Size of string + 1	0x00	UINT8	End of string byte 0x00.
	Size of string + 2			

11 Ibeo API/ECU Set Filter Command – Data type 0x2010

Before the Ibeo API or ECU sends data after connecting to it (default port 12002), a filter command must be sent.
Data is encoded in network byte order / big endian format.

Bytes	Offset	Data	Data type	Description
2	0	Command ID: Set filter	UINT16	0x0005
2	2	Number of following list entries	UINT16	The number of following entries in the filter list. Example: set to 0x0002 to receive all data types.
2 * n	4	Set of start and stop to define ranges of data types	2 x UINT16 per filter value pair	Number (n) of values defining the range of data types. Example: set to 0x0000 0xffff (2x UINT16) to receive all data types (n = 2).
	4 + 2 * n			

Complete hex data of this example command looks like this:
af fe c0 c2 00 00 00 00 00 00 00 08 00 00 20 10 00 00 00 00 00 00 00 00
00 05 00 02 00 00 ff ff
First line is the Ibeo data header, second the command data.

The command will be replied by Ibeo API or ECU with data type 0x2020.
The reply data is 0x0005 (the received command ID).