Programmer's manual

SLPxxx sensors family

TINP Communication Protocol

Robust, long range laser profiler for harsh environments

Triple-IN

LASER TECHNIC

Table of contents

Revisions

Date	FW	Description	Ref
2021.12.14	0.0.00.01	First draft	4353
2021.12.20	0.0.00.01	Added description for feature-sets: inp0-inp3, out0-out3	4353
2022.01.10	5.1.00.10	Added description for new INFO payload	4353
2022.03.15	5.2.00.00	Updated to firmware improvements	4353
2022.10.09	5.2.00.00	Updated to firmware improvements	4353
2023.07.27	5.2.00.50	Updated to latest firmware	4353

1. Introduction

1.1 About this document

This document describes the Triple-IN's sensors Programmer's manual available in the following sensors:

- SLP100, SLP250, SLP300, SLP400 (referred as a series as SLPxxx)
- SLP HT
- SLP PRO

The information of this document refers to the firmware version `5.2.00.00` or earlier.

This document is part of a series of documents:

Manual	Targeted personnel	Content
Operating manual	Technical personnel	Transport, mounting and installation Wiring and maintenance Operating means, system configuration Technical data
Programmer's manual	Software developers	Data formats Commands and responses

If you or your colleagues have any comments on this manual, we would be grateful to hear from you. Please write to:

```
Triple-IN GmbH

Poppenbütteler Bogen 64
D-22399 Hamburg - Germany
Telephone +49(0)40 50091998
Mail support@triple-in.com
```

1.2 Web site with documents and firmware updates

The latest version of this and other documents and the latest firmware updates can be obtained from Triple-IN's web site.

2. Communication interface

The communication between a controlling program and the sensors is provided by an Ethernet interface.

Depending on the model, this could be 100MB/s or 1GB/s.

Please refer to the *Operating manual* of the specific sensor to get more information about the available interfaces.

2.1 UDP/IP transport protocol

The UDP transport protocol can be used to send commands to the sensor, receive responses and receive the online scan stream.

2.2 TCP/IP transport protocol

The TCP transport protocol can be used to send commands to the sensor and receive responses. It is possible but not recommended to use TCP/IP to receive online scan stream.

2.3 Why choose UDP for scan data stream

TCP was designed to be reliable in data transmission. This means that, if a packet of data is not correctly (without errors) received by the destination, the sender will try to transmit it again until it is received correctly.

In a real-time system though it is more important to have always the most recent data available even if it means to have some previous data lost.

The UDP transport protocol has a lot of advantages for this purpose:

- It is stateless, suitable for very large numbers of clients. It is used for example in streaming media applications such as IPTV
- The lack of re-transmission delays makes it suitable for real-time applications such as Voice over IP, online games, and many protocols built on top of the Real Time Streaming Protocol
- It works well in unidirectional communication and is suitable for broadcast information such as in many kinds of service discovery and shared information such as broadcast time or Routing Information

Protocol

All these attributes (and more) made the UDP protocol the best choice to send the scan data stream to the requesting consumers.

2.4 Sensor network configuration

In the described communication pattern, the sensor provides some functionalities accessible through the Ethernet. Each one of these functionalities is identified as a "*Service*", and the control computer is the "*Client*".

The sensor has the possibility to start sending a scan data stream automatically on power-up to a predefined **Client**, identified as the "*Default Consumer*" by a user defined combination of IP address and port. This specific function is called **Auto-start** and can be configured using the `asos` feature-set (see Paragraph 8.1).

The sensor socket addresses are a combination of an IP address and a port (which is mapped to the application program process). Every available combination identifies a **Service**.

The sensor has two different IP addresses configured and exposed to the outside:

- Predefined. Is automatically calculated from the serial number and cannot be changed. The network mask is `255.255.0.0` and it is calculated this way:
  `IP = 10.255.(serial / 100).(serial % 100)`
- Custom. This address can be modified by the user. It is initially set with a default value calculated similarly to the Predefined:
  `IP = 10.0.(serial / 100).(serial % 100)`

2.5 Timeout

The command interface of the firmware has a general maximum timeout of 10 seconds. Different operations on different sensor models could lead to longer timeouts.

Please refer to the specific sensor model's **Operating Manual** for further details.

2.6 Sensor services announcement

Most of Triple-IN's sensors can send a message containing information about the configuration of the sensor itself. This is useful to discover the IP addresses of one or more sensors connected to a network.

To receive this message, it is necessary to broadcast a specific command to the sensors network on a specific port.

Details on this protocol are defined later in this document.

2.7 Services

The following are the services available by default on a sensor. As an example, we list the IP addresses for a sensor with the serial number as `1234`:

Service IP Port	Protocol	Service	Description
10.255.12.34 3993	UDP/IP TCP/IP	Scan and commands	Command communication line for configuration and online data stream
10.0.12.34 3993	UDP/IP TCP/IP	Scan and commands	Command communication line for configuration and online data stream
0.0.0.0 6996	UDP/IP	Services announcement	Specific command sent to this channel will trigger the services broadcast
0.0.0.0 3007	TCP/IP	Update service	Positive power supply voltage
10.255.12.34 6969	UDP/IP TCP/IP	Scan and commands	Command communication line for PS communication protocol (1)
10.0.12.34 1024	UDP/IP TCP/IP	Scan and commands	Command communication line for PS communication protocol (1)

(1) Some Triple-IN's sensor models provide additionally the compatibility to the communication protocol used in Triple-IN's Psxxx-90 sensors series.
Please refer to the sensor Operation's manual for further information.
Description of the PS communication protocol can be found in the **Psxxx-90 Plus Programmer's Manual**.

2.8 Limitations

2.8.1. Concurrent connections

The Update service is limited to one connection at a time to avoid the chance to have multiple updates run at the same time.

Scan and command services can be used by a maximum of 10 clients at the same time.

2.8.2. Custom service IP and port

The IP address set by the user cannot be the same as the Predefined.

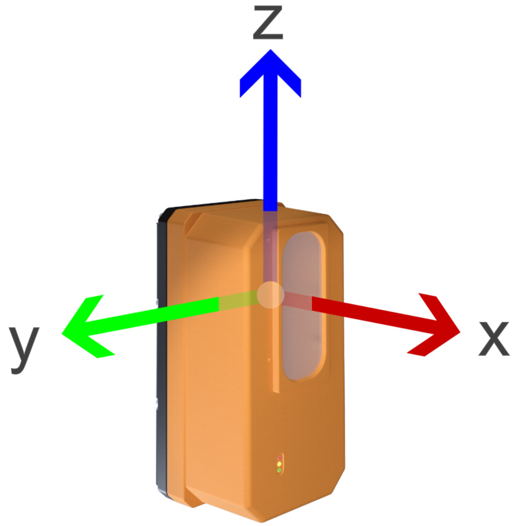
The Port set by the user cannot be one of: 1 to 1023, 3007, 3993, 4884, 6969 and 6996.

3. Coordinates system

This chapter describes the coordinates system referenced by this manual.

3.1 Sensor's reference system

The sensor's reference system is described in the picture below.



SLP coordinates system

- The X axis is growing in the measuring direction
- The Y axis is growing to the right direction, looking straight to the X
- The Z axis is growing to the up direction

A point measured by the sensor can be represented with spherical or cartesian coordinates.

Depending on the data format chosen by the user, the sensor can return one or the other coordinates.

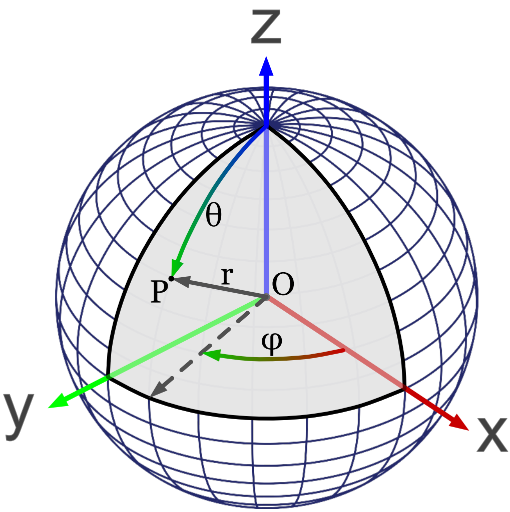
3.2 Spherical coordinates

To define the spherical coordinates system, the two reference orthogonal directions are defined as following:

- The *zenith* direction corresponds to the *Z* axis
- The *azimuth* direction corresponds to the *X* axis

The spherical coordinate of the point P are defined as follows:

- The radius *r* is the euclidean distance from the point *P* to the origin *O*
- The inclination *θ* (or polar angle) is the angle between the zenith direction and the line segment *OP*
- The azimuth (or azimuthal angle) *φ* is the signed angle measured from the azimuth reference direction to the orthogonal projection of the line segment *OP* on the reference plane



Spherical coordinates system on the SLP

The point P is identified by:

- P (r,θ,φ)

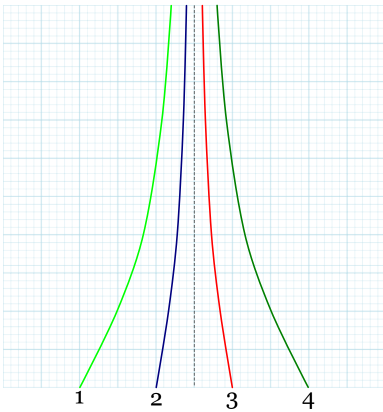
3.3 Cartesian coordinates

The Cartesian coordinates may be retrieved from the spherical coordinates (radius r, inclination θ, azimuth φ), where $r \in [0, \infty)$, $\theta \in [0, \pi]$, $\varphi \in [0, 2\pi)$, by:

```
X = r • sin( θ ) • cos( φ )
Y = r • sin( θ ) • sin( φ )
Z = r • cos( θ )
```

3.4 SLP-ML coordinate system

The SLP-ML is a special product that uses a clever polygon tilting to generate up to 4 scan lines lines.



Ideal projection on a flat surface generated by a SLP-ML with 4 scan lines

In those specific sensor model, only the two format Polar and XYZ are available, providing the spherical or Cartesian coordinates for every point on the scan lines.

Please note that the order of arrival of the scan line packets to the user does not follow the above numeration.

4. Data types and data formats

4.1 Byte order and basic data types

Whenever not differently specified, the sensor uses the little-endian byte order in the data definition.

Bit 0 Bit 31

Word at address 0000			
Half-word at address 0000		Half-word at address 0002	
Byte ad address 0000	Byte ad address 0001	Byte ad address 0002	Byte ad address 0003

Basic data types are shown in the next table:

Type name	Octets (bits)	Remark
Char	1 (8)	An Ansi character [-128..127]
Int8	1 (8)	A signed byte [-128..127]
UInt8	1 (8)	An unsigned byte [0..255]
Int16	2 (16)	A signed word [-32768..32767]
UInt16	2 (16)	An unsigned word [0..65535]
Int32	4 (32)	A signed double word [-2147483648..2147483647]
UInt32	4 (32)	An unsigned double word [0..4294967295]
Int64	8 (64)	A signed quad word [$-2^{31}..2^{31}-1$]
UInt64	8 (64)	An unsigned quad word [$0..2^{32}-1$]
String	4+N+1+pad	A buffer composed by a Uint32 containing N, followed by N x octets (Ansi chars) and a 0. N+1 is always rounded to 4

4.2 Specific data formats

4.2.1. Distances

Range values are stored unsigned in a total of `24 bits`. Unless otherwise specified, the value is stored and transferred into an `UInt32`.

The data unit is **1/10th** of a millimetre.

In some sensor models, a shifting factor can be applied to provide higher values to be transferred.

Some special values must be checked to neglect invalid measurements.

Bits	Value / Hex	Remark
Basic type: UInt32		
24	16777215/0×FFFFFF	Invalid
32	4294967295/0×FFFFFFFF	
24	16777214/0×FFFFFE	Echo has been classified as Noise
32	4294967294/0×FFFFFFFE	
24	16777213/0×FFFFFD	The power of the echo is very low
32	4294967293/0×FFFFFFFD	
24	16777212/0×FFFFFC	No echo has been detected
32	4294967292/0×FFFFFFFC	

As a general rule:

- With a 24bit value, every value bigger than `0×FFFFF0` should be discarded as invalid.
- With a 32bit value, every value bigger than `0×FFFFFFF0` should be discarded as invalid.

4.2.2. Echo pulse width

The echo pulse width can be used to represent the strength of the returning signal. Pulse widths are provided as `24 bits` values in pico-seconds and generally stored in a `UInt32`.

4.2.3. Target reflectivity

The reflectivity of a target surface depends on various parameters, such as the colour and the structure. Some Triple-IN sensors provide an estimated information about the target reflectivity.

The reflectivity is provided as a `UInt8` value from 0 to 255.

Value	Remark
Basic type: UInt8	
`0`	No echo
`100`	100% reflectivity
`101-254`	High reflectors, not convertible in percentage
`255`	Noisy echo

4.2.4. Echo number

Measuring through rain, through protection windows, or in dusty surroundings, several objects may reflect the laser beam, before or after the target of measuring interest.

Triple-IN's sensors can evaluate and return multiple echoes.

Sensor model(s)	Max. evaluated echoes	Max. returned echoes
PSxxx-90 Standard	4	1
PSxxx-90 Plus	4	4
SLPxxx	4	4
SLP PRO	6	6

Different data formats are provided to return the data measured. For the formats that provides one or two echo is given the possibility to chose the priority over the first and the last.

The **Echo number** field, when present in the data format, defines the order of the echo in the evaluation time line of a single beam.

4.2.5. Angles

The angle unit depends on a circle with 360000000 counts (millionth-degree). In 2D sensors, the scan start angle, the scan stop angle and the number of a measurement must be used to calculate the scan direction of a single laser pulse.

Angles are normally store in a `Int32`.

4.2.6. Temperature

The sensor handles the temperature readings in 0.1 Celsius. They are normally stored in a `Int16`.

Example: a reading of “352” means "35.2° Celsius".

4.2.7. Time stamp

The unit of the time stamp is microseconds and it is stored in a `UInt64` value.

Some Triple-IN’s sensors support a “One pulse per second” 1PPS signal. Each rising edge of the 1PPS signal is used to synchronise the internal time to the one provided by the NMEA input on the serial port, usually provided by a GPS device.

4.2.8. Strings and buffers

Strings are often used by the sensors to send information to the controlling computer. In that case, the concept of buffer is used.

String length	String chars	Terminator	4-bytes padding

For example, the string “123456789” will be transferred as the following flow of bytes (in hex):

String length	String chars	Terminator	4-bytes padding
09 00 00 00	31 32 33 34 35 36 37 38 39	00	00 00

The zero terminator is mandatory, so even if a string contains 8 characters (that are already padded to 4, like “12345678”), the terminator must be added, together with the necessary padding:

String length	String chars	Terminator	4-bytes padding
08 00 00 00	31 32 33 34 35 36 37 38	00	00 00 00

But if the string have, for example, length 7 like “1234567”, the padding bytes are not necessary:

String length	String chars	Terminator	4-bytes padding
07 00 00 00	31 32 33 34 35 36 37	00	

4.3 System status bit fields

Triple-IN's sensors perform a self test at power-up and continuously checks the overall status of the system.

The results of those tests are stored into 3 separate bit fields that are provided within the measurements data stream.

Bit field	Type	Description
Info	`UInt32`	Provides simple informational flags. They can be used, for example, to check if certain attributes are on or off without the need to ask for the information.
Warnings	`UInt32`	A bit set to 1 provides the information that some unexpected situations were detected and corrected. The sensor is still working but results could not be as expected. Usually these bits can be restored to normal with a power cycle. If this does not happen, the sensor should go to factory for maintenance
Errors	`UInt32`	A bit set to 1 provides the information that the specific behaviour of the sensor is not working properly. The sensor must be sent to factory for repair

The complete list of bits meaning, including a detailed description, can be found in Appendix B.

Some of the bits could be specific to particular sensor models. Please refer to the sensor's User's manual for a more detailed description of this fields.

4.4 CRC checksum

Triple-IN's sensors use two different checksum types.

4.4.1. CRC16 XMODEM

This Cyclical Redundancy Checking is used for small data packages and it is stored in a UInt16 basic type.

```
Polynomial        = 0x1021
Initial reminder = 0x0000
Final XOR value   = 0x0000
Ref. IN           = no
Ref. OUT          = no
Check word        = 0x31C3
```

The check word is the result of the algorithm executed on the char sequence "123456789".

4.4.2. CRC32

This Cyclical Redundancy Checking is used for the complete Triple-IN data package and it is stored in a UInt32 basic type.

```
Polynomial        = 0x04c11db7
Initial reminder  = 0xffffffff
Final XOR value   = 0xffffffff
Ref. IN           = yes
Ref. OUT          = yes
Check word        = 0xCBF43926
```

The check word is the result of the algorithm executed on the char sequence "123456789".

4.5 Error codes

All communication exchanges with Triple-IN's sensors may result in some kind of error. This could be depending on communication issues, sensor status or implementation related responses.

All the possible errors are identified by a numerical error code, usually negative and represented with an `Int32`.

A description of the Triple-IN's sensors error code can be found in Appendix C.

5. Communication packages

The communication between the control computer and the sensor is performed using data packages. Four main types of packages can be identified:

1. **Command**
   When the data package is sent by the control computer to the sensor, it is commonly called *Command*.
2. **Response**
   In case the data package is sent by the sensor to the control computer in response to a *Command*, it is called *Response* when the command was processed successfully
3. **Error**
   In case the data package is sent by the sensor to the control computer in response to a *Command*, it is called *Error* when the command processing was somehow unsuccessful
4. **Event**
   When the Message datagram is sent by the sensor to the control computer without a corresponding command is simply called Message.

Every data package follow the same basic structure.

The package header and payload are encapsulated between a Preamble and a Terminator.

Preamble	Header	Payload	Terminator

5.1 Preamble

Offset	Name	Type	Value
0	`Identifier`	`UInt32`	`0x54494e50` or “`TINP`”
4	`Length`	`UInt32`	The header and payload sizes combined.

- **Identifier**

  The preamble identifier marks the start of a Triple-IN data package.

  This identifier alone is not enough to ensure that the following data is really a Triple-IN data package. It is recommended to parse and check the Length information following this field and the Terminator data following the header and the payload to ensure the validity of the package.

- **Length**

  The value contained in this four bytes word is the length in octets of the header and the payload. This value does not include the Preamble and Terminator data.

  Due to UDP constraints, the maximum possible value of this field is `65451`.

  The minimum allowed value is `0` (no contents), identifying a `NULL` package.

5.2 Terminator

Offset	Name	Type	Value
0	`Identifier`	`UInt32`	`0×50494e54` or “`PINT`”
4	`CRC32`	`UInt32`	The CRC32 calculated on header and payload.

- **Identifier**

  The terminator identifier marks the end of a Triple-IN data package payload.

  As for the preamble identifier, this identifier alone is not enough to ensure the validity of the preceding data. It is recommended to use this in combination with the Preamble information and the next CRC32 field.

- **CRC32**

  This field contains the CRC32 value calculated on the package header and payload. Together with the Preamble data, the validity of this value ensures the validity of the header and the payload.

5.3 Header

Offset	Name	Type	Value
0	`Length`	`UInt8`	The length of the header in octets [1]
1	`Version`	`UInt8`	The version of this header (current: 1)
2	`Flags`	`UInt16`	Header and payload identification flags
4	`Command ID`	`UInt32`	Identifier for the data payload
8	`Sequence ID`	`UInt32`	Package sequence number
12	`Auth. token`	`UInt32`	Authorisation token
16	`Reserved`	`UInt32`	Reserved for future or internal use. Set to 0
20	`Reserved`	`UInt16`	Reserved for future or internal use. Set to 0
22	`CRC16`	`UInt16`	Calculated on the header bytes before CRC16

- **Length**

Contains the length of the packet header, including the length itself. It is always 4 bytes aligned.

- **Version**

Contains a number identifying the version of the header description.

This value shall be used to recognise if we are able to understand the header content. In case the version is not recognised as known, we must skip the entire package. Triple-IN will change the header description only when it is absolutely necessary for adding new attributes.

- **Flags**

This is a bit-field which contains specific information on the payload type and contents.

[1] For header version 1, the header length is always 24.

Offset	Bits	Name	Value
`0`	2	`Payload type`	Describes the type of the payload: 0 = Command 1 = Response 2 = Error 3 = Event
2	14	`Reserved`	The CRC32 calculated on header and payload.

- ## Command ID

Word identifying the command, response or event contained in the payload.

The command identifiers are 4 bytes words and every byte is the ASCII representation of a upper case English alphabet character.

This constraint can be also used as an extra check for packet data validation.

- ## Sequence ID

This field is at user disposal. Any value written here on a command sent to the sensor will be returned in the same field on the response generated by the sensor.

This will make easy for the control computer to asynchronously connect a response to the triggering command.

- ## Authorisation token

Almost all commands require a certain authorisation level (role) to be executed. As described in Chapter 5, it is necessary to login and get the authorisation token to send most of the command requests to the sensor.

Specific commands are allowed to be sent by the virtual Guest user, using the authorisation token `0`.

- ## CRC16

CRC16 calculated on the header contents excluding the CRC16 value itself.

On sending a command, this value can be optionally set to `0` and the sensor will ignore it and avoid the check, relying only on the data packet CRC32 value to check the data correctness.

5.4 Payload

The payload contents are completely dependent on the payload type and command ID defined in the header.

6. Authorisation system

Triple-IN's sensor access to system parameters and information is protected by an authorisation system.

Every command specifies which users can use it and some commands require also specific permissions for different purposes of the same command.

The list of available users is shown in the next table.

User	Default password	Description
`admin`	"`password`"	The administrative user with all rights
`developer`	"`password`"	User dedicated to developers
`operator`	"`password`"	For operator in the field that can change some parameters
`viewer`	"`password`"	View-only user
`guest`	< Cannot login >	The active user when no authorisation is requested

The `admin` user has the highest rank and possesses all the rights on all the available commands. It is also the only user that can change the passwords for the other users.

It is strongly recommended to change the password(1) of the `admin` user before setting the sensor into operation at the site, to avoid that unauthorised personnel can make any unwanted change on the sensor configuration.

The rights available for every user will be described in every single command in the following paragraphs.

6.1 Sessions

The authorisation system is based on the concept of session.

When a client connects to the sensor for the first time, the sensor creates a

(1) In case of forgotten `admin` password, the sensor has a procedure that allows also the `guest` user to restore the password. In this case, the password will not be restored to the default "`password`" but to a special "restore password" that will be provided by Triple-IN on request.

session associated to this specific client. Every operation carried on by this client will be performed in the context of the same session.

In case of a TCP connection, the session is directly linked to the existence of the TCP connection itself. When the TCP connection is established, a new session is created. When the TCP connection is closed or broken, the session is closed as well.

In case of a UDP connection, the session is identified by the client's IP:Port. If the control computer creates a new UDP connection from a different IP or port, the sensor will create a new session.

Since a UDP connection cannot be closed or broken, the session defines a timeout after which a session will be closed if no data packed is received. The control computer could use a timed NOOP (Paragraph 6.4.4) command to keep the connection alive.

When a session is closed, a new authorisation token must be requested to restore the desired user rights.

6.2 Roles

Every user has a role identifier associated with it. We can use this role identifier to check the result of an authorisation command.

User	Role	Role ID
`admin`	`admin`	`0x6689` / `26249`
`developer`	`developer`	`0x2690` / `9872`
`operator`	`operator`	`0xce08` / `52744`
`viewer`	`viewer`	`0x1f2e` / `7982`
`guest`	`guest`	`0`

7. Payloads

7.1 Commands

The following commands are supported by Triple-IN's sensors, in alphabetical order.

Command	Hex	Description
AUTH	0×48545541	Manages the authorisation flow
GVER	0×52455647	Requests the version string
INFO	0×4f464e49	Returns useful information about the sensor system
NOOP	0×504f4f4e	No effect. Used as a keep-alive command
PASS	0×53534150	Changes the password of the current or specified user
QRYF	0×46595251	Requests a feature-set
QRYI	0×49595251	Requests the inputs status
QRYM	0×4d595251	Requests the current scan mode
QRYO	0×4f595251	Requests the outputs status
RESP	0×50534552	Resets the password for a specified user
SCAN	0×4e414353	Requests a scan data stream
SETF	0×46544553	Sets the contents of a feature-set
SETM	0×4d544553	Sets the current scan mode or applies changed settings for the current
SETO	0×4f544553	Sets one of more outputs status
STOF	0×464f5453	Permanently stores the content of feature-sets

7.2 Responses

Every command can have a corresponding response with the same identifier. Additionally, the following response can be returned by the sensor on every command.

Response	Hex	Description
EREP	0×50455245	Returns an error code in response of an unidentifiable or malformed data package

7.3 Events

When the sensor is actively measuring, a continuous flow of scan data can be requested. This data stream sends so called events to the client computer without waiting for any command request.

Event	Hex	Description
`LDTA`	`0x4154444c`	Contains a 2D profile measured by the sensor

7.4 Authorisation management

7.4.1. AUTH

This command is used to acquire or release an authorisation on a specific role.

To acquire the authorisation, the username and password must be sent in the payload as a string buffer, separated by a colon character.

To release the authorisation, username and password must be empty, sending only the colon character.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	Yes
`guest`	Yes

- **Command payload contents**

Ofs	Type or Length	Description
`0`	`String`	String containing username and password separated by a colon. Eg: "`user:password`" to acquire the permissions or "`:`" to release them

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	Authorisation token
`4`	`UInt32`	Role identifier
`8`	`String`	Role name

7.4.2. PASS

Changes the password for the specified or current user.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	No
`guest`	No

- **Command payload contents**

Ofs	Type or Length	Description
`0`	`Buffer`	String containing username and password separated by a colon. Eg: "`user:passcode`"

When the “user” part is missing, the password will be changed for the current session user, also invalidating the authorisation and requiring the acquisition of a new authorisation token.

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	Authorisation token
`4`	`UInt32`	Role identifier
`8`	`String`	Role name

Authorisation token is always set to `0` when the password request change was directed to another user.

7.4.3. RESP

Restores the password for a specified user.

It is necessary to acquire the correct pass-code from Triple-IN in order to use this feature. When required, Triple-IN will provide the pass-code to the purchaser party of the sensor.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	Yes
`guest`	Yes

- **Command payload contents**

Ofs	Type or Length	Description
`0`	`String`	String containing username and pass-code(1) separated by a colon. Eg: "`user:passcode`"

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	Authorisation token
`4`	`UInt32`	Role identifier
`8`	`String`	Role name

7.4.4. NOOP

The no-operation command is used as a heartbeat for the session to ensure that connected client is still active. This is not necessary on TCP connections, but can be used as a no-effect command to test the response from the sensor.

The no-operation command does not contain any payload data.

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	Yes
`guest`	Yes

The NOOP command and response payloads are both empty.

7.5 Scan management

7.5.1. QRYM

Queries the sensor in which scan mode is set.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	Yes
`guest`	No

- **Command payload contents**

The payload is empty.

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	The scan mode. This is one of: - 0 = Idle - 2 = 2D scanning

7.5.2. SETM

Sets the sensor scan mode.

The switch between scan modes can require some time. During this time, some commands sent to the sensor can return a “System not ready” error.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	No
`guest`	No

- **Command payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	The scan mode. This is one of: - 0 = Idle - 2 = 2D scanning

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	The scan mode. This is one of: - 0 = Idle - 2 = 2D scanning

7.5.3. SCAN

The scan command is used to receive a continuous stream of measurement data from the sensor.

It is possible to use a different address and port as the target for the stream. In this case, data will be sent to the address and port specified via UDP datagrams.

Also a configurable timeout can be used to handle the timeout behaviour for a session.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	No
`guest`	No

- **Command payload contents**

Ofs	Type or Length	Description
0	UInt32	Options bit-field: 0 - Enable (1) or disable (0) the stream
4	UInt32	Destination IPv4. Set to 0 to send the stream to the command source client
8	UInt16	Destination port
10	UInt16	Reserved for future use
12	UInt32	Reserved for future use
16	UInt64	Session timeout in seconds

- **Response payload contents**

Ofs	Type or Length	Description
0	UInt32	Confirmation for the options bit-field: 0 - Stream enabled (1) or disabled (0)

7.6 Configuration management

The configuration of Triple-IN’s sensors is achieved using feature-sets.

Feature-sets are groups of attributes that, collected together, define a certain behaviour of the sensor.

Different feature-sets are distinguished through a feature-set identifier, which is a 4-bytes words with every byte as the ASCII representation of lower case English alphanumeric characters.

7.6.1. Feature-set descriptor

Attributes are transmitted as feature-sets. A descriptor is provided to define the data content transmitted.

In certain cases, only a subset of the attributes on a feature-set is transmitted, allowing for example to modify only some of the attributes concurring to influence a certain behaviour.

In the following table is shown the composition of the descriptor.

Ofs	Type	Name	Description
0	UInt32	ID	Feature-set identifier
4	UInt32	Enabled mask	Bit-field defining which attributes are transmitted
8	UInt32	Length	Length of attributes payload
12	UInt32		Reserved for future use

The attributes on a feature-set are identified by a number between 0 and 31, for a maximum of 32 attributes. This number also defines the order of transmission.

The second field of the descriptor, the bit-field, is used to declare which attribute is actually transmitted. Depending on the bit value at the position equivalent to the attribute number:

0. The corresponding attribute is not sent. This means that the data will not be present in the payload and the place will be occupied by the next attribute
1. The corresponding attribute is sent and the corresponding data will be present in the stream

7.6.2. QRYF

This command is used to query the content of a specific feature-set.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Depends on the feature-set id
`operator`	Depends on the feature-set id
`viewer`	Depends on the feature-set id
`guest`	No

- **Command payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	Feature-set identifier

- **Response payload contents**

The feature-set payload will always contain all the attributes.

Ofs	Type or Length	Description
`0`	`16`	Feature-set descriptor
`16`	`Variable`	The attributes composing the specific feature-set

7.6.3. SETF

This command is used to modify all or some of the attributes in a specific feature-set.

The changes made by SETF are not permanent unless a corresponding STOF command (see next paragraph) is used. This means that all the changes made with SETF will be lost at the next power cycle.

Roles access

Role	Access
`admin`	Yes
`developer`	Depends on the feature-set id
`operator`	Depends on the feature-set id
`viewer`	No
`guest`	No

Command payload contents

As described before, the sender can choose to transmit only a small set of attributes to partially change the feature-set's influenced behaviour.

The Enabled bit-field of the descriptor must be set accordingly to the actual attributes transmitted.

As an extension of this paradigm, also a subset of zero attributes is allowed.

Ofs	Type or Length	Description
`0`	`16`	Feature-set descriptor
`16`	`Variable`	The attributes composing the specific feature-set

Response payload contents

In the response payload, all the attributes of the feature-set are always transmitted, allowing the sender to check possible side effects of the operation (eg.: An attribute not transmitted by the command must be changed to allow the required behaviour).

Due to this, sending a **SETF** command with an empty subset of attributes will have the same effect of a **QRYF** command.

Ofs	Type or Length	Description
`0`	`16`	Feature-set descriptor
`16`	`Variable`	The attributes composing the specific feature-set

7.6.4. STOF

Stores all or some modified feature-sets.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Depends on the feature-set id
`operator`	Depends on the feature-set id
`viewer`	No
`guest`	No

- **Command payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	0 or the number of feature-set identifiers provided
`4`	`UInt32[N]`	Empty or array of feature-set identifiers

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	The number of feature-set identifiers successfully stored
`4`	`UInt32[N]`	Array of feature-set identifiers

7.7 I/O control

Setting the working mode of inputs and outputs is managed via feature-sets **inp0**, **inp1**, **out0** and **out1**.

To set the value of an output or to read the status of an input, we can use the two commands described in this paragraph.

7.7.1. QRYI

This command returns the status of the sensor's inputs.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	No
`guest`	No

- **Command payload contents**

Payload is empty.

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	Number of available inputs (N)
`4`	`UInt32[N]`	Input status `0` - The input is inactive `1` - The input is active

7.7.2. QRYO

This command returns the actual status of the sensor's outputs.

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	No
`guest`	No

- **Command payload contents**

Payload is empty.

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	Number of available outputs (N)
`4`	`UInt32[N]`	Output status `0` - The output is set to inactive `1` - The output is set to active

7.7.3. SETO

This command sets the status of the sensor's outputs.

Roles access

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	No
`guest`	No

Command payload contents

Ofs	Type or Length	Description
`0`	`UInt32`	Bit-mask of outputs to change
`4`	`Uint32[]`	Status of the outputs requested to change `0` - The output must be set to inactive `1` - The output must be set to active

The list of output statuses must be streamed in the same order of the bits, from the lowest to the highest.

Response payload contents

Ofs	Type or Length	Description
`0`	`UInt32`	Number of available outputs (N)
`4`	`UInt32[N]`	Output status `0` - The output must be set to inactive `1` - The output must be set to active

7.8 Informational commands

7.8.1. GVER

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	Yes
`guest`	Yes

- **Command payload contents**

Command payload is empty.

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`String`	A string representing the version

Example of a **GVER** string:

```
SLP250
[Firmware; 1.0.00.00; 2021/11/24]
Copyright (c) 2008-2021 Triple-IN GmbH - All rights reserved
```

7.8.2. INFO

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	Yes
`guest`	Yes

- **Command payload contents**

Ofs	Type or Length	Description
`0`	`UInt32`	Information type. See next table

- **Information types**

Info type	Description
`0`	Application state, scanning mode and diagnostic words
`10000`	System serial numbers, versions and basic features
`10001`	Dynamic diagnostic
`12321`	Internal power information
`12327`	Special dynamically updated values

- **Response payload type 0 contents**

Ofs	Type or Length	Description
0	UInt32	The sensor state values. Other numbers are reserved: 1 - Initialising 2 - Idle 3 - Scanning 4 - Switching 6 - Failure
4	UInt32	The scan mode, as described in **QRYM**, **SETM** commands
8	UInt32	Status bits(1)
12	UInt32	Warning bits(1)
16	UInt32	Error bits(1)

(1) See Appendix B

- **Response payload type 10000 contents**

Ofs	Type or Length	Description
0	UInt32	Info type: 10000
4	UInt32	Sensor model
8	UInt32	Sensor identifier
12	UInt32	Sensor firmware version
16	String	Sensor model name
	String	Manufacturing information
	UInt64	Processing unit serial number
	UInt64	Power unit serial number
	UInt64	Receiver module serial number
	UInt64	Laser module serial number
	Angle	Field of view starting direction
	Angle	Field of view ending direction
	UInt32	Minimum Pulse Repetition Frequency
	UInt32	Maximum Pulse Repetition Frequency
	String	Internal sensor information

Response payload type 10001 contents

Ofs	Type or Length	Description
0	UInt32	Info type: 10001
4	UInt32	Sensor identifier
8	UInt32	Status bits(1)
12	UInt32	Warning bits(1)
16	UInt32	Error bits(1)
20	Uint64[4]	Global bits(2)
52	String	Status bits description
	String	Warning bits description
	String	Error bits description
	String	Global bits description
	Int32	Internal temperature
	Int32	Laser temperature
	Int32	KEM Core temperature
	Int32	Internal pressure

(1) See Appendix B

(2) See Appendix C

- **Response payload type 12321 contents**

Ofs	Type or Length	Description
0	UInt32	Info type: 12321
4	UInt16	Power unit firmware version
6	UInt16	Processing unit 12V voltage
8	UInt16	Processing unit 5V voltage
10	UInt16	Processing unit 3v3 voltage
12	UInt16	Processing unit 1v8 voltage
14	UInt16	Reserved
16	UInt16	Reserved
18	UInt16	Reserved
20	UInt16	Processing unit overall current consumption
22	UInt16	Processing unit 5V current consumption
24	UInt16	Motor unit current consumption
26	UInt16	Power unit 5V voltage
28	UInt16	Power unit 24V voltage
30	UInt16	Power unit 24V line current consumption
32	UInt32	Build info

- **Response payload type 12347 contents**

Ofs	Type or Length	Description
0	UInt32	Info type: 12321
4	Int32	External encoder counter
6	Int32	Reserved
8	Int32	Reserved

7.9 Error response

Whenever a command is transmitted, the sensor will reply with a response.

As described in Chapter 5, a data package can be of four different types.

The only one that can be used by a control computer is the Command type. A Command data package can receive only two types of replies: Response and Error.

In case of Response, the command was executed successfully and the payload contains the expected data for the command identifier.

In case of Error, the command execution encountered an error and an Error data package type is returned, with the same command identifier as the Command and the payload described in the next paragraph.

In case the command data package cannot be validated or some error is found, an EREP response is returned.

7.9.1. EREP

This is a response-only type of payload. It can be returned by the sensor when it is not possible to recognise the data received or its validity cannot be ensured (Eg: Invalid CRC).

- **Roles access**

Role	Access
`admin`	Yes
`developer`	Yes
`operator`	Yes
`viewer`	Yes
`guest`	Yes

- **Response payload contents**

Ofs	Type or Length	Description
`0`	`Int32`	The error code(1)
`4`	`Buffer`	A string representing the version

7.10 Scan data

• LDTA

The LDTA event is generated when the sensor is working in 2D scan mode and a scan profile is completed.

The LDTA payload is divided into three sections.

Ofs	Type or Length	Description
`0`	`Header (128)`	The header, with scan attributes and sensor statuses
`128`	`Format (32)`	The description of the pulses data format
`160`	`Pulses`	The measured pulses, composed by one or multiple echoes

The `Header` and the `Format` structures have a size defined by the first `Uint32` field. The size value is expressed in bytes and includes the size field itself.

In this document are described the structures at the time of release.

To be able to correctly process the data, always use the content of the length field to parse the data, skipping the extra fields if necessary or not using the unknown ones.

Header

Ofs	Type or Length	Description
0	UInt32	Header size, including this field (currently 128)
4	UInt32	The version of this payload. Currently 0
8	UInt32	Status bits
12	UInt32	Warning bits
16	UInt32	Error bits
20	UInt32	Scan number
24	Timestamp[2]	Time stamp on first and last scan pulse in microseconds
40	Temperature	Internal temperature on package sending
42	Temperature	Laser temperature on package sending
44	Int32[2]	WAS coder on first and last scan pulse
52	Int32[2]	Reserved
60	Int32[2]	External coder on first and last scan pulse
68	UInt16[2]	Outputs state on first and last scan pulse
72	UInt16[2]	Inputs state on first and last scan pulse
76	UInt8	Mirror index
77	UInt8	Scan line index
78	UInt8	Channel index
79	UInt8	Channel flags
80	Angle[2]	External angle on first and last scan pulse
88	Angle[2]	WAS angle on first and last pulse
96	Temperature	Reserved
98	UInt16	Reserved
100	Int32	Internal pressure on packet sending
104	UInt64[3]	Reserved

Data format descriptor

Ofs	Type or Length	Description
`0`	`UInt32`	Header size, including this field (currently 32)
`4`	`UInt32`	The version of this payload. Currently 0
`8`	`Angle`	WAS angle on first scan pulse
`12`	`Angle`	WAS angle step
`16`	`UInt32`	Number of pulses
`20`	`UInt32`	First pulse index
`24`	`UInt8`	Number of echoes
`25`	`UInt8`	Echo format
`26`	`UInt8`	Echo size
`27`	`UInt8`	Echo type
`28`	`UInt8`	Range factor
`29`	`UInt8`	Packet info
`30`	`UInt8`	Pulse header size(1)
`31`	`UInt8`	Reserved

Pulses data

Every pulse is composed by an header (that can be empty, depending on the format) and the list of echoes.

Pulse size

Calculating the size occupied by a single pulse is easy and can be done with the following formula:

Firmware < 5.3:

`Pulse size = Number of echoes * Echo size`

Firmware >= 5.3:

`Pulse size = Pulse header size + (Number of echoes * Echo size)`

(1) Since firmware version `5.3`.
Do not use this field in older firmware version, is content is undefined!

Echoes data

The content of the echoes data depends on the echo format set by the user. The available formats are described in the following tables.

Echo format 3

Pulse header

Empty.

Echo data

Ofs	Type or Length	Description
0	3 (UInt)	Distance on three bytes only
3	UInt8	Signal or reflectivity

Echo format 4

Pulse header

Empty.

Echo data

Ofs	Type or Length	Description
0	UInt32	Distance

Echo format 6

Pulse header

Empty

- **Echo data**

Ofs	Type or Length	Description
0	UInt32	Distance
4	UInt8	Echo number
5	UInt8	Signal or reflectivity

Echo format 8

- **Pulse header**

Empty

- **Echo data**

Ofs	Type or Length	Description
0	UInt32	Distance
4	UInt32	Pulse width

Echo format 9

For space optimisation, this format uses octet sub-parts to store some data. The offset inside the octet is defined after a colon ":". Offset 0 identifies the least significant bit in the octet.

- **Pulse header**

Empty

Echo data

Ofs	Type or Length	Description
`0:0`	`UInt32`	Distance
`4:0`	`UInt(20 bits)`	Pulse width
`6:4`	`UInt(4 bits)`	Echo number
`7:0`	`UInt8`	Signal or reflectivity

Pulses and polar coordinates

In all the previous format, every pulse defines the echoes resulting from a laser trigger. The pulse data is then defined by the collection of the echoes packets and the polar angle of every single pulse can be calculated using the data in the Header and Format structures:

$$\text{Direction} = \Theta_0 + (i \times \Delta\Theta)$$

Where

- Θ_0 = Format.WAS angle on first scan pulse
- $\Delta\Theta$ = Format.WAS angle step
- i = Pulse index (0 based) inside the packet

Echo format 110[1]

The inclusive Polar format provides the direction information on every pulse, using the geometry described in Chapter 3.

Pulse header

Ofs	Type or Length	Description
`0`	`Int32`	Theta
`4`	`Int32`	Phi

[1] Since firmware version 5.3

- **Echo data**

Ofs	Type or Length	Description
0	UInt32	Distance
4:0	UInt(20 bits)	Pulse width
6:4	UInt(4 bits)	Echo number
7:0	UInt8	Signal or reflectivity

Echo format 111(1)

The inclusive XYZ coordinates format provides the coordinates of every single echo, using the geometry described in Chapter 3.

- **Pulse header**

Empty

- **Echo data**

Ofs	Type or Length	Description
0	Int32	X coordinate (1/10th of a millimeter)
4	Int32	Y coordinate (1/10th of a millimeter)
8	Int32	Z coordinate (1/10th of a millimeter)
12:0	UInt(20 bits)	Pulse width
14:4	UInt(4 bits)	Echo number
15:0	UInt8	Signal or reflectivity

(1) Since firmware version 5.3

8. Feature-sets

The following is a list of the available feature-sets.

Feature-set	Hex	Description
1pps	0×73707031	1PPS feature settings
asos	0×736f7361	Auto-start on power-up
dfs0	0×30736664	2D scan data format
ecod	0×646f6365	External incremental encoder settings
fltr	0×72746c66	Echo filters
inp0	0×30706e69	Properties and functions of the input 0
inp1	0×31706e69	Properties and functions of the input 1
out0	0×3074756f	Properties and functions of the output 0
out1	0×3174756f	Properties and functions of the output 1
netw	0×7774656e	Network configuration
s2d0	0×30643273	2D scan configuration
sup0	0×30707573	Start up settings

8.1 1pps - 1PPS feature settings

Sets the various attributes related to the 1PPS feature.

- **Roles access**

Role	Read	Write
admin	Yes	Yes
developer	Yes	Yes
operator	Yes	No
viewer	No	No
guest	No	No

- **Attributes**

Bit	Type	Name	Description
0	`UInt32`	Mode	Working mode: 0. 1PPS is disabled 1. NMEA sentence before 1PPS signal 2. NMEA sentence after 1PPS signal
1	`UInt32`	UART speed	The serial port bauds (default to 9600)
2	`UInt32`	Timeout	Timeout before rising a warning on missing 1PPS signal

8.2 asos - Auto-start on power-up

Sets the destination machine which will receive a UDP scan data stream automatically after power-up.

The feature can be enabled or disabled.

- **Roles access**

Role	Read	Write
`admin`	Yes	Yes
`developer`	Yes	Yes
`operator`	Yes	No
`viewer`	No	No
`guest`	No	No

- **Attributes**

Bit	Type	Name	Description
0	`String`	IP address	Destination IP address
1	`UInt32`	Port	Destination port
2	`UInt32`	Flags	Bit-field with flags

- **IP address**

  Must contain an IPv4 address of a machine receiving the scan data on power-up.

- **Port**

  The UDP/IP port on the receiving service. The port must be greater than 1023 and lower than 65536.

- **Flags**

Bit	Description
`0`	Status of the feature: `0` - Feature is disabled `1` - Feature is enabled

8.3 dfs0 - 2D scan data format

Defines the scan data format to be used in all the scan data streams.

- **Roles access**

Role	Read	Write
`admin`	Yes	Yes
`developer`	Yes	Yes
`operator`	Yes	No
`viewer`	No	No
`guest`	No	No

- **Attributes**

Bit	Type	Name	Description
0	UInt32	Echo format	The desired echo format. See LDTA event description (Paragraph 6.9.1) for the list of available formats
1	UInt32	Number of echoes	Number of echoes for every pulse(1)
2	UInt32	Echoes priority	Defines priority of first or last echoes in case **Number of echoes** is less than 3 0 = Last echo, then first 1 = First echo, then last

8.4 ecod - External incremental encoder settings

Defines the configuration of the optional external incremental encoder.

- **Roles access**

Role	Read	Write
admin	Yes	Yes
developer	Yes	Yes
operator	Yes	No
viewer	No	No
guest	No	No

(1) The maximum number of echoes are depending on the sensor model.

- **Attributes**

Bit	Type	Name	Description
0	UInt32	Enabled	0 = Disabled, 1 = Enabled
1	UInt32	Index logic	0 = Rising edge, 1 = Falling edge, 2 = Disabled
2	UInt32	Signals logic	0 = Active high, 1 = Active low
3	UInt32	Signals order	0 = A → B, 1 = B → A
4	Int32	Offset	An offset to be added to the internal counter
5	UInt32	Reset to zero	When set to 1, internal counter is set to 0

8.5 fltr - Echo filters

- **Roles access**

Role	Read	Write
admin	Yes	Yes
developer	Yes	Yes
operator	Yes	No
viewer	No	No
guest	No	No

- **Attributes**

Bit	Type	Name	Description
0	`UInt32`	Minimum pulse-width	When set, every echo with a pulse-width value below this number will be dropped. Set to `0` to disable
1	`UInt32`	Maximum pulse-width	When set, every echo with a pulse-width value above this number will be dropped. Set to `0` to disable
2	`UInt32`	Minimum range	When set, every echo with a range value below this number will be dropped. Set to `0` to disable
3	`UInt32`	Maximum range	When set, every echo with a range value above this number will be dropped. Set to `0` to disable
4	`UInt32`	Pulse-width minimum limit	Read-only. The minimum value that is possible to set for the Min. pulse-width
5	`UInt32`	Pulse-width maximum limit	Read-only. The maximum value that is possible to set for the Max. pulse-width
6	`UInt32`	Range minimum limit	Read-only. The minimum value that is possible to set for the Minimum range
7	`UInt32`	Range maximum limit	Read-only. The maximum value that is possible to set for the Maximum range

8.6 inp0 - Input 0 configuration

- **Roles access**

Role	Read	Write
`admin`	Yes	Yes
`developer`	Yes	Yes
`operator`	Yes	No
`viewer`	No	No
`guest`	No	No

- **Attributes**

Bit	Type	Name	Description
0	UInt32	Input mode	The input working mode. See below
1	UInt32	Signal logic	0 = Active high, 1 = Active low

- **Input mode**

The input mode defines how the signal comping to the input is handled by the sensor.

0. **Disabled**. The input signal is ignored
1. **Immediate**. The input signal momentary value is reported
2. **Detect active**. The input signal active value is detected and stored until the next request time. The request times are when the first beam of the scan and the last beam of the scan are triggered
3. **Detect inactive**. The input signal inactive value is detected and stored until the next request time. The request times are when the first beam of the scan and the last beam of the scan are triggered
4. **1PPS**. The input is the recipient of the 1PPS signal. The 1PPS feature must be enabled separately with the `1pps` feature-set

8.7 inp1 - Input 1 configuration

The same description as for the `inp0` applies.

Then the 1PPS feature is enabled in both inputs, the settings of `inp0` will take precedence and the input 1 will be disabled.

8.8 out0 - Output 0 configuration

- **Roles access**

Role	Read	Write
`admin`	Yes	Yes
`developer`	Yes	Yes
`operator`	Yes	No
`viewer`	No	No
`guest`	No	No

- **Attributes**

Bit	Type	Name	Description
`0`	`UInt32`	Output mode	The output working mode. See below
`1`	`UInt32`	Signal logic	`0` = Active high, `1` = Active low

- **Output mode**

The output mode defines how the signal comping to the input is handled by the sensor.

0. **Disabled**. The output value is ignored, the output is set to inactive
1. **Immediate**. The output value is set

8.9 out 1 - Output 1 configuration

The same description as for the `out0` applies.

8.10 netw - Network configuration

Allows to modify the sensor network configuration.

This feature-set's values must be changed with the utmost care! A mistake using this feature can cause the sensor to be unreachable from the outside.

- **Roles access**

Role	Read	Write
`admin`	Yes	Yes
`developer`	Yes	Yes
`operator`	Yes	No
`viewer`	No	No
`guest`	No	No

- **Attributes**

Bit	Type	Name	Description
0	`UInt32`	DHCP	`0` = Disabled, `1` = Enabled(1)
1	`String`	IP address	The IPv4 address of the sensor(2)
2	`String`	Network prefix	The `/prefix` for the network
3	`String`	Default gateway	The default gateway
4	`String`	DNS server	The DNS server to be used if necessary

8.11 s2d0 - 2D scan configuration

Changes 2D scan properties to adapt the scan to the application needs. With this feature-set, the scan profile resolution, the field of view and, when available, the sensor's pulse repetition frequency can be.

(1) When DHCP is enabled, all the other values are ignored

(2) It is recommended to avoid the 10.255.0.0/16 network. This is used as a fallback network to restore from wrong setups

- ## Roles access

Role	Read	Write
`admin`	Yes	Yes
`developer`	Yes	Yes
`operator`	Yes	No
`viewer`	No	No
`guest`	No	No

- ## Attributes

Bit	Type	Name	Description
0	`UInt32`	Scan mode	The desired scan mode (resolution)
1	`UInt32`	Pulse frequency	Pulse repetition frequency (when available)
2	`Int32`	Start angle	The starting angle of the field of view
3	`Int32`	Stop angle	The ending angle of the field of view

- ### Scan mode

Defines the desired scan resolution. Depending on the sensor model, several modes can be available:

Mode	Name	SLP	SLP PRO
1	Normal	0.09°, 1000 pts, 30 Hz	0.09°, 1333 pts, 50 Hz
2	Fast	0.18°, 500 pts, 60 Hz	0.18°, 666 pts, 100 Hz
3	Interlaced	0.0225°, 4000 pts, 7.5 Hz	0.018, 6666 pts, 10Hz
4	Fine	0.045°, 2000 pts, 15 Hz	0.036°, 3332 pts, 20 Hz
5	Normal+	n.a.	0.072°, 1666 pts, 40 Hz
6	Fast+	n.a.	0.144°, 833 pts, 80 Hz

8.12 sup0 - Start up settings

- **Roles access**

Role	Read	Write
admin	Yes	Yes
developer	Yes	Yes
operator	Yes	No
viewer	No	No
guest	No	No

- **Attributes**

Bit	Type	Name	Description
0	UInt32	Sensor scan mode	The desired state for the sensor after power-up: 0 - Idle mode. Motor and scan are stopped 2 - 2D scan mode

9. Sensor services announcement

Most of Triple-IN's sensors can send a message containing information about the configuration of the sensor itself. This is useful to discover the IP addresses of one or more sensors connected to a network.

9.1 SVCS message

To receive this informational packet, it is necessary to broadcast a specific packet to the sensors network on a specific port.

This packet does not follow the communication protocol data packet described in the previous chapters, but it simply consists in a packet of 4 octets:

Command	Hex	Octets
SVCS	0x53435653	0x53 0x56 0x43 0x53

This message can be sent directly to a single sensor or broadcast on an entire subnet, enclosed in a UDP datagram and directed to the port 6996.

9.2 SVCS response

As the SVCS message, the SVCS response does not follow the communication protocol data packet described in the previous chapters.

The response is enclosed in a UDP datagram and it is always sent as a broadcast message to the sensor's networks to the port 6996.

The SVCS response packet is structured as following:

Ofs	Length	Name	Description
0	32	Header	Basic information about the package
32	124	Addresses	List of IP addresses available on the sensor
156	868	Services	List of services provided by the sensor

- **Header**

Ofs	Type	Name	Description
0	UInt32	Packet tag	Contains 0×53435653 ("SVCS")
4	UInt32	Size	The size of this packet, including the header
8	UInt32	Message	The message type
12	UInt32	Version	Version of this packet
16	UInt32	Sensor identifier	The sensor identifier (serial number)
20	UInt32	Reserved	
24	UInt32	Sensor model	The sensor model number
28	UInt32	Reserved	

- **Message**

The type of the message. As of yet, only type 1 is implemented.

- **Version**

Two versions of the package are defined:

- 0×00010000: It is the version provided by Triple-IN's Psxxx-90+ sensors
- 0×00010001: It is the version provided by Triple-IN's SLP sensors

The only difference between the two versions is in the size of the Services array (see in the following paragraphs). The number provided by the PS sensors is 30, while the SLP sensors provide 36.

To correctly parse the messages coming from both sensors, it is enough to always check the written size and the count of services, instead of relying on the maximum number.

- **Addresses**

Ofs	Type	Name	Description
0	UInt32	Valid addresses	The number of valid addresses in the following list
4	Address[10]	Address record	Address entry

- **Address**

Ofs	Type	Name	Description
0	UInt32	Address type	The address type
4	UInt32	IP address	IP address in network byte order
8	UInt32	Network mask	Network mask in network byte order

- **Address type**

Can be one of:

Value	Description
0x00000000	The address type is unknown or invalid. Denotes an invalid entry
0x00000001	Address is an IPv4 address
0x80000000	Address is the "catch all" (`0.0.0.0` for IPv4). This is used in the list of services (see below) to identify a service listening to all addresses

- **Services**

Ofs	Type	Name	Description
0	UInt32	Valid services	The number of valid services in the following list
4	Service[36]	Address record	Service entry

- **Service**

Ofs	Type	Name	Description
0	UInt32	Service type	The service type
4	Address	Address	The address the service is listening to
16	UInt32	Port	The service port
20	UInt32	Flags	Service flags

- **Service type**

The service type identifies the data exchange format or the actual service provided by the entry.

It can be one of the following:

Value	Description
0	Invalid value. It means the service entry must not be used
1	Update service
2	Triple-IN's PS sensors protocol service
4	Triple-IN's Rotary Table sensor protocol service
11,12,13,14	Triple-IN's SLP protocol service (described in this document)
9998	The sensor service announcement

A. Glossary

- **Auto-start**

Enables the sensor to provide measurement results without requiring a start command from the controlling computer.

- **Digital inputs**

Discrete inputs used to send signals with various functions to the sensor. For example, the inputs can be used as PPS input.

- **Digital outputs**

Triple-IN's sensors provide programmable digital switching outputs with various functions. These outputs can be used for synchronisation signals or for special features.

- **Echo**

An echo identifies a single return from a target hit by a laser beam.

- **Encoder index**

Physical angle position of the encoder zero index.

- **External incremental encoder**

An encoder connected with a sensor to provide an external angular position. This is typically a horizontal rotation angle, but may be used as different purposes.

- **Feature-set**

  A collection of attributes used to configure a specific behaviour of a sensor.

- **Field of view (FOV)**

  Identifies the angular extension inside which the pulses are distributed.

  Can be identified by a single angle value (eg: 90°) or by a starting and a stopping direction (eg: 45° to 135°).

- **First echo**

  Signal detected as a return from the first target hit by the laser beam.

- **Focusing distance**

  Distance where transmitter, receiver and optics are focused on.

- **Interlacing**

  Interlacing is a common method of incrementally encode an image starting from lower resolution.

  Some models of Triple-IN's sensors use interlacing in some scan modes to improve the angle resolution.

- **KEM**

  Acronym for "Kontinuierliche Event Messung".

  Means "Continuous Event Measurement", Triple-IN's patented core technology to measure subsequent signal events with a high time resolution.

- **Laser beam**

  The ideal straight line traveled by the laser pulse.

  A single laser beam can cause multiple echoes in return.

- **Last echo**

Signal detected as a return from the last target hit by the laser beam.

Due to the limitations of the measuring sensors, which can evaluate and process a limited number of echoes, the **Last echo** is actually the last echo detected and processed. Therefore it is not necessarily the echo returned by the last physical target hit by the laser beam.

- **Mirror motor unit**

In a 2D sensor, it identifies the assembly including the motor, incremental encoder and reflective polygon, which, with its continuous rotation, builds the scan profile.

- **Mirror incremental encoder**

The angle encoder which provides the current position of the mirror and thus the current measurement direction for the leaser beams.

- **Multi echo**

The event measurement of many echo pulses from distributed or diffuse targets.

- **Operating temperature**

Temperature range where sensors work within their specification.

- **PoE - Power over Ethernet**

Power over Ethernet (PoE) injectors pass electrical power along with data on Ethernet cabling. A single cable is used to provide both data connection and electrical power to the sensors.

Some Triple-IN's sensors can use PoE as the source of power. Please refer to the sensor's Operating Manual of the single sensor for more information.

- ## PPS signal

A pulse per second (PPS) is an electrical time signal.

PPS signals are used for precise timekeeping and time synchronisation with other sensors. The PPS signal specifies the start of a second. The sensor combines the PPS functionality with the internal time stamp of a measurement and with an external time source.

- ## PRF - Pulse Repetition Frequency

The number of laser pulses emitted per second.

- ## Profile

A set of measurements formed by one or more scans.

- ## PSMU

Power Subsystem Management Unit.

Manages the powering of the various internal sensors, the heating system and monitors voltages and consumption.

- ## Pulse

A change in the amplitude of an electrical signal from a baseline value to a higher value, followed by a return to the baseline value.

- ## Pulse width

Time between the rising edge and the falling edge of a pulse.

- **Scan**

  Sequence of ToF measurements. Contains the series of returned echoes from the laser beam triggered within the defined **Field of View** and **Scan resolution**.

- **Scan resolution**

  Small angle between two subsequent laser beams in a scan.

- **Scan angle**

  The angular field of view of the sensor. See also **Field of View**.

- **Scan start angle**

  The direction where the first laser beam of a scan is emitted.

  Defines the first angle direction of the **Field of View**.

- **Scan stop angle**

  The direction where the last laser beam of a scan is emitted.

  Defines the last angle direction of the **Field of View**.

- **Scan synchronisation signal**

  Signal on a digital output that is triggered by the scan process.

  An output is set with the first laser beam of a scan and cleared with the last laser beam of a scan.

- **Sigma**

  The standard deviation calculated from a number of samples. For the KEM technology, the reported sigma of repeated measurements gives the precision of a single measurement.

- ## Spot

Projection of the laser beam on a target's surface.

- ## Spot divergence

The angle of the spreading of the laser beam.

- ## Spot size

A measure of the spreading of the laser beam over distance.

- ## Standard deviation

The standard deviation is a measure of the dispersion of a set of values. The standard deviation is usually denoted with the Greek letter sigma. It is defined as the root-mean-square (RMS) deviation of the values from their mean.

See also: **Sigma**

- ## Standard error

The standard deviation of the sampling distribution of the sample statistic. The standard error of the mean (SEM) of a sample is calculated as the standard deviation of the sample divided by the square root of the sample size.

For the KEM technology, the reported standard error gives the precision of an averaged measurement result.

- ## Statistical error

The spreading of measurement results under constant ambient conditions.

- **Storage temperature**

  The safe storage temperature to store a sensor above which damages may occur.

- **Systematical error**

  The **Standard Error** with infinite number of measurement samples.

- **TOF - Time Of Flight**

  Identifies a range measurement method using the time elapsed from the source trigger to the target hit and back.

B. System diagnostic bits

- **Status bits**

Bit	Name	Description
2	1PPS active	The 1PPS feature has been activated
21	Internal calibration	An internal calibration table is disabled
22	Internal calibration	An internal calibration table is disabled
23	Internal calibration	An internal calibration table is disabled
24	Internal calibration	An internal calibration table is disabled
25	Internal calibration	An internal calibration table is disabled
26	Internal calibration	An internal calibration table is disabled
27	Internal calibration	An internal calibration table is disabled
30	Heater on	The internal heater is on
31	Busy	The system is busy performing an operation. Response and functionalities can be affected

Warning bits

Bit	Name	Description
0	GPIO	An issue is detected in the GPIO subsystem. Could be a temporary only failure
1	PSMU	The low level power management system detected an issue. Could be that some of the internal power limits are exceeded
2	1PPS	1PPS signal is out of period tolerances
3	Encoder	One of the managed incremental encoder reported an error
19	Internal calibration	An internal calibration table is missing
20	Internal calibration	An internal calibration table is missing
21	Internal calibration	An internal calibration table is missing
22	Update	A recent update procedure has failed
23	Internal calibration	An internal calibration table is missing
24	Internal calibration	An internal calibration table is missing
25	Internal calibration	An internal calibration table is missing
26	Internal calibration	An internal calibration table is missing
27	Internal calibration	An internal calibration table is missing
28	Environment limits	The working limits of temperature and pressure are exceeded
29	Power limits	The working limits of power voltage and current are exceeded
30	Restart needed	The sensor needs a restart. A power cycle must be performed
31	Firmware timeout	An internal timeout has occurred. A restart is necessary

Error bits

Bit	Name	Description
0	1PPS failure	An error was detected in the 1PPS management
1	Trigger frequency	The maximum laser trigger frequency was exceeded
18	SAS encoder	An error was detected in the secondary incremental encoder
19	External encoder	An error was detected in the External incremental encoder
20	Mirror encoder	An error was detected in the mirror incremental encoder
21	Hardware version	Hardware version inconsistency detected
22	Configuration	An internal configuration error was detected
23	Calibration	An internal calibration error was detected
24	Network	A generic network error was detected
25	I/O	An unrecoverable error in the I/O sub-system was detected
26	KEM failure	Issue detected in the KEM measurement sub-system
27	Storage	An internal storage failure has occurred
28	Environment limits	The safety limits of temperature and pressure are exceeded
29	Power limits	The safety limits of power voltage and current are exceeded
30	Hardware error	A generic hardware issue was detected
31	PSMU system	The PSMU system is in error state or not reachable

C. Error codes

Error codes are usually represented by a negative integral number contained in a `Int32` type.

Code	Description
-2000	Physical device I/O error
-2001	I/O read error
-2002	I/O write error
-2003	Timeout expired
-2004	User break
-2005	CRC checksum error
-2006	Unknown command
-2007	Attribute is out of range
-2008	Access denied / Permission denied
-2009	Unsupported function
-2010	Invalid handle / Bad address
-2011	Division by zero
-2012	Array index is out of bounds
-2013	Internal buffer overflow
-2014	Fatal system error
-2015	System configuration error
-2016	Error in serialisation
-2017	KEM unit error
-2018	Motor unit failure
-2019	Temperature out of operating range
-2020	Front screen not clear
-2021	System not ready
-2022	Empty buffer
-2023	*Reserved*

About Triple-IN

Company Vision

Triple-IN's main efforts are focused in the development and commercialisation of powerful 2D and 3D laser distance sensors and systems based on its patented KEM technology. The KEM method is the pillar of the idea to transfer the performances of complex and expensive measurement systems in low-cost sensors for indoor and outdoor automation applications. The strong willing for innovation constantly pushes Triple-IN to find new solutions in optics, array and scanning technologies, also joining forces with leading universities, research facilities or high technology partners, working closely sharing experience and knowledge.

Triple-IN's Products

Continuous developments and innovations have led Triple-IN to the realisation of several outstanding products currently operating in the most various environments. From the harshest open pit mine to the biggest airport, the KEM technology is supporting partners to run their applications and offer customised and affordable solutions to their clients. The Pulse Sensors (PS), the Pulse Array Camera sensors (PAC) and the Volume Measurement System (VMS) are only some of the successful Triple-IN products, always at the top of the list of reliability, precision and convenience.

Triple-IN
LASER TECHNIC

Poppenbütteler Bogen 64
D-22399 Hamburg
Germany
Tel: +49 (0) 40 500 91998

info@triple-in.com
www.triple-in.com