2D LASER SCANNER
LMS-Q240(i)

General Description and Data Interfaces

RIEGL
LASER MEASUREMENT SYSTEMS
www.riegl.com

2D LASER SCANNER
LMS-Q240(i)

General Description and Data Interfaces

RIEGL Laser Measurement Systems GmbH
A-3580 Horn, Riedenburgstrasse 48, AUSTRIA
Tel.: +43-2982-4211, Fax.: +43-2982-4210
e-mail: office@riegl.co.at
www.riegl.com

Version 10/2006 CE
Rev. 19-11-2007 (V74.4)

Revision History:

Rev. 07-05-2007 (V73.4)
Rev. 18-01-2007 (V73.4)
Rev. 04-10-2006 (V70)

Contents

1 General

1.1 Special Terms Used in this Manual

The following part lists special terms and definitions used within this manual. If you are using a *RIEGL* scanner for the first time, we recommend to have at least a short looking through to get familiar with it.

Within the manual, these terms are written *italic* to indicate that a reference definition / explanation can be found in this chapter.

Amplitude: see Signal Intensity, below
Angle Encoder: An opto-mechanical device for measurement of angles. For *RIEGL* scanners, the angle encoders are directly coupled to line axis and frame axis. Angle encoders used in *RIEGL* scanners use a reference mark signal, which must be detected after powering up the system to provide absolutely scaled angle data.
Baud Rate: Transmission speed in bits per second. For RS232/RS422 ports, baud rates up to 115200 bits per second are supported. The baud rate can be configured by the user.
Beam Aperture: The beam diameter directly at the transmitter output lens.
Beam Divergence: Beam divergence θ equals the (full) cone angle of the laser beam at large distances from the instrument. The beam diameter d at a target at range R can be estimated as $d = (d_a^2 + R^2.\theta^2)^{0.5}$, where d_a denotes the beam aperture.
Beam Focus: For *RIEGL* instruments it denotes the distance from center of transmitter optics to the point where laser beam has smallest beam waist diameter. As most *RIEGL* instruments have a collimated beam, the beam focus is infinite.
Control Port: *RIEGL* scanners use 2 ports for data communication: 1.) a control port for setting/reading scan definition parameters and for scan control and 2.) a (fast) data port for transmission of scan result data. The control port is either the RS232 port of the scanner (local control) or a port of the LAN – TCP/IP interface (LAN control).
CRC: Cyclic Redundancy Check is a method for calculation of a checksum for data packages to recognize errors in data transmission.
Data Port: *RIEGL* scanners use 2 ports for data communication: 1.) a control port for setting/reading scan definition parameters and for scan control and 2.) a (fast) data port for transmission of scan result data. The data port is either the ECP parallel port of the scanner (local control) or a port of the LAN – TCP/IP interface (LAN control).

Diffusely Reflecting Target: Walls, bushes, trees are typical examples for diffusely reflecting targets. In fact all targets which do not behave like a mirror or a retro-reflector for the laser beam are denoted as diffusely reflecting targets. Diffusely reflecting targets scatter the incoming laser beam back into “all directions”. Please also refer to Retro-Reflective Target, below

ECP Parallel Port: Scanner parallel data output port to be connected to the parallel printer port (LPT) of a PC. The PC parallel port must be set to ECP mode in PC BIOS setup to be operated as data input on PC side. The ECP parallel port is used as data port in mode local control.

Encoder Reference Mark: see Angle Encoder, above

End Switch: An optoelectronical element used in *RIEGL* scanners for detection of angle range limits. The scanner electronics guarantee that the scan motions keeps within the end switch limits.

First Target: see Target Mode, below

LAN Control: LAN control means, that a scanner LAN-TCP/IP port (usually 20002) is used as control port and a second scanner LAN-TCP/IP port (usually 20001) is used as data port. Operating mode local control or LAN control is automatically selected at scanner power up. If LAN control ports are active, the local control ports are automatically inactive.

LAN - TCP/IP Interface: Scanner interface to be connected to a local area network (LAN), using TCP/IP protocol. Two ports, a control port and a data port, are supported for scanner control and for scan result data transmission.

LAN Link: A status information indicating that the scanner is connected to a LAN. LAN link detected does not indicate whether a connection is established. If at power up the instrument detects a LAN link, it automatically switches to LAN control mode.

Laser Class: Each laser device has to be assigned to one of four laser classes. The laser class informs the user on the potential hazard in operating the device. Laser class 1 instruments do not cause any hazard.

Laser Mirror Scanner: A device combining a laser distance meter module with a scanning mechanism, which is deflecting the measurement laser beam by a rotating or oscillating mirror.

Last Target: see Target Mode, below

Line Angle: Angle from the positive z-axis of the instrument to the beam axis of the laser beam.

Line angle Encoder: see Angle Encoder, above

Line Angle Step Width: Angular line angle distance between consecutive laser distance measurements within one scan line.

Line Down Scan: Line scan with decreasing line angle values.

Line Scan: With 2D scanners the scan pattern consists simply of a single scan line. With 3D scanners, a scan pattern consists of a number of scan lines forming a complete scan frame. The scan mechanism providing a scan line is denoted as the line scan.

Line Scan Data: Denotes the data acquired during the scan of a line. For short, these data are sometimes also denoted as scan line.
Line Scan Range: The line scan range denotes the actual angular range of a single line scan. The line scan range can be configured by stating the start angle of the line scan, the line angle step width, and the number of measurements per scan line. The maximum line scan range is limited.
Line Up Scan: Line scan with increasing line angle values.
Local Control: Local control means, that the scanner RS232 port is used as control port and the ECP parallel port is used as data port. Operating mode local control or LAN control is automatically selected at power up. If local control ports are active, the LAN control ports are automatically inactive.
Measurement Quality: A value indicating the percentage of emitted laser shots, which resulted in usable echo signals. This value is relevant for measurements, which average several single laser shot results. Measurement quality 100 means that all laser shots emitted resulted in useable data. Single laser shots (no averaging) can have measurement quality 0 (no echo) or 100 (echo) only.
Measurement Mode: This is the operating mode opposite to programming mode, i.e. the mode in which measurements and thus scans are carried out. After power up, *RIEGL* instruments start with measurement mode. Please also refer to Programming Mode, below
Mirror Facet: Polygon mirror wheels used in *RIEGL* laser mirror scanners have a set (typically 3 or 4) of mirrors arranged around the wheel perimeter. According to the beam cross section size, only a part of the facet can be used to deflect the beam. Areas close to the facet edges cannot be used. This implies a measurement gap between consecutive mirror facets for rotating scans.
Oscillating Scan: Consecutive line scans are realized by an oscillating mirror or polygon mirror wheel, thus line scans are alternating between line up and line down scans. For LPM series instruments scans are always composed of line up and line down scans.
Parallel Port: see ECP Parallel Port, above
Polygon Mirror Wheel: Key component of the scan mechanism with flat reflective surfaces (mirror facets) arranged around the wheel perimeter used within the LMS series instruments. With rotating or oscillating movement of the polygon mirror wheel, the measurement laser beam is deflected and thus scanned. *RIEGL* laser mirror scanners use a rotating / oscillating polygon mirror wheel for the line scan.
Pre Laser Shots: Laser shots emitted prior to the measurements of each line scan.
Programming Mode: Operating mode to set and interrogate instrument parameters. The programming mode is started by sending ^P (Ctrl-P, ASCII 10hex) to the instrument and is quit by sending the sequence "Q" + <Cr>, where <Cr> means carriage return (ASCII 0Dhex).
PPS: Denotes the synchronization pulse provided by the GPS unit. Usually a TTL level pulse with 1 **p**ulse **p**er **s**econd.
PRR: Laser **P**ulse **R**epetition **R**ate. The frequency of emitted laser pulses.

Retro-Reflective Target (Retro Reflectors): The retro-reflected beam is returned in almost the direction of the incident beam. Thus, the echo signal of a retro-reflective target is much stronger compared to signals from diffusely reflecting targets. Retro-reflective targets can be measured up to much large distances compared to diffusely reflecting targets.

Rotating Scan: A line scan method used with polygon mirror wheels, where for consecutive scan lines the laser beam is deflected by consecutive mirror facets of the polygon mirror wheel.

RS232 Port: Scanner serial data input/output to be connected to the serial RS232 port (COM) of a PC. The RS232 port is used as control port in mode local control.

Scan Line: A sequence of consecutive distance and angle measurements, where the measuring laser beam is deflected by a (fast) moving line scan mechanism. *RIEGL* laser mirror scanners use rotating or oscillating mirrors to deflect the laser beam for measurements of a scan line. See also Line Scan data, above

Serial Port: see RS232 Port, above

Signal Intensity: Signal strength of the received laser echo signal. The signal strength is affected by the reflectivity of the target, the distance to the target and by the atmospheric visibility. Signal intensity data of *RIEGL* scanners are not scaled in physical units. This manual uses the terms amplitude and intensity synonymously for signal intensity.

SyncTimer: Denotes an internally generated time information which is supplied with data. According to the *RIEGL* scanner type, SyncTimer data can be added to each measurement of a scan line or only once for a scan line (in this case typically indicating the time of the first laser shot of the scan line). The SyncTimer can be reset to 0 by an external TTL trigger pulse (typically supplied by a GPS module) to synchronize the time information to an external event.

Target Mode: *RIEGL* scanners support detection of **first target** or **last target.** If the laser beam hits a target, but a part of the beam passes by and hits further target(s) behind, several echo signals of the first and further target(s) are detected. The target mode selects, which of several targets is to be provided in the range data. If only one target is detected (no parts of the beam can pass by), first and last target mode display the same target (first target is last target).

A typical multiple target situation occurs with measurement to bushes, trees or fences located in front of a other objects (e.g. walls).

TCP/IP: Transmission Control Protocol / Internet Protocol denotes standard protocols for internet data transmission.

1.2 System Configuration

The **Laser Mirror Scanner LMS-Q240(i)** is a 2D laser scanner based upon accurately measuring the distance by means of electro-optical pulsed time-of-flight range measurement and upon fast scanning the laser beam by means of an opto-mechanical scan mechanism. The high range performance, the fast line scanning, and the overall system design makes the LMS-Q240(i) well suited for airborne laser scanning applications.

The laser scanner LMS-Q240(i) consists mainly of two subsystems, an accurate laser rangefinder electronics and a line scanning mechanism, installed in a rugged housing.

1.2.1 Distance Meter Module

The rangefinder system is based upon the principle of time-of-flight measurement of short infrared laser pulses.

A laser source emits infrared light pulses, which are collimated by a transmitter lens system. Via the receiver lens, part of the echo signal reflected by the target hits a photodiode which generates an electrical receiver signal. The time interval between transmitted and received pulses is counted by means of a quartz-stabilized clock frequency. The measured time value is passed to the internal microcomputer which processes the measured data and prepares it for data output.

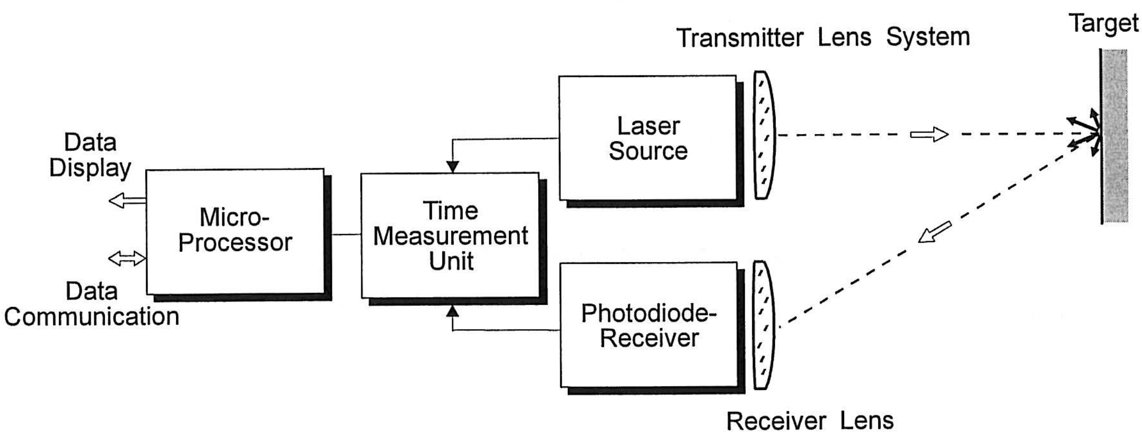


Fig. 1 Measurement principle of the pulsed distance meter

1.2.2 Scanner Mechanism

The scanner mechanism deflects the laser beam for range measurement into a precisely defined direction. Each *scan line* is composed of a number of pixels (single laser measurements).

The angular deflection of the laser beam is realized by a rotating *polygon mirror wheel (multifaceted mirror)*. The polygon-mirror is composed of flat reflective surfaces arranged around the wheel perimeter. The wheel rotates continuously at a fixed speed to provide repetitive unidirectional scans.

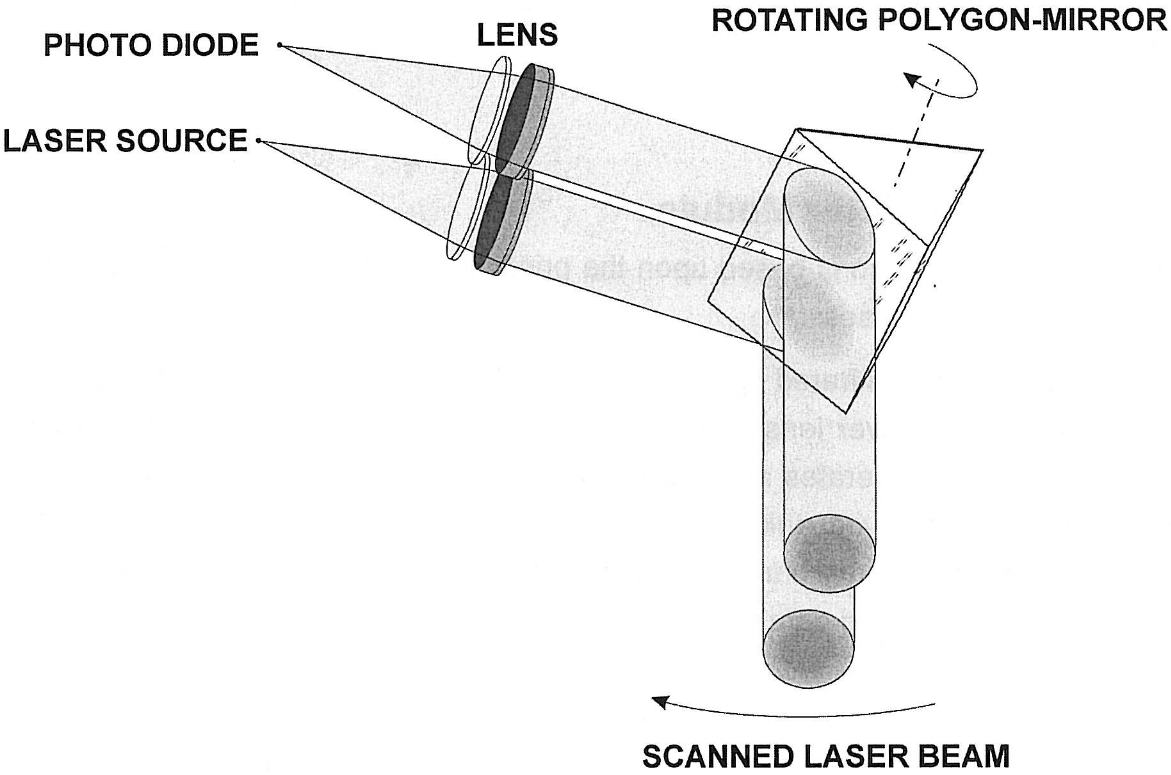


Fig. 2 Principle of beam deflection by a rotating mirror

Due to the finite aperture dimensions, only a fractional part of the polygon mirror surfaces (excluding the edge areas) can be used for scanning.

1.2.2.1 LMS-Q240(i) Timing Characteristic

As mentioned in chapter 1.2.2, only a part of the *mirror facets* can be used for data acquisition. At the edges of the facets the laser beam is split into two beams and no measurement is possible. The utilization of 30° out of 90° (60 degree angle range for the beam) for the Q240(i)-60 and 40° out of 120° (80 degree angle range for the beam) for the Q240(i)-80 results in a duty factor of 33.3 percent. That is the reason for gap times between two consecutive *scan lines*. Fig. 3 shows the timing situation for the LMS-Q240(i)-80.

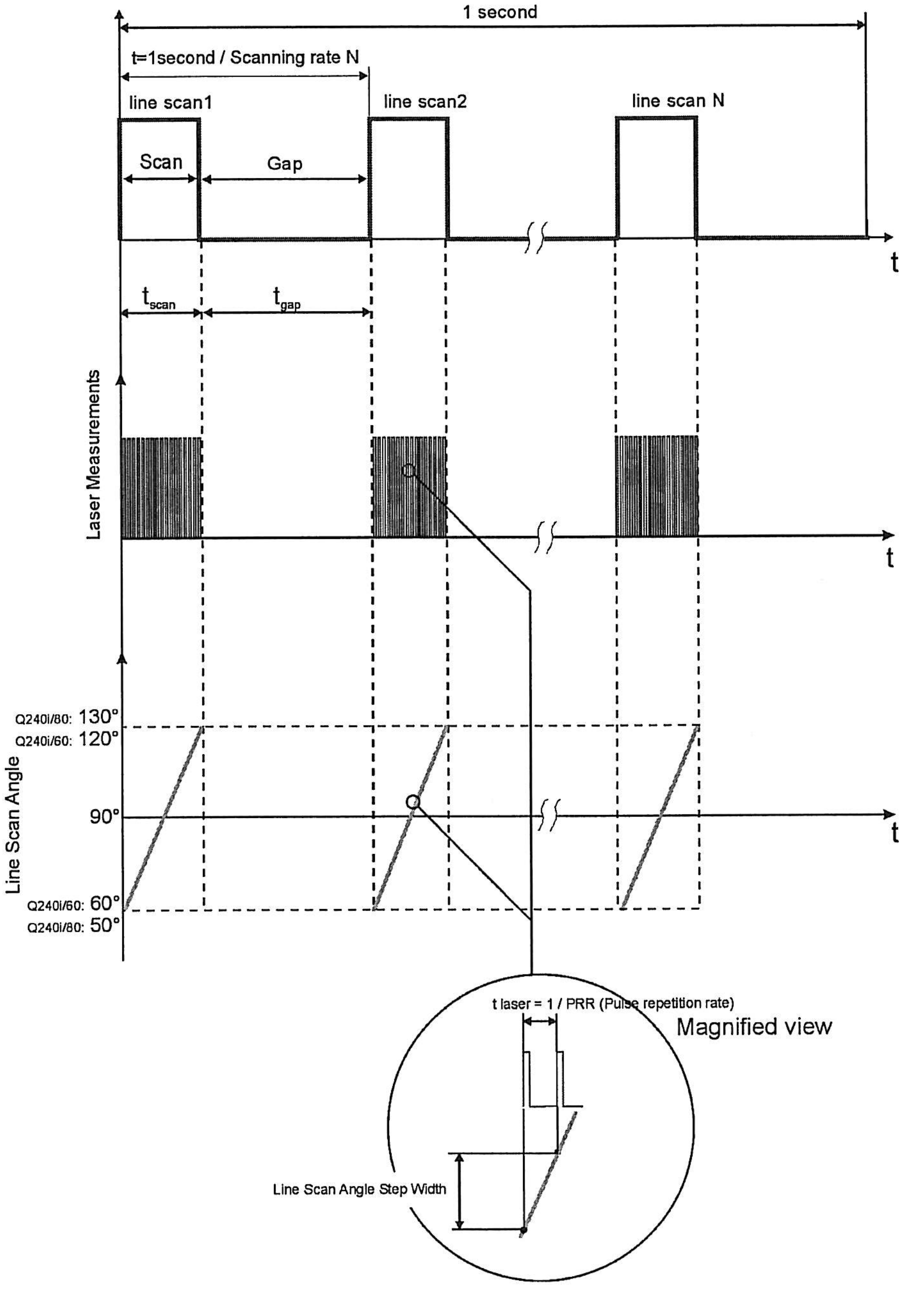


Fig. 3 Timing diagram for the scan mechanism

1.3 General System Set Up and Cabling

- Provide a suitable power supply for the laser scanner (please refer to technical manual part).
- Mount the laser scanner by means of the mounting threads (please refer to technical manual part).
- Connect the interface cables to the instrument. These are the
  - serial data cable + the parallel data cable for *local control* mode
  - the LAN interface cable for *LAN control* mode
  Please refer to technical manual part for cable details.
- Connect the instrument to the power supply using the power supply cable. Now switch on the power supply. The scanner starts scanning automatically.

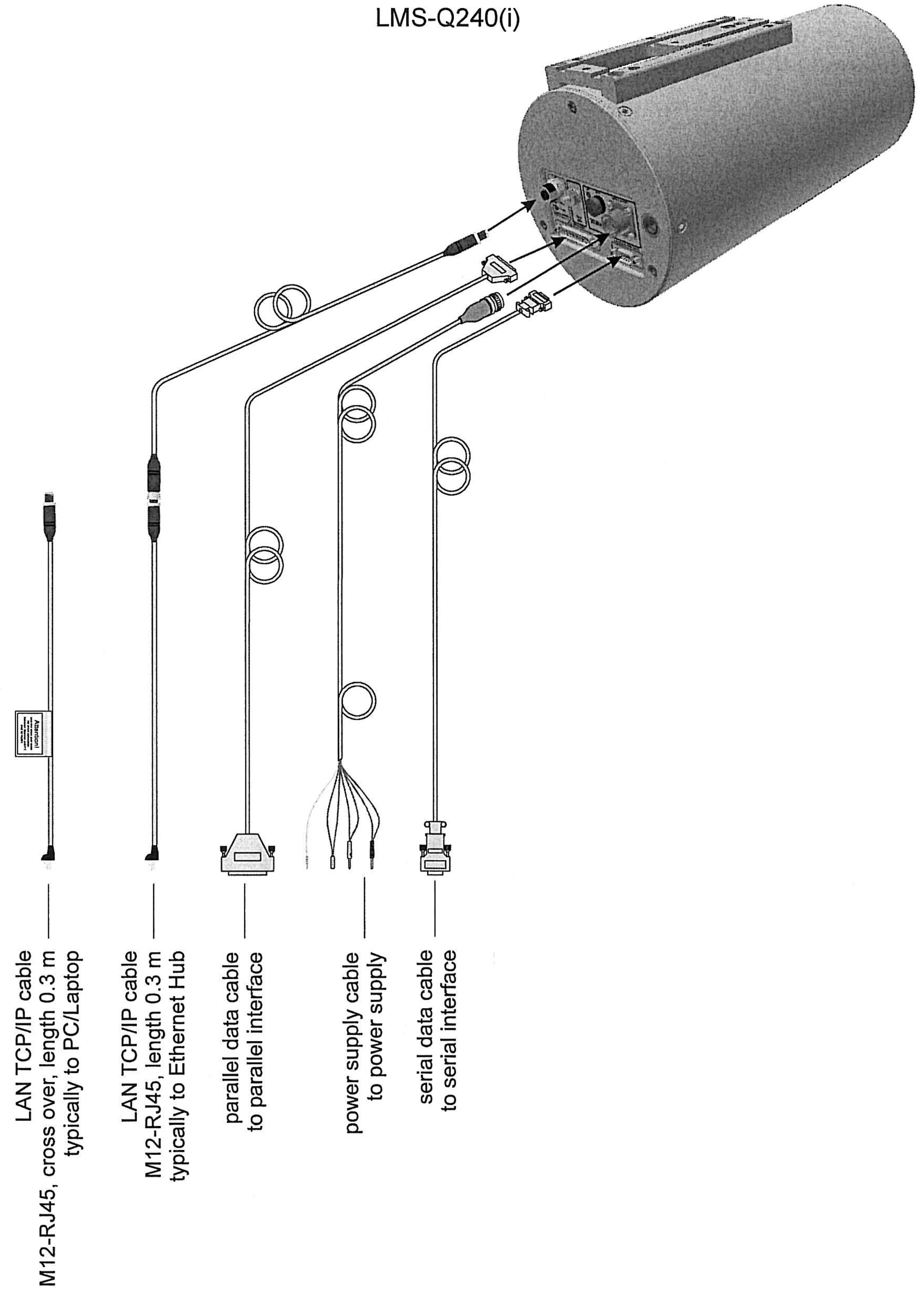


Fig. 4 Cabling of laser scanner LMS-Q240(i)

1.4 Definition of Axes

The following drawings show the definition of the coordinate system of the scanner:

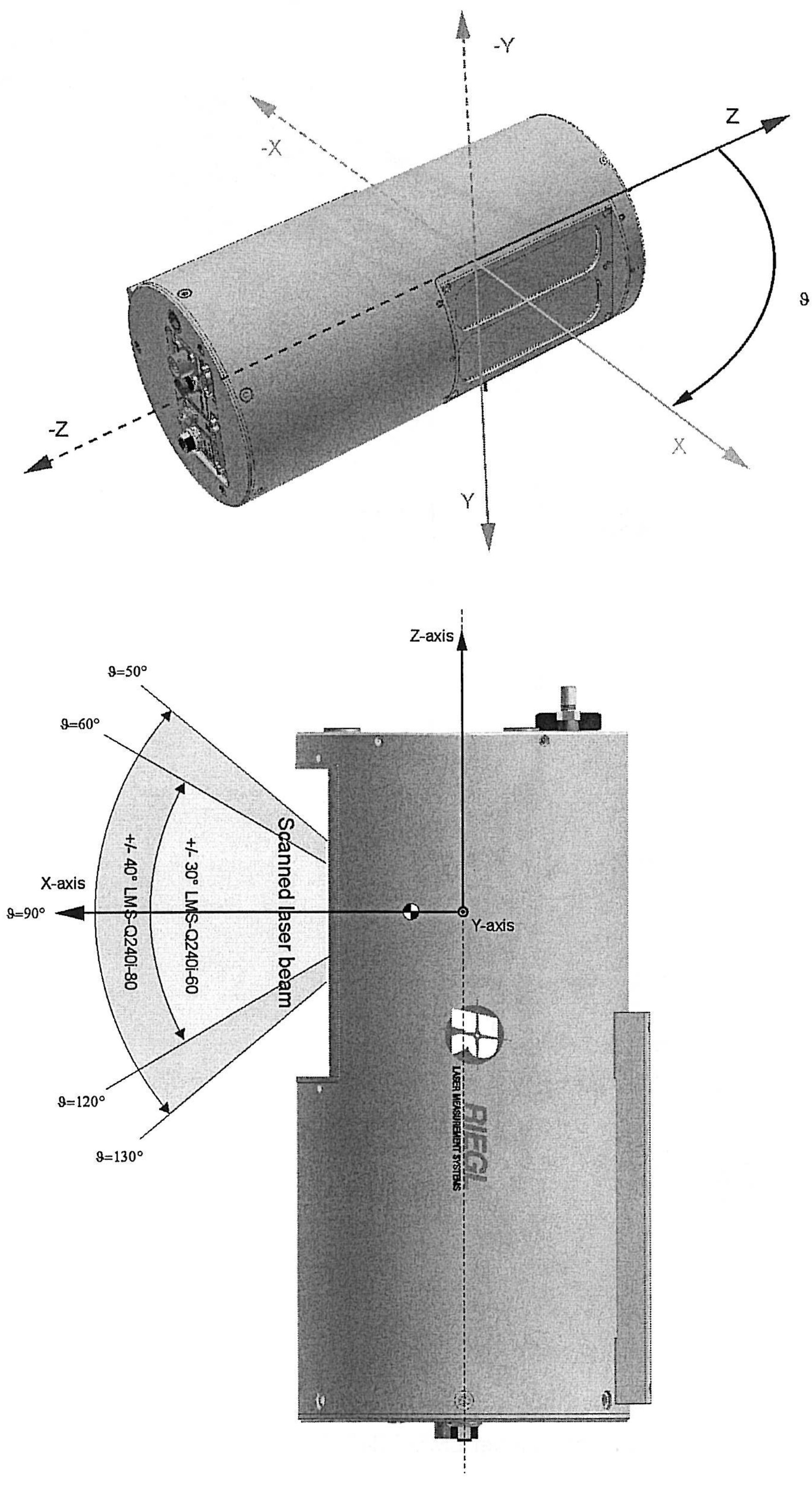


Fig. 5 Definition of axes

1.5 External Time Synchronization

The scanner provides an internal timer (called ***SyncTimer***) with a counting frequency of 100 kHz (quartz stabilized) and therefore a resolution of 10 microseconds. It is a 3 byte wide timer and automatically started with power up. With 3 bytes it overruns every 167.77216 seconds.

The *SyncTimer* can be cleared by supplying an external TTL pulse to a scanner input pin "Trigger" in order to synchronize the internal timer to an external event (e.g. 1pps GPS pulse). The pulse signal is called SYNC pulse. The SYNC pulse increments a counter, called **SyncCounter**.
If the pulses occur every second (as typically for GPS), the SyncCounter indicates seconds and the SyncTimer the part of the second since last change of second.

The edge type of the SYNC pulse clearing the SyncTimer is set to the rising edge when shipped. There is a user command to invert the input signal and thereby change the trigger to falling edge type (see 2.3.2.7 Trigger Input Inverter).
For specifications of this SYNC pulse refer to the technical part of this manual.

The trailer data sent at the end of each *scan line* provides the **SyncTimer and SyncCounter value of the first laser shot in scan data line.**

1.5.1 Optional Time Synchronization to GPS

An advanced synchronization to external time sources - typically GPS, which are able to provide TTL pulses every second - is available as an option.

The GPS synchronization uses the SyncTimer and the SyncCounter, but additional commands for synchronization to the external time source are available.

See details in chapter 2.6 Optional Time Synchronization to GPS

2 Data Interfaces

Two **interface modes** are available:

When using the scanner without *LAN - TCP/IP interface*, 2 interface ports are used:

- *Control port*: **serial *RS232*** (PC - COM) ***port*** for configuration/control
- *Data port*: **parallel *ECP*** (PC - LPT) ***port*** for fast scan data output

This mode is called ***Local control*** mode. It needs 2 cable connections between PC and scanner.

With the *LAN – TCP/IP interface*, these 2 interfaces (2 cable connections) can be replaced by one interface (one cable), using 2 ports:

- *Control port*: port number 20002 for configuration/control
- *Data port*: port number 20001 for fast scan data output

This mode is called ***LAN control*** mode.

All syntax rules and data format / structure descriptions of the following chapters are identically used for communication in *Local control* mode and *LAN control* mode: The rules relevant for the *Control port* (*Programming mode*, *Measurement mode*, parameters setting, control commands, error messages ...) and the structure definitions for the data interface (header, data structures, trailer ...) are identical.
For *LAN – TCP/IP* interface some specific commands for TCP/IP configuration (e.g. IP-address) have to be previously set using the *Local control* mode.

> **Restrictions on Data Utilization:**
> Measurement data of *RIEGL* Airborne Laser Scanner Systems and all data derived **MUST NOT BE USED** for flight planning, navigation, or flight control.
> The data of *RIEGL* Airborne Laser Scanners are intended to be used as a basis for surveying purposes only.
> *RIEGL* Laser Measurement Systems does not take on any responsibility for integration and/or certification of customer-specific system installations in or with aircrafts.

2.1 *Programming Mode* and *Measurement Mode*

Generally there are two **instrument modes** available:

- ***Measurement Mode***
- ***Programming Mode***

Measurement mode or *Programming mode* is selected by commands via the *Control port*.
The *Programming mode* is designed to be used to set and display measurement and control parameters. The *Programming mode* is started by sending a ^P (Ctrl P, ASCII 10hex) from the *Measurement mode* and finished with a Q<Cr> (like quit). <Cr> means Carriage return (ASCII 0Dhex). When *Programming mode* is finished, the instrument returns to *Measurement mode*.

Measurements and scans are carried out in the *Measurement mode*. After power up the instrument starts with *Measurement mode*.

For *Local control* mode, the communication parameters of the *RS232 port* are pre adjusted in factory to:

19200 baud
1 start bit
8 data bits
no parity
1 stop bit

> **Please note:** For achieving electromagnetic compatibility, use original *RIEGL* data cable for communications only!

2.2 *Control Port* Input /Output

2.2.1 Data Format in *Programming Mode*

The *Programming Mode* uses ASCII character strings to set parameters or ask for current parameter settings. After starting *Programming mode* with command ^P, the instrument replies with the message

*<Cr>[<Lf>]

where <Cr> means a Carriage return (0Dhex) and [<Lf>] means an optional Line Feed (0Ahex) (<Cr> or <Cr><Lf> sequence can be selected by the user via parameter CS) .

Basically the *Programming mode* works with a **command / reply concept**: A command is sent to the instrument, which answers with a reply message.

The first character(s) of the reply message always is (are)

** - when the last command could be interpreted correctly.
  - if a ^P has been sent. ^P starts or restarts the *Programming mode* and additionally clears the receive buffer (so when e.g. sending a ^P after an incorrect command string part, the incorrect characters already sent are cleared)

? when the last command could not be interpreted because
- the parameter value is out of range and/or
- an array index specified is out of range

?? when the last command could not be interpreted because:
- an unknown command was sent or
- the parameter cannot be accessed in the current access level

= when the value of a parameter was requested.

\ when the line is continued (the reply message consists of more than 1 lines)

Example:

Command	*Reply*	*Meaning*
T1<Cr>	*T1<Cr><Lf>	Measurement time T1
.T<Cr>	=T1<Cr><Lf>	Meas. time = T1
ABcd<Cr>	??ABCD<Cr><Lf>	ABCD is not a valid command

This example assumes that the separator <Cr><Lf> is selected.

Lower case letters of a command are converted to upper case letters internally. Line feeds <Lf>, following the <Cr> in the command string, are ignored. Spaces are ignored and therefore may occur everywhere in the command string.

2.2.2 Data Format in *Measurement Mode*

In *Measurement mode*, the *Control port* is used for

- **instrument control**
- **error and status message output**
- **result data output** (for scanners for debug purposes only!)

2.2.2.1 Instrument Control in *Measurement Mode*

In *Measurement Mode*, single **CONTROL characters** (ASCII 01hex ... 1Ahex) are used for operation control. These control characters are further represented by sequences "^" and a letter, e.g. ^A. The control characters ^A ... ^Z correspond to ASCII data 01hex ... 1Ahex.

Command	*Meaning*
^P (10hex)	From *measurement mode*, **start *programming mode*.** In *programming mode*, ^P restarts the program mode, therefore clears all characters already sent since the last carriage return (clears the input buffer).
^N (0Ehex)	**Switch on the laser (laser on only when additionally hardware laser lock is in status "on" and wheel in rotation,** see chapter 2.3.3.3 Switching the Laser Off and On, Laser Lock
^F (06hex)	**Switch off the laser**

2.2.2.2 Result and Status Output

Important note: In scanning mode the *Control port* result output must be switched off by command RO. Set RO8 to output result data to *Data port* only !

Result data output can be coded ASCII or BINARY. Mode ASCII or binary is selected by command RM (see chapter 2.3.3.1 Enabling the Desired Data Outputs for Results).
Status messages and error messages are always coded ASCII.

Data Format in Result Coding Mode ASCII

The ASCII data string has variable length and is delimited by <Cr> or <Cr><Lf> respectively. The data string is parted into separate blocks. The user can specify which **data blocks** are included into the data string.

The first character(s) within the block is(are) named the **block identifier**. **Block identifiers are always lower case letters**, where data (messages and status information) are always upper case letters. The following block identifiers are used:

r Distance, Range
a *Amplitude*, *Signal Intensity*
b *Line Angle*
q *Measurement Quality*
t SensorTimeStamp (*SyncTimer*)
m Message, status and error information

The length of the block depends on the data and is not constant. If the character following the identifier is a "+", a "-" or a ASCII-digit, the data block represents a number (e.g. the range in meters). If it is a letter, it represents status information.

Example: It is assumed that the output of range and *amplitude* is activated:

r123.4;a138<Cr><Lf>

Error and status information are messages and given in the following format: (e.g. error: supply voltage too low)

mERROR:LOW_BATT<Cr><Lf>

Result Data Format in Result Coding Mode BINARY

> **Please note:** Status and error messages are always given in ASCII mode, regardless of the result coding mode

The binary data string uses the most significant bit 7 (MSB) of data for synchronization purposes. The MSB is set to 1 for the first byte of the data string, and is set to 0 for the following bytes.
Data is included in the data string, when the corresponding bit in the Data format descriptor (see chapter 2.3.4.4 Selecting Data Blocks for the Data String) is set. Data is transmitted in order high to low byte.

Data then is issued in the following order:
Distance 3 bytes (if corresponding bit is set in F parameter)
Amplitude 1 byte (if corresponding bit is set in F parameter)
Line angle 4 byte (if corresponding bit is set in F parameter)
Measurement Quality 1 byte (if corresponding bit is set in F parameter)
SyncTimer (SensorTimeStamp) 4 byte (if corresponding bit is set in F parameter)

3 bytes Distance, order D1 – D2 – D3:
Distance [mm] = (D1 and 7Fhex) * 128 * 128 +
(D2 and 7Fhex) * 128 +
(D3 and 7Fhex)

1 byte *Amplitude* A1:
Amplitude [0..255] = (A1 and 7Fhex) * 2

4 bytes *Line angle*, order L1 – L2 – L3 – L4:
Line angle[degree/10000] = (L1 and 7Fhex) * 128 * 128 * 128 +
(L2 and 7Fhex) * 128 * 128 +
(L3 and 7Fhex) * 128 +
(L4 and 7Fhex)

1 byte Quality Q1:
Quality [0..100] = (Q1 and 7Fhex)

4 bytes *SyncTimer* (SensorTimeStamp), order T1 – T2 – T3 – T4:
Timer[10^{-5} s] = (T1 and 7Fhex) * 128 * 128 * 128 +
(T2 and 7Fhex) * 128 * 128 +
(T3 and 7Fhex) * 128 +
(T4 and 7Fhex)

Example:
Assume that F5 is set; then a data string

82 – 73 – 2F – 1C

means:
Distance = 2*128*128 + 115*128 + 47 = 47.663 mm
Amplitude = 56

As the MSB is set to 1 for the first byte only, a correct data reception procedure therefore should read data and wait for a byte with the MSB set to 1, then read a number of bytes according to the setting of the F command to read all data of 1 measurement. This method would automatically ignore all possible ASCII codes (status and error messages and *Programming mode*).

2.3 Parameters and Commands

2.3.1 Parameter Data Types in *Programming Mode*

The instrument supports the following data types:

Byte:	8-bit value, signed or unsigned
Integer:	16 bit value, signed or unsigned
Long:	32 bit value, signed or unsigned
String:	a sequence of characters
Command:	no value specified

These base types can be grouped to
Arrays: of byte, integer, long, string or command

Setting a parameter is done by specifying the **parameter name**. For arrays the name is followed by the **array range specification** given within brackets [ ]. For Bytes, Integers and Longs an **optional "="** may follow. For strings a "=" must follow. For data types byte, integer, long and string then the **value to be set** must follow.

Command	*Reply*	*Type*	*Meaning*
T=3	*T3	Byte	Setting measurement time
O-1000	*O-1000	Integer	Setting range offset – 1 m
W	*W	Command	Saving parameters

To **get the value of a parameter**, a point "." is located before the parameter name.

Command	*Reply*	*Meaning*
.T	=T3	Ask for current measurement time
.O	=O0	Ask for current offset
.ABC	?ABC	Don't know command ABC
.#SN	=#SN9991100	Ask for string serial number

If an **error is detected** (e.g. during execution of a command in *Programming mode* or previously in *Measurement mode*), all replies in *Programming mode* get an exclamation mark added. An error is pending until it is acknowledged by command "ERRACK" (so the exclamation mark is added to all command replies until the error is acknowledged).
See chapter 2.8 Errors and Error Handling for details.

Command	*Reply*	*Meaning*
W	*W!	Save, an error has occurred
.T	=T3!	Ask for current measurement time, error pending

Command	*Reply*	*Meaning*
^P	*!	Start of *Programming mode*, an error is already pending

2.3.2 Several Basic Commands

2.3.2.1 Starting and Finishing *Programming Mode*

Command	*Reply*	*Meaning*
^P	*	Start the *Programming mode*
Q<Cr>	***Q**	Quit the *Programming mode* and return to *Measurement mode*

Example:

Command	*Reply*	*Remark*
^P	*	*Programming mode* started
T0	*T0	Set measurement time T0
Q	*Q	Quit *Programming mode*

2.3.2.2 Reset

Command	*Reply*	*Meaning*
RESET<Cr>		Reset does an internal processor reset and a new start. Please note that the internal laser hardware is not reset (it still is power supplied), therefore this is not complete identical to switching off and on.

2.3.2.3 Setting Parameters to Default Values

Command	*Reply*	*Meaning*
DEFAULT<Cr>	*DEFAULT	This command sets several parameters to an initial default value.

The following parameters are set to the listed default status:

Default Setting	**Meaning**
T0	Measurement time T0
U0	Range unit meter
A2	Trigger mode free running
F13	data string includes range + *amplitude* + *line angle*

MQ50	Minimum measurement quality 50 percent
O0	Range Offset 0
CS1	*Control Port* interface: Line Separator <Cr> + <Lf>
RM0	*Control Port* interface: Result mode ASCII
RO8	Result output via *Data port* only
AL0	*Amplitude* window low value 0
AH255	*Amplitude* window high value 255
TS1	Last target measurement
XB1	*Data port* output: 1 measurement per block
XM1	*Data port* output: Hold 1 block in memory
XOS0	*Data port* output: Range in units of [1 mm]

2.3.2.4 Getting Help

Command	*Reply*	*Meaning*
.HELP	see example	Getting help to the available commands.
HELPFOR=[str]<Cr>	*HELPFOR[STR]	Restrict the list of commands to commands including the string [str]. If [str] is an empty string [], all commands are listed.
.HELPGROUPS	see example	Display all available help groups. Each available command belongs at least to one group.
.HELP[n]		Getting help for specific helpgroup n. A restriction of the command list by HELPFOR=[str] operates additionally.

Example for Help:

Command	**Reply**
HELPFOR=O	*HELPFOR=O
.HELP	\ O : User offset, Acc=RW, Integer[-32767,32767], Save=W
	=HELP
HELPFOR=C	*HELPFOR=C
.HELP	\ CB : Communication Baudrate, Acc=RW, Byte[0,9], Save=W
	\ CP : Communication Parity, Acc=RW, Byte[0,4], Save=W
	\ CS : Communication Separator, Acc=RW, Byte[0,1], Save=W
	=HELP
HELPFOR	*HELPFOR

Each help line to a parameter has the following structure:

\ParName : Short description , Access , Type and Range , Saving

Each line starts with "\" to indicate that another line follows.
ParName shows the parameter name to be entered; e.g. "O" is used to set the range offset. ParName[len] indicates an array type with len elements.

Short description describes the meaning of the parameter.

Access describes how the parameter can be used:
R = Read, W = Write, RW = Read and Write.
E.g. "HELP" can be used as Read command only (.HELP), "RESET" can be used as Write command only and "O" can be used as Read and Write command (writing and reading the offset).

Type and Range describes the parameter type and valid settings.
Byte is a 8 bit value, **Integer** a 16 bit value, **Long** a 32 bit value and **String** a character string. **Command** has no additional value to be set.
For Byte, Integer and Long the range of valid settings is indicated in the form **[min,max],** where min is the minimum possible setting and max is maximum possible setting. For strings the value within the brackets describes the maximum length.

"Save=W" means that the parameter setting can be saved with command "W", **"Save=#W"** means that the parameter setting can be saved with command "#W" (service level only) and "**Save=No**" means that nothing is saved.

Example for Help Groups:

Command	**Reply**
.helpgroups	\ 0: Basic \ 1: Info \ 2: Communication \ 3: Measurement \ 4: Laser \ 5: Scanner Basic \ 6: Scanner Extended \ 7: Optic \ 8: Angular Basic \ 9: Adjustment \ 10: Streams =HELPGROUPS
.help[1]	\ TIME : Current time, Acc=RW, String[9], Save=No \ DATE : Current date, Acc=RW, String[9], Save=No \ OPTIME[3] : Total operating-/Laser on-/ Motor on time, Acc=R, String[13], Save=No \ TEMP : Temperature, Acc=R, Byte[-127,127], Save=No =HELP[1]

2.3.2.5 Saving Parameters Permanently

Command	*Reply*	*Meaning*
W<Cr>	*W	Saves parameters permanently. That means that current settings are kept when the instrument is switched off and on again. Note that some time (typically tenth of seconds, but under certain circumstances up to 1-2 seconds) is needed to save data and to reply.

2.3.2.6 Total Instrument Operating Time

Command	*Reply*	*Meaning*
.OPTIME[0]<CR>	=OPTIME[0]hhhh:mm:ss	Total operating time of scanner, that is the total time the instrument has been power supplied. hhhh hours mm minutes ss seconds
.OPTIME[1]<CR>	=OPTIME[1]hhhh:mm:ss	Total laser operating time (laser on), format like OPTIME[0]
.OPTIME[2]<CR>	=OPTIME[2]hhhh:mm:ss	Total scan operating time (scanner in motion), format as for OPTIME[0]
.OPSECS[0]<CR>	=OPSECS[0]n	Total operating time of scanner in seconds, $0 \leq n \leq 2147483647$
.OPSECS[1]<CR>	=OPSECS[1]n	Total laser operating time in seconds, $0 \leq n \leq 2147483647$
.OPSECS[2]<CR>	=OPSECS[2]n	Total scan operating time in seconds, $0 \leq n \leq 2147483647$

2.3.2.7 Trigger Input Inverter

Command	*Range*	*Meaning*
TRIG_INVn<Cr>	$0 \leq n \leq 1$	0 = Inverter off: Clearing the SyncTimer with rising signal edge on trigger input. 1 = Inverter on: Clearing the SyncTimer with falling signal edge on trigger input

2.3.2.8 License Keys

Command	*Meaning*
LIC[0] *key_string0*<Cr> **LIC[1] *key_string1*<Cr>** **LIC[2] *key_string2*<Cr>** **LIC[3] *key_string3*<Cr>**	*key_string0, key_string1, key_string2* and *key_string3* represent license keys for PC drivers (e.g. RiScanLib) and software (e.g. RiScanPro) supplied by *RIEGL*. Therefore up to 4 license keys can be stored. When a valid license key is stored, no user input of the license key data at installation and/or software start up is necessary. If drivers and software are ordered together with the instrument, the correct keys are usually preprogrammed before shipping.

2.3.2.9 Date and Time

Command	*Meaning*
DATE=dd.mm.yy<Cr>	Set the current date, where dd means day of month, mm means month and yy the last two digits of the year. The value is transferred to the internal real time clock and does not need a separate save command. Example: DATE=01.12.05 sets current date to the first of December 2005. Needs Super User Password.
.DATE<Cr>	Query the current date
TIME=hh:mm:ss<Cr>	Set the current time, where hh means hour of the day, mm means minutes and ss means seconds. The value is transferred to the internal real time clock and does not need a separate save command. Example: TIME=11:55:00 means it is 5 minutes to twelve. Needs Super User Password.
.TIME<Cr>	Query the current time

2.3.3 Basic Measurement Parameters

2.3.3.1 Enabling the Desired Data Outputs for Results

Command	*Reply*	*Range*	*Meaning*
ROn<Cr>	***ROn**	$0 \le n \le 255$	Enabling the **output of measurement results** for different outputs: The bits of the value have the following meaning: bit 0: *Control port* result output bit 3: *Data port* result output Example: RO8 sets *Data port* output only, RO9 sets *Data port* + *Control port*

⚠ In scanning mode enable the *data port* output only, therefore set to RO8 !

Note that the command affects the **output of measurement results only**. **Errors and messages are reported on the *Control port*, even when the corresponding data output bit 0 in RO is cleared**. Similarly, the *Programming mode* is not affected by the setting for the *Control port* output in RO.

2.3.3.2 Selective Measurement of Strong Reflector Targets

Command	*Reply*	*Range*	*Meaning*
ALn<Cr>	***ALn**	$0 \le n \le 255$	Setting the minimum value of signal *Amplitude* values accepted
AHn<Cr>	***AHn**	$0 \le n \le 255$	Setting the maximum value of signal *Amplitude* values accepted

This feature allows to set a signal *amplitude* window to measure targets with a reflectivity within a certain range. E.g. to measure only targets equipped with reflectors and to make it insensitive for diffusely reflecting targets, set AL to a value of approx. 80 and AH to 255.

2.3.3.3 Switching the Laser Off and On, Laser Lock

The instrument provides a **software laser switch** and an external *input "laser safety lock*" (please refer to technical manual part for pinning). Both can be used to disable the laser emission for safety purposes.

To enable measurements (emission of laser shots), both software laser switch and laser safety lock must be set to status "laser on".

Command	Reply	Range	Meaning
LASERn<Cr>	*** LASERn**	$0 \leq n \leq 1$	Software laser switch n=0: laser off (like command ^F in *Measurement mode*) n=1: laser on (like command ^N in *Measurement mode*) At power up: n=1 (laser is on) **The laser is on, when software laser switch AND the external laser safety lock input are in status ON.**
.LASER_HWL<Cr>			Query input laser safety lock (hardware pin): n=0: laser off n=1: laser is switched on, if also the software laser switch is set to on.

2.3.3.4 Target Selection

Command	*Reply*	*Range*	*Meaning*
TSn<Cr>	***TSn**	$0 \leq n \leq 2$	0: measurement of FIRST TARGET 1: measurement of LAST TARGET 2: ALTERNATIG first / last target

If the laser beam hits a target, but a part of the beam passes by and hits further target(s) behind, several echo signals of the first and further target(s) are detected. The target mode selects, which of several targets is to be processed. If only one target is detected (no parts of the beam can pass by), first and last target mode display the same target (first target is last target).
The **LAST TARGET** mode is useful for measurement situations where targets in front (window glass, trees, bushes ...) partly block the sight to the desired measurement target. **FIRST TARGET** always processes the target hit by the beam at first, independent of further targets behind. **ALTERNATIG TARGET** alternates the first/last target mode, where for 2D scanners the target mode is alternated with each measurement, starting with first target for the first measurement in the scan line.

2.3.4 Adjusting Parameters for *Control Port* Mode

2.3.4.1 Setting the *Baud Rate* and Parity for *RS232/RS422 Port*

Command	*Reply*	*Range*	*Meaning*
CBn<Cr>	***CBn**	$0 \leq n \leq 9$	Setting the *Baud Rate* for communication via serial port n = 0: 150 n = 1: 300 n = 2: 600 n = 3: 1200 n = 4: 2400 n = 5: 4800 n = 6: 9600 n = 7: 19200 n = 8: 38400 n = 9:115200
CPn<Cr>	***CPn**	$0 \leq n \leq 4$	Setting Parity for communication via serial port n = 0: 8 Data Bits, no Parity n = 1: 8 Data Bits, even Parity n = 2: 8 Data Bits, odd Parity n = 3: 8 Data Bits, Parity = Mark (= No Parity, 2 Stop Bits) n = 4: 8 Data Bits, Parity = Space

These commands are utilized with the *Control port* in *Local mode* only.
Note that it is necessary to save parameters permanently by command "W" and to reset the instrument to activate new values of CB or CP.

2.3.4.2 Setting the Serial Mode *for RS232/RS422 Port*

Command	*Reply*	*Range*	*Meaning*
CMn<Cr>	***CMn**	$0 \leq n \leq 1$	Setting the Serial mode. n = 0: RS232 n = 1: RS422

Scanners usually use RS232 mode only!

This command is utilized with the *Control port* in *Local mode* only.
Note that it is necessary to save the parameter permanently by command "W" and to reset the instrument to activate new values of CM.

2.3.4.3 Setting the Line Separator

Command	*Reply*	*Range*	*Meaning*
CSn<Cr>	***CSn**	$0 \leq n \leq 1$	Setting the Line Separator for data string sent via *Control port*. n = 0: Carriage Return <Cr> n = 1: Carriage Return + Line Feed <Cr> + <Lf>

Note that it is necessary to save the parameter permanently by command "W" and to reset the instrument to activate new values of CS.

2.3.4.4 Selecting Data Blocks for the Data String

Command	Reply	Range	Meaning
Fn<Cr>	***Fn**	$0 \le n \le 65535$	Setting the data blocks forming the result output data string. The bits of the value have the following meaning: bit 0: Enable range data output bit 2: Enable *amplitude* data output bit 3: Enable *line angle* data output bit 5: Enable *measurement quality* data block bit 6: Enable *SyncTimer* data block Example: F5 sets output of range and *amplitude* value.

> ⚠ The setting of the parameter **F** also affects the data structure of the *Data Port.*

2.3.4.5 Coding Mode of the *Control Port* Result Output

Command	Reply	Range	Meaning
RMn<Cr>	***RMn**	$0 \le n \le 1$	Result coding mode. Coding the measurement result data of the *Control Port* Result output ASCII (0) or BINARY (1) **Note: The ASCII communication in *Programming mode* is not affected. Status and error messages are always given in ASCII mode, regardless of the setting of RM**

> ⚠ The setting of the RM – Parameter does not affect the data structure of the *data port,* which is used for scan data output. The setting of the RM parameter is relevant for debug purposes only.

2.3.5 Definition of Scan Pattern

2.3.5.1 Scan Pattern Basics

The scan pattern is mainly defined by the following parameters:

- line start angle
- line angle step width between measurements
- number of measurements forming one line
- laser pulse repetition rate for consecutive measurements in a line
- scanning mode

To set a scan pattern, first set all relevant parameters, then try to activate the parameters by sending the command "**SCN_APPLY**". The parameters are checked for consistency and the limits are checked; the command returns "*SCN_APPLY" for successful settings. If the parameters are not acceptable because system limits are exceeded (e.g. scan angle area or laser pulse rate out of limit), the command returns "?SCN_APPLY". In that case the parameter query ".SCN_APPLYERR" returns an error code indicating which of the parameters (or derived parameters) exceeds a system limit.

2.3.5.2 Scan Pattern Commands

Command	*Range*	*Meaning*
SCN_THETAS n <Cr>	-134217727 ≤ n ≤ 134217727, SCN_APPLY accepted range: SCN_THETALIM[0] ≤ n ≤ SCN_THETALIM[1]	**Line start angle** in units of 0.0001 degree
.SCN_THETALIM[0]<Cr> **.SCN_THETALIM[1]<Cr>**		Lower [0] and upper [1] limit for SCN_THETAS
SCN_THETAD n <Cr>	0 ≤ n ≤ 134217727, SCN_APPLY accepted range reduced !	***Line angle step width*** (angular difference between consecutive measurements) in units of 0.0001 degree
.SCN_THETADLIM<Cr>		Lower limit of SCN_THETAD, minimum *line angle step width*
SCN_THETAN n <Cr>	0 ≤ n ≤ 134217727, SCN_APPLY accepted range: 1 ≤ n ≤ SCN_THETANMAX	**Number of measurements** in one scan line
.SCN_THETANMAX<Cr>		Upper limit of SCN_THETAN, maximum possible number of measurements in a single line for *ranging online scan* or for *monitoring scan,* but NOT for *sampling monitoring scan*
SCN_RATE n <Cr>	0 ≤ n ≤ 65535, SCN_APPLY accepted range: SCN_RATELIM[0] ≤ n ≤ SCN_RATELIM[1]	**Laser pulse rate** (*PRR*) for consecutive measurements in a line.
.SCN_RATELIM[0]<Cr> **.SCN_RATELIM[1]<Cr>**		Lower [0] and upper [1] limit for laser pulse rate
.SCN_THETAVLIM[0]<Cr> **.SCN_THETAVLIM[1]<Cr>**		Lower [0] and upper [1] limit for resulting "scanned beam angular speed" in units of 0.0001 degree/second
SCN_TRIGMODE n <Cr>	0 ≤ n ≤ 2	0: **Continuous scan without trigger,** no trigger necessary 1: **Single line triggered** mode Acquire a single line, trigger with ^X 2: **Line sequence triggered** mode Acquire lines, trigger with ^X, stop acquiring with ^P (programming mode) see also chapter 2.3.5.3 "Scan Trigger Modes".

Command	*Range*	*Meaning*
SCN_APPLY<Cr>		**Apply the scan parameters previously handled in this table.** If parameters are acceptable and do not exceed any system limit, the reply string is *SCN_APPLY and the requested scan is started If parameters are not acceptable, the instrument replies "?SCN_APPLY". Further information, which parameter (or derived parameter) exceeds a limit is available via the command ".SCN_APPLYERR"
.SCN_APPLYERR<Cr>		Query for errors to previously executed command SCN_APPLY. See the list below for returned error codes
.SCN_APPLYERR_MSG<Cr>		Query the error of last SCN_APPLY as readable error message

SCN_APPLYERR Error codes	
Error	*Meaning*
0	No error; last SCN_APPLY successful
1	Laser pulse rate too high; lower SCN_RATE
2	Laser pulse rate too low; increase SCN_RATE
10	Line axis start angle too small for pre shots; Previously to the first measurement in a line Laser PRESHOTS are emitted; The resulting angular range between start of mirror facet and line start angle is too small to emit pre shots with the requested angular line step width; increase SCN_THETAS or decrease SCN_THETAD
11	Line axis start angle too small; increase SCN_THETAS
12	Line axis start angle too large; decrease SCN_THETAS
13	Line axis angle step width too small; increase SCN_THETAD
14	Line axis number of measurements too large; decrease SCN_THETAN
15	Line axis resulting scan area too large; decrease SCN_THETAD and / or SCN_THETAN
16	Line axis resulting scan area outside; the resulting scan area exceeds the mechanically limited angular range of the glass window
17	Line axis resulting speed too low; increase SNC_THETAD
18	Line axis resulting speed too high; decrease SNC_THETAD
19	SCN_THETAN too small, increase SCN_THETAN
60	With option Scan Sync only: Resulting line axis speed too low for Scan Sync, increase SCN_THETAD
61	With option Scan Sync only: Resulting line axis speed too high for Scan Sync, decrease SCN_THETAD
62	With option Scan Sync only: Missing PPS pulse
63	With option Scan Sync only: Invalid PPS pulse sequence; there must be one pulse per second only
64	With option Scan Sync only: Internal error; lock / unlock limits out of order

2.3.5.3 Scan Trigger Modes

There are 3 trigger modes supported:

- Continuous scan without trigger
- Single line triggered
- Line sequence triggered

In mode **continuous scan without trigger** the scan needs no trigger to start. Scan line data is sent via the *data port*

Mode **single line triggered** needs a trigger character ^X in *Measurement mode* to be sent via the Control port to acquire 1 single line.

Mode **line sequence triggered** needs a trigger character ^X in *measurement mode* to be sent via the Control port to start acquisition of scan lines until data acquisition is stopped by switching to *programming mode.*

A typical sequence of scanning in mode "line sequence triggered" consists of:

- Definition and setting of the scan pattern
- Applying the scan pattern
- Switching scanner to measurement mode
- Connecting the host computer to the *data port* for data acquisition
- Start the scan data acquisition by sending the trigger character ^X via the *control port.* The scan sequence starts with the next *mirror facet* (so data acquisition is line synchronized).

Note: For fast trigger sequences, especially in mode "single line triggered", it is recommended to **switch off the sound** (command SOUND=0), otherwise the sound generation would lower the scanner response time between trigger and start of data acquisition.

2.3.5.4 Additional Scanner Commands

Command	*Meaning*
.SCN_FACETS<Cr>	Query for **number of *mirror facets*** of polygon wheel, typically 3 or 4
. EALDESC[0]<Cr>	Getting the **total angle range of a full circle** for the *line scan* axis, typically 3600000
. EALDESC[1]<Cr>	Getting the **number of encoder lines of a full circle** for the *line scan* axis, typically 144000
. EALDESC[2]<Cr>	**If 0**: *line angle* is scaled in **gon;** angle resolution of *line scan* angle = 400 / EALDESC[1] **If 1**: *line angle* is scaled in **degree;** angle resolution of *line scan* angle = 360 / EALDESC[1]
.EAL <Cr>	Get the current *Line angle* in units of 0.0001 degree
SC_NOSCAN <Cr>	Stop the scanning movement
.SC_STATUS <Cr>	Reading the status from the scanning module. Reading this value is necessary to get additional error information for the error “mERROR:SCAN_STATUS”. For correct operation the value is 0. The bits of returned value (starting with bit index 0) have the following meaning: Bit 2: 1=Scanner supply voltage currently out of range Bit 3: 1= Scan movement not full speed Bit 4: 1= Scanner movement off Bit 6: 1= *Angle Encoder* missing marker pulse Bit 11: 1= Scan No Motion Error Bit 12: 1= Line Scanner PLL locked Error Bit 13: 1= Scanner Supply Voltage out of range Error Bit 14: 1= Scanner FPGA Boot Error Bit 15: 1= Scanner FPGA Download Error
.SC_STATUS_LIST <Cr>	Reading and interpreting bits of SC_STATUS, displaying a list of message lines for each bit set in SC_STATUS, each message line starting with “\”. SC_STATUS = 0 is displayed as line “\ MOTION_OK”. Example: *Command* — *Reply*: .SC_STATUS_LIST — \ MOTION_OK =SC_STATUS_LIST
.SC_ERROR <Cr>	Reading this value is necessary to get additional error information for the error “mERROR:SCAN_COMMUNICATION”. The bits of returned value (starting with bit index 0) have the following meaning: Bit 0: 1= No PDR Error Bit 1: 1= Bad Frame Ctrl Byte Error Bit 2: 1= Bad Checksum at last R/W Error Bit 3: 1= Bad Echo at last R/W Error

Command	*Meaning*
	Bit 4: 1= PDR at wrong position Error Bit 8: 1= Bad command or answer Error Bit 9: 1= Unknown command Error Bit 10: 1= PDR2 timeout Error Bit 12: 1= Error in Scanner, to be found in SC_STATUS **With Error acknowledgement command "ERRACK" the value is set to 0.**
.SC_ERROR_LIST <Cr>	Interpreting bits of SC_ERROR, displaying a list of message lines for each bit set in SC_ERROR, each message line starting with "\"
.SC_VERSION <Cr>	Reading the current software version of the scanning module

2.3.5.5 Scan Pattern Example

A typical command sequence in order to set a scan pattern with 0.1 degree *Line Angle Step Width* and full (80 degree) scan area would be:

Command	meaning
^P	enter *programming mode*
SC_NOSCAN< Cr>	stop the scan motion, see chapter 2.3.7.1 (this command is optionally; a new scan pattern can be applied in the following steps without stopping the scan motion of the scan pattern currently programmed)
SCN_THETAS500000<Cr>	Line start angle at 50 degree
SCN_THETAD1000<Cr>	*Line angle step width* 0.1 degree
SCN_THETAN801<Cr>	so scan range is 80 degree (first measurement at 50, last at 130 degree)
SCN_RATE30000<Cr>	measure with Laser PRR of 30 kHz
SCN_TRIGMODE2<Cr>	Set to line sequence triggered mode (start measurements with trigger ^X)
SCN_APPLY<Cr>	Apply the scan pattern and (re)start the scan motion, if scan pattern can be applied correctly
W<Cr>	save permanently, if desired, otherwise omit this command
Q>Cr>	quit *programming mode* Now the PC should connect to the *Data port*
^X	Start the line sequence; line scan data are sent via the *Data port* until a ^P is received to stop measurements To acknowledge the start command, the scanner sends message "mSCANNING" via the *Control Port* to indicate the start of measurements
.	... measurement of (multiple) lines
^P	stop the measurements
Q<Cr>	quit *programming mode*
^X	... and restart measurements of (multiple) lines

2.3.6 Several Additional Commands

2.3.6.1 Getting Version, Type and Serial Number Information

Command	*Meaning*
. #V<Cr>	Getting Software Version
. #VI<Cr>	Getting Instrument Ident
. #VN<Cr>	Getting Instrument Name
. #VM<Cr>	Getting Instrument Modification
. #SN<Cr>	Getting Instruments serial number

2.3.6.2 Getting Supply Voltage and Instrument Temperature

Command	*Meaning*
. V<Cr>	Getting supply voltage in units of 0.1 volt. This voltage is measured internally, therefore voltage loss from the supply cable can not be taken into account.
. TEMP<Cr>	Getting Temperature within instrument in units of degree Celsius. Note that the temperature within in the instrument usually is some degrees higher than ambient temperature.

2.3.6.3 Controlling the Integrated Buzzer

Command	Range	Meaning
BUZ_LOUDNESS n<Cr>	$0 \leq n \leq 255$	Setting the loudness of the integrated buzzer, where 255 corresponds to maximum value. At startup the value is set to 255. Needs Super User Password
SOUND n<Cr>	$0 \leq n \leq 255$	Setting the sound level n = 0: Sound off (note: beep sequences indicating e.g. focus calibration or emission of laser radiation for instruments with solid state laser, are not affected by this setting. These beep sequences cannot be switched off). n = 1: reduced sound level (indicating scan start and scan stop) n = 2: full sound level

2.3.7 Additional Low Level Commands

The following list of commands is usually not needed for standard usage.
It lists

- Commands for scan specification as used with earlier *RIEGL* scanner types (parameters derived automatically by command SCN_APPLY)
- Commands needed for debugging and testing
- General range finder commands, not needed for the scanning application

2.3.7.1 Low Level Scanning Commands

Command	*Range*	*Meaning*
RF_START_Ln <Cr>	0 ≤ n ≤ 1200000 (for Q240(i)-80) 0 ≤ n ≤ 900000 (for Q240(i)-60)	Setting the **start angle for the *Line Scan*** in units of 0.0001 degree. Note: With activated pre shots (RF_PRENUM_L>0) the angle of the first measurement is RF_START_L + (RF_DELTA_L x RF_PRENUM_L); RF_START_L then defines the angle of the first pre shot
RF_NUMBER_L n <Cr>	1 ≤ n ≤ XBMAX [1)] [1)] see chapter 2.4.3	Setting **the number of measurements per line**. Note: In scanning mode the setting of XB for the *Data Port* output (block buffer size) is automatically set equal to RF_NUMBER_L.
RF_PRENUM_L n <Cr>	0 ≤ n ≤ 65535	Setting the number of *pre laser shots* (prior to each *Line Scan*). This pre shots are necessary for warming up the laser to get results with best possible accuracy. **DO NOT CHANGE THIS VALUE.**
RF_DELTA_L n <Cr>	0 ≤ n ≤ 3599999	Setting the ***Line Scan angle step width*** between two consecutive laser shots in units of 0.0001 degree. Note: According to the speed of the Polygon wheel the angle increment must be high enough not to exceed the maximum laser repetition rate
SC_PRR_NOMINALn<Cr>	typically 29950 ≤ n ≤ 30050	Setting the laser pulse repetition rate for the scan; this means adjusting the scan speed according to the setting of RF_DELTA_L so that the requested pulse repetition rate results.
SC_SCAN <Cr>		Start the scanner motion (movement). Note: Successful SCN_APPLY also automatically starts the scan

Command	*Range*	*Meaning*
SC_WRITE <Cr>		Writes the current settings of parameters values starting with "SC_" like SC_START_F etc. to the scanning module to make them active! This command is executed automatically at startup sequence!
ICAN <Cr>		Super User Password
AUTOSCAN n <Cr>	$0 \leq n \leq 1$	Setting the instrument mode at startup. **For scanning AUTOSCAN must be set to 1**. Value 0 is used only for factory adjustments. Needs Super User Password
SCAN <Cr>		Start the scanning mode in range finder module. Note that the scanner (scanning movement) must be started separately by command SC_SCAN. With parameter AUTOSCAN = 1, commands SCAN and SC_SCAN are executed automatically Needs Super User Password
NOSCAN <Cr>		Stop the scanning mode in range finder module. Note that the scanner (scanning movement) must be stopped separately with command SC_NOSCAN. For debug purposes only. Needs Super User Password
SC_RESET <Cr>		Resets the scanning unit Needs Super User Password
.#MID_MAIN <Cr>		Getting the *Data Port* data output Measurement ID Main. Needs Super User Password
.#MID_SUB<Cr>		Getting the *Data Port* data output Measurement ID Sub. Needs Super User Password
.#PID_MAIN <Cr>		Getting the *Data Port* data output Parameter ID Main. Needs Super User Password
.#PID_SUB<Cr>		Getting the *Data Port* data output Parameter ID Sub. Needs Super User Password
.#LID_MAIN <Cr>		Getting the *Data Port* data output LeadIn ID Main. Needs Super User Password
.#LID_SUB<Cr>		Getting the *Data Port* data output LeadIn ID Sub. Needs Super User Password

Command	*Range*	*Meaning*
.#TID_MAIN <Cr>		Getting the *Data Port* data output Trailer ID Main. Needs Super User Password
.#TID_SUB<Cr>		Getting the *Data Port* data output Trailer ID Sub. Needs Super User Password

2.3.7.2 (Extended) Distance Meter Commands

The scanner includes a *RIEGL* distance meter module, which is set to a mode to fit to the application in the scanner. The following part lists these commands.

For correct scanning operation, some parameters must be set to specific values. These adjustments are done in the factory.

> Do not change these parameters, otherwise correct scanning operation cannot be guaranteed.

Command	*Range*	*Setting*	*Meaning*
Tn<Cr>	0 ≤ n ≤ 7	don't care	Selection of **measurement time**. Setting Meas. Time Laser shots / measurement T0 1/30000 sec 1 T1 1/15000 sec 2 T2 5/30000 sec 5 T3 1/300 sec 100 T4 1/15 sec 2000 T5 1/6 sec 5000 T6 1/3 sec 10000 T7 1/2 sec 15000 **In scanning mode the measurement time setting is ignored. Each measurement is a single shot measurement (T0) at a certain angle. The command is used for debug purposes in mode NOSCAN only.**
Un<Cr>	0 ≤ n ≤ 2	don't care	Selection of **range measurement unit** n = 0 : unit meter n = 1 : unit feet n = 2 : unit yards The conversion factors are: 1 meter = 3.28084 feet 1 meter = 1.0936 yards **For scanning: Note that units of measurement for *Data port* output are not affected, the *Data port* output always uses meters.**
An<Cr>	0 ≤ n ≤ 2	2	Selection of measurement trigger mode n = 0: Trigger external (TTL input) n = 1: Trigger via *Data Port* interface (^X) n = 2: Free running, automatic start **For scanning mode always select trigger mode A2**

Command	*Range*	*Setting*	*Meaning*
On<Cr>	-32767 ≤ n ≤ 32767	0	Setting a range offset value in units of [mm]. This offset is used to adapt the zero plane according to different mounting positions. Positive values increase the displayed range value. **For scanning mode always set offset 0**
MQn<Cr>	0 ≤ n ≤ 100	don't care	Setting the **minimum required measurement quality**. The quality of a measurement is relevant for results obtained by averaging of single shot measurements only (measurement times > T0). The quality is the percentage of emitted laser shots resulting in a detectable echo signal. Measurement results with quality values less than MQ indicate "no target" For single shot measurements the quality can only be 0 (no echo detected) or 100 (echo detected). According to the average rate of the selected measurement time, the minimum number of pulses necessary for a valid measurement result MQn is calculated internally. The resulting MQn value must be ≥ 1 (At least one single shot measurement is needed; it is internally set to 1, if the MQn value is less than 1).

2.4 *Data Port* Output

The *Data port* is an output port only. It is designed for fast scan data output. The physical output port can either be the *ECP parallel port* when operated in *Local control* mode or a port of the LAN - TCP/IP interface when operated in *LAN control mode*.

2.4.1 ECP Parallel Port

The *ECP parallel port* is a **8 BIT PARALLEL *DATA PORT***. It has been designed to be directly connected to an **ECP compatible LPT printer port of a PC** (this ports are also addressed *extended parallel ports* or *enhanced parallel ports*). ECP ports offer high data transfer rates. Although designed for personal computers, interfacing to other equipment is rather straight forward.
Specifications of the ECP mode can be found in IEEE standard 1284 -1994.

To read data from the *ECP parallel port* in *Local control* mode, *RIEGL* supports a driver for WINDOWS NT/2000/XP. Examples, written in C, show how to use functions

- RiPortOpen
- RiPortRead
- RiPortClose

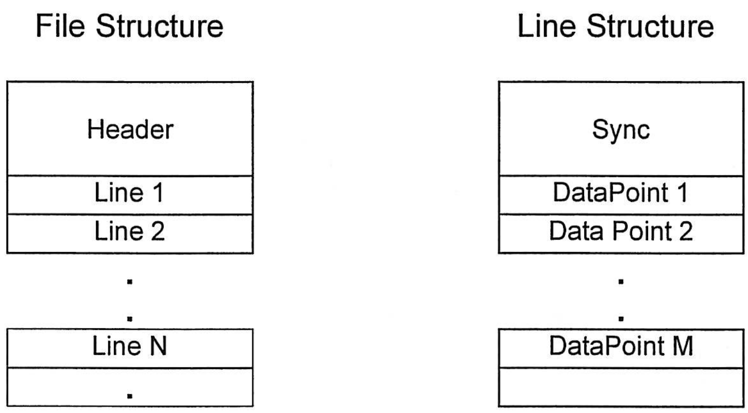
to read data from the *ECP parallel port.* (see the C++ source file RiPort.CPP)

With **RiPortOpen** the *ECP parallel port* is opened to receive data from the instrument. The first block read is the header block. Note that the header record is read only once after opening the ECP file (see below)

2.4.2 Reading Data from *Data Port*

Reading data from the *Data port* can be seen like reading a data file. As for a file, there is an open and a close function. The open function corresponds to connecting to the *Data port*, the close function to disconnecting from the *Data port*.

The "*Data port* file" starts with a **header record**, followed by **line records** containing the line data (measurements). Every line record consists of a synchronization sequence, followed by (a user definable number of) data points and finished with trailer. Consecutive line records are grouped to a **frame** building one scan image.



After opening the file, the header is read, followed by the data records. To reread the header, it is necessary to close the file and open it again.
As the header includes several data items depending on the parameter settings for the scan (e.g. number of measurements per *scan line*), it is necessary to close the file (disconnect from *Data port*) before changing relevant (scan-) parameters. In other words: Before changing parameters In *Programming mode* via the *Control port*, disconnect from *Data port*, make relevant parameter settings and changes then and return to *Measurement mode* before reconnecting to the *Data port*.

The following structure definitions use a notation
Name1 : BitSize1 - Name2 : BitSize2
Where NameX is the name of element X and BitSizeX the bit size of the element X.

2.4.2.1 Structure of Header

The header consists of three consecutive blocks

- Header Preamble Block
- Header Main Block
- Header Parameter Block

Header Preamble Block

HeaderSize:32 – DataSetLen:16 – ProtocolId:8 – HeaderID:8

Element name (size)	*Meaning*	*Value*
HeaderSize (unsigned 32 bit)	is the size of the header in bytes, including the bytes of HeaderSize. Therefore data starts HeaderSize bytes after the first byte (lowest byte of HeaderSize) of the header	specific
DataSetLen (unsigned 16 bit)	Length of Dataset in Bytes not including optional Sync field, *CRC*... (For each bit set in ProtocolID, 2 bytes have to be added to DataSetLen to jump to the next Data set.)	Scan specific
ProtocolID (unsigned 8 bit)	Each bit set describes a 16 bit field added to a dataset Bit 0 = 0: No Sync field present Bit 0 = 1: 16 Bit Sync field present Bit 1 = 0: No *CRC* field present Bit 1 = 1: 16 Bit *CRC* field present	typically 1
HeaderID (unsigned 8 bit)	is the Header identity	10

Header Main Block for Header ID 10

MeasOffset:16 - MeasSize:16 - MeasCount:16- LeadInIDMain:8 - LeadInIDSub:16 - MeasIDMain:8 - MeasIDSub:16 - TrailerIDMain:8 - TrailerIDSub:16 - ParameterIDMain:8 - ParameterIDSub:16

Element name (size)	*Meaning*	*Value*
MeasOffset (unsigned 16 bit)	Offset of the first byte of the Meas record from the start of the data. Adding the offset effectively 'jumps' over the LeadIn record of the data	0
MeasSize (unsigned 16 bit)	Size of the one Meas record (data point) in bytes. Knowing the size allows ignoring unknown fields. The n'th measurement thus has an offset of MeasOffset + n * MeasSize	Scan specific
MeasCount (unsigned 16 bit)	Number of measurement per Data set (line)	Scan specific
LeadInIDMain (unsigned 8 bit)	LeadIn Record ID Main; if Main and Sub are zero, this record is not present in the Data record	0
LeadInIDSub (unsigned 16 bit)	LeadIn Record ID Sub	0

Element name (size)	*Meaning*	*Value*
MeasIDMain (unsigned 8 bit)	Meas Record ID Main	130
MeasIDSub (unsigned 16 bit)	Meas Record ID Sub, interpret the bits similar to description of Parameter F	specific
TrailerIDMain (unsigned 8 bit)	Trailer Record ID Main , if Main and Sub are zero, this record is not present in the Data record	9
TrailerIDSub (unsigned 16 bit)	Trailer Record ID Sub	0
ParameterIDMain (unsigned 8 bit)	Parameter Record ID Main	8
ParameterIDSub (unsigned 16 bit)	Parameter Record ID Sub	0

Header Parameter Block

Parameter ID 8.0

SerialNumber:64 – RangeUnit:32 – AngleUnit:32 – TimerUnit:32 – PolarAngleID:8 – HWRes:8 – Target:8 – BeamAperture:16 – BeamDivergence:16 – BeamFocus:16 – BeamSeparationLength:16 - FactoryAdjustmentData:112x8 – TimeSyncEpochString:256 – TimeSyncSourceDesc:64 – SyncFlags:8

Element name (size)	*Meaning*	*Value*
SerialNumber (64 bit)	null terminated string, 8 characters, instrument serial number	specific
RangeUnit (floating point 32 bit)	Range measurement unit in meters [m]	0.001
AngleUnit (floating point 32 bit)	Angle unit in gon. Full circle is 400 gon.	0.0001111111
Timer Unit (floating point 32 bit)	SensorTimeStamp (SYNC Timer, time of measurement) unit in seconds [s].	0.00001
PolarAngleID (unsigned 8 bit)	Defining the rule for calculation of the *line angle* (polar angle) of the laser beam, based on the PolarAngle value supplied by the scanner. The factor 2 is necessary, because the beam angle is twice the polygon mirror angle due to the mirror reflection. 0: BeamPolarAngle = PolarAngle * AngleUnit >0: BeamPolarAngle = 2 * (PolarAngle mod (400/AngleUnit/PolarAngleID)) * AngleUnit In case PolarAngleID > 0 it reflects the number of facets of the spinning polygonal wheel of the line scanner	typically 3 or 4
HWRes (unsigned 8 bit)	for compatibility to older versions only	2
Target (unsigned 8 bit)	Setting of parameter TS, defining the target detection mode (see chapter 2.3.3.4 Target Selection)	0..2
BeamAperture (unsigned 16 bit)	Aperture of Measurement laser beam [0.1 mm].	specific

BeamFocus (unsigned 16 bit)	Measurement laser beam focused to distance [cm]	specific
BeamDivergence (unsigned 16 bit)	Divergence of Measurement laser beam [10 µrad]	specific
BeamSeparationLength (unsigned 16 bit)	Factory data used by *RIEGL* RiScanLib for further processing of raw data	specific
FactoryAdjustmentData (112 x unsigned 8 bit)	Data used by *RIEGL* RiScanLib for transforming raw data (range and angle) into Cartesian coordinates	specific
TimeSyncEpochString (32 x 8 bit)	Character string, null terminated, set with synchronization commands with optional feature "Time Synchronization to GPS". At instrument startup, set according to internal real time clock. Format "YYYY-MM-DDThh:mm:ss", e.g. "2005-12-24T12:34:56".	Specific
TimeSyncSourceDesc (8 x 8 bit)	Character string, null terminated, describing the source of time synchronization. At instrument startup, set to • "RTC" if instrument supports optional "Time Synchronization to GPS" "UNSYNC" if instrument does not support optional "Time Synchronization to GPS"	Specific
SyncFlags (unsigned 8 bit)	Bit 7: TIME SYNC NOT SUPPORTED 1 = "Time Synchronization to GPS" not supported Bit 6: SYNC_NEVER_EXECTUED 0 = at least 1 synchronization successful executed 1 = no successful synchronization executed Bit 5: LAST_PPS_MISSED 0 = last GPS pulse detected, correctly synchronized 1 = last GPS pulse missed Bit 4: ANY_PPS_MISSED 0 = never missed a GPS pulse since last synchronization 1 = missed at least one GPS pulse since last synchronization	Specific

Header Example

Hex sequence	Meaning
D2 00 00 00	HeaderSize is 210 bytes
4A 1F	DataSetLen: 8010 bytes per DataSet (line) (800 measurements x 10bytes + 10 bytes trailer)
01	ProtocolID = 1 (Sync Field on)
0A	HeaderID = 10
00 00	MeasOffset = 0
0A 00	MeasSize = 10, each measurement has 10 bytes
20 03	MeasCount = 800, 800 measurements per line
00	LeadInMain = 0
00 00	LeadInSub = 0
82	MeasIDMain = 130
4D 00	MeasIDSub = binary 01001101 (range + *amplitude* + *line angle* + *sync timer*)
06	TrailerIDMain = 9
01 00	TrailerIDSub = 0
04	ParameterIDMain = 8
02 00	ParameterIDSub = 0
39 39 39 33 33 37 31 00	Null terminated serial number string = “9993371”
6F 12 83 3A	RangeUnit = 0.001 m
51 04 E9 38	AngleUnit = 0.000111111 gon (corresponds to 0.0001 degree)
AC C5 27 37	Timer unit = 0.00001 seconds
03	PolarAngleID = 3, 3 *mirror facets*
02	HWRes = 2, hardware resolution
00	Target selection = 0, first target mode
A4 01	Beam Aperture 42.0 mm
2C 01	Beam Divergence 3.00 mrad
FF FF	Beam Focus at infinity
88 13	BeamSeparationLength
00 00 00 00	FactoryAdjustmentData 1
00 00 00 00	FactoryAdjustmentData 2
00 00 00 00	FactoryAdjustmentData 3
00 00 00 00	FactoryAdjustmentData 4
00 00 00 00	FactoryAdjustmentData 5
00 00 00 00	FactoryAdjustmentData 6
00 00 00 00	FactoryAdjustmentData 7
00 00 00 00	FactoryAdjustmentData 8
00 00 00 00	FactoryAdjustmentData 9
00 00 00 00	FactoryAdjustmentData 10
00 00 00 00	FactoryAdjustmentData 11
00 00 00 00	FactoryAdjustmentData 12
00 00 00 00	FactoryAdjustmentData 13
00 00 00 00	FactoryAdjustmentData 14
00 00 00 00	FactoryAdjustmentData 15
00 00 00 00	FactoryAdjustmentData 16
00 00 00 00	FactoryAdjustmentData 17
00 00 00 00	FactoryAdjustmentData 18
00 00 00 00	FactoryAdjustmentData 19
00 00 00 00	FactoryAdjustmentData 20
00 00 00 00	FactoryAdjustmentData 21
00 00 00 00	FactoryAdjustmentData 22
00 00 00 00	FactoryAdjustmentData 23
00 00 00 00	FactoryAdjustmentData 24
00 00 00 00	FactoryAdjustmentData 25
00 00 00 00	FactoryAdjustmentData 26
00 00 00 00	FactoryAdjustmentData 27
00 00 00 00	FactoryAdjustmentData 28
32 30 30 36	

2D 30 39 2D	
32 31 54 31	
32 3A 30 32	
3A 32 36 00	
00 00 00 00	
00 00 00 00	
00 00 00 00	TimeSyncEpochString = "2006-09-21T12:02:26"
47 50 53 00	
00 00 00 00	TimeSyncSourceDesc = "GPS"
00	SyncFlags = 0

2.4.2.2 Structure of Data in a Line

Consecutive data lines have the following structure:

Synchronization sequence (identical to DataSetLen in header)
Data point 1
Data point 2
.
Data point M
Trailer

Synchronization sequence (identical to DataSetLen in header)
Data point 1
.
.

The **Synchronization Sequence** is formed by 2 bytes representing the number of bytes in one line. This value is identical to the value of DataSetLen in the header.

Structure of a Data Point

Each data point with Measurement ID 130.x has the following structure:

Range:24 (if bit 0 of MeasIDSub is 1) - *Signal Intensity*: 8 (if bit 2 of MeasIDSub is 1) - AngleOfMirrorWheel:24 (if bit 3 of MeasIDSub is 1) – MeasurementQuality: 8 (if bit 5 of MeasIDSub is 1) – ShotSyncTimer:24 (if bit 6 of MeasIDSub is 1)

Element name (size) ***Byte Order***	***Meaning***
Range (unsigned 24 bit) R1-R2-R3	Distance to target [Parameter.RangeUnit] = R1 +256*R2 + 65536*R3
Amplitude (unsigned 8 bit) A1	Signal *amplitude* of received target echo signal[0..255] = A1
AngleOfMirrorWheel (unsigned 24 bit) L1-L2-L3	Angle of mirror wheel [Parameter.AngleUnit] = L1 +256*L2 + 65536*L3 **To calculate the laser beam angle, see Parameter. PolarAngleID**
MeasurementQuality (unsigned 8 bit) Q1	Measurement quality, for single shot measurements this is always 100 (measurement ok) or 0 (no measurement); not needed for standard scanner applications
ShotSyncTimer (unsigned 24 bit) T1-T2-T3	Time of laser shot with reference to the value supplied in the trailer (LineSyncTimer and LineSyncCounter). The value in the trailer specifies the time of the first laser shot in the scan line. The ShotSyncTimer TS = T1 + 256*T2 + 65536*T3 To calculate the time T of the shot with reference to time of synchronization (epoch-string): T[s] = LineSyncCounter + (LineSyncTimer + ShotSyncTimer) * TimerUnit. TimerUnit = 0.00001[s] and specified in the header.

Example:
It is assumed that the F parameter is set to 4Dhex (enable output of range + *amplitude + line angle + shotsync timer*):

Data following the header record could have the following format (in hex):

27 00
FD 09 00 48 10 20 16 00 00 00
DE 09 00 3E 04 22 16 03 00 00
F0 09 00 42 F8 23 16 06 00 00

Sync Sequence: 27 00

Range 1:	0009FDhex => 2557 x RangeUnit = 2.557 m
Amplitude 1:	48hex => 72
MirrorAngle1:	162010hex = 1450000 Q240(i)-80: => BeamAngle = (1450000 mod 1200000)*2*0.000111111 = 55.5555 gon = 50.0000 degree Q240(i)-60: => BeamAngle = (1450000 mod 900000)*2*0.000111111 = 122.2222 gon = 110.0000 degree
ShotSyncTimer1:	000000hex = 0 x TimerUnit = 0.0 sec since first measurement
Range 2:	0009DEhex => 2526 x RangeUnit = 2.526 m
Amplitude 2:	3Ehex = 62
MirrorAngle2:	162204hex = 1450500 Q240(i)-80: => BeamAngle = (1450500 mod 1200000)*2*0.000111111 = 55.6666 gon = 50.1000 degree Q240(i)-60: => BeamAngle = (1450500 mod 900000)*2*0.000111111 = 122.3333 gon = 110.1000 degree
ShotSyncTimer2:	000003hex = 3 x TimerUnit = 0.00003 sec since first measurement, ;shot 2 is approx 30 microseconds after shot 1

.

.

Structure of Trailer

The trailer with Trailer ID 9.0 as the following structure:

ScanStatus:8 – DataPortLineCounter:16 – GPSTimeSyncFlags:8 – LineSyncCounter:24 – LineSyncTimer:24

Element name (size)	***Meaning***
ScanStatus (unsigned 8 bit)	For future expansion.
DataPortLineCounter (unsigned 16bit) N1-N2	DataPortLineCounter = N1 + 256*N2 This Counter is incremented with each complete line transmitted to the internal DataPort port buffer. Consecutive DataPortLineCounter values of data read by the user system indicate, that no data is lost, or in other words, the system reading *Data port* data is fast enough not to loose any *scan line*.
GPSTimeSyncFlags (unsigned 8 bit) F1	This value reflects the current synchronisation status with GPS for time synchronisation and is identical with the value read by command ".SDT_SYNC_FLAGS", see chapter 2.6 Optional Time Synchronization to GPS for interpreting the single flag bits.
LineSync Counter (unsigned 24 bit) S1–S2–S3	LineSyncCounter = S1 +256*S2 + 65536*S3 Without optional feature "Time Synchronization to GPS": Counts the number of pulses on the input "Trigger". With optional feature "Time Synchronization to GPS": Counts the seconds since last successful synchronization. The value is latched with the first measurement of the scan line, so in conjunction with LineSyncTimer is **represents the time of the first measurement of the scan line**.
LineSyncTimer (unsigned 24 bit) T1-T2-T3	LineSyncTimer[Parameter.TimerUnit]= T1 +256*T2 + 65536*T3 Without optional feature "Time Synchronization to GPS": Counts the time in units of 10 microseconds (0.00001seconds) since last detection of a pulse on input "Trigger". Overflows after 167.77215 seconds to 0. With optional feature "Time Synchronization to GPS": See details in description to 2.6 The value is latched with the first measurement of the scan line, so in conjunction with LineSyncCounter is **represents the time of the first measurement of the scan line**.

Special Coded Status Data

There are several measurement situations where no distance measurement data is available:

Measurement situation	*Range*	*Amplitude*
No target available or badly reflecting target	0	0
Underflow, Distance value less than 0 (including offsets)	0	0

Note that errors and status message are not reported on the *data port* output.

2.4.2.3 Data Port Buffer

The *Data Port* of the instrument has a data buffer integrated. If the data receiving unit (the PC) is temporally inactive and cannot read data - e.g. when it has to handle other tasks – this will not cause loss of data.

This data buffer can be configured by the user via 2 parameters:

- The **number of data points** (measurements) **in a data block**
- The number of **blocks to be hold** in memory

> ⚠ Configuring the *data port* buffer is automatically done with command SCN_APPLY, so there is no need for the user to configure the *data port* buffer

These commands **divide the internal data buffer in blocks** (data lines) according to data block size, so each block can store the desired number of measurements.
The second parameter, the number of blocks to be hold, defines the maximum value of blocks to be hold in memory. Setting this hold parameter to N blocks means that the buffer **holds a maximum of N** blocks.
If N is large and reading of data is fast enough, the buffer will never fill up and data read is always last data.
On the other hand, if reading is interrupted for a longer time (e.g. because the data receiving unit is not interested on new data), this buffer will fill up. When full, each new data line then **overwrites the oldest of the N blocks** already filled.

Examples:

Blocks to be hold = 1; that means that 2 blocks are reserved. Start reading of a block (and interrupting the reading for even a long time) locks this block while the other block is always updated with the new data in the background.

Blocks to be hold = 2; that means that 3 blocks are reserved. Start reading of a block (and interrupting the reading for even a long time) locks one block while the other 2 blocks are filled alternatively in the background.

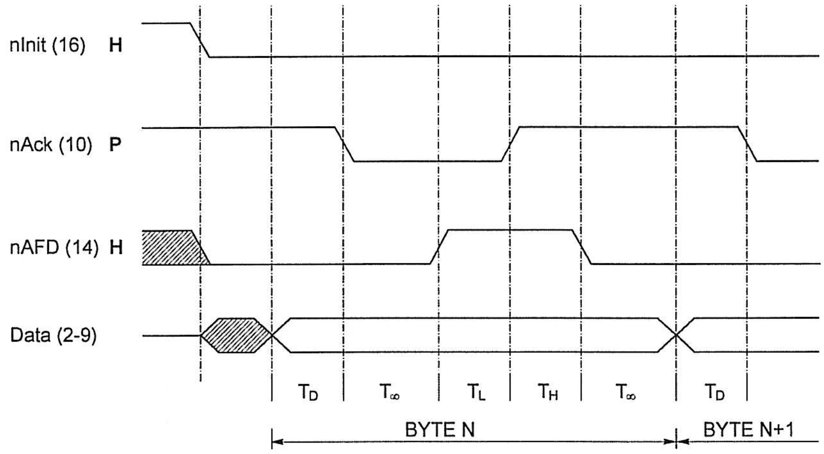
To guarantee, that **always new data is to be read** even when data reading is interrupted for longer times, set blocks to be hold to 1.

To guarantee, that **no data or as few as possible data is lost** even when data reading is interrupted for longer times, set blocks to be hold as large as possible (data is lost from that time on when the internal data buffer is full).

Data is set ready to be read via *Data port* as soon as a data block is filled. That means, if a block (line) is defined to store 10 measurements, the block can be read after 10 measurements (and not after the first one).

2.4.2.4 ECP Parallel Port Timing

Timing Diagram:



Time	Minimum	Maximum	Description
T_D	0		Minimum data set up time
T_∞	0	Infinite	Infinite response time
T_L	0	35 ms	Peripheral response time
T_H	0	1.0 s	Host response time

The timing diagram shows true voltage levels, which can be measured at the corresponding connector pins. The 'nInit' line controls direction of the data transfer. Since the only used direction is from the instrument to the computer, this line may be ignored on non personal computer based implementations. The data lines (2-9) must be input only on the computer in this case, and 'nInit' is held low permanently.

The instrument prepares for data transmission by placing data on the bus. After T_D the instrument then sets 'PeriphClk' (nAck) low to indicate to be ready to send the data. The computer then sets 'HostAck' (nAutoFd) high to acknowledge the handshake. The instrument then sets 'PeriphClk' (nAck) high. The computer is expected to accept the data and to set 'HostAck' (nAutoFd) low, completing the transfer.

2.4.3 Configuring the *Data Port* Output

> ⚠ Configuring the *data port* buffer is automatically done with command SCN_APPLY, so there is no need for the user to configure the *data port* buffer

Command	*Reply*	*Range*	*Meaning*
XBn<Cr>	***XBn**	$1 \le n \le$ XBMAX[1)]	Setting the number of measurements per block (line). Data is set ready on the *Data port* after n measurements (after whole data of a line is ready).
XMn<Cr>	***XMn**	$1 \le n \le$ XMMAX[1)]	Setting the number of blocks (lines) hold in memory. See description of *Data Port* output principles – Data Buffer chapter 2.4
XOSn<Cr>	***XOSn**	$0 \le n \le 6$	Setting the unit for range data; n = 0: 1 mm n = 1: 2 mm n = 2: 4 mm n = 3: 8 mm n = 4: 16 mm n = 5: 32 mm n = 6: 64 mm XOS is set to n=0, do not change it The unit and resolution of range data on the *Control Port* output is not affected.
.XS<Cr>			Query the *Data port* buffer size in words
XNn<Cr>	***XNn**	$0 \le n \le$ 65535	Set and query the DataPortLineCounter, sent with each line via the *Data Port* (see trailer description chapter). The counter e.g. can be reset to 0 before starting a scan sequence.

1) Due to the limitation of the internal buffer the values XB and XM have limits according to the following formulas:

[(BytesPerMeas * (XB-1) + TrailerLength + 5) div 2] * (XM+1) < XS
BytesPerMeas * (XB-1) + TrailerLength ≤ 65520
XM ≤ 8191

where **div** means an integer division. **TrailerLength** is the length of the trailer in bytes **BytesPerMeas** means the maximum number of bytes forming one measurement (11 bytes for Q240(i)). For both see chapter 2.4.2.2 Structure of Data in a Line.

Command	Reply	Range	Meaning
Fn<Cr>	***Fn**	$0 \leq n \leq 65535$	Setting the data blocks forming the result output data string. The bits of the value have the following meaning: bit 0: Enable range data output bit 2: Enable *amplitude* data output bit 3: Enable *line angle* data output bit 5: Enable *measurement quality* data block bit 6: Enable *SyncTimer* data block Example: F5 sets output of range and *amplitude* value.

2.5 LAN – *TCP/IP* Interface

2.5.1 Overview

The scanner provides a LAN interface. It implements 10Base-T/ 100Base-TX according IEEE802.3.

In your LAN, the scanner behaves like a data server. A client computer can connect to this server using *TCP/IP* protocol. The interface implements two ports:

- port 20001 for the *Data port*
- port 20002 for the *Control port*

Both ports are to be seen as re-routing the respective local ports (*RS232 port* => 20002 and *ECP parallel port* => 20001) to be accessible via a LAN. Thus, when the *LAN-TCP/IP interface* is active, the local interfaces, i.e. the *ECP parallel port* and the *RS232 port*, are inactive. There may be connections to these interfaces, but they will be 'dead'.

Port 20001 is the fast Data output port. Port 20001 does not accept any incoming data.

Port 20002 reroutes the *RS232 port*. Thus it provides the full functionality of the RS232-interface to the client (*Programming mode, Measurement mode*)

2.5.2 Activation

At startup of the instrument, the local interfaces are active (*Local mode*). Messages are sent via Local *control port = RS232 port.*
Depending on the setting of parameter IP_MODE,

- the LAN-*TCP/IP* interface remains deactivated (IP_MODE0).
- The instrument checks if a *LAN link* can be found (IP_MODE1), i.e. if the instrument is connected to a HUB (or PC) via a LAN cable. Message "mWAIT_LANLINK" indicates this status. If this link is found within the time specified by parameter IP_LANLINK_TIMEOUT, the instruments activates the LAN interface and deactivates the local interfaces (message "mLAN_CONTROL"). If the link is not found within this time, the LAN interface remains deactivated and the local interfaces keep active (message "mLOCAL_CONTROL").
- The instrument forces *LAN control* Mode (IP_MODE2). If a LAN link is detected, it immediately switches to LAN control mode. If not, it waits for a maximum of 3 seconds to receive the Super User password via the Local *control port = RS232 port* to switch to Local control mode. If the password is not received within this time, it switches to *LAN control mode.* Sending the password to enter Local control mode is necessary in emergency cases, e.g.
  - the IP address of the instrument is unknown, so no LAN connection can be established
  - a hardware problem makes a LAN connection impossible

2.5.3 Configuring the LAN Interface

Command	*Range*	*Meaning*
IP_ADDRn<CR>	000.000.000.000 ≤ n ≤ 255.255.255.255	Setting the IP address (default: 192.168.0.234)
IP_MODEn<CR>	0 ≤ n ≤ 2	Setting the activation mode of the LAN interface: n = 0: *Local mode* only; LAN interface is off, local interfaces (ECP parallel port , *RS232 port*) are active n = 1: LAN control mode is preferred (automatic switch to LAN control mode with detection of LAN link) n = 2: LAN control mode forced
IP_APPLY<CR>	-	Activates all IP-xxxx settings and resets the LAN – *TCP/IP* interface!

To understand the following parameters, detailed knowledge about *TCP/IP* is required. Thus be careful on modifying these settings!

Command	*Range*	*Meaning*
IP_CHAR_TIMEOUTn<CR>	$0 \leq n \leq 65535$.	Timeout for character transmission, unit [ms] .Needs Super User Password
IP_CON_TIMEOUTn<CR>	$0 \leq n \leq 3600$.	Timeout for *TCP/IP* connection, unit [s]: If client does not show any activity within this time, the instrument closes the connection. n = 0: no timeout, connection remains unlimited time $1 \leq n \leq 29$: behaves like n = 30 $30 \leq n \leq 3600$: timer activated Needs Super User Password
IP_TRANS_RETRIESn<CR>	$3 \leq n \leq 30000$	Maximum number of re-transmissions of a *TCP/IP* segment, if this number is exceeded, the instrument closes the connection. Needs Super User Password
IP_LANLINK_TIMEOUTn<CR>	$2000 \leq n \leq 65535$	Timeout for detection of a link on the LAN interface, unit [ms]. Needs Super User Password
IP_GATEWAYn<CR>	000.000.000.000 ≤ n ≤ 255.255.255.255	Setting the IP address of the gateway, this command is not used yet. Needs Super User Password
IP_SUBNETMASKn<CR>	000.000.000.000 ≤ n ≤ 255.255.255.255	Setting the subnet mask, this command is not used yet. Needs Super User Password

2.6 Optional Time Synchronization to GPS

An advanced synchronization to external time sources - typically GPS, which are able to provide TTL pulses every second - is available as an option.

The GPS synchronization uses the SyncTimer and the SyncCounter, but additional commands for synchronization to the external time source are available.

For synchronization it is necessary to feed the GPS data string to the controlling computer (PC) and the GPS pulse to the scanner trigger input only once for synchronization after system start up. To guarantee permanent GPS time resynchronization, it is recommended to keep the GPS pulse permanently connected to the scanner during further operation.
The controlling computer decodes time and date of the next following GPS pulse from the GPS data string, and sends a synchronization command including this time/date information to the scanner. Executing this command, the scanner waits for the next GPS pulse and clears SyncTimer and SyncCounter with this event. The time/date information of this synchronization is further called epoch and is provided in the scan data header. After successful synchronization, the time calculated by

ScannerTimeSinceSynchronisation [s] = SyncCounter + 0.00001 * SyncTimer

represents the time in seconds since last synchronization. Adding the time offset according to epoch data results in an absolute time of the day.

Each GPS pulse following time synchronization increments the SyncCounter by 1 and clears the SyncTimer. If the GPS pulse is missing, a scanner internal mechanism recognizes this pulse miss after 1.31072 seconds: The scanner then automatically increments the SyncCounter by 1 and restarts the SyncTimer with a start value of 31072, thereby guaranteeing correct time data, based on the scanner internal clock, further.

Note: As a consequence of the method described above, only GPS types providing 1 pulse per second (1PPS) are supported.

Synchronization flags, supplied with the scan data, give information about the current status of synchronization to GPS, allowing a quality indication for the time data.

In the scan data, time information then is supplied as a LineSyncCounter and a LineSyncTimer in the trailer of a scan line (and therefore once per scan line) and as a ShotSyncCounter with every measurement (every laser shot).

The LineSyncCounter and the LineSyncTimer are the SyncCounter and the SyncTimer of the FIRST MEASUREMENT in the scan line. The ShotSyncCounter is the time of the laser shot (measurement) with reference to the time of the first measurement. Therefore, the shot time with reference to the last scanner synchronization, is

ShotTime [s] = LineSyncCounter + 0.00001 * (LineSyncTimer + ShotSyncTimer)

To fit to *RIEGL* interface tools and libraries (RiScanLib), the epoch data string must use the following format:

YYYY-*MM*-*DD*T*hh*:*mm*:*ss*

YYYY represents the year, MM the month and DD the day of the month, and hh represents the hour, mm the minutes and ss the seconds.
A valid example is “2005-12-24T12:23:45”.

An additional synchronization-source-descriptor allows to specify a short name indicating, which source is used for synchronization. This e.g. can be the type of the GPS used.
At instrument start up, the time is synchronized to the scanner internal real time clock, the synchronization-source-descriptor is set to “RTC” (for **R**eal-**T**ime-**C**lock).
With each successful synchronization, the synchronization-source-descriptor is set to “UNKNOWN” and should be set afterwards to a describing name by the controlling computer.

Command	*Meaning*
SDT_ON=n<CR>	n=0: deactivate the feature “Time Synchronization to GPS”. n=1: activate the feature “Time Synchronization to GPS”
SDT_SET_EPOCH_SYNC=*string*<CR>	Wait for the next GPS pulse (1 PPS). If the pulse is detected within 1.31072 seconds, it is a correct synchronization event. With this event • the SyncTimer and the SyncCounter are set to 0 • the *string* is taken as new epoch data string, where *string* is a null terminated 8 character string • the flag “SYNC_NEVER_EXECTUED” in the synchronization flag byte is cleared. • the descriptor SDT_SOURCE_DESC is set to “UNKOWN”

Command	*Meaning*
	If no pulse is detected, the instrument returns "?SDT_SET_EPOCH_SYNC=*string*", indicating that synchronization failed. The epoch string is supplied in the header of the scan data.
SDT_SET_EPOCH=*string*<CR>	Setting the epoch without waiting for a GPS pulse. On reception of this command, • the SyncTimer and the SyncCounter are set to 0 • the *string* is taken as new epoch data string, where *string* is a null terminated 8 character string • the descriptor SDT_SOURCE_DESC is set to "UNKOWN" This synchronization can be used e.g. to synchronize to the time of the controlling computer (PC), used without pulses. The time calculation further is based on the internal scanner time clock, but synchronized once to the external time source. The epoch string is supplied in the header of the scan data.
SDT_SOURCE_DESC=*string*<CR>	Set the synchronization-source-descriptor to *string,* where *string* is a null terminated 8 character string. The synchronization-source-descriptor-string is supplied in the header of the scan data. The synchronization-source-descriptor is automatically set to • "RTC" (real time clock) at start up • "UNKNOWN" after successful synchronization • "UNSYNC" if the optional "Time Synchronization to GPS" - feature is not supported.
.SDT_SYNC_FLAGS<CR>	Query the synchronization status of the GPS Bit 7: TIME SYNC NOT SUPPORTED 1 = "Time Synchronization to GPS" not supported or currently switched off by command "SDT_ON=0" Bit 6: SYNC_NEVER_EXECTUED 0 = at least 1 synchronization successful executed 1 = no successful synchronization executed Bit 5: LAST_PPS_MISSED 0 = last GPS pulse detected, correctly synchronized 1 = last GPS pulse missed Bit 4: ANY_PPS_MISSED 0 = never missed a GPS pulse since last synchronization 1 = missed at least one GPS pulse since last synchronization Bit 3: TIME_SYNC_DEACTIVATED 1="Time Synchronization to GPS" deactivated by command "SDT_ON=0"

Command	*Meaning*
	The data is identical with the data field SyncFlags in the trailer supplied with each scan line.
SDT_SYNCHECK_OFF<CR>	Following the synchronization to the GPS by command SDT_SET_EPOCH_SYNC, the PPS pulse synchronized timer is compared to an internal timer during operation after each scan line. Loss of PPS pulses are permitted conditions (as several GPS modules temporarily can loss the PPS signal output), but multiple (double) pulses within 1 second are recognized as invalid and cause an error. This command switches off the PPS signal check and it is automatically re-enabled with the next synchronization by command SDT_SET_EPOCH_SYNC

If the optional "Time Synchronization to GPS" – feature is not supported or switched off by command "SDT_ON=0",

- no automatic increment of SyncCounter after 1.31072 seconds. SyncCounter and SyncTimer functions are compatible to older instruments, SyncTimer overflows after 167.77216 seconds and is cleared with an external pulse; SyncCounter counts the pulses.
- at instrument start up, the epoch string is set according to the internal real time clock (as with supported feature).
- the synchronization-source-descriptor is set to "UNSYNC".
- Synchronization flag byte indicates "TIME SYNC NOT SUPPORTED", other flags are not updated. If the feature is deactivated by the command "SDT_ON=0", the bit "TIME_SYNC_DEACTIVATED" is set additionally.

2.7 Scan Sync - Optional Rotation Synchronization

As an option, the scanner provides an operating mode, where the scanner rotation and therefore the resulting scan lines are synchronized to a reference pulse, fed to the scanner via the trigger input.
This option allows **phase synchronization**, or in other words, the user can specify the time of start of a scan line with respect to the reference pulse, which automatically implies an **integer number of lines per second LPS_INT** (see figure below: time between start of scan (1) and GPS-1PPS pulse edge)

$$LPS\_INT = \mathrm{int}\left(\frac{SCN\_THETAD * SCN\_RATE * 3}{2 * 3600000}\right)$$

where LPS_INT is calculated from the scan pattern values SCN_THETAD and SCN_RATE set (see chapter 2.3.5.2 Scan Pattern Commands). This integer value for the resulting lines per second is realized by keeping the requested angular distance between the laser shots SCN_THETAD, but adapting the laser pulse rate

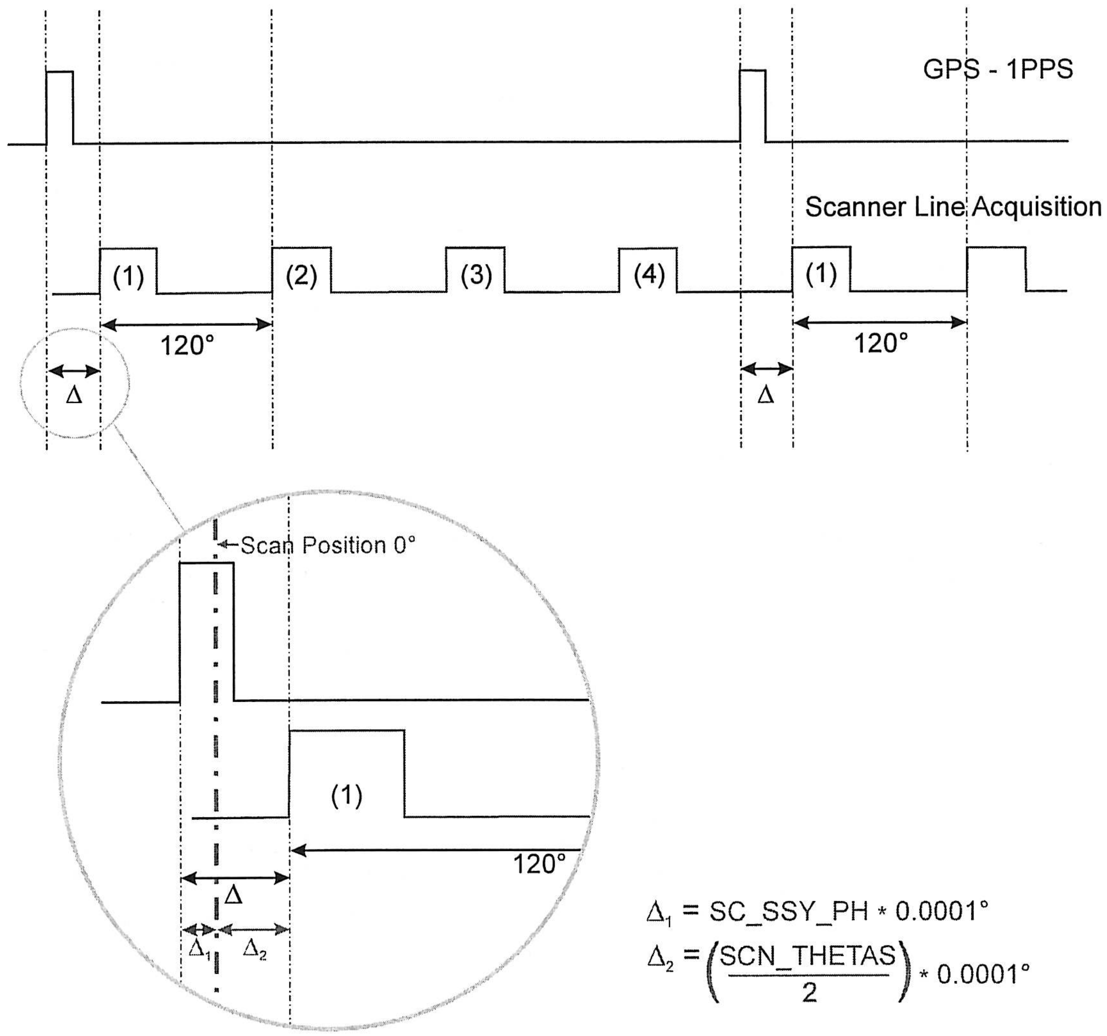
$$\#SC\_SSY\_PRR\_ADAPTED = \left(\frac{2 * 3600000 * LPS\_INT}{3 * SCN\_THETAD}\right)$$

> Note: If the resulting laser pulse rate falls below a critical minimum value, the value “LPS_INT+1” instead of “LPS_INT” is used in the formula above, resulting in a laser pulse rate a little bit higher than the requested laser pulse rate SCN_RATE.

Scan Sync expects an **external reference pulse with a period of 1 second** as typically supplied by GPS modules (PPS-pulse).
This GPS-PPS can be used both to synchronize the scan motion in Scan Sync as well as synchronize the *SyncTimer* to provide GPS synchronized time stamps (see chapter 2.6 Optional Time Synchronization to GPS)

Scan Sync can either be used

- to combine 2 or even more scanners and synchronize the scan motions to avoid interference between the lasers or
- to control the start of scan lines in a system using different sensor components based on GPS time stamping.



$$\Delta_1 = SC\_SSY\_PH * 0.0001°$$

$$\Delta_2 = \left(\frac{SCN\_THETAS}{2}\right) * 0.0001°$$

Fig. 6 Synchronization of a scanner to external PPS pulse, 4 lines per second, synchronization on positive edge of PPS (TRIG_INV=0); as Δ is approx. 40 degree in this figure and assuming SCN_THETAS is 500000, then SC_SSY_PH is approx. 15 degree.

The scanner provides the following commands to configure the Scan Sync operation mode:

Command	*Range*	*Meaning*
SC_SSY_PPS_ENBn<CR>	0 ≤ n ≤ 1	Deactivate (0) or activate (1) the Scan Sync mode. Needs Super user password. Deactivated: • Standard operation • No PPS pulse needed Activated: • Scan motion synchronized to PPS pulse • Resulting lines per second is integer value (automatic adapt of laser pulse rate) • PPS pulse needed permanently via trigger input
SC_SSY_PHn<CR>	0 ≤ n ≤ 1200000	Set the desired angular phase of the first scan within a second with reference to the PPS pulse, scaled in units of 0.0001 degree. With setting TRIG_INV = 0, the reference is to the positive edge of the PPS pulse, with setting TRIG_INV = 1 to the negative edge of the PPS pulse. NOTE: Angular scaling is based on polygon mirror wheel position: 120 degree angle range on polygon mirror wheel correspond to 240 degree angle range covered by the laser beam.
.SC_SSY_PHDIF <CR>		Query the difference between current phase and phase set point, scaled in units of 0.0001 degree. NOTE: Angular scaling is based on polygon mirror wheel position: 120 degree angle range on polygon mirror wheel correspond to 240 degree angle range covered by the laser beam.
.#SC_SSY_PRR_ADAPTED<CR>		Query the automatic adapted laser pulse rate to result in an integer value of LPS, see descriptions above.
SC_SSY_PPSJITn<CR>	10 ≤ n ≤ 500	Set the maximum accepted jitter of the PPS, scaled in per mille. The detection time of the PPS pulse is internally watched and compared to internal timers. Pulses out of an exact 1 second sequence or bad double pulses can be recognized and are indicated by an error. A jitter specified by this setting is accepted.
SC_SSY_PPSMISSn<CR>	1 ≤ n ≤ 10	Set the maximum accepted number of CONSECUTIVE misses of PPS pulses. Experience shows, that some types of GPS modules have temporarily losses of the PPS signal output. This setting enables a limited acceptance of this circumstance.

Some notes for operation in synchronized mode:

- The execution time of SCN_APPLY with mode Scan Sync enabled (SC_SSY_PPS_ENB=1) is enlarged, as SCN_APPLY waits until the scan motion is phase locked to the external PPS signal. If the scan cannot be locked within a certain timeout (typically 15 seconds), an error "SCAN_SYNC_CANNOT_LOCK_PHASE" occurs.
    - SCN_APPLY checks the GPS pulse when executed and returns with a "scan apply error", which can be queried by .SCN_APPLYERR:
      - if no PPS pulse can be detected at all
      - if a PPS pulse is out of the 1 pulse per second sequence (e.g. 2 pulses per second)
      - if the resulting scan speed gets too low or too high when modifying the LPS value to an integer as described above.
- When already scanning, the PPS pulse is further checked during operation. If the jitter of the pulse gets too high (parameter SC_SSY_PPSJIT) or extra pulses are detected, an error occurs. Additionally, if too many consecutive PPS pulses are missed (parameter SC_SSY_PPSMISS), an error indicates that.

2.7.1 Synchronization of 2 Scanners

Synchronization of 2 scanners is shown in the following picture:

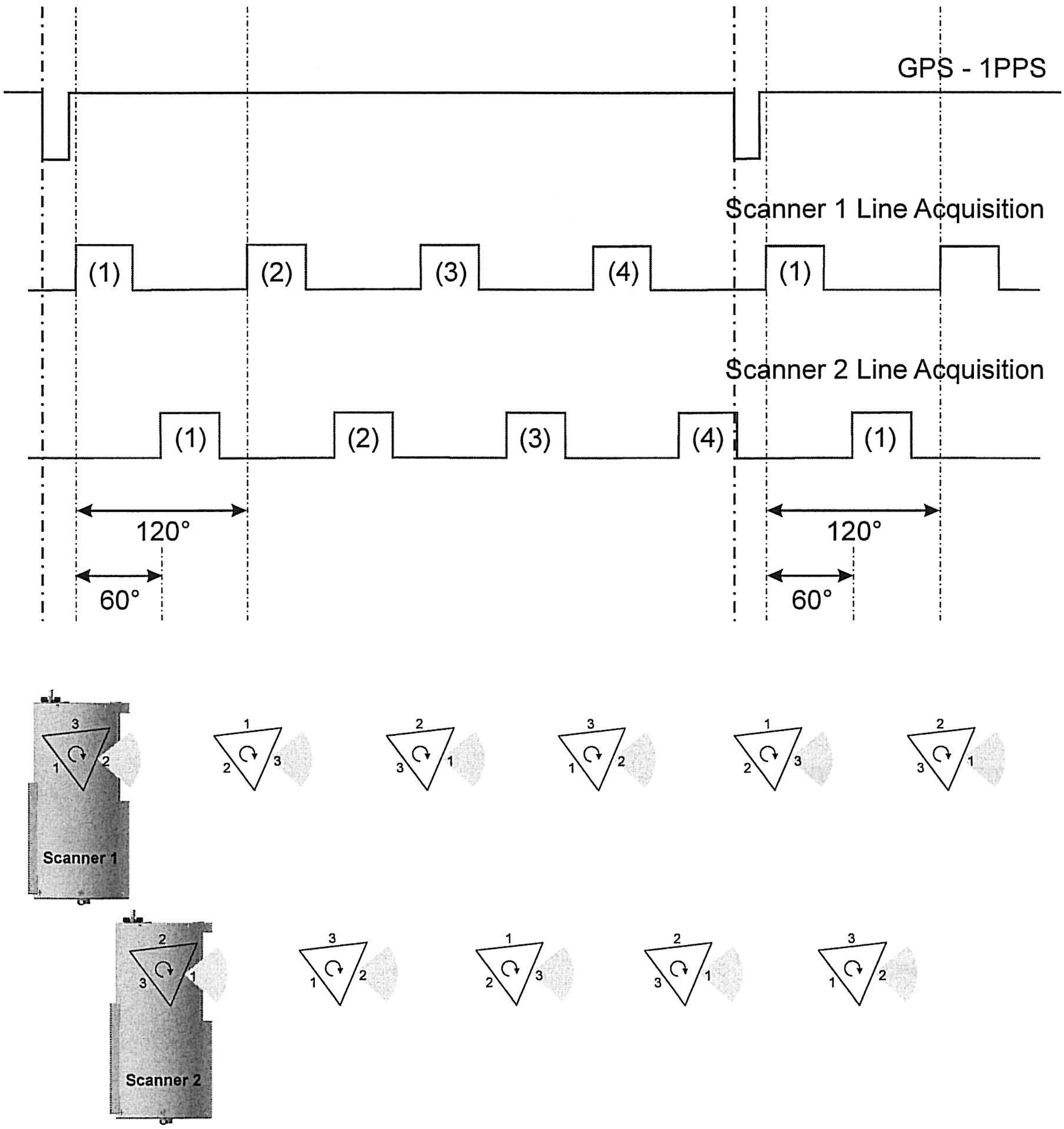


Fig. 7 Synchronization of 2 scanners to external PPS pulse, 4 lines per second, synchronization on negative edge of PPS (TRIG_INV=1), phase set point for scanner 1 is approx. 0 degree (SC_SSY_PH=0 and SCN_THETAS=500000), phase set point for scanner 2 is approx. 60 degree (SC_SSY_PH=600000 and SCN_THETAS=500000), phase difference between both scanners is 60 degree

Example:

- Two scanners should operate with identical scan pattern, but with a phase shift of 60 degree (on polygon mirror wheel), so the scanner 1 acquires a scan line while scan 2 is in a gap between 2 lines and vice versa.
- Scanning range 80 degree.
- Requested angular step width between consecutive laser shots is 0.1 degree.

SCN_THETAD=1000
=> LPS_INT = integer(12.5) = 12
=> #SC_SSY_PRR_ADAPTED = 28800; the laser pulse rate is modified to 28800 Hz

Commands for scanner 1:

```
SC_SSY_PPS_ENB=1
SCN_RATE=30000
SCN_THETAD=1000
SCN_THETAS=500000
SCN_THETAN=801
SC_SSY_PH=0
SCN_APPLY
```

Commands for scanner 2:

```
SC_SSY_PPS_ENB=1
SCN_RATE=30000
SCN_THETAD=1000
SCN_THETAS=500000
SCN_THETAN=801
SC_SSY_PH=600000
SCN_APPLY
```

2.8 Errors and Error Handling

The Scanner knows 4 different types of errors:

- **Fatal Errors**
- **Standard Errors with Requested User Intervention**
- **Standard Errors**
- **Warnings**

After power up the system checks and prepares several internal hardware components. Errors are pushed onto an internal error stack. Therefore it can happen, that more than 1 error is reported.

A **fatal error** causes a lock of measurement and in *Programming mode* only few commands are available (e.g. .HELP, .ERR, Q). The laser is switched off. Fatal error messages always start with “FATAL:”

A **standard error with requested user intervention** like a standard error locks the measurement; *Programming mode* is available and error messages start with “ERROR:”. These types of error need a user intervention, e.g. with a “high temperature error” the instrument must be cooled down or with a “low voltage error” the power supply voltage must be increased (perhaps battery changed).

A **standard error** causes a lock of measurement, but *Programming mode* is available. Standard error messages always start with “ERROR:”

Warnings are reported, but measurement can be (or is) continued. Warning messages always start with “WRNG:”

Errors occurred are pending until they are acknowledged with command “**ERRACK**”. When errors are acknowledged, measurement can be continued. Fatal errors can not be acknowledged.

In *Measurement mode* errors are **reported automatically as messages via the *Control port*** (e.g. “mERROR:LOW_BAT”) . In *Programming mode* an error is indicated by an exclamation mark added to the reply string before the carriage return. To get the error message, the command “.ERR” must be sent to receive a list of errors occurred:

Example:

Command	Reply	Meaning
W<Cr>	*W!	An error occurred
.ERR	\FATAL:FLASH_RW =ERR	

Command	*Meaning*
.ERR<Cr	Reading a list of all errors occurred. Each error message of a pending error is coded in a line starting with “\”.
ERRACK< Cr>	Error acknowledgement. No further errors are pending after this command is executed. Fatal errors cannot be acknowledged.
.ERRCNT<Cr>	Reading the number of errors occurred.
.ERRTYPE[n]<Cr>	Reading the error type of occurred error number n, starting count of n with 0. Therefore .ERRTYPE[0] returns the error type of the first (or only one) error pending. **Type(severity)** — **Error** 1 — Standard error 2 — Standard error with requested user intervention 3 — Fatal error
.ERRNUM[n]<Cr>	Reading the error code number (see table of error messages following, column “Err No”) of occurred error number n, starting count of n with 0.
.ERRSEV<Cr>	Reading the severity of the most severe error occurred.
.ERRMSG[n]<Cr>	Reading the error message (see table of error messages following) of occurred error number n. Note: E.g. n=0 means message of the first error occurred and not the error message of error with error code number 0.

A list of errors is given on the next pages. If fatal errors occur, please contact your sales representative.

2.9 Status and Error Messages

Error	Err No	Meaning
mERROR:TOO_MANY_ERRORS	1	Too many errors (>16), error stack overflow, not all errors can be reported
mWRNG:TOO_MANY_WARNINGS	2	Too many warnings (>16), not all warnings can be reported
mFATAL:FLASH_RW	11	Error in reading or writing flash memory
mFATAL:BAD_ECP	12	Error on *Data port* hardware module
mFATAL:LCA_DATA mFATAL:LCA_INIT mFATAL:LCA_DONE	13 14 15	Error in internal hardware
mFATAL:BAD_MEAS_HW	16	Error in internal measurement hardware
mERROR:FLASHR_BAD_CS	21	Error in reading permanently saved parameters, a check-sum error was found (but data was read!)
mFATAL:FLASHR_NOWRITE mFATAL:FLASHR_BAD_BACKUP	22 23	Error in reading permanently saved parameters.
mFATAL:FLASHR_BAD_FACT_ID	24	Factory parameter data is not fitting to the current software version. Factory data is not valid, but essential; cannot continue!
mERROR:FLASHR_BAD_USER_ID	25	User parameter data is not fitting to current software version. Some parameter settings may be wrong. Check all user parameters, reset them and save them again with command W
mFATAL:FLASHW_FILE_TOO_LONG mFATAL:FLASHR_BAD_TYPE	26 27	Could not save parameters permanently.
mERROR:SCAN_NO_MOTION	28	Error, frame scanner does not move.
mERROR:SCAN_COMMUNICATION	29	Error in communication between range finder unit and scanner unit. See parameter SC_ERROR for detailed reason of error
mERROR:SCAN_STATUS	30	Error in scanner unit. See parameter SC_STATUS for detailed reason of error
mWRNG:RS232_OVERFLOW mWRNG:RS232_OVERRUN	41 42	Serial interface error: some characters are lost in receiving. Possible reason: Commands are sent while data is saved with command W
mERROR:LOW_TEMP	45	The instrument internal temperature is too low
mERROR:HIGH_TEMP	46	The instrument internal temperature is too high
mERROR:LOW_BAT	47	The supply voltage is too low

Error	Err No	Meaning
mERROR:HIGH_BAT	48	The supply voltage is too high
mERROR:BAD_SLOPE	50	Internal error: Bad essential parameter for hardware configuration
mERROR:MEAS_BUFFER_OVERFLOW	51	Internal error: Buffer overflow, laser pulse rate (measurement rate) too high. For scanners: lower the laser pulse rate by increasing the *Line angle step width*.
mERROR:PARAMETERS_OUT_RANGE: n	52	n saved Parameters have been outside range, these parameters have been set to default.
mRGB_BAD_OFFSET_CAL	53	Error in RGB sensor (true color channel): could not calibrate correctly
mFATAL:BAD_STREAM_ID	54	Internal Error in downloading hardware specific data
mFATAL:SSL_TEMP_OUT_RANGE	57	Solid state laser types only: temperature out of range
mFATAL:SSL_SHUTTER FAILED	58	Solid state laser types only: shutter function failed
mFATAL:SID_RW_FAILED	70	Internal Error in Serial Interface Device
mERROR:PRR_OUT_OF_RANGE	71	Solid state laser types only: Laser Pulse rate out of range
mFATAL:PELTIER_TEMP_DEFECT	72	Internal heater/cooler temperature sensor defect
mFATAL:PELTIER _DEFECT	73	Internal heater/cooler mechanism defect
mERROR:PRR_INIT_OUT_OF_RANGE	74	Laser pulse repetition rate outside range
mFATAL: ECP_OVERFLOW	79	Internal critical *Data port* error
mWRNG:ECP_PORT_OPEN	82	Several parameters (e.g. the number of measurements per line RF_NUMBER_L) are displayed in the *Data port* Header (see chapter 2.4.2.1). When changing such parameters, the ECP parallel port must be closed, than parameters changed and *Programming mode* left, and the *data port* newly reopened, so the new changed header can be read by the user application. Therefore, close the ECP parallel port before changing such parameters.
mFATAL:FLASH_RW_FAILED	83	Writing data to the internal error flash memory failed
mERROR:FLASH_VERIFY_FAILED	84	Internal error in structure of error flash data record. This error may occur when updating instrument firmware - Contact *RIEGL* office or your sales representative in this case.
mERROR:TCPIP_BAD_MAC_ADDR	90	Ethernet MAC address invalid

Error	Err No	Meaning
mERROR:TCPIP_BAD_IP_ADDR	91	IP address invalid
mERROR:TCPIP_NOT_CONFIGURED	92	Internal LAN interface configuration error
mWRNG:TCPIP_DATA_PORT_OPEN	93	Several parameters (e.g. the number of measurements per line RF_NUMBER_L) are displayed in the *Data port* Header (see chapter 2.4.2.1). When changing such parameters, the LAN-*TCP/IP Data port* must be closed, than parameters changed and *Programming mode* left, and the *data port* newly reopened, so the new changed header can be read by the user application. Therefore, close the *Data port* before changing such parameters.
mFATAL:ECP_SELFCHECK_FAILED	97	Error on internal ECP and / or *TCP/IP* hardware: internal self check failed
mFATAL:HTR_EM_LOTEMP	100	Internal heater module error
mFATAL:HTR_HW_FAILED	101	Internal heater module error
mFATAL:LASER_MODULE_DEFECT	102	Internal laser module failed
mFATAL:LASER_MODULE_NOT_READY	103	Internal laser module did not get ready
mFATAL:NO_LASER_REF	104	Internal laser module failed
mFATAL:PRR_OUT_OF_RANGE	105	Internal laser module, unexpected *PRR*
mERROR:SCAN_SYNC_CANNOT_LOCK_PHASE	200	With option Scan Sync only: cannot lock to requested phase
mERROR:SCAN_SYNC_TOO_MANY_PPS_MISSING	201	With option Scan Sync only: More than SC_SSY_PPSMISS consecutive PPS pulses are missed
mERROR:SCAN_SYNC_PPS_JITTER_TOO_LARGE	202	With option Scan Sync only: PPS pulses do not occur in expected 1 second time distance: - jitter too large - more than 1 pulse per second - invalid double or multiple
mERROR:GPS_PPS_SEQUENCE_ERROR	208	With optional Time synchronization to GPS only: PPS pulses do not occur in expected 1 second time distance.

Message	Meaning
m##Q240I## m##Q240##	Startup message Q240i Startup message Q240
mWAITSCAN	Scanner unit is started, wait until the scanner has speed up to its requested speed.
mSCANNER_READY	Scanner is ready for measurement / data acquisition
mSCANNING	Data acquisition started
mWAIT_LANLINK	The instrument checks if a *LAN link* can be found, i.e. if the instrument is connected to a HUB (or PC) via a LAN cable
mLAN_CONTROL	The instrument has detected a *LAN link,* it switches to LAN mode an further communicates via LAN
mLOCAL_CONTROL	The instrument could not detect a *LAN link,* it keeps the LOCAL interfaces active and further communicates via RS232 and ECP.