Installation of SCANLAB XL *SCAN* Components and Initial Operation of the XL *SCAN* System

syncAXIS control V1.8.0

Document info:
Doc. Rev. 2.0.4 en-US

2022-08-25

SCANLAB GmbH
Siemensstr. 2a
82178 Puchheim
Germany

Tel.+49 (89) 800 746-0
Fax+49 (89) 800 746-199

info@scanlab.de
www.scanlab.de

(Doc. Rev. 2.0.4 en-US - 2022-08-25)

Contents

1 About this Manual

XL *SCAN* comprises system components from several manufacturers (SCANLAB, ACS, positioning stage manufacturer).

This manual describes:

- The components delivered by SCANLAB for an XL *SCAN* system:
  scan heads with objective, RTC6 boards, SCANLAB-USB dongle and software for synchronous control of scan head, positioning stage and laser
  (see Chapter 2.3 "Typical Scope of Delivery (SCANLAB-XL SCAN Components)", page 10),
- Other components of a multi-axis scan system which are typically integrated together with the SCANLAB-XL *SCAN* components to form a complete system (see overview in Figure 1, page 8 and Chapter 2.4 "Non-SCANLAB Components (Prerequisites)", page 13),
- The installation and initial operation of the scan system using the supplied "Installation_Project".
  Another purpose of the "Installation_Project" is to make source code available to system operators which demonstrates the implementation of the syncAXIS control software for controlling an XL *SCAN* system (for example, how to create a syncAXIS control instance, how marking patterns may be coded and executed).

Caution!

Read and observe all safety instructions in this manual!

SCANLAB accepts no liability for damages or consequential losses resulting from non-observance of this manual, in particular the safety instructions contained herein.

1.1 Related Documents

- excelli*SCAN* Manual
- RTC6 Manual
  (included in the syncAXIS control-software package, see page 49)
- "syncAXIS-DLL – Application Programming Interface" Manual
- Manual "syncAXIS Viewer"
- Manual "syncAXIS Configurator"
- "AN ACS Components in XL *SCAN* System"
  (published by the ACS and is to be requested there)
- "SLEC EtherCAT Node Installation Guide"
  (published by the ACS and is to be requested there)

1.2 Manufacturer

SCANLAB GmbH
Siemensstr. 2a
82178 Puchheim
Germany
Tel. +49 (89) 800 746-0
Fax +49 (89) 800 746-199
info@scanlab.de
www.scanlab.de

1.3 Glossary

ACS	Designates the manufacturer of machine control systems whose components must be used at present for XL *SCAN*.
API	Abbreviation of Application Programming Interface. Program part (for example, of the syncAXIS-DLL) which is available for other programs for connecting to the system (for example, functions of the syncAXIS-DLL). See "syncAXIS-DLL – Application Programming Interface" Manual.
BIOS	Basic Input/Output System. Is permanently stored in the Flash memory of the RTC6 board.
Buffer	Designates syncAXIS-DLL-internal temporary memories. See "syncAXIS-DLL – Application Programming Interface" Manual.
Buffer underrun	After execution start, the RTC6 board has processed all RTC6 microvector commands in its list memory because the syncAXIS-DLL has been too slow in the calculation and transmission of further RTC6 microvector commands. See "syncAXIS-DLL – Application Programming Interface" Manual, Chapter 2.7.1 "About the Buffers of the syncAXIS control Instances", page 42.
Dongle	See SCANLAB-USB dongle, page 11.
Error Mapping	"Stage error mapping". The static calibration of the positioning stage. (Note, the static calibration of the scan head is the scan head working field calibration.)
Flash memory	Non-volatile memory on the RTC6 board that replaces the EEPROM of the RTC5 board.
Handle	Computer programming term: abstract reference to a resource. In this manual, this term refers to a certain syncAXIS control instance. Its Handle value is assigned by **slsc_cfg_initialize_from_file**, see "syncAXIS-DLL – Application Programming Interface" Manual.
Input buffer	syncAXIS-DLL-internal temporary memory at the start of the processing chain. See Buffer and "syncAXIS-DLL – Application Programming Interface" Manual.
Job	Designates a mandatory sequence of Job functions (= **slsc_list_***).
Multi-Head	Designates an XL *SCAN* system where 1 syncAXIS control instance controls more than one excelli*SCAN* scan head and 1 positioning stage.
Multi-Instance	Designates the (optional) ability to run more than one syncAXIS control instance on a PC at the same time. Requires a Dongle that explicitly supports this option[(a)]. Multi-Instance and Multi-Head are basically compatible[(b)].
Multi-Stage	Designates the (optional) functionality of the syncAXIS control instance to be able to change the positioning stage. Requires a Dongle that explicitly supports this option.
syncAXIS control instance	A software object which is created in the PC-RAM when a valid `syncAXISConfig.xml` is called by a syncAXIS control-based user program. Every syncAXIS control instance can be addressed by a unique Handle. See "syncAXIS-DLL – Application Programming Interface" Manual, Chapter 2.7.1 "About the Buffers of the syncAXIS control Instances", page 42.

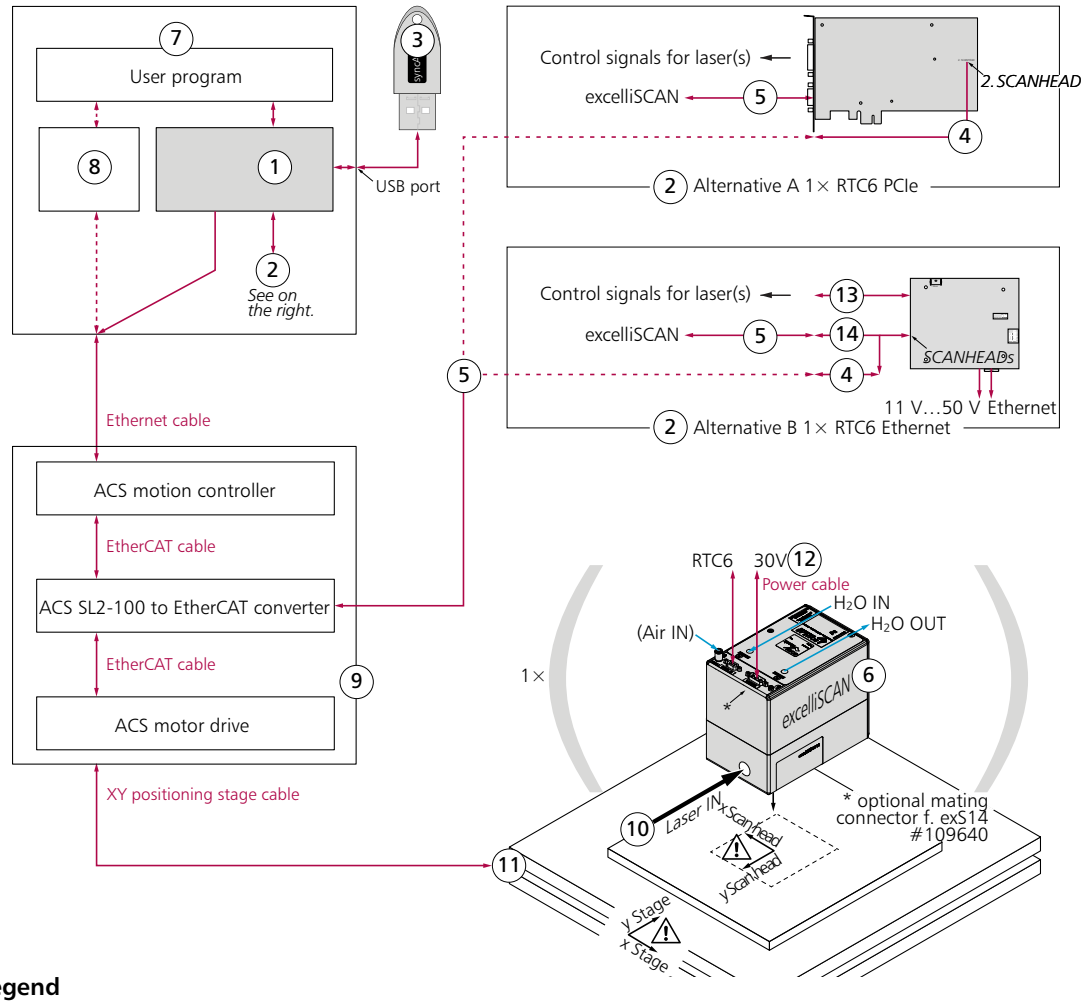
`syncAXISConfig.xml`	"XML configuration file". Although the file name can be freely chosen, it is denominated as "`syncAXISConfig.xml`" throughout this document. The complete tag descriptions can be found in "syncAXIS-DLL – Application Programming Interface" Manual.
Trajectory	Curve with time parameterization.

(a) The number of allowed syncAXIS control instances is coded on the Dongle.

(b) Not compatible with certain special systems where 2 syncAXIS control instances on a single PC control 2 master/slave connected RTC6 boards that feed into the same EtherCAT network.

2 System Overview

2.1 Typical Single-Head-Setup



Legend

a. SCANLAB XL *SCAN* components (SCANLAB's typical scope of delivery):

1 Software (syncAXIS control, RTC driver for Windows, `*.ct5` file)
2 RTC6 PCI Express Board (alternatively an RTC6 Ethernet Board can be used)
3 SCANLAB-USB dongle
4 SSHC slot bracket, see page 47
5 SL2-100 data cable
6 excelli*SCAN*-scan head with objective
13 Cable
14 Y cable

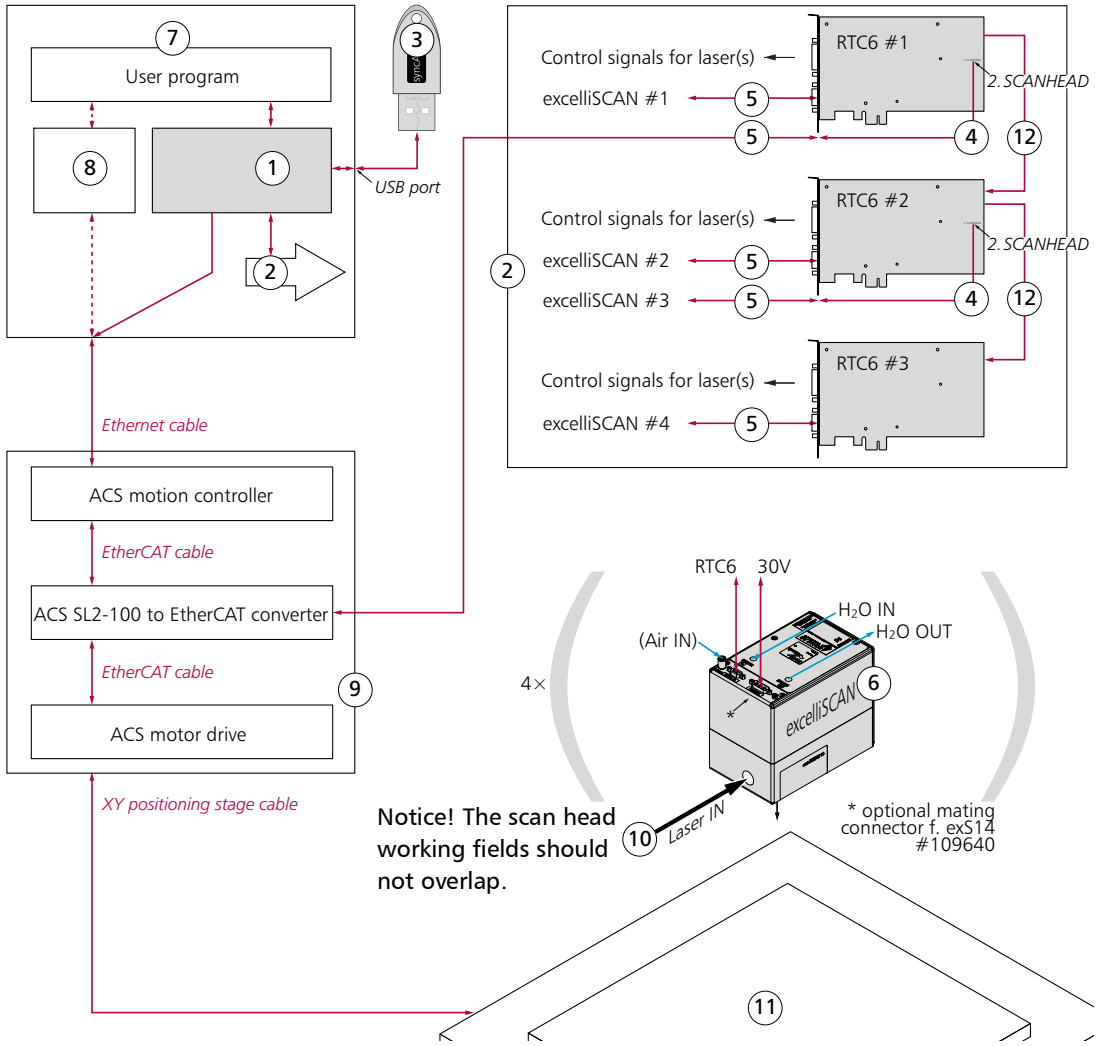
b. Non-SCANLAB components (Prerequisites):

7 Windows-PC and user program
8 ACS software
9 ACS hardware incl. cables
10 Laser incl. cable
11 Positioning stage
12 Power supply

1

Typical Single-Head-Setup. Scan system *after* integration of all XL *SCAN* components (schematic illustration).

2.2 Multi-Head-Setup (4 Scan Heads)



Legend

a. SCANLAB XL *SCAN* components
(SCANLAB's typical scope of delivery):

1 Software (syncAXIS control, RTC driver, `*.ct5` file)
2 RTC6 PCI Express Boards
3 SCANLAB-USB dongle
4 SSHC slot bracket, see page 47
5 SL2-100 data cables
(for example, #115428 = 5 m, not for drag chains)
6 excelliSCAN-scan head (4×), each with objective
12 Master/Slave cable

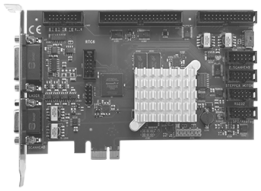
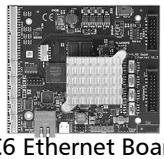
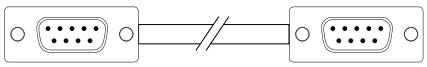
b. Non-SCANLAB components
(prerequisites):

7 Windows-PC and user program
8 ACS software
9 ACS hardware incl. cables
10 Laser incl. cable
11 Positioning stage

2

Typical 4-Head-Setup. Scan system *after* integration of all XL *SCAN* components (schematic illustration).

2.3 Typical Scope of Delivery (SCANLAB-XL *SCAN* Components)

SCANLAB-XL *SCAN* component	For 1-head setup	For 2-head-setup	For 3-head-setup	For 4-head-setup
excelli*SCAN*-scan head  – with objective	1	2	3	4
RTC6 PCI Express Board   RTC6 Ethernet Board – with option “syncA” and option “SCANa”	1	2	2	3
SSHC slot bracket  See also Figure 16, page 47. – #115132	1	1	2	2
SL2-100 data cable[a][b]  – Thereof always 1× for connection RTC6 / SL2-100 to EtherCAT converter[c] – Remaining ones for connections RTC6 / scan head – Each with 2 connectors 9-pin male D-SUB – for example, #115428 = 5 m.	2	3	4	5

(a) Users who do not purchase these cables from SCANLAB must observe page 56.

(b) Currently SCANLAB cables are not for drag chains.

(c) Shortened: “SLEC”.

SCANLAB-XL *SCAN* component	For 1-head setup	For 2-head-setup	For 3-head-setup	For 4-head-setup
Master/Slave cable  – #117241 l = 50 mm or  – #141553 l = 100 mm	0	1	1	2
Cable – #116048 l = 200 mm – Only required with RTC6 Ethernet Board(s), see Figure 1, page 8	1	–	–	–
Y cable – #116050 l = 200 mm – Only required with RTC6 Ethernet Board(s), Figure 1, page 8	1	–	–	–
SCANLAB-USB dongle  – #127456 for 1-head setup. – #139325 for 2-head-setup. – #139783 for 3-head-setup. – #138746 for 4-head-setup.	1	1	1	1
CD  – With syncAXIS control-software package `syncAXIS-<Version>.zip`. Content see Chapter 10 "Appendix C: syncAXIS control Software Package – Unzipped", page 48 – Observe the safety notice on page 12 – With manuals – With `ct5`-correction file fitting the objective	1	1	1	1

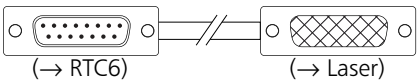
SCANLAB-XL *SCAN* component	For 1-head setup	For 2-head-setup	For 3-head-setup	For 4-head-setup
`syncAXISConfig.xml` – Separate, usually supplied by e-mail – This file is prepared specifically for each customer (that is, according to what the customer has communicated to the SCANLAB sales representative, values for `MotionDecompositionConfig`, marking speed and jump speed are entered). Observe the safety notices in Chapter 3 "Checking and Adapting the syncAXISConfig.xml", page 15.	1	1	1	1

Notice!

Even if the RTC6 board is already in use with certain (previously delivered) RTC6 files independently of syncAXIS control, then the RTC6 boards and syncAXIS control may only be used together with the RTC6 files from the respective syncAXIS control-software package `syncAXIS-<Version>.zip`.

2.4 Non-SCANLAB Components (Prerequisites)

The following components are typically *not* in SCANLAB's scope of delivery for XL *SCAN*, see also Figure 1, page 8. For a complete XL *SCAN* system, they must be available and fulfill at least the prerequisites mentioned in the following table.

Component	Prerequisites
Positioning stage and ACS components (hardware, software, cabling)	• Are installed, configured, ready-to-operate. • The stage error mapping (= "field correction" for the positioning stage) must already have been carried-out (important: *before* the scan head field correction[(a)] – in order that the rotation scan head working field vs. positioning stage working field can be compensated). • Suitable measures must be taken to ensure that the hardware does not execute any control values that could lead to damage.
Laser	• Is installed, configured, ready-to-operate. • Laser safety is ensured.
RTC6-to-Laser cable	• Is available. • Meets the laser manufacturer's specifications. • Connector 1: 15-pin male D-SUB. To be plugged into the RTC6. • Connector 2: as specified by the laser manufacturer. To be plugged into the laser. • Pin-out of connector 1 fits the LASER connector of the RTC6, see Figure 15, page 46. • Intended use: see Chapter 4.5 "Connecting RTC6 Boards(s) with other Components", page 19. • Sketch:  (→ RTC6) (→ Laser) • Note for Multi-head setups To date, XL *SCAN* (syncAXIS control V1.6.0) supports 3 RTC6 boards max. If a 4-head setup is to be operated with 4 lasers, the laser control signals of a board are to be duplicated on the hardware side. The laser control signals of all boards are identical. In general, only one card can be used for laser control provided the signals are split accordingly.

(a) Is described in Chapter 7.2 "Optimization – Calibrating the Scan Head Working Field", page 36.

Sub-System (cont'd)	Prerequisites (cont'd)
Power supply for the excelli*SCAN*- scan head	• Is installed and ready-to-operate (Installation may only be carried out by trained persons who have sufficient knowledge in the safe handling of electrical equipment). • Note: the mating connector for the excelli*SCAN* 14 power connector (#109640[(a)]) is optionally available.
Windows-PC	• Is equipped with an Intel i5 processor or better. • Is also equipped with an SSD. • Is installed, configured, ready-to-operate. • A USB port for the SCANLAB-USB dongle is free. • Depending on the number (see page 10) of RTC6 PCI Express Boards free PCIe slots and 1 or 2 (see page 10) free slots to mount the SSHC slot bracket. • Software development environment with C++ compiler (for example, MS Visual Studio 2015 Professional or later). • Prerequisite, if the "Installation_Project" (see page 48) is to be compiled by the user: CMake[(b)].

(a) Not for excelli*SCAN* 20.

(b) CMake (cross-platform make) files serve to easily generate MS Visual Studio Solution files. CMake (https://cmake.org) is a free and open-source cross-platform programming tool for the development and creating software. Using Cmake, make files and projects for many integrated development environments and compilers can be generated by script files (`CMakesList.txt`).

3 Checking and Adapting the syncAXISConfig.xml

Even though SCANLAB creates (based on information from the system operator concerning the in regards to planned system components) a suitable `syncAXISConfig.xml` and supplies them (in general, by e-mail in addition and separately to the syncAXIS control-software package).

However, the file must never be used to operate the XL *SCAN* system without prior verification and adaptation by the system operator.

This applies in particular if the system operator is not sure that he has correctly and completely communicated all current system components (or their technical specifications) to SCANLAB and/or whether system components were subsequently modified.

Caution!

Despite the limit values entered in the `syncAXISConfig.xml`, syncAXIS control may calculate control values and process speeds *outside* the system limits. Control values can be generated which could lead to property damage and/or personal injury. The system operator is responsible for ensuring that mechanisms are in place[(a)] to prevent the actual execution of harmful drive values.

In addition, a syncAXIS control-based user program must always be executed in simulation mode first (in order to perform a check for exceeding the system limits) before a real marking with laser, scan head and positioning stage is executed with it for the first time.

SCANLAB accepts no liability for damages or consequential losses resulting from non-observance of this manual.

(a) See corresponding safety chapters in "syncAXIS-DLL – Application Programming Interface" Manual.

Caution!

The system operator is responsible for ensuring that all values entered in the syncAXIS control `syncAXISConfig.xml` are meaningful and suitable for the XL *SCAN* system used.

For this purpose, consult the subsystem manufacturers and the corresponding documentations.

Notes

- For syncAXIS control ≥ V1.2, the XML system configuration file `syncAXISSysConfig.xml` is omitted!
- The installation package also contains templates (`syncAXISConfig.Template.xml`, `syncAXISConfig_MultiHead.Template.xml`) for `syncAXISConfig.xml` files. These are provided as a reference and must under no circumstances be used unchanged to operate XL *SCAN* systems.
- In the "syncAXIS-DLL – Application Programming Interface" Manual (in Appendix F: Reference of syncAXISConfig.xml Tags) as well as in the supplied XML scheme (syncAXIS_<major vers'n>_<minor vers'n>.xsd, for example, `syncAXIS_1_8.xsd`), users can find additional support (for example, on allowed value ranges and arguments, inline documentation etc.).

Objectives

- The `syncAXISConfig.xml` is prepared for the next step in such a way that safe execution of syncAXIS control user programs is guaranteed.

Materials

- `syncAXISConfig.xml`[(1)] (and for reference, the XML scheme, see page 15, as well as the "syncAXIS-DLL – Application Programming Interface" Manual)
- xml/text editor or syncAXIS Configurator.

(1) See page 15.

Procedure

(1) Open `syncAXISConfig.xml` (write access) with your preferred xml/text editor.

(2) Go through the contents, check the entered values and make the necessary changes or entries. Among others:
- syncAXIS control safety features[1] (`DynamicViolationReaction` as well as `MonitoringLevel`)
- RTC6 board configuration (serial number, connectors used for scan head and positioning stage)
- usable positioning stage work area
- usable scan head work field
- max. speed, acceleration, jerk of positioning stage
- `ct5`-file(s)

Also observe the corresponding notes in the "syncAXIS-DLL – Application Programming Interface" Manual. There you will also find the documentation of the xml tags.

(3) Save the file.

(1) See Chapter 2.2 "About the SAFE Use of syncAXIS control – General Approach".

4 Installing the SCANLAB-XL *SCAN* Components

In this Chapter:

4.1 Preparing the Non-SCANLAB Components

Prepare the Non-SCANLAB components of the scan system (Positioning stage and ACS components, Laser, RTC6-to-Laser cable and Power supply) as illustrated in Chapter 2.4 "Non-SCANLAB Components (Prerequisites)", page 13.

Also observe "AN ACS Components in XL SCAN System".

4.2 Installing excelli*SCAN*(s)

The installation and establishment of the operational readiness is fully described in the excelli*SCAN* manual, see there. In the installation situation, make sure in particular that:

- the excelli*SCAN* scan head has the specified working distance
- the laser focus is guided on the surface of the marking substrate(1)

Notes

- With setups where the scan heads are moved/positioned it must also be ensured that their hoses for water and air as well as electrical cables are rated for this requirement (for example, drag chain suitability, corresponding material flexibility and bending radii, suitable strain-reliefs and fittings).

(1) See excelli*SCAN* Manual, Chapter 5.3 Adjustment and Alignment.

4.3 Only Applies with RTC6 PCI Express Boards – Extending the Windows-PC with RTC6 Components

Objectives

- The Windows-PC is equipped with the corresponding number of RTC6 PCI Express Board(s) and SSHC slot bracket(s). If several boards are used, these are Master/Slave connected. The RTC6 Driver is installed to Windows.
  - Single-Head-Setup see Figure 1, page 8
  - 4-Head-Setup see Figure 2, page 9

Materials

- Windows-PC
- n× RTC6 PCI Express Board, n see page 10
- n× SSHC slot bracket, n see page 10
- n× Master/Slave cable, n see page 11
- RTC6 driver for Windows (in
  `\---RTC6`
  `|   \---Driver`)
- RTC6 Manual (included in the syncAXIS control-software package, see page 49)

Procedure

(1) Install the RTC6 PCI Express Board(s) and the SSHC slot bracket(s) to the into Windows-PC. If you have several boards, connect them as Master/Slave (Master/Slave cable).
See also RTC6 Manual (included in the syncAXIS control-software package, see page 49).

(2) Plug the SSHC slot bracket connector, see Figure 16, page 47, into the 2. SCANHEAD connector of the respective board:
  - Single-Head-Setup see Figure 1, page 8
  - 4-Head-Setup see Figure 2, page 9

(3) Close the PC housing and start Windows.

(4) Install the RTC6 driver for Windows.

(5) At this point, *do not* install any other software from the RTC6 Software Package. The the RTC6 files `*.dll`, `*.rbf`, `*.out`, `*.dat` are installed later must always be taken from the syncAXIS control-software package (never from the RTC6 Software Package(1)), see Chapter 4.6, page 19.

(1) For syncAXIS control user programs, you must always use the RTC6 files from the syncAXIS control-software package.

4.4 Only Applies with RTC6 Ethernet Boards – Install RTC6 Ethernet Board

Notes

- To date, RTC6 Ethernet Boards are only possible with a 1-head setup, see Figure 1, page 8.
- RTC6 Ethernet Boards do not require the RTC6 driver for Windows.

Objectives

- The RTC6 Ethernet Board is installed (for example, electrical cabinet).

Materials

- 1× RTC6 Ethernet Board, see page 10
- 1× SSHC slot bracket, see page 10
- 1× Cable, see page 11
- 1× Y cable, see page 11
- RTC6 Manual (included in the syncAXIS control-software package, see page 49)

Procedure

(1) Install the RTC6 Ethernet Board as described in the RTC6 Manual (included in the syncAXIS control-software package, see page 49) according to Figure 1, page 8.

(2) At this point, *do not* install any other software from the RTC6 Software Package.
The the RTC6 files `*.dll`, `*.rbf`, `*.out`, `*.dat` are installed later must always be taken from the syncAXIS control-software package (never from the RTC6 Software Package(1)), see Chapter 4.6 "Unzipping syncAXIS Software on the Windows-PC and Plugging in SCANLAB-USB Dongle", page 19.

4.5 Connecting RTC6 Boards(s) with other Components

Objectives

- The RTC6 PCI Express Board is electrically connected to SL2-100 to EtherCAT converter, laser and excelli*SCAN*.
  - Single-Head-Setup see Figure 1, page 8
  - 4-Head-Setup see Figure 2, page 9

Materials

- Windows-PC from "Only Applies with RTC6 PCI Express Boards – Extending the Windows-PC with RTC6 Components", page 18
- 1× SL2-100 data cable
  - For the connection RTC6 / SL2-100 to EtherCAT converter
- n× SL2-100 data cable
  - For the connection RTC6 / scan head, n see page 10
- n× RTC6-to-Laser cable
  - For the connection RTC6 / laser n depends on your requirements

Procedure

(1) Connect RTC6 and SL2-100 to EtherCAT converter.
Use the SL2-100 data cable.
Plug the connector 1 into the SSHC slot bracket connector (see Figure 17, page 47) of the Master-RTC6 and the connector 2 into the corresponding connector of the SL2-100 to EtherCAT converter.

(2) Connect RTC6(s) and excelli*SCAN*(s).
Use the SL2-100 data cable(s).
  - Single-Head-Setup see Figure 1, page 8
  - 4-Head-Setup see Figure 2, page 9
    (other than shown in this drawing excelli*SCAN*#4 could be alternatively connected to the 2. SCANHEAD connector of RTC6-board #3)

Note: The SCANHEAD connector of the RTC6 PCI Express Board is shown in Figure 14, page 46.

(3) Connect RTC6(s) and laser.
Use the RTC6-to-Laser cable(s).
Note: The LASER connector of the RTC6 PCI Express Board is shown in Figure 15, page 46.

4.6 Unzipping syncAXIS Software on the Windows-PC and Plugging in SCANLAB-USB Dongle

Objectives

- The Windows-PC is equipped with the syncAXIS control-software package and the SCANLAB-USB dongle (that is, in all succeeding steps users have access to relevant files such as RTC6 files, "Installation_Project" etc.).

Materials

- Windows-PC from "Only Applies with RTC6 PCI Express Boards – Extending the Windows-PC with RTC6 Components", page 18
- SCANLAB-USB dongle
- syncAXIS control-software package

Procedure

(1) Plug the SCANLAB-USB dongle into a free USB port of the PC.

(2) Unzip the syncAXIS control-software package (`*.zip` file) on the PC (for example, Windows Desktop) to a target folder of your choice.

(3) Get an overview of the contained files. See also "Appendix C: syncAXIS control Software Package – Unzipped", page 48.

5 Starting-up the XL *SCAN* System – Software

In this Chapter:

Notes

- These steps are relevant for all users who want to put the XL *SCAN* into operation.
- These are the only steps to be performed for users who have an "office dongle" and want to become familiar with the system through simulations.

5.1 Preparing `Installation_Project.exe`

Do one of the following:

- Chapter 5.1.1 "Alternative 1 – Generating the "Installation_Project" for MS Visual Studio and Compiling `Installation_Project.exe`", page 20
- Chapter 5.1.2 "Alternative 2 – Using the Delivered `Installation_Project.exe`", page 21

5.1.1 Alternative 1 – Generating the "Installation_Project" for MS Visual Studio and Compiling `Installation_Project.exe`

In this step, the "Installation_Project"(1) is setup (using CMake(2) and the supplied cmake files) in the software development environment and compiled. Furthermore, `syncAXISConfig.xml` (adapted earlier by the system operator, see Chapter 3 "Checking and Adapting the syncAXISConfig.xml", page 15) is made available in the expected folder.
The "Installation_Project" is used for all subsequent steps in this manual.

Objectives

- "Installation_Project" is setup in the software development environment (here: as MS Visual Studio Solution) The `Installation_Project.exe` compiles without errors.

Materials

- Software development environment (MS Visual Studio 2015 Professional or later)
- CMake (supposed installation location: `C:\Program Files\CMake\bin\cmake.exe`)
- Supplied files for CMake in the folder `\---Demo`.
- `syncAXISConfig.xml` from Chapter 3 "Checking and Adapting the syncAXISConfig.xml", page 15

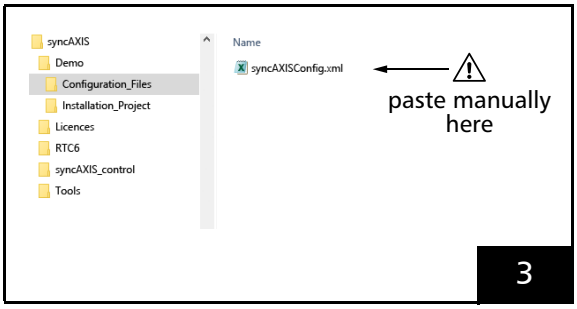
(1) See also page 5.
(2) See footnote on page 14.

Procedure

(1) Copy `syncAXISConfig.xml` from Chapter 3 "Checking and Adapting the syncAXISConfig.xml", page 15 to the folder

```
\---Demo
|   \---Configuration_Files
```

(of the unzipped package, see step 2, page 19), see Figure 3, page 21.



To step 1, page 21: Prior to executing the batch file, `syncAXISConfig.xml` must be present at the indicated location (this path is then automatically entered in `IncludePath.h`, see also `INIT`).

(2) With an editor, open the file

```
\---Demo
|   |   GenerateCMakeTestProject.bat
```

. If necessary, adjust the MS Visual Studio version with the one you are using, for example, "14 2015" to "15 2017".

(3) Execute `GenerateCMakeTestProject.bat`. In the subfolder `Build_syncAXIS`, a "Solution" for MS Visual Studio is created.

(4) Start MS Visual Studio.

(5) In the subfolder `Build_syncAXIS`, open the solution file `*.slu`.

(6) Build the executable `Installation_Project.exe`.

5.1.2 Alternative 2 – Using the Delivered `Installation_Project.exe`

This step is applies to users who do not have a C++ development environment installed on their Windows-PC and, for example, want to use a third-party GUI Solution.

The console application "`Installation_Project.exe`" (see Figure 4, page 22) from the syncAXIS control-software package is being prepared to carry out further steps in this manual.

Objectives

- `Installation_Project.exe` is ready to be executed

Materials

- `syncAXISConfig.xml` from the preceding step
- `Installation_Project.exe` from

```
\---Tools
|   \---syncAXIS_Installation
```

Procedure

(1) Copy `syncAXISConfig.xml` to the directory where `Installation_Project.exe` is located also.

(2) In `syncAXISConfig.xml` under `BaseDirectoryPath`, enter the path from step 1.

(3) In `syncAXISConfig.xml` under `ProgramFileDirectory`, enter the absolute path of

```
\---RTC6
|   \---ProgramFiles
```

.

5.2 Starting `Installation_Project.exe` and Initializing syncAXIS control instance in Simulation Mode

`Installation_Project.exe` is available from the preceding step Chapter 5.1 "Preparing `Installation_Project.exe`", page 20.

In this step, a syncAXIS control instance in simulation mode(1)(2) and operation mode "`ScannerAndStage`" is to be created with it.

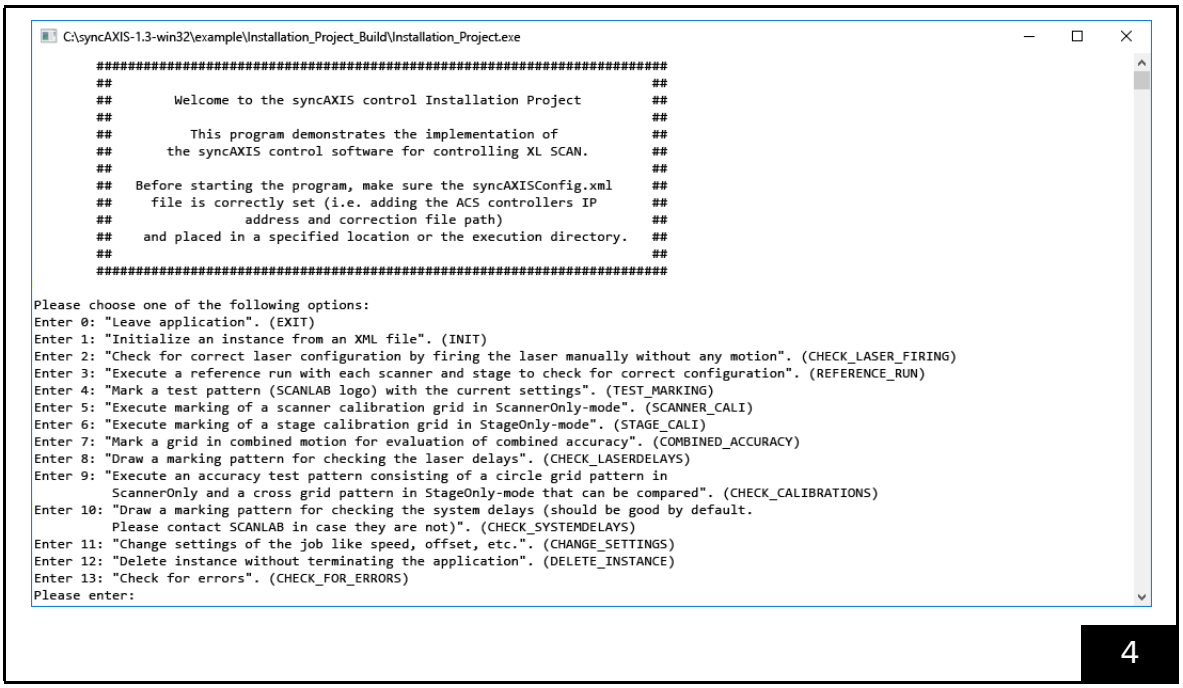
(1) In `syncAXISConfig.xml`:

```
<cfg:SimulationMode>true
</cfg:SimulationMode>
```

If initialization is successful , `Installation_Project.exe` shows "finished with return value 0". Among other things, this means:

- SCANLAB-USB dongle is plugged in and is recognized
- The `syncAXISConfig.xml` is correct on a software-technical level
  - It is valid against the corresponding xml scheme (for example, no negative speed values are entered)

(2) In simulation mode, the executable file does not actually control the laser/positioning stage/RTC6 PCI Express Board. Also the positioning stage communication is not monitored yet.



`Installation_Project.exe` opens as console application.

To create the syncAXIS control instance: enter 1.

To query errors: enter 13.

To terminate the created syncAXIS control instance (without closing this console application): enter 12.

To close this console application: enter 0.

Objectives

- A syncAXIS control instance is created in simulation mode

Materials

- `syncAXISConfig.xml` from the preceding step
- `Installation_Project.exe` from Chapter 5.1 "Preparing `Installation_Project.exe`", page 20

Procedure

(1) Make sure that in `syncAXISConfig.xml` is entered:
- simulation mode(1)
- An IP address for the ACS Motion Controller (however, the format is checked only)
- Path to the `ct5` file

(2) Start `Installation_Project.exe`.
A new window opens, see Figure 4, page 22.

(3) To initialize the syncAXIS control instance (`INIT`), enter:
'1' – return key – ['1' or '2'(2)] – return key.

If successful, the window shows "`Finished with return value 0`".
If not successful, the window shows "`Finished with return value <nn>`".
- To query errors (`CHECK_FOR_ERRORS`), enter:
  - '13' – return key
- To delete the syncAXIS control instance (`DELETE_INSTANCE`), enter:
  - '12' – return key
- Fix the error and repeat this step 3.

(4) To exit (`EXIT`) `Installation_Project.exe`, enter:
- '0' – return key

(1) See Footnote (1), page 22.

(2) '1' if you have had compiled `Installation_Project.exe` yourself (chap. 5.1.1) and '2' if not (chap. 5.1.2).

5.3 Simulating “TEST_MARKING” in Operation Mode “ScannerAndStage”

In the preceding step “Starting Installation_Project.exe and Initializing syncAXIS control instance in Simulation Mode”, page 22 it has been made sure that a syncAXIS control instance can be created successfully by using Installation_Project.exe and the specified syncAXISConfig.xml.

Before the very first execution of any marking pattern with syncAXIS control, the marking pattern must be simulated beforehand (= in this step).

> **Caution!**
>
> Despite the entered limit values in syncAXISConfig.xml, syncAXIS control can, under some circumstances, calculate control values and process speeds that *exceed* the system limits.
>
> Therefore, always initially run syncAXIS control-based user programs in simulation mode (to check for system limit violations) before first-time execution of real-world marking with the laser, scan head and positioning stage.

This way, users can ensure that there are no limit value exceedances (position and dynamic violations) contained in the control values (for this, the generated simulation file is to be visually inspected and evaluated, for example, with the SCANLAB tool “syncAXIS Viewer”, see Figure 5, page 25). With syncAXIS Viewer Affected legs can be highlighted in syncAXIS Viewer (see syncAXIS Viewer-Manual).(1)

Objectives

- A simulation file of “TEST_MARKING” can be created and then opened by syncAXIS Viewer
- It is ensured that the control values do not contain any scan device and positioning stage limit value exceedances

Materials

- Installation_Project.exe from the preceding step
- syncAXISConfig.xml (simulation mode is entered(2), operation mode “ScannerAndStage” is entered(3)) from the preceding step
- SCANLAB tool “syncAXIS Viewer” to display and evaluate the simulation file

Procedure

(1) Start Installation_Project.exe.

(2) To initialize the syncAXIS control instance (INIT), enter:
- ‘1’ – return key – [‘1’ or ‘2’(4)] – return key

(3) To generate the TEST_MARKING simulation file, enter:
- ‘4’ – return key – then the actual object size (in mm, should be a little bit smaller than the scan head working field) – y

(4) To delete the syncAXIS control instance (DELETE_INSTANCE), enter:
- ‘12’ – return key

(5) To exit (EXIT) Installation_Project.exe, enter:
- ‘0’ – return key

(1) The syncAXISConfig.xml supplied by SCANLAB is based on the customer’s specifications for scan device and positioning stage limit values. This “initial” file is created in such a way that none of the Installation_Project.exe marking patterns should cause limit value exceedances (leave the parameter values unchanged!). If you nevertheless observe a limit value exceedance, contact SCANLAB support immediately and do not carry out any marking on the XL *SCAN* hardware setup.

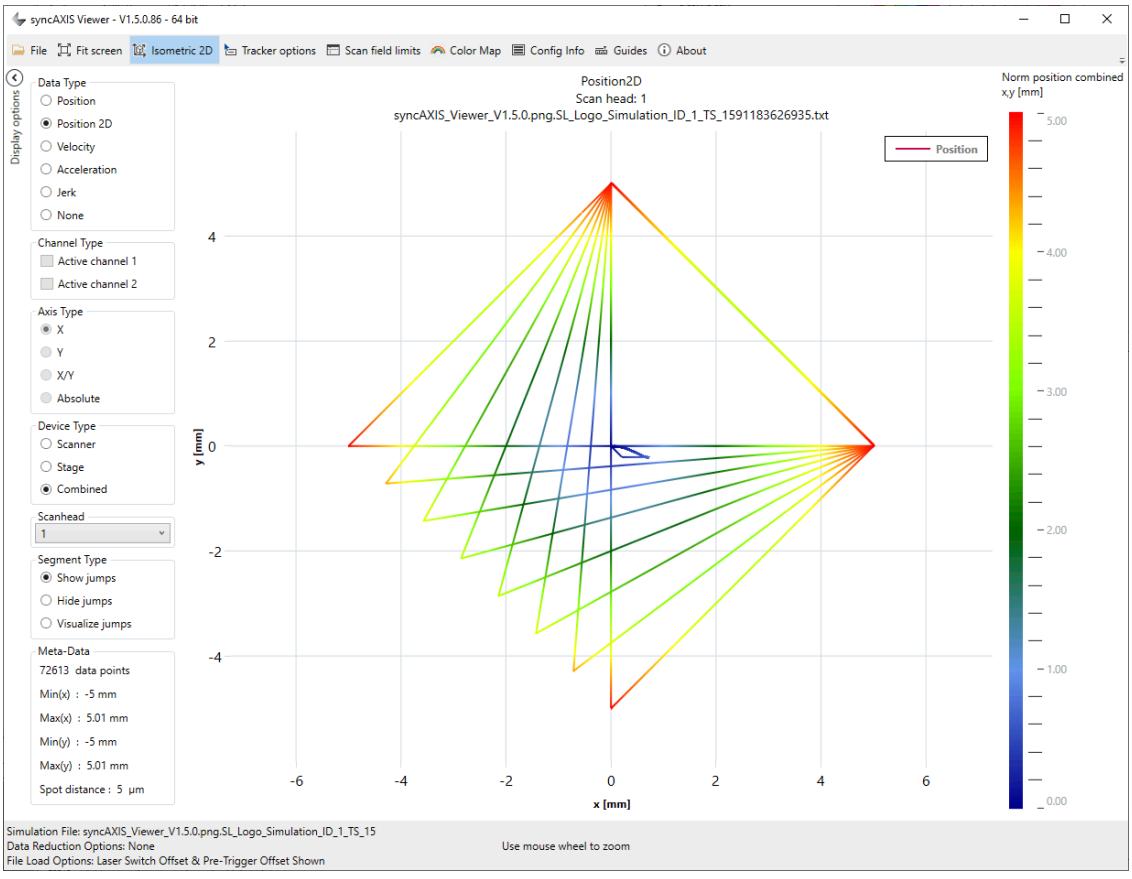
(2) In syncAXISConfig.xml:
<cfg:SimulationMode>true
</cfg:SimulationMode>

(3) In syncAXISConfig.xml:
<cfg:InitialOperationMode>ScannerOnly
</cfg:InitialOperationMode>

(4) ‘1’ if you have had compiled Installation_Project.exe yourself (chap. 5.1.1) and ‘2’ if not (chap. 5.1.2).

(6) Check the simulation file (for example, in "syncAXIS Viewer") for limit value exceedances (position and dynamic violations) in control values. if necessary, adapt the `syncAXISConfig.xml` and repeat this procedure from step 1.
See also "syncAXIS-DLL – Application Programming Interface" Manual, Chapter 2.6 "About Optimizing syncAXIS control-based User Programs", page 36.

Do not proceed to the next step "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26 until you no longer detect any limit violations.



5

SCANLAB tool "syncAXIS Viewer" V1.5 with open simulation file. As plot, 'Position 2D' is set.

6 Starting-up the XL *SCAN* System – Hardware

> **Caution!**
>
> There is a risk of injury from hardware components for laser processing that are involved in the following steps.
>
> Make sure that all these components are ready for operation and that all safety precautions have been taken.

6.1 Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode

In the preceding step "Simulating "TEST_MARKING" in Operation Mode "ScannerAndStage"", page 24 the marking pattern has been simulated and any limit value exceedances have been removed.

Therefore, in this step (with `Installation_Project.exe` and `syncAXISConfig.xml` from the preceding step) you may try to initialize:

- a syncAXIS control instance
  - in hardware mode(1)
  - in operation mode "`ScannerAndStage`"(2)

If the initialization is successful, then the `Installation_Project.exe` window shows: "`Finished with return value 0`".

This means among other things:

- The SCANLAB-USB dongle is plugged in and recognized
- Hardware is correctly configured in `syncAXISConfig.xml`
- Hardware is correctly configured on ACS Motion Controller
- Hardware is correctly cabled
- Hardware is correctly initialized and all components are ready for operation
- The entries in `syncAXISConfig.xml` are correct these are not necessarily "optimal" yet The entries in the `syncAXISConfig.xml` are correct, but not necessarily already ?optimal?

Objectives

- A syncAXIS control instance is created in hardware mode.

Materials

- `Installation_Project.exe` from the preceding step
- `syncAXISConfig.xml` (simulation mode is still entered) from the preceding step

Procedure

(1) Make sure that the `syncAXISConfig.xml` contains the correct values for:

- hardware mode = `<cfg:SimulationMode>false`(1)
- ACS Motion Controller IP address `(ACSController)`
- Path to RTC6 files(`ProgramFileDirectory`)
- Path to ct5 file(s) `(CorrectionFilePath)`
- RTC6 Serial number `(SerialNumber)`

`StageConfig`:

- `StageAxisX`
- `StageAxisY`
- `SlecEtherCATNodeID`, see Figure 6, page 27

(1) In `syncAXISConfig.xml`:
`<cfg:SimulationMode>false`
`</cfg:SimulationMode>`

(2) In `syncAXISConfig.xml`:
`<cfg:InitialOperationMode>ScannerAndStage`
`</cfg:InitialOperationMode>`

(2) Start `Installation_Project.exe`.
A new window opens, see Figure 4, page 22.

(3) To initialize the syncAXIS control instance (`INIT`), enter:
- '1' – return key – ['1' or '2'[1]] – return key

An IP connection is established to the ACS Motion Controller, the positioning stage and scan device are acquired and the RTC6 board is addressed.

If successful, the window shows "`Finished with return value 0`".
If not successful, the window shows "`Finished with return value <nn>`".
See adjacent Notes.

– To query errors (`CHECK_FOR_ERRORS`), enter:
  - '13' – return key

– To delete the syncAXIS control instance (`DELETE_INSTANCE`), enter:
  - '12' – return key

– Fix the error and repeat this step 3.

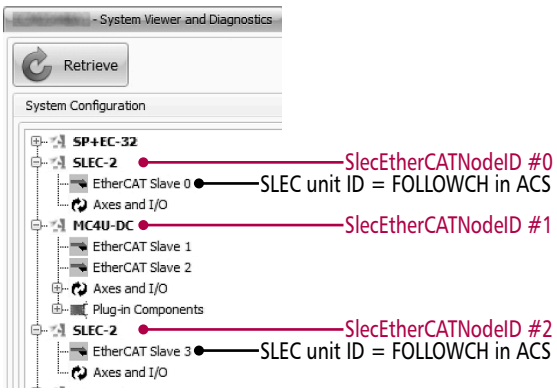
(4) To exit (`EXIT`) `Installation_Project.exe`, enter:
- '0' – return key

(1) See Footnote 4, page 24.

Notes

- Errors often occur in this step "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26. If "`Finished with return value 0`" is not shown, then usually there is a configuration or cabling error:
  – A common error is the confusion of `SlecEtherCATNodeID` and `Slec unit ID`. For identification, see Figure 6, page 27. `SlecEtherCATNodeID` is specified in the `syncAXISConfig.xml` and may have a different value than the `Slec unit ID`. The `Slec unit ID` is used on the ACS Motion Controller in the `FOLLOWCH` variable.
  – Re-check the wiring.
  – Make sure that all devices are live.
  – Make sure that EtherCAT communication exists between the Windows-PC and the ACS Motion Controller.
  – Perform a power cycle with the RTC6 board, in case there is a corresponding error.
- Frequently occurring error messages and their interpretation are shown in Chapter 14 "Appendix G: About Error Handling with syncAXIS control", page 59 and "syncAXIS-DLL – Application Programming Interface" Manual, Chapter 2.7.1 "About the Buffers of the syncAXIS control Instances", page 42.

ACS SPiiPlus MMI Application Studio



6

ACS SPiiPlus MMI Application Studio > System Viewer and Diagnostics: identifying `SlecEtherCATNodeID`s and `Slec unit ID`s. See also Notes, page 27. For further information, refer to "AN ACS Components in XL SCAN System" and "SLEC EtherCAT Node Installation Guide".

6.2 Verifying Whether Laser Outputs Radiation

In the preceding step Chapter 6.1 "Starting Installation_Project.exe and Initializing syncAXIS control Instance in Hardware Mode", page 26 a syncAXIS control instance in hardware mode(1) has been successfully initialized with the Installation_Project.exe and the syncAXISConfig.xml.

In this step 6.2, the laser is to be controlled by this syncAXIS control instance. CHECK_LASER_FIRING is used for this purpose.

After the laser has been activated, it keeps emitting until it is deactivated again. Therefore, make sure in advance that all safety precautions are taken.

If the laser actually emits after its activation, this means:

- Laser and RTC6 board are correctly connected
- In the syncAXISConfig.xml at least TTL level, power, pin assignment and pulse generation are defined "correctly" (insofar as laser radiation is actually outputted)
- The communication to the RTC6 board is working

After it has been determined that the laser emits radiation, CHECK_LASER_FIRING can (optionally) also be used on this occasion to optimize power value and pulse frequency value (so that there will be a clean workpiece material processing later on).

Objectives

- Activate the laser (via the RTC6 board) with syncAXIS control in Mode "Manual Positioning".
- Optionally: In addition, adjust power values and pulse rate values so that "good" marking results can be achieved.

Materials

- Installation_Project.exe from the preceding step
- syncAXISConfig.xml from the preceding step
- Marking substrate (incl. spares)

(1) See Footnote 1 on page 26.

Notes

- By CHECK_LASER_FIRING, neither scan head nor positioning stage are moved.

Prerequisites

- All subsystems (for example, RTC6 board, laser) are ready-to-operate.
- A marking substrate has been inserted.

Procedure

- ⚠ **Warning!** Risk of injury due to laser radiation! Comply with laser safety regulations!

(1) Make sure that syncAXIS control is going to operate in hardware mode(2).

(2) Start Installation_Project.exe and initialize a syncAXIS control instance (INIT) as described in Chapter 6.1 "Starting Installation_Project.exe and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(3) Start CHECK_LASER_FIRING as follows:
- '2' – return key (the Job is started and prompts whether to activate the laser) – then to activate the laser beam, y – Warning! in the direct connection, laser radiation is outputted: return key – then press a key on the keyboard to end the radiation – Loop: you are asked again whether to activate the laser(3); to exit the loop enter: n – return key.

(4) To delete the syncAXIS control instance (DELETE_INSTANCE), enter:
- '12' – return key

(5) To exit (EXIT) Installation_Project.exe, enter:
- '0' – return key

(2) In syncAXISConfig.xml:
<cfg:SimulationMode>false
</cfg:SimulationMode>

(3) Optional at this point: Change power value and pulse rate value in syncAXISConfig.xml and repeat loop.

6.3 Verifying the Correct Communication with Scan Head and Positioning Stage

The preceding step Chapter 6.2 "Verifying Whether Laser Outputs Radiation", page 28 shows that the RTC6 board successfully communicates with the 'first' hardware component, the laser. The communication between the RTC6 board and the laser is the basic prerequisite for proofing the scan head movement.

In this step 6.3, a syncAXIS control instance in hardware mode is used to control the other hardware components scan head and positioning stage.

For this, `REFERENCE_RUN` is executed.

In `REFERENCE_RUN`, the positioning stage first travels a rectangle (size: entered by the user) and then the scan head (size: 10 mm). In the corners of the scan head rectangle, the laser is briefly activated to detect a scan head movement.

These operations are carried out in Mode "Manual Positioning". Therefore, neither Trajectory planning (incl. motion decomposition) nor communication to the ACS components (via SL2-100 to EtherCAT converter) is involved.

Instead, only the correct communication and functionality of the components are checked.

If `REFERENCE_RUN` is executed as described, this means:

- Communication to ACS components via Ethernet TCP/IP works
- ACS axes are ready to move
- In `syncAXISConfig.xml`, `StageAxisX` and `StageAxisY` are correctly defined
- The operation of the excelli*SCAN* works correctly

Ziel

- The correct communication and functionality of the hardware components scan head and positioning stage is confirmed

Materials

- `Installation_Project.exe` from the preceding step
- `syncAXISConfig.xml` from the preceding step
- Marking substrate (incl. spares)

Prerequisites

- All subsystems (for example, the laser) are ready-to-operate.
- A marking substrate has been inserted.

Procedure

- ⚠ **Warning!** Risk of injury due to laser radiation! Comply with laser safety regulations!
- ⚠ **Warning!** Risk of injury due to positioning stage motion! No persons in the danger zone!
- ⚠ **Caution!** Risk of property damage due to positioning stage motion! No foreign objects in the danger zone!

(1) Make sure that syncAXIS control is going to operate in hardware mode[(1)].

(2) Start `Installation_Project.exe` and initialize a syncAXIS control instance (`INIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(3) Start `REFERENCE_RUN` as follows:
- '3' – return key – then the speed with which the positioning stage is to be moved – return key – then the rectangle size to be travelled by the positioning stage – return key – y – return key.
  The Job is calculated for Mode "Manual Positioning" (no Trajectory planning, no motion decomposition).
  The successful completion of the processes is indicated by "`Finished with return value 0`".

(4) To delete the syncAXIS control instance (`DELETE_INSTANCE`), enter:
- '12' – return key

(5) To exit (`EXIT`) `Installation_Project.exe`, enter:
- '0' – return key

(1) See Footnote on page 28.

6.4 Marking the (Previously in 5.3) Simulated "TEST_MARKING"

In the preceding steps it was shown that:

- The hardware is ready for operation
- The system components communicate with each other

The operation still to be performed with syncAXIS control is the combined laser processing by scan head and positioning stage.

In this case, communication with the positioning stage axes is not via Ethernet TCP/IP, but via the SL2-100 to EtherCAT converter. In addition, the list memory of the RTC6 board is filled with RTC6 commands and then executed in real time.

In Chapter 5.3 "Simulating "TEST_MARKING" in Operation Mode "ScannerAndStage"", page 24, you simulated and evaluated the `TEST_MARKING` Job and then ensured that no limit value exceedance will occur during execution. Therefore, in this step 6.4, you now may execute this Job on the XL *SCAN* system.

A successful completion of 6.4 means:

- The hardware is correctly configured
- The hardware is cabled correctly
- The hardware is initialized correctly
- The entries in the `syncAXISConfig.xml` are technically correct, but not necessarily "optimal"

Objectives

- `TEST_MARKING` is actually (= in hardware mode) marked successfully:
  – In operation mode "`ScannerAndStage`"

Materials

- `Installation_Project.exe` from the preceding step
- `syncAXISConfig.xml` from the preceding step
- Marking substrate (incl. spares)

Prerequisites

- All subsystems (for example, the laser) are ready-to-operate.
- A marking substrate has been inserted.

Procedure

- ⚠ **Warning!** Risk of injury due to laser radiation! Comply with laser safety regulations!
- ⚠ **Warning!** Risk of injury due to positioning stage motion! No persons in the danger zone!
- ⚠ **Caution!** Risk of property damage due to positioning stage motion! No foreign objects in the danger zone!

(1) Make sure that syncAXIS control is going to operate in hardware mode(1).

(2) Make sure that syncAXIS control is going to operate in operation mode "`ScannerAndStage`"(2).

(3) Start `Installation_Project.exe` and initialize a syncAXIS control instance (`INIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(1) See Footnote on page 28.

(2) See Footnote on page 26.

(4) Start `TEST_MARKING` as follows:
- '4' – return key – then the actual size of the marking object (in mm, should be smaller than the combined total working field) – y. The Job is calculated for operation mode "`ScannerAndStage`" including Trajectory planning and with motion decomposition. The successful completion of the processes is indicated by "`Finished with return value 0`".

(5) To delete the syncAXIS control instance (`DELETE_INSTANCE`), enter:
- '12' – return key

(6) To exit (`EXIT`) `Installation_Project.exe`, enter:
- '0' – return key

(7) Continue with the following procedure "Evaluation".

Evaluation

(1) If the marking has not been completed, check for error messages. For this, you can use the log file (see `<cfg:LogfilePath>`) as well as the ACS SPiiPlus MMI Application Studio. Observe also Chapter 14 "Appendix G: About Error Handling with syncAXIS control", page 59 and "syncAXIS-DLL – Application Programming Interface" Manual, Chapter 2.7.1 "About the Buffers of the syncAXIS control Instances", page 42.

(2) Perform a visual inspection of the marking result under magnification (for example, with reflected-light microscope):
- If it roughly meets the expectations from the simulation, continue with Chapter 7 "Optimization and Verification", page 32
- If it deviates much from the simulation, check again the correct axis assignment, movement of all components, etc. and perform step 6.4 again.

7 Optimization and Verification

The installation is complete after Chapter 6.4 "Marking the (Previously in 5.3) Simulated "TEST_MARKING"", page 30 since the communication of all components have been successfully demonstrated and first markings have been made.

In order to achieve optimal marking results and successful work with XL *SCAN*, only a few optimization and verification steps are required.

In the steps described below, it is assumed that you are familiar with the handling of `Installation_Project.exe` and that you carry out a simulation before each marking execution for safety reasons.

Required for all steps are:

- `syncAXISConfig.xml`
- `Installation_Project.exe`
- Marking substrates

In this Chapter:

7.1 Optimization – Finding Laser Delays that Suit Your Application

In this step 7.1, the laser delay values are to be optimized:

- `LaserSwitchOffsetTime` – Time shift for the laser signals output)(1)
- `LaserPreTriggerTime` – Time to trigger the laser signal in advance, if a mark segment is executed(1)

For this, `CHECK_LASERDELAYS` is executed. `CHECK_LASERDELAYS` contains an 11 × 11 raster of small marking objects, see Figure 7, page 33. The laser delays are automatically changed between the marking objects:

- `LaserSwitchOffsetTime` is increased in positive x direction
- `LaserPreTriggerTime` is increased in positive y direction

Recommended values to start with (in step 4, page 34) are:

- `LaserSwitchOffsetTime`: –40 µs…+10 µs
- `LaserPreTriggerTime`: –10 µs…+10 µs

(1) See "syncAXIS-DLL – Application Programming Interface" Manual, Figure 41, page 296.

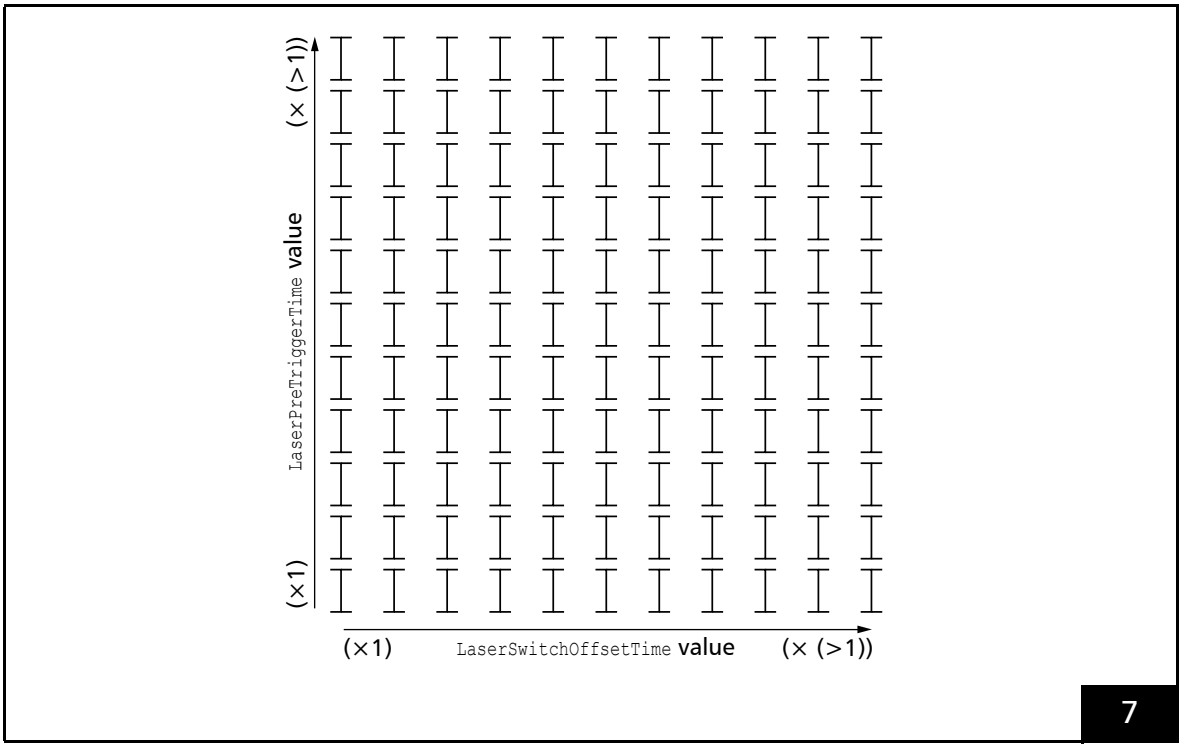
If necessary, repeat step 7.1 several times to reduce the laser delay variation between the marking objects.

After marking, evaluate the result and choose the marked object that best represents your application, Figure 8, page 35. With this selection you determine your `LaserSwitchOffsetTime` and `LaserPreTriggerTime` values (which you can use as start values for another optional run of step 7.1).

Notice!

In this step 7.1 the laser delays are adjusted before the scan head working field calibration is performed in step 7.2. This is important because a good marking result is a prerequisite for a calibration.

Nevertheless, after the scan head working field calibration 7.2 (mirror positioning changes), this step 7.1 should be performed again.



Defined raster in `CHECK_LASERDELAYS`.

Objectives

- Find the optimal `LaserSwitchOffsetTime` value and `LaserPreTriggerTime` value for your application and enter them in `syncAXISConfig.xml`.

Materials

- Marking substrate (incl. spares)

Prerequisites

- The marking patterns must have been first simulated (as always before a very first execution) to ensure that no limit violations are contained in the control values.
- All subsystems (for example, the laser) are ready-to-operate.
- A marking substrate has been inserted.

Procedure

- ⚠ **Warning!** Risk of injury due to laser radiation! Comply with laser safety regulations!
- ⚠ **Warning!** Risk of injury due to positioning stage motion! No persons in the danger zone!
- ⚠ **Caution!** Risk of property damage due to positioning stage motion! No foreign objects in the danger zone!

(1) Start `Installation_Project.exe` and initialize a syncAXIS control instance (`INIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(2) Start `CHECK_LASERDELAYS` as follows:

- '8' – return key – then the max. speed your scan head can achieve (the higher the speed the greater the influence of 'badly' chosen laser delays) – return key – then the `LaserSwitchOffsetTime` start value – return key – then the increment value for `LaserSwitchOffsetTime` – return key – then the `LaserPreTriggerTime` start value – return key – then the increment value for `LaserPreTriggerTime` – return key – y – return key.
  The Jobs are calculated for operation mode "`ScannerOnly`" including Trajectory planning, however, without motion decomposition.
  The successful completion of the processes is indicated by "`Finished with return value 0`".

(3) Exit `Installation_Project.exe` (`DELETE_INSTANCE` > `EXIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(4) Continue with the following procedure "Evaluation".

Evaluation

(1) If the markings have not been completed, check for error messages. For this, you can use the log file (see `<cfg:LogfilePath>`) as well as the ACS SPiiPlus MMI Application Studio. Observe also Chapter 14 "Appendix G: About Error Handling with syncAXIS control", page 59 and "syncAXIS-DLL – Application Programming Interface" Manual, Chapter 2.7.1 "About the Buffers of the syncAXIS control Instances", page 42.

(2) Perform a visual inspection of the marking result under magnification (for example, with reflected-light microscope). It should look similar to Figure 8, page 35.

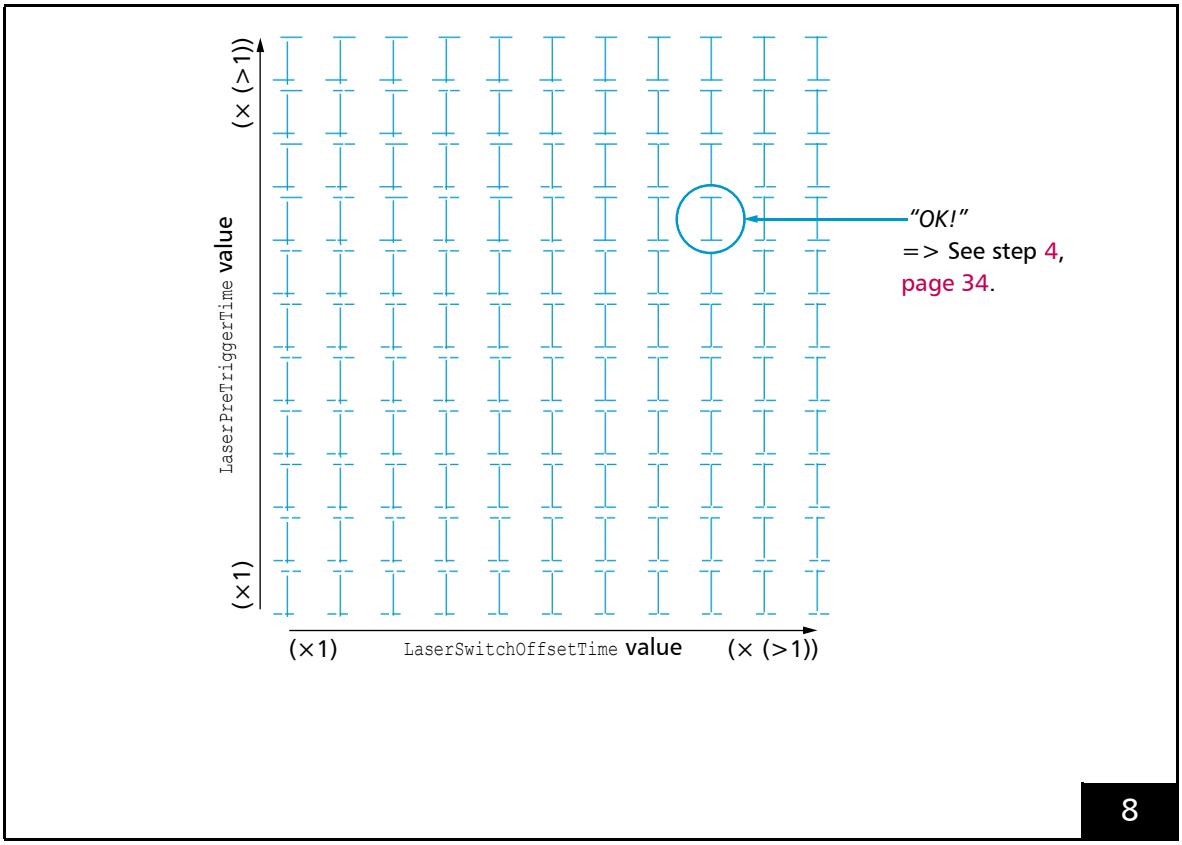
(3) Identify the most suitable marking object for your application in the marking result (Figure 8, page 35). Some relevant criteria are shown in Figure 9, page 35.

(4) Identify the `LaserSwitchOffsetTime` value and `LaserPreTriggerTime` value which has been used for it. Assuming the ideal marking is at position 9 in x direction and at position 4 in y direction, then the following applies:

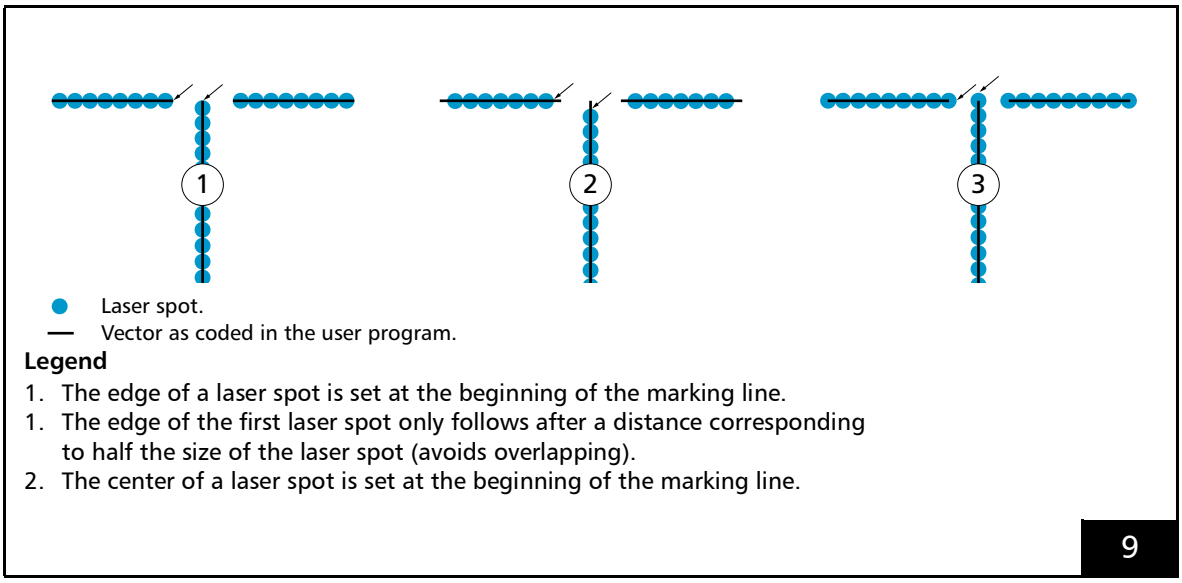
- `LaserSwitchOffsetTime` = $-40\ \mu s + (9-1) \times 5\ \mu s = 0\ \mu s$
- `LaserPreTriggerTime` = $-10\ \mu s + (4-1) \times 2\ \mu s = -4\ \mu s$

(5) Enter these values in `syncAXISConfig.xml`.

(6) Optionally, you can repeat the 7.1 several times with slight variations of these values each time.



Evaluating marking result (example), see page 34 in Chapter 7.1 "Optimization – Finding Laser Delays that Suit Your Application", page 33.



Criteria for marking result evaluation in step 3.

7.2 Optimization – Calibrating the Scan Head Working Field

In this step, the working field of each scan heads is to be calibrated. The objective of this "working field calibration" procedure is to obtain an *optimized* `ct5` file per scan head[1][2].

Here, an approach that uses SCANLAB correXion Pro and a customer's own coordinate measuring machine[3] is described.

For an alternative approach, see Chapter 13 "Appendix F: Calibrating the Scan Head Working Field – Alternative Using CalSheet Software", page 57.

A further "joint" calibration is not necessary.

Basic Process Flow

(1) Meet the Prerequisites, page 37.

(2) Define a raster (for example, a regular array of filled circles; usually 11 × 11, always odd numbers), which extends over the entire scan head working field.
This marking pattern is contained in `SCANNER_CALI` .

(3) Define a single raster object (filled circle).
This marking pattern is contained in `SCANNER_CALI`.

(4) Leave positioning stage at (0,0) mm (working field center = "zero position") and mark the raster only by the scan head (without positioning stage motion).

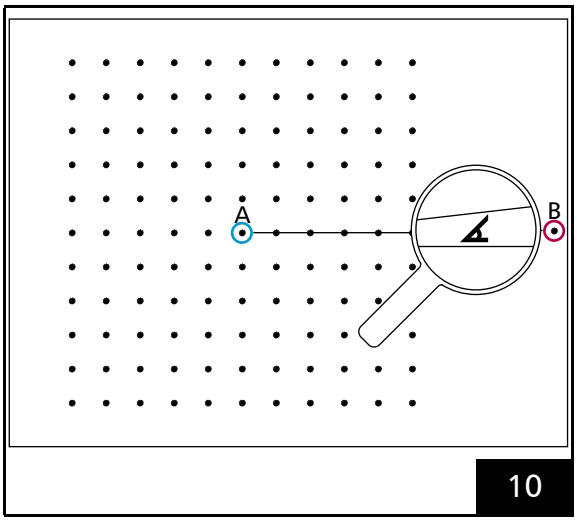
(5) Move the positioning stage (either only in X or only in Y direction) such that the working field center is now outside of the just marked raster.

(6) Mark the single raster object in the working field center.

(7) Measure the marking result, see Figure 10, page 36, using a coordinate measuring machine[4].
Remark: the coordinate measuring machine measures the position of the 11 × 11 raster objects relative to the coordinate system of the positioning stage (as its x-axis, the straight line from "A" to "B" is used).

(8) Create a `dat` file based on the results of the measurement results[5].

(9) Generate *optimized* `ct5` file using correXion Pro.



Marking result in step 1, page 38: typically, scan head coordinate system and positioning stage coordinate system are slightly rotated relative to each other at a certain angle. Therefore, if a non-optimized ct5 file is used, scan head coordinate system center point (A blue) and positioning stage coordinate system center point (B red) are not at the same level. In the optimized ct5 file this angle shall be compensated.
For this, the measuring machine measures the position of the 11 × 11 grid points (=scan head coordinate system) relative to the positioning stage coordinate system (as x axis of the positioning stage coordinate system the line from "A to B" is used).

(1) Essential characteristics of the optimized `ct5` file are:
1. With scan head-working field control, the error is even smaller than with the default delivered `ct5` file (= better scan head working field calibration).
2. It compensates the rotation of positioning stage vs. scan head.

(2) The procedure (Chap. 7.2) should be repeated, if you encounter scan head drift effects (to compensate for these drift effects).

(3) All users who decide to carry out a procedure which is not described in this manual must observe the safety notice on page 38.

(4) The position of the circles can, for example, be determined by the center of gravity in the respective image tool.

(5) See notes in correXion Pro Manual and the template file in the program folder of correXion Pro.

Materials

- A standard `ct5` file for marking the marking patterns.
- Suitable[1] marking substrates (incl. spares).
- A customer's own coordinate measuring machine (must include image processing software). The coordinate measuring machine must be able to capture and to evaluate the marking result.
- correXion Pro incl. manual.

Prerequisites

Prior to performing this field calibration procedure, the following items must be fulfilled:

- Compensation of the positioning stage static error has been carried out (important prerequisite for alignment correction)
- The Error Mapping from ACS for the positioning stage has been applied and is active.
- Beam adjustment and alignment has been performed – see the manual "Installation and Operation" of the scan head, section "Adjustment and Alignment" regarding the values of "tilt" and "displacement".
- Fine tuning of the focal distance and of the laser power parameters[2] and laser delays[3] has been performed. The marking result quality is important for image evaluation. Pay particular attention to the fact that the speed in this step is selected comparatively low, so that the laser radiation input per distance is rather high. Adjust the laser power parameters accordingly.

(1) It must be possible to evaluate the marking result.

(2) See Chapter 6.2 "Verifying Whether Laser Outputs Radiation", page 28.

(3) See Chapter 7.1 "Optimization – Finding Laser Delays that Suit Your Application", page 33.

- Marking must be carried out by using a syncAXIS control-based user program.
- Use relatively low speeds for marking and jumping (for example 0.25 m/s) in order to avoid dynamic following errors as much as possible. The main objective of this procedure is the static scan head working field calibration. Therefore, a high accuracy is necessary (not a high speed).
- All subsystems (for example, the laser) are ready-to-operate.
- A marking substrate has been inserted.

Procedure

- ⚠ **Warning!** Risk of injury due to laser radiation! Comply with laser safety regulations!
- ⚠ **Warning!** Risk of injury due to positioning stage motion! No persons in the danger zone!
- ⚠ **Caution!** Risk of property damage due to positioning stage motion! No foreign objects in the danger zone!

(1) Start `Installation_Project.exe` and initialize a syncAXIS control instance (`INIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(2) Start `SCANNER_CALI` as follows:

- '5' – return key – then the raster size (smaller than the scan head working field size) – the marking is carried out in operation mode `ScannerOnly`) – return key – then the number of raster objects (11×11 are recommended) – return key – then the speed to move the positioning stage – return key – y – return key.
  The Jobs are each calculated for operation mode "`ScannerOnly`" and "`StageOnly`" including Trajectory planning, however, without motion decomposition.
  The successful completion of the processes is indicated by "`Finished with return value 0`".

(3) Exit `Installation_Project.exe` (`DELETE_INSTANCE` > `EXIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(4) Continue with the following procedure "Evaluation".

Evaluation

(1) Measure the marking result, see Figure 10, page 36, using a coordinate measuring machine.

- Select the reference system in such a way that the single point after moving the positioning stage together with the point in the center of the raster results in the reference x-axis for the measurement. In this case, the scan head working field is automatically aligned to the positioning stage working field by the correction file. If the measurement methodology does not provide such a feature, the angle must be determined manually. This must either be included in the evaluation of the measurement data or later used as an alignment matrix (see "syncAXIS-DLL – Application Programming Interface" Manual) in `syncAXISConfig.xml`.

(2) Create a `dat` file based on the measurement results. Information on form and content can be found in the correXion Pro Manual.

(3) Generate your *optimized* `ct5` file with the correXion Pro software and the `dat` file.

Notice!

After this step 7.2 scan head working field calibration (mirror positioning changes), step 7.1 should be performed again.

(4) Make sure that from now on only the optimized `ct5` file from step 3 is used within XL *SCAN*. For this purpose you need to change the `syncAXISConfig.xml` correspondingly, see `<cfg:CorrectionFileList>`.

Notice!

- Important note for users who perform scan head-working field calibration procedures differently than described in this chapter or in Chapter 13 "Appendix F: Calibrating the Scan Head Working Field – Alternative Using CalSheet Software", page 57:
  - You need to determine the "Scan Field Calibration K" value yourself.
  - You need to enter this value into the `syncAXISConfig.xml`[(a)].

(a) At `<cfg:CorrectionFilePath CalibrationFactor = -1.0>`. -1.0 is the default entry and needs to be changed accordingly. For -1.0, the value is read-out from the `ct5` file. However, there might be cases where no/a wrong value is saved in your optimized `ct5` file (for example, because the optimized `ct5` file has been generated using a customer's own tool).

7.3 Verification – Checking the Static Calibration Accuracies of Scan Head and ACS Axes

The following criteria have been processed before:

- The static calibration of the scan head (= scan head working field calibration)
- The static calibration of the positioning stage (= Error Mapping)
- Adjustment of the laser delays

Verification of these calibrations (described in this chapter) is recommended.

A possible approach is illustrated in Figure 11, page 39. 2 rasters (here: 19 × 19(1) raster objects (=grid "points") are to be defined as marking pattern. They should fit into the scan head working field. The raster object is a cross for one raster and a circle for the other(2).

(1) With odd numbers, the central raster object is going to be at (0|0).

These marking patterns are contained in `CHECK_CALIBRATIONS`.

Then, both marking patterns (sequence does not matter) are marked onto one and the same marking substrate (*do not move in the meantime*):

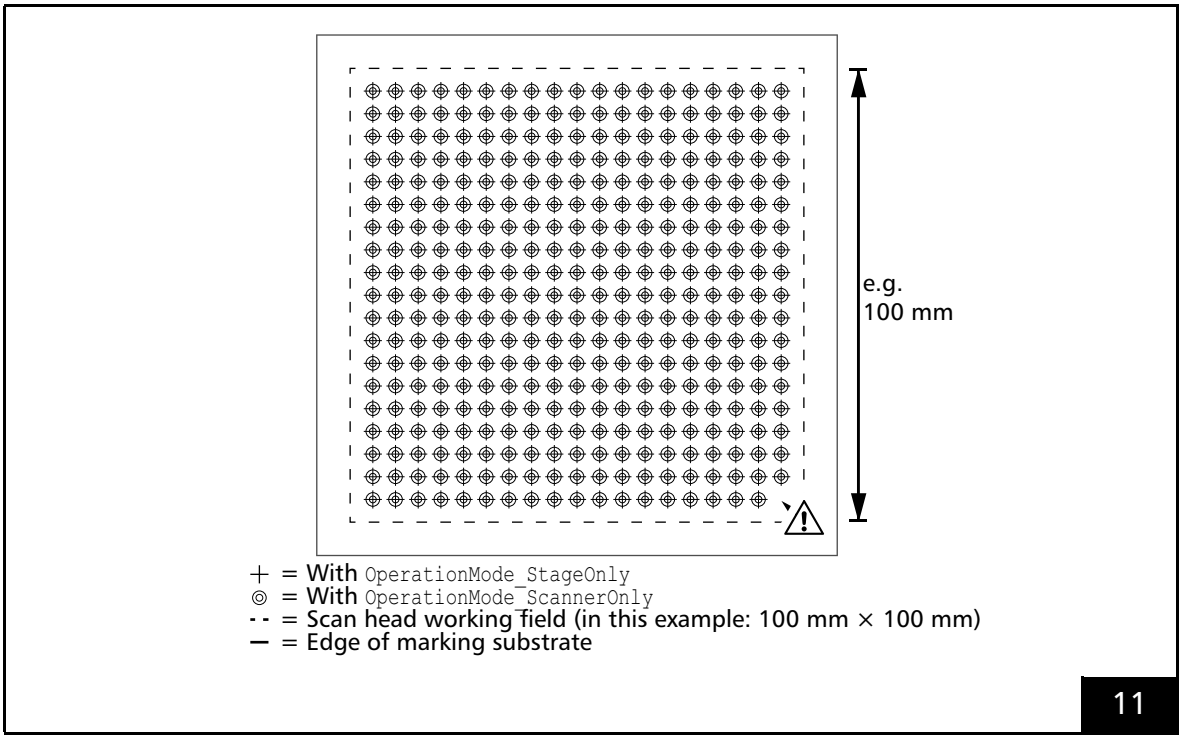
- The raster of crosses in operation mode "`StageOnly`"
- The raster of circles in operation mode "`ScannerOnly`"

The marking result is then visually inspected. Criteria:

- No rotation of the grids to each other
- The corresponding individual raster objects are superimposed (as close as possible)

If needed, also a quantitative evaluation using a coordinate measuring machine can be carried-out.

(2) By omitting one raster object (analogously to QR codes here: downright) you can see immediately the orientation of the grid in the marking result . Additional advantage: see caption of Figure 11, page 39.



Exemplary marking result in Chapter 7.3 "Verification – Checking the Static Calibration Accuracies of Scan Head and ACS Axes", page 39.
Both marking patterns have been defined without a raster object downright (warning triangle). If there is a marking at this position in the marking result nevertheless, then the coordinate systems are not correctly defined.

Objectives

- The quality of the calibrations of the positioning stage and scan head is confirmed.

Materials

- Marking substrate (incl. spares)
- If a quantitative evaluation is needed: customer's coordinate measuring machine (or camera-based solution)

Prerequisites

- The marking patterns must first be simulated (as always before a very first execution) to ensure that no limit violations are contained in the control values.
- All subsystems (for example, the laser) are ready-to-operate.
- A marking substrate has been inserted.

Procedure

- ⚠ **Warning!** Risk of injury due to laser radiation! Comply with laser safety regulations!
- ⚠ **Warning!** Risk of injury due to positioning stage motion! No persons in the danger zone!
- ⚠ **Caution!** Risk of property damage due to positioning stage motion! No foreign objects in the danger zone!

(1) Start `Installation_Project.exe` and initialize a syncAXIS control instance (`INIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(2) Start `CHECK_CALIBRATIONS` as follows:

- '9' – return key – then the raster size (smaller than the scan head working field size) – one of the markings are carried out in operation mode `ScannerOnly`) – return key – then the number of raster objects (recommended: 15 × 15) – return key – then the speed to move the positioning stage – return key – y – return key.
  The Jobs are each calculated for operation mode "`ScannerOnly`" and "`StageOnly`" including Trajectory planning, however, without motion decomposition.
  The successful completion of the processes is indicated by "`Finished with return value 0`".

(3) Exit `Installation_Project.exe` (`DELETE_INSTANCE` > `EXIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(4) Continue with the following procedure "Evaluation".

Evaluation

(1) If the markings have not been completed, check for error messages. For this, you can use the log file (see `<cfg:LogfilePath>`) as well as the ACS SPiiPlus MMI Application Studio. Observe also Chapter 14 "Appendix G: About Error Handling with syncAXIS control", page 59 and "syncAXIS-DLL – Application Programming Interface" Manual, Chapter 2.7.1 "About the Buffers of the syncAXIS control Instances", page 42.

(2) Perform a visual inspection of the marking result under magnification (for example, with reflected-light microscope). The objective is to have all crosses and circles are perfectly superimposed:

- If this is the case visually, a quantitative evaluation by means of a coordinate measuring machine is recommended in addition. Then quantitative statements about the scan head working field calibration can be made. As a rule, the static positioning accuracy of the positioning stage is higher than that of the scan head.
- If this is not the case, at least one of the two calibrations is not ideal. The steps in Chapter 7.4 "Verification – Checking the Synchronization of all System Components", page 41 and Chapter 7.5 "Verification – Determining the Combined Motion Accuracy", page 43 should not be performed in this state. Find the imprecise calibration and carry out the calibration steps again.

7.4 Verification – Checking the Synchronization of all System Components

The following criteria have been processed before:

- The static calibration of the scan head (= scan head working field calibration)
- The static calibration of the positioning stage (= Error Mapping)

In this step 7.4 it should be verified whether the movements of scan head and the positioning stage are synchronous.

SCANLAB recommends that you carry out this verification at least once. This is because deviations due to incorrect synchronization of movements are then excluded.

The scan head and positioning stage can be easily synchronized via a delay of the theoretical profile because:

- There is no tracking error(1)
- The theoretical profile and the actual profile are *constantly* offset in time(1)

The approach is illustrated in Figure 12, page 41: 42 parallel lines are used as marking pattern, of which 2 each lie on a straight line. The lines are orthogonal to the arrangement direction. The first 21 lines are arranged in one direction (here called "positive"), the second 21 lines in the opposite direction (here called "negative"). This marking pattern is contained in `CHECK_SYSTEMDELAYS`.

(1) Applies to each of the devices.

The positioning stage movement is in positive direction for the first row and in negative direction for the second row(2).

In order to achieve a high speed, the positioning stage is to be accelerated to the highest possible speed. For this purpose, syncAXIS control incorporates jumps and waiting points to accelerate and decelerate the positioning stage.

If scan head and positioning stage are not optimally synchronized, the lines in the two rows are not perfectly marked lying on a straight line. Instead, they have a certain offset to each other. This is the case when the delay in the execution of one of the two components is not equal(3) to the planned one. Since the positioning stage moves orthogonally to the marked lines at constant speed, only an offset is to be expected. However, the lines will remain parallel.

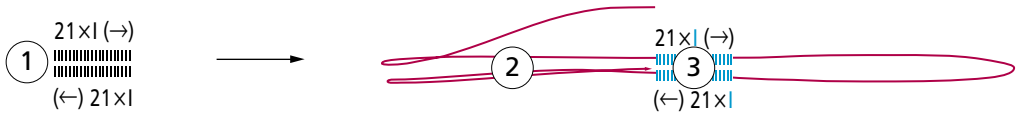
In order to exclude all influences of all axes, the pattern described above is also marked in all 4 spatial directions.

The marking results are then visually checked. Criteria:

- For each pattern, the 21 pairs of lines should lie on a straight line.

(2) According to the low-pass filter, which is responsible for motion decomposition.

(3) In practice, this should not be the case. Nonetheless, a verification of this assumption should be carried out.



Legend
1. Marking pattern as defined.
3. Motion of positioning stage only.
4. Motion combined.

12

Explanations see Chapter 7.4 "Verification – Checking the Synchronization of all System Components", page 41.

Objectives

- The time synchronization of the scan system and the positioning stage is verified.

Materials

- Marking substrates (incl. spares)

Prerequisites

- The marking pattern must first be simulated (as always before a very first execution) to ensure that no limit violations are contained in the control values.
- All subsystems (for example, the laser) are ready-to-operate.
- A marking substrate has been inserted.

Procedure

- ⚠ **Warning!** Risk of injury due to laser radiation! Comply with laser safety regulations!
- ⚠ **Warning!** Risk of injury due to positioning stage motion! No persons in the danger zone!
- ⚠ **Caution!** Risk of property damage due to positioning stage motion! No foreign objects in the danger zone!

(1) Start `Installation_Project.exe` and initialize a syncAXIS control instance (`INIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(2) Start `CHECK_SYSTEMDELAYS` as follows:

- '10' – return key – then the maximum possible positioning stage speed – return key – then the maximum possible positioning stage travel range – return key – y – return key.
  The Job is calculated for operation mode "`ScannerAndStage`" including Trajectory planning and motion decomposition
  (Pay attention to a realistic `FilterBandwidth` value; 1…2 Hz is recommended; see in your initial `syncAXISConfig.xml` from SCANLAB).
  The successful completion of the processes is indicated by "`Finished with return value 0`".

(3) Exit `Installation_Project.exe` (`DELETE_INSTANCE` > `EXIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(4) Continue with the following procedure "Evaluation".

Evaluation

(1) If the marking has not been completed, check for error messages. For this, you can use the log file (see `<cfg:LogfilePath>`) as well as the ACS SPiiPlus MMI Application Studio. Observe also Chapter 14 "Appendix G: About Error Handling with syncAXIS control", page 59 and "syncAXIS-DLL – Application Programming Interface" Manual, Chapter 2.7.1 "About the Buffers of the syncAXIS control Instances", page 42.

(2) Perform a visual inspection of the marking result under magnification (for example, with reflected-light microscope). For each pattern, the 21 pairs of lines should lie on a straight line:

- f this is the case, you can continue with the next step.
- If this is not the case, your machine will behave unexpectedly. Your marking results in combined motion will not be ideal. In this case, consult SCANLAB.

7.5 Verification – Determining the Combined Motion Accuracy

The following criteria have been processed before:

- The scan head working field calibration (= static calibration of the scan head)
- The static calibration of the positioning stage (= Error Mapping)
- Adjustment of the laser delays
- Verification of the time synchronization of scan head and positioning stage

In this step 7.5, the accuracy of the combined motion can now be determined.

For this purpose, for example, a homogenous grid of marking objects (hereinafter referred to as "raster objects") is suitable. To keep the additional measurement errors to a minimum, select the grid in such a way that the highest accuracy is achieved during a measurement.

For this, `COMBINED_ACCURACY` is executed.

`COMBINED_ACCURACY` offers as raster objects: circle, 2 concentric circles, cross, point (spiral from inside to outside, surrounded by a circle).

Optionally, in `COMBINED_ACCURACY` it can be selected that an extended reversal movement for the positioning stage is to be carried out after each grid line (see step 2 and bottom left in Figure 13, page 45). Then a fast movement of the positioning stage during marking is possible and the working field can be used extensively.

Without this extended reversal movement (see step 2 and bottom right of Figure 13, page 45), the positioning stage (due to the low pass filter design of the motion decomposition) slows down before the raster line is complete and the scan head may experience a large deflection (which may exceed its working field).

SCANLAB recommends that you try both alternatives (= with/without extended reversal movement).

Notes

- At this point of installation, your XL *SCAN* setup is basically ready for operation and optimally adjusted. Users should now try out some settings and familiarize themselves with the XL *SCAN* scan system. Carry out a visual inspection and measure the marking results in each case.

Objectives

- The accuracy of the XL *SCAN* scan system in combined motion has been determined.

Materials

- Marking substrates (incl. spares)

Prerequisites

- The marking pattern must first be simulated (as always before a very first execution) to ensure that no limit violations are contained in the control values.
- All subsystems (for example, the laser) are ready-to-operate.
- A marking substrate has been inserted.

Procedure

- ⚠ **Warning!** Risk of injury due to laser radiation! Comply with laser safety regulations!
- ⚠ **Warning!** Risk of injury due to positioning stage motion! No persons in the danger zone!
- ⚠ **Caution!** Risk of property damage due to positioning stage motion! No foreign objects in the danger zone!

(1) Start `Installation_Project.exe` and initialize a syncAXIS control instance (`INIT`) as described in Chapter 6.1 "Starting `Installation_Project.exe` and Initializing syncAXIS control Instance in Hardware Mode", page 26.

(2) Start `COMBINED_ACCURACY` as follows:

- '7' – return key – then the total raster size (smaller than the positioning stage travel range) – return key – then the number of raster objects – return key – then the approximate size of the scan head working field to be used – return key – then the type of raster objects (1…4) – return key – whether an extended reversal movement for the positioning stage movement is to be coded (y or n).
  For "y": The approximate highest positioning stage speed reached along the raster line (value must be found by simulation, which must take place before each marking anyway for safety reasons) – return key – `FilterBandwidth` value from your `syncAXISConfig.xml` – return key – scaling factor of the deceleration/acceleration movement (start value, for example, 0.5) – return key – y – return key.

With “n”: y – return key.
For “y” and “n”, the following applies:
The Job is calculated for operation mode “ScannerAndStage” including Trajectory planning and motion decomposition
(Pay attention to a realistic FilterBandwidth value; 1…2 Hz is recommended; see in your initial syncAXISConfig.xml from SCANLAB).
The successful completion of the processes is indicated by “Finished with return value 0”.

(3) Exit Installation_Project.exe (DELETE_INSTANCE > EXIT) as described in Chapter 6.1 “Starting Installation_Project.exe and Initializing syncAXIS control Instance in Hardware Mode”, page 26.

(4) Continue with the following procedure “Evaluation“.

Evaluation

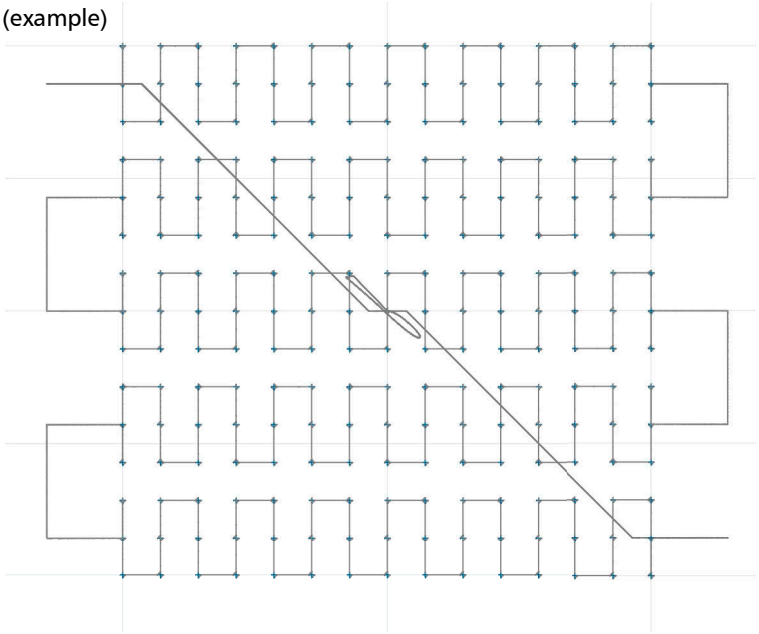
(1) If the marking has not been completed, check for error messages. For this, you can use the log file (see <cfg:LogfilePath>) as well as the ACS SPiiPlus MMI Application Studio. Observe also Chapter 14 “Appendix G: About Error Handling with syncAXIS control”, page 59 and “syncAXIS-DLL – Application Programming Interface” Manual, Chapter 2.7.1 “About the Buffers of the syncAXIS control Instances”, page 42.

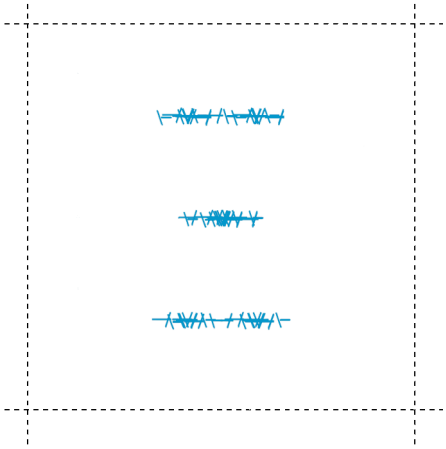
(2) Perform a visual inspection of the marking result under magnification (for example, with reflected-light microscope) as well as a measurement of the raster objects.

Interpretation

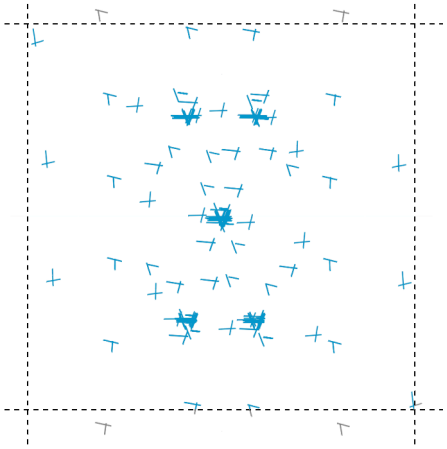
- It is to be expected that the combined total dynamic error is not significantly greater than the sum of the respective static errors provided that:
  - All previous tests and verifications were positive
  - The simulation showed that all control values are within the system limits
  - The position error tolerances are realistically set in the ACS software

Marking in total (example)





ONLY scan head portion WITH prolonged reversal movement of positioning stage (see step 2, page 43)



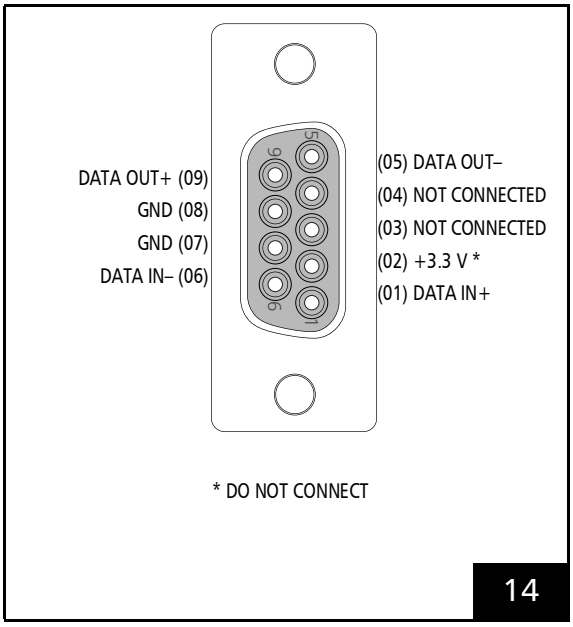
ONLY scan head portion WITHOUT prolonged reversal movement of positioning stage (see step 2, page 43)

13

Explanations see Chapter 7.5 "Verification – Determining the Combined Motion Accuracy", page 43.

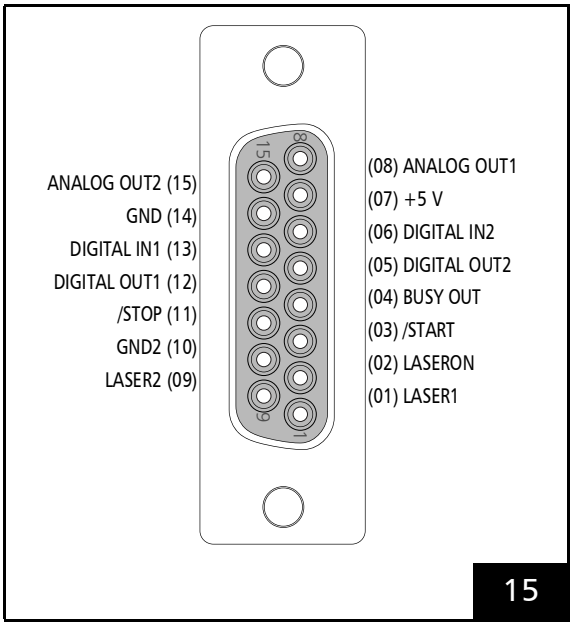
8 Appendix A: RTC6 PCI Express Board – Connectors

8.1 SCANHEAD



RTC6 PCI Express Board: SCANHEAD connector.
D-SUB 09-pin, female. Pin-out.

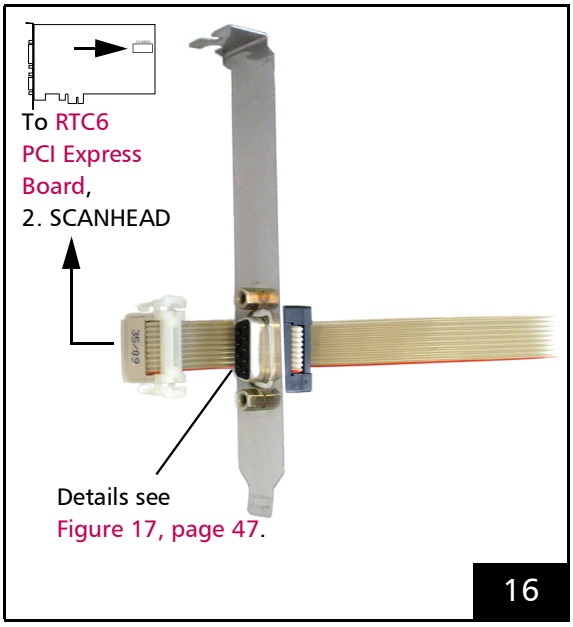
8.2 LASER



RTC6 PCI Express Board: LASER connector.
D-SUB 15-pin, female. Pin-out.

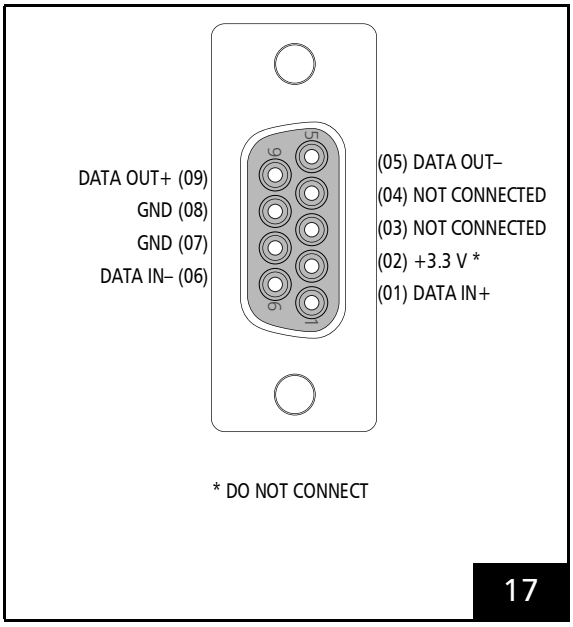
9 Appendix B: SSHC Slot Bracket (#115132)

9.1 View (not Installed)



SSHC slot bracket: view (not installed).

9.2 Connector



SSHC slot bracket: connector (leads the 2. SCANHEAD connector signals of the RTC6 to the exterior.
D-SUB 09-pin, female. Pin-out.

10 Appendix C: syncAXIS control Software Package – Unzipped

Example: `syncAXIS-1.8.zip`.

```
|   Readme.txt                                  Readme file (english only).
|   ReleaseNotes.txt                            Release Notes file (english only).
|   syncAXIS_control_License_Agreement.pdf      License agreement (english).
|   syncAXIS_control_Lizenzvertrag.pdf          License agreement (german).
|   syncAXIS_V1.8.0_API_de-DE.pdf               API and XML reference (german).
|   syncAXIS_V1.8.0_API_en-US.pdf               API and XML reference (english).
|   syncAXIS_V1.8.0_Installation_de-DE.pdf      Installation manual (german).
|   syncAXIS_V1.8.0_Installation_en-US.pdf      Installation manual (english). This document.
|
\---cmake                                       Folder with files for CMake, see page 14.
|   |
|   |   syncAXISConfig.cmake                    syncAXIS cmake target.
|   |   syncAXISConfigVersion.cmake             syncAXIS version for cmake target.
|
\---Demo                                        Folder with "Installation_Project" sources.
|   |                                           Prepared for CMake, see page 14.
|   |   CMakeLists.txt                          Main file for CMake.
|   |   GenerateCMakeTestProject.bat            Batch file. To generate the MS Solution via CMake.
|   |   ResetBuild.bat                          Batch file. Deletes the Build_syncAXIS folder, if it
                                                exists.
|   |   syncAXIS.cmake                          File for CMake.
|   |
|   \---Configuration_Files                     Empty folder. Purpose: see Figure 3, page 21.
|   |
|   \---Installation_Project
|       |   CMakeLists.txt                      File for CMake.
|       |   IncludePath.h.in                    File for CMake.
|       |
|       \---include
|       |       ConfigFileSearch.h              File for CMake.
|       |       DemoFunctions.h                 File for CMake.
|       |       InputTypes.h                    File for CMake.
|       |       Jobs.h                          File for CMake.
|       |       MainLoop.h                      File for CMake.
|       |       MarkingPatterns.h               File for CMake.
|       |       OperationTypesProcedures.h      File for CMake.
|       |       syncAXISControl.h               File for CMake.
|       |
|       \---source
|       |       ConfigFileSearch.cpp            File for CMake.
|       |       DemoFunctions.cpp               File for CMake.
|       |       InputTypes.cpp                  File for CMake.
|       |       Jobs.cpp                        File for CMake.
|       |       Main.cpp                        File for CMake.
```


```
|       |       MainLoop.cpp                          File for CMake.
|       |       MarkingPatterns.cpp                   File for CMake.
|       |       OperationTypesProcedures.cpp          File for CMake.
|       |       syncAXISControl.cpp                   File for CMake.
|
\---Licences                                          Folder with license texts.
|       (several files)
|
\---RTC6                                              Folder with files and tools for the RTC6.
|   |   RTC6_Doc.Rev.1.0.*de-DE.txt                   RTC6 Manual (german).
|   |   RTC6_Doc.Rev.1.0.*en-US.txt                   RTC6 Manual (english).
|   \---Driver                                        Folder with the RTC6 board driver for Windows.
|   |   |   (several files in several folders)
|   |
|   \---ProgramFiles                                  Folder with the RTC6 fles.
|   |       RTC6BIOSETH_35.out                        BIOS for RTC6 Ethernet Board.
|   |       RTC6BIOSOUT_23.out                        BIOS for RTC6 PCI Express Board.
|   |       RTC6Dat.dat                               Binary auxilliary file for the RTC6.
|   |       RTC6Eth.out                               Program file for RTC6 Ethernet Board DSP.
|   |       RTC6Out.out                               Program file for RTC6 PCI Express Board DSP.
|   |       RTC6RBF.rbf                               Firmware file for RTC6 PCI Express Board FPGA.
|   |
|   \---Tools
|       \---iSCANcfg                                  Folder with the tool iSCANcfg.exe. Diagnosis and
|       |       (several files)                       configuration program for iDRIVE-scan systems/
|       |                                             RTC boards incl. manual.
|       |
|       \---RTC6conf                                  Folder with the tool RTC6conf.exe. Diagnosis and
|       |       (several files)                       configuration program for RTC6 boards incl. description.
|       |
|       \---SleepMode                                 Folder with a script to deactivate all
|       |       (several files)                       Windows sleep and hibernate modes incl. description.
|
```


```
\---syncAXIS_control                                Folder with files and tools for syncAXIS control.
|   |
|   \---bin
|   |   \---dll
|   |   |       RTC6DLL.dll                        32-bit RTC6-DLL file. Needed by syncAXIS-DLL.
|   |   |       syncAXIS.dll                       The 32-bit syncAXIS-DLL.
|   |   |       xerces-c_3_2.dll                   DLL file. Needed by syncAXIS-DLL.
|   |   |
|   |   \---lib
|   |   |       syncAXIS.lib                       Visual C++ import library.
|   |   |
|   |   \---Wrapper
|   |       \---C#                                 Folder with auxilliary files for software development
|   |           (several files)                    under C#.
|   |
|   \---bin64
|   |   \---dll
|   |   |       RTC6DLLx64.dll                     64-bit RTC6-DLL file. Needed by syncAXIS-DLL.
|   |   |       syncAXIS.dll                       The 64-bit syncAXIS-DLL. As 64-bit variant.
|   |   |       xerces-c_3_2.dll                   DLL file. Needed by syncAXIS-DLL.
|   |   |
|   |   \---lib
|   |   |       syncAXIS.lib                       Visual C++ import library.
|   |   |
|   |   \---Wrapper
|   |       \---C#                                 Folder with auxilliary files for software development
|   |           (several files)                    under C#.
|   |
|   \---Configuration
|   |       syncAXIS_1_8.xsd                       XML scheme for syncAXISConfig.xml.
|   |       syncAXISConfig.Template.xml            Template for syncAXISConfig.xml files.
|   |                                              Do not use in real operation!
|   |       syncAXISConfig_MultiHead.Template.xml  Template for Multi Head syncAXISConfig.xml files.
|   |                                              Do not use in real operation!
|   \---include
|   |       syncAXIS.h                             Header file (among others, for implicit linking
|   |                                              to syncAXIS.dll).
|   |       syncAXISDefinitions.h                  Header file.
|
```

```
\---Tools
|   \---syncAXIS_Configurator                    Folder with the tool syncAXIS Configurator
|   |                                            incl. manual.
|   |       (several files)
|   |
|   \---syncAXIS_Installation                    Folder with the tool Installation_Project.exe
|   |                                            The manual is located in the root directory.
|   |       (several files)
|   |
|   \---syncAXIS_MasterSlaveSynchronizer         Folder with the tool only for special systems
|   |                                            incl. manual.
|   |       (several files)
|   |
|   \---syncAXIS_Viewer                          Folder with the tool syncAXIS Viewer
|   |                                            incl. manual.
|   |       (several files)
```

11 Appendix D: About the “Installation_Project”

In syncAXIS control-software package an example code for C++ is provided.

It is located in the folder:

```
\---Demo
|   \---Installation_Project
```

The sample project “Installation_Project” is intended to demonstrate the easy handling of the syncAXIS-DLL and to illustrate simple programming concepts, for example, like loading asynchronous lists and executing lists, see Chapter List Handling in “syncAXIS-DLL – Application Programming Interface” Manual.

In the following, the different actions 0…13 of the “Installation_Project” are listed and described.

The Jobs are also used in several chapters this manual to allow easy installation and configuration as well as to verify the correctness of all settings.

Input	Short label	Description	Demonstration purpose
0	EXIT	To exit this user program. This calls **slsc_cfg_delete**. Used in Chapter 5.2 “Starting Installation_Project.exe and Initializing syncAXIS control instance in Simulation Mode”, page 22 and others.	**slsc_cfg_delete** must always be called when terminating the user program (whether intentionally or unintentionally) to ensure that the program is terminated in the correct order.
1	INIT	To initialize the syncAXIS control instance based on the settings in the syncAXISConfig.xml specified(*). (*) INIT offers paths to choose from: the working directory in which the Installation_Project.exe has been started and \---Demo | \---Configuration_Files. Used in Chapter 5.2 “Starting Installation_Project.exe and Initializing syncAXIS control instance in Simulation Mode”, page 22 and others.	To initialize a syncAXIS control instance only one syncAXIS control function is required. All settings are “safely” stored in syncAXISConfig.xml and are read in by this function. In hardware mode this means: creating the syncAXIS control instance, initialization of the RTC6 board, synchronization with the ACS components, setting of all defined parameters on the RTC6 board and in the user program, execution of the laser initialization routine (if defined), etc.
2	CHECK_LASER_FIRING	To configure the laser. To check the correct laser communication. Useful tool to optimize focus, laser parameters, etc. Used in Chapter 6.2 “Verifying Whether Laser Outputs Radiation”, page 28.	To demonstrate the Mode “Manual Positioning”. See Chapter 2.12 in the “syncAXIS-DLL – Application Programming Interface” Manual.

Input	Short label (cont'd.)	Description (cont'd.)	Demonstration purpose (cont'd.)
3	REFERENCE_RUN	To check the communication of the system components, that is, positioning stage, scan device and laser. Used in Chapter 6.3 "Verifying the Correct Communication with Scan Head and Positioning Stage", page 29.	To demonstrate the Mode "Manual Positioning". See Chapter 2.12 in the "syncAXIS-DLL – Application Programming Interface" Manual.
4	TEST_MARKING	First test marking to check the functionality of all components including syncAXIS control Trajectory planning and ACS communication via the ACS SL2-100 to EtherCAT converter. First impression of corner rounding thanks to blending or laser switching points in time in the corners of the marking result. Used in Chapter 5.3 "Simulating "TEST_MARKING" in Operation Mode "ScannerAndStage"", page 24 and Chapter 6.4 "Marking the (Previously in 5.3) Simulated "TEST_MARKING"", page 30.	Simplest test program in the entire "Installation_Project". Shows the Job structure in the function `writeJob()`, as well as to parallel list loading in an asynchronous thread. Furthermore, function `startJob()` shows the concept of querying the execution status.
5	SCANNER_CALI	To mark a grid in operation mode `ScannerOnly` to calibrate the scan device. The points can be measured and passed to correXion pro in order to create an optimized correction file. Ideally, the mechanical x axis is used as reference for the measurement. This Job moves the x axis a few millimeters and then marks a reference point in the working field center. Used in Chapter 7.2 "Optimization – Calibrating the Scan Head Working Field", page 36.	Use of different operation modes (`ScannerOnly`). Simple test program with simple structure consisting of several syncAXIS control Jobs.
6	STAGE_CALI	Not used for installation in this manual. To mark a large grid in operation mode `StageOnly`. This marking pattern can be matched with a calibration plate (glass master) to create the Error Mapping of the mechanical axes.	Use of different operation modes (`StageOnly`). Simple test program with simple structure consisting of several Jobs.

Input	Short label (cont'd.)	Description (cont'd.)	Demonstration purpose (cont'd.)
7	COMBINED_ACCURACY	To mark a grid of selectable grid elements in operation mode ScannerAndStage (combined motion). Can be used for measuring and evaluating. To reach higher positioning stage velocities, a positioning stage path optimization motion segment can be included. Used in Chapter 7.4 "Verification – Checking the Synchronization of all System Components", page 41 and Chapter 7.5 "Verification – Determining the Combined Motion Accuracy", page 43.	Most complex Job. Many grid elements available. Usage of the function **slsc_list_wait_with_laser_off** for optimization of the positioning stage motion. If you are interested in more details, ask SCANLAB for support on "path optimization".
8	CHECK_LASERDELAYS	To mark a grid of objects (similar to "H") in operation mode ScannerOnly at maximum speed to optimize the laser switching points in time. Along the grid, LaserSwitchOffsetTime and LaserPreTriggerTime are adjusted so that the object with the optimal laser switching points in time can be selected from the array of objects. These values can then be used to carry out this test again with finer iteration steps or to directly use the newly determined laser delay values. Used in Chapter 7.1 "Optimization – Finding Laser Delays that Suit Your Application", page 33.	Functional (low demonstration effect).
9	CHECK_CALIBRATIONS	To mark a grid in operation mode ScannerOnly (circles) and another one in operation mode StageOnly (crosses). The grids can be compared to check the calibration quality. Used in Chapter 7.3 "Verification – Checking the Static Calibration Accuracies of Scan Head and ACS Axes", page 39.	Functional (low demonstration effect).

Input	Short label (cont'd.)	Description (cont'd.)	Demonstration purpose (cont'd.)
10	CHECK_SYSTEMDELAYS	To mark rows of lines orthogonal to mechanical motion. The lines are executed in positive and negative directions, and then repeated for all 4 spatial directions. The objective is to check whether the lines of both mechanical motion directions are collinear or whether an offset (in the direction of the mechanical motion) can be seen. In case the lines are not collinear (offset in the direction of the mechanical motion), the positioning stage motion is not perfectly synchronized with the scan device motion. If this is the case, contact SCANLAB. An arrow indicates the mechanical direction of motion. Used in Chapter 7.4 "Verification – Checking the Synchronization of all System Components", page 41.	Functional (low demonstration effect).
11	CHANGE_SETTINGS	Not used for installation in this manual. To adapt a few settings like speeds and operation modes. Note that most Jobs use own settings anyhow. Useful for the Job TEST_MARKING and to apply an offset, if the to-be-marked area is not in the positioning stage center.	Use of Configuration functions (**slsc_cfg_***). Reaction to user input or peripheral input.
12	DELETE_INSTANCE	To destroy the syncAXIS control instance. Subsequently INIT must be carried out to reload the syncAXISConfig.xml and to create a new syncAXIS control instance. Used, for example, in Chapter 5.2 "Starting Installation_Project.exe and Initializing syncAXIS control instance in Simulation Mode", page 22.	Use of the function **slsc_cfg_delete** for syncAXIS control instance destruction, for example, in order to release all components or to reload syncAXISConfig.xml.
13	CHECK_FOR_ERRORS	To query all system errors and output to the console. Used, for example, in Chapter 5.2 "Starting Installation_Project.exe and Initializing syncAXIS control instance in Simulation Mode", page 22.	Demonstrates querying all error codes. Unlike the RTC6 DLL, syncAXIS control does not accumulate error codes!

12 Appendix E: Cable for Data Transmission According to the SL2-100 Protocol – Requirements

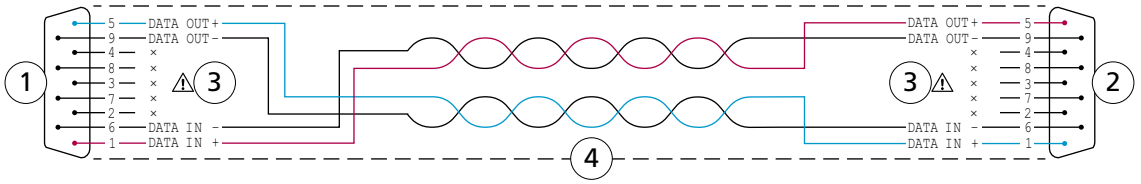
In the typical scope of delivery of SCANLAB components for XL *SCAN* there are (at least two) cables for data transmission according to the SL2-100 protocol(1):

- 1× SL2-100 data cable for the connection RTC6 / SL2-100 to EtherCAT converter
- 1× SL2-100 data cable for the connection RTC6 / scan head

Caution!

In case of repair, replacement and also in case of self- constructions the internal wiring of these cables must be observed, see Figure 18, page 56. Otherwise, the electronics of the connected devices is damaged.

(1) Available at SCANLAB are (not for drag chains): #115428 (length 5 m), #115430 (length 10 m), #115431 (length 15 m).



Legend

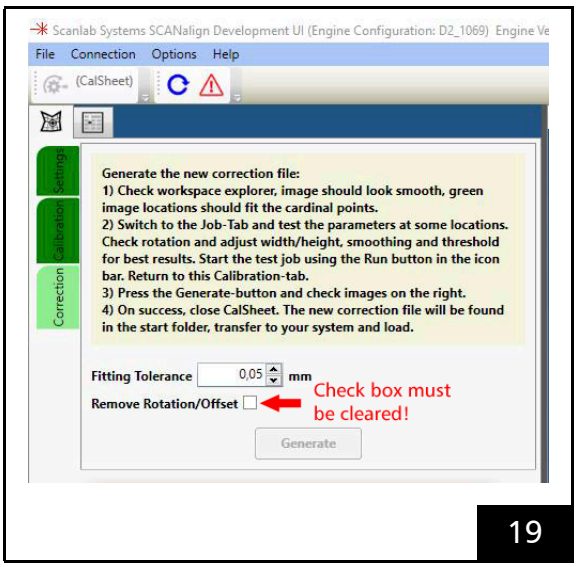
1. To RTC6 PCI Express Board. Connector: D-SUB 09-pole, male.
2. To excelli*SCAN* scan head (or, SL2-100 to EtherCAT converter). Connector: D-SUB 09-pole, male.
3. ⚠ **Caution!** no wires! The cable must have 4 wires only as shown in this figure. Otherwise, the electronics of the connected devices may gets damaged.
4. Shielding.

18

Cable for data transmission according to the SL2-100 protocol: requirements and pin assignments.

13 Appendix F: Calibrating the Scan Head Working Field – Alternative Using CalSheet Software

- This chapter describes an alternative to procedure described in Chapter 7.2 "Optimization – Calibrating the Scan Head Working Field", page 36 (described there with correXion Pro). The same Prerequisites, see page 37, apply.
- For this field calibration procedure, CalSheet software V2.2.5 is a prerequisite(1). Only from this version on, on tab 'Correction', there is the check box **Remove Rotation/Offset**, see Figure 19, page 57.
  Before you generate the new (=optimized) correction file clear this check box.



CalSheet software V2.2.5: make sure that the indicated check box is cleared.

- In this procedure, the calibration plate from the CalSheet package(2) is *not used*.
  Functionally, the calibration plate is replaced by a marked grid of solid lines (see below).

13.1 Defining Two Different Calibration Patterns

This field calibration procedure with the CalSheet software V2.2.5 bases on two different calibration patterns. They must cover the entire scan system working field and are marked on one and the same marking substrate, see Figure 20, page 58:

- 1 raster of crosses (regular array of "×"es; always odd numbers, usually 11 × 11)
  This calibration pattern is going to be marked only by scan head motions (that is, no positioning stage motions).
- 1 grid of solid lines
  This calibration pattern is going to be marked only by positioning stage motions (that is, no scan head motions).
  This marking functionally replaces the calibration plate.

(1) In order to use the CalSheet software here, an image scanner is also required.

(2) The typical CalSheet package scope of delivery includes a calibration plate.

13.2 Marking the Two Calibration Patterns

Notes

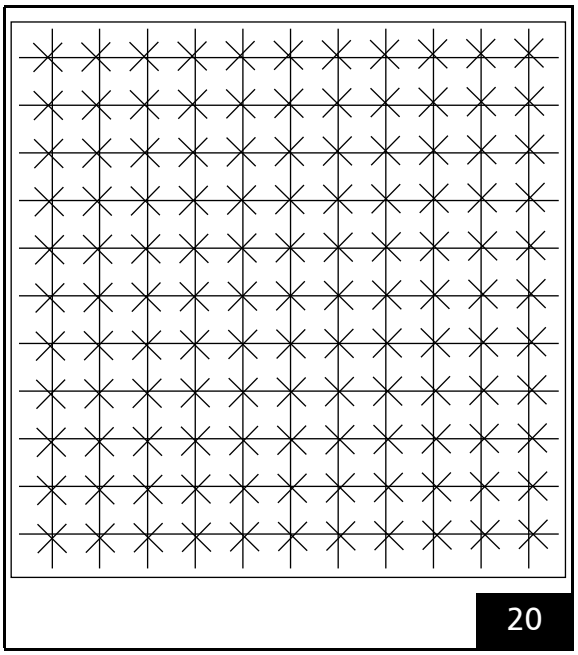
- For later image evaluation, it is essential that the marking result exhibits sharp edges. With an aligned and adjusted beam, see Prerequisites, page 37, this already should normally occur automatically. Furthermore, you can adjust the laser delay or speed settings for better results.
- Use relatively low speeds for marking and jumping (for example 0.25 m/s) in order to avoid dynamic following errors as much as possible. The main goal here is the static field calibration, therefore, a high accuracy is necessary (not a high speed).

To Mark the Both Calibration Patterns – Exemplary Procedure

(1) Mark the cross pattern (regular array of "×"es). Only use the scan head for marking, the positioning stage stays on (0,0) position.

(2) Do not move the marking substrate.

(3) On the same marking substrate, mark the grid of solid lines. Only move the positioning stage for marking. The galvanometer scanners of the scan head stay fixed at (0,0), see Figure 20, page 58.



Marking substrate with marking result after step 3 (with both calibration patterns) for evaluation by the CalSheet software. Typically, some × centers and the grid line intersections do not coincide.

13.3 Generating the Optimized ct5 File with CalSheet-Software

(1) Scan the marking substrate from step 3 (see Figure 20, page 58) with an office scanner.

(2) Load the scanned image to CalSheet software V2.2.5 and generate the new (= optimized) `ct5` file as described in the CalSheet manual. In the process, observe Figure 19, page 57!

14 Appendix G: About Error Handling with syncAXIS control

Warning!

Code sections in this document must never be executed on actual XL *SCAN* systems without prior adaptation and simulation. Otherwise there is a risk of personal injury and damage to property. Disclaimer: SCANLAB accepts no liability for damages or consequential losses resulting from non-observance of this warning. SCANLAB does not take any responsibility on the correctness or functionality of these code sections.

Using a complex system with a large variety of functionalities, many components have to be configured correctly and some pitfalls that have to be avoided when programming the system.

To make the error handling as easy as possible for integrators and users, syncAXIS control offers many ways to identify problems starting from return values as the “first level” all the way down to a logging mechanism. Compared to RTC6 programming, the scope of error handling is significantly greater with syncAXIS control.

While using the RTC6 API only few commands provide a status return value like **init_rtc6_dll** or **load_program_file**, every syncAXIS control command provides a status return value where 0 always indicates successful execution, see Chapter 14.1 “Return Values of the syncAXIS control Functions”, page 60.

Similar to the RTC6 API, syncAXIS control API contains a function **slsc_ctrl_get_error**, see Chapter 14.2 “slsc_ctrl_get_error”, page 61. In contrast to the RTC6 command **get_error**, the syncAXIS control function does not return an accumulated error code, but an error specific error value paired with an error message for each specified error (“second level”). The number of errors occurred can be queried by **slsc_ctrl_get_error_count**.

Completely new compared to the RTC6 API is syncAXIS control logging mechanism (“third level”), see Chapter 14.3 “Logging in syncAXIS control”, page 62. Configurable in the `syncAXISConfig.xml`, users have the option to enable a logging to the application console and/or to an text file. There are 3 different log levels.

Typically, most issues arise when initializing the XL *SCAN* system. The function **slsc_cfg_initialize_from_file** reads the `syncAXISConfig.xml` and starts the communication with all hardware components like the ACS Motion Controller, ACS SL2-100 to EtherCAT converter and RTC6 board(s) as well as 3rd party software included in the syncAXIS control-software package.

14.1 Return Values of the syncAXIS control Functions

Other than RTC6 commands, each syncAXIS control function provides a return value, see also Figure 21, page 60.

The return value 0 always indicates that the called syncAXIS control-function has been executed successfully.

If the return value is … 0 , the value indicates

- either an unintended use of a certain function that does not necessarily cause further problems in the future (for example, calling a syncAXIS control function in a mode it is not allowed to be used)
- or a serious problem (for example, automatic stop of the Master-RTC6 board)

In the latter case, the syncAXIS control instance is put to an error state.

Most subsequent functions return the value 1 until the syncAXIS control instance is reinitialized (that is, re-establish communication to the hardware and rebuild the planning objects).

To check for more specific information on the occurred problem, the **slsc_ctrl_get_error** function can be used, see Chapter 14.2 "slsc_ctrl_get_error", page 61.

```
OK =                                0x0000,
InErrorState =                      0x0001,
ErrorOccured =                      0x0002,
NotAllowedWithoutInitialization =   0x0004,
NotAllowedInExecuting =             0x0008,
BufferFull =                        0x0010,
NotReadyForExecution =              0x0020,
UnplausibleOrUnknownParameter =     0x0040,
JobStructureNotValid =              0x0080,
Undefined0x0100 =                   0x0100,
NotAllowedInCurrentConfiguration =  0x0200,
Undefined0x0400 =                   0x0400,
NotAllowedInCurrentMode =           0x0800,
InvalidPosition =                   0x1000,
Timeout =                           0x2000,
XmlLoadError =                      0x4000,
Initialization failed =             0x8000,
Undefined0x10000 =                  0x10000,
Undefined0x20000 =                  0x20000,
HandshakeFailed =                   0x40000,
Undefined0x80000 =                  0x80000,
Undefined0x100000 =                 0x100000,
Undefined0x200000 =                 0x200000,
Undefined0x400000 =                 0x400000,
Undefined0x800000 =                 0x800000,
Undefined0x1000000 =                0x1000000,
Undefined0x2000000 =                0x2000000,
Undefined0x4000000 =                0x4000000,
Undefined0x8000000 =                0x8000000,
UnknownDevice =                     0x10000000,
Undefined0x20000000 =               0x20000000,
MaxInstancesReached =               0x40000000,
InvalidOrMissingDongle =            0x80000000
```

21

Example: return values in syncAXIS control. See "syncAXIS-DLL – Application Programming Interface" Manual, Chapter 4 "Standard Return Values of the syncAXIS-DLL Functions", page 279.

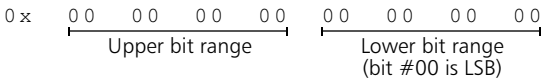
14.2 slsc_ctrl_get_error

Other than with RTC6 commands, syncAXIS control function **slsc_ctrl_get_error** does not return an accumulated error code, but an error specific value including an error message (“second level”).

To iterate through all the errors that have occurred, **slsc_ctrl_get_error_count** can be used to query the number of errors.

The two functions **slsc_ctrl_get_error** and **slsc_ctrl_get_error_count** can be used, for example, to easily write a function that outputs the error messages of all errors that have been occurred, see Figure 22, page 61.

The returned error code is a 64-bit value with separate meanings for the upper and lower 32 bits:



The upper 32 bits define the group of error codes for the lower 32 bits.

For example,
`0x00 00 00 00 nn nn nn nn` is an RTC6 error code.

The meaning of all error codes is described in the “syncAXIS-DLL – Application Programming Interface” Manual. In particular, you find further information there in Chapter 5 ”Error Codes with slsc_ctrl_get_error, Log File and Console“, page 282 on error codes whose lower bit range is not entirely 0.

```
// C++ code section for educational purposes only.
// Do not execute this code on actual XL SCAN systems without prior modification and simulation!
// Observe the safety notices and disclaimer on page 59.

void checkForErrors(size_t SLHandle, uint32_t& ReturnValue)
{
    size_t ErrorCount = 0;
    ReturnValue |= slsc_ctrl_get_error_count(SLHandle, &ErrorCount);
    std::cout << "Error count: " << ErrorCount << std::endl;
    for (size_t Counter = 0; Counter < ErrorCount; ++Counter)
    {
        uint64_t ErrorCode = 0;
        const static size_t ErrorTextSize = 1000;
        char ErrorText[ErrorTextSize];
        ReturnValue |= slsc_ctrl_get_error(SLHandle, Counter, &ErrorCode, ErrorText, ErrorTextSize);
        std::cout << "Error with error code: " << std::hex << ErrorCode  << std::dec << std::endl;
        std::cout << "Error text: " << ErrorText << std::endl;
    }
    return;
}
```

22

Simple function that queries all errors present in a syncAXIS control instance and outputs them to the console.

14.3 Logging in syncAXIS control

As a “third level”, the syncAXIS control logging mechanism can be used for a more advanced problem analysis, See “syncAXIS-DLL – Application Programming Interface” Manual, Chapter 2.8 “About the Logging in syncAXIS control”, page 47.

14.4 Tips for Error Handling with syncAXIS control

- Check the return values of all syncAXIS control functions in your user program. If these are ≠ 0, cancel the Job execution.
  In case of an error:
  – Query the errors that have occurred by **slsc_ctrl_get_error_count** and **slsc_ctrl_get_error**, see also Figure 22, page 61.
  – If necessary, consult the “syncAXIS-DLL – Application Programming Interface” Manual for error codes that occurred, if their lower bit range is not 0.
  – Read the logged messages in the log file carefully. Sometimes these contain additional information about the cause of the error.
- In case of problems with the positioning stage:
  – Check if errors are shown in ACS SPiiPlus MMI Application Studio.
- Usually the error messages that occur provide good information about the cause of the error, so you can fix your problem yourself.
  – If not: Contact SCANLAB support. In this case, it is helpful if you describe your procedure in as much detail as possible and include the generated log file.

15 Change Index

The following are changes in this manual due to the technical evolution of the product as well as significant editorial changes.

Changes to document revision 2.0.4 en-US from document revision 2.0.3 en-US

Name of chapter / function table	Notes / Changes
Global	Document Revision • 2.0.4 en-US applies to syncAXIS control-software package • V1.8.0
RTC6 Ethernet Board, page 10	Software change. Deleted: “To date, only for 1-head setup“.
Chapter 10 “Appendix C: syncAXIS control Software Package – Unzipped”, page 48	Software change. File names.

Notes