excelli*SCAN* Scan Heads – Functional Principle of *SCAN*ahead Servo Control and Operation by RTC6 Boards

RTC6 Software Package V1.6.1

RTC6 boards and excelli*SCAN*
Doc. Rev. 1.0.8 en-US

February 6, 2020

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. 1.0.8 en-US - February 6, 2020)

Contents

1 Overview

1.1 Introduction

SCANLAB excelli*SCAN* scan heads are equipped with *SCAN*ahead servo control. Thus they exhibit different dynamics than scan heads with conventional control.

In particular, you need to consider timing differences when synchronously operating your scan head, laser and peripheral equipment.

Here, the RTC6 command set provides special *SCAN*ahead commands that considerably simplify adapting of existing RTC6 applications (for conventional scan heads) into programs for excelli*SCAN* scan heads with *SCAN*ahead control. Basically, just add two lines of code to your existing RTC6 program:

```
set_scanahead_params( 1, 1, 1, 0, 0, 0 );
```

```
activate_scanahead_autodelays( 1 );
```

However, the RTC6 board must be equipped with the “SCANa” option to execute this program code.

All details about these and other commands are described in this manual. RTC6 boards

1.2 About this Manual and Other Documents

This manual describes user-relevant differences between excelli*SCAN* scan heads with *SCAN*ahead servo control and scan heads with conventional control. And it describes all special RTC6 commands reserved exclusively for operating *SCAN*ahead-based excelli*SCAN* scan heads.

In this way, this manual complements the RTC6-Manual. To program complete RTC6 applications for *SCAN*ahead-based excelli*SCAN* scan heads, you have to observe both manuals. Also observe the notes in your scan system manual.

Notes

- This manual refers to the RTC6 Software Package V1.6.1.
- All references to the RTC6-Manual refer to Doc. Rev. 1.0.3.

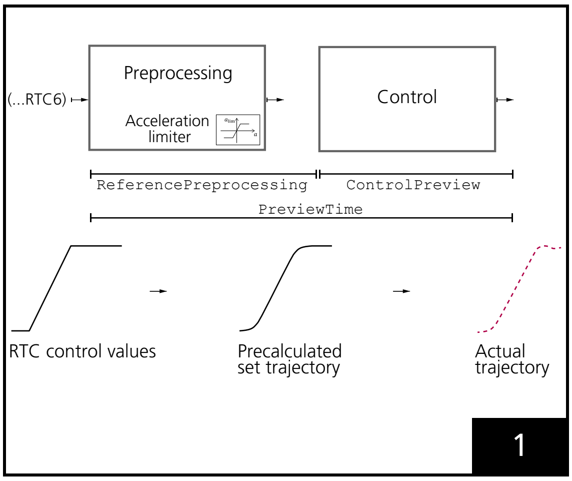
2 excelli*SCAN* Scan Heads with *SCAN*ahead Servo Control

2.1 Introduction

All excelli*SCAN* scan heads are equipped with *SCAN*ahead servo control. This innovative control concept[1]:

- Produces no tracking error and
- Exploits the maximum acceleration capability of the galvanometer scanners.

Figure 1 shows how *SCAN*ahead servo control is structured. It consists of 2 parts[2], “Preprocessing”[3] and “Control”[4].



1

Structure of *SCAN*ahead servo control (timing depictions are not true-to-scale).

The RTC6 board puts out RTC control values. The preprocessing unit analyzes the occurring accelerations therein. Accelerations which the scan system cannot execute are replaced by accelerations, which can actually be executed (“the accelerations are limited”).

The result is a precalculated set trajectory of acceleration-limited control values. The required time for this is depicted as `ReferencePreprocessing` in figure 1 and figure 4.

The galvanometer scanner control successfully traverses the precalculated set trajectory (actual trajectory in figure 1) – but only with a temporal offset (the control preview time `ControlPreview`, see figure 1 and figure 4). This enables full usage of scan system dynamics.

The entire temporal offset between RTC control values and galvanometer scanner motions (that is, `ReferencePreprocessing` plus control precalculation time `ControlPreview`) is called `PreviewTime`.

The term `PreviewTime` highlights the excelli*SCAN* scan head’s need to know the RTC control values in advance by this amount of time to ensure punctual execution.

For synchronous laser control, the laser control signals likewise need to take `PreviewTime` into account. This also applies for any other control signals intended to be transmitted synchronously with scanner motion.

If the RTC6 board is used, synchronous execution of galvanometer scanner movement, laser control and other peripherals is ensured.

The precalculated set trajectories with limited, constant set accelerations cause acceleration time to vary in accordance with speed changes. This means scanner delays and laser delays needed for taking acceleration time into account are also dependent on speed.

The RTC6 board can automatically calculate the required delays. Therefore, SCANLAB recommends the RTC6 board for operating excelli*SCAN* scan heads with *SCAN*ahead servo control, see Chapter 3.1 ”Introduction”, page 21.

[1] Thus, excelli*SCAN* scan heads cannot be operated like conventional scan heads.

[2] Each is a special firmware which runs on the servo board for the galvanometer scanners.

[3] A preprocessing for the ”Control” occurs, that is, the RTC control values are postprocessed.

[4] Controls the positions for the both galvanometer scanners.

2.2 Comparing excelli*SCAN* Scan Heads to Scan Heads with Conventional Servo Control (for example, intelli*SCAN*)

2.2.1 What Remains Unchanged with excelli*SCAN* Scan Heads?

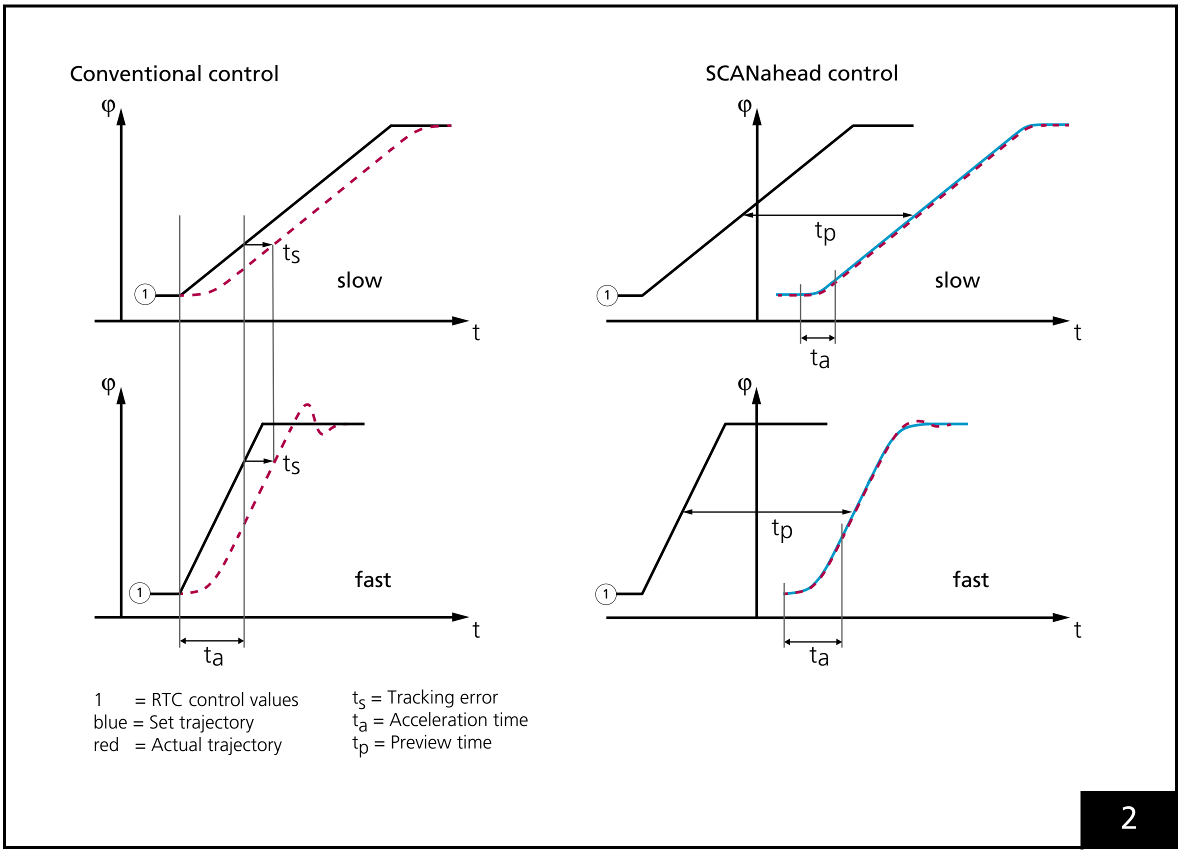
What is described in the RTC6-Manual, Chapter 7.2 "Delay Settings – Coordinating Scan Head Control and Laser Control", page 134 also applies when controlling an excelli*SCAN* scan head by an RTC6 board:

- The galvanometer scanner motions are temporally offset from the RTC control values by the `PreviewTime` (not: tracking error!). For details, see the table in Chapter 2.2.2 "Comparison of Scan Heads with Conventional Control and SCANahead Control", page 8.
- The temporal stream of actual values (=actual trajectory) is smoothed (due to the limited acceleration potential of galvanometer scanners) in comparison to the stream of RTC control values(1).
- To compensate the temporal offset between RTC control values and actual trajectories when synchronizing galvanometer scanner motions and laser signals, the RTC6 allows setting of scanner and laser delays.
  The meaning of delays for the excelli*SCAN* is the same as for scan systems without *SCAN*ahead control. The same applies to the commands **set_scanner_delays** and **set_laser_delays** to set the delays.
- The preferable alternative is to let the RTC6 board automatically calculate and set scanner delays and laser delays, see Chapter 3.1 "Introduction", page 21.

(1) That is, any curves within the control value trajectory get reduced.

2.2.2 Comparison of Scan Heads with Conventional Control and *SCANahead* Control

SCANahead control makes excelli*SCAN* scan heads behave differently in some respects than scan heads with conventional control. These differences are contrasted in figure 2 and the following table.



Comparison: Conventional control and *SCANahead* control.

Scan Head with conventional control	**Scan Head with *SCAN*ahead control, for example,** excelli*SCAN*
• Scan heads with conventional control exhibit a characteristic temporal offset called tracking error t_s, see figure 2. The tracking error t_s results from the implemented servo control structure. The tracking error duration t_s is a metric for the dynamics of a scan head. The tracking error t_s is (at ”normal” speeds) practically independent of scan speed. It is also affected by the selected tuning.	• For scan heads with *SCAN*ahead control, the temporal offset is not tracking error t_s. Instead it is the fixed, pre-defined precalculation time `PreviewTime` t_p, see figure 2. The precalculation time `PreviewTime` t_p is qualitatively different from tracking error t_s. And it is not a metric for scan head dynamics. Instead, a physically traversable trajectory gets precalculated within the `PreviewTime` t_p.
• Tracking error t_s can introduce undesirable artifacts during marking, for example, “necking” in circles and arcs. You can partially compensate such artifacts by specifying over-dimensioned set radii.	• Vectors can be traversed even at very high speeds. Necking artifacts during circle/arc marking only start to occur at accelerations exceeding `Amax`[a]. Therefore, these necking artifacts can *not* be compensated by specifying larger set radii!
• For tracking error t_s, a tolerance (maximum value) is specified (tested by SCANLAB). But the exact value can be serial-number-dependent (minor variation within the tolerances are possible). The exact value cannot be queried, and must instead be experimentally determined.	• The `PreviewTime` t_p is a serial-number-independent value. It is permanently stored in the firmware and can be queried from the scan head.
• The trajectory of RTC control values gets smoothed by servo control. The result of smoothing is not precisely known in advance. And it is dependent on the selected tuning. In negative acceleration phases, significant undesirable overshoot may occur.	• From the trajectory of RTC control values, a traversable acceleration-limited set value trajectory gets calculated and transmitted to the servo control, see figure 1. Deviation of the final actual value trajectory from the RTC control value trajectory is therefore known in advance (because it was precalculated).

Scan Head with conventional control (Continued)	**Scan Head with *SCAN*ahead control, for example,** excelli*SCAN* **(Continued)**
- *The duration of acceleration phases is practically speed-independent*. Thus the same amount of time is always required to reach the desired target speed. This means that, particularly at low speeds, the acceleration potential is not fully exploited and the process times of applications are therefore not optimal.	- *SCAN*ahead control ensures constant acceleration (at the maximum capabilities of the scan head) in acceleration phases. *The duration of acceleration phases is thus minimized, although speed-dependent*. Resultingly, scanner delays and laser delays need adjusting in accordance with marking speed.
- You can set scanner delays and laser delays: These do *not* need adjusting for changes in speed. There are rules of thumb for sizing the delays, but truly optimal delays always require empirical determination for each case of application. Furthermore, delay values need determining for each tuning.	- You can set scanner delays and laser delays: They must be set *differently in accordance with speed*. Thanks to constant acceleration and precalculated set positions, the relationship between speed and optimum delays is likewise mathematically known in advance. The RTC6 board can automatically determine the optimal speed-dependent scanner delays and laser delays. It then also sets them dynamically in real time during list execution. For this, the user activates RTC6 autodelay functionality by **set_scanahead_params** and **activate_scanahead_autodelays**`( 1 )` as described in Chapter 3.1 ”Introduction”, page 21 with figure 6 and figure 7. If the user does *not* want to use RTC6 autodelay functionality (**set_scanahead_params** and **activate_scanahead_autodelays**`( 0 )`), then: – Speed-dependent delays “must” be determined and – appropriately set before each change of speed

(a) Therefore, at very high speeds or for very small radii. See also empirical formula in ”Notes on Arc Commands”, page 29.

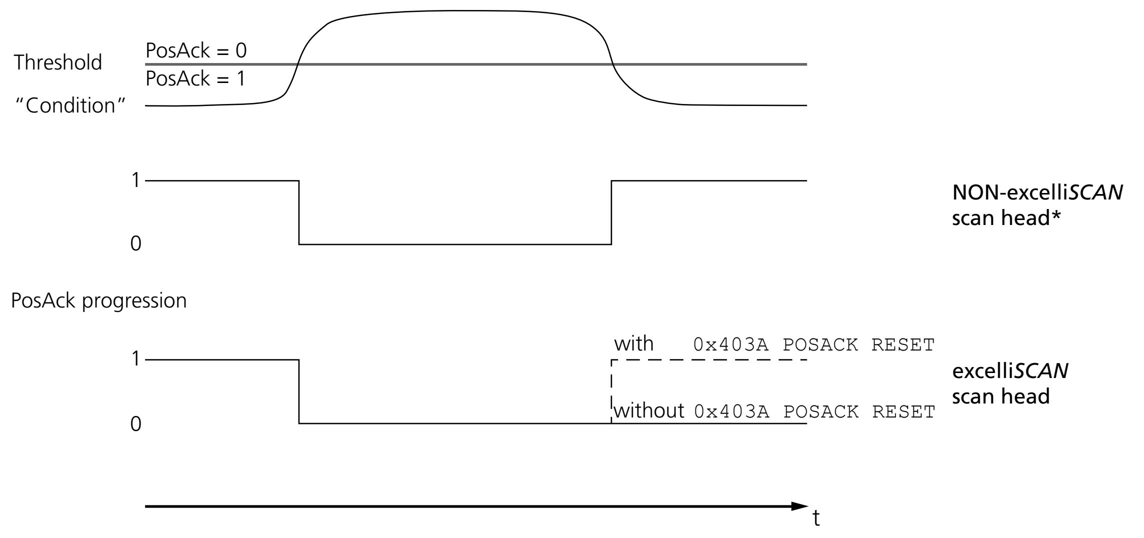
2.2.3 Returned Data Signal Differences: PosAck Status Signal Behavior after Limit Exceedance

Both scan heads with i*DRIVE* technology and excelli*SCAN* scan heads return the PosAck status signal at 10 μs intervals. However, the following differences apply after exceedance of the threshold value, see figure 3:

- For non-excelli*SCAN* scan heads, the PosAck bit gets automatically reset to 1 as soon as the position error is smaller than the limit value.
- For excelli*SCAN* scan heads, the PosAck bit is not automatically reset to 1 as soon as the position error is smaller than the limit value. That is, the PosAck bit remains at 0.
  This lets you determine after the event whether a limit exceedance has been occurred.
  To reset the PosAck bit to 1, you must execute `control_command( Data = 0x403A )`.

Notes

- The PosAck threshold value itself is set similarly, both on scan heads with i*DRIVE* technology and excelli*SCAN* scan heads. To do so, you must execute `control_command( Data = 0x15n )`, where n [0…FF] is the desired threshold value. However, with excelli*SCAN* scan heads the PosAck threshold value cannot be saved for subsequent new starts or resets. this is because ***UpdatePermanentMemory*** (see page 16) has no effect.



* with digital control and PosAck signal transmission,
for example, SCANLAB scan heads with i*DRIVE* technology

3

PosAck signal for non-excelli*SCAN* and excelli*SCAN* scan heads.

2.2.4 Returned Data Signal Differences: New Status Value HEAD_BUSY

New for RTC6 boards together with i*DRIVE* systems is the returnable status value HEAD_BUSY (bit #23).

This status value only gets returned by excelli*SCAN* scan heads. It can be queried by the **get_status** control command, see also the RTC6-Manual Chapter 6.4.3 "List Execution Status", page 98.

A prerequisite for this is that **set_scanahead_params** with valid parameters has been called. That is, the RTC6 board is configured to operate excelli*SCAN* scan heads.

The HEAD_BUSY status value is returned when:

- A list is being processed
- A list has been finished, but scan head output is still in progress (temporal offset = `PreviewTime`)
- The marking's final still-in-progress `LaserOffDelay` has not yet expired, that is, so long as the laser is still on

This allows waiting until the actual end of marking before an equipment controller closes a shutter or power is cut to the scan head, etc.

In contrast, if the laser was manually switched on (**laser_signal_on**) after the list ended but before expiration of `PreviewTime`, then its LASEROFF is not be waited for.

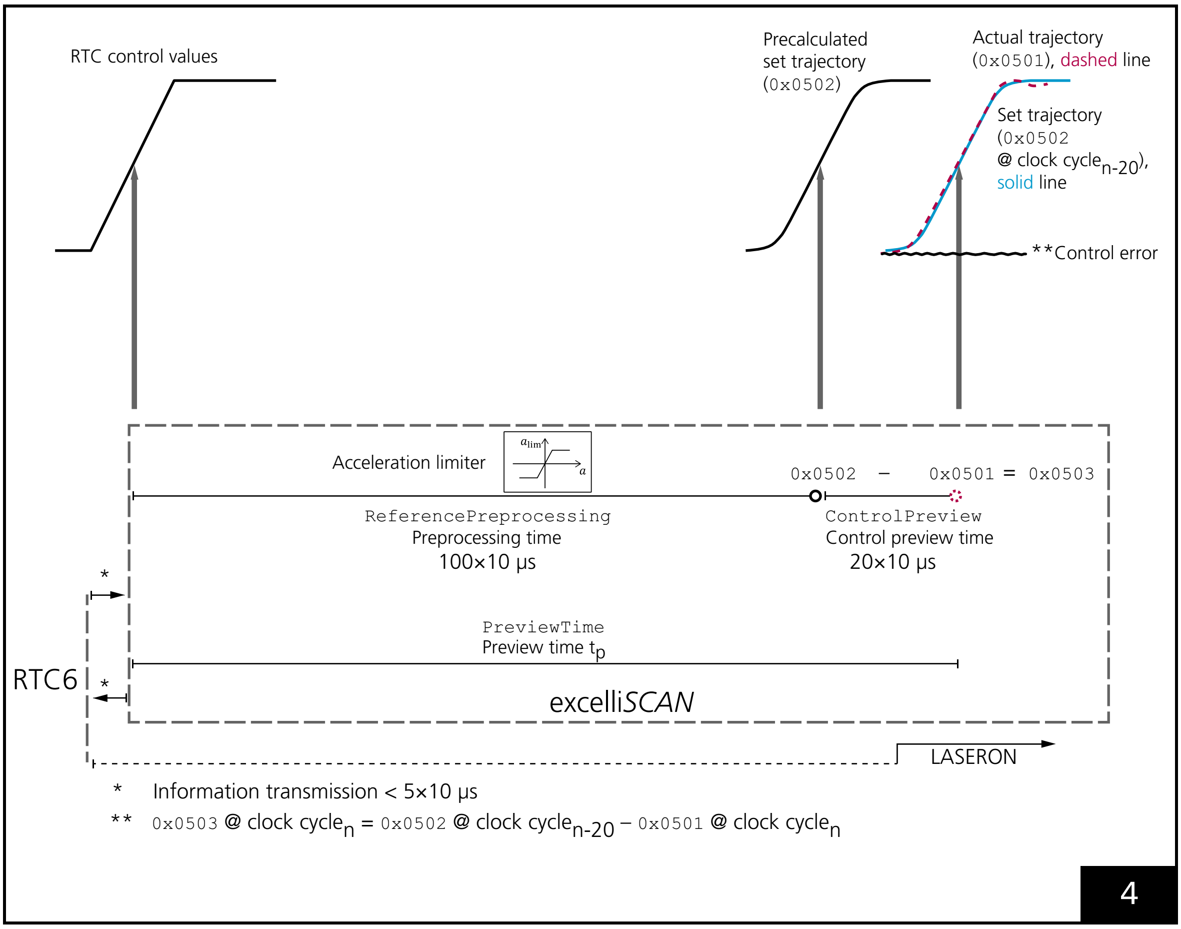
2.2.5 Returned Data Signal Differences: Change of Meaning for 0x0502 Signal

The value `0x0502` returned by excelli*SCAN* scan heads has a different meaning than for non-excelli*SCAN* scan heads, see chapter 2.3 "excelliSCAN-specific Changes and Novelties with control_command" and Chapter 2.2.7 "Timing Diagram of excelliSCAN Scan Head with SCANahead Control", page 13.

2.2.6 New Returned Data Signal: New Type for SetControlDefinitionMode

In response to the **SetControlDefinitionMode** command, excelli*SCAN* scan heads return the value 4 (*SCAN*ahead trajectory control), see page 16 in chapter 2.3 "excelliSCAN-specific Changes and Novelties with control_command".

2.2.7 Timing Diagram of excelli*SCAN* Scan Head with *SCAN*ahead Control


RTC control values
Precalculated set trajectory (0x0502)
Actual trajectory (0x0501), dashed line
Set trajectory (0x0502 @ clock cycle$_{n-20}$), solid line
**Control error
Acceleration limiter
a_{lim}
a
0x0502 − 0x0501 = 0x0503
ReferencePreprocessing
Preprocessing time
100×10 µs
ControlPreview
Control preview time
20×10 µs
PreviewTime
Preview time t_p
*
*
RTC6
excelli*SCAN*
LASERON
* Information transmission < 5×10 µs
** 0x0503 @ clock cycle$_n$ = 0x0502 @ clock cycle$_{n-20}$ − 0x0501 @ clock cycle$_n$
4

Timing diagram of excelli*SCAN* scan head.

Note: In the timing diagram (figure 4), specified absolute time values are for the excelli*SCAN*.

For excelli*SCAN* scan heads, the signal `0x0502` is the already acceleration-limited precalculated set position (angular position). This value can be queried with `control_command( Data = 0x0502 )`.

This `0x0502` signal is delayed with respect to RTC control values by `ReferencePreprocessing` (excelli*SCAN*: 100 × 10 μs clock cycles, that is, 1.0 ms).

Each precalculated set position value `0x0502` is reached after the control preview time `ControlPreview` (excelli*SCAN*: exactly 20 × 10 μs clock cycles, that is, 0.2 ms).

Therefore, the course of actual positions `0x0501` is delayed, namely in regards to the

- precalculated set positions `0x0502` by 0.2 ms
- RTC control values by 1.2 ms (`PreviewTime`)

For excelli*SCAN*, the signal `0x0503` (position error) is the difference between the precalculated set position `0x0502` (at 1.0 ms after the RTC control value) and the actual position `0x0501` at 1.2 ms after the RTC control value (and 0.2 ms after the precalculated set value). This value can be queried by `control_command( Data = 0x0503 )`.

The galvanometer scanner position error – even during acceleration phases – is very small (that is, practically a straight line). You can therefore select a very small PosAck threshold value (in comparison to conventional scan heads) without the limit value being exceeded and the PosAck bit getting set.

As the example timing diagram shows, the position error is calculated as follows:
`0x0503` @ clock cycle$_{n}$ =
`0x0502` @ clock cycle$_{n-20}$ – `0x0501` @ clock cycle$_{n}$.
Users who intend to calculate the control error themselves from log files have to observe this relationship.

With logging software, the signal `0x0503` is therefore depicted as x-axis-parallel line (that is, nearly horizontal line at 0).

2.3 excelli*SCAN*-specific Changes and Novelties with control_command

<table>
<tr><th><b>Ctrl Command</b></th><th colspan="4"><b>control_command</b></th></tr>
<tr><td>Function</td><td colspan="4">Sends a control command to an excelli<i>SCAN</i> scan head (firmware versions >5050).</td></tr>
<tr><td>Call</td><td colspan="4"><code>control_command(Head, Axis, Data)</code></td></tr>
<tr><td rowspan="12">Parameters</td><td><code>Head</code></td><td colspan="3">See RTC6-Manual.</td></tr>
<tr><td><code>Axis</code></td><td colspan="3">See RTC6-Manual.</td></tr>
<tr><td rowspan="10"><code>Data</code></td><td colspan="3">See RTC6-Manual.</td></tr>
<tr><td><b><i>Code</i></b>$_H$</td><td colspan="2"><b><i>Command and parameter values (Code</i></b>$_L$<b><i>)</i></b></td></tr>
<tr><td rowspan="8">05$_\mathrm{H}$</td><td colspan="2"><b><i>SetMode</i></b>: See RTC6-Manual. As described there:
– Each <i>Code</i>$_L$ parameter value corresponds to a specific data type.
– The standard setting is <i>Code</i>$_L$ = 00$_\mathrm{H}$ (XY2-100 status word).
excelli<i>SCAN</i> scan systems have an SL2-100 interface. They therefore return a signed 20-bit status value.</td></tr>
<tr><td><b><i>Code</i></b>$_L$</td><td><b><i>Data type returned by scan system</i></b></td></tr>
<tr><td>01$_\mathrm{H}$</td><td>For NON-excelli<i>SCAN</i> and excelli<i>SCAN</i> scan heads, the following applies:
Actual position (angular position of galvanometer scanners).
Bits [-2^{19} … $2^{19}-1$].
See Chapter 2.2.7 "Timing Diagram of excelliSCAN Scan Head with SCANahead Control", page 13 and figure 4.</td></tr>
<tr><td>02$_\mathrm{H}$</td><td>For NON-excelli<i>SCAN</i> scan heads, the following applies:
Set position (galvanometer scanners' angular position).
Bits [-2^{19} … $2^{19}-1$].
For excelli<i>SCAN</i> scan heads, the following applies:
Precalculated set position (angular position of galvanometer scanners).
Bits [-2^{19} … $2^{19}-1$].
See Chapter 2.2.7 "Timing Diagram of excelliSCAN Scan Head with SCANahead Control", page 13 and figure 4.</td></tr>
<tr><td>03$_\mathrm{H}$</td><td>For NON-excelli<i>SCAN</i> and excelli<i>SCAN</i> scan heads, the following applies:
Position error (= set position – actual position).
Bits [-2^{19} … $2^{19}-1$].
See Chapter 2.2.7 "Timing Diagram of excelliSCAN Scan Head with SCANahead Control", page 13 and figure 4.</td></tr>
<tr><td>15$_\mathrm{H}$</td><td>Axis 1 (X-axis): Temperature of the servo board (ISB).
Axis 2 (Y axis): Temperature of the servo add-on board for evaluating the encoder signals.</td></tr>
<tr><td>64$_\mathrm{H}$</td><td>As of firmware version 5100.
See Chapter 2.5 "Advanced Settings for "Spot Distance Control"", page 20:
<code>TimeShift</code> relative to the SCANLAB default offset.</td></tr>
<tr><td>65$_\mathrm{H}$</td><td>As of firmware version 5100.
See Chapter 2.5 "Advanced Settings for "Spot Distance Control"", page 20:
total <code>TimeShift</code>.</td></tr>
</table>

<table>
<tr><th>Ctrl Command</th><th colspan="4">control_command</th></tr>
<tr><td rowspan="10">Parameters
(cont'd)</td><td rowspan="10"></td><td>$0A_H$</td><td colspan="2"><b><i>UpdatePermanentMemory</i></b>.
This command has no effect on excelli<i>SCAN</i> scan heads.</td></tr>
<tr><td rowspan="3">$0E_H$</td><td colspan="2"><b><i>SetControlDefinitionMode</i></b>
As described in the RTC6-Manual. Additionally, the value 4 is the tuning type for excelli<i>SCAN</i>s.</td></tr>
<tr><td><b><i>Code</i></b>$_L$</td><td><b><i>Value returned by scan head</i></b></td></tr>
<tr><td>00_H…
03_H</td><td>Bits #4…7 Tuning type:
(…)
= 4: <i>SCAN</i>ahead servo control.</td></tr>
<tr><td>11_H</td><td colspan="2"><b><i>SelectControlDefinition</i></b>.
This command has no effect on excelli<i>SCAN</i> scan heads.
Only one tuning is available for excelli<i>SCAN</i> scan heads.</td></tr>
<tr><td>12_H</td><td colspan="2"><b><i>SetPositionScale</i></b>.
This command has no effect on excelli<i>SCAN</i> scan heads.
With excelli<i>SCAN</i> scan heads, the effective calibration cannot be changed.</td></tr>
<tr><td>40_H</td><td colspan="2"><b><i>ResetPOSACK</i></b>.
For excelli<i>SCAN</i> scan heads, this command lets you reset the PosAck warning. The only allowed parameter value for this command is Code_L = $3A_{H.}$ See also RTC6-Manual, Chapter 8.1.6 ”Configuring the PosAcknowledge Threshold Value“, page 208 and the Notes on <b>control_command</b> as well as in this document Chapter 2.2.3 ”Returned Data Signal Differences: PosAck Status Signal Behavior after Limit Exceedance“, page 11.
Note: Because the <b><i>UpdatePermanentMemory</i></b> command has no effect on excelli<i>SCAN</i> scan heads, the PosAck threshold value cannot be saved for subsequent new starts or resets.</td></tr>
<tr><td>50_H</td><td colspan="2">As of firmware version 5100.
See Chapter 2.5 ”Advanced Settings for “Spot Distance Control”“, page 20: dT_{high}.</td></tr>
<tr><td>51_H</td><td colspan="2">As of firmware version 5100.
See Chapter 2.5 ”Advanced Settings for “Spot Distance Control”“, page 20: dT_{low}.</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td>Comments</td><td colspan="4"><i>General notes:</i>
• Firmware version a.bb.c is read as abbc, for example, 5100 corresponds to version 5.10.0, and 5054 to version 5.05.4.</td></tr>
<tr><td>Version info</td><td colspan="4">As of version DLL 601.</td></tr>
<tr><td>References</td><td colspan="4"><b>get_value</b>, <b>get_values</b>, <b>get_head_status</b>, <b>set_trigger</b>, <b>set_trigger4</b>, <b>get_waveform</b>, <b>get_last_error</b></td></tr>
</table>

2.4 Notes on Wobbling with excelli*SCAN* Scan Heads

Wobbel mode combines the output position's regular linear motion (that is, the underlying marking) with an elliptical or figure-of-8 motion (= wobbel shape). This causes the laser beam to be guided spirally through the image field. Here, both accelerations and both speeds are superimposed.

Very demanding wobbel markings with the excelli*SCAN* scan head are achievable, provided that neither its maximum acceleration A_{max} nor its maximum speed V_{max} are exceeded. The following applies:

$$A_{Marking} + A_{Wobbel} \le A_{max}$$

and

$$V_{Marking} + V_{Wobbel} \le V_{max}$$

Both the maximum acceleration A_{max} and maximum speed V_{max} are factory-set in the excelli*SCAN* scan head.

They are queried by **get_scanahead_params**(`HeadNo` = 256).

The values are returned as programming bits[1] relative to RTC clock cycles:

- A_{max} as programming bits/clock cycle²
- V_{max} as programming bits/clock cycle.

2.4.1 Converting Queried Values

In practical application, such data should be in the form of values in m/s and m/s² for the image plane.

To convert maximum acceleration A_{max} from programming bits[1]/clock cycle² to m/s²[2]:

$$\frac{A_{max}\ [\text{bits/clock cycle}^2]}{K\ [\text{bits/mm}]} \times \frac{0.001\ [\text{m/mm}]}{(0.00001)^2\ [\text{s}^2/\text{clock cycle}^2]}$$

To convert maximum speed V_{max} from programming bits[1]/clock cycle to m/s:

$$\frac{V_{max}\ [\text{bits/clock cycle}]}{K\ [\text{bits/mm}]} \times \frac{0.001\ [\text{m/mm}]}{0.00001\ [\text{s/clock cycle}]}$$

Examples:

- excelli*SCAN* 14: A maximum acceleration of 39 bits/clock cycle² and a calibration factor of K ≈ 7700 bits/mm[3] equate to a maximum acceleration in the image plane of 50600 m/s².
- excelli*SCAN* 14: A maximum speed of 2465 bits/clock cycle and a calibration factor of K ≈ 7700 bits/mm[3] equate to a maximum speed in the image plane of 32 m/s.

[1] Not: galvanometer scanner control bits! Programming bits are coordinates in the image field. Galvanometer scanner control bits correspond to a galvanometer scanner rotation angle.

[2] Query `Amax` is by `get_scanahead_params`(`HeadNo` = 256). Observe the units specified in the command description for **get_scanahead_params**.

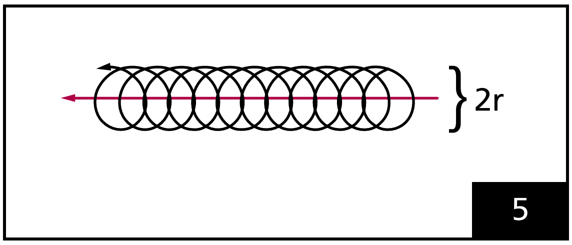
[3] Theoretically assumed example value that might occur for a calibration of 11.7° mechanical, an objective focal length of 160 mm and a 1-to-1 correction file. Actual values are listed in the `*_ReadMe.txt` files that SCANLAB supplies along with the correction file.

2.4.2 Estimating Maximum Wobbel Frequencies

The following two scenarios illustrate how users can estimate a sensible value for the maximum wobbel frequency f_{max}.

Here, it is assumed that a desired wobbel line thickness is to be achieved using the “circle” wobbel shape, see figure 5.

- The wobbel line thickness is twice the radius of the “circle” wobbel shape (“2r” in figure 5).
- The “circle” wobbel shape radius r is defined in a parameter of the appropriate RTCcommand. The following applies:
  r = `Transversal` = `Longitudinal`,
  that is,
  - **set_wobbel_mode** (`Transversal` = r, `Longitudinal` = r, `Freq` = f_{max}, `Mode` = 0)
    or,
  - **set_wobbel** (`Transversal` = r, `Longitudinal` = r, `Freq` = f_{max}).



Marking vector (red) and “circle” wobbel shape (black).
2r indicates the resulting wobbel line thickness.

Notes

- Nevertheless, users might define wobble command parameter values for Amplitude (`Transversal`, `Longitudinal`) and Frequency (`Freq`) so unfavorable that set acceleration exceeds A_{max}. Then the desired wobbel line thickness is not achieved, because marking executes at A_{max}.

Scenario A) Marking Vector and “Circle” Wobbel Shape

At a constant speed, acceleration is null.

If the marking vector has reached the specified marking speed – after an acceleration phase – then the following applies:

$$A_{Marking} = 0$$

together with (as previously mentioned)

$$A_{Marking} + A_{Wobbel} \leq A_{max}$$

Hence, the total A_{max} of the excelli*SCAN* scan head can be used for the wobbel shape[1].

With *uniform circular motions*, the following applies to circular acceleration a:

$$a = r \times (2\pi \times f)^2$$

Here, r is the circle radius and frequency f is the number of circular iterations per unit of time (note: $2\pi \times f$ is the circle frequency ω).

By setting a to the excelli*SCAN* scan head’s maximum acceleration A_{max} and solving for frequency f, we obtain the maximum wobbel frequency f_{max} for the marking vector:

$$f_{max} = \frac{1}{2\pi}\sqrt{\frac{A_{max}}{r}}$$

Notes

- To apply the above equation for an ellipse, set Radius r to the longer of the two half axes.
- To check whether the maximum speed of the excelli*SCAN* scan head is exceeded, use the following equation:

  $$V_{Marking} + V_{Wobbel} \leq V_{max}$$

  Whereby the following applies to the trajectory speed V_{Wobbel} of a circle during wobbling:

  $$V_{Wobbel} = 2\pi \times r \times f_{Wobbel}$$

[1] Valid not only for “circle”, but also generally for any wobble shape (because $A_{Marking} = 0$).

Scenario B) Arc Element and “Circle” Wobbel Shape

For the circular acceleration $A_{Marking}$ of an arc element the following applies:

$$A_{Marking} = \frac{V_{Marking}^{2}}{r_{Arc}}$$

The acceleration available for wobbel motion is correspondingly reduced by $A_{Marking}$.

For a “circle” wobbel shape – using the equation in Scenario A – the maximum wobbel frequency f_{max} is:

$$f_{max} = \frac{1}{2\pi}\sqrt{\frac{(A_{max} - A_{Marking})}{r_{Wobbel}}}$$

Notes

- To apply the above equation for an ellipse, set Radius r to the longer of the two half axes.
- The trajectory speeds of arc element and wobbel shape are superimposed in accordance with:

  $$V = V_{Marking} + 2\pi \times f_{Wobbel} \times r_{Wobbel}$$

2.5 Advanced Settings for "Spot Distance Control"

For the "Spot Distance Control" functionality of "Automatic Laser Control" (**set_auto_laser_control**(`Ctrl` = 7)), *SCAN*ahead systems like the excelli*SCAN* provide temporally corrected position and speed signals.

SCANLAB automatically corrects temporal offsets known in advance. A typical value for an excelli*SCAN* 14 are 6 × 10 µs clock cycles.

Not known in advance are temporal offsets for example, caused by the laser reaction times. These appear in the marking result as varying spot distances in acceleration and deceleration phases.

To compensate for these offsets, the following functionalities are available with *SCAN*ahead systems as of Firmware 5.10.0 (see Comments, page 16):

- "Setting the Temporal Offset"
- "Reading Back the Temporal Offset"

2.5.1 Setting the Temporal Offset

The position and speed signals used for the "Spot Distance Control" functionality can be temporally shifted by `TimeShift` (in seconds), see also Chapter 2.5.3 "Identifying Suitable Temporal Offset Values", page 20[1]:

- ±8 × 10 µs clock cycles max.
- 160 µs/2^{16} resolution
- Has an additive effect on the signal propagation time compensation preset by SCANLAB
- With positive values, the signals are shifted forward in time, with negative values, they are shifted backward

`TimeShift` must be converted into the values dT_{high} and dT_{low}:

$$dT = 2^{15} + \mathtt{TimeShift} \times \frac{2^{16}}{160\ \mu s}$$

Allowed value range for dT: 0…2^{16}–1.

$$dT_{high} = (dT >> 8)\ \&\ 0xff$$

$$dT_{low} = dT\ \&\ 0xff$$

The values for dT_{high} and dT_{low} are sent to the SCANahead system as follows:

- `control_command`(0x5000 + dT_{high})
- `control_command`(0x5100 + dT_{low})

2.5.2 Reading Back the Temporal Offset

The *SCAN*ahead system returns the temporal offset set in units of [2^{16}/160 µs] after:

- `control_command`(0x564)
  temporal offset set by the user
- `control_command`(0x565)
  total temporal offset

Result in s = **get_value**(…) × 160 µs/2^{16}.

2.5.3 Identifying Suitable Temporal Offset Values

A suitable value for `TimeShift` can be found with marking experiments by changing `TimeShift` until the variation of the pulse distances in acceleration and deceleration phases is minimal.

[1] "Signal propagation time compensation".

3 RTC6 *SCAN*ahead Functions

3.1 Introduction

As in Chapter 2.1 "Introduction", page 6 already explained, the excelli*SCAN* scan head internally calculates a set trajectory with limited, constant accelerations, whereby the acceleration time depends on speed changes. Resultingly, the scanner and laser delays needed for taking acceleration time into account depend on speed.

The RTC6 board can automatically calculate the required delays. For this and other reasons, SCANLAB recommends the RTC6 board for operating excelli*SCAN* scan heads.

Two commands "activate" RTC6 *SCAN*ahead functionality for suitable control of excelli*SCAN* scan heads:

(1) **set_scanahead_params**`( Mode =1 )`

The RTC6 board queries relevant information from the scan head, processes it (correction file) and holds it for further usage. Additionally, the RTC6 board laser control section gets prepared for taking `PreviewTime` into account. This affects LASER_DIGITAL_OUT1/2 etc., see "Notes on Controlling Peripherals" on page 30.

(2) **activate_scanahead_autodelays**`(1)`

The RTC6 board is instructed:

- To use the results of (1).
- To ignore delays set in source code by **set_scanner_delays** and **set_laser_delays**.
- To automatically calculate (for high contour fidelity) optimal speed-dependent scanner delays and laser delays[1].
- To also set these dynamically in real time during list execution.
- To likewise automatically calculate all required parameters when Sky Writing is used (for example, run-in and run-out motions), see also "Notes on Sky Writing", Seite 29.

Thus, users can effortlessly and immediately create markings with high contour fidelity [2] (without the need to determine and optimize delays).

Of course, users are also free to tailor marking results using additional suitable RTC6 commands:

- *Scanner delays* are affected by the `CornerScale` and `EndScale` parameters[3] of the two commands **set_scanahead_line_params** and **set_scanahead_line_params_list**.
  The effects of these parameters are shown in figure 6.
  You can choose between optimal contour fidelity and optimal process speed at the cost of contour fidelity (*process optimization*).
- *Laser delays* are affected by the `AccScale`[4] parameter of the two commands **set_scanahead_line_params** and **set_scanahead_line_params_list**.
  The effect of this parameter is shown in figure 7 which demonstrates a line marked using three different parameter values: the energy deposition at line ends is influenced by partially or wholly show/hide acceleration phases.
- Also available for *fine-tuning* laser delays are the RTC6 commands **set_scanahead_laser_shifts** and **set_scanahead_laser_shifts_list**. They let you set a temporal offset for the laser switching time points, for example, to compensate signal propagation times[5].

[1] For all commands that require delays, that is, jump, mark, arc commands.

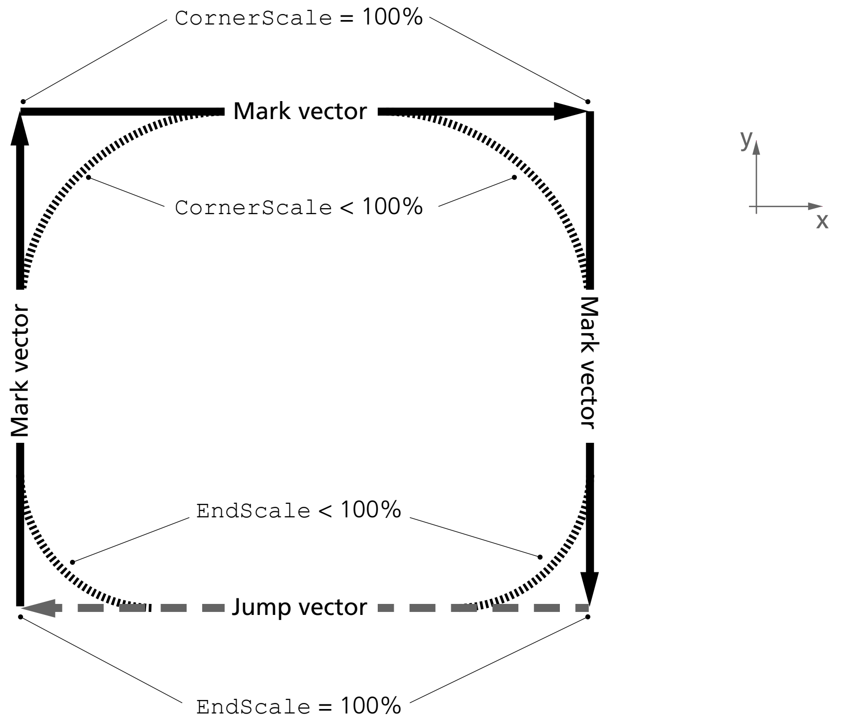
[2] Provided that the specified marking speed makes sense. However, the jump speed is automatically reduced to suitable values.

[3] These parameters only affect the galvanometer scanner positioning. They *do not* affect points in time when the laser is to be switched on and off.

[4] This parameter only affects points in time when the laser is to be switched on and off. It *does not* affect the galvanometer scanner positioning!

[5] The following applies for ≥ DLL 605: the transport delay from the RTC6 board to the scan head (20 µs) is automatically taken into account.

Parameter `CornerScale` and `EndScale` affect the scanner delays

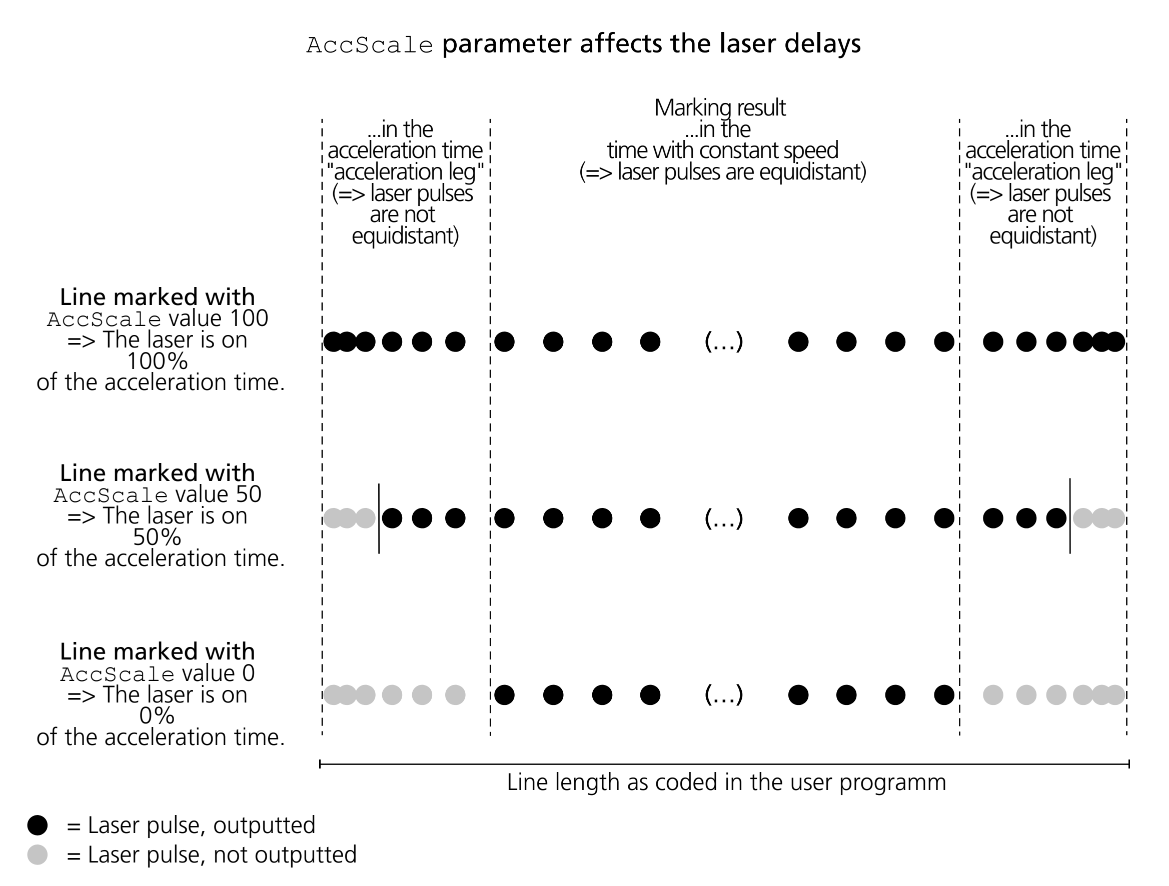


`CornerScale` This parameter controls the marking accuracy
at mark vector to mark vector alternations (“rounding of corners”).
100% = sharp corners.

`EndScale` This parameter controls the marking accuracy
at mark vector to jump vector alternations (”line end”) and
at jump vector to mark vector alternations (”line start”).
100% = straigth line ends/line starts.

6

set_scanahead_line_params and **set_scanahead_line_params_list**: the parameters `CornerScale` and `EndScale` affect the scanner delays.



AccScale parameter affects the laser delays

7

set_scanahead_line_params and **set_scanahead_line_params_list**: the parameter `AccScale` affects laser delays.

With `AccScale` values < 100 the marked line lengths are shorter than they are actually coded in the user program. With these values the laser is not always on during acceleration times. That is, portions of the acceleration legs are not marked. The acceleration leg lengths are proportional to the square of the acceleration time.
The portion of each acceleration leg which is *not marked* corresponds to (1 – `AccScale` value / 100)^2 × 100%.
For example, 25% of each acceleration leg is not marked with an `AccScale` value 50.

Note: To achieve the line lengths as coded in the user program *completely and with equidistant laser pulse spacing*, you must use Sky Writing.

3.2 Quick Migration Guide

This chapter gives advanced users the following practical information for quickly transitioning to an excelli*SCAN* scan head with an RTC6 board:

- Example source code for a user program, see Chapter 3.2.1 ”Example Source Code“, page 25
- Tips on adapting existing RTC user programs for the excelli*SCAN*, see Chapter 3.2.2 ”Modifying Existing User Programs for excelliSCAN Scan Heads – Basic Steps“, page 26
- In addition it is briefly mentioned, which restrictions exist due to not usable functionalities, see Chapter 3.2.3 ”Modifying Existing User Programs for excelliSCAN Scan Heads – Observing Restrictions“, page 27

Further details are provided starting with Chapter 3.3 ”Modifying Existing User Programs – Further Steps, Background Info on SCANahead and excelliSCAN Functions“, page 28.

3.2.1 Example Source Code

The following example code snippets illustrate basic elements of an RTC6 user program to initialize the `RTC6DLL.dll` and RTC6 board (that is, it is not a complete program code).

Because this example uses implicit linking, the files `RTC6impl.h` and `RTC6DLL.lib` are required. Additionally, the program needs to be able to load `RTC6DLL.dll`. If the operating system can not find `RTC6DLL.dll` during program start, then it responds with an error message and terminates the program.

```
// SAMPLE SOURCE CODE SNIPPETS: TO MARK A SQUARE.
//
// RTC6 header file for implicit linking to RTC6DLL.dll.
// also link to Visual C++ import library RTC6DLL.lib, to compile an executable
#include “RTC6impl.h”

// scanahead line params
CornerScale = 100;    // polyline: 100%: max precision, 0% min execution time
EndScale = 100;       // mark, jump: 100%: max precision, 0% min execution time
AccScale = 100;       // 100%: laser is on during acceleration and deceleration
                      //   0%: laser is off during acceleration and deceleration

// enable/disable variable polygon delay or sky writing
VariablePolygonDelay = 0;
SkyWriting = 0;
int main()
{

// initialize
init_rtc6_dll();
if( 0 != load_program_file( NULL ) ) { return -1; }  // load RTC firmware
if( 0 != load_correction_file( NULL, 1, 2 ) ) { return -2; }
set_mark_speed_ctrl( 25000 );
set_jump_speed_ctrl( 50000 );

// laser control
set_laser_control( 0 );                              // high active laser signals
set_laser_mode( 1 );                                 // mode 1 supports YAG
set_firstpulse_killer( 640 );
set_laserpulses_ctrl( 320, 128 );

// scanahead: calculate delays automatically. excelliSCAN scan head must be connected and powered.
set_scanahead_params( 1, 1, 1, 0, 0, 0 );            // place anywhere after load_correction_file
activate_scanahead_autodelays( 1 );
set_scanahead_line_params( CornerScale, EndScale, AccScale );
set_scanahead_laser_shifts( 1600, 1600 );             // fine tuning LaserON and LaserOFF. 1 bit = 1/64
µs

// sky writing and variable polygon delay
set_delay_mode( VariablePolygonDelay, 0, 2^30, 0, 0 );
set_sky_writing_para( SkyWriting, 0, 0, 0 );
set_sky_writing_mode( 3 );
set_sky_writing_limit( 0.5 );

// create and execute list
set_start_list( 1 );
jump_abs( 50000, -50000 );
mark_abs( 50000, 50000 );
mark_abs( -50000, 50000 );
mark_abs( -50000, -50000 );
mark_abs( 50000, -50000 );
jump_abs( 0, 0 );
set_end_of_list();
execute_list( 1 );
}
// END OF SAMPLE SOURCE CODE SNIPPETS.
```

3.2.2 Modifying Existing User Programs for excelli*SCAN* Scan Heads – Basic Steps

To operate an excelli*SCAN* with an RTC6 board, SCANLAB recommends modifying existing user programs as described below. For more details, see the following chapters, as well as the RTC6-Manual. It is assumed that it is a complete RTC6 or RTC5 user program for operating a SCANLAB scan head with conventional control, for example, an intelli*SCAN*. user program

(1) After the **load_correction_file** command at the beginning of your user program, be sure to insert the command **set_scanahead_params** (for parameters, see the command description).

This activates the *SCAN*ahead functionality of the RTC6 board (incl. laser control!). `Mode = 1` initiates direct querying of the scan head for scan-head-specific information on `PreviewTime`, `Vmax` and `Amax`. Therefore, an excelli*SCAN* scan head must be connected to the RTC6 board and already and switched-on during user program runtime.

(2) Insert **activate_scanahead_autodelays**`(1)`.
This switches on automatic calculation of scanner delays and laser delays.
All **set_scanner_delays** and **set_laser_delays** commands in subsequent code lines have no effect, see also notes for **activate_scanahead_autodelays**.

(3) If the original program is an RTC5 user program, then also make the following changes (for more detailed and comprehensive information, see the RTC6-Manual):

- change `init_rtc5_dll` to `init_rtc6_dll`
- change `rtc5_count_cards` to `rtc6_count_cards`
- change `free_rtc5_dll` to `free_rtc6_dll`
- do *NOT*[1] change `set_rtc5_mode` and `set_rtc4_mode`
  to `set_rtc6_mode`.
  If neither `set_rtc4_mode` nor `set_rtc5_mode` is called, then the default state with RTC5 boards is `set_rtc5_mode`. Insert `set_rtc5_mode` into the RTC6 user program to automatically convert certain parameter values from RTC5 to RTC6 units.

These modifications should already suffice for successful execution of your user program.

Any source code sections for variable polygon delays and Sky Writing should thereby also be runnable.

Check whether your code is affected by the restrictions mentioned next, in Chapter 3.2.3 ”Modifying Existing User Programs for excelliSCAN Scan Heads – Observing Restrictions”, page 27.

[1] This section is intended as a quick guide: for detailed information (for example, 3D applications etc.), see the RTC6-Manual.

3.2.3 Modifying Existing User Programs for excelli*SCAN* Scan Heads – Observing Restrictions

While continuing to modify your user program for the excelli*SCAN* scan head, keep in mind the following:

Non-Usable Functionalities of the RTC6 Board and RTC6 DLL

- *After activation of SCAN*ahead *functionality by* **set_scanahead_params**, the following RTC6 functionalities are *not* usable (status ≤ DLL 615):
  - **load_varpolydelay**
  - **laser_on_pulses_list** cannot be used for excelli*SCAN* scan heads, if external signal pulses are required at DIGITAL IN1 (**laser_on_list** can be used with excelli*SCAN*)

Non-Usable i*DRIVE* Functionalities

- The following i*DRIVE* functions are not available for excelli*SCAN* scan heads:
  - Changing the effective calibration **(SetPositionScale)**
    `control_command( Data = 0x12xx )`
- The following i*DRIVE* functions are not available for excelli*SCAN* scan heads with firmware version ≤ 5.05.4 (see Comments, page 16):
  - Switching between tunings (**SelectControlDefinition**)
    `control_command( Data = 0x11xx )`

  See Chapter 2.3 ”excelliSCAN-specific Changes and Novelties with control_command”, page 15.

3.3 Modifying Existing User Programs – Further Steps, Background Info on *SCAN*ahead and excelli*SCAN* Functions

While continuing to modify your user program for the excelli*SCAN* scan head, keep in mind the following:

- "Notes on Using Commands for Scanner Delays and Laser Delays", page 28
- "Notes on Arc Commands", page 29
- "Notes on Polygon Delays", page 29
- "Notes on Sky Writing", page 29
- "Notes on Controlling Peripherals", page 30
- "Notes on Unique Characteristics of Returned Data Signals", page 30

Notes on Using Commands for Scanner Delays and Laser Delays

When using commands with the excelli*SCAN* to synchronize scan motions and laser control signals, observe the following:

- All relevant *SCAN*ahead parameters (article-number-specific) are stored in the excelli*SCAN* scan head during manufacturing. Although **set_scanahead_params**`( Mode = 2 )` allows other values, we recommend only using the values queried from the excelli*SCAN* scan head. Mode 2 is primarily intended for software development without an attached excelli*SCAN* scan head.
- For excelli*SCAN* scan heads, the actual position of the scan motion lags the control values by `PreviewTime`, see figure 1[1]. For synchronous laser control, the laser control signals also need to take `PreviewTime` into account.
  This can be done manually by appropriately modifying parameters for the **set_scanner_delays** and **set_laser_delays** commands. But here, you must take the excelli*SCAN*'s speed-dependent acceleration time into account.
  The preferable alternative is to simply use the **activate_scanahead_autodelays** command for automatic calculation of scanner delays and laser delays. Then, the following applies:
  – The existing command parameters for **set_scanner_delays** and **set_laser_delays** are invalidated by `activate_scanahead_autodelays( Mode = 1 )` and not modified. They become immediately effective again if automatic delay calculation gets deactivated by `activate_scanahead_autodelays( Mode = 0 )`.
  – `set_scanahead_params( Mode = 1 or 2, HeadNo, TableNo, PreviewTime, Vmax, Amax )` defines parameters for automatic calculation of scanner delays and laser delays together with `activate_scanahead_autodelays( Mode = 1)`.
- If needed, throughput of laser-marked workpieces can be increased, if lower quality is acceptable ("process optimization"): see **set_scanahead_line_params**`( CornerScale, EndScale, AccScale )`.
  See also figure 6 and figure 7.
- For *fine-tuning* laser delays (for example, to compensate signal propagation times to the laser or to take laser system switching behavior into account), see **set_scanahead_laser_shifts** `( dLasOn, dLasOff )`.

[1] This temporal offset differs quantitatively and qualitatively from tracking error in scan heads without *SCAN*ahead control.

Notes on Arc Commands

- excelli*SCAN* scan heads have a nominal tracking error of 0[1]. Tracking-related artifacts (for example, necking during small, quick circular motions) do not occur so long as the maximum acceleration of the head is not exceeded (for example, when in the source code a too large speed in relation to the circular diameter has been specified).
- To estimate maximum acceleration within the image field, the maximum acceleration in programming bits[2] (image field coordinates) can be queried by `get_scanahead_params( HeadNo = 256 )`. For conversion to m/s^2, see Chapter 2.4.1 "Converting Queried Values", page 17.
- If you use **arc_abs**, **arc_abs_3d**, **arc_rel**, **arc_rel_3d**, note that these commands also apply rounding upon microstepping. Thus, the speed which the RTC6 commands to the excelli*SCAN* may differ from the programmed (in **set_mark_speed**) marking speed. In cases where the acceleration (resulting from the commanded speed) exceeds the maximum acceleration `Amax` of the excelli*SCAN*, then circles are smaller as programmed. The allowed speed range for a given circular radius can be estimated by the following rule of thumb:
  $v^2/r < 0{,}7$ `Amax`[3][4][5].
- When using **timed_arc_abs** and **timed_arc_rel**, it should be noted that these commands do not round during microstepping, provided execution time is specified in multiples of 10 μs. Instead the following applies:
  $v^2/r <$ `Amax`[3][4][5].

Notes on Polygon Delays

- On the excelli*SCAN*, variable polygon delays work similarly to scan heads without *SCAN*ahead control.
  To activate variable polygon scanner delays, you still must call **set_delay_mode** with `VarPoly` > 0.
  If automatic delay calculation (**activate_scanahead_autodelays**) is switched on, its parameters `EdgeLevel`, `MinJumpDelay` and `JumpLengthLimit` have no effect. But they become immediately effective if automatic delay calculation gets switched off.
  In contrast, the `DirectMove3D` parameter is always effective.
- For excelli*SCAN*, **load_varpolydelay** has no effect. No user-defined variable polygon delay tables are available.

Notes on Sky Writing

- For excelli*SCAN*, Sky Writing functionality has the same effect as on scan heads without *SCAN*ahead control. All three modes are selectable.
  To activate Sky Writing functionality, you must still call the command **set_sky_writing** or **set_sky_writing_para** with `Timelag` > 1/4 μs.
  If automatic delay calculation (**activate_scanahead_autodelays**) is switched on, then `Timelag` (except for the actual activation of Sky Writing) and the parameters `Nprev` and `Npost` have no effect. But they immediately become effective, if automatic delay calculation gets switched off.
  In contrast, the `LaserOnShift` parameter is always effective.
- If Sky-Writing is activated, only the parameter `EndScale`[6] is effective.
- The other parameters[6] `CornerScale` and `AccScale` are not applied – except there is *no* genuine Sky Writing movement:
  - If Sky-Writing Mode 3 is activated and the angular limit is undercut, `CornerScale` is effective.

[1] The output delayed by `PreviewTime` is not tracking error in the usual sense.

[2] See footnote (1) on page page 17.

[3] Query `Amax` by `get_scanahead_params( HeadNo = 256 )`. Observe the units specified in the command description for **get_scanahead_params**.

[4] Applies to frequencies ≤ 6.25 kHz; $f = v / (2\pi \times r)$.

[5] Valid for excelli*SCAN* with any objective. Insert either control bits or programming bits into the formula). See also first bullet in this list.

[6] See **set_scanahead_line_params**, **set_scanahead_line_params_list**.

Notes on Coordinate Transformations

When using commands to transform coordinates, observe the following:

- A coordinate transformation should *not* change the shape and size[1] of the marking, that is, the (2 x 2) total matrix M (see RTC6-Manual, Chapter 8.2 "Coordinate Transformations", page 210) is ideally a pure rotation matrix.

Notes on Controlling Peripherals

When using commands to control peripherals (LASER connector pins ANALOG OUT 1 and ANALOG OUT 2, EXTENSION 1 and EXTENSION 2 headers), observe the following.

If the *SCAN*ahead functionality of the RTC6 board has been activated by **set_scanahead_params** (`PreviewTime` > 0) the following applies:

(1) The control commands **write_io_port**, **write_8bit_port**, **write_da_x**, **write_da_1** and **write_da_2** are immediately executed ("asynchronous").

(2) The list commands **write_io_port_list**, **write_8bit_port_list**, **write_da_x_list**, **write_da_1_list** and **write_da_2_list** are executed delayed by the `Previewtime`, that is, simultaneous with the galvanometer scanners ("axissynchronous").

(3) For a "laser synchronous" output of a laser power control, **set_laser_power** must be used instead of the list commands from (2).

If the *SCAN*ahead functionality of the RTC6 board has not been activated (`PreviewTime` = 0), then asynchronous and axissynchronous execution (1) and (2) are identical.

With the new laser control mode of the RTC6 (`set_dsp_mode( 3 )`), LASERON delays during a series of short vectors can extend across one or several subsequent vectors. The RTC5-compatible synthetic scanner delays in DSP mode 2 are omitted. Output is then synchronous with each vector's corresponding LASERON delay.

With the RTC5-compatible laser control (`set_dsp_mode( 2 )`) axissynchronous and lasersynchronous execution are identical.

Notes on Unique Characteristics of Returned Data Signals

Some of the excelli*SCAN*'s returned data signals differ from those of other i*DRIVE* scan heads, for example, intelli*SCAN*.

- **PosAck Status Value with excelli*SCAN***
  If the threshold is exceeded, then the PosAck bit gets set to 0. It remains 0 until it gets set by `control_command( Data = 0x403A )`.
  See Chapter 2.2.3 "Returned Data Signal Differences: PosAck Status Signal Behavior after Limit Exceedance", page 11.
- **Position error value with excelli*SCAN***
  `control_command( Data = 0x0503 )`
  This is the difference between the precalculated set position (at 100 clock cycles after the RTC control value) and the actual position (at 120 clock cycles after the control value or 20 clock cycles after the set value).
  See Chapter 2.2.7 "Timing Diagram of excelliSCAN Scan Head with SCANahead Control", page 13.

[1] Quality losses are likely, if size changes more than 5%.

3.4 RTC6 Commands - Exclusively for *SCAN*ahead Systems

This chapter describes RTC6 commands intended exclusively for scan systems with *SCAN*ahead control, for example, the excelli*SCAN*.

This command type is identifiable by the name element “**_scanahead_**”.

All are also available as multi-board commands. Multi-board command names begin with the prefix **n_**.

Notes

- Universally usable RTC6 commands are described in the RTC6-Manual.

(1) con control command

(2) us undelayed short list command

Ctrl Command	activate_scanahead_autodelays
Function	Switches on or off the automatic (dynamic) calculation of the scanner delays and laser delays. Returns the current mode after the command has been executed.
Call	`CurrentMode = activate_scanahead_autodelays( Mode )`
Parameters	`Mode` As a signed 32-bit value. `-1`: Only returns the current mode. The current mode is not changed. `0`: Switches off automatic calculation. `1`: Switches on automatic calculation (`Mode = 1`) if **set_scanahead_params** has been called previously. Otherwise, `Mode` remains 0 after the command has been executed.
Result	Current mode. As an unsigned 32-bit value.
Comments	• If `Mode = 0`, then normal scanner delays and laser delays are used (see **set_scanner_delays** and **set_laser_delays**). If `Mode = 1`, then scanner delays and laser delays are calculated and set automatically. Values for `Mode = 0` do not get overwritten. After switch-off, they remain available in `Mode = 0` for usage (as with non-excelli*SCAN* scan heads). • **activate_scanahead_autodelays** also affects Sky Writing: the Sky Writing command parameters `Timelag` (except for the actual activation of Sky Writing), `Nprev` and `Npost` have no effect. However, they are going to have immediate effect once the automatic delay calculation is switched off. In contrast, the parameter `LaserOnShift` is always effective.
RTC4→RTC6	New command.
RTC5→RTC6	New command.
Version info	Available as of version DLL 600, OUT 600.
References	**activate_scanahead_autodelays_list**, **set_scanner_delays**, **set_laser_delays**

Undelayed Short List Command	activate_scanahead_autodelays_list
Function	As **activate_scanahead_autodelays**, but an undelayed short list command.
Call	`CurrentMode = activate_scanahead_autodelays_list( Mode )`
Parameters	`Mode` As an unsigned 32-bit value. For values, see **activate_scanahead_autodelays**.
Comments	• **activate_scanahead_autodelays_list** takes effect upon the next to-be-calculated delay. • `Mode = -1` is not available with **activate_scanahead_autodelays_list**.
RTC4→RTC6	New command.
RTC5→RTC6	New command.
Version info	Available as of version DLL 600, OUT 600.
References	**activate_scanahead_autodelays**

Ctrl Command	get_scanahead_params
Function	Queries certain parameters from the specified scan head. The attached scan head must be an excelli*SCAN*.
Restriction	**get_scanahead_params** can be executed only, if no list is currently being processed. Otherwise, the **get_last_error** return code gets set to `RTC6_BUSY` and error code 5 is returned.
Call	`Error = get_scanahead_params( HeadNo, &PreviewTime, &Vmax, &Amax )`
Parameters	`HeadNo` Scan head connector number. As an unsigned 32-bit value. Allowed values: = 1: Scan head connector 1. = 2: Scan head connector 2 (activation required). = 256: RTC6 board (no scan head connector). The parameters `Vmax` and `Amax` are returned converted to image field coordinates.
Returned parameter values	`PreviewTime` Pre-calculation time for galvanometer scanner control in [10 µs]. As a pointer to an unsigned 32-bit value. `Vmax` For `HeadNo` 1, 2: Velocity limit as control bits (galvanometer scanner rotation angle) [bits/10 µs]. For `HeadNo` 256: Velocity limit as programming bits (image field coordinates) [bits/10 µs]. As a pointer to an unsigned 32-bit value. `Amax` For `HeadNo` 1, 2: Acceleration limit as control bits (galvanometer scanner rotation angle) [bits/(10 µs)^2]. For `HeadNo` 256: Acceleration limit as programming bits (image field coordinates) [bits/(10 µs)^2]. As a pointer to a 64-bit IEEE floating point value.
Result	Error code. As an unsigned 32-bit value. Value Description 3 No excelli*SCAN* scan head is attached or its excelli*SCAN* tuning is not active (**get_last_error** return code: `RTC6_PARAM_ERROR`). 5 A list is currently active (**get_last_error** return code: `RTC6_BUSY`). 6 `HeadNo` = `0` or `HeadNo` >2 has been specified as parameter (**get_last_error** return code: `RTC6_PARAM_ERROR`). 8 The RTC6 board is not responding. Probably a program has not been loaded yet (**get_last_error** return code: `RTC6_TIMEOUT`). 11 A PCI error occurred (**get_last_error** return code: `RTC6_SEND_ERROR`).

Ctrl Command	**get_scanahead_params**
Comments	• The returned parameter values are also used by **set_scanahead_params**. • `Vmax` is the maximum angular velocity of the galvanometers. `Vmax` *is not* the marking speed in the image plane. The returned parameter value unit type is therefore [bits/10 µs] and *not* [bits/ms]. Distortion by the correction file is not yet taken into account in the returned parameter value (see **set_scanahead_params**). • `HeadNo` = `256` only returns meaningful values if **set_scanahead_params** was previously called. • The values returned by `HeadNo` = `256` are scaled. Do not use them with `set_scanahead_params( Mode = 2 )`because then they would be scaled again. • For dual *SCAN*ahead systems (for example, 2 excelli*SCAN* scan heads; requires Option “Second Scan Head Control”) it is recommended that the single *SCAN*ahead systems used have as similar properties as possible (aperture, lens, correction file): For automatic calculation of the scanner delays and laser delays, among other things, the parameter `Amax` of scan-system `HeadNo` is required. The delays are only calculated for this scan system, but are also used unchanged for the second one. If the two scan systems differ too much for example, in parameter `Amax`, unwanted deviations may occur between the markings.
RTC4→RTC6	New command.
RTC5→RTC6	New command.
Version info	Available as of version DLL 600, OUT 600. Last change with version DLL 605: data type of `Amax`.
References	**set_scanahead_params**, **get_last_error**

Ctrl Command	**set_scanahead_laser_shifts**
Function	Shifts the LaserOn signals and LaserOff signals temporally forward or back.
Call	`set_scanahead_laser_shifts( dLasOn, dLasOff )`
Parameters	`dLasOn` Time shift of LaserOn signals in [1/64 µs]. As a signed 32-bit value. `dLasOff` Time shift of LaserOff signals in [1/64 µs]. As a signed 32-bit value.
Comments	• **set_scanahead_laser_shifts** lets you *fine-tune* the laser delay, for example, to compensate laser signal propagation times or to take the laser system switching behavior into account. • If **set_scanahead_params** has not been already called and automatic calculation has not been activated (**activate_scanahead_autodelays** `Mode = 0`), then the values only get stored, but not applied. But they are applied as soon as required conditions are fulfilled. • There is a transport delay from the RTC6 board to the scan head of 20 µs. As of DLL 605 this is automatically taken into account. `dLasOn` and `dLasOff` parameter values optimized for < DLL 605 must be decreased by 20 µs to be suitable for ≥ DLL 605.
RTC4→RTC6	New command.
RTC5→RTC6	New command.
Version info	Available as of version DLL 600, OUT 600. Change as of version DLL 603, OUT 603, RBF 607: increased parameter resolution is now 1/64 µs instead of 1/2 µs.
References	**set_scanahead_laser_shifts_list**, **set_scanahead_params**, **activate_scanahead_autodelays**

Undelayed Short List Command	**set_scanahead_laser_shifts_list**
Function	Like **set_scanahead_laser_shifts**, but an undelayed short list command.
Call	`set_scanahead_laser_shifts_list( dLasOn, dLasOff )`
Parameters	`dLasOn` See **set_scanahead_laser_shifts**. `dLasOff` See **set_scanahead_laser_shifts**.
Comment	• **set_scanahead_laser_shifts_list** takes effect upon the next to-be-calculated delay.
RTC4→RTC6	New command.
RTC5→RTC6	New command.
Version info	Available as of version DLL 600, OUT 600.
References	**set_scanahead_laser_shifts**

Ctrl Command	set_scanahead_line_params
Function	Influences the quality of marking results at runtime. Smaller percent values increase throughput at the expense of quality.
Call	`set_scanahead_line_params( CornerScale, EndScale, AccScale )`
Parameters	`CornerScale` Corner sharpness with mark/mark sequences in percent. 100% = sharp corners. As an unsigned 32-bit value. `EndScale` Marking accuracy at mark/jump and jump/mark transitions. 100% = straight line ends. As an unsigned 32-bit value. `AccScale` Determines the portion of the acceleration *time* (not: distance traversed) in which the laser is active, in percent. 100% = entire acceleration time. As an unsigned 32-bit value.
Comments	• Only values from 0% to 100% are useful. Higher values do not improve quality (that is, corners cannot be even sharper than sharp), but instead only extend marking times. • If **set_scanahead_params** has not been already called, then the values only get stored, but not applied. • The parameters `CornerScale` and `EndScale` only affect the galvanometer scanner positioning. They *do not* affect points in time when the laser is to be switched on and off. • The parameter `AccScale` only affects points in time when the laser is to be switched on and off. It *does not* affect the galvanometer scanner positioning! • If Sky-Writing is activated, only the parameter `EndScale` is effective. • The other parameters `CornerScale` and `AccScale` are not applied – except there is *no* genuine Sky Writing movement: – If Sky-Writing Mode 3 is activated and the angular limit is undercut, `CornerScale` is effective. • The parameter `EndScale` is effective, if Sky Writing is activated (here, the following applies as well: smaller percent values increase throughput at the expense of quality).
RTC4→RTC6	New command.
RTC5→RTC6	New command.
Version info	Available as of version DLL 600, OUT 600.
References	**set_scanahead_line_params_list**

Undelayed Short List Command	**set_scanahead_line_params_list**
Function	Like **set_scanahead_line_params**, but an undelayed short list command.
Call	`set_scanahead_line_params_list( CornerScale, EndScale, AccScale )`
Parameters	`CornerScale` See **set_scanahead_line_params**. `EndScale` See **set_scanahead_line_params**. `AccScale` See **set_scanahead_line_params**.
Comment	• **set_scanahead_line_params_list** takes effect upon the next to-be-calculated delay.
RTC4→RTC6	New command.
RTC5→ RTC6	New command.
Version info	Available as of version DLL 600, OUT 600.
References	**set_scanahead_line_params**

Ctrl Command	set_scanahead_params
Function	Activates the RTC6-*SCAN*ahead functionality suitable to control an excelli*SCAN* scan head. The laser control and digital inputs/outputs for the peripherals are prepared for an excelli*SCAN* scan head, that is, delayed by `PreviewTime`. To have also the scanner delays and laser delays calculated automatically, **activate_scanahead_autodelays** must be called.
Restriction	**set_scanahead_params** can only be executed, if no list is currently active. Otherwise, the **get_last_error** return code gets set to `RTC6_BUSY` and error code 5 is returned. **set_scanahead_params** can only be executed, if the "SCANa" option is enabled. Otherwise, the **get_last_error** return code gets set to `RTC6_REJECTED` and error code 1 is returned.
Call	`Error = set_scanahead_params( Mode, HeadNo, TableNo, PreviewTime, Vmax, Amax )`
Parameters	`Mode` = 0: Deactivates the RTC6-*SCAN*ahead functionality. The additional parameters `TableNo, PreviewTime, Vmax, Amax` are not taken into account. Default after **load_program_file**. Mode, for example, for operating intelli*SCAN* scan heads. = 1: Queries parameters of the scan head with the specified `HeadNo` and applies them. The parameters `PreviewTime, Vmax, Amax` are not taken into account. See also comments that follow. Mode for operating attached excelli*SCAN* scan heads. = 2: Apply the parameters of this command. Mode, for example, for software development. An excelli*SCAN* scan head does not need to be connected. As an unsigned 32-bit value.
	`HeadNo` Number of the scan head connector. Allowed values: = 1: Scan head connector 1. = 2: Scan head connector 2 (activation required). As an unsigned 32-bit value.
	`TableNo` Allowed value range 1…8. A correction file should have been loaded by **load_correction_file**(…, `TableNo`, …). This is used for converting the parameters `Amax` and `Vmax` from control bits (galvanometer scanner rotation angle) into programing bits (image field coordinates). See **get_scanahead_params**. As an unsigned 32-bit value.
	`PreviewTime` Pre-calculation time for galvanometer scanner control in [10 µs]. This parameter is ignored in `Mode = 0` and `Mode = 1`. As an unsigned 32-bit value.
	`Vmax` Velocity limit as control bits (galvanometer scanner rotation angle) [bits/10 µs]. This parameter is ignored in `Mode = 0` and `Mode = 1`. As an unsigned 32-bit value.
	`Amax` Acceleration limit as control bits (galvanometer scanner rotation angle) [bits/(10 µs)^2]. This parameter is ignored in `Mode = 0` and `Mode = 1`. As a 64-bit IEEE floating point value.

Ctrl Command	**set_scanahead_params**
Result	Error code. As an unsigned 32-bit value. Value Description 1 “SCANa” option is not present. (**get_last_error** return code: `RTC6_REJECTED`). 3 No excelli*SCAN* scan head attached or excelli*SCAN* tuning is not active (**get_last_error** return code: `RTC6_PARAM_ERROR`). 5 A list is currently active (**get_last_error** return code: `RTC6_BUSY`). 6 `HeadNo` = `0` and `HeadNo` >2 (**get_last_error** return code: `RTC6_PARAM_ERROR`). 7 The determined scaling factor exceeds 16 or is below 1/16 (possibly no correction file was loaded). The scaling factor is set to 1. 8 The RTC6 board is not responding. Probably a program has not been loaded yet (**get_last_error** return code: `RTC6_TIMEOUT`). 11 A PCI error occurred (**get_last_error** return code: `RTC6_SEND_ERROR`)
Comments	• The `Vmax` and `Amax` parameters are used for automatic calculation of scanner delays and laser delays, see **activate_scanahead_autodelays**. • The values for `Vmax` and `Amax` are specified as control bits (galvanometer scanner rotation angles). Information is read-out from the correction file and used for an internal conversion into programing bits (sample values). Therefore, the correction file needs to be specified in the `TableNo` parameter. At conversion time, the correction file must have been already loaded onto the RTC6 board, but does not yet need to be assigned. The system cannot reliably detect correction files that have not been loaded. The `Vmax` and `Amax` values which have been converted into programing bits can be queried by **get_scanahead_params(** `HeadNo`= `256 )`. • For error return information, see **get_scanahead_params**. • `Mode = 1` results in aborted command execution (`Error = 3`) if no excelli*SCAN* is attached and active. All *SCAN*ahead functionality is deactivated. Parameters specified for `Mode = 2` are not applied! • **set_scanahead_params**`( Mode = 1)` cannot be executed when an automatic laser control is active that uses read-out data from an *iDRIVE* scan system, see **set_auto_laser_control**. Please deactivate temporarily.

Ctrl Command	set_scanahead_params
Comments (cont'd)	• **set_scanahead_params** waits until the last session has finished traversing, that is, the HEAD_BUSY signal (bit #23) from **get_status** is no longer set. In any case, **set_scanahead_params** waits no longer than 255 × [10 µs].
RTC4→RTC6	New command.
RTC5→RTC6	New command.
Version info	Available as of version DLL 600, OUT 600. Last change with version DLL 605: data type of `Amax`.
References	**get_scanahead_params**, **activate_scanahead_autodelays**, **get_last_error**, **get_status**, **load_correction_file**

Ctrl Command	set_scanahead_speed_control
Function	Controls the scanner delays either depending on the target speed or actual achievable speed.
Call	`set_scanahead_speed_control( Mode )`
Parameters	`Mode` Modus. As an unsigned 32-bit value.
Comments	• `Mode` with: – Bit #0 controls mark commands – Bit #1 controls jump commands – Bit #n = 0 controls the scanner delays depending on the target speed – Bit #n = 1 controls the scanner delays depending on the actual achievable speed • Default after **load_program_file**: `Mode` = 2.
RTC4→RTC6	New command.
RTC5→RTC6	New command.
Version info	Available as of version DLL 610, OUT 610. Last change with version OUT 611: adjustment of laser delays.
References	–

4 Change Index

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

Changes from document revision 1.0.7 to document revision 1.0.8

Name of chapter / command	Notes / Changes
Global	Document Revision 1.0.8 applies to RTC6 Software Package V1.6.1.
set_scanahead_laser_shifts	Editorial change. Command name corrected.
set_scanahead_laser_shifts_list	Editorial change. Command name corrected.

Notes