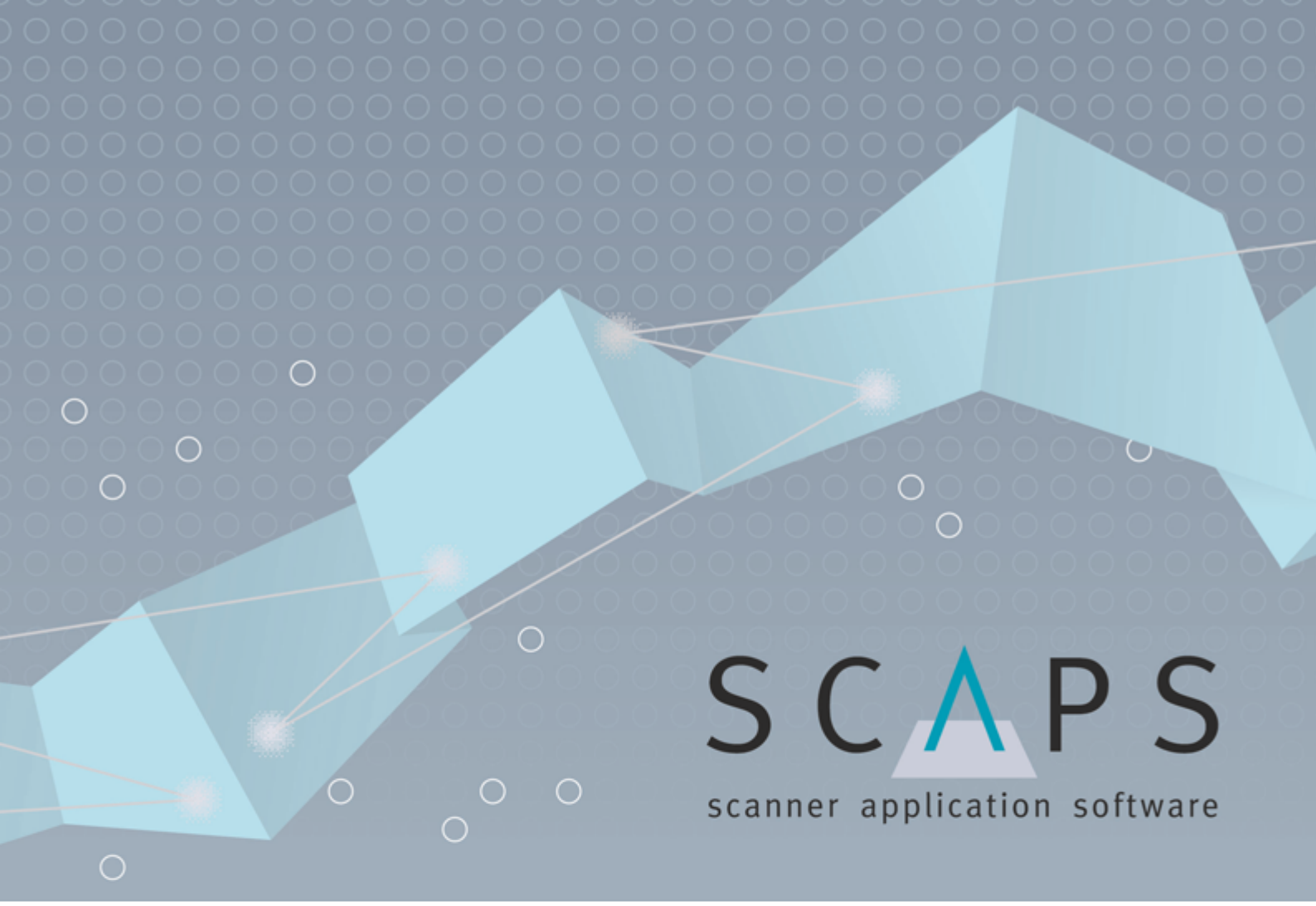


SAMLight Manual

Rev. 1.11.0, 19/11/2018

Table of Contents

1 Introduction 13

1.1 Overview 13
1.2 Position within the system 15
1.3 Safety 17

2 Installation 17

2.1 System Requirements 17
2.2 Download SAMLight 18
2.3 Installing SAMLight 18
2.4 Copy or Backup Settings 20

3 Hardware Settings 20

3.1 sc_setup.exe 20
3.2 Scanner Settings 22
3.3 Card Settings 24
3.3.1 Pin Assignment 24
3.3.2 USC-1 Settings 26
3.3.3 USC-2 Settings 28
3.3.3.1 Correction Settings 31
3.3.3.2 Analog In 32
3.3.4 USC-3 Settings 33
3.3.4.1 Correction Settings 36
3.3.4.2 Analog In 37
3.3.5 RTC-3 Settings 38
3.3.6 RTC-4 Settings 40
3.3.7 RTC-5 Settings 43
3.3.7.1 Auto Laser Control 46
3.3.8 RTC-6 Settings 47
3.3.8.1 Auto Laser Control 50
3.3.9 RTC Scanalone Settings 51

4 Motion Control Settings 53

4.1 Jog Dialog 55

4.2 Home / Shift Dialog ... 55
4.3 Step & direction motion controller ... 56
4.3.1 Type 14 - USC-2 stepper controller ... 57
4.3.1.1 Motion Settings Dialog Type 14 ... 65
4.3.2 Type 8 - Generic stepper controller ... 66
4.3.2.1 Motion Settings Dialog Type 8 ... 73
4.3.3 Stepper I/O parameters ... 74
4.4 Other motion controller ... 77
4.4.1 Type 1 - IMS Stepper Drives ... 77
4.4.2 Type 4 - External custom controller ... 78
4.4.3 Type 5 - IMS MDrive ... 79
4.4.4 Type 6 - Faulhaber motion controller ... 87
4.4.5 Type 7 - isel IT Stepper Controller / DNC ... 88
4.4.6 Type 9 - Generic RS232 interface ... 93
4.4.7 Type 10 - SHS 2000 Star ... 94
4.4.8 Type 11 - Jena Ecostep100 ... 97
4.4.9 Type 12 - IO Switcher ... 101
4.4.10 Type 13 - Isel CanApi Controller ... 102
5 Global Settings ... 103
5.1 Reset License ... 103
5.2 View ... 104
5.3 Optic ... 106
5.3.1 USC Cards ... 106
5.3.1.1 Edit Lens Init Job Dialog ... 109
5.3.2 RTC Cards ... 110
5.3.3 Min/Max ... 112
5.4 Laser ... 113
5.5 Shortkeys ... 115
5.6 General ... 116
5.6.1 Shift Map ... 119
5.6.2 Months Map ... 120
5.6.3 Day Map ... 121
5.6.4 Year Map ... 122
5.6.5 Gloabal Sequences Reset Times Dialog ... 122
5.6.6 Job Save/Load Dialog ... 123

5.7 Remote ... 124
5.8 IO ... 127
5.8.1 Message Input Combination ... 130
5.9 Extras ... 132
5.10 3D ... 134
5.11 User Level ... 135
5.11.1 Access Rights ... 136
5.12 Card ... 137
5.13 Trigger ... 137
5.14 Report ... 138
6 Pen Settings ... 139
6.1 Edit Pens ... 142
6.1.1 Main Settings for Pens ... 144
6.1.2 Scanner Settings for Pens ... 146
6.1.2.1 Wobble Settings ... 148
6.1.3 Miscellaneous Settings for Pens ... 151
6.1.3.1 Perforation Dialog ... 154
6.1.4 Drill Settings for Pens ... 155
6.1.5 Ramping Settings for Pens ... 156
6.1.6 Bitmap Settings for Pens ... 158
6.1.7 Pen Paths ... 160
6.2 Pen Advanced ... 161
6.2.1 Power Map ... 161
6.2.2 System PixelMap ... 164
7 User Interface ... 166
7.1 Menu Bar ... 166
7.1.1 File ... 167
7.1.1.1 Job Format ... 168
7.1.1.2 Job Properties ... 170
7.1.2 Edit ... 171
7.1.2.1 Spacing Advanced ... 172
7.1.2.2 ArrayCopy ... 173
7.1.2.3 ArrayPolarCopy ... 174
7.1.3 Extras ... 174

7.1.3.1 Teach / Relocate Reference ... 174
7.1.3.2 Step / Repeat ... 176
7.1.3.3 Bitmap Marking ... 178
7.1.4 User ... 180
7.1.5 Window ... 180
7.1.6 Help ... 180
7.2 Toolbars ... 180
7.2.1 File Toolbar ... 181
7.2.2 Camera Toolbar ... 181
7.2.3 View Level Toolbar ... 182
7.2.4 Geometry Object Toolbar ... 183
7.2.5 Functionality Object Toolbar ... 184
7.2.5.1 Data Wizard ... 185
7.2.5.2 ParameterFinder ... 189
7.2.6 Alignment and Spacing Toolbar ... 194
7.2.7 Extras Toolbar ... 194
7.2.8 Stepper Position Toolbar ... 195
7.2.9 3D Surfaces Toolbar ... 195
7.2.10 Special Sequences Toolbar ... 195
7.2.11 Analog In Toolbar ... 199
7.3 Entity List ... 200
7.3.1 Entity List ... 200
7.3.1.1 Index Entities with pro-/postfix Dialogs ... 204
7.3.2 Point Editor ... 205
7.4 View 2D ... 207
7.4.1 Operations ... 207
7.4.1.1 View Properties ... 209
7.4.1.1.1 General ... 210
7.4.1.1.2 Colors ... 210
7.4.1.1.3 MultiHead ... 212
7.4.2 Print Preview ... 213
7.5 Entity Property Sheet ... 214
7.6 Status Bar ... 215
8 Entities (Objects) ... 216
8.1 Entity Hierarchy ... 216

8.2 Geometry Objects 217
8.3 Barcode 219
8.3.1 Barcode Format 220
8.3.1.1 Code-39 Ex 221
8.3.1.2 GS1 Barcodes 221
8.3.1.3 QR Code 222
8.3.1.4 QR Code EX 222
8.3.1.5 Code 128 224
8.3.2 Scaling 224
8.3.3 Barcode Extended 224
8.3.4 Barcode Reader 228
8.3.5 Size Limits 229
8.4 Bitmap 229
8.4.1 Bitmap Extended 233
8.4.2 Marking Bidirectional 234
8.4.3 Black and White 236
8.4.4 Grayscale 237
8.5 Serial Number 237
8.5.1 Serial Number Formats 239
8.5.2 Serial Number as Barcode 239
8.5.3 Serial Number Advanced 241
8.5.4 Automate Serialization 242
8.5.4.1 ASCII File 242
8.5.4.2 Excel Table 243
8.5.4.3 Example 243
8.6 Date Time 245
8.6.1 Date Time Advanced 245
8.7 Text2D 248
8.7.1 Text2D Properties 249
8.8 Control objects 250
8.8.1 I/O Control Objects 251
8.8.2 Analog Output Control Object 252
8.8.3 Executable Control Object 253
8.8.4 Motion Control Object 254
8.8.5 Trigger (USC and RTC5 only) 256

8.8.6 AutoCalib (RTC only) ... 257
8.8.7 SetOverride Control Object ... 258
8.8.8 ScJump Control Object ... 259
9 Entities Properties ... 259
9.1 Transformations ... 259
9.1.1 2D Transformations ... 259
9.1.2 3D Transformations ... 261
9.2 Hatch ... 262
9.2.1 Hatch Style ... 265
9.3 Entity Info ... 266
9.4 Element Info ... 268
9.5 Styles ... 268
9.5.1 Edit Styles ... 271
10 Import-Export ... 271
10.1 Import ... 271
10.1.1 Point Cloud Files ... 275
10.1.2 Import Advanced ... 275
10.1.2.1 Advanced for several formats ... 277
10.1.2.2 Advanced for AI ... 277
10.1.2.3 Advanced for CNC ... 278
10.1.2.4 Advanced for DXF Version1 ... 279
10.1.2.5 Advanced for MCL ... 279
10.1.2.6 Advanced for PLT ... 280
10.1.3 Vector File Formats ... 280
10.2 Export ... 281
11 Mark ... 282
11.1 Mark Dialog ... 284
11.1.1 Redpointer ... 285
11.2 Trigger Dialog ... 286
11.3 Mark Preview ... 287
11.3.1 Preview Window ... 287
11.3.1.1 Command View ... 289
11.3.1.2 Line Info View ... 289
11.3.1.3 OpticModuleProperties ... 290

11.4 SAMLight Job IO Selection ... 292
12 Splitting ... 293
12.1 Angular Splitting ... 296
12.2 1D Planar Splitting ... 298
12.3 2D Planar Splitting ... 300
12.4 1D Mark on the Fly (USC and RTC-5 only) ... 303
12.5 Ring Splitting ... 305
13 Option MOTF ... 308
13.1 Encoder Signals ... 309
13.2 MOTF Multiplier ... 310
13.3 Simulation Mode ... 311
13.4 Hardware setup ... 311
13.4.1 Card Specific: USC-1 ... 311
13.4.2 Card Specific: USC-2/3 ... 313
13.4.3 Card Specific: RTC cards ... 316
13.5 Calibration ... 319
13.5.1 Optic Calibration - static setup ... 319
13.5.2 USC-1 Specific Calibration MOTF ... 320
13.5.3 USC-2/3 Specific Calibration MOTF ... 322
13.5.4 Tips to optimize MOTF performance ... 325
13.6 Endless MOTF ... 325
13.7 Examples ... 329
13.7.1 Trigger based offset ... 329
13.7.2 Assembly Line ... 330
13.7.3 Rotational MOTF (RMOTF) ... 332
13.7.4 Rotated scan head ... 333
14 Option Flash (USC-2/3 only) ... 334
14.1 Supported Objects ... 336
14.2 Flash Jobs and Settings ... 337
14.3 Job processing ... 338
14.3.1 Up/Download ... 340
14.3.2 Execution ... 341
14.3.3 Flash Job IO Selection ... 342
14.4 System ... 342

14.5 MultiCard ... 344
15 Multiple Heads ... 346
15.1 Option MultiHead ... 346
15.1.1 Installation ... 347
15.1.1.1 Password ... 347
15.1.1.2 Setup Tool ... 347
15.1.1.3 Optic Settings ... 349
15.1.1.4 View2D ... 350
15.2 Option Head2 ... 351
15.2.1 Installation ... 352
15.2.2 Fixed Job Offset ... 352
15.2.3 Variable Entity Offset ... 354
15.3 MultiCard (USC-2/3 only) ... 355
16 Option FlatLense (USC only) ... 355
17 Option Optic3D ... 355
17.1 Features ... 355
17.1.1 3D Surfaces ... 356
17.1.1.1 Cylinder ... 356
17.1.1.2 STL Projection ... 358
17.1.1.3 Tilted Surface ... 361
17.1.1.4 Sphere ... 363
17.1.2 Marking on curved parts ... 365
17.1.3 Deep Engraving ... 369
17.2 Requirements & Settings ... 369
17.2.1 SCAPS USC cards ... 369
17.2.2 SCANLAB RTC cards ... 372
18 Option SAM3D ... 374
18.1 Main Window ... 375
18.1.1 Import Folder ... 377
18.2 Job Processing ... 377
18.2.1 Toolbar ... 378
18.2.2 Mouse Mode ... 379
18.2.3 3D View Properties ... 380
18.2.4 Slicing ... 380

18.2.5 Hatching ... 382
18.2.6 Marking ... 383
18.2.7 Special Sequences ... 383
18.2.8 Styles for Layers ... 386
18.2.8.1 Beam Compensation ... 387
18.2.8.2 Handling Up and Downskin ... 388
18.2.8.3 Using Num Loops ... 389
18.3 Client Control ... 390
19 Client Control Interface ... 397
19.1 Principle of operation ... 397
19.2 Implementations ... 398
19.3 Command Set ... 400
19.3.1 Application ... 401
19.3.2 Remote ... 403
19.3.3 System Settings ... 404
19.3.4 Mark Settings ... 406
19.3.5 Entity Settings ... 409
19.3.6 Pen Settings ... 413
19.3.7 Job Commands ... 414
19.3.8 General Commands ... 419
19.3.9 Async Mode ... 422
19.4 Constants ... 423
19.4.1 Long Value Types ... 423
19.4.2 Double Value Types ... 432
19.4.3 String Value Types ... 440
19.4.4 Long Data Ids ... 443
19.4.5 Double Data Ids ... 453
19.4.6 String Data Ids ... 457
19.4.7 Long Cmd Ids ... 460
19.5 Examples ... 464
19.5.1 Rotate output matrix ... 464
19.5.2 Fit to entity ... 465
19.5.3 Set array ... 465
19.5.4 Get and set outputs and inputs ... 466
19.5.5 Get and set text properties ... 466

19.5.6 Retrieve entities ... 467
19.5.7 Motion control ... 468
19.5.8 Precalculating the mark time ... 470
19.5.9 Create a beam compensated copy of an entity ... 471
19.5.10 Create a SAMLight View2D screenshot ... 471
19.5.11 Save splitted job file as tiles ... 472
19.5.12 Deactivating visual updates ... 472
19.5.13 Mirror entity on Y axis ... 474
19.5.14 Correct UCF ... 475
19.5.15 Camera image as background in View 2D ... 475
19.6 Optimize Performance ... 475
19.6.1 Example 1 ... 478
19.6.2 Example 2 ... 479
20 How to ... 481
20.1 Use Simple Fonts ... 481
20.1.1 Simple Fonts Format ... 481
20.1.2 Generate Fonts ... 481
20.1.2.1 Scaps Font Format ... 482
20.1.2.2 Scaps Converter ... 483
20.2 Command Line Parameters ... 490
20.3 Customize Program / Language ... 491
20.3.1 Personalize Program ... 491
20.3.1.1 Installation of User Data ... 492
20.3.1.2 Customize Laser Names ... 493
20.3.2 Customize Language ... 493
20.3.2.1 Global Settings ... 493
20.3.2.2 Resource Editor ... 494
20.3.2.2.1 String Editor ... 495
20.4 Use camera as background image ... 496
20.5 Accelerate SAMLight ... 497
20.6 Generate Dots ... 499
21 Backgrounds ... 499
21.1 Scanner and Laser delays ... 499
21.2 USC Position Transformation ... 503

21.3 Pixelmode 505
21.3.1 Pulse Modulation 506
21.3.2 Generating a scanner bitmap 507
Index 510

1 Introduction

1.1 Overview

Welcome to the SAMLight scanner application. This documentation describes the standard scanner application SAMLight. SAMLight is an application to control scanheads and lasers in order to do marking on different materials, to do 3D marking, welding, cutting and many more. The user interacts using the graphical user interface including dialogs and editors. The options Optic3D, SAM3D, Marking On The Fly, Flash, Client Control Interface, and Multihead marking are also explained. In more detail:

1. Standard SAMLight functionality:

Standard features are marking of objects and entities either created in the SAMLight editor or loaded / imported using the Load or Import functionality. Standard objects and entities are:

- Geometries like rectangle, ellipse, lines, points
- Barcodes
- Bitmaps
- Serial Numbers
- Date Time Objects
- Motion Controllers
- Control objects which are:
  - Set Output Bits (digital signals)
  - DAC output (analog signals)
  - Timer
  - Wait for Input
  - Override Power / Frequency / Speed
  - Start an executable
  - Motion Control Objects
  - MOTF Control Objects
- Import / Export of various file formats (Adobe Illustrator, DXF, etc...)
- Mark Preview Window
- Assign parameters like Speed, Frequency, Power, Laser Delays to Pens for marking

Additionally there are several marking modes, as:

- Using external Trigger to start marking
- Teach / Relocate Reference Points
- Step / Repeat
- Bitmap Marking
- Marking On The Fly
- Splitting modes: 1D, 1D MOTF, 2D, Angular, Bitmap Splitting

For Real Time entity manipulation there is a tool called Data Wizard. This tool can prepare entities to be grouped for hatching or to sort the marking order if these entities were imported using the Import functionality. The Data Wizard can hugely increase marking performance.

- Create / Edit language resource files to adapt the SAMLight GUI to your language
- Create custom fonts with the laser font converter

2. Optic3D Marking Functionality:

Load and generate 3D data. The marking focus will be shifted in Z-direction by an additional focal lens. This can be used for Deep Engraving and marking 3D surfaces. More details will be given in the chapter 'Option Optic3D'.

3. SAM3D Marking - Rapid Prototyping / Stereolithography:

Import / Export 3D data called STL or CLI files. The 3D will be done by slicing the source file then moving the marking target object in Z-direction with a motion device between marking the slices. This means the object is marked layer for layer. Possible features are Up- and Downskin Marking and Beam Compensation. More details will be given in the chapter 'Option SAM3D'.

4. Marking On The Fly:

This functionality will mark data on a target object that is constantly moving in X or Y direction (in some cases it is also possible to deflect in X and Y direction simultaneously). More details will be given in the chapter 'Option MOTF'.

5. Flash:

USC-2 cards can be run in Standalone Mode. Therefore they have to be connected via Ethernet Telnet Interface or RS232 plain ASCII terminal. This will allow to give commands to the USC-2 like Start Mark, Get Time, ... without an instance of SAMLight is running. It is also possible to access more than one scanner card from the PC. This is called MultiCard. More details will be given in the chapter 'Option Flash'.

6. Remote Control - Client Control Interface:

The Client Control Interface is based on ActiveX / COM calls sent to SAMLight from an external program on the same PC or even from an application on a different PC connected via Ethernet. The Client Control Interface supplies a huge amount of functions used to load, save, edit and mark the jobs in SAMLight and to work with all kinds of entities inside that job. More details will be given in the chapter 'Client Control Interface'.

7. Multiple Scanner Controller Cards:

It is possible to control more than one scanner card at the same time. Differentiate between MultiHead and Head2. In MultiHead up to six different Scanner Controller Cards can be controlled by one instance of SAMLight. Each scanhead having its own working field in the View2D and its own laser source. Head2 is used for USC-2, RTC4 and RTC5 cards. Then a second scanhead can be connected to the same card and will mark the same content as the primary head.

1.2 Position within the system

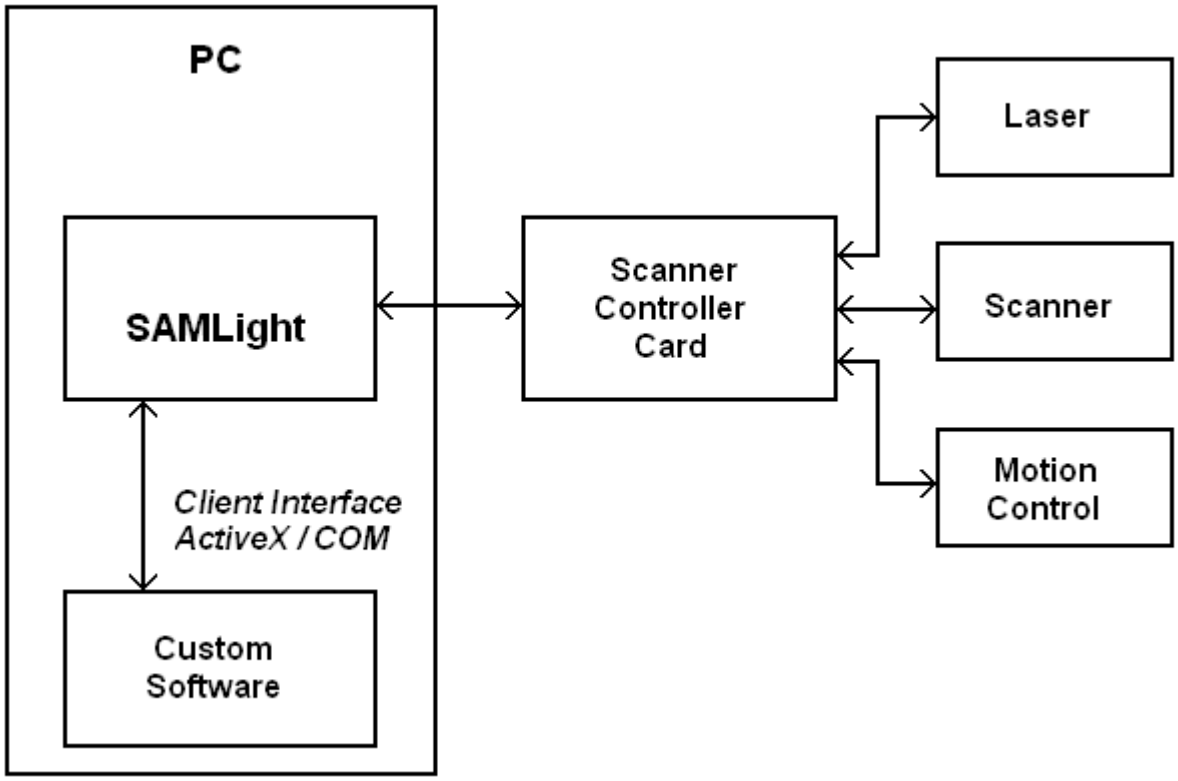


Figure 1: Position within the system

Possible configurations for using SAMLight are:

1. One USC or RTC card with SAMLight:

A single scanner controller card can be included in the system. For SAMLight standard mode simply connect the USC card via USB cable or if using a USC-2 it is also possible to connect via Ethernet. The RTC card has to be fitted in one of the PCI slots of the PC. Configuration of the hardware is explained in chapter "Hardware Settings".

2. One USC-2 card using Flash mode:

Flash mode is only available for USC-2 cards. For the Flash mode the USC-2 doesn't have to be connected to the PC via USB cable but it is necessary to connect it via an RS232 terminal to send ASCII commands to the card or connect via a Telnet Client using an Ethernet connection. However it is not necessary to run SAMLight.

3. Remote control of SAMLight:

It is possible to communicate with SAMLight via the Client Control Interface. The Client Control application can be run inside the same PC or it can be run on a remote PC and then communicate via TCP or plain ASCII commands. The Client Control commands are explained in the chapter Client Control Interface.

4. Using more than one Scanner Controller Card:

It is possible to access more than one scanner controller card at the same time. This feature is called Multi-Head. One can adress up to 6 scanner controller cards with SAMLight. Then each of the cards control a unique laser. If using the Head2 feature one scanner controller card can control 2 scanheads using one single laser. The marking of the secondary scanhead is the same as for the primary scanhead. This mode is also possible with the Flash. If there are more than one USC-2 card, each card can be addressed using for e.g. the visible USC-server. If using the visible USC server with more than one USC-2 card please be careful

when using the button InfoView, because it is not always clear which of the cards is used for this button.

5. Using JobIO select mode:

The JobIO select mode is a kind of half SAMLight plus a half of Remote control. The job that should be marked is issued via the Input pins of the card. These are the Opto_In bits of the USC cards or the DigiIO pins of the RTC card. This mode can be enabled when SAMLight is already running or it can be enabled while in Flash mode. Detailed description will be given in chapter Mark for SAMLight and chapter Option Flash for the Flash mode.

1.3 Safety

The goods delivered by SCAPS are designed to control a laser scanner system. Laser radiation may effect a person's health or may otherwise cause damage. Prior to installation and operation compliance with all relevant laser safety regulations has to be secured. The client shall solely be responsible to strictly comply with all applicable and relevant safety regulations regarding installation and operation of the system at any time.

The goods will be delivered without housing. The client shall be solely responsible to strictly comply with all relevant safety regulations for integration and operation of the goods delivered.

2 Installation

This chapter describes the system requirements to run SAMLight, where to download the SCAPS installer and how to install SAMLight.

2.1 System Requirements

Software requirements:

SAMLight will run under the following Microsoft Windows operating systems:

- 10 - 32/64 (complete support including SecureBoot starts with SAM version 3_6_5_20161230_004 or 3_6_0_20170116_0001)
- 8 - 32/64
- 7 - 32/64
- Embedded Standard 7 - 32
- Vista - 32/64
- XP - 32/64 (only up to SAM version 3_6_5_0124_20170907 or 3_6_0_0012_20170714, newer SAM versions are not supported)
- XP Embedded - 32 (only up to version 3_6_5_0124_20170907 or 3_6_0_0012_20170714, newer SAM versions are not supported)

In order to run SAMLight correctly it is necessary that all current Windows Updates have been installed. For example:

- *On Windows 7 x64, KB3033929 must be installed for it to recognize the Jungo driver signature.*
- *On Windows Vista, 7 and 8, KB3118401 must be installed, because of a bug in the universal C runtime.*

These updates can be installed via automatic Windows updates or manually from the provided links.

In order to avoid unwanted laser emissions or malfunctions during the operation procedure it is necessary to deactivate the Windows hibernation or any power saving mode.

The hardware requirements are not strictly defined. Depending on the application and to get a reasonable working speed you need much better components. Minimum requirements for SAMLight are very low:

- 500 MHz x86-CPU
- 256 MB RAM
- 512 MB HDD

For big job files the following is recommended:

- 8 GB RAM or more
- CPU with high clock rate, because SAMLight only runs on a single core
- 64bit Windows (allows SAMLight to use up to 4 GB RAM)
- SSD or fast HDD

Increased hardware requirements could be necessary for the following applications:

- Many Bitmaps → more RAM required
- Large Jobs with many vector data → more RAM required
- Using mode "Job IO Select" → more RAM required, because of buffering of jobs
- Using complex jobs or using often calculations on the vector data → more RAM required
- Using option SAM3D → more RAM required (for slices) → because of more calculation operations for slicing; also more hard disc memory is recommended to buffer the slice data which have to be stored during the use of SAMLight; faster graphic is required for View3D
- Using option Optic3D → no additional requirements needed
- Using option MultiHead controlling many heads (up to six) → more RAM and processor power is recommended

2.2 Download SAMLight

The latest installer of SAMLight can be downloaded at www.scaps.com or is available on CD.

In the following text, **<SCAPS>** is a placeholder of the software installation path. By default, the SCAPS software will be installed into the folder **C:\scaps\sam2d**.

2.3 Installing SAMLight

Installing SAMLight:

Run sc_sam_setup_v_3_4_X_2015MMDD.exe and follow the instructions.

Administrator rights on the PC are necessary for the software installation. If you want to change the scaps installation path by a new installation, please uninstall the old version before completely. If necessary, make a backup of your scaps system folder, job and correction files before. It is also recommended to make a backup of these files after successful hard- and software setup.

Files / settings created by the setup:

- The setup creates an environment variable SCAPS_SAM which is set to be the installation folder, by default it is C:\scaps\sam2d\.
- sc_*.dll and sc_*.ocx files in \WINDOWS\system32\ (for 32bit) or \Windows\SysWOW64\ (for 64bit).
- sam_light.exe in <SCAPS>\
- sc_light_settings.sam → contains all program settings for SAMLight. It is recommended to make a backup of this file after setup. The file can be found in the folder <SCAPS>\system\.

Configuring SAMLight:

If SAMLight has been installed sucessfully on the PC the next step is to activate the software by connecting the dongle and / or USC cards. Therefore plug in the dongle in a free USB slot of the PC and / or connect the USC cards via USB cable. If you use RTC cards you should put them in a PCI slot of the PC (if not already done).

After connecting the hardware to the PC the system may take some time to install the drivers automatically. If you are not sure that the drivers are installed correctly you can check this with the Windows Device

Manager. If using USC cards there should be the Jungo device present as well as an USC card entry. The USB dongle is activated if there is a CBUSB entry in the USB-Controller entry. If using a RTC card there should e an entry RTC... in the entry SCANLAB.

Now start the tool sc_setup.exe. You will be asked to enter the password for the connected dongle.

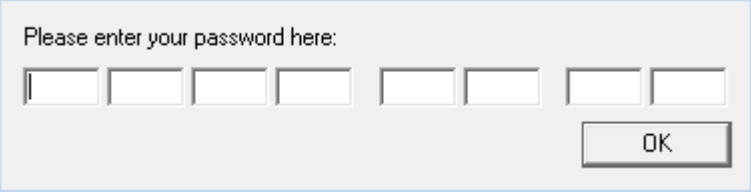


Figure 2: Password Dialog

You can also create a txt file sc_password.txt with the dongle specific password and store this within folder C:\scaps\sam2d\system. During first start, if the software does not find the sc_#_#.scl file, where #_# is the full dongle ID, it looks for sc_password.txt and automatically inputs the key stored here.

When using an old password, which consists only of 16 or 24 characters, please leave the text fields at the end empty.

There is a shortcut for entering the password: First select the password and press Ctrl-C (copy). Then click with the left mouse button on the first password entering field and press Ctrl-V (paste). Now press ENTER two times. The password should be entered correctly now.

In the menu Resource you can specify which language you want to use.

2.4 Copy or Backup Settings

If you want to **update the SAMLight version on your PC**, please first make a backup of the system (by copying the 8 files mentioned below to a place safe).

If you want to **build up a new system on another PC** with the same parameters as the old one - the SAMLight version may be different - , please first install the new SAMLight version and then copy the following 8 files from the old system folder to the new one:

- sc_light_settings.sam
- sc_motion_settings.txt
- sc_motion_<MOTION TYPE>_settings.txt (depends on the motion type defined in sc_motion_settings.txt, refer to Motion_Controller).
- sc_usc_card_ids.txt
- sc_usc.cfg
- hosts.allow
- hosts.deny
- sc_resource_sc_<LANGUAGE>.sam; Only required if the language is NOT english.
- sc_resource_settings.sam

It is also necessary to copy the correction file - for example "cor_neutral.ucf" - from the old system to the new one. The path of the correction file is defined in sc_light_settings.sam. Please refer, depending on the scanner controller card from chapter 3.3.2 to 3.3.8.

3 Hardware Settings

This chapter describes how to set up the different scanner controller cards.

3.1 sc_setup.exe

Now that SAMLight is configured the next step is to configure the scanner controller card(s). This is done by starting the program sc_setup.exe in <SCAPS>\tools. The window looks like:

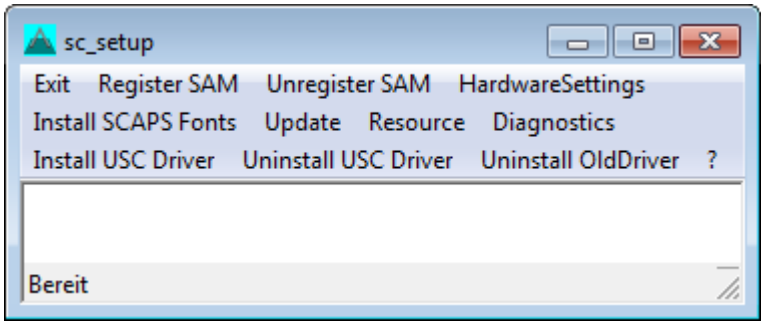


Figure 3: sc_setup.exe Dialog

Menu Item	Description
Exit	Closes the software.
Register SAM	Registers the SCAPS DLLs.
Unregister SAM	Unregister the SCAPS DLLs.
HardwareSettings	Opens the HardwareSettings dialog.
Install SCAPS Fonts	Installs the SCAPS laser fonts.
Update	Updates from a previous version.
Resource	Edits the language resources.

Diagnostics	Displays information about the dongle that is plugged in,software version and DLL versions.
Install USC Driver	Installs the USC driver software.
Uninstall USC Driver	Uninstalls the USC driver software.
Uninstall OldDriver	Uninstalls the old USC-1 driver software.
?	Opens the About dialog.

Table 1: sc_setup.exe Menu

By clicking on Diagnostics → Dongle you can check if a dongle is connected to the PC and if it is recognized by SAMLight.

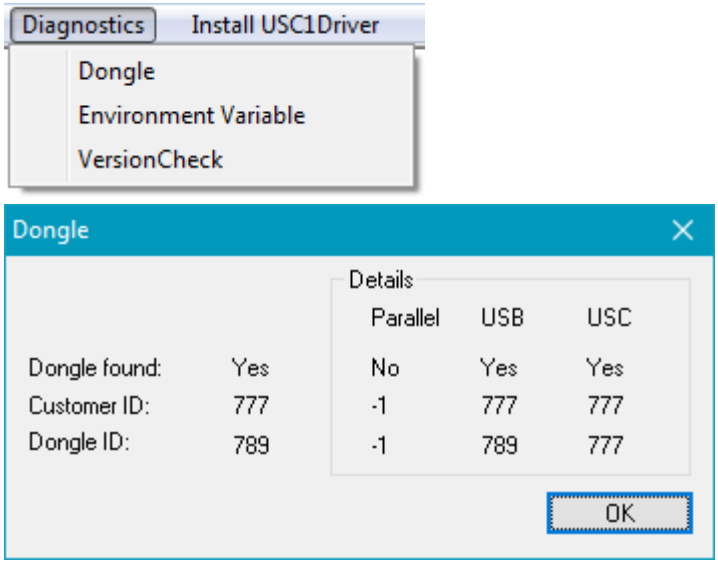


Figure 4: Diagnostics → Dongle window

To setup the scanner controller card(s) click on HardwareSettings. The following Dialog will appear:

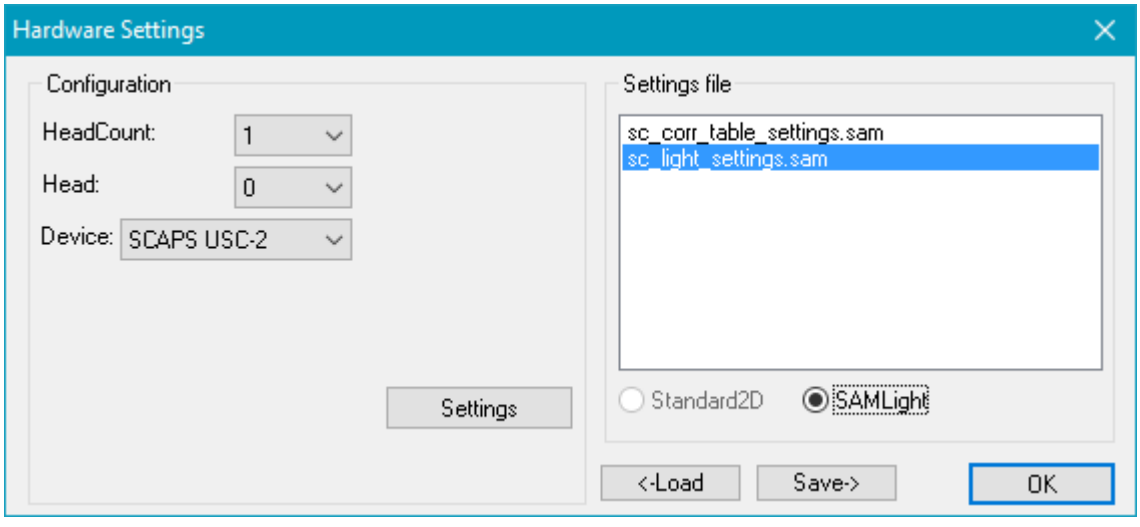


Figure 5: Hardware Settings Dialog

Choose the settings file 'sc_light_settings.sam' in the Settings File box and click on Load.

Head Count: Leave '1' if using only one scanner controller card. If you want to use more than one card click on the drop box button and choose the amount of scan heads you want to use. If using more than one Head (= Scanner controller card), select the Head number for each head and then assign the proper device using

the Device drop box.

3.2 Scanner Settings

To see this dialog go to <SCAPS>\tools\sc_setup.exe → HardwareSettings, "choose settings file" → Settings:

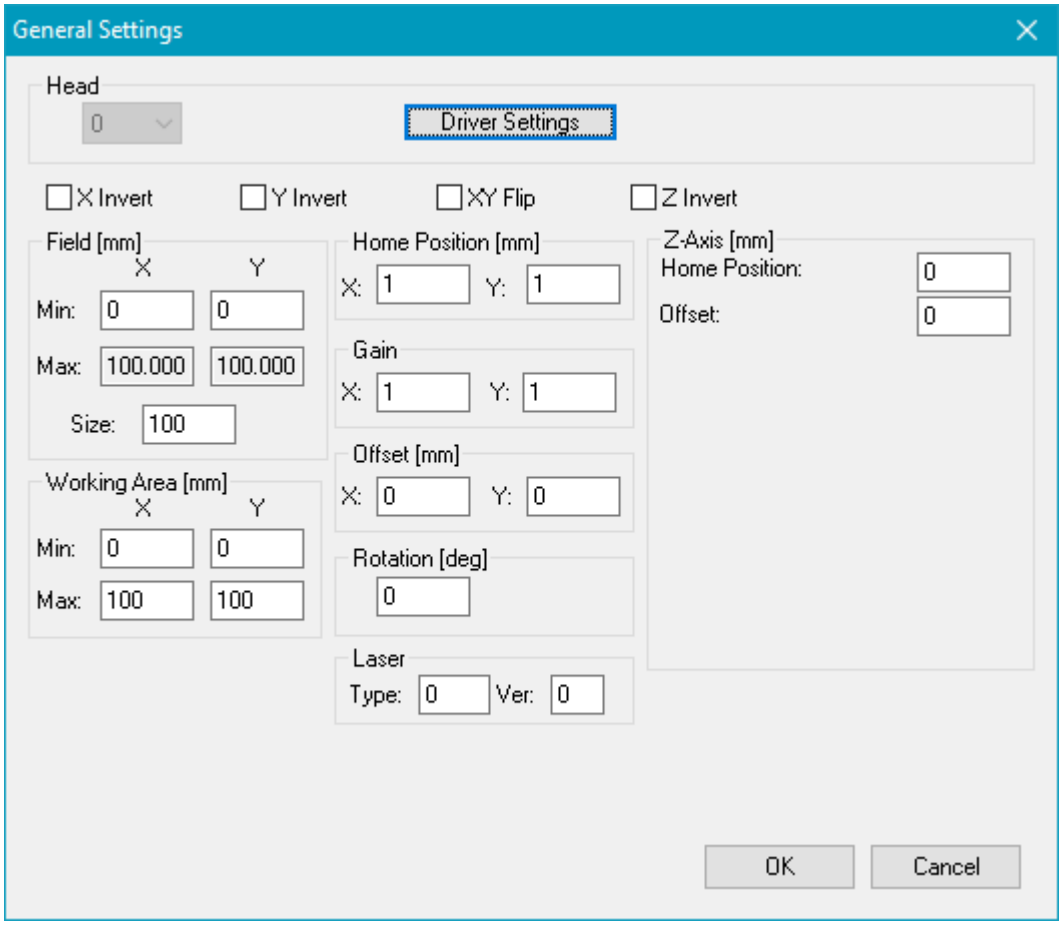


Figure 6: General Settings Dialog

Head: Choose the Head for which the driver settings should be edited. The drop down menu is only enabled if the *HeadCount* is greater than 1.

Driver Settings: Opens the Driver Settings dialog where you can adjust hardware specific settings.

Invert: Each scanner axis can be inverted separately.

Flip: The X and the Y axis can be flipped, so that the X axis gets the Y coordinates and vice versa.

Field [mm]: The size of the *field* has to be typed in in mm.

Min: This value can be negative, so that the field can be set up symmetrically to the origin.

Max: This value is computed as *Min* + *Size*.

Size: The edge length is *Size*. The field is always a square.

Working Area [mm]: The Working Area has a rectangular shape. The area is defined by *Min* and *Max*. The *Working Area* has to be within the *Field*. Consider rotation too.

Home Position [mm]: If *HomeJump* is enabled, the scanner goes back to this position after marking.

Gain: The gain values are there to slightly compensate X/Y gain errors to achieve a quadratic field.

Offset [mm]: With the *offset values* you can slightly compensate X/Y offset errors to achieve the theoretical midpoint of the scanner field. Global offset errors which have the same deviation in X and Y direction should be corrected by changing the *Field* parameters.

Rotation [deg]: The scanner output will be rotated counterclockwise by this angle.

Laser: Type in *Type* and *Ver* as advised by SCAPS.

Z-Axis [mm]: This option is only for Optic3D.

Z Home Position [mm]: If *HomeJump* is enabled, the scanner goes back to this position after marking.

Z Offset [mm]: Adds an offset to all Z values.
USC cards: XY coordinates are not compensated, like Pen Defocus.
RTC cards: XY coordinates are compensated, unlike Pen Defocus.

3.3 Card Settings

By clicking on Driver Settings you can configure the scanner controller card with more detail. The configuration depends on the Type of the used card: RTC or USC card. In the next chapters for each card the settings are described.

3.3.1 Pin Assignment

Port	USC-1 / USC-2	RTC3	RTC4
Analog A	37-pin connector DAC_A → pin 10	9-pin SUB-D laser connector (VB3) Analog OUT1 → pin 4, only available if jumper is set accordingly, see RTC3 installation manual	9-pin SUB-D laser connector (VB3) Analog OUT1 → pin 4, only available if jumper is set accordingly, see RTC4 installation manual
Analog B	37-pin connector DAC_B → pin 29	9-pin SUB-D laser connector (VB3) Analog OUT2 → pin 2, only available if jumper is set accordingly, see RTC3 installation manual	9-pin SUB-D laser connector (VB3) Analog OUT2 → pin 2, only available if jumper is set accordingly, see RTC4 installation manual
8 Bit Output	37-pin connector LP 0 → pin 19 LP 1 → pin 37 LP 2 → pin 18 LP 3 → pin 36 LP 4 → pin 17 LP 5 → pin 35 LP 6 → pin 16 LP 7 → pin 34	LASER extension connector L0 → pin 1 L1 → pin 3 L2 → pin 5 L3 → pin 7 L4 → pin 9 L5 → pin 11 L6 → pin 13 L7 → pin 15 / pin 17 only if jumper is set accordingly, see RTC3 installation manual	LASER extension connector L0 → pin 1 L1 → pin 3 L2 → pin 5 L3 → pin 7 L4 → pin 9 L5 → pin 11 L6 → pin 13 L7 → pin 15 / pin 17 only if jumper is set accordingly, see RTC4 installation manual
Q-Switch / Lasermodul	37-pin connector LaserA → pin 13	9-pin SUB-D laser connector (VB3) Laser1 → pin 1	9-pin SUB-D laser connector (VB3) Laser1 → pin 1
LaserGate	37-pin connector Laser_Gate → pin 31	9-pin SUB-D laser connector (VB3) LaserOn → pin 2	9-pin SUB-D laser connector (VB3) LaserOn → pin 2
IO	37-pin connector OPTO_IN 0 → pin 1 OPTO_IN 1 → pin 20 OPTO_IN 2 → pin 2 OPTO_IN 3 → pin 21 OPTO_IN 4 → pin 8 OPTO_IN 5 → pin 9 OPTO_OUT 0 → pin 3 OPTO_OUT 1 → pin 22 OPTO_OUT 2 → pin 4 OPTO_OUT 3 → pin 23 OPTO_OUT 4 → pin 27 OPTO_OUT 5 → pin 28	RTC3 I/O Extension Board, 68-pin SCSI Connector DIGITAL_IN0 ... DIGITAL_IN11, oDIGITAL_IN12+- ... oDIGITAL_IN15+- DIGITAL_OUT0 ... DIGITAL_OUT15 For proper I/O handling, use the SCAPS Marking Active signal instead of the RTC busyout signal.	40-pin extension DIGITAL IN0 ... DIGITAL IN15 DIGITAL OUT0 ... DIGITAL OUT15 For proper I/O handling, use the SCAPS Marking Active signal instead of the RTC busyout signal.
Ext. Trigger	37-pin connector	9-pin SUB-D laser connector (VB3)	9-pin SUB-D laser connector (VB3)
Start	OPTO_IN 0 → pin 1	/START → pin 8	/START → pin 8
Stop	OPTO_IN 1 → pin 20	/STOP → pin 9	/STOP → pin 9

Port	RTC5 / RTC6	SCANalone
Analog A	15-pin SUB-D laser connector Analog OUT1 → pin 8	9-pin SUB-D laser connector Analog OUT1 → pin 4, only available if jumper is set accordingly, see RTC SCANalone installation manual
Analog B	15-pin SUB-D laser connector Analog OUT2 → pin 15	9-pin SUB-D laser connector Analog OUT2 → pin 2, only available if jumper is set accordingly, see RTC SCANalone installation manual
8 Bit Output	EXTENSION 2 connector L0 → pin 1 L1 → pin 3 L2 → pin 5 L3 → pin 7 L4 → pin 9 L5 → pin 11 L6 → pin 13 L7 → pin 15 / pin 17 only if jumper is set accordingly, see RTC5 installation manual	LASER extension SUB-D connector L0 → pin 1 L1 → pin 2 L2 → pin 3 L3 → pin 4 L4 → pin 5 L5 → pin 6 L6 → pin 7 L7 → pin 8 / pin 9 only if jumper is set accordingly, see RTC SCANalone installation manual
Q-Switch / Lasermodul	15-pin SUB-D laser connector Laser1 → pin 1	
LaserGate	15-pin SUB-D laser connector LaserOn → pin 2	
IO	40-pin extension DIGITAL IN0 ... DIGITAL IN15 DIGITAL OUT0 ... DIGITAL OUT15 For proper I/O handling, use the SCAPS Marking Active signal instead of the RTC busyout signal.	37-pin extension1-SUB-D connector DIGITAL IN0 ... DIGITAL IN15 DIGITAL OUT0 ... DIGITAL OUT15 For proper I/O handling, use the SCAPS Marking Active signal instead of the RTC busyout signal.
Ext. Trigger	15-pin SUB-D laser connector	9-pin SUB-D laser connector
Start	/START → pin 3	/START → pin 8
Stop	/STOP → pin 11	/STOP → pin 9
Marking in progress		

3.3.2 USC-1 Settings

Card Settings for USC-1 card. This dialog can also be opened when SAMLight is running: *Menu bar → Settings → System → Optic → Advanced...:*

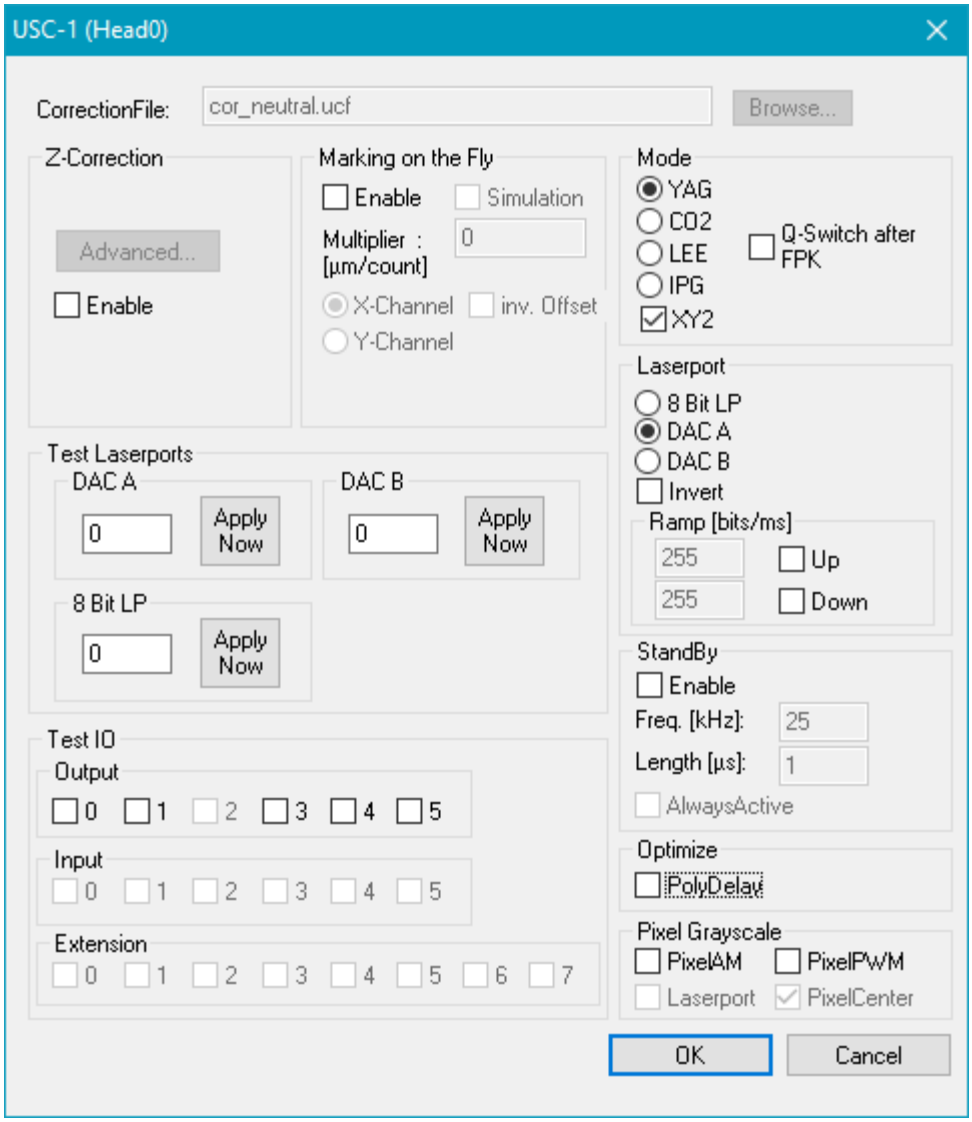


Figure 7: Card Settings for USC-1 Card

CorrectionFile: Click on the Browse button to select the correct correction file for this card.

MarkingOnTheFly: Requires the MarkingOnTheFly SAMLight option. MOTF related parameters are described in chapter Card Specific: USC-1.

Test Laserports:

DAC A/B: Here you can test the Digital Output A/B. This can be observed with a DB-37 Diagnostic Board attached to the USC-1 card. The minimum value for the digital output is 0, the maximum is 255.

8 Bit LP: Example: If 5 is entered and *ApplyNow* is pressed, the first and the third bit of the digital port are *high*.

Test IO: The IO-Port is a 6-bit port. For the USC-1 scanner card the first output bit is predefined for marking in progress. The first input bit is predefined for external start, the second for stop.

Mode: Here you can choose the type of the laser (YAG, CO2, LEE, IPG).

XY2: Defines that the scanner driver signals are standard XY2-100

ParaDel. [ms]: This field is only available for Lasermodes 588-1 and 588-2. The value causes the software to wait after a pulse width change had been sent to the interface. The unit of the entered value is [µs].

Make sure that the 'XY2' check box is checked if not mentioned differently.

Q-Switch after FPK: Sets Q-Switch after the first pulse killer.

Laserport: Defines the port that sends the power signal for the laser (8 Bit LP, DAC A, DAC B).

Invert: Here the laser power value can be inverted. Note: In case of 8 Bit LP this checkbox does not invert the bits.

Ramp: Here you can define a velocity to increase or decrease the power of the laser port.

StandBy:

Enable: Globally enables standby mode.

Freq.: Q-Switch frequency of the laser pulses.

Length: Q-Switch length in µs.

AlwaysActive: Standby mode is also used when the laser signal is on.

The settings are done after leaving the global dialog Settings. Standby Settings for pens overwrite these settings if enabled for a pen as soon as this pen is used.

Optimize:

PolyDelay: If this is selected the length of the polygon delay gets varied depending on the angle between two successive vectors.

Pixel Grayscale:

PixelAM: Enables Amplitude Modulation.

PixelPWM: Enables Pulse Width Modulation. For more details see chapter Pulse Modulation.

Laserport: If checked the selected Laserport gets used for the output of PixelAM, else port DAC B is taken.

3.3.3 USC-2 Settings

Card Settings for USC-2 card. This dialog can also be opened when SAMLight is running: *Menu bar → Settings → System → Optic → Advanced...:*

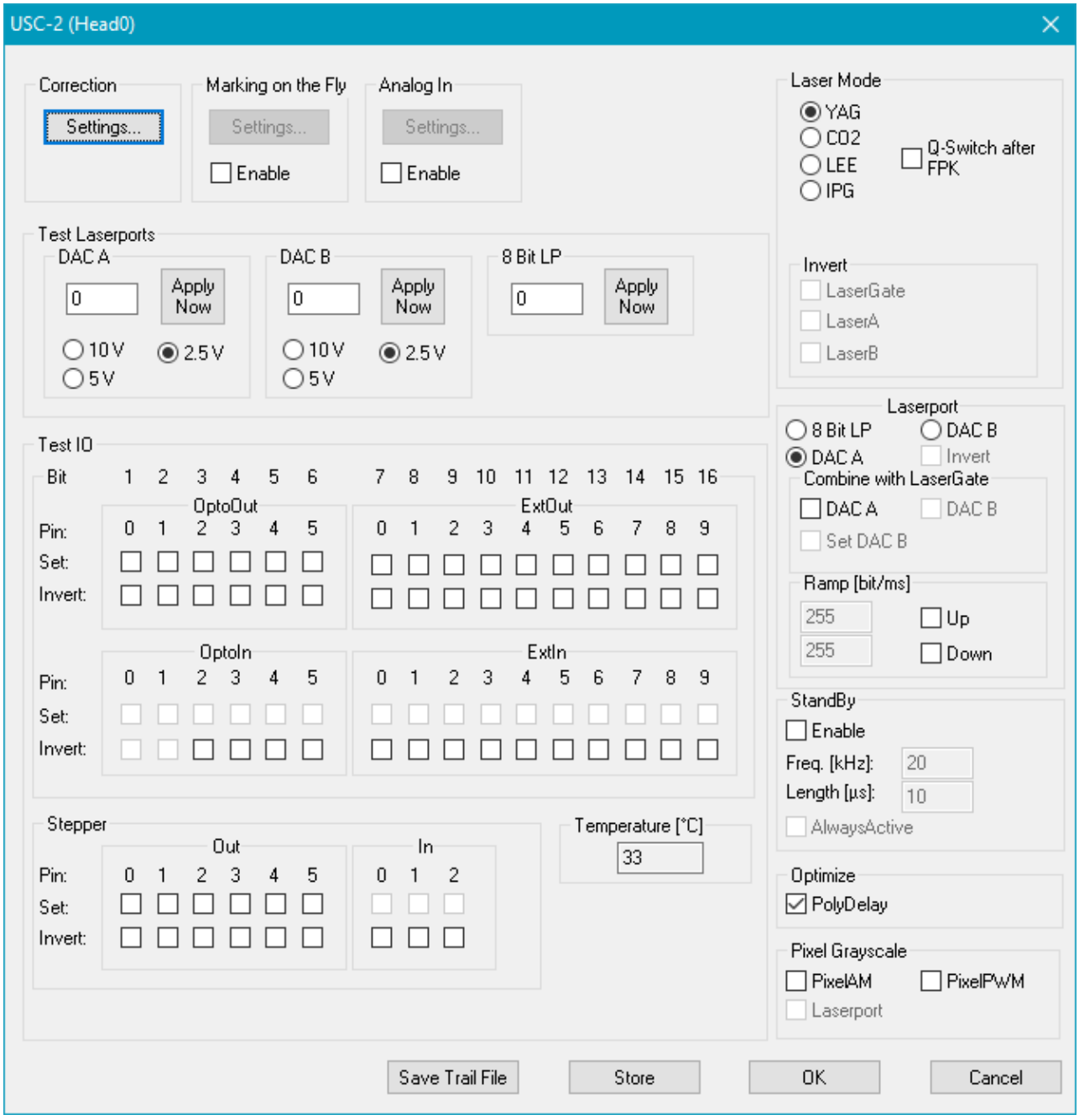


Figure 8: Card Settings for USC-2 Card

Correction:

Settings...: Click this button to open the Correction File Dialog for the USC-2 card. See chapter USC-2 correction settings.

Marking On The Fly (MOTF): Requires the MarkingOnTheFly SAMLight option.

Settings...: See chapter Card Specific: USC-2.

Enable: Check this to enable MOTF.

Analog In:

Settings...: There an Analog Input signal can be used that affects the scanner position. See chapter Analog In.

Enable: Check this to enable *Analog In*.

Test Laserports:

DAC A/B: Here you can test the Digital Output A/B. This can be observed with a DB-37 Diagnostic Board attached to the USC-2 card. The minimum value for the digital output is 0, the maximum is 4095.

8 Bit LP: Here you can test the 8 bit laserport by entering a value from 0 to 255. This can be observed with a DB-37 Diagnostic Board connected to the USC-2 card.

Test IO:

OptoOut/In: Here you can test if the bits of OptoOut and OptoIn can be set correctly.

ExtOut/In: Here you can test the additional IO bits. These IOs can be used for JobSelection Mode or to control external devices.

Stepper:

Out/In: If connected test the IOs of a stepper motor.

Mode: Here you can choose the type of the laser (YAG, CO2, LEE, IPG).

Q-Switch after FPK: Sets the Q-Switch after the first pulse killer.

ParaDel. [ms]: This field is only available for Lasermodes 588-1 and 588-2. The value causes the software to wait after a pulse width change had been sent to the interface. The unit of the entered value is [µs].

Invert: Laser Gate, Laser A, Laser B: Here the laser signals can be set to be *active low* or *active high*.

For security reasons the USC-2 Laser signals can only be changed with <SCAPS>\tools \sc_setup.exe → HardwareSettings, "choose settings file" → Settings → Driver Settings, not in SAMLight.

Laserport: Defines the port that sends the power signal for the laser (8 Bit LP, DAC A, DAC B).

Invert: Shows the status of the Laserport. Just for "LEE" Laser Mode the 8 Bit LP is inverted.

Combine with LaserGate: If DAC A or DAC B is chosen under Laserport it is possible to turn off the laser power signal with the LaserGate. If LaserGate is going down DAC A or DAC B will also go down to 0. If the checkbox *Set DAC B* is activated the DAC A will not go to 0 when LaserGate is going down but it will be set to the value that is defined under *Test Laser ports → DAC B*.

Ramp: Defines the velocity of the increasing or decreasing of the power of the laser port.

StandBy:

Enable: Globally enables standby mode.

Freq.: Q-Switch frequency of the laser pulses.

Length: Q-Switch length in µs.

AlwaysActive: Standby mode is also used when the laser signal is on.

Optimize:

PolyDelay: If this is selected the length of the polygon delay gets varied depending on the angle

between two successive vectors.

Pixel Grayscale:

PixelAM: Enables Amplitude Modulation.

PixelPWM: Enables Pulse Width Modulation. For more details see chapter Pulse Modulation.

Laserport: If checked the selected Laserport gets used for the output of PixelAM, else port DAC B is taken.

Save Trail File: Saves the USC-2 trail file to <*SCAPS*>*intermed*\. The file name corresponds to *sc_usc2_trail_head_x.txt*, where x stands for the head number.

Store: Stores the settings to the USC-2 EPCS (this is necessary for stand-alone operation). Make sure that the settings fit to the laser and other machinery. The settings will be loaded during powering on the card.

3.3.3.1 Correction Settings

By clicking on the Correction Button in *Settings → System → Optic → Advanced → Correction → Settings* the correction dialog of the USC-2 card is being opened:

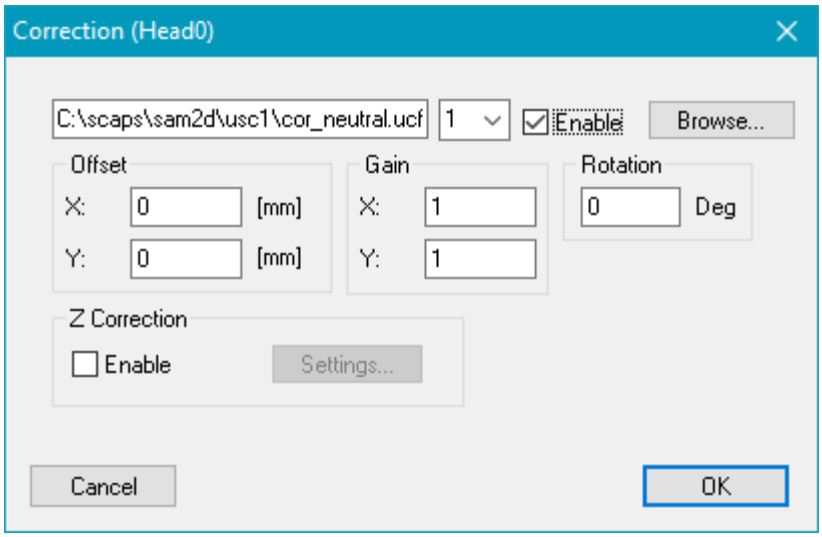


Figure 9: Correction Settings for USC-2 Card

Offset: Define an offset of the marking result that is calculated on the scanner card and thus a "Galvo Coordinates out of range" errror will not be shown if the translated entity extends to the outside of the working area.

Gain: Define a Gain for the marking result that is calculated on the scanner card and thus a "Galvo Coordinates out of range" errror will not be shown if the scaled entity extends to the outside of the working area.

Rotation: Enter a rotation angle here. Rotation center will be the middle point of the working area. Please remind that this rotation is calculated in the scanner card just one step before marking output and thus a "Galvo Coordinates out of range" errror will not be shown if the rotated entity extends to the outside of the working area.

Z-Correction: If enabled and clicking on Settings... the Z Correction Table Dialog will be opened.

The Offset, Gain and Rotation functions could be interesting if you have several Jobs on the Flash of the USC-2 card and then if you want to set an Offset, Gain or Rotation for all these jobs. Since the calculation is done on the card, the parameters would be applied to all jobs. If you would use the functions from the Optic dialog you would have to change the parameters for each job and manually and reload them into the Flash which would be much more effort.

3.3.3.2 Analog In

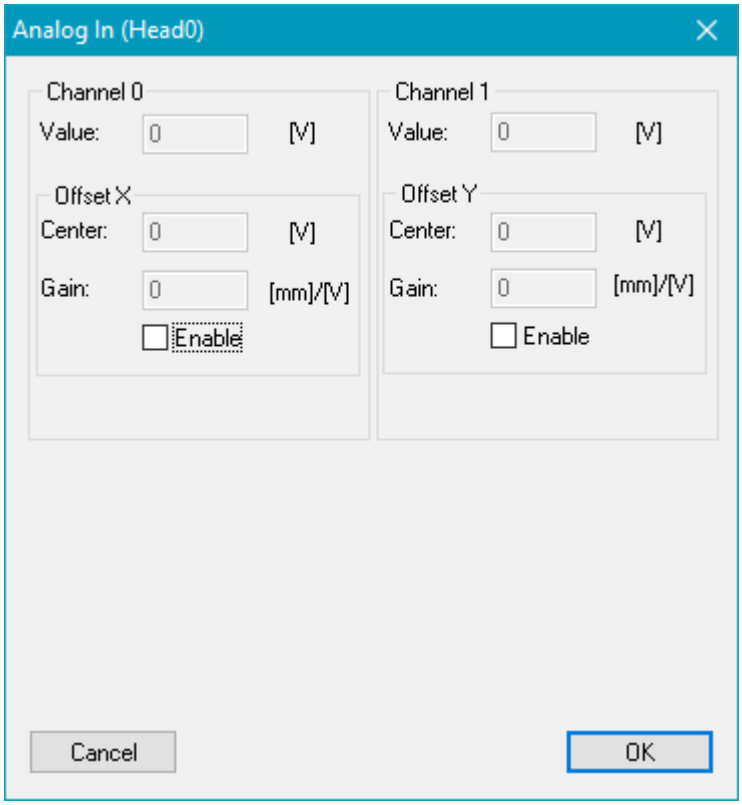


Figure 10: Analog-In Dialog

This dialog offers the possibility to set an offset of scanner position in x and y direction by putting a voltage on pin AnalogIn 0 (AI 0) and pin AnalogIn 1(AI 1).

Value: Value shows the voltage value been placed on pin Analog In.

Center: In this checkbox you can define a voltage value with which the Offset is 0 mm.

Gain: In this checkbox you define the width of offset per Volt.

Example: set Center = 5V, Gain = 2 mm/V for Offset X, when you place 6V on pin AI 0, the scanner gets an offset of 2mm in x direction.

3.3.4 USC-3 Settings

Card Settings for USC-3 card. This dialog can also be opened when SAMLight is running: *Menu bar → Settings → System → Optic → Advanced...:*

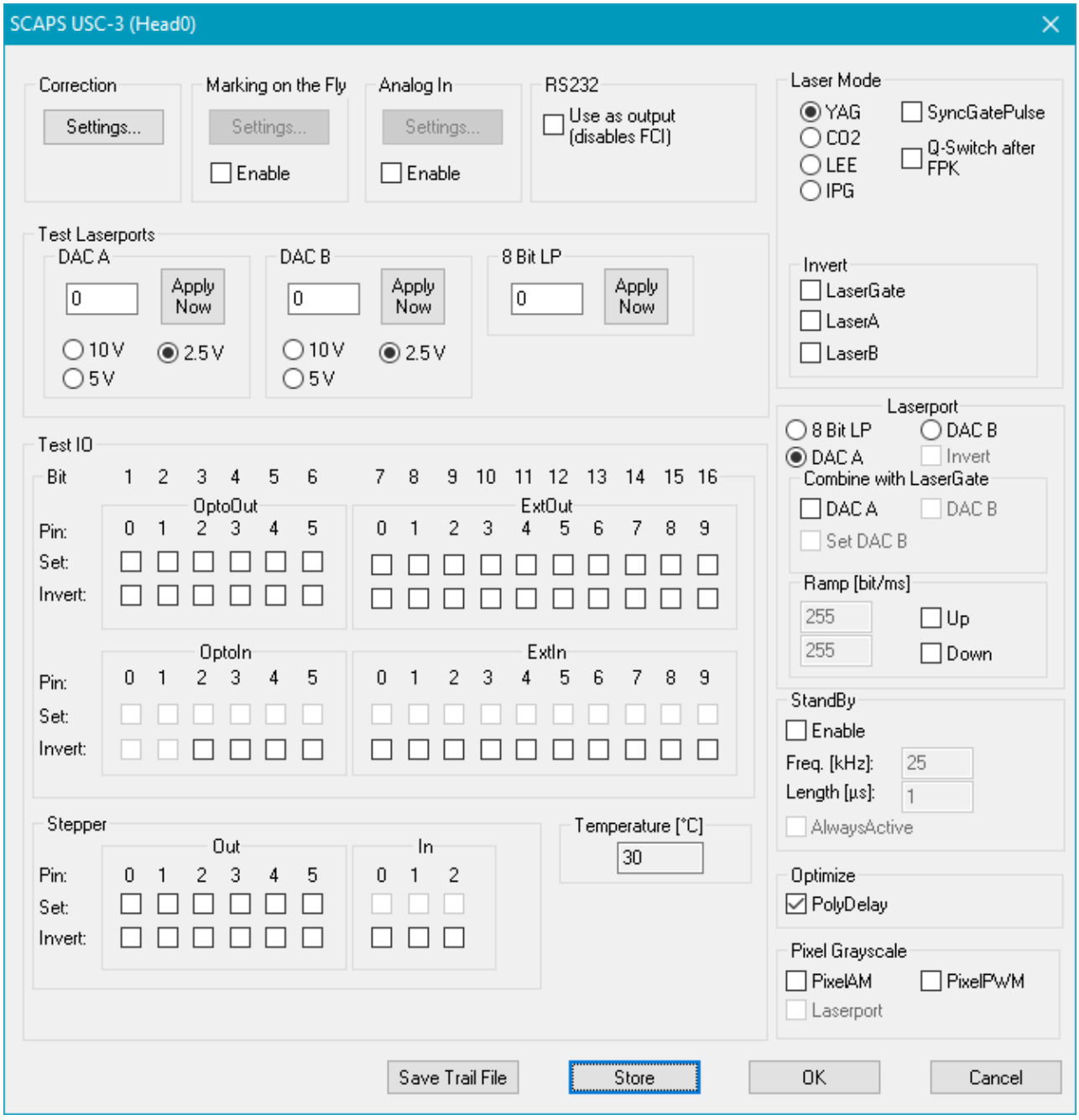


Figure 11: Card Settings for USC-3 Card

Correction:

Settings...: Click this button to open the Correction File Dialog for the USC-3 card. See chapter USC-3 correction settings.

Marking On The Fly (MOTF): Requires the MarkingOnTheFly SAMLight option.

Settings...: See chapter Card Specific: USC-3.

Enable: Check this to enable MOTF.

Analog In:

Settings...: There an Analog Input signal can be used that affects the scanner position. See chapter Analog In.

Enable: Check this to enable *Analog In*.

RS232:

Use as Output (disables FCI): Activates RS232 as Output for controlling laser and motion controller, therefore Flash Control Interface (FCI) to send Flash commands by RS232 is disabled.

Test Laserports:

DAC A/B: Here you can test the Digital Output A/B. This can be observed with a DB-37 Diagnostic Board attached to the USC-3 card. The minimum value for the digital output is 0, the maximum is 4095.

8 Bit LP: Here you can test the 8 bit laserport by entering a value from 0 to 255. This can be observed with a DB-37 Diagnostic Board connected to the USC-3 card.

Test IO:

OptoOut/In: Here you can test if the bits of OptoOut and OptoIn can be set correctly.

ExtOut/In: Here you can test the additional IO bits. These IOs can be used for JobSelection Mode or to control external devices.

Stepper:

Out/In: If connected test the IOs of a stepper motor.

Mode: Here you can choose the type of the laser (YAG, CO2, LEE, IPG).

Q-Switch after FPK: Sets the Q-Switch after the first pulse killer.

ParaDel. [ms]: This field is only available for Lasermodes 588-1 and 588-2. The value causes the software to wait after a pulse width change had been sent to the interface. The unit of the entered value is [µs].

Invert: Laser Gate, Laser A, Laser B: Here the laser signals can be set to be *active low* or *active high*.

For security reasons the USC-3 Laser signals can only be changed with <SCAPS>\tools \sc_setup.exe → HardwareSettings, "choose settings file" → Settings → Driver Settings, not in SAMLight.

Laserport: Defines the port that sends the power signal for the laser (8 Bit LP, DAC A, DAC B).

Invert: Shows the status of the Laserport. Just for "LEE" Laser Mode the 8 Bit LP is inverted.

Combine with LaserGate: If DAC A or DAC B is chosen under Laserport it is possible to turn off the laser power signal with the LaserGate. If LaserGate is going down DAC A or DAC B will also go down to 0. If the checkbox *Set DAC B* is activated the DAC A will not go to 0 when LaserGate is going down but it will be set to the value that is defined under *Test Laser ports → DAC B*.

Ramp: Defines the velocity of the increasing or decreasing of the power of the laser port.

StandBy:

Enable: Globally enables standby mode.

Freq.: Q-Switch frequency of the laser pulses.

Length: Q-Switch length in µs.

AlwaysActive: Standby mode is also used when the laser signal is on.

Optimize:

PolyDelay: If this is selected the length of the polygon delay gets varied depending on the angle between two successive vectors.

Pixel Grayscale:

PixelAM: Enables Amplitude Modulation.

PixelPWM: Enables Pulse Width Modulation. For more details see chapter Pulse Modulation.

Laserport: If checked the selected Laserport gets used for the output of PixelAM, else port DAC B is taken.

Save Trail File: Saves the USC-3 trail file to <*SCAPS*>*intermed*\. The file name corresponds to *sc_usc3_trail_head_x.txt*, where x stands for the head number.

Store: Stores the settings to the USC-3 EPCS (this is necessary for stand-alone operation). Make sure that the settings fit to the laser and other machinery. The settings will be loaded during powering on the card.

3.3.4.1 Correction Settings

By clicking on the Correction Button in *Settings → System → Optic → Advanced → Correction → Settings* the correction dialog of the USC-3 card is being opened:

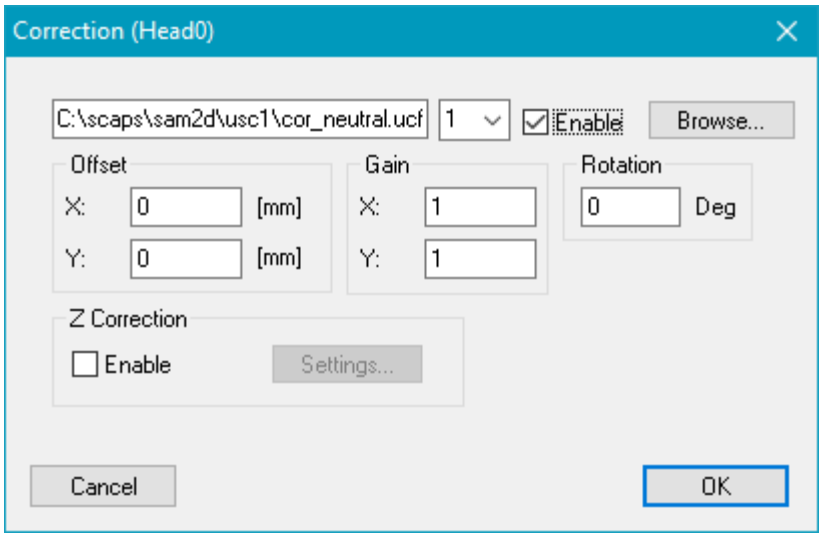


Figure 12: Correction Settings for USC-3 Card

Offset: Define an offset of the marking result that is calculated on the scanner card and thus a "Galvo Coordinates out of range" errror will not be shown if the translated entity extends to the outside of the working area.

Gain: Define a Gain for the marking result that is calculated on the scanner card and thus a "Galvo Coordinates out of range" errror will not be shown if the scaled entity extends to the outside of the working area.

Rotation: Enter a rotation angle here. Rotation center will be the middle point of the working area. Please remind that this rotation is calculated in the scanner card just one step before marking output and thus a "Galvo Coordinates out of range" errror will not be shown if the rotated entity extends to the outside of the working area.

Z-Correction: If enabled and clicking on Settings... the Z Correction Table Dialog will be opened.

NOTE

The Offset, Gain and Rotation functions could be interesting if you have several Jobs on the Flash of the USC-3 card and then if you want to set an Offset, Gain or Rotation for all these jobs. Since the calculation is done on the card, the parameters would be applied to all jobs. If you would use the functions from the Optic dialog you would have to change the parameters for each job and manually and reload them into the Flash which would be much more effort.

3.3.4.2 Analog In

By clicking on the Correction Button in *Settings → System → Optic → Advanced → Correction → Settings* the correction dialog of the USC-3 card is being opened:

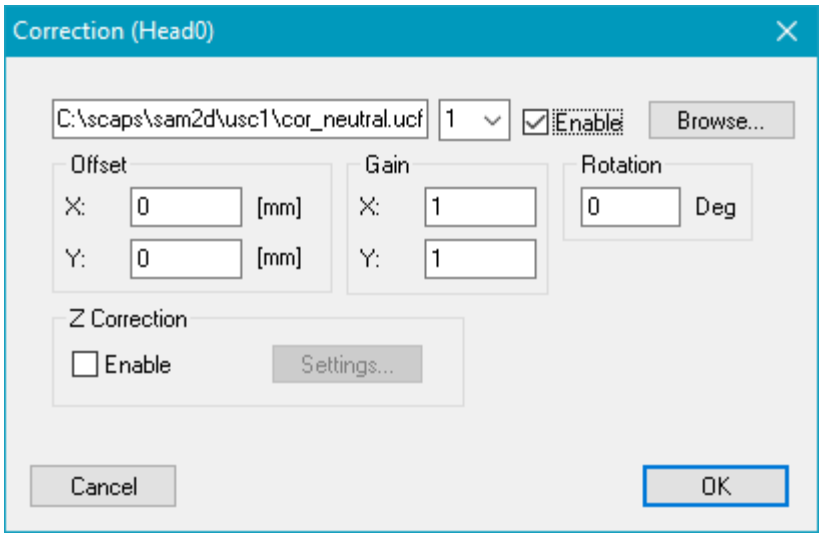


Figure 13: Correction Settings for USC-3 Card

Offset: Define an offset of the marking result that is calculated on the scanner card and thus a "Galvo Coordinates out of range" errror will not be shown if the translated entity extends to the outside of the working area.

Gain: Define a Gain for the marking result that is calculated on the scanner card and thus a "Galvo Coordinates out of range" errror will not be shown if the scaled entity extends to the outside of the working area.

Rotation: Enter a rotation angle here. Rotation center will be the middle point of the working area. Please remind that this rotation is calculated in the scanner card just one step before marking output and thus a "Galvo Coordinates out of range" errror will not be shown if the rotated entity extends to the outside of the working area.

Z-Correction: If enabled and clicking on Settings... the Z Correction Table Dialog will be opened.

3.3.5 RTC-3 Settings

Card Settings for RTC-3 card. This dialog can also be opened when SAMLight is running: *Menu bar → Settings → System → Card → Advanced... → Driver Settings*

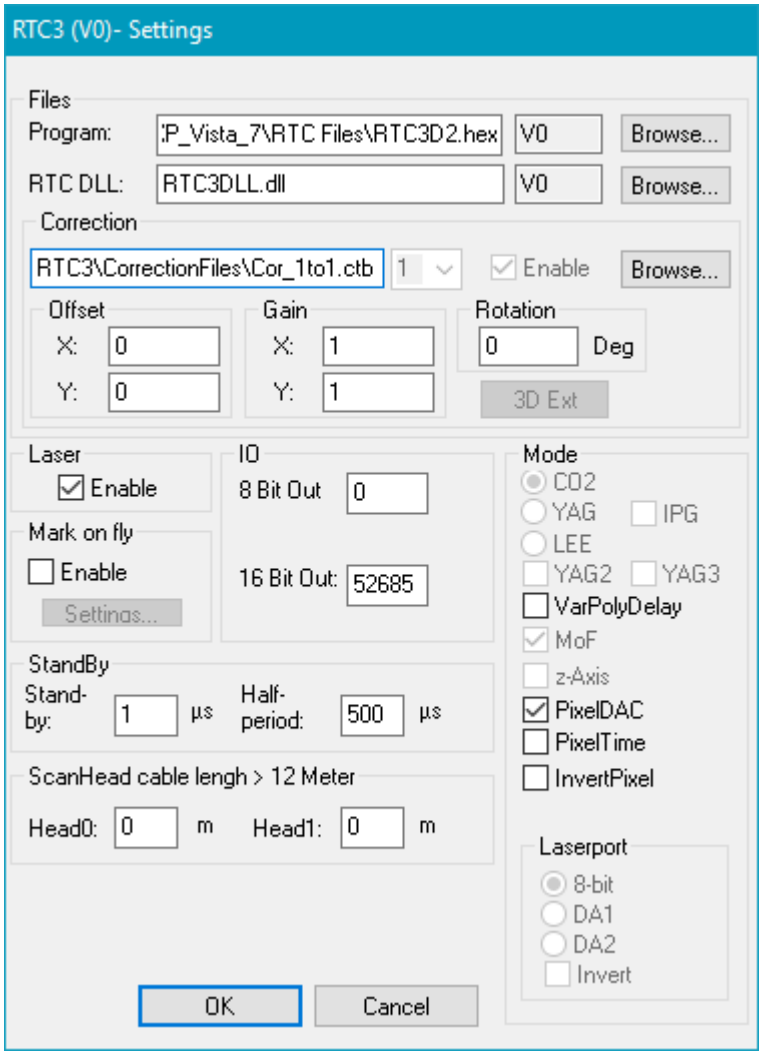


Figure 14: Card Settings for RTC3 Card

Files:

Program: Specifies the program file. This file is delivered together with the scanner card. The standard extension is *.hex. Click on the *Browse* button to make a search on the files location or type in the file name into the edit window.

RTC DLL: Specifies the *.dll file. This file is delivered together with the scanner controller card.

Using a 64-bit operating system you should nevertheless load a 32-bit RTC DLL, not the 64-bit one.

Correction: Specifies the location of the correction file. This file is delivered together with the scanner card. The standard extension is *.ctb. Click on *Browse* button to make a search on the files location or type in the file name into the edit window. It is possible to load two different correction files. This is mainly used in connection with the secondary head feature of the RTC3.

Offset, Gain, Rotation: Allow a global adjustment for each correction file. These features are mainly used to

adjust the fields of both heads when a secondary head is used. See also: Chapter Optic Settings Dialog.

3D Ext: Opens a dialog, where a Z-Table can be defined. For detailed information have a look at the RTC3 manual. See chapter Optic3D for RTC cards.

Laser: Globally enables or disables laser output.

Mark on Fly: Requires the MarkingOnTheFly RTC option. MOTF related parameters are described in chapter Card Specific: RTC cards.

StandBy: Globally enables standby mode.

Stand-by: Q-Switch length in µs for stand-by modus. If this is set to zero the stand-by mode is switched off.

Half-period: Half of the laser pulse period for stand-by modus.

Settings are done after leaving the global dialog Settings. Standby Settings for pens will overwrite these settings if enabled for a pen as soon as this pen is used.

ScanHead cable length > 12 Meter: Put in the cable length of each head if one cable is longer than 12m.

IO:

Lamp/8-bit: Sets the 8 bit or one of the digital outputs of the RTC Card during start up (as selected under Laserport). The 8 bit Output corresponds to the write_8bit_port command of the RTC.

16 Bit Out: Sets the 16 bit output of the RTC Card during start up.

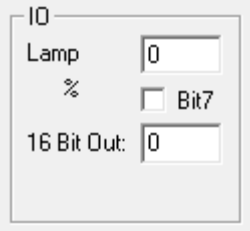


Figure 15: If *LEE Mode* is selected, the eighth bit is selectable separately

Mode: Here you can choose the type of the laser.

VarPolyDelay: If checked the length of the polygon delay gets varied depending on the angle between two successive vectors.

MoF: Shows whether the scanner card is able to do MarkingOnTheFly or not.

z-Axis: Indicates whether the card is able to do 3D Marking or not.

PixelDAC: Enables Amplitude Modulation.

PixelTime: Enables Pulse Width Modulation. For more details see chapter Pulse Modulation.

InvertPixel: Inverts bitmap pixels.

Pixel Mode 0: See chapter Pixelmode.

Laserport: Defines the port that sends the power signal for the laser if the laser is not a CO2 Laser. For a CO2 Laser the power signal is done by modulating the Laser A signal.

Invert: Here the laser power value can be inverted. Note: In case of 8-bit this checkbox does not invert the bits.

3.3.6 RTC-4 Settings

Card Settings for RTC-4 card. This dialog can also be opened when SAMLight is running: *Menu bar → Settings → System → Card → Advanced...→ Driver Settings*

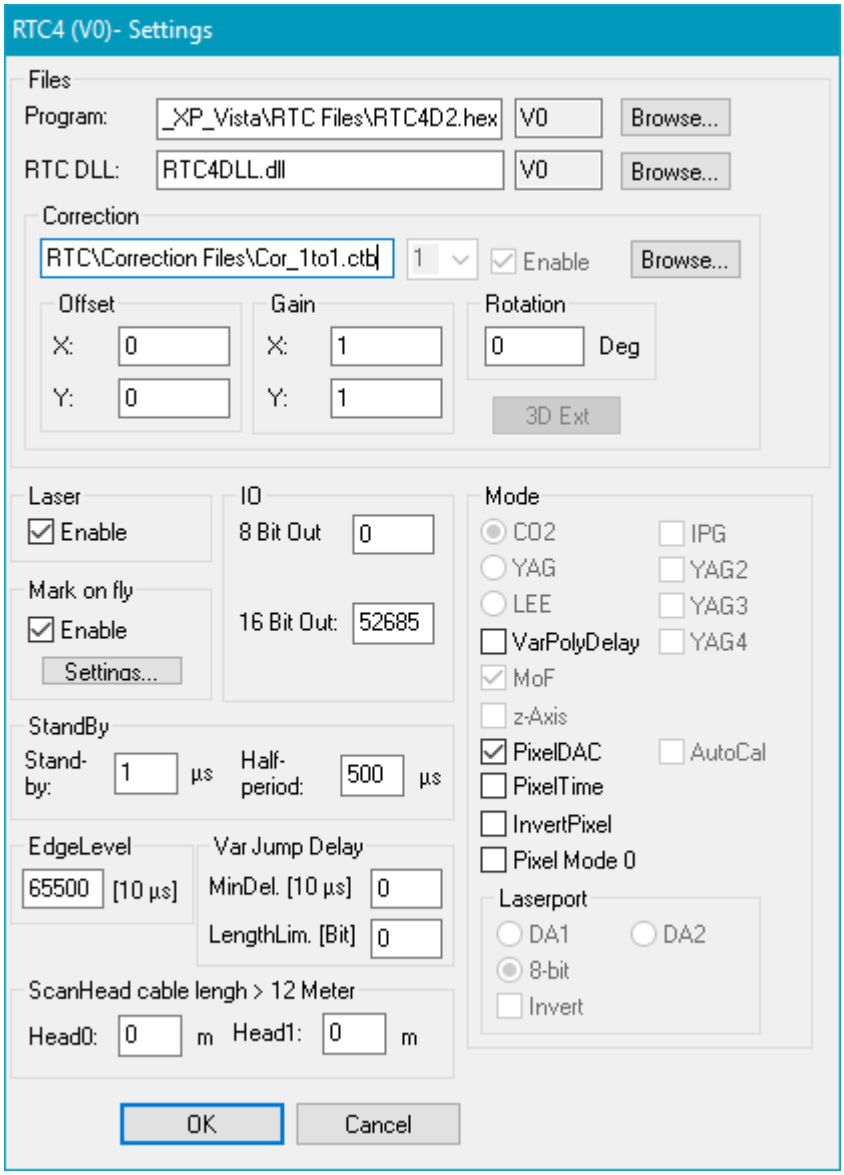


Figure 16: Card Settings for RTC4 Card

Files:

Program: Specifies the program file. This file is delivered together with the scanner card. The standard extension is *.hex. Click on the *Browse* button to make a search on the files location or type in the file name into the edit window.

RTC DLL: Specifies the *.dll file. This file is delivered together with the scanner controller card.

Using a 64-bit operating system you should nevertheless load a 32-bit RTC DLL, not the 64-bit one.

Correction: Specifies the location of the correction file. This file is delivered together with the scanner card. The standard extension is *.ctb. Click on *Browse* button to make a search on the files location or type in the

file name into the edit window. It is possible to load two different correction files. This is mainly used in connection with the secondary head feature of the RTC4.

Offset, Gain, Rotation: Allow a global adjustment for each correction file. These features are mainly used to adjust the fields of both heads when a secondary head is used. See also: Chapter Optic Settings Dialog.

3D Ext: Opens a dialog, where a Z-Table can be defined. For detailed information have a look at the RTC4 manual. See chapter Optic3D for RTC cards.

Laser: Globally enables or disables laser output.

Mark on Fly: Requires the MarkingOnTheFly RTC option. MOTF related parameters are described in chapter Card Specific: RTC cards.

StandBy: Globally enables standby mode.

Stand-by: Q-Switch length in µs for stand-by modus. If this is set to zero the stand-by mode is switched off.

Half-period: Half of the laser pulse period for stand-by modus.

Settings are done after leaving the global dialog Settings. Standby Settings for pens will overwrite these settings if enabled for a pen as soon as this pen is used. Also please remind that after marking has finished the settings from the Home Jump Style Pen are used.

EdgeLevel: The variable Polygon Delay gets very high if the angle of two successive vectors is close to 180° . This can lead to burn in effects. To prevent this an *EdgeLevel* value can be defined. If the Polygon Delay between two mark commands is greater than this value the RTC4 card switches off the laser after the first command and after the laser-off delay is over. Then a new polygon marking with the second vector will be started.

Var Jump Delay: Normally after a jump command a constant jump delay is inserted. But for very small jumps it is not necessary to have such a long jump delay. The jump delay can be reduced without loosing marking quality. The miminum delay is the delay for a jump of length 0.

IO:

Lamp/8-bit: Sets the 8 bit or one of the digital outputs of the RTC Card during start up (as selected under Laserport). The 8 bit Output corresponds to the write_8bit_port command of the RTC.

16 BIT Out: Sets the 16 bit output of the RTC Card during start up.

Figure 17: If *LEE Mode* is selected, the eighth bit is selectable separately

Mode: Here you can choose the type of the laser.

YAG2, YAG3 and YAG4: YAG4 corresponds to Laser Mode 4 like described in the SCANLAB RTC manual - YAG2 and YAG3 respectively.

VarPolyDelay: If checked the length of the polygon delay gets varied depending on the angle between two successive vectors.

MoF: Shows whether the scanner card is able to do MarkingOnTheFly or not.

z-Axis: Indicates whether the card is able to do 3D Marking or not.

PixelDAC: Enables Amplitude Modulation.

PixelTime: Enables Pulse Width Modulation. For more details see chapter Pulse Modulation.

InvertPixel: Inverts bitmap pixels.

Pixel Mode 0: See chapter Pixelmode.

AutoCal: Displays the ScAutoCalib button within the functionality object toolbar to generate AutoCalib control objects in the entity list, see section AutoCal Control Object.

Laserport: Defines the port that sends the power signal for the laser if the laser is not a CO2 Laser. For a CO2 Laser the power signal is done by modulating the Laser A signal.

Invert: Here the laser power value can be inverted. Note: In case of 8-bit this checkbox does not invert the bits.

3.3.7 RTC-5 Settings

Card Settings for RTC-5 card. This dialog can also be opened when SAMLight is running: *Menu bar → Settings → System → Card → Advanced...→ Driver Settings*

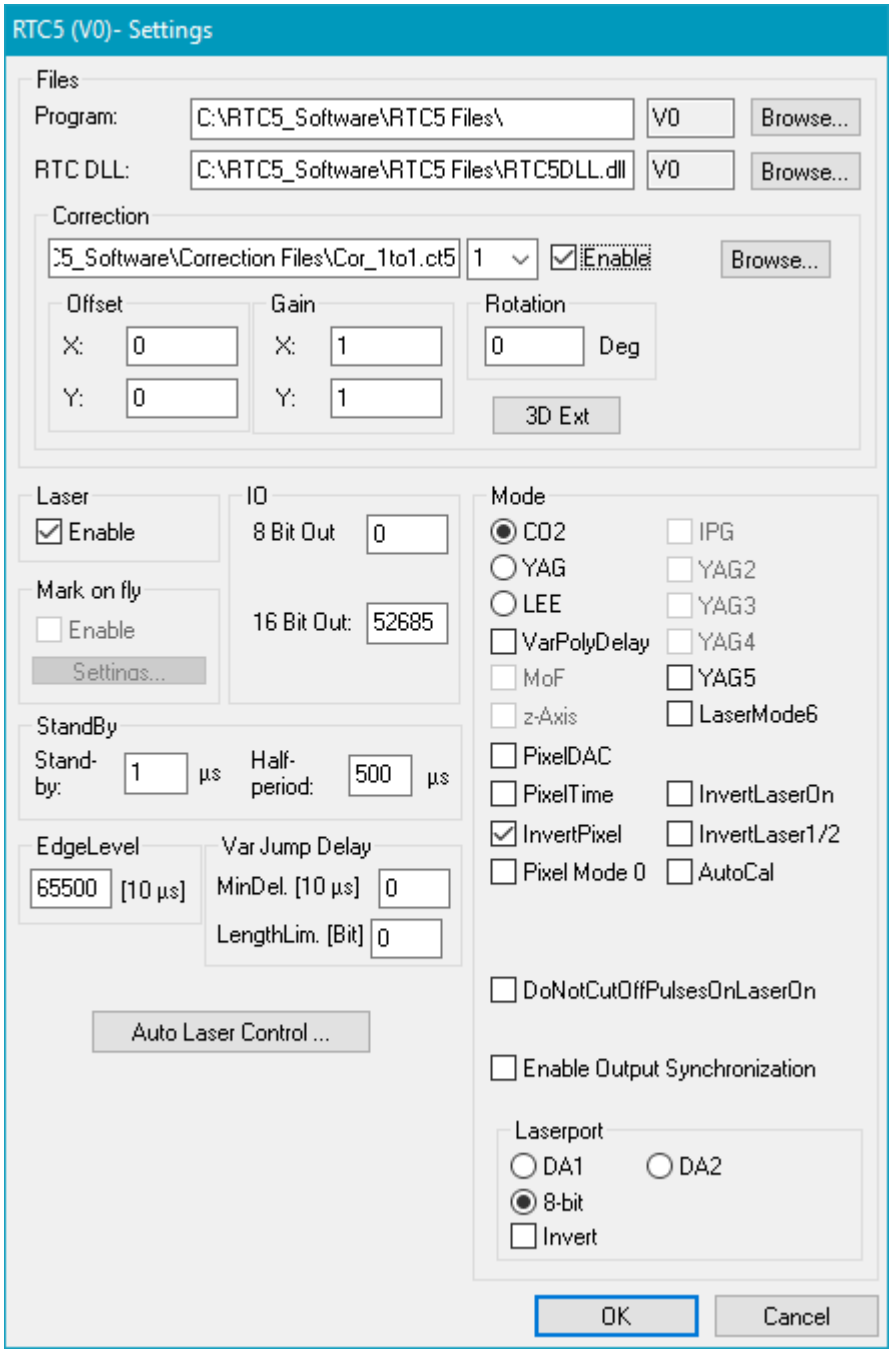


Figure 18: Card Settings for RTC5 Card

Files:

Program: Specifies the path to the *RTC5 Files* folder, delivered together with the scanner controller card driver.

RTC DLL: Specifies the path to the *RTC5DLL.dll* file. This file is in the *RTC5 Files* folder, delivered together with the scanner controller card driver.

Using a 64 bit operating system you should nevertheless load a 32 bit RTC5DLL.dll, not the 64 bit RTC5DLLx64.dll.

Correction: Specifies the location of the correction file. This file is delivered together with the scanner card. The extension is *.ctb or *.ct5. Click on *Browse* button to make a search on the files location or type in the file name into the edit window. It is possible to load two different correction files. This is mainly used in connection with the secondary head feature of the RTC5.

Offset, Gain, Rotation: Allow a global adjustment for each correction file. These features are mainly used to adjust the fields of both heads when a secondary head is used. See also: Chapter Optic Settings Dialog.

For the RTC5 card the X and Y Gain values have to be equal.

3D Ext: Opens a dialog, where a Z-Table can be defined. For detailed information have a look at the RTC5 manual. See chapter Optic3D for RTC cards.

Laser: Globally enables or disables laser output.

Mark on Fly: Requires the MarkingOnTheFly RTC option. MOTF related parameters are described in chapter Card Specific: RTC cards.

StandBy: Globally enables standby mode.

Stand-by: Q-Switch length in µs for stand-by modus. If this is set to zero the stand-by mode is switched off.

Half-period: Half of the laser pulse period for stand-by modus.

Settings are done after leaving the global dialog Settings. Standby Settings for pens will overwrite these settings if enabled for a pen as soon as this pen is used.

EdgeLevel: The variable Polygon Delay gets very high if the angle of two successive vectors is close to 180° . This can lead to burn in effects. To prevent this an *EdgeLevel* value can be defined. If the Polygon Delay between two mark commands is greater than this value the RTC5 card switches off the laser after the first command and after the laser-off delay is over. Then a new polygon marking with the second vector will be started.

Var Jump Delay: Normally after a jump command a constant jump delay is inserted. But for very small jumps it is not necessary to have such a long jump delay. The jump delay can be reduced without loosing marking quality. The miminum delay is the delay for a jump of length 0.

IO:

Lamp/8-bit: Sets the 8 bit or one of the digital outputs of the RTC Card during start up (as selected under Laserport). The 8 bit Output corresponds to the write_8bit_port command of the RTC.

16 BIT Out: Sets the 16 bit output of the RTC Card during start up.

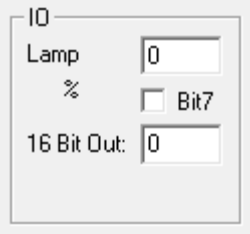


Figure 19: If *LEE Mode* is selected, the eighth bit is selectable separately

Mode: Here you can choose the type of the laser.

YAG2, YAG3 and YAG4: YAG4 corresponds to Laser Mode 4 like described in the SCANLAB RTC manual - YAG2 and YAG3 respectively.

LaserMode6: A synchronization signal will be given at Laser1 output for *laser active* and *laser standby* operation. Please see RTC5 manual for detailed information.

VarPolyDelay: If checked the length of the polygon delay gets varied depending on the angle between two successive vectors.

MoF: Shows whether the scanner card is able to do MarkingOnTheFly or not.

z-Axis: Indicates whether the card is able to do 3D Marking or not.

PixelDAC: Enables Amplitude Modulation.

PixelTime: Enables Pulse Width Modulation. For more details see chapter Pulse Modulation.

InvertPixel: Inverts bitmap pixels.

Pixel Mode 0: See chapter Pixelmode.

InvertLaserOn: Inverts the LaserOn status bit (15-pin SUB-D Laser connector).

InvertLaser1/2: Inverts the Laser1 and Laser2 status bits (15-pin SUB-D Laser connector).

AutoCal: Displays the ScAutoCalib button within the functionality object toolbar to generate AutoCalib control objects in the entity list, see section AutoCal Control Object.

DoNotCutOffPulsesOnLaserOn: The final pulse is fully executed despite completion of the LASERON signal. Please consult RTC5 manual for Ctrl Command set_laser_control at Bit #0 Pulse Switch Setting.

Enable Output Synchronization: Synchronize scanner signals with a freely running laser. Please read RTC5 manual section 7.4.10 for detailed information.

Laserport: Defines the port that sends the power signal for the laser if the laser is not a CO2 Laser. For a CO2 Laser the power signal is done by modulating the Laser A signal.

Invert: Here the laser power value can be inverted. Note: In case of 8-bit this checkbox does not invert the bits.

Auto Laser Control...: By clicking this button a new window opens where a position or speed control of the laser can be defined. For more details pleas refer to the RTC5 manual. See chapter Auto Laser Control.

3.3.7.1 Auto Laser Control

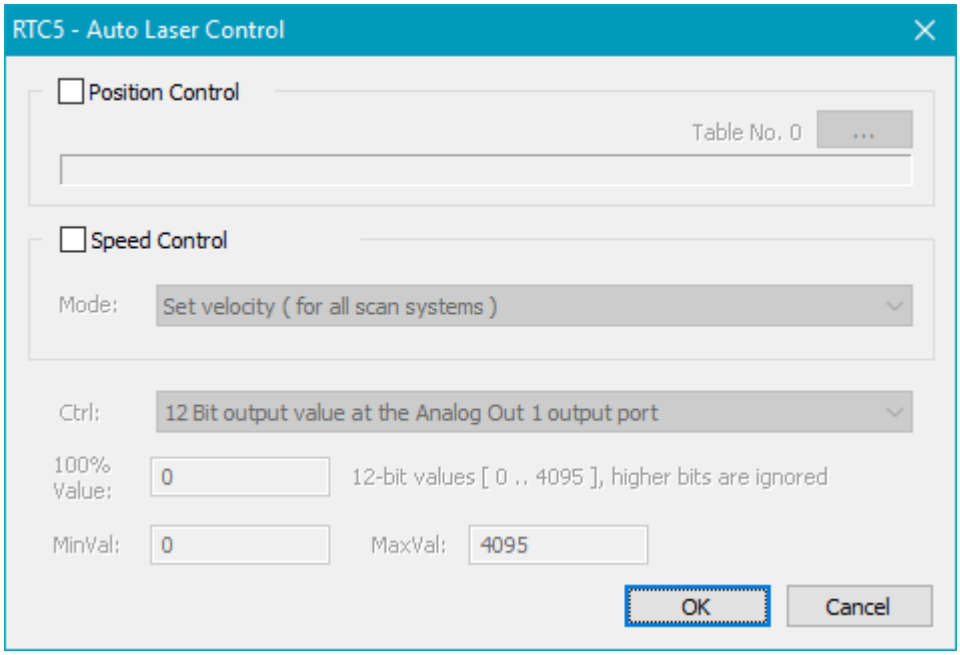


Figure 20: Auto Laser Control Settings for RTC5 Card

Position Control: Activates the position-dependent laser control, which performs an automatic position-dependent correction. For more information please look at RTC5 manual at subsection Position-Dependent Laser Control.

Speed Control: Activates the speed-dependent laser control, which performs an automatic speed-dependent correction. For more information please look at RTC5 manual at subsection Speed-Dependent Laser Control.

3.3.8 RTC-6 Settings

Card Settings for RTC-6 card. This dialog can also be opened when SAMLight is running: *Menu bar → Settings → System → Card → Advanced...→ Driver Settings*

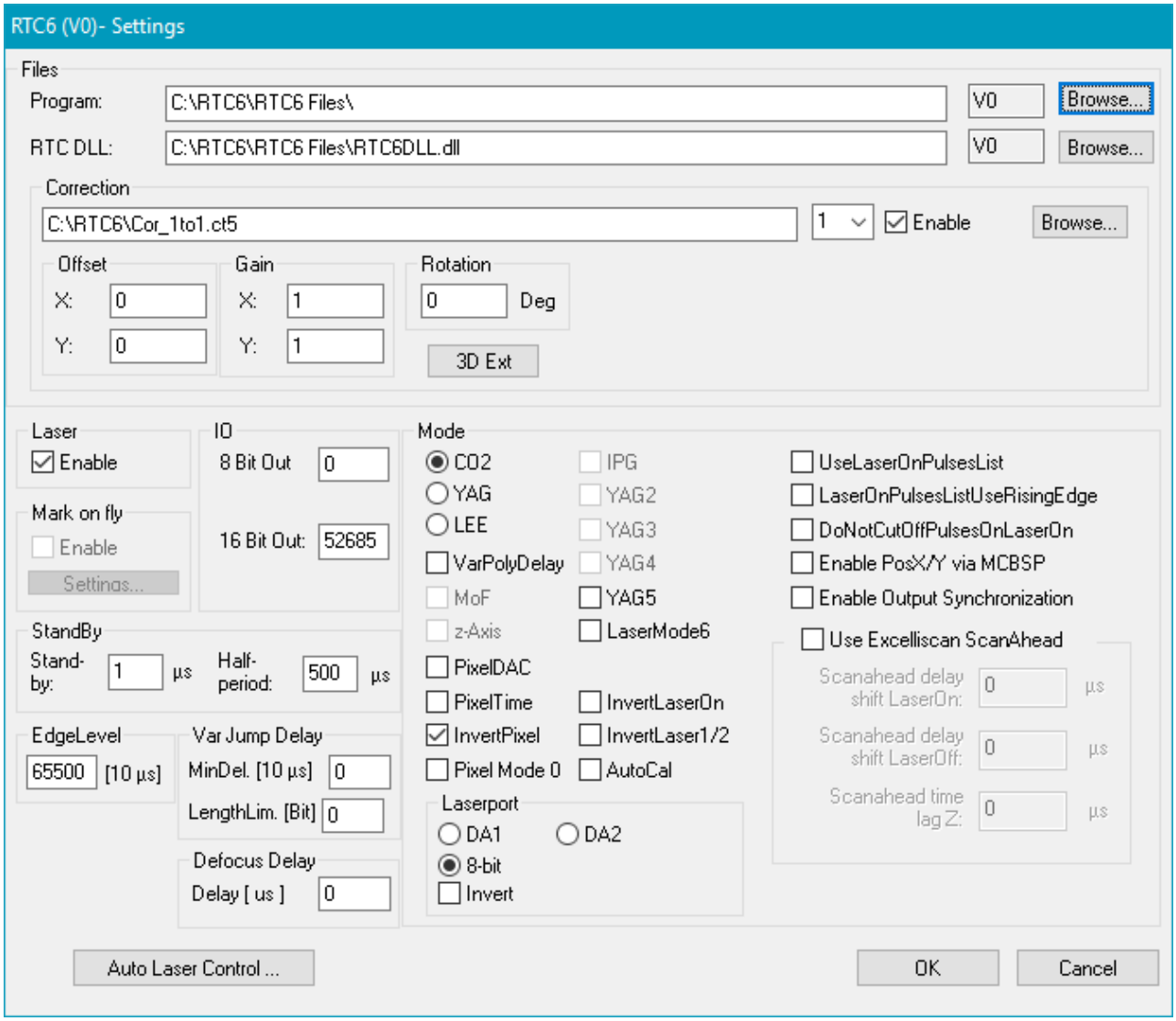


Figure 21: Card Settings for RTC6 Card

Files:

Program: Specifies the path to the *RTC6 Files* folder, delivered together with the scanner controller card driver.

RTC DLL: Specifies the path to the *RTC6DLL.dll* file. This file is in the *RTC6 Files* folder, delivered together with the scanner controller card driver.

NOTE

Using a 64 bit operating system you should nevertheless load a 32 bit RTC6DLL.dll, not the 64 bit RTC6DLLx64.dll.

Correction: Specifies the location of the correction file. This file is delivered together with the scanner card. The extension is *.ctb or *.ct5. Click on *Browse* button to make a search on the files location or type in the file name into the edit window. It is possible to load two different correction files. This is mainly used in connection with the secondary head feature of the RTC6.

Offset, Gain, Rotation: Allow a global adjustment for each correction file. These features are mainly used to adjust the fields of both heads when a secondary head is used. See also: Chapter Optic Settings Dialog.

For the RTC6 card the X and Y Gain values have to be equal.

3D Ext: Opens a dialog, where a Z-Table can be defined. For detailed information have a look at the RTC6 manual. See chapter Optic3D for RTC cards.

Laser: Globally enables or disables laser output.

Mark on Fly: Requires the MarkingOnTheFly RTC option. MOTF related parameters are described in chapter Card Specific: RTC cards.

StandBy: Globally enables standby mode.

Stand-by: Q-Switch length in µs for stand-by modus. If this is set to zero the stand-by mode is switched off.

Half-period: Half of the laser pulse period for stand-by modus.

Settings are done after leaving the global dialog Settings. Standby Settings for pens will overwrite these settings if enabled for a pen as soon as this pen is used.

EdgeLevel: The variable Polygon Delay gets very high if the angle of two successive vectors is close to 180° . This can lead to burn in effects. To prevent this an *EdgeLevel* value can be defined. If the Polygon Delay between two mark commands is greater than this value the RTC6 card switches off the laser after the first command and after the laser-off delay is over. Then a new polygon marking with the second vector will be started.

Var Jump Delay: Normally after a jump command a constant jump delay is inserted. But for very small jumps it is not necessary to have such a long jump delay. The jump delay can be reduced without loosing marking quality. The miminum delay is the delay for a jump of length 0.

IO:

Lamp/8-bit: Sets the 8 bit or one of the digital outputs of the RTC Card during start up (as selected under Laserport). The 8 bit Output corresponds to the write_8bit_port command of the RTC.

16 BIT Out: Sets the 16 bit output of the RTC Card during start up.

Figure 22: If *LEE Mode* is selected, the eighth bit is selectable separately

Mode: Here you can choose the type of the laser.

YAG2, YAG3 and YAG4: YAG4 corresponds to Laser Mode 4 like described in the SCANLAB RTC manual - YAG2 and YAG3 respectively.

LaserMode6: A synchronization signal will be given at Laser1 output for *laser active* and *laser standby* operation. Please see RTC6 manual for detailed information.

VarPolyDelay: If checked the length of the polygon delay gets varied depending on the angle between two successive vectors.

MoF: Shows whether the scanner card is able to do MarkingOnTheFly or not.

z-Axis: Indicates whether the card is able to do 3D Marking or not.

PixelDAC: Enables Amplitude Modulation.

PixelTime: Enables Pulse Width Modulation. For more details see chapter Pulse Modulation.

InvertPixel: Inverts bitmap pixels.

Pixel Mode 0: See chapter Pixelmode.

InvertLaserOn: Inverts the LaserOn status bit (15-pin SUB-D Laser connector).

InvertLaser1/2: Inverts the Laser1 and Laser2 status bits (15-pin SUB-D Laser connector).

AutoCal: Displays the ScAutoCalib button within the functionality object toolbar to generate AutoCalib control objects in the entity list, see section AutoCal Control Object.

DoNotCutOffPulsesOnLaserOn: The final pulse is fully executed despite completion of the LASERON signal. Please consult RTC5 manual for Ctrl Command set_laser_control at Bit #0 Pulse Switch Setting.

Enable Output Synchronization: Synchronize scanner signals with a freely running laser. Please read RTC5 manual section 7.4.10 for detailed information.

Use Excelliscan ScanAhead: Activates the SCANahead control functionality of an excelliSCAN scan head. Important: the excelliSCAN must be powered before starting SAMLight. Additional RTC6 SCANahead parameters can be set at Pen Misc Settings.

Scanahead delay shift LaserOn: Sets the time delay for the LaserOn signal in µs. Please consult RTC6+excelliSCAN manual for Ctrl Command set_scanahead_laser_delay_shift for more information.

Scanahead delay shift LaserOff: Sets the time delay for the LaserOff signal in µs. Please consult RTC6+excelliSCAN manual for Ctrl Command set_scanahead_laser_delay_shift for more information.

Scanahead time lag Z: Sets the time delay for the z time lag compensation in µs. Please consult RTC6+excelliSCAN manual for Ctrl Command set_timelag_compensation for more information.

Laserport: Defines the port that sends the power signal for the laser if the laser is not a CO2 Laser. For a CO2 Laser the power signal is done by modulating the Laser A signal.

Invert: Here the laser power value can be inverted. Note: In case of 8-bit this checkbox does not invert the bits.

Auto Laser Control...: By clicking this button a new window opens where a position or speed control of the laser can be defined. For more details pleas refer to the RTC5 manual. See chapter Auto Laser Control.

3.3.8.1 Auto Laser Control

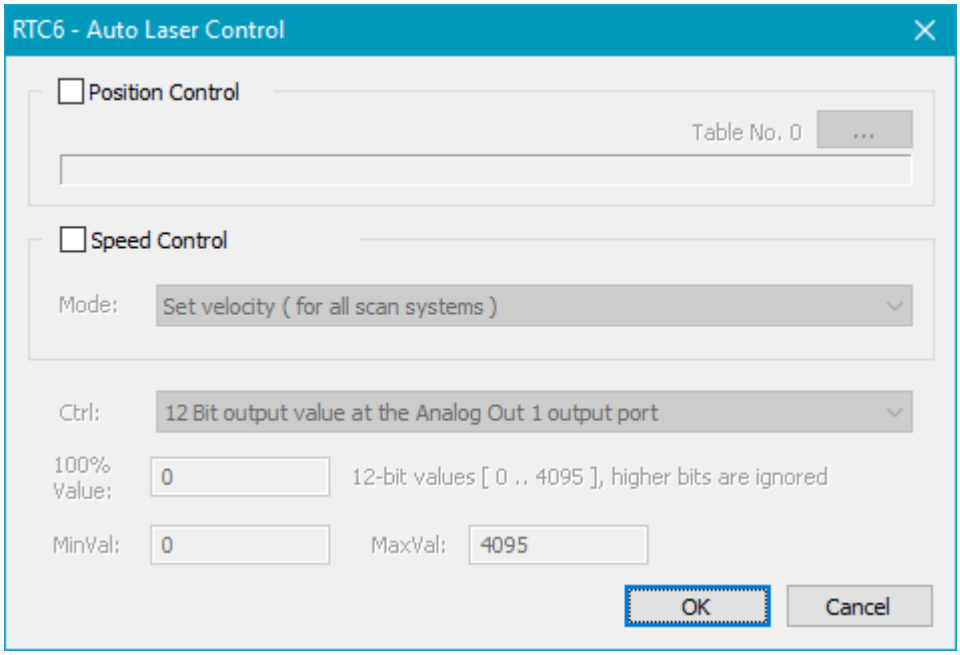


Figure 23: Auto Laser Control Settings for RTC6 Card

Position Control: Activates the position-dependent laser control, which performs an automatic position-dependent correction. For more information please look at RTC5 manual at subsection Position-Dependent Laser Control.

Speed Control: Activates the speed-dependent laser control, which performs an automatic speed-dependent correction. For more information please look at RTC5 manual at subsection Speed-Dependent Laser Control.

3.3.9 RTC Scanalone Settings

Card Settings for RTC Scanalone card. This dialog can also be opened when SAMLight is running: *Menu bar → Settings → System → Card → Advanced...> Driver Settings*

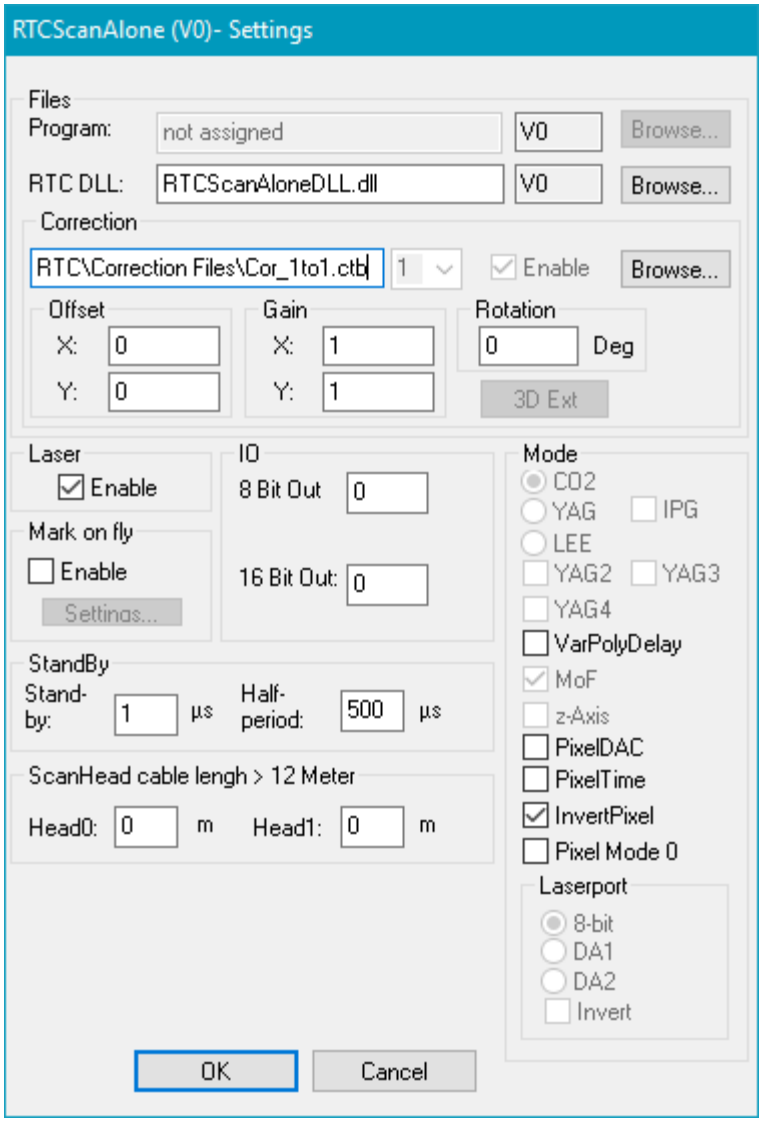


Figure 24: Card Settings for ScanAlone Card

Files:

Program: Specifies the program file. This file is delivered together with the scanner card. The standard extension is *.hex. Click on the *Browse* button to make a search on the files location or type in the file name into the edit window.

RTC DLL: Specifies the *.dll file. This file is delivered together with the scanner controller card.

Correction: Specifies the location of the correction file. This file is delivered together with the scanner card. The standard extension is *.ctb. Click on *Browse* button to make a search on the files location or type in the file name into the edit window. It is possible to load two different correction files. This is mainly used in connection with the secondary head feature of the RTCScanAlone.

Offset, Gain, Rotation: Allow a global adjustment for each correction file. These features are mainly used to adjust the fields of both heads when a secondary head is used. See also: Chapter Optic Settings Dialog.

3D Ext: Opens a dialog, where a Z-Table can be defined. For detailed information have a look at the RTC

ScanAlone manual.

Laser: Globally enables or disables laser output.

Mark on Fly: Requires the MarkingOnTheFly SAMLight option.

Settings...: Opens the Marking On The Fly Dialog. For an explanation how to do settings for mark on fly see chapter How to Mark on the Fly.

Enable: Check this to enable *Marking on the fly*.

StandBy: Globally enables standby mode.

Stand-by: Q-Switch length in µs for stand-by modus. If this is set to zero the stand-by mode is switched off.

Half-period: Half of the laser pulse period for stand-by modus.

Settings are done after leaving the global dialog Settings. Standby Settings for pens will overwrite these settings if enabled for a pen as soon as this pen is used.

ScanHead cable length > 12 Meter: Put in the cable length of each head if one cable is longer than 12m.

IO: Sets the 8 Bit or one of the digital outputs of the RTC Card during start up (as selected under Laserport). The 8 Bit Output corresponds to the write_8bit_port command of the RTC.

Lamp/8-bit: Sets the 8 bit or one of the digital outputs of the RTC Card during start up (as selected under Laserport). The 8 bit Output corresponds to the write_8bit_port command of the RTC.

16 BIT Out: Sets the 16 bit output of the RTC Card during start up.

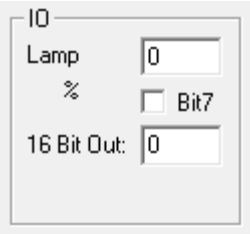


Figure 25: If *LEE Mode* is selected, the eighth bit is selectable separately

Mode: Here you can choose the type of the laser.

VarPolyDelay: If checked the length of the polygon delay gets varied depending on the angle between two successive vectors.

MoF: Shows whether the scanner card is able to do MarkingOnTheFly or not.

z-Axis: Indicates whether the card is able to do 3D Marking or not.

PixelDAC: Enables Amplitude Modulation.

PixelTime: Enables Pulse Width Modulation. For more details see chapter Pulse Modulation.

InvertPixel: Inverts bitmap pixels.

Pixel Mode 0: See chapter Pixelmode.

Laserport: Defines the port that sends the power signal for the laser if the laser is not a CO2 Laser. For a CO2 Laser the power signal is done by modulating the Laser A signal.

4 Motion Control Settings

The type of motion controller needs to be defined in the text file ***sc_motion_settings.txt***. This file can be found in the folder <*SCAPS*>*system*\.

Type=#: The type of the motion controller, where # stands for a number that specifies the controller (see table below):

Type=#	Name of motion controller	Corresponding settings file in <SCAPS>\system\
0	Disable motion control	none
1	IMS Stepper Drives	ims_settings.txt
2	Standard (not supported)	not supported any more
3	Microcon Drive (not supported)	not supported any more
4	External custom controller	defined in customized *.dll
5	IMS MDrive	sc_motion_mdrive_settings.txt
6	Faulhaber motion controller	sc_motion_faulmc_settings.txt
7	Isel IT Stepper Controller / DNC	sc_motion_iselit_settings.txt
8	Generic stepper controller	sc_motion_stepper_settings.txt
9	Generic RS232 interface	sc_motion_generic232_settings.txt
10	SHS 2000 Star	sc_motion_shstar2000_settings.txt
11	Jena Ecostep100	sc_motion_jenaecostep100_settings.txt
12	IO Switcher	sc_motion_ioswitcher_settings.txt
13	Isel CanApi controller	sc_motion_isel_settings.txt
14	USC-2 stepper controller	sc_motion_stepper_settings.txt

Table 2: Available motion types

Direct Motion Control: In the following all features of the Direct Motion Control are described. Depending on the capabilities of the used motion controller some of the functions may be not available.

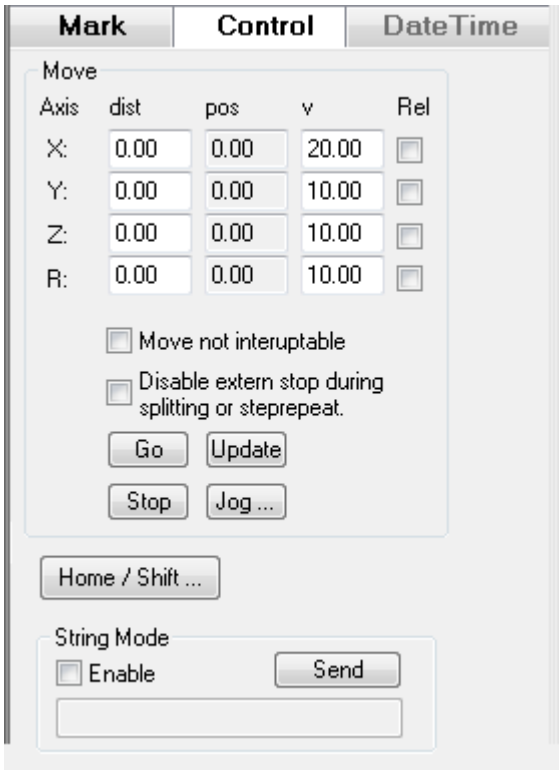


Figure 26: Direct motion control dialog

Move:

Axis: Movement distance and speed definitions for the axis. Each axis requires one control card.

Rel: If checked a relative movement is performed instead of an absolute movement.

Go: Moves all axes to the defined values above.

Stop: Stops the movement. In some cases you can loose the current position! Then a homing is necessary to re-calibrate your axes.

Update: By pressing this button the actual position is updated. This might be necessary if manual motions are done. The *Update* Button is not available for stepper motor controllers.

Jog...: Opens the Jog Dialog.

Home / Shift: Opens direct motion control Home / Shift dialog. Homing of a single axis is available by right-clicking on this button and then selecting the desired axis.

String Mode: A RS-232 string command can be send to the motion controller directly on *Send*.

4.1 Jog Dialog

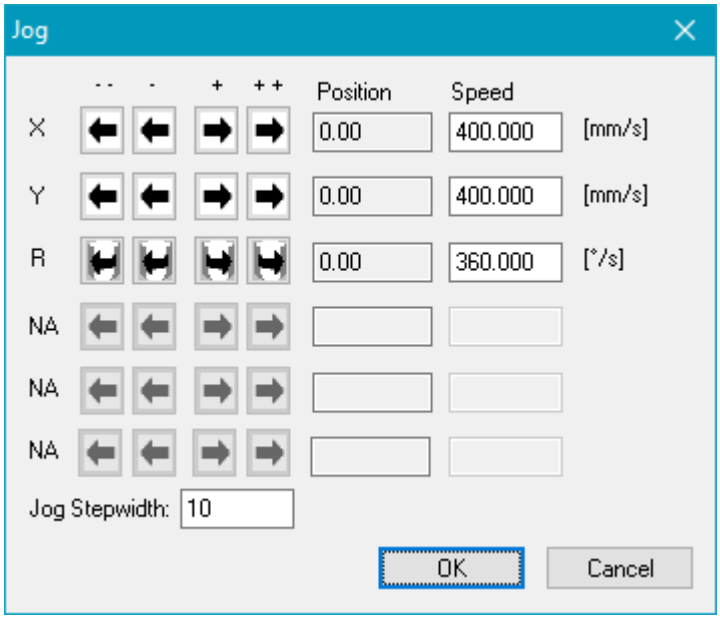


Figure 27: Jog dialog

Jog: Clicking on this button opens a dialog where the motor can be moved in both directions by "Jog Stepwidth" (++) or by "Jog Stepwidth / 10" (+). The default jog stepwidth value can be set in *Settings→System→Extras*. If the dialog is closed and opened again, this value is restored to the value that has been set in the menu. For each axis an independent speed can be used.

4.2 Home / Shift Dialog

Figure 28: Direct motion control Home / Shift dialog

Home: When leaving the *Home / Shift Dialog* with *OK*, all axes with enabled *Home* checkbox will start the homing procedure.

Shift: When the direct motion control Home / Shift dialog is opened, the current Shift values are shown. When leaving the Home / Shift dialog with *OK*, the shift values will be applied. To reset the current shift of an axis, set the shift value to *0*. The *Home* and *Shift* functionality can be combined, first the homing procedure is performed, then the shifts will be applied.

4.3 Step & direction motion controller

The following step and direction motion controllers are supported.

4.3.1 Type 14 - USC-2 stepper controller

Key features

- Available for USC-2 cards only
- Up to 6 axes can be controlled
- All axes can move at the same time if they have the same speed. Otherwise they will move one after another. The X-, Y- and Z-axis can move interpolated so that they reach the final position of the movement at the same time.
- The IOs of the USC-2 card are used to control the stepper
- Control the motor via step and direction signal
- The signals are controlled on the USC-2 card directly
- The maximum possible frequency of the step signals is 16666.7 Hz
- Available in Flash-Mode
- Fast homing procedures available
- Keeps position values of axes after a movement was interrupted
- It can be used via regular motion control in SAM programming, or via G-Code commands (RS-232, Telnet or flash commands)

Required settings

- Set 'Type=14' in sc_motion_settings.txt in <*SCAPS*>\system\
- Define sc_motion_stepper_settings.txt in <*SCAPS*>\system\ as described below
- Enable 'Motion Control' in SAMLight (Settings→System→Extras).
- Use SAMLight Version later than May 2013 and use related firmware on the USC-2 card

How to create the settings file

- The file 'sc_motion_stepper_settings.txt' is a plain text file that contains different configuration parameters.
- A line which begins with a # sign is interpreted as a comment and will be ignored.
- All parameters have to start exactly at the beginning of a line.
- Global parameters have to be specified only once at the head of the settings file.
- Axis-specific parameters have to be defined for all axes separately.
- The axis-specific (and the optional) parameters have to be arranged in one block of parameters per axis. Each block begins with the parameter 'axis'.
- The optional parameters do not have to be specified in the case you do not want to use them.
- A parameter is defined by typing <parameter>=<value>, parameter and values are described below,
  e.g. 'axis=0' defines the parameter 'axis' with the value '0'
- To save the stepper settings on the USC-2 card (if you want to use it in Flash-mode) you need to click the 'Store' button in the System→Settings→Optic→Advanced dialog in SAMLight.

Global parameters (type 14): These parameters have to be defined once at the head of the settings file.

Debug	Enables the debug log.	
	Value	Function
	0	Disable debug mode
	1	Enable debug mode
	Enable debug mode to log debug information in '<*SCAPS*> \system\sc_motion_stepper_debuglog.txt'. Only enable this option if you need it, because there can be huge amounts of data that are logged into this file.	

DisableHomingDuringStartUp	No homing on startup.	
	Value	Function
	0	Enable homing on startup of SAMLight
	1	Disable homing on startup of SAMLight
	If set to '1', homing is just performed when pressing 'Control→Home' and not when software is started.	

Axis parameters (type 14): All of the following parameters have to be defined for each axis after the global parameters in the settings file. Each of these axes can be configured differently.

axis	Sets the number of axis.	
	Value	Function
	0...5	Index of axis
	Zero based index of up to 6 axes. Defines the order of axis movement and display order in 'SAMLight→Control'. This parameter has to be at the beginning of each axis specific parameter block.	

dname	Sets the name of axis.	
	Value	Function
	a...Z	Name of axis
	Only the first character is used as axis-name in 'SAMLight→Control'.	

mode	Sets the type of the axis.	
	Value	Function
	POSITION	Defines a straight axis [mm].
	ANGLE	Defines a rotational axis [°].
	Depending on the mode of the axis several parameters have to be adjusted (factor, incperrot, defspeed, see below).	

incperrot	Converts degrees into steps for ANGLE mode.	Unit: steps/360°
	This value defines how many increments are needed for a whole rotation. 'factor' has to be equal to (1/360)*'incperrot'. This parameter is only used in ANGLE mode.	

factor	Converts mm (or °) into steps.	
	POSITION mode, unit: steps/ mm	ANGLE mode, unit: steps/°
		Value has to be equal to **(1/360)*'incperrot'**
	This value defines how many increments are needed for 1mm (or 1°). Depending on the factor of the axis several parameters have to be adjusted (incperrot, llimit, hlimit, hslimit, accel, decel, refspeed, refspeed2, refpos see below).	

llimit	Sets the lower limit.	Unit: steps
	No movement below this limit will be possible. Minimum value for llimit is '-1E13. To get the 'llimit' in steps like it is required here the value in mm (or °) has to be multiplied by the 'factor'.	

hlimit	Sets the upper limit.	Unit: steps
	No movement above this limit will be possible. Maximum value for hlimit value is '1E13' To get the 'hlimit' in steps like it is required here the value in mm (or °) has to be multiplied by the 'factor'.	

hslimit	Sets the upper speed limit.	Unit: steps/s
	No speed above this speed limit will be possible. The maximum possible frequency of the direction signals is 16666.7 Hz. To get the 'hslimit' in steps/s like it is required here the value in mm/s (or °/s) has to be multiplied by the 'factor'.	

defspeed	Sets default speed displayed in SAMLight user interface.	
	POSITION mode, unit: mm/s	ANGLE mode, unit: °/s
	This value has to be smaller or equal than 'hslimit'/'factor'.	

accel	Sets increase of speed.	Unit: steps/s²
	The values '0' and '-1' are not valid. If you want to avoid acceleration in general you can set 'accStartSpeed=hslimit/ factor'. To get the 'accel' in steps/s² like it is required here the value in mm/s² (or °/s²) has to be multiplied by the 'factor'.	

decel	Sets decrease of speed.	Unit: steps/s²
	Please use only positive values for this parameter. The values '0' and '-1' are not valid. If you want to avoid deceleration in general you can set 'decStopSpeed=hslimit/factor'. To get the 'decel' in steps/s² like it is required here the value in mm/s² (or °/s²) has to be multiplied by the 'factor'.	

accStartSpeed	Sets start speed.	
	POSITION mode, unit: mm/s	ANGLE mode, unit: °/s
	This value has to be positive and smaller or equal than 'hslimit'/'factor'. If the in SAMLight demanded speed is smaller than the value for this parameter the axis will move with the lower speed without acceleration.	

decStopSpeed	Sets stop speed.	
	POSITION mode, unit: mm/s	ANGLE mode, unit: °/s
	This value has to be positive and smaller or equal than 'hslimit'/'factor'. If the in SAMLight demanded speed is smaller than the value for this parameter the axis will move with the lower speed without deceleration.	

stepIO	Sets output bit of the scanner controller card for the step signal.
	Value: refer to Stepper I/O parameters

dirIO	Sets output bit of the scanner controller card for the direction signal.
	Value: refer to Stepper I/O parameters

dirvalue	Sets the polarity of the 'dirIO' signal for positive movement.	
	Value	Function
	0	low active
	1	high active

DelayAfterDir	Sets a delay which is executed after change of the direction.
	This command is used to specify a delay in us to be waited between a change in direction and the first step in the new direction.

refIO	Sets input bit of the scanner controller card for the reference signal.
	Value: refer to Stepper I/O parameters
	Since all axes move successively the 'refIO' bit can be the same for all axes. Although the parameter 'refIO' has the same value for multiple axes it still has to be defined for every axis.

refvalue	Sets the polarity of the 'refIO' signal.	
	Value	Function
	0	low active
	1	high active

refmode	Sets the behavior of homing movement.	
	Value	Function
	Common reference modes:	
	0	No homing movement, current position is set to 'refpos' Use 'refmode=0' together with 'RefOnlyForHome=1'.
	1	Go to switch in neg. dir. and leave it in pos. dir.
	4	Go to switch in pos. dir. and leave it in neg. dir.
	Uncommon reference modes:	
	2	Go to switch in neg. dir. and leave it in neg. dir.
	5	Go to switch in pos. dir. and leave it in pos. dir.
	11	Go to switch in neg. dir. and leave it in pos. dir. (no timeout)
	44	Go to switch in pos. dir. and leave it in neg. dir. (no timeout)

refspeed	Sets the speed of the homing movement.	Unit: steps/s
	This value defines how fast the motor moves in case of a homing movement to find the reference switch. Reference movements are USC-2 controlled and have a much higher maximum speed than stepper type 8. When no reference signal is detected the homing movement is stopped after 10000 mm (or °). To get the 'refspeed' in steps/s like it is required here the value in mm/s (or °/s) has to be multiplied by the 'factor'.	

refspeed2	Sets the speed while leaving the reference switch.	Unit: steps/s
	If refspeed2 is '-1', 'refspeed/4' is used. When the reference signal is not released the switch leaving movement is stopped after 100 mm (or °). To get the 'refspeed2' in steps/s like it is required here the value in mm/s (or °/s) has to be multiplied by the 'factor'.	

refpos	Sets the home position of the axis.	Unit: steps
	This value will be set after SAMLight startup and after a homing movement. refpos should be outside the position limits 'llimit' and 'hlimit' to avoid that the reference switch is activated during normal movement. To get the 'refpos' in steps like it is required here the value in mm (or °) has to be multiplied by the 'factor'.	

Optional axis parameters (type 14): All of the following optional parameters can be defined for each axis. Each axis can be configured differently. If a parameter is not specified the corresponding feature will not be used.

<table>
<tr><td rowspan="4">RefOnlyForHome</td><td colspan="2">Sets movement behavior for this axis when reference switch is active during a normal movement.</td></tr>
<tr><td>Value</td><td>Function</td></tr>
<tr><td>0</td><td>Motion stops if reference switch is active.</td></tr>
<tr><td>1</td><td>Motion does not stop if reference switch is active (for normal movements).</td></tr>
<tr><td colspan="2">The behavior of the homing process will not change due to this parameter. The default value (if not defined) for this parameter is '0'.</td></tr>
</table>

Example sc_motion_stepper_settings.txt file for 1 straight and 1 rotational axis for 'Type=14':

```
# Global parameters:
Debug=0
DisableHomingDuringStartUp=1

# Z-axis parameters:
axis=0
dname=Z
mode=POSITION

factor=200.0
llimit=-400000
hlimit=400000
hslimit=6000
defspeed=20.0
accel=2000
decel=2000
accStartSpeed=5.0
decStopSpeed=5.0

stepIO=216
dirIO=217
dirvalue=0

refIO=16
refvalue=1
refmode=0
refspeed=2000
refspeed2=500
refpos=0.0
RefOnlyForHome=1

# R-axis parameters:
axis=1
dname=R
mode=ANGLE

factor=10.0
incperrot=3600
llimit=-1E13
hlimit=1E13
hslimit=6000
defspeed=400.0
accel=2000
decel=2000
accStartSpeed=5.0
decStopSpeed=5.0

stepIO=218
dirIO=219
dirvalue=0

refIO=17
refvalue=1
refmode=0
refspeed=2000
refspeed2=500
refpos=0.0
RefOnlyForHome=1
```

GCode cmd	Description
GCL M0	Stop execution (aborts moves).
GCL M46	Starts execution. This is required after GCL M0 for further motion commands.
GCL F?	Set movement speed to ? mm/s (e.g. GCL F10.0).
GCL G1??	Move to ??, where ?? is axis identifier (X, Y, Z, A, B, C) followed by position in mm (e.g. G1X1.5000). Multiple axis can be combined (e.g. G1X1Y2.5Z0)
GCL G90	Switch to absolute mode. After this all move coordinates are interpreted as absolute values

	(default).
GCL G91	Switch to relative mode. G1 commands are interpreted relative to previous position.
GCL G28?1	Home axis ?, where axis is X, Y, Z, A, B, C. Only one axis at a time.
GCL Q????	Get current axis position. ???? can be 7032-7037, for axis X, Y, Z, A, B, C respectively.

Table 3: Stepper GCode commands for a USC-2 in Flash mode

Necessary steps for changing from stepper type 8 to stepper type 14:

- The parameters 'sfactor', 'SignalAxisMoving', 'SignalAxisMovingStateBitPosition', 'SignalAxisMovingState', 'SignalAxisMovingStatePreDelay', 'MoveThisAxisFirst' and 'corr#' are not supported in the stepper type 14 any more.
- Optionally you can define the parameter 'RefOnlyForHome'. If you define 'RefOnlyForHome=1' you will have the behavior like you are used to from stepper mode 8.
- The units for 'accel' and 'decel' have been changed to steps/s². This leads to a linear acceleration and deceleration of the speed of the axes. You need to adjust the values for these parameters as well. In stepper type 14 the values for these parameters will be much higher. The values '0' and '-1' are not valid in stepper type 14.
- The 'refmode's 3 and 6 are no longer supported in stepper type 14.
- In stepper type 14 the X, Y and Z axes can move in parallel so please do not use the same value for different axes for the parameter 'refIO'. Do not use external stop ('refIO=1') as well because fast homing mode is not necessary in stepper mode 14 any more.

4.3.1.1 Motion Settings Dialog Type 14

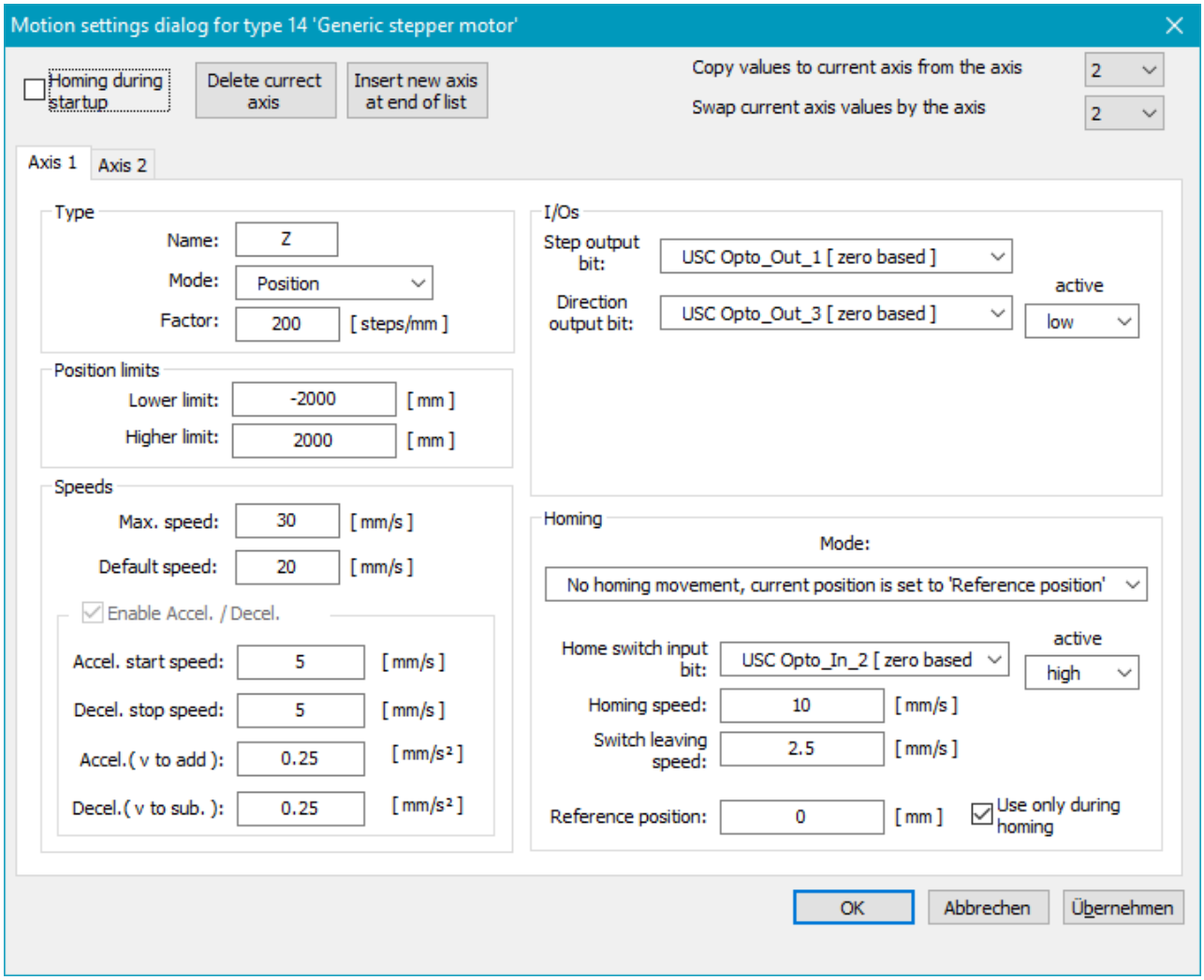


Figure 29: Direct motion control Home / Shift dialog

This dialog can be found in SAMLight→Settings→Extras→Motion Settings Dialog.

4.3.2 Type 8 - Generic stepper controller

Key features

- Available for USC and RTC controller cards
- Up to 7 axes can be controlled
- Axes move one after another
- The IOs of the scanner controller card are used to control the stepper
- Control the motor via step and direction signal
- Homing procedures available

Required settings

- Set 'Type=8' in sc_motion_settings.txt in <*SCAPS*>\system\
- Define sc_motion_stepper_settings.txt in <*SCAPS*>\system\ like described below
- Enable 'Motion Control' in SAMLight (Settings→System→Extras).

How to create the settings file

- The file 'sc_motion_stepper_settings.txt' is a plain text file that contains different configuration parameters.
- A line which begins with a # sign is interpreted as a comment and will be ignored.
- All parameters have to start exactly at the beginning of a line.
- Global parameters have to be specified only once at the head of the settings file.
- Axis-specific parameters have to be defined for all axes separately.
- The axis-specific (and the optional) parameters have to be arranged in one block of parameters per axis. Each block begins with the parameter 'axis'.
- The optional parameters do not have to be specified in the case you do not want to use them.
- A parameter gets defined by typing <parameter>=<value>, parameter and values are described below,
  e.g. 'axis=0' defines the parameter 'axis' with the value '0'

Global parameters (type 8): These parameters have to be defined once at the head of the settings file.

Debug	Enables the debug log.	
	Value	Function
	0	Disable debug mode
	1	Enable debug mode
	Enable debug mode to log debug information in '<*SCAPS*>\system\sc_motion_stepper_debuglog.txt'. Only enable this option if you need it, because there can be huge amounts of data that are logged into this file.	

DisableHomingDuringStartUp	No homing on startup.	
	Value	Function
	0	Enable homing on startup of SAMLight
	1	Disable homing on startup of SAMLight
	If set to '1', homing is just performed when pressing 'Control→Home' and not when software is started.	

Axis parameters (type 8): All of the following parameters has to be defined for each axis after the global parameters in the settings file. Each of these axes can be configured differently.

axis	Sets the number of axis.	
	Value	Function
	0...6	Index of axis
	Zero based index of up to 7 axes. Defines the order of axis movement and display order in 'SAMLight→Control'. This parameter has to be at the beginning of each axis specific parameter block.	

dname	Sets the name of axis.	
	Value	Function
	a...Z	Name of axis
	Only the first character is used as axis-name in 'SAMLight→Control'.	

mode	Sets the type of the axis.	
	Value	Function
	POSITION	Defines a straight axis [mm].
	ANGLE	Defines a rotational axis [°].
	Depending on the mode of the axis several parameters has to be adjusted (factor, incperrot, sfactor, defspeed, see below).	

incperrot	Converts degrees into steps for ANGLE mode.	Unit: steps/360°
	This value defines how many increments are needed for a whole rotation. 'factor' and 'sfactor' have to be equal to (1/360)*'incperrot'. This parameter is only used in ANGLE mode.	

factor	Converts mm (or °) into steps.	
	POSITION mode, unit: steps/ mm	ANGLE mode, unit: steps/°
		Value has to be equal to **(1/360)*'incperrot'**
	This value defines how many increments are needed for 1mm (or 1°). Depending on the factor of the axis several parameters has to be adjusted (incperrot, sfactor, llimit, hlimit, hslimit, accel, decel, refspeed, refspeed2, refpos see below).	

sfactor	Sets speed factor.	
	POSITION mode, unit: steps/ mm	ANGLE mode, unit: steps/°
	Value has to be equal to **'factor'** for POSITION and ANGLE mode.	

llimit	Sets the lower limit.	Unit: steps
	No movement below this limit will be possible. Minimum value for llimit is '-1E13'. To get the 'llimit' in steps like it is required here the value in mm (or °) has to be multiplied by the 'factor'.	

hlimit	Sets the upper limit.	Unit: steps
	No movement above this limit will be possible. Maximum value for hlimit is '1E13'. To get the 'hlimit' in steps like it is required here the value in mm (or °) has to be multiplied by the 'factor'.	

hslimit	Sets the upper speed limit.	Unit: steps/s
	No speed above this speed limit will be possible. To get the 'hslimit' in steps/s like it is required here the value in mm/s (or °/s) has to be multiplied by the 'factor'.	

defspeed	Sets default speed displayed in SAMLight user interface.	
	POSITION mode, unit: mm/s	ANGLE mode, unit: °/s
	This value has to be smaller or equal than 'hslimit'/'sfactor'.	

accel	Sets increase of speed (in steps per second) per step.	Unit: steps/s per step
	A value of '-1' or '0' disables acceleration. In this case the step frequency corresponds to the applied speed in SAMLight. This is not recommended because of possible hardware damage and miscalibration. Please consider the unit which leads to an exponential acceleration.	

decel	Sets decrease of speed (in steps per second) per step.	Unit: steps/s per step
	A value of '-1' or '0' disables deceleration. In this case the step frequency just stops when the in SAMLight applied position has been reached. This is not recommended because of possible hardware damage and miscalibration. Please use only positive values for this parameter. Please consider the unit which leads to an exponential deceleration.	

accStartSpeed	Sets start speed.	
	POSITION mode, unit: mm/s	ANGLE mode, unit: °/s
	This value has to be positive and smaller or equal than 'hslimit'/'factor'. If the in SAMLight demanded speed is smaller than the value for this parameter the axis will move with the lower speed without acceleration.	

decStopSpeed	Sets stop speed.	
	POSITION mode, unit: mm/s	ANGLE mode, unit: °/s
	This value has to be positive and smaller or equal than 'hslimit'/'factor'. If the in SAMLight demanded speed is smaller than the value for this parameter the axis will move with the lower speed without deceleration.	

stepIO	Sets output bit of the scanner controller card for the step signal.
	Value: refer to Stepper I/O parameters

dirIO	Sets output bit of the scanner controller card for the direction signal.
	Value: refer to Stepper I/O parameters
	Since all axes move successively the 'dirIO' bit can be the same for all axes. Although the parameter 'dirIO' has the same value for multiple axes it still has to be defined for every axis.

dirvalue	Sets the polarity of the 'dirIO' signal for positive movement.	
	Value	Function
	0	low active
	1	high active

DelayAfterDir	Sets a delay which is executed after change of the direction.
	This command is used to specify a delay in us to be waited between a change in direction and the first step in the new direction.

refIO	Sets input bit of the scanner controller card for the reference signal.	
	Value	Function
	Stepper I/O parameters	Default homing, 'refmode=(1..6)': In type 8 reference movements are software controlled and (due to the limitations of the pc) the maximum speed is much lower and the signal is noisy (jitter).
	1 (ext. stop)	Fast homing (USC only): If this value is used together with 'refvalue=1' and 'refmode=(1 or 4)' the movement to the reference switch is performed with normal speed. This value is recommended only if you use just one axis.
	20	No homing: Use this value and 'refmode=0' if no reference switch is used.

refvalue	Sets the polarity of the 'refIO' signal.	
	Value	Function
	0	low active
	1	high active

refmode	Sets the behavior of homing movement.	
	Value	Function
	Common reference modes:	
	0	No homing movement, current position is set to 'refpos'
	1	Go to switch in neg. dir. and leave it in pos. dir.
	4	Go to switch in pos. dir. and leave it in neg. dir.
	Uncommon reference modes:	
	2	Go to switch in neg. dir. and leave it in neg. dir.
	3	Go to switch in neg. dir. and stay there
	5	Go to switch in pos. dir. and leave it in pos. dir.
	6	Go to switch in pos. dir. and stay there

refspeed	Sets the speed of the homing movement.	Unit: steps/s
	This value defines how fast the motor moves in case of a homing movement to find the reference switch. Reference movements are software controlled and have due to limitations of the PC a much lower maximum speed. To get the 'refspeed' in steps/s like it is required here the value in mm/s (or °/s) has to be multiplied by the 'factor'.	

refspeed2	Sets the speed while leaving the reference switch.	Unit: steps/s
	If refspeed2 is '-1', 'refspeed'/4 is used. To get the 'refspeed2' in steps/s like it is required here the value in mm/s (or °/s) has to be multiplied by the 'factor'.	

refpos	Sets the home position of the axis.	Unit: steps
	This value will be set after SAMLight startup and after a homing movement. refpos should be outside the position limits 'llimit' and 'hlimit' to avoid that the reference switch is activated during normal movement. To get the 'refpos' in steps like it is required here the value in mm (or °) has to be multiplied by the 'factor'.	

Optional axis parameters (type 8): All of the following optional parameters can be defined for each axis. Each axis can be configured differently. If no parameter is specified the corresponding feature will not be used.

SignalAxisMoving	Sets an output bit during an axis movement.	
	Value	Function
	0	inactive
	1	active

SignalAxisMovingStateBitPosition	Sets output bit of the scanner controller card for the 'SignalAxisMoving' signal.
	Value: refer to Stepper I/O parameters

SignalAxisMovingState	Sets the polarity of the 'SignalAxisMoving' signal.	
	Value	Function
	0	low active
	1	high active

SignalAxisMovingStatePreDelay	Delays the start of the motion.	Unit: ms
	Use SignalAxisMovingStatePreDelay to add a delay before the start of the motion.	

MoveThisAxisFirst	Sets homing priority of this axis.	
	Value	Function
	0	disable homing priority
	1	enable homing priority
	This parameter can only be used for one axis. If it is defined for more than one axis the parameter will be ignored for all axis.	

corr1 0.0 0.0 **corr2 500.0 100000.0** **corr3 1000.0 200000.0** **corr4 1500.0 300000.0** **corr5 2000.0 400000.0** **<corr#> <mm#> <steps#>**	Correction table instead of parameter 'factor' for POSITION mode.
	This parameter is defined without '=' sign in the form <corr#> <mm> <steps>.
	This correction table can be used to compensate nonlinearities of a straight axis. If 'factor' is defined, the correction table is ignored. Use for corr1 the same step value as for 'llimit' and for corr5 the same step value as for 'hlimit'.

Example sc_motion_stepper_settings.txt file for 1 straight and 1 rotational axis for 'Type=8':

```
# Global parameters:
Debug=0
DisableHomingDuringStartUp=1

# Z-axis parameters:
axis=0
dname=Z
mode=POSITION

factor=200.0
sfactor=200.0
llimit=-400000
hlimit=400000
hslimit=6000
defspeed=20.0
accel=50
decel=50
accStartSpeed=5.0
decStopSpeed=5.0

stepIO=1
dirIO=3
dirvalue=0

refIO=2
refvalue=1
refmode=0
refspeed=2000.0
refspeed2=500.0
refpos=0.0

# R-axis parameters:
axis=1
dname=R
mode=ANGLE

factor=10.0
incperrot=3600
sfactor=10.0
llimit=-1E13
hlimit=1E13
hslimit=6000
defspeed=400.0
accel=50
decel=50
accStartSpeed=5.0
decStopSpeed=5.0

stepIO=4
dirIO=5
dirvalue=0

refIO=3
refvalue=1
refmode=0
refspeed=2000.0
refspeed2=500.0
refpos=0.0
```

4.3.2.1 Motion Settings Dialog Type 8

In Settings → System → Extras there is a button "Motion settings dialog". If you press it you will access the settings dialog GUI for the stepper motor type 8:

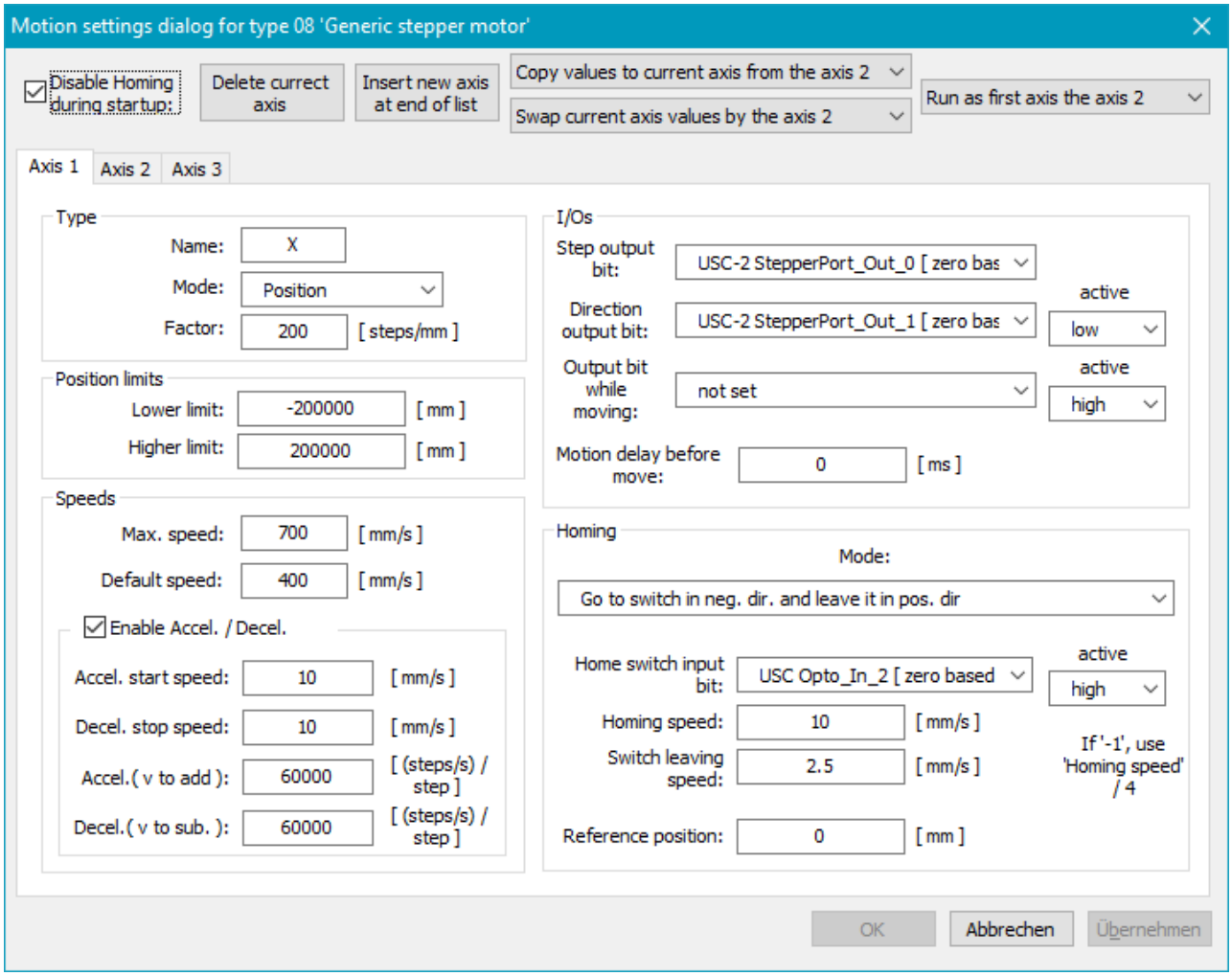


Figure 30: Stepper GUI for motor type 8

Here you can configure the motor like described in the file <SCAPS>\system \sc_motion_stepper_settings.txt.

Type:

Name: Sets name of axis. Has to be an unique single capital letter for each axis.

Mode: There are two valid axis modes:

'POSITION' - straight axis in [mm]

'ANGLE' - rotational axis in [°]

Factor:

For 'mode=POSITION': Unit: [steps/mm]

For 'mode=ANGLE' : Unit: [steps/ °], value has to be (1/360°)*'incperrot'

Converts mm (or °) used in SAMLight into steps.

Position limits: in [steps]

These two values define the lower and upper limit of the axis.

These values are read as double to increase the value range: [-1E300, 1E300]

The limits should not contain 'refpos'.

Speeds:

Default speed:

For 'mode=POSITION': Unit: [mm/sec]

For 'mode=ANGLE' : Unit: [°/sec]

Sets default speed displayed in SAMLight user interface.

Accel. start speed / Decel. stop speed: Unit: [mm/sec]

These two values define the start and stop speed of the motor.

Accel. / Decel.:

Unit: increase of [steps/second] per step, def. value: 50. Please consider the unit which leads to an exponentially acceleration.

A value of '0' or '-1' disables acceleration /deceleration.

I/Os:

Step output bit: The output bit for the step signal can be chosen here.

Direction output bit: The output bit for the direction signal can be chosen here.

Motion delay before move: Unit: [msec]. Use this parameter to delay the start of the motion.

Homing:

Mode: This mode defines the behavior of homing.

4.3.3 Stepper I/O parameters

Here you can find all available I/O parameters and the corresponding pin assignment for the motion type 14 (USC-2 stepper controller) and type 8 (generic stepper controller) for the sc_motion_stepper_settings.txt file.

USC cards:

Input pin	Stepper settings param	Output pin	Stepper settings param
Opto-insulated Inputs		**Opto-insulated Outputs**	
OptoIn_0	Reserved for trigger start	OptoOut_0	Reserved for marking active
OptoIn_1	Reserved for external stop	OptoOut_1	1
OptoIn_2	2	OptoOut_2	2, if not used for red pointer
OptoIn_3	3	OptoOut_3	3
OptoIn_4	4	OptoOut_4	4
OptoIn_5	5	OptoOut_5	5
		Laser Ports, if not used for laser control	
		LP_0	100
		LP_1	101
		LP_2	102
		LP_3	103

Input pin	Stepper settings param	Output pin	Stepper settings param
		LP_4	104
		LP_5	105
		LP_6	106
		LP_7	107
Following pins are only available for USC-2			
Digital Inputs		**Digital Outputs**	
DigiIn_0	6	DigiOut_0	206
DigiIn_1	7	DigiOut_1	207
DigiIn_2	8	DigiOut_2	208
DigiIn_3	9	DigiOut_3	209
DigiIn_4	10	DigiOut_4	210
DigiIn_5	11	DigiOut_5	211
DigiIn_6	12	DigiOut_6	212
DigiIn_7	13	DigiOut_7	213
DigiIn_8	14	DigiOut_8	214
DigiIn_9	15	DigiOut_9	215
Stepper inputs		**Stepper Outputs**	
SmIn_0	16	SmOut_0	216
SmIn_1	17	SmOut_1	217
SmIn_2	18	SmOut_2	218
		SmOut_3	219
		SmOut_4	220
		SmOut_5	221

Table 4: Stepper I/O parameters for USC cards

RTC cards:

Input pin	Stepper settings param	Output pin	Stepper settings param
Digital Inputs		**Digital Outputs**	
DigiIn_0	0	DigiOut_0	0
DigiIn_1	1	DigiOut_1	1
DigiIn_2	2	DigiOut_2	2
DigiIn_3	3	DigiOut_3	3
DigiIn_4	4	DigiOut_4	4
DigiIn_5	5	DigiOut_5	5
DigiIn_6	6	DigiOut_6	6
DigiIn_7	7	DigiOut_7	7
DigiIn_8	8	DigiOut_8	8
DigiIn_9	9	DigiOut_9	9
DigiIn_10	10	DigiOut_10	10
DigiIn_11	11	DigiOut_11	11
DigiIn_12	12	DigiOut_12	12
DigiIn_13	13	DigiOut_13	13
DigiIn_14	14	DigiOut_14	14
DigiIn_15	15	DigiOut_15	15

Table 5: Stepper I/O parameters for RTC cards

4.4 Other motion controller

The following motion controllers (RS-232, CAN, etc.) are supported.

4.4.1 Type 1 - IMS Stepper Drives

After leaving the program an *ims_settings.txt* file is created in the folder <*SCAPS*>*system* where controller settings can be done in case it is configured for the Ims motion controller.

For example:

ComPort = 1	**serial interface number, alternatively: "ComPort = USC1" (CH_O)**
ComSettings = 9600,N,8,1	Portname := baud rate, parity, word length, stop bits, flow control
PartyMode = 1	drive controller in party mode (for multiple axis) → With the current version the party mode needs to be used.
EncoderMode = 0	Read back encoder values (0/1)

For each axis the following settings need to be done.

AxisName: Definition of the name of the axis (one letter) where the commands are being sent to.

AxisMode: Either angle or pos. Sets the units in the user interface.

AxisScale: Number of increments per unit. Taken for pos mode.

AxisIncPerRot: Number of increments for one rotation, relevant for angle mode.

EncoderAxis: Defines if the motor has an encoder, values: *0* or *1*.

SpeedScale: Factor for speed. In case the motor drives another wheel this factor is needed for the wheel to achieve the entered speed. The default value is *1*.

MinSpeed: Minimum Speed in steps per second.

MaxSpeed: Maximum Speed in steps per second.

DefaultSpeed: Default Speed in steps per second.

Pressing the Home button in the control motion property page the "G 0" command is sent to comport.
If Auto Variable Resolution mode is used the speed will be the entered speed divided with the resolution factor.

4.4.2 Type 4 - External custom controller

SAMLight supports an interface for a customized implementation of a motion controller *.dll. A sample package is available on our homepage (http:\\www.scaps.com) or on request. The name of the external motion controller is sc_motion_control_ext_<DN>.dll where <DN> corresponds to the defined controller name in sc_motion_settings.txt. The *.dll has to be located in the same directory as sam_light.exe or in C:\Windows\System32\

Motion controller type: The external custom controller and the device name needs to be defined in the text file **sc_motion_settings.txt**. This file can be found in the folder <SCAPS>\system\.

Type=4 Sets the type of the motion controller to the external custom controller.

DeviceName=<DN> Specifies the name of the device, which determines the name of the *.dll that is loaded during runtime to access the motion controller functionality (see above).

4.4.3 Type 5 - IMS MDrive

Key features

- Available for USC and RTC controller cards
- Up to 7 axes can be controlled
- Axes operate in party mode and move at the same time (without synchronization)
- The MDrive motor is controlled via a serial port, e.g. COM port of the PC or the RS-232 interface of an USC controller card
- Homing procedures available

Required settings

- Before SAMLight can communicate and control the MDrive motor you need to define some parameters directly at your MDrive motor (e.g. via IMS Terminal or Windows HyperTerminal, refer to 'Hardware settings (type 5)' described below).
- Control all MDrives in IMS Terminal or Windows HyperTerminal before you try to control them by SAMLight to make sure the settings of the MDrives are correctly. Use the same settings for the HyperTerminal as for SAMLight ('PortName', 'PortParity' and 'PortBaudRate').
- Set 'Type=5' in sc_motion_settings.txt in <*SCAPS*>\system\
- Define sc_motion_mdrive_settings.txt in <*SCAPS*>\system\ like described below.
- Enable 'Motion Control' in SAMLight (Settings→System→Extras).

How to create the settings file

- The file 'sc_motion_mdrive_settings.txt' is a plain text file that contains different configuration parameters.
- A line which begins with a # sign is interpreted as a comment and will be ignored.
- All parameters have to start exactly at the beginning of a line.
- Use integers for every parameter value, decimal points will be ignored.
- Global parameters have to be specified only once at the head of the settings file.
- Axis-specific parameters have to be defined for all axes separately.
- The axis-specific and the optional parameter have to be arranged in one block of parameters per axis. Each block begins with the parameter 'axis'.
- The optional parameters do not have to be specified in the case you do not want to use them.
- A parameter gets defined by typing <parameter>=<value>, parameter and values are described below,
  e.g. 'axis=0' defines the parameter 'axis' with the value '0'

Hardware settings (type 5): These settings have to be defined directly at every MDrive motor.

Connect only one motor for the hardware settings to the COM port and repeat this steps for each MDrive motor.

Necessary steps for every motor before SAMLight can control MDrive motors.	
Before SAMLight can communicate and control the MDrive motor you need to define the following parameters directly at your MDrive motor (e.g. via IMS Terminal or Windows HyperTerminal). The actual MDrive commands may vary a bit compared to the commands you find here. For further information see the MDrive manual.	
Suggested MDrive command	Function
PR AL	Prints all MDrive values: Echoes all motor parameters. While party mode is disabled no prefix '?' is used.
DN=?	Defines device name: '?' has to be a single capital character. Use a unique value '?' for each MDrive motor. E.g. 'DN=X' (if party mode is disabled)
PY=1	Enables party mode: Party mode is necessary to address the commands from SAMLight to the right MDrive motor, even you want to use only one motor.
If the MDrive is in party mode you need to specify every command with the prefix '?' which corresponds to the device name. This is necessary to address commands to the right device because all motors communicate through a single COM port.	
?IP	Initializes parameters for the '?'-axis: Discards all temporary changes and applies the stored values of the MDrive motor.
?EM=2	Disables Echo mode for the '?'-axis: Suppresses echoing of almost all MDrive commands.
?DE=1	Enable the motor driver for the '?'-axis.
?S	Saves settings for the '?'-axis: E.g. 'XS' (if party mode is enabled) E.g. 'S' (if party mode is disabled)
?PR AL	Prints all MDrive values for the '?'-axis: Echos all parameters stored directly at the motor.

Global parameters (type 5): These parameters have to be defined once at the head of the settings file.

PortName	Defines the serial port used for the MDrive motors.	
	Value	Function
	COM#	Uses a COM port of the PC where '#' is the port number (e.g. 'COM1').
	USC-1	Uses the RS-232 interface of the USC card at the 37-pin connector (CH_0). For further information please refer to the corresponding USC manual.
	USC-2	

PortParity	Defines the parity mode for the COM port.	
	Value	Function
	0	No parity
	1	Odd parity
	2	Even parity
	3	Mark parity
	4	Space parity
	The default value that is used by most IMS MDrives is '0'.	

PortBaudRate	Defines the data rate for the COM port.	Unit: Bd (bit/s)
	The default value that is used by most IMS MDrives is '9600'.	

TimeOut	Sets the connection timeout.	Unit: s
	Sets how long it shall tries to access the communication between interface and the motion controller. If you have connection problems the debug mode (see below) could be helpful.	

Debug	Enables the debug log.	
	Value	Function
	0	Disable debug mode
	1	Enable debug mode
	Enables debug mode to log debug information in <*SCAPS*> \system\sc_motion_mdrive_debuglog.txt'. Only enable this option if you need it, because there can be huge amounts of data that are logged into this file.	

DisableHomingDuringStartUp	No homing on startup.	
	Value	Function
	0	Enable homing on startup of SAMLight
	1	Disable homing on startup of SAMLight
	If set to '1', homing is just performed when pressing 'Control→Home' and not when software is started.	

RtsControl	Enables or disables RTC (Request to Send) for the RS232 connection for MDrive's RS232-USB-adapter if used.	
	Value	Function
	0	set RTS_CONTROL_DISABLE
	1	set RTS_CONTROL_ENABLE
	If this parameter is not set the internal default state '1' is used. This parameter is available starting with installer 3.7.5 Build 0033.	

CheckForProgramRunning	Enables or disables checking for a MDrive subroutine after homing.	
	Value	Function
	0	Disable CheckForProgramRunning
	1	Enable CheckForProgramRunning
	If this parameter is not set the internal default state is '1'. In case after homing an own subroutine is used that returns a '1' for 'PR BY' (Print Busy), CheckForProgramRunning should be set to '0'. This parameter is available starting with installer 3.7.5 Build 0035.	

Axis parameters (type 5): All of the following parameters has to be defined for each axis after the global parameters in the settings file. Each of these axes can be configured differently.

axis	Sets the number of axis.	
	Value	Function
	0...6	Index of axis
	Zero based index of up to 7 axes. Defines the display order in SAMLight. This parameter has to be at the beginning of each axis specific parameter block.	

dname	Sets the name of axis.	
	Value	Function
	A...Z	Name of axis
	Has to correspond to the device name 'DN' which is stored directly at the MDrive motor. (Refer to 'Hardware settings (type 5)'). Has to be an unique single character for each MDrive motor.	

mode	Sets the type of the axis.	
	Value	Function
	POSITION	Defines a straight axis [mm].
	ANGLE	Defines a rotational axis [°].
	Depending on the mode of the axis several parameters has to be adjusted (corr#, incperrot, sfactor, llimit, hlimit, defspeed, see below).	

corr1 -1000 -20000000 **corr2 -500 -10000000** **corr3 0 0** **corr4 500 10000000** **corr5 1000 20000000** **<corr#> <mm#> <steps#>**	Correction table converts mm into steps and defines the range of the axis for POSITION mode.
	These five parameters are defined without '=' sign in the form <corr#> <mm#> <steps#>
	Setup for linear corr table: The range of the axis in mm corresponds to the difference of the max. value <mm5> and min. value <mm1>. The other mm-values have to be linearized. The step-values can be calculated by: **<steps#> = <mm#> * factor** The factor can be determined experimentally if not known. Non-linear corrections: This table can be used to compensate non-linearities of a straight axis by adjusting the values. For ANGLE mode these parameters are ignored, the parameter 'incperrot' is used instead.

incperrot	Converts degrees into steps for ANGLE mode.	Unit: steps/360°
	This value defines how many increments are needed for a whole rotation. The default value is 51200 and depends on the MDrive setup 'MS' (microsteps). It can be calculated by **200 * 'MS'**. For POSITION mode this parameter is ignored, the <corr#> table is used instead.	

sfactor	Sets speed factor.	
	POSITION mode, unit: steps/mm	ANGLE mode, unit: steps/°
	Value has to be equal to the factor of the corr table for a linear corr table.	Value has to be equal to **(1/360)*'incperrot'**.
	Converts mm/s (or °/s) used in SAMLight into steps/s.	

llimit	Sets the lower limit for the axis.	Unit: steps
	No movement below this limit will be possible. This value is used for POSITION and ANGLE mode and has to be equal to the **<steps1>** value of the corr table.	

hlimit	Sets the upper limit for the axis.	Unit: steps
	No movement above this limit will be possible. This value is used for POSITION and ANGLE mode and has to be equal to the **<steps5>** value of the corr table.	

hslimit	Sets the upper speed limit.	Unit: steps/s
	No speed above this speed limit will be possible. To get the 'hslimit' in steps/s like it is required here the value in mm/s (or °/s) has to be multiplied by the 'sfactor'.	

defspeed	Sets default speed displayed in SAMLight user interface.	
	POSITION mode, unit: mm/s	ANGLE mode, unit: °/s
	This value has to be smaller or equal than 'hslimit'/'sfactor'.	

Optional axis parameters (type 5): All of the following optional parameters can be defined for each axis. Each axis can be configured differently. If no parameter is specified the corresponding feature will not be used.

invertDir	Defines the motion direction.	
	Value	Function
	0	Default rotation
	1	Inverted rotation

ival	Sets initialization parameters.
	On startup of SAMLight every axis can be initialized with this ival parameter. This parameter can exist more than once and will be executed in the same order as they appear in the settings file. Each parameter corresponds to any MDrive command the drive supports.
	Recommended initialization parameters (refer to your MDrive manual for further information): ival=IP ival=EM 2 ival=DE

<table>
<tr><td rowspan="4">hval</td><td colspan="2">Sets homing parameters.</td></tr>
<tr><td colspan="2">With hval parameters MDrive commands a homing procedure for each axis can be defined. This parameter can exist more than once and will be executed in the same order as they appear in the settings file. The homing procedure will be executed on startup of SAMLight and by a manual homing. The homing procedure at startup of SAMLight can be disabled with global parameter 'DisableHomingDuringStartUp'.</td></tr>
<tr><td colspan="2">Suggested Mdrive commands for programming a homing procedure (for further information refer to MDrive manual):
'PG', 'HI', 'HM', 'P', 'S1'</td></tr>
<tr><td>Example 1:

No homing movement

Current position is set to <position> in steps.

hval=P <position></td><td>Example 2:

Referencing using a home switch

I/O number one is used for the home switch (low active); the drive is moving into negative direction with <speed> in steps/s until that input goes to low triggered by a home switch. fterwards the current position is set to <position> in steps.

hval=PG 100
hval=VM <speed>
hval=S1 1,0
hval=HM 1
hval=H
hval=P <position>
hval=E
hval=PG
hval=EX 100</td></tr>
</table>

Example sc_motion_mdrive_settings.txt file for 1 straight and 1 rotational axis for 'Type=5':

```
# Global parameters:
PortName=COM1
PortParity=0
PortBaudRate=9600
TimeOut=10
Debug=0

# X-axis parameters:
axis=0
dname=X
mode=POSITION
corr1 -1000 -20000000
corr2  -500 -10000000
corr3     0         0
corr4   500  10000000
corr5  1000  20000000
sfactor=20000
llimit=-20000000
hlimit=20000000
hslimit=500000
defspeed=25
invertDir=0

ival=IP
ival=EM 2
ival=DE 1
```

```
hval=P 0

# R-axis parameters:
axis=1
dname=R
mode=ANGLE
incperrot=51200
sfactor=142
hslimit=500000
defspeed=250
invertDir=0

ival=IP
ival=EM 2
ival=DE 1

hval=P 0
```

4.4.4 Type 6 - Faulhaber motion controller

The Faulhaber motion controller series are supported directly. To enable it, please configure the Faulhaber type 6 like it was described above.

For configuring the Faulhaber controller interface, a configuration file *sc_motion_faulmc_settings.txt* is required in the same location where the general motion settings file exists. There can be used one motion controller of this type at the same time, it is configured for the Z-axis fixed and can be used to perform positional changes. The Faulhaber configuration file can be used to specify different parameters like the COM-port, the data rate and others more. The file itself is a plain textfile that contains different statements. Additionally there can be comments within the file that begin with a # sign directly at the beginning of a line. These comments are ignored completely. Following parameters, that have to start exactly at the beginning of a line, are supported for configuring the interface.

PortName=xxx : Specifies to which port the Faulhaber controller is connected to, here for *"xxx"* e.g. COM1 has to be set.

PortBaudRate=yyyy : Defines the data rate (in bps) the COM port has to work with.

corr1 mm inc
corr2 mm inc
corr3 mm inc
corr4 mm inc
corr5 mm inc

The *corr* commands define a conversion and correction table from metric positions to incremental positions of the used Z-axis. This table consists of 5 entries where *corr1* defines the smallest possible value and *corr5* the biggest one. Here more than only a factor is given to allow it a user to equalize non-linear variances. The syntax of the *corr* table commands requires a metric value *mm* in millimeters and the appropriate incremental value *inc* that is equal to this metric position.

The incremental value of corr1 must be smaller or equal than the parameter llimit and the incremental value of corr5 must be equal or bigger than the value of parameter hvalue that are described below.

llimit=yyy : This parameter defines the lower limit the motion controller can drive to with the specified axis (in unit increments). Independent from the values that are sent from the program, the controller will never be driven to a value that is smaller than the one set here.

hlimit=yyy : This parameter defines the higher limit the motion controller can drive to using the current axis (in unit increments). Independent from the values that are sent from the program, the controller will never be driven to a value that is bigger than the one set here. So with *llimit* and *hlimit* a range can be defined where the motion controller is allowed to work within.

4.4.5 Type 7 - isel IT Stepper Controller / DNC

Key features

- Available for USC and RTC controller cards
- Up to 3 axes can be controlled
- Axes can move synchronized
- The motion controller is connected via a COM port of the PC
- Homing procedures are available

Required settings

- Set 'Type=7' in sc_motion_settings.txt in <*SCAPS*>\system\
- Define sc_motion_iselit_settings.txt in <*SCAPS*>\system\ like described below.
- Enable 'Motion Control' in SAMLight (Settings→System→Extras).

How to create the settings file

- The file sc_motion_iselit_settings.txt' is a plain text file that contains different configuration parameters.
- A line which begins with a # sign is interpreted as a comment and will be ignored.
- All parameters have to start exactly at the beginning of a line.
- Global parameters have to be specified only once at the head of the settings file.
- Axis-specific parameters have to be defined for all axes separately.
- The axis-specific and the optional parameters have to be arranged in one block of parameters per axis. Each block begins with the parameter 'axis'.
- The optional parameters do not have to be specified in the case you do not want to use them.
- A parameter gets defined by typing <parameter>=<value>, parameter and values are described below,
  e.g. 'axis=0' defines the parameter 'axis' with the value '0'.

Global parameters (type 7): These parameters have to be defined once at the head of the settings file.

PortName	Defines the serial port used for the isel it controller.	
	Value	Function
	COM#	Defines the COM port of the PC where '#' is the port number (e.g. 'COM1'). The USC RS232 port cannot be used for isel it controllers.

PortBaudRate	Defines the data rate for the COM port.	Unit: Bd (bit/s)
	The default value that is used by most isel controllers is '9600'.	

TimeOut	Sets the connection timeout.	Unit: s
	Sets how long it shall tries to access the communication between interface and the motion controller. If you have connection problems the debug mode (see below) could be helpful.	

DeviceNumber	Defines the serial port used for the isel it controller.	
	Value	Function
	#	The device number specifies which specific controller has to be accessed. At the controller hardware that number can be changed with the DNC command @<DN>G<DNnew>. The default value is '0'.

Debug	Enables the debug log.	
	Value	Function
	0	Disable debug mode
	1	Enable debug mode
	Enables debug mode to log debug information in <*SCAPS*> \system\sc_motion_iselit_debuglog.txt'. Only enable this option if you need it, because there can be huge amounts of data that are logged into this file.	

DisableHomingDuringStartUp	No homing on startup.	
	Value	Function
	0	Enable homing on startup of SAMLight
	1	Disable homing on startup of SAMLight
	If set to '1', homing is just performed when pressing 'Control→Home' and not when software is started.	

switch1	Enables the home switch for 'axis=0' for older motion controller	
	Value	Function
	0	Disable home switch
	1	Enable home switch
	This parameter is only valid for older motion controller, which support the "IE" command. Set the value to '0' for a newer motion controller and define the homing procedure with 'hval' commands.	

switch2	Enables the home switch for 'axis=1' for older motion controller	
	Value	Function
	0	Disable home switch
	1	Enable home switch
	This parameter is only valid for older motion controller, which support the "IE" command. Set the value to '0' for a newer motion controller and define the homing procedure with 'hval' commands.	

level1	Sets the polarity of the home switch for 'axis=0'	
	Value	Function
	0	low active
	1	high active
	This parameter is only valid for older motion controller, which support the "IE" command.	

level2	Sets the polarity of the home switch for 'axis=1'	
	Value	Function
	0	low active
	1	high active
	This parameter is only valid for older motion controller, which support the "IE" command.	

Axis parameters (type 7): All of the following parameters have to be defined for each axis after the global parameters in the settings file. Each of these axes can be configured differently.

axis	Sets the number of axis.	
	Value	Function
	0...2	Index of axis
	Zero based index of up to 3 axes. Defines the display order in SAMLight. This parameter has to be at the beginning of each axis specific parameter block. The axes are named x, y and z automatically. Start with index '0' for the first axis.	

mode	Sets the type of the axis.	
	Value	Function
	POSITION	Defines a straight axis [mm].
	ANGLE	Defines a rotational axis [°].
	Depending on the mode of the axis several parameters has to be adjusted (factor, corr#, incperrot, sfactor, llimit, hlimit, defspeed, see below).	

factor	Converts mm into steps for POSITION mode.	Unit: steps/mm
	This value defines how many increments are needed for 1mm. Depending on the factor of the axis several parameters has to be adjusted (sfactor, defspeed, see below). This parameter is only valid for SAMLight versions >= 3_3_5_0224. With older SAMLight versions the corr table has to be used.	

incperrot	Converts degrees into steps for ANGLE mode.	Unit: steps/360°
	This value defines how many increments are needed for a whole rotation. Depending on the factor of the axis several parameters has to be adjusted (sfactor, defspeed, see below).	

sfactor	Sets speed factor.	
	POSITION mode, unit: steps/ mm	ANGLE mode, unit: steps/°
	Value has to be equal to **'factor'**.	Value has to be equal to **'incperrot/360'**.
	Converts mm/s (or °/s) used in SAMLight into steps/s.	

llimit	Sets the lower position limit.	Unit: steps
	No movement below this limit will be possible. Minimum value is '-8388606'.	

hlimit	Sets the upper position limit .	Unit: steps
	No movement above this limit will be possible. Maximum value is '8388607'.	

hslimit	Sets the upper speed limit.	Unit: steps/s
	No speed above this speed limit will be possible. To get the 'hslimit' in steps/s like it is required here the value in mm/s (or °/s) has to be multiplied by the 'sfactor'.	

defspeed	Sets default speed displayed in SAMLight user interface.	
	POSITION mode, unit: mm/s	ANGLE mode, unit: °/s
	This value has to be smaller or equal than 'hslimit/sfactor'.	

Optional axis parameters (type 7): All of the following optional parameters can be defined for each axis. Each axis can be configured differently. If no parameter is specified the corresponding feature will not be used.

`corr1 -8000.0 -400000.0`
`corr2 -4000.0 -200000.0`
`corr3 0.0 0.0`
`corr4 4000.0 200000.0`
`corr5 8000.0 400000.0`
 	
<corr#> <mm#> <steps#>
	Correction table instead of parameter 'factor' for POSITION mode.

	This parameter is defined without '=' sign in the form <corr#> <mm> <steps>.
	This correction table can be used to compensate nonlinearities of a straight axis. If 'factor' is defined, the correction table is ignored. Use for corr1 the same step value as for 'llimit' and for corr5 the same step value as for 'hlimit'.

<table>
<tr><td rowspan="4">hval</td><td colspan="2">Sets homing parameters.</td></tr>
<tr><td colspan="2">With hval parameters a homing procedure for each axis can be defined. This parameter can exist more than once and will be executed in the same order as they appear in the settings file. The homing procedure will be executed on startup of SAMLight and by a manual homing.</td></tr>
<tr><td colspan="2">Refer to the isel manual for the commands.</td></tr>
<tr><td>Example 1, no homing, set current position of 3 axes to 0:

hval=n7</td><td>Example 2, homing of 3 axes:

hval=d<xSpeed>, <ySpeed>, <zSpeed>
hval=R7</td></tr>
</table>

Example sc_motion_iselit_settings.txt file for 1 straight and 1 rotational axis for 'Type=7':

```
# Global parameters:
PortName=COM1
PortBaudRate=9600
TimeOut=30
DeviceNumber=0
Debug=0
DisableHomingDuringStartUp=1
switch1=0
switch2=0
level1=0
level2=0

# x axis parameters:
axis=0
mode=POSITION
factor=50.0
sfactor=50.0
llimit=-8388606.0
hlimit=8388607.0
hslimit=10000
defspeed=100.0
hval=n1

# y axis parameters:
axis=1
mode=ANGLE
incperrot=18000
sfactor=50.0
llimit=-8388606.0
hlimit=8388607.0
hslimit=10000
defspeed=180.0
hval=n2
```

4.4.6 Type 9 - Generic RS232 interface

This special interface can be used to access controllers by sending plain strings to them via RS232. To enable it, please configure the Generic RS232 type 9 like it was described above. Controlling the connected device is done exclusively by using strings that are sent to it. There is no axis enabled within the motion control property pane. Carriage return and line feed characters (\r\n) are added to all strings automatically.

To configure the generic interface, a configuration file *sc_motion_generic232_settings.txt* is required in the same location where the general motion settings file exists. This configuration file can be used to specify different parameters like the COM port, the data rate, the initialization and others more. The file itself is a plain text file that contains different statements. Additionally there can be comments within the file that begin with a # sign directly at the beginning of a line. These comments are ignored completely. Following parameters, that have to start exactly at the beginning of a line, are supported for configuring the interface:

PortName=xxx : Specifies to which port the controller is connected to, here for *"xxx"* e.g. COM1 has to be set. To use the RS232 interface of the USC card (CH_0), set *PortName=USC-1*. All values from COM1 to COM9 are allowed. Higher than COM9 is not allowed.

PortBaudRate=yyyy : Defines the data rate (in bps) the com port has to work with.

PortByteSize=z : Sets the byte size in bits of the COM port.

PortParity=x : Specifies the parity mode for the port, following values are supported:
- 0 - No parity
- 1 - Odd parity
- 2 - Even parity
- 3 - Mark parity
- 4 - Space parity

PortStopbits=y : Defines the stopbits for the port. The following values are supported: 1, 1.5, 2.

SendEolValue=z : Sends an end of line value as a decimal character code. The following values are possible:
- 0 - do not send: Carriage return plus line feed is added to string
- 1 - send: The EolValue is added to string

EolValue=xxx: Sets the End of line value as a decimal character code. Values from *0* to *255* are supported. For example set *EolValue=13* for carriage return.

TimeOut=yy : This value specifies how long the interface shall retry to access the communication interface and the motion controller until it fails with a time out in case such an operation is not successful.

Debug=y : For special debugging purposes it is possible to log several information into a file *sc_motion_generic232_debuglog.txt* that will be created at the position where the settings file is located at. With the value *1* the debugging mode is turned on, when it is set to *0* debugging is disabled and no data are put into the log file.

There can be huge amounts of data that are logged into such a file so enable this option only if it is necessary.

ival=sssss : The initialization of the controller can be done using every command it supports. To specify which of these controller commands have to be used during the initialization of it, the *ival* parameter can be used. Different to the preceding parameters this one can exist more than once in a configuration file. Here a second *ival* will not overwrite the value of the preceding one. So for every controller command that has to be sent one single line starting with this parameter has to be set. The controller commands are executed in the same order as the *ival* parameters appear in the settings file. The commands *sssss* that are defined with this parameter are used for initialization and therefore sent as very first to the drive.

4.4.7 Type 10 - SHS 2000 Star

The SHS Star 2000 motion controller series is supported directly. To enable it, please configure the drive type number 10 like it was described above.

For configuring the SHS controller interface, a configuration file *sc_motion_shstar2000_settings.txt* is required in the same location where the general motion settings file exists. The SHS Star 2000 configuration file can be used to specify different parameters like the COM port, the data rate, the initialization and others more. The file itself is a plain text file that contains different statements. Additionally there can be comments within the file that begin with a # sign directly at the beginning of a line. These comments are ignored completely. Following parameters, that have to start exactly at the beginning of a line, are supported for configuring the interface:

PortName=xxx : Specifies to which port the SHS controller is connected to, here for *"xxx"* e.g. COM1 has to be set.

PortBaudRate=yyyy : Defines the data rate (in bps) the com port has to work with. The default value that is used by most SHS controllers is 19200.

PortParity=y : This parameter changes the parity mode for the port, here for *y* following values are supported:

- 0 - No parity
- 1 - Odd parity
- 2 - Even parity
- 3 - Mark parity
- 4 - Space parity

PortRTS=y : Using this parameter the ready to send behavior can be configured like it is necessary for some serial protocols. Here following values are possible for *y*:

- 0 - disable RTS fully
- 1 - enable standard RTS mode
- 2 - enable RTS handshake mode
- 3 - enable RTS in control toggle mode

TimeOut=yy : The *TimeOut* value specifies how long the interface shall retry to access the communication interface until it fails with a time out in case such an operation is not successful.

Debug=y : For special debugging purposes it is possible to log several information into a file *sc_motion_shstar2000_debuglog.txt* that will be created at the position where the settings file is located at. With the value *1* the debugging mode is turned on, when it is set to *0* debugging is disabled and no data are put into the log file.

There can be huge amounts of data that are logged into such a file so enable this option only if it is necessary.

All these values are global ones and therefore they are valid for everything that is controlled by that motion controller. This is important for the multi-axes-mode where it is possible to handle up to five axes using this controller interface. Each of these axes can be configured in a different way. To do that, the *axis* statement that defines for which *axis* the following parameters are valid has to be use. In the worst case when three axes have to be used with completely different configurations all the following parameters have to exist five times, separated by the different calls of *axis*:

axis=y : Specifies for which axis the following configuration parameters are valid for. If this statement is not used somewhere in a configuration file it is assumed that the third axis is used. Else a value out of the allowed range 0..4 can be set here. The axis number is also used as an address for the controller. It has to be configured with the same number so that the controller is able to access it.

mode=POSITION : The SHS Star 2000 motion controller supports two operational modes for an axis: ANGLE and POSITION that can be set using this statement. Depending on this mode values can be entered

in degrees or mm (inch/bits) only. A mixture between angular and planar movements is not possible for the same axis. Depending on that mode some of the following settings are ignored.

corr1 mm inc
corr2 mm inc
corr3 mm inc
corr4 mm inc
corr5 mm inc

The *corr* commands define a conversion and correction table from metric positions to incremental positions of the currently specified (or default) axis. This table consists of 5 entries where *corr1* defines the smallest possible value and *corr5* the biggest one. Here more than only a factor is given to allow it a user to equalize non-linear variances. The syntax of the *corr* table commands requires a metric value in millimeters and the appropriate incremental value that is equal to this metric position.

The incremental value of corr1 must be smaller or equal than the parameter llimit and the incremental value of corr5 must be equal or bigger than the value of parameter hvalue that are described below. These values are used for the mode POSITION.

llimit=yyy : This parameter defines the lower limit the motion controller can drive to with the specified axis (in unit increments). Independent from the values that are sent from the program, the controller will never be driven to a value that is smaller than the one set here. This value is used for the mode *POSITION*.

hlimit=yyy : This parameter defines the higher limit the motion controller can drive to using the current axis (in unit increments). Independent from the values that are sent from the program, the controller will never be driven to a value that is bigger than the one set here. So with *llimit* and *hlimit* a range can be defined where the motion controller is allowed to work within. This value is used for the mode *POSITION*.

incperrot=yyy : Comparing to the correction table that defines the conversion factors from increments to positions and back this value has to be used to define a factor that specifies the relation between the number of increments and the appropriate angle for an axis. With *incperrot* it has to be specified how many increments are equal to one complete rotation (360°). This value is used for the operation mode *ANGLE*.

sfactor=yyy : For the speed conversion from mm/sec to increments/sec or from degrees/sec to increments/sec the speed factor parameter "sfactor" is used. Here it has to be specified how many increments/sec are equal to a speed of one mm/sec or one degree/sec. The distinction between the unit mm/sec or degree/sec is made by the current operational mode, the first one is true for mode "POSITION", the second for "ANGLE".

hslimit=yyy : This parameter can be used to set a maximum speed in unit increments/second. If a speed value sent from the application is larger than the value *yyy* after conversion using the *sfactor*, the speed setting sent to the drive is limited to the value given with this parameter. The high speed limit parameter is used for both operation modes.

defspeed=yyy : Using this parameter only the behavior of the user interface is influenced, but not the functionality of the drive. Here a default speed value *yyy* can be set in unit mm/second that is displayed within the scanner application by default.

dname=c : When the motion controller is operated in party mode, that means with more than only one SHS at the same communication line, a single controller hardware needs to be accessed by using its device name. The parameter *dname* specifies such a name for the current axis. That name has to consist of a single char and it of course has to be configured and saved directly at the SHS motion controller hardware itself. Setting this name is necessary when more than one axis is configured using the *axis* command. Here for every axis a different name has to be specified. In case the *axis* command is not used and the controller operates using the default axis number two, the party mode for the motion controller interface and the hardware has to be set by sending the initialization command *PY 1* explicitly (please see the command *ival* below). Elsewhere the motion controller interfaces does not assume a party mode operation and ignores this name.

ival=sssss : The initialization of a single axis of the controller can be done using every command the drive supports. To specify, which of these controller commands have to be used during the initialization of it, the *ival* parameter can be used. Different to the preceding parameters this one can exist more than once in a configuration file. Here a second *ival* will not overwrite the value of the preceding one. So for every controller command that has to be sent one single line starting with this parameter has to be set. The controller commands are executed in the same order as the *ival* parameters appear in the settings file. The

commands *sssss* that are defined with this parameter are used for initialization and are therefore sent as very first to the drive. The commands that can be set here are equal to the ones that are sent to the drive. Since the SHS controllers work with binary data they have to be entered as single byte hexadecimal values separated by a space without the CRC. This checksum is added to every command byte sequence automatically. So for an example such an entry could look like this:

- ival=0xFC 0x20 0x01
- Here 0xFC is the start byte, 0x20 defines the axis number the command is valid for and the length of it (according to the specification of the SHS Star 2000 commands), 0x01 is the command itself. The missing CRC byte does not has to be set, it is added by the controller automatically for every command.

hval=sssss : Similar to the *ival* parameter *hval* can be used to define the operation during a movement to the home position for the current or the default axis. Here the operation that has to be performed to reach the home position has to be programmed using one or more *hval* parameters that are executed in the same order as they appear in the configuration file. The syntax of the byte value commands is the same like described above for the *ival* command.

The original *sc_motion_shstar2000_settings.txt* settings file that is delivered with this scanner software contains some default settings that have to be changed according to the target environment of the motion controller. Additionally there are some example initialization and homing sequences defined within this file for several scenarios. Here the desired parts of these examples simply have to be uncommented. Other default parameters may have to be removed or commented out. As it can be seen there for both, the *ival* and the *hval* parameter, complete programs can be defined that work within the motion controller to perform the initialization or the homing of it.

4.4.8 Type 11 - Jena Ecostep100

The *ECOSTEP 100* motion controller series is supported directly. To enable it, please configure the drive type number *11* like it was described above.

For configuring the *ECOSTEP* controller interface, a configuration file *sc_motion_jenaecostep100_settings.txt* is required in the same location where the general motion settings file exists. The *ECOSTEP* configuration file can be used to specify different parameters like the COM port, the data rate, the initialization and others more. The file itself is a plain text file that contains different statements. Additionally there can be comments within the file that begin with a # sign directly at the beginning of a line. These comments are ignored completely. Following parameters, that have to start exactly at the beginning of a line, are supported for configuring the interface:

PortName=xxx : Specifies to which port the *ECOSTEP* controller is connected to, here for *"xxx"* e.g. COM1 has to be set.

PortBaudRate=yyyy : Defines the data rate (in bps) the com port has to work with, the default and only value that is used by *ECOSTEP 100* controllers is *9600*.

PortParity=y : This parameter changes the parity mode for the port. Here for *y* the following values are supported:

- 0 - No parity
- 1 - Odd parity
- 2 - Even parity
- 3 - Mark parity
- 4 - Space parity

PortRTS=y : Using this parameter the ready to send behavior can be configured like it is necessary for some serial protocols. Here the following values are possible for *y*:

- 0 - disable RTS fully
- 1 - enable standard RTS mode
- 2 - enable RTS handshake mode
- 3 - enable RTS in control toggle mode

TimeOut=yy : This value specifies how long the interface shall retry to access the communication interface until it fails with a time out in case such an operation is not successful.

Debug=y : For special debugging purposes it is possible to log several information into a file *sc_motion_jenaecostep100_debuglog.txt* that will be created at the position where the settings file is located at. With the value *1* the debugging mode is turned on, when it is set to *0* debugging is disabled and no data are put into the log file.

There can be huge amounts of data that are logged into such a file so enable this option only if it is necessary. With (Debug=1) additional status request commands are send to the controller (and are reported in the log).

All these values are global ones and therefore they are valid for everything that is controlled by that motion controller. This is important for the multi-axes-mode where it is possible to handle up to five axes using this controller interface. Each of these axes can be configured in a different way. To do that, the *axis* statement that defines for which axis the following parameters are valid has to be used. In the worst case when three axes have to be used with completely different configurations all the following parameters have to exist five times, separated by the different calls of *axis*:

axis=y : Specifies for which axis the following configuration parameters are valid for. If this statement is not used somewhere in a configuration file it is assumed that the third axis is used. Else a value out of the allowed range *0..4* can be set here. The axis number is also used as address for the controller. It has to be configured with the same number so that the controller is able to access it.

mode=POSITION : The *ECOSTEP* motion controller supports two operational modes for an axis: *ANGLE* and *POSITION* that can be set using this statement. Depending on this mode values can be entered in

degrees or mm (inch/bits) only. A mixture between angular and planar movements is not possible for the same axis. Depending on that mode some of the following settings are ignored.

llimit=yyy : This parameter defines the lower limit the motion controller can drive to with the specified axis (in unit increments). Independent from the values that are sent from the program, the controller will never be driven to a value that is smaller than the one set here. This value is used for the mode *POSITION*.

hlimit=yyy : This parameter defines the higher limit the motion controller can drive to using the current axis (in unit increments). Independent from the values that are sent from the program, the controller will never be driven to a value that is bigger than the one set here. So with *llimit* and *hlimit* a range can be defined where the motion controller is allowed to work within. This value is used for the mode *POSITION*.

sfactor=yyy : For the speed conversion from mm/sec to increments/sec or from degrees/sec to increments/sec the speed factor parameter *sfactor* is used. Here it has to be specified how many increments/sec are equal to a speed of one mm/sec or one degree/sec. The distinction between the unit mm/sec or degree/sec is made by the current operational mode, the first one is true for mode *POSITION*, the second for *ANGLE*.

hslimit=yyy : This parameter can be used to set a maximum speed in unit increments/second. If a speed value sent from the application is larger than the value *yyy* after conversion using the *sfactor*, the speed setting sent to the drive is limited to the value given with this parameter. The high speed limit parameter is used for both operation modes.

defspeed=yyy : Using this parameter only the behavior of the user interface is influenced, but not the functionality of the drive. Here a default speed value *yyy* can be set in unit mm/second that is displayed within the scanner application by default.

incperrot=yyy : Comparing to the correction table that defines the conversion factors from increments to positions and back this value has to be used to define a factor that specifies the relation between the number of increments and the appropriate angle for an axis. With *incperrot* it has to be specified how many increments are equal to one complete rotation (360°). This value is used for the operation mode *ANGLE*.

DevnoEcoId=c : *Device No. / ECO Id*. represents the device id of the *ECOSTEP* host base station. Allowed values are: *1 .. 15*.

dname=c : Sets the character this device has to be displayed with.

ival=sssss : These parameters can contain sequences of *Jena ECOSTEP 100* commands as hex-numbers that perform operations on the drive during initialization. For a detailed description of these parameters please refer to the specifications of the controller.

The checksum and the deviceID do not have to be added to the command sequences. This is done automatically. An optional command identifier text (is logged in Debug mode), separated by a blank character, terminates the line.

E.g.:
The following three commands enable the drive to move (may be axis specific).
After the motion Controller default initialization no moving is possible
due to end position monitoring.

- ival=0x23 0x70 0x21 0x00 0x00 0x00 0x00 0x00 inputPolarityToDefault
- ival=0x23 0x71 0x21 0x02 0x00 0x00 0x00 0x00 pLockConfiguration
- ival=0x23 0x72 0x21 0x02 0x00 0x00 0x00 0x00 nLockConfiguration

hval=sssss : These parameters can contain sequences of *Jena ECOSTEP 100* commands as hex-numbers that perform operations on the drive during a homing sequence. For a detailed description of these parameters please refer to the specifications of the controller.

The checksum and the deviceID do not have to be added to the command sequences. This is done automatically.

E.g.:
homing to zero position (normal absolut positioning operation with default speed to zero position)
- hval=0x23 0x81 0x60 0x00 0x00 0x7E 0x04 0x00 setSpeed
- hval=0x23 0x7A 0x60 0x00 0x00 0x00 0x00 0x00 setPos
- hval=0x23 0x60 0x60 0x00 0x01 0x00 0x00 0x00 modeAbsolutPositioning
- hval=0x23 0x40 0x60 0x00 0x3F 0x00 0x00 0x00 executeRun

eval=sssss : These parameters can contain sequences of Jena EcoStep100 commands as hex-numbers that perform operations on the drive during a program exit. For a detailled description of these parameters please refer to the specifications of the controller.

The checksum and the deviceID do not have to be added to the command sequences. This is done automatically.

E.g.:
store all parameters permanent
eval=
- 0x23 - readWrite
- 0x10 - objLsb
- 0x10 - objMsb
- 0x01 - subIndex
- 0x73 - data0
- 0x61 - data1
- 0x76 - data2
- 0x65 - data3
- storeAllParameters - (for SAMLight Debug purposes)

ECOSTEP 100 / 200 Command format

```
  byte    id;
// id of ecostep100 station [ 1 .. 15 ]

  byte    readWrite;
// Send from PC to ECO controller:
// - 23 hex = Send command 4 Byte data ( Byte 4..7 contain an 32Bit-value )
// - 2B hex = Send command 2 Byte data ( Byte 4, 5 contain an 16Bit-value )
// - 2F hex = Send command 1 Byte data ( Byte 4 contains an 8Bit-value )
// - 40 hex = Read data object from slave
//
// Send from ECO controller to PC:
// - 43 hex = Command 40 successfully send, Byte 4..7 contain an 32Bit-value
// - 4B hex = Command 40 successfully send, Byte 4, 5 contain an 16Bit-value
// - 4F hex = Command 40 successfully send, Byte 4 contains an 8Bit-value
// - 60 hex = Commands 23 or 2B or 2F successfully send
// - 80 hex = Commands 23 or 2B or 2F or 40 not successfully send, Byte 4..7 contain error

  byte    objLsb;
// obj id: last significant Byte

  byte    objMsb;
// obj id: most significant Byte

  byte    subIndex;
// obj subIndex
```

```
  byte     data0;
// if controlword:
// Bit0 = Switch On
// Bit1 = Disable Voltage
// Bit2 = Quick Stop
// Bit3 = Enable Operation
// Bit4 = operating specific
// Bit5 = operating specific
// Bit6 = operating specific
// Bit7 =  Fault Reset
//
// if statusword:
// Bit0 = Ready to Switch on
// Bit1 = Switched On
// Bit2 = Operation Enable
// Bit3 = Fault
// Bit4 = Voltage Disabled
// Bit5 = Quick Stop
// Bit6 = Switch on Disable
// Bit7 = Warning

  byte     data1;
// if controlword:
// Bit0 = Halt
// Bit1 = reserved
// Bit2 = reserved
// Bit3 = manufacturer specific
// Bit4 = manufacturer specific
// Bit5 = manufacturer specific
// Bit6 = manufacturer specific
// Bit7 = manufacturer specific
//
// if statusword:
// Bit0 = manufacturer specific
// Bit1 = Reserved
// Bit2 = Setpoint reached
// Bit3 = Limit reached
// Bit4 = Setpoint confirmation / halt / found reference
// Bit5 = Contouring error / reserved / reference error
// Bit6 = Commutation found
// Bit7 =  Reference found

  byte     data2;
// data byte 2

  byte     data3;
// data byte 3

  byte     checksum;
// inverted sum( Byte 0 .. Byte 8 )

  string   szPacketName;
// for debug purposes
```

4.4.9 Type 12 - IO Switcher

With this controller special drive types can be used that start a movement after they received one single digital signal. That means using this driver it is not possible to stop software-controlled at an exact position. The driver only initiates the movement and the drive itself moves to a (predefined) position or by a (predefined) distance. When the drive has reached that position it gives back a signal to the software. This controller supports up to five motors. To enable this type of controller, please configure the IO switcher type 12 like it was described above.

For configuring the IO switcher motor controller interface itself, a configuration file *sc_motion_ioswitcher_settings.txt* is required in the same location where the general motion settings file exists. The IO switcher configuration file can be used to specify different parameters like the IOs for the different output and input signals and others. The file itself is a plain text file that contains different configuration parameters. Additionally there can be comments within the file that start with a # sign directly at the beginning of a line. These comments are ignored completely. Following parameters, that have to start exactly at the beginning of a line, are supported for configuring the interface:

axis=y : Specifies for which axis the following configuration parameters are valid. If this statement is not used somewhere in a configuration file it is assumed that the first axis is used. That statement assigns all following parameter to this axis until another *axis* parameter is found within the settings file. That means using the *axis* parameter it is possible to configure more than one axis. Here all parameters have to be repeated for every axis and every section needs to be started by such an *axis* statement. The axis number itself also specifies the position within the motion control panel of the scanner software the axis input fields will appear.

startOutput=y : This parameter specifies which output number has to be used to send the signal for starting the movement to the drive. The value given for *y* is a number that specifies the output port of the scanner card and the output pin that is connected with the input of the drive. If a value in the range *0..15* is entered here then a standard digital output of the scanner card is used. When the value is in the range *100..107* the pins *0..7* of the laser port are used for sending the step signal.

The last method can be used only if the laser port is NOT used for controlling the laser. Else the results can be undefined and may harm your equipment seriously.

doneInput=y : This parameter specifies which input number has to be used for the movement feedback. At this input a signal is expected that is sent from the drive to the controller to tell that the final position was reached. The value given for *y* is a zero-based number that specifies the input pin that switch is connected with.

doneInputValue=y : The preceding parameter an this one are related to each other: *doneInputValue* defines what signal at the input with number *doneInput* is expected to be interpreted as the information: *the movement was completed*. Here for *y* the values *1* and *0* are possible.

dname=c : This parameter specifies a name for the current axis that is used to display the name within the scanner software. That name has to consist of a single character.

4.4.10 Type 13 - Isel CanApi Controller

To use this motion control define type 13 in the file sc_motion_settings.txt. The corresponding settings file is sc_motion_isel_settings.txt. In the following the possible parameters are explained:

Debug=x: The debug mode can be enabled in case of problems, here different important information are put into a debug log named 'sc_motion_ext_isel_debuglog.txt' that is located at the same position like this settings file. To enable debugging set x=1 else set x=0.

Dll=C:\CNCWorkbench\Control\CAN\CanApi.dll: Set the Path to Isel Can Api Dll file. If path contains blank(s), set whole path between double quotes.

Ini=C:\\CNCWorkbench\\Control\\CAN\\CAN_PCI_1_Axis.ini: Set the Path to Isel Can Api Ini file. If path contains blank(s), set whole path between double quotes.

SwitchOffPowerState=x: If set to '1', a call to 'SetPowerState(1)' is done before each movement. After each movement (including homing) 'SetPowerState(0)' is called. 'SetPowerState(0)' disables axis amplifiers (switching them off), 'SetPowerState(1)' enables axis amplifiers (switching them on).

axis=x: Axis index (zero based)

llimit=x: Axis range - Low Limit ['um' in case POSITION, '1/100°' in case ANGLE]

hlimit=x: Axis range - High Limit ['um' in case POSITION, '1/100°' in case ANGLE]

5 Global Settings

In this chapter the global settings of SAMLight are listed. The corresponding menu in SAMLight is *Settings → System* found in the Menubar of the User interface.

5.1 Reset License

In SAMLight it is possible to redefine the license key. You can use it, if you have received an update password from SCAPS and to activate it. Therefore click in the menu on *Settings → Reset license*. A questioning window will appear, which you have to answer with OK. Then the usual password enter dialog will be displayed where you can enter the new password. If you copy the password out of a text editor first, you can easily paste it by clicking in the first enter field and then press Ctrl-V and Enter. After upgrading of a license, a new password is supposed to replace the old password, in this situation, a restart of SAMLight is necessary for resetting license.

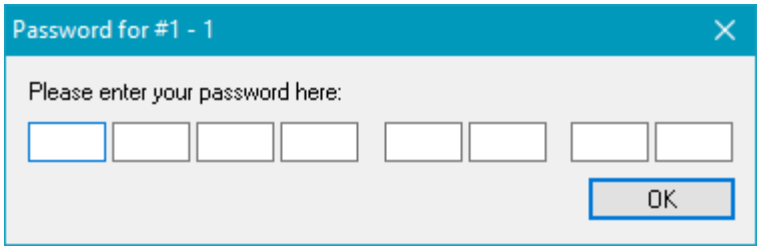


Figure 31: Reset License password Dialog

5.2 View

The following dialog can be reached by Menu item *Settings → System → View*.

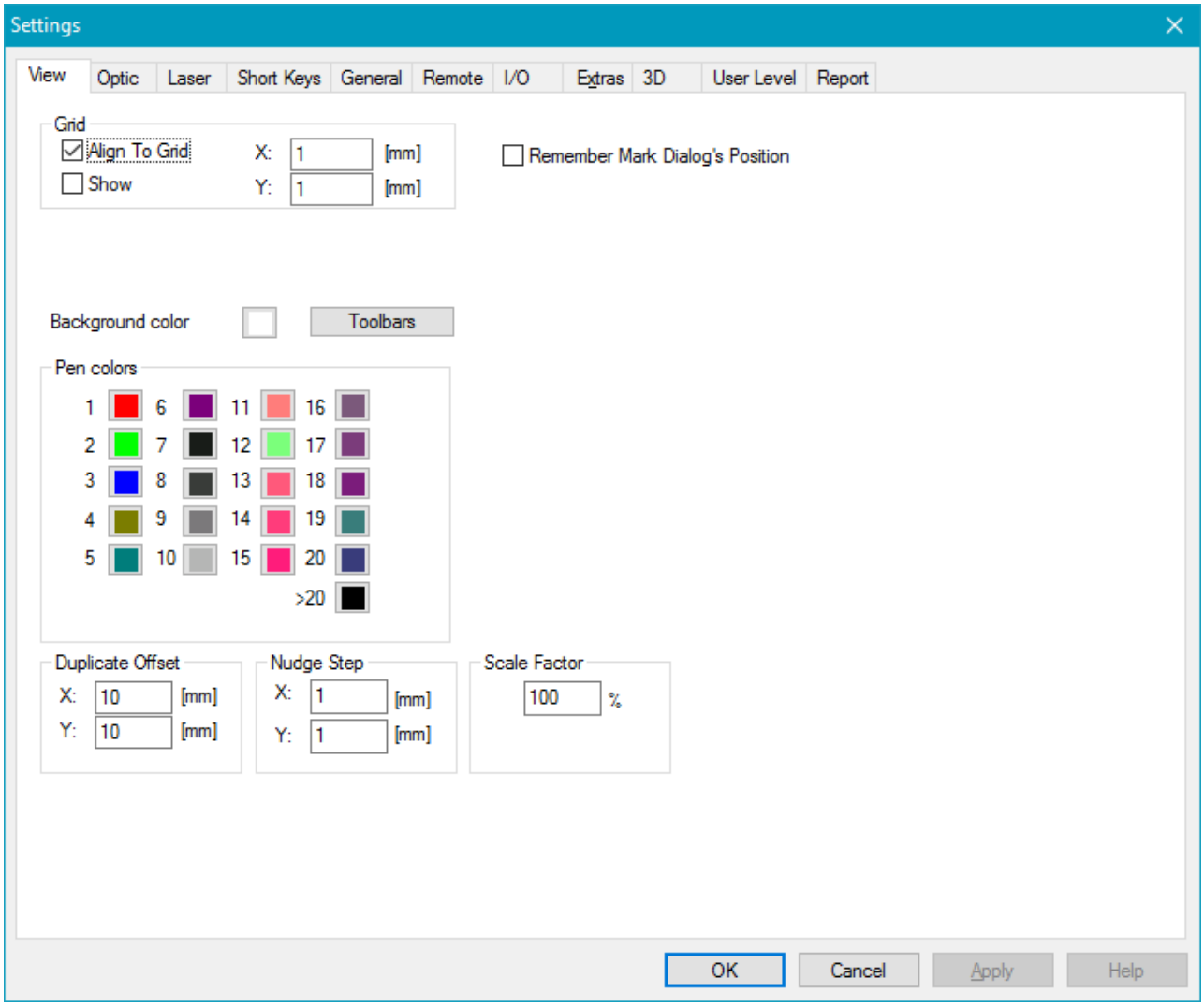


Figure 32: View Settings Dialog

Grid:

Align To Grid: If checked each new object placed in the View 2D will be aligned to the grid.

Show: If checked the grid will be displayed in the View 2D.

X, Y: These two values define the grid size.

Remember Mark Dialog's Position: If checked, the position of the mark dialog is remembered permanently (also after restarting SAMLight).

Background color: Clicking on this button opens a color dialog, where the background color of the View 2D can be defined (also possible in ViewProperties).

Toolbars: Clicking on this button opens a dialog where the user can choose which of the available toolbars is shown. See chapter Toolbars.

Pen colors: In this box, a color can be assigned to each pen (also possible inViewProperties). An object exposured by a special pen will be displayed with the assigned color in the View 2D. To get information about how to define pens see chapter Edit Pens.

Copy Offset:

X, Y: These two values define the copy offset in x and y direction. For example, select an object in the View 2D and click *Menu bar → Edit → Copy*. This creates a copy of the selected object which is placed next to the original translated by the copy offset. If the copy offset is zero in both directions the created copy will cover the original.

Nudge Step:

X, Y: These two values define the nudge step in x and y direction. The nudge step is used for the operations *nudge left*, *right*, *up* and *down* described in chapter Edit.

Scale Factor: This value is taken for the scaling of the red pointer in the mark dialog.

5.3 Optic

5.3.1 USC Cards

The following dialog can be reached by Menu item *Settings → System → Optic*. The dialog allows to define the settings of the geometrical dimensions of the scanner field for the selected *head*:

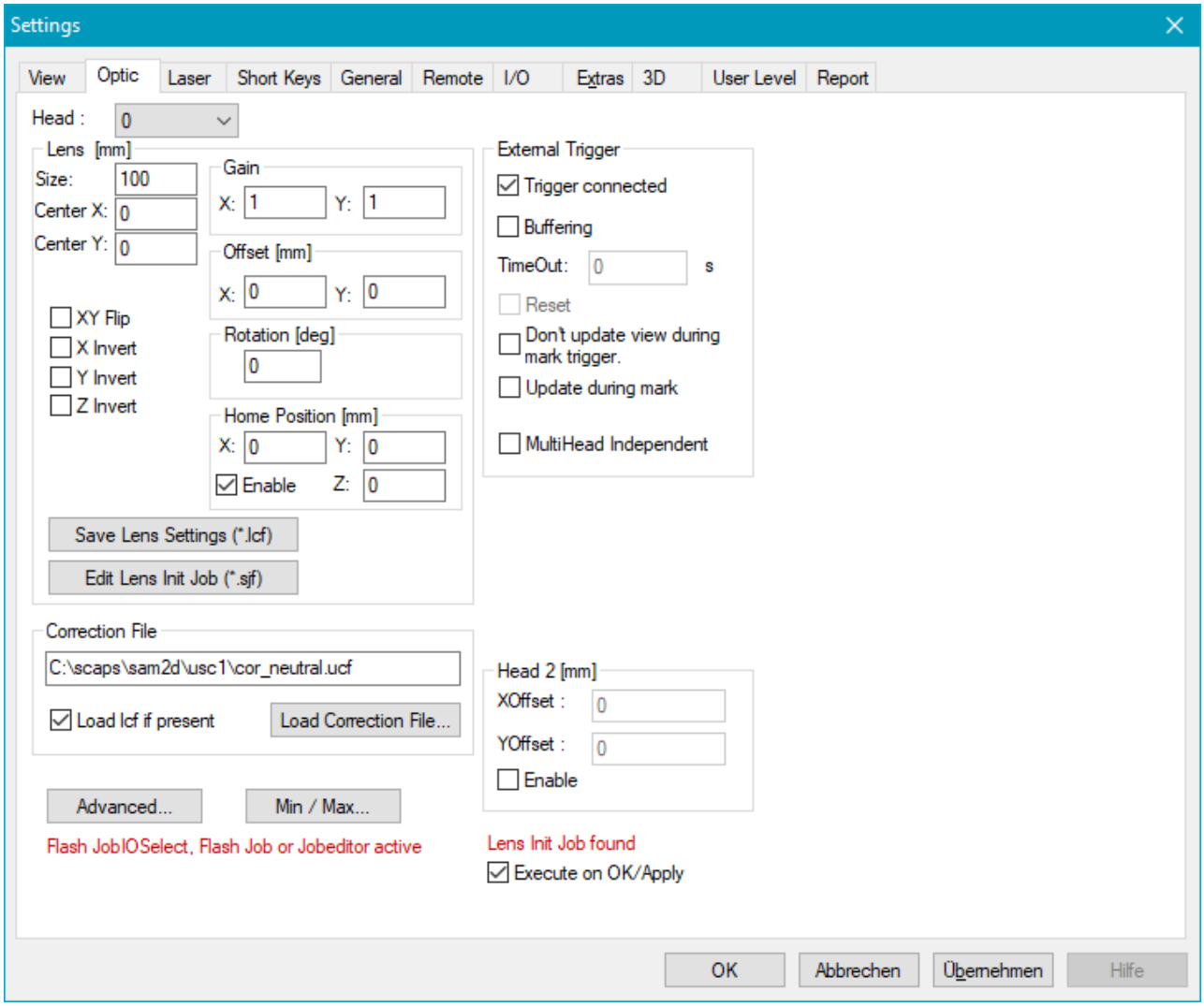


Figure 33: Optic Settings for USC cards

Head: Here the head number can be selected if more than one head is present. For this setting the software option MultiHead is required.

Lens [mm]:

Size: Specifies the maximum scanning field. This is the maximum field the scanner system can drive. In general it depends on the optical system (like lenses, etc.), the scanning distance and the maximum scanning angle of the scanner system. On 16 bit scanning systems the field extent corresponds in general to the bit values -32767 to 32767 respectively.

If the dimensions of output do not correspond to the dimensions of the drawn object, this can be corrected by changing the field size. Thereby the new field size is being calculated out of the current field size in the following way:

corrected field size = current field size * C / O

C is the current output dimension and **O** is the original dimension of the drawn object.

Center X, Y: Defines the center of the scanner field according to the world coordinate system.

XY Flip: Exchanges X and Y axis.

X, Y, Z Invert: Inverts the axis direction. *Z Invert* requires the SAM software options Optic3D or SAM3D.

Gain X, Y: The X and Y gain values are thought to compensate slightly X, Y gain errors to archive a quadratic field. Values less than 1 will reduce the scanner field. Values greater than 1 will expand the scanner field. Global gain errors which have the same deviation in X and Y directions should be corrected by changing the size of the scanner field (see above).

Example for calculating the size of the scanner field:

If for example the correction table has a step size of 550 bits/mm and the scanner has a 16 bit scanning system, the

size of the scanner field = 65535 bits * mm / 550 bits = 119.16 mm

Considering a Gain factor:

scanner field in x direction = 119.16 mm * *Gain X*

scanner field in y direction = 119.16 mm * *Gain Y*

Offset X, Y: The offset values are thought to slightly compensate X, Y offset errors to achieve the theoretical midpoint of the scanner field. Global offset errors which have the same deviation in X and Y directions should be corrected by changing the field X, Y min values in *field*.

Rotation [deg]: The scanner output is rotated by an angle that is entered here.

The Gain, Offset and Rotation values entered here will not affect the Optic Matrix. To change the Optic Matrix you have to go to Advanced → Correction → Settings.

Home Position X, Y, Z: If this option is enabled, the home position is the position where the scanner is located when no scanning takes place i.e. during handling activity. The home position is in normal cases outside the working area but it must be inside the scanner field.

Save lens settings (*.lcf): A lens settings file stores all lens specific information. The file will be generated or updated if you click this button. The name of the *.lcf file is bound to the name of the correction file you want to use and needs to be in the same directory. If you do not want to load a *.lcf file together with your correction file you can un-check 'Load lcf if present'. Please use the button 'Load Correction File...' to load the correction and *.lcf file. The data stored in the lens settingsfile include the whole parameters in the Lens field, the 4 parameters under *Settings → System → Laser → Redpointer* and the Redpointer delay in *Mark Dialog → Advanced*.

Edit Lens Init Job (*.sjf): Opens the Edit_Lens_Init_Job_Dialog. If a job should be executed every time a lens is chosen, a lens initiation job can be created. This job will be executed when switching to the corresponding correction file via this dialog. A typical job could be: waiting for an input, setting an output, executing a motion control or running an executable. The name of the lens init job (*.sjf file) must equal the name of the selected correction file and needs to be in the same directory (automatically done when using the button). When switching correction files, the message 'Lens Init Job found' indicates if an existing lens init job file for this correction file has been found. To suppress the execution of this file please un-check 'Execute on OK/Apply' before clicking apply/ok. If the job should not be executed any more it can be deleted in the corresponding directory.

Correction File Settings: Here a correction file can be specified. It is used to compensate optical distortions (barrel / pincushion) of the scan head. Usually the scan head manufacturer can provide a correction file for their products. For USC cards the standard extension is *.ucf*. By clicking on 'Load Correction File...' a

browser dialog opens. After selecting a correction file the corresponding lens settings file (*.lcf) and lens init job (*.sjf) will be opened together with the correction file if these files are stored at the same directory. That means using this button different lens- and correction file configurations can be managed.

Advanced...: See dialog USC-1 Card Settings or USC-2 Card Settings.

Flash JobIOSelect, Flash Job or Jobeditor active: Appears for USC-2 or USC-3 card if Flash JobIOSelect Mode, a Flash Job or a Jobeditor license is used and deactivates the "Advanced..." button. To activate it in case of Flash JobIOSelect Mode, please open Flash dialog at *Menu bar → Extras → Flash* an press "STOP IOSelect" button, which then turns to "START" button.

Min:/Max: Opens the Min/Max Dialog to define the range of the values speed, frequency and first pulse killer length of the laser.

External Trigger:

Trigger connected: enables the external trigger and the flags described below. This checkbox activates the "External trigger" in the Mark dialog and "Mark - Trigger".

Buffering: If in trigger mode, the job is buffered (several times) on the scanner card. This allows a faster start of the marking in response to the external trigger signal. The buffer is refilled before it is drained. In buffering trigger mode, the Status Outputs do not work.

TimeOut: If waiting for the trigger, the buffer on the scanner card will be cleared after there was no external trigger signal for longer than a given time. While clearing and refilling the buffer an external trigger signal is ignored. The time out value needs to be bigger than the time for marking.

Reset: If this option is enabled it extends the timeout operation. In this case not only the buffer of the scanner card is cleared but the sequence is reset too. That means when a timeout occurs and this checkbox is selected all serial numbers are reset to their defined starting values automatically.

Don't update view during mark trigger: If this option is enabled the view will not be updated when using Mark → Trigger as long as the Mark Trigger window is active.

MultiHead Independent: This allows to trigger the scan heads independently from each other when the Mark→Trigger dialog is used. This might be useful if one of the scan heads has to mark for example 5 objects while the other scan head will only mark one object at the same time. This function will not work together with serial number objects nor with DateTime objects.

Z Dimension: This requires the software option Optic3D.

Optic settings can be changed in the optic settings dialog which can be reached by Menu item Settings → System → Optic, but there is also an other way:

1. Close all SAM based applications (Check this with task-manager).
2. Start *sc_setup.exe* from folder <*SCAPS*>*tools*\
3. Click on Menu *Hardware Settings → Settings*

Head 2 (*only available for USC-2*):

X, Y Offset: Define an offset for the second scan head. If everything is set to 0 then the second scan head will mark on the same position as the first head is marking.

Enable: Activate this checkbox to enable the usage of the second scan head.

If the MultiHead option is used, then the Head 2 options are disabled. Instead there is an additional option in the External Trigger options: MultiHead Independent. Therefore see Optic Settings Dialog USC-1.

Lens Init Job found: Appears if a correction file is loaded and an appropriate Lens Init Job is available.

5.3.1.1 Edit Lens Init Job Dialog

The following dialog can be reached by using menu item *Settings → System → Optic. Here for a specific correction file (*.ucf) an init job can be defined which will be executed automatically if the corresponding correction file with the same name is chosen via the 'Load Correction File...' button.*

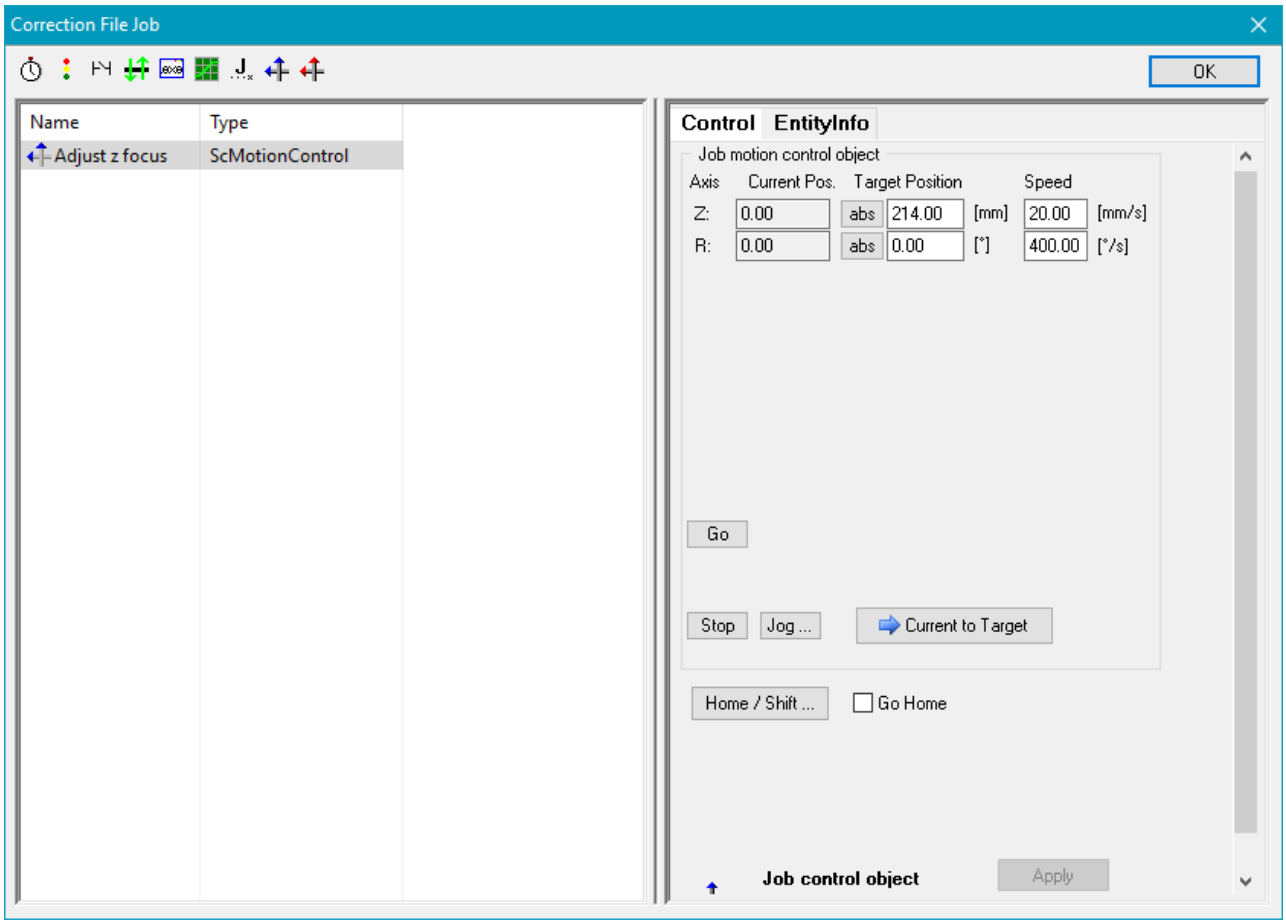


Figure 34: Correction File Job Dialog

5.3.2 RTC Cards

The following dialog can be reached by Menu item *Settings* → *System* → *Optic*. The dialog allows to define the settings of the geometrical dimensions of the scanner field for the selected *head*:

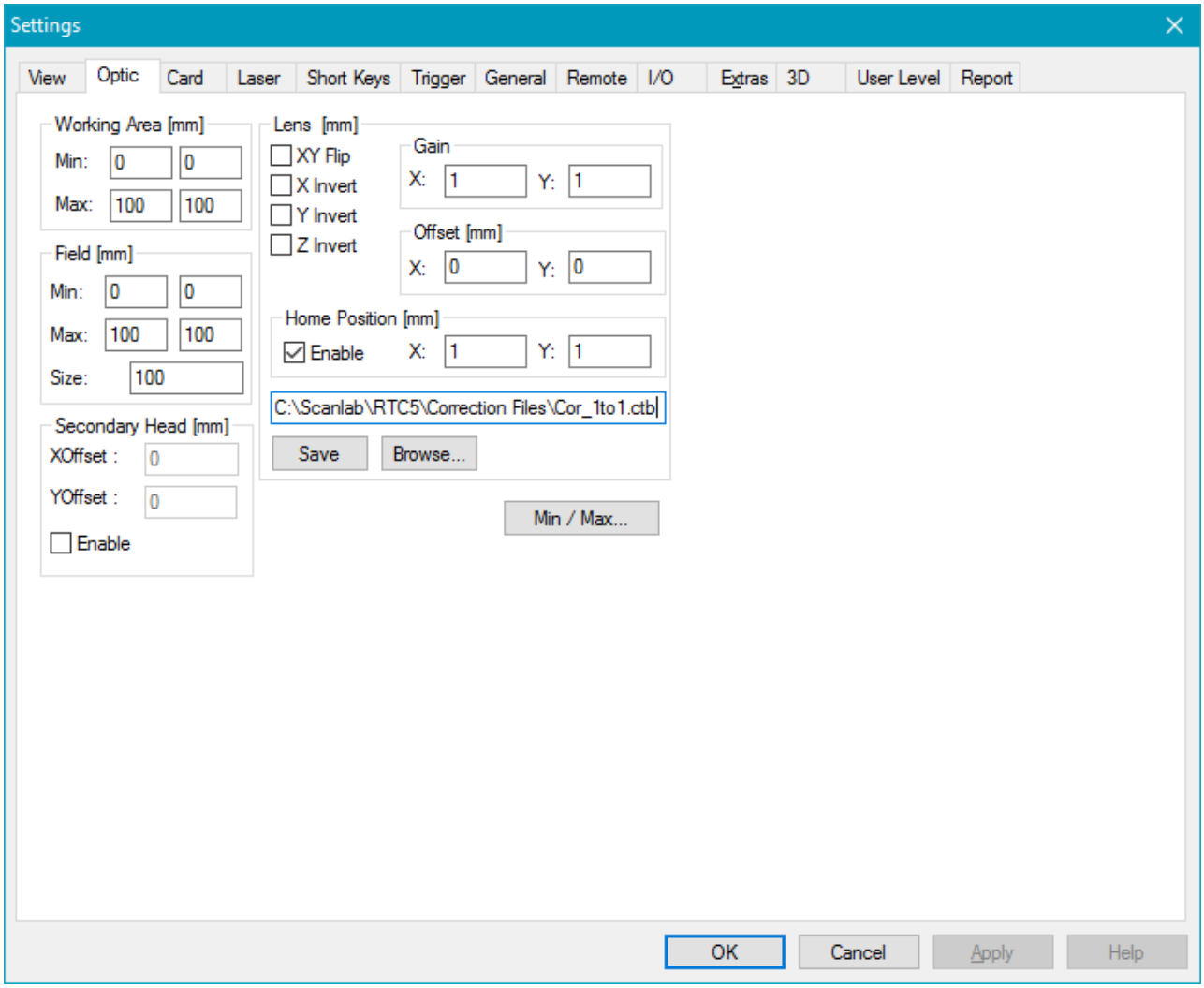


Figure 35: Optic Settings for RTC cards including SCANalone

Working Area [mm]:

Min:/Max: Specifies the Min and Max X, Y scanning field on which the exposure should take place. The working area must be inside the *field* dimensions.

Field [mm]:

Min:/Max: Specifies the Min and Max X, Y scanning field.

Size: Specifies the maximum scanning field.

Secondary Head [mm]: This field is available for RTC3/RTC4 and SCANAlone only. It allows to control two heads with one card. After enabling it there are two working areas shown on the job editor.

X, YOffset: Defines the offset of the second head to the first head. For more information about how to use a second head see also the chapter Option Multihead.

Lens [mm]:

XY Flip: Exchanges X and Y axis.

X, Y, Z Invert: Inverts the axis direction. *Z Invert* requires the SAM software options Optic3D or SAM3D.

Gain X, Y: The X and Y gain values are thought to compensate slightly X, Y gain errors to archive a quadratic field. Values less than 1 will reduce the scanner field. Values greater than 1 will expand the scanner field. Global gain errors which have the same deviation in X and Y directions should be corrected by changing the size of the scanner field (see above).

Example for calculating the size of the scanner field:

If for example the correction table has a step size of 550 bits/mm and the scanner has a 16 bit scanning system, the

size of the scanner field = 65535 bits * mm / 550 bits = 119.16 mm

Considering a Gain factor:

scanner field in x direction = 119.16 mm * *Gain X*

scanner field in y direction = 119.16 mm * *Gain Y*

Offset [mm] X, Y: The offset values are thought to slightly compensate X, Y offset errors to achieve the theoretical midpoint of the scanner field. Global offset errors which have the same deviation in X and Y directions should be corrected by changing the field X, Y min values in *field*.

Home Position X, Y, Z: If this option is enabled, the home position is the position where the scanner is located when no scanning takes place i.e. during handling activity. The home position is in normal cases outside the working area but it must be inside the scanner field.

Correction File Settings: Here the correction file can be specified. This file normally is delivered by the scan head manufacturer and needs to fit to the scanner. For RTC-cards the standard extension is *.ctb*. When the *Save* button is pressed the Lens settings from this panel are stored related to the currently used correction file. By clicking on *Browse...* such a correction file and its related settings can be loaded afterwards. That means using these buttons different lens- and correction file configurations can be managed.

Min:/Max: Opens the Min/Max Dialog to define the range of the values speed, frequency and first pulse killer length of the laser.

5.3.3 Min/Max

By pressing *Min /Max...* in the Menu *Settings → System → Optic* dialog following dialog appears. The settings define the range of the values *Speed* [mm/s], *Frequency* [kHz] and *First Pulse* [µs] killer length of the laser.

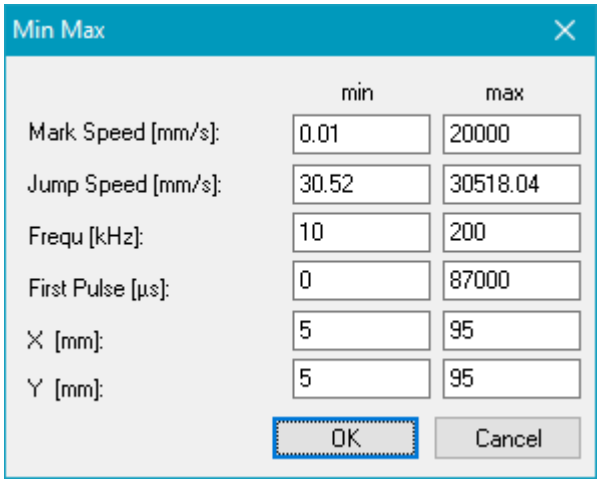


Figure 36: Min Max Dialog

When using the USC-1 or USC-2 card additional min max values, *X [mm]* and *Y [mm]* coordinates, can be set to define the working area inside the scanner field.

The Min and Max values of X and Y are automatically adapted to the field size values (default), if Size / Center X / Center Y in the optic dialog are being changed.

The Min and Max value would be saved in pen settings. So, if "save pens on exit" in General is not enabled, don't forget to click the button "Save Pens Now".

Although the Min and Max values are saved in pen settings, after loading a job file with pen settings, the Min and Max values in system would not be overwritten.

5.4 Laser

The following dialog can be reached by using menu item *Settings* → *System* → *Laser*, here the laser and the shutter control can be configured.

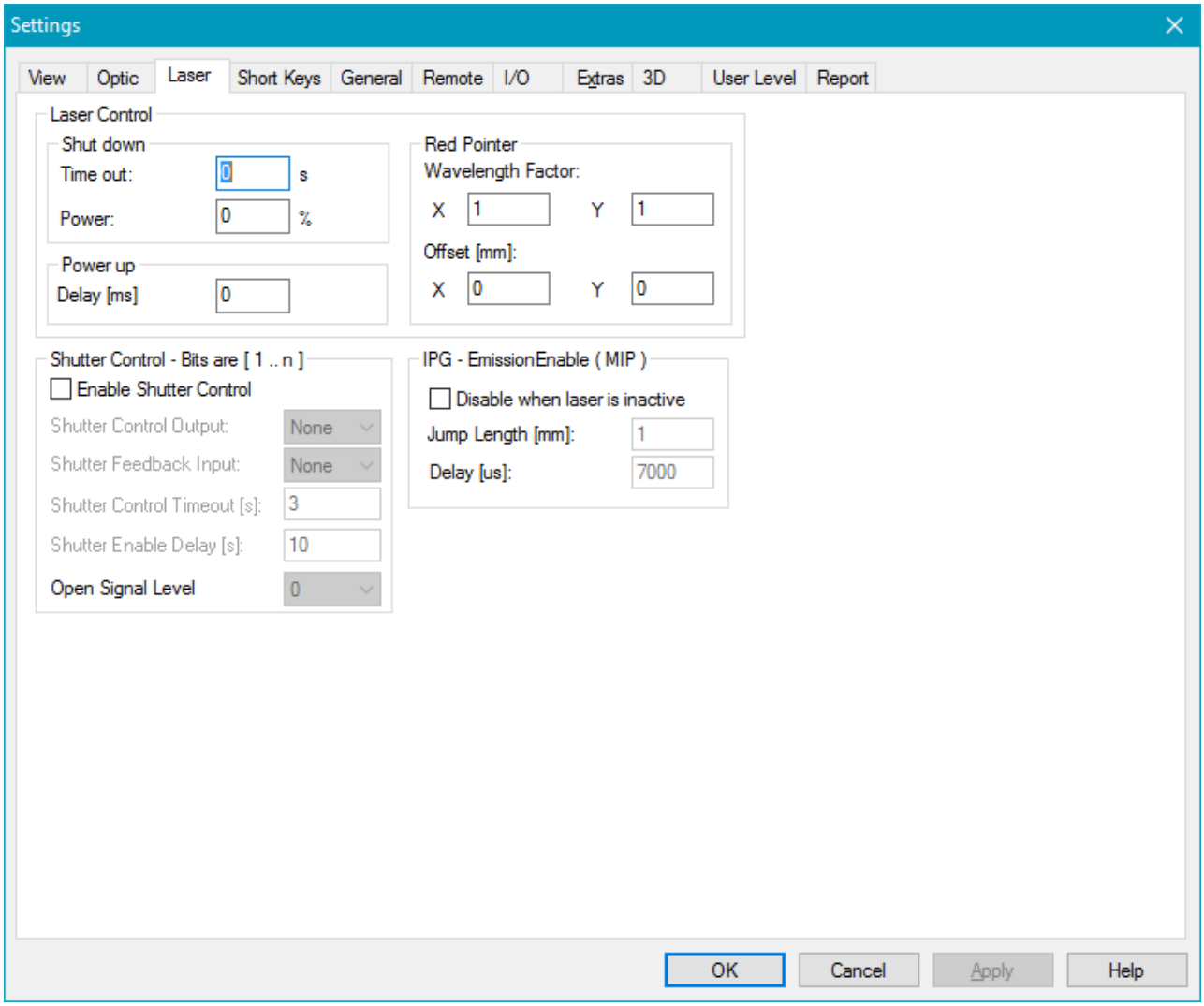


Figure 37: Laser Settings Dialog

Laser Control:

Shut down (power save mode):

Time Out: The Laser is going to power save mode after a defined time in which the laser was not active. If a value of 0 is entered here the laser power save mode feature is disabled.

Power: Percentage of laser power for the power save mode. When the power is '0' the laser will shut down.

Power up:

Delay: When the laser is in power save mode and is going to normal power again a delay can be defined for the power up procedure. In this power up time the laser power will ramp from the power save to full power. The laser gate and the marking in process signal will not be active during the power up procedure. After the power up procedure the laser is in normal mode again and the marking procedure will begin.

Red pointer:

Wavelength Factor: Correction for the red pointer in X and Y direction caused by different wavelength of the laser. The factor is determined experimentally. When marking one reference point with the red pointer and the laser, the vector length R from center to the point marked with the red pointer and the vector length L from center to the point marked with the laser needs to be measured in x and y direction. Wavelength Factor = Red pointer / Laser.

Offset: Static X/Y offset correction only for the red pointer caused by the different wavelengths of the laser. The factor is determined experimentally (see Wavelength Factor).

Shutter Control:

Enable Shutter Control: If this checkbox is selected the shutter control functionality is enabled and all following parameters are used for it.

Shutter Control Output: Here the digital output that is used to control the shutter can be defined. It sends opening and closing signals to the shutter depending on current state of the application. If the shutter is closed an opening signal is sent before the laser is turned on e.g. for marking operations. If marking has finished and the enable delay (please see input field described below) has elapsed a close signal is sent to the shutter.

Shutter Feedback Input: This combo box can be used to select a digital input that can be used for reading back the current opening or closing state of the shutter.

Shutter Control Timeout: Here a timeout value can be configured that is related to the "Shutter Feedback Input". If the shutter is controlled by the application to open or close an error message is displayed and the current operation is cancelled if that operation takes longer than the timeout that is configured here. This timeout requires an input where the current state of the shutter can be read back.

Shutter Enable Delay: With this delay the shutter can be closed automatically and without any user interaction. If more seconds have been gone after the last time the laser was turned on, the shutter is closed automatically. So this field defines a delay after the last marking operation after that the shutter will be closed.

Open Signal Level: The level of the output signal (high or low) that has to be used to open the shutter. In order to close the shutter the other signal is used. With this option the behaviour of the scanner software can be modified in order to fit to the used shutter hardware.

IPG - EmissionEnable: This field is available only if using an USC-1 or USC-2 card and if *IPG* is selected under *Settings → System → Optic → Advanced → Mode*. If an IPG laser is connected this mode allows it to switch off Emission Enable (connected to OPTO_OUT_0) during jumps.

SwitchOffDuringJumps: If this checkbox is activated then the IPG laser will be switched off during jumps.

Jump Length [mm]: Jumps that are longer than this value will lead to a switch off.

Delay [µs]: When switching on after a jump, this delay is executed before the scanner continues with the next vector.

5.5 Shortkeys

The following dialog can be reached by Menu item *Settings → System → Short Keys*.

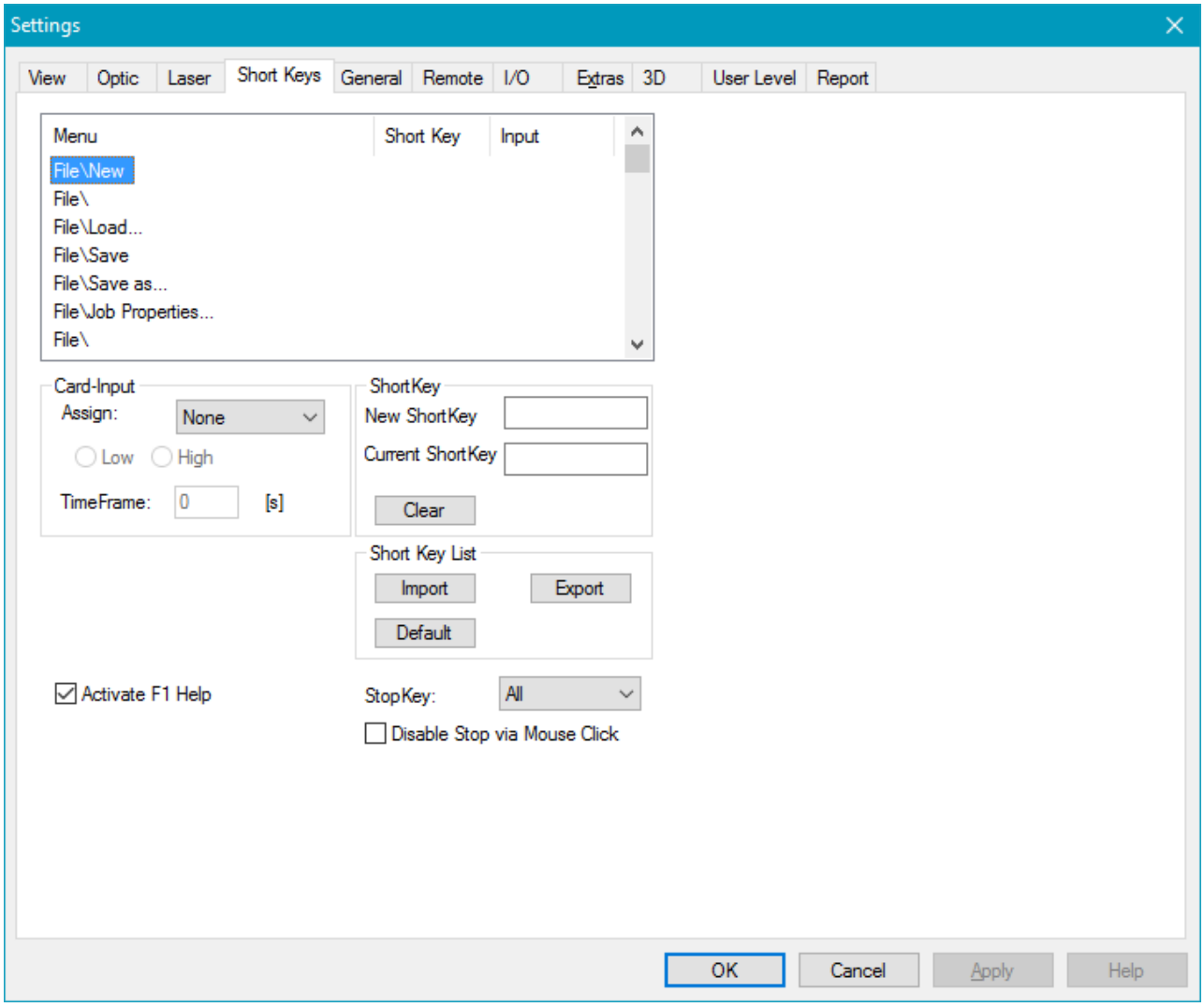


Figure 38: Short Keys Settings Dialog

Card-Input (only available for USC-1 and USC-2): To each short key a card input can be assigned. There are six input bits available (IN1 to IN6) which corresponds to the OPTO_IN0 to OPTO_IN5 pins of the USC card. The default assignment is None. The effect is that a special short key will be triggered if the assigned INx is high/low if high/low was selected.

TimeFrame: Defines a repetition rate of the operation the input bit is assigned to. The operation will be executed as long as the input bit is low (or high) with the repetition rate 'TimeFrame' in seconds. The lower limit for this value is about 200 ms.

Short Key: For each menu item a user defined short key can be assigned: Select the requested menu item from the list. Click the empty field *New Short Key* and press the corresponding short key on the keyboard (for example F2) to assign it to the selected menu item.

Activate F1 Help: Activates the context sensitive help. Therefore the F1 key is predefined.

The default short key for the start buttons inside the mark dialog is F1. So if Activate F1 Help is selected the short key defined for the menu item Mark → Start will be taken for those buttons instead.

Short Key List: With *Import* a short key file (*.sam) can be imported. The short key list defined in the dialog can be exported by *Export*. *Default* restores the short key list to its default values.

StopKey: The keys that stop marking can be selected within this combo box.

Disable Stop via Mouse Click: If you enable this function, marking can only stopped with the defined StopKeys.

5.6 General

The following dialog can be reached by the Menu item *Settings → System → General*.

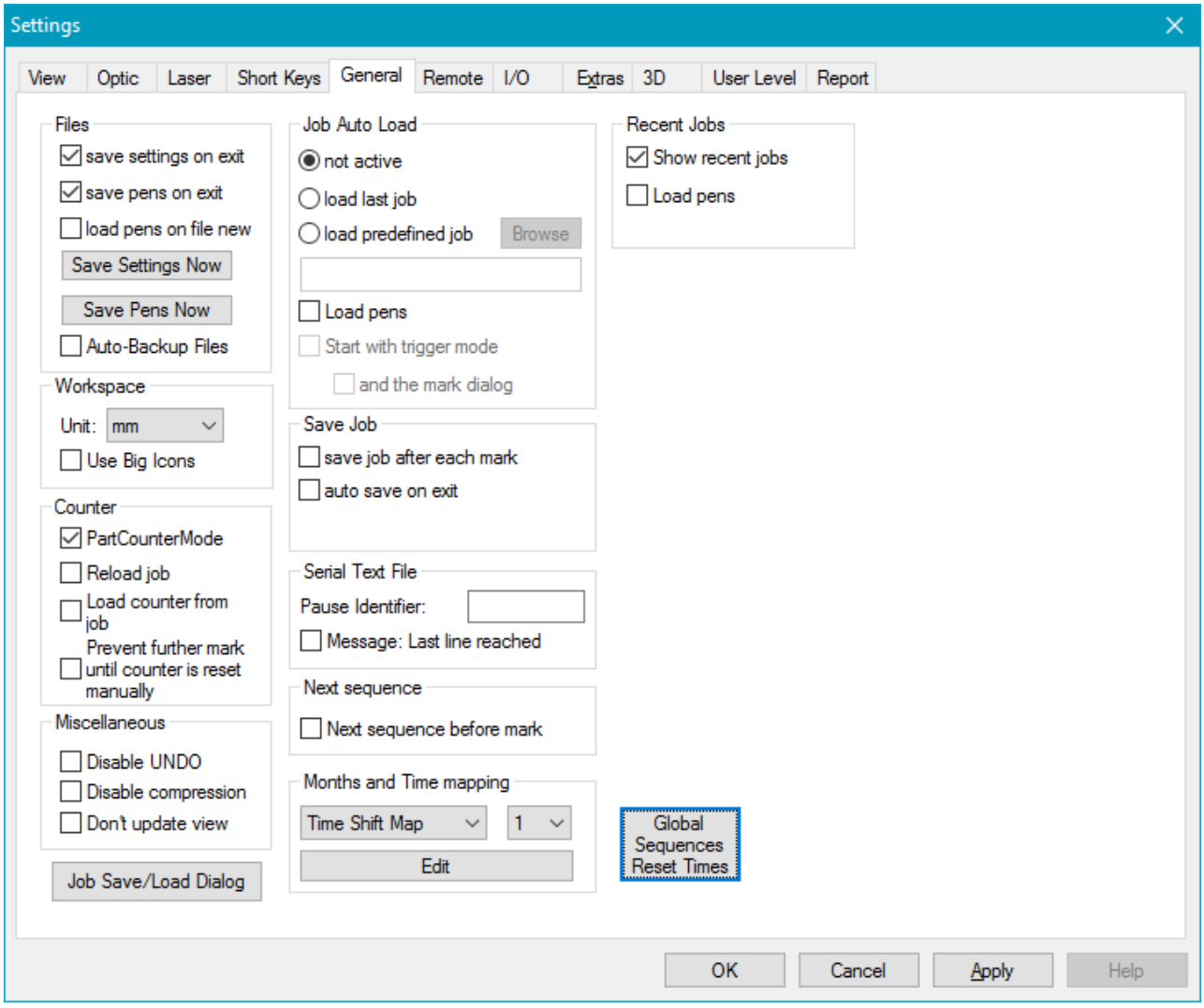


Figure 39: General Settings Dialog

Files:

save settings on exit: If checked, the modified settings will be saved on every program exit.

save pens on exit: If checked, the pens will be saved on every program exit and overwrite the last ones.

load pens on file new: If checked, the pens are loaded by clicking on *Menu bar → File → New*.

Save Settings Now: Saves the modified settings now and overwrites the last ones.

Save Pens Now: Saves the pens now and overwrites the last ones.

Auto-Backup Files: Creates automatically backups with a continuous number in range 1..99 when this box is checked. To restore such a backup, the desired *.sjb file has to be renamed into *.sjf and can be loaded normally afterwards. The backup file names consist of the original file name plus the backup number and the new extension .sjb.

Job Auto Load Box:

not active: No job will be loaded at program start up.

load last job: The last edited job will be loaded at start up.

load predefined job: If this is activated browse through the file tree (button *browse*) and choose a *.sjf file to be your predefined job or type in the absolute path to this file in the edit line below. The chosen file will be loaded at start up.

Load pens: If this is checked the pens of the job will be loaded as well, otherwise only the objects. This only makes sense if pens are saved within this job. See Job Format.

Start with trigger mode: If checked the application switches into trigger mode after loading a job at start up or after loading a job in Job Select input mode. The effect is the same as loading a job and starting the trigger mode by clicking *Menu bar → Mark → Trigger*.

and the mark dialog: Defines if you want to start with the trigger dialog (standard) or with the mark dialog. Indeed, in some versions of SAMLight, the trigger dialog forces the user to start marking manually before accepting trigger signals.

Save Job:

save job after each mark: Saves the job after each mark.

auto save on exit: The job is saved automatically when closing SAMLight.

Workspace:

Unit: Here on can define the length unit of the workspace. Three settings are possible: Millimeters (mm), Inch (inch), Bits (bits)

Use Big Icons: When this field is checked, bigger icons with double width and height are used after the next program start for all toolbars. This feature is useful e.g. in environments with limited input capabilities like touchscreens. Other settings like the sizes of menus and window title bars are not subject to the application. These properties are managed by the operating system exclusively. To get menu entries and title bars that are big enough to grip them via a touch, please change the appropriate operation system settings.

Counter:

PartCounterMode: Enables counting number of parts per marking process. The counter will be set to the number of marked parts while marking instead of being set to the number of marking sequences. One part is made of the number of objects defined in the job properties. If for example NumObjectsPerPart is set to 2 and there are 10 objects in the joblist, the counter gets incremented with 5 after each marking sequence.

If quantities is defined and it is not a multiple of the number of parts in the joblist, the software offers to delete the overlaying objects, so that they do not get marked within the last sequence. To ensure that the original job will not be changed, the following check box can be selected.

Reload job: Reloads the job after quantity got reached.

Load counter from job: The counter information can be stored in the job. When loading the job the counter is also set to the old value.

Serial Text File:

Pause Identifier: The string that is defined here is used as a pause identifier for serial numbers read from an ASCII file. When the identifier is found in the current line of the assigned text file it causes a break of the marking and a pop up window which asks for continue is opened.

For how to use ASCII for serialization see chapter How To / Automate Serialization.

Message: Last line reached: Gives a message if the current line of the serial number text file which needs to be assigned to the serial number is empty. The user has the possibility to reset the serial number sequence with committing the message. Instead or in addition you can also set an I/O bit if the last line of the text file has been reached.

Miscellaneous:

Disable UNDO: If this checkbox is selected the UNDO-and REDO-functionality is disabled completely. That means changes within a job have to be reverted manually, the automated function is disabled. It is recommended to disable the UNDO-functionality in case of speed or memory problems on smaller computer systems (embedded systems which are not used to create and edit jobs but to control the marking process only). Depending on the size and complexity of a job this option is able to save nameable amounts of memory and calculation time.

Disable Compression: If this checkbox is selected the zip compression of a saved *.sjf file is deactivated. This can help to save big job files on computer systems with low main memory.

Don't update view: If this checkbox is selected the View2D is not updated any more. This can help to save processing time, when entities are updates (date/time and serial number entities, reimport of bitmaps, ...)

Next sequence (not for Splitting):

Next sequence before mark: If you start the marking from the mark dialog at first the serial numbers in the job will be increased and then marking will start.

Months and Time mapping:

Time Shift Map: Opens a dialog where the working shift placeholder of date time can be mapped to working shifts times, see chapter Shift Map.

Year Map, Months Map, Day Map: The naming of the years, months and days can also be fully customized.

Combo Box: Up to 4 different maps can be defined at the same time. To address a certain shift map with the % placeholder it is necessary to put the corresponding number before the letter. E.g if you want to adress Time Shift Map number 2 you would have to write %2T. If no number is specified then number 1 is assumed.

In Flash jobs (USC-2/3) only the first shift/year/month/day map is available and will be used instead of the applied map number. The map has to be stored to the USC-2 card by clicking the STORE button in Optic settings→advanced.

Global Sequence Reset Times: When activated, serial numbers which are assigned to a global sequence can be reset. Chose the sequence in the drop down menu and set the desired reset date, time and period. See chapter Global Sequences Reset Times Dialog.

Job Save/Load Dialog: Opens a dialog where the defaults for saving and loading of jobs can be edited. See

chapter Job Save/Load Dialog.

5.6.1 Shift Map

The following dialog allows to define a string mapping for working shifts. The according placeholder of Date Time is automatically replaced with these string definitions regarding the time of day. For how to define this placeholder, see the format definitions of Date Time object. The dialog can be reached by *Menu bar → Settings → System → General → Shift Map*.

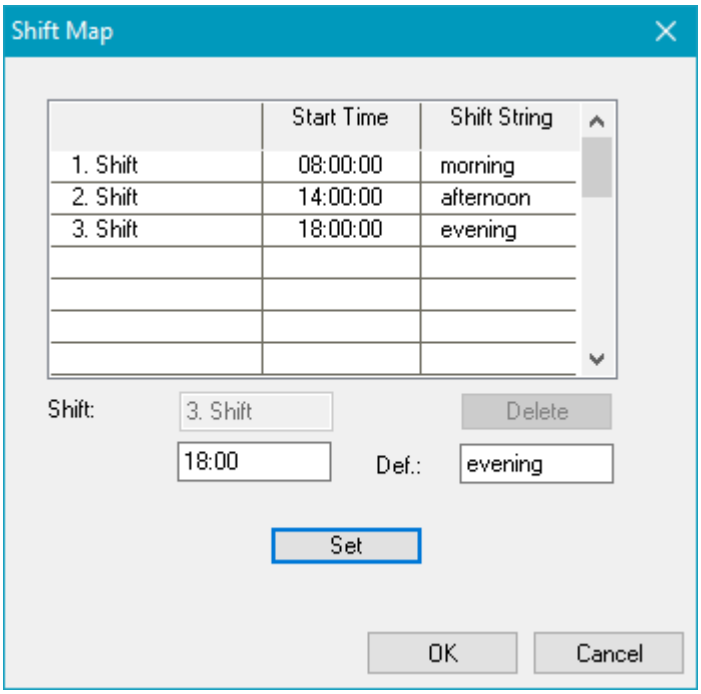


Figure 40: Shift Map Dialog

Shift: Shows the working shift which is currently selected.

Delete: Deletes the currently selected working shift.

Shift Start: Defines a start time for the selected working shift. The shift ends with the definition of a following shift.

Time format: hour:minute:second, Range: hour 0-23, minute 0-59, second 0-59.

Def.: Defines a string for the shift placeholder of date time to the given shift time.

Set: Creates a new shift if it is not defined yet, otherwise overwrites the selected working shift. The newly defined working shift gets numbered and sorted according to its start time.

NOTE

In Flash jobs (USC-2/3) only the first shift map is available and will be used instead of the applied map number. The map has to be stored to the USC-2 card by clicking the STORE button in Optic settings→advanced.

5.6.2 Months Map

The following dialog allows the user to enter a customized naming for each month of the year. The dialog can be reached by *Menu bar → Settings → System → General → Month Map*.

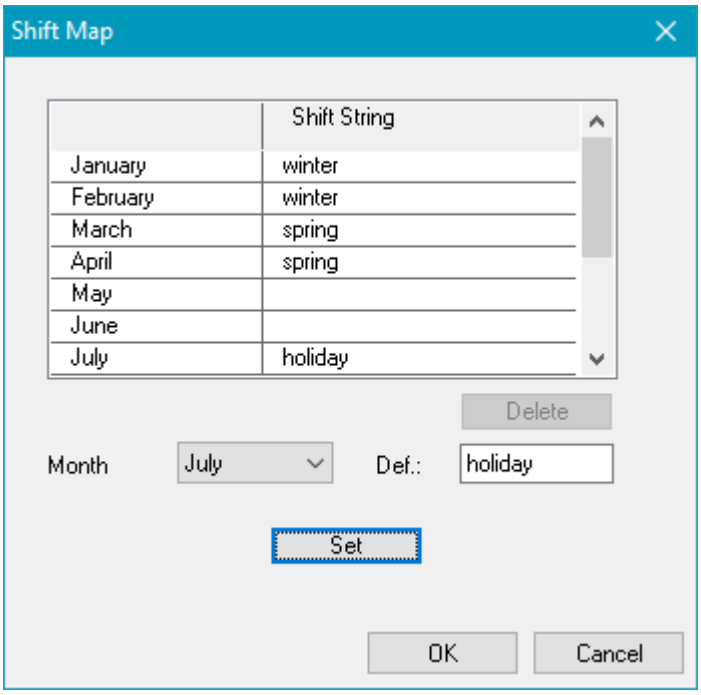


Figure 41: Months Map Dialog

Month: Tells which month has to be renamed.

Delete: Deletes the current selected working shift.

Def.: Set desired shifting text here.

Set: Enters the modification in the shift list.

NOTE

In Flash jobs (USC-2/3) only the first month map is available and will be used instead of the applied map number. The map has to be stored to the USC-2 card by clicking the STORE button in Optic settings→advanced.

5.6.3 Day Map

The following dialog allows the user to enter a customized naming for day of a month. The dialog can be reached by *Menu bar → Settings → System → General → Day Map*.

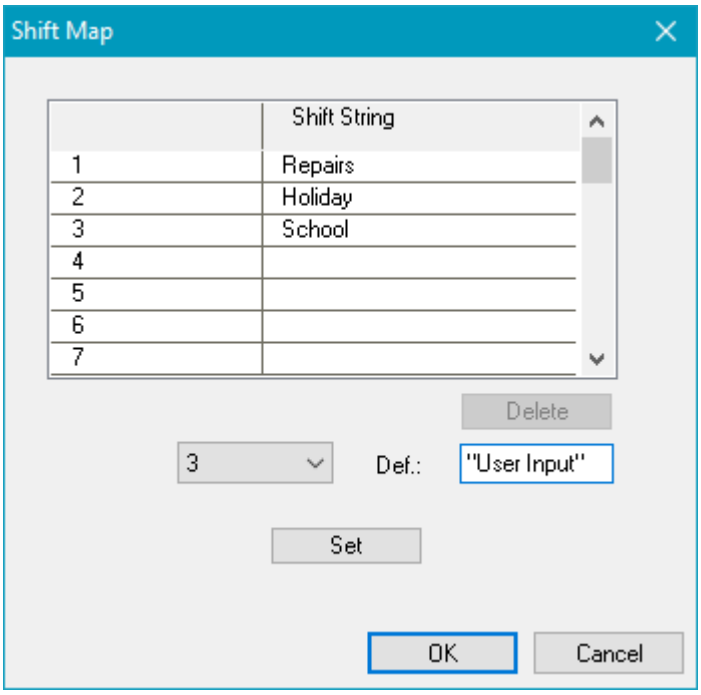


Figure 42: Day Map Dialog

Day: Tells which day has to be renamed.

Delete: Deletes the current selected working shift.

Def.: Set desired shifting text here.

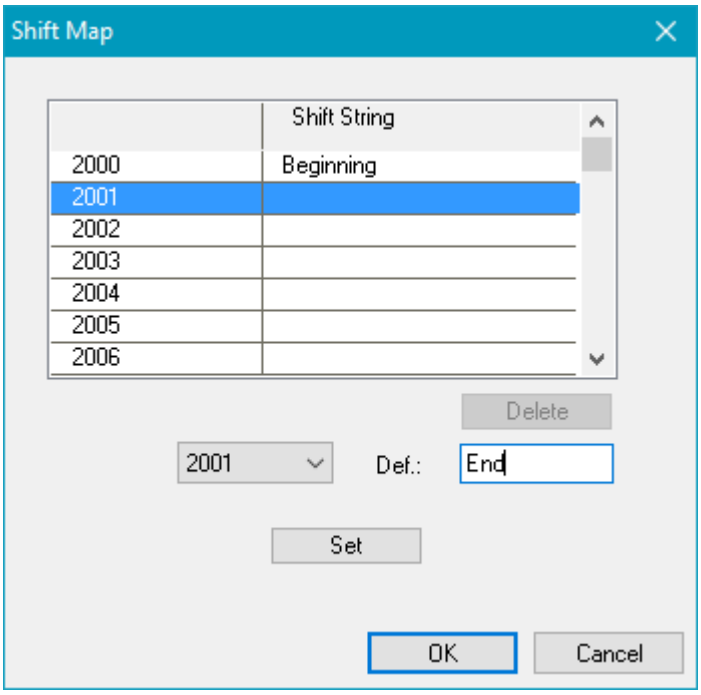
Set: Enters the modification in the shift list.

NOTE

In Flash jobs (USC-2/3) only the first day map is available and will be used instead of the applied map number. The map has to be stored to the USC-2 card by clicking the STORE button in Optic settings→advanced.

5.6.4 Year Map

The following dialog allows the user to enter a customized naming for a year. The dialog can be reached by *Menu bar → Settings → System → General → Year Map*.



Year: Tells which year has to be edited.

Def.: Define the edit string.

Set: Enters the modification in the shift list.

Figure 43: Year Map Dialog

NOTE

In Flash jobs (USC-2/3) only the first year map is available and will be used instead of the applied map number. The map has to be stored to the USC-2 card by clicking the STORE button in Optic settings→advanced.

5.6.5 Gloabal Sequences Reset Times Dialog

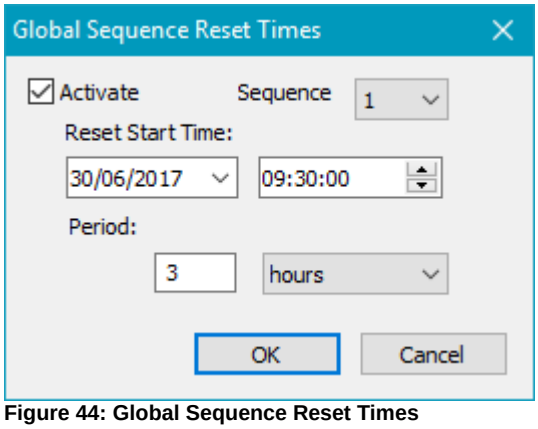


Figure 44: Global Sequence Reset Times

5.6.6 Job Save/Load Dialog

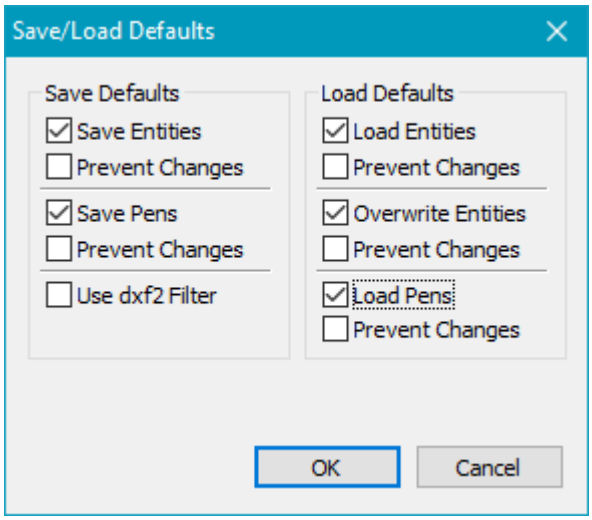


Figure 45: Save/Load Defaults Dialog

In each of the 5 fields there are 2 checkboxes. The upper checkbox has 3 states: On, Off, Gray. If Gray, the behaviour of the dialog is as usual: The last used state is used. The second checkbox *Prevent Changes* makes this default setting so that it can not be changed by the user anymore.

5.7 Remote

The settings for an external control are available in *Menu bar → Settings → System → Remote*:

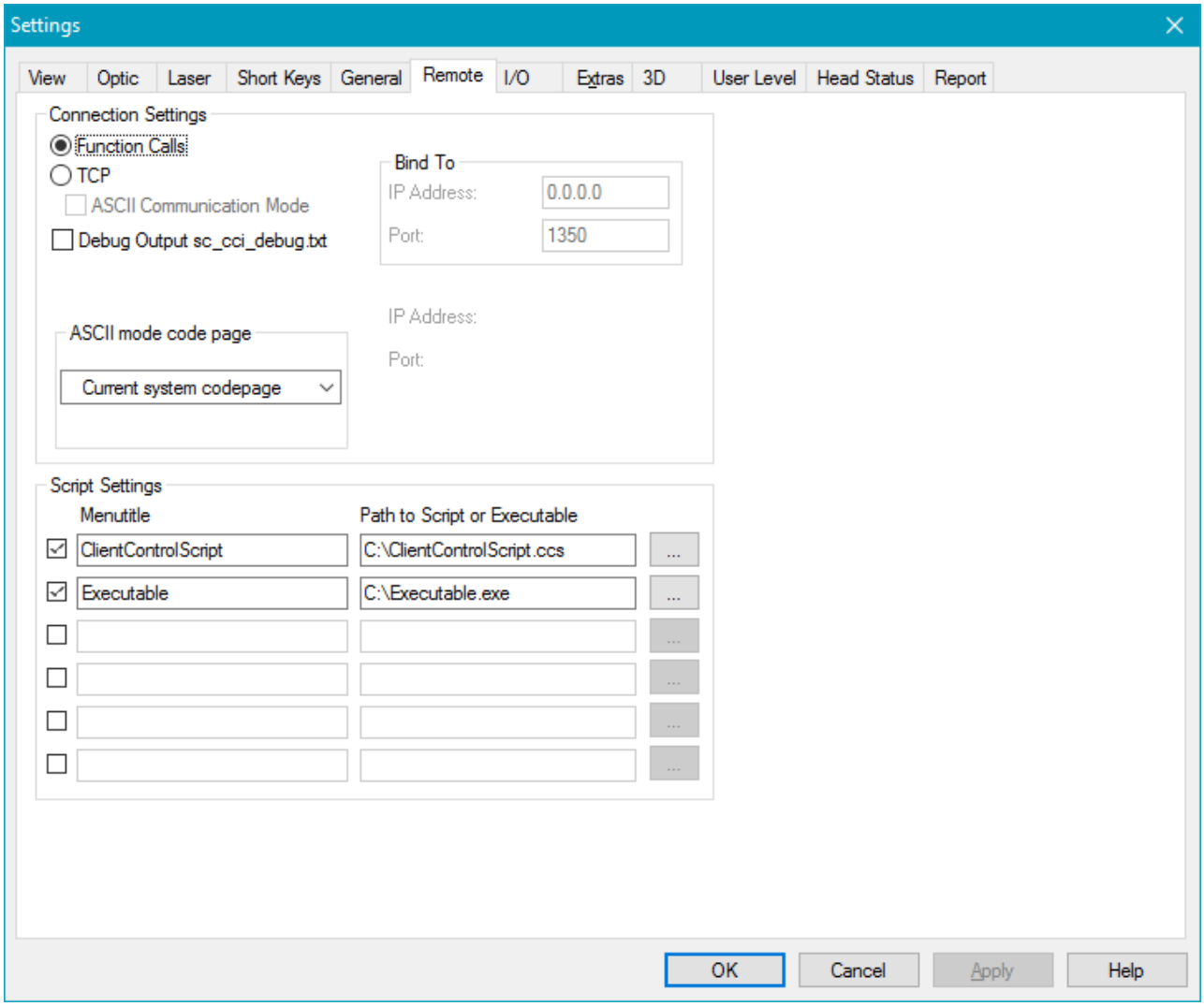


Figure 46: Remote Settings Dialog

Connection Settings: Here it is possible to define the connection method that an external CCI application uses to access the scanner application.

Function Calls: This option is only achievable inside one PC. That option has to be chosen if the controlled software and the application that controls it via the client control interface are running at the same host.

TCP: If the program should be controlled over the network using the client control interface, this option has to be selected. Normally the TCP protocol should be used. Only one TCP connection can be established at the same time.

ASCII Communiation Mode: If checkbox is activated it is not necessary to send the initialization string "SAM CCI Plain\n" to start SAMLight Client Control in ASCII communication mode. This is particularly helpful if SAMLight Client Control is always used in ASCII communication mode and also for testing and debugging your own SAMLight Client Control ASCII communication application.

Debug Output sc_cci_debug.txt: If this checkbox is active, a debug text file is written after each ClientControlInterface command. When SAMLight starts the file <SCAPS>\system\sc_cci_debug.txt will be deleted. The first CCI command after SAMLight start creates the text file and writes the debug

information into it and saves the file. All further CCI commands (in the same SAMLight session) will append the file with their debug information and save the file. The timing resolution in sc_cci_debug.txt is 16ms.

ASCII mode code page: For TCP ASCII Communication Mode normally the current used system code page of Windows is used to send the complete command string to SAMLight. If a different code page must be used to set or get text contents (especially relevant to set a text of a text entity in another language) you can select and set the required code page from the drop-down list.

Bind To:

IP Address: This is the IP address of the PC on which SAMLight is running. Here following values are possible:

- *0.0.0.0* if the software has to be accessible from everywhere
- One of the host systems IPs if it uses more than one and if the software has to be accessible by only one specific IP (Depending on the setup of your local network, this should be something like *192.168.1.100*. Please ask your system administrator for details.)
- *127.0.0.1* if only local connections have to be accepted. This case is a more theoretical one and normally should be used for testing purposes only because for plain local connections the *Function Calls* option can be used.

Port: Number defines where the software has to be accessible. Here any value in the range of *1...65535* is possible, but it has to be noticed that a port can be used only once per IP. So for an example if there is already a webserver running at a system, port number *80* cannot be used. Therefore it is recommended to use port numbers *>1024*. They are reserved for custom usage.

On the other hand the application that controls the software using the *SCAPS.ScSamlightClientCtrl* client control interface has to open its connection to the server socket configured here using ScOpenTCPConnection(). If the plain ASCII communication has to be used, opening of a TCP/IP connection depends on the operating system and the programming language. In C the appropriate function calls to do that mainly are *socket()* and *connect()*, for sending data *send()* is used and *recv()* for the reception of an answer.

The network access to the scanner application can be restricted to some specific IPs to avoid that connection attempts are successful from hosts that are not allowed to control the application. These IPs can be defined using the two files *hosts.allow* and *hosts.deny* that are located within *<SCAPS>\system* of the installation directory of the scanner application. Both files expect IP numbers in the format "*WWWW.XXXX.YYYY.ZZZ*".

Within *hosts.allow* all the IPs that are allowed to connect to the scanner application can be listed. If there are some IP numbers define here, connections are allowed only from these ones exclusively. If there are no IPs defined, connections are allowed from all IPs except the ones that are listed within *hosts.deny*. The file *hosts.deny* can contain a list of IP numbers with the format "*WWWW.XXXX.YYYY.ZZZ*" and defines which hosts are not allowed to control the application. If there a no IPs listed within this file, no hosts are forbidden explicitly to connect to the scanner application.

Script Settings: Besides the possibility to have an external application that is under the control of the user it is also possible to start such an application from within the scanner application. To do that the following settings are used:

Menutitle: Specify the name which is shown in the *Special* entry of the Menu bar. This name is used to address the application that has to be started out of SAMLight.

Path to Script or Executable: This script or *.exe is executed whenever the related name in the *Special* entry in the menu bar is selected. The following programs are supported:

- *Executables* (*.exe) that are started and that should access the *Client Control Interface* of the application
- *Client Control Scripts* (*.ccs) that contain ASCII-CCI-commands that are interpreted and executed. These scripts may contain comments using # as a delimiter.

After these scripts are executed within the context of the scanner application there is no need to perform any initialization or opening of a connection. The commands to control the application can be used directly.

5.8 IO

The settings dialog described here can be reached by selecting the menu item *Settings → System → IO*.

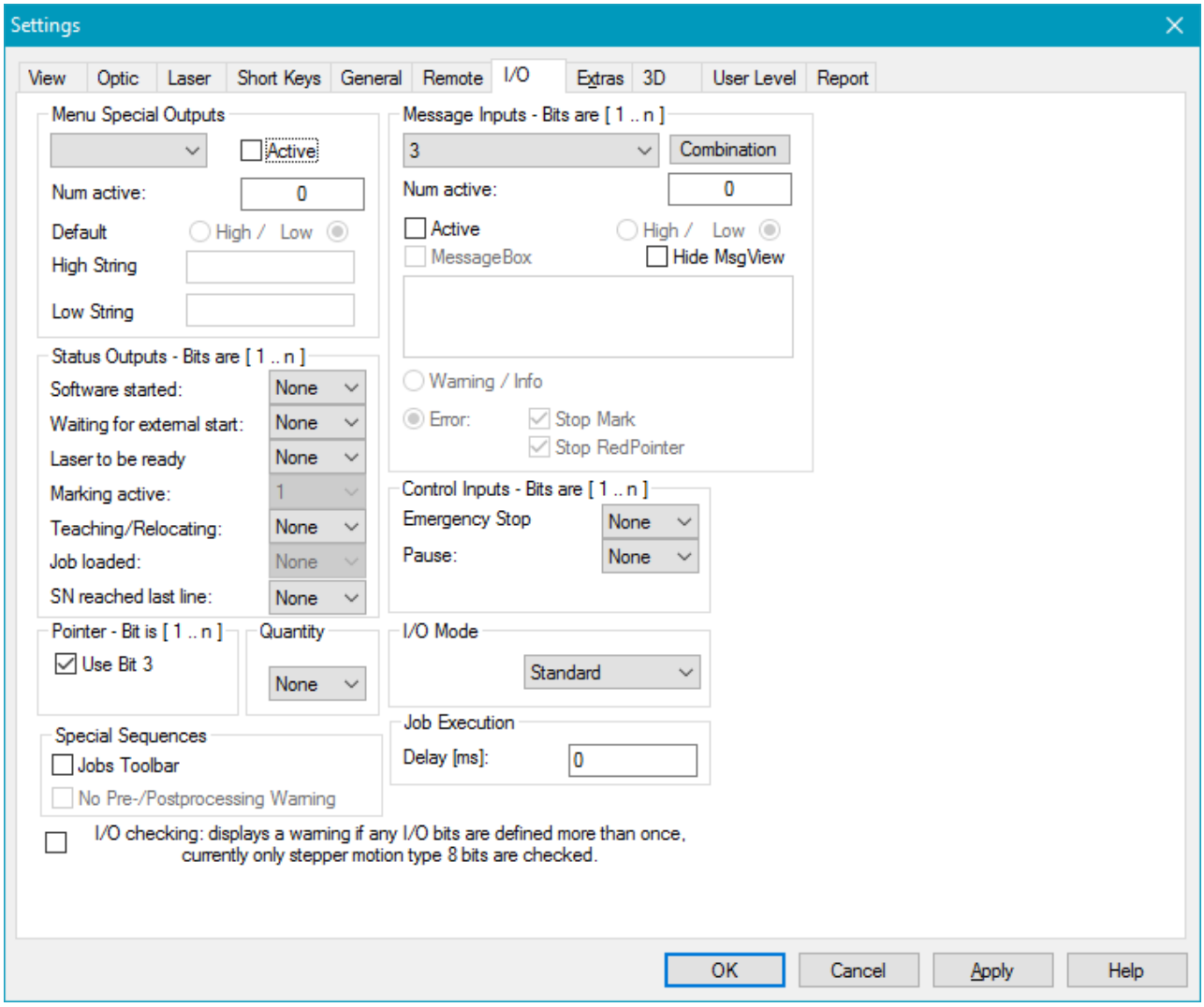


Figure 47: IO Settings Dialog

Menu Special Outputs: It is possible to insert new menu items for switching IOs on and off. The bits of the IO port which will be controlled from special menu points can be defined in the dialog. If one bit is selected and the *Active* checkbox is enabled a string (*Num Active*) which indicates the current state of the bit can be defined. Under the following menu the items for switching the IOs on and off will be inserted.

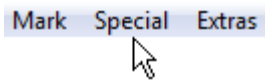


Figure 48: Mark Special Outputs Menu

Default: Defines the default state of the special output that is shown in the menu.

High String: Defines the String that is shown when the special output is set to Low.

Low String: Defines the String that is shown when the special output is set to High.

Status Outputs: This block defines state IOs that can be switched on an off according to specific program and usage actions. Using the combo boxes it is possible to assign a special output pin for such an action. Using this functionality an integration of external equipment can be done.

Software started: The selected bit is set to *high* as long as the software is running.

Waiting for external start: Set to *high* if in trigger mode. This also stays *high* during marking.

Laser to be ready: Set to *high* if trigger or mark dialog is open.

Marking active: Set to *high* during marking of a job. Only for USC cards this signal is hardware controlled. For RTC cards this signal is software polled with the result of jitter. For proper I/O handling, this signal has to be used, not the RTC busyout.

Teaching / Reloacting: If an output pin is defined here it is set to *high* on every time the teaching or relocating dialog is active.

Job loaded: The selected bit is set to *high* if a non empty job is loaded in the View.

SN reached last line: If a file is assigned to a serial id and the last line of this file has been reached this bit is set. To change the state afterwards you have to reset the serial id or load a new job. Instead or in addition you can pop up a message box if the last line of the file has been marked.

Pointer:

Use Bit 3: Bit 3 of the IO port is used to indicate that the red pointer is active.

Invert: This option is only available for RTC cards and USC-1. It inverts bit 3 for controlling the red pointer.

Quantity: Here an output can be defined that goes *high* when a predefined number of mark quantities has been reached. The predefined number of mark quantities can be set up in Mark→Counter→Set Quantities.

Special Sequences: This part of the settings panel handles the special sequences that can be executed during program startup and exit. To avoid collisions between an job externally selected this option is available only in I/O-Mode *Standard*. By default, these special jobs and together with them the Special Jobs Toolbar are disabled.

Jobs Toolbar: Enables the Special Sequences Toolbar and the related functionality that allows it to execute jobs at program startup and / or exit.

No Pre-/Postprocessing Warning: If this box is selected, this special security warning is disabled and all special jobs (except the mainjob) are executed immediately and with no separate user interaction when the program is started or exited. Please handle with care! If this option is used it has to be secured by the user that nobody can be injured by potentially dangerous pre- or postprocessing jobs.

Message Inputs: Input bits can be used to cause a message output. The selected bit must be activated to send the defined message by the *Active* check box. The message appears if the selected bit is either *high (H)* or *low (L)* according to the selected radio button. An error or a warning message is displayed in the message view dialog as well as in the statusbar. If *Hide MsgView* is activated, the message view is invisible.

RTC input bits:

- 0 .. 15 (Extension 1 connector, Digital_In 0 .. 15)

USC input bits:

- 3 .. 6 (37-pin connector, Opto_In2 .. 5)

USC-2/3 additions:

- 7 .. 16 (Extension connector, Digi_In0 .. 9)
- Status #0 .. #19: These status bits refer to the 20-bit word of the XY2-100 feedback from the scan head. The user may define a unique message / error for any of these bits.
- Error warning info. These warnings can be reset in the status bar.
  - Out of field (OF) is set when a MOTF overflow occured.
  - Out of data with laser on (OD) is set when a buffer underrun occured.

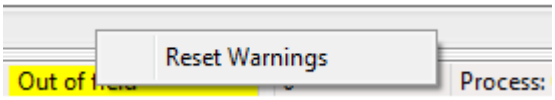


Figure 49: Reset OF and OD by right-click on the warning followed by a left-click on Reset Warnings

Num active: Counts the number of activated Message Inputs.

Combination: This button opens a dialog *Message Input Combination*, which can be used to combine two or three input bits.

Control Inputs:

Emergency Stop: If an input is selected here it is handled as a watch for an emergency stop condition. That means if the selected input goes to low all marking operations are stopped and a special emergency stop dialog is displayed. This dialog blocks all other operations and stays in front of the screen until normal operation is resumed by pressing the "Resume Operation" button. This button becomes active and can be pressed only after the selected emergency stop input goes to high. When the resume button is pressed the application is brought back into its initial state. That means the connected motion controllers are driven to their home position automatically before the emergency stop dialog disappears and before a user can continue with normal operation.

It is recommended to connect the appropriate input pin before this option is enabled. An open input normally is recognized as a low-signal so that leaving the IO settings dialog would put the application in the emergency stop state immediately.

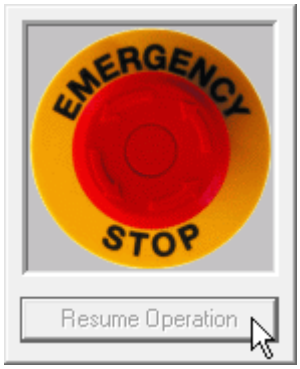


Figure 50: Emergency Stop

Pause: This functionality is only available with an USC-2 card. Here you can select an input bit that pause the job. The laser is switched off after a mark command, after a PolyEnd or during a jump command.

I/O Mode: There are three SAMLight IO modes (which differ from the Flash JobIOSelection mode) that can be chosen:

- **Standard**: the settings described above can be made including the freely definable Message Inputs.
- **SAMLight JobIOSelect**: these inputs are disabled. Here, the OptoIn_2...OptoIn_5 input signals are used to select and load jobs (up to 15) externally triggered.
- **SAMLight JobIOSelect Ext**: the Digital Input Pins DigiIn_0...DigiIn_7 of the USC-2/-3 card to select the jobs (up to 255).

For more information see SAMLight Job IO Selection.

Job Execution:

Delay [ms]: Defines an execution delay which is the time between a mark output signal is given and the execution. Only for USC cards the marking active signal is hardware controlled. For RTC cards the marking active signal is software polled with the result of jitter.

Input and Output bit values:

USC cards:

Input pin	Output pin	Bit value
Opto-insulated Inputs and Outputs		
OptoIn_0 [a]	OptoOut_0 [c]	1
OptoIn_1 [b]	OptoOut_1	2
OptoIn_2	OptoOut_2 [d]	3
OptoIn_3	OptoOut_3	4
OptoIn_4	OptoOut_4	5
OptoIn_5	OptoOut_5	6
Following pins are only available for USC-2/3		
Digital Inputs and Outputs		
DigiIn_0	DigiOut_0	7
DigiIn_1	DigiOut_1	8
DigiIn_2	DigiOut_2	9
DigiIn_3	DigiOut_3	10
DigiIn_4	DigiOut_4	11
DigiIn_5	DigiOut_5	12
DigiIn_6	DigiOut_6	13
DigiIn_7	DigiOut_7	14
DigiIn_8	DigiOut_8	15
DigiIn_9	DigiOut_9	16
Stepper Inputs and Outputs		
SmIn_0	SmOut_0	17
SmIn_1	SmOut_1	18
SmIn_2	SmOut_2	19
	SmOut_3	20
	SmOut_4	21
	SmOut_5	22

Table 6: I/O bit values for USC cards

RTC cards:

Input pin	Output pin	Bit value
Digital Inputs and Outputs		
DigiIn_0	DigiOut_0	1
DigiIn_1	DigiOut_1	2
DigiIn_2	DigiOut_2	3
DigiIn_3	DigiOut_3	4
DigiIn_4	DigiOut_4	5
DigiIn_5	DigiOut_5	6
DigiIn_6	DigiOut_6	7
DigiIn_7	DigiOut_7	8
DigiIn_8	DigiOut_8	9
DigiIn_9	DigiOut_9	10
DigiIn_10	DigiOut_10	11
DigiIn_11	DigiOut_11	12
DigiIn_12	DigiOut_12	13
DigiIn_13	DigiOut_13	14
DigiIn_14	DigiOut_14	15
DigiIn_15	DigiOut_15	16

Table 7: I/O bit value for RTC cards

[a]: Reserved for trigger start

[b]: Reserved for external stop

[c]: Reserved for marking active

[d]: Only reserved for the red pointer, if the red pointer is active

5.8.1 Message Input Combination

Combine Inputs: The input bits intended to be combined are selected via the drop down menus (Input bit A, Input bit B, Input bit C). For a combination of only two bits via the first two drop down menus, chose -1 for Input bit C. After clicking on *Combine Inputs*, a combination of the selected bits will appear in the lower drop down menu and in the List of all Message Input bits.

The combination of two bits creates four options (*HH, HL, LH, LL*), the combination of three bits creates eight options, respectively. Each bit can only be used once, either in one single combination or directly without a combination.

Remove Combination: In order to cancel an existing combination, select the combination in the drop down menu and click on *Remove Combination*. Please note that this action is irreversible!

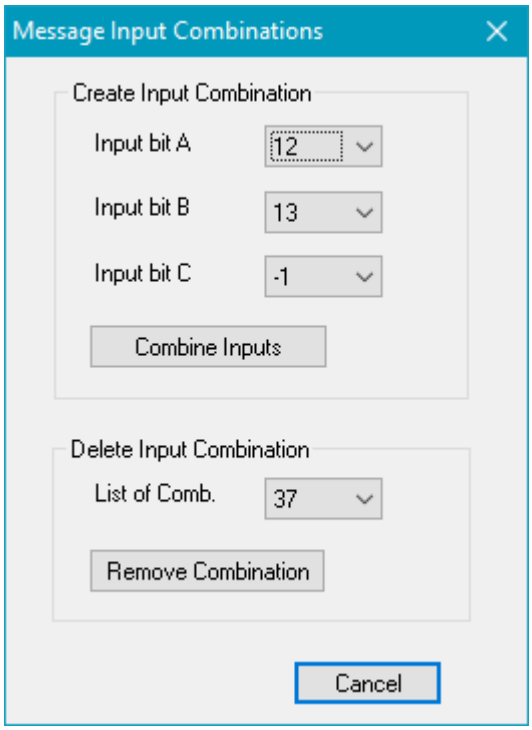


Figure 51: Message Input Combinations Dialog

5.9 Extras

The settings dialog described here can be reached by selecting the menu item *Settings → System → Extras*.

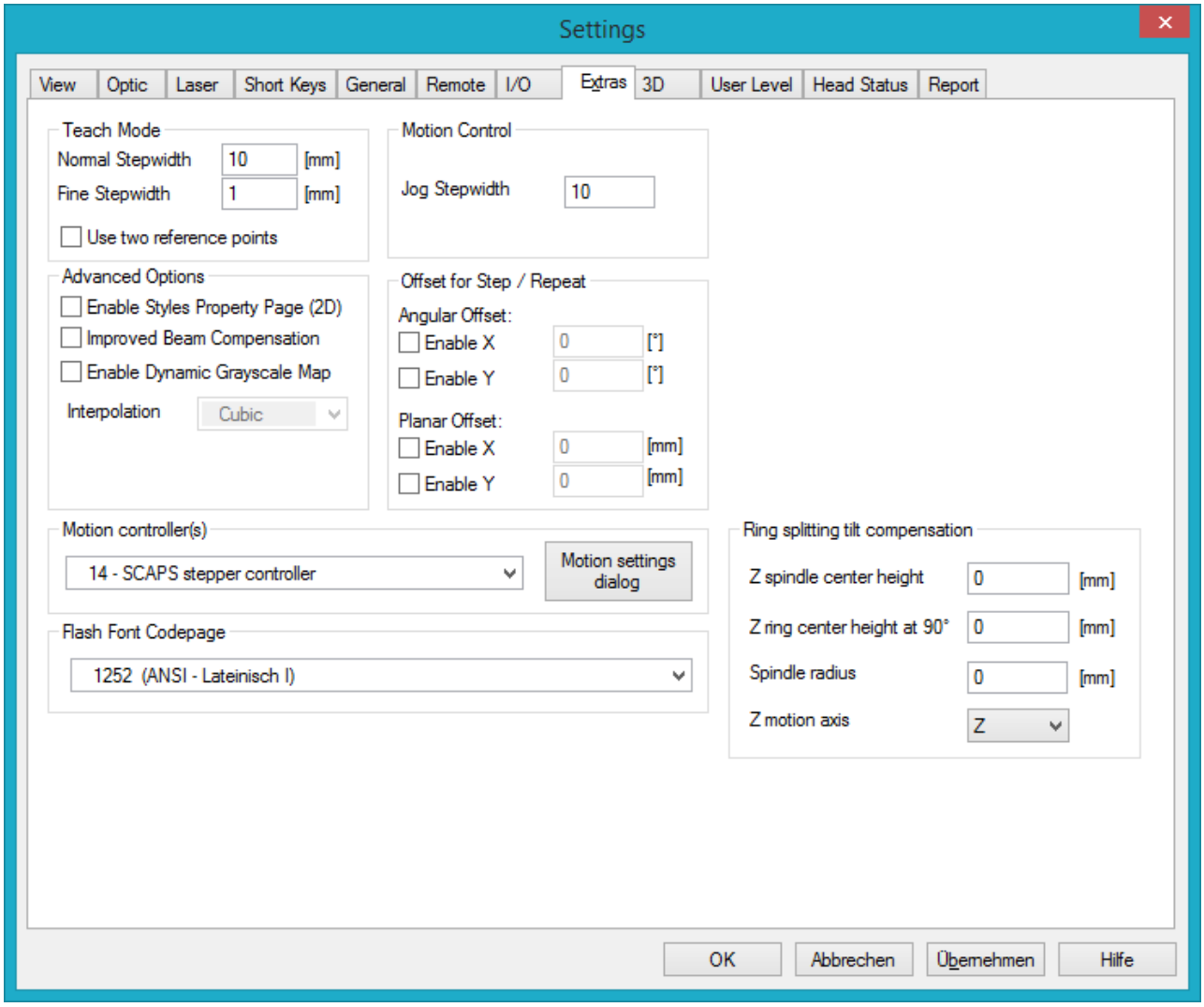


Figure 52: Extras Settings Dialog

Teach Mode: This block is related to the teaching / relocating mode. Using this mode it is possible to teach reference positions for a job that are related to a specific work piece. When this work piece was exchanged and the new one has a different position and / or rotation angle comparing to the preceding one, it is possible to modify the job so that it fits to the new position. To do that, the relocating function can be used. The parameters that can be defined here influence the teaching / relocation behavior in following ways.

Normal Stepwidth: Defines the normal stepwidth that is used in the teaching / relocating dialog to move the laser pointer

Fine Stepwidth: Defines the smaller, more exact stepwidth that is used in the teaching / relocating dialog to move the laser pointer

Use two reference points: When this box is checked, the relocation can be done using two reference points. With one point it is possible to equalize a work pieces translation in parallel to the preceding position only. When two reference points are used, a rotation can be calculated too. The new position of the work piece doesn't need to be exactly parallel to the preceding one.

An output pin that is toggled every time the teaching / relocating dialog is opened and closed can be defined in IO Settings (e.g. switch a camera on and off).

Motion Control: This area is related to motion control elements.

Jog Stepwidth: Default value for the stepwidth that is used for the Jog movements.

Advanced Options:

Enable Styles property page (2D): Enables the 2D styles property page.

Use improved BC: Activates improved Beam Compensation Algorithm for Hatching and Beam Comped Copy.

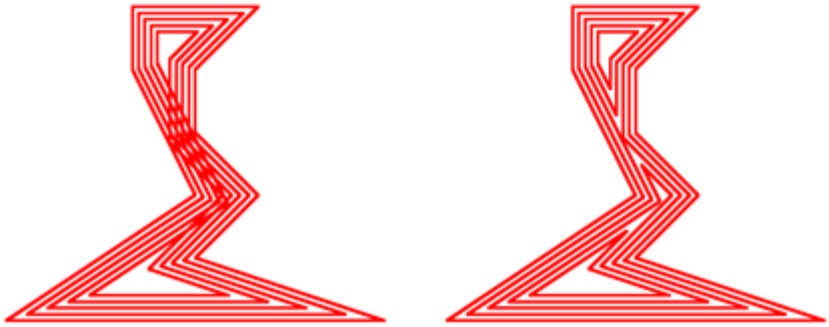


Figure 53: Left: old BC Algorithm, right: improved BC Algorithm

Enable Dynamic Grayscale Map: Activates a dynamic System PixelMap. Three different modes of interpolation are available in the drop down menu: Linear, Cubic or Hermite.

Offset for Step / Repeat: Defines a global step / repeat offset.

Motion Type settings: Here you can choose the motion type which is corresponding to the value in the file ***sc_motion_settings.txt***. Please refer to the chapter Motion Controller for further information. The motion settings dialog is coming soon. If you choose entry 0 here the motion control will be deactivated. For motion type 8 = Stepper motor there is a motion settings GUI which can be accessed by the button "Motion settings dialog".

Flash Font Codepage: Here you can choose which font codepage should be saved to the flash of a USC-2 card. This is necessary if you want to change the dynamic text of a serial number (or barcode, DateTime etc.) on the flash.

Ring splitting tilt compensation: This feature is to be used with Ring Splitting.Here you can type in the key value of your equipment so that SAMLight can calculate the correct compensation.

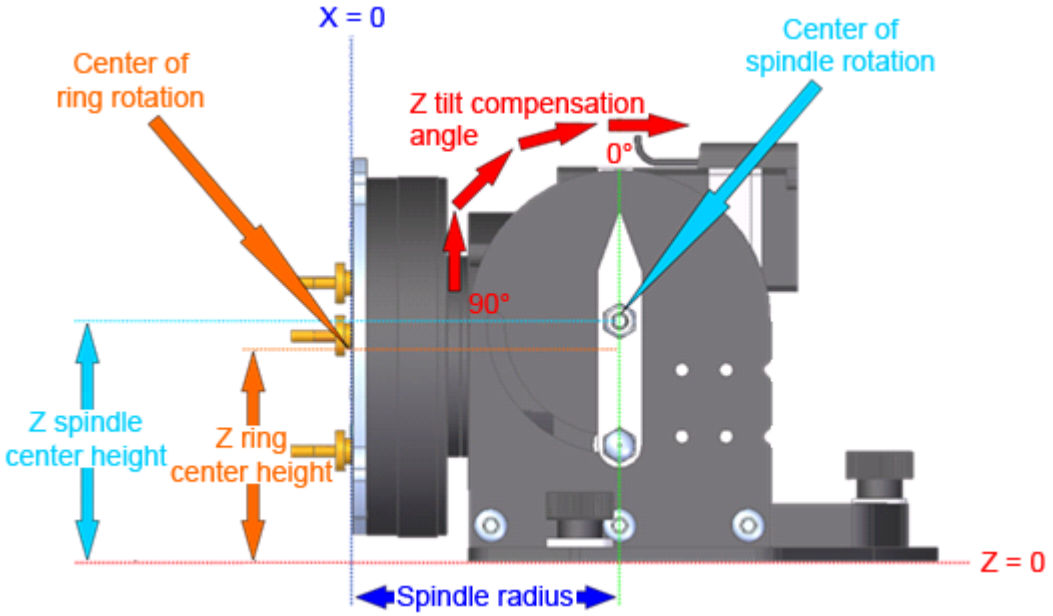


Figure 54: Ring splitting tilt compensation

5.10 3D

This option is only available for SAM software with the SAM3D option.

The following dialog can be reached by Menu item *Settings → System → 3D* and covers several 3D marking functionalities like they are useful e.g. for rapid prototyping.

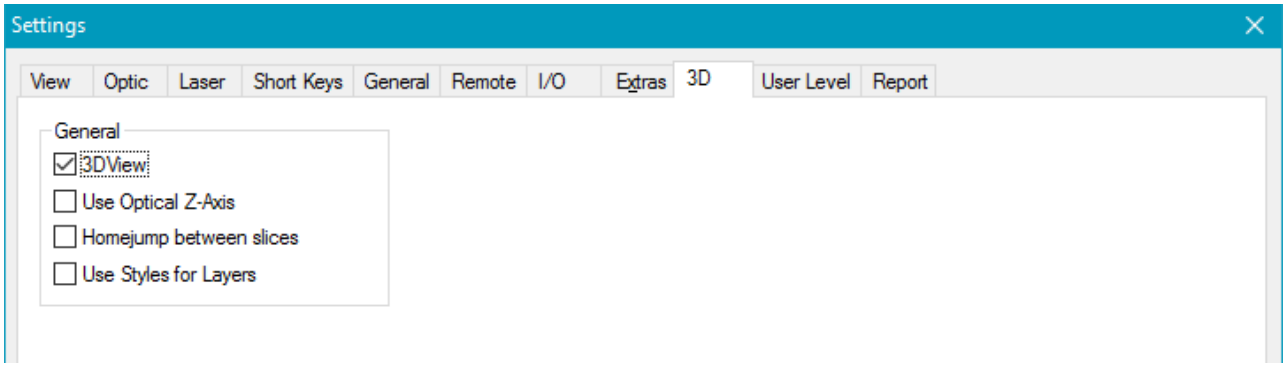


Figure 55: 3D Settings Dialog

General:

3DView: Enables / Disables SAM3D mode. The software must be restarted for the change to take effect.

Use Optical Z-Axis: Check this option, when you want to shift the focus optically with a 3D scan head.

Home jump between slices: Performs a home jump after each marked slice. If it is checked and *Settings → System → Optic → Home Position* is disabled, there will not be a home jump, but the laser power of the HomeJumpStyle is set after each marked slice.

Use Styles for Layers: Activates the Styles Property Page for SAM3D.

5.11 User Level

The following dialog can be reached by Menu item *Settings → System → User Level*.

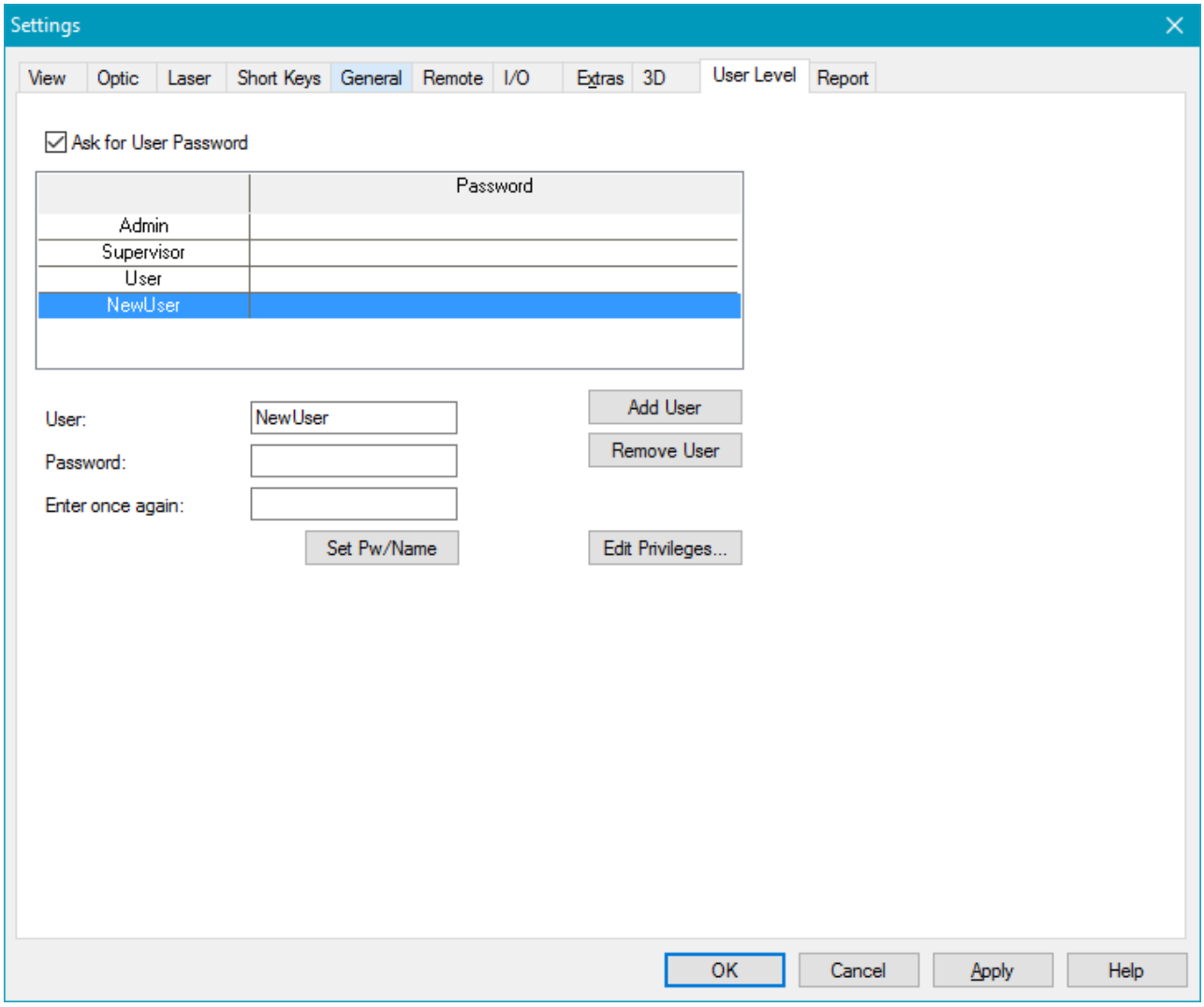


Figure 56: User Level Settings Dialog

The following entries are possible only if the right for password assignment is given, see Access Rights.

Ask for User Password: If checked the user is asked for the user name and the password before the software starts. If it is not checked the software starts with full functionality.

User: Displays the selected user.

Password / Enter once again: The password needs to be entered twice.

Set Pw/Name: Applies the password that is entered in the edit field to the selected user and changes the name of the user if wanted.

Add User: Adds a new User to the list.

Remove User: Removes User from the list.

Edit Privileges: Opens a dialog where it is possible to for define access user rights.

If a user has no password this user will be a default user. For a login with an invalid password the default user is taken. If more than one default user exists, the first one from the list is taken.

5.11.1 Access Rights

The following dialog opens by clicking on the Edit Privileges button on the User Level page.

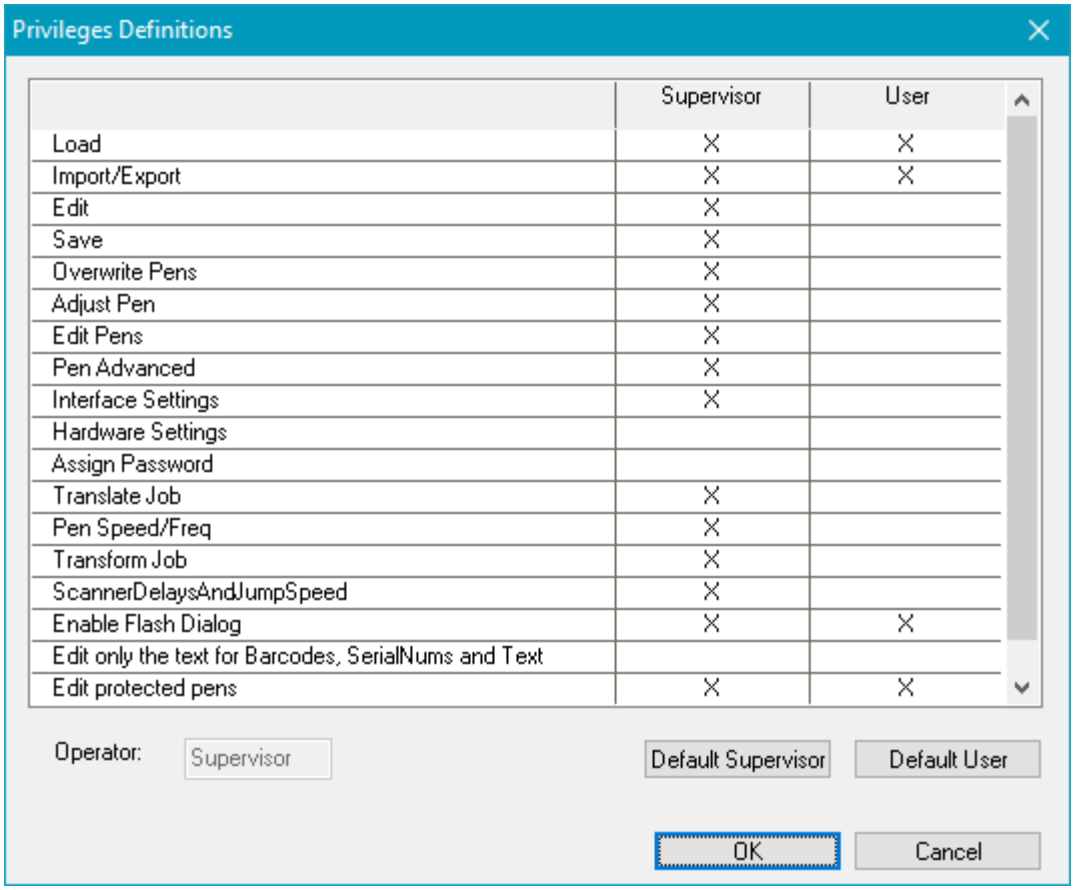


Figure 57: Access Rights Dialog

The features can be enabled or disabled by clicking on the table fields.

Default Supervisor: Sets the supervisors default settings to the selected operator.

Default User: Sets the users default settings to the selected operator.

Access Rights:

Load: Allows to load job files

Import/Export: Allows import and export.

Edit: Allows to setup and edit a job.

Save: Allows to save a job.

Overwrite Pens: Allows to overwrite pens.

Adjust Pen: Allows to set a pen to an object.

Edit Pens: Allows to edit pens.

Mark Advanced: Enables the mark advanced button.

Interface Settings: Enables following property pages of *Settings → System*: View, ShortKeys, General, Extras

Hardware Settings: Enables following property pages of *Settings → System*: Optic, Laser, IO-Settings, Remote

Assign Password: Enables property page User Level of *Settings → System*.

Translate Job: Allows to translate entities even if the Edit privilege is deactivated.

Pen Speed/Freq: Allows to modify the Speed and the Frequency of the current pen.

Transform Job: Allows to Translate, Scale, Mirror and Rotate entities even if the Edit privilege is deactivated.

ScannerDelaysAndJumpSpeed: Allows to change Scanner Delays and Jump Speed if Pen Speed/ Freq. is activated also.

Edit only the text for Barcodes, SerialNums and Text: Allows to edit the text in the property page Text2D but not the rest of the parameters on this page.

Edit protected pens: Allows to change the status of Protected Pens.

5.12 Card

These settings for RTC cards are described in section Card Settings.

5.13 Trigger

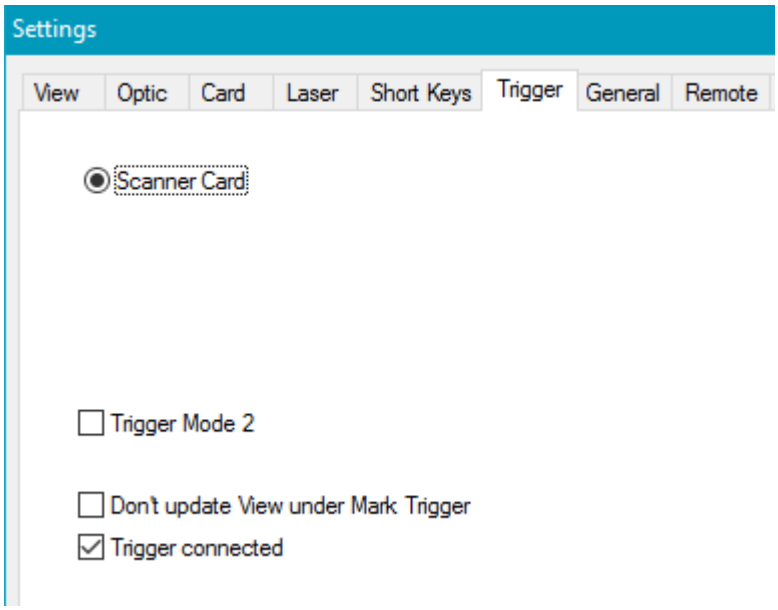


Figure 58: Trigger Dialog

Trigger Mode 2: Serial numbers and Date Time objects are not being update between marking by trigger.

This allows shorter delay between triggering.

Don't update View under Mark Trigger: The view area will not be updated while the mark trigger dialog is open.

Trigger connected: This flag allows to use the trigger mode.

This special trigger mode is only available for RTC cards.

5.14 Report

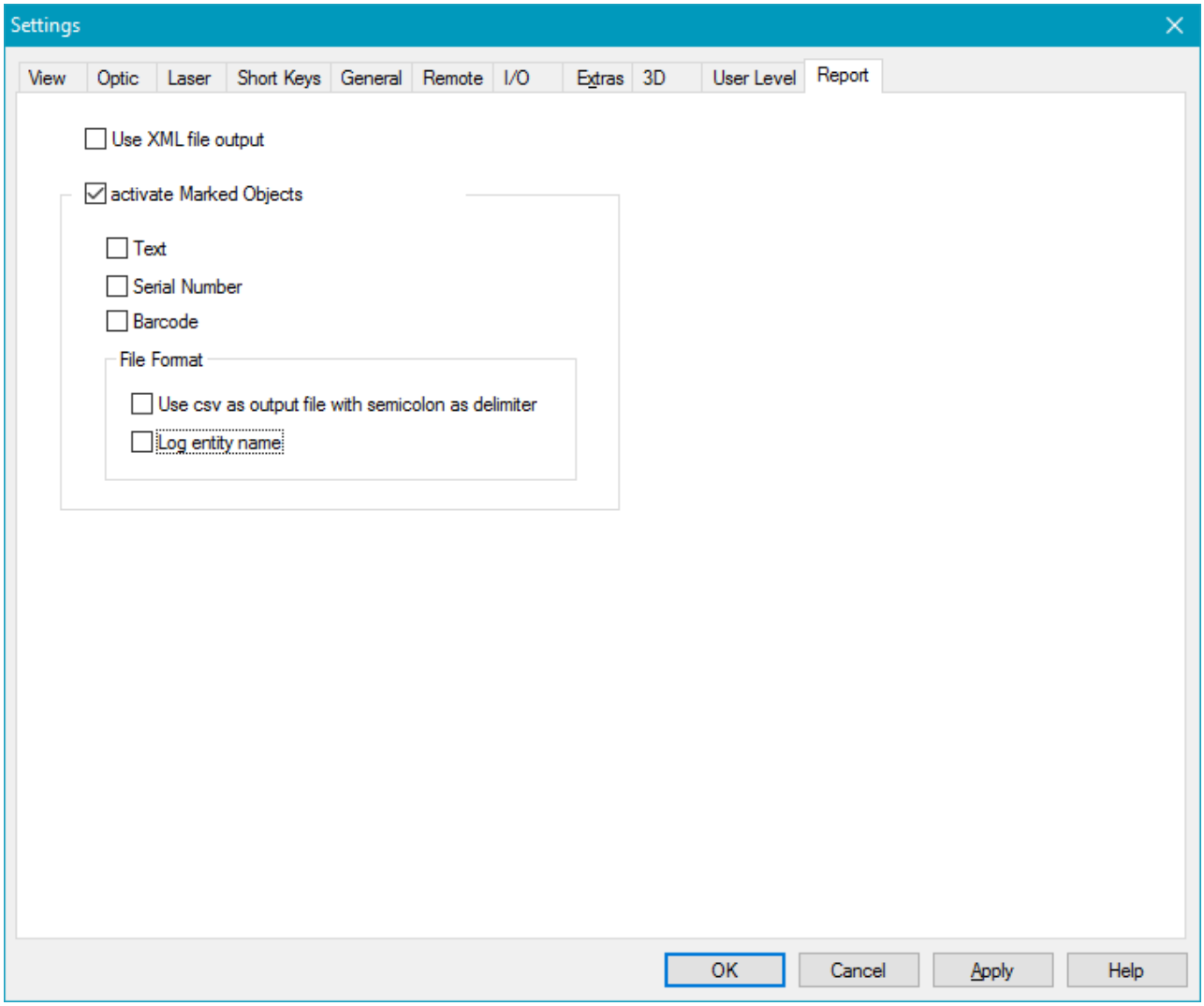


Figure 59: Report Dialog

activate: Activate the report file output. If active each marking produces an output to a file named <Date>_<JobPath>.txt or .cvs in which the marked objects are listed. The file will be located at <SCAPS> \reportfiles.

Text, Serial Number, Barcode: Choose which entities should be included in the report file.

File Format: If checked the output file will be in csv-format if not it will be a simple txt-file.

Log entity name: If checked the entity name will be additionally included in the report file.

6 Pen Settings

Each entity is assigned to a pen. This is by default pen 1. To be visualized on the screen each pen can have a different color. Where to set the color, see in chapter View. The mark property page seen below can be used to assign a pen to the selected entity. The parameters are different for YAG and CO2 lasers.

Pen 255 is used for the red pointer.

Pen 256 is used for the home jump. When the scanner moves to the home position (usually after executing) pen 256 is used. For example this feature can be used to switch off the lamp current after marking. See power control of pen. Pen 256 can be edited by clicking Advanced... → HomeJumpStyle (Pen #256).

There are several possibilities to save and load the pen settings:

Save pens into SAMLight settings file (*.sam)	Load pens from SAMLight settings file (*.sam)
• Save pens on SAMLight exit. Define in 'Settings → System → General → Files → Save Pens on exit' if the pens should be saved when closing SAMLight. • Save pens directly via 'Settings → System → General → Files → Save Pens Now'.	• Load pens on SAMLight start. • Load pens on 'File → New'. Define in 'Settings → System → General → Files → load pens on file new' if the current pens should be reverted.
Save pens into SAMLight job file (*.sjf)	**Load pens from SAMLight job file (*.sjf)**
• Save a job directly via 'File → Save (as)'. Define in 'Settings → System → General → Job Save/Load Dialog' the default state of the 'Pens' checkbox. • Save a job via 'save job after each mark' if the current job already contains the pen settings. • Save a job via 'auto save on exit' if the current job already contains the pen settings.	• Load a job directly via 'File → Load'. Define in 'Settings → System → General → Job Save/Load Dialog' the default state of the 'Pens' checkbox. • Load a job via 'File → RecentJobs'. Define in 'Settings → System → General → Recent Jobs → Load pens' if the job pens should be loaded as well. • Load a job via 'JobAutoLoad'. Define in 'Settings → System → General → Job Auto Load → Load pens' if the job pens should be loaded as well.

Table 8: Different possibilities to save and load pens

In general, the Pen Settings are saved in <SCAPS>\system\sc_light_settings.sam if Save Pens On Exit is enabled or "Save Pens Now" is used.
But, the Pen Settings can also be saved within a job (.sjf). Be careful not to load any Pen Settings with a job and then accidentally overwrite your global Pen Settings in sc_light_settings.sam via "Save Pens On Exit".*

It is highly recommended to backup sc_light_settings.sam every time after having done any important or time consuming configuration of the Pen Settings.

Apply pens to hatches, LineArrays or PolyLines:

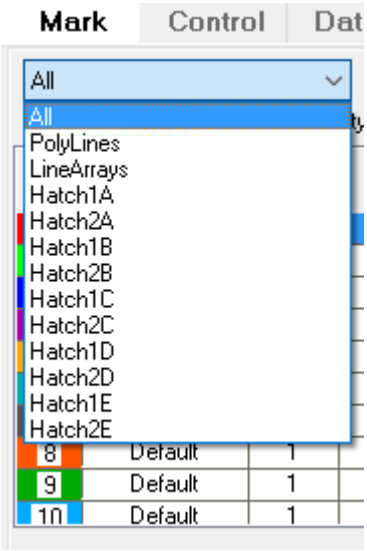


Figure 60: Drop-down menu to apply a pen for hatches, LineArrays and PolyLines

For selected entities a pen can be applied to each of 10 available hatches, LineArrays and PolyLines. The currently applied pen for an entry of the drop-down menu will be highlighted when selected. To reset this settings select 'All' and apply a pen. For complex configurations we recommend the styles feature.

YAG laser:

Figure 61: Mark property page with Pen Settings for YAG Laser

Pen: pens are listed by number

Name: Name of the pen

Speed: Galvo mark speed in mm/s.

Power: Laser power in Watt. For calibration see chapter PowerMap.

Frequency: Frequency in kHz.

The grid is sortable through clicks on the respective column.

Override [%]: The override factors can be used to increase or decrease all values for all pens during mark process. The pen itself will not be changed.

Edit...: Press the edit button to define the settings of the currently selected pen. Alternatively, doubleclick on any field in the line of the selected pen (blue in Fig. **61**).

Advanced...: Within the advanced dialog settings for the PowerMap, the HomeJumpStyle and the System PixelMap can be done.

Apply: Applies the selected pen to the current selected object.

CO2 laser:

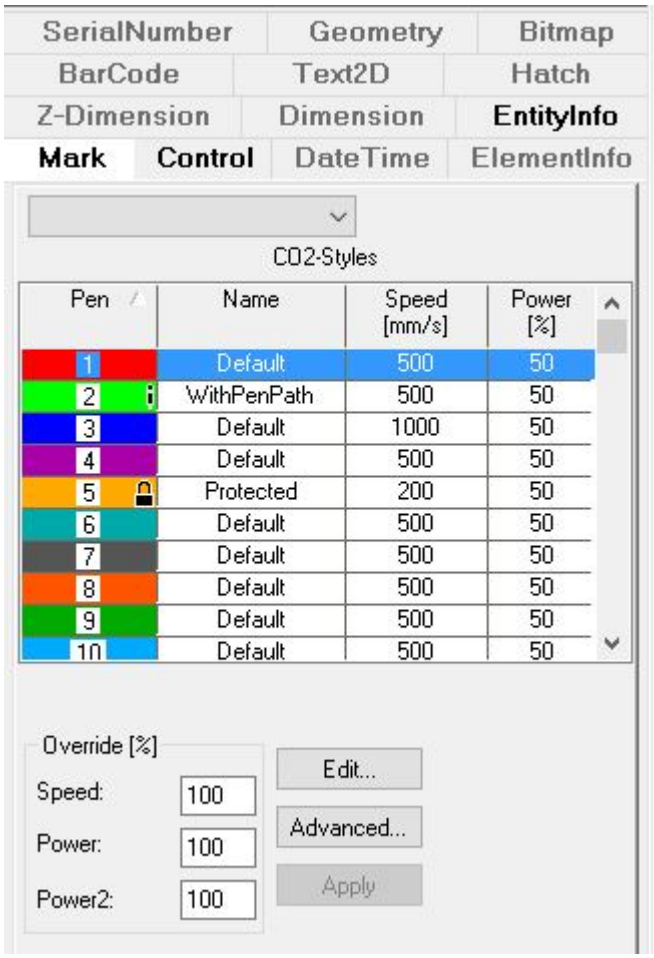


Figure 62: Mark property page with Pen Settings for CO2 Laser

Pen: pens are listed by number

Name: Name of the pen

Speed: Galvo mark speed in mm/s.

Power1: LaserPower1 signal in percentage of period.

Power2: LaserPower2 signal in percentage of period.

The grid is sortable through clicks on the respective column.

Override [%]: The override factors can be used to increase or decrease all values for all pens during mark process. The pen itself will not be changed.

Edit...: Press the edit button to define the settings of the currently selected pen. Alternatively, doubleclick on any field in the line of the selected pen (blue in Fig. **62**).

Advanced...: Within the advanced dialog settings for the PowerMap, the HomeJumpStyle and the System PixelMap can be done.

Apply: Applies the selected pen to the current selected object.

6.1 Edit Pens

Each single pen can be edited by pressing the *Edit...* button in the *mark* property page (Alternatively, doubleclick on any field in the line of the selected pen). The pen that has to be set is selectable in the combobox. Also some variables provide *All* buttons to apply the setting to all pens.

For YAG lasers, the following dialog appears:

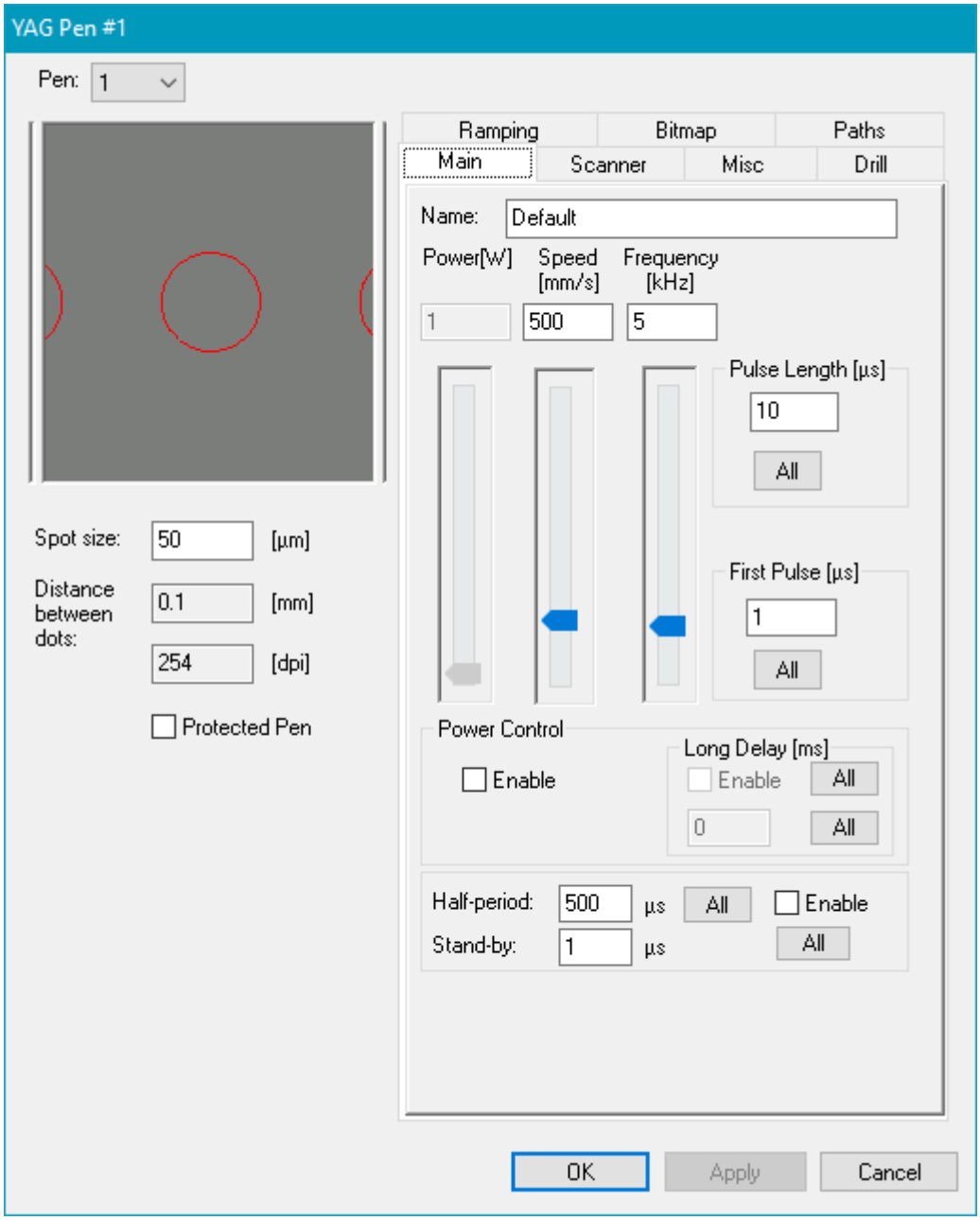


Figure 63: Pen Settings Dialog for YAG Laser

Pen: The drop down menu can be used to switch between pens.

Spot size: This value is used for the display of the spots.

Distance between dots: Gives information on the distance between pixels in bitmap mode.

Protected Pen: When activated, the pen is protected. Thus, the parameters cannot be changed any more (also not by the application of an 'All' button in any other pen). It will be possible by user level to deny editing

pens for specified user groups. This protected flag is only stored in the SAMLight settings and not in the pen settings. That means it won't be exported via pens included in a job file. If a job with included pens is loaded, a protected pen will be overwritten anyway. It is possible to overwrite a protected pen by CCI.

For further information on the Main tab, see Main Settings for Pens.

For CO2 lasers, the following dialog appears:

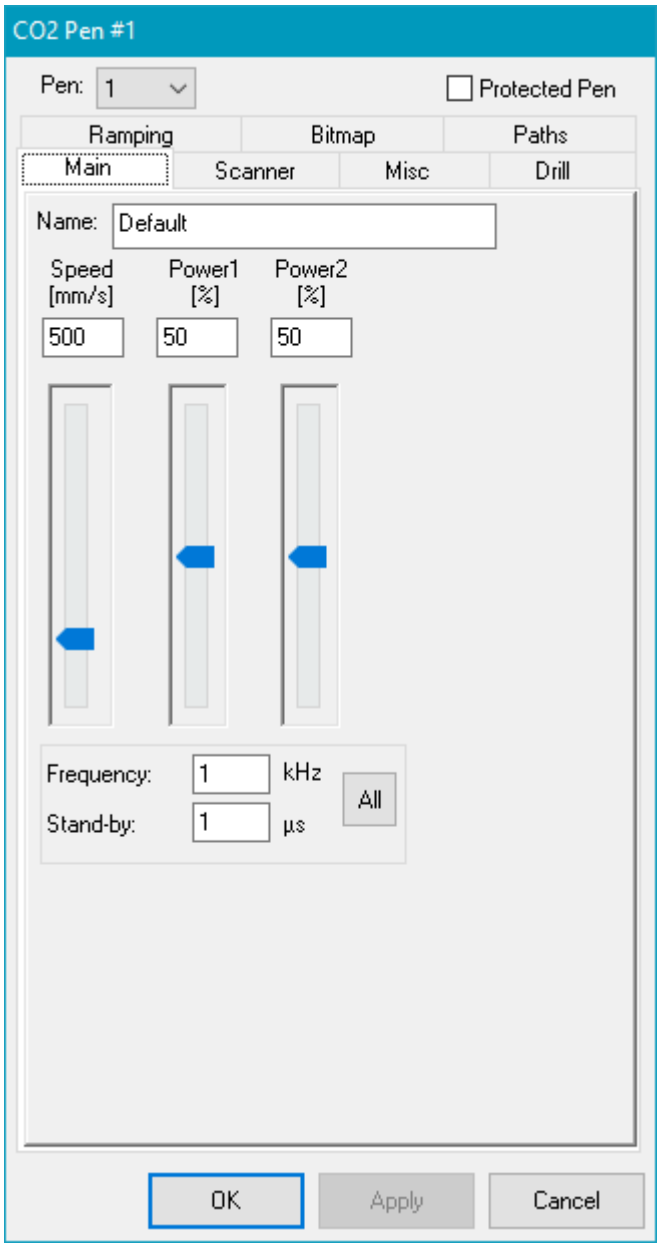


Figure 64: Pen Settings Dialog for CO2 Laser

6.1.1 Main Settings for Pens

The following page can be found under *Mark → Edit... → Main*.

YAG laser Pen → Main settings:

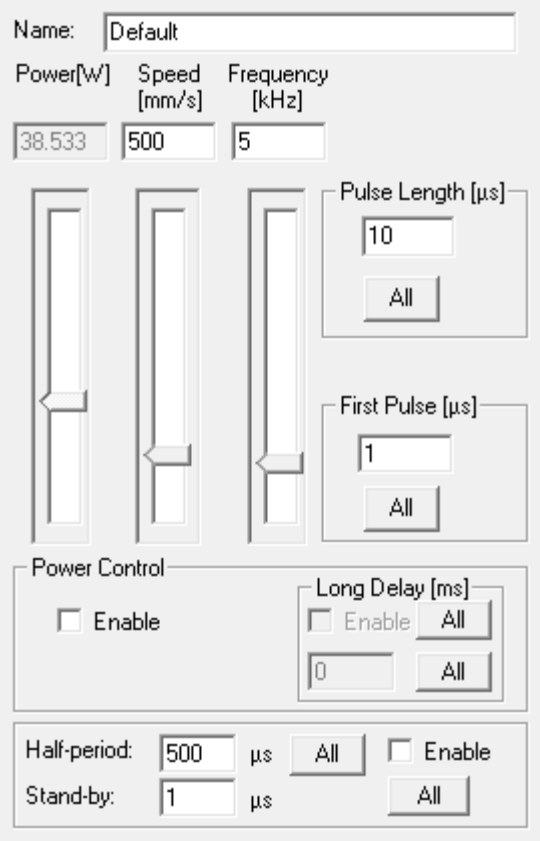


Figure 65: Main Pen Settings for YAG Laser

Power: Power of the laser lamp for selected pen. To redefine the power Power Control has to be enabled.

Speed: Marking speed of the selected pen. Min and max values are defined in the optic settings.

Frequency: Q-Switch frequency of the laser pulses. Min and max values are defined in the optic settings.

Pulse Length: Q-Switch length in µs.

If pulse length ≥ 1/frequency, a red notice 'cw' will appear to indicate continuous mode.

First Pulse: First pulse killer length in µs. Min and max values are defined in the optic settings.

Power Control: Enable or disable laser power and power control for this style.

> **Long Delay:** If enabled and the power is changing, the system will wait the indicated period before it will continue.

S**tandby:**

> **Half-period:** Half of the laser pulse period for standby mode.
>
> **Stand-by:** Q-Switch length in µs for standby mode. The stand-by mode can be globally set in the card settings.

CO2 laser Pen → Main settings:

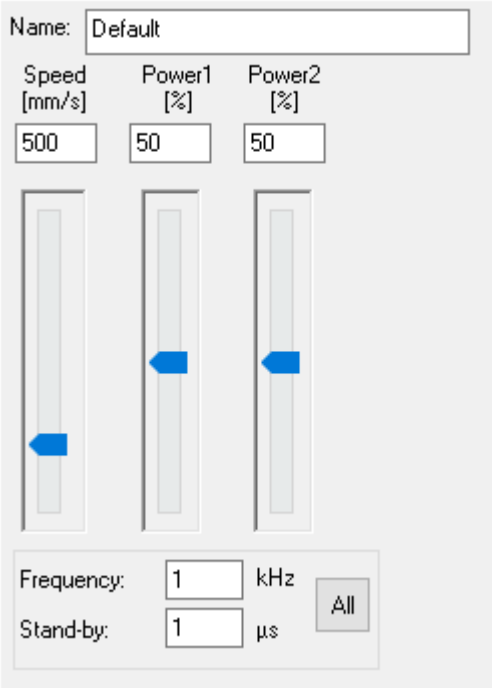


Figure 66: Main Pen Settings for CO2 Laser

Speed: Marking speed of selected pen.

Power1: Pulse length of laser signal1 in %.

Power2: Pulse length of laser signal2 in %.

Frequency: Frequency of the laser pulses.

Stand-by: Stand-by pulse length in µs for stand-by mode (for both signals identic). The stand-by mode can globally be set in optic settings for scanner card.

All: Pressing this button applies Frequency and Stand-by to all pens.

6.1.2 Scanner Settings for Pens

The following page can be found under *Mark → Edit... → Scanner*.

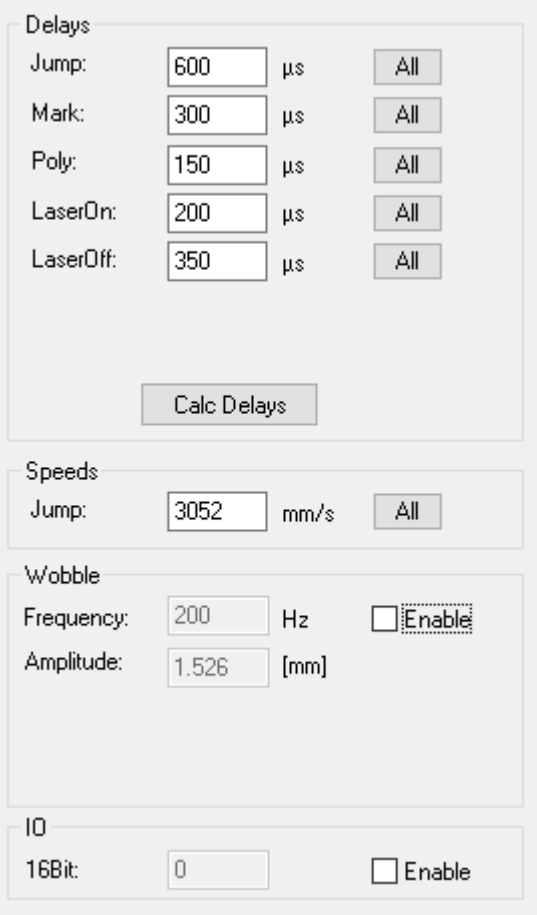


Figure 67: Scanner Delay Settings

Delays:

Jump: This delay is issued at the end of a jump.

Mark: This delay is issued at the end of a line.

Poly: This delay is issued inside a PolyLine.

LaserOn: The time the controller card waits from the beginning of the output of the first microstep before it switches on the laser. The LaserOn delay can have a negative value too. If so the Scanner moves to the start position of the vector and waits for LaserOn µs. This is the time the laser needs to start up. When the laser is ready then the scanner will begin moving.

LaserOff: The time the controller card waits from the beginning of the output of the last microstep before it switches off the laser.

All: The *All* buttons apply the related value to all 255 pens.

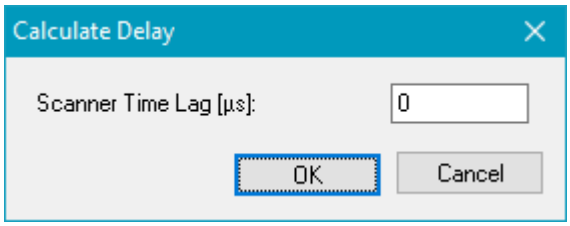


Figure 68: Calculate Delay Dialog

Calc Delays: If this button is pressed a dialog opens where the time lag value of the used scanner has to be entered (in unit microseconds). Based on this value the five delay values are recalculated. The resulting delays then can be used as a base for own optimizations. The time lag is a scanner-dependent parameter and should be found within the scanner specification.

Speeds: Jump: Jump speed of the selected pen. Min and max values are defined in the optic settings. Pen change is applied before jump.

IO: Each pen can send an 8 bit signal entered to the 8 bit port.

Delays:

NOTE

The scanner delays influence the time of the scanning process. To optimize the delays it is recommended to set the scanner delays on high values and define the laser delays first. Then the scanner delays can be optimized. Some conditions should be considered:

First delay rule:

LaserOff > LaserOn

In case of very short mark commands the mark command may end before *LaserON* command is issued and to guarantee that the *LaserOff* command is not issued before *LaserOn* command is issued, the *LaserOff* delay must be greater than the *LaserOn* delay.

Second delay rule:

Mark + LaserOn > LaserOff

In case there are two marking commands in succession, the *LaserOff* command after *mark* command 1 should be issued before the *LaserOn* command for *mark* command 2 gets issued. Therefore the *mark* delay plus the *LaserOn* delay must be longer than the *LaserOff* delay.

Delay	Effect if delay is too short	Effect if delay is too long
Jump	Oscillations could occur at the start of a vector.	No problem, just increasing mark time.
Mark	The last part of a vector is turned towards the direction of the jump vector.	No problem, just increasing mark time.
Poly	The corners of the polyline are rounded off.	Burn-in effects at the corners.
LaserOn	Burn-in effects at the beginning of a vector.	The first part of a vector is not marked.
LaserOff	The last part of a vector is not marked.	Burn-in effects at the end of a vector.

Table 9: Marking effects of scanner and laser delays

More detailed explanations can be found at Scanner and Laser delays.

6.1.2.1 Wobble Settings

Marking is possible with a wobble on top of the scanner path. This wobble can be activated in the pen settings on the Scanner tab of the pen property page. Depending on the utilized scanner card, different modes of wobble are selectable. Depending on the mode of wobble, different parameters are modifiable. In figure 69, the parameters available are explained.

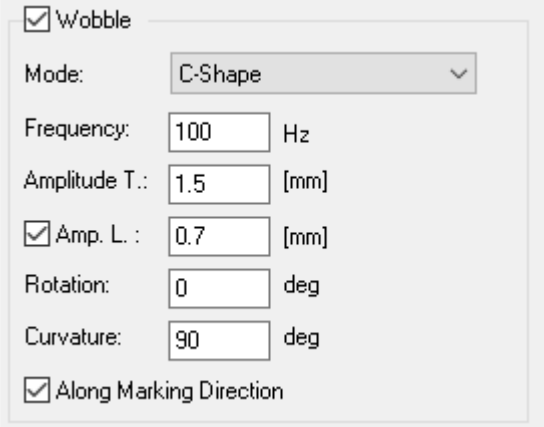


Figure 69: Wobble options on the scanner tab of the pen settings

Mode: The desired shape of the wobble is selected with the mode. Depending on the mode, some of the following parameters are not available.

Frequency: The selected shape is repeated with this frequency. The frequency range is [1, 5000] Hz.

Amplitude T.: This amplitude modifies the elongation of the shape. For some modes, this is the only parameter specifying the size. Then, the parameter is called "Amplitude". For some modes, two different amplitudes can be specified. In this case, this amplitude is transversal. The range for the amplitude is [1, 10000] bit.

Amp. L.: For some modes, two different amplitudes can be specified. In this case, this amplitude is the longitudinal one. It needs to be activated with the checkbox.

Rotation: Most modes can be rotated in respect to the default direction. The range for rotation is [-180, 180] deg.

Curvature: This parameter defines the opening angle for C-Shape mode. The range for curvature is [1, 720] deg.

Along Marking Direction: Most wobble shapes can be aligned in respect to the marking direction. If unchecked, the alignment of the wobble shape is along the X-axis.

Following modes are available for different scanner control cards:

	Circle	Sine	Ellipse	8-Shape	Double-8	Zig-Zag	C-Shape
USC-1/2, RTC3/4	✓	–	–	–	–	–	–
RTC5	✓	✓	✓	✓	–	–	–
USC-3	✓	✓	✓	✓	✓	✓	✓

Table 10: Available wobble modes for different cards. The RTC5 wobble modes have some restrictions.

Following wobble parameters are available for different wobble modes:

	Circle	Sine	Ellipse	8-Shape	Double-8	Zig-Zag	C-Shape
Frequency	✓	✓	✓	✓	✓	✓	✓
Amplitude T.	✓	✓	✓	✓	✓	✓	✓
Amp. L.	–	–	✓	✓	✓	–	✓
Rotation	–	✓	✓	✓	✓	✓	✓
Curvature	–	–	–	–	–	–	✓
Along Mark. Dir.	–	✓	✓	✓	✓	✓	✓

Table 11: Available wobble parameters for different wobble modes

For each mode of the wobble, an example of the scanner path is given in the following.

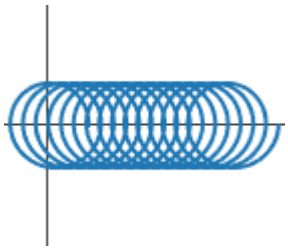


Figure 70: Circle wobble

Circle: The circle wobble can be used with all USC cards and all RTC cards. If the circle wobble is selected, all vectors and jumps are executed with this circular movement of the scanner.
Frequency and Amplitude can be changed. The amplitude defines the radius of the circle.

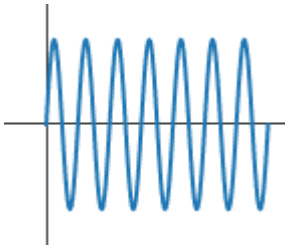


Figure 71: Sine wobble

USC-3 Sine: The sine wobble is available for the USC-3 card only.
Frequency, Amplitude and Rotation can be changed. The amplitude defines the height of the sinus. The shape can be rotated in respect to the default or marking direction (depending on the state of the flag "Along Marking Direction").

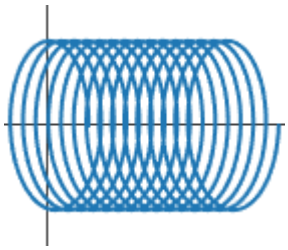


Figure 72: Ellipse wobble

USC-3 Ellipse: The ellipse wobble is available for the USC-3 card only.
Frequency, both amplitudes and Rotation can be changed. The amplitudes define the major and the minor axis of the ellipse. The shape can be rotated in respect to the default or marking direction (depending on the state of the flag "Along Marking Direction").

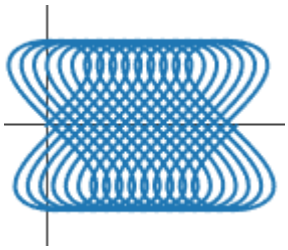


Figure 73: 8-Shape wobble

USC-3 8-Shape: The 8-shape wobble is available for the USC-3 card only.
Frequency, both amplitudes and Rotation can be changed. The transversal amplitude defines the height of the shape, the longitudinal amplitude the width. The shape can be rotated in respect to the default or marking direction (depending on the state of the flag "Along Marking Direction").

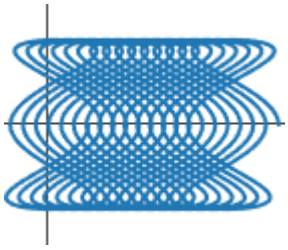


Figure 74: Double-8 wobble

USC-3 Double-8: The double-8 wobble is available for the USC-3 card only.
Frequency, both amplitudes and Rotation can be changed. The transversal amplitude defines the height of the shape, the longitudinal amplitude the width. The shape can be rotated in respect to the default or marking direction (depending on the state of the flag "Along Marking Direction").

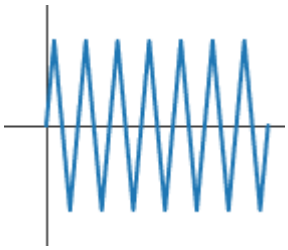


Figure 75: Zig-Zag wobble

USC-3 Zig-Zag: The zig-zag wobble is available for the USC-3 card only.
Frequency, Amplitude and Rotation can be changed. The Amplitude defines height of the shape. The shape can be rotated in respect to the default or marking direction (depending on the state of the flag "Along Marking Direction").

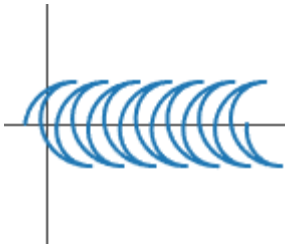


Figure 76: C-Shape wobble

USC-3 C-Shape: The C-shape wobble is available for the USC-3 card only.
Frequency, both amplitudes, Rotation and Curvature can be changed. The transversal amplitude defines the height of the shape, the longitudinal amplitude the width. The shape can be rotated in respect to the default or marking direction (depending on the state of the flag "Along Marking Direction").

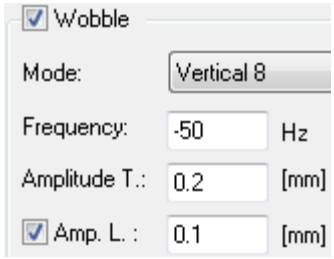


Figure 77: RTC5 wobble

RTC5 wobble modes: Ellipse, Horizontal 8 and Vertical 8.

With the Amp. L. checkbox (global, not pen specific) it is possible to set a different value for transversal and longitudinal amplitude.

- If the amplitudes are equal, the wobble shape is aligned in respect to the marking direction.
- If the amplitudes are unequal, the alignment of the wobble shape is along the X-axis.

Negative wobble frequencies are allowed and will result in a clockwise wobble motion. RTC5 wobble frequency range is [-6000 Hz, 6000 Hz].

6.1.3 Miscellaneous Settings for Pens

On this page skywriting, defocus and mark flags can be defined for a pen. The property page can be found under *Mark → Edit... → Misc*.

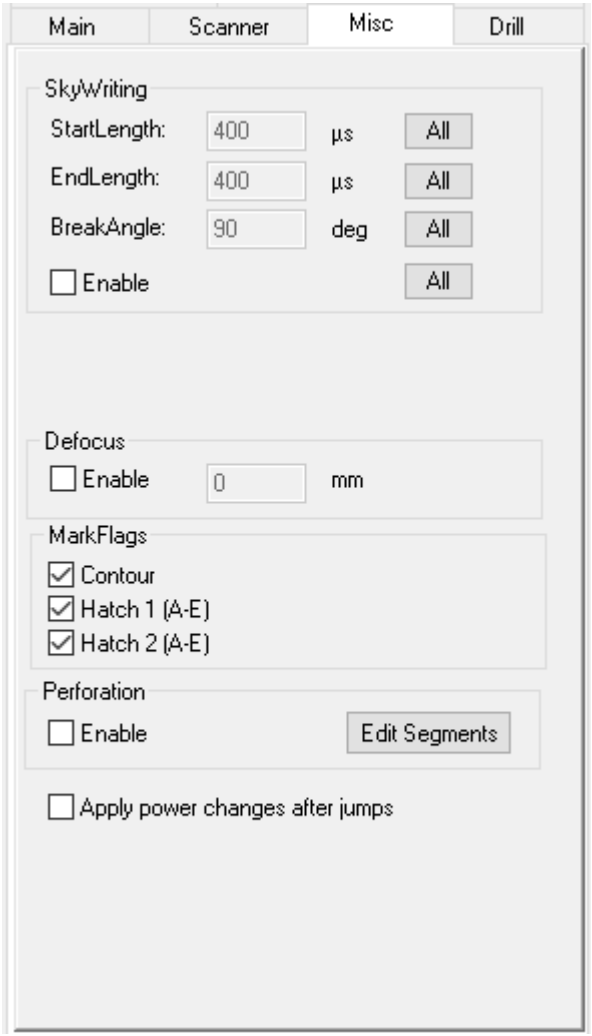


Figure 78: Misc Pen Settings

SkyWriting: SkyWriting is used to avoid marking while acceleration and deceleration of the mirrors to achieve very exact marked vertices. Therefore the feature adds a acceleration (StartLength) and a deceleration length (EndLength) to vectors. It is required to find new values for the laser and the scanner delays.

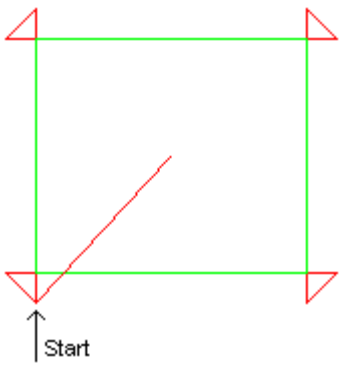


Figure 79: Example of skywriting

Parameters	Suggested values	Effect if value is too low	Effect if value is too high
StartLength	= ScannerLag	Laser shots are too narrow at the start of a vector.	No problem, just increasing mark time.
EndLength	= ScannerLag	Laser shots are too narrow at the last part of a vector	The last part of a vector is marked too long.
Jump Delay	normal	Oscillations could occur at the start of a vector.	No problem, just increasing mark time.
Mark Delay	= 0µs	No problem.	No problem, just increasing mark time.
Poly Delay	normal	The corners of the polyline are rounded off.	Burn-in effects at the corners.
LaserOn Delay	= StartLength + ScannerLag	The first part of a vector is marked too soon.	The first part of a vector is not marked.
LaserOff Delay	= 1µs	First delay rule for SkyWriting must be true.	The last part of a vector is marked too long.

Table 12: Suggested starting values and marking effects of SkyWriting parameters

ScannerLag is the time delay between XY2-100 signal and scan head. Depending on the scan head acceleration, a SkyWriting StartLength value higher than 'ScannerLag' could be required.

First delay rule for SkyWriting:

LaserOFF + StartLength + EndLength > LaserON

Second delay rule (unchanged):

Mark + LaserOn > LaserOff

StartLength: Scanner head starts before marking vector.

EndLength: Scanner moves after marking vector.

BreakAngle: Only relevant for PolyLines: Skywriting is enabled if angle 'a' between two successive mark vectors is greater than or equal to the BreakAngle value.

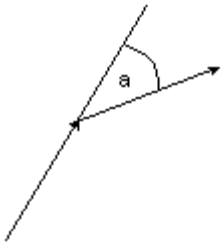


Figure 80: Break Angle

All: The All buttons applies the value to all 255 pens.

Enable: Enables skywriting for marking.

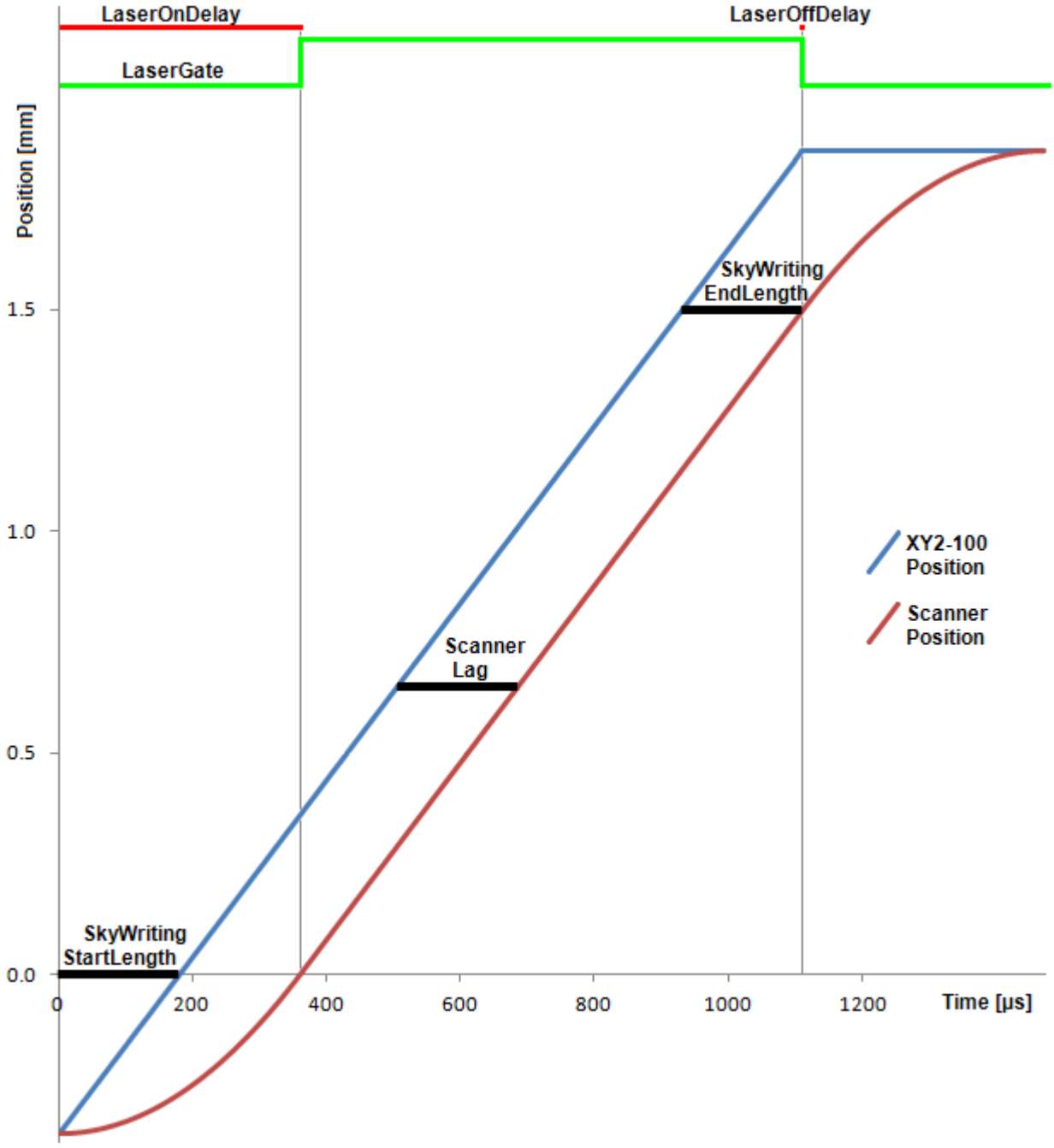


Figure 81: Skywriting diagram

Defocus: This activates a focus shift on the Z-Axis. The unit of pen defocus is:

- USC cards with FlatLense and Optic3D license: [mm]
- USC-1 cards with FlatLense but without Optic3D license: *Defocus not available*
- USC-2/3 cards with FlatLense but without Optic3D license: [2^{16} bit / FieldSize]
- RTC cards with SCANLAB 3D option: [2^{16} bit / FieldSize]
- RTC cards without SCANLAB 3D option: *Defocus not available*

MarkFlags: Contour or Hatch1/2 can be activated for every pen.

The SkyWriting parameters can also be used for black and white bitmaps.

Perforation: opens the perforation dialog.
Apply power changes after jumps: Delays power changes of the laser after the jump.

Scan ahead line params (RTC6): If a RTC6 card is set in SAMLight settings following additional group box appears and gets activated if "Use Excelliscan ScanAhead" checkbox is enabled at RTC6 Settings dialog.

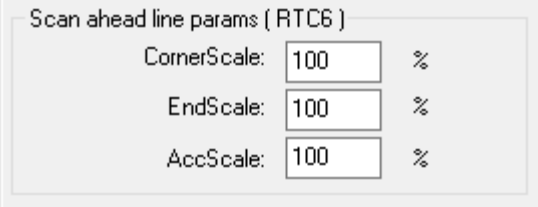


Figure 82: Scan ahead line params (RTC6)

CornerScale: Corner sharpness in percent. 100% = sharp corners.

EndScale: Marking accuracy at mark/jump and jump/mark transitions. 100% = straight line ends.

AccScale: Determines the portion of the acceleration time (not: distance traversed) in which the laser is active, in percent. 100% = entire acceleration time.

For more information about the 3 RTC6 SCANahead parameters above please consult chapter 3 and Ctrl Command set_scanahead_line_params of RTC6 +excelliSCAN manual.

6.1.3.1 Perforation Dialog

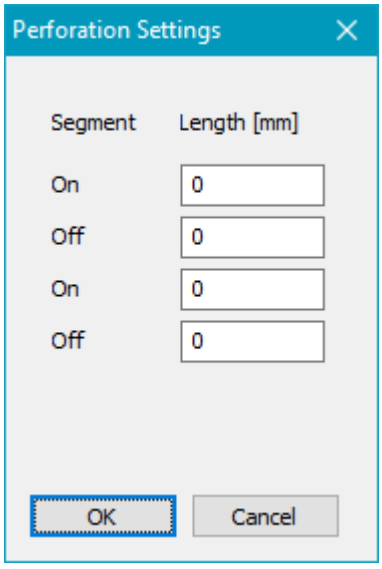


Figure 83: Perforation Dialog

Perforation: if enabled vectors can be marked as dashed lines. At “Edit Segments” dialog four different On/Off segment lengths can be set, which defines the dashed line. The marking result can be displayed in advance by Preview Window in combination with Mark Preview.

6.1.4 Drill Settings for Pens

The following page can be found at *Mark → Edit... → Drill*. If enabled, single points are scanned with drill mode.

YAG laser Pen → Drill settings:

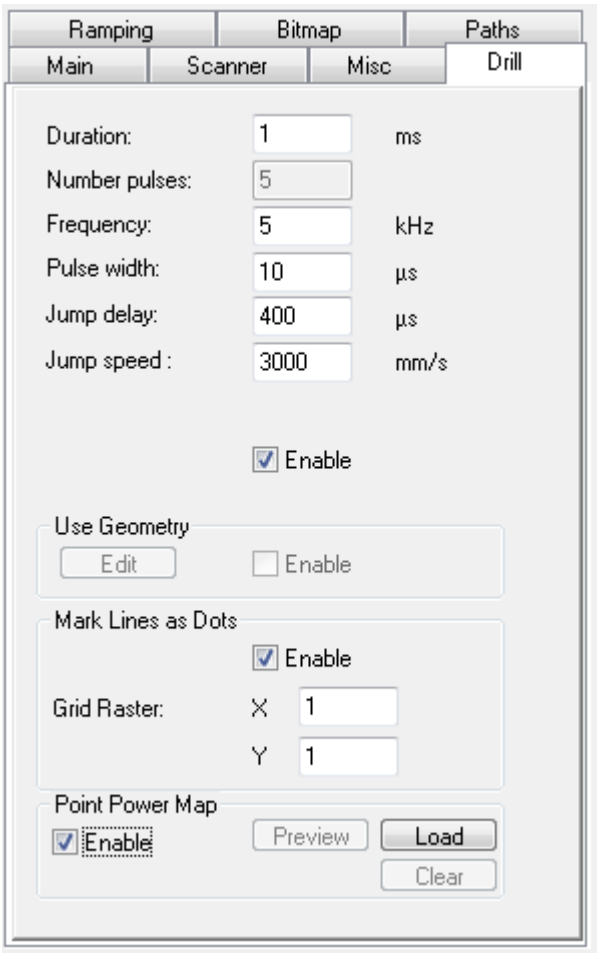


Figure 84: Drill Settings for YAG Laser

Duration: Time for scanning one point.

Number pulses: Displays the resulting number of pulses.

Frequency: Frequency of the laser pulses.

Pulse width: Q-Switch length in µs.

Jump delay: Delay between the jump to the point and start marking this point.

Jump speed: Speed to jump to a point.

Use Geometry: If enabled, clicking on "Edit" opens a window where a simple geometry can be created. This geometry will be marked at every point instead of just marking a single point.

Point Power Map: Load a bitmap in the background that defines the greyvalue for each pixel. Then if a dot is defined somewhere in the working area the power for the marking will be set by the Point Power bitmap. If *Preview* is greyed out, there is no bitmap assigned and the power will be taken from the pen power setting. If a bitmap was loaded with *Load* the bitmap is scaled to fit inside the fieldsize. The bitmap power values are affected by the Power Map also.

Mark Lines as Dots: If the enable checkbox is set, lines will be marked as dots that lie on the defined Grid Raster.

CO2 laser Pen → Drill settings:

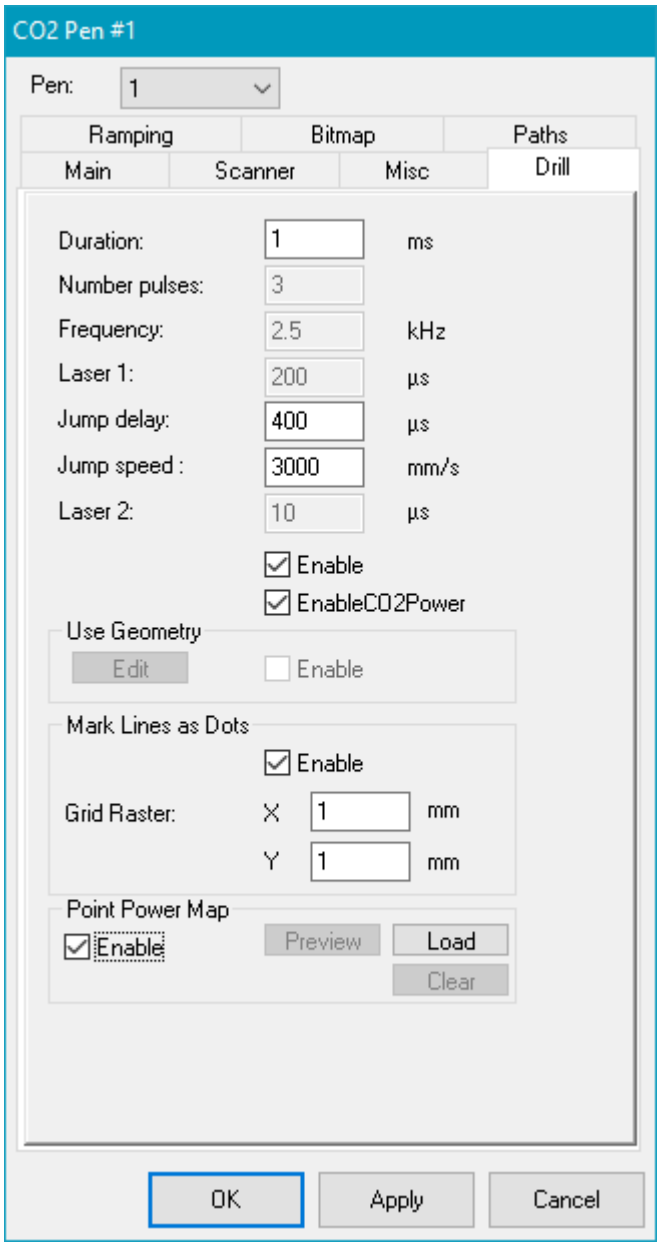


Figure 85: Drill Settings for CO2 Laser

Duration: Time for scanning one point.

Number pulses: Displays the resulting number of pulses.

Frequency: Frequency of the laser pulses.

Laser 1: Pulse length of laser signal1 in µs.

Jump delay: Delay between the jump to the point and start marking this point.

Jump speed: Speed to jump to a point.

Laser 2: Pulse length of laser signal2 in µs.

EnableCO2Power: If enabled Frequency, Laser1 and Laser2 are disabled, the values of these parameters are taken from the main page of pen settings instead.

Point Power Map: Load a bitmap in the background that defines the greyvalue for each pixel. Then if a dot is defined somewhere in the working area the power for the marking will be set by the Point Power bitmap. If *Preview* is greyed out, there is no bitmap assigned and the power will be taken from the pen power setting. If a bitmap was loaded with *Load* the bitmap is scaled to fit inside the fieldsize. The bitmap power values are affected by the Power Map also.

6.1.5 Ramping Settings for Pens

The following dialog can be reached at *Mark → Edit → Ramping*. Ramping allows to smoothly increase or decrease the speed or the power of the pen at the beginning or the end of the marking. It is also possible to add marking vectors at the beginning or the end of a marking in order to slowly increase the amount of energy delivered to the sample object.

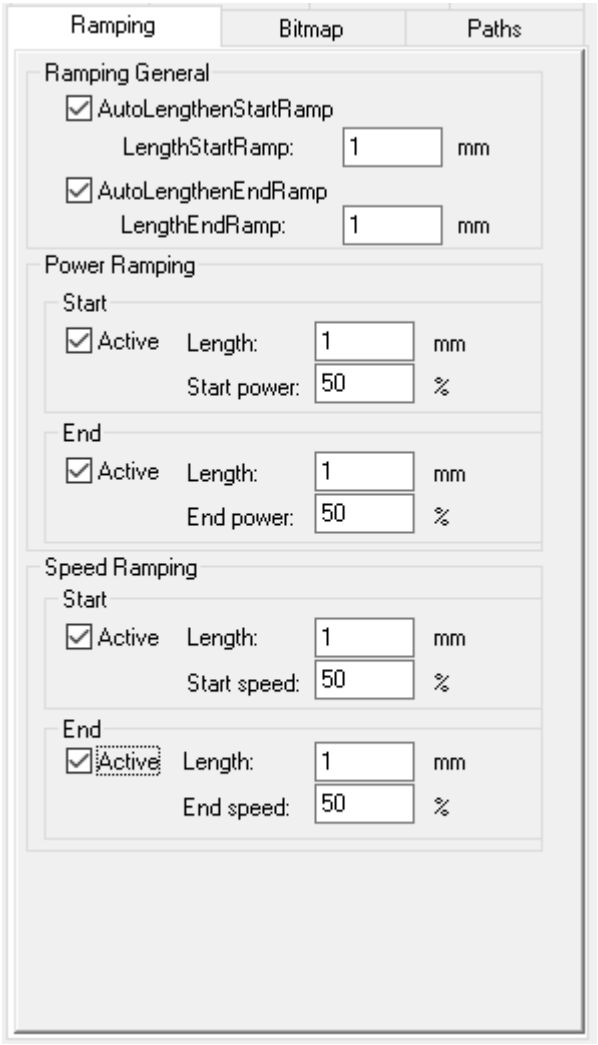


Figure 86: Ramping Settings Dialog

Ramping General:

AutoLengthenStartRamp: Add marking vectors before the beginning of the actual marking.

AutoLengthenEndRamp: Add marking vectors behind the end of the actual marking.

Power Ramping:

Start: Enable power ramping at the beginning of the marking. This smoothly increases the power of the laser from a given start value Start power to 100 % within the Length in mm. The checkbox Activate must be checked to enable this functionality.

End: Enable power ramping at the end of the marking. This smoothly decreases the power of the laser from 100 % to a given End power within the Length in mm. The checkbox Activate must be checked to enable this functionality.

Speed Ramping:

Start: Enable speed ramping at the beginning of the marking. This smoothly increases the speed of the scanner from a given start value Start speed to 100 % within the Length in mm. The checkbox Activate must be checked to enable this functionality.

End: Enable speed ramping at the end of the marking. This smoothly decreases the speed of the scanner from 100 % to a given End speed within the Length in mm. The checkbox Activate must be checked to enable this functionality.

6.1.6 Bitmap Settings for Pens

Grayscale:

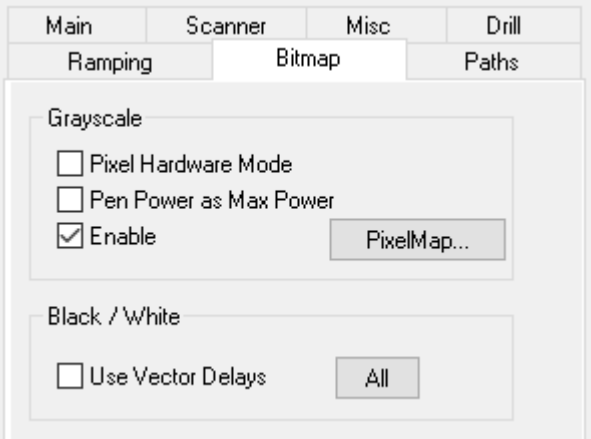


Figure 87: Bitmap Settings Dialog

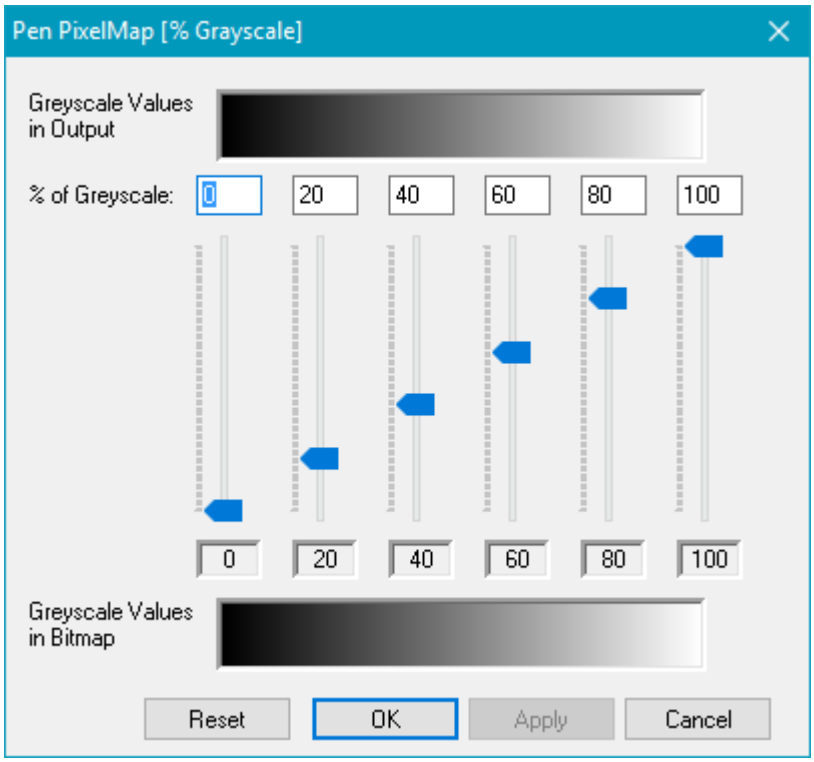


Figure 88: Pen PixelMap Dialog

Grayscale:

Pixel Hardware Mode: This checkbox should be enabled if using a gray scale bitmap.

Pen Power as Max Power: Uses the current pen power as maximum power for gray scale bitmap marking. As in all grayscale modes the power map is ignored. The factor is calculated by the current power value divided by the maximum power value.

PixelMap: This should be used to correct non-linearities of the material. For example for some materials there is a minimum laser power for which a marking result can be observed. For general laser dependent non-linearities you can also use the System PixelMap which will adjust the greyscale values after they have been mapped by the Pen PixelMap. For further information please refer to adjusting bitmap greyscale values.

Black / White: If the checkbox "Use Vector Delays" is activated, the vector delays set in Scanner will be applied.

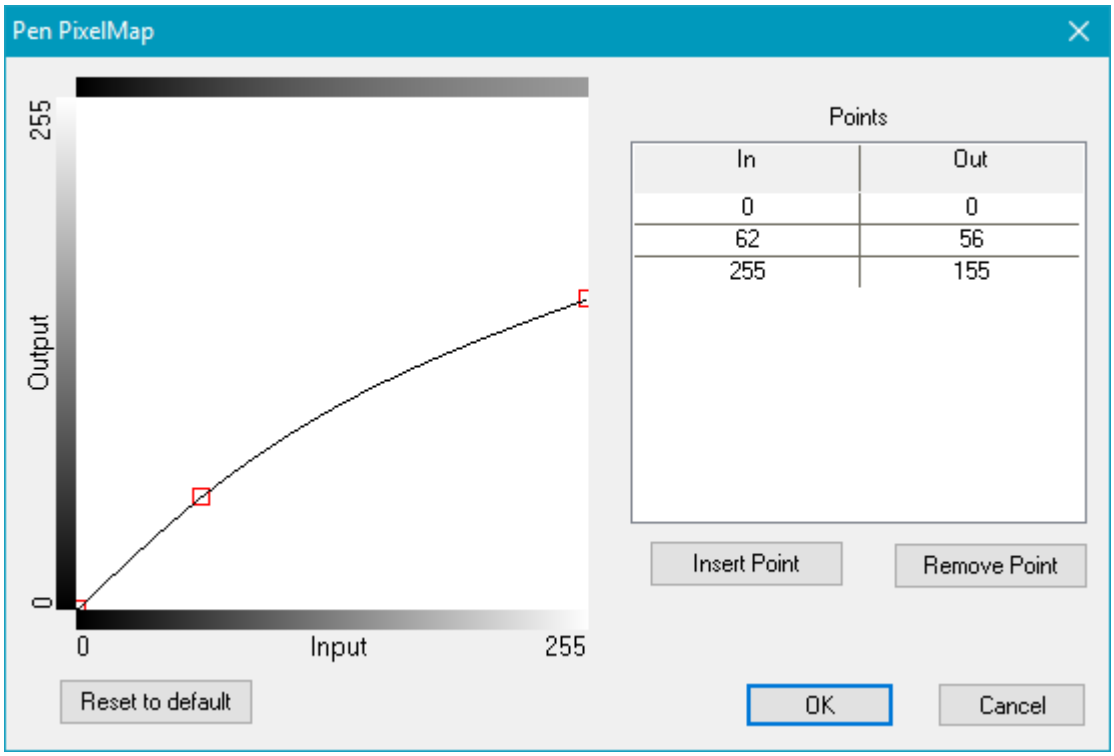


Figure 89: Dynamic Pen PixelMap Dialog

The dynamic Pen PixelMap and the mode of interpolation is activated via Settings → System → Extras → Enable Dynamic Grayscale Map. Up to 255 individual interpolation points can be set by clicking on the line in the input-output view or by 'Insert Point'. Points can be changed by editing the values in the table or by dragging the red boxes in the input-output view.

This Pen PixelMap applies a new grayscale value to grayscale bitmaps according to the linear, cubic or hermite interpolated input-output values in the table of Points.

6.1.7 Pen Paths

The Pen Path property sheet can be opened via the entity property sheet *Mark* by double-clicking on a pen or by clicking the *Edit* button. The appearance is as follows:

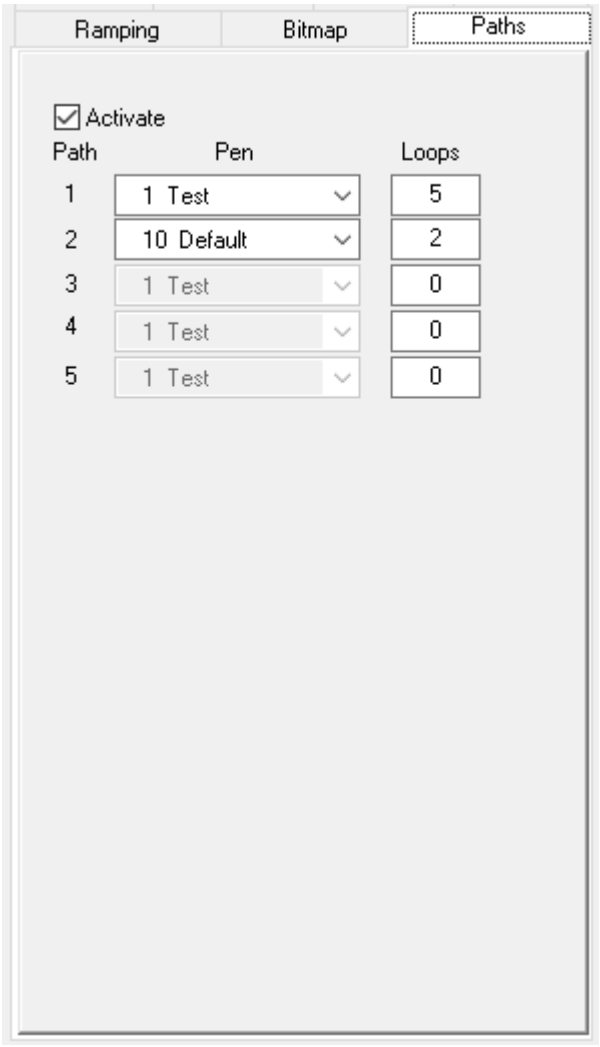


Figure 90: Pen Paths Dialog

Activate: If checked, the pen path function is active.

Path: A maximum of 5 different pens can be assigned to the path.

Pen: Define which of the 256 pens should be assigned to the position in the pen path.

Loops: Define how often a position in the pen path is repeated.

To activate a new pen path go to the edit field below *Loops* and enter a number greater than 0. Then click on *Apply*. Now the pen field of the defined path becomes active. Choose the desired pen by clicking on the drop down box in the pen field.

The Loops feature can be combined with the Mark Loop Count of the Entity info property sheet. By default the loops of the Pen path are executes first and then the entity is repeated with the Mark Loop Count. This behavior is different for splitting. In splitting (angular or 1D planar) the Mark Loop Count is executed first and then the Pen path Loops are executed.

If entities are grouped there is an additional checkbox present in the Entity Info property sheet in the field *Group*: The checkbox *PenPaths* can be activated. If so, the Pen Path loops are executed first and then the individual Mark Loop Counts of the entities are executed. If not checked the order is the other way around.

It is not possible to create Pen Path self-recursions. This means, if a pen is included as a pen path of another pen, then the pen paths of the included pen will not be executed. So the included pen will be treated as a normal pen.

6.2 Pen Advanced

The following dialog appears when pressing the *Advanced...* button on the M*ark* property page.

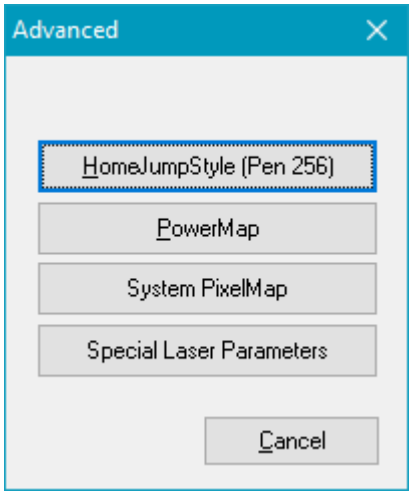


Figure 91: Pen Advanced Dialog

Head Selection: For Multiple Head applications HomeJumpStyle, PowerMap and System PixelMap can be set up head independent.

HomeJumpStyle (Pen 256): Opens the Edit dialog for Pen #256 which is used for the home jump.

PowerMap: The power map can be used to calibrate the laser in case the signals given to the laser and the resulting laser power do not behave linearly.

System PixelMap: The pixel map can be used to calibrate the laser in case the signals given to the laser and the resulting pixel occurrence do not behave linearly.

Special Laser Parameters: For some laser types (not visible if not available), pre-definition of values for special parameters like pulse length or APD index is possible.

6.2.1 Power Map

The PowerMap can be edited by clicking Advanced... on the mark property page and then clicking on Power Map. The power of the laser is controlled by 8 bit values coming out of the controller board. The behaviour is not always linear. The power map can be used to calibrate the laser, means to find out the 8 bit values for the exact power value. This map is helpful to transfer jobs between systems using laser sources with different power.

Dialog for YAG laser:

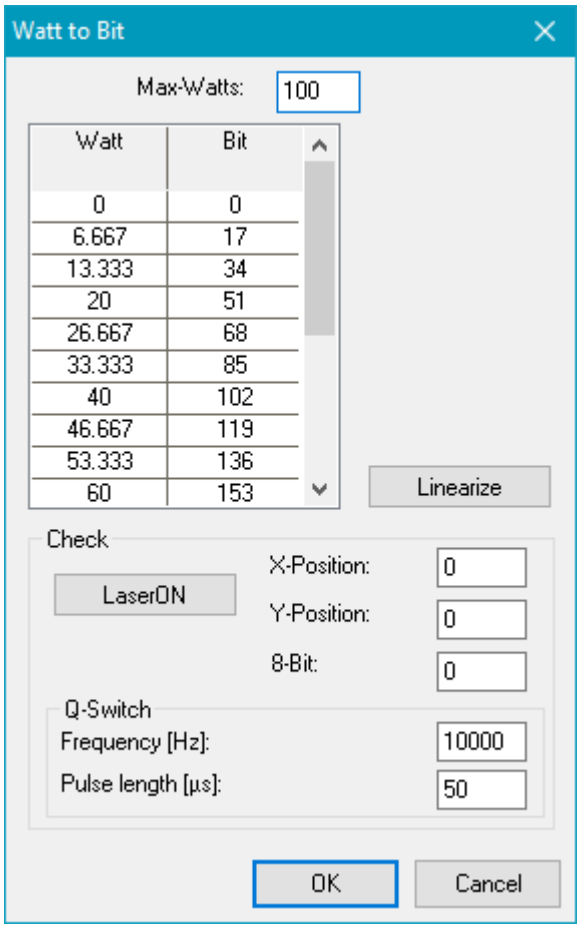


Figure 92: Power Map for YAG Laser

Max-Watts: Maximum power which can be emitted.

List: Edit this list to map a resulting power of laser pulse to a 8 bit value. The Watt column is divided into sixteen equidistant values from 0 to Max-Watts whereas the Bit column is editable.

Linearize: Calculating power values so that the Bits increase linearly from 0 Watt to Max Watts.

Check: The Check field is for measuring the laser power in watt on a specific X-/Y-Position working with the edited Q-switch settings under a 8-bit value signal.

LaserON: With this button the laser can be switched on and off again.

For calibration the following procedure is suggested:

1. Measure the maximum power by putting out a 8-bit value of 255.
2. Type in this max value in the Max-Watts field and press RETURN.
3. Now the Watt values in the list are updated.
4. Find out the corresponding 8-bit value for each of the given Watt values.

Dialog for CO2 laser:

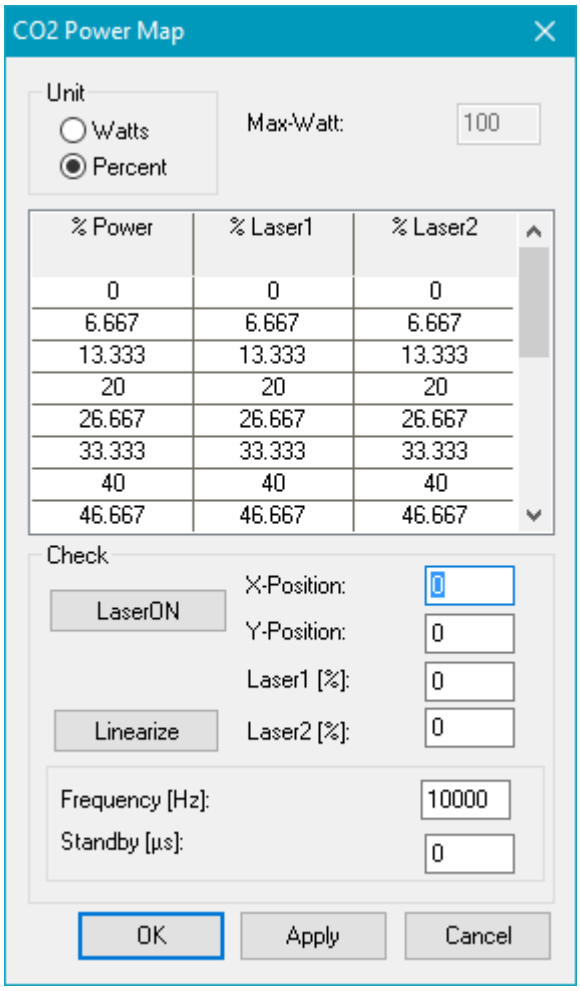


Figure 93: Power Map for CO2 Laser

Percent: The power is given in percentage.

List: Edit this list to map a resulting power of laser pulse to the laser signal 1 and 2. The % Power column is divided into sixteen equidistant values from 0% to 100% whereas the %Laser1 and %Laser2 columns are editable.

Check: The Check field is for measuring the laser power in % on a specific X-/Y-Position working with the edited Frequency under defined Laser1 and Laser2 signals.

LaserON: With this button the laser can be switched on and off again.

6.2.2 System PixelMap

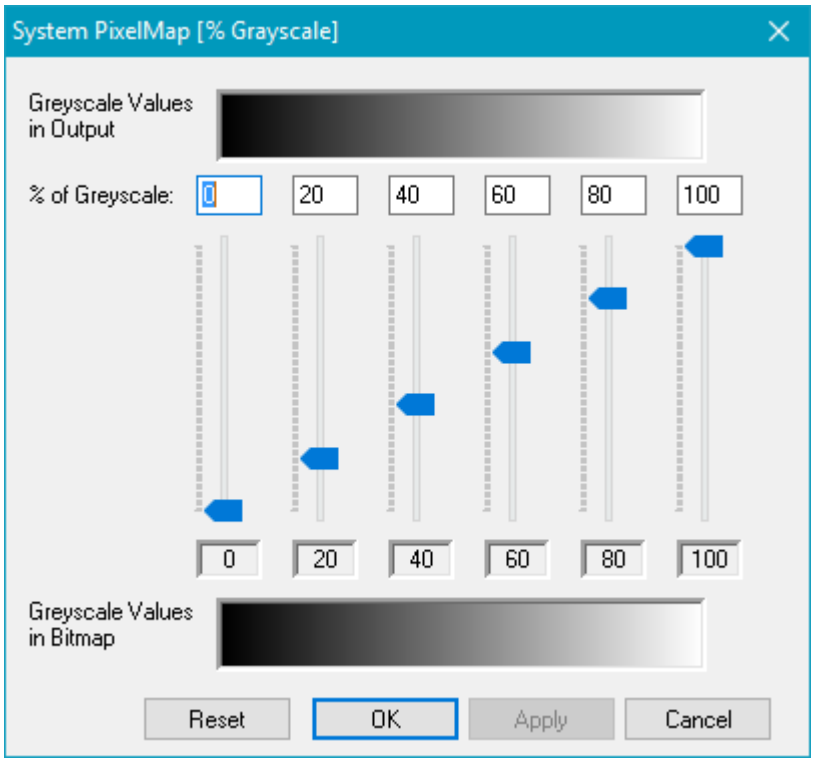


Figure 94: System PixelMap Dialog

The global System PixelMap applies a new greyscale value to every greyscale bitmap of the job. The grayscale values of bitmaps are already adjusted by the Pen PixelMap before they are mapped again by this System PixelMap. This pixel map can be used to consider general laser settings. For further information refer to adjusting bitmap greyscale values.

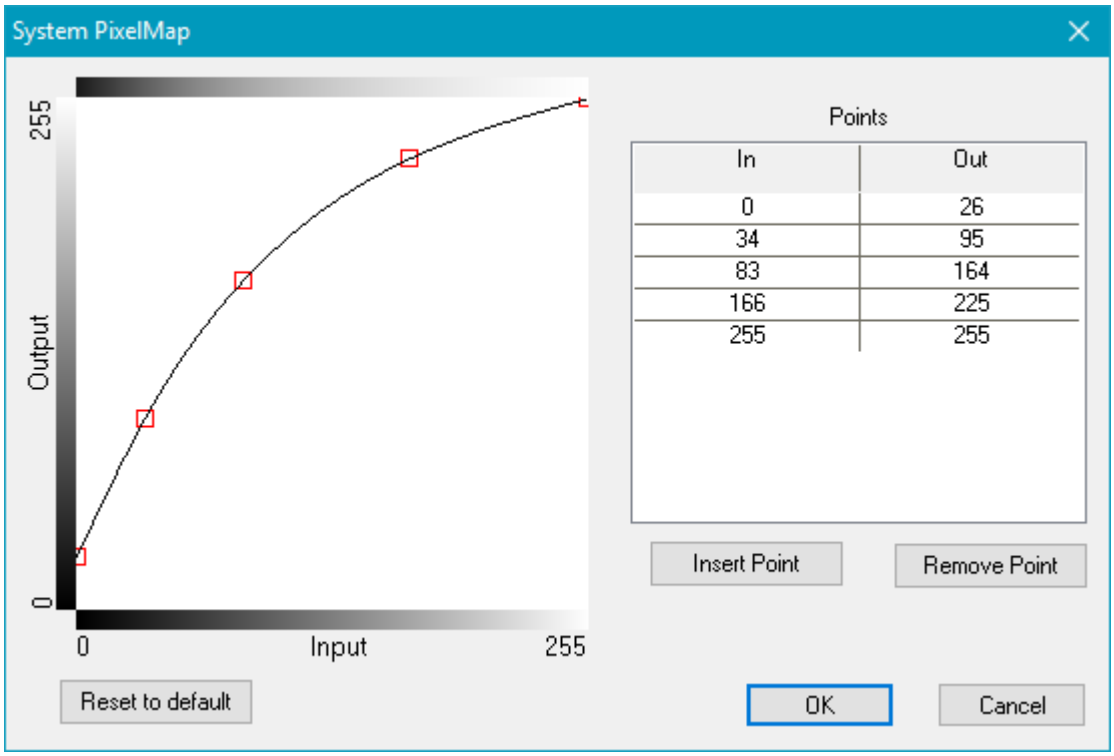


Figure 95: Dynamic System PixelMap Dialog

The new System Pixel Map and the mode of interpolation is activated via Settings → System → Extras → Enable Dynamic Grayscale Map. Up to 255 individual interpolation points can be set by clicking on the line in the input-output view or by 'Insert Point'. Points can be changed by editing the values in the table or by dragging the red boxes in the input-output view.

This system pixel map applies a new grayscale value to grayscale bitmaps according to the linear, cubic or hermite interpolated input-output values in the table of Points.

7 User Interface

At start up the application displays the user interface like it is shown below.

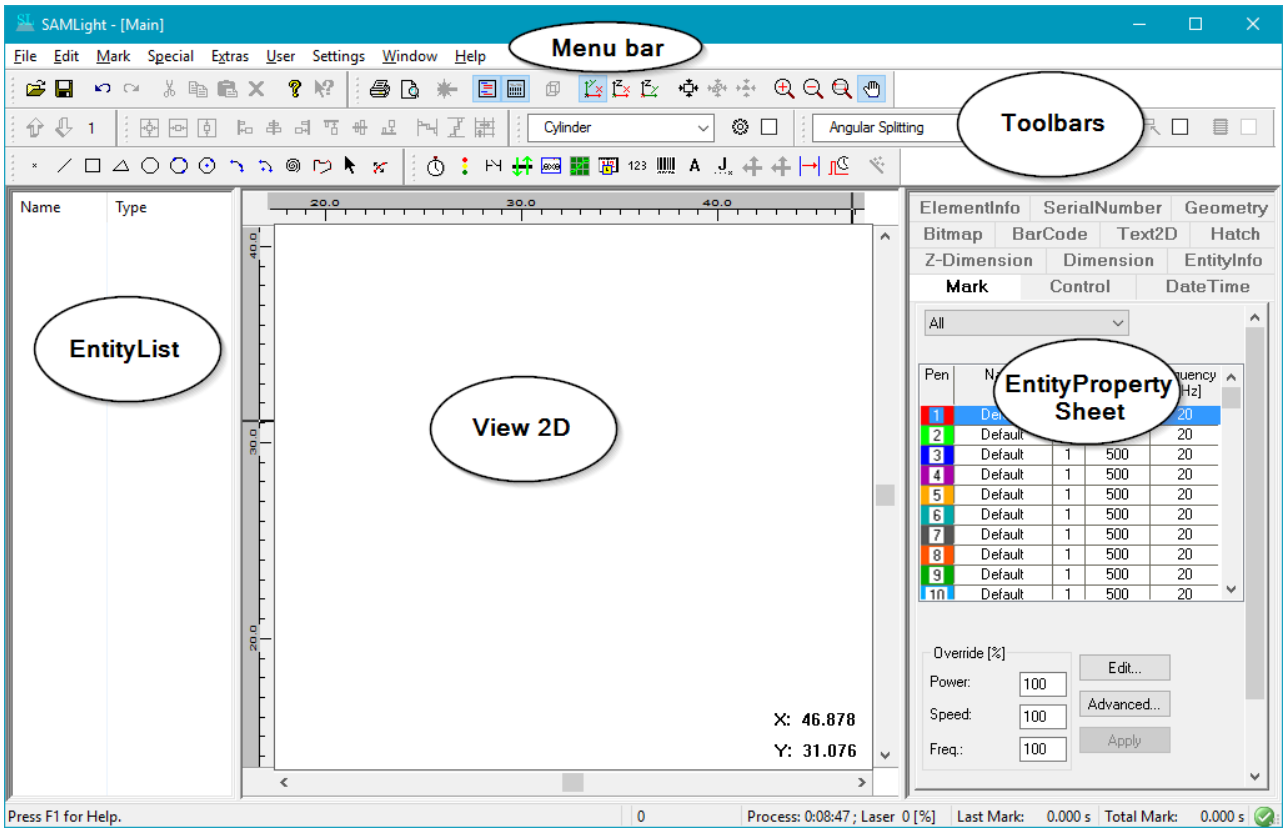


Figure 96: Main Window

Topics of User Interface:

- Menu bar
- Toolbars
- Main Window: Entity List / View 2D / Entity Property Sheet

7.1 Menu Bar

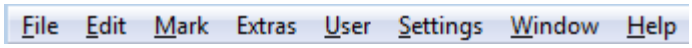


Figure 97: Menu bar

Topics of the Menu bar:

- File
- Edit
- Mark
- Extras
- User
- Settings
- Window
- Help
- Special: The menu item Special is shown, if there are user defined Script settings.

7.1.1 File

New: Prepares for a new job and it deletes all current entities.

Load...: Opens a dialog to read jobfiles in SCAPSJobFile-Format (*.sjf) format. See chapter Job Format.

Save...: Saves the current job in sjf format.

Save as...: Opens a dialog to save the current job under a new file name (*.sjf). See chapter Job Format.

Job Properties...: See chapter Job Properties.

Import...: Import of data in format, for more details see chapter Import.

Extension	Description
*.ai	Adobe Illustrator (AI) is a vector graphics file format.
*.gbr	Gerber Format (GBR) is a vector graphics file format for printed circuit boards.
*.gif	Graphics Interchange Format (GIF) is a raster graphics file format.
*.job	GSI PC-Mark job file format containing vector graphics.
*.tif	Tagged Image File (TIF) is a raster graphics file format.
*.txt	Point Cloud Data is a ASCII format containing 3D vertices.
*.274x	RS-274X is an extended Gerber Format, see *.gbr.
*.bmp	Bitmap (BMP) is a raster graphics file format.
*.cmx	Corel Metafile Exchange (CMX) is a vector graphics file format.
*.cnc	CNC G-Code is a language to control CNC (Computer Numerical Control) machines.
*.dst	Tajima DST is an embroidery vector graphics file format.
*.dwg	DraWinG (DWG) is a binary CAD file format.
*.dxf	Drawing Exchange Format (DXF) is a CAD file format.
*.emf	Enhanced Metafile (EMF) is a raster graphics file format.
*.jpg	Joint Photographic Experts Group (JPEG) is a compressed image format.
*.mcl	Marker Control Language (MCL) is a GSI PC-Mark vector graphics file format.
*.pcx	Personal Computer Exchange (PCX) is a raster graphics file format.
*.plt	Hewlett-Packard Graphics Language (HPGL) Plotter File (PLT) is a language format for printing line drawings, specifically designed for 2D plotters.
*.png	Portable Network Graphics (PNG) is a raster graphics file format.
*.saf	SAF is a SCAPS archive.
*.svg	Scalable Vector Graphics (SVG) is a 2D vector graphics file format.
*.tga	TGA or TARGA is a raster graphics file format.
*.twain	TWAIN is a software protocol and applications programming interface between software and scanner.

Table 13: Available import formats

Export...: Export of selected entities in HPGL (PLT) or SCAPS Archive (SAF) format. See chapter Export.

Print...: Prints the current View2D.This function works only if a printer is installed. See chapter View 2D.

PrintPreview: Shows a print preview of the current View2D. This function works only if a printer is installed. See chapter View 2D.

PrinterSettings: Shows the printer settings dialog.

Exit: Closes the SAMLight software.

7.1.1.1 Job Format

Menu bar → File → Load opens a dialog to load a new job from a SJF file (SCAPS Job Format). On the right hand side, there is a preview window. Directly below, there is a display box of all available entries inside the currently selected job file.

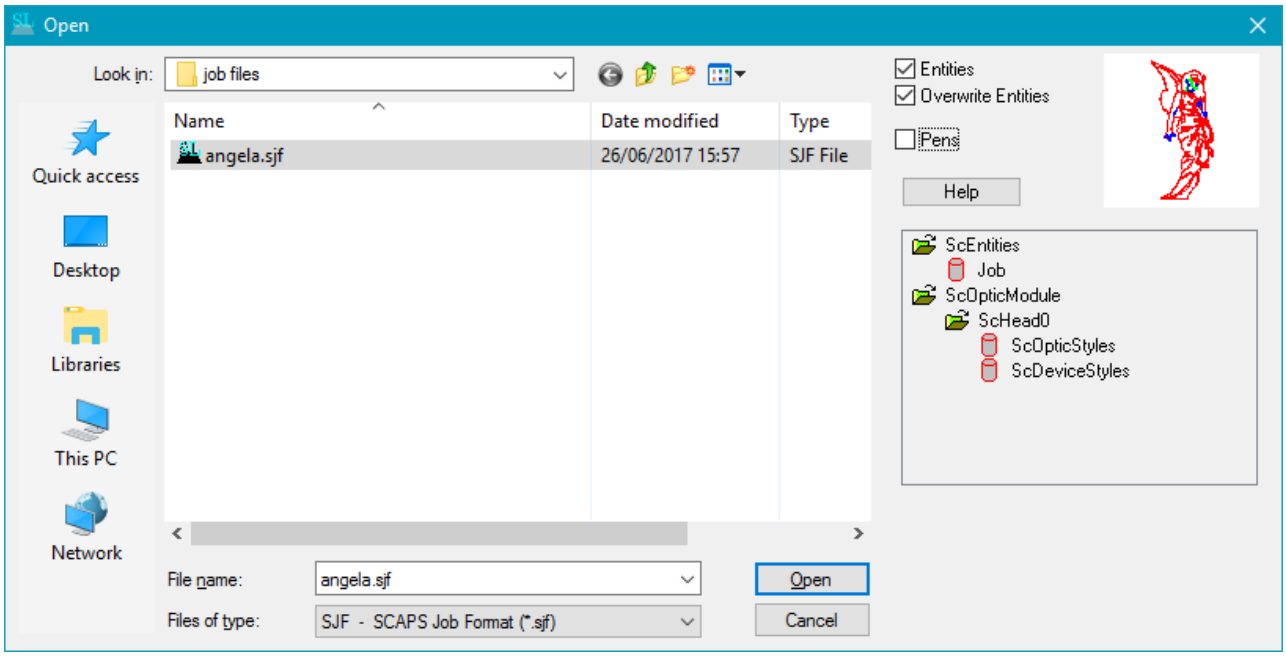


Figure 98: Open File Dialog

Menu bar → File → Save saves the current job. If there is no job name defined it is called *SaveAs*. *Menu bar → File → SaveAs* opens a dialog to save the current job under a new name.

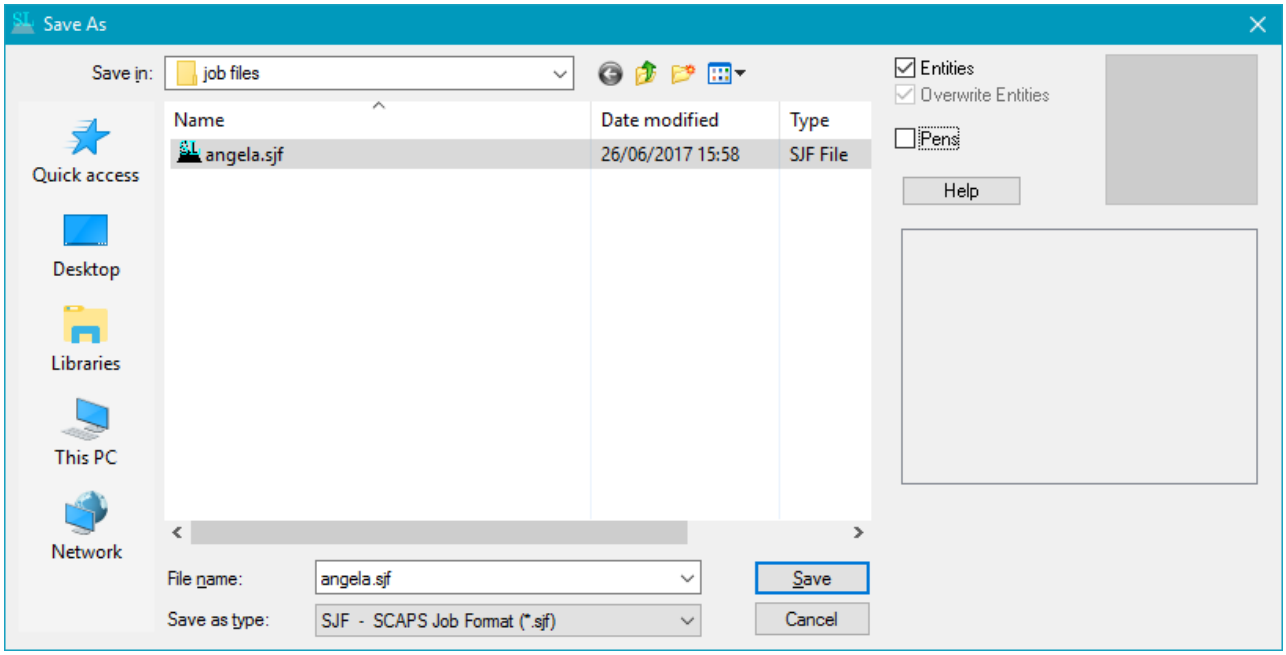


Figure 99: Save File As Dialog

Entities: If selected the entities of the selected file are loaded / saved.

Overwrite entities: This Check button is only active for dialog Load. If activated the entities of the current job are deleted when the job is loaded. If not the job entities are added to the current job.

Pens: If selected, the pens of the job are loaded / saved.

In general, the Pen Settings are saved in <SCAPS>\system\sc_light_settings.sam if "Save Pens On Exit" is enabled or "Save Pens Now" is used.
It is possible to save and load the Pen Settings within a job (.sjf). Be careful not to load any Pen Settings with a job and then accidentally overwrite other Pen Settings when leaving SAMLight while "Save Pens On Exit" is activated.*

It is highly recommended to backup sc_light_settings.sam every time after having done any important or time consuming configuration of the Pen Settings.

7.1.1.2 Job Properties

JobProperties...: This will open a dialog where the user can type in additional information about the job. This information will be saved within the jobfile:

Figure 100: Job Properties Dialog

NumObjectsPerPart: Defines the number of top level entities that are treated as one part. This affects the Part Counter Mode. See *Mark → Counter* and *Settings → General → Counter.*

7.1.2 Edit

Undo: Undo of the last operation. Not all operations support Undo. Undo is a command that erases the last change done to the current job reverting it back to the preceding state. The opposite of undo is redo, please see above for details.

Redo: Redo of last Undo. The operation that has been reverted by a Undo-operation is re-done.

Move: moves the position of the selected entity or entities in the entity list. The sequence in the entity list determines the sequence of marking.

Delete: Deletes the selected entities.

Duplicate: Copies the selected entities and brings them to view level 1 of the Entity List. See also chapter Entity List.

ArrayCopy: This creates an array of copies of the selected entities (objects). See chapter ArrayCopy.

ArrayPolarCopy: This creates the specified number of copies of the selected entities (objects) arranged in a circular way. See chapter ArrayPolarCopy.

Select: Provides functions for selecting the enteties one by one, etc. Works in the first view level. You can select all, first, previous, next and last entity of your entity list in the job.

Group: Groups the selected entities and puts them into an entities group. See also chapter Object Hierarchy.

UnGroup: Ungroups the selected entities group. The view level of all entities inside the group will be decreased by one. See also chapter Object Hierarchy.

Align...: If at least two objects are selected they can be aligned with the border or center of their outlines. See also chapter Align and Spacing Toolbar.

Spacing...: If at least three objects are selected they can get evenly distributed inside their common outline which is possible in horizontal or vertical direction. To do more specific spacing see dialog Spacing Advanced.

Nudge...: Translates a selected object by a small step. The nudge step is user defined in *Menu bar → Settings → System → View*. The Nudge 10 functions use 10 times of the Nudge step for the translation.

Center...: Translates a selected object so that the center of the working area becomes the center of the object. You can center along horizontal, vertical or both axes.

Rehatch All: Rehatches all entities of the job using the hatch values that are specified for them. When an entity is hatched and afterwards e.g. scaled the hatch lines are influenced by this scaling operation and are no longer conform to the hatch parameters that have been set before. By calling this operation such modifications are removed and the original hatch values are restored.

Set Pen Number: Assigns a pen (1..10) to the current job.

7.1.2.1 Spacing Advanced

The following dialog can be found under *Edit → Spacing → Advanced...*

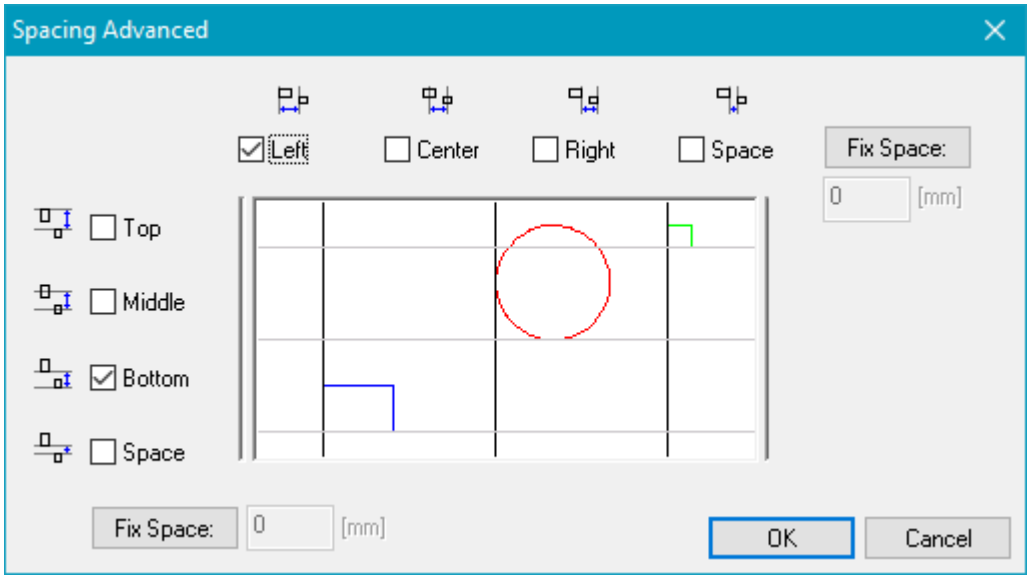


Figure 101: Spacing Advanced Dialog

The view in the middle of the dialog gives an example of the selected spacing.

Left: The spaces between the left outlines of successive elements are set equidistant inside the common outline.

Center: The spaces between the center of successive elements are set equidistant inside the common outline.

Right: The spaces between the right outline borders of each element are set equidistant inside the common outline.

Space: The spaces between the right outline border and the left outline border of the following object are set equidistant inside the common outline.

Fix Space: If one of the described attributes is defined an according fix space can be defined as well.

7.1.2.2 ArrayCopy

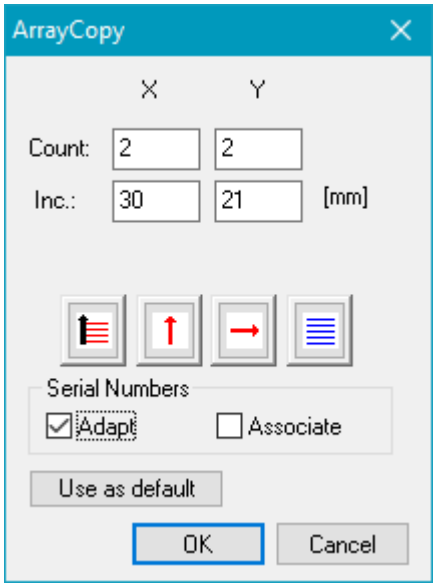


Figure 102: ArrayCopy Dialog

Count: defines how many copies to make for x and y direction.

Inc.: defines the distance between the copies in x and y direction.

The buttons in the middle define where and how to put the copies (this will be interesting for adapting serial numbers). In the case shown in the dialog screenshot above, the copies will be placed (and enumerated in the case of serial numbers).

1. columns first (rows first is possible by clicking on the first button)
2. above (below is possible by clicking on the second button) and right (left is possible by clicking on the third button) of the original one
3. all columns / rows due to 1. are ordered the same way first to last (alternating first to last / last to first is possible by clicking on the fourth button)

Adapt: If checked the ArrayCopy of a serial number would result in an array of serial numbers where the copies are enumerated from the number of the original one up to n. If not checked all copies will get the same number as the original one.

Associate: If checked the ArrayCopy of serial numbers results in one serial so that no serial number gets repeated. This check button is only enabled if Adapt is selected and association is possible. Creating one serial after an array copy is only possible if the actual values of the current serial differ in a constant step.

7.1.2.3 ArrayPolarCopy

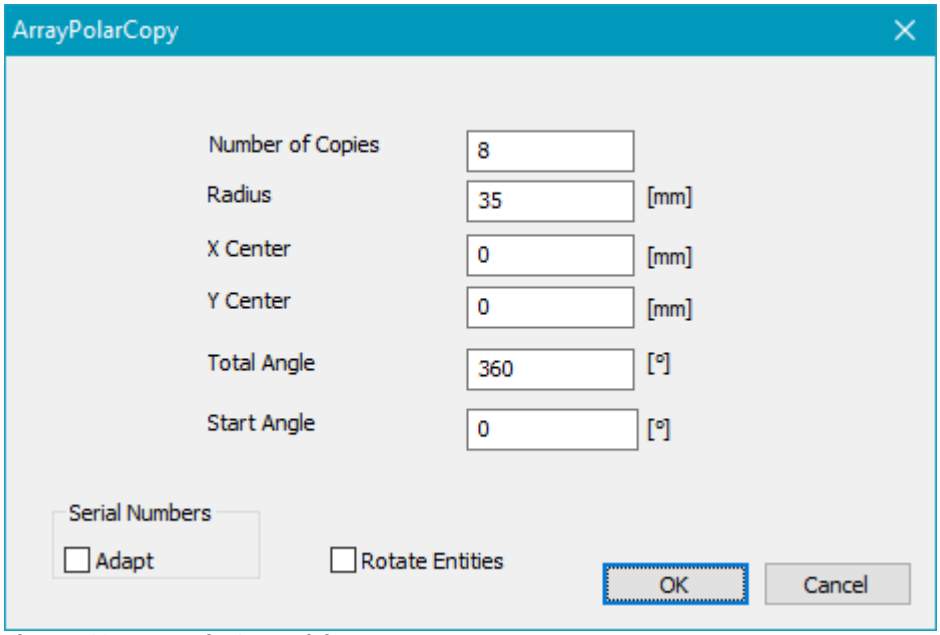


Figure 103: ArrayPolarCopy Dialog

Number of Copies: defines how many copies to make (inclusive original).

Radius: defines the radius of the circle on which the entities will be placed. The absolute position of the circle is specified by **X Center** and **Y Center**.

Total Angle: defines the angle over which the entities will be spread evenly.

Start Angle: defines the angle for the first entity to be placed.

Adapt: If checked, the ArrayPolarCopy copies of a serial number will be enumerated from the number of the original one up to n. If not checked, all copies will get the same number as the original one.

Rotate Entites: If checked, the copies will be aligned radially. If not checked, all copies will be aligned identically to the original entity.

7.1.3 Extras

Topics of Extras:

- Teach Reference - Teach new reference position(s)
- Relocate Reference - Recall and use reference position(s)
- Flash - opens the Flash dialog for standalone operation
- Splitting - split a job into pieces to mark the parts separately on a ring or with a working area that is smaller than the job itself
- Step / Repeat - repeat marking of the same job for a defined time including a movement to modify the position where that job is marked
- Bitmap Marking - adapt and mark a bitmap line by line on a ring

7.1.3.1 Teach / Relocate Reference

This feature allows you to define a reference position and relocate entities to this position. The reference position can be chosen with the 'Teach Reference' dialog. The translation can be applied to the job with the 'Relocate Reference' dialog. The red pointer (output bit 3) is active while these dialogs are open to show the position on the working field. This feature can be useful when it is difficult to move the workpiece.

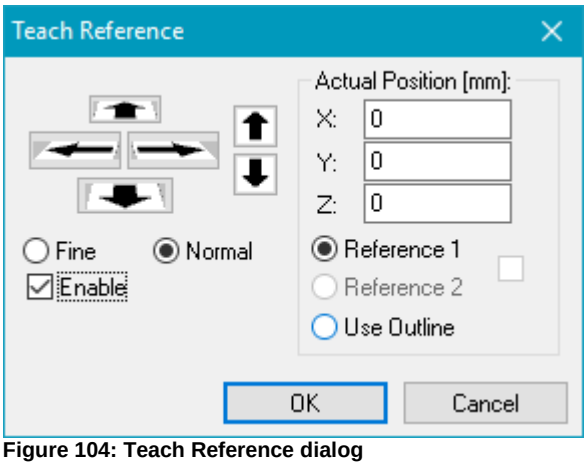


Figure 104: Teach Reference dialog

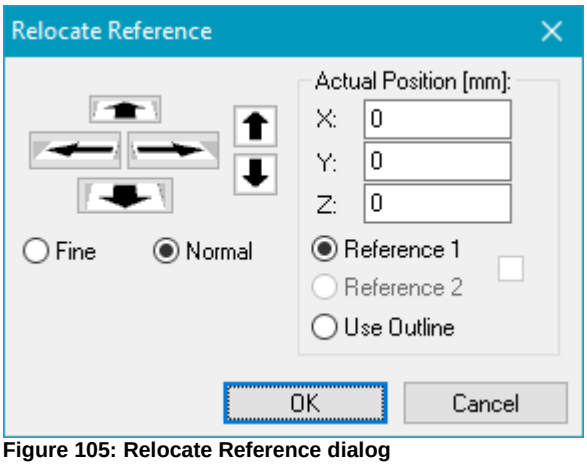


Figure 105: Relocate Reference dialog

A typical process with one reference position contains following steps:

1. Open 'Extras → Teach Reference' and choose a reference position.
2. Leave this dialog with 'OK' to store the reference position into the job (it is recommended to save that job afterwards).
3. Place a workpiece below the scan head.
4. Open 'Extras → Relocate Reference' and move the red pointer to the desired place on the target to get the marking result on the right place of the workpiece.
5. Leave this dialog with 'OK' to shift the entities in the View2D.
6. Mark the workpiece.

The teaching will be done only once, steps 3) - 6) need to be repeated for every workpiece.

Relocate uses the speed defined in Settings → System → Optic → Min/Max → max Jump Speed

For teaching and relocating reference points the related dialogs offer the following functions. These functions can be controlled by the keys described in brackets:

ArrowKeys (cursor up, down, left, right): Move the currently selected reference point in X- and Y-axis. The width of such a movement depends on the stepwidth.

DepthKeys (page up, down): Move the actual reference point in Z-axis (depth coordinate). The width of such a movement depends on the stepwidth.

Fine/Normal (toggle with Shift): Switch between the normal and the fine stepwidth, these values can be configured.

Enable: This checkbox is only available in the teach dialog. It can be used to enable or disable teaching and relocating for a job.

Actual Position: X, Y and Z are absolute coordinates of the currently selected reference points

Reference 1 / 2 (toggle with CTRL): Check *'Settings → System → Extras → Teach Mode, Use two reference points'* to enable a second reference position. With two reference positions it is possible to shift and **rotate** the entities in the View2D.
The one that is currently selected is changed using the cursor keys. If the checkbox between these both radio buttons is selected, the behavior is slightly different. In this case the reference point two is moved relatively to reference point one. This can be used for a raw location of the working piece in a first step to avoid moving wide distances for both reference points separately. If this box is unchecked, both reference points are changed completely independent from each other.

Use Outline: Instead of a single reference point the outline of a job can also be used to select a position. If this option is checked it behaves same as the reference position 1 and can be used to teach or relocate

position changes. Because an outline is always a rectangle with its sides parallel to X and Y axis this option can be used only in positioning mode with one reference point.

7.1.3.2 Step / Repeat

The step and repeat marking offers the possibility to repeat the same job for a defined number of times with some specific movements between every marking step. That is useful e.g. in cases when the job has to be marked at different positions on a working piece or to different working pieces that are located on the same working area. The movement can be:

- a rotation of the working piece performed by an external drive
- a planar movement of the working piece performed by external drives
- a planar movement performed by the translation of the current job while the working piece stays on its position

To use this feature following steps are necessary:

- configuration and enabling of the external motion control to perform the automatic movement of the object that has to be marked in case an external drive has to be used
- definition of the total number of repeats that have to be done for the same job during one marking cycle
- definition of the speed the motion controller has to drive the external hardware with between two marking steps in case an external drive is used
- selection of a Step/Repeat mode (Angular or Planar mode) and configuration of its specific settings

Configuring the *Motion Control* has to be done at *Menu bar → Settings → System → Extras*. Here in the field *Motion Control* the checkbox *Active* has to be selected. Additionally the motion control configuration file for the used controller has to be configured. Depending on the *Step/Repeat* mode that has to be used, the motion controller needs to be set up for angular or planar movements. The configuration dialog will only offer these modes and axes for configuration that are enabled using the correct *Motion Control* mode. If the motion controller was configured in a way that no axes are available for rotating or that the wrong axes (not X and Y) are configured for moving, there will be no configuration possibilities available in the *Step/Repeat* settings dialog beside the planar mode that translates the whole job to simulate some kind of movement. The following dialog can be opened via *Menu bar → Extras → Step / Repeat → Settings.*

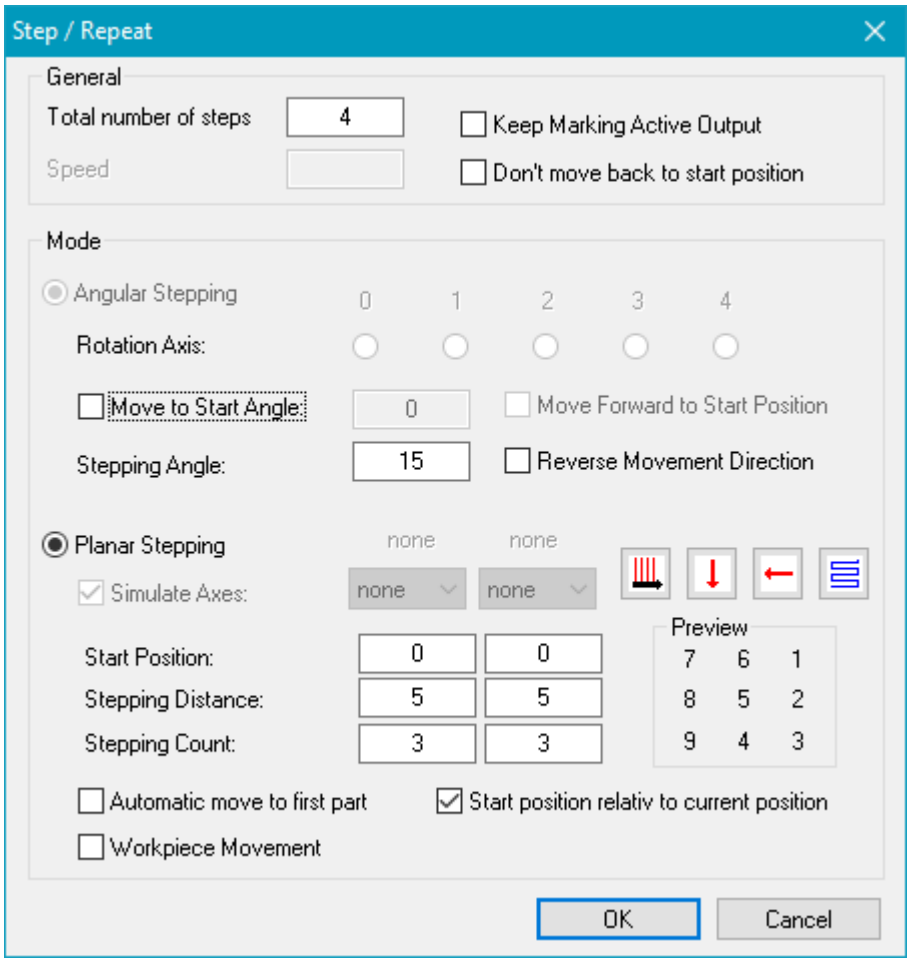


Figure 106: Step / Repeat Dialog

General: At the top of the dialog window the common settings that are valid for all Step/Repeat modes can be made. The *total number of steps* and the motion *speed* can be defined here.

Keep Marking Active Output: In normal operation mode the *digital_output_0* is set to 1 every time a split is marked and it is set back to 0 during the movement operation. This behavior can be changed by enabling the checkbox *Keep Marking Active Output*. If it is active then the marking signal via *digital_output_0* would stay at 1 as long as the complete job including all splits is marked.

Workpiece Movement: If the scan head is not moving but the workpiece is moving instead, then all relative movements have to be inverted.

Mode:

Angular Stepping: The angular mode has to be used for environments where the working pieces can be moved by performing a rotation. Here the following has to be defined:

- which axis has to be used for rotating (X, Y or Z)
- the starting angle (*Move to Start Angle*) and where it has to be moved back after performing a full marking Cycle
- the angle the working piece has to be moved during a step (Stepping Angle)

Move to Start Angle: This checkbox allows you to enable or disable the movement to the starting angle.

Forward to Start Position: Here you can force a movement that completes a full rotation instead of going back to the origin.

Planar Stepping: The lower part of the configuration window has to be used for setting up the planar

"step and repeat" operation. Here movements within one level and in two directions X and Y can be defined. Accordingly the motion controller needs to be configured in a way that the two axes X and Y are available for planar movements. Else this option can work in movement simulation mode only when the complete job is translated instead. This option can be enabled by setting the checkbox Simulate Axes. In this case no start position can be used. Here it is defined by the current position of the jobs entities within the working area.

For both axes the following values can be set:

Start Position: The starting position to which the motion controller moves to at the beginning of a full Step/Repeat marking cycle

Stepping Distance: The distance the axes have to be moved in a defined direction for every repeated marking

Stepping Count: The stepping count defines how often the job is marked.

Additionally there are four buttons that define the order and direction of the movements that are performed after every mark. Here the following things can be defined:

- the main stepping direction (mark rows or columns first)
- the direction for the Y axis (move in positive or negative direction using the defined distance)
- the direction for the X axis (move in positive or negative direction using the defined distance)
- the movement type (unidirectional or bidirectional)

According to the direction and movement types that are selected here the preview below of these buttons shows the order and positions of the marks like they will be executed.

Other like it is known from the rotary mode here no option exists that allows it to force a movement back to the starting position. Instead of this it is possible to use the option "Move Forward and Back" that performs a special optimized movement. Here the first one goes forward and works exactly like defined by the movement direction buttons but after finishing it, it stays at the reached position. The next movement is reverse to the one defined by the buttons so that it goes back to the starting position.

Automatic move to first part: If the center of the field is at the same position as the entity that has to be marked, this option enables the software to find the starting point automatically.

Start position relative to current position: For moving workpieces on a conveyor or by a motor this option sets the starting position of the following split relative to the actual position.

Don't move back to start position: If this option is unchecked, the scan head moves back to the defined starting angle or position, after the last marking cycle has been done.

Enable Step/Repeat: When all settings are made conform to the used working environment the Step/Repeat marking mode has to be enabled by using M*enu bar → Extras → Step/Repeat → Enable Mode*. Now starting a marking operation no longer results in one single marking cycle. Instead the marking is repeated until the configured number of steps is reached.

7.1.3.3 Bitmap Marking

Bitmap Marking (or Bitmap Rotary) allows you to mark one single bitmap on rings continuously without the need to split the bitmap in parts manually. Comparing to the standard splitting functionality this one

- requires a connected motor, which is controlled via step and direction signals. SAMLight motion type must be "**Type 8 - Generic stepper controller**" or "**Type 14 - USC-2 stepper controller**".
- requires a bitmap, which is not rotated in SAMLight
- modifies the bitmap automatically to fit to the resolution of the drive and the desired bitmap marking parameters; here the dither step value is an important parameter

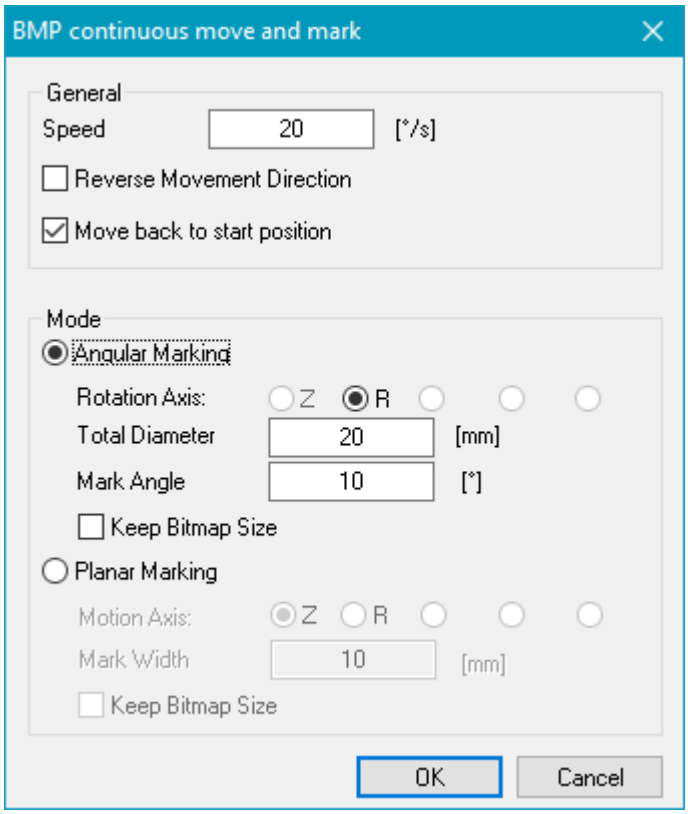


Figure 107: Bitmap Marking Dialog

The following steps are necessary to mark a bitmap along one of its axes:

1. The motion driver has to be configured for rotary or planar marking mode
2. A bitmap has to be imported. Its scanner bitmap has to be created and configured according to the desired results. If the option "Scan XDir" is set for that bitmap the bitmap splitting functionality will recognize this automatically and will perform all checks and calculations using the correct bitmap direction.
3. The Continuous Bitmap Marking setup dialog that can be found in submenu "Settings" has to be called to set up the rotation or motion axis, the diameter of the ring, the marking angle for angular marking as well as the rotation direction and the mark width or planar marking.
4. The Continuous Bitmap Marking feature has to be enabled for the next marking process by *Menu bar → Extras → Bitmap Marking → Enable Mode*.

General:

Speed: The speed the connected drive has to be moved in between two lines of the bitmap

Reverse Movement Direction: Reverses the direction the drive moves e.g. to mark the inner part of a ring

Mode:

Angular Marking:

Rotation Axis: Selects the axis of the drive that has to be used for the movement. This option is important when there are more than one angular axes configured for a motion drive, but it does not influence the splitting direction of the bitmap. If there is no angular axis configured the complete angular marking mode is not available.

Total Diameter: The diameter of the ring that has to be marked. This value is used to check if the laser is able to create a result without gaps. Is the dither step of the bitmap much smaller than the drives resolution the operation would fail.

Mark Angle: Specifies which part of the ring has to be marked with the bitmap. Depending on this

value the size of the bitmap is modified to fit. In case the option *Keep Bitmap Size* is not chosen.

Keep Bitmap Size: If that option is set the bitmap is not scaled in order to get a size that results in the specified marking angle, here the source bitmap is left untouched.

Planar Marking:

Motion Axis: Selects the axis of the drive that has to be used for the movement. If there is no planar axis configured the complete planar marking mode is not available.

Mark Width: Indicates the width of the marking.

Keep Bitmap Size: If that option is set the bitmap is not scaled in order to get the size that is specified by the mark width, here the source bitmap is left untouched.

7.1.4 User

You can login as a defined user at *Menu bar → User → Login*. It is enabled with user password mode, see section User Level.

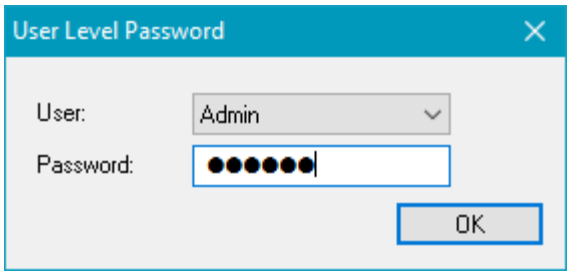


Figure 108: User Level Password Dialog

7.1.5 Window

Maximize: Maximizes the actually displayed window (Main Window or Preview Window).

Tile: Shows the Main Window and the Preview Window next to each other.

Preview: Shows the Preview Window.

Main: Shows the Main Window.

7.1.6 Help

Contents: Opens the help window.

About...: Shows an information window with the version number.

7.2 Toolbars

The following toolbars are available:

- File Toolbar
- Camera Toolbar
- View Level Toolbar
- Geometry Object Toolbar
- Functionality Object Toolbar
- Align and Spacing Toolbar

- Extras Toolbar
- Stepper Position Toolbar
- 3D Surfaces Toolbar
- Special Sequences Toolbar

The toolbars can be activated/deactivated in *Menu bar* → *System* → *Settings* → *View:*

Clicking on the button *Toolbars* opens the following dialog:

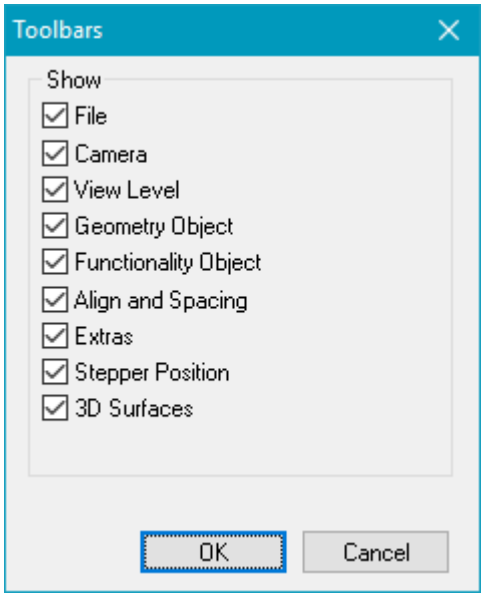


Figure 109: Select Toolbars Dialog

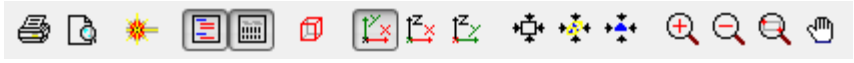
Checking / Unchecking an item will activate / deactivate the corresponding toolbar.

7.2.1 File Toolbar

File Toolbar: This toolbar provides the following shortcuts (from left to right):

- *Load* an existing jobfile (*.sjf). See chapter Job Format.
- *Save* the actual job. See chapter Job Format.
- *Undo* the last operation.
- *Redo* the previously undo operation.
- *Cut* the selected entity.
- *Copy* the selected entity.
- *Paste* the copied entity.
- *Delete* an item from the Entity List.
- *Help* - If this button is activated you will get context related help for several dialogs and controls by clicking on it.

7.2.2 Camera Toolbar



Print: Prints the current View2D.This function works only if a printer is installed. See chapter View 2D.

 **Print preview:** Shows a print preview of the current View2D. This function works only if a printer is installed. See chapter View 2D.

 **Mark:** Opens the Mark Dialog.

 **ShowEntity List:** Shows and hides the Entity List on the left side of the View2D. See chapter Entity List.

 **ShowPropSheet:** Shows and hides the Entity Property Sheet on the right side of the View 2D. See chapter Entity Property Sheet.

 **3D View:** Select at least one entity and click this button to open the 3D view. This feature is available only with Optic 3D.

 **Plan View xy:** Clicking on this button changes the perspective of the View2D to plain view. This feature is available only with Optic 3D.

 **Side View xz:** Clicking on this button changes the perspective of the View2D to side view. This feature is available only with Optic 3D.

 **Side View yz:** Clicking on this button changes the perspective of the View2D to side view. This feature is available only with Optic 3D.

 **Fit All:** Clicking on this button fits the view to the scanner field.

 **Fit All Entities:** Clicking on this button fits the view to all entities in the view.

 **Fit Selected:** Clicking on this button fits the view to the current selected entities.

 **Zoom Plus:** Clicking on this button zooms in by factor 2.

 **Zoom Minus:** Clicking on this button zooms out by factor 2.

 **Zoom Window:** Clicking on this button allows the user to do a user defined zoom window. To define a zoom window follow these steps:

- After clicking this button move the mouse to the first corner of the window.
- Click the left mouse button and keep it pressed.
- Drag the mouse to the second window corner and release the left mouse button.

 **Hand Tool:** Clicking on this button activates the following functions:

- Drag: With the mouse the working space of view 2D can be dragged.
- Free zoom: With the mouse wheel you can zoom in or out relative to the mouse pointer.
- Centered zoom: By holding CTRL and drag with the left mouse button the user can zoom in or out.

7.2.3 View Level Toolbar

 **View Level Toolbar:** This toolbar provides two arrow buttons for increasing or decreasing the View Level of the Entity List. The third button of the View Level Toolbar displays the level number of the actual View Level. For more detailed information see the chapter Entity List.

7.2.4 Geometry Object Toolbar

Point: Creates a point by clicking the left mouse button at the desired position in the View 2D.

Line: Creates a straight line defined by a start and end point: Click the left mouse button at the desired start position in the View 2D. Move the mouse to the desired end position and click the left mouse button again.

Rectangle: Creates a rectangle in two steps: Click the left mouse button at the desired position of the left upper corner in the View 2D. Move the mouse while keeping the left mouse button pressed to the desired position of the right lower corner and release the left mouse button. To change the geometry of the rectangle after its creation see the chapter Rectangle. To change the position and/or the orientation of the rectangle see the chapter View 2D → Manipulation of objects.

Triangle: Creates a triangle that is defined by its three corners: Click the left mouse button at the desired position of first corner in the View 2D. Click the left mouse button at the desired position of second corner. Click the left mouse button at the desired position of third corner. To change the position and/or the orientation of the triangle see the chapter View 2D → Manipulation of objects.

Ellipse: Creates an ellipse: Click the left mouse button at the desired position in the View 2D. Move the mouse while keeping the left mouse button pressed until the ellipse has the requested size. To change the geometry of the ellipse's after its creation see the chapter Ellipse. To change the position and/or the orientation of the ellipse see the chapter View 2D → Manipulation of objects.

Circle - 3 Points: Creates a circle out of 3 Points. Click the left mouse button at three desired positions one after another in the View 2D. Change the circle by moving those three contact points. Change the segment count within the Geometry Property Page.

Circle - Center Radius: Creates a circle out of two Points: the center of the circle and one point which is an element of circle. Click left mouse button at the two desired positions one after another in the View 2D. Change the circle by moving those two contact points. Change the segment count within the Geometry Property Page.

Arc - 3 Points: Creates an arc out of 3 Points which are elements of the arc. Click the left mouse button at the three desired positions one after another in the View 2D. Change the arc by moving those three contact points. Change the segment count within Geometry Property Page.

Arc - Center Angle: Creates an arc out of 3 Points: the center of the arc and two points which are element of the arc and define the beginning and the end of the arc. Click the left mouse button at the three desired positions one after another in the View 2D. Change the arc by moving those three contact points. Change the segment count within Geometry Property Page.

Spiral: Creates a Spiral by clicking the left mouse button at the desired position in the View 2D.

Polyline: Creates a PolyLine defined by a sequence of points: Click successively the left mouse button at the desired positions in the View 2D to generate a sequence of points. To finish the sequence click the right mouse button and choose one operation of the provided two operations "Finish" or "Close&Finish". To change the position and/or the orientation of the Polyline see the chapter View 2D → Manipulation of objects.

Select: Switch to entity select mode. See chapter View 2D → Selection.

Point Editing: Switch to point editing mode. See chapter Entity List → Point Editor.

7.2.5 Functionality Object Toolbar

Timer: Creates a default timer entity. Please refer to section IO Control Objects.

Wait For Input: Creates a default WaitForInput entity. Please refer to section IO Control Objects.

Set Output: Creates a default SetOutput entity. Please refer to section IO Control Objects.

Set Override: Creates a ScOverride entity. By clicking on it a ScOverride Icon is shown in the entity list. For more information see chapter SetOverride Control Objects.

Set Executable: Creates a ScExecutable entity. By clicking on it a ScExecutable Icon is shown in the entity list. For more information see chapter Executable Control Object.

Set Analog Output: Creates a default SetAnalogOutput entity. Sets a value in percent of the output ports A or B.

Motion Control: Creates a default MotionControl entity. This button is only available if Motion Control is activated in the menu *Settings* → *System* → *Extras*. Please refer to section Motion Control.

Motion Control Go: Creates a special MotionControl entity. SAMLight will continue with the next entities without waiting for the motion to be ended. Please remind that the motion command will continue even if the marking in progress signal is off at the end of the entity list.

Date Time: Creates a default DateTime entity. Please refer to section Date Time.

Serial Number: Creates a default Serial Number object. Please refer to section Serial number.

Barcode: Creates a default Barcode entity. Please refer to section Barcode.

Text2D: Creates a default Text2D entity. Please refer to section Text2D.

Jump: Creates a default jump entity. Please refer to section ScJump Control Object.

Motf Offset: Defines an offset for a marking on-the-fly application. When this offset has elapsed a trigger event will be released. For more details on how this feature can used to set up advanced MOTF-jobs, please refer to the section Trigger Control Objects.

Wait For Trigger: The execution of a job is stopped until a trigger event is detected. This can be an external hardware trigger or a trigger signal released by a preceding Motf Offset object. For more details on how this feature can used to set up advanced MOTF-jobs, please refer to section Trigger Control Objects.

Data Wizard: Allows to do different data manipulations on the selected objects. Please refer to section Data Wizard.

 **Parameter Finder:** Allows to optimize pen parameters. Please refer to section Parameter Finder.

7.2.5.1 Data Wizard

The following dialog appears after pressing the magic wand in the object toolbar. All operations are done on the selected polylines.

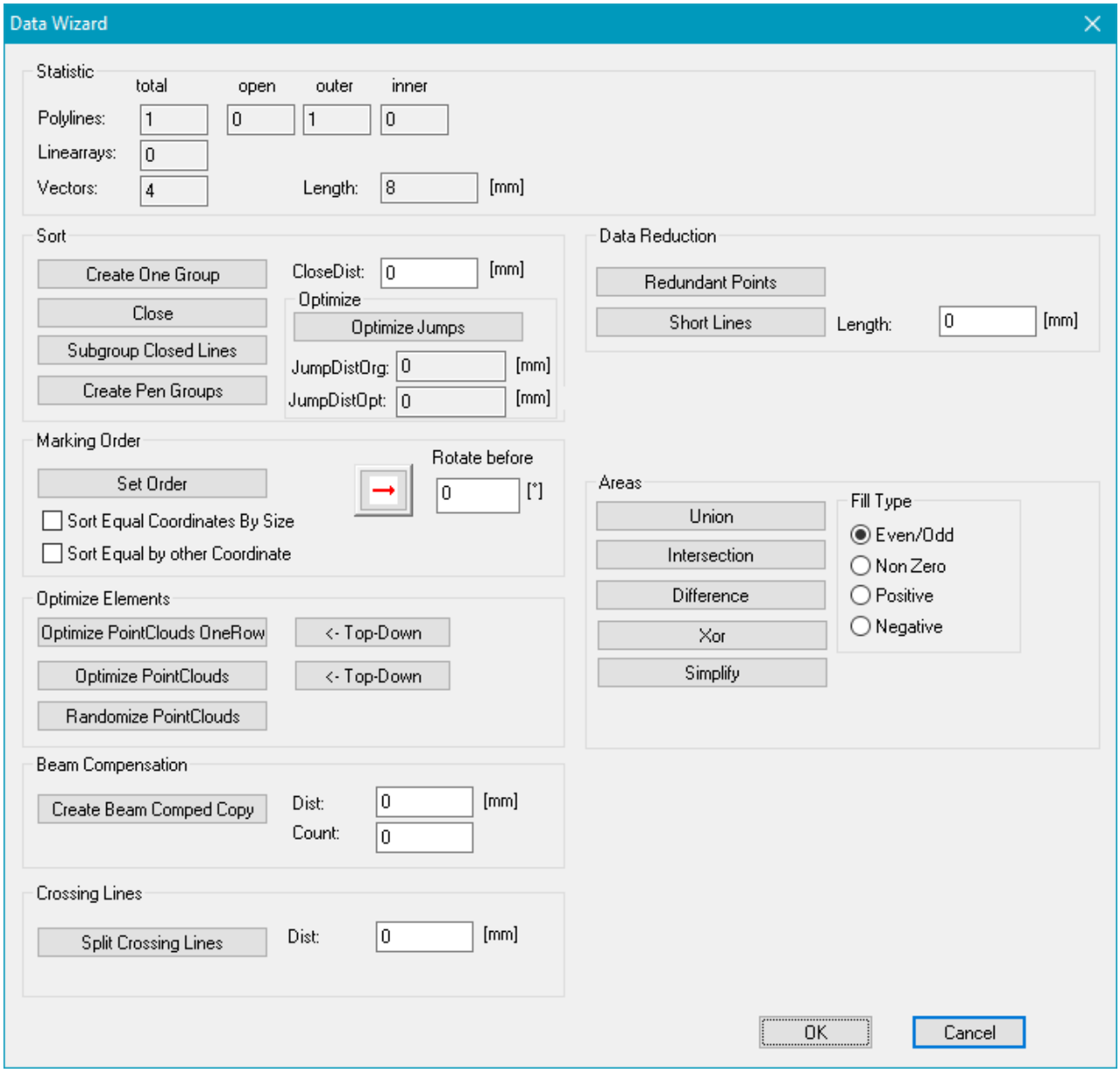


Figure 110: Data Wizard Dialog

Statistic:

total: Number of all objects

outer: Objects which are orientated counterclockwise.

inner: Objects which are orientated clockwise.

Length: Total Length in mm

Sort:

Create One Group: Can be used to optimize the order of the vectors to minimize the marking-time. All selected ScPolyLine2D objects will move into a new ScPolyLines2D folder. Then two polylines will be closed to a new polyline if the distance between them is smaller than CloseDist (in mm). If CloseDist is set to '0' no polylines will be closed. Afterwards all polylines will be sorted in order to minimize jumps. If necessary and possible polylines will be flipped to optimize the marking order of the polylines and thus the marking-time as well.

Close: Sorts and closes open polylines if the distance of open points is smaller than given CloseDist.

Subgroup Closed Lines: Groups selected Subgroups in one Layer.

Create Pen Groups: Puts and sorts all selected objects in one new main group consisting of different subgroups - one for each pen. Depending on the laser type, this can reduce marking time when more than one pen is used for marking because the number of switches between the different pens is reduced.

CloseDist: Distance below open polylines are closed.

Optimize Jumps: the total jump distance between poly lines would be optimized. This feature is only designed for vector graphics. Before clicking the button, a group of entities must be chosen first. After clicking OK, a new group of entities with optimized distance would be created on the position of the original entities. At the end, the original entities are to be deleted. This step is suggested to be made at last, because the optimized group could not be edited as the original entities. The home jump distance is not calculated in the algorithm. If the home jump is not enabled, the default start position is (-10,000, -10,000, -10,000). That means, enabling home position or not may differ the optimized result.

JumpDistOrg: shows the total jump distance of the original group entities.

JumpDistOpt: shows the total jump distance of the optimized group entities.

Marking Order:

The field Marking Order provides the functionality for automatically arranging the marking order of a chosen group of entities.

- Click the button containing a red arrow to change the primary sorting rule.

 sorts entities in ascending order according to the x-coordinate of their most left points.

 sorts entities in descending order according to the x-coordinate of their most right points.

 sorts entities in ascending order according to the y-coordinate of their lowest points.

 sorts entities in descending order according to the y-coordinate of their highest points.

- activate the check box to enable the secondary sorting rule.

Sort Equal Coordinates By Size: the entities with the same coordinate would be sorted by their size from small to large.

Sort Equal Coordinates By Other Coordinate: the entities with the same coordinate in the primary sorting rule would be sorted by the other coordinate from most left to most right or from lowest to highest.

- **Rotate before:** rotates the chosen entities counterclockwise around the source point by the number defined in the text box in degree before sorting.

Performs the sorting accordingly to the settings above.

Optimize Elements: The following options work for ScPointCloud2D entities.

Optimize PointClouds OneRow: Sort point clouds in stripes with x direction. The width of a stripe is the side length of a square with the area of the point cloud entity divided by the number of points. This kind of sort should improve the marking speed of point clouds.

Optimize PointClouds: Sort point clouds in stripes with x direction. The width of a stripe is 3 times the side length of a square with the area of the point cloud entity divided by the number of points. This kind of sort should improve the marking speed of point clouds.

Randomize PointClouds: Resort point clouds randomly distributed.

Beam Compensation:

Create Beam Comped Copy: This option creates scaled down inner copies or rather scaled up outer copies of each selected object depending on the orientation of the object.

Dist: Distance between the created copies.

Count: Number of copies.

Crossing Lines:

Split Crossing Lines: This feature is designed for polyline, number and text to avoid double marking on the point of two crossing lines. The split will be made on a copy in the same position of the original entities.

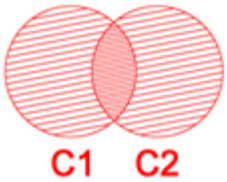
Dist: Distance of the split into two parts at the breaking point.

Data Reduction:

Redundant Points: Removes points of a straight line which do not define the straight line.

Short Lines: Removes points which are located on a polyline and which have a small distance to the neighboring points of the polyline.

Length: describes the distance of the points.



Areas: There are four boolean operations to combine two areas, for example C1 and C2:

Union (OR): New area where either C1 or C2 or both are filled

Intersection (AND): New area where only the intersection of C1 and C2 is filled

Difference (NOT): New area where where everything that is not C2 is filled

Exclusive Or (XOR): New area where only the non-intersecting areas are filled

Simplify: Self-intersections from the supplied polygon will be removed by performing a boolean union operation.

Fill Type: The filling rules define the handling of areas and holes. There are four filling rules for the four boolean operations:

Even/Odd: Only odd numbered sub areas are filled.

Non Zero: All non zero sub areas are filled.

Positive: All sub areas with winding counts bigger than 0 are filled.

Negative: All sub areas with winding counts smaller than 0 are filled.

Manual Split:

Splits the selected Polyline with respect to the X,Y or Z axis at the desired co-ordinate.

Round:

Each X or Y coordinate of the selected entity will be rounded to the next full value given by the input field

in [mm].

Some of the functions change the organization of the selected object. For example: If using "Create One Group" on a serial number this will change it into a plain text entity.

7.2.5.2 ParameterFinder

The Parameter Finder is a tool that helps to optimize pen parameters. It is structured as a wizard which guides the user through the different steps of:

1. Define the pen which should be optimized, the grid and the entity to be used for testing
2. Choose the "X Parameter" to be varied in horizontal direction
3. Choose the "Y Parameter" to be varied in vertical direction
4. Mark, selection of the best result, fine tuning.

When clicking on the icon of the Parameter Finder, the following dialog appears:

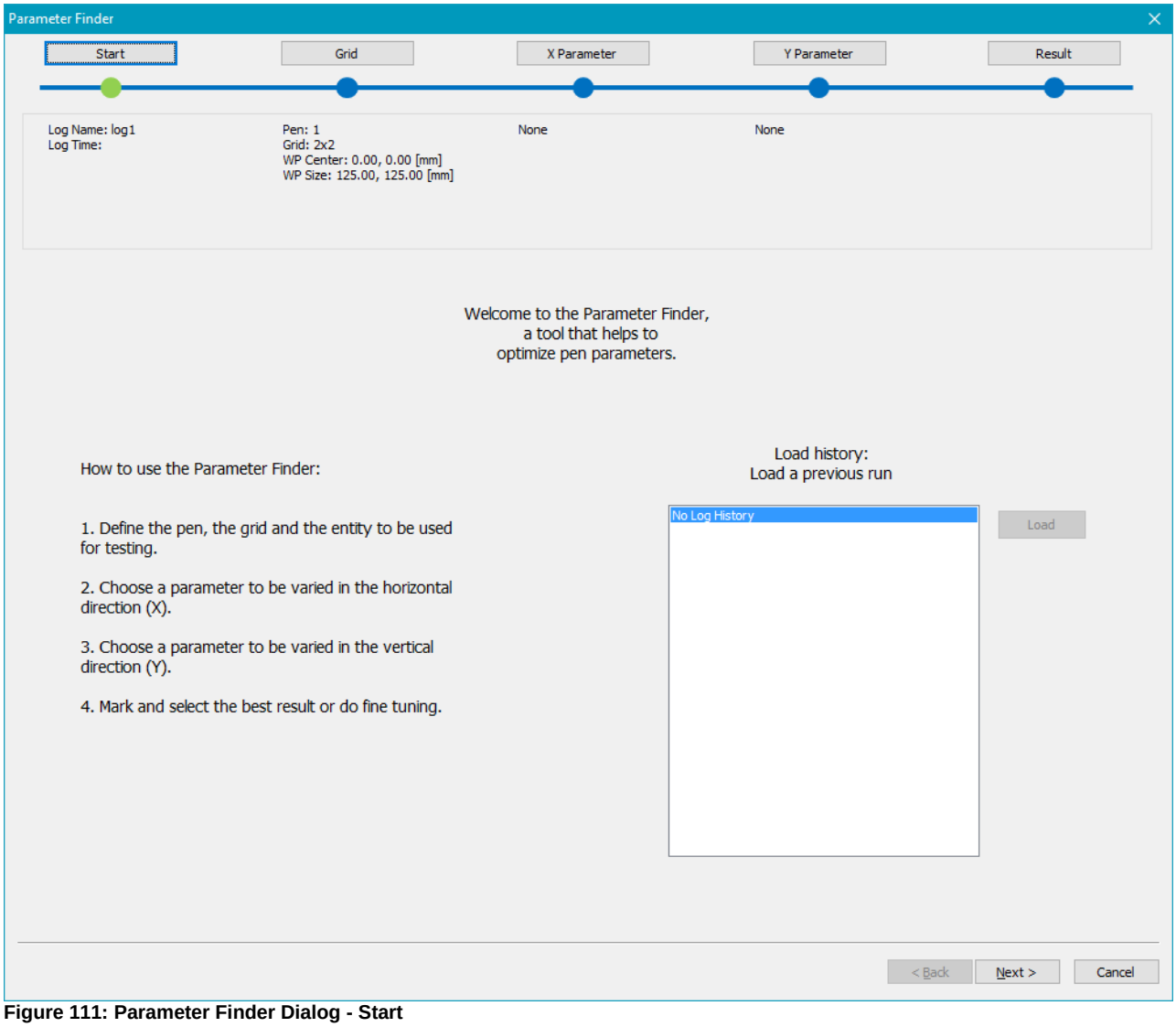


Figure 111: Parameter Finder Dialog - Start

Navigation is possible either with the buttons "Forward" and "Back" or by clicking directly on the button

representing the desired page in the navigation line at the top.
On the blue line, the current page is highlighted in green.
For each button of the navigation line, a summary of the currently applied values is given in the box underneath.
For each marking, a log is created which can be used to restore the parameters used for that particular marking by clicking on the load button.

1. Define the pen which should be optimized, the grid and the entity to be used for testing

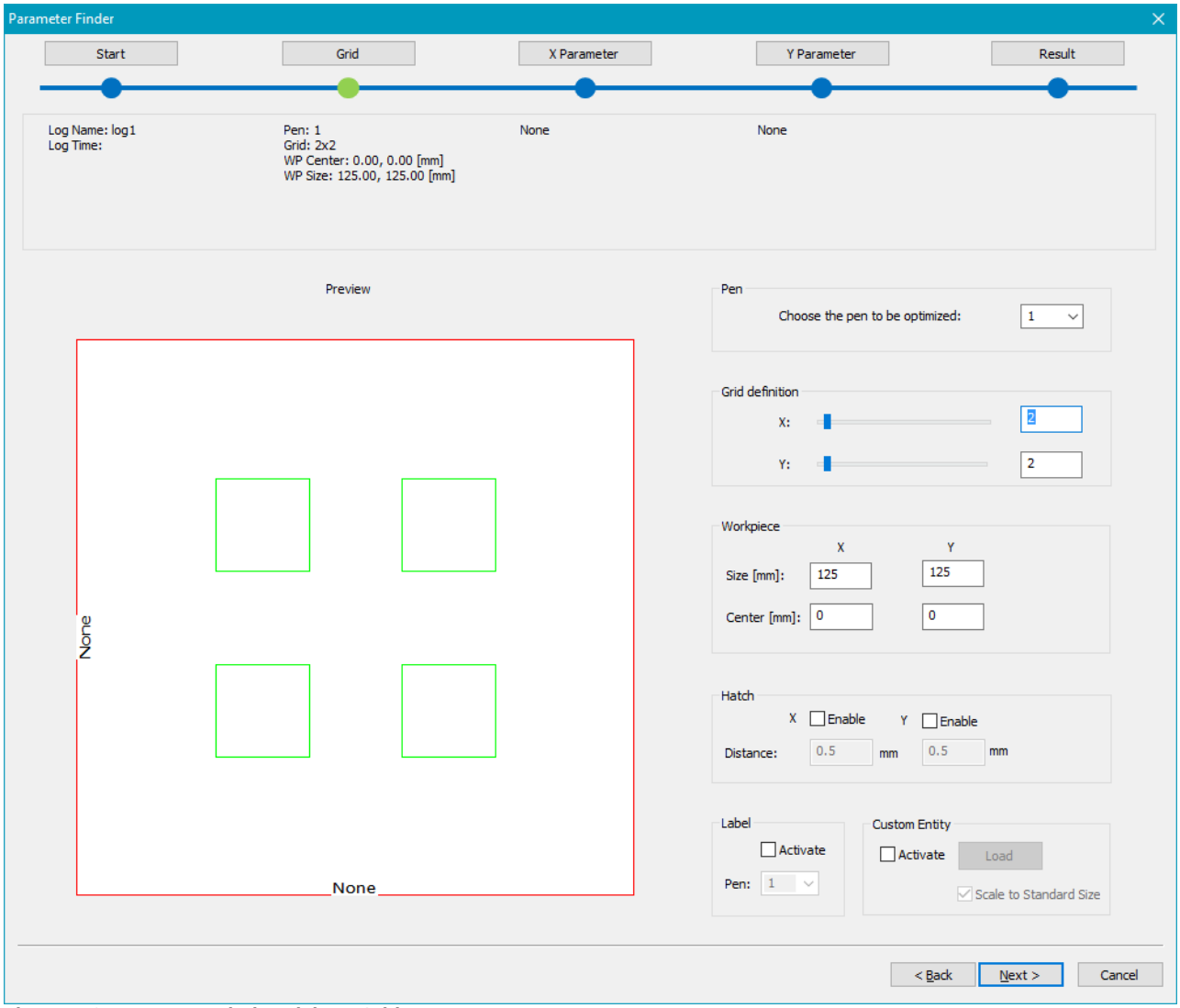


Figure 112: Parameter Finder Dialog - Grid

Preview: A preview is shown on the left half of the dialog. Changes made in the parameter definition will be automatically displayed in the preview.

Pen: Use the drop down menu to choose the pen which should be optimized. The parameter values set in this pen will be taken as start values for the parameters to be chosen on the pages X Parameter and Y Parameter of the Parameter Finder.

Grid definition: Define the total amount of entities in X and in Y direction.

Workpiece: Define the size and center of the workpiece. This is the area on which the entities of the grid will be equally distributed automatically.

Hatch: Activate a hatch (unidirectional) in X or in Y direction. If activated the distance between individual

hatch lines can be adjusted.

Label: Activate a label to be marked for each single entity. This label can be marked with a pen different than the pen which was chosen for optimization before.

Custom Entity: The default entity is a square which is scaled automatically with the grid size. To use an entity different than a simple square, the functionality "Custom Entity" can be used. When activated, a job can be loaded. The entities in this job will then be taken at each entry of the grid. Scaling of the job is possible to make sure it fits within the workpiece. The preview always shows rectangles as a representation of each entity or job.

2. Choose the "X Parameter" to be varied in horizontal direction

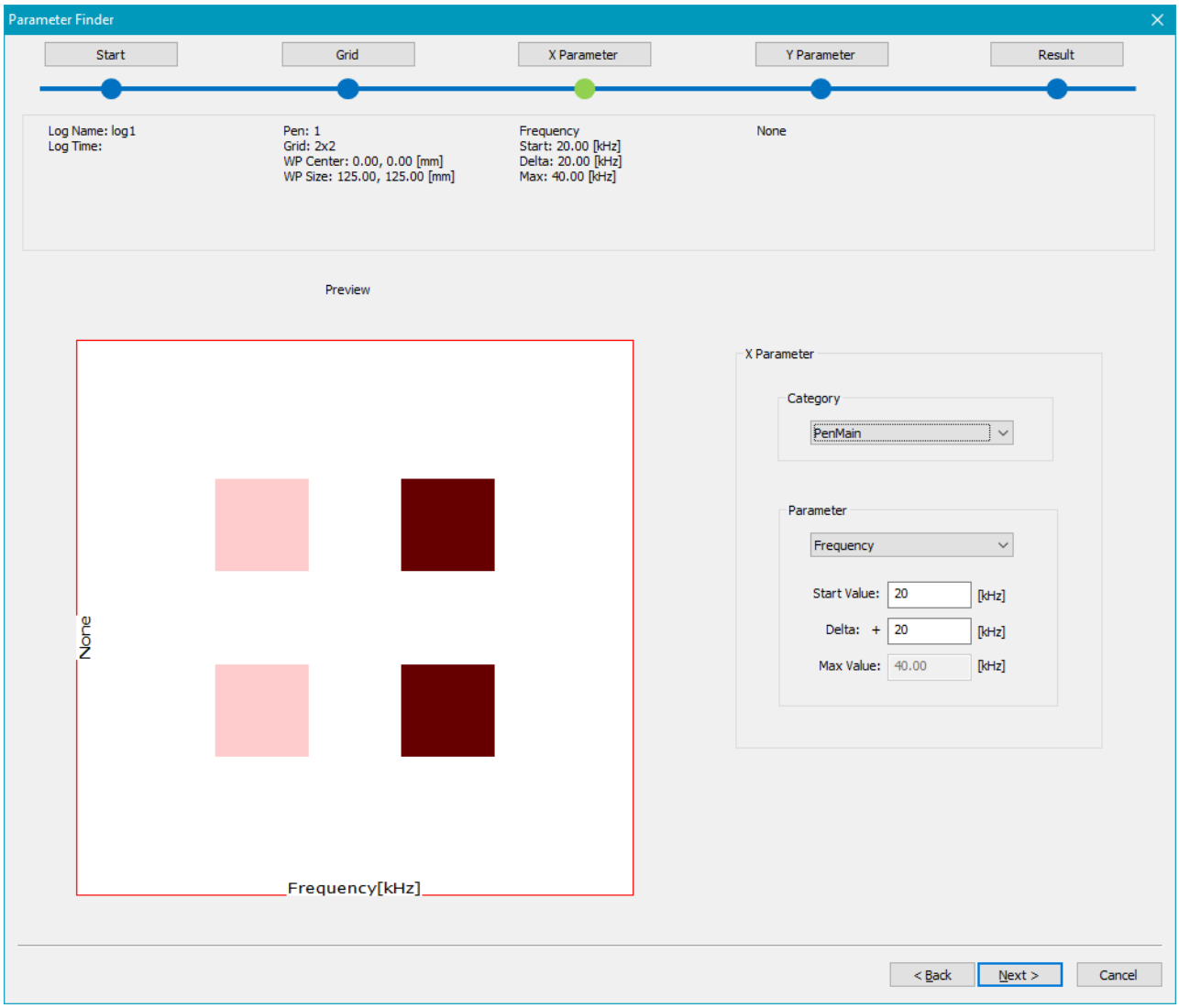


Figure 113: Parameter Finder Dialog - X Parameter

Preview: The color gradient indicates the variation of the parameter chosen in X direction. A label is given on the bottom according to the parameter chosen.

X Parameter: Specify the parameter which should be varied in horizontal direction.

Category: Choose a category from the drop down menu. Parameters are categorized according to the pen property pages.

Parameter: Choose a parameter from the drop down menu. Per default, the current value of this parameter

given in the pen is taken as start value. The start value and the delta (difference between two entities in X direction) can be adjusted. The resulting maximal value is calculated automatically.

3. Choose the "Y Parameter" to be varied in vertical direction

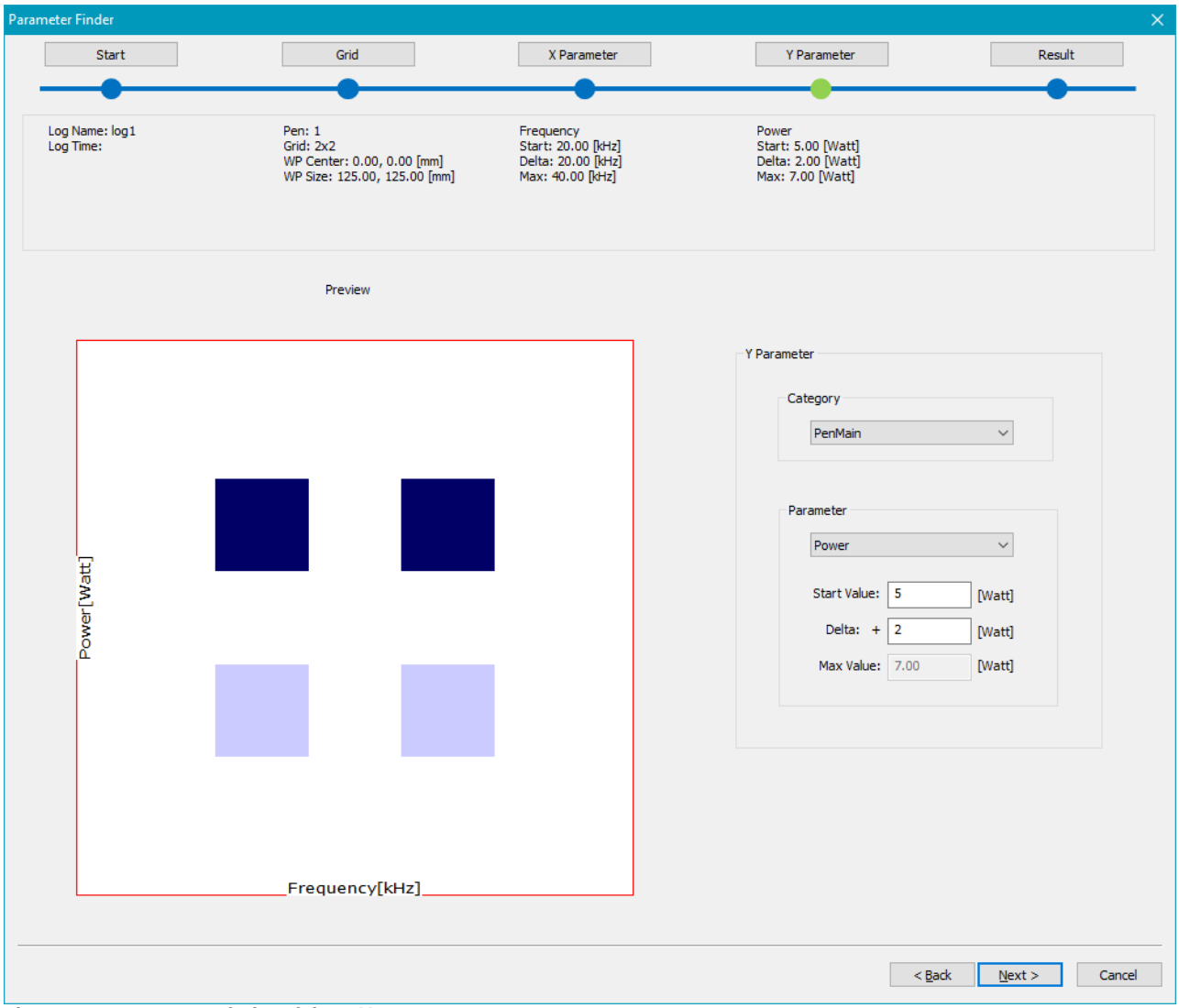


Figure 114: Parameter Finder Dialog - Y Parameter

Preview: The color gradient indicates the variation of the parameter chosen in Y direction. A label is given on the bottom according to the parameter chosen.

Y Parameter: Specify the parameter which should be varied in vertical direction.

Category: Choose a category from the drop down menu. Parameters are categorized according to the pen property pages.

Parameter: Choose a parameter from the drop down menu. Per default, the current value of this parameter given in the pen is taken as start value. The start value and the delta (difference between two entities in Y direction) can be adjusted. The resulting maximal value is calculated automatically.

4. Mark, selection of the best result, fine tuning

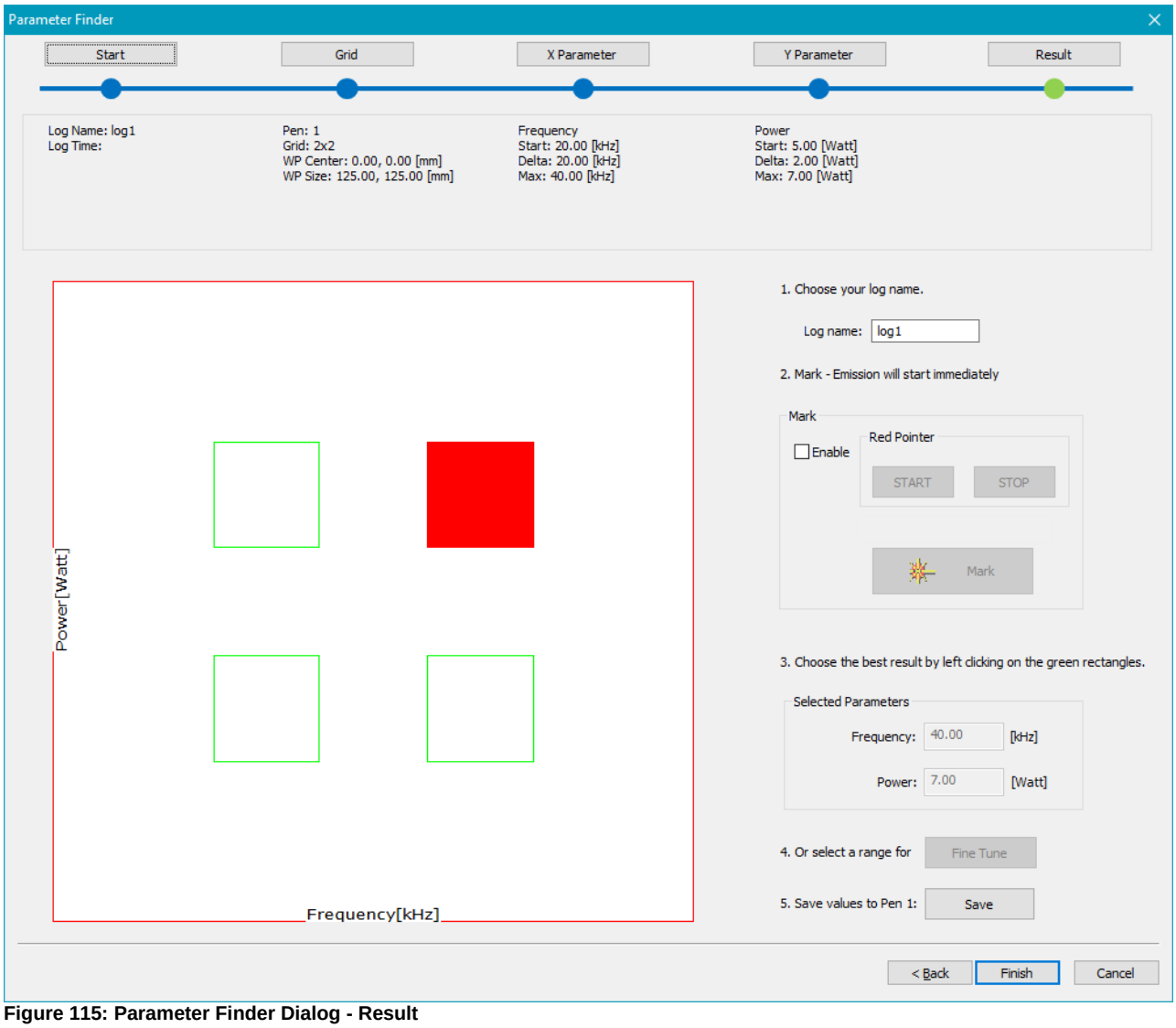


Figure 115: Parameter Finder Dialog - Result

Preview: On this page, the preview is used for selection of the best result (left click, red) or selection of an area for fine tuning (right clicking, blue).

1. Choose a log name: For each mark, a log is created which can be loaded on the Start page of the Parameter Finder. Specify the name of this log here.

2. Mark - Emission will start immediately: When enabled, the red pointer can be used with the buttons Start and Stop. Clicking on the mark button will start the marking.

3. Choose the best result by left clicking on the green rectangle: After having marked, the best result in reality can be detected and chosen by left clicking on the respective rectangle in the preview (will be highlighted in red). The exact parameter values will be given in the fields of "Selected Parameters". To save the chosen values to the pen, click on the Save button in point 5.

4. Or select a range for: If no best result can be identified, an area can be chosen for fine tuning. The area can either be chosen by right clicking on several rectangles or by dragging with pressed left mouse button (area will be highlighted in blue).

5. Save values to Pen x: If a best result has been chosen, the parameter values can be stored to the pen chosen on the Grid page.

7.2.6 Alignment and Spacing Toolbar

Align Left: Active if at least two objects are selected. The objects are aligned left to the left outside object.

Align Center: Active if at least two objects are selected. The objects are aligned horizontally to the center of their common outline.

Align Right: Active if at least two objects are selected. The objects are aligned right to the right outside object.

Align Top: Active if at least two objects are selected. The objects are aligned top to the top outside object.

Align Middle: Active if at least two objects are selected. The objects are aligned vertically to the center of their common outline.

Align Bottom: Active if at least two objects are selected. The objects are aligned bottom to the bottom outside object.

Spacing Horizontal: Active if at least three objects are selected. The objects are distributed evenly inside their common outline.

Spacing Vertical: Active if at least three objects are selected. The objects are distributed evenly inside their common outline.

Spacing Advanced: Opens a dialog where more specific spacing can be done. See dialog Spacing Advanced.

Center Both: The selected entities would be centered both horizontally and vertically.

Center Horizontally: The selected entities would be centered horizontally.

Center Vertically: The selected entities would be centered vertically.

7.2.7 Extras Toolbar

This toolbar can be used to directly access the functions for Splitting a job and for Step/Repeat marking. After the same functionality is available via the related Extras menu items it is disabled by default. This toolbar can be activated within the settings before it can be used. It offers following functionalities:

Drop down menu: This menu allows you to select a certain splitting mode. Then this mode can be edited using the button *Splitting Settings*.

Splitting Settings: This button offers direct access to the splitting settings dialog where several splitting parameters can be configured.

Resplit Job: This button is enabled when the splitting mode is activated for a job. This is the case when the check box right beside that toolbar button is selected. It can be used to re-split the current job in order to make changes to that job become valid for the splitted data too.

Step/Repeat Settings: This button offers direct access to the settings dialog for the Step/Repeat parameters.

Reset Position: In case the Step/Repeat mode is activated (the check box right beside that button is selected) it can be used to reset the position of the current object. That causes - depending on the Step/ Repeat mode - either a repositioning of the used geometry to its original position or a movement of the used drives so that the starting position is reached.

Bitmap Splitting: If a scanner bitmap is present in the entity list then this bitmap can be splitted in order to mark on a round surface.

7.2.8 Stepper Position Toolbar

This toolbar is available only if USC-2 stepper type 14 is used. If activated you can see the axis position of all defined axis. The values corresponds to the values displayed in the control property page. The units are mm for straight axis and ° for angular axis.

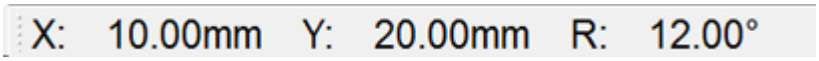


Figure 116: Stepper position toolbar

7.2.9 3D Surfaces Toolbar

This toolbar is available only if the option Optic3D license is present. The toolbar is showing mode drop-down list , settings button and enable checkbox. How to use the 3D Surface feature is explained here.

Figure 117: 3D surcfaces toolbar

7.2.10 Special Sequences Toolbar

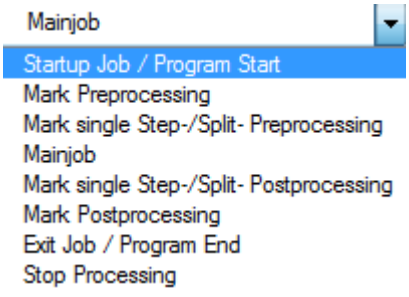


Figure 118: Special Sequences Toolbar

The Special Sequences Toolbar can be made visible in *Settings → System→ IO → Special Sequences → Jobs Toolbar*. This Toolbar allows you to switch between different jobs that have a specific purpose and are executed at a specific moment in program flow. When such a special job beside the main job is selected within the toolbar a dialog opens where the elements of that sequence can be edited. These special sequences are not available in flash or trigger-mode.

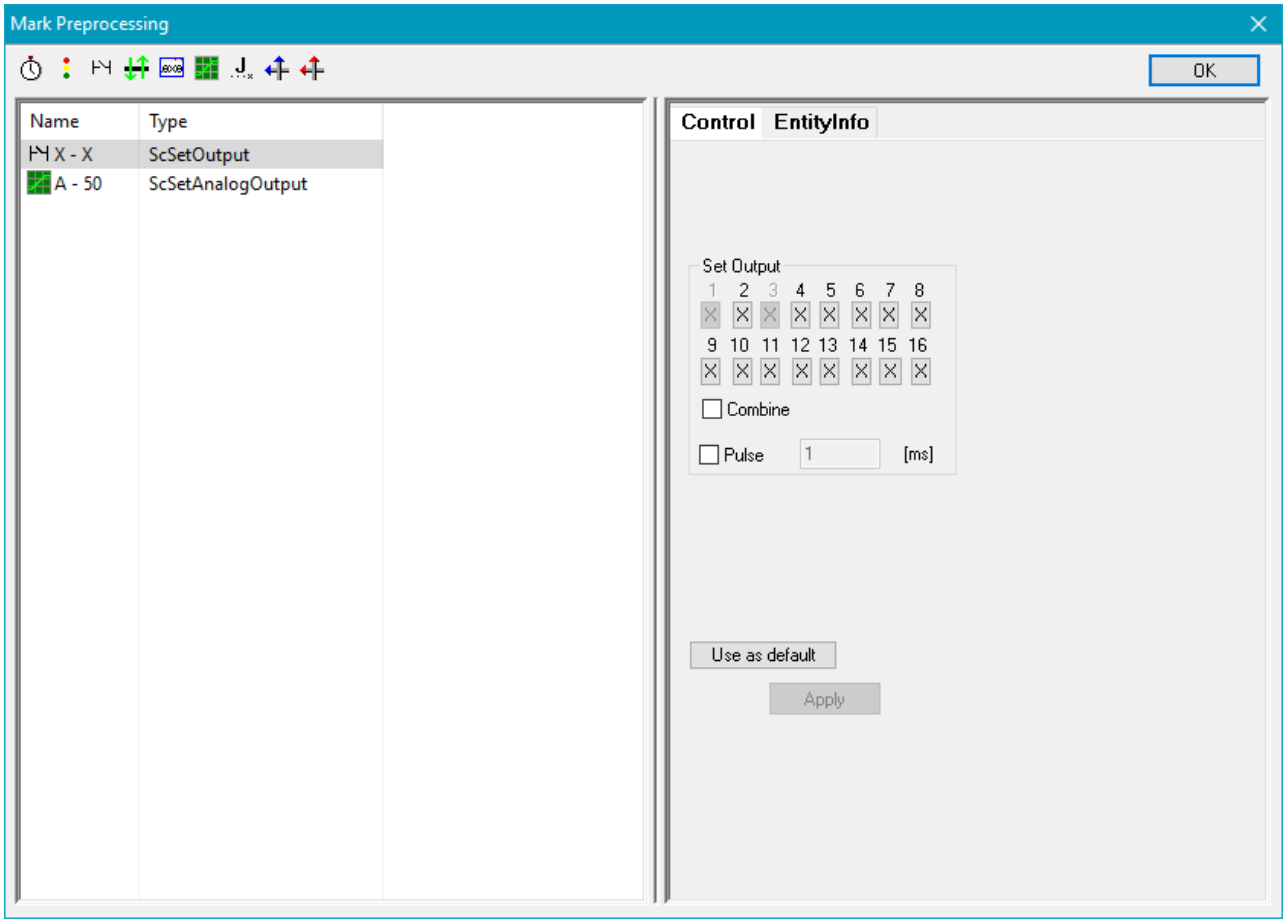


Figure 119: Mark Preprocessing Dialog

Here the toolbar elements that are known out of the main window can be used. Their settings and parameters can be edited in a similar way. The following special sequences (special jobs) can be chosen:

Startup Job / Program Start: If a job is defined here it is executed directly after program startup to e.g. initialize special external equipment. A marking operation performed by this preprocessing job is not ranked as a normal operation like the mainjob and therefore not counted as one quantity. Due to security reasons it is recommended to avoid potentially dangerous operations like laser marking operations or heavy movements within this job.

Mainjob: This job is the default job. The items defined here are processed during a marking operation and are counted as one quantity each. This job is the same like the only one that is executed in case the special job functionality is turned off.

Exit Job / Program End: This job is the counterpart of the Startup Job. It is executed when the program is shutting down. Such a Postprocessing Job can be used e.g. to deinitialize external equipment. If a marking operation is performed here it is recommended to avoid potentially dangerous operations within this job.

Mark Preprocessing: This Job is a specific one that is executed directly before the main marking job is executed. If a Splitting or Step/Repeat operation is performed, this job is executed once before the full operation starts. In SAM3D the mark preprocess will only be performed before the first slice of the job. If you

start the build with another slice but the first one no mark preprocessing job will be executed.

Mark Postprocessing: If the marking of the main job is finished or when the user has pressed the stop-button during marking then the job that is defined here is executed.

Here no dangerous operations like additional marking operations should be executed. There would be the high risk that if somebody presses stop but instead of stopping an other marking process is started. This special sequence should be used only for deinitialization operations that are necessary after marking, e.g. to set some outputs to defined values.

Mark single Step- / Split- Preprocessing: If Splitting or Step/Repeat is used a marking in progress sequence can be defined which will be done e.g. after an axis movement has finished. This sequence will be executed right before the laser starts to mark the next mark job.

Mark single Step- / Split- Postprocessing: If Splitting or Step/Repeat is used this sequence will be executed right after the making procedure has finished and before an axis movement starts.

Slice Preprocessing: This Job is available in 3D-mode only and is executed directly before a single slice of a 3D object is marked. Here several control elements can be added e.g. to move a Z-table that modifies the vertical position or to set specific output pins that perform that task.

Slice Postprocessing: This Job is available in 3D-mode only and is executed directly after a single slice of a 3D object was marked. Here several control elements can be added e.g. to move a Z-table that modifies the vertical position or to wait for specific input pins.

Stop Processing: This task will be performed in case of external stop (via OPTO_IN_1) is recognized if the stop button in the mark dialog is clicked or if the control command client ScStopMarking () is recognized. Here, several control elements can be added for example to move a Z-table that changes the vertical position or to wait for dedicated input pins.

Using the toolbar shown above it is possible to switch between these jobs and then to perform all normal operations for the actually selected job. Because these special jobs are no common operation they are disabled by default so that only the mainjob is visible and useable. The pre- and postprocessing jobs can be enabled using the Special Sequences Settings.

The Special Sequence settings can be saved in each Job file with the checkbox "Save pre/ post jobs in 'sjf' job file" enabled in File → Job Properties dialog in a SAMLight version later than 3_6_5_0011.

Figure 120: Job Properties

When you load a job file, you can check in the loading dialog whether there is special sequences information already included in the job file with "SAMLightJobs".

In a version later than 3_6_5_0011, by loading a job file with special sequences, the special sequences in job file would overwrite the special sequences in the global SAMLight settings. Please note that "Startup Job / Program Start" and "Exit Job / Program End" are not saved in the job specific special sequences.

In a version earlier than 3_6_5_0012, by loading a job file with special sequences, this information can be read in the loading dialog, but the special sequences would not be loaded.

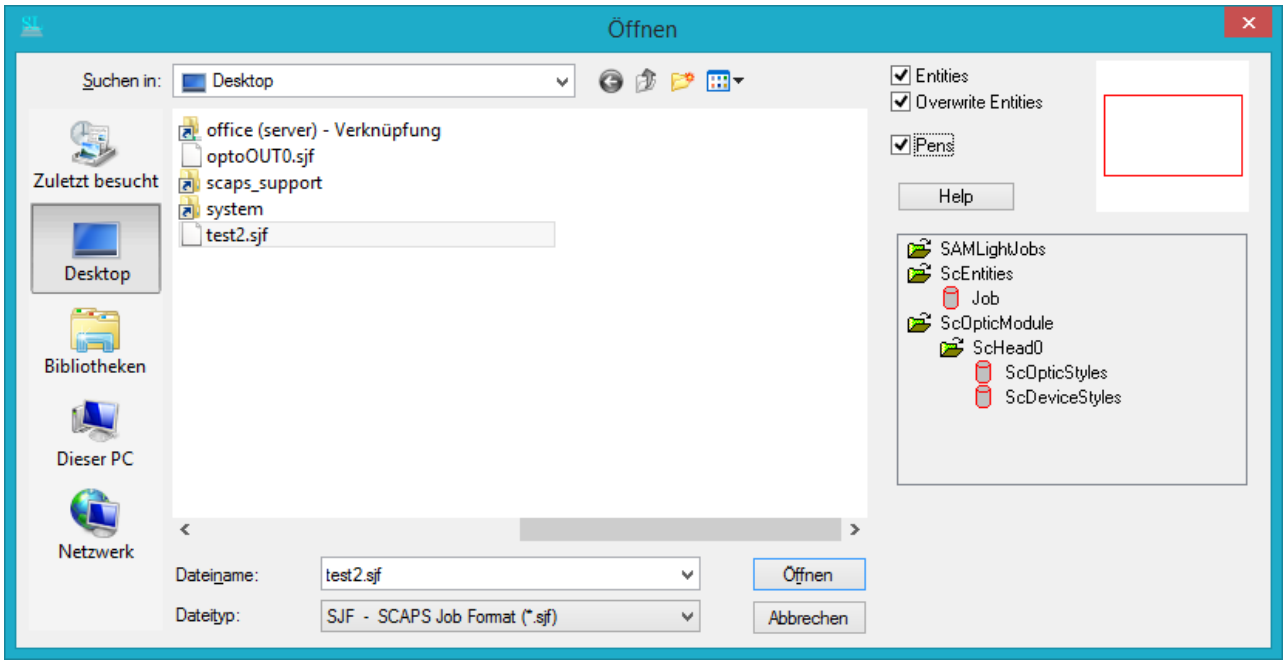


Figure 121: loading dialog

7.2.11 Analog In Toolbar

The Analog In toolbar is available only if USC-2/3 is used. It can be configured for the two 10 bit analog inputs (values from 0-1023) of USC-2/3 card. The input signals must be connected to ANA_CH_0 or ANA_CH_1 in respect to AIN_GND. The current bit value of the analog input bit can be checked at sc_usc_server.exe in visible mode at InfoView.

The input voltage should not exceed 10.3 V.

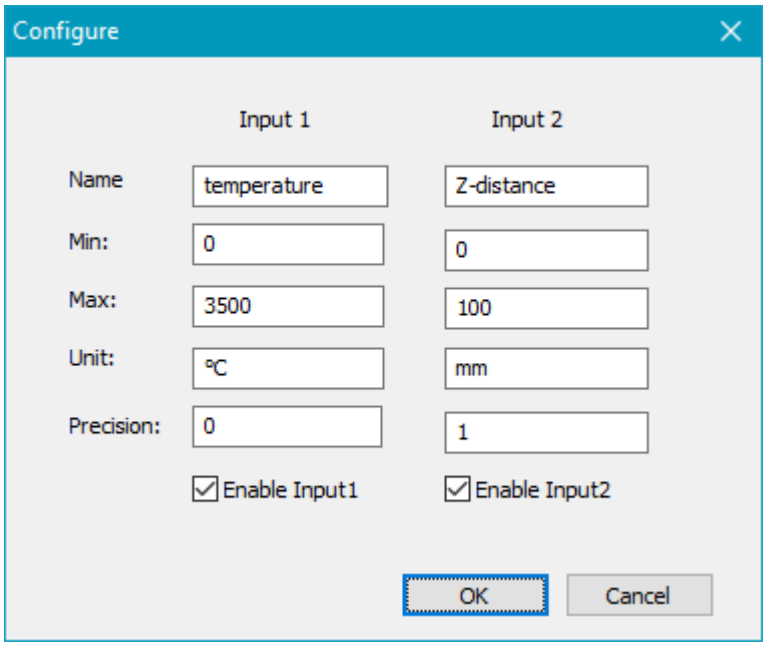


Figure 122: Analog In dialog

Input 1: Settings for analog input ANA_CH_0.

Input 2: Settings for analog input ANA_CH_1.

Name: Here a name can be set for the Analog Input bit that can be used to decide what its purpose.

Min: Here the minimum of the unit range have to be set for the Analog Input bit, which should corresponds to a digital value of 0 or 0 V.

Max: Here the maximum of the unit range have to be set for the Analog Input bit, which should corresponds to a digital value of 1023 or about 10.3 V.

Unit: Here the physical unit of the Analog Input bit can be set.

Precision: Here the number of decimal places can be set. Please keep in mind that the displayed numerical resolution is limited to (Max value - Min value) / 1024.

Enable Input1: Activates the display of analog input ANA_CH_0.

Enable Input2: Activates the display of analog input ANA_CH_1.

If activated the Analog In toolbar displays the converted voltage according to settings of Analog In dialog in accordance with the set names and units there.

Figure 123: Analog In toolbar

7.3 Entity List

The Entity List appears in two different modes:

- Entity List
- Point Editor

The *Entity List* shows all the entities in the current job and visualizes their logical structure / hierarchy in a ListView. It provides operations for moving, copying and sorting entities as well as exploring entities that contain sub-entities. The *Point Editor* is a tool to visualize and modify the point description of an entity.

7.3.1 Entity List

The Entity List shows all the entities in the current job and visualizes their logical structure/hierarchy in a ListView. It provides operations for moving, copying and sorting entities as well as exploring entities that contain sub-entities.

Definition entity: An entity is either an element or a container or a group of entities. An element keeps real geometric data like lines, points and pixels and a so-called container contains elements.

Example: For example a simple description of a desk: The desk is a group of the sub-entities *legs* and *desk_top*, where the *desk_top* is a container that contains the geometry of the desk top. The group *legs* contains the sub-entities *leg1*,...,*leg4* which are containers that contain the geometry of a leg.

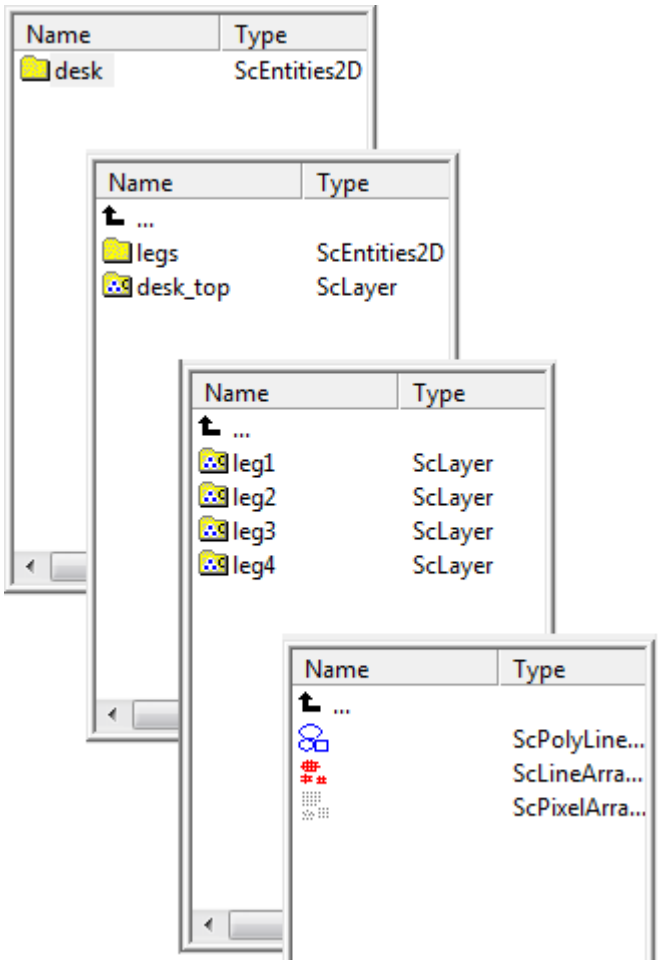


Figure 124: Entity List

The last picture shows the elements of the container *leg1*.

The levels of the Entity List: To understand the levels think again of the desk example above. The entity *desk* is on level 1 and its contents are on level 2. The entity *legs* is a sub-entity of *desk* (level 2), its contents are on level 3 and so on. For a detailed description of the object hierarchy see chapter Object Hierarchy.

Operations:

Exploring: By double clicking an entity which contains other entities (for example a Group) the ListView will step inside and show the entities inside the selected one.

Move: This is the standard operation of a drag and drop process: It moves the selected entity or entities. Here following possibilities are available:

- Move an entity within the job to change its position
- Move a grouped entity out of the group by dragging it onto the level up arrow
- Move an entity into an existing group by dragging it onto the target ScEntities2D while the *ShiftKey* is hold down
- Move an entity/ entities in an group by options to previous, to next, to top and to bottom.

Copy: This works in the same way as Move, pressing the CONTROL-key additionally during the drag and drop process puts a copy of the selected entity in the drop folder.

Remark to Move/Copy: Not every move or copy operation is allowed. So, for example the move of a line array into a PolyLines group or the move of a element inside a container out to an other container can not be executed since this would corrupt the data consistency.

Group: Several entities of the same level can be grouped by clicking on them while holding the *ShiftKey* down and choosing *Group* from menu *Edit → Group*. This procedure will create a new group on the actual level that gets the selected entities as sub-entities.

The following operations are provided by pressing the right mouse button. Here the entity list context menu appears that offers the following functionalities:

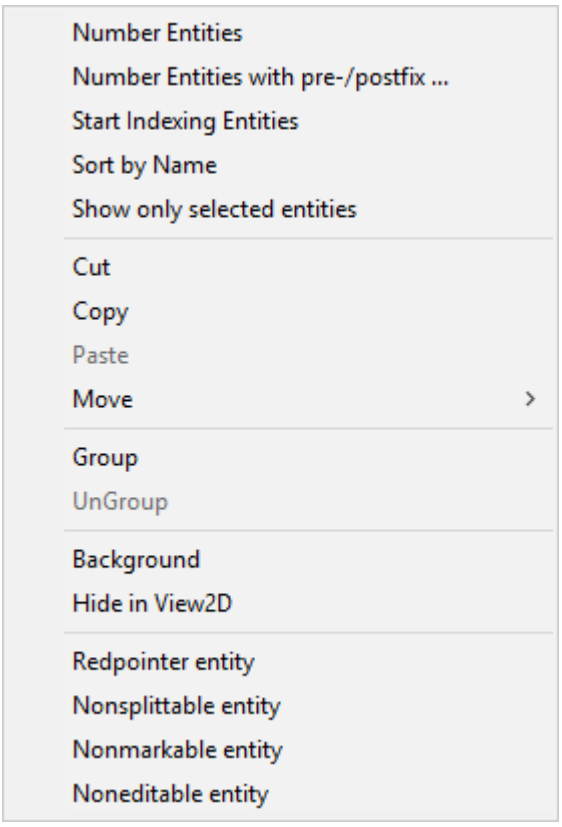


Figure 125: Entity List Context Menu

NumberEntities: All Entities, regardless if they are selected or not, are being named with a number starting from 1.

NumberEntities with pre-/postfix: Opens the corresponding dialog. All Entities are being named starting

from a freely selectable number with with a freely selectable incremental value and / or a defined pre or postfix.

Start / Stop Indexing Entities: Opens the corresponding dialog If selecting this the first time all Entity Names are being deleted. Now the user can choose between indexing options. If this is done the user may index any entity by clicking on it with the left mouse button. By default the indexing starts with 1 and is incremented by 1 after each mouse click. If the user wishes to return to normal mode again, he selects *Stop Indexing Entities*.

Sort By Name: Selecting this function sorts the entities on the actual level by their name. The name comparison is case sensitive and starts at the first letter of the name.

Remark to SortByName: The entities with the names *1*, *2*, *12* will be sorted to *1*, *12*, *2*, because the first letter of entity *12* is 1 and of entity *2* is 2. So the entity *2* gets the position after entity *12*. To solve that problem the user may name the entities like *001*, *002*, *012*.

Remark to changing the order of the entities: The entities of a group will always be marked from the top first to the bottom last in the ListView. So changing the order in the ListView can be used to change the order of exposure during the mark process.

Cut: The currently selected entities are cut out of the list and they are put into the internal entity clipboard for use in further operations.

Copy: Comparing to *Cut* using this operation the selected entities aren't removed from the entities list but a copy of them it put into the internal entity clipboard too.

Paste: If there are some entities held in the internal clipboard they can be copied out of it into the entity list using the current position.

For this operations the same restrictions are valid than described for the Drag-and-drop-copy operation described above.

Move: moves the position of the selected entity or entities in the entity list. The sequence in the entity list determines the sequence of marking.

Group: Groups the selected entities and puts them into an entities group. See also chapter Object Hierarchy.

UnGroup: Ungroups the selected entities group. The view level of all entities inside the group will be decreased by one. See also chapter Object Hierarchy.

Back-/Foreground: This option can be used to use a selected bitmap entity as background object or to use a background object as normal entity. When a bitmap-object is put to the background, it isn't selectable in the View 2D any longer but only within the object list. Additionally it won't be marked. Such images can be used as a template for creating new objects, here e.g. a technical drawing can be used. After importing a bitmap to the current job it is possible to scale and position it so that the resulting dimensions are correct and that it fits to the desired working area. After that was done it can be put to the Background so that it doesn't influence the current job any longer but can be used as template. If that background drawing isn't required any longer, it can be selected within the Entity List and then put to the Foreground using this menu item again. Now it is possible to remove it from the job. In the list entities that are used as background are marked by a white folder symbol instead of a yellow one.

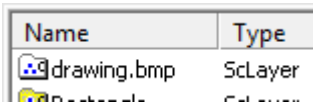


Figure 126: Hidden Entity in View2D

Hide-/Show in View2D: Compared to the previously described background option this one works different. When an entity is set to mode "Hide in View2D" it is no longer visible in View 2D but it still can be marked. It can be selected within the Entity List and it can be moved, scaled and translated when it is selected. This function can be used to hide elements in a job that overlap each other. That is useful in cases where some

of these overlapping objects can't be seen because the other ones cover them completely. Using that *Hide* option some of the covering entities can be made invisible without removing them from the marking result. In the entity list entities that are hidden from the View2D are indicated by a dimmed folder symbol instead of a solid yellow one.

Redpointer Entity: When an entity is marked as Redpointer Entity this entity will only shown when using the Red Pointer if the checkbox "Redpointer Entity" is activated in the Mark Dialog.

(Non)splittable Entity: When an entity is marked as nonsplittable its appearance within the View2D doesn't change and it is highlighted by red brackets within the Entity List. An entity that is marked in this way will behave different for splitted jobs: Independent from their position within the original job they will be marked before the splits are processed. If an entity state is changed to splittable or nonsplittable when the splitting mode is active, that job has to be resplitted. So this functionality can be used to exclude some parts of the job from being splitted. These excluded objects will be marked as one piece when a splitted job is processed. If the splitting mode isn't active this selection doesn't influence the job.

(Non)markable Entity: This can be used to exclude the selected entities from the marking process. By selecting again the selected entities are being included again.

(Non)editable Entity: This can be used to define an entity as non editable. But the entity is still included in the marking process.

7.3.1.1 Index Entities with pro-/postfix Dialogs

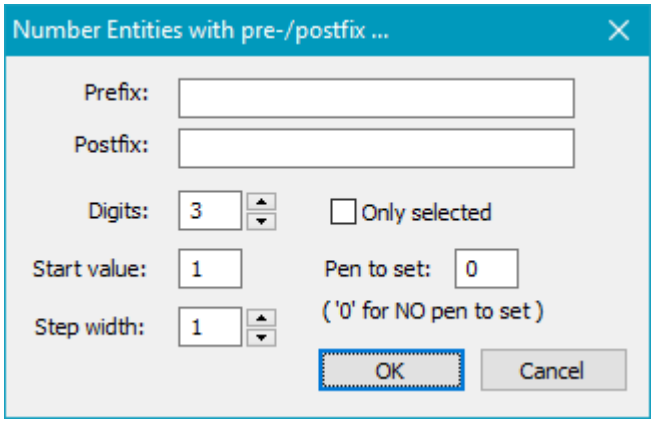


Figure 127: Number Entities with pre-/postfix Dialog

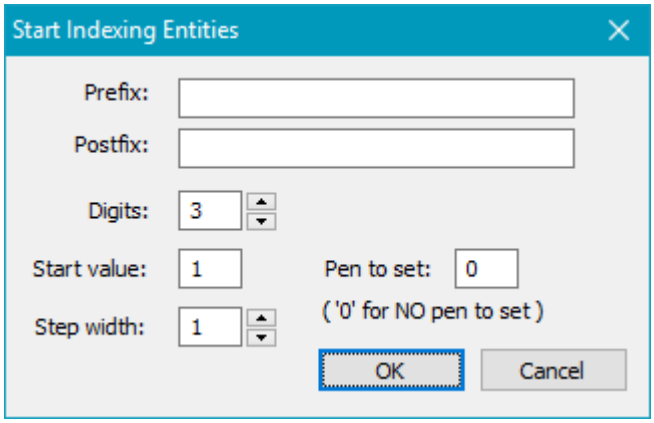
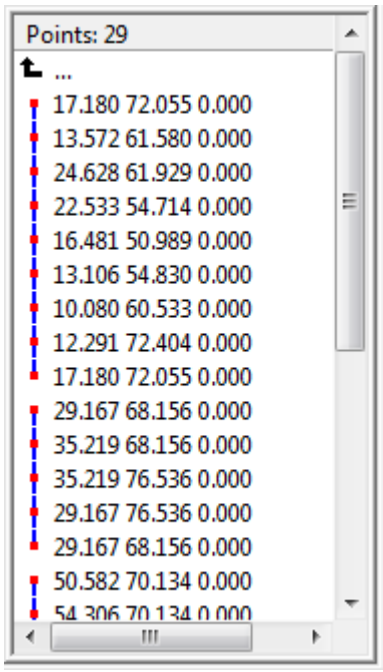


Figure 128: Start Indexing Entities Dialog

7.3.2 Point Editor

Purpose: The Point Editor is a tool that visualizes and modifies the point description of an entity.
Point Editor View of the Entity List: Before doing a point editing operation please switch to the point editing mode (use the Object Toolbar, button ). Then select one or more objects in the same way as described in View 2D → Selection. If more than one object is selected the objects must be grouped (use *Edit → Group*) to perform point editing. One can also perform these two steps the other way around.

After selecting objects in point editing mode the Entity List will switch to the Point Editor View as shown in the screenshot below.



The Point Editor View shows all points of the selected objects in a single list.

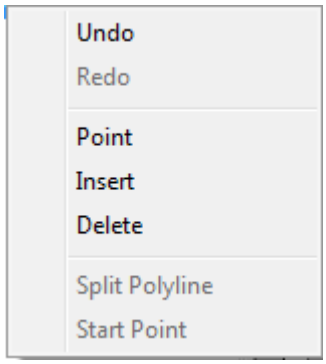
All the points which belong to one PolyLine are connected by a blue line in that list.

The red dots mark single points and the 2D/3D coordinates of a point are displayed next to the red dot. The coordinates are 3D only with Optic3D.

The caption of the Point Editor View displays the number of points.

Figure 129: Point Editor View

Context Menu: To modify the points in the list or the list itself the context menu provides five operations. They are described next to the screenshot below. Select one or more items from the list by mouse click while keeping the *Ctrl* pressed. After the selection keep the *Ctrl* pressed and click the right mouse button. Then the context menu will be displayed. The points of the selected objects are marked in the View 2D by black dots. Clicking with the right mouse button on such a black dot will display the context menu too.



The context menu provides:

Undo: Undo the last operation.

Redo: Redo the last Undo.

Point: Opens the Edit Items dialog like it is shown below.

Insert: Inserts a new point in between the two adjacent points.

Delete: Deletes the selected points.

Figure 130: Context Menu

Point Editing with the Edit Items dialog: To edit one or more points use the Edit Items dialog shown below. One can get this dialog via the context menu (described above) or by double clicking on a point item in the list.

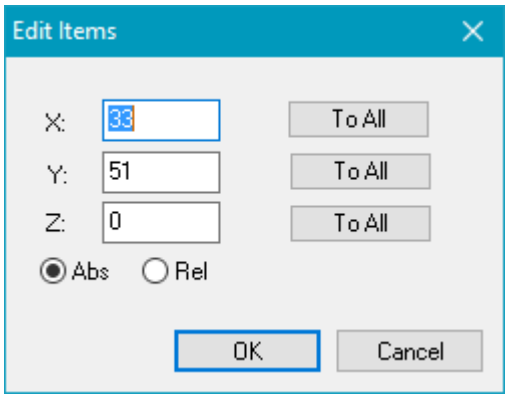


Figure 131: Edit Point Coordinates Dialog

X,Y,Z: Edit these values to change the coordinates of a single point. Remark: Z coordinate is only available with Optic3D.

The Z-Coordinate is only available with the Optic3D Option.

Radio buttons: There are two update modes for updating point coordinates which can be switched by the radio buttons. Abs is the default mode.

Abs mode: The point coordinates are updated by a substitution of the old coordinates by the new ones.

Rel mode: The point coordinates are updated by adding the new values to the actual coordinates.

To All: These buttons are only active if more than one point is selected. To update the X coordinates of all selected points change the value of the X field and click the *To All* button next to it. Updating of the Y or Z coordinate for all points is done analogously.

Examples: To project all selected points on the plane Z = 1 select update mode Abs, change the value in the Z field to 1 and click the ToAll - Button next to the Z field. To move all selected points in Y direction by 3 units select update mode Rel, change the value in the Y field to 3 and click the ToAll - Button next to Y input field.

7.4 View 2D

Purpose: The View2D displays the geometry of all the entities in the current job. It provides basic operations for the manipulation of objects.

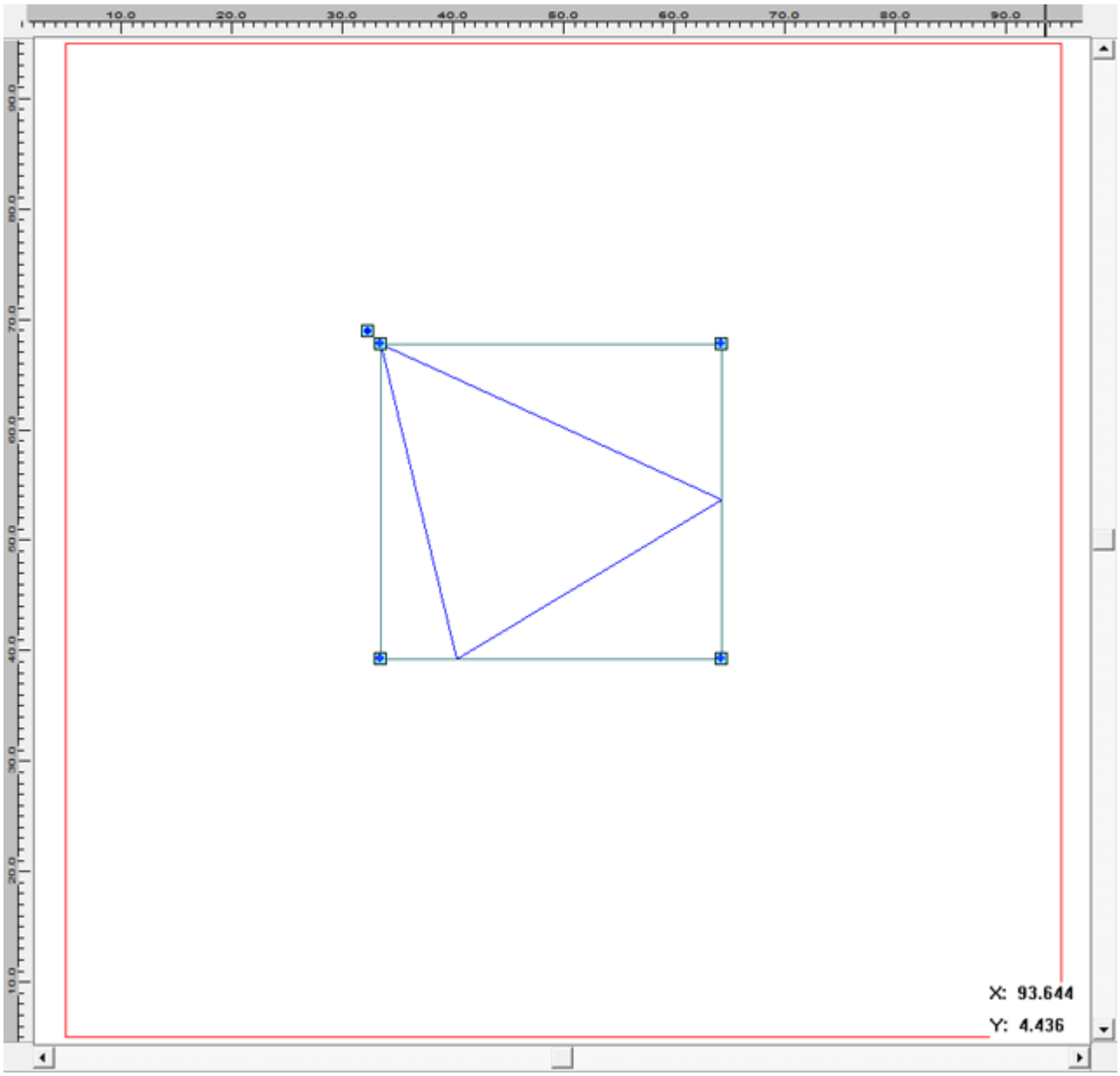


Figure 132: View 2D

Overview: The picture above is a screenshot of the View2D. The red bounding box marks the boundary of the workspace/working area. The values in the lower right corner are the coordinates of the mouse pointer (if the mouse pointer is inside the View2D). The axes of the coordinate system are marked on the left and the upper boundary of the View2D and its origin is in the lower left corner of the workspace/working area. For more details see chapter Operations and Print Preview.

7.4.1 Operations

Creation of objects: To create objects in the View2D use the Object Toolbar.

Selection: Before doing a selection operation switch to the select mode (use the Object Toolbar, button 

), unless you want to do a point editing operation (see Entity List → Point Editor).

First way: Selection via the Entity List: Select an entity, i.e. an object or a group of objects, by clicking on it in the Entity List. This will show a modify box for the selection, like it is shown for the triangle in the screenshot. To select more than one entity hold down Ctrl and select the entities by clicking on them in the Entity List.

Second way: Selection via the View2D by mouse interaction: To select one or more objects draw a bounding box that covers all objects to select. To draw this bounding box click the left mouse button at the chosen position of the left upper corner and move the mouse while keeping the left mouse button pressed to the chosen position of the right lower corner of the bounding box. Then release the left mouse button. This will show a modify box that contains all the objects in the bounding box.

Third way: Selection via the View2D by mouse clicks: To select only one object click on it in the View2D. To select more than one object, hold down Ctrl and select the entities by clicking on them it in the View2D.

Manipulation of objects: Before an object can be modified it must have been selected. To do this follow the instructions described above (*Selection*). The modify box, shown after the selection, provides three operation modes for transformations which can be switched through by the button.

Translation mode: In this mode the corners of the modify box show the symbol . To translate the selected object click on this symbol, keep the left mouse button pressed and drag the mouse to the new position.

Scaling mode: In this mode the corners of the modify box show the symbol and the edges show the symbol . To scale the object in x and y direction simultaneously click on a corner symbol, keep the left mouse button pressed and drag the mouse. To scale only in x or y direction click on an edge symbol and do the same.

Rotating and beveling mode: In this mode the corners of the modify box show the symbol and the edges show the symbol . To rotate the object around the rotation centre click on a corner symbol, keep the left mouse button pressed and drag the mouse. To bevel the object click on an edge symbol and do the same.

The manipulations described above are not object specific. To modify object specific properties use the Entity Property Sheet.

View2D Settings: To modify the settings of the View2D use the dialog *Menu bar → Settings → View*.

Context Menu: When right click somewhere in the View2D the context menu appears:

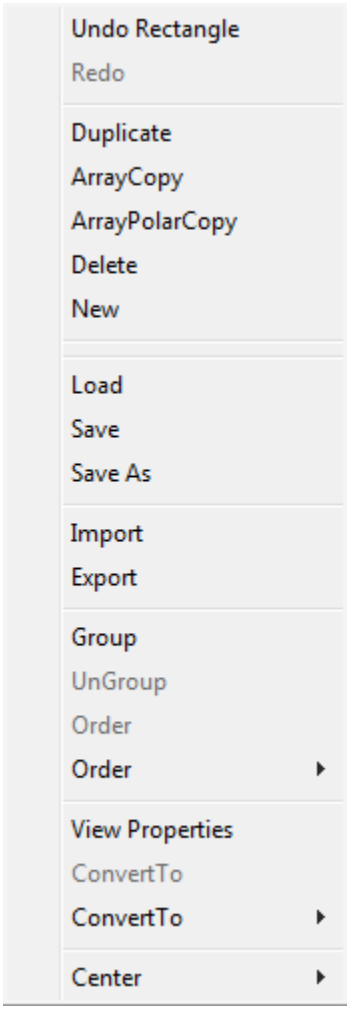


Figure 133: Context Menu

Duplicate: Make a copy of the object

ArrayPolarCopy: Make copies of the object and place them on a circle line. If you click on it a dialog will appear to ask you the Starting Angle, Number of Copies, Radius and Center X,Y.

Import / Export: Same function as in Menu Bar

Flip: Reverse marking order for Polylines.

7.4.1.1 View Properties

When right click somewhere in the View2D the context menu appears with the entry View Properties.
In the following, the three tabs General, Colors and MultiHead of the View Properties dialog are explained.

7.4.1.1.1 General

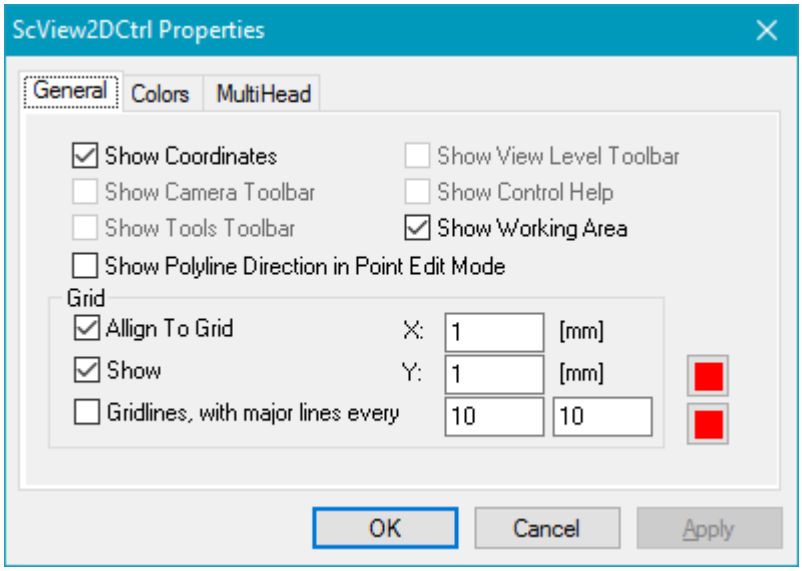


Figure 134: ScView2DCtrl_General

With these check boxes in the area above, the user can decide which information should be displayed.

Grid:

Align To Grid: If checked each new object placed in the View 2D will be aligned to the grid.

Show: If checked the grid will be displayed in the View 2D.

X, Y: These two values define the grid size.

7.4.1.1.2 Colors

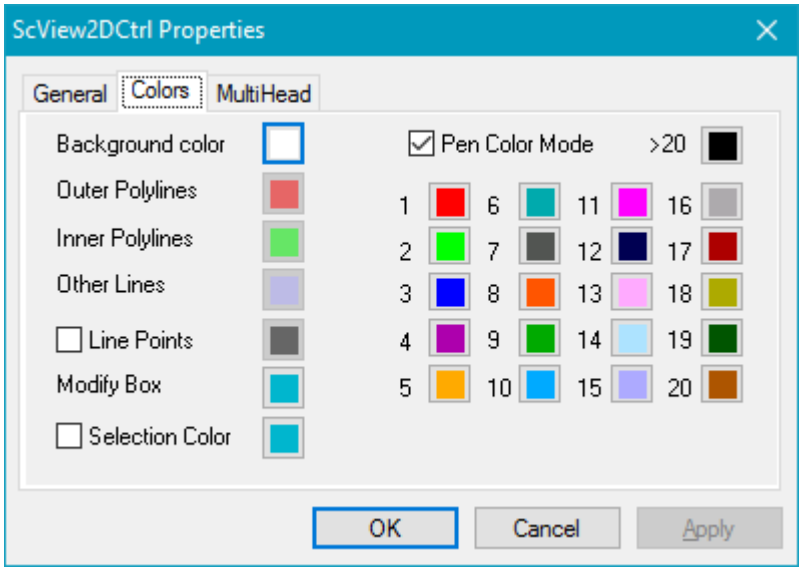


Figure 135: ScView2DCtrl_Colors

With these check boxes the user can define the colors of background, point of crossing lines and the selected entities. Here, the user can also define the colors of the different pens (also possible in the View Settings Dialog).

Default pen colors:

Pen number	Color	RGB HEX	RGB DEC
1		ff0000	255 000 000
2		00ff00	000 255 000
3		0000ff	000 000 255
4		aa00aa	170 000 170
5		ffaa00	255 170 000
6		00aaaa	000 170 170
7		555555	085 085 085
8		ff5500	255 085 000
9		00aa00	000 170 000
10		00aaff	000 170 255
11		ff00ff	255 000 255
12		000055	000 000 085
13		ffaaff	255 170 255
14		aaffff	170 255 255
15		aaaaff	170 170 255
16		aaaaaa	170 170 170
17		aa0000	170 000 000
18		aaaa00	170 170 000
19		005500	000 085 000
20		aa5500	170 085 000
>20		000000	000 000 000

Table 14: Default pen colors

Pen colors are used, when Pen Color Mode is enabled.

Default View2D object colors:

View2D object	Color	RGB HEX	RGB DEC
Background color		ffffff	255 255 255
Outer Polylines [a]		ff0000	255 000 000
Inner Polylines [a]		00ff00	000 255 000
Other Lines [a]		aaaaff	170 170 255
Line Points [b]		000000	000 000 000
Modify Box		00b6cc	000 182 204
Selection Color [c]		00b6cc	000 182 204

Table 15: Default View2D object colors

[a]: Colors for Outer Polylines, Inner Polylines and Other Lines are used, when Pen Color Mode is disabled.

[b]: Color for Line Points is used, when Line Points is enabled.

[c]: Selection Color is used, when both Selection Color and Pen Color Mode are enabled.

7.4.1.1.3 MultiHead

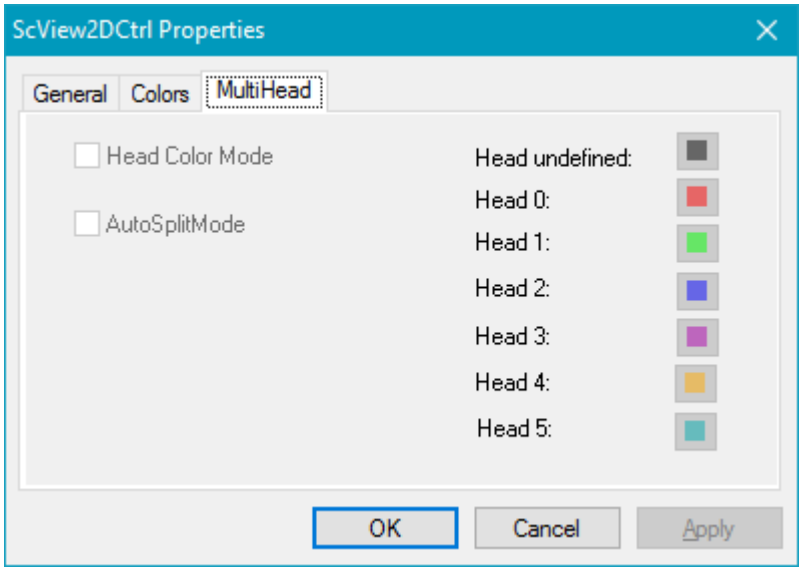


Figure 136: ScView2DCtrl_multihead

These check boxes are only available for MultiHead users, with which the colors of the different heads marking can be defined. If Head Color mode is checked the colors of the entities will be defined here if not, the colors of the pens are used.

Default MultiHead colors:

Head number	Color	RGB HEX	RGB DEC
Head undefined		000000	000 000 000
Head 0		ff0000	255 000 000
Head 1		00ff00	000 255 000
Head 2		0000ff	000 000 255
Head 3		aa00aa	170 000 170
Head 4		ffaa00	255 170 000
Head 5		00aaaa	000 170 170

Table 16: Default MultiHead colors

MultiHead colors are used, when Head Color Mode is enabled.

7.4.2 Print Preview

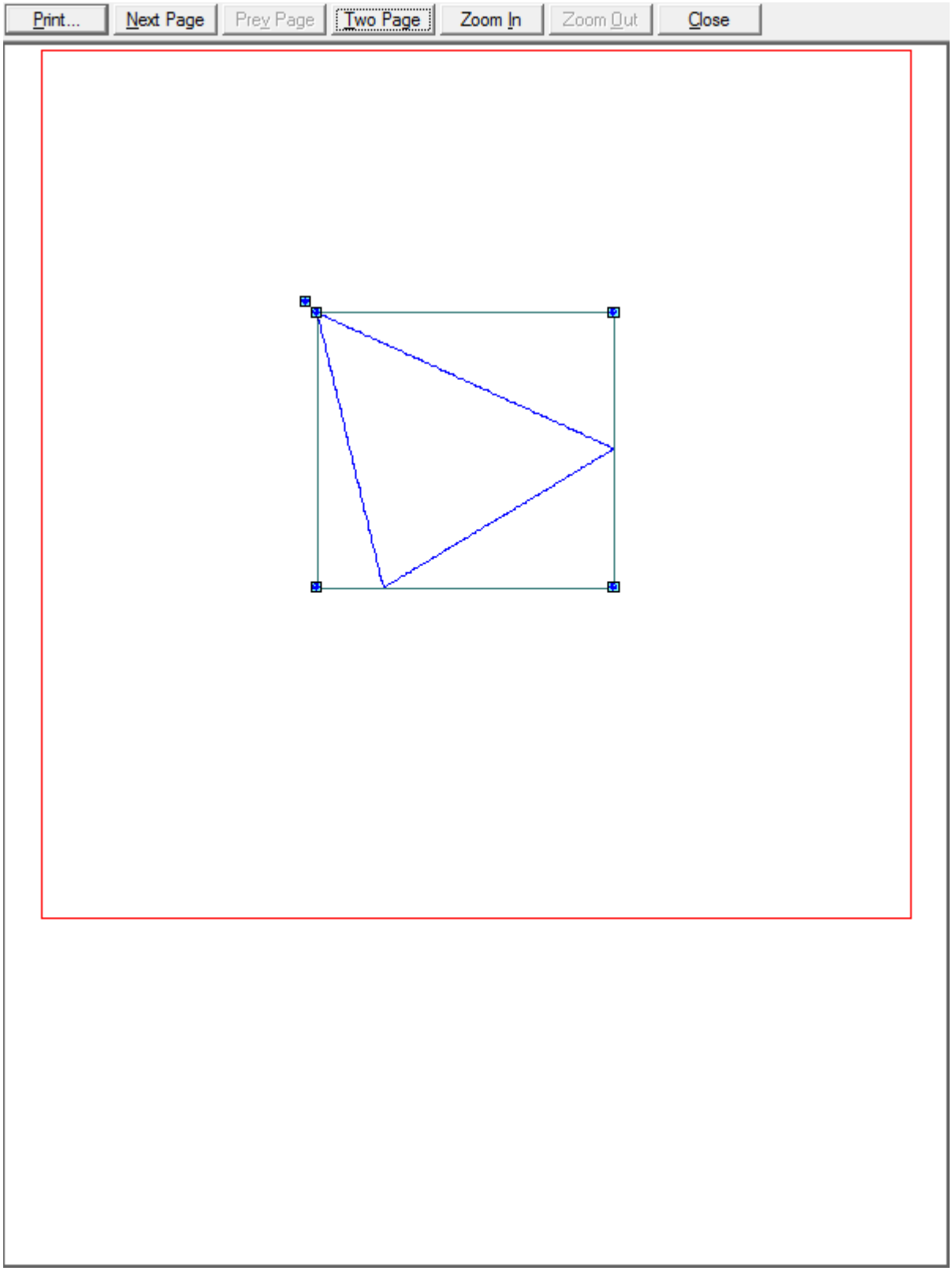


Figure 137: Print Preview Window

The print preview can be reached by the menu *File* → *Print Preview* or by the Camera Toolbar. This is a standard windows print preview, for further information see the windows help.

7.5 Entity Property Sheet

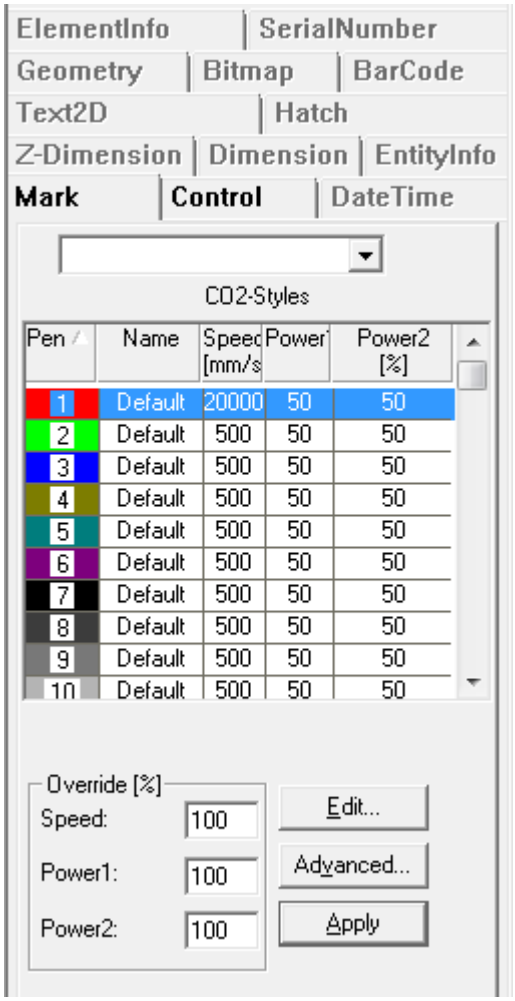


Figure 138: Entity Property Sheet

The Entity Property Sheet contains one Property Page for each possible property.

Only those Property Pages are active which correspond with a property available for the selected object.

For a detailed description of a special property page follow these links:

- BarCode
- Bitmap
- Control
- DateTime
- Dimension
- ElementInfo
- Entity Info
- Geometry
- Hatch
- Mark
- SerialNumber
- Styles
- Text2D
- Z-Dimension

The property page Z-Dimension is only available with Optic3D.

7.6 Status Bar

The mark status bar displays information about the mark process:

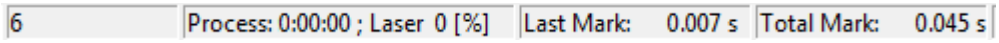


Figure 139: Mark Status Bar

From left to right you can see the number of quantities, the process time in seconds and percentage of laser on time, the duration of the last mark process and the duration of the whole mark process.

8 Entities (Objects)

This chapter describes the various kind of objects that can be created within SAMLight or that can be loaded as a job file.

8.1 Entity Hierarchy

Entities: All objects in the application are entities by definition. The three main categories of entities are elements, containers and groups and in general an entity is either an element or a container or a group of entities. The elements keep the real geometric data like lines, points and pixels, and the so called containers contain elements.

Elements:

Primitive Elements: Primitive elements are single points and single straight lines.

LineArray, PolyLine and PixelArray: *LineArray*, *PolyLine* and *PixelArray* are the next level of elements. *LineArray* and *PolyLine* represent sets of straight lines and *PixelArray* represents a set of pixels (points with gray values).

The difference between LineArray and PolyLine is as follows: The purpose of the element *LineArray* is to represent a set of lines. So the items of a *LineArray* are single straight lines described by their start and end point. In contrary to this a *PolyLine* is itself a line consisting of straight lines. So the description of a *PolyLine* is a sequence of points and the straight lines connect two subsequential points. They keep items of the corresponding kind in sequential order. For a *PolyLine* the item is a point p. Rectangles, triangles and ellipses are special closed *PolyLines*. So they are derived from *PolyLine*. A hatch is a special *LineArray* and therefore is derived from *LineArray*.

LineArrays,PolyLines and PixelArrays: *LineArrays*, *PolyLines* and *PixelArrays* are sets of special elements. A *LineArrays set* contains one or more *LineArray* elements. A *PolyLines set* contains one or more *PolyLine* elements. A *PixelArrays set* contains one or more *PixelArray* elements.

Containers: Container entities always consist of specific predefined numbers and types of subentities.

Layer: The Layer contains exactly three elements: one *PolyLines*, one *LineArrays* and one *PixelArrays*. This entity type is used for geometrical objects with a heterogeneous object description which are especially hatchable objects. For example if one creates a rectangle (see Object Toolbar) an entity of type Layer will be created in the Entity List (see Entity List) because a rectangle is a hatchable object. The Polylines of the Layer keep the PolyLine that bounds the rectangle and the *LineArrays* will keep the hatches (see Hatch).

8.2 Geometry Objects

Geometric objects and their settings are explained here: Rectangle, Ellipse, Spiral

1. Rectangle:

Click on the Rectangle icon  to create a rectangle:

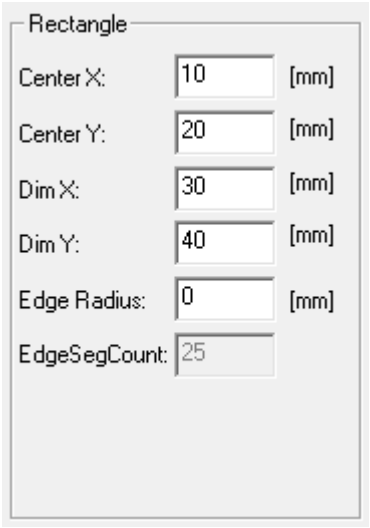


Figure 140: Rectangle properties

Rectangle:

Center, Dimension: It is possible to define the center and the dimension of the rectangle.

Edge Radius: One can generate a rectangle with round edges by defining an edge radius and the EdgeSegCount (Number of edges) to generate for each edge.

2. Ellipse:

Click on the Ellipse icon  to create an ellipse:

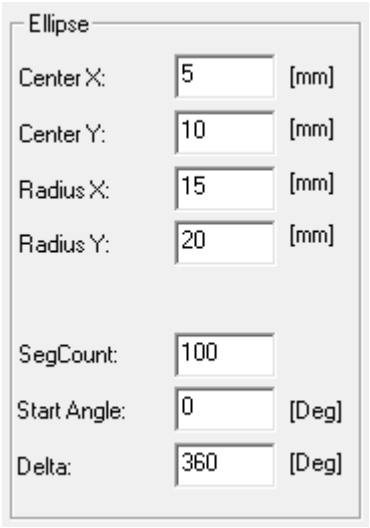


Figure 141: Ellipse properties

Ellipse:

Center X, Center Y: These two values define the center coordinates of the ellipse.

Radius X, Radius Y: With these values the radius in horizontal and vertical direction can be changed. If the values are equal the resulting object is a circle.

Segment count: The ellipse consists of a number of segments. This number is calculated to the complete 360° ellipse.

Start Angle, Delta: With these parameters it is possible to define arcs. Here the start angle defines the angle at which the elliptic or circular arc has to start relative to the x-axis and delta defines the angle of the arc relative to the start angle.

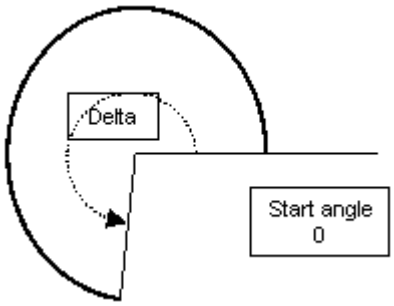


Figure 142: Example of an arc

3. Spiral:

A Standard Spiral can be created with the object toolbar button  .

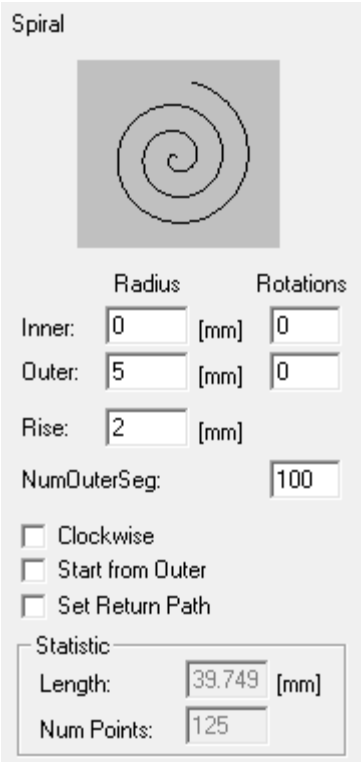


Figure 143: Spiral properties

Radius: Defines the inner and outer radius of the spiral.

Rotations: Defines the number of rotations on the inner or outer circle of the spiral.

Rise: Defines the distance between succeeding circles inside the spiral.

NumOuterSeg: Defines the number of segments of the outer circle. This proportion is taken as for the whole spiral.

Clockwise: Defines the orientation of the convolution.

Start from Outer: Defines the start point of the polyline.

Set Return Path: If checked a return path is added and the spiral ends at its start point.

Statistic:

Length: Length of the polyline.

Num Points: Number of points on the polyline.

You can change the view to Point Editing to see the order the single points will be marked.

8.3 Barcode

Generate a barcode by pressing the  Barcode2D button in the object toolbar. Move the mouse to the desired position and press the left mouse button. The barcode property page appears.

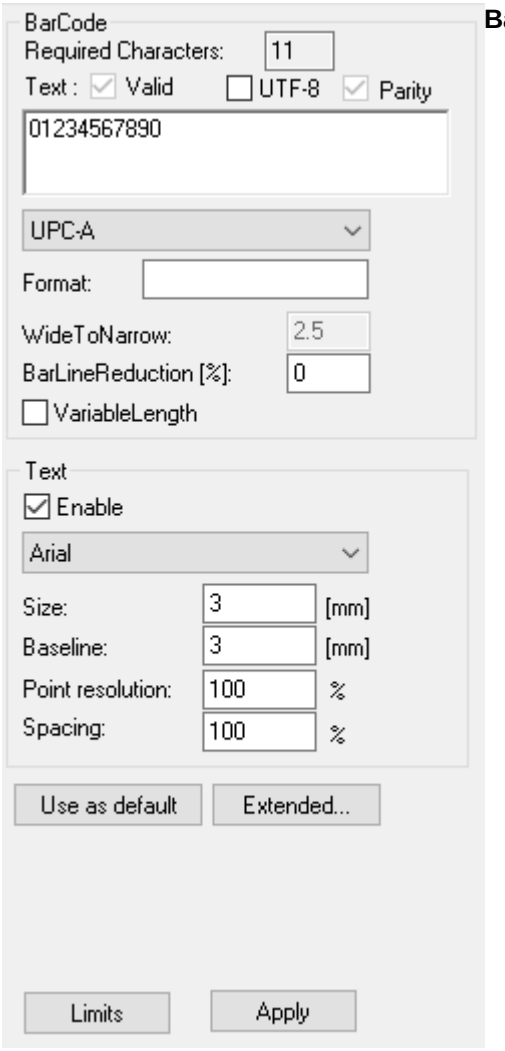


Figure 144: Barcode Dialog

BarCode:

Text: Enter any number, a date or some text into the text field.

Required Characters: Some barcodes require a specific number of characters which is displayed in this field. If 1.. is displayed no specific number of characters is required.

Valid: This is checked when the characters in the text field are valid for the selected barcode type.

Format: See subchapter Format.

Choose the desired BarCode type in the drop down menu above 'Format'.

WideToNarrow: Relation between wide and narrow barcode lines and accordingly between the wide and narrow spaces among the barcode lines. The selectable number ranges between 2 and 3.

BarLineReduction: Reduces the size of the single bars.This is not available for DataMatrixEx, Dotcode, Code39Ex, 2of5Ex and Code128(2).

VariableLength: If this is activated, the size of the barcode changes with the length of characters encoded. E.g. the length of a generated EAN-128 barcode changes almost linear with the length of the text. If this is deactivated the size of the barcode outline doesn't change with the text size.

Text:

Size: Specifies the size of the text.

Baseline: Distance between the baseline of the barcode to the baseline of the text, see example below.

Point resolution: See chapter Text Properties.

Spacing: See chapter Text Properties.

Use as default: Uses the properties of the currently selected barcode object for the generation of new barcodes. The program saves these settings also for a new program start in case save settings on exit is checked in the general settings.

Extended...: See subchapter Extended

Figure 145: Example of a barcode

Limits: Opens the 'Size Limits' dialog. Length and Height limits of the barcode can be specified here.

Supported Barcode Types:

1D: 2 of 5, 2 of 5 EX, 3 of 9, Codabar, Codablock-F, Code 128 Subtype A, Code 128 Subtype B, Code-128, Code-128(2), Code-16K, Code-39, Code-39 EX, Code-49, Code-93, EAN, EAN-128, EAN-13, EAN-13+2, EAN-13+5, EAN-14, EAN-8, EAN-8+2, EAN-8+5, Ex Code39, Ex Code93, Expanded, Expanded-Stacked, GS1-128, GS1-128 CC-A/B, GS1-128 CC-C, I-2/5, ITF-14, ITF-6, Limited, MicroPDF417, Omnidir, PDF417, PostNet-A, PostNet-C, PostNet-Cp, RSS, SSCC, Stacked, Stacked-Omnidir, Truncated, UPC-A, UPC-A+2, UPC-A+5, UPC-E

2D: Aztec, DataMatrix[1], DataMatrixEx[1], DataMatrixRect[1], Dotcode, GS1-DataMatrix[1], Micro QR Code, QR Code, QR Code EX

[1] These Barcodes comply with ECC200 and ISO/IEC 16022:2000.

8.3.1 Barcode Format

It is possible to differentiate between text encoded by the barcode and text shown below the barcode. Therefore a control code is given. To enable following control code, enter *%H* into the format field of the barcode property page. Note: It is not supported for all types of barcodes.

Control Code	Meaning
%b	Start to be only in barcode
%h	Start to be in human readable text only
%e	End of control code
%%	Insert a % sign

Figure 146: Available barcode control codes

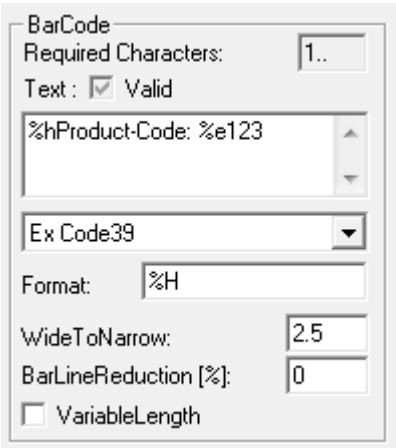


Figure 148: Example output

Figure 147: Example of a control code

For 2D barcodes it is also possible to enter 4 values separated by a comma. Then the text of the barcode is wrapped around it.

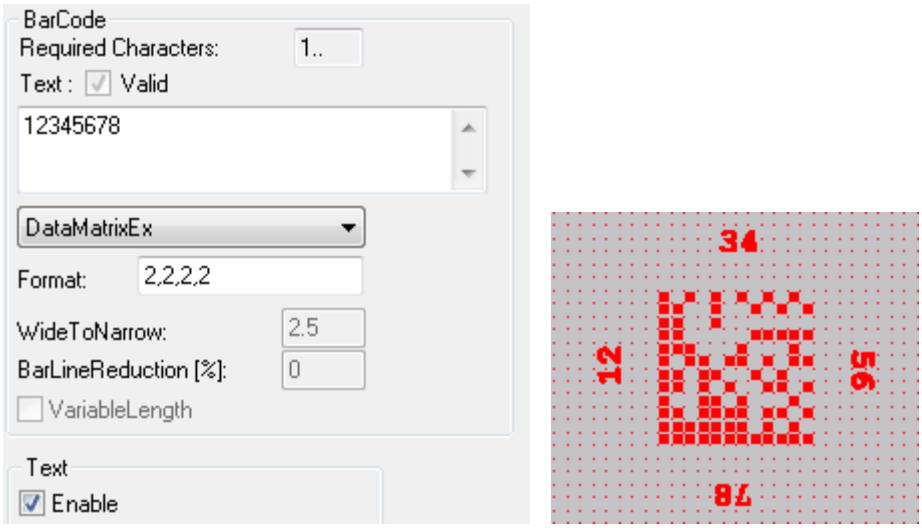


Figure 149: Example of a special formatted barcode

8.3.1.1 Code-39 Ex

For Code-39 Ex, the following Extended dialog is available:

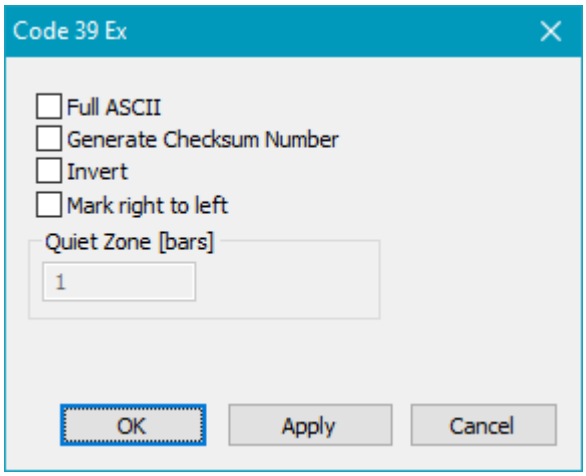


Figure 150: Barcode Extended for Code-39 Ex

BarCode Extended dialog for Code-39 Ex:

Full ASCII: When using ASCII encoding, the range of characters can be extended. But make sure your barcode reader is compatible with ASCII.

Generate Checksum Number: Adds a checksum which is compatible with ASCII encoding.

Invert: When enabled, inverted barcodes can be used in standalone mode. To improve readability, the quiet zone could be increased.

Mark right to left: Changes the direction of marking.

Quiet Zone [bars]: Is only relevant for inverted barcodes. Adds the specified number of bars on the right and left hand side.

8.3.1.2 GS1 Barcodes

If selecting a GS1 barcode, for example GS1-DataMatrix, the application identifiers defined by GS1 have to be in brackets followed by its value. Application Identifiers are for example:

Identifier	Description

(01)	GTIN (Global Traded Item Number)
(10)	Batch or Lot Number
(11)	Product Date (as YYMMDD)
(15)	Best Before Date (as YYMMDD)
(21)	Serial Number
(400)	Purchase Order Number
(422)	Country of Origin (as ISO code)

Table 17: GS1-DataMatrix Application Identifiers

8.3.1.3 QR Code

The QR-Code is a special 2D barcode format. In addition to numerics, it can also contain the letters from a to z and special characters like +, -, /, % ... . To enable this option, click on *Extended...* in the BarCode property page. Then in the window that pops up select *Byte* in the field *Barcode Mode*.

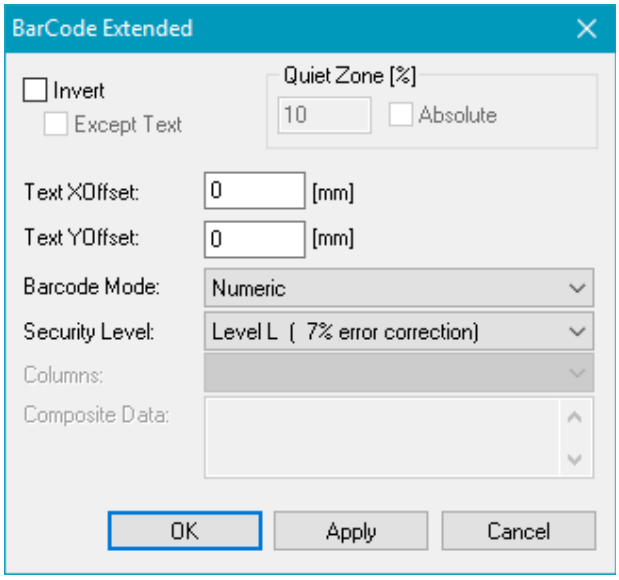


Figure 151: Barcode Extended for QR Code

BarCode Extended for QR Code:

Barcode Mode: Choose the desired mode in the drop down menu. Options are 'Numeric', 'Alpha', 'Byte', and 'Kanji'.

Note that, depending on the selected mode, different types of characters are allowed. You will get direct feedback, via the 'Valid' checkbox (see Barcode).

In addition, the selected mode also changes the number of valid characters!

For further information on Figure 151 see Barcode Extended.

8.3.1.4 QR Code EX

A special barcode type QR Code EX is available. This barcode is like a QR Code barcode with the additional feature that an Entity can be embedded in the center of the Barcode. Therefore select QR Code EX in the barcode drop down box and click on *Extended.* The following dialog will be shown:

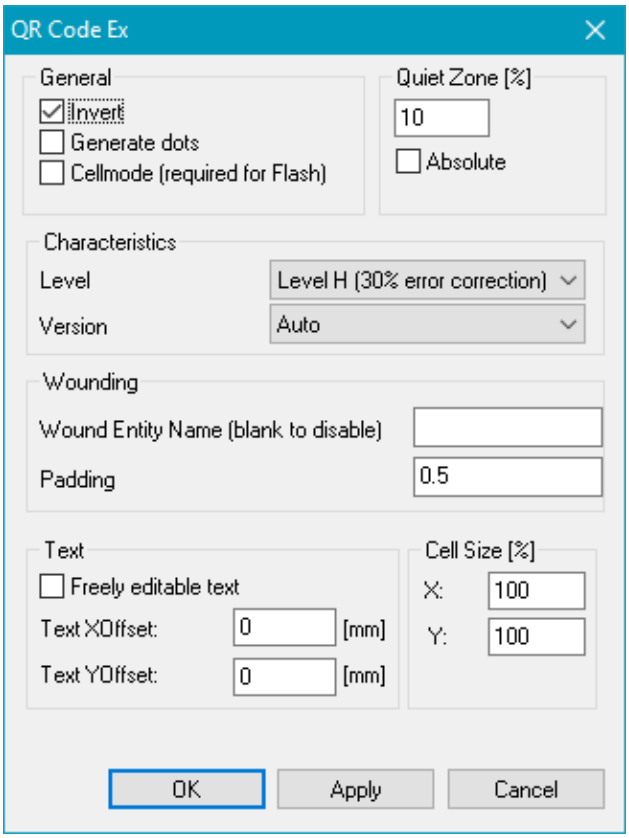


Figure 152: QR Code EX extended dialog

Figure 153: QR Code with "USC" wounding

General:

Invert: Inverts the barcode.

Generate dots: The barcode will consist of dots.

Cellmode (required for Flash): The barcode will consist of cells. Cellmode is required for Flash applications.

Quiet Zone [mm]: Apply a border to the barcode.

Wounding: Embed an Entity inside the QR Code.

Wound Entity Name: Name of the entity that has to be embedded inside of the QR code. If the edit field is blank no entity will be embedded.

Padding: Define a border around the embedded entity.

QR Code mode is automatically adjusted to the type of characters in the text field (see Barcode).
(e.g. '12345' = numeric mode, '12345A' = alphanumeric mode, '12345Aa' = binary mode)

With this change of mode, the number of allowed characters often changes as well.

8.3.1.5 Code 128

In SAMLight, the Code 128 barcode is also available.

This barcode has three subtypes: A,B and C.
Subtypes A and B can contain letters while subtype C only holds numerical values. Subtype A and B can be selected separately. Subtype C will be used automatically if you use Code 128 and the data only contains numerical values.

8.3.2 Scaling

When scaling barcodes that have the text feature enabled, the text is normally not scaled with the barcode. Only for those barcodes that can have the checkbox "Extended → Text freely editable" enabled, like DataMatrixEx or QRCodeEX can scale the text with the barcode when this checkbox is activated. Mirroring can lead to unwanted results for those barcodes that have no "Extended → Text freely editable" checkbox.

8.3.3 Barcode Extended

Press the Extended... button in the barcode property page to get dialog in Fig. 154 . If the selected barcode is a DataMatrixEx, a Code 128 (2), a 2 of 5 EX or a Code-39 Ex, then see below.

For QR Code and QR Code EX, there are also special dialogs.

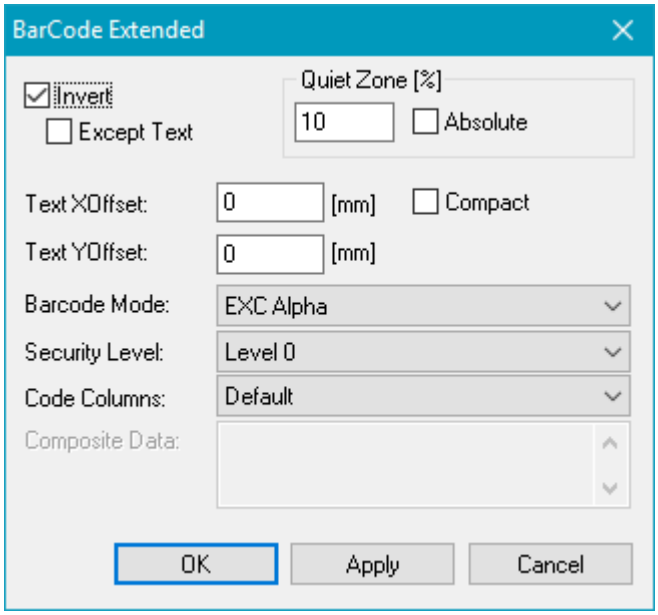


Figure 154: Barcode Extended Dialog

Invert: Inverts the dark and bright parts of the barcode. The barcode must be bordered to prevent the outer barcode lines from disappearing. Therefore a QuietZone is defined, see example below. The width of the zone can be given absolute units or in percentage of the width. The selectable percentage ranges from 1 to 50.

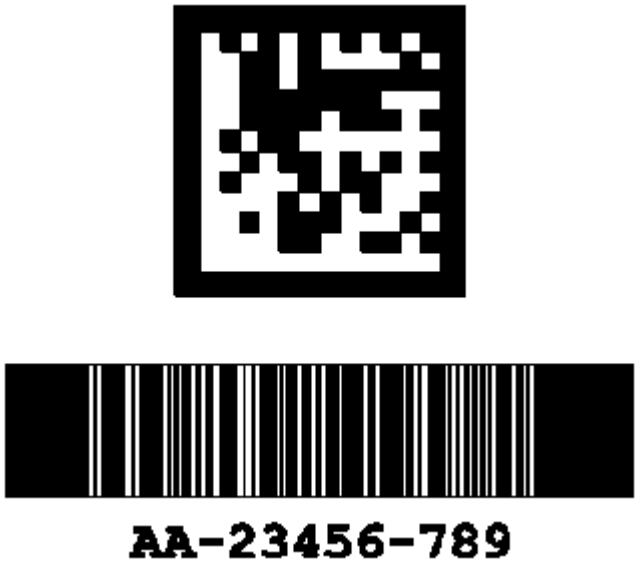


Figure 155: Inverse Barcode Example

Except Text: This suboption of Invert excludes inverting the text when Invert is selected and is implemented at the moment just for the EAN-13-Barcodes.

Text XOffset: This value can be used to move the position of the text in horizontal direction that is activated for the barcode. For positive values the offset point of the barcode text is moved to the left, for negative values it is moved to the right.

Text YOffset: This value can be used to move the position of the text in vertical direction that is activated for the barcode. For positive values the offset point of the barcode text is moved down, for negative values it is moved up.

Barcode Mode / Security Level / Code Columns: These parameters are optional and depend on the barcode type that was chosen. So the number of options, the meaning of the possibly available options and the parameters that can be selected here are defined by the related barcode specification.

The following Barcodes have a Mode option: Aztec, Code 16k, Maxicode, MicroPDF417, PDF417, QR Code, RSS

The following Barcodes have a Security Level: Aztec, MicroPDF, PDF417, QR Code

The following Barcodes have Columns: Composite, MicroPDF417, PDF417

Security Level:

For the QR-Code the following Security Levels for error correction are available:

- Level 0: 7%
- Level 1: 15%
- Level 2: 25%
- Level 3: 30%

A Security Level of 30% means, that the error correction can read the Barcode if up to 30% of the Barcode is disrupted. For different Barcodes there exist a different number of Security Levels with different error correction capabilities.

If a **DataMatrixEx** object is selected, the following dialog opens after pressing the Extended... button in the barcode property page.

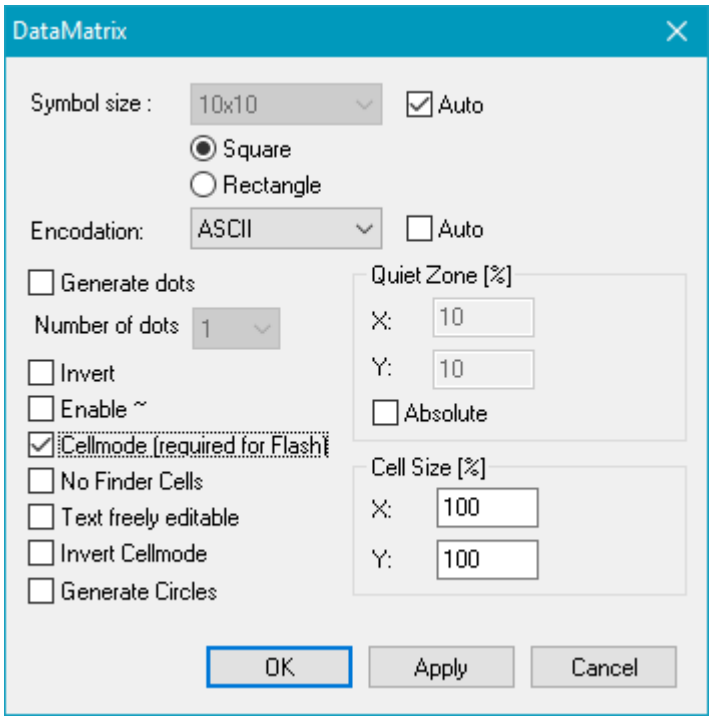


Figure 156: Data Matrix Extended Dialog

Symbol size: Number of cell rows and columns of the DataMatrix.

Auto: Chooses the smallest possible size inside the combobox for the selected barcode object.

Encodation: Choose the type of the encodation. Available encodations are:

- ASCII
- Base256
- C40
- Text
- ANSI X12
- EDIFACT
- GS1 - Application Identifiers must be written with round brackets, like (01). The **FNC1** characters must not be written, they are added automatically.

Auto: Chooses the encodation for which the selected barcode text is being optimally compressed.

Example GS1 DataMatrixEx:

- Barcode type: DataMatrixEx
- Encodation: GS1
- Text: (01)03453120000011(17)191125(10)ABCD1234
- Encoded string: **FNC1**010345312000001117191125 10ABCD1234

Figure 157: GS1 DataMatrixEx example

Generate dots: If Generate dots is checked the barcode consists of single points. (For other 2D barcodes, please refer to Generate Dots.)

Invert: If Generate dots is not selected then the barcode can be inverted. If inverted the barcode must be bordered to prevent the outer barcode structures to disappear. Therefore a *Quiet Zone* is defined. The width of the zone can be given in absolute units or in the percentage of the width. The selectable percentage ranges between 1 and 50.

Enable~: Allows to encode 3-digit decimal values. The format is *~d* followed by 3 digits. For example: *~d038*

Cellmode (required for Flash): If Cellmode is checked, the barcode consists of a closed polygon for each cell. Cellmode is required for Flash applications. See below for examples.

No Finder Cells: If the barcode is generated by cells then the finder zone is drawn as one closed polygon.

Text freely editable: If checked the user may define an arbitrary text independent of the content of the DataMatrix.

Invert Cellmode: If this checkbox is activated then the DataMatrix will be inverted. So it is possible to define a Quiet Zone. The width of the Quiet Zone can be defined in unit Cells. This works within SAMLight and with the USC-2 Flash but not with the FEB-1 board.

Generate Circles: If this checkbox is activated, each cell in cellmode is a circle, not a square.Generate Circles is only available if Cellmode is activated.

Cell Size: Active if Generate Cells is checked. The size of a single cell can be defined. If it is 100% the cells contact each other.

Quiet Zone: Generates a border of defined thickness around the barcode.

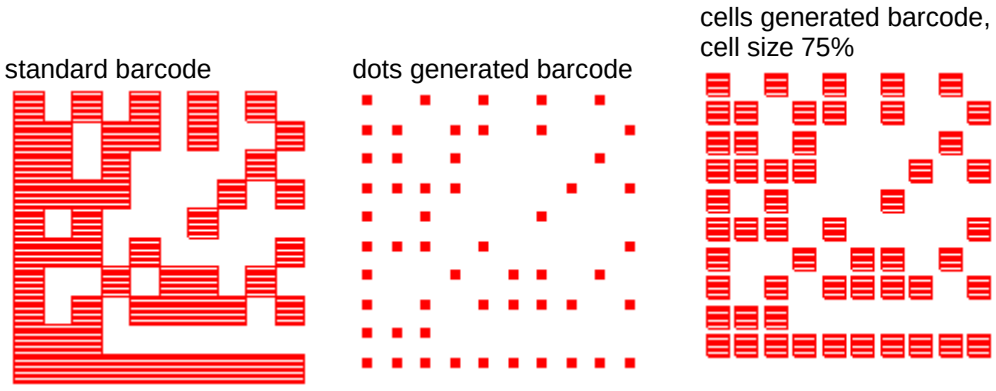


Figure 158: DataMatrixEx Examples

If a **Code 128 (2)** object is selected, the following dialog opens after pressing the Extended... button in the barcode property page.

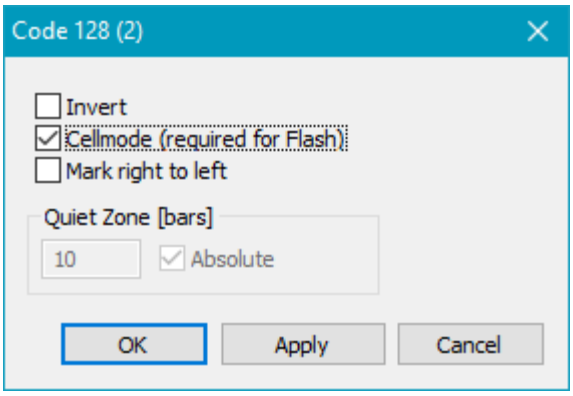


Figure 159: Code 128 (2) Extended Dialog

Invert: The barcode will appear inverted.

Cellmode: The barcdoe will consist of cells.

Quiet Zone: Generates a border of defined thickness around the barcode.

If a **2 of 5 EX** object is selected, the following dialog opens after pressing the Extended... button in the barcode property page.

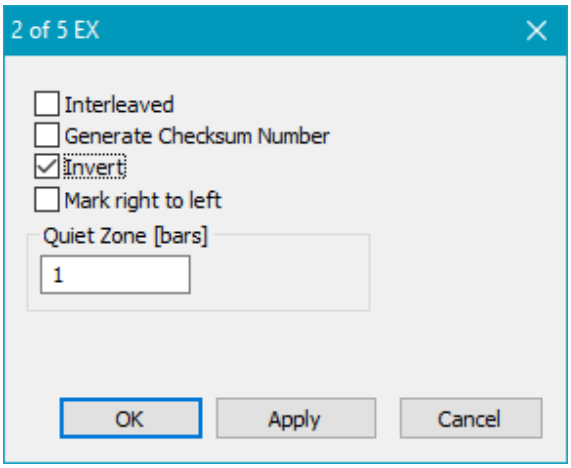


Figure 160: 2 of 5 EX Extended Dialog

Interleaved: In order to use this feature the barcode has to have an even number of ciphers. Then the barcode will be made smaller while increasing the information density.

Invert: The barcode will appear inverted.

Quiet Zone: Generates a border of defined thickness at the left and right end of the barcode.

If a **Code-39 Ex** object is selected, the following dialog opens after pressing the Extended... button in the barcode property page.

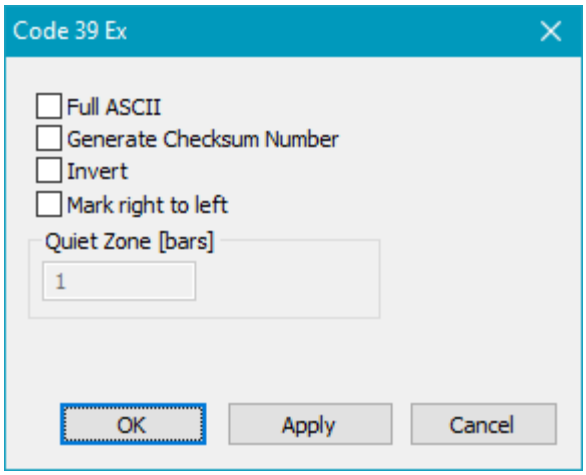


Figure 161: Barcode Extended for Code-39 Ex

BarCode Extended dialog for Code-39 Ex:

Full ASCII: When using ASCII encoding, the range of characters can be extended. But make sure your barcode reader is compatible with ASCII.

Generate Checksum Number: Adds a checksum which is compatible with ASCII encoding.

Invert: When enabled, inverted barcodes can be used in standalone mode. To improve readability, the quiet zone could be increased.

Mark right to left: Changes the direction of marking.

Quiet Zone [bars]: Is only relevant for inverted barcodes. Adds the specified number of bars on the right and left hand side.

8.3.4 Barcode Reader

It is possible to connect a barcode reader to the PC and to read barcodes in SAMLight. Therefore create a serial number in the Job. Then in the Serial Number Property Sheet click on the *Advanced* button. In the window that is popping up activate the checkbox *Barcode reader mode* in the field *Popup edit box*. Now if you do a mark a window is popping up after the mark has finished, see the picture below:

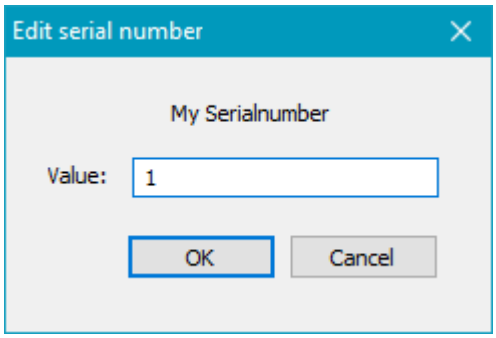


Figure 162: Edit serial number dialog

Here you can enter an arbitrary number or, if a barcode reader device is connected to the PC, you can read any barcode with this device and the value of the barcode will be assigned to the serial number in the job.

8.3.5 Size Limits

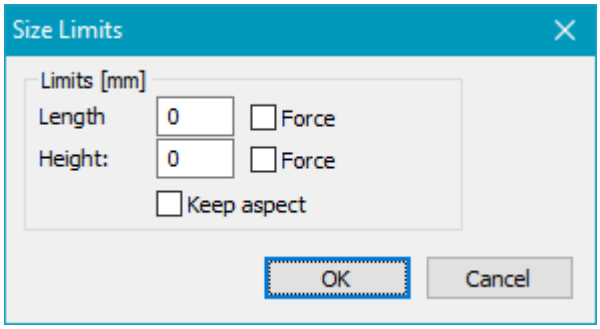


Figure 163: Size Limits Dialog

8.4 Bitmap

After importing a Windows Bitmap it will be converted and displayed as a 8bit grey image. To bring it to the scanner output it is necessary to calculate a Scanner Bitmap. The generation can be done in the property page of the bitmap. After selecting the bitmap the bitmap property page becomes active:

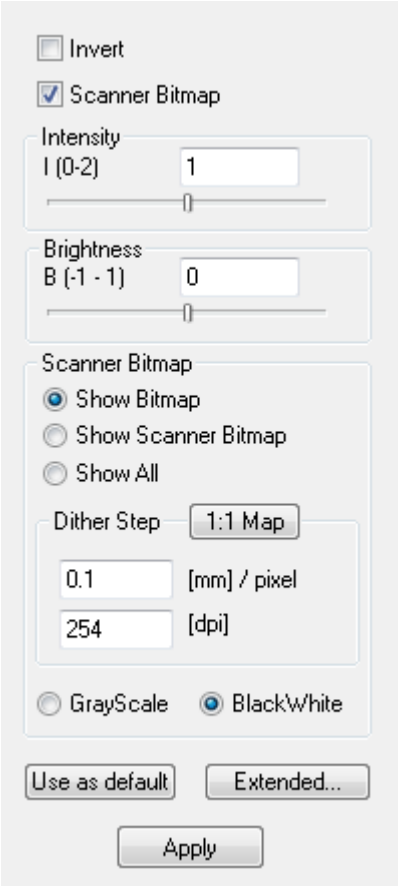


Figure 164: bitmap property page

Invert, Intensity, Brightness: Work on the original bitmap

Scanner Bitmap: Select this check button to generate the Scanner Bitmap. Press Apply. If this is not selected, no scanner bitmap will be created or if it already exists it will be deleted after Apply is pressed.

Scanner Bitmap:

Show Bitmap: Shows the original bitmap.

Show Scanner Bitmap: Shows the scanner bitmap.

Dither Step: Defines the resolution for each pixel inside the scanner bitmap. If for example Dither Step = 0.1, the distance between two neighbouring pixels is 0.1, means one mm on the scanner bitmap contains 10 pixels in x-direction * 10 pixels in y-direction. The resolution should be in the range of the laser focus. You can also define the Dither Step via dpi (dots per inch).

GrayScale: If enabled the grayscale values will be kept when transforming the bitmap to the scanner bitmap.

BlackWhite: If enabled the grayscale values will be approximated by specific placements of black and white pixels to give the impression of gray values. This is a similar method as with a Black/White LaserJet. There are two different algorithms available in Bitmap Extended for this mode: Random Noise and Floyd Steinberg.

The grayscale mode is only available for scanner cards with an appropriate mode. To scan bitmaps generated in grayscale mode the hardware mode has to be enabled.

For the dithered mode *LaserOn* delay is set to *1* µs and *LaserOff* delay to 10 µs by software. *Jump Speed* is set to *Mark Speed*, except for the jump between two lines in one direction mode the *Jump Speed* and the *Jump Delay* defined within pen settings is used. In all other cases the scanner delays are set to *0*. In addition the dithered mode uses the skywriting parameters of the pen if enabled.

Extended...: For how to set up the parameters for marking gray scaled bitmaps in bi-directional mode, see sub-chapter Marking Bidirectional. For the other *Extended...* features see Bitmap Extended.

Apply: Starts the generation of the Scanner Image if the check button Scanner Bitmap is checked. Else deletes an existing Scanner Image.

Example:

Figure 165: original picture (left) and scanner image in error diffuse mode (right)

Hardware Mode: For drawing bitmaps in grayscaled mode the hardware mode has to be set which means that scanner movement and laser burning are done at the same time. To enable the hardware mode it is necessary to select the pulse width modulation mode PixelPWM or the amplitude modulation mode PixelAM inside the optic settings advanced dialog. Also the selection of both is possible. In addition the hardware flag inside the scanner settings for pens has to be checked.

If hardware mode is checked, it is recommended to set the LaserOn delay within this page to 1 µs and the LaserOff delay to 2 µs.

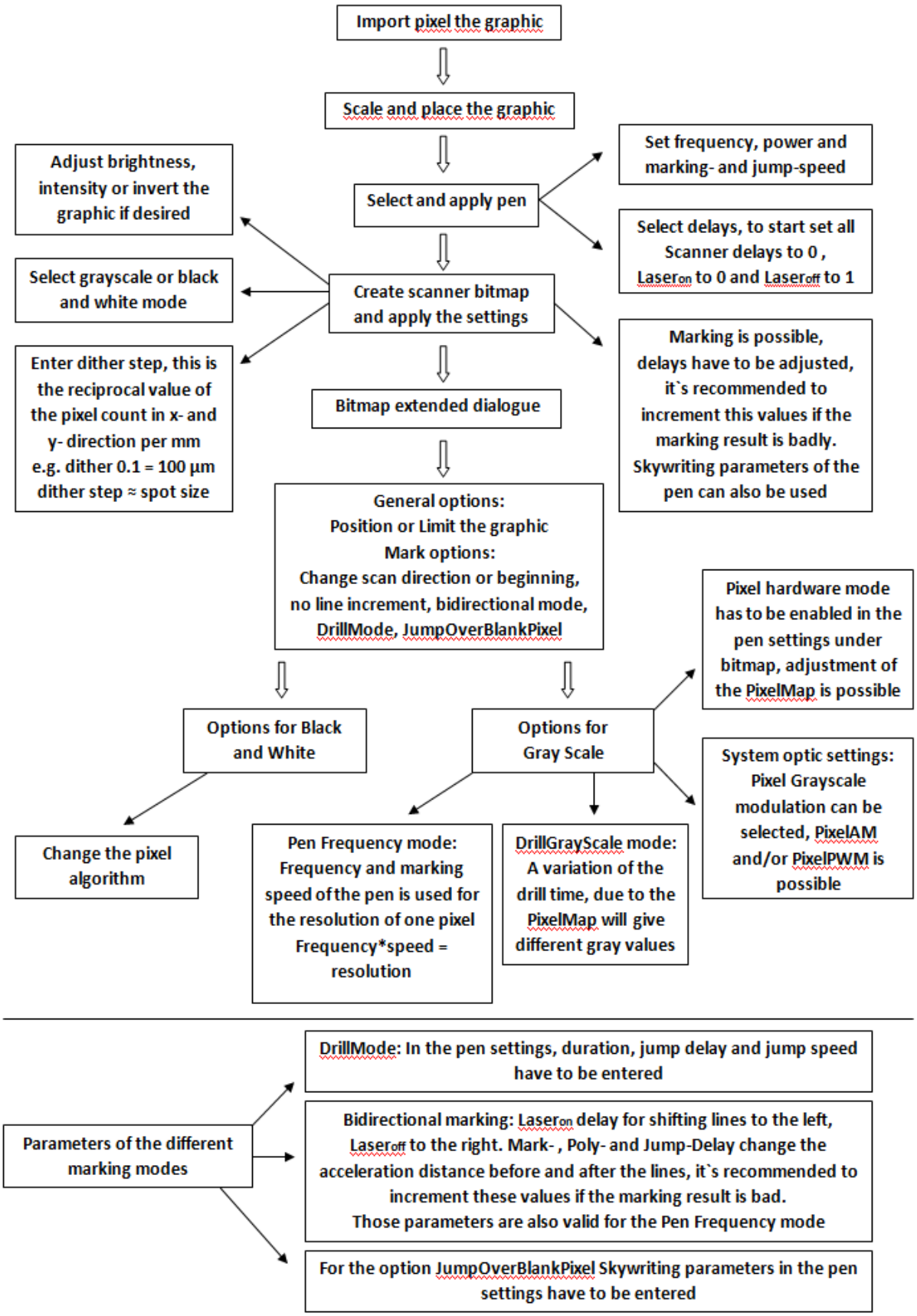


Figure 166: HowTo flowchart for Bitmap Marking

For details about pixelmode see chapter Backgrounds.

8.4.1 Bitmap Extended

Press the Extended... button in the bitmap property page to get the dialog shown below.

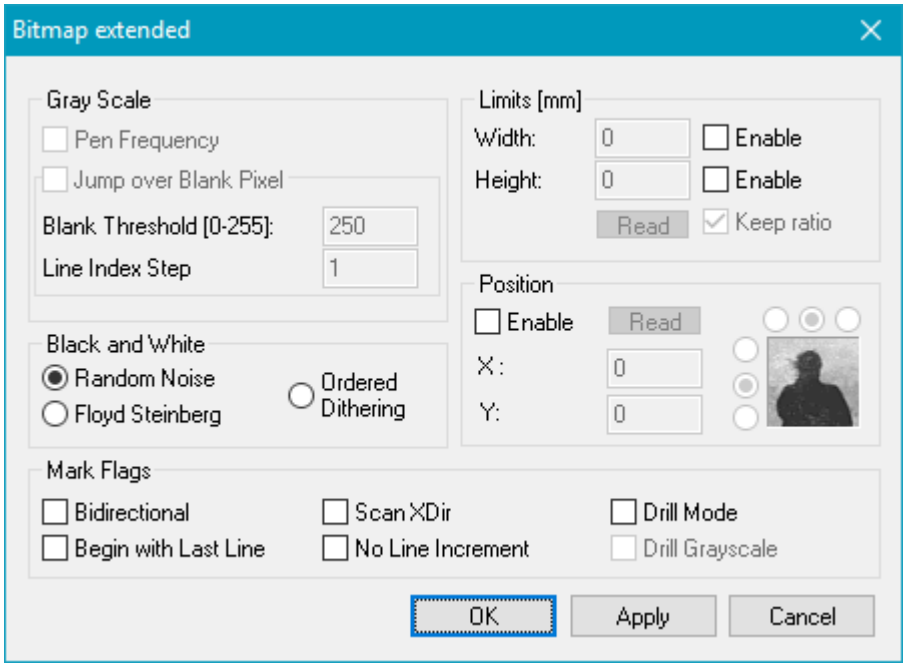


Figure 167: Bitmap Marking Extended Dialog

The Following settings are valid for Reimport which is useable for a Client Interface application.

Gray Scale:

Pen Frequency: Check this box to take the pen frequency for marking bitmaps. This will adapt the resolution of one pixel line to the speed and the frequency of pen.

Jump over Blank Pixel: To optimize marking time blank pixels will be skipped. SAMLight will skip all first and last pixels of a bitmap line with a gray value higher or equal than the threshold value 'Blank Threshold'. Blank pixels in between nonblank pixels will be jumped with MarkSpeed, not with JumpSpeed.

Blank Threshold: This value has a range from 255 (white) to 0 (black). The default value is 250.

Line Index Step: This feature can be used to reduce local heating of the target material at marking gray scale bitmaps. If 'Line Index Step' = x > 1 the bitmap will not be marked line by line in one run but in x runs where in each run every xth line will be marked. For example a Line Index Step of '3' results in this marking order: 1, 4, 7, 10, .., 2, 5, 8, 11, .., 3, 6, 9, 12, ..

Black and White:

Random Noise: Creates a rougher scanner bitmap than Floyd Steinberg, but doesn't tend to produce moiré patterns. As the name suggests, if used several times Random Noise generates different scanner bitmap patterns for the same bitmap.

Floyd Steinberg: Creates a smoother scanner bitmap than Random Noise, but tends to produce moiré patterns. Always generates identical scanner bitmaps for the same bitmap if applied several times.

Limits: Define Width and/or Height as a placeholder property for reimport. With keep ratio the aspect ratio is kept.

Position: Define a position for the bitmap placeholder. The coordinates of the reference point are defined with the X and Y edit fields. The radiobuttons define the point of attack of the bitmap.

Mark Flags:

Bidirectional: If checked the scanner does not jump to the beginning but to the end of the next line if it reaches the end of a line. For more details please refer to Marking Bidirectional.

Begin with Last Line: If checked the scanner begins drawing from the last line instead of the first line of the bitmap.

ScanXDir: The default scanning direction is y, so the scanner moves from bottom to top while scanning. To choose x as scanning direction activate this checkbox.

No Line Increment: If checked the scanner draws all bitmap lines into one line. This is necessary if for example the workpiece itself is rotated during marking.

DrillMode: The pixels of a bitmap will be handled as single points. If the drill mode in the Pen is active, the points will be marked with the drill mode property page for pens. Each pixel will be handled if its gray value > 0. So as default it is a black / white mode.

DrillGrayscale: Only available in combination with the DrillMode. The drill time on each pixel depends on its gray value. If the pixel is white, the drill time that is adjusted in the pen is executed. If it is black, the drill time is zero. In between the drill time of a pixel grows linearly with its gray value.

8.4.2 Marking Bidirectional

Usually the bitmaps are marked in one-directional mode. In the following it is described how to set up the parameters for marking in bi-directional mode. Press the *Extended...* button in the bitmap property page to show the dialog below and check the bidirectional box.

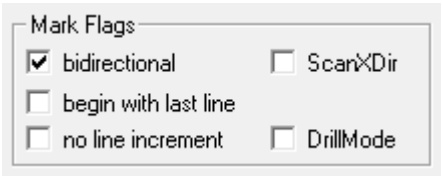


Figure 168: bitmap property page

This picture shows the way the laser should go to mark the bitmap properly.

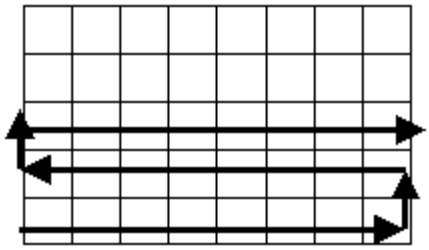


Figure 169: not corrected marking path

But this way of marking will cause picture mistakes. One reason is that a scanner needs a short startup time to reach constant speed and a constant signal frequency. The other reason is the delay of the scanner. To solve these problems an acceleration length was introduced:

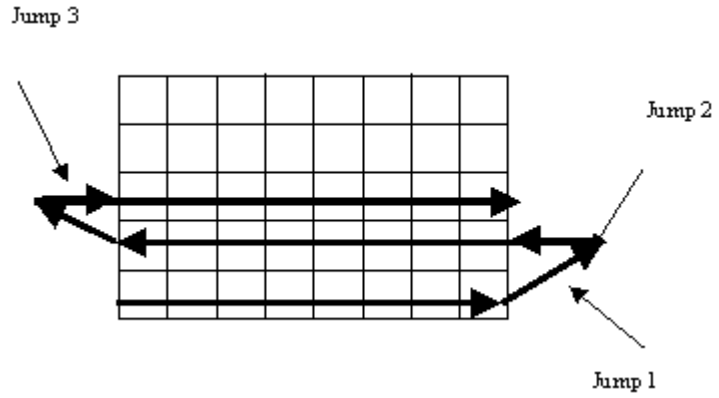


Figure 170: corrected marking path

The used jump speed is set equal to the speed of the pixelmarking. To enable the PixelShifting delays of the scanner the settings that are defined within scanner settings page for pens are used. *LaserOn* and *LaserOff* delays will automatically be set to *1*. The parameters stored in *LaserOn* and *LaserOff* will be used for calculating the shift of every second line: *LaserOn* for shifting left, *LaserOff* for shifting right.

8.4.3 Black and White

This is how Black and White (B&W) bitmap marking works:

Within a line:

- Automatically used parameters:
  - LaserOn Delay = 1 µs
  - LaserOff Delay = 10 µs
  - Mark Delay = 0 µs
  - Poly Delay = 0 µs
  - Jump Delay = 0 µs
  - Jump Speed [mm/s] = Mark Speed [mm/s]
- Gate Frequency [kHz] = Mark Speed [mm/s] / Dither Step [mm] * 10-3
- Laser pulses / Pixel = Laser Frequency [kHz] / Gate Frequency [kHz]
- Skywriting:
  - StartDistance [mm] = StartLength [µs] * Mark Speed [mm/s] * 10-6
  - EndDistance [mm] = EndLength [µs] * Mark Speed [mm/s] * 10-6

Between lines:

- Jump Speed [mm/s] = Jump Speed [mm/s]
- Jump Delay [µs] = Jump Delay [µs]
- Bidirectional mode:
  - Shift of odd lines [mm] = (LaserOff - LaserOn Delay [µs]) * Mark Speed [mm/s] * 10-6
  - Jump Speed [mm/s] = Mark Speed [mm/s]

8.4.4 Grayscale

At the marking process the greyscale value of the scanner bitmap will correspond to the laser power. That is why the greyscale adjusting can be used for fine-tuning your mark result and consider non linear behavior of the laser and the material. The brightness and the intensity can be changed for the original bitmap in the bitmap property page. After that the greyscales of the scanner bitmap can be adjusted via two different pixel maps:

Scanner Bitmap (entity property) ⇩ **Pen PixelMap** (pen property) ⇩ **System PixelMap** (global setting) ⇩ **Laser** (PWM or AM)	The original scanner bitmap greyscales will be adjusted by the pen pixel map first and afterwards by the system pixel map again like you can see in the following flow chart: • Pen PixelMap (useful to consider material behavior) Before marking every greyscale bitmap which is assigned to this pen will be changed by this pixel map. • System PixelMap (useful to consider general laser settings like minimum power) Before marking every greyscale bitmap will be changed by this pixel map. • Power Map is not used for gray scale marking at all.

Table 18: Order of greyscale adjustment

8.5 Serial Number

Generate a Serial Number object by clicking the 

button in the toolbar. Then move the mouse to the desired position and press the left mouse button. The serial number property page appears:

Serial Number:

Actual Value: Shows the actual value. Can not be edited. It is calculated by:

ActualValue = StartValue + IncrementValue * `RoundDown`((SequenceNumber `Modulo` ResetCount) / BeatCount)

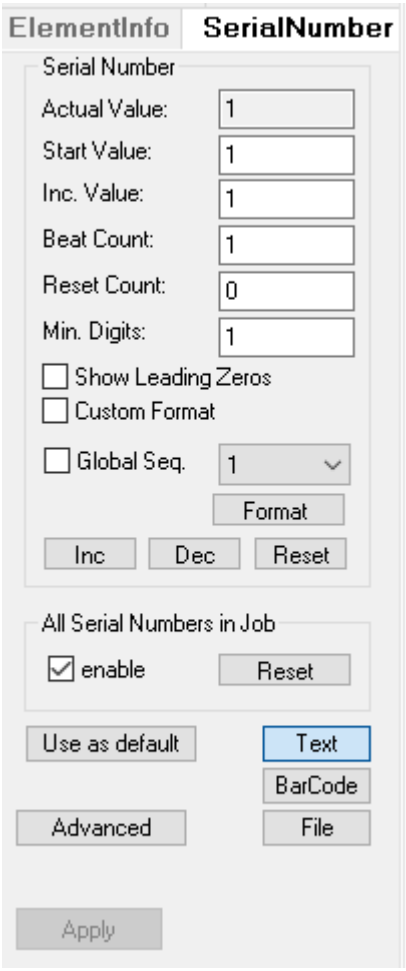


Figure 171: SerialNumber page

Start Value: Value to start with.

Inc. Value: Increment step after each beat.

Beat Count: After beat count exposures the serial number will be incremented.

Reset Count: After reset count exposures the serial number will be reset. That means it is set to the start value.

Min. Digits: Minimum number of displayed digits.

Show Leading Zeros: If activated leading zeros are displayed.

Custom Format: If activated an encoded format for serial number can be defined when pressing the Format button.

Global Seq.: Global Sequences are serial numbers which goes across all jobs. So it is possible to use the same serial number in different jobs. Setting of a global reset time is available in Settings → System → General.

Format: If the button Text is selected this switches to the Text property sheet. The format of the text can be defined here. If the button BarCode is selected this switches to the BarCode property sheet.

Inc: Manually increments the selected serial number.

Dec: Manually decrements the selected serial number.

Reset: Sets the selected serial number to the start value.

All Serial Numbers in Job:

enable: Enables all serial numbers in the job as serial numbers. If disabled no increment takes place while marking.

Reset: Resets all serial numbers in the job to the start value.

Use as default: Uses the properties of the currently selected serial number object for the generation of new serial numbers. The program saves these settings also for a new program start in case save settings on exit is checked in the general settings.

Text: The serial number will be displayed as text.

BarCode: The serial number will be displayed as barcode. Automatisation in flash mode only works for a barcode created as a serial number via this option.

File: By pressing this button, you can use a text or excel file to readout namings for serialization. See also: Automate Serialization.

Advanced: Opens the Serialnumber and Date Time dialog.

8.5.1 Serial Number Formats

For the serial numbers the format description is similar to that used in the C-language:

%[flags] [width] [.precision] [optional parameter] type

flags: 0 shows leading zeros

width: Defines the total width of the number including the decimal point. This has only an effect if leading zeros is defined and the width is defined bigger than the width of serial number plus decimal point plus precision, so that leading zeros appear.

precision: Digits after the decimal point.

optional parameter: L or l. L will restrict the serial number to show only 'width' digits and will show leading zeros. l will do the same but will not show leading zeros.

type: f double values

Format examples:

Example	Format	Description
10.000	%6.3f	3 digits after the decimal point
10	%6.0f	0 digits after the decimal point
000010	%06.0f	show leading zeros
012	%03.0Lf	restrict to 3 digits and show leading zeros
12	%03.0lf	restrict to 3 digits and do not show leading zeros

Figure 172: Formating examples

Format code and text can be entered simultaneously.

The encoding will only work if Custom Format is selected inside the SerialNumber property page.

8.5.2 Serial Number as Barcode

If the Serial Number is displayed as a Barcode it is possible to reference other text elements of the same job and include their contents into the current serial number. This feature is useful especially when the serial number barcode has to contain encoded information that are available as human-readable text within the same job too. To include the data of another text entity this object must first have an entity name. As a second step that name needs to be referenced in format "<$entity_name>" within the custom format field of the serial number barcode. Here "<$" and ">" are delimiters for that part of the serial number format that has to be replaced dynamically. If the name between these delimiters is not defined within the job no replacement is done and the full placeholder is displayed.

As an example: There is a text object named "TText" within the job that contains the text "My Information". Within the serial number barcode the following custom Format is defined: "%1.0f / <$TText>". Resulting from that the serial number barcode object would display the current serial count plus the text " / My Information":

- "1 / MyInformation"
- "2 / MyInformation"
- "3 / MyInformation"
- "4 / MyInformation"
- "5 / MyInformation"
- ...

The Main Window will show the Entity List, the View2D and the Entity Property Sheet like in the following pictures:

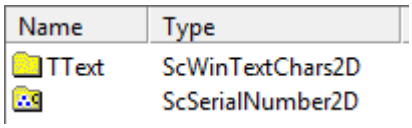


Figure 173: Entity List with Text object and Serial Number Barcode

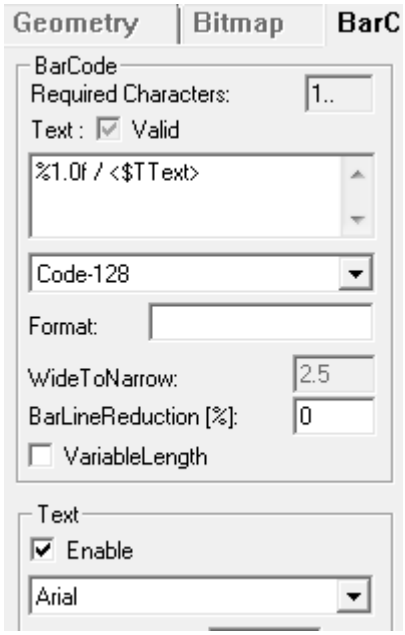


Figure 174: Property Sheet with Serial Number Barcode

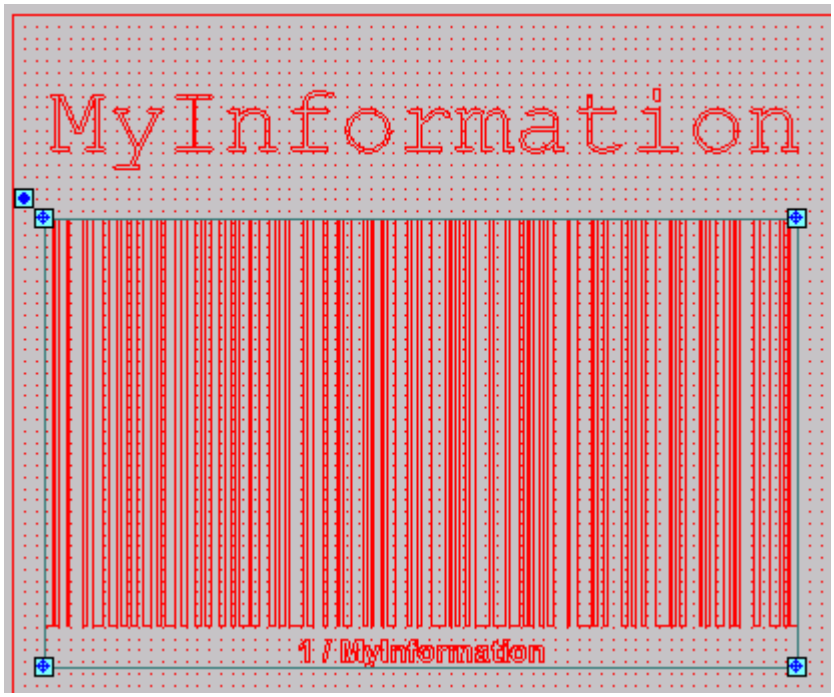


Figure 175: View2D with Text to be included and Serial Number Barcode

Serial numbers are updated once per marking process. If such a serial number barcode references to text entities of the job using the <$entity_name> syntax it will contain the information of the element named "entity_name" that is visible at that specific moment when the update is performed. If the entity with that name is changed after the serial number was updated, the contents of the serial number barcode are not updated automatically. This means the that the serial number barcodes will stay with the old value until the next marking cycle was finished or until the sequence was updated from the menu. The same is true if such a serial number barcode object is newly created: it will not show the contents of any referenced object until the sequence was updated.

8.5.3 Serial Number Advanced

The following dialog can be reached by clicking on the Advanced button in the entity property sheet of the serial number.

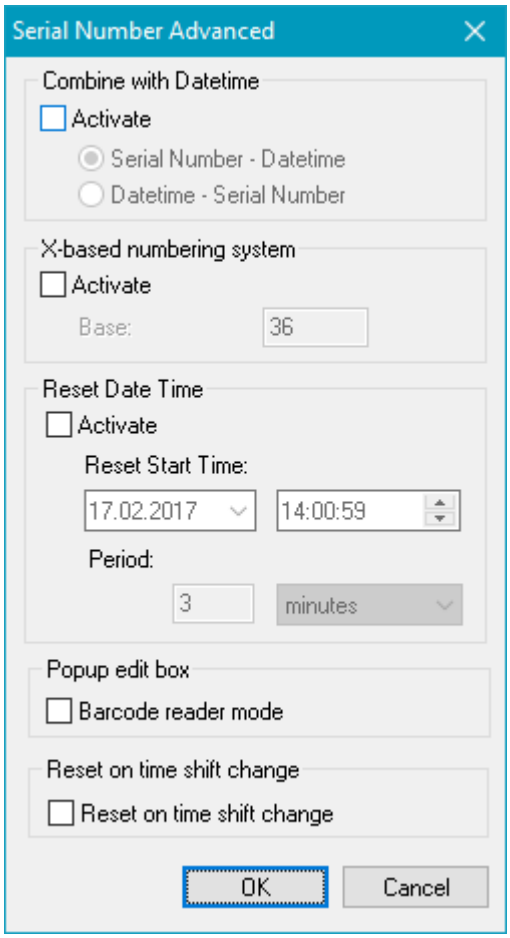


Figure 176: Serial Number Advanced Dialog

Combine with Datetime:

Activate: Adds a DateTime element to the Serialnumber.

Check: The combined string exists of Serialnumber + Date Time or vice versa.

X-based numbering system:

Activate: Allows to define a user defined base for the display of the Serialnumber.

Base: Base of numbering system, accepts values from 2 up to 36. Base 2 means binary, base 10 means decimal system and for a hexadecimal system the base is 16.

Reset Date Time: If activated the serial number will be reset after a defined amount of time. The first reset will be at the specified Reset Start Time. The following resets will be repeated after the defined Period.

Popup edit box: The Barcode reader mode can be activated. Please refer to "Barcode Reader".

Reset on time shift change: The serial number is reset whenever a shift defined in the Shift Map is changed.

The format of each element, the Serialnumber and Date Time, can be edited afterwards. Therefore Custom Format in the Serialnumber property page needs to be checked. A combined Serialnumber and Date Time element can also be displayed as a barcode. However, not each Date Time format can be converted to a adequate barcode. For example the signs ".", ":" and "/" are not taken!

The digit definitions of Customer Format are not taken for a x-based numbering system.

8.5.4 Automate Serialization

A serial defined in a text- or excel-file can be assigned to a serial number. The file contents are being read, while each row means an increment according to the serial number. This allows to handle long lists in an easy way and to define text for the entities in an independent way.

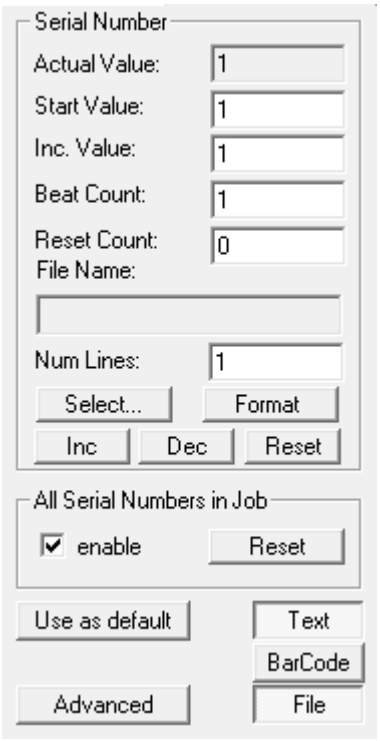


Figure 177: Serial Number property page

Pressing the File button in the Serial Number property page shows following additional fields.

File Name: Displays the file that is assigned to the selected serial number.

Select: Opens a dialog with a browse button to select a text or excel file.

Num Lines: Defines the number of lines the serial number should be assigned to.

One file can get assigned to more than one serial number.

Chapter Serial Number for the general properties of serial numbers which are also available when using the ASCII Serialization.

8.5.4.1 ASCII File

A serial can be defined in an ASCII file. Each line in the file matches to one serial number naming. The entries are indexed from 1 on. After an increment the next line of the ASCII File gets set to the assigned serial number. If a Pause Identifier is defined in Settings General the Pause Identifier string will not be assigned to a serial number object but cause a break of marking.

See chapter Automate Serialization for how to assign a file to a serial number.

8.5.4.2 Excel Table

This chapter explains how to assign an excel file (*.xls or *.xlsx) to a serial number. After pressing the select button in the serial number property page a following dialog appears.

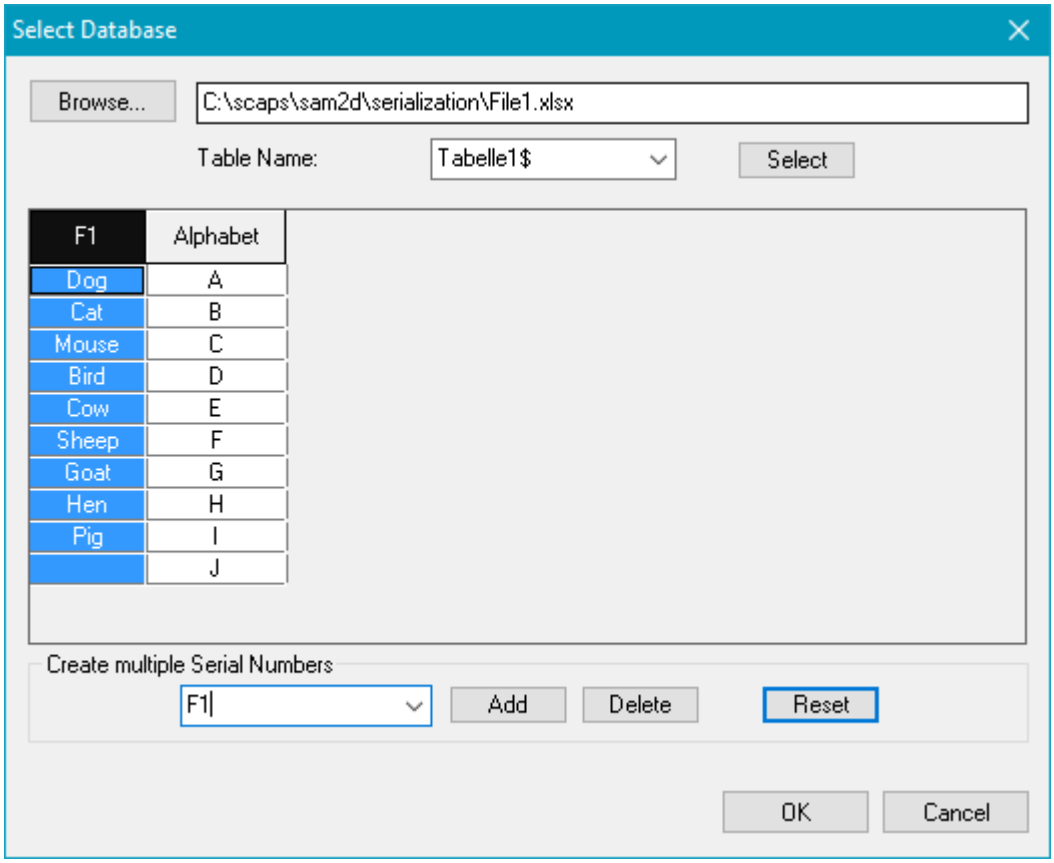


Figure 178: Select File for Serial Number Dialog

Browse: A dialog appears to select a file.

After selecting an excel file the first ten rows of table are being shown. The first row of the table file is taken as a caption of the serial and is not assigned to the serial number. If there is no entry in the first row an automatic naming is taken instead like "F1". The column which is being marked will get assigned to the selected serial number after pressing OK.

If the excel file is being changed, the file needs to be reassigned to the serial number.

*In case you have not installed Microsoft Excel 2007 or later on the PC SAMLight is running on and you are using the file format *.xlsx, please download the patch:* Microsoft access Database Engine Redistributable

Create multiple Serial Numbers: To create additional serial numbers add the according table head strings into the combo list below. Therefore select a column and press Add. After confirming with OK the serial numbers is assigned to the terms of the combo list. If the combo box is empty the current selected column item will be assigned to the serial number.

8.5.4.3 Example

The following explains how to control serialization with the help of an ASCII file.

Assumption: There are 3 pens to be marked at the time.

The names that are assigned to the pens needs to be saved in a text or excel file, for example:

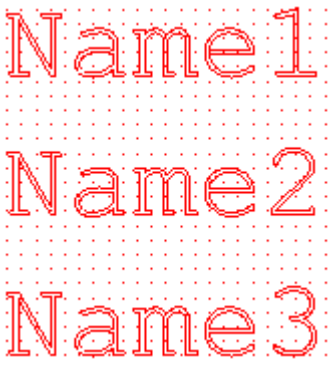
- Name1

- Name2
- Name3
- Name4
- Name5
- Name6
- Name7
- Name8
- Name9
- Name10
- ...

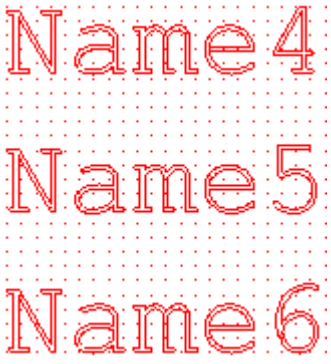
Now 3 serial numbers need to be created and the ASCII file has to be assigned to each of them. For the correct mapping of the serial string to the serial entity the following setup is used. See also: Chapter Serial Number.

	Start Value	Inc Value
SerialNUM1 :	1	3
SerialNUM2 :	2	3
SerialNUM3 :	3	3

Result:



That means the serial number itself is used as an index into the ASCII file. SerialNUM1 starts at index 1 and will be incremented by 3 after every mark. The corresponding settings for the another 2 serial numbers lead to the next mark as shown next:



8.6 Date Time

Generate a date and time object by pressing the DateTime button  in the object toolbar. Move the mouse to the desired position and press the left mouse button.

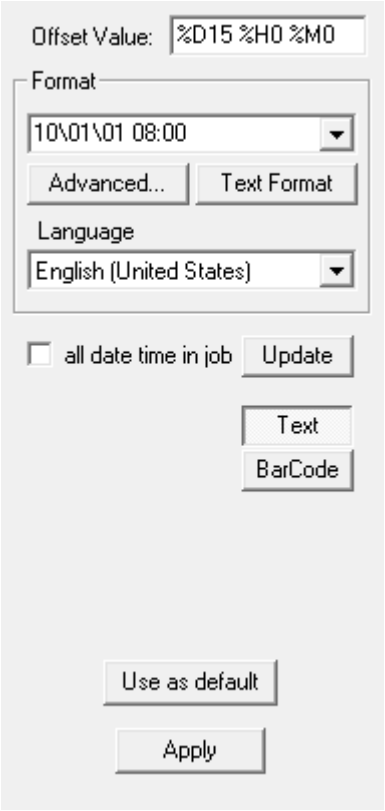


Figure 179: Date Time Dialog

Offset Value: Offset values for day, hour and minutes. The values can be negative or positive.

Format:

List with the currently available date time formats.

Advanced...: Allows to edit the date time format list. See subchapter Advanced.

Text Format: Switches to the Text property sheet. The text format can be defined here.

Language: List with all available languages.

Update: Pressing this button updates all date time objects in the job, if all date time in job is selected. Else only the selected date time object will be updated.

Text: Generates the date object as text.

BarCode: Generates the date object as barcode.

Use as default: Uses the properties of the currently selected date time object for the generation of new date time objects. The program saves these settings also for a new program start in case save settings on exit is selected in the general settings.

8.6.1 Date Time Advanced

Press the Advanced... button in the DateTime Property page to show this dialog.

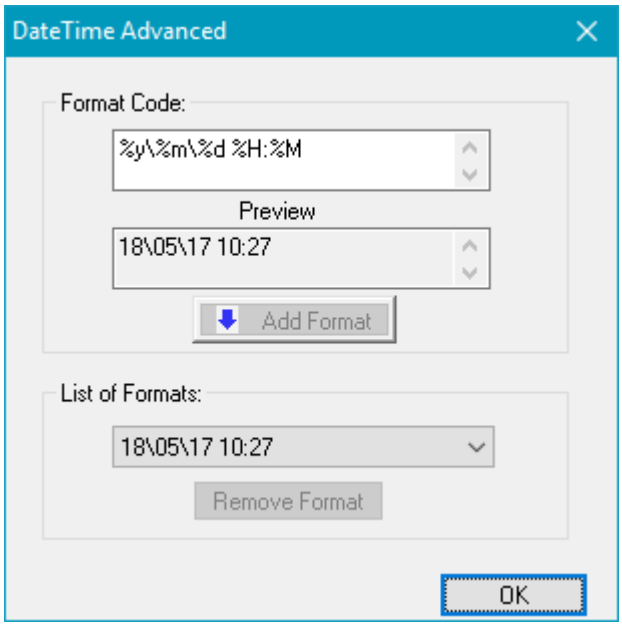


Figure 180: Date Time Advanced Dialog

Code: Enter an encoded format definition for date time. The standard formats are not editable.

Convert: Pressing the arrow button converts the format definition given in Code and extends the date time format list.

Format: List of defined date time formats.

New: Creates an empty format. It is editable in the Code field.

Delete: Deletes current format from format list. The standard formats are not deletable.

Cancel: Leaves DateTime Advanced without taking changes.

OK: Changed list is valid. It will be shown in the DateTime property page also after a new start of the program.

Format definitions:

Format	Description
%1d	Day of the month as digits without leading zeros for single-digit days.
%2d	Day of the month as digits with leading zeros for single-digit days.
%3d	Abbreviated day of the week as specified by a LOCALE_SABBREVDAYNAME* value, for example, "Mon" in English (United States). Windows Vista and later: If a short version of the day of the week is required, your application should use the LOCALE_SSHORTESTDAYNAME* constants.
%4d	Day of the week as specified by a LOCALE_SDAYNAME* value.
%1M	Month as digits without leading zeros for single-digit months.
%2M	Month as digits with leading zeros for single-digit months.
%3M	Abbreviated month as specified by a LOCALE_SABBREVMONTHNAME* value, for example, "Nov" in English (United States).
%4M	Month as specified by a LOCALE_SMONTHNAME* value, for example, "November" for English (United States), and "Noviembre" for Spanish (Spain).
%1y	Year represented only by the last digit.
%2y	Year represented only by the last two digits. A leading zero is added for single-digit years.
%4y	Year represented by a full four or five digits, depending on the calendar used. Thai Buddhist and Korean calendars have five-digit years. The %4y pattern shows five digits for these two calendars, and four digits for all other supported calendars. Calendars that have single-digit or two-digit years, such as for the Japanese Emperor era, are represented differently. A single-digit year is represented with a leading zero, for example, "03". A two-digit year is represented with two digits, for example, "13". No additional leading zeros are displayed.
%1g %2g	Period/era string formatted as specified by the CAL_SERASTRING value. The %1g and %2g format pictures in a date string are ignored if there is no associated era or period string.
%a	Abbreviated weekday name
%A	Full weekday name
%b	Abbreviated month name
%B	Full month name
%c	Date and time representation appropriate for locale
%C	Month as character digit (A-L)
%d	Day of month as decimal number (01 – 31)
%H	Hour in 24-hour format (00 – 23)

%I	Hour in 12-hour format (01 – 12)
%j	Day of year as decimal number (001 – 366)
%k	Weekday as decimal number (1 - 7; Sunday is 7)
%K	Weekday as decimal number (1 - 7; Sunday is 1)
%L	Month mapping Placeholder, see Months Map.
%l	Day mapping Placeholder, see Day Map.
%m	Month as decimal number (01 – 12)
%M	Minute as decimal number (00 – 59)
%o	Year mapping placeholder
%O	Year as a single character representation, ASCII character 'H' represents Year 2000.
%p	Current locale's A.M./P.M. indicator for 12-hour clock
%q	Week of year as decimal number, with Monday as first day of week (01 – 53)
%Q	Week of year as decimal number, with Sunday as first day of week (01 – 53)
%r, %R	Year as a single decimal number representation (0 - 9; Eg. year 2008 is 8)
%S	Second as decimal number (00 – 59)
%T	Working Shift Placeholder, see chapter Shift Map
%v	Hour to letter representation ('A' - 'Z'; 0:00 h is 'A')´
%w	Weekday as decimal number (0 – 6; Sunday is 0)
%x	Date representation for current locale
%X	Time representation for current locale
%*	Year without century, as decimal number (00 – 99) and corresponding week of year as decimal number
%y	Year without century, as decimal number (00 – 99)
%Y	Year with century, as decimal number
%z, %Z	Time zone name or abbreviation; no characters if time zone is unknown
%%	Percent sign

Table 19: Format definitions for Date Time objects

Format	Description
%#a, %#A, %#b, %#B, %#p, %#X, %#z, %#Z, %#%	# flag is ignored.
%#c	Long date and time representation, appropriate for current locale. For example: "Tuesday, March 14, 1995, 12:41:29".
%#x	Long date representation, appropriate for current locale. For example: "Tuesday, March 14, 1995".
%#d, %#H, %#I, %#j, %#m, %#M, %#S, %#U, %#w, %#W, %#y, %#Y	Remove leading zeros (if any).

Table 20: Special format definitions for Date Time objects

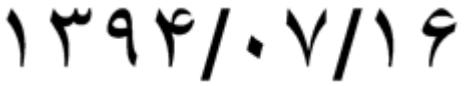


Figure 181: Example of Persian DateTime object

The Persian date 1394/07/19 is created with identifiers %4y/%2M/%2d. *"Text2D → Extended... → Decimal number substitution → Persian digit number substitution"* must be enabled to change the figures.

8.7 Text2D

Generate a text object by pressing the  Text button in the Object-Toolbar. Then move the mouse to the desired position and press the left mouse button. The text property page appears:

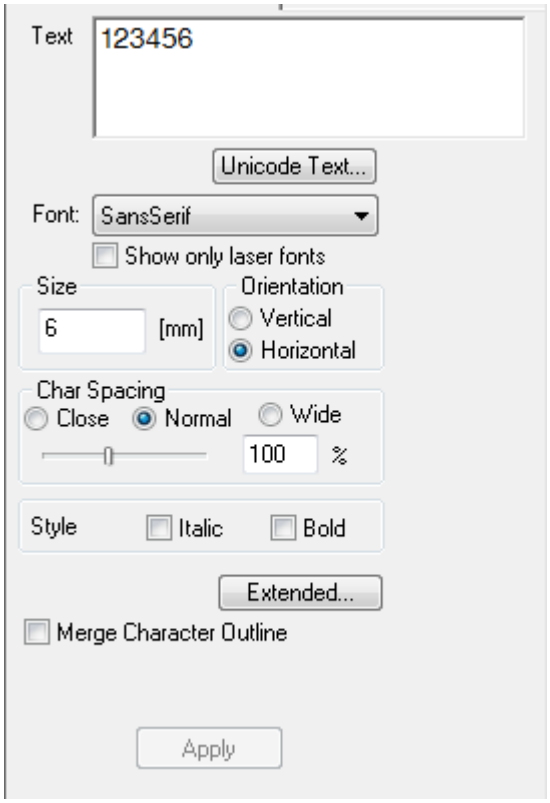


Figure 182: Text2D Dialog

Text: Input field for the text that is generated.

Unicode Text...: This will open a dialog box where special characters can be entered. For some languages this is needed when it is not possible to enter the character in the Text edit box.

Font: List with all available True Type fonts.

Show only laser fonts: Only true type fonts generated with the sc_font_convert tool are shown in the Font List. These fonts are special true type fonts and the text generator will generate simple line characters for a fast marking processes. For more detailed information and how to generate your own simple fonts see Generate Simple Fonts.

Size: Maximum height of the characters.

Orientation: Horizontal or vertical text.

Char Spacing: Spacing between the single characters in percent.

Style: Italic or Bold characters. Bold characters are not available for Simple Fonts.

Extended...: Opens the Text2D Properties Dialog for more features.

Merge Character Outline: If text characters within one ScWinTextChars2D entity intersect, the outlines of these characters will be merged. See Fig. 183 for an example: on the left hand side, Merge Character Outline is enabled, on the left hand side it is not. Merging of character outlines also works automatically for characters in serial numbers.

Figure 183: Example for Merge Character Outline.

8.7.1 Text2D Properties

Clicking the *Extended...* button in the Text2D property page the following dialog appears:

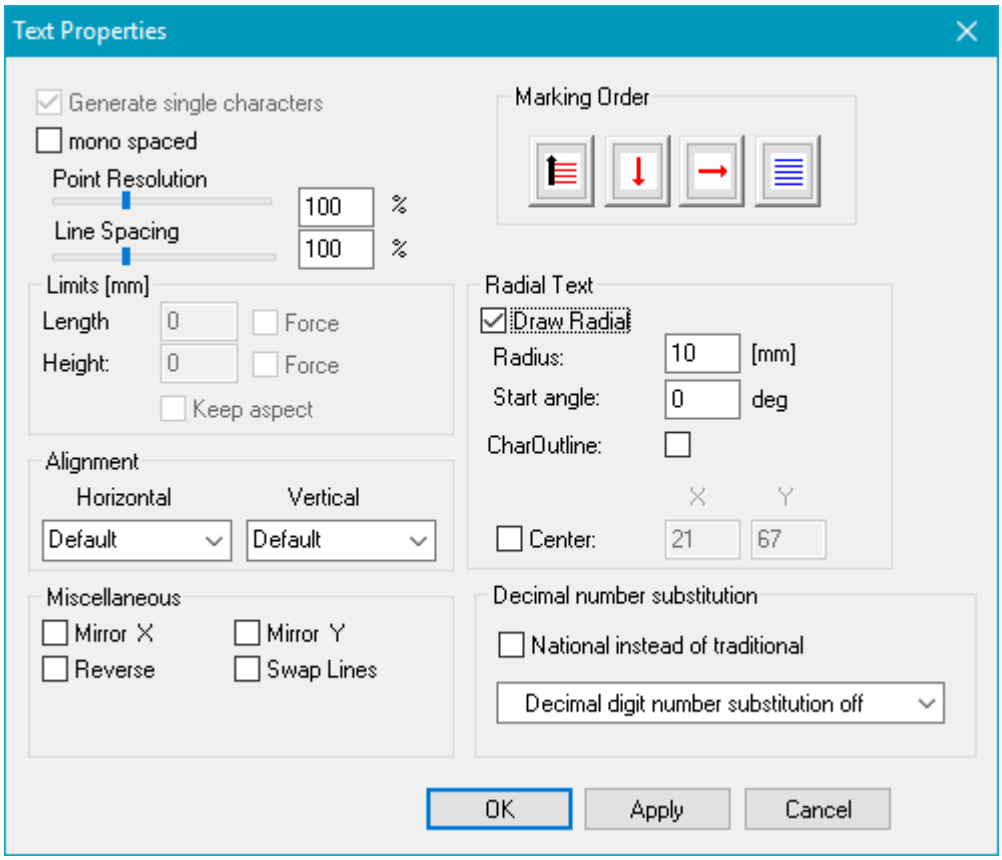
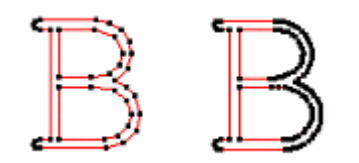


Figure 184: Text Properties Dialog

Generate single chars: When this option is selected all characters in the text string are generated separately, which means that each character is stored in one separate ScWinText2D object. This has the advantage that each character can be accessed and by this way each character can be hatched with a different hatch style for example. If this is not necessary the option should be switched off to fasten operations like transformation and rendering.

mono spaced: If selected, the distance between two characters is constant.

Point Resolution: The resolution of the lines. Not available for Simple Fonts.



30 % **100 %**

Line Spacing: Spacing between the lines in percent.

Limits: Defines maximum length / height. If Force is checked the text is adapted to the entered length / height. For radial mode the angle of the radial text can be limited. To define an angle limit *CharOutline* needs to be checked.

Alignment: The horizontal and vertical alignment of the text can be defined in relationship to the generation point.

Draw Radial: Activating radial mode. Radius and Start angle define the position of the characters.

CharOutline: Takes the outlines of character into account when they get disposed radial. Active if radial mode is selected.

Center: Here the initial center coordinates of a radial text can be specified. If this option is selected the center of the radial text is positioned once at the coordinates specified with X and Y.

Use as default: Uses the properties of the currently selected text object for the generation of new text. The program saves these settings also for a new program start in case save settings on exit is checked in the general settings.

Marking order: The buttons are only active if generation of single characters is checked. The state of the buttons is changeable by clicking.

Sets the main direction of the marking order of characters. If case y is selected as main direction the characters get sorted line by line, if case x is selected as main direction they get sorted column by column.

Sets the orientation of the y direction.

Sets the orientation of the x direction.

With this state an unidirectional sort is defined. Also bidirectional is selectable, which results in a zigzag sorting order.

8.8 Control objects

A control object can be defined in between the chronological process of the job. Provided objects are:

- ScTimer
- ScWaitForInput
- ScSetOutput
- ScSetAnalogOutput
- ScExecutable
- ScMotionControl
- ScMotfOffset
- ScWaitForTrigger
- ScAutoCalib
- ScOverride
- ScJump

8.8.1 I/O Control Objects

There are three different control objects available.


The control objects *ScTimer*, *ScWaitForInput* and *ScSetOutput* can be accessed in the toolbar. When a control object is created it appears in the entity list.

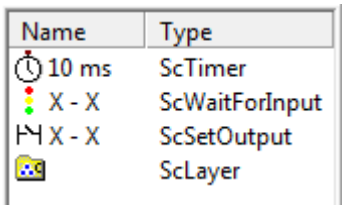


Figure 185: Entity list with I/O Control Obejects

To change the properties of a control object it has to be selected by clicking on it in the entity list. Then the Control property page can be activated and the properties can be changed:

ScTimer:

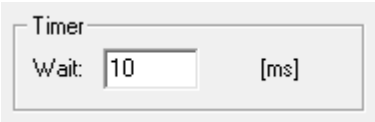


Figure 186: ScTimer

Wait: Interrupts the marking process for n ms, in this example 10 ms.

ScWaitForInput:

Figure 187: Wait For Input

Wait For Input: Stops the marking process until the specified input bit(s) of the IO port is set to high or low.

X: Do not care / Ignore input bit.
0: Wait for corresponding *bit state 0*.
1: Wait for corresponding *bit state 1*.

Combine: Allows to select multiple input bits. All need to be in the selected state to proceed.

Bit position count starts with "1", but corresponds to bit "0" at the hardware!

Message: If the check button *Active* is selected a message box appears containing the text defined in *Message* when the specified input bit(s) is(are) *high*/*low*. The marking process continues after the message box has been replied to.

Default name of the control is for current shown state: '5 - H' (masked bit 5, wait for bit high state (H))

ScSetOutput:

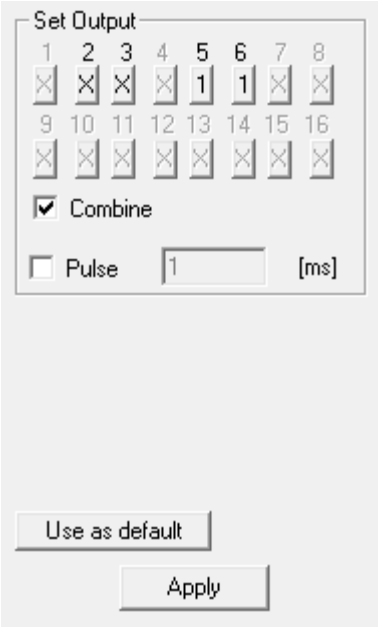


Figure 188: Set Output

Set Output: Sets the specified output bit of the IO port to high/low.

X: Do not care / Ignore input bit.
0: Wait for corresponding *bit state 0*.
1: Wait for corresponding *bit state 1*.

Combine: Allows to set multiple output bits at the same time.

NOTE

Bit position count starts with "1", but corresponds to bit "0" at the hardware!

Pulse: If selected the output bit gets set to its previous state again, n ms after the bit was set.

Default name of the control is for current shown state: '0036 - 0034' (mask is hexadecimal 0036, output state for this mask is hexadecimal 0034)

8.8.2 Analog Output Control Object

With this control object  you can set the Analog Output Value of the DAC A or DAC B to the desired value:

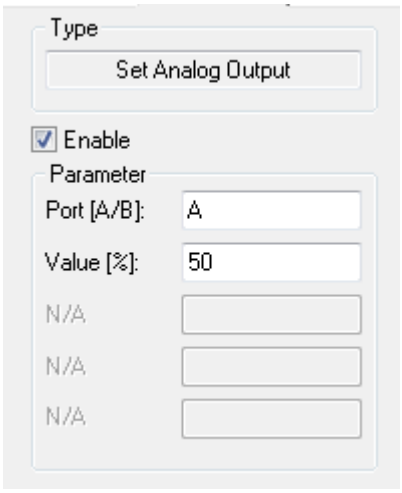


Figure 189: Set Analog Output

Enable: Enables the control.

Parameter:

Port [A/B]: Choose port A or B for control

Value [%]: Define output value in %

8.8.3 Executable Control Object

An executable control object can be used to start a program at a defined position within the job. Therefore click on the icon in the toolbar. Then a new entry will be generated in the entity list. In the control property page at the right hand several parameters can be entered.

Figure 190: Executable Control Object

Enable: If activated the control object is activated and will be performed within the next marking process.

Parameter:

Path: Defines the full path of the program that should be executed.

Arg1/Arg2: Enter command line arguments here.

8.8.4 Motion Control Object

Before the ScMotionControl object can be used the motion controller must be set up.


The ScMotionControl object can be accesed in the *Object Toolbar*. Click on the motion control object to modify it within the *Control* property page. In this page the parameters can be assigned to the object. These can be motion control parameters as well as string parameters.

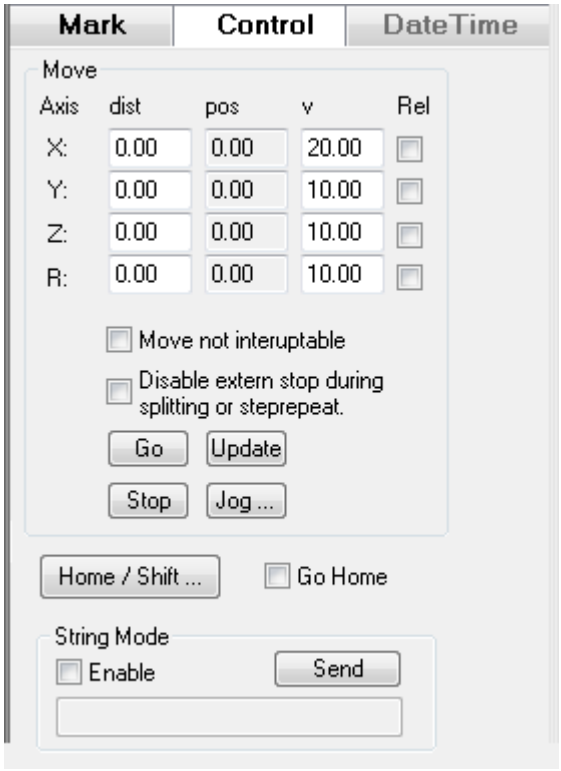


Figure 191: Job control object motion dialog

Move:

Axis: Movement distance and speed definitions for the axis. Each axis requires one control card.

Rel: If checked a relative movement is performed instead of an absolute movement.

Go Home: This checkbox must be enabled for setting up a homing procedure in a ScMotionControl object. In the Home / Shift dialog the related axes must be chosen.

String Mode: A RS-232 string command can be send to the motion controller.

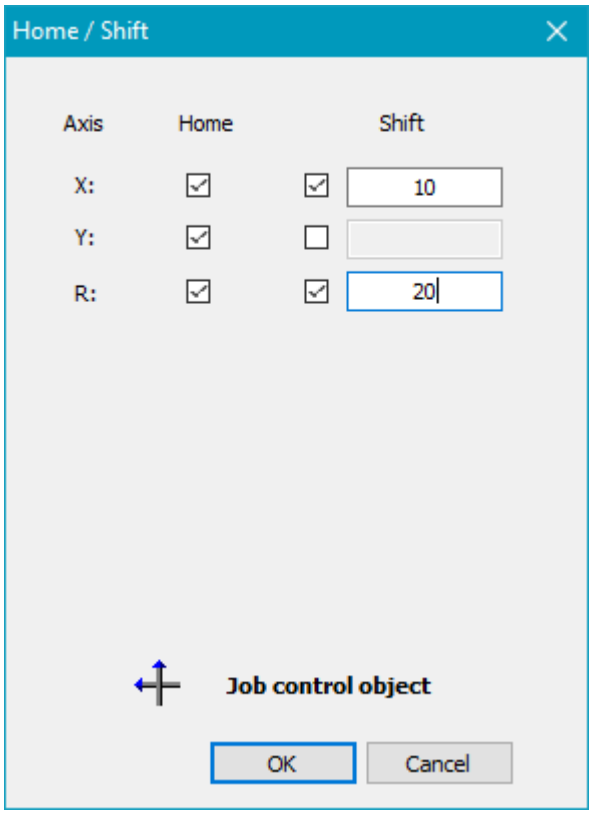


Figure 192: Job control object motion Home / Shift dialog

Home: When the ScMotionControl object is executed, all axes with enabled *Home* checkbox will start the homing procedure.

Shift: When the ScMotionControl object is executed, the shift values will be applied. To reset the current shift of an axis, enable the checkbox and set the value to *0*. The *Home* and *Shift* functionalities can be combined, first the axis will do the homing procedure, then the shift will be applied.


The ScMotionControlGo object (not for motion type 8) has the same functionality as the MotionControl object described above. The difference is that SAMLight will continue with the following entities immediately, so SAMLight will not wait until the motion is finished. Please mind that the motion command will continue even if the marking in progress signal is off at the end of the entity list.

8.8.5 Trigger (USC and RTC5 only)

For USC-1, USC-2 and RTC5 scanner cards, the software offers some extended functionality to handle trigger events, to react on them delayed and to generate internal trigger signals. This functionality is useful e.g. to set up jobs that are larger than the available working area or to have defined distances between elements of a job. To set up such an advanced job, there are Trigger Control Objects available within the functional object toolbar.

 ScWaitForTrigger pauses the job execution until a trigger signal is received.

- USC-1: Trigger Control objects do not work in MOTF simulation mode.
- USC-1/2/3: The trigger signal can come from an external input or from a preceding ScMotfOffset control object.
- RTC5: The ScWaitForTrigger control object does only work with a preceding ScMotfOffset entity. In order to wait for an external signal a ScWaitForInput control object has be used.

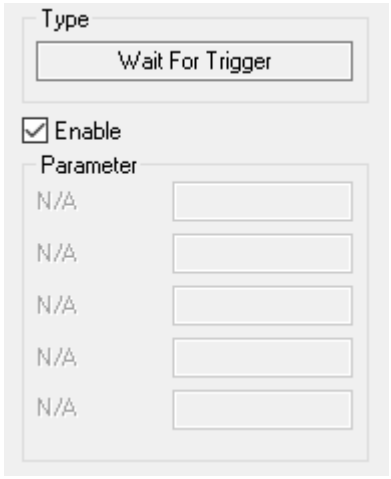


Figure 193: Wait For Trigger

Wait For Trigger:

Enable: If the check check box is enabled, this entity will pause the marking procedure until an internal or external trigger is recognized.

For USC cards, the external start is equal to the signal of the OPTO_IN_0 bit. Therefore, this entity can be used instead of a ScWaitForInput entity which is listening to bit number 1.

An internal trigger can be defined with a ScMotfOffset entity which is described below.

ScWaitForTrigger control objects do not work in MOTF simulation mode for USC-1 cards.

 ScMotfOffset (option MOTF required) defines a distance in the entities property page Control.

Figure 194: Wait For Trigger

MOTF Offset:

Enable: If the check check box is enabled, a distance in mm can be defined. This distance will be measured in MOTF encoder counts like it is defined in the card-specific MOTF settings. After the distance has been reached, this entity will generate an internal trigger pulse. This internal pulse can be recognized by a ScWaitForTrigger entity.

React to external trigger: By default a ScWaitForTrigger entity will wait for an external or an internal trigger. That means, the pausing of an ScWaitForTrigger entity can be interrupted by an external trigger which will lead to a shorter distance than defined. This can be avoided by activating this check box. In this case, the first external trigger will start the pausing defined in the ScWatiForTrigger entity and every other external trigger will be ignored until the internal trigger of the ScMotfOffset entity has been sent.

When the MOTF distance (calculated by encoder counts) has increased by the ScMotfOffset distance, an internal trigger signal is generated. The internal trigger signal behaves like an external trigger, so the external trigger signal must be low. The ScMotfOffset control object itself does not cause any delay or waiting operation.ScMotfOffset and ScWaitForTrigger together can be used in MOTF applications to separate two marking objects with the distance of ScMotfOffset. This allows you to set up a job where the marking result is bigger than the working area. The order of the entities in the MOTF job should look like this:

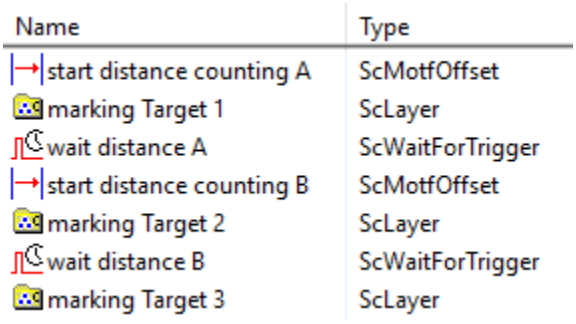


Figure 195: Entity list with Trigger Control Objects

Assume we use the Trigger Mode. Job process would be:

- One external trigger signal starts the job process
- Counting for distance A starts
- Target 1 is marked
  (marking must be done before distance A is traversed)
- Waiting until distance A has passed by
  (external trigger must be low)
- Counting for distance B starts
- Target 2 is marked
  (marking must be done before distance B is traversed)
- Waits until distance B is passed by
- Target 3 is marked
- Job done

ScMotfOffset and ScWaitForTrigger together can also be used to define an initial distance offset after that the first marking part has to be started.

8.8.6 AutoCalib (RTC only)

For RTC4 and RTC5 scanner cards, an AutoCalib control object can be inserted into the entity list to perform an automatic self-calibration of an appropriate scan head, which supports this functionality. The corresponding ScAutoCalib symbol  within the functionality object toolbar has to be activated by checkbox *AutoCal* within *sc_setup.exe* → *Hardware Settings* → *Settings* → *Driver Settings* → *Mode* before. To change the properties of an AutoCalib control object it has to be selected by clicking on it. Then the *Control* property page can be activated and the properties can be changed:

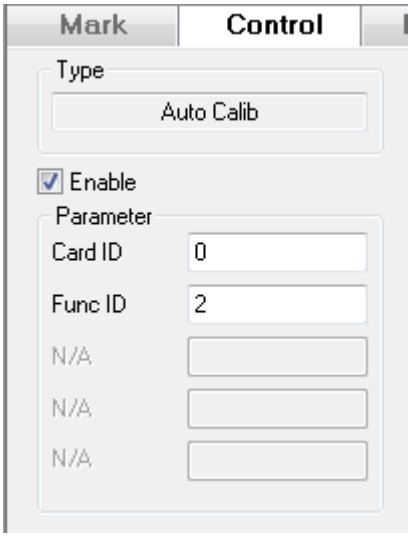


Figure 196: AutoCal Control Object

Enable: This checkbox activates the AutoCal control object.

Parameter:

Card ID: Primary card: 0, secondary card: 1, default = 0

Func ID:

1. Turn auto calibration mode on and go to the reference position for an initial calibration
2. Recalibrate
3. Turn off auto calibration mode

The Func ID parameters are different from those of the RTC manuals.

The output value can be checked by the Command View command *AutoCal* (here it corresponds to the RTC manuals).

8.8.7 SetOverride Control Object

Inserting a SetOverride control object  into the entity list, the marking speed, the power and the frequency of the applied pen can be changed using the Override functionality of the Mark property page. This is useful for determining the optimal marking speed, power and frequency parameters by marking an array of identical entities with different speed, power and frequency. The test array can be created by copying a row of entities, in which a SetOverride control entity changes one parameter of the following array entity. In addition the SetOverride control entity at the beginning of the row changes the second parameter.

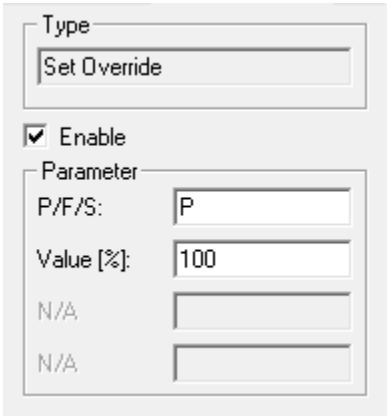


Figure 197: SetOverride Control Object

Enable: This check box activates the *SetOverride* control object.

Parameter:

P/F/S:
- P = Power
- F = Frequency
- S = Speed

Value[%]: The override factor is expressed in percent and can be used to increase or decrease the P/S/F value of the pen during the mark process for the marking of all the following entities in the entity list.

8.8.8 ScJump Control Object

Inserting a ScJump control object  into the entity list, creates a jump with the possibility to define specific X, Y, and Z coordinates. Alternatively, the home position can be addressed.

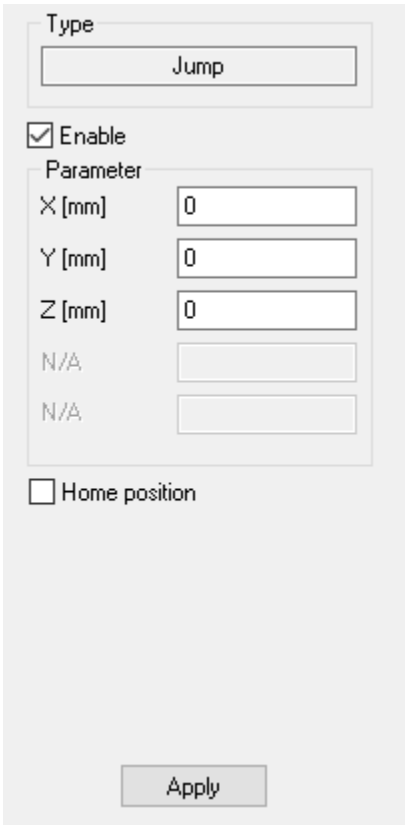


Figure 198: ScJump Control Object

Enable: This check box activates the *ScJump* control object.

Parameter:

X/Y/Z:
- X = X coordinate in mm
- Y = Y coordinate in mm
- Z = Z coordinate in mm

Home position: When activated, the jump parameters from the home position are applied.

9 Entities Properties

This chapter describes how to change the properties of the entities.

9.1 Transformations

Translation, scaling, rotation and slant operations can be done on all geometric objects. If the optic dimension 3D tool is available, also 3D Transformations are possible.

9.1.1 2D Transformations

For the transformations by mouse, see chapter View2D, Manipulation of Objects.

Transformations by keyboard: The dimension property page can be used to change the dimension of the selected entity. This page has no Apply button. The Translate, Scale, Rotate and Outline operations can be executed separately by typing in the requested values and clicking the corresponding Go button.

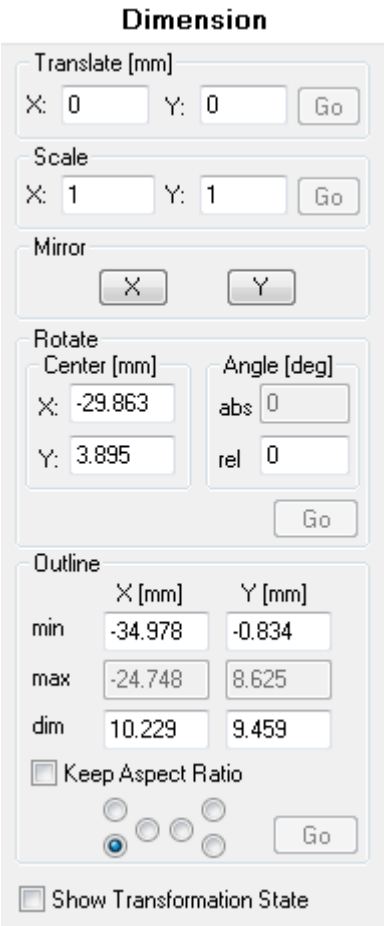


Figure 199: 2D Transformation Dialog

Translate: Translation values X and Y are relative values to the actual position.

Scale: Scale values X and Y are relative values to the actual size of the entity.

Rotate:

Center: The default coordinates of the rotation center are the center coordinates of the selected object. After each transformation the center of the object is recalculated.

Outline:

min:

X: Positions the left border of the outline

Y: positions the lower border of the outline.

max:

X: Positions the right border of the outline

Y: positions the upper border of the outline.

dim: Defines the width X and the height Y of the outline. If Keep Aspect Ratio is selected, the relation between X and Y value keeps constant after an outline transformation.

center: Active if the right middle radio button is selected. Positions the middle point of outline.
Radio buttons: Shows according min/max x/y edit fields, e.g. with selecting the upper left check box the upper left outline point can be repositioned.

Show Transformation State: When this is selected a dialog appears which shows the absolute translation, scale and rotate values of the selected object. See image below:

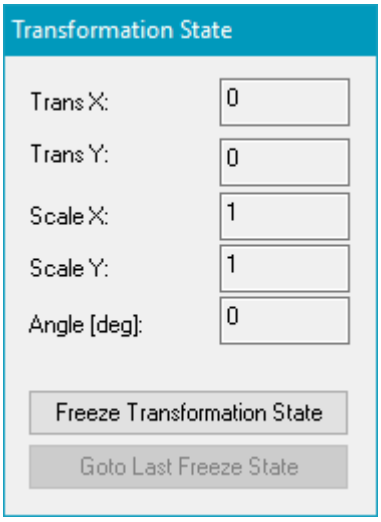


Figure 200: 2D Transformation State

TransX, TransY: Shows the translation in X and Y direction.

ScaleX, ScaleY: Shows the scaling in X and Y direction.

Angle: Shows the rotation angle of the entity.

Freeze Transformation State: Saves the current state, so that it can be reproduced later if some changes have been made and you wish to make them undo.

Goto Last Freeze State: Reproduce the state that was previously saved with "Freeze Transformartion State".

9.1.2 3D Transformations

The Z-Dimension property page can be used to change the z-dimension of the selected entity. This page has no Apply button. The Translate, Scale and Outline operations can be executed separately by typing in the requested values and clicking the corresponding Go button.

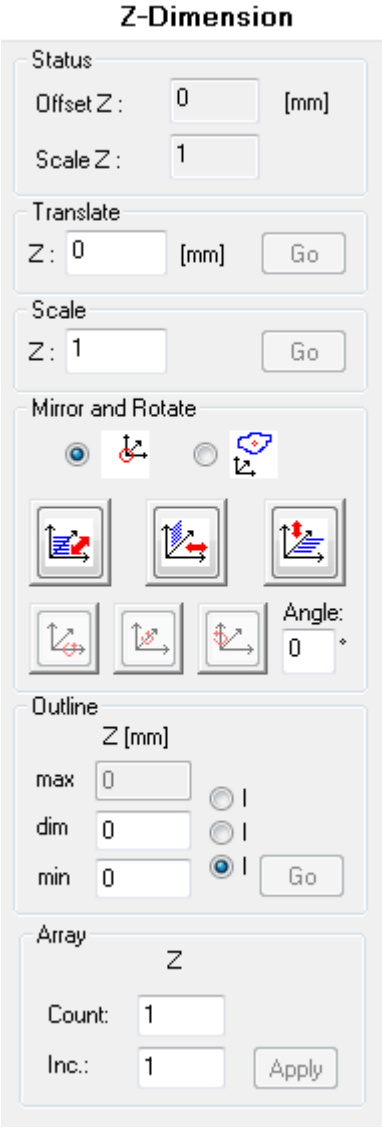


Figure 201: 3D Transformation Dialog

Mirror and Rotate: With the radio button the center of mirroring and rotation is selected.

 The center is the origin point of the world coordinate system.

 The center conforms with the center of the selected object.

Pressing the first mirror button the selected objects gets mirrored to the xz-plane. The painting plane is the xy-plane.

The rotation buttons are becoming activate when an angle is entered. The angle range is between -360 and 360 degrees.

Outline:

max, min: Positions the outline in the z-coordinate.

dim: Scales the object in z direction to the width given in dim.

Array: The array functionality allows to create reference copies of the selected entity in Z dimension. Reference copies in X and Y dimension can be found in the Entity Info property page.

Count: Define the number of copies in Z dimension.

Inc.: Define the distance between the copies in Z dimension.

The selectable radio buttons aside allow to enable any of the outline value fields, but disable each field which is not editable for the same translation to avoid conflicts.

The Z-Dimension property page requires the Optic3D option.

9.2 Hatch

This page can be used to generate hatching for entities which have closed PolyLines. Single objects as well as clustered groups can be hatched. A pen for a specific hatch can be selected on the mark property page.

For a Hatch to be actually marked, the according checkboxes 'MarkFlags' have to be enabled in the Pen Settings of the selected Pen (*Mark → Edit... → Misc*).

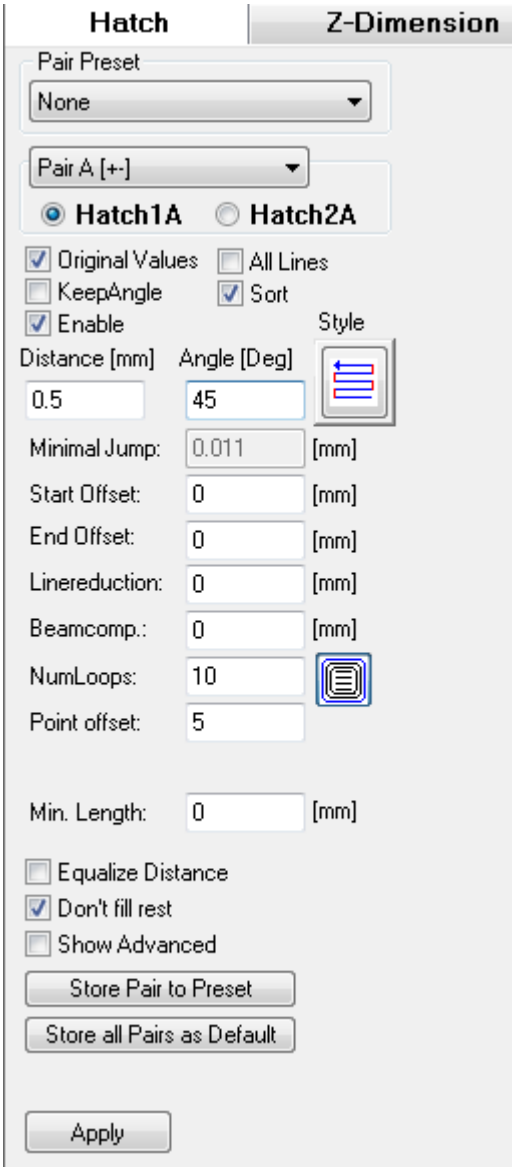


Figure 202: Hatch Dialog

Pair Preset: Access up to 10 different preset parameters.

Pair A [--]: Access up to 5 pairs of hatches (A, B, C, D and E) totaling to 10 possible hatches. The [--] indicates that there is no hatch enabled. A [+-] would indicate that the Hatch1A is enabled.

Hatch1A, Hatch2A: Chose a hatch within a hatch pair.

Order of hatching: 1A, 1B, 1C, 1D, 1E, 2A, 2B, 2C, 2E

Original Values: This switch is useful if the entities have been scaled or rotated after hatching and allows to rehatch the entity with the original values.

KeepAngle: If checked, the angle is relative to the entities rotation when rehatching the object. So if rotating a hatched entity and rehatching the entity with this flag set, the hatch lines are relative to the entity rotation angle.

Enable: If this option is being chosen the current hatch definition is activated.

All Lines: If this is not checked, ScPolyLine2D structures are used for calculating hatches. If checked, also ScLineArray2D objects are taken into account.

Sort: Changes the order of the hatch lines to decrease the total jump length. It also can flip hatch lines (change the marking direction). Below you can find an example, the marking order of the hatch is 1, 2, 3, 4, 5. If Sort is not enabled the hatch order is simple from bottom to top. At the crossing points the sort can cause problems depending on the material / application.

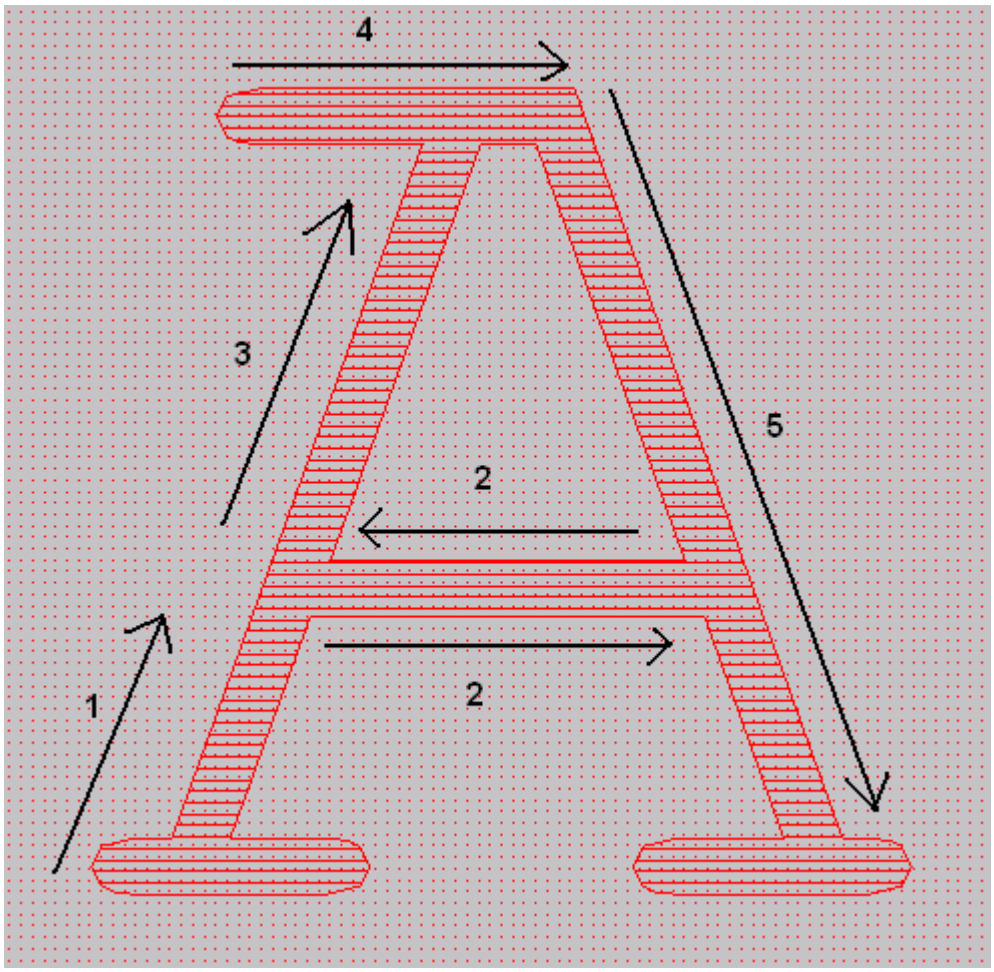


Figure 203: Hatching order with Sort Flag activated

Distance: Distance between two hatch lines in mm.

Angle: The angle to the x-axis in degrees.

Style: The movement direction can be defined by clicking on the button with the bitmap. The blue lines on the bitmap show the mark lines and their direction. The red lines show the scanner jumps. See Hatch Styles for further information.

Minimal Jump: For continuous hatch styles and a minimal jump distance can be set. If the minimal jump distance is smaller than the distance between the end and beginning of two hatch lines, then the hatch lines are not connected. See the example below.

Start Offset: Defines the start distance between the outer line of the object and the first hatch line. The default value is 0 which means that the value entered at "Distance" is taken as Start Offset.

End Offset: Defines the end distance between the outer line of the object and the last hatch line.

NOTE

This distance gets approximated from the hatcher as the conditions Start Offset and Distance do not allow an exact definition of End Offset (for an exact definition of End Offset see: Equalize Distance).

Linereduction: Shortens the hatch lines.

Beamcompensation: Allows the reduction for the hatches to avoid overburning. See also the example picture below.

NumLoops: The number of inner loops can be defined. See the example of blue concentric circles below which was created by a circle with NumLoops. By clicking on the button, the inner loops are activated. Then, you can choose between marking order from outside to inside  or from inside to outside  .

Point Offset: Only active when NumLoops is enabled. This will make the hatching lines start at different points from the hatch. For example if you have a circle hatched with NumLoops so that you have some concentric circles every circle marking will start at a different angle.

Min. Length: If this value is bigger than 0, all hatch lines shorter as this value are not generated.

Line step: If the value is bigger than 1, then the processing sequence of the hatch lines is not from one hatch line to the next, but only every xth hatch line is marked first, den with an offset of 1 the adjacent one until all hatch lines are marked, e.g. Line step = 3: first hatch lines 1,4,7, ... until the end are marked, then the lines 2,5,8, ... and finally the lines 3,6,9, ... With this feature the energy deposit on the material can be reduced and distributed more uniform and so a penetration can be prevented.

Equalize Distance: Changes Distance so that Start Offset and End Offset can exactly be set by the hatcher.

Don't fill rest: If checked only NumLoops gets hatched. The object gets not hatched inside the loops.

Show Advanced: Opens a dialog where additional hatch properties can be activated.

Store Pair to Preset: This button stores the pair which is actually selected in the drop down menu (A, B, C, D, or E) to the Pair Preset selected in the drop down menu (Hatch style 1, Hatch style 2, ...,or Hatch style 9). Note that the Pair Preset has to be selected first, then the pair can be chosen, can be adjusted, and can be stored.

Store all Pairs as Default: Uses the actual hatch style of the currently selected object as default for new entities. Thus, the settings of all 5 pairs of the drop down menu (A, B, C, D, or E) are stored directly and can be applied to a new entity. Note that the program saves these settings also for a new program start in case save settings on exit is checked in the general settings.

Apply: After defining the distance the angle and the style clicking on button Apply will generate the hatch lines.

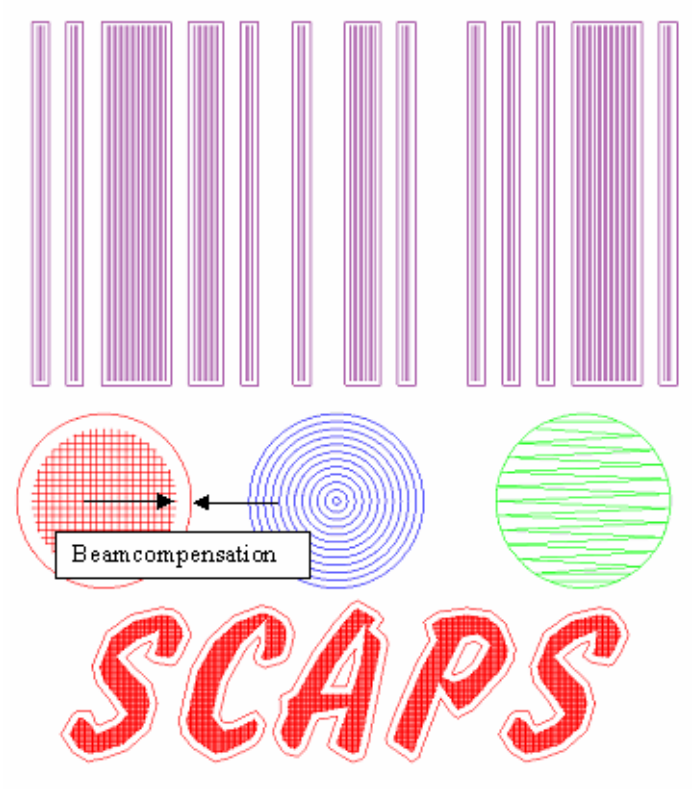


Figure 204: Hatch examples

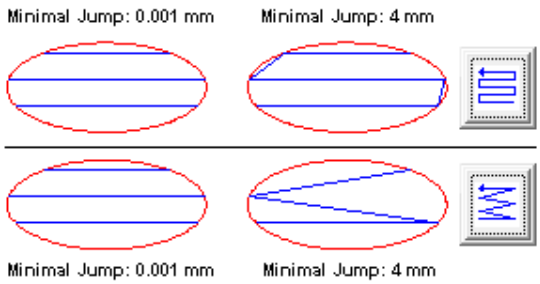


Figure 205: Minimal Jump examples

9.2.1 Hatch Style

The following hatch styles can be chosen:

The marking direction is alternating. Jumps are executed between the hatch lines.

The marking direction is conserved from left to right. Jumps are executed between the hatch lines.

The marking direction is conserved from right to left. Jumps are executed between the hatch lines.

Hatch lines are closed and follow the outline of the entity. Jumps are executed between the hatch lines.

For this continuous hatch style, the marking direction is alternating. Hatch lines are connected (laser remains on) if the distance between end and start point of lines is smaller than the minimal jump distance.

For this continuous hatch style, the marking direction is alternating. Hatch lines are connected (laser remains on) if the distance between end and start point of lines is smaller than the minimal jump distance.

Please note: this style leads to non-parallel hatch lines.

Switching between the different hatch styles is done by clicking on the hatch style icon. The blue lines on the bitmap show mark lines and their direction. The red lines show the scanner jumps.

9.3 Entity Info

The Entity Info property page shows the type and the name of the selected entity.

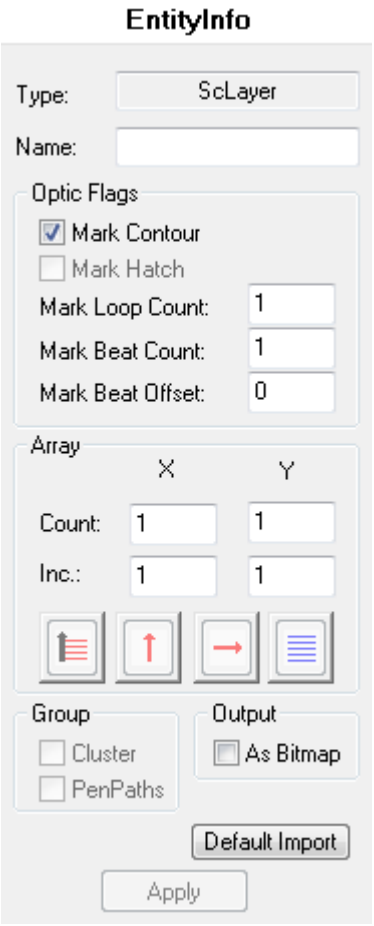


Figure 206: Entity Info Dialog

Name: Here a name can be set for an entity that can be used to decide what its purpose is within the job, to access it via the Client Control Interface or to reference it from within a serial number.

Optic Flags: The markable flags can be changed within this page separately for the contour and for the hatches generated with the hatcher. This flags define whether to mark the entity or not.

> **Mark Loop Count:** Defines how often an entity is marked when a mark command is issued. For example, a circle can be marked 100 times with every mark command to form a hole inside a material. If -1 is entered here the entity will be marked infinitely often.
>
> **Mark Beat Count:** Defines after how many subsequent mark commands an entity should be part of the mark.
>
> **Mark Beat Offset:** Allows a shift of the Mark Beat Count for every entity.

Array: The array functionality allows to create reference copies of the selected entity in X and Y dimension. Reference copies in Z dimension can be found in the Z-Dimension property page.

> **Count:** Define the number of copies in X and Y dimension.
>
> **Inc.:** Define the distance between the copies in X and Y dimension.

Group:

> **Cluster:** If the entity is a group, it can be clustered. A clustered group will be hatched as one single object while in a unclustered group each entity of groups is hatched separately.
>
> **PenPaths:** If not activated the entities in a group will be marked entity after entity each entity being as often repeated as the individual PenPath feature is configured. If checked all entites are marked once in one loop and the next loop again until the PenPath of the entity with the lowest pen number is completed.

Output As Bitmap: If an entity is selected then by activating this checkbox the entity is transformed to a Bitmap. After that the Bitmap property page becomes available and the bitmap can be modified like a normal bitmap.

Marking examples:

Assume there are 3 entities inside a job with following settings:

Entity	Mark Loop Count	Mark Beat Count	Mark Beat Offset
1	1	1	0
2	10	2	0
3	5	3	1

Table 21: Example for three entities with different mark loop count, mark beat count and mark beat offset

With these settings the following mark sequence can be generated:

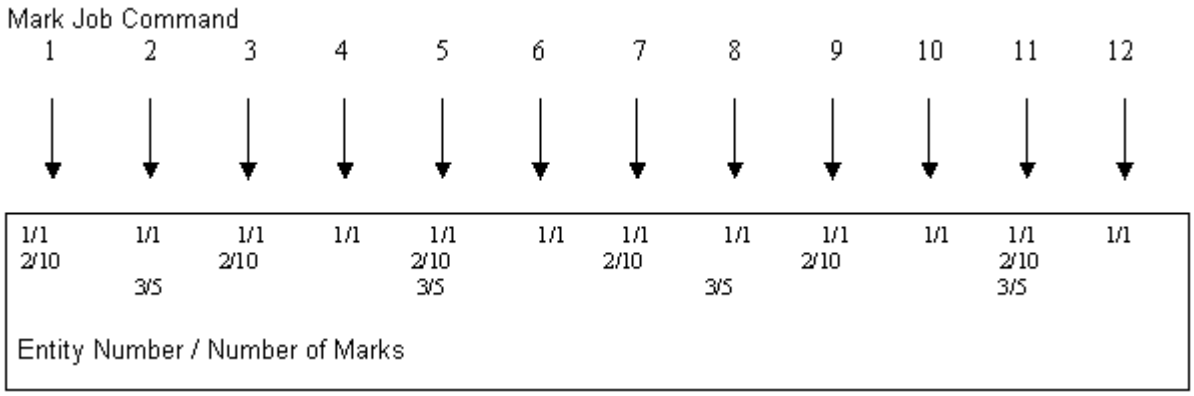


Figure 207: Example for marking order with mark loop count, mark beat count and mark beat offset

- Entity 1 is marked every mark one time (default).
- Entity 2 is marked every second mark and then 10 times.
- Entity 3 is marked every third mark and then 5 times. The Beat Count is offset by 1. So the first time the entity will be marked is on Mark Command 2.

9.4 Element Info

Gives information about a single polyline structure.

Figure 208: Element Info

Vectors: Number of lines of the selected polyline.

Length: Length of the polyline.

Outer / Inner: Orientation of the polyline

outer: counterclockwise

inner: clockwise.

Open: The polyline is open or the polyline's structure it belongs to is open.

9.5 Styles

The style property page has to be enabled in the *Settings* → *System* → *Extra* dialog.

A style consists of several properties and can be assigned to an entity or a group. It is possible to store these information from a single entity to a style as well. You can create a style library for different scenarios, for example different materials, deep engrave or cleaning. The styles will be stored in your SAMLight settings file (default: <*SCAPS*>\system\sc_light_settings.sam).

Here is a list of the properties a style can handle:

- Pen (A pen with a pen-path can be used as well)
- Hatches
- Mark loop count
- Mark contour
- Mark hatch

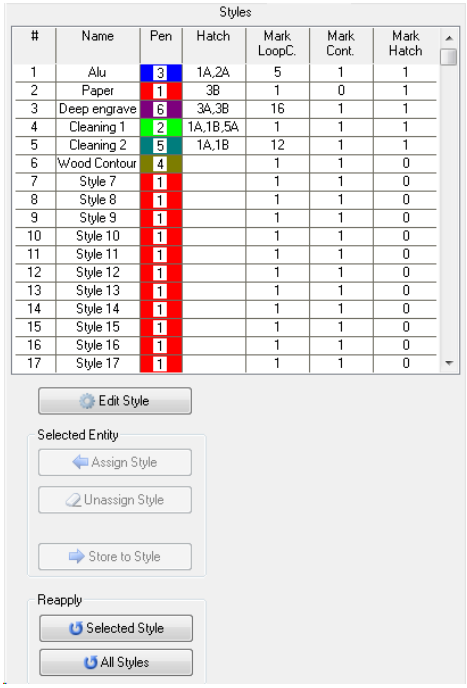


Figure 209: Style Property Page

Edit Style: Opens the Edit Style Dialog

Selected Entity: These three buttons only affects the entities which are selected in the entity list.

Assign Style: Assigns the selected style to the selected entities. All properties of the style will be copied to the entities.

Unassign Style: Unassigns the style of the selected entities. The properties of the entities will not be changed.

Store to Style: All properties of a single selected entity will be stored in the selected style.

Reapply: If a style has been changed after it has been assigned to entities the parameters of these entities will not be changed automatically due to the changes of the style. If you want to change the parameters of the entities you have to reapply the styles.

Selected Style: Copies the current defined properties of the selected style to all entities assigned to this style.

All Styles: Copies the current defined properties of all styles to all entities assigned to each style.

Import styles from a *.ssf settings-file: with right mouse click on the styles table there is an option to import a *.ssf file with all the saved styles settings. This feature works also in SAM3D style.

Export styles to a *.ssf settings-file: with right mouse click on the styles table there is an option to save all the edited styles in a *.ssf settings-file. This fearute works also in SAM3D style.

Coloring the style list: A style assigned to a selected entity can be seen in the color of the current row in the style property page. If the color is **blue** [1] the selected entity has not been assigned to any style. If the color of the current style is **green** [2] the entity has been assigned to the highlighted style and the style has not been changed since the assignment. If the color is **brown** [3] the selected entity has been assigned to the highlighted style but the style has been changed afterwards. If you want to update the parameters of the selected entity corresponding to the changes of the assigned style you have to reapply the style.

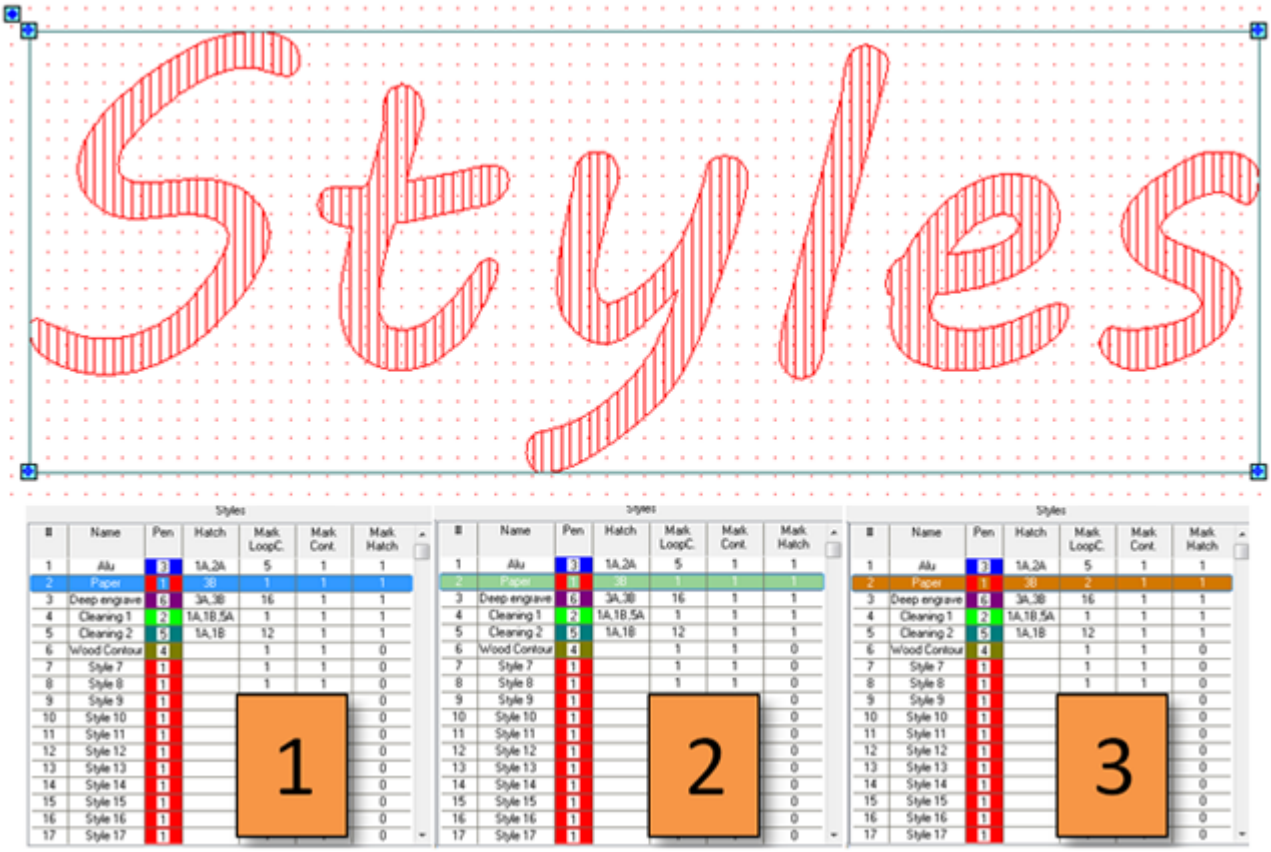


Figure 210: Applying a style to a selected entity: 1) blue row: no style assigned, 2) green row: style assigned, 3) brown row: style has been assigned but after that the style has been changed

NOTE

There are two different ways to activate/deactivate the marking behavior of the contour and the hatch. The first one is in EntityInfo and the second possibility is in the pen-settings. The style property page shows only the information of the EntityInfo. If you have deactivated mark contour or mark hatch in the pen-settings you cannot see it in the style property page but the deactivation will be recognized in the marking process.

9.5.1 Edit Styles

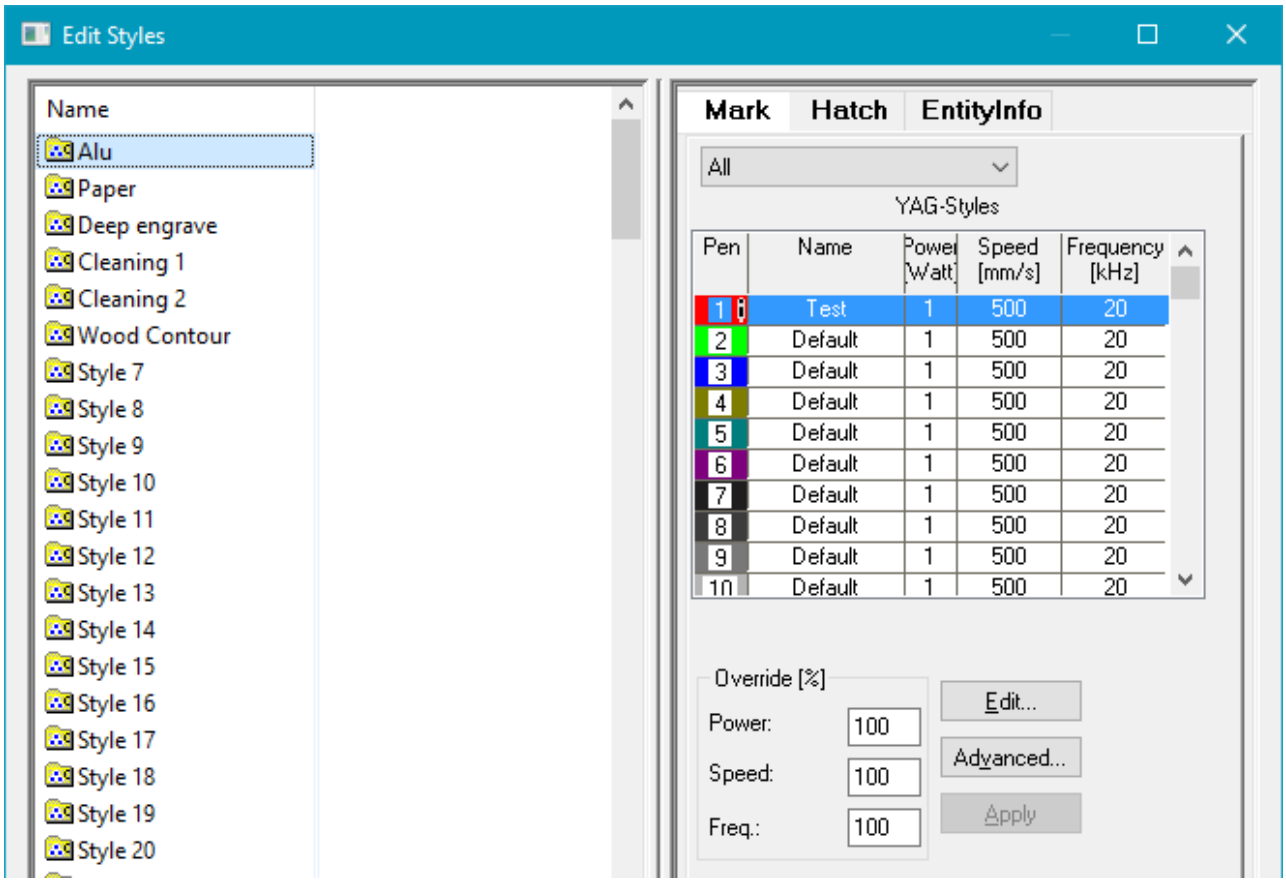


Figure 211: Edit Styles Dialog

Edit Style Dialog: The tabs of the Edit Style Dialog corresponds to the known property pages Mark, Hatch and EntityInfo.

10 Import-Export

This chapter describes how to import and export different file formats into SAMLight.

10.1 Import

Menu bar → File → Import opens a Dialog Box to import Graphic files and puts them into a new layer.

Available import formats are:

Extension	Description
*.ai	Adobe Illustrator (AI) is a vector graphics file format.
*.gbr	Gerber Format (GBR) is a vector graphics file format for printed circuit boards.
*.gif	Graphics Interchange Format (GIF) is a raster graphics file format.
*.job	GSI PC-Mark job file format containing vector graphics.
*.tif	Tagged Image File (TIF) is a raster graphics file format.
*.txt	Point Cloud Data is a ASCII format containing 3D vertices.

*.274x	RS-274X is an extended Gerber Format, see *.gbr.
*.bmp	Bitmap (BMP) is a raster graphics file format.
*.cmx	Corel Metafile Exchange (CMX) is a vector graphics file format.
*.cnc	CNC G-Code is a language to control CNC (Computer Numerical Control) machines.
*.dst	Tajima DST is an embroidery vector graphics file format.
*.dwg	DraWinG (DWG) is a binary CAD file format.
*.dxf	Drawing Exchange Format (DXF) is a CAD file format.
*.emf	Enhanced Metafile (EMF) is a raster graphics file format.
*.jpg	Joint Photographic Experts Group (JPEG) is a compressed image format.
*.mcl	Marker Control Language (MCL) is a GSI PC-Mark vector graphics file format.
*.pcx	Personal Computer Exchange (PCX) is a raster graphics file format.
*.plt	Hewlett-Packard Graphics Language (HPGL) Plotter File (PLT) is a language format for printing line drawings, specifically designed for 2D plotters.
*.png	Portable Network Graphics (PNG) is a raster graphics file format.
*.saf	SAF is a SCAPS archive.
*.svg	Scalable Vector Graphics (SVG) is a 2D vector graphics file format.
*.tga	TGA or TARGA is a raster graphics file format.
*.twain	TWAIN is a software protocol and applications programming interface between software and scanner.

Table 22: Available import formats

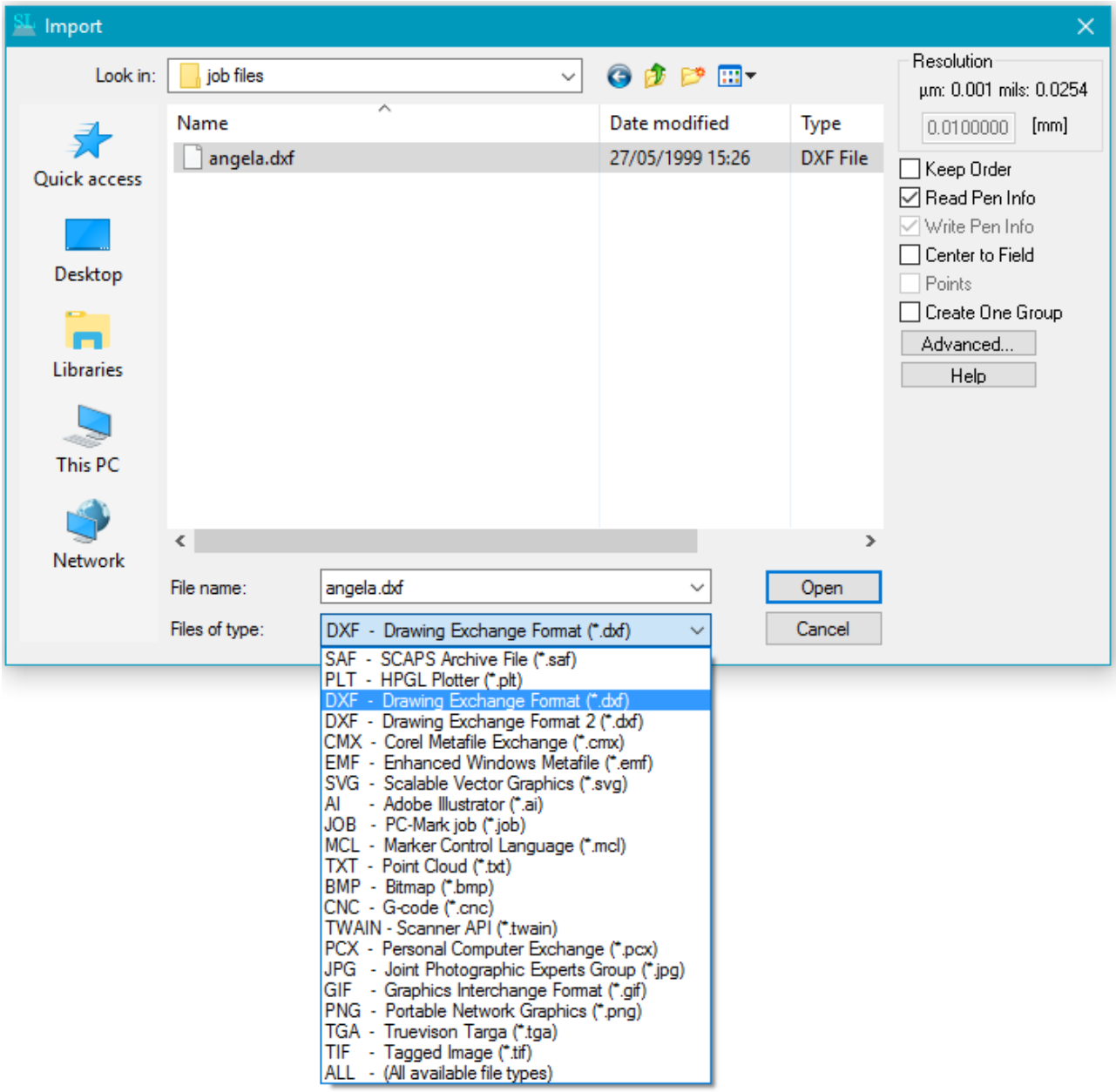


Figure 212: Import Dialog

Resolution: Defines how to transform the vectors from the file in case no units are given. The resulting unit is millimeter.

Keep Order: If this is selected all data is written in LineArrays to make sure that the exposure order is the same as the order inside the file. Less memory is needed for loading and saving a file. The disadvantage is that closed PolyLines can not be hatched after the import. The example below shows this behavior.

Read Pen Info: Enables the interpretation of the Pen information given in the File.

Advanced...: Allows to influence the import of the drawing more detailed, see chapter Import Advanced.

Example "Keep Order":

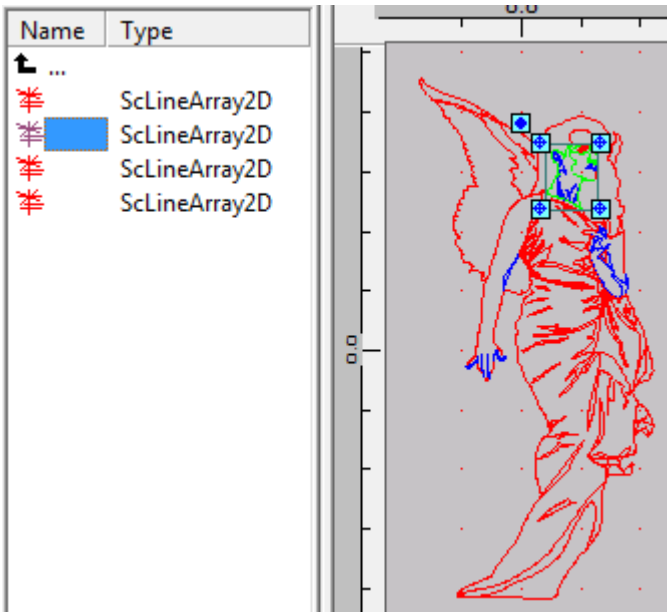


Figure 213: Example for Keep Order enabled

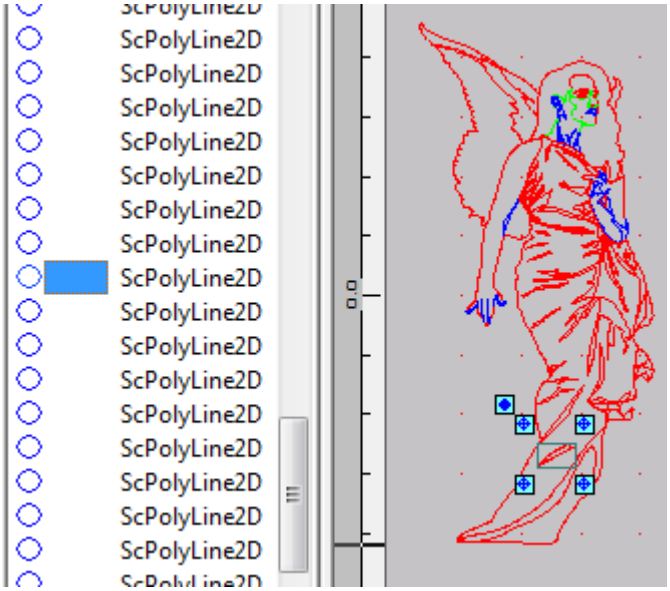


Figure 214: Example for Keep Order disabled

Activated options for available file formats are:

	Resolution	KeepOrder	ReadPenInfo	Points	Preview	PenColors
SAF	—	—	✓	—	✓	—
PLT	✓	✓	✓	✓	—	—
DXF	—	✓	✓	—	—	—
DXF v2	—	—	✓	—	—	✓
CMX	—	—	✓	—	—	✓
EMF	—	—	✓	—	—	✓
SVG	—	—	✓	—	—	✓
AI	—	✓	✓	—	—	—
JOB	—	—	✓	—	—	—
MCL	—	—	✓	—	—	—

BMP	—	—	—	—	—	—
PCX	—	—	—	—	—	—
TXT	—	—	—	—	—	—
TWAIN	—	—	—	—	—	—
JPG	—	—	—	—	—	—
GIF	—	—	—	—	—	—
PNG	—	—	—	—	—	—
TGA	—	—	—	—	—	—
TIF	—	—	—	—	—	—

Table 215: X: available, 0: not available

For Import of a SVG file, the Pen Colors are mapped as follows:

- Each RGB channel value of the import file and of each SAMLight pen is divided by 64 and truncated. Thus, all values are mapped into one of the following blocks (0-63, 64-127, 128-191, 192-255).
- Then, the first SAMLight pen which matches the SVG color blocks in R, G and B channel, is taken. If no corresponding pen is found, the next undefined pen is taken, starting with 21.

10.1.1 Point Cloud Files

A point cloud file needs to have following format.

```
x y
10.1 15.13
2 6
```

Importing this creates two points with coordinates x = 10.1, y = 15.13 and x = 2, y = 6.

10.1.2 Import Advanced

For the following file formats, the button *Advanced* (see figure 216) is active to influence the display of the drawing.

- AI
- CMX
- CNC
- DWG
- DXF Version1
- DXF Version2
- EMF
- GBR
- MCL
- PLT
- SVG
- 274x

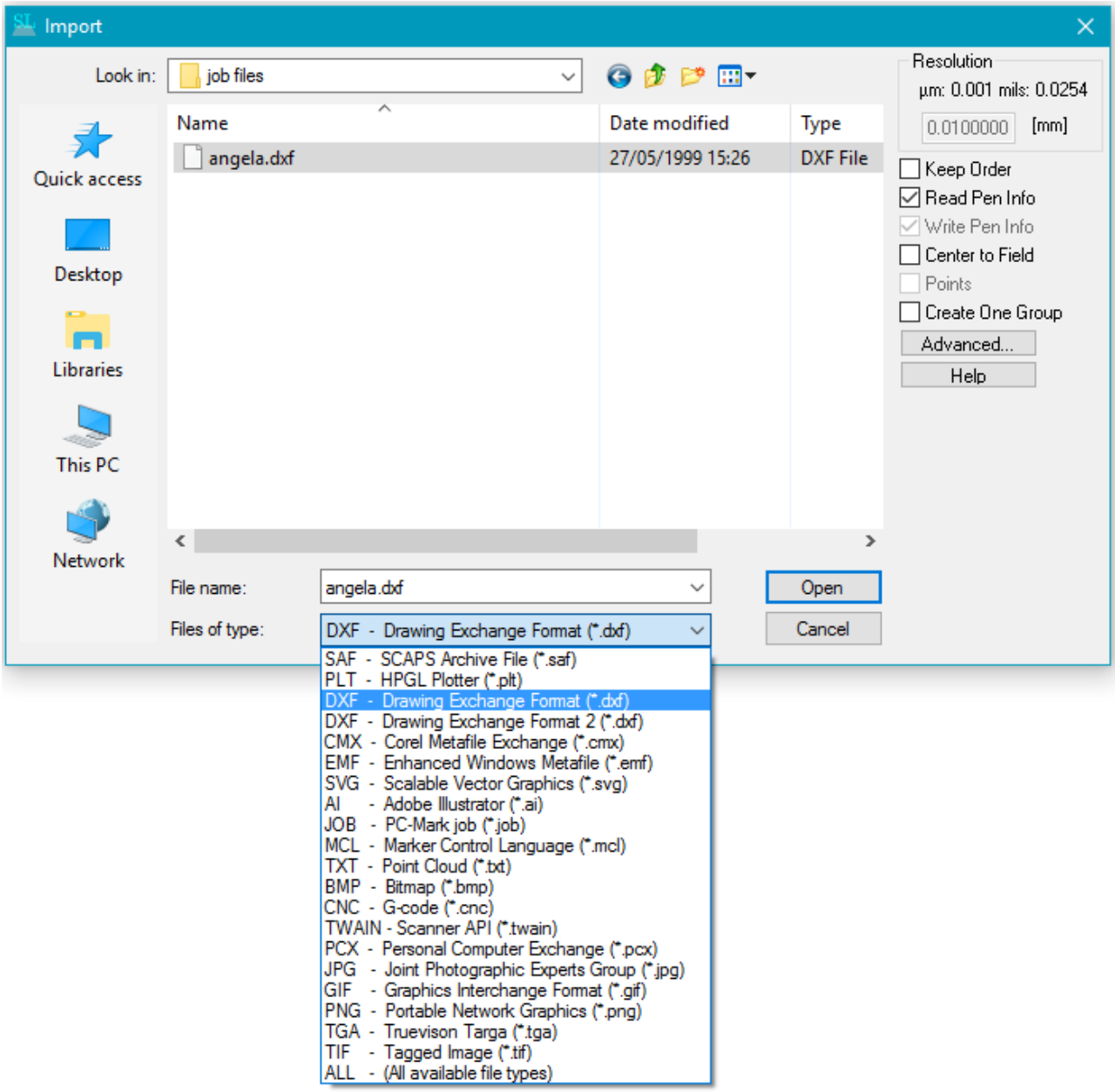


Figure 216: Import dialog

10.1.2.1 Advanced for several formats

For files of format CMX, DWG, DXF Version2, EMF, GBR, SVG and 274x, the following *Advanced Styles* dialog is given:

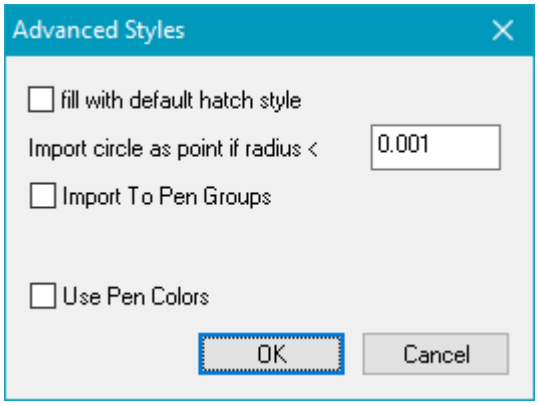


Figure 217: Advanced Styles for other files dialog

fill with default hatch style: Can be selected to hatch filled objects.

Import circle as point if radius < ...: A limit can be given up to which size a circle should be imported as a point.

Import to Pen Groups: If activated, all entities are sorted by pen and are imported to pen entities.

Use Pen Colors: Read the colors of pens from a SVG file. A prerequisite is, however, that the corresponding colors must also exist in the pen list of the Mark property page: The user can add or change any color via ViewProperties or in the View Settings Dialog. Please take off similar colors in the pen list to avoid confusion. By using this check box, the check box of "read pen info" in the Import Dialog should also be activated. This function is now only available for import from a SVG file. If the check box is not checked, the colors of SVG file would be ignored and the pens will be used in the order of the pen list on Mark property page.

10.1.2.2 Advanced for AI

For AI formats, the following *Advanced Styles* dialog is given:

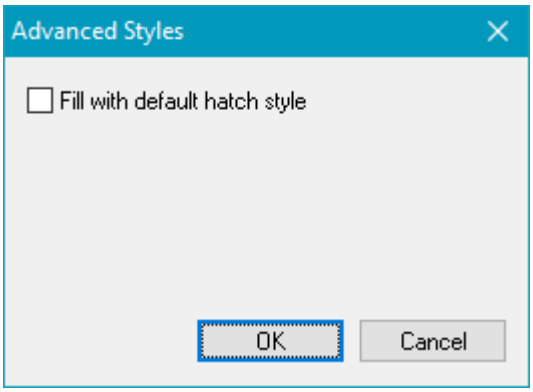


Figure 218: Advanced Styles for AI dialog

Fill with default hatch style: Is unchecked by default.When unchecked, no hatch is taken.

When checked, the default hatch pair is taken to fill all closed polylines. To set a default hatch pair, enable a hatch A and the corresponding hatch B, click on "Store all Pairs as default" and confirm the dialog "Save the state of all hatch 'Enable' checkboxes as well?" with yes.

10.1.2.3 Advanced for CNC

For CNC formats, the following *GCode Advanced Styles* dialog is given:

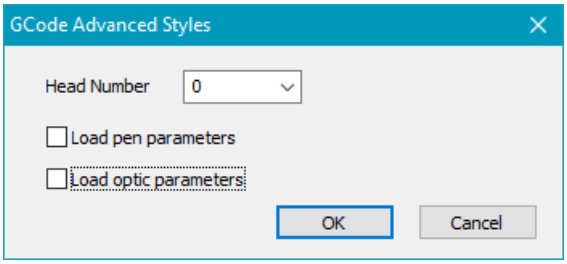


Figure 219: Advanced Styles for CNC dialog

Head Number: For a MultiCard system, specify the head to which the pen and optic parameters should be loaded.

Load pen parameters: Is unchecked by default. If activated, pen parameters (Pen settings) are imported.

Load optic parameters: Is unchecked by default. If activated, optic parameters (Field settings) and MOTF parameters are imported.

Load pen parameters: the values of the following G-Code commands can be loaded to the pen parameters:

$SC_LASERPOWER
$SC_FREQUENCY
$SC_JUMPSPEED
$SC_MARKSPEED
$SC_JUMPDELAY
$SC_MARKDELAY
$SC_POLYDELAY
$SC_LASERONDELAY
$SC_LASEROFFDELAY
$SC_STDBYPERIODE
$SC_STDBYPULSEWIDTH

Load optic parameters:

- the values of the following G-Code commands can be loaded to the optic parameters:

$SC_FIELDXMIN
$SC_FIELDYMAX
$SC_FIELDYMIN
$SC_FIELDXMAX
$SC_FIELDXGAIN
$SC_FIELDYGAIN
$SC_FIELDXOFFSET
$SC_FIELDYOFFSET
$SC_FIELDAXISSTATE

- the values of the following G-Code commands can be loaded to the MOTF parameters:

$SC_LASERPORT
$SC_MOFCHAN
$SC_MOFFACTOR0
$SC_MOFFACTOR1

10.1.2.4 Advanced for DXF Version1

For files of format DXF Version1 (DXF Files *.dxf), the following *Advanced Styles* dialog is given:

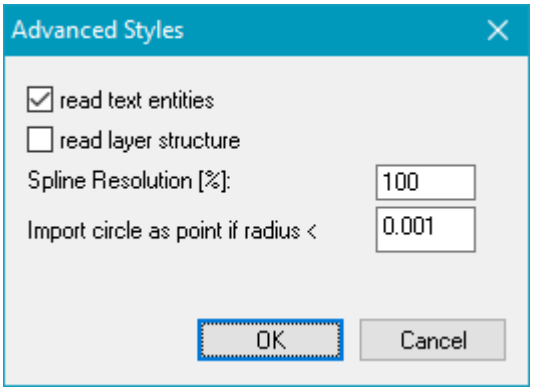


Figure 220: Advanced Styles for DXF dialog

read text entities: Is checked by default. If activated, text entities are imported.

read layer structure: Is checked by default. If different layers are existing within the imported dxf file, these layers are imported to different layers entities in View Level 2 of the entity list.

Spline Resolution [%]: Sets the resolution of splines, if imported. If the resolution is low, imported splines will look angular.

Import circle as point if radius < ...: A limit can be given up to which size a circle should be imported as a point.

10.1.2.5 Advanced for MCL

For files of format MCL, the following *Advanced Styles* dialog is given:

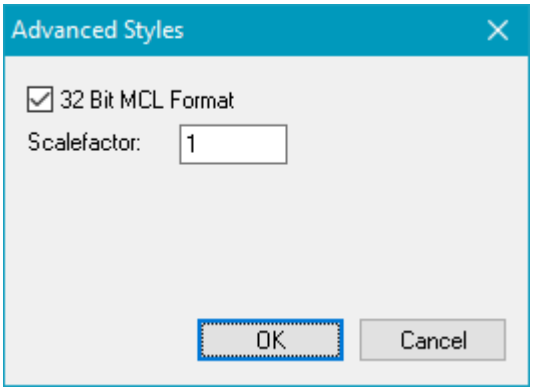


Figure 221: Advanced Styles for MCL dialog

32 Bit MCL Format: Is checked by default. If not activated, there is a limitation to a resolution of 16 bit ([-32768...32767]).

Scalefactor: Defines a factor for scaling of the MCL file. The default factor is 1.

10.1.2.6 Advanced for PLT

For PLT formats, the following *HPGL Advanced* dialog is given:

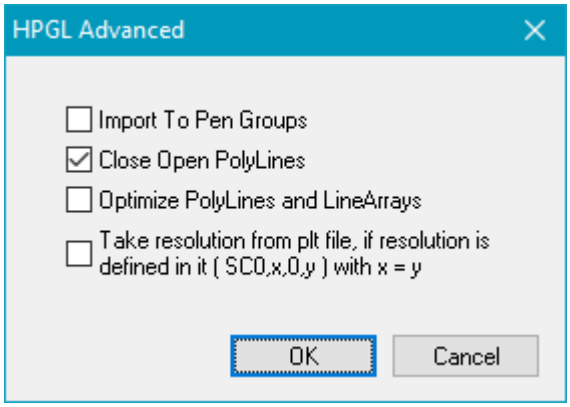


Figure 222: HPGL Advanced dialog

Import To Pen Groups: All objects painted with one pen are merged into one superior entity.

Close Open PolyLines: Is checked by default. Means that PolyLines are closed if there is a distance between start and endpoint which is smaller than the selected resolution.

10.1.3 Vector File Formats

DXF Files (Standard):

- No Text import. (To activate text import, use "read text entities".)
- Supports versions 10,12,13 and 14.

DXF Files Version 2:

- Supports all current variations of the DXF format (versions 2.6, 6, 9, 10, 11, 12, 13, 14, 15(2000), 18(2004)).
- Imports: Layer, line, arc, circle, ellipse, PolyLine, text (no reliable text position), vertex

DWG Files:

- DWG 12, 13, 14, 15 (DWG2000) and 18(DWG2004-2006)

Corel Presentation Exchange Files (CMX):

- Supported versions are 5, 6, 7, and 8.

Scalable Vector Graphics files:

- Imports: group, circle, line, polyline, Polygon, Polydraw, arc, (text).

AI Files:

- Supported Adobe versions are:
- Adobe Illustrator 5, 7, 8, 9, 11, 12
- AI8_CreatorVersion 9.0, 10.0, 11.0.0, 12.0.1, 13.0.0

10.2 Export

Menu bar → File → Export opens a Dialog Box to export selected entities into graphic files.

Available export formats are:

Extension	Description
*.saf	SAF is a SCAPS archive.
*.plt	Hewlett-Packard Graphics Language (HPGL) Plotter File (PLT) is a language format for printing line drawings, specifically designed for 2D plotters.
*.cnc	CNC G-Code is a language to control CNC (Computer Numerical Control) machines. Option Flash license is required for cnc export.

Table 23: Available export formats

11 Mark

Start: Opens the Mark Dialog to control the mark process.

MarkHatchesFirst: If this item is checked the hatches will be marked first.

MarkOnlySelected: If this item is checked, a mark process started by *Menu bar → Mark → Trigger* will only mark the selected entities.

Reset...:

ResetSequence: Resets/Restarts a mark sequence. Please refer to section Entity Info to get details on how one can define a mark sequence.

ResetSerialNumber: Resets all Serial Numbers of the job to their defined start values.

ResetCounter: Resets all Mark Statusbar information.

Sequence/SerialNumber: Executes ResetSequence and ResetSerialNumber.

Counter: The Counter menu handles all counter related functionality:

SetQuantities...: Opens a dialog where the quantities can be defined that is how many times a mark process is repeated.

Edit Counter...: Here a starting value for the counter can be set.

Increment:

Sequence: Increments the current sequence of the job. Please refer to section Entity Info to get details on how one can define a mark sequence.

SerialNumber: Increments all SerialNumbers of the job.

SerialNumber by...: Increments all SerialNumbers of the job by an arbitrary number.

Sequence/SerialNumber: Increments the sequence and serial numbers of the current job. Please refer to section Entity Info to get details on how one can define a mark sequence.

Decrement:

Sequence: Decrements the current sequence of the job. Please refer to section Entity Info to get details on how one can define a mark sequence.

SerialNumber: Decrements all SerialNumbers of the job.

SerialNumber by...: Decrements all SerialNumbers of the job by an arbitrary number.

Sequence/SerialNumber: Decrements the sequence and serial numbers of the current job. Please refer to section Entity Info to get details on how one can define a mark sequence.

Sequence/SerialNumber: Updates the sequence and serial numbers of the current job. Please refer to section Entity Info to get details on how one can define a mark sequence.

TimeInfo...: Shows the TimeInfo dialog below.

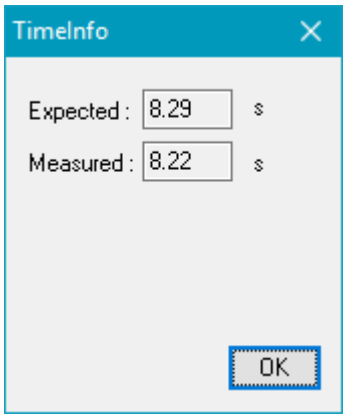


Figure 223: TimeInfo Dialog

Expected: Calculates the expected duration of the mark process in seconds depending on the job and settings information.

Measured: Displays the measured duration of the last mark process.

Preview: See chapter Mark Preview.

11.1 Mark Dialog

To open the Mark Dialog go to *Menu bar → Mark → Start*.

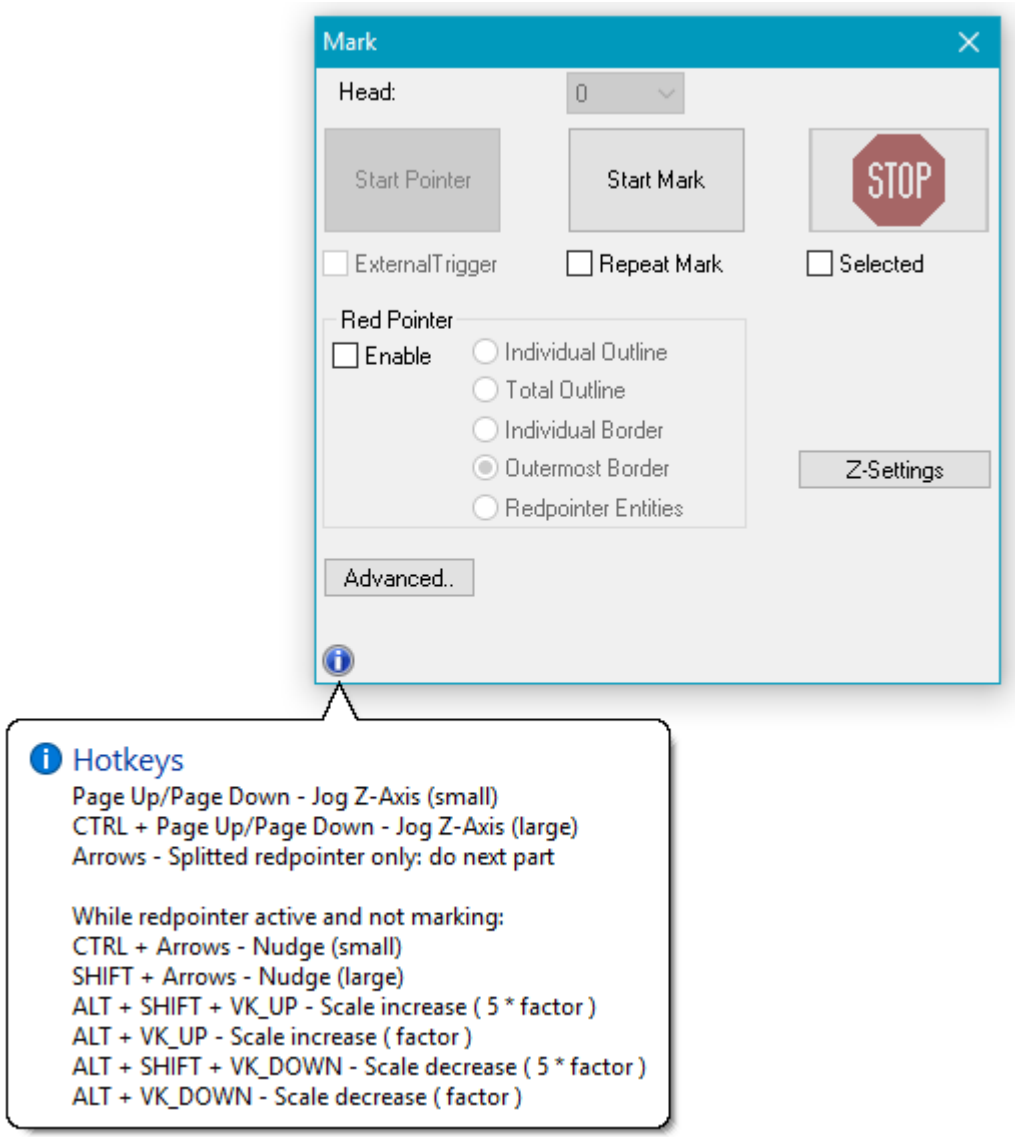


Figure 224: Mark Dialog

Head: This field is active if there is more than one scan head. In this case one of the available heads can be chosen to be used for the marking operation.

Start Mark (the yellow laser warning sign): Starts the mark process.

Stop (the red stop sign): Stops the mark process.

Each time a mark process is triggered (by clicking on Start Mark, by pressing F1 or by external trigger) all objects that have to be marked are sent down to the optic device and are exposured.

External Trigger: If checked the mark process can be started by an external trigger signal received by the optic device.

Repeat Mark: If checked the mark process will be repeated.

Selected: If checked only selected entities are considered in the mark process.

Redpointer: See chapter Redpointer.

Advanced: See chapter Redpointer.

Z-Settings: See chapter Deep Engraving.

Hotkeys: Some extra functionality to jog the z-axis or to nudge or scale entities. Controlling the Jog Z-Axis via (CTRL +) Page Up/ Page Down is only available for motion controller type 14.

11.1.1 Redpointer

Start Pointer: Starts the red pointer. This button is active only if the red pointer is enabled. For the red pointer *pen #255* is used (see chapter Mark Settings). While the redpointer is outlining, scale and nudge can be done with the short keys listed below.

- **Ctrl-ArrowKey:** Nudge
- **Shift-ArrowKey:** 5 * Nudge
- **Alt-ArrowUp/Down:** Scale
- **Shift-Alt-ArrowUp/Down:** 5 * Scale

Nudge Step and Scale Factor can be defined within Settings → System → View.

The StopKey in Settings → System → Short Keys should be set to Esc/Space.

Enable: Enables the red pointer.

Individual Outline: If checked each individual outline of the objects is drawn.

Total Outline: If checked the complete outline of all objects will be drawn by the red pointer, otherwise only the outline of one object will be drawn.

Individual Border: If checked, the red pointer draws the real geometry of the object.

Outermost Border: If checked, the red pointer draws the outermost individual borders of objects.

Redpointer Entities: If checked, only red pointer entity checked object(s) will be drawn. An object can be checked as redpointer entity in the entity list window.

Advanced: The following dialog opens after clicking on the Advanced button.

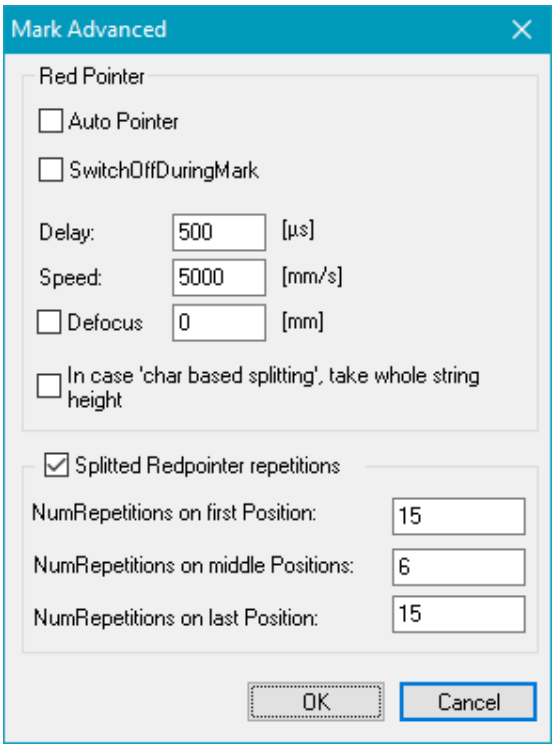


Figure 225: Mark Advanced Dialog

Auto Pointer: If enabled, the red pointer starts automatically after each marking showing the red pointer shape of the next marking sequence.

SwitchOffDuringMark: If enabled, the red pointer is switched off during mark.

Delay: Value determines delay of the red pointer.

Speed: Value determines speed of the red pointer.

Defocus: This activates a focus shift on the Z-Axis.

In case 'char based splitting' : If enabled, the height of the whole string is taken for outlining with the Red Pointer. See example in Fig.226:

Figure 226: Green for flag disabled, blue for flag activated.

Splitted Red Pointer repetitions: Can be checked only if entity is a split entity. You can set for first, middle and last split parts in between a red pointer repetition count. If unchecked (default), the red pointer runs endless on first split part. If the entity is not a split entity, this option is deactivated.

11.2 Trigger Dialog

Trigger: Sends all selected (the selected definition is set in Mark Dialog) entities to the optic device. The controller board lights up each time it receives a trigger impulse. If this menu item was selected a dialog window with a Stop-button appears as long as this trigger state is active and as long as the program waits for an external trigger to start marking of the current job. Depending on the type of scanner card beside the mark start trigger a mark stop signal can be sent to stop the execution of a marking process. In this case the window will be closed. The same happens when the Stop-button of that window is pressed. Here it is assumed that something happened that required immediate stopping of the marking process (e.g. because some kind of special maintenance was necessary). Such events have to be acknowledged by the user explicitly by selecting this menu item again when the marking process has to be restarted using the external trigger.

Figure 227: Mark Trigger Dialog

11.3 Mark Preview

Generates a preview of the actual job. Switch to the Preview Window by clicking *Menu bar → Window → Preview*. Then, to see the preview of your marking, click *Menu bar → Mark → Preview*. The preview window looks like the following:

11.3.1 Preview Window

The Preview Window shows the line outputs of one list execute. A line in this context is a straight line connecting two points. Depending on the line type the lines are drawn in different colours as shown in the screenshot below. The color of the selected line is white, the others lines are colored according to the OpticModuleProperties.

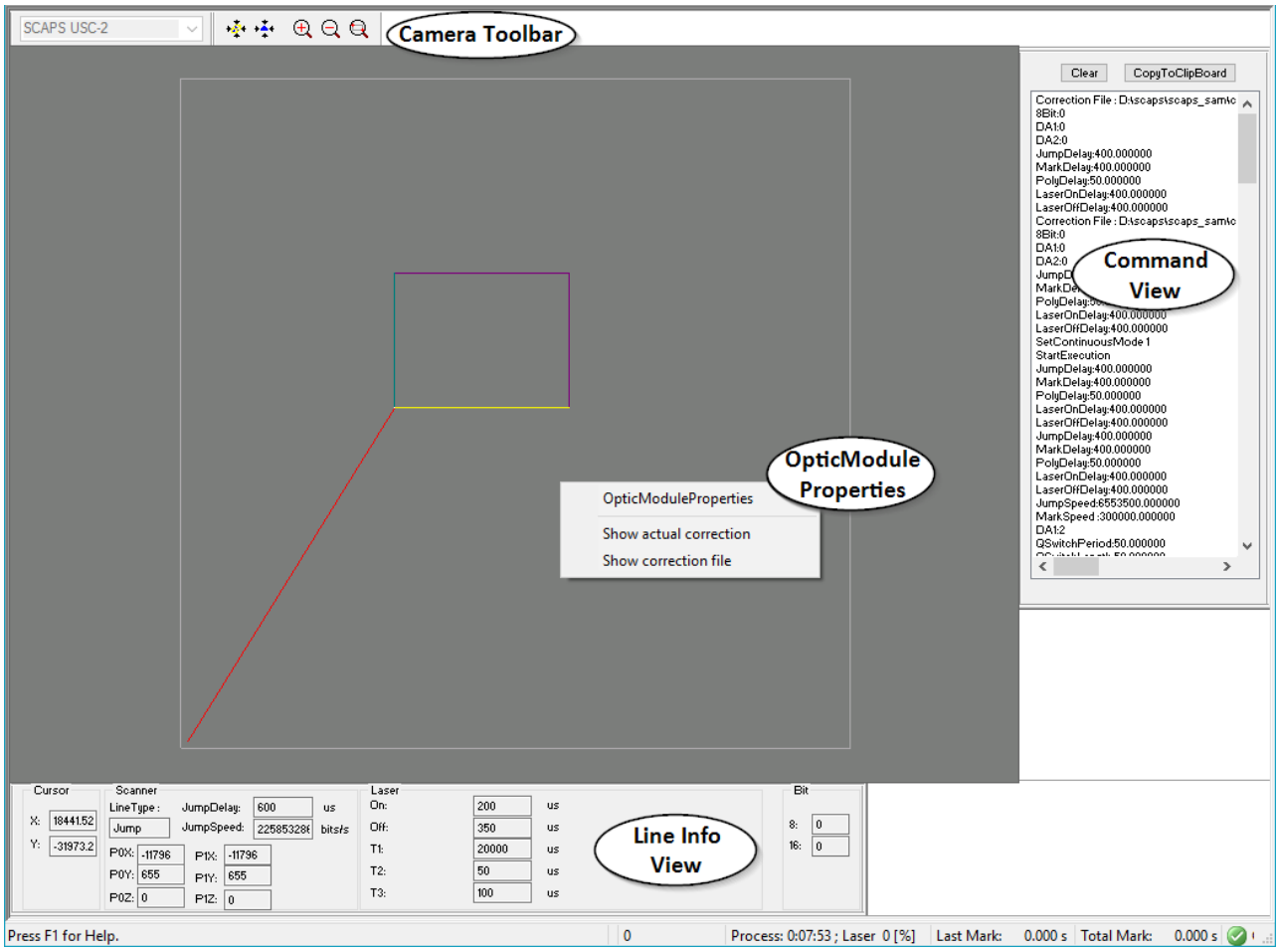


Figure 228: Mark Preview Window

Toolbars: The Preview Window has its own toolbar, because the standard toolbars belong to the Main Window and are not active while the Preview Window is in the foreground. The Camera Toolbar is just the *light* version of the Main Window's *Camera Toolbar*. So refer to chapter Camera Toolbar.

Views: Command View and Line Info View

OpticModule Properties: When clicking with the right mouse button in the Preview Window, the OpticModuleProperties will appear.

11.3.1.1 Command View

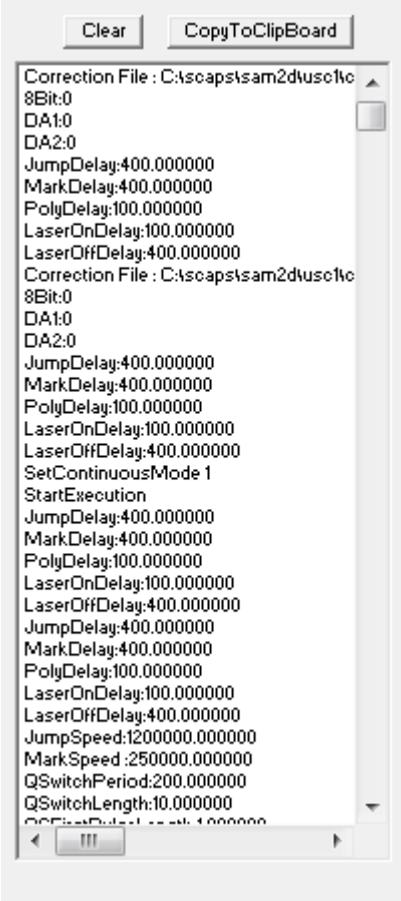


Figure 229: Command View

The Command View displays the first 1000 commands that are being sent to the preview window in a list box.

The displayed commands are compatible to the commands sent to the controller card in hardware mode.

If the button *CopyToClipBoard* is pressed, the content of the list box is copied to the clip board. This makes it easier to search for special commands.

This box is only enabled in simulation mode.

11.3.1.2 Line Info View

The Line Info View shows the line settings for the line to which the mouse pointer is moved. The specific line will be highlighted. This box is enabled in simulation mode only.

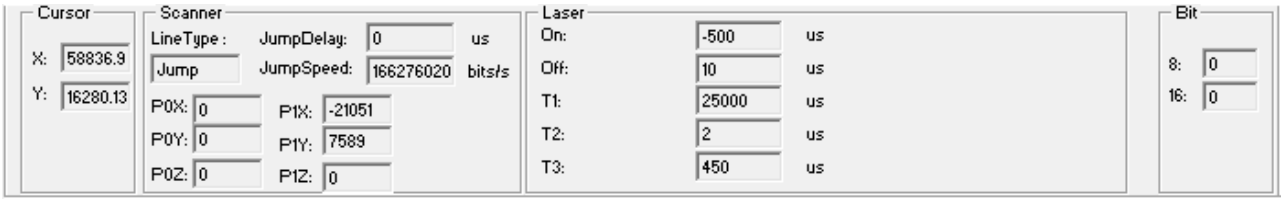


Figure 230: Line Info View

Cursor: Current cursor position in bits.

Line Type: Line type of the marked line like Jump, Mark, PolyA, etc.

P0X-P1Z: Start- and end-point co-ordinates of the selected line.

Delay: Scanner delay at the end of the line.

Speed: Speed of the scanner.

Laser: Laser on/off delay settings during execution of this line and additional the parameters T1 to T3.

11.3.1.3 OpticModuleProperties

If Multihead License is active: By Clicking with the right mouse button in the Preview window the following dialogs are being opened:

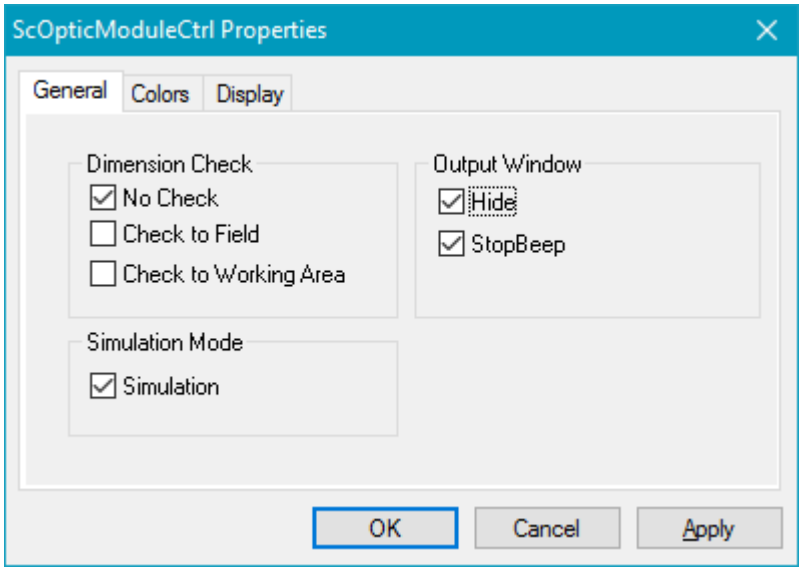


Figure 231: General of ScOpticModuleCtrl

Dimension Check: With these options, the user can check whether all the entities are inside of the Field or Working area before marking is started. If No Check is chosen, the marking will be executed until it gets out of the boundary.

Simulation Mode: This function is only designed for a user who has his own simulation application. Simulation Mode means there is no hardware output. The whole marking process will be simulated by software only, for example the time of Mark could be evaluated.

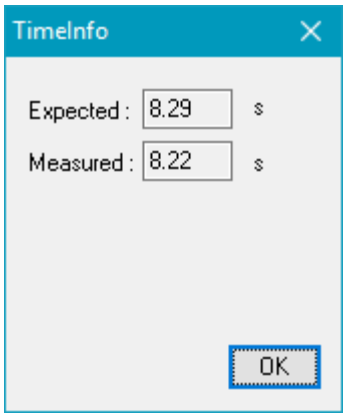


Figure 232: Time Info

The Simulation Mode will be automatically switched off by clicking on Mark in mark-dialogue of SAMLight.

Output Window: This is also only designed for a user who has his own Start-Mark-Application. The output window refers to the Mark trigger window. In SAMLight, it is default that the mark trigger window is being hidden. But in the user's own application, he could set it up here and decide if he wants a beep when he clicks stop.

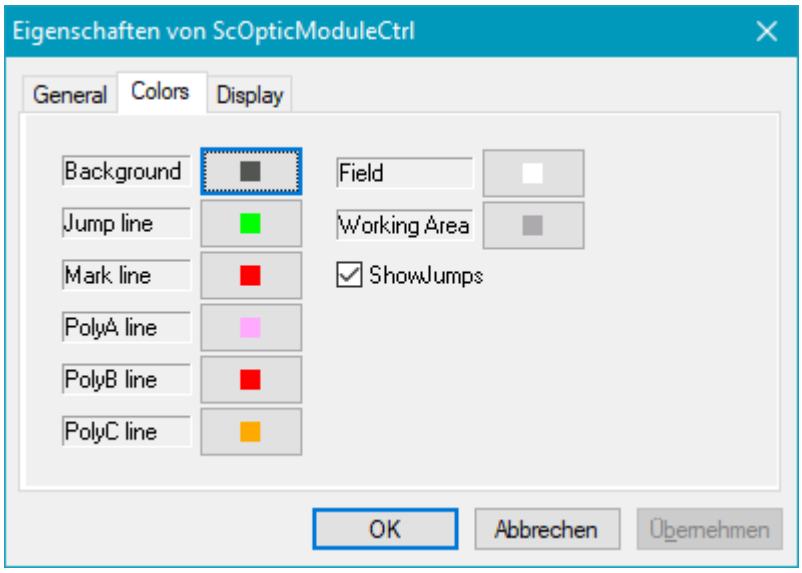


Figure 233: Colors

In the Window/1 Preview, click Mark → Preview, then the entities in Window/2 Main are plotted in different colors, which are defined in this dialog.

Default preview colors:

Line type	Color	RGB HEX	RGB DEC
Background		555555	085 085 085
Jump line		00ff00	000 255 000
Mark line		ff0000	255 000 000
PolyA line		ffaaff	255 170 255
PolyB line		ff0000	255 000 000
PolyC line		ffaa00	000 170 170
Field		ffffff	255 255 255
WorkingArea		aaaaaa	170 170 170

Table 24: Default preview colors

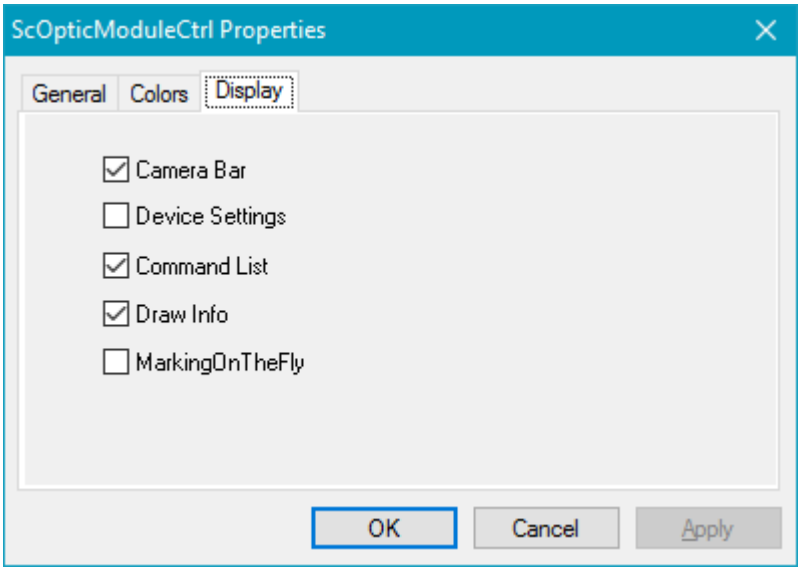


Figure 234: Display

With these check boxes, the user can activate or deactivate some toolbars or windows in the Preview Window.

11.4 SAMLight Job IO Selection

If the I/O mode is set to SAMLight Job IO Selection, loading of a job file can be invoked via input control. In this mode - according to the bit pattern of the external inputs and according to the related number that is created out of this pattern - a predefined job is loaded from disk and available for marking afterwards. If a job is loaded that way, it has to be marked afterwards using an external trigger. The appropriate option for the external trigger start has to be set before.
There are six hardware inputs, whereof the last 4 inputs define a number which identifies the job file:

Input	Function
OptoIn_0	Start Mark (Trigger)
OptoIn_1	Stop Mark (external stop)
Input selection pins	see table 25

Table 25: Input bits used to control the process.

The job files that have to be loaded depending on the input signals need to be saved in the folder <*SCAPS*> *\jobfiles* of the installation directory and they must have the following name structure:

jobname_nnnn.sjf

Here *jobname* stands for any freely definable characters and *nnnn* stands for the job number that is related to the input pin pattern (see table 27).

Card Type	Input selection pins
USC	OptoIn_2 ..._5
USC-2 Ext JobSelect	DigiIn_0 ..._7 up to 255 different job files
RTC	DigiIn_2 ..._5

Table 26: Input Bits used for different scanner control cards.

OptoIn_2	OptoIn_3	OptoIn_4	OptoIn_5	Job name
0	0	0	0	Empty Job
1	0	0	0	loadjob_00**01**.sjf
0	**1**	0	0	loadjob_00**02**.sjf
1	**1**	0	0	loadjob_00**03**.sjf
0	0	**1**	0	loadjob_00**04**.sjf
1	0	**1**	0	loadjob_00**05**.sjf
0	**1**	**1**	0	loadjob_00**06**.sjf
1	**1**	**1**	0	loadjob_00**07**.sjf
0	0	0	**1**	loadjob_00**08**.sjf
1	0	0	**1**	loadjob_00**09**.sjf

0	**1**	0	**1**	loadjob_00**10**.sjf
1	**1**	0	**1**	loadjob_00**11**.sjf
0	0	**1**	**1**	loadjob_00**12**.sjf
1	0	**1**	**1**	loadjob_00**13**.sjf
0	**1**	**1**	**1**	loadjob_00**14**.sjf
1	**1**	**1**	**1**	loadjob_00**15**.sjf

Table 27: Job selection table

Job number 0 is defined as an empty job. Job files with this number will be ignored.

To control the process the outputs are defined as follows (for USC cards):

Output	Function
OptoOut_0	*MarkingActive*
OptoOut_3	Software (Trigger) ready → *StartMark* can be set
OptoOut_4	New job was loaded successfully, signal will be reset with next external start (*StartMark*)

Table 28: Output bits used to control the process.

12 Splitting

This option allows you to split a job into pieces and then mark them part by part. That automatic split marking is useful e.g. for marking round objects or objects that are bigger than the working area. Jobs are always split completely as long as there are no entities marked as nonsplittable.

The Splitting Dialogs can be reached via the Extras Toolbar or by selecting the appropriate menu item.

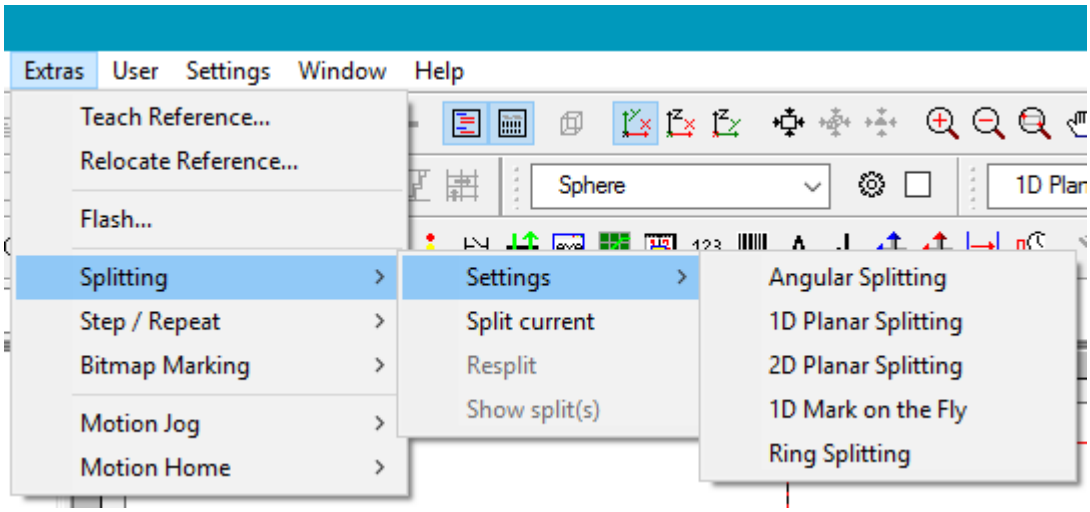


Figure 235: Splitting Dialogs

For rotational marking (*Angular Splitting* mode) can be done by performing the following steps:

- Enable external motion control to perform the automatic rotation of the round object that has to be marked.
- Define the diameter of the object.
- Define a rotational angle that describes the segment that should be marked at the same time.
- Define the motion controllers axis that has to be used for the rotation movements, that axis only defines the drive that has to be used for the motion, it does not influence the axis the job is split for.
- Turn on the splitting mode to enable the current job for rotational marking.

For planar marking (*Planar Splitting* modes) the following steps are necessary:

- Enable external motion control to perform the automatic movement of the flat object that has to be marked.
- Define the size of a split part that has to be marked at the same time.
- Define the motion controllers axis that has to be used for the planar movements. This axis only defines the drive that has to be used for the motion. It does not influence the axis the job is split for.
- Turn on the splitting mode to enable the current job for marking it part by part.

For *Mark on the Fly* of split parts following steps are necessary:

- You need a *Marking on The Fly* featured license of the SAM software.
- Define the size of one split part that has to be marked on the fly.
- Turn on the splitting mode to enable the current job for marking it part by part.

The Motion Control for the first two modes can be activated at *Menu Bar → Settings → System → Extras → Motion Control*. Additionally the motion control configuration file has to be configured for the used controller as described in motion control .

If all of the Rotation Axis or Motion Axis radio buttons are disabled there is a misconfiguration with the motion controller and the application will be unable to perform the correct rotational or planar movements.

To activate splitting mode go to *Extras → Splitting → Split current* or use the related icons in the Extras Toolbar . This mode can be turned off by clicking on this menu button again. Then the job is restored to non-split mode.

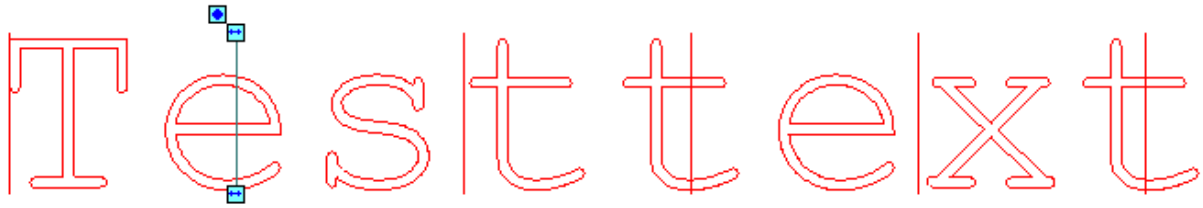


Figure 236: Example of a Split Job

When the splitting of a job was successful, its appearance in the *View 2D* is changed. Now there are several additional lines. They are the cutting edges that have been created conformly to the values *Total Diameter* and *Splitting Angle* (in angular mode) or *Splitting Size* (in planar mode). These splitting lines can be changed to optimize the result. Click on such a line and enter the object level 2 using the level toolbar. Now you can select a single splitting line. To move such a line click on the small blue box and then drag it in the desired direction.

When such a line is moved beyond one of the neighbour cutting lines it is automatically forced to a position back within its two neighbors. Interleaving the cuts that are represented by these lines is an invalid operation. If there are parts of the job that are located outside of the outer cutting lines after editing, they will not be marked.

After dragging a cutting line to a new position the split job is recalculated automatically. This operation may require some seconds depending on the complexity of the job. Marking such a job can be done like before. Here all the split parts are marked one by one until the complete job is done. Editing the cutting lines resets the current marking position so that the next marking operation starts with the first segment of that job.

To get the correct marking result it is very important that the axes of the motors correspond to the axes of the split job. Normally they should both form a right handed coordinate system.

If a motor movement is desired, that is independent of the splitting, it has to be defined as a nonsplittable entity, see chapter Entity List.

12.1 Angular Splitting

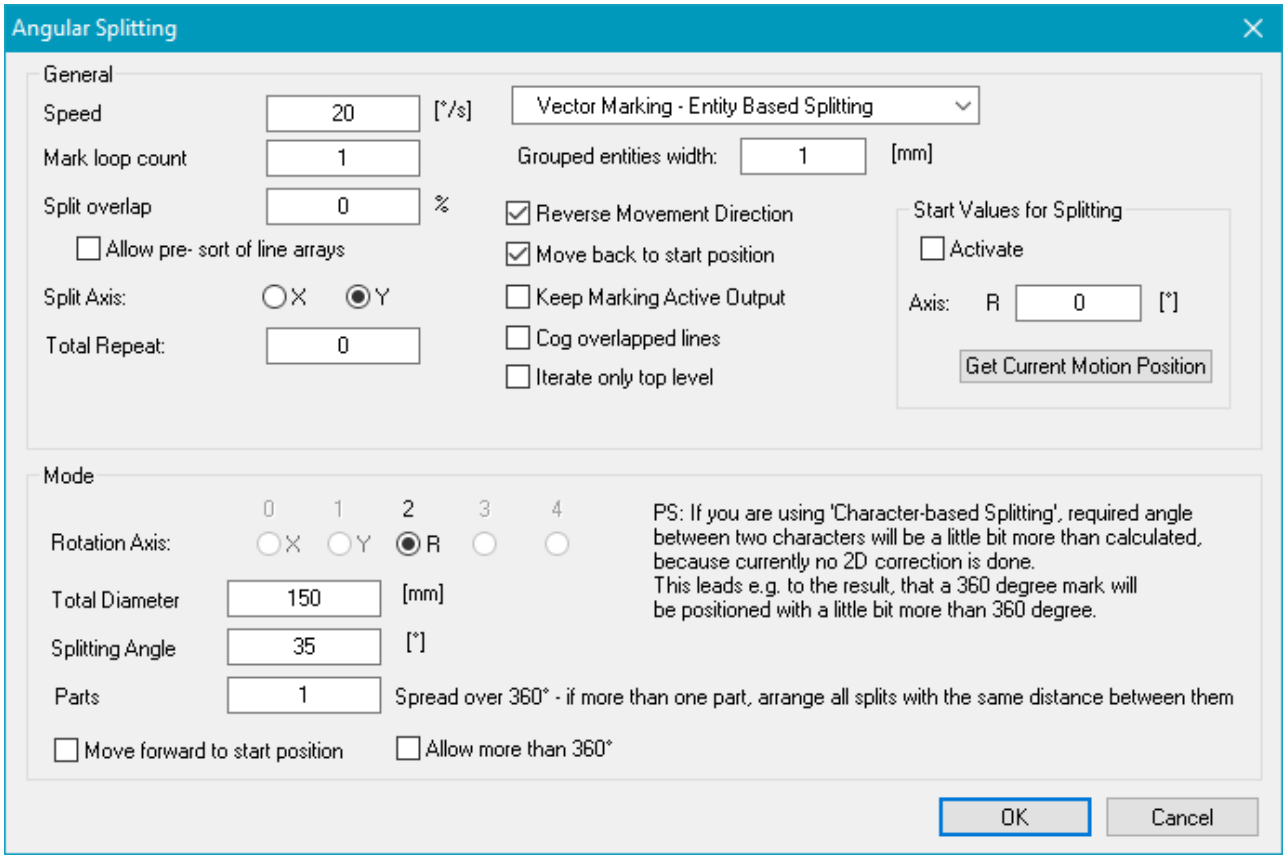


Figure 237: Angular Splitting Dialog

General:

Speed: This is a general parameter of planar and angular operation modes. It defines the speed used for the connected drive. The speed defined here overrides the speed defined in the Control submenu of the entity property page.

Mark Loop Count: Since the individual Mark Loop Count values of the entities are ignored in splitting mode, it is possible to define a job-global loop count here. The *Mark Loop Count* defines how often the marking of each split part is repeated before the motion device moves to the next split part.

Split overlap: With this option you can increase the area of a single split part.

Allow pre- sort of line arrays: The split algorithm converts all line objects into Line Arrays. By grouping them together the positioning of the splits is optimized and the process of the cutting is accelerated. An exampe is a DXF import which contains letters that are stored as simple polylines. But this can change the order of the split marking process.

Split Axis: The axis that defines the splitting direction for the geometry can be set here.

Total Repeat: After marking the last split part the scanner repeats marking all parts depending on this value.

Combo Box: Possible split options are:

Vector Marking - Fixed Split Size: Regardless on the positions of the entities the split size is always the same. Some entities may be cut.

Vector Marking - Entity Based Splitting: This Split algorithm avoids cutting polylines and group single entities. If the total size of two or more entities does not exceed the set entities width (see above) the entities will be centered and marked together. The dashed lines in the

View2D represents the center lines of the grouped entities, the external motion controller will stop at this positions for marking. With *Extras → Splitting → Show split(s)* you can preview the grouped entities.

Grouped entities width: With this value it is possible to group entities to decrease the marking time. The *Rotation Axis* dimension of the group outlines will not exceed this value. Entities with a greater *Rotation Axis* dimension than the grouped entities width will not be grouped with other entities.

Vector Marking - Character Based Splitting: If a job consists of exactly one text object that consists of one single line this option causes the splitting algorithm to place the splits between the single characters of that text object to avoid that geometry is cut.

Reverse Movement Direction: If the workpiece is being moved instead of the scan head, all movements go in the opposite direction.

Move back to start position: If this option is checked the scan head returns to the starting position after each marking. If unchecked, the scan head stops at the position where the marking finished and the next marking will start from there.

Keep Marking Active Output: In normal operation mode the *digital_output_0* is set to 1 every time a split is marked and it is set back to 0 during the movement operation. This behavior can be changed by enabling the check box *Keep Marking Active Output*. If it is active then the marking signal via *digital_output_0* would stay at 1 as long as the complete job including all splits is marked.

Cog overlapped lines: If the entity contains a hatch with many parallel lines the quality of the marking could be enhanced, if those lines are split on a cogged edge than a uniform one. If all lines are split on a uniform edge heating issues can occur and the marking quality may suffer.

Iterate only top level: Top level entities will not be split into smaller parts but marked as a whole.

Start Values for Splitting: Enable the *Activate* checkbox to define start values for the splitting. Define the starting position of the splitting routine with the numbers in *axis*. The values are absolute values. With the button *Get Current Motion Position* it is possible to set the values of the current motion controller as start values.

Mode: The *Angular Splitting* mode for marking on round objects. The object is rotated between two marking cycles to mark the whole circumference of the object. To use this mode at least one axis of the connected *motion controller* has to be configured for angular operation.

Rotation Axis: The axis of the connected drive that has to be used for rotating the workpiece. This option only chooses the axis and has no influence on the splitting direction.

Total Diameter: The diameter of the object that has to be marked. In case of a cylindrical object this would be the diameter of the base surface.

The circumference that results from the diameter of the base surface cannot be smaller than the total width of all job entities.

Splitting Angle: This angle in degrees describes the size of one split part.

Figure 238: Rotation Object

Parts ... Spread over 360°: If the total diameter and the resulting surface of the cylinder is big enough, more than one copy of the job can be marked on this surface. The different markings are distributed homogeneously over the total generated surface of the cylinder.

Allow more than 360°: This allows a continuous rotation with an angle greater than 360°.

12.2 1D Planar Splitting

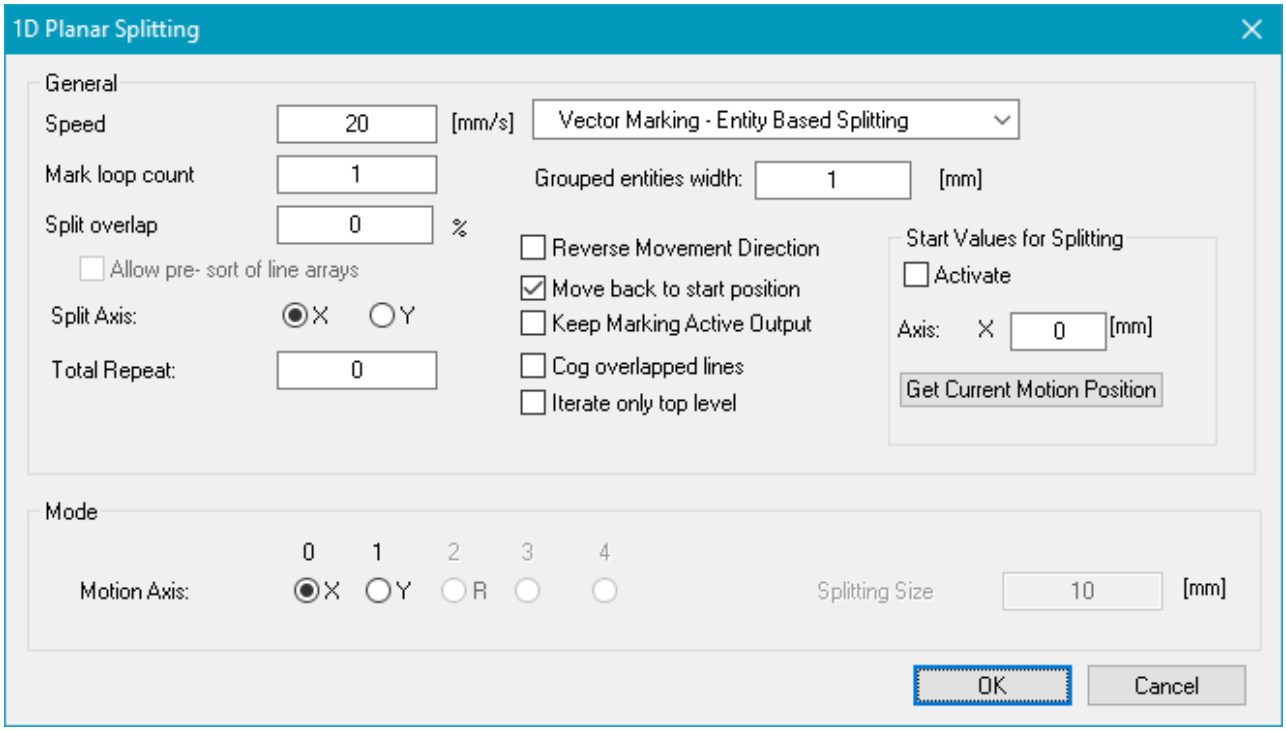


Figure 239: 1D Planar Splitting Dialog

General:

Speed: This is a general parameter of planar and angular operation modes. It defines the speed used for the connected drive. The speed defined here overrides the speed defined in the Control submenu of the entity property page.

Mark Loop Count: Since the individual Mark Loop Count values of the entities are ignored in splitting mode, it is possible to define a job-global loop count here. The *Mark Loop Count* defines how often the

marking of each split part is repeated before the motion device moves to the next split part.

Split overlap: With this option you can increase the area of a single split part.

Allow pre- sort of line arrays: The split algorithm converts all line objects into Line Arrays. By grouping them together the positioning of the splits is optimized and the process of the cutting is accelerated. An exampe is a DXF import which contains letters that are stored as simple polylines. But this can change the order of the split marking process.

Split Axis: The axis that defines the splitting direction can be set here.

Total Repeat: After marking the last split part the scanner repeats marking all parts depending on this value.

Combo Box: Possible split options are:

- **Vector Marking - Fixed Split Size:** Regardless on the positions of the entities the split size is always the same. Some entities may be cut.

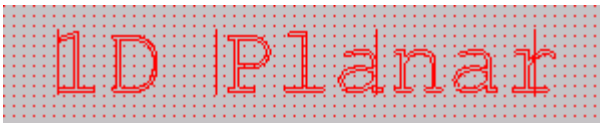


Figure 240: Fixed Split Size

- **Vector Marking - Entity Based Splitting:** This Split algorithm avoids cutting polylines and group single entities. If the total size of two or more entities does not exceed the set entities width (see above) the entities will be centered and marked together. The dashed lines in the View2D represents the center lines of the grouped entities, the external motion controller will stop at this positions for marking. With *Extras → Splitting → Show split(s)* you can preview the grouped entities.
  - **Grouped entities width:** With this value it is possible to group entites to decrease the marking time. The *Rotation Axis* dimension of the group outlines will not exceed this value. Entities with a greater *Rotation Axis* dimension than the grouped entites width will not be grouped with other entities.

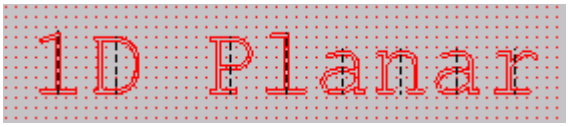


Figure 241: Entity Based Splits

- **Vector Marking - Character Based Splitting:** If a job consists of exactly one text object that consists of one single line this option causes the splitting algorithm to place the splits between the single characters of that text object to avoid that geometry is cut.

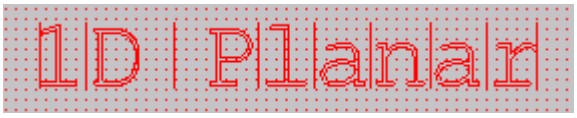


Figure 242: Character Based Splits

Reverse Movement Direction: If the workpiece is being moved instead of the scan head, all movements go in the opposite direction.

Move back to start position: If this option is checked the scan head returns to the starting position after each marking. If unchecked, the scan head stops at the position where the marking finished and the next marking will start from there.

Keep Marking Active Output: In normal operation mode the *digital_output_0* is set to 1 every time a split is marked and it is set back to 0 during the movement operation. This behavior can be changed by enabling the checkbox *Keep Marking Active Output*. If it is active then the marking signal via *digital_output_0* would stay at 1 as long as the complete job including all splits is marked.

Cog overlapped lines: If the entity contains a hatch with many parallel lines the quality of the marking could be enhanced, if those lines are split on a cogged edge than a uniform one. If all lines are split on a uniform edge heating issues can occur and the marking quality may suffer.

Iterate only top level: Top level entities will not be split into smaller parts but marked as a whole.

Do not recalculate split lines: Available only for Fixed Split Size.

Start Values for Splitting: Enable the *Activate* checkbox to define start values for the splitting. Define the starting position of the splitting routine with the numbers in *axis*. The values are absolute values. With the button *Get Current Motion Position* it is possible to set the values of the current motion controller as start values.

Mode: The *1D Planar Splitting* mode is for splitting the current job in one direction, either X or Y, related to the contents displayed in the *View 2D*. To use this mode at least one axis of the connected *motion controller* has to be configured for planar operation.

Motion Axis: Independent from the split axis this option can be used to choose a drive to perform the movement for the split parts.

Splitting Size: The size of one single split can be defined here. This size needs to be smaller than the working area in the same direction.

12.3 2D Planar Splitting

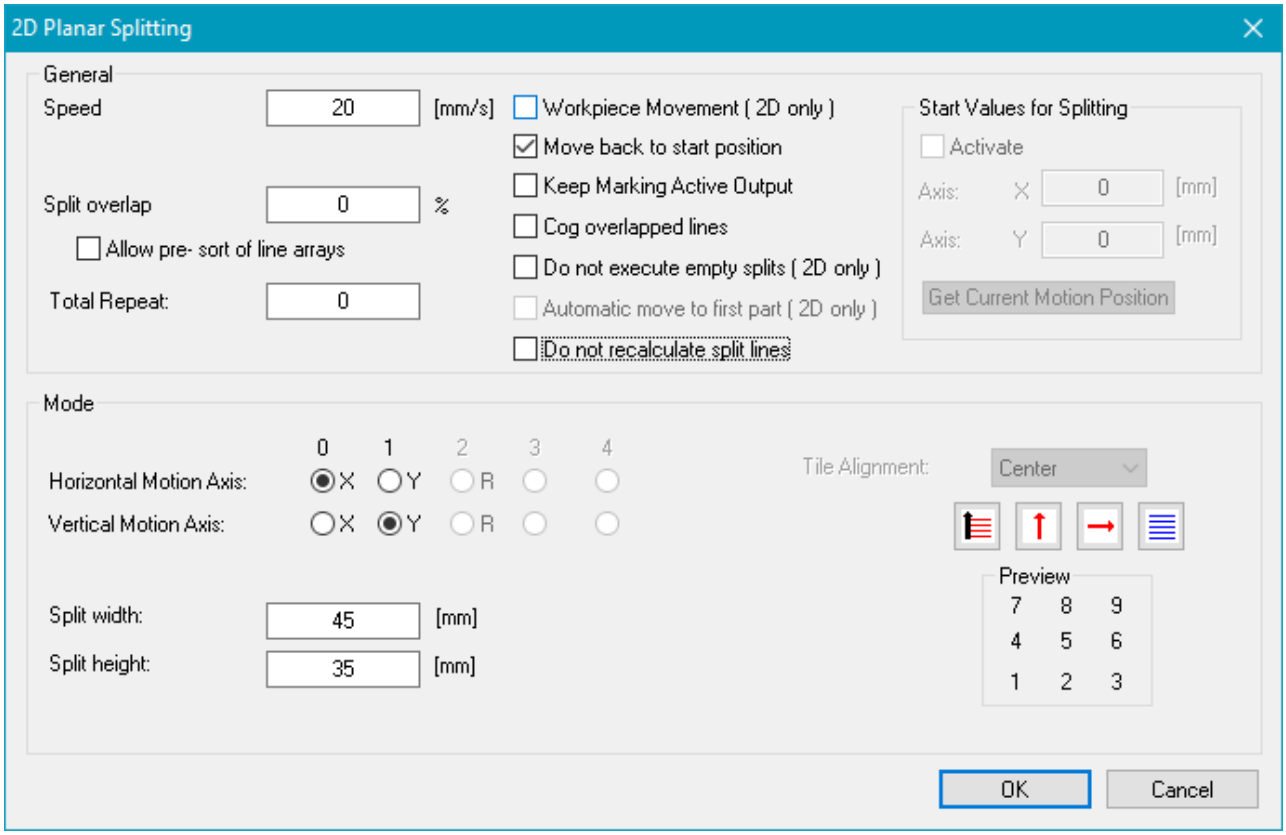


Figure 243: 2D Planar Splitting Dialog

General:

Speed: This is a general parameter of planar and angular operation modes. It defines the speed used for the connected drive. The speed defined here overrides the speed defined in the Control submenu of the entity property page.

Mark Loop Count: Since the individual Mark Loop Count values of the entities are ignored in splitting mode, it is possible to define a job-global loop count here. The *Mark Loop Count* defines how often the marking of each split part is repeated before the motion device moves to the next split part.

Split overlap: With this option you can increase the area of a single split part.

Allow pre- sort of line arrays: The split algorithm converts all line objects into Line Arrays. By grouping them together the positioning of the splits is optimized and the process of the cutting is accelerated. An exampe is a DXF import which contains letters that are stored as simple polylines. But this can change the order of the split marking process.

Split Axis: The axis that defines the splitting direction can be set here.

Total Repeat: After marking the last split part the scanner repeats marking all parts depending on this value.

Workpiece Movement (2D only): If the workpiece is moving instead of the scanhead, then all relative movements can to be inverted here.

Move back to start position: If this option is checked the scan head returns to the starting position after each marking. If unchecked, the scan head stops at the position where the marking finished and the next marking will start from there.

Keep Marking Active Output: In normal operation mode the *digital_output_0* is set to 1 every time a split is marked and it is set back to 0 during the movement operation. This behavior can be changed by enabling the checkbox *Keep Marking Active Output*. If it is active then the marking signal via *digital_output_0* would stay at 1 as long as the complete job including all splits is marked.

Cog overlapped lines: If the entity contains a hatch with many parallel lines the quality of the marking could be enhanced, if those lines are split on a cogged edge than a uniform one. If all lines are split on a uniform edge heating issues can occur and the marking quality may suffer.

Do not execute empty splits (2D only): If this option is enabled then the scanner will not move to empty split parts.

Automatic move to first part (2D only): Provided that the split is centered on the working area and that the motion axes are on position 0,0 then the user does not have to specify where the split should begin. The motion device moves automatically to the start position of the split.

Start Values for Splitting: Enable the *Activate* checkbox to define start values for the splitting. Define the starting position of the splitting routine with the numbers in *axis X* and *axis Y*. The values are absolute values. With the button *Get Current Motion Position* it is possible to set the values of the current motion controller as start values.

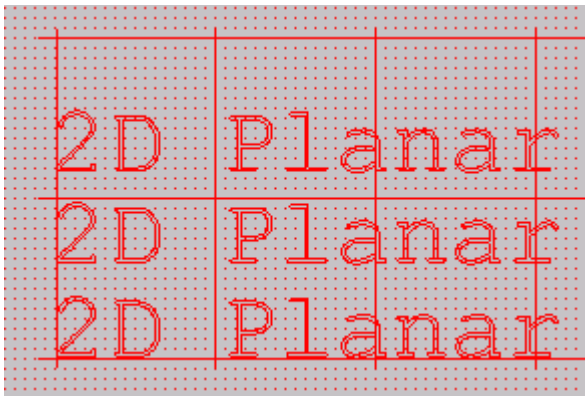


Figure 244: 2D Planar Split

Mode: The 2D mode is for splitting the current job in two directions. Therefore no split axis definition is necessary here. Using this splitting mode a XY-table should be used to place the working piece at the correct position. To use this mode at least two axes of the connected *motion controller* have to be configured for planar operation.

Horizontal Motion Axis: The motion axis that performs the horizontal movements can be defined here. You cannot use the same axis as *Vertical Axis*.

Vertical Motion Axis: The motion axis that performs the vertical movements can be defined here. You cannot use the same axis as *Horizontal Axis*.

Split width: You can define the horizontal split width here. The split width needs to be smaller than the working area in the horizontal direction and defines the resulting horizontal part size.

Split height: You can define the vertical split height here. The split height needs to be smaller than the working area in the vertical direction and defines the resulting vertical part size.

Tile Alignment: Here you can define where the marking on the scanner field is performed. For example: If *Center* is selected, the split parts will be marked on the center of the scanner field.

If the split parts are positioned at a border of the working area you can not use a gain factor greater than 1 in that borders direction, because this results in a "galvo coordinates out of range error" during marking.

Below of these options there are buttons that can be used to define the splitting and marking order for the split parts. A preview displays what order is currently chosen. The marking order is equal to the numbers in the preview area.

Figure 245: Order of split parts Marking

Here the splitting routine performs marking one column after another. The scan head always goes back to the top, the vertical movement direction during a split is always the same. This is a waste of time depending on the velocity of the motion device.

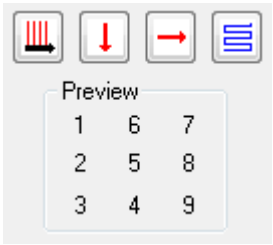


Figure 246: Optimised order of split parts marking

In this case the splitting routine performs marking of the columns as well but it will not move back to the vertical start position without marking. Instead it will perform the marking when the vertical motion device is heading back to the border. This saves time.

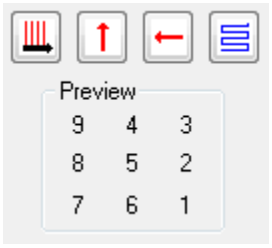


Figure 247: Defining the position of the split parts

With the two buttons in the middle the position of the first split part can be defined. With the left button the order of the remaining split parts can be defined.

12.4 1D Mark on the Fly (USC and RTC-5 only)

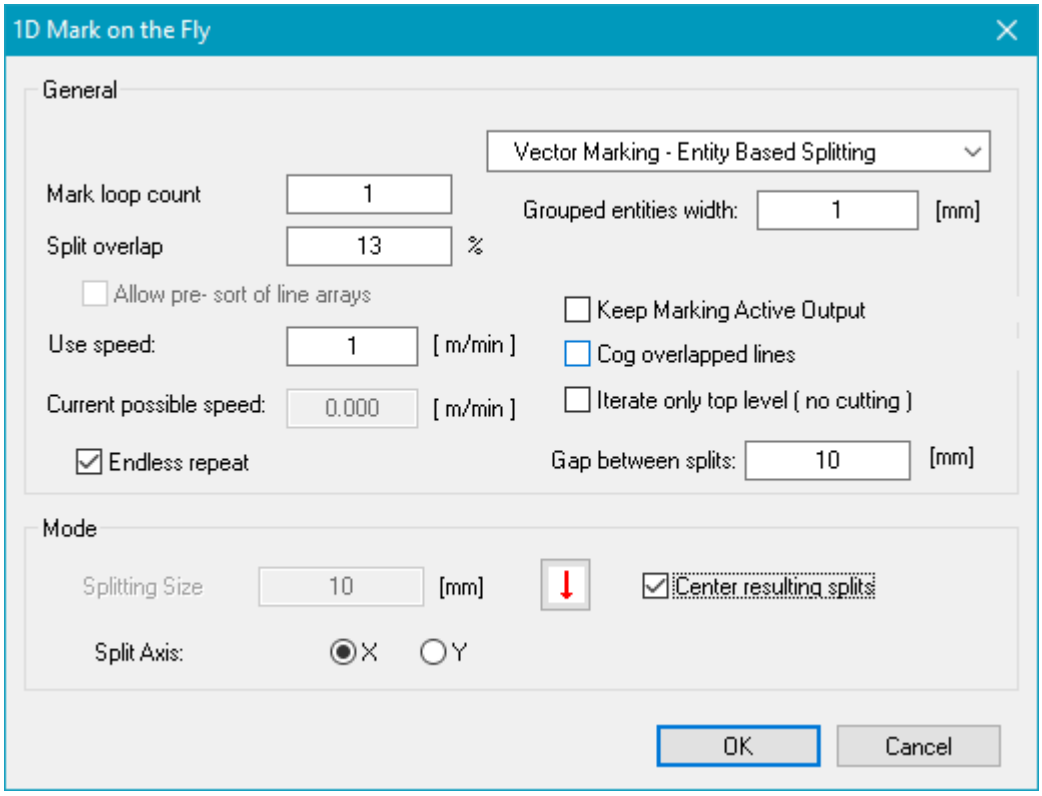


Figure 248: 1D Mark on the Fly Dialog

General:

Speed: This is a general parameter of planar and angular operation modes. It defines the speed used for the connected drive. The speed defined here overrides the speed defined in the Control submenu of the entity property page. This option is ignored for the *1D Mark on the Fly* mode, because here the application does not control the movement speed but reacts on it.

Mark Loop Count: Since the individual Mark Loop Count values of the entities are ignored in splitting mode, it is possible to define a job-global loop count here. The *Mark Loop Count* defines how often the marking of each split part is repeated before the motion device moves to the next split part.

Split overlap: With this option you can increase the area of a single split part.

Allow pre- sort of line arrays: The split algorithm converts all line objects into Line Arrays. By grouping them together the positioning of the splits is optimized and the process of the cutting is accelerated. An exampe is a DXF import which contains letters that are stored as simple polylines. But this can change the order of the split marking process.

Combo Box: Possible split options are:

Vector Marking - Fixed Split Size: Regardless on the positions of the entities the split size is always the same. Some entities may be cut.

Vector Marking - Entity Based Splitting: This Split algorithm avoids cutting polylines and group single entities. If the total size of two or more entities does not exceed the set entities width (see above) the entities will be centered and marked together. The dashed lines in the View2D represents the center lines of the grouped entities, the external motion controller will stop at this positions for marking. With *Extras → Splitting → Show split(s)* you can preview the grouped entities. Now bitmaps are supported in 'Entity based splitting'. Older 1D Marking On The Fly Splitting Dialog entry 'Character based splitting' is now included in the new option 'Entity based splitting'. 'Entity based splitting' allows use of more than one character string, but e.g.

character rotation is not supported in case 'Entity based splitting'.

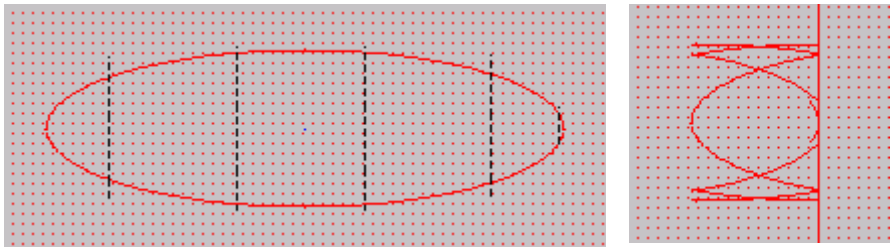


Figure 249: 1D Mark On Fly Split before and after activation of Mark → Trigger

Bitmap Marking: It is possible to mark a bitmap on the fly. The bitmap dimension may exceed the working area in the direction the belt is moving but not in the other direction.

Grouped entity width: The width of one splitting part. Not visible if *Iterate only top level* is checked.

Keep Marking Active Output: In normal operation mode the *digital_output_0* is set to 1 every time a split is marked and it is set back to 0 during the movement operation. This behavior can be changed by enabling the checkbox *Keep Marking Active Output*. If it is active then the marking signal via *digital_output_0* would stay at 1 as long as the complete job including all splits is marked.

Cog overlapped lines: If the entity contains a hatch with many parallel lines the quality of the marking could be enhanced, if those lines are split on a cogged edge than a uniform one. If all lines are split on a uniform edge heating issues can occur and the marking quality may suffer.

Iterate only top level: Top level entities will not be split into smaller parts but marked as a whole.

Endless Repeat: With this checkbox all split parts will be repeated endlessly.

Gap between splits: Here you can define a gap between two repetitions.

Mode: This field is only available for USC cards and only editable if *Marking on the Fly* is enabled within the settings. The *1D Mark on the Fly* mode is similar to the preceding one except the fact that the motion is not controlled by a connected *motion controller*. Here the object is moved continuously by an external drive like it is known for "on the fly" applications in general.

Split Axis: The axis that defines the splitting direction can be set here.

Splitting Size: The size of one single split can be defined here. This size needs to be smaller than the working area in the same direction.

Center resulting splits: This option allows you to center all split parts.

NOTE

If this splitting mode is active the marking via Start → Mark is disabled. This is because starting mark via the Mark Dialog would have a delay of about 200 ms and this would affect the MOTF marking result because marking would start to late. However the Redpointer can be activated to see where the MOTF marking will occur. Marking can only be started via Mark → Trigger.

*Split View: Extras → Splitting → Show split(s) → enable "Original positions/ Positions as in job to mark", in this way, a split job file with motions is generated automatically and can be saved in *.sjf file, which could be converted to *.unf file.*

12.5 Ring Splitting

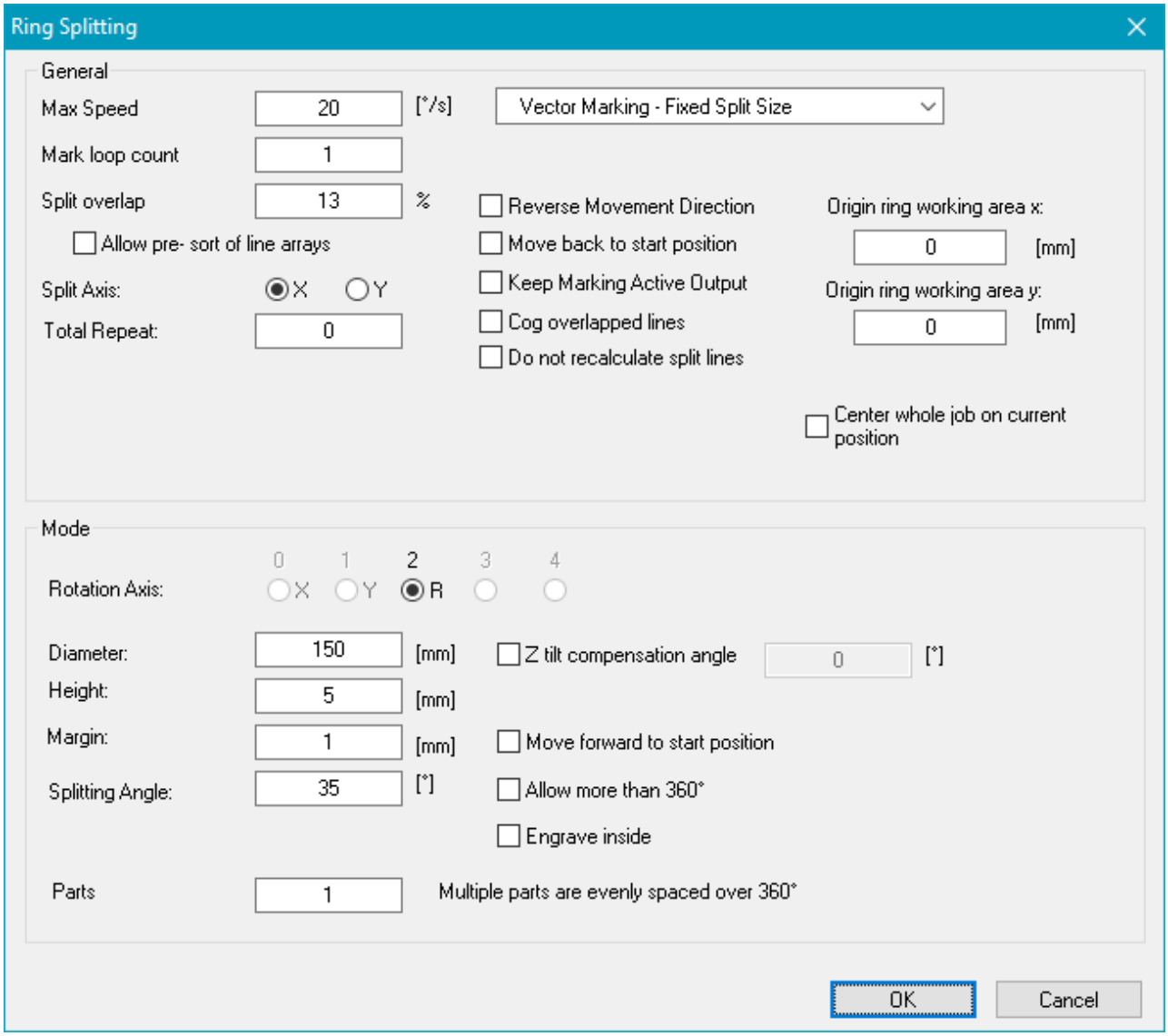


Figure 250: Ring Splitting Dialog

General:

Speed: This is a general parameter of planar and angular operation modes. It defines the speed used for the connected drive. The speed defined here overrides the speed defined in the Control submenu of the entity property page.

Mark Loop Count: Since the individual Mark Loop Count values of the entities are ignored in splitting mode, it is possible to define a job-global loop count here. The *Mark Loop Count* defines how often the marking of each split part is repeated before the motion device moves to the next split part.

Split overlap: With this option you can increase the area of a single split part.

Allow pre- sort of line arrays: The split algorithm converts all line objects into Line Arrays. By grouping them together the positioning of the splits is optimized and the process of the cutting is accelerated. An exampe is a DXF import which contains letters that are stored as simple polylines. But this can change the order of the split marking process.

Split Axis: The axis that defines the splitting direction can be set here.

Total Repeat: After marking the last split part the scanner repeats marking all parts depending on this value.

Combo Box: Possible split options are:

Vector Marking - Fixed Split Size: Regardless on the positions of the entities the split size is always the same. Some entities may be cut.

Vector Marking - Entity Based Splitting: This Split algorithm avoids cutting polylines and group single entities. If the total size of two or more entities does not exceed the set entities width (see above) the entities will be centered and marked together. The dashed lines in the View2D represents the center lines of the grouped entities, the external motion controller will stop at this positions for marking. With *Extras → Splitting → Show split(s)* you can preview the grouped entities.

Grouped entities width: With this value it is possible to group entites to decrease the marking time. The *Rotation Axis* dimension of the group outlines will not exceed this value. Entities with a greater *Rotation Axis* dimension than the grouped entites width will not be grouped with other entities.

Vector Marking - Character Based Splitting: If a job consists of exactly one text object that consists of one single line this option causes the splitting algorithm to place the splits between the single characters of that text object to avoid that geometry is cut.

Bitmap Marking: This option allows to mark a bitmap without the need to split the bitmap in parts manually. Between every line of the bitmap the motor moves.

Keep Bitmap Size: If that option is set the bitmap is not scaled in order to get a size that results in the specified marking angle, here the source bitmap is left untouched.

Reverse Movement Direction: If the workpiece is being moved instead of the scan head, all movements go in the opposite direction.

Move back to start position: If this option is checked the scan head returns to the starting position after each marking. If unchecked, the scan head stops at the position where the marking finished and the next marking will start from there.

Keep Marking Active Output: In normal operation mode the *digital_output_0* is set to 1 every time a split is marked and it is set back to 0 during the movement operation. This behavior can be changed by enabling the checkbox *Keep Marking Active Output*. If it is active then the marking signal via *digital_output_0* would stay at 1 as long as the complete job including all splits is marked.

Cog overlapped lines: If the entity contains a hatch with many parallel lines the quality of the marking could be enhanced, if those lines are split on a cogged edge than a uniform one. If all lines are split on a uniform edge heating issues can occur and the marking quality may suffer.

Iterate only top level: Top level entities will not be split into smaller parts but marked as a whole.

Origin ring working area x: Define a starting offset in X-direction.

Origin ring working area y: Define a starting offset in Y-direction.

Mode: The *Ring Splitting* mode is for splitting the current job on a ring.

Rotation Axis: Independent from the split axis this option can be used to choose a drive to perform the rotation for the split parts.

Diameter: Diameter of the ring.

Height: Transversal dimension of the ring.

Margin: Shows an optical helping line that reduces the height. No influence on marking output.

Splitting Angle: This angle in degrees describes the size of one split part.

Parts...Spread over 360°: The whole splitting job is repeated on the whole object in equidistant pieces so that the marking result is homogeneously distributed on the whole object.

Z tilt compensation angle: This feature is to be used with Ring splitting tilt compensation in Global settings → Extras.

Move forward to start position: When finished with marking and moving to start position this defines how the start position is reached again.

Allow more than 360°: This allows a continuous rotation with an angle greater than 360°.

Engrave inside: Engrave on the inside of the ring.

13 Option MOTF

This feature "Marking On The Fly" (MOTF) is for marking moving targets on a product line. General information on a (MOTF) setup is given here.
In sub-chapters, further detail is given on how to work with encoder signals or without them in simulation mode, on the card-specific hardware setup, on how to calibrate a MOTF system and on the possibility of doing endless MOTF or rotational MOTF. Finally, some examples can be found at the end.

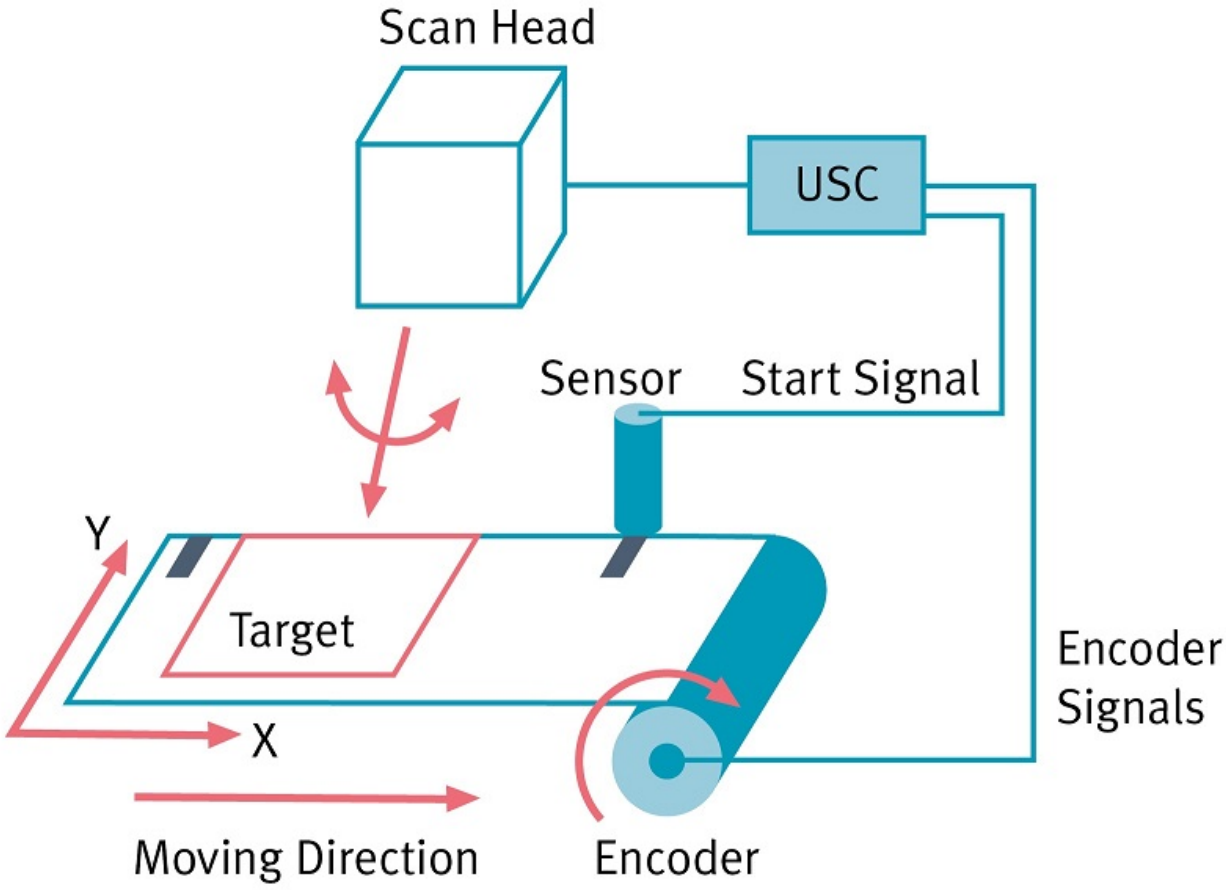


Figure 251: A typical MarkingOnTheFly (MOTF) setup

The target piece which has to be marked is placed on a moving conveyor belt that surpasses the scan head at a specific position. The scanner has to know when to start marking and how fast the target is moving. Latter is done by the encoder: The movement of the target is converted by the encoder. For a specific distance of the target (in the above example along the X-axis), the encoder gives a specific amount of counts. The information of this conversion is given by the multiplier. Movement is possible in different directions. The distance information from the encoder is sent to the scanner card which appropriately corrects the marking vector such that it fits to the moving target. The starting signal is sent by the trigger sensor: The sensor converts a specific optical label on the conveyor belt into an electrical trigger signal.

The MOTF_CH0 (on the 37-pin connector) and MOTF_CH1 (on the 40-pin connector) signals are filtered by a digital filter unit. For information on the cut-off frequency, see the corresponding hardware manual. After the decoding, a counter counts the incoming count pulses and is incremented or decremented according to signal (see encoder signals for further details). In some application the belt movement direction and speed remains constant. In this case the counts can be generated by an internal simulation generator eliminating the need for an encoder.

In order to calibrate the counter according to the scanner field units [typically in bits, mm or inch], the counter value is multiplied by a user definable signed constant. The resulting MOTF compensation is added to the marking information to form the final signals for scan head control.

13.1 Encoder Signals

The flow chart in figure 252 shows the general MOTF hardware setup with encoder signals.

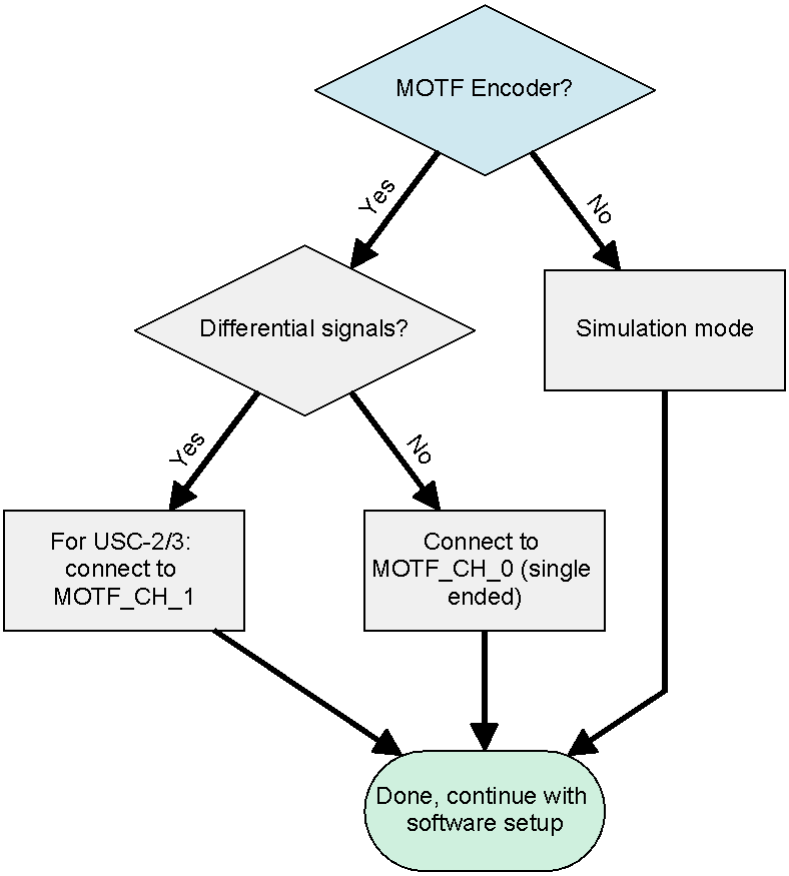


Figure 252: Flow Chart MOTF hardware setup

Verify the GND connection between the USC/RTC card and the encoder to adjust the level. To avoid noisy signals use short cables and check the power supply of the encoder.

The USC cards are designed to handle two 90° shifted encoder signals (track A and track B) delivered by standard commercial encoders. The USC decoder interprets each transition, whether the count impulse is on track A or track B of the respective channel. The interpreted belt direction depends on the phase shift between both tracks.

Phase shift B after A:

If the phase shift between track A and track B is + 90° (refer to figure 253, upper part), the belt is interpreted as moving in the default direction (let's call it forward). In this case, the encoder counts positive which is necessary for advanced MOTF features (like ScMotfOffset). Each falling or rising edge of track A or track B will increment the encoder counter by + 1. That means each track period T will lead to 4 encoder counts in total.

Phase shift B before A:

If the phase shift between track A and track B is - 90° (refer to figure 253, lower part), the belt is interpreted as moving in the non-default direction (let's call it backward). In this case, each the encoder counts negative. Each falling or rising edge of track A or track B will increment the encoder counter by - 1. That means each track period T will lead to - 4 encoder counts in total.

To change the counting direction, the signals of track A and track B can be swapped either by changing the

wiring of the tracks or by using Swap A/B in the software. The direction of MOTF compensation may need to be adjusted as well after changing the direction of the encoder counter. Please refer to the corresponding hardware specific sub-chapters for further information.

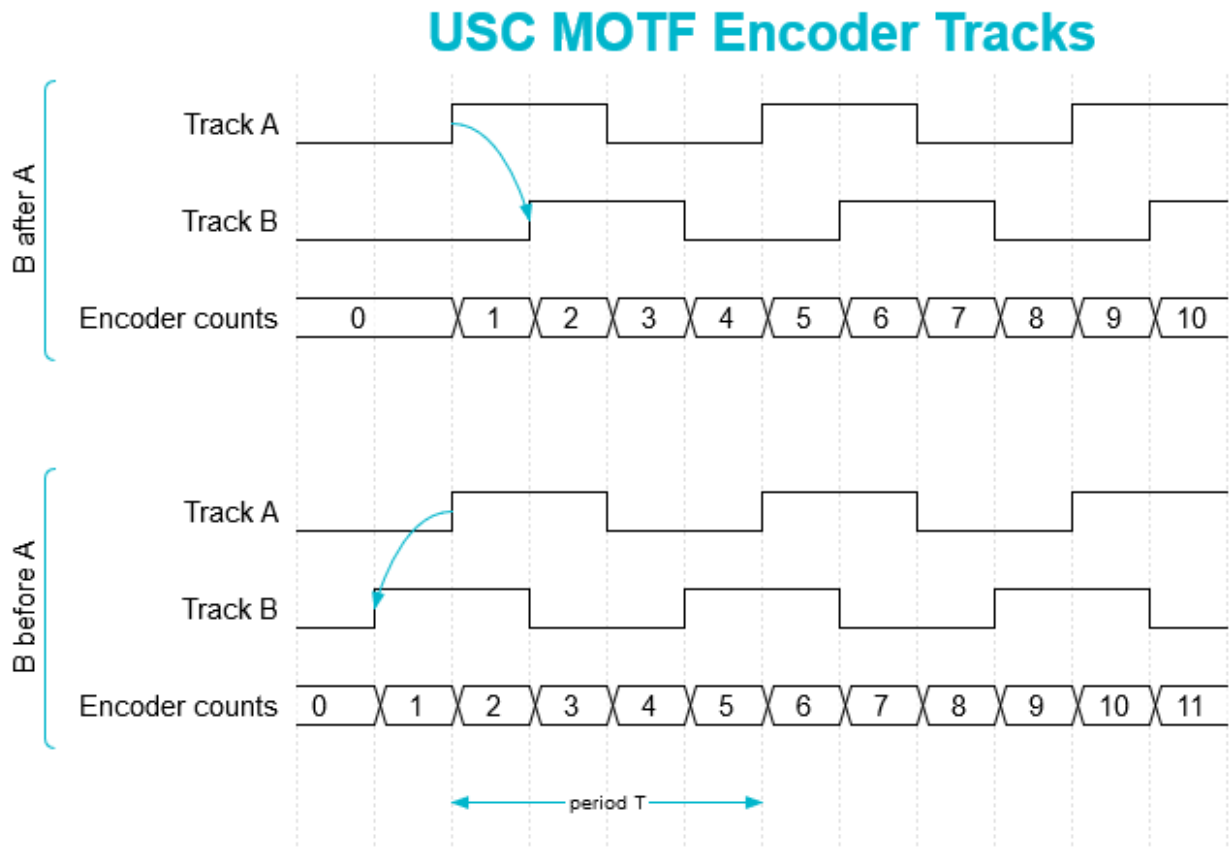


Figure 253: Encoder signals (single ended). Note that each period T consists of 4 encoder counts.

Using encoder signals, it is necessary to specify the conversion between the counts of the encoder and the corresponding distance on the work piece. This conversion is given by the multiplier in combination with the optic settings.

NOTE

The amplitude of the encoder signals should be in the range of 2.5 V to 5.0 V. The signals should be as low-noisy as possible, since noise could be interpreted as additional pulses. Possibly, you can use a differential encoder and MOTF_CH1 (40-pin connector).

*In case of problems, verify the switching threshold levels (*cut-off frequency*) of the encoder signals with an oscilloscope and/or use differential signals.*

13.2 MOTF Multiplier

The MOTF multiplier is the central factor to translate encoder counts into mm or bits. To calibrate this factor very well is crucial for correct vector compensation and encoder based distance measurements which are used by ScMotfOffset objects for example. The calibration routine is card specific and described in the following chapters. However, some general remarks and notes are given here.

There are two ways to determine the correct factor for the MOTF multiplier:

1. By theory:
   - In the manual of the encoder, the factor increments per rotation is defined as "I" in the unit counts/360°.
   - The circumference of the reel that is attached to the encoder is known as "U". The unit is mm/360°.

- Then, the factor F can be calculated as the ratio of (4 times I) and U:

$$F = \frac{4 \cdot I}{U} = \frac{4 \cdot 2000 \frac{counts}{360°}}{2\pi \cdot 50 \frac{mm}{360°}} = 25.46 \frac{counts}{mm}$$

- Card specific calculations can be found in the following chapters (Hardware setup).

A possible slack between the reel and the conveyor is not considered. However the calculation gives a good approximation.

The factor I must be multiplied by 4 because each rising and falling edge of both tracks counts. Please have a look at figure 253 for further details on the MOTF signals.

2. By experiment:
   - See USC-1 Specific Calibration
   - See USC-2/3 Specific Calibration

13.3 Simulation Mode

If the belt is moving with a constant speed, the simulation mode can be used instead of connecting an encoder. In simulation mode, the constant speed can be defined in meters per minute (m/min) or in millimeters per seconds (mm/s). All MOTF compensations and ScMotfOffset objects are based on this defined speed.

ScMotfOffset control objects do not work in MOTF simulation mode for USC-1 cards.

13.4 Hardware setup

In this chapter, information on how to set up a Marking On The Fly system is given for the following cards:

- USC-1
- USC-2/3
- RTC

For each card, a table gives an overview on the hardware connections available for marking on the fly. Information on how to calibrate a MOTF system are given in chapter Calibration.

13.4.1 Card Specific: USC-1

For the USC-1 card, only the single ended MOTF_CH_0 channel of the 37 pin connector is available:

USC-1
single ended tracks:
37-pin connector: MOTF_CH_0_A
37-pin connector: MOTF_CH_0_B

Table 29: Possible MOTF connection for USC-1

Here, the individual options for a USC-1 card are described. To get to the dialog window below (figure 254), select *Menu bar → Settings → System → Optic → Advanced.*

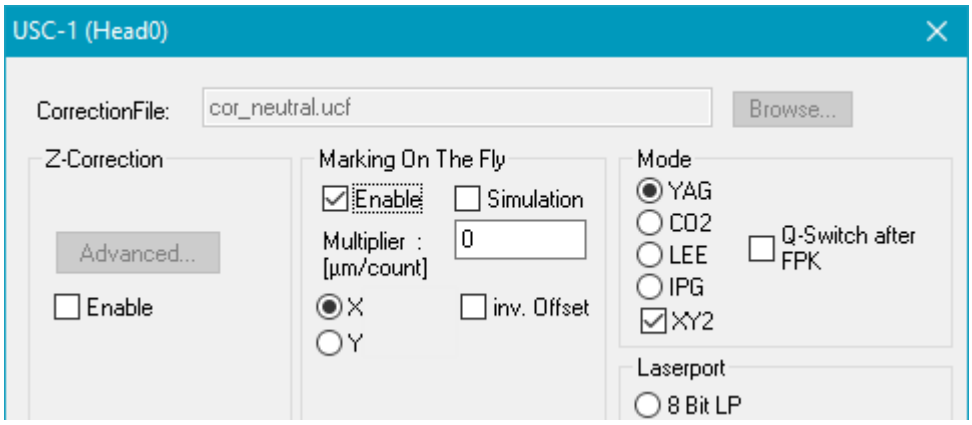


Figure 254: MOTF Settings for USC-1 Cards

Marking On The Fly:

Enable: Activate this checkbox to enable Marking On The Fly.

Simulation: Activate this checkbox to enable Simulation Mode.

Multiplier [µm/count]: Enter the µm/count factor here. If you enter a negative value, the MOTF compensation is switching the direction.
Example of a USC-1 MOTF-Multiplier calculation. In this calculation the radius of the encoder wheel is 50 mm and the encoder is generating 2000 pulses per track (A and B) and full rotation:

$$Multiplier\left[\frac{\mu m}{count}\right] = \frac{circumference\ [\mu m]}{4 \cdot increment} = \frac{2\pi \cdot 50000 \frac{\mu m}{360°}}{4 \cdot 2000 \frac{counts}{360°}} = 39.27 \frac{\mu m}{count}$$

X/Y: Choose the direction in which the target is moving during Marking On The Fly (for neutral optic settings).

inv. Offset: ScMotfOffset control objects can only be positive. It is possible that the MOTF compensation is set up correctly but the MOTF counter value is decreasing (counting negative). In this case, you have to enable this checkbox to be able to use ScMotfOffset control objects.

NOTE *ScMotfOffset control objects do not work in MOTF simulation mode for USC-1 cards.*

Only the following configurations allow correct compensation and advanced MOTF features (like ScMotfOffset):

Setup			USC-1 MOTF settings	
MOTF direction	**Phase shift track A - B**	**Orientation of MOTF coordinate**	**Sign of MOTF multiplier**	**Inv. Offset checkbox**
→ or ↑	B after A	normal	positive	unchecked
→ or ↑	B after A	inverted	negative	checked
→ or ↑	B before A	normal	negative	unchecked
→ or ↑	B before A	inverted	positive	checked
← or ↓	B after A	normal	negative	checked
← or ↓	B after A	inverted	positive	unchecked
← or ↓	B before A	normal	positive	checked
← or ↓	B before A	inverted	negative	unchecked

Table 30: USC-1 MOTF settings for different setups

Phase shift track A - B (refer to chapter Encoder Signals):

- B after A: track B is 90° (pi/2) phase-shifted to track A.
- B before A: track B is -90° (-pi/2) phase-shifted to track A.

Orientation of MOTF coordinate:

- normal: the MOTF axis is not inverted in SAMLight → Settings → System → Optic
- inverted: the MOTF axis is inverted in SAMLight → Settings → System → Optic

The orientation of the coordinate system of the UCF correction file has no influence on the MOTF settings.

It is highly recommended to set 'Settings → System → Optic → Rotation' to '0':

- The rotation rotates the entity, but not the MOTF direction.

If a rotation is required, implement the rotation directly in the UCF correction file.

If the marking is started with an external trigger (sensor), it is recommended to start the marking in trigger mode (Mark →Trigger) to avoid jitter of the position.

13.4.2 Card Specific: USC-2/3

For the USC-2/3 card, two MOTF channel are available.

- The MOTF signals of Channel 0 (37-pin connector) are single-ended: MOTF_CH0_A and MOTF_CH0_B.
- The MOTF signals of Channel 1 (extension 40-pin connector) are differential: MOTF_CH1_A+, MOTF_CH1_A-, MOTF_CH1_B+ and MOTF_CH1_B-. Do not use this channel single ended!

USC-2/3
single ended tracks:
37-pin connector: MOTF_CH_0_A
37-pin connector: MOTF_CH_0_B
differential tracks:
40-pin connector: MOTF_CH_1_A+
40-pin connector: MOTF_CH_1_A-
40-pin connector: MOTF_CH_1_B+
40-pin connector: MOTF_CH_1_B-

Table 31: Possible MOTF connection for USC-2/3

Here, the individual options for a USC-2/3 card are described. To get to the dialog window below (figure 255), select *Menu bar → Settings → System → Optic → Advanced.* Enable Marking On The Fly and then go to *Settings*.

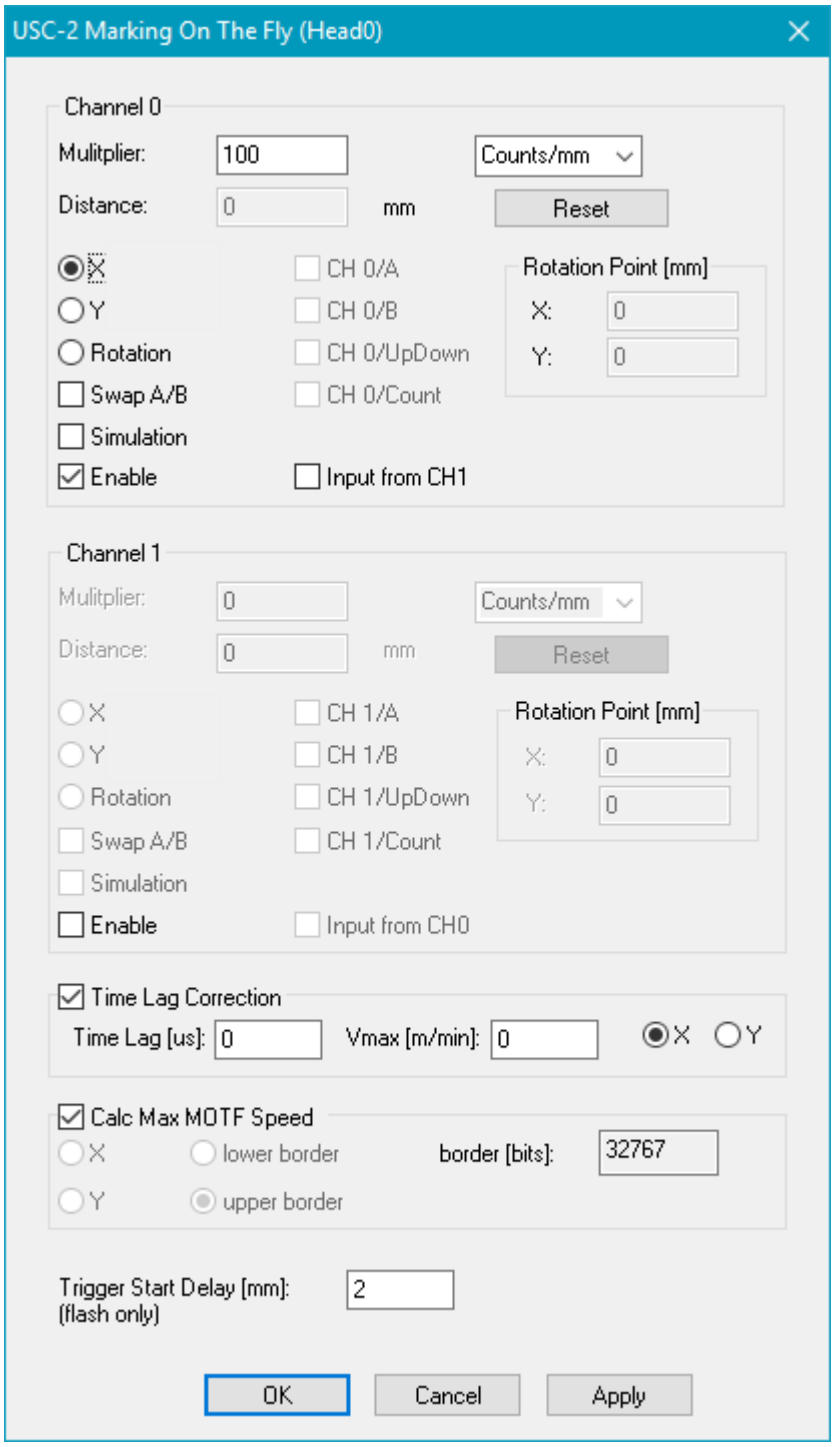


Figure 255: MOTF Settings for USC-2/3 Cards

Channel 0/1: There are two separate channels for MOTF on USC-2/3, Channel 0 and Channel 1.

Enable: Activate this checkbox to enable the desired channel for Marking On The Fly. It is possible to use both (see example rotated scan head).

Multiplier: Unit in [Counts/mm]. When using *Simulation* mode, choose [mm/s] or [m/min] as unit. Using a negative value inverts the direction of the MOTF compensation.
Example of a USC-2/3 MOTF-Multiplier calculation. In this calculation the radius of the encoder wheel is 50 mm and the encoder is generating 2000 pulses per track (A and B) and full rotation:

$$Multiplier\left[\frac{counts}{mm}\right] = \frac{4 \cdot increment}{circumference\ [mm]} = \frac{4 \cdot 2000 \cdot \frac{counts}{360°}}{2\pi \cdot 50 \cdot \frac{mm}{360°}} = 25.46\frac{counts}{mm}$$

Distance: If an encoder is used or if simulation is enabled, the distance of the Marking On The Fly movement is shown here. Normally, this number starts to increment as soon as the encoder or simulation mode is activated and when the belt is moving.

X/Y: Choose the axis in which the scanner is moving during Marking On The Fly (for neutral optic settings).

Rotation: A special feature which is only available for a USC-2/3 card is the rotational Marking On The Fly. Therefore activate the radio button *Rotation*. Then, choose the appropriate units - either Counts/deg or Deg/sec (Deg = Degrees). Finally choose the center of the rotation in *Rotation Point*. You might want to first center the job in the View2D before setting the center X and Y values.

Swap A/B: If activated, the direction of encoder counting will be inverted. The effect of this checkbox is identical to changing the hardware wiring between track A and track B.

Simulation: Activate this checkbox to enable the Simulation Mode.

Reset: This button resets the distance counter to 0.

Input from CH1/0: When activated, the encoder signals of the other MOTF channel are used for MOTF correction. The state of Swap A/B is also taken form the source channel. See Rotated scan head for an example of application.

Time Lag Correction: To compensate the delay of the scan head, a **'Time Lag [µs]'** can be defined. Depending on the current MOTF speed this result in an offset. **'Vmax [m/min] / Time Lag [µs]'** is the maximum of this offset. The **'X'** and **'Y'** radio buttons define the dimension of the Time Lag Correction. This feature only works when the encoder pulses result in a positive increase of the distance value. After any changes of the Time Lag Correction, the 'Settings → System → Optic → Advanced → Store' button must be clicked.

Calc Max MOTF Speed: In standalone (flash) mode, it is possible to check the maximal possible MOTF speed of the current job. Steps to use this feature:

- Enable Calc Max MOTF Speed. Only the checkbox must be enabled, the other settings are taken from the MOTF settings above. If both MOTF channels are in use, the CH1 settings are taken.
- Click Settings → System → Optic → Advanced → Store.
- Stop the MOTF belt movement such that there are no more encoder pulses. If the calculation is done while the belt is moving, the result will be less accurate.
- Mark a job in flash mode. Please note that the calculation is only working for jobs without any control objects.
- Start the server in visible mode (<SCAPS>\system\sc_usc_server /v). Select the USC-2/3 card and click on the InfoView button and scroll down to 'MaxMOTFSpeed [m/min]'.
- The 'MaxMOTFSpeed [m/min]' value should be a close approximation of the maximum possible MOTF speed for the current job.

Trigger Start Delay: When a value for the delay is entered, a ScMotfOffset and a ScWaitForTrigger object will be added to the top of the job list. These control objects will delay the job execution until the distance[mm] is reached after the external trigger signal. There must be only one trigger pulse and this pulse must be shorter than the "Distance[mm] / MOTF-Speed[mm/s]". This feature is only implemented for standalone mode (flash mode).

Only the following configurations allow correct compensation and advanced MOTF features (like ScMotfOffset or endless loop applications):

Setup			USC-2/3 MOTF settings	
MOTF direction	Phase shift track A - B	Orientation of MOTF coordinate	Sign of MOTF multiplier	Swap A/B checkbox
→ or ↑	B after A	normal	positive	unchecked
→ or ↑	B after A	inverted	negative	unchecked
→ or ↑	B before A	normal	negative	checked
→ or ↑	B before A	inverted	positive	checked
← or ↓	B after A	normal	negative	unchecked
← or ↓	B after A	inverted	positive	unchecked
← or ↓	B before A	normal	positive	checked
← or ↓	B before A	inverted	negative	checked

Table 32: USC-2/3 MOTF settings for different setups

Phase shift track A - B (refer to chapter Encoder Signals):

- B after A: track B is 90° (pi/2) phase-delayed to track A (of the corresponding channel).
- B before A: track B is -90° (-pi/2) phase-delayed to track A (of the corresponding channel).

Orientation of MOTF coordinate:

- normal: the MOTF axis is not inverted in SAMLight → Settings → System → Optic
- inverted: the MOTF axis is inverted in SAMLight → Settings → System → Optic

The orientation of the coordinate system of the UCF correction file has no influence on the MOTF settings.

It is highly recommended to set 'Settings → System → Optic → Rotation' to '0'.

- The rotation rotates the entity, but not the MOTF direction.

If a rotation is required, implement the rotation directly in the UCF correction file.

Do not forget to press the 'Store' button and save the settings.

In 'sc_usc_server → Info View', the following should be true if MOTF is set up correctly:

- MotfCNT counts positive when 'sign of MOTF multiplier' is positive. MotfCNT counts negative when 'sign of MOTF multiplier' is negative.
- MotfCNTS counts always positive.
- MotfSpeed is always positive.

If the marking is started with an external trigger (sensor), it is recommended to start the marking in trigger mode (Mark →Trigger) to avoid jitter of the position.

13.4.3 Card Specific: RTC cards

Here, the individual options for RTC cards (RTC3, RTC4, RTC5 and RTC6) are described. **Make sure that the hardware license for the RTC card (from Scanlab) includes Marking on the Fly.**
For RTC cards, the following MOTF channel are available:

RTC3/4/5/6
differential tracks:
16-pin connector: Encoder X1+
16-pin connector: Encoder X1-
16-pin connector: Encoder X2+

RTC3/4/5/6
16-pin connector: Encoder X2-
16-pin connector: Encoder Y1+
16-pin connector: Encoder Y1-
16-pin connector: Encoder Y2+
16-pin connector: Encoder Y2-

Table 33: Possible MOTF connection for RTC cards

To get to the dialog, choose *Menu bar → Settings → System → Card → Advanced → Driver Settings for RTC3, RTC4, RTC5 or RTC6*. Enable Marking On The Fly and then go to *Settings*.

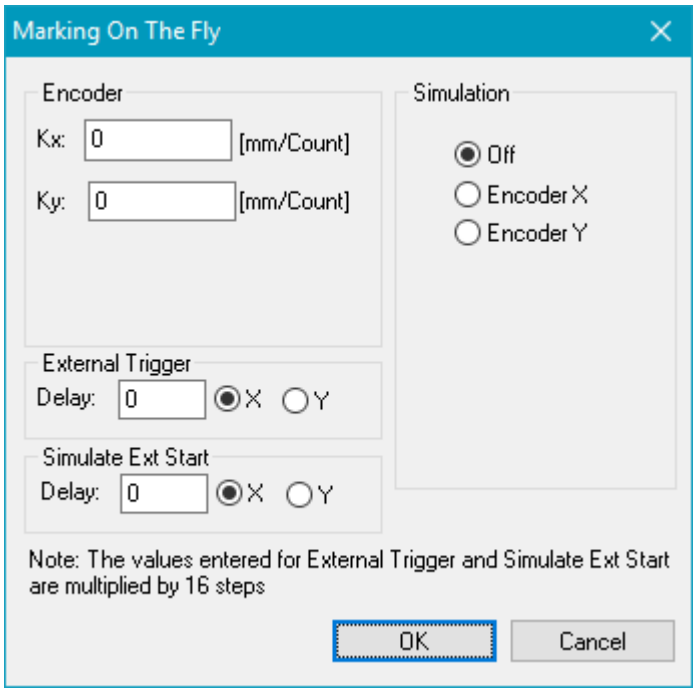


Figure 256: MOTF Settings for RTC Cards

Encoder Kx/Ky: Distance in x/y direction per encoder count. Example of a RTC MOTF-Kx/Ky calculation. In this calculation the radius of the encoder wheel is 50 mm and the encoder is generating 2000 pulses per track (A and B) and full rotation:

$$Kx/Ky\left[\frac{mm}{count}\right] = \frac{circumference\ [mm]}{4 \cdot increment} = \frac{2\pi \cdot 50\frac{mm}{360°}}{4 \cdot 2000\frac{counts}{360°}} = 0.03927\frac{mm}{count}$$

External Trigger: Allows to define an offset between external trigger pulse and the start of the job process. The offset is defined by *Delay.*

- RTC3/4: *Simulation* must be *Off*. The *Delay* value is internally multiplied by 16 counts.
- RTC5: The unit of *Delay* is encoder counts.

Simulate Ext Start: Allows to repeat the job automatically. The gap between the jobs is defined by *Delay.*

- RTC3/4: *Simulation* must be *Off*. The *Delay* value is internally multiplied by 16 counts.
- RTC5: The unit of *Delay* is encoder counts.
- The TriggerMode (Mark → Trigger) must be used.
- The initial job must be started with an external trigger signal on /START.
- Depending on the direction of the belt, a positive or negative number must be entered.

Simulation: Encoder pulses will be simulated with a constant pulse frequency of 1 MHz.

Off: No simulation
Encoder X: Simulation in X-direction
Encoder Y: Simulation in Y-direction

If the marking is started with an external trigger (sensor), it is recommended to start the marking in trigger mode (Mark →Trigger) to avoid jitter of the position.

13.5 Calibration

In the following, a general calibration routine is described.
This consists of:

- calibration of the optic system in the static situation, meaning without any movement of the belt
- calibration of the (well-calibrated static system) in the dynamic situation for the USC-1 card
- calibration of the (well-calibrated static system) in the dynamic situation for the USC-2/3 card

The following jobs are useful for calibration: a simple text object like "ABC" to check the orientation, a circle for easy observation of the compensation and a grid to check the accuracy.

Figure 257: ABC-circle job

ABC-circle.sjf consists of the letters A, B and C in a single laser font and a circle.

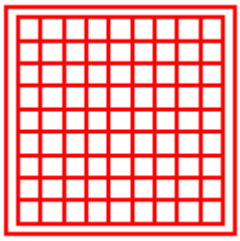


Figure 258: Grid job

Grid.sjf consists of grid vectors (e.g. hatch a square) to check the X and Y aspect ratio. If calibrating the static system, the grid may cover the whole field size.

13.5.1 Optic Calibration - static setup

Before starting with any Marking On The Fly application, the optic system needs to be well-calibrated in the static situation (no belt movement). A precisely calibrated static system (correction file and optic settings) is absolutely required and really critical for a good marking results in the dynamic system.

It is recommended to start with neutral optic settings:

SAMLight → Settings → System → Optic:

- Disable X invert, disable Y invert and disable XY flip.
- Set X gain to 1.0, Y gain to 1.0 and rotation to 0.0°.

SAMLight → Settings → System → Optic → Advanced → Correction, Settings:

- Set X gain to 1.0, Y gain to 1.0 and rotation to 0.0°.
- If necessary, inversion and flip of axes can be included in the correction file *.ucf using <SCAPS>\tools\sc_corr_table.

Start (without any movement of your target) to check the calibration in the static situation:

- Make sure your encoder does not send any pulses.
- Mark a job like ABC-circle.sjf: Is the orientation (X and Y axes) of the marking result correct - is the text "ABC" readable?
  - If not, the correction file must be manipulated. This can be done using <SCAPS>\tools\sc_corr_table (correct the orientation).

- Mark a job like Grid.sjf: Are the X and Y aspect ratios correct? In other words: is 1 mm in the drawing very precisely 1 mm on the target in vertical and horizontal direction throughout the whole working area (not only in the middle of the field)? This is very important for 1D MarkOnTheFly split applications, because the start point and the end point have a quite big distance within the scanner field in such an application. This is even more critical with increasing speed of the line and with increasing size of the closed shapes.
  - If not, adapt the field size until the dimension in the direction which you want to use as MOTF axis is correct. Mark again: are the X and Y aspect ratios correct?
    - If not, adapt the gain value of the axis, which is not used as MOTF axis. Mark again: are the X and Y aspect ratios correct?
      - If not, improve the correction file with sc_corr_table. Please find further information on how to work

with sc_corr_table online.

13.5.2 USC-1 Specific Calibration MOTF

Before starting with any Marking On The Fly application, the optic system needs to be well-calibrated in the static situation (no belt movement).

1. Check the proper alignment of the scanning field and the scanning direction?

- This can be tested e.g. with marking one line in the moving direction without moving the belt and marking a single point (with a mark loop count of -1) with the slowly moving belt on top of the last marking.
- If the alignment is not good, this can be solved by adjusting the hardware setup or within the correction file.

2. Mark without MOTF settings (without compensation) in SAMLight:

- Start the belt in the default direction with a slow movement. Mark ABC-circle.sjf. It is ok, if the marking result is not good because this marking is a reference to compare with a marking with MOTF compensation (see 259 1)).

3. Activate the MOTF compensation in SAMLight:

- Activate SAMLight → Settings → System → Optic → Advanced: Marking on the Fly by clicking on the checkbox 'Enable'.
- Select Channel 0 or Channel 1 according to the electrical connection of the encoder.
- Select X-Channel or Y-Channel according to the movement direction of the belt. Note: due to a flip in the optic settings the correct channel might be not obvious.
- As a starting value, set the multiplier to the value which can be calculated following the formula here.
- Leave the dialog with OK.

4. Mark with MOTF settings (with compensation) in SAMLight:

- Start the belt in the default direction with the same slow movement as in 2. Mark a job like ABC-circle.sjf. Compare the marking result with the result in 2.
  Is the marking result getting better with compensation or getting worse? This can be easily seen when looking at the gap of the circle. There are 3 different cases (see figure 259 2)).
  - o If the result is getting worse (see figure 259 2)a)) - e.g. the start and end of the circle are farther apart, change the wiring of your encoder or change the direction of the compensation by changing the sign of the MOTF multiplier value. See table "USC-1 MOTF settings for different setups" for more details.
  - o If the result is getting better (see figure 259 2)b)) - e.g. the start and end of the circle are coming closer, adjust the MOTF multiplier.
  - o If the result is getting better but you run into overcompensation (see figure 259 2)c)) - e.g. the start and end of the circle changed orientation in the direction of the movement, fine tune the MOTF multiplier.

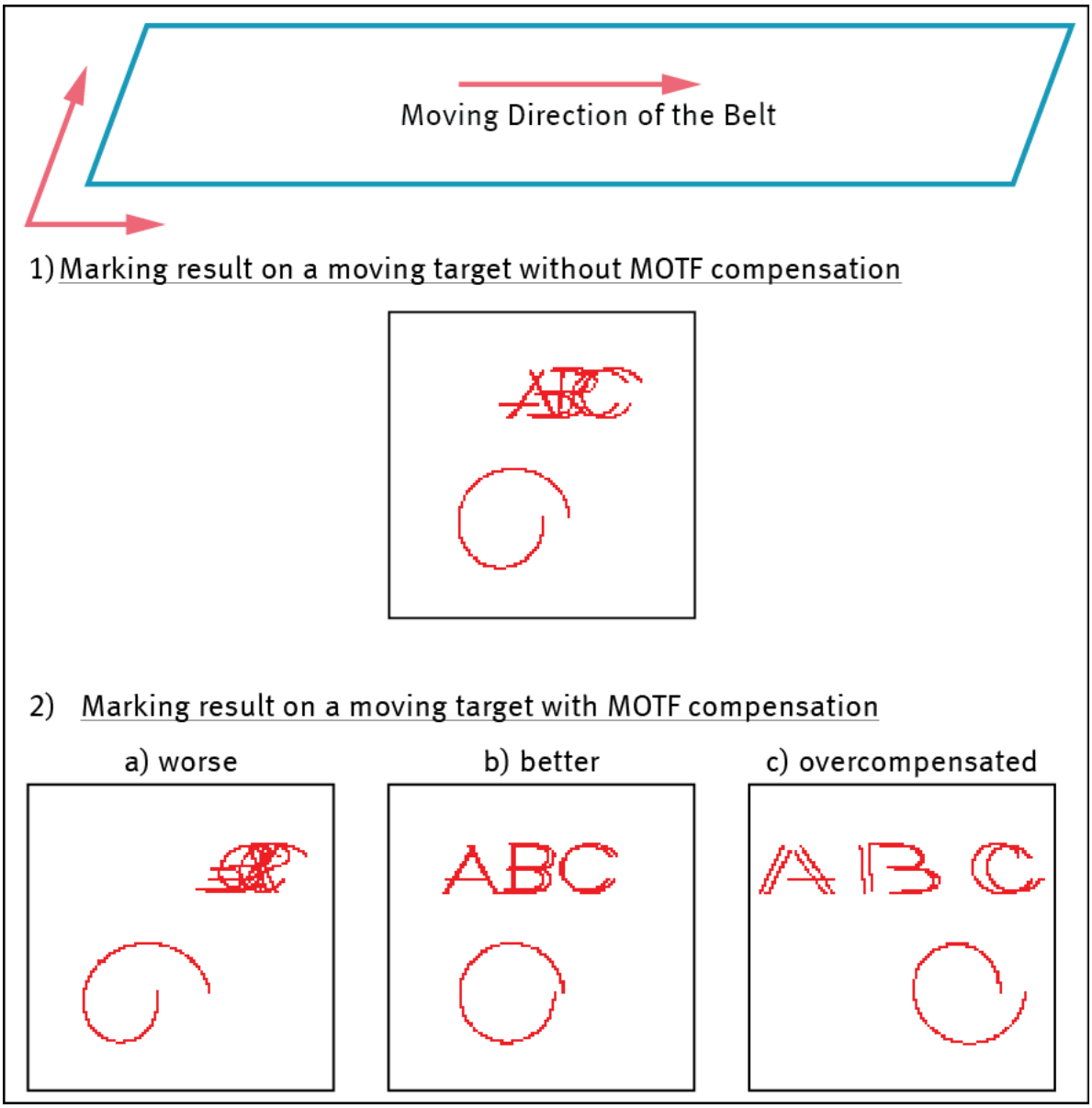


Figure 259: Different cases of MOTF compensation. For a moving belt to the right, an example of a marking result of ABC-circle.sjf without MOTF compensation is shown in 1). When the MOTF compensation is activated in SAMLight, the marking result depends on the precise value of the MOTF multiplier. Three different marking results for three different MOTF multiplier values are shown in 2).

5. **Mark with MOTF settings a job containing 2 entities separated by a ScMotfOffset entity.**
   - Create a job containing 2 entities, e.g. two circles (or a circle and a square) with identical outline dimension and center both entities in x and y. Add an ScMotfOffset entity a the beginning and specify a certain distance, e.g. 10mm. Add an ScWaitForTrigger entity after the first ScLayer entity (e.g. the circle). See picture in figure 260.
   - Start the marking. Are both entities marked?
     - o If you are just seeing the first entity, change the checkbox invert Offset in SAMLight. See table "USC-1 MOTF settings for different setups" for more details. "
   - Is the distance between the entities as specified within the ScMotfOffset entity?
     - o If not, it could be that the marking time of the first entity was too long for the offset. Try with a larger offset or faster marking speed or smaller entity. Alternatively, the initial external trigger was applied longer than the corresponding offset time. Another possibility is that the 2D calibration of the static setup or the MOTF multiplier is not yet optimal.

Name	Type
offset	ScMotfOffset
circle	ScLayer
trigger	ScWaitForTrigger
square	ScLayer

Figure 260: Example job containing 2 ScLayer entities, ScMotfOffset and ScWaitForTrigger

13.5.3 USC-2/3 Specific Calibration MOTF

Before starting with any Marking On The Fly application, the optic system needs to be well-calibrated in the static situation (no belt movement).

The flow chart in figure 261 shows an overview of the general steps of the MOTF software setup calibration. Please proceed with the steps below.

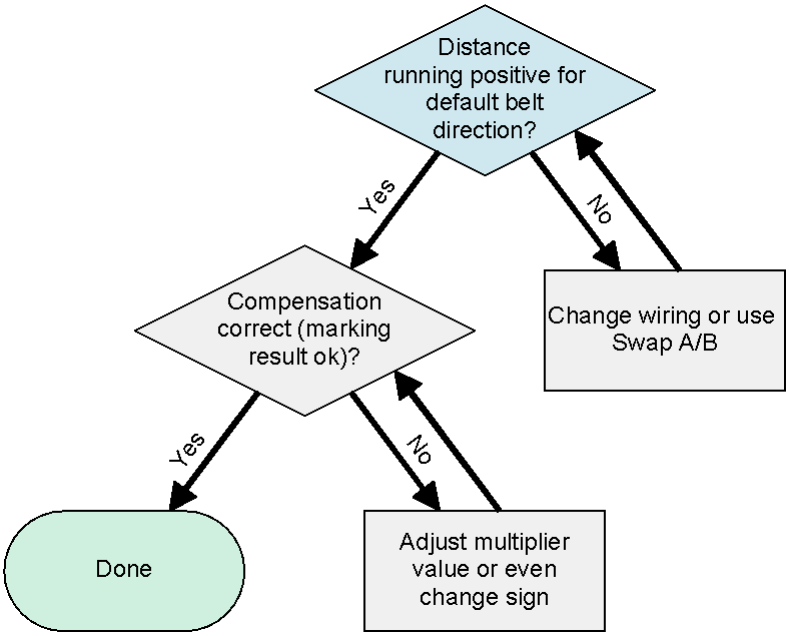


Figure 261: Flow Chart MOTF software setup

1. Check the proper alignment of the scanning field and the scanning direction?

- This can be tested e.g. with marking one line in the moving direction without moving the belt and marking a single point (with a mark loop count of -1) with the slowly moving belt on top of the last marking.
- If the alignment is not good, this can be solved by adjusting the hardware setup or within the correction file.

2. Mark without MOTF settings (without compensation) in SAMLight:

- Start the belt in the default direction with a slow movement. Mark ABC-circle.sjf. It is ok, if the marking result is not good because this marking is a reference to compare with a marking with MOTF compensation (see 262 1)).

3. Activate the MOTF compensation in SAMLight:

- Activate SAMLight → Settings → Optic → Advanced → Marking on the Fly by clicking on the checkbox 'Enable'. Then, please reopen the Advanced dialog.
- Enable Channel 0 for MOTF inputs on the 37 pin connector or enable Channel 1 for the MOTF inputs on the 40 pin extension connector (according to the electrical connection of the encoder).
- Select X-Channel or Y-Channel (or Rotation) according to the movement direction of the belt. Note:

due to a flip in the optic settings the correct channel might be not obvious.
- Select the multiplier unit: counts/mm. The units m/min or mm/s are also available (mm/s is only for Simulation Mode with a reference of 100 kHz).
- As a starting value, set the multiplier to the value which can be calculated following the formula here.
- Click on Apply and on the store button in the Optic → Advanced dialog after any MOTF settings changes.

4. Start the belt with a slow movement in the default direction:

- Is the value of 'SAMLight → Settings → Optic → Advanced → Marking on the Fly → Distance' counting positive?
  - If not, change the wiring of the MOTF tracks (A and B) or use the checkbox "Swap A/B" in Settings → System → Optic → Advanced → MOTF.
  - Make sure that when driving in the default direction, the distance counter will move in positive direction. Otherwise for example the ScMotfOffset control object will not work properly.

- Move the MOTF belt for different defined distances. Is the distance value displayed in the software correctly?
  - If not, adjust the MOTF multiplier value.

5. Mark with MOTF settings (with compensation) in SAMLight:

- Start the belt in the default direction with the same slow movement as in 2. Mark a job like ABC-circle.sjf. Compare the marking result with the result in 2.
  Is the marking result getting better with compensation or getting worse? This can be easily seen when looking at the gap of the circle. There are 3 different cases (see figure 262 2)).
  - If the result is getting worse (see figure 262 2)a)) - e.g. the start and end of the circle are farther apart, change the wiring of your encoder or change the direction of the compensation by changing the sign of the MOTF multiplier value. See table "USC-2 MOTF settings for different setups" for more details.
  - If the result is getting better (see figure 262 2)b)) - e.g. the start and end of the circle are coming closer, adjust the MOTF multiplier.
  - If the result is getting better but you run into overcompensation (see figure 262 2)c)) - e.g. the start and end of the circle changed orientation in the direction of the movement, fine tune the MOTF multiplier.

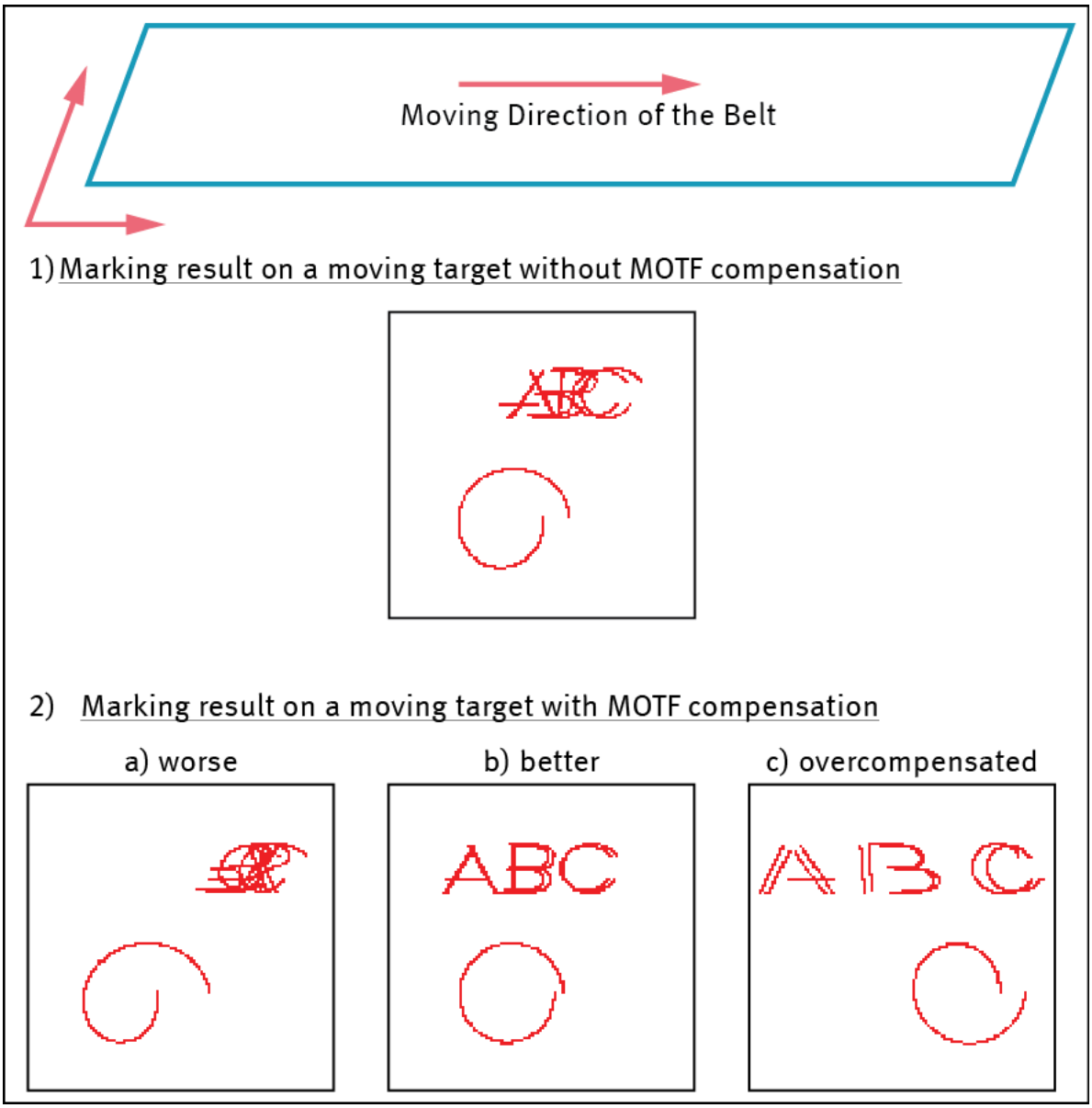


Figure 262: Different cases of MOTF compensation. For a moving belt to the left, an example of a marking result of ABC-circle.sjf without MOTF compensation is shown in a. When the MOTF compensation is activated in SAMLight, the marking result depends on the precise value of the MOTF multiplier. Three different marking results for three different MOTF multiplier values are shown in b.

6. **Mark with MOTF settings a job containing 2 entities separated by a ScMotfOffset entity.**
   - Create a job containing 2 entities, e.g. two circles (or a circle and a square) with identical outline dimension and center both entities in x and y. Add an ScMotfOffset entity a the beginning and specify a certain distance, e.g. 10mm. Add an ScWaitForTrigger entity after the first ScLayer entity (e.g. the circle). See picture in figure 263
   - Start the marking. Are both entities marked?
     - o If you are just seeing the first entity, change the checkbox invert Offset in SAMLight. See table "USC-2 MOTF settings for different setups" for more details. "
   - Is the distance between the entities as specified within the ScMotfOffset entity?
     - o If not, it could be that the marking time of the first entity was too long for the offset. Try with a larger offset or faster marking speed or smaller entity. Alternatively, the initial external trigger was applied longer than the corresponding offset time. Another possibility is that the 2D calibration of the static setup or the MOTF multiplier is not yet optimal.

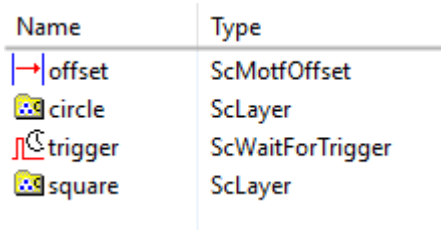


Figure 263: Example job containing 2 entities, ScMotfOffset and ScWaitForTrigger

The amplitude of the encoder signals should be in the range of 2.5 V to 5.0 V. The signals should be as low-noisy as possible, since noise could be interpreted as additional pulses. Possibly, you can use a differential encoder and MOTF_CH1 (40-pin connector).

13.5.4 Tips to optimize MOTF performance

Tips for increasing of the maximum possible MOTF speed:

- If using Flash with FCI Calls: Increase MOTF Return Speed (MRS) to get less MOTF dead time (for USC-2/3).
- Job set up: Place the marking object in the opposite of the MOTF direction in the working area to increase the usable MOTF area. Make sure that the gap to the next object is still big enough for the jump.
- Optic settings: Enable HomeJump and set the position to the start point of the job file to decrease the start jump.
- Pen settings: Decrease scanner delays and increase scanner speeds, do not forget the HomeJumpStyle (pen 256).
- Use standalone mode (flash mode) to get less MOTF dead time.

13.6 Endless MOTF

SAMLight provides the option to do endless cutting or endless looping, giving the possibility to fold a polyline in order to pre-compensate the MOTF movement.
For these Flash-only-features (only for USC-2/3), the SAMLight Option Flash and Option MOTF are required. Then, the job needs to be prepared in the following way:

1. First, the unfolded polyline is created in SAMLight. The entity must not be more than one single polyline (no jumps allowed). The Y-coordinate of the first and of the last point of this polyline must be identical, the X-coordinate of these two points must be different.
2. Then, this polyline is folded
   a. either while being exported as CNC with the advanced option of endless cut or endless loop (*File → Export*, choose *GCode Files (*.cnc)* as file type and open the *Advanced... dialog*; see figure 265)
   b. or while being saved to the Flash memory. (*Extras → Flash*, activate the checkbox *SJF → CNC → UNF* and open the *settings* dialog on the right; see figure 265).
3. If the polyline was folded by the export function of SAMLight, this CNC file can then be loaded to the USC card (e.g. per FTP server) and convert to UNF file. It can also be imported into SAMLight for the purpose of illustration (see figure 264).

In Endless Cut and in Endless Loop modes, the target speed should be close to a nominal MOTF speed V(MOTF). For MOTF speed variances, a look-up-table for the scanner speed and for the laser power is provided for up to 16 controller cards. Please note that the outline of the polyline can be larger than the field size of your scanner along the MOTF direction before the convolution has been executed. If you want to save the values of the CNC export in your jobfile, check *File → Job Properties... → MOTF Multiplier, Save in Job / Load from Job*. To save the setting for the look-up-table (see illustration in figure 266) for scanner speed and laser power, you have to select the polyline you want to mark. Then, go to *File → Export*, choose *GCode Files (*.cnc)* as file type and open the *Advanced... dialog.*

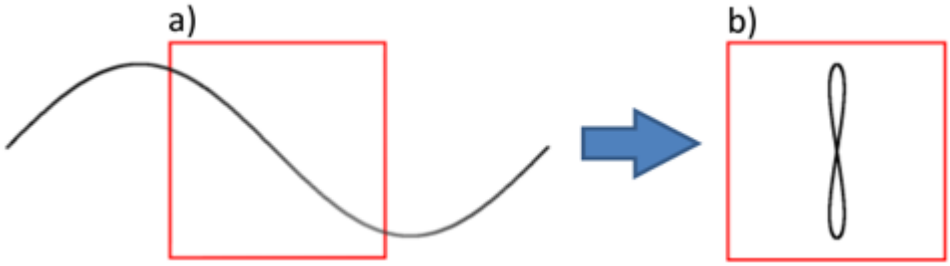


Figure 264: Illustration of Endless Loop: a) A sine curve with an outline larger than the field size, b) after CNC export the sine curve is folded to a closed form which fits into the field size and can be marked endlessly.

NOTE

*In this feature the vector compensation is not based on MOTF encoder values. Instead the vector calculation is pre-calculated by folding the polylines. This is done by generating the *.UNF job for a nominal speed and will be adjusted by internal override parameter which can be defined in lookup tables for speed and laser power. However, the internal override procedure is not done in real time but at the beginning at each job. If the belt speed is not constant this functionality will lead into small distortions of the vectors.*

The following endless MOTF dialog (figure 265) can be reached via *Extras → Flash*, activation of the checkbox *SJF → CNC → UNF* and opening of the *Settings* dialog on the right.

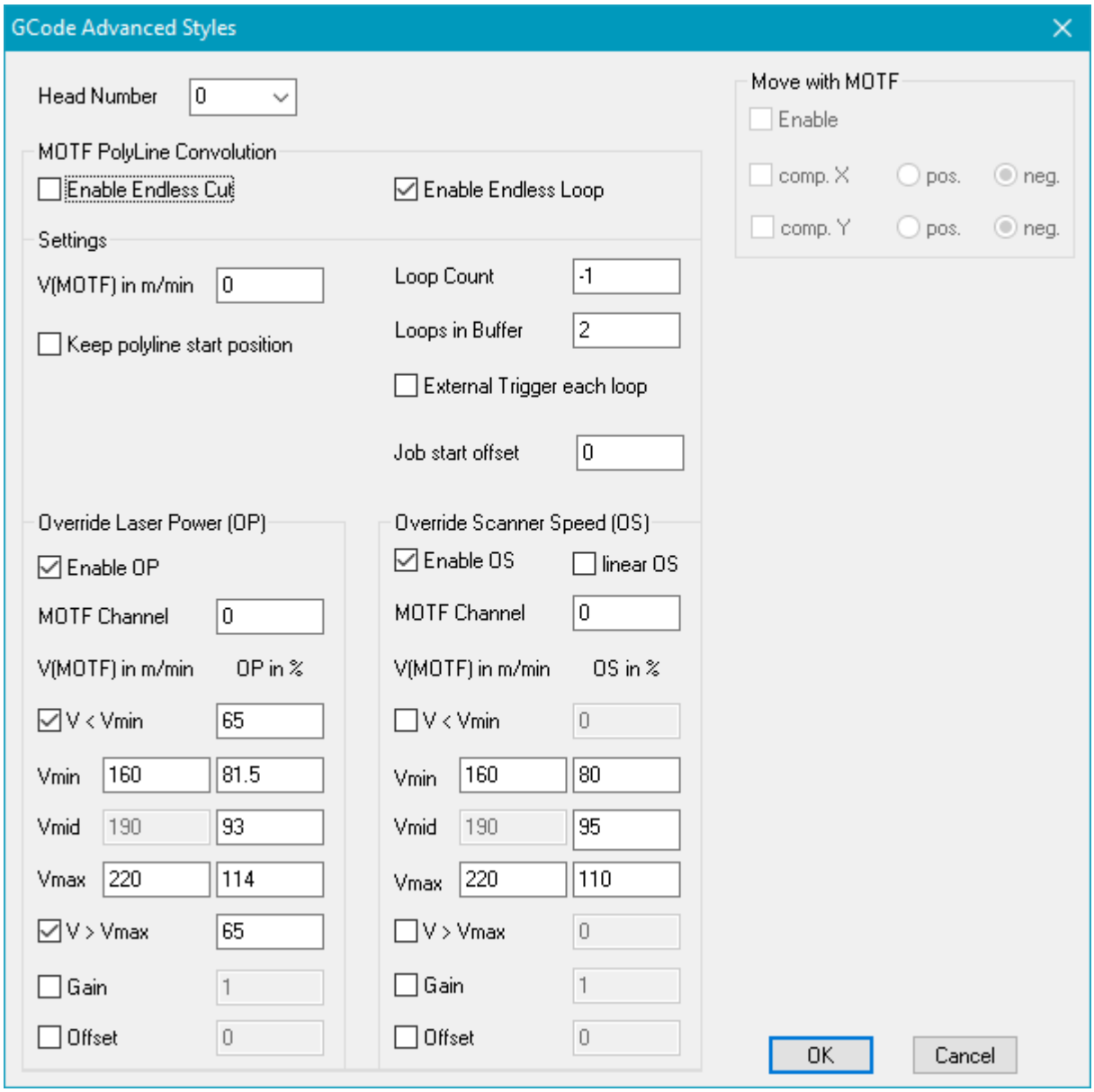


Figure 265: Endless MOTF Dialog

Head Number: Chooses the card number. For single card applications, choose '0'.

Enable Endless Cut: Folds the polyline(s) by the nominal MOTF speed V(MOTF).

Requirements of the polyline(s): The entity must be a single polyline or series of polines (no jumps between vectors are allowed).

V(MOTF) in m/min: Sets the nominal MOTF speed. This speed is used for the folding of the vectors.

Enable Endless Loop: Folds the polyline into a closed form. Laser will be left on all the time during marking.

Requirements of the polyline: The entity must be a single polyline (no jumps between vectors are allowed), the Y-coordinate of the first and the last point of the polyline have to be the same whereas the X-coordinates of these two points must be different.

V(MOTF) in m/min: Sets the nominal MOTF speed. This is used for calculating the scanner speed if override speed is disabled.

Keep polylines start position: When disabled, the folded polyline will be centered. When activated, the

start position of the polyline will not be changed. In this case, the start point of the polyline needs to be within the valid field.

External Trigger each loop: The USC card waits for external trigger via OPTO_IN_0. This can be used in combination with endless cut.

Loop Count: Define the number of loops the polyline will be marked, type -1 for endless marking.

Loops in Buffer: Buffered jobs are used to avoid gaps in the marking result, but buffered jobs will not be updated by OP / OS values.

Job start offset: A single MOTF offset is added at the beginning of the job. It is not included in the loop and only included if not in trigger mode.

Enable OP, Enable OS: For MOTF speed variances a look-up-table for scanner speed and laser power can be defined. This can be done by defining an lower and upper limit of the MOTF speed. In the range between Vmin and Vmax three nodes are set equidistantly, refer to the picture below. If Override Laser Power (OP) / Override Scanner Speed (OS) is enabled the laser power / scanner speed will be overwritten at the beginning of each job.

linear OS: Uses the nodes of OP for OS as well.

MOTF Channel: Defines the source of the MOTF encoder signal: *0* = MOTF-Channel 0 at the 37 pin connector, *1* = MOTF-Channel 1 at the 40 pin extension connector.

V < Vmin: Defines a constant scanner speed / laser power if MOTF speed is below the range of the look-up-table. If disabled, Vmin is used instead. The internal minimum override value is 20%, so do not use smaller values.

Vmin, Vmid, Vmax: Nodes of the look-up-table. Speed values in [m/min], override value in [%].

V > Vmax: Defines a constant scanner speed / laser power if MOTF speed is above the range of the look-up-table. If disabled, Vmax is used instead.

Gain: All nodes values of the look-up-table will be multiplied by this factor.

Offset: An additionally offset value can be applied to all node values.

The following figure illustrates the look-up-table of speed and power values for endless MOTF.
These setting for the look-up-table for scanner speed and laser power can be saved after selection of the polyline you want to mark via *File → Export*, choose *GCode Files (*.cnc)* as file type and open the *Advanced... dialog.*

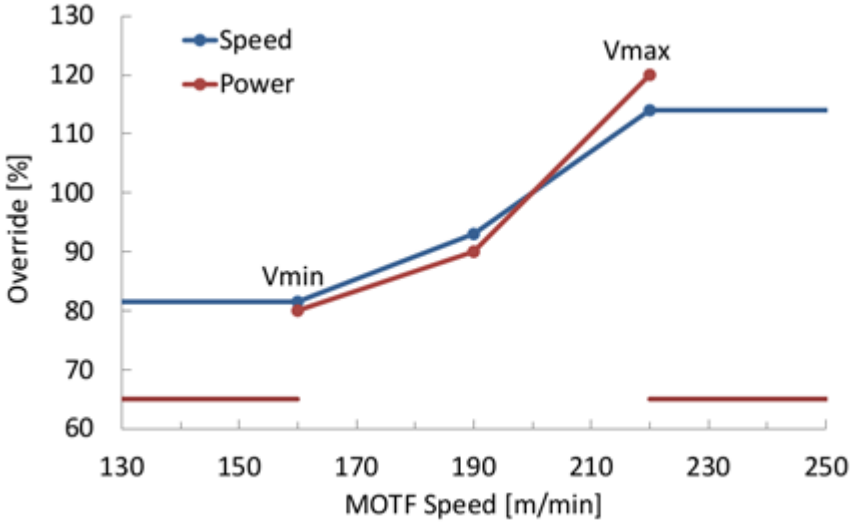


Figure 266: Illustration of Speed- and Power-look-up-table in endless MOTF

13.7 Examples

Here, some examples for MOTF jobs are given.

13.7.1 Trigger based offset

Marking On The Fly can be used to set up an assembly line production with many targets passing by one after another through the scanner field.

Figure 267 shows an example of such an assembly line. Each marking is started by an external trigger signal and the distance between the print mark for the trigger signal and the target is always the same. For **USC-1/-2/-3 and RTC-5 cards**, the Trigger Control Objects can be used to generate such an offset.

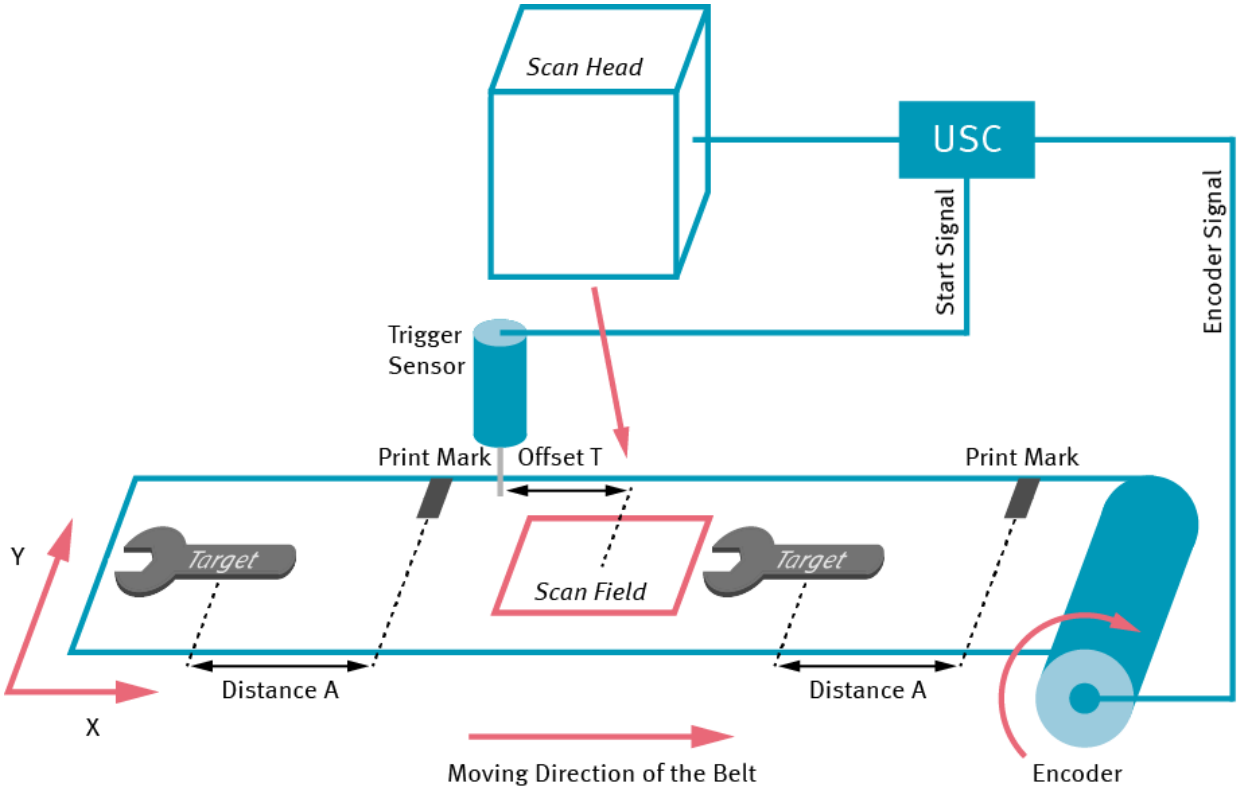


Figure 267: Assembly line with Marking On The Fly: the trigger sensor gives a start signal for each print mark to mark the target. Print mark is located distance A apart from the target.

The job for such a setup can look like in figure 268:

Name	Type	
Group	ScEntities2D	Mark Loop Count = -1
...		
Distance A	ScMotfOffset	
Internal Trigger after distance A	ScWaitForTrigger	
Target	ScEntities2D	

Figure 268: Job for Trigger based offset with Marking On The Fly

Please note that the distance which needs to be specified in the ScMotfOffset entity of the job depends on the positioning of the entity within the scan field (working area in SAMLight) as well as by the positioning of the trigger sensor in respect to the scan field (Offset T in figure 267).

13.7.2 Assembly Line

Marking On The Fly can be used to set up an assembly line production with many targets passing by each after another through the scanner field.

If there is no external trigger signal for each part, but only for the first one and if the distance between the parts is always the same, the assembly line can be realized by the following solution (discriminate between USC and RTC cards).

For USC-1/-2/-3 and RTC-5 cards, the Trigger Control Objects can be used together with the *Mark Loop Count*. In this case, set up the job as wished for one marking including all the trigger control objects. Then, select those objects in the entity list and group them all together with *Edit → Group*. Finally, set the *Mark Loop Count* of this group to *-1* in the *Entity Info* property sheet.

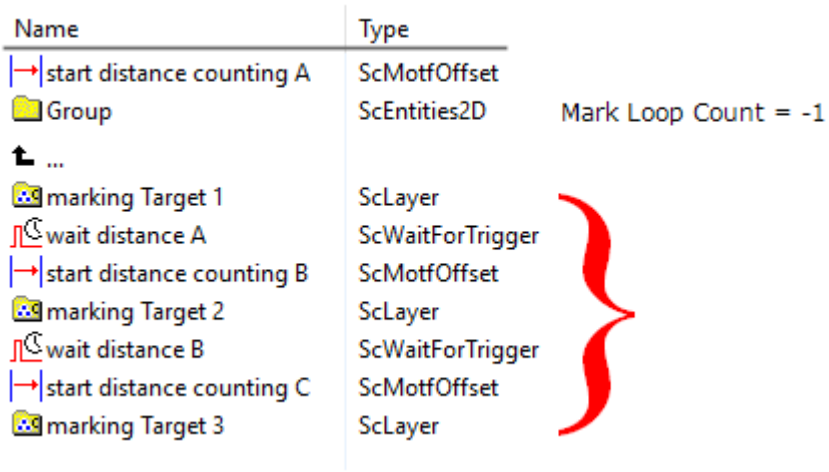


Figure 269: Example for MOTF Job for an Assembly Line

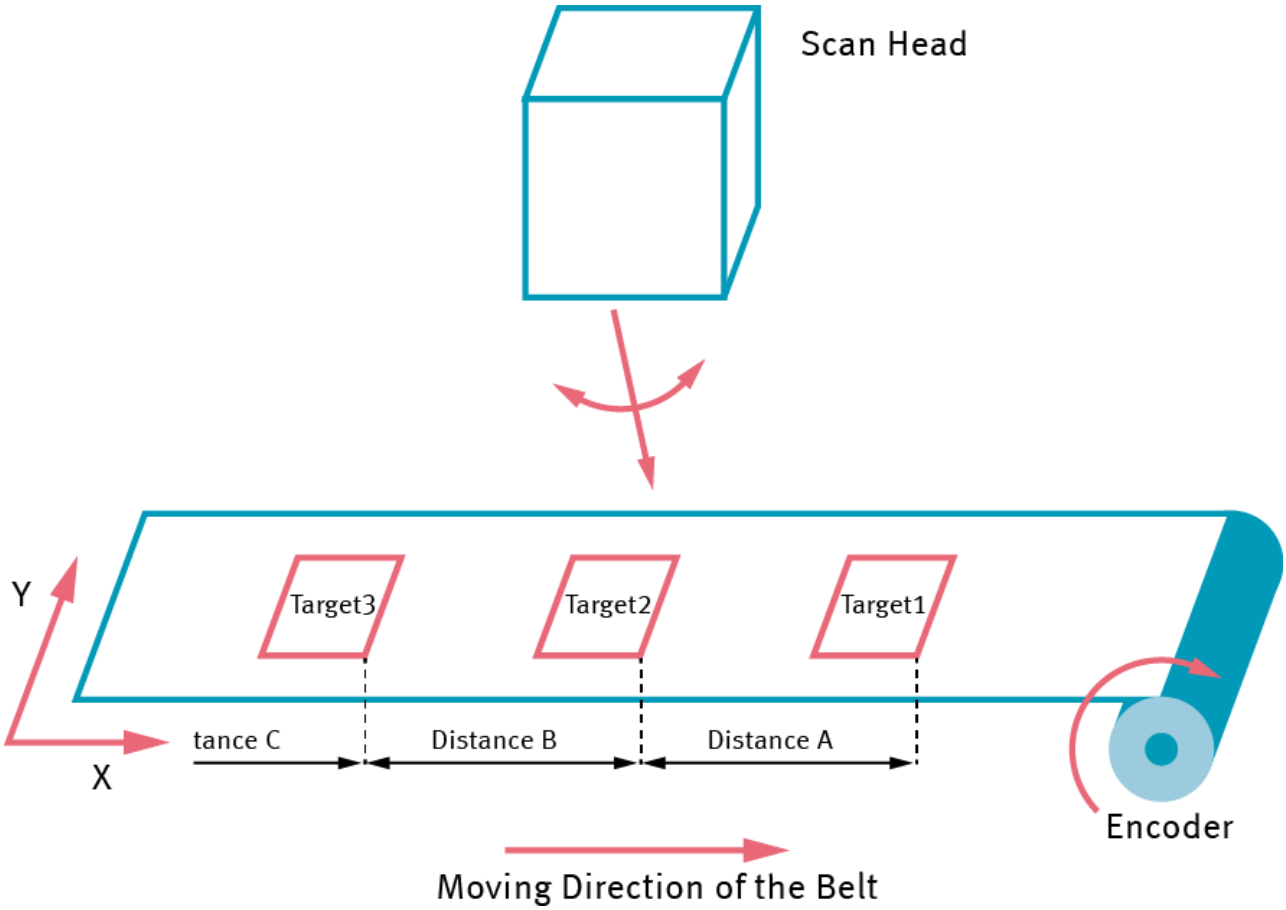


Figure 270: Assembly line with Marking On The Fly

For RTC-3/-4, the trigger control objects are not available. Looping is possible with the Simulate Ext Start feature.

For RTC-5 cards, the looping can alternatively be done with the Simulate Ext Start feature.

NOTE

You need an encoder for the RTC cards as well as for the USC-1 card. The simulation mode can not be used for this. Only the USC-2 and USC-3 card can handle the simulation mode together with the ScWaitForTrigger and ScMotfOffset objects.

13.7.3 Rotational MOTF (RMOTF)

Marking on the Fly on a rotary disk is possible for the USC-2/3 cards and serves for marking on the bottom of bottles which are rotated by a disk.

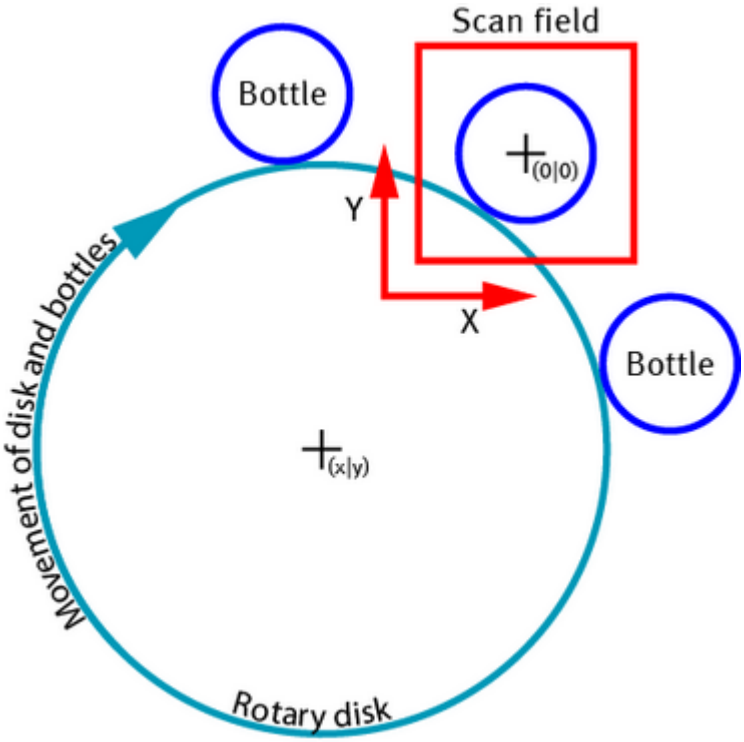


Figure 271: Setup RMOTF

Figure 271 shows a rotary disk (light blue) which is rotating clockwise. Bottles (dark blue) are rotated by this disk and thereby pass by the scan field (red) with the origin of the coordinate system (0|0). The center of the rotary disk (x|y) is given in respect to the origin within the scan field.

Please follow the following steps to set up such a system.

1. Step one:
   a. Make sure that you use the proper correction file.
   b. Make sure that the optic is calibrated in the static situation, i.e. 1 mm in the software equals 1 mm at the scanner output. Use the field size value for calibration.
   c. In figure 271, we use a 100 mm scan field, from -50 mm to 50 mm, so that the origin has the coordinates (0|0).
   d. Setup a job that contains a dot with the coordinates of your origin (in our example (0|0), figure 271).
   e. Go to the pen settings and enable the drill mode.
   f. The drill time, i.e. "Duration", depends on the disk speed.

2. Step two:
   a. RMOTF: off → disable Marking On The Fly in the Settings dialog (Settings → System → Optic → Advanced).
   b. Rotary disk: runs with a defined speed (deg / s)
   c. Start marking
   d. The marking result should be a segment of a circle where the distance from the rotary disk center to the origin of the scanfield equals the radius.
   e. The length of the segment depends on the drill duration and the disk speed.

3. Step three:
   a. RMOTF: on → enable Marking On The Fly in the Settings dialog
   b. Rotary disk: stands still
   c. Enable MOTF, simulation and choose "Rotation". See figure 271 to determine the correct coordinates of the Rotation Point (the point is (-x|-y) in this example).

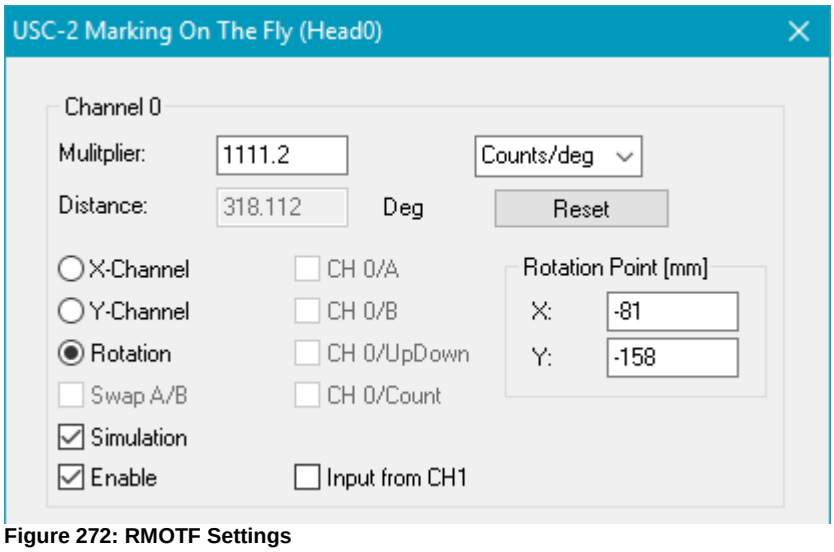


Figure 272: RMOTF Settings

d. Start marking
e. The result should be the same as in step two.
f. If the result is different from step two:
   i. The Multiplier takes effect on the length.
   ii. The Rotation Point takes effect on the radius.

4. Step four:
   a. Repeat step three until you get the same output as in step two. Radius and length must fit.

5. Step five:
   a. RMOTF: on → enable Marking On The Fly in the Settings dialog.
   b. Rotary disk: runs with a defined speed (deg / s).
   c. Correct the Multiplier so that it fits to the disk speed. Note: Consider the algebraic sign!
   d. Start marking
   e. The marking result must be a point!

6. Step six:
   a. RMOTF: on → enable Marking On The Fly in the Settings dialog.
   b. Rotary disk: runs with a defined speed (deg / s).
   c. Remove the point from the job and draw a square.
   d. Start marking
   e. The marking result must be a square!

13.7.4 Rotated scan head

In order to enlarge the marking length possible within the scan field, the scan head can be mounted with an angle between the X-axis of the scan head and the X-axis of the assembly line. To use the maximal possible marking length, the scan head can be rotated 45° in respect to the moving belt. Then, the MOTF multiplier must be adapted. Due to geometric considerations, the MOTF multiplier needs to be reduced by a factor of $\sqrt{(1/2)}$ in the case of a rotation of 45°.

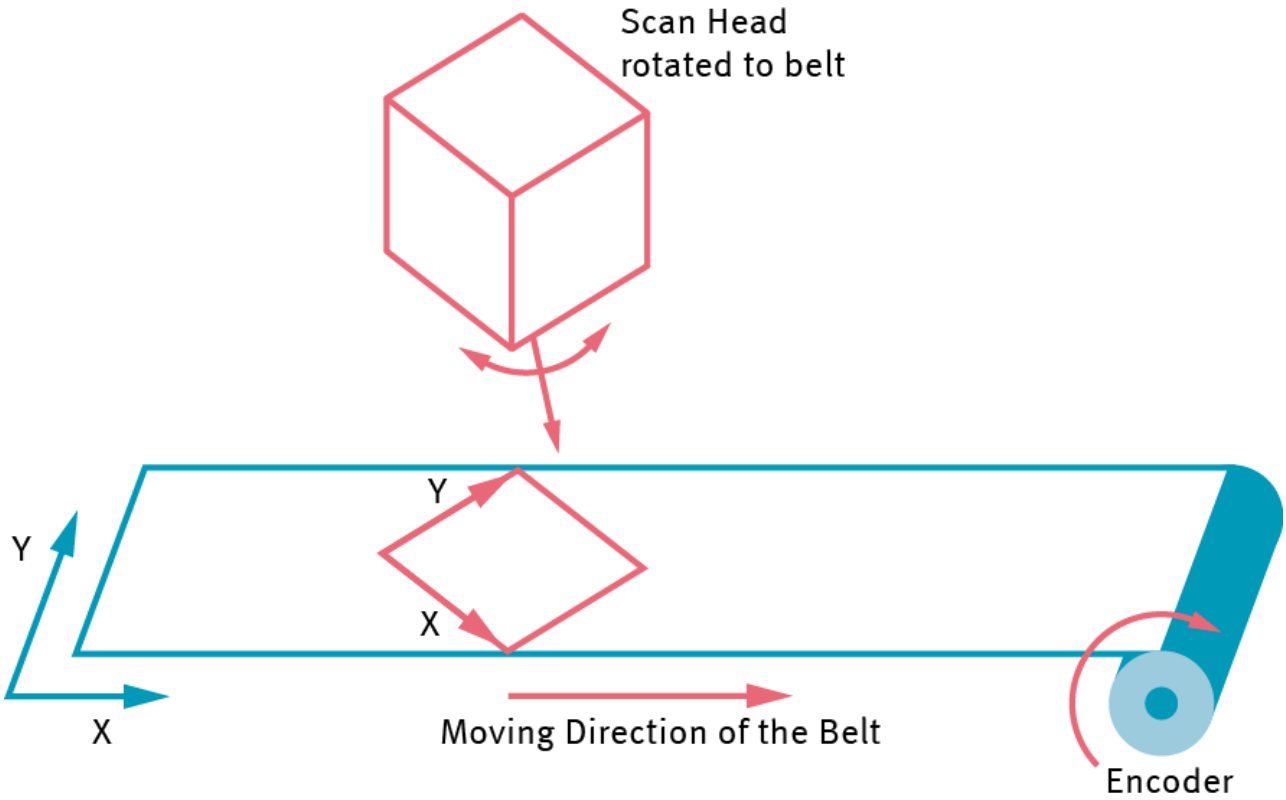


Figure 273: The scan head's coordinate system is rotated in respect to the belt's coordinate system.

14 Option Flash (USC-2/3 only)

The Flash option is available with an USC-2 card. In the past it was also available with an USC-1 card together with an FEB-1 board. The operation of the flash program is the same for these two setups. The FEB-1 (Flash Extension Board) is an optional card to the USC-1 which allows to store marking jobs which then can be executed in a stand alone mode means without a PC involved. The control in stand alone mode is possible via RS232 commands or the USC-2 or USC-1 and FEB-1 digital IO's can be used to select the desired job and start/stop the marking process. The job preparation is done with a PC based editor. The jobs are uploaded to the flash over the same way you connected the card to SAMLight (USB or Ethernet).

Figure 274: USC-2 Card

14.1 Supported Objects

The job preparation is done like for normal marking jobs without flash with the help of the job editor. It is recommended to save jobs to the flash only if they give a correct marking result in normal operation mode.

Flash supported objects/features:

- Entities:
  - Geometries
  - Hatch
  - Dynamic serial numbers (up to 64)
  - Serial number barcodes [a]
  - Date time
  - Text2D
  - Groups (mark loop count)
- Pen properties:
  - Skywriting
  - Power ramping
  - Drill mode (Use Geometry included)
  - Pen paths
- Control objects:
  - ScTimer
  - ScWaitForInput [b]
  - ScSetOutput [b]
  - ScSetAnalogOutput
  - ScMotionControl
  - ScMotfOffset
  - ScWaitForTrigger
  - ScOverride
- Marking on the fly (MOTF)
- Motion control stepper type 14
- Flash Control Interface (refer to USC-2 manual)

Not supported objects/features in flash:

- Splitting
- Bitmaps
- Red pointer
- References to other text elements <$EntityName>
- Serial number barcodes [a]
- Serial numbers from files
- Special sequences (pre-/post-processing)
- Motion Control (except stepper type 14)
- Laser power save mode
- Shutter control
- Status outputs and control inputs (except MIP)
- Automatic set/reset of DST (daylight save time)
- Client Control Interface
- Executable control objects
- SAM3D
- Optic3D
- MultiHead
- AnalogIn

[a]: Only the following barcodes can be generated in standalone mode dynamically:

- DataMatrixEx
- QR Code EX,
- Dotcode
- PostNet-A/-C/-Cp
- Code-128 (2)
- EAN-8
- 2 of 5 EX
- Code-39 EX

For DataMatrixEx, QR Code EX and Code-128 (2) it is necessary to activate the checkbox 'generate cells' in the barcode extended dialog. The feature 'enable text' does not work in flash.

[b]: When JobIOSelectMode is used, make sure I/O bits are not in conflict with the JobIOSelectMode sync bits.

On the flash, the field correction and laser source (CO2, YAG, LEE, IPG, ...) information are global. Mixing up jobs generated with different settings can lead to unpredictable results.

The global settings must be saved to the USC-2 card by clicking the 'STORE' button in Settings→System→Optic→Advanced. Please find further information in chapter Flash Jobs and Settings.

14.2 Flash Jobs and Settings

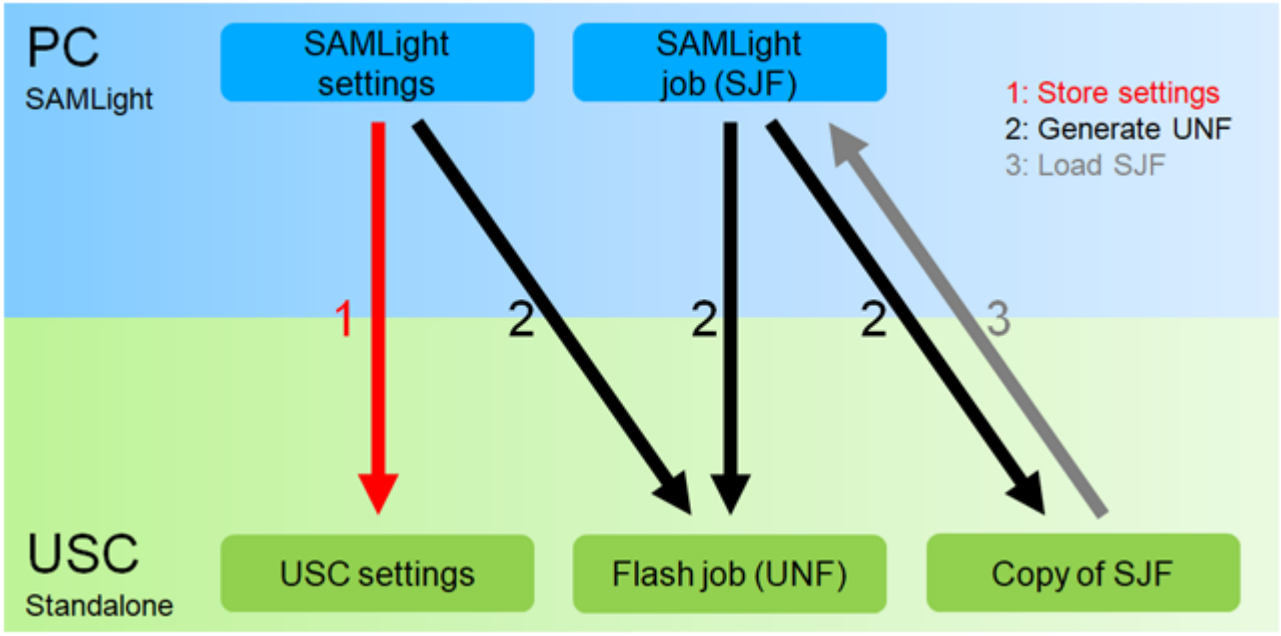


Figure 275: Handling of jobs and settings

The handling of jobs and settings for the USC standalone mode (Flash) as seen in figure 275 is explained in the following:

- **1: Store settings:** For standalone mode, the USC settings are used. Therefore, it is necessary to store the SAMLight settings on the USC card via the Store button. Please refer to figure 276 and figure 277.
- **2: Generate UNF:** For standalone mode, the Flash job (UNF) is created from the SAMLight job (SJF) together with a subset of the SAMLight settings and is saved on the USC card and in <SCAPS>\jobfiles\. Different SAMLight settings in combination with the same SJF will lead to different UNFs. Furthermore, a copy of the SJF is also stored on the USC card as part of the UNF file. This SJF file cannot be addressed individually.
- **3: Load SJF:** The copy of the SJF on the USC card can be loaded to SAMLight. However, the USC settings and the UNF cannot be loaded to SAMLight.

Different SAMLight settings in combination with the same SJF will lead to different UNFs.

Flash Jobs Handling:

The file formats SJF and UNF differ in several aspects:

- in the SJF format, objects are stored with the information of all position coordinates in mm and thus can be freely edited each time, the job is loaded. When you change the settings of a pen, the change will be valid for all objects which have this pen applied.
- in the UNF format, the information of coordinates of objects is no longer stored in mm. Instead, the position information is converted into bit values. The same is true for pen settings, which means, that the information of pen settings is already converted to direct commands. Thus, you don't have real pen settings anymore to change.

A Flash job can be created / loaded via one of the following ways:

- Create UNF from SJF or CNC and load to card:
  - SAMLight → Extras → Flash
  - Client Control Interface Command ScProcessFlashJob

- G-Code conversion with Flash command CGF

- Load existing UNF to card:
  - SAMLight → Extras → Flash
  - Client Control Interface Command ScProcessFlashJob
  - Visible USC Server → Flash → Load
  - FTP Server and Flash command JLA

Flash Settings Handling:

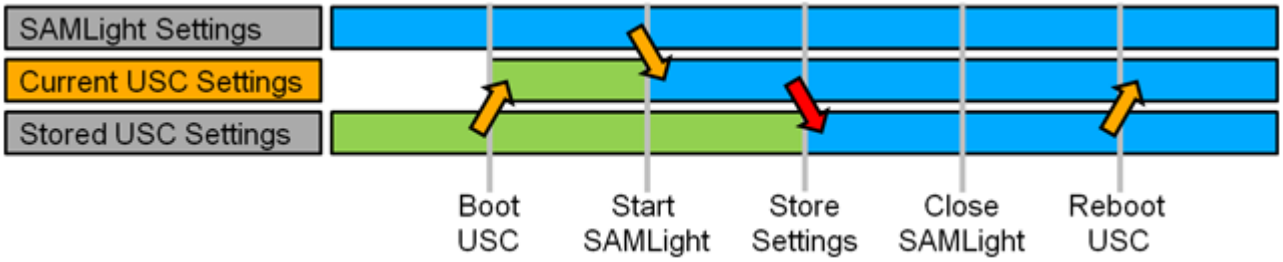


Figure 276: Correct handling: SAMLight and USC settings with proper storing

The current USC settings on the USC card change with different events, see figure 276:

- At boot of the USC card, the current USC settings are loaded from the stored USC settings.
- At SAMLight start, the settings of SAMLight overwrite the current USC settings.
- To avoid loss of these current USC settings after reboot of the USC card (see Fig.277), the current settings need to be stored. Storing the settings on the USC card can be done via the Store.
- The SAMLight settings remain as current USC settings even after SAMLight has been closed.
- The storing ensures that the SAMLight settings and the current USC settings are identical even after a reboot of the USC card.

Common mistakes without storing of settings:

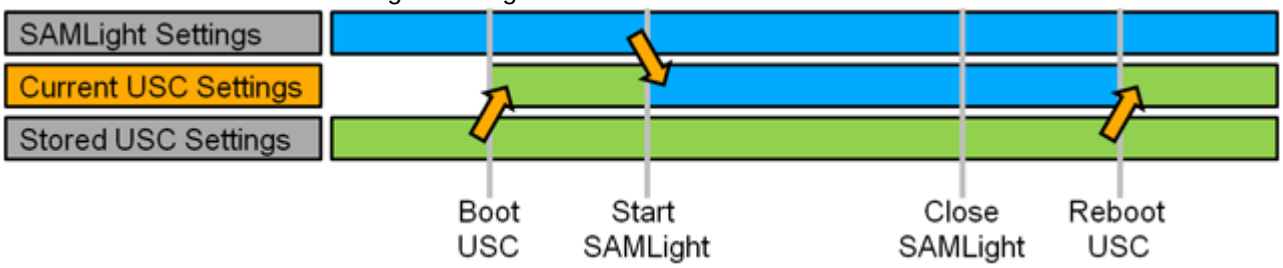


Figure 277: Common mistake: SAMLight and USC settings without storing leading to change of settings after reboot

The current USC settings on the USC card change with different events, see figure 277:

- At boot of the USC card, the current USC settings are loaded from the stored USC settings.
- At SAMLight start, the settings of SAMLight overwrite the current USC settings.
- These SAMLight settings remain as current USC settings even after SAMLight has been closed.
- **But, when the USC card is re-powered, the current USC settings are loaded from the stored USC settings. Then, if the current USC settings do not match the SAMLight settings, the behavior of the USC card could be different than before.**

14.3 Job processing

In general the job preparation will start with the editor which allows to setup fixed and variable geometries together with marking parameters and process flow elements. Also more general operation modes like the laser source, scanner field settings including correction file assignments and other modes like marking on the fly will be defined there.

The prepared job can be saved in SJF format for later reload or it can be converted to UNF format which can be executed by the flash. Uploading and executing of SJF files itself is not possible. By default the editor will attach the SJF file to the UNF file. This allows a reverse processing of a generated UNF file to its original SJF file which finally then can be reimported into the editor.

The general dialog to achieve this functions will be started by *Menu bar → Extras → Flash*.

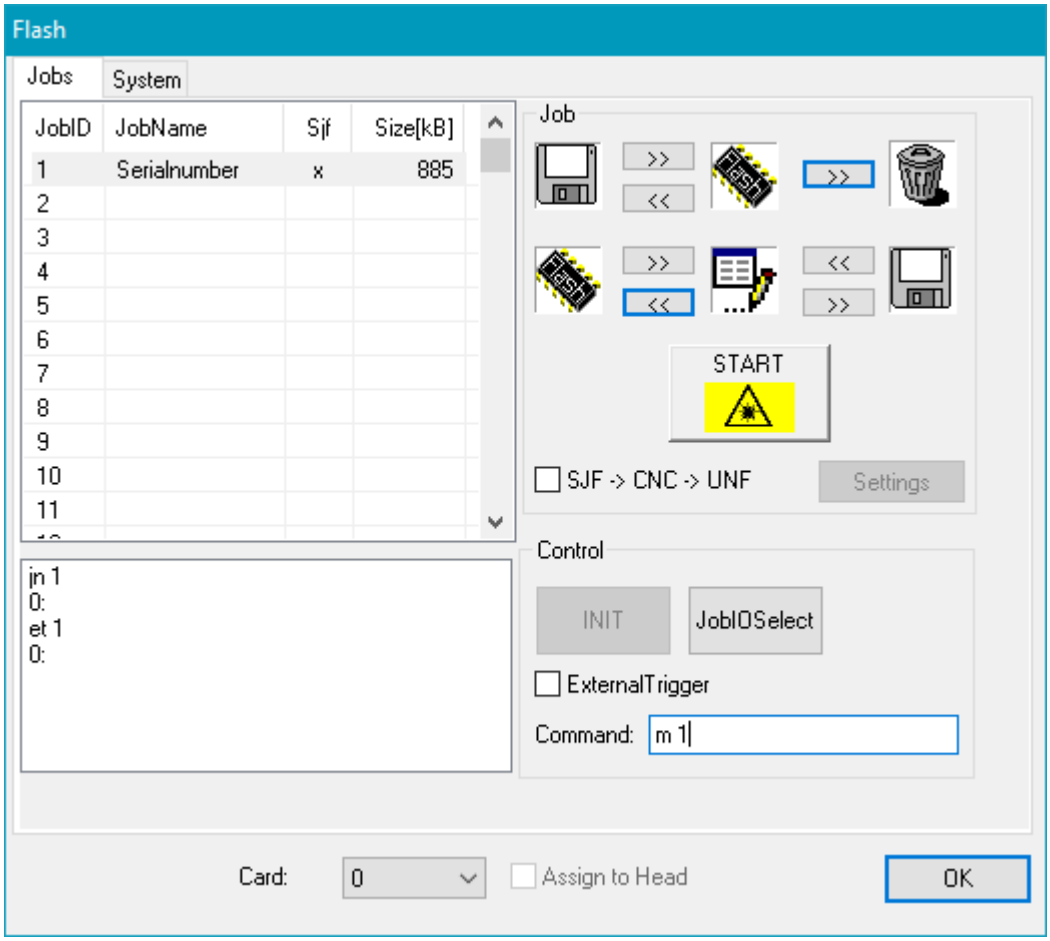


Figure 278: Flash Main Dialog

14.3.1 Up/Download

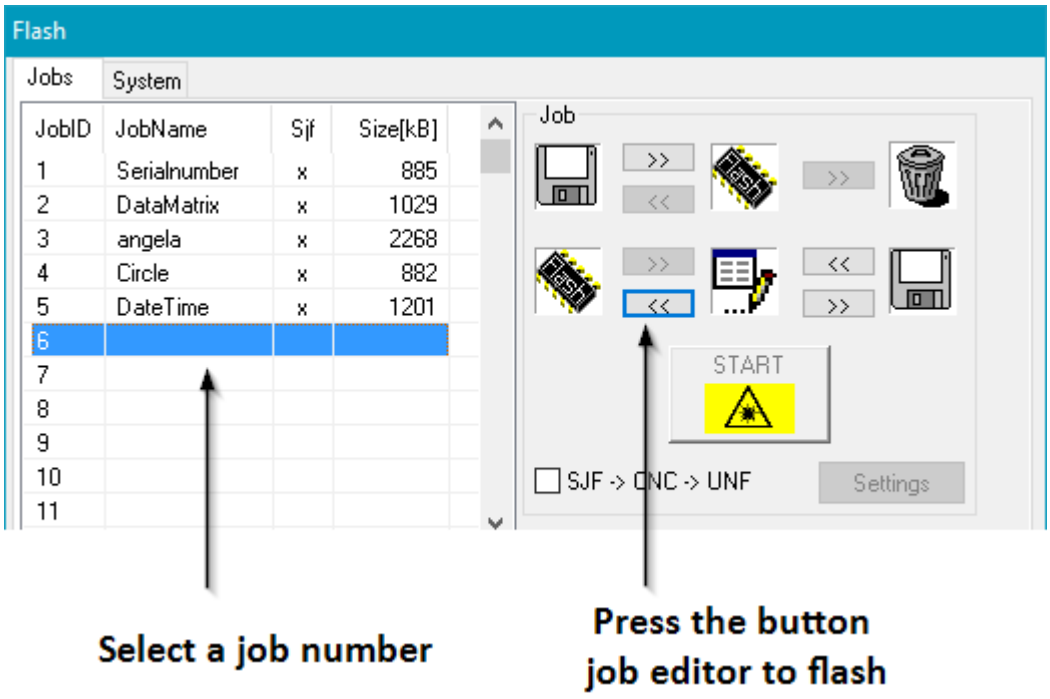


Figure 279: Upload from editor to flash

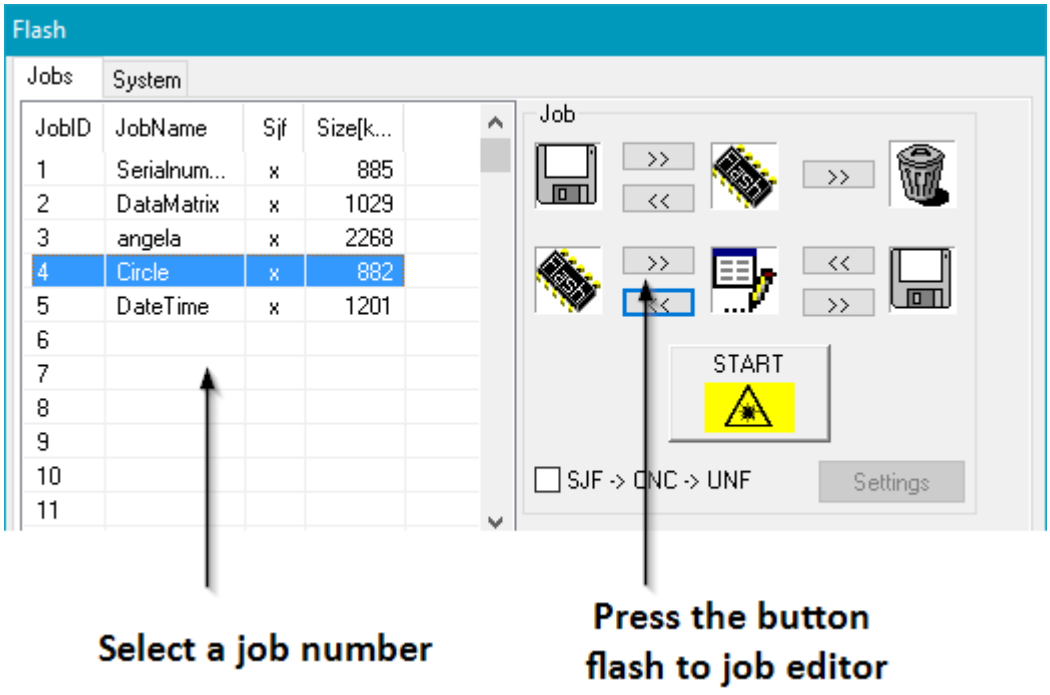


Figure 280: Download to editor from flash

14.3.2 Execution

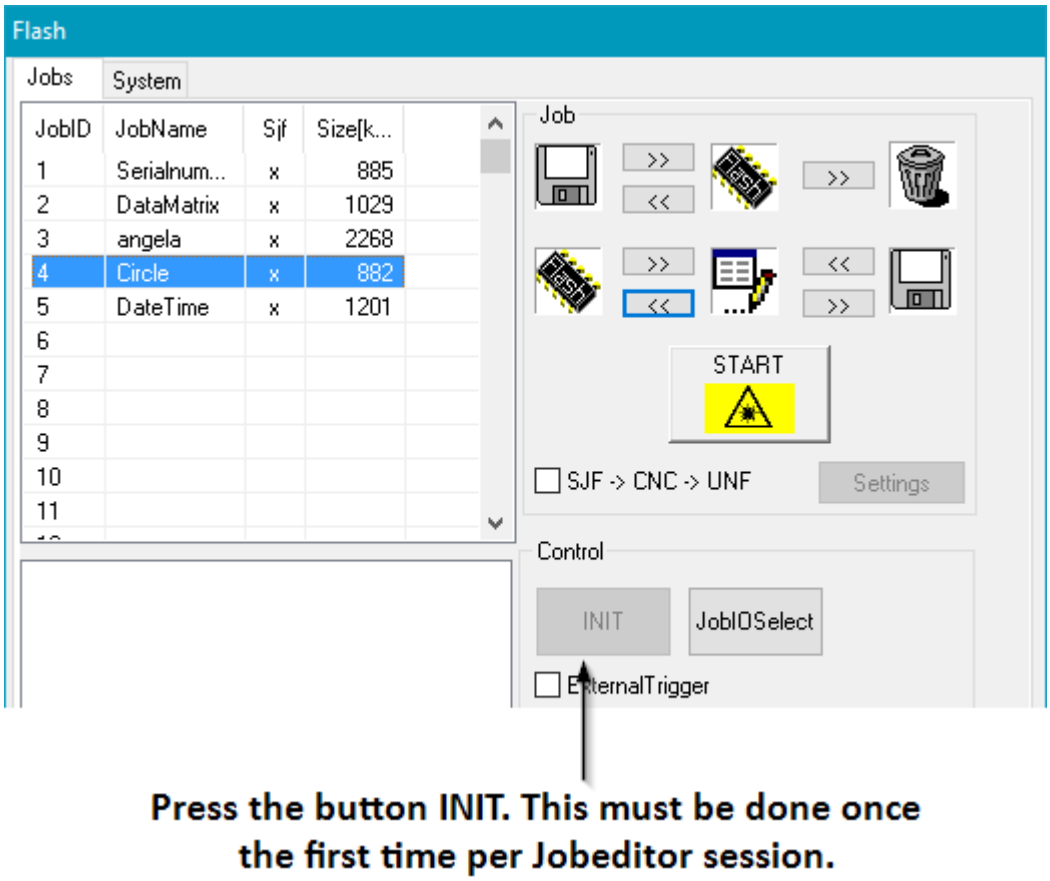


Figure 281: Initialisation of the Flash

The INIT button is only used by USC-1 + FEB-1, it is not used by USC-2 cards so it is grayed out.

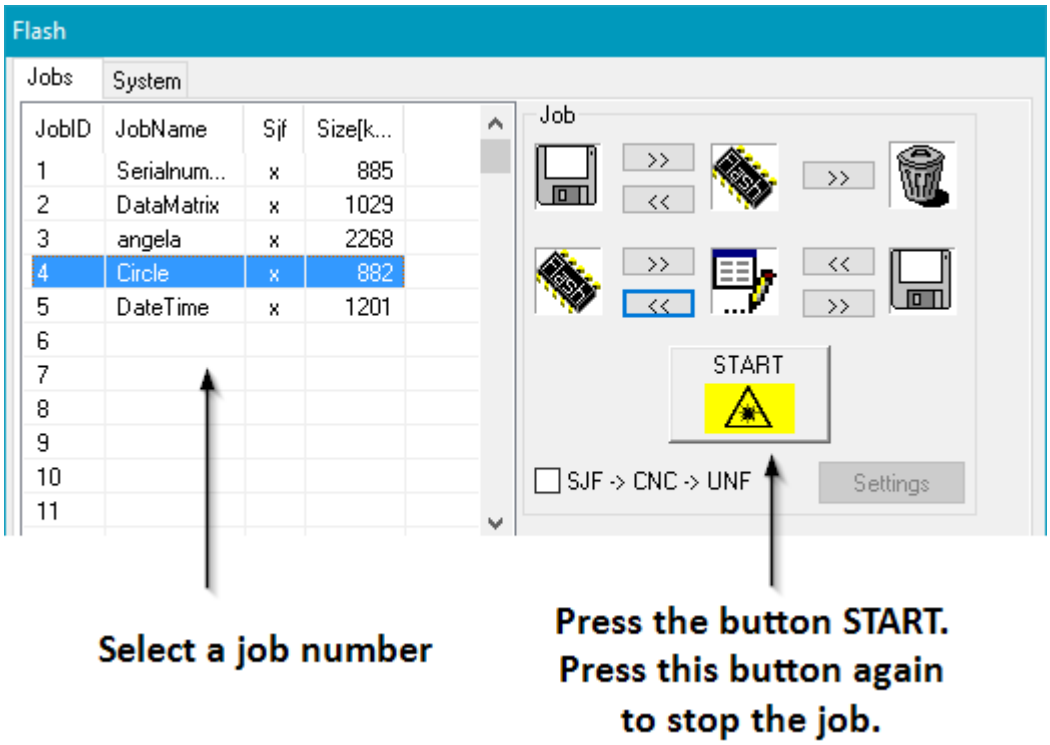


Figure 282: Start/Stop the job

14.3.3 Flash Job IO Selection

By clicking on the button "JobIOSelect", the Flash Job IO Select mode can be activated in Flash mode. Then jobs can be loaded and marked via the OptoIn bits of the USC-2 card. See chapter Flash IO Job Selection.

14.4 System

The system property page is used to initialize the flash memory, update the system section and to define the boot configuration in stand alone mode. It also shows some status flags for diagnostics purpose.

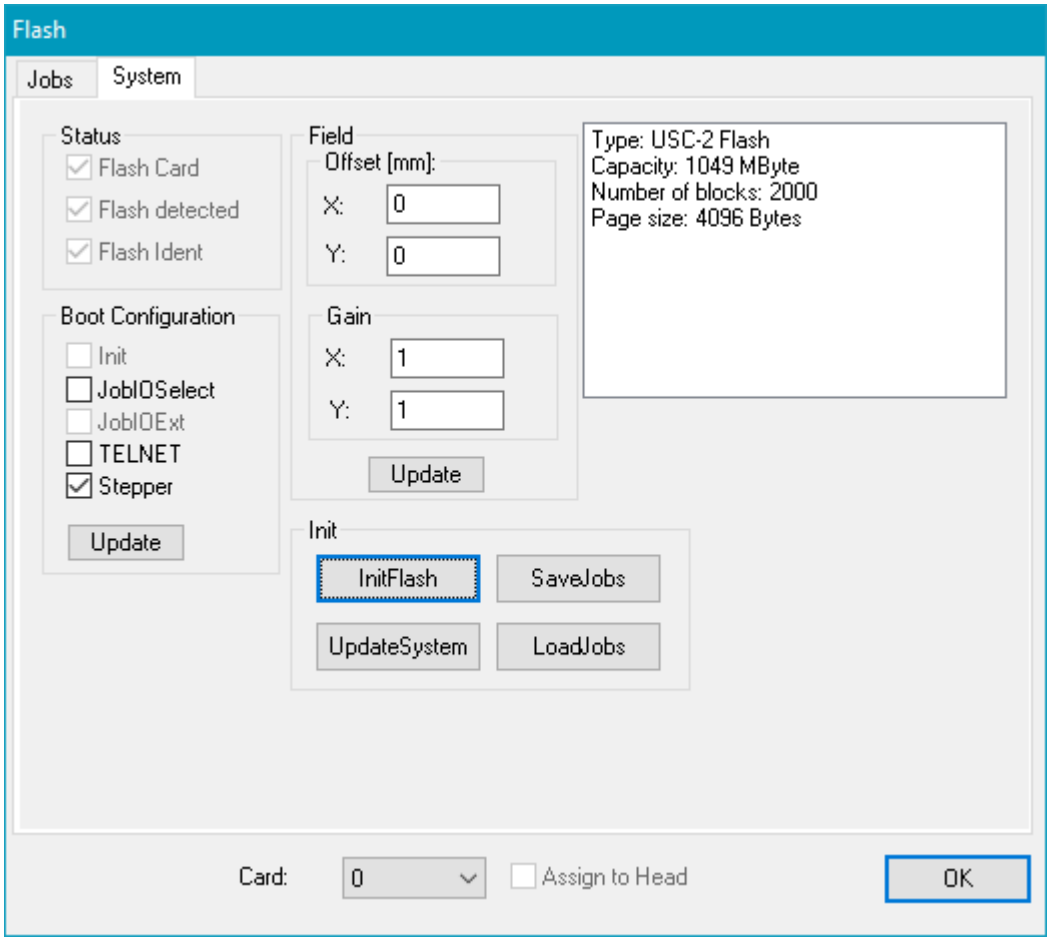


Figure 283: Flash System Dialog

Status:

Flash Card: Shows that a flash card hardware is detected.

Flash detected: The flash software runs and has returned the flash type.

Flash Ident: The flash was successfully initialized and has valid information stored.

Boot Configuration:

InitFlash: Flash card starts and makes an auto initialization. Asks if you want to keep your jobfiles (by using *SaveJobs* and *LoadJobs*). All other settings on the USC-2 card will be restored to default values. Please save your settings to USC-2 again by clicking *Menu bar → Settings → System → Optic → Advanced → Store*.

JobIOSelect: The Flash card starts in *JOBIOSelect* mode.

JobIOExt: In combination with JobIOSelect, this mode enables the inputs [0..7] of the Extension connector, so that 255 jobs can be selected. Only available with USC-2 card.

TELNET: It is possible to send commands to the card via TELNET Port 23. Only available with USC-2 card.

Stepper: Shows if stepper mode is used. You can enable it in USC server (visible mode).´

Click Update and repower the USC-2 card for changes to take effect.

Init:

InitFlash: Erases the complete flash. All data is lost.

UpdateSystem: Updates the flash system block only. Job information is preserved.

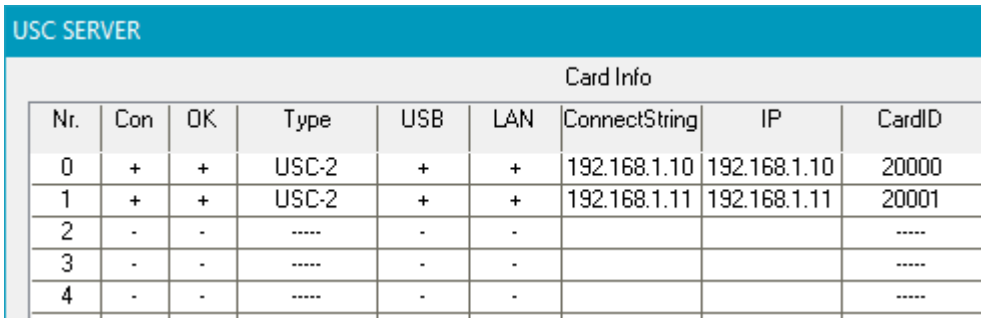
SaveJobs: Saves the jobs stored on the flash to <*SCAPS*>*jobfiles**save_jobs_head_X_YYMMDD*\, while X is the head number and YYMMDD the date.

LoadJobs: Loads the jobs saved by *SaveJobs* to the flash.

14.5 MultiCard

The feature 'MultiCard' is included in the license 'Flash' and is not compatible with MultiHead. The most common application is to run up to six USC-2 cards in standalone mode (flash mode) and use one SAMLight license to prepare job files and do the flash job management of all cards.

1. Close SAMLight and start the server in visible mode (<SCAPS>\system\sc_usc_server /v).
2. Connect only one USC-2 card at a time via both USB and Ethernet.
3. Select the USC-2 card in the server and click on the USC2-LAN button and define the Ethernet settings. Leave the dialog with the Connect button. After that you can check the '+' sign in the LAN column of the usc server.
4. Repeat steps 2 and 3 for each USC-2 card.
5. Unplug all USC-2 cards.
6. Define the order of the USC-2 cards in '<SCAPS>\system**sc_usc_card_ids.txt**'. The order of the dongle IDs in this file corresponds to the head (card) number as you can find it in the mark- and flash-dialog.
7. The IP addresses of every USC-2 card should now appear as a connect number in '<SCAPS>\system **sc_usc.cfg**'. The server attempts to connect only to these IP addresses. The order of the IP addresses in this list does not matter.
8. Enable the checkbox 'MultiCard' in sc_usc_server (<SCAPS>\system\sc_usc_server /v).
9. Connect all USC-2 cards via Ethernet, double check the order of the cards in the visible server.
10. Go to <SCAPS>\tools\sc_setup.exe → HardwareSettings and configure each USC-2 card individually and save the changes to the settings file.
11. Do not forget to click on the Store button in the Driver Settings for each card to save the settings to the USC-2 cards as well.
12. Finally the setup is done. Any card connected to the sc_usc_server can be controlled in different ways:
    a. SAMLight: Marking directly in SAMLight (not in flash mode)
    b. SAMLight: Flash job management in Extras → Flash
    c. SAMLight: Flash job management with Client Control Interface calls
    d. Telnet: Use the FTP server to add jobs, use Flash Control Interface calls to manage flash joblist
    e. sc_usc_server: Flash job management with the Save, Load and Delete buttons in '<SCAPS>\system \sc_usc_server /v → Flash'



Nr.	Con	OK	Type	USB	LAN	ConnectString	IP	CardID
0	+	+	USC-2	+	+	192.168.1.10	192.168.1.10	20000
1	+	+	USC-2	+	+	192.168.1.11	192.168.1.11	20001
2	-	-	-----	-	-			-----
3	-	-	-----	-	-			-----
4	-	-	-----	-	-			-----

Figure 284: MultiCard setup in sc_usc_server

```
Connect0=192.168.1.10       20000
Connect1=192.168.1.11       20001
Connect2=                   0
Connect3=                   0
```

Figure 285: Connect strings in sc_usc.cfg **Figure 286: Head (card) assignment in sc_usc_cards_ids.txt**

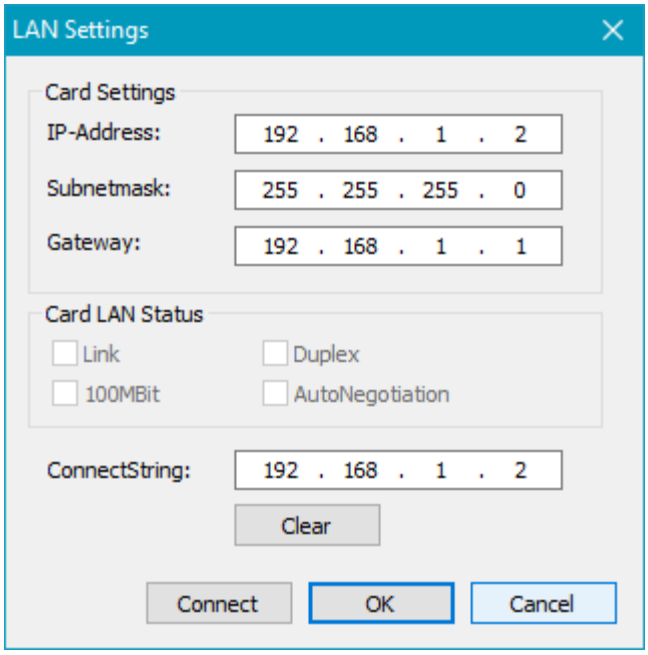


Figure 287: LAN Settings Dialog

Card Settings:

Here you can set the IP address, subnetmask and gateway of the USC-2 card.

ConnectString:

The IP adress that is defined here will be listed in the file 'sc_usc.cfg'. The server will try to connect to those IP adresses only which are listed in this file.

Connect:

By clicking 'Connect' the sc_usc_server will write the card settings to the connected USC-2 card.

15 Multiple Heads

The Option MultiHead allows to control up to six scan heads simultaneously. There are two different modes:

- The "MultiHead with multiple cards" mode requires more than one scanner controller card, each card controlling a scan head and a separate laser. This mode allows the marking of different data on the two or more heads. To use this feature, the optional SAM software license *MultiHead* is required. All heads are sharing one job file which is split on the several scan heads.
- The "Two heads with one card" mode allows to use two heads connected on one scanner card (if the scanner card is able to do this). Both heads are marking the same data from the same job file at the same time. This mode is included in the SAMLight or the SAM Standard Components, no extra software option is required but for the USC-2 card the hardware option *Head 2* is needed.

15.1 Option MultiHead

The Option MultiHead allows to build up scanner applications with a simultaneous vector output to up to six scan heads. In this case there will be one job file for all scanheads.

For this simultaneous output and the installation of more than one (up to 6) scanner driver cards, the MultiHeadOpticModule license is required. If the license for the MultiHeadView is present, the View2D shows all installed working areas with the overlapping region. The data is edited as there would be one big output. The MultiHeadView provides automatic splitting functionality to see on which head the vectors belong.

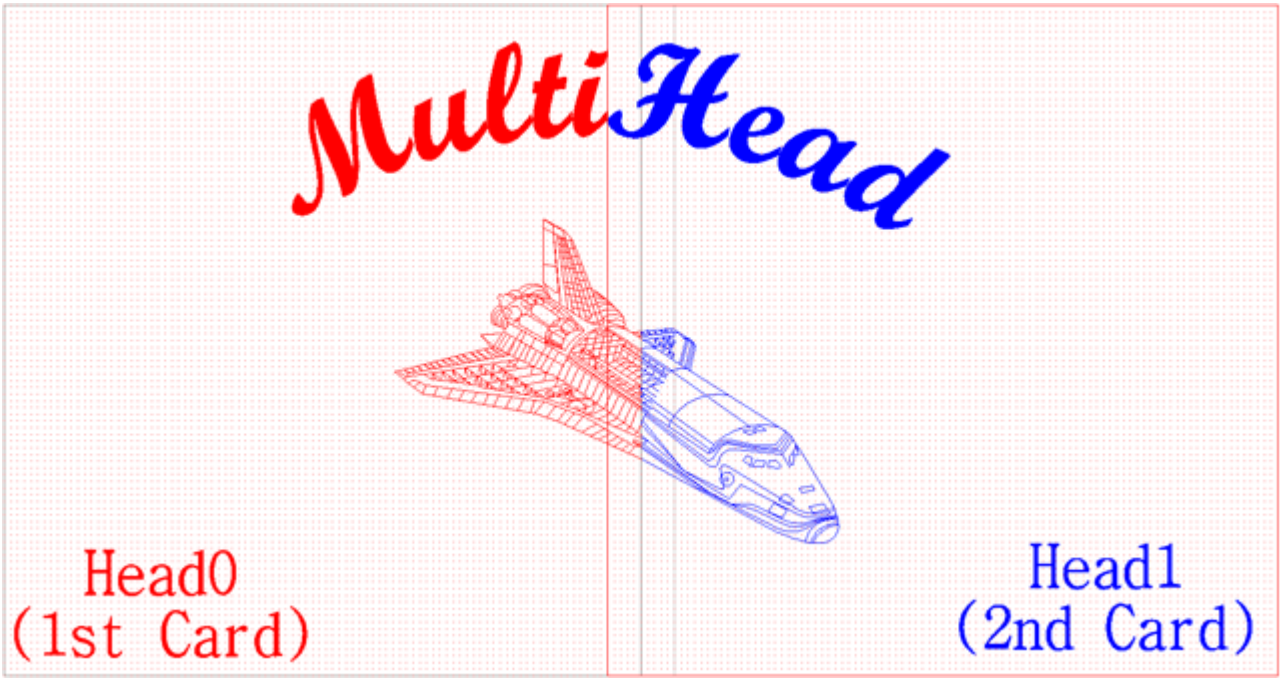


Figure 288: Job with two different working fields for two different scan heads

External trigger signal (MultiHead):

Trigger mode: Each card needs an trigger input signal. All cards must have finished their marking before the next job execution can start.

MarkDialog with "ExternalTrigger": This is not the same as trigger mode. Here the trigger input for all cards is card 0.

ControlObjects (MultiHead): When a ControlObject is part of the job all cards wait for all cards to finish their previous marking process then the ControlObject is executed.

ScMotionControl: Step and direction signals are generated only on card 0.

ScSetOutput, ScOverride, ScSetAnalogOutput: These control objects act on each card.

ScWaitForInput, ScWaitForTrigger: After all cards have finished their previous marking process, each card is waiting for an input signal on itself.

Motion Control: the step and direction signals for motion control devices (stepper type 8 and 14) are only provided at the first card (Head0). The necessary settings are defined in the chapter Motion Control Settings.

Marking on the fly (MOTF): the encoder signal for the MOTF compensation has to be provided to each card. The corresponding MOTF settings has to be defined for each card as well. For more information please also refer to your scanner controller card manual.

15.1.1 Installation

There are four steps to getting started to use multiple heads with multiple cards:

1) Activating software with a password containing the MultiHead license
2) Definition of the hardware settings in the Setup Tool
3) Definition of optic settings
4) Definition of automatic vector splitting in the View2D

15.1.1.1 Password

The software is delivered together with a dongle and a password. To use the multi head components, the password has to include these components. If there is a previous installation on the PC, the following steps are necessary:

1) Install the new software under the same installation folder as the old one. Default folder is C:\scaps\sam2d \.
2) Delete or rename the file sc_##_"dongle_number".scl.
3) During the first start of the SCAPS software there appears a dialog for typing in the new password.

15.1.1.2 Setup Tool

The settings for the software are saved in a *.sam file located in the folder <*SCAPS*>*system*\. For the Standard2D software the name of this settings file is *sc_settings.sam*. For SAMLight it is *sc_light_settings.sam*. This file also stores the head count and the type of the installed cards. The file can be edited with the *sc_setup.exe* program located in <*SCAPS*>*tools*\. To do this it is necessary to close all SAM applications before.

Starting *sc_setup.exe* and selecting menu point *Hardware Settings* shows following dialog.

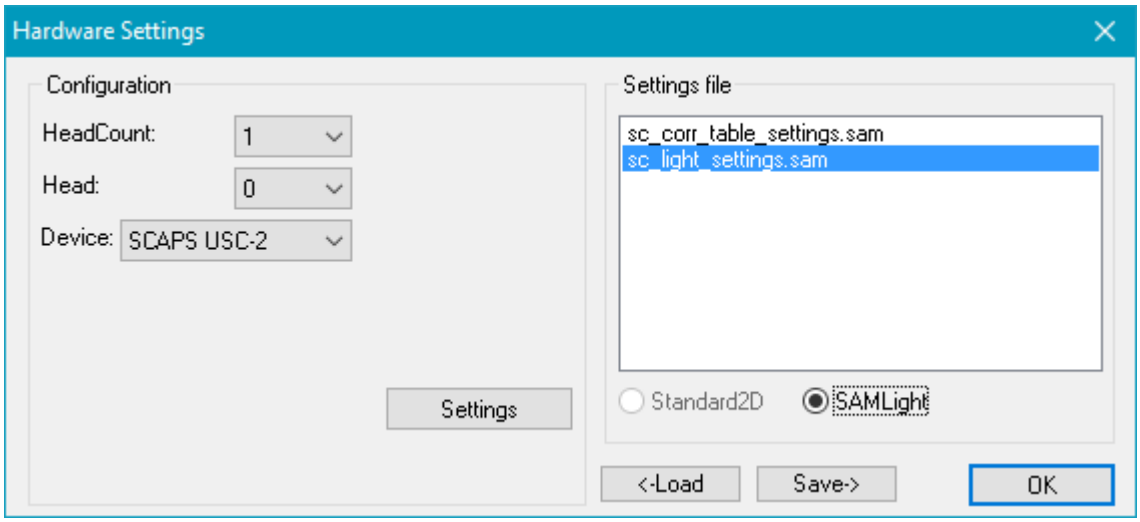


Figure 289: Hardware Settings Dialog

To change your settings, the following steps are required:

1) The software looks for all *.sam files located in <SCAPS>\system\.
2) Select your settings file and press <-Load.
3) Define the total head count.
4) Select the head and the installed device for this head. Repeat this step for all installed heads.
5) Press the Settings button to define the optic settings for the heads.

15.1.1.3 Optic Settings

Get this dialog by clicking on *Settings* at the *Hardware Settings* dialog. See chapter Setup Tool.

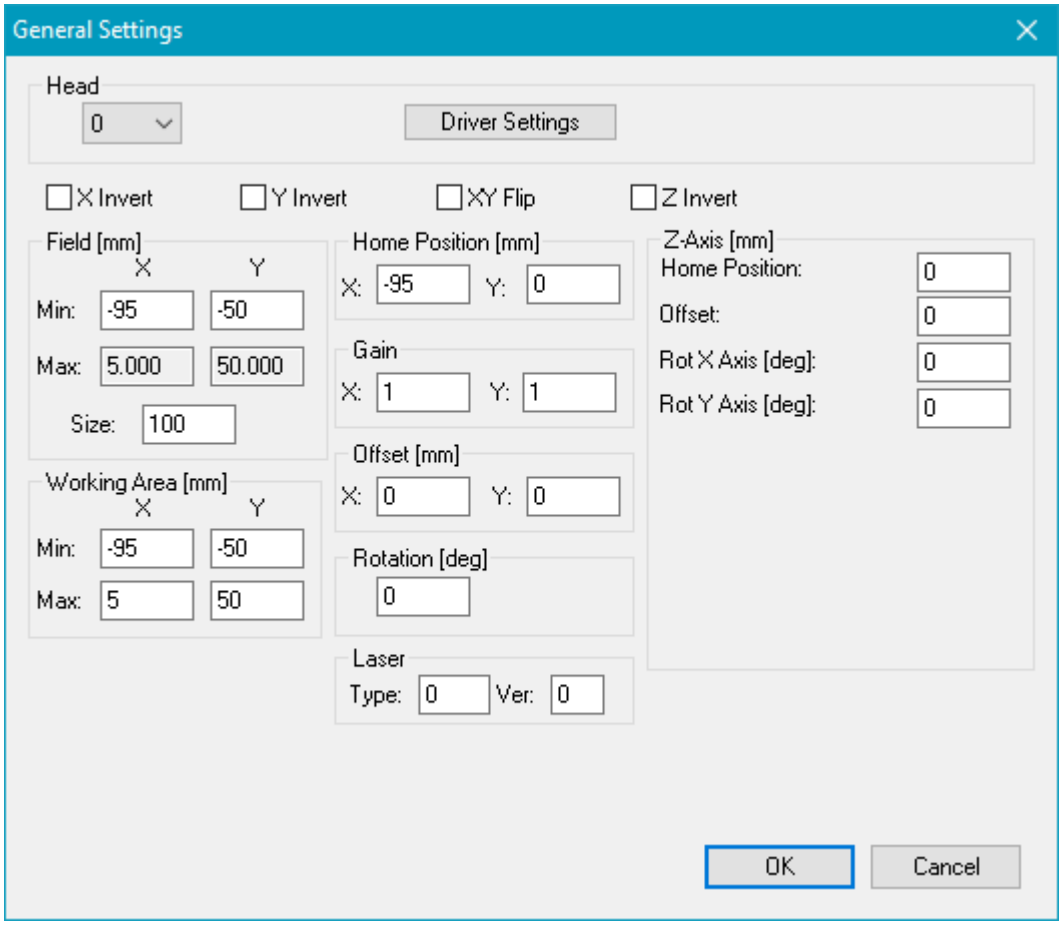


Figure 290: General Settings for Head 0

Select the head with the box at the top. Then define the field and the working area for this head. The values shown in the dialog above define a field with 100 mm size for each head. The total center in the middle of the two fields and an overlapping area of 10 mm in x direction.

The corresponding values for head 1 are shown below:

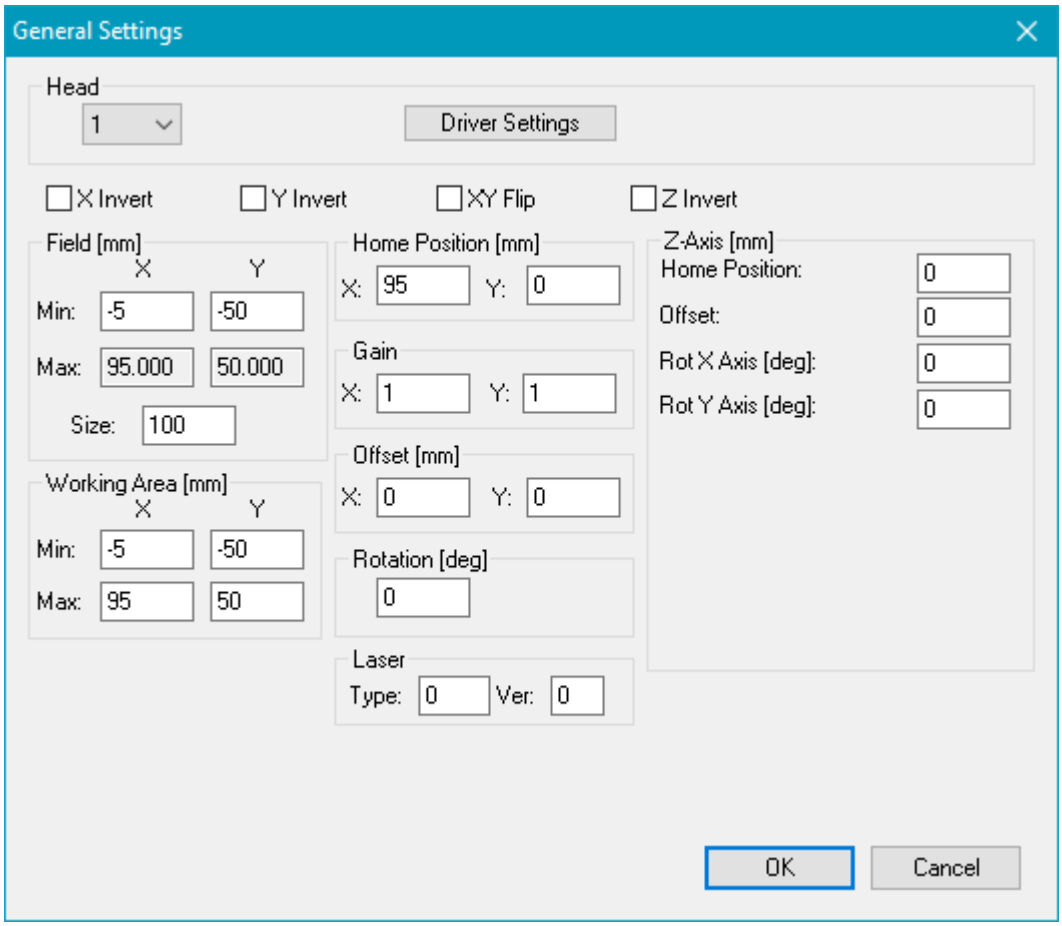


Figure 291: General Settings for Head 1

When the field and the working area have been defined, the driver specific settings have to be defined for each head. Select Head 0 and press button Driver Settings. Here define the specific settings as correction file location, the DLL files etc. for the scanner card. Repeat this step for all installed heads/cards. The optic specific setting can also be changed within the software by selecting *Menu bar → Settings → System → Optic*.

In the View2D for MultiHeads you will see the resulting working area for both heads.

15.1.1.4 View2D

If there are more than one head installed, the View2D provides some additional functionality. The most important is the split function for precalculation of the vectors and defining a head for each vector. When the software starts, the View2D shows the working areas for all installed heads and the overlapping region. See the screenshot in the chapter Multiple Cards.

The *split* function can be called directly from the context menu (click right mouse button in the View2D) → *MultiHeadSplit*. There is also an *AutoSplit* function available. It can be activated within the *ViewProperties* dialog. This dialog can be reached by the context menu (right click of your mouse) and selecting *ViewProperties* or by a direct call to the *ViewProperties*:

Figure 292: Context menu (left) and properties (right) of View2D for MultiHead

Checking the *AutoSplitMode* recalculates and splits the vectors after each change of the job. For not too complex jobs this option is very helpful. The head color mode shows the drawings in head specific colors. Once the *Head Color Mode* checkbox is activated, the colors in the view2D for MultiHeads do not tell which pen is used but instead the colors mean which head is used for marking the entity.

NOTE

In case of an error message "Galvos out of range" when the marking should start, the Splitting has not been activated. To solve the problem, right click in the View2D and either do a MultiHeadsplit or activate the AutoSplitMode in ViewProperties → MultiHead.

15.2 Option Head2

This mode allows to send the same data to two heads through one scanner controller board.

NOTE

The scanner controller has to have this feature.

Example:

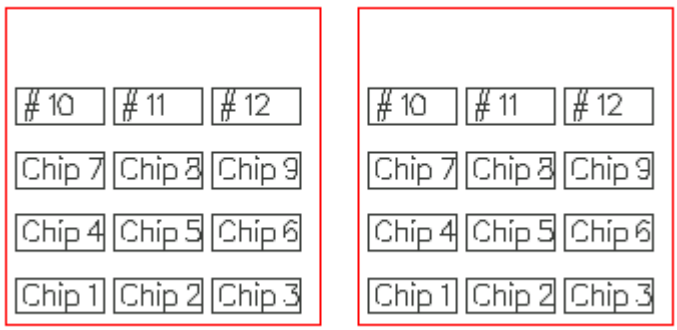


Figure 293: Example for Head2, both scan heads are marking the same vectors

15.2.1 Installation

Here the Settings for the two heads with one card for an USC-2 card are described. The following dialog is achievable via *Menu bar → Settings → System → Optic*.

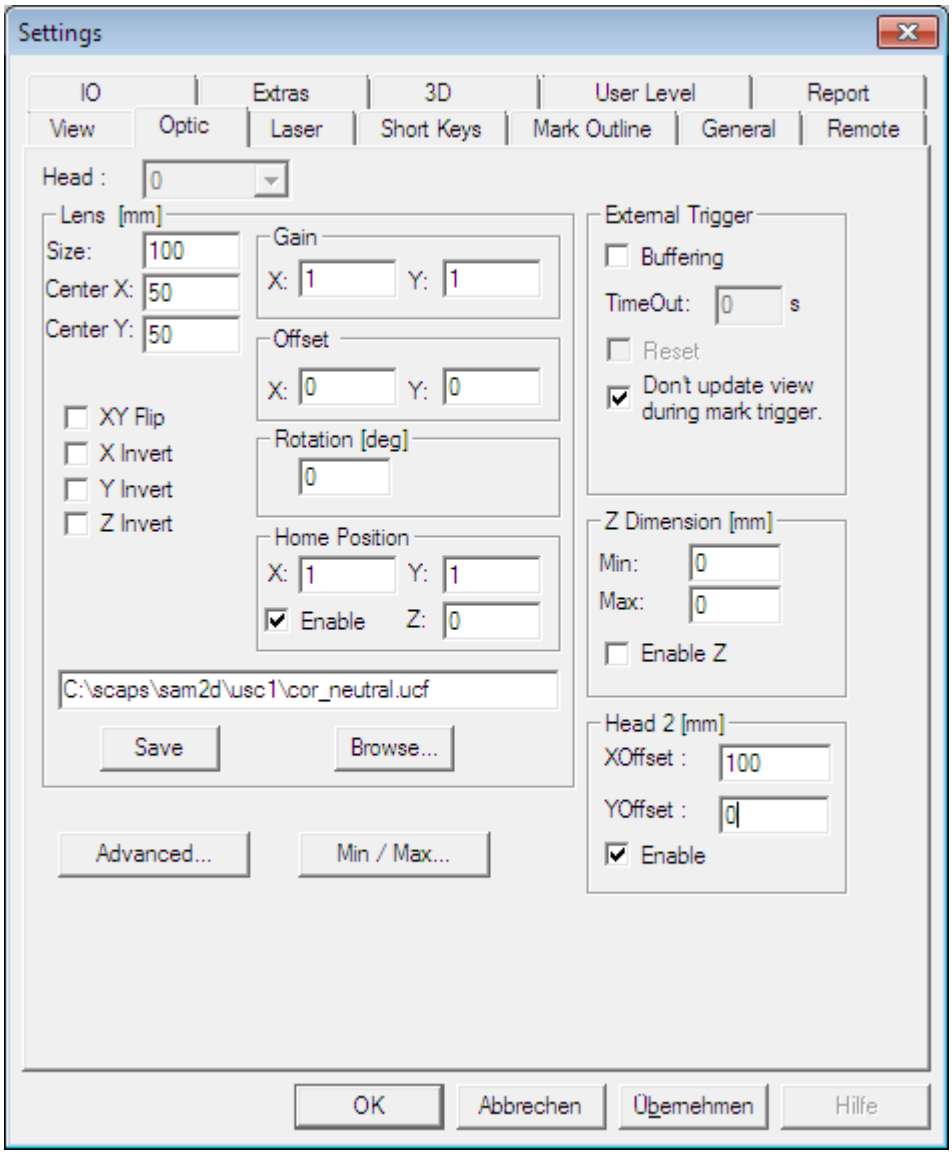


Figure 294: Optic Settings for Head 2

Step 1: Click on *Advanced... → Correction, Settings...* for the correction file dialog. From the drop down box select *2* and activate the *Enable* checkbox right from it. Now click on *Browse* and select the appropriate correction file for the second scan head. After doing this the second scan head is activated.

Step 2: This step is not necessary. In the lower right corner find the field *Head 2 [mm]*. Here enter the offset values for the second head in X and Y-direction. Finally enable the checkbox *Enable* to activate a *X* and *Y Offset* of the second scan head which will only affect the View but not the scanner output.

15.2.2 Fixed Job Offset

It is possible to define a *JobOffset* for the job that is used for the output to the secondary head. If the job offset is the same as the *Secondary Head Offset*, the output takes place at the same relative position inside the two fields. The maximum *Offset* in one direction is ±30000 bits of 65536. That corresponds to ±45 mm of a 100 mm working area or 45%.

Inside SAMLight, the *JobOffset* can be defined within the dialog *File → JobProperties*:

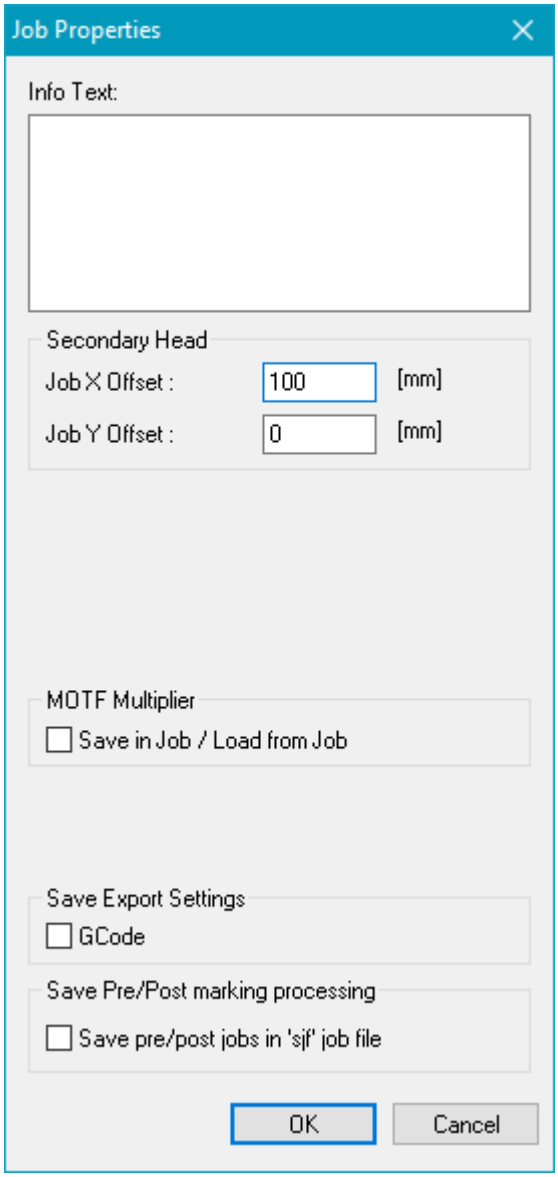
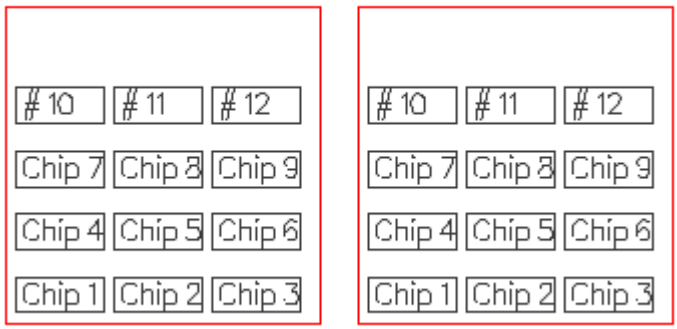
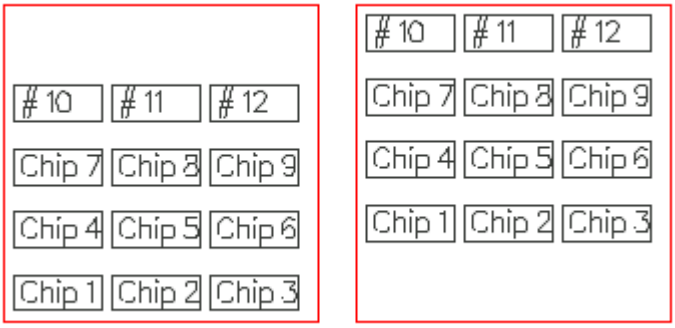


Figure 295: Job Properties Dialog

Inside the View, the job is displayed as follows:



The objects are on the same relative position within the two heads as the Job Offset is the same as the Head Offset. If we change the JobOffset to x = 100, y = 20, we get the following:



The JobOffset is saved within the sjf file.

15.2.3 Variable Entity Offset

For the secondary scan head, a relative offset to the fixed job offset with job properties can be defined with inserting a *ScSetSecondaryHeadOffset* entity. The specified offset in this entity is applied to the subsequent entities in the entity list.

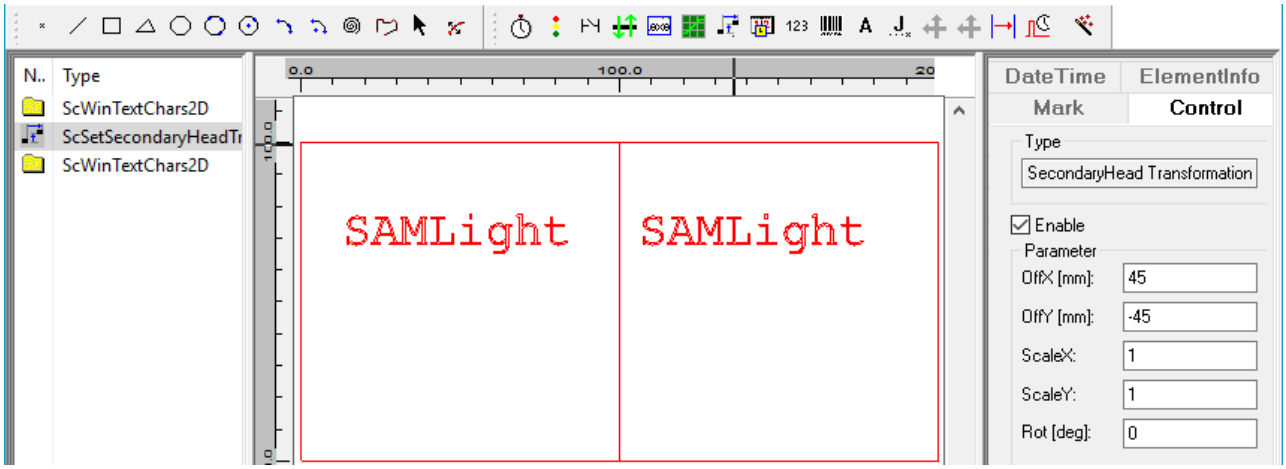


Figure 296: Secondary Head Offset

Click on the  button in the functionality object toolbar and a *ScSetSecondaryHeadOffset* entity is inserted in the entity list on the left. Select this new entity in the entity list and the property page *Control* on the right side becomes active. Click on it. To define an offset, the *Enable* checkbox has to be selected and the relative X and Y offsets have to be entered. Press the *Apply* button. Now a new entity, e.g. a copy of the entity before, is inserted in the entity list behind the *ScSetSecondaryHeadOffset* entity. When marking the entity list, the copy is marked now with the specified offset. A new subsequent *ScSetSecondaryHeadOffset* entity inserted into the entity list overwrites the specified offsets of the *ScSetSecondaryHeadOffset* entity before. Remember that the X and Y offsets of the *ScSetSecondaryHeadOffset* entity are always defined relative to the offset with job properties and that the total offset of *ScSetSecondaryHeadOffset* entity plus the offset with job properties has to be ≤ 45% of the working area.

15.3 MultiCard (USC-2/3 only)

The description of the feature MultiCard can be found in the Option Flash chapter.

16 Option FlatLense (USC only)

This option enables the Z data channel of the XY2-100 Interface of the USC card. It is required to control 3 axis scan heads. Even if this is an hardware option, it can be enabled by the FlatLense license for every USC card.

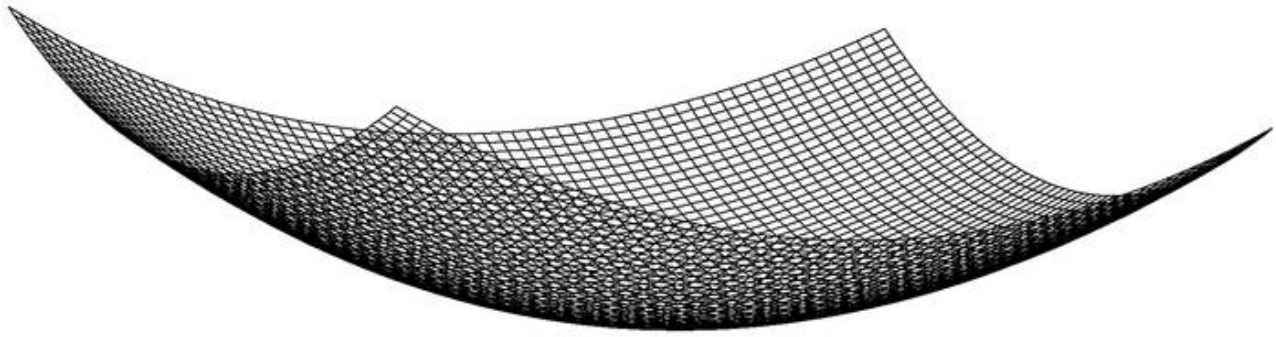


Figure 297: 3D UCF correction file

Flat surface marking without F-Theta lense: If the 3 axis scan head is used for marking on flat parts, the third scanner axis is used instead of a F-Theta objective to keep the laser focus constant on a plane surface, the option FlatLense is required and a 3D correction file has to be used which also contains Z bit values for every correction point.

Defocus: If you have a USC card with the option FlatLense (3D UCF correction file, no Optic3D) the software has no factor to calculate mm to bit. That is why the unit of pen defocus is [2^{16} bit / FieldSize] instead of defined unit in SAMLight.

Real 3D vectors marking: If the 3 axis scan head is used for marking on curved parts (real 3D vectors), both options FlatLense and Optic3D are required.

If a RTC card is installed Real 3D vectors marking is being enabled with the 3D option on the card and the Optic3D option.

17 Option Optic3D

This chapter describes the 3D marking functionality. For example how to set up marking on a curved surface.

17.1 Features

The following features are implemented:

- Wrap objects around 3D Surfaces
- Import of 3D DXF files (DXF Files Version 2)
- Translation of objects in X,Y and Z direction
- Rotation of objects around X,Y and Z axis
- Manual manipulation of points and 3D vectors

17.1.1 3D Surfaces

Toolbar: Enable the 3D surface toolbar in *Settings → System → View → Toolbars*. The toolbar is showing mode drop-down list settings button and enable the checkbox at the right from the cog wheel.

Figure 298: 3D Surfaces toolbar

Mode: Here the 3D Surface mode can be chosen. At the moment *Cylinder, STL file, Tilted Surface and Sphere* is available.

1. **Cylinder**
2. **STL Projection**
3. **Tilted Surface**
4. **Sphere**

17.1.1.1 Cylinder

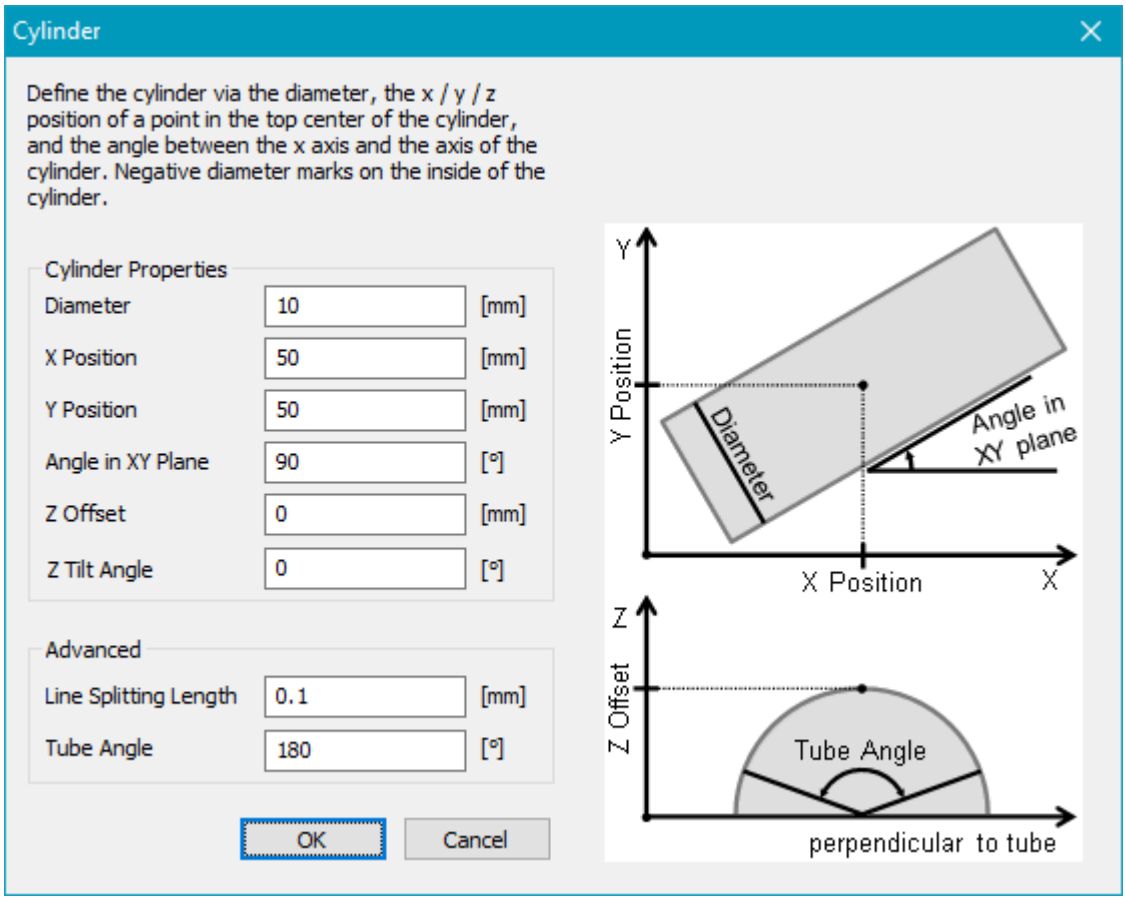


Figure 299: 3D Surfaces Cylinder dialog

Cylinder: Select Cylinder in the mode drop-down list and enable it. Click on  to open the settings dialog.

Cylinder properties: Defines the diameter and position of the cylinder.

Diameter: Defines the diameter of the tube. Negative values are also allowed to mark inside a

cylinder.

X, Y Position: Defines the X, Y position of the tube.

Angle in XY Plane: Defines the tube angle in the XY plane.

Z Offset: Defines the Z position of top of the tube. If the *Diameter* is negative this value defines the Z position of bottom of the tube.

Z Tilt Angle: Defines the Z slanting angle.

Advanced: Extended settings of tube marking.

Line Splitting Length: Every vector on the tube will be split into small sub vectors, to bend it on the tube.

Tube Angle: Defines the maximal angle of the tube for the vector bending. This parameter affects the distance of the blue dashed lines in the View2D.

View2D: Four blue dashed lines indicates the location and size of the cylinder in the View2D. The View2D shows the vectors not bent around the tube.

Outer blue dashed lines: Defines the unwrapped size of the *Tube Angle*. Only vectors between these lines will be bend.

Inner blue dashed lines: Defines the wrapped size of the *Tube Angle*.

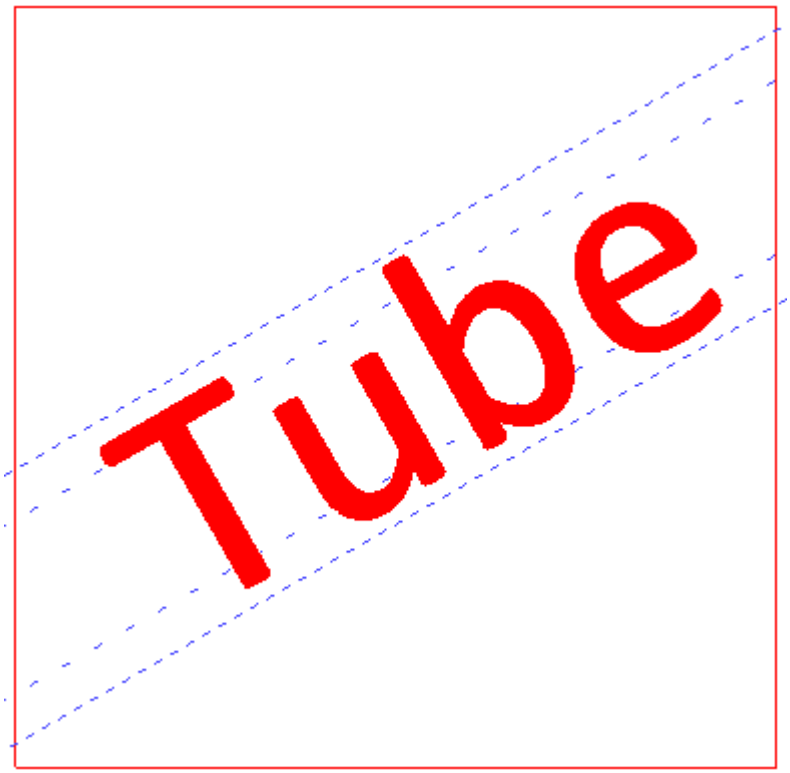


Figure 300: 3D Surfaces Cylinder View2D

3D preview: The bending of the vectors can be seen in the 3D view. Select at least one entity and click the *3D View* button  .

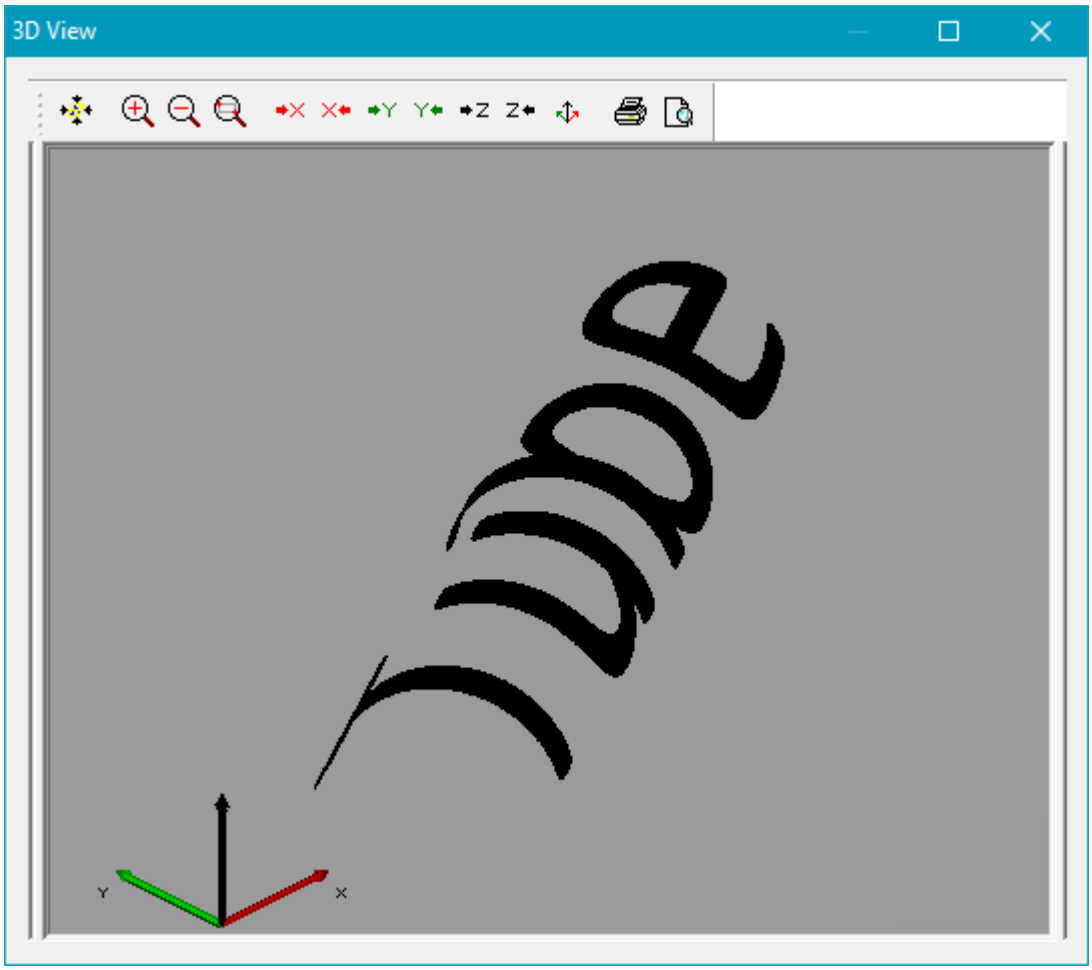


Figure 301: 3D Surfaces 3D preview.

17.1.1.2 STL Projection

Figure 302: 3D Surfaces toolbar

Click on the cog wheel to open the properties dialog:

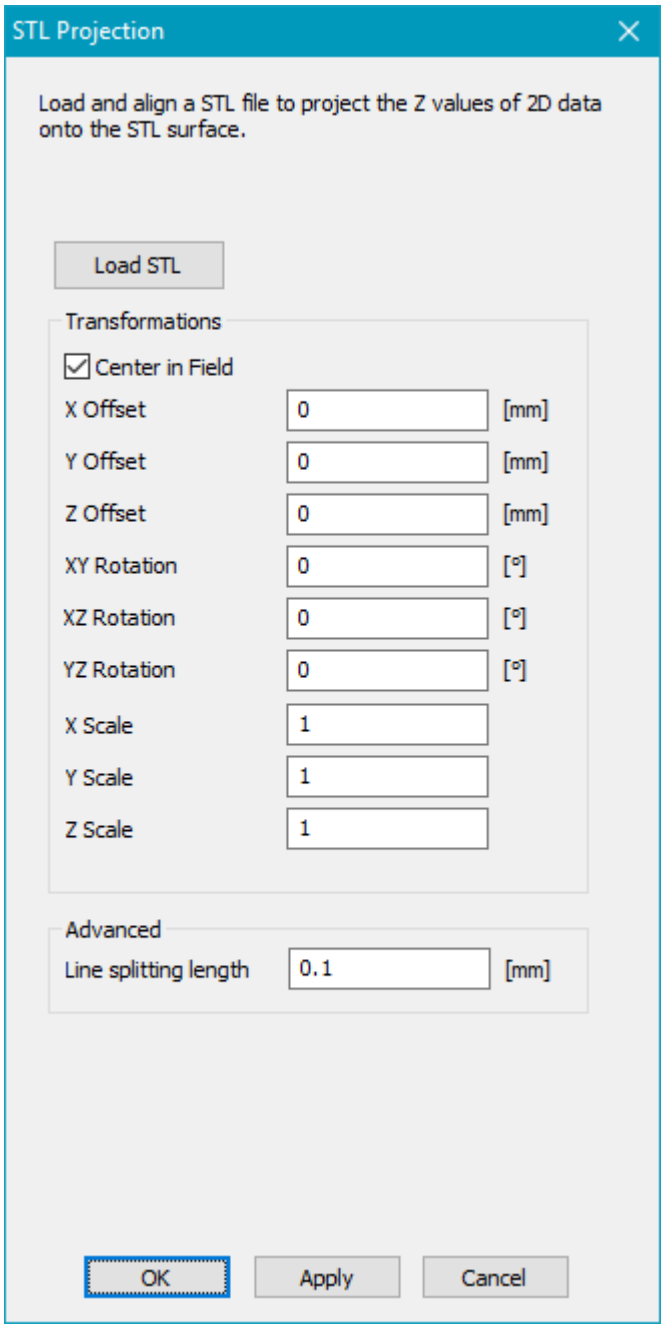


Figure 303: 3D STL property dialog

Click on "Load STL" to load a STL file. This file will be used as the surface on which the 2D vector job should be projected.

Enable the checkbox right to the cog wheel. Now the STL surface appears in the View 2D.

Below see an example where the 3D surface is a sphere and the 2D job is a simple square:

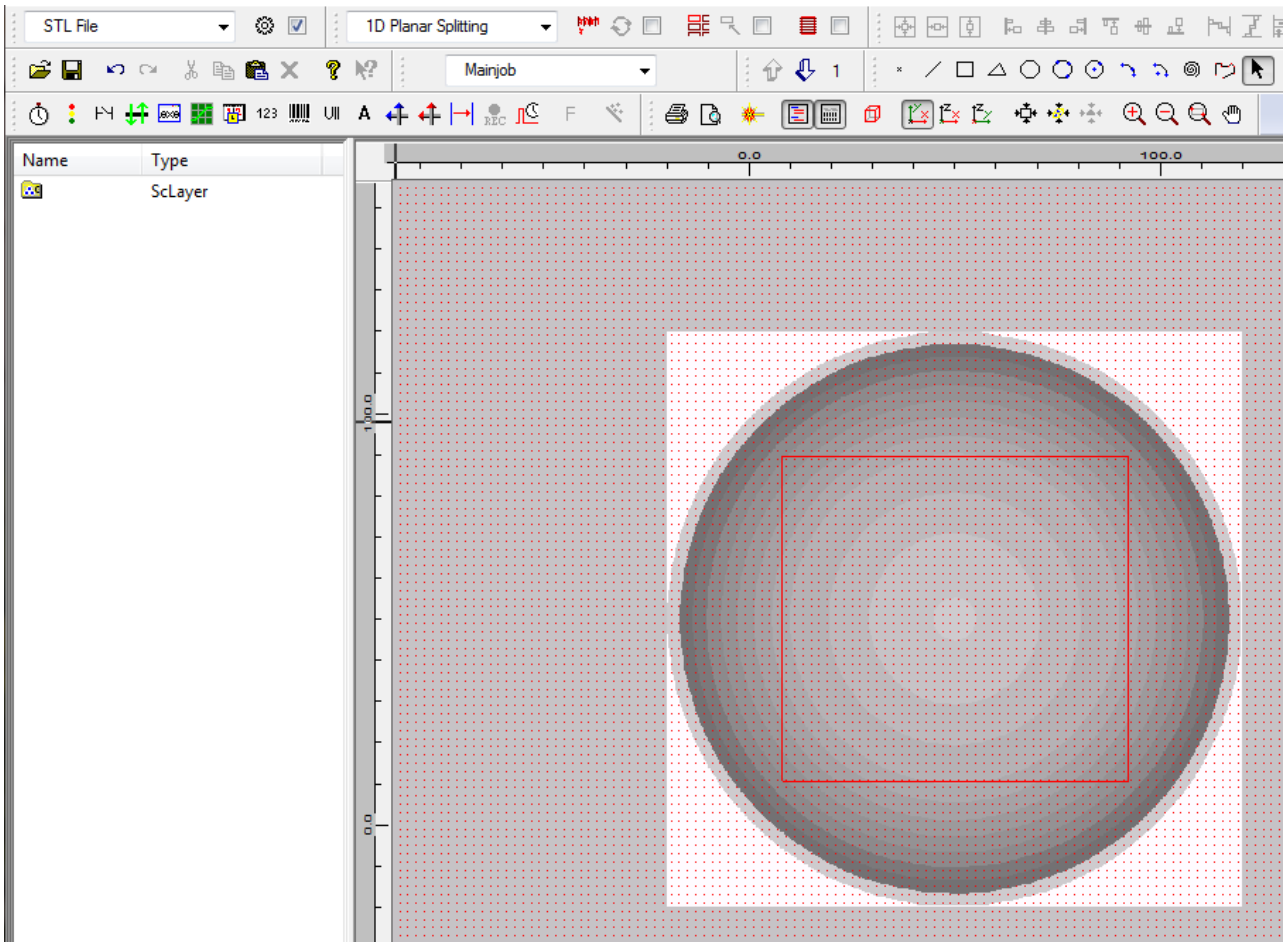


Figure 304: 2D Square projected on 3D STL sphere surface

When clicking on the Optic3D View icon  you can see how the square is projected on the sphere:

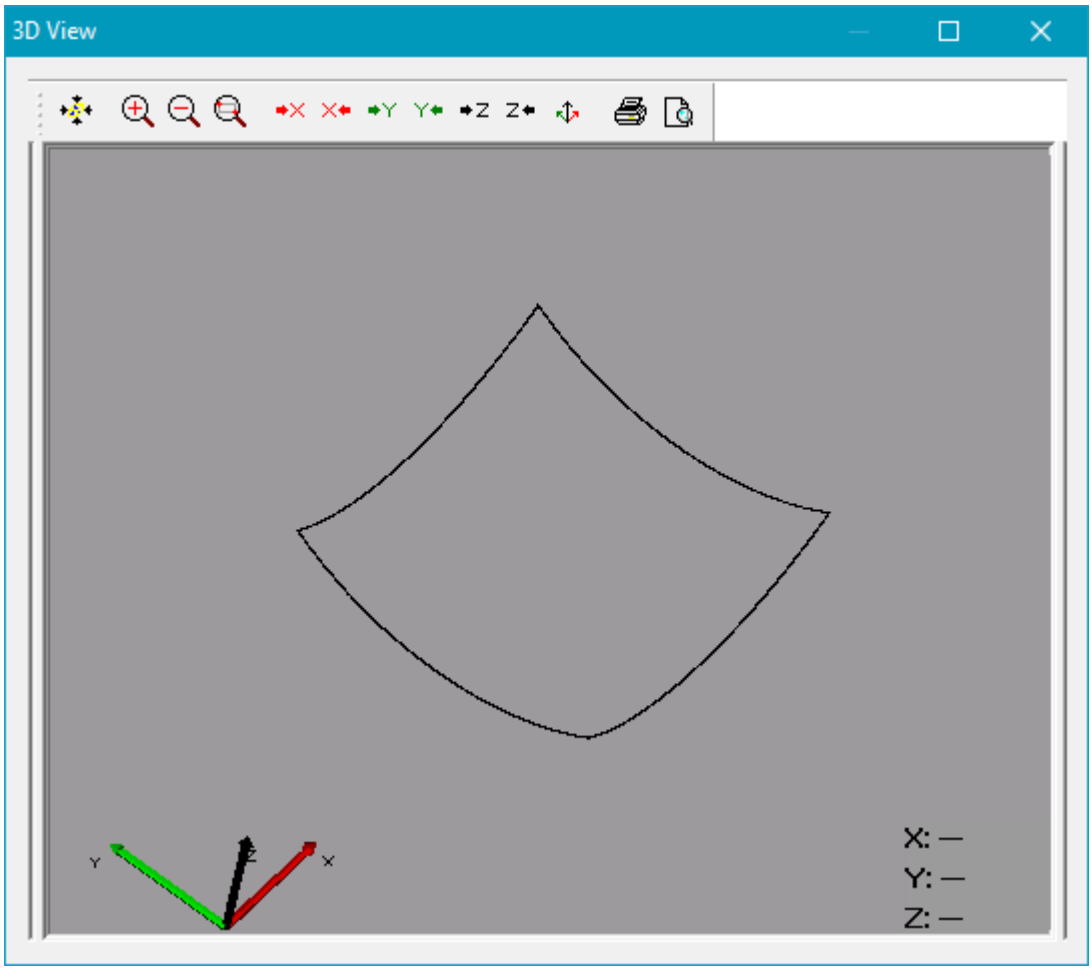


Figure 305: 3D View of the projected square.

17.1.1.3 Tilted Surface

Figure 306: 3D Surfaces toolbar

Click on the cog wheel to open the properties dialog:

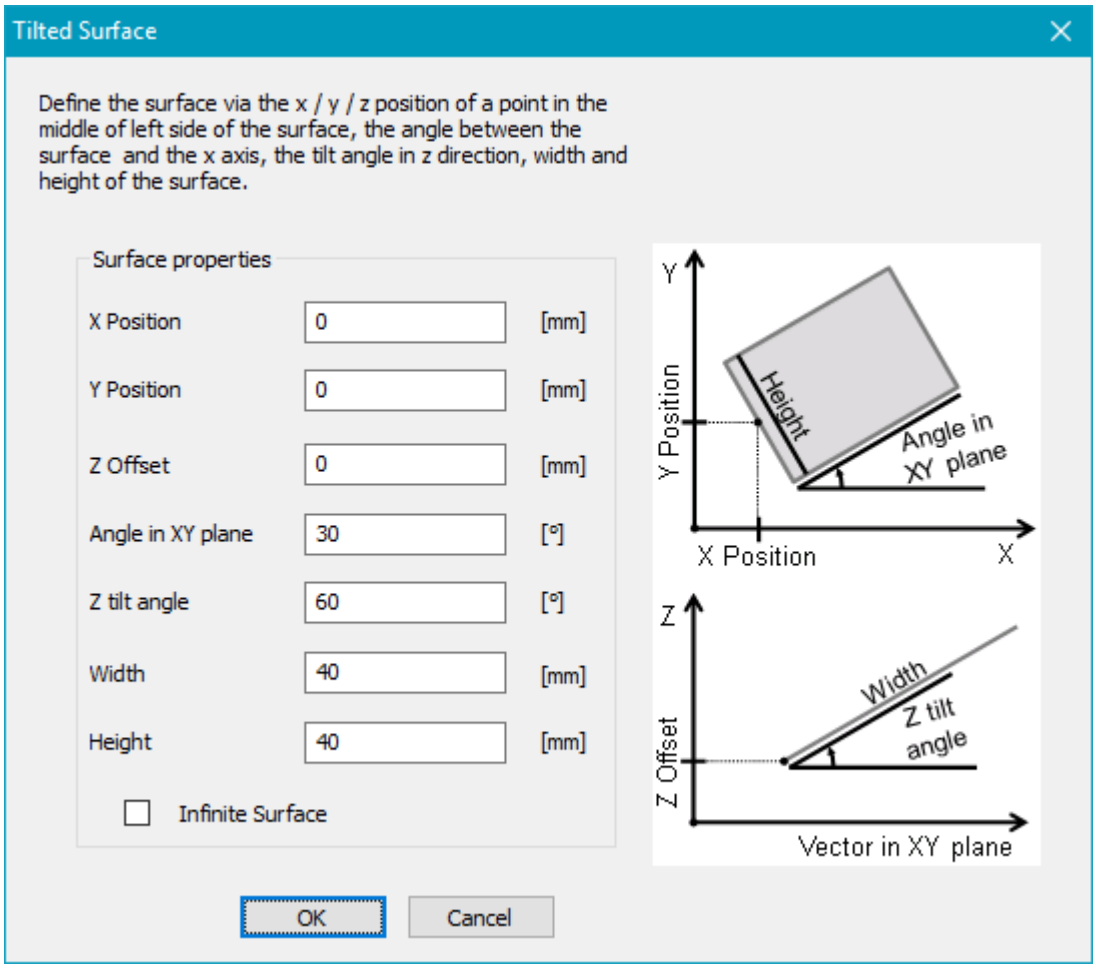


Figure 307: 3D Tilted Surface property dialog

Enable the checkbox right from the cog wheel to see the tilted surface in the View2D.

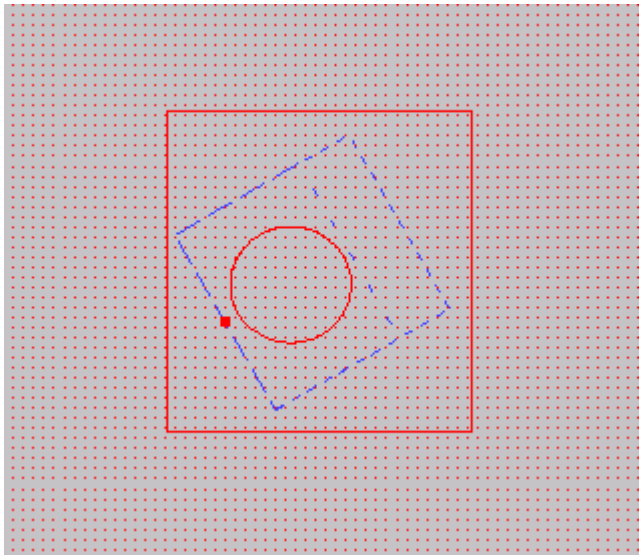


Figure 308: 3D Tilted Surface in View2D

The dashed line in the middle of the square indicates the projection of the end of the surface on the XY plane.

The result is shown in the 3D View:

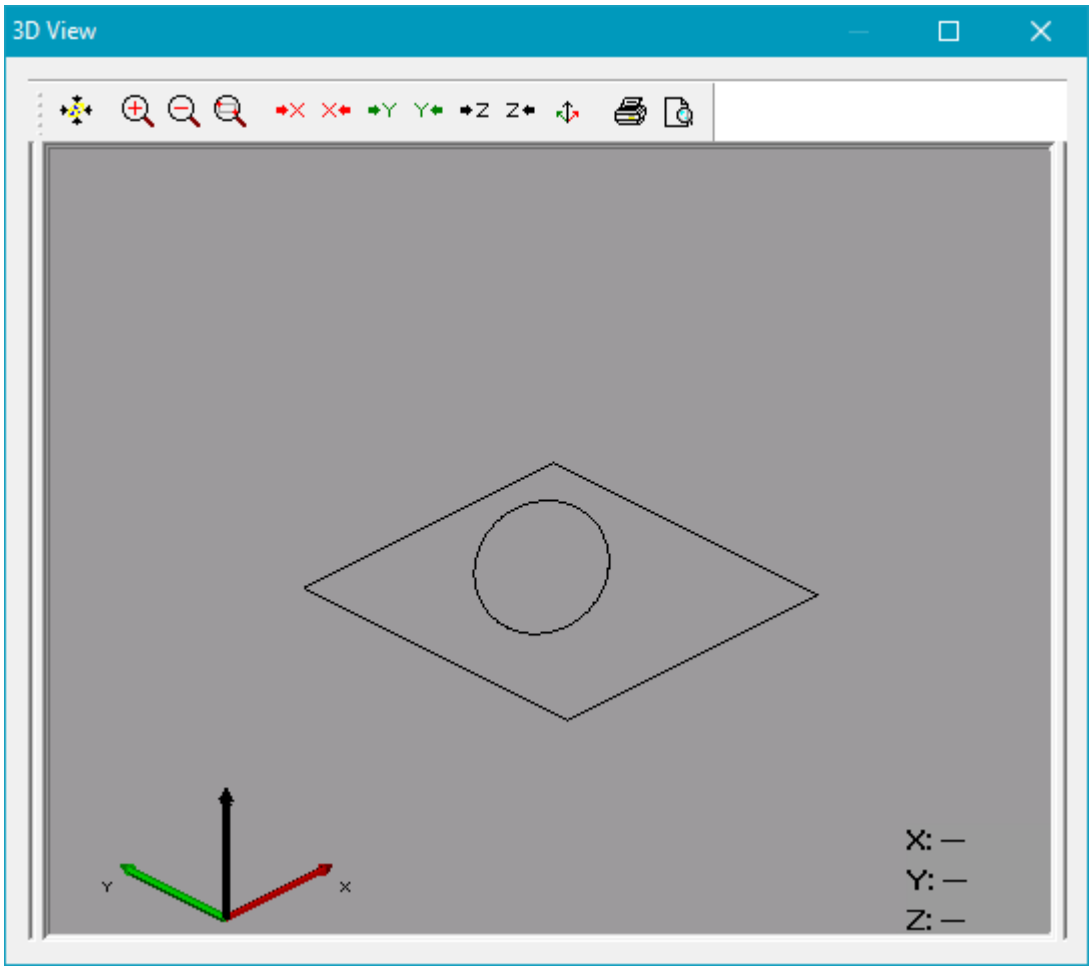


Figure 309: 3D Tilted Surface circle inside of plane XY square in View3D

17.1.1.4 Sphere

Figure 310: 3D Surfaces toolbar

Click on the cog wheel to open the properties dialog:

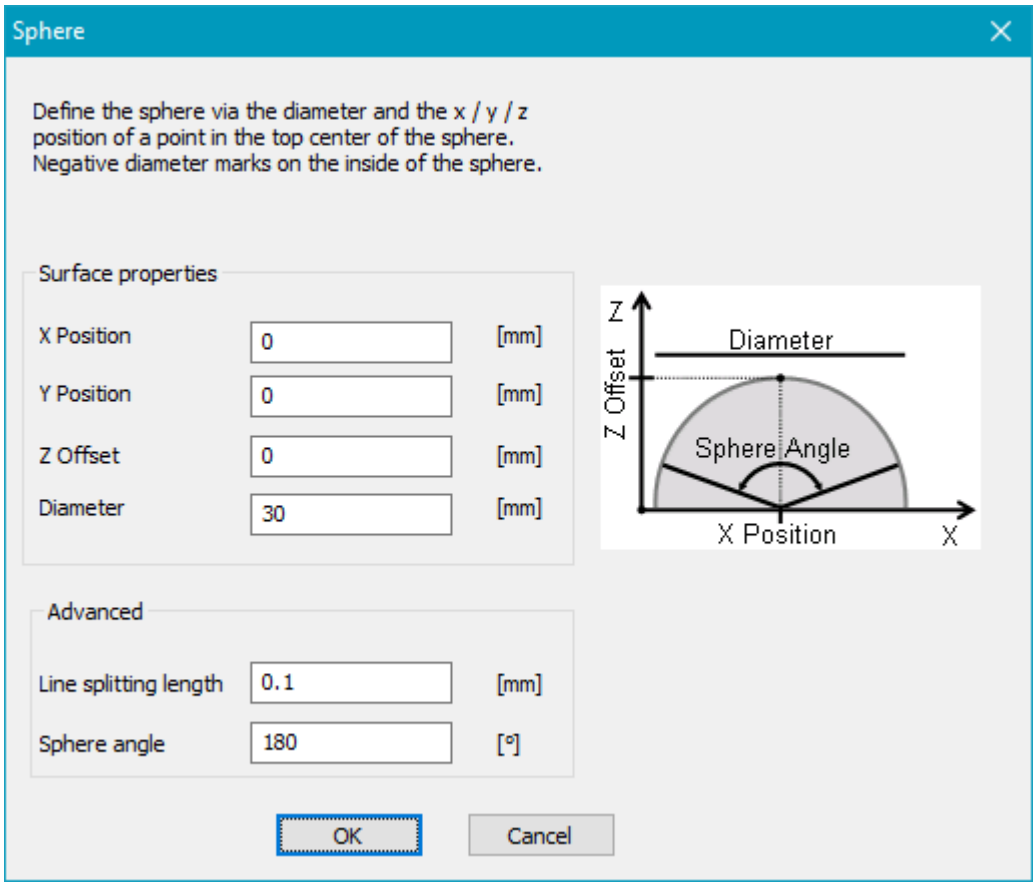


Figure 311: 3D Sphere property dialog

Enable the checkbox right from the cog wheel to see the sphere in the View2D.

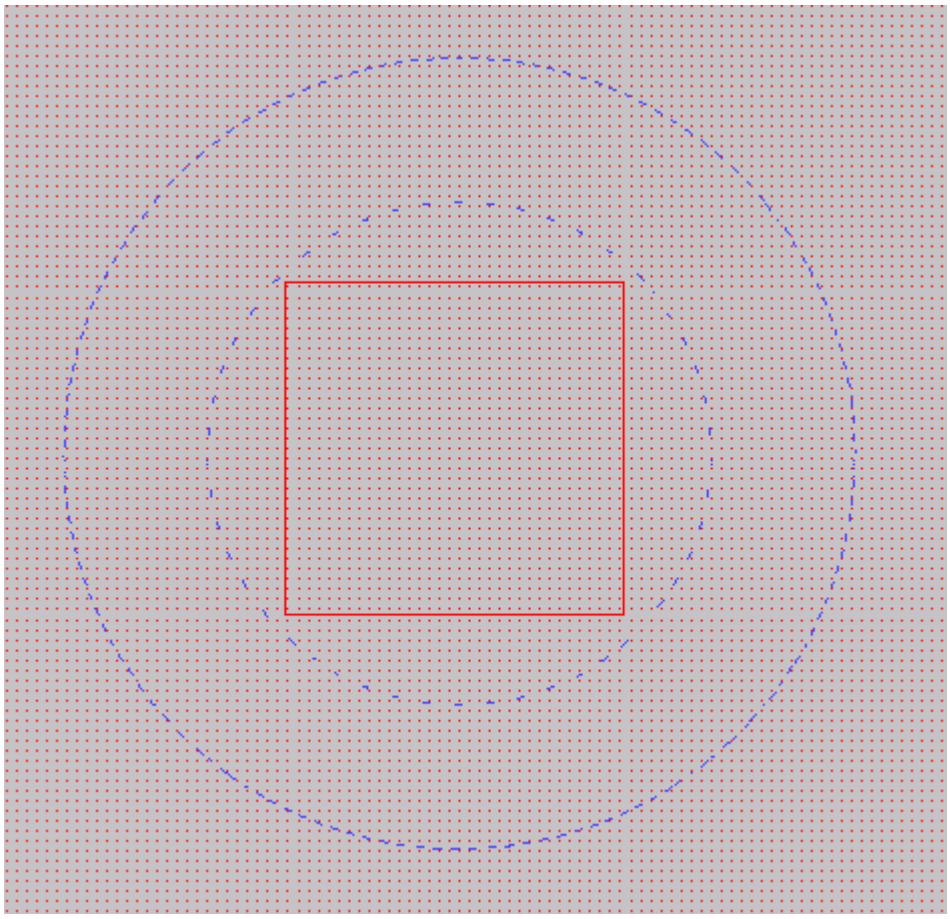


Figure 312: 3D Spherein View2D

17.1.2 Marking on curved parts

With Optic3D 3D DXF files can be imported. A translation of objects in z-direction, their rotation around all axis as well as a manual manipulation of 3D-vectors are additionally possible. The marking of 3D-line structures is only possible with special hardware (suitable scanner controller card / 3-axis scan head). For this a special correction file is needed, see Requirement & Settings. The range of the z-direction depends on the stroke of the scanner. The three axis scanner is used for marking on curved parts, means the scanner can mark lines with varying Z-values. 3D vectors can be imported (*File → Import...*) by a 3D DXF file (*DXF Files Version 2*) and the Z-values of lines / points can be edited.

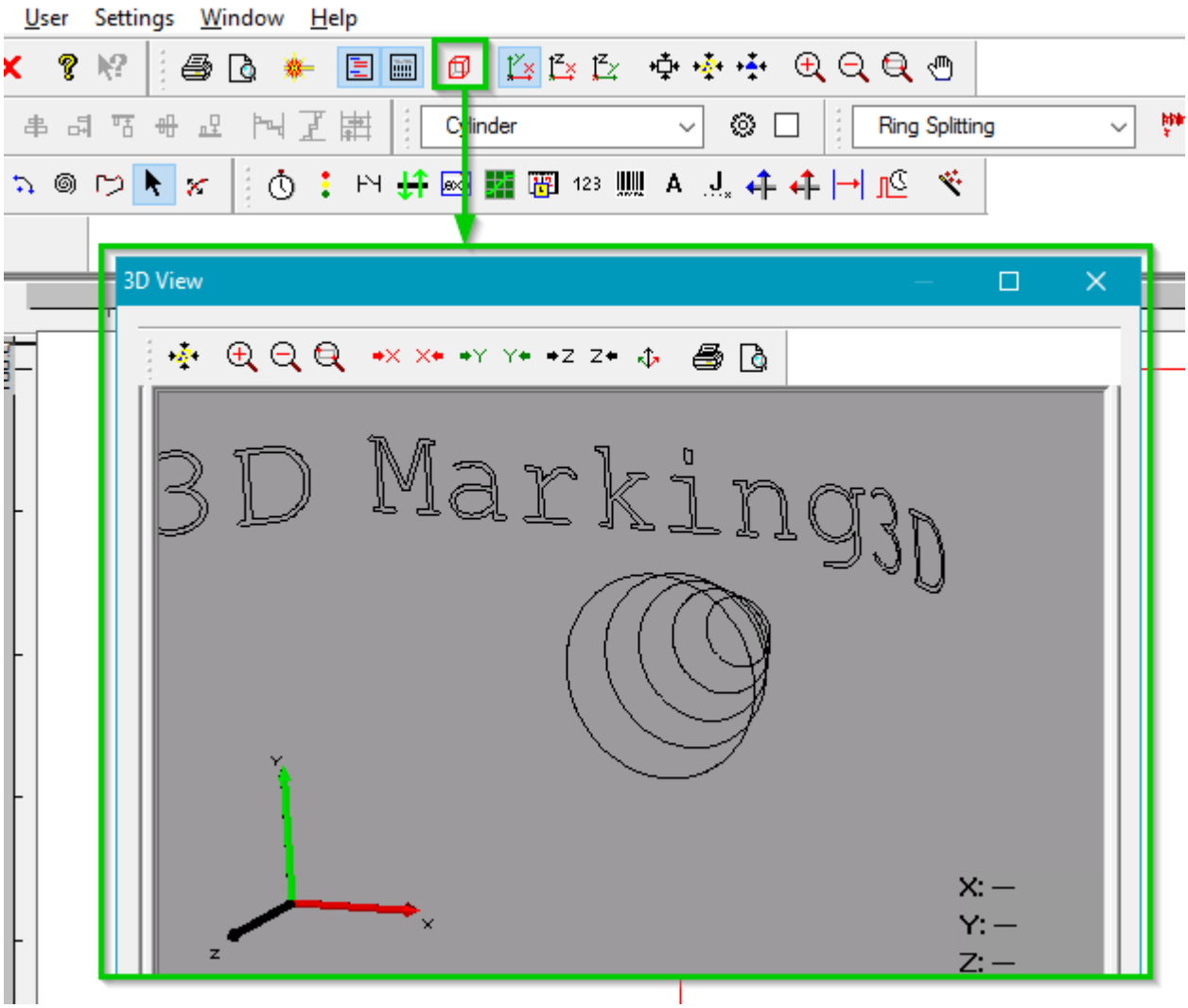
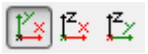


Figure 313: Example of 3D object in optic 3D view

In the 3D View (*cube* button) you can have a closer look at your 3D object by rotating the point of view.



3D view buttons: With these buttons you can view the 3D object in different planes.

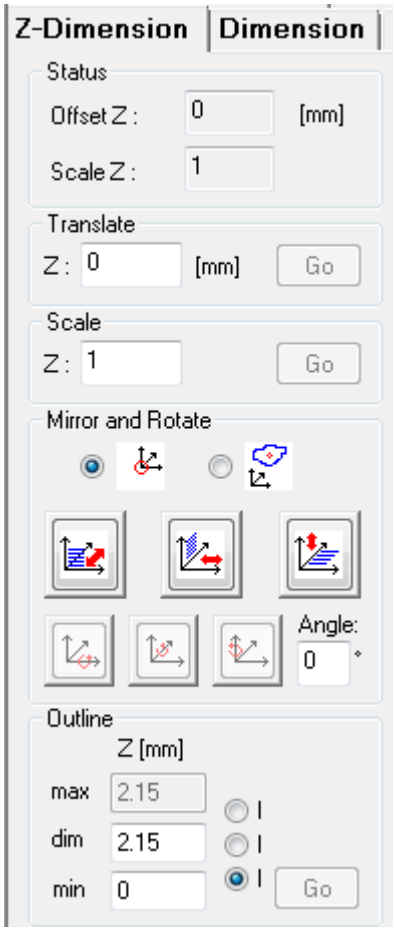


Figure 314: Z-Dimension property page

In the Z-Dimension property page you can translate and scale objects in the z direction and perform mirror and rotation operations.

Status:

Offset Z: Difference of Z value between the original object and the object that has evolved due to transformations.

Scale Z: Difference of Z scaling between the original object and the object that has evolved due to transformations.

Translate:

Z: Translate whole object in Z-direction.

Scale:

Z: Scale whole object in Z-direction.

Mirror and Rotate: Here you can choose the center of rotation.

Mirroring: Mirrors the object on the X, Y or Z plane.

Rotate around X, Y or Z axis: Rotate a specific angle around the X, Y or Z axis.

Outline:

max: Maximum z-Coordinate of the object.

dim: Z-dimension of the object.

min: Minimum Z-Coordinate of the object.

By pressing the *point edit mode* button in the toolbar, you can edit single points of the 3D object.

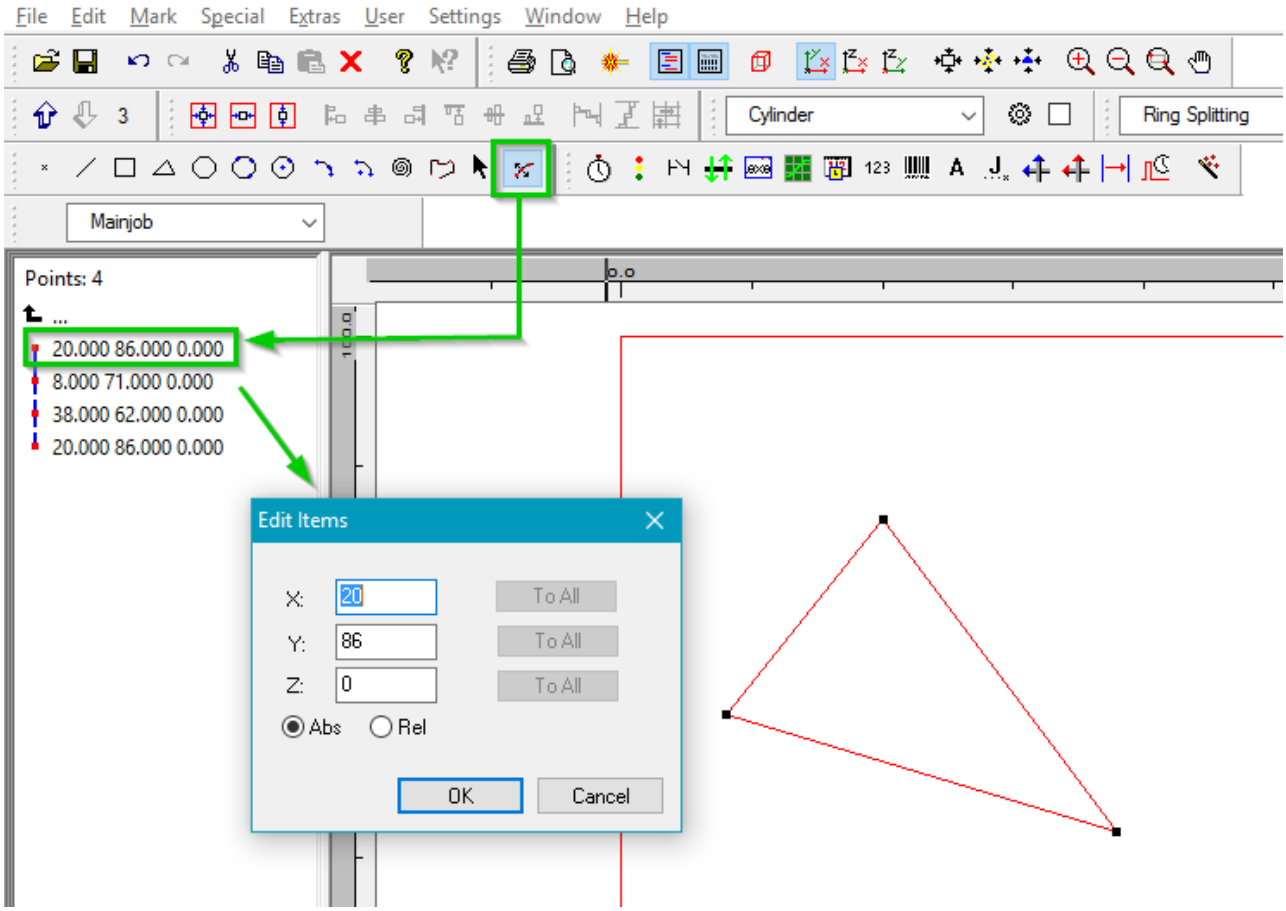


Figure 315: Optic3D point edit mode

17.1.3 Deep Engraving

This feature is used when the object in the editor should be marked several times at different Z-heights, e.g. to do deep engraving on a material. The dialog can be accessed via the menu *Mark → Z-Settings.*

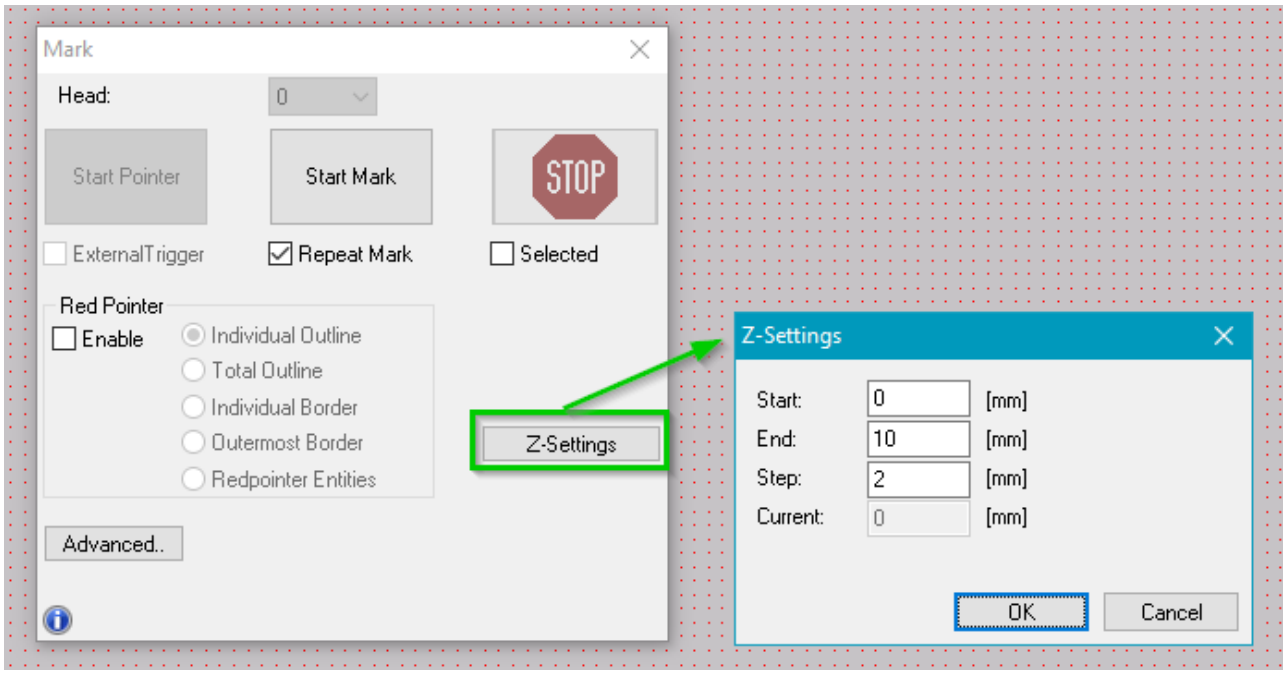


Figure 316: Deep Engraving Dialog

This is how this feature works:

1. Loop: Z = Start
2. Loop: Z = Start + Step*1
3. Loop: Z = Start + Step*2

...

i. Loop: Z = Start + Step*(i-1)

...

n. Loop: Z = End - Step

After last marking the Depth is equal to End, because the last marking is at End - Step and erodes the Step value. The Z-Settings are being stored within the jobfile.

17.2 Requirements & Settings

In the following the requirements and settings for the different scanner controller cards are described.

17.2.1 SCAPS USC cards

Requirements for SCAPS USC-1 and USC-2 cards:

- Correction File: 2D ucf file from SCAPS.
- Required SCAPS option: FlatLense and Optic3D.

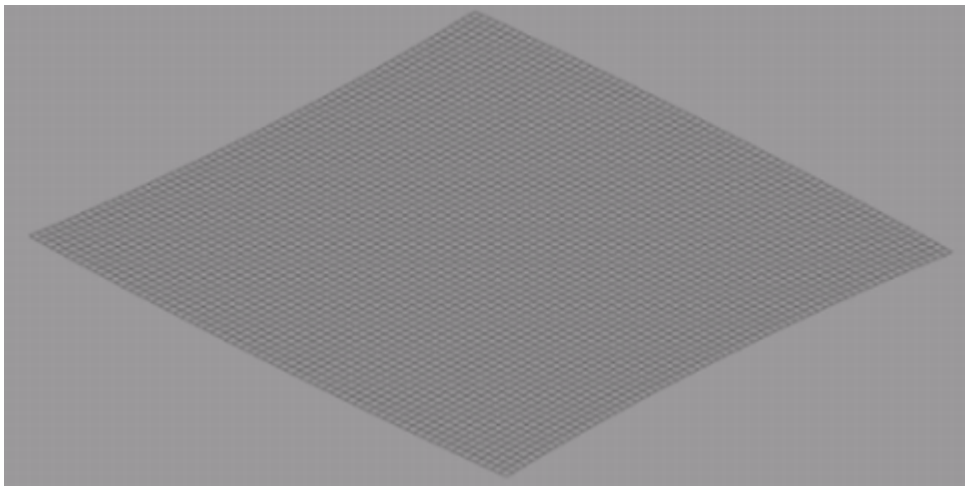


Figure 317: Shape of a 2D Correction File

Settings: For USC-1 and USC-2 cards enter the following Dialogs *Settings → System → Optic*:

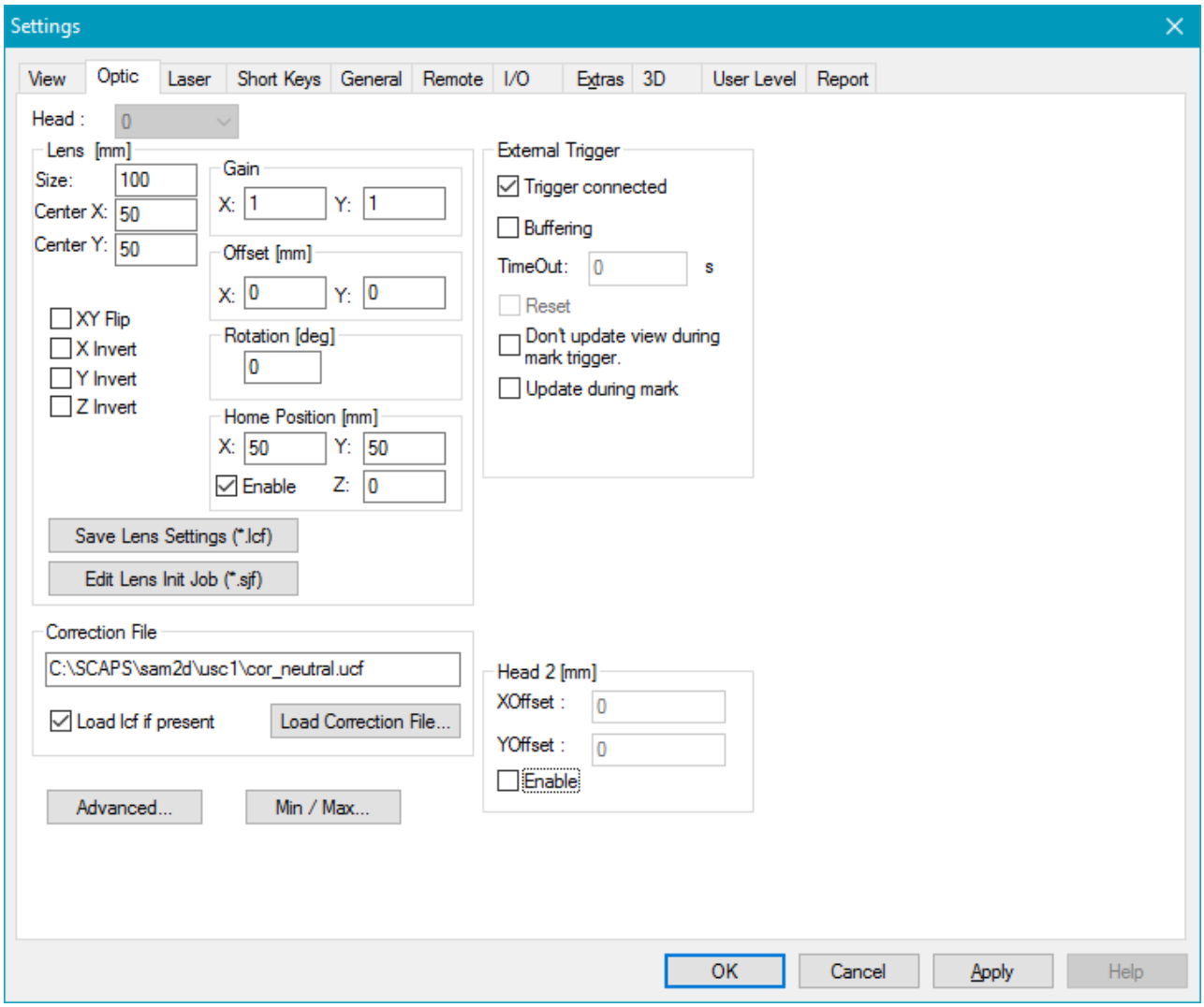


Figure 318: Settings → Optic Dialog for Z-Dimension

In this dialogue the XY-fieldsize and the correction file can be defined.

Find the following Dialog for USC-1 in *Settings → System → Optic → Advanced → Z-Correction (Enable) →*

Advanced and for USC-2 in *Settings → System → Optic → Advanced → Correction, Settings → Z-Correction, Settings.*

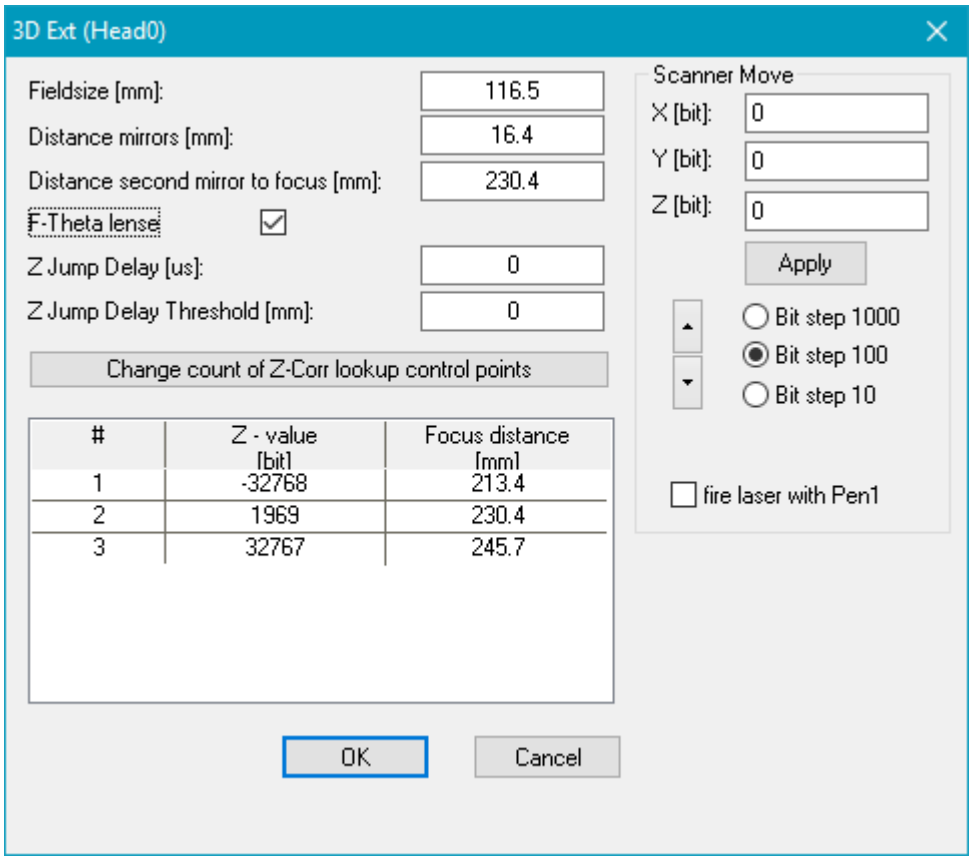


Figure 319: Z-Correction Settings

Fieldsize: The size of the XY-field needs to be identical with the fieldsize set in Settings→System→Optic.

Distance Mirrors: Distance between the mirrors in mm.

Distance second mirror to focus: Distance between the second mirror to the center of the field in the focus plane.

F-Theta lense: Should be checked if in addition to the 3rd scanner axis, a F-Theta lense is mounted.

Z Jump Delay [µs]: Due to the fundamental difference of the Z axis it could be necessary to increase the jump delay after a big change in the Z value. The *'Z Jump Delay'* is added to the normal jump delay when a jump has a Z dimension greater than *'Z Jump Delay Threshold [mm]'*. A value of *'0'* disables this feature.

Z Jump Delay Threshold [mm]: If a jump has a Z dimension greater than this threshold the 'Z Jump Delay [µs]' is added to the normal jump delay.

Change count of Z corr lookup control points and lookup table: For compensating the nonlinearity of the relation z axis position to focus distance, a lookup table can be defined. The number of control points can be between 2 and 32. The lookup table contains the DAC value plus the distance from second mirror to focus when adjusting this DAC value. The points must be ordered ascending by the 'Focus Distance [mm]' value.

XYZ Move: A helper function for finding the proper lookup table values.

High / Normal / Fine: Define Step Width for XYZ Move.

Recommended calibration procedure:

- Get the correction file from Scanhead manufacturer and send it to SCAPS (info@scaps.com) for being converted into a UCF file format containing no z values

- Setup hardware to right working distance (z=0 plane).
- Find the right lens size value to get a correct aspect ratio when marking for example a rectangle. 10 mm in drawing have to be 10 mm on marking (in z=0 plane). The lens value defines how many mm are related to the 65536 bits.
- Get the right mirror mounting distance values.
- Within the z calibration dialog, type in values for fieldsize and the 2 distance mirror values.
- For getting the right calibration values for the lookup table
- Best would be having a z stage for calibration of different z heights. The valid DAC range is from -32768 to +32767. Two general ways to generate the lookup table. Either predefining DAC and varying z or the other way around.
- E.g. you create 3 control points and enter following values for DAC.
- Then you search for the focal planes d1, d2 and d3 with the XYZ Move window. For compensating further nonlinearities you have to create further control points in the lookup table.

	DAC [bits]	[mm]
1	-32768	d1
2	0	d2
3	32767	d3

Table 34: Example Z lookup table. d1 < d2 < d3

17.2.2 SCANLAB RTC cards

Requirements for SCANLAB RTC3, RTC4 and RTC5 cards:

- Required SCAPS option:
  - **Optic3D**
- Required SCANLAB hardware:
  - RTC board
  - RTC option "Controlling the Third Axis of a 3-Axis System"
  - RTC option "Second Scan Head Connector" (if Z channel is wired to 2nd scan head connector)
- Required software:
  - **32bit 3D** drivers (for example: RTC4D3.hex)
  - Correction file: **3D** CTB (or CT5) correction file from SCANLAB

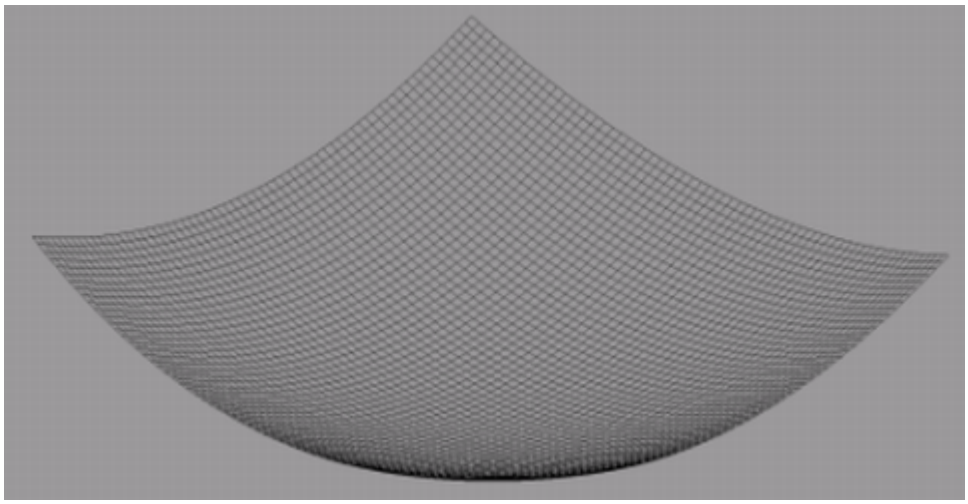


Figure 320: Shape of a 3D Correction File

Settings: For 3D marking with RTC boards, a 3D program file and 3D correction file has to be set. In the 3D Ext dialog the values A,B,C for the Z-table can be defined. You can find this dialog at *Settings → System →*

Card → Advanced → Driver Settings → 3D Ext. For the values please ask the scan head manufacturer.

If Z channel is wired to the 2nd scan head connector, enable the 2nd head and assign the same correction file at *Settings → System → Card → Correction.*

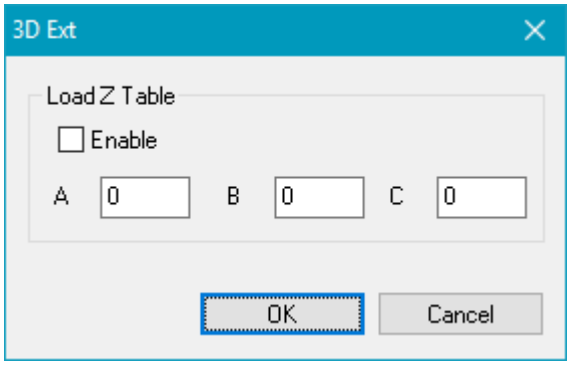


Figure 321: 3D Ext Dialog for RTC cards

18 Option SAM3D

This chapter describes the 3D functionality of SAMLight. Before working in the 3D mode *Menu bar → Settings → System → 3D → General, 3DView* must be checked. Then click on *OK* and restart the software.

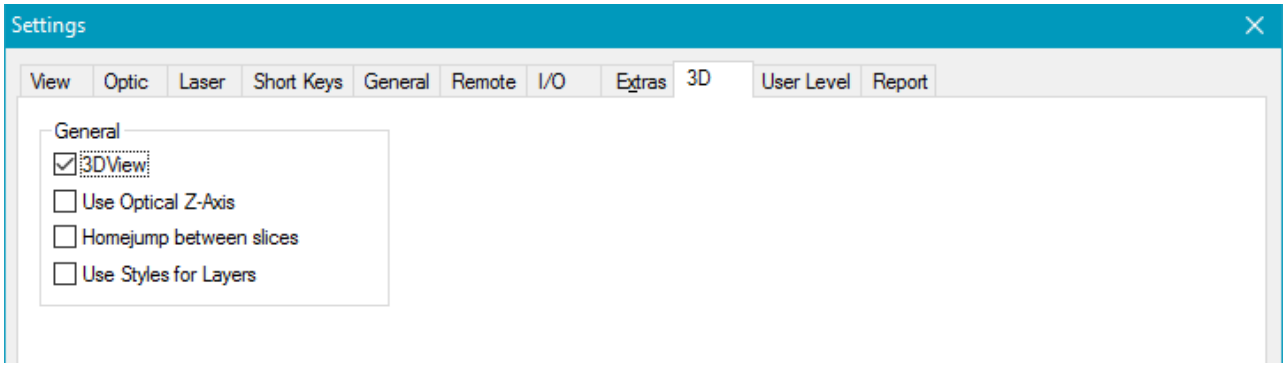


Figure 322: 3D Settings Dialog

General:

3DView: Enables / Disables SAM3D mode. The software must be restarted for the change to take effect.

Use Optical Z-Axis: Check this option, when you want to shift the focus optically with a 3D scan head. Please note: you need a license for Optic3D to use this option.

Home jump between slices: Depending on the settings of the home jump in *Settings → System → Optic → Home Position* there are two different options available:

- If the home jump is enabled a home jump will be performed after each marked slice.
- If the home jump is disabled there will not be a home jump, but the laser power of the HomeJumpStyle is set after each marked slice.

Use Styles for Layers: Enables the Styles property page to assign different pens and hatches to the sliced layers of the 3D object. See chapter "Styles for Layers".

18.1 Main Window

In the following the main window of SAM3D is shown. This window appears after software start.

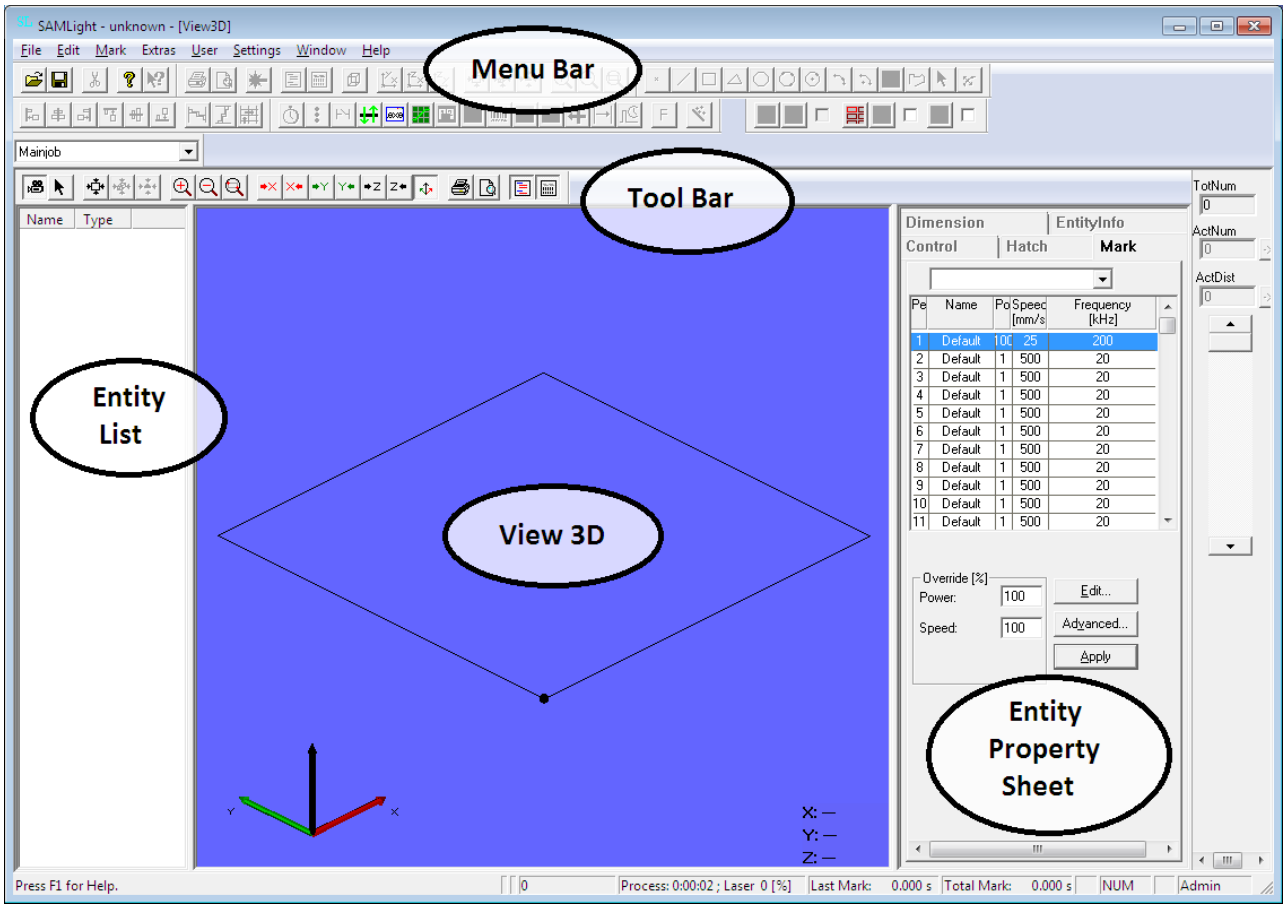


Figure 323: Main Window for 3D Mode

It is very similar to the main window of SAMLight. It consists of the *Menu Bar*, the *Toolbar*, the *Entity List*, the View 3D and the *Entity Property Sheet*. The *Menu Bar* contains only the open and save file functions. The *Toolbar* consists of functions to change the view. The *Entity List*, the *View 3D* and the *Entity Property Sheet* are similar to the items in the SAMLight 2D application. The rotation in SAM 3D is around a vector, which can be defined in Property page→ dimension. The black quad in the View 3D shows the working area in the X and Y directions.

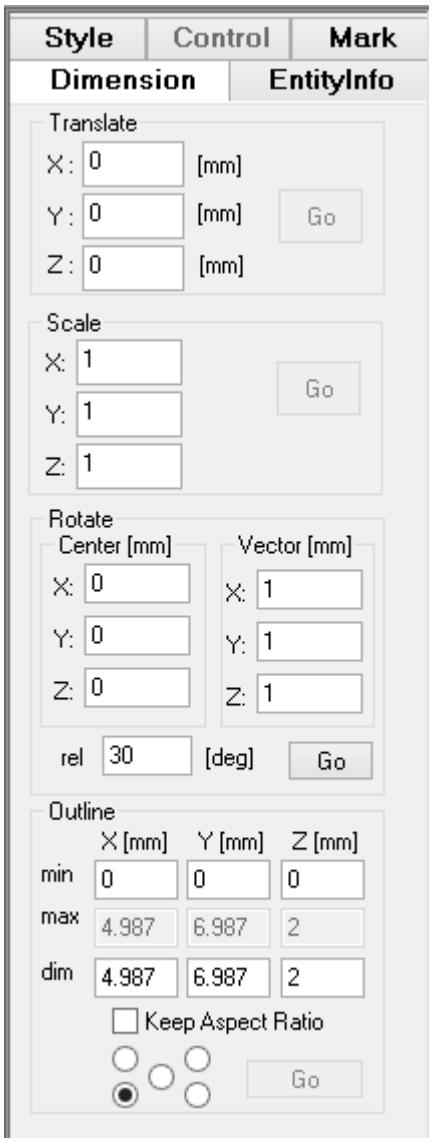


Figure 324: Dimension

18.1.1 Import Folder

This dialog allows to import several slices in one turn and creates an ScLayerSolid out of them.

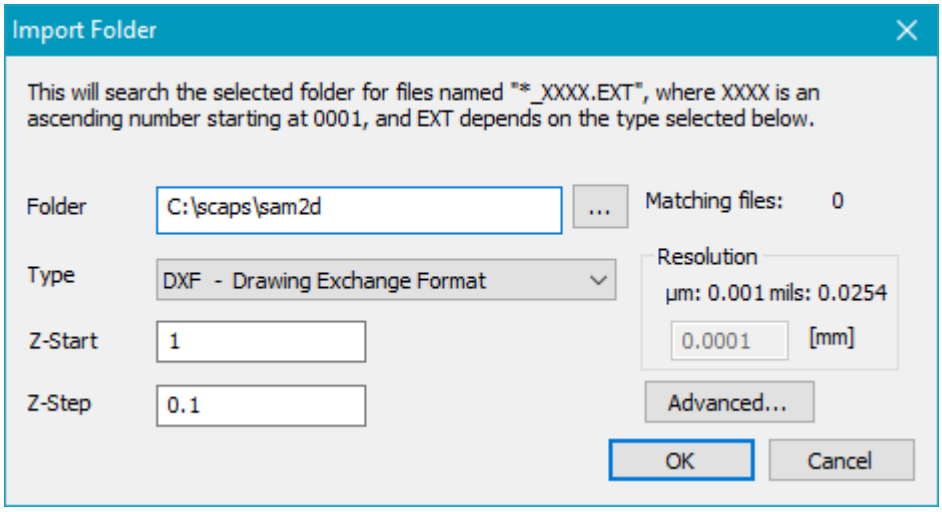


Figure 325: Import Folder Dialog

18.2 Job Processing

In SAM3D it is not possible to create a job like it is in the SAMLight 2D application. A job must be loaded from a *.s3d file or imported from a *.stl, *.cli or *.slc file. To load a *.s3d file click on *Menu bar → File → Load...* . To import a *.stl, *.cli or *.slc file go to *Menu bar → File → Import...* . Once a file has been loaded or imported the 3D object is shown.

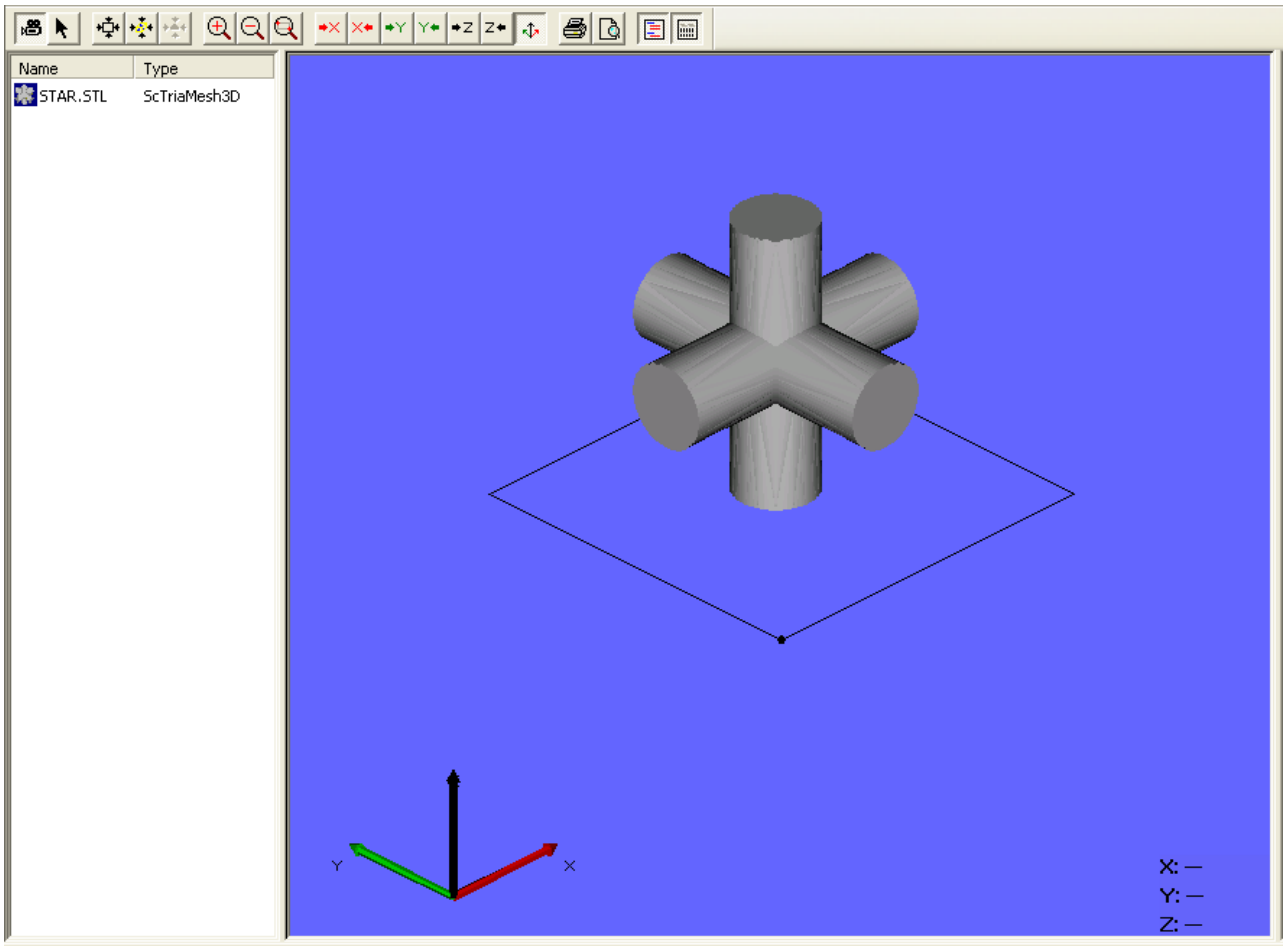


Figure 326: View 3D

On the left side the entity list shows the name and the type of the 3D object. Now the following actions to modify the view are possible:

18.2.1 Toolbar

 **Toolbar:** The Toolbar offers some basic functionality to modify the view of the object.

 **Camera Mode:** If this is activated you can turn around the 3D object by clicking on the View 3D and moving the mouse while the left mouse button is being pressed.

 **Selection Mode:** If activated you can select an object in the view.

 **Fit View To Working Area:** The view is zoomed so that the whole working area is visible.

 **Fit To Entities:** The view is zoomed so that the entities are visible with maximum dimension.

 **Fit To Selected:** The view is zoomed so that the selected entities are visible with maximum dimension.

 **Zoom In:** The objects in the view are enlarged.

 **Zoom Out:** The objects in the view are scaled down.

 **Custom Zoom:** You can choose a region in the view that will be enlarged. Therefore click with the left mouse button on one corner of the desired region. While the mouse button is pressed move the mouse to the other corner and release the mouse button.

 **View Along Axis:** Watch the object along the X,Y or Z axis.

 **Standard 3D View:** Watch the object from a standard 3D view point.

18.2.2 Mouse Mode

There are two different mouse modes available to transform the entities. One is the Transform Camera mode and the other one is the Transform Entity mode. Both transformations work with the mouse while pressing a key on the keyboard:

Camera Transform: Activate Camera Transform by clicking on the Camera symbol  in the toolbar. Now the following actions are possible (**LMB = Left Mouse Button**):

- **LMB:** Rotate around the horizontal or vertical axis
- **Ctrl + LMB:** Scale
- **Shift + LMB:** Translate
- **Shift + Ctrl + LMB:** Rotate around the z-Axis

Entity Transform: Activate Entity Transform by clicking the Selection symbol  in the toolbar. Now the following actions are possible:

- **LMB:** Select and unselect the entity
- **Ctrl + LMB:** Selection of multiple entities
- **Shift + LMB:** Translate selected entities
- **Shift + Ctrl + LMB:** Rotate selected entities. The rotation axis depends on the view. In the standard view (exceptional ISO) the rotation axis is the axis which comes out of the view. In all other views the rotation axis is the Z-Axis of the workpiece coordinate system.

18.2.3 3D View Properties

Click the right mouse button while the mouse pointer is inside the View 3D to open the View Properties.

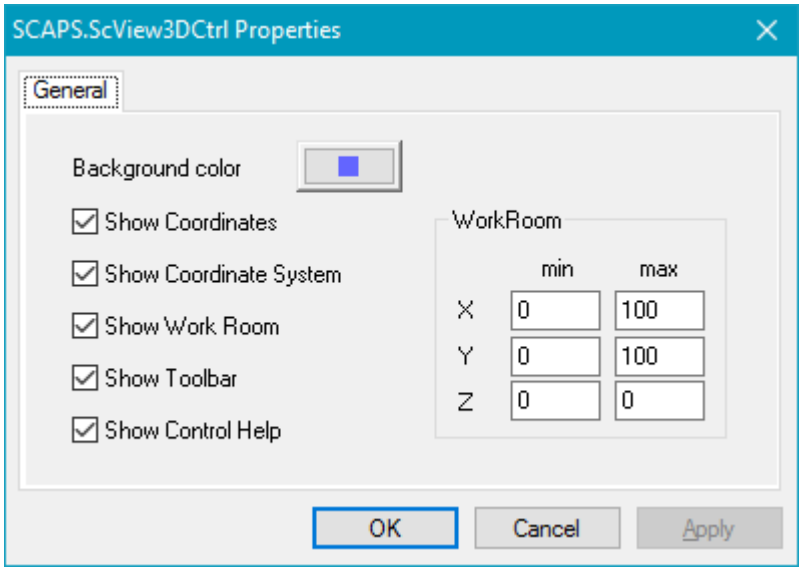


Figure 327: 3D View Properties Dialog

Here, choose the background color, activate or deactivate the options and define a work room.

18.2.4 Slicing

Before you can mark a 3D object it has to be sliced. This means it has to be decomposed in many 2D layers which then can be marked as normal 2D objects. After marking a layer a motor has to move the target object to the next z-position or the z-focus of the scan head has to be set to the next z-position. You can choose between these two options by checking or unchecking *Menu bar → Settings → System → 3D → Use Optical Z-Axis*. To slice the object choose *Menu bar → Edit → Slice*. The following dialog appears:

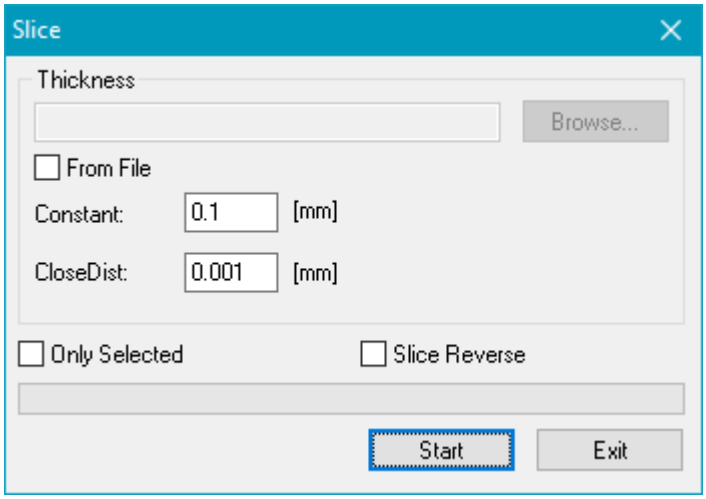


Figure 328: Slicing Dialog

Thickness:

From File: The slicing information can be read from a file too. This file is a plain *.txt file with the following structure:

- 2
- number of slices

- [R]
- target_dist;stepwidth pen_number [hatch_number]
- target_dist pen_number [hatch_number]

Here the *2* is a version number that corresponds to the internal structure of the file. This field is mandatory. The next line contains a number that is equal to the number of slices that have to be created for that file. This number is used for internal calculations and to provide an expressive progress bar. The third line specifies if all following *target_dist* parameters are relative to the base of the mesh (*R* set) or if the *target_dist* specifies absolute values in the used coordinate system (empty line without *R*). Now all following lines describe the slices itself. Here two methods are possible. The first one describes a *target_dist* and a *stepwidth*. Here the *stepwidth* is used for the thickness of all slices until the specified target distance is reached. Using this syntax it is possible to define a range of slices just by using one single line. The second syntax provides the possibility to define one single slice. Here the *target_dist* specifies where this single slice has to end. Both methods of defining slices use the preceding slice position as starting point. The thickness of a slice results out of the difference between both values. Additionally a pen number is specified in both cases. This one-based pen number is assigned to the related slices. And as a third, optional parameter the number of the hatch can specified that has to be applied to that slice. It corresponds to the default hatch styles of the hatch property pane. To use such a predefined hatch, here the required parameters have to be stored and the related hatch 1 and/or 2 has to be enabled.

As an example such a slice definition file can look like this:

- 2
- 13
- R
- 0.10;0.01 1
- 0.11 2
- 0.15 1
- 0.17 3

Here the file version number is *2* to exactly specify this format is used. The number of sliced defined within that file is *13* and all given distances are relative to the starting coordinate of the 3D mesh. First there is a range of slices defined. Here ten slices have to be created from position *0.00* to position *0.10* with a thickness of *0.01*. The pen 1 is assigned to all these ten slices. Following these three slices are created:

- a single slice from 0.10 to 0.11 (= thickness of 0.01) where pen 2 is assigned to
- a single slice from 0.11 to 0.15 (= thickness of 0.04) where pen 1 is assigned to
- a single slice from 0.15 to 0.17 (= thickness of 0.02) where pen 3 is assigned to

Reverse Slicing direction: If you want to reverse the direction of the slicing from bottom to top to top to bottom write R;S in the second line of the file. The Slice Reverse checkbox must NOT be activated.
Constant: Choose the distance between two successive layers in [mm].

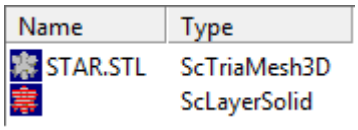
CloseDist: If the distance between two Polylines is smaller than this value they are closed automatically. If a *0* is entered here this functionality is disabled.

Only Selected: Only those objects that were selected are being sliced.

Slice Reverse: Normally the order of the slicing is from bottom to top moving along the z-axis in positive direction. This affects the order of the marking too. A possible application is deep engraving.

Start: Start slicing and create a new entity which holds the sliced layers.

Once the object has been sliced, a new entity is shown in the entity list and on the right hand of the main window a slice control becomes active:



The entity which holds the sliced layers is of the type *ScLayerSolid*. The entity which holds the original information of the 3D object is of the type *ScTrialMesh3D*.

The Slice Control allows to access the slices and get information about it.

TotNum: The total number of the slices.

ActNum: The number of the actually selected slice.

ActDist: The z-distance from the ground of the actually selected slice.

Scroll bar: Click on the bar and move the mouse up or down to select a slice. The actually selected slice will also be shown in the View 3D as a green contour which indicates the layer.

To slice an entity more than once will lead to a corresponding number of set of layers (ScLayerSolid entities). The slice distances may vary. For every set of layers you can define a different hatch, pen etc. If you have more than one set of layers the total number of slices corresponds to the number of slices with different z-value. If slices from different set of layers have the same z-value they will be merged in the same layer to avoid motor activities.

18.2.5 Hatching

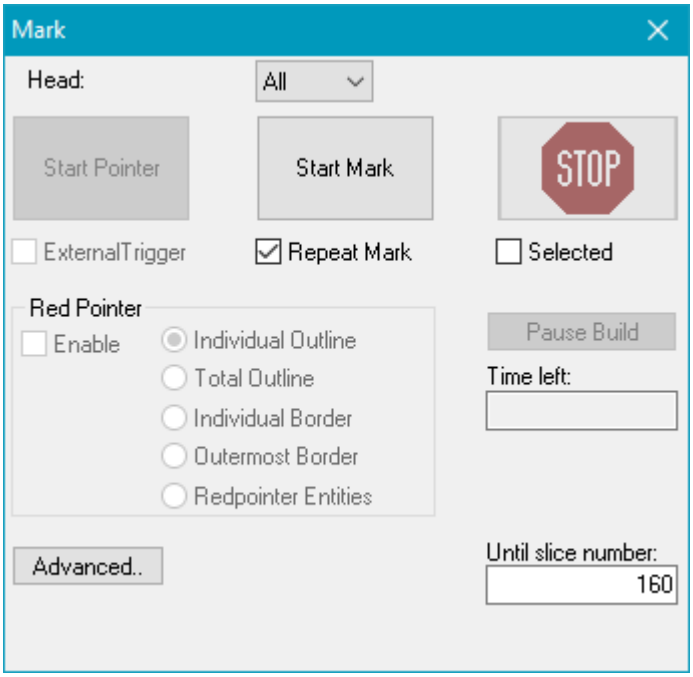
After slicing select the *ScLayerSolid* in the entity list and go to the hatch property page to create hatches for the slices. The SAM3D hatch property page shows one additional value compared to the 2D hatch property page:

Var. Angle: A variable angle from -360° to +360° can be defined to rotate the hatch angle between the slices.

In the mark property page different pens can be chosen for the outline (PolyLines), Hatch1 and Hatch2.

18.2.6 Marking

By clicking on the mark icon  or by selecting *Menu bar* → *Mark* → *Start* the mark dialog is opened:



Start Mark: Starts the marking process. The first slice which will be marked is the number of the actually selected slice.

Repeat Mark: If checked all slices (layers) will be marked one after another. If not checked only the actually selected slice will be marked.

Selected: If checked only the selected ScLayerSolid entities will be built.

Until slice number: The marking will stop if this slice number has been reached.

Pause Build: The running build process will be paused after the current slice has been marked completely. If a slice post-process has been defined it will be executed before the pausing. If you click the button again during a paused build process the next layer will be marked.

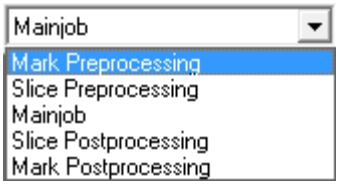
Figure 329: 3D Marking Dialog

18.2.7 Special Sequences

Like in 2D Mode it is also possible to define special jobs which are executed before or after the marking process. This can be the movement of configured motion controllers as well as *ScOverride* entities or entities that wait for an external input signal or entities that set a special output signal. Since the functionality is exactly the same in 2D Mode, please refer to the chapter Special Sequences in *User Interface* → *Toolbar*.

For example:

First setup a Mark Preprocessing.



In the window that opens create a Motion Control by clicking on the Motion Control Icon  .

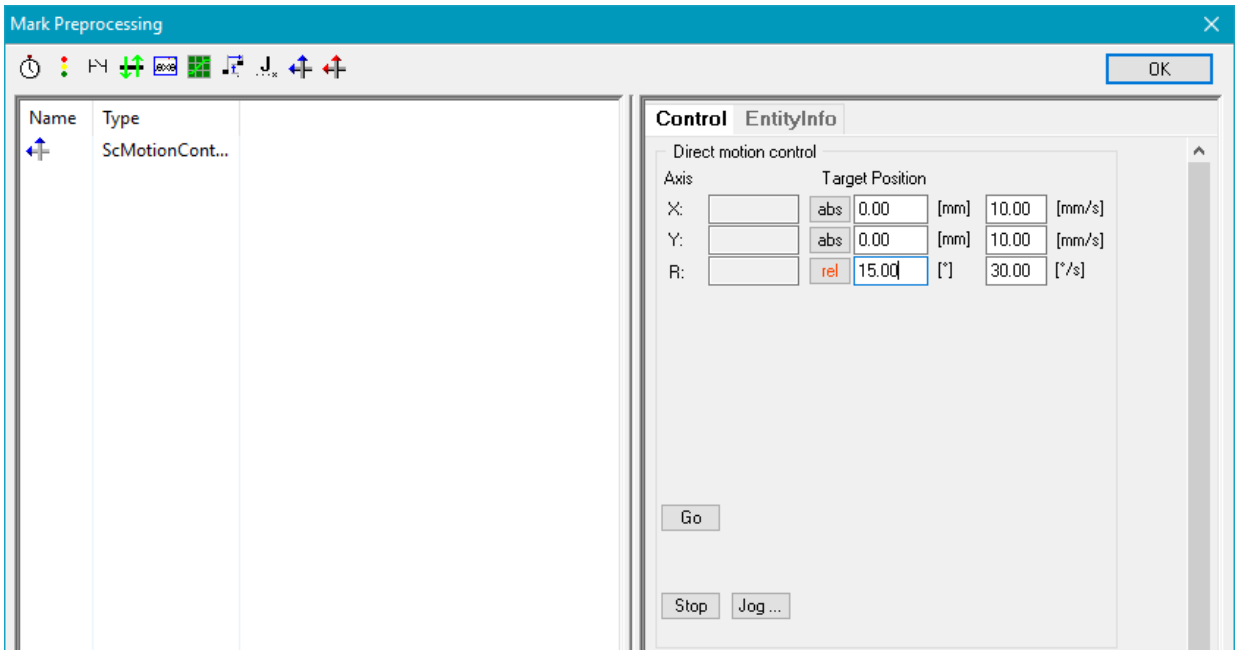


Figure 330: Mark Preprocessing Dialog

In this case the motor will move to its start position at X=Y=Z=0 before the marking is started. Now select *Slice Postprocessing* from the *Jobs toolbar*.

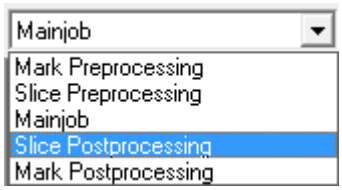


Figure 331: Jobs toolbar

Then create there a motion control entity:

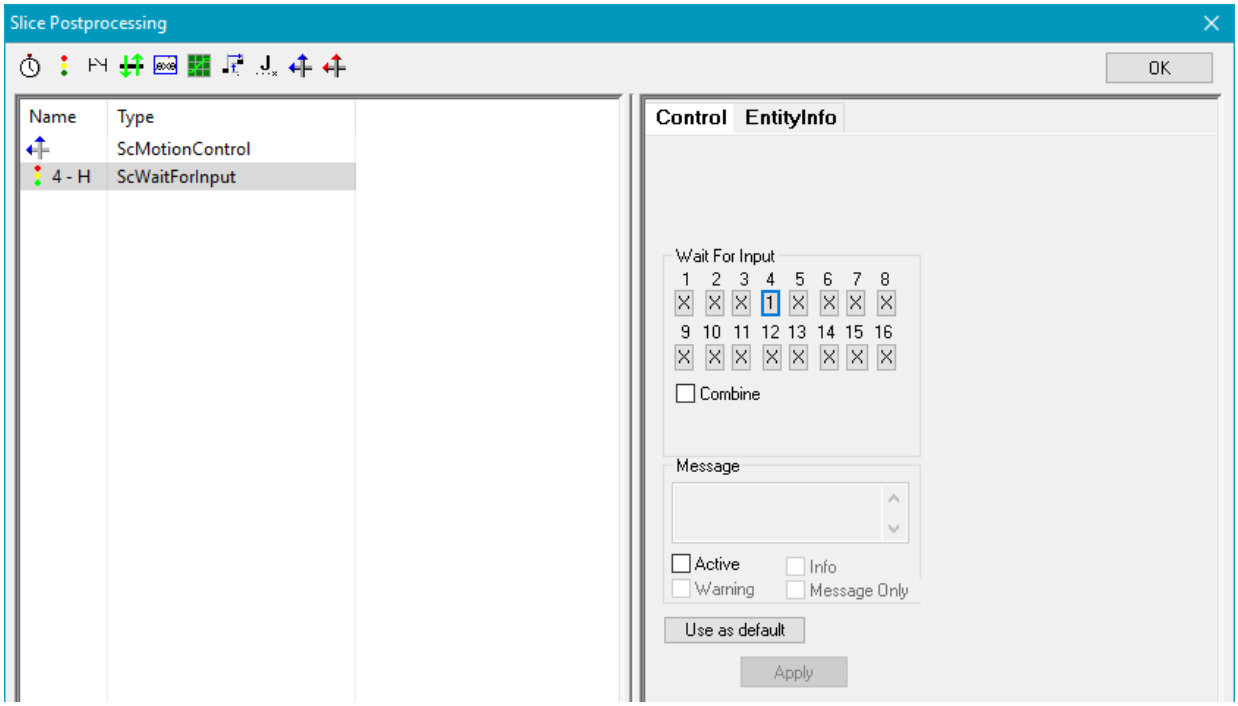


Figure 332: Slice Postprocessing Dialog

In this example a motion control is inserted after every mark of a slice which will move the Z-axis *0.1* mm and leave the actual X and Y position. Now the job is ready to be marked as a whole 3D Object.

Job Properties: It is possible to store the jobs in the Jobs Toolbar when saving the 3D job in a *.s3d file. Therefore go to File → Job Properties and enable the checkbox "Save 3D pre/post jobs in 's3d' job file".

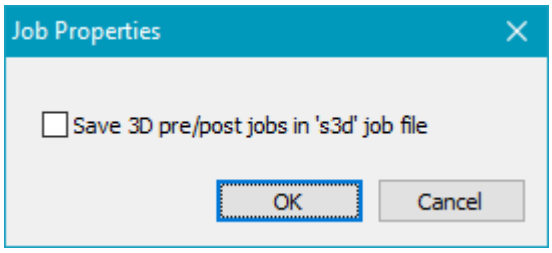


Figure 333: Save 3D pre/post jobs in file

18.2.8 Styles for Layers

To use this feature the checkbox "Use Styles for Layers" has to be activated in Settings → System → 3D. Then The Styles property page will be visible. This feature lets you define Pen, Hatch, Mark Loop Count, Mark Contour and Mark Hatch for each Styles as well as enables you to use the Upskin and Downskin features.

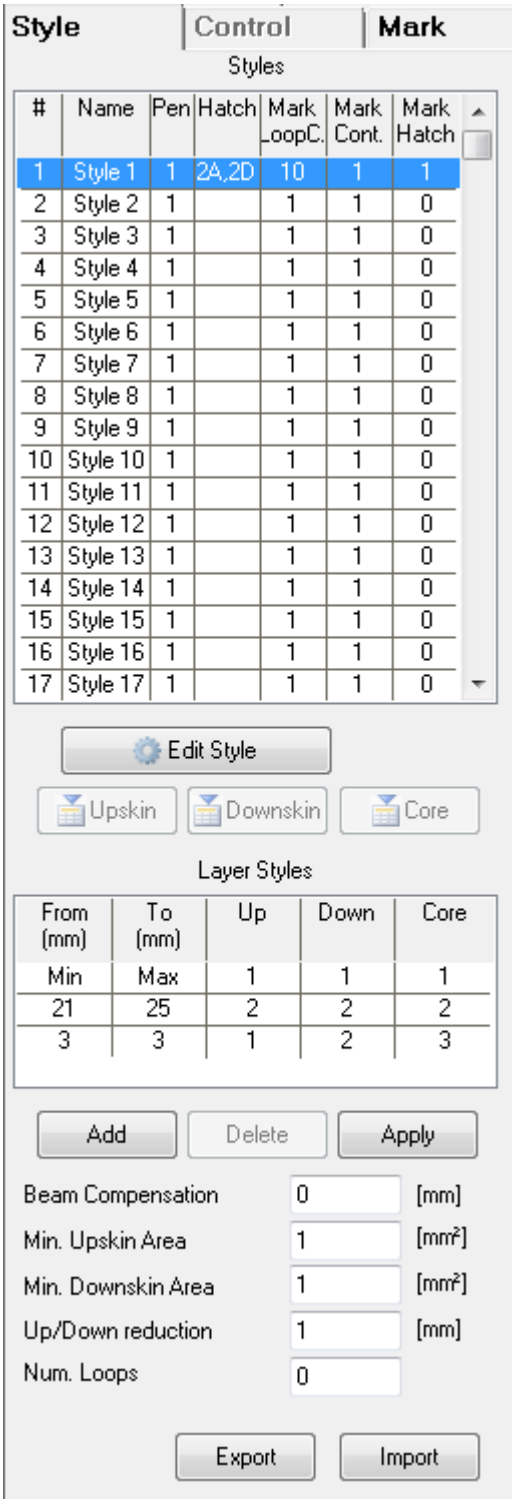


Figure 334: Styles Properties

Edit Style: Opens a dialog in which you can define for each Style a Pen, Hatch, and Mark Flags.

Upskin, Downskin, Core: If a ScLayerSolid entity is available, which is the case if you have sliced a ScTriaMesh3D object, you can define each style to Up- or Downskin or to Core for each Layer. Upskin is the area of the actual slice that is not topped by the following slice, Downskin is the area of the slice which has no slice beneath it.

From - To: Defines a range of ActDist for the selected Style. In this case Style 1 is applied for the starting layer to the end layer interrupted by Style 2 for layers between 21 to 25 mm and also interrupted by different settings for layer at 3 mm.

Add: Add new From - To definition for Styles.

Delete: Deletes a From - To definition.

Beam Compensation: Reduces the marking area for all the layers.

Export / Import: Export or import Styles file. The type of the file is *.txt.

18.2.8.1 Beam Compensation

The Beam Compensation will reduce the marked area so that those objects that are not being marked become larger than before. Or said in another way, the marking area of the marked objects is reduced. See example below:

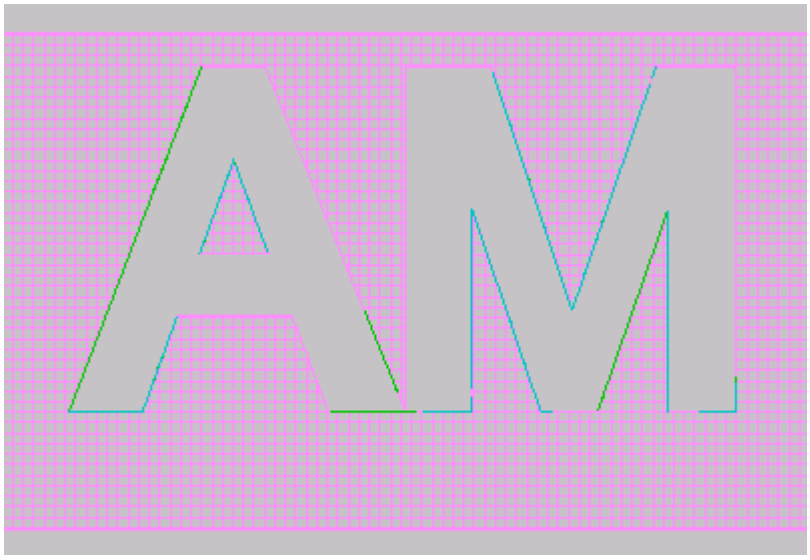


Figure 335: Object with no Beam Compensation

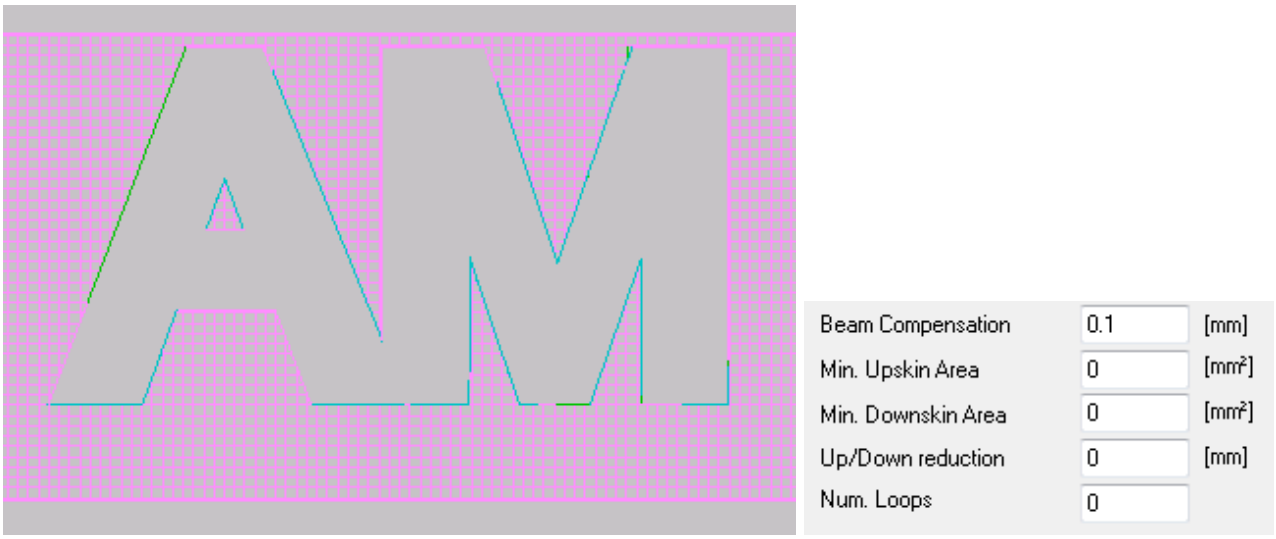


Figure 336: Object with Beam Compensation = 0.1 mm

18.2.8.2 Handling Up and Downskin

Sometimes it is necessary to prepare the data regarding Up and Downskin. An example is given below:

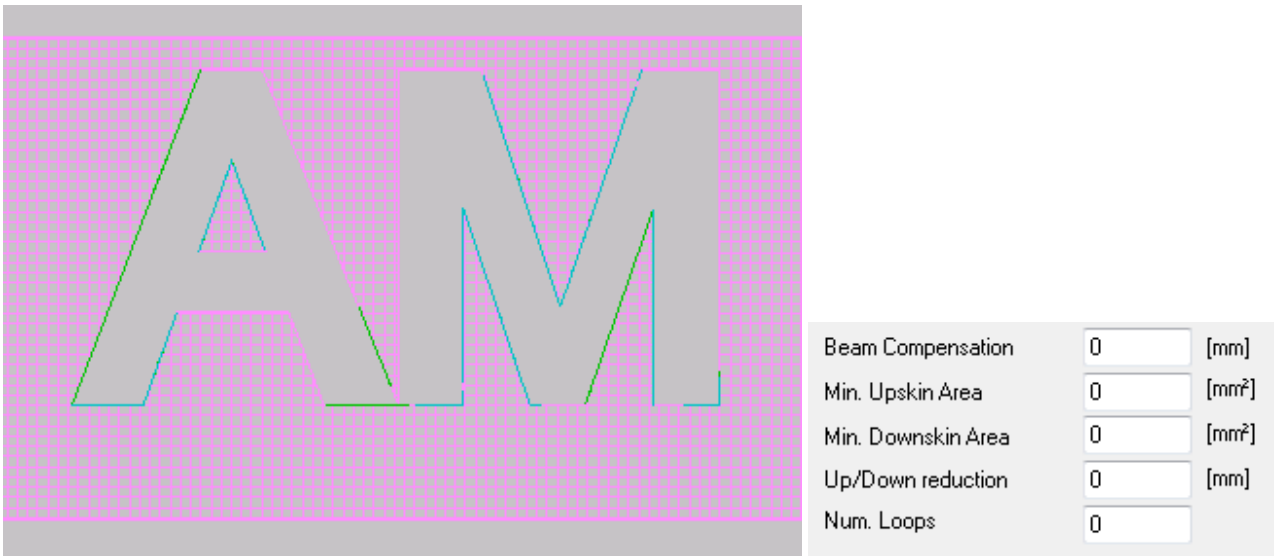


Figure 337: Up and Downskin on a hatched object

In the picture above a 3D contour is hatched and at the borders of the object are Upskin (blue lines) and Downskin (green lines) areas. To discard these areas one can define a Min. Upskin or Min. Downskin area. See the following pictures:

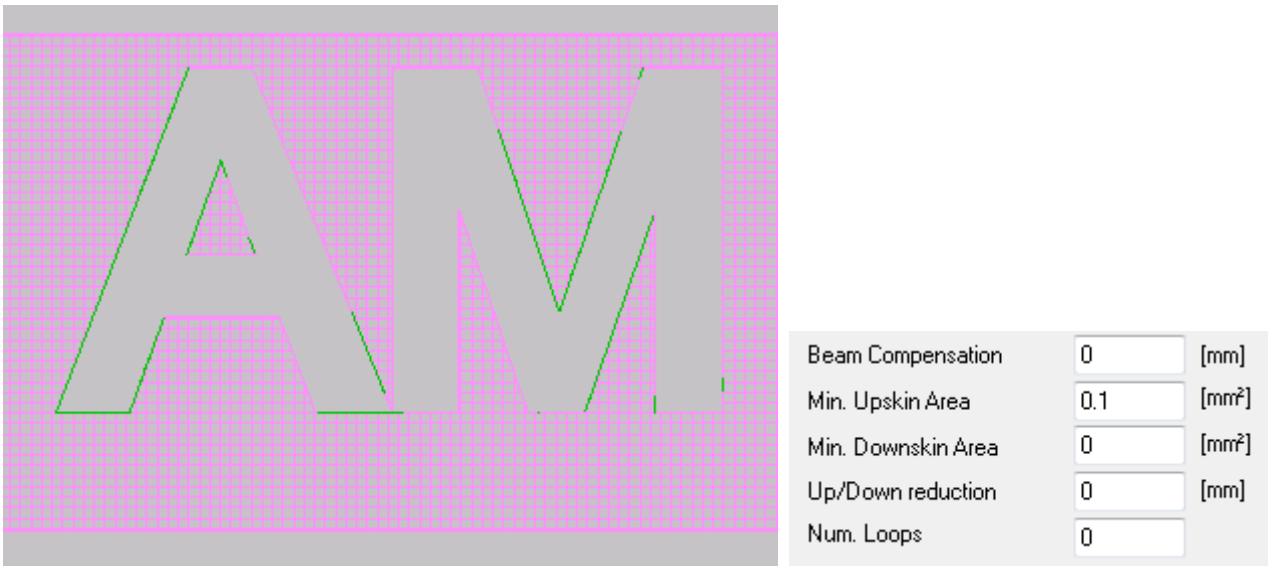


Figure 338: Upskin erased with Min. Upskin Area

In the picture above it is shown how the blue lines = Upskin are erased by setting Min. Upskin Area = 0.1 mm².

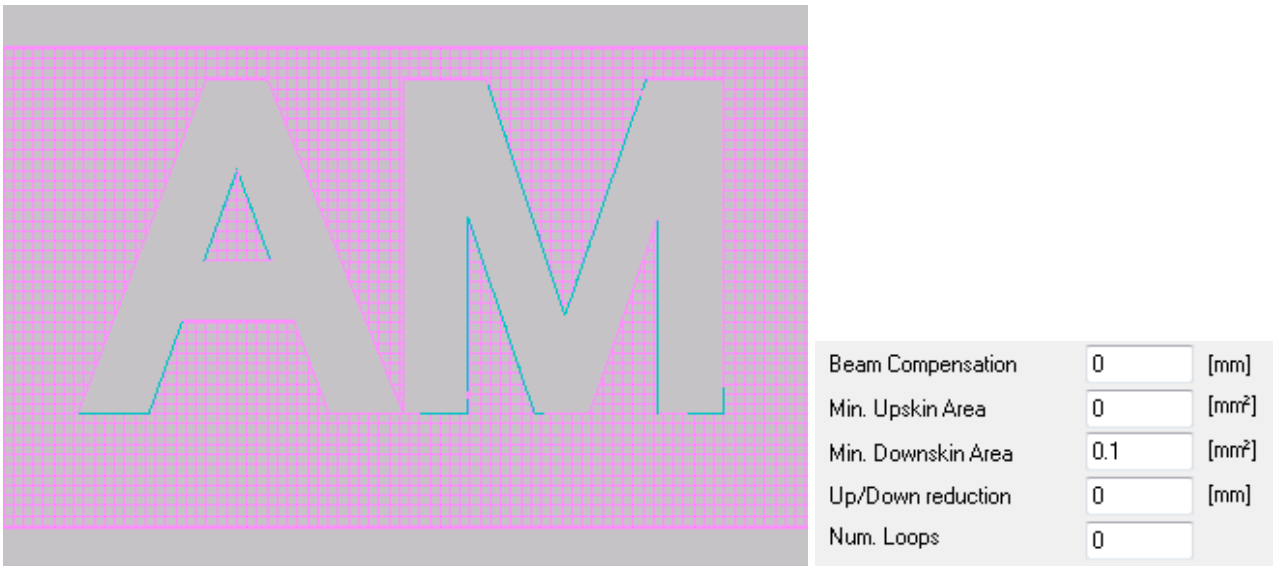


Figure 339: Downskin erased with Min. Downskin Area

In the picture above it is shown how the green lines = Downskin are erased by setting Min. Downskin Area = 0.1 mm².

Figure 340: Up and Downskin erased with Up/Down redcution

In the picture above it is shown how the green lines = Downskin and the blue lines = Upskin are erased by setting Up/Down reduction = 0.1 mm.

18.2.8.3 Using Num Loops

If necessary the hatching lines can be aligned to the border line of the hatched object. This can be done by defining a distance of these lines with Beam Compensation and the amount of lines using Num Loops. Below is an example for Beam Compensation = 0.1 mm and Num Loops = 5.

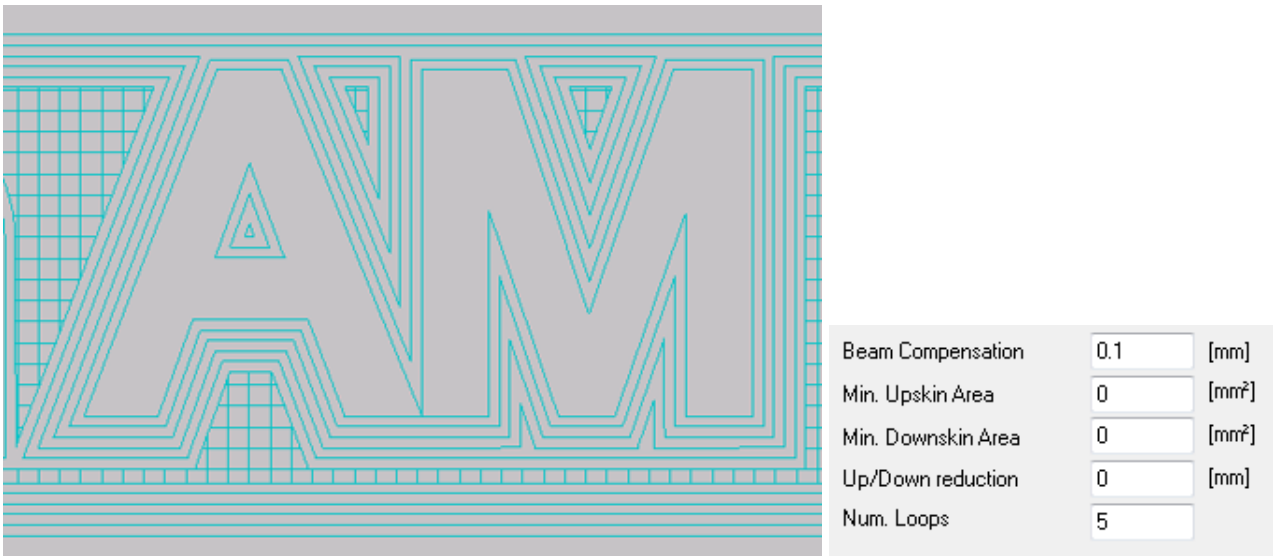


Figure 341: Using Num Loops for a hatched object

18.3 Client Control

Here the SAMLight Client Control commands that work with the SAM3D mode are described. For basic information see also the chapter Programming Interface. In the following all possible commands are shown. Some commands can also be executed by an ASCII transmission via a ccs script.

Function: **long ScIsRunning()**

Can be used for checking whether the scanner application software is running or not. If it finds a running instance of it the function returns 1. If this function doesn't find a running instance of the application it makes no sense to use any of the other Client Control Interface functions.

Function: **long ScGetInterfaceVersion()**

Returns the version number of the used Client Control Interface to make sure that the running instance of the scanner application supports functions etc.

Function: **long ScShutDown()**

ASCII: **long ScCciShutDown()\n**

Terminates the scanner software. After that no more commands should be given.

Function: **long ScShowApp(long Show)**

ASCII: **long ScCciShowApp(long Show)\n**

Allows to switch the scanner application into one of the common show states of Windows.

SW_HIDE **= 0**

Hides the window and passes activation to another window.

SW_SHOWMINIMIZED **= 2**

Activates the window and displays it as an icon.

SW_SHOWMAXIMIZED = 3

Activates the window and displays it as a maximized window.

SW_SHOW = 5

Activates the window and displays it in its current size and position.

SW_RESTORE = 9

Activates and displays the window. If the window is minimized or maximized, Windows restores it to its original size and position.

Function: **long ScMarkEntityByName("@@@Scaps.SpecialTag.3d.All.Layers@@@", long WaitForMarkEnd)**

To use this function the 3D object has to be sliced first. Then this command will mark the whole 3D object, one layer after another. The Flag WaitForMarkEnd can be *1* or *0*. If it is *0* the function returns immediately and the client application can start other tasks while the scanner application is marking in the background. If it set to *1* then the application waits until the marking has finished.

Function: **long ScExecCommand(1)**

Pops up a message box within SAMLight. This can be used to check the communication between the applications.

Function: **long ScLoadJob(BSTR FileName, 1, 1, 1)**

Loads a *.s3d job specified by *FileName* into the controlled scanner application.

Function: **long ScImport(BSTR EntityName, BSTR FileName, BSTR Type, 0, 0);**

Imports a file from the path *FileName*. The name of the new entity will be *EntityName*. Type can be 'stl', 'cli' or 'slc'.

Type can be 'style' as well if you want to import a layer **styles** table. In this case *EntityName* specifies which ScLayerSolid entity you want to assign to this style table and *FileName* has to be a *.txt file.

Function: **long ScExport(BSTR EntityName, BSTR FileName, BSTR Type, 0, 0);**

Exports the layer **styles** table of the ScLayerSolid entity specified with *EntityName* to the *.txt file *FileName*. Please use 'style' as *Type*.

Exports the slices with Entity type ScLayerSolid to a specified path with *fileName*.cli and format *cli. Only one ScLayerSolid can be exported at a time.

Function: **long ScTranslateEntity(BSTR EntityName, double X, double Y, double Z)**

Translates the entity specified by EntityName. If EntityName is an empty string the complete job is translated.

Function: **long ScScaleEntity(BSTR EntityName, double ScaleX, double ScaleY, double ScaleZ)**

Scales the entity specified by EntityName using the scaling factors ScaleX, ScaleY and ScaleZ for the three possible scaling directions. If EntityName is set to an empty string the complete job is scaled.

Function: **long ScRotateEntity3D(BSTR EntityName, double px, double py, double pz, double vx, double vy, double vz, double Angle)**

Rotates the 3D volume entity specified by EntityName. The vector V (0, 0, 0, vx, vy, vz) starting at point P (px, py, pz) defines the rotation axis. The rotation Angle is specified in degrees (°). The rotation is counterclockwise for positive values. Splitted entities can be rotated if vector V (vx, vy, vz) is of the form (0, 0, vz).

Function: **long ScSlice(BSTR EntityName, BSTR LayerSolidName, double sliceThickness, long doSliceOnlySelected, long doReverseDirection);**

Slices the entity specified with *EntityName* with a slice distance *sliceThickness*. The generated ScLayerSolid entity will have the name *LayerSolidName*. The flags *doSliceOnlySelected* and *doReverseDirection* can be 0 or 1. These flags corresponds to the flags in the Slice dialog.

Function: **double ScGetWorkingArea(long Index)**

Returns a single value for the size and dimension of the working area. The index can be one of the constants defined here:

#define SC_SAMLIGHT_OUTLINE_INDEX_MIN_X = 0

#define SC_SAMLIGHT_OUTLINE_INDEX_MIN_Y = 1

#define SC_SAMLIGHT_OUTLINE_INDEX_MIN_Z = 2

#define SC_SAMLIGHT_OUTLINE_INDEX_MAX_X = 3

#define SC_SAMLIGHT_OUTLINE_INDEX_MAX_Y = 4

#define SC_SAMLIGHT_OUTLINE_INDEX_MAX_Z = 5

Function: **long ScSetLongValue(long Type, long Value)**

ASCII: **long ScCciSetLongValue(long Type, long Value)\n**

Sets a long value. The possible values for Type are:

scComSAMLightClientCtrlLongValueTypeOptoIO = 4

Sets the OptoIO bits. Value can be in the range *0..63*

scComSAMLightClientCtrlLongValueTypeJobExecutionDelay = 12

Set the Job Execution Delay to Value. The unit is [ms].

scComSAMLightClientCtrlLongValueTypeSliceFrom = 45

scComSAMLightClientCtrlLongValueTypeSliceTo = 46

If you do not want to mark all slices, you can choose the first

and last slice, which should be marked.

scComSAMLightClientCtrlLongValueTypeCurrentSliceNum = 47

Sets the current selected slice.

Function: **long ScGetLongValue(long Type)**

Reads a long Value. The possible values for Type are:

scComSAMLightClientCtrlLongValueTypeOptoIO = 4

Reads the OptoIO bits.

scComSAMLightClientCtrlLongValueTypeJobExecutionDelay = 12

Reads the Job Execution Delay. The unit is [ms].

scComSAMLightClientCtrlLongValueTypeDongleUserNumber = 39

Reads the Customer ID (User ID).

scComSAMLightClientCtrlLongValueTypeDongleUserNumber = 40

Reads the Dongle ID (System ID, Card ID).

scComSAMLightClientCtrlLongValueTypeGetTotalSlices = 44

Reads the total amount of slices of the current job file.

scComSAMLightClientCtrlLongValueTypeSliceFrom = 45

scComSAMLightClientCtrlLongValueTypeSliceTo = 46

Reads the first/last slice, which will be marked.

scComSAMLightClientCtrlLongValueTypeGetCurrentSliceNum = 47

Reads the current selected slice.

Function: **double ScGetEntityOutline(BSTR EntityName, long Index)**

ASCII: **double ScCciGetEntityOutline(string EntityName, long Index)\n**

Returns the outline of the current slice of the ScLayerSolid specified by *EntityName*. If *EntityName* is an empty string, the function returns the job outline. The following constants can be set for *Index* to specify which outline value has to be returned:

scComSAMLightClientCtrlOutlineSliceIndexMinX = 6

Returns the minimum position in X direction.

scComSAMLightClientCtrlOutlineSliceIndexMinY = 7

Returns the minimum position in Y direction.

scComSAMLightClientCtrlOutlineSliceIndexMinZ = 8

Returns the minimum position in Z direction.

scComSAMLightClientCtrlOutlineSliceIndexMaxX = 9

Returns the maximum position in X direction.

scComSAMLightClientCtrlOutlineSliceIndexMaxY **= 10**

Returns the maximum position in Y direction.

scComSAMLightClientCtrlOutlineSliceIndexMaxZ **= 11**

Returns the maximum position in Z direction.

Function: **double ScGetDoubleValue(long Type)**

Reads a double value. The possible values for Type are:

scComSAMLightClientCtrlDoubleValueTypeLastMarkTime **= 21**

Read the marking time of the last mark. The unit is seconds.

scComSAMLightClientCtrlDoubleValueTypeLast ExpectedMarkTime **= 34**

Read the expected marking time. The unit is seconds.

Function: **long ScGetStringValue(long Type, string Name)**

Reads a string value. The possible values for Type are:

scComSAMLightClientCtrlStringValueTypeGetLast ErrorMessageInput **= 14**

Puts the last error message into the string *Name*.

scComSAMLightClientCtrlStringValueTypeGetLast InfoMessageInput **= 15**

Puts the last info message into the string *Name*.

scComSAMLightClientCtrlStringValueTypeUserValue **= 20000 ... 20009**

Put the string stored with the previous command into the string Name. This command together with the preceding one can be used to store and read job specific data. The first parameter is a decimal value which acts as the same index as in the preceding command.

Function: **long ScSetStringValue(long Type, string Name)**

Sets a string value. The possible values for Type are:

scComSAMLightClientCtrlStringValueTypeUserValue **= 20000 ... 20009**

Put the string value Name into an unused and hidden entity inside the job. The number *20000* acts as index. The values from *20001* to *20009* can be used with the same functionality.

scComSAMLightClientCtrlStringValueTypeSaveLayerAdjustabl eDPI **= 24**

Creates a bitmap of the working area and saves the bitmap file into given path. The resolution is set with ScSetDoubleValue() with type scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapDPI.

scComSAMLightClientCtrlStringValueTypeSaveView3DVariableSiz e **= 25**

scComSAMLightClientCtrlStringValueTypeSaveView3DFull = 26

These types can be used to create a screenshot from the main view and all contained entities. The file name where to save the captured bitmap file is given as a string parameter. The several command types differ only in the maximum picture size (width or height) of the bitmap file: 160 pixels, 320 pixels, custom defined size or full size. This function only works if the main window of the controlled scanner application is visible and not hidden by something else. So the window cannot be minimized and no sscreensaver should be active.

Function: **long ScSetDoubleValue(long Type, double Value)**

Sets a double value. The possible values for Type are:

scComSAMLightClientCtrlDoubleValueTypeUserValue = 20000 ... 20014

Put the double value Value into an unused and hidden entity inside the job. The number *20000* acts as index. The values from *20001* to *20014* can be used with the same functionality.

scComSAMLightClientCtrlDoubleValueTypeSaveView3DBitmapDPI = 44

Sets the resolution of the bitmap, refert to ScSetStringValue() with type *scComSAMLightClientCtrlStringValueTypeSaveLayerAdjustableDPI*.

scComSAMLightClientCtrlDoubleValueTypeSaveView3DBitmapVariableSize = 81

Set width of Bitmap that is created with *scComSAMLightClientCtrlStringValueTypeSaveView2DVariableSize*.

Function: **double ScGetDoubleValue(long Type)**

Reads a double value. The possible values for Type are:

scComSAMLightClientCtrlDoubleValueTypeUserValue = 20000 ... 20014

Read the double value that was set with the previous command. This command together with the preceding one can be used to store and read job specific data. The first parameter is a decimal value which acts as the same index as in the preceding command.

Function: **long ScGetIDStringData(long Type,long DataId, BSTR *Data)**

With this function strings can be retrieved for a specific purpose (parameter *Type*) using an additional attribute *DataId* that may be e.g. an index number or an identifier. For *Type* one of the following values has to be set:

scComSAMLightClientCtrlStringDataIdGetEntityName = 21

This is used to retrieve the name of an entity by its index. The index is zero based and starts from the top level going through the tree on the base of the first object, going on with the siblings.

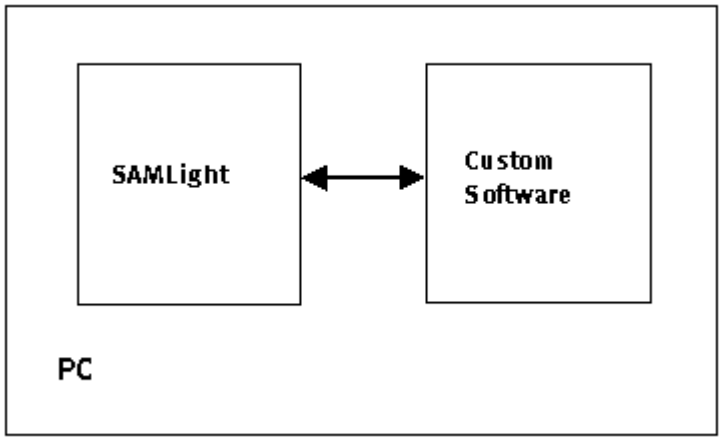
scComSAMLightClientCtrlStringDataIdGetEntityType = 22

This is used to retrieve the type of an entity by its index. The index is zero based and starts from the top level going through the tree on the base of the first object, going

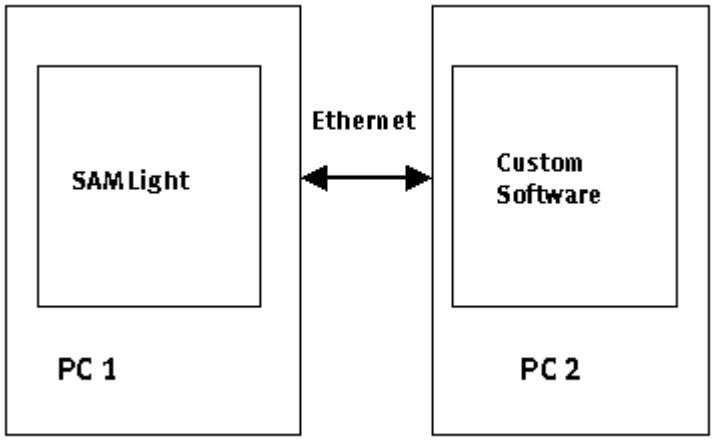
on with the siblings.

19 Client Control Interface

This manual shows how to control SAMLight out of another software, either inside one PC or in between two PCs:



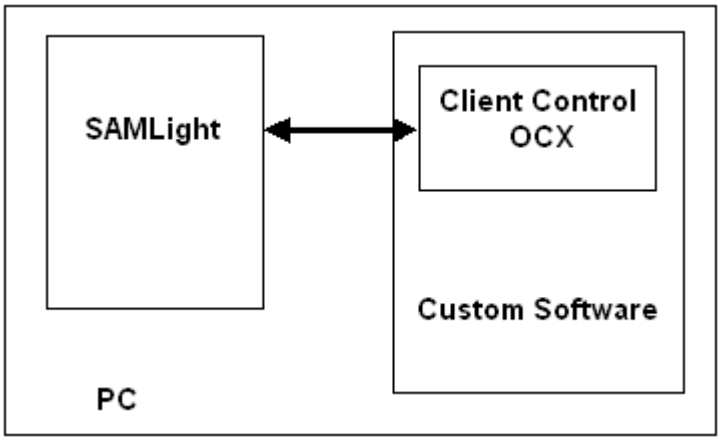
or:



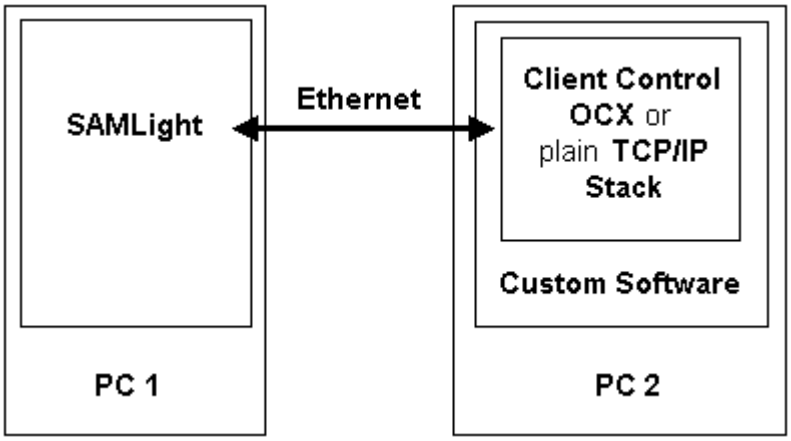
19.1 Principle of operation

The ActiveX control **SCAPS.ScSamlightClientCtrl** provides the functions to control the SAMLight application out of another software. This allows fast development of the Client software with different Windows development environments. Beside of that there exists the possibility to access the Client Control Interface without the protocol encapsulation of the **SCAPS.ScSamlightClientCtrl**. In this case plain ASCII-commands are sent via the TCP/IP protocol stack of an arbitrary system.

Since SAMLight version 3_3_5_0233 the client controls can be used with 32-bit and 64-bit custom software .



or:

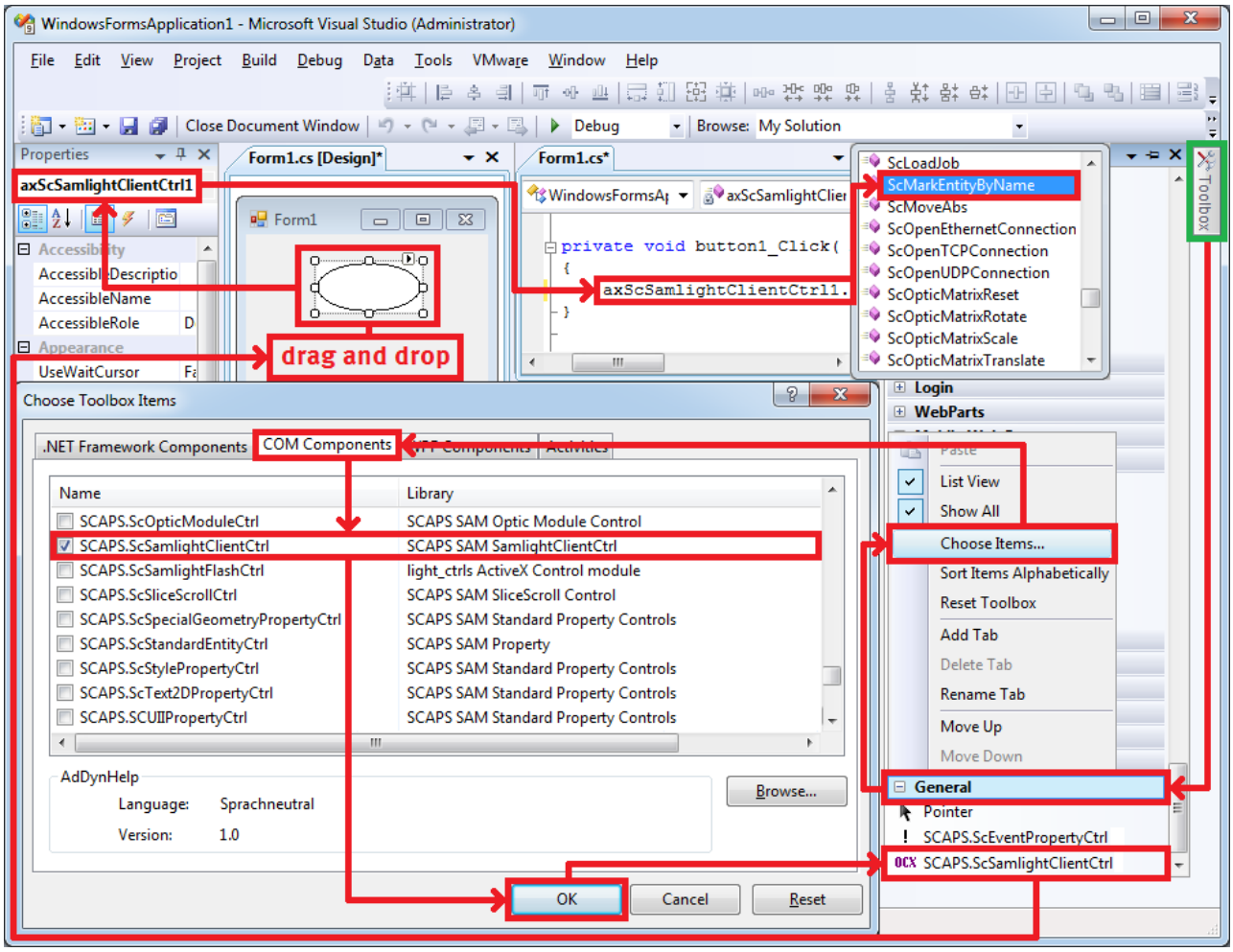


19.2 Implementations

The following implementations of the SAMLight Client Control programming interface are available, which are compatible regarding function calls and constants but different regarding the error handling return values:

1. SAMLight Client Control (sc_samlight_client_ctrl.ocx, since installer 3.3.5 Build 232 to 3.4.5 Build 148 also a 64 bit version is available: sc_samlight_client_ctrl_x64.ocx. Because of a 8.3 filename convention problem with sc_samlight_client_ctrl.ocx the name of the 64 bit version was changed to sc_x64_samlight_client_ctrl.ocx with installer 3.4.5 Build 149.)

This SAMLight Client Control implementation is normally linked via drag and drop of the SAMLight Client Control ocx onto the form of the SAMLight Client Control application.



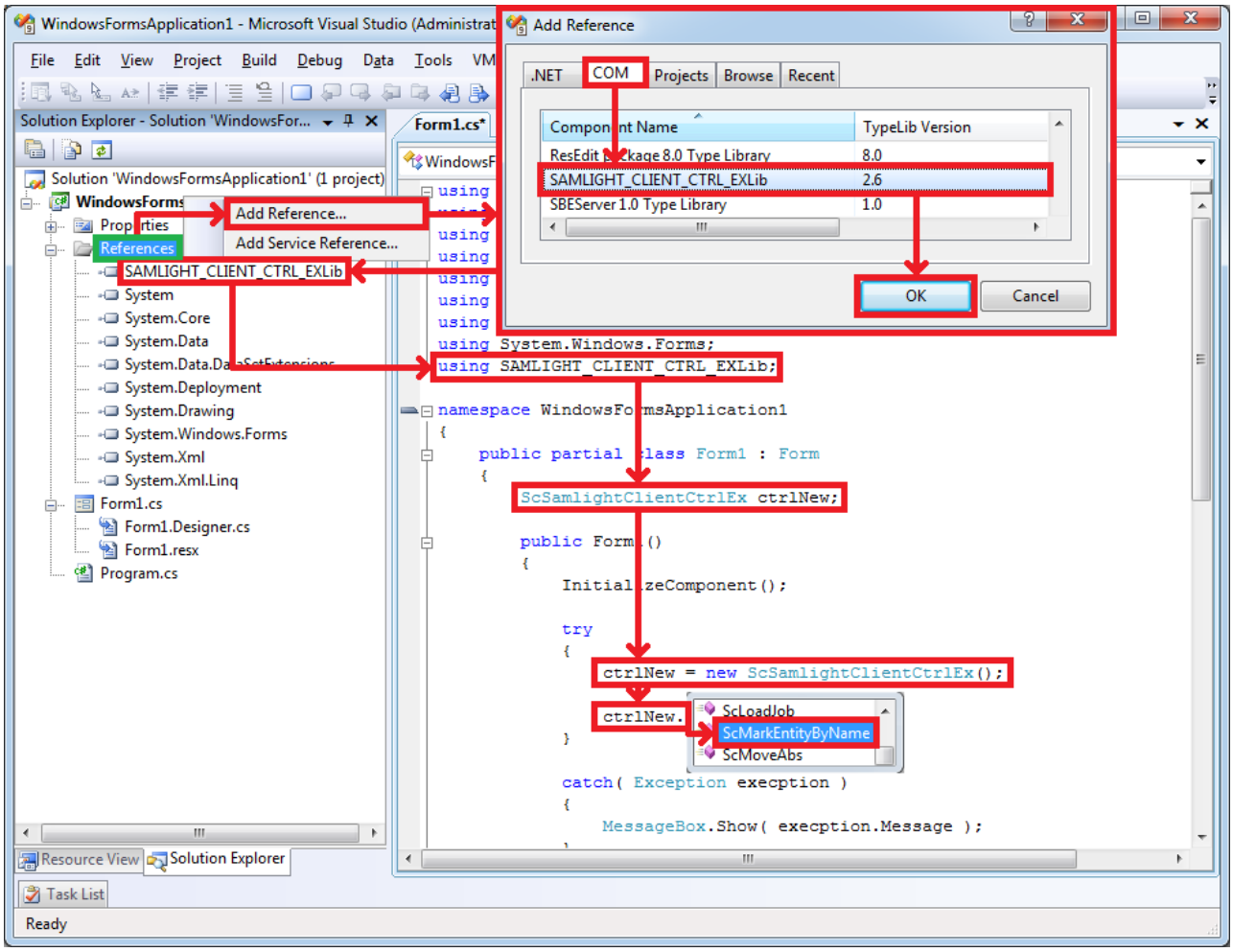
The possible return values of the SAMLight Client Control function calls are normally just 0 or 1, but can also be a long, double or string value.

With the CCI command scComSAMLightClientCtrlLongValueTypeNewCciErrorReturn, more error return values are activated. The possible function error return values for more detailed error analysis are:

- -10000 : IDispatch error #17: no named entity found
- -10001 : IDispatch error #18: job is empty
- -10002 : IDispatch error #19: function returned 0 (not ok)
- -10003 : IDispatch error #20: not allowed in 3D mode
- -10004: IDispatch error #22: no valid license

2. SAMLight Client Control Ex (Ex for Extended, included since installer 3.5.5 Build 0002: 32 bit version sc_ex_samlight_client_ctrl.dll and 64 bit version sc_ex64_samlight_client_ctrl.dll)

The SAMLight Client Control Ex interface uses the same SAMLight Client Control commands and constants but offers for each command a more detailed error return value. SAMLight Client Control Ex is added via Reference.



The possible HRESULT function error return values for more detailed error analysis are:

- 0x80040200 : Unknown error please inform SCAPS by e-mail: info@scaps.com
- 0x80040201 : IDispatch error #1: an event could not call a subscriber: Samlight unavailable
- 0x80040202 : IDispatch error #2: tcp timed out
- 0x80040211 : IDispatch error #17: no named entity found
- 0x80040212 : IDispatch error #18: job is empty
- 0x80040213 : IDispatch error #19: function returned 0 (not ok)
- 0x80040214 : IDispatch error #20: not allowed in 3D mode
- 0x80040215 : IDispatch error #21: no outline because entities are empty
- 0x80040216: IDispatch error #22: no valid license

Under C#, if one of these HRESULT are returned by a ScSamlightClientCtrlEx function, the C# RCW (runtime callable wrapper) of the function converts this error (facility ITF) into a corresponding .Net COM Exception, containing the information above.

SAMLight Client Control Ex cannot be used in combination with TCP ASCII Communication Mode.

19.3 Command Set

The following commands and functions that are described are provided by the SCAPS.ScSamlightClientCtrl control that has to be added to a software project in order to remote-access the scanner software. Most of these functions have a return value of type long. If the detailed description above does not specify a different

behavior this value tells if the function could be sent to the controlled application successfully or not.

Beside of that the description of the plain ASCII commands that are sent via a TCP/IP protocol stack can be found here too. They work using a handshake: Every command sent will be acknowledged by an answer that is at least a single line feed. Normally operations return a *1* in case of success and *0* else. Please note that all commands are single lines of text. So depending on the used programming language numbers have to be converted. Every command line that has to be sent consists of a unique name and a list of parameters encapsulated in brackets. After that such a line has to be finished using a line feed / new line (**\r\n** or **\n**) as delimiter. The answer that is sent after the operation contains the same delimiter.

Directly after a connection to the scanner application was opened, an initialization string "SAM CCIPlain\n" has to be sent if ASCII Communiation Mode checkbox is not activated to set up the communication interface for the ASCII communication mode. Without that initialization ASCII based communication is only possible with ASCII Communiation Mode checkbox activated. All command texts are available as C-defines within a header file ScCciCommands.h. These defines can be used to construct a full command line containing all required parameters. All these names are described in the following order: The name of the function call and the name of the ASCII command. For the ASCII commands there are three data types available: string (some text that has to be quoted fully), long and double.

19.3.1 Application

Function: **long ScIsRunning()**

ASCII: **-**

Can be used for checking whether the scanner application software is running or not. If it finds a running instance of it the function returns *1*. If this function does not find a running instance of the application it makes no sense to use any of the other Client Control Interface functions.

Function: **long ScGetInterfaceVersion()**

ASCII: **long ScCciGetInterfaceVersion()\n**

Returns the version number of the used Client Control Interface to make sure that the running instance of the scanner application supports functions etc.

Function: **long ScShutDown()**

ASCII: **long ScCciShutdown()\n**

Terminates the scanner software. After that no more commands should be given.

Function: **-**

ASCII: **long ScCciTest(string Test)\n**

This function can be used to test the connection to the scanner application. In case of success, the returned value is 1 and the string Test that was sent is displayed by the scanner application within a message box that has to be closed manually.

Function: **long ScShowApp(long Show)**

ASCII: **long ScCciShowApp(long Show)\n**

Allows to switch the scanner application into one of the common show states of Windows. That function is useful to e.g. hide the controlled applications main window completely when a user has to access a custom graphical user interface only that communicates with the main application via this Client Control Interface.

SW_HIDE **= 0**

Hides the window and passes activation to another window.

SW_SHOWMINIMIZED **= 2**

Activates the window and displays it as an icon.

SW_SHOWMAXIMIZED **= 3**

Activates the window and displays it as a maximized window.

SW_SHOW **= 5**

Activates the window and displays it in its current size and position.

SW_RESTORE **= 9**

Activates the window and displays it. When the window is minimized or maximized is, it is reset to the original size and position.

19.3.2 Remote

The following functions have to be used only in case a remote connection via network and the Client Control ActiveX is used instead of direct function calls. There are no ASCII-commands. The functionality provided by these functions has to be implemented by the application itself as described above:

Function: **long ScOpenEthernetConnection(BSTR SenderAddr, long SenderPort, BSTR RecipientAddr, long RecipientPort)**

ASCII: **-**

This function is deprecated. Please use ScOpenTCPConnection() instead.

Function: **long ScOpenTCPConnection(BSTR RecipientAddr, long RecipientPort)**

ASCII: **-**

Opens an TCP connection to the scanner application through the network and returns 1 if the connection could be established successfully. The *RecipientAddr* is the IP and the *RecipientPort* is the Port of the application that has to be accessed using the client control interface as described above. For the recipient IP in every case the real IP of the host systems network interface card has to be entered where the application that has to be remote controlled is running. If that application was configured using *0.0.0.0* it is not allowed to enter that value here.

Function: **long ScCloseEthernetConnection()**

ASCII: **-**

Closes a previously opened connection.

19.3.3 System Settings

The working environment of the scanner application can be modified and explored with the following function set:

Function: **double ScGetWorkingArea(long Index)**

ASCII: **double ScCciGetWorkingArea(long Index)\n**

Returns a single value for the size and dimension of the working area. The index can be one of the constants defined here:

SC_SAMLIGHT_OUTLINE_INDEX_MIN_X	**= 0**
SC_SAMLIGHT_OUTLINE_INDEX_MIN_Y	**= 1**
SC_SAMLIGHT_OUTLINE_INDEX_MIN_Z	**= 2**
SC_SAMLIGHT_OUTLINE_INDEX_MAX_X	**= 3**
SC_SAMLIGHT_OUTLINE_INDEX_MAX_Y	**= 4**
SC_SAMLIGHT_OUTLINE_INDEX_MAX_Z	**= 5**

Function: **long ScOpticMatrixTranslate(double X, double Y, double Z)**

ASCII: **long ScCciOpticMatrixTranslate(double X, double Y, double Z)\n**

With this command it is possible to define an additional translation vector for the marking process.

Function: **long ScOpticMatrixRotate(double CenterX, double CenterY, double Angle)**

ASCII: **long ScCciOpticMatrixRotate(double CenterX, double CenterY, double Angle)\n**

With this command it is possible to define an additional rotation for the marking process. CenterX and CenterY define the rotation middle point and Angle is the rotation angle in radians. Please be careful to use the right decimal separator (. or ,) depending on your operating system.

Function: **long ScOpticMatrixScale(double ScaleX, double ScaleY)**

ASCII: **long ScCciOpticMatrixScale(double ScaleX, double ScaleY)\n**

With this command it is possible to define an additional output scale for the marking process. ScaleX and ScaleY define the scale factor in X and Y direction.

Function: **long ScOpticMatrixReset()**

ASCII: **long ScCciOpticMatrixReset()\n**

Resets the additional transformation output matrix. After the reset of the matrix, previous calls of ScOpticMatrixTranslate() and ScOpticMatrixRotate() become obsolete.

Function: **long ScGetOpticMatrix(long Index,double *Value)**

ASCII: **double ScCciGetOpticMatrix(long Index)\n**

Returns an element of the additional transformation output matrix. With Index values from 0..5 the six matrix elements M0..M5 can be retrieved using the variable the parameter Value points to. The additional transformation matrix is applied to all entities before the marking process. Each point coordinate will be transformed with the matrix elements according the following equations:

$$X_{new} = M_0 * X_{old} + M_1 * Y_{old} + M_2$$

$$Y_{new} = M_3 * X_{old} + M_4 * Y_{old} + M_5$$

Function: **long ScSetHead (long HeadID)**

ASCII: **long ScCciSetHead(long HeadID)\n**

This function is for multihead applications. It specifies a head that is used with the following commands. HeadID can have a number in range 0..n that is the number of the head to be set or -1 to apply commands for all heads.

Function: **long ScGetHead (long *HeadID)**

ASCII: **long ScCciGetHead()\n**

This function is for multihead applications. It returns the current head number that is active for the commands that are sent via CCI. The returned HeadID can have a number in range 0..n that is the number of the active head or -1 if all heads are active.

19.3.4 Mark Settings

The next few functions are marking related. That means as a precondition here some entities have to be available within the scanner application so that marking makes sense:

Function: **long ScIsMarking()**

ASCII: **long ScCciIsMarking()\n**

If the ScMarkEntityByName function was called with *WaitForMarkEnd* set to *0*, this function can be used for checking whether the actual marking process is already finished or not. The Function returns *1* if the scanner application is still marking.

Function: **long ScStopMarking()**

ASCII: **long ScCciStopMarking()\n**

Stops the current marking process.

Function: **long ScSetMarkFlags(long Flags)**

ASCII: **long ScCciSetMarkFlags(long Flags)\n**

The mark flags that can be set using this function define the general behaviour of the output process during the next call of ScMarkEntityByName(). Following marking flags are available and can be combined using a logical OR:

scComSAMLightClientCtrlMarkFlagWaitForTrigger = 1

The Mark function starts only after an external start. The WaitForMarkEnd parameter value of ScMarkEntityByName() flag has no effect.

scComSAMLightClientCtrlMarkFlagHideOutput = 4

Does not show the output window of the controlled scanner application during marking.

scComSAMLightClientCtrlMarkFlagDisableHomeJump = 8

Switch off the home jump after the mark.

scComSAMLightClientCtrlMarkFlagPreview = 16

Execute marking in preview mode. For an example see also: How to precalculate time.

scComSAMLightClientCtrlMarkFlagSelected = 32

Mark only selected entities and ignore all other ones.

scComSAMLightClientCtrlMarkFlagDisablePreProcessing = 64

Disable the pre-processing phase.

scComSAMLightClientCtrlMarkFlagDisablePost Processing = 128

Disable the post-processing phase.

scComSAMLightClientCtrlMarkFlagControlLoopByEntity = 1024

Normally if the Mark Loop Count of an Entity is greater than 1, for example 10, then if marking is active 10 copies of this entity are created internally during download to the card. If this flag is set, only 1 copy is created but looping 10 times on a lower software level. This saves time and memory. Only helpful for very complex jobs with very high loop count and for very high speed marking applications with minimum dead time between two marks. This flag only works for basic vector objects without advanced features as for example pen paths or event objects.

scComSAMLightClientCtrlMarkFlagCloseTriggerWindowAtStop = 2048

Close trigger window, if e.g. a ScStopExecution() is called.

Function: **long ScGetMarkFlags()**

ASCII: **long ScCciGetMarkFlags(long Flags)\n**

Returns the currently active mark flags as described above.

Function: **long ScSwitchLaser(long on_off)**

ASCII: **long ScCciSwitchLaser(long on_off)\n**

Switch the laser on or off depending on the value of the parameter *on_off* (*1* or *0*).

Function: **long ScMoveAbs(double x,double y,double z)**

ASCII: **long ScCciMoveAbs(double x,double y,double z)\n**

Jumps directly with the defined pen to the new position specified by x, y and z. The unit for the position is always mm. The pen has to be specified with ScSetPen() (see below) before ScMoveAbs() is called. If no pen was defined pen # 1 will be taken as default. The values will be affected by the correction file as well. During the jump the laser emission is turned off by default and can be turned on with ScSwitchLaser() (see above).

Function: **long ScSetPen(long pen)**

ASCII: **long ScCciSetPen(long pen)\n**

Defines the pen number that has to be used for the next *ScMoveAbs()* command and related *ScGet/ SetDoubleValue()* functions. Pen counting starts with *1*.

Function: **long ScGetPen(long *pen)**

ASCII: **long ScCciGetPen()\n**

Returns the currently defined and used pen. The function call returns the pen using the pointer pen handed over as parameter while the ASCII command sends back the pens value as a handshake answer. Pen counting starts with *1*.

Function: **-**

ASCII: **long ScCciResetSequence()\n**

The current sequence is reset.

Function: **-**

ASCII: **long ScCciResetCounter()\n**

The quantity counter will be reset.

Function: **-**

ASCII: **long ScCciResetSerialNumbers()\n**

This call resets all serial number entities to their start values.

Function: **-**

ASCII: **long ScCciIncSerialNumbers()\n**

This call increments all serial number entities.

Function: **-**

ASCII: **long ScCciDecSerialNumbers()\n**

This call decrement all serial number entities.

Function: **-**

ASCII: **long ScCciResplitJob()\n**

Resplit a job that is in splitting mode and was modified so that the split data have to be updated by this option.

Function: **long ScSetPixelMapForPen(long pen, long pixelzone0, long pixelzone1, long pixelzone2, long pixelzone3, long pixelzone4, long pixelzone5)**

ASCII: **long ScCciSetPixelMapForPen(long pen, long pixelzone0, long pixelzone1, long pixelzone2, long pixelzone3, long pixelzone4, long pixelzone5)\n**

Set the pen specific gray scale bitmap pixel map (6 zones, zone values can be *0 .. 100* (%), default is *0, 20, 40, 60, 80, 100* for the six zones). The Global gray scale bitmap pixel map is not affected by this. By using a pen specific pixel map, you are able to select different maps (by using different pens) in one job. The pen index is *0 .. maxPen-1*. It is possible to change the global Pixel-Map. Therefore use the pen number *1111*.

19.3.5 Entity Settings

The following functions can be used to manipulate, edit and change single entities within the current job. For most of them a parameter *EntityName* has to be used to specify which entity should be manipulated. As a precondition it is necessary that all entities within a job have non ambiguous, non empty names. These names can be set directly within the application using the related entity property page.

Function: **long ScMarkEntityByName(BSTR EntityName, long WaitForMarkEnd)**

ASCII: **long ScCciMarkEntityByName(string EntityName, long WaitForMarkEnd)\n**

Marks the entity with the name EntityName. If EntityName is an empty string (*""*), the complete job will be marked. If the second parameter *WaitForMarkEnd* is set to *0*, the function returns immediately and the client application can start other tasks while the scanner application is marking in background. *EntityName* together with the flag *scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon* can include more than one entity name. Therefore the names of the entities have to separated by a semicolon. If more than one entity with a given name exists, all of them are marked. If you are in 3D Mode it is possible to mark the whole 3D object. Therefore set the parameter *EntityName* to *@@@Scaps.SpecialTag.3d.All.Layers@@@*.

If ASCII communication is used the return of the CCI commands never wait for mark end. If the job process is finished can be checked with ScIsMarking().

Function: **long ScChangeTextByName(BSTR EntityName, BSTR Text)**

ASCII: **long ScCciChangeTextByName(string EntityName, string Text)\n**

The string of the text object with the given *EntityName* can be set to the string specified by the parameter *Text*. This function allows the dynamic change of text objects. The different text objects are identified by their name. It is also possible to change the text of more than one entities with one function call. Therefore set the Flag *scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon*. Then separate the different *EntityNames* with a semicolon ';'. If you want to give the each entity a different new string separate these strings by a vertical tab in the parameter *Text*. In C and C++ the vertical tab equals a '\v' whereas in Visual Basic the vertical tab is (*vbVerticalTab*). Then the number of *EntityNames* must be the same as the number of names in the parameter *Text*. An empty string for a name in the parameter *Text* is valid too.

Function: **double ScGetEntityOutline(BSTR EntityName, long Index)**

ASCII: **double ScCciGetEntityOutline(string EntityName, long Index)\n**

Returns the outline of the entity with *EntityName*. If *EntityName* is an empty string, the function returns the job outline. Consider that the *Z values* only give correct values if the optional *Optic3D/FlatLense* package is installed and enabled. Also the calculation of the *Z values* might take some time depending on the size of the object. The following constants can be set for *Index* to specify which outline value has to be returned:

scComSAMLightClientCtrlOutlineIndexMinX **= 0**

Returns the minimum position in X direction

scComSAMLightClientCtrlOutlineIndexMinY **= 1**

Returns the minimum position in Y direction

scComSAMLightClientCtrlOutlineIndexMinZ **= 2**

Returns the minimum position in Z direction

scComSAMLightClientCtrlOutlineIndexMaxX **= 3**

Returns the maximum position in X direction

scComSAMLightClientCtrlOutlineIndexMaxY **= 4**

Returns the maximum position in Y direction

scComSAMLightClientCtrlOutlineIndexMaxZ **= 5**

Returns the maximum position in Z direction

Function: **long ScGetEntityOutline2D(BSTR EntityName, double *MinX, double *MinY, double *MaxX, double *MaxY)**

ASCII: **long ScCciGetEntityOutline2D(string EntityName)\n**

With this command the total 2D outlines of all entities with *EntityName* can be get. If *EntityName* is an empty string, you get the outline of the whole job.

Function: **long ScGetEntityOutline3D(BSTR EntityName double *MinX, double *MinY, double *MaxX, double *MaxY, double *MinZ, double *MaxZ)**

ASCII: **long ScCciGetEntityOutline3D(string EntityName)\n**

With this command the total 3D outlines of all entities with *EntityName* can be get. If *EntityName* is an empty string, you get the outline of the whole job.

Function: **long ScTranslateEntity(BSTR EntityName, double X, double Y, double Z)**

ASCII: **long ScCciTranslateEntity(string EntityName, double X, double Y, double Z)\n**

Translates the entity specified by *EntityName*. If *EntityName* is an empty string the complete job is translated.

Function: **long ScScaleEntity(BSTR EntityName, double ScaleX, double ScaleY, double ScaleZ)**

ASCII: **long ScCciScaleEntity(string EntityName, double ScaleX, double ScaleY, double ScaleZ)\n**

Scales the entity specified by *EntityName* using the scaling factors *ScaleX*, *ScaleY* and ScaleZ for the three possible scaling directions. If *EntityName* is set to an empty string the complete job is scaled.

Function: **long ScScaleEntityCurPos(BSTR EntityName, double ScaleX, double ScaleY, double ScaleZ)**

ASCII: **long ScCciScaleEntityCurPos(string EntityName, double ScaleX, double ScaleY, double ScaleZ)\n**

Scales the entity specified by *EntityName* using the scaling factors *ScaleX*, *ScaleY* and ScaleZ for the three possible scaling directions. The entity is scaled at the current position, so the center of the entity before and after scaling stays at the same position. If *EntityName* is set to an empty string the complete job is scaled.

Function: **long ScRotateEntity(BSTR EntityName, double X, double Y, double Angle)**

ASCII: **long ScCciRotateEntity(string EntityName, double X, double Y, double Angle)\n**

Rotates the entity specified by *EntityName*. The origin of the rotation is specified by the parameters *X* and *Y*. The rotation *Angle* is specified in degrees (°). Please be careful to use the right decimal separator (.) or (,) depending on your operating system. The rotation is counterclockwise for positive values. If *EntityName* is an empty string the complete job is rotated.

Function: **long ScRotateEntity3D(BSTR EntityName, double px, double py, double pz, double vx, double vy, double vz, double Angle)**

ASCII: **long ScCciRotateEntity3D(string EntityName, double px, double py, double pz, double vx, double vy, double vz, double Angle)\n**

Rotates the entity specified by EntityName. The vector V (0, 0, 0, vx, vy, vz) starting at point P (px, py, pz) defines the rotation axis. The rotation *Angle* is specified in degrees (°). The rotation is counterclockwise for positive values.

Function: **long ScSetEntityLongData(BSTR EntityName, long DataId, long Data)**

ASCII: **long ScCciSetEntityLongData(string EntityName, long DataId, long Data)\n**

Sets specific data of the entity *EntityName* depending on the *DataId*. For a list of the valid *DataIds* and the corresponding data that can be used here see Long Data Ids.

Function: **long ScGetEntityLongData(BSTR EntityName, long DataId)**

ASCII: **long ScCciGetEntityLongData(string EntityName, long DataId)\n**

Gets specific data of the entity *EntityName* depending on the *DataId*. For a list of the valid *DataIds* and the corresponding return value please refer to Long Data Ids.

Function: **long ScSetEntityDoubleData(BSTR EntityName, long DataId, double Data)**

ASCII: **long ScCciSetEntityDoubleData(string EntityName, long DataId, double Data)\n**

Sets specific data of entity *EntityName* depending on the *DataId*. For a list of the valid *DataIds* and the corresponding Data see Double Data Ids.

Function: **long ScGetEntityDoubleData(BSTR EntityName, long DataId, double *Data)**

ASCII: **double ScCciGetEntityDoubleData(string EntityName, long DataId)\n**

Gets specific data of entity *EntityName* depending on the *DataId*. For a list of the valid *DataIds* and the corresponding return value see Double Data Ids.

Function: **long ScSetEntityStringData(BSTR EntityName, long DataId, BSTR Data)**

ASCII: **long ScCciSetEntityStringData(string EntityName, long DataId, string Data)\n**

Sets specific data of entity EntityName depending on the DataId. For a list of the valid DataIds and the corresponding Data see String Data Ids.

Function: **long ScGetEntityStringData(BSTR EntityName, long DataId, BSTR *Data)**

ASCII: **string ScCciGetEntityStringData(string EntityName, long DataId)\n**

Gets specific data of entity *EntityName* depending on the *DataId*. For a list of the valid *DataIds* and the corresponding return value see String Data Ids.

Function: **long ScDeleteEntity(BSTR EntityName)**

ASCII: **long ScCciDeleteEntity(string EntityName)\n**

Deletes the entity with the name *EntityName*.

Function: **long ScDuplicateEntity(BSTR EntityName, BSTR DuplicatedEntityName)**

ASCII: **long ScCciDuplicateEntity(string EntityName, string DuplicatedEntityName)\n**

Copies the entity with the name *EntityName* and generates an entity with the name *DuplicatedEntityName*.

19.3.6 Pen Settings

The following two functions can be used to set or get pen paths.

Function: **long ScGetPenPathForPen(short *pen, short *enable, short *penToUse1, long *loopOfPenToUse1, .., short *penToUse5, long *loopOfPenToUse5)**

ASCII: -

Use this function to get the pen path for the selected pen. The only parameter to specify is pen. The others will be read out of the pen. The return value will be 0 if there was an error

---

Function: **long ScSetPenPathForPen(short pen, short enable, short penToUse1, long loopOfPenToUse1, .., short penToUse5, long loopOfPenToUse5)**

ASCII: -

Use this function to set the pen path for the selected pen. You have to define all parameters. The return value will be 0 if there was an error

---

Example: Here you can read a programming example in C#. The code will go through pen 1 to 254 and will change the enable flag either from enabled to disabled or in the opposite direction.

```
int res = 0;

sc_com_pen_path penPath = new sc_com_pen_path();

penPath.penToUse = new short[ 5 ];
penPath.loopOfPenToUse = new int[ 5 ];

for( short i = 1; i < 254; i++ )
{
      penPath.pen = i;
      res = axScSamlightClientCtrl1.
            ScGetPenPathForPen( ref penPath.pen,
                                ref penPath.enable,
                                ref penPath.penToUse[ 0 ],
                                ref penPath.loopOfPenToUse[ 0 ],
                                ref penPath.penToUse[ 1 ],
                                ref penPath.loopOfPenToUse[ 1 ],
                                ref penPath.penToUse[ 2 ],
                                ref penPath.loopOfPenToUse[ 2 ],
                                ref penPath.penToUse[ 3 ],
                                ref penPath.loopOfPenToUse[ 3 ],
                                ref penPath.penToUse[ 4 ],
                                ref penPath.loopOfPenToUse[ 4 ] );
      Debug.Assert( res == 1 );

      if( penPath.enable == 1 )
      {
            penPath.enable = 0;
      }
      else
      {
            penPath.enable = 1;
```

```
      }

      res = axScSamlightClientCtrl1.
            ScSetPenPathForPen( ref penPath.pen,
                                ref penPath.enable,
                                ref penPath.penToUse[ 0 ],
                                ref penPath.loopOfPenToUse[ 0 ],
                                ref penPath.penToUse[ 1 ],
                                ref penPath.loopOfPenToUse[ 1 ],
                                ref penPath.penToUse[ 2 ],
                                ref penPath.loopOfPenToUse[ 2 ],
                                ref penPath.penToUse[ 3 ],
                                ref penPath.loopOfPenToUse[ 3 ],
                                ref penPath.penToUse[ 4 ],
                                ref penPath.loopOfPenToUse[ 4 ] );
      Debug.Assert( res == 1 );
}
```

19.3.7 Job Commands

In the following some functions are described that handle complete jobs and entity datasets in general:

Function: **long ScLoadJob(BSTR FileName, long LoadEntities, long OverwriteEntities, long LoadMaterials)**

ASCII: **long ScCciLoadJob(string FileName, long LoadEntities, long OverwriteEntities, long LoadMaterials)\n**

Loads the SJF or S3D job file specified by *FileName* into the controlled scanner application. A SJF job file can contain graphical data (entities) and / or scanner and laser parameters (materials). With the additional parameters it is possible to load only the graphical data (*LoadEntities=1*, *LoadMaterials=0*) or to load only the scanner parameters (*LoadEntities=0*, *LoadMaterials=1*) or load both (*LoadEntities=1*, *LoadMaterials=1*). If the parameter *OverwriteEntities* is set to *1*, the current job will be cleared before the new one is loaded. If *OverwriteEntities* is set to *0*, it is possible to merge the graphical information of different jobs. For S3D job files only the *OverwriteEntities* parameter is valid to control if the jobs have to be merged *0* or not *1*. The parameters *LoadEntities* and *LoadMaterials* are ignored.

Function: **long ScImport(BSTR EntityName, BSTR FileName, BSTR Type, double Resolution, long Flags)**

ASCII: **long ScCciImport(string EntityName, string FileName, string Type, double Resolution, long Flags)\n**

Imports a job file or a graphic from the file *FileName*. *FileType* can be any extension which is also available when using the import dialog. The newly created object will have the name that is specified by the parameter *EntityName*. Resolution defines the input resolution for some filters as PLT. *Flags* can have a combination of following values:

scComSAMLightClientCtrlImportFlagKeepOrder = 8

Keep the order of the entities.

scComSAMLightClientCtrlImportFlagReadPenInfo = 128

Read pen information.

scComSAMLightClientCtrlImportFlagGcodeLoadPenParameters = 8192

Load pen parameters of a CNC file to the head specified with

ScSetHead before. See Import Advanced for CNC.

scComSAMLightClientCtrlImportFlagGcodeLoadOpticParameters = 16384

Load optic parameters of a CNC file to the head specified with ScSetHead before. See Import Advanced for CNC.

scComSAMLightClientCtrlImportFlagPointCloud = 16384

Import of points into *PointClouds*.

scComSAMLightClientCtrlImportFlagNotTryToClose = 131072

Do not try to close polygons. This option applies to HPGL type files only.

scComSAMLightClientCtrlImportFlagOptimized = 524288

Only for HPGL files: Will do the same as the checkbox in the import dialog: Closing Polylines which end close to each other and delete double ones or ones that are very short.

scComSAMLightClientCtrlImportFlagBitmapReimport = 1048576

Reimports a Bitmap with *EntityName*.

scComSAMLightClientCtrlImportFlagDontUpdateView = 2097152

Do not update view when importing. This can be used to import several files quickly and update the view manually after the last import has been done.

scComSAMLightClientCtrlImportFlagVectorReimport = 8388608

Reimports Vector Graphics like DXF and PLT with *EntityName*.

scComSAMLightClientCtrlImportFlagToplevelOnly = 16777216

Reimports only entities in the first level of the job.

scComSAMLightClientCtrlImportFlagFillWithDefaultHatchStyle = 33554432

Only for SVG files. Will do the same as the checkbox in the import dialog: Hatch filled objects with the default hatch.

scComSAMLightClientCtrlImportFlagNoErrorMsg = 67108864

Suppresses Error Messages.

scComSAMLightClientCtrlImportFlagCreateOneGroup = 134217728

Automatically do Data Wizard → Create One Group.

scComSAMLightClientCtrlImportFlagCenterToField = 268435456

Center the import in the View2D.

scComSAMLightClientCtrlImportFlagImportToPenGroups = 536870912

Will do the same as the checkbox in the import dialog.

scComSAMLightClientCtrlImportFlagUsePenColors = 1073741824

Will do the same as the checkbox in the import dialog. This should be used with command 128.

scComSAMLightClientCtrlImportFlagProtected = 2147483648

Polyline and LineArray entites imported with this flag will not be shown in the View2D and will not be splitted. Vectors of entities imported with this flag will not be stored in a *.sjf file. However, it is possible to remember the last position of the protected entities within the *.sjf file. If you re-import an entity with the same name, the old position will be adjusted automatically. Protected entities can be rotated and translated but not scaled. The redpointer pre-view will only show the outline of protected entities.

Function: **long ScExport(BSTR EntityName, BSTR FileName, BSTR Type, double Resolution, long Flags)**

ASCII: -

Exports the job file or the specified entity to FileName. If EntityName is "" then the whole job is exported. Type can be "saf", "plt" or "cnc". Resolution should be set to 0.001 to keep the original size of the job. Flags can be the following values, which can be OR operated if several of them are needed:

scComLayerFile2DStyleExportPolyLines **= 16**

scComLayerFile2DStyleExportLineArrays **= 32**

scComLayerFile2DStyleCheckOrientation **= 64**

scComLayerFile2DStyleWritePens **= 256**

scComLayerFile2DStyleExportOnlySelected **= 1024**

scComLayerFile2DStyleWritePreview **= 8192**

Function: **long ScSaveJob(BSTR FileName, long Flags)**

ASCII: **long ScCciSaveJob(string FileName, long Flags)\n**

Saves the job to the file specified by *FileName* using the SJF file format. The *Flags* parameter can be a logical OR-combination of the following values:

scComSAMLightSaveFlagEntities **= 1**

Saves the entities to the job file.

scComSAMLightSaveFlagMaterials **= 2**

Saves the pens to the job file.

scComSAMLightSaveFlagUseCurrentJobName **= 4**

Uses the current job file name of the scanner application. The parameter *FileName* will be ignored in this case.

Function: **-**

ASCII: **long ScCciNewJob()\n**

The current job is deleted completely.

Function: **-**

ASCII: **long ScCciFitViewToWorkingArea()\n**

Change the view within the scanner application to view the full working area.

Function: **-**

ASCII: **long ScCciFitViewToEntities()\n**

Change the view within the scanner application to view all available entities.

Function: **-**

ASCII: **long ScCciFitViewToSelectedEntities()\n**

Change the view within the scanner application to view all selected entities.

Function: **long ScProcessFlashJob (string FileName, long JobNum, long Mode, long Flags)**

ASCII: **long ScCciProcessFlashJob(string FileName, long JobNum, long Mode, long Flags)\n**

This command allows you to access the flash jobs, see Flash Jobs and Settings. *Flags* must be 0. The *Mode* can have the following values:

scComSAMLightClientCtrlProcessFlashJobMode = 1
StoreCurrentToFlash

For standalone mode, the Flash job (UNF) is created from the SAMLight job (SJF) together with a subset of the SAMLight settings and is saved on the USC card and in <SCAPS>\jobfiles. This means, the creation of the identical UNF from the SJF always requires identical SAMLight settings. Furthermore, a copy of the SJF is also stored on the USC card. This mode of ScProcessFlashJob is only possible if the SJF already exists.
JobNum defines the flash job number. *FileName* and *Flags* are not used.

scComSAMLightClientCtrlProcessFlashJobMode = 2
LoadFromFlash

Loads the copy of the SJF job (which is stored on the USC) to SAMLight. JobNum defines the flash job number. However, the USC settings and the UNF cannot be loaded to SAMLight.
Please note: Different SAMLight settings in combination with the same SJF will lead to different UNFs.
FileName and *Flags* are not used.

scComSAMLightClientCtrlProcessFlashJobMode = 3
StoreFromDiskToFlash

Saves a job (*.unf or *.cnc) from hard disc (PC) to the USC flash.
JobNum defines the flash job number. *FileName* defines the complete path to the UNF file. *Flags* are not used.

Function: **long ScFlashCommand (string Command, long Flags, string Return)**

ASCII: **-**

Executes a command in the flash mode. The string *Command* is the flash command as you would type it in inside the flash dialog window and it has to be terminated by the ASCII characters 13 and 10 (Return and LineFeed). For the flash commands see the corresponding hardware manual. *Return* is an empty string variable that you have to declare in advance. *Flags* can be *0* or *2147483648*. If *0*, the client control waits until a carriage return from the Flash. If *2147483648* is set the client control will wait a certain time and then end the command.

Example for starting a marking:

```
String a = "";
axScSamlightClientCtrl1.ScFlashCommand( "M 1\r\n", 0, ref a );
MessageBox.Show( a );
```

19.3.8 General Commands

Function: **long ScExecCommand(long CmdID)**

ASCII: **long ScCciExecCommand(long CmdID)\n**

A specific command is executed. The parameter *CmdID* specifies what kind of command is executed, see Long CmdIDs.

Function: **-**

ASCII: **ScCciMotionGo**

Activates the motion that was previously defined with the motion drive related constants.

Function: **long ScSetDoubleValue(long Type,double Value)**

ASCII: **long ScCciSetDoubleValue(long Type,double Value)\n**

Sets different double values used for some functions. See the definition of the available type that specifies the kind of operation handed over using parameter Type in Double Value Types.

Function: **double ScGetDoubleValue(long Type)**

ASCII: **double ScCciGetDoubleValue(long Type)\n**

Returns the different double values used for some functions. The available types that can be handed over as parameter Type are defined in Double Value Types. If the operation failed for the ASCII command interface, the returned value is *NaN*.

Function: **long ScSetLongValue(long Type,long Value)**

ASCII: **long ScCciSetLongValue(long Type,long Value)\n**

Sets different long values used for some functions. See the definition of the available types in Long Value Types.

Function: **long ScGetLongValue(long Type)**

ASCII: **long ScCciGetLongValue(long Type)\n**

Returns the different long values used for some functions. The available types that can be used for parameter Type are defined in Long Value Types.

Function: **long ScSetStringValue(long Type,BSTR Value)**

ASCII: **long ScCciSetStringValue(long Type,string Value)\n**

Set different string values used for some functions. See the definition of the available types in String Value Types.

Function: **long ScGetStringValue(long Type, BSTR* Value)**

ASCII: **string ScCciGetStringValue(long Type)\n**

Returns the different string values used for some functions. The available types are defined in String Value Types. If the operation fails for the ASCII-command interface, the returned value is an empty string ("").

Function: **long ScSetStringLongValue(long Type,BSTR SValue,long LValue)**

ASCII: **long ScCciSetStringLongValue(long Type,string SValue,long LValue)\n**

Set different string values and a corresponding long value used for some functions. This command is normally used with a command set or target specified by *Type*. Then the string *SValue* is a more detailed definition of the command or specifies the operation exactly. See the definition of the available types in String Value Types.

Function: **long ScSetStringDblValue(long Type,BSTR SValue,double DValue)**

ASCII: **long ScCciSetStringDoubleValue(long Type,string SValue,double LValue)\n**

Set different string values and a corresponding double used for some functions. This command is normally used with a command set or target specified by *Type*. That means the string *SValue* is a more detailed definition of the command or specifies the operation of the operational complex *Type* exactly. See the definition of the available types in String Value Types.

Function: **long ScGetStringDblValue(long Type,BSTR SValue,double *DValue)**

ASCII: **double ScCciGetStringDoubleValue(long Type,string SValue)\n**

Retrieves a double value used for some functions. This command is normally used with a command set or a target specified by *Type*. That means the string *SValue* is a more detailed definition of the command or specifies the operation of the operational complex *Type* exactly. See the definition of the available types in String Value Types. If the operation failed for the ASCII command interface, the returned value is *NaN*.

Function: **long ScGetIDStringData(long Type,long DataId, BSTR *Data)**

ASCII: **string ScCciGetIDStringData(long Type,long DataId)\n**

With this function strings can be retrieved for a specific purpose (parameter *Type*) using an additional attribute *DataId* that is an index number. For *Type* one of the appropriate scComSAMLightClientCtrlStringDataId constants has to be set.

Function: **long ScSetIDStringData(long Type,long DataId, BSTR *Data)**

ASCII: **string ScCciSetIDStringData(long Type,long DataId)\n**

With this function strings can be assigned to entities using an additional *DataId* that is an index number. For *Type* one of the appropriate scComSAMLightClientCtrlStringDataId constants has to be set.

Function: **-**

ASCII: **long ScCciAutoCompensateOff()\n**

Scanner auto-calibration functionality (works only with hardware that supports it): turn auto calibration mode off.

Function: **-**

ASCII: **long ScCciAutoCompensateRef()\n**

Scanner auto-calibration functionality (works only with hardware that supports it): turn auto calibration mode on.

Function: **-**

ASCII: **long ScCciAutoCompensateCal()\n**

Scanner auto-calibration functionality (works only with hardware that supports it): recalibrate the scan head.

Function: **long ScSetMode(long Flags)**

ASCII: **long ScCciSetMode(long Flags)\n**

The following mode flags can be set using this function to define the general behavior of the software. The adjusted mode is valid for all following function calls. These flags are available and can be combined using a logical OR:

scComSAMLightClientCtrlModeFlagTopLevelOnly = 1

The next calls only search for objects in the top level of the job. This can be helpful to increase the performance.

scComSAMLightClientCtrlModeFlagDontUpdateView = 2

Suppress drawing of entities in the view or in the entity list. This can be helpful to increase the performance.

scComSAMLightClientCtrlModeFlagEntityNames SeparatedBySemicolon = 32

The parameter *EntityName* in *ScMarkEntityByName()* and others can include more than one entity name. They have to be separated by a semicolon. If the client control is used via a TCP connection then the maximum length of the parameter *EntityName* is *512* characters (*511* + an ending zero).

Function: **long ScGetMode()**

ASCII: **long ScCciGetMode()\n**

Returns the current active mode flag as described above.

19.3.9 Async Mode

It is possible to set the ClientCtrl globally in an Async Mode. See the following C# example:

```
axScSamlightClientCtrl1.ScSetLongValue((int)
 ScComSAMLightClientCtrlValueTypes.
 scComSAMLightClientCtrlLongValueTypeClientCtrlAsyncMode, 1 );
```

This command generates a new thread in which the following program is executed. So it runs parallel with all the other tasks of the system.

```
long res = axScSamlightClientCtrl1.ScLoadJob(
  "C:\\scaps\\sam2d\\jobfiles\\barcode.sjf", 1, 1, 1 );
Debug.Assert( res == 1 );
// If not 1, SAMLight is not started or this application is not running as administrator under
Windows Vista/7/8 or former command already running
while(axScSamlightClientCtrl1.ScGetLongValue((int)
  ScComSAMLightClientCtrlValueTypes.
  scComSAMLightClientCtrlLongValueTypeClientCtrlAsyncModeIsRunning )
  != 0 )
{
  Application.DoEvents();
}
res=axScSamlightClientCtrl1.ScGetLongValue((int)
 ScComSAMLightClientCtrlValueTypes.
 scComSAMLightClientCtrlLongValueTypeClientCtrlAsyncModeResult );
Debug.Assert( res == 1 );
MessageBox.Show( "Ready" );
```

This is implemented for the following commands:

- .ScLoadJob
- .ScMarkEntityByName
- .ScGetStringValue
- .ScExecCommand
- .ScImport
- .ScGetEntityOutline
- .ScTranslateEntity
- .ScRotateEntity
- .ScScaleEntity

When using a double return value like with ScGetOutline() one has to use ScGetDoubleValue(scComSAMLightClientCtrlDoubleValueTypeAsyncModeResult) in order to get the return value. A value of HUGE_VAL (+Infinity) indicates an error.

When using a string return value, ScGetStringValue(scComSAMLightClientCtrlStringValueTypeAsyncModeResult) can be used.

If you want to use a ClientCtrl command normally again you have to call the following function before:

```
axScSamlightClientCtrl1.ScSetLongValue((int)
  ScComSAMLightClientCtrlValueTypes.
  scComSAMLightClientCtrlLongValueTypeClientCtrlAsyncMode, 0 );
```

19.4 Constants

Constant Types	Related CCI Functions	
Long Value Types	ScSetLongValue()	ScGetLongValue()
Double Value Types	ScSetDoubleValue()	ScGetDoubleValue()
String Value Types	ScSetStringValue()	ScGetStringValue()
Long Data Ids	ScSetEntityLongData()	ScGetEntityLongData()
Double Data Ids	ScSetEntityDoubleData()	ScGetEntityDoubleData()
String Data Ids	ScSetEntityStringData()	ScGetEntityStringData()
Long CmdIDs	ScExecCommand()	

Table 35: Client Control Interface Constant Types

19.4.1 Long Value Types

These types can be used for the Type-parameter of the functions ScSetLongValue() and ScGetLongValue().

---

scComSAMLightClientCtrlLongValueTypeNumMarksCompleted = 1

Returns the number of marks. This is helpful in Mark Trigger mode.

---

scComSAMLightClientCtrlLongValueTypeHeadStatus = 2

Returns the status of the Scanhead. The following values can be returned, also a combination of them:

Temperature OK = 1

Power OK = 2

Position OK = 4

---

scComSAMLightClientCtrlLongValueTypeOptoIO = 4

This value is used to set the outputs or get the current input states. The pin to bit mapping of the inputs and outputs can be found in the IO chapter. An example can be found in the Get and set outputs and inputs chapter.

OPTO_OUT 0 is not influenced by this command since it is reserved for the mark in progress flag.

---

scComSAMLightClientCtrlLongValueTypeDeviceEnableFlagsSet = 8

The flag is related to the currently used pen. By default the *FlagsSet* is *0*, following values can be set:

scComStandardDeviceEnableFlagGroupStyle = 0

scComStandardDeviceEnableFlagGroupOptoOut = 1

Only with scComSAMLightClientCtrlLongValueTypeDeviceEnableFlagsValue a value is set.

---

scComSAMLightClientCtrlLongValueTypeDeviceEnableFlagsValue = 9

The flag is related to the currently used pen, ScSetPen(). It is used to specify a constant of type *scComStandardDeviceEnableFlagConstants* that has to be set or get for the previously defined *FlagsSet*.

The flag has to be enabled with scComSAMLightClientCtrlLongValueTypeDeviceEnableFlagsSet before.

Values for scComStandardDeviceEnableFlagGroupStyle:

scComStandardDeviceStyleFlagEnableLongDelay = 1

Enables Time delay between pen alternation

scComStandardDeviceStyleFlagEnablePortLaser = 2

Enables the currently selected Laser Port

scComStandardDeviceStyleFlagEnableWobble = 8

Enables Wobble

scComStandardDeviceStyleFlagEnableYAGStandBy = 32

Enables YAG StandBy

scComStandardDeviceStyleFlagEnablePixelOutput = 64

Sets Pixel Hardware Mode

scComStandardDeviceStyleFlagEnablePort1 = 4

scComStandardDeviceStyleFlagEnablePort2 = 128

scComStandardDeviceStyleFlagEnableDA1 = 512

scComStandardDeviceStyleFlagEnableDA2 = 1024

scComStandardDeviceStyleFlagEnableSkyWriting = 8192

scComStandardDeviceStyleFlagEnablePointUsePowerMap = 4194304

Values for scComStandardDeviceEnableFlagGroupOptoOut:

Value corresponds to *OptoOut* bits.

E.g. '5' (binary: 101) sets bit 0 and bit 2

---

scComSAMLightClientCtrlLongValueTypeTotalEntityNum = 10

By using this type parameter it is possible to count the total number of available entities including the ones that are nested in groups.

scComSAMLightClientCtrlLongValueTypeToplevelEntityNum = 11

This parameter can be used together with ScGetLongValue() to evaluate the top level entities. Using this value the number of entities is returned which is visible on the highest level, excluding the contents of groups.

scComSAMLightClientCtrlLongValueTypeJobExecutionDelay = 12

This type can be used to get/set the global parameter *job execution delay*, adjustable within the IO settings page. The unit is [ms].

scComSAMLightClientCtrlLongValueTypeBufferQueueLength = 14

This value can be used to get and set the command queue that can be accessed using the command ScSetEntityLongData() together with the flag *scComSAMLightClientCtrlStringDataIdFlagEnqueueCtrlCmd* and *scComSAMLightClientCtrlStringDataIdFlagEnqueueLastCtrlCmd* (please see below). If it is used it returns the current fill state of the internal command queue. By using the set command together with this option and together with the value *0* the internal command queue for buffered trigger mode job modifications is flushed completely and all enqueued commands are removed. Other values than *0* are invalid for the set operation. For more information about the queue functionality please refer to the Examples section.

scComSAMLightClientCtrlLongValueTypeQuantity = 5

Gets or sets the current quantity if the quantity counter is enabled.

scComSAMLightClientCtrlLongValueTypeMaxQuantity = 15

Gets or sets the current maximum quantity, after that a message is shown. When a value of *-1* is set, the quantity counter functionality is disabled.

scComSAMLightClientCtrlLongValueTypeOverridePen = 16

This constant defines the pen number that is used to mark all entities in the job. The pen numbers that are assigned to these entities in the View2D will be override. If a value of *0* is set then this functionality is disabled.

scComSAMLightClientCtrlLongValueTypeMotionAxis = 17

Axis value: *0* to *6*, or *-1* for all axis.

scComSAMLightClientCtrlLongValueTypeMotionWaitForEnd = 18

Value: *0* or *1*. *1* defines that the application waits until the motion has finished. With *0* the application comes back immediately. The default value is *1*. Note: the value *0* is not possible for homing.

scComSAMLightClientCtrlLongValueTypeMotionMoving = 19

Returns the state of motion defined with scComSAMLightClientCtrlLongValueTypeMotionAxis.

scComSAMLightClientCtrlLongValueTypeSpiG3Waveform = 20

Set or get Waveform for Spi G3 Laser.

scComSAMLightClientCtrlLongValueTypeSpiG3Cw = 21

Set or get CW mode for Spi G3 Laser. 0 = off, 1 = on.

scComSAMLightClientCtrlLongValueTypeLineRampingPowerStartRampActive = 22

scComSAMLightClientCtrlLongValueTypeLineRampingPowerEndRampActive = 23

scComSAMLightClientCtrlLongValueTypeLineRampingSpeedStartRampActive = 24

scComSAMLightClientCtrlLongValueTypeLineRampingSpeedEndRampActive = 25

These constants return and set the ramping flags of the current selected pen. See also ScSetPen() and ScGetPen().

scComSAMLightClientCtrlLongValueTypeGetOptoOut = 31

This value is used to get the output states. The pin to bit mapping of the outputs can be found in the IO chapter. An example can be found in the Get and set outputs and inputs chapter.

scComSAMLightClientCtrlLongValueTypeAngularSplittingParts = 33

Especially for *Rotary Splitting* you can define here how often the object should be marked distributed homogeneously on the surface.

scComSAMLightClientCtrlLongValueTypeLastAutoCompensateResult = 38

Returns the error code of the scanner auto-calibration functionality.

scComSAMLightClientCtrlLongValueTypeDongleUserNumber = 39

Gets the Customer ID (User ID).

scComSAMLightClientCtrlLongValueTypeDongleSystemNumber = 40

Gets the Dongle ID (System ID, Card ID).

scComSAMLightClientCtrlLongValueTypeSizePixelMap = 42

Returns the size of the Pixel Map, means the number of the available Grey value partitions.

scComSAMLightClientCtrlLongValueTypeSwitchToPane = 43

If the additional parameter is *0* then the *Mark Preview* window is shown. If the parameter is *1* the *Main Window* is shown.

scComSAMLightClientCtrlLongValueTypeGetTotalSlices = 44

Gets the total number if slices of a sliced job file in SAM 3D mode.

scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendSourceConstantAlpha = 50

This constant is used to specify the blend value of the bitmap that is blended in the SAMLight view with the constant scComSAMLightClientCtrlStringValueTypeBmpAlphaBlendPathBmp. The data parameter defines the opaqueness of the bitmap that is shown. The data parameter can be in the range from 0 to 255. 0 is fully invisible, 255 is fully opaque.

scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendCenterPointX = 51

This constant is used to specify the X position of the bitmap that is blended into the view with the corresponding String Value Type constant scComSAMLightClientCtrlStringValueTypeBmpAlphaBlendPathBmp.

scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendCenterPointY = 52

This constant is used to specify the Y position of the bitmap that is blended into the view with the corresponding String Value Type constant scComSAMLightClientCtrlStringValueTypeBmpAlphaBlendPathBmp.

scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendBmpDimX = 53

This constant is used to specify the X dimension of the bitmap that is blended into the view with the corresponding String Value Type constant scComSAMLightClientCtrlStringValueTypeBmpAlphaBlendPathBmp.

scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendBmpDimY = 54

This constant is used to specify the Y dimension of the bitmap that is blended into the view with the corresponding String Value Type constant scComSAMLightClientCtrlStringValueTypeBmpAlphaBlendPathBmp.

scComSAMLightClientCtrlLongValueTypeDrillEnable = 55

If this is used with ScGetLongValue() this will return 1 if Drill is active and 0 if not. If used together with ScSetLongValue() set the Value parameter to 1 to enable Drill and 0 to disable Drill mode.

scComSAMLightClientCtrlLongValueTypeDrillEnableCo2Power = 56

If enabled Frequency, Laser1 and Laser2 are disabled, the values of these parameters are taken from the main page of pen settings instead.

scComSAMLightClientCtrlLongValueTypeSelectRedpointerForMoveAbs = 57

With this flag the behaviour of the ScMoveAbs command is complete different. The ScMoveAbs commands will be executed as a list. The red pointer move list is

- started and cleared by setting *SelectRedpointerForMoveAbs* to '1'.
- filled by one or multiple ScMoveAbs commands.
- executed by setting *SelectRedpointerForMoveAbs* to '0'. After the execution the list is cleared.

The red pointer pen (Mark dialog → Advanced) is used for this moves.

scComSAMLightClientCtrlLongValueTypeEntityArrayCountX = 58

scComSAMLightClientCtrlLongValueTypeEntityArrayCountY = 59

Gets or sets the X/Y 'Count' value used by scComSAMLightClientCtrlStringDataIdArrayCopyHard.

scComSAMLightClientCtrlLongValueTypeGetHeadCount = 61

Get the total number of heads (cards).

scComSAMLightClientCtrlLongValueTypeSaveView2DBitmapMode = 63

The drawing of bitmaps of the SAMLight View2D generated by ScSetStringValue() with type scComSAMLightClientCtrlStringValueTypeSaveView2DAdjustableDPI can be adjusted:

0: lines and pixels are drawn normal
1: lines and pixels are drawn thicker

scComSAMLightClientCtrlLongValueTypeSelectHatchPair = 64

Get or selects a number for the Hatch Pair.0 corresponds to Pair A and 4 corresponds to Pair E. This command will not update the GUI. Instead this will only affect the other Client Control commands.

scComSAMLightClientCtrlLongValueTypeRedpointerMode = 65

Get or Set the redpointer mode. The possible parameters are:

1: individual outline
2: total outline
3: individual border
4: only redpointer entities
5: outermost border

---

scComSAMLightClientCtrlLongValueTypeSetLockSjfToDongleFlags = 66

This flag allows you to lock a SJF job file to specific dongles.

scComSAMLightClientCtrlLockToDongleFlagLockLoadToSystemId = 1

Locks the SJF job file to the Dongle ID (System ID, Card ID).

scComSAMLightClientCtrlLockToDongleFlagLockLoadToUserId = 2

Locks the SJF job file to the Customer ID (User ID).

scComSAMLightClientCtrlLockToDongleFlagLockExport = 4

Locks the export function of the SJF job file.

---

scComSAMLightClientCtrlLongValueTypeServerStatus = 67

With this constant the current state of the USC server connection can be read. The return value is a combination of the following flags:

SC_USC1_SERVER_STATUS_DEVICE_CONNECTED = 1

SC_USC1_SERVER_STATUS_DEVICE_OK = 2

SC_USC1_SERVER_STATUS_DEVICE_MISSING = 8

SC_USC1_SERVER_STATUS_DEVICE_USC_1 = 65536

SC_USC1_SERVER_STATUS_DEVICE_USC_2 = 131072

---

scComSAMLightClientCtrlLongValueTypeXYStatus = 68

Returns the status of the scanner (head 0) based on the status signal of the XY2-100 interface of the selected USC-2 card. If you use more than one USC-2 card you have to select the card with ScSetHead first. The status will be updated every 100 ms (approximately). For the interpretation of this signal please refer to your scanner manual.

scComSAMLightClientCtrlLongValueTypeXYStatus1 = 69

Returns the status of the scanner (head 1) based on the status signal of the XY2-100 interface of the selected USC-2 card. If you use more than one USC-2 card you have to select the card with ScSetHead first. The status will be updated every 100 ms (approximately). For the interpretation of this signal please refer to your scanner manual.

scComSAMLightClientCtrlLongValueTypeSimulationMode = 70

SAMLight is able to release the scanner controller card (simualtion mode) to use the card with an other program without closing SAMLight.

0: Simulation mode enabled
1: Simulation mode disabled

scComSAMLightClientCtrlLongValueType3D = 71

This command can be used to check if SAMLight is started in the 3D mode. It returns '1' for SAM3D and '0' for SAMLight 2D.

---

scComSAMLightClientCtrlLongValueTypeCorrectionMode = 81

Specifies the correction mode of scComSAMLightClientCtrlExecCommandCorrectSamLight:

scComSAMLightClientCtrlCorrectionModeOld = 0

Old algorithm: The grid points of the UCF file will be mapped through a linear interpolation of the 4 enclosing measure points. Finding the enclosing points could fail and lead to single grid point distortions. Postfix: _YYYYMMDDOld.ucf

scComSAMLightClientCtrlCorrectionModeRTS = 1

Rotation, Translation, Scale: Rotation, translation and scaling operations are used to minimizes

the sum of squared residuals of all calib points. Postfix: _YYYYMMDDRts.ucf

scComSAMLightClientCtrlCorrectionModeIDW = 2

Inverse Distance Weighting: The position of each grid point is influenced by the closest x calib points weighted by their inverse distance. The number of influence points can be chosen with scComSAMLightClientCtrlLongValueTypeCorrectionPoints. This algorithm tends to cause unwanted oscillations. Postfix: _YYYYMMDDIdw#.ucf

scComSAMLightClientCtrlCorrectionMode2dFit =4

2D Fit: Minimizes the sum of squared residuals (distance between the fit and the calibration points). The more points in sc_calib_points.txt, the better is the compensation of measuring errors.
It is possible to try different fit modes and to compare the results in *_fit_log.txt. The fit order can be selected by scComSAMLightClientCtrlLongValueType2dFitType. Postfix: _YYYYMMDDFit2D#.ucf

---

scComSAMLightClientCtrlLongValueTypeCorrectionPoints = 82

Specifies the number of influence points for each grid point for inverse distance weighting (IDW) mode. A value of '4' seems to lead to good results. IDW mode must be enabled with scComSAMLightClientCtrlCorrectionModeIDW.

---

scComSAMLightClientCtrlLongValueType3DSurfaceSetType = 86

Changes the 3DSurface mode:

0: Cylinder
1: STL projection
2: Tilted surface
3: Sphere

scComSAMLightClientCtrlLongValueTypeEnable3DSurface = 87

Allows to control the 3DSurface feature via CCI. The mode can be set with ScSetLongValue() with type scComSAMLightClientCtrlLongValueType3DSurfaceSetType.
The parameters can be set with ScSetDoubleValue() with types from scComSAMLightClientCtrlDoubleValueType3DSurfaceValue1 to 11. For the projection feature the STL file can be loaded by ScSetStringValue() with type scComSAMLightClientCtrlStringValueTypeLoadStl.
The possible parameters are:

0: Disable 3DSurface feature
1: Enable 3DSurface feature

---

scComSAMLightClientCtrlLongValueTypeEnableJobToolbar = 88

Allows you to enable or disable the Jobs Toolbar under I/O via CCI. The mode can be set with ScSetLongValue().

scComSAMLightClientCtrlLongValueTypeBitmapPixelHardwareMode = 89

scComSAMLightClientCtrlLongValueTypeBitmapPenPowerAsMaxPower = 90

scComSAMLightClientCtrlLongValueTypeEnableUsc2Motf = 91

Allows you to enable or disable the checkbox for marking on the fly settings in Settings/Optic/Advanced. This works not only for USC-2, but also for USC-1. This mod can be set with ScSetLongValue().

scComSAMLightClientCtrlLongValueType2dFitType = 92

Selects the 2D fit order to adjust a correction file (*.ucf) by <SCAPS>\system\sc_calib_points.txt. 2D fit algorithm must be enabled with scComSAMLightClientCtrlCorrectionMode2dFit. Postfix: _YYYYMMDDFit2D#.ucf. Available 2D fit orders are 1, 2, 3, 4, 5 and 6:

1 and 2: 2D Fit Basic: Recommended for measurements with big errors. Improved compensation of measurement errors. 1_Basic can only compensate linear (trapezoid) distortions and should not be used for the neutral correction file (cor_neutral.ucf).

3 and 4: 2D Fit Advanced: Recommended for most scenarios. 3_Advanced is the default mode.

5 and 6: 2D Fit Expert: Only recommended for precise measurements which means the measurement error is really small.

scComSAMLightClientCtrlLongValueTypeEnableBmpSplitting = 93

Allows you to enable or disable the checkbox for Bitmap Splitting (= Bitmap (Rotary) Marking) in the Extras Toolbar. This works only if a scanner bitmap is present in the entity list. This mod can be set with ScSetLongValue() and get with ScGetLongValue() .

scComSAMLightClientCtrlLongValueTypeAngularSplittingSplitAxis = 94

Selects the split Axis in Angular Splitting - 0: X Axis, 1: Y Axis.

scComSAMLightClientCtrlLongValueTypeEnableAllSerialNumbersInJob = 95

0: disables the check box "enable" under "All Serial Numbers in Job" on entity property page.

1: enables the check box "enable" under "All Serial Numbers in Job" on entity property page.

scComSAMLightClientCtrlLongValueTypeRingSplittingEnableZTiltCompens ation = 96

scComSAMLightClientCtrlLongValueTypeRingSplittingZMotionAxis = 97

scComSAMLightClientCtrlLongValueTypePauseBuild = 98

With the function - ScSetLongValue 1: To pause a mark. 0: to continue the mark.

With the function - ScGetlongValue 0: Mark is not paused. 1: Mark is pausing (The current layer is to be finished).
2: Mark is paused.

scComSAMLightClientCtrlLongValueTypeShowEntityList = 99

With the function - ScSetLongValue 0: to hide the entity list. 1: to show the entity list.

scComSAMLightClientCtrlLongValueTypeShowPropSheet = 100

With the function - ScSetLongValue 0: to hide the property sheet. 1: to show the property sheet.

scComSAMLightClientCtrlLongValueTypeDrillMarkLineAsDotsActive = 101

With the function - ScSetLongValue 0: disable mark line as dots. 1: enable mark line as dots.

scComSAMLightClientCtrlLongValueTypeProtectedPen = 102

With the function - ScSetLongValue 0: disable Protected Pen. 1: enable Protected Pen.

---

scComSAMLightClientCtrlLongValueTypeShowToolBars = 103

With this constant the current view of toolbars can be set or read. The return value is a combination of the following flags:

SC_SHOW_TOOLBAR_MAIN	**= 1**
SC_SHOW_TOOLBAR_CAMERA	**= 2**
SC_SHOW_TOOLBAR_VIEW_LEVEL	**= 4**
SC_SHOW_TOOLBAR_FUNC_OBJECTS	**= 8**
SC_SHOW_TOOLBAR_ALIGN	**= 16**
SC_SHOW_TOOLBAR_GEOM_OBJECTS	**= 32**
SC_SHOW_TOOLBAR_EXTRAS	**= 64**
SC_SHOW_TOOLBAR_STEPPER	**= 128**
SC_SHOW_TOOLBAR_SURFACE	**= 256**
SC_SHOW_TOOLBAR_ANALOG_IN	**= 512**

scComSAMLightClientCtrlLongValueType1dMotfReverse = 104

Enable MOTF first, with this function you can change the marking direction to plus/ minus with 1/0 for X or Y in 1D Mark on the Fly Dialog.

---

scComSAMLightClientCtrlLongValueTypeNewCciErrorReturn = 105

This flag activates extended error codes. Use with the function - ScSetLongValue 0: disable extended error codes, thus the return is 0 (error) or 1 (ok). 1: enable extended error codes.

scComSAMLightClientCtrlLongValueTypeAfterDefocusDelay = 106

Defines a delay after a pen defocus for RTC5 and RTC6 cards. The unit is us. The value will be divided by 10 us internally which why values below 10 us will result in 0 us.

scComSAMLightClientCtrlLongValueTypeStepAndRepeatTotalSteps = 114

scComSAMLightClientCtrlLongValueTypeStepAndRepeatPlanarMode = 115

scComSAMLightClientCtrlLongValueTypeStepAndRepeatPlanarModeSimulation = 116

scComSAMLightClientCtrlLongValueTypeStepAndRepeatFirstMoveAxis = 117

sets planar axis in 1(X), please refer to motion settings dialog for the axis index (zero based).

None is not allowed to be set.

scComSAMLightClientCtrlLongValueTypeStepAndRepeatSecondMoveAxis = 118

sets planar axis in 2(Y), please refer to motion settings dialog for the axis index (zero based).

Index of "none" is -1.

scComSAMLightClientCtrlLongValueTypeStepAndRepeatPlanarStepCountX = 119

scComSAMLightClientCtrlLongValueTypeStepAndRepeatPlanarStepCountY = 120

scComSAMLightClientCtrlLongValueTypeEnableStepAndRepeat = 121

enables extras -> Step/Repeat -> Enable mode.

19.4.2 Double Value Types

These types can be used for the Type parameter of the functions ScSetDoubleValue() and ScGetDoubleValue().

---

scComSAMLightClientCtrlDoubleValueTypeOverrideSpeed = 1

This command sets the global override speed (on the mark property page in SAMLight). The global override speed is given in percent and acts on all pens.

scComSAMLightClientCtrlDoubleValueTypeOverridePower = 2

scComSAMLightClientCtrlDoubleValueTypeOverrideFrequ = 3

scComSAMLightClientCtrlDoubleValueTypeOverridePower2 = 22

These four constants can be used to define override values for marking speed, power and frequency.

scComSAMLightClientCtrlDoubleValueTypeMarkSpeed = 4

This command sets the pen-specific mark speed (on the mark property page in SAMLight). The mark speed is given as absolute value in mm/s and acts on the pen specified with ScSetPen or on the pen which was set last.

scComSAMLightClientCtrlDoubleValueTypeJumpSpeed = 5

scComSAMLightClientCtrlDoubleValueTypeFrequency = 6

Set or get Frequency in Hz.

scComSAMLightClientCtrlDoubleValueTypeJumpDelay = 7

scComSAMLightClientCtrlDoubleValueTypeMarkDelay = 8

scComSAMLightClientCtrlDoubleValueTypePolyDelay = 9

scComSAMLightClientCtrlDoubleValueTypeLaserOnDelay = 10

scComSAMLightClientCtrlDoubleValueTypeLaserOffDelay = 11

scComSAMLightClientCtrlDoubleValueTypeScannerXPos = 12

scComSAMLightClientCtrlDoubleValueTypeScannerYPos = 13

scComSAMLightClientCtrlDoubleValueTypeScannerZPos = 14

Before requesting the scanner positions with these three constants they must be updated with

ScExecCommand(9)
(9 = scComSAMLightClientCtrlExecCommandUpdateScannerPos)

scComSAMLightClientCtrlDoubleValueTypePulseLength = 15

scComSAMLightClientCtrlDoubleValueTypeFirstPulseLength = 16

scComSAMLightClientCtrlDoubleValueTypeLaserPower = 17

scComSAMLightClientCtrlDoubleValueTypeSizePowerMap = 18

scComSAMLightClientCtrlDoubleValueTypePowerMapStartId = 19

ScGetDoubleValue(19) returns the double value X which can be used with ScSetDoubleValue((int)X) and ScGetDoubleValue((int)X) to get or set the first bit value of the PowerMap. All PowerMap bit values are accessible with the integer values from *X* to *X+15*. In case of CO2 laser [%] is used instead of [bit].

scComSAMLightClientCtrlDoubleValueTypeMaxPower = 20

scComSAMLightClientCtrlDoubleValueTypeLineRamping PowerStartRampValue = 26

scComSAMLightClientCtrlDoubleValueTypeLineRamping PowerStartRampLength = 27

scComSAMLightClientCtrlDoubleValueTypeLineRamping PowerEndRampValue = 28

scComSAMLightClientCtrlDoubleValueTypeLineRamping PowerEndRampLength = 29

scComSAMLightClientCtrlDoubleValueTypeLineRamping SpeedStartRampValue = 30

scComSAMLightClientCtrlDoubleValueTypeLineRamping SpeedStartRampLength = 31

scComSAMLightClientCtrlDoubleValueTypeLineRamping SpeedEndRampValue = 32

scComSAMLightClientCtrlDoubleValueTypeLineRamping SpeedEndRampLength = 33

scComSAMLightClientCtrlDoubleValueTypeLineRampingLengthenStart = 60

scComSAMLightClientCtrlDoubleValueTypeLineRampingLengthenEnd = 61

scComSAMLightClientCtrlDoubleValueTypeSkyWritingStartLength = 35

scComSAMLightClientCtrlDoubleValueTypeSkyWritingEndLength = 36

scComSAMLightClientCtrlDoubleValueTypeSkyWritingBreakAngle = 37

The unit for the BreakAngle is Radians.

scComSAMLightClientCtrlDoubleValueTypeDefocus = 62

scComSAMLightClientCtrlDoubleValueTypeHalfPeriod = 43

scComSAMLightClientCtrlDoubleValueTypeSpiLaserSimmer = 45

Set or get Simmer value for Spi Laser.

scComSAMLightClientCtrlDoubleValueTypeLaserCo2Power1 = 48

scComSAMLightClientCtrlDoubleValueTypeLaserCo2Power2 = 49

Angle in rad. These constants return and set the corresponding values of the currently selected pen. See also ScSetPen() and ScGetPen().

scComSAMLightClientCtrlDoubleValueTypeLastExpectedMarkTime = 34

Returns the expected marking time like SAMLight → Mark → TimeInfo. A previous mark preview is not required.

scComSAMLightClientCtrlDoubleValueTypeLastMarkTime = 21

This value is not pen specific. It can be used to evaluate the time that was needed for the last marking operation. The unit is in [s].

scComSAMLightClientCtrlDoubleValueTypeHomePosX = 23

scComSAMLightClientCtrlDoubleValueTypeHomePosY = 24

scComSAMLightClientCtrlDoubleValueTypeHomePosZ = 25

Gets or sets the home position.

scComSAMLightClientCtrlDoubleValueTypeMotionAxisPosition = 38

The position is always given in mm.

scComSAMLightClientCtrlDoubleValueTypeMotionAxisAngle = 39

scComSAMLightClientCtrlDoubleValueTypeMotionAxisPositionRelative = 40

scComSAMLightClientCtrlDoubleValueTypeMotionAxisAngleRelative = 41

scComSAMLightClientCtrlDoubleValueTypeMotionAxisSpeed = 42

These constants are motion drive related values.

scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapDPI = 44

Sets the resolution for a bitmap to be taken from view.

scComSAMLightClientCtrlDoubleValueTypeGainX = 46

scComSAMLightClientCtrlDoubleValueTypeGainY = 47

Using these constants the *gain X* and *gain Y* correction factor of the optic can be set or get.

scComSAMLightClientCtrlDoubleValueTypeAngularSplittingTotalDiameter = 50

scComSAMLightClientCtrlDoubleValueTypeAngularSplittingAngle = 51

Using these constants the diameter and angle for the *Splitting Mode* can be set or get.

scComSAMLightClientCtrlDoubleValueTypeHorizontalSplittingValue = 52

scComSAMLightClientCtrlDoubleValueTypeVerticalSplittingValue = 53

Using these constants the horizontal and vertical split size for the *Splitting Mode* can be set or get.

scComSAMLightClientCtrlDoubleValueTypeWobbleFrequency = 54

scComSAMLightClientCtrlDoubleValueTypeWobbleAmplitude = 55

Using these constants the frequency and amplitude for the *Wobble Mode* can be set or get.

scComSAMLightClientCtrlDoubleValueTypeStartSplittingPosX = 56

scComSAMLightClientCtrlDoubleValueTypeStartSplittingPosY = 57

Using these constants a start value for the first (PosX, not necessarily the X axis) and second (PosY, not necessarily the Y axis) axis in *Splitting Mode* can be set or get.
For example, when using Angular Splitting, the start value can be set with StartSplittingPosX. When using 2D planar splitting, the start value of the first axis is set with StartSplittingPosX, the start value of the second axis is set with StartSplittingPosY.

scComSAMLightClientCtrlDoubleValueTypeOffsetX = 58

scComSAMLightClientCtrlDoubleValueTypeOffsetY = 59

Using these constants the *offset X* and *offset Y* correction factor of the optic can be set or get.

scComSAMLightClientCtrlDoubleValueTypeMOFExtStartDelay = 65

Using these constants the MOTF trigger delay for RTC cards can be set or get.

scComSAMLightClientCtrlDoubleValueTypeDoublePara1 = 66

This value has multiple purposes:

- Data Wizard → Beam Compensation → Dist.
- Data Wizard → Marking Order → arrow direction
  0: left to right
  1: top to bottom
  2: right to left
  3: bottom to top

scComSAMLightClientCtrlDoubleValueTypeDoublePara2 = 67

Activate or deactivate the Checkbox „Sort Equal Coordinates By Size“ for the Data Wizard. 1 = activate, 0 = deactivate.

scComSAMLightClientCtrlDoubleValueTypeWorkingAreaMinX = 68

scComSAMLightClientCtrlDoubleValueTypeWorkingAreaMinY = 69

scComSAMLightClientCtrlDoubleValueTypeWorkingAreaMaxX = 70

scComSAMLightClientCtrlDoubleValueTypeWorkingAreaMaxY = 71

By using these constants the working area can be set or get.

scComSAMLightClientCtrlDoubleValueTypeDrillPeriod = 72

With the DrillPeriod you can change the frequency of the laser pulses (f = 1/T). T in [µs]. For CO2 Lasers this flag is associated with the value of Laser1 in the Drill Dialog.

scComSAMLightClientCtrlDoubleValueTypeDrillDuration = 73

Time for scanning one point. [µs]

scComSAMLightClientCtrlDoubleValueTypeDrillLength = 74

The drill length is the pulse width or the Q-switch length. [µs]. For CO2 Lasers this flag is associated with the value of Laser2 in the Drill Dialog.

scComSAMLightClientCtrlDoubleValueTypeDrillJumpSpeed = 75

Speed to jump to a point. [mm/s]

scComSAMLightClientCtrlDoubleValueTypeDrillJumpDelay = 76

Delay between the jump to the point and start marking this point. [µs]

scComSAMLightClientCtrlDoubleValueTypeDrillCo2HalfPeriod = 79

If EnableCO2Power is used you need to change the DrillCo2HalfPeriod to adjust the frequency of the laser pulses (f = 1 / (2*HalfPeriod)). HalfPeriod in [µs]

scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapVariableSize = 81

Sets the width of Bitmap that is created with *scComSAMLightClientCtrlStringValueTypeSaveView2DVariableSize.*

scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapX = 82

scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapY = 83

scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapXW = 84

scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapYW = 85

Determines the behaviour of s*cComSAMLightClientCtrlStringValueTypeSaveView2DAdjustableDPI*. X and Y are the middle point. XW and YW are width and height.

scComSAMLightClientCtrlDoubleValueTypeSelPointXPos = 86

scComSAMLightClientCtrlDoubleValueTypeSelPointYPos = 87

scComSAMLightClientCtrlDoubleValueTypeSelPointZPos = 88

Returns the x, y and z coordinate of a point which is selected in SAMLight. If more than one point is selected the command returns the coordinate of the first selected point. If no point is selected the return value is '0'.

scComSAMLightClientCtrlDoubleValueTypeLongDelay = 89

Gets or sets the long delay of the current pen which can be set with *ScSetPen()*. Unit is [µs].

scComSAMLightClientCtrlDoubleValueTypeOffsetZ = 90

Gets or sets the Z offset of RTC cards. This value is related to the value in <SCAPS>\tools*sc_setup.exe → HardwareSettings → Settings → Z-Axis, Offset*.
Requires Optic3D license.
USC cards: XY coordinates are not compensated, like Pen Defocus.
RTC cards: XY coordinates are compensated, unlike Pen Defocus.

scComSAMLightClientCtrlDoubleValueTypeSpeedMotfEntityBasedSplitting = 91

Gets or sets the 'use speed' value in the MOTF entity based splitting dialog.

scComSAMLightClientCtrlDoubleValueTypeEntityArrayStepX = 92

scComSAMLightClientCtrlDoubleValueTypeEntityArrayStepY = 93

Gets or sets the X/Y 'inc.' value used by scComSAMLightClientCtrlStringDataIdArrayCopyHard.

scComSAMLightClientCtrlDoubleValueTypeEntityBasedSplittingGroupedEntityWidth = 94

Gets or sets the 'grouped entity width' value in the MOTF entity based splitting dialog.

scComSAMLightClientCtrlDoubleValueTypeDoublePara3 = 95

Activate or deactivate the Checkbox „Sort Equal by other Coordinate" for the Data Wizard.

0: disable
1: enable

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue1 = 96

Cylinder	X position
Sphere	X position

Tilted surface	X position
STL projection	Center in field ('0.0' for disabled, '1.0' for enabled)

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue2 = 97

Cylinder	Y position
Sphere	Y position
Tilted surface	Y position
STL projection	X offset

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue3 = 98

Cylinder	Z offset
Sphere	Z offset
Tilted surface	Z offset
STL projection	Y offset

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue4 = 99

Cylinder	XY angle
Sphere	Line splitting length
Tilted surface	XY angle
STL projection	Z offset

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue5 = 100

Cylinder	Diameter / 2
Sphere	Sphere angle
Tilted surface	Z tilt angle
STL projection	XY rotation

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue6 = 101

Cylinder	Line splitting length
Sphere	Diameter
Tilted surface	Width ('-1.0' for an infinite surface)
STL projection	Line splitting length

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue7 = 102

Cylinder	Tube angle
Sphere	-
Tilted surface	Height ('-1.0' for an infinite surface)
STL projection	X scale

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue8 = 103

Cylinder	Z tilt angle
STL projection	Y scale

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue9 = 104

STL projection	Z scale

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue10 = 105

STL projection	XZ rotaion

scComSAMLightClientCtrlDoubleValueType3DSurfaceValue11 = 106

STL projection	YZ rotation

scComSAMLightClientCtrlDoubleValueTypeOpticRotation = 111

This constant is used to set or get the rotation value[deg] in Settings → System →Optic. It can be used

for USC cards and RTC cards.

scComSAMLightClientCtrlDoubleValueTypeLastPreviewTime = 112

Gets process time of last preview. There is a CCI example for precalculating the mark time .

scComSAMLightClientCtrlDoubleValueType2DSplitWidthValue = 113

scComSAMLightClientCtrlDoubleValueType2DSplitHeightValue = 114

scComSAMLightClientCtrlDoubleValueTypeRingSplittingZTiltAngle = 115

scComSAMLightClientCtrlDoubleValueTypeRingSplittingZSpindleCenterHeight = 116

scComSAMLightClientCtrlDoubleValueTypeRingSplittingSpindleRradius = 117

scComSAMLightClientCtrlDoubleValueTypeRingSplittingZRingCenterHeight = 118

scComSAMLightClientCtrlDoubleValueTypeFieldMinX = 119

scComSAMLightClientCtrlDoubleValueTypeFieldMinY = 120

scComSAMLightClientCtrlDoubleValueTypeFieldMaxX = 121

scComSAMLightClientCtrlDoubleValueTypeFieldMaxY = 122

scComSAMLightClientCtrlDoubleValueTypeIPGPulseLength = 123

scComSAMLightClientCtrlDoubleValueTypeMinFrequency = 124

scComSAMLightClientCtrlDoubleValueTypeMaxFrequency = 125

scComSAMLightClientCtrlDoubleValueTypeDrillMarkLineAsDotsGridX = 126

scComSAMLightClientCtrlDoubleValueTypeDrillMarkLineAsDotsGridY = 127

scComSAMLightClientCtrlDoubleValueTypeStepAndRepeatPlanarStartX = 132

scComSAMLightClientCtrlDoubleValueTypeStepAndRepeatPlanarStartY = 133

scComSAMLightClientCtrlDoubleValueTypeStepAndRepeatPlanarStepX = 134

scComSAMLightClientCtrlDoubleValueTypeStepAndRepeatPlanarStepY = 135

scComSAMLightClientCtrlDoubleValueTypeStepAndRepeatSpeed = 136

scComSAMLightClientCtrlDoubleValueTypeStyleIdPixelMapZone = 4096

This constant is used to set or get the values for the *Pixel Map*. The first *Pixel Map* entry has the index *4096*, the second *4097* and so on ... . At the moment the *Pixel Map* consists of 6 entries.

scComSAMLightClientCtrlDoubleValueTypeUserValue = 20000

This value can be used to store or receive a double value into an unused and hidden entity in the job. This entity will be generated automatically unless it is already existing. The values from *20001* to *20015* can be used with the same functionality.

scComSAMLightClientCtrlDoubleValueTypeUsc1MotfCh0Multiplier = 65539

This constant is used to set or get the multiplier value for X-Channel.

scComSAMLightClientCtrlDoubleValueTypeUsc1MotfCh1Multiplier = 65540

This constant is used to set or get the multiplier value for Y-Channel.

scComSAMLightClientCtrlDoubleValueTypePrimaryHeadOffsetX = 65546

scComSAMLightClientCtrlDoubleValueTypePrimaryHeadOffsetY = 65547

Using these constants the *offset X* and *offset Y* correction factor of the primary head can be set or get.

scComSAMLightClientCtrlDoubleValueTypePrimaryHeadGainX = 65548

scComSAMLightClientCtrlDoubleValueTypePrimaryHeadGainY = 65549

Using these constants the *gain X* and *gain Y* correction factor of the primary head can be set or get.

scComSAMLightClientCtrlDoubleValueTypePrimaryHeadRotate = 65550

Using this constant the *rotational angle* of the primary head can be set or get.

scComSAMLightClientCtrlDoubleValueTypePrimaryHeadEnable = 65551

Using this constant the primary head can be activated or deactivated.

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadOffsetX = 65552

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadOffsetY = 65553

Using these constants the *offset X* (*Y)* correction factor of the secondary head can be set or get.

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadGainX = 65556

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadGainY = 65557

Using these constants the *gain X* (*Y)* correction factor of the secondary head can be set or get.

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadRotate = 65558

Using this constant the rotational angle of the secondary head can be set or get.

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadEnable = 65559

Using this constant the secondary head can be activated or deactivated.

scComSAMLightClientCtrlDoubleValueTypeUsc1MotfSimulate = 65638

scComSAMLightClientCtrlDoubleValueTypeUsc1MotfUseYChannel = 65640

Using this constant to switch between X-Channel(0) and Y-Channel(1).

scComSAMLightClientCtrlDoubleValueTypeUsc1MotfInvertOffset = 65661

scComSAMLightClientCtrlDoubleValueTypeUsc2MotfCh0Enable = 65662

scComSAMLightClientCtrlDoubleValueTypeUsc2MotfCh1Enable = 65663

scComSAMLightClientCtrlDoubleValueTypeUsc2MotfCh0Simulate = 65664

scComSAMLightClientCtrlDoubleValueTypeUsc2MotfCh1Simulate = 65665

scComSAMLightClientCtrlDoubleValueTypeUsc2MotfCh0Multiplier = 65670

scComSAMLightClientCtrlDoubleValueTypeUsc2MotfCh1Multiplier = 65671

Get or Set the MOTF multiplier for USC-2 or RTC-5 cards.

scComSAMLightClientCtrlDoubleValueTypeUsc2MotfCh0Counter = 65718

scComSAMLightClientCtrlDoubleValueTypeUsc2MotfCh1Counter = 65719

19.4.3 String Value Types

The following types can be used for the *Type* parameter of the functions ScSetStringValue() and ScGetStringValue().

---

scComSAMLightClientCtrlStringValueTypeRs232BaudRate = 1

Sets the Baud rate of the RS232 output. This String Value Type is only valid for USC cards. The RS232 interface supports the following Baud rates: *2400, 4800, 9600, 19200, 28800, 38400* and *57600*.

scComSAMLightClientCtrlStringValueTypeRs232OutputString = 2

Sets the RS232 Output string. This Sting Value Type is only valid for USC cards.

scComSAMLightClientCtrlStringValueTypeRs232Mode = 3

Sets the RS232 Mode. This String Value Type is only valid for USC cards. The following modes are available:

Mode	DataBits	StopBits	Parity
0	8	1	not used
1	8	1	odd
2	8	1	even
3	8	1	1
4	8	1	0

scComSAMLightClientCtrlStringValueTypeJobFileName = 4

Returns the name of the current job file with its complete path.

scComSAMLightClientCtrlStringValueTypeSaveView2D160 = 6

scComSAMLightClientCtrlStringValueTypeSaveView2D320 = 7

scComSAMLightClientCtrlStringValueTypeSaveView2DVariableSize = 8

scComSAMLightClientCtrlStringValueTypeSaveView2DFull = 9

These types can be used to create a screenshot from the main view and all contained entities. The file name where to save the captured bitmap file is given as a string parameter. The several command types differ only in the maximum picture size (width or height) of the bitmap file: 160 pixels, 320 pixels, custom defined size or full size. This function only works if the main window of the controlled scanner application is visible and not hidden by something else. So the window cannot be minimized and no screensaver should be active.

scComSAMLightClientCtrlStringValueTypeControlCmdCW300 = 50

The constant above has to be used to access a separate laser controller using an additional command string. For a detailed description on how to use these commands and how to access the specific controller, please refer to the specification that should be delivered together with the control (*sc_ht_use_the_CW300_rs232.pdf*). This constant is only valid for the functions ScSetStringValue(), ScSetStringLongValue(), ScSetStringDblValue() and ScGetStringDblValue().

scComSAMLightClientCtrlStringValueTypeMotionString = 11

Defines a string for send as RS232 string to the port defined in the motion settings.

scComSAMLightClientCtrlStringValueTypeSaveView2DAdjustableDPI = 12

Creates a bitmap holding the working area and saves the bitmap file into given path. The resolution is set with ScSetDoubleValue() with value type scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapDPI. The line and pixel thickness can be adjusted by ScSetLongValue() with value type scComSAMLightClientCtrlLongValueTypeSaveView2DBitmapMode.

scComSAMLightClientCtrlStringValueTypeCorrectionFile = 13

Using this constant a new correction file can be set or the current path of the correction file can be get.

scComSAMLightClientCtrlStringValueTypeSaveSplitsJobFileName = 21

Using this constant can be used to set the path and file name for to save a spitted job file as an entity list of split tiles. This constant has to be executed before ScExecCommand scComSAMLightClientCtrlExecCommandSaveSplitsAsEntities, see example "Save split job file as tiles".

scComSAMLightClientCtrlStringValueTypeCurrentPenName = 22

Using this constant the name of a pen can be set or get together with ScSetPen().

scComSAMLightClientCtrlStringValueTypeCorrectionFileHead2 = 23

Using this constant a new correction file name for the secondary scanhead can be set or get.

scComSAMLightClientCtrlStringValueTypeGetLastErrorMessageInput = 14

scComSAMLightClientCtrlStringValueTypeGetLastInfoMessageInput = 15

Using these constants it is possible to get the last error or info message that can be set via *Settings → System → IO → Message Inputs* for the input bits of the used scanner card.

scComSAMLightClientCtrlStringValueTypeSetToTopLevelEntity = 17

Sets the entity inside the job with the name that is specified by the other parameter to the top level entity.

scComSAMLightClientCtrlStringValueTypeStringPara1 = 19

This parameter is used to define the source entity, which will be used for the beam comped copy function.

scComSAMLightClientCtrlStringValueTypeStringPara2 = 20

This parameter is used to define the copied entity, which will be used for the beam comped copy function.

scComSAMLightClientCtrlStringValueTypeBmpAlphaBlendPathBmp = 27

This constant loads an .bmp image to the background of the SAMLight View. The filename is specified by the string parameter. No entity will be created in the SAMLight editor. Related constants are scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendBmpDimX and ...DimY which you can find under Long Value Types and are used to specify the dimension of the bitmap as well as scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendCenterPointX and ...PointY which are also located at Long Value Types and are used to specify the position of the bitmap.

scComSAMLightClientCtrlStringValueTypePenPixelMap = 28

Loads a bitmap to the Point Power Map of the currently used pen, ScSetPen(). The Point Power Map can be enabled with scComSAMLightClientCtrlLongValueTypeDeviceEnableFlagsValue → scComStandardDeviceStyleFlagEnablePointUsePowerMap

scComSAMLightClientCtrlStringValueTypeCorrectionFileLcf = 29

Using this constant a new correction file can be set or the current path of the correction file can be get. If a *.lcf file exists with the same name and in the same folder as the *.ucf file the lens parameters will be applied as well. If more than one scanhead is to be operated, the head number has to be specified with ScSetHead() first. It is not possible to call this constant during marking.

scComSAMLightClientCtrlStringValueTypeCorrectionFileLcfLensInit = 30

Using this constant a new correction file can be set or the current path of the correction file can be get. If a *.lcf file exists with the same name and in the same folder as the *.ucf file the lens parameters will be applied as well. If a *.sjf file exists with the same name and in the same folder as the *.ucf file the Lens Init Job will be executed as well. If more than one scanhead is to be operated, the head number has to be specified with ScSetHead() first. It is not possible to call this constant during marking.

scComSAMLightClientCtrlStringValueTypeLoadStl = 32

Loads a STL file for the 3DSurface STL feature.

scComSAMLightClientCtrlStringValueTypeUserValue = 20000

This value can be used to store or receive a string value into an unused and hidden entity in the job. This entity will be generated automatically unless it is already existing. The values from *20001* to *20009* can be used with the same functionality.

19.4.4 Long Data Ids

The following is a list of valid DataIds for the functions ScSetEntityLongData() and ScGetEntityLongData(). These DataIds are split into two parts: the first one **(grey)** can be used for OR-concatenating the parameter that is handed over to ScSetEntityLongData() and influences the behavior of the set method when the operation is performed that is specified by the second part of values. These second part of values that populate the lower 16 bit of the parameter DataId are not organized as a flag set and therefore cannot be OR-concatenated. Here one of them has to be used exclusively.

---

scComSAMLightClientCtrlLongDataIdFlagDontUpdateView **= 65536**

If this bit is set, the view will not be refreshed. This can be helpful for increasing performance.

scComSAMLightClientCtrlLongDataIdFlagDontUpdateEntity **= 131072**

If this bit is set, the entity will not be regenerated and updated. This can be helpful for increasing performance.

scComSAMLightClientCtrlStringDataIdFlagEnqueueCtrlCmd **= 524288**

scComSAMLightClientCtrlStringDataIdFlagEnqueueLastCtrlCmd **= 1048576**

These flags can be used only when marking in triggered mode and if USC-1 internal buffering is enabled. They cause the application not to execute the related command immediately but to put them into a queue. This queue of commands then is executed without any feedback after one entity was marked due to an external trigger signal. Here all commands sent with *scComSAMLightClientCtrlStringDataIdFlagEnqueueCtrlCmd* are executed for the same trigger signal as long as the flag *scComSAMLightClientCtrlStringDataIdFlagEnqueueLastCtrlCmd* is used. That means every sequence of calls to ScSetEntityLongData() finished with *scComSAMLightClientCtrlStringDataIdFlagEnqueueLastCtrlCmd* using these flags causes modifications of the job per trigger signal. This flag is useful for very fast modifications of a job in triggered/buffered operation mode. For more information about the queue functionality please refer to the Examples section.

scComSAMLightClientCtrlLongDataIdFlagToplevelOnly **= 2097152**

If this bit is set, it will only be searched for entities in first level of the job. This can be helpful to increase the performance.

---

scComSAMLightClientCtrlLongDataIdUserData **= 1**

Then the parameter *Data* contains a long value with the data to store inside the entity.

---

scComSAMLightClientCtrlLongDataIdTextAlignment **= 2**

Then the parameter *Data* contains a long value with the alignment flag fields with the following possible values:

scComSAMLightClientCtrlTextAlignmentCenter **= 1**

scComSAMLightClientCtrlTextAlignmentLeft **= 2**

scComSAMLightClientCtrlTextAlignmentRight **= 4**

scComSAMLightClientCtrlTextAlignmentTop **= 8**

scComSAMLightClientCtrlTextAlignmentBottom **= 16**

scComSAMLightClientCtrlTextAlignmentMiddle **= 32**

scComSAMLightClientCtrlTextAlignmentRadialCenter **= 64**

scComSAMLightClientCtrlTextAlignmentRadialEnd **= 128**

scComSAMLightClientCtrlTextAlignmentLineLeft **= 256**

scComSAMLightClientCtrlTextAlignmentLineRight **= 512**

scComSAMLightClientCtrlTextAlignmentLineCenter **= 1024**

---

scComSAMLightClientCtrlLongDataIdEntitySelected **= 3**

This function can be used to change the selection state of entities or to get information whether the entity is selected. The parameter *Data* has to be *0* or *1*.

scComSAMLightClientCtrlLongDataIdEntityArrayCountX **= 4**

This function can be used to change the array x count value of the entity.

scComSAMLightClientCtrlLongDataIdEntityArrayCountY **= 5**

This function can be used to change the array y count value of the entity.

scComSAMLightClientCtrlLongDataIdEntityArrayStepX **= 6**

This function can be used to change the array x step value of the entity. The unit factor of the long value is *0.001*.

scComSAMLightClientCtrlLongDataIdEntityArrayStepY **= 7**

This function can be used to change the array y step value of the entity. The unit factor of the long value is *0.001*.

---

scComSAMLightClientCtrlLongDataIdEntityArrayOrderFlags **= 8**

This function can be used to change the output order of the array. Then the parameter *Data* contains a flag field with a combination out of the following values:

scComSAMLightClientCtrlEntityArrayOrderFlagMainDirX **= 1024**

Use X as the main direction for array copying.

scComSAMLightClientCtrlEntityArrayOrderFlagNegX **= 256**

Go from left to right in horizontal direction if this flag is set, else go from right to left.

scComSAMLightClientCtrlEntityArrayOrderFlagNegY **= 512**

Go from top to bottom in vertical direction, else go from bottom to top.

scComSAMLightClientCtrlEntityArrayOrderFlagBiDir **= 2048**

Use bidirectional mode (horizontal), that means array copy is done into one direction and back instead of using only one direction, jump back and start from the beginning using the same direction again.

---

scComSAMLightClientCtrlLongDataIdTextCharFlags **= 9**

Then the parameter *Data* contains a long value with the char flag fields with the following possible values (can be combined logically):

ScComSAMLightClientCtrlLongDataIdTextCharFlagMonoSpaced **= 3**

scComSAMLightClientCtrlLongDataIdTextCharFlagItalic = 65536

scComSAMLightClientCtrlLongDataIdTextCharFlagRadial = 131072

scComSAMLightClientCtrlLongDataIdTextCharFlagRadialAlignToCharOutline = 262144

scComSAMLightClientCtrlLongDataIdTextCharFlagReverseOrder = 524288

scComSAMLightClientCtrlLongDataIdTextCharFlagMirrorCharOnXAxis = 1048576

scComSAMLightClientCtrlLongDataIdTextCharFlagMirrorCharOnYAxis = 2097152

scComSAMLightClientCtrlLongDataIdTextCharFlagSwapLines = 4194304

scComSAMLightClientCtrlLongDataIdTextCharFlagSetToLimitLength = 8388608

scComSAMLightClientCtrlLongDataIdTextCharFlagSetToLimitHeight = 16777216

scComSAMLightClientCtrlLongDataIdTextCharFlagSetToLimitKeepAspect = 33554432

scComSAMLightClientCtrlLongDataIdTextCharFlagOrderXDown = 67108864

sets/gets the third Marking order symbol in Text Properties dialog-window.

scComSAMLightClientCtrlLongDataIdTextCharFlagOrderYUp = 134217728

sets/gets the second Marking order symbol in Text Properties dialog-window.

scComSAMLightClientCtrlLongDataIdTextCharFlagOrderYMainUp = 268435456

sets/gets the first Marking order symbol in Text Properties dialog-window.

scComSAMLightClientCtrlLongDataIdTextCharFlagOrderBiDir = 536870912

sets/gets the fourth Marking order symbol in Text Properties dialog-window.

scComSAMLightClientCtrlLongDataIdTextCharFlagRadialCenterMode = 1073741824

Sets the 'center' checkbox for text in radial mode with the corresponding ScSetEntityDoubleData():
scComSAMLightClientCtrlDoubleDataIdTextRadialCenterX
scComSAMLightClientCtrlDoubleDataIdTextRadialCenterY

---

scComSAMLightClientCtrlLongDataIdTextFontAvailable = 10

This constant will examine if for the given Entityname the specified font is available on the PC on which SAMLight is currently running. The return value will be 1 if the fonts is available 0 if it is not.

---

scComSAMLightClientCtrlLongDataIdBitmapMode = 49

This DataId is used to manipulate a named bitmap. Please note that the scanner bitmap is created after every call to ScSetEntityLongData() as long as the flag scComSAMLightClientCtrlDoubleDataIdFlagDontUpdateEntity is not set. On the other hand the scanner bitmap is not created if brightness, ditherstep or intensity are modified. Here in every case it is required to set the flags. The data that can be set with this *DataId* correspond to the functionality of the *Bitmap* property page:

scComSAMLightClientCtrlLongDataIdBitmapModeInvert = 1

scComSAMLightClientCtrlLongDataIdBitmapModeGreyscale = 2

scComSAMLightClientCtrlLongDataIdBitmapModeDrillmode = 4

scComSAMLightClientCtrlLongDataIdBitmapModeBidirectional = 8

scComSAMLightClientCtrlLongDataIdBitmapModeStartlastline = 16

scComSAMLightClientCtrlLongDataIdBitmapModeNolineincr = 32

No increment of line position.

scComSAMLightClientCtrlLongDataIdBitmapModeShowBitmap = 256

scComSAMLightClientCtrlLongDataIdBitmapModeShowScanner = 512

The last two data flags define what bitmap has to be displayed into the view. If none of them is set, nothing is shown and the imported bitmap disappears.

scComSAMLightClientCtrlLongDataIdBitmapModeScanXDir = 1024

scComSAMLightClientCtrlLongDataIdBitmapModePenFrequency = 2048

scComSAMLightClientCtrlLongDataIdBitmapModeJumpOverBlankPixels = 4096

scComSAMLightClientCtrlLongDataIdBitmapModeDrillGreyscale = 8192

---

scComSAMLightClientCtrlLongDataIdTextWeight = 50

By using this DataId the style of a text can be modified. For the weight following Data constants are defined:

scComCharWeightThin = 100

scComCharWeightExtraLight = 200

scComCharWeightLight = 300

scComCharWeightNormal = 400

scComCharWeightMedium = 500

scComCharWeightSemiBold = 600

scComCharWeightBold = 700

scComCharWeightExtraBold = 800

scComCharWeightHeavy = 900

For an italic style refer to data values of the scComSAMLightClientCtrlLongDataIdTextCharFlags DataId.

---

scComSAMLightClientCtrlLongDataIdEnableHatching1 = 51

scComSAMLightClientCtrlLongDataIdEnableHatching2 = 52

Using these DataIds the both hatchers can be enabled, disabled or the actual state can be retrieved. Disabling is done by setting the appropriate data field to *0*. A value not equal *0* defines the hatching

mode. Here the following values are possible:

1) Wavy line without marking the jumps
2) Horizontal left to right without marking the jumps
3) Horizontal right to left without marking the jumps
4) Rotational, applies only to rectangle, ellipse and triangle structures in the current version
5) Wavy line including the jumps
6) Zigzag

scComSAMLightClientCtrlLongDataIdEntityMarkLoopCount = 55

This function can be used to change the mark loop count value of the entity.

scComSAMLightClientCtrlLongDataIdEntityMarkBeatCount = 56

This function can be used to change the mark beat count value of the entity.

scComSAMLightClientCtrlLongDataIdEntityMarkStartCount = 57

This function can be used to change the mark beat offset value of the entity.

---

scComSAMLightClientCtrlLongDataIdEntityMarkFlags = 58

This function can be used to change the mark flags (mark contour and mark hatch) of the entity. The possible values of the flags are:

scComSAMLightClientCtrlLongDataIdEntityMarkFlagMarkContour = 1

scComSAMLightClientCtrlLongDataIdEntityMarkFlagMarkHatch = 2

---

scComSAMLightClientCtrlLongDataIdEntitySetPen = 60

The adjusted pen number of an entity can be changed.

scComSAMLightClientCtrlLongDataIdEntitySetTimerValue = 61

By using this constant the time value of a ScTimer object can be changed. The unit for the related long value is in [ms].

scComSAMLightClientCtrlLongDataIdEntitySetInOutValue = 62

Here a new bit can be set for a ScSetOutput entity to set an output or for a ScWaitForInput entity to wait for a signal on the related input.

The long value that is handed over together with this constant cannot be a combination of several bits. Here exactly one bit has to be defined. Values with more than one bit or with no bit set are not allowed and may lead to undefined results.

scComSAMLightClientCtrlLongDataIdEntitySetOutputPulse = 64

Using this parameter a time for a ScSetOutput entitiy output pulse can be defined. A long value that is greater than *0* enables the output pulse functionality and sets this value in [µs] units. If a value of *-1* is handed over the output pulse is disabled and the value specified using *scComSAMLightClientCtrlLongDataIdEntitySetInOutValue* and *scComSAMLightClientCtrlLongDataIdEntitySetInOutLevel* is set as long as it is not replaced by an other one.

scComSAMLightClientCtrlLongDataIdEntitySetInOutLevel = 65

The entities ScSetOutput and ScWaitForInput may act on different signal levels. Using this constant that level can be defined: Setting it to *0* means a *low* signal is set to the output pin or a *low* level will be expected at the input defined with *scComSAMLightClientCtrlLongDataIdEntitySetInOutValue*. If a *1* is set as long value a *high* signal is set or expected. Other values than *0* for *low* and *1* for *high* are not allowed here.

scComSAMLightClientCtrlLongDataIdEntityGetTimerValue = 66

Here the current delay time of the ScTimer entity is read.

scComSAMLightClientCtrlLongDataIdEntityGetInOutValue = 67

Using this constant the bit can be read that will be set by the related ScSetOutput entity or the related ScWaitForInput will wait for.

scComSAMLightClientCtrlLongDataIdEntityGetOutputPulse = 69

If there is an output pulse value defined for a ScSetOutput entity using this constant its long value can be read. If the output pulse feature is disabled for this entity, *-1* is returned.

scComSAMLightClientCtrlLongDataIdEntityGetInOutLevel = 70

The ScSetOutput and ScWaitForInput entities can act or react on a definable level *low* or *high*. Using this constant the current configuration of the entity can be evaluated. If a *1* is returned the entity uses a *high* signal, in case of a *0*, the *low* signal is used.

scComSAMLightClientCtrlLongDataIdEntitySerialStartValue = 71

Get or set the start value of a serial number. This value is used after a serial number is reset.

scComSAMLightClientCtrlLongDataIdEntitySerialIncrValue = 72

Get or set the increment value for a serial number object.

scComSAMLightClientCtrlLongDataIdEntitySerialCurrValue = 73

Get the current value of a serial number.

scComSAMLightClientCtrlLongDataIdEntityGetPen = 74

The pen number of the specified entity can be retrieved.

---

scComSAMLightClientCtrlLongDataIdEntityOpticFlags = 75

This function can be used for changing the *Mark Contour* and the *Mark Hatch* flags of an object. The parameter is a bit field defining which flags have to be set:

scComSAMLightClientCtrlLongDataIdEntityOpticFlagContour = 1

scComSAMLightClientCtrlLongDataIdEntityOpticFlagHatch = 2

---

scComSAMLightClientCtrlLongDataIdEntitySetAsBackgroundEntity = 78

Set the entity specified by *EntityName* to the background or foreground. To set it to the background use *1* as data index. To set it into the foreground use *0* as data index. The data index is the third and last parameter of the corresponding *ScSetEntityLongData* command.

scComSAMLightClientCtrlLongDataIdEntitySerialBeatCount = 79

Get or set the beat count of the serial number.

scComSAMLightClientCtrlLongDataIdEntitySerialResetCount = 80

Get or set the reset count of the serial number.

---

scComSAMLightClientCtrlLongDataIdSetHatchFlags1 = 81

This works not only for 2D but also in SAM3D situation.

scComSAMLightClientCtrlLongDataIdClearHatchFlags1 = 91

scComSAMLightClientCtrlLongDataIdSetHatchFlags2 = 82

scComSAMLightClientCtrlLongDataIdClearHatchFlags2 = 92

Get or set the hatch flags for the given entity. Use *SetHatchFlags* to check the flag and *ClearHatchFlags* to uncheck the flag. In the following the possible parameters are described:

scComSAMLightClientCtrlLongDataIdHatchFlagAllLines = 1024

scComSAMLightClientCtrlLongDataIdHatchFlagNoSort = 256

scComSAMLightClientCtrlLongDataIdHatchFlagKeepAngle = 524288

scComSAMLightClientCtrlLongDataIdHatchFlagEqualizeDistance = 16777216

scComSAMLightClientCtrlLongDataIdHatchFlagDontFillRest = 16384

scComSAMLightClientCtrlLongDataIdHatchFlagPolyLineBeamComp = 8192
This flag activates the NumLoops feature of the hatch. It will create a beam compensation from outside to inside of the object. This works not only for 2D but also in SAM3D situation.

scComSAMLightClientCtrlLongDataIdHatchFlagBeamComp LoopReverseOrder = 33554432
This flag has to be used together with the hatch flag PolyLineBeamComp and will generate NumLoops from inside to outside of the object.

---

scComSAMLightClientCtrlLongDataIdBarcodeSetFlags = 101

scComSAMLightClientCtrlLongDataIdBarcodeClearFlags = 102

Different parameters are available:

scComSAMLightClientCtrlLongDataIdBarcodeFlagVariableLength = 1

scComSAMLightClientCtrlLongDataIdBarcodeFlagInvert = 2

scComSAMLightClientCtrlLongDataIdBarcodeFlagDisableAutoQuiet Zone = 4

scComSAMLightClientCtrlLongDataIdBarcodeFlagQuietZoneAbsolut e = 8

scComSAMLightClientCtrlLongDataIdBarcodeFlagGenerateCheckCo de = 16

Barcode → Extended → Auto Parity is only available for *Code-93*, *I-2/5* and *Ex Code93*

scComSAMLightClientCtrlLongDataIdBarcodeFlagInvertExceptText = 32

scComSAMLightClientCtrlLongDataIdBarcodeFlagInvertCellMode = 64

scComSAMLightClientCtrlLongDataIdBarcodeFlagCompactMode **= 128**

Barcode → *Extended* → Compact is only available for *PDF417*

---

scComSAMLightClientCtrlLongDataIdDataMatrixSymbolSize **= 105**

Parameter corresponds to the sizes in combo box, starting with *1*.

---

scComSAMLightClientCtrlLongDataIdDataMatrixSetSymbolMode **= 103**

scComSAMLightClientCtrlLongDataIdDataMatrixClearSymbolMode **= 104**

Different parameters are available:

scComSAMLightClientCtrlLongDataIdDataMatrixExSymbolMode Rectangle **= 1**

scComSAMLightClientCtrlLongDataIdDataMatrixExSymbolMode AutoSize **= 65536**

scComSAMLightClientCtrlLongDataIdDataMatrixExSymbolMode AutoEncodation **= 131072**

scComSAMLightClientCtrlLongDataIdDataMatrixExSymbolModeDots **= 262144**

scComSAMLightClientCtrlLongDataIdDataMatrixExSymbolModeTilde **= 524288**

scComSAMLightClientCtrlLongDataIdDataMatrixExSymbolModeCells **= 1048576**

scComSAMLightClientCtrlLongDataIdDataMatrixExSymbolMode NoFinderCells **= 2097152**

scComSAMLightClientCtrlLongDataIdDataMatrixExSymbolModeEllipse **= 4194304**

If *Generate Cells* is activated the cells are now small circles.

scComSAMLightClientCtrlLongDataIdDataMatrixExSymbolMode TextFreelyEditable **= 8388608**

---

scComSAMLightClientCtrlLongDataIdDataMatrixEncoding **= 106**

Set or get the Encoding of the *Data Matrix*

Different parameters are available:

scComSAMLightClientCtrlLongDataIdDataMatrixExEncodationAscii **= 1**

scComSAMLightClientCtrlLongDataIdDataMatrixExEncodationBase256 **= 2**

scComSAMLightClientCtrlLongDataIdDataMatrixExEncodationC40 **= 3**

scComSAMLightClientCtrlLongDataIdDataMatrixExEncodationText **= 4**

scComSAMLightClientCtrlLongDataIdDataMatrixExEncodationAnsiX12 **= 5**

scComSAMLightClientCtrlLongDataIdDataMatrixExEncodationEdifac **= 6**

t

---

scComSAMLightClientCtrlLongDataIdDataMatrixNumberOfDots = 124

Set or get the number of points per cell for DataMatrixEx. Valid values to set are 1, 4, 9, 16, otherwise returns error.

---

scComSAMLightClientCtrlLongDataIdDataBarcodeTextEnable = 107

For barcodes: If the additional parameter is *0* then text is disabled, if it is *1* text is enabled.

---

scComSAMLightClientCtrlLongDataIdEntitySetAsHiddenEntity = 112

Hides or shows the entity in View2D specified by the *EntityName*. To hide it in View2D use *1* as data index. To show it in View2D use *0* as data index. The data index is the third and last parameter of the corresponding *ScSetEntityLongData* command.

---

scComSAMLightClientCtrlLongDataIdSpiralNumInnerRotations = 113

scComSAMLightClientCtrlLongDataIdSpiralNumOuterRotations = 114

scComSAMLightClientCtrlLongDataIdSpiralNumOuterSegments = 115

Adjust Inner, Outer Rotations and Segments as described in Job Editor → Geometry Objects → Spiral

scComSAMLightClientCtrlLongDataIdSpiralFlags = 116

The Flag can be one of the following values or the logical OR combination of them:

1 = Clockwise

2 = Start from Outer

4 = Set Return Path

scComSAMLightClientCtrlLongDataIdEntityGroupPenPaths = 118

Has the same functionality as the checkbox Group → PenPaths in the Entity Info Property page.

scComSAMLightClientCtrlLongDataIdEntityGroupCluster = 119

Has the same functionality as the checkbox Group → Cluster in the Entity Info Property page.

scComSAMLightClientCtrlLongDataIdEntityMirrorOnPlane = 120

Has the same functionality as the buttons of page Z-Dimension → Mirror and Rotate.

The type can be one of the following values:

1 for xz-plane with the center of field

2 for yz-plane with the center of field

3 for xy-plane with the center of field

4 for xz-plane with the center of entity

5 for yz-plane with the center of entity

6 for xy-plane with the center of entity

scComSAMLightClientCtrlLongDataIdEntityGenerate = 125

is developed for "connected" serial number/ barcode and Text2D (share the same name in EntityInfo) -- when the text is changed, the barcode would be changed in the same way. This command has the same functionality as the button "apply" in the Barcode page after changing the text in SAMLight. Note: Flag = 1 is required for this command.

scComSAMLightClientCtrlLongDataIdEntityCenter = 129

The type can be one of the following values:

6 for center x

7 for center y

8 for center both

19.4.5 Double Data Ids

The following is a list of valid DataIds for the functions ScSetEntityDoubleData() and ScGetEntityDoubleData(). These DataIds are split into two parts: the first one **(grey)** can be used for OR-concatenating the parameter that is handed over to ScSetEntityDoubleData() and influences the behavior of the set method when the operation is performed that is specified by the second part of values. These second part of values that populate the lower 16 bit of the parameter DataId are not organized as a flag set and therefore cannot be OR-concatenated. Here one of them has to be used exclusively.

---

scComSAMLightClientCtrlDoubleDataIdFlagDontUpdateView **= 65536**

If this bit is set, the view will not be refreshed. This can be helpful to increase the performance.

scComSAMLightClientCtrlDoubleDataIdFlagDontUpdateEntity **= 131072**

If this bit is set, the entity will not be regenerated and updated. This can be helpful to increase the performance.

scComSAMLightClientCtrlDoubleDataIdFlagToplevelOnly **= 262144**

If this bit is set, it will be only searched for entities in the first level of the job. This can be helpful to increase the performance.

---

scComSAMLightClientCtrlDoubleDataIdTextSize **= 1**

The parameter data contains the size of the text.

scComSAMLightClientCtrlDoubleDataIdTextCharSpacing **= 2**

Change the character spacing of the string. 1 equals 100%.

scComSAMLightClientCtrlDoubleDataIdTextLengthLimit **= 3**

Change the length limit of the text object.

scComSAMLightClientCtrlDoubleDataIdTextHeightLimit **= 4**

Change the height limit of the text object.

scComSAMLightClientCtrlDoubleDataIdTextRadius **= 5**

Change the radius of a radial text.

scComSAMLightClientCtrlDoubleDataIdTextStartAngle **= 6**

Change the start angle of a radial text. Units are in [rad].

scComSAMLightClientCtrlDoubleDataIdTextRadialCenterX **= 9**

scComSAMLightClientCtrlDoubleDataIdTextRadialCenterY **= 10**

Sets the center of a radial text. The 'center' checkbox can be enabled with ScSetEntityLongData() with the IdTextCharFlag 'RadialCenterMode'.

scComSAMLightClientCtrlDoubleDataIdBitmapIntensity **= 33**

Changes the intensity of the related bitmap but does not recreate the appropriate scanner bitmap. Please refer to the bitmap mode flags for more information.

scComSAMLightClientCtrlDoubleDataIdBitmapBrightness **= 34**

Changes the brightness of the related bitmap but does not recreate the appropriate scanner bitmap. Please refer to the bitmap mode flags for more information.

scComSAMLightClientCtrlDoubleDataIdBitmapDitherstep = 35

Changes the rasterization size of the scanner bitmap but does not recreate it immediately. The recreation is done by setting the required bitmap mode flags.

scComSAMLightClientCtrlDoubleDataIdTextOrientation = 36

Changes the orientation of a text. With a parameter of *0* for the data field of ScSetEntityDoubleData() the text is oriented in horizontal direction. If the parameter is *pi/2* then the text is displayed vertically. The following constants define hatching related values.

Some of them depend on the hatching mode selected using the scComSAMLightClientCtrlLongDataIdEnableHatching constants.

scComSAMLightClientCtrlDoubleDataIdHatchDistance1 = 37

scComSAMLightClientCtrlDoubleDataIdHatchDistance2 = 45

Modifies or retrieves the hatch distance for the hatch number one or two.

scComSAMLightClientCtrlDoubleDataIdHatchAngle1 = 38

scComSAMLightClientCtrlDoubleDataIdHatchAngle2 = 46

Using this DataId the hatch angle for the first or second hatch can be get or set. The angle has to be specified in [rad].

These four constants for Hatch distance and Hatch angle can be used for drill geometry entity as well. If it is desired to get and set the values of entities in the Drill Geometry, "drill_geom_" should be added at the beginning of the entity name. For example, if the entity name is "Test", in the CCI call the entity name should be modified to "drill_geom_Test". Please note that the unit of angle in CCI project is [rad], whereas the unit in SAMLight is [deg].

scComSAMLightClientCtrlDoubleDataIdHatchMinjump1 = 39

scComSAMLightClientCtrlDoubleDataIdHatchMinjump2 = 47

Hatch minimal jump value for hatch one or two

scComSAMLightClientCtrlDoubleDataIdHatchStartoffset1 = 40

scComSAMLightClientCtrlDoubleDataIdHatchStartoffset2 = 48

The start offset value for the first or second hatch

scComSAMLightClientCtrlDoubleDataIdHatchLinereduct1 = 41

scComSAMLightClientCtrlDoubleDataIdHatchLinereduct2 = 49

Specifies the amount of line reduction for the first or second hatch

scComSAMLightClientCtrlDoubleDataIdHatchEndoffset1 = 42

scComSAMLightClientCtrlDoubleDataIdHatchEndoffset2 = 50

The end offset value for the first or second hatch

scComSAMLightClientCtrlDoubleDataIdHatchBeamcompensation1 = 43

scComSAMLightClientCtrlDoubleDataIdHatchBeamcompensation2 = 51

Beam compensation value for the first or second hatch

scComSAMLightClientCtrlDoubleDataIdHatchNumloops1 = 44

This works not only for 2D but also in SAM3D situation.

scComSAMLightClientCtrlDoubleDataIdHatchNumloops2 = 52

Number of loops for the first or second hatch

scComSAMLightClientCtrlDoubleDataIdBarcodeLinereduction = 69

The line reduction of the barcode in percent for reducing the size of a barcode line.

scComSAMLightClientCtrlDoubleDataIdMotfOffset = 70

The distance of a ScMotfOffset control object.

scComSAMLightClientCtrlDoubleDataIdEntityRotationAngle = 71

This constant will return the current rotation angle of the entity in [deg]. If the entity name is empty the return value is *0*.

scComSAMLightClientCtrlDoubleDataIdDataMatrixCellSizeX = 72

scComSAMLightClientCtrlDoubleDataIdDataMatrixCellSizeY = 73

Parameter is a value between *0* and *1*.

---

scComSAMLightClientCtrlDoubleDataIdSpiral2DInnerRadius = 74

scComSAMLightClientCtrlDoubleDataIdSpiral2DOuterRadius = 75

scComSAMLightClientCtrlDoubleDataIdSpiral2DRise = 76

Adjust Inner and Outer Radius as well a the Rise Parameter of the Spiral.

scComSAMLightClientCtrlDoubleDataIdEllipse2DRadiusX = 77

scComSAMLightClientCtrlDoubleDataIdEllipse2DRadiusY = 78

scComSAMLightClientCtrlDoubleDataIdEllipse2DCenterX = 79

scComSAMLightClientCtrlDoubleDataIdEllipse2DCenterY = 80

These constants change the radius or center position of the ellipse.

---

scComSAMLightClientCtrlDoubleDataIdDataMatrixQuietZoneX = 81

Set or get the X Quiet Zone of a DataMatrixEx. If the checkbox "Absolute" is activated the Quiet Zone is given in mm. If you need to set different paramters for X and Y you have to clear the flag scComBarFlagDisableAutoQuietZone with scComSAMLightClientCtrlLongDataIdBarcodeClearFlags. If not, the Quiet Zone is given as a scale factor (1 = 100 %).

scComSAMLightClientCtrlDoubleDataIdDataMatrixQuietZoneY = 82

Set or get the Y Quiet Zone of a DataMatrixEx.If the checkbox "Absolute" is activated the Quiet Zone is given in mm If you need to set different paramters for X and Y you have to clear the flag scComBarFlagDisableAutoQuietZone with scComSAMLightClientCtrlLongDataIdBarcodeClearFlags. If not, the Quiet Zone is given as a scale factor (1 = 100 %).

scComSAMLightClientCtrlDoubleDataIdHatchPointOffset1 = 83

scComSAMLightClientCtrlDoubleDataIdHatchMinLength1 = 84

scComSAMLightClientCtrlDoubleDataIdHatchPointOffset2 = 85

scComSAMLightClientCtrlDoubleDataIdHatchMinLength2 = 86

ScComSAMLightClientCtrlDoubleDataIdDataMatrixDistanceBetweenDots = 87

Get the distance between two points of DataMatrixEx.

ScComSAMLightClientCtrlDoubleDataIdSecondaryHeadTransformationOffsetX = 88

ScComSAMLightClientCtrlDoubleDataIdSecondaryHeadTransformationOffsetY = 89

ScComSAMLightClientCtrlDoubleDataIdSecondaryHeadTransformationScaleX = 90

ScComSAMLightClientCtrlDoubleDataIdSecondaryHeadTransformationScaleY = 91

ScComSAMLightClientCtrlDoubleDataIdSecondaryHeadTransformationRotation = 92

These SecondaryHeadTransformation commands are only available for USC-2/3 with license for Head2 and activated Head2. The values are corresponding to the parameter of a ScSetSecondaryHeadTransformation entity.

19.4.6 String Data Ids

The following is a list of valid DataIds for the function ScSetEntityStringData(), ScGetEntityStringData() and other string-related functions. These DataIds are split into two parts: the first one **(grey)** can be used for OR-concatenating the parameter that is handed over to ScSetEntityStringData() and influences the behavior of the set method when the operation is performed that is specified by the second part of values. These second part of values that populate the lower 16 bit of the parameter DataId are not organized as a flag set and therefore cannot be OR-concatenated. Here one of them has to be used exclusively.

---

scComSAMLightClientCtrlStringDataIdFlagDontUpdateView **= 65536**

If this bit is set in the upper 16 bit of the parameter *DataId*, the view will not be refreshed. This can be helpful to increase the performance.

scComSAMLightClientCtrlStringDataIdFlagDontUpdateEntity **= 131072**

If this bit is set in the upper 16 bit of the parameter *DataId*, the entity will not be regenerated and updated. This can be helpful to increase the performance.

scComSAMLightClientCtrlStringDataIdFlagToplevelOnly **= 2097152**

If this bit is set, it will only be searched for entities in the first level of the job. This can be helpful to increase the performance.

scComSAMLightClientCtrlStringDataIdFlagSelected **= 4194304**

Can be used together with the types *scComSAMLightClientCtrlStringDataIdGetToplevelEntity* or *scComSAMLightClientCtrlStringDataIdSetToplevelEntity* to get or set the name of top level entity which is selected in SAMLight.

---

scComSAMLightClientCtrlStringDataIdTextFontName **= 1**

The parameter data contains the font name.

scComSAMLightClientCtrlStringDataIdTextText **= 2**

The parameter data contains the string of the text object.

scComSAMLightClientCtrlStringDataIdGetToplevelEntity **= 17**

This value can be used together with the function SCGetIdStringData() to get the name of an entity that is specified by a zero based index number. Use scComSAMLightClientCtrlStringDataIdGetEntityName when you want to iterate trough all levels.

scComSAMLightClientCtrlStringDataIdSetBarcodeType **= 19**

Using this value a barcode object of type *ScBarCode12Chars2D* can be modified. Together with a call to ScSetEntityStringData()) a new barcode type can be specified and set for that object. The string parameter the function expects is the name of the barcode type. There for an example *EAN* or *I-2/5* can be used. This value can be combined with the flags for the upper 16 bit like described above.

scComSAMLightClientCtrlStringDataIdGetBarcodeType **= 20**

This value can be used to retrieve the type name of a *ScBarCode12Chars2D*. Since this does not modify the displayed barcode the upper 16 bits are ignored. So the view flags do not have any influence on an appropriate call to ScGetEntityStringData().

scComSAMLightClientCtrlStringDataIdGetEntityName **= 21**

This is used to retrieve the name of an entity by its index. The index is zero based and starts from the top level going through the tree on the base of the first object, going on with the siblings. Use scComSAMLightClientCtrlStringDataIdGetToplevelEntity when you want to iterate only through top level.

scComSAMLightClientCtrlStringDataIdGetEntityType = 22

This is used to retrieve the type of an entity by its index. The index is zero based and starts from the top level going through the tree on the base of the first object, going on with the siblings.

scComSAMLightClientCtrlStringDataIdSetEntityName = 23

This is used to set the name of an entity by its index. The index is zero based and starts from the top level going through the tree on the base of the first object, going on with the siblings. If not deactivated with a mode flag this will set the names of sub-entities also (groups, etc.).

scComSAMLightClientCtrlStringDataIdEntitySerialASCIIFileName = 24

This is used to change the file name of a serial number list in txt format. This is recommended to use with scComSAMLightClientCtrlExecCommandResetSerialNumber together.

scComSAMLightClientCtrlStringDataIdSetToplevelEntity = 25

This is used to set the name of an entity by its index. The index is zero based and starts from the top level going through the tree on the base of the first object, going on with the siblings. This will set the names of top-level-entities exclusively.

scComSAMLightClientCtrlStringDataIdSetMotionCtrls = 26

This allows to store motion data into a motion control object that is inside the currently loaded job. The format for the *Data* parameter is:

1) For one axis, parameters are separated by semicolon:
   < axis Index: *0..6* or *-1* for all axes > (can not be empty)
   < position as float value > (can be an empty string (""))
   < speed as float value > (can be an empty string)
   < relative movement *0* or *1* > (can be an empty string)
2) For all axes:
   Up to *7 one axis* strings, separated by one "*blank"* (" ").
3) For more than one motion entity - with one single function call:
   As standard for this case there are multiple all axes strings separated by a vertical tab ("\v").
   To activate this mode call:
   ScSetMode (ScGetMode() Or
   'scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon')
   Multiple entity names must be separated by a semicolon.

scComSAMLightClientCtrlStringDataIdSerialNumberFormatString = 27

This allows to get or set the format string of a Date Time Object.

scComSAMLightClientCtrlStringDataIdArrayCopyHard = 28

Performs an Array Copy where the copies are unique entities with unique names. The array copy parameters are specified with these Long and Double values:

scComSAMLightClientCtrlLongValueTypeEntityArrayCountX
scComSAMLightClientCtrlLongValueTypeEntityArrayCountY
scComSAMLightClientCtrlDoubleValueTypeEntityArrayStepX
scComSAMLightClientCtrlDoubleValueTypeEntityArrayStepY

scComSAMLightClientCtrlStringDataIdTranslate = 29

This allows to translate the entity or entities defined by EntityName. If used for multiple entities the Flag scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon with ScSetMode(...) has to be set. In this case the parameter Data holds for each single entity the translation vector as "X;Y" separated by a vertical Tab.

scComSAMLightClientCtrlStringDataIdRotate = 30

This allows to rotate the entity or entities defined by EntityName. If used for multiple entities the Flag scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon with ScSetMode(...) has to be set. In this case the parameter Data holds for each single entity the rotation vector as "CenterX;CenterY;Angle" separated by a vertical Tab.

scComSAMLightClientCtrlStringDataIdOutlineAndRotate = 31

This allows to get the Outline and the Rotation Angle of the Entities defined by EntityName. If used for multiple entities the Flag scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon with ScSetMode(...) has to be set. The parameter Data will hold the Outlines and Rotation Angle in a list in the format "MinX;MinY;MaxX;MaxY;Angle".

scComSAMLightClientCtrlStringDataIdBarCodeFormatString = 32

This allows to set the Format String of a Barcode (if supported)

scComSAMLightClientCtrlStringDataIdSetMotionCtrlsString = 33

This allows to store a new string into a string motion control object that is inside the currently loaded job.

scComSAMLightClientCtrlStringDataIdSpecialPenAndMore = 34

This allows to pass on or several pens separated by semicolon as 'entity name', e.g. "1" or "1;2;3" and to get the following 7 pen main parameters as return string separated by semicolon in the following order: marking speed [mm/s], power [Watt], frequency [Hz], CO2 power1 [pulse length of laser signal1 in %], CO2 power2 [pulse length of laser signal2 in %], pulse length [µs], first pulse [µs]. The values for another pen are separated by a 'vertical tab' '\v'. If the return string is send via network, it's maximum length could be 512 characters. Every value of the return string is a double value with 6 decimal places.

19.4.7 Long Cmd Ids

Constants for the function ScExecCommand().

---

scComSAMLightClientCtrlExecCommandTest = 1

Pops up a message box within SAMLight. This can be used to check the communication between the applications.

scComSAMLightClientCtrlExecCommandResetSequence = 2

This resets the marking sequence to its initial state. This is important for jobs using the beat count and beat offset parameters.

scComSAMLightClientCtrlExecCommandNewJob = 3

Deletes the current job.

scComSAMLightClientCtrlExecCommandFitViewToWorkingArea = 4

Fits the view to the working area.

scComSAMLightClientCtrlExecCommandFitViewToAllEntities = 5

Fits the view to all entities in the job.

scComSAMLightClientCtrlExecCommandFitViewToSelectedEntities = 6

Fits the view to the selected entities in the job.

scComSAMLightClientCtrlExecCommandResetCounter = 7

Resets the mark quantity counter. See chapter Mark Status Bar.

scComSAMLightClientCtrlExecCommandResetSerialNumber = 8

Resets the serial number.

scComSAMLightClientCtrlExecCommandUpdateScannerPos = 9

Update the position information of the scanner.

scComSAMLightClientCtrlExecCommandAutoCompensateOff = 10

Scanner auto-calibration functionality (works only with hardware that supports it): Turns auto calibration mode off

scComSAMLightClientCtrlExecCommandAutoCompensateRef = 11

Scanner auto-calibration functionality (works only with hardware that supports it): Turns auto calibration mode on and go to reference position for initial calibration

scComSAMLightClientCtrlExecCommandAutoCompensateCal = 12

Scanner auto-calibration functionality (works only with hardware that supports it): Recalibrates

scComSAMLightClientCtrlExecCommandResplitJob = 13

Resplit a job that is in *Splitting Mode* and was modified so that the split data have to be updated by this option.

scComSAMLightClientCtrlExecCommandMotionStopMove = 14

Stops motions which are defined with scComSAMLightClientCtrlLongValueTypeMotionAxis

scComSAMLightClientCtrlExecCommandMotionHome = 15

Calls the homing function for motions which are defined with scComSAMLightClientCtrlLongValueTypeMotionAxis

scComSAMLightClientCtrlExecCommandMotionGo = 16

Executes the movement for motions which are defined with *scComSAMLightClientCtrlLongValueTypeMotionAxis*. Thereby the value *scComSAMLightClientCtrlLongValueTypeMotionWaitForEnd* is taken into account. The position is defined with scComSAMLightClientCtrlDoubleValueTypeMotionAxisPosition or rather scComSAMLightClientCtrlDoubleValueTypeMotionAxisAngle according to the motion settings. Before sending a new *scComSAMLightClientCtrlExecCommandMotionGo* command it is important to wait until the last drive has stopped. See example Motion Control in the Examples section.

scComSAMLightClientCtrlExecCommandMotionSendString = 17

Sends a RS232 string to the port defined within the motion settings. The string is empty by default and can be defined with scComSAMLightClientCtrlStringValueTypeMotionString.

scComSAMLightClientCtrlExecCommandMotionUpdatePos = 18

Asks the motion drive for the currently stored position. The motion number is defined with scComSAMLightClientCtrlLongValueTypeMotionAxis.

scComSAMLightClientCtrlExecCommandStopExecution = 19

Stops the execution: If scComSAMLightClientCtrlMarkFlagWaitForTrigger is set with *ScSetMarkFlags* and mark is called with *ScMarkEntityByName*, the trigger mark dialog opens. The command closes the trigger mark dialog and disables trigger mode. This is necessary for example if a job is edited and new data need to be given to the scanner card. Then the trigger mode needs to be stopped and the marking has to be started again

scComSAMLightClientCtrlExecCommandRedPointerStart = 20

Starts the red pointer with the settings set at *Redpointer* in the mark dialog and opens the mark dialog if its closed.

scComSAMLightClientCtrlExecCommandRedPointerStop = 21

Stops the red pointer. The Mark dialog must be open.

scComSAMLightClientCtrlExecCommandUpdateViewNow = 22

Refreshes the view. Redraws all entities even if flag mode scComSAMLightClientCtrlModeFlagDontUpdateView is set.

scComSAMLightClientCtrlExecCommandIncSerialNumber = 23

Increment all serial numbers in the job.

scComSAMLightClientCtrlExecCommandDecSerialNumber = 24

Decrement all serial numbers in the job.

scComSAMLightClientCtrlExecCommandView3DViewIso = 26

switch a 3D View into ISO mode.

scComSAMLightClientCtrlExecCommandRehatchAll = 28

Updates the hatch of all entities in the current job.

scComSAMLightClientCtrlExecCommandCreateBeamCompedCopy = 37

Create a beam comped copy of the entity that is defined via *ScSetStringValue(scComSAMLightClientCtrlStringValueTypeStringPara1, "Name of Entity")*. The copy will be stored in the entity defined by *ScSetStringValue(scComSAMLightClientCtrlStringValueTypeStringPara2, "Name of Copy")*. The *Dist* parameter is defined by *ScSetDoubleValue(scComSAMLightClientCtrlDoubleValueTypeDoublePara1, Distance in mm)*

scComSAMLightClientCtrlStringValueTypeSaveSplitsJobFileName = 38

Using this constant can be used to save a spitted job file as an entity list of split tiles. This constant has to be executed after *ScSetStringValue scComSAMLightClientCtrlStringValueTypeSaveSplitsJobFileName*, see example "Save split job file as tiles".

scComSAMLightClientCtrlExecCommandCheckIfJobIsInField = 40

It returns 0, if the whole job is not inside the field or no 2D data in job exists.

scComSAMLightClientCtrlExecCommandExitTriggerMode = 41

Stops the *TriggerMode*.

scComSAMLightClientCtrlExecCommandOpenMarkDialog = 48

Opens the *Mark dialog*.

scComSAMLightClientCtrlExecCommandCloseMarkDialog = 49

Closes the *Mark dialog*.

scComSAMLightClientCtrlExecCommandCorrectSamLight = 50

Executes the correction of the currently used correction file with <SCAPS>\system\sc_calib_points.txt. The new correction file is applied in SAMLight and it is saved in the same folder as the previous used one. The correction mode can be chosen with scComSAMLightClientCtrlLongValueTypeCorrectionMode. An example can be found in the chapter Correct UCF. In sc_corr_table, this function is called 'Correct UCF by calibration points'.

scComSAMLightClientCtrlExecCommandSaveSettingsNow = 51

Saves the current settings to the settings file.

scComSAMLightClientCtrlExecCommandSortJobByName = 52

Sorts objects inside a group by their name if the name of the group is specified with scComSAMLightClientCtrlStringValueTypeSetToTopLevelEntity. If no group name is specified this sorts the top level entities of the job.

scComSAMLightClientCtrlExecCommandUndo = 54

Executes the UNDO command.

scComSAMLightClientCtrlExecCommandWizardOrder = 55

Executes the Data Wizard command "Set Order". There can be set 3 parameters with Client Control. See Double Value Types scComSAMLightClientCtrlDoubleValueTypeDoublePara1/2 and 3. This will affect all entities in the job.

scComSAMLightClientCtrlExecCommandWizardCreateOneGroup = 57

Executes the Data Wizard command "Create One Group".

scComSAMLightClientCtrlExecCommandStoreFlashSettings = 58

Stores the settings to the USC-2 EPCS (this is necessary for stand-alone operation). Make sure that the settings fit to the laser and other machinery. The settings will be loaded during powering on the card.

scComSAMLightClientCtrlExecCommandGroupEntities = 59

Groups the manually or by Long Data Ids scComSAMLightClientCtrlLongDataIdEntitySelected selected entities and puts them into an entities group.

scComSAMLightClientCtrlExecCommandUngroupEntities = 60

Ungroups the manually or by Long Data Ids scComSAMLightClientCtrlLongDataIdEntitySelected selected entities group. The view level of all entities inside the group will be decreased by one.

19.5 Examples

Demo program: A demo application *sc_client_control_interface_csharp* shows the integration of the *Client Control Interface*. It is a Micrsoft Visual Studio C# project which is available for free at

http://www.download.scaps.com/index.php?c=1&f=/downloads/client_control_interface/sc_client_control_interface_csharp.zip .

19.5.1 Rotate output matrix

The following function uses the OCX to mark an entity with the name *RotateEntity*. Then it rotates the output matrix for 20 degrees around the middle of the entity center and marks it again. This step will be repeated 19 times (example written in C#).

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

AxSAMLIGHT_CLIENT_CTRL_OCXLib.AxScSamlightClientCtrl m_samlight = axScSamlightClientCtrl1;

if( m_samlight.ScIsRunning() == 0 )
{
    MessageBox.Show( "SAMLight not found", "Warning" );
    return;
}

double min_x, min_y, max_x, max_y;
double center_x, center_y;
long i;
double ang_inc;
double act_angle;
string entity_name = // set entity name here, for example "RotateEntity";

// first we calculate the center of the entity
min_x = m_samlight.ScGetEntityOutline( entity_name, 0 );
min_y = m_samlight.ScGetEntityOutline( entity_name, 1 );
max_x = m_samlight.ScGetEntityOutline( entity_name, 3 );
max_y = m_samlight.ScGetEntityOutline( entity_name, 4 );
center_x = ( min_x + max_x ) / 2;
center_y = ( min_y + max_y ) / 2;

// here we do the loop
ang_inc = 20 * 2 * Math.PI / 360;
act_angle = 0;
for( i = 0; i < 19; i++ )
{
    m_samlight.ScOpticMatrixReset();
    m_samlight.ScOpticMatrixRotate( center_x, center_y, act_angle );
    m_samlight.ScMarkEntityByName( entity_name, 1 );
    act_angle = act_angle + ang_inc;
}
```

19.5.2 Fit to entity

The following C# source code shows the function to fit the view to a specific entity.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

string EntityName = // set entity name here, for example "EntityName";

// unselect all
axScSamlightClientCtrl1.ScSetEntityLongData( "",
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEntitySelected |
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView, 0 );

// select
axScSamlightClientCtrl1.ScSetEntityLongData( EntityName,
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEntitySelected, 1 );
axScSamlightClientCtrl1.ScExecCommand(
    ( int )
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandFitViewToSelectedEnti
ties );
```

19.5.3 Set array

The following C# source code shows the function to change the array parameters of an entity. It creates an array 3 x 4 with a step size of 3.3 in x and 4.5 in y direction. Each line is being marked from right to left.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

string EntityName = // set entity name here, for example "EntityName";

axScSamlightClientCtrl1.ScSetEntityLongData( EntityName,
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEntityArrayCountX |
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView, 3 );
axScSamlightClientCtrl1.ScSetEntityLongData( EntityName,
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEntityArrayCountY |
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView, 4 );
axScSamlightClientCtrl1.ScSetEntityLongData( EntityName,
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEntityArrayStepX |
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView,
3300 );
axScSamlightClientCtrl1.ScSetEntityLongData( EntityName,
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEntityArrayStepY |
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView,
4500 );
axScSamlightClientCtrl1.ScSetEntityLongData( EntityName,
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEntityArrayOrderFlags |
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView,
    ( int ) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlEntityArrayOrderFlagNegX );
axScSamlightClientCtrl1.ScExecCommand(
    ( int )
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandFitViewToAllEntities
);
```

19.5.4 Get and set outputs and inputs

The following C# functions demonstrate the advanced mask method to get and set outputs of the scanner control card via CCI. The pin to bit mapping of the inputs and outputs can be found in the IO chapter:

```
void Set_Output( int OutputBit, bool High )
{
  int Flag = 1 << ( OutputBit - 1 );
  int Mask = ( ~Flag ) << 16;
  if( !High )
    Flag = 0;
  int OptoIO = Mask | Flag;
  axScSamlightClientCtrl1.ScSetLongValue(
    ( int )
SAMLIGHT_CLIENT_CTRL_OCXLib.ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeO
ptoIO,
    OptoIO );
  return;
}

bool Get_Output( int OutputBit ) // USC only
{
  int Flag = 1 << ( OutputBit - 1 );
  int GetOptoOut = axScSamlightClientCtrl1.ScGetLongValue(
    ( int )
SAMLIGHT_CLIENT_CTRL_OCXLib.ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeG
etOptoOut );
  if( ( GetOptoOut & Flag ) == Flag )
    return true;
  else
    return false;
}

bool Get_Input( int InputBit )
{
  int Flag = 1 << ( InputBit - 1 );
  int OptoIO = axScSamlightClientCtrl1.ScGetLongValue(
    ( int )
SAMLIGHT_CLIENT_CTRL_OCXLib.ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeO
ptoIO );
  if( ( OptoIO & Flag ) == Flag )
    return true;
  else
    return false;
}
```

19.5.5 Get and set text properties

The following C# source code shows how to manipulate the properties of a text object by using the SAMLight Client Control interface.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

int dontupdate = ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateEntity;
string EntityName = // set entity name here, for example "TextEnity";

// set the text properties of the entity with name EntityName
axScSamlightClientCtrl1.ScSetEntityStringData( EntityName, dontupdate | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdTextText, "NewText" );
axScSamlightClientCtrl1.ScSetEntityStringData( EntityName, dontupdate | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdTextFontName, "Arial" );
axScSamlightClientCtrl1.ScSetEntityDoubleData( EntityName, dontupdate | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextSize, 2 );
axScSamlightClientCtrl1.ScSetEntityDoubleData( EntityName, dontupdate | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextCharSpacing, 0.95 );
axScSamlightClientCtrl1.ScSetEntityDoubleData( EntityName, dontupdate | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextLengthLimit, 4.5 );
axScSamlightClientCtrl1.ScSetEntityDoubleData( EntityName, dontupdate | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextHeightLimit, 1.5 );
axScSamlightClientCtrl1.ScSetEntityDoubleData( EntityName, dontupdate | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextRadius, 11.5 );
```

```
axScSamlightClientCtrl1.ScSetEntityDoubleData( EntityName, dontupdate | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextStartAngle, Math.PI / 2 );

int flags = axScSamlightClientCtrl1.ScGetEntityLongData( EntityName, ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdTextCharFlags );

// force to radial text
flags = flags | ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdTextCharFlagRadial;
axScSamlightClientCtrl1.ScSetEntityLongData( EntityName, ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdTextCharFlags, flags );

// get the text properties of the entity with name EntityName
string str = "";
double val = 0;

axScSamlightClientCtrl1.ScGetEntityStringData( EntityName, ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdTextText, ref str );
MessageBox.Show( "Text: " + str, "Text" );

axScSamlightClientCtrl1.ScGetEntityStringData( EntityName, ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdTextFontName, ref str );
MessageBox.Show( "FontName: " + str, "FontName" );

axScSamlightClientCtrl1.ScGetEntityDoubleData( EntityName, ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextSize, ref val );
MessageBox.Show( "TextSize: " + val.ToString(), "TextSize" );
```

19.5.6 Retrieve entities

The following C# code uses SAMLight Client Control first to list all toplevel entities and then all entities including their possible entity names by message box. If further parameters of all entities like the set pen number or the Mark Loop Count should be requested, empty entities have to be renamed to a unique entity name first. Then parameter has to be requested by entity settings command and entity name has to be reset again. To speed up an automatic entity scan the command `ScSetMode` can be used similar as in Optimize Performance Example 2.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

AxSAMLIGHT_CLIENT_CTRL_OCXLib.AxScSamlightClientCtrl m_samlight = axScSamlightClientCtrl1;

string name = "", type = ""; // important: the BSTR-pointer has to be initialized, otherwise the
COM-interface may crash!
int i, count;

// step through all available toplevel entities
m_samlight.ScSetMode( ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlModeFlagTopLevelOnly );
count = m_samlight.ScGetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeTopLevelEntityNum );
if( count > 0 )
{
    for( i = 0; i < count; i++ )
    {
        // get the name of the entity at the index position "i"
        m_samlight.ScGetIDStringData( ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdGetToplevelEntity, i, ref name );
        if( name == "" )
            name = "'empty'";
        m_samlight.ScGetIDStringData( ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdGetEntityType, i, ref type );
        MessageBox.Show( "entity number: " + ( i + 1 ) + ", entity name: " + name + ", entity type:
" + type, "total number of toplevel entities: " + count.ToString() );
    }
}
else MessageBox.Show( "No top level entities found", "total number of toplevel entities: 0" );

// step through all available entities
m_samlight.ScSetMode( 0 );
count = m_samlight.ScGetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeTotalEntityNum );
if( count > 0 )
```

```
{
    for( i = 0; i < count; i++ )
    {
        // get the name and type of the entity at the index position "i"
        m_samlight.ScGetIDStringData( ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdGetEntityName, i, ref name );
        if(name == "") name = "'empty'";
        m_samlight.ScGetIDStringData( ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdGetEntityType, i, ref type );
        MessageBox.Show( "entity number: " + (i + 1) + ", entity name: " + name + ", entity type: "
+ type, "total number of entities: " + count.ToString() );
    }
}
else MessageBox.Show( "No entities found", "total number of entities: 0" );
```

19.5.7 Motion control

The following C# source code shows for MDrive motors (Type=5) how to set the absolute/relative position/ angle and speed of two drives, execute the starting command and retrieve a message that both drives (1. part) and the first/second (2. part) drive has stopped respectively. In this context it is important to wait until the last drive has stopped before sending a new starting command (MotionGo; therefore the stop messages). Otherwise the motion control would not work correctly. If one drive reaches its position limit given in the motion settings file or has already reached its new position before, it does not move any more.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

// the application comes back immediately
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionWaitForEnd
, 0 );

// 1. part: check if all MDrive motors have stopped
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionAxis, 0 );
axScSamlightClientCtrl1.ScSetDoubleValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTypeMotionAxisPosi
tion, 100 );
axScSamlightClientCtrl1.ScSetDoubleValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTypeMotionAxisSpee
d, 4 );

axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionAxis, 1 );
axScSamlightClientCtrl1.ScSetDoubleValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTypeMotionAxisAngl
eRelative, 720 );
axScSamlightClientCtrl1.ScSetDoubleValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTypeMotionAxisSpee
d, 1 );

axScSamlightClientCtrl1.ScExecCommand( ( int )
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandMotionGo
);

// -1 means all axes
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionAxis, -
1 );
do
{
    axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionAxis, -
```

```
1 );
} while( axScSamlightClientCtrl1.ScGetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionMoving )
== 1 );

MessageBox.Show( "All motors have stopped." );

// 2. part: check if single MDrive motor has stopped
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionAxis, 0 );
axScSamlightClientCtrl1.ScSetDoubleValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTypeMotionAxisPosi
tionRelative, 100 );

axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionAxis, 1 );
axScSamlightClientCtrl1.ScSetDoubleValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTypeMotionAxisAngl
e, 720 );
axScSamlightClientCtrl1.ScExecCommand( ( int )
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandMotionGo
);

int axis0 = 1;
int axis1 = 1;

do
{
    if( axis0 == 1 )
    {
        axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionAxis, 0 );
        if( axScSamlightClientCtrl1.ScGetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionMoving )
== 0 )
        {
            MessageBox.Show( "Motor with axis 0 has stopped." );
            axis0 = 0;
        }
    }

    if( axis1 == 1 )
    {
        axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionAxis, 1 );
        if( axScSamlightClientCtrl1.ScGetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeMotionMoving )
== 0 )
        {
            MessageBox.Show( "Motor with axis 1 has stopped." );
            axis1 = 0;
        }
    }
} while ( ( axis0 == 1 ) || ( axis1 == 1 ) ) ;
```

19.5.8 Precalculating the mark time

```
// Get expected mark time of complete job
double LastExpectedMarkTime = axScSamlightClientCtrl1.ScGetDoubleValue(
  ( int )
SAMLIGHT_CLIENT_CTRL_OCXLib.ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTyp
eLastExpectedMarkTime );

// Alternative: Get process time of chosen entities
// Enable MarkFlagPreview
int MarkFlags = axScSamlightClientCtrl1.ScGetMarkFlags();
MarkFlags = MarkFlags | ( int )
SAMLIGHT_CLIENT_CTRL_OCXLib.ScComSAMLightClientCtrlMarkFlags.scComSAMLightClientCtrlMarkFlagPreview
;
int CciReturn1 = axScSamlightClientCtrl1.ScSetMarkFlags( MarkFlags );

// Enable ModeFlagEntityNamesSeparatedBySemicolon
int ModeFlags = 0;
int CciReturn2 = axScSamlightClientCtrl1.ScGetMode( ref ModeFlags );
ModeFlags = MarkFlags | ( int )
SAMLIGHT_CLIENT_CTRL_OCXLib.ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlModeFlagEntityNames
SeparatedBySemicolon;
int CciReturn3 = axScSamlightClientCtrl1.ScSetMode( ModeFlags );

// Preview of entities 'a and 'b'
int CciReturn4 = axScSamlightClientCtrl1.ScMarkEntityByName( "a;b", 1 );

// Get process time of last preview
double LastPreviewTime = axScSamlightClientCtrl1.ScGetDoubleValue(
  ( int )
SAMLIGHT_CLIENT_CTRL_OCXLib.ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTyp
eLastPreviewTime );
```

19.5.9 Create a beam compensated copy of an entity

The following C# example will create a beam compensated copy out of the entity "Entity" with the name "BeamCompedEntity". The parameter *Dist* is defined in line 3 in mm.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeStringPara1,
"Entity" );
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeStringPara2,
"BeamCompedEntity" );
axScSamlightClientCtrl1.ScSetDoubleValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTypeDoublePara1,
0.1 );
axScSamlightClientCtrl1.ScExecCommand( ( int )
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandCreateBe
amCompedCopy );
```

19.5.10 Create a SAMLight View2D screenshot

The following example shows how to create SAMLight View2D screenshots (example written in C#)

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

string path = // set path here, for example
Environment.GetFolderPath( Environment.SpecialFolder.Desktop );

// generating fixed size colored View2D screenshots with SAMLight Windows opened in background
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeSaveView2D160, path + "\
\ScreenshotView2D160.bmp" );
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeSaveView2D320, path + "\
\ScreenshotView2D320.bmp" );
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeSaveView2DFull, path + "\
\ScreenshotView2DFull.bmp" );

// generating variable size colored View2D screenshot with SAMLight Windows opened in background
axScSamlightClientCtrl1.ScSetDoubleValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapVariableSiz
e, 640 ); // set bitmap size in pixel
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeSaveView2DVariableSize,
path + "\\ScreenshotView2DVariableSiz.bmp" );

// generating black and white View2D screenshot with just SAMLight running background
axScSamlightClientCtrl1.ScSetDoubleValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapDPI, 200 );

axScSamlightClientCtrl1.ScSetLongValue(   ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeSaveView2DBitmapMode, 0 ); //
 0: lines and pixels are drawn normal
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeSaveView2DAdjustableDPI,
path + "\\ScreenshotView2DAdjustableDPI_DrawnNormal.bmp" );

axScSamlightClientCtrl1.ScSetLongValue(   ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeSaveView2DBitmapMode, 1 ); //
 1: lines and pixels are drawn thicker
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeSaveView2DAdjustableDPI,
path + "\\ScreenshotView2DAdjustableDPI_DrawnThicker.bmp" );
```

19.5.11 Save splitted job file as tiles

The following C# example shows how to split a job file and save it as an entity list of single split tiles with full access, which can be used as a basis to manually setup a split job file according to own needs (motion entities have to be added manually). To use this example a splitting mode with reasonable splitting parameters and motion axes has to be defined in SAMLight before.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

axScSamlightClientCtrl1.ScExecCommand( ( int)
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandNewJob );
axScSamlightClientCtrl1.ScLoadJob( Environment.ExpandEnvironmentVariables( @"%SCAPS_SAM%" ) + "\
\jobfiles\\demo.sjf", 1, 1, 0 );
ScCenterEntityByEntityName( "" );
axScSamlightClientCtrl1.ScExecCommand( ( int )
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandResplitJob ); //
scComSAMLightClientCtrlExecCommandResplitJob = 13
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeSaveSplitsJobFileName,
Environment.ExpandEnvironmentVariables( @"%SCAPS_SAM%" ) + "\\jobfiles\
\demo_splitted_as_tiles.sjf" );
axScSamlightClientCtrl1.ScExecCommand( ( int )
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandSaveSplitsAsEntities
); // scComSAMLightClientCtrlExecCommandSaveSplitsAsEntities = 38
axScSamlightClientCtrl1.ScLoadJob( Environment.ExpandEnvironmentVariables( @"%SCAPS_SAM%" ) + "\
\jobfiles\\demo_splitted_as_tiles.sjf", 1, 1, 0 );
ScCenterEntityByEntityName( "" );
}

private long ScCenterEntityByEntityName( string entityname )
{
    double MIN_X = axScSamlightClientCtrl1.ScGetWorkingArea( 0 ); //
SC_SAMLIGHT_OUTLINE_INDEX_MIN_X = 0
    double MIN_Y = axScSamlightClientCtrl1.ScGetWorkingArea( 1 ); //
SC_SAMLIGHT_OUTLINE_INDEX_MIN_Y = 1
    double MAX_X = axScSamlightClientCtrl1.ScGetWorkingArea( 3 ); //
SC_SAMLIGHT_OUTLINE_INDEX_MAX_X = 3
    double MAX_Y = axScSamlightClientCtrl1.ScGetWorkingArea( 4 ); //
SC_SAMLIGHT_OUTLINE_INDEX_MAX_Y = 4

    double CENTER_WORKING_AREA_X = ( MAX_X + MIN_X ) / 2;
    double CENTER_WORKING_AREA_Y = ( MAX_Y + MIN_Y ) / 2;

    double min_x = axScSamlightClientCtrl1.ScGetEntityOutline( entityname, 0 ); //
scComSAMLightClientCtrlOutlineIndexMinX = 0
    double min_y = axScSamlightClientCtrl1.ScGetEntityOutline( entityname, 1 ); //
scComSAMLightClientCtrlOutlineIndexMinY = 1
    double max_x = axScSamlightClientCtrl1.ScGetEntityOutline( entityname, 3 ); //
scComSAMLightClientCtrlOutlineIndexMaxX = 3
    double max_y = axScSamlightClientCtrl1.ScGetEntityOutline( entityname, 4 ); //
scComSAMLightClientCtrlOutlineIndexMaxY = 4

    double center_entity_outline_x = ( max_x + min_x ) / 2;
    double center_entity_outline_y = ( max_y + min_y ) / 2;

    return axScSamlightClientCtrl1.ScTranslateEntity( entityname, CENTER_WORKING_AREA_X -
center_entity_outline_x, CENTER_WORKING_AREA_Y - center_entity_outline_y, 0 );
}
```

19.5.12 Deactivating visual updates

The following C# source code shows how to increase the speed of Client Control commands by deactivating visual updates in the View2D area.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

// Deactivate visual updates in the View2D area:
axScSamlightClientCtrl1.ScSetMode( ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlModeFlagDontUpdateView );
```

```
// Insert here your Client Control commands with visual effects like
ScTranslateEntity, ScScaleEntity, ScRotateEntity, ScChangeTextByName, ...

// Reactivate visual updates in the View2D area:
axScSamlightClientCtrl1.ScSetMode( 0 );  // deactivate ScSetMode flags
axScSamlightClientCtrl1.ScExecCommand( ( int )
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandUpdateVi
ewNow );
```

19.5.13 Mirror entity on Y axis

The following C# source code shows how to mirror an entity on the Y axis. To mirror on X axis, just switch the -1,1 to 1,-1 in the ScScaleEntity command. The Entity has the name "A" in this example.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

string EntityName = "A";
double Xmin, Ymin, Xmax, Ymax, Xc, Yc;

// Get Co-ordinates of the center point of the entity
Xmin = axScSamlightClientCtrl1.ScGetEntityOutline( EntityName, 0 );
Ymin = axScSamlightClientCtrl1.ScGetEntityOutline( EntityName, 1 );
Xmax = axScSamlightClientCtrl1.ScGetEntityOutline( EntityName, 3 );
Ymax = axScSamlightClientCtrl1.ScGetEntityOutline( EntityName, 4 );
Xc = ( Xmin + Xmax ) / 2;
Yc = ( Ymin + Ymax ) / 2;

// Translate entity to the center of the working field
axScSamlightClientCtrl1.ScTranslateEntity( EntityName, -Xc, -Yc, 0 );

// Mirror Entity
axScSamlightClientCtrl1.ScScaleEntity( EntityName, -1, 1, 1 );

// Translate entity back to its original position
axScSamlightClientCtrl1.ScTranslateEntity( EntityName, Xc, Yc, 0 );
```

19.5.14 Correct UCF

The function 'sc_corr_table → Correction → Correct UCF by calibration points' can also be used via CCI.

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

// Select Fit2D correction algorithm
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeCorrectionMode,
( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlCorrectionMode2dFit );

// Select Fit2D order, for example 3_Advanced
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueType2dFitType, 3 );

// Execute the correction with <SCAPS>\system\sc_calib_points.txt and apply the new
correction file in SAMLight.
axScSamlightClientCtrl1.ScExecCommand( ( int )
ScComSAMLightClientCtrlExecCommandConstants
.scComSAMLightClientCtrlExecCommandCorrectSamLight );
```

19.5.15 Camera image as background in View 2D

Regarding camera integration please look at the following example in C# at section "Use camera as background image" in chapter "How To".

19.6 Optimize Performance

The performance of SAMLight and thus your client control application can often be optimized by suppressing time consuming and unnecessary actions of the program. If a property of an entity (like the text or the size) has been changed by client control the entity will be regenerated and updated. Furthermore the View2D in SAMLight will be updated as well. Usually these updates of the entity or the View2D are not necessary after every client control call. Here are some examples how the performance can be increased:

Execute SAMLight Client Control command only for the top level entities of a job file:

If it is not necessary to execute a SAMLight Client Control command for all subentities of a top level entity then the iteration through all subentities can be globally suppressed and the SAMLight Client Control command is executed for the top level entity only:

ScSetMode () with flag scComSAMLightClientCtrlModeFlagTopLevelOnly

Disable automatic View2D update of SAMLight:
If you are not using the View2D of SAMLight because you launch SAMLight in hidden mode or your application is automatized without needing a visual view you can disable the View2D completely:

ScSetMode () with flag scComSAMLightClientCtrlModeFlagDontUpdateView

Execute SAMLight Client Control command for several entities by only one function call with Entity Names Separated By Semicolon:

The time consuming part of a client control application can be the COM call itself. The processing time for

the client interface application can be reduced if you want to use the same CI call for several entities. An example is changing the text of several entities with just one COM call. This can be done by using semicolon separated entity names which has to be activated by:

ScSetMode() with flag scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon

To find more information about which client control calls are possible and how to use it please refer to the corresponding documentation or see the example below.

Update SAMLight View2D manually:

If you have disabled the automatic View2D update of SAMLight but want to update it once:

ScExecCommand() with flag scComSAMLightClientCtrlExecCommandUpdateViewNow

Disable View2D entity update if you change more than one property of the same entity at once:

Like mentioned above every entity change will be shown in View2D after any change of its properties. If you want to change more than one property of the same entity you can suppress View2D update of the entity so the change of the entity will not be shown after after every call.

ScSetEntityLongData() with flag scComSAMLightClientCtrlLongDataIdFlagDontUpdateView

ScSetEntityDoubleData() with flag scComSAMLightClientCtrlDoubleDataIdFlagDontUpdateView

ScSetEntityStringData() with flag scComSAMLightClientCtrlStringDataIdFlagDontUpdateView

Supress entity update if you change more than one property of the same entity at once:

Like mentioned above every entity will be regenerated and updated after any change of its properties. If you want to change more than one property of the same entity you can suppress this behavior so that the entity will not be regenerated and updated after after every call.

ScSetEntityLongData() with flag scComSAMLightClientCtrlLongDataIdFlagDontUpdateEntity

ScSetEntityDoubleData() with flag scComSAMLightClientCtrlDoubleDataIdFlagDontUpdateEntity

ScSetEntityStringData() with flag scComSAMLightClientCtrlStringDataIdFlagDontUpdateEntity

Please make sure that you are not using this flag at the last change of an entity. Otherwise this entity will not be regenerated and updated at all.

Address top level entities only:

For many CI calls you address an entity by its name or ID to set or get its parameters. To find this entity SAMLight is looking for the name or ID in every group or subgroup of the entire entity hierarchy. To decrease the time which is necessary to find the corresponding entity you want to address the search can be limited to top level entities:

ScSetEntityLongData() with flag scComSAMLightClientCtrlLongDataIdFlagToplevelOnly

ScSetEntityDoubleData() with flag scComSAMLightClientCtrlDoubleDataIdFlagToplevelOnly

ScSetEntityStringData() with flag scComSAMLightClientCtrlStringDataIdFlagToplevelOnly

Use trigger mode:

In this mode the next job to be marked is prepared and sent to the controller card before the mark call has

been sent. As soon as the external trigger is recognized by the controller card the marking process will start. If this mode is not enabled the job will be prepared and sent to the controller card after the external trigger was recognized.

ScSetMarkFlags() with flag scComSAMLightClientCtrlMarkFlagWaitForTrigger

19.6.1 Example 1

To demonstrate how to use the CI calls properly we want to give an example code. Consider you have 20 barcode entities in SAMLight with the entity names barcode1, barcode2, ..., barcode20. We want to change for each barcode

- The text
- Enable Hatch1 with style 'wavy line without marking the jumps'
- Enable Hatch2 with style 'zigzag'

A common way to program this changes in C# would be:

```
AxScSamlightClientCtrl m_samlight = axScSamlightClientCtrl1;

//step to each of the 20 barcode entities
for( int i = 1; i <= 20; i++ )
  {
  //generate the entity name and the entity text for each of the 20 barcodes
  string entity_name = "barcode" + Convert.ToString( i );
  string entity_text = "This is the text of " + entity_name;

  //change the text of each entity
  m_samlight.ScChangeTextByName( entity_name, entity_text );

  //enable Hatching1
  m_samlight.ScSetEntityLongData( entity_name, ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEnableHatching1, 1 );

  //enable Hatching2
  m_samlight.ScSetEntityLongData( entity_name, ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEnableHatching2, 6 );
  }
```

This way will work properly but after every of the 60 CI calls the corresponding entity will get regenerated and the View2D of SAMLight will be updated. This needs a lot of time.

We will now demonstrate how the same goal can be reached by using 3 CI calls instead of 60. Furthermore every entity will be regenerated only once and the View2D of SAMLight will only be updated after the last entity change was completed:

```
AxScSamlightClientCtrl m_samlight = axScSamlightClientCtrl1;

//Generate one string 'all_entity_names' with all entity names separated by ";"
//and generate one string 'all_entity_texts' with all entity textes separated by "\v"

string all_entity_names = "barcode1";
string all_entity_texts = "This is the text of barcode 1";

for( int i = 2; i <= 20; i++ )
  {
  all_entity_names = all_entity_names + ";" + "barcode" + Convert.ToString( i );
  all_entity_texts = all_entity_texts + "\v" + "This is the text of barcode" +
Convert.ToString( i );
  }

//enable entity name separated by semicolon mode
//disable View2D
//enable top level entities only
m_samlight.ScSetMode( ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon |
                      ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlModeFlagDontUpdateView |
                      ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlModeFlagTopLevelOnly);

//change all texts of all entities
```

```
m_samlight.ScChangeTextByName( all_entity_names, all_entity_texts );

//enable Hatching1 and Hatching2 for all entities, suppress entity update after enabling Hatching
1
//so the entities will be updated after the enabling of Hatching 2 (second call)
m_samlight.ScSetEntityLongData( all_entity_names,
                        ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEnableHatching1 |
                        ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdFlagDontUpdateEntity , 1 );
m_samlight.ScSetEntityLongData( all_entity_names,
                        ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEnableHatching2, 6 );

//reset ScSetMode
m_samlight.ScSetMode( 0 );

//manually update View2D
m_samlight.ScExecCommand( ( int )
ScComSAMLightExecCommandConstants.scComSAMLightClientCtrlExecCommandUpdateViewNow );
```

19.6.2 Example 2

The following C# example shows and measures how much SAMLight Client Control performance constants can accelerate the executed SAMLight Client Control commands .

In this case the cursive font, text size, font type and the text content of a Text2D entity with entity name "text entity" are set 100 times as substitute for 100 different text entities to show how SAMLight Client Control performance constants can be used.

```
...
using System.Diagnostics;
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

//SAMLight Client Control commands to change a text entities: cursive font,
//text size, font type; then text content is set

//Standard commands without acceleration, execution speed: 1,0 x

Stopwatch stopWatch = new Stopwatch();
stopWatch.Start();

for (int i = 1; i <= 100; ++i)
{
  axScSamlightClientCtrl1.ScSetEntityLongData("text entity",
      (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdTextCharFlags,
      (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdTextCharFlagItalic);
  axScSamlightClientCtrl1.ScSetEntityDoubleData("text entity",
      (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextSize, 20);
  axScSamlightClientCtrl1.ScSetEntityStringData("text entity",
      (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdTextFontName, "Arial");
  axScSamlightClientCtrl1.ScChangeTextByName("text entity", "new text");
}

stopWatch.Stop();
TimeSpan mt = stopWatch.Elapsed;
string MeasuredTime = String.Format("{0:00}.{1:00}", mt.Seconds, mt.Milliseconds);

MessageBox.Show("measured time without acceleration: " + MeasuredTime + " s");

stopWatch.Reset();


//General possibility to accelerate SAMLight Client Control commands by ScSetMode(),
//execution speed: ~ 1,5 - 2,0 x faster in this example

axScSamlightClientCtrl1.ScSetMode((int)
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlModeFlagTopLevelOnly |
                                  (int)
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlModeFlagDontUpdateView);

stopWatch.Start();
```

```
for (int i = 1; i <= 100; ++i)
{
    axScSamlightClientCtrl1.ScSetEntityLongData("text entity",
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdTextCharFlags,
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdTextCharFlagItalic);
    axScSamlightClientCtrl1.ScSetEntityDoubleData("text entity",
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextSize, 20);
    axScSamlightClientCtrl1.ScSetEntityStringData("text entity",
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdTextFontName,
"Arial");
    axScSamlightClientCtrl1.ScChangeTextByName("text entity", "new text");
}

axScSamlightClientCtrl1.ScSetMode(0);

axScSamlightClientCtrl1.ScExecCommand(
    (int)
ScComSAMLightClientCtrlExecCommandConstants.scComSAMLightClientCtrlExecCommandUpdateViewNow);

stopWatch.Stop();
mt = stopWatch.Elapsed;
MeasuredTime = String.Format("{0:00}.{1:00}", mt.Seconds, mt.Milliseconds);

MessageBox.Show("measured time with medium acceleration: " + MeasuredTime + " s");

stopWatch.Reset();


//Special possibility to accelerate ScSetEntityLongData, ScSetEntityDoubleData or
//ScSetEntityStringData commands, execution speed: ~ 3,5 x faster in this example

stopWatch.Start();

for (int i = 1; i <= 100; ++i)
{
    axScSamlightClientCtrl1.ScSetEntityLongData("text entity",
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdTextCharFlags |
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView |
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateEntity |
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagToplevelOnly,
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdTextCharFlagItalic);
    axScSamlightClientCtrl1.ScSetEntityDoubleData("text entity",
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlDoubleDataIdTextSize |
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView |
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateEntity |
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagToplevelOnly, 20);
    axScSamlightClientCtrl1.ScSetEntityStringData("text entity",
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlStringDataIdTextFontName |
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateView |
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagDontUpdateEntity |
        (int) ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdFlagToplevelOnly,
"Arial");
    axScSamlightClientCtrl1.ScChangeTextByName("text entity", "new text");
}

stopWatch.Stop();
mt = stopWatch.Elapsed;
MeasuredTime = String.Format("{0:00}.{1:00}", mt.Seconds, mt.Milliseconds);

MessageBox.Show("measured time with high acceleration: " + MeasuredTime + " s");

stopWatch.Reset();
```

20 How to

This chapter contains tutorials for special topics.

20.1 Use Simple Fonts

This tutorial describes how to use the simple line fonts (Laser Fonts) in the SAM modules.

20.1.1 Simple Fonts Format

The simple fonts have to be stored as Windows TrueType fonts. They are generated with the SCAPS converter. The SAM software detects the type and only generates lines. The advantage of using the TrueType Format is that the fonts can also be displayed in any Windows software supporting TrueType Fonts.

Figure 342: Example TrueType fonts in SAMLight

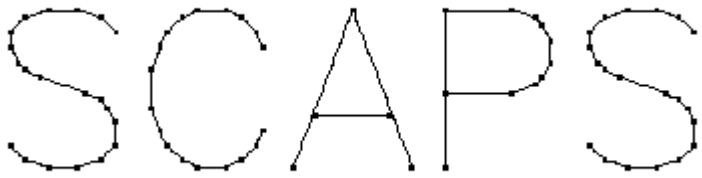
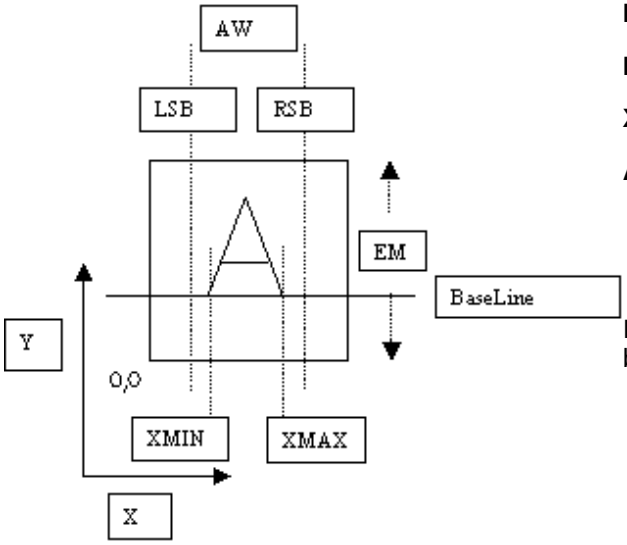


Figure 343: A closer look to a single line font

20.1.2 Generate Fonts

For the generation of "Simple" Windows True Type Fonts, SCAPS provides a converter tool for the translation of ASCII format files (SCAPS Font Format) to Windows True Type.

Each character of the font is defined with respect to a square character cell. The origin of the cell is the lower left corner. Also, each character (or also called Glyph) has a minimum X value XMIN and a maximum X Value XMAX (the bounding box X-co-ordinates).



EM: Size of square character cell.

MIN: Beginning of the character.

XMAX: End of the character.

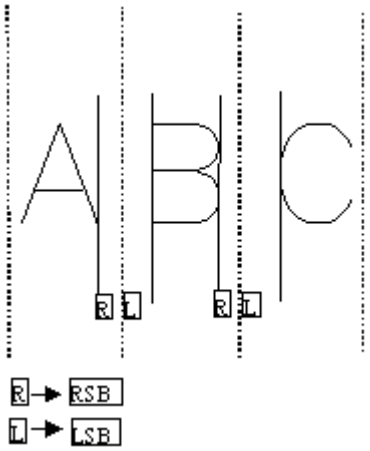
AW: Advanced Width as

AW = LSB + RSB +(XMAX-XMIN) , or

RSB = AW-LSB-(XMAX-XMIN)

In True Type only AW and LSB are defined. RSB can be calculated according the above equation.

Figure 344: TrueType font parameters 1



LSB: LeftSideBearing – Defines the gap between the RSB of the last character to the beginning of this character (XMIN).

RSB: RightSideBearing - Defines the gap between the end of this character (XMAX) to the LSB of the next character.

Figure 345: TrueType font parameters 2

The baseline defines the line between the ascent and descent parameters of the font.

Figure 346: TrueType font parameters 3

The sum of Ascent and Descent plus an additional line spacing is the height of the font. BaseLine, Height, Ascent and Descent are global parameters of a specific font design.

20.1.2.1 Scaps Font Format

The SCAPS Font Format is an ASCII font description file format easy to read and to generate. For an example look at the file sc_straight_prop.sff in the folder INSTALLDIR/fonts which is the font used in the example below. Version 2.0 works with a EM of 8000 units in contrast to version 1.0 which works with an EM of 800 units.

Character description:

```
// CHAR 39
```

```
SI1280,640;SP1;PU;PA4000,7040;PD;PA4000,5360;PU;
```

Each character begins with // CHAR #, where # is the ANSI code of the character. The command SI specifies the AW and LSB values as described above. The command SP is optional and will be ignored. In the following line up to the next character description there is the geometrical information of the lines. The point information is stored in HPGL format by using the commands PU, PD and PA.

Header: The header is necessary for the font converter.

```
// SCAPS FONT FILE
// VERSION 2.0
```

Baseline: Y Distance from (0,0)

```
// BASELINE 2000
```

Examples:

```
// CHAR 0
SI8000,0;SP1;PU;PA0,0;PA4720,3440,4480,2960,4480,2480,4960,2000,5440,
2000,5920,2240,6160,2720,6160,3200,5680,3680,5200,3680;PU;PD;PA8000,
0,8000,8000,0,8000,0,0;PU;
// CHAR 33
SI1760,640;SP1;PU;PA4000,7040;PD;PA4000,3680;PU;SP1;PU;PA4000,2480;
PD;PA3760,2240,4000,2000,4240,2240,4000,2480;PU;
// CHAR 34
SI1760,640;SP1;PU;PA4240,7040;PD;PA4000,6800,3760,6320,3760,5840,
4000,5600,4240,5840,4000,6080;PU;
// CHAR 35
SI4880,640;SP1;PU;PA4120,8000;PD;PA2440,320;PU;PA5560,8000;PD;PA3880,
320;PU;PA2440,4880;PD;PA5800,4880;PU;PA2200,3440;PD;PA5560,3440;PU;
// CHAR 36
SI4640,640;SP1;PU;PA3520,8000;PD;PA3520,1040;PU;PA4480,8000;PD;
PA4480,1040;PU;SP1;PU;PA5680,6320;PD;PA5200,6800,4480,7040,3520,
7040,2800,6800,2320,6320,2320,5840,2560,5360,2800,5120,3280,4880,
4720,4400,5200,4160,5440,3920,5680,3440,5680,2720,5200,2240,4480,
2000,3520,2000,2800,2240,2320,2720;PU;
// CHAR 37
SI5600,640;SP1;PU;PA6160,7040;PD;PA1840,2000;PU;SP1;PU;PA3040,7040;
PD;PA3520,6560,3520,6080,3280,5600,2800,5360,2320,5360,1840,5840,
1840,6320,2080,6800,2560,7040,3040,7040,3520,6800,4240,6560,4960,
6560,5680,6800,6160,7040;PU;SP1;PU;PA5200,3680;PD;
// CHAR 38
SI6080,640;SP1;PU;PA6400,4880;PD;PA6400,5120,6160,5360,5920,5360,
5680,5120,5440,4640,4960,3440,4480,2720,4000,2240,3520,2000,2560,
2000,2080,2240,1840,2480,1600,2960,1600,3440,1840,3920,2080,4160,
3760,5120,4000,5360,4240,5840,4240,6320,4000,6800,3520,7040,3040,
6800,2800,6320,2800,5840,3040,5120,3520,4400,4720,2720,5200,2240,
5680,2000,6160,2000,6400,2240,6400,2480;PU;
// CHAR 39
SI1280,640;SP1;PU;PA4000,7040;PD;PA4000,5360;PU;
```

20.1.2.2 Scaps Converter

This tool 'sc_font_converter.exe' converts SCAPS Font Files into Windows True Type Font files. It can be found in the directory <SCAPS>\tools. It is currently limited to fonts with a character IDs from 0 to 255. The converter tool maps the EM size of 8000 to a design size of 10 mm. This means, that all numbers and dimensions shown in the converter are valid for a font generated with 10 mm height.

Please note that this tool will only run with a present SCAPS dongle and not with a trial key.

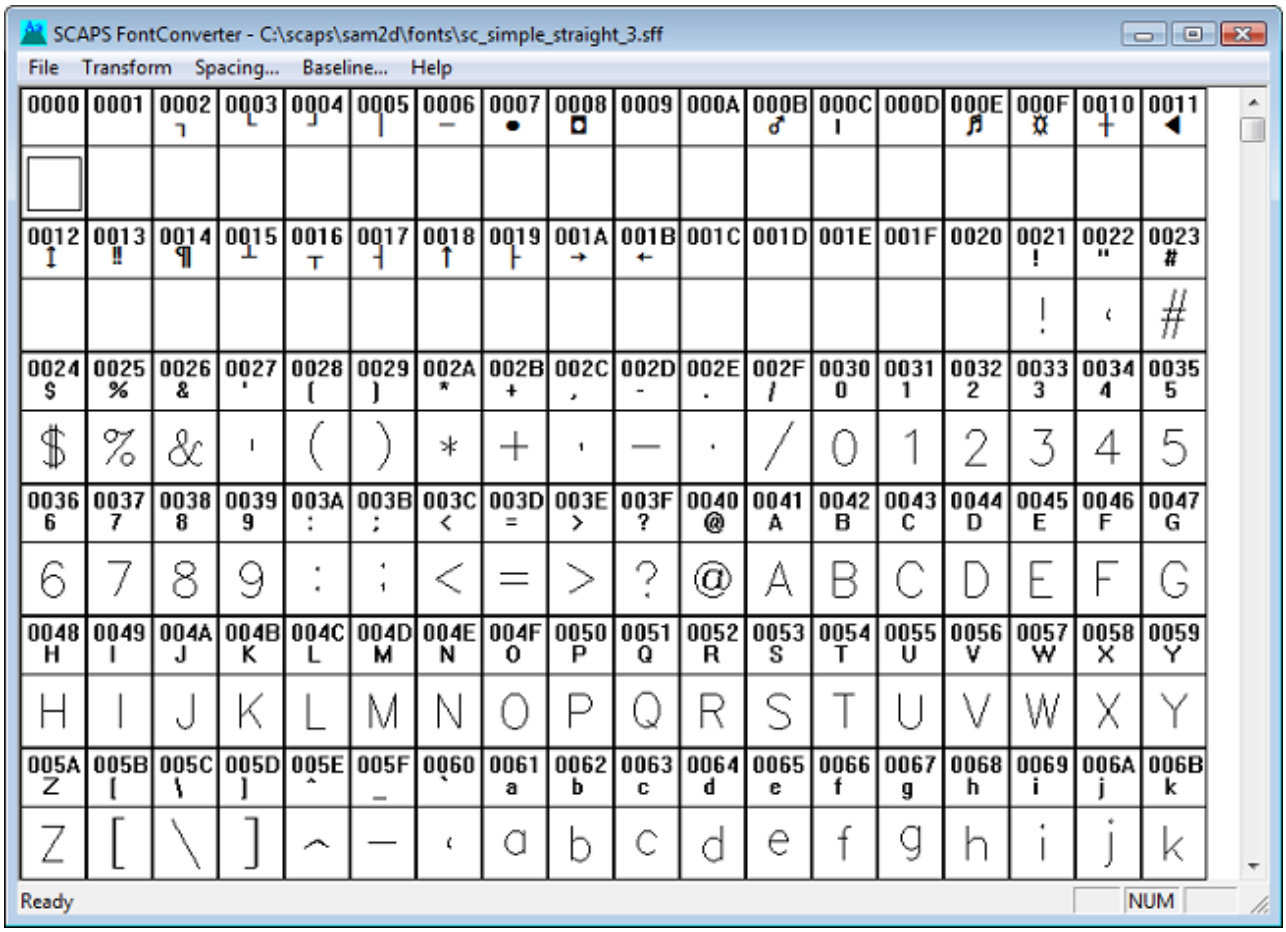


Figure 347: Font Converter Main View

File: The commands New, Load ; Save and Save As are handling *.sff files. Character 0 will always have a rectangle glyph with maximum size ((0,0)(8000,8000)) EM → ((0,0)(10,10)) mm Design, independent from what is stored in the *.sff file. Clicking on the glyph cells highlights the corresponding cell. Double Clicking on the cell activates the Edit View. The command Convert opens the Convert Dialog.

Transform: The dialog scale allows a global uniform character scale. Baseline, AW and LSB parameters will be scaled also. The scale origin is (0,0). Character 0 (the reference character) will not be scaled.

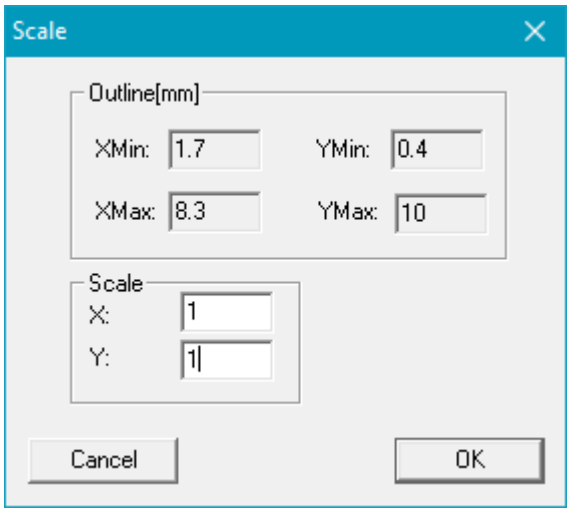


Figure 348: Transform Font Dialog

For version 1.0 Fonts a scaling factor of 10 leads to comparable results with version 2.0. The outline fields

show the maximal and minimal values of the font in both directions. The outlines should not exceed 10 mm in both directions. Otherwise you can not calculate with a 10 mm font height.

Spacing: The dialog spacing allows the definition of global calculation parameters for AW and LSB. See also the chapter "Generate Fonts".

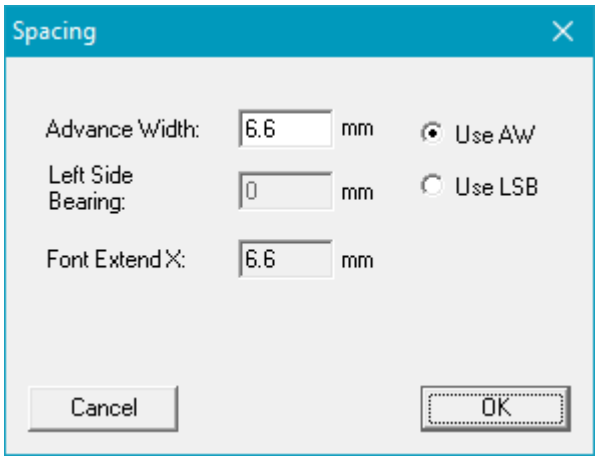


Figure 349: Font Spacing Dialog

UseAW: The AW parameter will be kept constantly. LSB and RSB will calculated to be equal.

AW = LSB + RSB +(XMAX-XMIN) (LSB==RSB==SB)

SB=(AW-(XMAX-XMIN))/2

This setting leads to monospaced fonts.

Use LSB: The LSB parameter will be kept constantly. LSB and RSB will calculated to be equally.

AW = LSB + RSB +(XMAX-XMIN) (LSB==RSB==SB)

AW=2*SB+(XMAX-XMIN)

This setting leads to variable spaced fonts. The FontExtend X field shows the X-Dimension (XMAX-XMIN) of the largest character in x-direction inside the font.

Baseline:

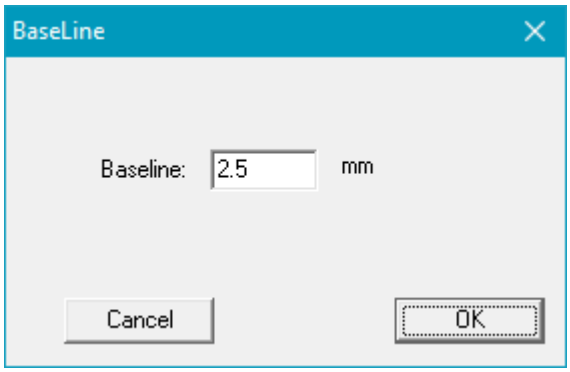


Figure 350: Font Baseline Dialog

With this dialog the user may define the base line of the font. The default value is 2.5 mm.

Edit View: The EditView allows the editing of the glyph polygons and the AW and LSB parameters for this glyph. You can reach this dialog by double-clicking the corresponding character cell. You can also import a HPGL,SAF or DXF file and attach it to the character. The outline of the character, the AW and LSB should not exceed the 10 mm outline box. The base line cannot be changed because this is a global parameter and applies to all characters in a font. Look to the ScView2DCtrl documentation for more details for the editing functionality.

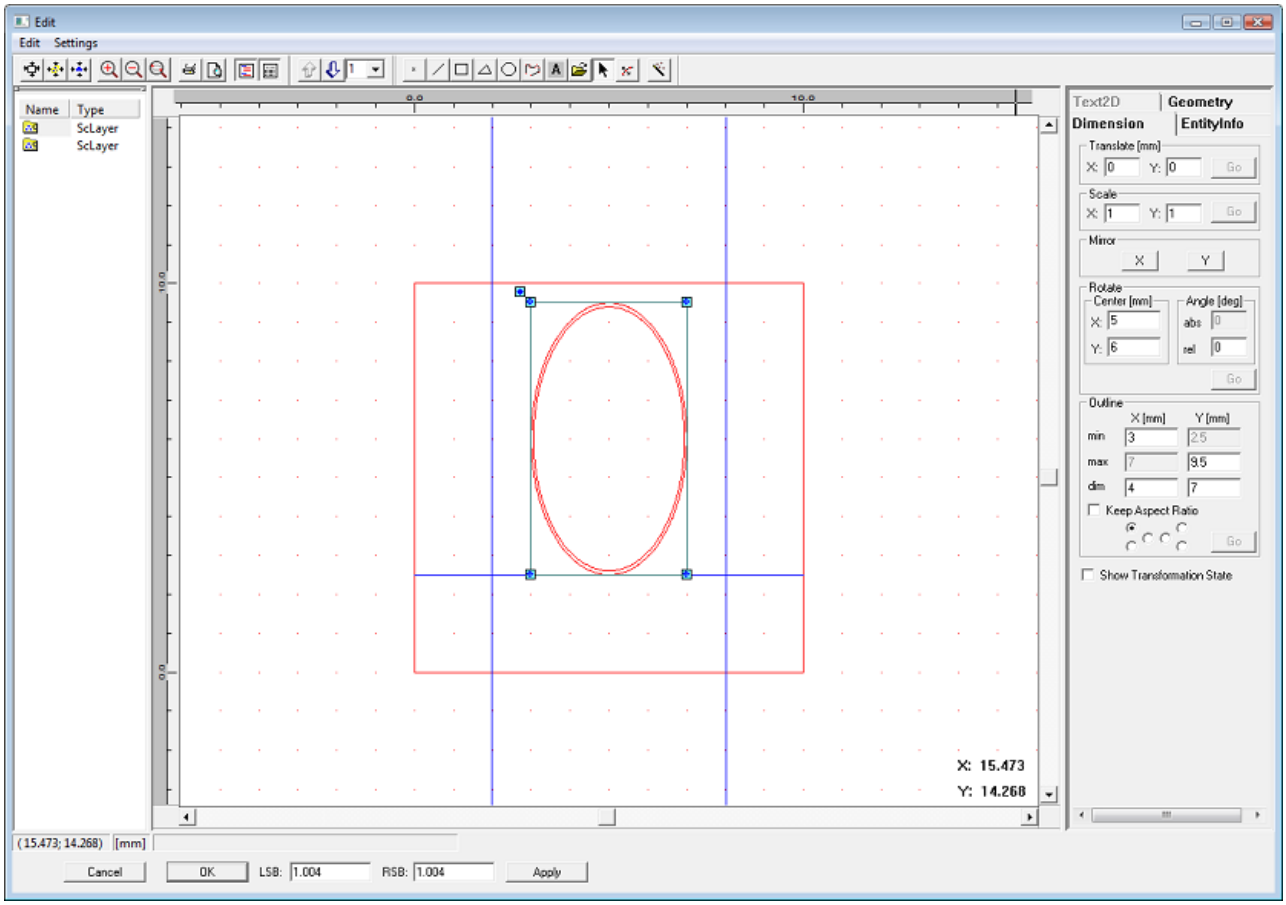


Figure 351: Font Edit View

Convert:

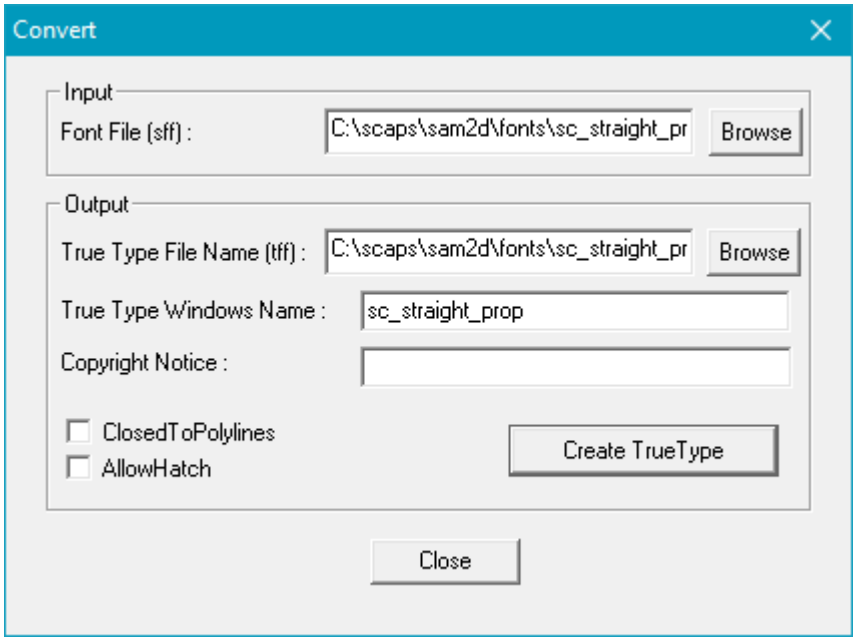


Figure 352: Font Convert Dialog

Parameters:

SCAPS Font File (sff)	File that has to be converted into the True Type Format
True Type file Name (ttf)	True Type Font file that has to be generated
True type Windows Name	The name of the Font
Copyright Notice	Company name of the font designer
ClosedToPolyLines	The True Type font has closed PolyLines. The WinText2D entity generation process creates closed PolyLines out of the character outlines whenever it is possible.

Closed PolyLines will be checked regarding their orientation before they are stored to the True Type Font file.

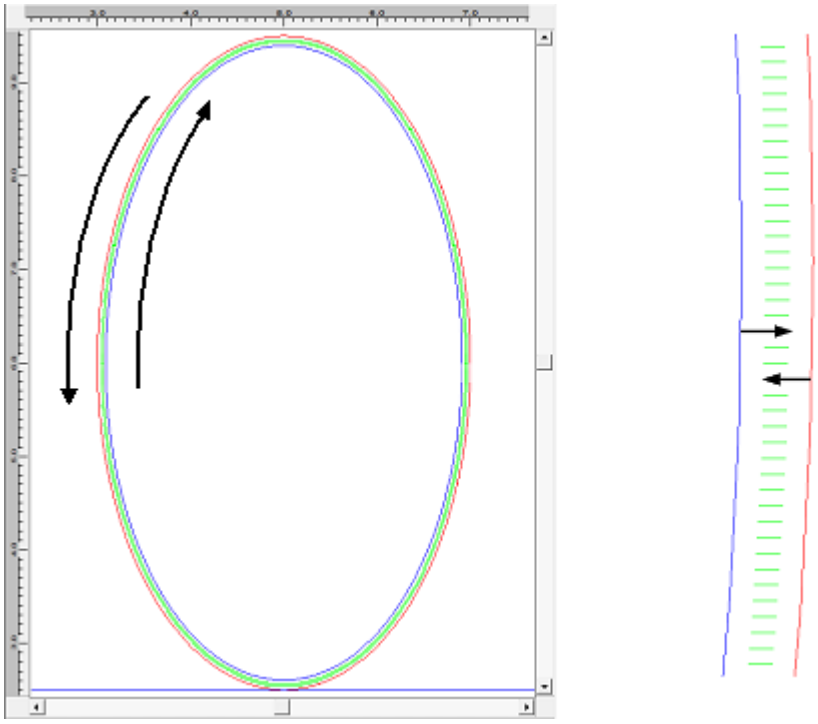


Figure 353: PolyLine Orientation Example

Outer PolyLines are oriented counterclockwise, inner PolyLines in the opposite. This is necessary to allow the Hatch beam compensation algorithm to determine the direction of the compensation as shown above for the letter '0'.

After filling in the parameters and clicking "Create True Type" button the font has to be installed by copying the file into the directory WINDOWS\FONTS. Now the font can be used in SAM2D for the generation of Single Line Characters. You may also have a preview of the font by double clicking on the *.ttf file inside Windows Explorer. Please make sure that a font with the same name is not already installed. In this case Windows will always take the installed font for displaying and you can not see the selected *.ttf file.

The program does not save the current loaded sff file. If you want to convert the current file you have to save it first.

Adding own property pages:

The font converter allows to add own property pages to the Edit View. The ASCII file sc_font_convert.prp located inside the SAM system folder can be manipulated. This works in the similar way as adding a property page by calling of the ScEntityPropertySheet.ScAddPage(Name) command. By default the following pages are added:

```
SCAPS.ScEntity InfoPropertyCtrl
SCAPS.ScDimensionPropertyCtrl
```

```
SCAPS.ScGeometryPropertyCtrl
SCAPS.ScText2DPropertyCtrl
```

To add a user property page the following line may be added:

```
SCAPS.ScUserPropertyCtrl.UserPropertyPageName
```

where UserPropertyPageName stands for a valid property page registered on the system.

20.2 Command Line Parameters

The scanner application can be started using different command line parameters. These parameters control its behavior in different ways. The following values are supported:

command line parameter	description
/JobEditor	SAMLight is started in JobEditor mode. Hardware output is not possible in this mode.
/SettingsFile=<*.sam>	With this parameter it is possible to define the settings file the software is using. The settings file is always stored in the folder <SCAPS>\system\.
/ActiveCard=<0, 1, …, 5>	If multiple scanner cards are connected to the PC, this command line parameter allows to select a card to be used with SAMLight. The number of the card is zero based and the valid range is from 0 to 5. Use the command line parameter /SettingsFile=<*.sam> as well, to chose a settings file. A single card settings file is required.
/StartupDelay=<sec>	This parameter delays the startup of the application for the given time period sec (in unit seconds). This may be necessary if it is started automatically out of the autostart-folder. Here it may happen that the application is executed from Windows before all necessary drivers are loaded. Using such a delay it is made sure that the application does not try to access the scanner card before it is made available by the operation system. Using this parameter the splash screen appears with no delay so that the user is informed that everything goes well.
/LoadJob=<path*.sjf>	Using this parameter a job defined by the path can be loaded during startup automatically. Please note that this option can be overwritten by the appropriate settings within the scanner application where you can define a job for loading on startup too.
/TriggerMode=<0, 1>	This option can be used only if a job was selected for loading using the preceding parameter. If the TriggerMode is set to 1 the application switches to trigger mode automatically after loading that job.
/3D	The application starts in 3D mode if this option is specified and if the appropriate license is available. This option does not overwrite the appropriate settings. If you want to use this option to toggle the program execution mode the auto save option of the general settings should be turned off, else the temporarily enabled 3D mode would be saved.
/Hidden	SAMLight starts invisible in the background.
/DisableMessageBeep	Will supress the Beep after some SAMLight actions (e.g when marking is finished)
/DisableHomingStopButton	The STOP button during a homing procedure on startup of SAMLight can be disabled.
/MarkTriggerUpdateBeforeEnd	Marking and updating will be done simultaneously, so that there will be no delay after the end of the Marking. (only available for Mark→ Trigger)
/UseDirectWrite	Use DirectWrite API instead of Uniscribe API for text layout and glyph rendering.
/StepperMotionFile	For motion type 8 or 14 a specific stepper configuration file 'StepperMotionFile' can be used instead of sc_motion_stepper_settings.txt. The motion type (8 or 14) has to be defined in sc_motion_settings.txt. The file 'StepperMotionFile' need to be located in the directory <SCAPS>\system\.

Table 36: Available command line parameters for SAMLight

Usage Example: This example describes how to create two icons on the Windows desktop, one starting the scanner application using card number 0 in YAG mode, the other starting it using card number 1 in CO2 mode.

Install the two cards: First the two cards with the drivers have to be installed properly.

Create two settings files: Within windows explorer, go into folder <SCAPS>\system\ and make two copies of the existing file sc_light_settings.sam. Rename them to sc_light_settings_yag.sam and sc_light_settings_co2.sam.

Setup: Start <SCAPS>\tools\sc_setup.exe, go to menu HardwareSettings, select the file sc_light_settings_yag.sam, press "Load" and set up the card for YAG mode. Save the settings and repeat this step for the CO2 file.

Create Windows shortcuts: Create two shortcuts of sam_light.exe. Change the name of the shortcuts to "sam_light YAG" and "sam_light CO2". Right click a shortcut and go to Properties.

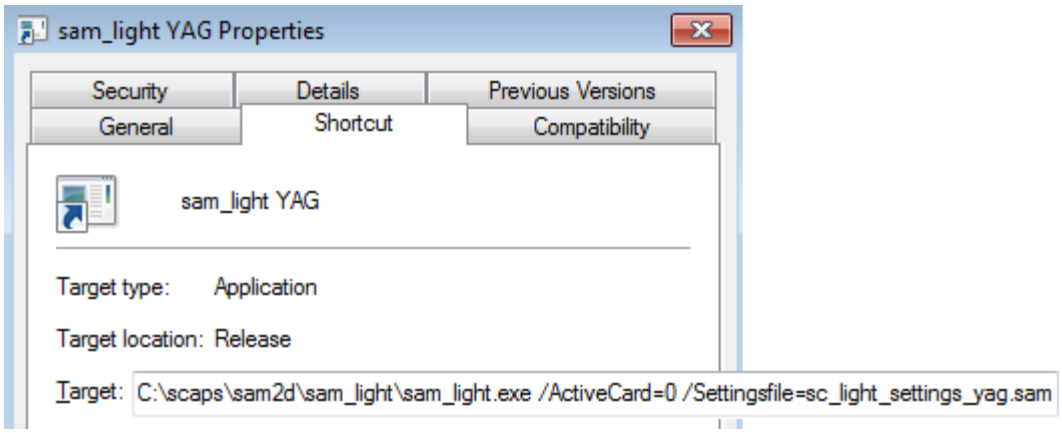


Figure 354: Creating a Shortcut to SAMLight with command line parameters

Set the command line paramters: Inside the property page of the shortcut, define the program arguments. In this example we add the following command line parameters:

```
/ActiveCard=0 /Settingsfile=sc_light_settings_yag.sam
```

Other settings file: Apply for the "sam_light CO2" shortcut the following command line parameters:

```
/ActiveCard=1 /Settingsfile=sc_light_settings_co2.sam
```

The entire string in "Target" should be: C:\scaps\sam2d\samlight\sam_light.exe /ActiveCard=1 / Settingsfile=sc_light_settings_co2.sam

20.3 Customize Program / Language

This chapter might be helpful, if you want to define your own outfit of the application by creating an own application title, a bitmap with your logo and an icon as an application identifier. Another feature explained here is the change of the strings on the windows to provide different languages for the user interface.

20.3.1 Personalize Program

All files to personalize the appearance are stored in the folder <*SCAPS*>*system*\. The following files can be substituted by personal ones.

sc_light_icon.ico : This is the desktop icon.

sc_light_logo.bmp : This is the startup logo.

sc_light_name.txt : This is the name of the software.

20.3.1.1 Installation of User Data

To install special user data, create a directory with the name *data* in the same directory (of the installation medium), where the installer-exe-file is located. ??

Valid user data are: (*)

The following settings-, logo-, icon- and help-files:

- sc_light_icon.ico
- sc_light_logo.bmp
- sc_light_name.txt
- sc_light_settings.samsc_settings.sam
- sc_help_sl_english.chm
- sc_resource_settings.sam

Correction files: These are all *.ucf files with corresponding descriptions as *.txt files. A description file called *filename.txt* will be installed only if there exists a correction file called *filename.ucf*, unless the text file is *sc_light_name.txt*.

Resource files: These are *.sam files, the names of which are beginning with *sc_resource*, e.g. *sc_resource_sc_german.sam.* Files in the data directory with other names are not valid and will not be installed.

Controlling the installation of user data: To control the installation of these files, create a text file called *sc_data_info.txt* in the data directory. This file contains a line for each file with the following information :

filename=flag : Filename can be one of the names in **(*)** and where flag can be one of the following values:

ow : Overwrite, the file on the target system will be overwritten with the corresponding file from the data directory

au : Ask user, if the file actually exists on the target system the installer will ask the user to overwrite or not, if the file does not exist on the target system the corresponding file from the data directory will be copied.

no : No overwrite, if the file actually exists on the target system it will not be overwritten, if the file does not exist on the target system the corresponding file from the data directory will be copied.

Files in the data directory which are not listed in *sc_data_info.txt* will be treated as if the no-flag was set. If *sc_data_info.txt* is empty or does not exist then all files in the data directory will be treated as if the no-flag was set.

Example for sc_data_info.txt:

```
sc_light_icon.ico=au
sc_help_sl_english.chm=ow
```

Do not type white spaces in front of the filename or in between filename and "=" or in between "=" and the flag.

Example: Installation of SamLight with a Chinese resource:

create a data directory containing the files:

- sc_resource_sc_chinese.sam
- sc_resource_settings.sam (referring to sc_resource_sc_chinese.sam)
- sc_data_info.txt

where *sc_data_info.txt* contains the lines:

```
sc_resource_sc_chinese.sam=ow
```

```
sc_resource_settings.sam=ow
```

20.3.1.2 Customize Laser Names

It is possible to change the Laser Name string in the Mark Property Page and the Pen window captions. Therefore a new text document "override_strings.txt" has to be created in the scaps system folder. Then the user must add LaserName = <Customer Name> to the text document. The entry <Customer Name> will be shown in the Mark Property Page and the Pen window captions.

20.3.2 Customize Language

The *ScResourceManager* allows to change nearly all strings used in the SAMLight. It is possible to redefine the appearance of dialog boxes as well as to change messages. With this powerful feature you can generate a SAMLight user interface for your own language and make working at the machine even easier.

20.3.2.1 Global Settings

To activate the Resource Edit mode the following steps are necessary:

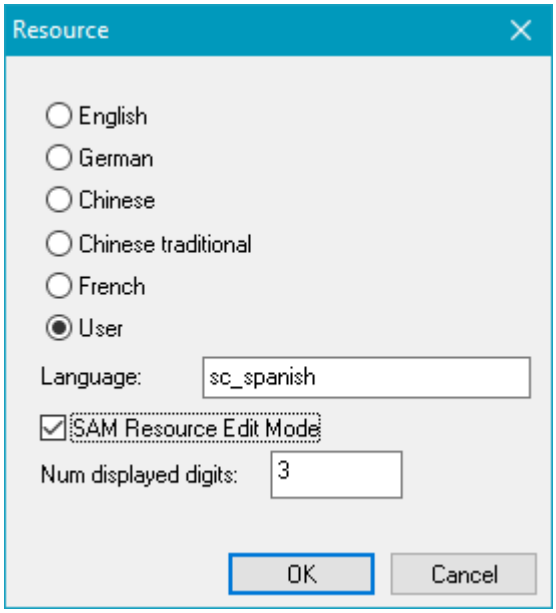


Figure 355: Resource Dialog

1. Close all SAM based applications, check it in the taskmanager.
2. Start *sc_setup.exe* from folder <*SCAPS*>*tools*\
3. Click on the menu *Resource*.
4. Select a default language or enter an *User* language.
5. If *User* is selected, *SAM Resource Edit Mode* is click able to enable resource edit mode.
6. Close *sc_setup.exe* and start SAMLight.
7. Do the translations for the user interface, as described in the chapter Resource Editor.
8. When all translation is done quit your application and switch off *SAMResourceEditMode* again.

When the SAM based application runs and the *SAMResourceEditMode* is enabled, some *EditResource* buttons in the property page of the *ScView2DCtrl* and in the dialogs of the *OpticModul* and the *ScOpticModuleCtrl* appear. These buttons are the entry to the *ResourceEditor*. The string in *Language* is used to generate the corresponding resource file. The file containing all user defined resources is located in<*SCAPS*>*system*\ and it is named *sc_resource_YOURLANGUAGE.sam*. To put your resources from one PC to another you just need to copy this file and to define the language string within *sc_setup.exe*.

20.3.2.2 Resource Editor

Clicking on the *EditResource* button within one dialog shows the the following window. The strings given in this dialog have to be translated to get a new language appearance.

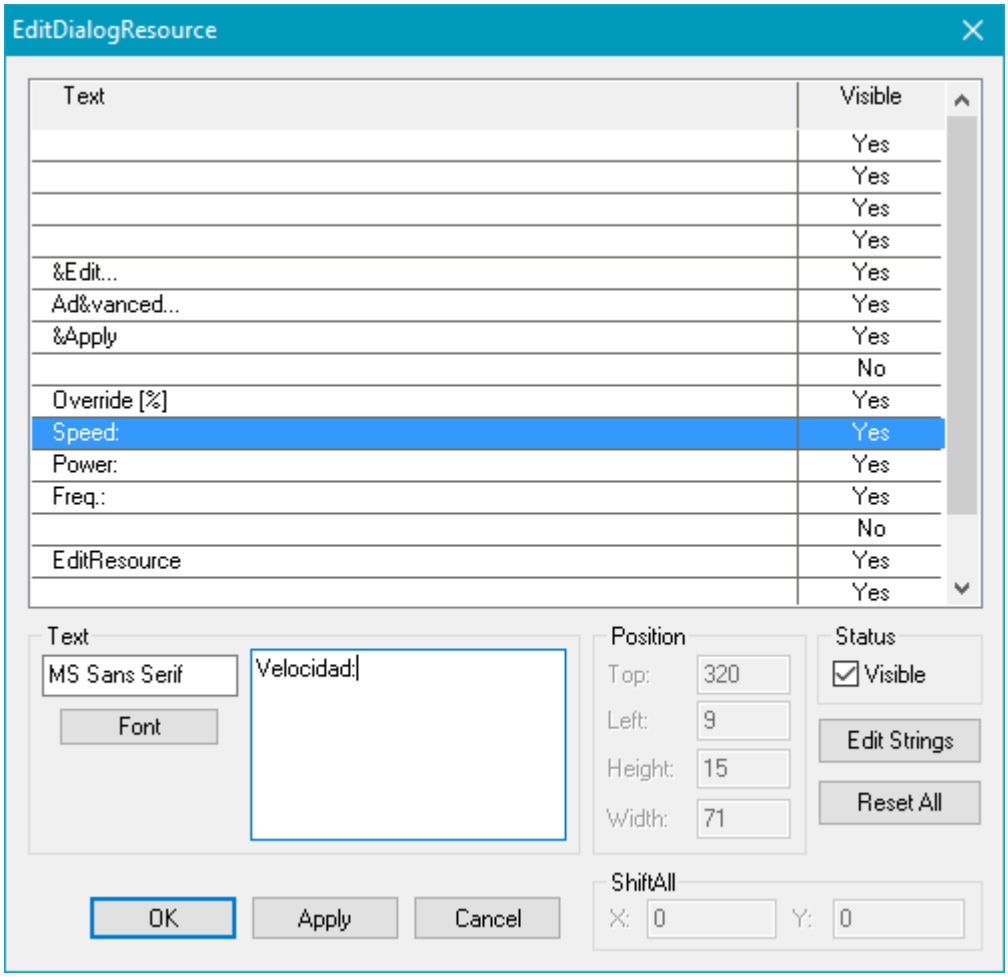


Figure 356: Resource Editor Dialog

List view: On the list view the user can select the string that will be edited by clicking on it. The list also shows strings of the contents of text boxes. These values do not have to be changed.

Text: The text selected in the view list appears in the *Text* window and can be changed there.

Font: With the button *Font* it is possible to define a font for the window string. Please make sure that the selected font is also available on the end user system.

Position: The size and the position of the selected window can be changed.

Status: If *Visible* is chosen the selected string will be visible for this language. The status is displayed in the list view.

Edit Strings: This button is the entry to the String Editor for all SAM modules on this PC. Some dialogs have dynamic string setting. For example the *GeometryPropertyPage* strings are set depending on the type of the selected entity. In these cases the strings have to be defined in the string editor. For example the *Rectangle* string is defined in string module *StandardProp, String ID 11*.

Reset All: Imports the default English resources. The current window texts are reset to default English after pressing the *OK* button.

ShiftAll: After pressing *Apply* all strings of the current window are shifted by the values given in X and Y.

20.3.2.2.1 String Editor

Clicking on the *EditStrings* button within the dialog of the *Resource Editor* shows the following dialog:

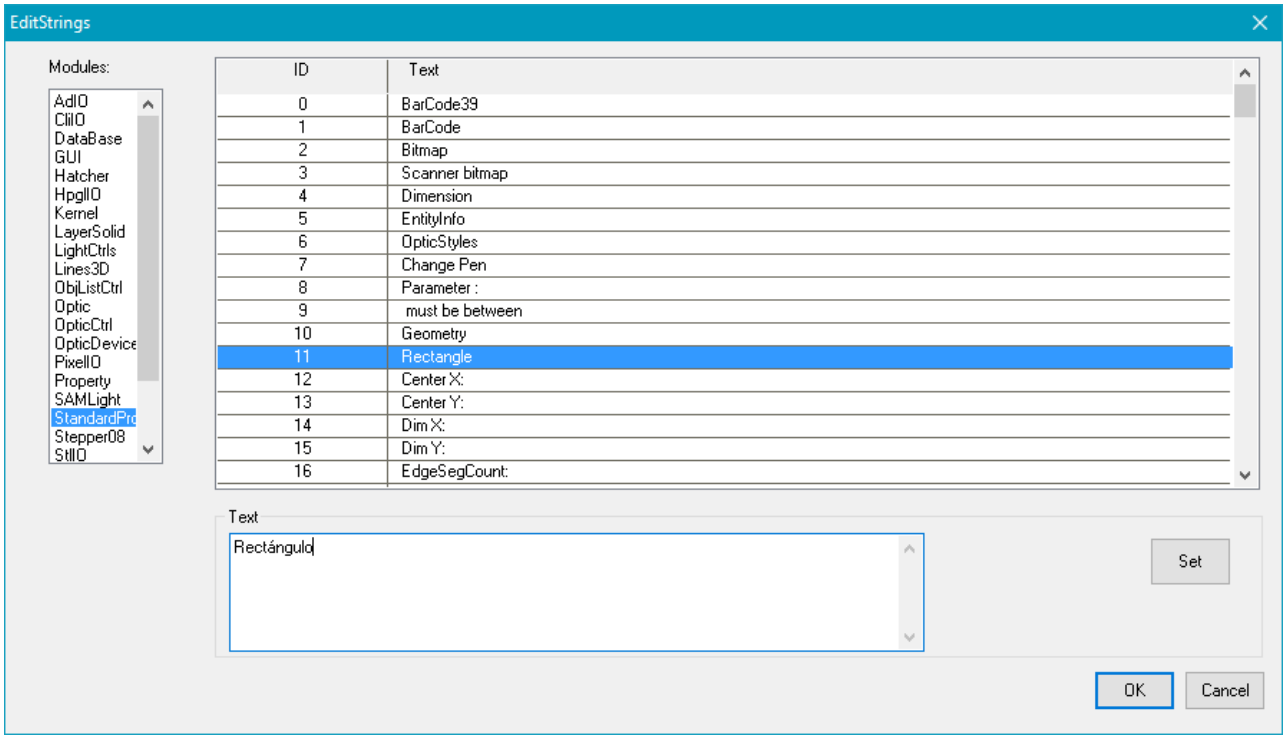


Figure 357: String Editor Dialog

Modules: On the left side there is a list with all activated SAM modules. Every module contains specific strings. To modify the strings it is necessary to select the module, then select the string and then change the text.

Set: Clicking *Set* modifies the corresponding string.

The most important strings are inside the Kernel, StandardProp, Optic, OpticCtrl and View2DCtrl modules. It is not possible to modify the menu for the Standard2D application, because this application is delivered with the source code and changes have to be done there. The SAMLight menu can be changed by selecting SAMLight and defining the desired strings. Also for SAMLight it is necessary to modify the LightCtrls string module. The changes of the String Editor are only visible after a restart of SAMLight.

20.4 Use camera as background image

For some applications it is helpful to have a camera image as background in the View 2D so you can place the entities to be marked in the right position and the background image will not be marked. A common way to do this is to use the Client Control function calls of SAMLight to automate the whole procedure. Once you can get a bitmap from the camera device you can use this as a background image.

You can use the SAMLight Client Control to automate the View 2D background image and you can choose between two options.

1st option:

1. Save the camera image as a bitmap (*.bmp) somewhere on the harddisk.

2. Use the following excerpt of C# source code to blend in a bitmap as View 2D background without creating an entity in SAMLight:

```
...
using SAMLIGHT_CLIENT_CTRL_OCXLib;
...

axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendCenterPointX,
0 );
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendCenterPointY,
0 );
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendBmpDimX, 300 );
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendBmpDimY, 300 );
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeBmpAlphaBlendPathBmp, "C:\
\TestBitmap.bmp" );
axScSamlightClientCtrl1.ScSetLongValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValueTypeBmpAlphaBlendSourceConstantAl
pha, 128 );
```

This code sample assumes that the path of the bitmap file is C:\TestBitmap.bmp and assumes an X and Y dimension of 300 pixels each. For the other parameters please refer to the manual chapter Client_Control_Interface. There you will find the explanation for the used Client Control calls.

3. Once a bitmap has been loaded into the View2D it can always be updated by the single command:

```
axScSamlightClientCtrl1.ScSetStringValue( ( int )
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlStringValueTypeBmpAlphaBlendPathBmp, "C:\
\TestBitmap.bmp" );
```

2nd option:

1. Save the camera image as a bitmap (*.bmp) somewhere on the harddisk (for example C:\TestBitmap.bmp). If the X, Y dimension or position of this bitmap does not fit it can be adjusted in Bitmap → Extended dialog of the background image. This sample bitmap has to have an unique entity name. For example "BackgroundImage" which can be set in the Entity Info property page.

2. The camera should now update the bitmap (for example C:\TestBitmap.bmp).

3. Use the following excerpt of C# source code to automate the re-import of the bitmap.

```
axScSamlightClientCtrl1.ScImport( "BackgroundImage", "C:\\TestBitmap.bmp", "bmp", 1, ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlImportFlagBitmapReimport );
axScSamlightClientCtrl1.ScSetEntityLongData( "BackgroundImage", ( int )
ScComSAMLightClientCtrlFlags.scComSAMLightClientCtrlLongDataIdEntitySetAsBackgroundEntity, 1 );
```

For further details please refer to the SAMLight manual chapter Client_Control_Interface.

20.5 Accelerate SAMLight

For many applications it is important to accelerate the execution speed of SAMLight. The following SAMLight settings show how this can be done:

1st option: Enable the following check boxes at SAMLight menu bar Settings → System → General :

Disable UNDO: speeds up SAMLight for all entity generating and editing processes as the UNDO and REDO functionality is disabled completely. Depending on the size and complexity of a job this option is able to save namable amounts of memory and calculation time.

Disable Compression: can help to save big job files on computer systems with low main memory.

Don't update view: can help to save processing time, when entities are updates (date/time and serial number entities, reimport of bitmaps, ...)

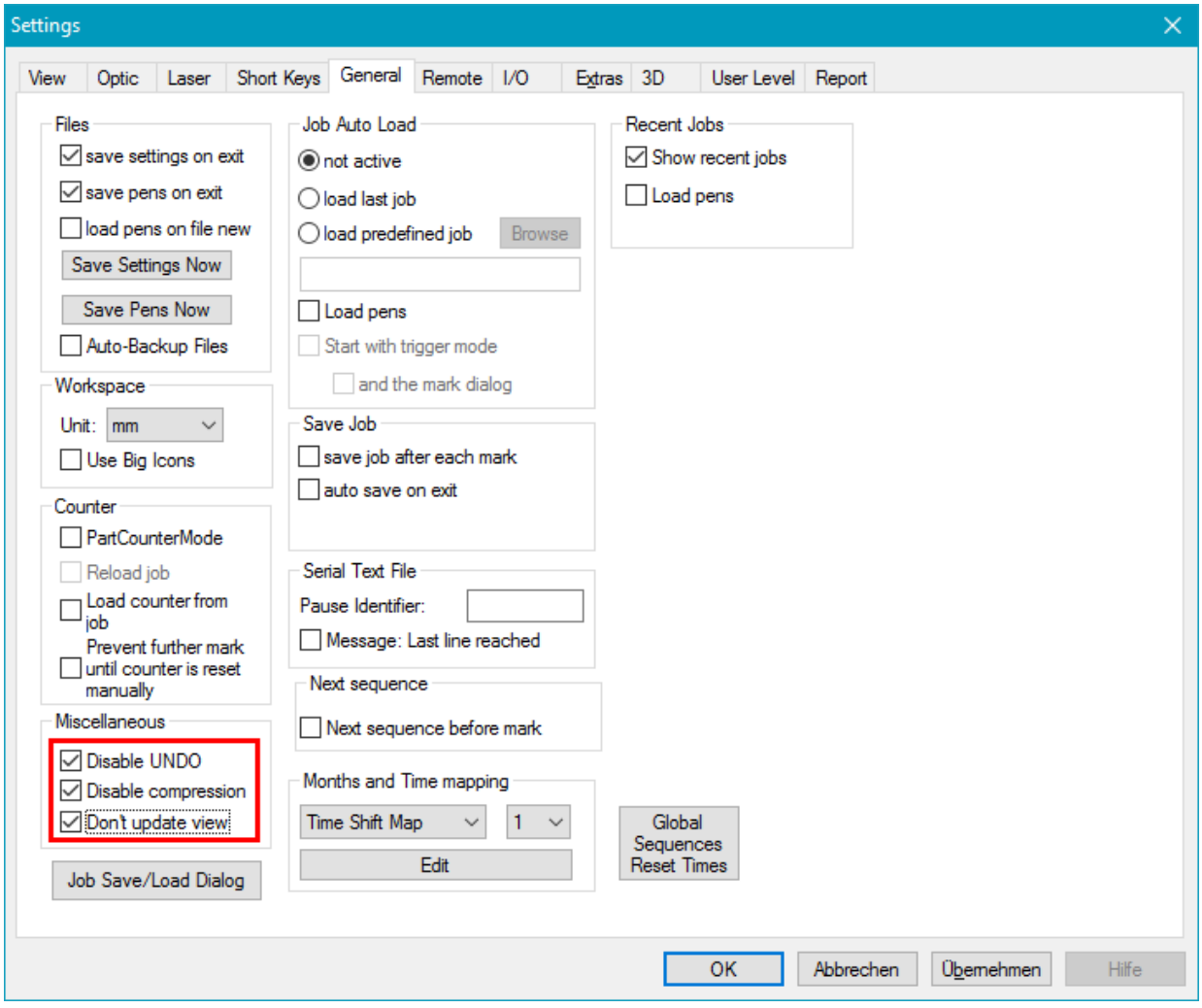


Figure 358: check boxes at General settings for accelerating SAMLight

2nd option: In case the used job file contains text entities like Text2D, serial number, date/time or bar code entities with text the following check box at Text2D property page → Extended settings should be deactivated.

Generate single characters: deactivating this check box can accelerate the generation or all kind of text entities, especially if many text entities are used in a job file.

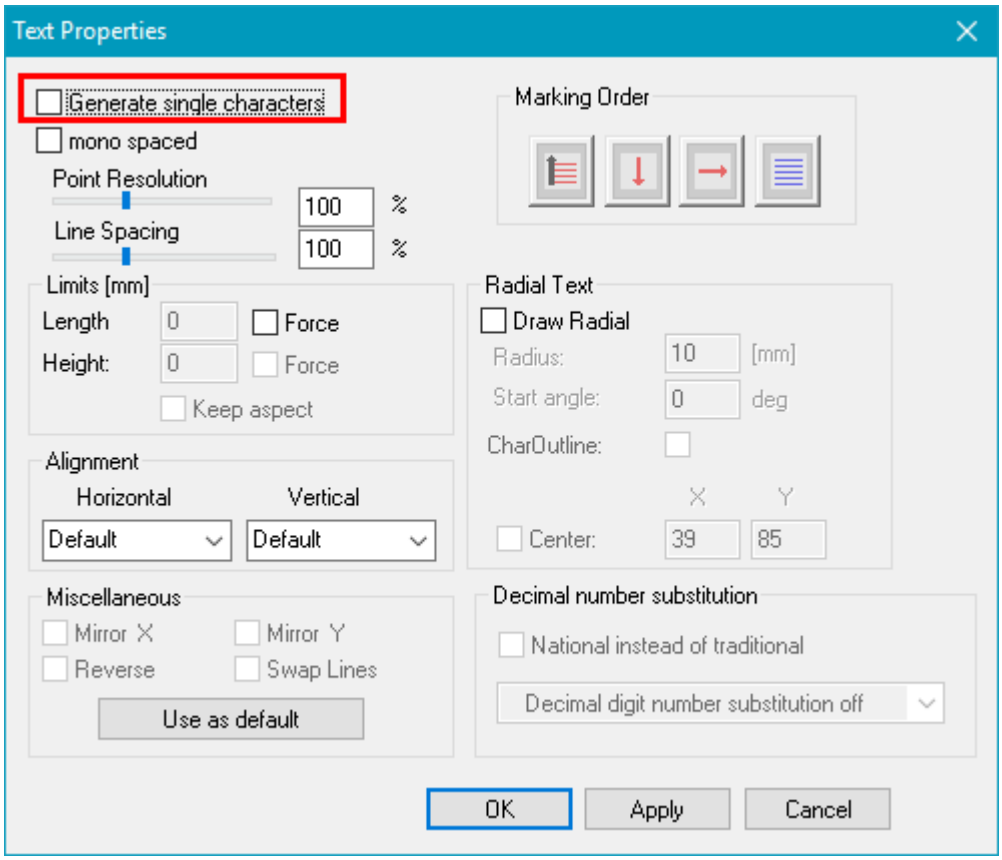


Figure 359: check boxes at Text2D property page Extended settings for accelerating text generation of SAMLight

3rd option: In case SAMLight Client Control programming is used for automating and remote controlling of SAMLight, the SAMLight Client Control commands and constants described at Optimize Performance can be used.

20.6 Generate Dots

For marking a datamatrix with a design of dots, the following can be done:

1. Use DataMatrixEx and activate "Generate Dots" in the Extended Dialog.

2. Marking of any other datamatrix barcode

   a. Create a datamatrix
   b. Calculate the size of a single module of the datamatrix
   c. Hatch the datamatrix with the parameters (Hatch property page)
      i. Distance = size of one single module
      ii. Start Offset = half of the size of the single module
      iii. Linereduction = very close to the start offset
   d. Enable "Mark Lines as Dots" and set the grid raster values to the module size (Pen property page - Drill)
   e. Enable "Use Geometry" and apply a hatched circle with a size smaller than the module (Pen property page - Drill)
   f. Disable "Contour" for marking only the dots of the datamatrix (Pen property page - Misc)

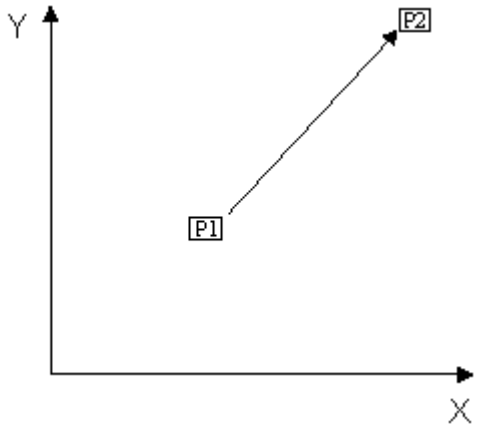
21 Backgrounds

In this chapter miscellaneous theoretical explanations are given for scanner card and program specific functioning.

21.1 Scanner and Laser delays

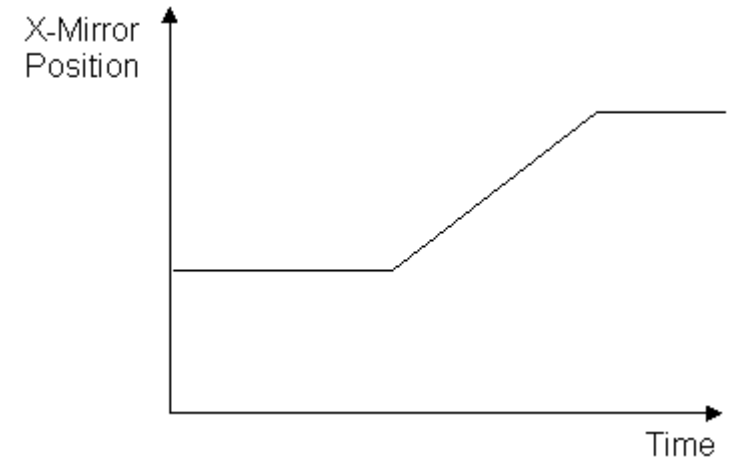
The scanner and laser delays are defined in the laser style parameters dialogs. This chapter gives a short explanation of the delay terms.

Assume that the XY Mirror system is commanded to go from P1 to P2 in XY-plane with a desired speed v.

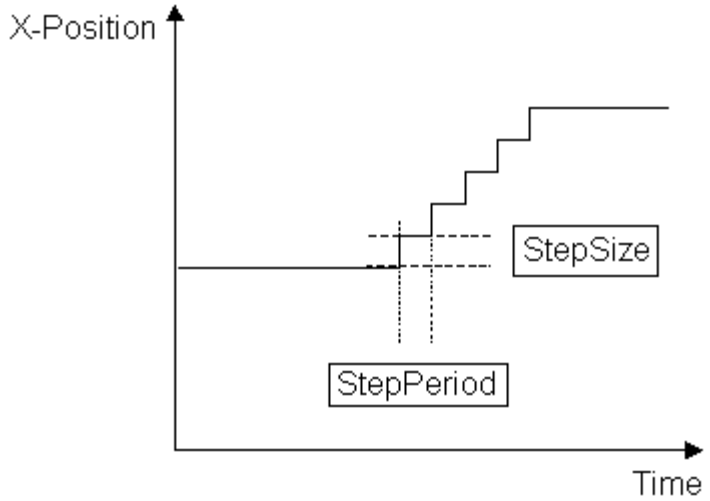


In the following only the X-Mirror is covered, the Y-Mirror is completely analogue.

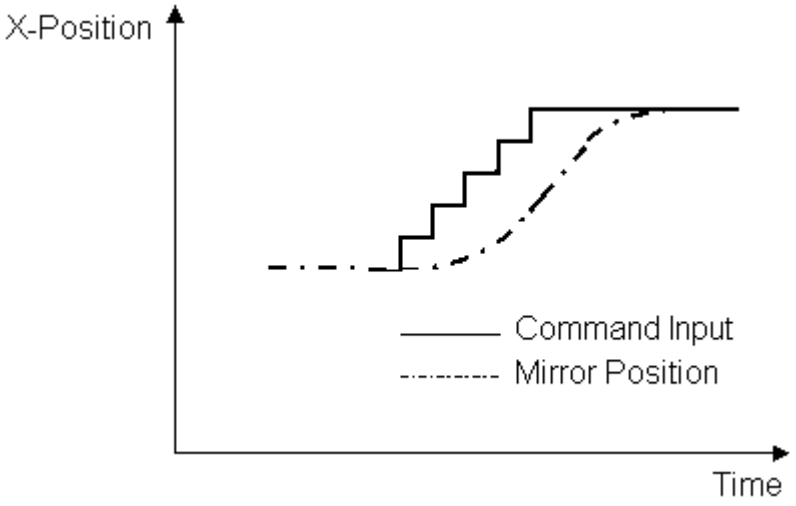
The move command for the X-Mirror looks as follows:



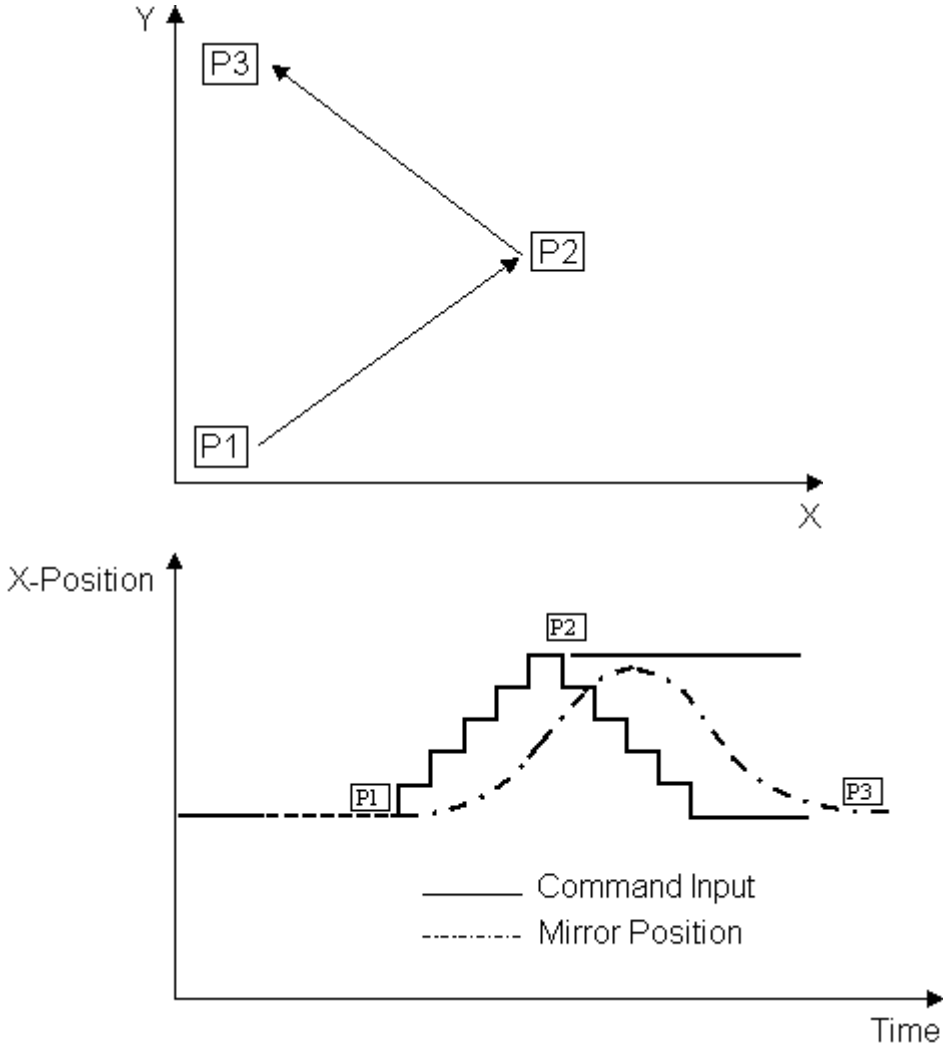
Since the controller card is not able to output values in an arbitrary short time period it has to approximate the desired curve in so called 'microsteps' with a time length of StepPeriod – typically 10 to 50 µs and a position change of StepSize. The requested speed v is the quotient from StepSize/StepPeriod.



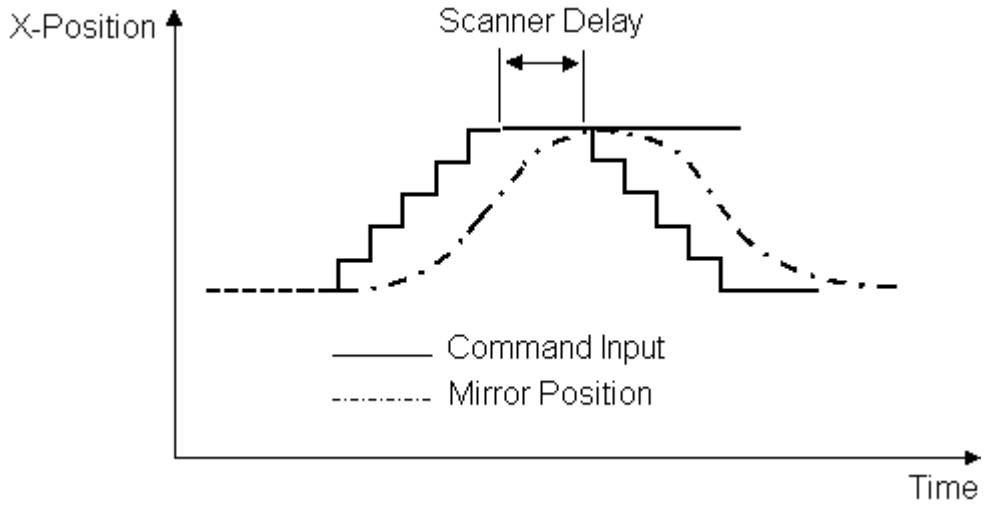
Since the X-Scanner with attached mirror is an inert system it can not follow the controller commands in short time but has some time lag.



Due to this fact a command input like shown below would lead to the result that the X-Mirror would never reach position P2.



For this reason the controller card inserts a user definable delay between the end of the last vector and the start of the new vector.

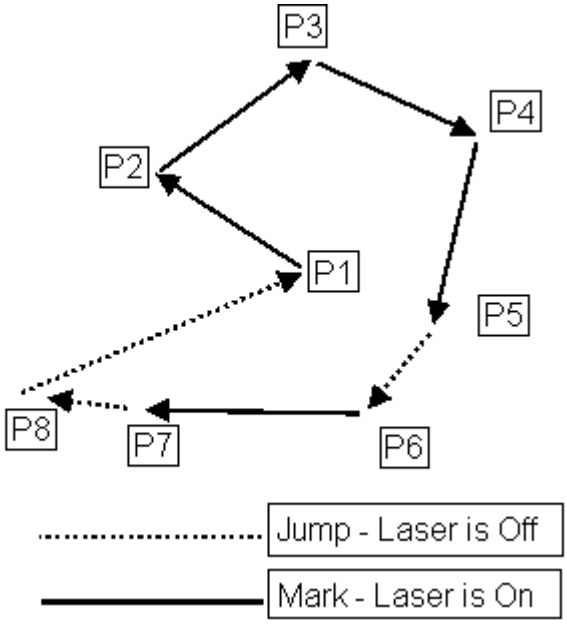


There are 3 kinds of scanner delays:

Delay	When are the delays used?
Jump	Points P1,P6 and P8 in the picture below.

Mark	Points P5 and P7 in the picture below.
Poly	Points P2,P3 and P4 in the picture below.

Table 37: Example of scanner delays

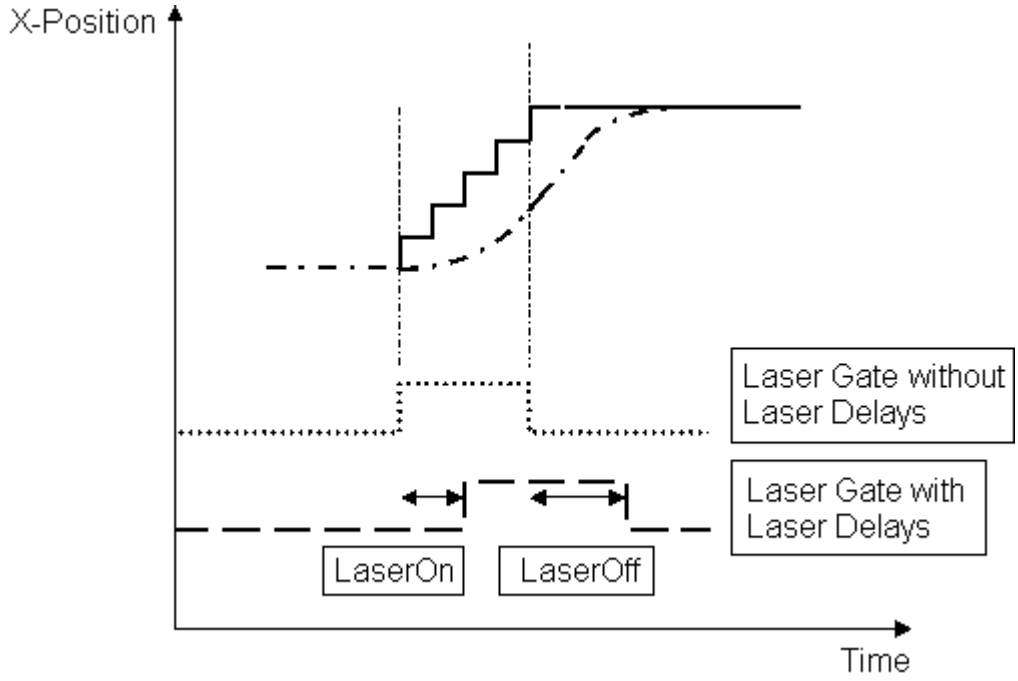


Since the X-Mirror can not accelerate in an arbitrary short time to the requested speed or deaccelerate to zero speed, two additional delays are used to control the delayed switch on and off of the laser gate signal. These are the laser on and the laser off delay.

LaserON delay: Time beginning from the output of the first microstep the controller card waits before it switches on the laser.

LaserOFF delay: Time beginning from the output of the last microstep the controller card waits before it switches off the laser.

Example for laser delays:



21.2 USC Position Transformation

Depending on the scanner controller card and marking mode (SAMLight or standalone) the coordinates stored in SAMLight job file (*.sjf) or standalone job file (*.unf) are transformed like shown in table 38.

	Vector transformation	USC-1 SAMLight	USC-2 SAMLight	USC-2 Flash [OF]	USC-3 SAMLight	USC-3 Flash [OF]	Settings paths and flash notes
PC		SJF	SJF	SJF	SJF	SJF	**SAMLight:** Coordinates saved in SAMLight job file (*.sjf)
PC	CCI ScOpticMatrix	↓	↓	✗	↓	✗	**SAMLight Client Control Interface (CCI) ScOpticMatrix,** adjustable by CCI commands ScOpticMatrixTranslate, ScOpticMatrixRotate, ScOpticMatrixScale and ScOpticMatrixReset
PC	3D Surface mapping	↓	↓	✗	↓	↓	3D surface **toolbar**
PC	SAMLight lens settings	↓	↓	↓	↓	↓	**Settings** → **Optic.** Order: - XY field center, XY gain, rotation, XY offset (with gain), XY invert, XY flip **sc_setup** → z-Offset
PC	PC Z bit value calculation [O3]	↓	↓	✗	✗	✗	**Settings** → **Optic** → **Advanced** → **Correction** → Z Correction **Pen Settings** → **Misc** → Defocus [mm]
USC				UNF		UNF	**Flash:** Coordinates saved in flash job file (*.unf). SAMLight lens settings are used for UNF generation.
USC	Flash global optic matrix [OF]	✗	✗	↓	✗	↓	**Flash:** Standalone only, not used in SAMLight, not accessible via SAMLight. Available Flash Control Interface commands: TMB, SC, RT, TRB
USC	XY head specific optic matrix	✗	↓	↓	↓	↓	**Settings** → **Optic** → **Advanced** → Correction **Flash:** Available Flash Control Interface commands: TRH (InfoView)
USC	Z head specific optic matrix	✗	✗	✗	↓	↓	**Settings** → **Optic** → **Advanced** → Correction **Flash:** Available Flash Control Interface commands: TRH (InfoView)
USC	MOTF [OM] AnalogIn [AI] Wobble	↓	↓	↓	↓	↓	**Settings** → **Optic** → **Advanced** → Marking on the Fly **Settings** → **Optic** → **Advanced** → Analog In **Pen settings** → **Scanner** → Wobble
USC	USC Z bit value calculation [O3]	✗	✗	✗	↓	↓	**Settings** → **Optic** → **Advanced** → **Correction** → Z Correction **Pen Settings** → **Misc** → Defocus [mm]
USC	correction file (*.ucf) mapping	↓	↓	↓	↓	↓	**Settings** → **Optic** → **Advanced** → Correction **Z value** of *.ucf: please refer to [ZV]
USC	FlatLense Defocus [3FL]	✗	↓	✗	↓	✗	**Pen Settings** → **Misc** → Defocus **[2^{16} bit / FieldSize]** Add a static value on z output
		XY2-100 signals to scan head					

Table 38: Vector transformations

[OF]: Option Flash required.

[O3]: Option Optic3D required. Optic3D is not possible with USC-2 and standalone mode.

[OM]: Option MOTF required.

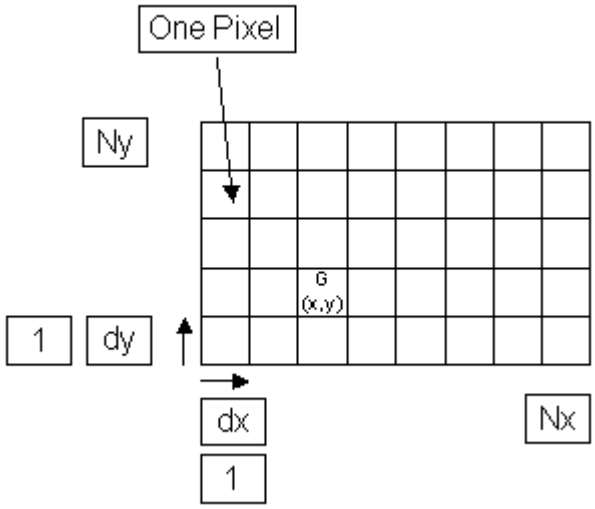
[AI]: AnalogIn is only available for USC-2 / USC-3.

[ZV]: Option FlatLense: Z bit values from UCF file are used. Option Optic3D (+ FlatLense): Z bit values from UCF file are ignored.

[3FL]: Option FlatLense required, if Optic3D license is used the defocus value is included in Z bit value calculation.

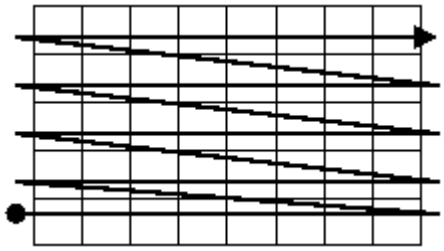
21.3 Pixelmode

This chapter describes how the scanning of bitmaps works with the USC-1 and the RTC3 scanner card. The USC-1and RTC3 card provide a special mode for raster images (Bitmaps).



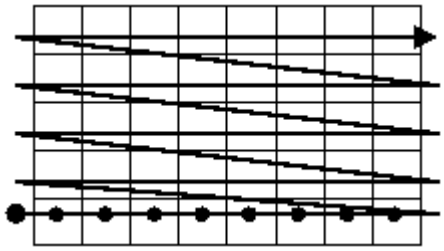
Each pixel inside a bitmap has the same X and Y dimension dx,dy. dx and dy itself may be different. The bitmap consists of Nx pixels in X direction and Ny pixels in Y direction. Each pixel has a Grayvalue G (x,y) from *0* to *1* which is typically transformed to a Grayvalue range from *0* to *255*.

The USC-1 and RTC3 raster modes allow to move the scanner across the bitmap by simultaneously modulating the laser control signal. Within this chapter it is assumed that the scanner movement is performed like shown below.



The scanner starts at the lower left corner, moves over the first line (X-direction) with a defined speed, jumps back to the start of the second line and so on.

Special mode for RTC3 and RTC4 card: If the hardware mode is selected the RTC3 provides two different modes for the movement of the scanner itself. In Mode 0 (shown below for the first line) every pixel position is reached within one scanner step command.



Then the scanner stays at the pixel a certain time before it jumps to the next location. In Mode 1 the scanner moves over the pixels with a constant speed applying many microsteps between the single pixels. In the following scanner mode 1 (constant speed over the pixel line) is assumed.

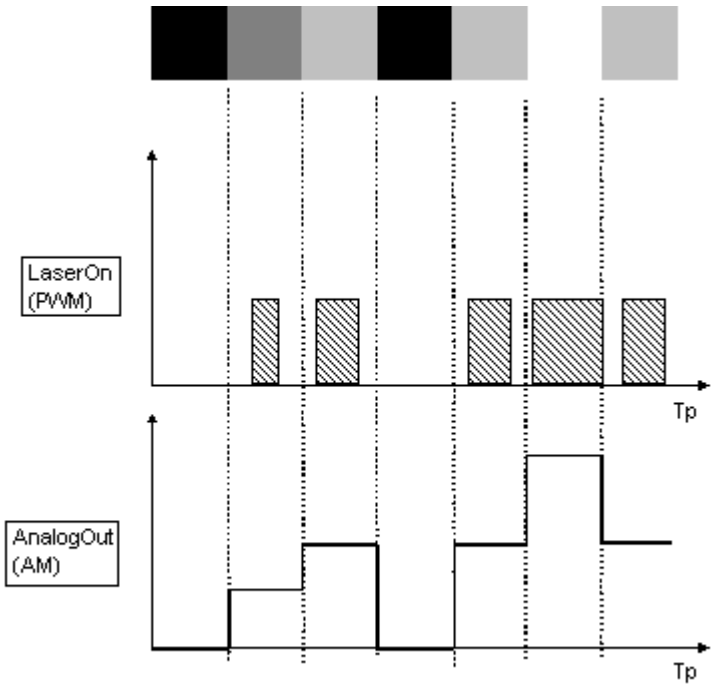
21.3.1 Pulse Modulation

Two modes are provided by the USC-1 and RTC3 to modulate the laser. In general terms they can be described as:

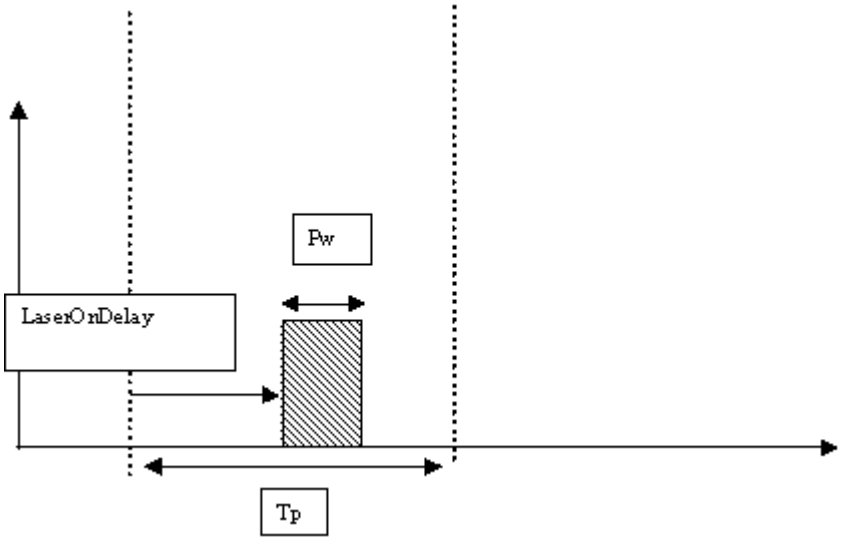
a) Pulse Width Modulation (PWM)
b) Amplitude modulation (AM)

In PWM mode the LaserON Signal is modulated. In AM mode the analog output value of the Laserport is modulated. For a given speed V the time for one pixel is calculated by:

Tp = dx / V



For the scanner card, Tp are multiples of 10 µs. For PWM the pulse with PW inside Tp is calculated according the following formula:



Pw = (Tp - LON - LOFF) * GrayScaleValue

GrayScaleValue	defined from 0 to 1 (normally in 1/256 Steps)
LON	LaserOnDelay
LOFF	LaserOffDelay

LON has the special effect that it offsets the start of the pulse within Tp.

Amplitude Modulation (AM)

For AM the GrayScaleValue will be transformed linearly to the analog output value.

Analogoutput = GrayScaleValue * MaxOutput

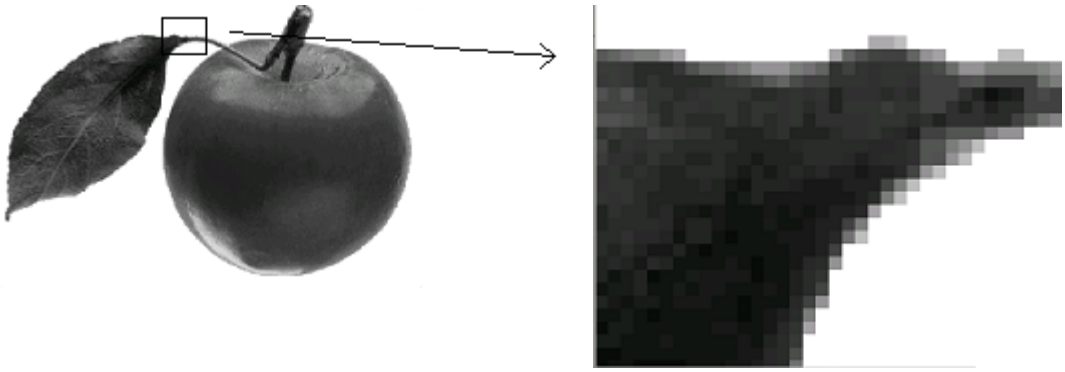
MaxOutput corresponds to maximal achievable Output voltage of the Digital to Analog Converter Output.

If one of the following conditions is valid no output takes place:

- **Tp < 10 OR Tp > 655350**
- **(Tp - LON - LOFF) < = 0**
- **dx == 0 AND dy == 0**

21.3.2 Generating a scanner bitmap

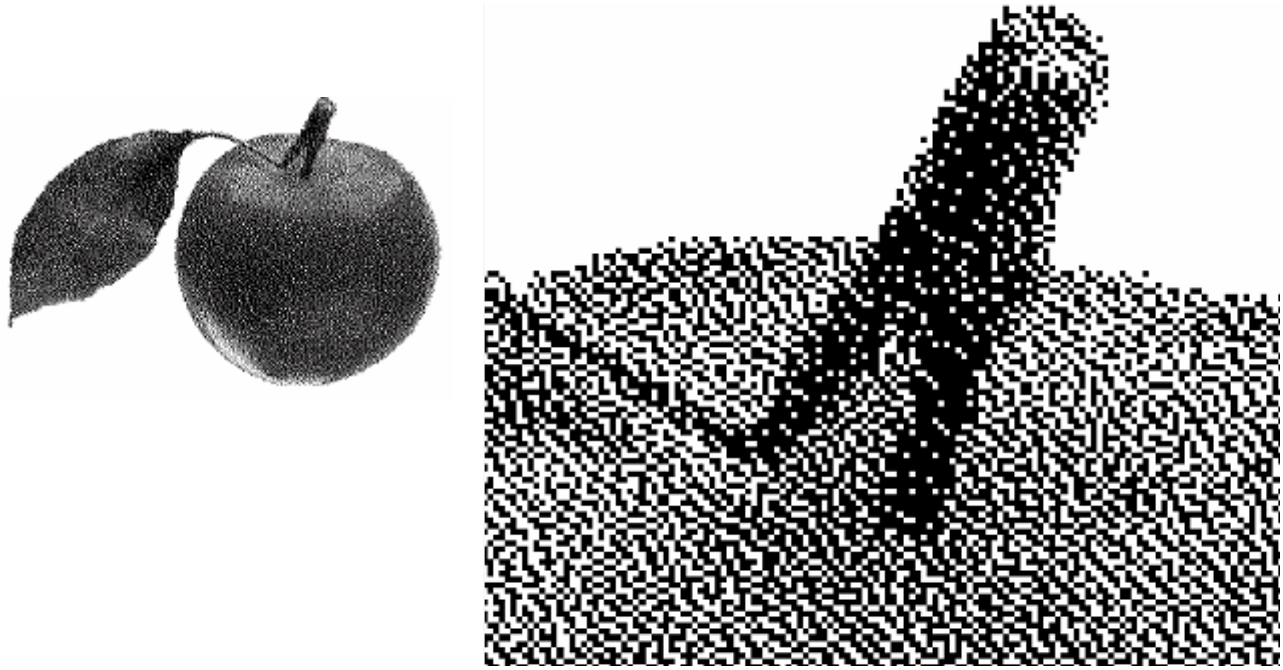
In the following the parameter setting with SAM for the apple.bmp is shown.



The apple.bmp consists of a gray scale bitmap with 255 different possible gray scale values G (x,y). After loading the bitmap the user may scale it according to his needs. Depending on the X and Y dimension and Nx, Ny the bitmap will have a specific dx,dy value for each pixel. To prepare the bitmap for the scanner output the user has to generate a so called scanner bitmap out of the original bitmap. This is done with the help of the Bitmap property page.

For an original Gray Scale bitmap there are two possibilities to generate a scanner bitmap:

a) Error diffusion method:



With this method the gray scale values will be approximated by specific placements of black and white pixels to give the impression of gray values. This is a similar method as with a Black/White LaserJet.

b) GrayScale method:

With this method the gray scale values will be kept when transforming them to the scanner bitmap.

Dither step: For both methods a so called *Dither step* parameter can be selected which defines the dx and dy value (which are both equal to Dither step) for each pixel inside the scanner bitmap. The original pixel number Nx,Ny will be changed to new ones Nx=DX/DitherStep with DX as the X Dimension of the bitmap, and Ny=DY/Ditherstep respectively. With this a gray scale bitmap can be transformed into larger pixel size,

for example:

Index

- 2 -

2D Transformations
By keyboard 259

- 3 -

3D Transformations 261

- A -

Access Rights 136
Alignment 194
Text2D 249
Array Copy 171
Assembly Line 330
AutoCal 257

- B -

Barcode 219
Data Matrix 224
Beam Compensation 185
Bitmap 178, 229
Extended 233
Marking Bidirectional 234
Break Angle 151

- C -

Camera 181
Code Format
Barcode 220
Date Time 245
Enable~ 224
Serial Number 239
Command Line Parameters 490
Control
Objects 251
RS232 254
String Mode 254

- D -

Data Wizard 185
Delays 499

- E -

Edit Pens 142
Drill Settings 155
Main 144
Miscellaneous Settings 151
Scanner Settings 146
Edit Resource
Dialog 494
String Editor 495
Element Info 268
Entity List 200
Overview 200
Point Editor 205
Entity Offset 354
Entity Property Sheet 214
Executable 253
Export 281
External Trigger 284
Extras 174

- F -

F1 Help 115
Flash 334
Fonts
Converter 483
Generate 481
Scaps Font Format 482

- G -

Geometry Objects 217
Grid 104

- H -

Home Jump Style 161

- I -

Import 271
Advanced 275
File Formats 280
Indexing 200
Installation 347, 352
User Data 492
IO Job Selection 292

- J -

Job Format 168
Job Properties 352

- L -

Language 493

- M -

Mark 139
Advanced 161
Edit 142
Mark Dialog 284
Mark Menu 282
Marking on the Fly 308
Simulation Mode 311
Marking Order 249
Menus
Edit Menu 171
File Menu 167
Help 180
Mark Menu 282
Overview 166
Window 180
Multi Head 347
MultiHead 346

- N -

Nudge Step 104

- O -

Object Hierarchy 216

- P -

Password 135
Pen Colors 104
Pen Groups 275
Pen Paths 160
Personalize 491
Pixel Map 161
Pixelmode 505
Point Cloud 275
Point Editor 205
Power Map 161
Power Save Mode 113
Preview Window 287
Programming
Command Set 400
Constants 423
Examples 464
Programming Interface 397
Property Page
Date Time 245
Entity Info 266
Hatch 262
Mark 139
Text2D 248

- Q -

Quiet Zone 224

- R -

Radial Text 249
Red pointer
Wavelength Factor 113
Redpointer 284

- S -

ScOpenEthernetConnection 400
Secondary Head 351
Serial Number 237
Serialization
ASCII 242
Automate 242
Example 243
Excel 243
Set Output 251
Set Override 258, 259
Settings 113, 115
3D 134
Card 137
Default Shortkeys 115
Drill 155
Extras 132
General 116
IO 127
Laser 144
Laser Calibration 161
Optic 106
Pens 142
Scanner 146

Settings 113, 115
Shortkeys 115
View 104
Simple Fonts 481
Single Characters 249
Skywriting 151
Slicing 380
Sort 185
Spacing 194
Spacing Advanced 172
Splitting 293
Status Bar 215
Step & Repeat 176

- T -

Teach Reference 174
Text2D
Properties 249
Timeinfo 282
Timer 251
Toolbars 104
Camera Toolbar 181
Extras 194
File Toolbar 181
Geometry Object 183
Object Toolbar 184
Overview 180
Special Sequences 195
View Level Toolbar 182
Trigger 256

- U -

Use ASCII for Serialization 242
Use Pen Colors 277
Use Simple Fonts 481
User Interface 166
User Level 135
User Login 180

- V -

View 2D
Operations 207
Overview 207
Print Preview 213

- W -

Wait for Input 251

- Z -

Z-Axis 374