SAMLight Manual

Rev. 1.7.2, 07.01.2013

Table of Contents

1 Introduction

1.1 Version History

Date	Changes
14.10.2011	Initial version
09.11.2011	Added new Client Control Command ID and Data Value Types for "Beam Comped Copy"
06.12.2011	Added new Clinet Control Constant: scComSAMLightClientCtrlLongDataIdEntitySetAsBackgroundEntity
15.12.2011	New USC-2 functionality for combining DAC and LaserGate signals
30.03.2012	Added information for YAG2, YAG3 and YAG4 modes for RTC cards
19.06.2012	Added description for new feature: Day Map
11.09.2012	Added new Client Control Constant Flag: scComStandardDeviceStyleFlagEnablePortLaser

Table 1.1: Version history

1.2 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.

1.3 Overview

Welcome to the Sam-Light scanner application. The documentation describes the standard scanner application SAMLight. SAMLight is an application to control scanheads and lasers in order to do marking on different materials, 3D marking, welding, cutting and many more. The user interacts using the graphical user interface including dialogs and editors. The options Marking on the fly, Multihead marking, Flash and SAM3D are also explained.

1.4 Position within the system

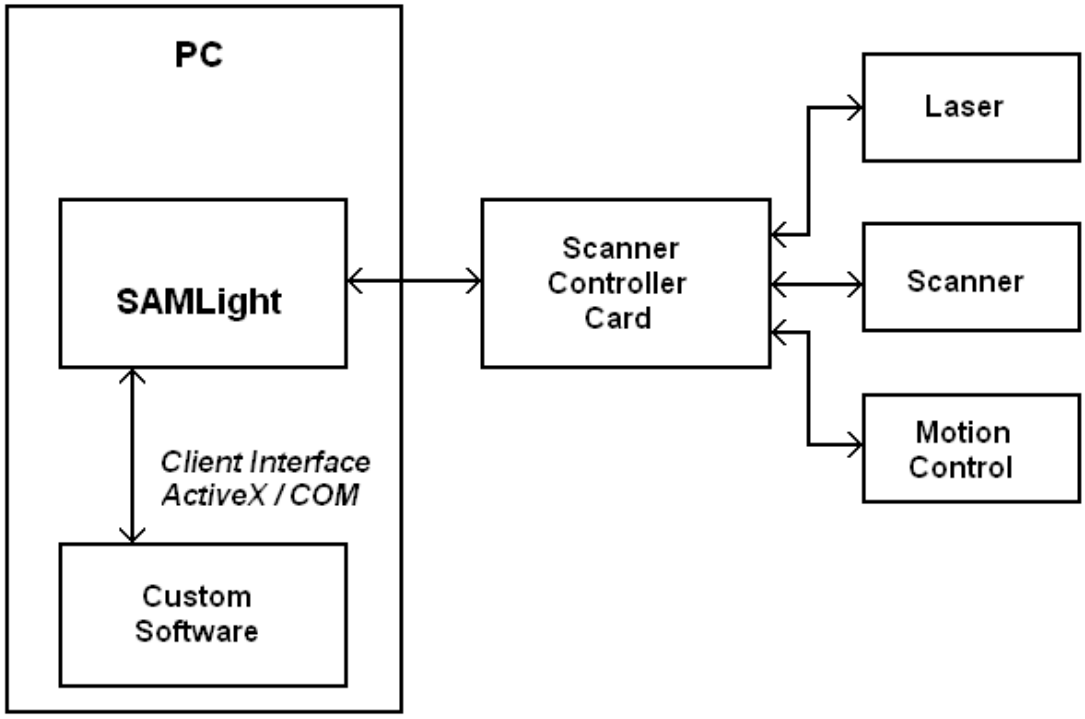


Figure 1.1: Position within the system

1.5 Features

- Create Jobs with all kind of geometrical objects, barcodes, serial numbers, date and time objects and bitmaps
- Import and Export of different file formats
- Split huge jobs into small pieces to be marked one after another
- Marking on the Fly, Store jobs to Flash cards, 3D Marking, Control of multiple scanheads
- Implement custom software controls via the Client Interface
- Control digital and analog output and input signals
- Control multiple scanheads with multiple cards or control two scanheads with one card
- Create custom laser fonts with the SCAPS font converter
- Control one or more motors to displace the working area
- Define Pen settings with different marking velocities, scanner and laser delays and laser power output also ramping and drill mode are available
- Choose between language resource files or create them

2 First Steps

The latest installer of SAMLight can be downloaded at:

SCAPS Download

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 to: C:\scaps\sam2d\.

2.1 Installation and Setup

Installing SAM Software

Run sc_sam_setup_v_3_1_X_YYYYMMDD.exe and follow the instructions.

Note:
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.

Installing the dongle

1. Plug in the dongle at the USB connector of the PC (not necessary if there is an USC-1 or USC-2 scanner card installed).
2. Start SAMLight and type in your password:

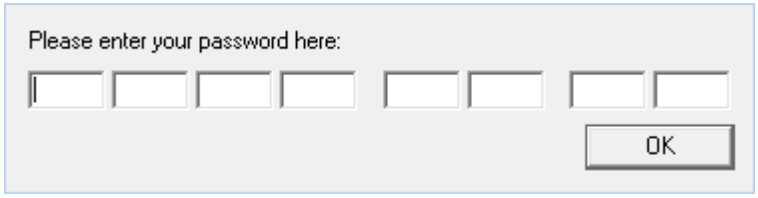


Figure 2.1: Password Dialog

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

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

Installing scanner driver card
Install the scanner driver board and the software drivers as described in the manufacturer's manual. Copy the correction file and the driver files (dll...) in a separate folder of the hard disk.

Register scanner driver card
Before using the scanner driver card with the SAM software, the type of card, the laser type and the location of some files must be defined. Other settings can also be modified directly in SAMLight and the Standard2D software. See sc_setup for further information.

1. Open INST_DIR \ tools \ sc_setup.exe, menu "Hardware Settings".
2. Select the name of the settings file (sc_light_settings.sam for SAMLight, sc_settings.sam for Standard2D) and click on "‹-Load".
3. Select the Device Type and go to "Driver Settings".
4. After all settings are done click on "Save-›" before closing the Hardware Setings dialog

Card settings
Define the location of the files for the scanner driver card within SAMLight->Settings->System->Card or Standard2D->Settings->Scanner->Global->Driver Settings.

Optic settings
Within the optic settings (SAMLight->Settings->System->Optic , Standard2D->Settings->Scanner->Global), the field size, working area and the home position can be defined. These settings are very important to get the correct size of the marking result.

2.2 sc_setup.exe

The sc_setup provides all functionalities to setup the hardware and software. It is located in the folder <SCAPS>\tools.

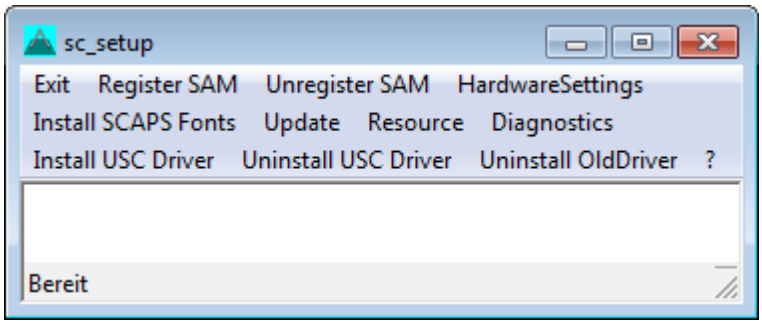


Figure 2.2: sc_setup.exe Dialog

Menu Item	Description
Exit	exit the software
Register SAM	register the SCAPS dlls
Unregister SAM	unregister the SCAPS dlls
HardwareSettings	open the "GeneralSettings" dialog
Install SCAPS Fonts	install the SCAPS laser fonts
Update	update from a previous version
Resource	edit the language resources
Diagnostics	display information about the dongle that is plugged in, software version and dll versions
Install USC Driver	install the USC driver software
Uninstall USC Driver	uninstall the USC driver software
Uninstall OldDriver	uninstall the old USC-1 driver software
?	open the "About" dialog

Table 2.1: sc_setup.exe Menu

2.2.1 Hardware Settings

sc_light_settings.sam is the default settings file for SAMLight. sc_settings.sam is the default SAM2D settings file. These files are stored in <SCAPS>\system. All settings which are described below, will be stored to the settings file.

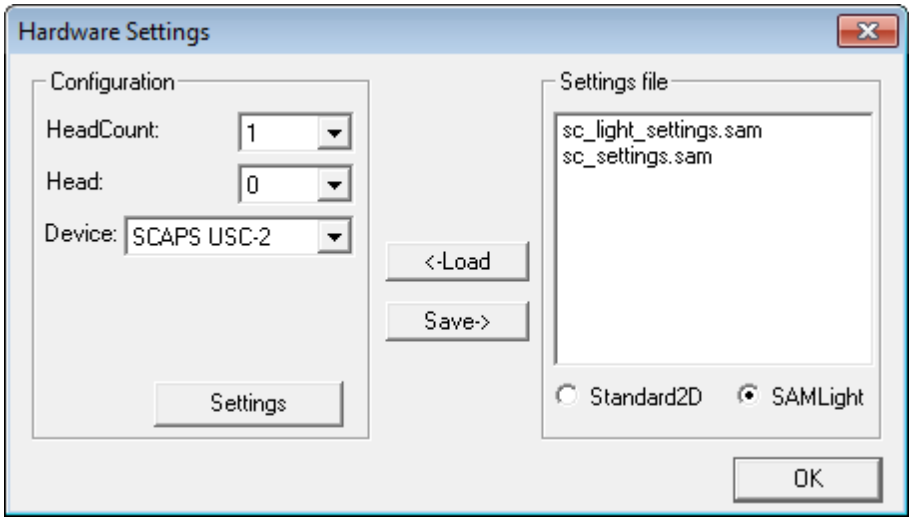


Figure 2.3: Hardware Settings Dialog

- Choose the settings file from the list on the right side and click "<-Load". When using the radio buttons, "Standard2D" loads sc_settings.sam and "SamLight" loads sc_light_settings.sam.
- HeadCount: choose the number of cards

Note: When using USC-2 with Head2 license, it's still one card!

- Head: Select the head which has to be setup.
- Device: Select the hardware device (USC-1/2, RTC-3/4/5, etc.) for the previously selected head.
- Press "Settings" to get to the next dialog.

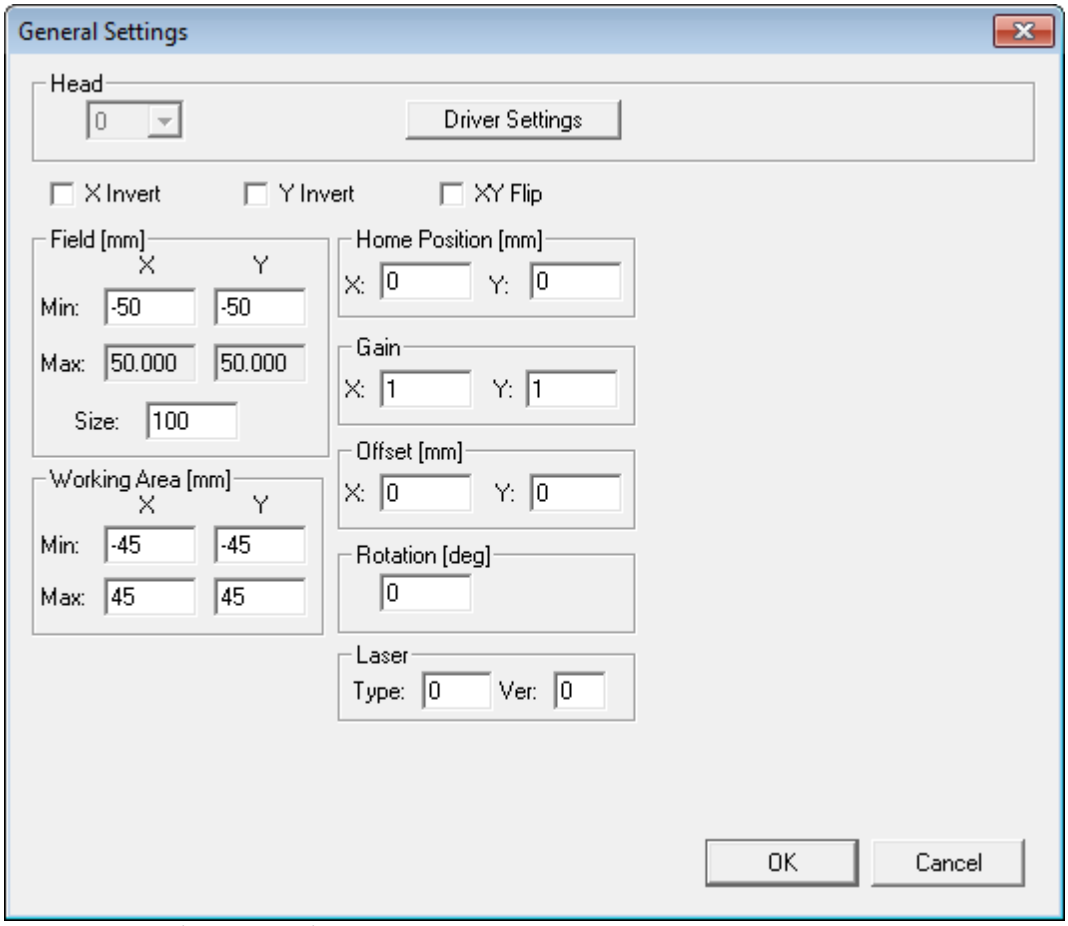


Figure 2.4: General Settings Dialog

Head: Choose the Head for which the "Driver Settings" should be edited.
The drop down menu will be enabled only if the 'HeadCount' in the previous dialog is bigger than 1.

Driver Settings: Opens the Driver Settings dialog where hardware specific settings can be edited.

Invert / Flip:
- 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 field size has to be typed in in mm.
- The field is always a square. The edge length is 'Size:'.
- The 'Max:' value is computed as 'Min:' + 'Size:'.
- The 'Min:' value can be negative, so that the field can be set up symmetrically to the origin.

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 the Rotation too.

Home Position [mm]:
- defines the position, from where the scanner starts its movement
- If HomeJump is enabled, the scanner goes back to this position after marking.

Gain:	The gain values are thought to slightly compensate X/Y gain errors to achieve a quadratic field.
Offset [mm]:	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 direction should be corrected by changing the field X/Y min values in the Field edit group.
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. - 'Home Position': defines the position, from where the movement of the scanner starts. - If HomeJump is enabled, the scanner goes back to this position after marking.

2.2.1.1 Driver Settings

The Driver Settings dialog can be used to setup the SAMLight startup defaults for the specified hardware. For a more detailed instruction please refer to: Card Settings.

2.2.2 Information

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
- sam_light.exe
- *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_SAM\system.
- *sc_CUSTOMERID_DONGLEID.scl* -› password file, see folder SCAPS_SAM\system.

Manual

After installing the software, there is a subfolder /doc, which contains the manual *sc_manual_sl_English.pdf*. The manual *sc_manual_sl_English.pdf* contains all important help for the user, with special chapters as:

- *How to \ ...\ Customize Language* -› Shows how to translate the user interface into other languages.
- *How to \ Simple Fonts* -› Explains the use of the font editor for creating customized laser fonts.
- *External Control \ Programming Interface* -› Shows how to use the remote capabilities of SAMLight by program calls.

Remark: The manual *sc_manual_sl_English.pdf* has the same content as the help file *sc_help_sl_English.chm*, which is in folder SCAPS_SAM\hlp.

2.2.3 Diagnostics

The menu item "Diagnostics" -> "Dongle" displays the following dialog.

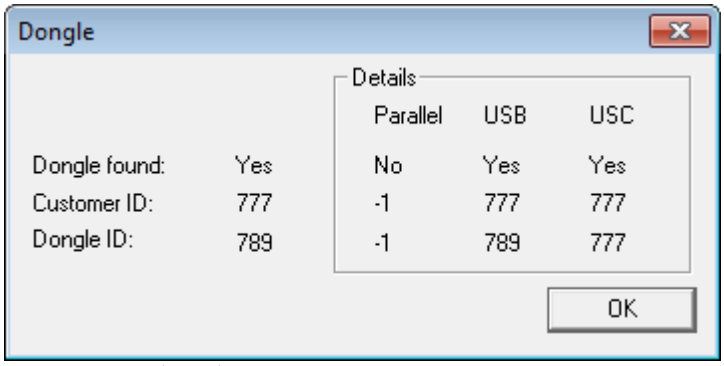


Figure 2.5: Dongle Dialog

Note:
Use the sc_usc_server.exe in visible mode to display the dongle information of multiple USC cards!

3 Getting Started

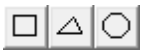
This chapter describes some of the basic functionality of the scanner software and offers users the possibility to start using the application easily. The first steps described here of course are more or less incomplete, a deeper knowledge of the application and all of its often complex features and functionalities will be given later within this documentation.

3.1 Creating A Job

When the scanner application has started successfully the main window appears. Here it is possible to define a new job. A "Job" is some kind of simple program flow that contains all the entities that have to be marked in the order they are marked. Such a job can be created and edited here. The order the entities are marked later can be seen and changed within the so called Entity List. That list can be made visible by selecting the appropriate button within the toolbar:

The other button right beside it opens the "Property Sheet" area on the right hand side. This area will be used later to change the properties of selected entities, define styles for marking and do some fine tuning for them.

First a simple job has to be created. As a first example a rectangle, a circle and a triangle have to be added to the big area in the middle that acts as some kind of editor for the jobs geometry data. The desired primitives can be found as toolbar buttons again:



Here the desired objects can be found. First click on the rectangle button. Now you are able to place that primitive within the working area by choosing the desired position of its top left corner. To do that click with the mouse at the appropriate coordinates within the view and hold the left mouse button down. Now drag the mouse to the position you want to have for the lower right corner of the

rectangle and release its left button at this position. During dragging the mouse the rectangle becomes visible. After that operation there exists a first entity not only within the view in the middle but also in the entities list on the left hand side.

Next the circle has to be drawn. Here the same procedure is necessary. The positions where the mouse button is pressed and released define the corners of a box that enclose the circle. If the horizontal and vertical distance between both points aren't equal, an ellipse is drawn instead of a circle.

For the triangle the order of steps is slightly different. After selecting the triangle-button in the toolbar the first mouse click defines one edge of the triangle. Now you can move (but in this case not drag) the mouse to the position of the second point, click to set that position and finally move to the position where the third edge of the triangle has to be located at. A third click using the left mouse button now ends that operation and finishes the triangle.

The result is a first simple job that could be marked now by opening the marking dialog in menu Mark -› Start:

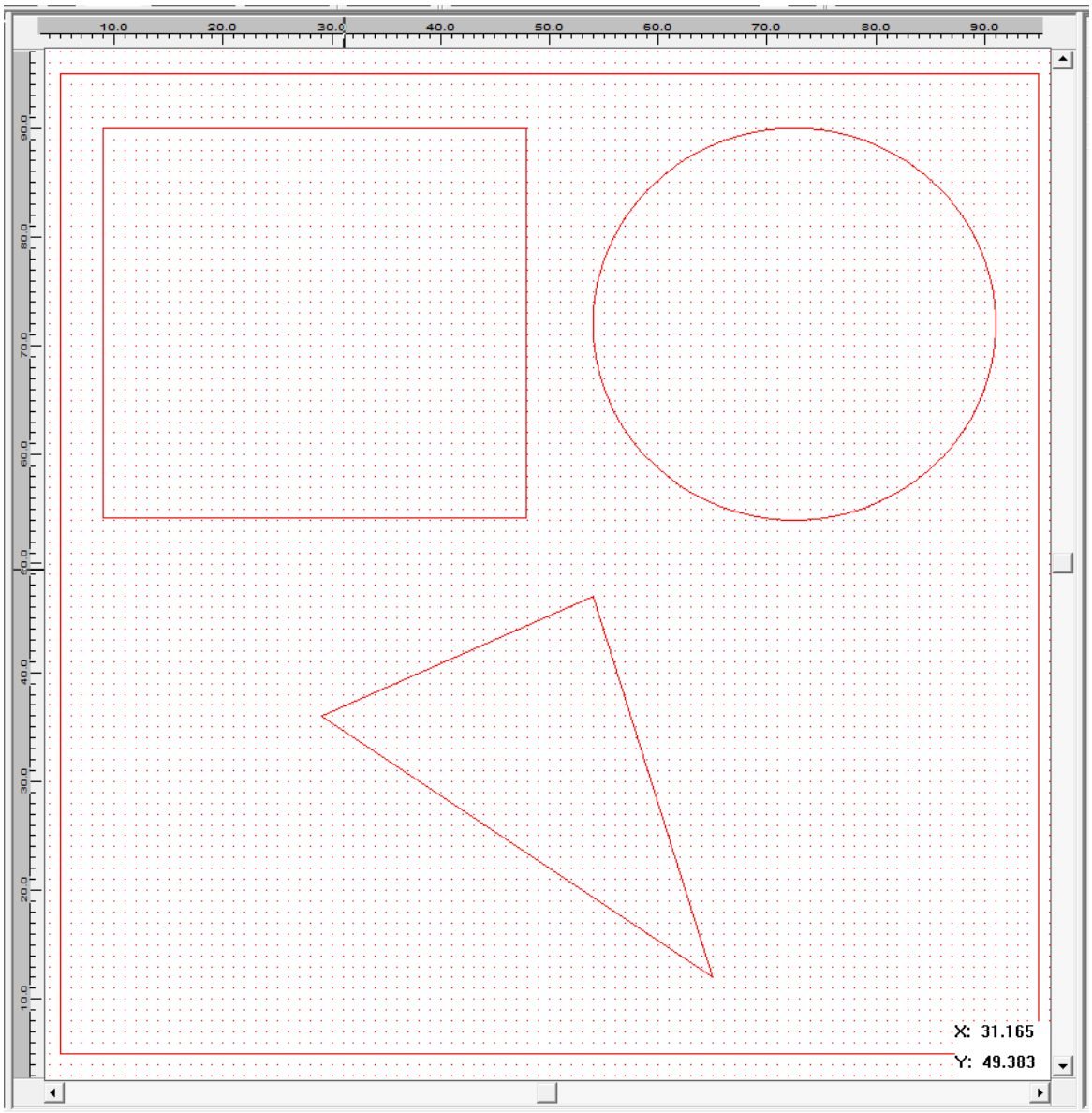


Figure 3.1: A Simple Job

3.2 Modifying Entities

As a next step the small job that currently was created has to be modified. So the rectangle has to be marked 10 times using a different laser power. The circle has to be modified so that it becomes an ellipse and its position has to be changed. The triangle has to be filled so that after marking not only its contours are visible afterwards.

First the rectangle has to be selected either directly in the entity list on the left hand side or by clicking its geometry in the view. Resulting from that some tab panes on the windows right hand side become active. First the "Entity Info" pane is required. Using the input field "Mark Loop Count" the number of loops per single marking operation can be defined, that means how often that entity is repeated during one marking cycle. Here the new value 10 has to be entered. Afterwards this operation needs to be finished by pressing the "Apply"-button.

Please note: This button has to be pressed after every change within this property pane. Else the new settings wouldn't take effect. That is true not only for the "Entity Info" and the rectangle but for all entity types and their properties.

The picture above shows three interesting things: The rectangle is selected in the entity list on the left. Here it is named "ScLayer" because that is the entity type that is used to hold geometry data like they are required for such primitives. The geometry itself is highlighted in the editor view in the middle and all the property pages that offer valid parameters for this kind of entity type on the right hand side are active and editable. Conform to the example described above the settings for the mark loop count have been changed:

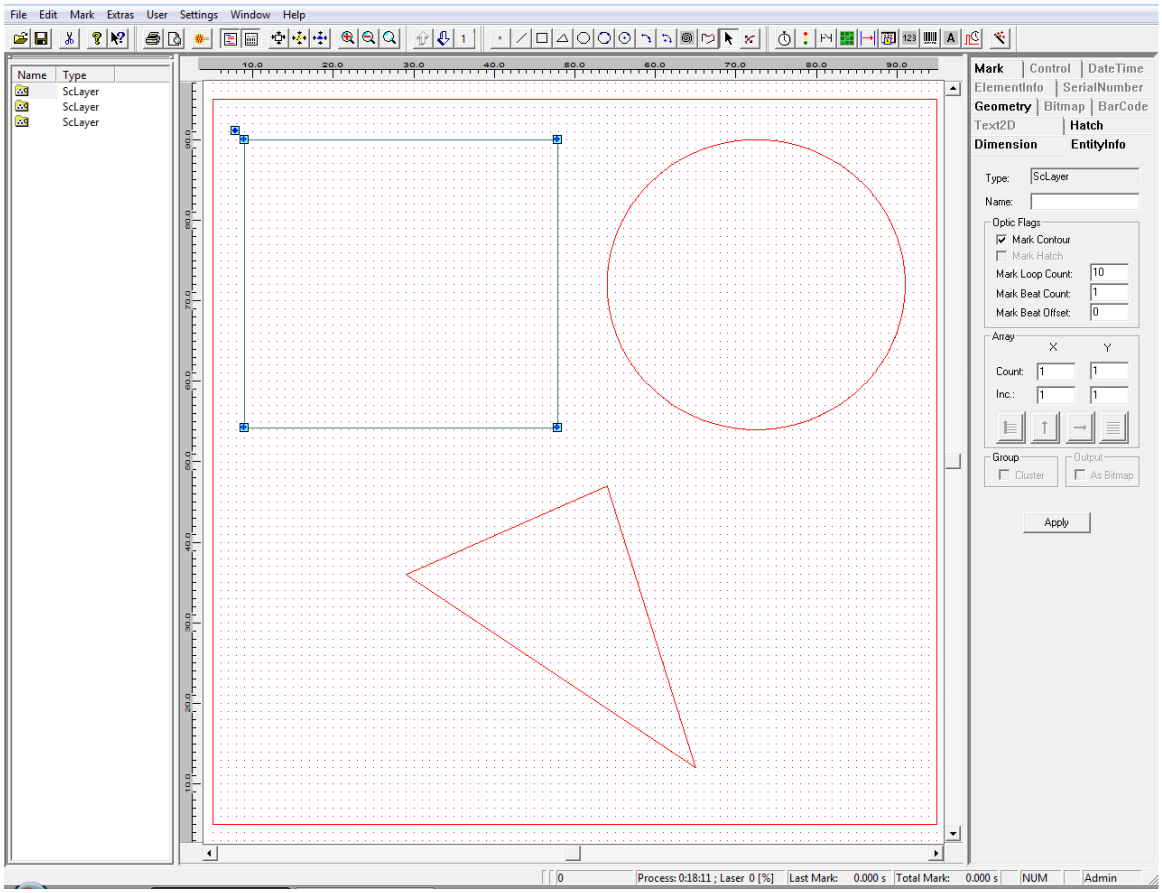


Figure 3.2: Modifying an Entity

As a next step different laser settings are required for that entity. For every element of a job a separate pen can be chosen. There are separate laser settings definable for every pen so that these pens are the way to modify the laser parameters for the rectangle. Another tabpane that is active and usable when a markable entity like the rectangle is selected, is the "Mark"-pane. Now here the list of available pens can be seen. By pressing the "Edit..." button a dialog opens where the settings for the currently selected pen can be changed. Depending on the laser type several values like laser power, frequency and others can be changed. To set a different pen for the rectangle it is necessary to choose the new pen from the list within the "Mark"-pane and apply that change. Now the new values will be used during the next marking process.

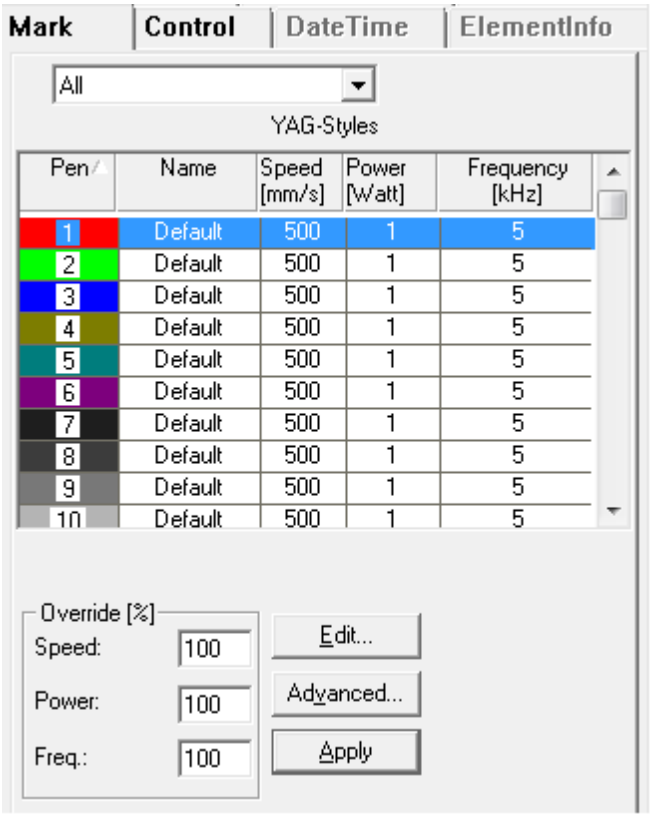


Figure 3.3: Pen Settings

As a next step the circle is selected in order to change its position. When it is highlighted a box is drawn that surrounds that circle. At the edges of the box some small areas can be seen that are important for the desired operation. These small, blue-green boxes directly at the bounding rectangles edges can be used for dragging the entity and for changing its position. Beside of that the additional box on the upper left corner of the bounding box can be clicked to change the editing mode for the entity. For the circle the modes "translate" (default value), "scale" (after clicking it once) and "rotate" (after clicking it a second time) can be chosen.

To create an ellipse out of the circle there are two ways possible: in scale mode it is possible to change the circles size in horizontal or vertical direction only by dragging the green-blue boxes that are located at the bounding rectangles side. A more exact method would be to use the "Geometry" property page that becomes active after the circle was selected. Here new values for its X- and Y-radius can be entered. Applying these changes causes the appropriate modification of the geometry of the circle:

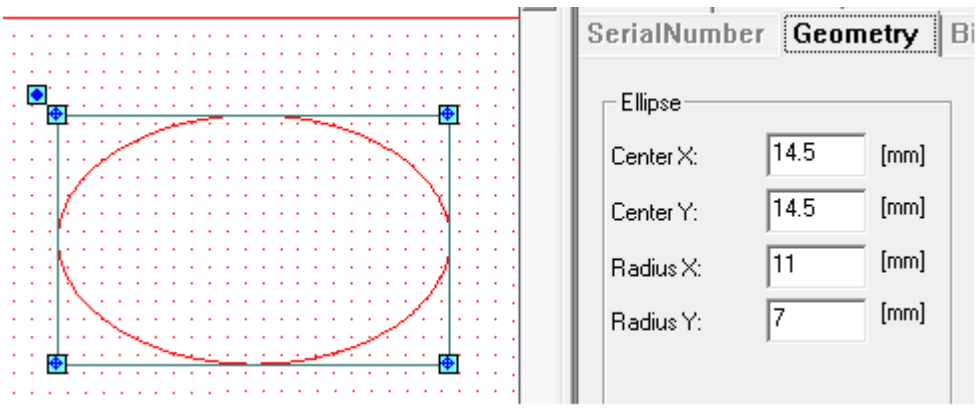


Figure 3.4: Geometry Settings

As a last step a modification has to be applied to the triangle. This is performed by using the property pages too. To access them the triangle needs to be selected by left-clicking its geometry in the middle view or by selecting the last entry within the entity list at the left hand side. Now a property page named "Hatch" becomes active. Here up to two hatch patterns can be defined that are used to fill the geometry. To hatch an object the "Enable"-checkbox for one of both hatch patterns has to be selected and a "Hatch Distance" needs to be defined. That distance specifies how far two single hatch lines that are used to fill an object during marking are away from each other. Applying these setting causes a change in the middle view, now the hatch lines are added visibly to the triangle:

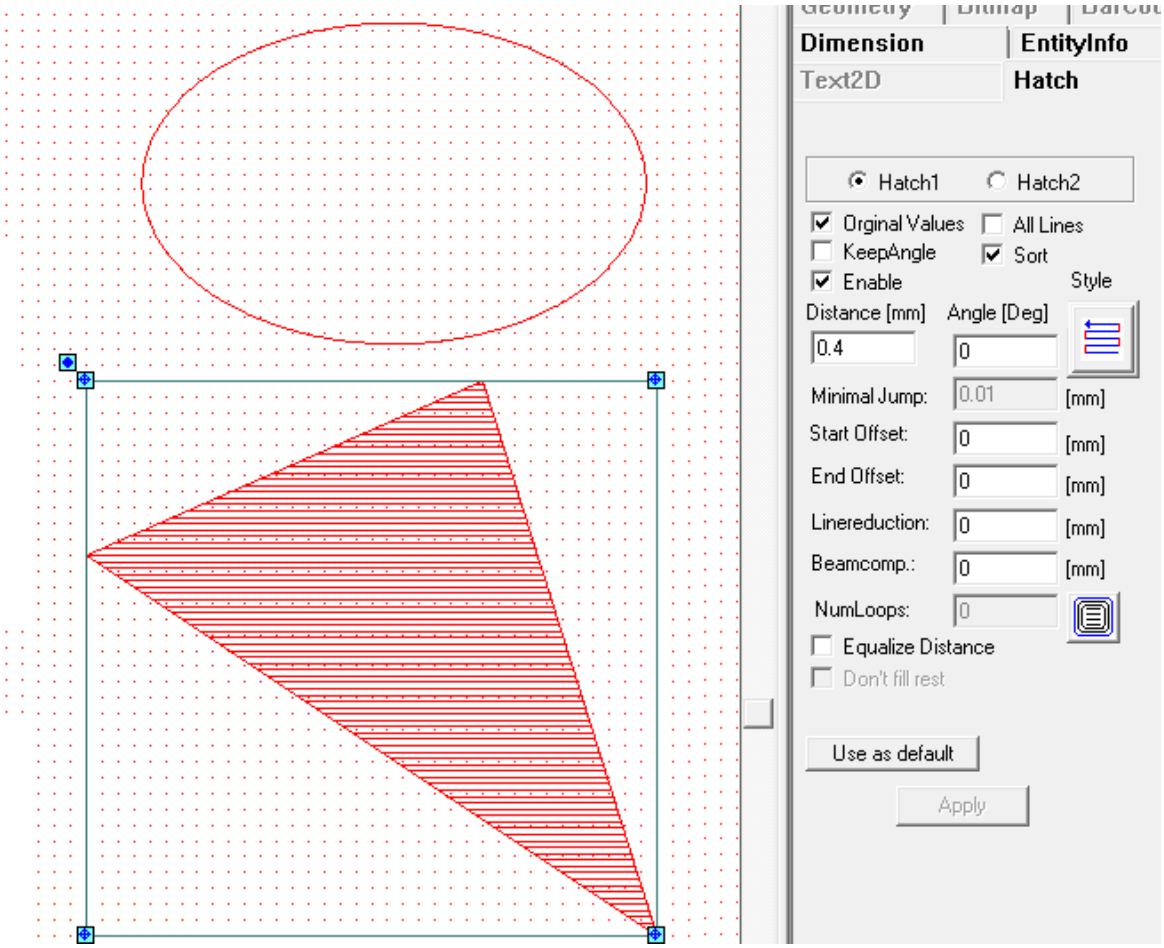


Figure 3.5: Hatch Options

After all these modifications have been made with the job the next marking process shows a completely different result.

4 User Interface

At start up the application displays the user interface like it is shown below:

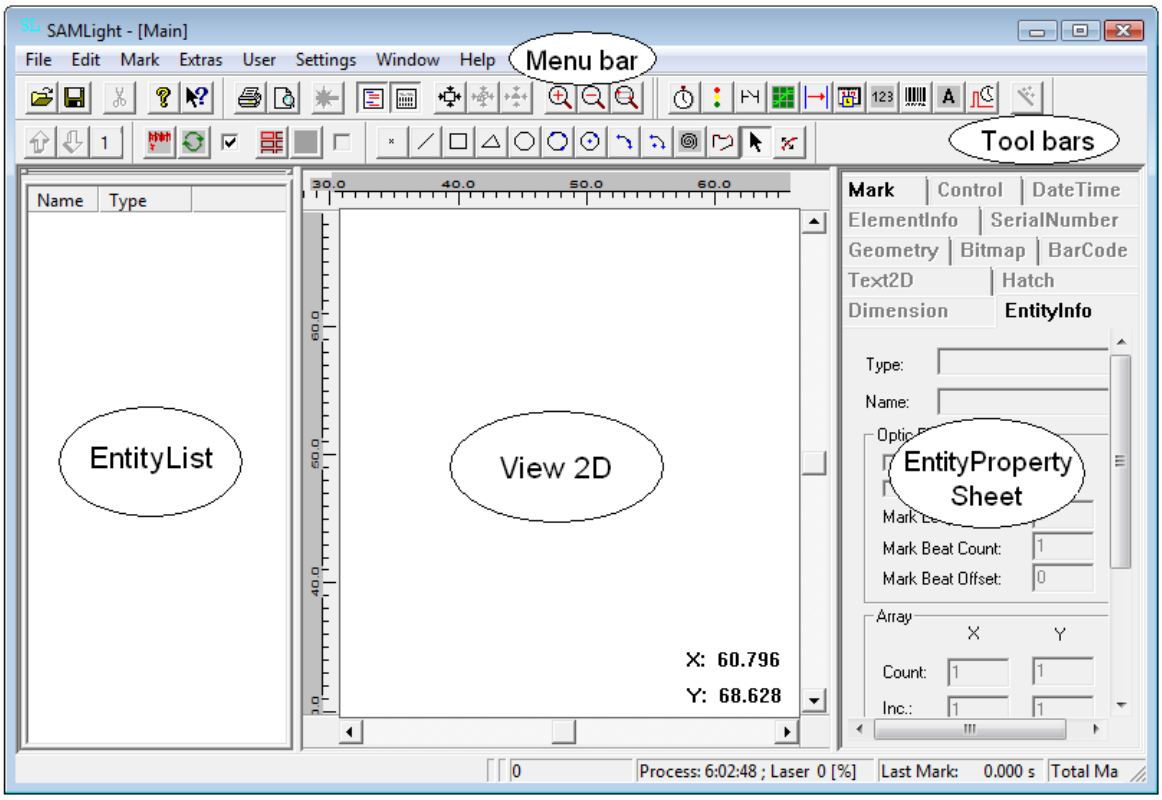
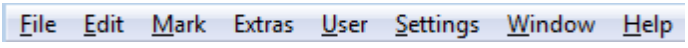


Figure 4.1: Main Window

Topics of User Interface:

1. Menu bar
2. Tool bars
3. Main Window: Entity List / View 2D / Entity Property Sheet
4. Preview Window

4.1 Menu Bar



Topics of the Menu bar:

- File
- Edit
- Mark
- Extras
- User
- Settings
- Window
- Help

- Special: The menu item "Special" will be added to the Menu bar, if there are user-defined IO Settings.

4.1.1 File

New
Prepares for a new job and it deletes all current entities.

Load...
Opens a dialog to read jobfiles in SCAPS .sjf (SCAPSJobFile-Format) format. See chapter Job Format.

Save...
Saves the current job in sjf format. If this menu item is clicked for the first time, it will behave like **Save as...** See chapter Job Format.

Save as...
Opens a dialog to save the current job under a new file name (in sjf format). See chapter Job Format.

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.

Import...
Import of data in format:

- HPGL (*.plt)
- Bitmap (*.bmp)
- Autocad(*.dxf)
- SCAPS Archive (*.saf)
- Adobe Illustrator (*.ai)
- PC-Mark (*.job)
- MCL (*.mcl)
- PCX (*.pcx) file format
- DXF Version 2 (*.dxf)
- Corel Presentation Exchange (*.cmx)
- Enhanced Windows Metafile (*.emf)
- Point Cloud (*.txt)
- From Scanner (*:twain)
- Scalable Vektor Graphics (*.svg)

See chapter Import.

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
Quits the application.

4.1.2 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.

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 opens the dialog below:

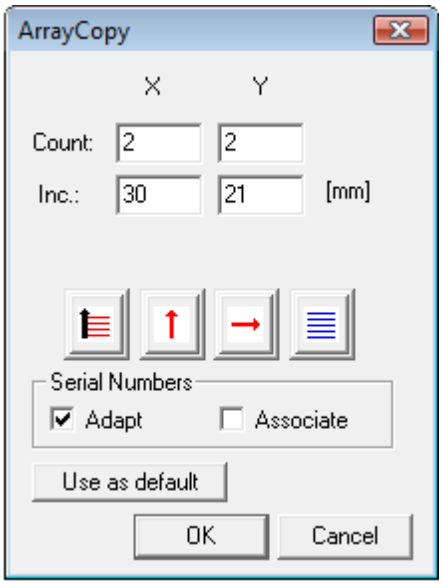


Figure 4.2: Array Copy Dialog

ArrayCopy creates an array of copies of the selected object, where:
- 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)

Serial Numbers

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.

Select:
Provides functions for selecting the objects one by one etc.. Works in the first view level.

All:
Selects all objects in the job.
First:
Selects the first object in the job.
Previous:
Selects the previous object in the job.
Next:
Selects the next object in the job.
Last:
Selects the last object 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. There are three possible centering methods:

Horizontal:
Centers the selected object along the horizontal axis
Vertical:
Uses the vertical axis to center the currently selected object
Both:
Centers using both axes (horizontal and vertical)

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.

4.1.2.1 Spacing Advanced

The following dialog can be found under Edit->Spacing->Advanced...

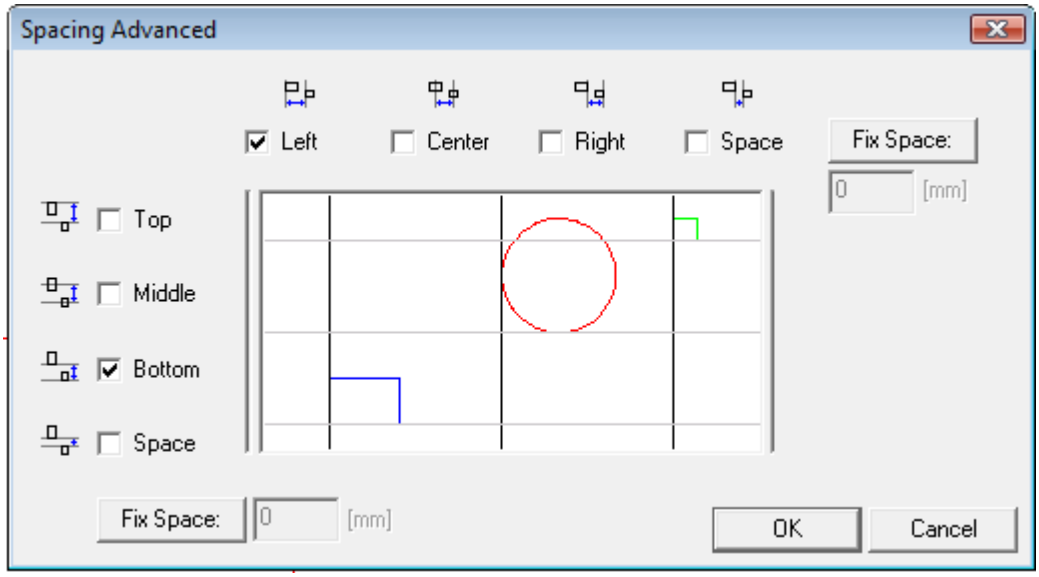


Figure 4.3: Spacing Advanced Dialog

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.

4.1.3 Mark

Start:
Opens the Mark Dialog for to control the mark process.

Trigger:
Sends all / all selected (the selected definition is set in Mark Dialog) entities to the optic device. The controller board exposures then 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.

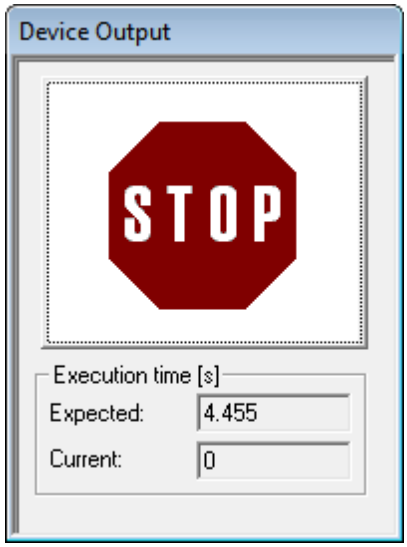


Figure 4.4: Mark Trigger Dialog

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...:
Reset Sequence:
Resets/Restarts a mark sequence. Please refer to section EntityInfo to get details on how one can define a mark sequence.
Reset SerialNumber:
Resets all Serial Numbers of the job to their defined start values.
Reset Counter:
Resets all Mark Statusbar information.
Reset 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 EntityInfo 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 EntityInfo to get details on how one can define a mark sequence.

Decrement:

Sequence:
Decrements the current sequence of the job. Please refer to section EntityInfo to get details on how one can define a mark sequence.

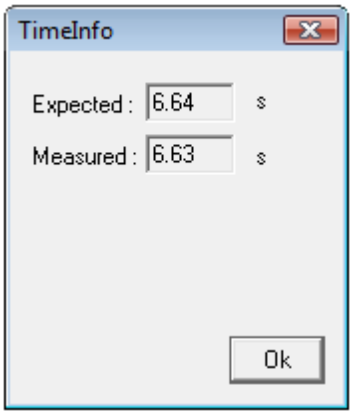
SerialNumber:
Decrements all SerialNumbers of the job.

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 EntityInfo 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 EntityInfo to get details on how one can define a mark sequence.

TimeInfo...:
Shows the TimeInfo dialog below.



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:
Generates a preview of the actual job. Switch to the Preview Window by clicking Menu bar->Window->Preview to see the result.

Teach Reference and **Relocate Reference**:
Opens the teaching / relocating dialog for teaching or relocating the position of an object that has to be marked.

4.1.3.1 Mark Dialog

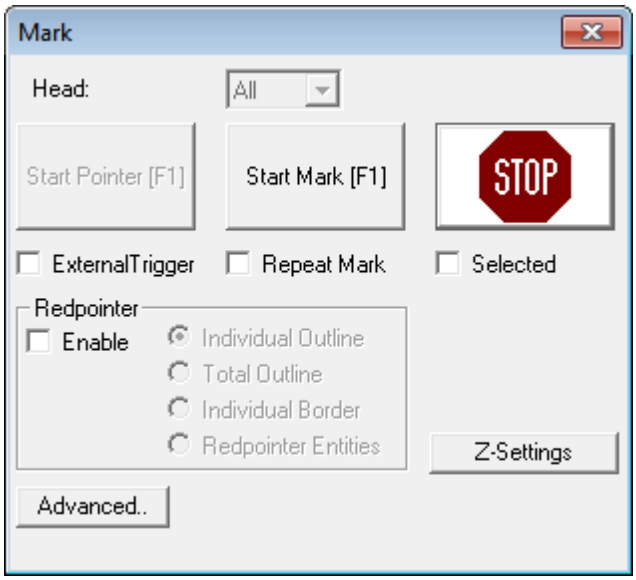


Figure 4.5: Mark Dialog

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.

Buttons:

Start Pointer: Starts the red pointer. This button is active only if the red pointer is enabled. For the red pointer pen #254 is used (see chapter Mark Settings). While the redpointer is outlining, scale and nudge can be done with the short keys listed below.
Ctrl + Arrow Key: Nudge
Shift + Arrow Key: 5 * Nudge
Alt + Arrow Up/Down: Scale
Shift + Alt + Arrow Up/Down: 5 * Scale
Note: Nudge Step and Scale Factor can be defined within Settings->System->View.

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 or by pressing F1 - or by external trigger) all objects that have to be marked are sent down to the optic device and are exposured.

Check buttons:

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 - Box:

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 redpointer draws the real geometry of the object.
Redpointer Entities:	If checked, only 'redpointer 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 'Advanced'-button:

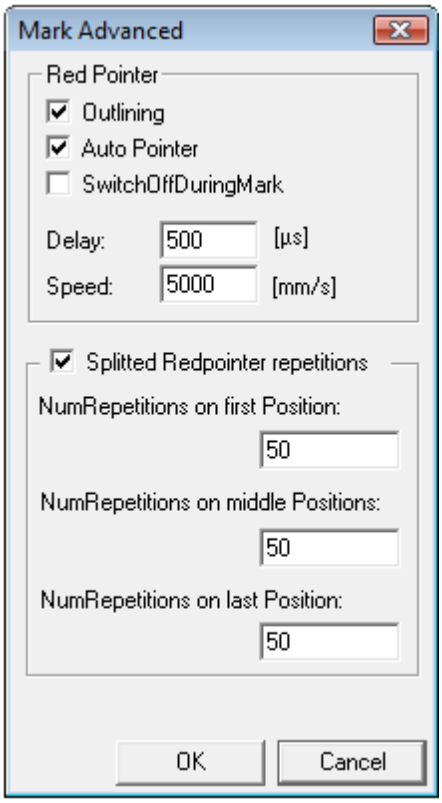


Figure 4.6: Mark Advanced Dialog

Red Pointer:

Outlining: If checked, drawing of outline is activated.
Auto Pointer: If checked, the red pointer starts automatically before each marking.
SwitchOffDuringMark: If checked, the red pointer is switched off during mark.

Edit Input field(s):
Values 'Delay' and 'Speed' determine red pointer delay and red pointer speed.

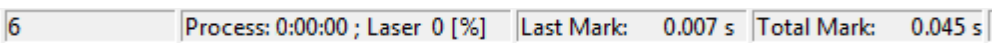
Splitted Red Pointer repetitions:

Can be checked only if entity is a splitted entity. In this case:

- For first, last and splitting parts in between a red pointer repetition count can be set.
- If unchecked (default), the red pointer runs endless on first split part.

If the entity is not a splitted entity, this option is deactivated and the red pointer runs endlessly as expected.

4.1.3.2 Mark Status Bar



The mark status bar displays information about the mark process (from left to right):

1. Number of quantities.
2. The process time in seconds and the percentage of time when the laser was on.

3. The duration of the last mark process in seconds.
4. The duration of the whole mark process in seconds.

4.1.4 Extras

The Extras menu contains several special functionalities:

Teach Reference - Teach new reference position(s)
Relocate Reference - Recall and use reference position(s)
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

4.1.4.1 Teach / Relocate Reference

These two menu items can be used for teaching and relocating reference positions.

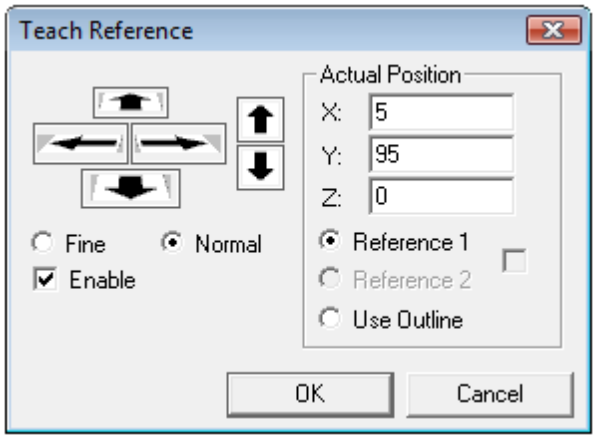


Figure 4.7: Teach Reference Dialog

Teaching and relocating can be used to define reference positions for a job or work piece that have to be recalled and used for a position correction later. That is useful when it is more difficult to change the position of the object that has to be marked instead of relocating these references. Here a typical working process contains the following steps:

1. One or two reference points are taught for a work piece using specific, non-ambiguous and easy to remember positions that are related to the work piece.
2. When the Dialog above is left using "OK" these reference points are stored into the job so it is recommended to save that job afterwards.
3. When a marking process was finished and the marked object was exchanged, the relocate dialog has to be opened.
4. Now the positions of the reference points are checked. They are compared with the new position of the work piece and - if necessary - corrected so that they afterwards are located at the same position as before (related to the work piece, not the working area).
5. After the relocate dialog has been left the job is modified in a way so that it fits to the work piece exactly as desired.

The steps 3 to 5 need to be repeated for every marking pass.

For teaching and relocating reference points the related dialog offers the following functions. These functions can be controlled by the keys described in brackets:

Cursor Buttons (CURSOR UP, DOWN; LEFT, RIGHT): Move the currently selected reference point in horizontal direction using the same level (X- and Y-axis). The width of such a movement depends on the selected stepwidth.

Depth buttons (PAGE UP, DOWN): Move the actual reference point in vertical direction (Z-axis, depth coordinate). The width of such a movement depends on the selected 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 Block:

X, Y, Z: Absolute coordinates of the currently selected reference points

Reference 1 / 2 (toggle with CTRL): Switch between the both reference points. 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. In this case only if reference point two is selected it is moved independently from the first one. This mode can be used for a raw location of the working piece in a first step and avoids the necessity to move both reference points separately for the possibly wide distances. If this checkbox isn't selected, both reference points are changed completely independent from each other.
Please Note: when the program is configured to use only one reference point, these three buttons aren't available.

Use Outline: Instead of a single reference point the outline of a job can be used too for selecting a position. If this option is selected it behaves exactly like 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.

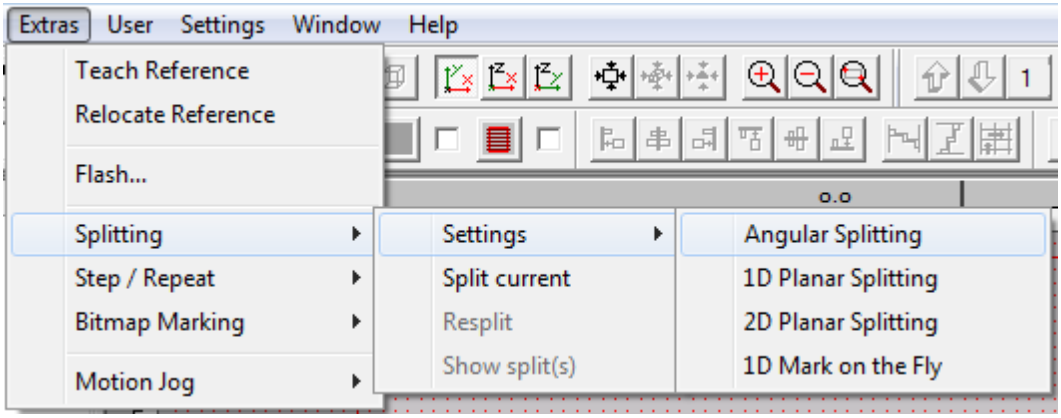
When the teach dialog is left using the "OK"-button, the one or two reference points are stored for the current job and can be used for a next relocation process. Together with these coordinates the information is stored if one or two reference points have to be used. That means if the option "two reference points" is changed in the settings afterwards, the teach process has to be repeated to let this modification become effective for the current job.

When the relocate dialog is left with "OK", the modification of the current job according to the new reference positions is done immediately so that the new work piece can be marked with no additional action.

4.1.4.2 Splitting

Splitted marking allows it to split a job into pieces and then mark it part by part. That automatic splitted marking is useful for e.g. marking round objects or large objects that are bigger than the working area. Jobs are always splitted completely except if there are some entities marked as nonsplittable.

The Splitting Dialogs can be reached via the Extras Toolbar or by selecting the appropriate menu item, like it is shown below:



Rotational marking (Angular Splitting mode) can be done by performing the following steps:
- enabling of the external motion control to perform the automatic rotation of the round object that has to be marked
- definition of the diameter of the object that has to be marked
- definition of the rotational angle that describes the segment that has to be marked at the same time
- definition of 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 doesn't influence the axis the job is splitted for
- turning on the splitting mode to enable the current job for rotational marking

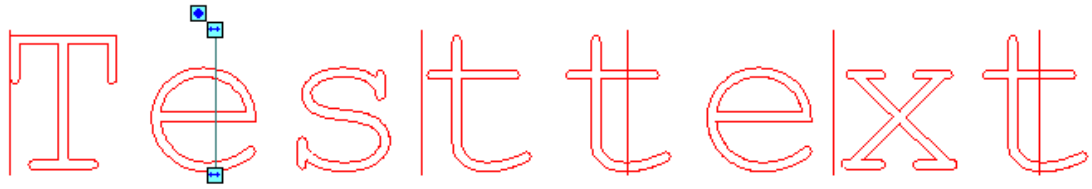
For **planar marking** (Planar Splitting modes) the following steps are necessary:
- enabling of the external motion control to perform the automatic movement of the flat object that has to be marked
- definition of the size of a splitted part that has to be marked at the same time
- definition of the motion controllers axis/axes that has/have to be used for the planar movements. This axis/axes only define(s) the drive that has to be used for the motion. It doesn't influence the axis the job is splitted for.
- switching on the splitting mode to enable the current job for marking it tile by tile

For **On-The-Fly-marking** of split parts following steps are necessary:
- Marking on The Fly feature
- definition of the size of one splitted part that has to be marked on the fly
- switching on the splitting mode to enable the current job for marking it tile by tile

Configuring the MotionControl for the first two modes has to be done using the IO settings panel that can be found at menu Settings->System. Here in field "Motion Control" the checkbox "Active" has to be selected. Additionally the motion control configuration file has to be configured for the used controller as described here.

General Note: If there is none of the **Rotation Axis** or **Motion Axis** radio buttons enabled there is a misconfiguration with the motion controller and the application will not be able to perform the correct rotational or planar movements.

Switching to the splitting mode where the current job is cut into pieces can be performed using the menu **Extras -› Splitting -› Split current**. This mode can be turned off by selecting this menu item again. Then the job is restored to normal, non-splitted mode. To activate splitting mode the related check box of the Extras Tool Bar can be used too.



When the splitting of the job has been done successfully, its appearance is changed. Now the job contains several additional lines. They mark the cutting lines 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: Left 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.

Please note: 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. That is why interleaving the cuts that are represented by these lines isn't a valid operation. If there are parts of the job that are located outside of the outer cutting lines after editing, they aren't marked afterwards.

After dragging a cutting line to a new position the splitted job is recalculated automatically using these new settings. 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 splitted parts are marked step by step until the complete job has been finished. Editing the cutting lines resets the current marking position so that the next marking operation starts with the first segment of that job.

Caution: To get the correct marking result it is very important that the axes of the motors correspond to the axes of the splitted job. Normally they should both form a right handed co-ordinate system.

Please note: If a motor movement is desired, that is independent of the splitting and should be executed before or after the splitting, it has to be defined as a "Nonsplittable Entity". See chapter: Entity List.

4.1.4.2.1 Angular Splitting

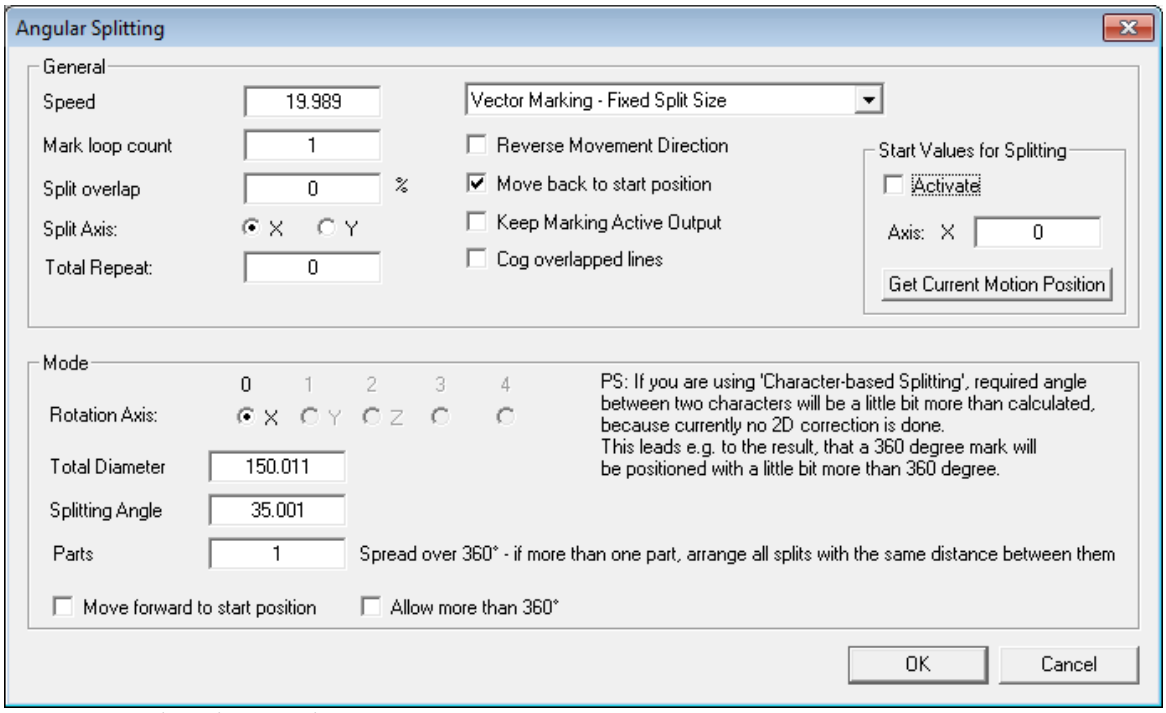


Figure 4.8: Angular Splitting Dialog

General

The option **Speed** is a general movement setting for planar and angular operation modes. It defines the speed that is used for the connected drive. The speed that is defined here overrides the speed that is 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 splitted part is repeated before the motion device moves to the next splitted part.

Split overlap:
With this option one can increase the area of a single split.

Split Axis:
Here the axis that defines the splitting direction for the geometry has to be set.

Total Repeat:
If a value greater than 0 is entered here, then after finishing the split the scanner will move back to the start position and the split will be repeated as often as entered.

Combo Box:
Possible Split options are:
- *Vector Marking - Fixed Split Size*
- *Vector Marking - Entity Based Splitting*
- *Vector Marking - Character Based Splitting*

The **Character Based Splitting** option is available only for the one-dimensional splitting modes. 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 scanhead, all movements go in the opposite direction.

If the option **Move back to start position** is unchecked the movement back to the starting position is not performed after a full split marking cycle. If else the software goes back to the position where the marking operation has started from.

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 activated then the marking active 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 object contains a hatch with many parallel lines then it is better for the quality of the marking, if those lines aren't splitted with a uniform edge but it is better if the lines are cogged. This is because if all lines are splitted on a uniform edge the workpiece becomes hot on that edge and the marking quality is reduced.

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

This mode allows to mark on round objects. The object is being rotated between two marking cycles so that the laser can 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 working piece. 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. Please note: The circumference that results from the diameter of the base surface of course can't be smaller than the total width of all the jobs entities!

Splitting Angle: The angle in unit degrees that describes the size of one piece.

Parts ... Spread over 360°: If the total diameter and the resulting generated 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°.

4.1.4.2.2 1D Planar Splitting

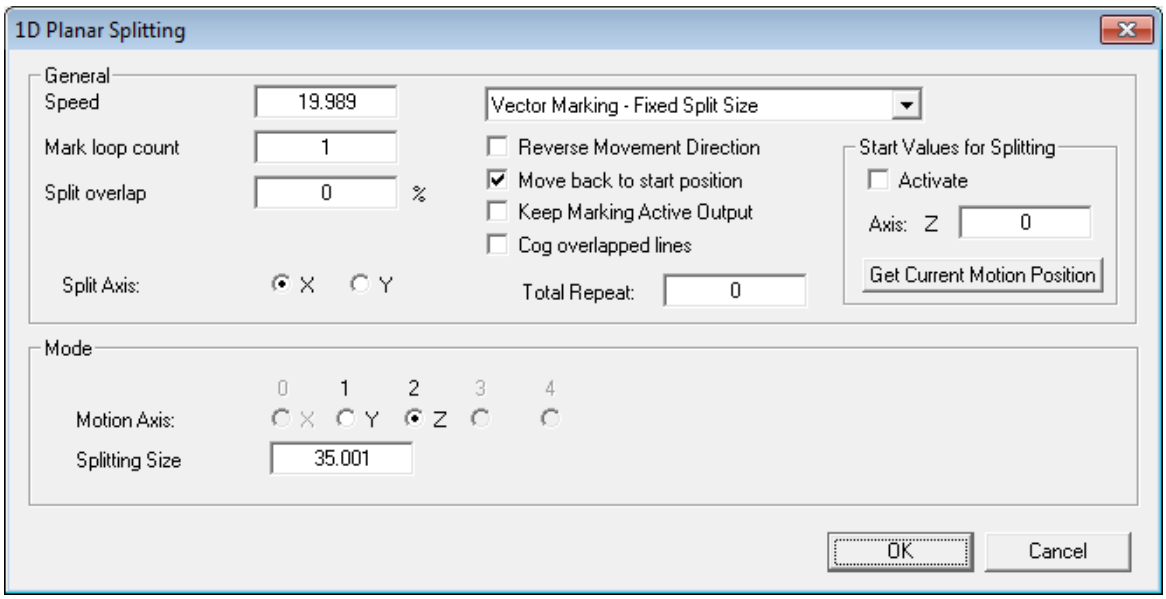


Figure 4.9: 1D Planar Splitting Dialog

General

The option **Speed** is a general movement setting for planar and angular operation modes. It defines the speed that is used for the connected drive. The speed that is defined here overrides the speed that is 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 splitted part is repeated before the motion device moves to the next splitted part.

Split overlap:
With this option one can increase the area of a single split.

Split Axis:
Here the axis that defines the splitting direction for the geometry has to be set.

Total Repeat:
If a value greater than 0 is entered here, then after finishing the split the scanner will move back to the start position and the split will be repeated as often as entered.

Combo Box:
Possible Split options are:
- *Vector Marking - Fixed Split Size*
- *Vector Marking - Entity Based Splitting*
- *Vector Marking - Character Based Splitting*

The **Character Based Splitting** option is available only for the one-dimensional splitting modes. 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 scanhead, all movements go in the opposite direction.

If the option **Move back to start position** is unchecked the movement back to the starting position is not performed after a full split marking cycle. If else the software goes back to the position where the marking operation has started from.

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 activated then the marking active 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 object contains a hatch with many parallel lines then it is better for the quality of the marking, if those lines aren't splitted with a uniform edge but it is better if the lines are cogged. This is because if all lines are splitted on a uniform edge the workpiece becomes hot on that edge and the marking quality is reduced.

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

Using this mode the current job can be splitted in one direction, either X or Y related to the contents displayed in the View2D. 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 splitted tiles.

Splitting Size:
Here the size of one single split can be defined. This size needs to be smaller than the working areas size in the same direction.

4.1.4.2.3 2D Planar Splitting

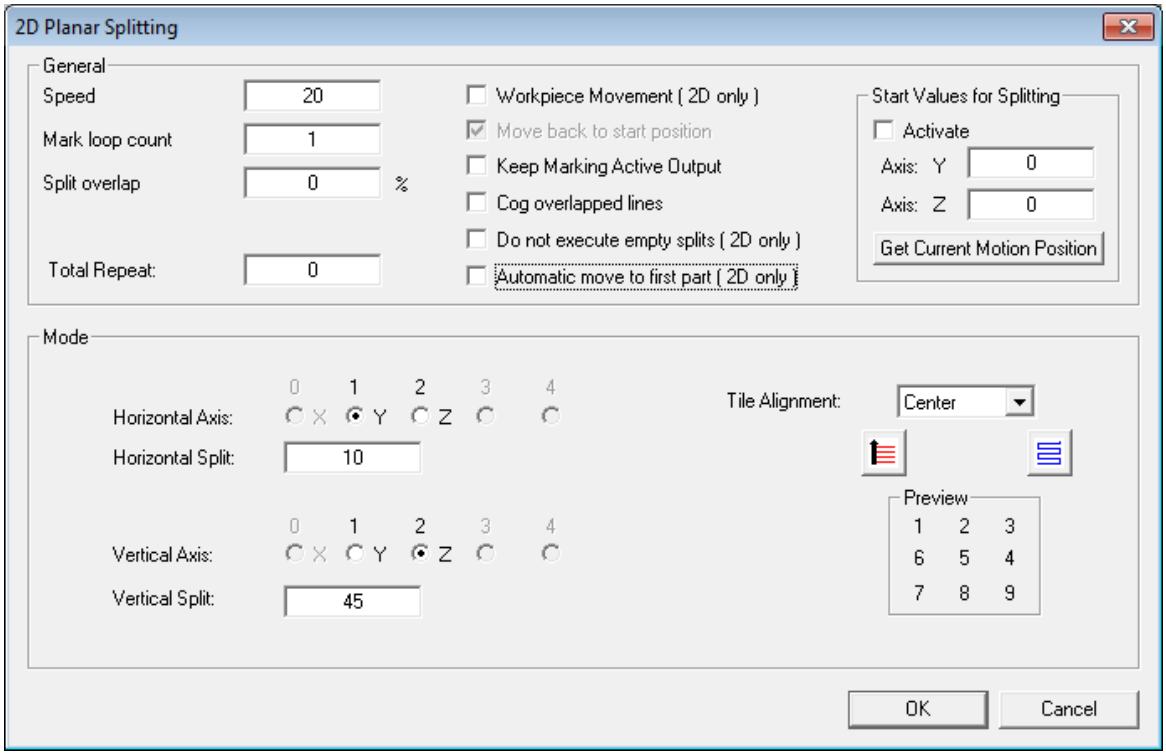


Figure 4.10: 2D Planar Splitting Dialog

General

The option **Speed** is a general movement setting for planar and angular operation modes. It defines the speed that is used for the connected drive. The speed that is defined here overrides the speed that is 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 splitted part is repeated before the motion device moves to the next splitted part.

Split overlap:
With this option one can increase the area of a single split.

Total Repeat:
If a value greater than 0 is entered here, then after finishing the split the scanner will move back to the start position and the split will be repeated as often as entered.

Workpiece Movement:
If not the scanhead is moving but the table on which the workpiece is placed is moving, then all relative movements have to be inverted.

If the option **Move back to start position** is unchecked the movement back to the starting position is not performed after a full split marking cycle. If else the software goes back to the position where the marking operation has started from.

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 activated then the marking active 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 object contains a hatch with many parallel lines then it is better for the quality of the marking, if those lines aren't splitted with a uniform edge but it is better if the lines are cogged. This is because if all lines are splitted on a uniform edge the workpiece becomes hot on that edge and the marking quality is reduced.

Do not execute empty splits:
If enabled then the scanner will not move to empty split regions but to the next region where a split has to be marked.

Automatic move to first part:
Provided that the split lies centered on the working area and that the motion axes are on position 0,0 then the user doesn't have to specify where the split should begin. Then the motion device moves automatically to the start position of the split.

Start Values for Splitting:
This option is available for Planar, 2D Planar and Angular Splitting mode. 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.

Mode

The 2D mode allows to split a 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 Axis: Here the drive that performs the horizontal movements can be defined. This axis must be a different one than the one used for the vertical axis.

Horizontal Split: Here the split size in horizontal direction can be defined. This size needs to be smaller than the size of the working areas in the same direction and defines the resulting tiles size in horizontal direction.

Vertical Axis: Using these buttons an existing drive axis can be defined to act as drive for the vertical movements. This axis must be a different one than the one used for the horizontal axis.

Vertical Split: Here the split size in vertical direction can be defined. This size needs to be smaller than the working areas size in the same direction and defines the resulting tiles size in vertical direction.

Tile Alignment: Here you can define where the marking on the scanner field is being performed. For example: If "Center" is selected then the splitted pieces will be marked on the center of the scanner field. PLEASE NOTE: if the tiles are positioned at a border of the working area you can not use a gain factor ›1 in that borders direction, because this would result in an "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 tiles. Together with these buttons a preview displays what order is currently choosen. Here the order of positions is equal to the numbers counting up in the preview area:

Here the splitting routine performs marking one column after another. It always goes back to the vertical start position so that the vertical movement direction during a split is always the same. This costs time depending on the velocity of the motion device.

Here 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.

4.1.4.2.4 1D Mark on the Fly

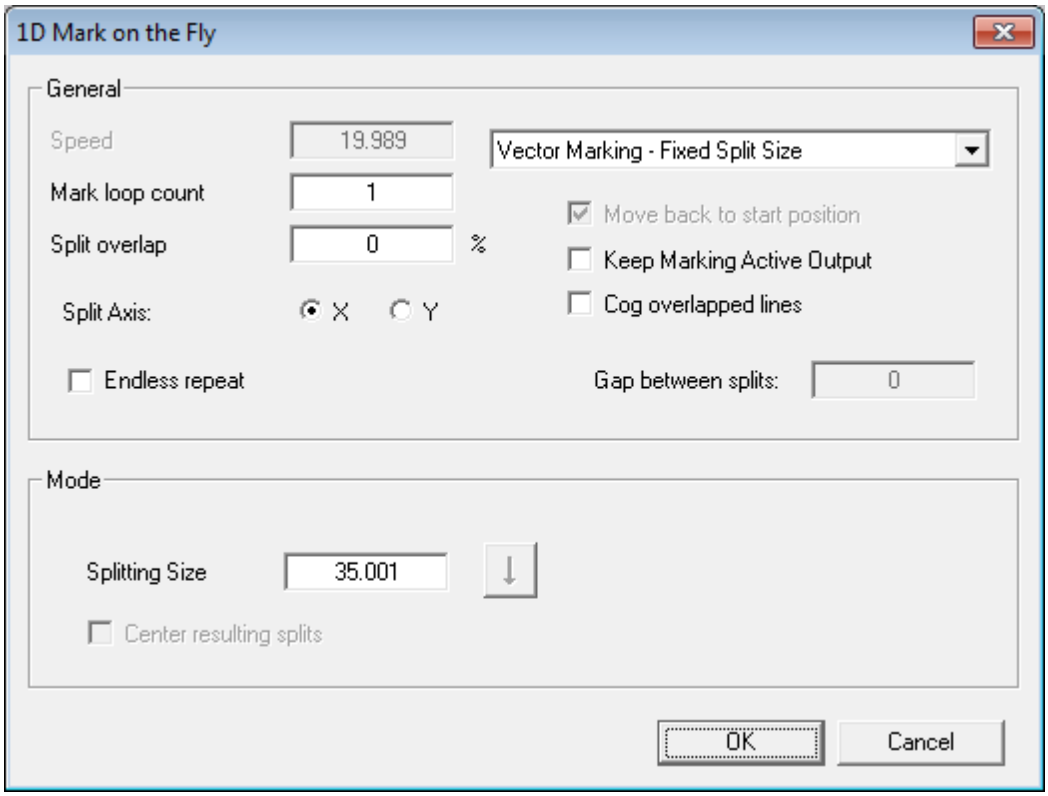


Figure 4.11: 1D Mark on the Fly Dialog

General

The option **Speed** is a general movement setting for planar and angular operation modes. It defines the speed that is used for the connected drive. The speed that is defined here overrides the speed that is defined in the Control submenu of the Entity Property Page. This option is ignored for the On-The-Fly mode, because here the application doesn't 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 splitted part is repeated before the motion device moves to the next splitted part.

Split overlap:
With this option one can increase the area of a single split.

Split Axis:
Here the axis that defines the splitting direction for the geometry has to be set.

Combo Box:
Possible Split options are:
- *Vector Marking - Fixed Split Size*
- *Vector Marking - Entity Based Splitting*
- *Vector Marking - Character Based Splitting*

The **Character Based Splitting** option is available only for the one-dimensional splitting modes. 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.

If the option **Move back to start position** is unchecked the movement back to the starting position is not performed after a full split marking cycle. If else the software goes back to the position where the marking operation has started from.

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 activated then the marking active 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 object contains a hatch with many parallel lines then it is better for the quality of the marking, if those lines aren't splitted with a uniform edge but it is better if the lines are cogged. This is because if all lines are splitted on a uniform edge the workpiece becomes hot on that edge and the marking quality is reduced.

Mode

This field is available only for USC cards and editable only if marking on the fly is enabled within the settings. This 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.

Splitting Size:
The split size that describes the size of one splitted piece.

4.1.4.2.5 Ring Spitting

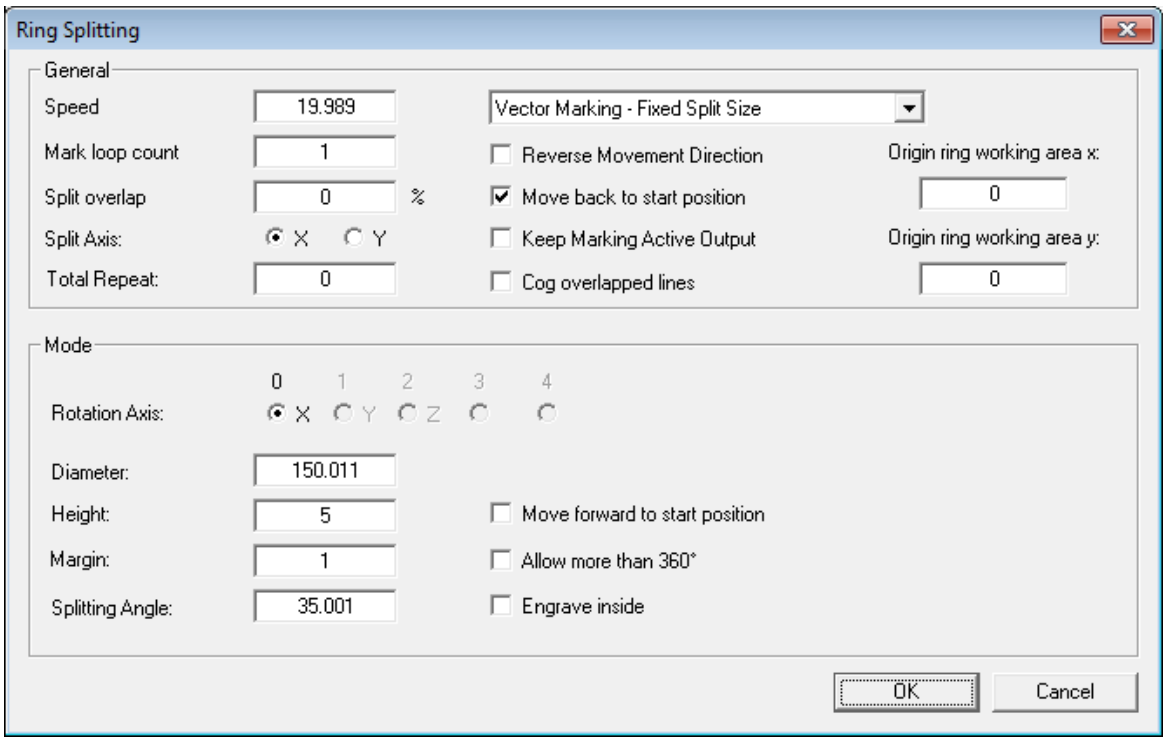


Figure 4.12: Ring Splitting Dialog

General

The option **Speed** is a general movement setting for planar and angular operation modes. It defines the speed that is used for the connected drive. The speed that is defined here overrides the speed that is 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 splitted part is repeated before the motion device moves to the next splitted part.

Split overlap:
With this option one can increase the area of a single split.

Split Axis:
Here the axis that defines the splitting direction for the geometry has to be set.

Total Repeat:
If a value greater than 0 is entered here, then after finishing the split the scanner will move back to the start position and the split will be repeated as often as entered.

Combo Box:
Possible Split options are:
- *Vector Marking - Fixed Spit Size*
- *Vector Marking - Entity Based Splitting*
- *Vector Marking - Character Based Splitting*
- *Bitmap Marking*

The **Character Based Splitting** option is available only for the one-dimensional splitting modes. 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 scanhead, all movements go in the opposite direction.

If the option **Move back to start position** is unchecked the movement back to the starting position is not performed after a full split marking cycle. If else the software goes back to the position where the marking operation has started from.

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 activated then the marking active 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 object contains a hatch with many parallel lines then it is better for the quality of the marking, if those lines aren't splitted with a uniform edge but it is better if the lines are cogged. This is because if all lines are splitted on a uniform edge the workpiece becomes hot on that edge and the marking quality is reduced.

4.1.4.3 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 MotionControl has to be done using the IO settings panel that can be found at the menu Settings->System. 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 MotionControl 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 the menu Extras -› Step / Repeat -› Settings.:

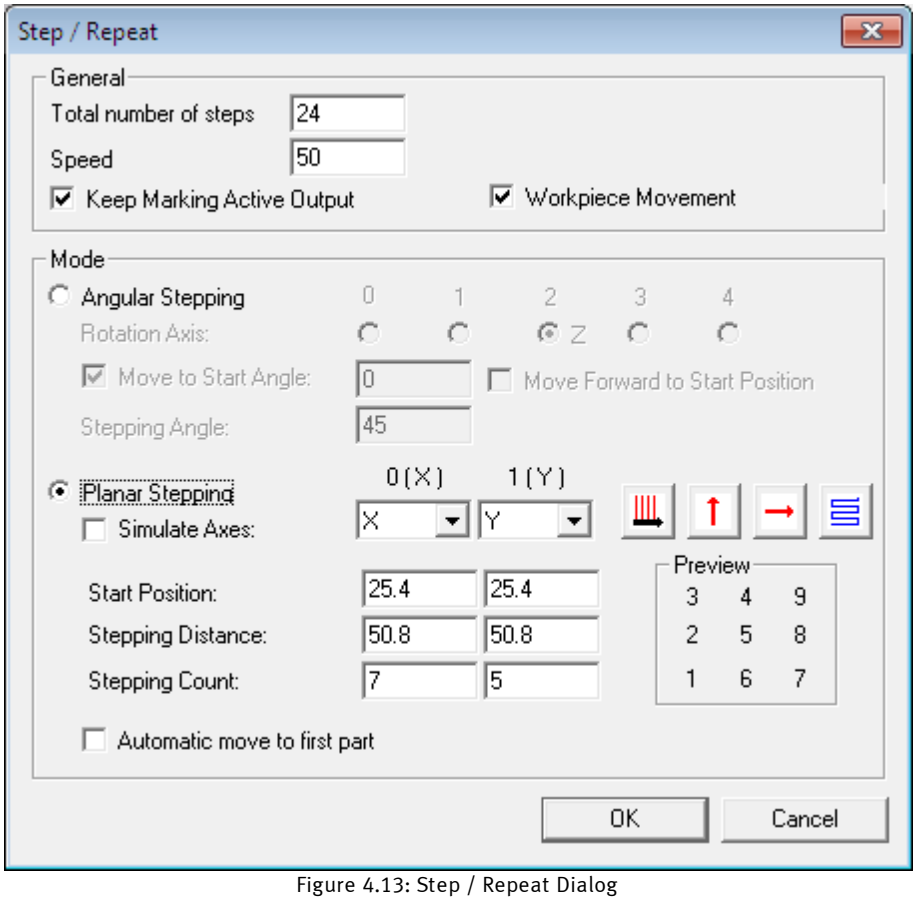


Figure 4.13: Step / Repeat Dialog

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 the job is marked and it is set back to 0 during the movement operation. This behavior can be changed by enabling this checkbox. If activated the marking active signal via digital output 0 would stay at 1 as long as the complete job sequence including all movements is marked.

Workpiece Movement:

If the scanhead isn't 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**)

The checkbox **Move to Start Angle** allows it to enable or disable the movement to the starting angle.

The checkbox **Move Forward to Start Position** forces 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:
- the starting position to which the motion controller moves to at the beginning of a full Step/Repeat marking cycle
- the distance the axes have to be moved in a defined direction for every repeated marking
- the number of steps that have to be performed

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.

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 the menu 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.
Between two marking steps a movement is performed conforming to the settings that have been made above (rotation by the **Stepping Angle** or movement using the defined stepping distances and directions). After the last marking cycle has been done, a last movement that moves back to the defined starting angle or position is performed.

4.1.4.4 Bitmap Marking

There is a special functionality available that allows it 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 supported motion driver and a connected motion drive
- modifies the bitmap automatically to fit to the drives resolution and the desired bitmap marking parameters; here the dither step value is an important parameter

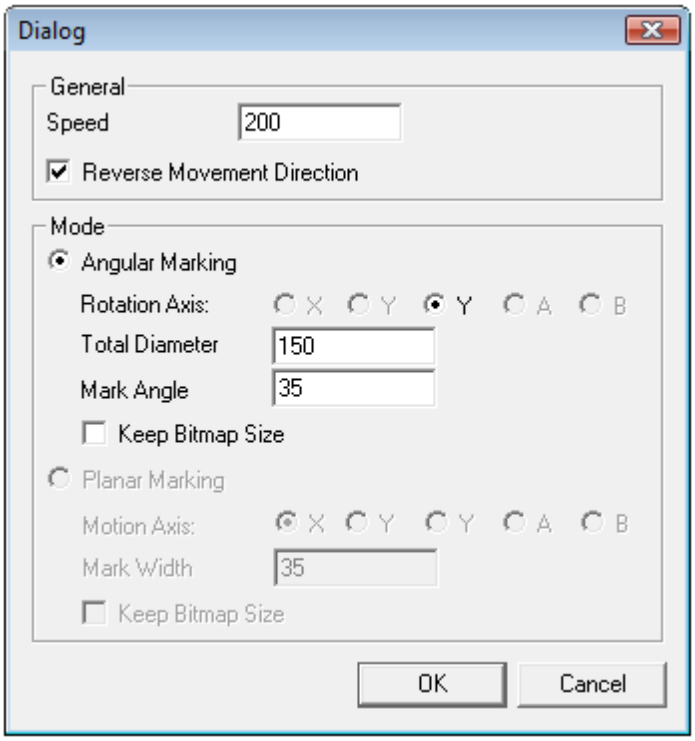


Figure 4.14: Bitmap Marking Dialog

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 bitmaps 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 calling the submenu "Enable".

Within the Continuous Bitmap Marking setup dialog mentioned above the following options can be found and configured:

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

Angular Marking

Rotation Axis - selects the axis of the drive that has to be used for the movement; this option is important in case there is more than one angular axis 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 with no gaps, in case the dither step of the bitmap is 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 exactly in case the option "Keep Bitmap Size" is not choosen

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

4.1.5 User

Menu->User->Login allows to login as a defined user. It is enabled with user password mode, see section User Level.

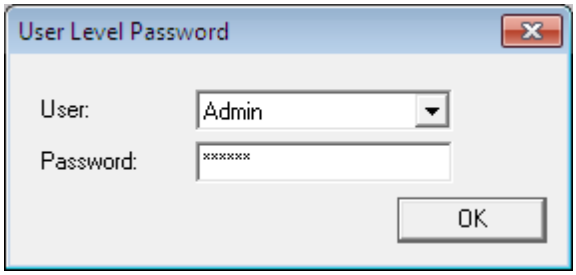


Figure 4.15: User Level Password Dialog

4.1.6 Settings

System:
Shows a dialog with settings pages for:

- View
- Optic
- Laser
- Shortkeys
- General

- Remote
- IO
- Card
- Extras
- User Level

Reset License:
When this menu item is selected the current software license is reset so that the software is unlicensed afterwards. This menu item should be used only in cases where the license needs to be replaced by a different one. After the next program start the application will ask for a license key, without entering a new (or of course the old one) the software can't be used - so please handle this function with care!

Units:
Allows to switch between the units 'mm', 'inch' and 'bit'.

4.1.6.1 Laser

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

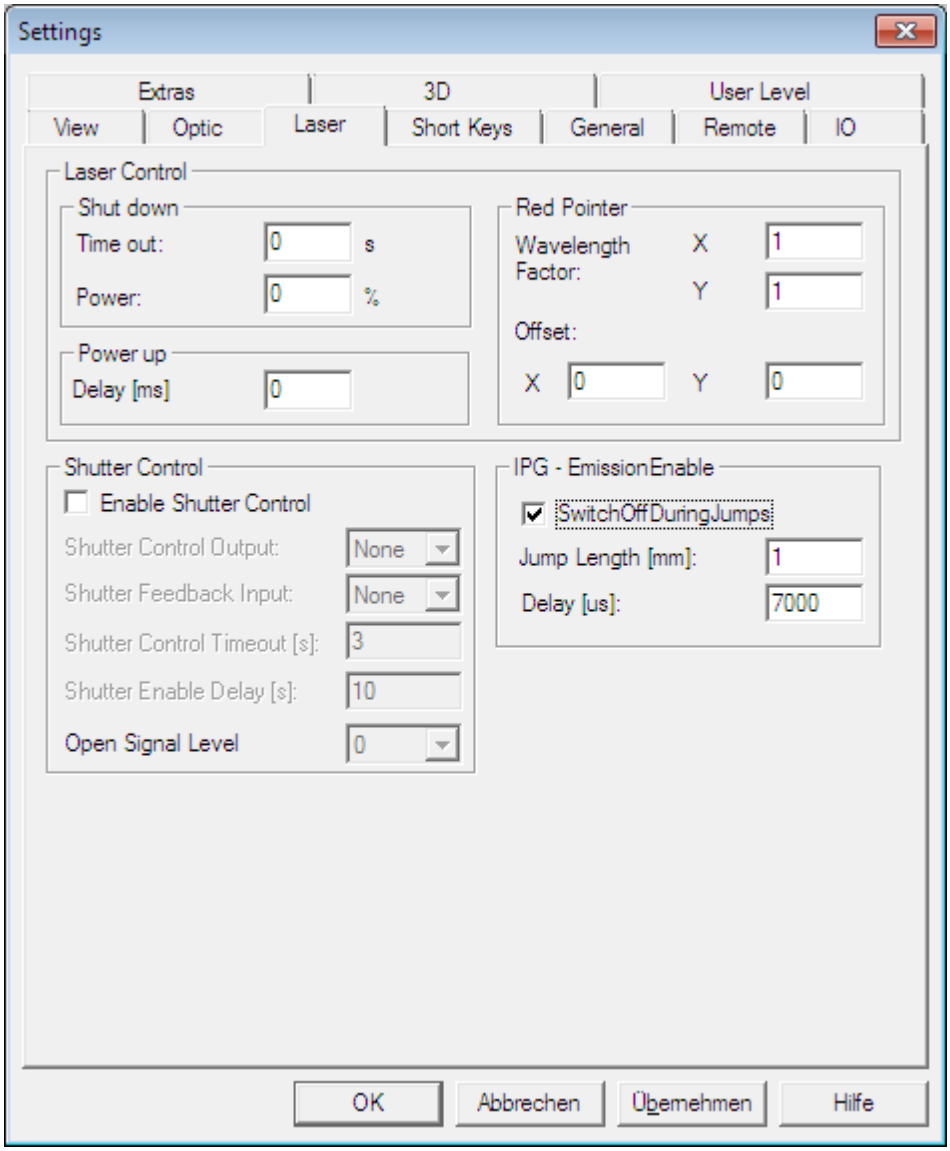


Figure 4.16: Laser Settings Dialog

Within the field "Laser Control" the following laser related parameters can be set:

Shut down

Time Out:
The Laser is being shut down after a defined time span in which the laser was not active. If a value of 0 is entered here the laser shutdown feature is disabled.

Power:
Percentage of power for the laser to be shut down after time out.

Power up

Delay:
Time delay for power up.

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 X = R in x direction / L in x direction,
Wavelength Factor Y = R in y direction / L in y direction.

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).

Within the field "Shutter Control" the configuration for an external shutter can be set:

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.

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.

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.

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 of EmissionEnable.

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

4.1.6.2 View

The following dialog can be reached by Menu item "Settings->System->View".

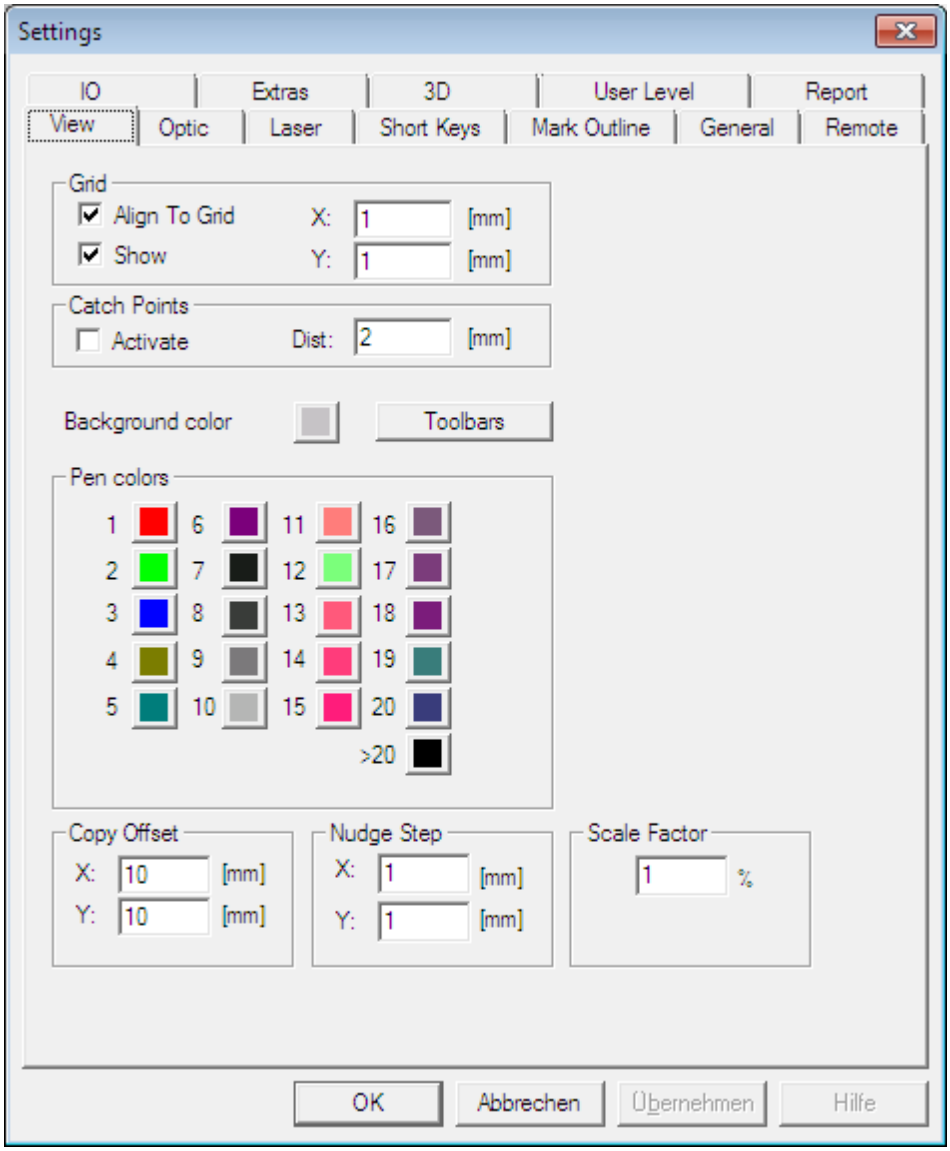


Figure 4.17: View Settings Dialog

Grid:

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

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

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

Button background colour:

Clicking on this button opens a colour dialog, where the background colour of the View 2D can be defined.

Button tool bars:

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 colour can be assigned to each pen. An object exposured by a special pen will be displayed with the assigned colour in the View2D. 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.
Example:	Select an object in the View2D 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, Down described in chapter Edit.

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

4.1.6.3 Shortkeys

The following dialog can be reached by Menu item "Settings->System->Short Keys".

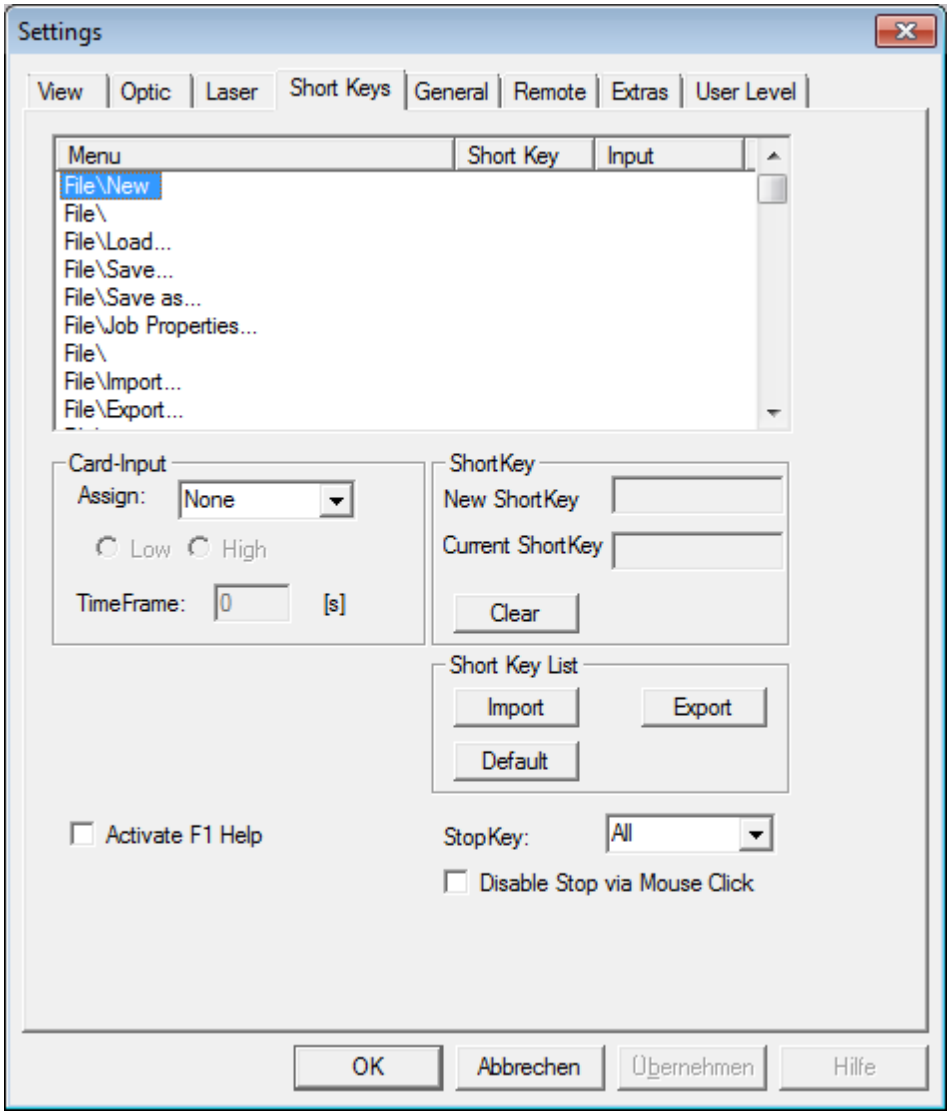


Figure 4.18: Short Keys Settings Dialog

Card-Input (only available with USC1):
To each short key a card input can be assigned, from IN1 up to IN6. 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.

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.

Note: 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. If *Disable Stop via Mouse Click* is enabled, marking can only stopped with the defined StopKeys.

4.1.6.4 General

The following dialog can be reached by the Menu item "Settings->System->General".

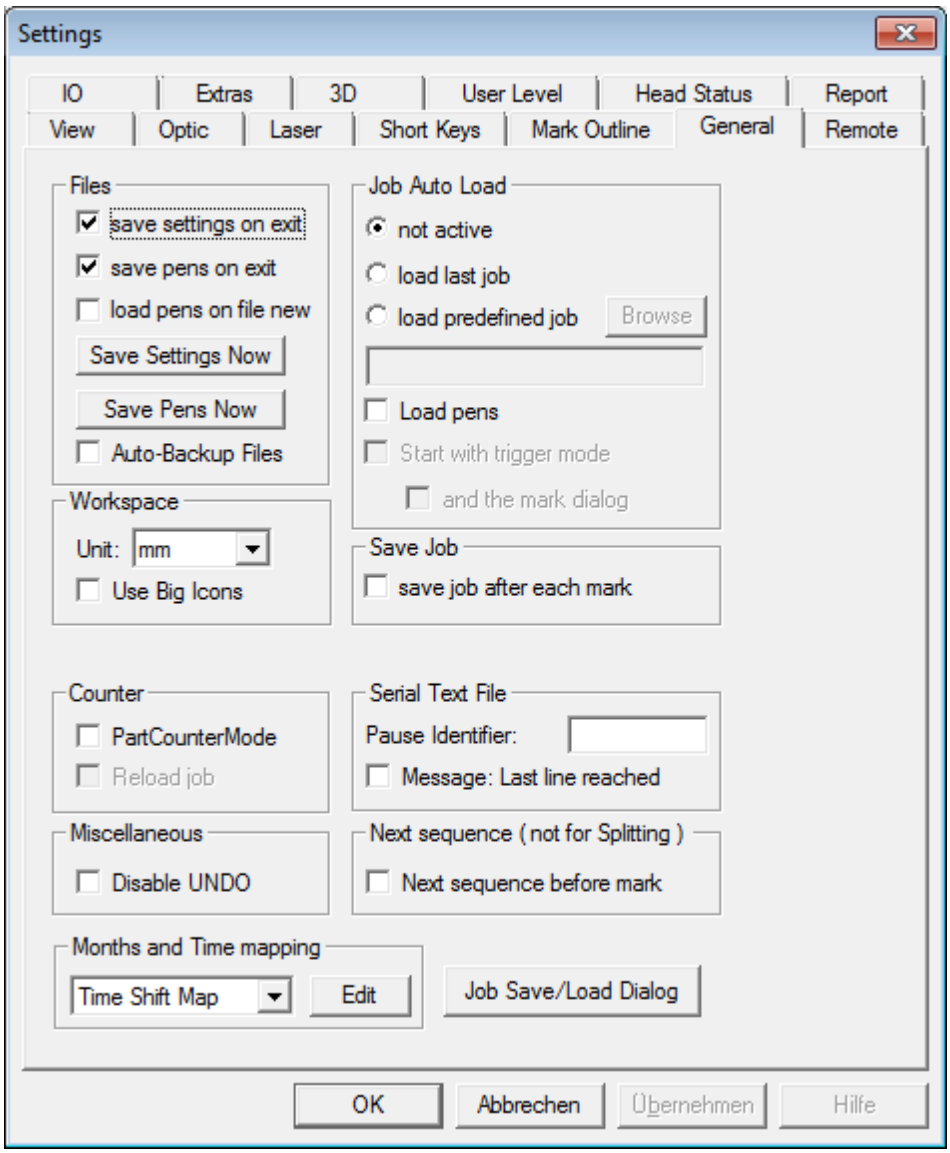


Figure 4.19: General Settings Dialog

Files Block:

Checkbuttons:

Save settings on exit: If checked the possibly modified settings will be saved on every program exit.
Save pens on exit: If checked the pens will be saved on every program exit.
Load pens on file new: If checked the pens are loaded by clicking on File->New.
Auto-Backup Files: Save-operations create 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 a .sjf file and can be loaded normally afterwards. The backup filenames consist of the original filename plus the backup number plus the new extension .sjb.

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

Job Auto Load Box:

Radio buttons:
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 pre-defined job or type in the absolute path to this file in the edit line below. The chosen file will be loaded at start up.

Checkbuttons (active only if "Load last job" or "Load predefined job" is activated):
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 on 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.

Workspace Box:
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.

Note: 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.

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.
Remark: 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.

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 isn't available. It is recommended to disable the UNDO-functionality in case of speed or memory problems on smaller computer systems (preferredly embedded systems which aren't 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 namable amounts of memory and calculation time.

Next sequence:

Next sequence before mark:
If this checkbox is selected then if starting the marking from the mark dialog at first the serial numbers in the job will be increased and then marking will start.

Date Time:

Shift Map: Time Shift Map: Opens a dialog where the working shift placeholder of date time can be mapped to working shifts times.
Months Map: The naming of the months can also be fully customized.
See: Chapter Shift Map.

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

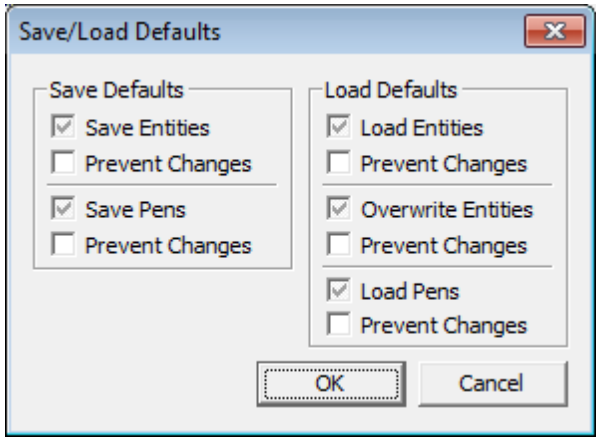
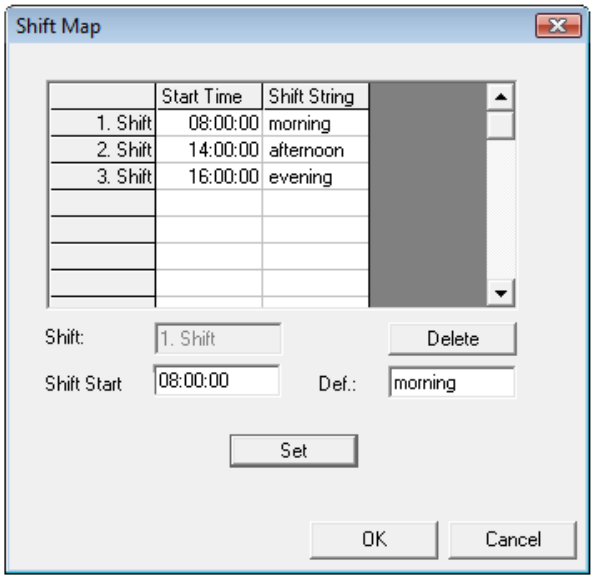


Figure 4.20: Save/Load Defaults Dialog

In each of the 5 fields there are 2 checkboxes. The upper checkbox has 3 states: On, Off, Grey. If Grey 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.

4.1.6.4.1 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->Settings->System->General->Shift Map.



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.

Figure 4.21: Shift Map Dialog

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.

4.1.6.4.2 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->Settings->System->General->Month Map.

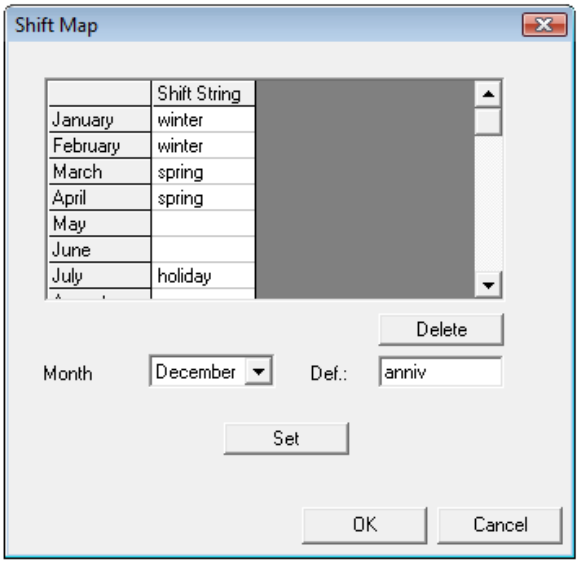


Figure 4.22: Months Map Dialog

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.

4.1.6.4.3 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->Settings->System->General->Day Map.

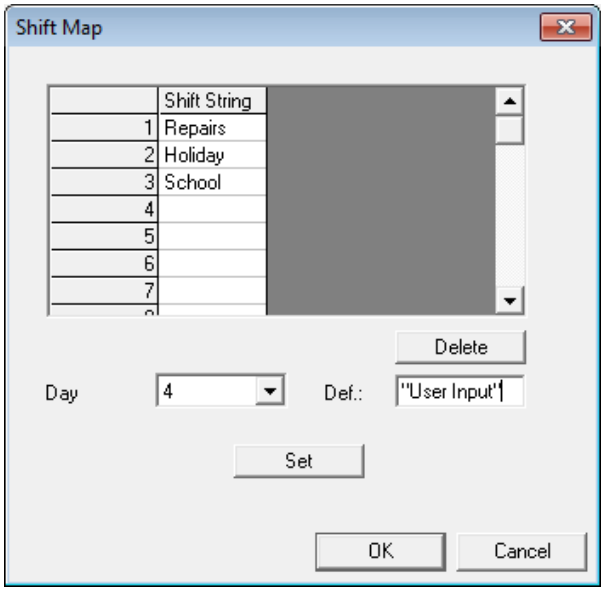


Figure 4.23: Day Map Dialog

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.

4.1.6.5 3D

ϟ Available with option SAM3D only.

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.

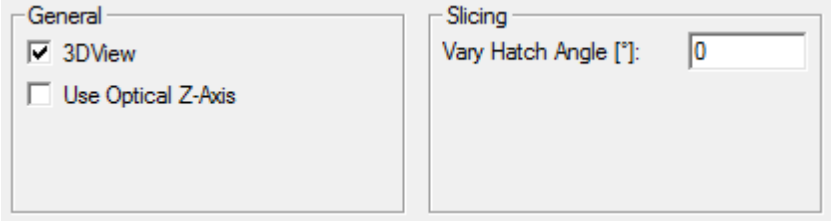


Figure 4.24: 3D Settings Dialog

General

3DView:
Enables the 3D view and 3D functionalities. To activate this change the software has to be restarted.

Use Optical Z-Axis:
Normally when marking a 3D object a motor moves the object after every slice so that the new slice can be marked. If this checkbox is enabled there will be no motion of the object with the motor but instead the focus of the scanner will be shifted accordingly to the 3D marking.

Slicing

Vary Hatch Angle:
Here an angle has to be entered that describes by how much degrees the hatch angle is varied for every slice. If this value is bigger than 0 the hatch-angle variation is enabled. That means for a variation angle of 13° that the first slice is hatched with a value of 0°, the second one with 13° the third one with 26° then using 39° and so on. If no angle is defined, all slices are hatched with the same value.

4.1.6.6 Extras

The settings dialog described here can be reached by selecting the menu item "Settings->System->Extras".

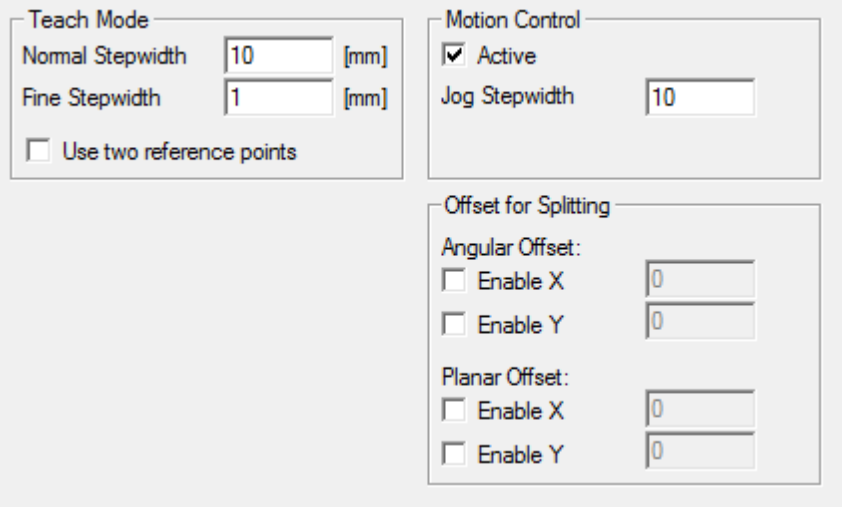


Figure 4.25: Extras Settings Dialog

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.

 Please note: 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. The only option that can be set here enables the usage of an motion controller.

Active:	With this feature for example an external motor can be controlled (a stepper motor or any other kind of the various supported types). After activating Motion Control the software has to be started again. Please refer to section Motion Control for some more information about how to configure and use an external motion controller.

4.1.6.7 Trigger

Only for RTC cards: A special trigger mode is available:

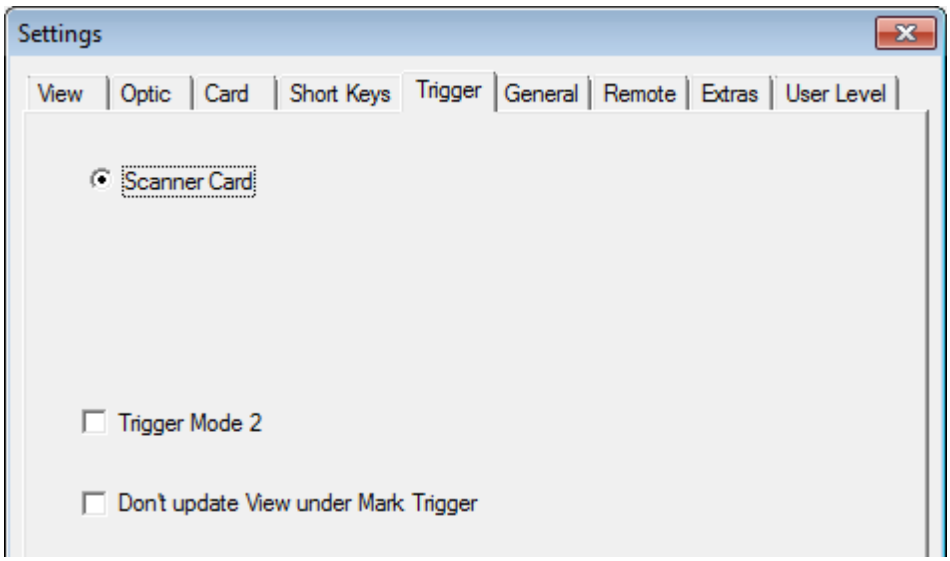


Figure 4.26: Trigger Dialog

Trigger Mode 2:
Serial numbers and Date Time objects are not being update between marking by trigger. This allows shorter delay between triggering - a faster marking.

4.1.6.8 User Level

The following dialog can be reached by Menu item "Settings -› System -› User Level".

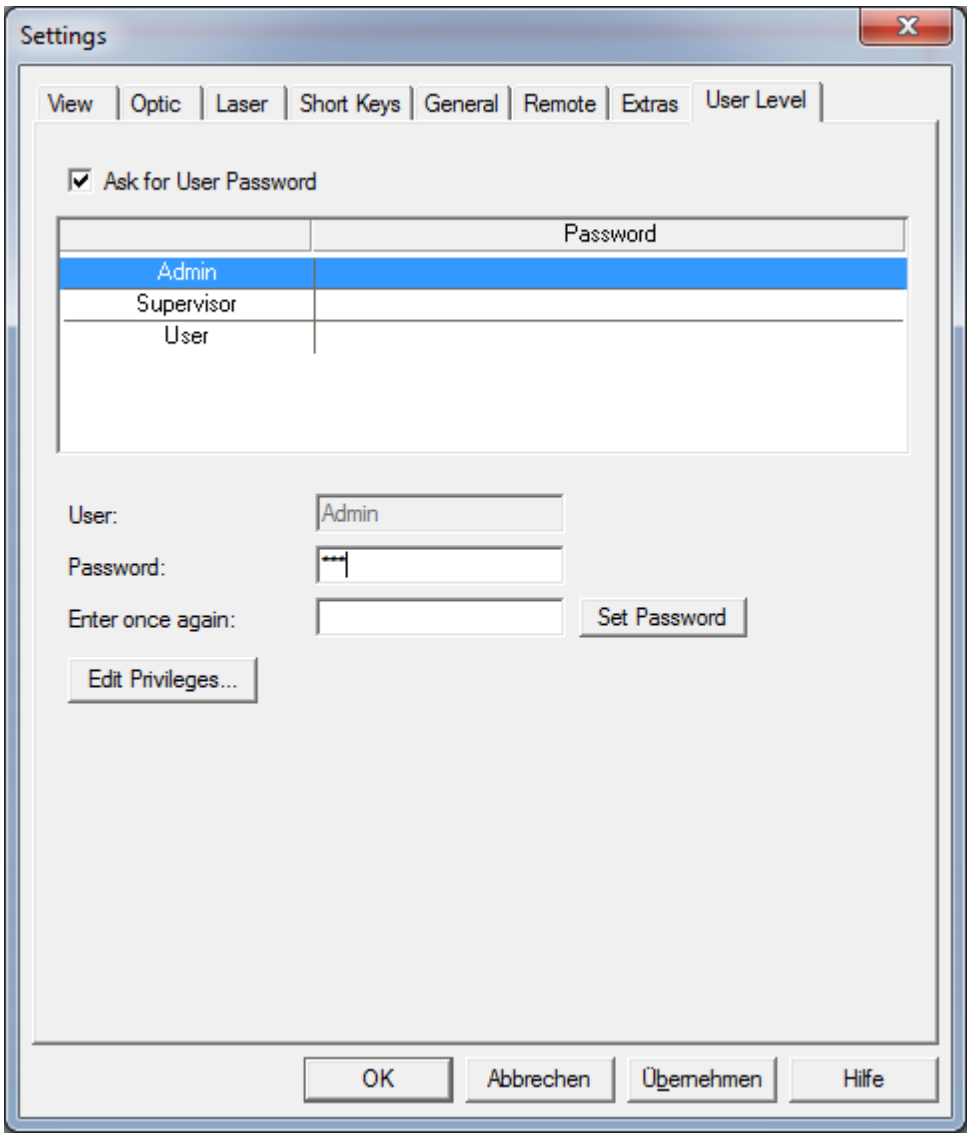


Figure 4.27: User Level Settings Dialog

Note: 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 Password:

Applies the password that is entered in the edit field to the selected user.

Edit Privileges:
Opens a dialog where it is possible to for define access user rights.

Remark:
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.

4.1.6.8.1 Access Rights

The following dialog opens by clicking on the "Rights"-Button on the User Level page.

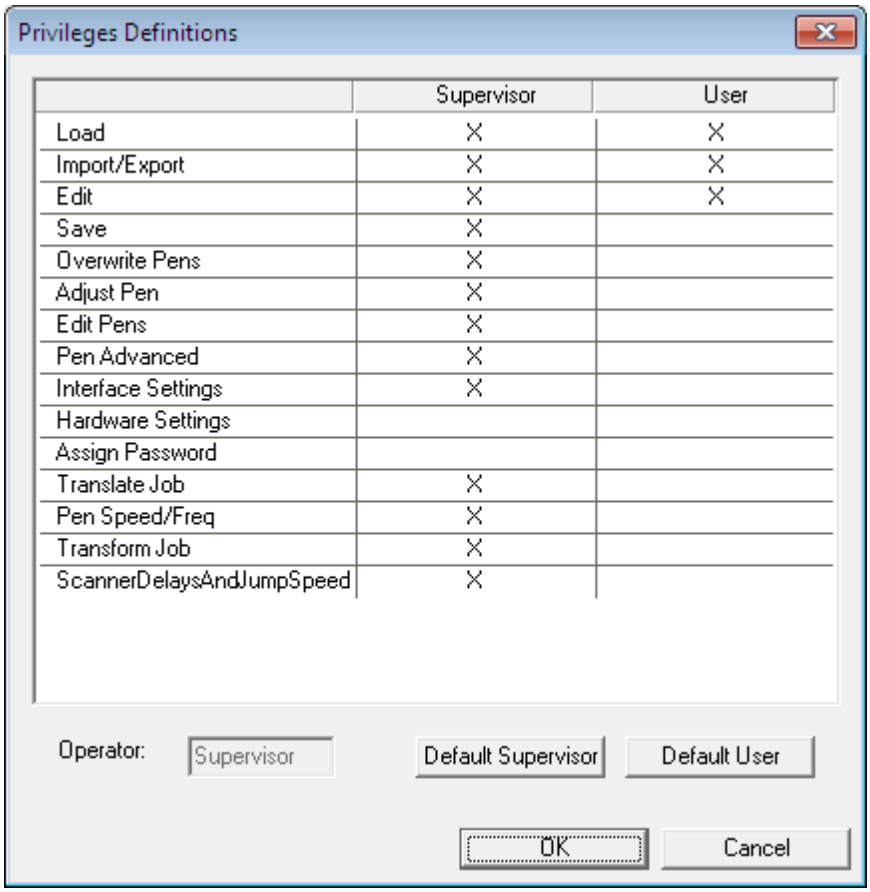


Figure 4.28: Access Rights Dialog

The features are 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 dialog: View, ShortKeys, General, Extras
Hardware Settings:	Enables following property pages of Settings->System dialog: Optic, Laser, IO-Settings, Remote
Assign Password:	Enables property page User Level of Settings->System dialog.
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.
ScannerDelaysAnd JumpSpeed:	Allows to change Scanner Delays and Jump Speed if Pen Speed/Freq. is activated also.

4.1.6.9 Optic

These settings are described in section Optic / Optic Settings.

4.1.6.10 Card

These settings are described in section Optic / Card Settings.

4.1.6.11 IO

The settings dialog described here can be reached by selecting the menu item "Settings->System->IO".

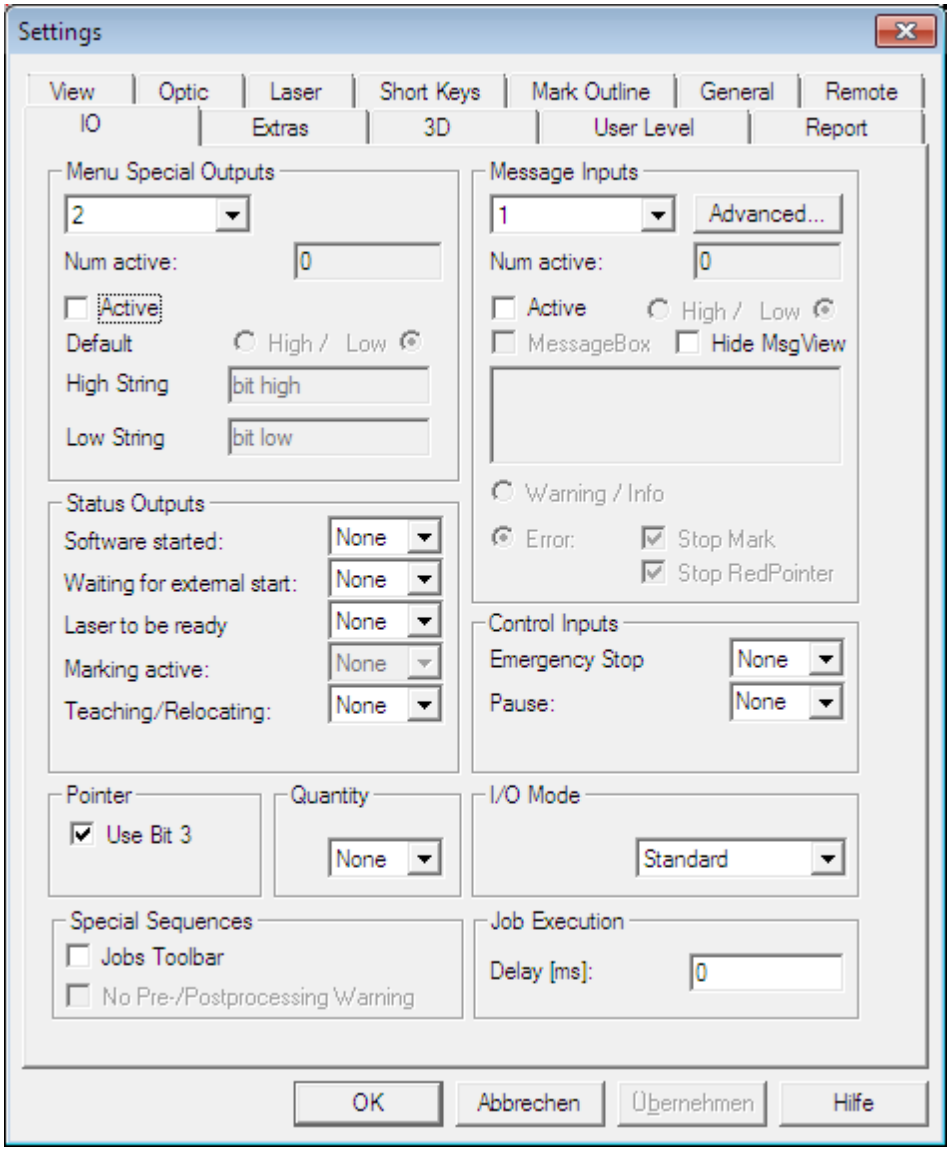
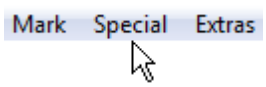


Figure 4.29: IO Settings Dialog

Menu Special Outputs

It is possible to insert new menu items for switching IO's 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* box is activated a string which indicates the current state of the Bit can be defined. Under the following menu the menu items for switching the IOs on and off will be inserted:



Status Outputs of the IO-Port

This block defines state IOs that can be switched on an off according to specific program and usage actions. Using the comboboxes 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. Attention: this stays HIGH also 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.
Teaching / Relocating:	If an output pin is defined here it is switched on every time the teaching or relocating dialog is active.

Pointer

Use Bit 3:
Bit 3 of the IO-Port is used to indicate that the red pointer is active.

Invert:
Only for RTC cards: 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.

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 not** checked, the user is asked before a job is executed during program startup or exit. That is necessary due to security reasons because such a job could contain dangerous movements or laser operations. 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!

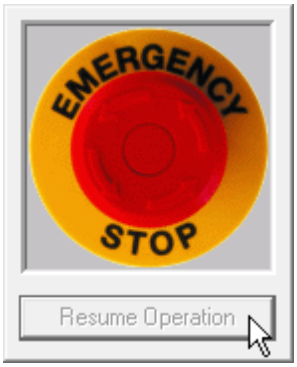
Message Inputs
The IO-Port provides 6 input Bits. Within this dialog the input Bits 3 to 6 can be defined to cause a message output. The selected Bit must be activated to send the defined message by the *Active* check box. The message appears if the selected Bit is either high or low according to the selected radio button. An error or warning message is displayed in the messageView dialog as well as in the second statusbar pane. If 'Hide MsgView' is activated, messageView is switched to invisible.

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.

Please note: 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.



Inhibit Laser:
Using this option it is possible to create some kind of additional, external laser modulation. An input can be chosen that disables the laser as long as this input is set. That means all movements of the scanhead are performed like before and the marking process is not influenced in that way. But the laser on signal of the scanner card is turned on or off depending on the state of the selected input.

Note: This mode works with trigger mode (Menu->Mark->Trigger) only!

Pause:
This functionality is only available with an USC-2 card. Here you can select an Input Bit that will halt the job if it goes to high while the job is executed and if the laser is switched off. The laser is switched off after a mark command or after a PolyEnd or during a jump command.

I/O Mode
There are two I/O modes that can be selected. In mode **Standard** the settings described above can be made including the freely definable Message Inputs. These inputs are disabled in **Job Select** mode. Here the input signals are used to select and load a job externally triggered. Only for USC-2: **JobSelect USC-2 Ext:** You can use the Digital Input Pins DIn0...DIn7 of the USC-2 card to select the jobs. For more information see Jobselect Mode.

Job Execution
Defines an execution delay which is the time between a mark output signal is given and the execution.

4.1.7 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.

4.1.8 Help

Contents:
Opens the help window.

About...:
Shows an information window with the version number.

4.2 Tool Bars

The following tool bars are available:

- File Tool Bar
- Camera Tool Bar
- View Level Tool Bar
- Geometry Object Tool Bar
- Functionality Object Tool Bar
- Align and Spacing Tool Bar
- Extras Tool Bar

The tool bars can be activated/deactivated in Menu bar->System->Settings->View:
Clicking on the button "Tool bars" will open the following dialog:

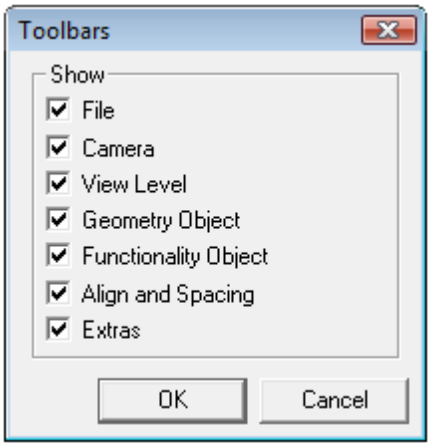


Figure 4.30: Select Toolbars Dialog

Checking/Unchecking an item will activate/deactivate the corresponding tool bar.

- Special Sequences Tool Bar

4.2.1 File Tool Bar

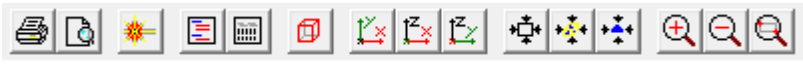
The IO tool bar provides the following shortcuts (from left to right):

- Load an existing jobfile (*.sjf). See chapter Job Format.
- Save the actual job. See chapter Job Format.
- Delete an item from the Entity List.
- Info. This opens the AboutBox.
- Help. If this button is activated you will get context related help for several dialogs and controls by clicking on it.

4.2.2 View Level Tool Bar

The View Level tool bar 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.

4.2.3 Camera Tool Bar




Print
Prints the current View 2D.This function works only if a printer is installed. See chapter View 2D.


Print preview
Shows a print preview of the current View 2D. This function works only if a printer is installed. See chapter View 2D.


Mark
Opens the Mark Dialog.


ShowEntityList
Shows and hides the EntityList on the left side of the View 2D. 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.

 **Plan View xy**
Clicking on this button changes the perspective of the View 2D to plain view. This feature is available only with Optic 3D.

 **Side View xz**
Clicking on this button changes the perspective of the View 2D to side view. This feature is available only with Optic 3D.

 **Side View yz**
Clicking on this button changes the perspective of the View 2D 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 the 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.

4.2.4 Functionality Object Tool Bar

 **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.

 **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".

 **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.

 **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.

4.2.4.1 Data Wizard

The following dialog appears after pressing the magic wand in the object toolbar. All operations are done on the selected polylines.

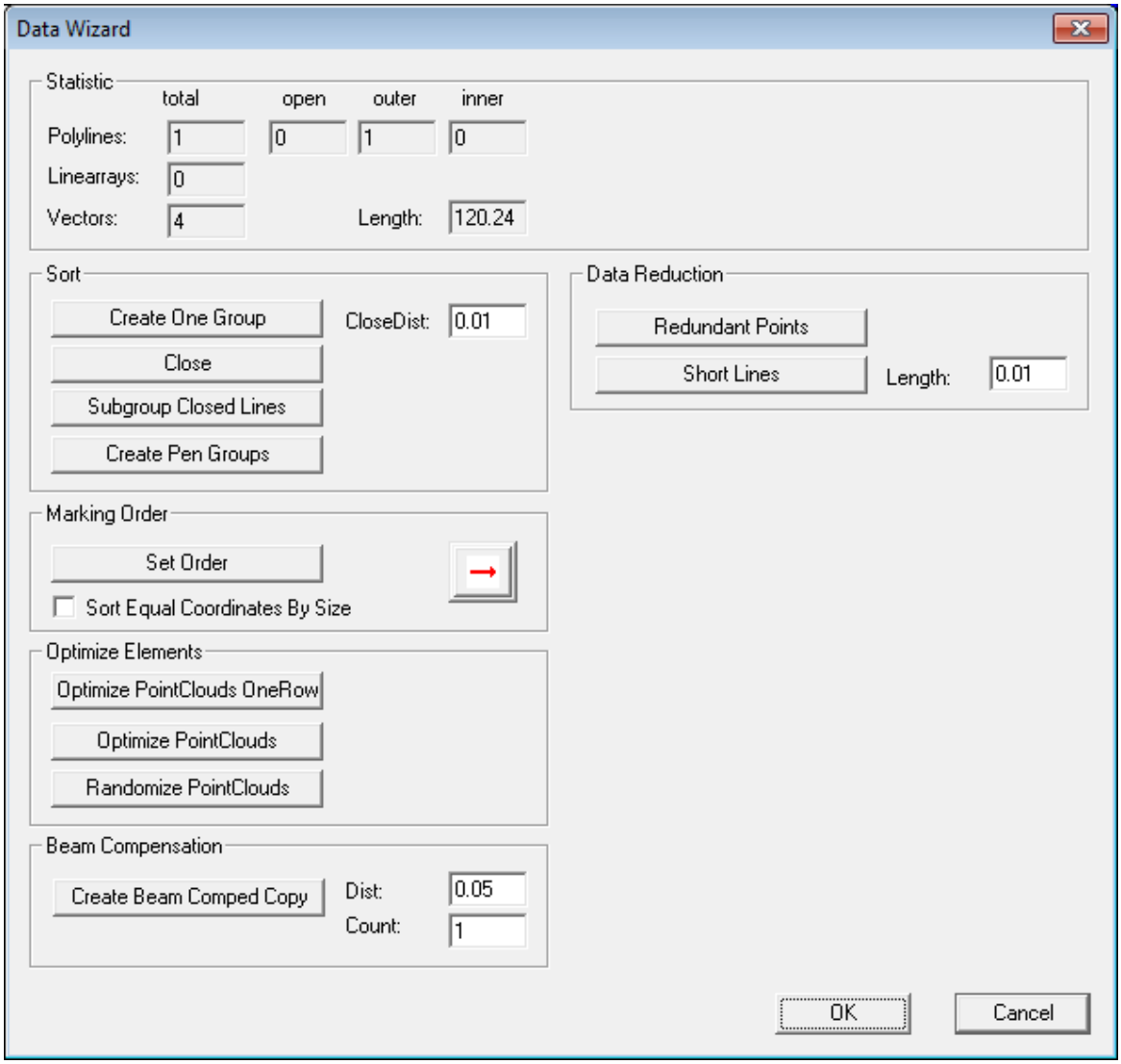


Figure 4.31: Data Wizard Dialog

Statistic:
total: Number of all objects
outer: Objects which are orientated contra clockwise.
inner: Objects which are orientated clockwise.
Length: Total Length in mm

Sort:
Create One Group
Groups all selected objects directly into one Layer. Additionally sorts polyline points, closes open points that are smaller than given CloseDist and merges to one polyline if possible.

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.

Marking Order:
Set Order
Sorts entities inside one group according to the marking direction. When the option "Sort Equal Coordinates By Size" is selected, the sort algorithm behaves a little bit different: Here the additional sort condition "size" is used when polylines start at the same coordinates (according to the marking direction). If this option is not set the order of the polylines is left unchanged in such cases.

Optimize Elements:
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:
Clicking the button Create Beam Comped Copy 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.

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 does describe the distance of the points..

Note: 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.

4.2.5 Geometry Object Tool Bar

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.

4.2.6 Alignment and Spacing

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.

4.2.7 Extras Tool Bar

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 and it has to be activated within the settings before it can be used. It offers following functionalities:

The drop down menu allows 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 dialogue 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 dialogue for the Step/Repeat parameters.

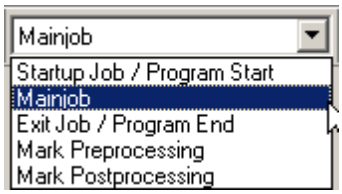
Reset Position

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.

4.2.8 Special Sequences



The Special Sequences Toolbar can be made visible if you enable the checkbox "Jobs Toolbar" under "Special Sequences" in the menu "Settings->System->IO". This Toolbar allows it 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 dialogue opens where the elements of that sequence can be edited:

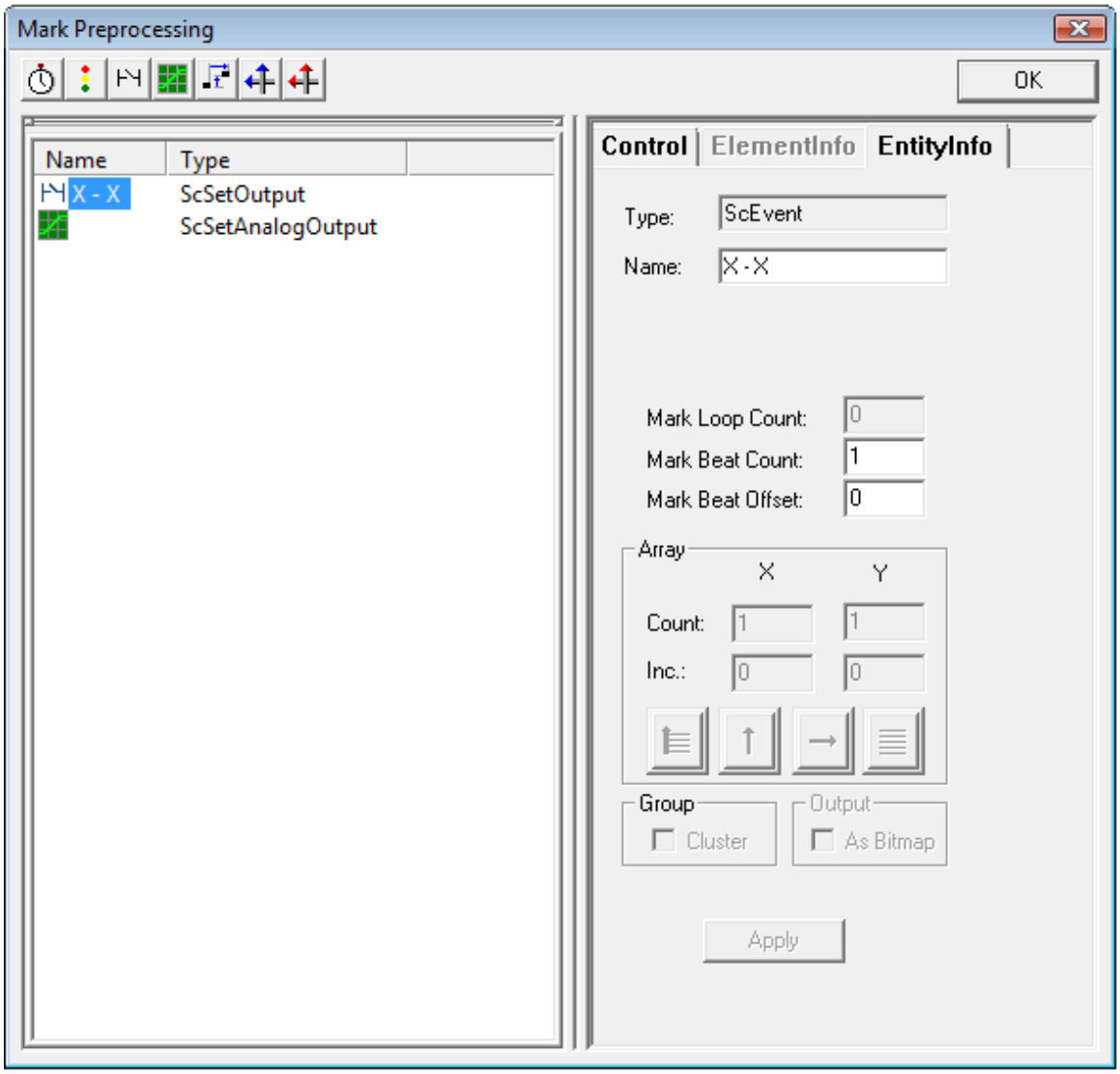


Figure 4.32: Mark Preprocessing Dialog

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:

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.

Startup Job / Preprocessing

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.

Exit Job / Postprocessing

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.

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. Please note: 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.

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.

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.

4.3 Main Window

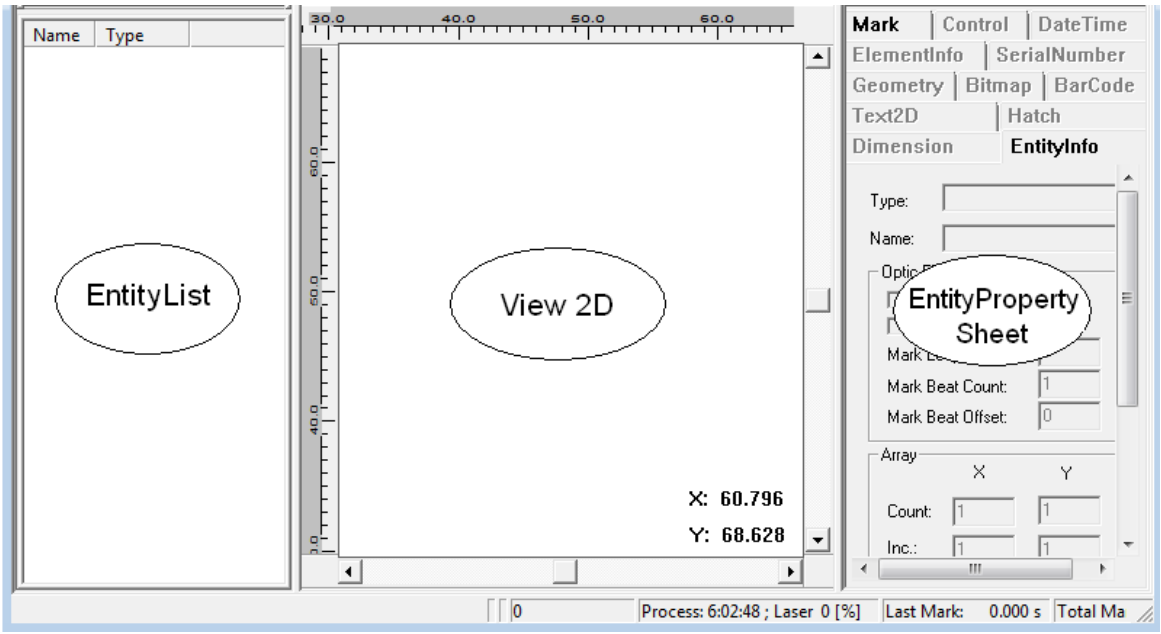


Figure 4.33: Main Window

The main window is separated into three parts (from left to right):

- Entity List
- View 2D
- Entity Property Sheet

4.3.1 Entity List

The Entity List appears in two different modes:

1. Entity List
2. 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.

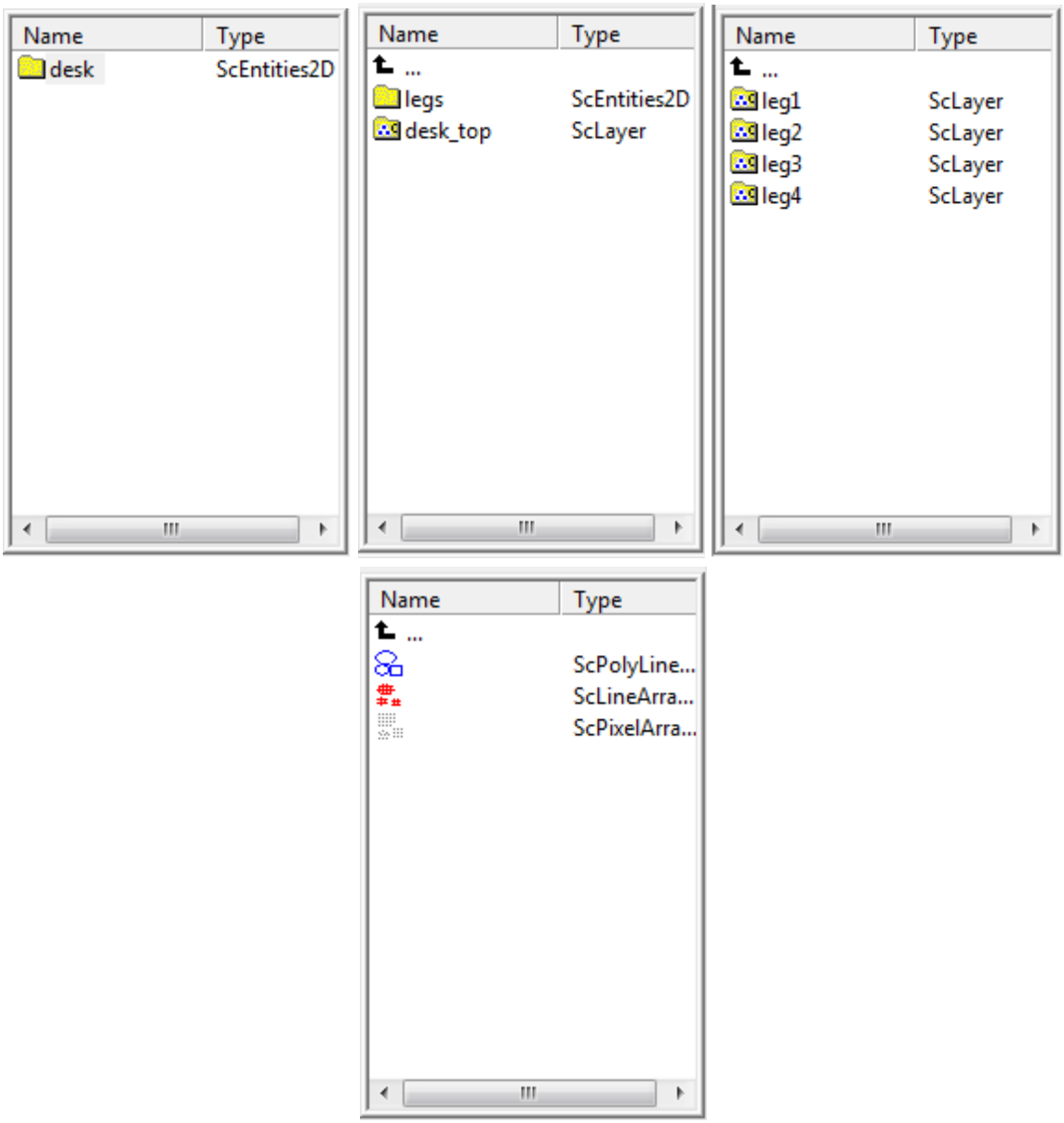
4.3.1.1 Entity List

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 of the notion "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 think of 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.



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" as a sub-entity of "desk" is on 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 SHIFT-key is hold down

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 keeping the Shift key pressed 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 a popup menu appears that offers the following functionalities:

NumberEntities:
All Entities, regardless if they are selected or not, are being named with a number starting from 1.

NumberEntities with pre-/postfix:
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:
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".

SortByName:
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".

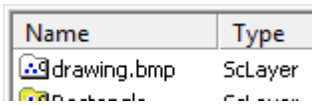
Remark to changing the order of the entities:
The entities of a group will always be marked from the top first to the bottom last in the ListView. So changing the order in the ListView can be used to change the order of exposure during the mark process.

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. Please note: for this operations the same restrictions are valid than described for the Drag-and-drop-copy operation described above.

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.



Hide-/Show in View 2D

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 View2D 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.

(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

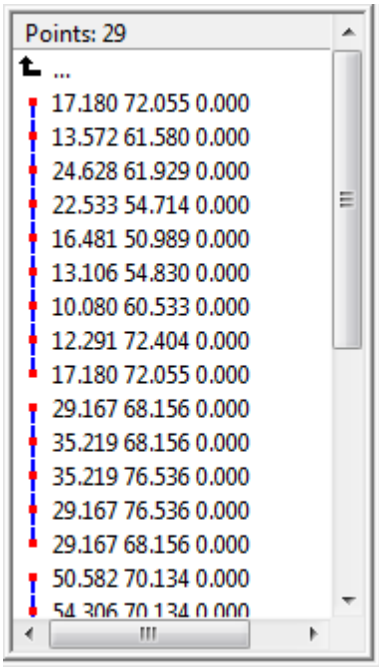
This can be used to exclude the selected entities from the marking process. By selecting again the selected entities are being included again.

4.3.1.2 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.



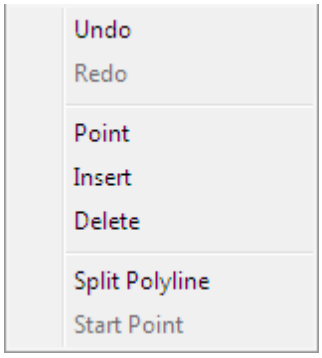
The Point Editor View shows all points of the selected objects in a single list.

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.

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 key pressed. After the selection keep the Ctrl key 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.

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.

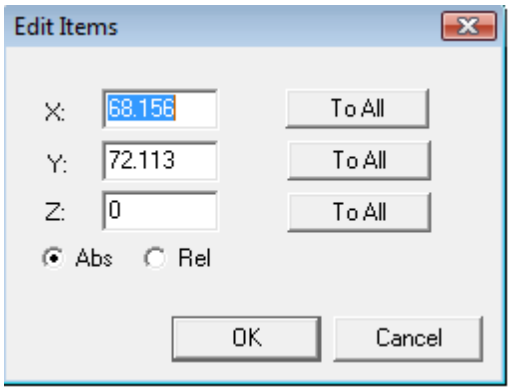


Figure 4.34: Edit Point Coordinates Dialog

X,Y,Z:

Edit these values to change the coordinates of a single point. Remark: Z coordinate is only available with Optic3D.

Radio buttons Abs, Rel:

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.

Cancel Button:

Cancels the dialog. No update of point coordinates will be done.

OK Button:

Applies the update to the point or points and ends the dialog. Remark: If more than one point is selected the update is performed on all selected points. That means a PolyLine will collapse to one point, which is probably not desirable. But some point update operations make sense to apply them to all selected points. Use the ToAll - Buttons for such operations.

ToAll - Buttons:

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 ToAll - 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.

4.3.2 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.

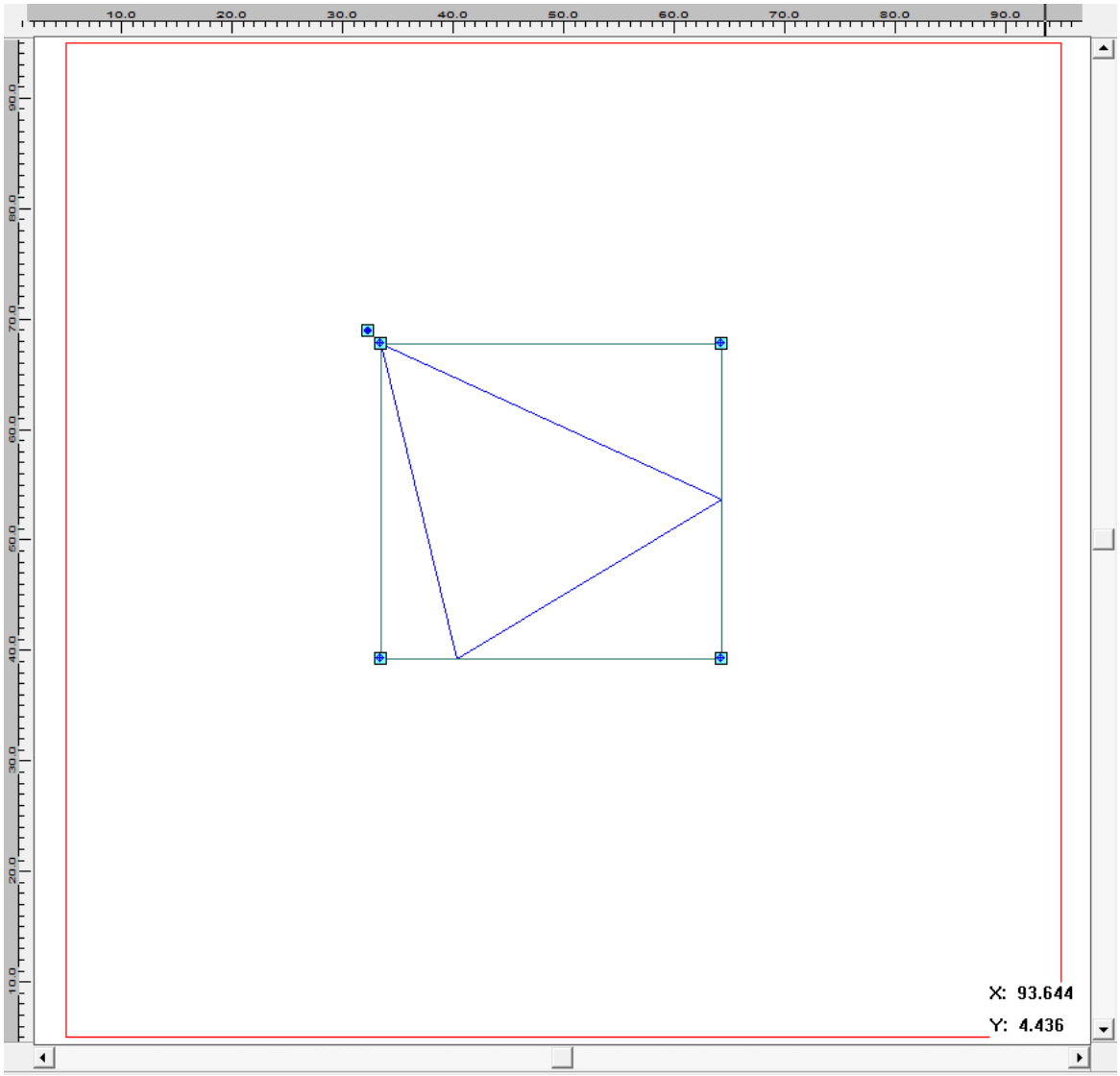


Figure 4.35: View 2D

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.

Remark: For more details see chapter Operations and Print Preview.

4.3.2.1 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 keep the Ctrl key pressed 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, keep the Ctrl key pressed 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.

1. Translate 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.

2. Scale 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.

3. Rotate and slant 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 slant the object click on an edge symbol and do the same.

The manipulations described above are standard for all objects and 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.

4.3.2.2 Print Preview

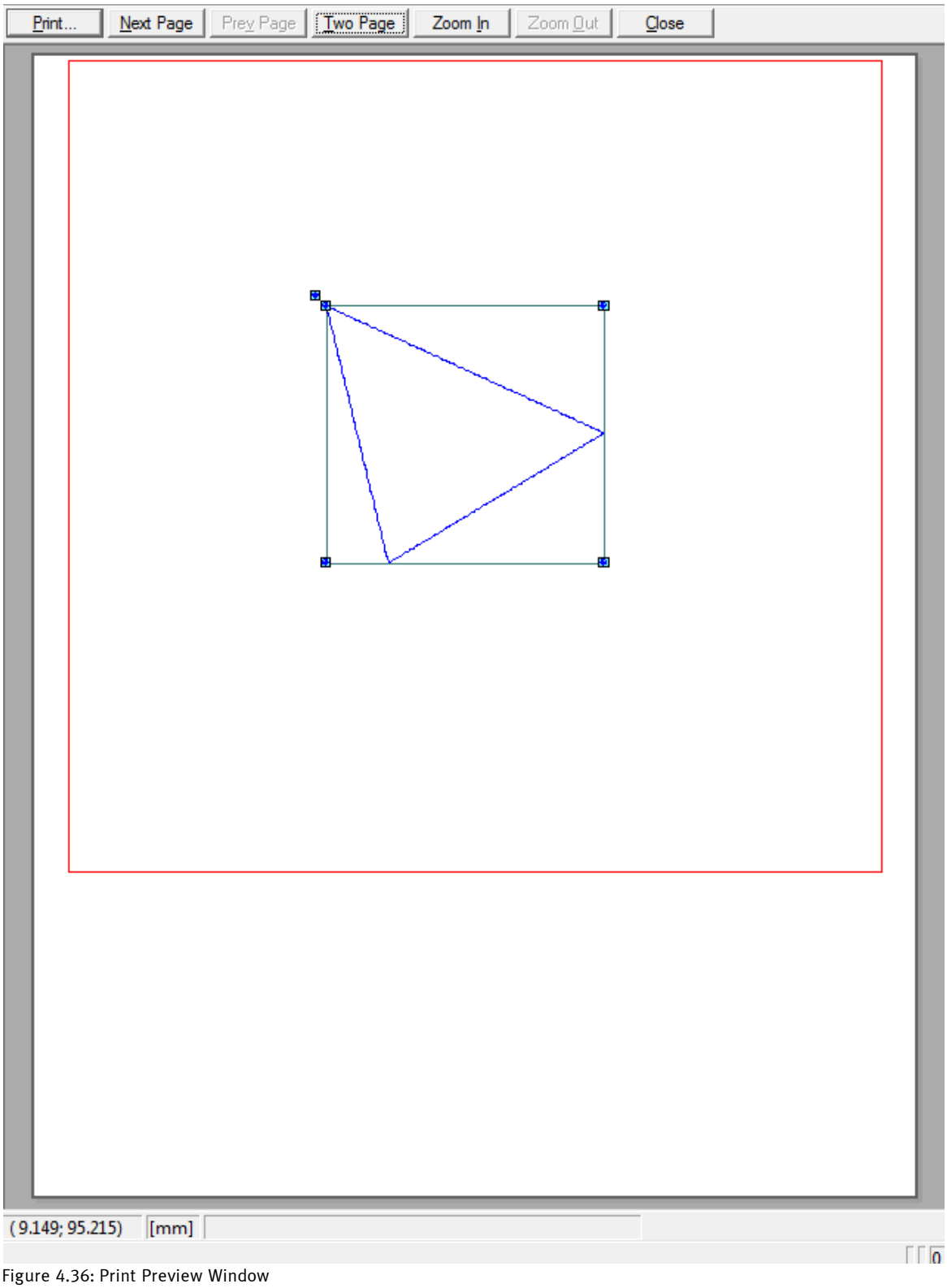


Figure 4.36: Print Preview Window

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.

4.3.3 Entity Property Sheet

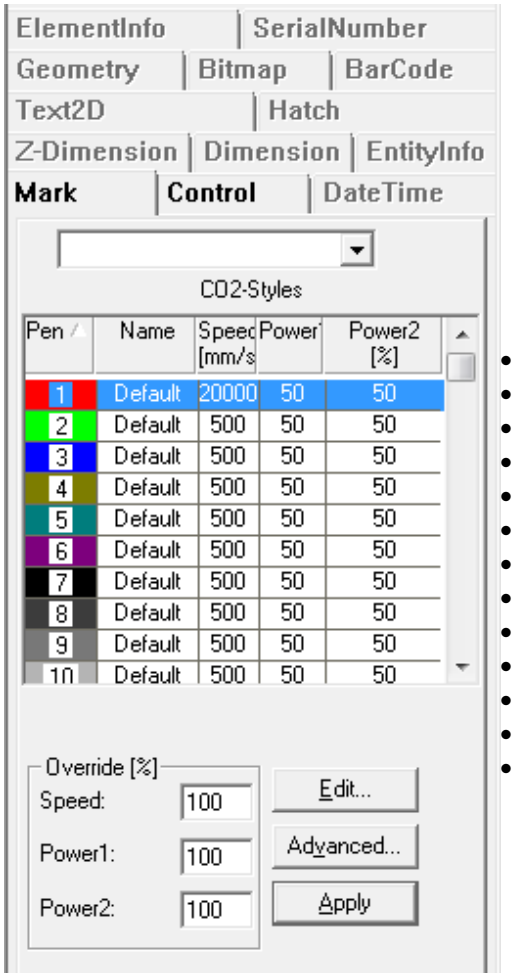


Figure 4.37: Entity Property Sheet

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:

- Mark
- Control
- DateTime
- SerialNumber
- Geometry
- Bitmap
- BarCode
- Text2D
- Hatch
- Z-Dimension
- Dimension
- EntityInfo
- ElementInfo

Remark: The Property Page Z-Dimension is only available with Optic3D.

4.4 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.

Color	Line Type
red	jump
green	mark, i.e. the line has laser on/laser off signal
yellow	start line of a PolyLine
cyan	end line of a PolyLine
violet	normal line
white	actual displayed

It also shows the outlines of the field and the working area.

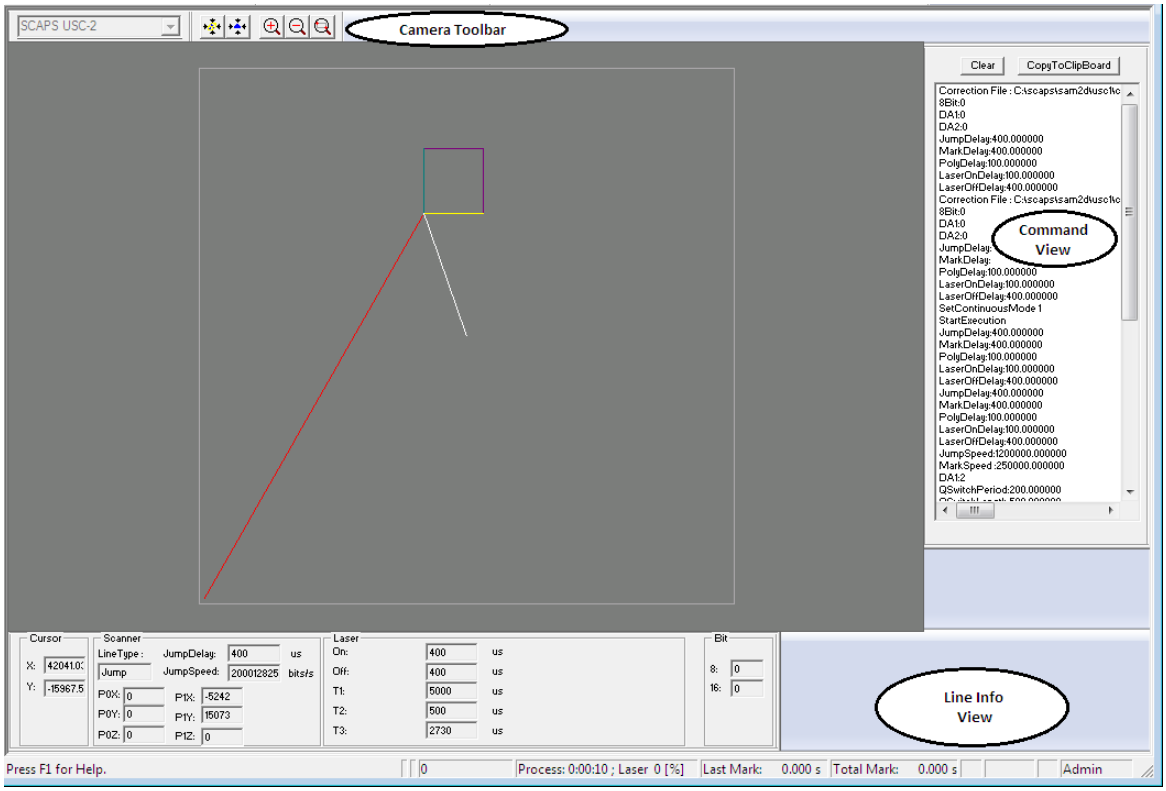


Figure 4.38: Mark Preview Window

Toolbars:

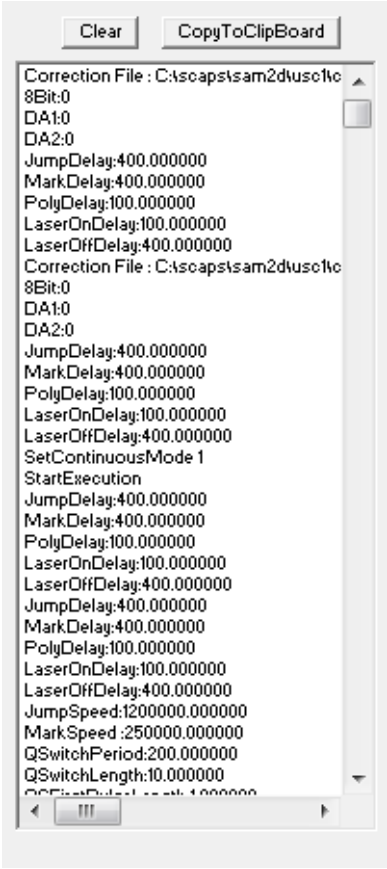
The Preview Window has its own toolbar, because the Standard Toolbars belong to the Main Window and are not active while the Preview Window is in the foreground. The Camera Toolbar is just the "light" version of the Main Window's Camera Toolbar. So refer to chapter Camera Toolbar.

Views:

Command View

Line Info View

4.4.1 Command View



The Command View displays the first 1000 commands that are being sent to the preview window in a list box.

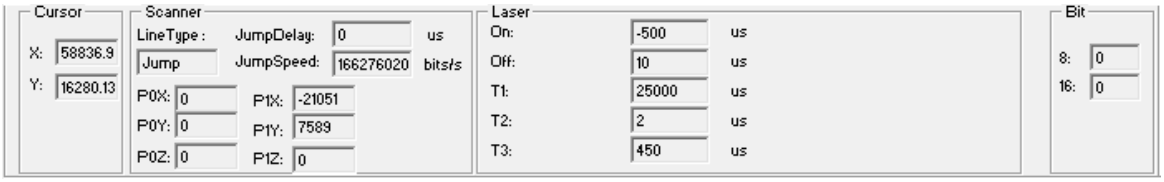
The displayed commands are compatible to the commands sent to the controller card in hardware mode.

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.

Remark:
This box is enabled in simulation mode only.

4.4.2 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.



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.

5 Job Editor

This chapter describes all the manipulations that can be done on geometrical objects. First, all settings which define Geometry Objects are explained. Second, information is given about the possibilities of Transformation. In Entity Info beside general geometric properties also some mark settings are provided for the selected entity. Additionally, this chapter describes how to hatch objects. For the import of vector files into the job editor see section Import. To save or load the job, see Job Format.

5.1 Geometry Objects

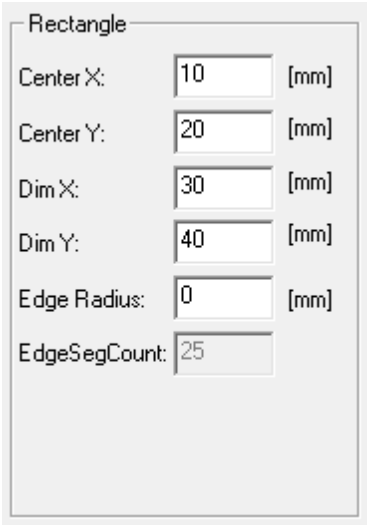
The following geometric objects and their settings are explained here:

- Rectangle, Ellipse
- Barcode
- Bitmap
- Serial Number
- Date Time
- Text2D

5.1.1 Geometry

Standard geometry can be created with the object tool bar. The Geometry Property Page is available for the geometry entities  Rectangle and  Ellipse.

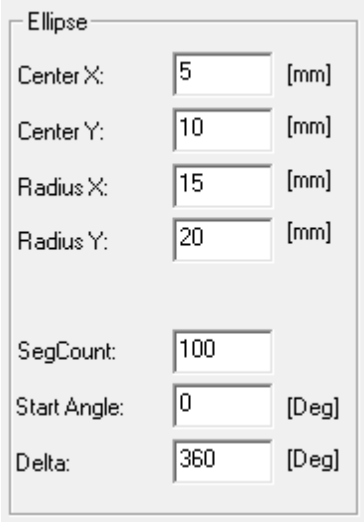
Rectangle



Center, Dimension:
It is possible to define the *centre* 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.

Ellipse



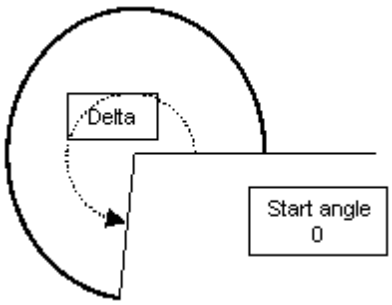
Defines the extension of an ellipse by the centre and the radius. The accuracy can be defined by the segment count.

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.

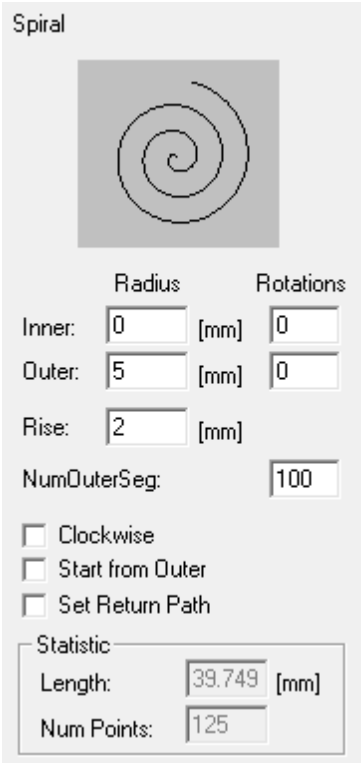
Segment count:
The ellipse consists of a number of segments. This number is calculated to the complete 360° ellipse.

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 at and delta defines the total size of this arc from this starting point.



5.1.2 Spiral

A Standard Spiral can be created with the object tool bar button  .



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.

5.1.3 Barcode

Generate a Barcode by pressing the  Barcode button in the Object-Tool bar. Move the mouse to the desired position and press the left mouse button. The barcode property page appears.

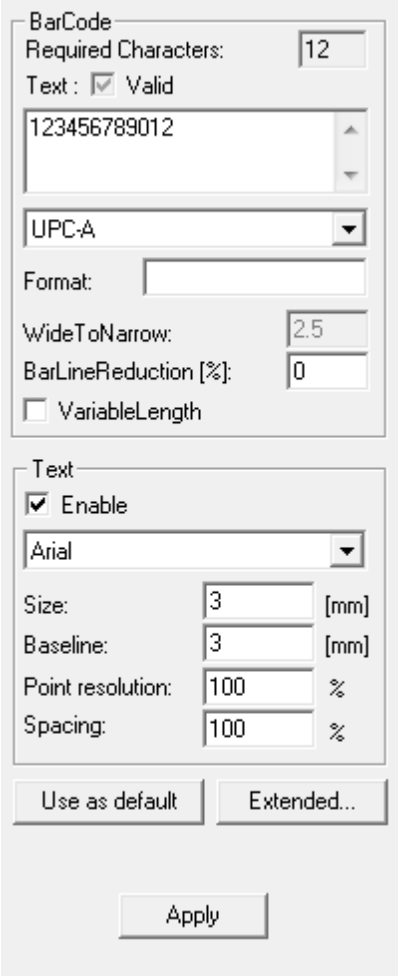


Figure 5.1: Barcode Dialog

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 bar code type.

Format:
See subchapter 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.

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

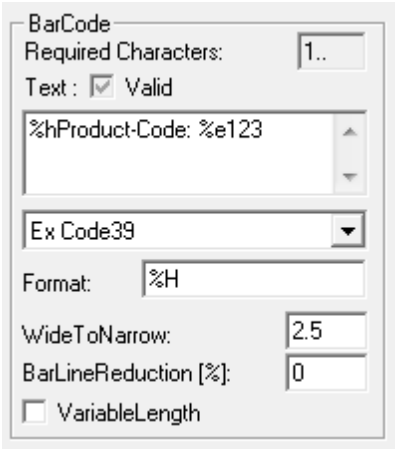
Example:

5.1.3.1 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.

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

Example:



5.1.3.1.1 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:

- (01) identifies a GTIN (Global Traded Item Number)
- (10) identifies a Batch or Lot Number
- (11) identifies a Product Date (as YYMMDD)
- (15) identifies a Best Before Date (as YYMMDD)
- (21) identifies a Serial Number
- (400) identifies a Purchase Order Number
- (422) identifies a Country of Origin (as ISO code)

5.1.3.1.2 QR-Code

The QR-Code is a special 2D Barcode format. 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".

5.1.3.2 Barcode Extended

Press the Extended... button in the barcode property page to get this dialog. If the selected barcode is a DataMatrixEx, see below.

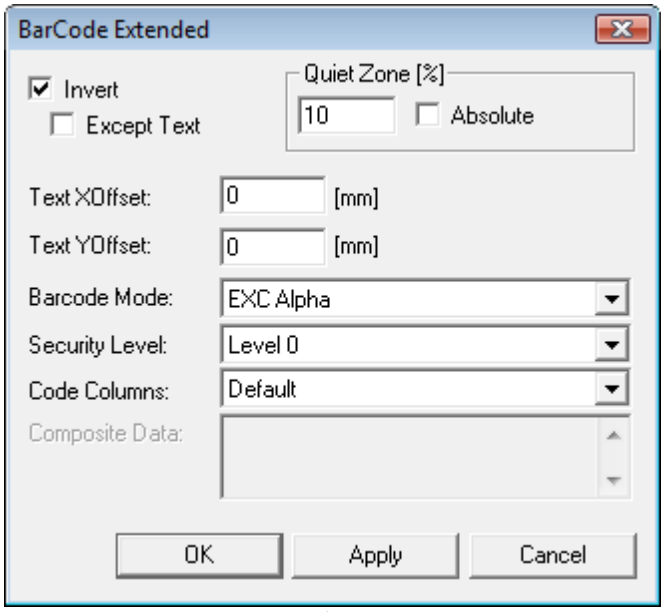


Figure 5.2: Barcode Extended Dialog

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.

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 right, for negative values it is moved to the left.

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.

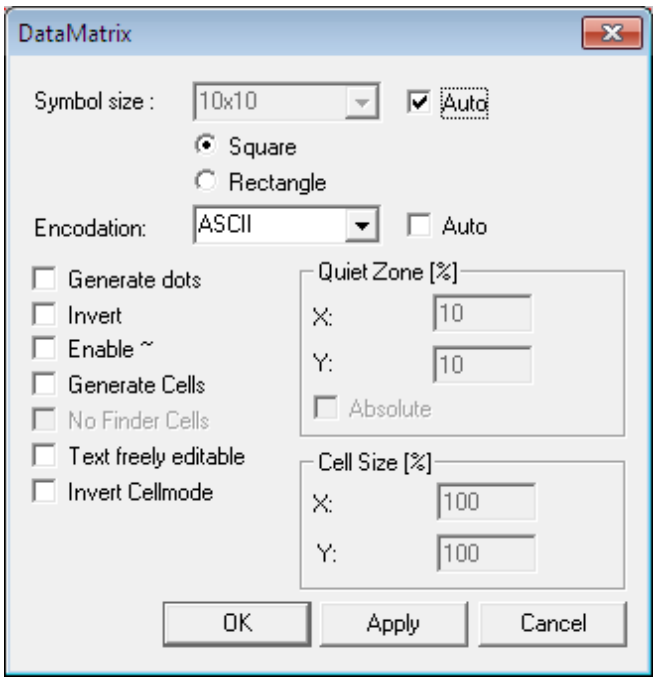


Figure 5.3: Data Martix Extended Dialog

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.

Auto:
Chooses the encodation for which the selected barcode text is being optimally compressed.

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 *QuietZone* 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".

Generate dots, Generate Cells:
If *Generate dots* is checked the barcode consists of single points. If *Generate Cells* is checked, the barcode consists of a closed polygon for each cell. See below for examples.

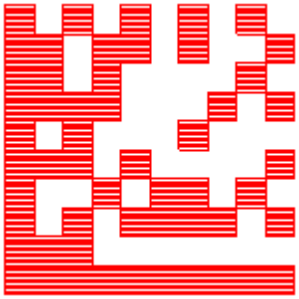
No Finder Cells:
If the barcode is generated by cells then the finder zone is drawn as one closed polygon.

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.

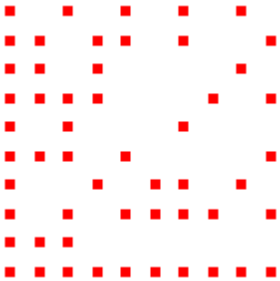
Cell Size
Active if Generate Cells is checked. The size of a single cell can be defined. If it is 100% the cells contact each other.

DataMatrixEx Examples:

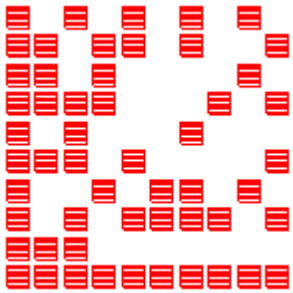
standard barcode



dots generated barcode

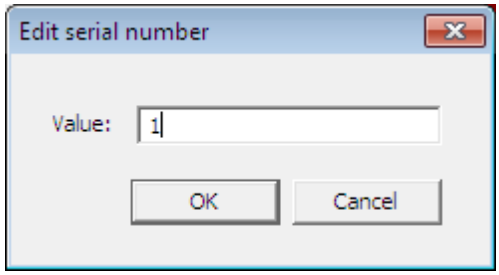


cells generated barcode, cell size 75%



5.1.3.3 Barcode Reader

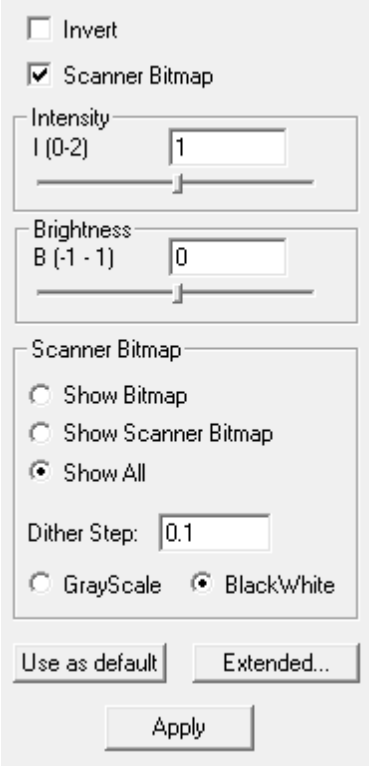
It is possible to connect a barcode reader to the PC and to read barcodes in SAMLight. Therefore create a serial number in the Job. Then in the Serial Number Property Sheet click on the "Advanced" button. In the window that is popping up activate the checkbox "Barcode reader mode" in the field "Popup edit box". Now if you do a mark a window is popping up after the mark has finished. See the picture below:



Here you can enter an arbitrary number or, if a bar code 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.

5.1.4 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:



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.

GrayScale:
If enabled the grayscale values will be kept when transforming the bitmap to the scanner bitmap.

Note: 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.

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.

For the dithered mode LaserOn delay is set to 1 and LaserOff delay to 11 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 subchapter 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:


Original


Scanner Image in error diffuse mode

Remark to 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 page for pens has to be checked.

Note: 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.

For details about pixelmode: see chapter Backgrounds.

5.1.4.1 Bitmap Extended

Press the Extended... button in the bitmap property page to get the dialog shown below.

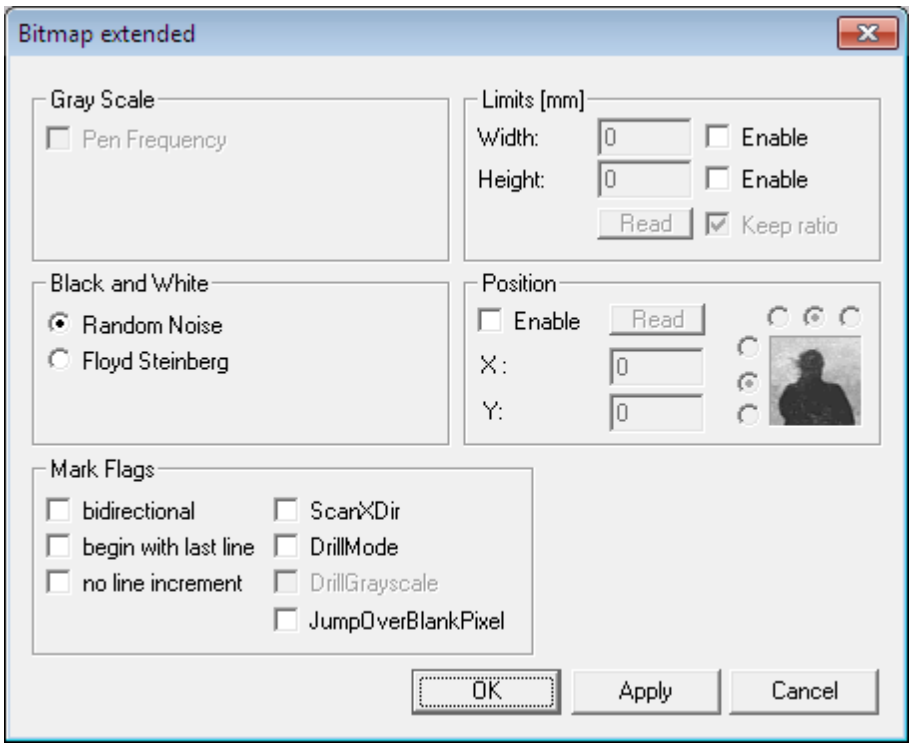


Figure 5.4: Bitmap Marking Extended Dialog

Note: 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.

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.

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.

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.

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.

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.

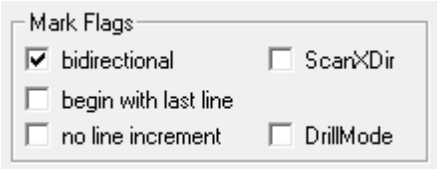
Limits
Define Width and/or Height as a placeholder property for reimport. With *keep ratio* the aspect ratio is kept.

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.

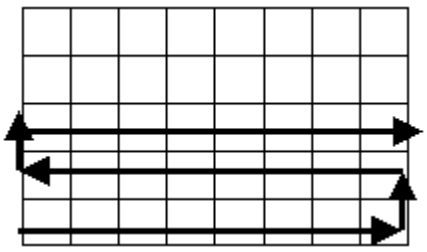
5.1.4.2 Marking Bidirectional

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.

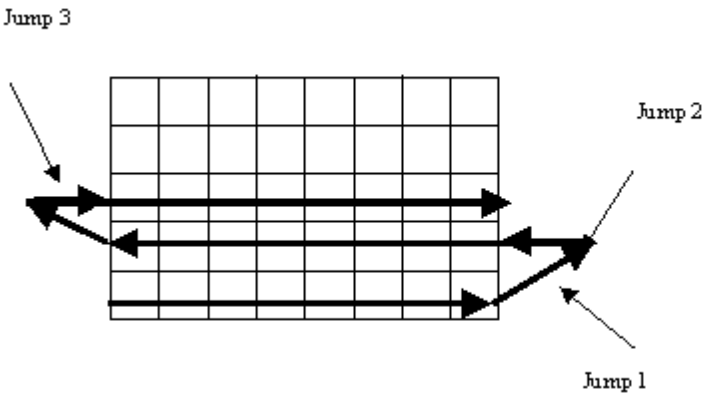


This picture shows the way the laser should go to mark the bitmap properly.



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:



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.

5.1.5 Serial Number

Generate a Serial Number object by clicking the 123 button in the toolbar. Then move the mouse to the desired position and press the left mouse button. The serial number property page appears:

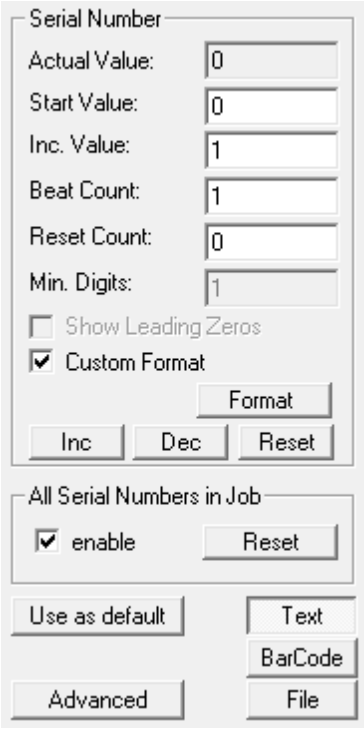


Figure 5.5: Serial Number Dialog

Serial Number

Actual Value:
Shows the actual value. Can not be edited.

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.

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.

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.

5.1.5.1 Serial Number Formats

For the serial numbers the format description is similar to that used in the C-language:

%[flags] [width] [.precision] 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.

type:
f double values

Format examples:

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


Remark:

- Format code and text can be entered simultaneously.
- The encoding will only work if *Custom Format* is selected inside the SerialNumber property page.

5.1.5.2 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:

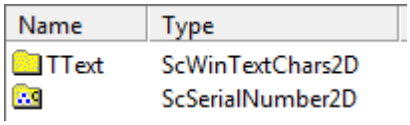


Figure 5.6: Entity List with Text object and Serial Number Barcode

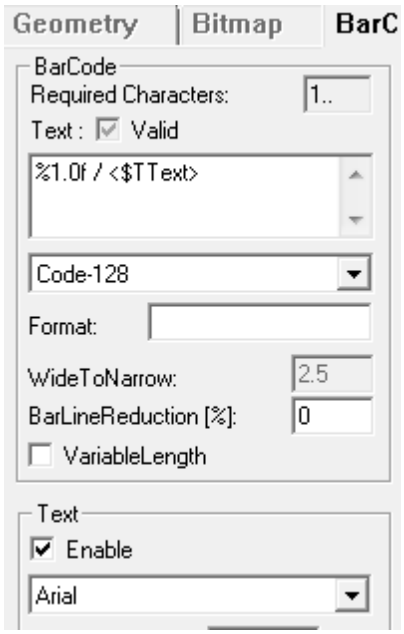


Figure 5.7: Property Sheet with Serial Number Barcode

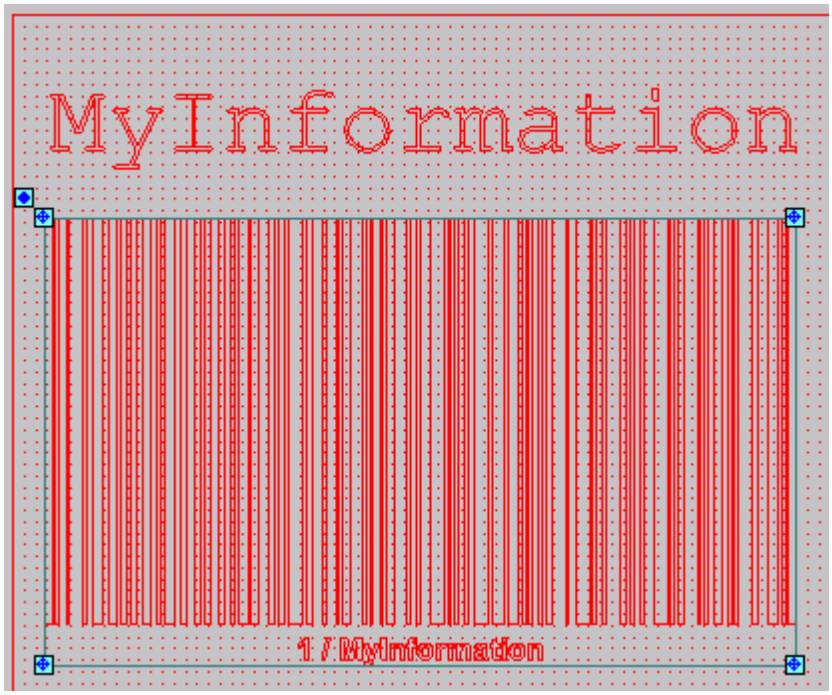


Figure 5.8: View2D with Text to be included and Serial Number Barcode

Please note: 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.

5.1.5.3 Serial Number Advanced

The following dialog can be reached by clicking on the Advanced button in the entity property sheet of the serial number.

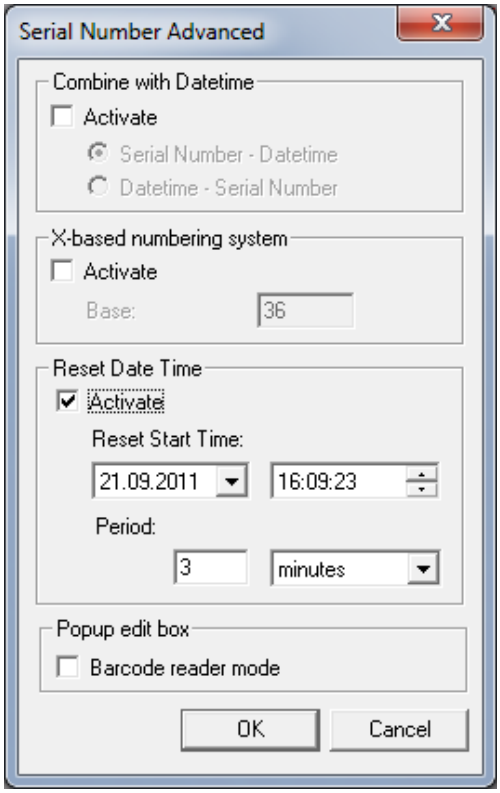


Figure 5.9: Serial Number Advanced Dialog

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".

Note: 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!

Note: The digit definitions of Customer Format are not taken for a x-based numbering system.

5.1.6 Date Time

Generate a date and time object by pressing the DateTime button  in the Object-Tool bar. Move the mouse to the desired position and press the left mouse button.

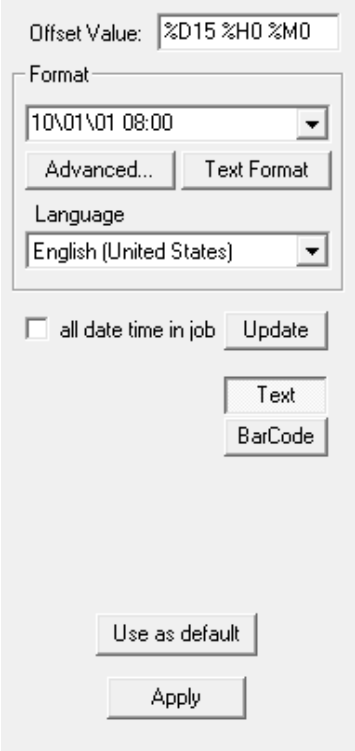


Figure 5.10: Date Time Dialog

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.

5.1.6.1 Date Time Advanced

Press the Advanced... button in the DateTime Property page to show this dialog.

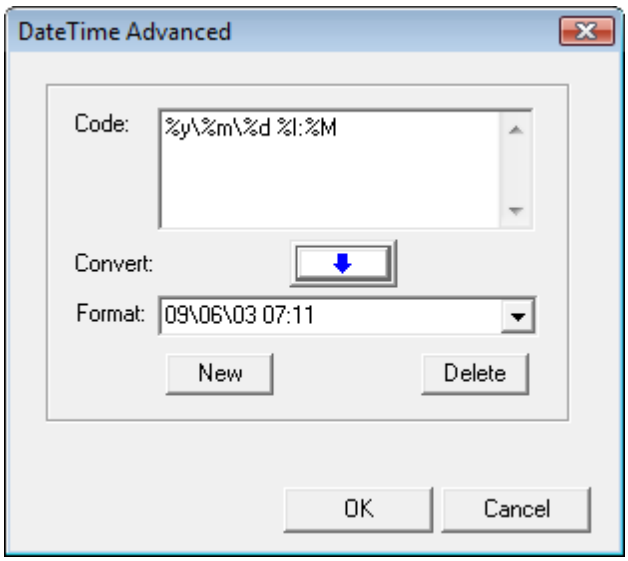


Figure 5.11: Date Time Advanced Dialog

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

%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 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 (00 – 51)
%Q Week of year as decimal number, with Sunday as first day of week (00 – 51)
%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
%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

Format Code

%#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).

5.1.7 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:

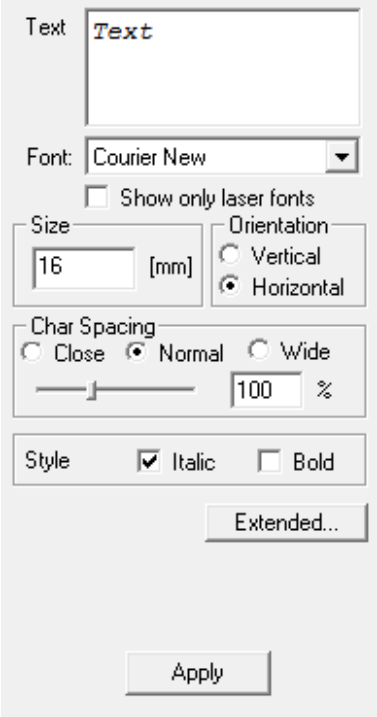


Figure 5.12: Text2D Dialog

Text:
Input field for the text that is generated.

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.

5.1.7.1 Text2D Properties

Clicking the Extended... button in the Text2D property page the following dialog appears:

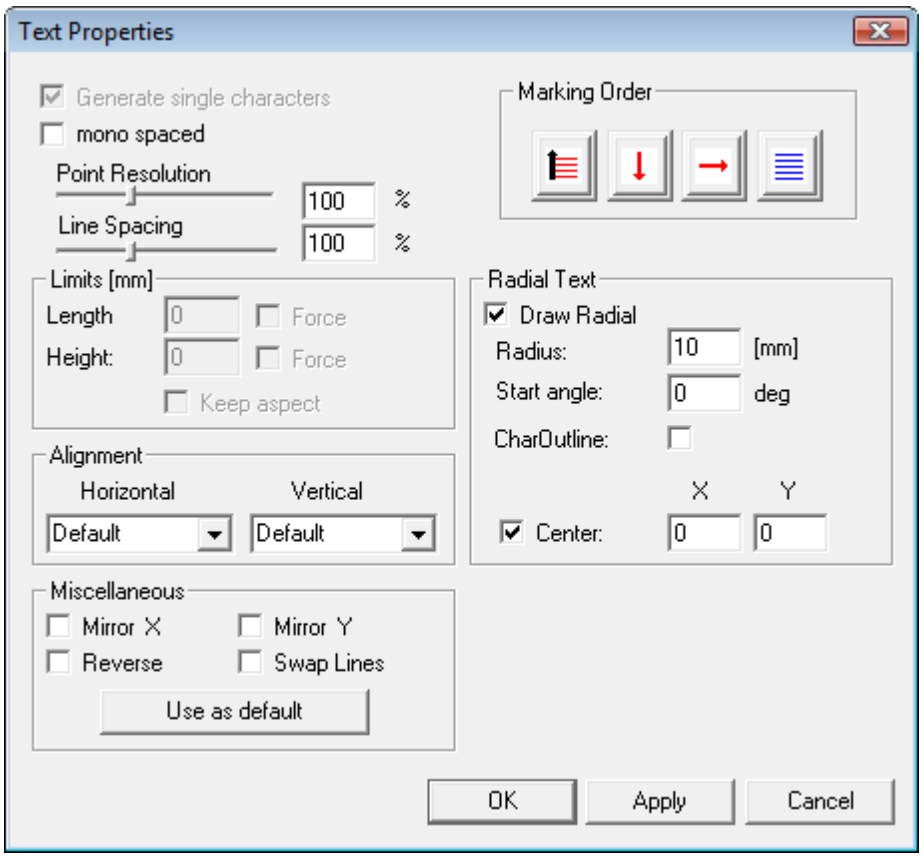
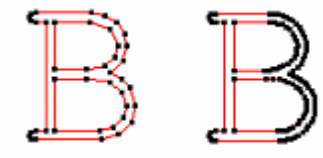


Figure 5.13: Text Properties Dialog

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.

Radial:
Activating radial mode. Radius and Start angle define the position of the characters.

CharOutline:
Takes the outlines of character in 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.

5.2 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.

5.2.1 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.

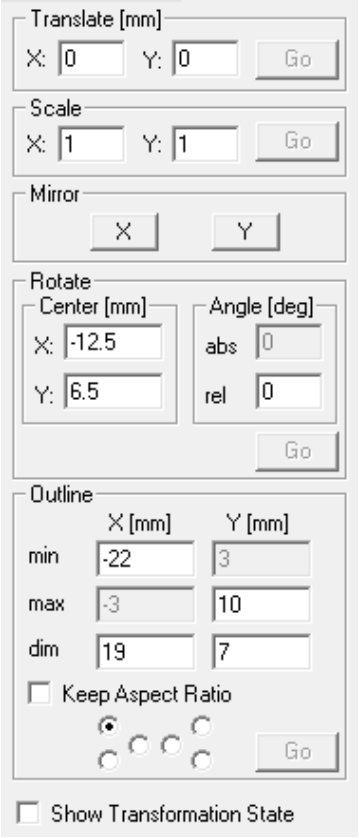


Figure 5.14: 2D Transformation Dialog

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.

5.2.2 3D Transformations

↯ Requires the Optic Dimension 3D tool

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.

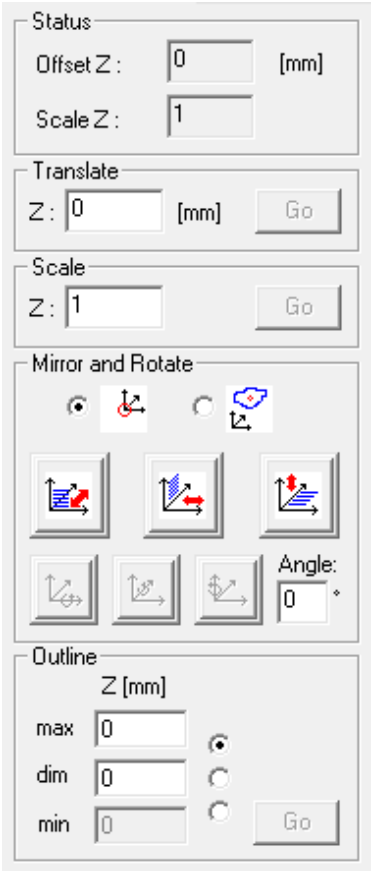


Figure 5.15: 3D Transformation Dialog

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.

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.

5.3 Element Info

Gives information about a single polyline structure.

Vectors:
Number of lines of the selected polyline.

Length:
Length of the polyline.

Outer / Inner:
Orientation of the polyline, outer->contra clockwise, inner->clockwise.

Open:
The polyline is open or the polylines structure it belongs to is open.

5.4 Entity Info

The Entity Info property page shows the type and the name of the selected entity.

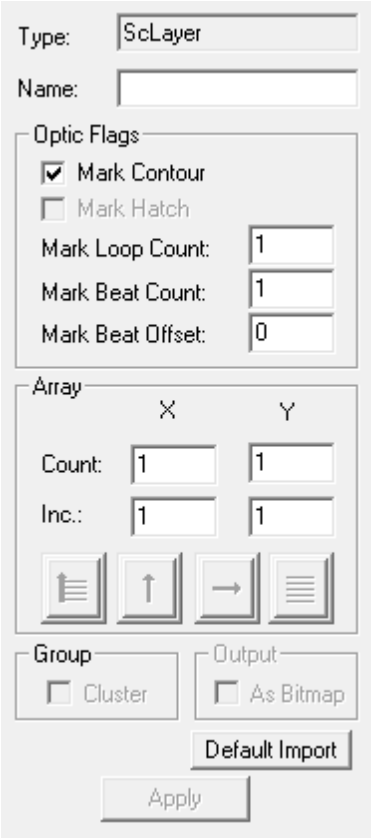


Figure 5.16: Entity Info Dialog

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.

Count:
Define the number of copies in X and Y direction.

Inc.:
Define the distance between the copies in X and Y direction.

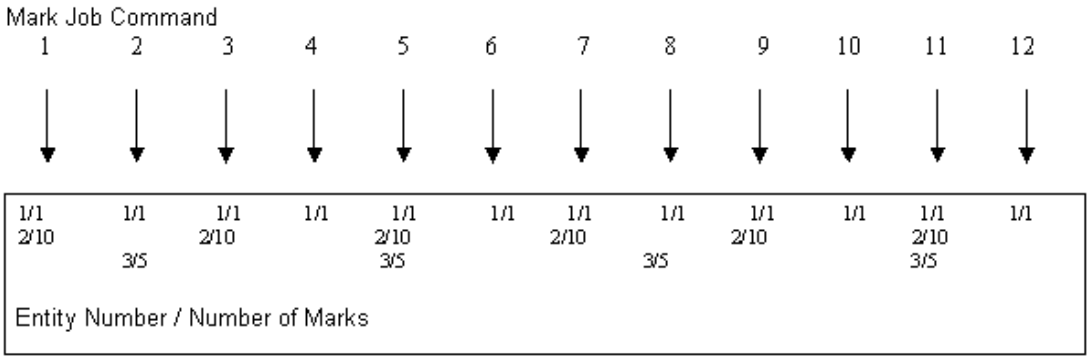
Group
If the entity is a group, it can be *clustered*. A clustered group will be hatched as one single object while in a unclustered group each entity of groups is hatched separately.

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

With these settings the following mark sequence can be generated:



- 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.

5.5 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.

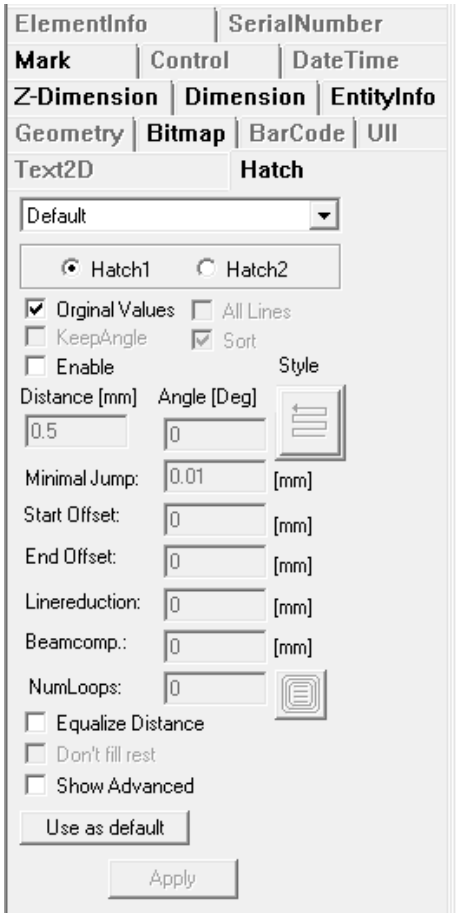


Figure 5.17: Hatch Dialog

Hatch1, Hatch2:
Each entity can have a cross hatch with two different hatchings. The radio buttons allow to switch between the two hatching definitions.

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.

AllLines:
If this is not checked, ScPolyLine2D structures are used for calculating hatches. If checked, also ScLineArray2D objects are taken into account.

Sort:
Sorts the order of the lines to get minimal jumps. Below is an image that describes this feature. The order of the hatch is 1,2,3,4,5:

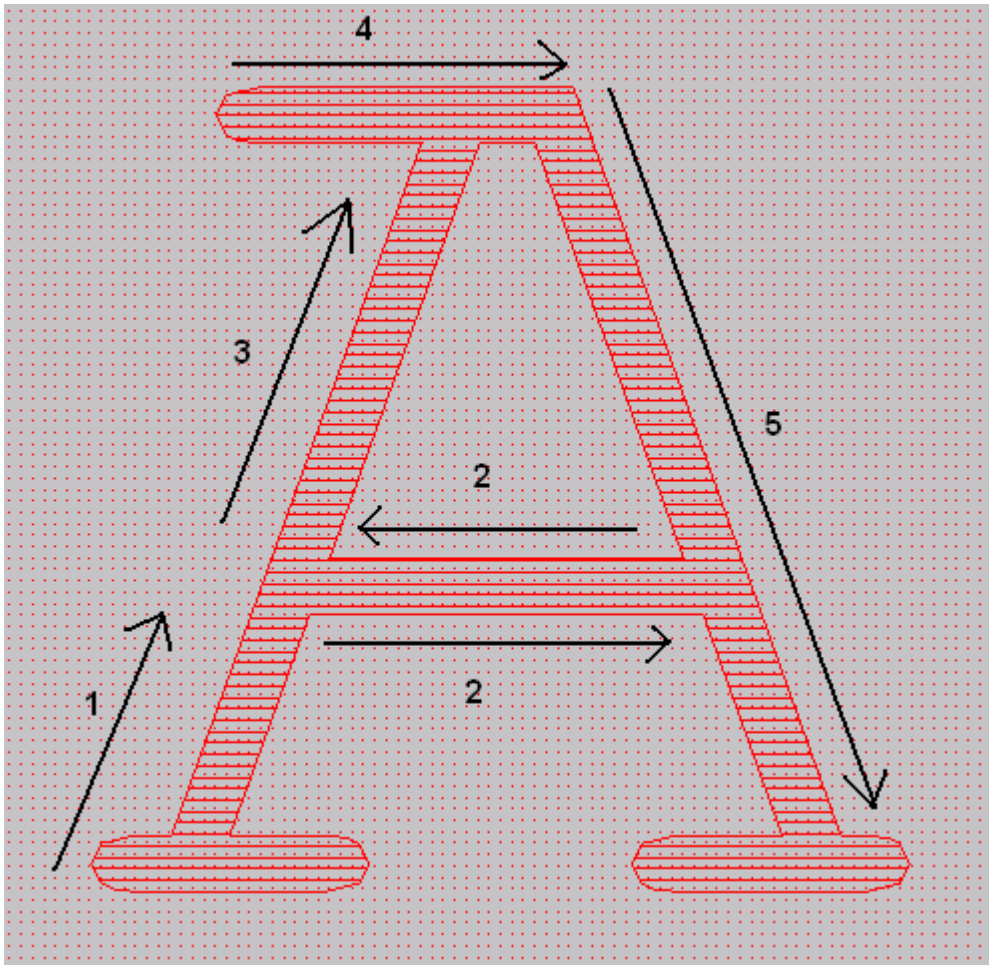


Figure 5.18: Hatching order with Sort Flag activated

If "Sort" is not activated the Hatch order is simple from bottom to top. At the crossing points the sort can cause problems depending on the material / application.

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.

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).

Line reduction:
Shortens the hatch lines.

Beam compensation:
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.

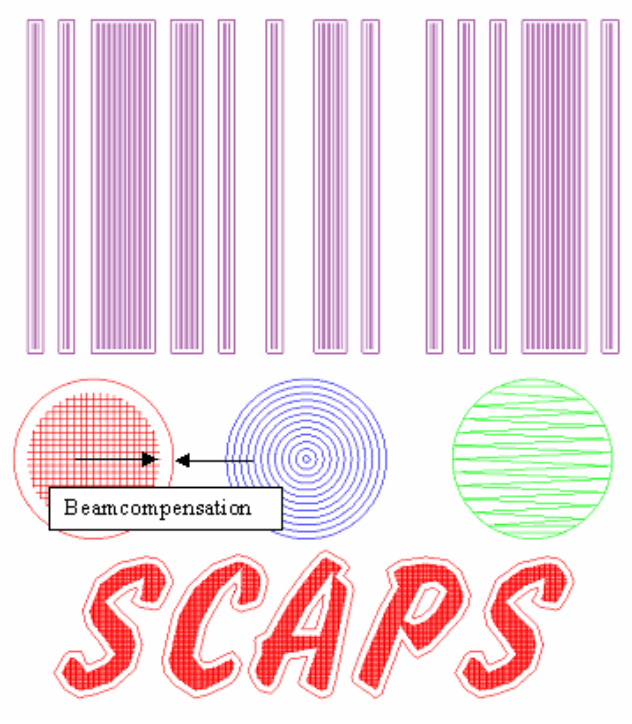
Equalize Distance:
Changes Distance so that Start Offset *and* End Offset can exactly be set by the hatcher.

Don't fill rest:
If checked only NumLoops gets hatched. The object gets not hatched inside the loops.

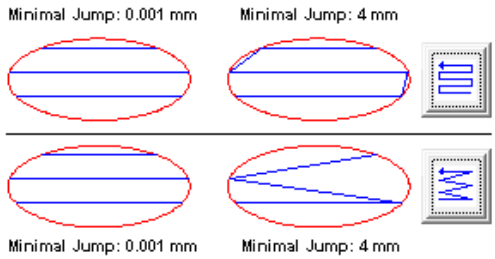
UseAsDefault:
Uses the actual hatch style of the currently selected object as default for new entities. The program saves these settings also for a new program start in case save settings on exit is checked in the general settings.

Apply:
After defining the distance the angle and the style clicking on button Apply will generate the hatch lines.

Example:



Example Minimal Jump:



5.6 Job Format

Menu File->Load opens a Dialog Box to load a new job from a sjf (SCAPS Job Format) file. On the right side there is a preview window. And at the bottom of this is a display box of all available entries inside the currently selected job file.

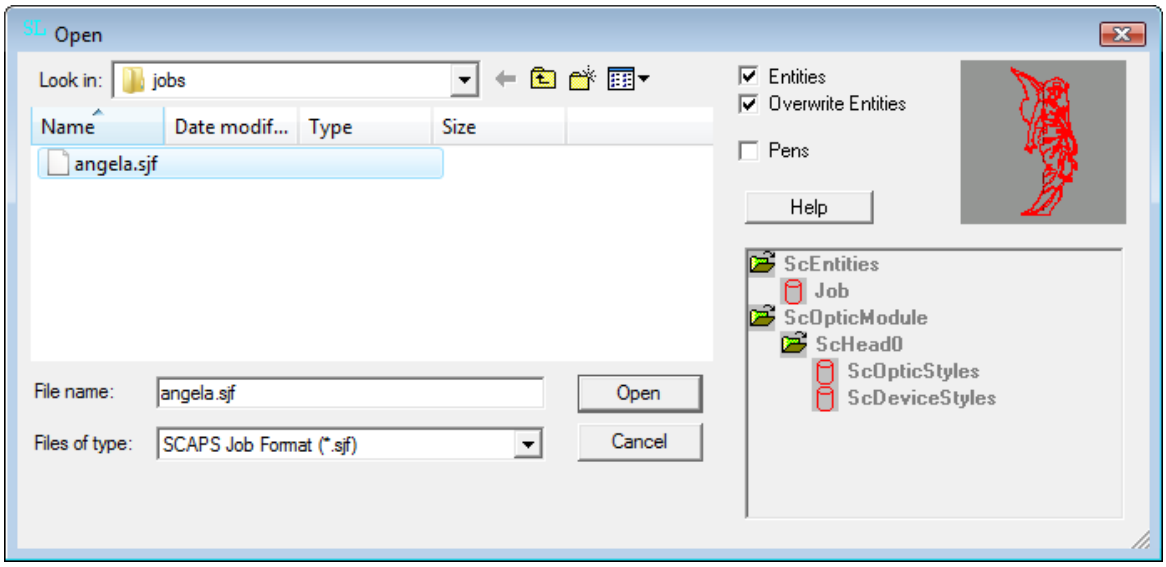


Figure 5.19: Open File Dialog

Menu File->Save saves the current job. If there is no job name defined it is called SaveAs. Menu File-

›SaveAs opens a Dialog Box to save the current job under a new name.

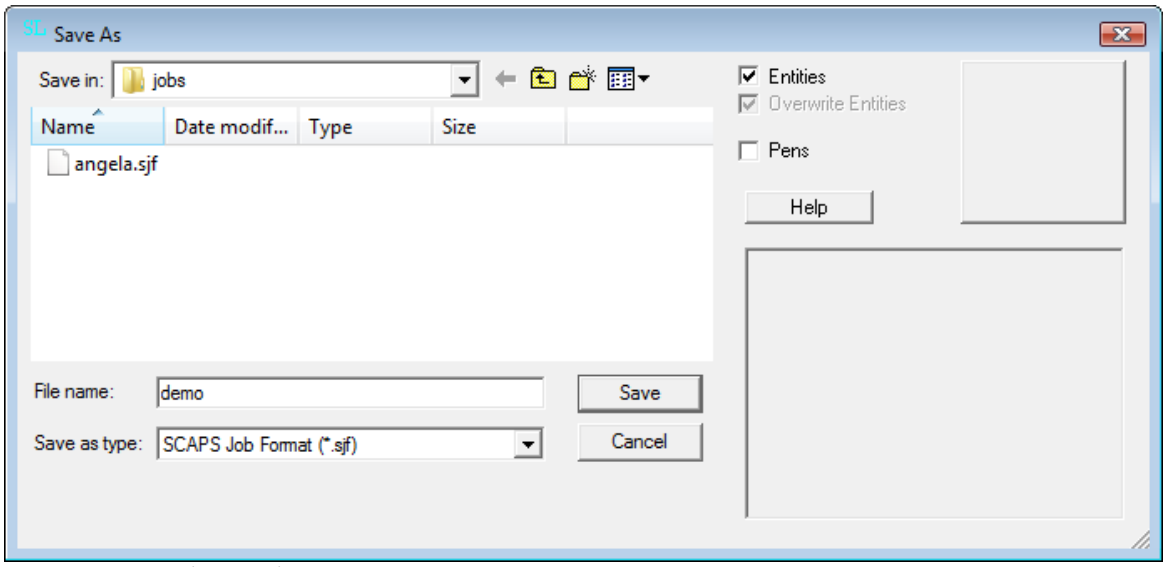


Figure 5.20: Save File As Dialog

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.

5.7 Import

Menu File->Import opens a Dialog Box to import Graphic files and puts them into a new layer.

Available formats are:

- SCAPS Archive Files
- HPGL Plot Files
- DXF Files
- Corel Presentation Exchange Files
- Enhanced Windows Metafile Files
- Scalable Vector Graphics Files
- Adobe Illustrator Files
- PC-MARK Files
- MCL Files
- Point Cloud Files
- Windows Bitmaps
- PCX Files

Note: In the SAMLightEntry version only the formats plt, dxf, dst, txt, cnc and bmp are available for import.

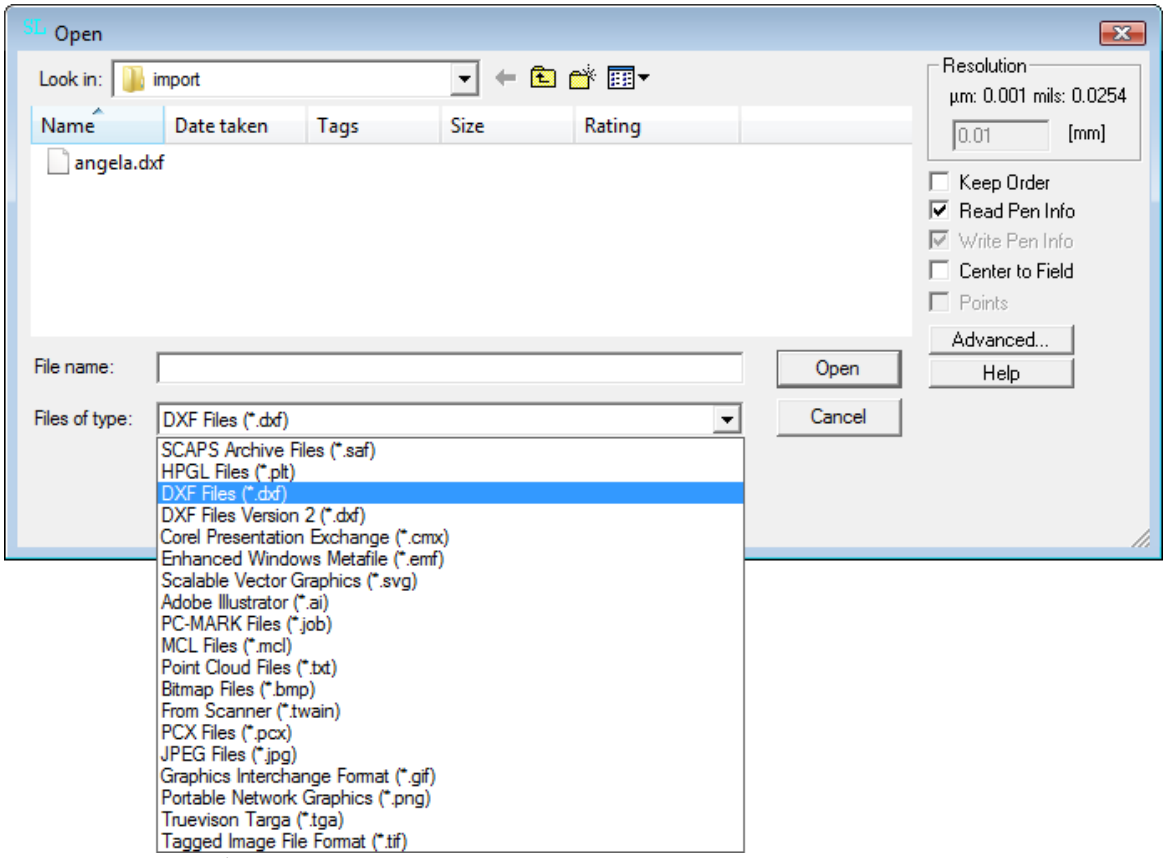


Figure 5.21: Import Dialog

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":

checked:

not checked:

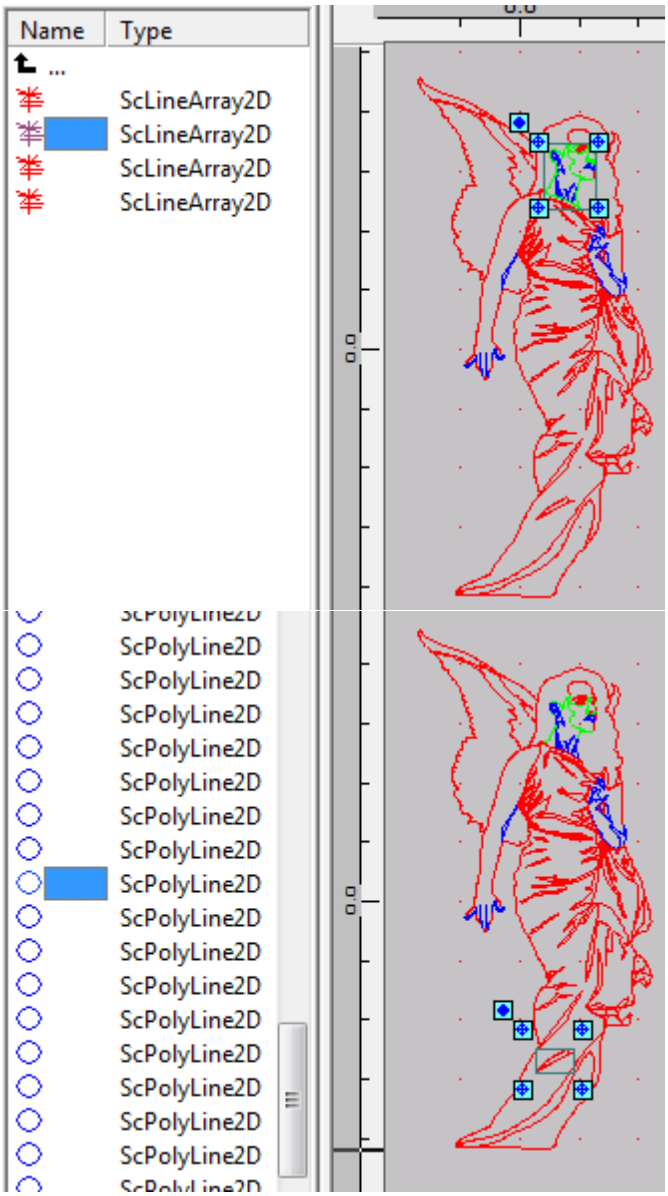


Figure 5.22: Example for Keep Order

Activated options for available file formats are:

	Resolution	KeepOrder	ReadPenInfo	Points	PreView
SAF	0	0	X	0	X
PLT	X	X	X	X	0
DXF	0	X	X	0	0
Version2 DXF	0	0	X	0	0
CMX	0	0	X	0	0
EMF	0	0	X	0	0
SVG	0	0	X	0	0
AI	0	X	X	0	0
JOB	0	0	X	0	0
MCL	0	0	X	0	0
BMP	0	0	0	0	0

PCX	0	0	0	0	0

"X": selectable; "0": preallocated

5.7.1 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.

5.7.2 Import Advanced

For some file formats the button "Advanced" is active to influence the display of the drawing.

For *PLT formats* the following "Advanced Styles" dialog is given:

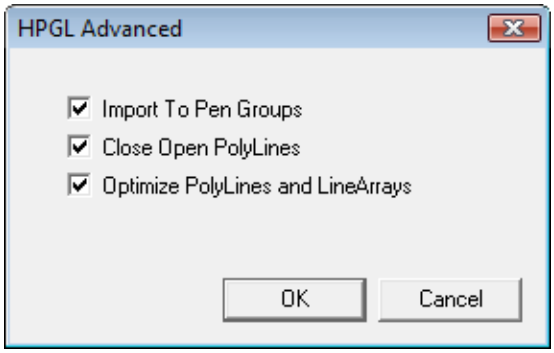


Figure 5.23: HPGL Advanced Dialog

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.

For files of format *DXF Version1* (DXF Files *.dxf) the following "Advanced Styles" dialog is given:

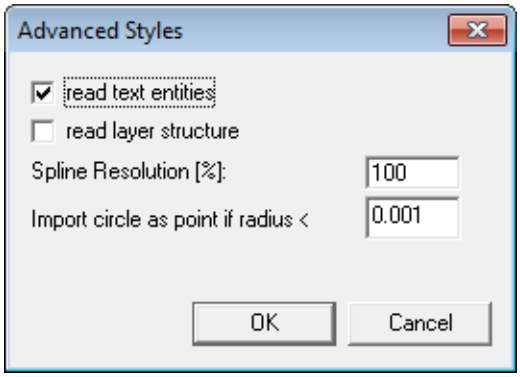


Figure 5.24: Advanced Styles for DXF Dialog

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.

For files of format *DXF Version2*, *DWG*, *CMX*, *EMF* and *SVG* following "Advanced Styles" dialog is given:

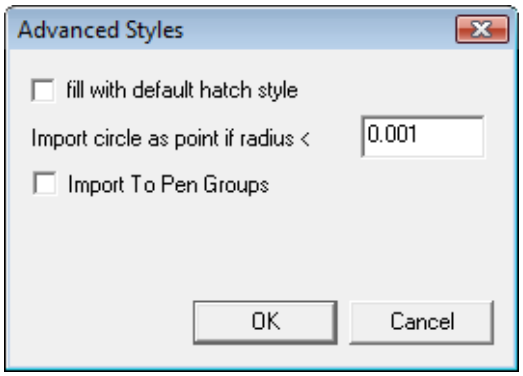


Figure 5.25: Advanced Styles for Other Files Dialog

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.

5.7.3 Vector File Formats

DXF Files (Standard):
No Text import.
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:
Supported versions are 5, 6, 7, and 8.

Scalable Vector Graphics files:
Imports: group, circle, line, polyline, Polygon, Polydraw, arc, (text).

5.8 Export

Menu File->Export opens a Dialog Box to export selected entities into graphic files. Available formats are SCAPS saf and HPGL plt. For HPGL file the Resolution and to write the pen information can be defined.

6 Optic

In the following chapter the settings which have an optical effect on marking objects are described. These settings are card settings as well as pen definitions and affect the hardware of the scanning system.

6.1 Optic Settings

The optic settings dialog is for setting the geometrical dimensions of the scanner field. Field *size* and *gain* define the resolution. For a detailed definition and a calculation example see below.

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 fieldsize is being calculated out of the current field size in the following way (C is the current output dimension and O is the original dimension of the drawn object):

corrected field size = current field size * C / O

Gain:
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 / 550 bits)mm ≈ 119.16 mm.

Considering a Gain factor:

scanner field in x direction = (65535 / 550)mm * GainX
scanner field in y direction = (65535 / 550)mm * GainY.

Or in other words:

550 bits/mm = (65535 bits/ size) * Gain

Note:
Optic settings can be done in the optic settings dialog which can be reached by Menu item "Settings->System->Optic", but also before software start. Therefore:

1. Quit all SAM based applications.
2. Start sc_setup.exe from folder SCAPSINSTALLDIR\tools
3. Click on Menu "Hardware Settings"->Settings

6.1.1 Optic Settings Dialog USC-1

Requires USC-1 scanner card.

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":

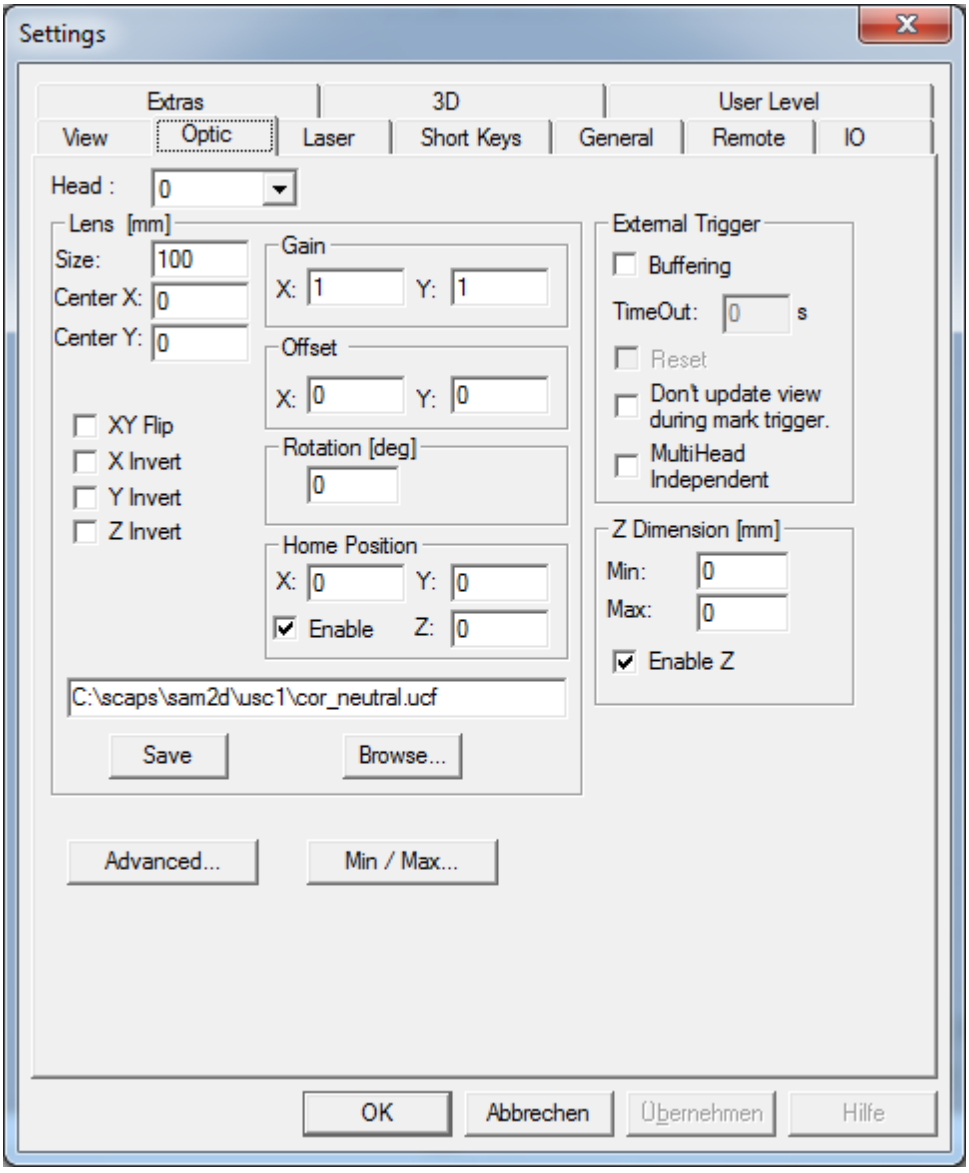


Figure 6.1: Optic Settings for USC-1 Card

Lens

Size:
Specifies the maximum scanning field.

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

XY Flip:
Exchanges X and Y axis

X Invert:
Inverts the X axis direction

Y Invert:
Inverts the Y axis direction

Z Invert:
Inverts the Z axis direction, ↯Requires the Optic Dimension 3D tool

Gain:
The gain values are thought to slightly compensate X/Y gain errors to achieve a quadratic field. For how to calculate size and gain please refer to section optic settings.

Offset:
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 the Field edit group.

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

Home Position:
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 is delivered with the scanner card. For USC-cards the standard extension is ".ucf".
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.

Advanced...
See dialog USC-1 Card Settings.

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

Buffering:
If in trigger mode this allows to buffer the job on the scanner card.

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.

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.

MultiHead Independent:
This allows to trigger the Scanheads independently from each other. This might be useful if one of the scanheads has to mark for example 5 objects while the other scanhead will only mark one object at the same time. This function will not work together with Serialnumber objects nor with DateTime objects.

ZDimension

Requires the Optic Dimension 3D tool
The enable *EnableZ* flag must be set to use the Z translator. *Min* and *Max* define the Z scaling factor which must be set properly. Each Z value given in the world coordinates [mm] will be transferred to a bit value [+/-32k] according the following formula:

Z[bit] = Z[mm] * K, with K as
K= (65535 / (ZMax-ZMin)) - 32768

6.1.2 Optic Settings Dialog USC-2

Requires USC-2 scanner card.

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":

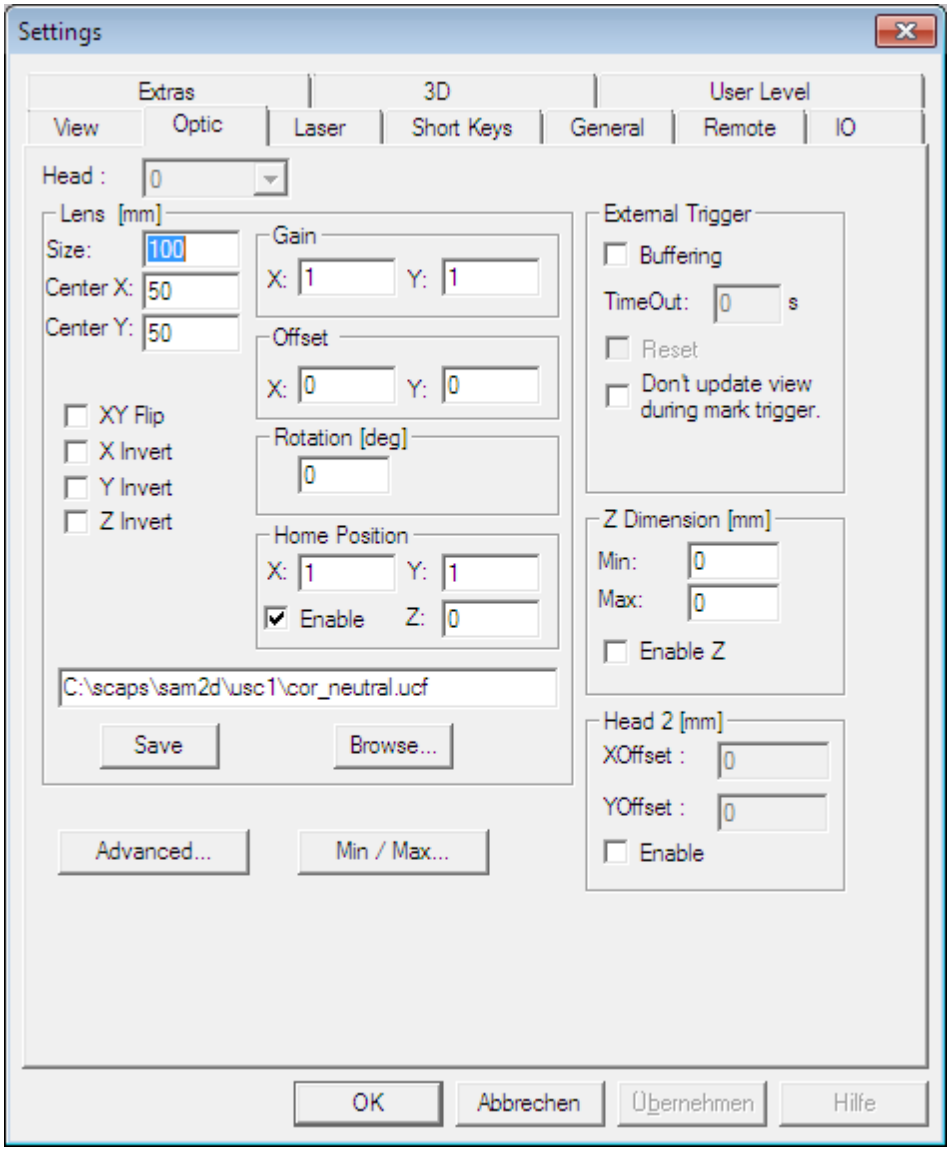


Figure 6.2: Optic Settings for USC-2 Card

Lens

Size:
Specifies the maximum scanning field.

Center:
Defines the center of the scanner field according the world coordinate system.

XY Flip:
Exchanges X and Y axis

X Invert:
Inverts the X axis direction

Y Invert:
Inverts the Y axis direction

Z Invert:

Inverts the Z axis direction, ↯Requires the Optic Dimension 3D tool

Gain:
The gain values are thought to slightly compensate X/Y gain errors to achieve a quadratic field. For how to calculate size and gain please refer to section optic settings.

Offset:
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 the Field edit group.

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

Home Position:
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 must be inside the scanner field.

Correction File Settings:
Here the correction file can be specified. This file is delivered with the scanner card. For USC-cards the standard extension is ".ucf".
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.

Advanced...
See dialog USC-2 Card Settings.

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

Buffering:
If in trigger mode this allows to buffer the job on the scanner card.

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.

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.

ZDimension

Requires the Optic Dimension 3D tool
The enable *EnableZ* flag must be set to use the Z translator. *Min* and *Max* define the Z scaling factor which must be set properly. Each Z value given in the world coordinates [mm] will be transferred to a bit value [+/-32k] according the following formula:

Z[bit] = Z[mm] * K, with K as
K= (65535 / (ZMax-ZMin)) - 32768

Head 2

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

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

Note: 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".

6.1.3 Optic Settings Dialog RTC3/4/5, SCANalone, HC3

↯ This dialog appears if scanner card RTC3/4/5, SCANalone or HC3 is used.

The following dialog can be reached by the Menu item "Settings->System->Optic". The dialog allows to define the settings of the geometrical dimensions of the scanner field.

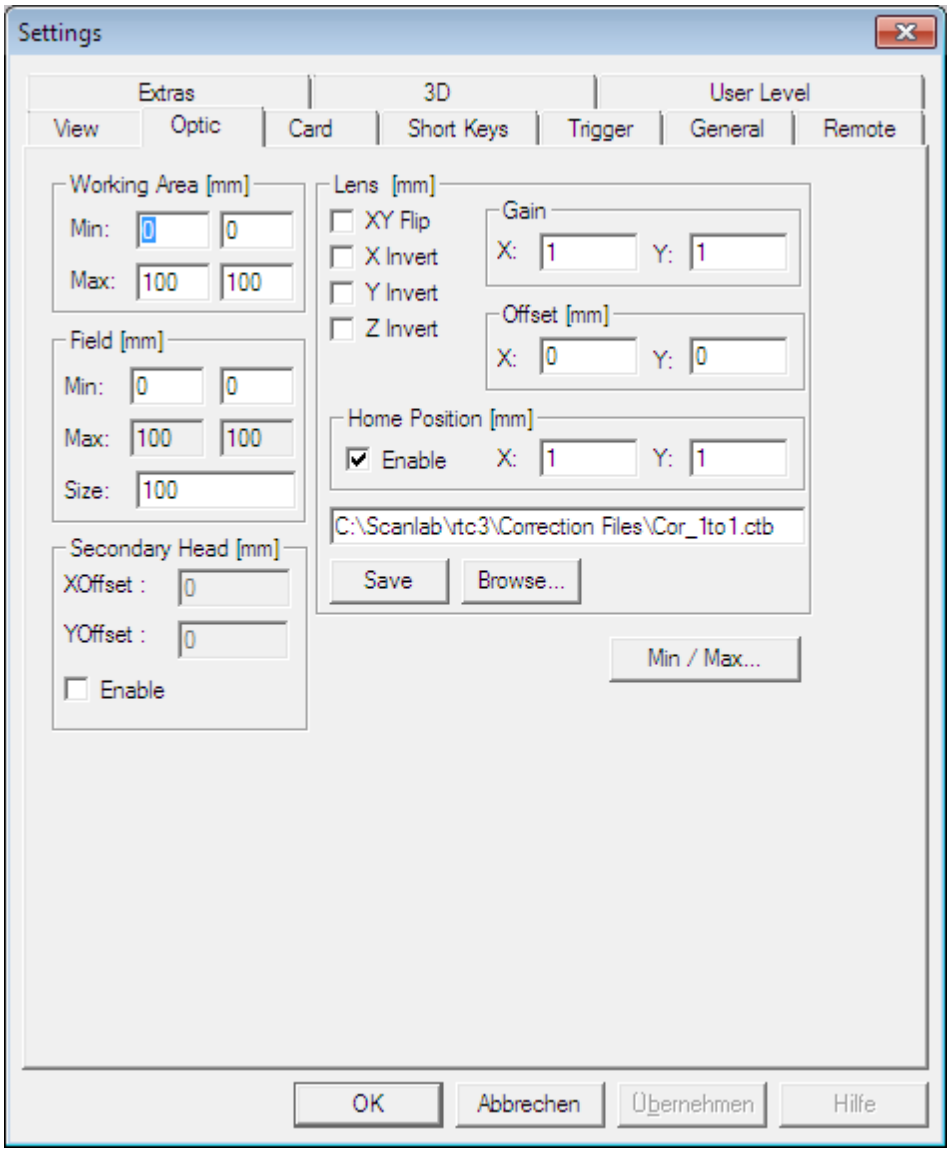


Figure 6.3: Optic Settings for RTC and HC3 Cards

Lens

XY Flip:
Exchanges X and Y axis

X Invert:
Inverts the X axis direction

Y Invert:
Inverts the Y axis direction

Z Invert:
Inverts the Z axis direction, ↯Requires the Optic Dimension 3D tool

Gain:
The gain values are thought to slightly compensate X/Y gain errors to archive a quadratic field. For how to calculate size and gain see chapter optic settings.

Offset:
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 the Field edit group.

Home Position:
The home position is the position where the scanner is located when no scanning takes place i.e. during handling activity. In normal case the home position is outside the working area but obligatory inside the scanner field.

Correction File Settings:
Here the correction file can be specified. This file normally is delivered by the scanhead manufacturer and needs to fit to the scanner, the scanner card and the used optic. 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 two buttons several different lens- and correction file configurations can be managed.

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

Working Area

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

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

Size:
Specifies the maximum scanning field.

Secondary Head
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 jobeditor. XOffset/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.

6.1.4 Optic Min/Max

By pressing *Min /Max...* in the Menu -› Settings -› Optic dialog following dialog appears. The settings define the range of the values *Speed*, *Frequency* and *First Pulse* killer length of the laser.

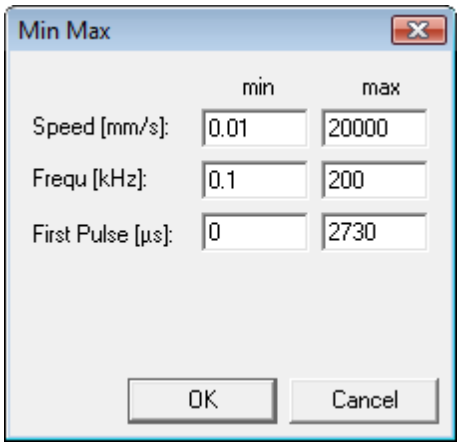


Figure 6.4: Optic Min/Max Dialog

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

Note: The min and max values of X and Y are automatically adapted to the field size values (default) again if Size / Center X / Center Y in the optic dialog are being changed.

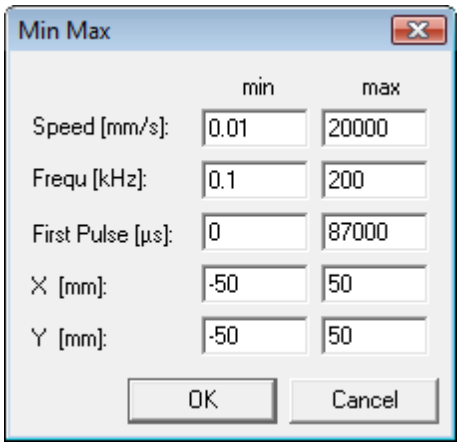


Figure 6.5: Min/Max for USC Cards

6.2 Card Settings

There are different scanner cards available which are supported by SAMLight. In the following the card setting possibilities for the RTC3, RTC4, RTC5, SCANalone, HC3 and the USC-1, USC-2 scanner card are described.

Additional settings dialog "Card" for RTC3/4/5 and SCANalone scanner card:
This dialog can be reached under the menu: Settings -› System -› Card.

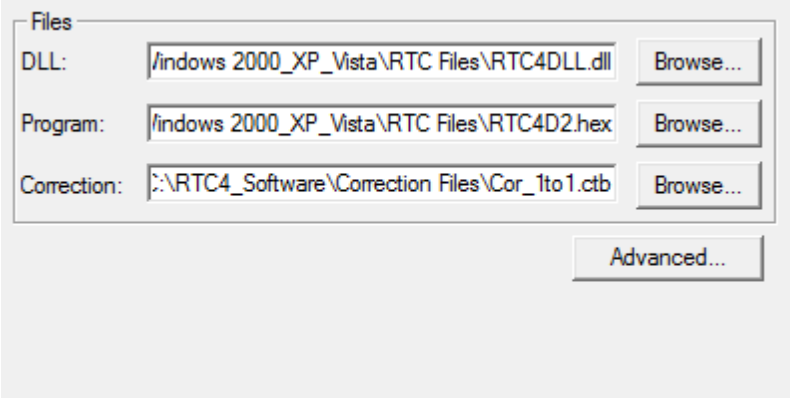


Figure 6.6: Additional Settings for RTC Cards

Note: There is no Program File to browse for in case a HC3 card is used.

Note: When using a 64-Bit operating system you have to install the 32-Bit drivers for the RTC cards because SAMLight is a 32-Bit application.

6.2.1 Port Overview

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 / Laser-modulation	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*	40-pin extension *DIGITAL IN0* ... *DIGITAL IN15* *DIGITAL OUT0* ... *DIGITAL OUT15*
External 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

Marking in progress	*OPTO_OUT 0* -› pin 3		

Port	RTC5	SCANalone	HC3
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	Available in combination with the IO2 extension board.
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	Port B of the HC3
Q-Switch / Laser-modulation	15-pin SUB-D laser connector *Laser1* -› pin 1		9-pin connector pin 2
LaserGate	15-pin SUB-D laser connector *LaserOn* -› pin 2		25-pin connector External IO pin 15, pin 16
IO	40-pin extension *DIGITAL IN0* ... *DIGITAL IN15* *DIGITAL OUT0* ... *DIGITAL OUT15*	37-pin extension1-SUB-D connector *DIGITAL IN0* ... *DIGITAL IN15* *DIGITAL OUT0* ... *DIGITAL OUT15*	
External Trigger:	15-pin SUB-D laser connector	9-pin SUB-D laser connector	9-pin connector External IO
Start	*/START* -› pin 3	*/START* -› pin 8	*/START* -› pin 4
Stop	*/STOP* -› pin 11	*/STOP* -› pin 9	*/STOP* -› pin 3

Marking in progress			pin 5

Table 6.1: Ports and Connections for Scanner Controller Cards

6.2.2 USC-1 Card Settings

↯ Requires USC-1 scanner card

If the registered scanner card is an USC-1 card, pressing the Advanced... button in Menu->Settings->Optic the following dialog appears. Here all the card settings have to be done.

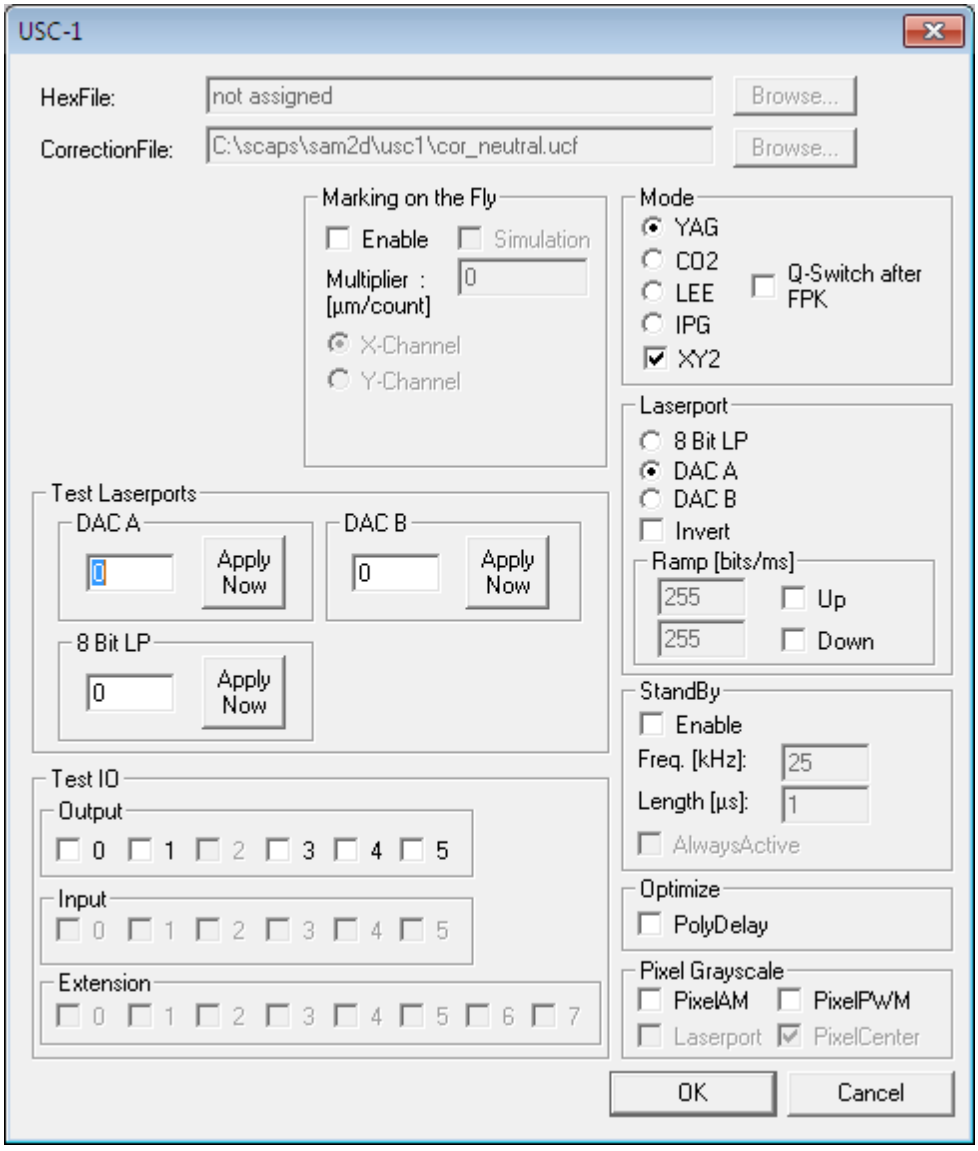


Figure 6.7: Card Settings for USC-1 Card

HexFile
Click on the Browse button to select the correct program file for this card.

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

MarkingOnTheFly

↯ Requires the MarkingOnTheFly Module
For an explanation how to do settings for mark on fly see chapter How to Mark on Fly.

Enable:
Enable this checkbox to enable Marking on the fly

Simulation:
Encoder pulses will be simulated with a constant pulse frequency of 100 kHz.

Multiplier:
Distance in x/y-direction per encoder count (movement of the target is only possible in one direction). The multiplier may also have a negative sign. This inverts the compensation direction.

X/Y Channel:
Compensate along the Scanner X or Y axis.

Test Laserports

8-Bit Digital:
Example: If '5' is entered then after pressing *ApplyNow* 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

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

Laserport
Defines the laser power port. Note: For bitmaps which are amplitude-modulated the analog port B, *DA2,* is always used.

Ramp:
Defines a velocity to increase or decrease the power of the laser port.

StandBy

Enable:
Globally enables stand-by mode.

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

Length:
Q-Switch length in µs.

AlwaysActive:
Stand-by 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.

Laserport:
If checked the selected Laserport gets used for the output of PixelAM, else port DA2 is taken.

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

6.2.3 USC-2 Card Settings

Requires USC-2 scanner card

If the registered scanner card is an USC-2 card, pressing the Advanced... button in Menu->Settings->Optic the following dialog appears. Here all the card settings have to be done.

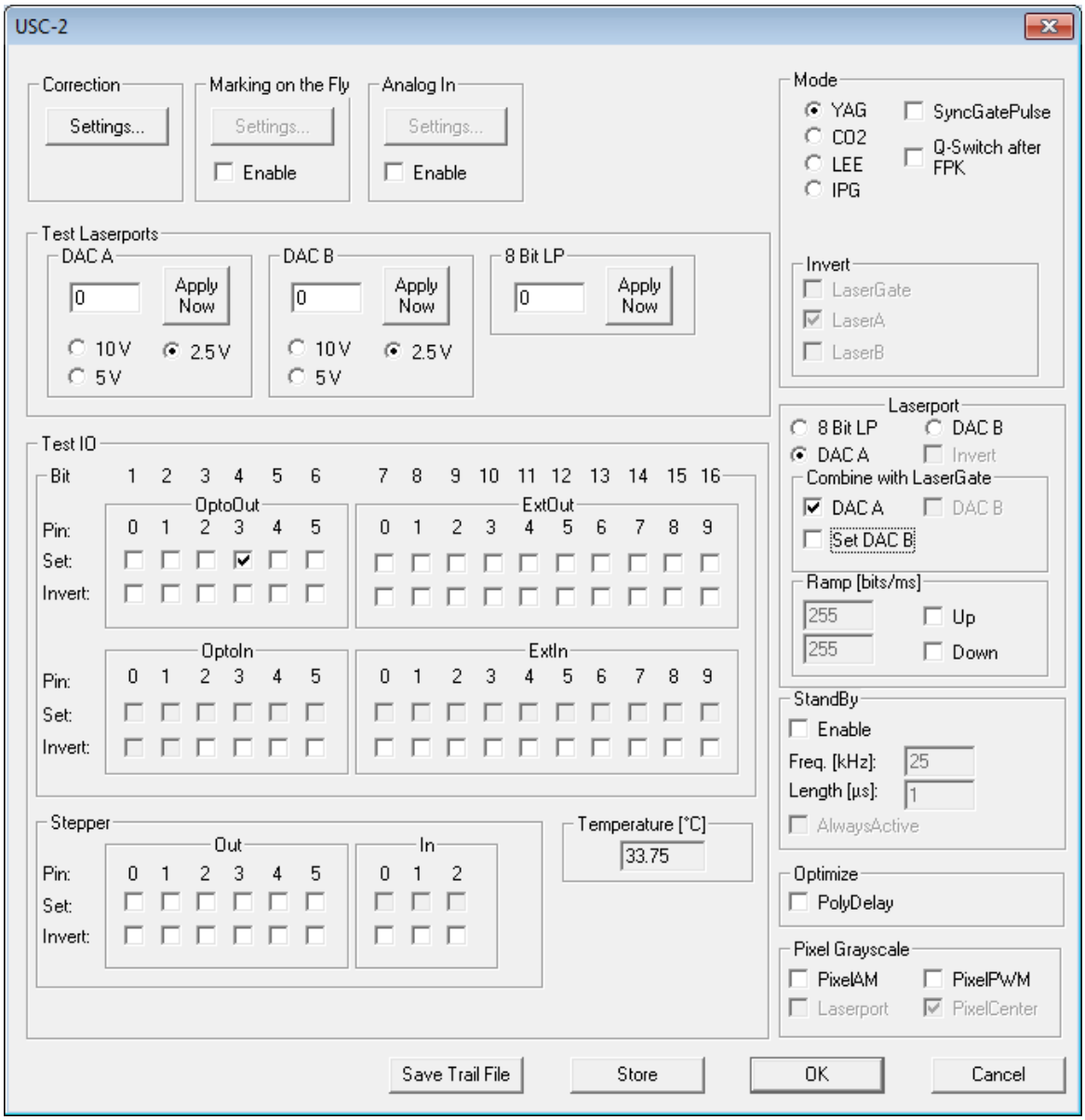


Figure 6.8: Card Settings for USC-2 Card

Correction

Settings...
Click this button to open the Correction File Dialog for the USC-2 card.

Marking On The Fly
If the checkbox "Enable" is activated then clicking on "Settings..." will open the Marking On The Fly Dialog. Please refer to chapter: Option MOTF.

Analog In

If enabled then by clicking on "Settings..." a new window pops up. There an Analog Input signal can be used that affects the scanner position.

Test Laserports

DAC A/B:
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 value is 4095.

8 Bit LP:
Test the 8 Bit Laserport. Enter a value from 0 to 255 here. This can be observed with a DB-37 Diagnostic Board attached to the USC-2 card.

Test IO

OptoOut/In:
Test if the bits of OptoOut and OptoIn can be set correctly.

ExtOut/In:
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

YAG, CO2, LEE, IPG:
Choose the type of the laser here.

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

Invert

Laser Gate, Laser A, Laser B:
Here the laser bit signals can be set to be Active Low or Active High.

Laserport
Defines the laser power port. For bitmaps which are amplitude-modulated the analog port B, DA2, is always used.

8-Bit, DAC A, DAC B:
Choose between three possible input signals for the laser power.

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 closing of 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 stand-by mode.

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

Length:
Q-Switch length in µs.

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.

AlwaysActive:
Stand-by mode is also used when the laser signal is on.

Optimize

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

Pixel Grayscale

PixelAM:
Enables Amplitude Modulation.

Laserport:
If checked the selected Laserport will be used for the output of PixelAM, else port DA2 is taken

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

Buttons

Store:
Store 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.

6.2.4 RTC3 Card Settings

Requires RTC3 scanner card

If the registered scanner card is a RTC3 card, pressing the "Driver Settings" button in the menu "Settings -› System -› Card -› Advanced..." the following dialog appears. Here all the card settings have to be done.

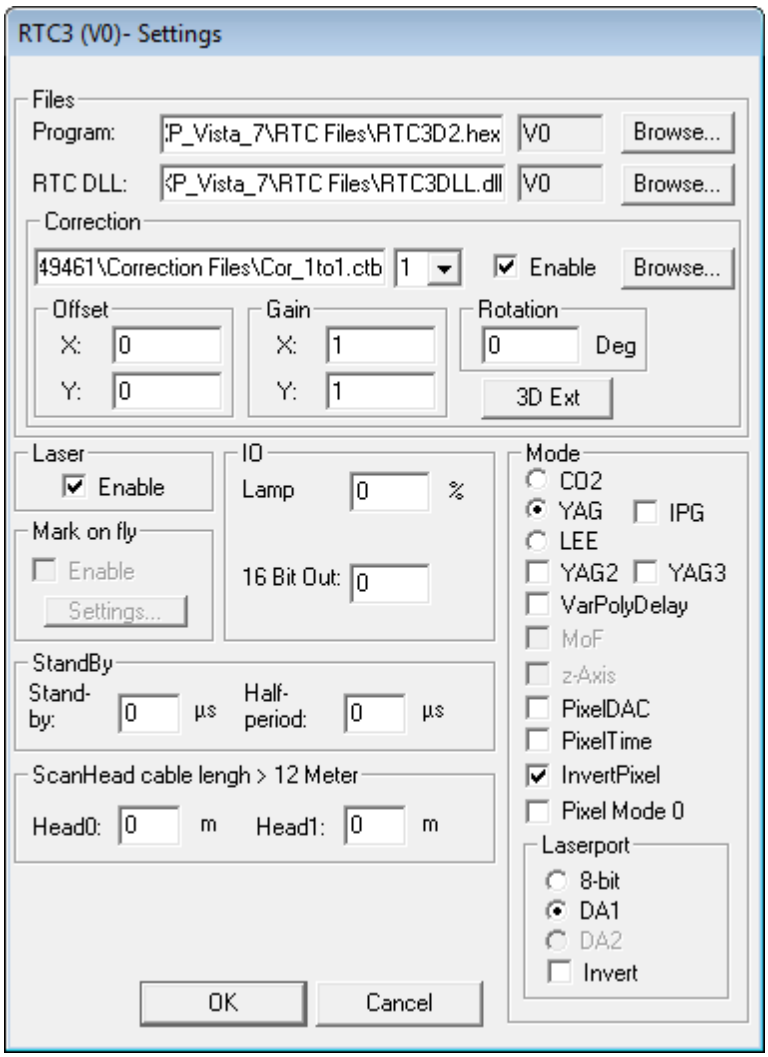


Figure 6.9: Card Settings for RTC3 Card

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.

Correction
Specifies 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 2 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:
A dialog pops up where a Z-Table can be defined. For detailed information please see the RTC3 manual.

Laser
Globally enables/disables laser output.

Mark on fly
If enabled, the *Settings...* button gets activated. For the settings see chapter "Option MOTF".

StandBy
Globally specifies the stand-by pulses.

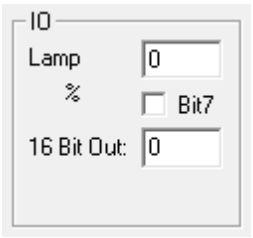
Stand-by:
Q-Switch length in µs for stand-by modus. If this is set to zero the stand-by mode is switched off.

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

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

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.

If *LEE Mode* is selected, the eighth Bit is selectable separately:



Mode

IPG:
This mode has been added recently. IPG mode is now also available for the RTC3 card.

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.

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.

6.2.5 RTC4 Card Settings

Requires RTC4 scanner card

If the registered scanner card is a RTC4 card, pressing the "Driver Settings" button in the menu "Settings -› System -› Card -› Advanced..." the following dialog appears. Here all the card settings have to be done.

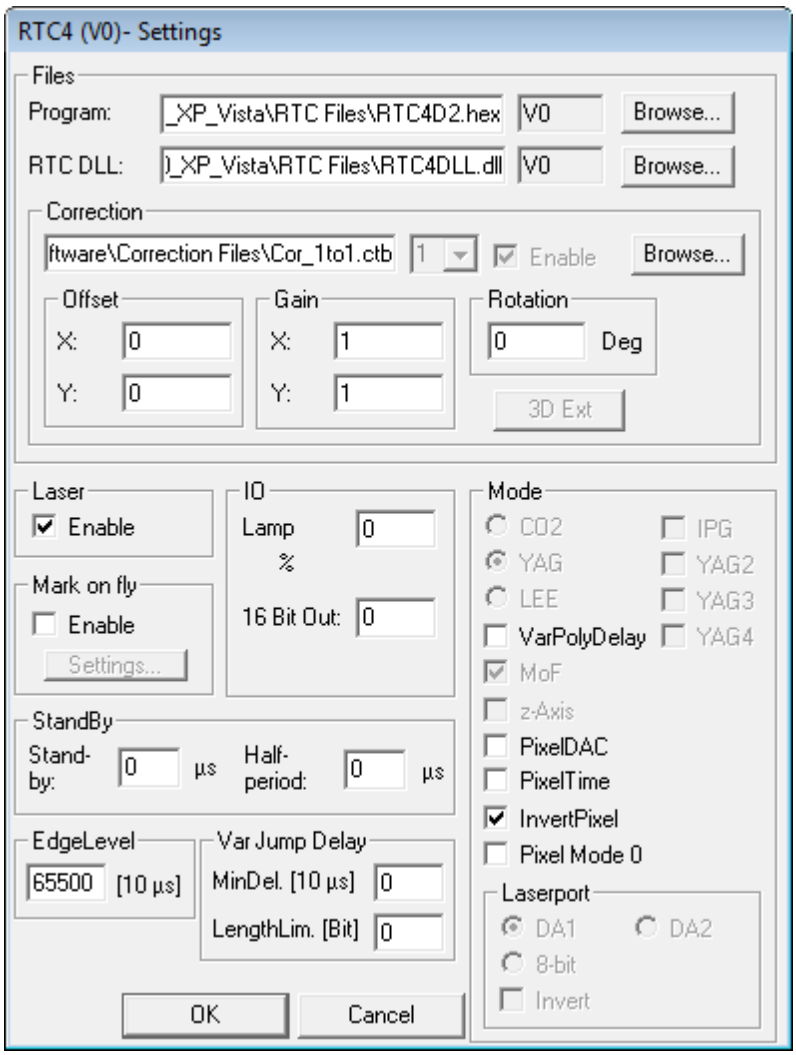


Figure 6.10: Card Settings for RTC4 Card

Files

Program:

Specifies the program file. This file is delivered together with the scanner card. The standard extension is *.HEX. Click on 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 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 2 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:
A dialog pops up where a Z-Table can be defined. For detailed information please see the RTC4 manual.

Laser
Globally enables/disables laser output.

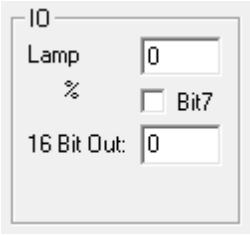
Mark on fly
If enabled, the *Settings...* button gets activated, for the settings see chapter Mark on Fly.

StandBy
Globally specifies the stand-by pulses.

Stand-by: Q-Switch length in µs for stand-by modus. If 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*. Stand-by Settings for pens will overwrite these settings if enabled for a pen as soon as this pen is used.

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 respectively.

If *LEE Mode* is selected, the eighth Bit is selectable separately:



Mode
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.
z-Axis: Indicates whether the card is able to do 3D Marking.
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.

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

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.

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 Edge Level Value can be defined. If the Polygon Delay between two Mark or Arc 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 it starts a new Polygon Marking with the second vector.

Var Jump Delay
Normally after a Jump command a constant Jump Delay is inserted. But for very short 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.

6.2.6 RTC5 Card Settings

Requires RTC5 scanner card

If the registered scanner card is a RTC5 card, pressing the Advanced... button in Menu->Settings->Card the following dialog appears. Here all the card settings have to be done.

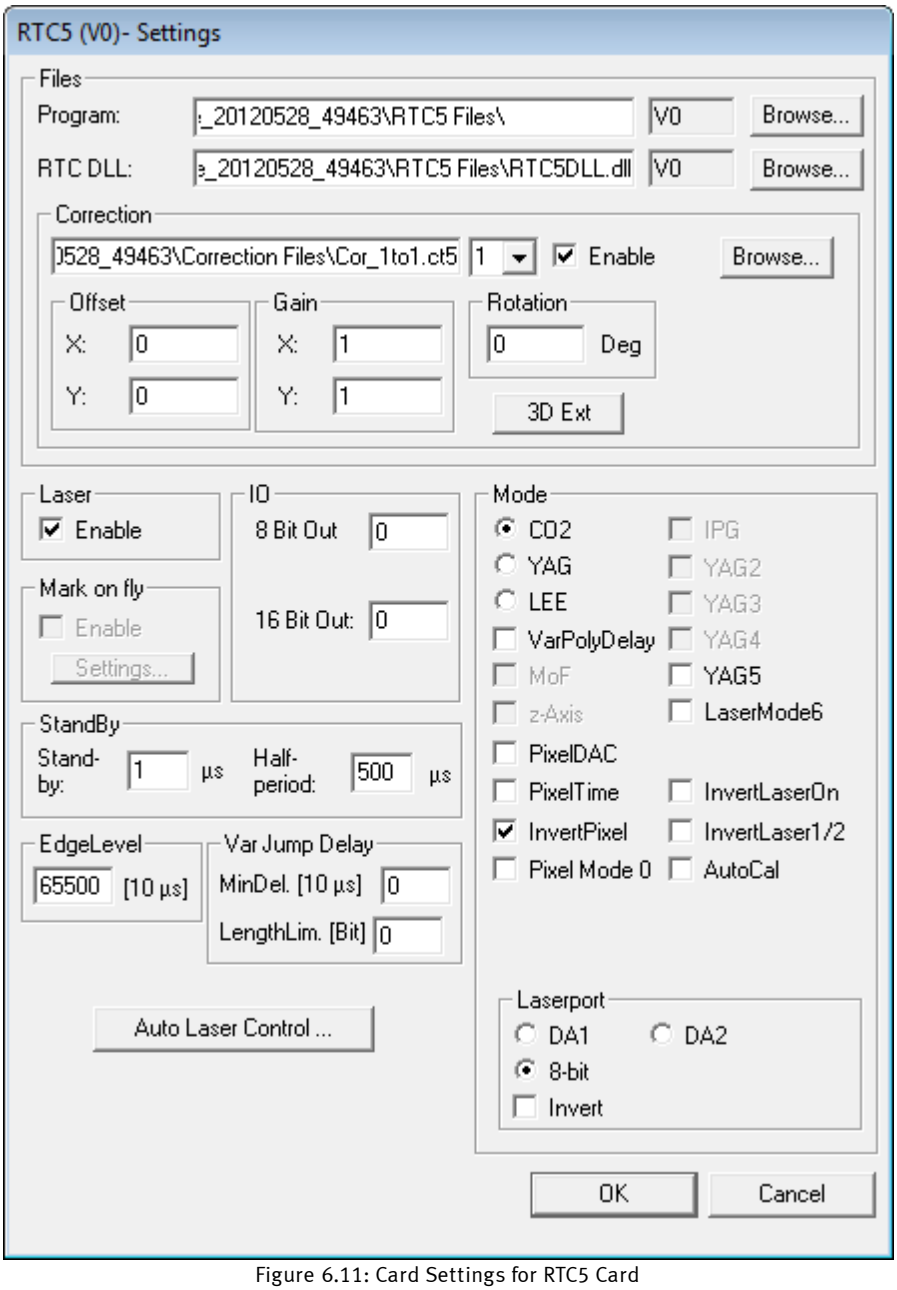


Figure 6.11: Card Settings for RTC5 Card

Files

Program:
Specifies the folder that contains the program file. This file is delivered together with the scanner card. Click on the Browse button to make a search on the folder location. The name of the folder can be "RTC5Files".

RTC Dll:
Specifies the Dll file. This file is delivered together with the scanner controller card.

Correction
Specifies 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 2 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.

Attention: For the RTC5 card the x and y Gain values have to be equal.

3D Ext:
A dialog pops up where a Z-Table can be defined. For detailed information please see the RTC5 manual.

Laser
Globally enables/disables laser output.

Mark on fly
If enabled, the *Settings...* button gets activated, for the settings see chapter Mark on Fly.

StandBy
Globally specifies the stand-by pulses.

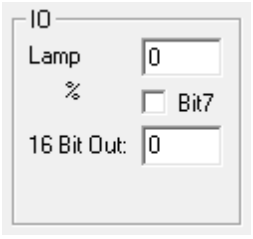
Stand-by:
Q-Switch length in µs for stand-by modus. If zero, the stand-by mode is switched off.

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

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

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 respectively.

If *LEE Mode* is selected, the eighth Bit is selectable separately:



Mode

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.

z-Axis:
Indicates whether the card is able to do 3D Marking.

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)

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

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.

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 Edge Level Value can be defined. If the Polygon Delay between two Mark or Arc 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 it starts a new Polygon Marking with the second vector.

Var Jump Delay
Normally after a Jump command a constant Jump Delay is inserted. For very short Jumps however it is not necessary to have such a longJump Delay. The Jump Delay can be reduced without loosing marking quality. The Miminum Delay is the delay for a Jump of length 0.

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

6.2.7 RTC ScanAlone Card Settings

Requires RTC ScanAlone scanner card

If the registered scanner card is a RTC ScanAlone card, pressing the Advanced... button in Menu->Settings->Card the following dialog appears. Here all the card settings have to be done.

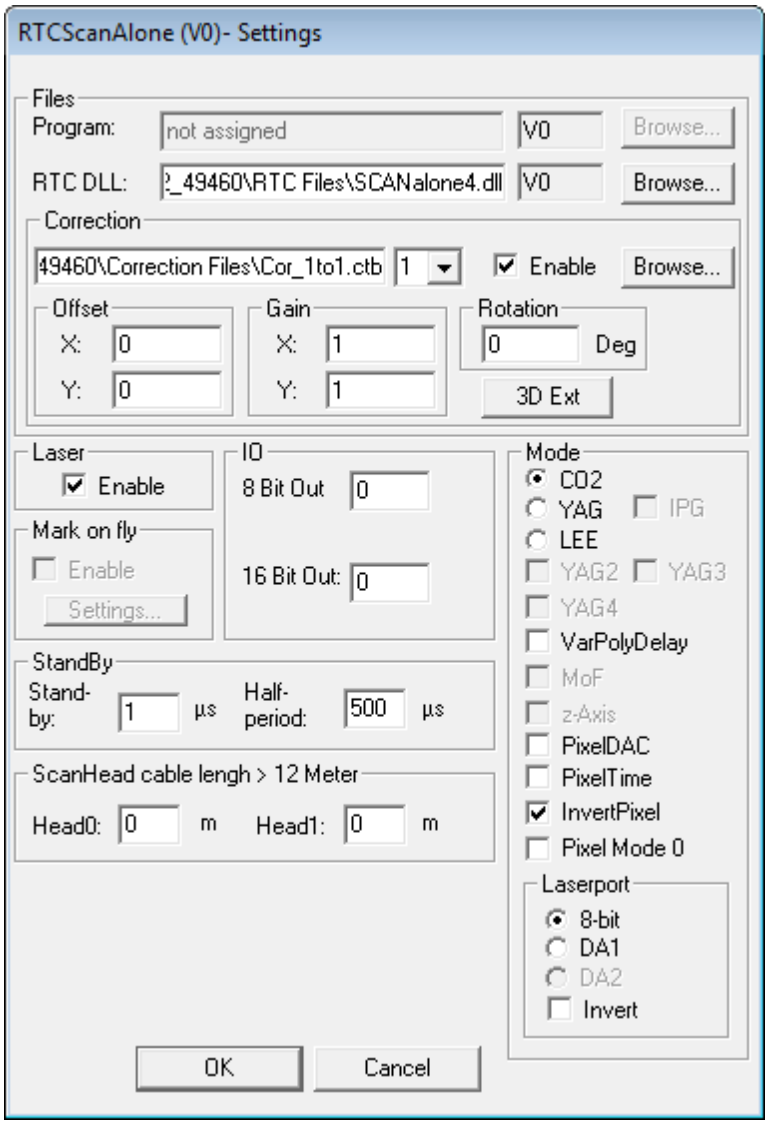


Figure 6.12: Card Settings for ScanAlone Card

Files

Program:
For this card type a program file must not be defined.

RTC Dll:
Specifies the Dll file. This file is delivered together with the scanner controller card.

Correction
Specifies 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 2 different correction files. This is mainly used in connection with the secondary head feature of the RTC ScanAlone.

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:
A dialog pops up where a Z-Table can be defined. For detailed information please see the RTC ScanAlone manual.

Laser
Globally enables/disables laser output.

Mark on fly
If enabled, the *Settings...* button gets activated, for the settings see chapter Mark on Fly.

StandBy
Globally specifies the stand-by pulses.

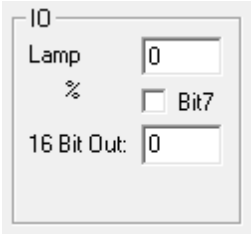
Stand-by:
Q-Switch length in µs for stand-by modus. If zero, the stand-by mode is switched off.

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

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

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 respectively.

If *LEE Mode* is selected, the eighth Bit is selectable separately:



Mode

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.

z-Axis:
Indicates whether the card is able to do 3D Marking.

PixelDAC:
Enables Amplitude Modulation.

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

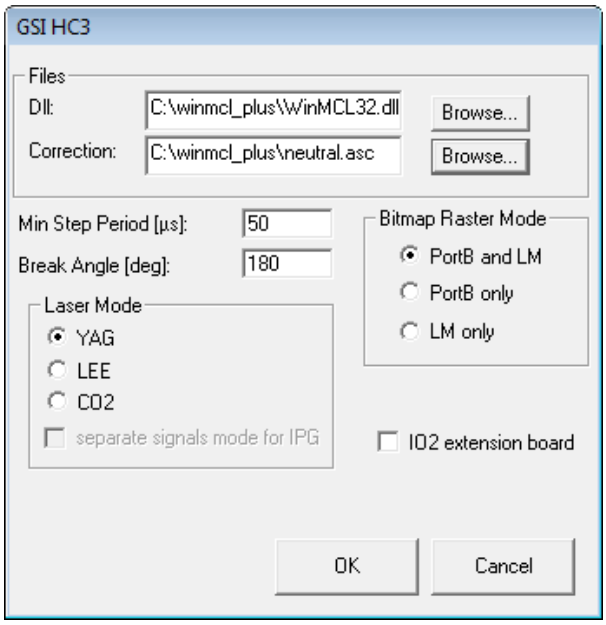
InvertPixel:
Inverts bitmap pixels.

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.

6.2.8 HC3 Card Settings

If the registered scanner card is a HC3 card, pressing the Advanced... button in Menu->Settings->Card the following dialog appears. Here all the card settings have to be done.



Files

Dll:
The path and filename of the WinMCL Plus dll.

Correction:
The path and filename of the correction file.

Min Step Period:
The minimum time period for each micro vector.

Break Angle:
If the angle between between two successive vectors is equal or higher than the break angle, the software will insert a mark delay instead of a polygon delay.

Figure 6.13: Card Settings for HC3 Card

Bitmap raster mode:
Select output port in bitmap hardware mode.

Laser Mode:
The laser mode can only be changed out of the sc_setup.exe tool.

YAG:
The power of the laser is controlled through 8 bit.

LEE:
The power of the laser is controlled through 7 bit inverted.

CO2:
The power of the laser is controlled through pulse width modulation.

Separate signals mode for IPG:
This indicates that the SSMode registry entry is set. This mode is mainly used for IPG lasers. If the software works in this mode, the limits for Frequency, PulseWidth etc. are read out of the registry.

IO2 extension board:
This has to be checked if the IO2 extension board is mounted on the HC3. If YAG mode is active, the power value controls the analogue output on the IO2 board.

6.3 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 Menu. 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.

Important:
Pen 256 is used for the HomeJump. When the scanner moves to the HomePosition (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).

YAG laser

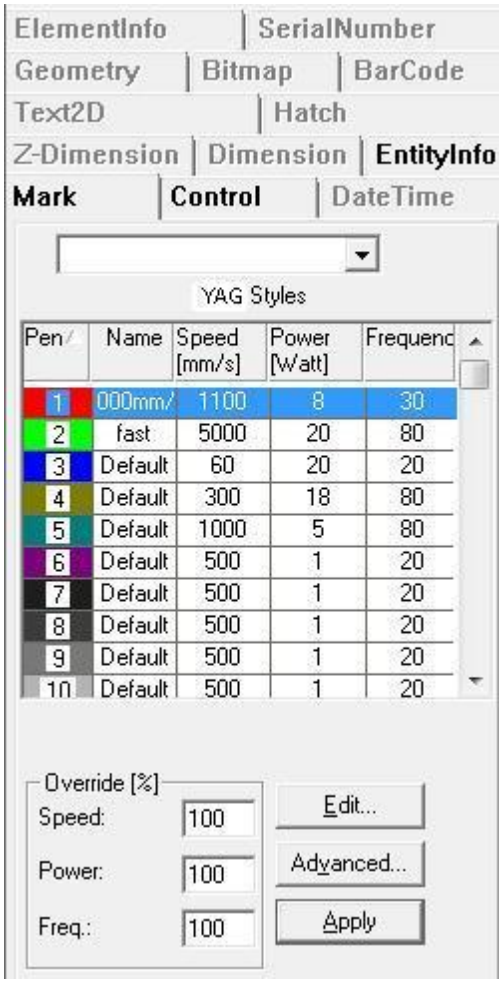


Figure 6.14: Pen Settings for YAG Laser

Name:
Name of the pen

Speed:
Galvo mark speed in mm/s.

Power:
Laser power in Watt. For calibration see chapter Power map.

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.

Advanced...:
Within the advanced dialog settings for the PowerMap and the HomeJumpStyle can be done.

Apply:
Applies the selected pen to the current selected object.

CO2 laser

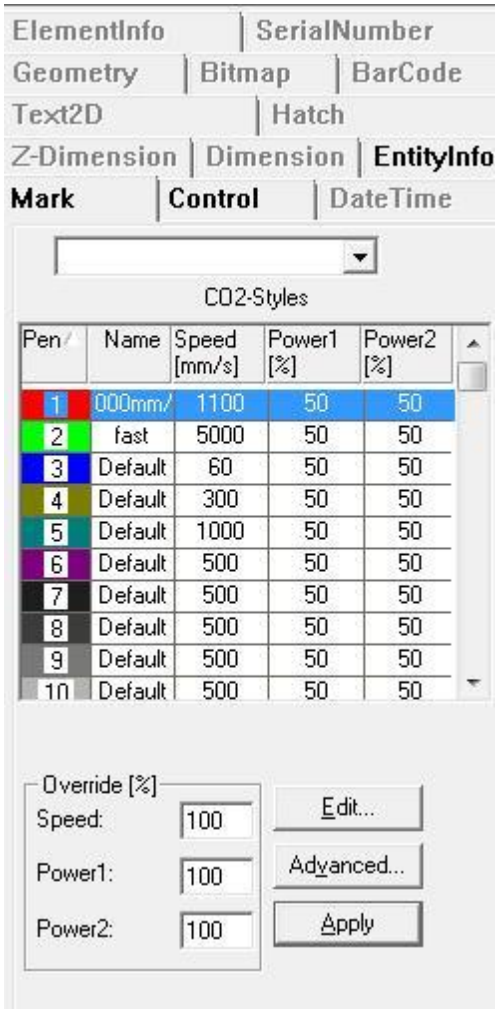


Figure 6.15: Pen Settings for CO2 Laser

Name:
Name of the style.

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.

6.3.1 Edit Pens

Each single pen can be edited by pressing the Edit... button in the mark property page. 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:

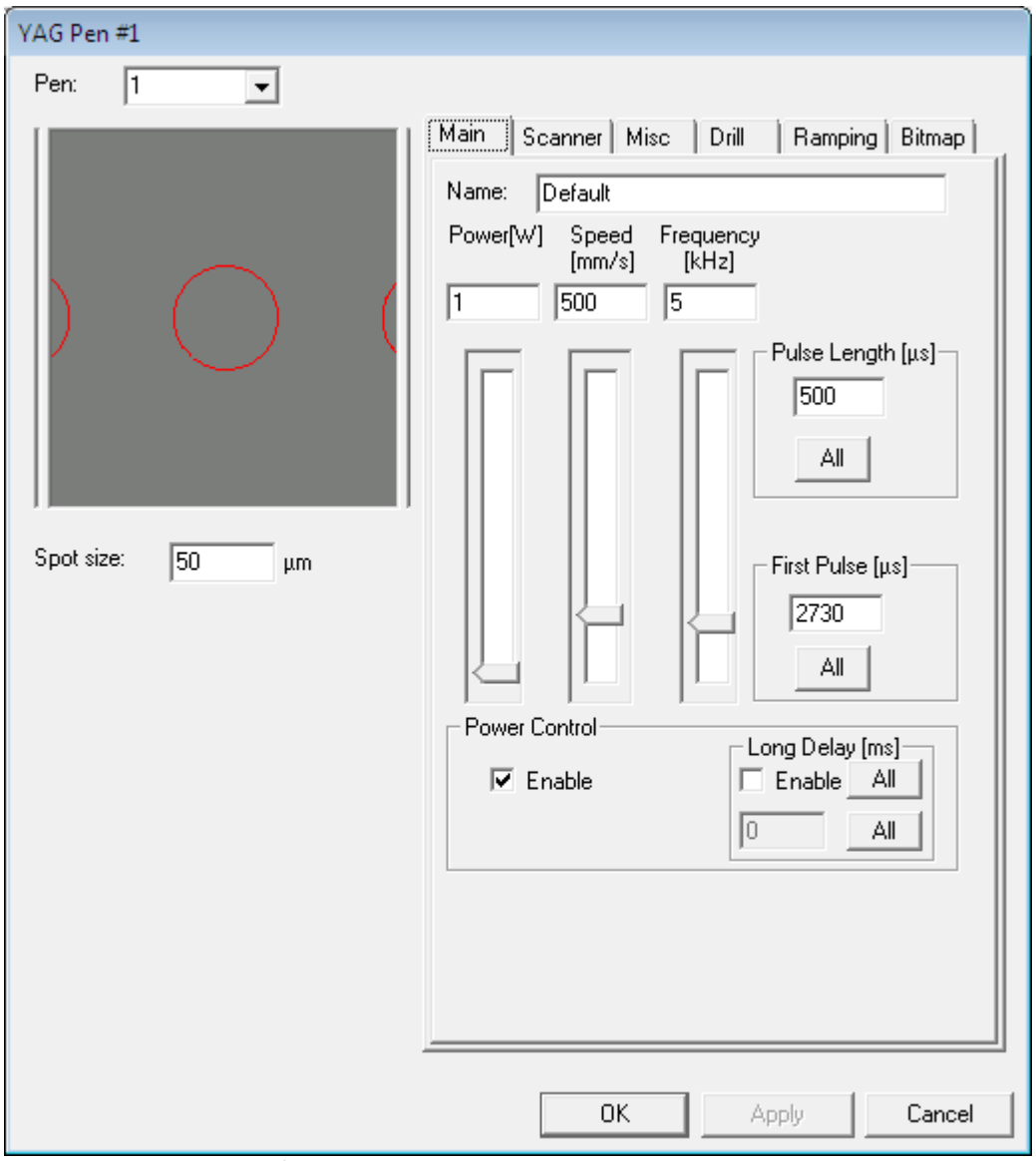


Figure 6.16: Pen Settings Dialog for YAG Laser

Spot size:
This value is used for the display of the spots.

For a description of the main page see Main Settings for Pens.

6.3.1.1 Main Settings for Pens

The following page can be found under Mark->Edit...->Main.

YAG laser settings:

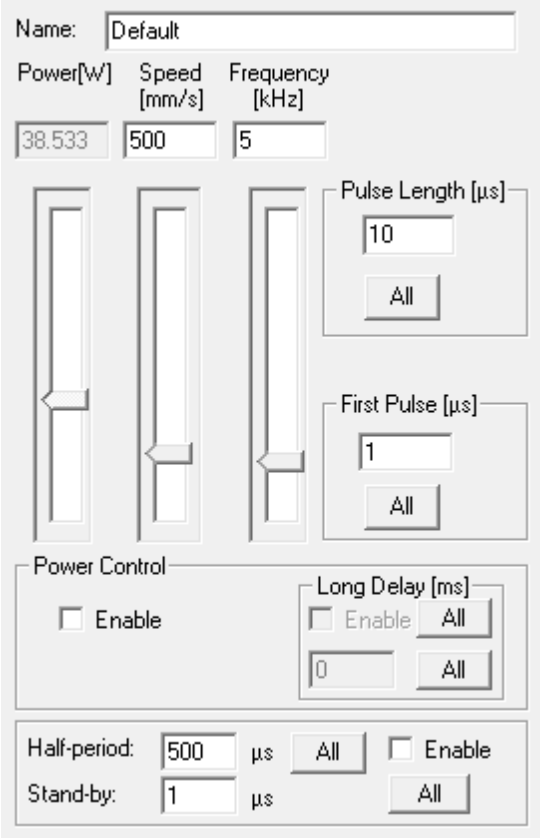


Figure 6.17: Main Pen Settings for YAG Laser

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.

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 LongDelay ms before it will continue.

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

Stand-by:
Q-Switch length in µs for stand-by mode. The stand-by mode can be globally set in the card settings.

CO2 laser settings:

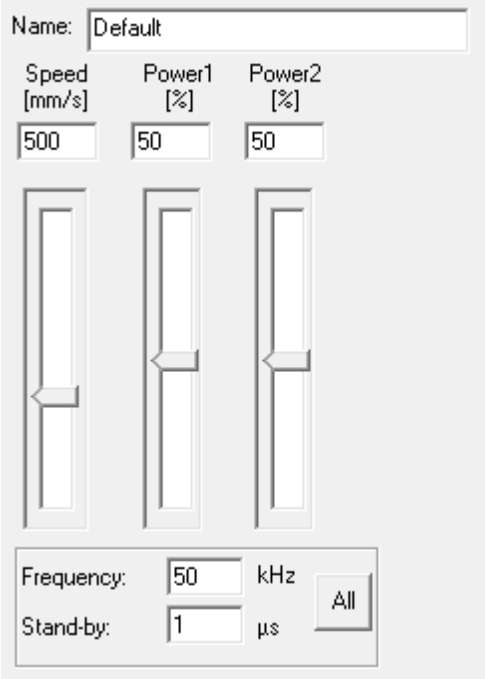


Figure 6.18: Main Pen Settings for CO2 Laser

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.

6.3.1.2 Scanner Settings for Pens

Following page can be found under Mark->Edit...->Scanner.

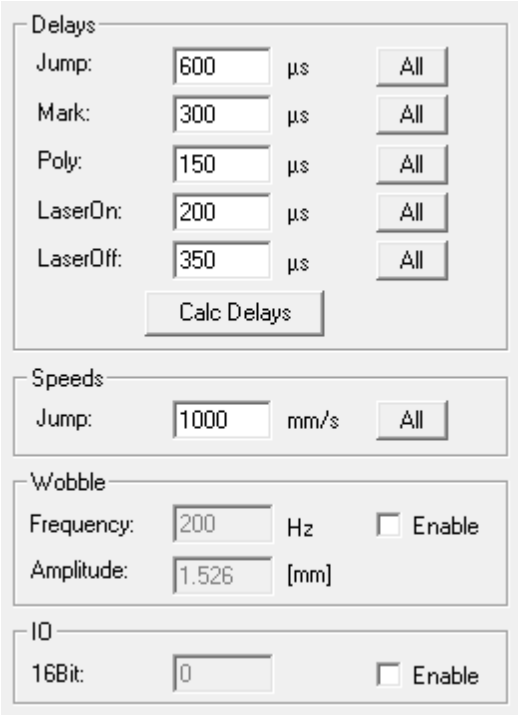


Figure 6.19: Scanner Delay Settings

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.(*)

LaserOff:
The time the controller card waits from the beginning of the output of the last microstep before it switches off the laser.

(*) Laser On:
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.

All:
The *All* buttons apply the related value to all 255 pens.

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 speed of the pen. A big jump speed needs a big jump delay.

Wobble
If enabled linear vectors are marked as circular vectors. The result is a wider line. Frequency and Amplitude have to be coordinated.

IO
Each pen can send an 8 Bit signal entered to the 8Bit Port. For an example see USC-1 Optic settings.

Pixel

Hardware:

For the illumination of scanner images generated in gray-scale mode the hardware mode has to be set, for more information see the Bitmap description.

Remarks:

The scanner delays, which are the jump, mark and poly delay, 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. After that the scanner delays have to be defined. Some conditions should be considered:

1. *LaserOff delay › LaserOn delay*

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 not be smaller than the LaserOn delay.

2. *Mark delay › LaserOff delay - LaserOn delay*

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.

Also see:

More detailed explanations under Scanner and Laser delays.

6.3.1.3 Miscellaneous Settings for Pens

On this page skywriting can be defined for a pen. To achieve very exact marked vertices the scanner starts and stops scanning movement outside of the marked vectors. The property page to define skywriting can be found under Mark->Edit...->Misc.

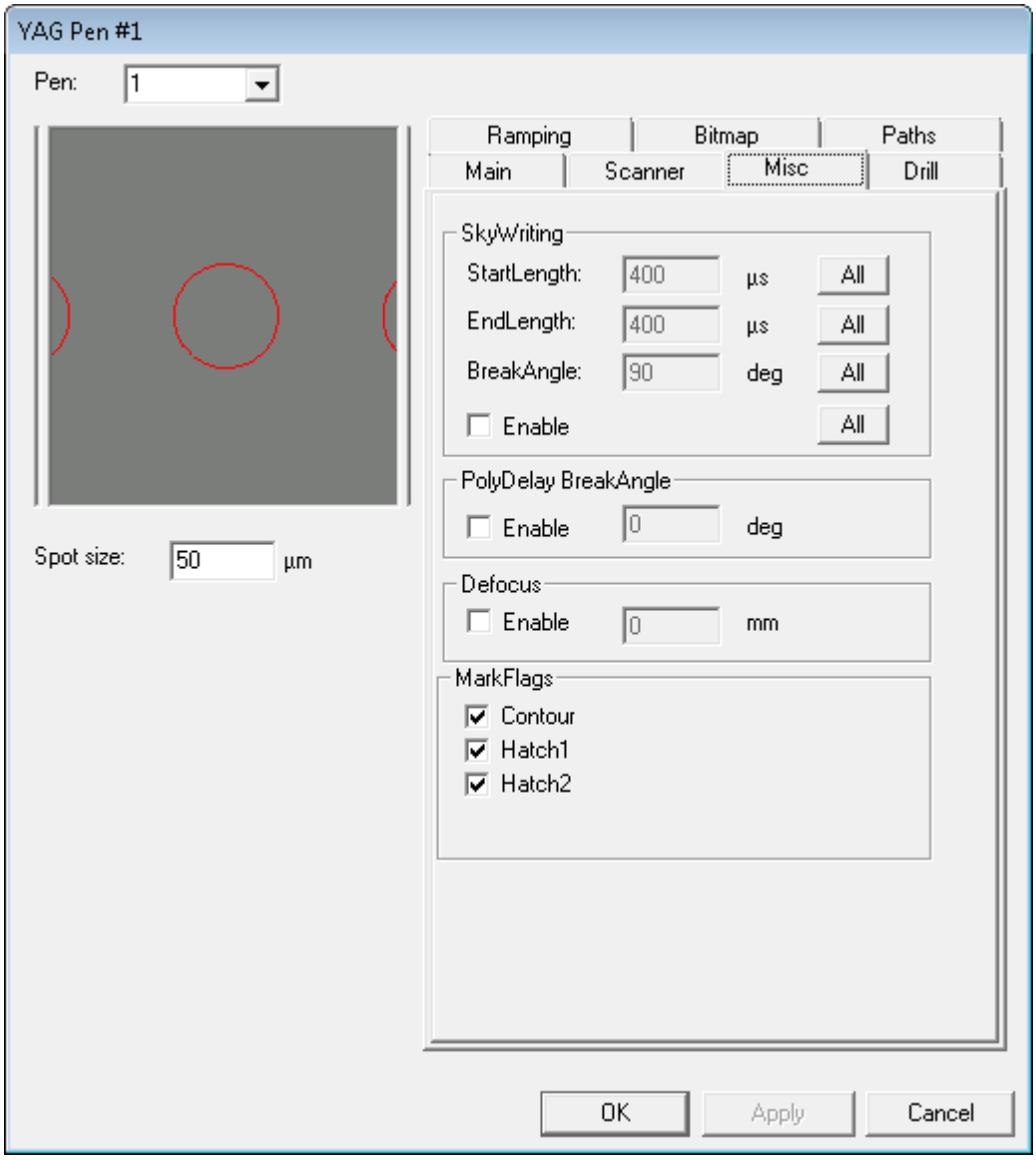


Figure 6.20: Misc Pen Settings

SkyWriting

SkyWriting makes the vectors of a polygon slightly longer than normal. The parameters can not be defined totally freely. They have to be adjusted regarding the current laser and scanner delays. As a simple rule please do not use a value larger than Poly Delay / 2 for the SkyWriting values. So for example: If you have a Poly Delay of 200 us then the SkyWriting Lengths should not exceed 100 us.

Further you have to regard the following:

It is believed that the skywriting mode turns off the laser in time at the corner and turns it on again at the right time at the beginning of the new line. But this is not true. The skywriting Start Length is linked with the LaserOn Delay and the skywrting End Length is linked to the LaserOff Delay.

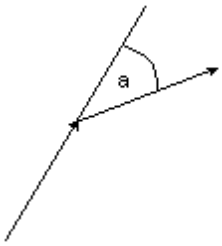
Now the LaserOn delay has to be adjusted so that the skywriting Start Length is so big, that the laser will be turned on exactly at the beginning of the new line. If the laser is turned on too early, then you should increase the Laser On Delay or decrease the skywriting Start Length. If the laser is turned on too late then do the opposite.

Similar for the Laser Off Delay and the the skywriting End Length. Please adjust parameters so that the laser is turned off exactly at the end of the line. Either you change the Laser Off Delay or you change the skywriting End Length. So for example, if the line extends over the end of the corner reduce Laser Off Delay or increase the skywriting End Length.

StartLength:
Scanner head starts before marking vector.

EndLength:
Scanner moves after marking vector.

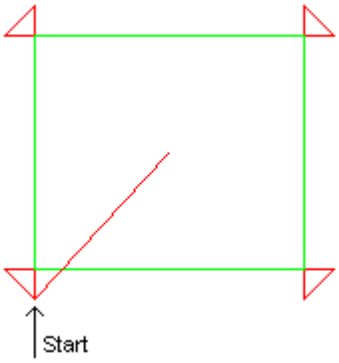
BreakAngle:
Only relevant for PolyLines: Skywriting is enabled if angle 'a' between two successive vectors is bigger than or equal to the entered BreakAngle.



All:
The *All* buttons applies the value to all 255 pens.

Enable:
Enables skywriting for marking.

Example:



Note: The SkyWriting parameters can also be used for bitmaps in dithered mode (BreakAngle

is irrelevant in that case).

MarkFlags:
Contour or Hatch1/2 can be activated for every pen.

6.3.1.4 Drill Settings for Pens

The following page can be found under Mark->Edit...->Drill. If enabled, single points are scanned with drill mode.

YAG laser settings:

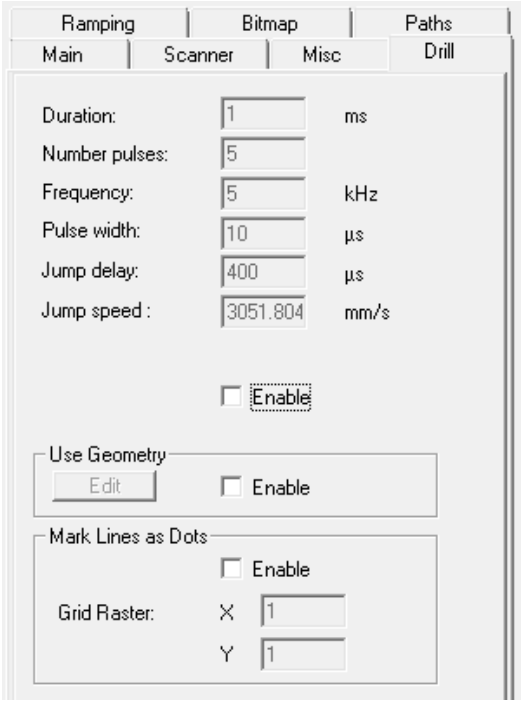


Figure 6.21: Drill Settings for YAG Laser

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.

CO2 laser settings:

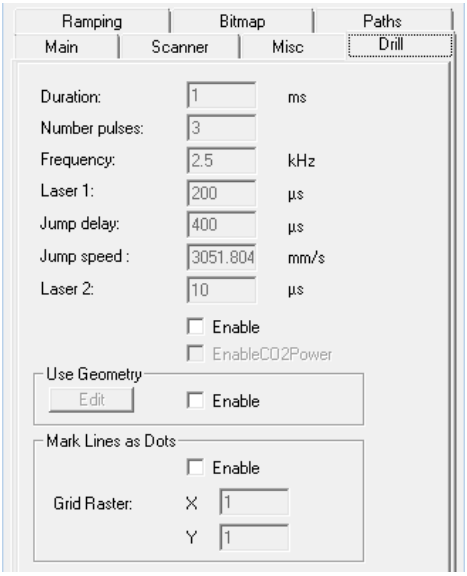


Figure 6.22: Drill Settings for CO2 Laser

Laser 1:
Pulse length of laser signal1 in µs.

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.

6.3.1.5 Ramping Settings for Pens

The following dialog can be reached under 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.

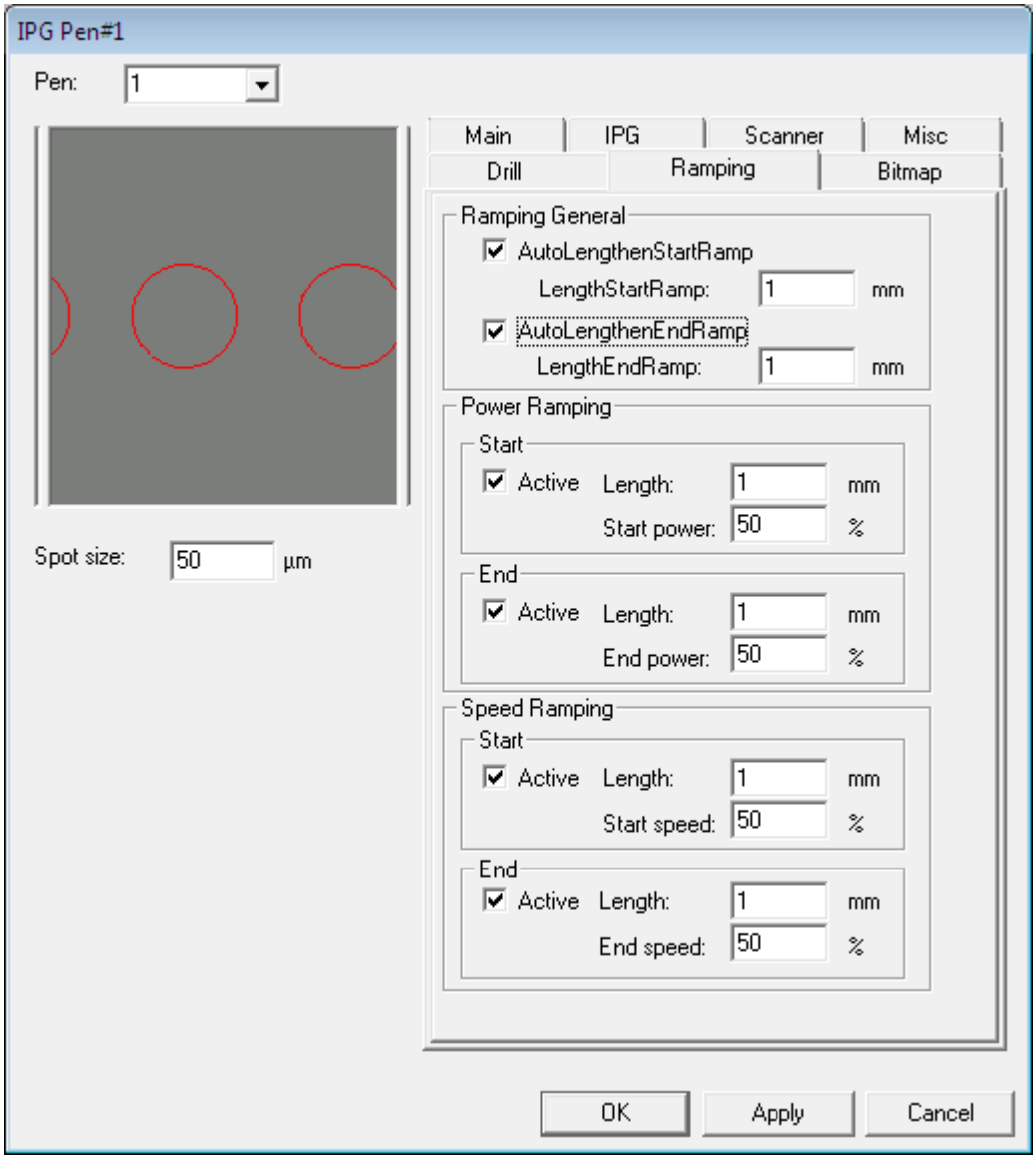


Figure 6.23: Ramping Settings Dialog

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.

6.3.1.6 Bitmap Settings for Pens

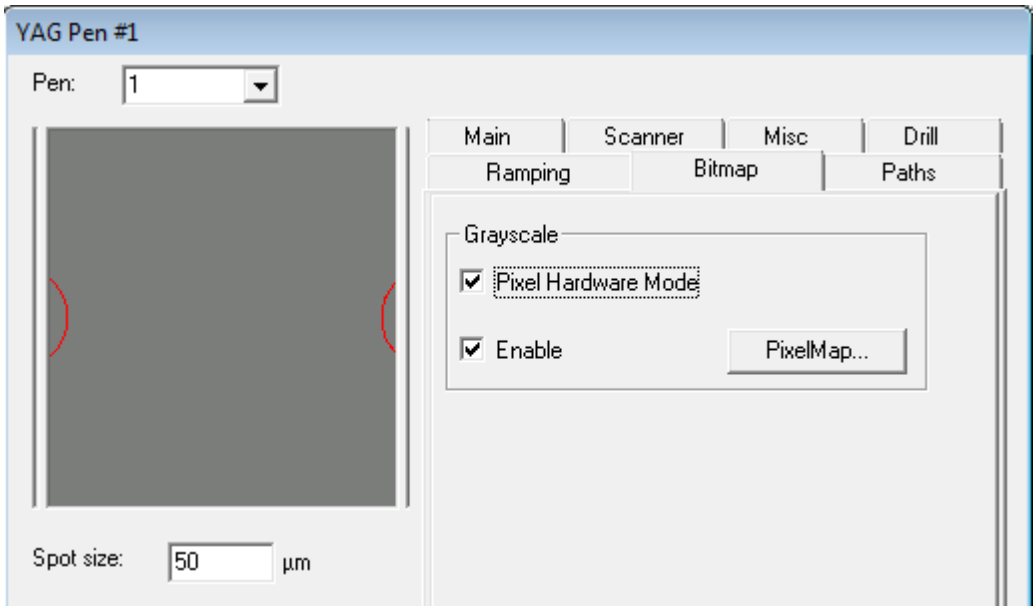


Figure 6.24: Bitmap Settings Dialog

Pixel Hardware Mode
This checkbox should be enabled if using a gray scale bitmap.

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.

6.3.1.7 Pen Paths

The Pen Path property sheet can be opened via the Entity Property Sheet "Mark" by doubleclicking on a pen or by clicking the Edit button. The appearance is as follows:

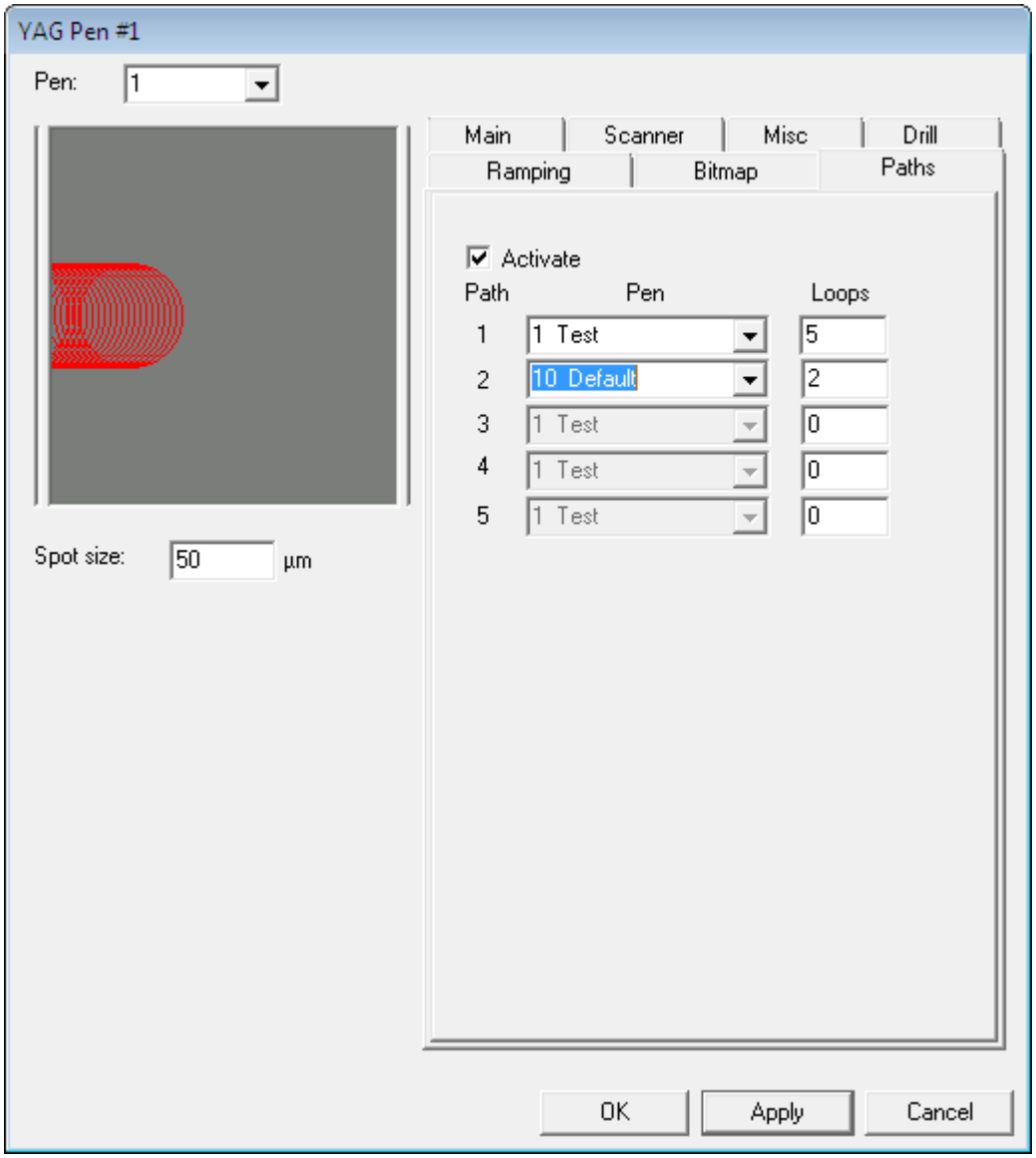


Figure 6.25: Pen Paths Dialog

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.

6.3.2 Pen Advanced

The following dialog appears when pressing the Advanced... button on the mark property page.

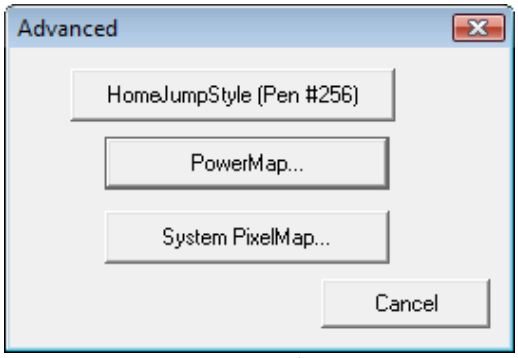


Figure 6.26: Pen Advanced Dialog

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.

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.

6.3.2.1 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:

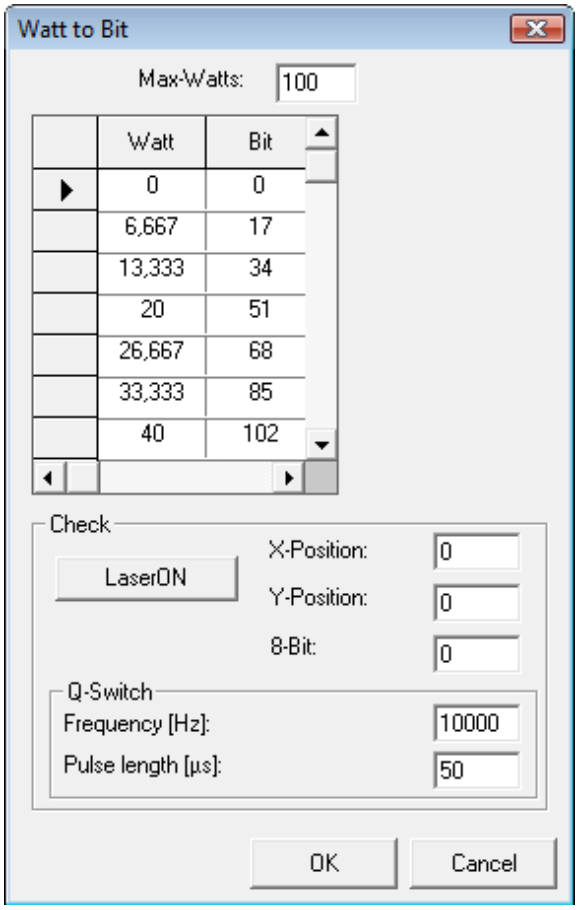


Figure 6.27: Power Map for YAG Laser

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.

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:

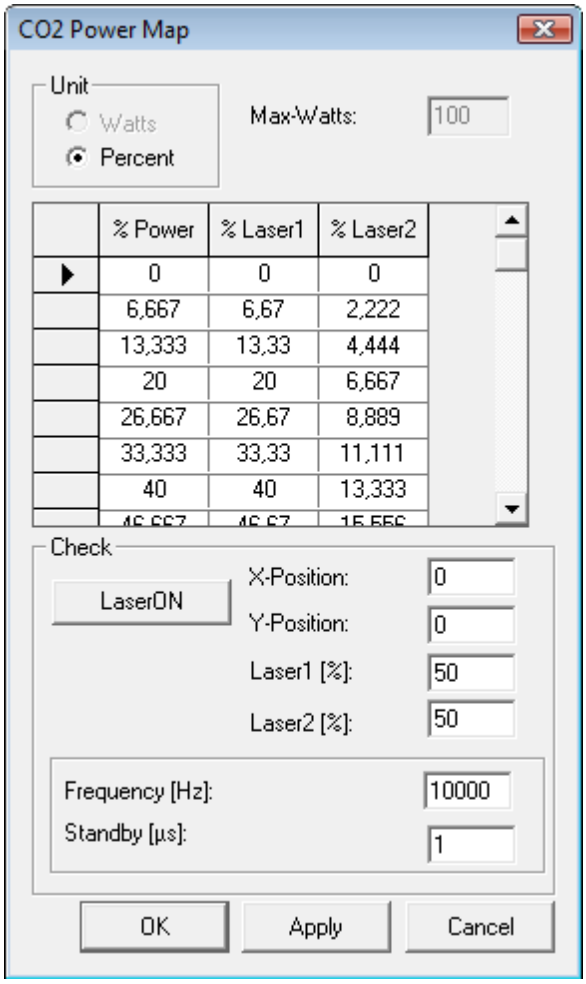


Figure 6.28: Power Map for CO2 Laser

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.

7 External Control

To control the scanning process it is possible to define output signals and wait for special input signals of the I/O-Port. Additional signals can be sent via RS232 to define a move of an external axis in z-direction for example. A Programming Interface is provided to control the application from outside.

 Note: For the RTC3 scanner card the I/O signals are only available if the RTC3 I/O Extension Board is connected.

7.1 IO Modes

This chapter describes how to use special IO modes to manage marking processes via external control.

7.1.1 IO Job Selection

If the Job Select I/O Mode is active, 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 that is loaded this 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 jobfile:

- [OPTO_IN_0] StartMark
- [OPTO_IN_1] StopMark
- [OPTO_IN_2 - OPTO_IN_5] 4* job number, range of the number: 1 - 15

The jobfiles that have to be loaded depending on the input signals need to be saved in the folder "sam2d\jobfiles" of the installation directory and they must have the following name structure:

x...x_nnnn.sjf.

Here **x...x** stands for any freely definable characters and **nnnn** stands for the job number that is related to the input pin pattern (for example "loadjob_0001.sjf" is a valid filename for the first job).

Remark: Job number 0 is defined as an empty job. Jobfiles with this number won't be loaded but ignored.

To control the process the outputs are defined as follows:

- [OPTO_OUT_0] MarkingActive
- [OPTO_OUT_3] Software (Trigger) ready -> StartMark can be set
- [OPTO_OUT_4] New job was loaded successfully, signal will be reset with next external start (StartMark)

7.2 Control objects

A control object can be defined in between the chronological process of the job. Provided objects are:

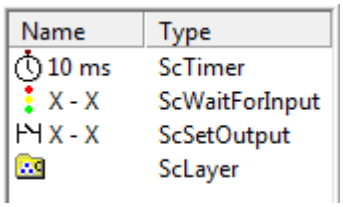
- Timer
- Wait for Input

- Set Output
- Executable
- Control Motion
- Motf Offset
- Wait For Trigger
- AutoCal
- SetOverride

7.2.1 I/O Control Objects

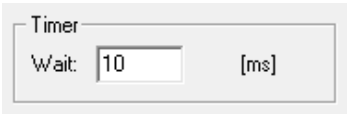
There are three different control objects available. These are "ScTimer", "ScWaitForInput" and "ScSetOutput". The control objects can be created with the toolbar:

When a control object is created it appears in the entity list:



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:



Wait:
Interrupts the marking process for n ms, in this example 10 ms.

ScWaitForInput:

Wait For Input:
Stops the marking process until the specified input Bit(s) of the IO-Port is set to high or low.

X: Don't care / ignore input bit.
0: Wait for corresponding bit state '0'.
1: Wait for corresponding bit state '1'.

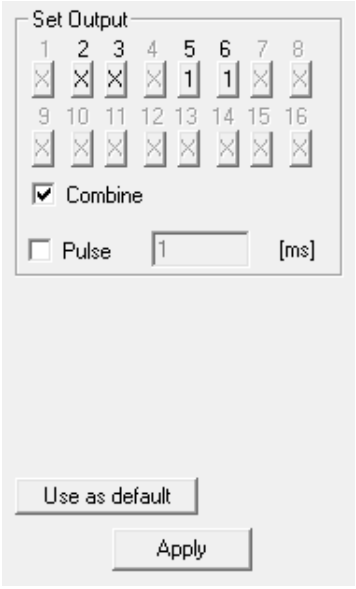
Combine:
If this is selected, it is possible to select a multiple bits state to wait for.

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:



Set Output:
Sets the specified output Bit of the IO-Port to high/low.

X: Don't care / ignore output bit.
0: Set corresponding bit to '0'.
1: Set corresponding bit to '1'.

Combine:
If check button *Combine* is selected, it is possible to select a multiple bits state for output.

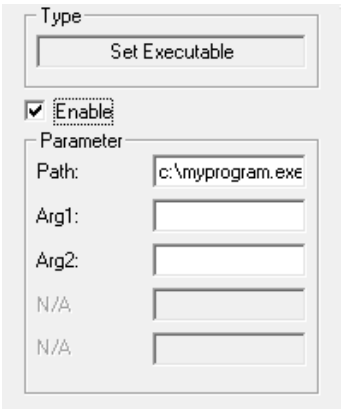
Bit position count starts with '1', but corresponds to bit '0' at the hardware !!!

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)

7.2.2 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 tool bar. 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.



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.

7.2.3 Motion Control

Generate a Motion Control by pressing the  button in the Object Tool bar which is only available if it is activated by the Menu->Settings->System->Extras dialog. The generated object appears in the entity list. 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. Additionally, an output signal can be sent directly.

Control Settings:
The controller needs to be defined in the text-file *sc_motion_settings.txt*. This file can be found in the folder "C:\scaps\scaps_sam\system".

To configure the motion settings file the following options are available:

Type=y - the type of the motion controller, where y stands for a number that specifies the controller:

1 - IMS Stepping Drives
3 - MicroCon
4 - external custom controller
5 - IMS MDrive
6 - Faulhaber motion controller
7 - stepper motor via Isel IT controller / DNC
8 - generic stepper motors that use signal lines for step and direction (reference switch optionally)
9 - generic RS232 interface
13 - CanApi Isel controller

DeviceName=ssss - this option is used only when an external, custom controller is used. Here ssss specifies the name of the device that is identical to the base-name of external DLL that has to be loaded during runtime to access the motion controller functionality. Here the DLL has the name sc_motion_control_ext_ssss.dll and the file where the device is configured is named sc_motion_ssss_settings.txt

The Motion Control Entity ScMotionControl

In the following the maximum number of features and functionalities of the ScMotionControl entity are described. Depending on the capabilities of the used motion controller some of the functions may be not available.

Figure 7.1: Motion Control Dialog

Move

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

Rel:
If chosen the distance will be added to the actual position.

Go:
Pressing this button the entered move control signal is sent.

Stop:
Stops the movement.

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:
Clicking on this button opens a dialog where the motor can be moved in either direction by small steps:

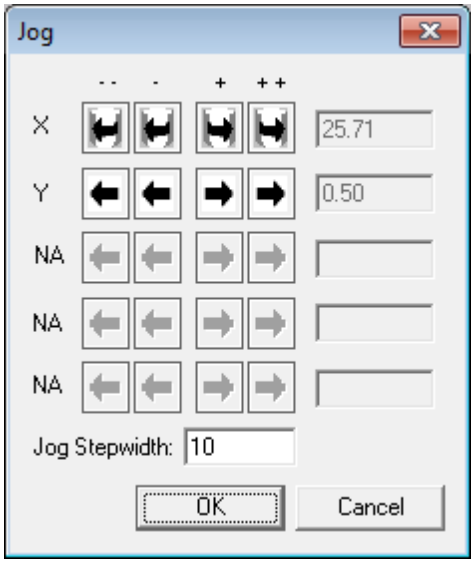


Figure 7.2: Jog Dialog

The default Jog Stepwidth value can be set in the menu Settings -› System -› Extras. If the dialog is being closed and being opened again, this value will be restored to the value that has been set in the menu.

Home:
Move to the home position.

↯ Movements are only possible if hardware for stepper motor is installed.

String Mode

Enable:
If chosen the selected motion control object is defined as a string control instead of a motion control. Enter a string in the edit field and press *Apply* to assign a string to the motion control object.

Send:
Pressing this button the string in the edit field will be sent.

Apply:
The Apply button appears only if a motion control object is selected. Pressing this button assigns the settings to the selected motion control object.

7.2.3.1 Ims Motion Control

After leaving the program an *ims_settings.txt* file is created in the folder "scaps\scaps_sam\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"
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.

Remark:
- 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.

7.2.3.2 MDrive Motion Control

Using the IMS MDrive Motion Controller

Hardware Setup

Additional to the settings definable within the sc_motion_mdrive_debuglog.txt like described below, there are some settings which need to be stored durable into the MDrive controller to work properly. This is the party mode variable, the motion drive name and the echo mode. The echo mode needs to be set to 2.

The IMS MDrive motion controller series is supported directly. To enable it, please configure the MDrive type 5 like it was described above.

For configuring the MDrive controller interface, a configuration file **sc_motion_mdrive_settings.txt** is required in the same location where the general motion settings file exists. The MDrive 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 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. The 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 MDrive 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 IMS MDrives is 9600.

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

TimeOut=yy

The "TimeOut" 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 isn't successful.

Debug=y

For special debugging purposes it is possible to log several information into a file **sc_motion_mdrive_debuglog.txt** that will be created at the position where the settings file is located at. With the value 1 ("Debug=1") the debugging mode is turned on, when it is set to 0 ("Debug=0") debugging is disabled and no data are put into the log file. Please note: 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 party mode where it is possible to handle up to three axes using this controller interface. Each of these axes can be configured in a different way. To do so, 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 three times, separated by the different calls of "axis":

axis=y

Specifies for which axis the following configuration parameters are valid for. If this statement isn't used somewhere in a configuration file it is assumed that the third axis is used. If it is used, the party-mode is enabled immediately so that no additional calls of the "PY"-command have to be made somewhere.

mode=POSITION

The MDrive 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. Please note: 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 which is 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 being operated in party mode, that means with more than only one MDrive 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 character and it of course has to be configured and saved directly at the MDrive 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. If 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). Else the motion controller interface doesn't 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 its initialization, the "ival" parameter can be used. Different from 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.

If there is an initialization value "ival=PY 1" the interface sends this parameter to the controller and internally enables the syntax for party mode. On the other hand, if "ival=PY 0" is found, the interface sends this value and disables the party mode syntax internally. That method of switching party mode on and off is not supported within a motion controller program. It can be executed only directly. If the "axis" command (see above) is used to configure and use more than one axis with this interface, this command mustn't be used. Here the party mode syntax is activated by default and has to be enabled by default at the motion controller hardware too.

hval=ssss

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. These are executed in the same order as they appear in the configuration file.
If there is a initialization value "hval=PY 1" the interface sends this parameter to the controller and internally enables the syntax for party mode. On the other hand, if "hval=PY 0" is found during sending the homing command sequence, the interface sends this value and afterwards disables the party mode syntax internally. That method of switching party mode on and off is not supported within a motion controller program. It can be executed only directly. If the "axis" command (see above) is used to configure and use more than one axis with this interface, this command mustn't be used. Here the party mode syntax must stay activated during runtime.

The original sc_motion_mdrive_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 un-commented. 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.

7.2.3.3 Faulhaber Motion Control

Using the 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. Please note: 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.

7.2.3.4 Isel Motion Control

Using the Isel IT Stepper Motor Motion Controller

The Isel stepper motor motion controller series of type "IT" is supported directly. That special kind of controller works with DNC-commands that are sent via RS232 directly. To enable it, please configure the Isel type 7 like it was described above.

For configuring the Isel controller interface, a configuration file **sc_motion_iselit_settings.txt** is required in the same location where the general motion settings file exists. The Isel 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 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 Isel 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 Isel controllers is 9600.

TimeOut=yy

The "TimeOut" 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 isn't successful.

DeviceNumber=y

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> (Isel CNC OS 5.x). The default value is 0.

Debug=y

For special debugging purposes it is possible to log several information into a file **sc_motion_iselit_debuglog.txt** that will be created at the position where the settings file is located at. With the value 1 ("Debug=1") the debugging mode is turned on, when it is set to 0 ("Debug=0") debugging is disabled and no data are put into the log file. Please note: 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 this motion controller. This is important for the possibility to handle up to three 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 three times, separated by the different calls of "axis":

axis=y

Specifies for which axis the following configuration parameters are valid for. If this statement isn't used somewhere in a configuration file it is assumed that the first (=X) axis is used. The valid range for "y" goes from 0 to 2 for the three axes X, Y and Z. For the Isel controllers it is necessary to have the X-axis (axis number 0) available and configured in every case.

mode=POSITION

The Isel 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 within the scanner application 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. Please note: 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. For the total limits following dependencies are true (related to unit increments): corr1 <= llimit < hlimit <= corr5.
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. For the total limits following dependencies are true (related to unit increments): corr1 <= llimit < hlimit <= corr5.
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. For the total limits following dependencies are true (related to unit increments): corr1 <= llimit < hlimit <= corr5.
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.

hval=ssss

The 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 original sc_motion_iselit_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 un-commented. Other default-parameters may have to be removed or commented out. As it can be seen there for the "hval" parameters, complete command sequences can be defined.

7.2.3.5 Stepper Motor Control

Using the Generic Motion Controller for Stepper Motors

Using this controller type stepper motors can be controlled directly using the IOs or the Laser Port of the scanner card. This controller supports up to three motors that can be driven using a step and a direction signal line plus an optional reference switch via an input. To enable this type of controller, please configure the generic stepper motor type 8 like it was described above. If more than one drive is used movements take place one after another, so that in every case only one axis moves at the same time.

If the reference switch is used there are some limitations: Reference movements are software controlled and have due to the limitations of the host operation system a much lower maximum speed. Nevertheless there is a configuration where the movement to the reference switch can be done as fast as normal movements. To get some more information about that mode please refer to the parameters **refIO**, **refvalue** and **refmode**.

For configuring the stepper motor controller interface itself, a configuration file **sc_motion_stepper_settings.txt** is required in the same location where the general motion settings file exists. The stepper motor configuration file can be used to specify different parameters like the IOs for the different output and input signals, the behavior for the reference switch that is searched during a movement to the home position and others more. The file itself is a plain textfile 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:

Debug=y

For special debugging purposes it is possible to log several information into a file **sc_motion_stepper_debuglog.txt** that will be created at the position where the settings file is located at. With the value 1 ("Debug=1") the debugging mode is turned on, when it is set to 0 ("Debug=0") debugging is disabled and no data are put into the log file. Please note: there can be huge amounts of data that are logged into such a file so enable this option only if it is necessary!

DisableHomingDuringStartUp=y

The possible values of DisableHomingDuringStartUp are 0 (default) and 1. If homing has been activated (refmode≠0) and a homing movement is not desired during software start, this parameter has to be set to 1. Homing can then be performed after a software start by pressing the 'Home' button in the Control property page.

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

axis=y

Specifies for which axis the following configuration parameters are valid for. If this statement isn't used somewhere in a configuration file it is assumed that the third axis is used.

mode=POSITION

The stepper 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. Please note: 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

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.

stepIO=y

This parameter specifies which output number has to be used to send the step-clock to the motor. The value given for "y" is a number that specifies the output port of the scanner card and the output

pin the clock-input of the motor is connected with. If a value in range **1..5** is entered here, the standard output is used. When the value is in range **100..107** the pins 0..7 of the laser port are used for sending the step signal. Please note: the last method can be used only if the laser port is NOT used for controlling the laser. Otherwise the results can be undefined and may harm your equipment seriously. If you are using an USC-2 card: OPTO_OUT0..5 correspond to range **0..5** or **200..205**, DIGI_OUT0..9 of the Extension connector to range **206..215** and SM_OUT0..5 of the Stepper connector to range **216..221**.

dirIO=y

This parameter specifies which output number has to be used to send the direction-signal to the motor that specifies into which direction it has to move. The value given for "y" is a number that defines the port that has to be used and the output pin number the dir-input of the motor is connected with. If a value in the range **1..5** is entered here, the appropriate standard output 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. Please note: the last method of course can be used only if the laser port is NOT used for controlling the laser. Otherwise the results can be undefined and may harm your equipment seriously. If you are using an USC-2 card: OPTO_OUT0..5 correspond to range **0..5** or **200..205**, DIGI_OUT0..9 of the Extension connector to range **206..215** and SM_OUT0..5 of the Stepper connector to range **216..221**.

dirvalue=y

This parameter is related to the preceding one: it defines the level the direction output has to go to for a movement in positive direction. According to that here "y" can be 0 (for ground signal) or 1 (for high-signal). It is recommended to set the value here, so that the axis moves from left to right with positive counting sense. If this is inverted then it is necessary to invert the paramter refmode also. This parameter is described below.

accel=yy

With this parameter an acceleration value in unit increments/s*s can be defined to start movements smoothly. If a value of -1 is set for "yy" this feature is disabled completely and the motor tries to start movements with an nearly infinite acceleration.

decel=yy

With this parameter an deceleration value in unit increments/s*s can be defined to stop movements smoothly. If a value of -1 is set for "yy" this feature is disabled completely and the motor tries to stop movements with an nearly infinite deceleration.

refIO=y

This parameter specifies which input number has to be used to check the reference switch. The value given for "y" is a zero-based number that specifies the input pin the switch is connected with. If no reference switch has to be used that feature can be disabled using the "refmode 0" (please refer below). If the reference input 1 is used together with a reference value of 1 and a reference mode that automatically leaves the reference switch after hitting it the movement to the reference switch is performed as fast as all other movements. If you are using an USC-2 card: OPTO_IN0..5 correspond to range **0..5,** DIGI_IN0..9 of the Extension connector to range **6..15** and SM_IN0..3 of the Stepper connector to range **16..18**.

refvalue=y

The preceding parameter and this one are related to each other: "refvalue" defines what signal at the input with number "refIO" is equal to a actuated reference switch. Here for "y" the values "1" and "0" are possible. If the reference value is set to 1 together with a reference input 1 and a reference mode that automatically leaves the reference switch after hitting it the movement to the reference switch is performed as fast as all other movements.

refspeed=yyy

The reference speed defines which speed the drive has to move with in case of a home-movement to find the reference switch.

refmode=y

Using this parameter the behavior of the drive can be defined when the reference switch is actuated during a movement to the home position. If the dirvalue had been inverted then this value has to be changed properly too. Here for "y" the following values can be set:

0 - there is no homing movement, the value of the "refpos"-parameter is set at the position the drive currently has
1- the drive moves in negative direction to find the reference switch and leaves it with a quarter of the defined "refspeed" in positive direction
2 - go to the reference switch in negative direction and leave it with "refspeed"/4 in negative direction in case a movement to the home position is requested by the software
3 - go to switch in negative direction and stay there
4 - in this mode the drive goes to the reference switch in positive direction and leaves it with "refspeed"/4 in negative direction after it was actuated
5 - go to the reference switch in positive direction and leave it with a quarter of the "refspeed" in positive direction
6 - the drive moves to the switch in positive direction and stays there after it was actuated

Please note: If the fast referencing mode has to be used where the movement to the reference switch is as fast as all other movements (refIO and refvalue set to 1 to enable that mode) here a mode has to be chosen that causes the drive to leave the switch automatically (refmode 1, 2, 4 or 5). Otherwise the drive would stay at this position and would block all other operations - including marking.

refpos=yyyy

After the reference switch was actuated a new value for the current position is set or - if no movement to the home/reference position is defined above - this value is set during the initialization of the application. Here the parameter "refpos" defines the value that has to be set using unit increments.

7.2.3.6 SHS Star 2000 Motion Control

Using the SHS Star 2000 Series Motion Controller

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 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 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 isn't 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 ("Debug=1") the debugging mode is turned on, when it is set to 0 ("Debug=0") debugging is disabled and no data are put into the log file. Please note: 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 isn't 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. Please note: 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 doesn't 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:

```
hval=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 doesn't 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 un-commented. 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.

7.2.3.7 Generic RS232 Control

Using the generic RS232 Controller 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 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 controller is connected to, here for "xxx" e.g. COM1 has to be set. To use the RS232 interface of the USC-1 card, set PortName=USC-1.

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 - don't 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 isn't 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 ("Debug=1") the debugging mode is turned on, when it is set to 0 ("Debug=0") debugging is disabled and no data are put into the log file. Please note: 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.

7.2.3.8 Jenaer Antriebstechnik ECOSTEP 100 Motion Control

Using the ECOSTEP 100 Series Motion Controller

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 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 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 isn't 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 ("Debug=1") the debugging mode is turned on, when it is set to 0 ("Debug=0") debugging is disabled and no data are put into the log file. Please note: 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 isn't 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 EcoStep100 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.

Please note: the checksum and the deviceID don't 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 EcoStep100 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.

Please note: the checksum and the deviceID don't 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.

Please note: the checksum and the deviceID don't 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:
    //
    // Bit7     Bit6     Bit5     Bit4     Bit3     Bit2     Bit1     Bit0
    // Fault Reset
    //          operating specific
    //                   operating specific
    //                            operating specific
    //                                     Enable Operation
    //                                              Quick Stop
    //                                                       Disable Voltage
    //                                                                Switch On
    //
    // if statusword:
    //
    // Bit7     Bit6     Bit5     Bit4     Bit3     Bit2     Bit1     Bit0
    // Warning
    //          Switch on Disable
    //                   Quick Stop
    //                            Voltage Disabled
    //                                     Fault
    //                                              Operation Enable
    //                                                       Switched On
    //                                                                Ready to Switch on

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

```
    //                                          Reserved
    //                                                manufacturer specific

byte    data2;
    // data byte 2

byte    data3;
    // data byte 3

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

string  szPacketName;
    // for debug purposes
```

7.2.3.9 IO Switcher / Indexer Drive

Using the IO Switcher Controller for Drives that require single signals to initiate a movement

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 textfile 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 isn't 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. Please note: 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.

7.2.3.10 Custom Motion Control

Using a Custom Motion Controller

If a motion controller has to be connected that is not supported, there are two possibilities: you can request the implementation of the support of your controller at the vendor of the scanner software or you can implement it by yourself. In the last case there is a interface specification and a reference implementation of a DLL available that describe and demonstrate how such a custom interface can be implemented. That package is available on our webpage or on request.

7.2.4 Trigger Control Objects

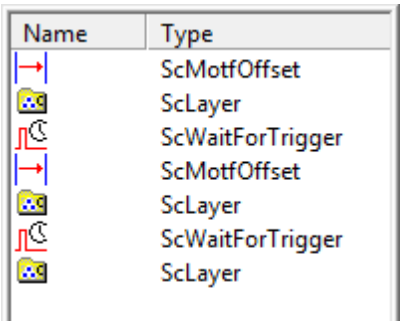
The scanner software offers some extended functionality to handle trigger events, to react on them delayed and to generate artificial triggers signals. This functionality is useful e.g. to set up jobs that are larger than the available working area or to have defined (large) distances between jobs or elements of a job. To set up such an advanced job there are some special elements available within the functional object toolbar.

 An entity "ScMotfOffset" can be used to define a distance after which a new trigger signal has to be generated. The distance can be set within the related entities property page "Control". There the desired value can be entered. The generated trigger signal itself is an internal one, that behaves like there would be an external trigger event. Such a ScMotfOffset defines the distance in a length unit (mm, inch or bits) beginning from the current marking position. The ScMotfOffset entity itself doesn't cause any delay or waiting operation, it works completely asynchronously. To react on such a trigger synchronously the following entity can be used:

 Using a "ScWaitForTrigger" the working flow within a job can be stopped until a trigger signal is received. Here it doesn't make any difference if it is a real trigger signal sent e.g. from an external source or if it is an artificial one sent by a preceding ScMotfOffset object. Both entities together can be used within marking on the fly applications to define exact distances within a job to for an example mark specific parts of that job only after that distance has gone by.

Both elements in most cases should be used together. One main application would be within a

marking on the fly environment where one part of a job has to be marked and a second and third one only after a specific movement distance has gone by. This is important for an example when objects have to be marked that are bigger than the total working area. For such an application a job should look like this:



1. ScMotfOffset that defines the distance after that the first part of the job has to be marked
2. first entity to be marked
3. ScWaitForTrigger to wait that the first distance has gone by
4. ScMotfOffset that defines the distance after that the second part of the job has to be marked
5. second entity to be marked
6. ScWaitForTrigger to wait that the second distance has gone by
7. third entity to be marked

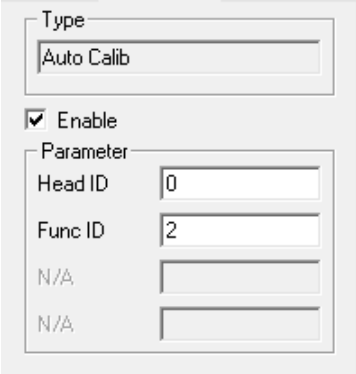
Here the following things are important: The asynchronous ScMotfOffset entity has to be used before a part is marked to not let the marking time of that part influence the distance (and implicitly the time) after that the artificial trigger is sent. It works in this way because ScWaitForTrigger waits for the trigger event of ScMotfOffset and because ScMotfOffset acts in background completely so that the marking process itself is not influenced as long as there is no ScWaitForTrigger object.

The same objects can be used to define an offset after that marking has to be started. Such an offset-definition consists of an ScMotfOffset and an ScWaitForTrigger object only. Here ScMotfOffset again defines the distance and ScWaitForTrigger waits until that distance has gone by before the rest of the job is executed.

7.2.5 AutoCal Control Objects

For RTC4 and RTC5 scanner cards, an AutoCal 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 AutoCal symbol  within the functionality object tool bar has to be activated by checkbox "AutoCal" within sc_setup.exe->"Hardware Settings"->"Settings"->"Driver Settings"->Mode before. To change the properties of an AutoCal 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:



Enable:
This checkbox activates the AutoCal control object.

Parameter

Head ID:
primary scan head: 0, secondary scan head: 1, default = 0

Func ID:
1: turn auto calibration mode on and go to the reference position for an initial calibration
2: recalibrate

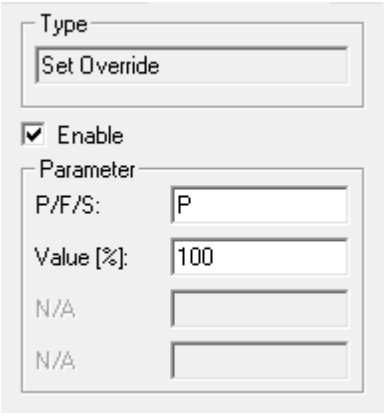
3: turn off auto calibration mode

Attention: 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).

7.2.6 SetOverride Control Objects

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.



Enable:
This checkbox activates the SetOverride control object.

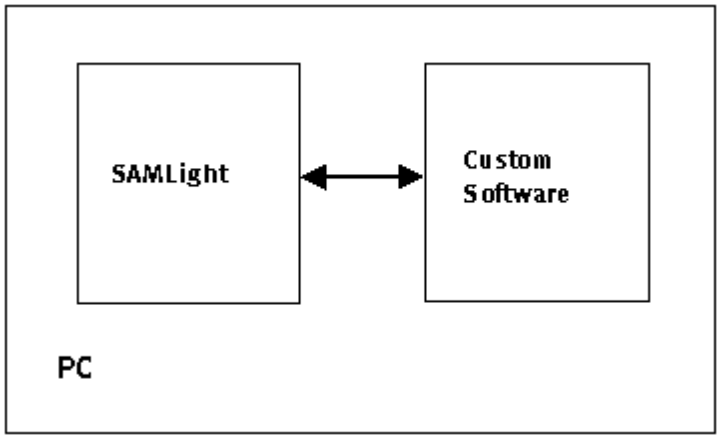
Parameter

P/F/S:
P = Power
F = Frequency
S = Speed

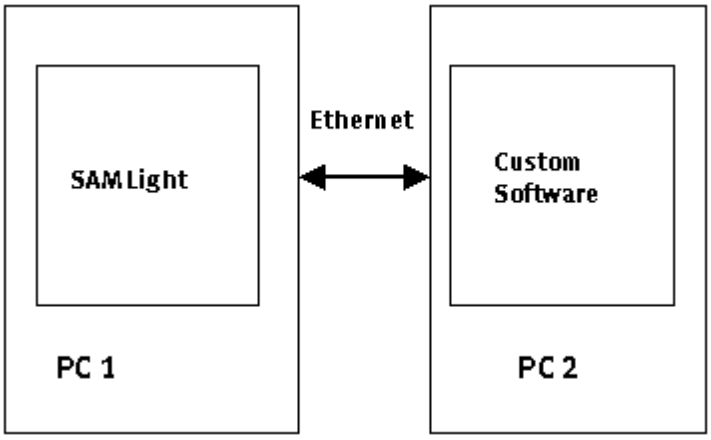
Value[%]:
The override factor is expressed in percent and can be used to increase or decrease the P/S/F value of the pen during the mark process for the marking of all the following entities in the entity list.

7.3 Programming Interface

This manual shows how to control SAMLight out of another software, either inside one PC or in between two PCs:

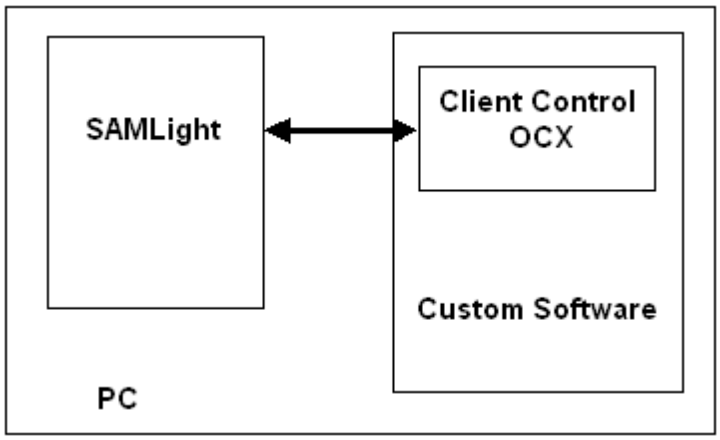


or:

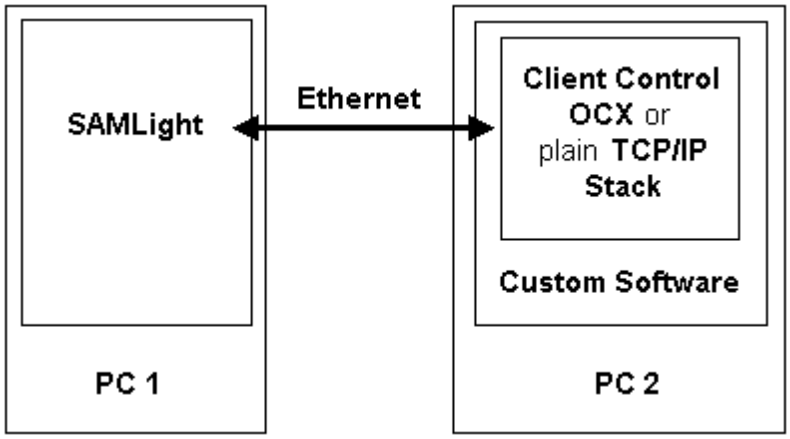


7.3.1 Principle of operation

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.



or:



7.3.2 Remote Settings

The settings for an external control are available in the Menu Settings -› System -› Remote:

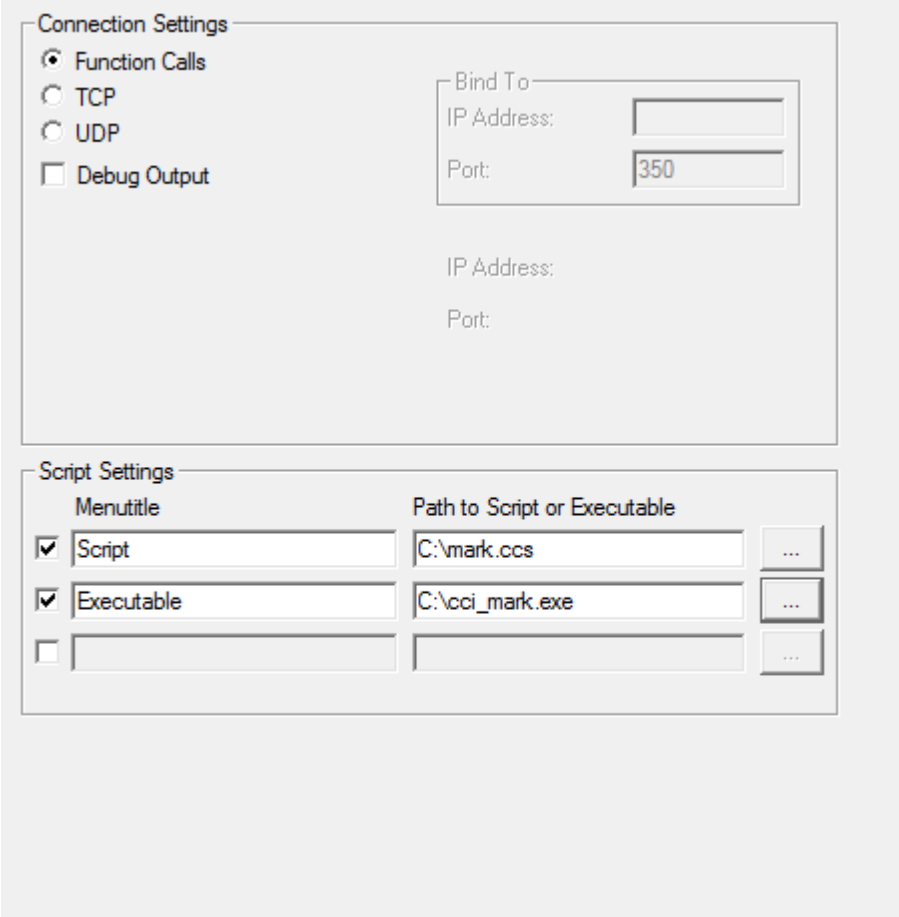


Figure 7.3: Remote Settings Dialog

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 / UDP:

If the program should be controlled over the network using the client control interface, one of these two options has to be selected. In this case some additional configuration has to be done. Normally the TCP protocol should be used. UDP is recommended only if a very fast connection is required. UDP doesn't allow to transfer bigger amounts of data. If more than only some bytes have to be transferred at the same time TCP should be chosen here. If you are using an UDP connection and notice connection problems, please switch to TCP here.

Debug Output:

If this checkbox is active a file sc_cci_debug.txt will be created in the SAMLight system folder. This file will contain all CCI commands that have been executed.

Bind To

IP Adress / Port:
As an IP Address the IP where the controlled softwares server socket has to be bound to has to be entered. 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 (This could be - depending on the setup of your local network - e.g. something like `192.168.1.100`. Please ask your system administrator for further 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 above can be used.

The **Port** number defines where the software has to be accessible. Here any value in the range 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 can't 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 ScOpenUDPConnection() or ScOpenTCPConnection(). If the plain ASCII communication has to be used, opening of a TCP/IP or UDP 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 the folder "scaps_sam\system" of the installation directory of the scanner application. Both files expect IP numbers in the format www.xxx.yyy.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 textfile "hosts.deny" can contain a list of IP numbers with the format www.xxx.yyy.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 adress 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. Please note: 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.

7.3.3 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 doesn't 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 CCI Plain\n"** has to be sent to set up the communication interface for the ASCII communication mode. Without that initialization no ASCII-based communication is possible! 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, the name of the ASCII-command and the appropriate define for 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.

7.3.3.1 Application

Function: **long ScIsRunning();**
ASCII: **-**
Define: **-**
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()**
ASCII: **long ScCciGetInterfaceVersion()\n**
Define: **SC_CCI_CMD_GET_INTERFACE_VERSION**
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**

Define: **SC_CCI_CMD_SHUTDOWN**
Terminates the scanner software. After that no more commands should be given.

Function: **-**
ASCII: **long ScCciTest(string Test)\n**
Define: **SC_CCI_CMD_TEST**
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**
Define: **SC_CCI_CMD_SHOW_APP**
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.

Show state specified by parameter Show	Value of the Show parameter
SW_HIDE - Hides the window and passes activation to another window.	0
SW_SHOWMINIMIZED - Activates the window and displays it as an icon.	2
SW_SHOWMAXIMIZED - Activates the window and displays it as a maximized window.	3
SW_SHOW - Activates the window and displays it in its current size and position.	5
SW_RESTORE - Activates and displays the window. If the window is minimized or maximized, Windows restores it to its original size and position.	9

7.3.3.2 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: **-**
Define: **-**
This function is deprecated. Please use ScOpenTCPConnection() or ScOpenUDPConnection() instead of it.

Function: **long ScOpenUDPConnection(BSTR SenderAddr, long SenderPort,BSTR RecipientAddr, long RecipientPort)**
ASCII: **-**
Define: **-**
Opens an UDP connection to the scanner application through the network and returns 1 if the connection could be established successfully. SenderAddr is the IP of the sending application for the controlled software (this server socket is used for answers from the controlled software back to the controlling one), SenderPort specifies the port number that has to be used. 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.

Function: **long ScOpenTCPConnection(BSTR RecipientAddr, long RecipientPort)**
ASCII: **-**
Define: **-**
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: **-**
Define: **-**
Closes a previously opened connection.

7.3.3.3 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**
Define: **SC_CCI_CMD_GET_WORKING_AREA**
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     0x0
#define SC_SAMLIGHT_OUTLINE_INDEX_MIN_Y     0x1
#define SC_SAMLIGHT_OUTLINE_INDEX_MIN_Z     0x2
#define SC_SAMLIGHT_OUTLINE_INDEX_MAX_X     0x3
#define SC_SAMLIGHT_OUTLINE_INDEX_MAX_Y     0x4
#define SC_SAMLIGHT_OUTLINE_INDEX_MAX_Z     0x5
```

Function: **long ScOpticMatrixTranslate(double X, double Y, double Z)**
ASCII: **long ScCciOpticMatrixTranslate(double X, double Y, double Z)\n**
Define: **SC_CCI_CMD_OPTIC_MATRIX_TRANSLATE**
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**
Define: **SC_CCI_CMD_OPTIC_MATRIX_ROTATE**
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**
Define: **SC_CCI_CMD_OPTIC_MATRIX_SCALE**
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**
Define: **SC_CCI_CMD_OPTIC_MATRIX_RESET**
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**
Define: **SC_CCI_CMD_GET_OPTIC_MATRIX**
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:

Xnew = Mo*Xold + M1*Yold + M2
Ynew = M3*Xold + M4*Yold + M5

Function: **long ScSetHead (long HeadID)**
ASCII: **long ScCciSetHead(long HeadID)\n**
Define: **SC_CCI_CMD_SET_HEAD**
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**
Define: **SC_CCI_CMD_GET_HEAD**
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.

7.3.3.4 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**
Define: **SC_CCI_CMD_IS_MARKING**
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**
Define: **SC_CCI_CMD_STOP_MARKING**
Stops the current marking process.

Function: **long ScSetMarkFlags(long Flags)**
ASCII: **long ScCciSetMarkFlags(long Flags)\n**
Define: **SC_CCI_CMD_SET_MARK_FLAGS**

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 = 0x0001
The Mark function starts only after an external start. The WaitForMarkEnd parameter value of ScMarkEntityByName() flag has no effect.

scComSAMLightClientCtrlMarkFlagHideOutput = 0x0004
Does not show the output window of the controlled scanner application during marking.

scComSAMLightClientCtrlMarkFlagDisableHomeJump = 0x0008
Switch off the home jump after the mark.

scComSAMLightClientCtrlMarkFlagPreview = 0x0010
Execute marking in preview mode. For an example see also: How to precalculate time.

scComSAMLightClientCtrlMarkFlagSelected = 0x0020
Mark only selected entities and ignore all other ones.

scComSAMLightClientCtrlMarkFlagDisablePreProcessing = 0x0040
Disable the pre-processing phase.

scComSAMLightClientCtrlMarkFlagDisablePostProcessing = 0x0080
Disable the post-processing phase.

scComSAMLightClientCtrlMarkFlagEntityNamesSeparatedBySemicolon = 0x0200
This flag is no longer supported by the SAMLight Client Control Type Library from build 0613 on.
See scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon for more detail.

Function: **long ScGetMarkFlags()**
ASCII: **long ScCciGetMarkFlags(long Flags)\n**
Define: **SC_CCI_CMD_GET_MARK_FLAGS**
Returns the currently active mark flags as described above.

Function: **long ScSwitchLaser(long on_off)**
ASCII: **long ScCciSwitchLaser(long on_off)\n**
Define: **SC_CCI_CMD_SWITCH_LASER**
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**
Define: **SC_CCI_CMD_MOVE_ABS**
Directly jumps to the new position specified by x, y and z. Jumpspeed and delays are set according to the pen defined through a previous call to ScSetPen().

Function: **long ScSetPen(long pen)**
ASCII: **long ScCciSetPen(long pen)\n**
Define: **SC_CCI_CMD_SET_PEN**

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**
Define: **SC_CCI_CMD_GET_PEN**
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**
Define: **SC_CCI_CMD_RESET_SEQUENCE**
The current sequence is reset.

Function: **-**
ASCII: **long ScCciResetCounter()\n**
Define: **SC_CCI_CMD_RESET_COUNTER**
The quantity counter will be reset.

Function: **-**
ASCII: **long ScCciResetSerialNumbers()\n**
Define: **SC_CCI_CMD_RESET_SERIAL_NUMBERS**
This call resets all serial number entities to their start values.

Function: **-**
ASCII: **long ScCciIncSerialNumbers()\n**
Define: **SC_CCI_CMD_INC_SERIAL_NUMBERS**
This call increments all serial number entities.

Function: **-**
ASCII: **long ScCciDecSerialNumbers()\n**
Define: **SC_CCI_CMD_DEC_SERIAL_NUMBERS**
This call decrement all serial number entities.

Function: **-**
ASCII: **long ScCciResplitJob()\n**
Define: **SC_CCI_CMD_RESPLIT_JOB**
Resplit a job that is in splitting mode and was modified so that the splitted 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**
Define: **SC_CCI_CMD_SET_PIXEL_MAP_FOR_PEN**
Set the pen specific gray scale bitmap pixel map (6 zones, zone values can be 0 .. 100 (percent) - 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.

7.3.3.5 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 has to 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**
Define: **SC_CCI_CMD_MARK_ENTITY_BY_NAME**
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@@@".

Function: **long ScChangeTextByName(BSTR EntityName, BSTR Text)**
ASCII: **long ScCciChangeTextByName(string EntityName, string Text)\n**
Define: **SC_CCI_CMD_CHANGE_TEXT_BY_NAME**
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 cahnge the text of more than one entities with one function call. Therefore set the Flag scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon. Then separate the different EntityNames with a ';'. If you want to give the different Entiie each 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**
Define: **SC_CCI_CMD_GET_ENTITY_OUTLINE**
Returns the outline of the entity with the name EntityName. If EntityName is an empty string, the function returns the job outline. The Index can be one of the values listed above. Please note that the Z values only give correct values if the optional Optic3D/Lines3D 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
Return the minimum position in X direction

scComSAMLightClientCtrlOutlineIndexMinY = 1
Return the minimum position in Y direction

scComSAMLightClientCtrlOutlineIndexMinZ = 2
Return the minimum position in Z direction

scComSAMLightClientCtrlOutlineIndexMaxX = 3

Return the maximum position in X direction

scComSAMLightClientCtrlOutlineIndexMaxY = 4
Return the maximum position in Y direction

scComSAMLightClientCtrlOutlineIndexMaxZ = 5
Return the maximum position in Z direction

Function: **long ScTranslateEntity(BSTR EntityName, double X, double Y, double Z)**
ASCII: **long ScCciTranslateEntity(string EntityName, double X, double Y, double Z)\n**
Define: **SC_CCI_CMD_TRANSLATE_ENTITY**
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**
Define: **SC_CCI_CMD_SCALE_ENTITY**
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 ScRotateEntity(BSTR EntityName, double X, double Y, double Angle)**
ASCII: **long ScCciRotateEntity(string EntityName, double X, double Y, double Angle)\n**
Define: **SC_CCI_CMD_ROTATE_ENTITY**
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 contra clockwise for positive values. If EntityName is an empty string the complete job is rotated.

Function: **long ScSetEntityLongData(BSTR EntityName, long DataId, long Data)**
ASCII: **long ScCciSetEntityLongData(string EntityName, long DataId, long Data)\n**
Define: **SC_CCI_CMD_SET_ENTITY_LONG_DATA**
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**
Define: **SC_CCI_CMD_GET_ENTITY_LONG_DATA**
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**
Define: **SC_CCI_CMD_SET_ENTITY_DOUBLE_DATA**
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**
Define: **SC_CCI_CMD_GET_ENTITY_DOUBLE_DATA**
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**
Define: **SC_CCI_CMD_SET_ENTITY_STRING_DATA**
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**
Define: **SC_CCI_CMD_GET_ENTITY_STRING_DATA**
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**
Define: **SC_CCI_CMD_DELETE_ENTITY**
Deletes the entity with the name EntityName.

7.3.3.6 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**
Define: **SC_CCI_CMD_LOAD_JOB**
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 t o 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**
Define: **SC_CCI_CMD_IMPORT**
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 = HEX 8
Keep the order of the entities.

scComSAMLightClientCtrlImportFlagReadPenInfo = HEX 80
Read pen information.

scComSAMLightClientCtrlImportFlagPointCloud = HEX 4000

Import of points into PointClouds.

scComSAMLightClientCtrlImportFlagToPenGroups = HEX 10000
Put all entities with the same pen into a separate group.

scComSAMLightClientCtrlImportFlagNotTryToClose = HEX 20000
Do not try to close polygons. This option applies to HPGL type files only.

scComSAMLightClientCtrlImportFlagBitmapReimport = HEX 100000
Reimports a Bitmap with EntityName.

scComSAMLightClientCtrlImportFlagVectorReimport = HEX 800000
Reimports Vector Graphics like DXF and PLT with EntityName.

scComSAMLightClientCtrlImportFlagToplevelOnly = HEX 1000000
Reimports only entities in the first level of the job.

Function: **long ScSaveJob(BSTR FileName, long Flags)**
ASCII: **long ScCciSaveJob(string FileName, long Flags)\n**
Define: **SC_CCI_CMD_SAVE_JOB**
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**
Define: **SC_CCI_CMD_NEW_JOB**
The current job is deleted completely.

Function: **-**
ASCII: **long ScCciFitViewToWorkingArea()\n**
Define: **SC_CCI_CMD_FIT_VIEW_TO_WORKING_AREA**
Change the view within the scanner application to view the full working area.

Function: **-**
ASCII: **long ScCciFitViewToEntities()\n**
Define: **SC_CCI_CMD_FIT_VIEW_TO_ENTITIES**
Change the view within the scanner application to view all available entities.

Function: **-**
ASCII: **long ScCciFitViewToSelectedEntities()\n**
Define: **SC_CCI_CMD_FIT_VIEW_TO_SELECTED_ENTITIES**
Change the view within the scanner application to view all selected entities.

Function: **long ScProcessFlashJob (string Name, long JobNum, long Mode, long Flags)**
ASCII: -
Define: -
Copies a specified job from the SAMLight editor to the Flash or the other direction. If Mode is set to 1 the job is loaded to the Flash, if Mode is set to 2 the job is saved from Flash to the SAMLight Editor. Flags can only be 0.

Function: **string ScFlashCommand (string Command, long Flags, string Return)**
ASCII: -
Define: -
Execute a Flash Command. Flags can be 0 or 0x80000000. If 0, the Client Control waits until a Carriage Return is returned from the Flash. If 0x80000000 is set the Client Control will wait a certain time and then end the command.

7.3.3.7 General Commands

Function: **long ScExecCommand(long CmdID)**
ASCII: **-**
Define: **-**
A specific command is executed. The parameter CmdID specifies what kind of command is executed, see Long CmdIDs.

Function: -
ASCII: **ScCciMotionGo**
Define: **-**
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**
Define: **SC_CCI_CMD_SET_DOUBLE_VALUE**
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**
Define: **SC_CCI_CMD_GET_DOUBLE_VALUE**
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**
Define: **SC_CCI_CMD_SET_LONG_VALUE**
Sets different long values used for some functions. See the definition of the available types in Long Value Types.

Set OPTO_OUT:

Type (Decimal): 4
Value: Sets the Opto Out Port

Example:

ScSetLongValue(4,2); //Sets OPTO_OUT 1 to "1" and OPTO_OUT 2... OPTO_OUT 5 to "0"
Please note that OPTO_OUT 0 is not influenced by this command since it is reserved for the mark in progress flag.

Function: **long ScGetLongValue(long Type)**
ASCII: **long ScCciGetLongValue(long Type)\n**
Define: **SC_CCI_CMD_GET_LONG_VALUE**
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.

Get OPTO_IN:

Type (Decimal): 4
Value: Returns the Opto in Port

Example:

long OptoInput;
OptoInput = ScGetLongValue(4);

Function: **long ScSetStringValue(long Type,BSTR Value)**
ASCII: **long ScCciSetStringValue(long Type,string Value)\n**
Define: **SC_CCI_CMD_SET_STRING_VALUE**
Set different string values used for some functions. See the definition of the available types in String Value Types.

USC-1 card:

Type (Decimal): 1
Value: Sets the Baud rate of the RS232 output

The RS232 interface supports the following Baud rates:
2400 , 4800 , 9600, 19200, 28800, 38400 , 57600.

Example:

ScSetStringValue(1,"9600"); //Sets the baud rate to 9600.

Type (Decimal): 2
Value: Sets the RS232 Output string

Example:

ScSetStringValue(2,"Hallo"); //Writes the string "Hallo" to the RS232 port.

Type (Decimal): 3
Value: Sets the RS232 Mode

The following modes are available:

Mode	DataBits	StopBit	Parity
0	8	1	Not used
1	8	1	Odd
2	8	1	Even
3	8	1	1
4	8	1	0

Example:
ScSetStringValue(3,"0") ; //Sets the RS232 port to mode 0

Function: **long ScGetStringValue(long Type, BSTR* Value)**
ASCII: **string ScCciGetStringValue(long Type)\n**
Define: **SC_CCI_CMD_GET_STRING_VALUE**
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 "".

USC-1 card:

Type=1 and Type=3 return the current settings of baud rate and mode. Please refer to long ScSetStringValue(long Type, BSTR Value);

Type (Decimal): 2
Value: Returns the RS232 input string

Example:
Dim RS232Return as String;
ScGetStringValue(2, RS232Return);
MsgBox RS232Return;

Function: **long ScSetStringLongValue(long Type,BSTR SValue,long LValue)**
ASCII: **long ScCciSetStringLongValue(long Type,string SValue,long LValue)\n**
Define: **SC_CCI_CMD_SET_STRING_LONG_VALUE**
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**
Define: **SC_CCI_CMD_SET_STRING_DOUBLE_VALUE**
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**
Define: **SC_CCI_CMD_GET_STRING_DOUBLE_VALUE**

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**
Define: **SC_CCI_CMD_GET_ID_STRING_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 appropriate scComSAMLightClientCtrlStringDataId-constants has to be set.

Function: **long ScSetIDStringData(long Type,long DataId, BSTR *Data)**
ASCII: **string ScCciSetIDStringData(long Type,long DataId)\n**
Define: **SC_CCI_CMD_SET_ID_STRING_DATA**
With this function strings can be assigned to entities using an additional DataId that may be e.g. an index number or an identifier. For Type one of the appropriate scComSAMLightClientCtrlStringDataId-constants has to be set.

Function: **-**
ASCII: **long ScCciAutoCompensateOff()\n**
Define: **SC_CCI_CMD_AUTO_COMPENSATE_OFF**
Scanner auto-calibration functionality (works only with hardware that supports it): turn auto calibration mode off.

Function: **-**
ASCII: **long ScCciAutoCompensateRef()\n**
Define: **SC_CCI_CMD_AUTO_COMPENSATE_REF**
Scanner auto-calibration functionality (works only with hardware that supports it): turn auto calibration mode on.

Function: **-**
ASCII: **long ScCciAutoCompensateCal()\n**
Define: **SC_CCI_CMD_AUTO_COMPENSATE_CAL**
Scanner auto-calibration functionality (works only with hardware that supports it): recalibrate the scanhead.

Function: **long ScSetMode(long Flags)**
ASCII: **long ScCciSetMode(long Flags)\n**
Define: **SC_CCI_CMD_SET_MODE**
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 = 0x1
The next calls only search for objects in the top level of the job. This can be helpful to increase the performance.

scComSAMLightClientCtrlModeFlagDontUpdateView = 0x2
Suppress drawing of entities in the view or in the entity list. This can be helpful to increase the performance.

scComSAMLightClientCtrlModeFlagEntityNamesSeparatedBySemicolon = 0x20
The parameter EntityName in ScMarkEntityByName() can include more than one entity name. They have to be separated by a semicolon. If the Client Control is controlled 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**
Define: **SC_CCI_CMD_GET_MODE**
Returns the current active mode flag as described above.

Function: **long ScFlashCommand(LPCTSTR Command, 0, BSTR *Return);**
ASCII: -
Define: -
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.

Function: **long ScProcessFlashJob("",long JobNum, 1, 1)**
ASCII: -
Define: -
Loads the currently opened job in SAMLight to the Flash using the Job number given in JobNum.

7.3.3.8 Async Mode

It is possible to set the ClientCtrl globally in an Async Mode. See the following C# example:

```
axScSamlightClientCtrl1.ScSetLongValue((int)
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValue
TypeClientCtrlAsyncMode, 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/Windows 7 or former command already running

while(axScSamlightClientCtrl1.ScGetLongValue((int)
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValue
TypeClientCtrlAsyncModeIsRunning ) != 0 )
{
      Application.DoEvents();
}

res=axScSamlightClientCtrl1.ScGetLongValue((int)
ScComSAMLightClientCtrlValueTypes.scComSAMLightClientCtrlLongValue
TypeClientCtrlAsyncModeResult );
Debug.Assert( res == 1 );

MessageBox.Show( "Ready" );
```

This is by now 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 (scComSAMLightClientCtrl**Double**ValueTypeClientCtrlAsyncModeResult) in order to get the return value. A value of HUGE_VAL (+Infinity) indicates an error.

If you want to use a ClientCtrl command normally again you have to call the following function before:

```
axScSamlightClientCtrl1.ScSetLongValue((int)
ScComSAMLightClientCtrlValueTypes
scComSAMLightClientCtrlLongValueTypeClientCtrlAsyncMode, 0 );
```

7.3.4 Constants

Long Value Types
These types can be used for the Type-parameter of the functions ScSetLongValue() and ScGetLongValue().

scComSAMLightClientCtrlLongValueTypeNumMarksCompleted = HEX 01
Returns the number of marks. Helpful in Mark Trigger mode.

scComSAMLightClientCtrlLongValueTypeOptoIO = HEX 04
This value can be used to set the Opto-IOs or to get their current input state. If used together with ScSetLongValue() this will set the OptoOut bits. If used together with ScGetLongValue() the OptoIn bits are read.

scComSAMLightClientCtrlLongValueTypeDeviceEnableFlagsSet = HEX 08
The flag is related to the currently used pen. By default the FlagsSet is 0, following values can be set enabled:

scComStandardDeviceEnableFlagGroupStyle = 0
scComStandardDeviceEnableFlagGroupOptoOut = 1

Note: Only with scComSAMLightClientCtrlLongValueTypeDeviceEnableFlagsValue a value is set.

scComSAMLightClientCtrlLongValueTypeDeviceEnableFlagsValue = HEX 09
The flag is related to the currently used pen. It is used to specify a constant of type scComStandardDeviceEnableFlagConstants that has to be set or get for the previously defined FlagsSet. Note: The flag has to be enabled with scComSAMLightClientCtrlLongValueTypeDeviceEnableFlagsSet before.

Enabled with scComStandardDeviceEnableFlagGroupStyle:

scComStandardDeviceStyleFlagEnableLongDelay = HEX 1
Enables Time delay between pen alternation

scComStandardDeviceStyleFlagEnablePortLaser = HEX 2
Enables the currently selected Laser Port

scComStandardDeviceStyleFlagEnableWobble = HEX 8
Enables Wobble

scComStandardDeviceStyleFlagEnableYAGStandBy = HEX 20
Enables YAG StandBy

scComStandardDeviceStyleFlagEnablePixelOutput = HEX 40
Sets Pixel Hardware Mode

scComStandardDeviceStyleFlagEnablePort1 = HEX 4
scComStandardDeviceStyleFlagEnablePort2 = HEX 80
scComStandardDeviceStyleFlagEnableDA1 = HEX 200
scComStandardDeviceStyleFlagEnableDA2 = HEX 400

scComStandardDeviceStyleFlagEnableSkyWriting = HEX 2000
Enables SkyWriting

Enabled with scComStandardDeviceEnableFlagGroupOptoOut:
Sets Opto Output Bits

scComSAMLightClientCtrlLongValueTypeTotalEntityNum = HEX 0A
Using this type parameter it is possible to count the total number of available entities including the ones that are nested in groups.

scComSAMLightClientCtrlLongValueTypeToplevelEntityNum = HEX 0B
This type parameter can be used together with ScGetLongValue() to evaluate the top level entities. That means using this value that number of entities is returned which are visible on the highest level, excluding the contents of groups.

scComSAMLightClientCtrlLongValueTypeJobExecutionDelay = HEX 0C
This type can be used to get/set the global parameter "job execution delay", adjustable within the settings IO page. Unit is [ms].

scComSAMLightClientCtrlLongValueTypeBufferQueueLength = HEX 0E
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 from it. Other values than 0 are not valid for the set-operation. For more information about the queue functionality please refer to the Examples section.

scComSAMLightClientCtrlLongValueTypeQuantity = HEX 05

Gets or sets the current quantity if the quantity counter is enabled.

scComSAMLightClientCtrlLongValueTypeMaxQuantity = HEX 0F
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 = HEX 10
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 = HEX 11
Axis value: 0 to 4, or -1 for all axis.

scComSAMLightClientCtrlLongValueTypeMotionWaitForEnd = HEX 12
Value: 0 or 1. 1 defines that the application waits until the motion has finished. With 0 the application comes back immidiately. The default value is 1.

scComSAMLightClientCtrlLongValueTypeMotionMoving = HEX 13
Returns the state of motion defined with scComSAMLightClientCtrlLongValueTypeMotionAxis.

scComSAMLightClientCtrlLongValueTypeLineRampingPowerStartRampActive = HEX 16
scComSAMLightClientCtrlLongValueTypeLineRampingPowerEndRampActive = HEX 17
scComSAMLightClientCtrlLongValueTypeLineRampingSpeedStartRampActive = HEX 18
scComSAMLightClientCtrlLongValueTypeLineRampingSpeedEndRampActive = HEX 19
These constants return and set the ramping flags of the current selected pen. See also ScSetPen() and ScGetPen().

scComSAMLightClientCtrlLongValueTypeGetOptoOut = HEX 1F
This constant can be used to read the current state of the OptoOut Bits. For USC cards the lowest bit can be used as an indicator for marking in progress.

scComSAMLightClientCtrlLongValueTypeAngularSplittingParts = HEX 21
Especially for Rotary Splitting define here how often the object should be marked distributed homogeneously on the surface.

scComSAMLightClientCtrlLongValueTypeLastAutoCompensateResult = HEX 26
Returns the error code of the scanner auto-calibration functionality.

scComSAMLightClientCtrlLongValueTypeDongleUserNumber = HEX 27
Gets the user number of the dongle.

scComSAMLightClientCtrlLongValueTypeDongleSystemNumber = HEX 28
Gets the system number of the dongle.

scComSAMLightClientCtrlLongValueTypeSizePixelMap = HEX 2A
Returns the size of the Pixel Map, means the number of the available grey value partitions.

scComSAMLightClientCtrlLongValueTypeSwitchToPane = HEX 2B
If the additional parameter is 0 then the Mark Preview Window is shown. If the parameter is 1 the Main Window will be displayed.

scComSAMLightClientCtrlLongValueTypeSelectRedpointerForMoveAbs = HEX 39
Activates the redpointer for successive ScMoveAbs commands.

Double Value Types
These types can be used for the Type-parameter of the functions ScSetDoubleValue() and ScGetDoubleValue().

scComSAMLightClientCtrlDoubleValueTypeOverrideSpeed = 1
scComSAMLightClientCtrlDoubleValueTypeOverridePower = 2
scComSAMLightClientCtrlDoubleValueTypeOverrideFrequ = 3
scComSAMLightClientCtrlDoubleValueTypeOverridePower2 = 22
These constants can be used to define override-values for marking speed, power and frequency.

scComSAMLightClientCtrlDoubleValueTypeMarkSpeed = 4
scComSAMLightClientCtrlDoubleValueTypeJumpSpeed = 5
scComSAMLightClientCtrlDoubleValueTypeFrequency = 6
scComSAMLightClientCtrlDoubleValueTypeJumpDelay = 7
scComSAMLightClientCtrlDoubleValueTypeMarkDelay = 8
scComSAMLightClientCtrlDoubleValueTypePolyDelay = 9
scComSAMLightClientCtrlDoubleValueTypeLaserOnDelay = 10
scComSAMLightClientCtrlDoubleValueTypeLaserOffDelay = 11
scComSAMLightClientCtrlDoubleValueTypeScannerXPos = 12
scComSAMLightClientCtrlDoubleValueTypeScannerYPos = 13
scComSAMLightClientCtrlDoubleValueTypeScannerZPos = 14
scComSAMLightClientCtrlDoubleValueTypePulseLength = 15
scComSAMLightClientCtrlDoubleValueTypeFirstPulseLength = 16
scComSAMLightClientCtrlDoubleValueTypeLaserPower = 17
scComSAMLightClientCtrlDoubleValueTypeSizePowerMap = 18
scComSAMLightClientCtrlDoubleValueTypeMaxPower = 20
scComSAMLightClientCtrlDoubleValueTypeLineRampingPowerStartRampValue = 26
scComSAMLightClientCtrlDoubleValueTypeLineRampingPowerStartRampLength = 27
scComSAMLightClientCtrlDoubleValueTypeLineRampingPowerEndRampValue = 28
scComSAMLightClientCtrlDoubleValueTypeLineRampingPowerEndRampLength = 29
scComSAMLightClientCtrlDoubleValueTypeLineRampingSpeedStartRampValue = 30
scComSAMLightClientCtrlDoubleValueTypeLineRampingSpeedStartRampLength = 31
scComSAMLightClientCtrlDoubleValueTypeLineRampingSpeedEndRampValue = 32
scComSAMLightClientCtrlDoubleValueTypeLineRampingSpeedEndRampLength = 33
scComSAMLightClientCtrlDoubleValueTypeLineRampingLengthenStart = 60
scComSAMLightClientCtrlDoubleValueTypeLineRampingLengthenEnd = 61
scComSAMLightClientCtrlDoubleValueTypeSkyWritingStartLength = 35
scComSAMLightClientCtrlDoubleValueTypeSkyWritingEndLength = 36
scComSAMLightClientCtrlDoubleValueTypeSkyWritingBreakAngle = 37
scComSAMLightClientCtrlDoubleValueTypeDefocus = 62
scComSAMLightClientCtrlDoubleValueTypeHalfPeriod = 43
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. This can be used in combination with a previous mark operation in preview mode. See sample.

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 seconds.

scComSAMLightClientCtrlDoubleValueTypeHomePosX = 23
scComSAMLightClientCtrlDoubleValueTypeHomePosY = 24
scComSAMLightClientCtrlDoubleValueTypeHomePosZ = 25
Get and set the home position.

scComSAMLightClientCtrlDoubleValueTypeMotionAxisPosition = 38
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 and get.

scComSAMLightClientCtrlDoubleValueTypeAngularSplittingTotalDiameter = 50
scComSAMLightClientCtrlDoubleValueTypeAngularSplittingAngle = 51
Using these constants the Diameter and Angle for the Splitting Mode can be set and get.

scComSAMLightClientCtrlDoubleValueTypeHorizontalSplittingValue = 52
scComSAMLightClientCtrlDoubleValueTypeVerticalSplittingValue = 53
Using these constants the Horizontal and Vertical split size for the Splitting Mode can be set and get.

scComSAMLightClientCtrlDoubleValueTypeWobbleFrequency = 54
scComSAMLightClientCtrlDoubleValueTypeWobbleAmplitude = 55
Using these constants the Frequency and Amplitude for the Wobble Mode can be set and get.

scComSAMLightClientCtrlDoubleValueTypeStartSplittingPosX = 56
scComSAMLightClientCtrlDoubleValueTypeStartSplittingPosY = 57
Using these constants a start value for the first and second axis in splitting mode can be set and get.

scComSAMLightClientCtrlDoubleValueTypeOffsetX = 58
scComSAMLightClientCtrlDoubleValueTypeOffsetY = 59
Using these constants the offset X and offset Y correction factor of the optic can be set and get.

scComSAMLightClientCtrlDoubleValueTypeMOFExtStartDelay = 65
Using these constants the MOTF Trigger Delay for RTC cards can be set and get.

scComSAMLightClientCtrlDoubleValueTypeDoublePara1 = 66

This constant is used to define the Dist parameter for the beam comped copy function.

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 to receive a double value into an unused and hidden entity inside the job. This entity will be generated automatically unless it is already existing. The dezimal values from 20001 to 20014 can be used with the same functionality.

scComStandardDeviceMiscPrimaryHeadCorOffsetX = 65546
scComStandardDeviceMiscPrimaryHeadCorOffseY = 65547
Using these constants the offset X and offset Y correction factor of the primary head can be set and get.

scComSAMLightClientCtrlDoubleValueTypePrimaryHeadGainX = 65548
scComSAMLightClientCtrlDoubleValueTypePrimaryHeadGainY = 65549
Using these constants the gain X and gain Y correction factor of the primary head can be set and get.

scComSAMLightClientCtrlDoubleValueTypePrimaryHeadRotate = 65550
Using this constant the rotational angle of the primary head can be set and get.

scComSAMLightClientCtrlDoubleValueTypePrimaryHeadEnable = 65551
Using this constant the primary head can be activated or deactivated.

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadOffsetX = 65552
scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadOffsetY = 65553
Using these constants the offset X and offset Y correction factor of the secondary head can be set and get.

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadGainX = 65556
scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadGainY = 65557
Using these constants the gain X and gain Y correction factor of the secondary head can be set and get.

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadRotate = 65558
Using this constant the rotational angle of the secondary head can be set and get.

scComSAMLightClientCtrlDoubleValueTypeSecondaryHeadEnable = 65559
Using this constant the secondary head can be activated or deactivated.

scComSAMLightClientCtrlDoubleValueTypeWorkingAreaMinX =68
scComSAMLightClientCtrlDoubleValueTypeWorkingAreaMinY =69
scComSAMLightClientCtrlDoubleValueTypeWorkingAreaMaxX =70
scComSAMLightClientCtrlDoubleValueTypeWorkingAreaMaxY =71
Using these constants the working area can be set and get.

String Value Types

The following types can be used for the Type-parameter of the functions ScSetStringValue() and ScGetStringValue().

scComSAMLightClientCtrlStringValueTypeJobFileName = 4
Returns the name of the current job file with its complete path.

scComSAMLightClientCtrlStringValueTypeSaveView2D160 = 6
scComSAMLightClientCtrlStringValueTypeSaveView2D320 = 7
scComSAMLightClientCtrlStringValueTypeSaveView2DFull = 9
These types can be used to create a screenshot from the main view and all contained entities. The filename where to save the captured BMP-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 or the full, not scaled size. To let this function work successfully it has to be made sure that the main window of the controlled scanner application is visible and not hidden by something else. So the window can't 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 valid only for the functions ScSetStringValue(), ScSetStringLongValue(), ScSetStringDblValue() and ScGetStringDblValue().

scComSAMLightClientCtrlStringValueTypeMotionString = 11
Defines a string for "send as RS232 string" to the port defined within 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() and Value Type scComSAMLightClientCtrlDoubleValueTypeSaveView2DBitmapDPI.

scComSAMLightClientCtrlStringValueTypeCorrectionFile = 13
Using this constant a new correction file name can be set or the current one can be fetched.

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.

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.

scComSAMLightClientCtrlStringValueTypeUserValue = 20000
This value can be used to store or to receive a string value into an unused and hidden entity inside the job. This entity will be generated automatically unless it is already existing. The

dezimal values from 20001 to 20009 can be used with the same functionality.

Long Data IDs

The following is a list of valid DataIds for the functions ScSetEntityLongData() and ScGetEntityLongData().

scComSAMLightClientCtrlLongDataIdFlagDontUpdateView = HEX 10000
If this bit is set, the view will not be refreshed. This can be helpful for increasing performance.

scComSAMLightClientCtrlLongDataIdFlagDontUpdateEntity = HEX 20000
If this bit is set, the entity will not be regenerated and updated. This can be helpful for increasing performance.

scComSAMLightClientCtrlStringDataIdFlagEnqueueCtrlCmd = HEX 80000
scComSAMLightClientCtrlStringDataIdFlagEnqueueLastCtrlCmd = HEX 100000
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. If this flag is set the return value of the appropriate set-function specifies if the command could be added to the queue successfully or if the queue is full. For the last case the function has to be called again using the same parameters after some time. For more information about the queue functionality please refer to the Examples section.

scComSAMLightClientCtrlLongDataIdFlagToplevelOnly = HEX 200000
If this bit is set, it will only be searched for entities in first level of the job. This can be helpful for increasing performance.

scComSAMLightClientCtrlLongDataIdUserData = 1
The parameter Data then contains a long value with the data to store inside the entity.

scComSAMLightClientCtrlLongDataIdTextAlignment = 2
The parameter Data then contains a long value with the alignment flag fields with the following possible values:

scComSAMLightClientCtrlTextAlignmentCenter = HEX 1
scComSAMLightClientCtrlTextAlignmentLeft = HEX 2
scComSAMLightClientCtrlTextAlignmentRight = HEX 4
scComSAMLightClientCtrlTextAlignmentTop = HEX 8
scComSAMLightClientCtrlTextAlignmentBottom = HEX 10
scComSAMLightClientCtrlTextAlignmentMiddle = HEX 20
scComSAMLightClientCtrlTextAlignmentRadialCenter = HEX 40
scComSAMLightClientCtrlTextAlignmentRadialEnd = HEX 80
scComSAMLightClientCtrlTextAlignmentLineLeft = HEX 100
scComSAMLightClientCtrlTextAlignmentLineRight = HEX 200

scComSAMLightClientCtrlTextAlignmentLineCenter = HEX 400

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. The parameter Data then contains a flag field with a combination out of the following values:

scComSAMLightClientCtrlEntityArrayOrderFlagMainDirX = HEX 400
Use X as the main direction for array copying.

scComSAMLightClientCtrlEntityArrayOrderFlagNegX = HEX 100
Go from left to right in horizontal direction if this flag is set, else go from right to left.

scComSAMLightClientCtrlEntityArrayOrderFlagNegY = HEX 200
Go from top to bottom in vertical direction, else go from bottom to top.

scComSAMLightClientCtrlEntityArrayOrderFlagBiDir = HEX 800
Use bi-directional 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
The parameter Data then contains a long value with the char flag fields with the following possible values:

scComSAMLightClientCtrlLongDataIdTextCharFlagItalic = HEX 10000
scComSAMLightClientCtrlLongDataIdTextCharFlagRadial = HEX 20000
scComSAMLightClientCtrlLongDataIdTextCharFlagRadialAlignToCharOutline = HEX 40000
scComSAMLightClientCtrlLongDataIdTextCharFlagReverseOrder = HEX 80000
scComSAMLightClientCtrlLongDataIdTextCharFlagMirrorCharOnXAxis = HEX 100000
scComSAMLightClientCtrlLongDataIdTextCharFlagMirrorCharOnYAxis = HEX 200000
scComSAMLightClientCtrlLongDataIdTextCharFlagSwapLines = HEX 400000
scComSAMLightClientCtrlLongDataIdTextCharFlagSetToLimitLength = HEX 800000
scComSAMLightClientCtrlLongDataIdTextCharFlagSetToLimitHeight = HEX 1000000
scComSAMLightClientCtrlLongDataIdTextCharFlagSetToLimitKeepAspect = HEX 2000000

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 isn't 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 = HEX 0001
scComSAMLightClientCtrlLongDataIdBitmapModeGreyscale = HEX 0002
scComSAMLightClientCtrlLongDataIdBitmapModeDrillmode = HEX 0004
scComSAMLightClientCtrlLongDataIdBitmapModeBidirectional = HEX 0008
scComSAMLightClientCtrlLongDataIdBitmapModeStartlastline = HEX 0010
scComSAMLightClientCtrlLongDataIdBitmapModeNolineincr = HEX 0020
No increment of line position

The following two Data-flags define what bitmap has to be displayed into the view. If none of them is set, nothing will be shown and the imported bitmap will disappear.

scComSAMLightClientCtrlLongDataIdBitmapModeShowBitmap = HEX 0100
scComSAMLightClientCtrlLongDataIdBitmapModeShowScanner = HEX 0200

scComSAMLightClientCtrlLongDataIdTextWeight = 50
Using this DataId the style of a text can be modified. Please note: for an italic style refer to data values of the scComSAMLightClientCtrlLongDataIdTextCharFlags DataId. 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

scComSAMLightClientCtrlLongDataIdEnableHatching1 = 51

scComSAMLightClientCtrlLongDataIdEnableHatching2 = 52
Using these DataIds the both hatchers can be enabled or disabled or the actual enabling 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 start count 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:

SC_SAMLIGHT_CLIENT_CTRL_LONG_DATA_ID_
ENTITY_MARK_FLAG_MARK_CONTOUR 0x1

SC_SAMLIGHT_CLIENT_CTRL_LONG_DATA_ID_
ENTITY_MARK_FLAG_MARK_HATCH 0x2

scComSAMLightClientCtrlLongDataIdEntitySetPen = 60
The adjusted pen number of an entity can be changed.

scComSAMLightClientCtrlLongDataIdEntitySetTimerValue = 61
Using this constant the time value of a ScTimer object can be changed. The unit for the related long-value is milliseconds.

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. Please note: The long value that is handed over together with this constant can't 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 unit milliseconds). 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 isn't 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 aren't 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 / Set the start value of a serial number. This value is used after a serial number is reset.

scComSAMLightClientCtrlLongDataIdEntitySerialIncrValue = 72
Get / Set the increment value for a serial number object.

scComSAMLightClientCtrlLongDataIdEntitySerialCurrValue = 73
Get the current (actual) value for 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 bitfield defining which flags have to be set:

scComSAMLightClientCtrlLongDataIdEntityOpticFlagContour = HEX 1
scComSAMLightClientCtrlLongDataIdEntityOpticFlagHatch = HEX 2

scComSAMLightClientCtrlLongDataIdEntitySetAsBackgroundEntity = 78
Set the entity specified by ‹Entity Name› 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.

Double Data Ids
The following is a list of valid DataIds for the functions ScSetEntityDoubleData() and ScGetEntityDoubleData()

scComSAMLightClientCtrlDoubleDataIdFlagDontUpdateView = HEX 10000
If this bit is set, the view will not be refreshed. This can be helpful for increasing performance.

scComSAMLightClientCtrlDoubleDataIdFlagDontUpdateEntity = HEX 20000
If this bit is set, the entity will not be regenerated and updated. This can be helpful for increasing performance.

scComSAMLightClientCtrlDoubleDataIdFlagToplevelOnly = HEX 40000
If this bit is set, it will be only searched for entities in the first level of the job. This can be helpful for increasing 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. Unit: Rad.

scComSAMLightClientCtrlDoubleDataIdBitmapIntensity = 33
Changes the intensity of the related bitmap but doesn't re-create the appropriate scanner bitmap. Please refer to the bitmap mode flags for more information.

scComSAMLightClientCtrlDoubleDataIdBitmapBrightness = 34
Changes the brightness of the related bitmap but doesn't re-create 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 doesn't re-create 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. Please note: 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 hatcher number one/two.

scComSAMLightClientCtrlDoubleDataIdHatchAngle1 = 38
scComSAMLightClientCtrlDoubleDataIdHatchAngle2 = 46
Using this DataId the hatch angle for the first/second hatch can be get or set. Please note: the angle has to be specified in radians.

scComSAMLightClientCtrlDoubleDataIdHatchMinjump1 = 39
scComSAMLightClientCtrlDoubleDataIdHatchMinjump2 = 47
Hatch minimal jump value for hatch one/two

scComSAMLightClientCtrlDoubleDataIdHatchStartoffset1 = 40
scComSAMLightClientCtrlDoubleDataIdHatchStartoffset2 = 48
The start offset value for the first/second hatch

scComSAMLightClientCtrlDoubleDataIdHatchLinereduct1 = 41
scComSAMLightClientCtrlDoubleDataIdHatchLinereduct2 = 49
Specifies the amount of line reduction for the first/second hatch

scComSAMLightClientCtrlDoubleDataIdHatchEndoffset1 = 42
scComSAMLightClientCtrlDoubleDataIdHatchEndoffset2 = 50
The end offset value for the first/second hatch

scComSAMLightClientCtrlDoubleDataIdHatchBeamcompensation1 = 43
scComSAMLightClientCtrlDoubleDataIdHatchBeamcompensation2 = 51
Beam compensation value for the first/second hatch

scComSAMLightClientCtrlDoubleDataIdHatchNumloops1 = 44
scComSAMLightClientCtrlDoubleDataIdHatchNumloops2 = 52
Number of loops for the first/second hatch

scComSAMLightClientCtrlDoubleDataIdBarcodeLinereduction = 69
The line reduction of the barcode in percent for reducing the size of a bar code line.

scComSAMLightClientCtrlDoubleDataIdMotfOffset = 70
The distance of a ScMotfOffset control object.

scComSAMLightClientCtrlDoubleDataIdEntityRotationAngle = 71
This constant will return the current rotation angle of the entity in degrees. If the entity name is empty the return value is 0.

String Data Ids

The following is a list of valid DataIds for the function ScSetEntityStringData(), ScGetEntityStringData() and other string-related functions. These DataIds are splitted into two parts: the first one - here

marked with a "HEX" 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 aren't organized as a flag set and therefore can't be OR-concatenated. Here one of them has to be used exclusively.

scComSAMLightClientCtrlStringDataIdFlagDontUpdateView = HEX 10000
If this bit is set in the upper 16 bit of the parameter DataID, the view will not be refreshed. This can be helpful for increasing performance.

scComSAMLightClientCtrlStringDataIdFlagDontUpdateEntity = HEX 20000
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 for increasing performance.

scComSAMLightClientCtrlStringDataIdFlagToplevelOnly = HEX 200000
If this bit is set, it will only be searched for entities in the first level of the job. This can be helpful for increasing performance.

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.

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 doesn't modify the displayed barcode the upper 16 bits are ignored. So the view flags don't 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.

scComSAMLightClientCtrlStringDataIdGetEntityType = 22
This is used to retrieve the type of an entity by its index. The index is zero based and starts from 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...).

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 data into the motion controls that are 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.

Long CmdIDs
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
Fit the view to the working area.

scComSAMLightClientCtrlExecCommandFitViewToAllEntities = 5
Fit the view to all entities in the job.

scComSAMLightClientCtrlExecCommandFitViewToSelectedEntities = 6
Fit 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.

scComSAMLightClientCtrlExecCommandAutoCompensateOff = 10
Scanner auto-calibration functionality (works only with hardware that supports it): turn auto calibration mode off

scComSAMLightClientCtrlExecCommandAutoCompensateRef = 11
Scanner auto-calibration functionality (works only with hardware that supports it): turn 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): recalibrate

scComSAMLightClientCtrlExecCommandResplitJob = 13
Resplit a job that is in splitting mode and was modified so that the splitted 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.

scComSAMLightClientCtrlExecCommandRedPointerStop = 21
Stops the red pointer.

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.

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*)

7.3.5 Examples

Demo program
A demo application "samlight_client_cpp" shows the integration of the Client Control Interface. It is a Visual C++ project which is available for free from SCAPS GmbH.

Rotate output matrix
The following function uses the OCX to mark an entity with the name "RotateEntity". Then it rotates the output matrix for 10 degrees around the middle of the entity centre and marks it again. This step will be repeated 20 times.

```
if(m_samlight.ScIsRunning()==0)
{
      MessageBox("SAMLight not found","Warning",MB_OK);
      return;
}
double  min_x,min_y,max_x,max_y;
double  center_x,center_y;
long    i;
double  ang_inc;
double  act_angle;
CString entity_name = "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 = 10. * 6.28  / 360.;
act_angle = 0.;
for(i=0;i<20;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;
}
```

FitToEntity

The following Visual Basic source code shows the function to fit the view to a specific entity. Here the ActiveX is utilized too.

```
Call ScSamlightClientCtrl1.ScSetEntityLongData("",
scComSAMLightClientCtrlLongDataIdEntitySelected Or
scComSAMLightClientCtrlLongDataIdFlagDontUpdateView, 0)      ' unselect
all
Call ScSamlightClientCtrl1.ScSetEntityLongData(EntityName,
scComSAMLightClientCtrlLongDataIdEntitySelected, 1)  ' select
Call ScSamlightClientCtrl1.ScExecCommand
(scComSAMLightClientCtrlExecCommandFitViewToSelectedEntities)
```

SetArray

The following Visual Basic source code shows the function to change the array parameters of an entity by using the OCX interface to the Client Control. It creates an array 3* 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.

```
Call ScSamlightClientCtrl1.ScSetEntityLongData(EntityName,
scComSAMLightClientCtrlLongDataIdEntityArrayCountX Or
scComSAMLightClientCtrlLongDataIdFlagDontUpdateView, 3)
Call ScSamlightClientCtrl1.ScSetEntityLongData(EntityName,
scComSAMLightClientCtrlLongDataIdEntityArrayCountY Or
scComSAMLightClientCtrlLongDataIdFlagDontUpdateView, 4)
Call ScSamlightClientCtrl1.ScSetEntityLongData(EntityName,
scComSAMLightClientCtrlLongDataIdEntityArrayStepX Or
scComSAMLightClientCtrlLongDataIdFlagDontUpdateView, 3300)
Call ScSamlightClientCtrl1.ScSetEntityLongData(EntityName,
scComSAMLightClientCtrlLongDataIdEntityArrayStepY Or
scComSAMLightClientCtrlLongDataIdFlagDontUpdateView, 4500)
Call ScSamlightClientCtrl1.ScSetEntityLongData(EntityName,
scComSAMLightClientCtrlLongDataIdEntityArrayOrderFlags Or
scComSAMLightClientCtrlLongDataIdFlagDontUpdateView,
scComSAMLightClientCtrlEntityArrayOrderFlagNegX)
Call ScSamlightClientCtrl1.ScExecCommand
(scComSAMLightClientCtrlExecCommandFitViewToAllEntities)
```

Get and set text properties

The following Visual Basic source code shows how to manipulate the properties of a text object by using the OCX interface.

```
Dim DONTUPDATE As Long
DONTUPDATE = scComSAMLightClientCtrlLongDataIdFlagDontUpdateView Or
scComSAMLightClientCtrlLongDataIdFlagDontUpdateEntity

' we set the text properties of the entity with name EntityName
Call ScSamlightClientCtrl1.ScSetEntityStringData(EntityName, DONTUPDATE
Or scComSAMLightClientCtrlStringDataIdTextText, "NewText")
Call ScSamlightClientCtrl1.ScSetEntityStringData(EntityName, DONTUPDATE
Or scComSAMLightClientCtrlStringDataIdTextFontName, "Arial")
Call ScSamlightClientCtrl1.ScSetEntityDoubleData(EntityName, DONTUPDATE
Or scComSAMLightClientCtrlDoubleDataIdTextSize, 2)
Call ScSamlightClientCtrl1.ScSetEntityDoubleData(EntityName, DONTUPDATE
Or scComSAMLightClientCtrlDoubleDataIdTextCharSpacing, 0.95)
Call ScSamlightClientCtrl1.ScSetEntityDoubleData(EntityName, DONTUPDATE
Or scComSAMLightClientCtrlDoubleDataIdTextLengthLimit, 4.5)
Call ScSamlightClientCtrl1.ScSetEntityDoubleData(EntityName, DONTUPDATE
Or scComSAMLightClientCtrlDoubleDataIdTextHeightLimit, 1.5)
Call ScSamlightClientCtrl1.ScSetEntityDoubleData(EntityName, DONTUPDATE
Or scComSAMLightClientCtrlDoubleDataIdTextRadius, 11.5)
Call ScSamlightClientCtrl1.ScSetEntityDoubleData(EntityName, DONTUPDATE
Or scComSAMLightClientCtrlDoubleDataIdTextStartAngle, 3.14 / 2)

Dim flags As Long
flags = ScSamlightClientCtrl1.ScGetEntityLongData(EntityName,
scComSAMLightClientCtrlLongDataIdTextCharFlags)

 ' now force radial text
flags = flags Or scComSAMLightClientCtrlLongDataIdTextCharFlagRadial
Call ScSamlightClientCtrl1.ScSetEntityLongData(EntityName,
scComSAMLightClientCtrlLongDataIdTextCharFlags, flags)

' we get the text properties of the entity with name EntityName
Dim str As String
Dim val As Double
Call ScSamlightClientCtrl1.ScGetEntityStringData(EntityName,
scComSAMLightClientCtrlStringDataIdTextText, str)
MsgBox str, vbOKOnly, "Text"

Call ScSamlightClientCtrl1.ScGetEntityStringData(EntityName,
scComSAMLightClientCtrlStringDataIdTextFontName, str)
MsgBox str, vbOKOnly, "FontName"

Call ScSamlightClientCtrl1.ScGetEntityDoubleData(EntityName,
scComSAMLightClientCtrlDoubleDataIdTextSize, val)
MsgBox val, vbOKOnly, "TextSize"
```

Retrieve Entities

The following C++ code fragment uses the ActiveX to retrieve the number of entities and afterwards fetches all their names. Here within this example these names are displayed within a message box. Instead of that these names could be used to manipulate these entities too.

```
BSTR     name;
long     i,cnt;

// first evaluate how much entities are available
cnt=m_samlight.ScGetLongValue
(scComSAMLightClientCtrlLongValueTypeToplevelEntityNum);
if (cnt>0) for (i=0; i<cnt; i++)
{
      // important: the BSTR-pointer has to be initialized, elsewhere
      // the COM-interface may crash!
      name=NULL;
      // get the name of the entity at the index position "i"
      m_samlight.ScGetIDStringData
      (scComSAMLightClientCtrlStringDataIdGetToplevelEntity,i,&name);
      MessageBox(CString(name));
}
```

Motion Control

The following Visual Basic source code shows 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 wouldn't work correctly. If one drive reaches its position limit given in the motion settings file or has already reached its new position before, it doesn't move any more.

```
ScSamlightClientCtrl1.ScSetLongValue
scComSAMLightClientCtrlLongValueTypeMotionWaitForEnd, 0 ' the
application comes back immediately

' 1. part
ScSamlightClientCtrl1.ScSetLongValue
scComSAMLightClientCtrlLongValueTypeMotionAxis, 0
ScSamlightClientCtrl1.ScSetDoubleValue
scComSAMLightClientCtrlDoubleValueTypeMotionAxisPosition, 100
ScSamlightClientCtrl1.ScSetDoubleValue
scComSAMLightClientCtrlDoubleValueTypeMotionAxisSpeed, 4

ScSamlightClientCtrl1.ScSetLongValue
scComSAMLightClientCtrlLongValueTypeMotionAxis, 1
ScSamlightClientCtrl1.ScSetDoubleValue
scComSAMLightClientCtrlDoubleValueTypeMotionAxisAngleRelative, 720
ScSamlightClientCtrl1.ScSetDoubleValue
scComSAMLightClientCtrlDoubleValueTypeMotionAxisSpeed, 1

ScSamlightClientCtrl1.ScExecCommand
(scComSAMLightClientCtrlExecCommandMotionGo)

ScSamlightClientCtrl1.ScSetLongValue
```

```
scComSAMLightClientCtrlLongValueTypeMotionAxis, -1 ' -1 means all axes

Do
   ScSamlightClientCtrl1.ScSetLongValue
scComSAMLightClientCtrlLongValueTypeMotionAxis, -1
Loop While ScSamlightClientCtrl1.ScGetLongValue
(scComSAMLightClientCtrlLongValueTypeMotionMoving)
MsgBox "All drives have stopped."

' 2. part
ScSamlightClientCtrl1.ScSetLongValue
scComSAMLightClientCtrlLongValueTypeMotionAxis, 0
ScSamlightClientCtrl1.ScSetDoubleValue
scComSAMLightClientCtrlDoubleValueTypeMotionAxisPositionRelative, 100

ScSamlightClientCtrl1.ScSetLongValue
scComSAMLightClientCtrlLongValueTypeMotionAxis, 1
ScSamlightClientCtrl1.ScSetDoubleValue
scComSAMLightClientCtrlDoubleValueTypeMotionAxisAngle, 720

ScSamlightClientCtrl1.ScExecCommand
(scComSAMLightClientCtrlExecCommandMotionGo)

Dim axis0 As Integer: axis0 = 1
Dim axis1 As Integer: axis1 = 1

Do

If axis0 = 1 Then
  ScSamlightClientCtrl1.ScSetLongValue
scComSAMLightClientCtrlLongValueTypeMotionAxis, 0
  If ScSamlightClientCtrl1.ScGetLongValue
(scComSAMLightClientCtrlLongValueTypeMotionMoving) = 0 Then
    MsgBox "Drive with Axis 0 has stopped."
    axis0 = 0
  End If
End If

If axis1 = 1 Then
  ScSamlightClientCtrl1.ScSetLongValue
scComSAMLightClientCtrlLongValueTypeMotionAxis, 1
  If ScSamlightClientCtrl1.ScGetLongValue
(scComSAMLightClientCtrlLongValueTypeMotionMoving) = 0 Then
    MsgBox "Drive with Axis 1 has stopped."
    axis1 = 0
  End If
End If

Loop While axis0 = 1 Or axis1 = 1
```

Handle Texts

This C example code uses the ASCII-protocol to modify a text, rotate it by 45° and evaluate its total width. The creation of the socket connection - that is completely operating system and programming

environment dependent - is not shown here. Also sending and receiving of the data is not part of that example. Here the assumption is made that the function DoSend() sends the command line while DoReceive() performs the reception of the answer.

```
char msgStr[SC_CCI_MAX_COMMANDLENGTH]; // a buffer for the ASCII command
lines that have to be sent, the
                                       // its maximum size is defined in
ScCciCommands.h

...                          // open the socket connection here
DoSend(SC_CCI_INITSTRING); // send the initialization string to set up
the interface for the ASCII protocol


sprintf(msgStr,SC_CCI_CMD_CHANGE_TEXT_BY_NAME,"MyEntity","NewText");
                             // create the command line that has to be
sent, that function call results in
                             // a command "ScCciChangeTextByName
("MyEntity", "NewText")\n" that is stored
                             // in msgStr
DoSend(msgStr);              // send the message via the already opened
socket
MessageBox(DoReceive());     // receive the answer and display it within a
MessageBox

sprintf(msgStr,SC_CCI_CMD_ROTATE_ENTITY,"MyEntity",100,100,45);
                             // creates the command "ScCciRotateEntity
("MyEntity", 100, 100, 45)\n"
DoSend(msgStr);              // execute that command...
DoReceive();                 // ...and receive the handshake answer

sprintf(msgStr,SC_CCI_CMD_GET_ENTITY_OUTLINE,"MyEntity",0);
DoSend(msgStr);              // send the command to get the minimum
horizontal position of the entities
                             // outline
outline_min_x=atof(DoReceive());
                             // convert the returned string into a double
value; please note: if fetching of
                             // the outline value failed, the returned
string is "NaN" (=not a number)
DoSend("ScCciGetEntityOutline(\"MyEntity\", 3)\n");
                             // an alternative possibility to send a
command, here all values are statically
                             // so that it is not necessary to "construct"
the command using sprintf()
outline_max_x=atof(DoReceive());
width=outline_max_x-outline_min_x; // calculate the width of the entity
"MyEntity"
```

Precalculating the mark time

```
Dim flags As Long
Dim save_flags As Long
```

```
Dim expected_time As Double
flags = ScSamlightClientCtrl1.ScGetMarkFlags
save_flags = flags
flags = flags Or scComSAMLightClientCtrlMarkFlagPreview
ScSamlightClientCtrl1.ScSetMarkFlags flags
ScSamlightClientCtrl1.ScMarkEntityByName EntityName, 1
ScSamlightClientCtrl1.ScSetMarkFlags save_flags
expected_time = ScSamlightClientCtrl1.ScGetDoubleValue
(scComSAMLightClientCtrlDoubleValueTypeLastExpectedMarkTime)

MsgBox expected_time, vbOKOnly, "expected Marking Time [s]"
```

Fast access to job entities in buffered trigger mode

In buffered trigger mode of the USC-1 the data can be marked very fast. That fast marking operation makes it necessary to access the related job entities in a different way because the normal method to send a modify command, to wait if it is finished and then to start marking may be too slow for some special applications. Therefore an internal queue can be used. In this case the next element of this queue is fetched after every external trigger signal. If there is no more information within the queue, the job is left unchanged, else it is modified according to the enqueued commands before the next marking operating takes place.

```
// Set the marking flags: external trigger
// (the buffering mode has to be enabled manually within the
// application settings) and no home jump (to save some time)
mFlags=m_samlight.ScGetMarkFlags();
mFlags|=scComSAMLightClientCtrlMarkFlagWaitForTrigger|
scComSAMLightClientCtrlMarkFlagDisableHomeJump;
m_samlight.ScSetMarkFlags(mFlags);

// fetch my first set of data
getMyNextMarkData();

// pre-fill the buffer for the first marking operations completely
while (true)
{
   txt=formatMyMarkData();

   // the following command works faster than the standard
ScChangeTextByName(), the parameters shown
   // here suppress the view-update to save some time and they put the
command into the queue
   if (m_samlight.ScSetEntityStringData(EntityName,
                                        scComSAMLightClientCtrlStringDat
aIdFlagDontUpdateView|
                                        scComSAMLightClientCtrlStringDat
aIdFlagEnqueueLastCtrlCmd|
                                        scComSAMLightClientCtrlStringDat
aIdTextText,
                                        txt))
   {
      // fetch my next set of data if the last one could be put to the
queue successfully
      getMyNextMarkData();
```

```
   }
   else
   {
      // the queue is full so marking can be started next
      break;
   }
}

i=0;
// first increase the priority of the feeding application; that might be
necessary to ensure that the
// data can be delivered at least as fast as they are fetched out of the
queue by the marking process
// Warning: high thread priorities can be critical on some systems if
they consume too much computing
// power and therefore disturb the operation of other applications!
SetThreadPriority(GetCurrentThread(),THREAD_PRIORITY_TIME_CRITICAL);

// start marking in triggered mode
m_samlight.ScMarkEntityByName("",0);
while (i<800)
{
   txt=formatMyMarkData();
   if (m_samlight.ScSetEntityStringData(EntityName,
                                        scComSAMLightClientCtrlStringDat
aIdFlagDontUpdateView|
                                        scComSAMLightClientCtrlStringDat
aIdFlagEnqueueLastCtrlCmd|
                                        scComSAMLightClientCtrlStringDat
aIdTextText,
                                        txt))
   {
      // fetch next set of my data if the last one could be put to the
queue successfully
      getMyNextMarkData();
   }
   else
   {
      // the queue is filled completely so it is highly recommended to
let the application sleep
      // at least as long as the duration of one marking operation (to
save computing power for
      // other applications
      Sleep(250);
   }
}
SetThreadPriority(GetCurrentThread(),THREAD_PRIORITY_NORMAL);
m_samlight.ScStopMarking();
```

Create a beam comped copy of an entity

The following example will create a beam comped copy out of the entity "Entity" with the name "BeamCompedEntity". The parameter Dist is defined in line 3 in mm.

```
ScSetStringValue
```

```
(scComSAMLightClientCtrlStringValueTypeStringPara1,"Entity");
ScSetStringValue
(scComSAMLightClientCtrlStringValueTypeStringPara2,"BeamCompedEntity");
ScSetDoubleValue(scComSAMLightClientCtrlDoubleValueTypeDoublePara1,0.1);
ScExecCommand(scComSAMLightClientCtrlExecCommandCreateBeamCompedCopy);
```

8 How to

This chapter contains tutorials for special topics.

8.1 Use Simple Fonts

This tutorial describes how to use the simple line fonts (Laser Fonts) in the SAM modules.

8.1.1 Simple Fonts Format

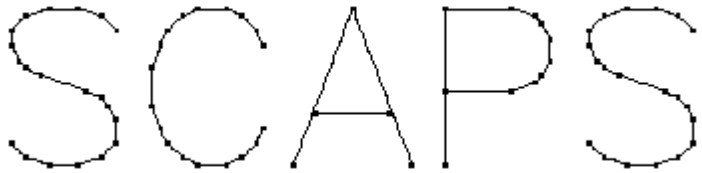
The simple fonts have to be stored as Windows True Type Fonts. They are generated with the SCAPS converter. The SAM software detects the type and only generates lines. The advantage of using the True Type Format is that the fonts can also be displayed in any Windows software supporting True Type Fonts.

Fonts in Microsoft Word:

Fonts in SAM2D View2D:

Zoom in Windows True Type:

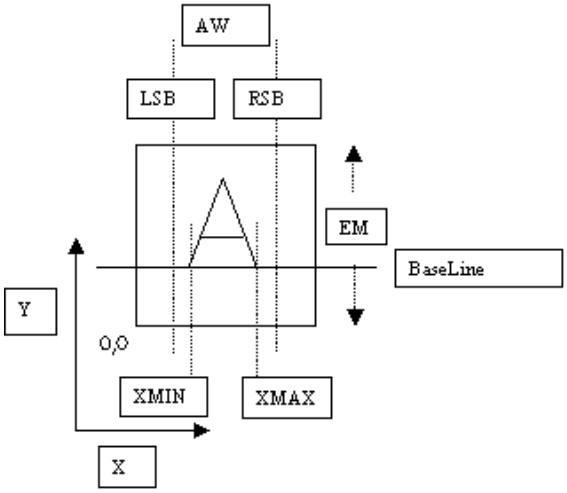
Zoom in SCAPS Simple True Type with point display:



8.1.2 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).



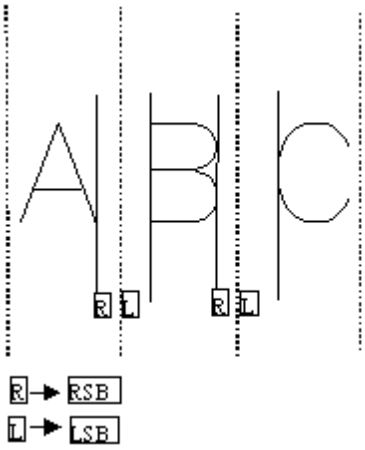
EM:
Size of square character cell.

XMIN:
Beginning of the character.

XMAX:
End of the character.

AW:
Advanced Width as
AW = LSB + RSB +(XMAX-XMIN) , or
RSB = AW-LSB-(XMAX-XMIN)

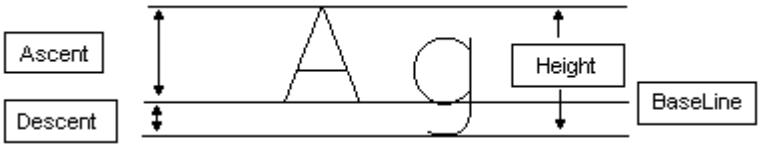
In True Type only AW and LSB are defined. RSB can be
calculated according the above equation.



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.

The baseline defines the line between the ascent and descent parameters of the font.



The sum of *Ascent* and *Descent* is the height of the font. *BaseLine*,*Height*, *Ascent* and *Descent* are global parameters of a specific font design.

8.1.2.1 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.

Example:

```
// CHAR 0
SI8000,0;SP1;PU;PA0,0;PA4720,3440,4480,2960,4480,2480,4960,2000,5440,2000,5920,224
0,6160,2720,6160,3200,5680,3680,5200,3680;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;PA4000,6800,3760,6320,3760,5840,4000,5600,4240,5
840,4000,6080;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;P
U;PA5680,6320;PD;PA5200,6800,4480,7040,3520,7040,2800,6800,2320,6320,2320,5840,2
560,5360,2800,5120,3280,4880,4720,4400,5200,4160,5440,3920,5680,3440,5680,2720,5
200,2240,4480,2000,3520,2000,2800,2240,2320,2720;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;PA6400,5120,6160,5360,5920,5360,5680,5120,5440,4
640,4960,3440,4480,2720,4000,2240,3520,2000,2560,2000,2080,2240,1840,2480,1600,2
960,1600,3440,1840,3920,2080,4160,3760,5120,4000,5360,4240,5840,4240,6320,4000,6
800,3520,7040,3040,6800,2800,6320,2800,5840,3040,5120,3520,4400,4720,2720,5200,2
240,5680,2000,6160,2000,6400,2240,6400,2480;PU;
// CHAR 39
SI1280,640;SP1;PU;PA4000,7040;PD;PA4000,5360;PU;
```

Header:

```
// SCAPS FONT FILE
// VERSION 2.0
```

The header is necessary for the font converter.

Baseline:

```
// BASELINE 2000
```

Y Distance from (0,0)

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.

8.1.2.2 Scaps Converter

This tool converts SCAPS Font Files into Windows True Type Font files. It is currently limited to fonts with a character id's 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.

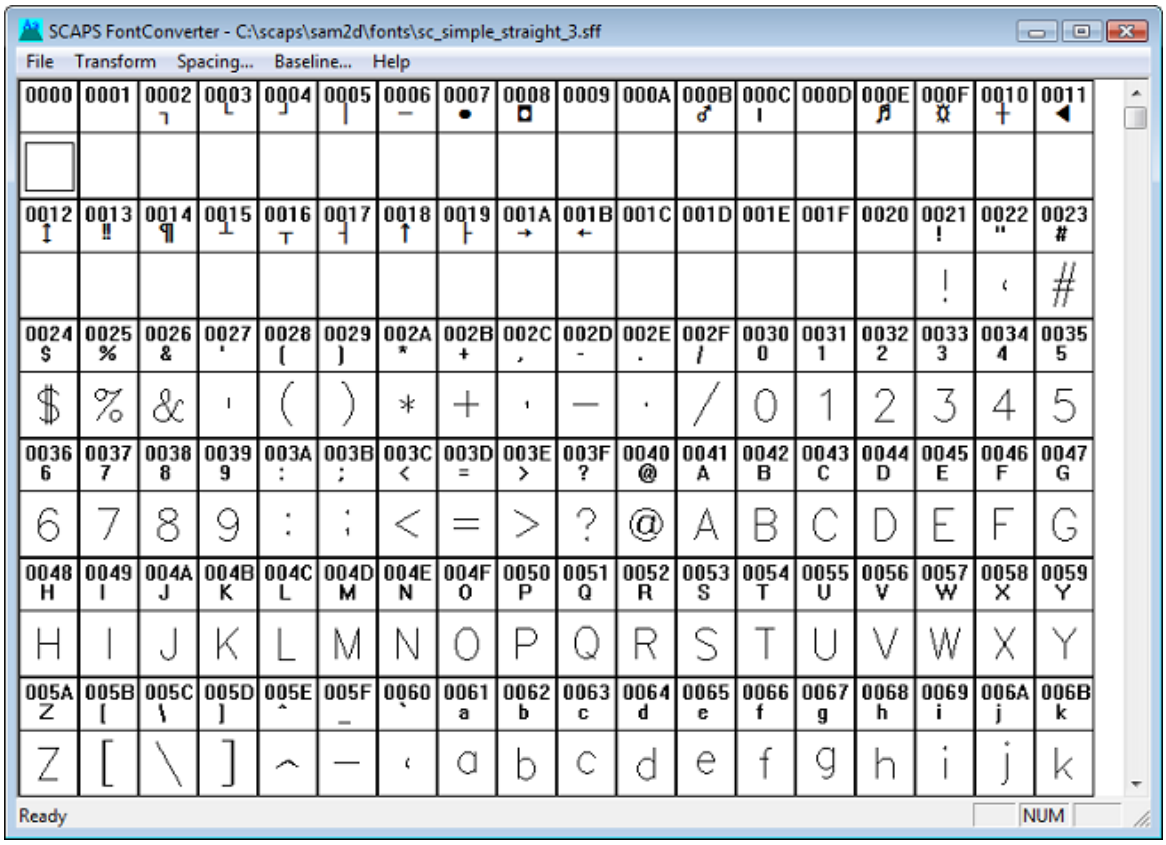


Figure 8.1: Font Converter Main View

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.

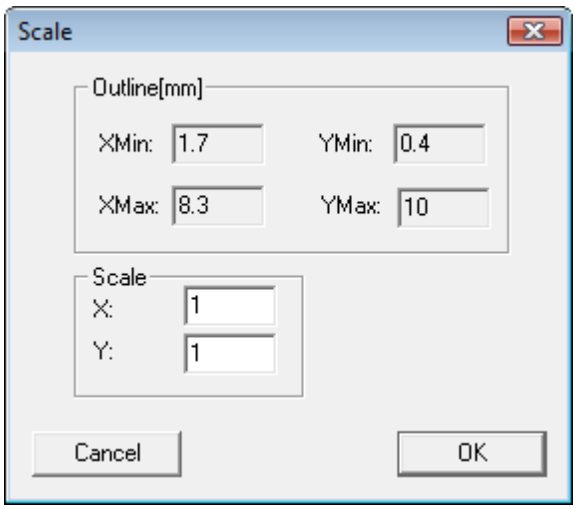


Figure 8.2: Transform Font Dialog

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".

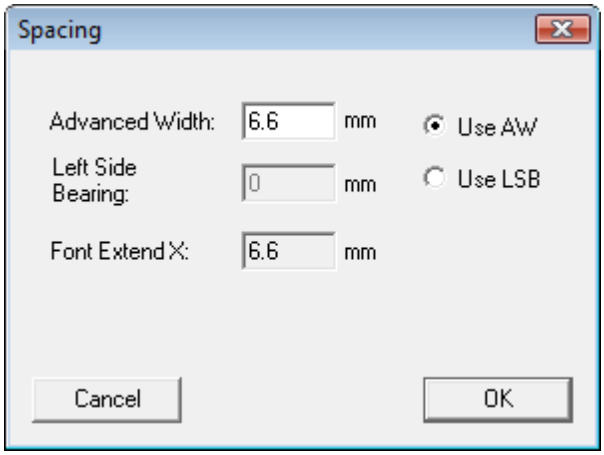


Figure 8.3: Font Spacing Dialog

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:

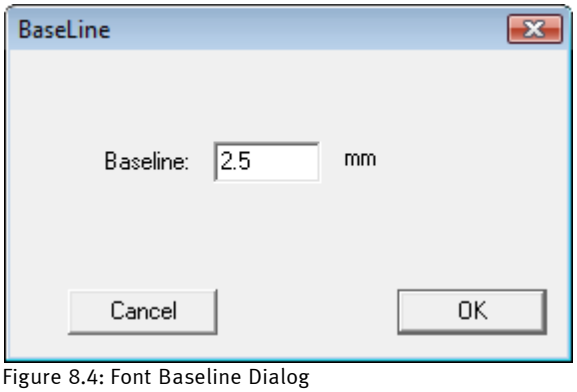


Figure 8.4: Font Baseline Dialog

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.

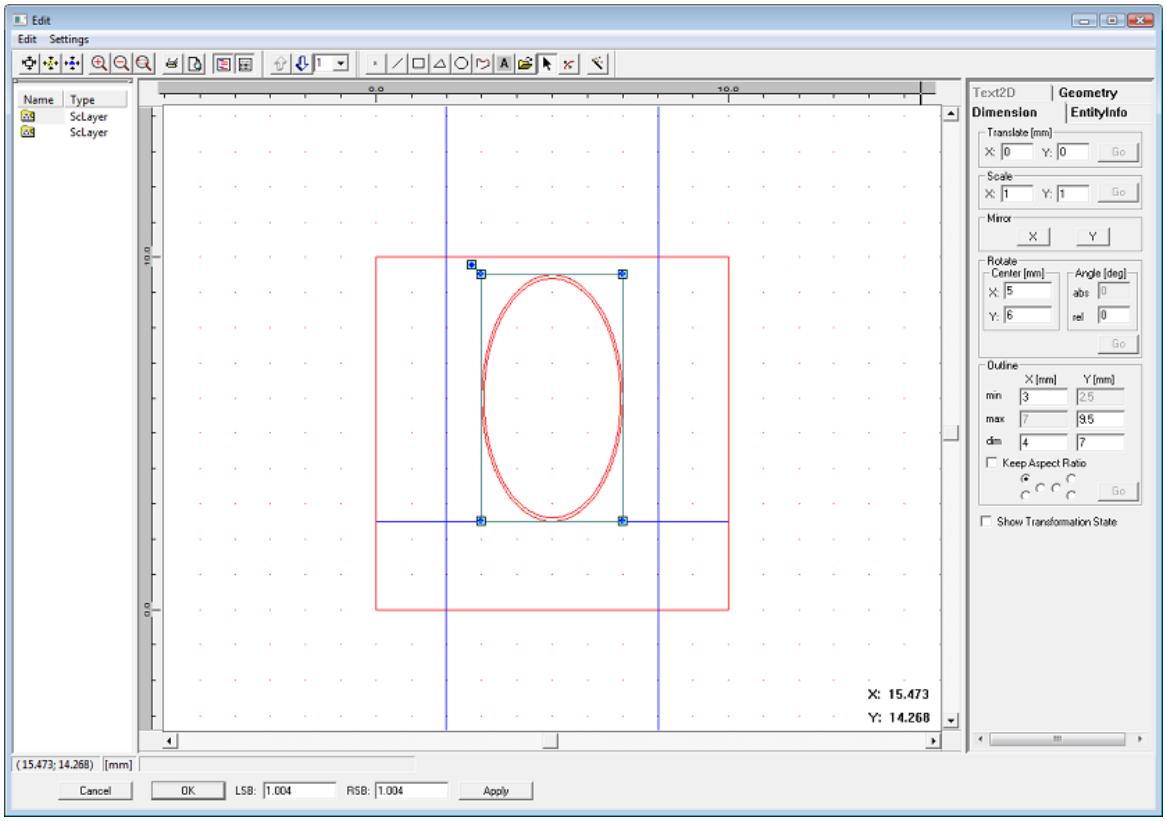


Figure 8.5: Font Edit View

Convert:

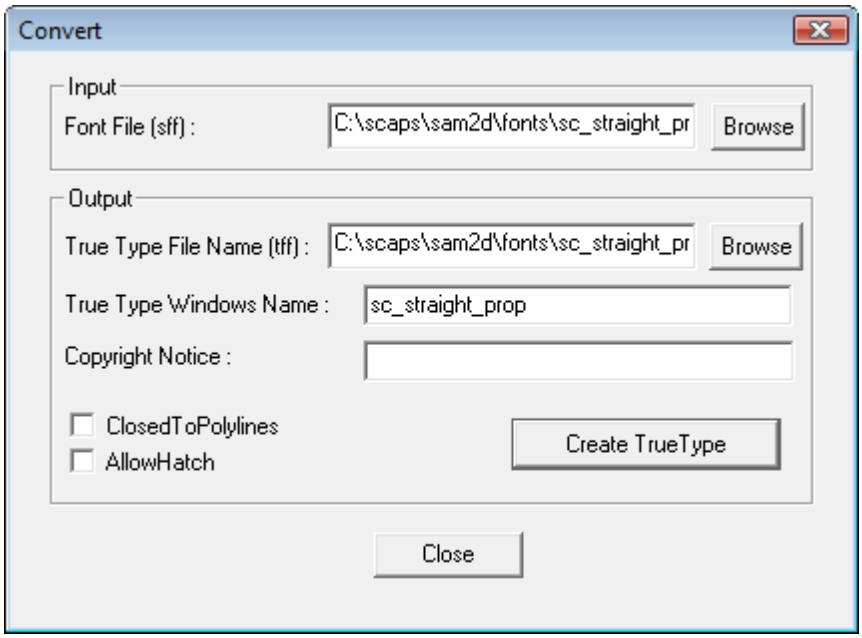


Figure 8.6: Font Convert Dialog

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.

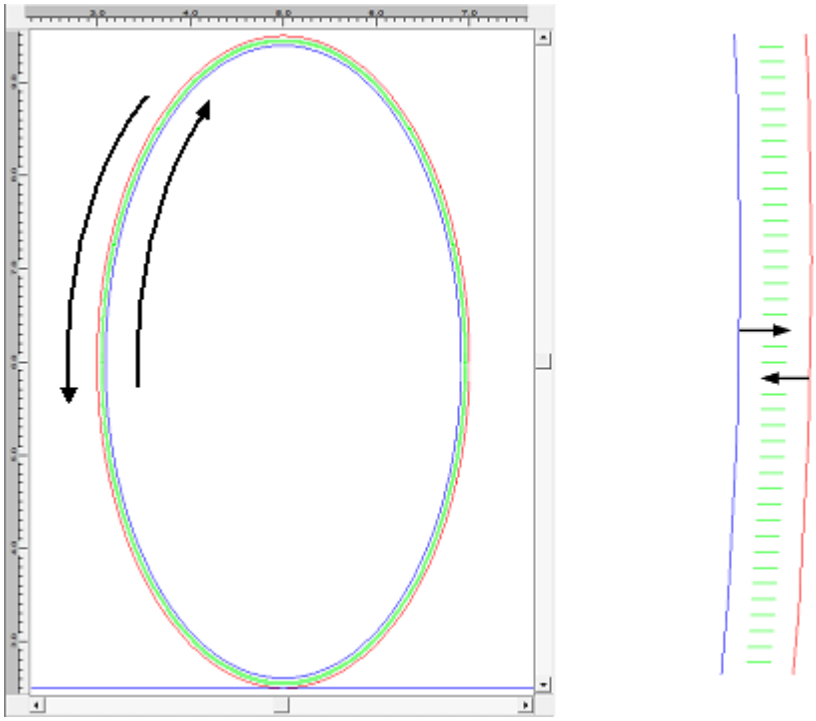


Figure 8.7: PolyLine Orientation Example

Outer PolyLines are oriented counter clockwise, 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.

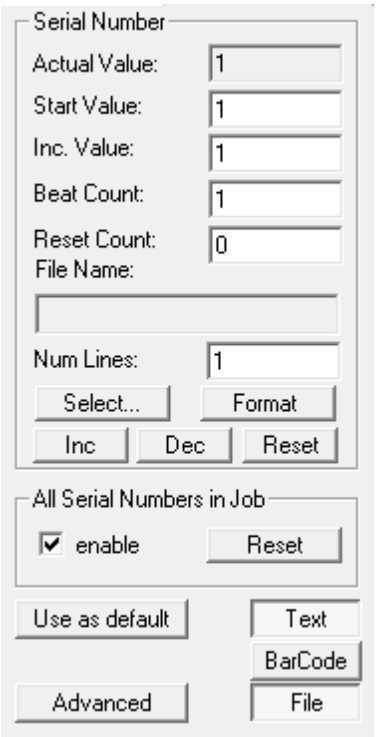
Adding own property pages:
The font converter allows to add own property pages to the Edit View. The ASCII file sc_font_convert.prp located inside the SAM system folder can be manipulated. This works in the similar way as adding a property page by calling of the ScEntityPropertySheet.ScAddPage(Name) command. By default the following pages are added:

SCAPS.ScEntityInfoPropertyCtrl
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.

8.2 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.



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.

Remark: One file can get assigned to more than one serial number.
See: Chapter Serial Number for the general properties of serial numbers which are also available when using the ASCII Serialization.

8.2.1 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.

8.2.2 Excel Table

This chapter explains how to assign an excel file to a serial number. After pressing the select button in the serial number property page a following dialog appears.

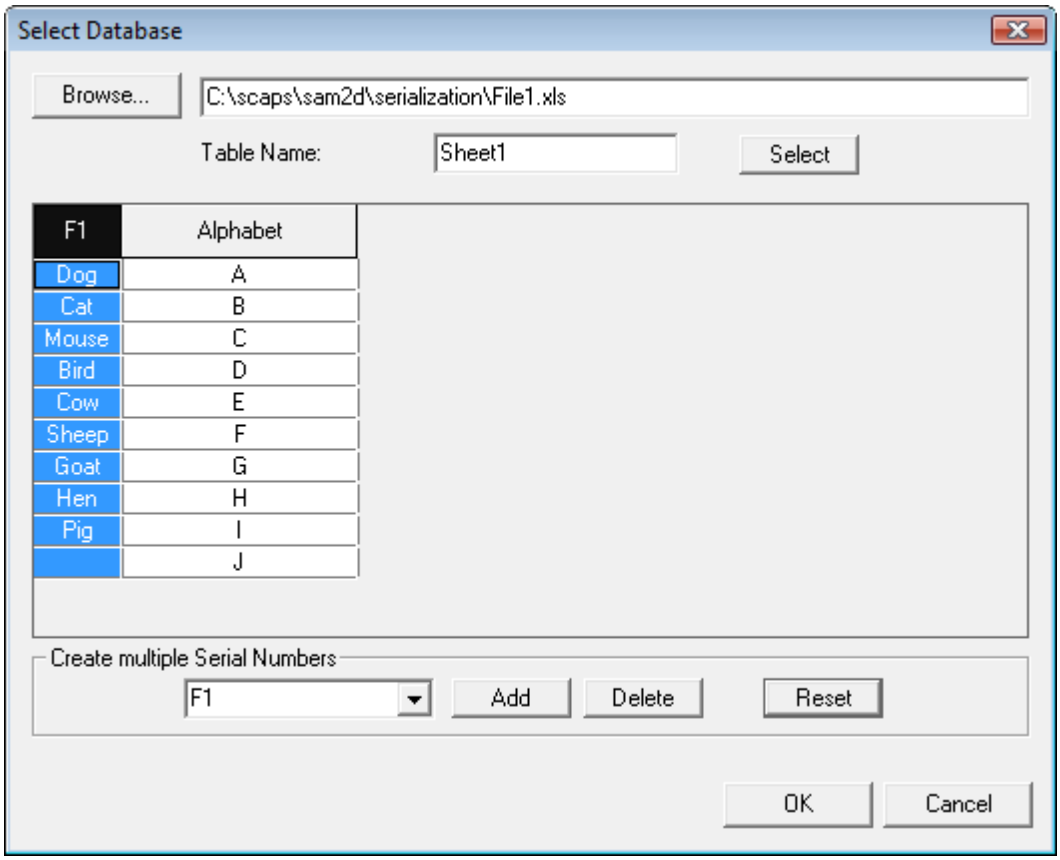


Figure 8.8: Select File for Serial Number Dialog

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.

Remark: If the excel file is being changed, the file needs to be reassigned to the serial number.

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.

8.2.3 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.

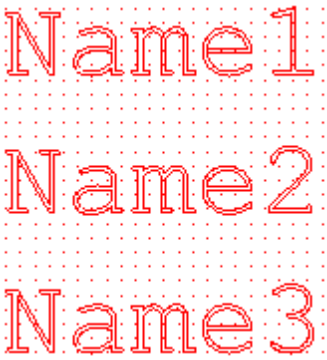
The names that are assigned to the pens needs to be saved in a text or excel file, for example:

Name1
Name2
Name3
Name4
Name5
Name6
Name7
Name8
Name9
Name10
:
:

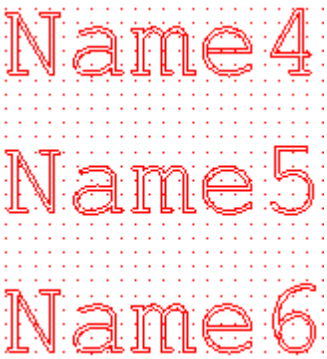
Now 3 serial numbers need to be created and the ASCII file has to be assigned to each of them. For the correct mapping of the serial string to the serial entity the following setup is used. *See also:* Chapter Serial Number.

	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:



8.3 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:

/JobEditor
SAMLight is started in JobEditor mode. Hardware output is not possible in this mode.

/Settingsfile=sc_light_settings_1.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_SAM%\system" (here %SCAPS_SAM% is the environment variable pointing to the installation path of the application).

/ActiveCard=‹number›
If more than one identical scanner cards are installed on the PC and if the driver of the card type supports multi card mode, it is possible to select the card to be used with this parameter. The number of the card is zero based. The valid range is 0-3.

/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 doesn't trie 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_to_jobfile›
Using this parameter a job defined by the path_to_jobfile 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 doesn't 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.

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_SAM%\system and make two copies of the existing file sc_light_settings.sam. Create:

%SCAPS_SAM%\sc_light_settings_yag.sam
%SCAPS_SAM%\sc_light_settings_co2.sam

Setup
Start %SCAPS_SAM%\tools\sc_setup.exe, go to menu *HardwareSettings*, select the file sc_light_settings_yag.sam, press "Load" and select your card type and within the settings the YAG mode. Save these settings and repeat this step for the CO2 file.

Create the icons
Create a sam_light.exe shortcut on the Windows desktop. You can drag the sam_light.exe located at %SCAPS_SAM%\samlight to the Windows desktop. Within the property of the newly created icon change the name of the application to e.g. "sam_light YAG".

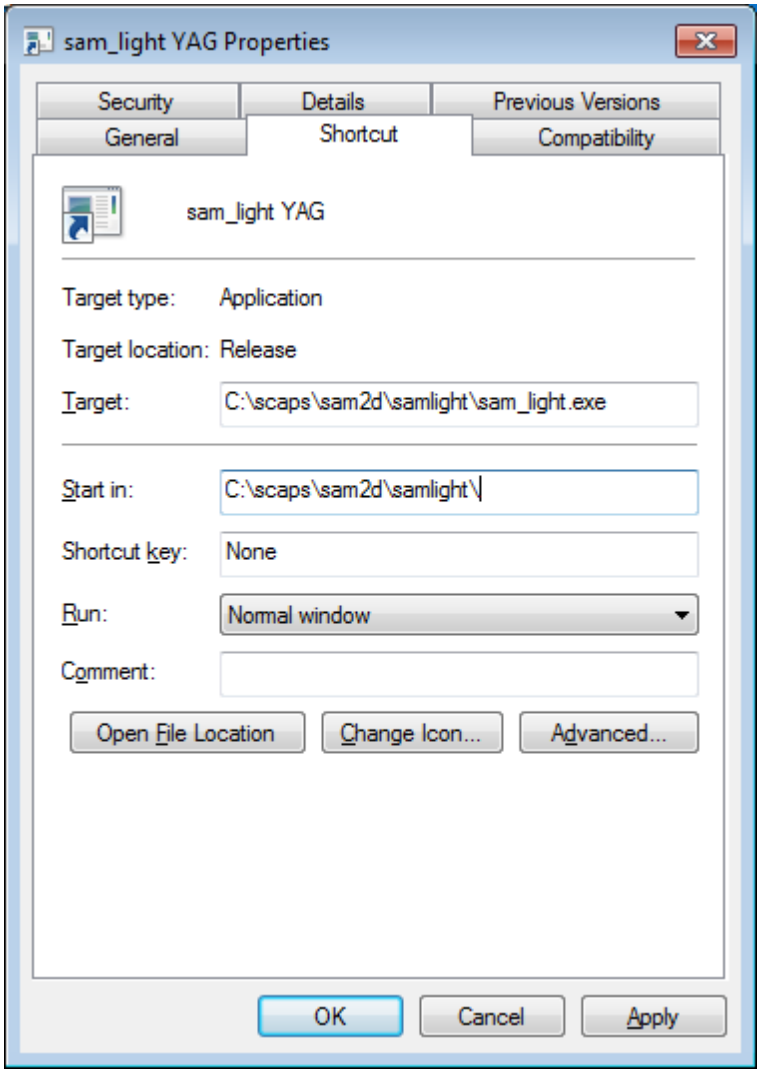


Figure 8.9: Creating a Shortcut to SAMLight

The property page can be opened by selecting the icon with the right mouse button.

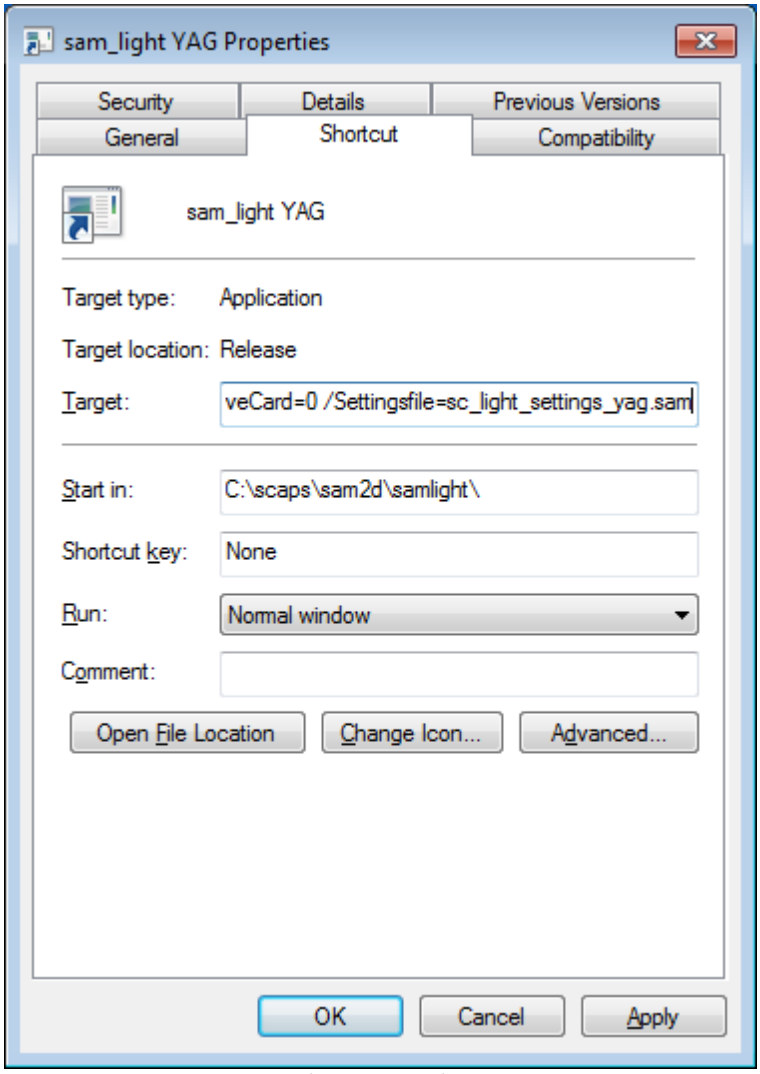


Figure 8.10: Apply a Settingsfile to the Shortcut

Now inside the property page of the icon, define the program arguments. Add the arguments, in this example:

/ActiveCard=0 /Settingsfile=sc_light_settings_yag.sam

The entire string in "Target" should be shown as follows...

C:\scaps\sam2d\samlight\sam_light.exe /ActiveCard=0 /Settingsfile=sc_light_settings_yag.sam

Now repeat these steps for the CO2-system.

8.4 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.

8.4.1 Personalize Program

All files to personalize the appearance are stored in the folder ...\scaps\scaps_sam\system. The following files can be substituted by personal ones.

sc_light_icon.ico
To define a personal icon for the desktop save it as sc_light_icon.ico into the folder ...\scaps \scaps_sam\system.

sc_light_logo.bmp
To generate a personal start window save your bitmap as sc_light_logo.bmp into the folder ...\scaps \scaps_sam\system.

sc_light_name.txt
To give the software an individual name, write this name into sc_light_name.txt.

8.4.1.1 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.sam
sc_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 txt-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 controll 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

where 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 doesn't 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 doesn't 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

Remark:
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

8.4.2 Customize Language

The ScResourceManager allows to change nearly all Strings used in the SAM Modules. It is possible to redefine the appearance of dialog boxes as well as to change messages. With this powerful feature the user can generate SAM user interfaces for his own language and make working at the machine easier to learn.

8.4.2.1 Global Settings

To activate the Resource Edit mode the following steps are necessary:

1. Quit all SAM based applications.
2. Start sc_setup.exe from folder SCAPSINSTALLDIR\tools
3. Click on the menu "Resource" to open the following dialog:

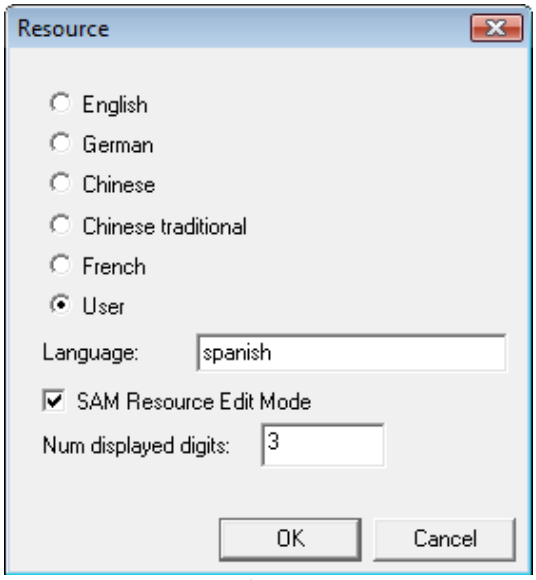


Figure 8.11: Resource Dialog

4. Select a default language or enter an User language.

5. If *User* is selected, *SAM Resource Edit Mode* is checkable to enable resource edit mode.

6. Quit sc_setup.exe and start your SAM software.
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 active, some "EditResource" buttons in the PropertyPage of the ScView2DCtrl and in the dialogs of the "OpticModule" 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 SCAPSINSTALLDIR\system and its name is 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 with sc_setup.exe.

8.4.2.2 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.

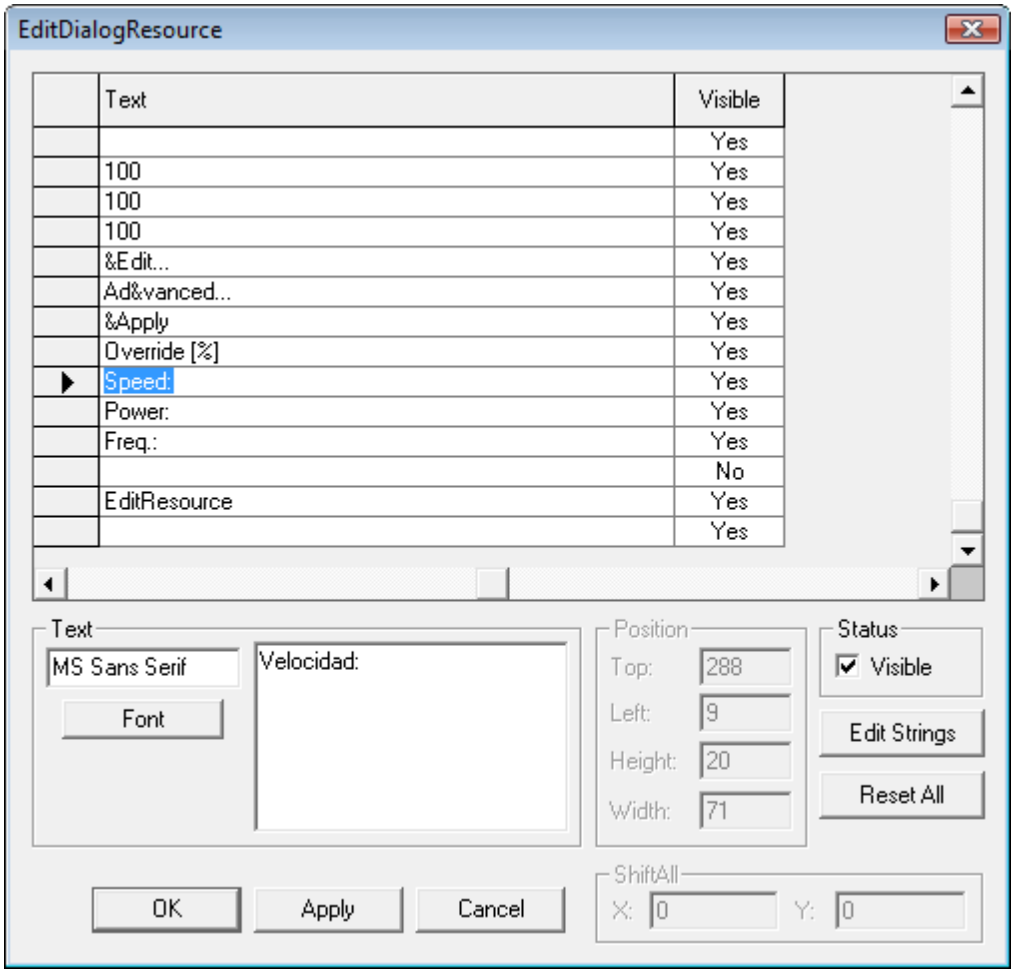


Figure 8.12: Resource Editor Dialog

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 don't 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:
The *Edit Strings* 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.

8.4.2.2.1 String Editor

Clicking on the *EditStrings* button within the dialog of the Resource Editor shows the following dialog:

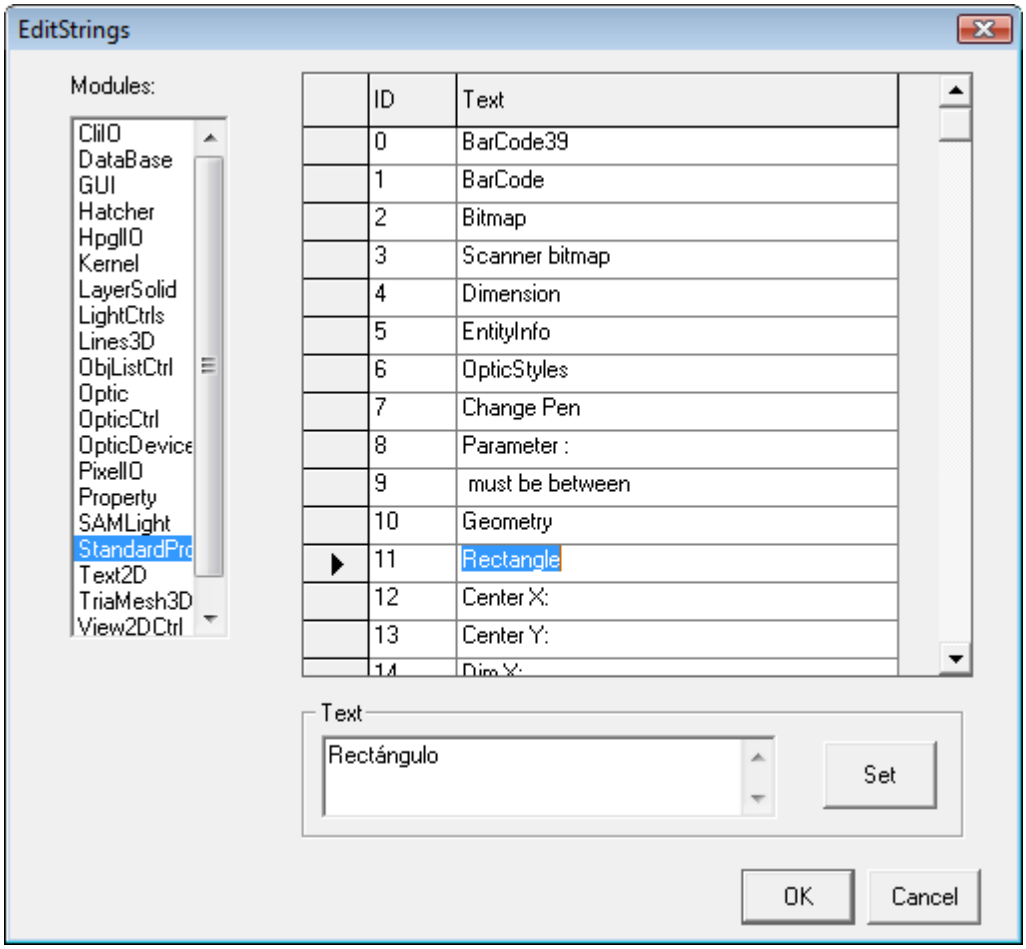


Figure 8.13: String Editor Dialog

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.

Note:

- The most important strings are inside the Kernel, the StandardProp, the Optic, the OpticCtrl and the 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 done after a restart of SAMLight.

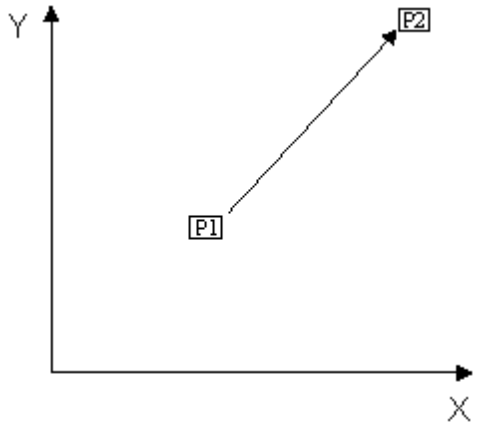
9 Backgrounds

In this chapter miscellaneous theoretical explanations are given for scanner card and program specific functioning.

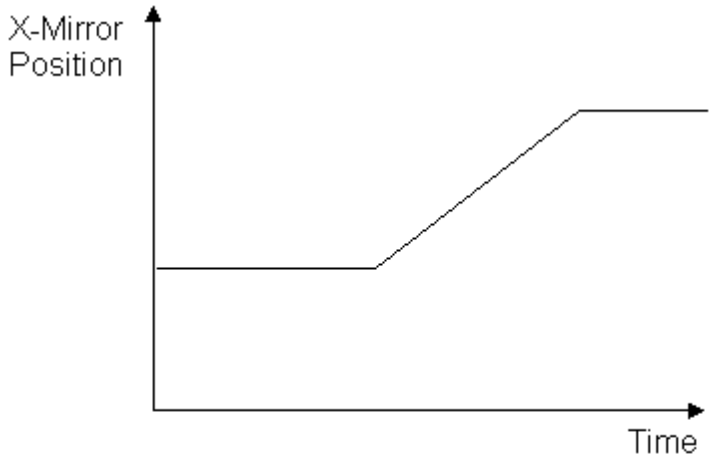
9.1 Scanner and Laser delays

The scanner and laser delays are defined in the laser style parameters dialogs. This chapter gives a short explanation of the delay terms.

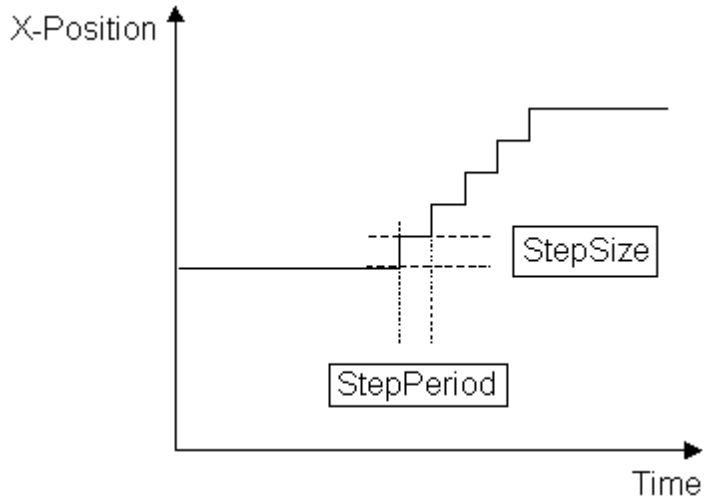
Assume that the XY Mirror system is commanded to go from P1 to P2 in XY-plane with a desired speed v.



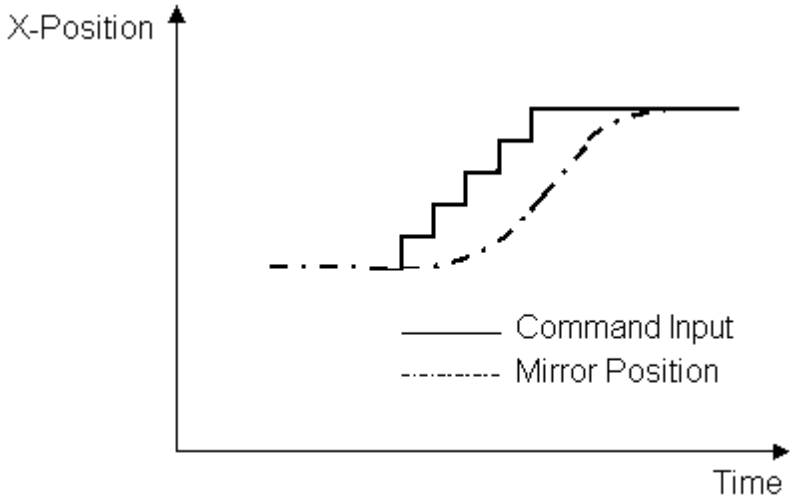
In the following only the X-Mirror is covered, the Y-Mirror is completely analogue.
The move command for the X-Mirror looks as follows:



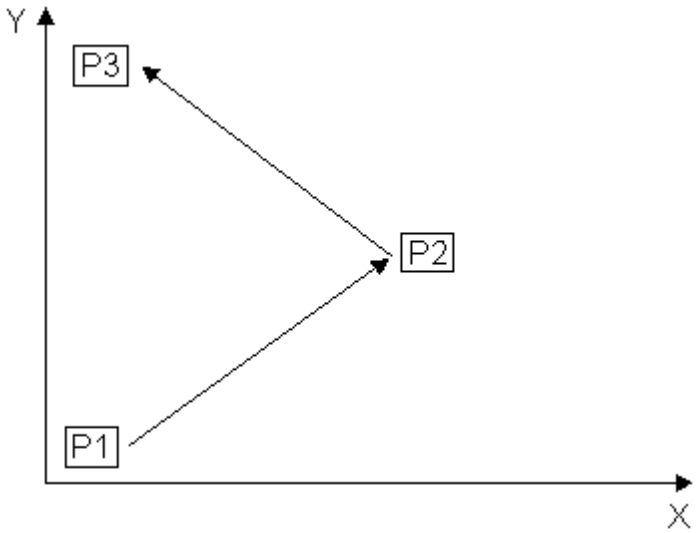
Since the controller card is not able to output values in an arbitrary short time period it has to approximate the desired curve in so called 'microsteps' with a time length of StepPeriod – typically 10 to 50 µs and a position change of StepSize. The requested speed v is the quotient from StepSize/ StepPeriod.

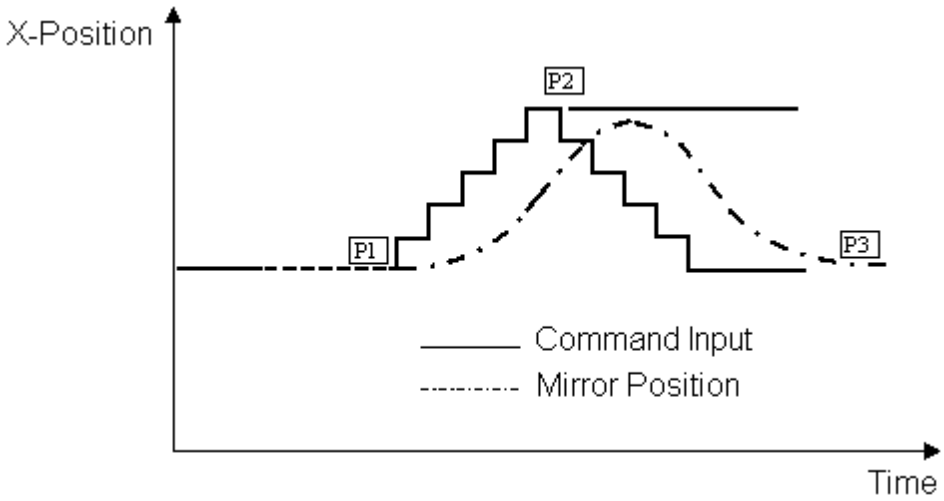


Since the X-Scanner with attached mirror is an inert system it can not follow the controller commands in short time but has some time lag.

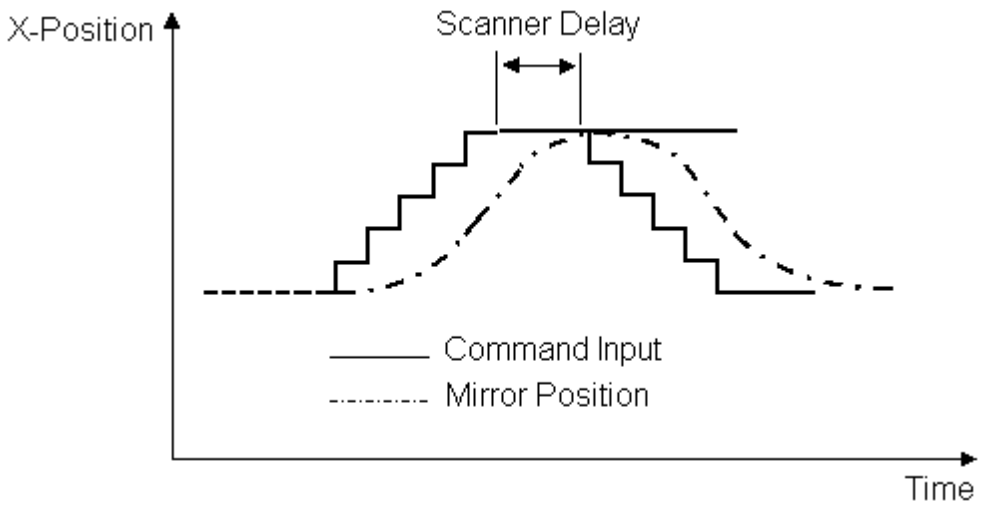


Due to this fact a command input like shown below would lead to the result that the X-Mirror would never reach position P2.





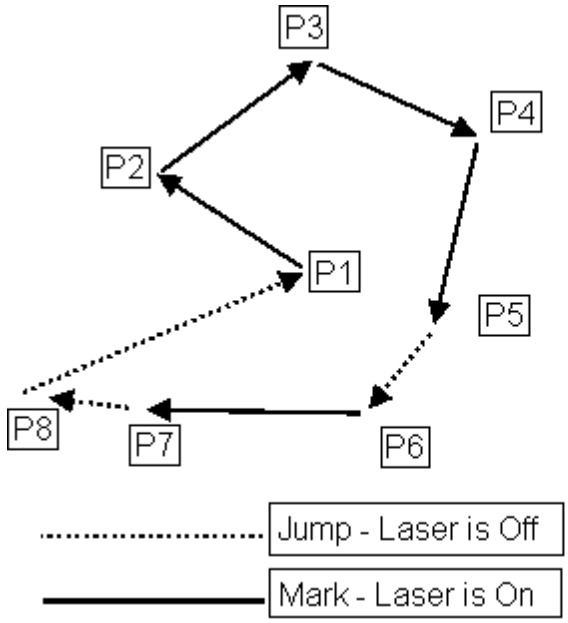
For this reason the controller card inserts a user definable delay between the end of the last vector and the start of the new vector.



There are 3 kinds of scanner delays.

Example for the scanner delays:

JumpDelay:	Points P1,P6 and P8 in the picture below.
MarkDelay:	Points P5 and P7 in the picture below.
PolyDelay:	Points P2,P3 and P4 in the picture below.

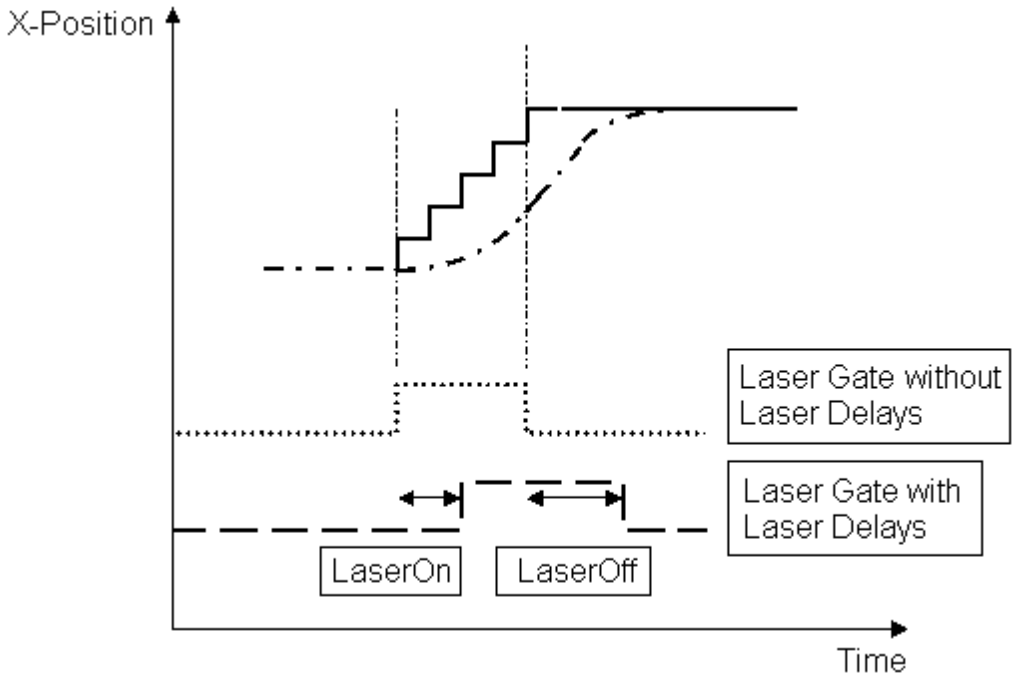


Since the X-Mirror can not accelerate in an arbitrary short time to the requested speed or deaccelerate to zero speed, two additional delays are used to control the delayed switch on and off of the laser gate signal. These are the laser on and the laser off delay.

LaserON delay: Time beginning from the output of the first microstep the controller card waits before it switches on the laser.

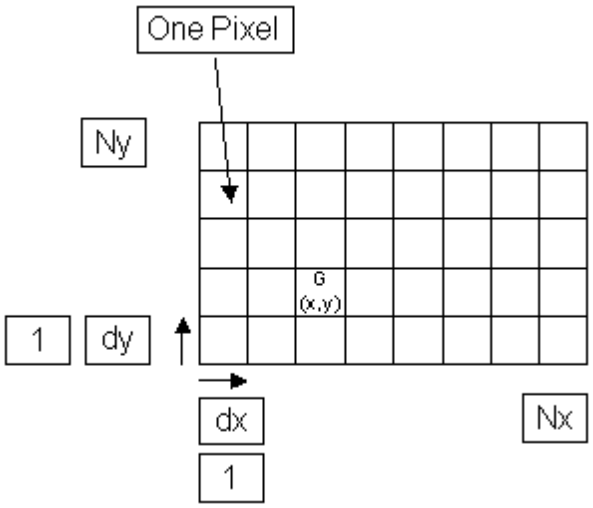
LaserOFF delay: Time beginning from the output of the last microstep the controller card waits before it switches off the laser.

Example for laser delays:



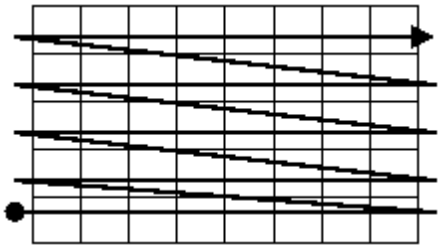
9.2 Pixelmode

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).



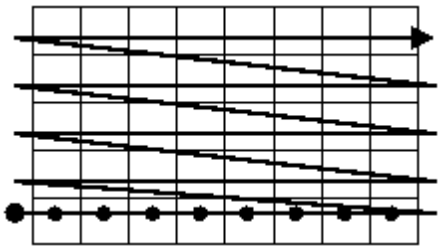
Each pixel inside a bitmap has the same X and Y dimension dx,dy. dx and dy itself may be different. The bitmap consists of Nx pixels in X direction and Ny pixels in Y direction. Each pixel has a Grayvalue G (x,y) form 0 to 1 which is typically transformed to a Grayvalue range from 0 to 255.

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.

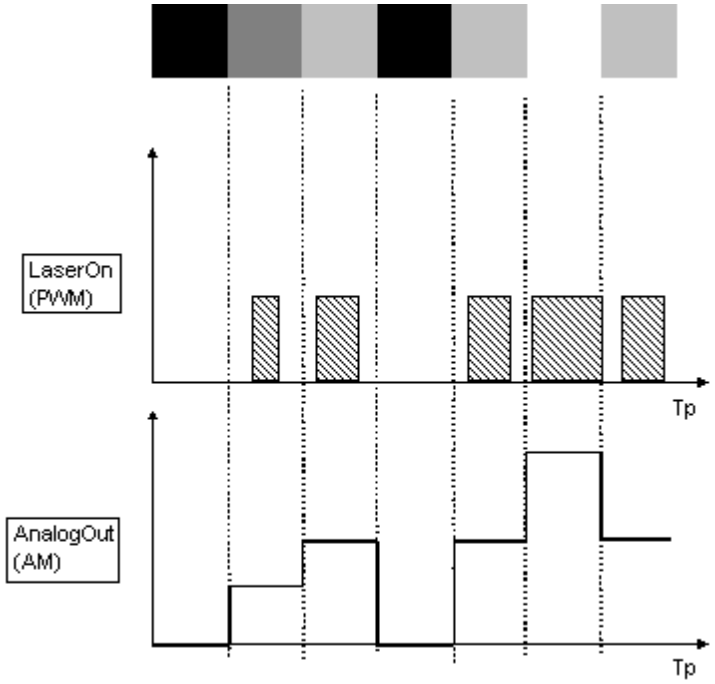
9.2.1 Pulse Modulation

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)

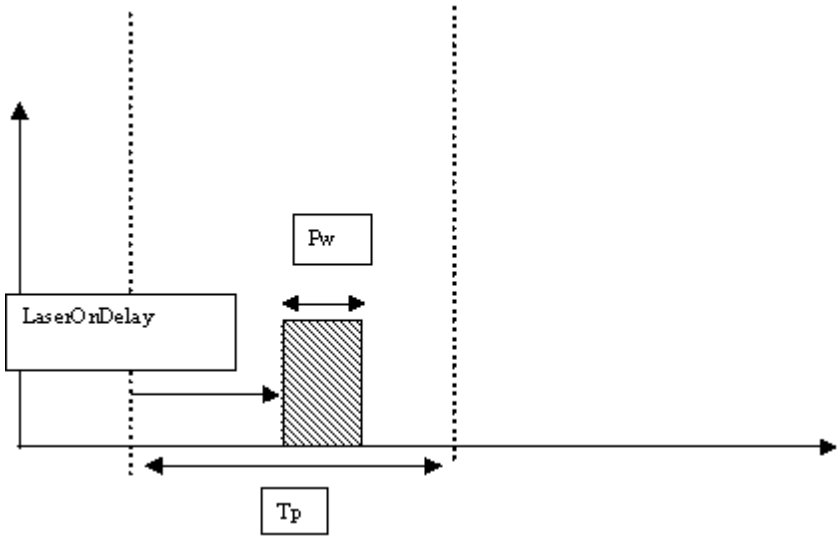
In PWM mode the LaserON Signal is modulated. In AM mode the analog output value of the Laserport is modulated. For a given speed V the time for one pixel is calculated by

$$Tp = dx / V.$$



Pulse Width Modulation (PWM)

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.

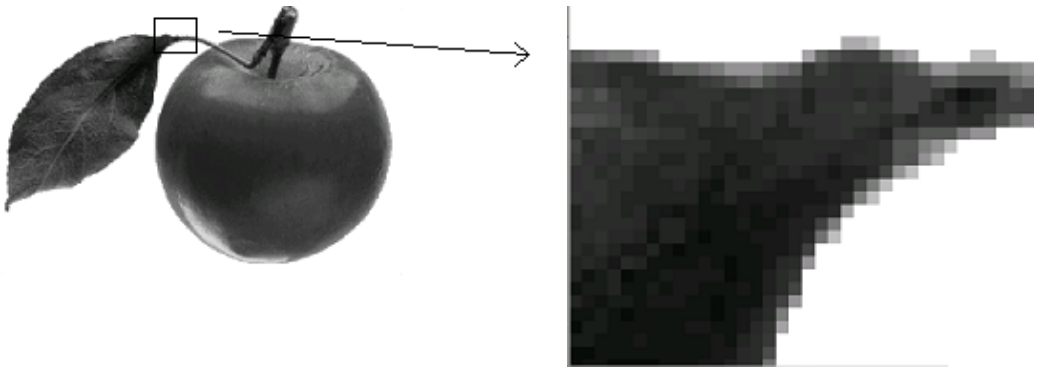
Analogoutput = GrayScaleValue * MaxOutput

MaxOutput corresponds to maximal achievable Output voltage of the Digital to Analog Converter Output.

Remark:
If one of the following conditions is valid no output takes place:
1. Tp ‹ 10 OR Tp › 655350
2. (Tp - LON - LOFF) ‹ = 0
3. dx == 0 AND dy == 0

9.2.2 Generating a scanner bitmap

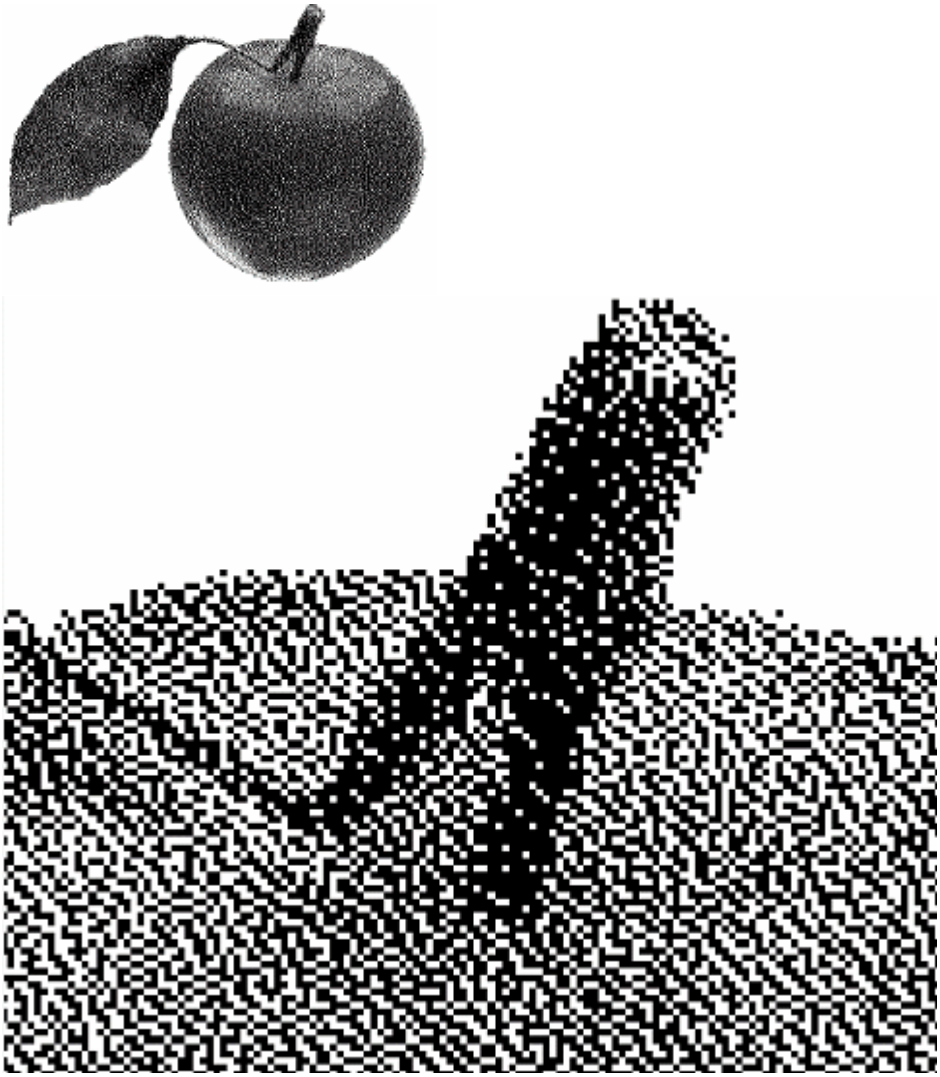
In the following the parameter setting with SAM for the 'apple.bmp' is shown.



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:

9.3 Object 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.

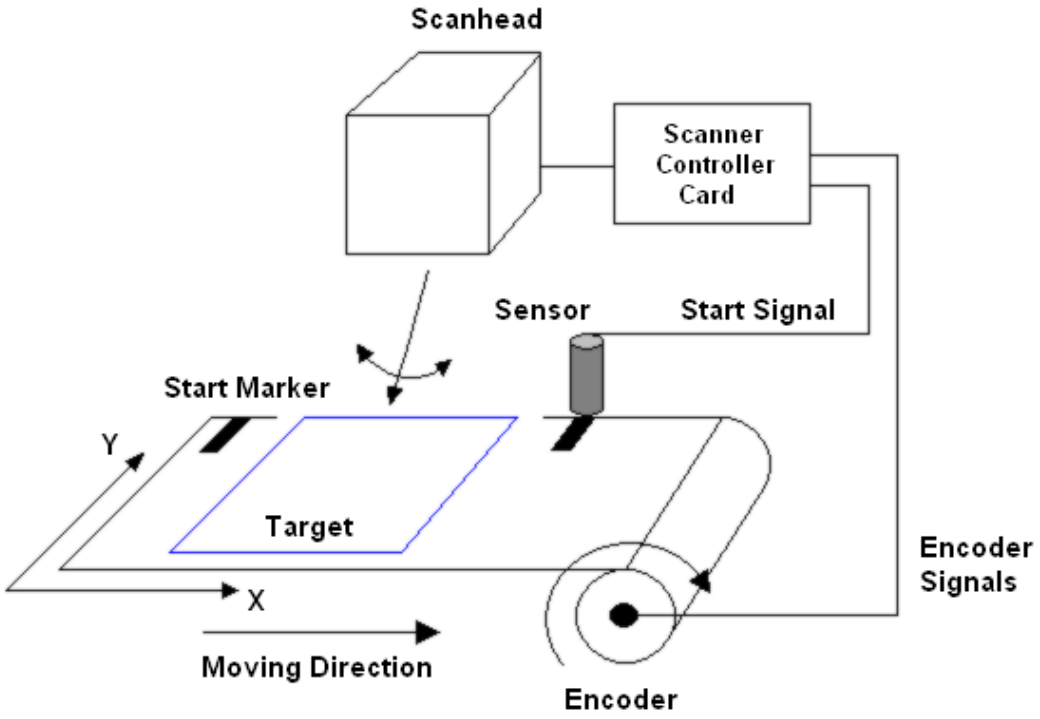
Layer:
The Layer contains exactly three elements: one PolyLines, one LineArrays and one PixelArrays. This entity type is used for geometrical objects with a heterogeneous object description which are especially hatchable objects. For example if one creates a rectangle (see Object Toolbar) an entity of 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).

10 Option MOTF

This feature is for marking targets on a product line. The chapter describes the settings within the application which have to be done for MarkingOnTheFly .

10.1 General Overview

The typical MarkingOnTheFly setup is shown below:



The target piece which has to be marked is placed on a moving conveyor belt that surpasses the scanhead 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 measured 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. To measure the relationship of encoder counts to the travelled distance the constant Kx is introduced:

$$Kx = XDistance\ [mm] / Counts$$

If, for example, the encoder gives 2500 Counts within 100 mm, then the Kx would be 100 / 2500 = 0.04. Kx also may be negative if the moving direction goes into the opposite. If Kx is defined, the encoder counts are translated to a distance. Then this distance information is being sent to the scanner card which appropriately corrects the marking vector to fit it on the moving target. The starting signal is sent by the **Sensor**. The sensor scans the conveyor belt for a specific label and sends a signal to the scanner controller card if this label has reached the position of the sensor.

For more details or if problems occur, please look at the SCAPS FAQ manual:

http://download.scaps.com/downloads/hardware/index.php?c=1&f=sc_faq.pdf

10.2 Simulation Operation Mode

If the belt is moving with a constant speed then the simulation mode can be used instead of connecting an encoder. The simulation mode simulates the encoder device by creating pulses with a defined frequency, for example 100 kHz. The number of pulses is then converted into a distance information in mm with the Kx value. This distance information is used by the scanner card to adjust the scanner for Marking on the Fly.

10.3 Card Specific: USC-1

Here the individual options for an USC-1 card are described. To get to the dialog window below select "Settings->System->Optic" in the menu bar and then click on the "Advanced" button.

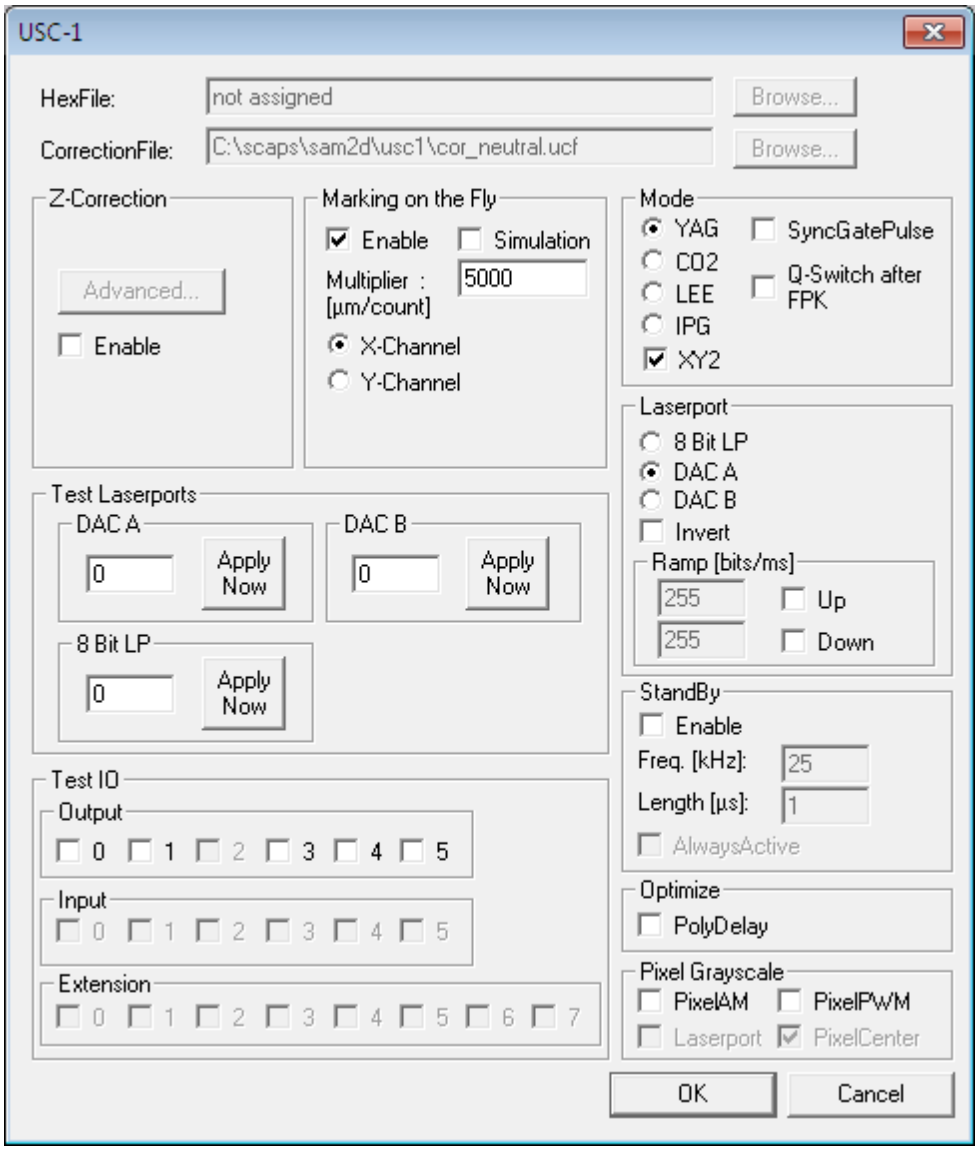


Figure 10.1: MOTF Settings for USC-1 Cards

Marking on the Fly:

Enable:
Activate this checkbox to enable Marking On The Fly.

Simulation:
Activate this checkbox to enable Simulation Operation Mode.

Multiplier [µm/count]:
Enter the Kx value here and note that it has to be entered in µm/count. If you enter a negative value the scanner will move to the other direction.

X/Y-Channel:
Here you can choose in which direction the target is moving during the marking on the fly.

10.4 Card Specific: USC-2

Here the individual options for an USC-2 card are described. To get to the dialog choose "Settings›System" from the Menu Bar. Then choose "Optic" from the register menu and click on "Advanced". Activate the "Enable" Checkbox in "Marking on the Fly" and then click on "Settings" above the checkbox.

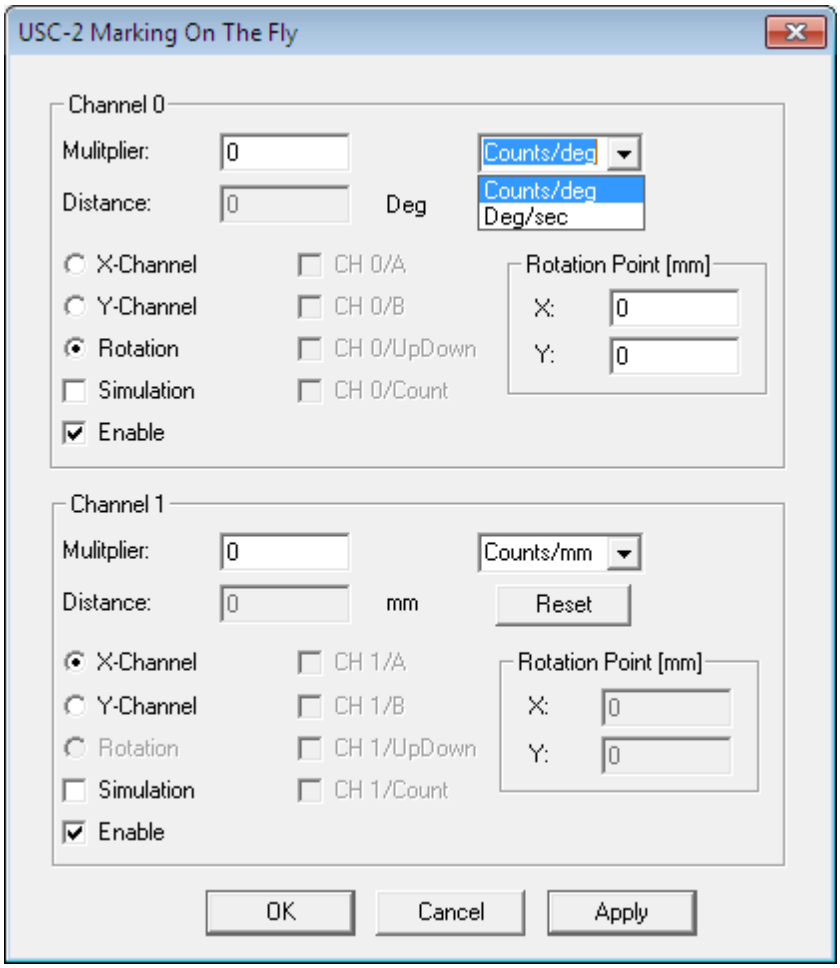


Figure 10.2: MOTF Settings for USC-2 Cards

Channel 0 / Channel 1:

Multiplier:
Enter the inverse Kx value here. If you enter a negative value the scanner will move to the other direction.

Distance:
If an encoder is used or simulation is enabled then 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-Channel:
Here you can choose in which direction the scanner is moving during the marking on the fly.

Simulation:
Enable this checkbox to enable the simulation mode as described in the chapter "Simulation Operation Mode".

Enable:
Activate this checbox to enable the channel for Marking On The Fly.

Drop Box:
Choose the unit for the inverse Kx multiplier. Default is Counts/mm.

Reset:
Resets the Distance counter to 0.

Rotation:
A special feature which is only available for an USC-2 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.

10.5 Card Specific: RTC cards

Here the individual options for RTC cards are described. These include RTC3, RTC4 and RTC5 as well as the RTCScanAlone card.To get to the dialog choose "Settings->System" from the Menu Bar. Then choose "Card" and click on "Advanced". There click on "Driver Settings". A new window "RTCxxx-Settings" pops up. There in the field "Mark o fly" activate the "Enable"-checkbox and then click on "Settings".

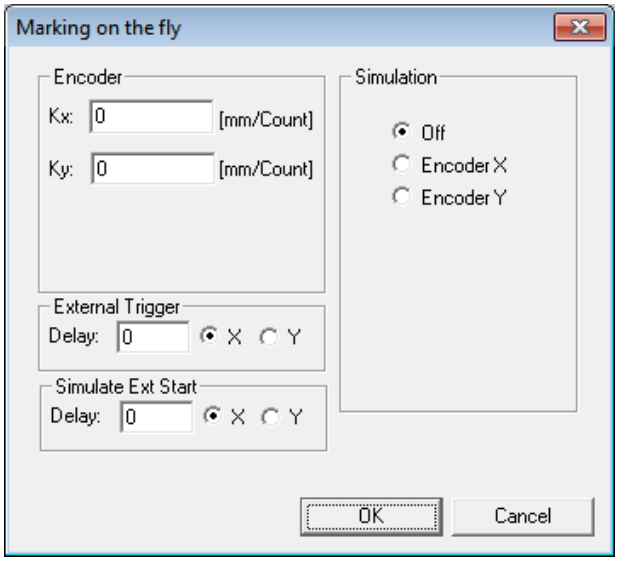


Figure 10.3: MOTF Settings for RTC Cards

Encoder:

Kx/Ky:
Distance in x/y direction per encoder count. Enter the Kx or Ky value here.

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

External Trigger:

Delay:
If marking on the fly should be started with an external trigger then the value here gives a delay in the unit of encoder counts. For that Simulation has to be set to "Off".

Simulate Ext Start:
Allows to insert trigger signals automatically after the specified number of counts.

Delay:
If there is no encoder the external trigger can be simulated and the delay is specified in the unit of encoder counts.

10.6 Examples

Here some examples for MOTF jobs are being introduced.

10.6.1 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 it can be done by the following solution:

The solution is different for USC cards and RTC cards. For the USC cards the Trigger Control Objects can be used together with the Mark Loop Count. Here 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 the menu Edit -› Group. Then in the EntitInfo property sheet set the Mark Loop Count of this group to -1.

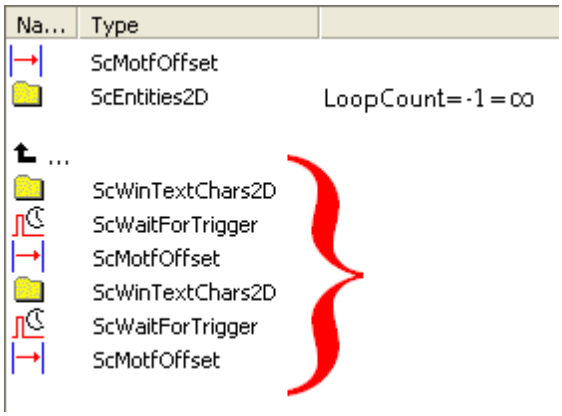
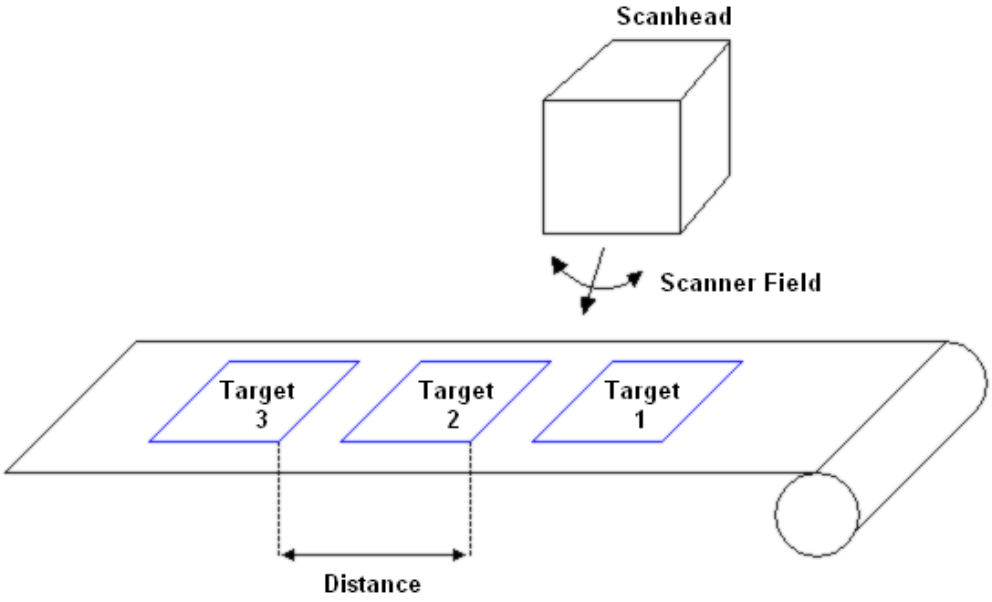


Figure 10.4: Example for MOTF Job for an Assembly Line

The RTC cards have a unique dialog to do this. Therefore look at the chapter "Card Specific: RTC cards". Please note that 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 card can handle the simulation mode together with the ScWaitForTrigger and ScMotfOffset objects.



11 Option Flash

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 (**F**lash **E**xtension **B**oard) 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 USC-1/USC-2 USB line.

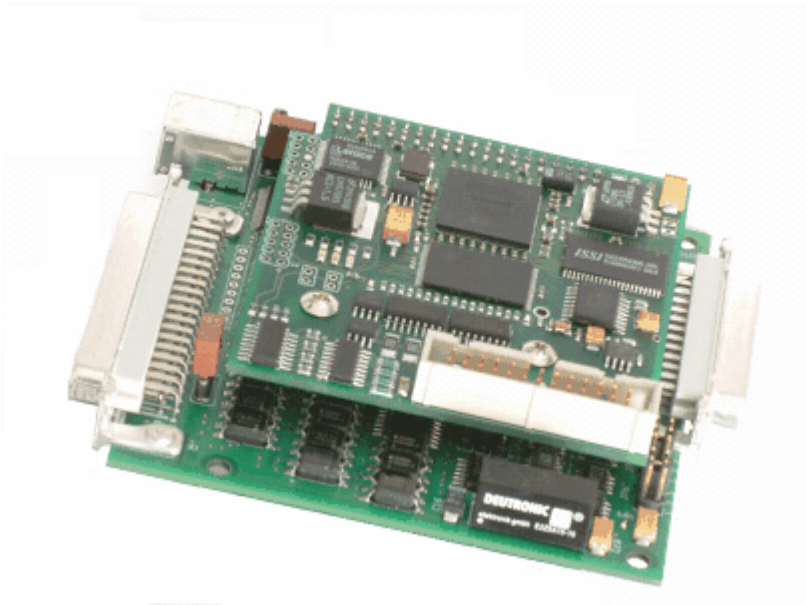


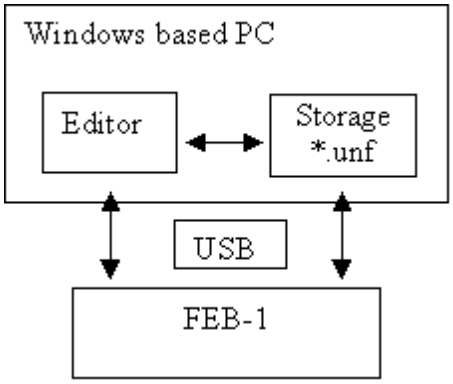
Figure 11.1: USC-1 Card equipped with a FEB-1 board

11.1 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.

Below the different possibilities are shown.



The general dialog to achieve this functions will be started by selecting the menu point Extras/Flash.

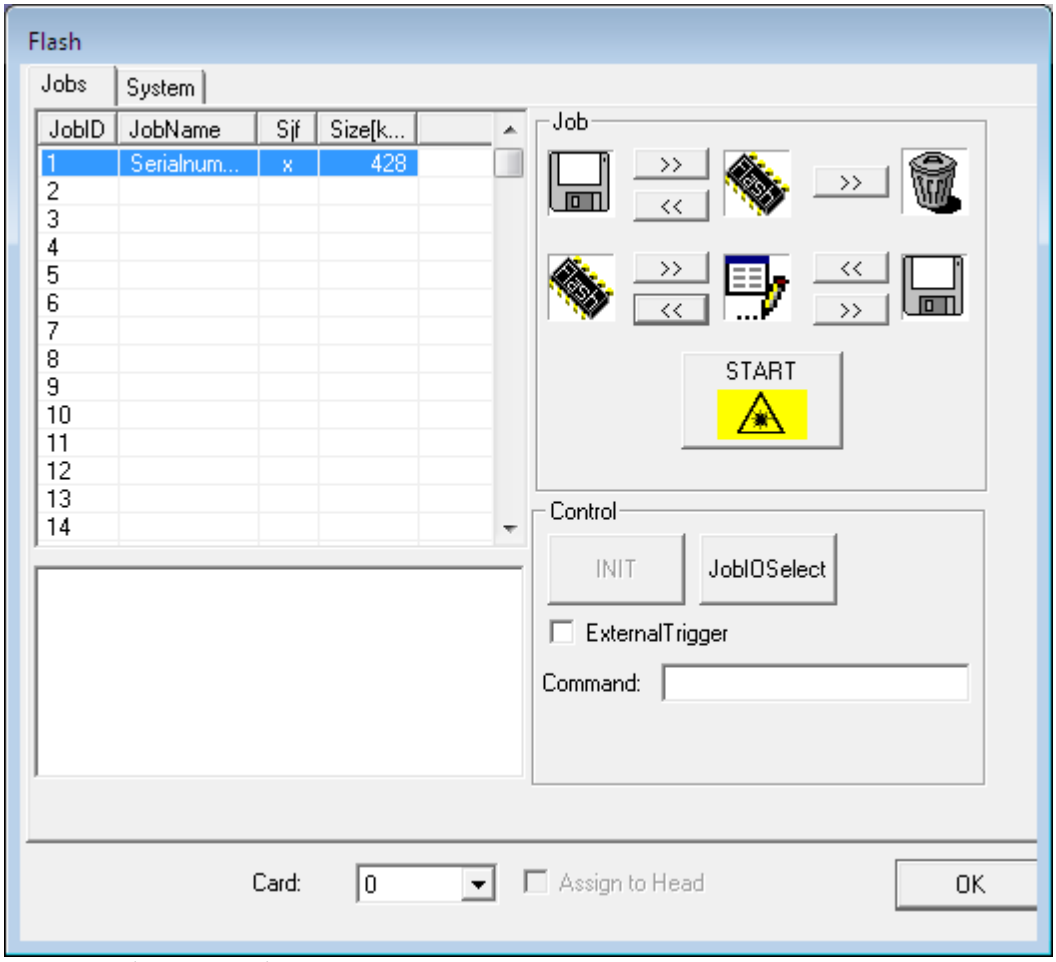


Figure 11.2: Flash Main Dialog

11.1.1 Preparation

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.

There are currently several limitations which must be taken into account when generating jobs:

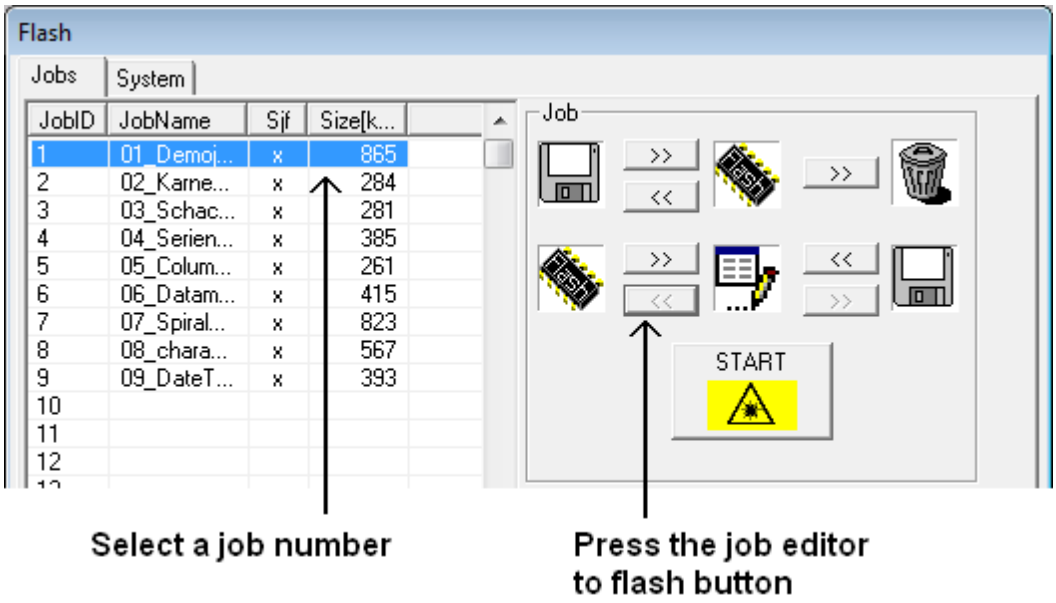
- Serial number entities with variable bar code are not supported
- Bitmaps are not supported.

↯ On the Flash the field correction and laser source (CO2,YAG,LEE,IPG...) information are global. Mixing up jobs generated with different settings in that respect can lead to unpredictable result.

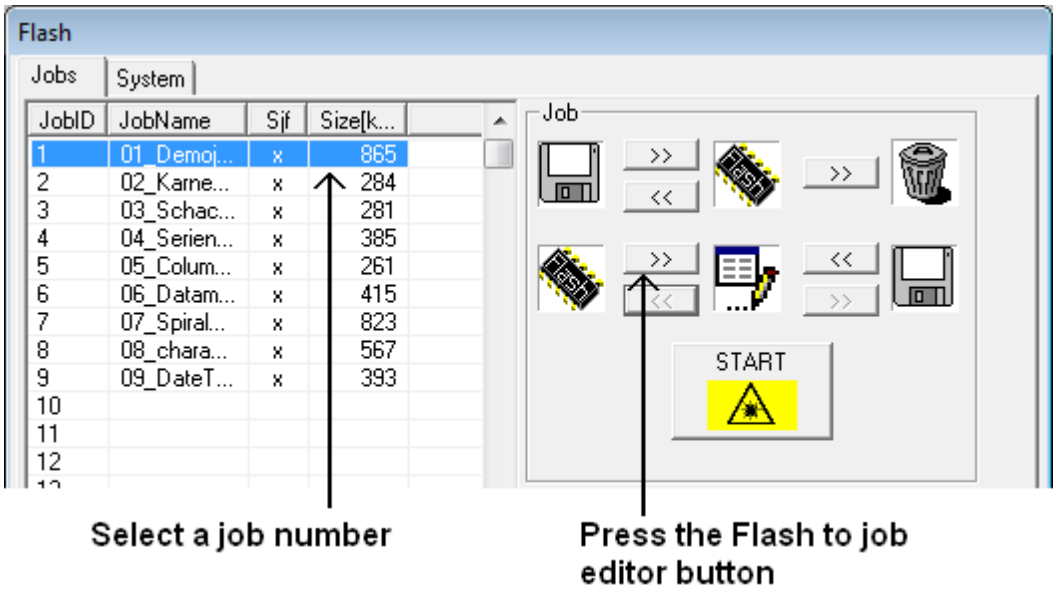
Each job can keep its own pen assignments. If serial numbers with text entities are generated the corresponding font tables are attached to the generated unf file and therefore they are unique to the job. After job preparation it should be save to disk.

11.1.2 Up/Download

Upload from editor:

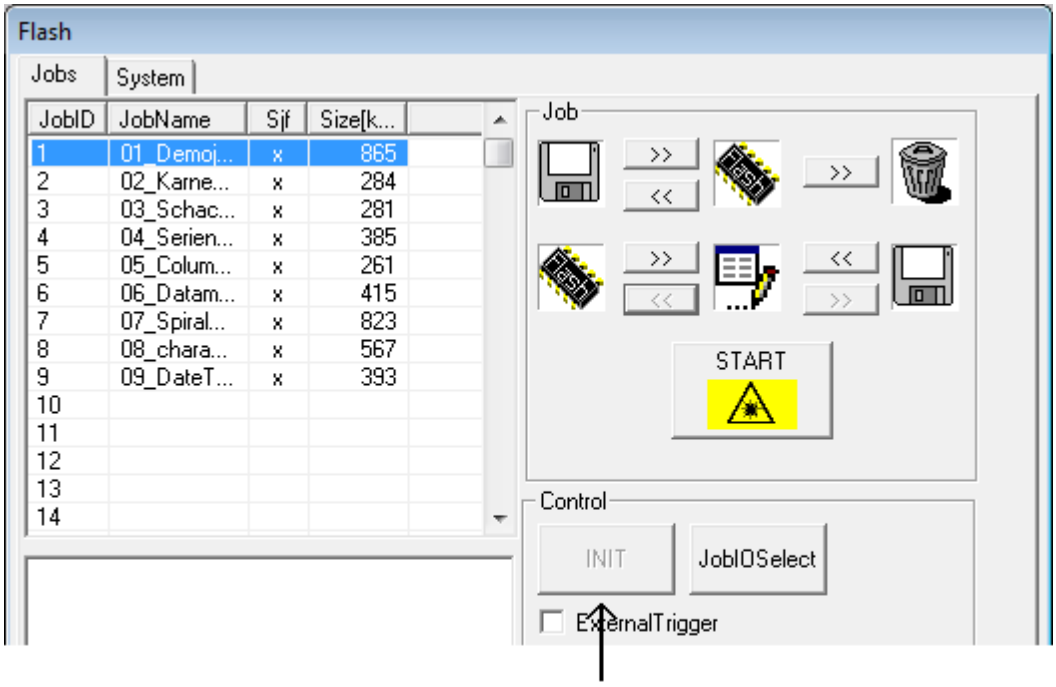


Download to editor:

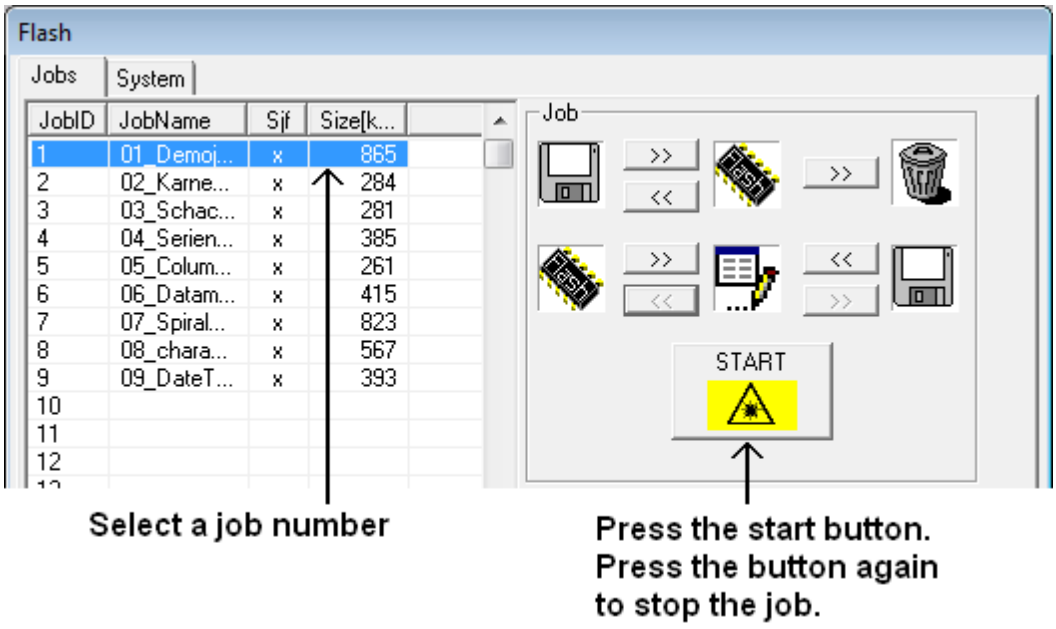


11.1.3 Execution

Initialisation of the Flash:



Start/Stop the job:



11.2 Memory layout

The Flash memory is separated into a system and data section. In the system section global flash information , flash programs, correction file and other initialisation data are stored. In the data section the jobs data are stored. The number of storable jobs is limited by the size of the job data section and the minimum memory allocation size which is 16 Kbytes. This leads to a maximum number of 3968 jobs. In the current release the number of storable jobs is limited to 256.

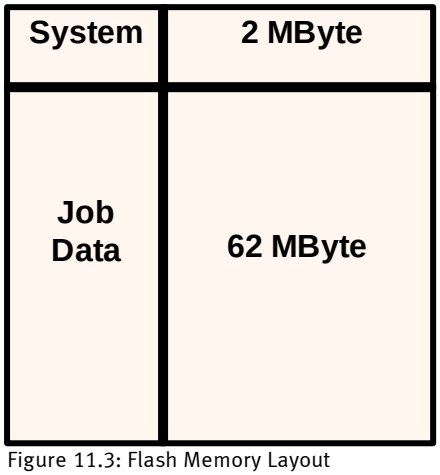


Figure 11.3: Flash Memory Layout

11.3 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.

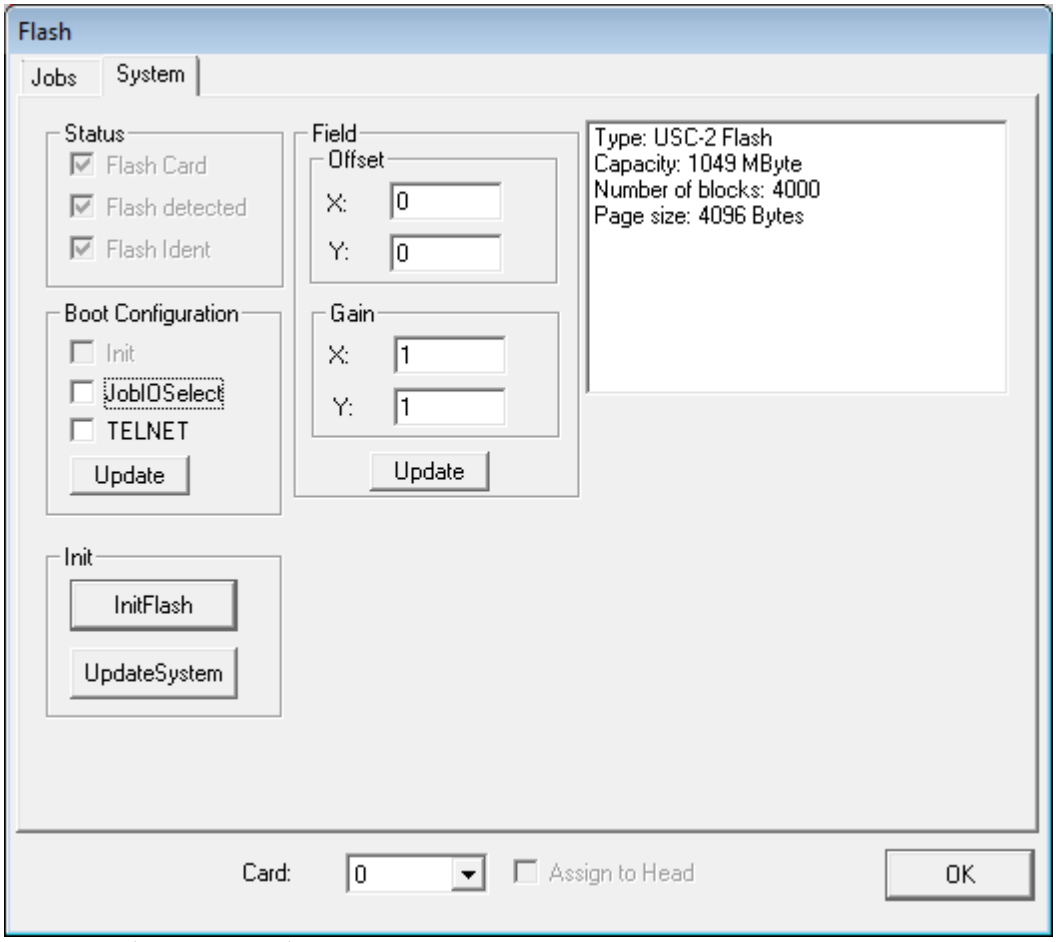


Figure 11.4: Flash System Dialog

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

Init:
Flash card starts and makes an auto initialisation.

JobIOSelect:
The Flash card starts in JOBIOSelect mode.

TELNET:
It is possible to send commands to the USC-2 via TELNET Port 23.

For all options to take effect it is required to press on the "Update" Button and then reconnect the USC-2 card.

Init

InitFlash:
Erases the complete flash. All data is lost.

UpdateSystem:
Updates the flash system block only. Job information is preserved.

12 Option Multihead

The SAM multi head module allows to control many 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.

- 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 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.

12.1 Multiple Heads with Multiple Cards

The SAM multi head modules allow to build up scanner applications with a simultaneous vector output to up to 4 scan heads.

Note: For this simultaneous output and the installation of more than one (up to 4) 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 fix which vector is sent to which head.

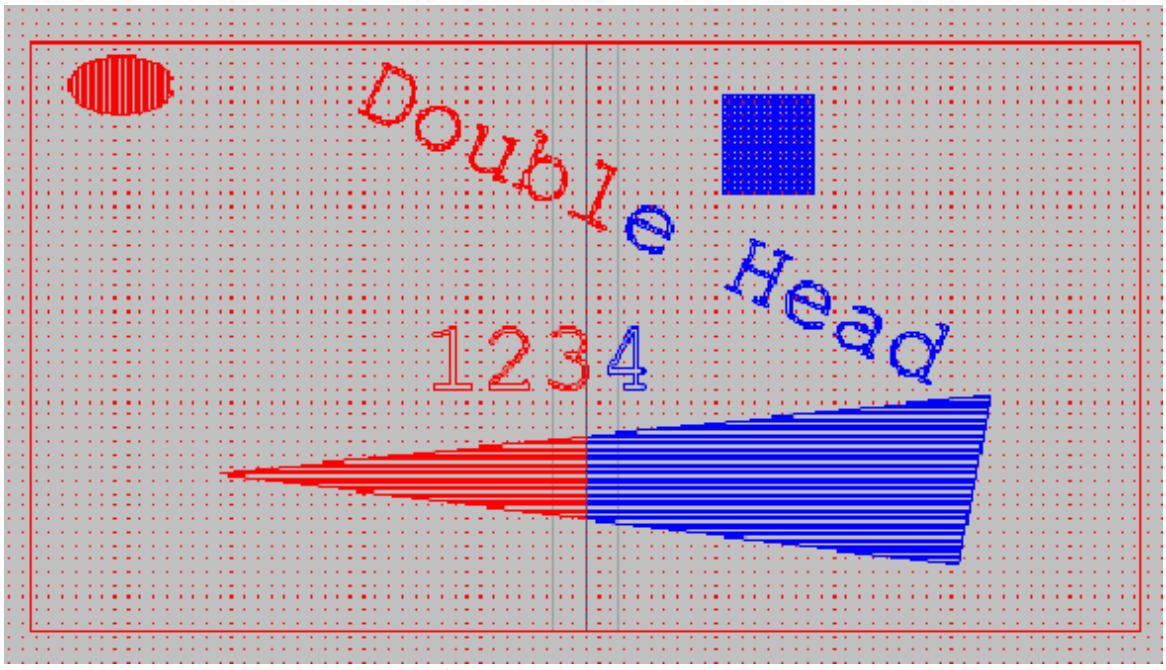


Figure 12.1: Job with two different working fields for two different scanheads

12.1.1 Installation

There are four steps to getting started to use multiple heads with multiple cards:

1. Activating software with a password containing the MultiHeadOpticModule 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

12.1.1.1 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 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.

12.1.1.2 Setup Tool

The settings for the software are saved in a *.sam file located in the folder SCAPSINSTALLDIR\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 SCAPSINSTALLDIR\tools. To do this it is necessary to close all SAM applications before.

Starting sc_setup and selecting menu point "HardwareSettings" shows following dialog:

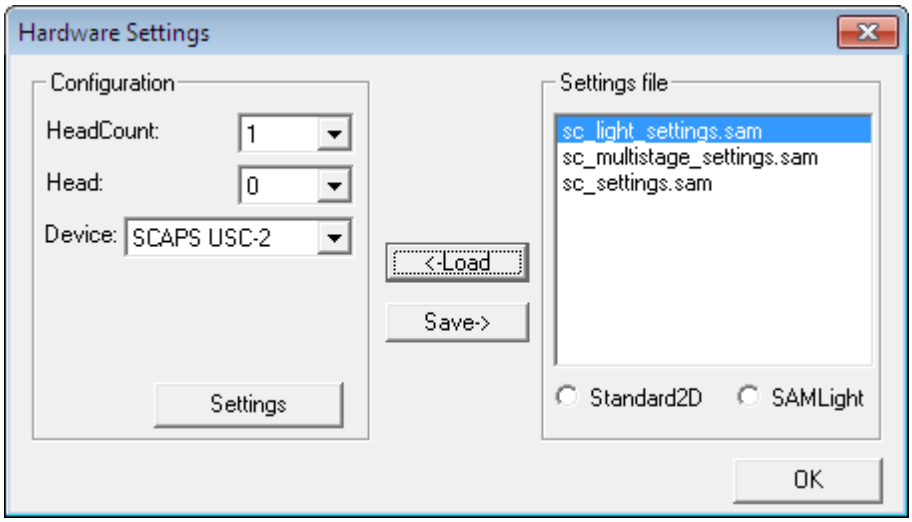


Figure 12.2: Hardware Settings Dialog

To change your settings, the following steps are required:

1. The software looks for all *.sam files located in SCAPSINSTALLDIR\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.

12.1.1.3 Optic Settings

Get this dialog by clicking on "Settings" at the Hardware Settings dialog. See chapter Setup Tool.

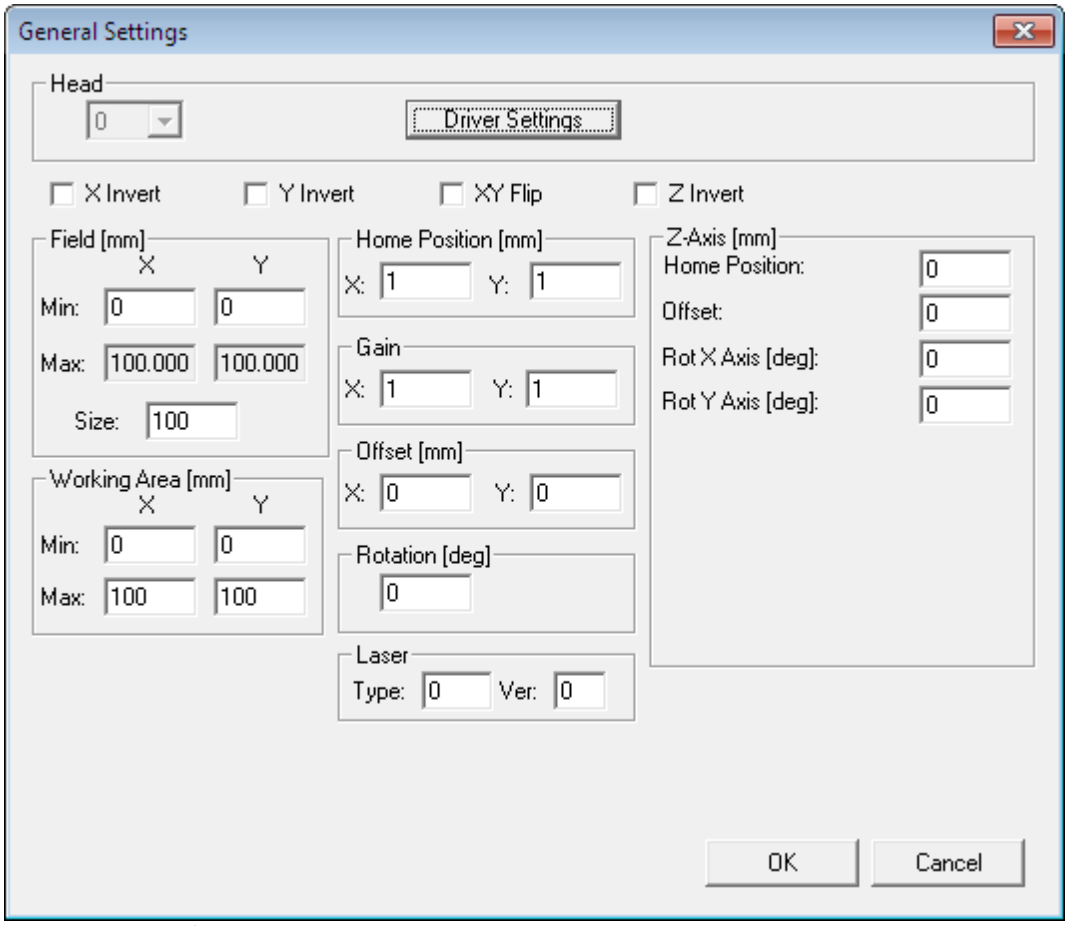


Figure 12.3: General Settings for Head 0

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:

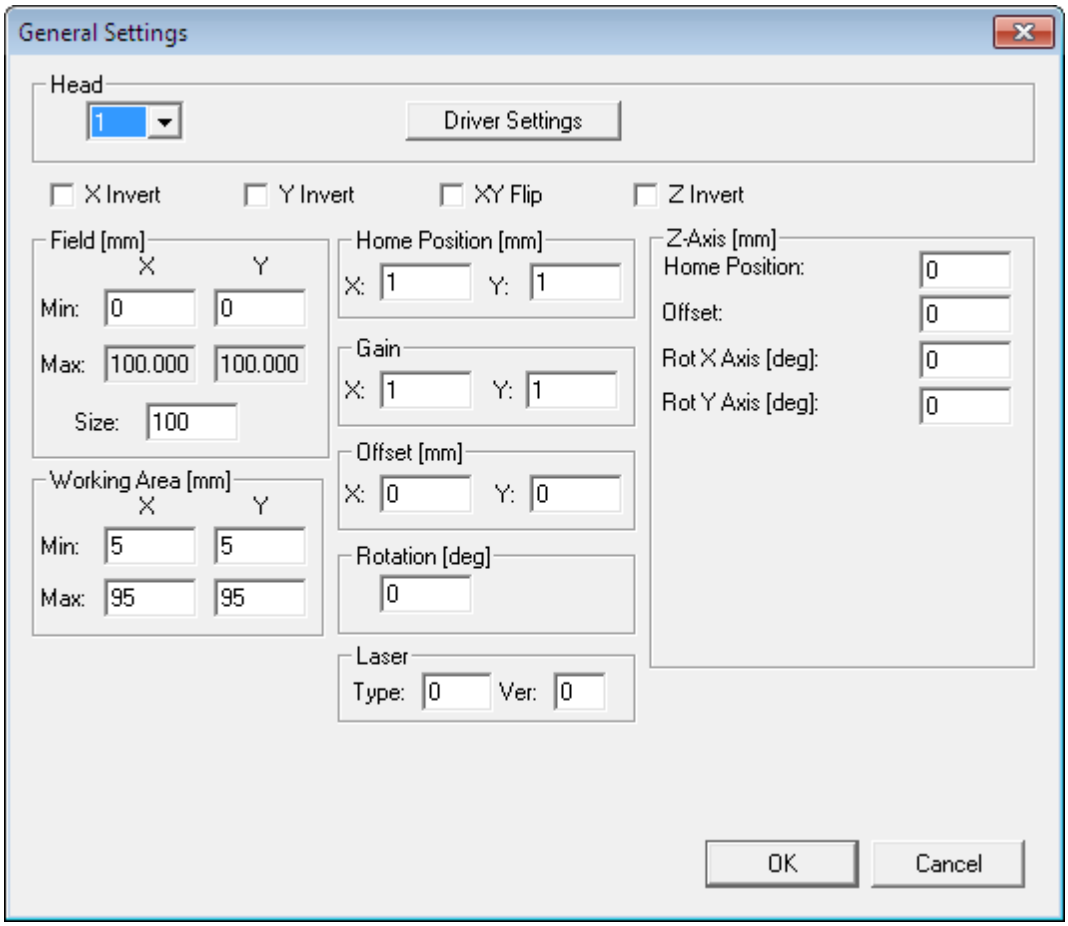


Figure 12.4: General Settings for Head 1

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 the Menu item "Settings->System->Optic".

12.1.1.4 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 and pressing ViewProperties or by a direct call to the "ViewProperties":

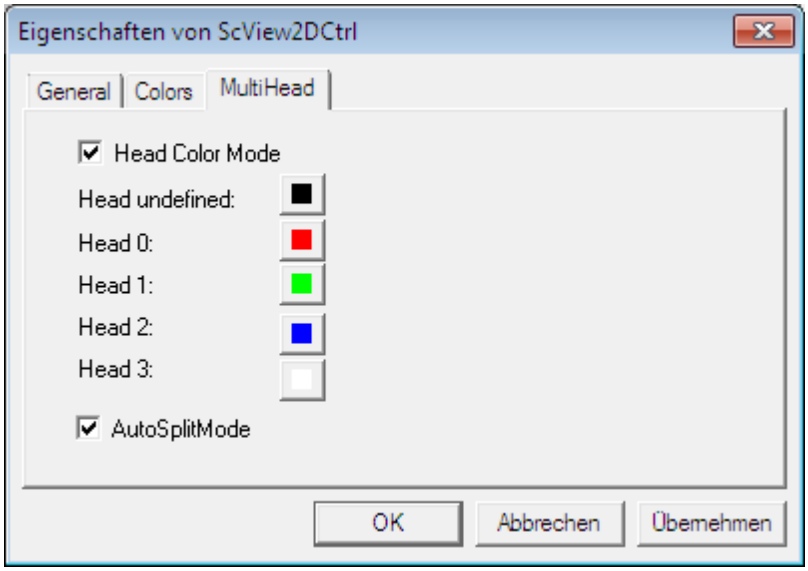


Figure 12.5: Properties of View2D

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 do not tell which pen is used but instead the colors mean which head is used for marking the entity.

12.1.2 Working with more Heads

When selecting the AutoSplit option, there is no significant difference in the behavior compared to an one head system. The OpticModuleControl shows a simulation device for each installed head.

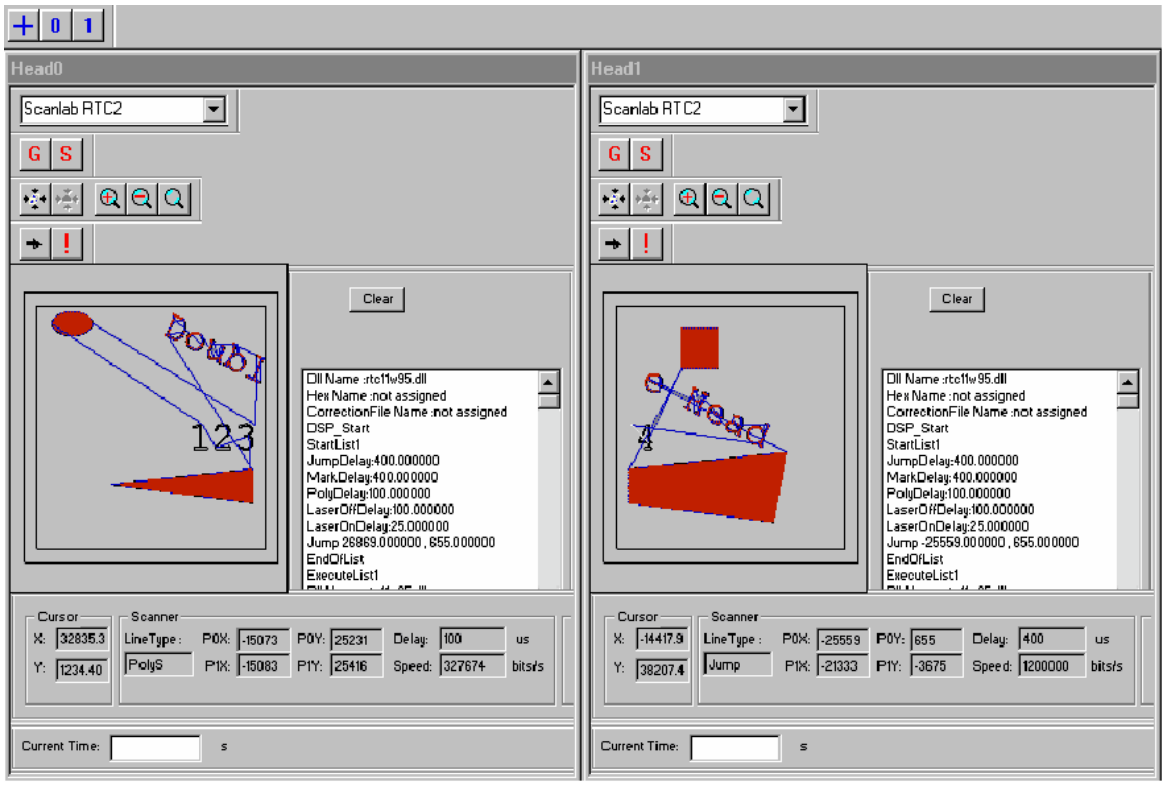
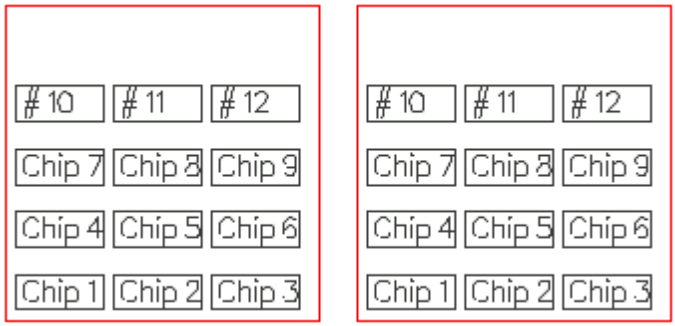


Figure 12.6: Mark Preview for two scanheads

12.2 Two Heads with One Card

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:



12.2.1 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 Settings->System->Optic.

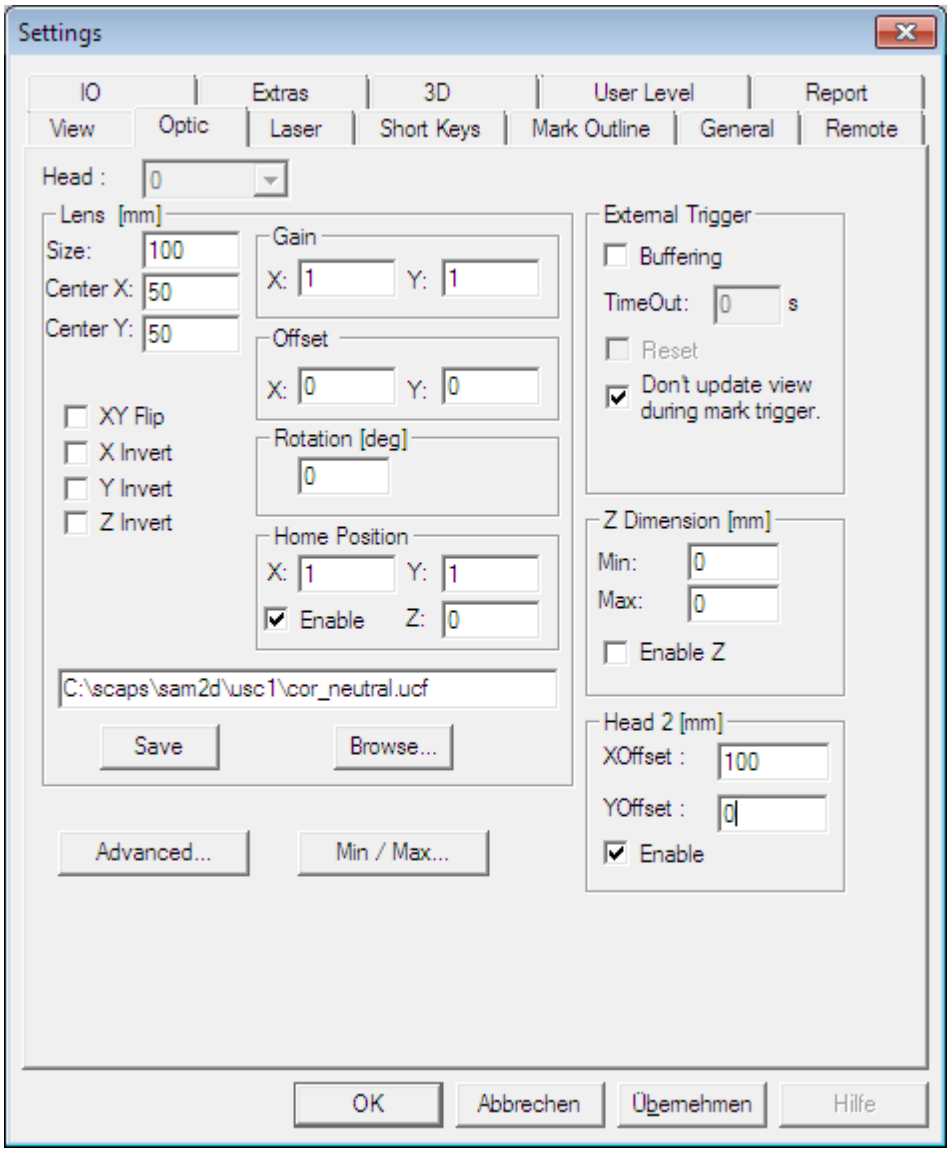


Figure 12.7: Optic Settings for Head 2

Step 1:
Click on "Advanced..." then on "Settings..." in the field "Correction". The correction file dialog appears. 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 scanhead. After doing this the second scanhead 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 scanhead which will only affect the View but not the scanner output.

12.2.2 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":

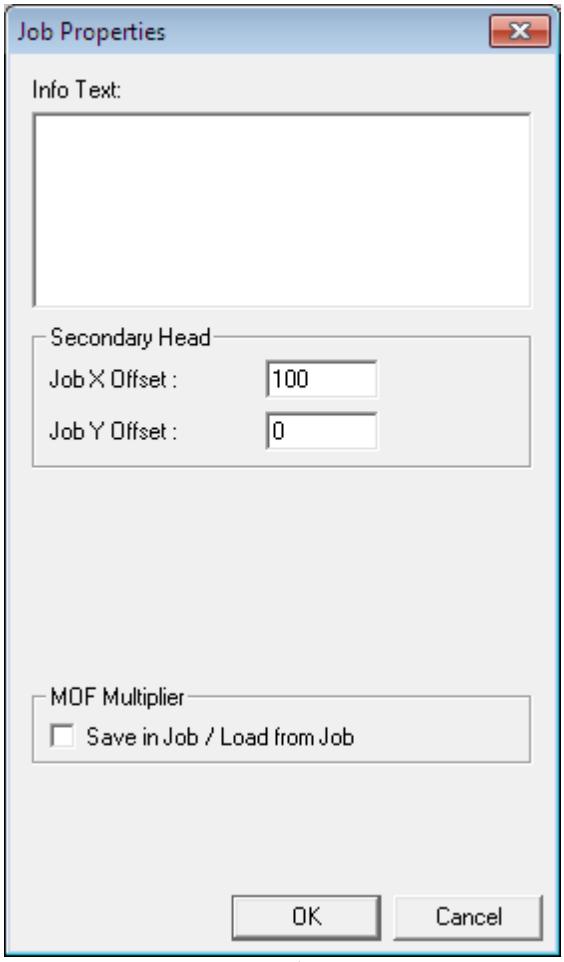
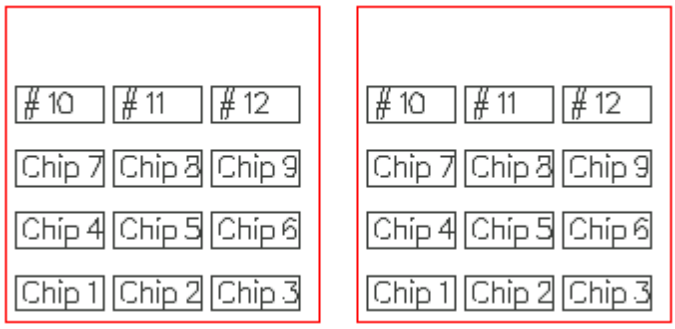


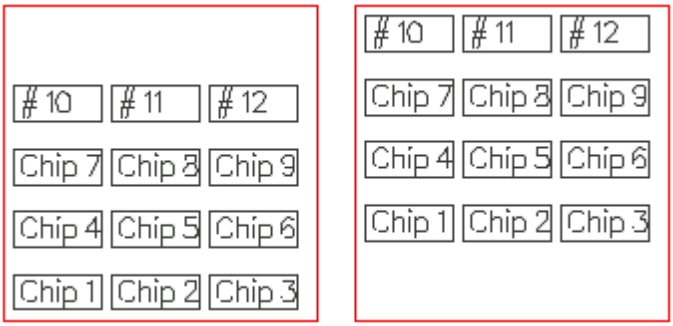
Figure 12.8: Job Properties Dialog

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.

12.2.3 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.

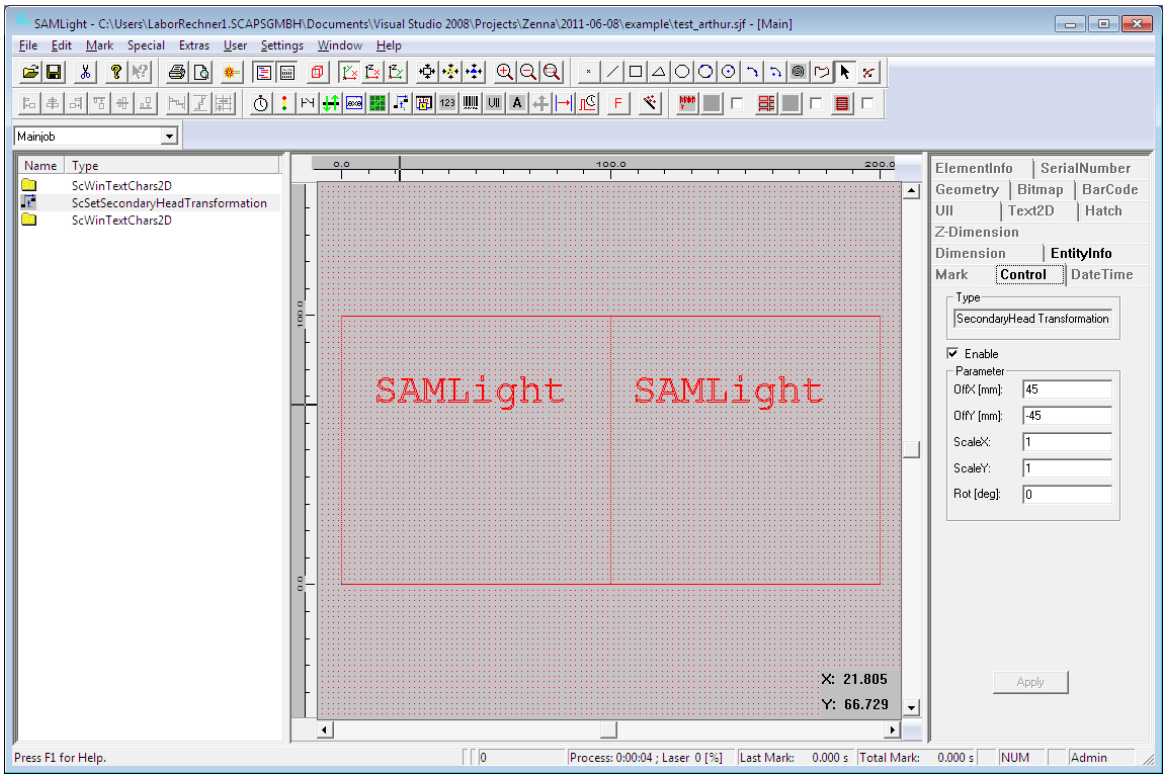


Figure 12.9: Secondary Head Offset

Click on the  button in the functionality object tool bar 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.

13 Option SAM3D

This chapter describes the 3D functionality of SAMLight. Before working in the 3D mode of SAMLight the checkbox "3DView" in the menu Settings -› System -› 3D has to be checked. Then click on OK and restart the software:

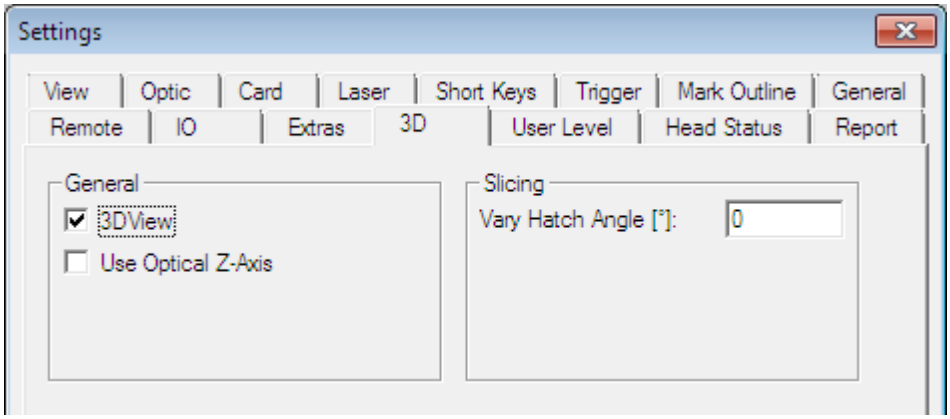


Figure 13.1: 3D Settings Dialog

13.1 Main Window

In the following the main window of SAM3D is shown. This window appears after software start.

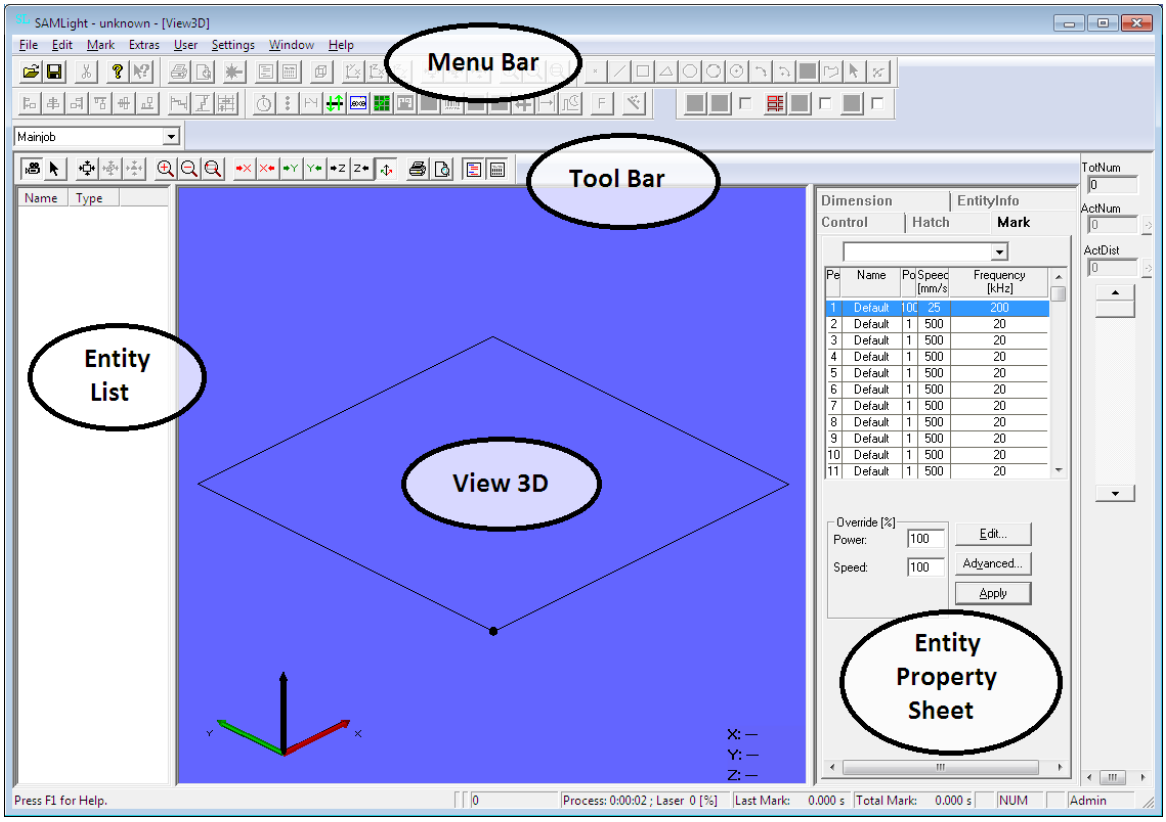


Figure 13.2: Main Window for 3D Mode

It is very similar to the main window of SAMLight. It consists of the Menu Bar, the Tool Bar, the Entity List, the View 3D and the Entity Property Sheet. The Menu Bar contains only the open and save file functions. The Tool Bar 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 black quad in the View 3D shows the working area in the X and Y directions.

13.2 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 or .cli file. To load a .s3d file click on the menu item File -› Load. To import a .stl or .cli file go to File -› Import... . Once a file has been loaded or imported the View3D shows the 3D object:

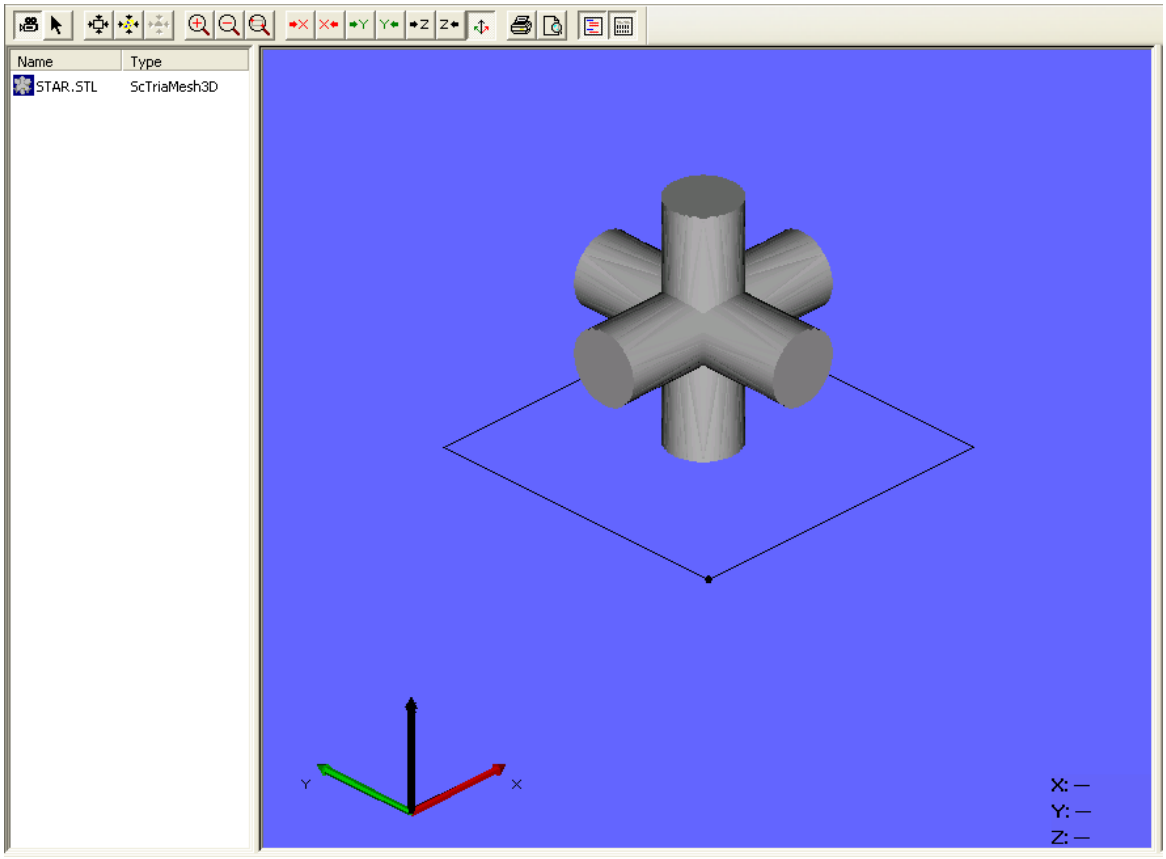


Figure 13.3: View 3D

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:

13.2.1 Tool Bar

The Tool Bar 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.

13.2.2 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 Tool Bar. 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 Tool Bar. 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.

13.2.3 View Properties

Click the right mouse button while the mouse pointer is inside the View 3D to open the View Properties. The following window will pop up:

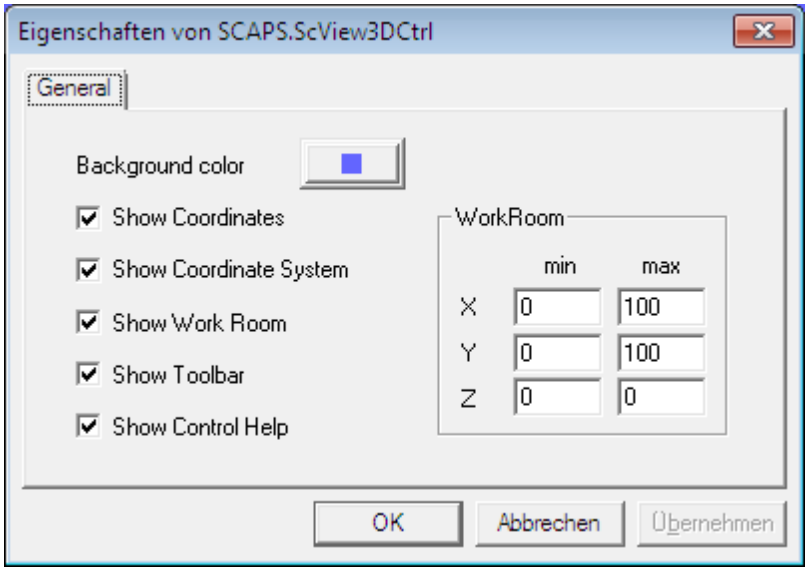
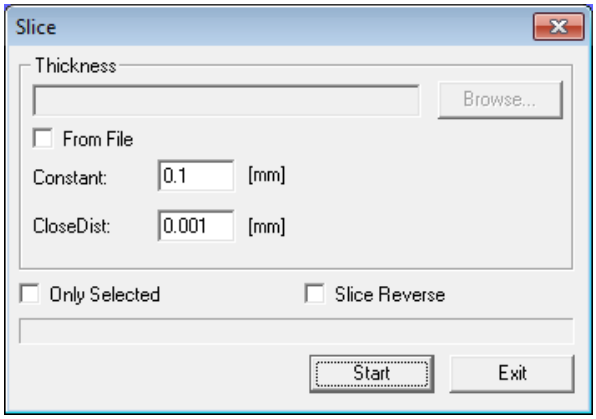


Figure 13.4: 3D View Properties Dialog

Here choose the background color, activate or deactivate the options and define a working room.

13.2.4 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 scanhead has to be set to the next z-position. You can choose between these two options by checking or unchecking the checkbox "Use Optical Z-Axis" in the menu Settings -› System -› 3D. To slice the object choose the menu item Edit -› Slice. The following dialog appears:



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.

Figure 13.5: Slicing Dialog

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.

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:

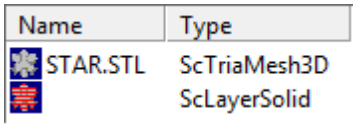
```
2
13
R
0.10;0.01 1
0.11 2
0.15 1
0.17 3
```

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 number 1 is assigned to all these ten slices. Following these three slices are created:

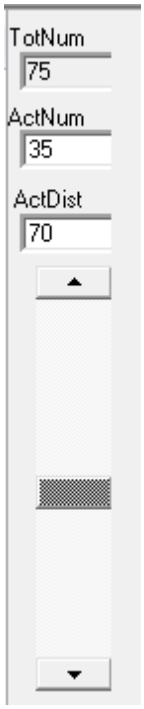
- a single slice from 0.10 to 0.11 (= thickness of 0.01) where pen 2 is assigned to
- a single slice from 0.11 to 0.15 (= thickness of 0.04) where pen 1 is assigned to
- a single slice from 0.15 to 0.17 (= thickness of 0.02) where pen 3 is assigned to

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.

13.2.5 Marking

By clicking on the mark icon  or by selecting the menu Mark -› Start the mark dialog is being opened:

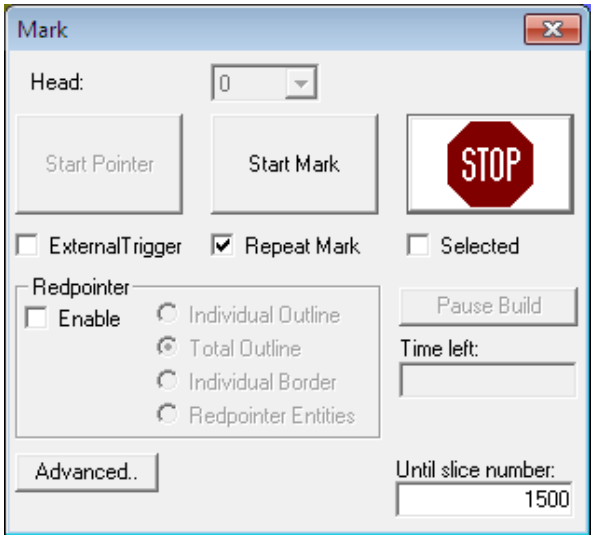


Figure 13.6: 3D Marking Dialog

Start Mark:
Starts the marking process.

Repeat Mark:
If checked all slices (layers) will be marked one after another. If not checked only the actually selected slice will be marked.

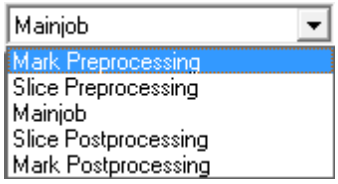
Until slice number:
The marking will stop if this slice number has been reached.

13.2.6 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 (User Interface -› Tool Bar).

For example

First setup a Mark Preprocessing:



In the window that opens create a Motion Control by clicking on the Motion Control Icon  .

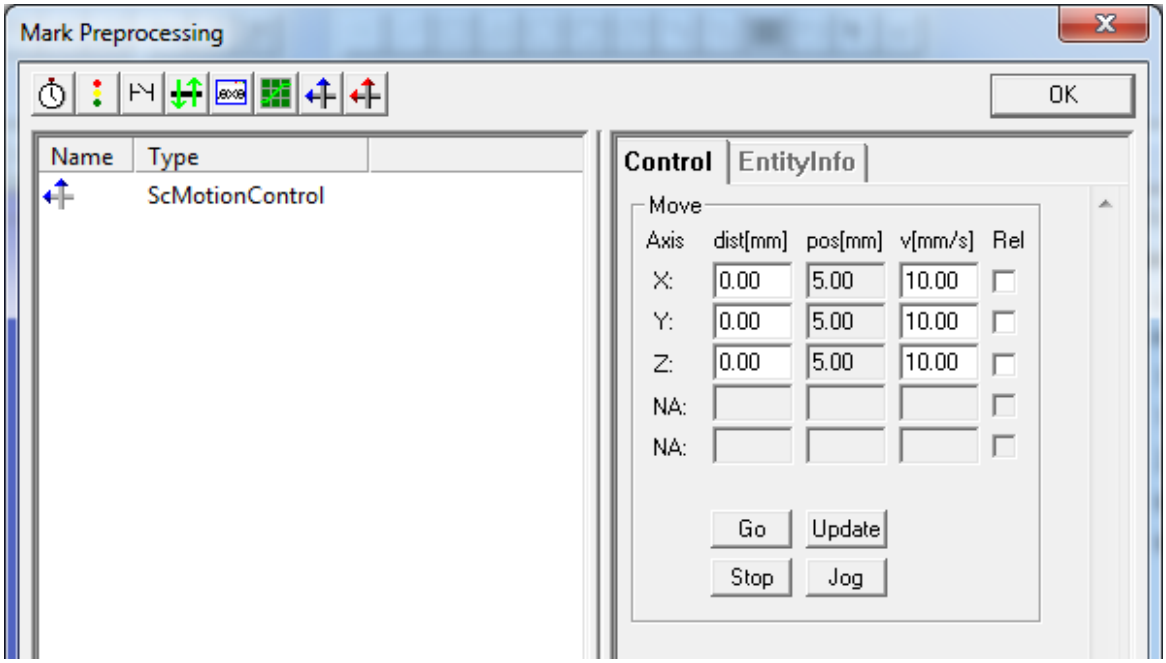
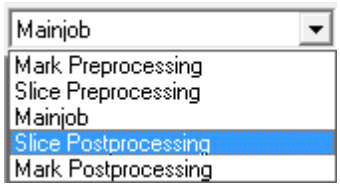


Figure 13.7: Mark Preprocessing Dialog

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 like it is shown below:



Then create a motion control entity in there:

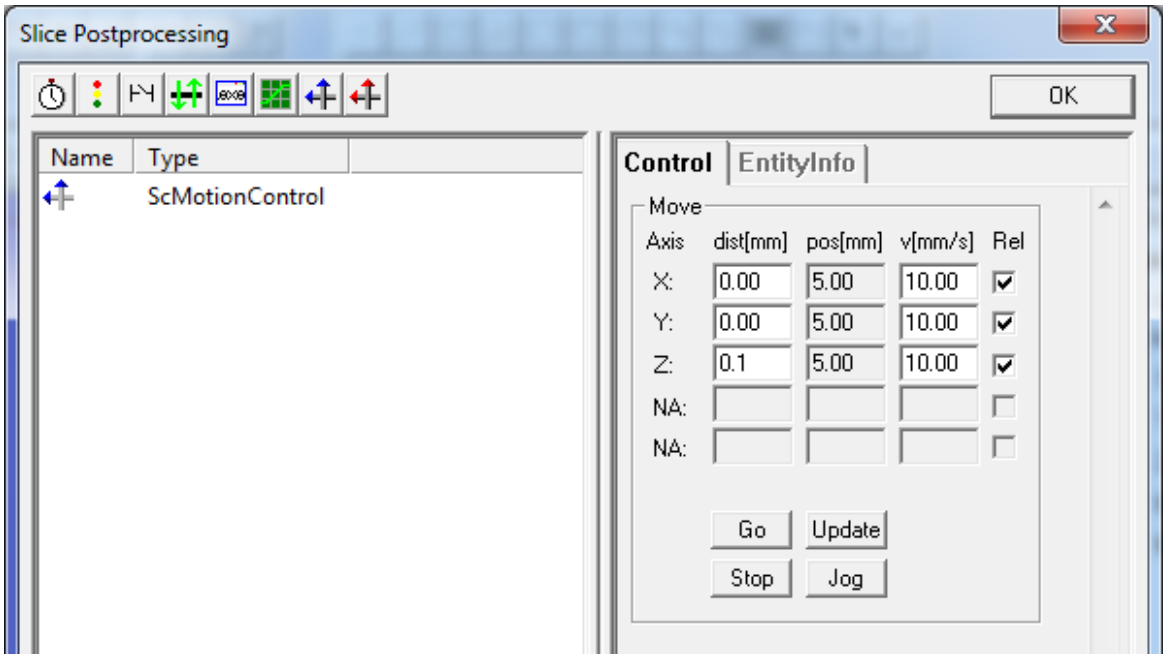


Figure 13.8: Slice Postprocessing Dialog

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.

13.3 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.

Show state specified by parameter Show	Value of the Show parameter
SW_HIDE - Hides the window and passes activation to another window.	0
SW_SHOWMINIMIZED - Activates the window and displays it as an icon.	2
SW_SHOWMAXIMIZED - Activates the window and displays it as a maximized window.	3
SW_SHOW - Activates the window and displays it in its current size and position.	5
SW_RESTORE - Activates and displays the window. If the window is minimized or maximized, Windows restores it to its original size and position.	9

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: **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].

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 Dongle User ID.

scComSAMLightClientCtrlLongValueTypeDongleUserNumber = 40
Reads the Dongle System ID.

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.

scComSAMLightClientCtrlDoubleValueTypeLastExpectedMarkTime = 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:

scComSAMLightClientCtrlStringValueTypeGetLastErrorMessageInput = 14
Puts the last Error Message into the string Name.

scComSAMLightClientCtrlStringValueTypeGetLastInfoMessageInput = 15
Puts the last Info Message into the string Name.

Function: **long ScSetStringValue(20000, string Name)**

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.

Function: **long ScGetStringValue(20000, string Name)**
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 ScSetDoubleValue(20000, double Value)**
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.

Function: **double ScGetDoubleValue(20000)**
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.

Index

- 2 -

- 3 -

- 8 -

- A -

- B -

- C -

- D -

- E -

- F -

- G -

- S -

- T -

- U -

- V -

- W -

- Z -