



Raith B.V.
De Dintel 27a
5684 PS Best
The Netherlands

Phone +31 499 336 880
Fax +31 449 336 899

www.raith.com

EBPG

Reference Manual

Document: B906505
Version: 0.9

CONFIDENTIAL



Managing Director
Erwin Mueller

Chamber of Commerce
Eindhoven 33286867

ABN AMRO N.V.
IBAN : NL28 ABNA 0247 4414 30
BIC : ABNANL2A

VAT Number
NL001751773B01

EBPG Reference Manual

Abstract

This document defines the usage of the EBPG 5000 Plus/5200/5150 electron beam lithography tool with Linux based operator control software.

Copyright © 2017 by Raith B.V. All rights reserved.

This software and its associated documentation contain confidential and proprietary information of which the Copyright and other intellectual property rights belongs to Raith B.V. unless otherwise indicated. The software and its documentation are furnished under a license agreement and may be used or copied only in accordance with the terms of the license agreement. No part of the software and documentation may be reproduced, transmitted or translated in any form or by any mean, electronic, mechanical, manual, or otherwise, without prior written permission of Raith B.V. or as expressly provided by the license agreement.

All reasonable steps have been taken to ensure that this publication is correct and complete, but should any user be in doubt about any detail, clarification may be sought from Raith B.V. or their accredited representatives. The information in this document is subject to change without notice and should not be construed as a commitment by Raith B.V.

Raith B.V. accepts no responsibility for any errors that appear in this document.

Document Number: B906505

Date: December 13, 2019

Issue: 0.9 draft

Author: various

Printed in the Netherlands. This document is originally written in English.



Corrections and changes to this manual should be forwarded by email to:
Lithography-manuals@Raith-litho.com, Giving full details of the changes required.

TABLE OF CONTENTS

Preface	1
For technical advice/service.....	1
TRAINING	1
1. Control Software.....	2
1.1 Linux configuration	2
1.2 Beams software structure	2
1.2.1 Global data	3
1.2.2 HILL SHELLY	4
1.2.3 COLL SHELLY.....	4
1.3 Beams environment and installation.....	4
1.4 User environment and installation	5
1.4.1 Local set up	7
1.4.2 Operator / supervisor control modes	7
1.4.3 Changing environment	8
1.5 Command line interface.....	9
1.5.1 Basic commands	9
1.5.2 Define commands.....	9
1.5.3 Action commands	9
1.5.4 Information commands	10
1.6 Graphical interfaces.....	10
1.7 Raith Linux Tools	11
2. Loading and Unloading Substrates	12
2.1 Fitting a substrate or mask-plate into a holder	12
2.1.1 Mask-plate	12
2.1.2 Substrate	13
2.2 Coarse alignment of substrate for direct write	14
2.2.1 Measuring the rotation	15
2.2.2 Adjusting the rotation	15
2.3 Holder loading/unloading in/from airlock	15
2.3.1 Vent the airlock:	15
2.3.2 Fit the holder in the loader (EBPG5000 Plus)	16
2.3.3 Fit the holder in the loader (EBPG5200)	18
2.3.4 Pump the airlock	20
2.4 Holder loading/unloading on/from stage.....	20
2.5 Holder initialisation.....	21
2.5.1 Holder parameters	21
2.5.2 Setting up the holder parameters	22
2.5.3 When and how to update a holder marker position.....	23
3. Machine Setup	25
3.1 Deflection: resolution and beam step size.....	25

3.2	Stage coordinates.....	29
3.2.1	Homing	29
3.2.2	Limit switches	29
3.2.3	Stage locking	29
3.2.4	Symbols	29
3.2.5	Stage distortion corrections	30
3.3	Z-stage.....	31
3.3.1	Z-stage homing.....	31
3.3.2	Z-stage Work levels.....	32
3.3.3	Z-stage movement.....	32
3.3.4	HGTZSTAGE parameter	33
3.4	Detectors	34
3.4.1	PM detectors.....	34
3.4.2	TEM detectors	35
3.4.3	SED detector	35
3.5	Final aperture controller.....	38
3.5.1	Commands to operate the automatic aperture selector	38
3.5.2	Initial set-up of the apertures	39
3.5.3	Normal operation	40
3.5.4	Check the status of the aperture controller	41
3.5.5	Check the parameters in the controller.....	41
3.5.6	Modify the parameters in the controller	42
3.5.7	Diagnostics of the aperture controller.....	44
3.5.8	Errors and status aperture controller	45
3.5.9	Disable the motors of the aperture controller	45
3.6	Imaging	46
3.6.1	SEM scan generator	46
3.6.2	Configuring SEM monitor parameters	46
3.6.3	pattern generator	48
3.7	Beam creation.....	48
3.7.1	Initial lens estimates	48
3.7.2	Coarse lens values and aperture alignment.....	49
3.7.3	Column alignment.....	50
3.7.4	Conjugate beam blanking	52
3.7.5	Final lens setting.....	53
3.7.6	Check column set-up	53
3.7.7	Set up of a spot range	53
3.7.8	Checking angular intensity	54
3.7.9	Beam diameter versus beam current	54
3.7.10	Defocussed beam.....	55
3.8	Settling time	55
3.8.1	Beam settling	55

3.9	Calibrations.....	57
3.10	Corrections	57
3.10.1	Height measurement	57
3.10.2	Heightoffset.....	58
3.10.3	Focus and stigmator table	60
3.10.4	Fine Focus Shift.....	60
3.10.5	Main field distortion table.....	60
3.10.6	Stage correction for magnetic effects	61
3.10.7	Stage correction for orthogonality and gain.....	61
3.10.8	Stage distortion table	61
4.	Marker Search.....	62
4.1	Introduction	62
4.2	Type of markers.....	62
4.3	Marker signal	64
4.4	Adjustable parameters for marker search	66
4.5	Marker routine.....	71
4.5.1	Adjust PM	73
4.5.2	Spiral search.....	77
4.5.3	Coarse search	79
4.5.4	Measure marker level	82
4.5.5	Measurement video noise level / Setting noise reduction filter	84
4.5.6	Fine search algorithm	87
4.6	Rectangle marker search	90
4.6.1	Define rectangle marker	90
4.6.2	Searching rectangle marker	91
4.6.3	Remove rectangle marker	91
4.7	Cross marker search	92
4.7.1	Define cross marker.....	92
4.7.2	Searching cross marker.....	93
4.7.3	Remove cross marker.....	93
4.7.4	Marker cross routine (HBAR and VBAR).....	93
4.8	Marker JOY.....	94
4.8.1	Define a JOY marker for manual SEM search	94
4.8.2	Creating a JOY marker width user own script	94
4.8.3	Searching JOY marker with SEM (without script)	94
4.8.4	Searching JOY marker with script	94
4.8.5	Remove joy marker.....	95
4.9	Image markers.....	96
4.9.1	Theory.....	96
4.9.2	Define an image marker	97
4.9.3	Searching an image marker	98
4.9.4	Image marker in CJOB	99

4.9.5	Removing an image marker	100
4.9.6	Searching Routine of image markers	100
4.9.7	Tips, Tricks and Error handling.....	104
4.10	Marker search with image grabbing	109
4.10.1	Image grabbing.....	109
4.10.2	Image correlation	111
4.10.3	User defined marker search	112
4.10.4	Short guide line for image marker search	114
4.11	Marker search templates	116
5.	Patterns	118
5.1	Writing strategy.....	118
5.2	GPF format	118
5.3	Exposure of GPF patterns	120
5.4	Estimate exposure time of GPF pattern	120
5.5	Data preparation	120
5.5.1	pctxt tool	121
5.5.2	penrose tool	122
5.5.3	Hill Shelly Library functions	123
5.5.4	gpfctx and gtxgpf tool	123
5.6	GPF tools.....	123
5.6.1	ppd tool	123
5.6.2	gpffreq.....	124
5.6.3	gpfcheck	125
5.6.4	gpftif and gpfraw	125
5.7	Field scaling.....	126
6.	BEAMS Exposure Procedures	129
6.1	Layouts	129
6.1.1	Simple array's	129
6.1.2	Exposure area's.....	129
6.1.3	Dose update	129
6.2	Alignment.....	129
7.	Job Procedures	130
7.1	Substrate mapping	130
7.1.1	Table positions.....	130
7.1.2	Using table positions.....	131
7.1.3	Three ways to create a substrate map (transformation).....	131
7.1.4	Applying a substrate map	133
7.1.5	Table positions and substrate maps.....	133
7.1.6	Practical applications	134
7.1.7	Command overview	137
7.2	Height mapping.....	137
7.2.1	Height measurement before and during writing	137

7.2.2	Marker deflection	140
7.3	Logging	140
8.	Advanced Operation.....	141
8.1	Creating dedicated procedures	141
8.2	Recovery procedures	141
8.3	User defined command procedures	142
8.4	Error handling	144
8.4.1	“C”-applications.....	144
8.4.2	Bash scripts	144
8.5	Machine Adjustment Procedures (MATPROC)	147
9.	Acceptance Tests.....	148
10.	Engineering Tools.....	149
10.1	Adjust_field	149
11.	User Tools	151
11.1	gds2clay.....	151
12.	Diagnostics	153
12.1	Electronics diagnostics	153
12.2	Stage controller (LSC)	153
12.2.1	Cstage	153
12.3	Spectrum analyser.....	153
12.3.1	Cspectrum	153
13.	Exception Conditions	154
13.1	Vacuum Failure.....	154
13.2	Mains Failure	154
14.	Miscellaneous	155
14.1	Mnemonics	155
14.2	Special environment variables.....	155
14.3	Debugging information	155
14.3.1	Debugging	155
14.3.2	Codes	156
14.3.3	Setting a debug value	156
14.3.4	Example of debug output.....	118
14.3.5	Cdebug	119
14.3.6	Create a logfile with debug information	123
15.	Installation Master Control Computer	124
15.1	Hardware requirements	124
15.2	Current configuration	124
15.3	Software requirements	125
15.4	RHEL Installation	125
15.5	BEAMS500P software configuration	126
15.6	Additional software configuration.....	127
15.7	Template files	128

16.	Troubleshooting	129
17.	References & Revision history	130
17.1	References	130
17.2	Revision History	130

PREFACE

The purpose of this manual is to provide comprehensive information to enable an operator to use an EBPG5000 Plus, EBPG5150 and EBPG5200 lithography system in the most efficient manner.

This manual only covers the “EBPG5000 Plus”, “EBPG5000 PlusES”, EBPG5150 and EBPG5200 tools, which have Linux-based control software (BEAMS V09_xx).

This document is part of a full set of documentation available for users and supplied during installation. The manuals are available in PDF form, generally being supplied on CD. It should be used in conjunction with the following manuals, as appropriate:

- “EBPG5000 Plus Command set manual (BEAMS)”.
- “Pre-installation Guide EBPG5000 Plus and EBPG5000 PlusES” (B900948 and 893291)
- “Linux workstation -system managers guide” (893381),

For technical advice/service

contact the local Raith service engineer

Service: +31 (0)499 336888

Fax: +31 (0)499 336899

Email: TSC.Helpdesk@raith-litho.com

TRAINING

Only personnel trained by Raith engineers should carry out service operations. Generally, only Raith engineers will carry out service operations. However, suitably trained customers may carry out a limited set of service and maintenance operations, based on the guidelines given here.

1. CONTROL SOFTWARE

The e-beam tool is controlled from a PC Workstation, operating under the Linux operating system. The Beamwriter EBPG Application Machine Software (BEAMS) is installed in a separate account with account name beams. The user operates the tool from a user account. One of these accounts has administrator rights, which implies that every action with the tool can be executed from this account. All other user accounts have restricted rights, which implies that only those actions can be executed which are required to do “exposure jobs”. By default factory setup uses the account name pg for the user account with administrator rights. No other user account are created as this is considered to a customer task. Facilities to create restricted user accounts are available. Note that this can be done as root only.

The passwords of the beams software account and the administrator user account are set at the factory to a default (beamwriter). Raith advises the customer not to change the passwords to provide service engineers readily access to the system, or, in case the user wishes to change the password, make sure the new passwords (also for the root account!) are known to the Raith engineer.

In this chapter the factory configuration is described. Deviations from the standard set-up are possible but are strongly discouraged. In addition, making changes to the configuration will require advanced skills from the local engineers. Raith made their best effort to make the BEAMS control software flexible and yet convenient to use. In case you experience inconveniences please let us know, so we can discuss and find a solution or alternative for your problem.

Pattern preparation computer

Patterns are generated from a separate ‘pattern preparation computer’, or ‘data prep.’ computer, that is linked to the e-beam control computer by Ethernet for file transfer. This pattern preparation computer generates the fractured data file (GPF file format) that is used by the EBPG5000 Plus, using fracturing software such as CATS (trademark of Synopsys) or BEAMER (trademark of Genisys).

1.1 Linux configuration

Usually the PC Workstation has been configured at the factory. The Linux installation is from Redhat RHEL 5. Details about the installation are to be found elsewhere in this manual.

1.2 Beams software structure

The BEAMS control software is a comprehensive system, controlling all major automated subsystems. These include stage movement, beam formation, deflection, pattern generation, monitoring and logging as well as Job Control File execution. This system uses a range of instructions with an intuitive structure, which simplifies job preparation.

In Figure 10 the structure of BEAMS is shown. Horizontally, the commands are shown, grouped into verbs and options. Vertically the various layers are indicated. Commands are transferred from the user command interface layer (COLL SHELLY), through a library of C-functions (HILL SHELLY) to the subsystems.

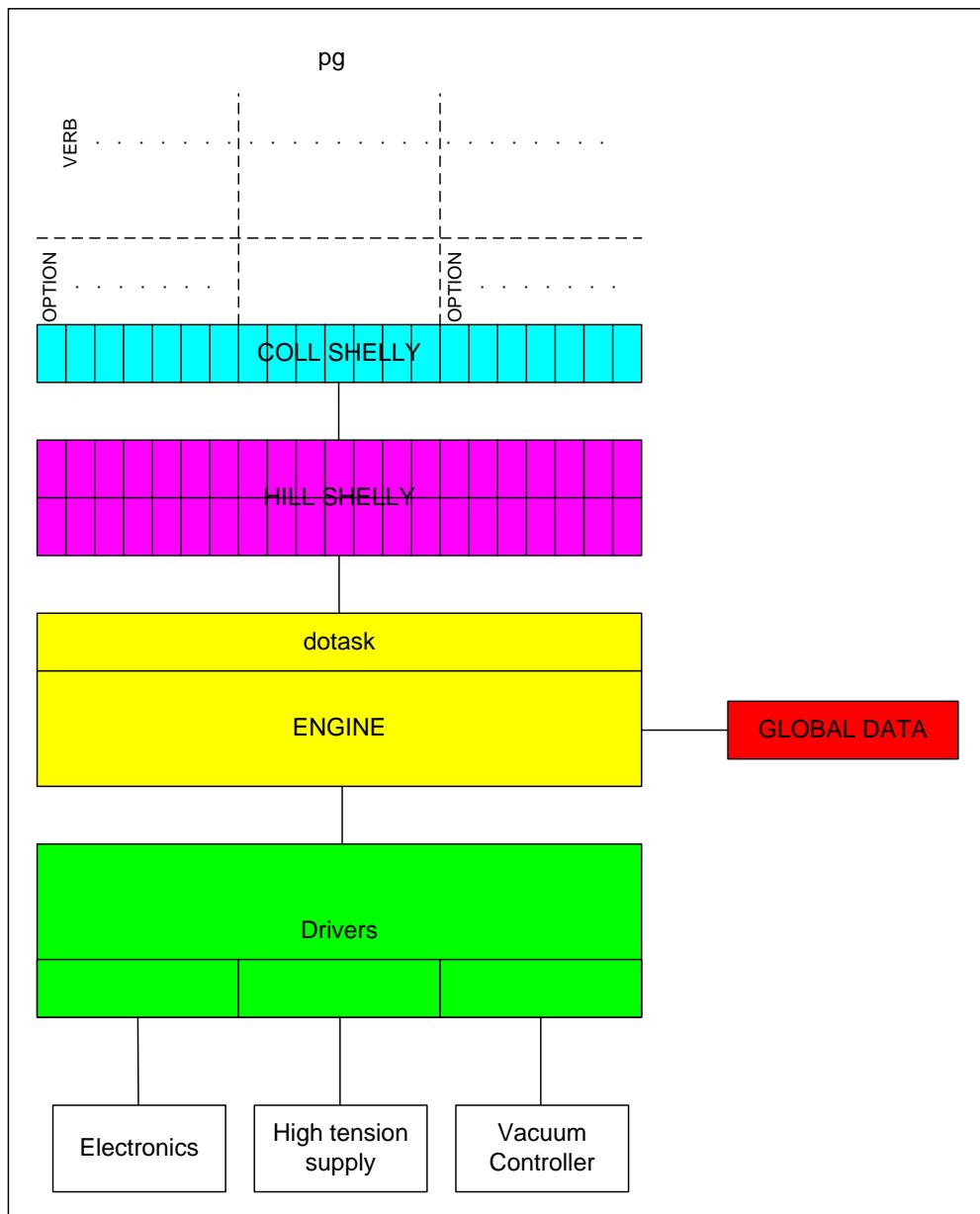


Figure 1 - BEAMS software structure Block Diagram

1.2.1 Global data

The status of the machine is stored in global data. The global data has been separated from the code and is stored in permanent memory.

Only after a reboot of the workstation must the global data be re-installed and subsequently initialized with the command “pg ebgc coldstart” or “pg ebgc hotstart” command from the file <machine name>_globaldata.

When a command crashes (which should not occur in the first place), the global data will not be destroyed. The global data can be accessed by more than one process at the time. However, at any time only one process should be allowed to control the EBG. All other processes are allowed to interrogate the system only. A mechanism to control the operator/supervisor mechanism is provided.

1.2.2 HILL SHELLY

The HILL SHELLY layer consists of a library of C-functions, which is the basic interface to control the machine. With the HILL SHELLY library, it is possible to develop custom made commands / job procedures or applications. All the standard BEAMS commands are developed with this library as well as many of the acceptance tests, engineering and user tools, which are included in the BEAMS distribution (see chapter 10). Applications can vary from advanced marker search procedures, statistics measurements. Detailed information about creating custom made procedures can be found in chapter 8.

For a description of all the functions see the BEAMS LIBRARY REFERENCE MANUAL [2]

1.2.3 COLL SHELLY

All the BEAMS commands pg <verb> are described in detail in the BEAMS REFERENCE MANUAL [1]. The BEAMS command pg <verbs> has been built using the HILL SHELLY library. Each BEAMS command verb/option corresponds with a function in the HILL SHELLY library.

Each command fires up the program pg, executes the appropriate software parts depending on the verb and optional additional parameters, and terminates when it is finished returning an error status to the operating system. Usually the command (name of the verb) is the action undertaken.

1.3 Beams environment and installation

The software distribution is a full package and contains.

- Command software
- Installation procedures
- Acceptance test procedures
- Job procedures
- Various engineering and user tools
- Documentation
- Software for subsystems (firmware)

The BEAMS software is installed in the beams account. In this account at the top level there is a script "beams" which can be used by the customer for installation of a release. Executing the script with the -h option provides the customer a help with the description of the other options provided:

```
> beams -h
```

The command

```
> beams -i <release version>
```

will install a release version. It is assumed that the gzipped tar file (xxx.tar.gz) is already uploaded to the login directory of the beams account.

e.g.

```
> beams -i y09_03e
```

will install a release y09_03e from the corresponding tar file. In case this release was already installed, the original release directory name will be renamed with a timestamp in the name. As long this directory is not deleted, it is always possible to revert back to the original release.

The BEAMS control software is available as a zipped tar file via the ftp server (ftp.Raith.nl) or on a CD on request from the Helpdesk at the Technical Support Centre at Best.

In case you have an external connection to the internet you can obtain and install a release directly from the Raith ftp server:

```
> beams -gi <release version>
```

Various environment variables are defined to point to specific BEAMS areas:

BEAMS=/home/beams

BEAMS_VERSION=y09_03e

BEAMS_RELEASE=y09_03e

BEAMS_PATH=/home/beams/y09_03e

BEAMS_ARCHIVE=/home/beams/y09_03e/archive

BEAMS_SETUP=/home/beams/y09_03e/setup

BEAMS_INSTALL=/home/beams/y09_03e/install

BEAMS_TFTPBOOT=/home/beams/y09_03e/tftpboot

BEAMS_FIRMWARE=/home/beams/y09_03e/firmware

BEAMS_INCLUDE=/home/beams/y09_03e/include

BEAMS_LIB=/home/beams/y09_03e/lib

BEAMS_MAN=/home/beams/y09_03e/man

BEAMS_DOC=/home/beams/y09_03e/doc

BEAMS_MATPROC=/home/beams/y09_03e/matproc

BEAMS_ACC=/home/beams/y09_03e/acc

BEAMS_ACC_PATTERNS=/home/beams/y09_03e/acc/patterns

BEAMS_SCRIPTS=/home/beams/y09_03e/scripts

BEAMS_GPF=/home/beams/y09_03e/gpf

BEAMS_JMAN=/home/beams/y09_03e/jman

BEAMS_USR=/home/beams/y09_03e/usr

BEAMS_ENG=/home/beams/y09_03e/eng

BEAMS_DIAG=/home/beams/y09_03e/diag

BEAMS_CJOB=/home/beams/y09_03e/cjob

BEAMS_CVIEW=/home/beams/y09_03e/cview

1.4 User environment and installation

The user account usually has the name pg. The root of the account contains a standard set of directories. Some of these belong to the “user” pg, others exists for holding machine specific data. Apart from the machine specific script pg_local, there should NOT reside any other files in this

directory.

user specific directories are:

users	user specific “nested” users / projects
bin	user specific binaries
scripts	user specific scripts
patterns	user specific patterns
images	user specific images
jobs	user specific job scripts
log	user specific log files

machine specific directories are:

archive	holds the archive files
data	holds data files
errlog	holds various error logging files
history	holds history files

Note: The directory Desktop is Linux specific.

To control the machine, BEAMS uses various machine specific environment variables:

PG_MACHID=B001

PG_IP_PLA=192.168.2.202

PG_IP_EHT=192.168.2.203

PG_IP_CNC=192.168.2.204

PG_IP_MOXA=192.168.2.205

PG_IP_DVM=192.168.2.206

PG_IP_VC=192.168.2.207

PG_IP_ZDRV=192.168.2.208

PG_IP_TMPM=192.168.2.209

PG_SERIAL_DVM=/dev/ttyS0

PG_PROMPT_OPTION=\[033[0;39m\]

PG_ARCHIVE=/home/pg/archive

PG_JOYSTICK=/home/pg/joystick

PG_ERRLOG=/home/pg/errlog

PG_DATA=/home/pg/data

PG_ACCDATA=/home/pg/data

```

PG_USER=SUPER
PG_USERID=pg
PG_USERPATH=/home/pg

PG_USERS=/home/pg/users
PG_PATTERNS=/home/pg/patterns
PG_PATTERNS_PATH=.:./home/pg/patterns:/home/beams/y09_03e/acc/patterns
PG_IMAGES=/home/pg/images
PG_IMAGES_PATH=/home/pg/images
PG_BIN=/home/pg/bin
PG_SCRIPTS=/home/pg/scripts
PG_JOBS=/home/pg/jobs
PG_LOG=/home/pg/log

```

1.4.1 Local set up

The pg_local script contains various machine and site specific configurations. This implies that this script will never be touched when a new BEAMS release is installed. In case a particular change is required for a new BEAMS release, the customer will be instructed to do so by himself. Some settings can be changed to alter the behaviour of specific parts of the software. This will be indicated in this manual where the specific parts are described. At the end of this script the customer can add more site specific commands.

The script is executed from .bashrc. In case there is NO interactive terminal available, pg_local will immediately return (Note that pg_local must be sourced !):

```

#Execute ~/pg_local
if [ -f ~/pg_local ]; then
    . ~/pg_local
fi

```

Installing the environment will display the following lines at the terminal screen

installing beams environment

installing pg environment

installing aliases

installing operator environment

Installing project pg environment

1.4.2 Operator / supervisor control modes

The operator / supervisor mechanism attempts to prevent control of the system from two terminals at the same time. This could destroy a job which is executing: e.g. selecting another pattern while an exposure job is running from another process (terminal) will destroy the executing job. Obviously the system is protected at the lowest level against mechanical damage, e.g. against moving the stage and simultaneously trying to unload / load a holder.

In case of supervisor control, the process can only interrogate the system, i.e. request for parameters en other information. All processes have at least supervisor control and is noticeable by the fact that the prompt is normally displayed. In case of an operator process, the process has full control of the system. This can be observed by the fact the terminal prompt is displayed in bold.

For a process, to acquire operator this is requested from a server (daemon) process is running to keep track of all terminals which have operator or supervisor control. Each login will undertake this action via the pg_local script. In particular a WARNING will be given when already an OPERATOR process is active, as you might be interfering with someone else

Other process with operator control

***** BE CAREFUL *****

Obviously, this mechanism is NOT full proof; it just prevents users to make mistakes by accident.

A slightly more rigorous protection is available. For this the line in pg_local:

```
#export PG_OPERCONTROL=strict
```

must be uncommented. In this situation, the user, which will become the first process with operator control, will receive also a PIN CODE. This code must be used to create the next terminal with operator control. This mechanism aims to ty all operator processes to a single person.

The commands (aliases to \$BEAMS_SCRIPTS/pg_environment script) to obtain / change the control mode are:

```
> oper <optional pincode>
> super
```

Example:

```
[pg@reagan service]service> oper
```

Other process with operator control

***** BE CAREFUL *****

```
[pg@reagan service]service> super
```

In case you do not remember the pin code, just issue the oper command in a terminal which has already operator control, and the pin code will be displayed again.

1.4.3 Changing environment

Although not mandatory, the EBPG is in almost all cases operated from a single account. Even in the case of multiple accounts, it is useful to split patterns, scripts, jobs, logging data into various users / projects. At this stage the following commands (aliases to \$BEAMS_SCRIPTS/change_environment script) are available:

- ce changes environment to another user at the same level
- ced changes environment to one level down

Note that it is allowed to specify multiple users / projects on the command line to indicate the hierarchy.

Executing one of these commands will modify the users specific environment variables. Note that the pattern are search for in the environment variable PG_PATTERNS_PATH. This functions similar to the Linux PATH environment variable, but just for patterns. The final directory to be searched for will be the \$BEAMS_ACC_PATTERNS directory for acceptance patterns.

```
[pg@reagan pipo]pip0> echo $PG_PATTERNS_PATH
.:~/home/pg/users/service/users/pipo/patterns:/home/pg/users/service/patterns:/home/pg/patterns:/home/beams/y09_03e/acc/patterns
```

Note that the user pg is a special user, i.e. the top user

Examples:

```
[pg@reagan pg]~> ced service pipo
Installing project pipo environment
[pg@reagan pipo]pip0> ce pg
Installing project pg environment
[pg@reagan pg]~>
```

1.5 Command line interface

The command line interface (COLL SHELLY) is the main interface to control the machine. All Beams commands have the following general structure:

```
> pg <verb> <keyword> /<qualifiers> <parameters> # Comment
```

Except for the <verb>, all components are optional, depending on the specific <verb>. Comment is always optional. At least one space or tab is required between the components. Additional spaces and tabs can be used between components to improve legibility. Long command lines can be continued on the next line by putting a “\“ sign at the end of the current line. Usually, the <verb> indicates the action.

The usage of the BEAMS commands is independent of the shell. By default bash is the standard shell installed by Redhat.

1.5.1 Basic commands

- pg get to obtain the value of a machine parameter
- pg set to change the value of a machine parameter
- pg update update a variable with a unit

1.5.2 Define commands

- pg layout Defines the layout of patterns on the substrate
- pg image Obtains an image of the substrate
- pg map Defines a mapping mode of the substrate
- pg marker Handles the marker definition and marker search functions
- pg table Defines stage position identifiers

1.5.3 Action commands

- pg adjust: Adjusts machine settings. Adjusting amounts to a controlled variation of the appropriate hardware and/or software settings in order to optimize a certain machine parameter, e.g. the beam quality or a change in some machine parameter.

- pg aperture Commands to control automatic aperture changer
 - pg archive The pg archive command allows a group of related machine parameters or calibration data to be saved, and restores at some later time, so that the mode of operation of the pattern generator can be changed quickly
 - pg calculate Calculates machine parameters
 - pg deselect Deselect a pattern or the dvm signal
 - pg display To activate the display/joystick function to show and control a subset of machine parameters
 - pg do Executes the specified script for the current layout definition
 - pg dvm The pg dvm command is used to automatically control the digital voltmeter (HP34401A) via the appropriate keyword
 - pg ebgp Commands related to the installation, start-up and shut down etc.
 - pg eht Control the Standalone EHT unit via the appropriate keyword in accordance with profiles and parameters that have been previously define with the pg set command
 - pg expose Exposes the substrate according to the specified keyword. If no keyword is specified by default the keyword pattern is used
 - pg measure Performs various measurements, including engineering measurements
 - pg move Move stable to specified position and if required also initializes the coordinate system
 - pg select Handles the selection of a new pattern, new holder data or new marker data
 - pg substrate Handles the loading and unloading of substrates

1.5.4 Information commands

- pg information The pg information command enables the operator to do a non-destructive read-out of groups of parameters, currently active or from a stored database

1.6 Graphical interfaces

With the HILL SHELLY library, it is possible to develop custom made commands / job procedures or applications. All the standard BEAMS commands are developed with this library as well as many of the acceptance tests, engineering and user tools, which are included in the BEAMS distribution (see chapter 10). Applications can vary from advanced marker search procedures, statistics measurements. Detailed information about creating custom made procedures can be found in chapter 8.

1.7 Raith Linux Tools

On the master control computer a set of Raith specific convenience tools are available. The tools set has a version number for reference. Which commands are available can be found by just entering at the command prompt

```
> tools
```

More detailed information can be obtained by entering at the command prompt

```
> tools <command>
```

2. LOADING AND UNLOADING SUBSTRATES

The following sections cover fitting a substrate into a holder, alignment, and loading into the tool. Finally, transferring the holder to the stage is described.

2.1 Fitting a substrate or mask-plate into a holder

Substrates and mask-plates must be loaded manually into the holders. Use gloves while handling the holder and substrate.

2.1.1 Mask-plate

A loading jig is used for exchanging masks in the holders. This is illustrated below.

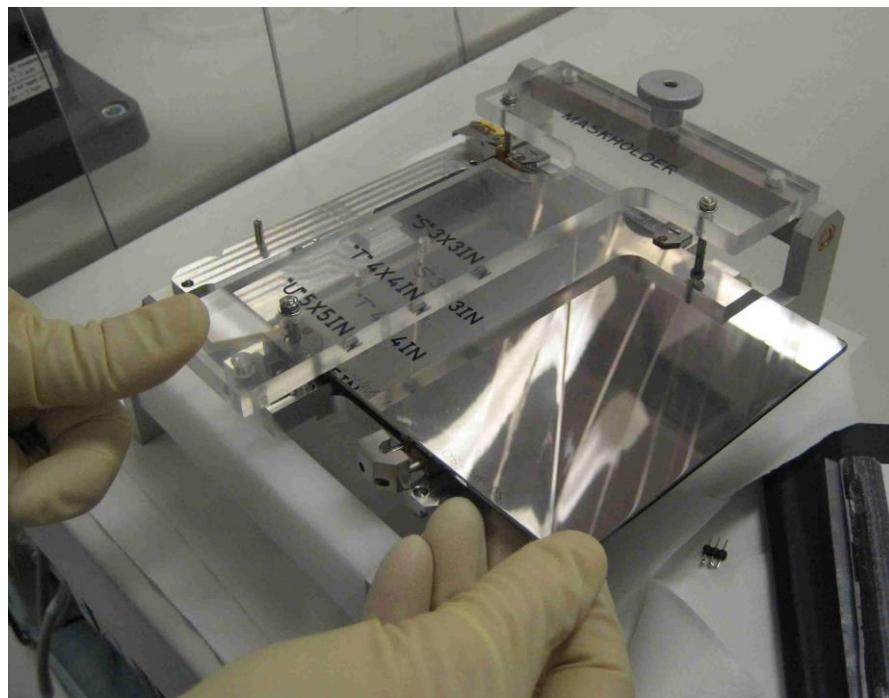


Figure 2 - inserting a mask-plate into the loading jig (EBPG5000 Plus)

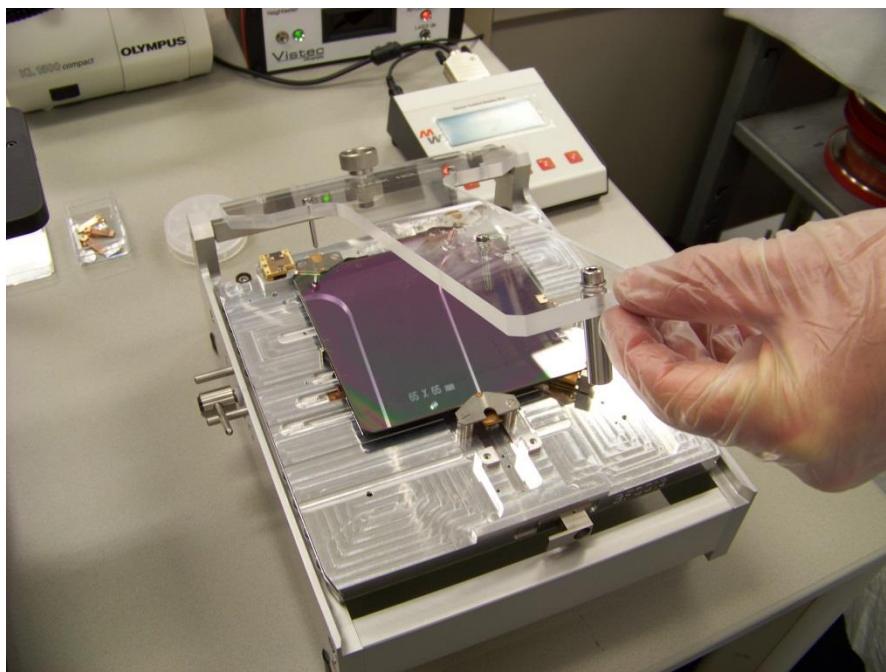


Figure 3 - inserting a mask-plate into the loading jig (EBPG5200)



Figure 4 - locking the holder to the loading jig with leaver (EBPG5200).

To load a mask-plate:

- Slide the mask-plate under the springs on the holder
- Install the earthing spring clip.
- Pull out the corner-stop, rotate it, and latch it on the corner.

2.1.2 Substrate

Substrates are loaded into the holder, as illustrated below

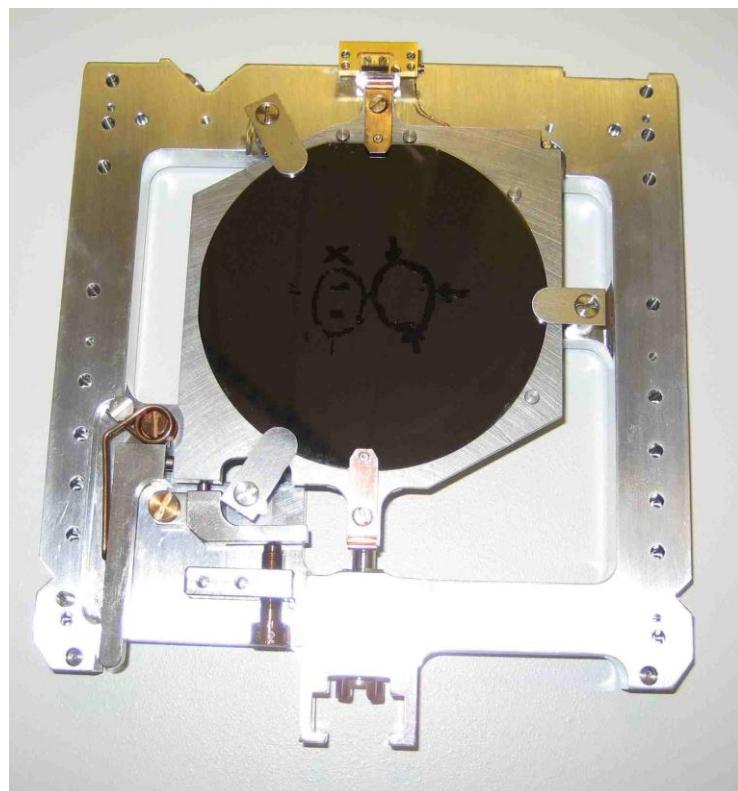


Figure 5 - substrate on a holder (EBPG5000 Plus)

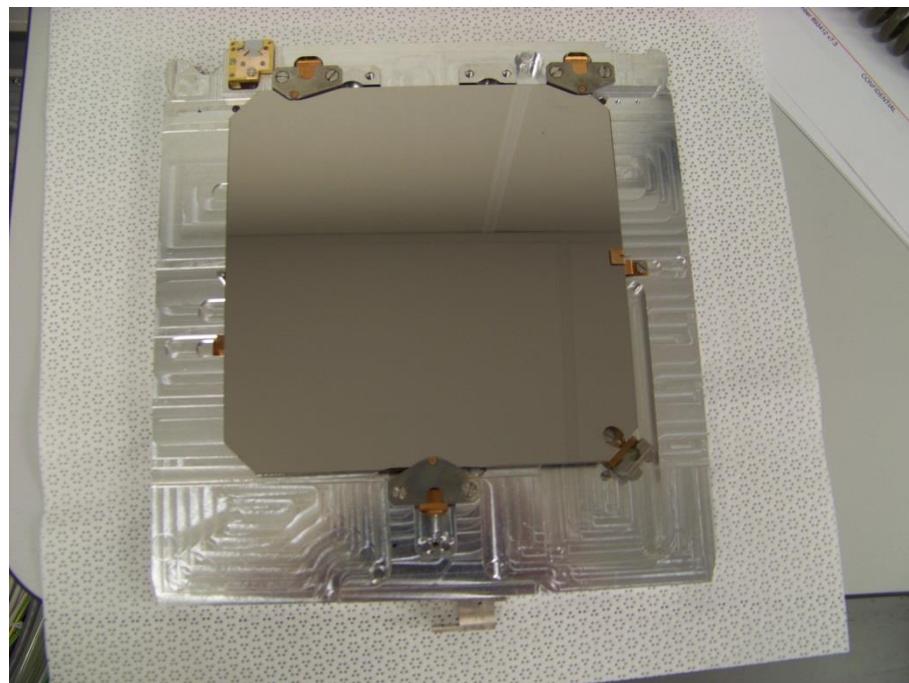


Figure 6 - substrate on a holder (EBPG5200)

2.2 Coarse alignment of substrate for direct write

The substrate must be rotated so that the pattern is roughly aligned to the stage axes if direct write is to be carried out. This is in order for the machine to be able to place the pattern with the best accuracy and in order to find the global marks easily (and possibly automatically).

Clamp the substrate to the table on which it sits using the small screw operated clamps around the

outside of the table. These ensure that when the table is rotated the substrate moves with the table and does not stick to the front face referencing points.

2.2.1 Measuring the rotation

Place the holder on the alignment microscope, which is provided for this purpose and clamp it into position. Focus on the pattern on the substrate. Find a feature which is repeated across the substrate at intervals in X or Y. Move the stage in the Y direction to the next feature and observe in the microscope whether the second feature is shifted to the left or right relative the first (Or move the stage in the X direction and observe in the microscope whether the second feature is shifted up or down.). If the relative movement is more than about 3 μm per mm distance between the two features then an adjustment to the rotation should be made as described below.

2.2.2 Adjusting the rotation

Make an adjustment to the table rotation using the thumbscrew to the side of the substrate. Measure the rotation of the wafer as described above. Repeat the adjustment until the rotation is $< 3 \mu\text{m}/\text{mm}$. However, it is strongly advised to do better, e.g. $< 1 \mu\text{m}/\text{mm}$, i.e as good as possible. It will pay-off in the quality of the lithography.

2.3 Holder loading/unloading in/from airlock

Use gloves while handling the holder and substrate.

The holder, with its substrate or mask-plate, is loaded into the loader/airlock as follows:

The EBPG5000 Plus system has a permanently installed 10-holder cassette, into which holders are inserted.

The EBPG5000 PlusES system has a 2-holder cassette, which is removed from the loader, to allow holders to be inserted.

Some of the illustrations below show the 10-holder system, but a similar procedure applies to the 2-holder systems installed on EBPG5000 PlusES systems.

2.3.1 Vent the airlock:

1. Press 'SET LOCK VENT' on the vacuum control panel, as shown below.

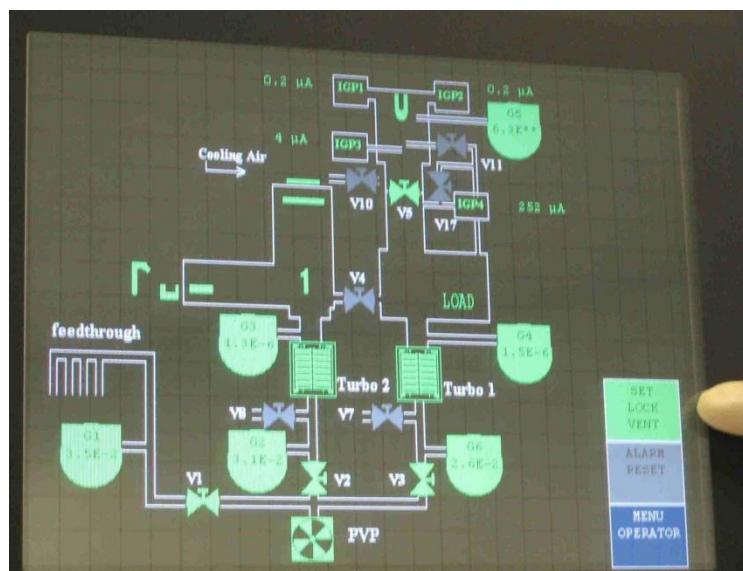


Figure 7 - vacuum control panel (EBPG5000 Plus with panel view)

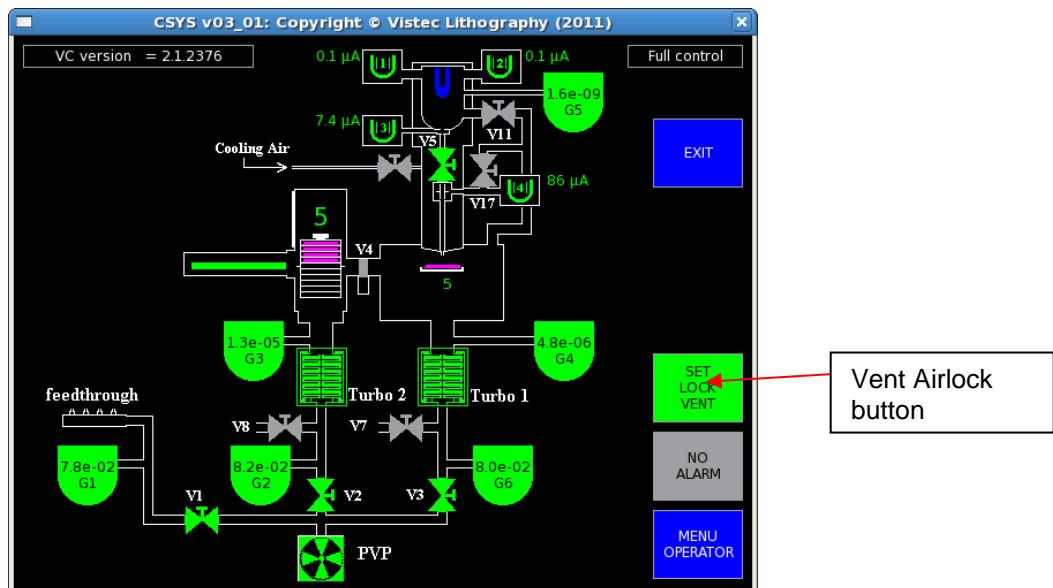


Figure 8 - vacuum control panel (EBPG5000 Plus / EBPG5200 Csys)

2. Wait while the airlock turbo-pump spins down. The vacuum panel shows Turbo 2 as 'B' (for braking). Also, the cassette moves to the 'LOAD' position, as shown by the cassette numbers (e.g. "1" in the above figure) changing; when complete it shows 'L'.
3. Wait for the completion of the vent: valve V8 shows green during the vent. This normally takes a few minutes. When complete V8 shows as a grey colour. This means that the airlock may be opened.

2.3.2 Fit the holder in the loader (EBPG5000 Plus)

1. Open the airlock door as illustrated below, by lifting the handle on the door.

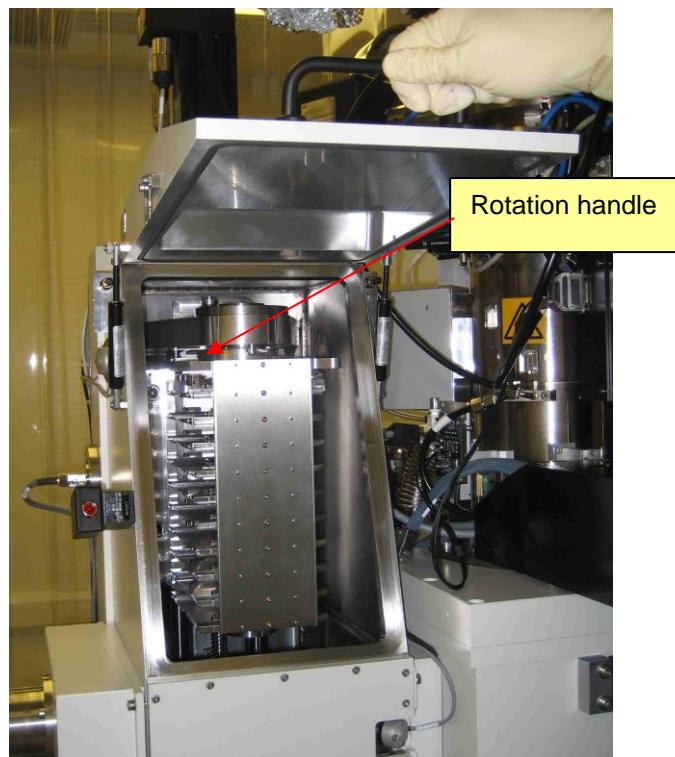


Figure 9 - The Loader with its door open

2. Rotate the 10-holder cassette clockwise by using the rotation handle, until the holders are to the front.
3. Place the holder into an empty position in the cassette, as shown below.

	IMPORTANT	The holder must be orientated so that the 'U'-shaped fitting is inserted into the cassette, as below
--	------------------	--

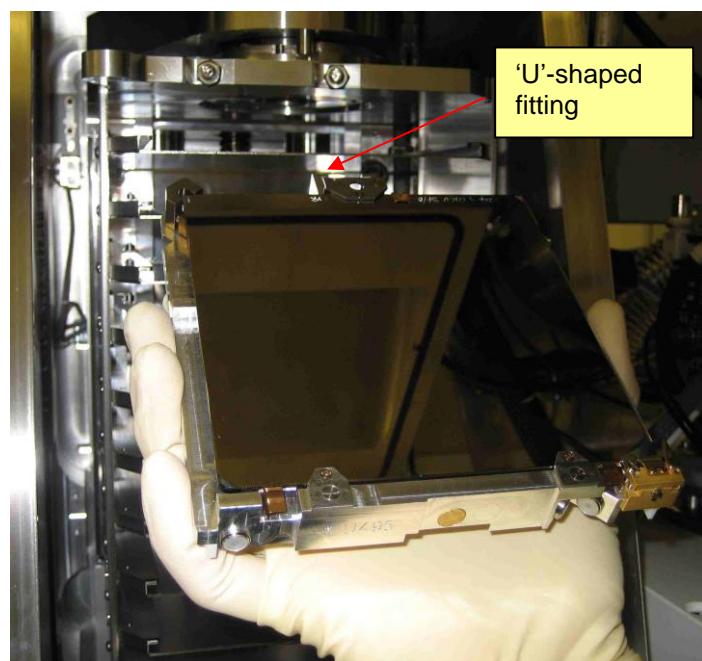


Figure 10 - loading a holder- holder orientation

4. Push the holder into the cassette slot – it slides in the ‘v-way’ spring clips until fully inserted. The following picture shows the holder loaded into the cassette



Figure 11 - Loading a holder- final position

5. Rotate the cassette anti-clockwise to orientate it correctly for the stage.
6. Close the airlock door.

	CAUTION	Keep fingers clear as the door is closed- to avoid any possibility of trapping fingers.
--	----------------	---

2.3.3 Fit the holder in the loader (EBPG5200)

1. Open the airlock door as illustrated below, by lifting the handle on the door.

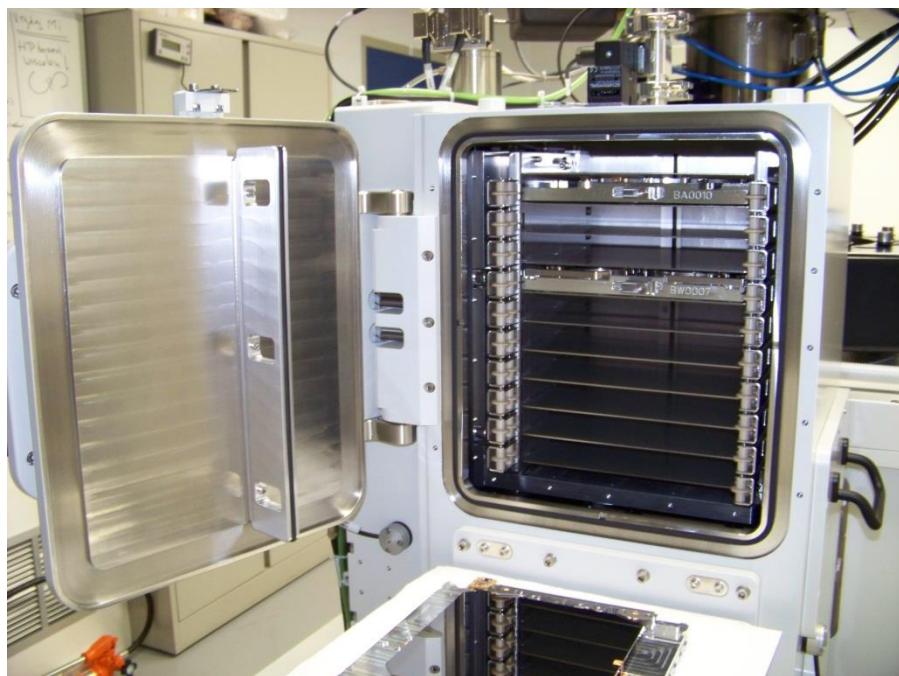


Figure 12 - The Loader with its door open

2. Place the holder into an empty position in the cassette, as shown below.

	IMPORTANT	The holder must be orientated so that the 'U'-shaped fitting is inserted into the cassette, as below
--	------------------	--

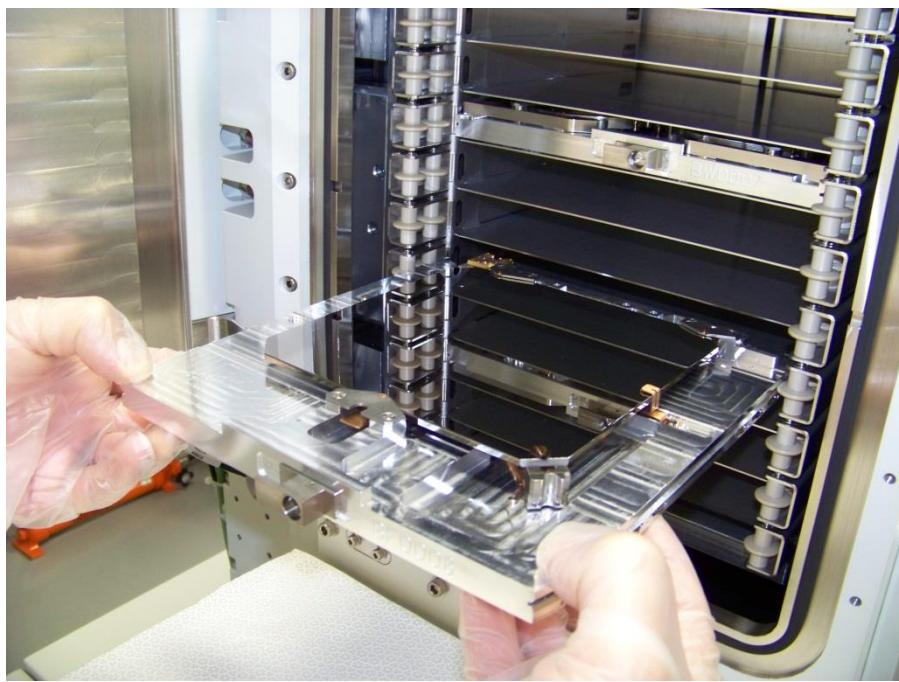


Figure 13 - loading a holder

3. Push the holder into the cassette slot – it slides in the 'v-way' spring clips until fully inserted. The following picture shows the holder loaded into the cassette

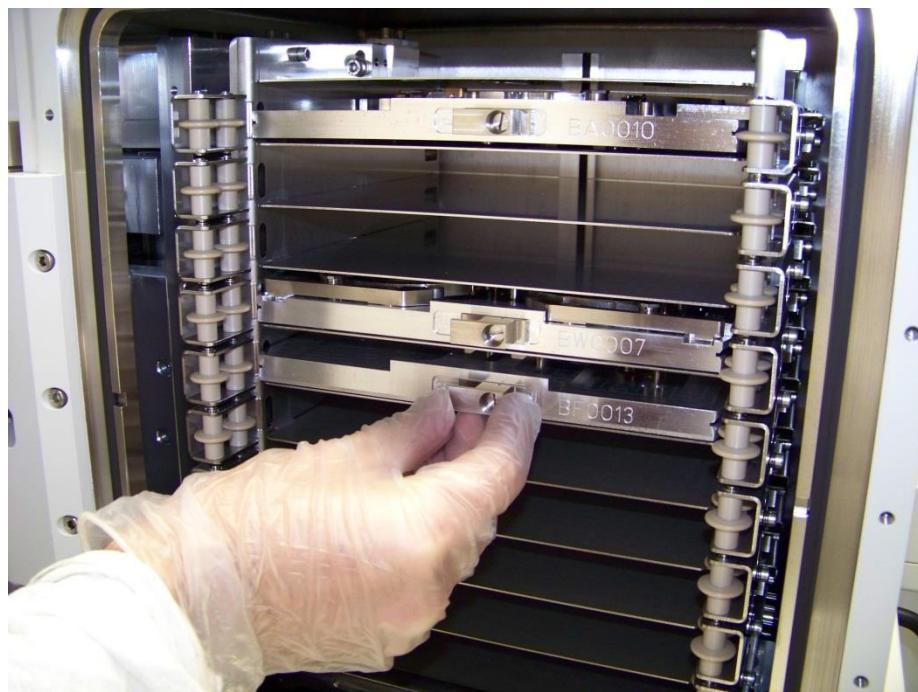


Figure 14 - Loading a holder- final position

4. Close the airlock door.

	CAUTION	Keep fingers clear as the door is closed- to avoid any possibility of trapping fingers
--	----------------	--

2.3.4 Pump the airlock

1. Press 'SET LOCK VACUUM' on the vacuum control panel: this will cause V1 & V3 to close (go grey) and V2 to open (go green). Note that the turbo starts accelerating (shows a green 'A') then the cassette is initialised. I.e. it cycles through the cassette positions to identify holders; e.g. displaying a '1' (as in Figure 31) if holder 1 is loaded.
2. Wait until gauge G3 goes green; this should happen when the pressure is less than 5×10^{-5} Torr.

The airlock is now under vacuum and the holder is ready to be transferred to the stage.

2.4 Holder loading/unloading on/from stage

When the pressure in the airlock is low enough, holders can be transferred to / from the stage. This is described below.

At the operator workstation type

```
> pg info subs
```

... This displays the holder positions. E.g. shows an arrow against '1' if the holder was loaded to position 1.

Type

```
> pg subs load 1 (or other number, as appropriate) to load holder 1 to the stage.
```

If an existing holder is on the stage this will be unloaded to an empty cassette position.



For ultimate pattern placement accuracy and low drift it may be necessary to wait for an hour or longer to allow the temperature of the holder and substrate to stabilise. This time can be reduced by matching temperatures of the sample and holder to the airlock temperature before loading.

2.5 Holder initialisation

When a holder is loaded onto the stage, the holder parameters should be restored into the control software. This is done by a single command. The command makes use of a holder table. First step is to setup the holder parameters.

2.5.1 Holder parameters

The holder parameters specify the specific holder

for an example.

:	holder	type	size	name	marker	cup	centre	movpar			
:			[inch]		x [mm]	y [mm]	x [mm]	y [mm]	x [mm]	y [mm]	[ms]
: B027 holders:											
0	Stage	m8	STAGE	1.5055 39.032	1.5015	32.677	105.000	105.000			
1	titan	m4	R204	40.216 1.1779	35.477	1.1772	105.000	105.000			
2	aluminum	m5	AF0570	15.055 15.032	15.015	20.677	105.000	105.000			
3	aluminum	m6	R206	40.216 1.1779	35.477	1.1772	105.000	105.000	100		
11	GaAs	w1	U600	14.859 76.197	14.866	80.950	105.000	105.000			
21	GaAs_a	w2	U624a	14.859 76.197	14.866	80.950	105.000	105.000			
22	GaAs_b	w2	U624b	14.859 76.197	14.866	80.950	105.000	105.000			
31	GaAs	w3	U625	14.859 76.197	14.866	80.950	105.000	105.000			
41	Si_a	w4	AB0058	15.388 17.433	15.454	22.336	105.000	105.000			
42	Si_b	w4	AB0058	15.388 17.433	15.454	22.336	105.000	105.000			
43	Si_c	w4	AB0058	15.388 17.433	15.454	22.336	105.000	105.000			
44	Si_d	w4	AB0058	15.388 17.433	15.454	22.336	105.000	105.000			
51	Si	w5	R207	14.880 16.571	14.894	22.080	105.000	105.000			
61	Si	w6	R208	14.880 16.571	14.894	22.080	105.000	105.000			
81	Si	w8	R209	14.880 16.571	14.894	22.080	105.000	105.000	200		

The columns in the above table are described below.

Index/name

The first column (index) and the fourth column (name) are used to select a specific holder. The name must be a unique string containing no spaces

Holder descr

The second column, holder description, is a short string explaining what type of holder it is. The string can be altered upon customer's specification, no spaces allowed.

Size

The third column is the holder size in inches, precede with a w or m to indicate wafer or mask. For the value, decimals are allowed.

Marker

The marker position is the position of the calibration marker in millimetres relative to the home position.

Cup

The cup position is the position of the faraday cup for measuring the beam current. The position is relative to the home position after a "pg adjust table coordinates" command on the calibration marker.

Center

The center position is the position of the substrate center. The position is relative to the home position after a "pg adjust table coordinates" command on the calibration marker.

Movpar

Optional the movpar parameter can be specified in case this value is holder specific.

2.5.2 Setting up the holder parameters

The holder parameters specify the mechanical characteristics of a holder. The holder parameters are stored in an ASCII file in the archive directory (cd \$PG_ARCHIVE).

The file looks identical to the holder parameter list in the table. To add a new holder in the table, open the file archive.holders for editing and fill-in the next items:

Index

A random chosen number, the index does not need to be successive. A commonly used order is the slot position in the airlock.

Holder description

A string with no spaces, specifying the type of holder.

Size

Stating the substrate size in inch preceded with a w or m. A decimal values are allowed.

Marker position

The position of the calibration marker relative to the home position. Execute the "pg move home" command, move to the expected marker position. Locate the marker with the SEM facility on the system. Let the control software find the marker, "pg move marker /relative 0,0". Record the marker position in the table, use the command "pg get tab /measure", this will return the stage coordinates relative to the home position. Store the marker position into the control software with the command "pg marker calibrate <tab_x>,<tab_y>. Fill-in the stage coordinates as found with "pg get tab /measure".

Cup

The cup position is the position of the faraday cup for measuring the beam current. Do an "pg move marker" to find the calibration marker. Execute a "pg adjust table coordinates" to attach the coordinate

system to the marker position. Go to the expected cup position; center the cup on the SEM monitor. Get the table positions with the command “pg get tab” and copy these into the table.

Center

The center position is the position of the substrate center. The position is supplied with the datasheet of the holder.

Movpar

Optional movpar parameter.

1. If a holder has more substrate tables then make a new entry in the holder table and set the other substrate center.
2. Finally -Save the ASCII file.

2.5.3 When and how to update a holder marker position

When a marker is use, it will be contaminate gradually, this will result that the measure spot size will grow.

By previous experiments where a beam of 5nA (apart of 300μm) is repeatedly used for marker search and the spot size and fine focus are measured. In Figure 15 results are plotted. From the graph it can be seen that the fine focus does not change much when the spot size is less than 75nm.

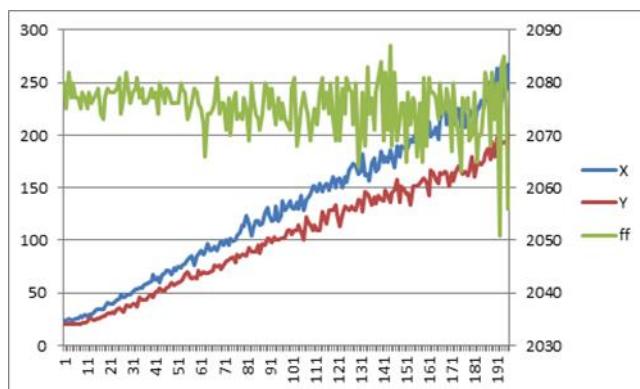


Figure 15 Measure spot size and fine focus over time.

The images of high-resolution work on HSQ prove that with a spot size of (53nm, 50nm), the system still performs high quality lithography.

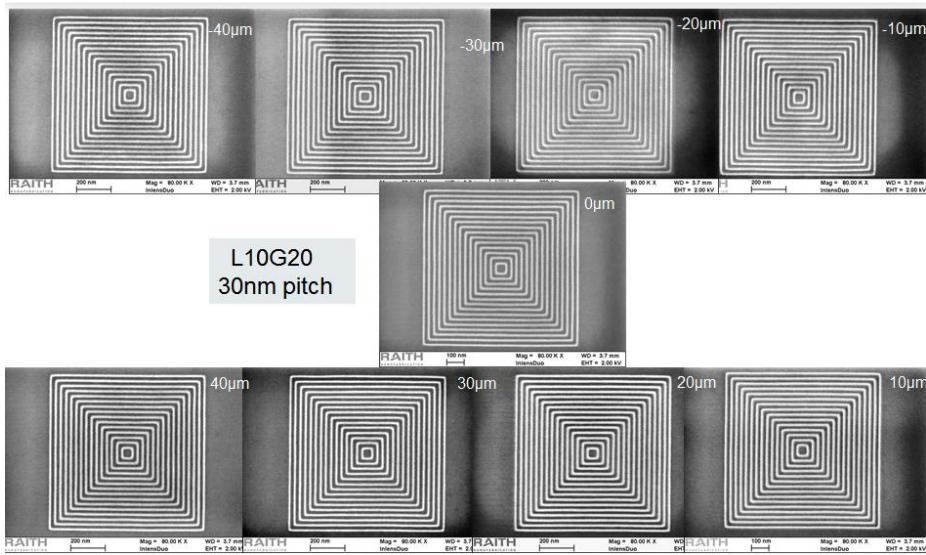


Figure 16 Nested squares at difference substrate heights.

Procedure

Presetting and testing

1. From the command prompt, enter `pg info archive beam` to show the list of beams.
2. Enter `pg archive restore beam 5nA_300um` to restore a 5nA beam with an apart of 300μm.
3. Enter `pg adjust resolution 1nm` to set resolution to 1nm
4. Enter `pg adjust ebpg 508` to calibrate beam. For a large field machine (1mm) use 508 and for a small field machine (500um) use value 254.
5. Enter `pg measure spot size` to measure and show the spot size.
6. If the spot size is smaller than 75nm no further action is needed, when the spot size is larger than 75nm then continue with step 7.

Select new calibration marker

7. Enter `pg move home` to reset the coordinate system.
8. Enter `mvm` or `pg move marker` to move stage to current calibration marker.
9. Move the stage to the next marker and center it by using `cmon` (or `csem`) .
10. Enter `mvm /rel 0,0` or `pg move marker 0,0 /relative` to let the machine center the marker.
11. Enter `pg measure spot size` to measure and show the spot size.
12. If the spot size is smaller than 75nm then continue with step 13 to save this position, when the spot size is larger than 75nm then repeat from step 9.

Save selected calibration marker

13. Enter `pg holder save` to save all holder parameters to `archive.holders`.



`archive.holders` is file which contents holders parameter and is located in `$PG_ARCHIVE`. To show enter `more $PG_ARCHIVE/archive.holders` or to edit enter `gedit $PG_ARCHIVE/archive.holders`

3. MACHINE SETUP

3.1 Deflection: resolution and beam step size

The deflection system of the EBPG consists of a double deflection: the main field deflection and the sub field deflection. The latter often also referred to as trap field deflection. With the fast sub deflection shapes can be exposed at a maximum pixel frequency of 50 MHz. The slower main deflection is used to position the shape within the main field.

The resolution settings of both deflections are independent. This makes it rather complex to setup both deflections in such a way that proper patterns are to be exposed.

The basic commands to set, resp. get information of, the resolution are:

```
> pg adjust resolution [<resolution> [bss=<val>] [main=<val>] [trap=<val>]]
> pg information adjust resolution
```

e.g.

```
> pg adjust resolution
```

In this case there are no parameters specified. That will imply that BEAMS will obtain the parameters from the selected file; In case there is no pattern selected an error message will be given, indicating there is no pattern selected:

```
> pg adjust resolution
%ENG_E_NOMNPA      - : no main pattern has been declared
> pg adjust resolution 20nm
> pg information adjust resolution
resolution      : 20.000000 nm      max main field size : 436.906667 um
beam step size : 20.000000 nm      max trapezium size  : 4.524878 um
main resolution: 6.66666667 nm [3]
trap resolution: 0.48780488 nm [41]
```

In this case BEAMS will set the optional values automatically to values, which provides the largest field sizes from main and trap to achieve best throughput. This strategy should be also pursued by packages as CATS and LayoutBEAMER. The above example and next to follow in this section are for the HR configuration at 100 kV operation, i.e. the main deflection has a resolution range from 1.25 nm to 7.8125 nm. The largest integer yielding a main resolution less than 7.8125 nm is 3, as indicated. By default the beam step size is equal to the resolution. The sub deflection has a resolution range from 0.078125 nm to 0.48828125 nm. In this case the largest integer which yields a value less than 0.48828125 nm is 41 as indicated.

Note that the fact that the main and sub resolution values look “funny”, i.e. with many digits, is of no concern. These values will be the result of the calibration of the full field for resp. main and sub field and therefore accurate.

The resulting main field sizes are then determined by the DAC size of the deflection multiplied with the resolution

The next step is to provide also the beam step size as a parameter. In normal circumstances, the beam step size should be a multiple of the resolution to avoid rounding problems. Choosing a beam step size larger than the resolution provides better placement compared to the resolution of the shape. This is in particular useful for exposure of grids with a very accurate periodicity, but larger beam step size to maintain the throughput.

```
> pg adj reso 20nm /bss=40nm
> pg information adjust resolution
resolution      : 20.000000 nm      max main field size : 436.906667 um
```

```

beam step size : 40.000000 nm      max trapezium size : 4.524878 um
main resolution: 6.66666667 nm [3]   trap resolution: 0.48780488 nm [82]

```

As stated above, the main and sub deflection are independent and it is possible to set a bss NOT a multiple of the resolution; but a warning will be given:

```

> pg adj reso 20nm /bss=30nm
%ENG_W_BRNOMUL - beam step size is not a multiple of the resolution: can
give rounding problems
> pg information adjust resolution
resolution : 20.000000 nm      max main field size : 436.906667 um
beam step size : 30.000000 nm      max trapezium size : 4.524677 um
main resolution: 6.66666667 nm [3]
trap resolution: 0.48387097 nm [62]

```

Finally it is possible to overrule the default selection of the main and/or trap field resolution as shown below. Obviously this has an impact of the field size (depending on the field size limitations due to kV as well) and therefore the throughput (more stage moves).

```

> pg adj reso 20nm /bss=40nm /main=5nm /trap=0.4nm
> pg information adjust resolution
resolution : 20.000000 nm      max main field size : 327.680000 um
beam step size : 40.000000 nm      max trapezium size : 4.524800 um
main resolution: 5.00000000 nm [4]
trap resolution: 0.40000000 nm [100]

```

The dependence of the field size as function of resp. resolution and beams step size show the well known "saw tooth relations" as depicted in the figures 22 – 24. In particular from figure 24 it can be observed the relevance of the 14 bit resolution of the sub deflection. For common beam step sizes, i.e. 1nm and higher, the sub field deflection size at 100 kV is always at or close to its maximum.

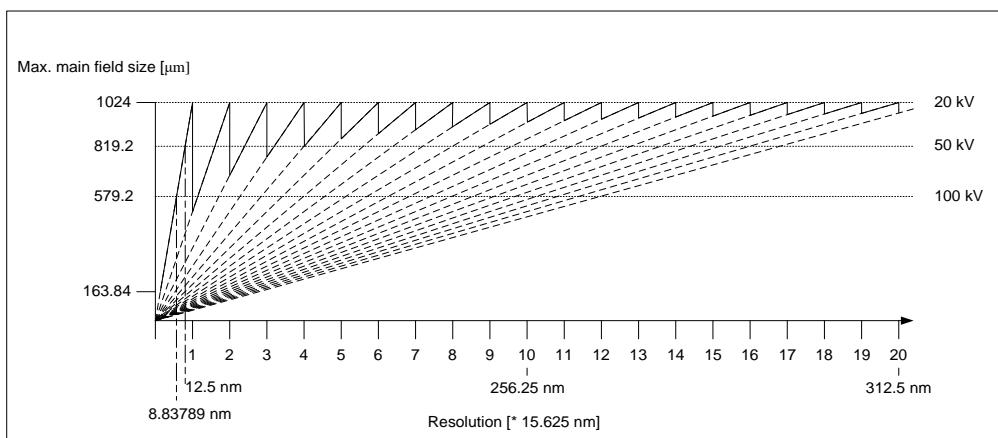


Figure 17 - Dependence of maximum main field size on the resolution for 16 bit HS system.

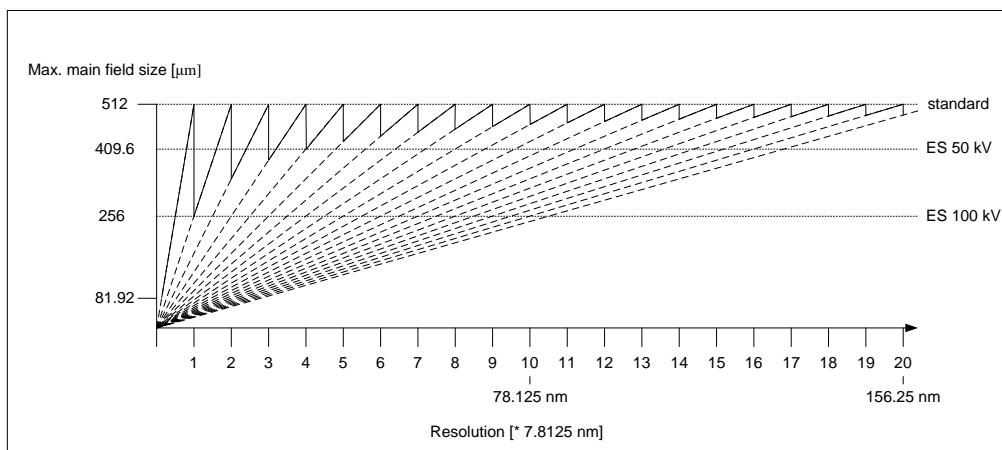


Figure 18 - Dependence of maximum main field size on the resolution for 16 bit HR system.

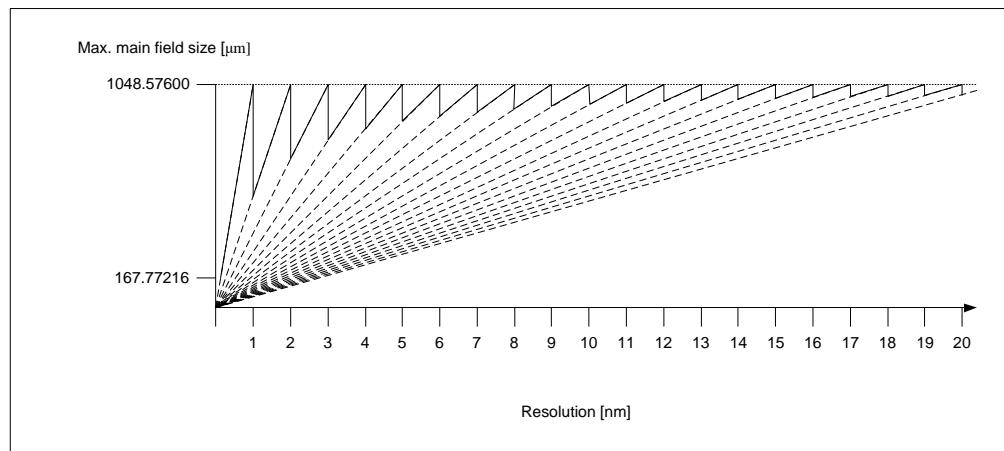


Figure 19 - Dependence of maximum main field size on the resolution for 20 bit HS system.

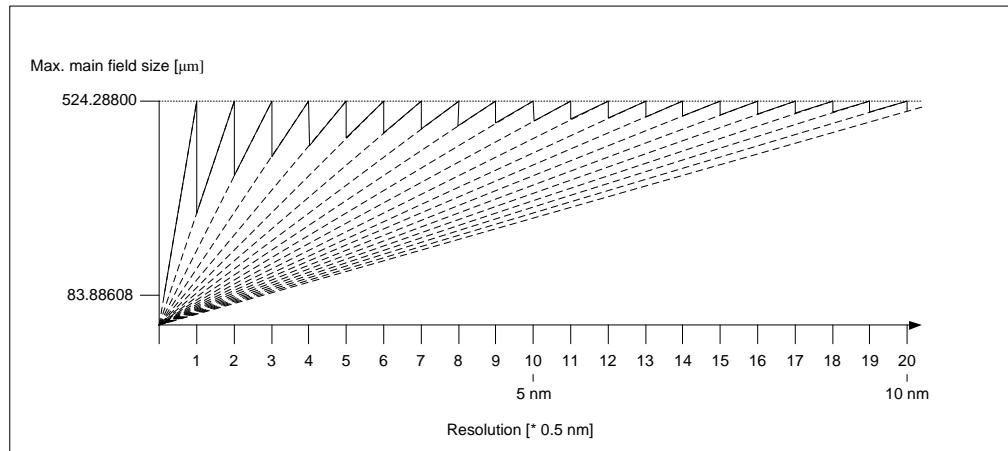


Figure 20 - Dependence of maximum main field size on the resolution for 20 bit HR system.

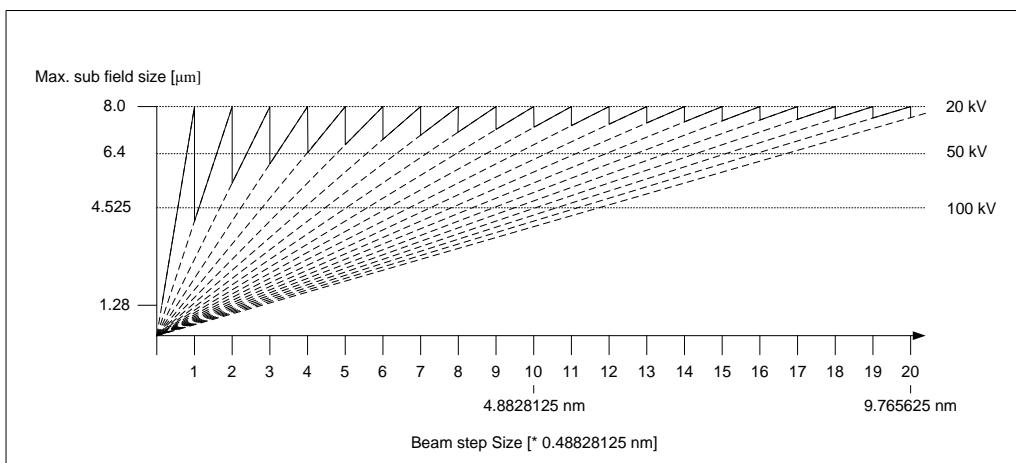


Figure 21 - Dependence of maximum sub field size on the resolution.

	CAUTION	The sub field size dependence is the same for both, 16 bit and 20 bit main deflection systems
---	----------------	---

Because of the much smaller address grid of the trap deflection, This can be used to improve shape placement as well, using the combination of main deflection and trap deflection offset. This is shown in the example below:

```
> pg information adjust resolution
resolution : 0.500000 nm           max main field size : 98.304000 um
beam step size : 0.500000 nm       max trapezium size : 4.096000 um
main resolution: 1.50000000 nm [1/3]
trap resolution: 0.25000000 nm [2]
```

Actually, it is even possible to increase the main resolution even further and with it the main field size. Altogether it is possible to achieve over 22 bit addressing resolution.

```
> pg adjust resolution 0.5nm /main=7.5nm
> pg information adjust resolution
resolution : 0.500000 nm           max main field size : 491.520000 um
beam step size : 0.500000 nm       max trapezium size : 4.096000 um
main resolution: 7.50000000 nm [1/15]
trap resolution: 0.25000000 nm [2]
```

Or, pushing it even further

```
> pg adjust resolution 0.1nm /bss=1nm /main=7nm
> pg information adjust resolution
resolution : 0.100000 nm           max main field size : 458.752000 um
beam step size : 1.000000 nm       max trapezium size : 1.638400 um
main resolution: 7.00000000 nm [1/70]
trap resolution: 0.10000000 nm [1]
```

Obviously this extreme situation has the penalty of relatively small trap fields due to the small trap field resolution. This strategy is valid for both, 16 and 20 bit main deflection systems because the sub deflection is for both systems the same.

	CAUTION	Here we are discussing addressing grids up to 22 bit. This does NOT imply accuracy up to 22 bit !
---	----------------	---

3.2 Stage coordinates

3.2.1 Homing

3.2.2 Limit switches

3.2.3 Stage locking

```

Stage Controller Tester EBPG5000 FAST for both boards X and Y
X: disMot=0 lscMode=12 homed=1 ready=1 configured= 1 (0)
Y: disMot=0 lscMode=12 homed=1 ready=1 configured= 1 (0)
Get LSC Control/Status Register (always X and Y)
      X   Y      X   Y
Reset LSC Board    0   0  Selftest: EPLD    0   0
Disable Motor     0   0  Selftest: DPRAM   0   0
Find Home          0   0  Selftest: ROM     0   0
Stage is homing   0   0  Selftest: EEPROM  0   0
HomePos found     1   1  Selftest: LRAM    0   0
Failure(s) detected 0   0  Selftest: GRAM    0   0
Stage INWINDOW    1   1  Selftest: TEMP    0   0
Temp Alert         0   0
Temp Failure       0   0
UpLoading Busy     0   0
LimitSW1.side 0 Far 0   0
LimitSW2.side 0 Far 0   0  RECOVBSY        0   0
LimitSW1.side 1 Home 0   0  Stage In PIWIN   1   1
LimitSW2.side 1 Home 0   0  Stage In PIDACWIN 1   1
LSC_VME_READY     1   1  Stage READY      1   1
LSC is configured  1   1  FREEMOVEN     0   0

```

Ready ssh1: 3DES | 21, 61 | 24 Rows, 80 Cols | Linux

Figuur 1 Dependence of maximum main field size on the resolution for 20 bit HR system

Figure 22 - Dependence of maximum main field size on the resolution for 20 bit HR system.

3.2.4 Symbols

A position of the stage can be stored as a symbol with the following optional attributes:

substrate mapping

marker id

The maximum number of symbols which can be stored is 100.

examples:

The following command will create the symbol L1 with the current stage position and the current mapping mode:

```
> pg tab L1
```

In case the symbol already exists, a warning will be issued. In case a symbol needs to be updated, the stage coordinate must be specified. In case this must be the current stage position use the command

```
> pg tab L1 `pg get tab`
```

The command

```
> pg tab L2 32mm,56mm
```

will create the symbol L2 with the specified position in the current mapping mode. In case a symbol

needs updating, use the command:

```
> pg tab L2 `pg get tab`
```

To add to this position a marker with id "dwmar" use:

```
> pg tab L3 32mm,56mm /marker=dwmar
```

The symbols are used with the command

```
> pg move tab L3
```

The attributes are automatically applied, i.e. when a marker id is associated with the symbol, automatically a marker search will be performed at the particular stage position.

The attributes of a symbol can be modified with resp.

/MAP=<map id>

/NOMAP

/MARKER=<marker id>

/NOMARKER

To remove the symbol from the table, the /DELETE qualifier must be applied. Obviously all other qualifiers are in that case redundant. It is allowed to use wild cards (use in that case double quotes around the symbol id).

	CAUTION	Note that when a mapping is deleted (see next section) also the symbols defined in that mapping mode will be deleted.
--	----------------	---

To obtain the contents of a symbol, use the command

```
> pg information tab L3
```

To obtain all defined symbols use

```
> pg information tab "*" 
```

3.2.5 Stage distortion corrections

Stage distortion corrections

3.3 Z-stage

The EBPG5200 has an option for a z-stage. This way it is possible to lower the stage (compared to conventional systems) by 10mm. This allows to compensate for height differences for substrates and even makes it possible to do exposures on non-flat surfaces.

A machine with a zstage has 2 working levels.

HIGH, which is the old working distance. The same as systems without a zstage.

LOW is 10mm lower (as seen from the outside). Low, in this case, means the distance from the final lens is 10mm bigger!

The zstage consists of 2 legs that carry the normal x-y-stage. These 2 legs run in synchronised mode to make sure it stays level when it is moving. On the travel, there are 2 index points. These can be found during initialising and define the 0-level for the z-coordinate system.

When you drive down the z-stage to the low working level, the Final lens rotation influence changes the direction of the main deflection system (compared to the stage coordinate system). This causes a rotation of about 7 degrees.

NOTE: the laser height meter only works at the HIGH work level. On the LOW work level, everything is relative to the calibrated height! The height at which you calibrate is, by definition, always 0. Everything else is relative to this calibrated height. Therefor at the low working level, you can only measure the height with the marker method and for that you need markers!

3.3.1 Z-stage homing

If the zstage coordinate is undefined or if Beams finds out numbers don't match correctly, the zstage needs to be homed again.

This can be done with the command:

➤ pg move zhome

If the coordinate system is defined, the zstage will move until it is under the index markers and then move up slowly to find them. Once they are found, the zstage will move to the previous defined working level.

If the z-coordinate system is not well defined, the zstage will slowly move down, searching the index markers. When they are both found, the stage is moved down a bit and then searches them again in the upward motion. After they are found, the zstage will move to the previous defined working level.

The last method can be forced with the command:

➤ pg move zhome /init

But in most cases, the /init can be omitted because the routine will do so, if needed, anyway.

3.3.2 Z-stage Work levels

Systems with a zstage have 2 work levels. HIGH, closest to the Final Lens. And LOW, 10mm down, further away from the Final Lens.

The Coordinate systems is set up in the way you would see it if you look via the SEM image, from the column, looking down.

You can select a working level with the command:

```
> pg set wl high
```

or

```
> pg set wl low
```

This will then set all kinds of things in the machine! It will select the setup of that worklevel for the current high tension setting. Select tables, corrections, bath temperature settings, move the x-y- stage to the home position, adjust the x-y coordinate system (for low working level) etc.

When changing the work level to low, the distance from the final lens to the substrate becomes bigger. This means that the rotation influence is bigger and the maximum field becomes bigger. The rotation effect is about 7 degrees and the maximum fieldsize will increase by 25%. Also the final lens current for this level is a lot lower than at the high work level.

To get things set up, the operator should give the following commands after every work level change:

- Adjust the resolution
- Restore a beam archive

After you set a work level, the zstage will move to the defined encoder position. This, then, becomes the new '0'-level for the system. Moving up or down is thus independent from the worklevel

3.3.3 Z-stage movement

After the zstage is set up, the position of the z-limit switches is measured with the command:

```
>pg measure stage zlimits
```

When completed correctly, the limitswitch positions minus 50um define the useable z-stage range.

The useable range depends on the selected work level and can be retrieved by:

```
>pg get maxstageztravel
```

NOTE: the returned values depends on the selected work level!

The zstage can be moved with the command:

```
>pg move zpos <position>
```

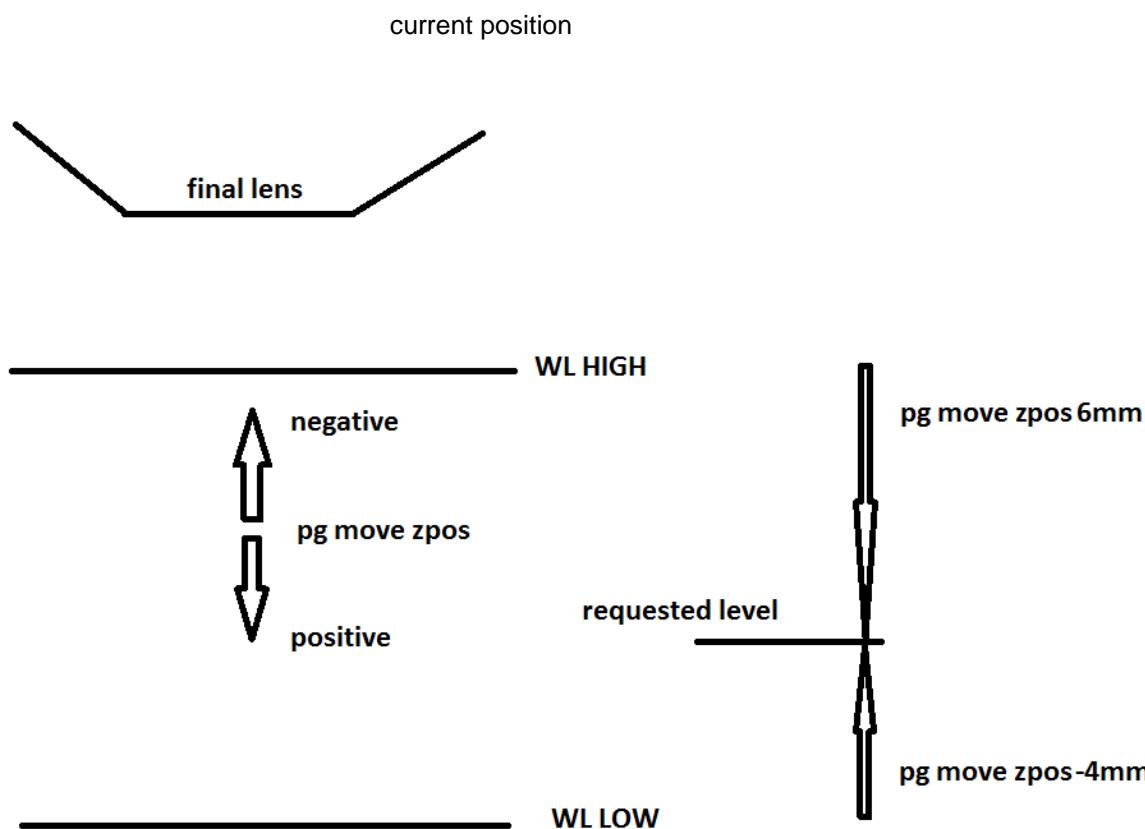
And has an optional /relative qualifier.

A few examples:

`>pg move zpos 100um` => moves the zstage 100um away from the final lens (down)

`>pg move zpos 0mm` => brings the zstage back to the defined worklevel position

`>pg move zpos -10um /relative` => move the zstage 10um closer to the final lens, relative to the



The above picture shows the work levels in relation to the final lens. It also shows that if you want to move to the "requested level", you will need to move +6mm when working at work level HIGH or -4mm when working at the LOW level!

3.3.4 HGTZSTAGE parameter

The machine parameter HGTZSTAGE can have the following values:

- 0) The zstage is not used at all for height correction. It will always stay at the current value, unless it is manually changed.
- 1) The zstage is used to bring the substrate to the calibration marker height. So, when the machine is calibrated, the height is measured. During the exposure, the measured height (with the heightmeter) is used and compensated with the zstage, so that the resulting height is the same as the height of the calibration marker.
- 2) Both the calibration marker height as well as the substrate height is set to '0'. During the calibration, the marker height is measured with the heightmeter and compensated with the zstage so that the measured height is '0'. During the exposure, the same happens on the substrate.

NOTE: This function normally uses the laser height meter system. This also means the limits of +/- 100um are valid for this routine! If you need to compensate for bigger height differences, you need to create user defined routines!!!

NOTE 2: Because the laser height meter is used, this routine does not work on the low working level!

NOTE 3: It is however possible to create a heightmap. When the height is “measured” the value will come from the map! You do need to create a heightmap and set the HMAPINV to 1,

3.4 Detectors

The tool is equipped with a set of 4 PM detectors to detect backscattered electrons. The 4 detectors are indicated with resp. NORTH, SOUTH, EAST and WEST, i.e. the orientation of the detector with respect to the coordinates system of the stage. Optional one or two TEM detectors are mounted on the stage.

3.4.1 PM detectors

The command to select the PM detector is

```
> pg select detector PM
> pg info select detector
```

Individual detectors can be switched on / off with the command

```
> pg set detect <value>
```

The value is the sum of:

North 0x04

South 0x01

East 0x02

West 0x08

All backscatter detectors selected:

detector PMHV

North 4047

South 4116

East 4061

West 4155

The following machine parameters are available to control the PM detectors:

BLEV Photomultiplier black level setting

BLEVx x photomultiplier black level setting (x=N,S,E,W)

PMHV Photo multiplier high voltage setting

PMHVx x photo multiplier high voltage setting (x=N,S,E,W)

To balance the 4 detectors a backscatter detector table has been set-up. This is normally a service issue and executed during a (preventive) maintenance action when required.

The command to add a current set of settings for all 4 detectors is

```
> pg adjust bsd load
```

or to load a specific entry in the table:

```
> pg adjust bsd load 7
```

An example of the command to change an entry (in this example 8) in the table:

```
> pg adjust bsd set 8 3100,3180,3270,3265 1560,1560,1560,1580
```

The values are the respective PMHV list and BLEV list.

To clear the entire table us the command

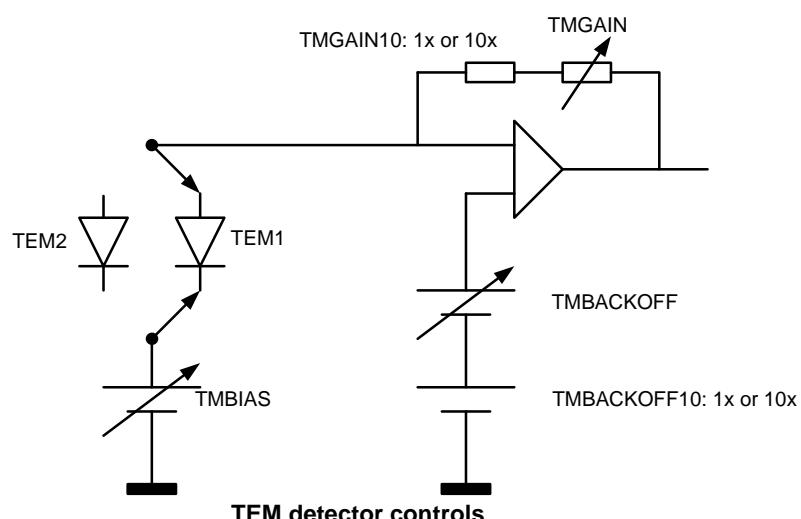
```
> pg adjust bsd clear
```

resp. for a specific entry (e.g. 8)

```
> pg adjust bsd clear 8
```

3.4.2 TEM detectors

```
> pg select detector TEM1
> pg info select detector
Transmission detector 1 selected
BackOff DAC value      : 4095
Bias DAC value         : 0
Continuous gain DAC value : 4095
Fixed gain              : 0 (1x)
Fixed gain backoff      : 0 (1x)
```



The following machine parameters are available to control the TEM detector function:

TMBACKOFF	Back ground offset adjustment
TMBACKOFF10	Fixed back ground offset 10x
TMBIAS	Bias voltage adjustment
TMGAIN	Fixed gain 10 x
TMGAIN10	Gain adjustment

3.4.3 SED detector

The secondary electron detector is a available as an option. One of the standard PM detectors (should be the SOUTH detector 1) has been modified to enable secondary electron detection (Everhart-

Thornley). The modified configuration is shown below.

To penetrate the conductive layer of aluminium (approx.. 50 nm), the secondary electrons are accelerated to at least 4 keV to penetrate the scintillator (YAG). In practice the required voltage is between 5 - 6 kV.

```
> pg set sedhv <value>
```

value in bits; 1 bit corresponds with 2 V; range 0 - 3000 bits (default).

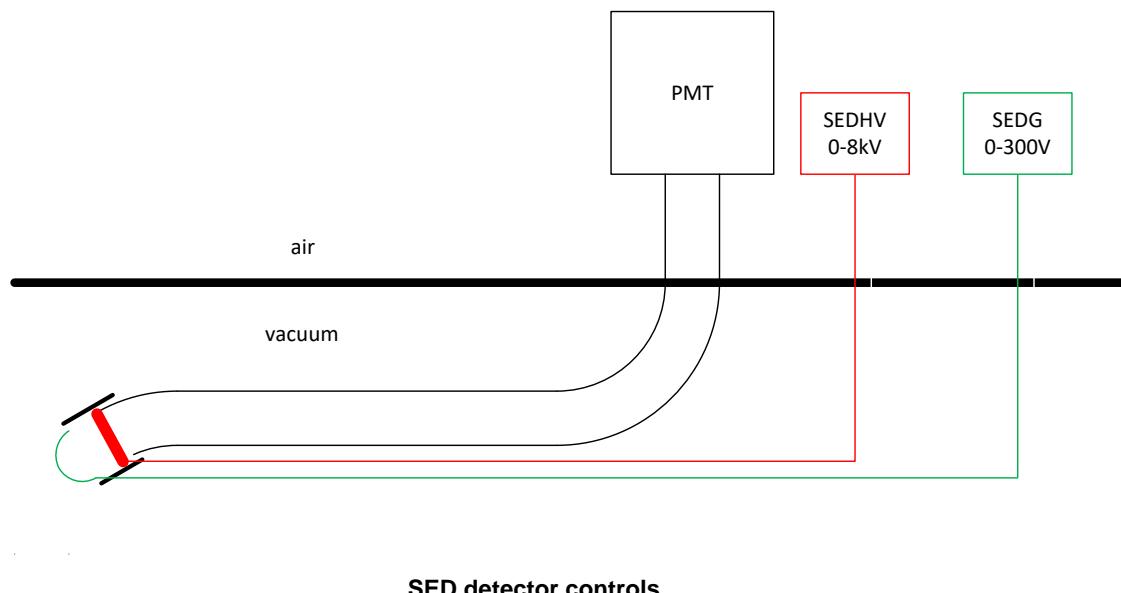
To suck the secondary electrons (0 - 80 eV kinetic energy) towards the detector, a voltage needs to be applied. In practice a voltage between 100 V and 200 V is sufficient. The maximum voltage is 409.5 V.

```
> pg set sedg <value>
```

value in bits; 1 bit corresponds with 0.1 V; range 0 - 2500 bits (default).

With the environment variables PG_SEDHVMAX and PG_SEDGMAX, the maximum values can be adjusted to a maximum of 4095. Take care adjusting this level. In case this is necessary, these should be defined in pg_local.

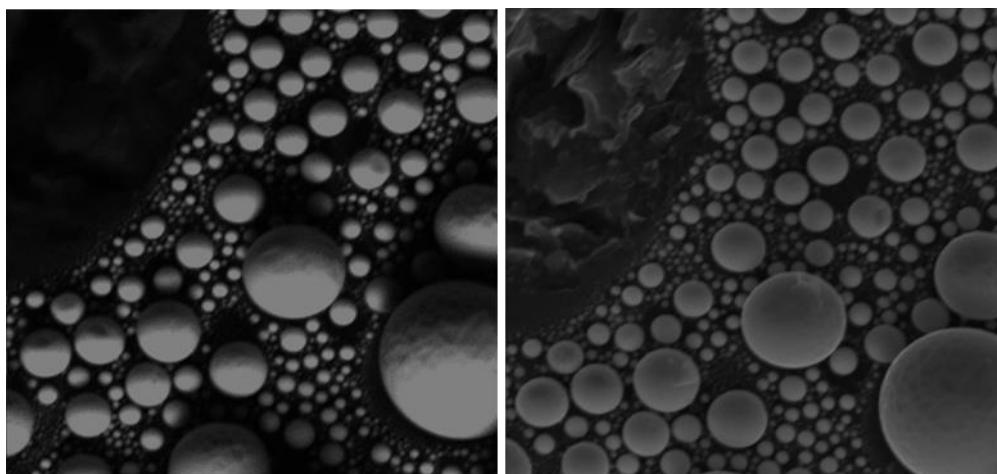
These parameters need to be tuned per system and should be executed by the Raith service engineer.



The detector is intended for SEM imaging visualize low contrast materials, e.g. graphene. The detector is used by switching on / off with the command

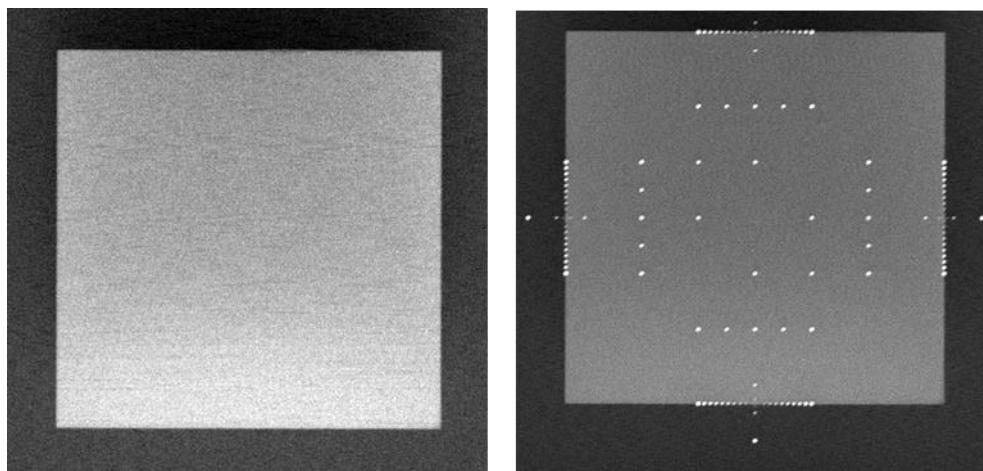
```
> pg set sed on|off
```

Note that switching the detector on can take some time, because the required voltages need to be ramped up. With environment variables PG_SEDHVDEL (default 1000.0 in ms) and PG_SEDHVINC (default 10 in bits) for the SEDHV ramping and environment variables PG_SEDGDEL (default 500.0 in ms) and PG_SEDGINC (default 100 in bits) the ramping of the SEDG can be modified.



SEM image with SED of tin balls @ 50 keV (magnification unknown).

(left: SED off right: SED on)



SEM image of an old 20 μm holder marker @ 50 keV.

(left: SED off right: SED on)

After switching on, the PM detector with the SED option must be selected to take advantage of the increased contrast.

```
> pg set detect 1
```

The PMHV level needs adjustment.

	CAUTION	Note that when the SED is switched on, due to the grid voltage applied on one side of the beam, the beam will be shifted with a small amount (0 .. ca. 1 μm) and the scan field might be slightly distorted.
--	----------------	---

The detector can be used for mark location and a calibration can be executed. However, care must be taken of the beam shift and a small field distortion. These influences are not quantified. It is very important not to switch off the SED detector during exposures to avoid a beam shift. In particular the

grid and acceleration voltage should not be changed. The calibration will correct the corners of the field. Possible main field distortions should be calibrated as usual with the SED and saved in a separate archive file.

	CAUTION	Note that using the SED for marker search and calibration is still under development . Each time the SED is used for calibration during exposure, the archive file with the main distortions must be restored and the detector must remain switched on all the time.
---	----------------	---

3.5 Final aperture controller

3.5.1 Commands to operate the automatic aperture selector

There are several commands to operate the controller. Before doing the setup, they will be explained.

`pg aperture calibrate`

This command is used to define the 3 positions of the apertures in the stick.

`> pg aperture calibrate <aperture number> <pos_long, pos_trans> <size>`

aperture number can be 1, 2 or 3

position, coordinates longitudinal and transversal

size is the aperture size.

Alternatively, just enter the command:

`> pg aperture calibrate`

and answer the questions.

`pg aperture command`

This is a command to send strings directly to the controllers. Only used in case of problems, debugging or changing parameters.

`> pg aperture command lve`

It is important to use /noanswer, if you do not expect an answer back.

`pg aperture help`

This command will show all possible commands that can be used with the "pg aperture command" function. To do something with them, you will probably need the manual of the controllers.

`pg aperture move pos`

This command will send a move command to the motors.

`> pg aperture move pos < pos_long, pos_trans > <option>`

`< pos_long, pos_trans >` : will move the motors to the given position

The /relative option makes a relative move to the current position.

`pg aperture move home`

this will re-initialise the controllers and then issue the HOME command.

This command can take around 30 seconds to complete.

```
pg select aperture
```

Here you can select one of the 3 predefined positions of the apertures.

```
> pg sel aperture <aperture_number>
```

<aperture_number> can be one of the 3 apertures (1, 2 or 3) or the aperture size.

If the aperture size matches one of the sizes in the tables, that one is selected.

The motors will be moved to the positions that were declared with the "pg aperture calibrate" command.

```
pg information aperture
```

With these commands you can get information about the status of the SMC controllers, motors and defined parameters.

```
> pg info aperture parameters
```

this command will show all the current parameters in the 2 smc controllers.

```
> pg info aperture status
```

this command will show the error registers of the controllers, the current position and information about the saved positions of the apertures.

```
pg get atab
```

The position that was set can be obtained with this command. To read back the real position from the controllers, use the /measure option.

3.5.2 Initial set-up of the apertures

After the hardware is installed, the apertures need to be found and set up.

The aperture should be manually setup and the position of the motors is noted down.

1 Initialize the controllers and move home.

```
> pg apert move home
```

2 Repeat the command

```
> pg apert move pos /relative 1000,0
```

(1 mm move and eventually in smaller steps) until the first motor reached the noted position

3 Repeat the command

```
> pg apert move pos /relative 0,1000
```

(1mm move) until the second motor reached the noted position.

4 Measure the beamcurrent to make sure the aperture is approximately at the right position. The value of the BeamCurrent should show if the expected aperture is selected.

5 Try to do an 'pg adj aperture'. This should show that the automatic aperture changer will adjust the knobs.

6 When the aperture is aligned, read the current position:

```
> pg get atab /measure
```

and note down the position

7 Save the current position to the aperture table:

```
> pg aperture calibrate <apert_pos> `mpgm atab` <apert_size>
```

8 Now move the right motor to the next aperture (around 3mm further) but do so in small steps.
If you switch on SEM, its easy to see when the new aperture is approximately aligned.

9 Repeat step 5 thru for this and the last aperture.

10 Now all 3 apertures are defined and can be selected with the command:

```
> pg select aperture <apert_number>
```

3.5.3 Normal operation

Defining an aperture

After an aperture is found (if needed by making small steps and the command 'pg aperture move pos /rel <long,trans>' you can define this position to one of the 3 aperture definitions:

```
> pg aperture calibrate <apert_nr> <pos_long,pos_trans> <apert_size>
```

Save the global data to keep this position:

```
> $pg save
```

The current stored positions can be verified with the command:

```
> pg info aperture status
```

Moving to an aperture

Manual selection of an aperture:

Select one of the 3 apertures with the command "

```
> pg select aperture <apert_nr>
```

where aper_nr = 1, 2 or 3, or

```
> pg select aperture 400
```

(if an aperture with size 400 was defined, this will be selected)

Automatic selection with archive restore beam:

When the command:

```
> pg archive restore beam <name>
```

is executed, Beams will check if there is an automatic aperture changer.

When the SMC is defined, Beams will select the aperture which was saved in the archive. After a successful selection, a "pg adjust aperture" will be done to make sure the aperture is aligned.

Optimize the aperture position

This is done with an automatic version of the :

```
> pg adjust aperture
```

If the SMC was detected by Beams, this will start the automatic procedure.

If needed, this position can be saved to the aperture position with the command:

```
> pg aperture calibrate <aper_nr> `mpgm atab` <apert_size>
```

After you do a "pg aperture calibrate", save the global data to save the positions:

```
> $pg save
```

3.5.4 Check the status of the aperture controller

With the command

```
> pg info aperture status
```

You will get the error status of both motors, the current position and the saved positions of the 3 apertures. When the motors don't want to move anymore, this is the way to find out what is wrong.

```
pg@reagan:~
```

```
[pg@reagan pg]~> pg info apert status
```

Errorstatus	Long	Trans
negativeEndOfRun	TRUE	TRUE
positiveEndOfRun	TRUE	TRUE
peakCurrentLimit	FALSE	FALSE
RMSCurrentLimit	FALSE	FALSE
shortCircuit	FALSE	FALSE
followingError	FALSE	FALSE
hommingTimeOut	FALSE	FALSE
wrongESPStage	FALSE	FALSE
DCVoltageTooLow	FALSE	FALSE
outputPowerExceeded	FALSE	FALSE

```
LONGITUDINAL, status = NOT REFERENCED ESP stage error
TRANSVERSAL, status = NOT REFERENCED ESP stage error

Position = -0.000000_mm, -0.000000_mm

Aperture values set in global data :

aperture 1    pos = 5.00000, 9.98000  aperture = 300 um
aperture 2    pos = 8.01400, 9.98500  aperture = 200 um
aperture 3    pos =11.04700,10.00400  aperture = 400 um
```

```
[pg@reagan pg]~>
```

In the picture above, no motors were connected to the controllers, therefor some errors showed up!

3.5.5 Check the parameters in the controller

With the command

```
> pg info aperture parameters
```

you will see all the parameters for both axis.

See picture below for the standard values. Only the Hysteresis compensation has been changed from 0 to 0.02.

```

pg@reagan:~ pg@reagan pg]~> pg info apert parameters

PARAMETER           LONG      TRANS
Acceleration          4          4
Backlash compensation   0          0
Hysteresis compensation 0.02      0.02
Driver voltage          24         24
Low pass filter for Kd 1000      1000
Following error limit    1          1
Friction compensation    0          0
HOME search type          4          4
Stage identifier          LTA-HL    LTA-HL
Jerk time                0.04      0.04
Derivative gain          29.7992   29.7992
Integral gain             993306   993306
Proportional gain          29799.2  29799.2
Velocity feed forward     14.8996   14.8996
HOME search velocity       0.5        0.5
HOME search time-out       110        110
Maximum/Peak output current limit 0.213      0.213
RMS output current limit   0.1065   0.1065
Average period for RMS curr. calc. 3          3
Controllers RS-428 address    2          2
Control loop state          1          1
Negative software limit      0          0
Positive software limit       25         25
Encoder increment value     7.37358e-06 7.37358e-06
Velocity                   1          1
Smartstage configuration      1          1

[pg@reagan pg]~>

```

3.5.6 Modify the parameters in the controller

To compensate for a hysteresis effect in the assembly, we need to change/verify one of the parameters in the controllers. All parameters are set to the default factory values, except for the hysteresis compensation.

To see all commands, see the command "pg aperture help".



```
[pg@reagan pg]~> pg apert help
SMC
The instruction manual is added in the BEAMS distribution

Command format:
  nnAAxx
nn - Optional or required controller address
AA - Command name
xx - Optional or required value or "?" to query current value

Summary of commands:

AC      Set / Get acceleration
BA      Set / Get backlash compensation
BH      Set / Get hysteresis compensation
DV      Set / Get driver voltage
FD      Set / Get low pass filter for Kd
FE      Set / Get following error limit
FF      Set / Get friction compensation
FR      Set / Get stepper motor configuration
HT      Set / Get HOME search type
ID      Set / Get stage identifier
JD      Leave jogging state
JM      Enable / disable keypad
JR      Set / Get jerk time
KD      Set / Get derivitave gain
KI      Set / Get integral gain
KP      Set / Get proportional gain
KV      Set / Get velocity feed forward
MM      Enter / Leave DISABLE state
OH      Set / Get HOME search velocity
OR      Execute HOME search
OT      Set / Get HOME search time-out
PA      Move absolute
PR      Move relative
PT      Get motion time for a relative move
PW      Enter / Leave CONFIGURATION state
QI      Set / Get motor's current limits
RA      Get analog input value
RB      Get TTL input value
RS      Reset controller
SA      Set / Get controller's RS-485 address
SB      Set / Get TTL output value
SC      Set / Get control loop state
SE      Configure / Execute simultaneous strated move
SL      Set / Get negative software limit
SR      Set / Get positive software limit
ST      Stop motion
SU      Set / Get encoder increment value
TB      Get command error string
TE      Get last command error
TH      Get set-pint position
TP      Get current position
TS      Get positioner error and controller state
VA      Set / Get velocity
VB      Set / Get base velocity
VE      Get controller revision information
ZT      Get all axis parameters
ZX      Set / Get SmartStage configuration
[pg@reagan pg]~>
```

To set a parameter in a controller, you first need to set it in CONFIGURATION mode.

AXIS 1 = longitudinal

AXIS 2 = transversal

The command structure is:

pg aperture command xxAAxx (/noanswer)

where: xx = axis (1 (longitudinal) or 2 (transversal))

AA = command, see pg aperture help

nn = value (use '?' to read a value back from the controller)

/noanswer : OPTION: do not check for a reply from the controller

To set the hysteresis compensation for axis 1 you would need to give the following commands:

```
> pg aperture command 1pw1 /noanswer
> pg aperture command 1bh0.02 /noanswer
> pg aperture command 1pw0 /noanswer
> pg aperture command 1bh?    ( will show the value )
```

the command 1pw0 may give some communication error, but it will work.

The new configuration can be checked with the command:

```
> pg info aperture parameters
```

To do the same with the transversal axis, replace the 1 by 2 in the above commands:

```
> pg aperture command 2pw1 /noanswer
> pg aperture command 2bh0.02 /noanswer
> pg aperture command 2pw0 /noanswer
> pg aperture command 2bh?    ( will show the value )
```

3.5.7 Diagnostics of the aperture controller

There is a very simple diagsmc program.

It can show info about the controllers and is handy to adjust the limits of the travels.

Errorstatus	Long	Trans
negativeEndOfRun	TRUE	TRUE
positiveEndOfRun	TRUE	TRUE
peakCurrentLimit	FALSE	FALSE
RMSCurrentLimit	FALSE	FALSE
shortCircuit	FALSE	FALSE
followingError	FALSE	FALSE
homingTimeOut	FALSE	FALSE
wrongESPStage	FALSE	FALSE
DCVoltageTooLow	FALSE	FALSE
outputPowerExceeded	FALSE	FALSE

LONGITUDINAL, status = NOT REFERENCED ESP stage error
TRANSVERSAL, status = NOT REFERENCED ESP stage error

Position = -0.000000_mm, -0.000000_mm

Press 'Esc' to return to main menu ...

It gives more or less the same information as the command "pg info aperture status" but is updated continuously.

3.5.8 Errors and status aperture controller

If for some reason the aperture controller is not moving, check the current status with the command:

```
pg info aperture status
```

Most errors are self-explicable but the positiveEndOfRun means that the mechanical limit switch has been detected. If, on the motor, you remove the small plastic cover over the scale, you will have a small Allen key on the far end. This small block determines the position where the limit switch is activated. Moving its position will give more/less travel.

It's VERY important that this is correctly set up during installation of the assy. A wrong setting of this switch together with a too high current limit might damage the assy or EPG system!

The status of the motors can show the following modes:

- NOT REFERENCED : the motor will not move anymore until its homed. Try to fix it with : "pg apert move home"
- CONFIGURATION: the controller is in configuration mode to set/change parameters. Leave the configuration mode with a "pw0" command or a "pg aperture move home"
- HOMING: stage is homing at the moment
- MOVING: stage is moving at the moment
- READY: stage is ready from HOMING, MOVING, DISABLE or JOGGING command.
- DISABLE: motors are disabled, encoders are still working tho!
- JOGGING: stage is in jogging mode (not used by us)

3.5.9 Disable the motors of the aperture controller

If, for some reason you would like to disable the motor(s), this can be set with a command.

Note: The commands will ONLY work if the controller status is READY.

In this case, the motors will be disabled, but the encoders are working! Its possible to move the motors by hand now!!

```
> pg aperture command mm0 /noanswer
```

will disable all motors.

```
> pg aperture command 1mm0 /noanswer
```

will disable motor axis 1 (longitudinal)

```
> pg aperture command 2mm0 /noanswer
```

will disable motor axis 2 (transversal)

To re-enable the motors, switch them from DISABLE to READY state with the commands:

```
> pg aperture command mm1 /noanswer
```

will re-enable all motors

```
> pg aperture command 1mm1 /noanswer
```

will re-enable motor axis 1

```
pg aperture command 2mm1 /noanswer
```

will re-enable motor axis 2

The only other way to totally disable the controller is to remove the power from the main controller! But by disabling only the motors, the encoders are still working, so you can still get positions of the motors in BEAMS.

3.6 Imaging

3.6.1 SEM scan generator

SEM imaging is done with a dedicated analogue SEM scan generator. A TV signal is generated and connected either to the SEM monitor or to a frame grabber. The latter is used to display the SEM image with e.g. xawtv on the Screen of the EBPQ control PC. This enables the user to display the SEM image also remotely. To start up the display use the command

```
> mon
```

This command will prevent the you to start up multiple monitors, because it this would slow down the PC.

With the UPG it is also possible to get images in SEM using the main/sub deflection instead of the analog frame grabber. This also brings some new options on how the scan is done.

3.6.2 Configuring SEM monitor parameters

The following parameters can be get and set with “pg get XXX” and “pg set XXX”. The GUI application “cmon” can be used to set and listen the following parameters: SEMMODE, SEMMAG, and SEM. The others have initial values defined in job.ini. The values in here are chosen for normal usage.

3.6.2.1 SEMMODE

This parameter is only available for systems with an UPG pattern generator!

This defines how the system scans over the substrate. Basically it let you choose in steps between “slow & accurate” and “fast & rough”. “Slow” means there are few image refreshes per second.

The values are:

0. **“Main Only”**

It always uses the Main Deflection to scan lines from left-to-right. It is accurate but slow.

1. **“Main Only Meander”**

Same as 0, but scans from right-to-left on the even lines, giving a small speed improvement and giving some extra information in case there are problems with the deflection configuration of the system (it will result in a clearly visible distortion between even- and uneven lines).

2. **“Sub Preferred”**

It will switch to Sub deflection as soon as the SEM image fits into one subfield. Causing it to be faster when zooming in enough.

3. **“Sub Preferred Meander”**

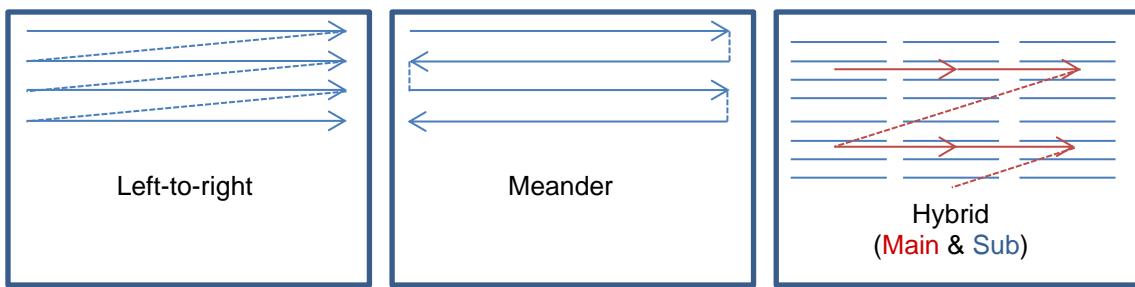
Same as 1 applied to the subfield.

4. **“Hybrid”**

It will combine main- and subfield to gain as much speed as possible. It will also give extra information in case there is a problem in the deflection configuration of the system.

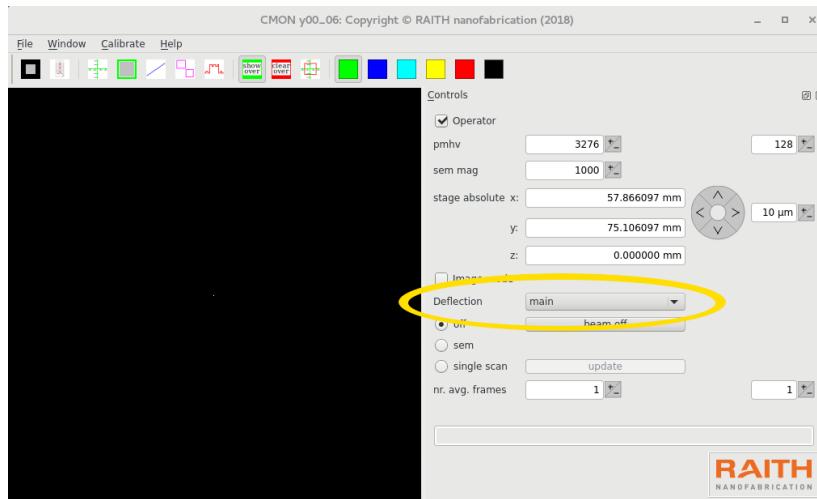
5. **“Hybrid Meander”**

It will meander in both main- and subfield. This is the fastest method but also the most sensitive towards deflection configuration (i.e. clearly visible distortions).



Notes:

- The system has a 'smearing' effect in its electron detection mechanism. This will be mostly visible in Hybrid mode.
- Semmode can also be set from the GUI in cmon:



3.6.2.2 SEMMBSBASE

This parameter is only available for systems with an UPG pattern generator!

Deflection scanning requires well-configured settling delays. If the settling delays are too small, the image will have distortions/displacements. If they are too high, the system will be needlessly slow. SEMBASE defines the Main Deflection base delay for every pixel.

3.6.2.3 SEMMBSFACTOR

This parameter is only available for systems with an UPG pattern generator!

It defines the Main Deflection delay relative to the distance between pixels

3.6.2.4 SEMSBSBASE

This parameter is only available for systems with an UPG pattern generator!

It is the same as SEMBASE but applied to Sub Deflection.

It is not applicable in SEMMODE 0 and 1.

3.6.2.5 SEMSBSFACTOR

This parameter is only available for systems with an UPG pattern generator!

It is the same as SEMFACTOR but applied to Sub Deflection.

It is not applicable in SEMMODE 0 and 1.

3.6.2.6 SEMMAG

It is the magnification factor of the scan. Larger magnification will also result in faster monitoring.

3.6.2.7 SEMRATE

This parameter is only available for systems with an UPG pattern generator!

It is the exponent that determines the capturing time for every pixel. Formula: $16 \text{ ns} \cdot 2^{\text{value}}$. It will affect the monitor speed considerably for higher values. 0 will be the fastest and will still give a decent image.

3.6.2.8 SEM

The state of the SEM monitor: 0 is “OFF”, 1 is “analog ON”, 2 = “digital ON (UPG only)”. By default, when using “cmon” to enable the SEM monitor, it will try to use digital if available, else it will switch to analog (i.e. “framegrabber”). If you start cmon with “cmon –analog”, it will always use the framegrabber.

3.6.3 pattern generator

Images can be also obtained using the pattern generator. These images exactly match with patterns, because the same deflection DAC’s are used. Either the main deflection or the sub deflection of the pattern generator is used for grabbing the image. Therefore the maximum size of the SEM image is determined by the maximum deflection range of the specific deflection.

The command to grab an image can take various parameters: origin (x,y) of the image in the main field, main DAC steps (x,y) per image pixel, number of image pixels (x,y) and finally the name of the image file. The maximum number of pixels, which can be grabbed is 1024x1024. Note that it is not mandatory to grab “square” images, nor the size in x or y as a power of 2.

The image will be saved as a raw data file and will have the extension .img. To store the conditions with which the image has been obtained a second file will be saved with the same name, but with extension .txt

The main deflection will be used by default. In case of usage of the sub deflection, the main deflection must be specified as a parameter.

The video level is a 16 bit unsigned value. It is possible to do sample averaging and/or frame averaging. Note that the parameter is the exponent of 2. Therefore the default value for both is 0, i.e. 1 sample per pixel and 1 frame per image.

The raw image can be most conveniently display using the program imagej. This is by default installed on the control PC. Note that you need to select 16 bit unsigned and little endian byte order for correct display.

3.7 Beam creation

3.7.1 Initial lens estimates

When you are starting from scratch, use the following initial lens setting @ 20 kV:

```
> pg set cc1 1600V
> pg set cc2 0.7A
> pg set f1 1.75A
```

Search for beam using the automatic tilt/shift adjustment

```
> pg set tlcur /binary 2047,2047
> pg set sftcur /binary 2047,2047
```

From the menu, plot a graph of the tilt vs. tilt with the shift current set to 2047,2047. Set the tilt current at the centre of the image. Plot the shift vs. shift. Optimise the tilt and shift currents in turn to the centre of the oval shaped beam on the graph

For LaB6:

```
> pg adjust beam hot /menu # select 1 and 2
```

For FEG: read BEAMS manual for ‘adjust beam align’ command

3.7.2 Coarse lens values and aperture alignment

Achieve a current of about 5 nA by setting CC1

```
> pg set cc1 <value>
```

At 20 kV, set the SEM magnification to 200

```
> pg set semmag 200
```

Over the marker plate, set CC2 and the final lens to achieve a focused image

```
> pg set cc1 <value>
> pg set f1 <value>
```

Locate and define the marker on the SEM-image.

```
> pg get tab
> pg marker cali 'tab'
```

Try to execute a marker search

```
> pg move marker
```

This might not succeed the first time. If not, improve the beam quality

Align the final aperture

```
> pg adjust final_lens # interactive command !
```

Adjust with the final lens mechanism to a minimum value

Rotation clockwise gives a positive increase.

Exit with 'CTRL_C'

LEFT KNOB	*	RIGHT KNOB	*
I *	0.16	*	-0.02
I *	0.14	*	-0.04
I *	0.13	*	-0.03
I *	0.12	*	-0.04
I *	0.12	*	-0.04
I *	0.12	*	-0.04
I *	0.12	*	-0.05
I *	0.11	*I	-0.05
I *	0.12	*	-0.04
I *	0.12	*	-0.04
I *	0.12	*	-0.04
I *	0.12	*	-0.04
I *	0.12	*	-0.04
I *	0.12	*	-0.04
I *	0.12	*	-0.04
I *	0.11	*	-0.04
I *	0.12	*	-0.04
I *	0.11	*	-0.04
I *	0.12	*	-0.04
I *	0.11	*	-0.04

```
Do you really want to abort adjust (default N) : y
```

Aborting command adjust

Restoring final lens value

Save the beam

```
> pg archive save beam <beam specification>
```

e.g.

```
> pg archive save beam first_beam
```

3.7.3 Column alignment

The column alignment can be done in a thorough way (when a new filament is installed) and a quick way (small adjustment after the system was shut down).

When a new filament is installed, you need to find a tilt/shift combination that gives some beam current. This has to be done manually or with the help of the adjust beam align command and making plots. There are millions of combinations for tilt and shift (for every tilt you can find a matching shift setting) but for only one combination you can create spots from pA to big spots without changing the alignment. The following explains how to get to that one setting.

Initial setup

The filament is run up to the correct values as shown in the datasheet.

Select the CC1 and CC2 values of an old 5 – 10nA spot. The CC2 value should be around 7000bits or so.

Search for some Beam current by changing Tilt or Shift in JOYSTICK while monitoring the Beam current. Write down the Tilt / shift values you found. You should have around 5nA now.

Put the marker in SEM with a magnification of about 500.

Decrease the CC2 value in steps of about 1000 bits. The screen should become brighter. If you loose intensity, compensate by changing the TILT! For the commands you can check below at chapter 4.3.2 SHIFT, but also include a tltcur /j /b in joystick for this time. (Alternatively you can set CC2 to '0' and run a TILT plot with the ABA command.)

Continue this until CC2 is set to 0 (binary).

Set the semmag to 1000. Now change the SHIFT setting in JOYSTICK. The marker should move and fade out. You do this in all 4 directions and then place the SHIFT to the center of the found values. (Write down all Shift values where the marker fades out and calculate the center). Write down the values of Tilt and Shift.

Restore the values of the 5nA spot, including the first Tilt/shift values and create a big spot.

Set CC2 to 12000 (binary). Increase CC1 to find the maximum Beam current. If the current fades away or decreases, check with TILT in Joystick if you are not drifting from the optimum. Adjust Tilt to maximum Beam current and then increase CC1 a bit. Continue until you get the maximum available beam current (normally between 350 and 450 nA). For the commands you can see the description below at chapter 4.3.1 TILT.

Probably the current Tilt/Shift values are completely different from what was found with the defocused small spot. The next step is to get the Tilt values to the ones we found with the defocused spot. Put BCM /m /f in a JOYSTICK window and change the TILT values a bit in the direction of the values where we want them to be. When the Beam current disappears, compensate by changing the shift values. Continue this until Tilt is set to the values we found with the defocused small spot. Write down the Tilt, Shift and CC1 values.

Go back to the CC1 and CC2 values of the 5nA spot (you should not have to change the Tilt/Shift values). Do an Adjust Final Lens.

From now on you can follow the method as described below, starting with SHIFT, and do about 3 iterations with the small and big spot. Then optimise with a big spot and the current over the field and all should be fine.

Tilt

Starting with a focussed beam, align the final aperture using the ‘pg adjust_final’ command. The goal is to get as much current as possible down the column. Begin at a very high value of CC2.

```
> pg move marker
> pg set cc2 /binary 12000          # typical value 12000 - 13000
> pg set cc1 1.6kV                 # start with the focused value
> semon
> pg display bcm /m /f
> pg display tltcur /b, pmhv, cc1, cc2 /b
```

With the joystick function, adjust the tilt current to maximize the current, using the BCM reading and the SEM-image to judge when the current is at maximum. Each time the SEM-image saturates, increase the PMHV to bring the marker back into view.

Increase CC1 slightly when the current is at maximum. This should give more current. Increase the PMHV again to compensate for CC1 and repeat the adjustment of the tilt current.

Repeat the whole process until “cross-over”, the point beyond which increasing CC1 will cause the current to drop. If everything goes well, the beam current should now be very high (> 400 nA @ 0.5 mA/sr). Record the binary tilt values.

Shift

Use CC1 to set the current back to about 5 nA. Create a defocused blob marker by setting CC2 to its minimum value. Set the SEM magnification to 1000.

If the current disappears when CC2 is set to ‘0’, the TILT is way off. You need to change TILT to get beam down with CC2 set to 0.

```
> pg set cc1 1.62kV                # focused value
> pg measure current
> pg get bcm /measure
> pg move marker
> pg set cc2 /binary 0
> semon
> pg display sftcur /b, pmhv, semmag, cc1, cc2 /b
```

Using the joystick, alter sftcur/b in one direction until the marker image begins to disappear as it is obscured by the aperture inner circumference.

Move the shift in one direction until the marker fades out in the SEM screen. Write down this value. Then move in the opposite direction until it fades out again. Average the 2 numbers and set the shift to this value.

Repeat this in the other axis.

Set the shift current to the values that were just found

Repeat the tilt and shift again until they are both set up as they are interdependent.

Create a focused spot via CC2 and FL. The final lens value should not differ greatly from previous machines. If it is much lower, the sensitivity must be changed by altering R80 on the FL board.

To check how good the column alignment is, change CC2/bin around 5000. As CC2/bin jumps and it degausses, there will be a noticeable movement in the position of the marker. The smaller this movement: the better the alignment. When the column alignment is perfect, this movement will be zero. Repeat this whole exercise for tilt and shift until the movement is reduced as much as possible.

To fine tune the Tilt/shift settings do the following:

- create a big spot (100 - 200nA)
- do a XMBC over the full field.
- Change TLTCUR a few bits in one direction
- Check current distribution again
- By changing the TILT a few bits, the current over the field should be almost perfect.

The found Tilt and Shift settings should work for ALL spots now!

If spots already exist, you need to save them all with the new TLT/SFT settings!

Quick Check of alignment

The easiest way to check if the basic alignment is ok, is to select a 5nA spot. Then do a MVM and put the marker in SEM. If you now set CC2 to '0' you can see if all is basically ok: If the screen is white (oversaturated) all is ok. If the screen goes black, the Tilt/shift combination is not the one optimal for the system.

The normal way to check if alignment is ok, is to select a 5-10nA spot and set the marker in SEM. If you change the CC2 binary by 5000 bits, the marker should not change position. If it moves visually, the alignment is not optimal.

Quick Optimisation after a filament shutdown

After the filament has been switched off and ran back up again, the alignment can change a bit. Normally the shift will not change a lot. Select a big spot (biggest one in use or 100-200nA) and perform a XMBC at the largest field. Optimise the result by changing the Tilt a few bits. When the result is optimal, use the current Tilt and Shift values for ALL the spots.

For big spots (approximately 50nA and more) it is advised to optimise them one by one by using the current over the field. A few bits change of TILT with big spots can have a big influence on the current distribution over the field!

To save the lens settings:

```
> pg archive save beam 4na
```

3.7.4 Conjugate beam blanking

To adjust the conjugate beam blanking, set sem on and move a corner of a marker to the centre of the image. Choose a SEMMAG between 20000 and 100000.

```
> pg move marker
> semon
> pg move position /relative 10,10
> pg set semmag 20000 # center marker corner at screen
> pg display sem, semmag, tab, pmhv, beam, cc1 /b, cc2 /b, ff, fl /b
```

Toggle between half blanking and non-blanking:

Change CC2 until the displacement between non- and half blanking is minimal. It should be less than 0.5 µm (preferably less than 100 nm). Focus with FF (or FL when the FF range is too small).

```
> pg set beam 0
> pg set bbshvadjust 500,7000          # values for half blanking
> sleep 2
> pg set beam 1
> pg set bbshvadjust 7000,7000
> sleep 2
```

3.7.5 Final lens setting

Once the CC1/CC2 combination is OK, adjust the FL value such that at zero height FF is 2047. Perform a pg adjust focus auto and check the FF value. Measure the height on the marker. FF should be height*12+2047. Align the aperture stick with

```
> pg adjust final_lens
```

command. Check that the column alignment and conjugation are still OK.

Check column set up:

To check the column set up, test the beam current and the beam diameter over the field (digital stigmator off) with

```
> xmfc 480,480 | 380,380 | 260,260
> xobq 480,480 | 380,380 | 260,260
> pg adjust beam hot /menu
> pg archive save beam <beam specification> # choose 1 and 2
```

3.7.6 Check column set-up

To check the column set-up, test the beam-current and spot over the field (digital stigmator off) with

```
> xmfc 480,480      # (20 kV)      380,380 (50 kV)      or 260,260 (100 kV)
> xobq 480,480      # (20 kV)      380,380 (50 kV)      or 260,260 (100 kV)
> pg adjust beam hot /menu          # select 1, 2
> pg archive save beam first_beam  # save this beam
```

3.7.7 Set up of a spot range

	CAUTION	The values found for shift and final lens must remain constant.
---	----------------	---

For 20 kV, it is possible that tilt and FF changes when the beam current changes. Increasing CC1 gives a bigger beam current. Focussing again is done with CC2. Create beam currents using CC1 and CC2, from 150 nA down to 50 pA, decreasing the beam current in steps of a factor 2.

Example set up:

Adjust CC1 to the desired beam current, refocusing the beam coarsely with CC2:

```
> pg adjust focus auto
fine focus deviates from optimum by -250 bits
> pg get height /measure
4.20 um
> pg information archive height
```

Note the value of the fine focus sensitivity (bits/μm). After and pg adjust focus auto, the EBPG5000 Plus will change the fine focus to account for the error in CC1. CC1 is set correctly when:

$$\text{Optimum fine focus} = 2047 + (\text{fine focus sensitivity} * \text{height})$$

$$= 2047 + (12.4 * 4.2)$$

$$= 2099$$

```
> pg get CC1 /binary
```

5600

The new value of CC1 can be estimated as:

$$\text{New CC1} = \text{Old CC1} + (0.6 * \text{fine focus deviation}) = 5600 + (0.6 * -250) = 5450.$$

```
> pg set CC1 /binary 5450
```

Repeat from pg adjust focus auto to make sure the fine focus is within tolerance. Recheck the column alignment and save the beam.

3.7.8 Checking angular intensity

Select an aperture of 600 mm or bigger and disable C2 with the command

```
> pg set lcr2 0
```

Measure the beam current:

```
> pg set beam 1
> pg display bcm /m /f
```

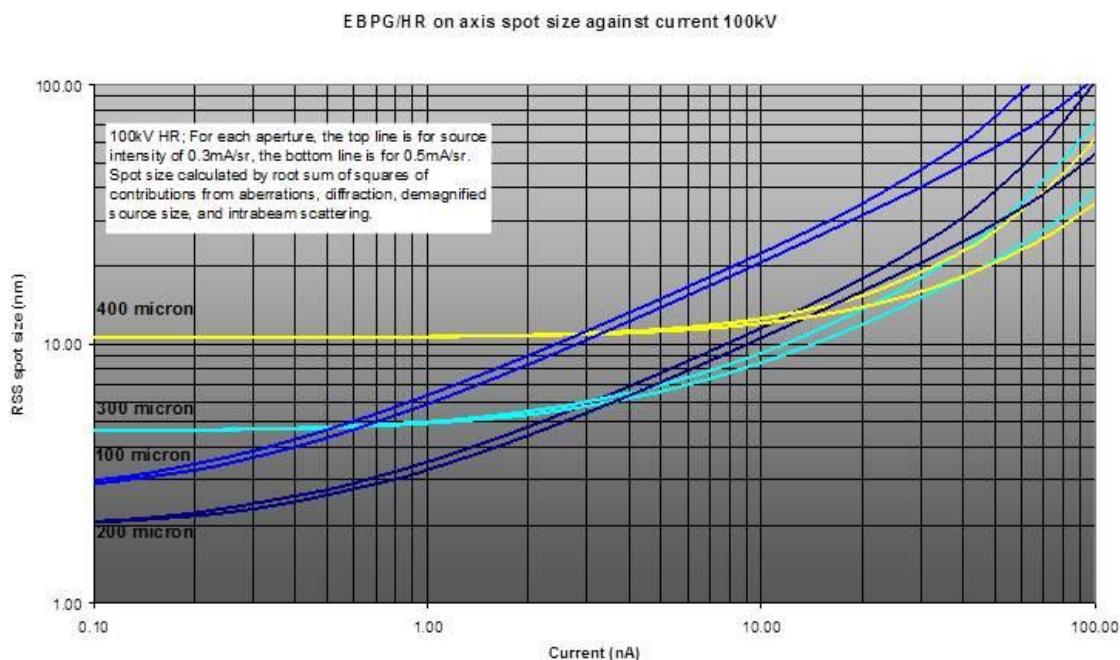
Use a second window and create the biggest beam current by increasing C1 while aligning the beam with:

```
> pg display cc1, tltcur /b, sftcur /b
```

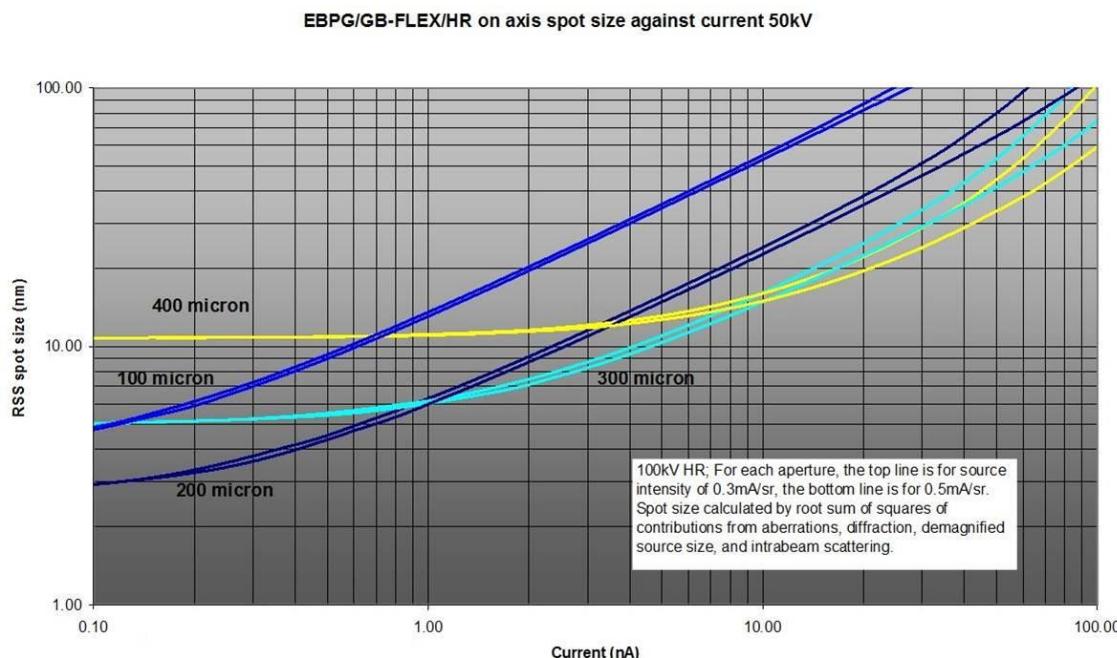
The biggest beam current you can achieve is the ANGULAR INTENSITY

3.7.9 Beam diameter versus beam current

The theoretical beam diameter as function of the beam current for 100, 200, 300 and 400 μm aperture



are shown below. Typical configuration of the tip uses a 0.4 mA/sr. opening angle.



3.7.10 Defocussed beam

Especially on FEG systems there can be a need to defocus the beam because the maximum beam diameter is only approximately 100 nm. Defocus is done by slight change of the final lens current. Each time the final lens setting has been changed a period of 30s is allowed for the final lens to stabilize. The BEAMS command to defocus the beam is

```
> pg adjust spot /defocus <diameter>
```

The default unit for the diameter is micron. When the beam is defocused a flag is set in the global data and pg information adjust epg will display that the beam is defocused.

pg adjust focus auto as well as recalibration during an exposure job will first remove the defocus setting, optimize the beam/calibrate the system and restore the defocus setting afterwards.

The defocus will be removed permanently with the command

```
> pg adjust spot /defocus /reset
```

Defocus of a beam is NOT saved with pg archive save beam. Also pg archive restore beam will remove any defocus.

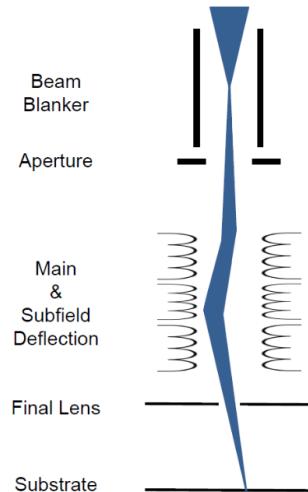
A typical command file using a defocused beam should look like:

```
> pg select pattern <pattern name>
> pg archive restore beam <beam specification>
> pg adjust focus auto
> pg adjust SPOT /defocus <diameter>
> pg adjust epg
> pg expose pattern
> pg adjust spot /defocus /reset
```

3.8 Settling time

3.8.1 Beam settling

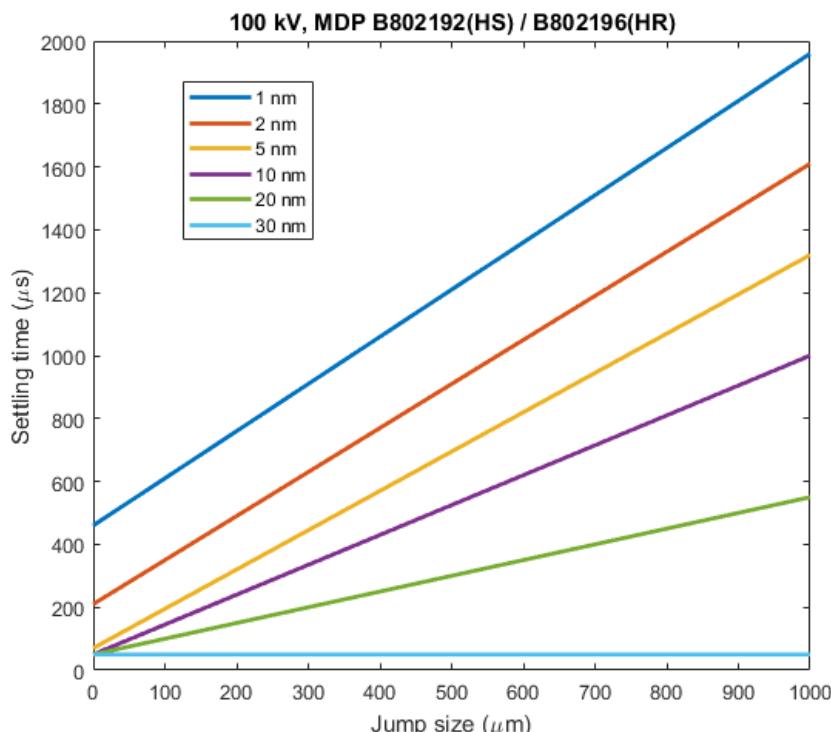
By moving beam from one position to another, the beam needs time to settle at correct coordinates. There are no active feedback loops in the system therefor time delays can be set by parameters.



The required settling time depends on the required accuracy of the position (placement accuracy). Usually this is a fraction of the used beam step size. For main field deflection the parameters are: MBSBASE, MBSFACTOR, MBSFOCUS, and for sub field deflections these are: SBSBASE and SBSFACTOR.

Main field

MBSBASE (ns)	Waiting time for every main deflection.
MBSFACTOR (ns/ μ m)	Waiting factor multiplied by deflection distance.
MBSFOCUS (ns/ μ m)	Focus factor multiplied by radius difference relative to center.
Main field settling time (ns) =	$\text{MBSBASE (ns)} + \text{MBSFACTOR} * \text{MAXxy}(r_1-r_2) + \text{MBSFOCUS} * \text{MAXxy}(r_1 - r_2)$



This plot shows the settling time to place first excel inside a range by different main field jumps. At x-axis cut is MBSBASE and direction coefficient is MBSFACTOR.

The more accuracy is needed, the longer the settling time needs to be. Remember the lines in this plot stand for the first excel of a shape. Depending on writing frequency, shape, smart fracture (like writing from inside out) this time can be reduced.

Table: Delay time to set for getting first excel in the accuracy range.

Accuracy	mbsbase [ns]	mbsfactor [ns/ μ m]
30 nm	50 000	0
20 nm	50 000	500
10 nm	50 000	950
5 nm	70 000	1250
2 nm	210 000	1300
1 nm	460 000	1500

Because the current systems use the digital correction tables to correct the focus with deflection, the MBSFOCUS have no significant effect on the spot placement.

Sub field

MBSBASE (ns)	Waiting time for every main deflection.
MBSFACTOR (ns/ μ m)	Waiting factor multiplied by deflection distance.
Subfield settling time (ns) = SBSBASE (ns) + SBSFACTOR * MAXxy(r ₁ -r ₂)	

Default

mbsbase	25000 ns	sbsbase	500 ns
mbsfactor	100 ns/ μ m	sbsfactor	500 ns/ μ m
mbsfocus	100 ns/ μ m		

3.9 Calibrations

3.10 Corrections

3.10.1 Height measurement

Height measurement is required to set the height dependant corrections for at a specific height of the substrate. The command to do this is

```
pg adjust height_comp
```

This command is also used as part of the exposure process, i.e. for each field, an adjust height compensation is executed.

There are various machine parameters available, which controls the behaviour of this procedure:

HGTINV	Substrate height compensation on or off, operates in conjunction with HGTMODE
HGTMODE	Selects behaviour after failure height reading
HGTRETRY	Number of retries when height measurement receives no interrupt within assigned time
HGTUSR	Returns the determined height from the height measurement procedure as outline in the Figure below
HGTAVG	Set optional height averaging around a mark
HGTZSTAGE	Compensate height with Z-stage instead of normal height compensation. Obviously this is only valid for systems with a Z-stage
HEIGHTOFFSET	Value added to the height reading
HMAPINV	Switch on height mapping. Use height mapping coefficients for height calculation, operates in conjunction with HMAPMODE. Only used when exposing
HMAPMODE	Indicates how the height map coefficients are used. Use of height mapping coefficients mode, operates in conjunction with HMAPINV
HMAPACCURACY	Decision level for using height map or measured height. If absolute value of measured minus calculated height is less than HMAPACCURACY then the measured height is used. Only used when exposing
HMAPCRITERION	Decision level for using height map or measured height. If measured height minus calculated height less than HMAPCRITERION times map sigma then the measured height is used. Only used when exposing
HMAPHIGHLIM	Decision level for using height map or measured height. If measured height less than HMAPHIGHLIM and greater than HMAPLOWLIM the measured height is used. Only used when exposing

HMAPLOWLIM Decision level for using height map or measured height. If measured height less than HMAPHIGHLIM and greater than HMAPLOWLIM the measured height is used. Only used when exposing

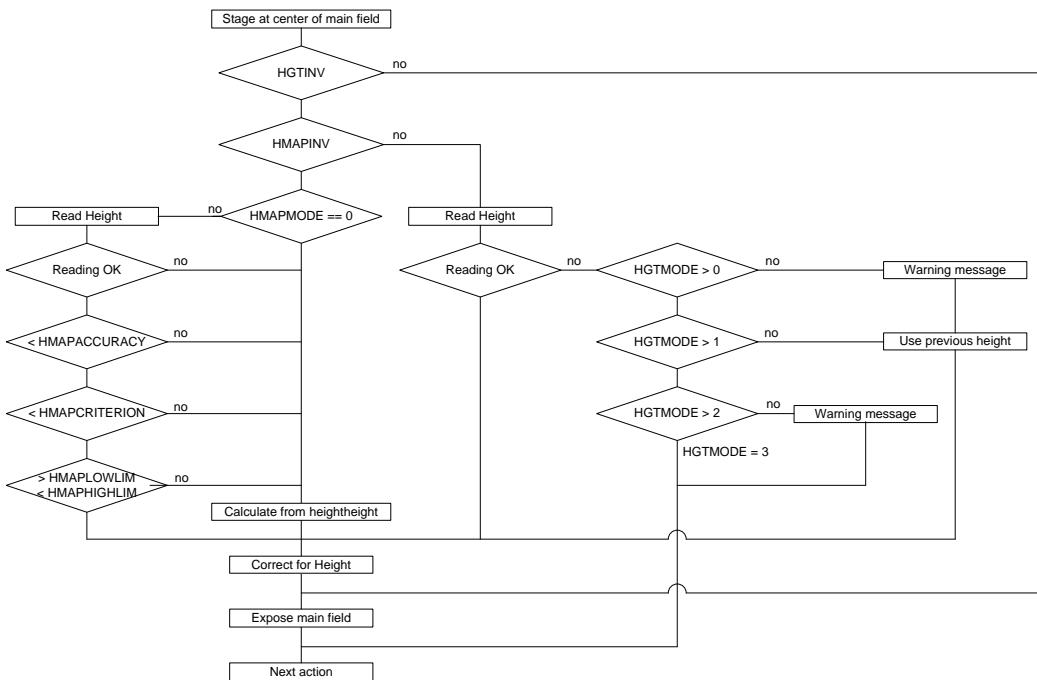


Figure 23 - Functionality adjust height compensation

3.10.2 Heightoffset

In Beams we have a machine parameter 'heightoffset'

The way this works, is as follows:

- During a calibration, the offset is NOT taken into account.
- During an adjust height comp, the offset is added to the determined height
- If you use the adjust height comp with a given height, the offset is NOT added..

The easiest way to see this is as follows:

You can switch on some extra heightoutput when you define : „export PG_PRINT_HTCOMP=1“

Here is an example of what happens when the heightoffset is set to '0' :

calibration:

```
[pg@reagan pg]~> pg adj eb 508
HTCOMP result:
height, height off-set ff : 3.0 0.0 2080
mg, mr : 34531,34707 32889,32396
tg, tr : 3029, 2884 2175, 2232
pg, pr : 31663,32444 30056,31894
```

[pg@reagan pg]~> pg move pos 100mm,100mm

Do an automated height compensation:

```
[pg@reagan pg]~> pg adj height comp
HTCOMP result:
height, height off-set ff : 19.9 0.0 2270
mg, mr : 35171,35343 33743,31535
tg, tr : 3059, 2915 2221, 2184
pg, pr : 31993,32765 30163,31783
```

Do a manual height compensation to a given height (10um):

```
[pg@reagan pg]~> pg adj height comp 10um
HTCOMP result:
height, height off-set ff : 10.0 0.0 2158
mg, mr : 34796,34970 33242,32039
tg, tr : 3041, 2897 2194, 2212
pg, pr : 31800,32577 30100,31848
```

Now we set the heightoffset to 10um:

[pg@reagan pg]~> mps heightoffset 10

Calibration:

```
[pg@reagan pg]~> pg adj eb 508
HTCOMP result:
height, height off-set ff : 3.0 10.0 2079
mg, mr : 34531,34707 32889,32396
tg, tr : 3029, 2884 2175, 2232
pg, pr : 31663,32444 30056,31894
```

The FF is still the same as above, no offset is used when calibrating!

[pg@reagan pg]~> pg move pos 100mm,100mm

Do an automated height compensation:

```
[pg@reagan pg]~> pg adj height comp
HTCOMP result:
height, height off-set ff : 29.9 10.0 2383
mg, mr : 35543,35715 34240,31031
tg, tr : 3076, 2932 2247, 2157
pg, pr : 32185,32953 30225,31718
```

The height is compensated for 29.9um instead of the 19.9um it was before!

Do a manual height compensation to a given height (10um):

```
[pg@reagan pg]~> pg adj height comp 10um
HTCOMP result:
height, height off-set ff : 10.0 10.0 2160
mg, mr : 34796,34970 33242,32039
tg, tr : 3041, 2897 2194, 2212
pg, pr : 31800,32577 30100,31848
```

The height is compensated for 10um, the offset is not being used now!

Do a manual height compensation to a given height (19.9um = real height for this position):

[pg@reagan pg]~> pg adj height comp 19.9um

HTCOMP result:

```
height, height off-set ff : 19.9 10.0 2272
mg, mr : 35171,35343 33743,31535
tg, tr : 3059, 2915 2221, 2184
pg, pr : 31993,32765 30163,31783
```

As you can see, the automatic adjust height comp will include the heightoffset, but if you enter a height, it's not being used!

If you enter it manually, you might as well take in account the offset when setting it.

3.10.3 Focus and stigmator table

3.10.4 Fine Focus Shift

When the Fine Focus is changed, this can also give a small movement of the beam position. During the “pg adjust ebpg” the Fine Focus shift is measured and coefficients are determined.

The following commands are available:

PG MEASURE FF_SHIFT [<nr_steps> <stepsize>]

Defaults are 7 measurements with 250 bit Interval.

During an adjust ebpg this is 5 measurements with a 240 bit interval.

PG INFO MEASURE FF_SHIFT

This will show the measured values, the correction coefficients and the corrections when the coefficients are used.

Machine parameter FFSHIFTINV

When set to 1, this will enable the stage position correction when an height compensation is done.

You can set the value with “pg set ffshiftinv <value>” or read the value with “pg get ffshiftinv”. Also the command “iep” (= pg information parameters) will show the current setting.

NOTE:

**This function is experimental for now! It has yet to be tested with actual exposures.
We don't know yet if this will improve exposures!! Can have unexpected side effects!**

3.10.5 Main field distortion table

This uses a lookup table in the DCV. To measure the digital corrections use the following sequence of commands.

```
> pg adjust ebpg <defl>,<defl>
> pg measure main distortion <defl>,<defl>
> pg info measure main
> pg adjust main distortion /digital /factor=<value>
```

This procedure should be repeated until the results are acceptable.

To activate the usage of the digital corrections table, use the command:

```
> pg set dcvdc 31,0
```

3.10.6 Stage correction for magnetic effects

3.10.7 Stage correction for orthogonality and gain

3.10.8 Stage distortion table

This correction table must be setup with a mask plate with a set of markers (same layout as rotcom plate), with well known positions (True Grid) obtained with an IPRO. The data from the IPRO must be available on the control PC.

The command to measure the plate is:

```
> pg measure stage distortions /file=<IPRO file>
```

Use the machine parameter TABDIST to switch usage of this table on / off.

4. MARKER SEARCH

4.1 Introduction

A marker is an object used to indicate a position. Markers are located on a holder, the stage (for a EBPG5200 only) and / or substrate. Markers on the stage and a holder are used for calibration of writing field and a marker on the wafer are mainly used to align the pattern to previous or future layer.

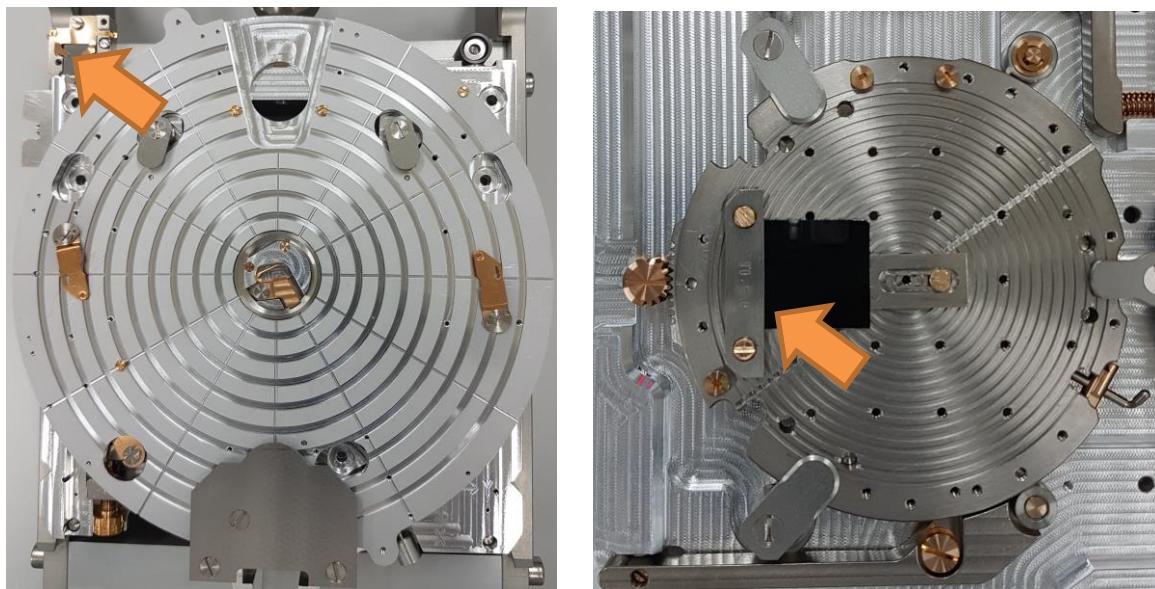


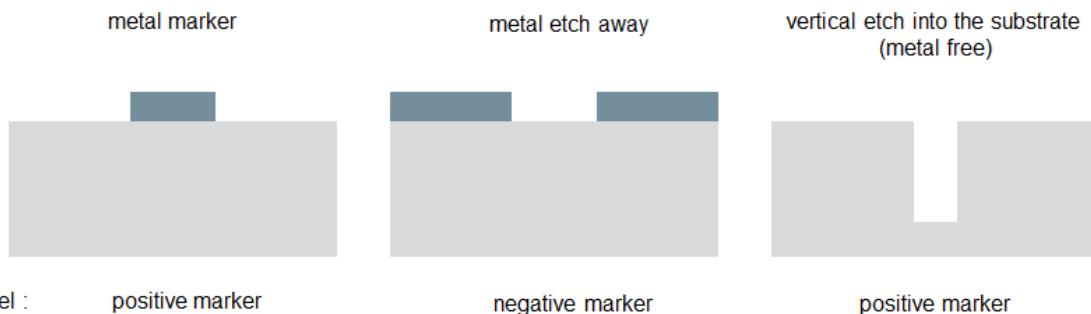
Figure 24 - Markers for calibration and alignment of the substrate.

In this chapter the diagrams are simplified and schematics are not to scale.

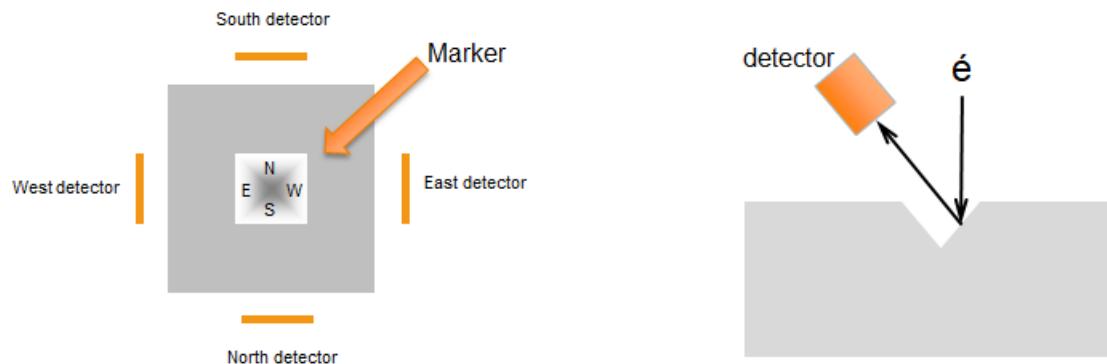
4.2 Type of markers

An EBPG can define 7 different types of markers; rectangle, cross, octagonal, topo, image, joy and PAM. The most common and standard marker of the EBPG is a rectangle type of marker in a square shape.

Rectangle, cross, octagonal and image markers can be detected by measuring the difference in video level (reflected signal) between marker and surrounding. When marker is brighter and has a higher video level than the surrounding then the marker is called a positive marker. If the marker is darker than the surrounding then marker is called a negative marker. This difference in video level can be created by using different atomic mass material for the marker and the surrounding or by etching vertical (4µm deep) into the substrate for metal free substrates.



Topo is a marker which has a rectangular shape where the slope of the marker has an angle of 54.7° . This can be made by etching a Si wafer with an orientation $<100>$ in KOH solution, this will create an upside down pyramid. The EBPG has 4 detectors which can be separately switched on and off to detect the marker side.



Joy marker gives the possibility to create users specified marker routine or ask for manual input. When joy marker is created it is possible to couple a script to this joy marker. When machine has to search for this marker it will run a script, if there is no script coupled to the joy marker, the EBPG will wait for an Enter press to continue. After the manual interference or at the end of script the EBPG expects that the stage is on centre of the marker.

PAM, Pre Alignment Marker, exists of an array of an odd numbers of markers, where from the centre the pitch increases by +1um. This will give one each marker an unique location in the matrix with respect to the neighbour markers.

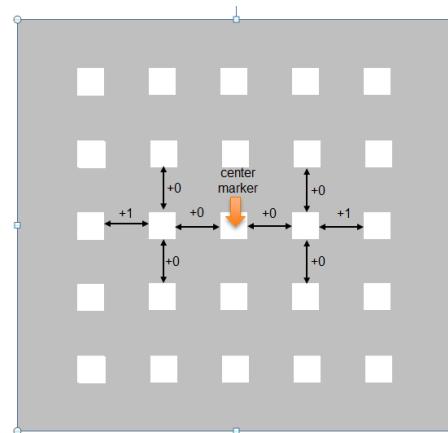


Table: Summary marker type

Type	Signal	Method	Use for
rectangle	Mass difference or	search edges and calculate center	Alignment, Calibration and

	etch perpendicular		Height measurement
cross	Mass difference or etch perpendicular	search edges and calculate center	Alignment
octagonal	Mass difference or perpendicular edge	search edges and calculate center	Alignment
topo	Edge of 54.7° angle	search slope and calculate center	Alignment
image	Mass difference	scan image and correlation	Alignment
joy		Script or manual interference	Alignment
PAM		Array of markers	Pre Global Alignment

4.3 Marker signal

The EPG has 4 detectors which can receive reflective electrons of the surface. The amount of electrons will be transferred to video level, the more electrons hit the detector the higher the video signal gets.

The amount of backscattering electron depends on atomic mass, thickness and surface roughness.

Atomic mass

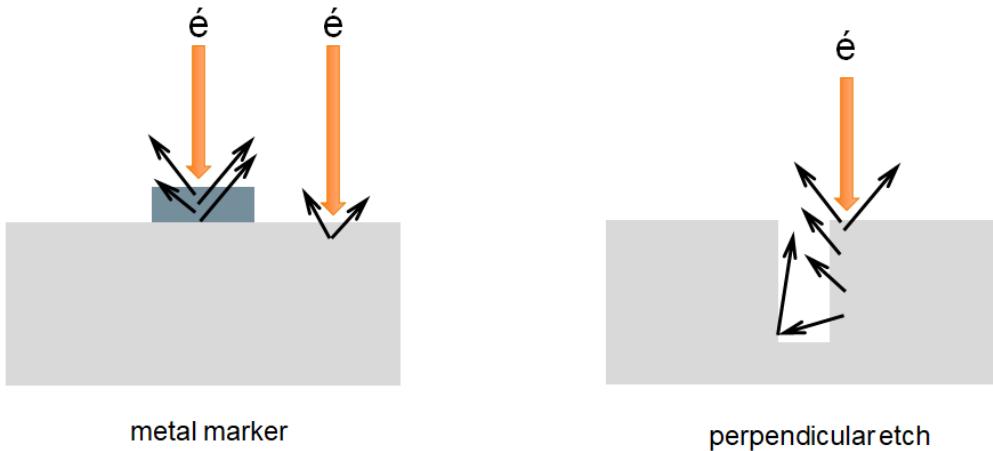
When the electron beam hit the substrate with 50kV or 100kV, most electrons go into the substrate. A few electrons are scatter back out of the surface and hit the detector. If the beam hits a material with a higher atomic mass then more electrons are scattering back. Silicon (Si) has atomic weight of 28u, Chromium (Cr) 52u, Tungsten (W) 184u, Platinum (Pt) 195u, Gold (Au) 197u and Aluminum (Al) 27u, check 'Periodic Table of the Elements' for more Atomic Weight. A marker of Aluminum on a Silicon substrate does not work because the mass difference between Aluminum and Silicon is -15%. This is too small and not enough electrons will be scattered back. A marker of Tungsten, Platinum or Gold on Silicon substrate has a mass difference of 5.75 till 6.16x Silicon. A marker of Chromium is 1.6x can work.

Thickness

The backscattering is also depending of the thickness of the marker. At 100kV high tension most electrons go into the substrate. By making a marker layer thicker there is more chance that an electron hits a heavy atom and is scatter back. For a marker of Tungsten, Platinum or Gold a thickness of 50-80nm or more is suggested. Suggestion of a chromium marker a thickness of 100-200nm or more is suggested. It's also depending on the substrate material and layer stack.

Surface roughness

At rough surface the surface area will be larger, there is more chance that an electron will leave the substrate. By etching into the substrate, locally the surface will be larger. The largest effect is achieved if etch in to the substrate has a perpendicular slope. For Silicon substrate the marker depths should be 4um or more. The benefit is that this process is metal free but metal markers are better to detect.



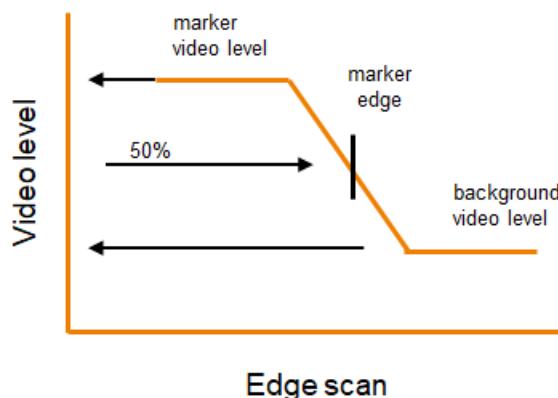
Edge

Marker search is based on detecting the centre of the marker slope.

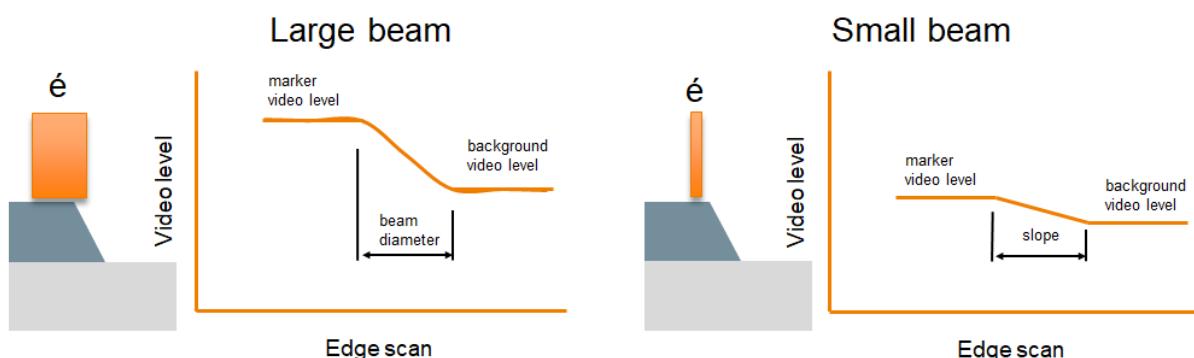
Marker can be detected by difference in video level between the marker and the background. Location of the marker is determined by measuring the edges of the markers and calculating the centre of the marker.

The edge of the marker for the EBPG is where the video signal level is in the middle between background and marker video level at 50%. 50%level will be calculate of by $(\text{video level background} + \text{video level marker}) / 2$.

$$50\% \text{ level} = (\text{background level} + \text{marker level}) / 2$$



The beam diameter has no influence on the location of the marker, only on the high of the video level.



The borders of the marker need to be a straight line. The roughness at the border will influence position and the alignment accuracy.

The best performance is achieved when the marker has a perpendicular slope.

4.4 Adjustable parameters for marker search

The parameters can be get with enter the command `pg INFO MARKER <marker ident> /test=par`

The parameters can be set with the command MARKER SET:

FORMAT	<code>pg marker set <marker ident> <parameter name> <value></code>
PARAMETERS	<p><i>Marker ident</i>: specifies the marker table for which the parameters has to be modified. It is also possible to set a parameter for all markers by typing an asterisk ("*") for marker table.</p> <p>Parameter name: specifies the parameter to be modified. This name must be one out of a marker search table.</p> <p>Value : integer; specifies the new value for the specifies parameter</p>
DESCRIPTION	Set marker search parameters. The names of the marker search parameters that can be specified for par_name are listed below.

Marker search parameters:

name	description	unit	value
contra	The maximum allowed contrast of video levels (negative/positive)	percentage	default: 90 range: [40,99]
mlvtol	Allowed tolerance on video level measurement on marker	dac	default: 150 range: [40,200]
blvtol	Allowed tolerance on video level measurement on background	dac	default: 150 range: [40,200]
eplmax	Edge positive video level measurement: max. of allowed range	dac	default: 1000 range: [800,1020]
eplmin	Edge positive video level measurement: min. of allowed range	dac	default: 300 range: [200,700]
hvalev	required video level for pmhv adjustment	dac	default: 650 range: [500,800]
msxtol	Allowed marker width tolerance in % of marker width. Set to 0 if not search in x-direction	percentage	default: 20 range: [0,40]
msytol	Allowed marker height tolerance in % of marker height. Set to 0 if not search in y-direction	percentage	default: 20 range: [0,40]
srcdir	Edge search direction specification: 1 = along y axis (vertical)		default: -1 range: -1 or 1

	0 = along x axis (horizontal) -1 = along x and y		
fserr	Maximum allowed position in which we find an error due to noise influence in the edge search routine in nm	nanometer	default: 20 range: [10,50]
israd	Initial search radius in micron used by find marker function, multiplied by factor 3 by move to marker function	nanometer	default: 50 range: [10,999]
isxstp	Step size in x-direction in percentage of marker width (for spiral search)	percentage	default: 30 range: [10,80]
isystp	Step size in y-direction in percentage of marker height (for spiral search)	percentage	default: 30 range: [10,80]
csixst	Initial step size in x-direction in percentage of marker width (for coarse search)	percentage	default: 15 range: [5,50]
csiyst	Initial step size in y-direction in percentage of marker height (for coarse search)	percentage	default: 15 range: [5,50]
n fsm	Number of measurements to perform along an edge for edge fine search algorithm		default: 20 range: [1,25]
nnlvm	Number of measurements to perform along an edge for noise rms video levels (edge fine search algorithm)		default: 1 range: [1,5]
nmvlm	Number of measurements to perform along an edge for marker video levels (edge fine search algorithm)		default: 1 range: [1,5]
nblvm	Number of measurements to perform along an edge for background video level (edge fine search algorithm)		default: 5 range: [1,25]
nelgm	Number of measurements to perform along an edge for edge (slope) length measurement (edge fine search algorithm)		default: 5 range: [1,25]
mtmacc	Required accuracy in nanometer (move marker)	nanometer	default: 20 range: [20,250]
mtmatt	allowed max number of attempts to		default: 10

	move marker to		range: [5,20]
edsx	positive edge distance along the edge (x or y) and around the edge middle position, to perform the video level, noise rms level and edge fine search measurements. (percent of marker size)	percentage	default: 30 range: [0,90]
edsy		percentage	default: 30 range: [0,90]
eshx	positive shift from marker edge to measurement position (x or y) to perform	percentage	default: 20 range: [5,50]
eshy		percentage	default: 20 range: [5,50]
cmerot	maximum allowed edge rotation in arcminute	Arcminute	default: 120 range: [20,200]
cmevstd	roughness standard deviation in nm	nanometer	default: 17 range: [2,25]
cmnnlm	Number of measurements to perform along an edge for noise rms video levels		default: 10 range: [1,25]
cmnmlm	Number of measurements to perform along an edge for marker video levels		default: 25 range: [1,25]
cmnblm	Number of measurements to perform along an edge for background video levels		default: 25 range: [1,50]
cmnem	Number of measurements to perform along an edge for edge fine search measurements		default: 25 range: [1,25]
cmnep	Number of positions along the edge for perform video level fine search		default: 64 range: [1,64]
cmedsx	positive edge distance along the edge (x or y) and around the edge middle position, to perform the video level, noise rms level and edge fine search measurements. (percentage of marker size)	percentage	default: 0 range: [30,150]
cmedsy		percentage	default: 0 range: [30,150]
cmeshx	positive shift from marker edge measurement position (x or y) to perform marker or background video level and noise rms level measurements (percentage of marker size)	percentage	default: 20 range: [5,50]
cmeshy		percentage	default: 20 range: [5,50]
cmfunc	check marker function handling: 0 = "normal" customer printout 1 = extended printout for test 2 = measure/print rms noise levels 3 = measure/print marker video levels 4 = measure/print background video levels 5 = measure/print edge positions		default: 0 range: [0,5]

timmsm	Time measurement switch used to measure the execution times of the find marker function algorithms: 0 = "normal" complete find marker 1 = stop before initial search for the marker 2 = stop after initial search for the marker 3 = stop after marker centre coarse search 4 = stop after pmhv adjustment 5 = stop after positive level initialisation 6 = stop after marker centre fine search there is no result storage if not zero		default: 0 range: [0,6]
detset	Marker edge search electron detector setting for test purposes *e = edge ,*d = detector choice: 0 = default setting for marker type 1 = all detectors 2 = we=wd,ee=ed, ne=nd,se=sd 3 = we=ed,ee=wd, ne=sd,se=nd 4 = we=ee=nd+sd , se=ne=wd+ed 5 = we=ee=wd+ed , se=ne=nd+sd 6 = we=ee=ne=se= west detector 7 = we=ee=ne=se= east detector 8 = we=ee=ne=se= north detector 9 = we=ee=ne=se= south detector		default: 0 range: [0,9]
beamon	beam on or off during the marker search (find marker / check marker properties) true/false = 1/0	boolean	default: 1 range: 1 or 0
edgsim	marker centre fine search edge simulation simulate east and south edge if true: east = west + marker size.x south = north – marker size.y true/false = 1/0	Boolean	default: 0 range: 1 or 0
fssstp	Fine step in main resolution units		16 bit systems: 1 20 bit systems: 8
csfstp	Coarse step in main resolution units		16 bit systems: 1 20 bit systems: 8

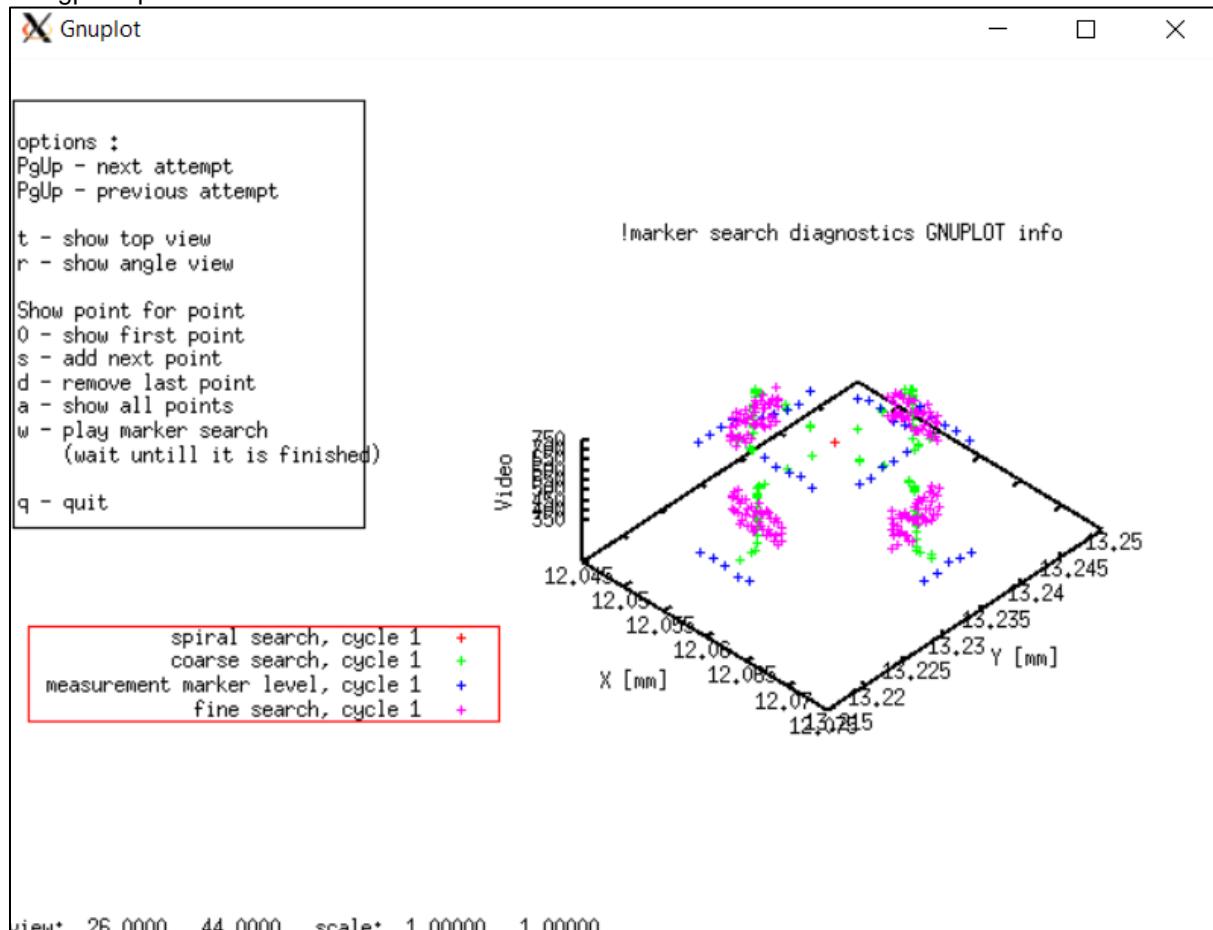
ind	name	val	def	ulim	llim	unit	PM marker level	Spiral search	Course search	Measure marker level	Noise Filter	Fine search	Machine related
1	contra	85	95	99	10	proc	X						
2	mlvtol	150	150	200	40	dac							
3	blvtol	150	150	200	40	dac	X						
4	eplmax	1000	1000	1020	800	dac							
5	eplmin	300	300	700	200	dac							
6	hvalev	650	650	800	500	dac	X						
7	msxtol	20	20	40	0	proc		X					
8	msytol	20	20	40	0	proc		X					
9	srkdir	-1	-1	1	-1		X	X			X		
10	fserr	20	20	50	10	nm				X			
11	israd	50	50	999	10	um	X						
12	isxstp	30	30	80	10	proc	X						
13	isystp	30	30	80	10	proc	X						
14	csixst	15	15	50	5	proc		X					
15	csiyst	15	15	50	5	proc		X					
16	n fsm	20	20	25	1						X		
17	nnlvm	1	1	5	1					X			
18	nmlvm	5	5	25	1				X				
19	nblvm	5	5	25	1			X					
20	nelgm	5	5	25	1					X			
21	mtmacc	20	20	250	1	nm							
22	mtmatt	10	10	20	5								

ind	name	val	def	ulim	llim	unit	PM marker level	Spiral search	Course search	Measure marker level	Noise Filter	Fine search	Machine related
23	edsx	30	30	90	0	proc		X	X	X			
24	edsy	30	30	90	0	proc		X	X	X			
25	eshx	20	20	50	5	proc		X	X				
26	eshy	20	20	50	5	proc		X	X				
27	cmerot	120	120	200	20	amin							
28	cmevsd	17	17	50	2	nm							
29	cnnnlm	10	10	25	1								
30	cnnnlm	25	25	50	1								
31	cnnblm	25	25	50	1								
32	cmnem	25	25	25	1								
33	cmnep	64	64	64	1								
34	cmedsx	30	30	150	0	proc							
35	cmedsy	30	30	150	0	proc							
36	cmeshx	150	150	1000	0	proc	X						
37	cmeshy	150	150	1000	0	proc	X						
38	cmfunc	0	0	5	0								
39	timmsm	0	0	6	0								
40	detset	0	0	9	0								
41	beamon	1	1	1	0								
42	edgsm	0	0	1	0								
43	fsstp	8	8	32	1						X		
44	csfstp	8	8	32	1						X		

4.5 Marker routine

Marker search consist out different modules. To see the routines which are used in the last marker search and the video levels, type on the command line diagmark or diagmark -p for a plot.

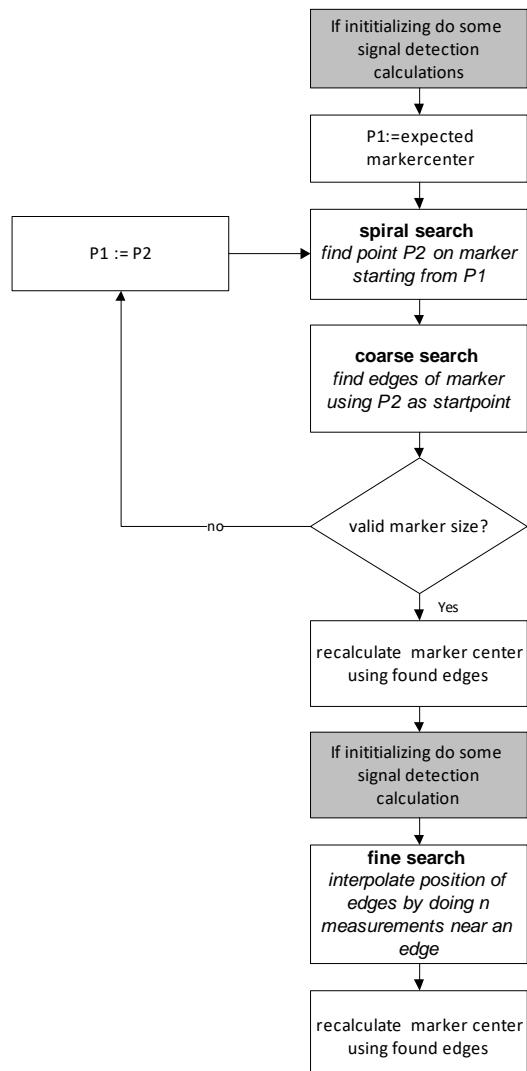
```
>diagplot -p # show last marker search
```



Modules:

- Adjust PM
- Spiral search
- Coarse search
- Measurement marker level
- Measurement marker slope
- Measurement video noise level
- Fine search

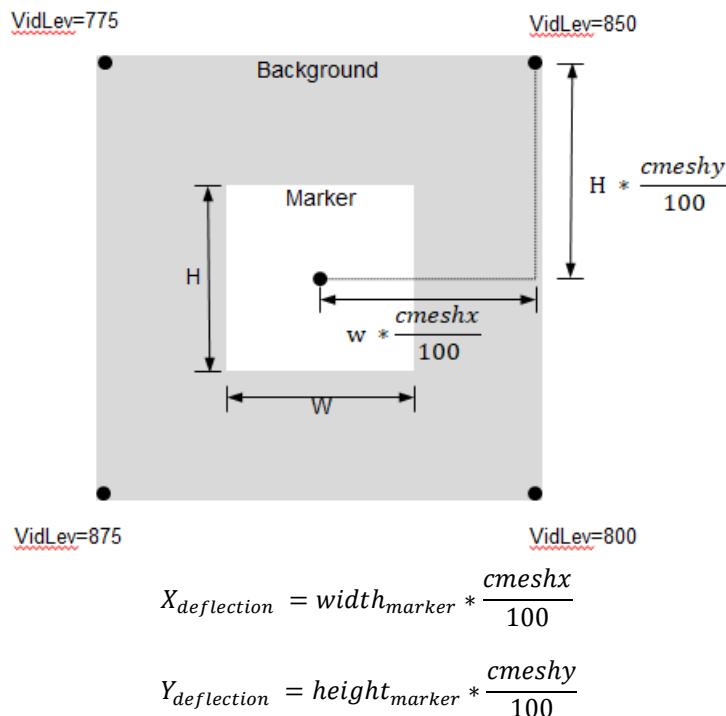
Depending on the history and measure result BEAM can skip or redo modules.

**Figure 25 - Schematics marker search algorithm**

4.5.1 Adjust PM

Adjust the background video level and calculate video level for marker and slope.

- Calculate the position where the background video level will be measured.



- Measure on 4 deflected positions the video level and select the position where the second highest or second lowest respectively if the marker level is positive or negative.
- Calculate acceptable range of video level.

$$\text{positive marker} \quad \begin{cases} \text{upper limit} = hvalev * \frac{\text{contrast}}{100} + blvtol \\ \text{lower limit} = hvalev * \frac{\text{contrast}}{100} - blvtol \end{cases}$$

$$\text{negative marker} \quad \begin{cases} \text{upper limit} = hvalev + blvtol \\ \text{lower limit} = hvalev - blvtol \end{cases}$$

- Change pmhv value so that the video level of selected position is in the upper and lower limit.
- Measure the background video level
- Calculate threshold marker video level for positive or negative marker

$$\text{positive marker} \quad \text{threshold } Marker_{video\ level} = \text{measured } Background_{video\ level} * \frac{100}{\text{contrast}}$$

$$\text{negative marker} \quad \text{threshold } Marker_{video\ level} = \text{measured } Background_{video\ level} * \frac{100}{\text{contrast}}$$

- Calculate threshold edge video level

$$\text{target } Edge_{video\ level} = \frac{\text{measured } Background_{video\ level} + \text{threshold } Marker_{video\ level}}{2}$$

Adjustable parameters

name	Description	unit	Value
contra	Maximum allowed contrast on video levels (negative or positive)	perc.	default: 90 range: [40,99]
blvtol	Tolerance video level on background video level (hvalev)	dac	default: 150 range: [40,200]
hvalev	Target video level for pmhv adjustment	dac	default: 650 range: [500,800]
cmeshx	Horizontal shift to measurement position in procentage of the marker width	perc	default: 150 range: [150,1000]
cmeshy	Vertical shift to measurement position in procentage of the marker height	perc	default: 150 range: [150,1000]

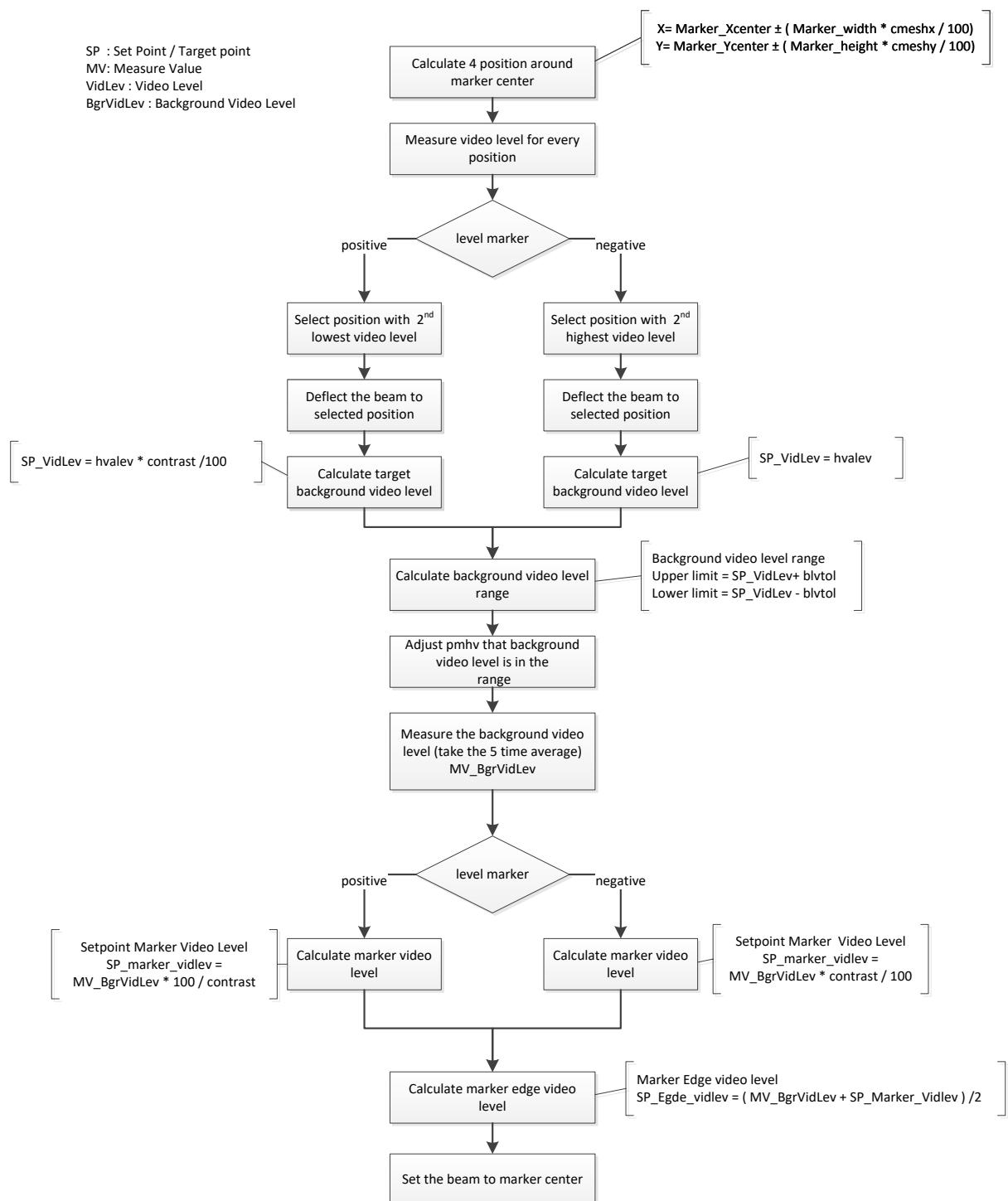
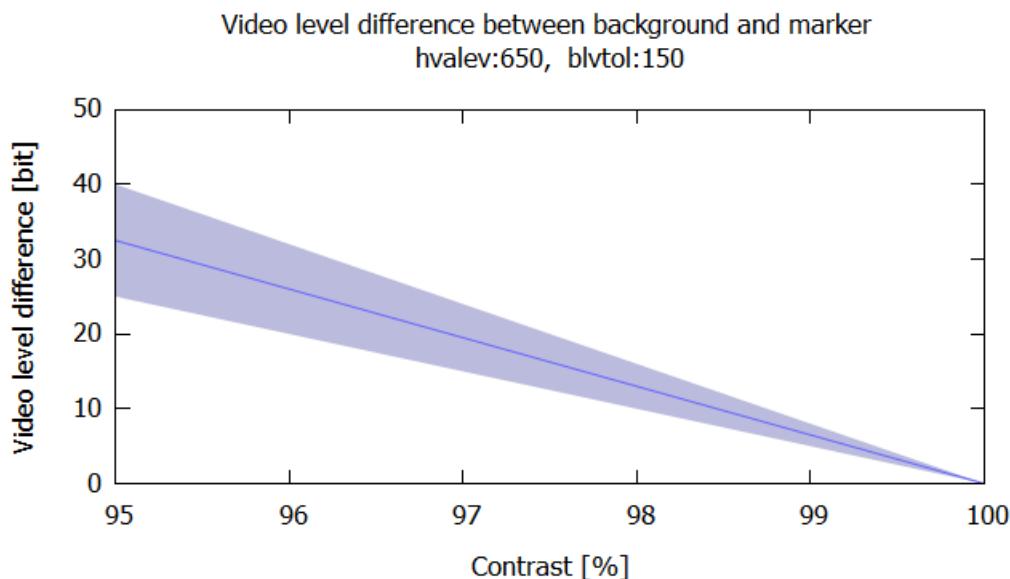


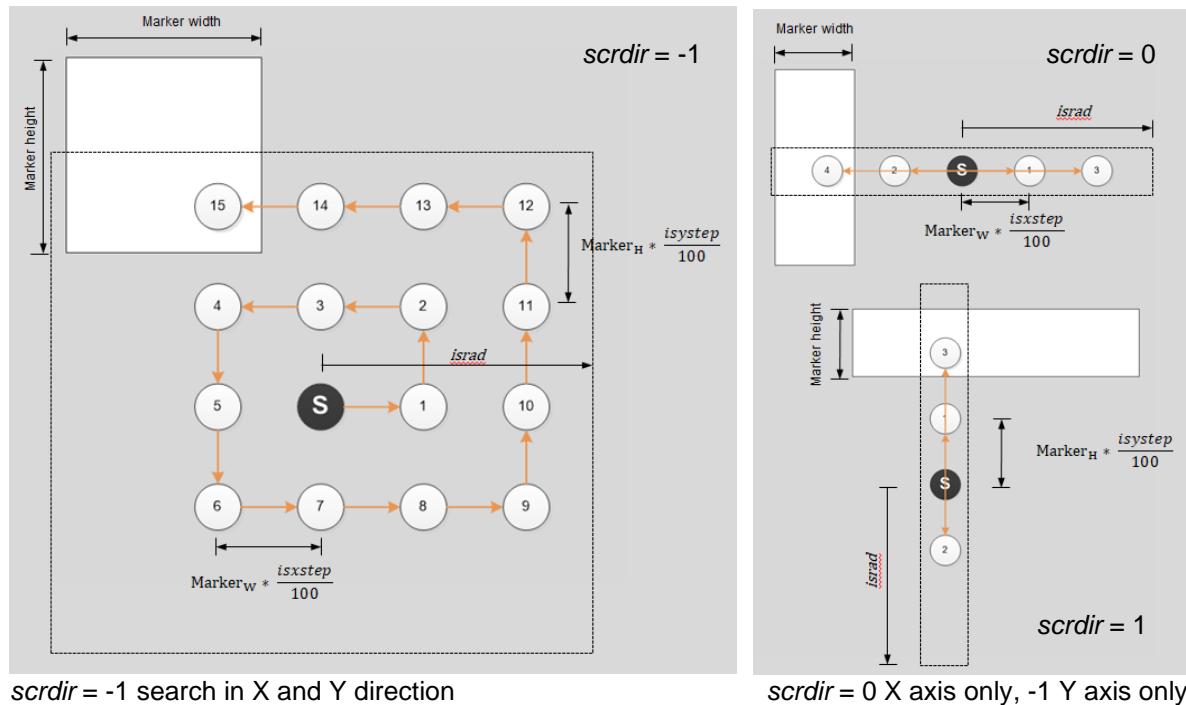
Figure 26 - Schematics calculation procedure



When the contrast is higher than difference between background and marker threshold value will be lower. In the spiral search routine, mains when the video level is higher than threshold value then search spiral search will pause and coarse routine will be started. Coarse will be to search for the edge but if threshold in the value is so low that the video (background) noise is higher then machine can not find the edge. An error will be occur and stop the routine.

4.5.2 Spiral search

Search around where the video level is higher than the threshold marker video level which is calculated at PM marker level.



scrdir = -1 search in X and Y direction

scrdir = 0 X axis only, -1 Y axis only

1. Beam is located undeflected on top of point S.
2. Calculate the step size

$$Xstep = \text{Marker}_{width} * \frac{\text{isxstep}}{100}$$

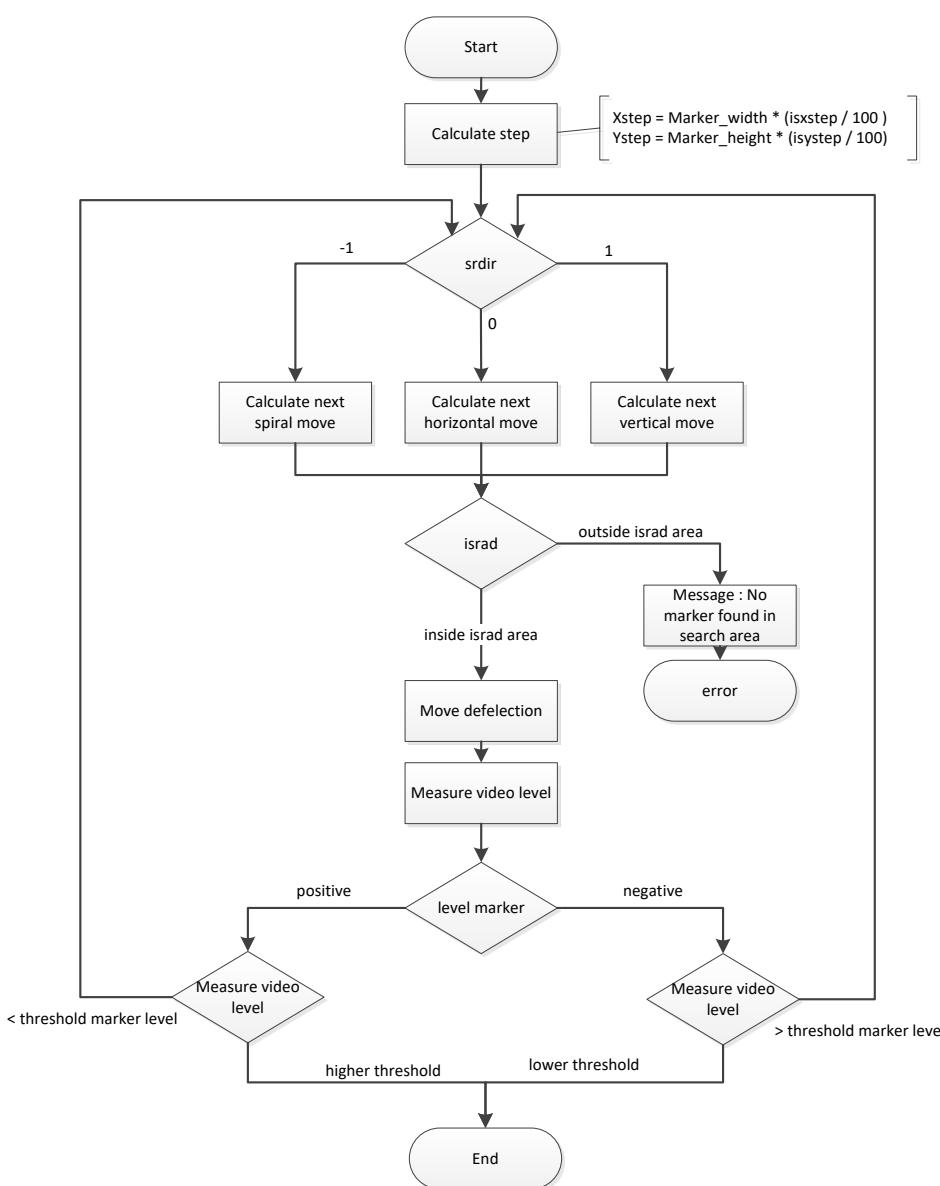
$$Ystep = \text{Marker}_{height} * \frac{\text{isystep}}{100}$$

3. Step to next exel,
 - If scrdir = -1 then in spiral movement around the start position (default).
 - If scrdir = 0 then in horizontal movement around the start position.
 - If scrdir = 1 then in vertical movement around the start position.
4. Measure the video level
5. Check if the video level is higher for a positive marker or lower for a negative marker than threshold marker level then go to next module, coarse search.
6. If the next position is in size the search radius (*israd*) then redo from step 2 else message "marker not found in search area" and stop.

Adjustable parameters

name	description	unit	value
israd	initial search radius in micron used by find marker function, multiplied by factor 3 by move to marker function	nm	default: 50 range: [10,999]

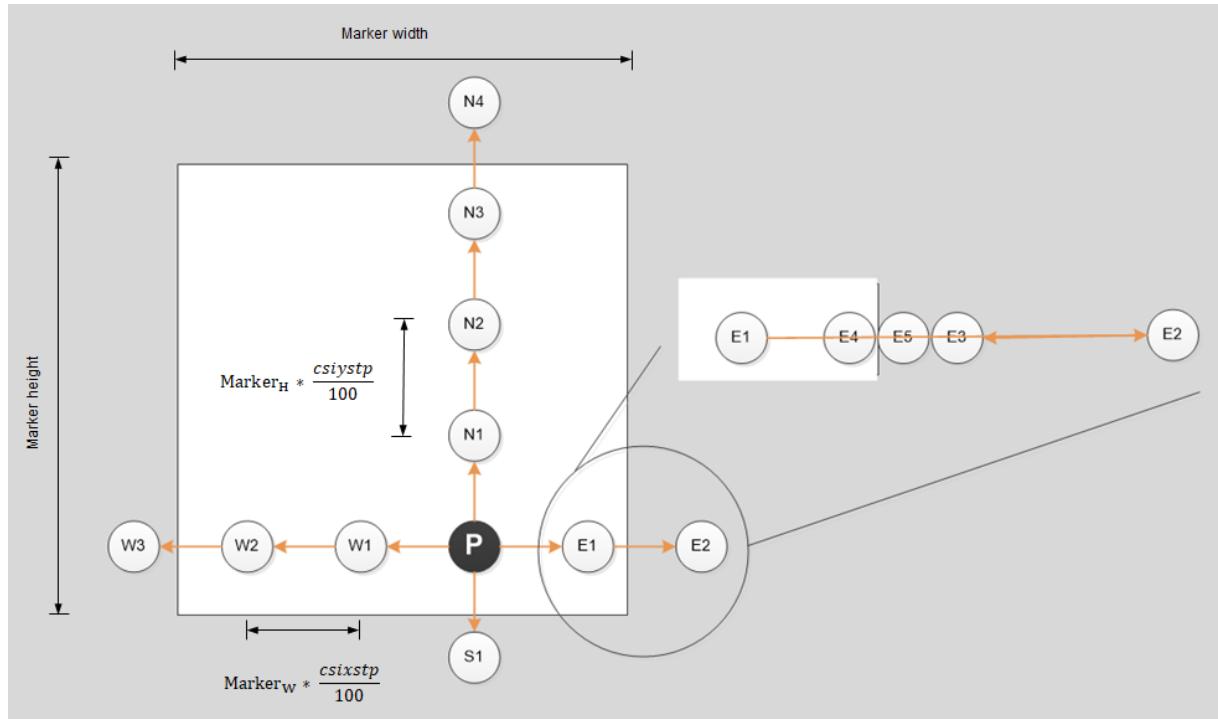
isxstp	Step size in x-direction in percentage of marker width (for spiral search)	perc.	default: 30 range: [10,80]
isystp	Step size in y-direction in percentage of marker height	perc.	default: 30 range: [10,80]
srcdir	Edge search direction specification: -1 = along x and y 0 = along x axis only 1 = along y axis only		default: -1 range: [-1 , 1]



4.5.3 Coarse search

Coarse search will search for the edge of the marker and check if the marker size is within tolerance.

Searching is done in two steps. The first step will move away from start in starting point P and find that the video level is the same as the background level. The second step then finds the marker edge by stepping forward and backward while decreasing the step size every time by 50% until the target edge level is reached.



1. Beam is located deflected on top of point P
2. Calculate the step size

$$Xstep = \text{Marker}_{\text{width}} * \frac{\text{csixstp}}{100}$$

$$Ystep = \text{Marker}_{\text{height}} * \frac{\text{csiystp}}{100}$$

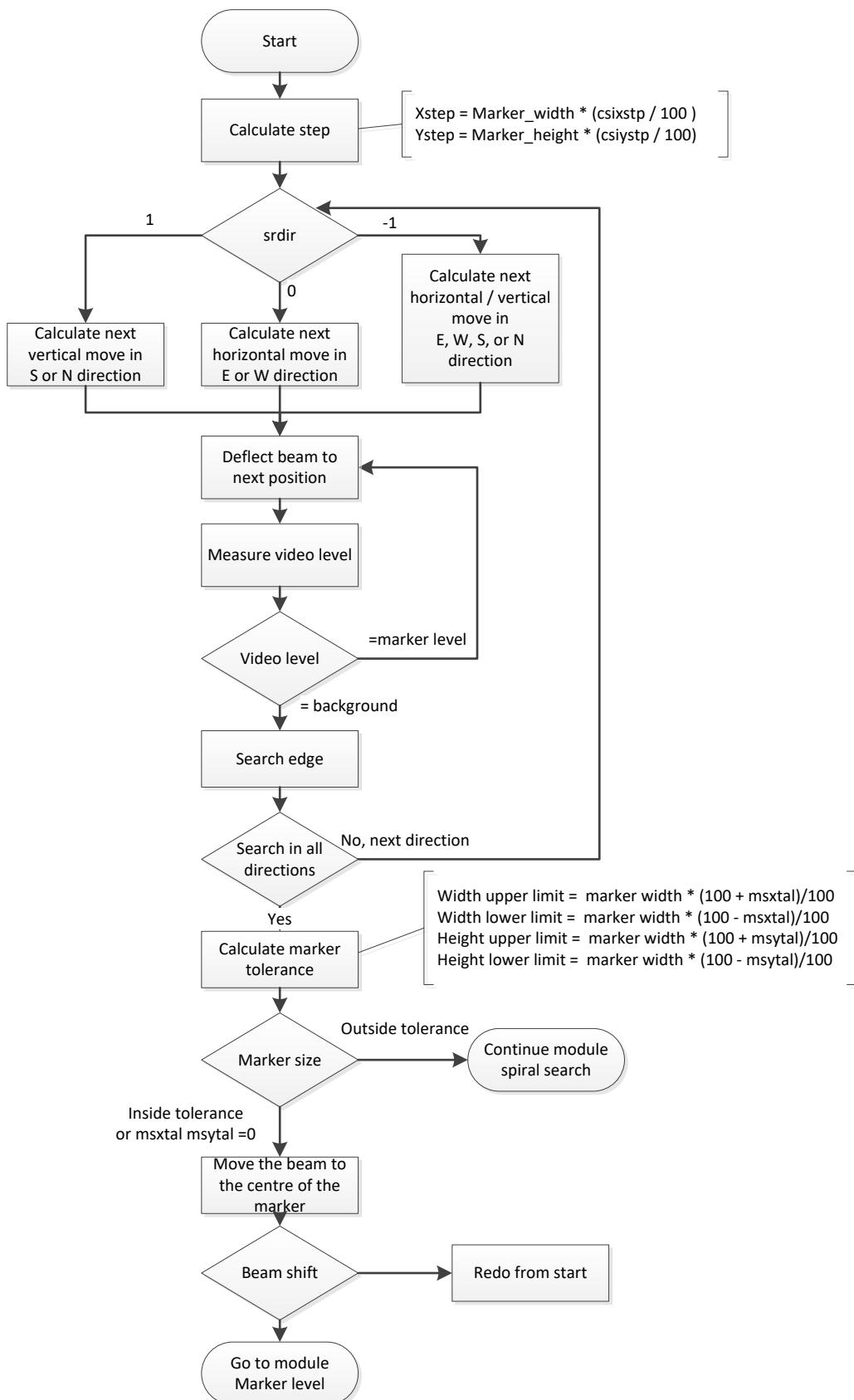
3. Step direction,
 - If scrdir = -1 then move in horizontal or vertical movement in east, west, south or north direction from the start position.
 - If scrdir = 0 then move in horizontal movement in east or west direction from the start position.
 - If scrdir = 1 then move in vertical movement in south and north direction from the start position.
4. Deflect to the next position
5. Measure video level
6. If the measured video level still the mark video level (tolerance) then go to step 4.
7. Start searching where the video level is target marker edge video level. Calculated in 4.5.1
8. Redo step 3 in the other directions
9. Calculate the measure marker size
10. Calculate the marker tolerance

$$\begin{aligned} \text{marker}_{\text{width}} &= \begin{cases} \text{upper limit} = \text{marker}_{\text{width}} * \frac{100 + \text{msxtol}}{100} \\ \text{lower limit} = \text{marker}_{\text{width}} * \frac{100 - \text{msxtol}}{100} \end{cases} \\ \text{marker}_{\text{height}} &= \begin{cases} \text{upper limit} = \text{marker}_{\text{height}} * \frac{100 + \text{msytol}}{100} \\ \text{lower limit} = \text{marker}_{\text{height}} * \frac{100 - \text{msytol}}{100} \end{cases} \end{aligned}$$

11. If the measure marker size outside upper and lower limit then go module spiral search to continue. The 'marker point' P is not really a point on the marker but for example a partial.
12. Move the beam to center of the marker
13. If the marker shift is more than then redo the coarse search.
14. Go to next module, Marker level.

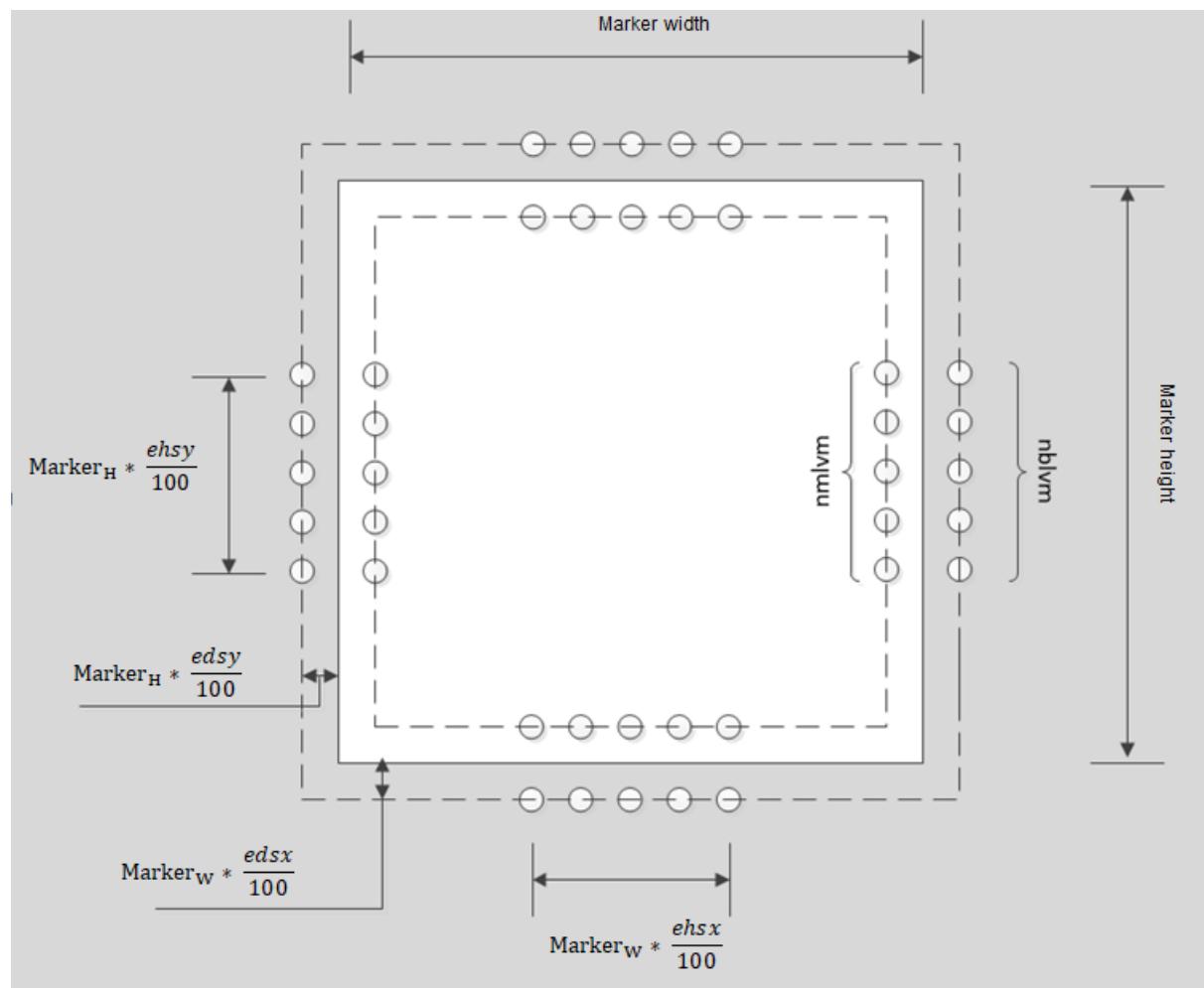
Adjustable parameters

name	description	unit	value
csixstp	Initial step size in x/y-direction in percentage of marker width/height	perc.	default: 30 range: [10,80]
csiystp		perc.	default: 30 range: [10,80]
srcdir	Edge search direction specification: 1 = along y axis only 0 = along x axis -1 = along x and y		default: -1 range: [-1 , 1]
msxtol	Allowed marker width/height tolerance in % of marker width/height. Set to 0 if not search in x/y-direction. Used to check if the found marker width/height	perc.	default: 20 range: [0,40]
msytol		perc.	default: 20 range: [0,40]



4.5.4 Measure marker level

Measure marker video level and calculate the edge video level for each side.



1. Measure marker video level, nmlvm number of points on one side marker
Measure background video level, nblvm number of points on the background.
2. Calculate by averaging the mean background and marker video levels for target edge video level
3. Redo for all side.

Adjustable parameters

name	description	unit	value
edsx	positive edge distance along the edge (x or y) and around the edge middle position, to perform the video level, noise rms level and edge fine search measurements. (percent of marker size)	perc	default: 30 range: [0,90]
edsy		perc	default: 30 range: [0,90]

eshx	positive shift from marker edge to measurement position (x or y)	perc ·	default: 20 range: [5,50]
eshy		perc ·	default: 20 range: [5,50]
nblvm	number of background video level measurements		default: 5 range: [1,25]
nmvilm	number of marker video level measurements		default: 1 range: [1,5]

4.5.5 Measurement video noise level / Setting noise reduction filter

The noise reduction filter is used to reduce the noise in the signal and the filter is as follows calculated:

- Difference between mean marker video level and mean background video level for an edge:

$$\text{DifMMBMedge} = \text{VidLev edge, marker, mean} - \text{VidLev edge, background, mean}$$

- Edge slope:

$$A_{edge} = (\text{VidLev marker, edge, 1} - \text{VidLev background edge, 1}) +$$

$$(\text{VidLev marker, edge, 2} - \text{VidLev background edge, 2}) +$$

....

+

$$(\text{VidLev marker, edge, nelgm} - \text{VidLev background edge, nelgm})$$

$$A_{edge, mean} = A_{edge} / nelgm$$

$$\text{Slopeedge} = (150 * \text{DifMMBMedge}) / A_{edge, mean}$$

$$\text{If } (\text{Slopeedge} > 1000) \quad \text{Slopeedge} = 1000$$

$$\text{If } (\text{Slopeedge} < 150) \quad \text{Slopeedge} = 150$$

- noise rms level for edge: NLedge. For fine search the rms is calculated by doing nnlvm measurement near an edge. If we are in positive video level mode the measurements are done on the background else if we are in negative video level mode the calculation is done on the marker (see figure 2.2.1)

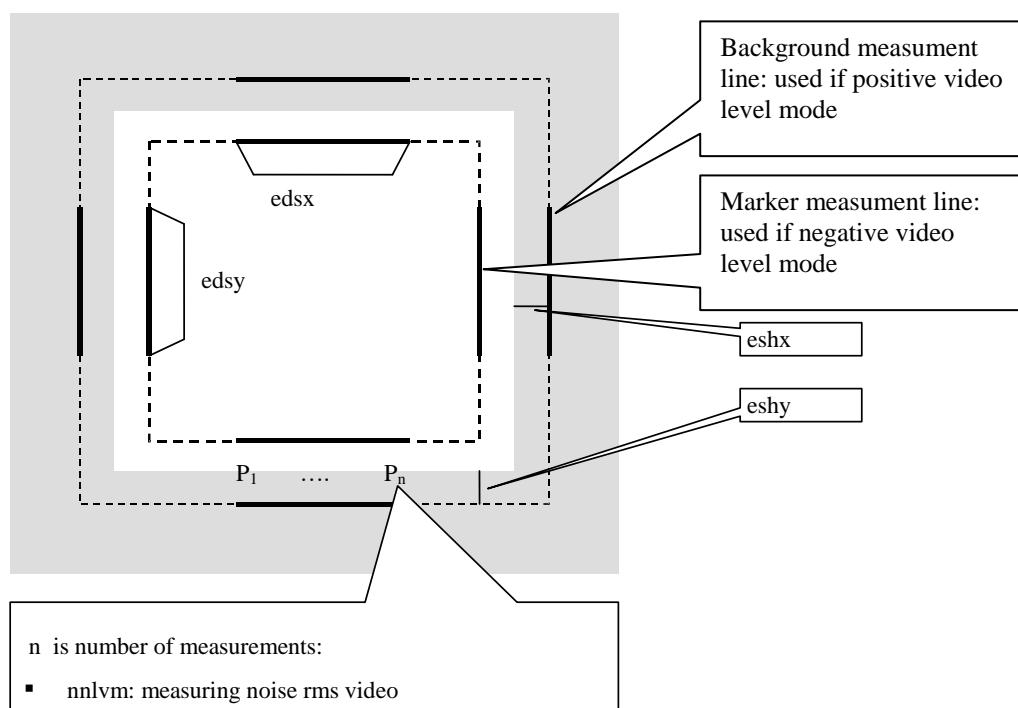


Figure 27 - Slope measurement

- $B_{edge} = 3 * NLedge / 5$
- $C_{edge} = (fserr * DifMMBMedge) / Slopeedge$
- Currently there are 8 filters which widths are defined in kHz. Higher kHz means smaller width (more waves within a distance X). We have to choose one of these filters. Figure 28 shows how this is done.

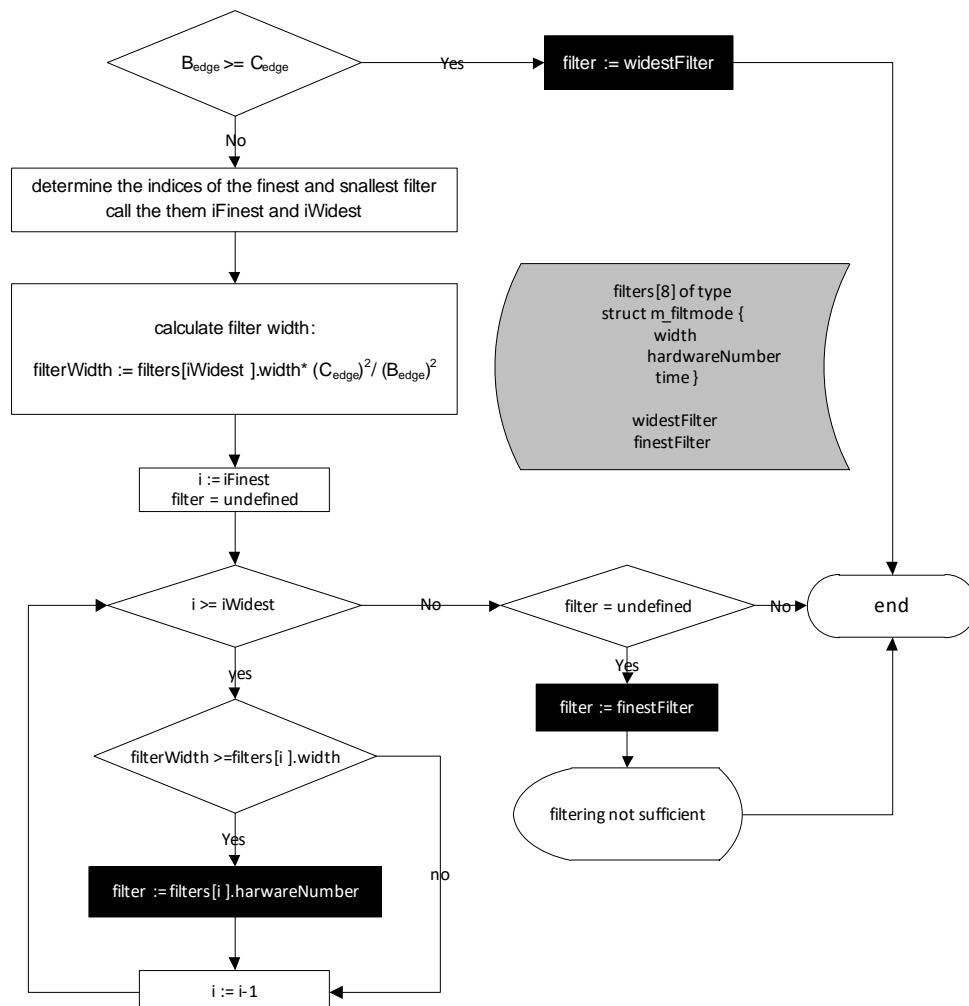


Figure 28 - Filter adjustment

Adjustable parameters

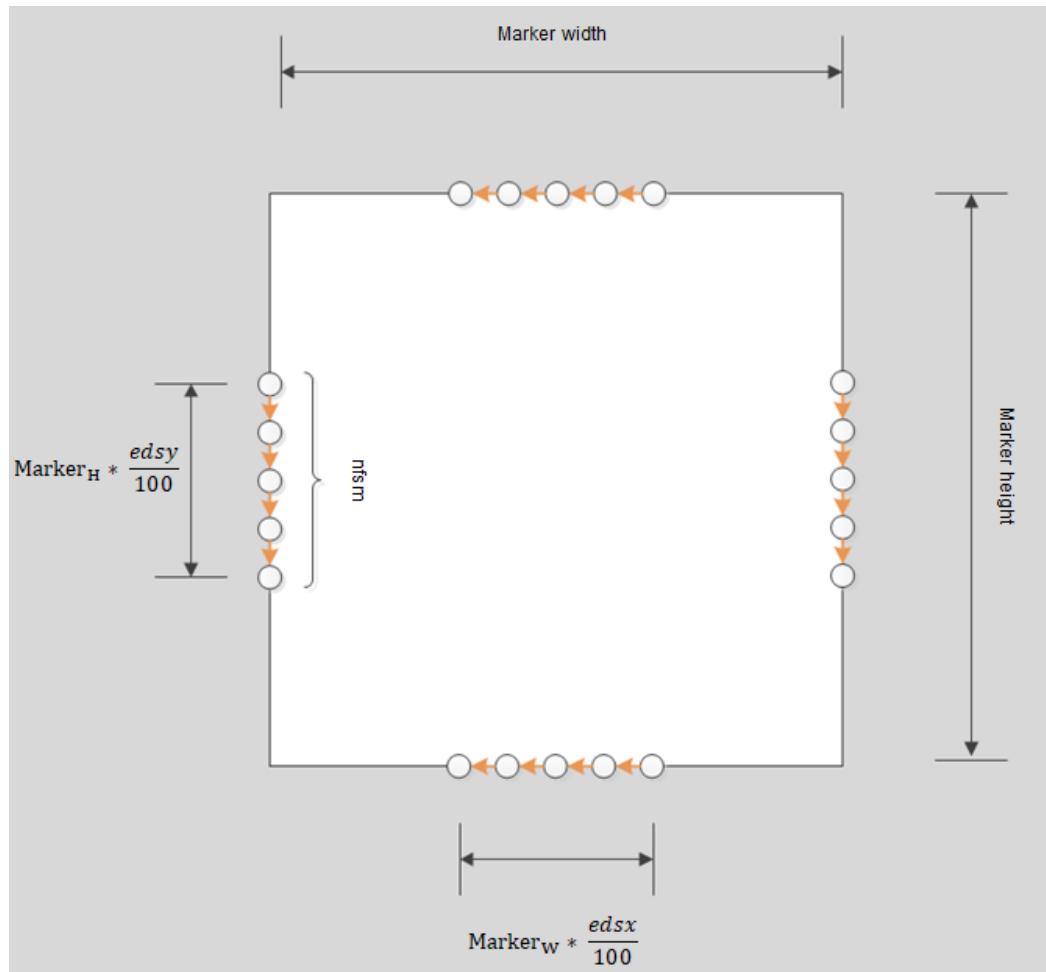
name	description	unit	value
edsx	4.5.5.1 positive edge distance along the edge (x or y) and around the edge middle position to perform noise rms level measurement (percent of marker size)	perc.	default: 30 range: [0,90]
edsy		perc.	default: 30 range: [0,90]
eshx	positive shift from marker edge to measurement position (x or y)	perc.	default: 20 range: [5,50]

eshy		perc.	default: 20 range: [5,50]
fserr	Maximum allowed position in which we find an error due to noise influence in the edge search routine in nm	nm	default: 20 range: [10,50]
nelgm	Number of measurements to perform along an edge for edge (slope) length measurement		default: 5 range: [1,25]
nnlvm	Number of measurements to perform along an edge for noise rms video		default: 1 range: [1,5]

4.5.6 Fine search algorithm

Until now we have seen how to use the spiral search and coarse search algorithm to determine the positions of the marker edges and centre. The accuracy of this marker search method is relatively low because it is based on one measurement for each edge. The video level can vary along the edge, therefore an interpolation with N measurements near along an edge is performed. The interpolation steps are as following:

- Given are the deflections and decision levels for each edge. The deflections have been calculated with the spirals - and coarse search algorithm. The length l_d of the line along we interpolate is set by the user. Calculate the start point of the interpolation by subtracting $l_d/2$ depending on the edge either from Xmarker centre or Ymarker centre. The step size is defined as l_d / n_{fsm} . n_{fsm} is a parameter that is specified by the user.



- At every step we determine the background and marker position and corresponding video levels using a linear search algorithm (it works similar as the first search step of the coarse search algorithm, see chapter 3.2) and interpolate a point with the formula given in figure 3.3.2

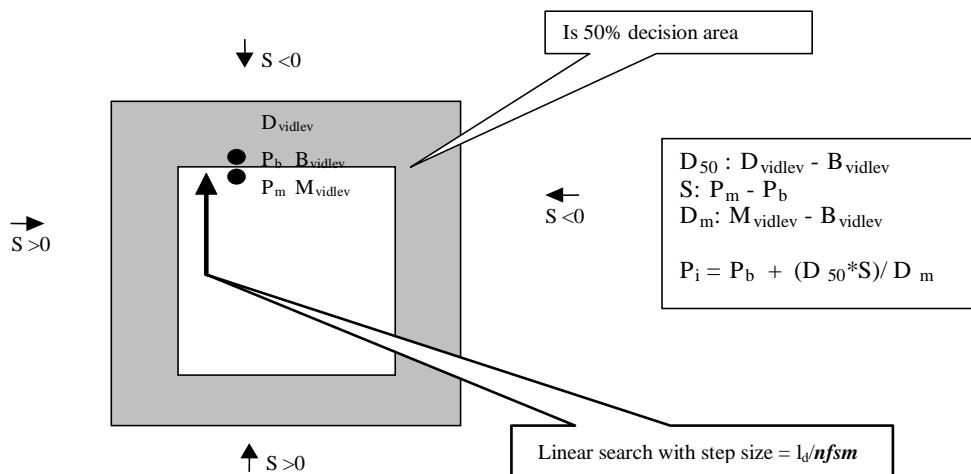


Figure 29 - Linear search during fine search procedure

- If D_{50} is large the edge probably lays outside the decision area. In this case we have to move P_b relatively much. The inverse applies if B_{vidlev} and D_{vidlev} are close together .
- S determines the sign of the offset (see figure above)
- If D_m is large the marker can be distinguished well from the background. In this case we have to move P_b relatively less. The inverse applies if the video level of the background and marker are close together.
- We calculate for every interpolated point its difference with the deflection of the edge that we are interpolating. The x or y position of the edge is than calculated by adding the average interpolation difference to the x or y deflection (see figure 3.3.3)

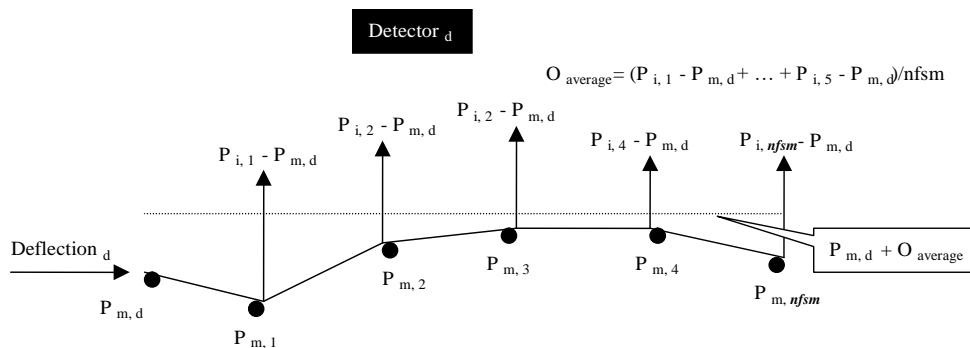


Figure 30 - Interpolation strategy

Adjustable parameters

name	Description	unit	value
edsx	x (width)/ y (height) interpolation length defined as a percentage of the marker width/height	perc.	default: 30 range: [0,90]

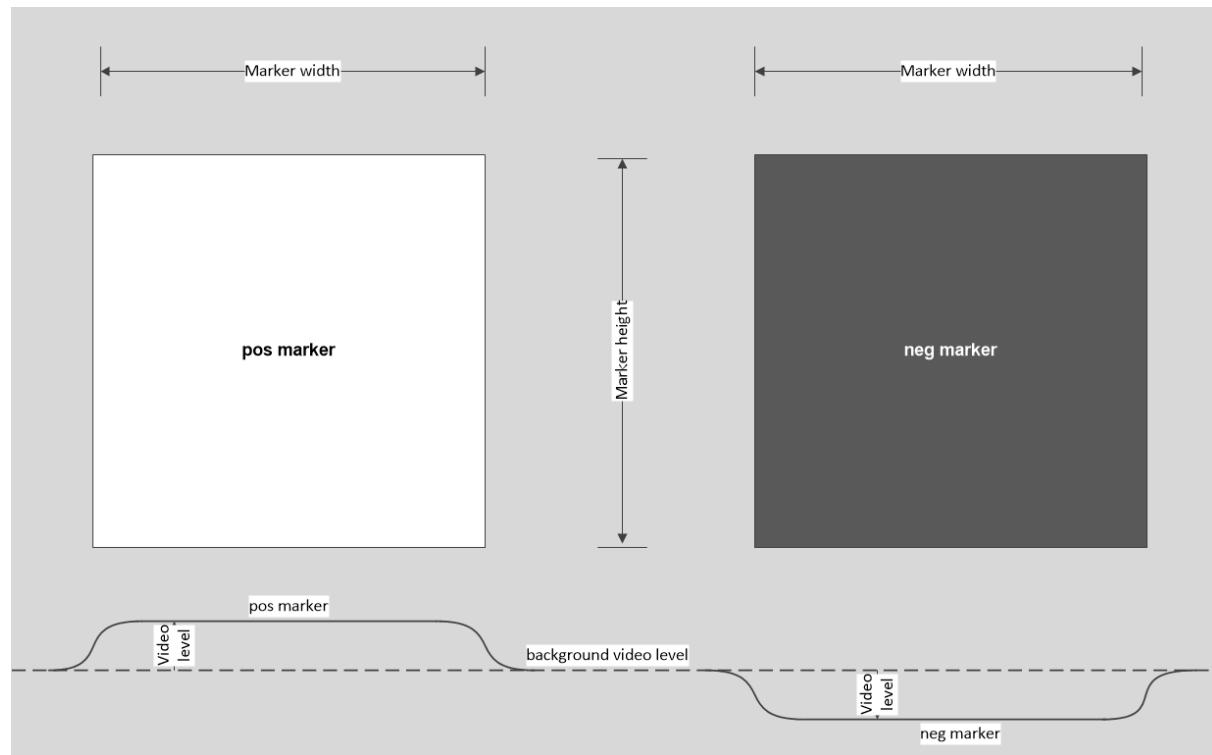
edsy		perc.	default: 30 range: [0,90]
n fsm	Number of measurements to perform along an edge		default: 20 range: [1,25]
srcdir	Edge search direction specification: -1 = along x and y 0 = along x axis 1 = along y axis only		default: -1 range: [-1 , 1]

4.6 Rectangle marker search

Rectangular markers are the default for the EBPG. The markers on the holder and stage are 20µm x 20µm square markers.

4.6.1 Define rectangle marker

Create a rectangle marker



- From the command prompt, enter
`pg marker create rectangular <level> <width>,<height> <marker_ident>`
 to create an marker where;
 <level> is pos or neg, depending if the marker has more or less brightness (video level) than the background
 <width> is width of the marker in µm
 <height> is height of the marker in µm
 <marker_ident>, name of the marker.



Reserved marker_ident names are HOLDER, TOPO, JOY, HBAR and VBAR.

- Enter `pg info marker ident <marker_ident> --test=par`, to print the marker search parameter table.
- Enter `pg marker set <marker_ident> <search parameter> <value>` to change a specification marker search parameter where,
- <search parameter> is the name of search parameter like in the list of step 1.

4.6.2 Searching rectangle marker

Search a rectangle marker

A calibration marker

1. From the command prompt, enter `pg move marker` or enter `mvm`
2. EBPG will do
 - Stage will move to the location of the holder marker
 - Start the search routine
 - When search routine is finished then the stage will be at the center of the marker
3. Enter `pg get tab` or enter `mpg tab` to get the stage coordinate.

User defined marker

1. Enter `pg move marker <Xposition>,<Yposition> [/rel] <marker_ident>` where
 - <Xposition> is the center of the marker in X
 - <Yposition> is the center of the marker in Y
 - [/rel] is optional and tells if the X and Y positions are relative to the current position
 - <marker_ident> is name of the marker
2. EBPG will do
 - Stage will move to the location
 - Start the search routine
 - When search routine is finished then the stage will be at the center of the marker
3. Enter `pg get tab` or enter `mpg tab` to get the stage coordinate.

Deflect marker

1. Enter `pg move marker <Xposition>,<Yposition> /rel /def <marker_ident>` where
 - <Xposition> is the center of the marker in X
 - <Yposition> is the center of the marker in Y
 - /rel will tell that the X and Y positions are relative to the current position
 - /def will tell that the marker has be done only by deflecting the beam
 - <marker_ident> is name of the marker
2. EBPG will do
 - Stage will at the current position
 - Marker will be search only with deflecting the beam
 - When search routine is finished the main beam deflected at the center of the marker
3. Enter `pg get markdefl` or enter `mpg markdefl` to get the main beam deflected, the relative position of the marker from the current stage position

4.6.3 Remove rectangle marker

Delete a rectangle marker.

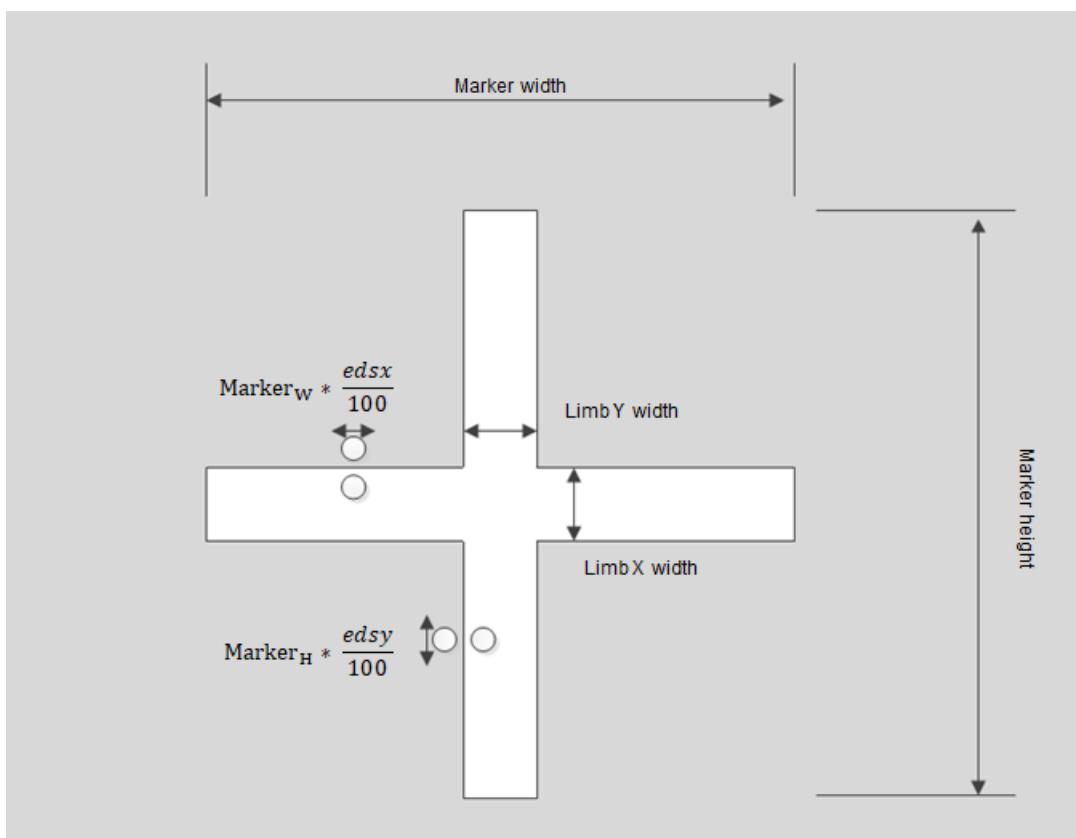
1. From the command prompt, `pg marker delete <marker_ident>`, where <marker_ident>, is the name of the marker to be deleted.
2. Enter `pg info marker ident "*" to get the list of markers`

4.7 Cross marker search

Cross markers on the holder and stage is a cross of 100µm x100µm with a limb width/height of 4µm.

4.7.1 Define cross marker

Create a rectangle marker



- From the command prompt, enter `pg marker create cross <level> <width>, <height> <width limb X>, <width limb Y> <marker ident name>` to create an marker where;
 <level> is pos or neg, depending if the marker has more or less brightness (video level) then the background
 <width> is width of the total marker in µm
 <height> is height of the total marker in µm
 <width limb X> is width of the horizontal marker limb in µm
 <width limb Y> is width of the vertical marker limb in µm
 <marker_iden>, name of the marker.



Reserved marker_iden names are HOLDER, TOPO, JOY, HBAR and VBAR.

- Enter `pg info marker ident <marker_iden> --test=par`, to print the marker search parameter table.
- Enter `pg marker set <marker_iden> <search parameter> <value>` to change a specification. Change the following to parameter to get better performance
`pg marker set <marker_iden> israd <2x width> # search range set to 2x size`
`pg marker set <marker_iden> msxtol 40 # marker size tolerance X for coarse`

```

search
pg marker set <marker_ident> msytol 40 # marker size tolerance Y for coarse
search
pg marker set <marker_ident> edsx 1      # measure at ( 1% * <X size>) of edge
pg marker set <marker_ident> edsy 1      # measure at ( 1% * <Y size>) of edge
pg marker set <marker_ident> nelgm 1      #Number measure on the edge

```

4.7.2 Searching cross marker

Search a cross marker.

Cross marker can only be search on axis for alignment.

1. Enter `pg move marker <Xposition>,<Yposition> [/rel] <marker_ident>` where
 <Xposition> is the center of the marker in X
 <Yposition> is the center of the marker in Y
 [/rel] is optional and tells if the X and Y positions are relative to the current position
 <marker_ident> is name of the marker
2. BEAMS will do
 - Stage will move to the location
 - Create two rectangle markers, HBAR and VBAR
 - Start the search routine, search for HBAR
 - Start the search routine, search for VBAR
 - When search routine is finished then the stage will be at the center of the marker
3. Enter `pg get tab` or enter `mpg tab` to get the stage coordinate.

4.7.3 Remove cross marker

Delete a rectangle marker.

1. From the command prompt, `pg marker delete <marker_ident>` where
 <marker_ident>, is the name of the marker to be deleted.
2. Enter `pg info marker ident "*" to get the list of markers`

4.7.4 Marker cross routine (HBAR and VBAR)

Searching the HBAR and VBAR search routine are idem as for rectangle marker, only HBAR has search parameter srkdir = 1 and VBAR has srkdir = 0 .

4.8 Marker JOY

Type JOY marker introduced to be able to "search for markers" that cannot be found by the standard marker search. There are 2 options to run a JOY marker for manual searching with SEM or running a user defined script.

4.8.1 Define a JOY marker for manual SEM search

1. From the command prompt, enter `pg marker create joy <marker ident name>`, to create an joy marker where;
`<marker_ident>`, name of the marker.



Reserved marker_ident names are HOLDER, TOPO, JOY, HBAR and VBAR.

4.8.2 Creating a JOY marker width user own script

1. From the command prompt, enter `ce xxxx` where `xxx` is the project environment.
2. Enter `pg marker create joy '<script name>' <marker ident name>` to create and joy marker with script where;
`<script name>`, name and path of the script
`<marker_ident>`, name of the marker.

4.8.3 Searching JOY marker with SEM (without script)

1. Enter `pg move marker <Xposition>,<Yposition> [/rel] <marker_ident>` where
`<Xposition>` is the center of the marker in X
`<Yposition>` is the center of the marker in Y
`[/rel]` is optional and tells if the X and Y positions are relative to the current position
`<marker_ident>` is name of the marker
2. BEAMS will do
 - Stage will move to the location
3. Use at this point the SEM image to move the stage. Be aware that the SEM deflection does NOT accurately match the deflection of the pattern generator. To obtain the best accuracy you need to calibrate against a marker search of the holder marker and indicate on the screen where this marker is found. Because this uses the automatic marker search of the tool it leaves the mark on axis. In case of the old style monitor mark on the monitor the corners of the holder marker. In case of Csem, you can place a box holding the marker. See help of Csem.
4. Enter return to continue or type q to quit. If JOY marker without script are started in Cebpg / queueing (cjob) then enter return or type q in the text area under the queue in Cepgb.

4.8.4 Searching JOY marker with script

1. Enter `pg move marker <Xposition>,<Yposition> [/rel] <marker_ident>` where
`<Xposition>` is the center of the marker in X
`<Yposition>` is the center of the marker in Y

[/rel] is optional and tells if the X and Y positions are relative to the current position
<marker_ident> is name of the marker

2. BEAMS will do
 - Stage will move to the location
 - Execute the script



Make sure that at the end of the your script that stage will on center of the marker (without defelection)

4.8.5 Remove joy marker

Delete a rectangle marker.

1. From the command prompt, `pg marker delete <marker_ident>` where <marker_iden>, is the name of the marker to be deleted.
2. Enter `pg info marker ident "*" to get the list of markers`

4.9 Image markers

Image correlation is used to compare two images. A scan image and a reference image and calculate the shift between them. Image correlation is should not be mix up with shape recognition, where a shape will be search in an area.

This chapter will explain;

- The theory behind image correction
- Creating, searching and deleting an image marker
- Searching routine of image markers
- Tip and tricks and what to do with errors.

Requirement:

Image correlation works on EBPG with beams version of 11a or higher.

UPG used command mps imagebase and mps imagefactor to scan speed to grab image

GPG add shift in to reference image (see 4.9.7.3)

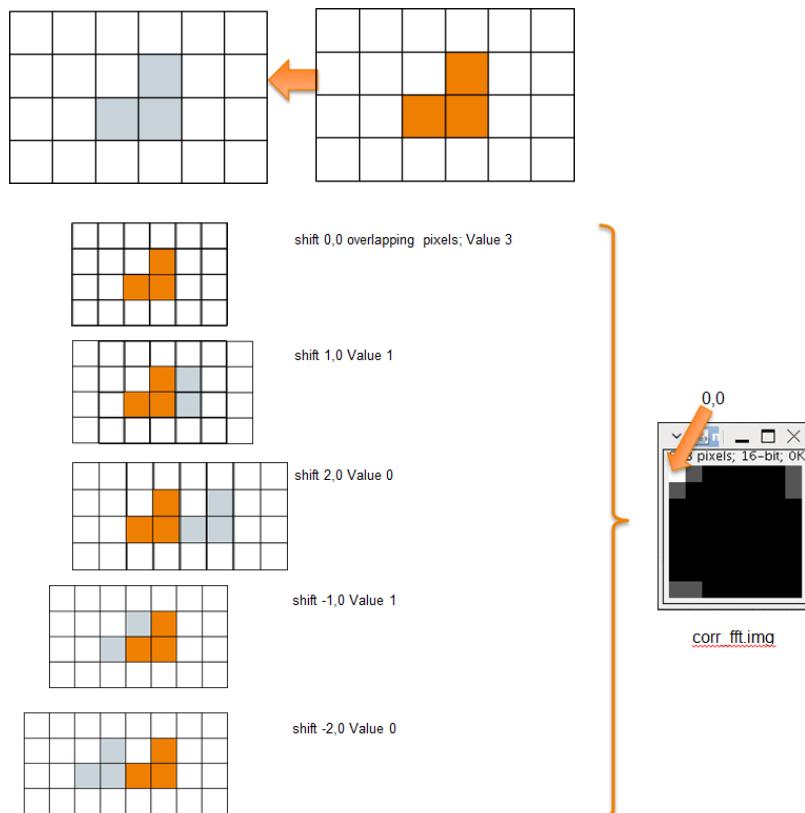
4.9.1 Theory

The correlation goes in two steps

1. Comparing images
2. Gaussians fit

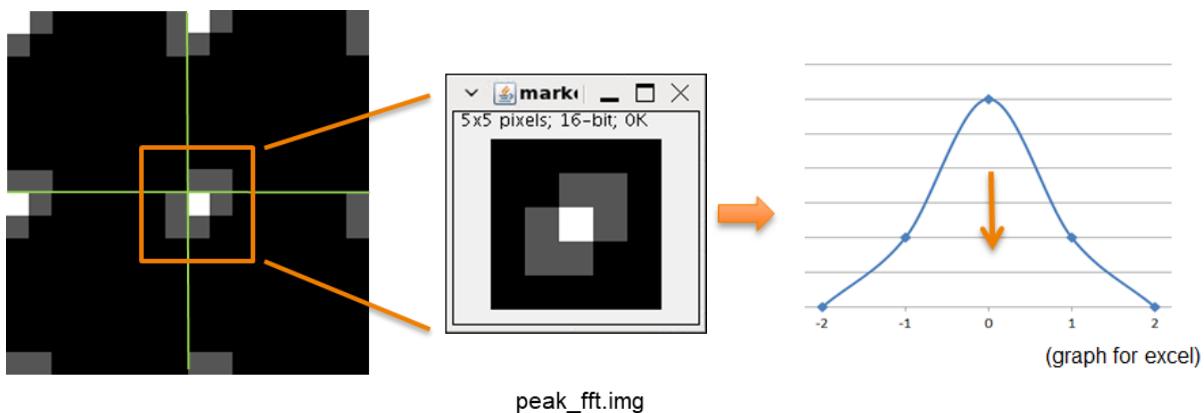
Comparing images

The scanned image will be compared with the reference image and count how many overlapping pixels the images have. By shifting scan image over the reference image and counting every time checking overlapping pixels this will create matrix, shift X,Y vs amount of overlaps. This correlation matrix can be plotted in filename_corr.img.



Gaussians fit

In the correlation matrix will be searched for the location where the highest value is. From that location, a matrix of N x N pixels will be clip out, and copied to peak matrix.



Peak matrix can be saved in filename_peak.img. A Gaussians fit will be fit on the peak matrix, and the position of the top is the sub shift. Add this sub shift to the location, which has found in the correlation matrix, then you get the total pixel shift. Multiplied with the pixel size this gives the shift in nm between the two images.

4.9.2 Define an image marker

Create an image marker.

Presetting

1. From the command prompt, enter `ce xxxx` where xxxx is the project environment.
2. Enter `cp <pattern_file.gpf> $PG_PATTERNS` to copy the gpf pattern file with markers into the patterns directory.
3. Enter `rm $PG_IMAGES/<pattern_file.img>` to remove the image file from the images directory.
4. Enter `rm $PG_IMAGES /<pattern_file.txt>` to remove the image data from into the images directory.

Location of marker in the gpf file

1. Enter `cview $PG_PATTERNS/<pattern_file.gpf>` to start Cview software and open gpf pattern file.
2. Search for the center of the marker.
3. Remember the coordinates of the center of the marker.

If you don't have a gpf of the marker, see paragraph 4.9.7.4



Created marker

1. Enter `pg marker create image <level> <pattern_file> <offset_x,offset_y> <marker_ident>` to create image marker where;
 - `<level>` is pos or neg, depending if the marker has more or less brightness (video level) then the background.
 - `<pattern_file>`, gpf pattern file with markers.
 - `<offset_x,offset_y>`, location of the center of the marker in gpf file.
 - `<marker_ident>`, name of the marker.

Reserved marker_ident names are HOLDER, TOPO, JOY, HBAR and VBAR.



2. Enter `pg info marker ident <marker_ident> -- test=par` to check scan parameters
3. Enter `pg marker set <marker_ident> mtmacc 30` set required accuracy in nanometers



Before archiving a defined marker in the global data, make sure the EPG can calibrate. After archiving in the global data, defined marker will remain during a coldstart.

Enter `pg adj ebpg `mpg maxcaldef1`` to do a calibration

Enter `pg ebpg save` to archive in the global data.

Reference image

1. Enter `gpfraw <pattern_file> <pattern_file>_coarse -- <offset_x,offset_y> <exel size_x, exel size_y> <points_x,points_y>` to create a reference image for coarse search, a large area.
Parameters where
 <pattern file>, gpf pattern file with markers
 --, Required, to stop parsing for negative X values.
 <offset_x,offset_y>, location of the center of the marker in gpf file.
 <exel size_x, exel size_y>, exel size in um.
 <points_x,points_y>, number of exels.
 A total scan area is (exel size_x * points_x),(exel size_y * points_y).
2. Enter `ijraw $PG_IMAGES/<pattern_file>_coarse.img` to start ImageJ software and check the reference image.
3. Enter `gpfraw <pattern_file> <pattern_file>_fine -- <offset_x,offset_y> <exel size_x, exel size_y> <points_x,points_y>` to create a reference image for fine search, a small area for higher accuracy.
Parameters idem as step 2.
4. Enter `ijraw $PG_IMAGES/<pattern_file>_fine.img` to start ImageJ software and check the reference image.

4.9.3 Searching an image marker

Searching image marker, the image marker has to be created.

Reference images exists

1. From the command prompt, enter `ce xxxx`, where xxxx is the project environment where the reference images are in the \$PG_IMAGES directory.
2. Enter `pg move marker <stage_x,stage_y> <marker_ident>` to start marker search, where;
 <stage_x,stage_y> stage coordinate where the marker is expected.
 <marker_ident> name of the marker.
3. Enter `pg get tab` to get the stage coordinates where the EPG found the marker.

Reference images don't exists

1. From the command prompt, enter `ce xxxx`, where xxxx is the project environment where the gpf pattern file is in the directory \$PG_PATTERNS.
2. Enter for coarse search the following lines
`export PG_IMAGE_COARSE_SIZE=<image width in mm>`
`export PG_IMAGE_COARSE_NUM=<number of exels>`
3. Enter for fine search the following lines
`export PG_IMAGE_FINE_SIZE=<image width in mm>`
`export PG_IMAGE_FINE_NUM=<number of exels>`

4. Enter `pg move marker <stage_x,stage_y> <marker_ident>` to start marker search, where
`<stage_x,stage_y>` stage coordinate where the marker is expected.
`<marker_ident>` name of the marker.
 Reference image will be created, and remaining
5. Enter `pg get tab` to get the stage coordinate where the EPG found the marker.



For more setting parameters, read paragraph 4.9.6.

4.9.4 Image marker in CJOB

1. From the command prompt, enter `ce xxxx`, where xxxx is the project environment where the gpf pattern file is in the directory `$PG_PATTERNS`.
2. Define first an image marker, like described in paragraph 4.9.2
3. In Cjob positioning the markers where is needed. (See Cjob software manual.)
4. In Cjob, in the expose module,
 - Check Init and Exit (ini + exi)
 - Enter filename for the ini-file



5. Press Init button. An edit window will open.
6. Copy following text in to the edit window.

```
#!/bin/bash

# remove previous image and settings
rm $PG_IMAGES/<marker_ident>*.img
rm $PG_IMAGES/<marker_ident>*.txt

export PG_IMAGE_COARSE_SIZE=0.10          # image width in mm
export PG_IMAGE_COARSE_NUM=500            # number of exels
export PG_IMAGE_COARSE_SAMPLE=0          # scan every exels 2^sample times and average
export PG_IMAGE_COARSE_FRAME=0          # scan every images 2^frame times and average
export PG_IMAGE_COARSE_SAVE="yes"        # save scan image in <marker_ident>_coarse.img
#unset PG_IMAGE_COARSE_SAVE             # do not save scanned image

export PG_IMAGE_FINE_SIZE=0.080          # image width in mm
export PG_IMAGE_FINE_NUM=500            # number of exels
export PG_IMAGE_FINE_SAMPLE=0          # scan every exels 2^sample times and average
export PG_IMAGE_FINE_FRAME=0          # scan every images 2^frame times and average
export PG_IMAGE_FINE_SAVE="yes"        # save scan image in <marker_ident>_fine.img
#unset PG_IMAGE_FINE_SAVE              # do not save scanned image

export PG_IMAGE_TIF=="yes"           # save scan image also in TIFF format
#unset PG_IMAGE_TIF                  # do not save in TIFF format

mps imagembsbase 10000      # set adding wait time in ns per excel (UPG only)
mps imagembsfactor 10000     # set adding wait time in ns/µm beam shift (UPG only)
```

7. Adjust `<marker_ident>`, coarse size, fine size, coarse num and fine num.
8. Press Save. To save the ini file
9. Press Close. To close the edit window.
10. Export the job, like normal proceeding.



To test the size and num parameters.

Do step 1 till 2 and copy the lines of step 6 in the command line.

Enter `mvm 0,0 /rel <marker_ident>` # An image marker search on the marker location.

Enter `cd $PG_IMAGES` to go to image directory.

Enter `ijraw <imagename>.img` to check the image

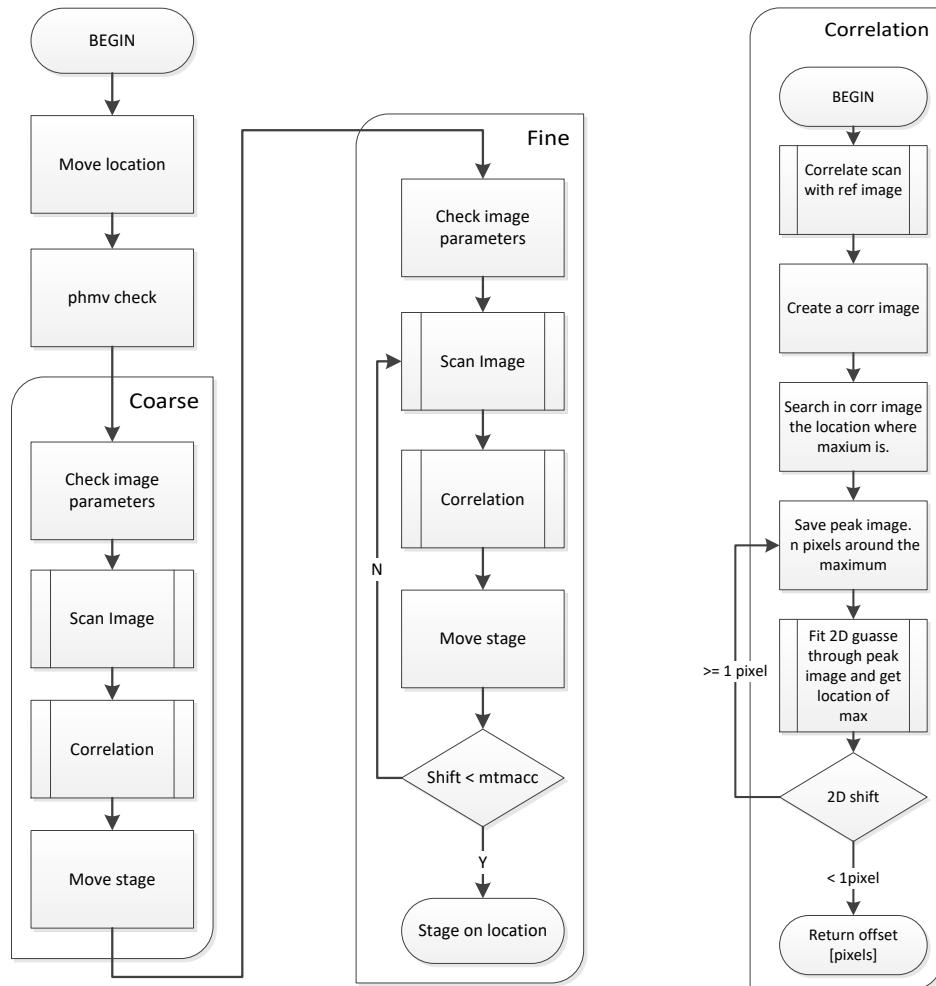
4.9.5 Removing an image marker

Delete an image marker.

1. From the command prompt, enter `ce xxxx`, where xxxx is the project environment where the reference images are in the `$PG_IMAGES` directory.
2. Enter `pg info marker ident <marker_ident>` to get information about the marker, where `<marker_ident>`, is the name of the marker to be deleted.
3. Read from the screen the pattern file name
4. Enter `rm $IMAGES/<pattern file name>_coarse.*` to remove coarse reference image and data.
5. Enter `rm $IMAGES/<pattern file name>_fine.*` to remove fine reference image and data.
6. Enter `pg marker delete <marker_ident>` to delete marker from the marker table.
7. Enter `pg marker ident "*" to get the marker table on the screen`

4.9.6 Searching Routine of image markers

Marker search consist of different modules PM Marker level, Coarse search, Fine search.



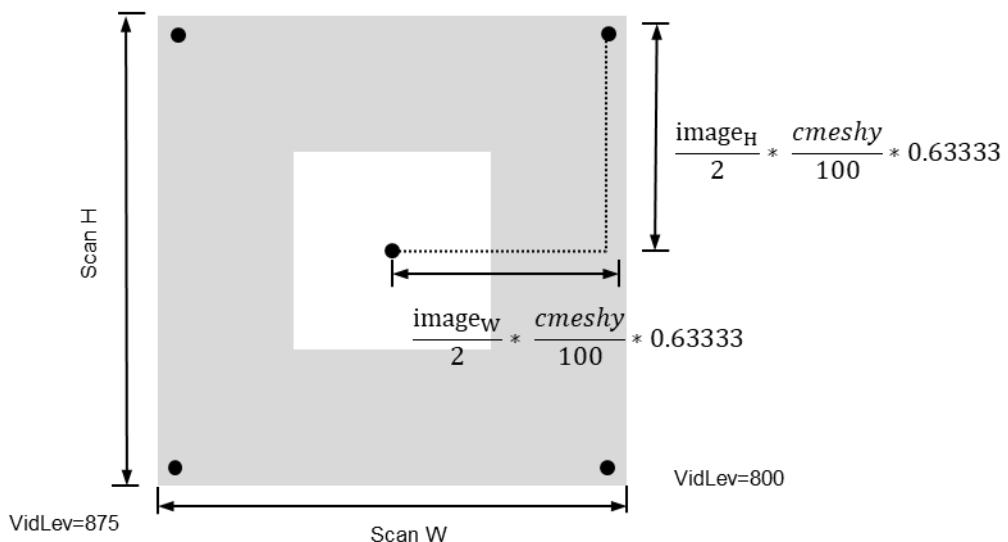
4.9.6.1 PM Marker level

Adjust the background video level.

1. Calculate the position here the background video level will be measured.

VidLev=775

VidLev=850



$$Y_{deflection} = height_{image} * 0.31666 * \frac{cmeshy}{100}$$

When cmeshx and cmeshy are default values (150%), to get a deflection position 95% of the image width and height a factor of (95% /150% =) 0.63333 for a 0.5 scan width or 0.31666 for a full scan width.)

2. Measure on 4 deflected positions the video level and select the position where the second highest or second lowest respectively if the marker level is positive or negative.
3. Calculate acceptable range of video level.

$$upper\ limit = hvalev + blvtol$$

$$lower\ limit = hvalev - blvtol$$

4. Change pmhv value so that the video level of selected position is in the upper and lower limit.

Adjustable parameters,

name	Description	unit	Value
blvtol	Tolerance video level on background video level (hvalev)	dac	default: 150 range: [40,200]
hvalev	Target video level for pmhv adjustment	dac	default: 650 range: [500,800]
cmeshx	Horizontal shift to measurement position in percentage of the marker width	perc	default: 150 range: [150,1000]
cmeshy	Vertical shift to measure position in percentage of the marker height	perc	default: 150 range: [150,1000]

set: Enter pg marker set <marker_iden> <parameter name> <value>

get: Enter pg info marker ident <marker_iden> --test=par

4.9.6.2 Coarse search

Alignment to coarse reference image.

1. Check if the coarse reference image exists.
 - If exists then uses exel size and number of exels which written in the coarse reference image.txt
 - If not exists then create coarse reference image and use the following parameters


```
export PG_IMAGE_COARSE_SIZE=0.10      # set image width in mm
          export PG_IMAGE_COARSE_NUM=200       # set number of exels
```
2. Scan image using the following parameters

3. mps imagebase <base time> # set adding wait time in ns per excel
 mps imagefactor <factor time> # set adding wait time in ns/ μ m beam shift
 export PG_IMAGE_COARSE_SAMPLE=0 # set number reading an excel (2^{sample})
 export PG_IMAGE_COARSE_FRAME=0 # set number of scans (2^{frame}) times
 export PG_IMAGE_COARSE_SAVE="yes" # set to save scan image in
 <marker_iden>_coarse.img
 unset PG_IMAGE_COARSE_SAVE # do not save scanned image
4. Correlate the scanned image with the reference image and save the result in <marker_iden>_<pattern_file>_coarse_fft.img
5. Search in image for max pixel, the pixel with the highest brightest level.
6. Search form the max pixel in 8 directions (horizontal, vertical and diagonal) for first pixel which has a brightness level more lower than 500 bits. Take the direction which has the longest distance to the max pixel. This ($\text{distance} * 2 + 1$, is area which is clipped out and save the result in <marker_iden>_<pattern_file>_peak_fft.img
7. Fit 3D gausses over the peak image, and find the location of the top.
8. Calculate the offset, $XY_{\text{offset}} = XY_{\text{top}} - XY_{\text{image center}}$
9. If offset is more than 1 pixel then add these offset to the max pixel and rerun form step 6.
10. Calculate the shift, $XY_{\text{shift}} = (XY_{\text{offset}} + XY_{\text{maxpixel}}) * \text{exel size}$
11. Move the stage relative with shift

Adjustable parameters

```
export PG_IMAGE_COARSE_SIZE=0.10      # image width in mm
export PG_IMAGE_COARSE_NUM=200        # number of exels
export PG_IMAGE_COARSE_SAMPLE=0       # scan every exels  $2^{\text{sample}}$  times and average
export PG_IMAGE_COARSE_FRAME=0        # scan every images  $2^{\text{frame}}$  times and average
export PG_IMAGE_COARSE_SAVE="yes"     # save scan image in <marker_iden>_coarse.img
unset PG_IMAGE_COARSE_SAVE          # do not save scanned image
```



With export and unset are local commando's, there setting will be not transfer to outer terminals.
 Using export and unset in cjob, place these commands in the ini file.
 Read out the value use echo \$PG_<parameter>

```
mps imagebase <base time>          # set adding wait time in ns per excel
mps imagefactor <factor time>       # set adding wait time in ns/ $\mu$ m beam shift
```



With mps <parameter> <value> or pg set <parameter> <value> these settings will be remembered by beams and are then available in other terminals.
 Using mps / pg set in cjob, place these commando's in the ini file.
 Read out the value use mpg <parameter> or pg get <parameter>

4.9.6.3 Fine search

Alignment to fine reference image.

- 1-11. Replace the word coarse with the word fine. Refer to section 4.9.6.2
12. If shift larger than mtmatc then rerun from step 2.

Adjustable parameters

```

export PG_IMAGE_FINE_SIZE=0.080      # image width in mm
export PG_IMAGE_FINE_NUM=500        # number of exels
export PG_IMAGE_FINE_SAMPLE=0       # scan every exels 2^sample times and average
export PG_IMAGE_FINE_FRAME=0        # scan every images 2^frame times and average
export PG_IMAGE_FINE_SAVE="yes"     # save scan image in <marker_ident>_fine.img
unset PG_IMAGE_FINE_SAVE          # do not save scanned image
mps imagebase <base time>         # set adding wait time in ns per excel (UPG only)
mps imagefactor <factor time>     # set adding wait time in ns/µm beam shift (UPG only)
pg marker set <marker_ident> mtmacc 30 # Required accuracy in nanometer (move marker)

```

name	Description	unit	Value
mtmacc	Required accuracy in nanometer (move marker)	nano meter	default: 20 range: [20,250]

set: Enter pg marker set <marker_ident> <parameter name> <value>

get: Enter pg info marker ident <marker_ident> --test=par

4.9.7 Tips, Tricks and Error handling

4.9.7.1 Error handling

If there are searching faults in fine search then replace the word coarse by fine

1. From the command prompt, enter ce xxxx, where xxx is the project environment where the reference images are in the \$PG_IMAGES directory.
2. Enter cd \$PG_IMAGES to change the current directory to the image directory

When error code is one of the following codes

- E_IMG_CORRERR, NAN correlation coefficient
check reference images and scan images
- E_IMG_MISAT, Image size doesn't match with reference image
check reference images and scan images
- E_IMG_BADCOOR, poor image correlation
check reference images and scan images
- E_IMG_MAXITER, Levenberg-Marquardt out of range
check nopeak images, scan images and reference images

Check reference images

1. Enter pg info marker ident <marker_ident> to get information about the marker, where <marker_ident>, is the name of the marker which has to be searched.
2. Read from the screen the pattern file name.
3. Enter ijraw \$IMAGES/<pattern file name>_coarse.img , to start ImageJ and open coarse reference image.
4. If the coarse reference image does not look what it should look like then do paragraph 4.9.2, session Location of marker in the gpf file and session reference image.

Check scanned image

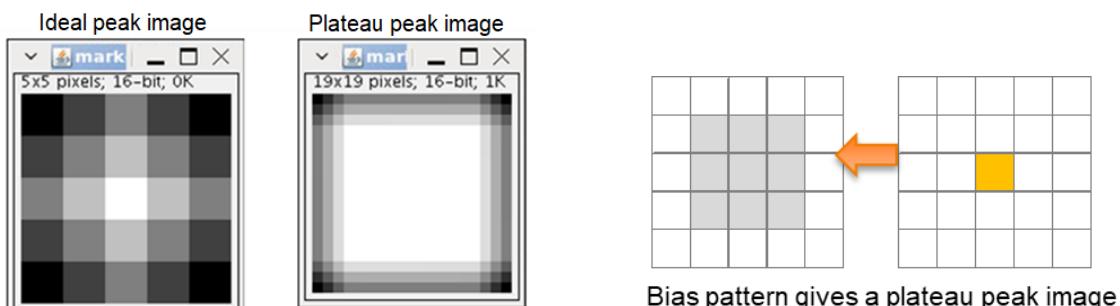
1. Enter `ijraw $IMAGES/<marker_ident>_coarse.img`, to start ImageJ and open coarse scan image.
2. If there is no `$IMAGES/<marker_ident>_coarse.img` then enter at prompt (and add in the `<jobfile>.ini` file) `export PG_IMAGE_COARSE_SAVE="yes"`, to save images.
Enter `pg move marker 0,0 /rel <marker_ident>`, to rerun the search and create the scan images.
3. Check the scan images and compare image with coarse reference image.
4. Check the scan images for brightness.
If the four corners of an image contains same brightness as the marker then do paragraph 4.9.6.1, change the position for auto brightness routine.
Or enter (or add to the `<jobname>.ini` file) `export PG_IMAGE_PMHV=<value>` to set a fix pmhv value.

Check peak image

1. Enter `pg info marker ident <marker_ident>` to get information about the marker, where `<marker_ident>`, is the name of the marker which has to be searched.
2. Read from the screen the pattern file name
3. Enter `ijraw $IMAGES/<marker_ident>_<pattern_file_name>_coarse_peak_fft.img`, to start ImageJ and open coarse peak image.
4. Check that there is one white pixel in the center. If there are more pixels white then do paragraph 4.9.7.2

4.9.7.2 Bias limitation

During the processing of the marker, marker can be smaller or wider than the design dimensions. These bias give the correlation get more overlap and instead of to get a Gaussian peak image, the peak image are more a plateau. The software can extent the 5x5 pixel peak image to a large images, like 19x19 but when this plateau / area get the big vs slope then the gauss fit cannot fit correctly.



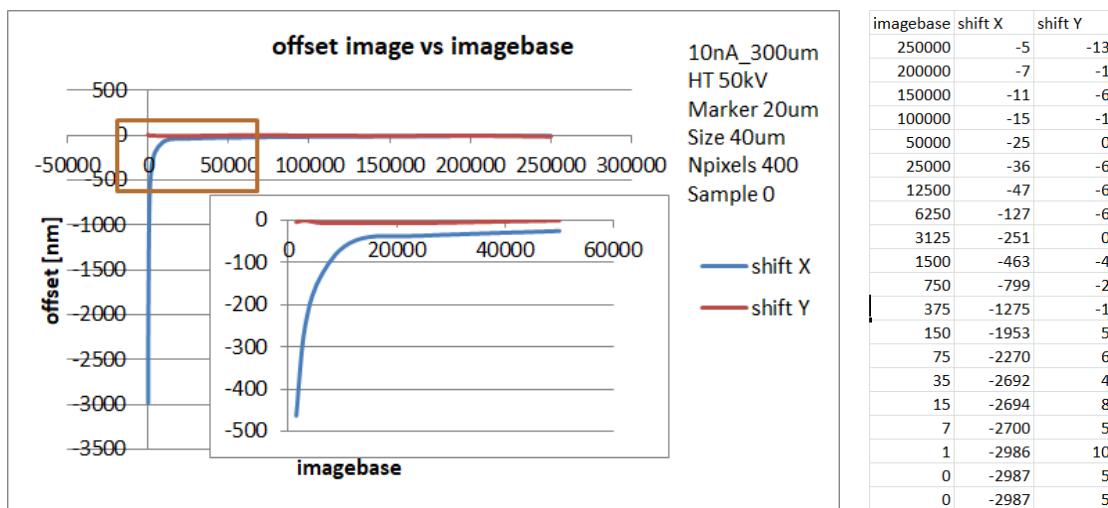
To solve this measure the marker size and add bias to the reference, so that the sizes are matches which each other.

4.9.7.3 Imagebase value

UPG

With the imagebase parameter the extra waiting time for each scan pixel can be set.

- Enter `mpg imagebase` to get the waiting time for each pixel in ns
- Enter `mps imagebase <value>` to set the waiting time for each pixel ns



To get value to set.

1. From the command prompt, enter `ce xxxx`, where xxxx is the project environment
2. Enter `pg arch restore beam <beam_name>`, where
`<beam_name>` is the name of the beam which can be found in the list `pg info arch beam`
3. Enter `pg adjust resolution 1nm` or `pg select pattern <patternname>` to set equal resolution which markers are be searched.
where `<patternname>` the filename.gpf is which to be exposed.
4. Enter `pg adjust ebpg `mpg maxcaldefl`` to calibrate the beam
5. Enter `cd $PG_SCRIPTS` to change directory to script directory
6. Enter `gedit imagescantime.sh` to start a text editor
7. Copy text below in the editor. This contains the bash header

```
#!/bin/bash
. script_init_template
```

8. Copy text below and add to the editor, this contains images scan settings.

```
export PG_IMAGE_COARSE_SIZE=0.10      # image width in mm
export PG_IMAGE_COARSE_NUM=200         # number of exels
export PG_IMAGE_COARSE_SAMPLE=0        # scan every exels 2^sample times and average
export PG_IMAGE_COARSE_FRAME=0        # scan every images 2^frame times and average
export PG_IMAGE_COARSE_SAVE="yes"      # save scan image in <marker_ident>_coarse.img
#unset PG_IMAGE_COARSE_SAVE          # do not save scanned image

export PG_IMAGE_FINE_SIZE=0.040       # image width in mm
export PG_IMAGE_FINE_NUM=200          # number of exels
export PG_IMAGE_FINE_SAMPLE=0        # scan every exels 2^sample times and average
export PG_IMAGE_FINE_FRAME=0        # scan every images 2^frame times and average
export PG_IMAGE_FINE_SAVE="yes"      # save scan image in <marker_ident>_fine.img
#unset PG_IMAGE_FINE_SAVE           # do not save scanned image
```

9. Copy text below and add to the editor, this creates the reference image and image marker.

```
echo "Create marker image rect20.gp"
dump=`cjob_shape rect20.gpf rectangle 20,20` # create marker

rm $PG_IMAGES/rect20_fine.* -f
rm $PG_IMAGES/rect20_coarse.* -f
pg marker create image pos rect20.gpf 0,0 imgrect20
```

10. Copy text below and add to the editor, this will do image marker search with different imagebase delay time.

```

echo " Offset : according to image marker"
echo " Shift : marker found by deflection"
echo " Delta : difference between"

for arg in "0" "10000" "20000" "50000" "60000" "70000" "80000" "100000"
do

mvm # move to the holder marker
mps mbsbase ${arg}
mps mbsfactor 10000
mps mbsfocus 0
mps imagebase ${arg}
mps imagefactor 10000

#move to image marker imgrect20 and get the remaining offset
rest=`pg move marker 0,0 /rel imgrect20 2>&1 | awk 'END{print}'` | awk '{print $8}'` 
rest=`pg update 0_um,0_um "+" ${rest}`

#search deflected to the holdermarker, get founded deflection
mvm 0,0 /rel /defl pos20
shift=`mpg markdefl` 
shift=`pg update 0_um,0_um "-" ${shift}`

#calculate the delta
delta=`pg update ${shift} "-" ${rest}` 
delta=`unit ${delta} "nm"`

# print on screen offset, shift and delta in microns
echo -n "base: ${arg} factor: `mpg imagefactor` "
echo "offset : `unit ${rest} \"nm\"` shift: `unit ${shift} \"nm\"` delta: `unit ${delta} \"nm\"` "

# print to file offset, shift and delta in microns
echo -n "base: ${arg} factor: `mpg imagefactor` " >> $PG_SCRIPTS/plot.dat
output="offset : `unit ${rest} \"nm\"` shift: `unit ${shift} \"nm\"` delta: `unit ${delta} \"nm\"` "
output=`echo ${output} | sed 's/_/_/g' | sed 's/,/_/g' `
echo ${output} >> $PG_SCRIPTS/plot.dat

# imagebase 0 not to datafile, but clear datafile
if [ $arg -eq 0 ]; then
  echo -n "" > $PG_SCRIPTS/plot.dat
fi

done

```

11. Copy text below and add to the editor, this will plot the data to screen

```

echo "press enter to quick"
gnuplot -e "plot 'plot.dat' using (column(2)):(column(17)) title 'x', 'plot.dat'
           using (column(2)):(column(19)) title 'y' ; pause -1"

exit 0

```

12. Save and close the text editor

13. Enter chmod +x imagescantime.sh to make the file excitable

14. ./imagescantime.sh to start the script

15. Check in the graph or table at which imagebase the shift is small enough. And use this imagebase value in the ini file of your job mps imagebase <value>

GPG:

1. Do the step 1 till 14 like the UPG.
2. Check in the table the shift value, this value is for all speeds same.
3. The negative shift value, as to be add to the position of the marker in the reference image.

example

4. Replace in step 9 0,0 by (0 -<shift X> , (0 -<shift Y>)
pg marker create image pos rect20.gpf -<shift X>,-<shift Y> imgrect20
5. Rerun ./imagescantime.sh to start the script

4.9.7.4 No GPF images

Create pattern file with cjob of a cross or rectangle.

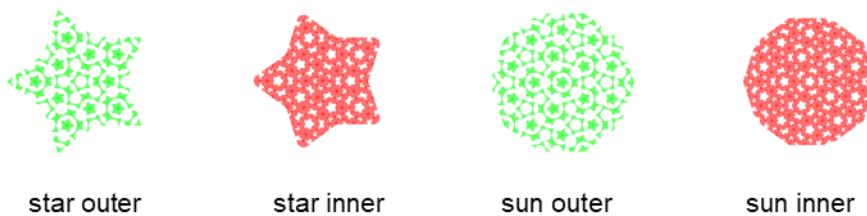
1. Open shape module in cjob
2. Select a cross or rectangle set the width and height
3. Set resolution and beam step size on 0.001 um
4. Select a beam and fill a dose in
5. Press OK
6. Go to the shape, and click on then right menu button, menu will pop-up and select CView.
7. Cview will open and gpf will load. On the bottom left of the CView window is mention which gpf file is opened.
8. From the command prompt, enter ce xxxx, where xxx is the project environment.
9. Enter cp \$PG_PATTERNS\cjob\<mentions gpf file> \$PG_PATTERNS\<pattern name>.gpf to copy the gpf pattern file with markers into the patterns directory.

4.9.7.5 Higher accuracy

By using penrose pattern, there will more borders this will be benefit for accuracy.

Penrose patterns are complex patterns of markers which are used for image correlation. The benefit of these patterns is that the alignment accuracy is improved due to more edges that are determined for measuring the marker position. Penrose patterns are less sensitive to pattern defects like mouse bites and alignment is even possible if several areas of the penrose pattern are missing after processing.

EBPG can create 4 type of penrose star outer, star inner, sun outer and a sun inner.



To create a penrose see 5.5.2 penrose tool

4.10 Marker search with image grabbing

See also chapter images markers.

With the pattern generator images can be grabbed. These grabbed images can be used to correlate and determine the shift of the two images with respect to each other. Together with the two positions where the two images were grabbed, the distance is known. This process can be used as a mark locate function. One of the images (reference) could be a template obtained from pattern data.

Steps

1. Grab the reference image (template image). This can be also an image of (another) marker or an artificial image obtained from e.g. pattern data.
2. Grab an image of the “marker” (marker image)
3. Execute the correlation. The result is a calculated offset between the template image and the marker image.
4. Move the stage. After the move the beam should over the center of the marker.

4.10.1 Image grabbing

The command to grab an image is:

```
pg image grab <offset_x,offset_y> <step_x,step_y> <num_x.num_y>
```

There are more optional parameters; please check the BEAMS reference manual. This command is used to generate an image pixel by pixel. Both the offset and step are in “main resolution steps”. This is very important to realize. That means that if a different resolution is chosen, the step size might change. The total pixel size of the image is the product of step and main resolution.

Since we use the pattern generator to create the image pixel by pixel, the speed of this command is strongly influenced by parameters such as IMAGEBASE and IMAGEFACTOR. If these parameters are changing to speed up the command, do not forget to switch them back as it might have negative effects on the lithography. The pixel size for the image grab is the product of the step parameter in the command and the main resolution unit.

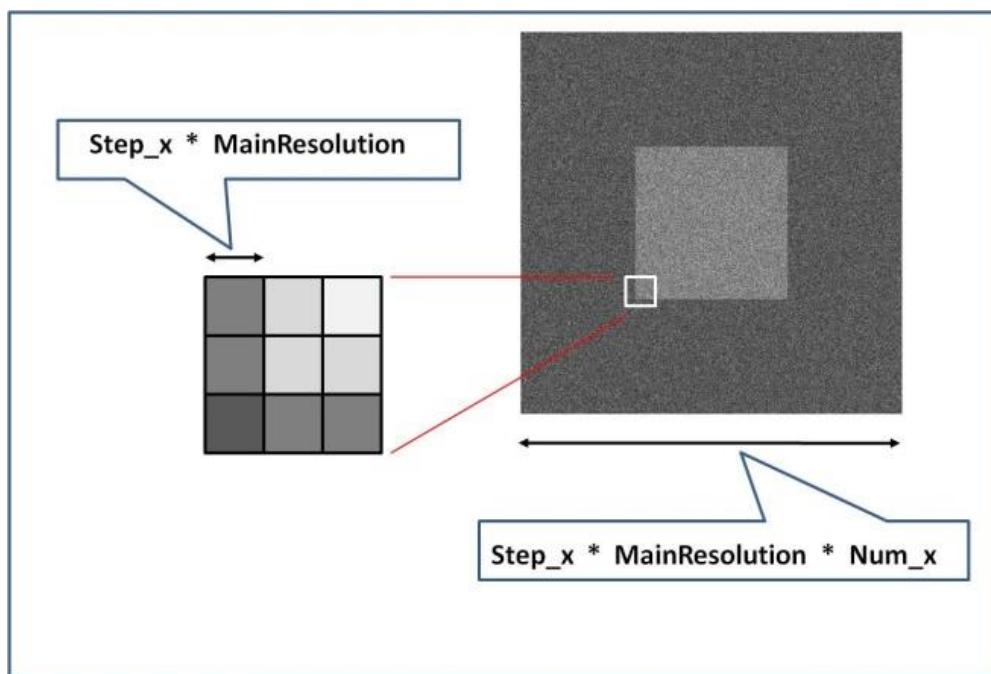


Figure 31 - Grabbed image of a holder marker

Since the image grabs are going to be used as an alignment marker, it is probably best to select the pattern that needs to be written first and determine the main resolution of that pattern. Use the output of the “pg info select pattern” command to obtain the value for the main resolution. In the example below it is 5nm

```

Pattern name      : hexa_small_dot.gpf
File name        : /home/pg/patterns/hexa_small_dot.gpf

Format           : GPF: internal memory mode
Shape Type       : TRAPEZIUM
Number of Blocks : 1
Main field placement : MEANDER
Number of Main bits : 16
Number of Sub bits : 14
Maximum MSF     : 256
Number freq. factors : 1
Minimum frequency : 1.000000
Maximum frequency : 1.000000
Pixel time       : 1016000.000000 (1.016e+06)
High Tension      : 100  kV
Number of blocks  :          1 ,          1
Block size        : 320.00000000 um, 320.00000000 um
Pattern size      : 149.86000000 um, 149.86000000 um
Reference shift   : 0.00500000 um, 0.00500000 um
Resolution        : 0.01000000 um
Beamstepsize      : 0.01000000 um
Main resolution    : 0.00500000 um
Trap resolution   : 0.00047619 um
  
```

The “pg image grab” command generates 2 files. One file with the measured values (unsigned 16 bit integers) for the video levels of each pixel. This file has the “img” extension. The second file is a text file with information about the image grab. That file has a “txt” extension. Both files are needed for the correlation.

To view the img file use command ij. This will open a window select in menu File – Import – Raw... A

window with title Open Raw... will open, select the img file. Set the following options:

- Image type: 16-bit Unsigned
- Width and height : <Image size [exels]> (can be find in txt file, Figure 32)
- Offset to first image: 0 bytes
- Number of images 1
- Gap between images: 0 bytes
- Check only Little-endian byte order.

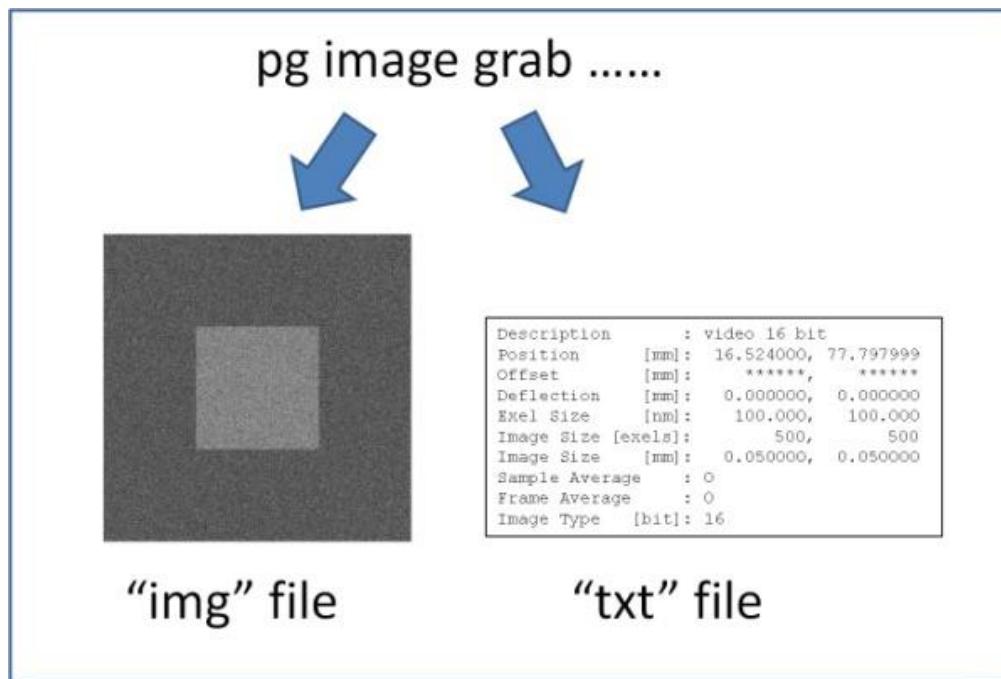


Figure 32 - Two files generated per image grab

4.10.2 Image correlation

The command to correlate two image is

```
pg image correlate <marker image> <template image>
```

The output of this command is a calculated offset between the 2 images. For each of the 2 images we need both the image file and the text file to be present.

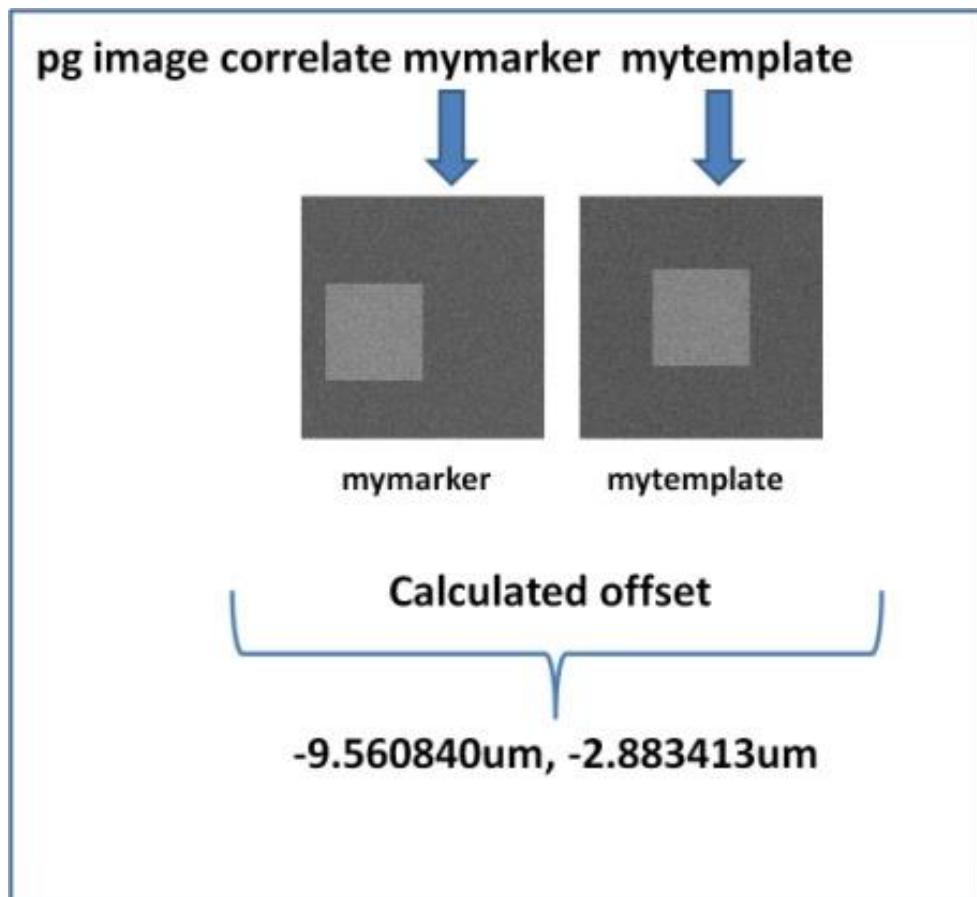


Figure 33 - The output of a correlation command is an offset

4.10.3 User defined marker search

This feature in and by itself adds a lot of flexibility and power to the EPG. The result of a successful marker search of an alignment marker is nothing more than a very accurate stage position. After a successful marker search the stage is moved such that the centre of the marker is under the non-deflected beam. For a typical alignment and distortion scheme multiple markers are to be located. For each of these we go through the following steps. Another issue which might sound trivial is very important: If a marker is not found, that position is not included in calculation for alignment and distortion.

standard marker search

- Move to expected position.
- Perform marker search routine with beam. This determines center of marker
- If no error occurs, continue with following step.
- Move stage to centre of marker.

This means:

After a successful marker search: Stage position is at marker position.

4.10.3.1 User defined marker search

The whole idea behind the user defined marker search is to allow the user to write a routine to determine the centre of a marker that the tool cannot find using its native software commands. This feature can also be used to add functionality to a marker search maybe as simple as printing extra

information for each marker search. The method explored in this document uses image grabs and image correlations to determine the centre of a marker. This “marker” can be anything as long as the same feature is repeated on the substrate.

Equally important is the fact that when using the user defined marker search, the user is responsible for letting BEAMS know whether a marker search was successful or not. That way the results of the failed marker search will not be used in any calculation. We use the return value of the script or program to determine success or failure.

4.10.3.2 The mechanism

The “joy” marker has been given this extra functionality of user defined marker search. In its original design the “joy” marker was meant to provide means for manual marker searches. The functionality has been expanded to allow user written routines instead of manual interaction. An environment variable is used to link the “joy” marker to a script or program. That environment variable is PG_USER_JOYMARKER. Then, every time when a “joy” marker is being located this script or program is being activated.

```
> cat usermarker1

#!/bin/bash
#
# script: /home/pg/scripts/usermarker1

pg move marker
echo "found holder marker"

> export PG_USER_JOYMARKER=/home/pg/scripts/usermarker1
> pg move marker joy

found holder marker
```

The only thing we would need to add to the script listed in example 1 would be a method to let BEAMS know after execution of the script whether the script ran successful or not. We will use the “exit” command for this in a bash script. A return value (via “exit” command) of “0” indicates success. A return value of “non-zero” indicates failure.

Now we can create relatively simple scripts with a combination of BEAMS and bash commands to provide us with the functionality of a marker search. This can also be accomplished using Hill_Shelly, the C-libraries of BEAMS, in conjunction with a C program.

In example 2 we are looking for a marker named “abc”. If that marker definition is not present the script will fail and effectively the marker search will fail. The combination of “trap” and “exit” command allows us to let BEAMS know when this user defined marker search has failed. A very simple implementation of user defined marker:

```
#!/bin/bash
#
# script: /home/pg/scripts/usermarker2

error_handler()
{
    echo "error occurred"
    exit 1
}

trap 'error_handler' ERR
mvm --rel 0,0 abc
echo "marker found"
exit 0
```

We can now extend the user defined marker search to include image grabbing. We have to grab a template image once and for each subsequent marker search we need to grab our marker image and correlate it with the template image. If these two steps are successful, the stage needs to be moved by the offset calculated with the correlation. The flow chart below shows all these steps.

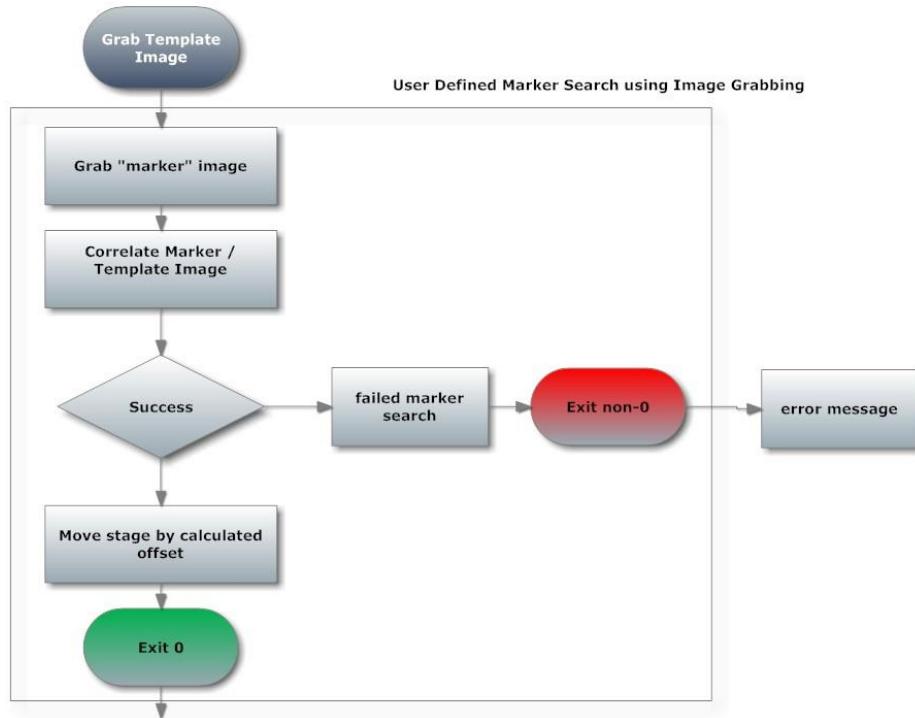


Figure 34 - Flowchart image grab marker search procedure

A script to encompass the described steps could look like this:

```

#!/bin/bash
#
# script: /home/pg/scripts/image_marker
error_handler()
{
    echo "error occurred"
    exit 1
}
trap 'error_handler' ERR
# grab the image
pg image grab
# correlate
pg image correlate
# move the stage
pg move pos
exit 0

```

4.10.4 Short guide line for image marker search

Step 0. Make sure you are in the right environment.

Step 1. Create an image of reference marker

Go to centre of the image marker, make sure it is the centre. If it is possible to do standard marker search.

Set the resolution equal to the pattern.

> pg select pattern <pattern_name>

Grab the image
 > pg image grab 0,0 100,100 512,512 refmark # scan size is 100 * 512 / resolution [um]
 Check the image
 > ij # Select from the menu : file – import – raw... - Select img file – Set options: type: 16
 unsigned, Width: 512, Height: 512; Offset: 0, Number of images: 1, Gap: 0 , Check only Little-
 endian byte order.

Step 2. Create JOY marker

> pg info marker ident “*”
 If there is a JOY marker dust not exist, then create one.
 > pg marker create rectangle positive 20,20 JOY # shape, size and background level is not
 imported

Step 3. Select in the CJOB, by marker search type JOY marker

Step 4. Add ini file, a line to run a script during JOY marker search
 export PG_USER_JOYMARKER=\$PG_SCRIPTS"/imgcorr"

Step 5. Create a script for JOY marker search

Go to the scripts directory and create with a text editor following file imgcorr
 > gedit imgcorr
 Add in the following lines.

```
#!/bin/bash
. script_init_template

position1=`mpg tab`  

pg image grab 0,0 100,100 512,512 imgmark # grab image (use same value as refmark)

trap '' ERR # Job will continue by error message

# good contrast then (check in for hand )
# correlation and movement
pg move pos `pg image correlate imgmark refmark` /relative

# if bad contrast then
position2=`pg image correlate imgmark refmark | head -2 | tail -1`  

pg move pos `pg update ${position2} "+" ${position1}` /relative

echo "Marker Calc: " ${position1} "Found: " `mpg tab`  

script_init_template # Switch back ON, stop by error
```

Save and make it executable

> chmod +x imgcorr

4.11 Marker search templates

The marker search requires marker information. To define markers, template tables with default marker search parameters are available and added to the marker definition.

The values in the templates can be changed with commands e.g.:

```
> pg marker table_template holder fsstp 8
> pg marker table_template topo mlvtol 180
> pg marker table_template user contra 90
```

The default values can be restored to the hard coded BEAMS values e.g.:

```
> pg marker table_template holder /reset
> pg marker table_template topo /reset
> pg marker table_template user /reset
```

A newly created marker, will copy the values from the “user” template table e.g.

```
> pg marker create rect pos 20,20 new_mark
```

Note that it is not allowed to create new markers with the name “holder” or “topo”.

To list the marker parameters for the new marker, use the command e.g.:

```
> pg info marker ident new_mark
```

or

```
> pg info marker ident new_mark /test=par
```

(for more options see BEAMS reference manual).

To change a parameter of the new marker use the command e.g.:

```
> pg marker set new_mark contra 95
```

To reset all parameters for a marker to the template values, use the command e.g.:

```
> pg marker reset new_mark
```

To reset the parameters from ALL USER defined markers to the values in the user table_template the command is :

```
> pg marker reset "*"
```

This will however not reset the tables for holder and topo. They can be reset by:

```
> pg marker reset holder      (values from holder table_template are copied)
> pg marker reset topo       (values from topo table_template are copied)
```

To set new values for the holder or topo marker, use the command e.g.:

```
> pg marker set holder contra 95
> pg marker set topo israd 100
```

The parameters are saved in global data. This means that after a change you need to save the global data, make sure that after a hot- or coldstart the new values are in place

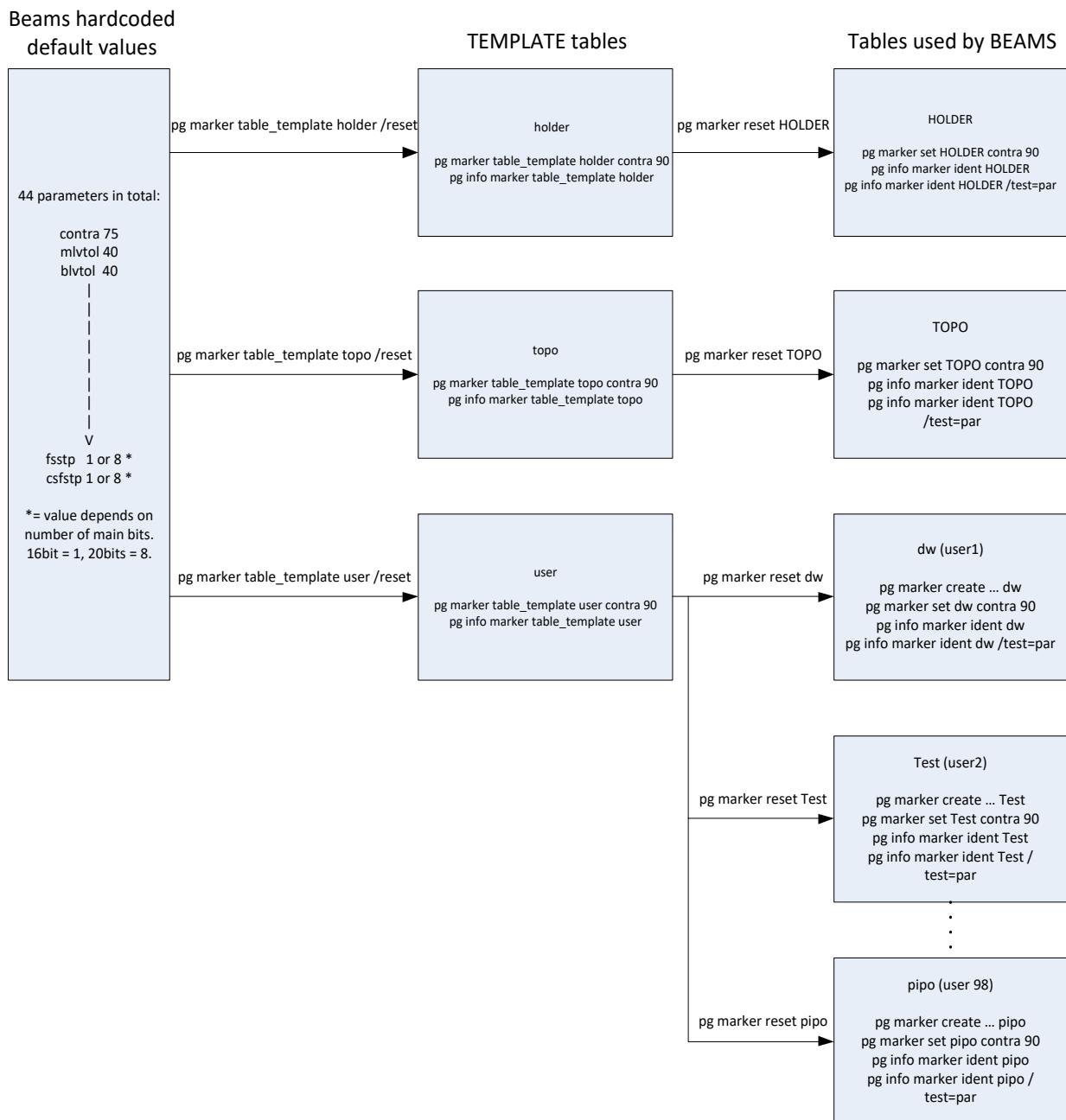


Figure 35 - Flowchart image grab marker search procedure.

5. PATTERNS

5.1 Writing strategy

Electron beam lithography tools have a limited deflection, which depends on the high tension. To obtain sufficient speed as well sufficient accuracy, a large, slow, main deflection is combined with a small, fast, sub field deflection. Usually patterns are much larger than a single main deflection field. Therefore patterns must be fractured into main fields. Due to the double deflection system, shapes are only exposed with the fast sub deflection and a main field must be further fractured into sub fields / shapes. This action is usually done off-line with a data converter (e.g. CATS or LayoutBEAMER).

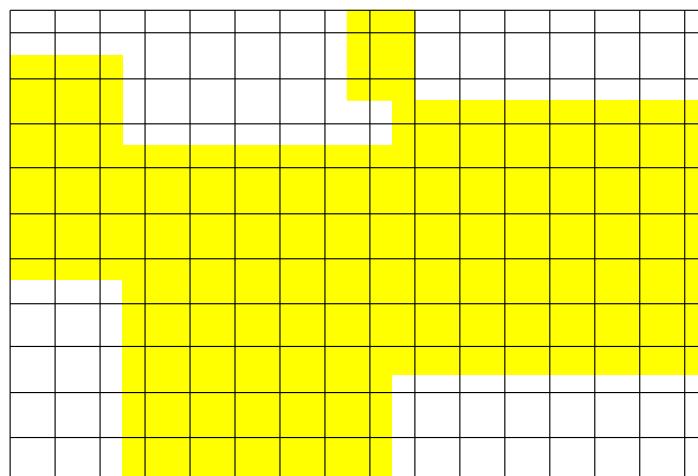


Figure 36 - Example of a main field fracturing

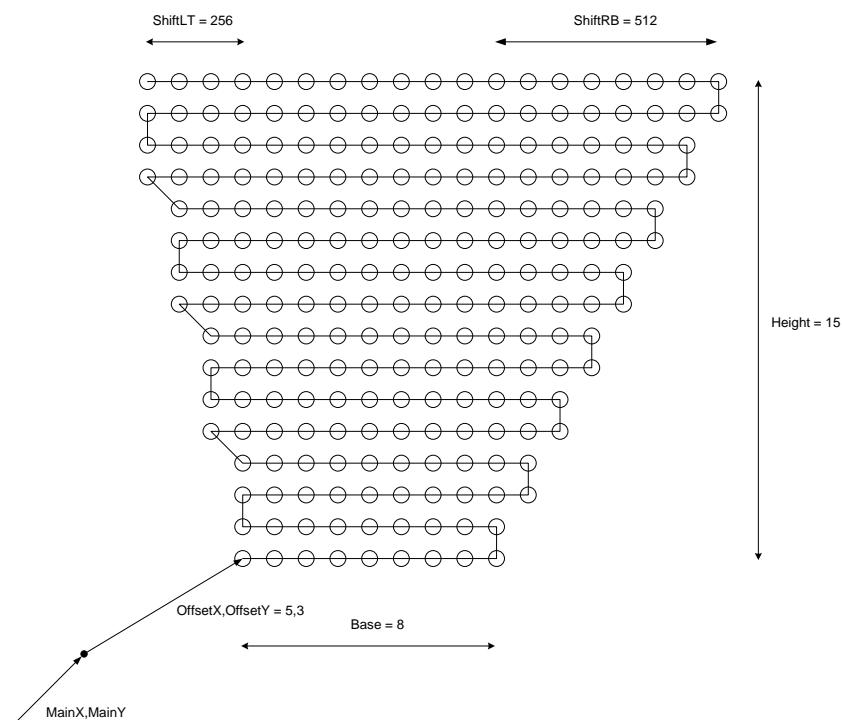


Figure 37 - Example of an exposure of a shape

5.2 GPF format

For the EBPG5000 Plus series the Generic Pattern Format (GPF) has been developed.

1. Storage of data which changes
2. Main field and subfield repetitions with two independent vectors (x,y)
3. X or Y oriented meander scan direction of exels in shape (e.g. Y-oriented trapezoids for wave guides)
4. Structures
5. Support of bitmaps and sequences
6. Flexible for future changes of DAC sizes

The GPF format is a binary format. However, an ASCII equivalent (GTX) has been defined and tools are available to convert from GPF to GTX and vice versa.

Both, the binary as well as the ASCII format has been described in the specification document Generic Pattern Format, Software Product Description [3] and is distributed with the BEAMS control software kit.

5.3 Exposure of GPF patterns

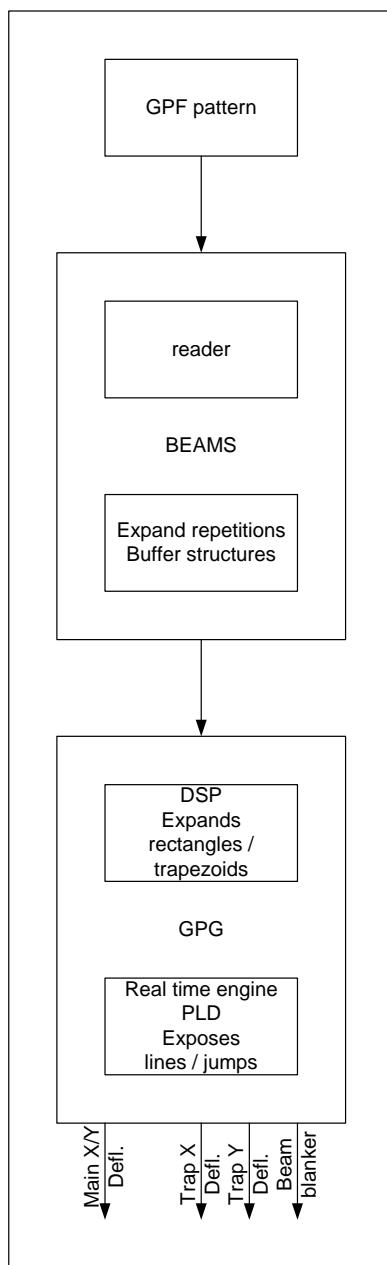


Figure 38 - Schematic overview of the pattern data handling during exposure

5.4 Estimate exposure time of GPF pattern

gpgtime tool

5.5 Data preparation

In most cases dedicated data preparation tools are used from external vendors to generate from the design the GPF file which can be handled by the tool. The GDSII input format is the most common design format, but others are available. See Figure xxx. The ASCII equivalent of GDSII is the CTXT format.

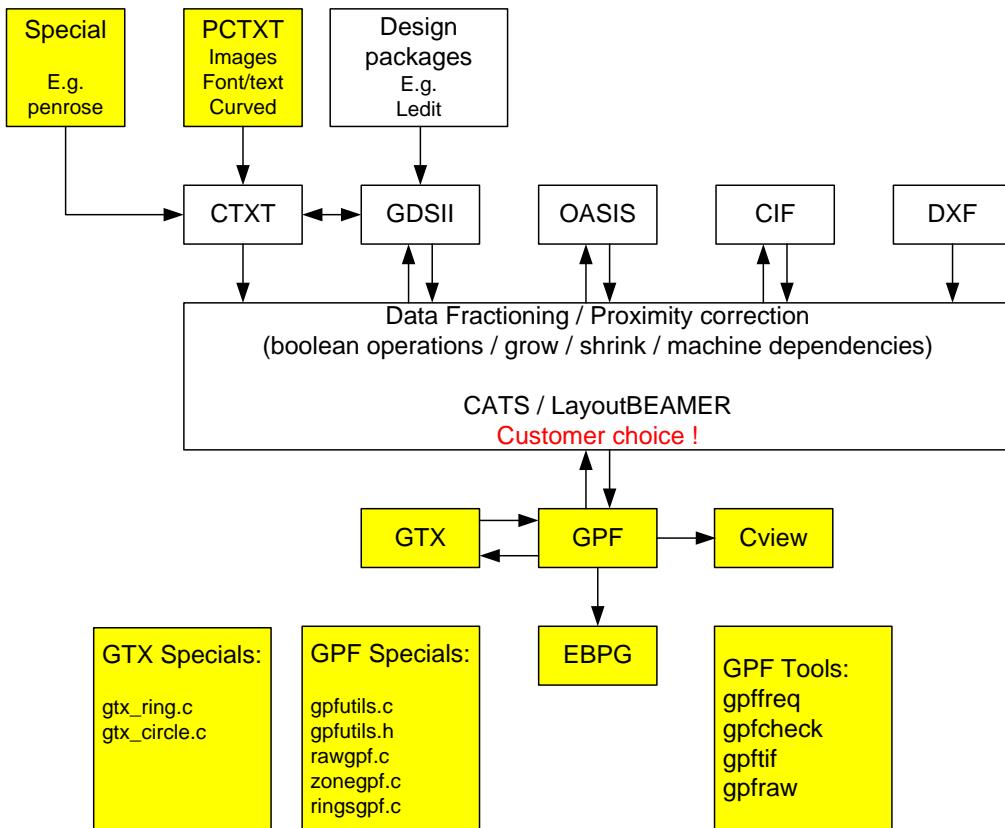


Figure 39 - Overview pattern data preparation

5.5.1 pctxt tool

The GDSII format, and accordingly the CTXT format has limitations. In particular it doesn't support rings or circles. Therefore we have developed a tool, which is a pre convertor. This tool converts some additional primitives into a CTXT compliant format file. The extra primitives are:

FONT

This primitive defines a bitmap font (linux style), to be used for successive TEXTx-elements

TEXTL, TEXTR and TEXTC

These primitives define a text string, using the type of TEXT-primitive position of the string and size of the character as a parameter

BOUND

With this primitive an area can be define in a domain (x or y direction) and two boundary functions. The area will be approximated with a “rectangular” polygon. The resolution is a parameter of this primitive. The functions are defines as a string, using the C-programming style and conventions.

CURVE

The CURVE primitive is identical to the BOUND primitive, except that it does NOT generate a polygon, but just rectangular boxes

IMAGE, IMAGENEG

This primitive reads an image from a file. The package ImageMagick's library is used to read the images.

The pctxt program is distributed as part of the cview package, which is part of the BEAMS control

software distribution kit. For detailed information and examples, use the command

```
> pctxt -h
```

5.5.2 penrose tool

Penrose patterns are used to create markers which are suitable for correlation methods for detection. see chapter xxx.

To generate these patterns is not a quick task: therefore we have added a tool, also as part of the cview package to create a ctxt file. The –h option provides detailed information

```
> penrose -h
```

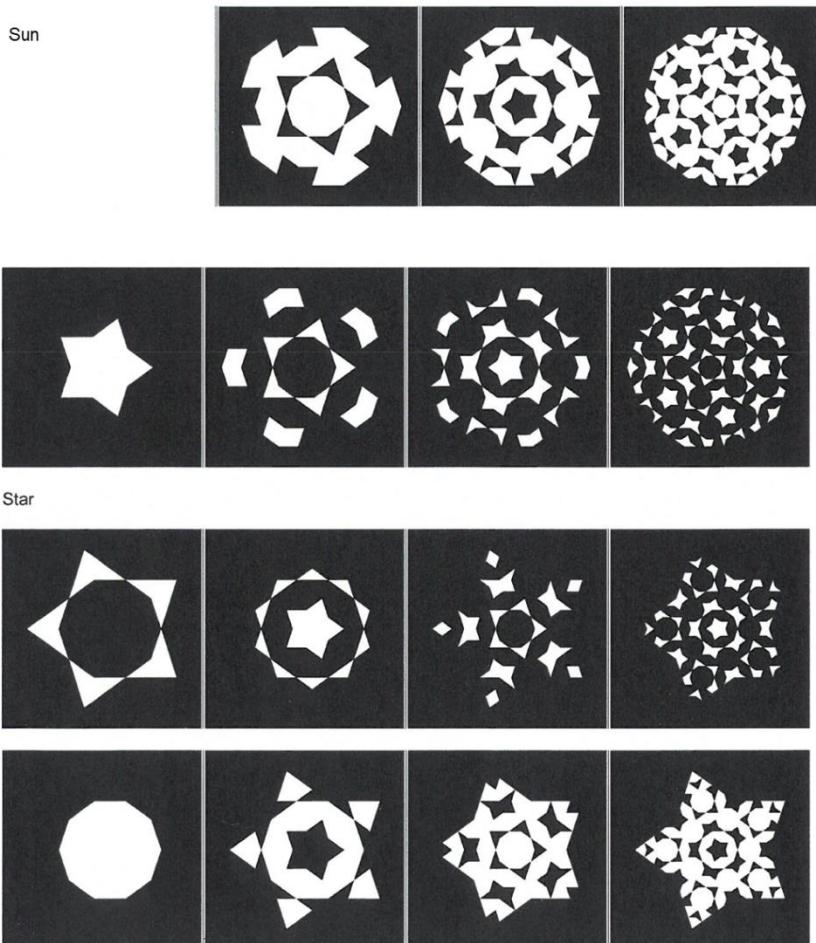
This program generates a CTXT (cq TXL) file for a penrose tiling

It uses a tiling (AXIOM) and the deflation method to generate sub tilings (see wikipedia on penrose tilings)

usage: penrose <model> <angle> <radius> <depth>

<model>	sun star
<angle>	orientation angle (degrees)
<radius>	radius (micron) of penrose tiling
<depth>	depth of deflation

User creation of GPF patterns



5.5.3 Hill Shelly Library functions

zone plates

concentric rings

5.5.4 gpf vtx and gtxgpf tool

gtx_circle.c

gtx_ring.c

5.6 GPF tools

5.6.1 ppd tool

This tool displays the directory contained by the environment variable PG_PATTERNS (not PG_PATTERNS_PATH!). The -h option provides the help function for this tool:

```
> ppd -h
This program displays the patterns available with extension .gpf and .GPF pointed
to by the environment variable $PG_PATTERNS.
The command takes as a parameter a file specification. No extension should be
specified, as default is .gpf or .GPF. Wildcards are allowed.
```

The command accepts the following options:

```

-h      displays this help
-i      displays pattern parameters:
        resolution, beam step size,
        main resolution, sub field resolution,
        field size x, field size y,
        pattern size x, pattern size y
-m      displays machine configuration

```

5.6.2 gpffreq

The gpffreq tools is used to display and/or modify the frequencies used in a GPF pattern file. The **-h** option provides a detailed help:

```

> gpffreq -h
usage:
    gpffreq -h
    gpffreq <gpf file>
    gpffreq <freq file> <new gpf file>
    gpffreq <gpf file> <new gpf file> <list>
parameters:
    <old gpf file> : gpf file to obtain frequencies
    <new gpf file> : gpf file with new frequencies
    <freq file>     : freq file containing frequency changes
    <list>          : list of pairs of clock numbers <old>,<new>
    <old>           : old clock number (hexadecimal or relative)
    <new>           : new clock number (hexadecimal or relative)
                    hexadecimal: 0x00000001 - 0xFFFFFFFF
                    relative:    0.0000000005 - 1.9999999995

options:
    -h           : displays this help information

```

DESCRIPTION

This program obtains all frequencies used in the gpf file up to the number of frequencies as specified in the header.

Optional this program can be used to change one or more frequencies. The required frequency file is most conveniently obtained by saving the output of this program for checking frequencies:

```
gpffreq <gpf file> > <freq file>
```

New frequencies can be specified by editing this file and adding to the frequency line, whose frequency needs a change the relative frequency. Note that at least sufficient digits need to be specified to change relative frequency. Note that sufficient digits need to be specified. The gpf file specification is the name of the new gpf file. The source gpf file is obtained from the freq. file (second line).

```
gpffreq <freq file> <new gpf file>
```

example of the contents of a frequency file:

```
GPFFREQ version y09_01a_Thu Jun 12 21:55:29 CEST 2008
wafer_pec.gpf
```

```
Number of frequency factors 32
Minimum frequency factor 0.5702963360 (0x48ff7868)
Maximum frequency factor 1.4170582662 (0xb5622a4f)
```

```
Scanning the pattern file for frequency factors .....
```

```
Frequency factors:
  1      0.5702963360      (0x48ff7868)
  2      0.5872889031      (0x4b2c4864)
  3      0.6047876994      (0x4d69aeef)
  4      0.6228079679      (0x4fb82be7)
  5      0.6413651616      (0x521840ed)
```

```

6      0.6604753295    (0x548a74a2)
7      0.6801547767    (0x570f4fc0)  0.6901547767
8      0.7004206893    (0x59a76299)
9      0.7212903933    (0x5c533e5d)
10     0.7427820195    (0x5f137b31)
11     0.7649138961    (0x61e8b2d4)
12     0.7877052515    (0x64d38693)
13     0.8111757762    (0x67d49b9b)
14     0.8353454988    (0x6aec99ef)
15     0.8602355015    (0x6e1c3269)
16     0.8858670304    (0x71641742)
17     0.9122623401    (0x74c5032a)
18     0.9394440702    (0x783fb40b)
19     0.9674357153    (0x7bd4eefb)
20     0.9962614398    (0x7f857eaf)  0x80000000
21     1.0259461291    (0x835233e8)
22     1.0565151167    (0x873be329)
23     1.0879950817    (0x8b436c3f)
24     1.1204129569    (0x8f69b118)
25     1.1537967860    (0x93af9cf3)
26     1.1881752270    (0x98162037)
27     1.2235781052    (0x9c9e3515)
28     1.2600358455    (0xa148dac6)
29     1.2975798748    (0xa61718eb)
30     1.3362426520    (0xab09ffcd)
31     1.3760573529    (0xb022a5b8)
32     1.4170582662    (0xb5622a4f)

```

Alternatively, a list of pairs of clock numbers can be specified on the command line. If the old value dose not occurs in the old file, the replacement is ignored.

```
gpfreq <old gpf file> <new gpf file> <old1,new1> <old2,new2> .....
```

5.6.3 gpfcheck

The gpfcheck tool checks the pattern for validity of the data, e.g. that the deflections are within their resp. range of the DAC. The -h option provides detailed information

```

> gpfcheck -h
usage: gpfcheck -h <gpf_file> [<log_file>]
<gpf_file> : gpf file that will be checked.
<log_file> : optional, output file name.
                  If not specified gpfcheck.log is created
-h          : print this help.

```

5.6.4 gpftif and gpfraw

The gpftif and gpfraw tools convert a pattern in resp. a tif image or a raw data image file. The -h options provides detailed information:

```

> gpftif -h
usage: gpftif [-h] <gpf file> [<x0>,<y0> <xs>,<ys> <xn>,<yn>]
<gpf file> : gpf file that will be converted.
                  default extension .gpf
                  default exel size resolution unit
-h          : Print this help.
optional parameters
<x0>,<y0> : offset of centre of image to centre of pattern (micron)
<xs>,<ys> : exel size (micron)
<xn>,<yn> : number of points
> gpfraw -h
usage: gpfraw [-h] <gpf file> [<x0>,<y0> <xs>,<ys> <xn>,<yn>]
<gpf file> : gpf file that will be converted.
                  default extension .gpf
                  default exel size resolution unit
-h          : Print this help.
optional parameters
<x0>,<y0> : offset of centre of image to centre of pattern (micron)

```

```
<xs>, <ys> : exel size (micron)
<xn>, <yn> : number of points
```

Exposing a substrate

5.7 Field scaling

The fabrication of DFB laser requires precise pitch control of gratings. Scaling is often used to change the pitch of gratings for fast throughput. Within a main field, the pattern can be scaled. The scaling is done by summing the output of corrections DACs with the output of the deflection DAC. The resolution of the scaling is approximately 0.6 ppm/bit for EBPG5150 and 5200. For example, if the nominal grating pitch is 202 nm, the smallest possible pitch changes are 0.00012 nm. The maximum scaling is $\pm 1\%$ with relative to the center of the field. The scaling factor is a 24 bit signed integer (-8,388,608 to 8,388,608). The value represents the scaling percentage multiplied by 10^6 . For instance, a value of 1,000,000 corresponds to 1% scaling. A value of -1,000,000 corresponds to -1% scaling.

System requirements

All EBPGs can use scaling.

Command

The command is:
Scale X|Y <value>

How to set up scaling

The typical way to set up scaling is described as below:

1. Design gratings with a nominal pitch, which is to be scaled later.
2. Fracture the pattern, upload it to the EBPG machine.
3. Use "gpfgtx" command to convert the pattern to an editable text file.
4. Insert "scale" command into this text file.
5. Use "gtxgpf" command to convert the text file back to gpf.
6. Use CView to check the pattern to make sure that the desired scale value is applied.

Example

Assume we need to expose 300 horizontal lines with the pitch of 202.404nm on Y. These lines are 25 μ m long.

We will use 202nm as the nominal pitch.

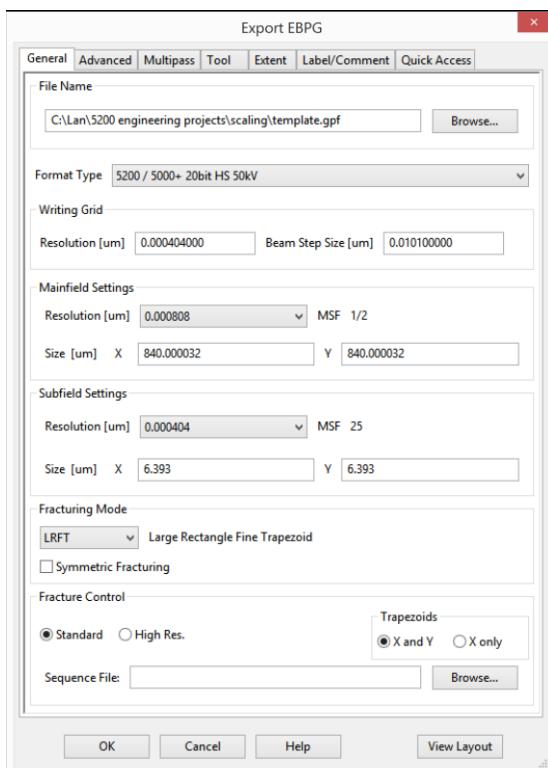
The pitch difference is

$$202.404 - 202 = 0.404 \text{ nm}$$

The scaling factor on Y

$$(0.404 \div 202) \times 100 \times 10^6 = 200000$$

We first create a GDSII file for the gratings with the pitch of 202nm. Then we use BEAMER to fracture the pattern, as shown below. For the purpose of demonstration, we will make these lines 3-pass lines.



Upload this pattern "template.gpf" to the machine, then use "gpfgtx" command to convert the pattern to a text file:

```
>gpfgtx template.gpf template.gtx
>gedit template.gtx
```

The file template.gtx looks like this:

```
!GPFGTX version y09_12a build: Fri Apr 12 10:16:45 CEST 2019
GTX "1.0.0"

HEADER
Title "Generic Pattern Format, c RAITH nanofabrication"
Version "1.43"
MainFieldResolution 0.0008080000000000,0.0008080000000000 ! (um)
SubFieldResolution 0.0004040000000000,0.0004040000000000 ! (um)
Resolution 1,1 ! 0.000404,0.000404 (um)
BeamStepSize 25,25 ! 0.010100,0.010100 (um)
PatternSize 61881,149575 ! 24.999924,60.428300 (um)
MainFieldSize 2079208,2079208 ! 840.000032,840.000032 (um)
SubFieldSize 632,632 ! 6.383200,6.383200 (um)
HighTension 5000000 ! (mV)
ShapeTypes TRAPEZIUM
SubFieldPlacement LOWERLEFT
NrMainFields 1
MainFieldPlacement MEANDER
FractureStyle SUBFIELD SUBRESOLUTION
NrMainFieldBits 20
NrSubFieldBits 14
MaxMSF 8192
MinFreqFactor BIN 0x80000000 !1.000000
MaxFreqFactor BIN 0x80000000 !1.000000
NrFreqFactors 1
PixelTime 2228400 !2.2284e+06
FileSize 1192
HeaderOffset 512
IndexOffset 1152
StructureOffset 0
SequenceSize 0
MarkerOffset 0
```

```

StructureSize          0
MarkerSize            0 ! 0 markers
PatternName           "template"
CreateDate            "25-4-2019"
CreateTime             "13:50"
ConverterName          "BEAMER"
ConverterVersion       "Layout ENGINE x64 Revision Number 5.07.000 , Dec"
SourceFormat           "CTXT"
END

FIELDOFFSET 0.000000,0.000000

FIELD 1,1 ! field:1 (0.000000 um, 0.000000 um, 0.000000 um) primitives:32
MoveStage   1
Frequency    LSW 0x0000
Frequency    MSW 0x8000
MSF          25
Base          LSW 15425
Height         LSW 50
Main          X LSW 508818
Main          Y LSW 486894
Offset        X LSW 0
Offset        Y LSW 1
Repetition Sub 2 Count 31
Repetition Sub 2 X 0
Repetition Sub 2 Y 500
Repetition Main 1 Count 2
Repetition Main 2 Count 8
Repetition Main 2 X LSW 0
Repetition Main 2 Y LSW 8000
Activate     TRAPEZIUM MRV1 MRV2 SRV2 RXB rb90 lt90 ! first shape
Main          X LSW 508818
Main          Y LSW 558894
Repetition Sub 2 Count 11
Activate     TRAPEZIUM MRV1 SRV2 RXB rb90 lt90 ! second shape
Base          LSW 15525
Main          X LSW 531983
Main          Y LSW 486894
Offset        X LSW 1
Repetition Sub 2 Count 31
Activate     TRAPEZIUM MRV2 SRV2 rb90 lt90 ! third shape
Main          X LSW 531983
Main          Y LSW 558894
Repetition Sub 2 Count 11
Activate     TRAPEZIUM SRV2 rb90 lt90 ! fourth shape

```

END

To add scaling factor into the gtx file, we just need insert the scaling factor into the gtx file, right in front of the first shape's activation. The e-beam system will apply this scaling factor to all the rest of the shapes.

```

Repetition Sub 2 Count 31
Repetition Sub 2 X 0
Repetition Sub 2 Y 500
Repetition Main 1 Count 2
Repetition Main 2 Count 8
Repetition Main 2 X LSW 0
Repetition Main 2 Y LSW 8000
Scale x 0
Scale y 2000
Activate     TRAPEZIUM MRV1 MRV2 SRV2 RXB rb90 lt90 ! first shape

```

Save this new file as template_gtx.gtx, then convert it back to gpf file:

```
>gtxgpf template_gtx.gtx template_gtx.gpf
```

Check this new pattern with CView to see whether the scaling has been applied.

6. BEAMS EXPOSURE PROCEDURES

6.1 Layouts

6.1.1 Simple array's

6.1.2 Exposure area's

6.1.3 Dose update

6.2 Alignment

7. JOB PROCEDURES

7.1 Substrate mapping

Moving the stage around in the standard XY coordinate system between certain locations (e.g. markers) can be a bit tiresome since the XY values are not always “nice”. This can be done in a more convenient and intuitive way using “table positions” and “substrate mappings”. An additional benefit of substrate mappings is that we can scale patterns ($\pm 0.5\%$) without interaction with CATS or LayoutBeamer.

The goal of this document is to introduce the concepts and applications of substrate mappings.

Substrate mappings make use of Coordinate Transformations. This is simply a conversion from one coordinate system to another. In a way when you've been using the EPG for a while you will have used this on the most simple level of a coordinate transformation, a shift.

The Adjust Table Coordinates (atc) command does nothing more than shifting the current coordinate position in such a way that the current table position exactly matches the expected calibration marker position. This is a key point for all the transformations we will be using. As with the atc command, the transformation is based on a set of observed and expected positions. The results of these transformations are called Substrate Maps and we give these maps a name.

These maps can be selected en deselected using their name. When no map is active we say that we are in the “absolute” coordinate system and can be viewed as the “absolute map”.

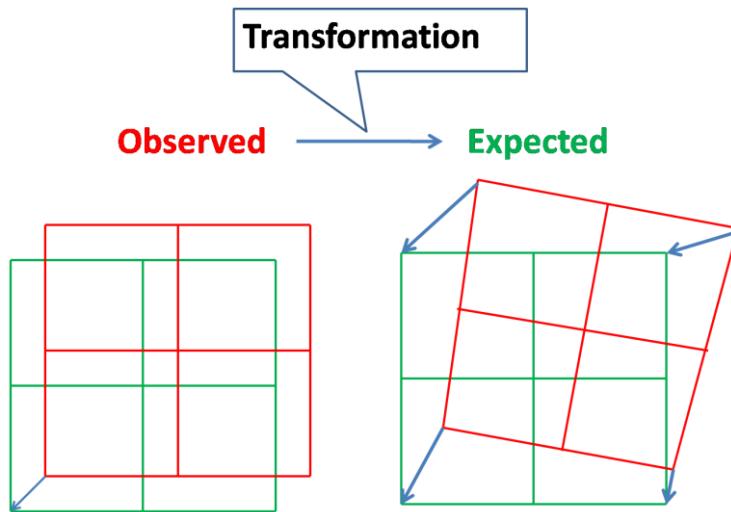


Figure 40 - Transformation

7.1.1 Table positions

To make creating the substrate maps a little bit easier we introduce the concept of Table Positions. These are (X,Y) coordinate pair that have a name and some other attributes that will be discussed later. To create table positions, we use the “pg table_position” command. To list all the defined table positions use the command “pg info table <pos_id>”

```
table_position <pos_id> [<pos_x,pos_y>] [--marker = <marker_id> ]
```

```

pos_id          = specifies name of defined table position
pos_x, pos_y   = Optional position specification.
                  If omitted, current stage position will be used.
--marker=<marker_id> = optionally attaches marker type to this position
-- delete        = deletes particular table position

> pg table_position expected1    10mm,10mm
> pg table_position observed1   11mm,11mm
> pg table position markerpos1  12mm,12mm --marker=p20
> pg info table
  identifier      coord (um)      mode      marker
expected1       10000.000, 10000.000 absolute   -
observed1       11000.000, 11000.000 absolute   -
markpos1        12000.000, 12000.000 absolute   p20

```

7.1.2 Using table positions

We can use these table positions in multiple ways. We can move to these particular positions by name by using the “pg move table_position” command. This by itself might not be that exciting but if the table position has a marker attribute, we can actually move to that position, locate the marker (using the –locate option) and update the position with the coordinates of where we located the marker (using the –update option) effectively making that table position equal to the marker position.

`move table_position <pos_ident> [option]`

- locate Locates the marker defined in the marker attribute for that position.
- update Updates the table position identifier with the position of the located marker.

Another way to use these table positions is as a set of observed and expected table positions to create a substrate map. There are actually multiple ways to create these substrate maps and we will discuss them next.

7.1.3 Three ways to create a substrate map (transformation)

7.1.3.1 Using Expected and Observed positions only.

Only the definitions of the table positions involved are needed. The stage will not move to those positions or do anything else. The only thing the “calculate substrate” command does is calculate a substrate map, it performs the transformation.

`calculate substrate <map_ident> e1,o1 [e2,o2 [e3,o3 [e4,o4]]]`

Calculates the substrate mapping coefficients based on the expected (e) and observed (o) positions and stores them in a substrate map with the specified name.

`information calculate substrate [<map_ident>]`

Displays the calculation of the substrate map specified or of the current map if no map_ident was specified.

```
> pg table eg1 50mm,50mm
> pg table eg2 70mm,70mm
> pg table og1 49.990mm,49.990mm
> pg table og2 70.010mm,70.010mm

> pg calculate substrate stretch eg1,og1 eg2,og2

> pg information calculate substrate stretch

Map      : stretch
order   : 3
origin  : 0.000000_mm, 0.000000_mm
translation at present position (5.353872_mm,5.083477_mm)
: -5.464613e-02_mm, -5.491652e-02_mm

Markers  :           expected (absolute)          observed (absolute)
             X [mm]           Y [mm]           X [mm]           Y [mm]
1       : 50.000000000 , 50.000000000 49.990000000 , 49.990000000
2       : 70.000000000 , 70.000000000 70.010000000 , 70.010000000

Transform:          to Absolute          from Absolute
                  X           Y           X           Y
offset  : -6.000000e-02 , -6.000000e-02 +5.994006e-02 , +5.994006e-02
scale   : +1.000000e-03 , +1.000000e-03 -9.99010e-04 , -9.99010e-04
rotation: +0.000000e+00 , -0.000000e+00 +0.000000e+00 , -0.000000e+00
keystone: +0.000000e+00 , +0.000000e+00 +0.000000e+00 , +0.000000e+00

Binary DAC corrections at present position:
          main     trap    pull-in
x gain  :     408      77      54
y gain  :     410      74      53
x rot   :       0       0       0
y rot   :       0       0       0
x key   :       0       0       0
y key   :       0       0       0
```

7.1.3.2 Using Markers

Creating the substrate map based on markers looks a lot like the previous command, but now we do two things differently:

1. We define the expected table positions with the “--marker” option
2. We do not define observed positions. The variable names used for the observed positions are actually place holders for the positions where the markers are actually found.

7.1.3.3 Setting Coefficients directly.

If the coefficients are available through other means we can set these coefficients and create a substrate map that way. This voids the need for expected and observed positions. The coefficients are applied in following way:

$$X' = \text{offsetX} + (1 + \text{scaleX}) * X + \text{rotationX} * Y + \text{keystoneX} * XY$$

$$Y' = \text{offsetY} + (1 + \text{scaleY}) * Y + \text{rotationY} * X + \text{keystoneY} * XY$$

```
adjust substrate <map_ident> e1,o1 [ e2,o2 [ e3,o3 [e4,o4]]]
```

The expected table positions are defined with a marker attribute. BEAMS will try to locate these markers and store the found positions in the “observed variables”, o1,o2 etc.

```
map substrate <map_ident> DELETE
map substrate <map_ident> <X|Y> [OPTION]
```

Either deletes a map completely (DELETE) or sets coefficients (OPTION).

OPTION:

- offset
- scale
- rotation
- keystone

7.1.4 Applying a substrate map

Now that we have created a substrate map (a transformation) we need a way to let the system know to actually use a particular map. For this we use the “pg select map substrate” command. Once selected we can move around as before except that now the coordinate system has been transformed according to our substrate map. Once we are in a different map we can create another map, a map within a map. However, as far as the map calculations go, all mapping coefficients are with respect to the absolute coordinate system

```
select map substrate <map_ident>
deselect map substrate
```

The reverse is also possible, deselecting a map. This will take you back to the absolute coordinate system even if a map has been selected while another map was active.

The substrate maps are not stored in files but directly in global data. That means that there is a limit to the number of substrate maps that can be created. Currently that limit is 20.

To delete a particular map use the “delete” keyword for the “map substrate” command. Use asterisk to between quotes to delete all substrate maps. The table positions that were created in a particular map will be deleted when that map is deleted. Table positions can also be deleted separately.

7.1.5 Table positions and substrate maps

Table positions can be created within an active substrate map. When listing the table position particular table positions might be listed with 1,2 or 3 lines of information. The one line version only reflects table positions that are created in the absolute map and if the current map is also absolute. When 2 or more lines are shown, following rules apply:

The first line always reflects the map in which the table position was created. For the other one or two lines: There is always an entry referring to the current map. There is also always an entry which reflects this table position's absolute position. To create table positions for a substrate map which is not the current one, the command "pg table position" has the "—map" option.

```
> pg info table

identifier          coord (um)      mode      marker

e1                 10000.000, 10000.000  absolute   -
                  6000.000, 6000.000  frommap1
o1                 10000.000, 10001.000  absolute   -
                  6000.000, 6001.000  frommap1
rell               5000.000, 5000.000  shift1    -
                  6000.000, 6000.000  absolute
                  2000.000, 2000.000  frommap1
rel2               8000.000, 8000.000  shift1    -
                  9000.000, 9000.000  absolute
                  5000.000, 5000.000  frommap1
m1                 10000.000, 10000.000  frommap1  p10
                  14000.000, 14000.000  absolute
m2                 20000.000, 10000.000  frommap1  p10
```

clear maps etc.

To list all the maps we can use the "pg info map subs "*" " command. This will give a very long output. It might be better to use the Linux "grep" command to filter out some of this information. Not only will this show which maps are available but also which one is active (selected).

```
> pg info map subs "*" | grep Map
```

```
Map      : shift1
Map      : 2point
Map      : stretch (Selected)
Map      : frommap1
Map      : stretch2
```

```
> pg select map subs stretch2
> pg info map subs "*" | grep Map
```

```
Map      : shift1
Map      : 2point
Map      : stretch
Map      : frommap1
Map      : stretch2 (Selected)
```

```
> pg deselect map subs
> pg info map subs "*" | grep Map
```

```
Map      : shift1
Map      : 2point
Map      : stretch
Map      : frommap1
```

7.1.6 Practical applications

Making a "nice " coordinate system.

When we load a substrate we were automatically bound to the absolute coordinate system. Especially when alignment markers are involved we could make our coordinate system a little bit nicer by making the markers reflection symmetric with respect to the origin (if markers are in rectangular configuration).

Let's say the markers are close (but not exactly) on 10mm,10mm 30mm,10mm 30mm,30mm and 10mm,30mm. We want these positions to be +/- 10mm, +/- 10mm.

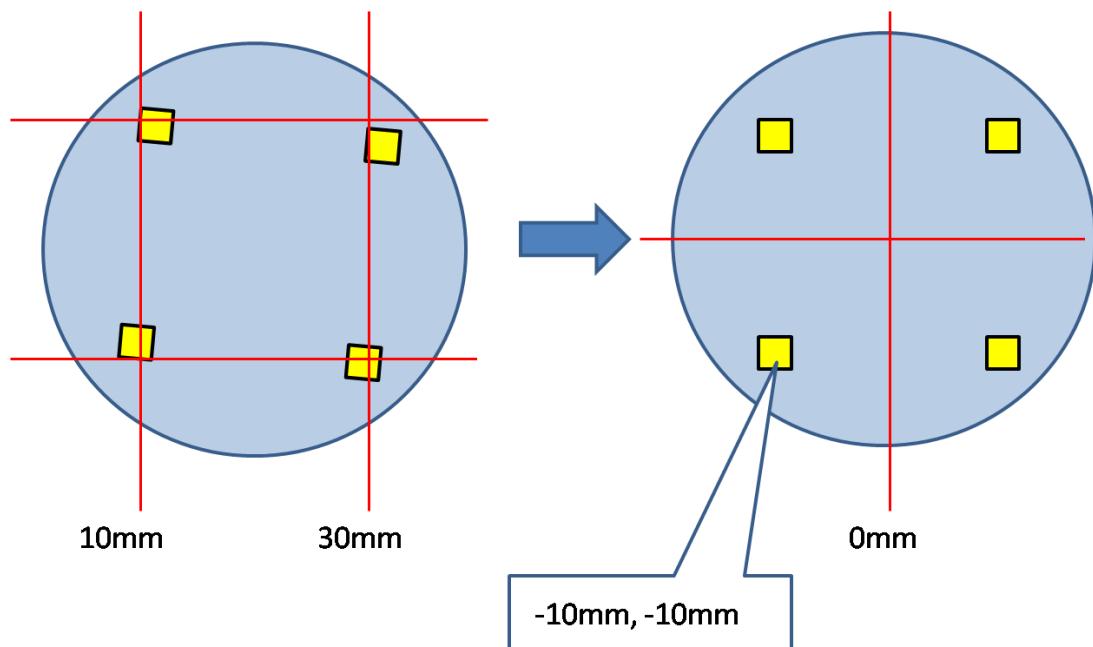


Figure 41 - Transformation

Step 1: Declare table positions for the expected marker positions.

Step 2: Locate the markers and update the positions as the actual marker positions

Step 3: Use the updated positions of the located markers and make them to be the “nice” coordinates.

```

pg table_position m1 10mm,10mm --marker=p10
pg table_position m2 30mm,10mm --marker=p10
pg table_position m3 30mm,30mm --marker=p10
pg table_position m4 10mm,30mm --marker=p10

# pg move table_position m1 --locate --update
# pg move table_position m2 --locate --update
# pg move table_position m3 --locate --update
# pg move table_position m4 --locate --update

pg table_position exp1 -10mm,-10mm
pg table_position exp2 10mm,-10mm
pg table_position exp3 10mm,10mm
pg table_position exp4 -10mm,10mm

> pg calculate substrate nice exp1,m1 exp2,m2 exp3,m3 exp4,m4

```

7.1.6.1 Scaling or rotating pattern (slightly).

When a pattern needs to be scaled or rotated slightly (+/- 0.5% scale , +/- 0.2 degrees rotation we can actually use the example of section 3.1 (map called “stretch”). Keep in mind that with any scaling or rotation there will be one point in the coordinate system which will be unaffected. This would typically be the center of the pattern.

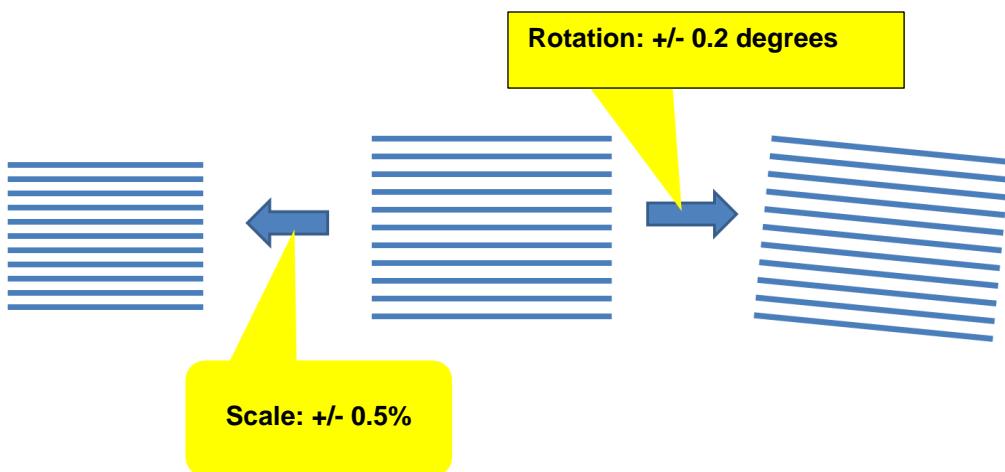
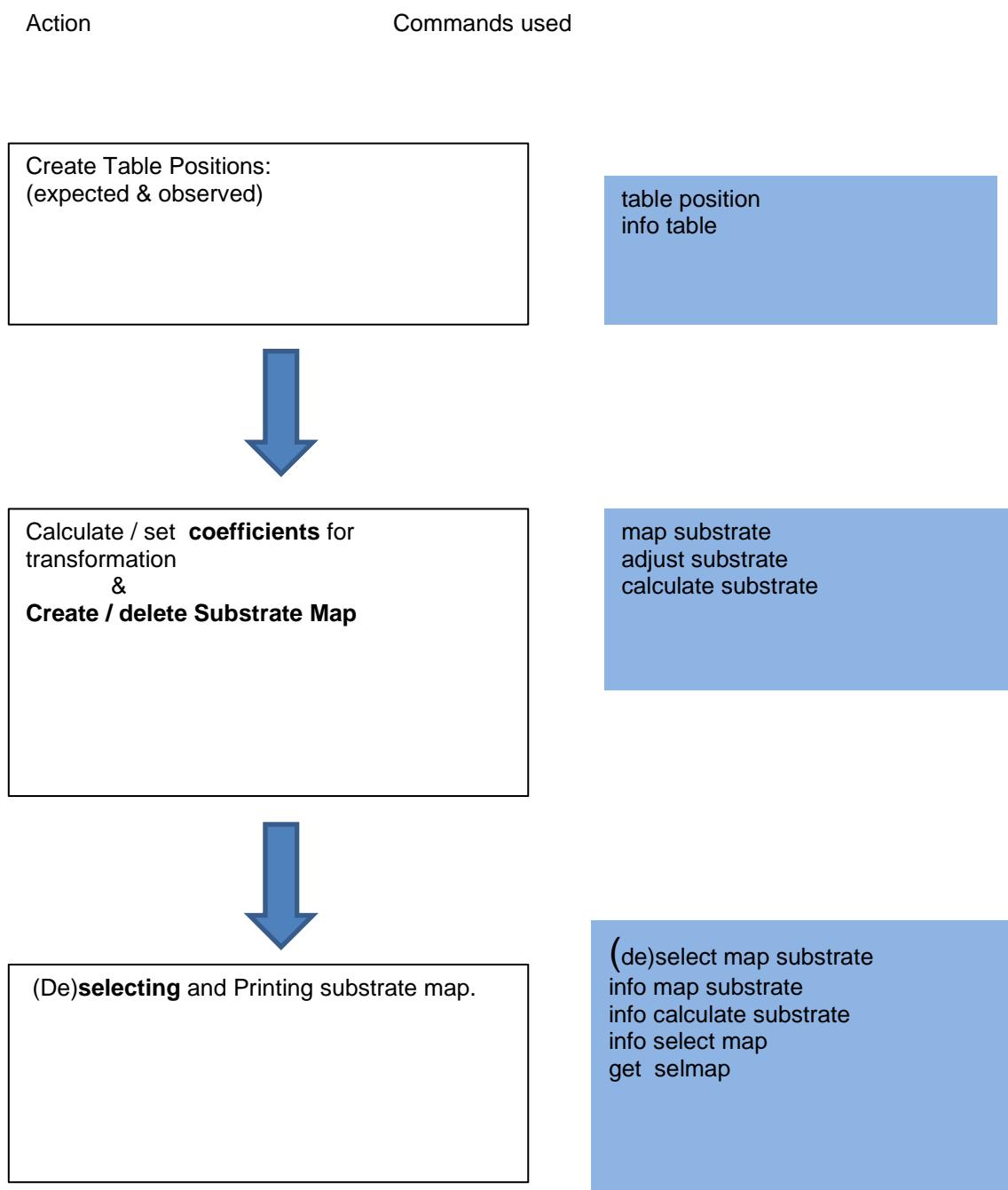


Figure 42 - Transformation

7.1.7 Command overview



7.2 Height mapping

7.2.1 Height measurement before and during writing

For control, measuring and checking the height during a job, there are 2 part. One part is in CJOB to set pre-measurement and other part is measurement during the writing of the job. This can be set in cjob and in ini-files.

The pre measurement can be set in in cjob under tab exposure, there you have 4 option none, check, current map and measure map. Witch are showed in Figure 73. During the writing can the height measurement behavior be set by the parameters HMAPINV and HMAPMODE these are placed the ini file.

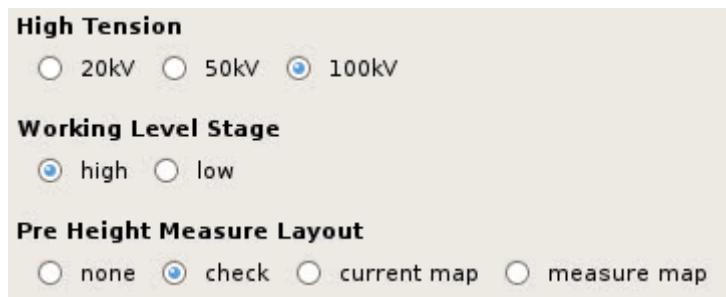


Figure 43 - Cjob Exposure options

7.2.1.1 Pre Height Measure Layout:

The 4 options for pre height measure layout:

- | | |
|-------------|---|
| None | No pre height measure check and during the writing set HMAPINV and HMAPMODE values will be used. |
| Check | <p>A pre height measure will be done. Default is a measurement 3 x 3 points over the layout boundary. If one of the points gives an error then writing will be abort.</p> <p>If all points are correct, then during writing the height measurement will be done as be set by HMAPINV and HMAPMODE .</p> <p>The default value for the pre measurement can be change by setting CJ_CHECKHEIGHTPOINTS , CJ_CHECKHEIGHTTOL and CJ_CHECKHEIGHTMAXBADPOINTS in the ini file. By “export parameter_name=value” and no space between parameter_name, equal symbol and value.</p> <p>CJ_CHECKHEIGHTPOINTS, number of points in one direction, the total number of points will be the square. Default value is 3.</p> <p>CJ_CHECKHEIGHTTOL is maximum deviation from average height. Value in µm.</p> <p>CJ_CHECKHEIGHTMAXBADPOINTS, the number of not correct measure point. Default value is 0.</p> |
| Current Map | No pre height measure check will be done, HMAPINV will be set to 1 and overrules setting in the ini file. Depending on the value of HMAPMODE an height measurement will be done or the calculated value will be used. |
| Measure Map | <p>A pre height measure will be done. Default is a measurement 11 x 11 points over the layout boundary. There is no error abort check. HMAPINV will be set to 1 and overrule the ini file setting. Depending of the value of HMAPMODE an height measurement will be done or the calculated value will be used.</p> <p>The default value for the pre measurement can be change by setting CJ_CHECKHEIGHTPOINTS in the ini file. By “export CJ_CHECKHEIGHTPOINTS =value” and no space between CJ_CHECKHEIGHTPOINTS, equal symbol and value.</p> <p>CJ_CHECKHEIGHTPOINTS, is the number of points in one direction, the total number of points will be the square. Default value is 11.</p> |

Table : Overview about options in pre height measure layout

	Option pre height measure layout			
	None	Check	Current Map	Measure Map
Pre Height Measure	No	Yes	No	Yes
Measure Area	-	3x3 layout boundary	-	11x11 layout boundary
Abort by error	-	Yes	-	No
Setting HMAPINV to			1	1
Measure Height during exposure	Yes/No*1	Yes/No*1	No*2	No*2

*1 Depending of HMAPINV and HMAPMODE , *2 Depending of HMAPMODE

7.2.1.2 Parameters HMAPINV and HMAPMODE .

By setting HMAPINV and HMAPMODE the behavior of the height measurement during the exposure can be set. The parameters can be set in a ini –file.

Ini-file

A ini file you can create in the jobs directory by making a textfile with has the same name as the .job file but with an extension .ini and make this file executable with command chmod 755 filename.ini

```
#!/bin/bash
# comment line
Echo "Setting HMAPINV and HMAPMODE"
pg set hmapinv 0
pg set hmapmode 0
```

Figure 44 - Example of a ini file

HMAPINV Switch on height mapping. HMAPINV has 2 options 0 or 1

0 is a measure height will be used (no map or calculation).

1 check HMAPMODE value.

HMAPMODE Use height map or height measurement with boundaries.

HMAPMODE as 4 options 0, 1, 2 or 3.

0 The calculated value from a height map will be used.

1 Height will be measured and if measure height value is in range compare with the calculate value then measure height will be used else the calculate height. Range can be set with HMAPACCURACY (real 0.0..100.0). Units in µm.

2 Height will be measured if measure height in range of X*Map sigma of calculated value then measure height will be used else the calculate height. X factor can be set with HMAPCRITERION (real 0.0..10.0)

3 Height will be measured if measure height between an upper and a lower limit then the measure height will be used else the calculate height value. Lower limit can be set by HMAPLOWLIM (real 100.0-HMAPHIGHLIM) and higher limit can be set by HMAPHIGHLIM (real HMAPLOWLIM..100.0). unit in µm.

Only used during writing.

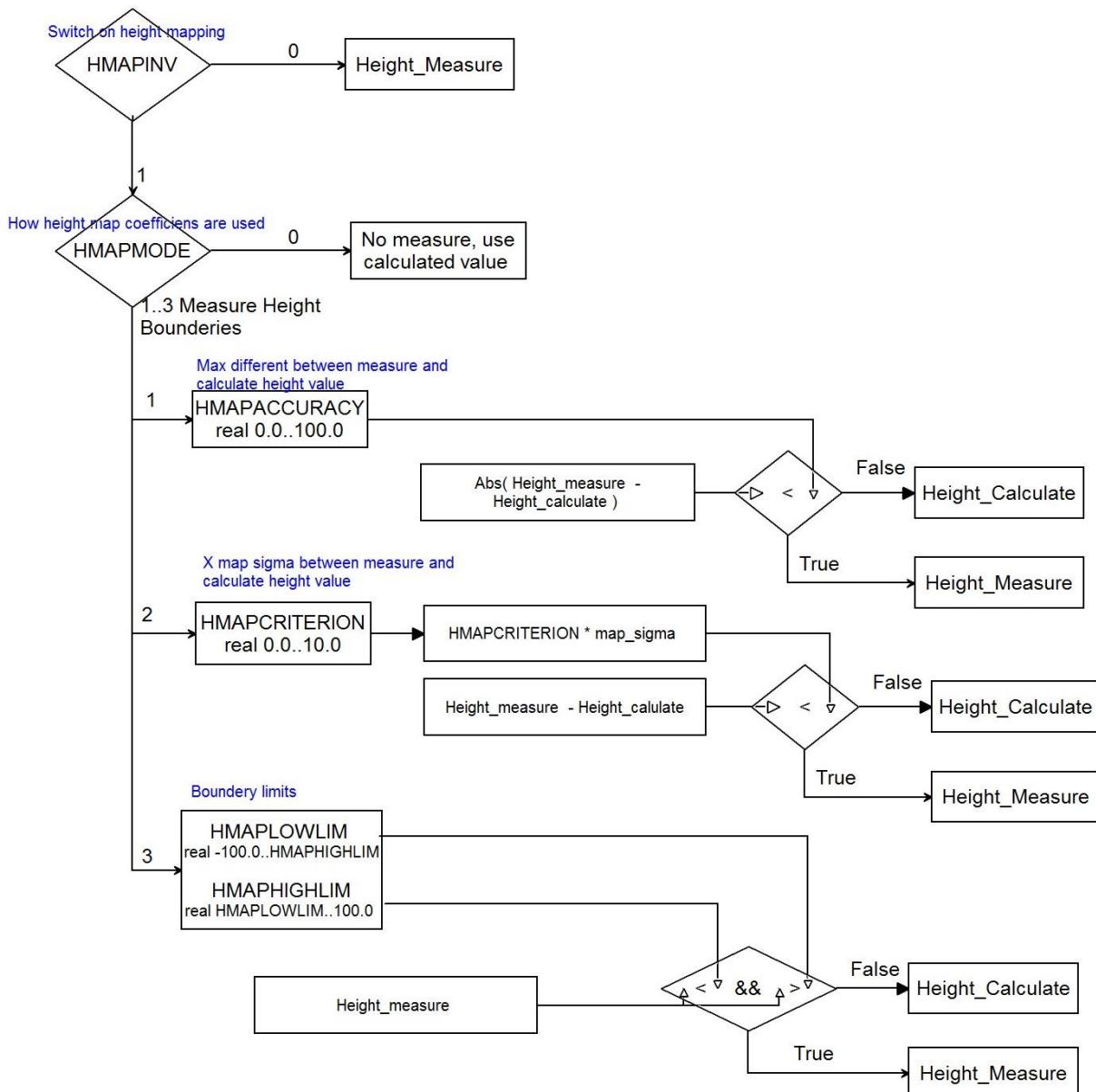


Figure 45 - Diagram HMAPINV and HMAPMODE

7.2.2 Marker deflection

7.3 Logging

8. ADVANCED OPERATION

8.1 Creating dedicated procedures

As an example the measurement of the height is shown, including error checking.

```
#include <stdio.h>
#include "beams.h"
int main (int argc, char ** argv)
{
    long status = HILL_I_NORMAL;
    mpar_values set1, set2;
    unsigned long measure;
    unsigned long binary;
    long partype;
    long par;

    measure = P_NOMEASURE;
    binary = P_NOBINARY;
    par = P_TAB;
    status = hill_mpar_get(&par, &measure, &binary, &partype, &set1, &set2);
    if (status != HILL_I_NORMAL) return (status);

    (void)printf("tab = %f,%f\n", set1.realvar, set2.realvar);

    measure = P_MEASURE;
    binary = P_NOBINARY;
    par = P_HEIGHT;
    status = hill_mpar_get(&par, &measure, &binary, &partype, &set1, &set2);
    if (status != HILL_I_NORMAL) return (status);
    switch (partype)
    {
        case MPAR_LD:
            (void)printf("height = %f\n", set1.realvar);
            break;
        default:
            (void) printf("height measurement invalid\n");
    }
    return (status);
}
```

8.2 Recovery procedures

8.3 User defined command procedures

Various tasks in the BEAMS control software can be overruled or additional task can be added. This is done by defining an environment variable to define user specific procedure. The contents of the variable should be the program or script to be executed. It must contain the full path. Note that it is allowed to use environment variables in the definition, e.g.

```
> export PG_USER_HEIGHT=$PG_SCRIPTS/my_height
```

Currently the following variables are defined:

- PG_USER_CALDRIFT
- PG_USER_CALFULL
- PG_USER_JOYMARKER

This enables the user to implement a user specific marker search procedure.

It is important to note that it is the user's procedure which needs to leave the stage at the position where the marker is expected to exist. Therefore the script / program should always end with the BEAMS command 'pg move position <pos_x,pos_y>'. The BEAMS control software will pick this position up and use it as the found marker position.

- PG_USER_IMAGEMARKER
- PG_USER_HEIGHT

This enables the user to exchange the measure height procedure with a user specific procedure, e.g. in case the height measurement fails.

It is important to note that it is the user's procedure which needs to set the height to the value required, so the script / program should always end with the BEAMS command 'pg set height <height>' command. Note that the height value is in μm .

- PG_USER_MAINFIELD
- PG_USER_MAINFIELD_MOVE
- PG_USER_MAINFIELD_EXPOSE

These 3 can be used to 'break in' to the exposure routine that is used to expose blocks (or main fields). It is important though, to know where exactly the user defined routines start as to prevent unexpected behaviour.

The expose (block) does the following (a bit simplified) :

- Check if pause_expose is enabled. If set to BLOCK, wait until it is cleared.
- If needed, check if the block is in an area that has to be exposed. If it should not expose, the rest is skipped!
- PG_USER_MAINFIELD is executed, if defined
- If needed, a cal full or cal drift is done
- The beam is switched off
- The SEM is switched off
- The stage is moved to the correct x-y (-z) position (with optional height measurement on the move, if needed)

- PG_USER_MAINFIELD_MOVE is executed, if defined
- One of the next corrections is calculated and added:
 - o dwcomp for 'old style' direct write
 - o mapcomp when corrections come from a mapping
 - o heightcomp, if height correction is not disabled
- Check if certain height measurement errors occurred and if this should abort the exposure or not
- Add rotcor correction , if needed
- PG_USER_MAINFIELD_EXPOSE is executed, if defined
- If EXPOSE_VERIFY is selected, the beam blanker control is switched to EILINK mode
- If BLOCKPRINT is enabled, fill the data structure with all relevant information and print the info
- Expose the block
- If EXPOSE_VERIFY was selected, switch the beam blanker control back to the pattern generator.

So, in short, with:

- PG_USER_MAINFIELD, you break in before ANYTHING is done, really.
- PG_USER_MAINFIELD_MOVE, you break in with the stage at the position where it will be exposing the next field.
- PG_USER_MAINFIELD_EXPOSE, you break in after all corrections have been added and set, just before the actual exposure.

The enables the user to add extra functionality just before the exposure of a main field of a pattern starts, i.e. after the move to the stage position where the field is exposed. It is useful to enable customers to display additional information, e.g. field nr, height, but also to make a small correction of the position.

8.4 Error handling

8.4.1 “C”-applications

“C”-applications

In C signal handlers can be defined:

```
void signal_handler(int sign)
{
    ...
}
```

and installed:

```
signal(sign,signal_handler);
```

For defining signal sign, the symbolic names as SIGINT, SIGTERM can be used or their numerical value. In the signal_handler, number signo is available as an integer. In general a SIGINT is generated when pressing Control-C on the terminal running the program, and it is trapped to start the SIGINT signal_handler.

When executing a DCL command file, a command with Control-C results in executing the code defined with the ON CONTROL_Y THEN ... command. According to remarks in the Compaq C RTL manual, Control_Y should raise a SIGQUIT signal, but returns to DCL without printing signal_handler stuff. Even not after typing CONTINUE, EXIT or STOP at the DCL prompt.

An alternative way to raise SIGINT or SIGQUIT is sending it to the process with the C function kill (pid, signo). Blocking of the SIGINT with sigblock(sigmask(SIGINT)) also blocks Control-Y give from the keyboard input (this means that Control-Y sends SIGINT to the C application; SIGQUIT sent from another process is still trapped).

8.4.2 Bash scripts

It is possible to use a more advanced error handling in scripts.

With Beams a script is supplied to handle errors that occur in bash scripts. This script is located at \$BEAMS_SCRIPTS/script_init_template.

A user can modify this script to get a more intelligent handling of possible errors in scripts. An example can be found in \$BEAMS_USR/templates/script_init_errorhandler_example

A simplified version of this file will be explained here in a bit more detail. The user can modify this to their needs and make the routines as “smart” as they would like.

Example for a script_init_template file

```
error_handler_crash_info()
{
    echo " "
    echo "# Error handler: severe error occurred at $(date)"
    date
    echo $MACH_ID
    pg get beam
    mpg tab
    mpqm tab
    iab
    pg get cup
    pg info marker ident
    echo " "
    return
}
```

```

}

error_handler_generic()
{
    echo "# Error handler: error occurred at $(date):"
    error_handler_crash_info
    exit 1
}

error_handler_expose()
{
    echo "# Error handler: error occurred at $(date)"
    error_handler_crash_info
    exit 1
}

error_handler_adjust_ebpg()
{
    echo " "
    echo "# Error handler: trying to recover from"
    echo "           adjust ebpg /sens failure... "
    echo " "
    echo "# MOVE HOME..."
    pg move home
    echo "# ADJUST PULL_IN COMP /SENS"
    pg adj pull_in /sens
    echo "# ADJUST EBPG / SENS..."
    pg adjust ebpg /sens
    echo "# last and crucial try..."
    trap 'error=$? ; error_handler_generic $error' ERR
    echo "# ADJUST EBPG / SENS..."
    pg adjust ebpg /sens
    return
}

error_handler ()
{
    beams_error=`pg get error`

    # The next pg command will clear the error
    # exception: pg get error or pg get command

    case "$beams_error" in
        BEAMS_E_NORMAL )
            # handle Linux errors
            echo "$1 is no beams error at line $2 in $0"
            exit $1
            ;;
        ENG_E_OPEABO )
            echo "Beams command aborted at line $2 in $0"
            jman -p
            exit $1
            ;;
        HILL_E_OPEABO )
            echo "Beams command aborted at line $2 in $0"
            jman -p
            exit $1
            ;;
        * )
            echo "beams error $beams_error in state $beams_state"
            case "$beams_state" in
                generic )
                    error_handler_generic
                    ;;
                adjust_ebpg )
                    error_handler_adjust_ebpg
                    ;;
                expose )

```

```

        error_handler_expose
        ;;
    * )
        echo "No recovery for Beams error $beams_error at line $2 in $0"
        exit $1
        ;;
    esac
    esac
}

interrupt_handler()
{
    echo "CTRL-C pressed; aborting script at line $2"
    exit 1
}

exit_handler()
{
    trap - ERR
}

# Execute "error_handler" after each command termination with
# non-zero status="$?"
trap 'error_handler $? $LINENO' ERR

trap 'interrupt_handler $? $LINENO' INT

# This restores the trap for ERR on exit
trap 'exit_handler' EXIT

# To let shell functions inherit the error trap use:
# This function works in bash-versions greater or equal to 3.xx only
if [ "$BASH_VERSINFO" -ge 3 ]; then
    set -E
fi

```

In the example above the routine “error_handler” is the one that is executed whenever an error occurs. Depending on the error, it will decide what action to take. If it was not a LINUX error or operator abort, it will check if a certain “beams_state” was defined. If so, that routine will be run and the user can decide if the program continues or aborts. (subroutine ends with either ‘return’ or ‘exit’)

In the main script, some modifications need to be made.

First of all, the script containing the error handler should be called.

Then, before commands or routines that have specific error handlers defined, the beams_state variable should be defined.

So, a script could look a bit like this:

```

#!/bin/bash

. $BEAMS_SCRIPTS/script_init_template
beams_state=generic
.
.
.
beams_state=adjust_ebpg
pg adjust ebpg
beams_state=generic

```

```
.  
. .  
beams_state=expose  
pg do "matrix_loop" matrix_field  
beams_state=generic  
  
exit
```

If, in the above script, an error in the PG ADJUST EBPG occurs, the error handler will try to recover that. In this case a MOVE HOME, ADJUST PULL_IN and ADJUST EBPG /SENS will be done. Then it will return to the original script.

During the exposure, It will just print extra information and then abort the script.

This way a number of routines can be added that take care of certain errors or problems during command execution.

8.5 Machine Adjustment Procedures (MATPROC)

The adjustment procedures are to be found in the directory defined with the environment variable BEAMS_MATPROC. The description of the functionality and usage of the procedures are built-in and can be displayed by executing the procedure with the -h option on the command line.

9. ACCEPTANCE TESTS

Acceptance test procedures can be found in the directory defined with the environment variable BEAMS_ACC. The description and usage of the tool are built-in and can be displayed by running the test with the -h option on the command line.

10. ENGINEERING TOOLS

Various engineering tools are available in the BEAMS suite. They can be found in the directory defined with the environment variable BEAMS_ENG. The description of the tool is built-in and can be displayed by executing the tool with the –h option on the command line.

10.1 Adjust field

When the machine is normally calibrating the main field, the system moves the calibration marker to an edge of the field and then finds that same marker, using the deflection.

Normally, during an exposure, 4 markers around the cell are found (on axis) and their positions are used to calculate a mapping.

In some cases the cells are very small (smaller than 1 mainfield) and for some reasons it would be good, if the machine can calibrate the main deflection on 4 small markers in the edges of that field. It should then only use the deflection to find the 4 markers and only move the stage to compensate for a shift, to get into the centre of the 4 markers.

For this, we can use the adjust_field tool!

The idea is that there are lots of small markers (for example 4x4um rectangular markers) with an interval that is a bit smaller than the used main field size.

Then you move the stage close to the centre of one of those fields, then start the command.

It will try to optimise the GAIN, ROTATION and KEYSTONE to get within the required tolerance.

This is all done with relative corrections, because the expose command will else overrule all determined new values!

This also means, that the user is responsible to reset the relative corrections when the scripts is done! Exposing with (wrong) relative corrections can give massive errors!

Since the program can be used to reset the relative corrections, but can ONLY reset them to 0, it's very important to make sure that your system is NOT using relative corrections for the normal exposures. If so, please write them down and restore their values after the script is finished!

Please make sure you read the notes below!!

USAGE:

Adjust_field <deflx,defly> <mark> [options]

deflx,defly deflection to the markers assuming the stage is in the centre of the markers
mark marker ident for the markers to be found

options:

- r reset, will put all relative corrections to 0 and will then exit the program.
- p=x where x is one of the following:
 - 1 prints correction coefficients
 - 2 prints additional information
 - 3 prints additional deflection values
- s=5 required stage position accuracy in nm. Default = 10nm.
- m=5 required main deflection accuracy in nm. Default = 10nm.

EXAMPLES:

`adjust_field -r`

resets all relative correction dacs to 0.

`adjust_field 100,100 neg4 -p=3 -s=2 -m=2.5`

Means the markers are spaced 200umx200um (distance to centre = 100,100um)

Marker type to look for is neg4

Print all info to the screen(correction coefficients, additional info and deflection values)

Keep calibrating until the stage is within 2nm of the calculated centre

Keep calibrating until the markers are found within 2.5nm of deflection error

NOTES:

- Before you use this command, note down the current relative corrections for your system. This can be done with the 'pg info adjust ebg' command.
- The relative corrections are not reset at the end of the program. You either need to do this manually or use this program with the -r option.
- Don't set the required accuracies too small. In that case, it will loop forever! Good systems should go down to 2nm or so. Maybe even a bit lower.
- It is advised to do a 'pg adjust height comp' before starting this command.
- It is also advised to switch off the rotcomp corrections! (pg set tabrc 0, make a note of the old setting first!)

When the program is finished, the stage should be in the centre of the field and the relative corrections should have 'corrected' any gain, rotation or keystone error.

11. USER TOOLS

Many user tools are available. They are located in the directory defined with the environment variable BEAMS_USR. The tools have a built-in help facility: run the tool with the -h option to display the help.

11.1 gds2clay

Import wafer layout from gds2 file with clay

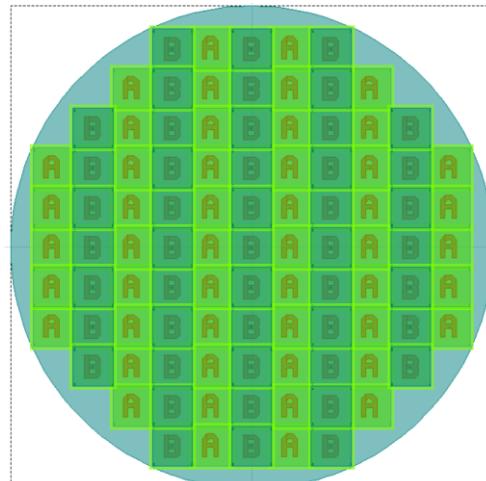
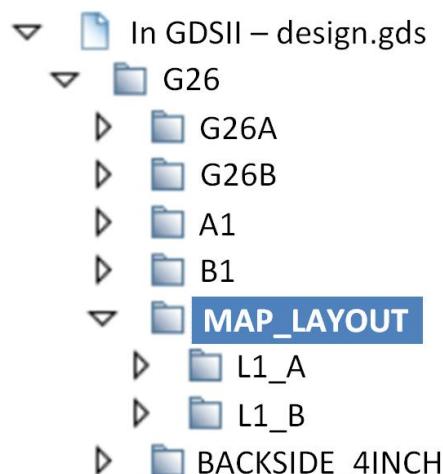
Sometimes on a wafer has different cells all spread over the wafer. In this case it is not easy to manually create cjob. It is preferred to import the wafer layout automatically. Therefore a gds2 file is required which contains a wafer layout, the different cells and a layer which describes the cell area.



Basic knowledge of cjob is required.

GDS2.

One of the main structures must contain the wafer layout with child structures which represents the different cells.



Example: MAP_LAYOUT is wafer layout and L1_A and L1_B contains the cell A and B.

The cell must contain a layer which is only used to indicate the cell area. In this layer a boundary box has to be drawn with the size of the limits of the cell. This will be used to calculate the center of box and also the position of the cell compared to wafer center.



Example: Layer 100, light green layer is used for boundary box. For all cells the same layer.

Create clay file

gds2clay is a program which can extract the coordinates of the cells and transform them to a clay file. A clay file is xml file which can be read by cjob.

14. From the command prompt, enter `ce xxxx`, where xxxx is the project environment o show the list of beams.
15. Enter `cd $PG_PATTERNS` to move to the patterns directory.
16. Enter `pwd` to get the current directory.
17. Copy the gds2 file to current directory.
 - a. Enter `gds2clay` to start the program (add option `-h` for help and more information)
 - b. Program asks for GDS file? Enter gds2 filename.
 - c. Program shows list of names of all main structures and asks for cell name? Enter structure which contains the wafer layout (example MAP_LAYOUT)
 - d. Program shows a list of layer number and marker box layer? Enter layer number with boundary (example 100)
 - e. Program shows a child structure name and ask for to enter the gpf pattern file name. These for all children.
 - f. Program generates a clay file.

Import clay in cjob

Clay file can be imported at the exposure and layout module.

1. Right click on an exposure or layout module and select Import Layout...
2. Navigate to clay file and select the file.
3. CJob will ask to enter beam, dose and optional markers for each cell.

12. DIAGNOSTICS

12.1 Electronics diagnostics

Various diagnostics tools exist for electronics functionality. These tools can be found in the directory define with the environment variable BEAMS_DIAG. All tools do have a built-in help facility launch the tool with the –h option for the help information.

12.2 Stage controller (LSC)

12.2.1 Cstage

12.3 Spectrum analyser

12.3.1 Cspectrum

13. EXCEPTION CONDITIONS

13.1 Vacuum Failure

In the situation of a FEG, the vacuum conditions in the gun are critical for the lifetime of the tip. Therefore a vacuum monitor keeps track of the vacuum levels and sends its status to the Standalone EHT unit.

The following levels are distinguished by the SAEHT:

Status	Level	IGP1+IGP2	Action
		5 μ A	
GOOD	0x0000	<1.5	Normal operation
POOR	0x0001	1.5 < I < 5.0	Run-up of a new emitter, or an emitter which has been off for more than 60 minutes may be started only when vacuum level GOOD is TRUE and may continue while vacuum level POOR is TRUE. If during run-up level POOR becomes TRUE, then run-up is interrupted until level GOOD is again TRUE.
BAD	0x0010	5.0 < I < 10.0	initiates an automatic rundown of the filament
FATAL	0x0011	\geq 10.0	executes a “gscreset”, which will set all values to 0 instantly

The levels of IGP current are defined in the vacuum monitor as advised for safe operation conditions for the FEG filament.

Note that the parameter GUNVAC (not to be confused with VACGUN!) can be set directly, however should be handled with great care: pg set gunvac 3 will reset the SAEHT instantly, however pg set gunvac 0 will set the level to GOOD. Note that these commands are equivalent with pg feg command “gunvac 0x0011” and with pg feg command “gunvac 0x0000”.

The vacuum timeout on the SAEHT can be disabled with the “gunvac” parameter on the standalone: pg feg command “gunvac 0x10000”

13.2 Mains Failure

For FEG systems the following procedure is in place:

All FEG systems should use a UPS to suppress system failure due to short mains supply interrupts, since these likely destroys the filament.

In case mains supply fails, the UPS will send a signal (MAINS FAILURE) to the Standalone EHT unit (SAEHT) to indicate that the mains supply is failing. However, the UPS will continue operating till the batteries are low. Some time (5 minutes?) before the UPS is going to fail as well, the UPS will send a signal to the SAEHT unit (UPS DOWN).

How the SAEHT will be shutdown?

14. MISCELLANEOUS

14.1 Mnemonics

For convenience for many commands abbreviations are available. These can be found in the directory defined by the environment variable BEAMS_MNEMONICS. Just list the directory contents for a full list:

```
>ll $BEAMS_MNEMONICS
```

14.2 Special environment variables

14.3 Debugging information

14.3.1 Debugging

To analyze problems with the machine control software, there is a debug option built into the software. Depending on what “level” is chosen, debug information is printed to the screen or redirected to a log file for further inspection.

In BEAMS versions prior to v09_10a, this is done by defining a certain environment variable. If it was defined, debug lines are printed to screen and could be redirected, or it could be put in a file by defining the environment variable “PG_DEBUGFILE” with a filename.

There are a number of aliases that would activate certain printout:

- debugon
- debugoff
- debugpgon
- debugeccon
- debugmvmon
- debugpmhvon
- debugzdrvon
- debugpgon
- debugupgon

Data would normally be output to the screen (stdout) and could be redirected, or it could be put in a file by defining the environment variable “PG_DEBUGFILE” with a filename. In that case, the output would be redirected to that file.

In BEAMS version v09_10a and higher a systematic approach has been introduced and nearly all information is printed. This results in large log file that can be filtered with a separate program. This way, we can select the information we need to analyze specific problems. To switch on debug mode the command

```
> pg set debug
```

is used.

The new strategy has some advantages and disadvantages:

- The old style was only active in the terminal where the debug was activated. Now, setting the debug value will activate it immediately. This means that as soon as another command is started (no matter which terminal) the debug will become active. A running command will not

print debug information (an exposure, for example . The way it reacts is a bit depended on how a program is started (process).

- There are now a lot more levels to choose from.
 - Log files are a bigger, but we can filter relevant information with the program CDEBUG.
 - It's now much easier to follow the flow through beams and find the exit status of routines.

14.3.2 Codes

We now have 64 options to choose from. Each bit activates a certain part of the logging.

ZSPD	0x00000000000000000000000000000001	DRIVER_BBS	0x00000000100000000000000000000000
ZALL	0x00000000000000000000000000000002	DRIVER_BATH	0x00000000200000000000000000000000
PGTHRД	0x00000000000000000000000000000004	DRIVER_CSYS	0x00000000400000000000000000000000
PGVME	0x00000000000000000000000000000008	DRIVER_DCV	0x00000000800000000000000000000000
MVMSMPL	0x00000000000000000000000000000010	DRIVER_DVM	0x00000001000000000000000000000000
MVMALL	0x00000000000000000000000000000020	DRIVER_ECC	0x00000002000000000000000000000000
DBG_PMHV	0x00000000000000000000000000000040	DRIVER_EHT	0x00000004000000000000000000000000
Spare1	0x00000000000000000000000000000080	DRIVER_EIL	0x00000008000000000000000000000000
Spare2	0x00000000000000000000000000000100	DRIVER_FSP	0x00000010000000000000000000000000
Spare3	0x00000000000000000000000000000200	DRIVER_GPF	0x00000020000000000000000000000000
Spare4	0x00000000000000000000000000000400	DRIVER_HBC	0x00000040000000000000000000000000
DBG_ENTRY	0x00000000000000000000000000000800	DRIVER_HMR	0x00000080000000000000000000000000
DBG_EXIT	0x00000000000000000000000000001000	DRIVER_IO	0x00001000000000000000000000000000
DBGVIEW	0x00000000000000000000000000002000	DRIVER_LID	0x00002000000000000000000000000000
DBGJMAN	0x00000000000000000000000000004000	DRIVER_LND	0x00004000000000000000000000000000
DBGREADER	0x00000000000000000000000000008000	DRIVER_LSC	0x00008000000000000000000000000000
DBGUTILS	0x00000000000000000000000000010000	DRIVER_MDD	0x00010000000000000000000000000000
DBGPARSER	0x00000000000000000000000000020000	DRIVER_MDP	0x00020000000000000000000000000000
DBGSIM	0x00000000000000000000000000040000	DRIVER_MPE	0x00040000000000000000000000000000
DIAGNOSTICS	0x00000000000000000000000000080000	DRIVER_PG	0x00080000000000000000000000000000
COLL	0x00000000000000000000000000100000	DRIVER_PIR	0x00100000000000000000000000000000
HILL	0x00000000000000000000000000200000	DRIVER_PLC	0x00200000000000000000000000000000
ENGINE	0x00000000000000000000000000400000	DRIVER_PMD	0x00400000000000000000000000000000
ENGINE_EHT	0x00000000000000000000000000800000	DRIVER_SCD	0x00800000000000000000000000000000
ENGINE_HC	0x00000000000000000000000001000000	DRIVER_SMC	0x01000000000000000000000000000000
ENGINE_MS	0x00000000000000000000000002000000	DRIVER_TDD	0x02000000000000000000000000000000
ENGINE_MD	0x00000000000000000000000004000000	DRIVER_TMD	0x04000000000000000000000000000000
ENGINE_PI	0x00000000000000000000000008000000	DRIVER_TMP	0x08000000000000000000000000000000
ENGINE_TD	0x00000000000000000000000010000000	DRIVER_VME	0x10000000000000000000000000000000
ENGINE_ZST	0x00000000000000000000000020000000	DRIVER_ZDRV	0x20000000000000000000000000000000
ENGINE_spare1	0x00000000000000000000000040000000	DRIVER_VC	0x40000000000000000000000000000000
DRIVER_ALD	0x00000000000000000000000080000000	DBGTIMER	0x80000000000000000000000000000000

14.3.3 Setting a debug value

There are several ways to set a debug value.

As a decimal value:

pg set debug 6144 (sets the bits for DBG ENTRY and DBG EXIT)

As a hexadecimal value:

pg set debug 0x00000000000001800 (also DBG ENTRY and DBG EXIT)

Or

pg set debug 0x1800 (same result as previous line)

Or with a keyword:

```
pg set debug on  ( sets everything on except for VME messages )
pg set debug off ( sets all bits to 0, switching off debugging )
```

Available keywords:

on	switches all on, except for VME
all	switches all on including VME (warning: this creates huge logfiles!)
off	switches all off, disable debugging
pg	shows pattern generator driver output
pgall	shows pattern generator driver output + thread
ecc	shows driver ecc output
mvm	shows limited marker search output
mvmall	shows extended marker search output including pmhv
pmhv	shows pmhv output
vme	shows vme output
zspd	shows zstage and clamp speed information
zall	shows all zstage and clamp debug output
csys	shows some csys debug output

All of the above also set the timestamp bit (DBGTIMER).

Of course, the data will only be printed if applicable. So, setting ZSPD on a system without piezobox (zstage or clamp) will not print much information!

The “old” aliases still work, but will set the appropriate keyword and bits.

14.3.4 Example of debug output

In the next example, the debug was set to “on” (only the first few lines of the output are shown)

The standard columns are then:

Timestamp,	debug code of function	-> function name : message
0,	0	-> debuglevel changed to : 0xFFFFFFFFFFFFFFF7
0,	8000000000001000	-> global_config_drivers : exit > ... status = 0
79872,	8000000000000800	-> global_htwl : entry < ...
91392,	800000000400000	-> global_htwl : ENGINE:global_htwl() htstat = 100
101632,	800000000400000	-> global_htwl : ENGINE:global_htwl() wlstat = 0

This then enables us to analyze the flow through the routines, the codes of these routines and check the time it takes between the printing of the lines.

14.3.5 Cdebug

The debugging will show more debug information than we used to have.

It holds most of the data we can obtain at the moment.

Now that we have the codes of the routines, we can filter the output and select only the debug lines that are relevant to the problem we are investigating.

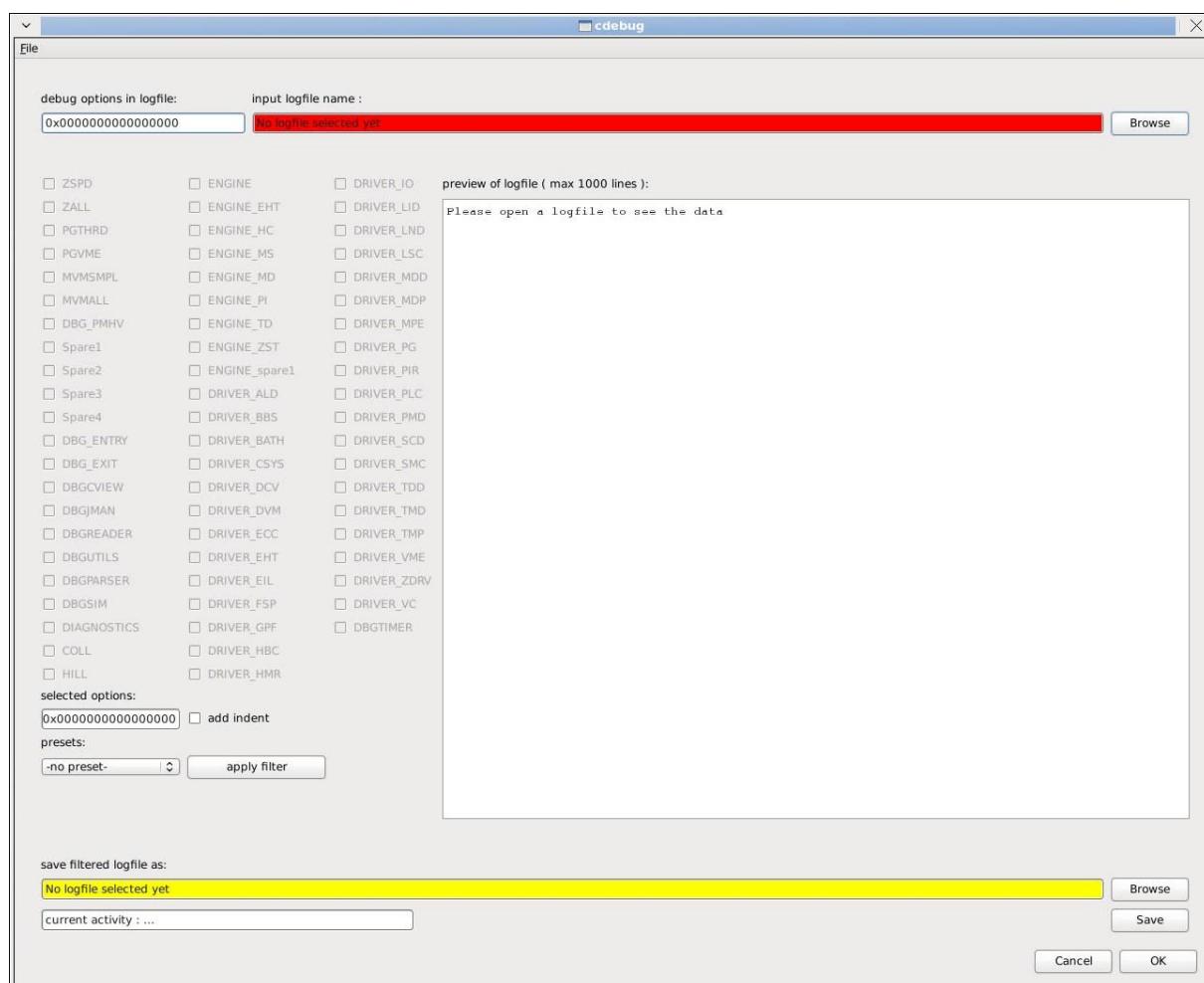
To filter the logfile, there is a tool called “cdebug”.

This is only meant to open a logfile, filter lines that are relevant and save a copy of the result.

It is not meant as a tool to search the log file! There are no search options, whatsoever.

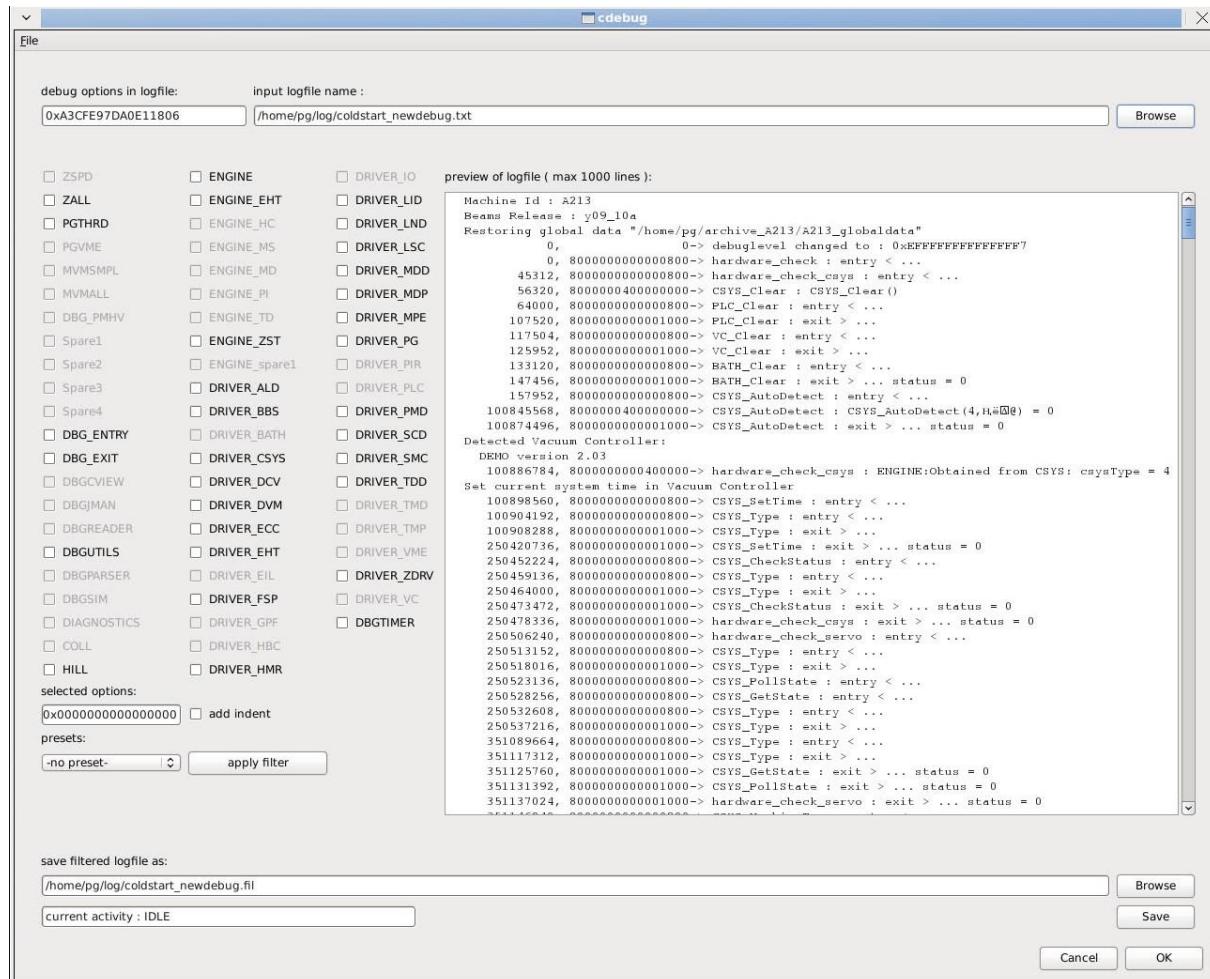
Calling the program will open a gui. If the name of a logfile is given, it will be opened immediately.

If no name is given, a log file must be opened by clicking on the browse button (right top).



A new window will pop up, showing the contents of the log directory of the current user.

Select the file that needs to be filtered.



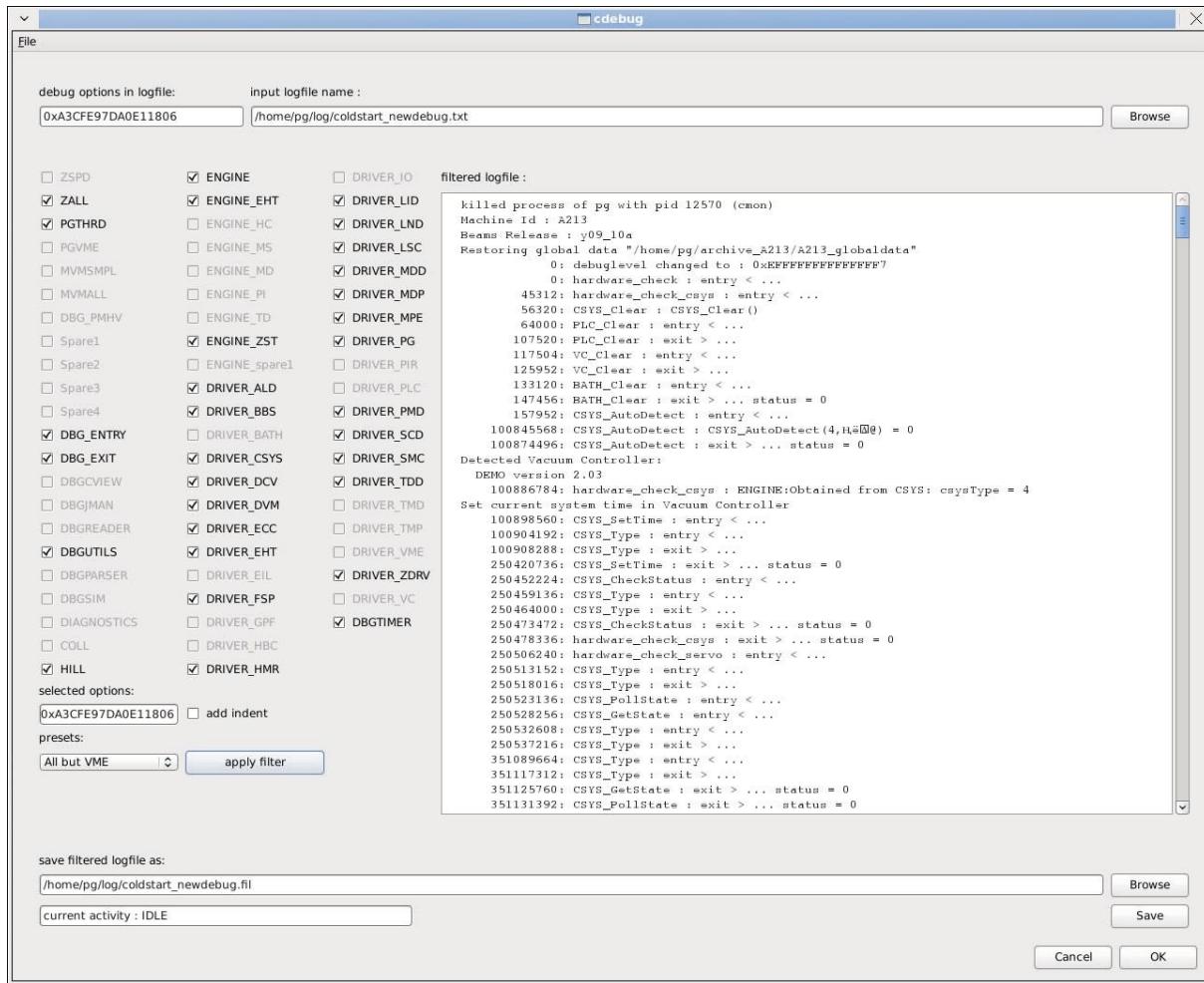
A file was opened and scanned for available debug options.

The options available in the file can now be selected on the left.

"debug options in logfile" = found codes in the opened logfile

"selected options" = code of all the bits of the selected options.

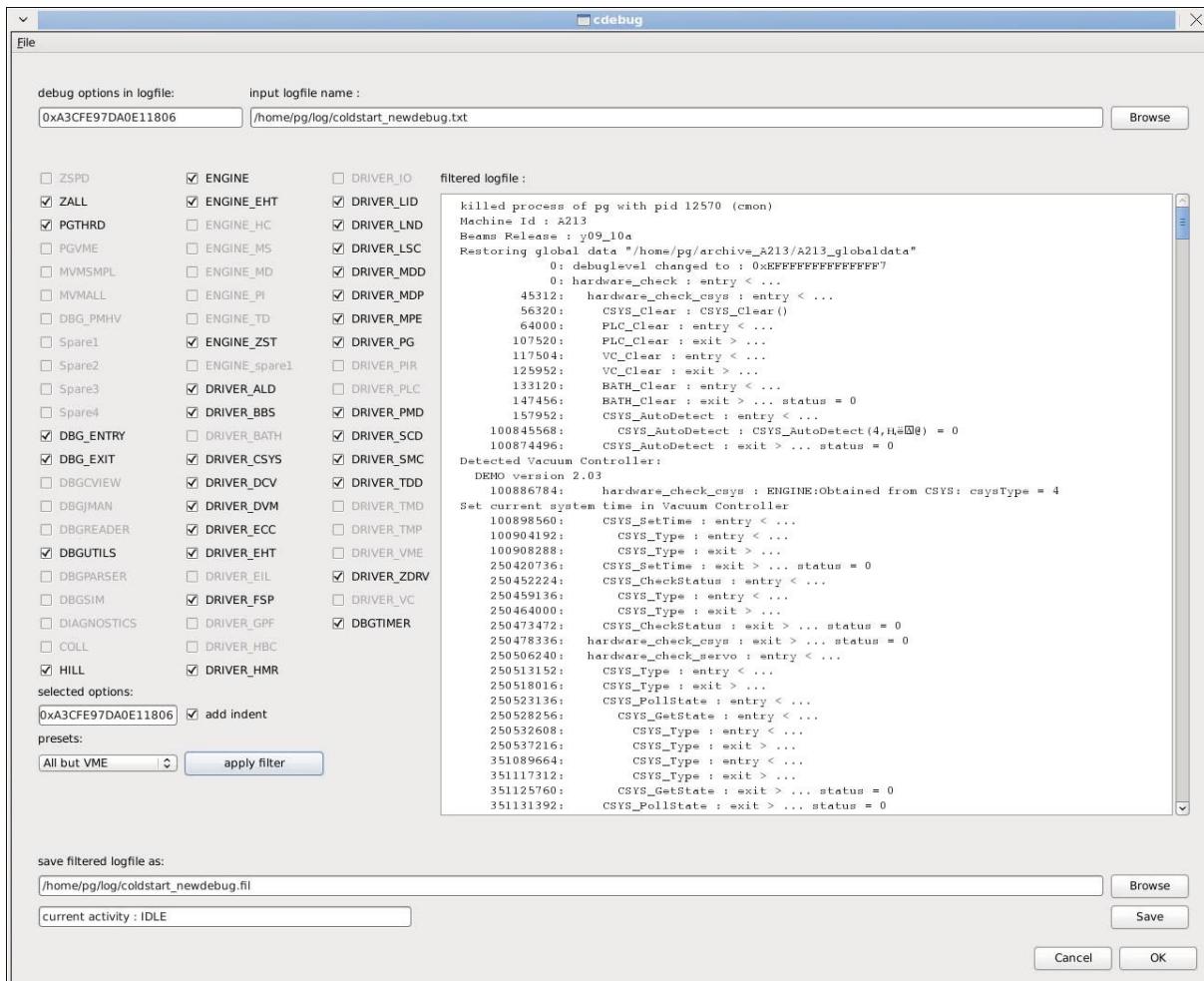
In the next example, the preset “All but VME” is selected, pressing the button “apply filter” results in the following:



Another option is to select the check box “add indent”. Everytime an ENTRY line is found, the indent is increased. When an EXIT line is found, the indent is decreased. This works ok as long as there are no threads running.

With indents its easier to see what happens in a routine and which subroutines are called from there. It makes following the flow easier.

With indents activated you will get the following:



You can now select a filename for the file to save, by clicking the browse button on the right bottom. If you are happy with the suggested filename (original filename with .fil extension) you can press the save button to save the filtered output to a new file.

This can be opened with an editor (for example with vi or kate) to analyze the problem.
If more or less data is needed, you can select/deselect options and save a new file.

14.3.6 Create a logfile with debug information

There are a few ways to create a logfile:

- Select the debug level : pg set debug on
- Start the command with redirection (mvm in this case) :
 - o pg move marker > mvm.txt
 - o pg move marker 2>&1 |tee -l mvm.txt
 - o export PG_DEBUGFILE=mvm.txt
- switch off debugging: pg set debug off

In case you want to create a debug log file from a coldstart:

- select the debuglevel : pg set debug on
- save the global data: \$pg save
- do a coldstart \$pg cold 2>&1 |tee -l coldstart.txt
- switch off debugging: pg set debug off
- save the globaldata \$pg save

15. INSTALLATION MASTER CONTROL COMPUTER

15.1 Hardware requirements

For controlling the EBP5000 Plus machine any computer system configurable for Red Hat Enterprise Linux 4, 64 bits (RHEL) with at least two Ethernet devices can be used. Further the computer must have at least one internal free PCI slot available for a CAEN v2818 communication device.

- Base Computer system configurable for RHEL4 64 bits
- 1 GB RAM
- 2 NIC's: each minimal 100mbps
- Graphics-card: resolution minimal 1280*1024
- Large Hard disk: > 100 GB
- 1 serial port
- 18" Monitor: resolution minimal 1280*1024
- 3-button mouse
- Keyboard

15.2 Current configuration

The configuration as shipped with the latest systems is specified below.

	CAUTION	This configuration is subject to change frequently. Please be advised to check with Raith for recent updates/changes
---	----------------	--

EBPG5000 Plus ES

Base System	1	HP Z800
Processor	1	Intel Xeon E5504 2.00GHz 4MB 1066 FSB Quad Core
Graphics	1	NVIDIA Quadro FX580 512MB PCIe Graphics
Optical	1	HP 16x SATA DVD+/- RW Drive
Keyboard	1	HP USB Standard Keyboard (US)
Mouse	1	HP USB Laser Scroll Mouse
Display	2	HP L2445w 24-inch Widescreen LCD Monitor
Operating System	1	Redhat Enterprise Linux 4 WS, 64 bit
Storage	1	SATA-disk: HP 160 GB SATA 3.0GB/s NCQ 7200 rpm Hard Drive
Memory	1	HP 1 GB (1x1GB) DDR3-1333 MHz ECC DIMM
Frame Grabber	1	ADLINK: PCIe-RTV24; 4-ch PCI Express x1 Frame Grabber

EBPG5000 Plus standard

Base System	1	HP Z800
-------------	---	---------

Processor	1	Intel Xeon E5504 2.00GHz 4MB 1066 FSB Quad Core
Graphics	1	NVIDIA Quadro FX580 512MB PCIe Graphics
Optical	1	HP 16x SATA DVD+/- RW Drive
Keyboard	1	HP USB Standard Keyboard (US)
Mouse	1	HP USB Laser Scroll Mouse
Display	2	HP L2445w 24-inch Widescreen LCD Monitor
Operating System	1	Redhat Enterprise Linux 4 WS, 64 bit
Storage	2	HP 450GB SAS 15K Hard Drive
Memory	1	HP 1 GB (1x1GB) DDR3-1333 MHz ECC DIMM
Frame Grabber	1	ADLINK: PCIe-RTV24; 4-ch PCI Express x1 Frame Grabber
CAD system:		
Base System	1	HP Z800
Processor	1	Intel Xeon E5504 2.00GHz 4MB 1066 FSB Quad Core
Graphics	1	NVIDIA Quadro FX580 512MB PCIe Graphics
Optical	1	HP 16x SATA DVD+/- RW Drive
Keyboard	1	HP USB Standard Keyboard (US)
Mouse	1	HP USB Laser Scroll Mouse
Display	2	HP L2445w 24-inch Widescreen LCD Monitor
Operating System	1	Redhat Enterprise Linux 4 WS, 64 bit
Storage (ES)	1	SATA-disk: HP 160 GB SATA 3.0GB/s NCQ 7200 rpm Hard Drive
Storage (standard)	2	HP 450GB SAS 15K Hard Drive
Memory	4	HP 1 GB (1x2GB) DDR3-1333 MHz ECC DIMM

15.3 Software requirements

- Red Hat Enterprise Linux 4 64 bit (RHEL4) distribution CD's: 1 - 5,
- Linux BEAMS5000P-distribution: beams_y09_03e.tar.gz,
- TFTP deamon (standard delivered with RHEL4)
- If necessary additional hardware drivers for graphics-card or NIC's.

15.4 RHEL Installation

The linux operating system can easily be installed by booting the computer system from "CD 1" of the RHEL distribution. With the Graphical installation program, almost all default answers can be used to make the installation successful. Default settings can be changed depending on the user's wishes and environments where the system will be installed, like time-zone's and network-configurations (e.g. DHCP or static IP-addresses).

However for controlling the machine correctly, only a few settings are required. These requirements are:

1. Network configuration

Be sure one Ethernet device connected to the EPG5000 Plus has the next IP-settings, e.g. eth1:

```
device IP-address Subnet-mask
eth1: 192.168.3.200 255.255.255.0
```

2. Firewall

Be sure the Ethernet device connected to the EPG5000 Plus is a trusted device or the firewall is disabled.

3. Software installation

It is recommended to install all software from the RHEL distribution CD's on the system.

4. Graphical Interface

For the best view of the working space, a dual monitor setup is advised: the screen resolution must be set to 1920*1200 (wide screen 24").

5. User Account

During the installation of RHEL the administrator will be asked for creating a user account. Because the required user accounts: "beams" and "pg" are created during the setup of the BEAMS-software, the creation of a user-account can be skipped during the Linux installation.

NOTE for improperly working hardware:

In case a piece of hardware is not working properly or not seen by the system at all, have a look on the Internet for a driver.

HINT:

Use "lspci [-n]" to get the hardware ID's where to search for on the internet.

For detailed information about installing RHEL4, see:

"Installation Guide of Red Hat Enterprise Linux".

15.5 BEAMS500P software configuration

The configuration of the mcc for BEAMS5000P can be done by a setup-script. This setup-script is in the BEAMS5000P distribution.

- Login as user "root"
- Create the home-directory of user "beams" wherein the BEAMS5000P-software will be installed:

```
[root@b023 ~] mkdir /home/beams
```

```
[root@b023 ~] cd /home/beams
```

- Get the BEAMS5000P-distribution software from the ftp-site:

```
[root@b023 beams] lftp ftp.Raith.nl -u downloads,beamwriter -e \
```

```
"cd beams5000p/linux; get beams_y09_03e.tar.gz; exit"
```

- Extract the just downloaded BEAMS5000P-distribution software:

```
[root@b023 beams] tar -xzf beams_y09_03e.tar.gz
```

- Finish the installation:

```
[root@b023 beams] ./y09_03e/setup/setup_mcc
```

This script checks some system-requirements whether they are installed, if not, they will be installed:

- "Raith" group
- "beams" user --> password is default "beamwriter"
- "beams" login-requirements
- "pg" user --> password is default "beamwriter"
- "pg" user directory and file structure
- "pg" machine ID : prompt for ID if file "pg_local" is missing
- "pg" login-requirements
- set "beams" owner-ship and protection
- set "pg" owner-ship and protection
- TFTP-server and settings

NOTE: the script "setup_mcc" can be run at any time. As mentioned above the script checks whether some system-requirements are set.

- Ready: Login as "pg"

15.6 Additional software configuration

A set of tools and configuration scripts which have nothing to do with the BEAMS-software, and are thus not required, are delivered with this BEAMS-package: "linux_tools". The distribution file is:

```
/home/beams/y09_03e/setup/tools_linux.tar.gz
```

And the installation-description is:

```
/home/beams/y09_03e/setup/tools_linux_install.txt
```

To see whether these tools are installed or see what these tools are, enter the command "tools", e.g.:

```
> tools

Raith Linux tools V02.02 (05-NOV-2008)

  astyle      backup      bye       cs
  dump_nfs   evaluate    fbfiles   findf
  gnome       listpath    lo        preference
  purge       rar         search    selinux
  systools   tar_gz_ftp  tar_gz_nfs tools
  tree        val2var2lists vms2linux Xcommands
```

For more detailed help, type:
 tools <command>

Further the following public packages are installed:

1. ImageJ, which is a public domain, java-based image processing program developed at the National Institute of Health.

15.7 Template files

In the BEAMS distribution there are various template files for copying to the local account from where the tool is controlled:

In \$BEAMS_SETUP can be found:

pg_bashrc	template to insert in .bashrc script of operating account
pg_local	setup of environment in account for operating the tool
beams	installation script for installing (new) BEAMS release

In \$BEAMS_SCRIPTS can be found:

script_template	template script to create a job
script_init_template	template script for error handling

In \$BEAMS_INCLUDE can be found:

Makefile_template	template Makefile for building hill_shelly application
demo.c	example source code for hill_shelly application

In \$BEAMS_GPF can be found:

Makefile_template	template Makefile for building a tool to create patterns
gpfpar_template	script to setup environment variables required to generate GPF patterns
gpf_utils.h	example source files to create patterns
gpf_utils.c	general functions used to create GPF patterns
ringsgpf.c	generates concentric ring patterns
zonegpf.c	generate zone plate patterns

16. TROUBLESHOOTING

17. REFERENCES & REVISION HISTORY

17.1 References

- [1] BEAMS command reference manual (COLL_SHELLY) (part of BEAMS software distribution)
- [2] BEAMS library reference manual (HILL SHELLY) (part of BEAMS software distribution)
- [3] Generic pattern Format, Software Product Description, B900048 (part of BEAMS software distribution)
- [4] Manual for DiagGPG V3.0
- [5] Evacuate and Load Control – FEG-version, B900037
- [6] EBPG 5000plus service engineers DAC diagnostics manual, document 893446 (part of BEAMS distribution)
- [7] Matproc1
- [8] Matproc 2
- [9] Matproc3
- [10] CJOB html documents

17.2 Revision History

This records summary details of changes to this document.

Date Issue Name Details

3 Nov 2006 1 H Romijn/T Jorden/Marcel.. First issue based on B900100.

RAITH

NANO FABRICATION



Managing Director
Erwin Mueller

Chamber of Commerce
Eindhoven 33286867

VAT Number
NL001751773B01

ABN AMRO N.V.
IBAN : NL28 ABNA 0247 4414 30
BIC : ABNANL2A