NCD-SWEET

Brief user guideline

This is a guide on the most common macros used while performing experiments at NCD-SWEET beamline. The described macros are accessible via:

- Command Line Interface (CLI) by using Spock
- Graphical User Interface (GUI) by using the sequencer of bl11ExpGui.

Before starting, some **warnings and advices**:

- Think before. Safety first.
- Before moving motors, ensure there will be NO collisions.
- All macros should be internally documented. To access the documentation, type in *Spock* the macro name followed by '?'. Example: 'wm?'
- The *floor coordinator* is your first point of contact during non-working hours when you have technical issues with the beamline and/or issues with the laboratories.

And in case of doubt: **Ask your local contact!**

Good luck with your experiments ☺

Internal Emergency number

4499

Floor coordinator numbers:

(+34 93.592) **4401**

(+34 93.592) **5401**

(+34) **608.018.721**

PRACTICAL INFORMATION

Beamline website and general information

You will find detailed and updated information at the NCD-SWEET beamline:

https://www.albasynchrotron.es/en/beamlines/bl11-ncd

User and password

- Your user name (proposal account) is composed by “u” + your proposal number. *e.g.: “u2020093939”*
- Your password has to be changed at the beginning of the experiment by visiting: https://albapassword.cells.es/

* Remember your user and password, since it will be needed for login in the beamline computer and remotely access to your data.

External data access:

All beamline users can download their experiments data connecting via SFTP:

➢ SFTP configuration
- Host: **userdata.cells.es**
- Port: **443**
- Username: <**proposal account**>
- Password: <**proposal password**>
- Directory: **/DATA**

* Common programs for SFTP connection are:

- Windows: WinSCP
- Linux clients: KDE/Dolphin, Ubuntu/Nautilus, Filezilla, command-line...
- Mac OS X clients: Cyberduck

Visit the following link for more information:

https://intranet.cells.es/Intranet/Help/BeamlinesHelp/sftp/index_html

Note: Currently ALBA guarantees data storage up to 6 months after your experiment terminates. We are currently working to extend this data storage time in the future.

BEAMLINE COMPUTER DESKTOP

To access the CLI, the BL11ExpGui and the most common programs, different icons can be found on the computer desktop:

Terminal & Spock

Terminal.desktop

- Spock is the CLI for sardana based on ipython
- Open a terminal window and type:

  spock ⏎

Device Restarter: beamline status

- Graphical interface for device status and restart

Pilatus (SAXS) and Rayonix (WAXS) detectors image visualizers

LaVue_Pilatus.desktop LaVue_Rayonix.desktop

Offline image visualizer

LaVue.desktop

Sweet program for aligning

Sweet.desktop

- Interacts with: sascan, sdscan, CScan2 macros
- Options: *Go maxval* | *Go minval* | *Go pos* | *Go edge*

Linkam frames calculator

N_of_frames_Calculator.ods

- To calculate the number of frames, acquisition time, etc. when running linkam temperature profiles

Beamline experimental hutch camera: ipcam39

- Via Firefox for setting up the camera. Once the camera is set, CLOSE Firefox or it will freeze the computer.

  ipcam39_Control.desktop

- Via a visualizer

  ipcam39.desktop

There are other cameras that can be open using the terminal.
For that you can type in a terminal tab: "**ipcam07**", "**ipcam22**" , "**ipcam39**"

BEAMLINE USER MACROS

A short guide

DATA MANAGEMENT

newSample	newSample <sample_name>

Function:

- Define a new folder in your main experiment directory.
- Define a new prefix for the detector filenames that can be overwriten by the macro newPrefix
- Subfolders can be defined by extending the path[2]

If the folder already contains detector data, an error message will be displayed and the new folder will NOT be created. [Data management rules]

Example:

- newSample sample0	*Define the new folder*
- newSample gisaxs/sample1	*Define the subfolder*

newPrefix	newPrefix <sample_prefix>

Function:

- It overwrites the detector filenames prefix defined by the macro newSample. The folder is kept unchanged.

Example:

- *newPrefix sample2*

BEAMLINE

shopen shopen

Function:
- Open the safety shutter (between the optical and experimental hutch)

Example:
- shopen

shclose shclose

Function:
- Close the safety shutter (between the optical and experimental hutch)

Example:
- shclose

fsopen fsopen

Function:
- Open fast shutter

fsclose fsclose

Function:
- Close fast shutter

pwopen pwopen

Function:
- Open the flight tube protection window

Example:
- pwopen

pwclose pwclose

Function:
- Close the flight tube protection window

Example:
- pwclose

DATA ACQUISITION

snap snap <acq_time> <nº_frames> <latency>

Function:
- Macro to take images and save them in the defined directory by "newSample"
- Default values: 1s, 1 image, minimum possible latency

Example:

snap	*Take 1 image of 1s exposure with minimum latency*
snap 0.1 3 0.5	*Take 3 images of 0.1 s with 0.5 s latency between the images*

snapascan snapascan <motor> <start_position> <end_position> <nº_intervals><acq_time>

Function:
- Scan in absolute positions while saving images in the defined path by "newSample"
- The units are mm for length or degree for angle

Example:
- *snapascan sx -1 1 10 0.1*

snapdscan snapdscan <motor> <start_position> <end_position> <nº_intervals><acq_time>

Function:
- Scan in relative positions to the current position while saving images in the defined path by "newSample"
- The units are mm for length or degree for angle

Example:
- *snapdscan sx -1 1 10 0.1*

snapmesh snapmesh <motor1> < start_position1> < end_position1> <nº_intervals1> <motor2> < start_position2> < end_position2> <nº_intervals2> <acq_time> <opt: bidirectional>

Function:
- mesh scan of two motors in absolute positions while saving images in the defined path by "newSample"

Example:
- *snapmesh sx -70 70 100 sz 45 75 200 0.1*

snapdmesh snapdmesh <motor1> < start_rel1> < end_rel1> <nº_intervals1> <motor2> < start_rel2> < end_rel2> <nº_intervals2> <acq_time> <opt: bidirectional>

Function:
- mesh scan of two motors in relative positions while saving images

Example:
- *snapdmesh sx -1 1 10 sz -2 2 20 0.1*

snapstepscan snapstep <motor1> < [positions]> <acq_time>

Function:
- Macro that runs a scan at different motor positions defined in a list
- If adapt_prefix is True (default: False) the motor position will be included in the filename

Example:
- *snapstep spitch [0 0.05 0.1 0.15 0.2] 0.5*

snapstep2scan snapstep2 <motor1> < [positions]> <motor2> < [positions2] <acq_time>

Function:
- Macro that runs a scan at different motor positions defined in a list
- Motor2 iters for each Motor1 position

Example:
- *snapstep2 sx [-10 0 10] spitch [0 0.05 0.1 0.15 0.2] 0.5*

ALIGNMENT

kt	kt <acq_time >

Function:
- Acquire image without saving it, just for visualization
- Default acquisition time: 1 s

Example:
- *kt 0.1*

wm	wm <motor>

Function:
- Print the current defined motor position

Example:
- *wm sx*

wslits	wslits

Function:
- Print the current slits position in a formatted table

wsample	wsample

Function:
- Print the current sample environment motor positions in a formatted table

mv	mv <motor> <position>

Function:
- Move the defined motor to the absolute target position

Example:
- *mv sx 5.4*

mvr	mvr <motor> <step>

Function:
- Increment the defined motor by the given value

Example:

- *mvr spitch 0.3*
- *mvr sx -0.5*

sascan sascan <motor> <start_rel> <end_rel><nº_intervals><acq_time>

Function:
- Scan with absolute motor positions (ascan) interacting with “sweet” program.
- Images will NOT be saved

Example:
- *sascan sx 5 7 20 0.1*

sdscan sdscan <motor><start_rel><end_rel><nº_intervals><acq_time>

Function:
- Scan with relative motor positions (dscan) interacting with “sweet” program.
- Images will NOT be saved

Example:
- *sdscan sx -1 1 20 0.1*

go go <option>

Function:
- Interact with SWEET program for scan. There are different options:
  “maxval”: data max, “minval”: data min, “pos”: fit position, “edge”: derivative position

Example:
- *go pos*

setm setm <motor >

Function:
- Redefine current motor position to the desired value

Example:
- *setm spitch 0*

resetm resetm <motor >

Function:
- Set the motor offset to 0

Example:
- *resetm spitch*

align_sample align_sample <full_beam_diode_current>

Function:
- Macro that aligns a GISAXS / GIWAXS sample, i.e. sz and spitch. It finds the sz threshold by looking at the diode counts while moving the sz motor and then it does a “sdscan spitch -0.3 0.3 30 0.1”.
- “SWEET” program must be running
- Input the current on the photodiode with full beam OPTIONAL after the first use
- WARNING: not recommended for substrates that can partially transmit the beam. Do not leave this macro working without surveillance. For the first sample, do a manual alignment.

Example:
- *align_sample 5E-5*

diode_value diode_value

Function:
- Macro to get the full diode counts, keeping in memory for its use with align_sample macro.
- It will move sz -1 mm and put the spicth at 0, so, use it carefully

Example:
- *diode_value*

align_sweet_option align_sweet_option <option>

Function:
- Macro to configure the autoalignment to follow the maxval or fitting pos of sweet program
- Options: “pos”, “maxval”

Example:
- *align_sweet_option maxval*

LINKAM

linkam_on linkam_on

Function:
- Restarts the device server and enables the linkam experimental channels

linkam_off linkam_off

Function:
- Disables the linkam experimental channels

linkam_ramp linkam_ramp <temperature> <rate>

Function:
- Start linkam ramp while showing the progress. It waits for the target temperature to continue
- Temperature: ºC
- Rate: ºC/min

Example:
- *linkam_ramp 300 10*

linkam_start_ramp linkam_start_ramp <temperature> <rate>

Function:
- Start linkam ramp in the background and continue
- Temperature: ºC
- Rate: ºC/min

Example:
- *linkam_start_ramp 300 1000*

linkam_hold_temperature linkam_hold_temperature

Function:
- Holds the current temperature

linkam_stop linkam_stop

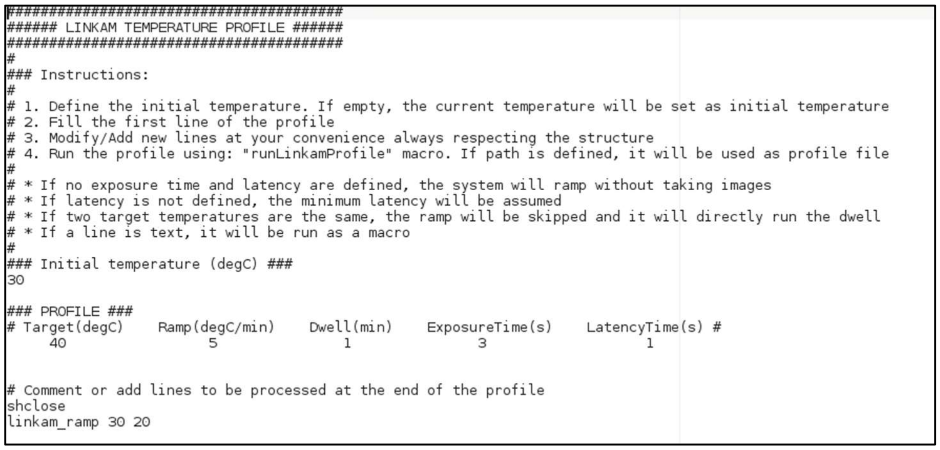
Function:
- Stop linkam temperature control

newLinkamProfile newLinkamProfile

Function:
- Create a new Linkam Profile file in the default experimental path

*Explanation:

A new text file (*bl11_linkam_profile.txt*) will be created in the experiment path base. The file is used to create a linkam profile for a sample.



```
#########################################
###### LINKAM TEMPERATURE PROFILE ######
#########################################
#
### Instructions:
#
# 1. Define the initial temperature. If empty, the current temperature will be set as initial temperature
# 2. Fill the first line of the profile
# 3. Modify/Add new lines at your convenience always respecting the structure
# 4. Run the profile using: "runLinkamProfile" macro. If path is defined, it will be used as profile file
#
# * If no exposure time and latency are defined, the system will ramp without taking images
# * If latency is not defined, the minimum latency will be assumed
# * If two target temperatures are the same, the ramp will be skipped and it will directly run the dwell
# * If a line is text, it will be run as a macro
#
### Initial temperature (degC) ###
30

### PROFILE ###
# Target(degC)    Ramp(degC/min)    Dwell(min)    ExposureTime(s)    LatencyTime(s) #
      40                5               1               3                  1


# Comment or add lines to be processed at the end of the profile
shclose
linkam_ramp 30 20
```

The file can be adapted to run the desired Linkam profile following the instructions.

runLinkamProfile runLinkamProfile <opt: file>

Function:
- Run the Linkam profile defined in the file
- If file is not defined, the default (Experimental_path/bl11_linkam_profile.txt) will be used
- If file path or relative path to the experimental route is defined it will be used
- A .txt file containing the linkam profile will be saved in the same folder than the images

Example:
- *runLinkamProfile*
- *runLinkamProfile /beamlines/bl11/…/myprofile.txt (full route)*
- *runLinkamProfile sample1/bl11_linkam_profile_s1.txt (relative route to experiment path)*

USER MACROS

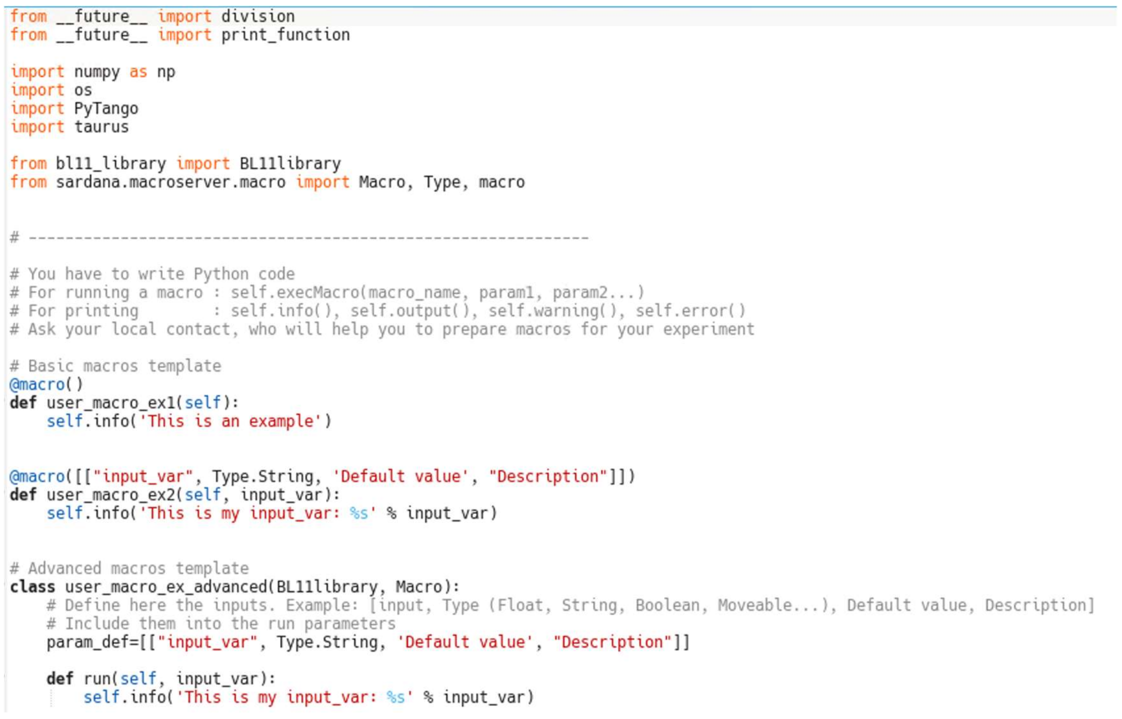
newUsermacro newUserMacro

Function:

- Create a new User Macro file in the default experimental path

*Explanation:

A new text file (*bl11_user_macros.txt*) will be created in the experiment path base. The file is used to create user defined macros that will be later loaded into the macro server



```
from __future__ import division
from __future__ import print_function

import numpy as np
import os
import PyTango
import taurus

from bl11_library import BL11library
from sardana.macroserver.macro import Macro, Type, macro


# -------------------------------------------------------------

# You have to write Python code
# For running a macro : self.execMacro(macro_name, param1, param2...)
# For printing        : self.info(), self.output(), self.warning(), self.error()
# Ask your local contact, who will help you to prepare macros for your experiment

# Basic macros template
@macro()
def user_macro_ex1(self):
    self.info('This is an example')


@macro([["input_var", Type.String, 'Default value', "Description"]])
def user_macro_ex2(self, input_var):
    self.info('This is my input_var: %s' % input_var)


# Advanced macros template
class user_macro_ex_advanced(BL11library, Macro):
    # Define here the inputs. Example: [input, Type (Float, String, Boolean, Moveable...), Default value, Description]
    # Include them into the run parameters
    param_def=[["input_var", Type.String, 'Default value', "Description"]]

    def run(self, input_var):
        self.info('This is my input_var: %s' % input_var)
```

relusermacro relusermacro

Function:

- Reloads the user macro file if any change is detected for being used in Spock. Then, the user macros will be accessible via Spock.

DETECTOR MACROS

✚	**roi_set**	roi_set <detector> <X1 > <Y1l> <X2l> <Y2>

Function:

- Set the detector ROI for the corresponding experimental channel
- If “full”, the complete detector is considered
- If no input, the current ROI is printed
- Pixel (0,0) in the bottom left part of the image

Example:

- *set_roi pilatusi 250 150 850 900*
- *pilatus_set_roi full*