

Manual SXM Remote Control for Python

for Python 3.5.2
with Anaconda 4.2
(to install matplotlib)
SXM.exe 26.4

Connection between Python and SMX Software via DEE.

Example of Python command

SendWait (s)

SXM remote → s is extended to Pascal like script and send to DDE Server in SXM.exe and waits for Message from DDE Server.

SXM.exe → save script as “remote.scr” and start, finally send message to DDE client

SXM remote → wait is finished/canceled.

Communication with the 'SXMRemote.py'-Library.

Use this in every SXM Remote Control Python file to get all the functions.

```
import SXMRemote
```

```
MySXM = SXMRemote.DDEClient("SXM", "Remote");
```

Call the different functions like this:

```
MySXM.function(Parameters);
```

Depending on what functions were needed there are different Python files.

1 General: general.py

- 1.1 Trigger callback when new data file has been written
- 1.2 Read name and path of data file written
- 1.3 Start/stop scanning
- 1.4 Check if scan is in progress
- 1.5 Trigger Callback when scan starts/stops

2 Scanner: scanner.py

- 2.1 Set/read scan area rotation angle
- 2.2 Set/read scan area width
- 2.3 Set/read number of scan points per scan line
- 2.4 Set/read number of scan lines
- 2.5 Set/Read scan area height
- 2.6 Set/Read scan area offset
- 2.7 Set/read raster time
- 2.8 Trigger callback when scan reaches end of scan line / Trigger callback when scan finishes retrace of scan line
- 2.9 Trigger callback when scan area has been scanned completely in “up”/”down” direction
- 2.10 Enable/disable drift compensation
- 2.11 Set/read drift compensation vector
- 2.12 Enable/disable plane subtraction scan mode
- 2.13 Set plane subtraction scan slope (X/Y)
- 2.14 Read and remember current tip position
- 2.15 Shift tip to new position
- 2.16. Return to remembered position

3 Feedback Loop: feedback.py

- 3.1 Enable/disable feedback loop
- 3.2 Set feedback mode
- 3.3 Set ratio between AFM and STM feedback mode
- 3.4 Set/read preamp tunneling current range
- 3.5 Set/read primary part of loop gain
- 3.6 Apply a relative Z-offset
- 3.7 Set slew rate for Z-offset changes
- 3.8 Read the absolute Z-position

4 Channel: Channel.py

5 Spectroscopy: Spectroscopy.py

- 5.1 Set mode of Spectroscopy
- 5.2 Set start/end value of (V/Z)-spectroscopy ramp
- 5.3 Set number of points of (V/Z)-spectroscopy ramp
- 5.4 Set sample time of (V/Z)-spectroscopy ramp
- 5.5 Trigger Callback when new data is available
- 5.6. Read acquired spectroscopy data (ReadSpectroscopyData.py)

6 Continuous Signal: collectData_sety.py

- 6.1 Read data
- 6.2 Enable/disable

7 Gap Voltage Control: bias.py

- 7.1 Set/read gap voltage
- 7.2 Set/read preamp gap voltage range
- 7.3 Enable/disable modulation voltage

8 Tip conditioning: TipCond.py

- 8.1 Set conditioning mode
- 8.2 Apply voltage pulse (Bias)
- 8.3 Configure and run Z-ramp

9 Coarse Control: coarse.py

- 9.1 Move tip
- 9.2 Initiate retract operation
- 9.3 Initiate auto approach operation
- 9.4 Trigger callback when auto approach operation finishes

1 General (general.py)

1.1 Callback when new file has been written

The function MyNewFileIsWritten(FileName) reads from the SXM *.ini. The first argument is the section and the second is the item in the ini file.

```
MySXM.GetIniEntry('Save','Path');
```

When the scan was successful it leaves a callback with the filename.

```
MySXM.SaveIsDone=MyNewFileIsWritten
```

1.2 Callback for Start and stop scan

Use this callback functions for check if the scan is on or off.

```
MySXM.ScanOnCallBack = MyScanIsOn  
MySXM.ScanOffCallBack = MyScanIsOff
```

1.3 Start and stop scan

```
XX    = 1    starts the scan.  
XX    = 0    stops the scan.
```

Better use a sleep between both functions.

```
My.SXM.execute(“ScanPara('Scan', XX);”,5000)
```

1.4 Check if scan is in progress

Reads the Scan Parameters, use it for check the scan (1 = on, 0 = off).

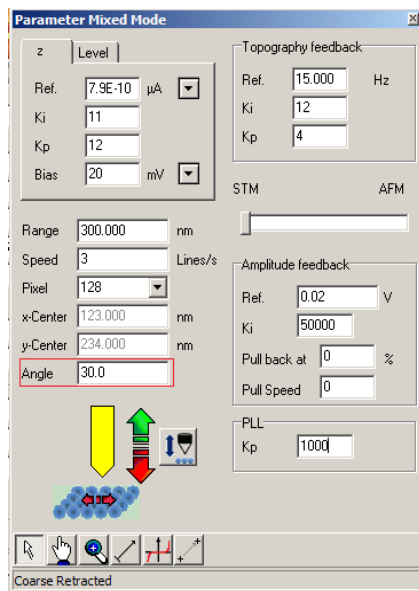
```
(MySXM.GetScanPara('Scan'))
```

2 Scanner (scanner.py)

2.1 Set/read scan area rotation area

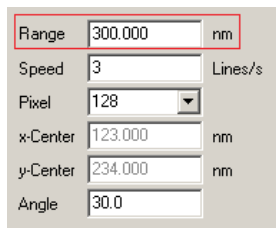
Use `MySXM.SendWait('Topic', XX)` to change the Scan Parameters.
For read the Scan Parameter use `MySXM.GetScanPara('Topic')`.

```
MySXM.SendWait("ScanPara('Angle', XX);");  
MySXM.GetScanPara('Angle');
```



2.2 Set/read scan area width

Changes the range in nm.

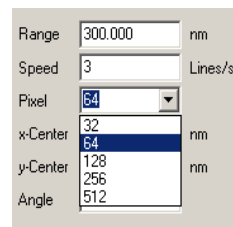


```
MySXM.SendWait("ScanPara('Range', XX);", 5000);  
MySXM.GetScanPara ('Range');
```

2.3 Set/Read number of scan points per scan line

Sets the number of Pixel.
There are 5 selection options for XX: 32, 64, 128, 256, 512.

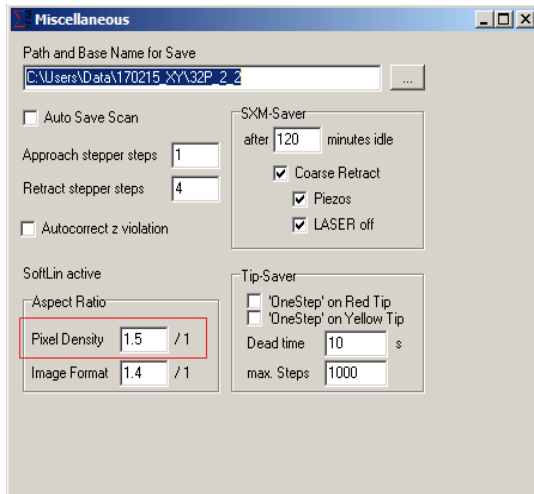
```
MySXM.SendWait ("ScanPara('Pixel', XX);", 5000);  
MySXM.GetScanPara ('Pixel');
```



2.4 Set/Read number of scan lines

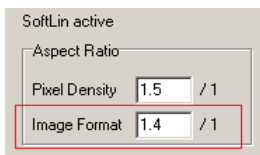
Pixel per line / Number of lines = PixelRatio. It's called 'Pixel Density' and it can be found 'Aspect Ratio' in the Miscellaneous Menu.

```
MySXM.SendWait ("ScanPara('PixelRatio', XX);", 5000);  
MySXM.GetScanPara('PixelRatio');
```



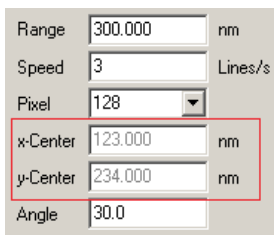
2.5 Set/Read scan area height

Width / height = AspectRatio. It's called 'ImageFormat' and you can find it under the point 'Aspect Ratio' in the Miscellaneous Menu.



```
MySXM.SendWait ("ScanPara('AspectRatio', XX);", 5000);  
MySXM.GetScanPara('Aspect Ratio');
```

2.6 Set/Read scan area offset



XX sets the X-/Y-center in nm.

```
MySXM.SendWait ("ScanPara('X', XX);", 5000);  
MySXM.GetScanPara('X');
```

```
MySXM.SendWait ("ScanPara('Y', XX);", 5000);  
MySXM.GetScanPara('Y');
```

2.7 Set/Read raster time

Sets the number of lines per second.

Range	300.000	nm
Speed	3	Lines/s
Pixel	128	
x-Center	123.000	nm
y-Center	234.000	nm
Angle	30.0	

```
MySXM.SendWait("ScanPara('Speed', XX);", 5000);  
MySXM.GetScanPara('Speed');
```

2.8 Trigger callback when scan reaches end of scan line / Trigger callback when scan finishes retrace of scan line

“b” = backward

“f” = forward

Number behind = scanline.

```
MySXM.Scan;
```

2.9 Trigger callback when scan area has been scanned completely in “up”/”down” direction

“u” = up

“d” = down

Number behind = scanline (0).

```
MySXM.Scan;
```

2.10 Enable/disable drift compensation

Disable drift compensation = set drift vector 0

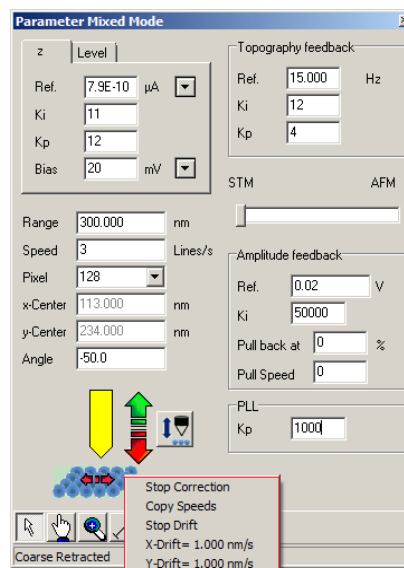
```
MySXM.SendWait("ScanPara('DriftX', 0);");  
MySXM.SendWait("ScanPara('DriftY', 0);");
```

2.11 Set/read drift compensation vector

The drift vector is visible in hint during mouse over
ScanPara-Form → below tip, on sample

```
MySXM.SendWait("ScanPara('DriftX', XX);");  
MySXM.SendWait("ScanPara('DriftY', XX);");
```

```
MySXM.GetScanPara('DriftX');  
MySXM.GetScanPara('DriftY');
```



2.12 Enable/disable plane subtraction scan mode

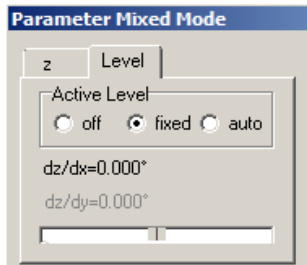
In SXM Scan Parameter-Form in the tab "Level".

Mode:

0 = off

1 = fixed

2 = auto



```
MySXM.SendWait("ScanPara('Slope', Mode);")
```

2.13 Set plane subtraction scan slope (X/Y)

Can be found in grid if the scanner file is correct.

Slope:

XX = 1.1

XX2 = -1.1

```
MySXM.SendWait("ScanPara('SlopeX', XX);");  
MySXM.SendWait("ScanPara('SlopeY', XX2);");
```

```
MySXM.GetScanPara('SlopeX')  
MySXM.GetScanPara('SlopeY')
```

2.14 Read and remember current tip position

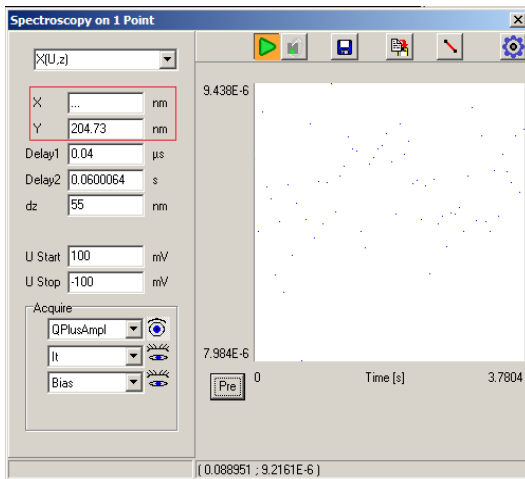
Read the x- and y-position in the DAC Channel (Menu Scale), save value as variable (oldXX, oldYY).

Scale		
DAC	ADC	Math
	Use	Name
DAC1	yes	Topo
DAC2	yes	Bias
DAC3	yes	x-direction
DAC4	yes	y-direction
DAC5	no	DA1
DAC6	no	DA2

```
MySXM.GetChannel(-2)    x-position  
MySXM.GetChannel(-3)    y-position
```

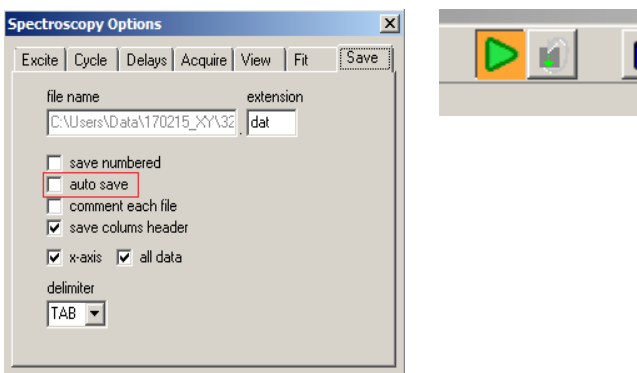

2.15 Shift tip to new position

Set a new position → x (XX) and y (YY).



```
MySXM.SendWait("SpectPara(1, XX);")  
MySXM.SendWait("SpectPara(2, YY);")
```

Start spectroscopy to move and turn auto save off.



```
MySXM.SendWait("SpectPara('AUTOSAVE', 0);")  
MySXM.SendWait("SpectStart;")
```

2.16 Return to remembered position

oldXX, oldYY (saved value from 2.14).

Start Spect to move the tip to the remembered position.

```
def BackToRememberedPosition():  
    MySXM.SendWait("SpectPara(1, oldXX);")  
    MySXM.SendWait("SpectPara(2, oldYY);")  
    MySXM.SendWait("SpectStart;")
```

```
back=BackToRememberedPosition()
```

3 Feedback Loop (feedback.py)

3.1 Enable/disable feedback loop

XX = 0 Feedback on
XX = 1 Feedback off

In the SXM Software you will find it under the menu zControl.

```
MySXM.SendWait("FeedPara('Enable', XX);")  
MySXM.GetFeedbackPara('Enable'),
```

3.2 Set feedback mode

In the SXM menu Parameter → Feedback. Use the mode 8 to have a more extensive user interface with the feedback settings.

XX	feedback mode
0	off
1	STM general
2	STM abs()
3	STM log(abs())
4	AFM contact mode
5	Amplitude R
6	Amplitude X
7	PLL AFM
8	STM + AFM

```
MySXM.SendWait("FeedPara('Mode', XX);")
```

3.3 Set ratio between AFM and STM feedback mode

You can change the ratio between AFM and STM with the bar.

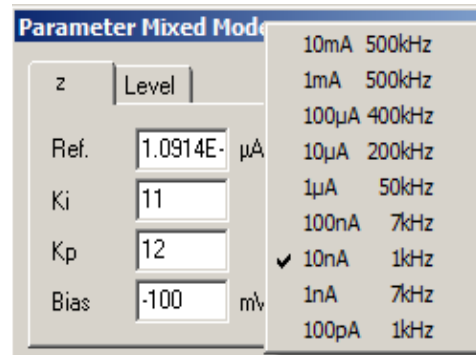
XX = ratio
XX = 0 AFM
XX = 100 STM



```
MySXM.SendWait("FeedPara('Ratio', XX);")
```

3.4 Set/read preamp tunneling current range

XX	range
0	10mA 500kHz
1	1mA 500kHz
2	100μA 400kHz
3	10μA 200kHz
4	1μA 50kHz
5	100nA 7kHz
6	10nA 1kHz
7	1nA 7kHz
8	100pA 1kHz



```
MySXM.SendWait("FeedPara('PreAmp', 1);")  
MySXM.GetFeedbackPara('PreAmp')
```

3.5 Set/read primary part of loop gain

XX1/XX2 is the value of the gain for ki/kp.

```
MySXM.SendWait("FeedPara('Ki', XX1);")  
MySXM.SendWait("FeedPara('Kp', XX2);")
```

Set gain for second loop (if enabled):

```
MySXM.SendWait("FeedPara('Ki2', XX1);")  
MySXM.SendWait("FeedPara('Kp2', XX2);")
```

Read Ki/Kp/Ki2/Kp2 value:

```
MySXM.GetFeedbackPara('Ki')
```

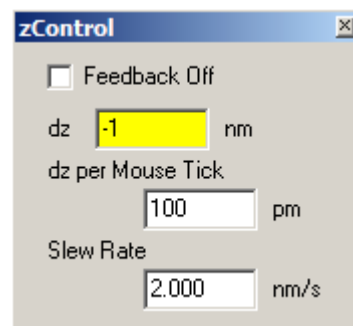
3.6 Apply a relative Z-offset

XX is the Offset in nm.

In this part the feedback have to be on!

Feedback on:

```
MySXM.SendWait("FeedPara('Enable', 0);")  
MySXM.SendWait("FeedPara('ZOffset', XX);")
```



3.7 Set slew rate for Z-offset changes

XX is Slewrate in nm/s, it can be found in the zControl menu.
In this part the Feedback have to be on!

```
MySXM.SendWait("FeedPara('ZOffsetSlew', XX);")  
MySXM.SendWait("FeedPara('Enable', 0);")
```

3.8 Read the absolute Z-position

All channels are listed with number and nae in the scale menu.
The Z-position is in Channel 1, so you have to call 0 (explained in next capture).

```
MySXM.GetChannel(0)
```

4 Channel (channel.py)

There are all the channels listed in the scale menu (DAC).

Scale								
	DAC	ADC	Math					
	Use	Name	Unit	Scale	Min	Max	Offset	Inv
DAC1	yes	Topo	nm	-8.77522E-8	-1.07E9	1.07E9	0	no
DAC2	yes	Bias	mV	9.66E-6	-1.07E9	1.07E9	3.3	no
DAC3	yes	x-direction	nm	2.28512E-6	-1.07E9	1.07E9	0	no
DAC4	yes	y-direction	nm	2.2231E-6	-1.07E9	1.07E9	0	no
DAC5	no	DA1	V	9.3E-9	-1.07E9	1E9	0	no
DAC6	no	DA2	V	9.3E-9	-1.07E9	1E9	0	no
DAC7	no	DA3	V	9.4E-9	-1.07E9	1E9	0	no
DAC8	no	not_used	V	8.6E-9	-1.07E9	1E9	0	no
DAC9	no	unknown	DAC9	1	-32000	32000	0	no
DAC10	yes	Frequency	Hz	0.00232838	-10000	2E9	0	no
DAC11	yes	Drive	V	5.47401E-10	-32000	32000	0	no
DAC12	no	unknown	DAC1	1	-32000	32000	0	no
DAC13	yes	QPlusAmpl	V	3.71857E-9	-1E9	1E9	0	no
DAC14	yes	Phase	°	0.001	-1.07E9	1.07E9	0	no
DAC15	yes	Lia1X	V	6.1236E-9	-1.07E9	1.07E9	0	no
DAC16	yes	Lia1Y	V	6.1236E-9	-1.07E9	1.07E9	0	no
DAC17	yes	Lia2X	A	1.386E-17	-1.07E9	1.07E9	0	no
DAC18	yes	Lia2Y	A	1.386E-17	-1.07E9	1.07E9	0	no
DAC19	yes	Lia3X	A	1.386E-17	-1.07E9	1.07E9	0	no
DAC20	yes	Lia3Y	A	1.386E-17	-1.07E9	1.07E9	0	no
DAC21	no	X	V	6.1236E-9	-1.07E9	1.07E9	0	no
DAC22	no	Y	V	6.1236E-9	-1.07E9	1.07E9	0	no
DAC23	yes	Lia1R	V	3.71857E-9	-32000	32000	0	no
DAC24	yes	Lia2R	A	8.41653E-18	-32000	32000	0	no
DAC25	yes	Lia3R	A	8.41653E-18	-32000	32000	0	no
DAC26	no	unknown	DAC2	1	-32000	32000	0	no
DAC27	no	unknown	DAC2	1	-32000	32000	0	no
DAC28	yes	Lia1Phi	°	0.001	-180000	180000	0	no
DAC29	yes	Lia2Phi	°	0.001	-32000	32000	0	no
DAC30	yes	Lia3Phi	°	0.001	-32000	32000	0	no
DAC31	no	unknown	DAC3	1	-32000	32000	0	no
DAC32	no	unknown	DAC3	1	-32000	32000	0	no
DAC33	no	unknown	DAC3	1	-32000	32000	0	no
DAC34	no	unknown	DAC3	1	-32000	32000	0	no
DAC35	no	unknown	DAC3	1	-32000	32000	0	no
DAC36	no	unknown	DAC3	1	-32000	32000	0	no
DAC37	no	unknown	DAC3	1	-32000	32000	0	no
DAC38	no	unknown	DAC3	1	-32000	32000	0	no

1-DACChannel = **XX** {0 to -37}

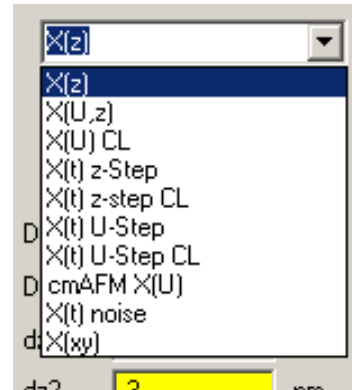
MySXM.GetChannel(**XX**);

Channel 3 = MySXM.GetChannel(-2)

5 Spectroscopy (Spectroscopy.py)

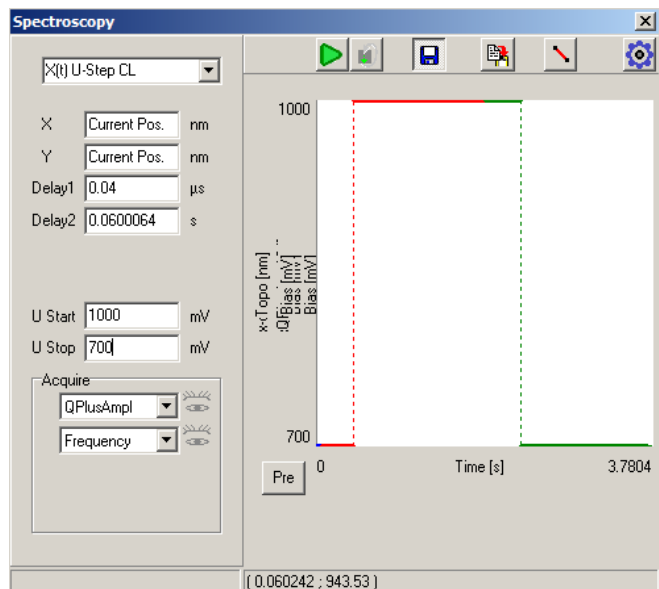
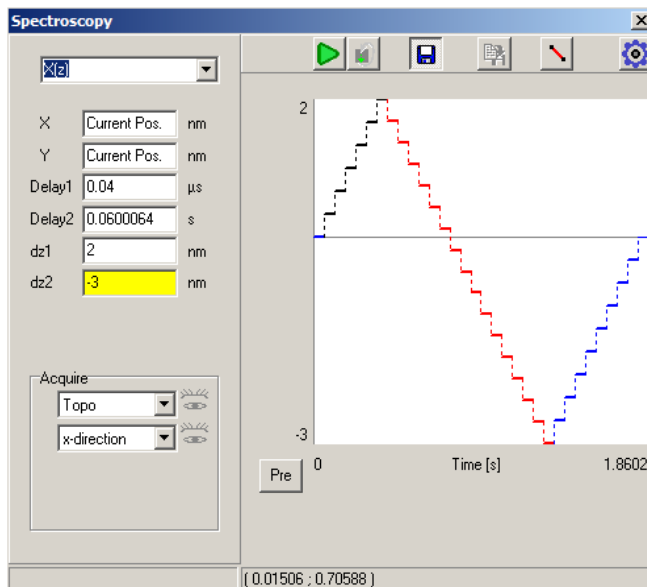
5.1 Set mode of Spectroscopy

XX	Mode of Spectroscopy
0	X(z)
1	X(U,z)
2	X(U) CL
3	X(t) z-step
4	X(t) z-step CL
5	X(t) U-step
6	X(t) U-step CL
7	cmAFM X(U)
8	X(t) noise
9	X(x,y)



0 is the combo box for mode selection. In some modes the input fields are changing, so it's important to select the mode first.

```
MySXM.SendWait("SpectPara(0, XX);")
```

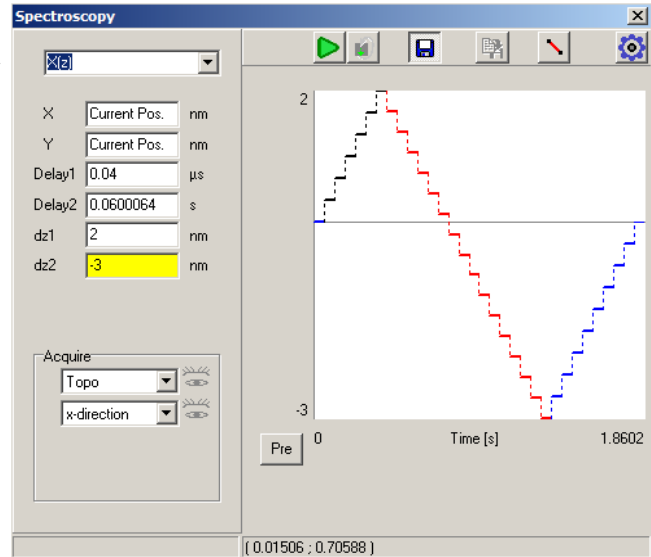


5.2 Set start/end value of (V/Z)-spectroscopy ramp

Sets the start and end value in nm.

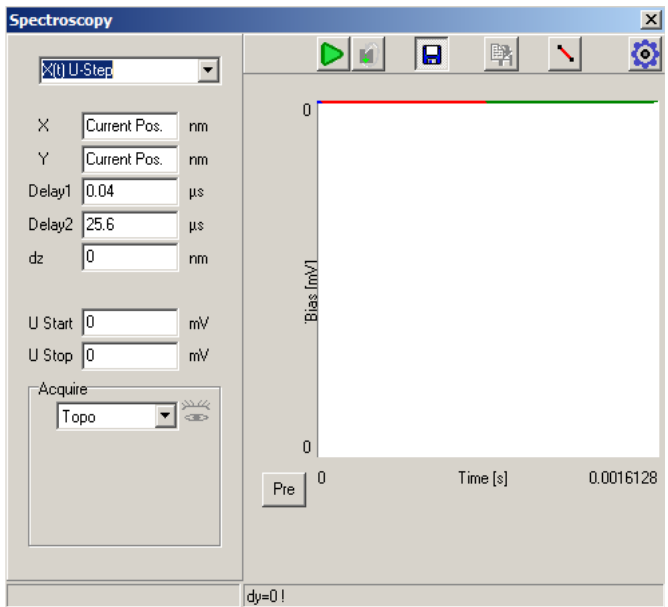
The first parameter (here 5 & 6) is the number of the user input field in the spectroscopy form. The second Parameter is the start- (dz1) and end-value (dz2).

```
MySXM.SendWait("SpectPara(5, dz1);")  
MySXM.SendWait("SpectPara(6, dz2);")
```



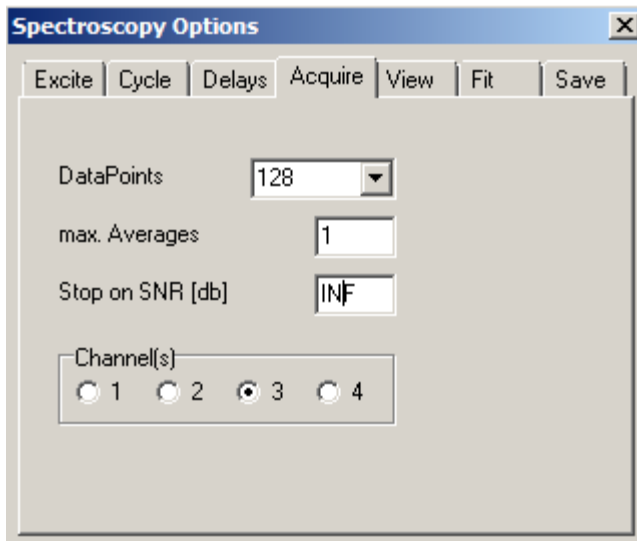
In another mode there can be other input fields. In the right example the input field 6 isn't available but for that there are two new input fields (7 & 8).

```
MySXM.SendWait("SpectPara(7, UStart);")  
MySXM.SendWait("SpectPara(8, UStop);")
```



5.3 Set number of points of (V/Z)-spectroscopy ramp

You can find it in the Spectroscopy Options menu at the point Acquire. There are 7 selection options for XX: 16, 32, 64, 128, 256, 512, 1024.



Spectroscopy Options

Excite Cycle Delays **Acquire** View Fit Save

DataPoints 128

max. Averages 1

Stop on SNR [db] INF

Channel(s)

☐ 1 ☐ 2 ☒ 3 ☐ 4

```
MySXM.SendWait("SpectPara('Points', XX);")
```

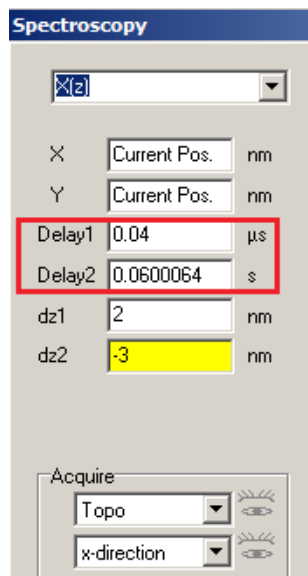
5.4 Set sample time of (V/Z)-spectroscopy ramp

The first parameter (here 3 & 4) is the number of the user input field in the spectroscopy form. The second parameter sets the delay with the parameters startX and stopX. Before you do this select the right spectroscopy mode.

Enter Delay1 in μ s and Delay2 in s.

```
MySXM.SendWait("SpectPara(3, Delay1);")
```

```
MySXM.SendWait("SpectPara(4, Delay2);")
```



Spectroscopy

X(z)

X Current Pos. nm

Y Current Pos. nm

Delay1 0.04 μ s

Delay2 0.0600064 s

dz1 2 nm

dz2 -3 nm

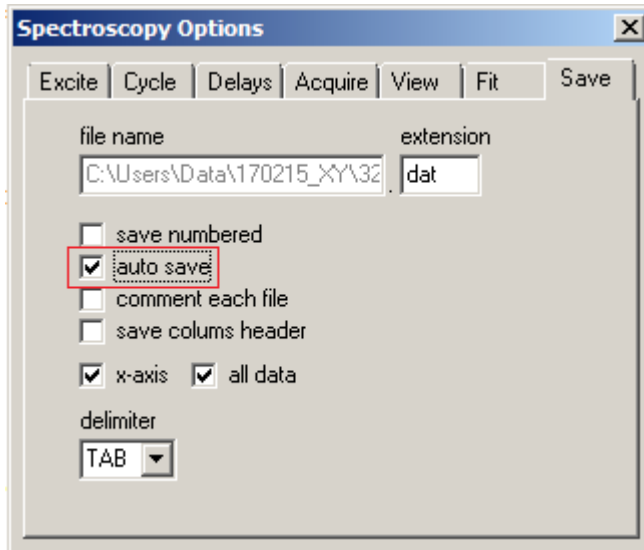
Acquire

Topo

x-direction

5.5 Trigger Callback when new data is available

Use the call `MySXM.SendWait("SpectStart;")` to start the spectroscopy.



AutoSave = 1 On; 0 = Off

`MySXM.SendWait("SpectPara('AUTOSAVE', 1);")`



AutoRepeat = 0 Off; 1 = On

`MySXM.SendWait("SpectPara('Repeat', 0);")`

With the Call `SpectSave` the spectroscopy data will be saved.

The function `MySpectSave(FileName)` prints a callback that the data is written and the filename.

`MySXM.SpectSave=MySpectSave`

5.6 Read acquired spectroscopy data (ReadSpectroscopyData.py)

Turn AutoSave on and AutoRepeat off.

AutoSave on = 1

AutoSave off = 0

```
MySXM.SendWait("SpectPara('AUTOSAVE', 1);")
```

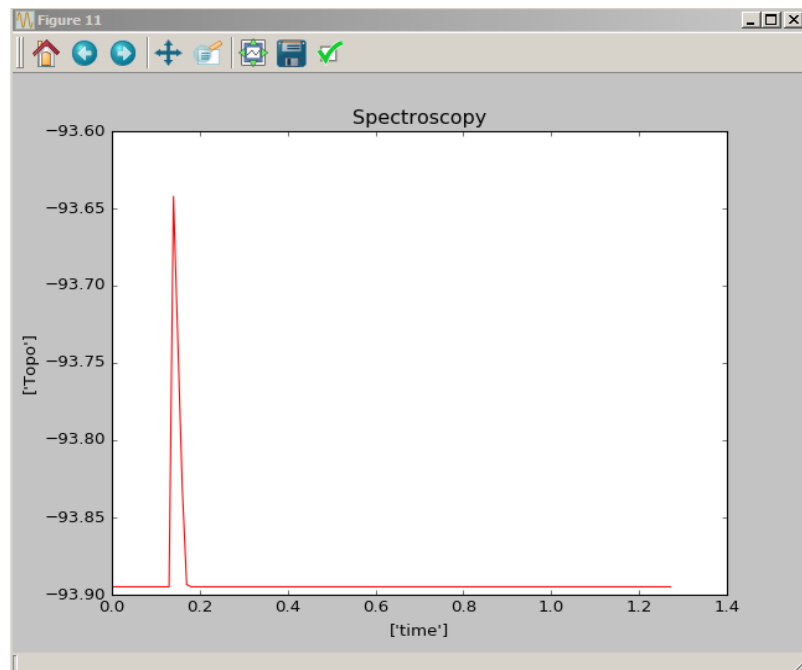
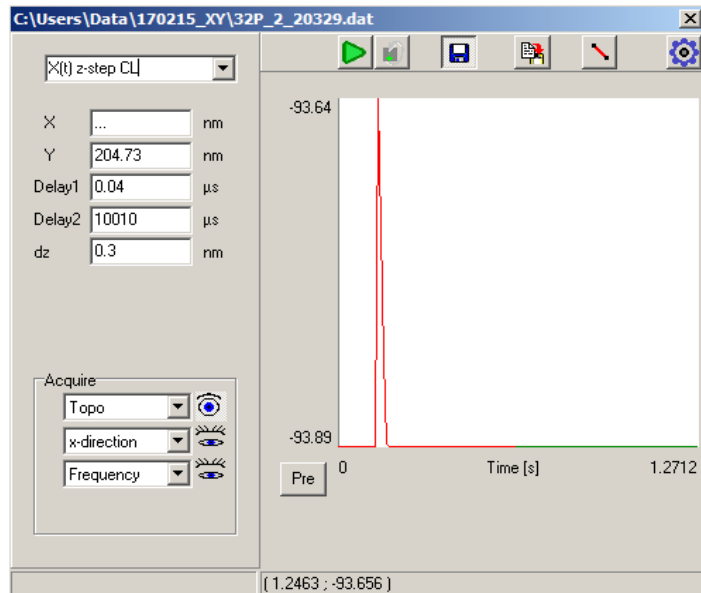
AutoRepeat off = 0

AutoRepeat on = 1

```
MySXM.SendWait("SpectPara('Repeat', 0);")
```

To plot the spectroscopy data it's necessary to import the „MyPlotFile.py“-file because it includes the PlotFile function. In this file it's possible to plot other columns of the file.

```
import MyPlotFile  
MyPlotFile.PlotFile(FileName)
```



6 Continuous signal (CollectData_sety.py)

6.1 read data

```
x=100                                → number of x values
data=[ ]
y=(ctypes.c_long*x)()                → creates an array of data
line, = self.sub.plot(y)
```

timer: used as stopwatch and to set time for read data.

while time.time()-tstart < **seconds**

read channel:

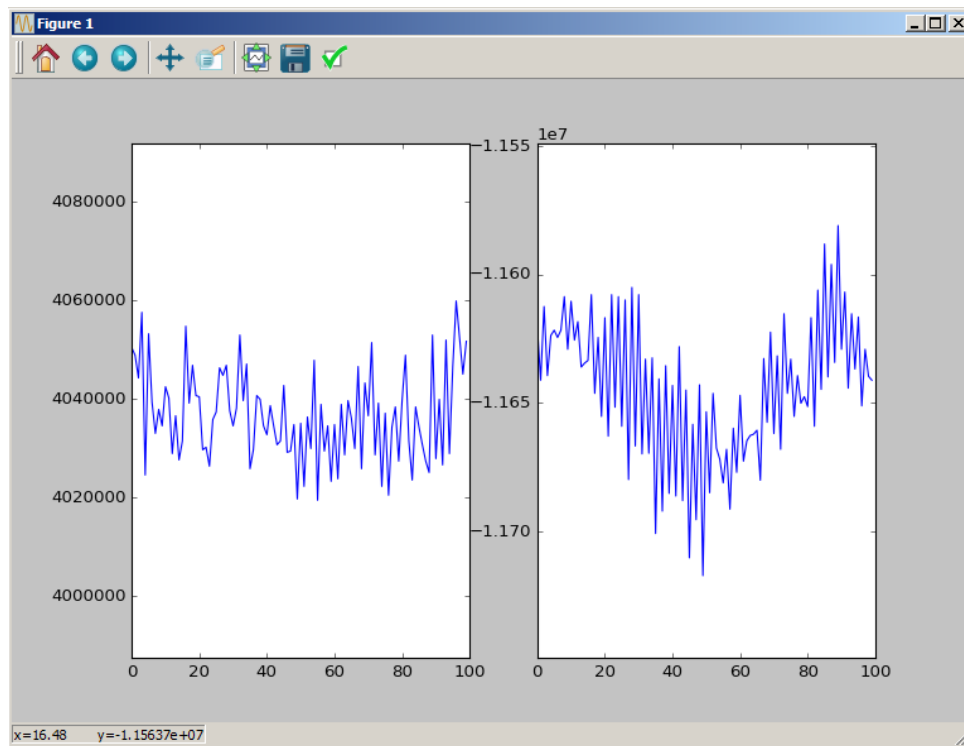
```
ptr=ctypes.cast(data, ctypes.POINTER(ctypes.c_long))
    → 1st 8 bits = value           Channel = ptr[i]
    → 2nd 8 bits = Channel number ChannelNr = Channel & 0xFF
```

```
self.sub.set_xlim([0, x]) ;          self.sub.set_ylim([Min1-32000, Max1+32000])
line.set_ydata(y)
```

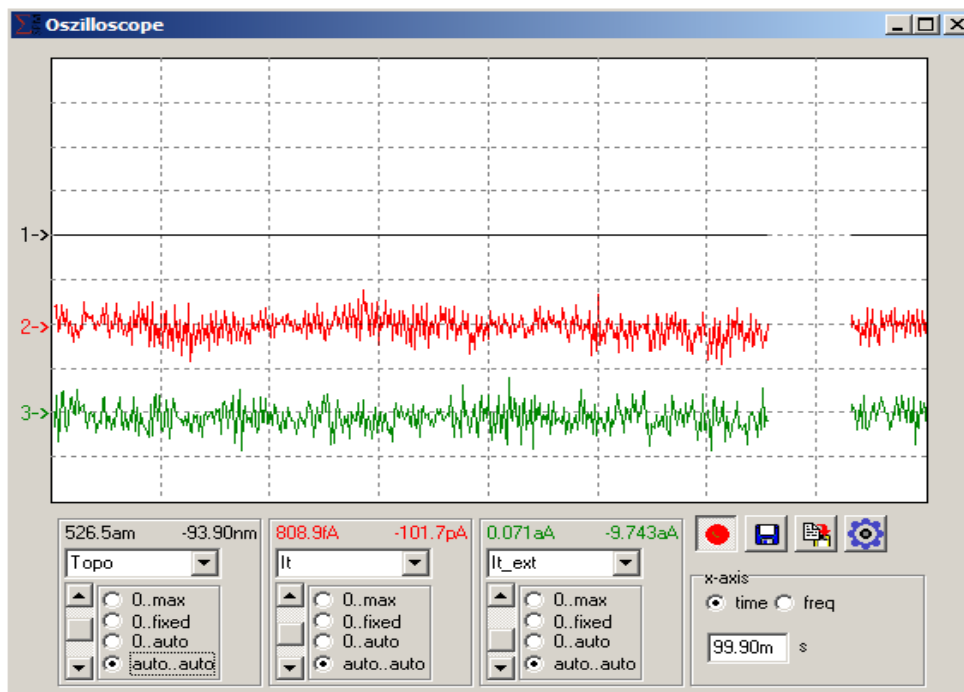
ChannelNr	Channel
0	IN A
1	LIAY
2	LIAX
3	SyncRadius
4	IN_B
5	SyncAngle
6	Lia2X
7	Lia2Y
8	Lia3X
9	Lia3Y
10	Lia4X
11	Lia4Y
12	zSlowData
13	Frame/Line Sync
14	zFastData
15	WaveCounter
16	LIA1R
17	LIA1Phi

(max. 14 channels; Nr. of Channel * frequency should <1MHz (depends on speed of PC))

```
for 1 data channel:    MySXM.SendWait("Collect('ChList', ChannelNr);")
for 2 data channels:   MySXM.SendWait("Collect('ChList', 0, 1);")
```



[plot 2 data channels]



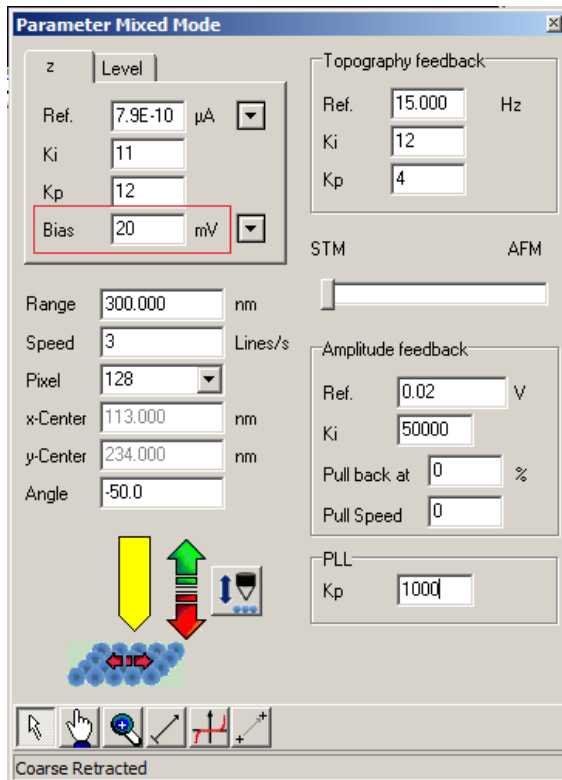
6.2 Enable/Disable

MySXM.SendWait("Collect('On', 1);") → switch on
 MySXM.SendWait("Collect('On', 0);") → switch off

7 Gap Voltage Control: bias.py

7.1 Set/read gap voltage

Bias=XX



```
MySXM.SendWait("FeedPara('Bias', XX);")  
MySXM.GetFeedbackPara('Bias')
```

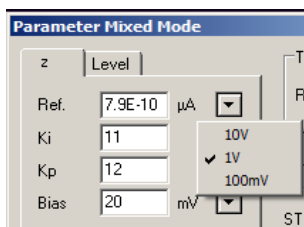
7.2 Set/read preamp gap voltage range

BiasDiv (XX):

XX = 0 → 10V

XX = 1 → 1V

XX = 2 → 100mV

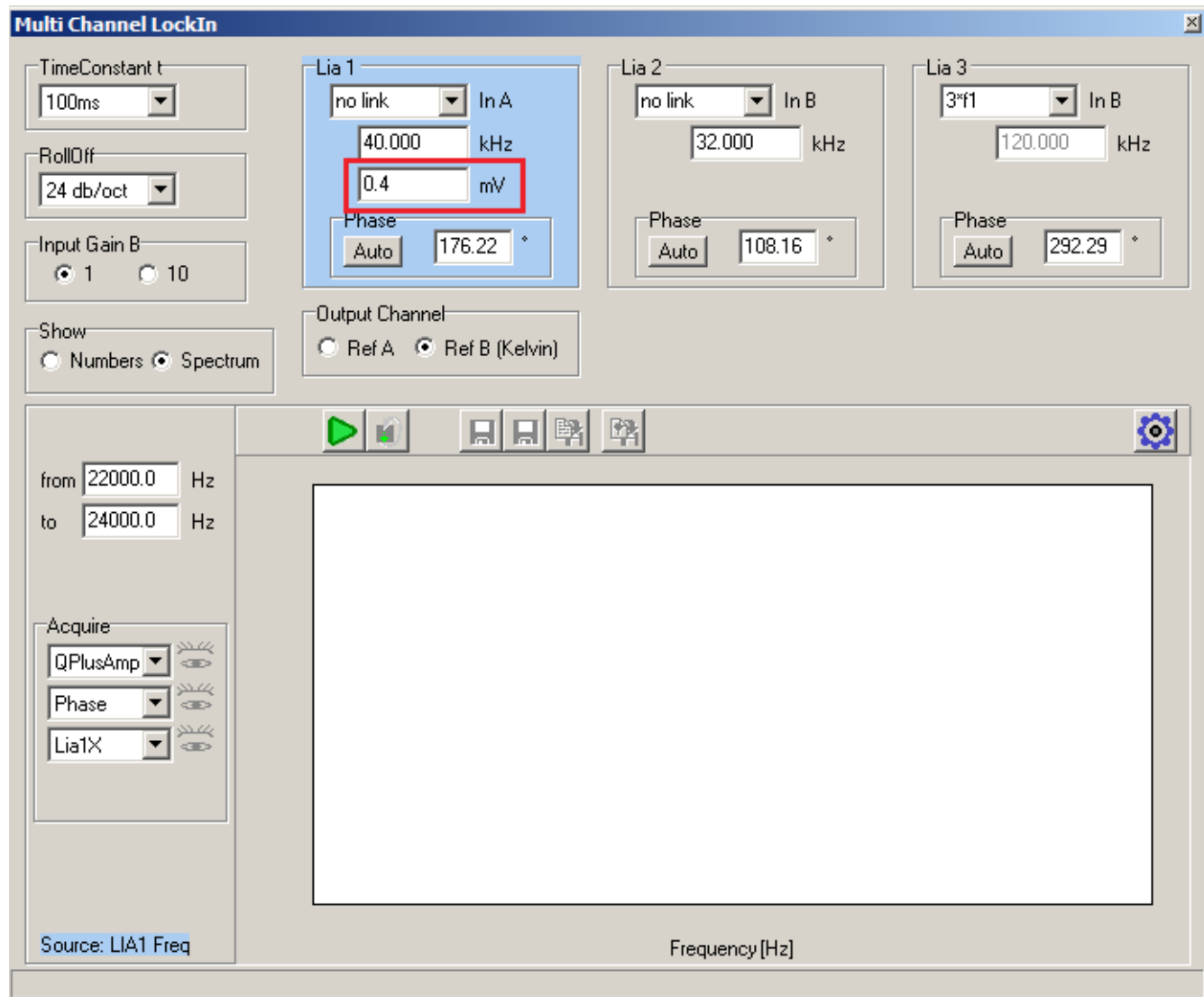


```
MySXM.SendWait("FeedPara('BiasDiv', XX);")  
MySXM.GetFeedbackPara('BiasDiv')
```

7.3 Enable/disable modulation voltage

Can be found in the MultiChannelLockIn Menu.

Drive = XX



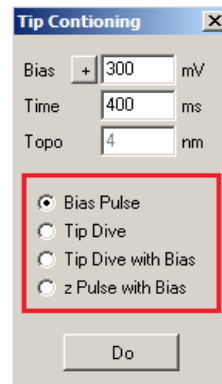
```
MySXM.SendWait("MLockin('LIA1Drive', XX);")
```

8 Tip conditioning: TipCond.py



8.1 Set conditioning mode

XX	Mode
0	Bias Pulse
1	Tip Dive
2	Tip Dive with Bias
3	Z-Pulse with Bias



```
MySXM.SendWait("TipCond('Mode', XX);")
```

8.2 Apply voltage pulse (Bias)

Change gap Voltage during tip conditioning, in units from SXM e.g. mV.

GapVoltage=XX

```
MySXM.SendWait("TipCond('Bias', XX);")
```

8.3 Configure and run Z-ramp

Set time for conditioning in ms.

DiveTime=XX

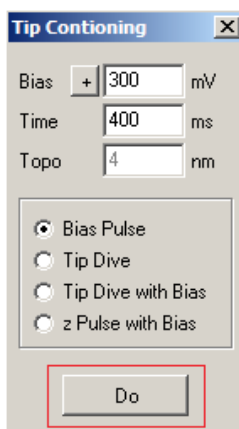
```
MySXM.SendWait("TipCond('Time', XX);")
```

Set dive dz during conditioning i nm, in menu called Topo.

dz=XX

```
MySXM.SendWait("TipCond('Dive', XX);")
```

Run Z-ramp:



```
MySXM.SendWait("TipCond('Do', 1);")
```

9 Coarse Control: coarse.py

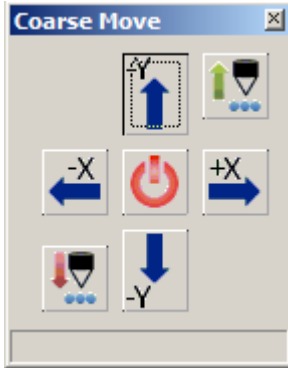
9.1 Move Tip



Move tip in X/Y direction (XX = number of steps):

+X/+Y → **XX**

-X/-Y → **-XX**



```
MySXM.SendWait("MOVE('CX', XX);")
```

```
MySXM.SendWait("MOVE('CY', XX);")
```

Z (forward/backward): > 0 → up

< 0 → down

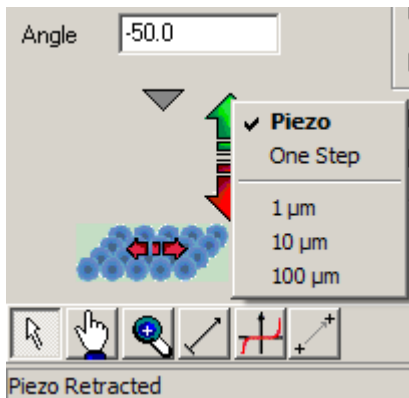
```
MySXM.SendWait("MOVE('CZ', XX);")
```

9.2 Initiate retract operation

Mode XX (from Scan Parameter Popup menu):

0 = Piezo

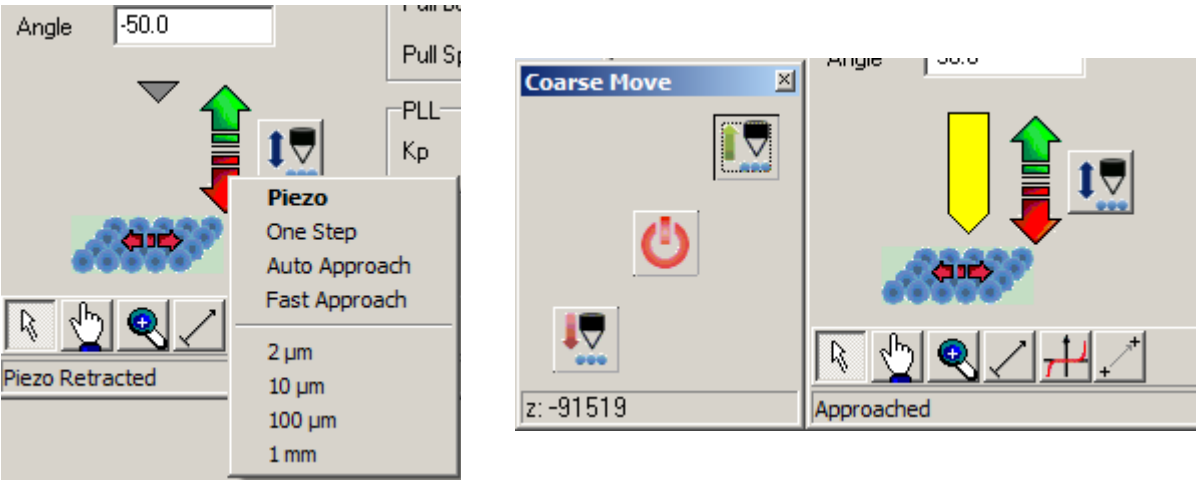
1 = OneStep



```
MySXM.SendWait("MOVE('Retract', XX);")
```


9.3 Initiate auto approach operation

XX	Mode
0	Piezo
1	OneStep
2	AutoApproach
3	fastAutoApproach



```
MySXM.SendWait("MOVE('Approach', XX);")
```

9.4 Trigger callback when auto approach operation finishes

States from FeedbackReal.MicState:

-1	Approached
0	Unknown
1	FineRetracted
2	CoarseRetracted
4	PoweredDown
5	Approaching

```
if (State.startswith(b'-1')):  
    print ('My Approached')
```

```
MySXM.MicState;
```