Language description

General Syntax

The general style is Pascal:

- ✗ There is no case sensitivity.
- ✗ The program starts with **begin** and stops at **end.**.
- ✗ **( )** - brackets are used in mathematical formulas.
- ✗ Strings are enclosed in **' '** (**" "** is also working).
- ✗ Decimal separator for floating point numbers is a point: ".".
- ✗ Each command is completed with a semicolon: "**;**".
- ✗ Available mathematical operations: **+ – * / ^ e**
- ✗ Available relational operations: **= > < <>**

Variables & Assignments

Variables are single characters only. The data format of all variables is real.
Predefined variables are:
Temp for temperature, **Uk** for the Kelvin potential and **Humi** for the Humidity, the names of the reference samples **AU**, **AL**, **HOPG**, **USER1** and **USER2** as well as **Ch** as result of GetChannel.

Example: `A := A + 1.2;`

Command Overview

Command	Meaning
ClrScr	Deletes the visible data on the data screen
Color (r,g,b)	Set the color of the next drawn data line
if ... then	Condition
Execute('filename')	Executes another script *./script/filename.scr*
Filename ('name1')	Defines the name of the data storage file as *./data/name1.txt*
Filename(input)	User set the filename in the Memo
for ... to ...	Loop definition
Freilauf(fl)	Defines the safety height for line and point measurements in steps
Goto *marke*	Jumps to the position *marke* defined as Label
Image(s,d)	Acquires a square image with s pixels and d mm x d mm size
Line(dir,s,d)	Measures *s* points from here to *d* along a line in direction *dir*
MoveRef(x,y,z)	Moves the contact to a new absolute position
MoveSonde(x,y,z)	Moves the sensor to a new absolute position
Point(x,y)	Measures data set at the absolute coordinate (x,y)
SetPoint(sp)	Sets the parameter SetPoint in V

SetChannel('string',value)	Sets the output D/A-channels
SetKelvin('string',value)	Changes variables for the Kelvin-Measurement
SetLockin('string', value)	Changes variables for the lockin amplifier
SetPoint(sp)	Sets the setpoint in V
ToggleTime(tt)	Sets the parameter Toggle Time in seconds
Wait(t)	Waits for a time t in seconds
Writeln('text',..)	Displays text in the Memo

Commands (alphabetical)

ClrSrc

deletes all visible data and spectra on the data screen.
Example: `ClrSrc;`

Color (*red,green, blue*)

sets the color for the next graph to be drawn in the diagram
Parameters: **red, green, blue:** integer 0 .. 255
Example: `Color(0,0,0)`   → sets the color to black
                     `Color(255,0,0)`   → sets the color to red

Image (*points,delta*)

Scans an image
Parameters: **points:** integer
                     delta: real
Example:
`Image(64,5);`   → scans an image with 64 x 64 points over a range of 5 mm x 5 mm

if   *condition*   then   *command*

Evaluation of conditions. ELSE is not supported.
Example:
`Point(100.56, 200.87);`
`if  (U < -1.6) then writeln('Potential in Al out of range =', U)`

Execute (scriptfilename)

Calls and executes another script file.
Parameters: scriptfilename: string
Example:
`Execute('Ref-HOPG');`   → calls the script " *./scipts/Ref-HOPG.scr* ".

FileName (datafile*name*)

defines a file name for the data file, which collects all data during the measurements.

Parameters:	**datafilename :** string	→ file is *./data/name.txt*
Example:	**Filename('test3');**	→ the data are written to *./data/test3.txt*

FileName(input)

The user gets the possibility to insert the desired filename directly in the Memo.

Example: *Filename(input);*

→ *Memo text:* **image5x5,** *the data are written to ./data/image5x5.txt*

for *start counter* to *stop counter* do begin *command(s) end;*

Loop definition based on an integer counter. The commands **begin** and **end** are always required.

Example (measures 20 points along a line in x-direction):

```
X:=100;  Y:= 100;
for i:=1 to 20 do begin
  Point(X,Y);
  X:=X+i/10;
end;
```

Freilauf (height)

defines a relative height above the sample surface in steps (for Single Point KP System) or mm (for Scanning KP systems), which is used during movements between point or linear measurements called with **point(x,y)** or **line('type',steps,endpos)**. **After** each call of **point** or **line**, the KP head is lifted to "Freilauf" relatively from its measurement position.

Parameters: **height:** real

Example: **Freilauf(5.1);**→ enters "5.1" in the entry "Freilauf" in the parameter section of the KP software resulting in 5.1 mm or 5 stepper steps movements

GetChannel(channelnumber)

Acquires the data of one A/D input channel.

Parameters: **channelnumber:** integer

Channelnumber	Meaning		Channelnumber	Meaning
0 / 1 / 4 / 5	Auxiliary In1 / In2 / In3 / In 4		16	Temperature (AD-In)
10 ...13	X, Y, R, Phi		17	Humidity
14	Uk [V]		18	Temperature (USB)
15	Kelvin Slope		19	Absolute Z-Height [mm]

Example: **GetChannel(10);**→ reads the input data of the reference electrode

writeln('X=',Ch);→ writes the result of GetChannel in the Memo

GetOffset

Starts the procedure, which detects the offset automatically channel.

Example: `GetOffset;`

Goto *marke*

Allows to jump to a *marke*. It requires:
- the definition of a label with a **LABEL** statement
- a **Goto** command followed by the labels name
- the label followed by a colon

Labels always start with a character..

Example:

```
LABEL foo, foo2;
begin
  Point(100,100);
  if U > 0.05 then Goto foo;
  Goto foo2;
foo:
  writeln('Reference Potential too large. U =',U);
foo2;
end.
```

Image (*steps, distance*)

starts an automated image acquisition from the current position towards the direction +X and +Y.

Parameters: **steps :** number of steps along x- and y-direction
(always square image)
distance : size of the image in mm

Example: **Image** `(128 , 20 );` → acquires an image with 128 x 128
point. The points are distributed over an area of 20 mm x 20 mm.
After each measured point, the tip is retracted to the height "Freilauf"; the next position is chosen
and the tip is automatically approached until the "SetPoint" is reached. After each line, the tip is
additionally retracted in a safer height for long distance movement. After the whole image, the tips
back to the start position. Data are stored in "./picture" folder.

Line (*type, steps, distance*)

starts an automated measurement along a line

Parameters: **type :** direction of the line scan, can be `'X','Y'` or `'t'` for time
steps : number of steps between start and end point
distance : relative position of end point vs. the current position in mm
or in seconds for the type = 't'

Example: **Line** `('X', 10 , 20 );` → measures 10 points along a line of
20 mm length in x-direction. After each measured point, the tip is retracted to the height "Freilauf";
the next position is chosen and the tip is automatically approached until the "SetPoint" is reached.

MoveRef (*x,y,z*) – not used in AFT_KP150 !

positions the top contact at the coordinates (X,Y, Z) given in mm.
Parameters: **x, y, z:** real
Example: `MoveRef(200,25.3,5.1);` → the contact is retracted to a safe height of z = 50 mm in z-direction; is laterally moved to the coordinate (x = 200 mm , y = 25.3 mm) – first X then Y movement – and vertically approached to the final height z = 5.1 mm. The safe height is defined in the user.ini as "SafeZPos".

MoveSonde (*x,y,z*)

positions the sensor head at the coordinates (X,Y, Z) given in mm.
Parameters: **x, y, z:** real
Example: `MoveSonde(76.2,50,2.6);` → the head is retracted to a safe height of z = 50 mm in z-direction; is laterally moved to the coordinate (x = 76.2 mm , y = 50 mm) – first X then Y movement – and vertically approached to the final height z = 2.6 mm.

Point (*x* , *y*)

measures a surface potential at the position (x,y) given in mm. The tip is retracted from the current position to the z-value "Freilauf" and moved to the new position. Then, the tip is approached automatically based on the value SetPoint. The Kelvin potential is acquired with an integration time given as 2 * ToggleTime.
Parameters: **x , y** : real
Example: `Point(100, 200);` → acquires a Kelvin potential at the point
                                                                          x = 100 mm and y = 200 mm.
                        `Writeln (Uk);`   → displays the result in the Memo

SetChannel (*'string', value*)

Set the D/A-converter outputs to certain values:
Parameters: **string :** 0...3   → channel number
                        value : real   → value in volts
Example: `SetChannel('2',1.2);` → sets the voltage at Aux-Out1 to 1.2 V

Channelnumber	Meaning		Channelnumber	Meaning
0	Sample		2	Out1
1	Tip		4	Out2

SetKelvin (*'string', value*)

Changes parameters in the Kelvin-Window as follows:
Parameters: **string :** **B**ias, **D**eadTime, **T**oggleTime, **O**ffset, **F**reilauf, **S**etPoint
                                    (or the 1st character: B, D, T, O, F, or S)
                        value : real   → value in displayed units
Example: `SetKelvin('F',0.1);` → sets the parameter Freilauf to 0.1 mm
                        or

```
SetKelvin('Freilauf',0.1);
```

SetLockin ('*string*', *value*)

Changes parameters in the Kelvin-Window as follows:

Parameters:	**string :** **F**requency, **A**mplitude, **P**hase, **H**armonic
	(or the 1st character: F, A, P, or H)
	value : real → value in displayed units
Example:	`SetLockin('F',1000);`→ sets the frequency to 1 kHz
	or
	`SetLockin('Frequency',1000);`

SetPoint (*value*)

defines the SetPoint for the distance feedback. The automated approach used in point or linear measurements called with **`point(x,y)`** or **`line('type',steps,endpos)`** stops, when the signal X reaches this value is given in V.

| **Parameters:** | **value :** real → value in volts |
| **Example:** | `SetPoint(0.0015);`→ the auto-approach stops at 1.5 mV |

ToggleTime(*value*)

defines the time for each measurement cycle (switching time of the applied backing potential) in s.

| **Parameters:** | **value** : real |
| **Example:** | `ToggleTime(0.8);` → the bias is 0.8 ms positive and 0.8 ms negative in all following measurements. |

Wait (*time*)

Allows to pause the operation a time given in seconds.

| **Parameters:** | **time** : real |
| **Example:** | `Wait(1.5);`   → waits for 1.5 seconds |

Writeln ('*text1*', *variable*, '*text2*', *variable2* ...)

Allows to write additional information in the memo. In the brackets, a series of strings, variables and numbers can be written, which are separated by commas. Strings should be enclosed in `' '`.

Parameters:	**text1, text2 :** string; **variable1, variable2** : real
Example:	`a := 5 ;` → sets the value of the variable *a* to 5
	`writeln('a=', a);` → writes "a = 5" in the Memo