SPECS®

Surface Analysis and Computer Technology

Control and Acquisition Software

ErLEED RFA-PC

User Manual

With Batch Programming

3.01.1

User manual for the ErLEED control and acquisition software *RFA-PC*
with batch programming.

Version 3.01.2 of the 25.01.2002.

SPECS order number for this manual: 78000136.

SPECS GmbH
Voltastr. 5
13355 Berlin
Germany
phone +49 30 467824-0, fax +49 30 4642083,
http://www.specs.de
support@specs.de

Chapter

T

Table of contents

Chapter

1

Introduction

1 Introduction

The software serves for controlling a LEED/AES ErLEED 3000D control unit by a RS232 interface. This interface is standard equipment of the unit. The unit can be operated in various modes. It is not possible to change the mode by the program presented here. These modes can be selected only by the plug on the back plate of the unit.

As mentioned earlier control of the unit is effected through the interface RS232. This control is in competition with control by the keyboard of the unit which cannot be deactivated with the software. The control with the keyboard has always priority to the commands entered via the interface. The display on the PC is updated frequently and the general display of the values has two fractional digits.

In the AES mode a lock-in amplifier can be controlled additionally. The following description refers to a full version of RFA-PC. In some (freeware) versions (labeled ErLEED-PC), however, the user is only capable to control all voltages but not to perform measurements, i.e. to record AES data. This applies for LEED customers only. These customers can skip everything related to AES.

Chapter

2

System requirements

2 System requirements

Requirements for operating the software are:

- Windows 95, 98, NT4, NT 5 or 2000
- Pentium PC is recommended
- at least 128MB RAM
- VGA Display with a resolution of at least 800 x 600
- one serial interface (two with an optional external lock-in amplifier)

Chapter

3

Additional hardware

3 Additional hardware

The devices and the PC are always connected through RS232 interfaces.

In order to record additional test signals in the AES mode an ADIODA-12 (of the brand WASCO) A/D-converter board can be integrated in the PC.

The drivers for the hardware components listed above are already integrated in this software version. In case other components, i. e. a different A/D boards, are to be implemented the software can be upgraded with the necessary drivers by the software producer.

An integrated lock-in amplifier is used. Optional an external lock-in amplifier of the type EG&G model 5106 or 5105 can be used in the operating mode AES.

Chapter

4

Installation

4 Installation

The software installation is implemented under Windows. Insert the floppy disc *ErLEED Controlsoftware (RFA-PC)* and start *SETUP.EXE* after having selected the disc drive in the file.

The installation of *runtime engine* of National Instruments and of the RFA-PC *control software* is implemented. There is a default for the directory for the RFA-PC. By the installation a program group is established over which the software can then be started.
Now the RFA-PC Software is completely installed. For LEED customers only the software is called ErLEED-PC as a RFA-PC (freeware) version with restricted functionality. The following does *not* apply for LEED customers with ErLEED-PC:

The serial number printed on the disc labeling has to be in accordance with the serial number on the unit. If this is not the case no operation is possible, the operating modes LEED and AES cannot be selected.
The serial number of the control software is being shown in the window ***About.****. The serial number of the unit is written on the back plate.*

A keyfile is installed with a set-up routine coming with the software in a separate folder. This keyfile has to be installed in the same directory than the software itself to ensure communication between the unit and the PC.

Chapter

5

Connecting the devices

5 Connecting the devices

Before starting-up the system and the software for AES the devices must be connected with the included cables as shown in *figures 1. Figure 1a* describes the connection of the ErLEED 3000D unit with an integrated lock-in amplifier. If you have an external lock-in amplifier, you must connect the devices as shown in *figure 1b.* As any COM-interfaces may be chosen, these figures are only an example.

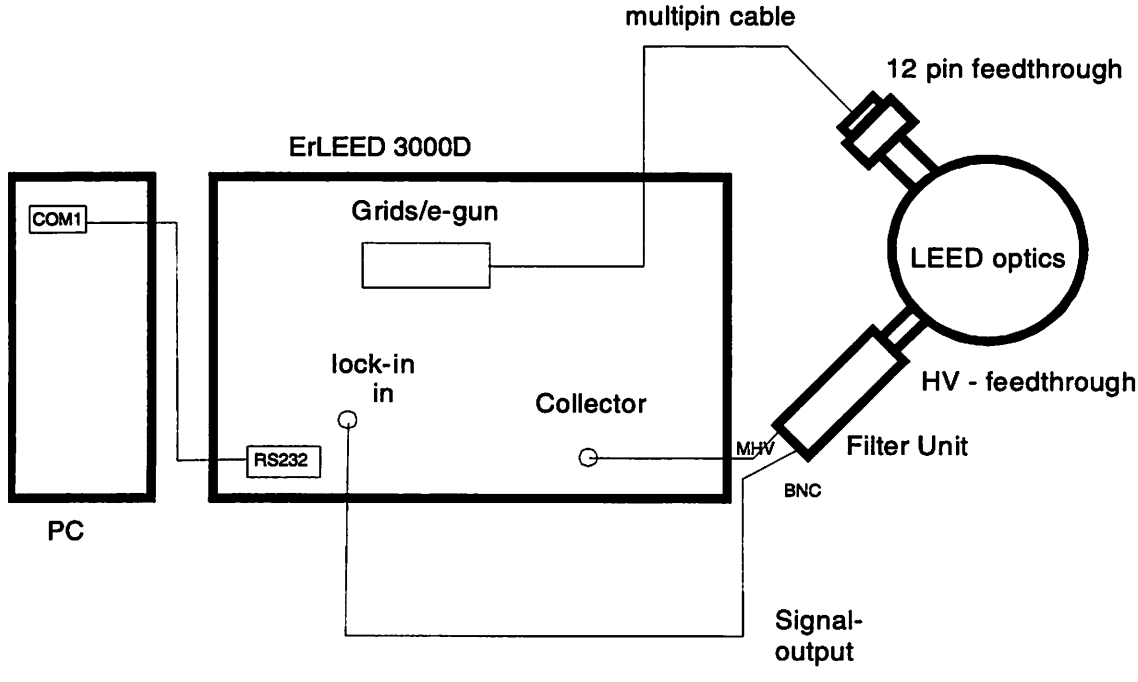

figure 1a: connecting the devices with integrated lock-in amplifier

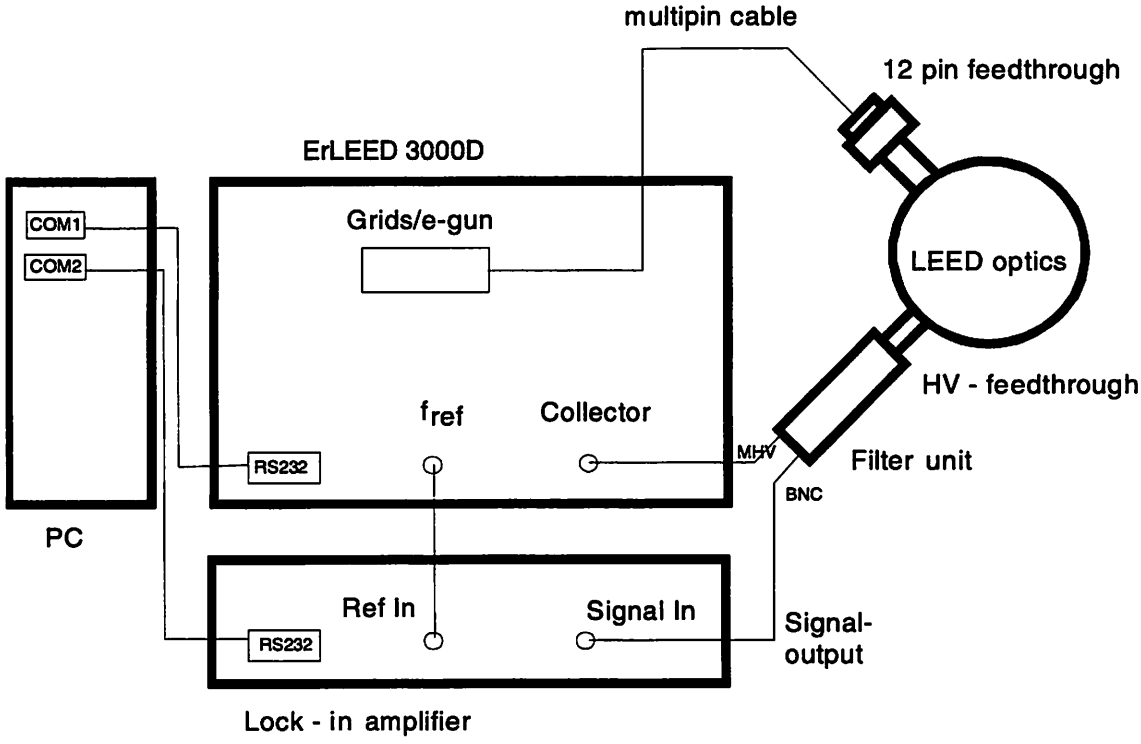

figure 1b: connecting the devices with external lock-in amplifier

Chapter 6

Starting-up the control software

6 Starting-up the control software

****

!!! Attention !!!

In the operating mode AES as well as in the LEED mode there is no modification of the present state of the unit when starting the control software, i. e. all configurations of the unit are saved. A complete parameter setting of the unit apart from the cathode is implemented only by loading the configuration file. There are input elements for adjusting individual voltages and a separate window for manipulating the cathode control.

Differences in the firmware of the unit

If the version number of the firmware of the unit is less than 1.1 a booting of the unit is implemented after starting the control software and adjusting the interface of the unit. The status display of the oscillator switches will be correct only if they are in OFF-state while starting the control software and if modifications are implemented only with the PC.

****

Switch on the PC and the unit after connection of the unit with the PC through the interface RS232. Now start the control program under Windows and adjust the interface to which the unit is connected under the menu item *Interface.* The control software starts immediately communicating with the unit. If the communication is without errors and the serial numbers are the same the software switches over to the present mode of the unit.
The operating mode LEED offers a main window for setting module voltages and a window for cathode control.
The graphics display is the main window in the operating mode AES. For adjusting the voltages click on the menu item *modules\aes-modules.* A window containing the variable voltages appears. If there is a lock-in amplifier of the type EG&G 5106 or 5105 it can be operated by a further COM-interface. For opening the interface select the menu item *Interface\Lockin* in the operating mode AES and click on the *open* key in the window *interface.* By selecting the menu item *lock-in!* a window is opened that displays the present configurations and in which the parameters of the lock-in amplifier can be adjusted; e. g. the function *Auto Quadrature Zero* can be triggered. After having set the ramp values under the menu item *meas\time\ramp* the execution of the ramp can be triggered by clicking on the *start* key in the main window.
The program is quitted by the key combination *ALT-F4* or by the *quit* key. An inquiry follows in order to prevent unintentional termination.

6.1 The Ini-file

When leaving the program the *Ini-file* is produced automatically. It is stored in the same directory as the program and contains information on the last program run. It serves to reset system configurations. The following data is stored in the Ini-file:

- Interface number for the unit
- Interface number for the lock-in amplifier
- last configuration file
- last batch file

If *autoopen* has been selected for the interface, the interface is opened when starting the program. Only the parameters of the active channels and their configuration of the configuration file applied last is loaded. The voltage values of the modules are not read-in by the file. The name of the batch file serves as default for a batch run.

6.2 Operating the software

Regarding the operation the software follows the Windows standard. Examples are activation of the next element by the *tabulator key* or selection of the desired element by the *control key* together with the underlined letter. Modal windows, that are framed blue, suppress the operation of all elements, until it is closed again.

Operating a numerical input element

By standard a numerical input element is structured as shown in *figure* 2: The parameter can be put in directly (numeric keypad). It can also be adjusted after selecting the field, clicking on the field name with the mouse or by addressing the field with the tabulator key, with the cursor keys *up/down*, or by activating the corresponding cursor on the left side of the numerals. When entering directly, the parameter will be taken only when the *return*-key has been pressed or an other element has been selected.
If there are no cursor keys in the entry window, this element is only for display. Then the parameter cannot be altered by the user.

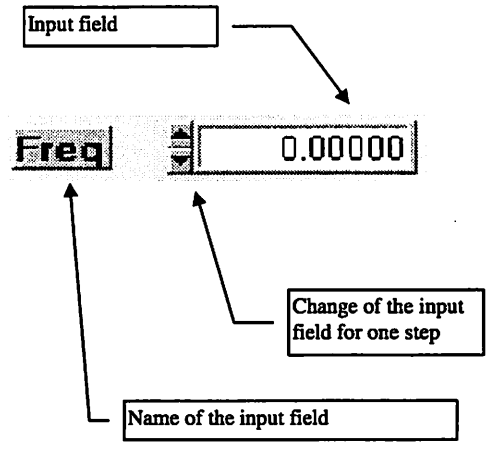

figure 2: numerical input element

Chapter

7

Operating modes of the unit

7 Operating modes of the unit

All figures in the following text have the structure of the software version 1.20. The indicated adjusted values are not necessarily the best in reality.

The control unit has various operating modes which exist also in the software. Additionally there is an operating mode of the software in which no unit is connected.

The following operating modes are possible:

1. No connection to the unit *NOTCON*
2. Unit in the operating mode *CAL/OFF*
3. Unit in the operating mode *LEED*
4. Unit in the operating mode *AES*

With the software the user can select the desired operating mode of the unit. With no connection between the PC and the unit (interface not open, unit not switched on), the program is in operating mode 1. If there is a connection the present operating mode of the unit is selected. The operating modes 2 or 3 and 4 depend on the connection of the unit to the optics and on the closed sense contact. By the software you can switch between the operating modes 3 and 4.

If any changes occur during the operation, e. g. the unit is booting, all input elements are locked and the user is informed. As looping of the optics may trigger a reset of the unit, and the PC does not realize whether there has been a change of mode or a reset at the unit, the PC-software does not switch automatically to the new mode.

All control elements for manipulation of data at the unit are blocked. Now you can store all data and measured values.

Then press the *checkunit* key in the menu *interface.* Now the new status is determined and the software automatically switches to the mode selected at the unit.

7.1 The interfaces

The interfaces for the unit and the lock-in are selected in the same way. The transmission parameters are defaults for devices, only the interface number can be modified. figure 3: shows the structure of the window opened by selecting the menu item *interface\unit* in the menu list. A driver is implemented for the RS232 for both devices. By the *parameter* key a window is opened displaying the present interface parameters and for adjusting the desired interface number for the device. The *open* key opens the interface and checks whether the appropriate device is connected. If this is not the case there is a failure report. As long as no interface has been activated there is a NONE in the display line of the active interface. The *close* key closes the connection.

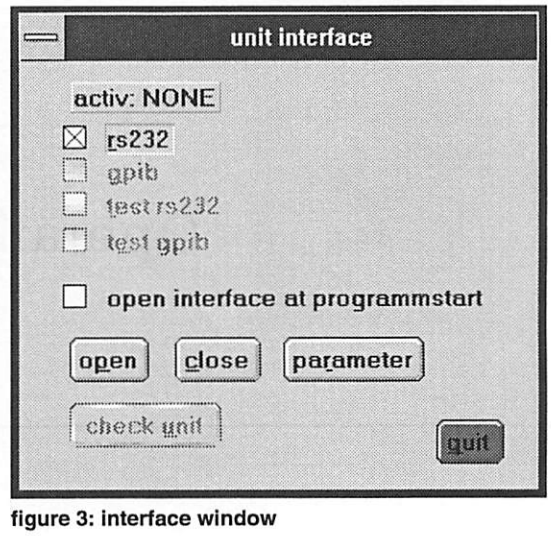

figure 3: interface window

The *quit* key terminates input and closes the window.

Activating the option *open interface at programstart* has an effect only when starting the program the next time. As described in *6.1 The Ini-file* the interface data is stored when quitting the program. If this option is activated the interface is opened automatically when starting the program. The same check-ups are implemented as when opening manually.

7.2 No connection to the unit

This mode is taken on if the interface has not been opened or if the unit has been switched off. The unit mode *CAL/OFF* is a different mode. Select an interface (e. g. COM1) under the menu item *interface* in order to set up a connection with the unit. figure 4: *operating mode not connected* shows the screen layout of this operating mode.

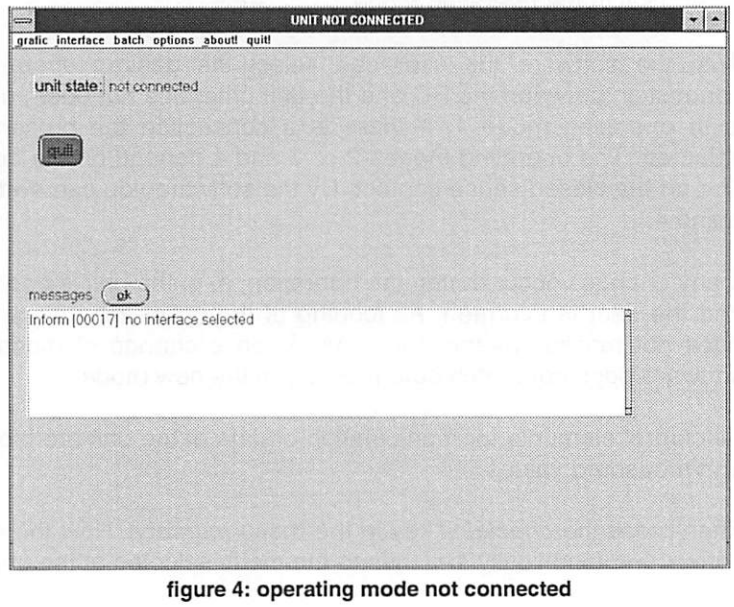

figure 4: operating mode not connected

7.3 Unit in the operating mode CAL/OFF

As long as the unit is in the mode CAL/OFF (unit switched on, no operating mode selected or during self-test) the program is also in this mode. If the self-test was successful and an operating mode has been selected for the unit the program also switches to this operating mode. Preconditions are that the unit is connected to the PC and the interface is open. The screen layout of this operating mode is shown in *figure 5: operating mode cal/off.* The user can modify the program status only by closing the interface to the unit or selecting an operating mode at the unit.

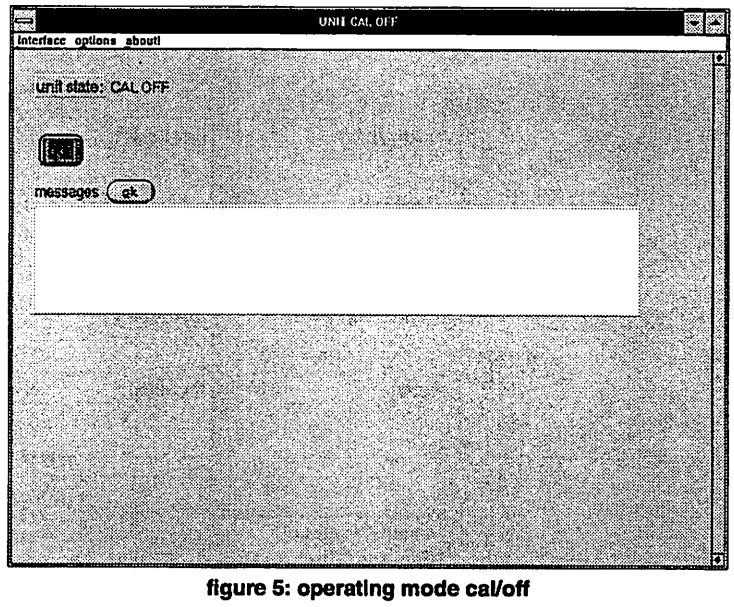

figure 5: operating mode cal/off

In this operating mode the unit is inquired constantly, hence changes of the operating mode are recognized immediately.

7.4 Unit in the operating mode LEED

If the LEED mode has been selected at the unit by plugging the optics connection cable into the main-jack and selecting the mode in the system menu of the unit, the program also branches into this mode after the unit has terminated booting and its connection has been set up.

7.4.1 Surface

The basic surface structure of this operating mode is shown in *Figure 6: Operating mode LEED.*

All the display elements of a module are put together into a frame. Input elements can be recognized by the cursors on their left side for voltage increasing/decreasing. The first element displayed after the module name is always the actual value indicating the value measured at the signal output. Slight differences may occur between the set nominal value and the measured actual value.

Depending on the module type the nominal value for *Gain/Offset/Value* can be preset. Regarding display, please note that in the case of certain modules the output voltage is related to the energy in a default function. This is the reason for a possible difference between the sum of the nominal values and the actual value. The nominal value is always greater or equal the actual value. The screen is an exception, as in this case the function has a negative sign..

There is a separate window for adjusting the cathode. Only the present state and the set values are displayed here.

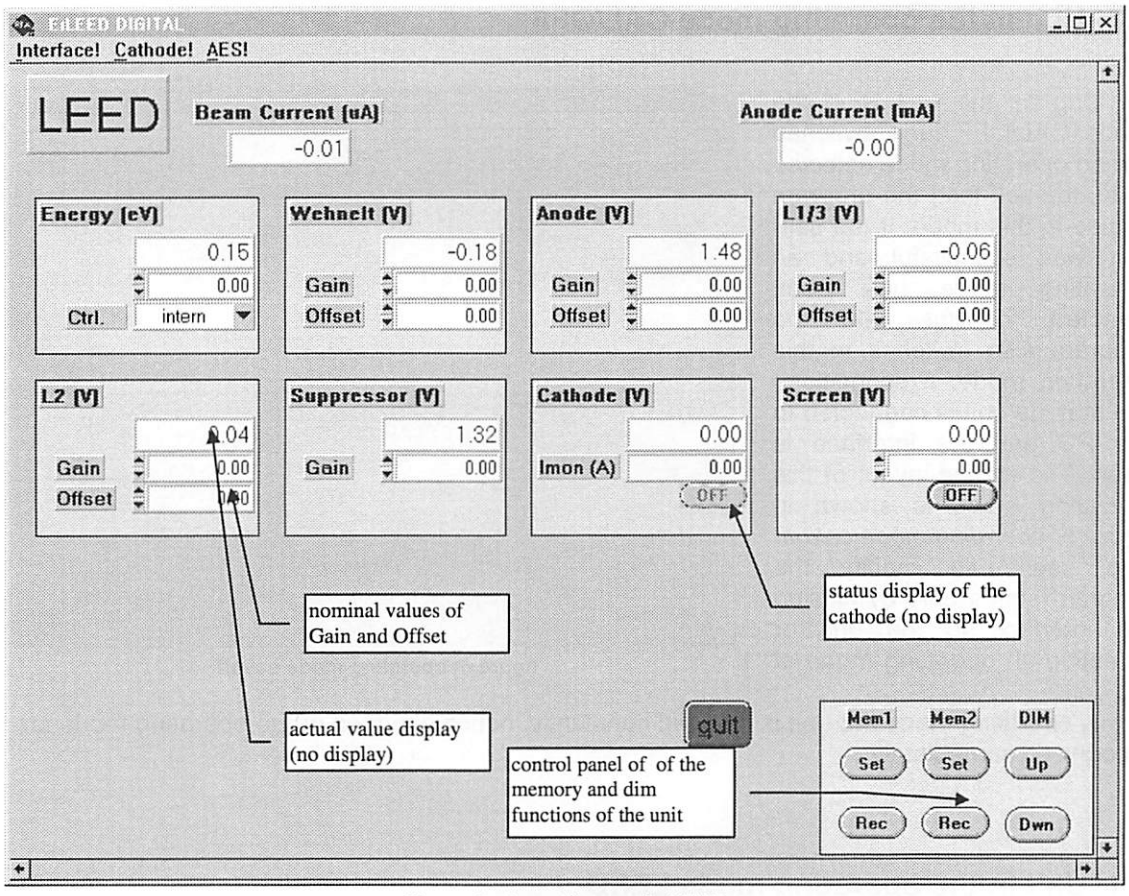

Figure 6: Operating mode LEED

7.4.2 Surface functions

The following surface functions are available:
file menu contains

- load/store configuration
- activation of an input window for comments

modules menu contains

- window for cathode adjustment

interface menu contains

- configuration of interface to the unit

options menu contains

- set-up window for service parameters (only with password)
- change of the unit mode

about menu contains

- information window

quit menu contains

- program end

Further elements:

- key for quitting the program after repeated inquiry
- display elements for all values
- configuration elements for all variable voltages and energies
- non-locking key for operating the memory functions of the unit (Mem1, Mem2).
- non-locking key for dimming the display of the unit (DIM)

The functions are similar to the functions of the AES mode and are explained there in detail.

7.5 Unit in the operating mode AES

If the operating mode AES has been selected at the unit by plugging the optics connection cable into the main-jack and selecting the mode in the system menu on the unit, the program also branches into this mode when the unit has finished booting and its connection has been set up. Unit control and lock-in amplifier control are important features of this operating mode. To this end the second RS232 is necessary. The measurement values of the lock-in amplifier can be re-read and stored. During a measurement run the measurement values can be displayed online in a graphics display with a vector line. All configured data and measured values can be saved in a configuration file and also loaded again. After a configuration file containing measured values has been loaded, these values are displayed in a graphics display. To avoid misinterpretations the graph and hence the measured values disappear when module parameters are altered or a new measurement is started.

7.5.1 Surface

The basic surface structure of this operating mode is shown in *figure 7: operating mode AES*.

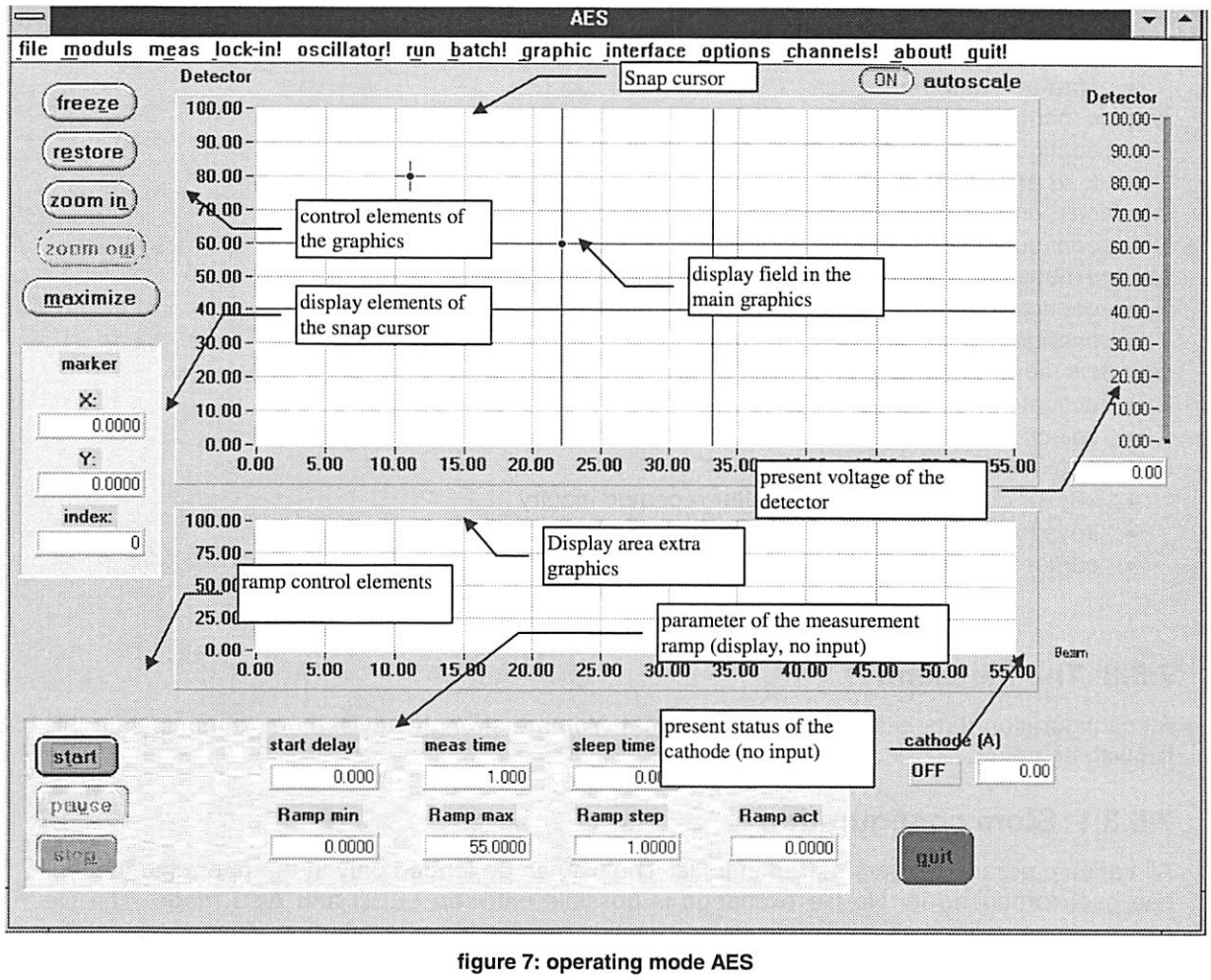

figure 7: operating mode AES

7.5.2 Surface functions

The following surface functions are available:
file menu contains
- load/store configuration
- opening of an input window for comments
- output of measurement values in DBF-format

modules menu contains
- configuration window for module voltages
- configuration window for the cathode
- to-zero fill of all voltages

meas menu contains
- configuration of ramp parameters
- shift of automeasure function

lock-in menu contains
- configuration of the lock-in amplifier

oscilator menu contains
- oscillator configuration

run menu contains
- ramp start

batch menu contains
- start of a batch run

graphic menu contains
- configure graphics options
- load additional graph

interface menu contains
- configure interfaces for unit and lock-in

options menu contains
- configuration window for service parameter (only with password))
- change of the unit mode

channels menu contains
- activate/deactivate channels
- selection of the measurand

Further elements:
- key for quitting the program after repeated inquiry
- keys for ramp control
- cursor for zoom and signal measurement in the main graphics

7.5.3 The file menu

All configuration data is stored in ASCII-format. You are strongly advised not to manipulate this file with an editor.

7.5.3.1 Store configuration

All variable parameters are stored in a file. The file can be loaded only in the operating mode it has been produced in. No file exchange is possible between LEED and AES mode. The file name can be defined either directly or by selection out of a file selection box. These files can be exchanged between different units.
As long as no manipulation of modules or ramp parameters has followed the measurement, the user can decide whether the measured values (graphical data) is also stored in the file together with the parameters. The parameters of the lock-in amplifier are always stored in the configuration file.

7.5.3.2 Load configuration

A configuration file generated in the AES mode is loaded. Before selecting the file the user can decide whether the lock-in parameters and a set of measurement values, if existing, are also to be loaded from the configuration file. The data on the measurement values switches over the defaults for the file extensions and in this way controls the display in the display box. It is always possible to load a file of an other group. If a file has been loaded containing measured values the values are then graphically displayed.
Not all parameters are read in when loading a configuration file. This concerns configurations of documentation such as the interface configuration, when and in which program version the file has been generated and parameters not desired by the user (lock-in amplifier or measurement data). If a data set of a selected file is faulty it will be ignored and not accepted, the standard values are set and the user is advised.
The parameters of the cathode are very critical, therefore they are treated separately. If there is any difference between the set and the read-in parameters the user is advised. You can then decide whether the new values are to be taken over. There is no automatic alteration of cathode parameters at the unit.
Any alteration of cathode energy must be triggered by the user in the cathode window.
After loading the data the configuration parameters for the modules are transferred to the unit. If a data set of measured values has been loaded it is displayed in the graphics. If parameters for the lock-in amplifier have been loaded they are set.

7.5.3.3 Store measured values in data base format

After a measurement run the generated data can be filed in data base format. This means that the values are put out in lines separated by tabulator and with a declaration line on top describing the individual columns. figure 8 shows an example.

Nr	Scan	Detector	CH--2	CH--3	CH--4	CH--5	Beam
0000	0.000000	0.000000	-3.649902	0.000000	-3.588867	0.000000	1164708.000000
0001	1.000000	0.000000	-3.154297	0.000000	-3.176269	0.000000	1357636.000000
0002	2.000000	0.000000	-2.978515	0.000000	-2.944336	0.000000	1383594.000000
0003	3.000000	0.000000	-3.156738	0.000000	-3.244629	0.000000	1308932.000000
.....							

figure 8: example of a data base file

You can use this file format to import the data into other programs.

7.5.3.4 Editor

This menu item activates the editor *NOTEPAD.EXE* included in the Windows standard package.

7.5.3.5 Print

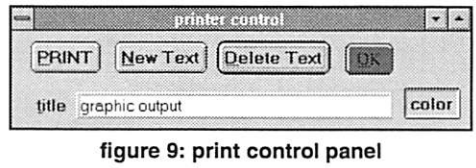

figure 9: print control panel

To document measurement results the main graph with all measurement signals can be printed by a Windows printer.
To this end a graphics window is opened that displays all graphics contained in the main graphics of the main panel. In addition the control window for print output is opened as shown in *figure 9.* In this window text windows can be generated that can be placed anywhere on the screen. A newly generated text window has to be activated before text input by clicking with the mouse. The text that has been put in appears together with the graphics on the printout, the headline of the text window serving to place it is not printed. The graphics panel can have any title you like. The result can be printed by a printer installed under Windows. The user can choose between a color and a black-and-white printout of the graphics.

Before starting to print you can select a printer, and, if necessary, adjust the parameters. The *ok* key starts the print procedure.

Take care that the parameters *X-Resolution* and *Y-Resolution* are in accordance with the selected printer. *Paper Width* and *Paper Height* must be indicated in 1/10 mm. This, however, refers to portrait format. If a postscript printer is used it might be necessary to implement paper alignment (portrait/landscape) in the print menu under Windows system control.

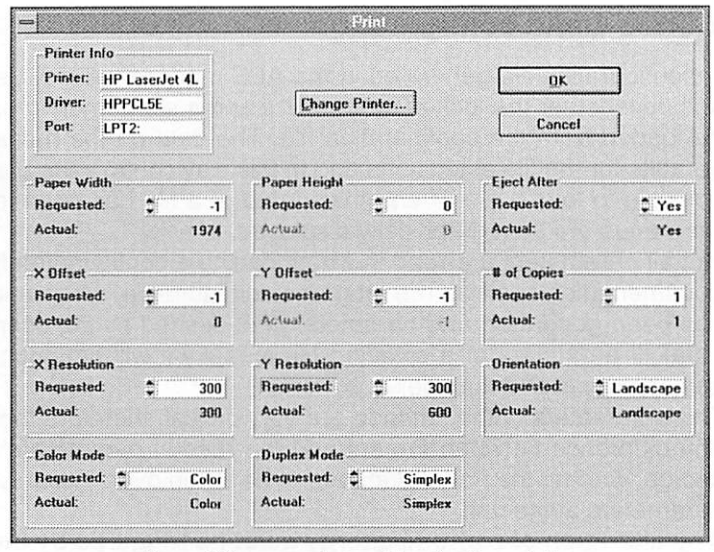

figure 10: parameter setting for the printer

figure 11 shows the window that is to be printed with the graphics and texts. In this example an additional graph was loaded in the main window (see *7.5.10.7 Load graph*), that is put out in the print window, too. This graph has a dotted line.

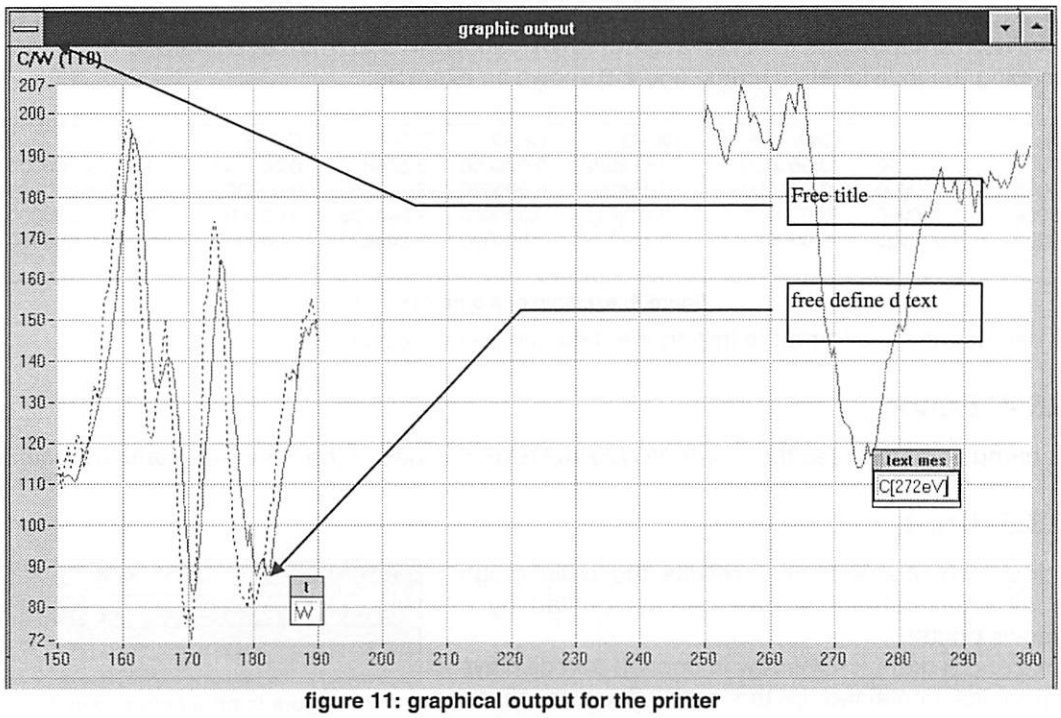

figure 11: graphical output for the printer

7.5.3.6 Comment

Selecting this menu item activates an input window for editing comments up to a length of 20 lines. This comment is stored in the configuration file, not in the Ini-file. When loading a configuration file the comment contained in it is also loaded. If a configuration is stored after a file has been loaded the loaded comment is stored in the new file, hence it is passed on.

7.5.4 The module menu

7.5.4.1 AES - modules

All display elements of a module are put together into a frame. Input elements can be recognized by the cursors for voltage increasing/decreasing on their left side. The first element displayed after the module name is always the actual value of the module indicating the actual value measured at the signal output . Slight deviations between the set nominal value and the measured actual value may occur. Depending on the module type the nominal value for Gain, Offset and Value is variable. Regarding display please note that, in the case of several modules, the output voltage is related to the energy in a preset function. This may result in a difference between the sum of the nominal value and the actual value. Then the nominal value is always greater of equal than the actual value.

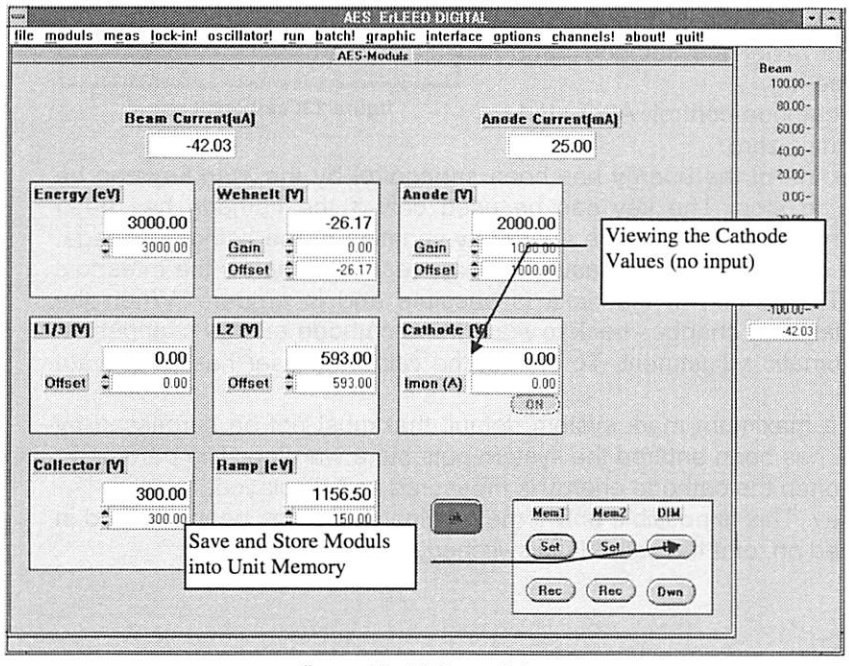

figure 12: AES -modules

This window displays the parameters set at the unit and contains also elements to manipulate them. The cursor keys can be operated with three setting speeds. The setting speed is determined by a second key pressed additionally (shift or control). If the cursor key is pressed only alterations of all modules but the collector are implemented by one volt increments. The collector has an increment of 50 volt. If the shift key is pressed additionally the increment is multiplied by 10, with the control key it is multiplied by 100.

There is a separate window for cathode control where only the actual status and the set parameters are displayed. The keys memory Set/recall and DIM up/Down correspond to the unit in their functions. The memory functions are executed by the unit and can be used e. g. for intermediate storage of module configurations. The memory function is triggered by double-clicking on the corresponding key, the DIM function by a simple clicking.

7.5.4.2 Set to zero

This function sets Gain/Offset/Value at zero volt in the case of all modules except from the cathode.

7.5.4.3 Cathode control

To protect the cathode it is controlled in a separate window. *figure 13:* shows the layout.
Cathode control is completely different from the control of the other modules. The cathode has a preset nominal value for the energy while the other modules have a preset value for the voltage. In order to protect the cathode from damage the voltage is adjusted by a maximum increment of 0.1 volt and a minimum delay of 1.0 seconds. The user can modify these parameters to get a smaller or lower increase. The smaller the resistance the smaller the increment should be. In the worst case the *set point* (see below) can be surpassed by ½ increment. The reason is the change of resistance by the heating. The faster the cathode ramp is implemented the bigger the time-lag.
The switch *cathode* activates cathode control. After start-up the cathode voltage always equals zero.

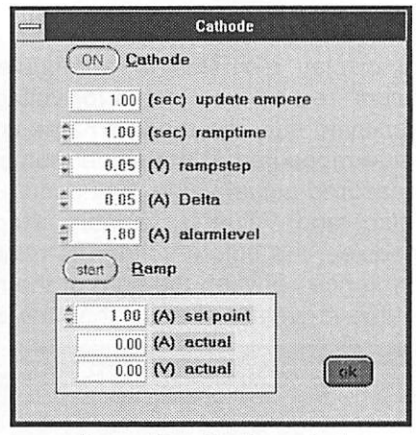

figure 13: cathode control

After the nominal value (set point) of the energy has been set control by the *start* key can be activated. The label changes to *stop*. The key can be used only if the cathode has been switched on. You receive the nominal value for the energy by variation of the cathode voltage. The abort condition is that the deviation of the measured energy (*actual*) is below the threshold (*delta*) defined by the user. The default of the delta is absolute and in ampere. When the nominal value is reached the labeling changes back to *start*. If the cathode energy changes, e. g. by heating, there is no automatic adjustment. To correct the value the user has to activate cathode control again.
The parameter *alarm level* is a maximum mark system default that must not be surpassed by the *set point*. If a greater value has been entered the system puts out a warning. The parameter *update ampere* indicates how often the cathode energy is measured and displayed.
The window is left by the *ok* key. This is possible only if the nominal value has been reached in the case the cathode is switched on, or if the cathode is switched off.

7.5.5 The Meas-menu

7.5.5.1 Setting of parameters for ramp control

The ramp parameters displayed in the main window can be adjusted in the window shown in *figure 14:*
The following parameters are to be set for ramp control:

1. start value of the ramp (*ramp min)*
1. final value of the ramp (*ramp max)*
2. increment (*ramp step)*
3. measurement time per step (*measure time)*
4. waiting time after setting a new value (*sleep time)*
5. waiting time after setting the first value, start delay (*start delay)*

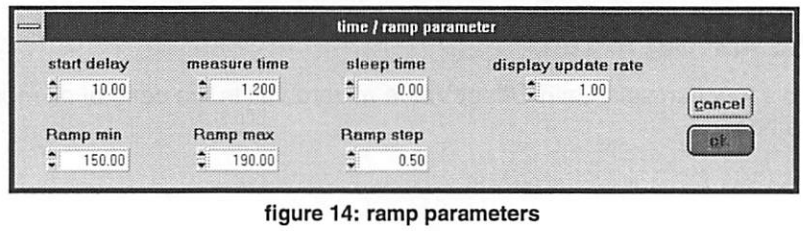

figure 14: ramp parameters

The configurations are variable only if no measurement is active. The numbers can be put in directly, they can be manipulated by the mouse or the cursor key.
If, with activated lock-in amplifier, the measurement time (*measure time*) per step is smaller than the time constant of the lock-in filter, the user is informed when leaving the window. You cannot modify the lock-in parameters in this window. There is a warning every time a measurement is started.

By the *ok* key the set parameters are accepted and the window is left. The *cancel* key leaves the window without accepting the modifications for storage.
The parameter *startdelay* serves to define a one-time delay at the beginning of measurement. As in the most cases the actual value of the ramp does not conform with the start value, a waiting time is necessary after setting the first ramp value, that may result in a considerable jump, in order to stabilize the system. Before a new measurement is started there is a waiting time indicated in the parameter *sleeptime* for all further steps, that are generally decisively smaller.

7.5.5.2 Automatic measurement

The active state of the switch *automeasure* is marked by the symbol ✓ before the menu item. If there is no measurement, i. e. no ramp is active, there is one measurement of the measurand set as main channel after the time set with the parameter *display update rate* has expired. The value is put out in the bar chart on the right of the graph. This display is shown also in *figure 12: AES -modules*. Hence the influence of a modification of module voltage on the measurand is clearly and immediately visible.

7.5.6 The lock-in menu

If you are using the integrated lock-in amplifier you get the panel shown in *figure15.*
All parameters relevant to the user for controlling the lock-in amplifier are displayed in the panel. There is a supervision of limit values for every element. The key *Find Peak* triggers the function to find the phase with the maximum X - Result from the amplifier.

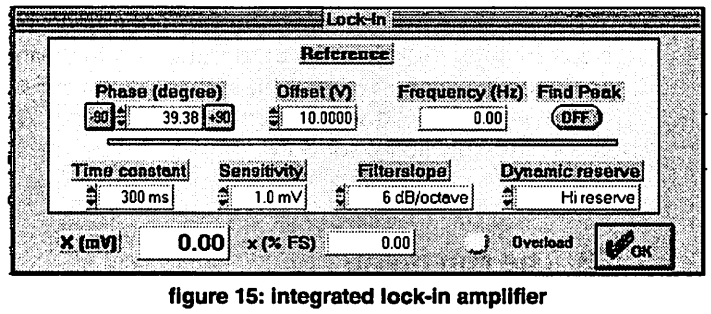

figure 15: integrated lock-in amplifier

If you are using the external lock-in amplifier, you get the panel shown in *figure 16*
All parameters relevant to the user for controlling the lock-in amplifier are displayed in the panel. There is a supervision of limit values for every element. The parameter descriptions are to be taken from the handbook of the amplifier.
The key *Auto quad* triggers the function *auto quadrature zero* at the lock-in amplifier. However, the amplifier does not echo after completion. The user can only supervise the signal phase and the X component that express the state of the function.
The key *Auto x offset* triggers the function auto offset at the lock-in. The current output signals of X and Y are moved to the zero line. A maximum of 20 % of the sensitivity can be used as offset. Manual setting of the parameters is also possible.

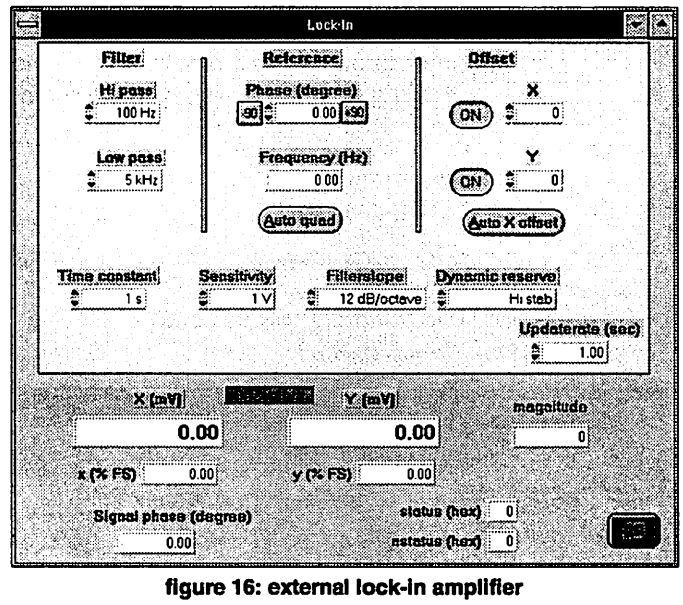

figure 16: external lock-in amplifier

Alterations of the configuration parameters are passed on to the amplifier immediately. The display elements in the lower third of the window, e. g. for the X and Y components , the signal phase etc. are up-dated permanently. Modifications of the configuration parameters are immediately visible. The percent displays for the X and Y components conform with the measured values returned by the lock-in. These are normed at 1000 independent of the

measurement area. With these values and based on the selected measurement area a conversion in milli- or microvolt is implemented.
If this window is opened and there is the status information *not active* concerning the lock-in the interface is not open. To get into the active state close this window and open the one for interface configuration. This window cannot be selected during measurement and it is closed with the *ok* key.

7.5.7 The oscillator menu

The window shown in *figure 17:* is for the configuration of the frequency generator.
The frequency, the amplitude and the coupling point (grid or primary energy) are variable. The reference frequency can also be doubled. This frequency is the reference of the lock-in. The frequency settings implemented here must be taken in consideration when configuring the lock-in amplifier, repeated adjustment of the phases might be necessary.
Value modifications are immediately accepted by the unit. By the *ok* key the window is left. The *cancel* key restores the old values and leaves the window. As described in section *Starting-up the control software* there may be differences between the status display of the switches of the unit and this window. If this is the case all switches of the unit should be in OFF-state before starting the unit and status modifications should be implemented only in ths window.

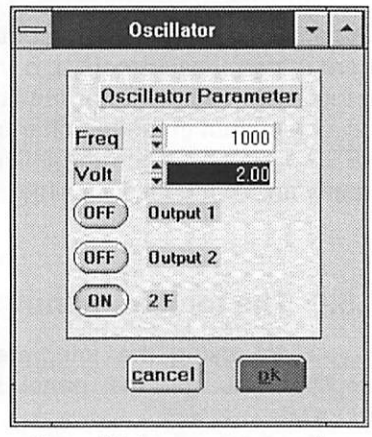

figure 17: frequency generator

7.5.8 The run menu

7.5.8.1 Start of a measurement

This menu item activates the start of a ramp run and the appropriate measurements of all active channels. It corresponds to the start key in the main menu.

7.5.9 The batch menu

This window is for controlling a batch run (for details on batch runs see *Batch programming* page 25).
By the *select* key a file can be selected of the file box. The *edit* key activates the NOTEPAD-editor with this file as default.

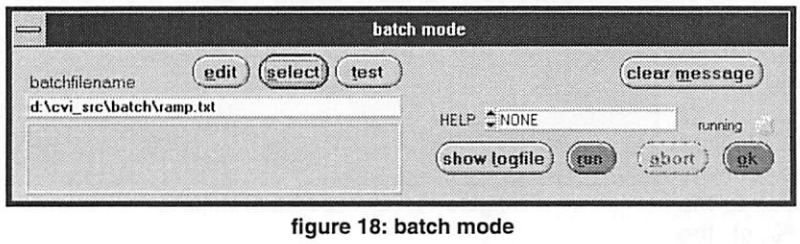

figure 18: batch mode

The *test* key calls a syntax text. Abort conditions of loops cannot be checked on their logical correctness. An error results in immediate abort of the action and display of the reason of the error.
The outputs in the batch mode are protocoled in a protocol file. This file is displayed by the *show logfile* key. The *run* key starts the batch run indicated in the file. If a ramp control is implemented the measured values are displayed in the graph. If an averaging is implemented the current measurement values and the average data are displayed. The *abort* key interrupts and terminates the active batch run immediately.

By the *help* key you receive a list containing the description of all commands for ramp control in the batch mode. The *ok* key leaves the window.
To have a better overview during long batch runs it is possible to delete the content of the output window for messages with the *clear message* key.

7.5.10 The graphics menu

Apart from the configuration of graphics display options and functions for the display of more than one measurement results the graphics menu contains also all functions for adjusting the graphics display represented as buttons in the main window.

7.5.10.1 Graphics options

The user can choose between automatic and manual scaling for the display in the main graphics. In the case of automatic scaling the abscissa is always scaled to the range of measured values, the ordinate is defined by the limit values of the ramp. During measurement a new scaling is necessary when the measured values leave the display area. If a segment is to be displayed only the user can define it, even during a measurement run. By the *set bounds to axis range* - key the limits of the abscissa and the ordinate are taken over. Hence an interesting area can be selected for the next measurement. Input is accepted only if the autoscale function is deactivated.
By the keys *ramp-min* and *ramp-max* the parameters set in the window for ramp parameters are accepted. In the field *units* texts for labeling the main graph can be entered.
The *cancel* key leaves the window without taking over the parameters. By the *ok* key the parameters are taken over when the window is left and the graphics display is adjusted.

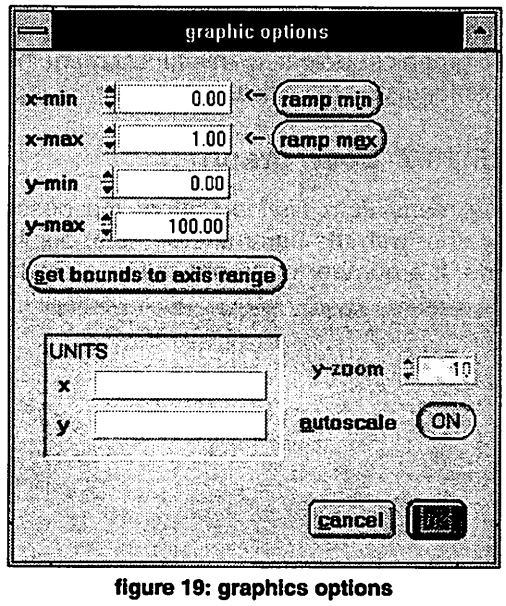

figure 19: graphics options

7.5.10.2 Freeze and release

No further update of the graphics display during measurement is implemented after activating the *freeze* function. I. e. all further measurement values are captured but not yet displayed. Here the zoom function can be implemented reasonably, as otherwise, with active *autoscale*, the graphics would be rescaled after each measurement value and the zoom area would be lost. The labeling changes to *release*. A repeated call deactivates the function. The graph is updated again. If measurement is terminated while the freeze function is active the function is terminated and the graph is put out completely. If there is no active measurement an activation has no effect.
This function exists also in the main window as command key.

7.5.10.3 Restore

The *restore* function triggers restructuring of the graph. This is necessary e. g. in order to get back to the original display after several zoom procedures. Switchovers of the autoscale function are not taken back.
This function exists also in the main window as command key.

7.5.10.4 Zoom in

The area framed by the two line cursors in the main graph is zoomed to the entire graphics display area.
This function also exists in the main window as command key.

7.5.10.5 Zoom out

The last *zoom-in* step is undone. Then the cursors frame again the area displayed before and, after a new *zoom in* the same area is displayed again.
This function exists also as command function in the same window.

7.5.10.6 Toggle main

This function enlarges the main graphics to nearly the entire area of the main window. Only the command keys on the left side of the main window remain, all other elements are covered by the graphics. Selecting this function again reduces the graphics to the original size.
This function also exists in the main window as command key (*maximize* button).

7.5.10.7 Load graph

The menu item *file/load* offers the possibility to load already taken measured values again together with the parameters and to display the graph. This can be done with only one file at a time. If a number of graphs is to be displayed together this is possible with this function. Up to six graphs can be displayed at the same time, the present measurement and five additional loaded graphs.

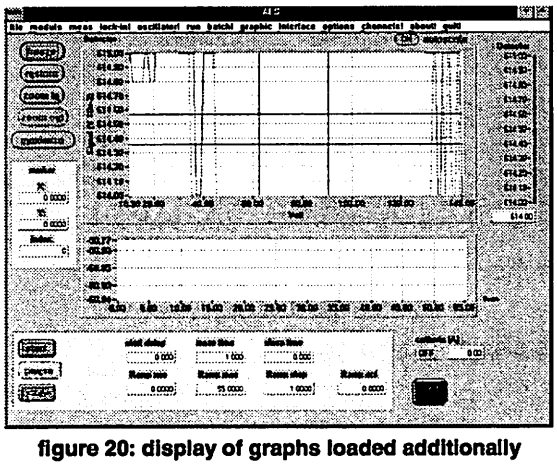

figure 20: display of graphs loaded additionally

A file is selected by direct input or a selection box. Of the file only the data of the channel selected as output channel for the main graph is loaded (see 7.5.13 The channels menu). After the graphics display the parameters for line color, kind of dash, factor and offset can be altered in an input window. The measured values are followed first by the offset addition, then the multiplication by the factor. This input window can be activated also by the snap cursor placed on the desired graph together with the right mouse key. The function for altering the parameters is available only for graphs loaded with *graphic load.*

The graphs loaded additionally cannot be stored. A change of the output channel, a restart of the measurement or the loading of a new configuration results in all graphs being deleted.

7.5.10.8 Delete Graph

To delete a single graph it must be marked with the snap cursor first. Selecting the menu item *delete graph* deletes this graph.

7.5.10.9 Delete all

This item deletes all graphs in the main and secondary graphics window.

7.5.11 The interface menu

This menu shows the interface drivers available at present. At the moment the drivers for the control unit, the lock-in and the A/D-card are implemented. In addition there is the command *test unit.*

The unit and the lock-in can be addressed only by the RS232 interface. The setting of the parameters has already been described in *The interfaces* page 10

7.5.11.1 A/D-card

PCI and ISA cards are supported by RFA-PC software. In case of NT based operation systems PCI cards are recommended.
No additional drivers are necessary for ISA cards. Settings are to be taken from the handbook of the card. The address of the card has to be set-up in the menu first. This address is stored in the ini-file and will be read upon each restart. After a change of address or a reboot read-access is tried. In case of an error further access will be denied. In case no card is installed select None in the menu. Opening an ini-file does not influence the address of the card.

Necessary drivers for PCI cards are supplied by the manufacturer coming with the card. Please refer to the manufacturers installation menu and select PCI in the menu. In case the mentioned driver is not loaded yet and the card is trying to be addressed the driver will be loaded and access will be tried automatically. In case an error occurs (no driver, no card) further access will be denied. In case no card is installed select None in the menu.

A memory access to the card, i.e. recording during a measurement, takes place only if the corresponding channels are activated (see *7.5.13 The channels menu*).

7.5.12 The options menu

7.5.12.1 System

After selecting the menu item *System* a window is opened for the entry of a password. To leave this window again at least one character must be entered. If the password has been correct a menu for the configuration of debug options is opened. These are not necessary for normal operation and slow down the execution speed considerably. Thus they are protected against accidental switch-on.

7.5.12.2 Unit mode

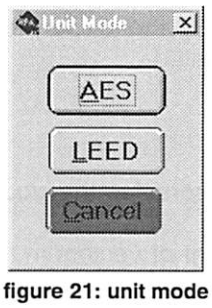

figure 21: unit mode

Selecting the menu item *unit mode* opens a window for the selection of the unit mode.
After the selection of one of the two modules the command is sent to switch over to the unit
The *cancel* - key closes the window without executing any action.

7.5.13 The channels menu

The *channels - menu* opens an input window to set the parameters for the measurement channels. Here is defined which channel is taken for measurement and which channel is displayed in the main graph. Furthermore the parameters of the A/D-converter, if integrated in the system, can be set.
There is a maximum number of six measurement channels. The *detector* is the measurement result of the lock-in amplifier, the *beam-current* is the measured beam current of the unit and four channels A/D-converter card. These four are deactivated if an A/D-card is integrated.

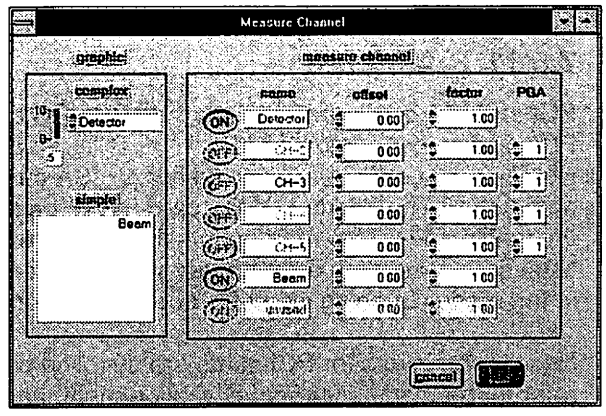

figure 22: adjustment of measurement channels

A name is to be assigned to each measurement channel and appears in the main window as labeling of the graphs. Each measurement channel can be assigned a factor and an offset. The result is generated by the formula:

*result = (measurement value + offset) * factor*

The parameter *PGA* indicates the amplification factor of the A/D-card for each of the four A/D-channels. It can have a maximum value of sixteen.
By the configuration element *complex* the channel is defined that is visible in the main graph. All the other active channels are displayed in the secondary graph. The display is switch-selectable also during measurement.
Every channel can be activated or deactivated for measurement. The channel for the main graph must be active always.
In the field *simple* the names of the channels are put out that are active and not defined as main channel.
Default for the measurement data is the measurement result of the lock-in being displayed in the main graph, the beam current in the secondary graph, No ramp can be executed if the interface to the lock-in is not activated but the lock-in chosen for measurement. To deactivate the lock-in as measurement channel an other channel has to be selected first for the display *complex*. Then the channel for the lock-in is to be deactivated .
If there is no A/D-card in the system or it has not been found under the indicated address the channels 2 to 5 are deactivated.
The *cancel* key leaves the window without taking over the parameters. By the *ok* key the window is left, the parameters are taken over and the graphics display is adjusted

7.5.14 The cursors

Three cursors are defined in the main graphics. There are two line cursors for the zoom function.
The cross-shaped cursor, also called *snpcursor,* is used for the measurement of vectors in the main window. It is always bound to a measurement vector and always located on a test point, never on the connection line between two test vectors. The field X and Y display the physical values of the marked test point. The field *index* displays the running number of the test point (rising from zero).
If additional vectors have been loaded with the function *Load Graphic* these can be measured also with the *snapcursor.*
The cursors can be operated with the mouse or the keyboard.
There are no cursors in the secondary graph. If a data set from the secondary graph is to be measured it must be transferred to the main graph by the channels menu.

Chapter

8

Batch programming

8 Batch programming

Batch routines serve to automate the execution of ramp control in connection with the averaging. The user can control the ramp by segments, jump uninteresting parts and implement an averaging with arbitrary abort conditions.

8.1 Implementing a batch run

First the batch program is to be set up. This program must be filed in an ASCII-file. The menu item „batch!" offers the standard editor of Windows *NOTEPAD.EXE.* The selection of the menu item "batch!" opens a window where the name of the file is to be entered by the user. The *select* key opens a selection box. A test run for a syntax check can be started by the *test* key. The result of this test is stored in the file "batch.log". After the selection of *run* the interpreter starts interpreting the file. Mistakes lead to an abort of the batch run. Outputs in the batch program (with the write-command) are stored in the file "batch.log".

8.2 General information

Default for the batch run is a file generated by the user describing the execution with the commands listed below. This file can be generated with any ASCII-editor. The batch window offers the possibility of starting an editor directly. In general this is the editor *notepad.exe* included in the Windows 3.1 delivery set.
A batch instruction must not be longer than a line, otherwise this is considered a syntax error. There is no differentiation between capitals and small letters.

8.3 Comment characters

A comment is started with the character „*" and ends with the end of the line. The comment character can have any place in a line.

8.4 Variables

The user cannot generate variables. There is a default for two fields, each containing 100 elements, a field for integer and a field for float variables. Referencing is implemented by the syntax *float[x]* or *int[x].* There is an additional default of 10 integer and 10 float variables

referenced by *int_x, float_x.* Indexing starts at 0. Double indexing of fields (e. g. *int[int[3]])* is not permitted, indexing by the control variables, however, is permitted. When addressing a field no blank is allowed between the field type and the index. There is also a field for strings with 100 elements of the lengh of 80.
example:

```
int[0]          * correct writing
int [0]         * faulty, as separated by blanks
int[int[1]]     * faulty, as double indexing
int[int_1]      * correct writing
```

8.5 Defines

Simple defines are permitted. These are treated as textual replacements. It must be possible to resolve them in a single step. Nesting defines is not permitted. Define instructions can have any place in the batch file and have an effect on the batchrun from that place onwards. Be careful in the case of defines in loops. In defines also, there is no differentiation between capitals and small letters. The defines *LAUF, Lauf,lauF* always reference the same variable.
example:

```
define MAX int[0]     * definition of the first element of the
                      * integer field as MAX
define LAUF  int_3    * definition of the first integer variable as
                      * LAUF
```

8.6 Operators

Only simple, no nested expressions are permitted for the logical conditions such as an if-inquiry. The operands permitted in the inquiries are integer or floats. Only the type integer was used in the examples. Strings cannot be taken for comparison.

8.6.1 Equality: ==

Check on exact equality.
example:

```
if int[3] == int[2]
  set  int[2]  50
endif
```

Check of two variables. Be careful with this command when using float variables.

8.6.1.1 Inequality: !=

Check on inequality.
example:

```
if int[3] != int[2]
  set  int[2]  50
endif
```

8.6.1.2 Greater than: >

Comparison of the magnitude of the operands.
example:

```
if int[3] > int[2]
  set  int[2]  50
endif
```

8.6.1.3 Greater-or-equal: >=

Comparison of the magnitude of the operands.
example:

```
if int[3] >= int[2]
  set  int[2]  50
endif
```

8.6.1.4 Less than: <

Comparison of magnitude of the operands
example:

```
if int[3] < int[2]
  set  int[2]  50
endif
```

8.6.1.5 Less-or-equal: <=

Comparison of magnitude of the operands
example:

```
if int[3] <= int[2]
  set  int[2]  50
endif
```

8.7 General batch commands

A command consists of a command term followed by one or more variables and/or constants, or by a condition. If the command is a function that supplies a return value e. g. a multiplication, the command term must be followed directly by a variable. This variable contains the result after the execution. After that the operands follow.
In case of all commands to manipulate figures (set, mul, etc) field elements of the type int or float can be but in. Optional parts are indicated in braces. Regarding arithmetical operations integer and floats are permitted as operands. The operands and the result variable do not need to be of the same type.

8.7.1 Addition

Addition of two numbers.
Syntax:

```
add value value/const value/const
```

Variable type : int, float
Value validity range: --
example:

```
add int[1] int[2] int[3]
```

Addition of the two operands *int[2]* and *int[3]*, Filing of the result in *int[1]*. The format of the result variable determines whether the operation is based on integer or float.

8.7.2 Delay

Waiting time
Syntax:

```
delay value
```

Value type: float
Time in seconds by which the execution of the next command is delayed.

8.7.3 Division

Division of two numbers.
Syntax:

```
div value value/const value/const
```

Value type : int, float
Value validity range: --
example:

```
div int[1] int[2] int[3]
```

Division of *int[2]* with *int[3]*, filing of the result in *int[1]*. The format of the result variable determines whether the operation is based on float or integer. The batch interpreter recognizes a zero, reports the error and stops the batch run.

8.7.4 if/else/endif

Inquiry of a condition. If the result is TRUE (!= 0) the instruction block following the condition is executed, otherwise the *else* instruction block is carried out , or the instructions after *endif* are carried out.
Syntax:

```
if condition
    instruction block
{else
    instruction block}
endif
```

An *if* statement does not need to contain an *else*-branch. The block is always terminated by *endif*.

8.7.5 main

Definition of the main program
Syntax:

```
main
 instruction block
endmain
```

The main program is framed by the two keywords *main* and *endmain.* All lines in the file following the keyword *endmain* are ignored.

8.7.6 Multiplication

Multiplication of two numbers
Syntax:

mul *value value/const value/const*

Value tpe: int, float
Value validity range --
example:

```
mul int[1] int[2] int[3]
```

Multiplication of the two operands int[2] and int[3], filing of the result in int[1]. The format of the result variable determines whether the operation is based on float or integer.

8.7.7 set (String)

Assignment of a string to a variable.
Syntax:

```
setstring value value/const
```

Value type : string array element
Value validity range: --
example:

```
setstring string[1]  "Hallo"      * Output of the content of
                                  * string[1] and float[2]
```

Assignment of the constant string "Hallo" to the stringfield 1. The string must be limited by double quotes.

8.7.8 set (numbers)

Assignment of a value to a variable.
Syntax:

```
set value value/const
```

Value type:int, float
Value validity range: --
example:

```
set int[1] int[2]
```

Assignment of the content of the field *int[2]* into the field *int[1].* the second operand can be a constant, too.

8.7.9 subroutine

8.7.9.1 Definition of a subroutine

Syntax:

```
subroutine name
 instruction block
endsub
```

This command defines a subroutine. It must be defined at the beginning of the batch part, in the initialization part before the *main* / endmain - block. Delivery parameters are not permitted. The maximum number of subroutines per program is 100.

8.7.9.2 Call of a subroutine

Syntax:

```
subroutine name
```

The program branches into the subroutine *name* generated by the user. Reaching the keyword *endsub* at the end of the subroutine results in the program line being carried on following the call of the subroutine. There is a maximum of 10 subroutines permitted.

8.7.10 Subtraction

Subtraction of two numbers.
Syntax:

```
sub value value/const value/const
```

Value type: Integer, float
Value validity range: --
example:

```
sub int[1] int[2] int[3]
```

Subtraction of the content of field int[3] from field int[2], Filing of the result in field int[1]. The format of the result variable determines whether the operation is based on integer of float.

8.7.11 while/endwhile

The instruction block will be performed as often until the condition is not TRUE(!= 0) anymore (rejecting loop).
Syntax:

```
while condition
    instruction block
endwhile
```

If the condition is WRONG from the beginning the instruction block in the loop will not be executed at all.

8.7.12 write,writeln

Data output in the log-file.
example:

```
  write string[1] int[2]          * output of the content of
                                  * string[1] and int[2]
  write "Hallo" float[2]          * output of the string „Hallo" and
                                  * of the content float[2]]
  writeln string[1] int[2]        * output of the content of
                                  * string[1] and int[2] and
                                  * of a line feed
```

Output of the content of strings and values into the log-file. The third instruction differs from the first two because it contains an output of a carriage return, too, that results in a new line (line feed and carriage return). It serves to generate a column format in the output file. If *ZERO* is put in for the number variable this element will not be put out.

8.8 Program example

```
* Program example of 25.08.95
* Author Jürgen Leißner

*
* Definition of variable names
*
define MAXFELD 100
define MIN int[1]
define MAX int[2]
define ERG int[0]
define run int_0
*
* Subroutine for defining Min and Max
*
subroutine minimum
  if MIN > MAX
    set erg max
    set max min
    set min erg
  endif
  write string[2] max
  writeln string[1] min
endsub

*
* begin of the main part
```

```
*

main
   set min 20
   set max 10
   setstring string[2] „Maximum"
   setstring string[1] „Minimum"

   subroutine minimum
   mul erg min 20
   writeln „result MUL (int):" erg
   mul float[0] min 20
   writeln „result MUL (float):" float[0]

   *
   *  field operation (integer)
   *
   set run 0
   writeln „field number  content" NULL
   while run < MAXFELD
       set int[lauf] run
       mul int[lauf] int[lauf] run
       write "" lauf
       writeln "" int[lauf]
       add run run 1
   endwhile
   *
endmain     * program end
```

8.9 Basics of ramp control

An important feature of batch programming in the operating mode AES is the possibility of segmented ramp control. From the surface only one ramp can be defined, that is executed continuously between the two limit values indicated. Possibly there is a large area to be covered containing only a few interesting segments. With batch programming you can measure this interesting segment and jump the other areas. There is a maximum number of 10 segments possible. The individual segments are put together in a single measuring set and displayed together in a graph. The limits of this measuring set and also the scaling of the abscissa of the graph are defined by the lowest and highest voltage existing. An averaging is also possible. To this end embed the part that executes the measurement in a loop while programming. The number of runs, abort criteria etc. can be determined by the user. The measurement data of every single run and the average data can be stored in a file. During measurement there is a

graphics display of the current measurement values and the average data. After every run the average data is recalculated and the graph is updated. After completion of a measuring series an optical data comparison is possible with the options loading of more than one graph described in *Load graph* on page *22*.
Segment definition and measurement can be implemented several times in a batch file. From the batch file all module voltages can be controlled and re-read. This offers the possibility of executing several measurements under different conditions.

8.9.1 Ramp control

Please note the following conditions for implementing a ramp run:
1. The limits of the segments must be defined before ramp execution. A maximum number of 10 segments is possible.
2. The segments must be in ascending sort .
3. The command *rampdefineend* sets the average counter at zero.
4. Calling *rampexecute* increments the average counter, executes the ramp in the defined segments and displays the current measurement values. The present measurement values are added to the existing values, weighted and displayed with the average counter.
5. If more than one segment definition is implemented in a batch run a new average cycle begins after each *rampdefineend.*
6. To execute a number of measurement runs set the ramp command in a loop.
7. The current measurement values as well as the average data can be saved in file.

8.9.2 Commands for ramp control

Apart from the commands listed before there is a set of commands needed especially for ramp control.

8.9.2.1 getvoltage

Read-out of a voltage value
Syntax:

```
getvoltage value string1 string2
```

Value type: Float
Value validity range: --
Unit: volt
The current voltage value named in string 1 is read out. Besides the voltage name also the type (value, gain, offset) must be defined in string 2. The labeling of the voltage must correspond to the one in the window „AES-modules".

8.9.2.2 measure

Triggers a measurement
Syntax:

```
measure
```

This command triggers a measurement with the present configuration. The command „getmeasvalue" reads out the measurement result. It serves, e. g. , for defining the starting condition for optimization.

8.9.2.3 rampexecute

Execution of a defined ramp

Syntax:

```
rampexecute
```

Implementation of ramp control within the set limits with summation and weighting of the values.

8.9.2.4 resultfile

Triggers the storing of data in file
Syntax:

```
resultfile string1 string2
```

Value type: string
Validity range: --
The average data is saved under the indicated file name (string1). The average data equals the sum of the measurement data divided by the number of average runs. An indicated extension is used for the file name. (compare intermediate storage of files). The second string is stored as comment within the file.

8.9.2.5 savefile

Triggers intermediate storage of data in file.
Syntax:

```
savefile string
```

Value type: string
Value validity range: --
Saving of the present measurement values under the indicated name. This data corresponds to the data taken from a simple ramp run. There is no indicated extension used for the file name. By standard an extension is started with „.000". After every storing the extension is increased by one. A maximum number of 20 different files can be used in a program. The command *setextension* can be used to start with an other extension.

8.9.2.6 setdir

Selection of directory for data filing.
Syntax:

```
setdir string
```

Value type: string
Value validity range: --
Default of the directory in which the files are stored. The name of the directory must end with a backlash.

8.9.2.7 setextension

Extension default for intermediate storage
Syntax:

```
setextension value
```

Value type: Integer
Value validity range: 0 .. 999
Extension default for the files of intermediate storage used in the first run. The number is increased by one after every run. If the number 999 is surpassed this is considered an error and the batch run is aborted.

8.9.2.8 setmeaschanel

Selection of the channel whose measurement value is to be taken for the averaging.
Syntax:

```
setmeaschanel value
```

Value type: Integer
Value validity range: 1..6
Number of the measurement channel taken for the averaging. The number must be between 1 and 6. The measurement channel is activated by this command. The interface cannot be opened here.

8.9.2.9 setmeastime

Setting of measurement time
Syntax:

```
setmeastime value
```

Value type: Float
Value validity range: 0.001 .. 100.0
Unit: seconds
If a lock-in amplifier is used its integration time cannot be altered this way. This is a delay.

8.9.2.10 setrampmax

Upper limit of a segment
Syntax:

```
setrampmax value
```

Value type: Float
Value validity range: -20.0 .. 2000.0
Unit: volt
Upper limit value of a ramp segment and thus last measurement point. It must be always greater than the minimum. If the minimum-maximum range cannot be devided integrally by the increment this value is taken additionally in the measurement series.

8.9.2.11 setrampmin

Lower limit value of a segment
Syntax:

```
setrampmin value
```

Value type: float
Value validity range: -20.0 .. 2000.0
Unit: volt

Lower limit value of a segment and thus first measurement point. In case it is not the first segment the limits may not overlap.

8.9.2.12 setrampstartdelay

Setting of a waiting time before a new measurement.
Syntax:

```
setrampstartdelay value
```

Value type: float
Value validity range: 0.001 .. 100.0
Unit: seconds
Waiting time of the system after the jump of the ramp voltage to a new segment. The start of a new measurement run is also counted as jump to a new segment. The time is started after the setting of the first ramp value of a segment. With this parameter the system can build up transient after a jump to the lower segment limit of the next element. Within the segment a delay can also be set by the parameter *setsleeptime*. Generally, however, this is considerably smaller as also the increment in a segment is considerably smaller than the jumps.

8.9.2.13 setrampstep

Increment for voltage variation of the ramp.
Syntax:

```
setrampstep value
```

Value type: float
Value validity range: 0.0 .. 2000.0
Unit: volt
Increment the defined segments are passed with. The increment must be defined before the segments.

8.9.2.14 setsleeptime

Sets the waiting time.
Syntax:

```
setsleeptime value
```

Value type: float
Value validity range: 0.001 .. 100.0
Unit: seconds
After setting the new voltage value the measurement is started only after the defined time. The parameter *setrampstartdelay* should be used for defining a delay after a jump to a new segment.

8.9.2.15 setvoltage

Sets a voltage value.
Syntax:

```
setvoltage string string value
```

Value type: float, constant
Value validity range: --
Unit: volt
The voltage indicated in the string is set at the defined value. Apart from the voltage name also the type (value, gain, offset) must be indicated. The voltage name must correspond to the one in the window „AES-modules".

8.9.3 Program examples

Both of the following program examples are included in the scope of supply and are stored in the directory the program has been installed in. Example 1 shows an averaging procedure with preset number of runs. Example 2 shows how module voltages can be manipulated from the batch.

8.9.3.1 example 1

```
* author: Jörg Scholtes
* state: 29.01.96      draw up: 29.01.96
* file : spec1.txt
*
* function:
* this file is for multiple integration of a quick measurement
* two segments are measured
*
*
* definition part
define runint_1
define AVERAGE 10

main
* definition of the ramp
  setrampstartdelay 10                  * waiting time after a
                                        * segment jump
                                        * in this example with 150V
                                        * and 250V ramp voltage
  setmeastime 0.12                      * measurement time per point
  setrampstep 0.5                       * increment for the ramp

  setextension 100                      * start extension for the
                                        * backup files
  setdir "c:\tmp\"                      * directory in which the
                                        * files are stored

                                        * definition of the ramp
  rampdefinestart
    setrampmin 150                      * elastic peak
    setrampmax 190

    setrampmin 250
    setrampmax 300
  rampdefineend                         * end of ramp definition
```

```
  set run 0                                  * default of the run variable
  while run < AVERAGE                        * execution of the ramp
    rampexecute                              * data determination, pass of
                                             * the segments defined above
    savefile "tmp"                           * intermediate storage
    resultfile "result2.dat" "testfile"      * result file with comment
    add run run 1
  endwhile

endmain

* ****** program end******
```

8.9.3.2 Example 2

```
  * author: Jürgen Leißner
  * state: 07.02.96      draw up: 07.02.96
  * file : setvolt.txt
  *
  * function
  * this file shows the possibilities for setting and return reading
  * of voltages
  **

  * definition part
  define value float_1
  define default float_2
  define name string[1] * variable for the voltage name
  define type string[2] * variable for the voltage type

  * program end
  main

  * ** energy read-out
  * direct indication of the parameters
  getvoltage value "energy" "value"* value read-out
  if value < 100
    setvoltage "energy" "value" 100* value setting
  endif
```

```
* ** anode read-out
getvoltage value "anode" "offset"      * value read-out
if value < 50
  setvoltage "anode" "offset" 50       * value read-out
endif
   * transfer of the parameters as variables
setstring name "anode"
setstring type "gain"
getvoltage value name type             * value read-out
write name ZERO                        * data output in the log file
writeln type value                     * all data in a line

* ** collector read-out
setstring name "collector"
setstring type "value"
getvoltage value name type             * value read-out
if value < 30.8
  set default 32.5
endif
if value > 35
  set default 33
endif
setvoltage name type default           * value setting
write name NULL                        * data output in the log file
writeln type value                     * all data in a line

endmain

* ****** program end ******
```

8.10 List of variable voltages

name	type
"energy"	value
"wehnelt"	gain, offset
"collector"	value
"anode"	gain, offset
"L13"	offset
"L2"	offset
"ramp"	value