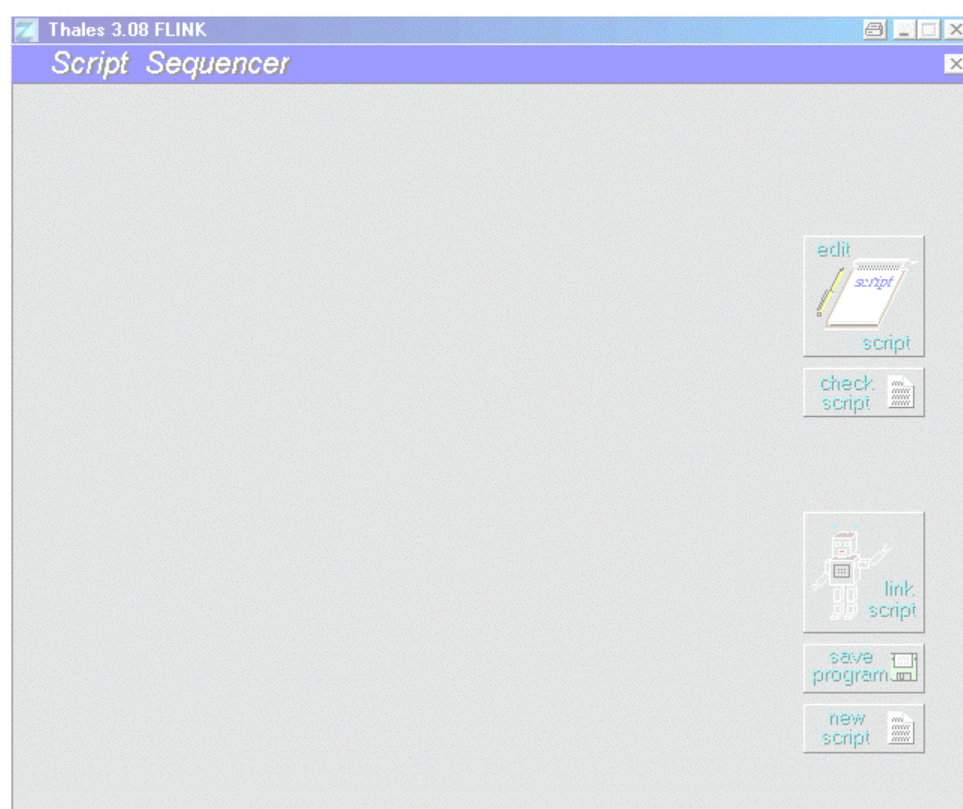


SCRIPT

Combination and Automation of Electrochemical Techniques and Measurement Data Analysis with Script

Practical Course 6 • Dr. Burkhard Röseler



Kronach Impedance Days 2010

Introduction

If an electrochemical workstation is used for different applications, a good portion of discipline is necessary to guarantee reproducibility for repeating measurements. In case of measurements consisting of several parts, which require manual settings, a strict reproducibility in time is not possible.

In the same way, repeating routine-jobs, e.g. consisting of cycles of measurement – saving – documentation, are a challenge regarding discipline and patience to the user, if manually triggered.

Of course, special programs, tailored to a special application, can solve the problems of reproducibility and automation. This possibility is only practicable if the problem is worth the deal of programming. A high starting energy to get acquainted with a programming tool is not acceptable. However, smaller jobs such as the chaining of different measurement and conditioning phases to one automatically running application can be handled easily. The SCRIPT feature of Thales is optimized to such applications, with the focus on a good scalability: With the extended programming language you are able to write even complex procedures.

Realization

In a first step, an electrochemical measurement job shall be run under reproducible conditions. The example deals with the recording of an impedance spectrum. For that purpose, a “model measurement” is set up which acts as a model for the automatic procedure. Due to this “model” function we can do without a time-consuming parameter set up (input of all conditions in the script procedure). Instancing the consideration of the momentary rest potential of the measuring object the second step shows how to adapt the procedure to changing conditions automatically.

The third step shows how to save the measurement data automatically to a pre-defined path. In a forth step we implement analysis functions of Thales into the script. For that purpose, the recent EIS measurement data are loaded under script control into the analysis program SIM. Then they are fitted to a previously created model. The automated analysis is completed by the saving of the fitting results as an ASCII formatted file to a pre-defined path.

Step five shows how to implement the graphics created during the script procedure into a document and how to print the document, copy it to the Windows clip-board and save it to the HD.

The experiment will take about 40 minutes. After that a discussion is planned and questions can be asked. If time will be remaining, additional topics can be demonstrated:

1. How to react on external conditions beyond the electrochemical environment (e.g. temperature)
2. How to process data and how to display them graphically

Step-by-step created SCRIPT source file

SCRIPT programming level 1

SCRIPT1

```

'***** Step 1 *****
MEAS_OPEN_EIS(65,"c:\thales\script\kit\rules","eis_rule")

'***** Step 2 *****
GETPARAM
POTENTIAL
Pset=Pact+0.1 'set potential 100mV more anodic than rest potential
Pot=-1       'potentiostat mode ON
SETUPECW     'update ECW
MEAS_EIS
Pot=0        'potentiostat mode OFF
SETUPECW     'update ECW

'***** Step 3 *****
FILE_NAME$=dt$+ti$
MEAS_SAVE_EIS(65,"c:\thales\script\kit\data",FILE_NAME$)

'***** Step 4 *****
CLRSIM                                     'initialize SIM
ANA_OPEN_EIS(65,"c:\thales\script\kit\data",FILE_NAME$) 'load meas file
ANA_OPEN_MOD(65,"c:\thales\script\kit\rules","fit_rule") 'load model
FITEIS(10,10000,25)                        'Fit, 25 smooth-samples

'***** Step 5 *****
ANA_PLOT_MOD(0)                            'display last curve
ANA_XASCII(65,"c:\thales\script\kit\data",FILE_NAME$)
CAD_OPEN(65,"c:\thales\script\kit\rules","cad_form") 'load document template
ANA_XGRAPH(0,0)                             'copy recent graphic to SIM at position x=0,y=0
CAD_DOC(-1)                                  'copy document to clip board (1=print)

SCRIPT_END
    
```

Usage of ANDI-BASIC variables

SCRIPT programming level 2

By usage of variables, the source will become more transparent and may be modified more easily. E.g. to change the data paths of rules and data we only have to modify two text variables instead of six text statements. Finally, different rules can easily be chosen by the usage of text variables.

SCRIPT1

```

RULE_PATH$="c:\thales\script\kit\rules"
DATA_PATH$="c:\thales\script\kit\data"
EIS_RULE$="eis_rule"
FIT_RULE$="fit_rule"
CAD_FORM$="cad_form"

MEAS_OPEN_EIS(65,RULE_PATH$,EIS_RULE$)
.
MEAS_SAVE_EIS(65,DATA_PATH$,FILE_NAME$)
.
ANA_OPEN_EIS(65,DATA_PATH$,FILE_NAME$)           'load meas file
ANA_OPEN_MOD(65,RULE_PATH$,FIT_RULE$)           'load model
.
ANA_XASCII(65,DATA_PATH$,FILE_NAME$)
.
CAD_OPEN(65,RULE_PATH$,CAD_FORM$)               'load document template
.
    
```

SCRIPT_END

Trigger by an external measurand (e.g. temperature)

To obtain a well structured source we can use subroutines or functions.

SCRIPT1

```

gosub TRIGGER
.
.
goto SC_END

TRIGGER::
    TRIGGER_TEMP=30           ' set treshhold equal 30C
    TRIG%=0                  ' reset trigger
    repeat                   ' start loop and repeat
        ACQUIRE(0)         ' read temperature
        TRIG%=TRIG%or(ACQ_VAL>=TRIGGER_TEMP) ' set trigger if temp higher than threshold
    until TRIG%              ' until final condition is reached
return                       ' return from subroutine

SC_END::

SCRIPT_END
    
```

The triggering example shown above can cause problems. What happens if the final condition will not be reached? The programm will hang up in a 'never ending loop'. The final version of the subroutine trigger (page 8) will demonstrate a possible exit e.g. by a key strike. Message boxes and script subroutines will be used to give as much information as possible to the user.

Usage of the subroutines 'START' and 'SUBn'

When you enter the programme 'script' you will end up in the main menu. The programme runs into a polling routine waiting for a key strike or for mouse action. Within the polling routine different subroutines (SUB1, SUB2, ..., SUB9) can be called. Before entering the polling routine the programme tries to call the subroutine named 'START'. Usually, the mentioned subroutines are not present in the script runtime file and calling errors will be reset by an exception handling.

```
.
MENU::
  gosubSTART
  gosubPICTMENU
  gosubPOLLING
.
POLLING::
  .
  getKEY$
  onSUB%gosubSUB1, SUB2, SUB3, SUB4, SUB5, SUB6, SUB7, SUB8, SUB9
  .
return
.
```

By definition of one of the subroutines within the script source code script may be forced to certain actions. E.g.:

```
START::
  DISPLAY(200,316, "Temperature / C", "NiCr/Ni")
  SUB%=1
return
```

This subroutine will open a display within the main menu's screen. Finally, the variable SUB% will be set equal one. Thus the polling routine will call the subroutine SUB1:: . In SUB1 finally the temperature will be measured and displayed in the instrument. This technique opens the possibility to measure and display data with the electrochemical workstation even before the script application has been started. In the following listing you will find the subroutines START:: and SUB1:: within the triggering script.

Source file for step 8 (optional)

user defined graphics

```
SCRIPT1
  fori%=0to24
    XYZ(i%,1)=sin(i%/10)+cos(i%/3)*.3
    XYZ(I%,0)=i%*1e-5
  nexti%
  PLOTTYPE(0.5,2.3,64,.3,2,0,2)
  PLOTFRAME(100,100,320,256,15,1,0,48,14,1)
  PLOTXYZ(0,10,4,2,0,3)
  PLOTAXIS(0,3,1,15)
  PLOTAXIS(1,2,1,15)
  PLOTLABEL(2,3,"Time / ", "s", 2,1)
  PLOTLABEL(3,2,"Current / ", "A", 3,1)
  getcount(1) a$
SCRIPT_END
```

Triggering an impedance measurement by a file

The IM6 is well suited to measure the properties of an electrochemical object however, it offers only few possibilities to control a complicated electrochemical set up. Considering a set up where external parameters must be set and controlled (e.g. partial pressures of gases, the temperature of the cell, the pressure of an object, etc., etc.) the IM6 may be included into that set up for impedance measurement, cyclic voltammetry, I/E curves, etc. A most elegant communication will be obtained by exchange files. When the controlling process will reach stability an exchange file will be written to the PC hard drive. Before that file will be received by the IM6 the IM6 itself must be send into a polling routine. Every second the IM will try to open the exchange file. If the file can be opened the triggering condition will become true and the IM will e.g. start an impedance measurement.

```

SCRIPT1
  gosub TRIGGER_MESSAGE
  gosub TRIGGER
  on-EXIT% goto MENU
  ' gosub MEAS, FIT, DOC etc.
  goto MENU

TRIGGER::
  TRIG%=0
  EXIT%=0
  SC_TI=fnTI(0)+1
  FILE$="c:\thales\temp\start.txt"
  repeat
    gosub WAITFILE
    gosub KEYBOARD
  until (TRIG%orEXIT%)
  return

WAITFILE::
  TI=fnTI(0)
  if (TI>=SC_TI) trueif
    open1,65,1,FILE$+"r"
    if (st=0) trueif
      TRIG%=-1
      close1
      scratch d2,u65,FILE$,d2,u65
    endif
    SC_TI=SC_TI+1
  endif
  return

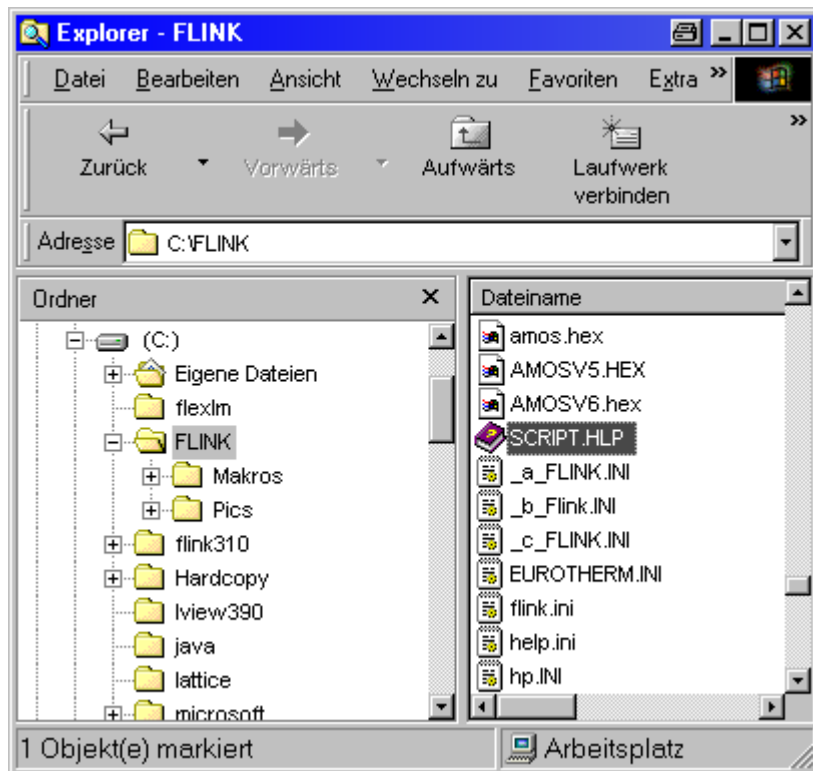
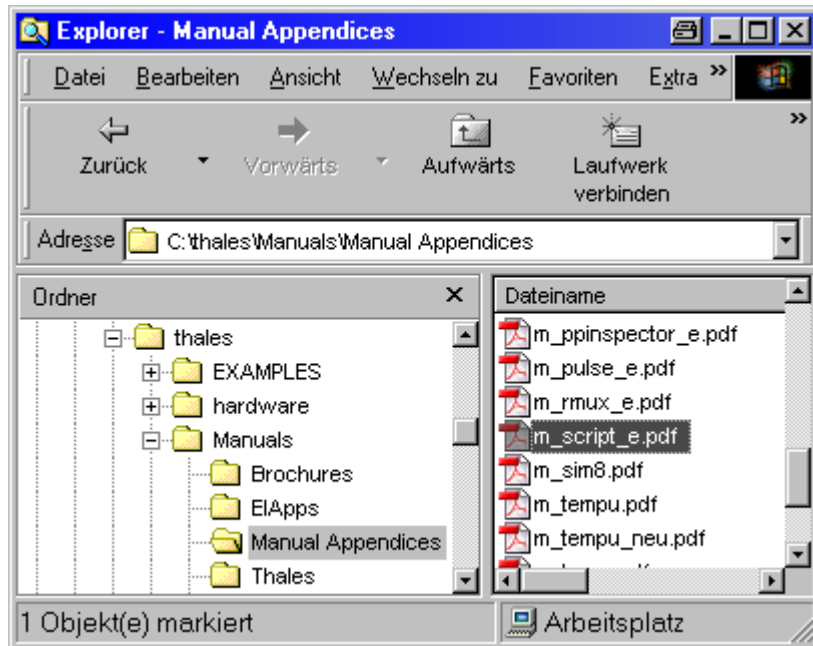
KEYBOARD::
  getKEY$
  EXIT%=EXIT%or(KEY$="e")
  return

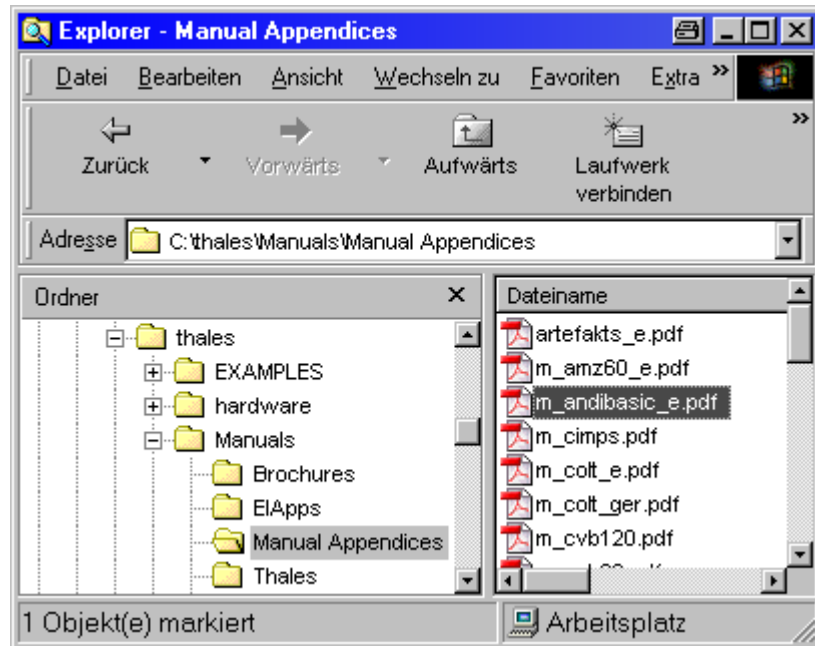
TRIGGER_MESSAGE::
  MESSAGE(0)
  MESSAGE$("Waiting for Trigger")
  MESSAGE$("File='start.txt'")
  MESSAGE$("")
  MESSAGE$("hit <e>-key to exit")
  MESSAGE(-1,24,20)
  return

SCRIPT_END
    
```

Help on script

Help on script programming can be found in different files of the Zahner THALES installation.









Finally, if no help will help and you're getting



Dipl.Phys.Univ.
Dr.rer.nat. **Burkhard Röseler**

Department Development
Software & Hardware
ZAHNER-elektrik GmbH&CoKG

Thüringer Str.12
D-96317 KRONACH

 +49-(0)9261-962119-18
 +49-(0)9261-962119-99
 br@zahner.de
 www.zahner.de

Final version of the script

SCRIPT1

```

RULE_PATH$="c:\thales\script\kit\rules"
DATA_PATH$="c:\thales\script\kit\data"
EIS_RULE$="eis_rule"
FIT_RULE$="fit_rule"
CAD_FORM$="cad_form"
TRIGGER_TEMP=30

gosub TRIGGER
on-EXIT%goto MENU

gosub MEAS_IMPEDANCE
gosub FIT_IMPEDANCE
gosub EVALUATION
goto MENU

MEAS_IMPEDANCE::
  MEAS_OPEN_EIS(65,RULE_PATH$,EIS_RULE$)
  GETPARAM
  POTENTIAL
  Pset=Pact+0.1 'set potential 100mV more anodic than rest potential
  Pot=-1       'potentiostat mode ON
  SETUPECW    'update ECW
  MEAS_EIS
  Pot=0       'potentiostat mode OFF
  SETUPECW    'update ECW
  FILE_NAME$=dt$+ti$
  MEAS_SAVE_EIS(65,DATA_PATH$,FILE_NAME$)
return

FIT_IMPEDANCE::
  CLRSIM                      'initialize SIM
  ANA_OPEN_EIS(65,DATA_PATH$,FILE_NAME$) 'load meas file
  ANA_OPEN_MOD(65,RULE_PATH$,FIT_RULE$)  'load model
  FITEIS(10,1000,25)           'Fit, 25 smooth-samples
return

EVALUATION::
  ANA_PLOT_MOD(0)              'display last curve
  ANA_XASCII(65,DATA_PATH$,FILE_NAME$)
  CAD_OPEN(65,RULE_PATH$,CAD_FORM$)    'load document template
  ANA_XGRAPH(0,0)              'copy recent graphic to SIM at position x=0,y=0
  CAD_DOC(-1)                  'copy document to clip board (1=print)
return

TRIGGER_MESSAGE::
  MESSAGE(0)
  MESSAGE$("Waiting for Trigger")
  MESSAGE$("Temp >= "+fnSTR$(TRIGGER_TEMP))
  MESSAGE$("")
  MESSAGE$("hit <e>-key to exit")
  MESSAGE(-1,24,20)
return

TRIGGER::
  TRIG%=0                      ' BASIC Flag    0=false, -1=true    reset Flag
  EXIT%=0                      ' BASIC Flag    0=false, -1=true    reset Flag
  repeat                        ' start loop and repeat until final condition is reached
    gosub READ_TEMP
    gosub KEYSTRIKE
    until (TRIG%orEXIT%)
return                          ' return from subroutine

```

```
READ_TEMP::
  TI=fnTI(0)                                ' read timer
  if(TI>=SC_TI) trueif                       ' if timer bigger than refresh time
    gosubGET_TEMP
    SC_TI=ti+.5                              ' increment refresh timer
    TRIG%=TRIG%or(ACQ_VAL>= TRIGGER_TEMP)    ' check trigger condition
  endif
  return

GET_TEMP::
  ACQUIRE(0)                                ' read temperature
  packa$,using"-###.#",ACQ_VAL              ' pack measured value to text variable
  DISPVAL(200,316,a$)                       ' and print into display on screen
  return

KEYSTRIKE::
  getKEY$                                    ' read keyboard
  EXIT%=EXIT%or(KEY$="e")                   ' exit if KEY was <e>
  return

START::
  DISPLAY(200,316,"Temperature / C","NiCr/Ni")
  SUB%=1
  return

SUB1::
  gosubGET_TEMP
  return

SCRIPT_END
```