Aqserver Documentation

Release 1a

Michael Taxis

March 30, 2017

The program is an acquisition tool for Siemens Simatic S7 PLCs. It was written using the python-snap7 module from https://pypi.python.org/pypi/python-snap7/ and the snap7 library from http://snap7.sourceforge.net . File handling is based on example source code for the book “Real-World Instrumentation with Python”.

Homepage http://www.taxis-instruments.de

Contact <aqserver at taxis-instruments dot de>

Author Michael Taxis <michael at taxis-instruments dot de>

CHAPTER 1

About

With this program you can record values from a Simatic S7 PLC to a CSV File. Depending on the number of values to record the scan time is changing, thus increasing with the number of values. It is possible to set trigger conditions, that will start a new file. The old file will be compressed and stored in a selectable directory tree. Using a pre- and post trigger time, you can set an overlap of old and new file. If you are not able to set a working trigger, the trigger can also be activated manually.

A config file must exist (default file is aqserver.cfg) that defines communication and other parameters and of course the S7 variables to scan. Open the config file in your preferred editor and read the comments on how to configure.

Values will be stored to a csv-file in subdirectory data. Filename can be specified in configfile. Once a configurable trigger is raised, a new datafile begins.

After trigger or when program is stopped, the data file will be compressed and stored to a path, that can also be specified in the config file.

When running several instances of the program, using different config files, values can be recorded from different PLCs at the same time.

Please note that this program is in an early stage of development. If you run into problems when using it, see the "Problems" section of this manual. Aqserver will not create a problem with your PLC program though.

Using Kst2 from https://kst-plot.kde.org the datafile can be read and plotted online during the recording. Also decompressed archives can be plotted to your liking.

An alternative would be LiveGraph from http://www.live-graph.org , but then the data files must be modified, because LiveGraph is expecting csv-files in a certain format (actually this is not provided by Aqserver at the moment).

CHAPTER 2

New in this version

28 March 2017

- added functionality for writing units for values to datafile
- added functionality for demo: creating data from ramdom function, without connection to plc. DEMO is shown in headline of aqserver
- added functionality for booloffset: boolean values within one byte are added an offset, so they can be shown in one plot without overlapping each other
- config values are now case sensitive. ATTENTION: config values IP, RACK and SLOT must be changed to lowercase in your old config files (ip, rack,slot). This was changed because the value names where all in lower case.

See also changelog for a complete history of changes.

Installation

Program can be obtained from http://www.taxis-instruments.de .

The program was written using Python 2.7.10 under Windows 7 - 32 bit.

3.1 Install from source

If you would like to run from the python source you must have a working Python 2.7 installation. For a full working Python environment I can recommend the Python (x,y) distribution from https://python-xy.github.io/ or Winpython from https://winpython.github.io/ for a Windows environment (just make sure you are using the latest Python 2 package). On Linux and MacOS environments Python should already be installed as part of the OS.

Program is using (at least parts of) the following Python libraries (not in any particular order..):

- time
- sys
- os
- shutil
- ctypes
- struct
- logging
- collections
- win32com
- python-snap7
- socket
- datetime
- gzip
- bz2
- binascii
- ConfigParser
- distutils

This libraries can be installed with pip.

Python-snap7 is a python wrapper for snap7 from http://snap7.sourceforge.net/. So make sure you have a working copy of this also on your machine.

3.2 Install with Windows Installer

To install the program you can use the provided installer. Run setup.exe as Administrator and follow instructions of the installer. A link to the program will be installed into the startmenu. The program will be installed to the directory “C:\Program Files (x86)\Aqserver\”. Additionally the folder “Aqserver” plus sub folders “help”, “log” and “data” will be created in the My Documents folder of the current user. In the “Aqserver” directory you will find the following files:

- test.bat: a Windows / DOS batch file that will start Aqserver with the test configuration. You can copy and edit this file to personalize it for different configuration. I am also using this to start Aqserver from Windows Explorer or to run multiple instances (one batch file per instance)
- test.cfg: an example config file that you can edit to match your requirements (good idea is to make a copy first..)

To uninstall run the uninstaller from the startmenu or uninstall from windows control panel.

3.3 Manual installation

To install the program in a Windows Environment without a working Python installation use a folder of your liking and copy the files aqserver.exe, aqserver.cfg, aqserver.chm and aqserver.pdf to this folder. Copy snap7.dll to your Windows\system32 directory. That’s all. No install or setup procedure. To remove the program just delete these files from your system. Nothing will remain.

CHAPTER 4

Configuration file for data acquisition server

In order to run Aqserver you need a configuration file where all required parameters for a acquisition from a S7PLC are stored. When Aqserver is starting, this configuration is scanned, the parameters a re checked for correctness, then the acquisition starts.

4.1 Prerequisites

Multiline settings must be indented with TAB, see "remarks"

In config file comments start with a "#": "#" at 1st column of line will be a comment, you can also use it to disable settings. Note that always one valid setting must be available

Use ";" for inline comments

Multiline comments must be indented with TAB, see "remarks": If you want to have some text in more than one line, following lines must be indented with TAB, see remarks in aqdata settings

4.2 Settings in config file

4.2.1 Aqdata settings

For information purposes, we can set some texts, that will be included in the data file header. You can add a text by placing the name starting in the first column followed by a "=" and the text you wish to be implemented.

Note: Also all config values are included in the datafile. When you open an de-archived datafile, you can copy these to make the same config file, if you have lost the original one.

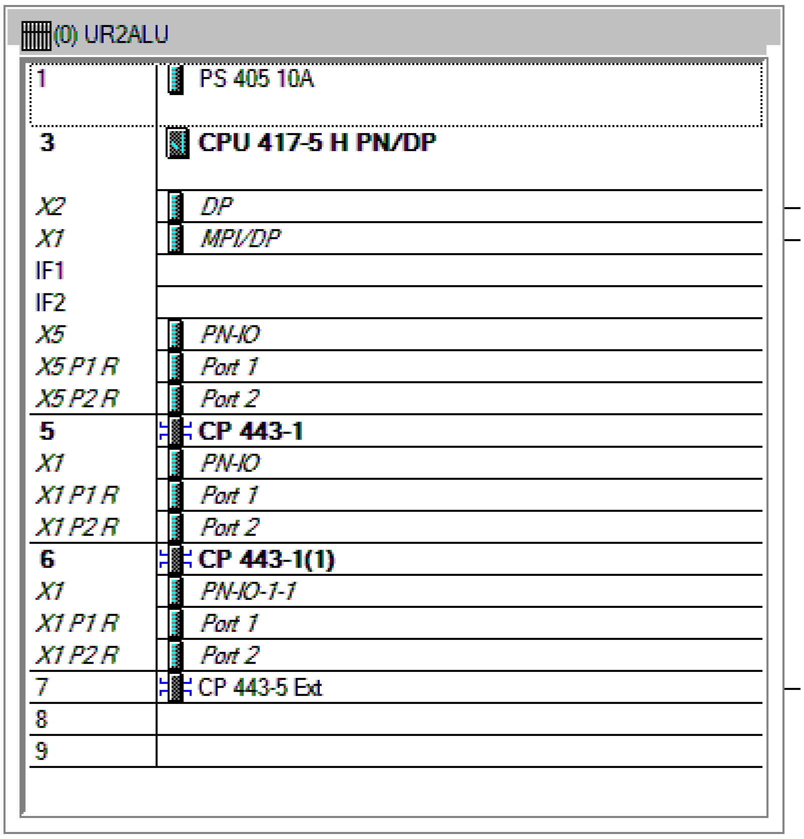
```
########################################################################
# aqdata settings
########################################################################
[aqdata]
place = somewhere

creator = Michael Taxis

machine = MyMachine

order = 123456

yourText = type whatever you desire

remarks = ; use tab for multiline values
```

```
    trials with double spreader, to get rid of married rolls
    trials for SF production
```

4.2.2 Communication settings

For communication with S7 we need following parameters:

- demo: aqserver can run in demo mode, values are created randomly, without communicating to a plc
- IP address: IP address of the PLC you try to reach. Try a PING before you start!
- rack number: normally is 0
- slot number: normally is number left of CPU in HW configuration, in example in picture below it is 3.
- maximum connection attempts: Program tries to connect to PLC. If it fails after this number of trials/attempts the program stops. Then check your configuration and connection. If the value is set to 0 the program will try forever. This can be useful if you have the program running without supervision. If not sucessful you can still stop it with CTRL-C. When communication is interrupted/disconnected during a recording session, it will try to reconnect. There will be no recording of values when disconnected.



(0) UR2ALU

1	PS 405 10A
3	**CPU 417-5 H PN/DP**
X2	*DP*
X1	*MPI/DP*
IF1	
IF2	
X5	*PN-IO*
X5 P1 R	*Port 1*
X5 P2 R	*Port 2*
5	**CP 443-1**
X1	*PN-IO*
X1 P1 R	*Port 1*
X1 P2 R	*Port 2*
6	**CP 443-1(1)**
X1	*PN-IO-1-1*
X1 P1 R	*Port 1*
X1 P2 R	*Port 2*
7	CP 443-5 Ext
8	
9	

```
##############################################################################
# communication settings
##############################################################################
[communication]

# setting up communication parameters to S7-PLC
# demo switch
# if 1 communication will be ignored and values will be randomly created
# if 0 then aqserver tries to connect to a plc
demo = 0
```

```
# IP address
# you can leave several settings , just comment with a leading "#"
#ip = 192.168.1.107
ip = 172.16.13.174

# rack number, see HW Config
rack = 0

# slot number of CPU, see HW Config
slot = 3

# maximum attempts for connection, 0 is trying forever
maxattempts = 10
```

Note: You can also try Aqserver with PLCSIM and NetToPlcsim from http://nettoplcsim.sourceforge.net/ on the same machine.

4.2.3 Miscellaneous settings

We need some general settings for aqserver, as follows:

- delimiter: is the character that separates the recorded values in the data file, normally we use ";". Be careful not to use the decimal separator that is used on your system (so don't use '.' or ',')!
- datafileprefix: here we can define a name, that is used to identify the data file. It is a prefix, because file name also includes a timestamp e.g.: MyProject20150804_173035.csv.gz
- datafile is the filename, without extension, where actual data are recorded. This is a csv-file. If you intend to run multiple instances of the program, in order to reach different PLCs, make sure that this name is different in all config files!
- autostart: defines whether recording is starting with program start, or is waiting for a start signal
- datapath: here we define were the compressed data files will be stored.
- usedir: defines, whether we use a directory structure as \yyyy\MM\dd\ when storing the archived files.
- scantime: scantime in milliseconds[ms], minimum is limited to 20 ms in program. This time is only an approximation, because it also depends on number of variables to scan, but use it to reduce filesize. The more scans the bigger the file. If you set scantime to 0, the program will read the data as fast as possible (Attention: big data file!). Depending on the number of values scantime of ~10 ms can be reached.
- maxrecords: This number defines the maximum number of records stored to one file. This limit the size of a datafile. Depending on the number of values per record you should check what number is applicable for you.
- **booloffset: set this to 1 and the bit in one byte will be offset by 2 as follows:** value + bit number * 2

  With this the bits can be shown in Kst within one plot without overlapping each other

bit	true	false
0	1	0
1	3	2
2	5	4
3	7	6
4	9	8
5	11	10
6	13	12
7	15	14

  if booloffset is 0 then only the boolean value (1 for true, 0 for false) will be stored.

```
########################################################################
# miscellaneous settings
########################################################################
[misc]

# miscellaneous values for setting up the acquisition server
# value delimiter in storage file
delimiter = ;

# prefix of data file name, e.g. a customer/project name or whatever
datafileprefix = MyProject

# data file name for actual data recording, without extension!
# e.g. if you use "filename", actual name will be "filename.csv"
datafile = recording

# autostart: when program is started decide whether acquisition is started(1)
# immediately or wait for start signal (0)
autostart = 0

# path for data files, use "\" for directory separation, with "\" at the end !
# e.g. datapath = D:\mydata\
datapath = F:\aqdata\MyProject\

# if 1 use directory structure datapath\yyyyy\MM\dd otherwise use only datapath
usedir = 1

# scantime in milliseconds [ms]
# if you just put 0 program will scan as fast as possible
# this will produce rather large data files!
# depending on number of values this value is just a hint ;-)
scantime = 100

# maximum number of records
# to avoid too big data files, a new one will be starfted after this number
# of recordings
maxrecords = 50000

# switch for offset of boolean values
# if 1 then boolean values in a byte (see values settings) will be offset by 2 as follows:
#
# value + bit number * 2
#
#  bit | true | false
#  ----+------+-------
#   0  |  1   +  0
#  ----+------+-------
#   1  |  3   +  2
#  ----+------+-------
#   2  |  5   +  4
#  ----+------+-------
#   3  |  7   +  6
#  ----+------+-------
#   4  |  9   +  8
#  ----+------+-------
#   5  | 11   + 10
#  ----+------+-------
#   6  | 13   + 12
#  ----+------+-------
#   7  | 15   + 14
# if booloffset is 0 then only the boolean value (1 for true, 0 for false) will be stored
booloffset = 1
```

4.2.4 Trigger settings

Trigger settings are used to start a new datafile, when a trigger event occurs. A trigger can also be raised manually by pressing key 't' on your keyboard! This trigger event is defined by the following three trigger settings:

- trgsignal: This is the "name" of the signal from the value section, that will trigger the event. Copy the name from the value section.
- trgcondition: This the condition for a comparison of the triggersignal with the trigger value. E.g. when the condition is "==" then the trigger will be raised, when value of trigger signal and trigger value match
- trgvalue: This is a constant, trgsignal is compared with it, to decide about the trigger event.

Further 2 values are used to overlap old and new file:

- pretrg: time that will be recorded to new file BEFORE the trigger in [s]. This is based on setting scantime.
- posttrg: time that will be recorded to old file AFTER the trigger in [s]. This is based on setting scantime.

```
########################################################################
# trigger settings
########################################################################
# when trigger condition is matched, then we close the old file after
# post-trigger time and start the new file and copy pre-trigger time
# and post-trigger recordings to new file
#    # condition is, with example:
# trgsignal trgcondition trgvalue
# rewind diameter [mm] = 0
#
[trigger]

# trigger signal, copy the name of the signal in [values] section,
# that you want to use as trigger signal
trgsignal = rewind diameter [mm]

# trigger condition, use >,>=,==, <=,<,!= as condition
# when conditon is matched, then we close the old file and start a new one
# trgcondition = >
# trgcondition = >=
# trgcondition = ==
trgcondition = <=
# trgcondition = <
# trgcondition = !=

# trigger value, with this value we compare the trigger signal
trgvalue = 0

# pre-trigger time in seconds [s]
# will still add pre-trigger/scantime lines to old file after trigger event
# e.g. pre-trigger is 60 seconds and scantime is 100 ms, then 600 lines
# will be recorded after trigger event
pretrg = 30

#post-trigger time in seconds [s]
# will copy last post-trigger/scantime lines from old to new file
# e.g. post-trigger is 60 seconds and scantime is 100 ms, then 600 lines will
# be copied after trigger event
posttrg = 30
```

4.2.5 Debug settings

The debug settings define whether and how we do some logging to debug the program.

For debugging we have to define a debug level, that defines what will be logged.

With level "0" we switch off logging completely, with level "1" everything will be logged Note that when you restart the program the log directory will be purged,leaving only the latest log-file

Parameter logfile defines the name of the logfile, without extension. Extension will always be ".log"

If parameter logts is 1 everytime we start the program a new log file will be created. If logts is 0 then we will always append to the default log file.

```
########################################################################
# debug settings
########################################################################
[debug]

# debug level
# set logging level to debug, write program actions
# to logfile
# 0 - no logging
# 1 - log INFO messages (default setting)
# 2 - log WARNING messages
# 3 - log DEBUG messages
# 4 - log ERROR messages
# 5 - log CRITICAL messages
# 6 - log EXCEPTION messages
dbglevel = 2

# name of logfile, without extension. Extension will be added as ".log"
logfile = aqserver

# add timestamp to logfile name 1 = yes, 0 = no
# if set to 1 a timestamp will be added to the lofile name. pls. note that a
# new logfile will be created, every time you start the server,
# when dbglevel is > 0
logts = 1
```

4.2.6 Value settings

In the config section values, we can list our PLC variables we want to read. A value definition consists of the name followed by an equal sign and the address of the variable to be read. In the name the unit of the value can be implemented in rectangular brackets []. The unit will be extracted from the name and written in an extra line in the datafile.

The definition of the address does not follow the S7 syntax, because our syntax includes the address, the format (bool, int, float) and the size of the variable in bytes (bool, byte, word, double word) in one parameter. Syntax is described in detail below.

Defining boolean values is a littlebit special, because we always read at least one byte from the PLC. Therefore we split this byte in 8 single booleans. To define which boolean out of 8 we want to record, we have to split the names of the single booleans with a ",".

If we omit text between commas, then this bool will be read but not recorded to the datafile.

See the following examples, where we compare S7 syntax with our definition syntax:

PLC syntax	Format	our definition
DB4615.DBD714	REAL	DB4615.DF714
ED 4	DINT	ED4
AW 4	INT	AW 4
DB4615.DBB6	INT	DB4615.DB6
DB4615.DBX6.1	BOOL	DB4615.DX6 (byte!)

```
########################################################################
# value settings
########################################################################
```

```
# here we define the S7 variables we want to read, and their formats
# here we define the S7 variables that we want to observe
# use following syntax:
#
######### how to define the names: ########################################
# use config value name with [ ] - brackets to define the unit of the value
# units will be separated from the name and put into the datafile
#
# boolean values:
# For boolean values (see format X above) a complete byte is read and then
# split into 8 bits
# To define names for the single bits use ',' to separate the names, e.g.:
#
# bit0,bit1,bit2,bit3,bit4,bit5,bit6,bit7 = DB1234.DX5
# Ventil 1, Ventil 2, Ventil 3, Ventil 4, Res1, Res2, Res3, Res4 = DB1234.DX5
#
# If you do not want all the bits, leave the name empty e.g.:
#
# bit0,,bit2,,,,, = DB1234.DX5
#
# This reads only bit0 and bit2
#
#
######### how to define the values: ############################################
# (S7 variable and format)
# DBn.AFn.x
#
# where:
# - DB is for data blocks or omitted if other area
# - n is DB number or omitted if other area
#
# - . only when data, omitted otherwise
#
# - A is area
#   - D for data
#   - M or F for flags
#   - E or I for inputs
#   - A or Q for outputs
#   - T for timers
#   - Z or C for counters
#
# - F is format:
#
#   - X - for BYTE in BOOL format, followed by byte address:
#   - n is whole number for byte address
#     (attention to address ranges of PLC)
#     will always be split in 8 single booleans
#
#   - B - for BYTE in int format, followed by byte address
#   - n is whole number for byte address
#   (attention to address ranges of PLC)
#
#   - W - for WORD, followed by byte address
#   - n is whole number for byte address
#   (attention to address ranges of PLC)
#
#   - D - for DOUBLE WORD, followed by byte address
#   - n is whole number for byte address
#   (attention to address ranges of PLC)
#
#   - F - for DOUBLE WORD in REAL format, followed by byte address
#   - n is whole number for byte address
#   (attention to address ranges of PLC)
```

```
#
#
[values]
rewind diameter [mm] = DB4615.DF714
webspeed actual [m/min] = DB4615.DF574
vibration left core chuck [mm/s] = DB4614.DF560
vibration right core chuck [mm/s] = DB4614.DF564
vibration rider roll [mm/s] = DB4614.DF568
#Klemmventil UM1,Klemmventil UM2,Klemmventil UM3,Klemmventil UM4,,,, = DB4614.DX564
```

Command line options

Acqserver's command line options are the following: (type 'python aqserver.py –help' to show the text below)

```
>python aqserver.py --help
usage: aqserver.py [-h] [-c CONFIG]

Siemens S7 data acquisition server

optional arguments:
  -h, --help            show this help message and exit
  -c CONFIG, --config CONFIG
                        specify name of configfile, defaults to aqserver.cfg

This program was written using the python-snap7 module and the snap7 library.
A configfile must exist (default file is aqserver.cfg) that defines
communication and other parameters and of course the S7 variables to scan.
Open this file in your preferred editor and read the comments on how to
configure.
Values will be stored to csv-file in subdirectory data. Filename can be
specified in configfile. Once a configurable trigger is raised, a new datafile
begins.
After trigger or when program is stopped, the data file will be compressed and
stored to a path, that can also be specified in the config file.
```

CHAPTER 6

Usage

6.1 Starting from a python console

In a python console program can be started with:

```
>python aqserver.py
```

or

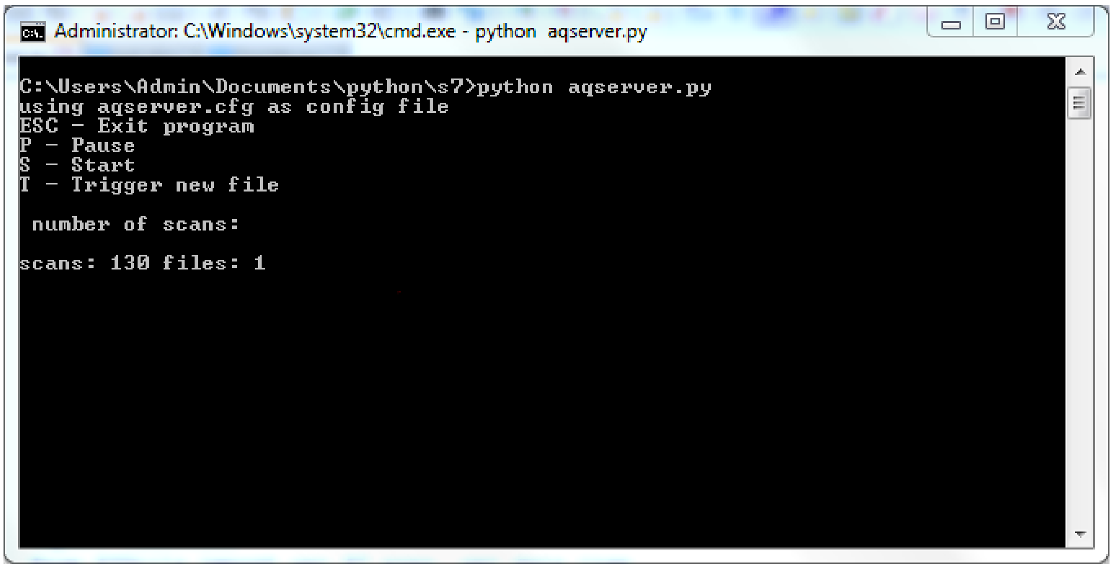
```
>aqserver.py
```

Starting the program without any consecutive arguments, will assume that you will use the default config file name "aqserver.cfg".

It is a good idea to keep a config file for every installation you use to record data. So in this case when not using the default config file you start with:

```
>aqserver.py -c my3rd.cfg
```

or whatever name you give your config file. It is good practice to always use the extension ".cfg" but it will also work without it. Just make sure you always use the complete filename including any extension.


Administrator: C:\Windows\system32\cmd.exe - python aqserver.py

```
C:\Users\Admin\Documents\python\s7>python aqserver.py
using aqserver.cfg as config file
ESC - Exit program
P - Pause
S - Start
T - Trigger new file

 number of scans:

scans: 130 files: 1
```

As you can see in the picture above, the program is displaying some options plus the number of scans in the current data file and the number of data files recorded so far. The options that you can use are as follows:

- ESC - Exit program : When you hit the "ESC" key the current file will be saved and then the program stops.

- P - Pause: recording can be paused with the “p” key, meaning nothing will be read from the PLC and nothing will be written to the file. This can be useful if you have a break in production, but you still want the file to be continued.
- S - Start: When program was paused or when config item autostart is set to 0 then you can start/continue recording with key ‘S”.
- T - Trigger new file: With the “t” key you can manually raise a trigger. The actual data file will be stored and a new file will be started, indicated by the increasing number of files. Also number of scans start with 1 again.

6.2 Using Windows installer version

When you have installed Aqserver using the Windows installer, then usage is slightly different. You also might not have a working python environment installed. The installer installs the aqserver exe to the directory “C:\Program Files (x86)\Aqserver\”. An example config file and a Windows / DOS batch file is copied to the ”..\My Documents\Aqserver\” directory. Once you have defined your config file you should edit a copy of the batch file accordingly, so that aqserver will be started using your new config file. Contents of the batch file:

```
"C:\Program Files (x86)\Aqserver\aqserver.exe" -c "path_to\test.cfg"
pause
```

Change path and name of your config file. Then Aqserver can be started with double clicking the batch file in Windows Explorer.

CHAPTER 7

Running multiple instances

If you have 2 (or more..) PLCs you want to monitor, you can do so, by running 2 separate instances of aqserver:

Create a second config-file, containing the data required for the second connection.

Note: Make sure you use a datafile name that is different to the other instance(s).

- use a different name for the config-file
- in the config-file set a different datafile name
- start acqserver from the commandline, using the -c option
- e.g. in Windows run:

```
>python aqserver.py -c my_new_config_file.cfg
```

If you have installed Aqserver with the provided Windows installer, you can use the batch file(s) in your ”..\My Documents\Aqserver\” folder to start the program. (See also “Installation” section of this manual). For every instance create a batch file that contains the correct settings for the used config file.

CHAPTER 8

Problems

When program exits unexspected or with a message, or when program crashes, then you should turn on debugging. By default debugging is off. To run the program with debugging enabled open your config file and set configuration value “dbglevel” to a value >0. (See below). 1 for INFO is the most verbose level. If you face problems / errors set at least to 3.

```
###############################################################################
# debug settings
###############################################################################
[debug]

# debug level
# set logging level to debug, write program actions
# to logfile
# 0 - no logging
# 1 - log INFO messages (default setting)
# 2 - log WARNING messages
# 3 - log DEBUG messages
# 4 - log ERROR messages
# 5 - log CRITICAL messages
# 6 - log EXCEPTION messages
dbglevel = 0
```

Then you should start the program again and after program has ended (again) you should check the log file in the sub directory log. There you should find some more detailed information why the program won’t run. Adjust (probably) your configuration and try to run again.

Problems can be caused by:

- PLC cannot be reached with communication parameters provided
- value in your list does not exist in PLC
- typo in configuration values

If you still can’t manage to make it run, send a copy of your logfile and the config file to my mail address (aqserver at taxis-instruments dot de).

8.1 Frequently asked questions

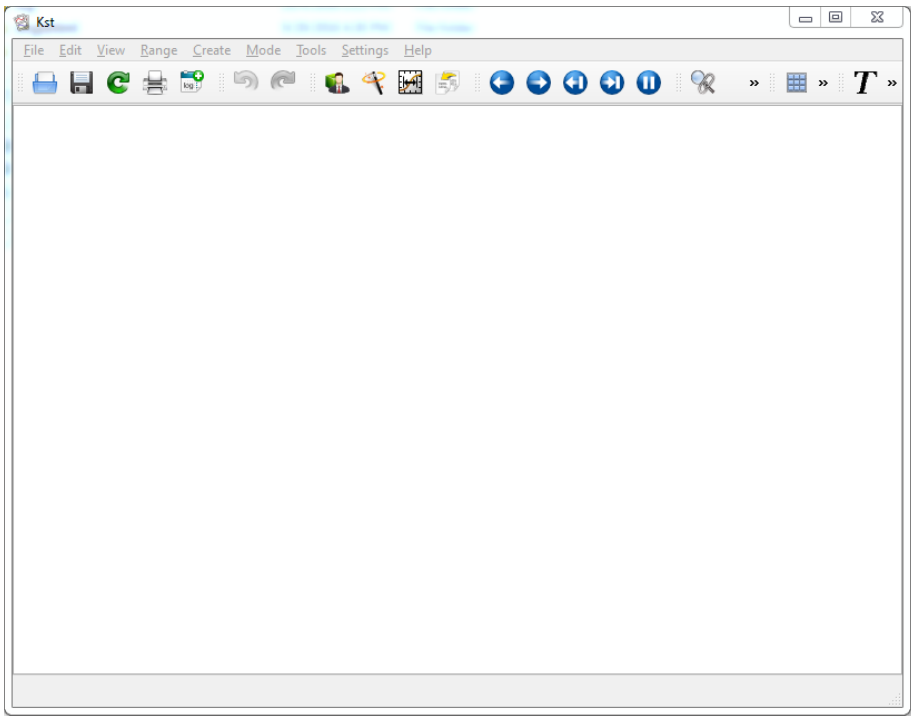
None so far. This will be updated as questions arise. Post your questions in the contact form on my homepage or send mail to <aqserver at taxis-instruments dot de>

CHAPTER 9

Introduction to Kst2

Kst2 is an open source graph and plot creator and visualizer. The program can be downloaded from from https://kst-plot.kde.org . Most recent Windows version is 2.08. On the homepage of the program some tutorial videos are provided. Maybe take a look at these before starting to use Aqserver and Kst2.

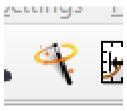
9.1 Usage

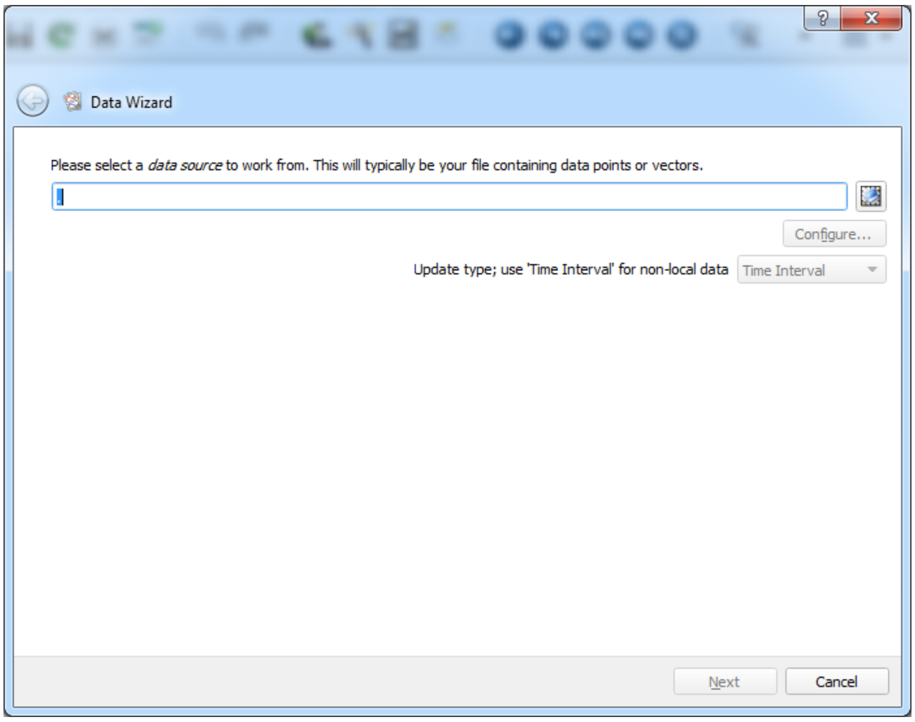


The easiest way to use Kst2 as an online chart visualizer together with Aqserver is to start with the data wizard.

Note: You can start the data wizard several times, also with the same file, when you want to add more graphs to your display. Also if you run multiple instances of Aqserver you can pick a different datafile.

Start it by clicking the following icon:





Data Wizard

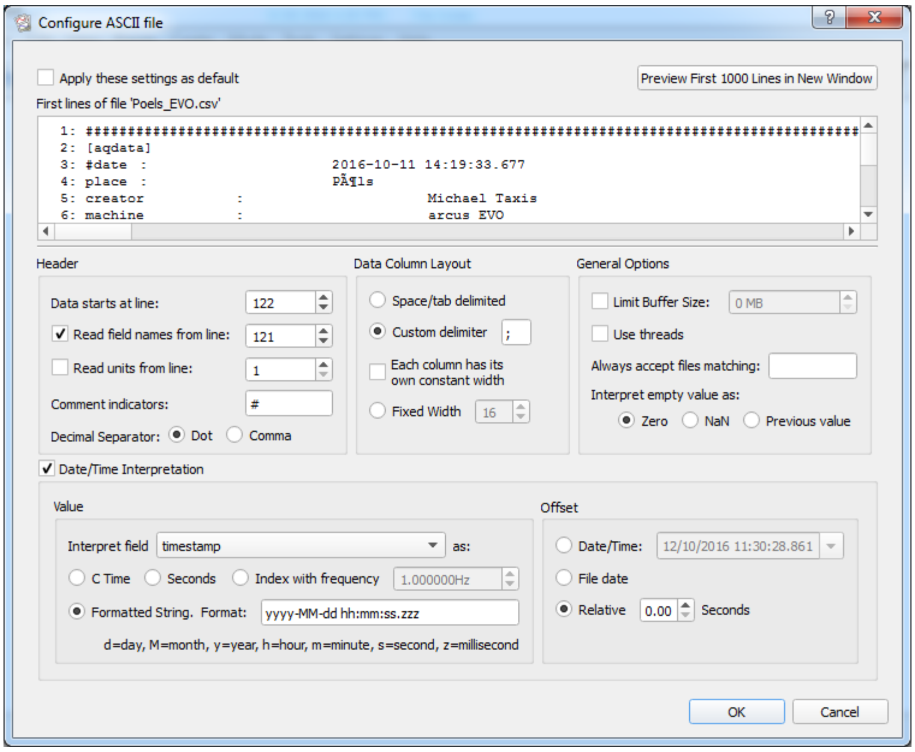
Please select a *data source* to work from. This will typically be your file containing data points or vectors.

Configure...

Update type; use 'Time Interval' for non-local data: Time Interval

Next | Cancel

The data wizard window will popup where you must select the active data file (this will be in your sub directory data). Browse to the file to select it and then click the configure button.



Configure ASCII file

☐ Apply these settings as default — Preview First 1000 Lines in New Window

First lines of file 'Poels_EVO.csv'

```
1: ####################################################################################
2: [aqdata]
3: #date  :                    2016-10-11 14:19:33.677
4: place  :                    PÃ¶ls
5: creator           :                    Michael Taxis
6: machine           :                    arcus EVO
```

Header

- Data starts at line: 122
- ✓ Read field names from line: 121
- ☐ Read units from line: 1
- Comment indicators: #
- Decimal Separator: [✓ Dot] Comma

Data Column Layout

- Space/tab delimited
- [✓ Custom delimiter] ;
- ☐ Each column has its own constant width
- Fixed Width 16

General Options

- ☐ Limit Buffer Size: 0 MB
- ☐ Use threads
- Always accept files matching:
- Interpret empty value as: [✓ Zero] NaN Previous value

✓ Date/Time Interpretation

Value

- Interpret field timestamp as:
- C Time Seconds Index with frequency 1.000000Hz
- [✓ Formatted String.] Format: yyyy-MM-dd hh:mm:ss.zzz
- d=day, M=month, y=year, h=hour, m=minute, s=second, z=millisecond

Offset

- Date/Time: 12/10/2016 11:30:28.861
- File date
- [✓ Relative] 0.00 Seconds

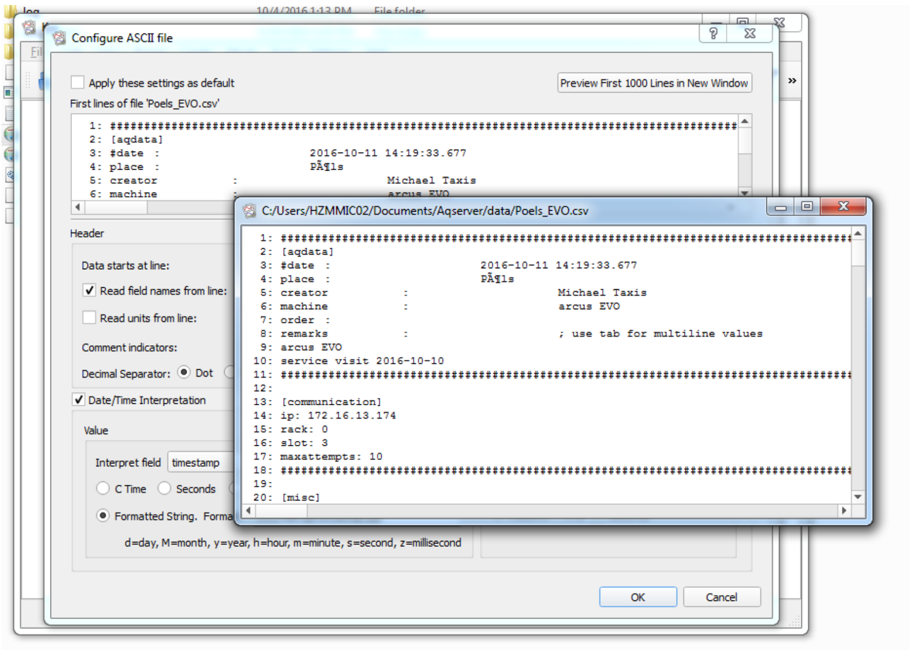
OK | Cancel

The "configure ASCII file" window will popup. Here you must configure the properties of your data file. Values you must adjust include:

- Data starts at line: scroll down the window showing your datafile until you reach the first recorded values. Note the number and put into the required field. If the first data line is not shown, try the button "Preview first 1000 lines in new window".
- Obviously the "Read field names from:" option must activated and the acording line number must be entered (see above value - 1)

- set the decimal separator
- set the Custom delimiter to the one you have used in the datafile
- activate “Date/Time Interpretation”
- scroll down the “Interpret field as” and pick the value timestamp. (If no selection is shown, exit with “Ok” and enter configuration again, then try to pick timestamp).
- activate “formatted string” and set the format to “yyyy-MM-dd hh:mm:ss.zzz”. Then our timestamp will be interpreted correctly.

Leave the window with ok. Set the update type to either time interval or you can try change detection. When using an already recorded archive you can use no update. Then click “Next” in the Data Wizard window.


Configure ASCII file

Apply these settings as default

Preview First 1000 Lines in New Window

First lines of file 'Poels_EVO.csv'

Header

Data starts at line:

✓ Read field names from line:

Read units from line:

Comment indicators:

Decimal Separator: ● Dot

✓ Date/Time Interpretation

Value

Interpret field timestamp

○ C Time ○ Seconds

● Formatted String. Forma

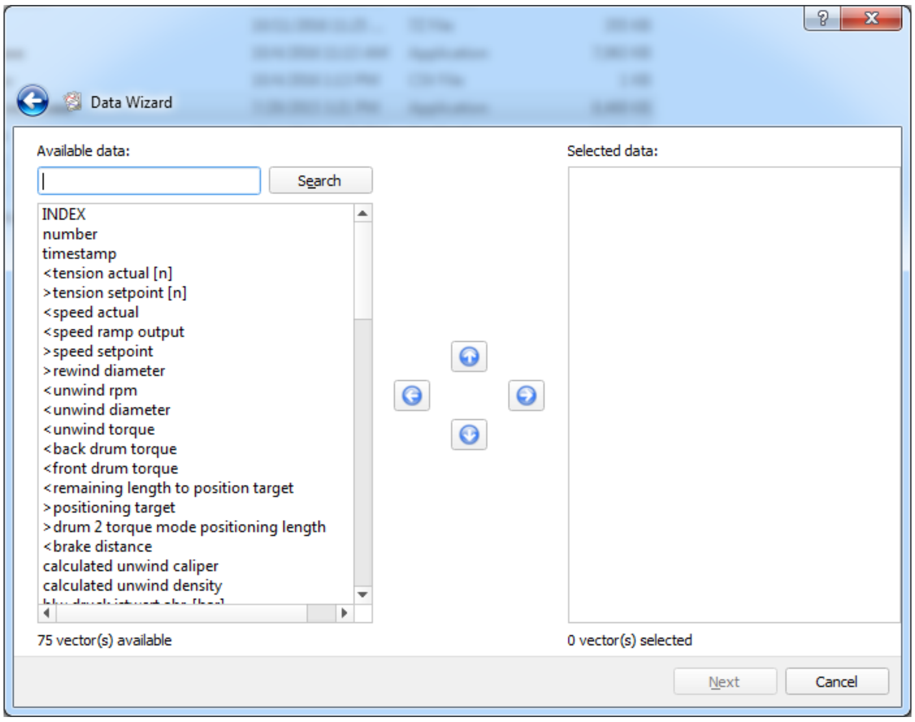
d=day, M=month, y=year, h=hour, m=minute, s=second, z=millisecond

OK Cancel

C:/Users/HZMMIC02/Documents/Aqserver/data/Poels_EVO.csv

```
 1: ##########################################################################################
 2: [aqdata]
 3: #date  :                         2016-10-11 14:19:33.677
 4: place  :                         PÃ¶ls
 5: creator           :                          Michael Taxis
 6: machine           :                          arcus EVO
 7: order  :
 8: remarks           :                          ; use tab for multiline values
 9: arcus EVO
10: service visit 2016-10-10
11: ##########################################################################################
12:
13: [communication]
14: ip: 172.16.13.174
15: rack: 0
16: slot: 3
17: maxattempts: 10
18: ##########################################################################################
19:
20: [misc]
```

Data wizard with “Preview first 1000 lines in new window”.


Data Wizard

Available data:

Search

INDEX
number
timestamp
<tension actual [n]
>tension setpoint [n]
<speed actual
<speed ramp output
>speed setpoint
>rewind diameter
<unwind rpm
<unwind diameter
<unwind torque
<back drum torque
<front drum torque
<remaining length to position target
>positioning target
>drum 2 torque mode positioning length
<brake distance
calculated unwind caliper
calculated unwind density

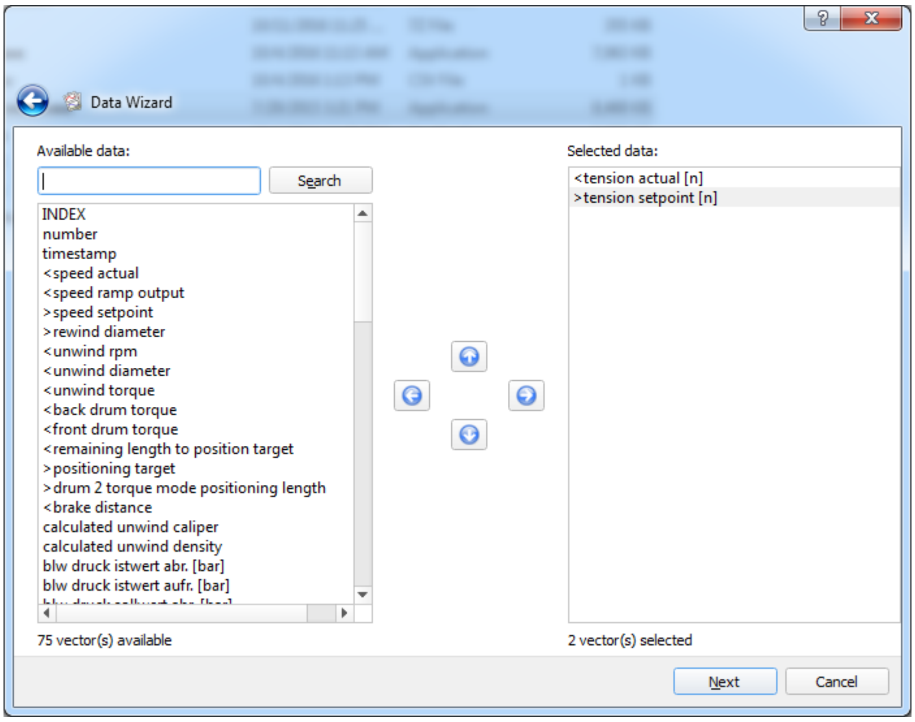
75 vector(s) available

Selected data:

0 vector(s) selected

Next Cancel

Pick the values you want to be shown in a graph and move them to the right window with the arow buttons. Click next



Available data:

```
INDEX
number
timestamp
<speed actual
<speed ramp output
>speed setpoint
>rewind diameter
<unwind rpm
<unwind diameter
<unwind torque
<back drum torque
<front drum torque
<remaining length to position target
>positioning target
>drum 2 torque mode positioning length
<brake distance
calculated unwind caliper
calculated unwind density
blw druck istwert abr. [bar]
blw druck istwert aufr. [bar]
```

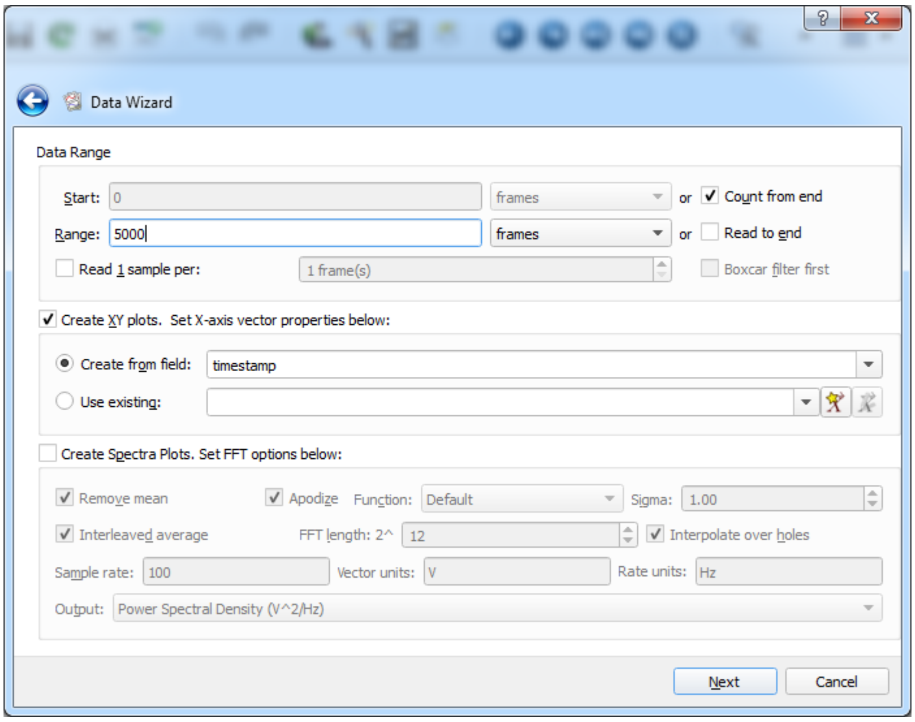
75 vector(s) available

Selected data:

```
<tension actual [n]
>tension setpoint [n]
```

2 vector(s) selected

Same with some values selected.



Data Range

Start: 0 frames or [✓] Count from end

Range: 5000 frames or [ ] Read to end

[ ] Read 1 sample per: 1 frame(s) [ ] Boxcar filter first

[✓] Create XY plots. Set X-axis vector properties below:

(●) Create from field: timestamp

( ) Use existing:

[ ] Create Spectra Plots. Set FFT options below:

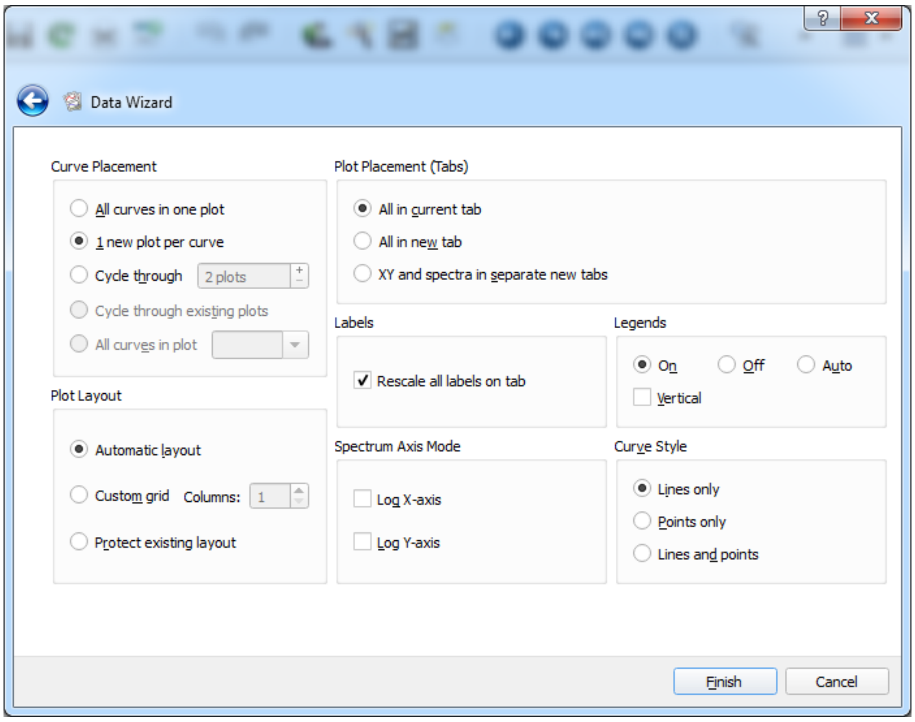
[✓] Remove mean [✓] Apodize Function: Default Sigma: 1.00

[✓] Interleaved average FFT length: 2^ 12 [✓] Interpolate over holes

Sample rate: 100 Vector units: V Rate units: Hz

Output: Power Spectral Density (V^2/Hz)

Experiment a little bit with data range settings. Activate “Create XY-plots”. For X-axis vector we select our field “timestamp”. Click next.



Data Wizard

Curve Placement
- All curves in one plot
- [✓ 1 new plot per curve]
- Cycle through 2 plots
- Cycle through existing plots
- All curves in plot

Plot Layout
- [✓ Automatic layout]
- Custom grid Columns: 1
- Protect existing layout

Plot Placement (Tabs)
- [✓ All in current tab]
- All in new tab
- XY and spectra in separate new tabs

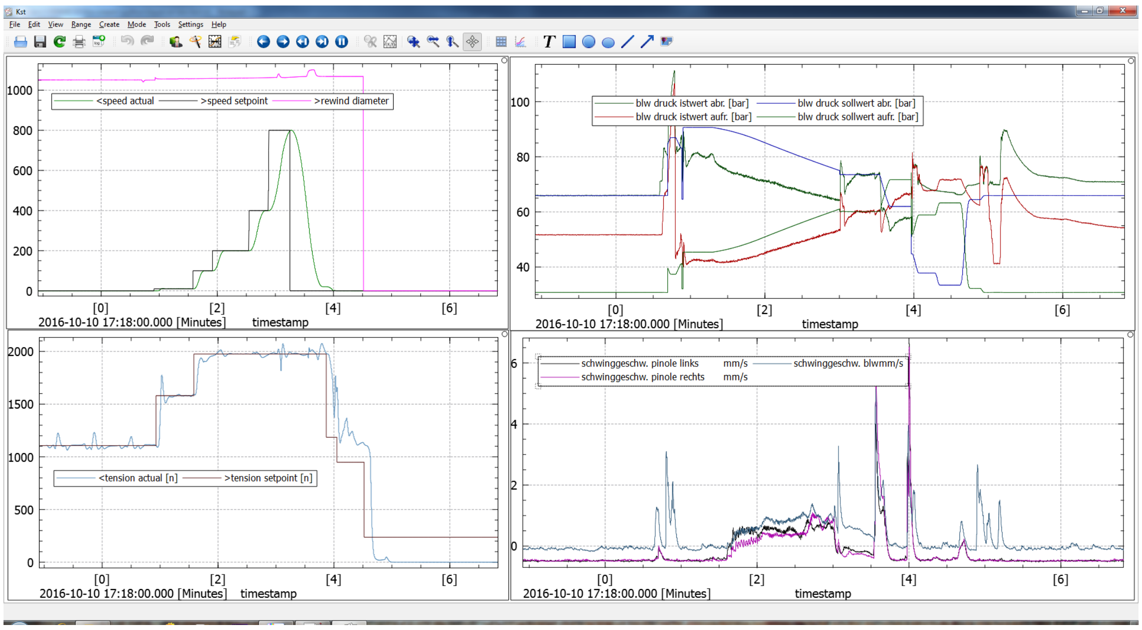
Labels
- [✓ Rescale all labels on tab]

Legends
- [✓ On] Off Auto
- Vertical

Spectrum Axis Mode
- Log X-axis
- Log Y-axis

Curve Style
- [✓ Lines only]
- Points only
- Lines and points

Then we setup the appearance of the graphs, either in one plot or 1 plot per curve.


Kst
File Edit View Range Create Mode Tools Settings Help
<speed actual
>speed setpoint
>rewind diameter
blw druck istwert abr. [bar]
blw druck sollwert abr. [bar]
blw druck istwert aufr. [bar]
blw druck sollwert aufr. [bar]
<tension actual [n]
>tension setpoint [n]
schwinggeschw. pinole links mm/s
schwinggeschw. blwmm/s
schwinggeschw. pinole rechts mm/s
[0]
[2]
[4]
[6]
2016-10-10 17:18:00.000 [Minutes]
timestamp

Finally we will be shown our recorded values. If you have picked an updating file, the curves will also update.

Note: When a trigger has been raised that starts a new datafile use the green round arrow key to reload your datafile. Otherwise Kst2 will not update with the new file.

For further information about Kst2 try their documentation, forums etc.

CHAPTER 10

Changelog

28 March 2017

- aqserver.py: added writing units for values to datafile
- aqserver.py: added functionality for demo: creating data from ramdom function, without connection to plc. DEMO is shown in headline of aqserver
- aqserver.py: added functionality for booloffset: boolean values within one byte are added an offset, so they can be shown in one plot without overlapping each other
- PrgUtils.py: added config.optionxform=str tor get_config for case sensitive config values. ATTENTION: config values IP, RACK and SLOT must be changed to lowercase in your new config files (ip, rack,slot)

17 March 2017

- aqserver.py: added config filename to display, so we know what is running
- aqserver.py: delete header file and temp files when program exits

15 March 2017

- aqserver.py: fixed error: when using multiple instances still same name for header file (and temp file) was used. This messed up data files. Now included datafileprefix in these filenames so they are unique.
- aqserver.py: for same reason logfile name was abandoned, using datafileprefix instead to have unique name

13 February 2017

- added german docs

13 October 2016

- updated docs
- created new installer using pyinstaller and NSIS. Problems before with py2exe.

11 October 2016

- added check for maximum records (from config) to avoid files getting too big
- added try ..except for reading config file

06 October 2016

- PrgUtils.py: added get username for all OS (Windows, Linux, MacOS)
- aqserver.py: change data and log directory, depending on OS. In Windows this will be sub directories of .. My Documents\Aqserver folder, in Linux/MacOS this will be in current users home directory (//home//user//Aqserver//..)

05 October 2016

- NSIS: tried to make setup.exe for Windows
- aqserver.py: added manual trigger always possible

- aqserver.py: added check for variables, and exit with config fault, problem to logfile

30 September 2016

- updating docs
- aqserver.py: check if argument for config file is list or string
- aqserver.py: added purging log directory, when debug is 0 (= no debugging), only actual logfile remains, with entry program start and end
- aqserver.py: improved error detection, connect and disconnect of client

29 September 2016

- updated docs
- compiled to exe with py2exe
- can test now with local Snap7 server
- added verification for configfile, data dir and log dir, if a dir is missing it will be created
- fixed keypress problem, program was not responding to every keypress

15 September 2016

- added try..except for communication error, program exits normal now, with message to user and logfile
- when client was already connected program tries to disconnect and to connect again (no exit)
- when scantime is set to 0 then program scans as fast as possible (no sleep)

28 November 2015

- modified aqserver(none OOP version), can now use directory structure yearmonthday for saved files

11 August 2015

- started refactoring and making OOP aware, with acqserver class (not yet finished)

10 August 2015

- tested and corrected trigger functionality
- tested S7 300 with plcsim and nettoplcsim

07 August 2015

- added filecopy routines to PrgUtils
- added trigger functionality to server (aqserver.py)

06 August 2015

- added config section for debug, with settings for debug level, logfile
- corrected error in endless loop that creates string with values
- create header template at start, that can be copied to further data files in case of trigger
- removed all sys.args, only*c configfile remains, all other settings in config file
- tried to start 2 instances of program
- use different configfile
- set different recording filename
- when started from 2 cmd environments with*c parameter and different configfile, it works!
- tested filecopy, rename, append, replace in file as preparation for trigger event actions

05 August 2015

- added data file name to config, so we can start aqserver several times using different settings, when scanning several PLCs

CHAPTER 11

Licensing

Aqserver is distributed as python source code or Windows setup program under Lesser General Public License version 3.0 (LGPLv3)

Basically this means that you can distribute your commercial software linked with Aqserver without the requirement to distribute the source code of your application and without the requirement that your application be itself distributed under LGPL.

A small mention to the project or the author is however appreciated if you include it in your applications.

11.1 Disclaimer of Warranty