Kontron Czech s.r.o
U Sirotčince 353/7
460 01 Liberec
Czech Republic
tel.: +420 485 100 272
+420 485 108 636
fax: +420 485 100 273
sales@kontron-czech.com
www.kontron-czech.com

OPC server Modbus&Jbus
Aspic 3.30
Aspic MP

Application's Example.

March 2006

Exercise Tasks

The aim of this project is to get familiar with Aspic 3.30 and its implementation to build SCADA/HMI systems. In this project we will use Modbus & Jbus OPC server in its simulation mode as a data source for our application.
This project could be realized using demo versions of upper mentioned software tools, which are available to download for free from manufacturer's web site www.kontron-czech.com .

Project targets:
The target of this project in Aspic 3.30 is to visualize 5 variables (items) from General Modbus & Jbus Master OPC server, and another three Aspic 3.30 internal variables.
The OPC server will be running in simulation mode. It would be advised to set different signal ranges to OPC items, and that to avoid generated signals interference while presenting them.

Advised ranges and types for OPC server's items:

Value1…….. 1- 10 read only
Value2…….. 15- 20 read only
Value3…….. 30- 35 read only
Value4……..100-110 read only
Value5…….. 0- 2 read and write.

Aspic 3.30 Memory Variables:

ValueA…….Mathematically computed sinus wave
ValueB…….controlled by two states buttons.
ValueC.....Connected to slider has 4 states.

First visualization page should contain the following objects:

1) Full graph that charts at least three trends (two OPC variables (Value 1 and 2) + memory variable ValueA).
2) Two „Value" objects at least (Values 1 and 2).
3) Scroll bar object (ValueC).
4) Progress bar object (Value3).
5) An object simulating monitoring led (On/ Off, Green/Black)-(ValueB).
6) Two state button for control. (ValueB).
7) Dynamic animation showing the state of (Value5). The figure will appear in three states related to variable's trend.
   Example the motor is in stand mode (Gray), Value5 = 0
   Example the motor is running (Green), Value5 = 1.
   Example motor failure (yellow), Value5 = 2

Second visualization page will contain a graph through the entire page. Graph setup will be identical to graph setup in the first visualization page. Switching pages to be done via dedicated buttons.

In General:

1) Archive all values 1-5 and A-C, use archiving to Aspic 3.30 archive.

2) Defined alarms are as mentioned here under, each of them needs an operator's Acknowledgement, and stored in archive.
   a. Audible alarm based on the ValueB- active whenever the monitoring led is green.
   b. Alarm based on Value5- active whenever motor failure occurs.
   c. Alarm based on Value1- active whenever exceeding the critical high limit of 9.
   d. Alarm based on Value2- active whenever exceeding the critical high limit of 19.
   e. Alarm based on Value2- active whenever exceeding the critical low limit of 15.
   f. Alarm based on Value3- active whenever exceeding the critical high limit of 30.

3) Gene the following reports:
   a. List of flagged alarms.
   b. Chart the course of Value1.
   c. Chart the course of Value2.
   d. Statement of Value3 values record with one minute scan rate.

List of software tools used:

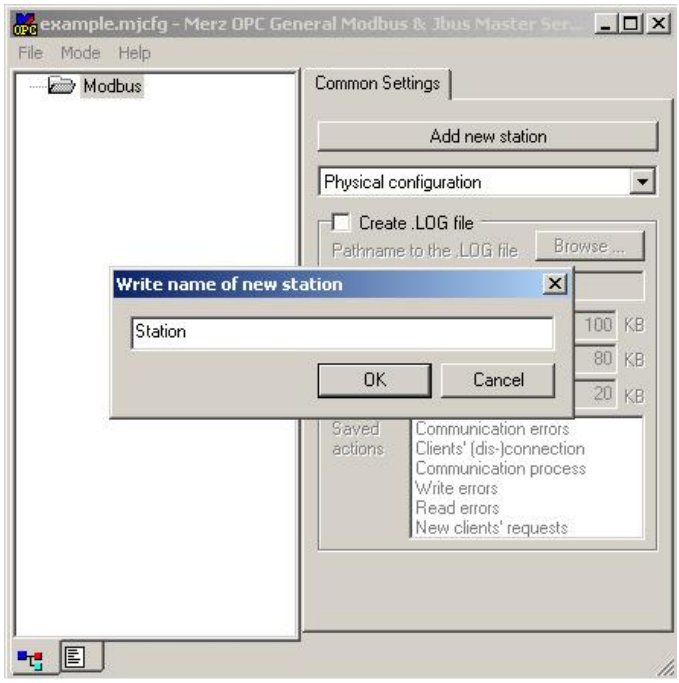
- OPC General Modbus & Jbus Master Server
- Aspic 3.30
- Aspic MP

OPC server as a data source and its configuration.

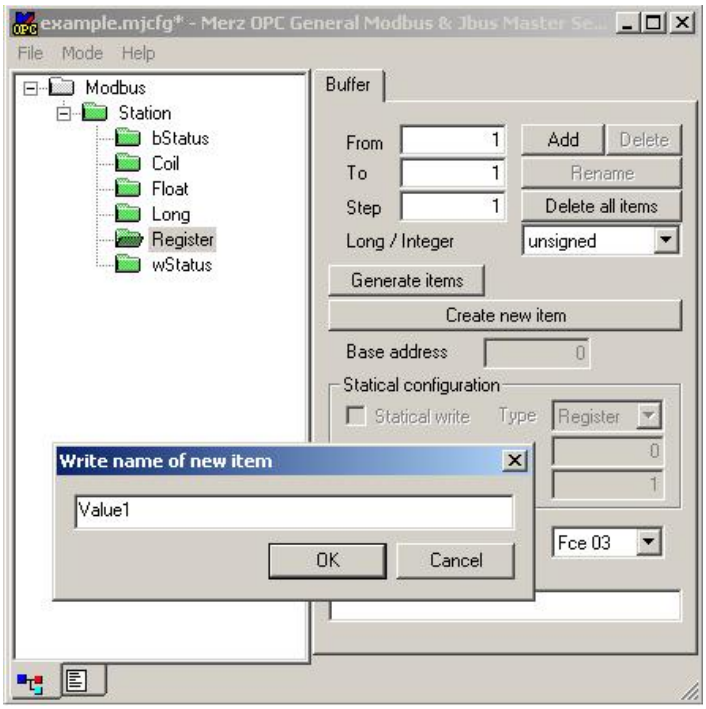
Data source in our example will be the Modbus & Jbus OPC server running in simulation mode. Simulation mode will enable the OPC server to generate a random data without connection to PLC. Pre-requirement is to have the OPC server Modbus & Jbus Installed. Its configuration steps are explained in the following procedure.

Configuration procedure:

1. Run OPC server configuration, and add new station, name it „Station“.

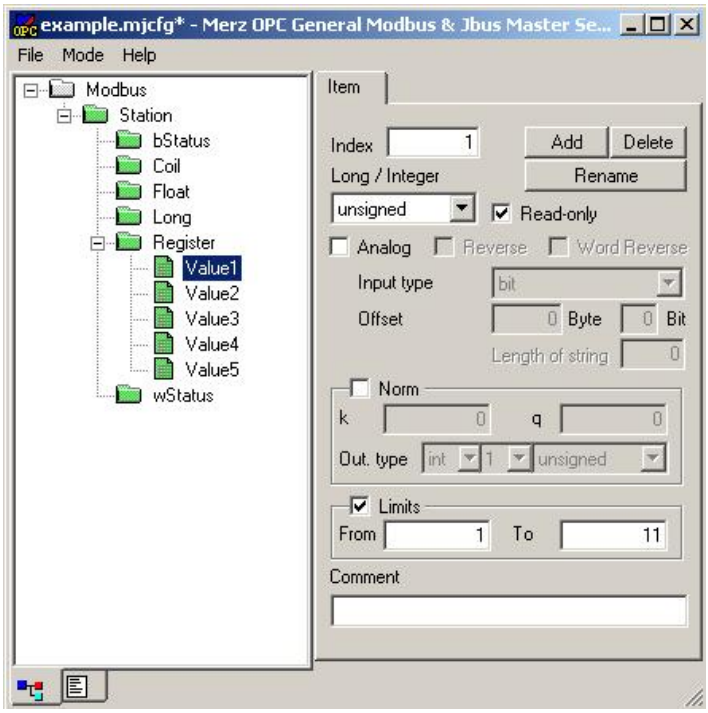


2. In the buffer „Register“ create the five new items we are going to deal with in this example. Those will be named „Value1“till „Value5“.

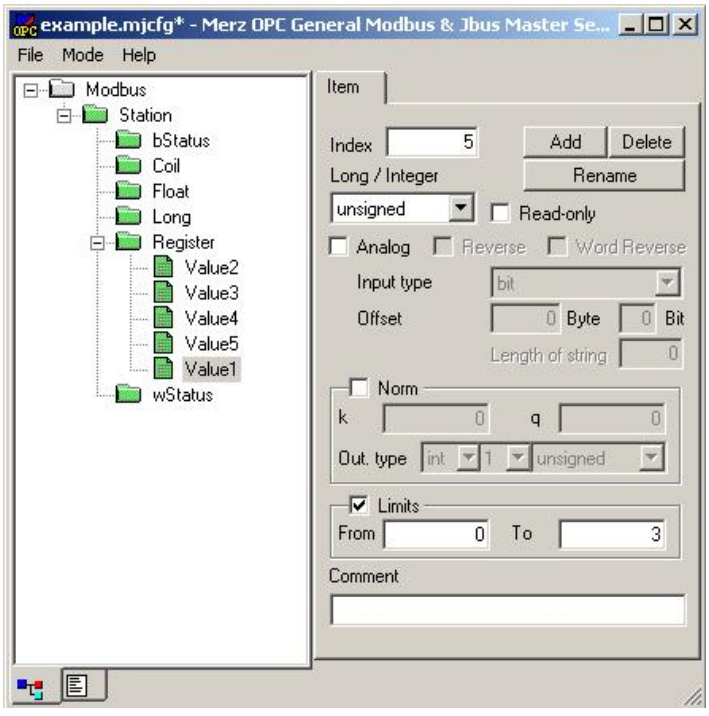


3. For „Value1“configure the following properties. Index value will be „1“, type is „unsigned“, select the checkbox property „Read-only“ and setup limits from 1 to 11.

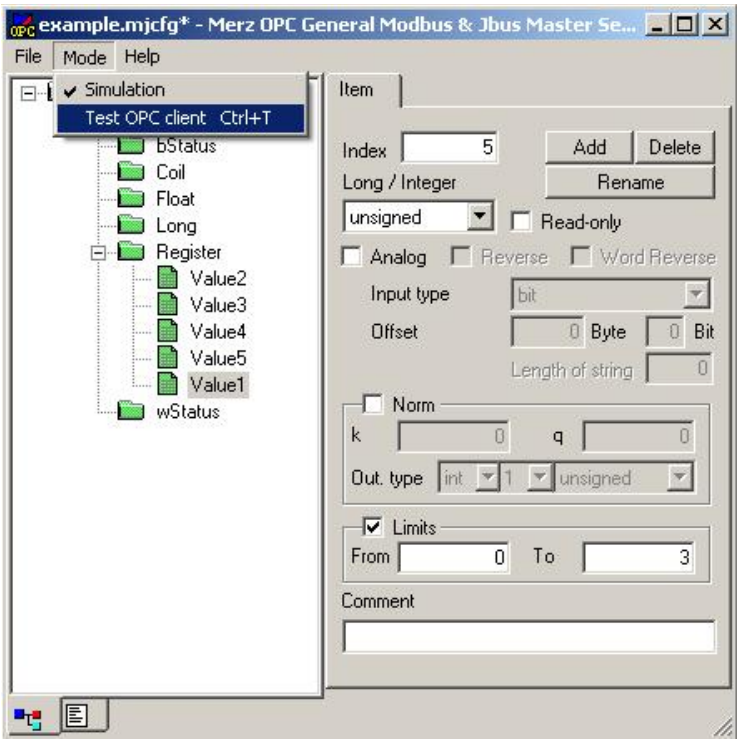
Note: Setting up limits, is essential in case that we need the random generated values from the OPC server running in simulation mode to be within defined limits. For „Value1“ requested range is from 1 to 10. Server simulation generates values till defined upper limit excluding the upper limit itself. While working with integers the upper limit to setup is 11.



4. Similarly setup the properties of other values. The value of index property will be individual for each item. As an example find introduced „Value5“ setup here under, where it differs from previous configuration in deselected checkbox property „Read-only“. This property in this example doesn't have a significant effect. It would have effect when the OPC server will be working in reality (connected to PLC) and not in simulation mode.

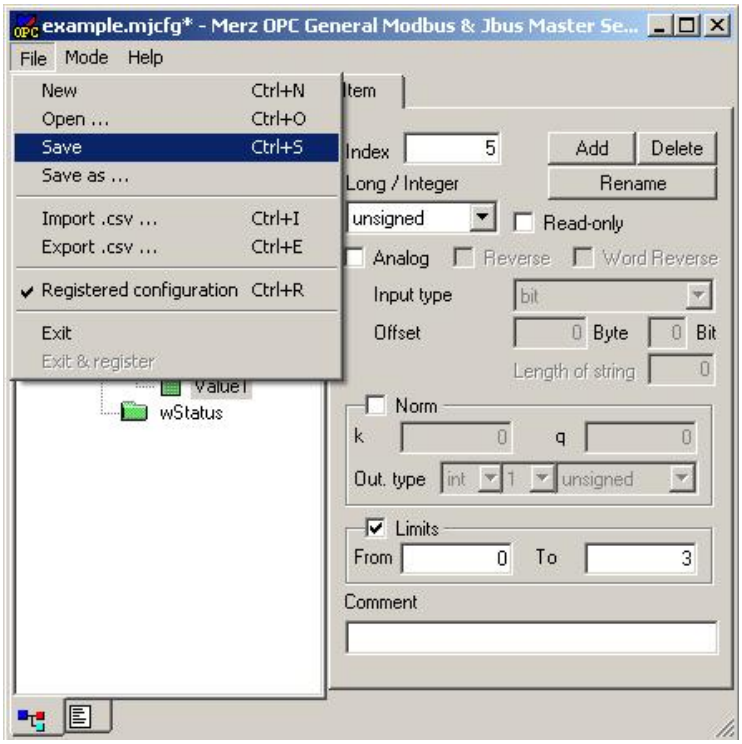


5. Setting up OPC server's simulation mode.

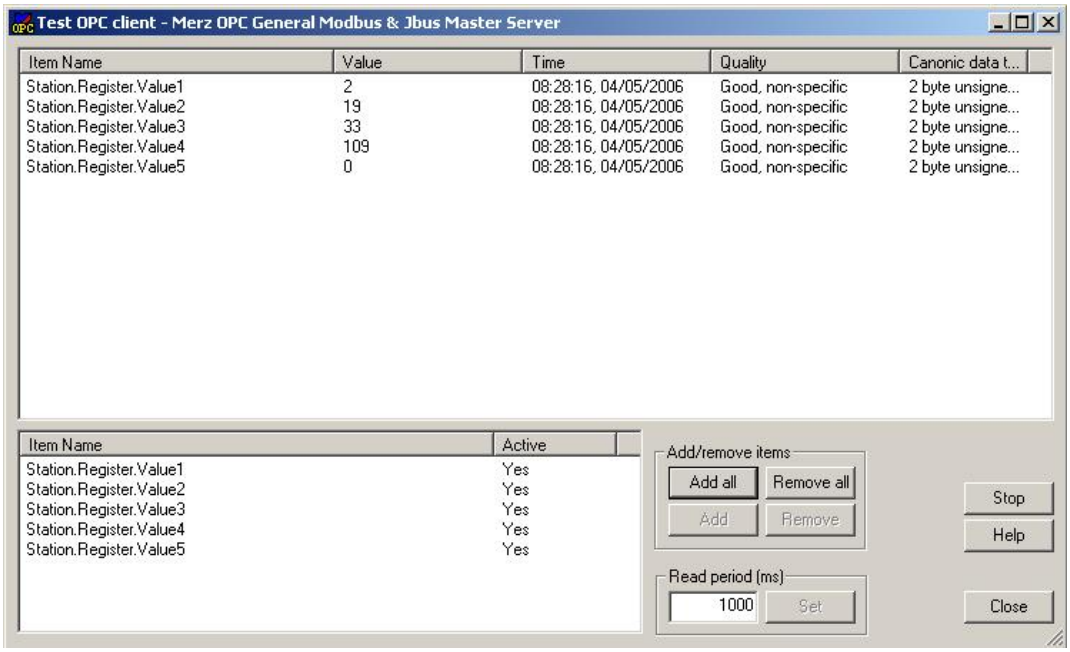


6. Saving created configuration and assigning it as registered.

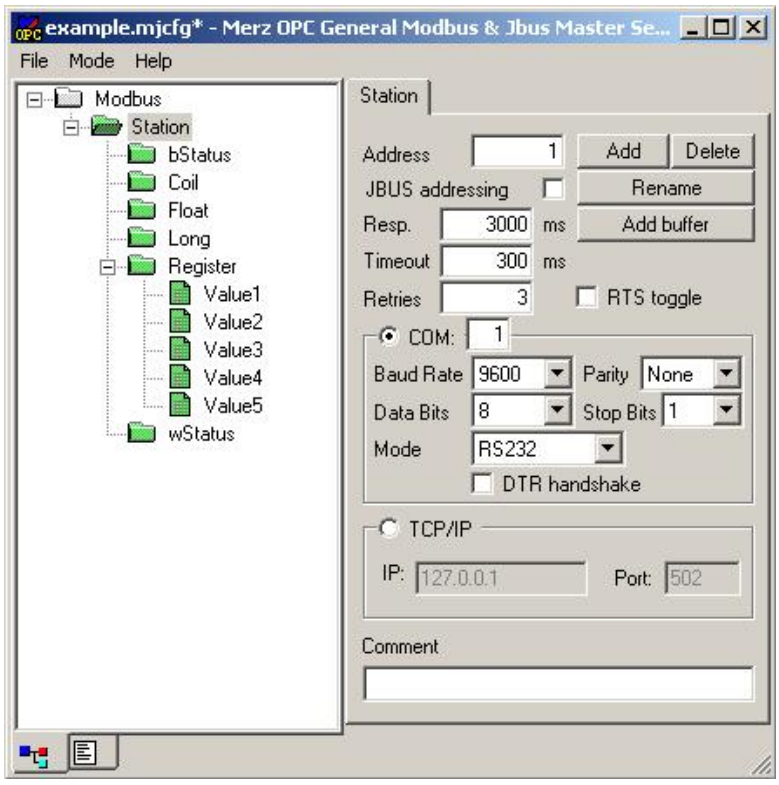
Note: Running OPC server can read only registered configuration. Registered configuration could be only one.



7. Run test OPC client, and add all. If quality displayed is „Good“ and displayed value is changing (for each item), then the OPC server in simulation mode for our example is ready.



Note: For lucidity, it would be better to setup data acquisition rate to 3000 ms (read / write data from / to PLC, in our case just random data generation from the OPC server running in simulation mode.). Slower change rate enables us to watch results in a better way. Configuration is done by changing the value of the property „Resp“ to 3000 ms, according to the following figure. The rest of properties (Address, COM port) could be aborted in our case, its configuration is essential in case of OPC server connection to PLC (not used in simulation mode).

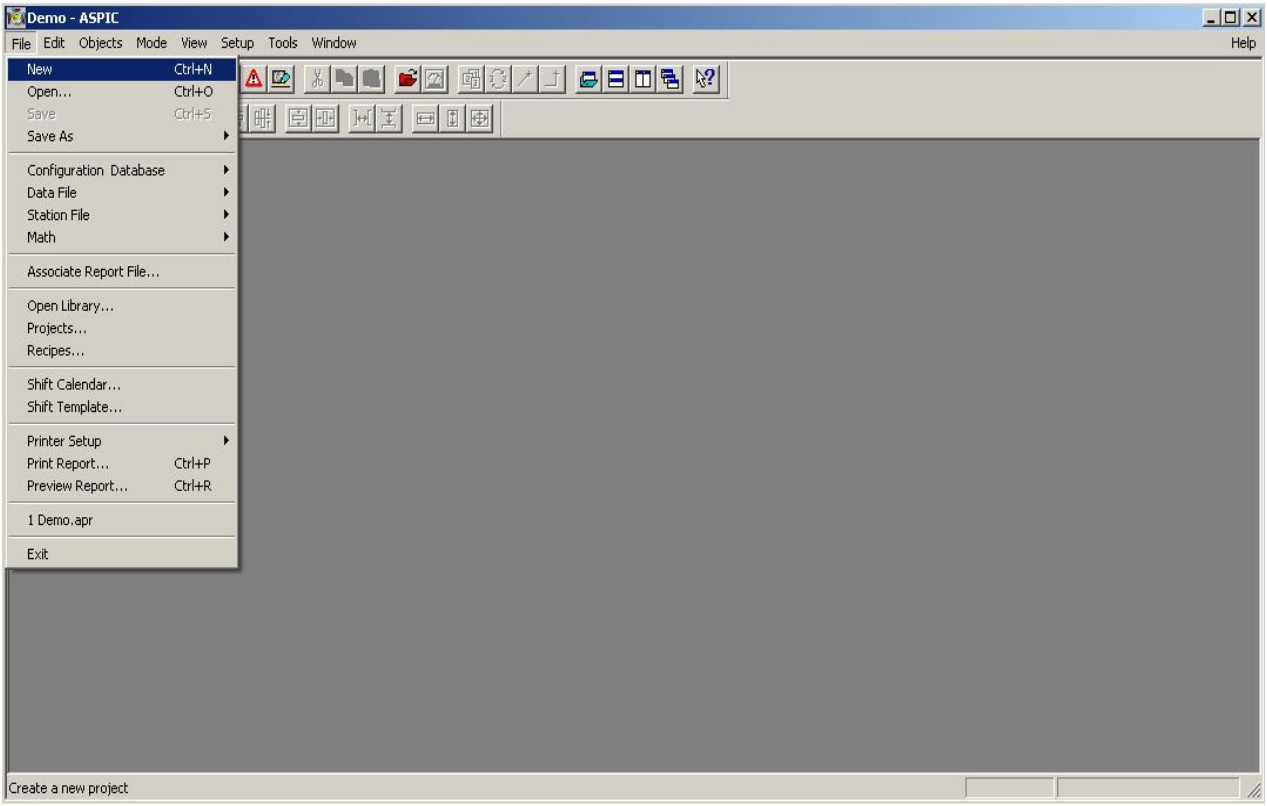


Aspic 3.30, project according to exercise tasks.

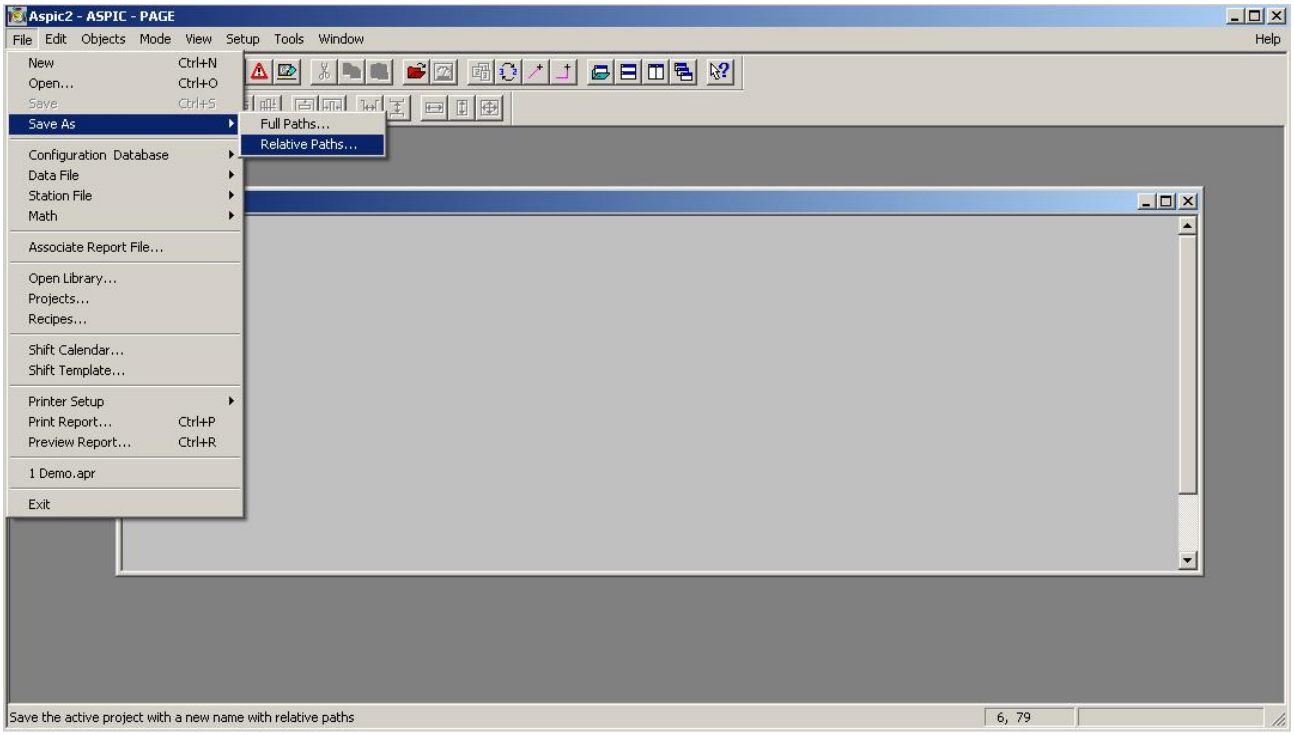
Detailed Aspic 3.30 user guide you can find on Aspic 3.30 on-line Wizard website (http://www.kontron-czech.com > Products > SCADA HMI).

Project creation procedure:

1. Create new project

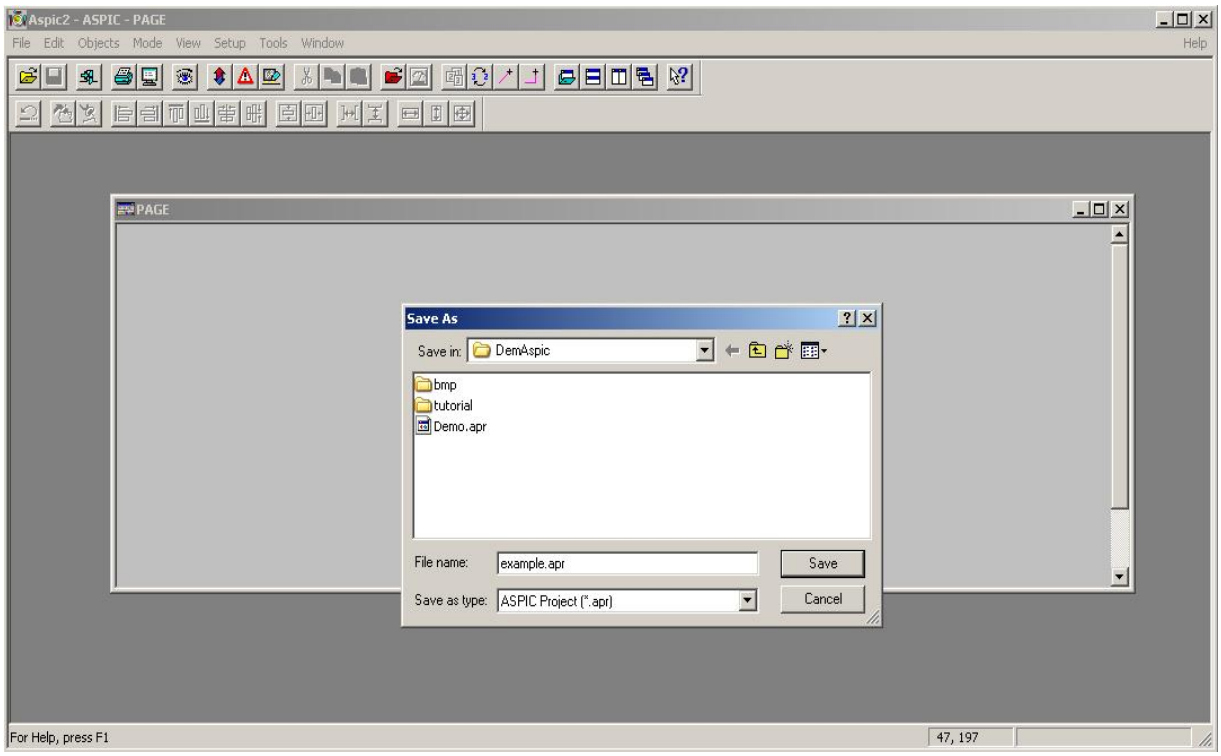


2. Saving the new project with a relative root.

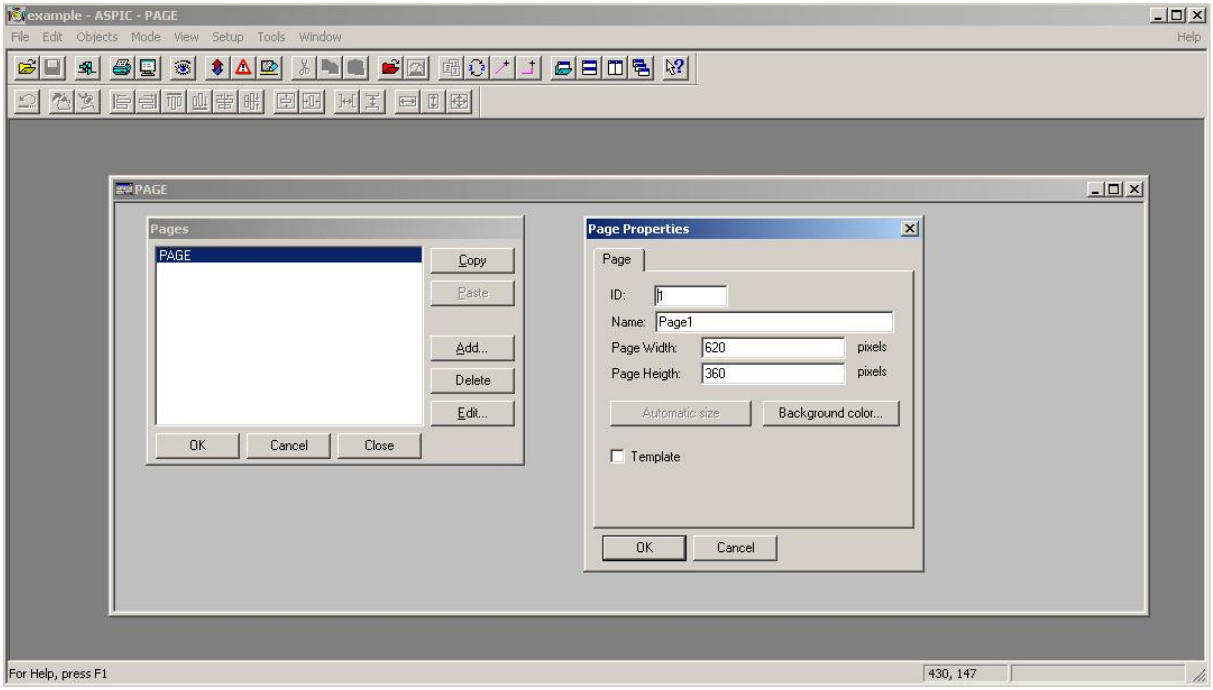


Note: Saving the project with a relative root is useful if we once need to shift the project into different folder.

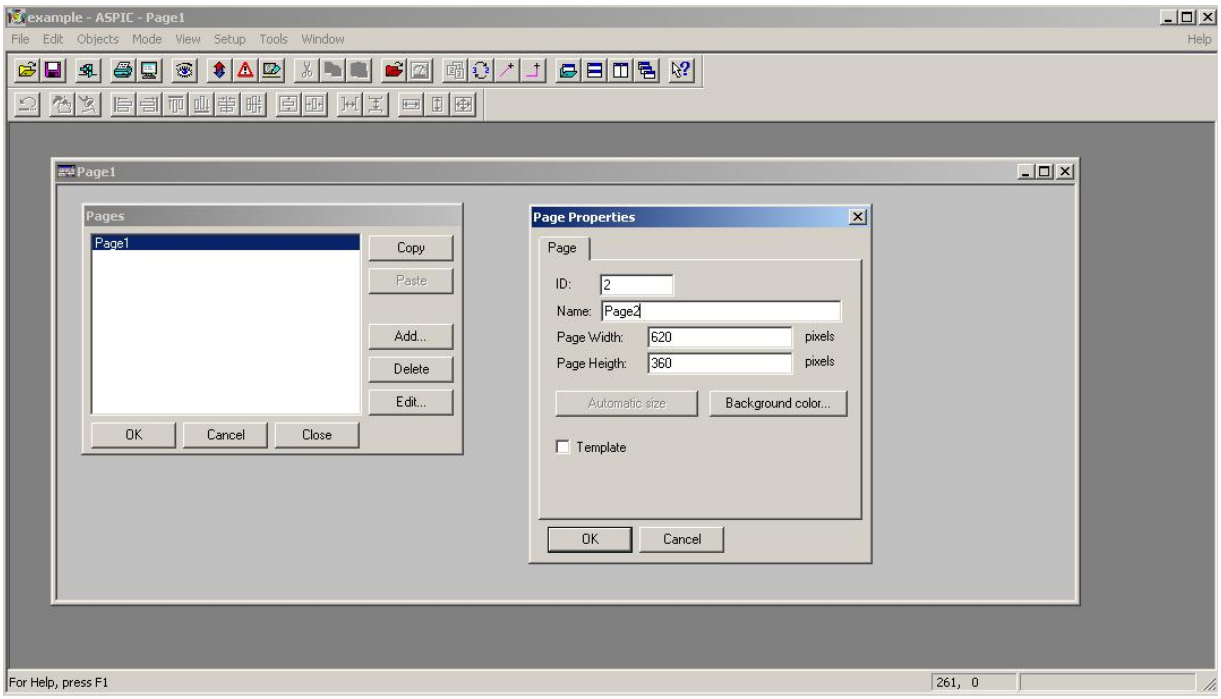
3. Save the project as „Example“and make sure that you save all of its parts. Project file *.apr, System’s file *.asf and data file *.adf. For simplicity keep the same name for all three files. .



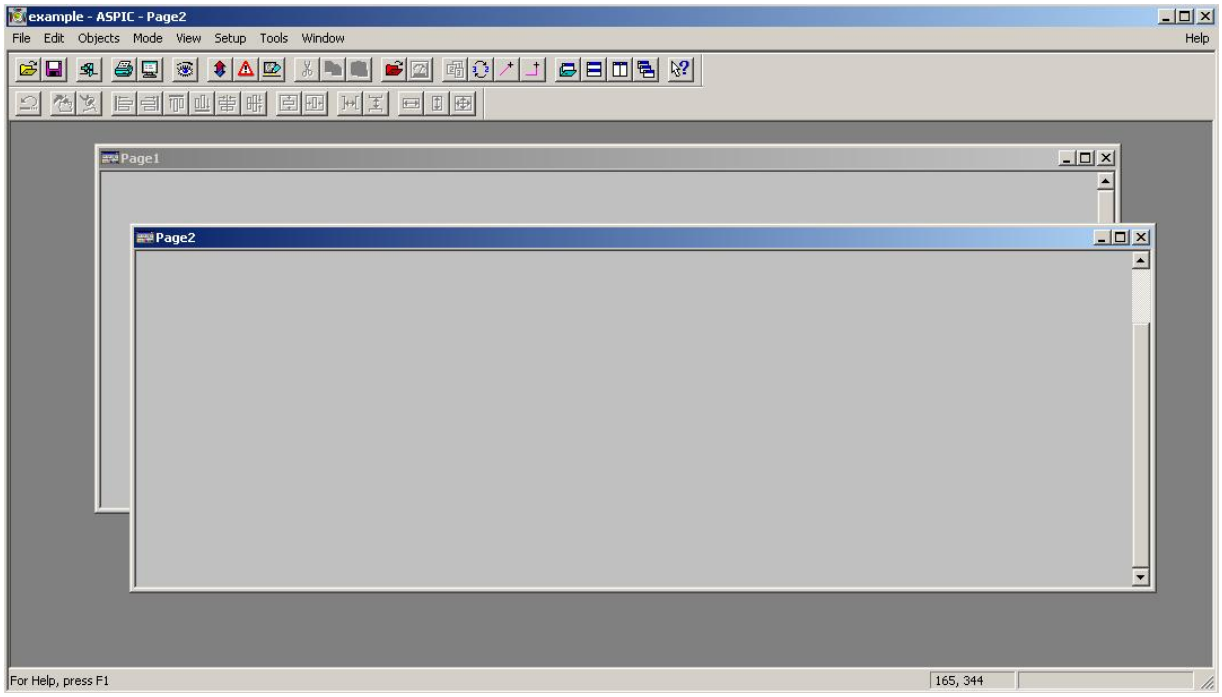
4. Configure page properties by selecting Window->Page (CTRL+w), in this window you will have all visualization project pages. Choose „PAGE“ click edit and change the Name property to „Page1“.



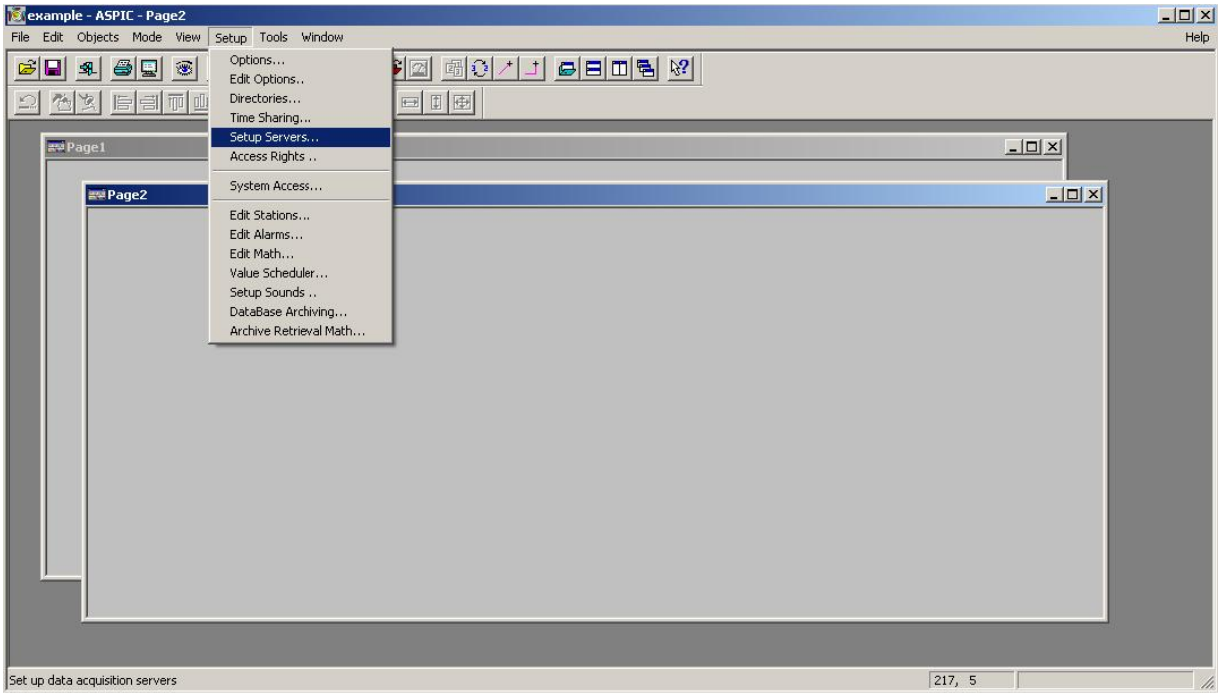
5. According to our example, we will add one more visualization page. Press the button „Add“. The second visualization page’s name will be „Page2“.



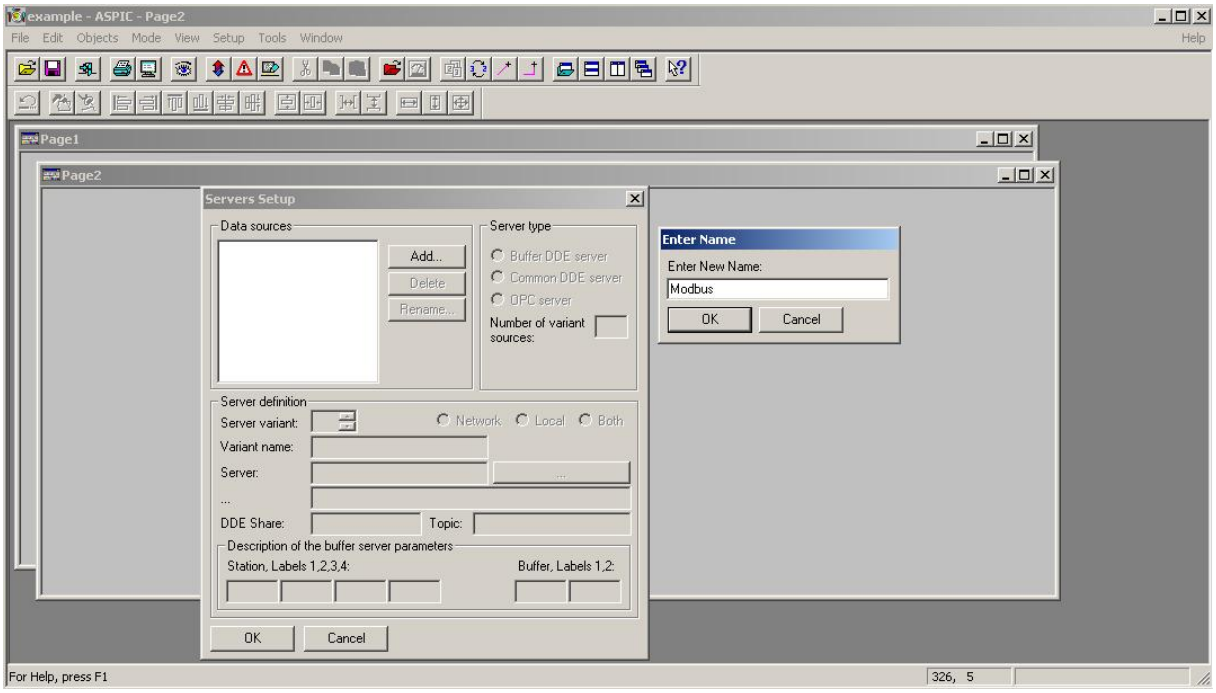
6. The following figure tells how Aspic 3.30 workspace will look like with two visualization pages.



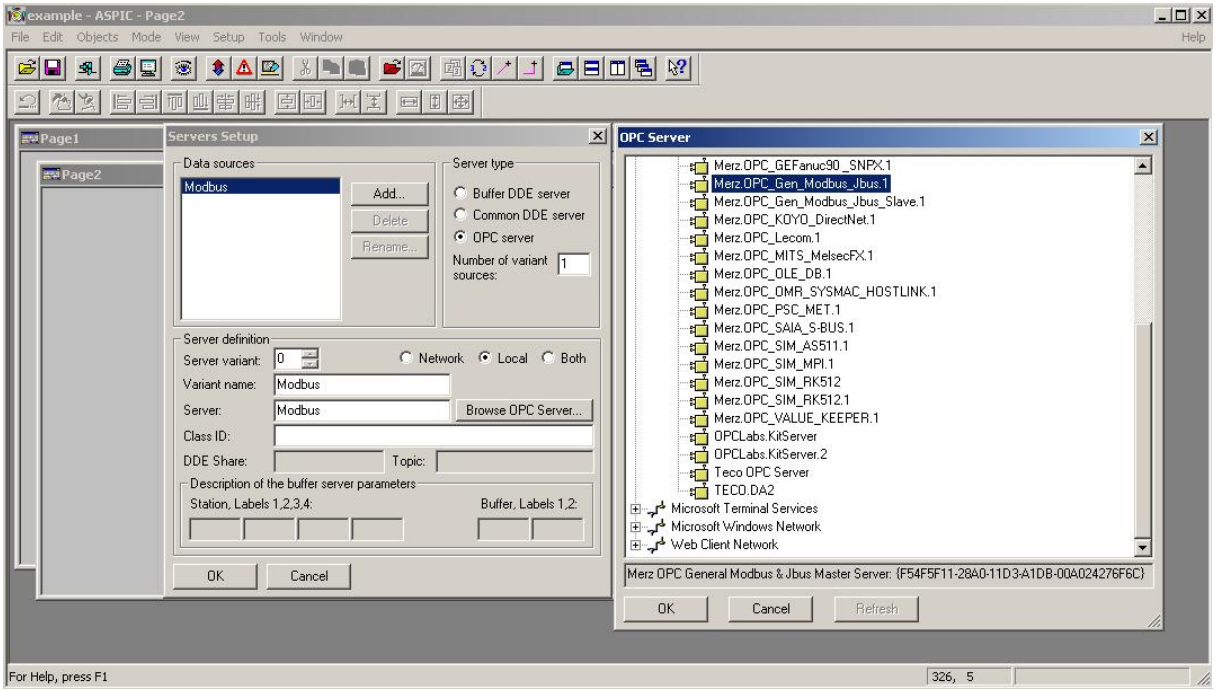
7. We will continue our example by setting up servers. Those servers will be our data sources. To data sources in Aspic 3.30 belongs DDE and OPC servers. In our example we will deal with the Modbus OPC Server running in simulation mode. The server that we have already configured above here in this document to serve further our example.



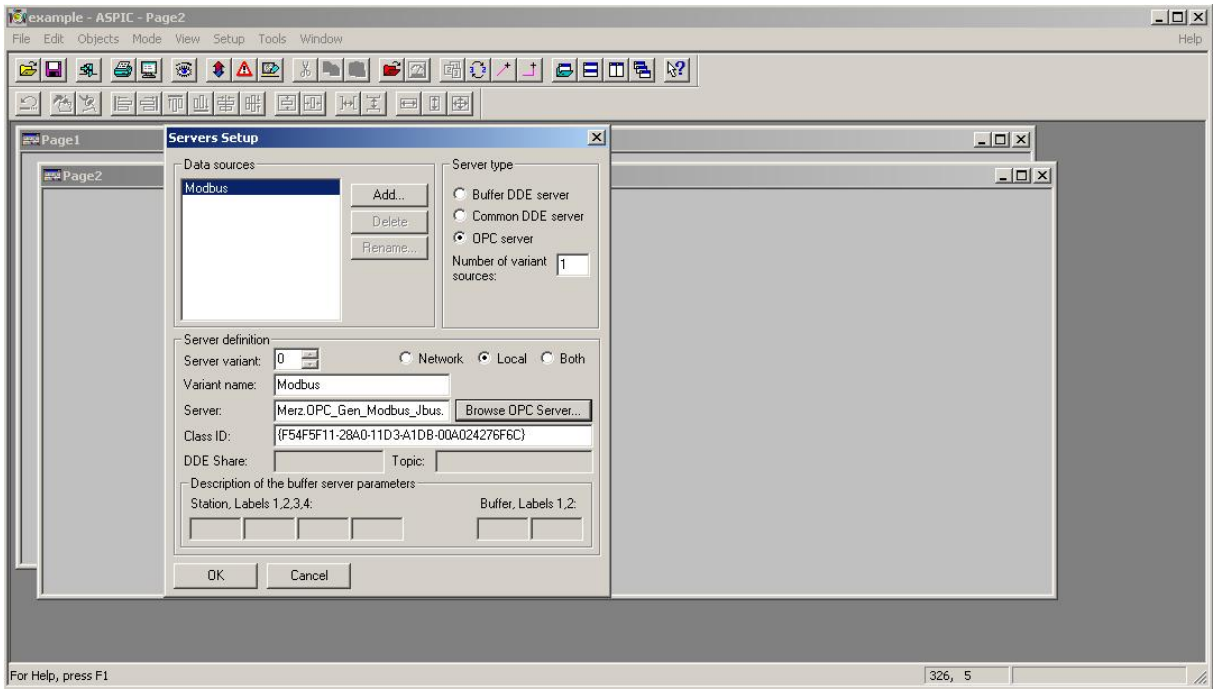
8. Push down the button „add“ in the dialogue „Data source“, and write down the name of the new server. In our example we will use the name „Modbus“. It would be the server’s is a symbolic name in Aspic 3.30; you can use any name you wish.



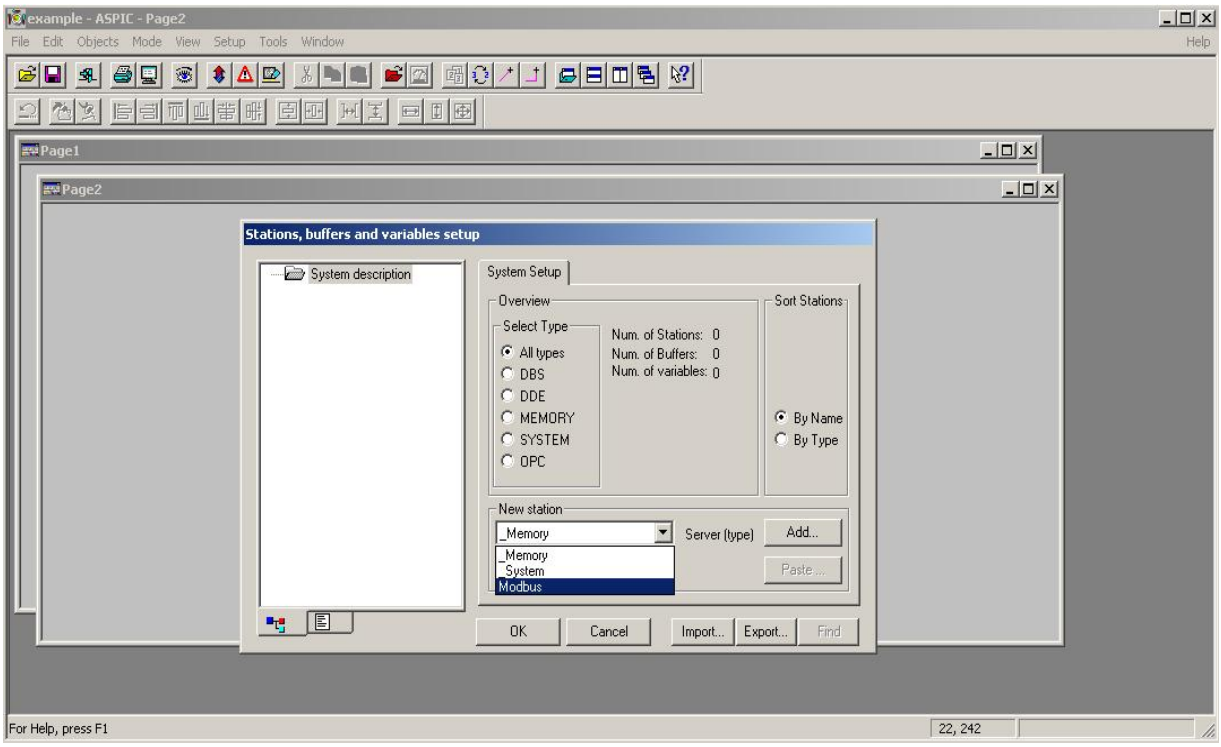
9. We have already got a symbolic name for our server in Aspic 3.30 settled. It is the time to set up its connection to the physical OPC server. Select the server name we have created and in the dialogue „Server type“ select „OPC Server“. In the dialogue „Server definition“ click the radio button „Local“ which means that ASPIC 3.30 will be looking for an OPC server on the same computer where it is running. In real world the OPC server might be running on any other computer in the same network, for such case choose „Network“or „Both“. This is out of the scope of our example. Click the button „Browse OPC server“ then you will have a list of all OPC servers installed on the local computer. OPC servers will be sorted according to OPC server's specification it is running in compliance with, OPC 1.0a by 1.0a, OPC 1.0a by 2.0, and OPC 2.0. Unpack the group OPC 2.0 and choose the OPC server „Merz.OPC_Gen_Modbus_Jbus.1“. Finally Click „OK“.



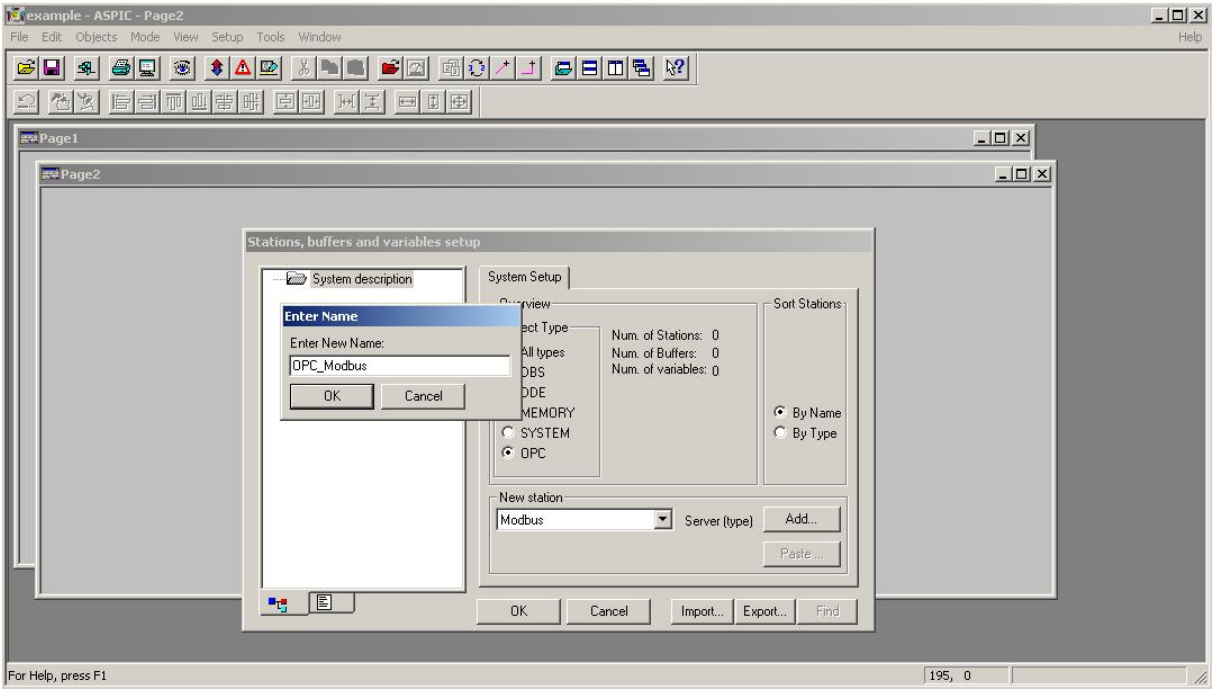
10. The following figure tells that server's setup is done successfully once items „Server“ and „Class ID“ in „server definition” dialogue are automatically fulfilled. Values of these two items are unique for each individual OPC server.



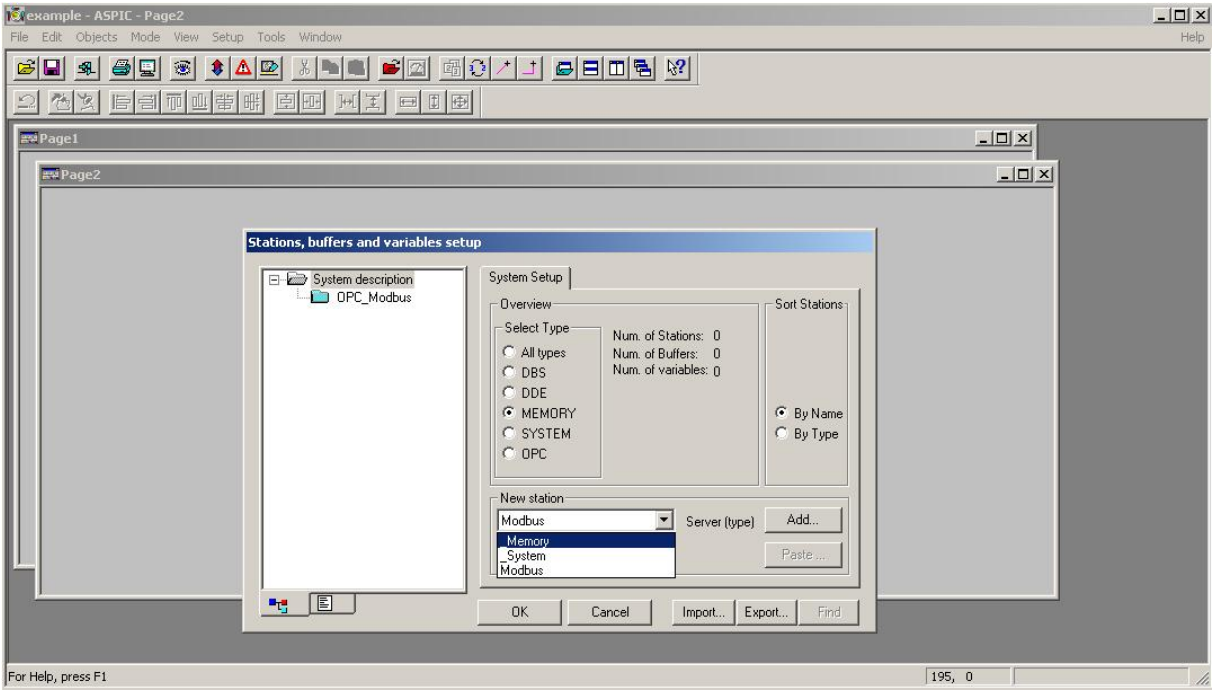
11. We have already established communication with the data source, now we need to create variables in ASPIC 3.30 and interconnect data. Click in tools panel. This dialogue is made for adding stations, buffers, and its variables. Choose your station type „Modbus“ the symbolic name that we have chosen while setting up data source, then press „add“.



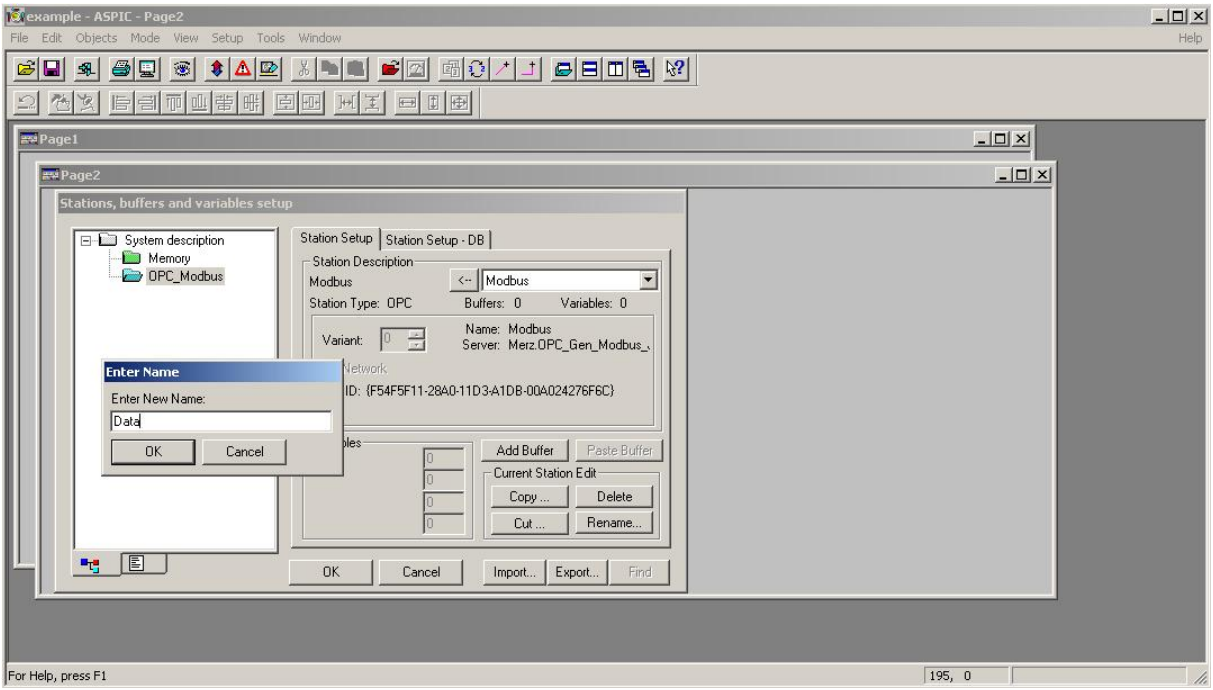
12. In the pop up window write down the new station's name. This could be whatever name of your choice, the best would be a name related to data source. In our example we have chosen „OPC_Modbus". In its sub-items we will add variables from Modbus & Jbus OPC server.



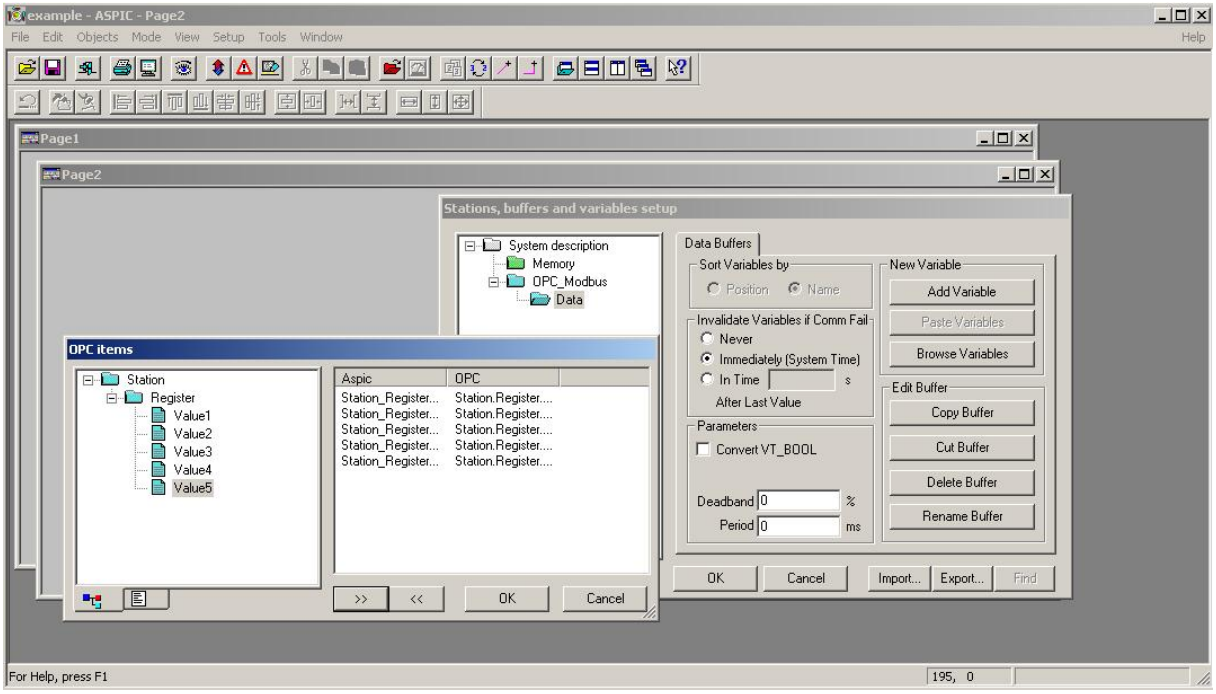
13. In our example and in every high-end application, matters can't be solved without variables irrelevant to data sources (OPC, DDE). These variables will have data essential for the scripting language (ASPIC 3.30 mathematical module). Those are called memory variables. Those have similar organization structure (station, buffer) as in variables interconnected to data sources (OPC, DDE). select new station „_Memory" then click add, write down its name in the pop up dialogue, the name could be so ever name of your choice, the best name would be the one related to data source. In our example memory station will be named „memory". We will fulfill its sub items by variables essential for mathematical module usage.



14. Organization structure of variables in ASPIC 3.30 is as follows, variables are sub items of buffers and buffers are sub items of stations. This organization structure is made for lucidity. We refer to variables using its name only, its location in a certain buffer or station isn't substantial. Create a buffer in the station „OPC_Modbus", by selecting „OPC_Modbus" station then clicking „Add Buffer" and name it „Data".

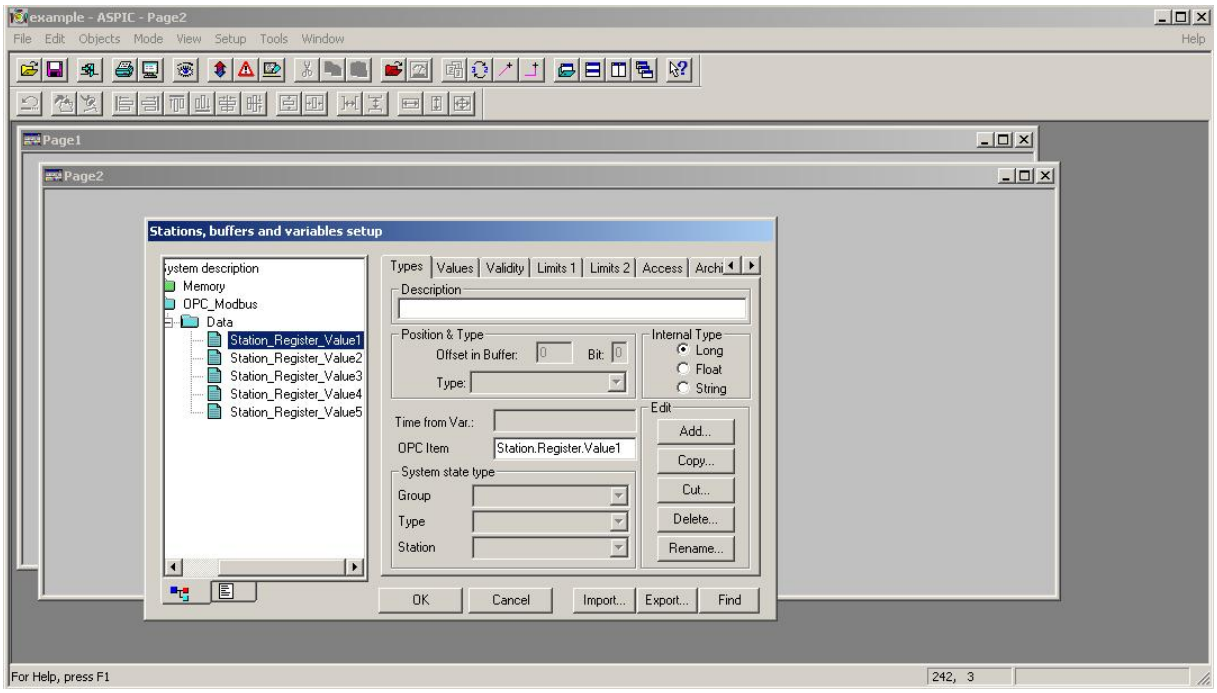


15. Choose newly created buffer „Data“ then press the button „Browse Variables“ in „New Variables“ dialogue, then a new pop up window will raise showing available variables that has been created in OPC server registered configuration, select variables required from the left side dialogue then using „>>” button copy them to the right side dialogue. If by mistake we select a variable and place it on the right side dialogue, the button „<<“will take care of its removal from the list of variables to be added to Aspic. With reference to our example’s matter we will need to add all available variables from „Value1” up to „Value5”. Addition ends by pressing the button „OK“.



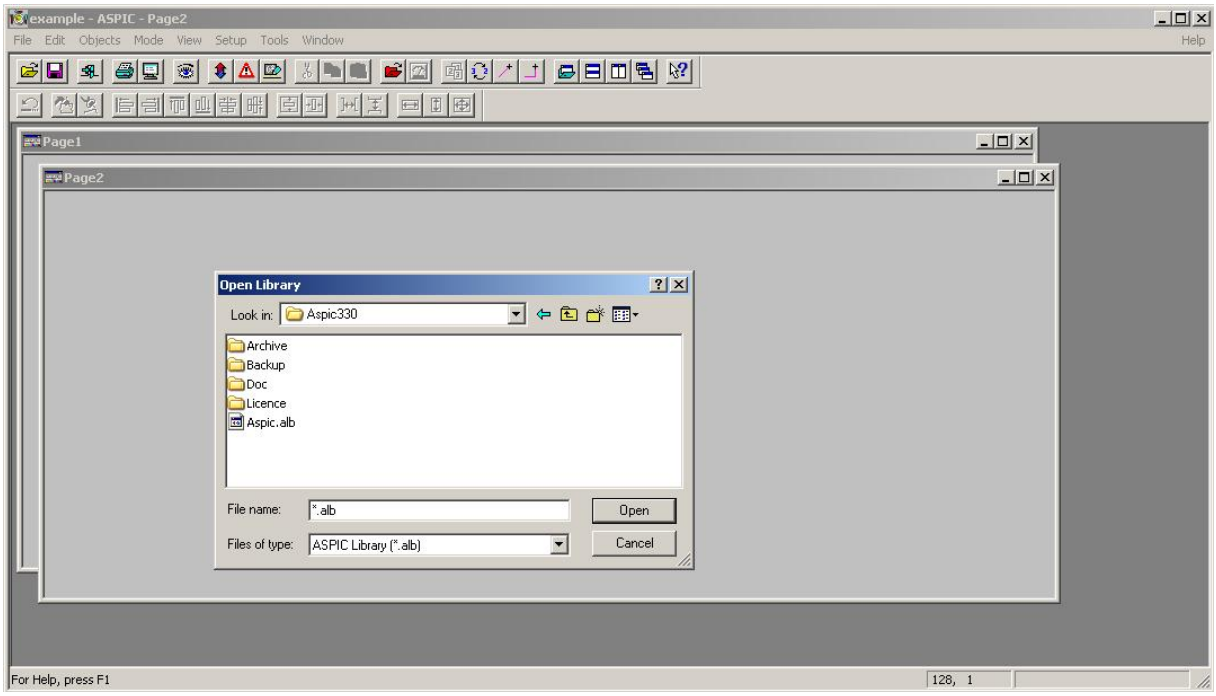
Note: inserting variables could be provided en masse by selecting more variables at once using „shift“key.

16. The state of inserted variables to Aspic 3.30 is shown in the following figure. Inserted variables in Aspic 3.30 will be named according to its configuration structure in OPC server. OPC server works with the structure of Station.Register.Variable. Aspic 3.30 renames inserted variables according to mentioned OPC server configuration structure just by replacing dots by underscore characters, so Aspic 3.30 internal variable name will be „Station_Register_Variable“, this name could be later arbitrarily exchanged. For simplicity we will discuss a concrete variable „Station_Register_Value1“, on the right side dialogue you will get a multiple choice of options placed on a multi-panel dialogue, each panel contains options concerning different functionality properties such as Type, Value, Validity, Archiving, Limits, ...etc. at the moment we keep these values in its default values. Later on we will discuss the setup of variable's properties (Archiving setup).



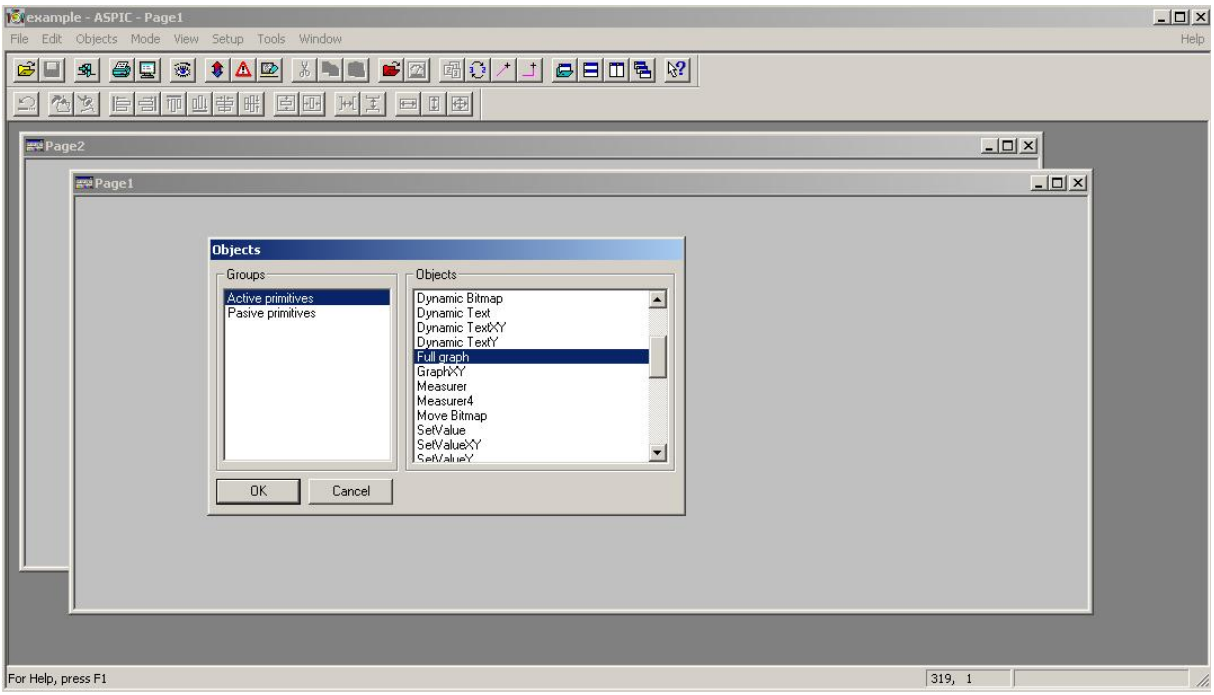
Note: for large projects, and in case that OPC server's configured variables are reasonably named , it would be better to keep the Internal variables names generated automatically by Aspic 3.30, where such names will make it easier to identify each variable's origin, to which station (PLC) it belongs, and which type.

17. It is the time to create the look of visualization pages, in our example we use two visualization pages, both are already created and named „Page1“ and „Page2“. Till the moment no objects are placed on these pages. Objects in Aspic 3.30 are presented as two types, passive and active. Active objects are objects that could be connected to variables, where the state or vision of such object changes proportionally to variable's value change, as example „value“, „full graph“, „button“, ...etc. Passive objects are as example „Frame“, „Text“...etc. Those objects can't be connected to variables and it doesn't change its state or vision while running the visualization. Basic passive and active Visualization objects are included in the library „Aspic.alb“, this library is placed in the folder c:\Aspic330\ (in case that you have used default settings while installing Aspic 3.30). The library could be opened by pressing  icon from tools panel, then choose Aspic.alb, by default it would be placed on „c:\Aspic330\Aspic.alb” finally press the button „Open“.



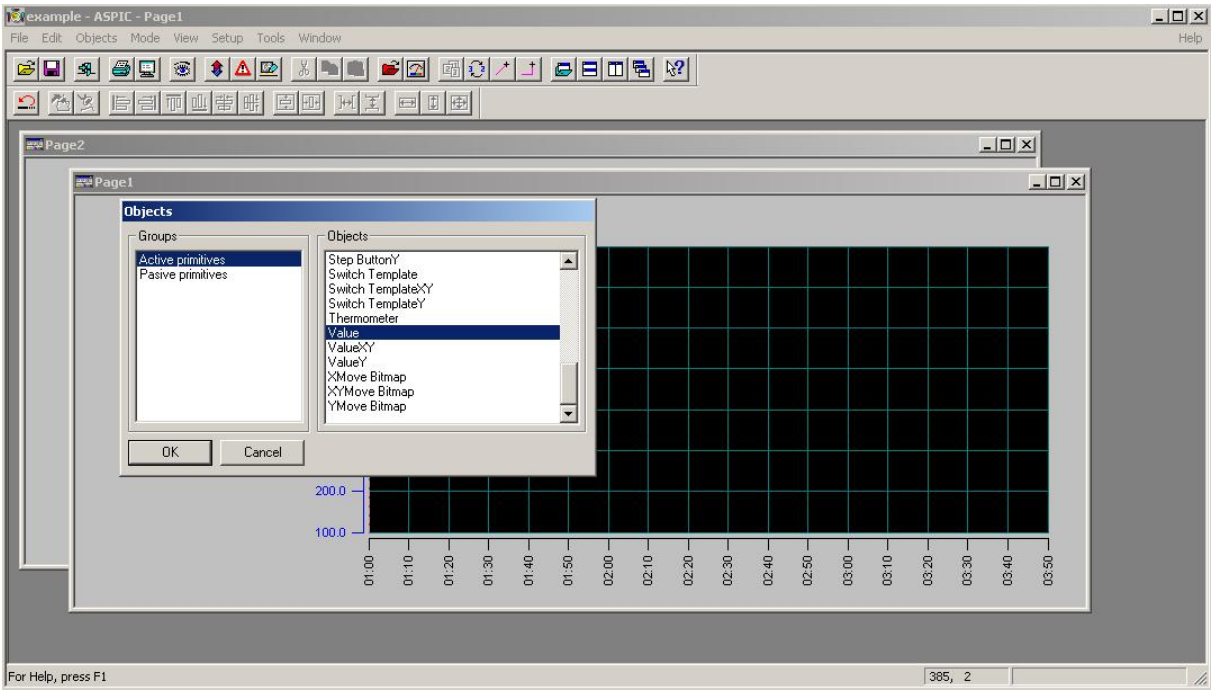
Note: in Aspic 3.30 you have the possibility to create your own objects libraries by assembling available primitive objects. Creating personal libraries are out of the range of this exercise.

18. The following figure shows the content of default library Aspic.alb. In the left dialogue „groups“ you can choose passive or active primitives, note the result of your choice in the right dialogue „objects“ where you will be able to select an object of your choice out of the primitives. In this exercise choose „Active primitives“ and the object „Full graph“. Place the object on „Page1“by pressing the button „OK“.

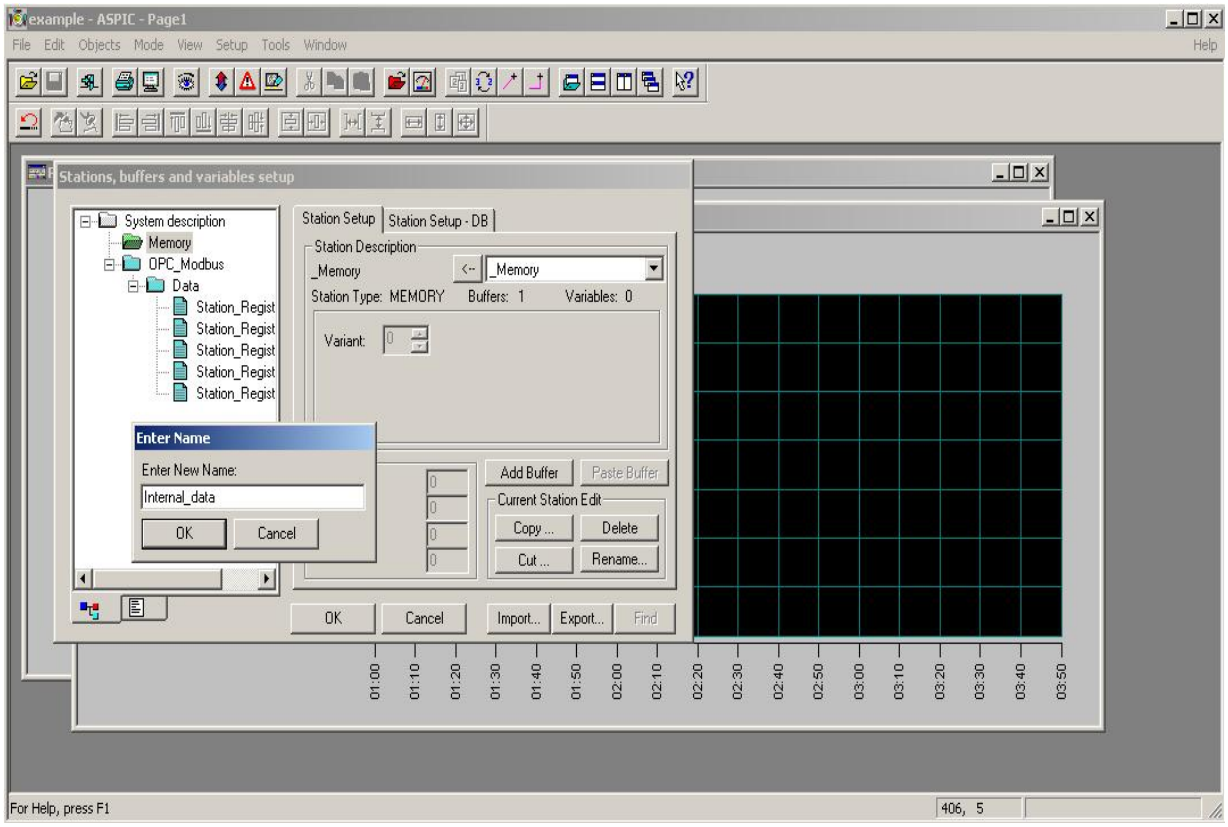


Note: chosen object will be placed on the page that has been active before opening object‘s library.

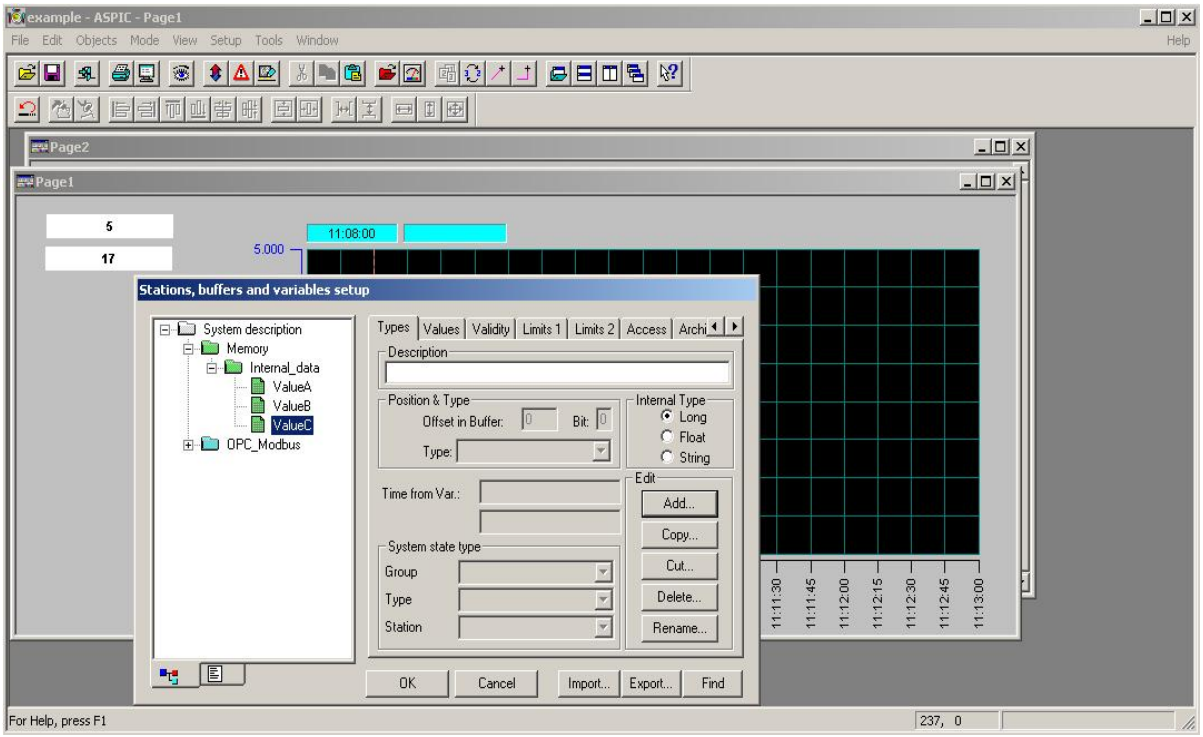
19. Conformably to your needs place and extend the graph object on the visualization page. Note the icon which tells us that objects library has been selected and opened, this icon appears on the tools panel, on the right side of objects library icon. By clicking the icon you open the currently selected library. In our case currently selected library will be Aspic.alb (as in previous point). From the list of active primitives objects select „Value“ and place it on the visualization page „Page1“ by clicking the button „OK“. Totally place two „value“objects on „Page1“.



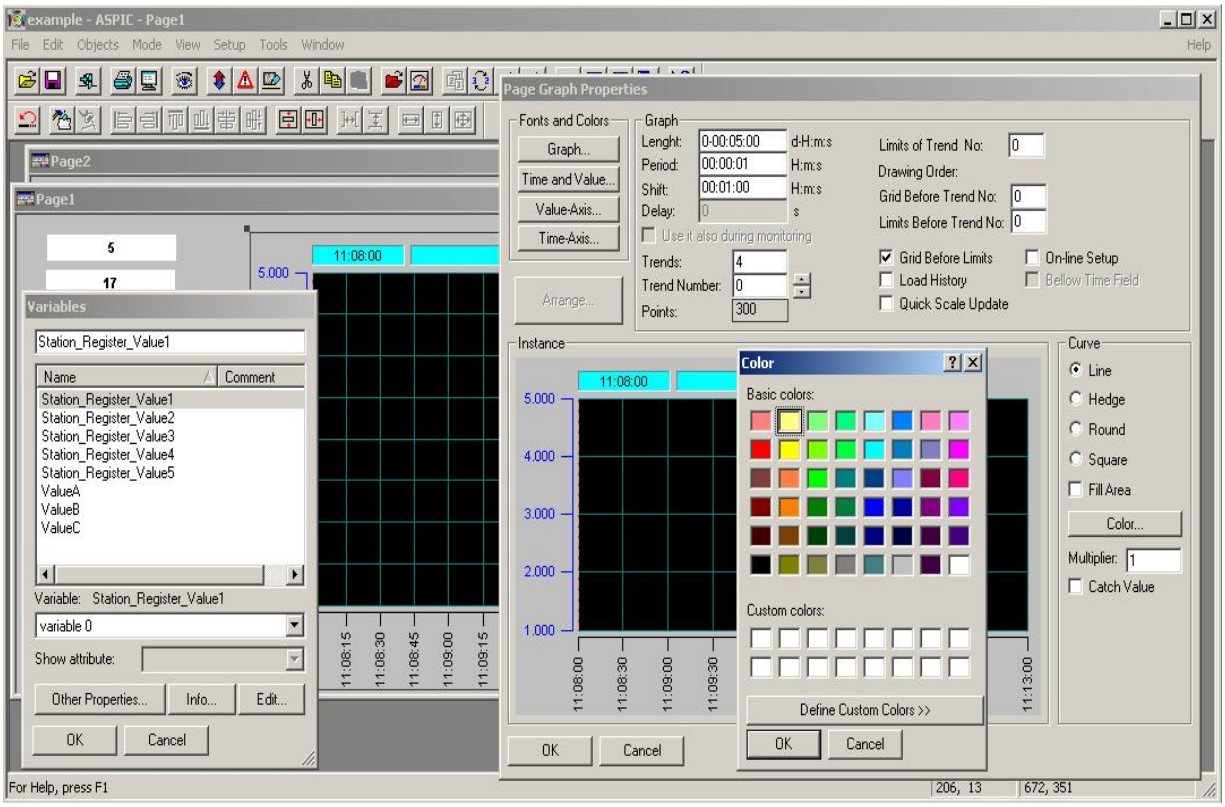
20. At the moment we have got three objects placed on „Page1“, two „Value“ objects and one „Full graph“ object. According to our exercise’s task, the graph object should chart three variables (Value1, Value2, ValueA). „ValueA“ is a memory type variable. In previous steps we have created the „Memory“ station, in which we will create the buffer „Internal_Data“.



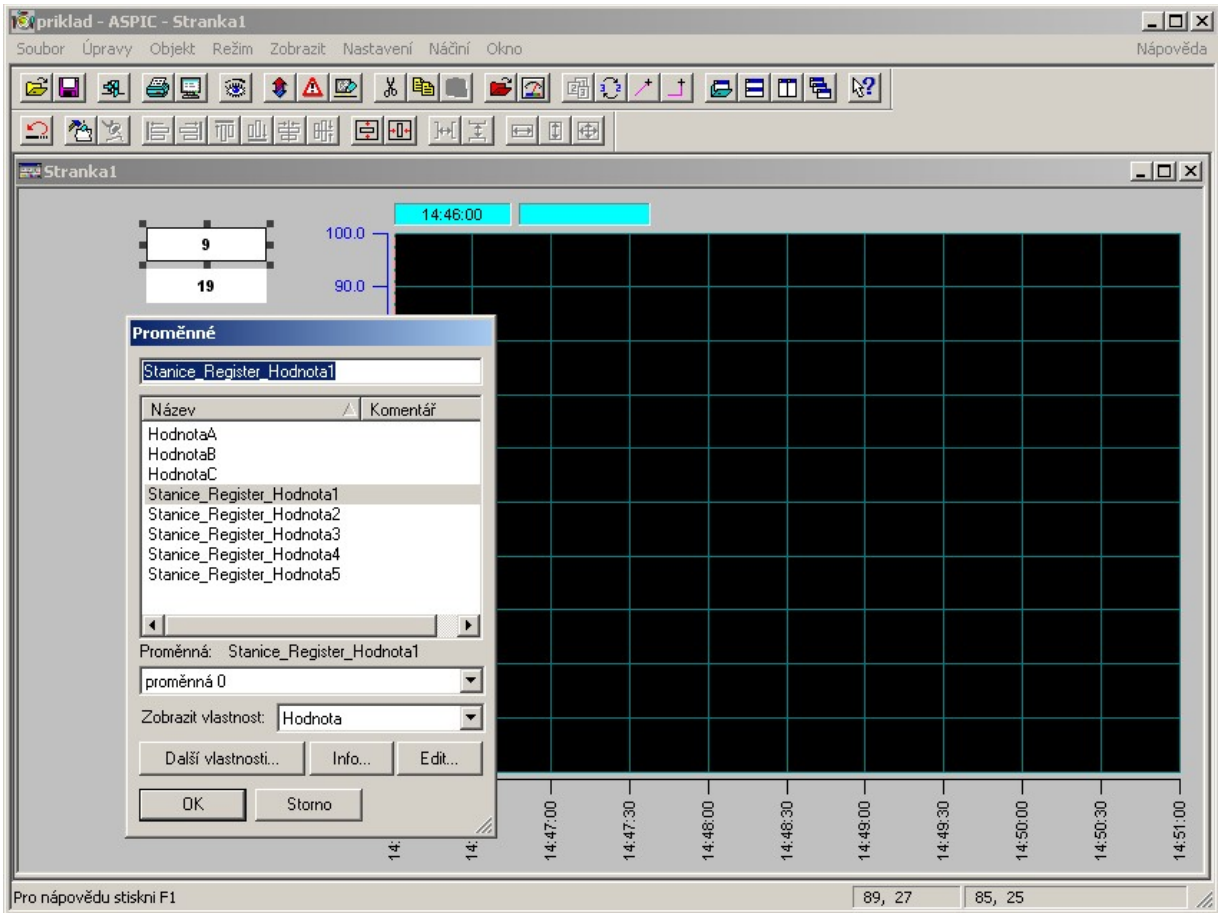
21. In the buffer „Internal_data“, create three variables with the following names „ValueA“ till „ValueC“. As long as we are planning to give „ValueA“ the values of a sine wave function, then we will setup the property „Internal type“ to „Float“ !. The properties of other memory variables we keep by default.



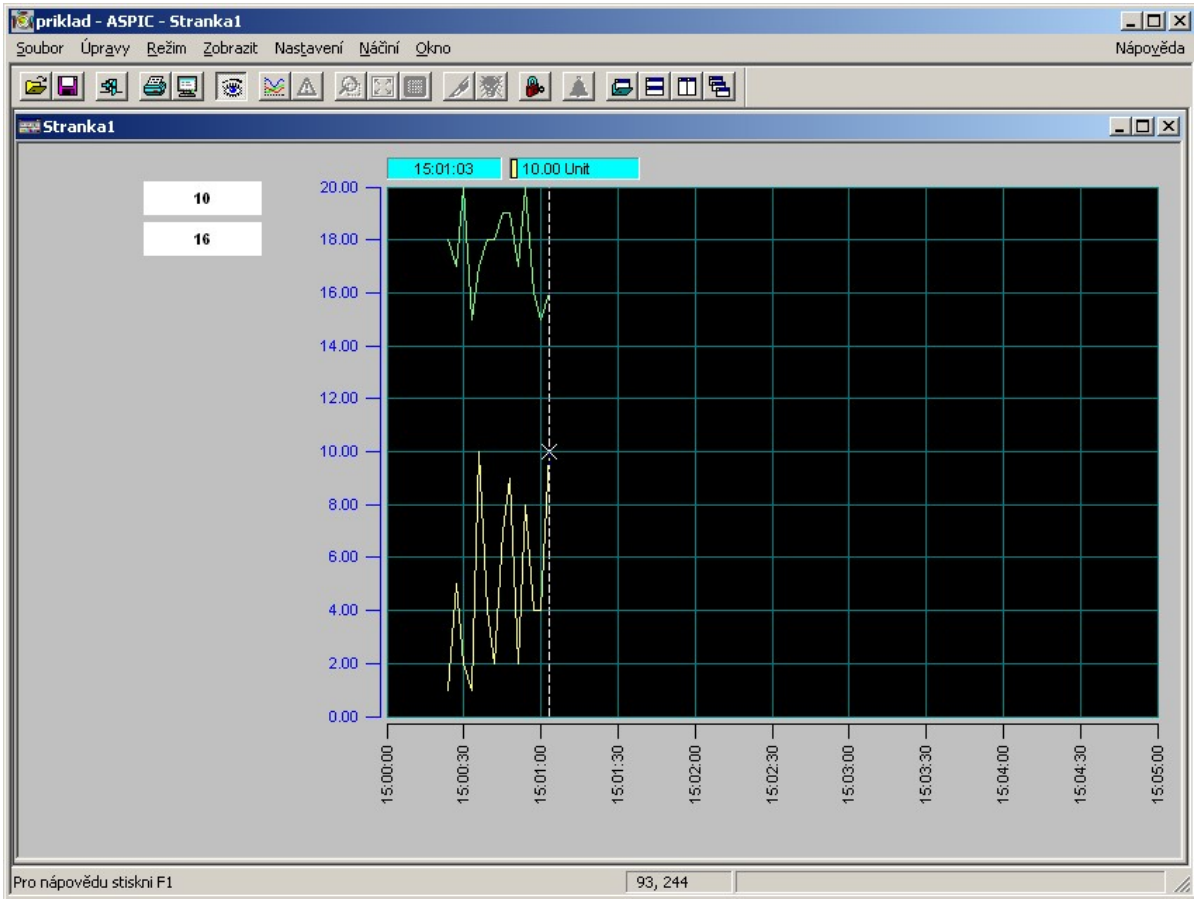
22. It is the time to connect configured variables to already created objects placed on the visualization pages. Double click the graph object, the „Variables dialogue“ will open. Before connecting variables to graph we need to do some configuration in „Page graph properties“ we get there by clicking „other properties“ button from variables dialogue. According to exercise scenario, we need to chart four variables, so the property („Trends“= 4). Exchanging the properties of each of these trends will be possible by selecting an individual trend (0 to 3) in „Trend Number“ property. Select the trend 0 and click the button „Color“ and from the pop up window choose a color for our 1st trend. Similarly set up a color for each trend. Each setup we finish by clicking the button „ok“. still remains connecting these trends to variables, and that would be achieved in variables dialogue where we have ended up, in this dialogue, for „variable 0“ select „Station_Register_Value1“, similarly „variable1“ select „Station_Register_Value2“, follows „variable 2“ select „ValueA“, then „varieble 3“ select „ValueC“. Finally we end up the dialogue by clicking „OK“.



23. As has been shown in the previous step, we will connect both value objects on the visualization page „Page1“ to variables. Double-click the first value object and in „Variables“ dialogue interconnect „Variable 0“ to the variable „Station_Register_Value1“. Similarly interconnect the second „Value“ object to „Station_Register_Value2“.



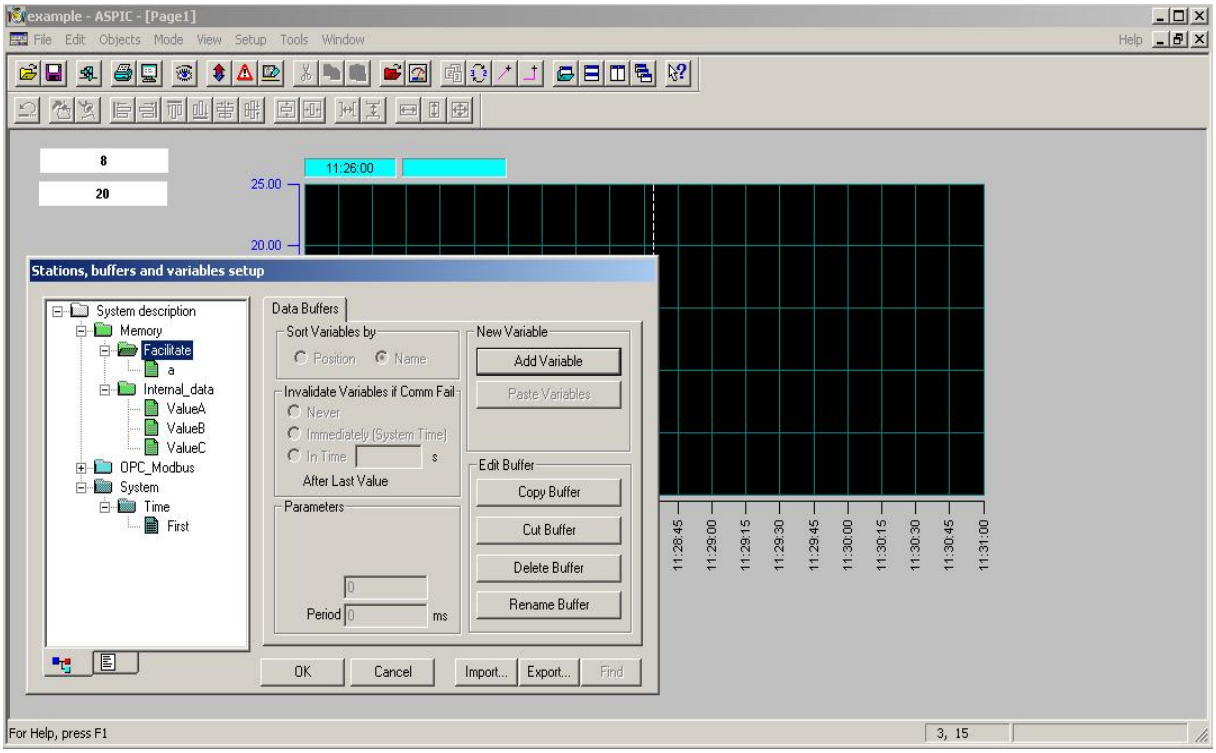
24. Even though the project didn't yet fulfill all exercise steps, we are able to run it. Running will be successful in case that all settings and setup parts has been done correctly. Please note that only two trends are visible in the graph (charts for Station_Register_Value1, Station_Register_Value2) the remaining two trends (charts for ValueA and ValueC) are not visible yet because their values are not initiated and still not assigned to any function. The value objects will show values of variables (Station_Register_Value1, and Station_Register_Value2). The visualization is started and stopped by clicking icon from tools panel, or by using the shortcut (CTRL+I).



Note: if values in „Value" objects are changing rapidly (more quickly than once per three seconds) then that is happening because you have forgotten to set up the „resp. Time "while configuring the variable in the OPC server. Please refer back to step number 7 of this exercise and setup the response time to 3000 ms, this setup in this example is provided for inspection reasons.

25. Open the dialogue for „Stations, Buffers and variables setup“ (see the step No. 11 of this exercise). Add a station of the type „_System“ name it „System“ and in this station create a buffer and name it „Time“. In „Time“ buffer create a variable and name it „First“ set up its type to „first cycle“. What for is this variable?. Aspic 3.30 sets „first cycle“ value to (true) only in the moment of first scripts program execution (during the period of the first loop). Scripting module in Aspic 3.30 is periodical, (the period could be configured, in our example we will keep the default configuration, that will allow the program to run „as much as possible“). Initializing memory variable's value is an essential action in most of visualization projects. Using the system variable „First“ will make such initialization possible, (during first cycle we will set initial values to memory variables).

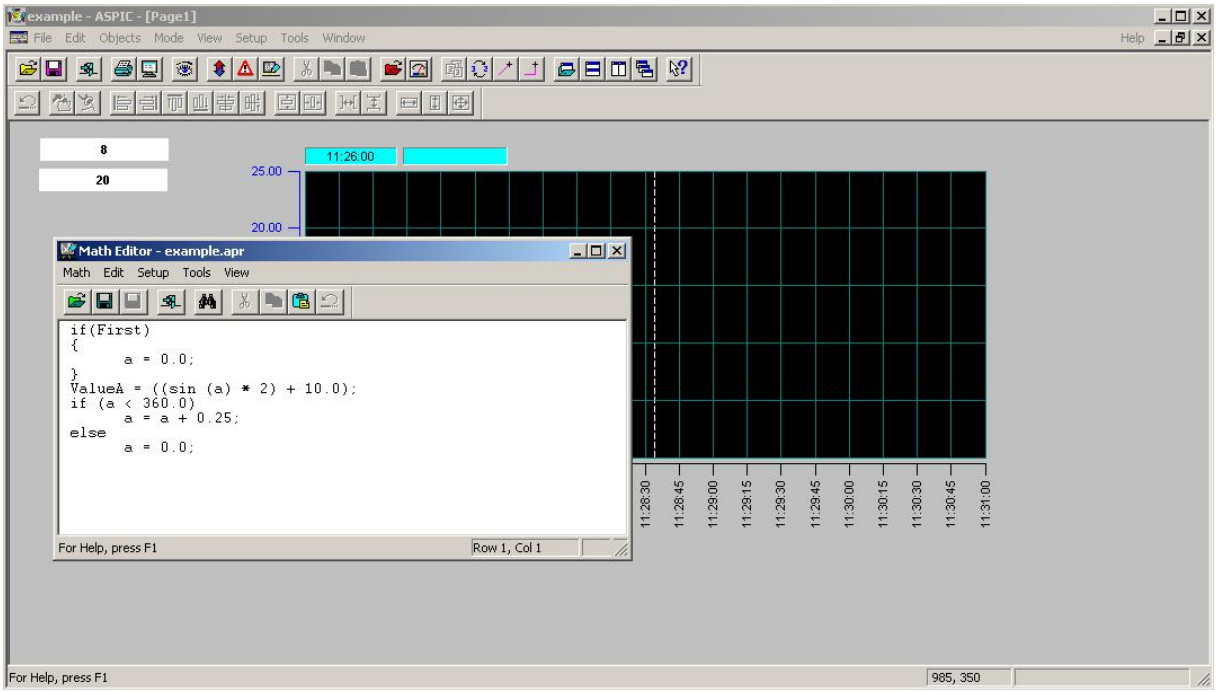
To already existing memory station „Memory“ we will add a new buffer called „Facilitate“, in „Facilitate“ create a new variable, name it „a“, and set its internal value to „float“ (why? This will be explained in the following steps), the variable „a“ we will need while writing the program in Aspic 3.30 mathematical module.



26. This step shows how to assign a sine wave function values to the variable „ValueA“. First of all run „Math editor“ by clicking on the icon , then insert the following code.

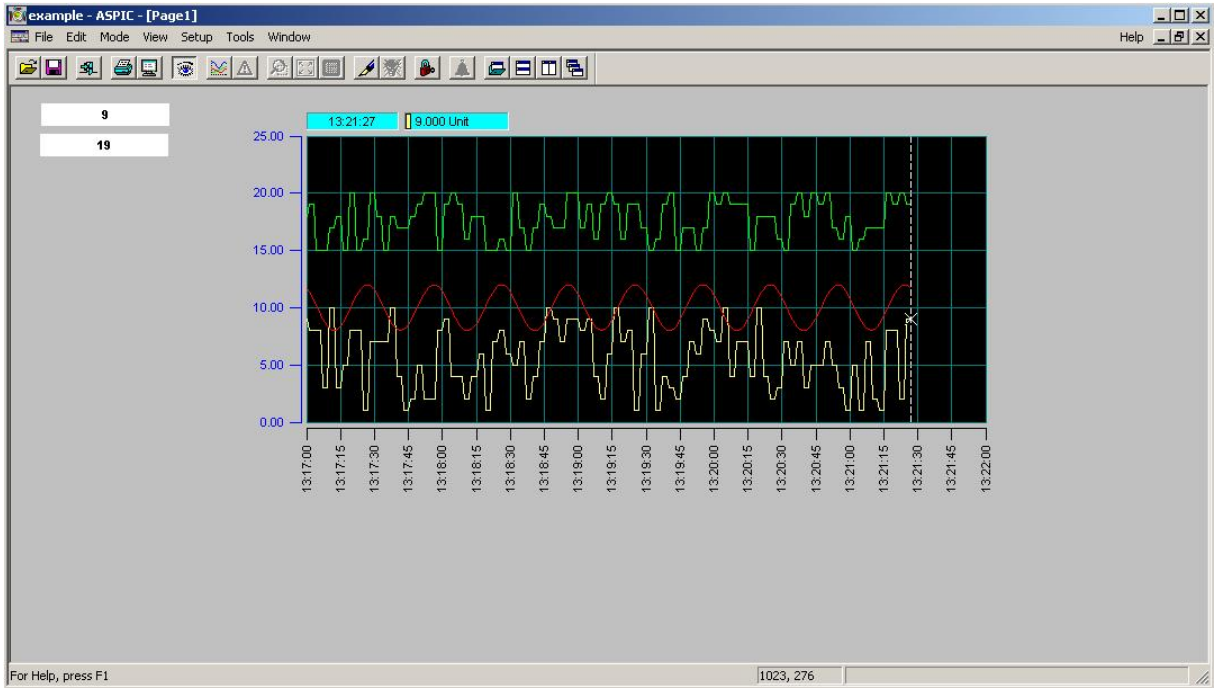
```
if(First)
{
  a = 0.0;
}
ValueA = ((sin (a) * 2) + 10.0);
if (a < 360.0)
  a = a + 0.25;
else
  a = 0.0;
```

While math module in its first cycle the memory variable „a“ will be initialized, where its value will be assigned to „0“, and a sine wave function based on „a“ value will be computed, where its result will be assigned to the memory variable „ValueA“ .In each other cycle where „a“ is smaller than „360“, „a“ variable’s value will be increased by „0.25“, sine wave function based on „a“ will be computed and its value will be assigned to „ValueA“, once „a“=360, „a“ value will be assigned to „0“ again and a sine wave function of that will be computed and its result will be assigned to „ValueA“. Save the code and close „Math editor“.

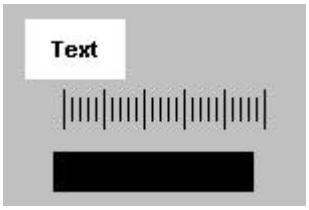


Note: if mathematical program cycle period is not set up to a certain value, then the shape of out coming trends will differ because of its relativity to computer’s performance.

27. Run the Visualization by clicking the icon . A sign wave will be charted in the graph.



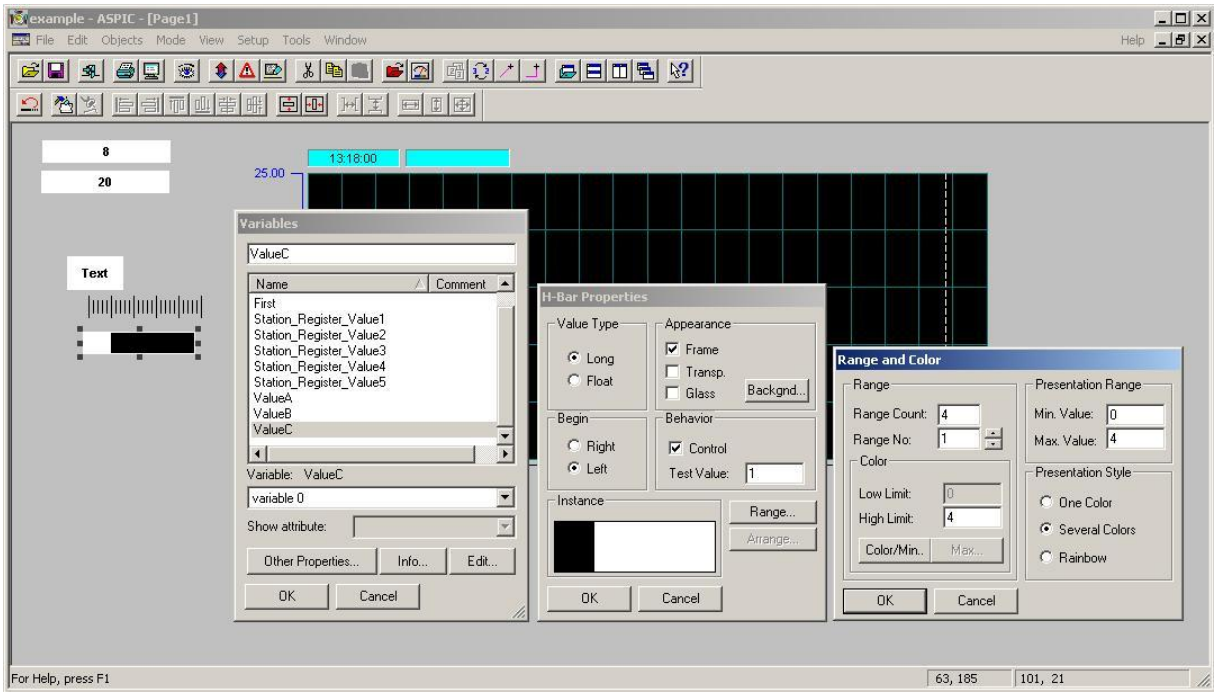
28. We will improve the project by placing more objects to visualization page „Page1“. Insert an active object „Bar_horizontal“, passive object „scale_horizontal“ and one „Text“ passive object. Configure text object then copy and paste it two more times, in total we should have three text objects. The following figure illustrates inserted objects.



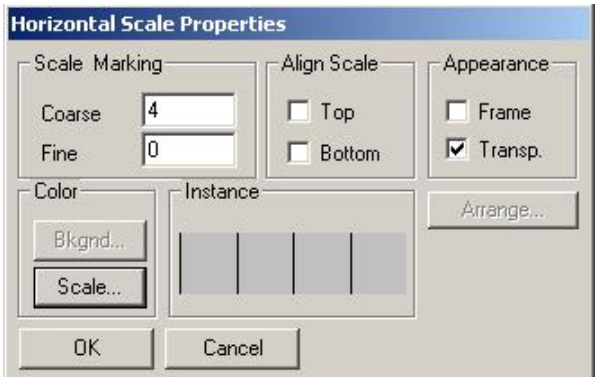
Now setup the properties of „Bar_vertical“, we will set up this object in a way that allows us to monitor and control the value of memory variable „ValueC“. Variable’s value should be integer from 0 to 4. In appearance dialogue set the „background“ color to white. Then click the button „range“ and set up „Range count“ to 4, set „Min. Value:“ to 0, and „Max. Value: “to 4, and „presentation style“ to „Several colors“.

For each „Range No:“ you can configure High limit, low limit, and a color. For the range „1“ set low limit to 0, High limit to 1 and black color. For the range „2“ set low limit to 1, High limit to 2 and black color, similarly set up the other limits and keep the black color.

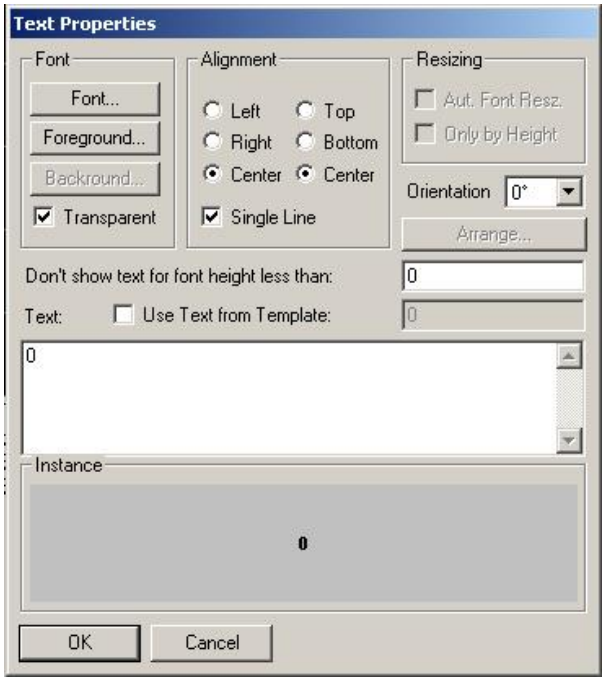
Connect the object „Bar_vertical“ to a varieble „ValueC“.



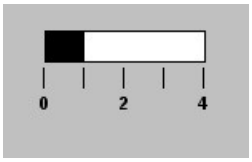
Set up passive object „scale_horizontal“ properties according to the following figure.



Set up passive object „Text“ properties as shown in the figure here under. Using this object we will create numbers that will be used with „scale_horizontal“ object. Choose transparent background and in the „Text:“ area insert 0. click OK., Object twice copy and paste using (CTRL + C, CTRL + V) and in pasted text objects change the text to 2 and 4.



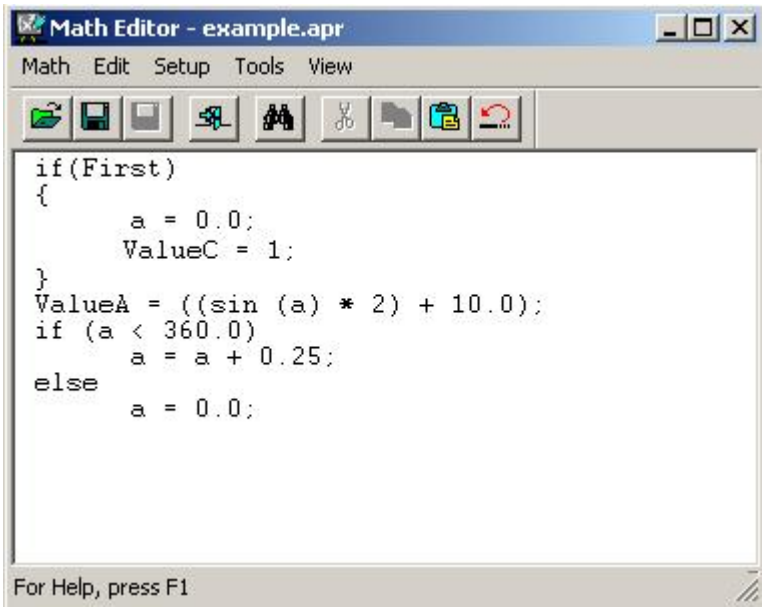
Now change object’s position and shape, according to the following figure. Aspic 3.30 enables assembling new object’s groups and its placements in user’s libraries, where it could be used in a user friendly way. User’s libraries are out of the range of this exercise. Please look for that in Aspic 3.30 user’s manual.



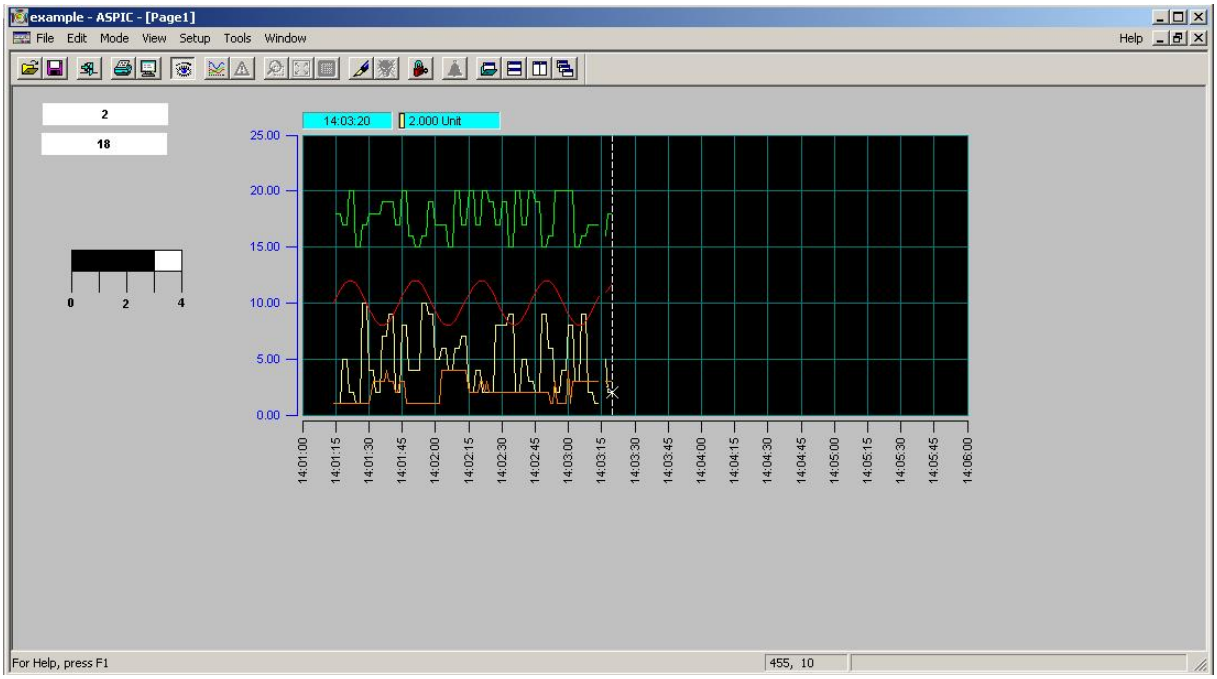
Don’t forget initializing memory variable “ValueC”. In the mathematical module’s code, place the following line.

```
ValueC = 1;
```

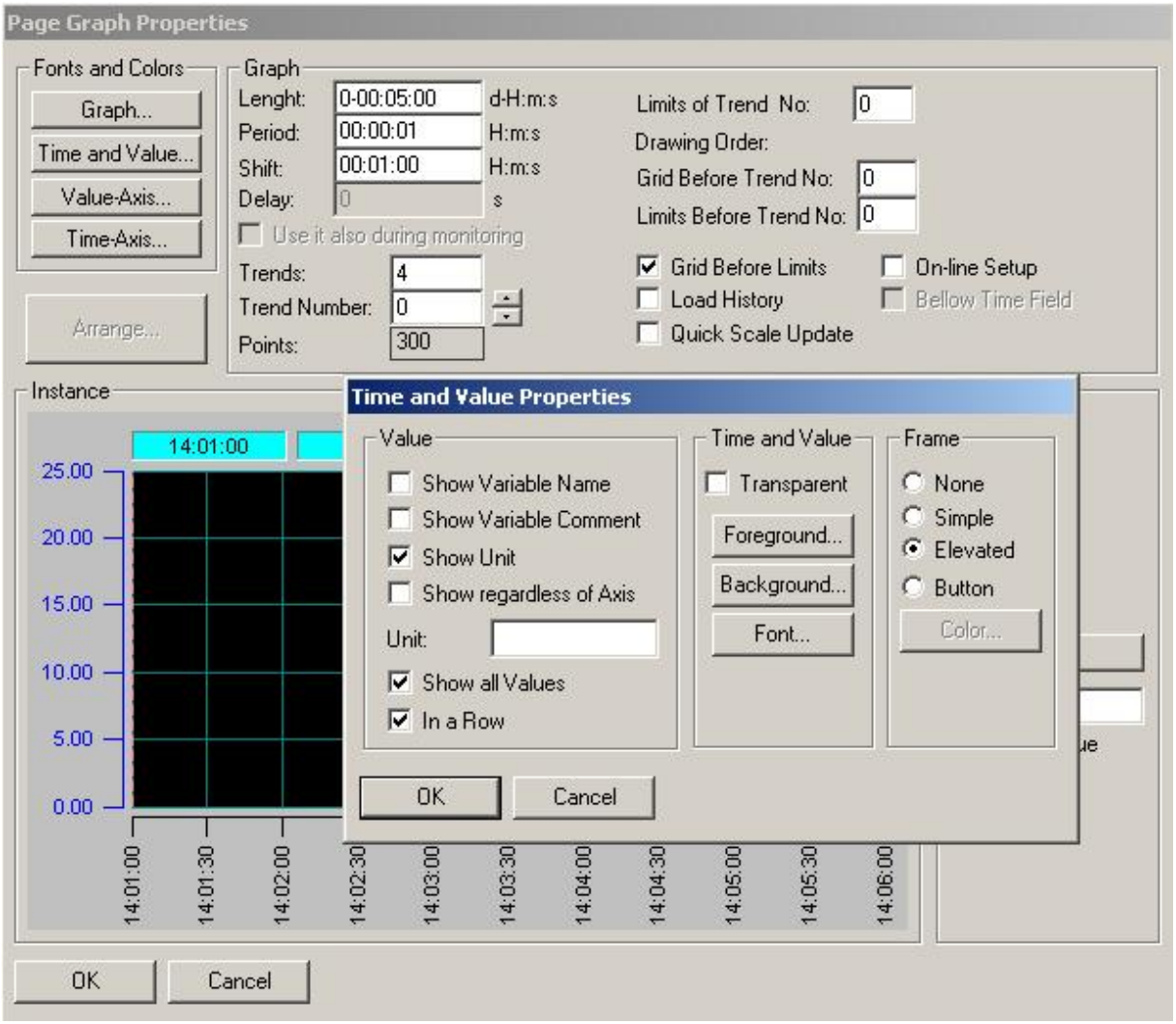
Add above mentioned line to „first cycle“ section, in order to have its value initialized to 1.



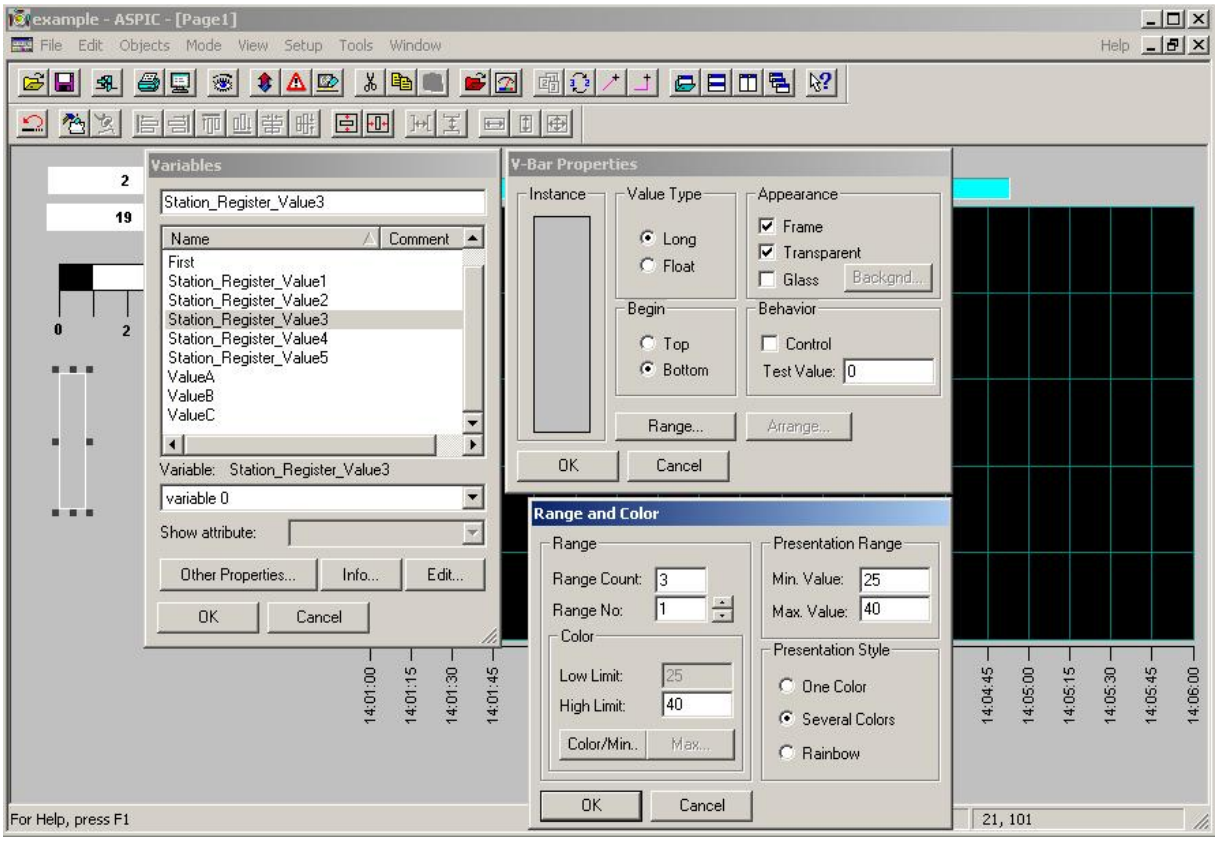
Run visualization, and by a mouse click on the object „Bar_horizontal“ change the value of „ValueC“ variable. You can clearly notice the change charted in the graph.



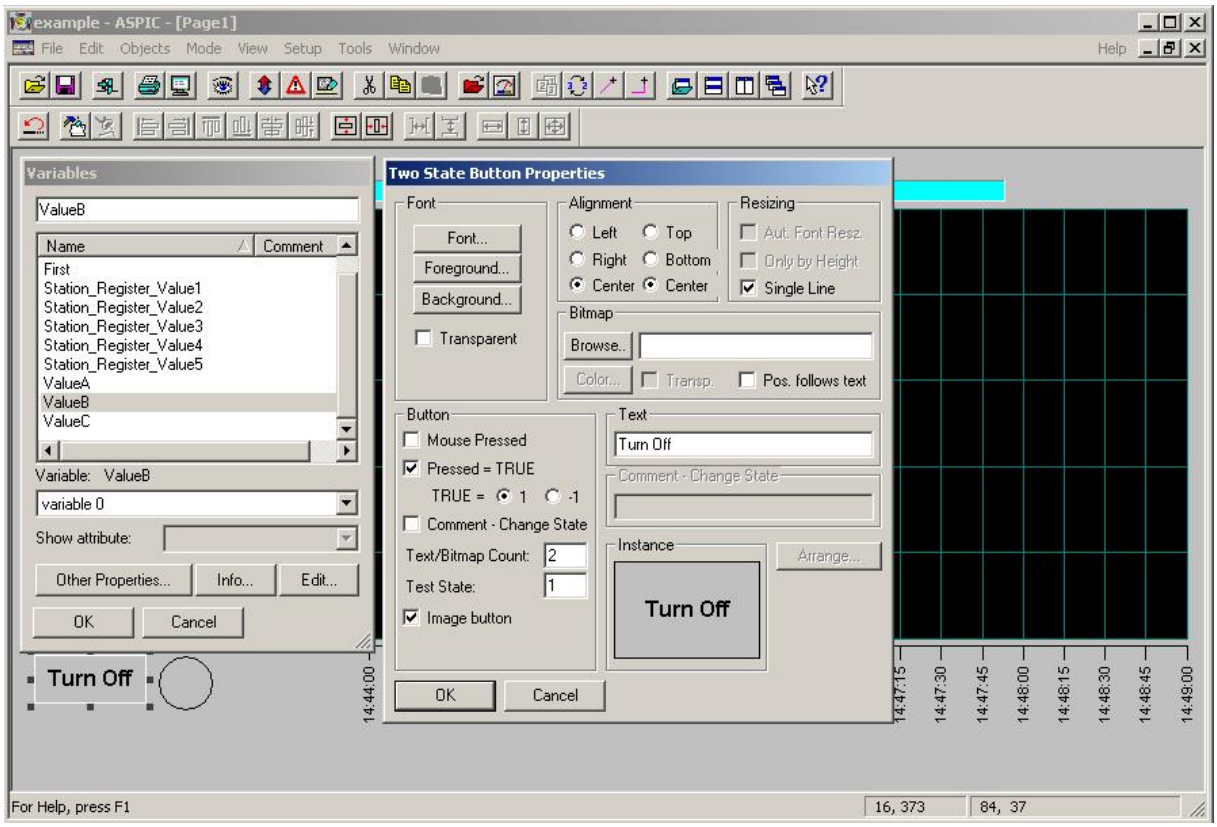
29. This step is about configuring further graph properties. Display „Page Graph Properties“ window, then click „Time and Value“ from „Fonts and Colors“ Dialogue. Since we are not using a quantified variables in the graph, then we keep the „Unit:“ item clear, and sign the check box „Show all values“ and „In a raw“ will automatically be selected. All variable’s charts will be visible in the „Graph“ object.



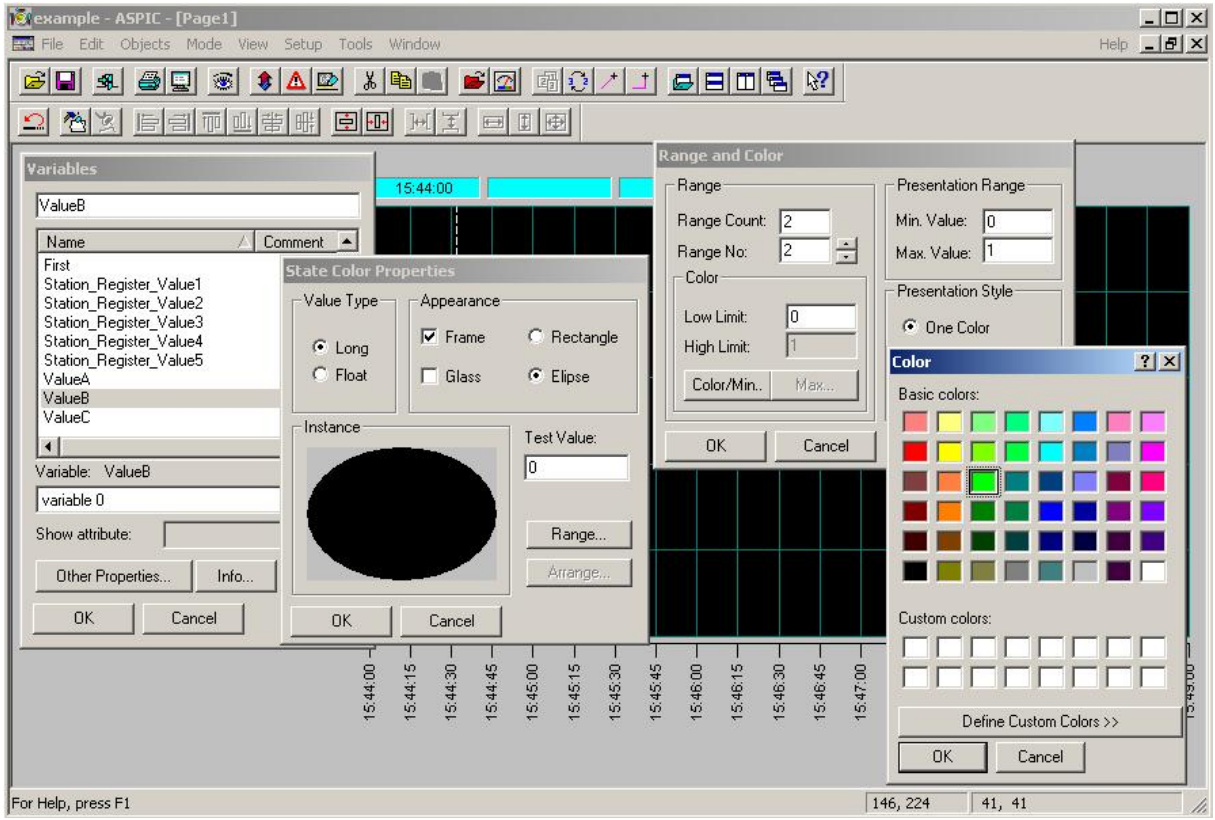
30. Add the active object „Bar_vertical“ to visualization page „Page1“, connect it to variable „Station_Register_Value3“, and configure its properties according to the following figure. for better example‘s plasticity set up presentation range to 25 and 40. The value of connected variable will vary in the range of 30 to 35.



31. Add the active objects „2State button“ and „State Color“. In this exercise we will simulate a button controlled lamp. Both objects will be connected to the memory variable „ValueB“. ValueB is memory variable so please don’t forget its initialization, (see step 28.) initialize it to value 0. In „2State button” create two text captions. For value 0 text caption will be „Switch On“, and for value 1 text caption will be „Switch Off“.

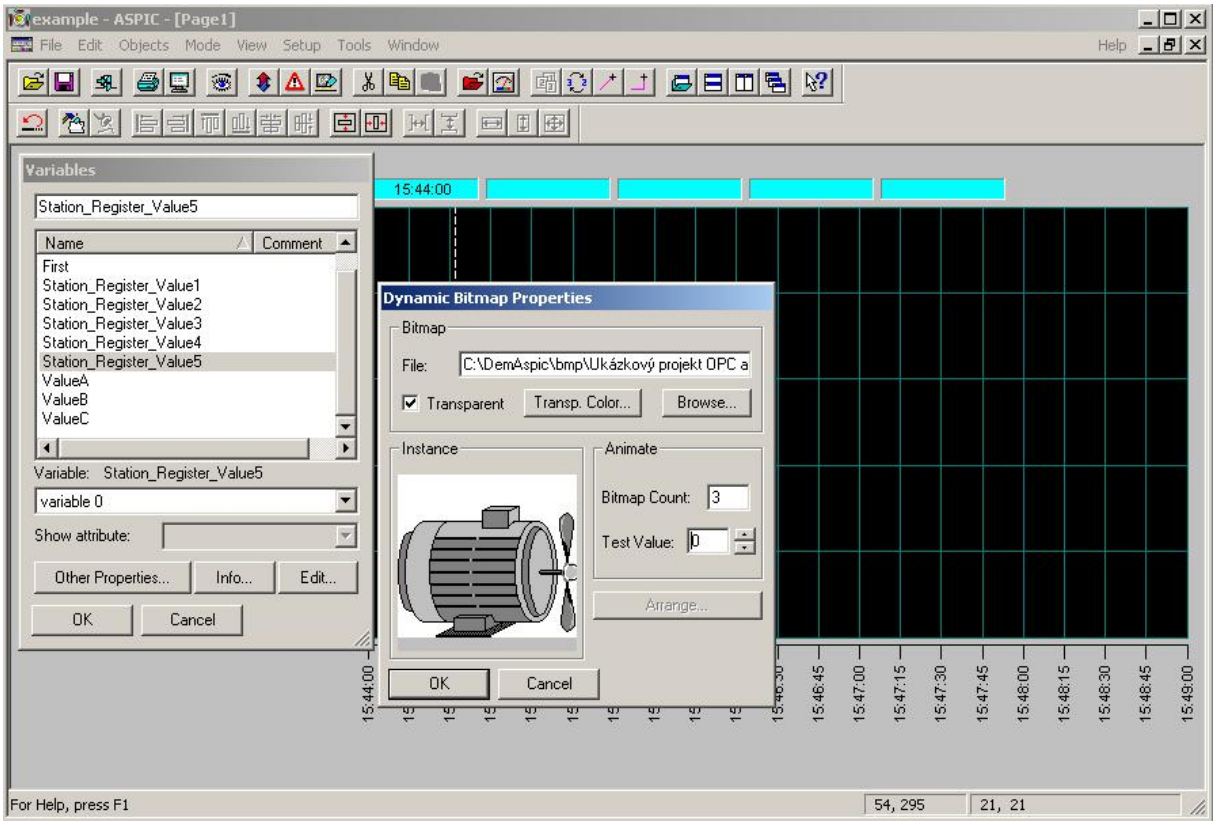


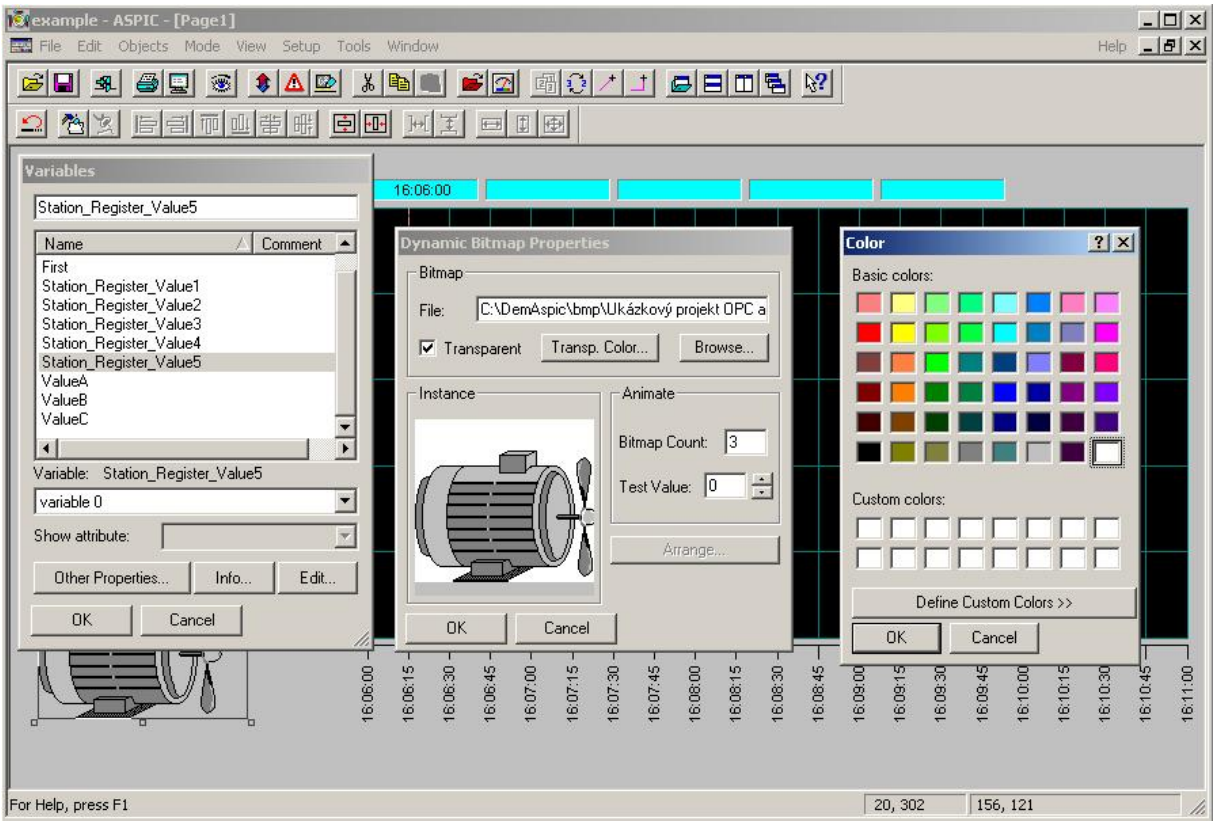
Set in the object „State Color“two range counts 0 and 1. For range no. 1 set a black color (Switched Off) where low and high limit will be 0 and 0, and for range no. 2 green color (Switched On) where low and high limits will be 0 and 1. This will simulate a lamp.



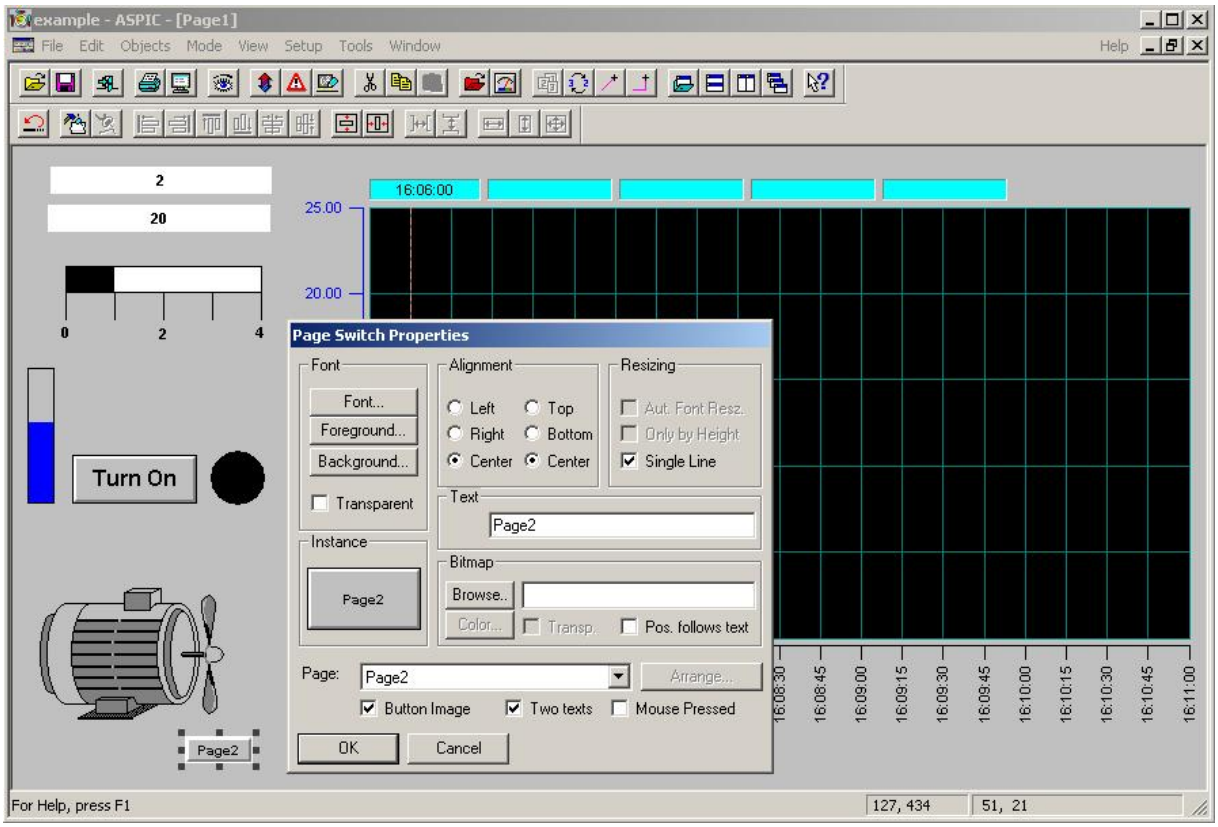
Confirm the result of each step by running the visualization.

32. Here we show how we can animate equipment's state, as example a motor. In this example we will use pictures that you can find as a part of demo project. Insert an active object „Dynamic bitmap" to visualization page „Page1"and connect it to variable „Station_Register_Value5". Its property „Bitmap Count" set up to 3, and for each state select a picture. For state „0" a photo showing a motor in stand mode, For state „1" a photo showing a motor in run mode, For state „3" a photo showing a motor failure. On the visualization page we will see the picture related to the current value of connected variable.





33. On the Visualization page „Page2“ add an active object „Full graph“, connect it to a available of your choice and reconfigure its properties. You can as example change graph „length“, and „trends“. Further add to each visualization page the passive object „Switch page“. Change its properties in each page in a way that switches to the other one. Confirm project’s functionality by running the visualization.

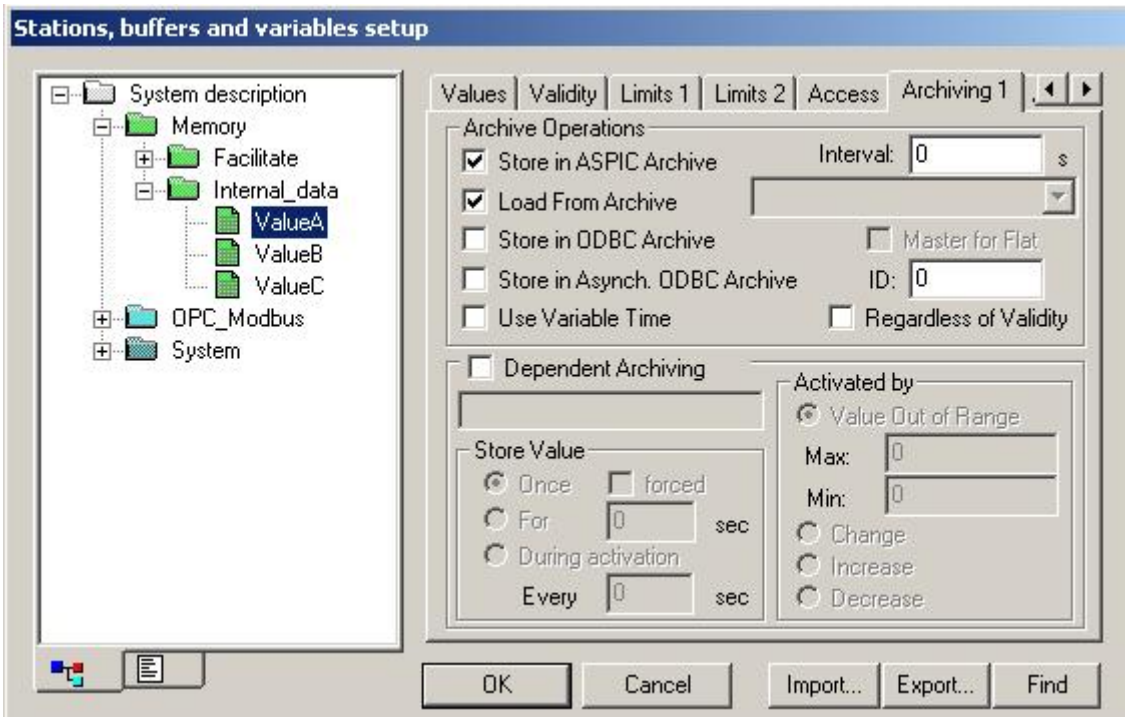


Note: In Aspic 3.30 development mode you can switch between visualization pages using menu item „Window“.

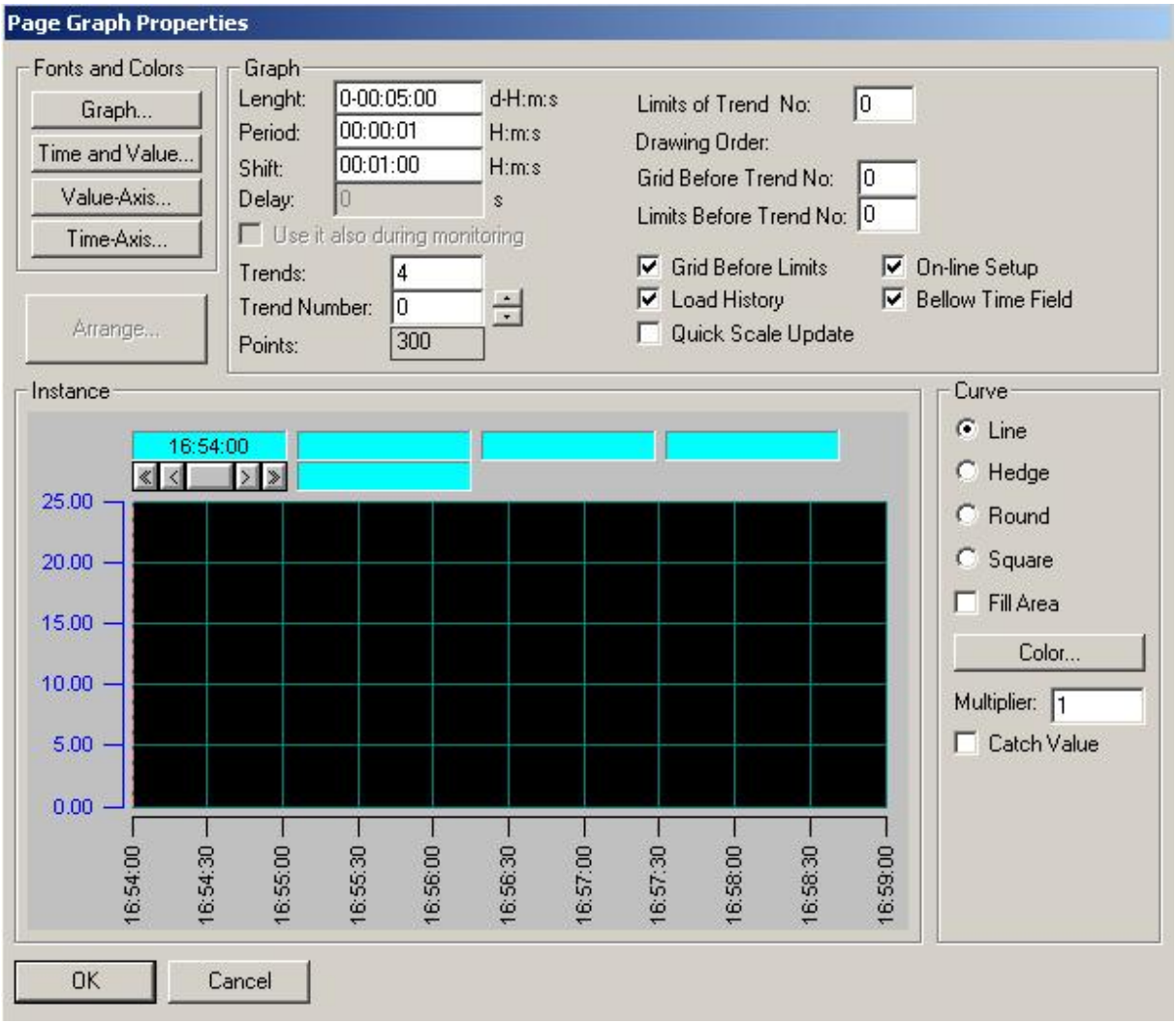
34. At this moment we have got all required objects placed on visualization pages, objects are connected to proper variables and shows correct values. The task of this exercise is to archive data from all variables „Station_Register_Value1" till „Station_Register_Value5" and „ValueA" till „ValueC". Aspic 3,30 is capable to archive to different resources, in this example we will show the most simple method for data archiving, which is archiving „to file" (ASPIC Archive).

    From main menu choose Setup -> Options then in „Archives" dialogue sign the check box „Create_archive".

Further in „Stations, buffers, and variables setup" window choose the dialogue „Archiving1" in this dialogue sign „Store in Aspic Archive" and „Load from Aspic Archive" check boxes. This setup we do for each and every variable we wish to archive.

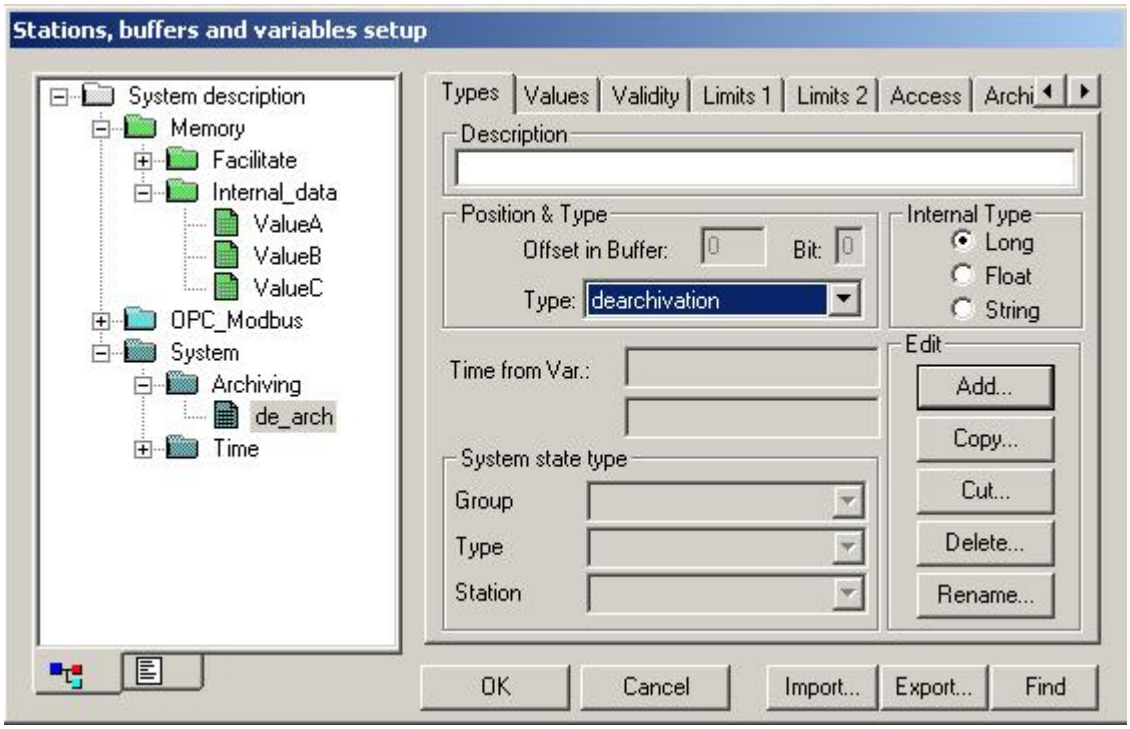


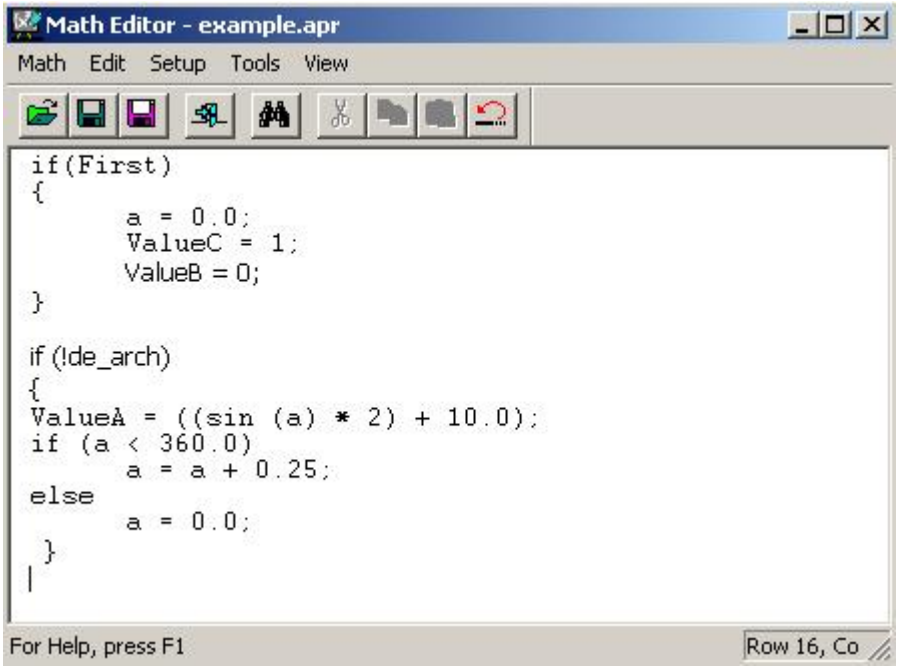
In order to automatically retrieve values from archive (to be charted in the graph) while visualization is running, sign the check box „Load history'in the dialogue „Page graph properties". This setup is global for the graph and it includes all variables charted in this graph.



Now run visualization, keep it running for a while (approximately 10 seconds), then restart the visualization. If you have correctly done the setup, then the graph will de-archive (retrieves) historical variable's values (data).

Note: some variable's values will not be de-archived. Such as „ValueA", because values of this variable is computed in mathematical module, when running the visualization de-archived data for such variables are overwritten by actual data computed. For troubleshooting this problem create a new system variable in the station „System" (system type) name it as example „de_arch" and select the type „Dearchiving". This variable will take the value true just while de-archiving. Further it would be enough to bind the computation of „ValueA" variable's value by the value of the variable „de_arch". See the figure here under. Try this correction.

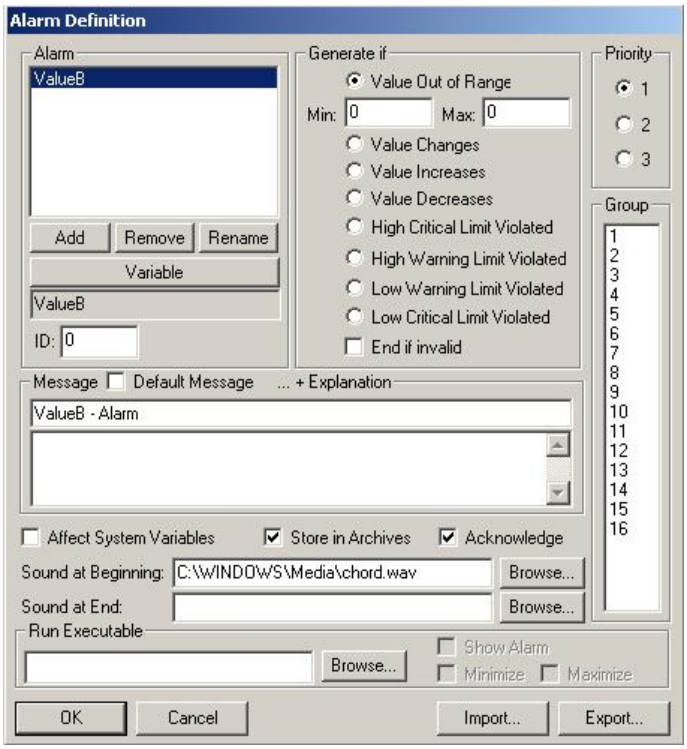

Stations, buffers and variables setup
System description
Memory
Facilitate
Internal_data
ValueA
ValueB
ValueC
OPC_Modbus
System
Archiving
de_arch
Time
Types
Values
Validity
Limits 1
Limits 2
Access
Archi
Description
Position & Type
Offset in Buffer:
Bit:
Type:
dearchivation
Internal Type
Long
Float
String
Time from Var.:
Edit
Add...
Copy...
Cut...
Delete...
Rename...
System state type
Group
Type
Station
OK
Cancel
Import...
Export...
Find


Math Editor - example.apr
Math
Edit
Setup
Tools
View
if(First)
{
a = 0.0;
ValueC = 1;
ValueB = 0;
}
if (!de_arch)
{
ValueA = ((sin (a) * 2) + 10.0);
if (a < 360.0)
a = a + 0.25;
else
a = 0.0;
}
For Help, press F1
Row 16, Co

35. Our task in this exercise will be creating an alarm system. We will see how to define alarms and archive them for reporting purposes.

    According to our example we have to create alarms that need an acknowledgement from the operator and only one of the alarms will be audible. In Aspic 3.30 alarms are defined in „Alarms_definition“ window, we get this window by clicking the icon from tools panel.

    a) We will define an audible alarm for the variable „ValueB” once its value is active (1), because „ValueB“ is a variable its value is controlled by a button placed on „Page1“.This will make functionality testing much easier. Open the window „Alarm_definition” and in alarm dialogue add new alarm call it „ValueB“ connect this alarm to the variable „ValueB“. The Variable „ValueB“is having only two states (0, 1), while its value is equal to one we wish to have an alarm, Then we will setup in „Generate if“ dialogue „Value out of range” where Min. and Max. Values will be „0“, keep „ID“ equal to „0“, in „Priority“ dialogue select „1“, which is the highest priority, in text dialogue write operators understandable description about the alarm. Select a Wav file to be played in „Sound_at_beginning“ (this could be a recorded sound message, in our case we have chosen a sound offered by the operation system). Sign the check boxes „Store in archive“ and „Acknowledgement“.

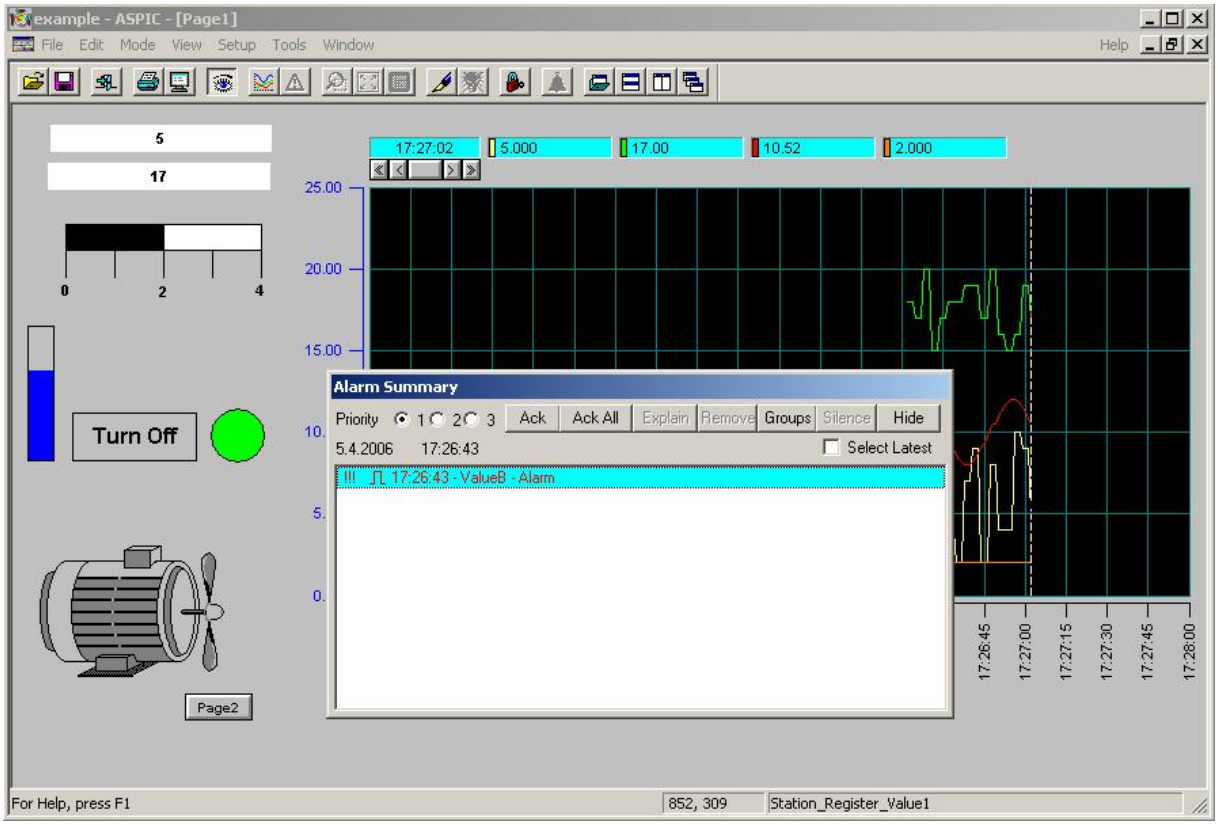


Note: In „Alarm Definition“ window their exit a wide variety of setup options, its functionality is out of the range of this project, further details you can find in Aspic 3.30 user's manual. One important property in Aspic 3.30 is the capability to run and configure the parameters of an external program once an alarm occurs. This property enables us to send alarms via e-mails, sms ...etc.

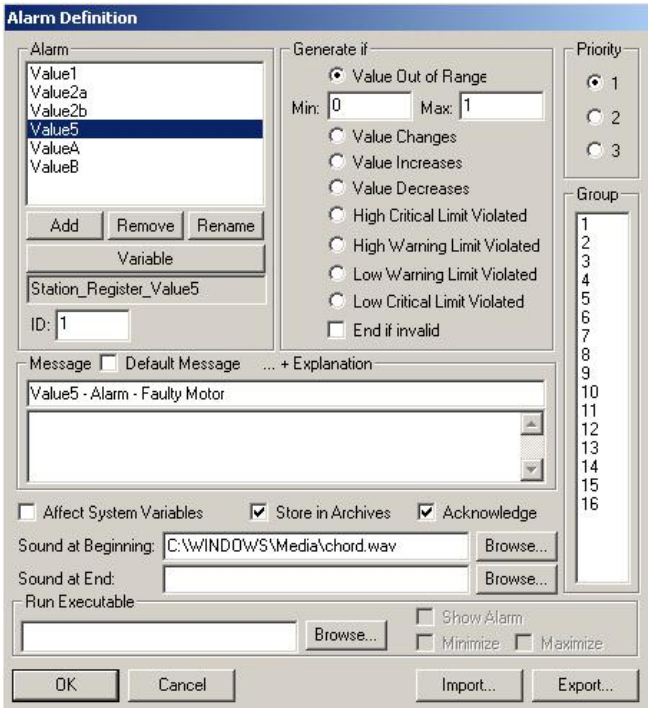
Now run the visualization and click the button „Turn On“ placed on the visualization page „Page1“.

The value of „ValueB“ variable changes from 0 to 1, an alarm will be flagged. Alarm is displayed in the window „Alarm Summary“.

Flagged alarm could be acknowledged, and operator's comments could be written, and in case of sufficient permissions alarms could be deleted.



b) Further we will define a motor failure alarm. Motor‘s state could be gained from „Station_Register_Value5“ variable’s value. Once „Station_Register_Value5“ variable’s value is equal or greater than 2, then we have a got a motor failure case.

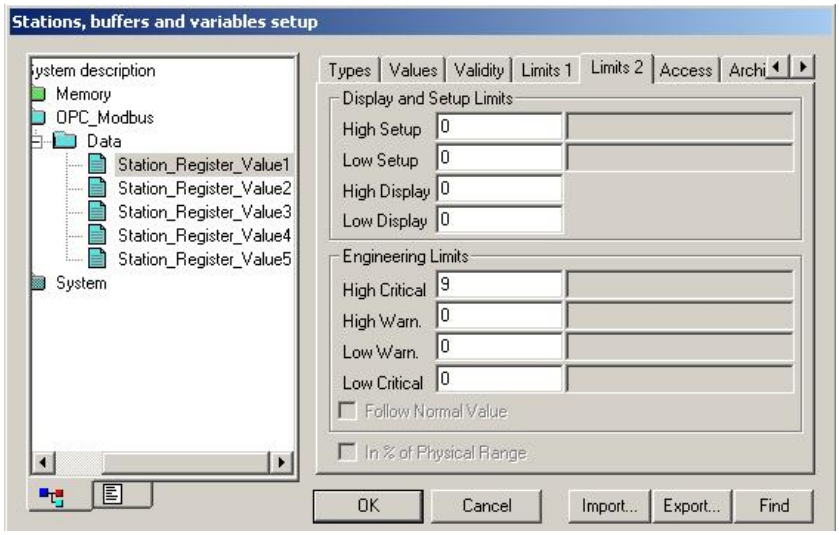


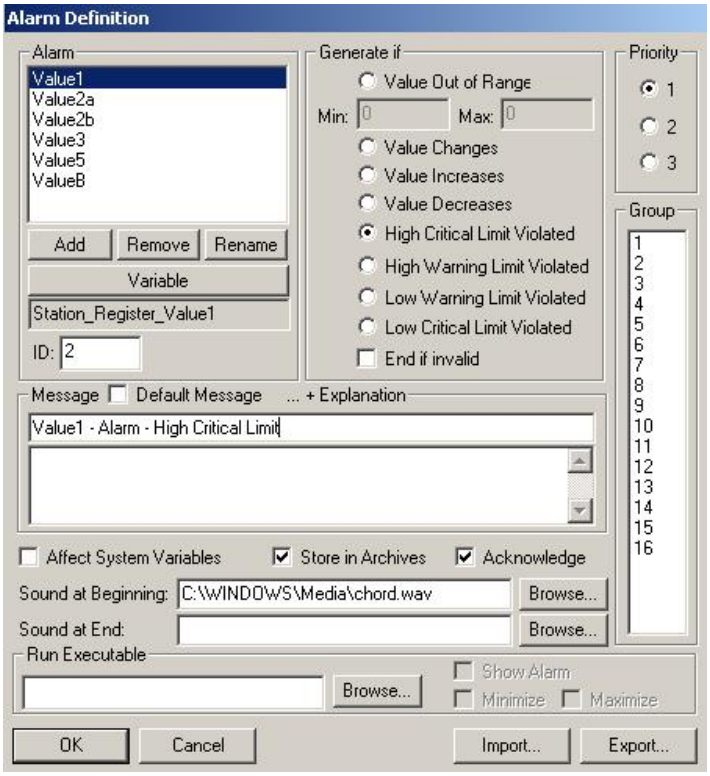
The Alarm „Value5“ connect to the variable „Station_Register_Value5“. Setup in „Generate if“ dialogue „Value out of range” where Min. and Max. Values will be „0“and „1“. Minimal value that will flag an alarm will be in this case „2“.

Test this alarm.

Note: The sound announcing alarms in Aspic 3.30 could be globally defined for certain alarm levels. For further details please refer back to Aspic 3.30 user's manual.

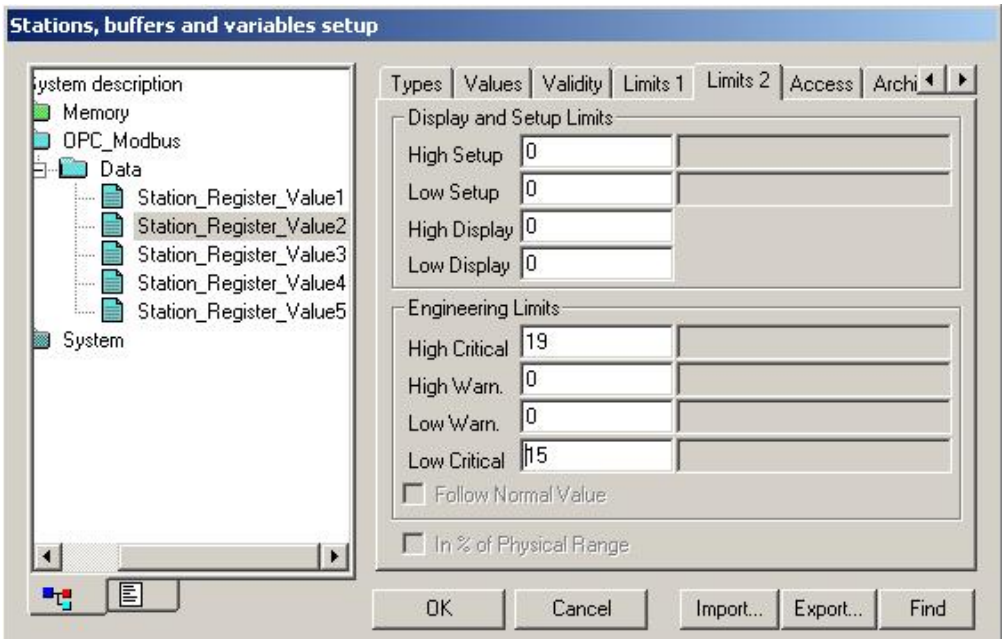
c) The task of this part is to flag an alarm when „Station_Register_Value1" value exceeds its „High Critical" limit. High critical limit could be defined for each individual variable. From the well known „Stations, buffers, and variables setup" window select the variable „Station_Register_Value1" and choose the dialogue „Limits2" from the multi-dialogue on the right side of the window, and setup „High_Critical" limit.

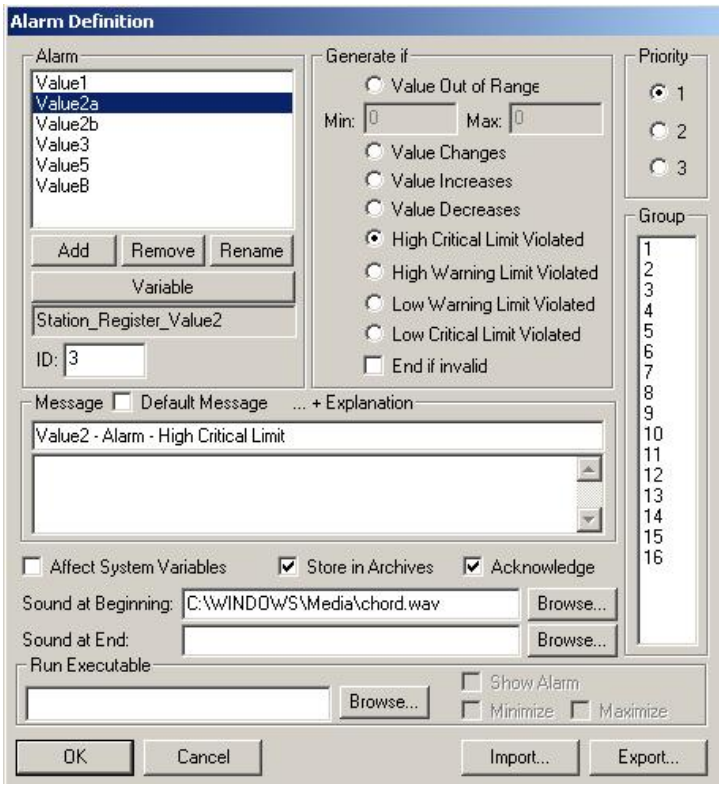


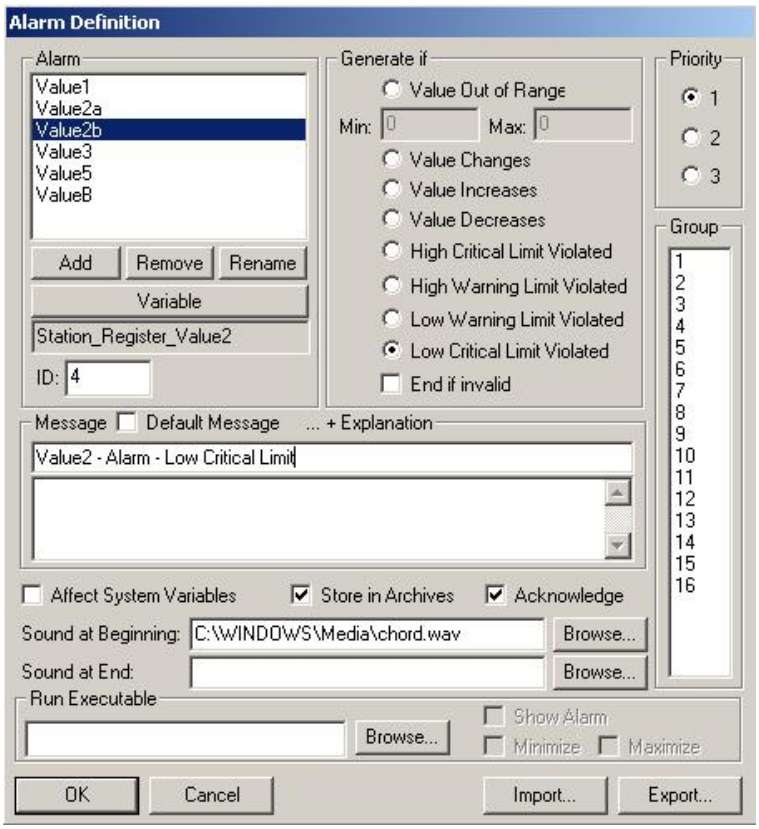


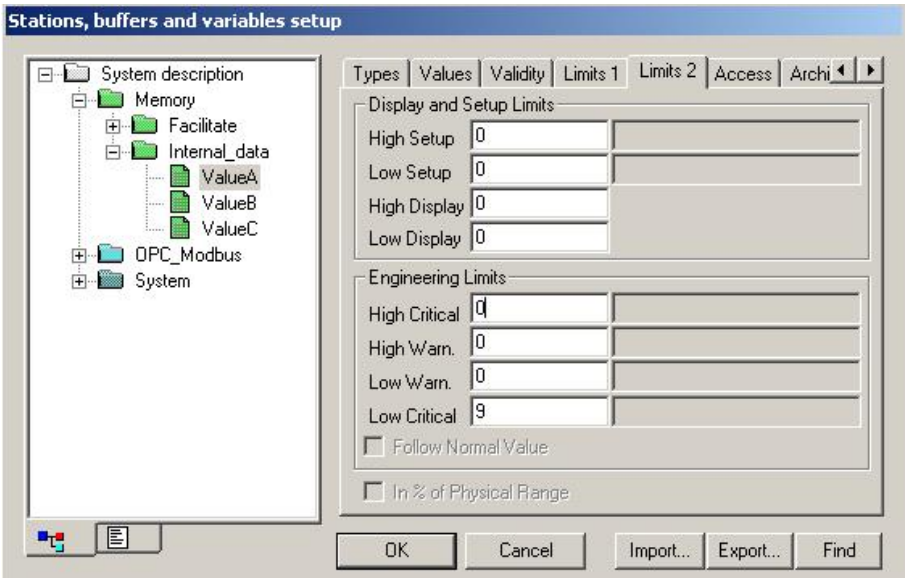
In the window „Alarm_definition" and in alarm dialogue add new alarm call it „Value1"connect this alarm to the variable „Value1" and in the dialogue „Generate if" select „High Critical Limit Violated". Further details setup according to provided figure here under.

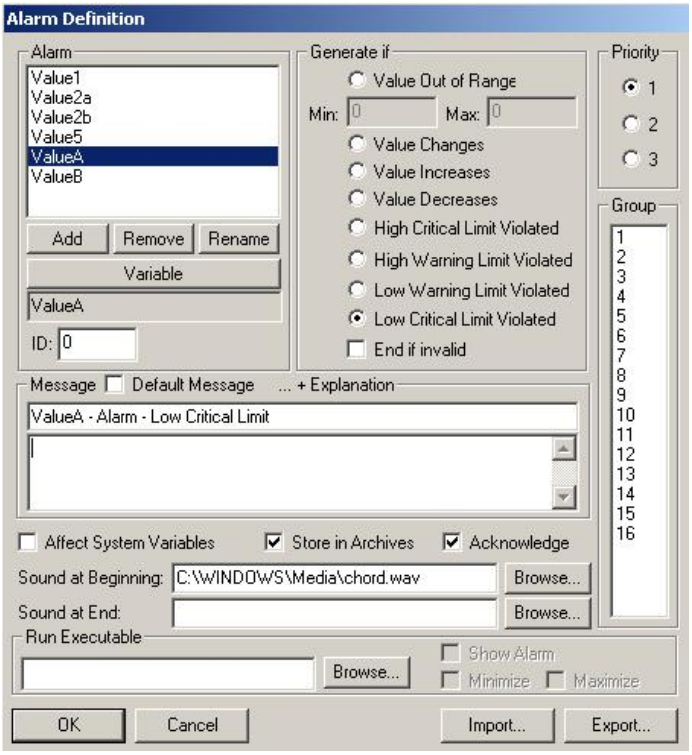
d) Similarly as in previous alarms add new alarm that will be flagged by exceeding high and low critical limits defined for the variable „Station_Register_Value2" (Low Limit = 15, High Limit =19). Add one more alarm based on low critical limit (= 9) of „ValueA". Don't forget to set high and low critical limits in „Stations, buffers, and variables setup" window.


Stations, buffers and variables setup
System description
Memory
OPC_Modbus
Data
Station_Register_Value1
Station_Register_Value2
Station_Register_Value3
Station_Register_Value4
Station_Register_Value5
System
Types
Values
Validity
Limits 1
Limits 2
Access
Archi
Display and Setup Limits
High Setup
Low Setup
High Display
Low Display
Engineering Limits
High Critical
High Warn.
Low Warn.
Low Critical
Follow Normal Value
In % of Physical Range
OK
Cancel
Import...
Export...
Find

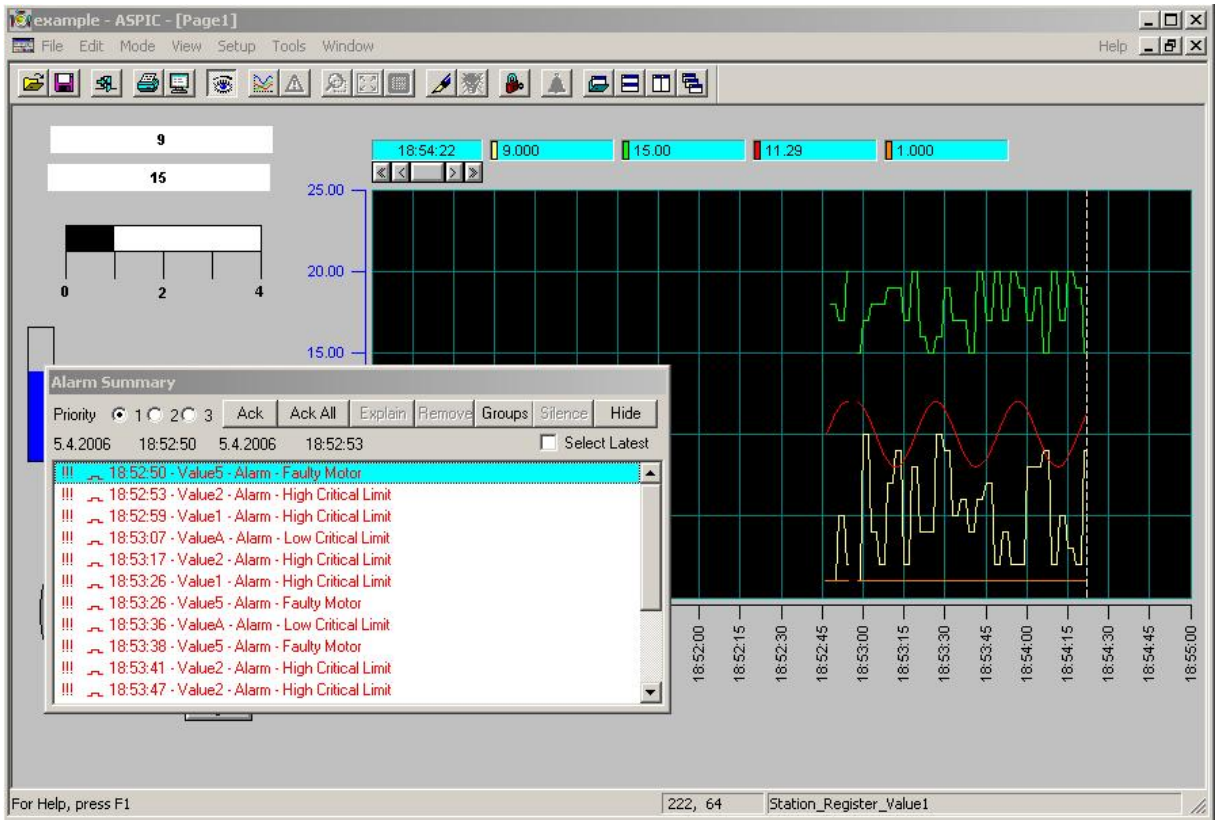

Alarm Definition
Alarm
Value1
Value2a
Value2b
Value3
Value5
ValueB
Add
Remove
Rename
Variable
Station_Register_Value2
ID:
Generate if
Value Out of Range
Min:
Max:
Value Changes
Value Increases
Value Decreases
High Critical Limit Violated
High Warning Limit Violated
Low Warning Limit Violated
Low Critical Limit Violated
End if invalid
Priority
Group
Message
Default Message
... + Explanation
Value2 - Alarm - High Critical Limit
Affect System Variables
Store in Archives
Acknowledge
Sound at Beginning:
C:\WINDOWS\Media\chord.wav
Browse...
Sound at End:
Run Executable
Show Alarm
Minimize
Maximize
OK
Cancel
Import...
Export...


Alarm Definition
Alarm
Value1
Value2a
Value2b
Value3
Value5
ValueB
Add
Remove
Rename
Variable
Station_Register_Value2
ID: 4
Generate if
Value Out of Range
Min: 0
Max: 0
Value Changes
Value Increases
Value Decreases
High Critical Limit Violated
High Warning Limit Violated
Low Warning Limit Violated
Low Critical Limit Violated
End if invalid
Priority
1
2
3
Group
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Message
Default Message
... + Explanation
Value2 - Alarm - Low Critical Limit
Affect System Variables
Store in Archives
Acknowledge
Sound at Beginning:
C:\WINDOWS\Media\chord.wav
Browse...
Sound at End:
Browse...
Run Executable
Browse...
Show Alarm
Minimize
Maximize
OK
Cancel
Import...
Export...


Stations, buffers and variables setup
System description
Memory
Facilitate
Internal_data
ValueA
ValueB
ValueC
OPC_Modbus
System
Types
Values
Validity
Limits 1
Limits 2
Access
Archi
Display and Setup Limits
High Setup 0
Low Setup 0
High Display 0
Low Display 0
Engineering Limits
High Critical 0
High Warn. 0
Low Warn. 0
Low Critical 9
Follow Normal Value
In % of Physical Range
OK
Cancel
Import...
Export...
Find



Double-check alarm system by running visualization.

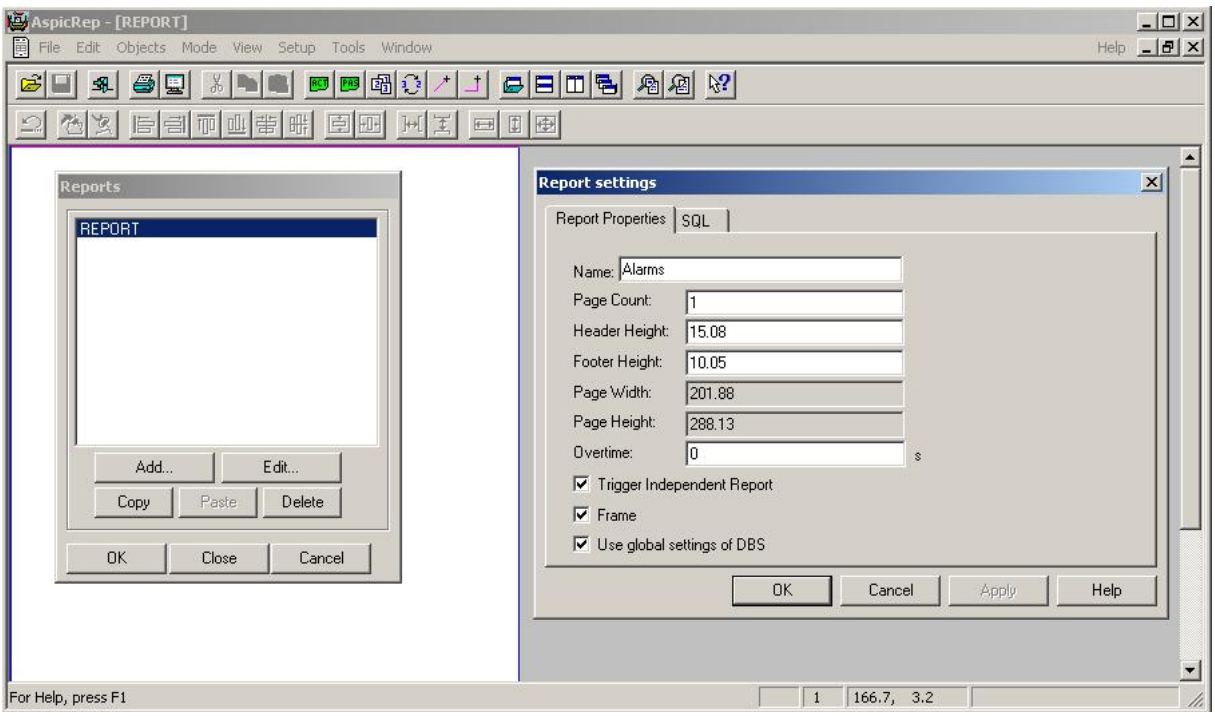


36. The last part of our project will be creating protocols (reports). Aspic 3.30 has its own tool for automatic report generation „AspicRep“. In „AspicRep“ you can create report templates. Report template is made out of passive and active objects similar to Aspic's visualization objects. Assembled objects in report's template will be fulfilled with historical data from Aspic archive or a database that aspic has been archiving to.

    Run AspicRep, Choose „File->New” from main menu, if no new file has been established yet.

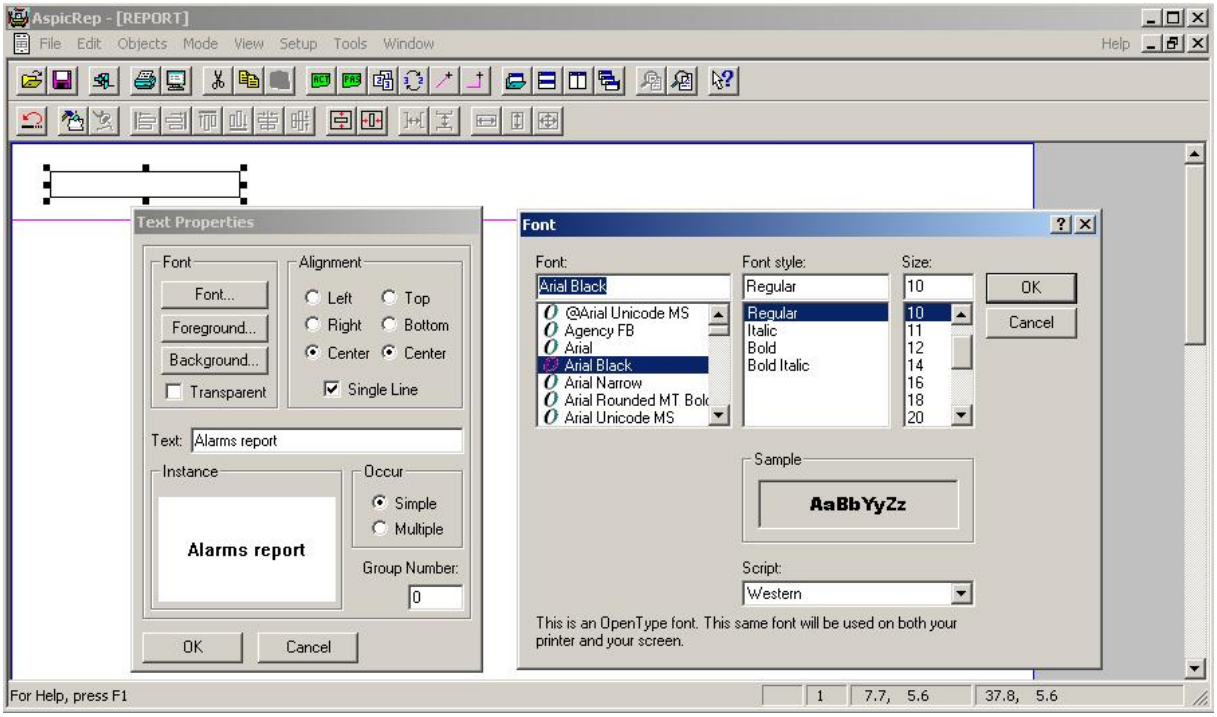
    From menu choose „Window -> Reports”, in the pop up window you have a list of reports. At the moment we have got only one report „REPORT“. Select it and click „Edit“, another pop up „Report_settings“ window will appear. We will rename report to „Alarms“, where we will develop alarms report. In the header height item insert 15, and for footer height set 10. Header and footer we will have on each protocol page, exit both pop up windows by clicking „OK“buttons.

    Now we need to save the report file, for this purpose use from menu „File -> Save as“. The best would be if you save the report file to the same folder that contains visualization project. It is not a condition. Name report file „example.apr“.

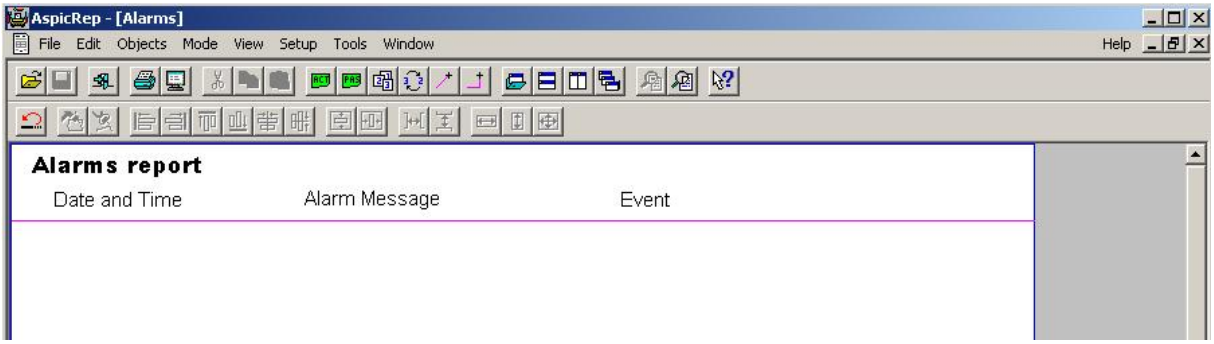


Note: print profile could be configured from „File -> Printer Setup -> All reports (Selected report)“ where we can select whether print orientation will be landscape or portrait. Such setup could be unified for all protocols or could be selected individually for each protocol. (The best for our alarm statement is to have it printed as portrait on A4, while graphs will look better as landscape on A4). Paper size depends on selected printer.

37. On report's page header place the passive object „Text" and use it to write report's headline. Click the icon of passive objects in tools panel and choose the object „Text". Place it on report's page. Its location and size realize approximately as shown in the following figure, open properties dialogue of this object by double-clicking it. Write required text „Alarms report", and reconfigure its font and alignment as you wish.



Further place in the header of report's page three more text objects that would be used to write captions of displayed alarms. Text and location choose approximately as shown in the following figure.

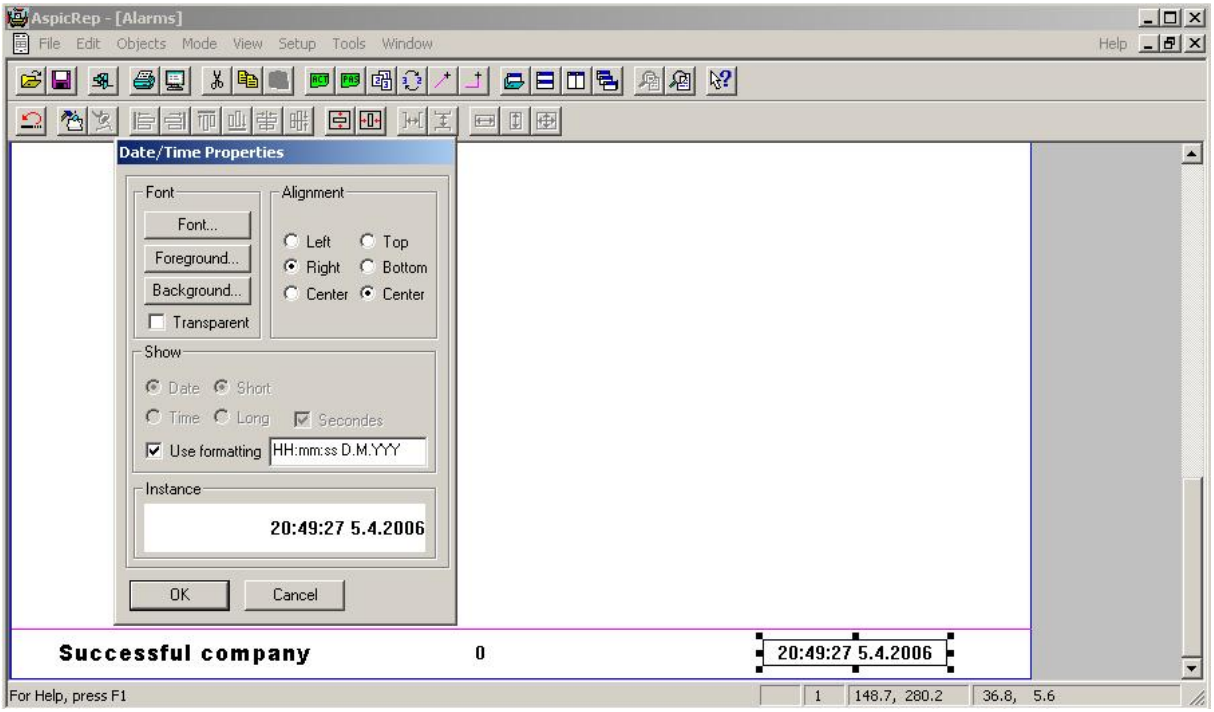


Note: Enlarging and shrinking report pages while creating reports, help a lot. For enlargement use icon, and for shrinking use .

Further we create the footer of report's page and to its area we place company's trade name (Logo could be placed as well), page number and project creation date.

In page's footer area place the following objects,

"Text" in which you write company's trade name, "Page Number" from passive objects and "Date/Time" passive object and configure its properties as shown in the figure here under. „HH:mm:ss D.M.YYY" format string is important. For further details about format strings please refer to user's manual.



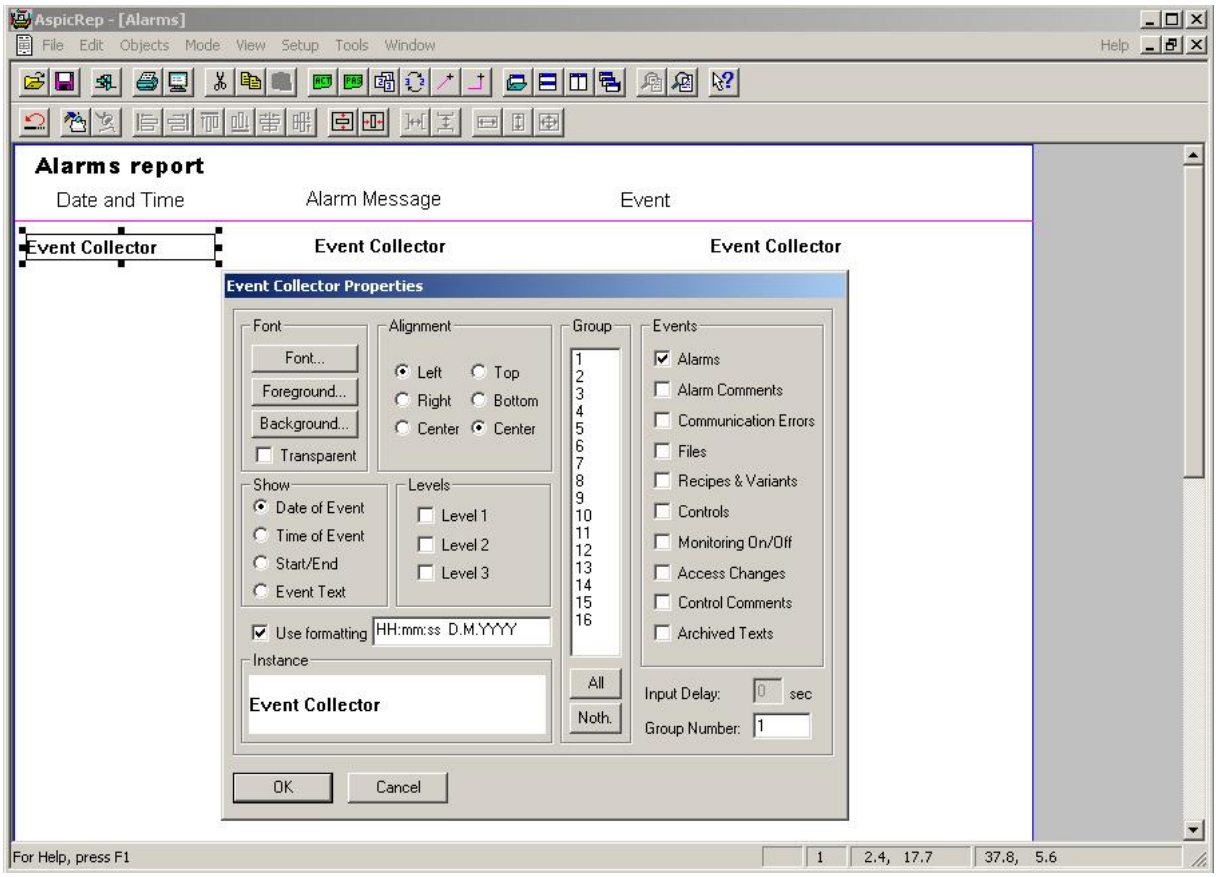
Now we have got the header and footer of report's page ready, still we need to fulfill this report with data, the best object for this job will be the active object „Event collector".

„Event collector" objects we place on the page and connect it to required variables that are defined in Aspic 3.30 system files. In order to use variables, report's project should be interconnected with system's file. From menu choose File -> Open station file then choose the system file of our project „example.asf".

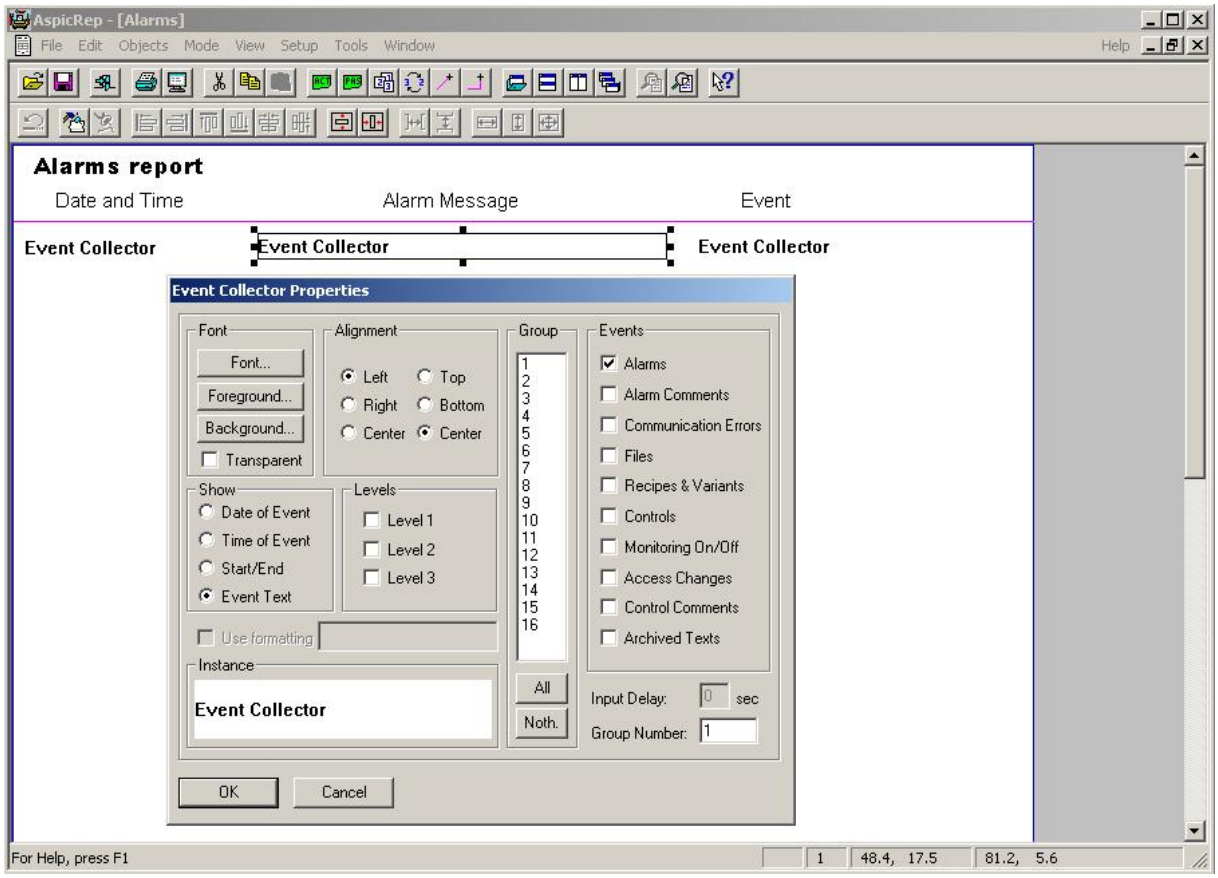
Likewise it is essential to associate Aspic 3.30 with report's project file. Switch to Aspic 3.30 application and from menu choose File -> Associate report file. Choose the file „example.arp".

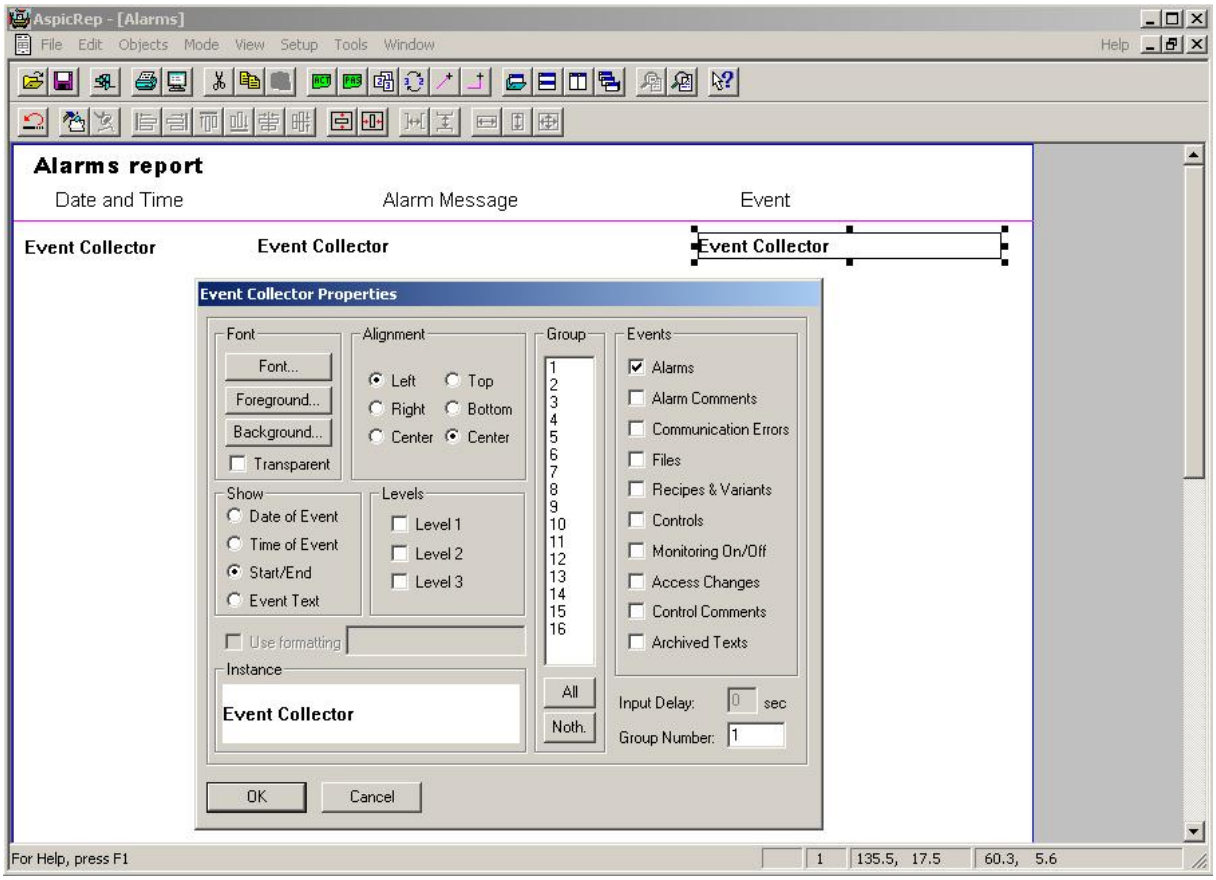
Now we are capable to run report generation from Aspic 3.30 by clicking the icon . You can try it, but be Aware of that their will be no data. On report's page you will have only header and footer.

On report's page place three „Event collector" active objects, in all of them sign the check box "Alarms" from "Events" dialogue. Set up in the "Show" dialogue for the first to "Date of Event" and display its date and time in the following format HH:mm:ss D.M.YYYY, for the second choose „Event Text", and for the third will choose „Start/End". Configure object's properties according to the following figures. It is essential to place all active objects in one group, for interlaced functionality.



Note: for mutual object's alignment use the following icons  from tools panel.



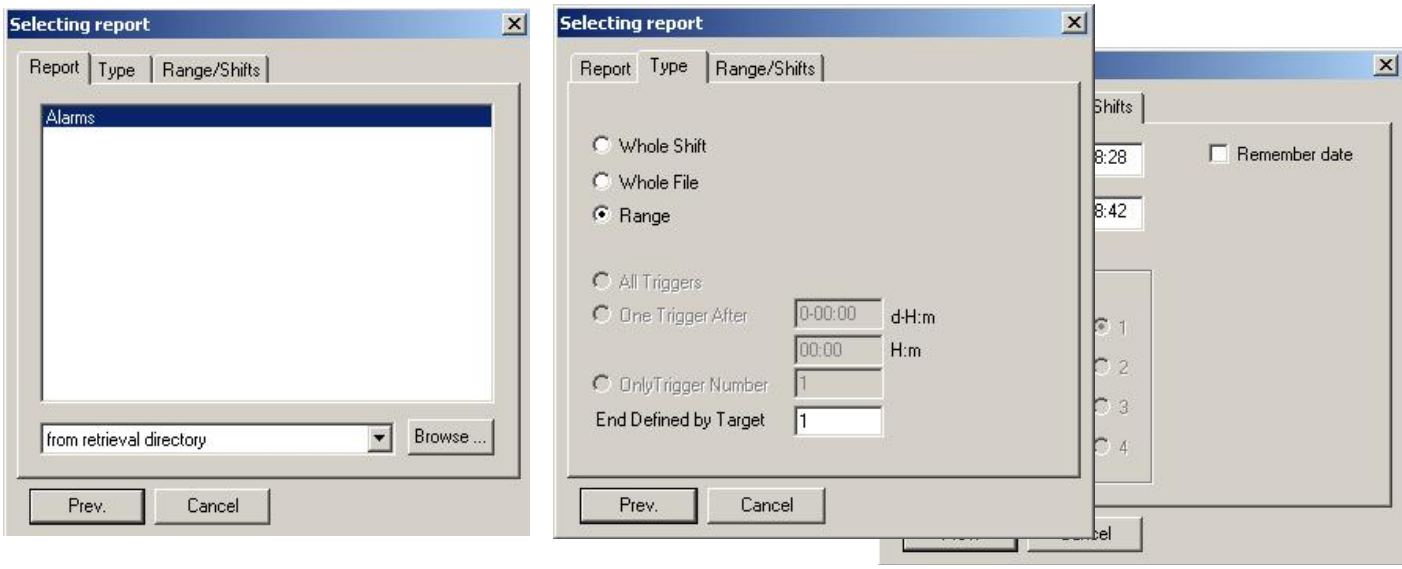


Save reports project close AspicRep application and shift activities to Aspic3.30, and click the icon  . It doesn't matter whether visualization is running or not.

In opened pop up window „Selecting report“ select required report (at the moment we do have only Alarms report) and choose that you wish to create it from retrieval directory (we have used archiving to Aspic 3.30 file).

From the dialogue „Type“ choose report’s type required, for this exercise choose time range.

In the following dialogue Range/Shifts select the time period. Don’t select too long period (because we do have too many alarms, for testing purposes 10 minutes period will be sufficient). When choosing period’s (From, To) values, remember when have you had the visualization running. (In order to display some results, by other words choose a period in which some alarms had happened).



On the following page displays an example out of created alarms report.

Alarms report

Date and Time	Alarm Message	Event
18:29:57 5.4.200	ValueB - Alarm	Begin
18:30:00 5.4.200	Value5 - Alarm - Faulty Motor	Begin
18:30:02 5.4.200	Value2 - Alarm - High Critical Limit	Begin
18:30:02 5.4.200	Value5 - Alarm - Faulty Motor	End
18:30:05 5.4.200	Value2 - Alarm - High Critical Limit	End
18:30:08 5.4.200	Value1 - Alarm - High Critical Limit	Begin
18:30:11 5.4.200	Value1 - Alarm - High Critical Limit	End
18:30:13 5.4.200	ValueA - Alarm - Low Critical Limit	Begin
18:30:23 5.4.200	ValueA - Alarm - Low Critical Limit	End
18:30:26 5.4.200	Value2 - Alarm - High Critical Limit	Begin
18:30:29 5.4.200	Value2 - Alarm - High Critical Limit	End
18:30:35 5.4.200	Value1 - Alarm - High Critical Limit	Begin
18:30:35 5.4.200	Value5 - Alarm - Faulty Motor	Begin
18:30:38 5.4.200	Value1 - Alarm - High Critical Limit	End
18:30:41 5.4.200	Value5 - Alarm - Faulty Motor	End
18:30:43 5.4.200	ValueA - Alarm - Low Critical Limit	Begin
18:30:47 5.4.200	Value5 - Alarm - Faulty Motor	Begin
18:30:50 5.4.200	Value2 - Alarm - High Critical Limit	Begin
18:30:50 5.4.200	Value5 - Alarm - Faulty Motor	End
18:30:52 5.4.200	ValueA - Alarm - Low Critical Limit	End
18:30:53 5.4.200	Value2 - Alarm - High Critical Limit	End
18:30:56 5.4.200	Value2 - Alarm - High Critical Limit	Begin
18:30:56 5.4.200	Value5 - Alarm - Faulty Motor	Begin
18:30:59 5.4.200	Value2 - Alarm - High Critical Limit	End
18:31:02 5.4.200	Value2 - Alarm - High Critical Limit	Begin
18:31:02 5.4.200	Value5 - Alarm - Faulty Motor	End
18:31:05 5.4.200	Value2 - Alarm - High Critical Limit	End
18:31:08 5.4.200	Value5 - Alarm - Faulty Motor	Begin
18:31:11 5.4.200	Value5 - Alarm - Faulty Motor	End
18:31:12 5.4.200	ValueA - Alarm - Low Critical Limit	Begin
18:31:17 5.4.200	Value2 - Alarm - High Critical Limit	Begin
18:31:20 5.4.200	Value2 - Alarm - High Critical Limit	End
18:31:22 5.4.200	ValueA - Alarm - Low Critical Limit	End
18:31:23 5.4.200	Value2 - Alarm - High Critical Limit	Begin
18:31:26 5.4.200	Value2 - Alarm - High Critical Limit	End
18:31:35 5.4.200	Value2 - Alarm - High Critical Limit	Begin
18:31:35 5.4.200	Value5 - Alarm - Faulty Motor	Begin
18:31:38 5.4.200	Value5 - Alarm - Faulty Motor	End

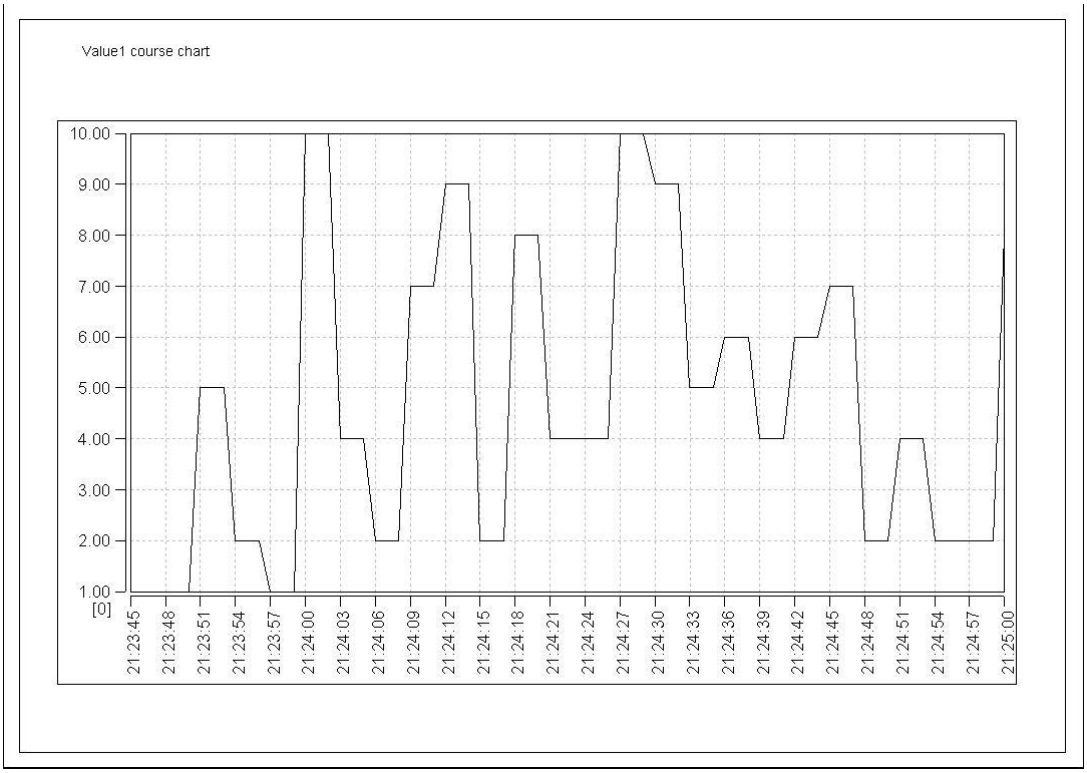
38. In this point we will create a graphical report that will chart the course of variable „Station_Register_Value1“ and „ValueA“ for a user defined period of time, and one more report that contains a table out of every minute record of the variable „Station_Register_Value3“ will be created as well.

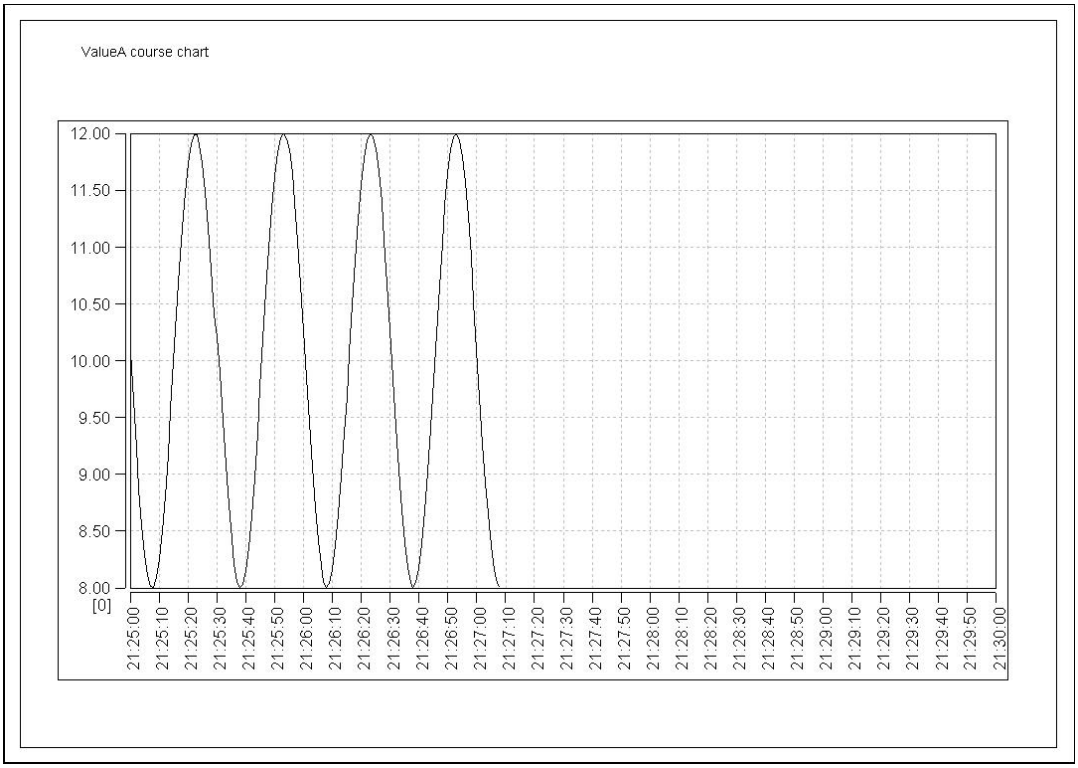
    Run AspicRep and in the already existing reports project, add new report „Window->reports“, „Add“. Name the report „Value1f“, don’t set up the header and footer of this report just keep its 0 values.

    Set printing orientation to landscape, on the report page place an active object „Graph“. Expand the graph over report’s page and double click it to display „Graph properties“ dialogue. In this dialogue setup “Period” to 00:00:01 (during this example we will scan each second, in reality choose technology related values). Further we connect this graph to the variable „Station_Register_Value1”, you are free to change Fonts & colors the way you wish. It is possible to chart more variables in one graph. In our example we will chart one variable.

    Place on report’s page the passive object „Text“ and write in it chart title, if you wish you can add two more text objects in order to label each axis’s units.

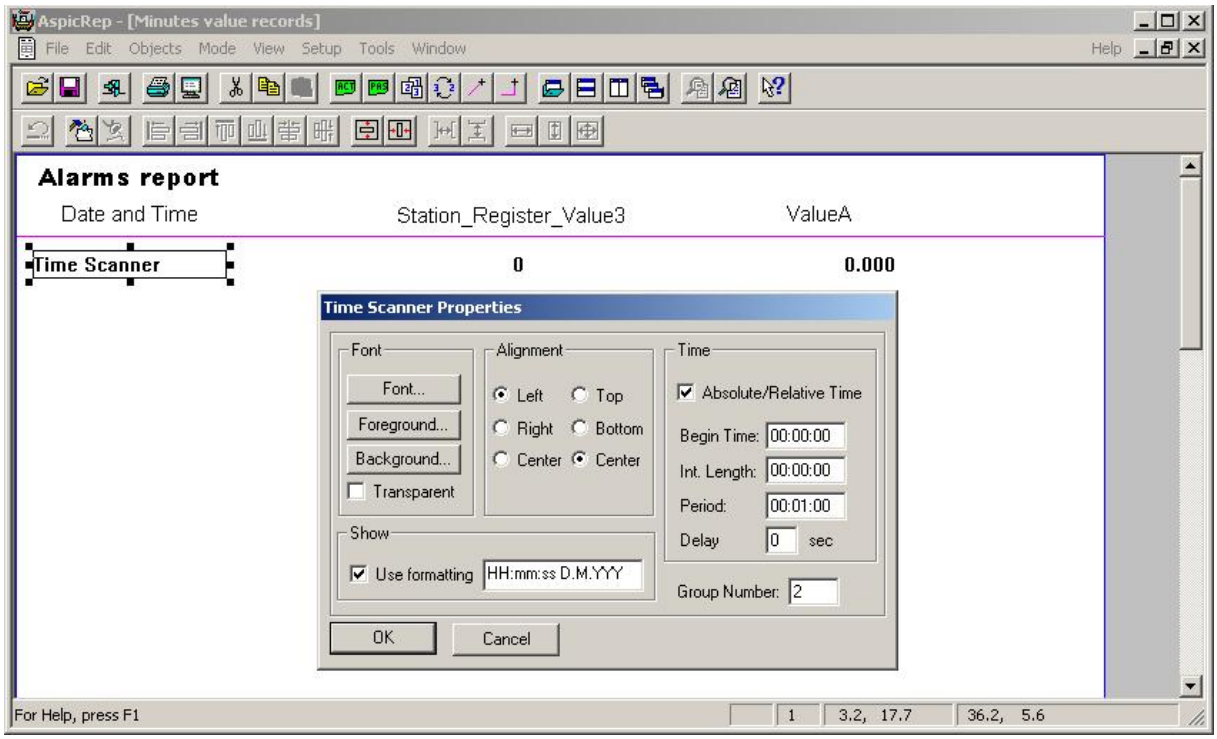
    Save report’s project and close the application AspicRep. Shift activities to Aspic 3.30 and display generated reports in a similar way as has been explained in alarms report.





The last required report is a statement of each minute value record of the variable „Station_Register_Value3“. Add new report and name it „Minute_value_records“. Header and footer we do create the same way as in alarm reports, just we change used text. Use the active object „Time scanner“ and configure it as shown in the following figure. For displaying variable's values use the object „Value“ and connect it to selected variable.

Don't forget placing all active objects in one group.



Check out created report. Choose Langer time interval, and print out minute records statement. Selection of proper time interval in which data has been archived is essential in order to not have an empty report.

Alarms report

Date and Time	Station_Register_Value3	ValueA
21:45:00 5.4.200	30	10.364
21:46:00 5.4.200	31	10.399
21:47:00 5.4.200	35	10.643
21:48:00 5.4.200	33	10.551
21:49:00 5.4.200	33	10.534
21:50:00 5.4.200	33	10.313
21:51:00 5.4.200	31	10.560
21:52:00 5.4.200	31	10.407
21:53:00 5.4.200	34	10.113

Note: in reports you are free to add more variables of your choice. In this report I have added each minute value records of the variable „ValueA“.