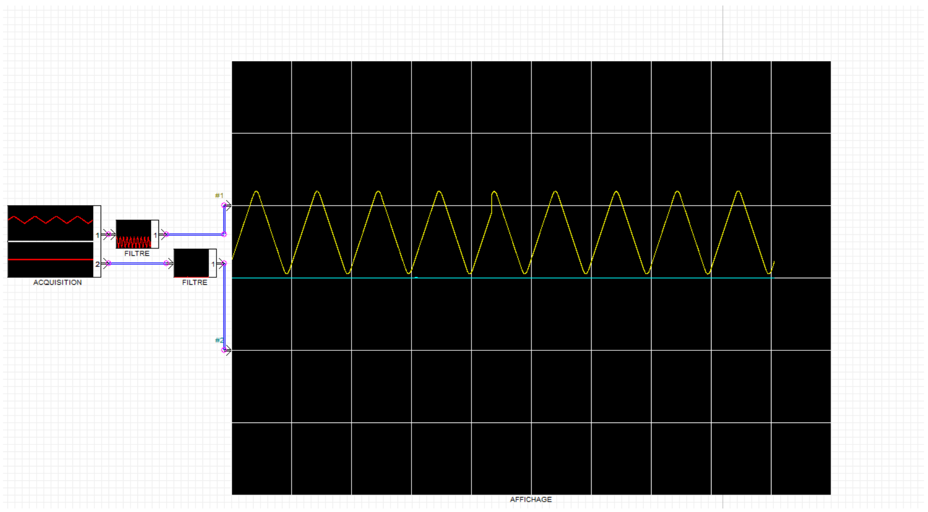
AUTOMDAQ



Sommaire

AUTOMDAQ 1
Main concept 5
Needed items 5
Creating diagrams 5
Copy datas to Excel 12
Blocks list 12
Acquisition 12
Display 15
Math 16
Saturation 16
Filter 16
To Automlab / Avegare 16
User defined blocks 16
Specific informations regarding data acquisition from serial link 22
Concept 22
Serial link format 22
Datas format 22
Synchronization 22
Example using an Arduino card 23

Main concept

AUTOMDAQ is an AUTOMGEN module for datas acquisition, datas processing and datas display

Needed items

AUTOMDAQ needs AUTOMGEN V>=8.105.

AUTOMDAQ can use the following datas acquisition systems:

- LabJack U12,

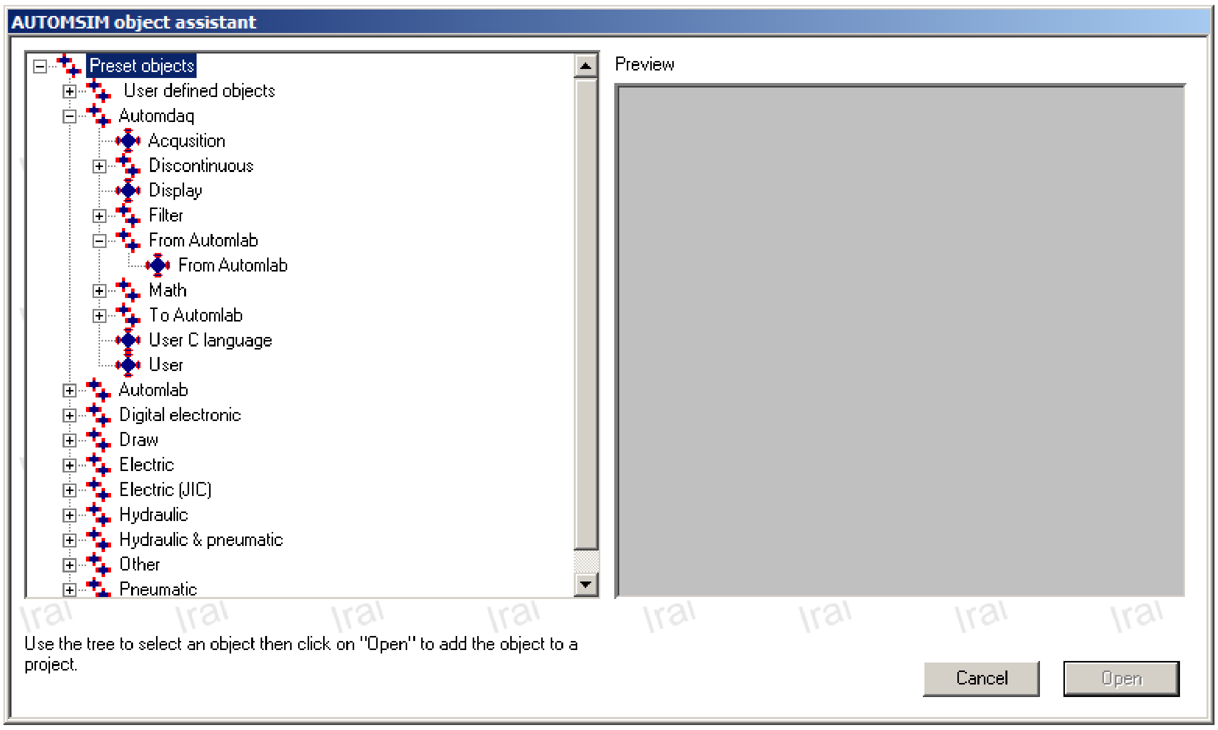
- National Instrument (Eg. 6008,6009),

- Any system with a serial link (Eg. Arduino),

Creating diagrams

AUTOMDAQ diagrams have to be made on AUTOMSIM folders. Those diagrams can coexist with all the other AUTOMGEN possible items.

For adding an AUTOMDAQ block, right click on an AUTOMSIM folder, select "Add an object" and then browse the "AUTOMDAQ" category.



Use the tree to select an object then click on "Open" to add the object to a project.

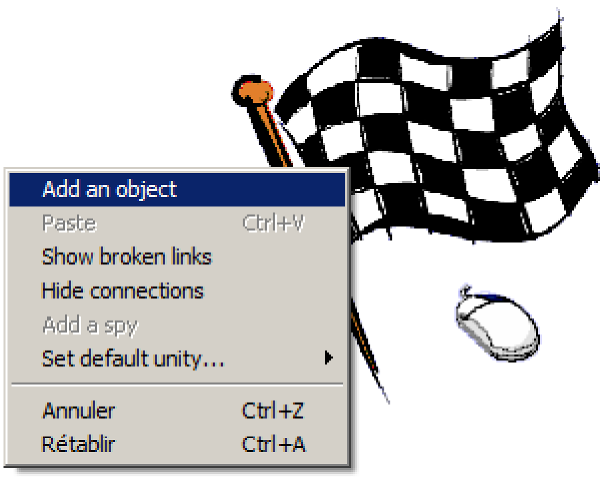
Working principle

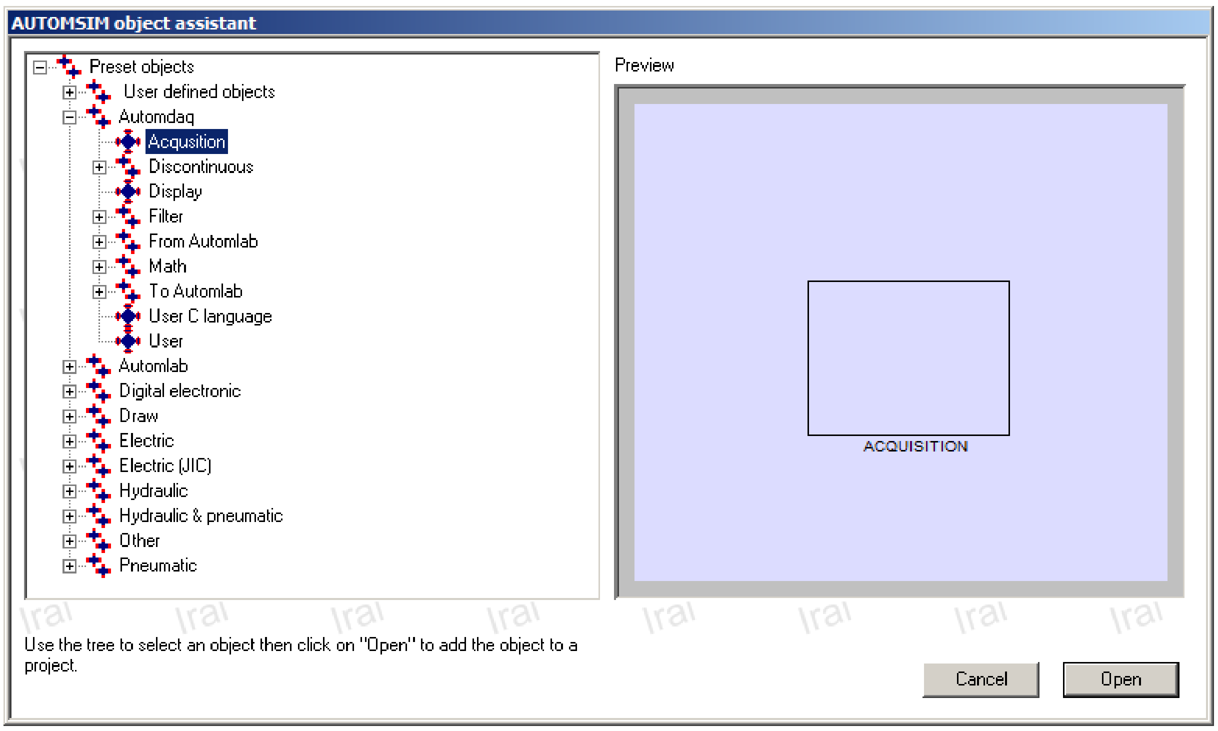
Each block can have one or more inputs on the left side and one or more outputs on the right side. Blocks can also have parameters.

The "Acquisition" object permits to get values from one or more input channels of an acquisition system (Labjack U12, National Instrument or a serial link).

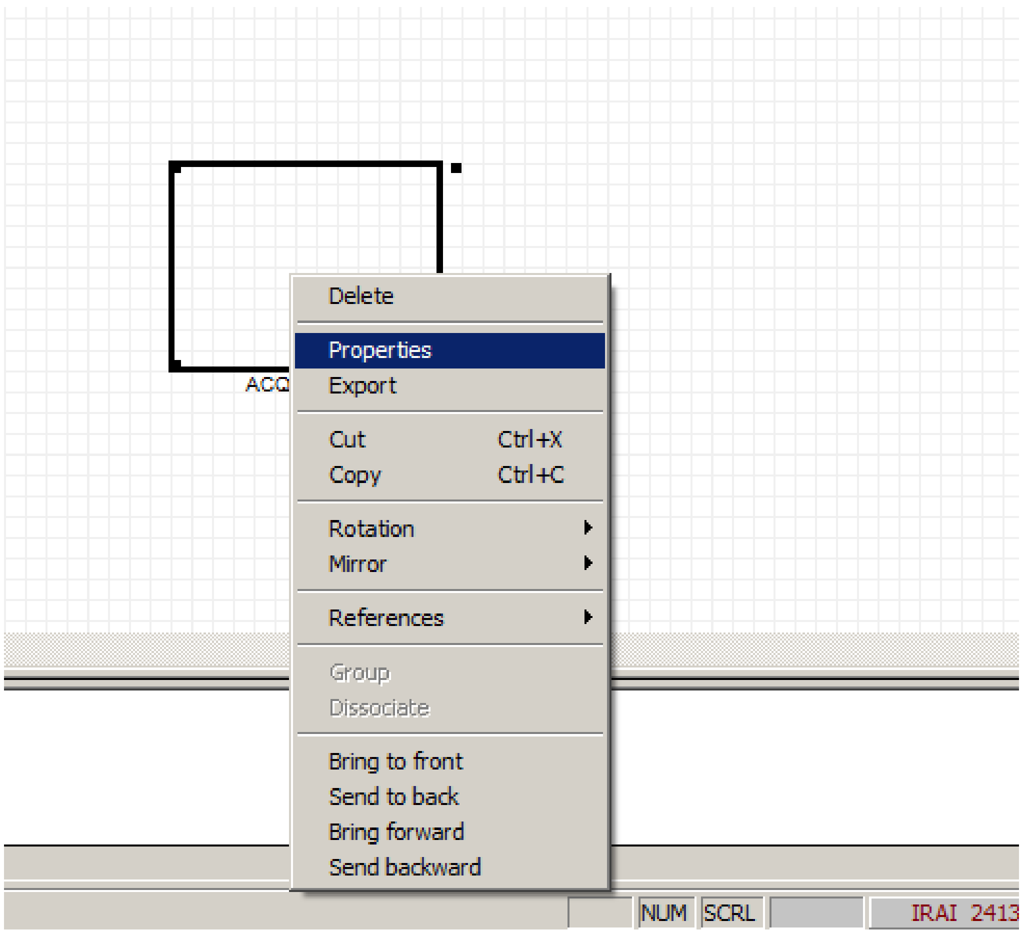
The "Display" object permits to display the acquired datas and to export them to Ms Excel.

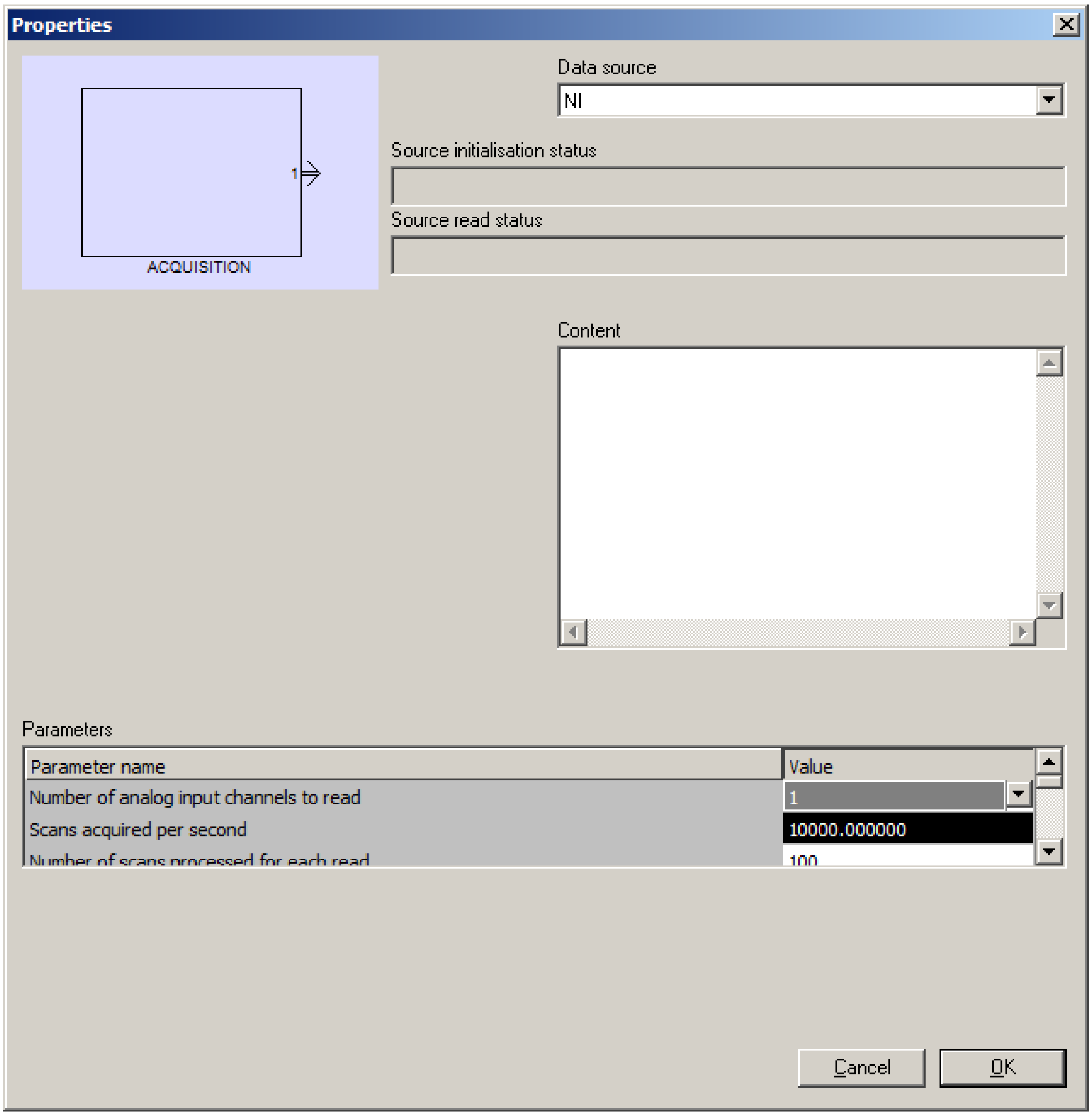
Example: creating a scope for displaying datas acquired on channel#1 of a National Instrument 6009 card with a 10Khz sampling time.





Use the tree to select an object then click on "Open" to add the object to a project.


ACQ
Delete
Properties
Export
Cut Ctrl+X
Copy Ctrl+C
Rotation
Mirror
References
Group
Dissociate
Bring to front
Send to back
Bring forward
Send backward
NUM
SCRL
IRAI 2413



Data source: NI

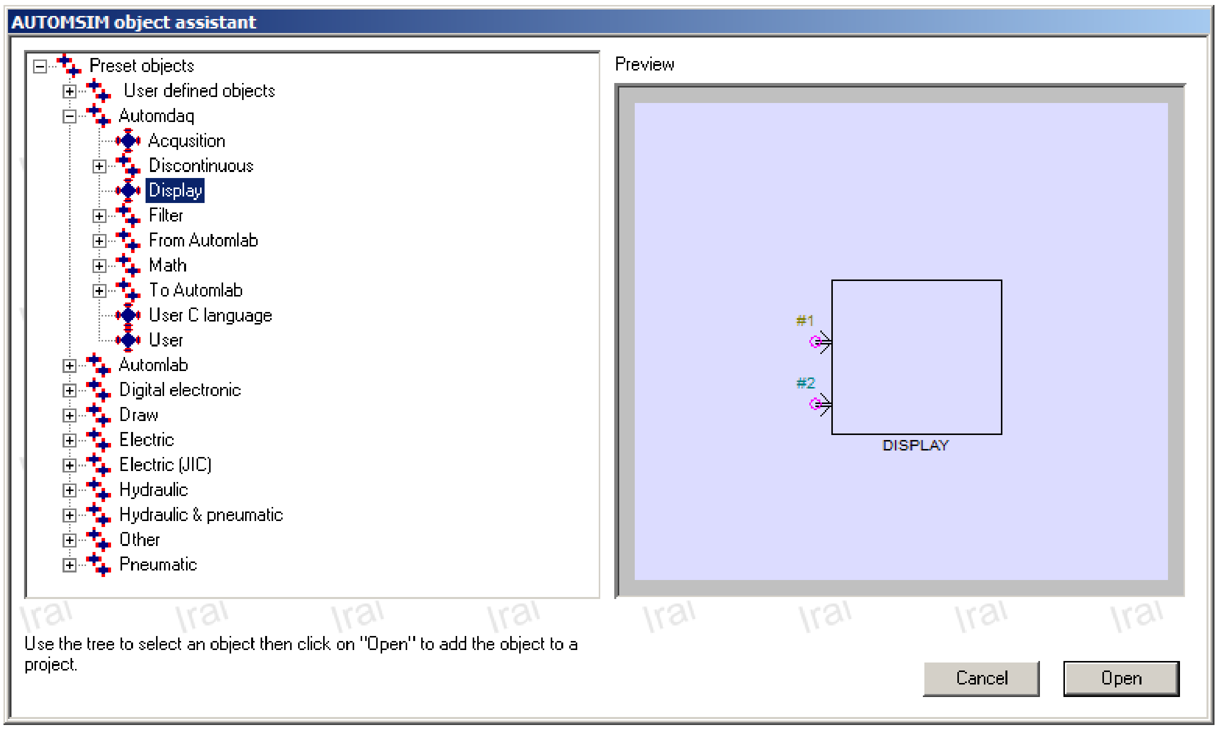
Source initialisation status

Source read status

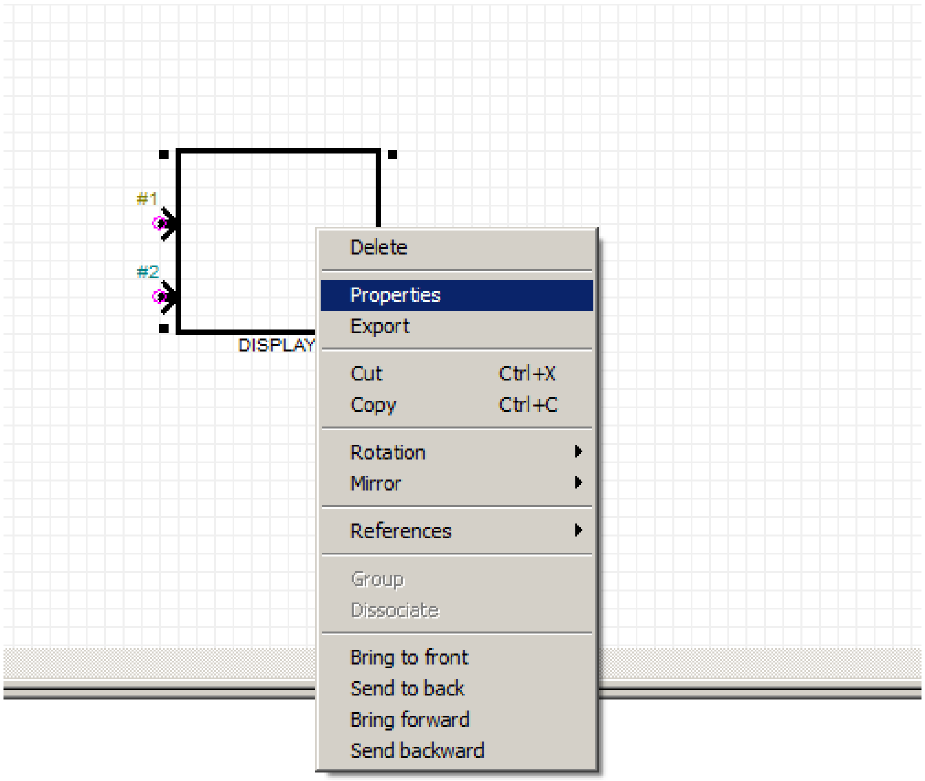
Content

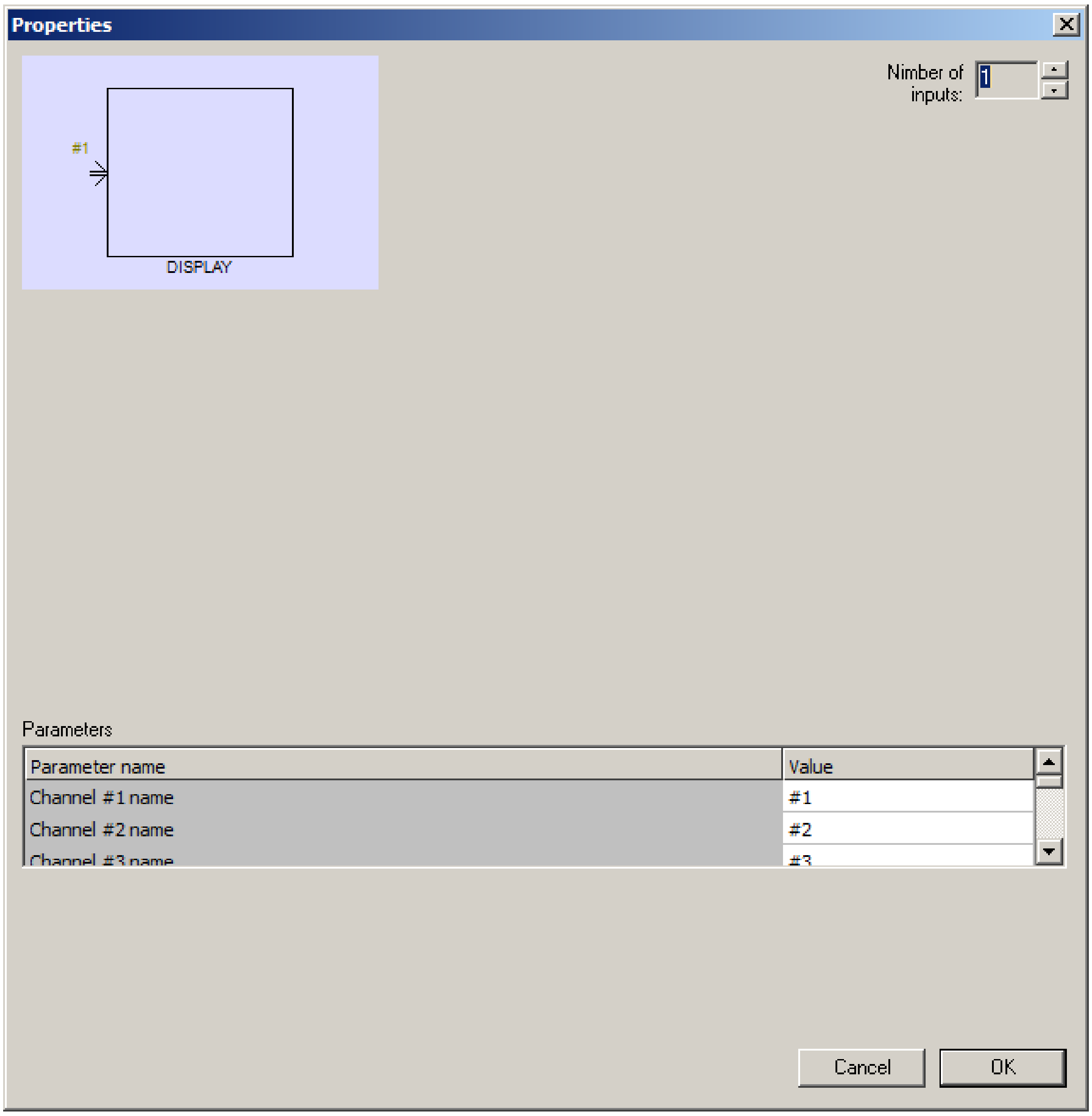
Parameters

Parameter name	Value
Number of analog input channels to read	1
Scans acquired per second	10000.000000
Number of scans processed for each read	100



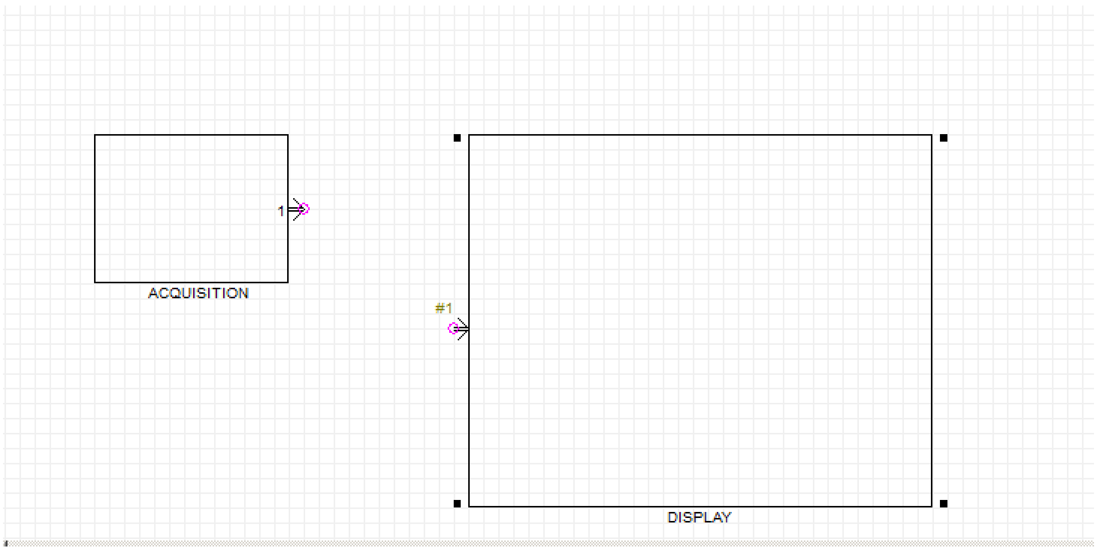
Use the tree to select an object then click on "Open" to add the object to a project.

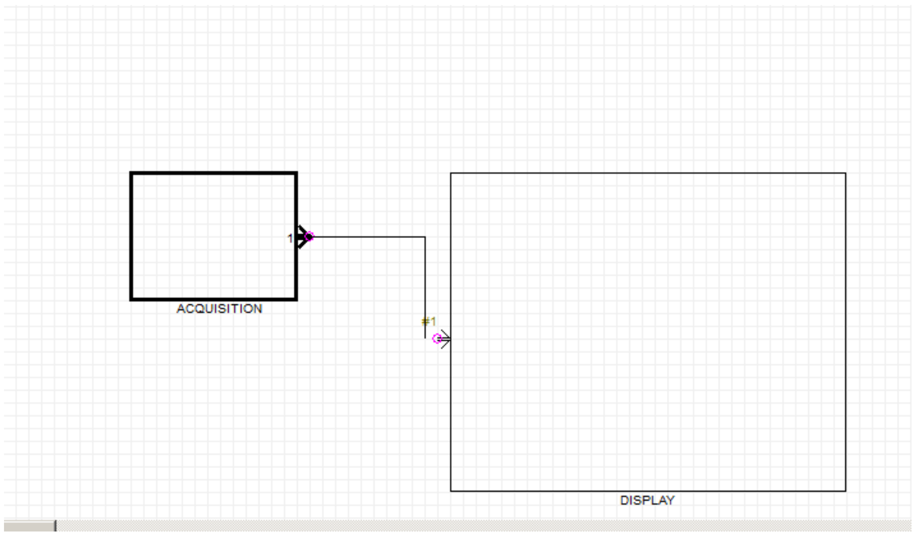




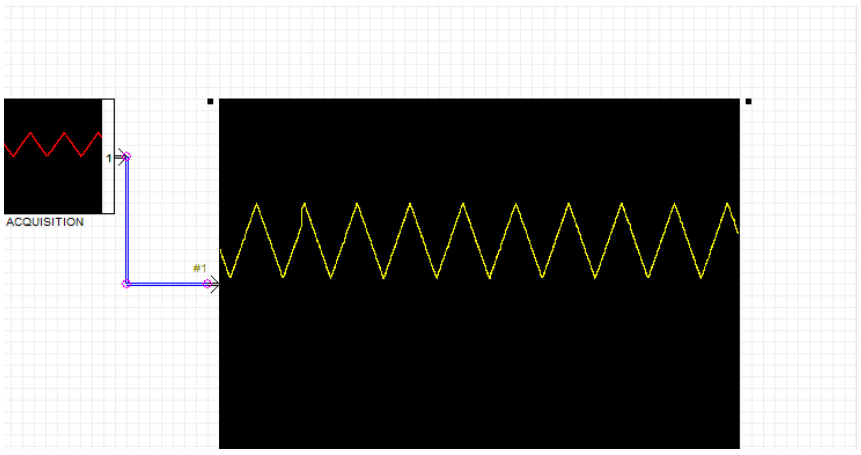
Parameters

Parameter name	Value
Channel #1 name	#1
Channel #2 name	#2
Channel #3 name	#3





Click on "GO" button.

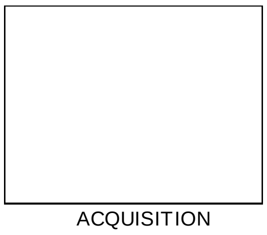


Copy datas to Excel

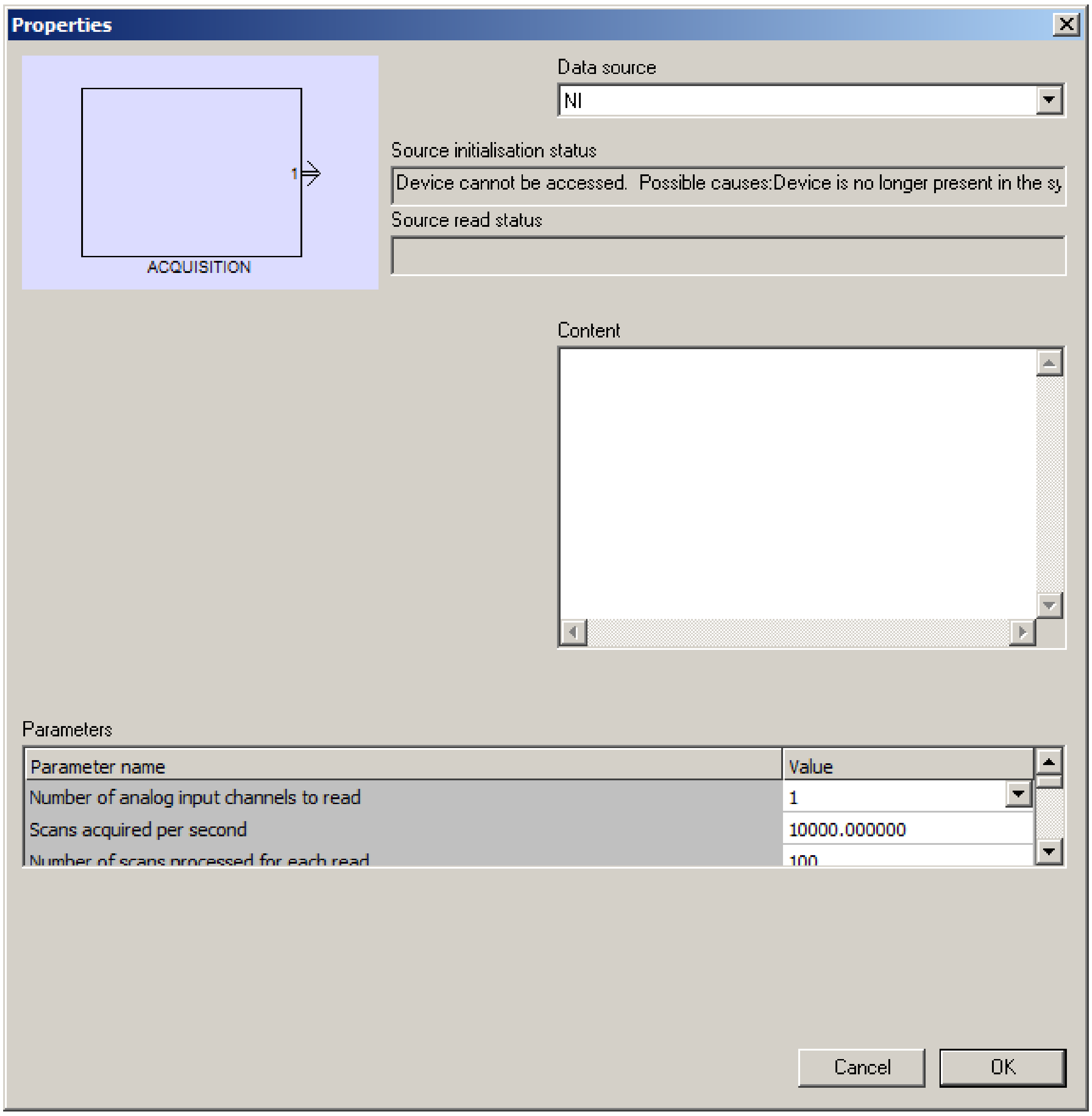
Right click on the "Display" object and select "Copy data to Excel format".

Blocks list

Acquisition



This block is used to acquire datas.



Data source: NI

Source initialisation status: Device cannot be accessed. Possible causes:Device is no longer present in the sy

Source read status:

Content

Parameters

Parameter name	Value
Number of analog input channels to read	1
Scans acquired per second	10000.000000
Number of scans processed for each read	100

"Data source" allows to select acquisition system : LabJack U12, NI (National Instrument), COM PORT (serial link)

"Status" areas display error while initializing acquisition system or while reading datas.

By default, a preview of the acquired datas is displayed on the block in RUN mode. "don't display values" can disable this feature.

LabJack U12 specific parameters

"Number of analog input channels to read": choice of the number of channels to use: 1, 2 or 4

"Number of scans processed for each read": number of scans to transfer from the Labjack U12 to the PC. This parameters may be changed according to the scan rate.

"Number of values stored in object for each channel": the object store in its internal memory the number of values selected in this parameter. Example: for a scan rate of 1000Hz, if this parameter is equal to 10000 then values corresponding to a 10 seconds record time will be stored.

"Scan acquired per second": defines the scan rate for each channel. For Labjack U12, this value must be between 400 et 8192.

Minimum and maximum value : lowest and highest value to be read.

NI specific parameters

"Number of analog input channels to read": choice of the number of channels to use: 1, 2 or 4

"Scan acquired per second": defines the scan rate for each channel. Please see the National Instrument informations of each material for limitations.

"Number of scans processed for each read": number of scans to transfer from the Ni card to the PC. This parameters may be changed according to the scan rate.

"Number of values stored in object for each channel": the object store in its internal memory the number of values selected in this parameter. Example: for a scan rate of 100Hz, if this parameter is equal to 10000 then values corresponding to a 100 seconds record time will be stored.

"Name of the channels": have to be set according the NI syntax for each channel. Eg. Dev1/ai0 = first analog channel for the card "Dev1".

Minimum and maximum value : lowest and highest value to be read.

Serial link specific parameters

"Com port number": com port to use. Eg. 2 means COM2.

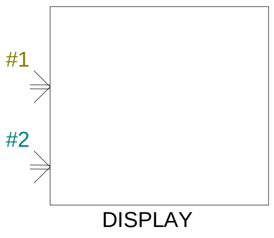
"Baudrate": speed of the serial link.

"Number of analog channels to read": number of analog channels to read from the source system: 1, 2, 3 or 4.

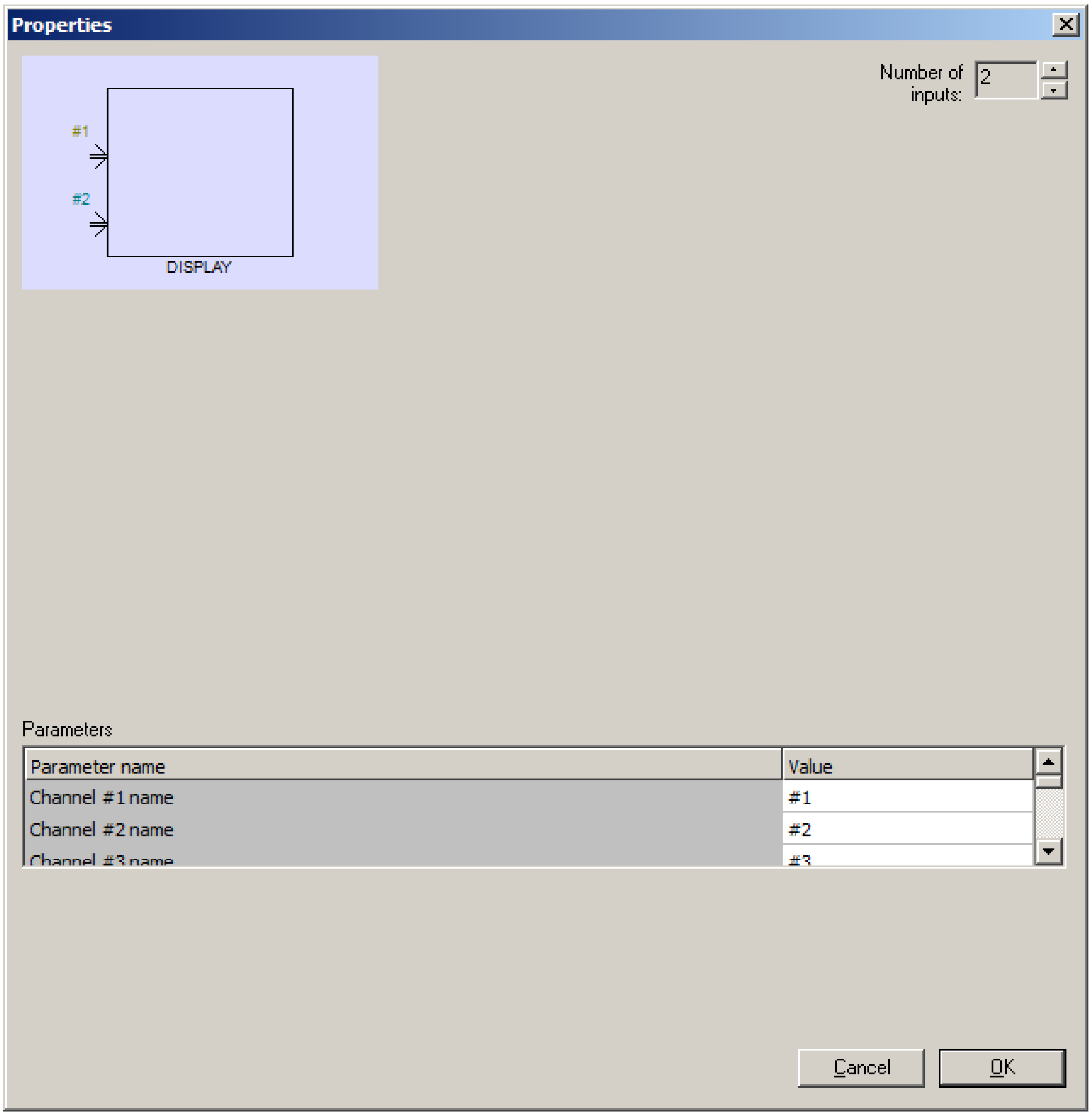
"Number of scans processed for each read": number of scans to transfer from the source system to the PC. This parameters may be changed according to the scan rate.

"Number of values stored in object for each channel": the object store in its internal memory the number of values selected in this parameter. Example: for a scan rate of 10Hz, if this parameter is equal to 10000 then values corresponding to a 1000 seconds record time will be stored.

Display



This object permits to display datas. It can be used as a scope or for displaying datas evolving during a long time.



Number of inputs: 2

Parameters

Parameter name	Value
Channel #1 name	#1
Channel #2 name	#2
Channel #3 name	#3

"Number of inputs" permits to define the number of channels to display (maximum 4).

Parameters

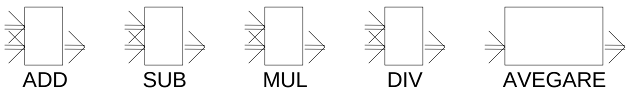
"Channel names": defines name displayed for each channel, these names will be also used for columns names when exporting to MS Excel.

"Minimum and maximum values": define values for scaling datas before displaying. Eg. if mini=0 and maxi=5 then 0 will be located at the bottom of the display area and 5 to the top of the area.

"Number of values stored in object for each channel": the object store in its internal memory and displays the number of values selected in this parameter. The right side of the object displays the most recent datas.

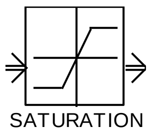
Synchronization: permits to define the synchronization mode of the datas, a synchronization level and an edge type. The displayed is synchronized (like the trigger function of a scope) when the value acquired on the selected channel crosses the selected level with the selected edge. The "Space between 2 compared values for triggering" parameter defines that the trigger test is made between scans spaced for avoiding false triggering on a dirty signal.

Math



These blocks process math calculation on datas flow.

Saturation



This clock performs a saturation on a datas flow by using min and max parameters.

Filter

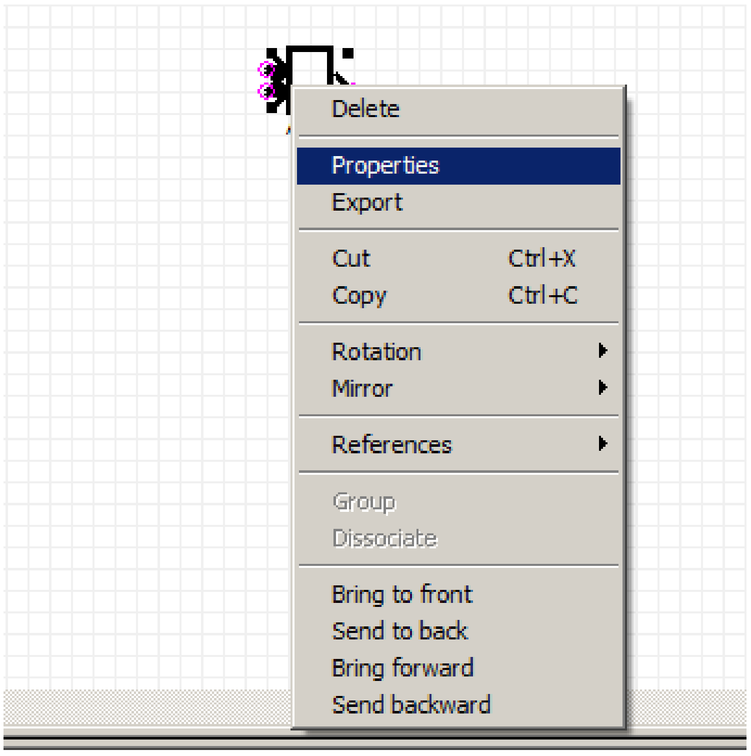
This block performs a filter (average calculation method).

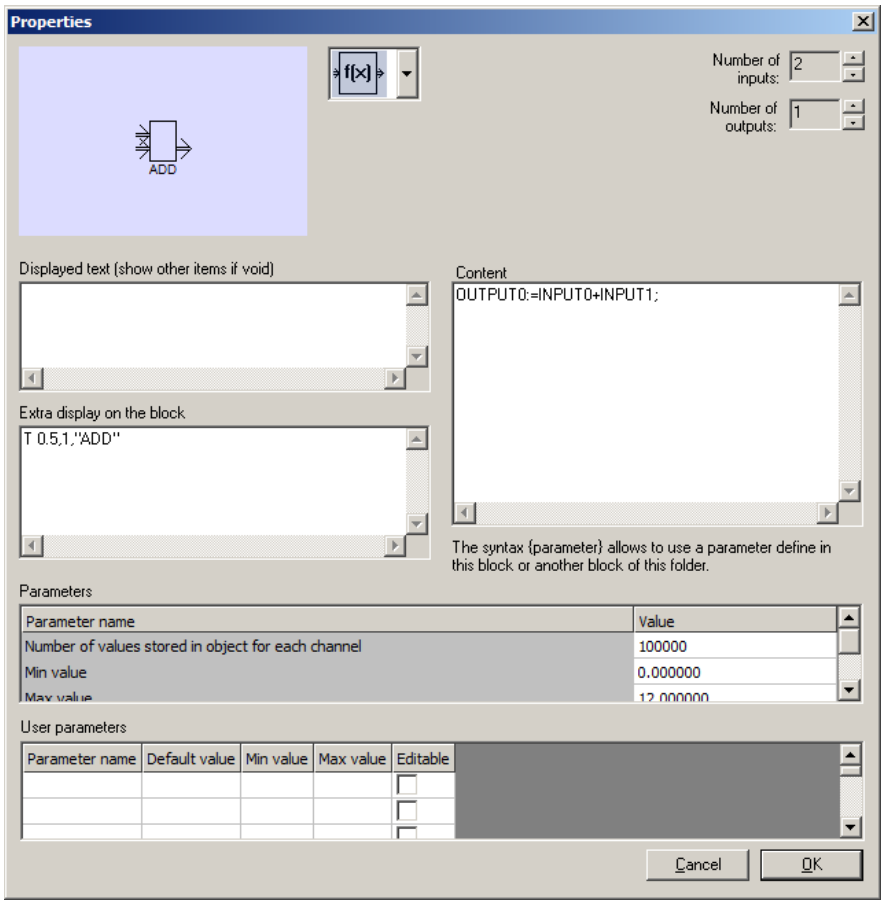
To Automlab / Avegare

Calculates the average on scans stored on the object and put the result to the AUTOMLAB output.

User defined blocks

The user defined blocks can be created by using ST literal language or C code. The user defined blocks can have up to 4 input channels and up to 4 output channels. The predefined blocks use internally the user blocks. To see whole parameters of those blocks, let the Shift key down while opening the properties dialog box. Example:




Properties

Number of inputs: 2

Number of outputs: 1

ADD

Displayed text (show other items if void)

Extra display on the block

T 0.5,1,"ADD"

Content

OUTPUT0:=INPUT0+INPUT1;

The syntax {parameter} allows to use a parameter define in this block or another block of this folder.

Parameters

Parameter name	Value
Number of values stored in object for each channel	100000
Min value	0.000000
Max value	12.000000

User parameters

Parameter name	Default value	Min value	Max value	Editable
				☐
				☐
				☐

Cancel OK

Datas flow

The principle of operation of the data flow is as follows in Automdaq: each acquisition of a data packet on all the channels of a system made by an "Acquisition" object, the packet is transmitted to each object connected to each output connection, the transmission is performed in series on the objects connected to the object and so on until the end of the chain. Each object has an internal buffer whose length can be changed in the settings of objects. If this buffer cannot contain the data transmitted while the oldest data is deleted.

The code written in the user defined blocks is executed each time a data packet is arriving.

User defined block using ST language

The literal code written for this type of block is an equation defining the relationship between one or more output flows and one or more input flows.

The code is only able to operate on the last received packet. This limitation is only for this type of user defined block. The C language block is able to access to all the datas stored into the object (see below).

The equation is as following:

output flow=input flow

Eg. :

OUTPUT0:=INPUT0+INPUT1;

which means: first output flow = first input flow + second input flow

"OUTPUTx" syntax (x between 0 and 3) is used to reference output flows from 1 to 4.

"INPUTx" syntax (x between 0 and 3) is used to reference input flows from 1 to 4.

"ETIME" syntax is used to reference elapsed time in seconds since the last code execution.

"TIME" syntax is used to reference time elapsed since the project is running.

User defined block using C language

The C code written for this type of block defines a function whose receives pointers on each input and output flows together with arriving packet length and stored datas length.

The function definition is:

```
__declspec(dllexport) void compute2(
float *fi0,
float *fi1,
float *fi2,
float *fi3,
float *fo0,
float *fo1,
float *fo2,
float *fo3,
unsigned len,
unsigned fullen,
float ETIME,
float TIME,
int b0)
```

fi0 to fi3: pointers to the beginning of each input flows. If a channel is not used, the pointer value is NULL.

len: number of scans of the incoming packet for each channel (same length for all the channels).

fo0 to fo3: pointers to the buffer of each output flows. If an output flow is not used, the pointer value is NULL. Each output flow buffer has a size of 'len' values (datas size of outgoing data is the same than size of incoming datas).

fullen : length (in number of values) of internal buffer for each channel.

ETIME : elapsed time between 2 calls to the "compute2" function

TIME : elapsed time since the project is running

b0 : 1 in the first call, then 0

Remarks:

1- when the "compute2 function is called, the incoming packet len counted in fullen.

Example of len and fullen evolution for a 100 scans packet length and a 500 scans intenal buffer length.

Function call #	len	fullen
1	100	100
2	100	200
3	100	300
4	100	400
5	100	500
6 and following	100	500

2- internal buffers values can be reached by decrementing pointers fi0 to fi3.

Parameters

The "user parameters" area of the objects properties permits to define parameters that can be defined by the user of the block.

The {name of the parameter} syntax must be used to reference the parameter in the code.

Code example: generating of the average of an input flow to an output flow.

```
__declspec(dllexport) void compute2(float *fi0,float *fi1,float *fi2,float *fi3,float *fo0,float *fo1,float *fo2,float *fo3,unsigned len,unsigned fullen,float ETIME,float TIME,int b0)

{

unsigned count;

// Loop for generating values on the output flow

for(count=0;count<len;count++)

  {

  float sum; // Totalize the n previous values

  int count2;

  int flen;

  // Compute the length of the available datas for calculating the average

  flen=count+(fullen-len);

  sum=fi0[count];

  for(count2=1;count2<=flen;count2++)

    {

    sum+=fi0[count-count2];

    }

  // Compute the average: total/number of values

  fo0[count]=sum/(flen+1);

  }

}
```

Hybrid user block "to Automlab"

This block permits to generate AUTOMLAB values on outputs. This type of blocks use C language. Keywords "OUTPUT0" to "OUTPUT3" are used to reference AUTOMLAB outputs 1 to 4.

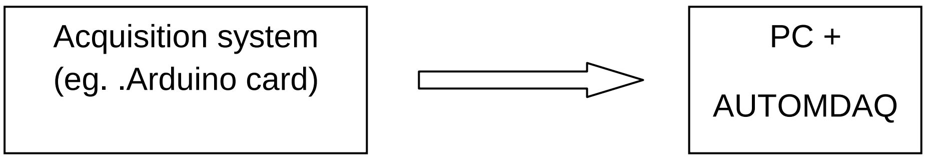
Example of code which is calculating an average:

```
float sum=0;

BEGINLOOP

sum=sum+INPUT0;

ENDLOOP

OUTPUT0=sum/LOOPLEN;
```

Specific informations regarding data acquisition from serial link

The "COM PORT" mode of the "Acquisition" object permits to use a serial link for acquiring datas.

Concept



In this mode, the scan rate is automatically computed regarding the incoming flow on the serial link.

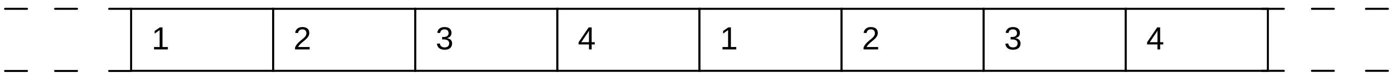
The acquisition system have to send datas continuously and regularly on the serial link.

Serial link format

The format is 8 data bits, 1 stop bit, and NO parity. The baud rate is setup in the acquisition object properties.

Datas format

Each scan is codes as a float value on 4 bytes (IEEE 745). If more than one channel is used, datas have to be sent alternatively for each channels. Example for 4 channels:



Synchronization

The actual protocol has no synchronization byte (only datas are transmitted), for this reason, AUTOMDAQ project must be in run mode before datas flow start to be send for avoiding a bas synchronization.

Example using an Arduino card

Arduino side program (example for one channel):

```
unsigned int convertedValue;  // variable to store the converted value

float voltageValue;           // variable to store the voltage applied to the analog pin

byte analogPin = 0;           // analog input pin


void setup(void) {

  Serial.begin(115200);

}


void loop(void){

  convertedValue = analogRead(analogPin);

  voltageValue = 5.0 * convertedValue / 1023;

  Serial.write((byte *)&voltageValue,4);

}
```

This program acquires one analog value, compute the value in volts and send the value as IEEE floating value on the serial link with a 115200 bauds speed (maximum speed compatible with Arduino and PC serial links).

Arduino side program (example for 2 channels):

```
unsigned int convertedValue; // variable to store the converted value

float voltageValue;          // variable to store the voltage applied to the analog pin

byte analogPin = 0;          // analog input pin

void setup(void) {

  Serial.begin(115200);

}

void loop(void){

  convertedValue = analogRead(analogPin);

  voltageValue = 5.0 * convertedValue / 1023;

  Serial.write((byte *)&voltageValue,4);

  convertedValue = analogRead(analogPin+1);

  voltageValue = 5.0 * convertedValue / 1023;

  Serial.write((byte *)&voltageValue,4);

}
```

Calculation of the scan rate: significative consuming time action is time taken for sending datas over serial link. With a 115200 bauds speed, time for sending 4 bytes (one scan) is:

115200 / 10 (8 data bits, 1 start bit, 1 stop bits) = 11520 bytes/second

So, the maximum scan rate is: 11520/4 4 (4 bytes per scan) = 2880 Hz.

For 2 channels, maximum scan rate is 1440 Hz, 960 Hz for 3 channels and 720 Hz for 4 channels.