Via Monte Nero, 40/B – 21049 TRADATE (VA) ITALY
Phone: +39 (0)331841070 Fax:+39 (0)331841950 - e-mail:datexel@datexel.it - www.datexel.it

UNI EN ISO 9001:2000

User Guide **Dev9K** v. 2.0

Integrated Development Environment to design and debug the applications based on the DAT9000 series controllers.
NOTE: this document is suitable for devices with these firmware versions:
9BB4, 9BB5, 9BA8, 9560, 9570

INDEX

1.1 - GENERAL DESCRIPTION

Dev9K is an Integrated Development Environment running under the Windows® Operative System that allows to design and debug the applications based on the DAT9000 series control devices. With Dev9K it is possible to set the DAT9000 series controllers to execute I/O read and write operations (DAT3000, DAT8000, DAT10000 series), mathematical and logic operations and timers. Moreover it is possible to read and write in real time the Internal Registers of the Controller or connect it directly to the slave devices connected to its Modbus Master Port.

1.2 – MINIMUM SYSTEM REQUIREMENTS

Operative System	Windows® 7 / 8 / 8.1 / 10
Available Hard Disk memory	50 MB

1.3 - PROCEDURE OF INSTALLATION

Close eventual active or background applications.
Insert the CD-ROM of installation in the driver.
Wait for the Autorun window opening.
If the Autorun function is disabled, open the CD-ROM and execute the installation file at the path:
<CD Driver>:/doc/download.html
Click on the Tools button.
Click on the “download” button in the DAT9000 section.
Follow the Installation Wizard.

1.4 - TERMINOLOGY

Description of terms and abbreviations used in the IDE and inside this manual:

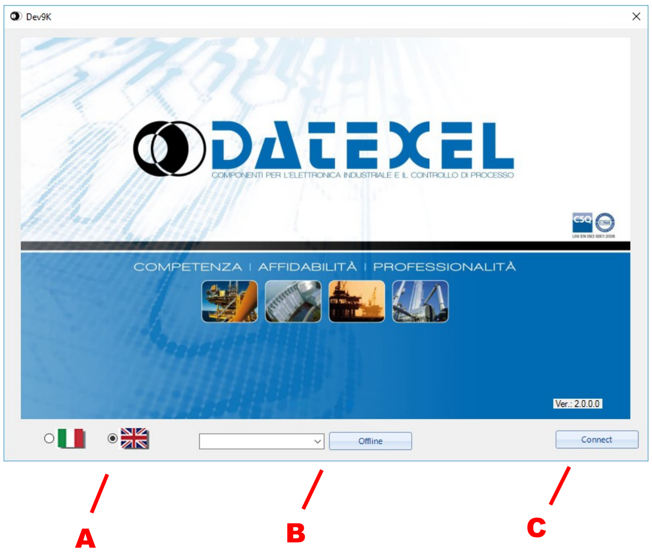
Controller	Device of the DAT9000 series.
Integrated Development Environment IDE	Instructions set to create, to debug and to test the Program.
Program	List of functions executed by the Controller.
Function Block F.B.	Each block constituting the Main Program. Each Function Block can contain a function.
Function	Logical, mathematical or flow operation executed from the Controller.
Argument	Each parameter contained in a function.
Register	Position of a variable in the volatile memory of Controller.
Retentive Register	Position of a variable in the non-volatile memory of Controller.

2.1 – OPEN AND INITIALIZE THE PROGRAM

Once Dev9k installed, to run it double click on the program icon **(Pict.2.1)** and give the administrator permissions.
The initialization window of the program will appear **(Pict.2.2)**
It is possible to:
- set the language **(Pict.2.2-A);**
- enter in offline mode choosing a device from the drop down list and clicking on button "*Offline*" **(Pict.2.2-B)**;
- enter in the search window to connect directly to a device **(Fig.2.2-C)**

Pict. 2.1

Pict. 2.2


Dev9K
DATEXEL
COMPONENTI PER L'ELETTRONICA INDUSTRIALE E IL CONTROLLO DI PROCESSO
COMPETENZA | AFFIDABILITÀ | PROFESSIONALITÀ
Ver.: 2.0.0.0
Offline
Connect
A
B
C

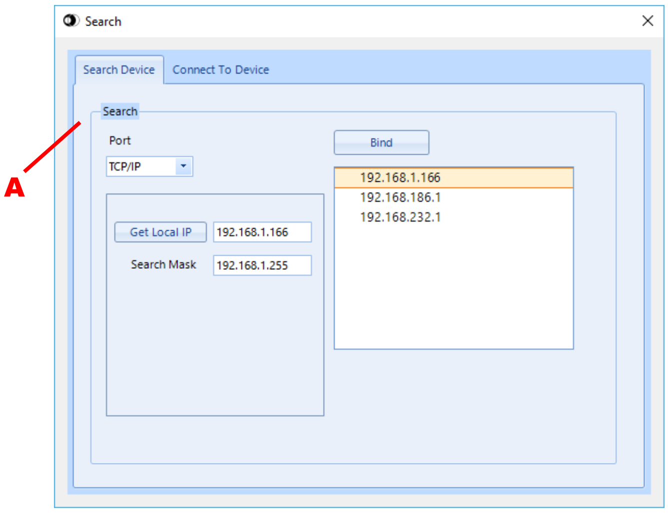
2.2 – CONNECT AND SEARCH DEVICE

Into the Search window there are two modes to connect to device:

- Search a device in the Net and connect to it **(Pict.2.3-A)**

- Connect directly to the device by its parameters (Ethernet or RS485/uUSB), inserting them in the fields and clicking the button "*OK*" **(Pict.2.3-B)**

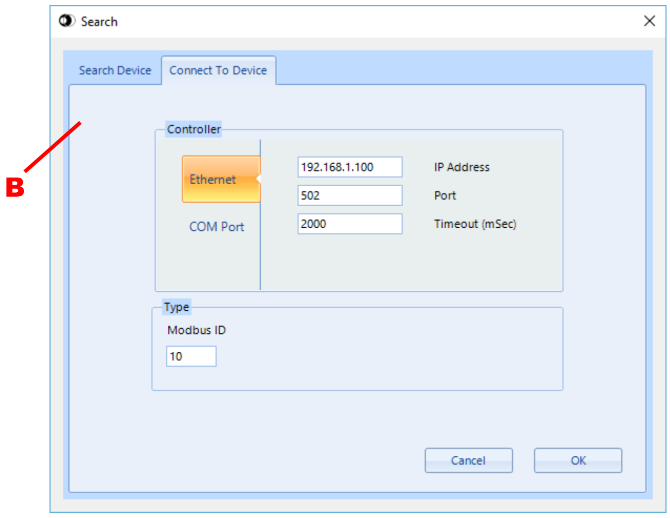
For details consult the paragraph 6.1 and 6.2.



Search
Port: TCP/IP
Get Local IP: 192.168.1.166
Search Mask: 192.168.1.255

192.168.1.166
192.168.186.1
192.168.232.1

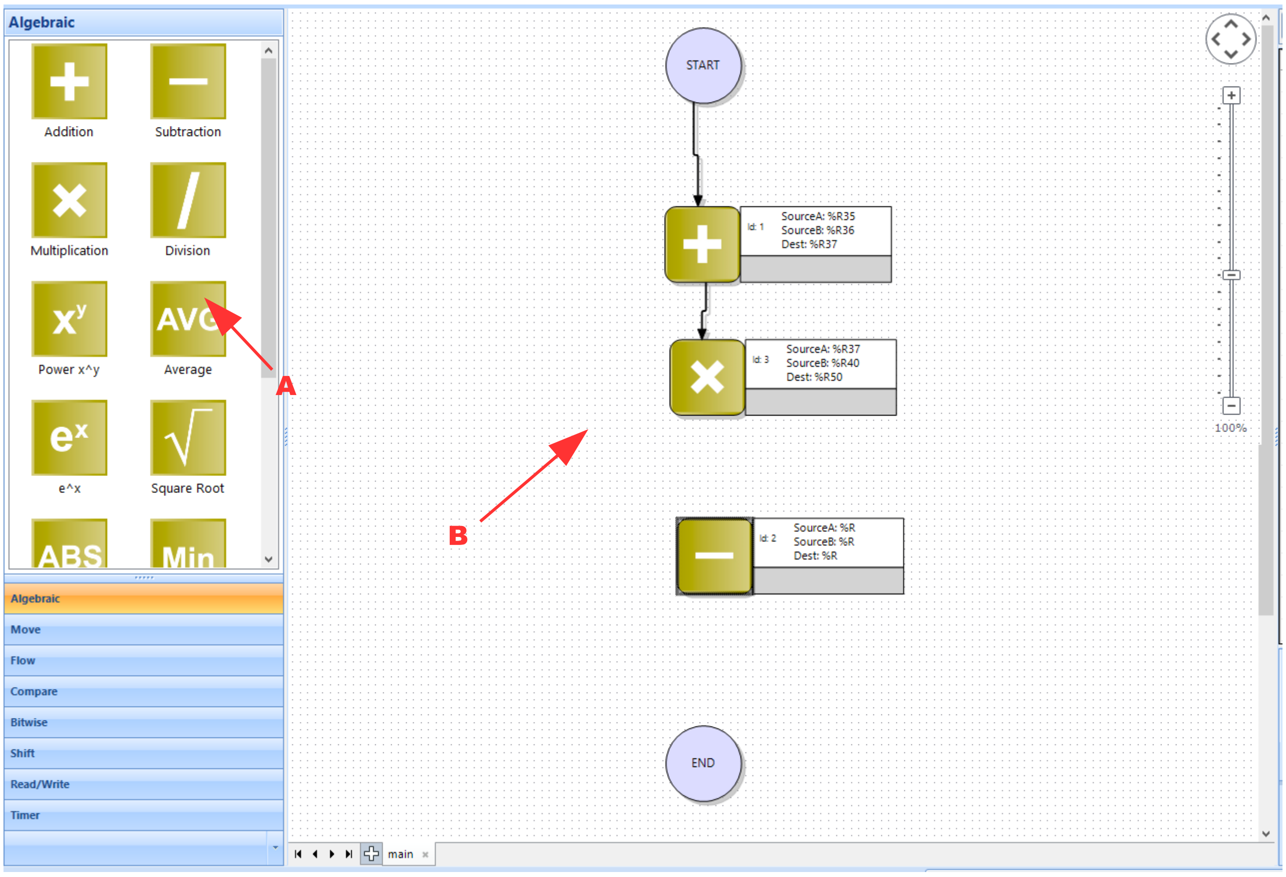
Pict. 2.3



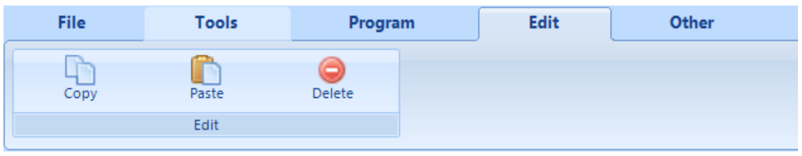
Controller
IP Address: 192.168.1.100
Port: 502
Timeout (mSec): 2000

Type
Modbus ID: 10

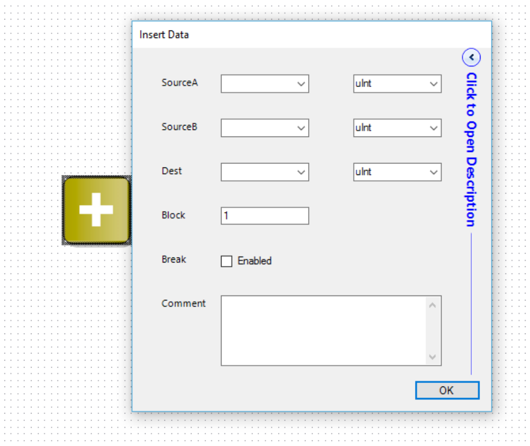
2.3 – CREATION OF A DIAGRAM



Pict. 2.4



Pict. 2.5



Pict. 2.6

- Insertion of function block

It is possible to insert a new function into the diagram dragging it from function block list **(Pict.2.4-A)** to working area of diagram **(Pict.2.4-B).** Once the drag and drop has been completed it will be opened a window for the insertion of data in the function **(Pict.2.6)** The functions in the list are grouped into functional groups.

- Move the function blocks in the working area

To move one or more function blocks select them (left click of the mouse) and then drag them in the desired position.

- Creation of the links between a block and another

To create a link between a block and another and control the flow of the diagram, the block from which the link starts from must be deselected. So click the left mouse button on the block and drag the link to the destination block.

- Modify the data of a function block

To modify data in a function block, double-click of the left mouse button on the desired function block. As result, the data entry window will open **(Pict.2.6).**

- Delete the function blocks

To delete one or more function blocks select the desired blocks and use one of following methods:
- press the button *CANC* of the keyboard
- click the right button of the mouse and then select *Delete*
- in the menu bar, in the section *Edit,* select the button *Delete* **(Pict.2.5)**

- Copy the function blocks

To copy one or more function blocks select the desired blocks and use one of following methods:
- press the buttons *CTRL+C* of the keyboard
- click the right button of the mouse and then select *Copy*
- in the menu bar, in the section *Edit,* select the button *Copy* **(Pict.2.5)**

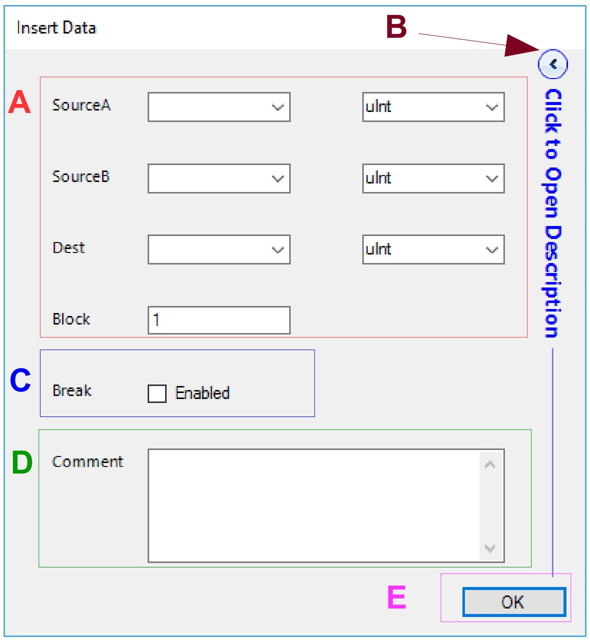
- Paste the function blocks

To paste one or more function blocks be sure to have copied them before and use one of following methods:
- press the buttons *CTRL+V* of the keyboard
- click the right button of the mouse and then select *Paste*
- in the menu bar, in the section *Edit,* select the button *Paste* **(Pict.2.5)**

2.4 INSERT DATA IN A FUNCTION

When the user modifies a Function Block or inserts another one, the data entry window will open **(Pict.2.7)**, that allows to set the arguments relative to the function selected.

Usually the first arguments **(Pict.2.7-A)** of the window identify and set data to the function (number of register, type of register, values, etc).
The structure is the same for each function: on the left there are the labels that suggest what is the relative argument, in the centre the values that the arguments take and on the right the drop down list with types of data. Some functions may have some additional buttons that allow to interact with the insertion of data.
In general:
Source: is the input data of the function. This can be a register or a constant. When the source is selected like constant(*K_Flt*), the insertion field become green **(Pict.2.8)**
Dest: is usually the register where the result of the function is written
Block: is usually the number of repetition of the operation in consecutive registers (Source and Dest)
Mask: is usually the desired mask to apply to the relative *Source* or *Dest*



Pict. 2.7

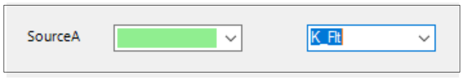
In the entry window there is popup menu on side **(Pict.2.7-B)** that describes the arguments of the function.

All the functions, in addition to the first arguments, have also:

Break **(Pict.2.7-C):** this argument allows to insert a break into the function. With this, after the download of the program, when the device is in debug and receives the command *Run to Break* it stops on the relative function with break and points out it.

Comment **(Pict.2.7-D):** this argument allows to insert a comment into the function block. By this the user can identify easily the block function in the diagram.

Pict. 2.8

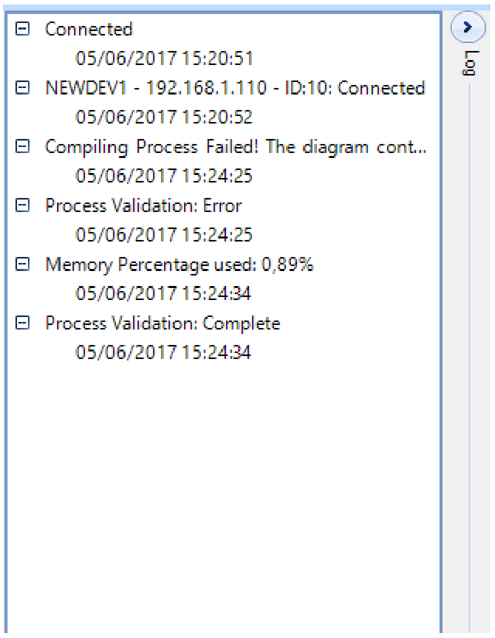


After the insertion of all data required for the selected function click the button OK **(Pict.2.7-E).** If a field is empty, it will appear the message "*Found an empty field, continue?"*

Refer to the section "Function Description" for how to insert the functions.

2.5 – LOG PANEL AND COMPILE

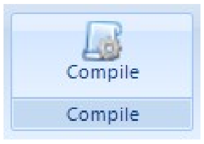
The Log Panel contains the information of connection and validation of the program **(Pict.2.9).**
After the insertion of function blocks, it is possible to compile the program clicking the button “*Compile* ” in the menu bar **(Pict.2.10)**. The result of the validation will be inserted in the log panel.
If there is an error in the validation it will be displayed in the panel with the phrase “*Process Validation: Error*”. If the validation ends correctly, in addition to the result (“*Process Validation: Complete”*), it will be displayed the percentage of Eeprom used by the program.



```
⊟ Connected
     05/06/2017 15:20:51
⊟ NEWDEV1 - 192.168.1.110 - ID:10: Connected
     05/06/2017 15:20:52
⊟ Compiling Process Failed! The diagram cont...
     05/06/2017 15:24:25
⊟ Process Validation: Error
     05/06/2017 15:24:25
⊟ Memory Percentage used: 0,89%
     05/06/2017 15:24:34
⊟ Process Validation: Complete
     05/06/2017 15:24:34
```

Pict. 2.9

<u>NOTE:</u>
It is possible to insert until 255 function blocks or until the available Eeprom Memory Space has been filled.



Pict. 2.10

3.1 – INTERNAL REGISTERS

The Internal Memory of the Controller is accessible by the register table **(Pict.3.1)** and it is composed of a 16 bit Register series divided as follows:

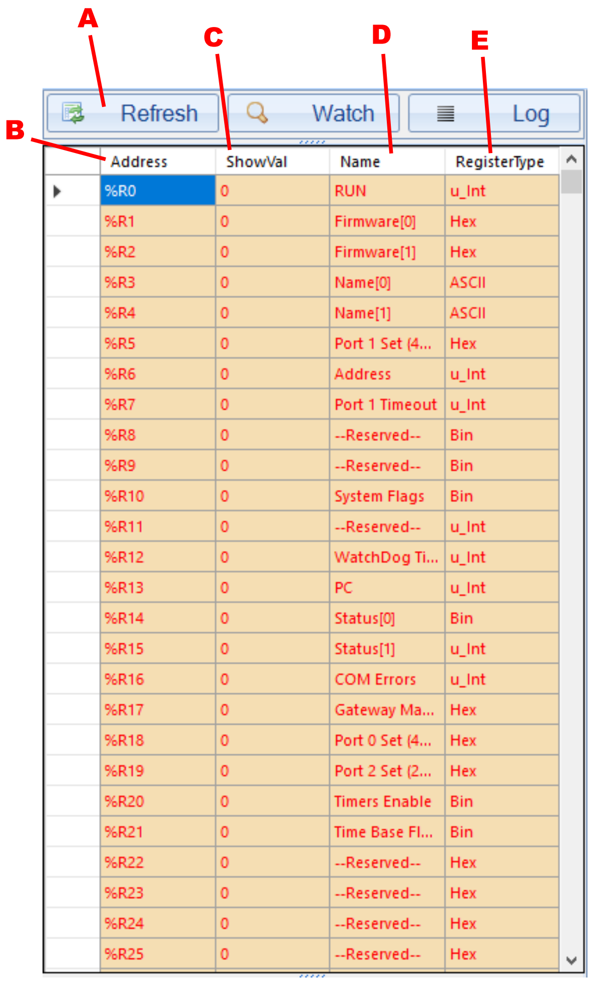
System Registers: contain the information about the status Controller. They are identified by orange rows.

General Purpose Registers: can be used in the Program to move the data or to execute calculation functions. Their rows have not colours.

Retentive Registers: can be used from the Program to move the data or to execute calculation functions. These Registers are saved in EEprom each time their values change and they are uploaded when the Controller is powered-on. They are identified by blue rows.

To update the Register's value, click the button "Refresh" **(Pict.3.1-A)**.
For each Register it is visualized:
<u>Address</u>: The register address **(Pict.3.1-B)**
<u>ShowVal</u>: The value contained into the Register **(Pict.3.1-C)**
<u>Name</u>: The register name **(Pict.3.1-D)**
<u>RegisterType</u>: The register data format **(Pict.3.1-E)**

To modify the name or the value of a Register, double click on the row of table regarding the Register.
Inside the "*Set Register*" window **(Pict.3.2)** it is possible to set the name of the Register (only for the *General Purpose Register* or *Retentive Registers*), to force the value contained in the Register (only if the Controller is connected) and to set the type of data format of the Register.



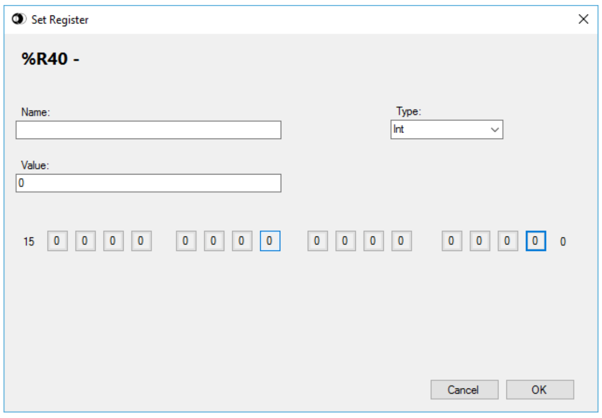
Address	ShowVal	Name	RegisterType
%R0	0	RUN	u_Int
%R1	0	Firmware[0]	Hex
%R2	0	Firmware[1]	Hex
%R3	0	Name[0]	ASCII
%R4	0	Name[1]	ASCII
%R5	0	Port 1 Set (4...	Hex
%R6	0	Address	u_Int
%R7	0	Port 1 Timeout	u_Int
%R8	0	--Reserved--	Bin
%R9	0	--Reserved--	Bin
%R10	0	System Flags	Bin
%R11	0	--Reserved--	u_Int
%R12	0	WatchDog Ti...	u_Int
%R13	0	PC	u_Int
%R14	0	Status[0]	Bin
%R15	0	Status[1]	u_Int
%R16	0	COM Errors	u_Int
%R17	0	Gateway Ma...	Hex
%R18	0	Port 0 Set (4...	Hex
%R19	0	Port 2 Set (2...	Hex
%R20	0	Timers Enable	Bin
%R21	0	Time Base Fl...	Bin
%R22	0	--Reserved--	Hex
%R23	0	--Reserved--	Hex
%R24	0	--Reserved--	Hex
%R25	0	--Reserved--	Hex

Pict. 3.1

3.2 - DATA FORMAT

Each *General Purpose or Retentive Register* can be read or written using one of the following data format :

u_Int	16 bit Unsigned Integer (0 ÷ 65535)
Int	16 bit Signed Integer (-32768 ÷ +32767)
u_Long	32 bit Unsigned Long (0 ÷ 4,294,967,295)
Long_	32 bit Signed Long (-2,147,483,648 ÷ +2,147,483,647)
Float	32 bit Floating Point
Hex	16 bit Unsigned Integer visualized as Hexadecimal characters (0000 ÷ FFFF)
ASCII	16 bit Unsigned Integer visualized as ASCII characters
Bin	16 bit Unsigned Integer visualized as binary code



%R40 -

Name:

Type: Int

Value: 0

15 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Pict. 3.2

NOTE: the 32 bit registers require the position of 2 registers and the second is identified as "*used*"
For the registers in ASCII format, when more than 16 bit are occupied (more than a register that is more than 2 characters), starting from the second register the registers are identified as "*String_Ram*".

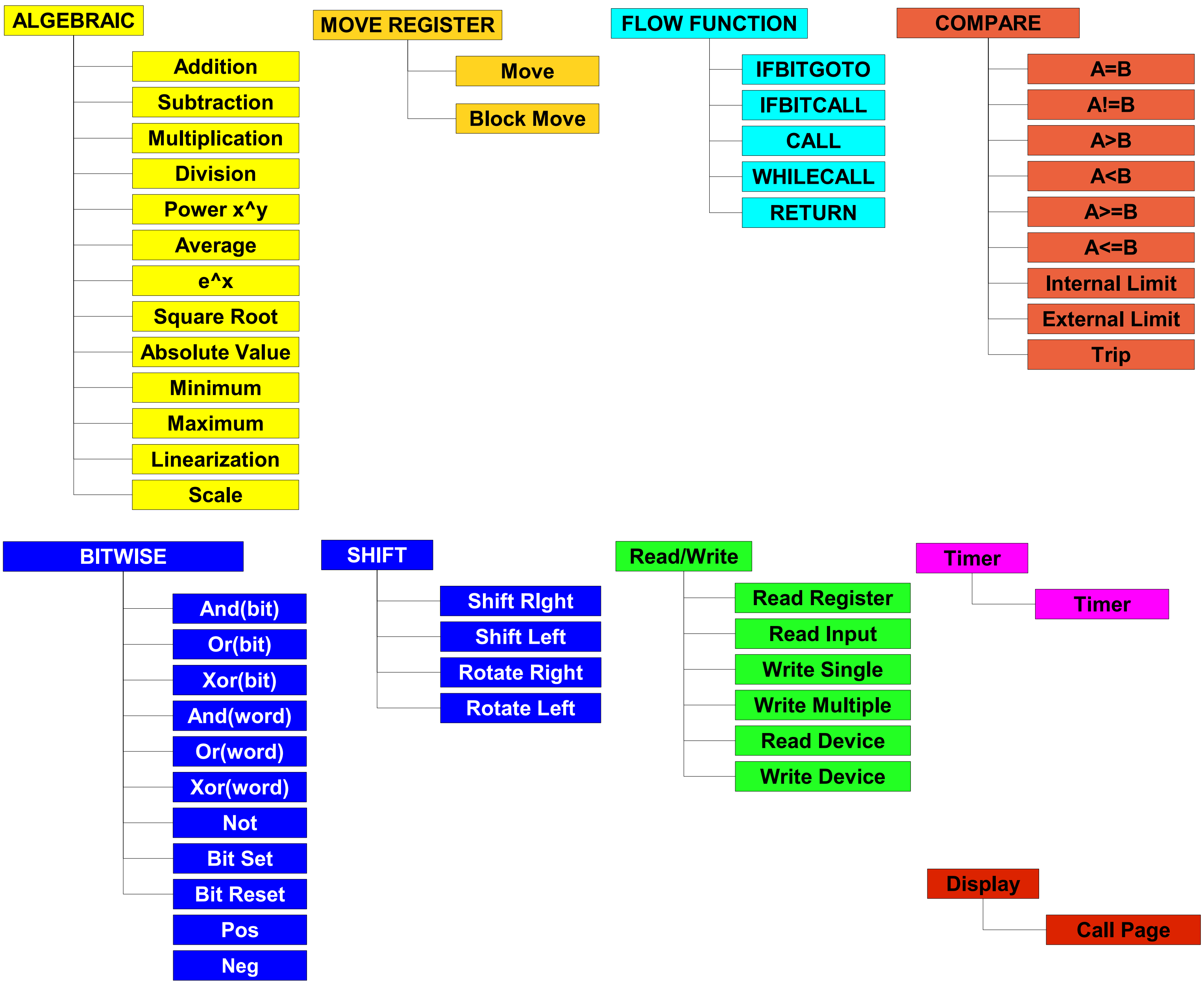
3.3 – MAPPING REGISTERS

	Device						
Register	DAT9000-DL DAT9000-USB	DAT9000IO-USB	DAT9000-DL-IO	DAT 9011-DL	DAT 9011-USB	DAT 9550	DAT 9550-AI
%R0	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--
%R1	Firmware [0]	Firmware [0]	Firmware [0]	Firmware [0]	Firmware [0]	Firmware [0]	Firmware [0]
%R2	Firmware [1]	Firmware [1]	Firmware [1]	Firmware [1]	Firmware [1]	Firmware [1]	Firmware [1]
%R3	Name [0]	Name [0]	Name [0]	Name [0]	Name [0]	Name [0]	Name [0]
%R4	Name [1]	Name [1]	Name [1]	Name [1]	Name [1]	Name [1]	Name [1]
%R5	Port 1 [Settings]	Port 1 [Settings]	Port 1 [Settings]	Port 1 [Settings]	Port 1 [Settings]	Port 1 [Settings]	--Reserved--
%R6	Node ID	Node ID	Node ID	Node ID	Node ID	Node ID	Node ID
%R7	Port 1 [Timeout RX]	Port 1 [Timeout RX]	Port 1 [Timeout RX]	Port 1 [Timeout RX]	Port 1 [Timeout RX]	Port 1 [Timeout RX]	--Reserved--
%R8	--Reserved--	Digital Inputs	Digital Inputs	Digital Inputs	Digital Inputs	Function Keys	Function Keys
%R9	--Reserved--	Digital Outputs	Digital Outputs	Digital Outputs	Digital Outputs	Actual Page	Actual Page
%R10	System Flags	System Flags	System Flags	System Flags	System Flags	System Flags	System Flags
%R11	--Reserved--	PowerUp Safe Outputs	PowerUp Safe Outputs	PowerUp Safe Outputs	PowerUp Safe Outputs	Display Options	Display Options
%R12	Watchdog Timeout	Watchdog Timeout	Watchdog Timeout	Watchdog Timeout	Watchdog Timeout	Watchdog Timeout	Watchdog Timeout
%R13	PC	PC	PC	PC	PC	PC	PC
%R14	Status [0]	Status [0]	Status [0]	Status [0]	Status [0]	Status [0]	Status [0]
%R15	Reset Timers	Reset Timers	Reset Timers	Reset Timers	Reset Timers	Reset Timers	Reset Timers
%R16	COM Errors	COM Errors	COM Errors	COM Errors	COM Errors	COM Errors	COM Errors
%R17	Gateway Mask [L-H]	Gateway Mask [L-H]	Gateway Mask [L-H]	Gateway Mask [L-H]	Gateway Mask [L-H]	Gateway Mask [L-H]	--Reserved--
%R18	Port 0 [Settings]	Port 0 [Settings]	Port 0 [Settings]	Port 0 [Settings]	Port 0 [Settings]	Port 0 [Settings]	Port 0 [Settings]
%R19	Port 2 [Settings]	Port 2 [Settings]	Port 2 [Settings]	Port 2 [Settings]	Port 2 [Settings]	Port 2 [Settings]	--Reserved--
%R20	Timers Enable	Timers Enable	Timers Enable	Timers Enable	Timers Enable	Timers Enable	Timers Enable
%R21	Time Base Flags	Time Base Flags	Time Base Flags	Time Base Flags	Time Base Flags	Time Base Flags	Time Base Flags
%R22-%R25	RTC	RTC	RTC	RTC	RTC	RTC	RTC
%R26	--Reserved--	--Reserved--	--Reserved--	Analog Input 0	Analog Input 0	--Reserved--	Analog Input
%R27	--Reserved--	--Reserved--	--Reserved--	Analog Input 1	Analog Input 1	--Reserved--	--Reserved--
%R28	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--
%R29	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--
%R30	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--
%R31	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--
%R32	--Reserved--	--Reserved--	--Reserved--	Analog Output 0	Analog Output 0	--Reserved--	--Reserved--
%R33	--Reserved--	--Reserved--	--Reserved--	Analog Output 1	Analog Output 1	--Reserved--	--Reserved--
%R34	--Reserved--	--Reserved--	--Reserved--	Input Type [1-0]	Input Type [1-0]	--Reserved--	Input Type
%R35	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose
||	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose
%R927	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose
%R928	--Reserved--	Freq input 0	Freq input 0	Freq input 0	Freq input 0	General Purpose	General Purpose
%R929	--Reserved--	Freq input 1	Freq input 1	Freq input 1	Freq input 1	General Purpose	General Purpose
%R930	--Reserved--	Freq input 2	Freq input 2	--Reserved--	--Reserved--	General Purpose	General Purpose
%R931	--Reserved--	Freq input 3	Freq input 3	--Reserved--	--Reserved--	General Purpose	General Purpose
%R932-933	--Reserved--	Counter input 0	Counter input 0	Counter input 0	Counter input 0	General Purpose	General Purpose
%R934-935	--Reserved--	Counter input 1	Counter input 1	Counter input 1	Counter input 1	General Purpose	General Purpose
%R936-937	--Reserved--	Counter input 2	Counter input 2	--Reserved--	--Reserved--	General Purpose	General Purpose
%R938-939	--Reserved--	Counter input 3	Counter input 3	--Reserved--	--Reserved--	General Purpose	General Purpose
%R940-959	--Reserved--	--Reserved--	--Reserved--	--Reserved--	--Reserved--	General Purpose	General Purpose
%R960	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose	Memory Registers	Memory Registers
||	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose	Memory Registers	Memory Registers
%R1023	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose	Memory Registers	Memory Registers
%R1024	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose		
||	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose		
%R1215	General Purpose	General Purpose	General Purpose	General Purpose	General Purpose		
%R1216	Memory Registers	Memory Registers	Memory Registers	Memory Registers	Memory Registers		
||	Memory Registers	Memory Registers	Memory Registers	Memory Registers	Memory Registers		
%R1219	Memory Registers	Memory Registers	Memory Registers	Memory Registers	Memory Registers		
%R1220-1223	Memory Registers	Memory Registers	Log Dir	Mem. Reg.	Log Dir		
%R1224	Memory Registers	Memory Registers	Memory Registers	Memory Registers	Memory Registers		
||	Memory Registers	Memory Registers	Memory Registers	Memory Registers	Memory Registers		
%R1280	Memory Registers	Memory Registers	Memory Registers	Memory Registers	Memory Registers		

IMPORTANTE: per la descrizione dettagliata dei registri, fare riferimento alla User Guide relativa a ciascun dispositivo.

4.1 – FUNCTION LIST

Functions selection tree, divided per functional group



4.2 – FUNCTIONS DESCRIPTION

Addition	Calculates the sum of two values
	Calculates the sum between an Internal Register and a constant or between two Internal Registers.

Arguments:

SourceA	Constant or Internal Register relative to the first operator
SourceB	Constant or Internal Register relative to the second operator
Dest	Internal Register wherein the result is written
Block	Number of repetition of the operation in consecutive registers (Source and Dest)

Subtraction	**Calculates the difference of two values**

Calculates the difference between an Internal Register and a constant or between two Internal Registers.

Arguments:

SourceA	Constant or Internal Register relative to the first operator.
SourceB	Constant or Internal Register relative to the second operator.
Dest	Internal Register wherein the result is written.
Block	Number of repetition of the operation in consecutive registers (Source and Dest)

Multiplication	**Calculates the multiplication of two values**

Calculates the multiplication between an Internal Register and a constant or between two Internal Registers

Arguments:

SourceA	Constant or Internal Register relative to the first operator.
SourceB	Constant or Internal Register relative to the second operator.
Dest	Internal Register wherein the result is written.
Block	Number of repetition of the operation in consecutive registers (Source and Dest)

Division	**Calculates the division between two values.**

Calculates the division between an Internal Register and a constant or between two Internal Registers.

Arguments:

SourceA	Constant or Internal Register relative to the first operator.
SourceB	Constant or Internal Register relative to the second operator.
Dest	Internal Register wherein the result is written.
Block	Number of repetition of the operation in consecutive registers (Source and Dest)

Power x^y	**Performs exponentiation between two values.**

Performs exponentiation between a register and a constant or between two registers.

Arguments:

SourceX	Constant or Internal Register relative to the base of the exponentiation
SourceY	Constant or Internal Register relative to the exponent of the exponentiation
Dest	Internal Register wherein the result is written
Block	Number of repetition of the operation in consecutive registers (Source and Dest)

Average	**Calculates the Arithmetic mean of N values.**

Calculates the Arithmetic mean of N Internal Registers values starting from Source (sum of the values / N).

Arguments:

Source	Address of the Internal Register containing the first value
N	Number of Internal Registers of which calculate the mean.
Dest	Internal Register wherein the result is written.

e^x **Performs exponential function of a value**

Performs the value of the exponential function with base e (Euler's number) and exponent contained in an Internal Register

Arguments:

SourceX	Internal Register relative to the exponent
Dest	Internal Register wherein the result is written.

Square Root **Calculates the square root of a value**

Calculates the square root of a value contained in an Internal Register.

Arguments:

Source	Internal Register relative to the input
Dest	Internal Register wherein the result is written.

Absolute Value **Calculates the absolute value of a value**

Calculates the absolute value of a value contained in an Internal Register.

Arguments:

Source	Internal Register relative to the input
Dest	Internal Register wherein the result is written.

Minimum **Calculates the minimum value of N registers**

Calculates the minimum value of N Internal Registers values starting from Source

Arguments:

Source	Address of the Internal Register containing the first value
N	Number of Internal Register within find the minimum value
Dest	Internal Register wherein the result is written.

Maximum **Calculates the maximum value of N registers**

Calculates the maximum value of N Internal Registers values starting from Source

Arguments:

Source	Address of the Internal Register containing the first value
N	Number of Internal Register within find the maximum value
Dest	Internal Register wherein the result is written.

Linearization — Calculates a value in function of a linearization table

Calculates the Linearization of a value in function of the linearization table selected. Refer to the section 5.1 “Insertion of Linearization Tables” for more information.

Arguments:

Source	Internal Register containing the value to linearize.
Function	Name of the Linearization table to be followed. The button *Table* opens the window “Tables”
Dest	Internal Register wherein the result is written.

Scale — Executes the proportional scaling of a value of a register

Executes the proportional scaling of a value contained in an Internal Register referring to the input and output ranges. The input range is defined by the limits Zero In and Span In. The output range is defined by the limits Zero Out and Span Out

Arguments:

Source	Internal Register relative to the input
Span In	Maximum value of the input range
Zero In	Minimum value of the input range
Dest	Internal Register relative to the output
Span Out	Maximum value of the output range
Zero Out	Minimum value of the output range

Move — Moves the value of an Internal Register or a Constant in an Internal Register

Writes in an Internal Register the value of a Constant (pre-set) or the value of another Register (copy). The value will be converted to the format selected for the Register of destination.

Arguments:

Source	Constant or Internal Register from which the value is read
Dest	Internal Register wherein the value is written
Block	Number of repetition of the operation in consecutive registers (Source and Dest)

Block Move — Moves a block of registers

Moves a block of N registers starting from Source in another block. It is possible to swap the 8 bit for the uInt registers and the 16 bit for uLong registers.

Arguments:

Source	Address of the Internal Register containing the first value
Number	Number of Internal Register to move
Dest	First Register in which for the quantity indicated in Number are moved the registers starting from Source

IFBITGOTO — Conditional jump executed in function of value of one bit

If the value of the bit set in the Source Register is 1, the next function executed by the program is the one connected with the green arrow *true*. If the value of the bit is 0, the next function executed by the program is the one connected with the red arrow *false*.

Arguments:

Source	Internal Register relative to the input
Bit	Bit that determines the jump (0÷15)

IFBITCALL	Call to a subroutine in function of value of one bit

If the value of the bit set in the Source Register is 1, the program executes a call to the page (subroutine) indicated in the field “*DestTrue*” and so to its first function after the *START.* If the value of the bit set in the Source Register is 0, the program does not execute any call and continues.
The called page must have an univocal name. It is possible to select one of the available pages from drop down list or to create another one writing a new name.

Arguments:

Source	Internal Register relative to the input
Bit	Bit that determines the call (0÷15)
DestTrue	Pointer to the page (subroutine)

CALL	Call to a subroutine

The program executes a call to the page (subroutine) indicated in the field “*Dest*” and so to its first function after the *START.* It executes a jump to the first function block of a subroutine. At the end of the subroutine (ended with the function Return), the program executes the function block after this. The called page must have an univocal name. It is possible to select one of the available pages from drop down list or to create another one writing a new name.

Arguments:

Dest	Pointer to the page (subroutine)

WHILECALL	Continuous call to a subroutine in function of value of one bit of a register

The program executes a continuous call to the page (subroutine) indicated in the field “*Dest*” until the value of the register *Source* is lower than a constant or another register indicated in the field *While<.*
If the register is equal or bigger, the program does not execute any continuous call and continues.
The called page must have an univocal name. It is possible to select one of the available pages from drop down list or to create another one writing a new name.

Arguments:

Source	Internal Register relative to the input
While<	Constant or Internal Register to compare to *Source*
Dest	Pointer to the page (subroutine)

RETURN	Return from a Subroutine

Indicates the end of a Subroutine. The Program will execute the Block after the one that has called the subroutine (function “*CALL*”, “*WHILECALL*”, “*IFBITCALL*”)

A=B	Comparison of value (A=B) between two registers or between a register and a constant

Executes a comparison between InputA and InputB (registers or constants). If the values are equals sets the bits of a destination register *Dest* in function of the setting of the parameter *Mask*.

Arguments:

InputA	Constant or Internal Register relative to the first input to compare
InputB	Constant or Internal Register relative to the second input to compare
Dest	Internal Register wherein the bits will be set
Mask	Mask used to set the register Dest (the button on side opens the window for the setting)

A!=B — Comparison of value (A!=B) between two registers or between a register and a constant

Executes a comparison between InputA and InputB (registers or constants). If the values are different sets the bits of a destination register *Dest* in function of the setting of the parameter *Mask*.

Arguments:

InputA	Constant or Internal Register relative to the first input to compare
InputB	Constant or Internal Register relative to the second input to compare
Dest	Internal Register wherein the bits will be set
Mask	Mask used to set the register Dest (the button on side opens the window for the setting)

A>B — Comparison of value (A>B) between two registers or between a register and a constant

Executes a comparison between InputA and InputB (registers or constants). If InputA is greater than InputB sets the bits of a destination register *Dest* in function of the setting of the parameter *Mask*.

Arguments:

InputA	Constant or Internal Register relative to the first input to compare
InputB	Constant or Internal Register relative to the second input to compare
Dest	Internal Register wherein the bits will be set
Mask	Mask used to set the register Dest (the button on side opens the window for the setting)

A<B — Comparison of value (A<B) between two registers or between a register and a constant

Executes a comparison between InputA and InputB (registers or constants). If InputA is lower than InputB sets the bits of a destination register *Dest* in function of the setting of the parameter *Mask*.

Arguments:

InputA	Constant or Internal Register relative to the first input to compare
InputB	Constant or Internal Register relative to the second input to compare
Dest	Internal Register wherein the bits will be set
Mask	Mask used to set the register Dest (the button on side opens the window for the setting)

A>=B — Comparison of value (A>=B) between two registers or between a register and a constant

Executes a comparison between InputA and InputB (registers or constants). If InputA is greater or equal than InputB sets the bits of a destination register *Dest* in function of the setting of the parameter *Mask*.

Arguments:

InputA	Constant or Internal Register relative to the first input to compare
InputB	Constant or Internal Register relative to the second input to compare
Dest	Internal Register wherein the bits will be set
Mask	Mask used to set the register Dest (the button on side opens the window for the setting)

A<=B — Comparison of value (A<=B) between two registers or between a register and a constant

Executes a comparison between InputA and InputB (registers or constants). If InputA is lower or equal than InputB sets the bits of a destination register Dest in function of the setting of the parameter *Mask*.

Arguments:

InputA	Constant or Internal Register relative to the first input to compare
InputB	Constant or Internal Register relative to the second input to compare
Dest	Internal Register wherein the bits will be set
Mask	Mask used to set the register Dest (the button on side opens the window for the setting)

Internal Limit — Comparison of value (Internal Limit) between a registers and two values: Maximum and Minimum

Executes a comparison between Input and two values: Maximum Value (MAX) and Minimum Value (MIN).
If the value of Input is between maximum and minimum, sets the bits of a destination register *Dest* in function of the setting of the parameter *Mask*.

Arguments:

Input	Constant or Internal Register relative to the input to compare
MAX	Constant or Internal Register relative to High Limit level
MIN	Constant or Internal Register relative to Low Limit level
Dest	Internal Register wherein the bits will be set
Mask	Mask used to set the register Dest (the button on side opens the window for the setting)

External Limit — Comparison of value (External Limit) between a registers and two values: Maximum and Minimum

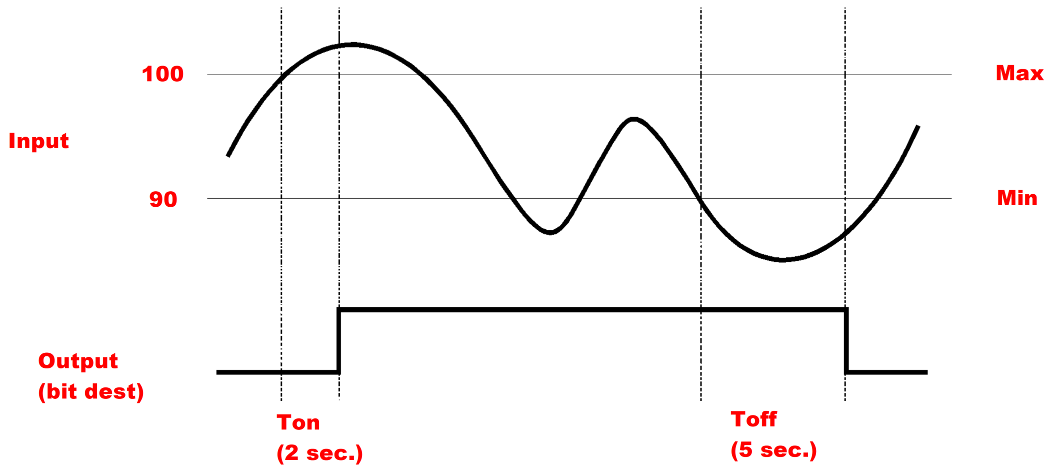
Executes a comparison between Input and two values: Maximum Value (MAX) and Minimum Value (MIN).
If the value of Input is external to maximum and minimum, sets the bits of a destination register *Dest* in function of the setting of the parameter *Mask*.

Arguments:

Input	Constant or Internal Register relative to the input to compare
MAX	Constant or Internal Register relative to High Limit level
MIN	Constant or Internal Register relative to Low Limit level
Dest	Internal Register wherein the bits will be set
Mask	Mask used to set the register Dest (the button on side opens the window for the setting)

Trip — Control of a Trip Alarm

Controls a Trip Alarm with setting of the Trip level, hysteresis and delay time for ON and OFF condition. If the input value is higher than the high trip level (Max) for a time longer than TimerOn, the bits selected in the output mask will be forced to 1. If the input value is lower than the low trip level (Min) for a time longer than TimerOff, the bits selected in the output mask will be forced to 0.
It is possible to use this Function Block to execute the operation “A>B”: set the trip levels with the same value and the delay times TimerOn and TimerOff = 0.
NOTE: the output status is updated at each execution of the Function after the end of the delay time: it is suggested to insert this Function Block in a zone of the Program continuously executed.
The graph **(Pict.4.1)** shows the working of a Trip alarm that goes on if the input signal is higher than 100°C for at least 2 seconds and goes off if the input signal is lower than 90°C for at least 5 seconds.



Pict. 4.1

Arguments:

Input	Internal Register containing the value to compare
Max	Constant or Internal Register relative to High Trip level
Min	Constant or Internal Register relative to Low Trip level
Dest	Internal Register wherein the bits will be set
Mask	Mask used to set the register Dest (the button on side opens the window for the setting)
NTimer	Number of Internal Timer to use (0÷15)
TimerOn	Delay time for Trip Alarm activation (ms)
TimerOff	Delay time for Trip Alarm de-activation (ms)

And (bit) — Executes the logical operation "AND" between two single bits.

Executes the logical operation "AND" on a single bit between a Register and a Constant or between two Registers. The value will be converted to the format selected for the Register of destination. The address of the source Register and the address of the Register of destination can be the same. After the execution of the logical operation only the bit set in the register of destination will be forced. In case of a 32 bit source Register or Constant (long) and a 16 bit Register of destination (integer), the most significant bits will be ignored.

Arguments:

SourceA	Constant or Internal Register relative to the first operator.
BitA	Bit selected for the first operator
SourceB	Constant or Internal Register relative to the second operator.
BitB	Bit selected for the second operator
Dest	Internal Register wherein the result is written.
BitDest	Bit selected for the register Dest

Or (bit) — Executes the logical operation "OR" between two single bits.

Executes the logical operation "OR" on a single bit between a Register and a constant or between two Registers. The value will be converted to the format selected for the Register of destination. The address of the source Register and the address of the Register of destination can be the same. After the execution of the logical operation only the bit set in the register of destination will be forced. In case of a 32 bit source Register or constant (long) and a 16 bit Register of destination (integer), the most significant bits will be ignored.

Arguments:

SourceA	Constant or Internal Register relative to the first operator.
BitA	Bit selected for the first operator
SourceB	Constant or Internal Register relative to the second operator.
BitB	Bit selected for the second operator
Dest	Internal Register wherein the result is written.
BitDest	Bit selected for the register Dest

XOr (bit) — Executes the logical operation "XOR" (Exclusive Or) between two single bits.

Executes the logical operation "XOR" on a single bit between a Register and a constant or between two Registers. The value will be converted to the format selected for the Register of destination. The address of the source Register and the address of the Register of destination can be the same. After the execution of the logical operation only the bit set in the register of destination will be forced. In case of a 32 bit source Register or constant (long) and a 16 bit Register of destination (integer), the most significant bits will be ignored.

Arguments:

SourceA	Constant or Internal Register relative to the first operator.
BitA	Bit selected for the first operator
SourceB	Constant or Internal Register relative to the second operator.
BitB	Bit selected for the second operator
Dest	Internal Register wherein the result is written.
BitDest	Bit selected for the register Dest

And (word) — Executes the logical operation "AND" between two values.

Executes the logical operation "AND" between a Register and a constant (mask) or between two Registers. The value will be converted to the format selected for the Register of destination. The address of the source Register and the address of the Register of destination can be the same.
After the execution of the logical operation, only the bits set as 1 in the mask will be forced; the bits set as 0 won't be modified. In case of a 32 bit source Register or constant (long) and a 16 bit Register of destination (integer), the most significant bits will be ignored. It is possible to use this function to force one or more bits of a Register as 0 (in the mask set as 0 the bits to force, set as 1 the other bits).

Arguments:

SourceA	Constant or Internal Register relative to the first operator.
SourceB	Constant or Internal Register relative to the second operator.
Dest	Internal Register wherein the result is written.
MaskDest	Mask used to set the register Dest (the button on side opens the window for the setting)

Or (word)

Executes the logical operation ”OR” between two values.

Executes the logical operation “OR” between a Register and a Constant (mask) or between two Registers. The value will be converted to the format selected for the destination Register. The address of the source Register and the address of the Register of destination can be the same.
After the execution of the logical operation only the bits set as 1 in the mask will be forced; the bits set as 0 won't be modified. In case of a 32 bit source Register or constant (long) and a 16 bit Register of destination (integer), the most significant bits will be ignored. It is possible to use this function to force one or more bits of a Register as 1 (in the mask set as 1 the bits to force, set as 0 the other bits).

Arguments:

SourceA	Constant or Internal Register relative to the first operator.
SourceB	Constant or Internal Register relative to the second operator.
Dest	Internal Register wherein the result is written.
MaskDest	Mask used to set the register Dest (the button on side opens the window for the setting)

Xor (word)

Executes the logical operation “XOR” (Exclusive Or) between two values.

Executes the logical operation “XOR” between a Register and a constant (mask) or between two Registers. The value will be converted to the format selected for the Register of destination. The address of the source Register and the address of the Register of destination can be the same.
After the execution of the logical operation only the bits set as 1 in the mask will be forced ; the bits set as 0 won't be modified. In case of a 32 bit source Register or constant (long) and a 16 bit Register of destination (integer), the most significant bits will be ignored. It is possible to use this function to invert (NOT) one or more bits of a Register (in the mask set as 1 the bits to invert, set as 0 the other bits).

Arguments:

SourceA	Constant or Internal Register relative to the first operator.
SourceB	Constant or Internal Register relative to the second operator.
Dest	Internal Register wherein the result is written.
MaskDest	Mask used to set the register Dest (the button on side opens the window for the setting)

NOT

Executes the inversion of one or more bits of a Register

Executes the inversion of one or more bit of a Register. After the execution of the logical operation will be forced only the bits set as 1 in the mask; the bits set as 0 won't be modified.

Arguments:

Source	Internal Register containing the value
Dest	Internal Register wherein the result is written.
MaskDest	Mask used to set the register Dest (the button on side opens the window for the setting)

Bit Set

Set to logic state High the bits of a register

Set to logic state High one or more bits of a register *Dest* in function of the mask *MaskDest.* It will be set only the bits set to 1 in the mask, the other will not be modified.

Arguments:

Dest	Internal Register wherein the bits will be set
MaskDest	Mask used to set the Register (the button on side opens the window for the setting)

Bit Reset — Set to logic state Low the bits of a register

Set to logic state Low one or more bits of a register *Dest* in function of the mask *MaskDest.* It will be reset only the bits set to 1 in the mask, the other will not be modified.

Arguments:

Dest	Internal Register wherein the bits will be set
MaskDest	Mask used to reset the Register (the button on side opens the window for the setting)

Pos — Intercepts the Rising Edge of one or more bits of a register

If the bits of the register *Source* have been selected in *MaskSource* and change state (from 0 to 1), then the corresponding bits of the register *Dest* will be forced to 1.
The register *Latch* is used as a temporary register and contains the previous value of the register *Source.*

Arguments:

Source	Internal Register containing the value.
MaskSource	Bit mask applied to the register (the button on side opens the window for the setting)
Latch	Temporary register (contains the previous value of the register *Source*)
Dest	Internal Register wherein the bits will be set

Neg — Intercepts the Falling Edge of one or more bits of a register

If the bits of the register *Source* have been selected in *MaskSource* and change state (from 1 to 0), then the corresponding bits of the register *Dest* will be forced to 1.
The register *Latch* is used as a temporary register and contains the previous value of the register *Source.*

Arguments:

Source	Internal Register containing the value.
MaskSource	Bit mask applied to the register (the button on side opens the window for the setting)
Latch	Temporary register (contains the previous value of the register *Source*)
Dest	Internal Register wherein the bits will be set

Shift Right — Shift to right the bits of a Register

Executes the shift of a Register to right: all of the bits are shifted of N positions to right. The most significant bits will be forced to 0.

Arguments:

Source	Constant or Internal Register containing the value.
Number	Number of shift to execute.
Dest	Internal Register wherein the result is written.

Shift Left — Shift to left the bits of a Register.

Executes the shift of a Register to left: all of the bits are shifted of N positions to left. The least significant bits will be forced to 0.

Arguments:

Source	Constant or Internal Register containing the value.
Number	Number of shift to execute.
Dest	Internal Register wherein the result is written.

Rotate Right	Rotates to right the bits of a Register

Executes the rotation of a Register to right: all of the bits are shifted of N positions to right. At each shift the most significant bit receives the value of the least significant bit.

Arguments:

Source	Constant or Internal Register containing the value.
Number	Number of shift to execute.
Dest	Internal Register wherein the result is written.

Rotate Left	Rotates to left the bits of a Register

Executes the rotation of a Register to left: all of the bits are shifted of N positions to left. At each shift the least significant bit receives the value of the most significant bit.

Arguments:

Source	Constant or Internal Register containing the value.
Number	Number of shift to execute.
Dest	Internal Register wherein the result is written.

Read Register	Reads registers from a generic Modbus Slave device with function Modbus 03

Reads the value of N registers from a generic Modbus slave device and writes the read values in the selected Internal Registers. To read it uses the function Modbus 03. In case of missing response or wrong response by the slave device, the Registers of destination are not updated and the value of the System Register “COM Errors” is increased.

Arguments:

Address	Modbus address of the slave device (1÷247)
Register	Address of the first Register to read (the mapping Registers starts from 0)
Number	Numbers of Registers to read (1÷16)
Dest	Address of the first Internal Register wherein the read values are written to.
Delay	Delay time between the reception of the response and the execution of the next instruction

Read Input	Reads registers from a generic Modbus Slave device with function Modbus 04

Reads the value of N registers from a generic Modbus slave device and writes the read values in the selected Internal Registers. To read it uses the function Modbus 04. In case of missing response or wrong response by the slave device, the Registers of destination are not updated and the value of the System Register “COM Errors” is increased.

Arguments:

Address	Modbus address of the slave device (1÷247)
Register	Address of the first Register to read (the mapping Registers starts from 0)
Number	Numbers of Registers to read (1÷16)
Dest	Address of the first Internal Register wherein the read values are written to.
Delay	Delay time between the reception of the response and the execution of the next instruction

Write Single	Writes single register of a generic Modbus Slave device with function Modbus 06

Writes the value of an Internal Register in a register of a generic Modbus Slave device. To write it uses the function Modbus 06. In case of missing response or wrong response by the slave device, the Registers of destination are not updated and the value of the System Register “COM Errors” is increased.

Arguments:

Address	Modbus address of the slave device (1÷247)
Register	Address of the Register to write (the mapping Registers starts from 0)
Source	Address of the Internal Register where the value to write is contained
Delay	Delay time between the reception of the response and the execution of the next instruction

Write Multiple — Writes multiple registers of a generic Modbus Slave device with function Modbus 16

Writes the values of N Internal Registers in the registers of a generic Modbus Slave device. To write it uses the function Modbus 16. In case of missing response or wrong response by the slave device, the Registers of destination are not updated and the value of the System Register “COM Errors” is increased.

Arguments:

Address	Modbus address of the slave device (1÷247)
Register	Address of the Register to write (the mapping Registers starts from 0)
Number	Number of the registers to write (1÷16)
Source	Address of the first Internal Register from which the values to write are contained
Delay	Delay time between the reception of the response and the execution of the next instruction

Read Device — Read registers from a Modbus Slave DAT3000/DAT10000 series device

Reads the I/O values from a Modbus slave DAT3000/DAT10000 series device and writes the values in the Internal Registers. The function will generate the proper Modbus command and will process the response.
In case of missing response or wrong response by the slave device, the Registers of destination are not updated and the value of the System Register “COM Errors” is increased.
Refer to the technical documentation of the device for the complete description of the I/O Registers.

Arguments:

Delay	Delay time between the reception of the response and the execution of the next instruction
Device	Type of device to read
Remote Address	Modbus address of the slave device (1÷247)
Resource	Type of resource to read (analog inputs, digital inputs, etc...)
From	First resource to read
To	Last resource to read
Dest	Address of the first Internal Register wherein the read values are written to.

Write Device — Write registers of a Modbus Slave DAT3000/DAT10000 series device

Writes the values of *N* Internal Registers in the I/O Register of a Modbus slave DAT3000/DAT10000 series device. The function will generate the proper Modbus command and will process the response. In case of missing response or wrong response by the slave device, the Registers of destination are not updated and the value of the System Register “COM Errors” is increased.
Refer to the technical documentation of the device for the complete description of the I/O Registers.

Arguments:

Delay	Delay time between the reception of the response and the execution of the next instruction
Device	Type of device to write
Remote Address	Modbus address of the slave device (1÷247)
Resource	Type of resource to write (analog outputs, digital outputs, etc...)
From	First resource to write
To	Last resource to write
Source	Address of the first Internal Register from which the values to write are contained

Timer — Activation of an Internal Timer

Sets an Internal Timer (0÷15) and starts to count. During the count, the bit relative to the Timer selected in the System Register “Timers Enable”, will be forced to 0. At the end of the count, the bit will be forced to 1.
It is possible to check the status of the bit to determine the end of the time set. The duration of the timer is set by a constant *mSec* (until 65535) or if the constant is 0 by a register *Reg*.

Arguments:

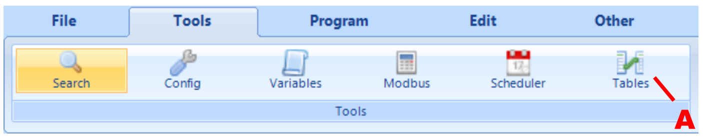
Timer	Number of the Internal Timer to enable (0÷15)
mSec	Timer Pre-set (milliseconds)
Reg	Register used to preset the Timer if mSec is 0

Call Page — Call and refresh the page on the Display

Call the refresh of the page visualized on the display. Once the function is executed the program continues with the next function.

5.1 – INSERTION OF LINEARIZATION TABLES

To insert or modify the Tables click the proper button in the menu bar **(Pict.5.1-A)** or the button *Table* of the function Linearization. Then it will be opened the window "*Tables*" **(Pict.5.2)**



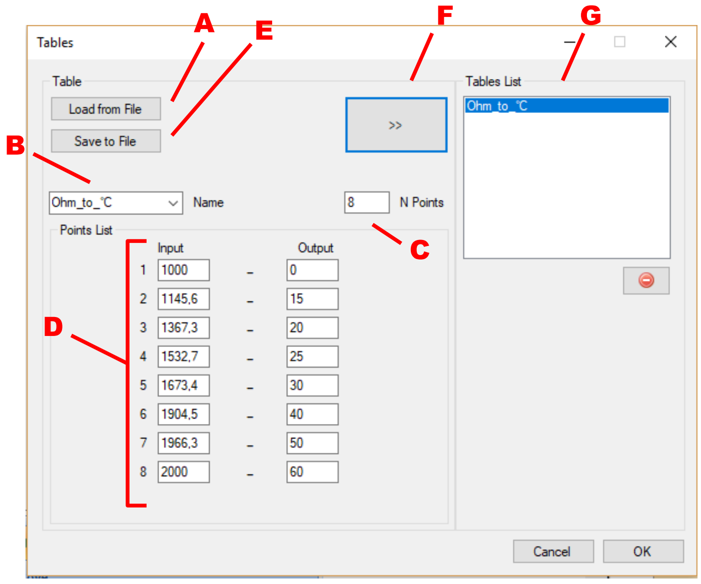
Pict. 5.1

To upload the points of a table from a file, click on the "*Load from File*" button **(Pict.5.2-A)**; the parameters relative to the table like the name, the number of points and the input and output values for each point will be loaded.
It is possible to modify the Name of the table **(Pict.5.2-B)** (the name must be unique, because it is used in the program to call that table). Set the number of points to insert in the table **(Pict.5.2-C)**.
It is possible to insert until 32 points. Each point is defined by the input and output values. The input values must be inserted in increasing order, while the output values can be inserted both in increasing and decreasing order. The example **(Pict.5.2-D)** shows how to create a 8 points table to linearize a RTD temperature sensor and to obtain the conversion Ohm/°C **(Pict.5.3)**.

When the insertion of points is complete, it is possible to save the table in a file, clicking on the "*Save to File*" button **(Pict.5.2-E);** by this command the name of the table, the number of points and the input and output values per each points will be saved.

To insert the table inside the Program, click on the ">>" button **(Pict.5.2-F)**. The "Table List" **(Pict.5.2-G)** will be updated with the name of the table just inserted and shows the tables available for the Program.

In the Program, when the block function Linearization is inserted, it is possible to select one of the tables available.
When the Function Block is recalled by the Program, the Controller executes a control between the value contained into the Internal Register and the points of the selected table and calculates by interpolation the output value.
In the example for an input value of 1789 Ohm, will be calculated an output of 35 °C (without linearization function the output should be 47,3 °C).



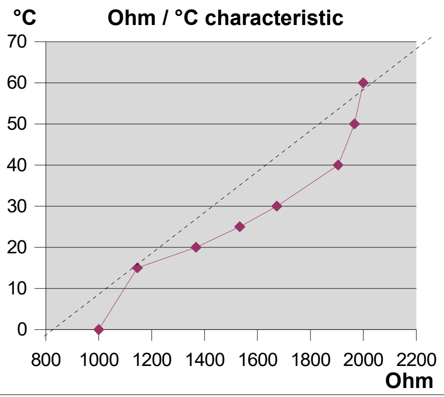
Name: Ohm_to_°C
N Points: 8

	Input		Output
1	1000	–	0
2	1145,6	–	15
3	1367,3	–	20
4	1532,7	–	25
5	1673,4	–	30
6	1904,5	–	40
7	1966,3	–	50
8	2000	–	60

Pict. 5.2

Example of table provided from the manufacturer of the sensor :

°C	Ohm
0	1000
15	1145,6
20	1367,3
25	1532,7
30	1673,4
40	1904,5
50	1966,3
60	2000



Pict. 5.3

5.2 – IP ADDRESSES TABLE "Server / Slave" TCP DEVICES

The DAT9000 series devices can operate as master not only on the RS485 serial port but also on the Ethernet interface using the Modbus TCP / IP protocol.

To due this is therefore necessary to enter the IP addresses of the *"Server"* devices to which Modbus TCP read and write commands are to be sent. The *"Server"* devices that can be inserted in the table are at most 8 **(Pict.5.4-F)**.

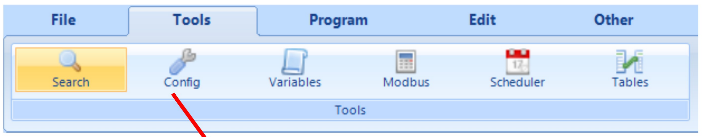
From the *Tools → Configuration* menu **(Pict.5.3-A)** click on the IP Table window to access it **(Pict.5.4)**

The IP table allows to associate to each "*IP Address*" **(Pict.5.4-B)** of the Server device, an "*RTU Node ID*" **(Pict.5.4-A)** which will be the equivalent of the modbus address / node used in the functions standard reading and writing. In fact, in the construction of the project, the same function blocks of Read Holding, Read Input, etc. will be used, independently of the modbus protocol.
Each server device inserted in the table has a physical address identifying "*Device Address*" **(Pict.5.4-C)** which, in most cases, is not significant (as DAT8000 series). This address corresponds to the real modbus address / node of the slave to be interrogated and serves the controller to build the frame of the TCP / IP modbus protocol. In the table it is necessary to insert the communication port which in the case of modbus TCP is 502 **(Pict.5.4-D)**

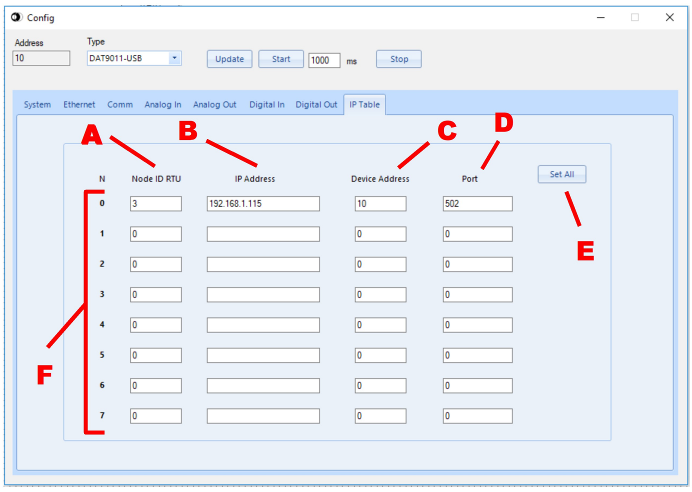
Once you have finished inserting / editing the IP table, click on the Set All button to save **(Pict.5.4-E)**.

For the application of the application to be loaded in the controller, all the function blocks provided in the software will be used **(Pict.5.5)**. Whereas, in the read and write function blocks, the one inserted in the IP table identified as the *RTU Node ID* **(Pict.5.5-A)** must be used as the modbus address (Address).

It is also possible to insert the devices in the IP table also via web pages on the "IP Table Configuration" page **(Pict.5.6)**. In this page it is possible to test the connected devices when the controller is in STOP, by clicking on *"Devices Test"* **(Pict.5.6-A)**.



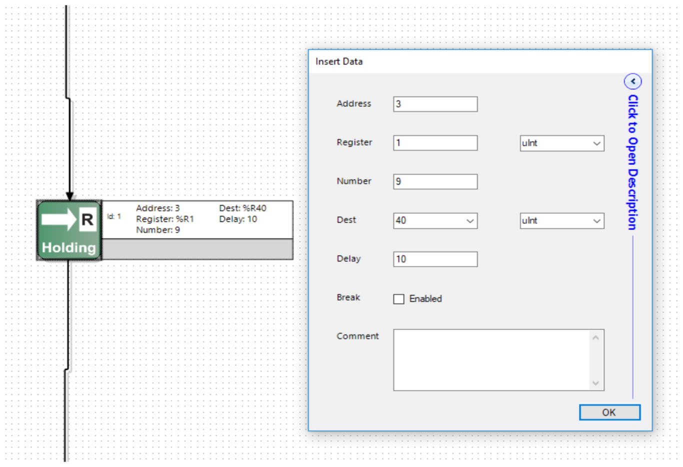
Pict. 5.3



Address: 10
Type: DAT9011-USB
1000 ms

N	Node ID RTU	IP Address	Device Address	Port
0	3	192.168.1.115	10	502
1	0		0	0
2	0		0	0
3	0		0	0
4	0		0	0
5	0		0	0
6	0		0	0
7	0		0	0

Pict. 5.4



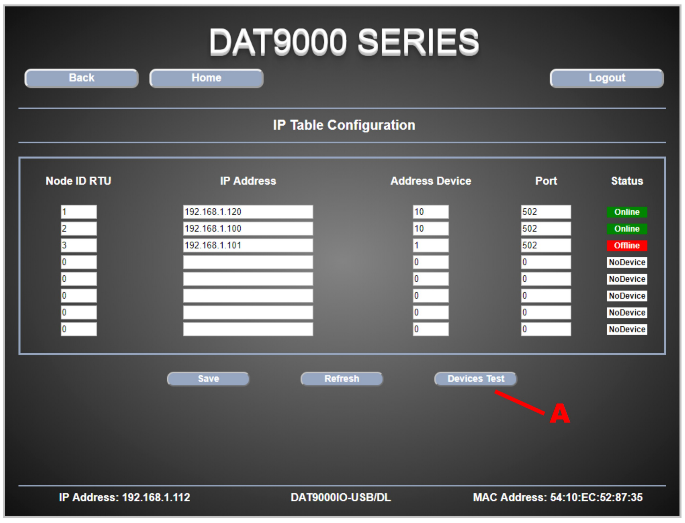
Id: 1
Address: 3
Register: %R1
Number: 9
Dest: %R40
Delay: 10

Insert Data
Address: 3
Register: 1 uInt
Number: 9
Dest: 40 uInt
Delay: 10
Break: ☐ Enabled
Comment:

Pict. 5.5

NOTES:
1) the DAT9000 series can interrogate both devices connected to the Ethernet interface in modbus TCP and slave devices connected to the RS485 Master serial port as long as the modbus addresses of the devices on the RS485 are different from those associated with the IP devices of the server devices in the table (RTU Node ID).

2) Before testing the devices via a web interface, make sure that the controller is in STOP mode.



DAT9000 SERIES

IP Table Configuration

Node ID RTU	IP Address	Address Device	Port	Status
1	192.168.1.120	10	502	Online
2	192.168.1.100	10	502	Online
3	192.168.1.101	1	502	Offline
0		0	0	NoDevice
0		0	0	NoDevice
0		0	0	NoDevice
0		0	0	NoDevice
0		0	0	NoDevice

IP Address: 192.168.1.112 DAT9000IO-USB/DL MAC Address: 54:10:EC:52:87:35

Pict. 5.6

6.1 – SEARCHING OF THE DEVICES CONNECTED

Connect the Controller to the Ethernet network and power-on it (refer to the data-sheet).
Clicking the button “*Search*” in the menu bar, the window “*Search*” **(Pict.6.1)** will be opened.
In the first list **(Pict.6.1-A)** there are the available networks of the PC in use.
Click the button “*Get Local IP*” **(Fig.6.1-B)** to visualize the local IP of the Personal Computer in the Net in use.
Once identified the correct Net, select it and click the button “Bind” **(Pict.6.1-C)** to assign the search function.
Then click the button “*Search*” **(Pict.6.2-A)** to search devices in the Net. If the Net is correct, and so there are one or more devices, the devices found will appear in the list. If the Net is not correct and the user wants to bind another Net, click the button “*Back*”**(Pict.6.2-B)**.
For each device found it is possible to read the relative IP address, the MAC number, the Mask, the Modbus Address and the Init status.
Click the right button of the mouse to open a drop down menu with some additional functions **(Pict.6.2-C)**.
Click *Set as Controller* to connect the device.
Click *Open Webpage* to open the web page of the device on the browser.
Click *Set IP address* to set a new IP address for the selected device.

6.2 – MANUAL CONNECTION TO THE DEVICE

In the window “*Search*” there is a section for manual connection through Ethernet or COM RS485/uUSB ports. Here it is possible to set communication parameters.

For the Ethernet **(Pict.6.3)**:
IP Address: IP address of the device
Port: Modbus/TCP socket reserved port (*502* for direct connection)
Timeout (mSec): Receiving Timeout for TCP commands
Modbus ID :Modbus node address (1 ÷ 247)

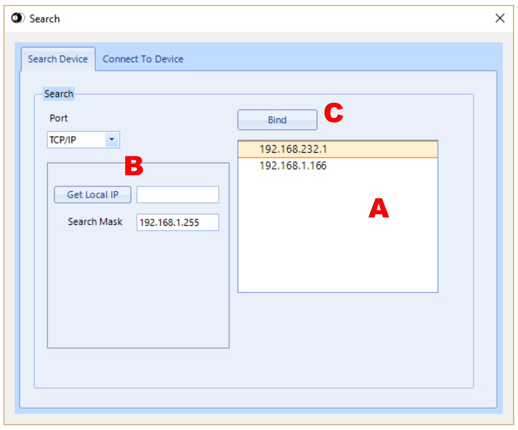
For COM port **(Pict.6.4)**:
Port Name: Name of the available COM ports
Baud Rate: Rate of serial transmission
Data Bits: Number of bits of transmission
Parity: Parity of the transmission
Stop Bits: Stop bit of the transmission
Handshake: Handshake Mode
Modbus ID : Modbus node address (1 ÷ 247)

To confirm the parameters, click on the “*OK*” button.

Click the button “Connect” **(Pict.6.5)** to connect automatically to the last device.
If the connection ends well, the message “*Connected*” will be visualized in the Status bar and in the Log panel; in case of error, refer to the section “*Troubleshooting*” to solve the problem.

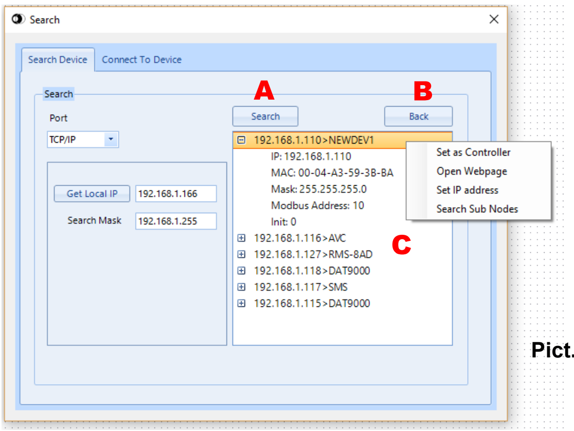
From this moment all of the reading, writing, programming and debugging operations will be sent only to the selected Controller.
If the user has to change the Controller, it is necessary to disconnect the Controller in use and connect to another Controller following one of the modalities described above.

<u>NOTE</u>: <u>the controller goes automatically offline after the socket timeout set into it has expired. In this case connect again to the device</u>



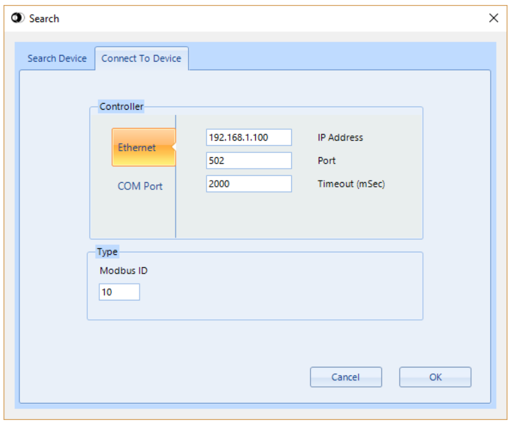
Port: TCP/IP
Search Mask: 192.168.1.255
192.168.232.1
192.168.1.166

Pict. 6.1



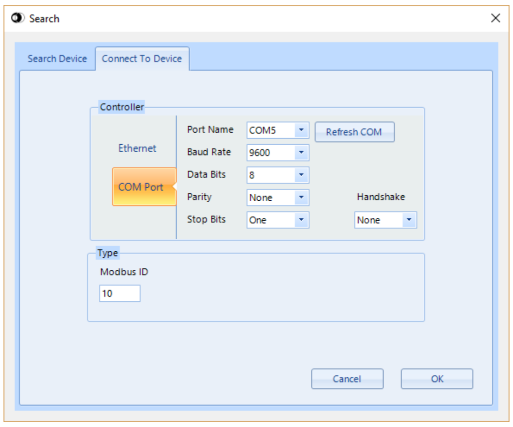
Port: TCP/IP
Get Local IP: 192.168.1.166
Search Mask: 192.168.1.255
192.168.1.110>NEWDEV1
IP: 192.168.1.110
MAC: 00-04-A3-59-3B-BA
Mask: 255.255.255.0
Modbus Address: 10
Init: 0
192.168.1.116>AVC
192.168.1.127>RMS-8AD
192.168.1.118>DAT9000
192.168.1.117>SMS
192.168.1.115>DAT9000

Pict. 6.2



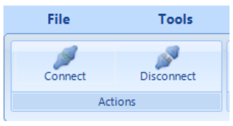
IP Address: 192.168.1.100
Port: 502
Timeout (mSec): 2000
Modbus ID: 10

Pict. 6.3



Port Name: COM5
Baud Rate: 9600
Data Bits: 8
Parity: None
Stop Bits: One
Handshake: None
Modbus ID: 10

Pict. 6.4

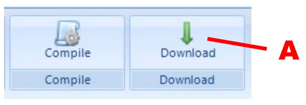


Pict. 6.5

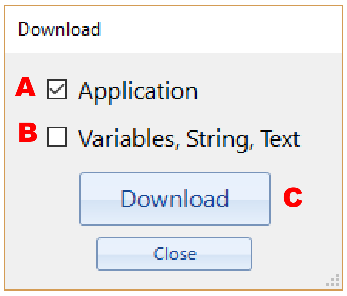
6.3 – PROGRAM DOWNLOAD

When the Program is complete and if the compiling ends correctly it is possible to download it in the internal memory of Controller.
Clicking the button “*Download*” in the menu bar **(Pict.6.6-A)**, the “*Download*” window will be opened **(Pict.6.7)**. The Download operations are allowed only in “Debug” mode (refer to the section “Debug Mode”).
It is possible to set two options:
“*Application*” **(Pict.6.7-A)** – Download the program in the internal memory of the Controller.
“*Variables, String, Text*” **(Pict.6.7-B)** – Download the settings of the variables, strings and texts.

Click the button “*Download*” **(Pict.6.7-C)** to download the selected options in the Controller.



Pict. 6.6

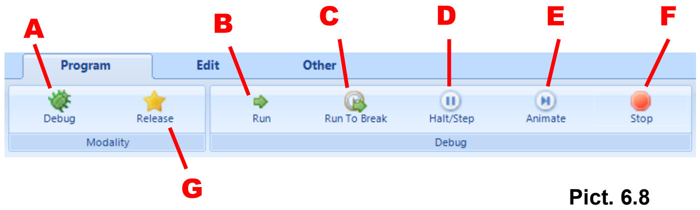


Pict. 6.7

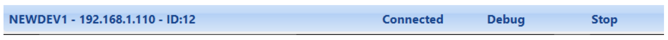
6.4 - DEBUG MODE

During the development of the Program, if the Controller is connected, click the button “*Debug*” **(Pict.6.8-A)** to activate the “Debug” mode.
In the Status bar the message “*Debug*” **(Pict.6.9)** will be visualized and in the menu bar the buttons to execute the following debug operations will be enabled:
“*Run*” **(Pict.6.8-B)** – Executes the Program continuously.
“*Run To Break*” **(Pict.6.8-C)** – Executes the Program up to the Break point
“*Halt/Step*” **(Pict.6.8-D)** – Interrupts the execution of the Program (“Run” condition) / executes the Program step by step (“Stop” condition)
“*Animate*” **(Pict.6.8-E)** – Simulates the evolution of the Program flow executing it step by step.
“*Stop*” **(Pict.6.8-F)** – Blocks the Program and reset it to the first Function Block.

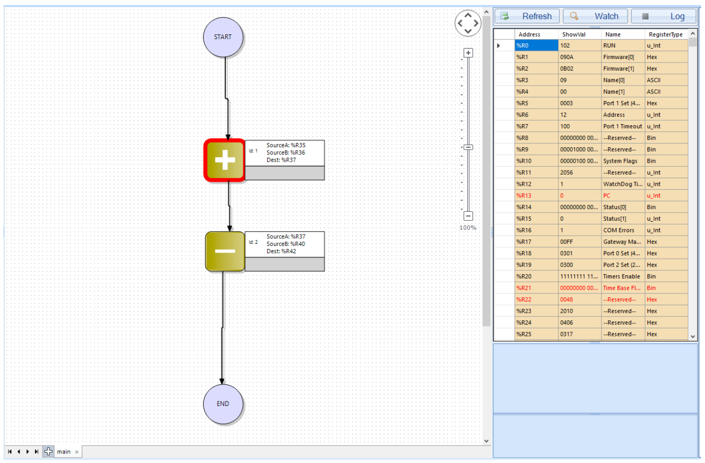
Then it is possible to follow the flow of the program, reading the state of the Controller and input registers from the register table.
When the Program is in Halt, Step or Animate the position of the function block to be executed is identified with the red border. In the same time in the register table the value of the Program Counter (“PC”) is updated **(Pict.6.10)**.



Pict. 6.8



Pict. 6.9

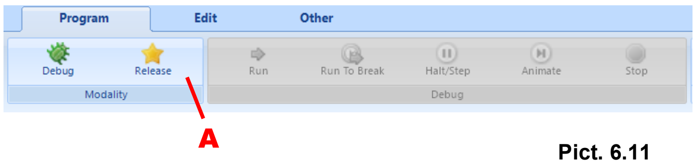


Address	ShowVal	Name	RegisterType
%R0	102	RUN	u_Int
%R1	090A	Firmware[0]	Hex
%R2	0B02	Firmware[1]	Hex
%R3	09	Name[0]	ASCII
%R4	00	Name[1]	ASCII
%R5	0003	Port 1 Set (4...	Hex
%R6	12	Address	u_Int
%R7	100	Port 1 Timeout	u_Int
%R8	00000000 00...	--Reserved--	Bin
%R9	00001000 00...	--Reserved--	Bin
%R10	00000100 00...	System Flags	Bin
%R11	2056	--Reserved--	u_Int
%R12	1	WatchDog Ti...	u_Int
%R13	0	PC	u_Int
%R14	00000000 00...	Status[0]	Bin
%R15	0	Status[1]	u_Int
%R16	1	COM Errors	u_Int
%R17	00FF	Gateway Ma...	Hex
%R18	0301	Port 0 Set (4...	Hex
%R19	0300	Port 2 Set (2...	Hex
%R20	11111111 11...	Timers Enable	Bin
%R21	00000000 00...	Time Base Fl...	Bin
%R22	0048	--Reserved--	Hex
%R23	2010	--Reserved--	Hex
%R24	0406	--Reserved--	Hex
%R25	0317	--Reserved--	Hex

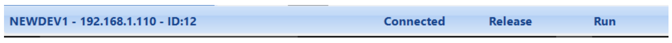
Pict. 6.10

6.5 - RELEASE MODE

After the phases of development and Debug it is possible to proceed with the “*Release*” mode, clicking the button “*Release*” **(Pict.6.11-A)**.
In the Status bar the message “*Release*” **(Pict.6.12)** will be visualized and in the menu bar the buttons to execute the debug and download operations will be disabled.
In the “*Release*” Modality, at the power-on, the Controller will be automatically set in “Run” condition, loading in the RAM memory the Program saved in the Internal Flash memory.
In this modality it is possible to read and write the Internal Registers.



Pict. 6.11



Pict. 6.12

6.6 - INIT MODE

All DAT9000 series devices are equipped with the INIT mode. This is a mode to access the device with the default parameters regardless of the configuration stored in EEPROM.
Following is the procedure for using the INIT mode which is also indicated on the User Guide of each device in the ***"PROCEDURES"*** section:

Over Ethernet:
- IP Address: XXX.XXX.XXX.XXX provided by the DHCP if is enabled or 192.168.1.174 if DHCP is disabled (verify that the IP is not already used and that the PC belongs to the same subnet)
- Modbus address: 10
  By these parameters it is possible to access the device in INIT mode to configure it or see the stored configuration.

To enter INIT, follow the procedure below:
1) Turn off the device;
2) Connect the INIT terminal to the -V terminal as shown in the technical datasheet of the device.
3) Turn on the device;
4) Connect to the device using the default parameters shown above.

When the user finishes working in INIT mode:
1) Turn off the device;
2) Remove the INIT connection;
3) Turn on the device and connect with the parameters known or configured in INIT mode.

Over RS485 slave or uUSB:
1) Switch off the device.
2) Connect only the device to be programmed to the RS485 slave or uUSB network.
3) Connect the INIT terminal to terminal V-.
4) Turn on the device.
5) Set the communication port with the following values
   Mode = Modbus RTU
   Baud-rate = 9600 bps
   Parity = None
   Bit number = 8
   Stop bit = 1
   Modbus address = 10

6) Read or program the desired settings in the registers using the Dev9k software.
7) Switch off the device.
8) Disconnect the INIT terminal from terminal V-.
9) Set the communication port with the programmed baud-rate
10) The module answers with the programmed address

ATTENTION:

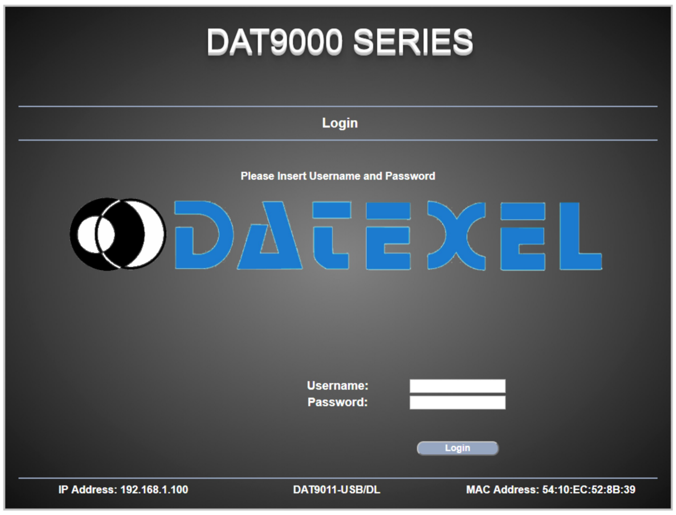
In INIT mode, the device's Debug / Stop mode is also forced. Therefore any program stored in EEPROM will not be executed.

6.7 – WEB INTERFACE

Through a Web Browser it is possible to access the Controller's Web Server to view the Web pages containing the configuration data and to download the Log files.
To access the integrated web server, open a browser on your PC and type the IP address of the device in the address bar of the browser.
- Factory IP Address: 192.168.1.100
<u>Warning: make sure that the PC is in the same subnet as the device in use (see user guide of the device).</u>
The factory / default access credentials that are requested on the "Login" page **(Pict. 6.13)** are:

- **Username:** Fact_user
- **Password:** Fact_pwd

After the first login, it is possible to change the credentials in the "Username and Password" section and the network parameters in the "Network Settings" page.



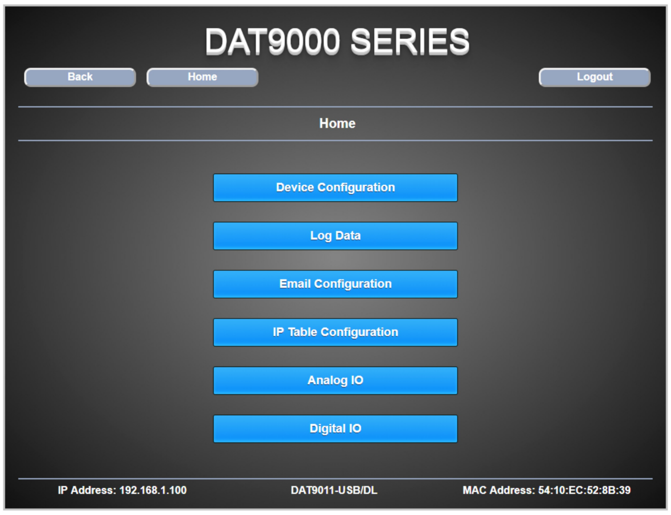
Please Insert Username and Password
Username:
Password:
IP Address: 192.168.1.100 DAT9011-USB/DL MAC Address: 54:10:EC:52:8B:39

Pict. 6.13

From the **Home** page **(Pict. 6.14)** it is possible to access the following pages:

Device Configuration: allows access to the device configuration menu.
From this menu it is possible to access the pages:

- <u>System Configuration:</u> configuration of system parameters (Netbios Name, Watchdog, modbus address, ...);
- <u>Username and Passoword:</u> modify access credentials;
- <u>Date and Time:</u> time and date setting;
- <u>Network Settings:</u> network parameters settings (IP address, Subnet mask, Gateway, DNS);
- <u>DDNS Settings:</u> setting connection parameters to the Dynamic DNS server, for remote connection with a dynamic IP (requires to subscribe to the service www.dyndns.com);
- <u>Modbus Settings:</u> setting of the communication parameters of the Master and Slave / uUSB serial ports;
- <u>Software Update:</u> allows the user to upload a new release of firmware or web pages;



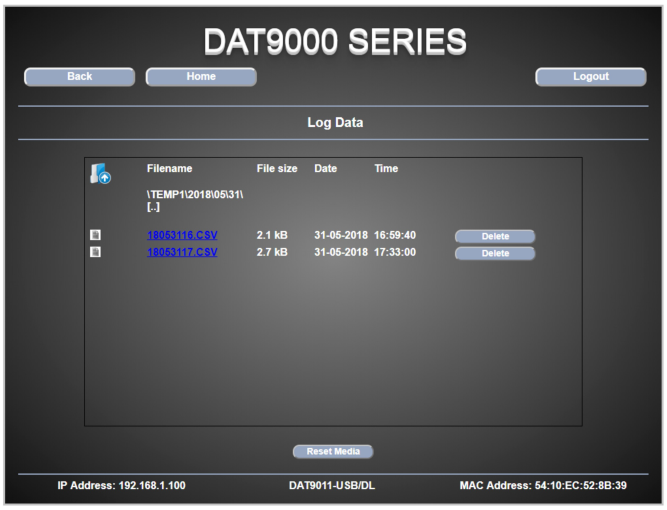
IP Address: 192.168.1.100 DAT9011-USB/DL MAC Address: 54:10:EC:52:8B:39

Pict. 6.14

Log Data: through this page **(Pict. 6.15)** it is possible to read the contents of the USB stick or SD card (depending on the model) and download the .CSV files with the DataLogger recordings.

Email Configuration: it allows to modify the parameters of the sender and recipients' email, the message body and the parameters related to the outgoing mail server.

<u>Warning</u>*:*
1) it is possible to insert only body of the message
*2) the outgoing mail server (SMTP) must allow the use of port **25** (not encrypted)*



\TEMP1\2018\05\31\
[..]

Filename	File size	Date	Time
18053116.CSV	2.1 kB	31-05-2018	16:59:40
18053117.CSV	2.7 kB	31-05-2018	17:33:00

IP Address: 192.168.1.100 DAT9011-USB/DL MAC Address: 54:10:EC:52:8B:39

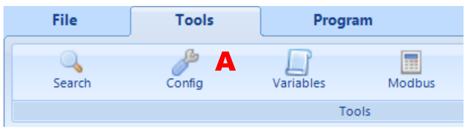
Pict. 6.15

IP Table Configuration: in this page it is possible to enter the IP addresses of the "Server" devices to be queried by the controller with Modbus TCP / IP protocol. A maximum of 8 "Server" devices can be entered in the table.

Analog IO: this page is only available for the DAT9011-USB and DAT9011-DL models as it allows the user to configure the analog inputs, display the measurements, write to the analog outputs and set the values of Safe and PowerUp.

Digital IO: this page is available for all models of the DAT9000-USB and DAT9000-DL models, as it allows to display the status of the digital inputs and outputs as well as the count of the pulse counters. It is also possible to change the status of the digital outputs, the Safe and PowerUp conditions.

6.8 - CONFIGURATION WINDOW



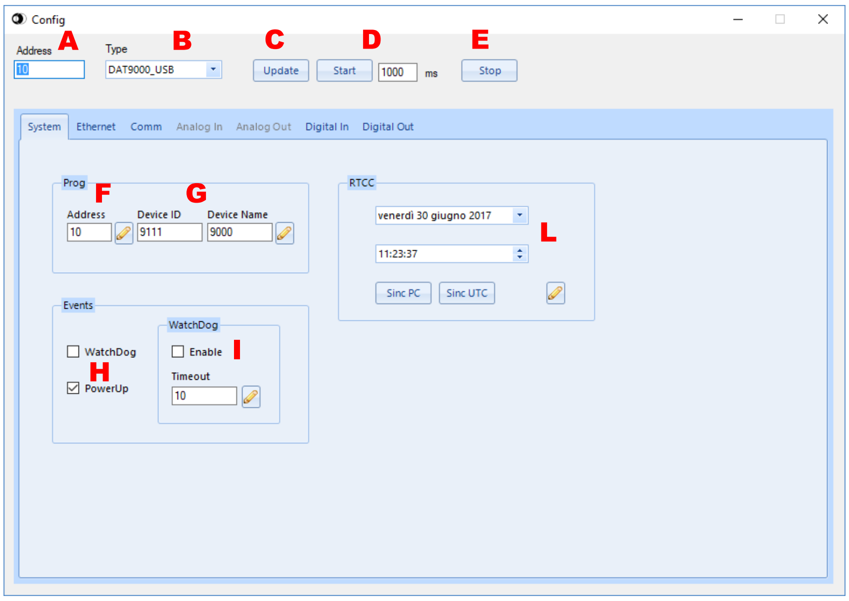
Pict. 6.16

Click in the menu bar the button “*Config*” to show the device configuration window **(Pict.6.16-A)**

In the configuration window **(Pict.6.17)** it is possible to visualize most of the editable parameters of a device.

The fields on the top of the window are:
- the Modbus address of the device to connect to **(Pict.6.17-A)**
- the model of the device connected **(Pict.6.17-B)**
- the button “*Update*” used to update the window and so the parameters available for the selected device **(Pict.6.17-C)**
- the button “*Start*” used to update cyclically the window and beside it there is the frequency of update **(Pict.6.17-D)**
- the button “*Stop*” to stop the cycling update **(Pict.6.17-E)**

The window is divided in tabs and internal sections visible or not in function of the firmware of the selected device. To modify a parameter usually there is a button with pencil near it.

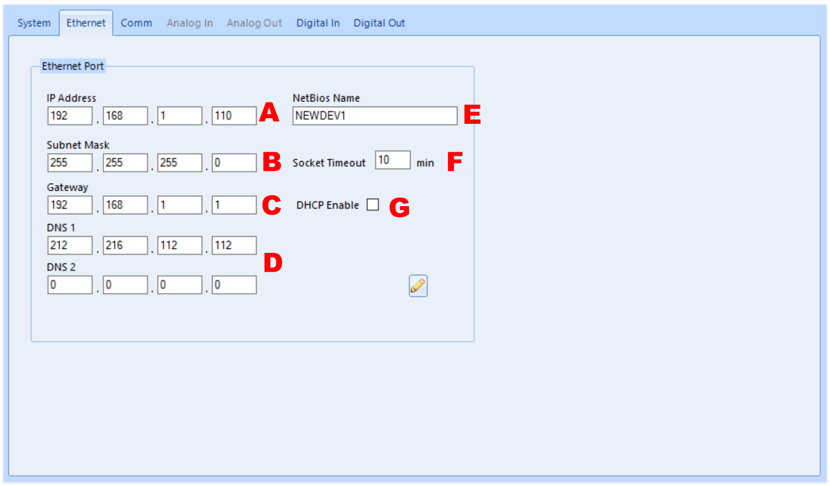


Address: 10
Type: DAT9000_USB
1000 ms
Prog — Address: 10, Device ID: 9111, Device Name: 9000
RTCC: venerdì 30 giugno 2017, 11:23:37
Events: ☐ WatchDog, ☑ PowerUp
WatchDog: ☐ Enable, Timeout: 10

Pict. 6.17

System

The System Tab contains the system data of the device:
- the Modbus address of the device **(Pict.6.17-F)**
- the ID and the name of the device **(Pict.6.17-G)**
- the WatchDog state and PowerUp state **(Pict.6.17-H)**
- the enable option of WatchDog with relative timeout time **(Pict.6.17-I)**
- the date and time measured by the device's RTC with sync button **(Pict.6.17-L)**
- in the display there is a section to set brightness, contrast and negative/positive option of display.

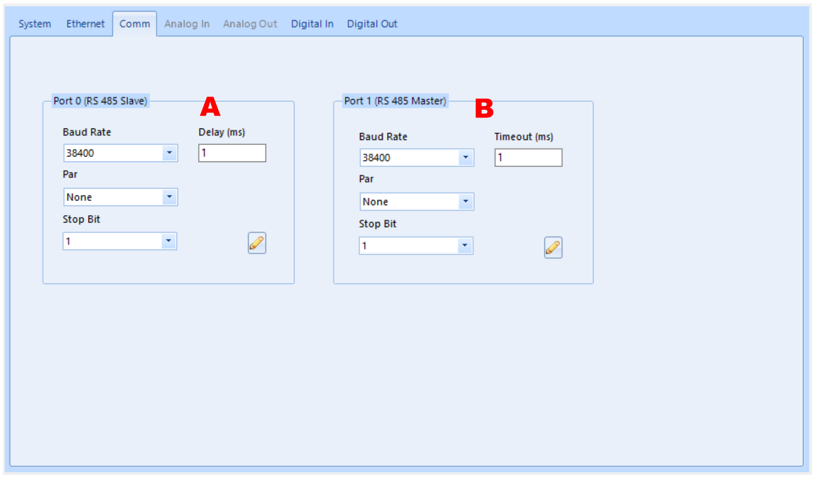


IP Address: 192 . 168 . 1 . 110
NetBios Name: NEWDEV1
Subnet Mask: 255 . 255 . 255 . 0
Socket Timeout: 10 min
Gateway: 192 . 168 . 1 . 1
DHCP Enable ☐
DNS 1: 212 . 216 . 112 . 112
DNS 2: 0 . 0 . 0 . 0

Pict. 6.18

Ethernet

The Ethernet Tab contains the Ethernet configuration data of the device:
- IP address of the device **(Pict.6.18-A)**
- Subnet Mask of the device **(Pict.6.18-B)**
- Gateway of the device **(Pict.6.18-C)**
- DNS 1 and DNS 2 of the device **(Pict.6.18-D)**
- NetBios Name of the device **(Pict.6.18-E)**
- Socket Timeout of the device expressed as minutes **(Pict.6.18-F)**
- Enable of DHCP for the device **(Pict.6.18-G)**



Port 0 (RS 485 Slave)

Baud Rate	38400
Delay (ms)	1
Par	None
Stop Bit	1

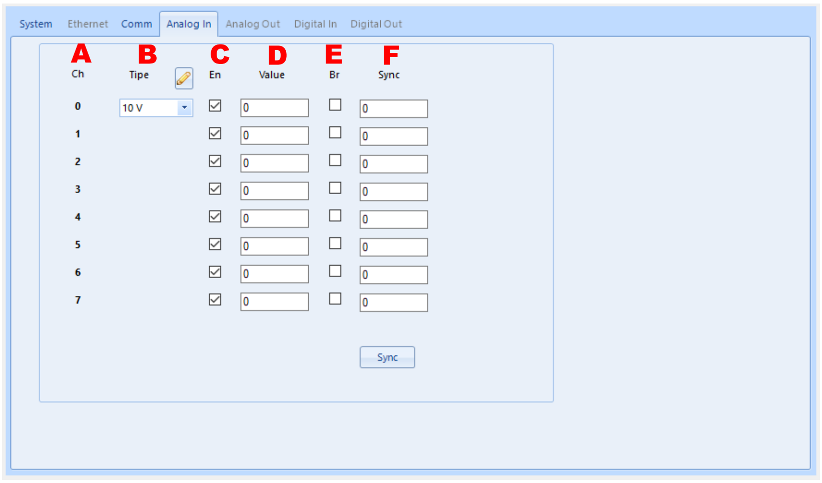
Port 1 (RS 485 Master)

Baud Rate	38400
Timeout (ms)	1
Par	None
Stop Bit	1

Pict. 6.19

Comm

The Comm Tab contains the configuration data of the Com Ports available on the device.
- Port 0 (RS485 Slave) with Baud Rate, Par, Stop Bit and Delay **(Pict.6.19-A)**
- Port 1 (RS485 Master) with Baud Rate, Par, Stop Bit and Timeout **(Pict.6.19-B)**

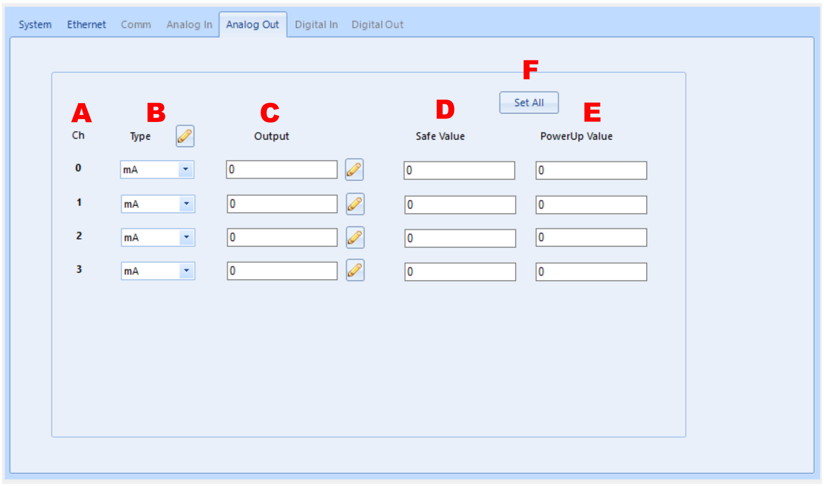


Ch	Tipe	En	Value	Br	Sync
0	10 V	✓	0	☐	0
1		✓	0	☐	0
2		✓	0	☐	0
3		✓	0	☐	0
4		✓	0	☐	0
5		✓	0	☐	0
6		✓	0	☐	0
7		✓	0	☐	0

Pict. 6.20

Analog In

The Analog In Tab contains the configuration data of the analog input of the device. So in function of the selected device there is a different number of analog inputs with different settings.
In general there are:
- the number of the channel **(Pict.6.20-A)**
- the type of input for the channel and the relative button with pencil to set the input selected **(Pict.6.20-B)**
- the enable of the channel **(Pict.6.20-C)**
- the measured value of the input channel **(Pict.6.20-D)**
- the break status of the channel **(Pict.6.20-E)**
- the sync value of the channel with the button to save the values **(Pict.6.20-F)**

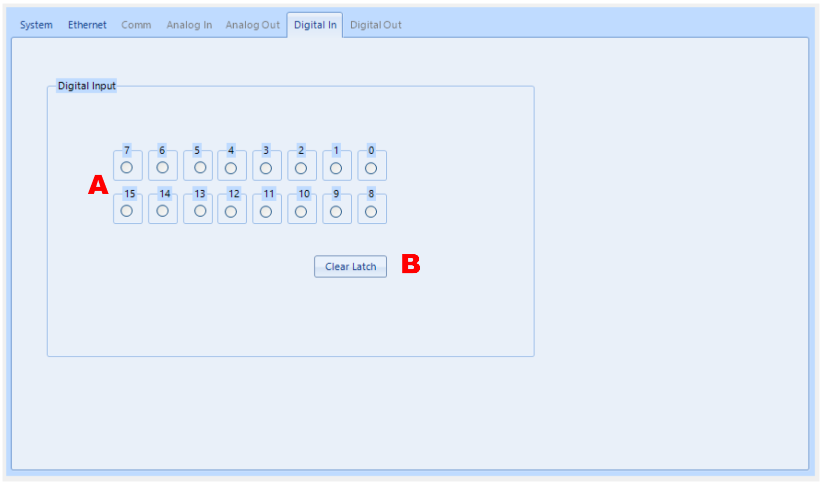


Ch	Type	Output	Safe Value	PowerUp Value
0	mA	0	0	0
1	mA	0	0	0
2	mA	0	0	0
3	mA	0	0	0

Pict. 6.21

Analog Out

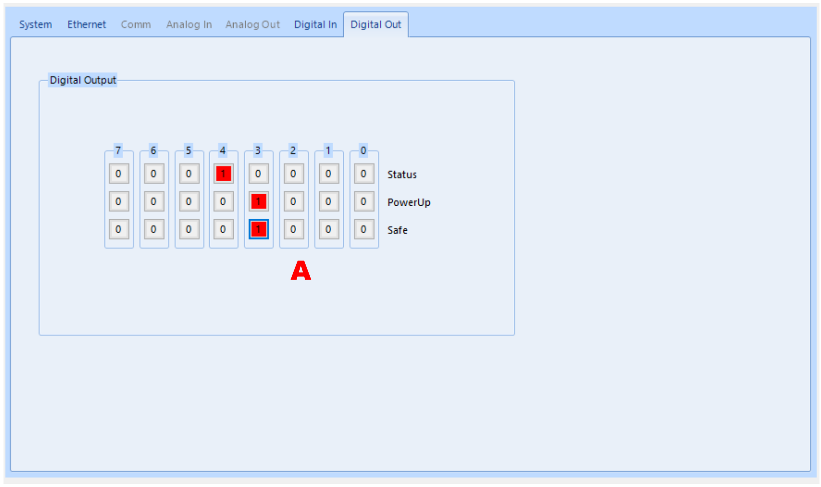
The Analog Out Tab contains the configuration data of the analog output of the device. So in function of the selected device there is a different number of analog outputs with different settings.
In general there are:
- the number of the channel **(Pict.6.21-A)**
- the type of output for the channel and the relative button with pencil to set the output selected **(Pict.6.21-B)**
- the value of the output channel **(Pict.6.21-C)**
- the Safe value of the channel when the device is in WatchDog **(Pict.6.21-D)**
- the PowerUp value of the channel when the device is restarted **(Pict.6.21-E)**
- the button to save Safe Value and PowerUp Value **(Pict.6.21-F)**



Pict. 6.22

Digital In

The Digital In Tab contains the states of digital inputs. In function of the selected device there are different numbers of inputs **(Pict.6.22-A).** If the digital input has logic state high the relative indicator is highlighted in green. There is a button to clear at the same time the states of all rising Latch and falling Latch of digital inputs.**(Pict.6.22-B)**.

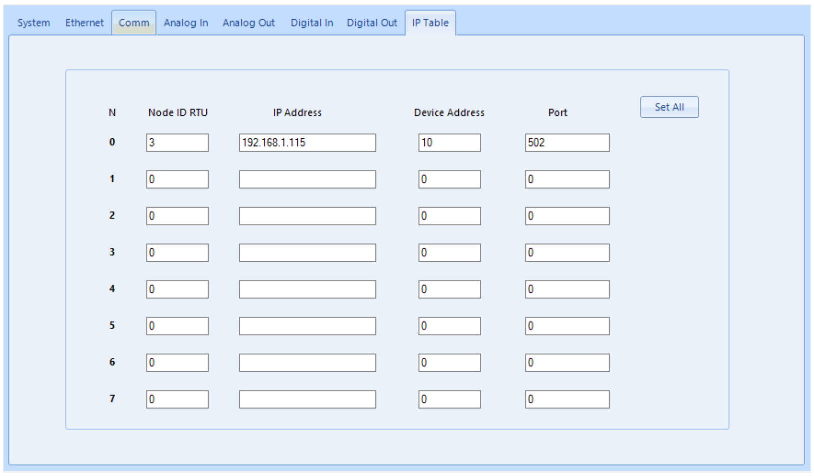


7	6	5	4	3	2	1	0	
0	0	0	1	0	0	0	0	Status
0	0	0	0	1	0	0	0	PowerUp
0	0	0	0	1	0	0	0	Safe

Pict. 6.23

Digital Out

The Digital Out Tab contains the states of digital outputs and the PowerUp and Safe values of the relative outputs. In base of the selected device there are different numbers of outputs **(Pict.6.23-A).**
It is possible to visualize and set the different states of the outputs clicking the proper buttons. When the output has logic state high the button is red.



N	Node ID RTU	IP Address	Device Address	Port
0	3	192.168.1.115	10	502
1	0		0	0
2	0		0	0
3	0		0	0
4	0		0	0
5	0		0	0
6	0		0	0
7	0		0	0

Pict. 6.24

IP Table

The IP table contains the IP addresses of the Slave / Server devices that can be queried on the Ethernet interface with the Modbus TCP / IP protocol **(Pict. 6.24).** A maximum of 8 IP addresses and therefore 8 server devices can be entered.

For more detailed information on the use and insertion of data in the IP table, refer to **paragraph 5.2** of this manual.

6.9 – SCHEDULER WINDOW

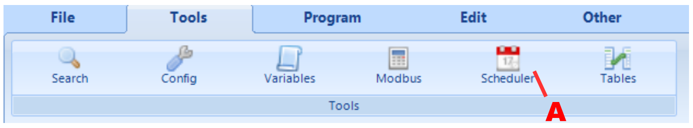
Click the button “*Scheduler*” in the menu bar to open the Scheduler window.**(Pict.6.24-A)**
The Scheduler window **(Pict.6.25)** allows to insert and modify the settings of data recordings, the emails and the synchronized scheduler.

To insert a new element drag it from left list **(Pict.6.25-A)** to right list **(Pict.6.25-B)**.
<u>Click the button “*Create New*” (Pict.6.17-C) to open a wizard window that guides the user to create element without manual insertion.</u>
The available elements are:
- *<u>CSV Standard</u>* : allows to execute a variables recording. The data are stored on USB pen in .CSV format; all files saved in the USB are accessible from the page “Download” of the Web pages.
- *<u>CSV Header</u>* : allows to execute a texts recording. The data are stored on USB pen in .CSV format; all files saved in the USB are accessible from the page “Download” of the Web pages.
- *<u>EMail</u>* : allows the device to send email. Server, Receivers, Body of email are settable from Web pages.
- *<u>Scheduler</u>* : allows to set a mask of bits of a register in order to control the logging events or the emails.

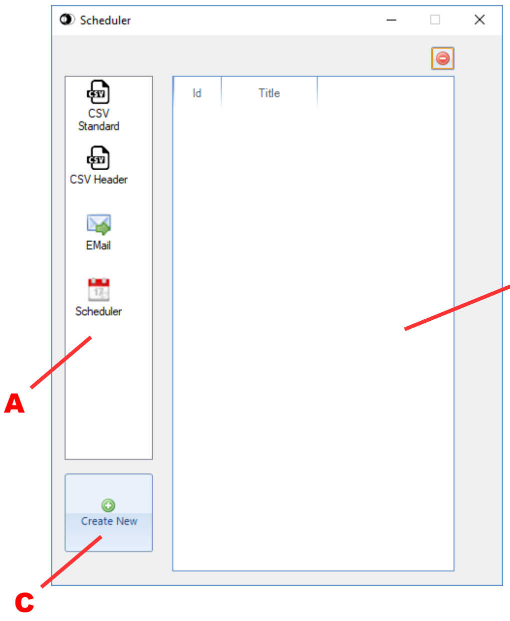
For each element it is possible to set:
- Profile Name **(Pict.6.26-A)**: name used to identify the element.
- Date and time of start and end **(Pict.6.26-B)**: indicates the period of validity of the element that works only between these two dates.
- Event that triggers the action **(Pict.6.26-C)**:
<u>by time</u>, the action is activated at regular intervals of time based on the time set;
<u>by trigger</u>, the action is activated at each variation of the bit's state of a register (rising edge or falling edge).

For CSV Standard, in addition to the common parameters, it is possible to set:
- Destination directory / Frequency to create file **(Pict.6.26-D)**: the destination directory indicates the directory where all the files of the current task are created. If the directory written does not exist, it will be created automatically. The parameter about the time set (hour, day, month, year) indicates the frequency for which a new file containing the data recordings is created and opened for writing. Each time that the period of time set expires the file previously created is closed. The file name is assigned in function of period of time chosen.
- List of variables to save **(Pict.6.26-E)**: draging the variables from the list on the left to the list on the right the fields composing the records will be created(columns of CSV).
To modify the variables click the button “*Variables*” (refer to section “Insertion of Variables, Strings, Texts”).

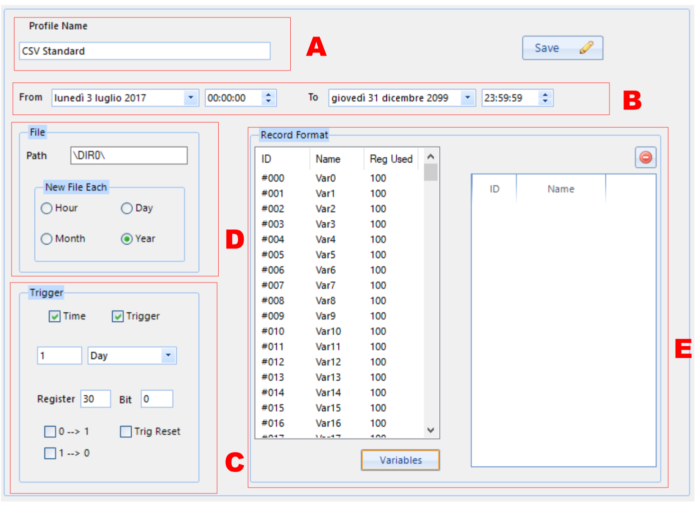
For CSV Header, in addition to the common parameters, it is possible to set:
- Destination directory / Frequency to create file **(Pict.6.27-A)**: the destination directory indicates the directory where all the files of the current task are created. If the directory written does not exist, it will be created automatically. The parameter about the time set (hour, day, month, year) indicates the frequency for which a new file containing the data recordings is created and opened for writing. Each time that the period of time set expires the file previously created is closed. The file name is assigned in function of period of time chosen.
- Record format **(Pict.6.27-B)**: Header Format is the text used for the header of the CSV file, Record Format is the text used for each record of the CSV file.
To modify the texts click the button “*Text*” (refer to section “Insertion of Variables, Strings, Texts”).



Pict. 6.24



Pict. 6.25



Profile Name: CSV Standard

From: lunedì 3 luglio 2017 00:00:00 To: giovedì 31 dicembre 2099 23:59:59

File — Path: \DIR0\
New File Each: Hour / Day / Month / [✓ Year]

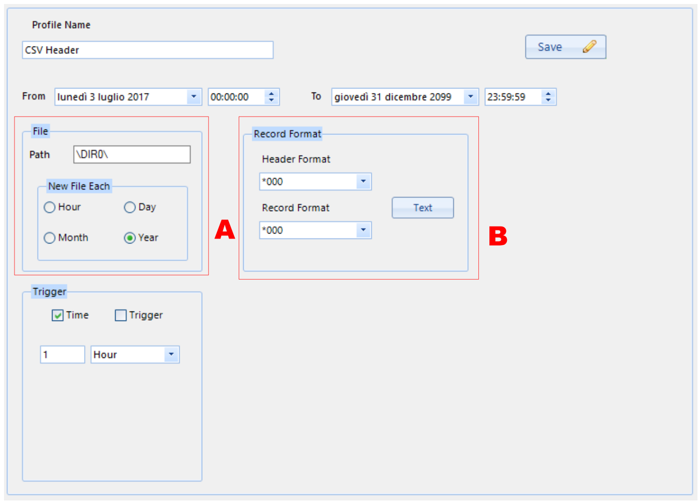
Trigger: [✓ Time] [✓ Trigger]
1 Day
Register 30 Bit 0
0 --> 1 Trig Reset
1 --> 0

Record Format

ID	Name	Reg Used
#000	Var0	100
#001	Var1	100
#002	Var2	100
#003	Var3	100
#004	Var4	100
#005	Var5	100
#006	Var6	100
#007	Var7	100
#008	Var8	100
#009	Var9	100
#010	Var10	100
#011	Var11	100
#012	Var12	100
#013	Var13	100
#014	Var14	100
#015	Var15	100
#016	Var16	100

ID	Name

Pict. 6.26



Profile Name: CSV Header

From: lunedì 3 luglio 2017 00:00:00 To: giovedì 31 dicembre 2099 23:59:59

File — Path: \DIR0\
New File Each: Hour / Day / Month / [✓ Year]

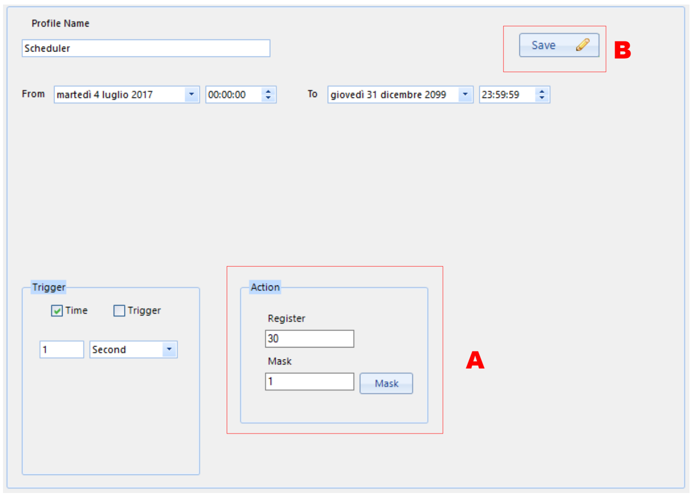
Record Format
Header Format: *000
Record Format: *000

Trigger: [✓ Time] Trigger
1 Hour

Pict. 6.27

For the Emails it is possible to set only the common parameters: Profile Name, Date and time of start and end, event that triggers the Email.
The parameters of the Email message are settable only from Web pages.

For the synchronized scheduler (Scheduler), in addition to the common parameters, it is possible to set the register in which to set the bits in function of a Mask. **(Pict.6.28-A)**



Profile Name	Scheduler
From	martedì 4 luglio 2017 00:00:00
To	giovedì 31 dicembre 2099 23:59:59
Trigger	[✓ Time] Trigger
	1 Second
Register	30
Mask	1

Pict. 6.28

<u>After insertion or modifying the parameters of an element, click the button “*Save*” to save it **(Pict.6.28-B)**.</u>

<u>IMPORTANT:</u>
<u>The device does NOT manage the DST.</u>

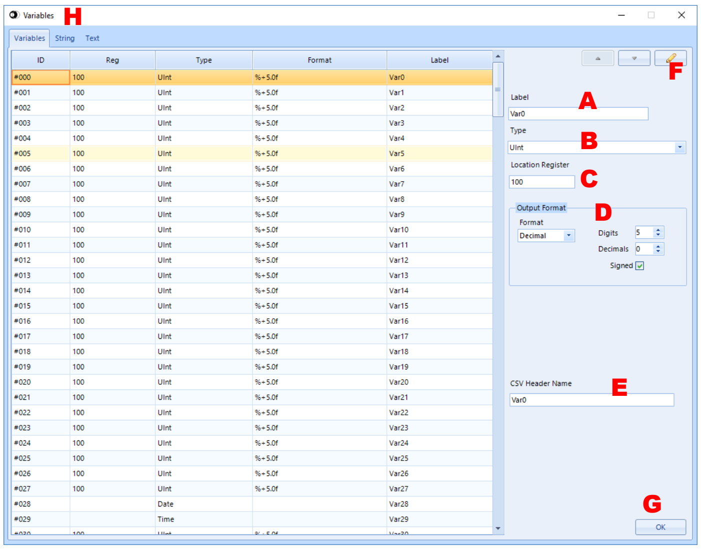
6.10 – INSERTION OF VARIABLES, STRINGS, TEXTS

The Variables window **(Pict.6.29)** allows to modify Variables, Strings and Texts to use in the CSV Data Recordings.

For each variable it is possible to set:
<u>*Label*</u> **(Pict.6.29-A)**: name of the variable (used only in the program as identifier)
<u>*Type*</u> **(Pict.6.29-B)**: type of variable. It can be UInt, Int, Ulong, Long, Float, String(Ram), String(Eprom), Date (variable to record the date), Time (variable to record the time)
<u>*Location Register*</u> **(Pict.6.29-C):** ram register of the device which the variable refers to. For the types Date and Time this field does not exist because the device automatically gets it.
<u>*Output Format*</u> **(Pict.6.29-D)**: format of the variable saved in the .CSV file. It can be set as Decimal, Exponential, or Hexadecimal, with sign, and the number of digits and decimals. For the types Date and Time this field does not exist.
<u>*Set Alternative*</u> **(Pict.6.30):** available only for types String(Ram) and String(Eprom). If selected, it allows to set a register that contains an alternative string**(Pict.6.30-A)** used when the bits of mask of a trigger register are 1 **(Pict.6.30-B)**
<u>*CSV Header Name*</u> **(Pict.6.29-E):** name of the header of the variable in the CSV file.

To confirm and save the parameters of a modified variable click the button with pencil **(Pict.6.29-F)** or the button “OK” that closes the window**(Pict.6.29-G)**.

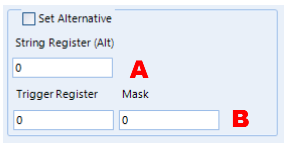
For strings and texts in their tabs **(Pict.6.29-H)** it is possible to modify the content inserting a maximum of 32 characters for the strings and a maximum of 512 characters for the texts. To confirm click the relative button with pencil.



ID	Reg	Type	Format	Label
#000	100	UInt	%+5.0f	Var0
#001	100	UInt	%+5.0f	Var1
#002	100	UInt	%+5.0f	Var2
#003	100	UInt	%+5.0f	Var3
#004	100	UInt	%+5.0f	Var4
#005	100	UInt	%+5.0f	Var5
#006	100	UInt	%+5.0f	Var6
#007	100	UInt	%+5.0f	Var7
#008	100	UInt	%+5.0f	Var8
#009	100	UInt	%+5.0f	Var9
#010	100	UInt	%+5.0f	Var10
#011	100	UInt	%+5.0f	Var11
#012	100	UInt	%+5.0f	Var12
#013	100	UInt	%+5.0f	Var13
#014	100	UInt	%+5.0f	Var14
#015	100	UInt	%+5.0f	Var15
#016	100	UInt	%+5.0f	Var16
#017	100	UInt	%+5.0f	Var17
#018	100	UInt	%+5.0f	Var18
#019	100	UInt	%+5.0f	Var19
#020	100	UInt	%+5.0f	Var20
#021	100	UInt	%+5.0f	Var21
#022	100	UInt	%+5.0f	Var22
#023	100	UInt	%+5.0f	Var23
#024	100	UInt	%+5.0f	Var24
#025	100	UInt	%+5.0f	Var25
#026	100	UInt	%+5.0f	Var26
#027	100	UInt	%+5.0f	Var27
#028		Date		Var28
#029		Time		Var29

Label	Var0
Type	UInt
Location Register	100
Output Format – Format	Decimal
Digits	5
Decimals	0
Signed	✓
CSV Header Name	Var0

Pict. 6.29



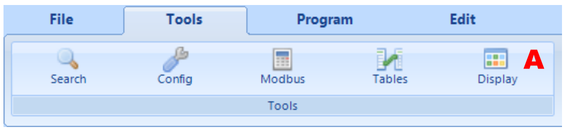
Set Alternative	☐
String Register (Alt)	0
Trigger Register	0
Mask	0

Pict. 6.30

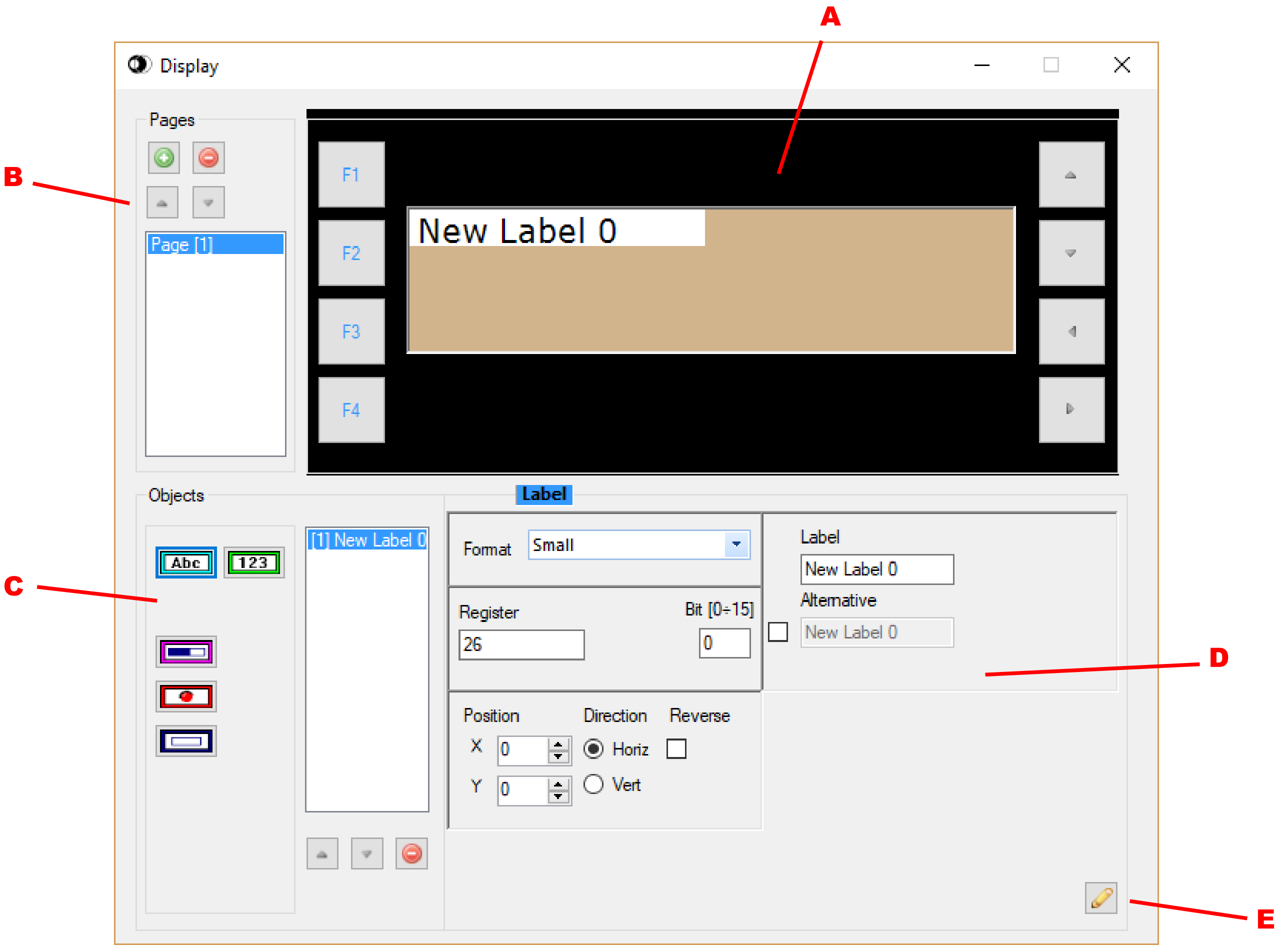
6.11 – GRAPHICS OBJECT MANAGER AND DISPLAY WINDOW

6.11A: WINDOW STRUCTURE

This menu of configuration must be used only to program the graphic display.
To open the *"Display"* window select in the Menu Bar "*Tools → Display*" **(Pict.6.31-A)**
The window is composed of:
- a graphic preview of the display **(Pict.6.32-A)** that allows the user to visualize the position and the structure of the objects inserted;
- a group of operational buttons to work on the structure of the graphic pages **(Pict.6.32-B)**
- a group of operational buttons to work on the graphic objects **(Pict.6.32-C)**
- a window to show the properties of the single graphic object **(Pict.6.32-D)**
- a button to save the properties of the selected object **(Pict.6.32-E)**



Pict. 6.31



Format Small
Label New Label 0
Alternative New Label 0
Register 26
Bit [0÷15] 0
Position
X 0
Y 0
Direction ● Horiz ○ Vert
Reverse ☐

Pict. 6.32

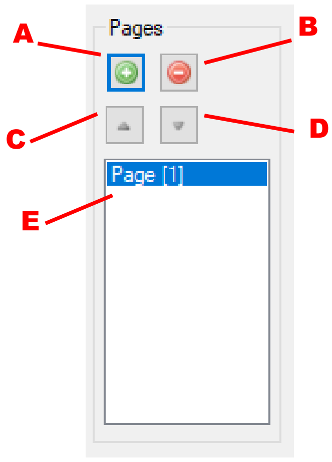
6.11B: GRAPHIC OBJECTS

By the *"Display"* window it is possible to create and set the visualization of the following objects:

-*page*: area of visualization of the objects.
-*label*: strings of 11 alphanumerical characters length; it is possible to visualize the characters that belong to the standard ASCII table (not extended). Note: for the character "°", use the character "^" (example: °C → ^C)
-*numeric value of a register:* visualization, in a format defined by the user, of the value of an Internal register.
-*progress bar:* progress bar with value of filling proportional to the value of an internal register;
-*rectangle:* creation and visualization of simple geometric shapes (rectangle);
-*picture:* creation and visualization of pictures where the filling condition depends of the logic state of the bit of a single register.

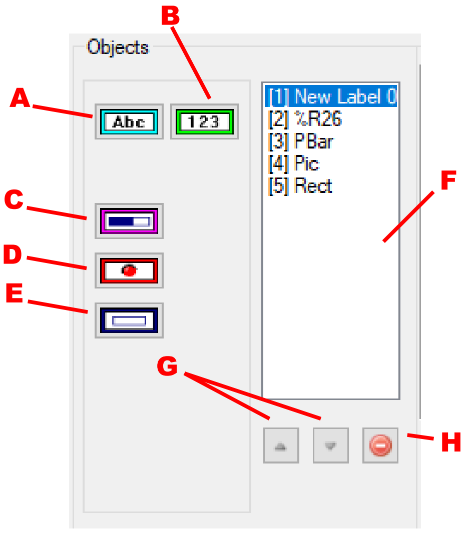
6.11C: CREATION OF THE DISPLAY WINDOW

Work on the graphic pages (Reference Pict.6.33)
To insert a graphic page click the button "*Insert page*" **(Pict.6.33A)**; to the page created will be assigned a progressive identification number and the page will appear in the list of the existing pages. In the list the page selected will be highlighted in blue **(Pict.6.33E).**In case of creation of several pages it is possible to use the scrolling buttons "*Page Up*" **(Pict.6.33C)** and "*PageDown*" **(Pict.6.33D)** to move into the list and select the desired page; the page preview will be updated in function of the page selected. To delete a page click the button "*Delete page*" **(Fig.6.33B).**



Pict. 6.33

Work on the graphic object (Reference Pict.6.34)
- *Label.*
To insert the object "Label", click the button *"Label"* **(Pict.6.34A)*.*** Inside the "*List Object*" **(Pict.6.34F)** the text "New Labe 0" will appear.
- *Numeric value of a register.*
To insert the object "Numeric value", click the button *"Number"* **(Pict.6.34B)*.*** Inside the *"List Object "* **(Pict.6.34F)** the text "%R26" will appear.
- *Progress bar.*
To insert the object "Progress bar", click the button *"Progress bar"* **(Pict.6.34C)*.*** Inside the *"List Object "* **(Pict.6.34F)** the text "Pbar" will appear.
- *Picture.*
To insert the object "Picture", click the button *"Picture"* **(Pict.6.34D).** Inside the *"List Object "* **(Pict.6.34F)** the text "Pic" will appear.
- *Rectangle.*
To insert the object "Rectangle", click on the button *"Rectangle"* **(Pict.6.34E).** Inside the *"List Object "* **(Pict.6.34F)** the text "Rect" will appear.

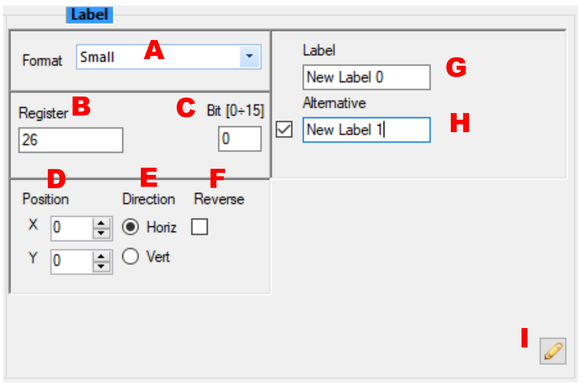


Pict. 6.34

The object inserted will be highlighted in blue inside the *"List Object ".*
Use the scrolling buttons *"Scroll Up"* and *"Scroll Down"* **(Pict.6.34G)** to move inside the *"List Object "* and to select the desired object.
To delete an object, select the object inside the "*List Object*" and click the button "*Delete Object*" **(Pict.6.34H).**

Properties of the graphic objects Window (Reference Pict.6.35)
Inside the Display window, each object is defined by specific properties.
The confirm button with the pencil **(Pict.6.35I)** saves the modifies applied to the selected graphic object and updates the preview of the display. In case of modify of an object and missed click of the confirm button, the modify won't be saved.

Label.
Format **(Pict.6.35A):**Allows to set the size of visualization of the character in two sizes:: Small(6x8p.) and Medium(12x16p.).
Register **(Pict.6.35B):** Allows to set the Internal Register that contains the bit to which the visualization of the dynamic text is connected to.
Bit [0÷15] **(Pict.6.35C):** Allows to set the bit of the Internal register to which the visualization of the dynamic text is connected to.
Position **(Pict.6.35D):** Defines the coordinates, expressed as pixel, of the dynamic text's position inside the graphic pages (X→horizontal: 0 up to 131; Y→vertical: 0 up to 31).
Direction **(Pict.6.35E):** Defines the orientation (horizontal or vertical) of the dynamic text inside the graphic page.
Reverse **(Pict.6.35F):** Defines the visualization of the object as direct or reverse referred to the background of the graphic page.
Label **(Pict.6.35G):** Allows to insert the string of alphanumerical characters (11 max.) that will be visualized if the logic state of the reference bit is 0.
Alternative **(Pict.6.35H):** Allows to insert the string of alphanumerical characters (11 max.) that will be visualized if the logic state of the reference bit is 1. If the flag alternative is not enabled, this parameter has not effect and is equal to Label.
Button *"Confirm"* **(Pict.6.35I).**

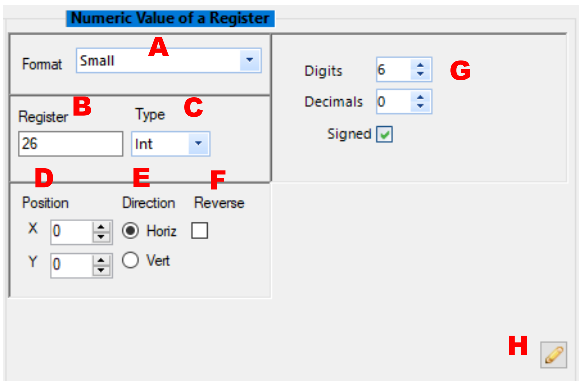


Format: Small
Register: 26
Bit [0÷15]: 0
Position X: 0
Position Y: 0
Direction: [✓ Horiz] Vert
Reverse: ☐
Label: New Label 0
☑ Alternative: New Label 1

Pict. 6.35

Numeric value of a register.

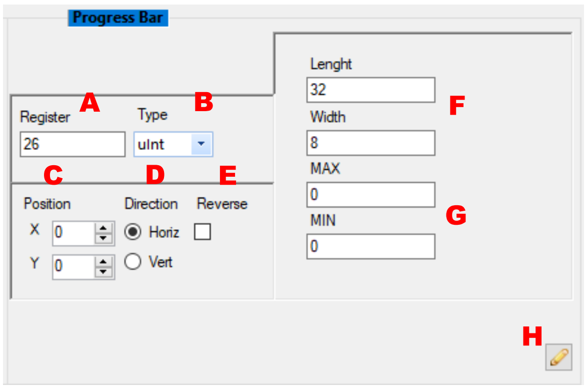
Format **(Pict.6.36A):** Allows to set the size of visualization of the object in three sizes: Small(6x8p.), Medium(12x16p.) and Large (24x32p.).
Register **(Pict.6.36B):** Allows to set the Internal Register of which the value is visualized .
Type **(Pict.6.36C):** Allows to set the format of the Internal Register's value to visualize.
Position **(Pict.6.36D):** Defines the coordinates, expressed as pixel, of the object's position inside the graphic page (X→horizontal: 0 up to 131; Y→vertical: 0 up to 31).
Direction **(Pict.6.36E):** Defines the orientation (horizontal or vertical) of the object inside the graphic page.
Reverse **(Pict.6.36F):** Defines the visualization of the object as direct or reverse referred to the background of the graphic page.
Digits, Decimals, Signed **(Pict.6.36G):** Defines the format of visualization of the value: digits is the number of integers before the commas, decimals is the number of decimals after the commas, signed is checked if the sign is used.
Button *“Confirm”* **(Pict.6.36H).**


Numeric Value of a Register
Format: Small
Register: 26
Type: Int
Position X: 0
Position Y: 0
Direction: [✓ Horiz] Vert
Reverse: ☐
Digits: 6
Decimals: 0
Signed: ✓

Pict. 6.36

Progress bar.

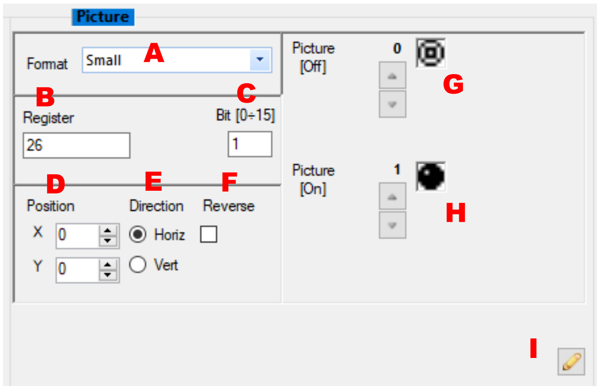
Register **(Pict.6.37A):** Allows to set the Internal Register to which the filling ratio of the bar is connected.
Type **(Pict.6.37B):** Allows to set the format of the Internal Register's value
Position **(Pict.6.37C):** Defines the coordinates, expressed as pixel, of the object's position inside the graphic page (X→horizontal: 0 up to 131; Y→vertical: 0 up to 31).
Direction **(Pict.6.37D):** Defines the orientation (horizontal or vertical) of the object inside the graphic page.
Reverse **(Pict.6.37E):** Defines the visualization of the object as direct or reverse referred to the background of the graphic page.
Dimensions **(Pict.6.37F):** Defines the dimensions (length x width), expressed as pixel, of the object.
Filling constants **(Pict.6.37G):** Defines the values of the Internal Register selected to which the events of Start (MIN) and Stop (MAX) of filling of the bar are connected.
Button *“Confirm”* **(Pict.6.37H).**


Progress Bar
Register: 26
Type: uInt
Position X: 0
Position Y: 0
Direction: [✓ Horiz] Vert
Reverse: ☐
Lenght: 32
Width: 8
MAX: 0
MIN: 0

Pict. 6.37

Picture.

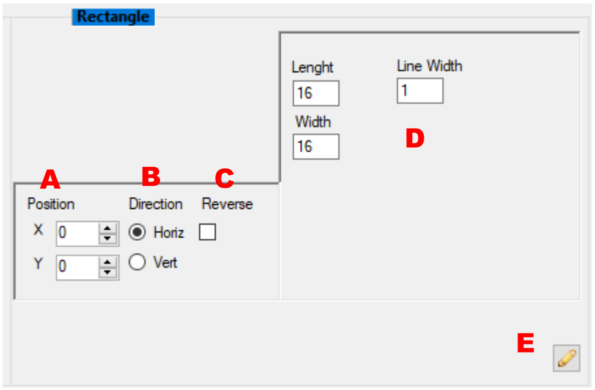
Format **(Pict.6.38A):** Allows to set the size of visualization of the object in three sizes: Small(6x8p.), Medium(12x16p.) and Large (24x32p.).
Register **(Pict.6.38B):** Allows to set the Internal Register to which the visualization of the picture is connected.
Bit **(Pict.6.38C):** Allows to set the bit of the Internal register to which the visualization of the picture is connected to.
Position **(Pict.6.38D):** Defines the coordinates, expressed as pixel, of the object's position inside the graphic pages (X→horizontal: 0 up to 131; Y→vertical: 0 up to 31).
Direction **(Pict.6.38E):** Defines the orientation (horizontal or vertical) of the object inside the graphic page.
Reverse **(Pict.6.38F):** Defines the visualization of the object as direct or reverse referred to the background of the graphic page.
Picture [Off] **(Pict.6.38G):** Indicates how the picture will be visualized if the logic state of the reference bit is 0.
Picture [On] **(Pict.6.38H):** Indicates how the picture will be visualized if the logic state of the reference bit is 1.
Button *“Confirm”* **(Pict.6.38I).**


Picture
Format: Small
Register: 26
Bit [0÷15]: 1
Position X: 0
Position Y: 0
Direction: [✓ Horiz] Vert
Reverse: ☐
Picture [Off]: 0
Picture [On]: 1

Pict. 6.38

Rectangle.

Position **(Pict.6.39A):** Defines the coordinates, expressed as pixel, of the object's position inside the graphic pages (X→horizontal: 0 up to 131; Y→vertical: 0 up to 31).
Direction **(Pict.6.39B):** Defines the orientation (horizontal or vertical) of the object inside the graphic page.
Reverse **(Pict.6.39C):** Defines the visualization of the object as direct or reverse referred to the background of the graphic page.
Dimensions **(Pict.6.39D):** Defines the dimensions (length, width and line tickness), expressed as pixel, of the object.
Button *“Confirm”* **(Pict.6.39E).**


Rectangle
Lenght: 16
Width: 16
Line Width: 1
Position X: 0
Position Y: 0
Direction: [✓ Horiz] Vert
Reverse: ☐

Pict. 6.39

7.1 – ETHERNET CONNECTION

On the Ethernet side, the Controller works like a Server, therefore for the connection to the LAN network it is necessary to follow the standards for the Ethernet connections. Hereafter some practical tips to connect the Controller are reported.

To connect the Controller directly to a PC, use a crossover cable. To connect the Controller to an Hub, Switch or Router, use a direct cable.

Due to their settings, it could happen that some Firewalls won't allow the communication with the Controller; this kind of problem could happen particularly in phase of Search: in case of communication problems it is suggested, if possible, to disable eventual active Firewalls on the Client PC or Router.

If the DHCP service (Dynamic Host Communication Protocol) is not in use, be sure that the IP, the Subnet Mask and the Gateway address of the Controller will be compatible with the settings of the LAN network which the Controller is connected to.

8.1 – IMPORTANT MESSAGES IN THE LOG PANEL OR IN POP-UP WINDOW

MESSAGE	POSSIBLE CAUSES	POSSIBLE SOLUTIONS
"Connected" ***"... - Connected"***	- The device is just connected using the correct parameters	
"Disconnected" ***"... - Disconnected"***	-The device previously connected has just been disconnected or it is not possible to communicate with it	If it is not possible to communicate with the device verify the physical connections and the parameters and reconnect to it *(Par 6.1, Par 6.2)*
"Timeout Connection"	- It is not possible to communicate with the device over Ethernet, because it is in Socket Timeout or it has been physical disconnected.	-The device is in Socket Timeout for no actions: reconnect it *(Par 6.1, Par 6.2)* -The device is physically disconnected from the Net: verify the connections and reconnect.
"Read/Write COM Error"	- An error occurs during the communication over Serial Port	- The device is physically disconnected: verify the connections and reconnect.
"Socket Timeout"	- An error occurs during the communication over Ethernet. The device is not in the Net or the inserted IP doesn't exist.	- The device is not in the Net or the inserted IP doesn't exist: verify the connection parameters, the connections and try to connect.
"Timeout Error"	- An error occurs during the communication over Ethernet. The device has a Modbus Address different from the one selected	-The device has a Modbus Address different from the one selected: verify the connection parameters, the connections and try to connect.
"Com Error"	- An error occurs during the communication over Serial Port. The device is not connected to the PC via Com port or wrong parameters have been inserted.	- The device is not connected to the PC via Com port: verify connections and reconnect. - Wrong parameters are inserted: verify Serial connection parameters (Com port, Baud Rate, Modbus address, etc...), and try to connect.
"Process Validation: Complete"	- The diagram realized doesn't contain structural errors and so it is possible to download it in the Eprom of the device	
"Process Validation: Error"	- The diagram realized contains structural errors. It is not possible to download it in the Eprom of the device	- Check the diagram looking for the errors
"Download Complete"	- Download Eprom successful.	
"No Device Connected".	- No connection to a device has been executed or the device have been previously disconnected from the program	- If no connection to a device has been executed and the program is in Offline mode, connect to a device. - If the device have been previously disconnected or a communication error occurs, verify communication parameters, the connections and try to connect.

8.2 – POSSIBLE CAUSES OF FAULT

EVENT	POSSIBLE CAUSES	POSSIBLE SOLUTIONS
It is not possible to power-on the Controller.	-The Controller is not correctly powered. -The value of the power supply value is lower than the specifications limits.	-Refer to the data-sheet of the Controller in use and verify the relative Technical Specifications.
There is not communication between the Host PC and the Controller.	-Ethernet port not correctly connected. -Modbus Slave port not correctly Connected or set. -Eventual interface between PC and Controller not correctly connected. -Wrong communication parameters.	-Refer to the section 7.1 -Refer to the data-sheets of the Controller and the Interface device in use. -Refer to the section 8.1
There is not communication between the Controller and one or more Modbus slave devices.	-Modbus Master port not correctly connected. -The slave device is not correctly powered -The slave device is not correctly connected on the RS-485 serial line. -Wrong communication parameters. -The Modbus addresses of the slave devices connected are not included in the range set in the System Register %R17 (Gateway Mask) .	-Refer to the section 8.1 -Refer to the data-sheets of the Controller and the Slave devices in use. -Slave device in INIT condition and Baud-rate of communication different of 9600 bps. -Verify the values of Gateway Mask.
The Program is not correctly executed or it is impossible to execute the Program.	-Wrong communication parameters. -The Controller is in “Debug” modality and in Halt, Stop or Break Point condition. -Wrong data-format of the Registers. -Wrong parameters of the Function Block. -Parameters of the eventual Slave devices connected not correctly inserted. -The Program has not been downloaded -Controller in INIT modality.	-Refer to the section 8.1 -Set the Controller in “Debug” modality and in “Run” condition or in “Release” modality. -Remove eventual Break Points. -Set the correct data-format of Registers. -Verify the parameters of Function Block (data-format, masks, tables, etc..). -Control the configuration of the Slave devices (type of input and output, etc..) -Download the Program. -Control if the INIT modality is active.
The configuration of the Controller is unknown.	-	-Set the Controller in “INIT” modality; the parameters of configuration of the Controller will be forced to the default values listed in section 6.6 .
The Controller is connected in “INIT” modality but is not executed (where foresee the LED “STS” doesn't blink) or there is not communication between the Host PC and the Controller.	-Controller not correctly connected. -Wrong port Baud Rate.	-Connect the terminal INIT to GND. -Switch-off and than power-on the Controller after the connection of the terminal INIT to GND. -Set the Baud-rate of the Slave Port as 9600 bps and Address 10.
The function “Search” doesn't find any Controller.	-There are not Controllers connected. -Controllers not correctly connected. -The Controllers connected by Ethernet port has been set with communication parameters not compatible with the Ethernet interface of the Host PC in use. -On the network there are active Firewall or routers that block the access to the Controller.	-Refer to the data-sheet of the Controller in use and verify the relative Technical Specifications. -Verify the parameters of the Ethernet interface of the Host PC. -Call the System Administrator in order to connect the controller to the network.

EVENT	POSSIBLE CAUSES	POSSIBLE SOLUTIONS
The functions Clock and Calendar (where foresee) don't work correctly.	-Battery low or absent. -Clock and Calendar parameters not correctly set in the proper Registers	-Change or insert the battery. -Control the parameters of the System Registers (refer to the sections 3.3 and 3.4).
The function "Search" doesn't find any Slave device.	-There are not slave devices connected . -The slave devices are not correctly connected. -The Controller which the slave devices are connected to has not been selected. -The Modbus addresses of the slave devices connected are not included in the range set in the System Register %R17 (Gateway Mask) or in the range set in the menu "Search". - The baud-rate of the Slave devices connected is not the same of that set for the Master port of the Controller. -The Timeout values are not correct.	-Refer to the data-sheets of the slave devices in use and verify the relative Technical Specifications. -Verify that the Controller selected is the same which the slave devices are connected to. - Verify the correspondence between settings and Modbus addresses of the slave devices. -Verify the values of Gateway Mask. -Control the baud-rate and delay time of the slave devices connected.
The Web Pages haven't been loaded.	-The IP address written in the address bar of the Internet browser is not the same of Controller's IP address. -The Controllers connected by Ethernet port has been set with communication parameters not compatible with the Ethernet interface of the Host PC in use. -On the network are active Firewall or Routers that block the access to the Controller	-Verify the IP address written in the address bar. -Verify the parameters of the Ethernet interface of the Host PC. -Call the System Administrator in order to connect the controller to the network.
The data saved as constant in the Register table, are not saved when the Controller is switched off.	- The data have been saved in General Purpose Registers instead of Retentive Registers .	-Save the constant values in Retentive Registers.

9.1 – E-MAIL CONFIGURATION

1. Introduction

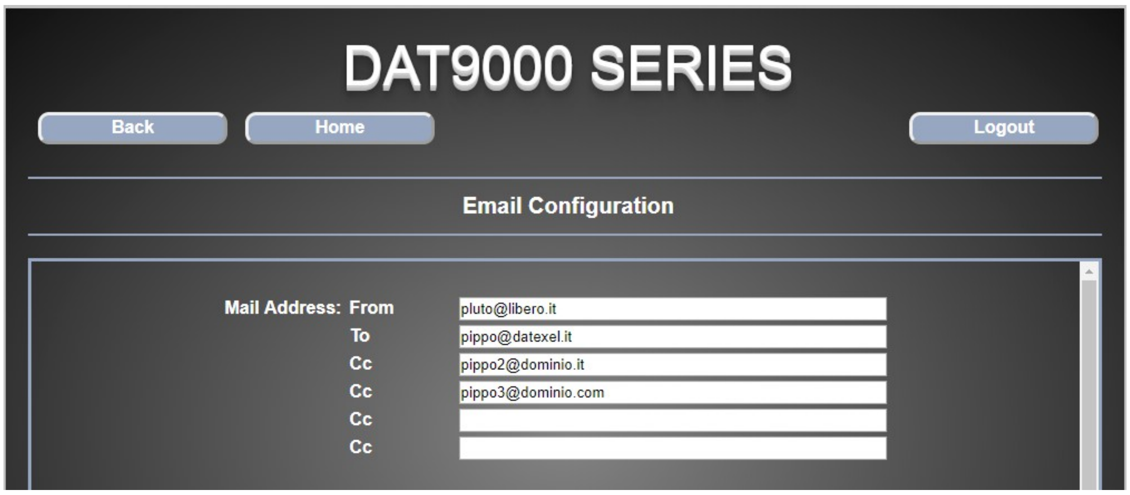
This Application Note shows how to send an e-mail on event or with a specific timestamp. The e-mail that is sent uses **port 25 without encryption**. It is possible to send one and only one e-mail. The message is written in the body of the email and can contain a fixed string of characters, numerical variables or the composition of one or more strings together with the variables.
Furthermore, it is also possible to add blocks of text.

2. E-mail configuration by DAT9000 web page

Once logged in by entering the correct *username and password*, the DAT9000 HomePage will appear.
By clicking on the **"Email Configuration"** button it is possible to access the e-mail setup page.

- Mail Address

From: e-mail address that uses port 25 without encryption (if not already available, it is necessary to create a compatible account).
_To:_ final destination e-mail address (can be any type of account).
_Cc:_ e-mail address of recipients in "Knowledge Copy" (can be any account).

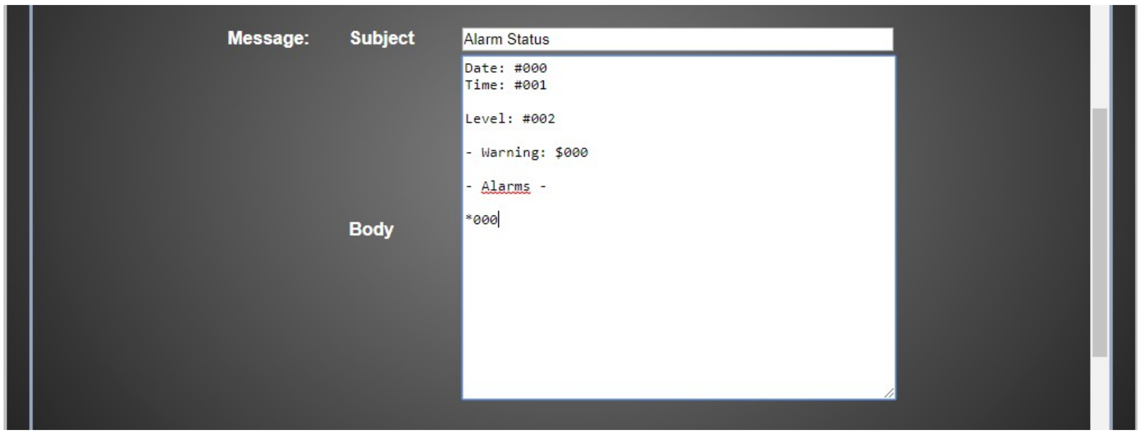


Mail Address:	From	pluto@libero.it
	To	pippo@datexel.it
	Cc	pippo2@dominio.it
	Cc	pippo3@dominio.com
	Cc	
	Cc	

- Message

Subject: subject of the e-mail.

Body: body or message to send. The variables #XXX, the strings $YYY and the blocks of text *ZZZ are defined through the Dev9k respectively in the **"Variables"**, **"String"** and **"Text"** windows. In the *"Variables"* window an internal register of the DAT9000 is associated with a format variable #XXX. In the "String" window it is possible to associate a string of characters with the format $ YYY. In the *"Text"* window it is possible to associate to the *ZZZ format a text longer than a simple string.



Message: Subject: Alarm Status

Body:

```
Date: #000
Time: #001

Level: #002

- Warning: $000

- Alarms -

*000
```

Server Config

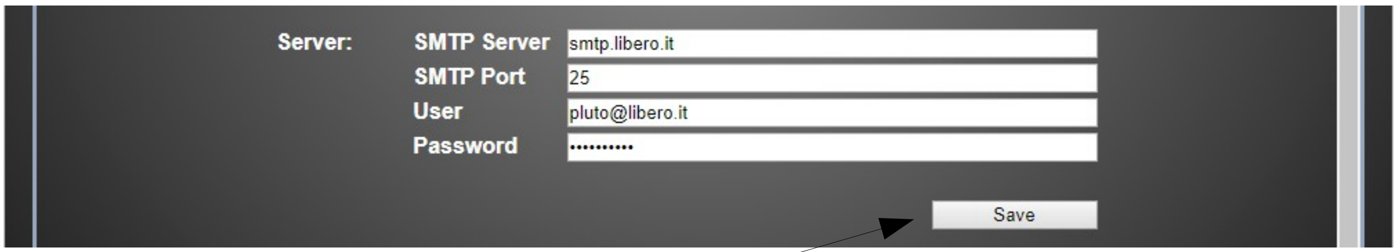
In this area configure the outgoing mail server that uses port 25 without encryption.

SMTP Server: is the outgoing mail server. It depends on the outgoing mail account's domain.
This data can be easily found on the net (in the example, the outgoing mail account is on “*Libero*”).

SMTP Port: is the port that is used by the outgoing mail server (in our case, port 25).

User: outgoing mail account.

Password: password of the outgoing mail account.



Server:

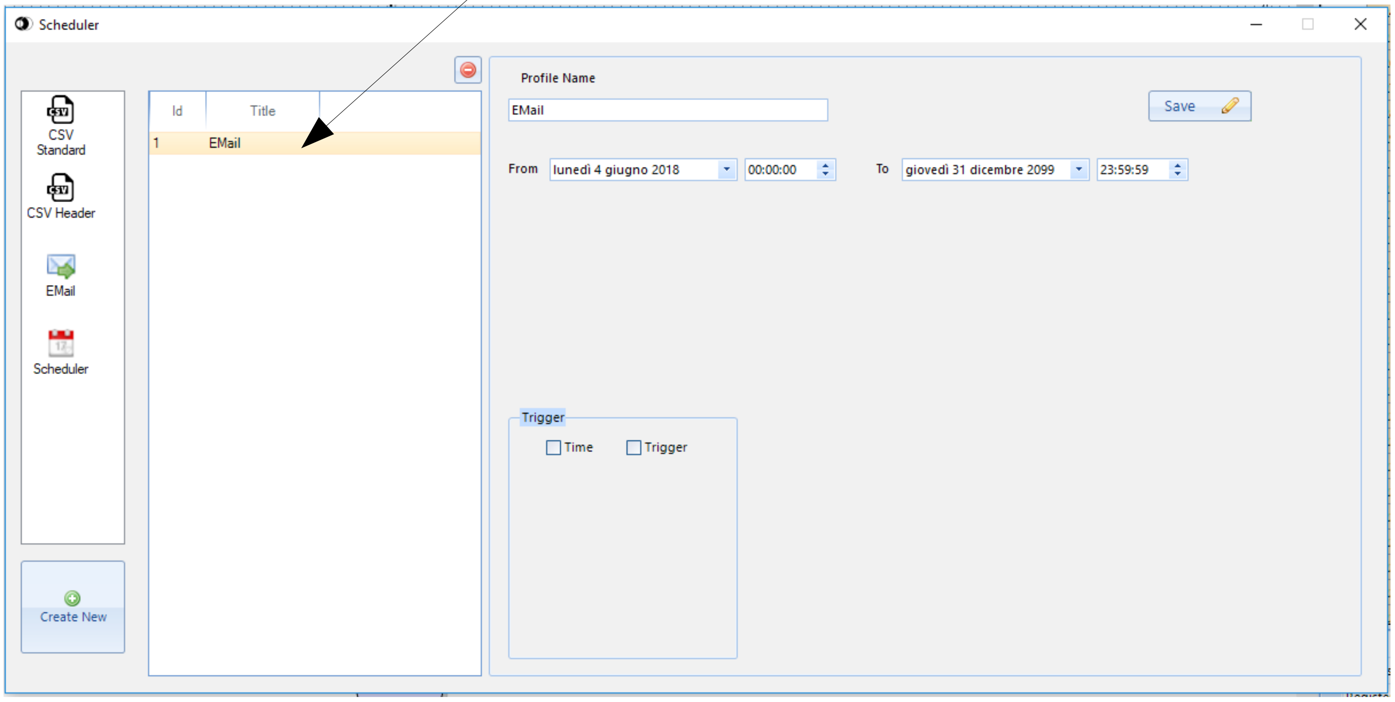
SMTP Server	smtp.libero.it
SMTP Port	25
User	pluto@libero.it
Password	••••••••••

Click on **"Save"** to save the changes of the entire page.

It is possible to send a test email after saving the changes by clicking on the **"Test"** button at the bottom left.

3. Configuration of e-mail timestamp in the Dev9k software

The email configured via the web page can be sent on an event (the event is associated to a bit) or with a well-defined timestamp (every hour, every day every 10 minutes, etc.).
To define this it is necessary to insert the "Mail" field through the Dev9k by accessing the *Tools → Scheduler* and clicking first on the symbol to select the "Mail" and after dragging the symbol in the white area closing to the right.
At this point the following window will appear:



Id	Title
1	EMail

Profile Name: EMail

From lunedì 4 giugno 2018 00:00:00 To giovedì 31 dicembre 2099 23:59:59

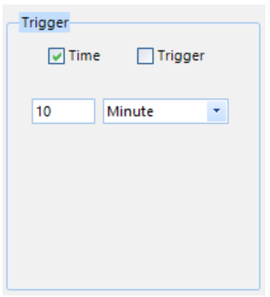
Trigger: Time Trigger

In the *"Trigger"* tag it is possible to set the event or the timestamp to send the e-mail. Therefore there are two possibilities to send an e-mail, *"by Time"* or *"on Event"*.

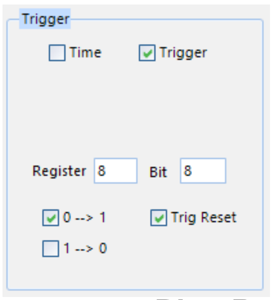
- By selecting the *"Time"* checkbox, the e-mail will be sent *"by Time"* ie periodically regardless of the status of any bit.
  The timestamp is set through the drop-down menu and the relative input field (Pict.. A).
  In the example, the e-mail will be sent automatically every 10 minutes.

- By selecting *"Trigger"*, the e-mail is sent in relation to an "Event". That can be related to the change of the status of a given bit, that belongs to a specific register (Pict. B). It is possible to select which status change generates the sending of the e-mail, ie sending on the rising edge of the bit (from 0 → 1) or on the falling edge of the bit (from 1 → 0).
  If *"Trig reset"* is flagged when the rising edge, when the e-mail is sent, the bit that has generated the event of sending e-mail is automatically resetted.
  In the example, the e-mail will be sent every time the bit 8 of the register 8 changes from 0 to 1 (on the rising edge). Having also selected the *"Trig Reset"* option, bit 8 of register 8 automatically resetted as soon as the e-mail is sent.

At the end of the parameter setting, click **"Save"** to save.


Trigger
[✓ Time] Trigger
10 Minute

Pict. A


Trigger
Time [✓ Trigger]
Register 8 Bit 8
[✓ 0 --> 1] [✓ Trig Reset]
1 --> 0

Pict. B

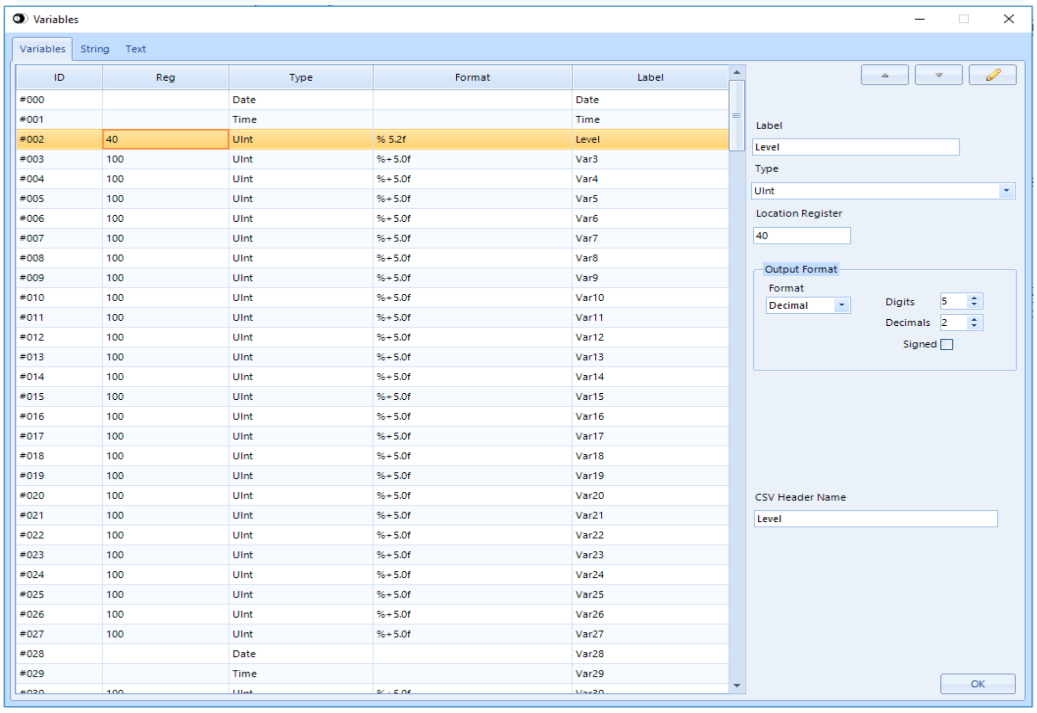
It is also possible to use the ***"Create New"*** *button which allows the user to enter the e-mail through a wizard.*

4. Setting variables, strings and text blocks in the Dev9k software

Variables #XXX, strings $YYY and text blocks *ZZZ can be inserted in the body of the mail and be defined in the Dev9k software.

To access the variable definition window, click on *Tools* on the menu and then on *Variables*:

- Variables



ID	Reg	Type	Format	Label
#000		Date		Date
#001		Time		Time
#002	40	UInt	% 5.2f	Level
#003	100	UInt	%+5.0f	Var3
#004	100	UInt	%+5.0f	Var4
#005	100	UInt	%+5.0f	Var5
#006	100	UInt	%+5.0f	Var6
#007	100	UInt	%+5.0f	Var7
#008	100	UInt	%+5.0f	Var8
#009	100	UInt	%+5.0f	Var9
#010	100	UInt	%+5.0f	Var10
#011	100	UInt	%+5.0f	Var11
#012	100	UInt	%+5.0f	Var12
#013	100	UInt	%+5.0f	Var13
#014	100	UInt	%+5.0f	Var14
#015	100	UInt	%+5.0f	Var15
#016	100	UInt	%+5.0f	Var16
#017	100	UInt	%+5.0f	Var17
#018	100	UInt	%+5.0f	Var18
#019	100	UInt	%+5.0f	Var19
#020	100	UInt	%+5.0f	Var20
#021	100	UInt	%+5.0f	Var21
#022	100	UInt	%+5.0f	Var22
#023	100	UInt	%+5.0f	Var23
#024	100	UInt	%+5.0f	Var24
#025	100	UInt	%+5.0f	Var25
#026	100	UInt	%+5.0f	Var26
#027	100	UInt	%+5.0f	Var27
#028		Date		Var28
#029		Time		Var29

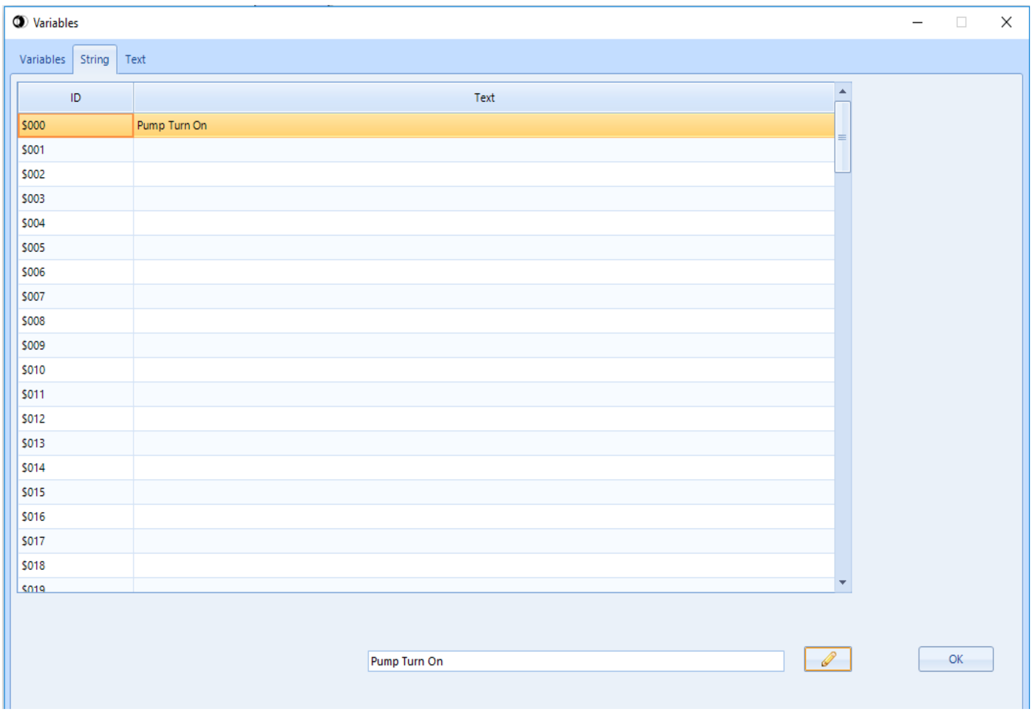
Label: Level
Type: UInt
Location Register: 40
Output Format
Format: Decimal
Digits: 5
Decimals: 2
Signed: ☐
CSV Header Name: Level

Through this window it is possible to associate the ID (#XXX) that will be written in the body of the mail to a predefined variable (Date and Time) or an internal register of the DAT9000. To enter the variables, select the line and modify the parameters in the table. Finally click on ✎ to edit the changes.

Repeat the operation for all the desired variables.

After entering and editing the variables with ✎, press **OK** to confirm.

- Strings



ID	Text
$000	Pump Turn On
$001	
$002	
$003	
$004	
$005	
$006	
$007	
$008	
$009	
$010	
$011	
$012	
$013	
$014	
$015	
$016	
$017	
$018	
$019	

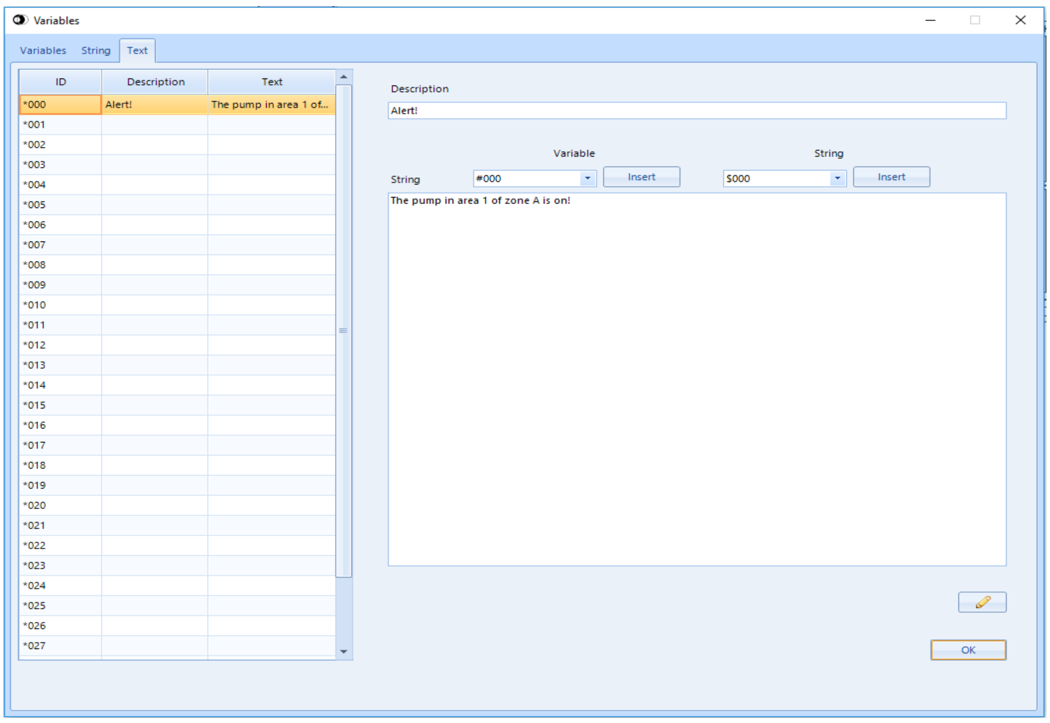
Pump Turn On

Through this window it is possible to associate the ID ($YYY) that will be written in the body of the email to a string of characters that can identify an alarm or any event. To enter the string, select the ID, write in the dedicated space and click on ✎ to edit the changes.

Repeat the operation for all the desired strings.

After entering and editing the strings with ✎, press **OK** to confirm.

- Text Blocks



ID	Description	Text
*000	Alert!	The pump in area 1 of...
*001		
*002		
*003		
*004		
*005		
*006		
*007		
*008		
*009		
*010		
*011		
*012		
*013		
*014		
*015		
*016		
*017		
*018		
*019		
*020		
*021		
*022		
*023		
*024		
*025		
*026		
*027		

Description: Alert!

String — Variable: #000 — String: $000

The pump in area 1 of zone A is on!

Through this window it is possible to associate the ID (*ZZZ) that will be written in the body of the email to a text. Enter the desired text in the appropriate area and click on ✎ to edit the changes.

It is possible to insert variables and strings either by writing the ID directly in the body of the text or by choosing from the drop-down menus relative to the IDs.

Repeat the operation for all the desired text blocks.

After entering and editing the text blocks with ✎, press **OK** to confirm.