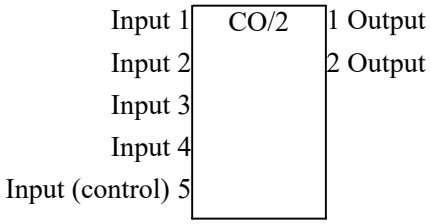
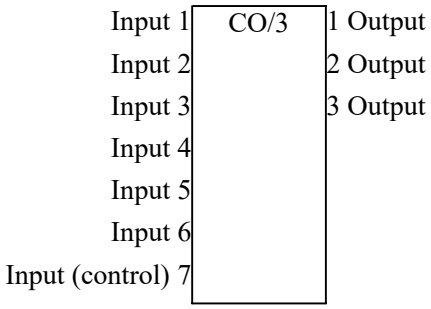
# 1 *CO/2* Change Over - Two Outputs




**Change over 2:** Transfers the values from the inputs to the outputs.
If the control input ( input 5) is set to "0" , then the outputs will receive the values from inputs "1" and "3" .
If the control input ( input 5) is set to "1", then the outputs will receive the values from inputs "2" and "4".

| Input 5 | Output 1 | Output 2 |
|---|---|---|
| 0 | Input 1 | Input 3 |
| 1 | Input 2 | Input 4 |

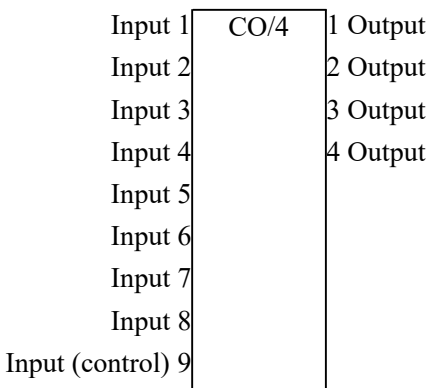
# 2 *CO/3* Change Over - Three Outputs




**Change over 3:** Transfers the values from the inputs to the outputs.
If the control input ( input 7) is set on "0", then the outputs will receive the values from inputs "1" , "3" and "5".
If the control input ( input 7) is set on "1", then the outputs will receive the values from inputs "2" , "4" and "6".

| Input 7 | Output 1 | Output 2 | Output3 |
|---|---|---|---|
| 0 | Input 1 | Input 3 | Input 5 |
| 1 | Input 2 | Input 4 | Input6 |

# 3 *CO/4* Change Over - Four Outputs

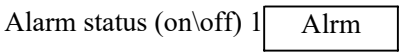


**Change over 4:** Transfers the values from the inputs to the outputs.
If the control input ( input 9) is set on "0" , then the outputs will receive the values from the inputs "1" , "3" , "5" and "7".
If the control input ( input 9) is set on "1", then the outputs will receive the values from the inputs "2" , "4" , "6" and "8".

| Input 9 | Output 1 | Output 2 | Output 3 | Output 4 |
|---|---|---|---|---|
| 0 | Input 1 | Input 3 | Input 5 | Input 7 |
| 1 | Input 2 | Input 4 | Input6 | Input 8 |

---

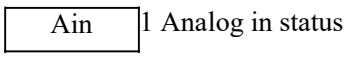
# 4 *Alrm* Alarm Flag




**Alarm function:**Alarm status.
The alarm status in the controller will change to ON once the function receive value "1" in its input .

---

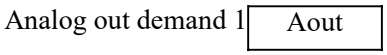
# 5 *Ain* Analog Input




**Analog input :**Provides the real time status of the relevant (internal number) analog input of the controller.

---

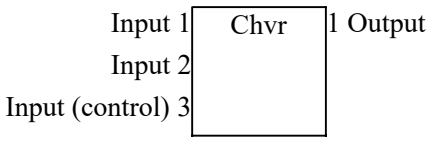
# 6 *Aout* Analog Output




**Analog out:** The proportional analog output singal for the relevant (intenral number) analog output of the controller . Important - output functions can be used only once in the program.

---

# 7 *Chvr* Change Over - single Output

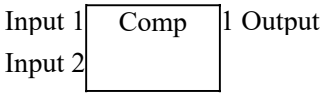



**Change over:**Transfers the values from the Inputs to the output of the function, depending on the status of the control Input .
Values of the two first Inputs are passed to the Output **according to the state of control input (3).**
If Input 3 = "0" then Input 1 is passed to the Output.
If Input 3 = "1" then Input 2 is passed to the Output.

| Input 3 | Output |
|---|---|
| 0 | Input 1 |
| 1 | Input 2 |

---

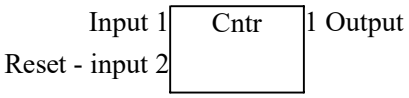
## 8 *Comp* Compare




**Comp function:**Compares between the first and the second input .
If Input 1 is bigger or **equal** to Input 2 then the Output will change from "0" to "1" .

---

## 9 *Cntr* Counter



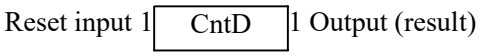

**Counter:** Counts the amount of pulses that received in Input 1 of the function. The count result appears at the output of the function.
In order to reset the counter, set the Input 2 status to "1".

Input 1 -pulse in.
Input 2 -reset.
Output -amount of pulses.

| Input 2 | Output |
|---|---|
| 0 | counting process (Input 1) |
| 1 | reset |

---

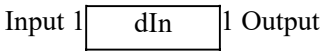
## 10 *CntD* Coun D.In




**Cntd:** Counts the amount of pulses received in the fast digital inputs of the controller. Displays the amount of pulses at the output of the function.
**Input -** Counter reset - if the input of the function receives value "1" the counter result (output) will be reset to 0 .
**Output -** Amount of pulses .

---

## 11 *dIn* Digital Input



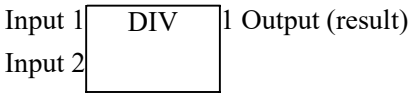

**DIn :** The output of the function provides the status of the relevant (internal number) digital input of the controller ("1" or "0").

---

## 12 *DOut* Digital Output


**Dout:** Digital (ON\OFF) output singal to be acitvated on relevant digital output of the controller.Important - output functions can be used only once in the program.

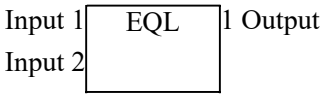
## 13 *DIV* Dividing one number with the other




**Div:** The DIV function divides input 1 on input 2 and displays the result at the output of the function.
**Output = Input 1 / Input 2.**

---

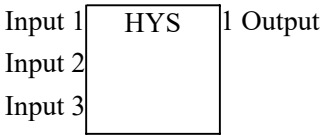
## 14 *EQL* If Input1 Equal to Input2



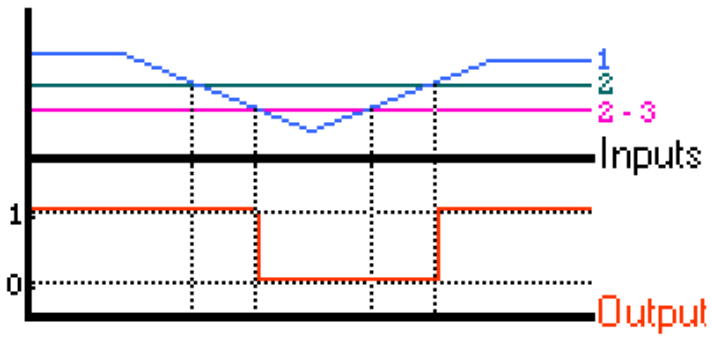

**Eql:** EQL function compares between input 1 and input 2, if the two inputs are equal the output will change to "1", else the output will stay "0".

---

## 15 *HYS* Hysteresis

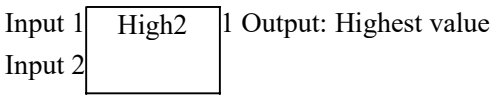



If Input 1 is higher or equal to Input 2, then the Output will change to "1".
If Input 1 is **lower** than Input 2 minus Input 3, the Output changes to "0".




---

## 16 *High2* Maximum value

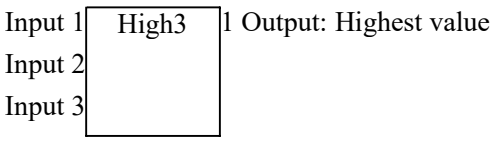



**High 2:** High 2 function displays the highest value between the two inputs by transfering it to the output.

**Input 1:** Value 1
**Input 2:** Value 2
**Output:** Displays the highest value between the inputs.

---

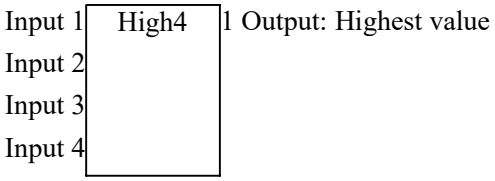
## 17 *High3* Maximum value 3




**High 3:** High 3 function displays the highest value between the three inputs by transfering the highest value to the output of the function.
**Input 1:** Value 1.
**Input 2:** Value 2.
**Input 3:** Value 3.
**Output:** Displays the highest value between the inputs.

---

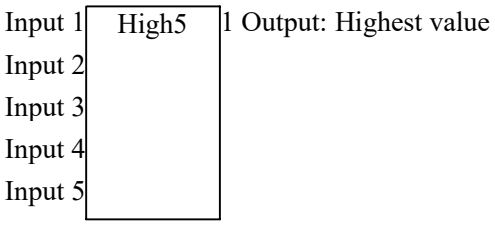
## 18 *High4* Maximum value 4




**High 4:** High 4 function displays the highest value between the four inputs by transfering the highest value to the output of the function.
**Input 1 to Input 4:** Inserted values.
**Output:** Displays the highest value between the four inputs

---

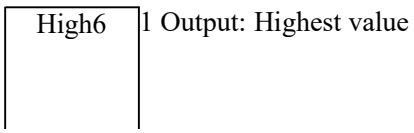
## 19 *High5* Maximum value 5

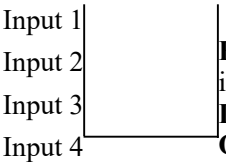



**High 5:** High 5 function displays the highest value between the five inputs by transfering the highest value to the output of the function.
**Input 1 to Input 5:**Inserted values.
**Output:** Displays the highest value between the five inputs.

---

## 20 *High6* Maximum value 6

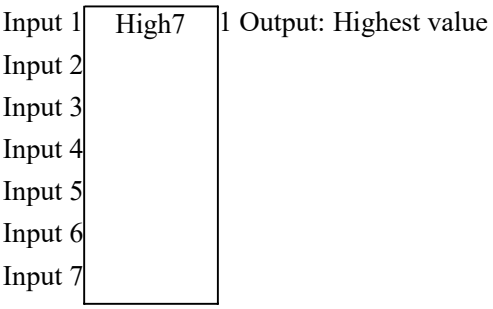





**High 6:** High 6 function displays the highest value between the six inputs by transfering the highest value to the output of the function.
**Input 1 to Input 6:** Inserted values.
**Output:** Displays the highest value between the inputs.

---

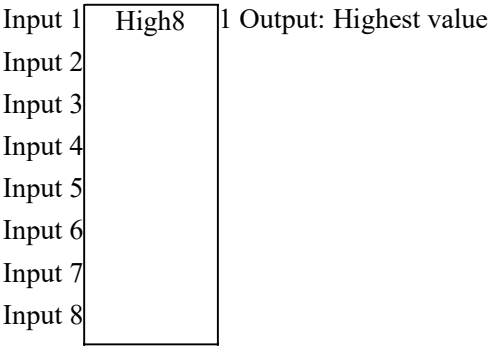
## 21 *High7* Maximum value 7




**High 7:** High 7 function displays the highest value between the seven inputs by transfering the highest value to the output of the function.
**Input 1 to Input 7:** Inserted values.
**Output:**Displays the highest value between the seven inputs.

---

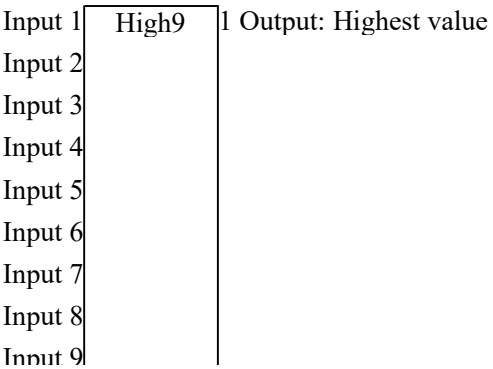
## 22 *High8* Maximum value 8




**High 8:**High 8 function displays the highest value between the eight inputs by transfering the highest value to the output of the function.
**Input 1 to Input 8:** Inserted values.
**Output:** Displays the highest value between the eight inputs.

---

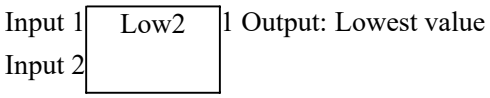
## 23 *High9* Maximum value 9



**High 9:**High 9 function displays the highest value between the nine inputs by transfering the highest value to the output of the function.
**Input 1 to Input 9:** Inserted values.
**Output:** Displays the highest value between the nine inputs.

---

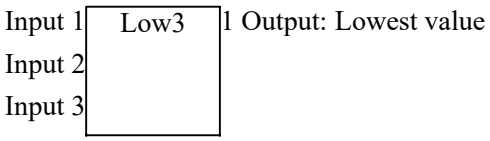
## 24 *Low2* Maximum value 2




**Low 2 function:** Low 2 function displays the lower value between the two inputs by transfering the lowest value to the output of the function.
**Input 1 to Input 2:** Inserted values.
**Output:** Displays the lowest value between the inputs.

---

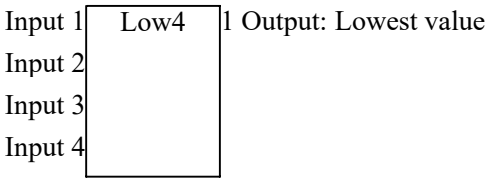
## 25 *Low3* Maximum value 3




**Low 3 function:** Low 3 function displays the lower value between the three inputs by transfering the lowest value to the output of the function.
**Input 1 to Input 3:** Inserted values.
**Output:** Displays the lowest value between the three inputs.

---

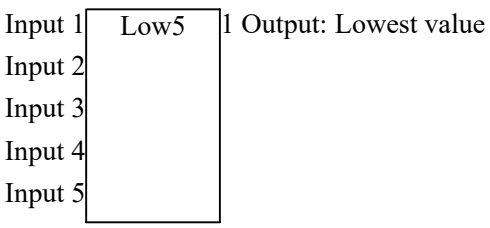
## 26 *Low4* Maximum value 4




**Low 4 function:** Low 4 function displays the lower value between the four inputs by transfering the lowest value to the output of the function .
**Input 1 to Input 4:** Inserted values.
**Output:** Displays the lowest value between the four inputs.

---

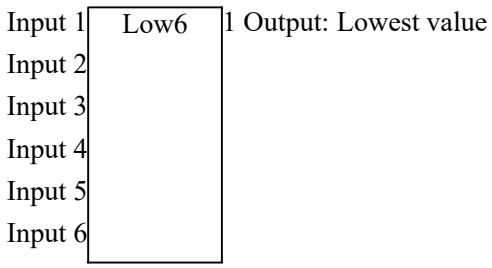
## 27 *Low5* Maximum value 5



**Low 5 function:** Low 5 function displays the lower value between the five inputs by transfering the lowest value to the output of the function .
**Input 1 to Input 5:** Inserted values.
**Output:** Displays the lowest value between the five inputs.

---

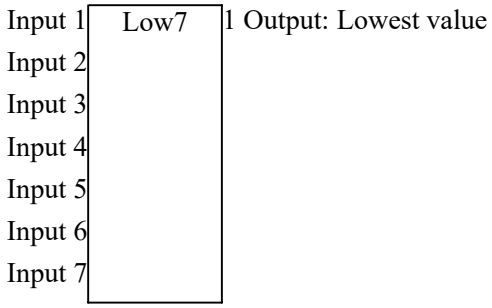
## 28 *Low6* Maximum value 6




**Low 6 function:** Low 6 function displays the lower value between the six inputs by transfering the lowest value to the output of the function .
**Input 1 to Input 6:** Inserted values.
**Output:** Displays the lowest value between the six inputs.

---

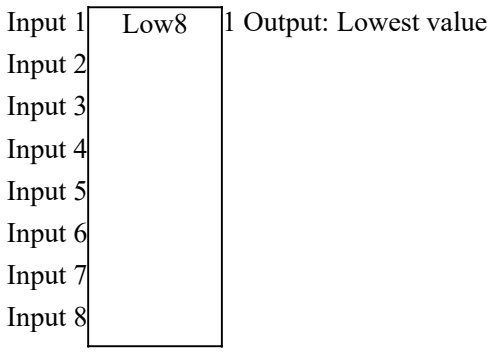
## 29 *Low7* Maximum value 7




**Low 7 function:** Low 7 function displays the lower value between the seven inputs by transfering the lowest value to the output of the function .
**Input 1 to Input 7:** Inserted values.
**Output:** Displays the lowest value between the seven inputs.

---

## 30 *Low8* Maximum value 8

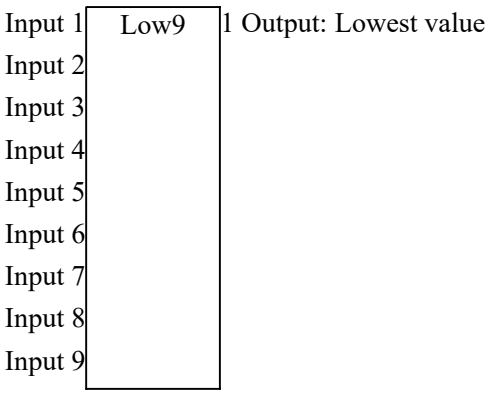



**Low 8 function:** Low 8 function displays the lower value between the eight inputs by transfering the lowest value to the output of the function .

**Input 1 to Input 8:** Inserted values.
**Output:** Displays the lowest value between the inputs.

---

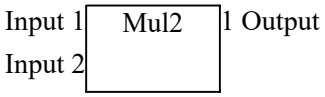
## 31 *Low9* Maximum value 9




**Low 9 function:** Low 9 function displays the lower value between the nine inputs by transfering the lowest value to the output of the function .
**Input 1 to Input 9:**Inserted values.
**Output:** Displays the lowest value between the inputs.

---

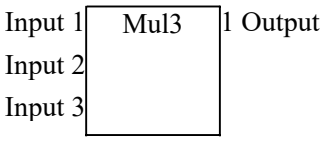
## 32 *Mul2* Multiply 2 values




**Mul 2:**The Output receives the result of the mathematical multiplication of the two Input values.
Output = Input1 * Input 2

---

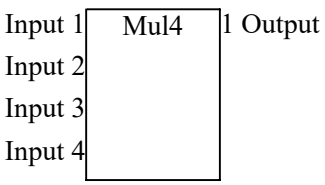
## 33 *Mul3* Multiply 3 values




**Mul 3:** The Output receives the result of mathematical multiplication of the three Input values.
Output = Input1 * Input 2 * Input 3

---

## 34 *Mul4* Multiply 4 values

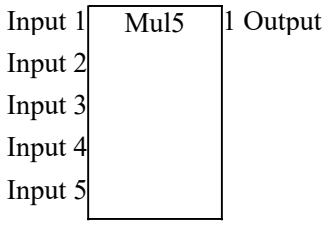



**Mul 4:**The Output receives the result of mathematical multiplication of the fout Input values.

Output = Input1 * Input 2 * Input 3 * Input 4

---

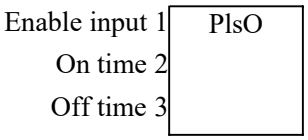
## 35 *Mul5* Multiply 5 values




**Mul 5:** The Output receives the result of mathematical multiplication of the five Input values.
Output = Input1 * Input 2 * Input 3 * Input 4 * Input 5

---

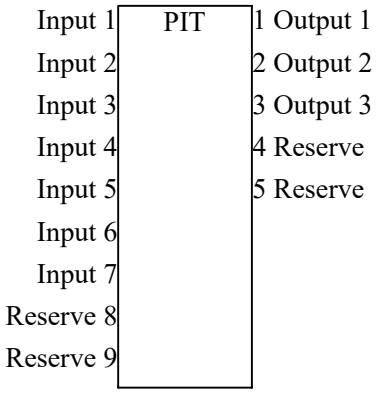
## 36 *PlsO* Pulse On D.Out




**Plso:** Plso function generates pulses on the relevant digital output (internal number of the function) as long as the enable (input 1) has "1". The time ON of the pulse and time OFF between pulses is defined in inputs 2 and 3.
Internal number: Relevant digital output.
Input 1: Enable.
Input 2: On time in seconds.
Input 3: OFF time in seconds.

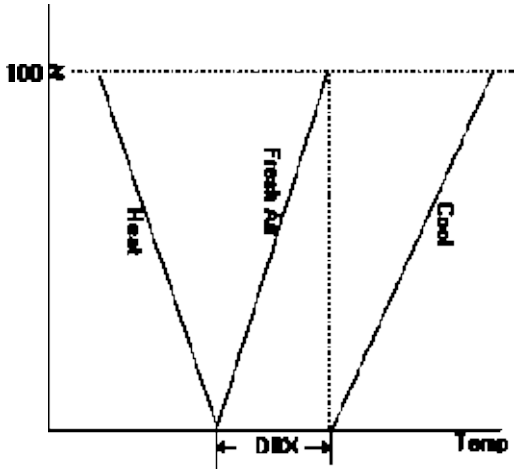
---

## 37 *PIT* PI Turbo



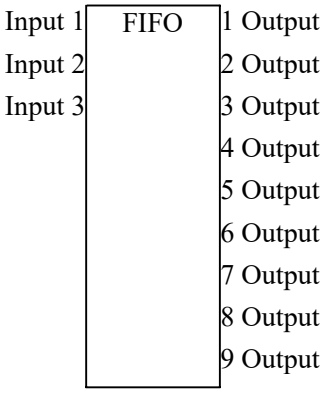

We recommend to use PIDT function instead (more advanced).
**PIT:** PIT function is used for PI control. .
Input 1: Measuared value (controlled value).
Input 2: Set Point .
Input 3: Freez Band.
Input 4: P proportional band.
Input 5: I time integral.
Input 6: Fresh air work range (economyzer - operating in the range between colling and heating, if not used please set to 0).
Input 7: Enable (1\0) .

Input 8: Operation mode: 1 = automatic, 2 = cooling only , 3 = heating only .
Input 9: Reserved
Output 1: Porportional Coolling command (0-100%).
Output 2: Proportional Heating command (0-100%).
Output 3: Proportional Fresh Air command (0-100%).
Output 4: reserved for debug mode.
Output 5: reserved for debug mode.




---

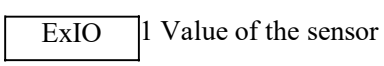
# 38 *FIFO* First In First Out




**FIFO:** This function managing a First In First Out line of outuput activations by activating the outputs of the function. For example: if input 1 receives value of 3 then ouputs 1,2 and 3 will be ON, if input 1 will change to 2 than ouput 1 will turn OFF while outputs 2 and 3 will remain ON, if input 1 will receive value 3 again then outputs 2 and 3 will remain ON in addition to Output 4 that will also turn ON.
Internal number: Number of the most applied number - how many outputs of the function are in use.
Input 1: The amount of active outputs.
Input 2: Output number that will be disabled and replaced by Output 9 instead (for example - fault back up).
Input 3: Binary representative of the amount of **inactive** outputs. For example: 5 =101 (in binary code), the inactive outputs will be 1 and 3 (for example - in case of fault in the activated devices).

---

# 39 *ExIO* External IO (CO)




**ExIO:** This function is dedicated to be used in CO monitoring systems, which are using Control Applications CO sensors only.The output of the functions is the value of the CO sensors. Internal number of the function is the

address of the CO sensor.

---

## 40 *NEG* Negative Number


**NEG:**The Output result is equal to the mathematical multiply of the Input by (-1).

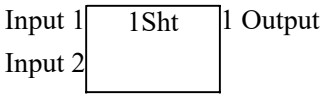
---

## 41 *NOP* No Operation


No operation

---

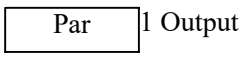
## 42 *1Sht* One Shot




**1Sht:** The function creates a pulse on its output (one shot) based on time in seconds as specified in input 2.
When input 1 change from 0 to 1 the output will change immediately to 1.
The output will change back to 0 just after the elapsed time as defined in input 2 or if input 1 changed back to 0 in the middle of the pulse.

---

## 43 *Par* Parameter




**Par:** Parameter (variable) - is used to insert free numerical data into the software which can be changed later on by communication (for example Set Point values, time delay values etc). Also can be used to create interlocks in different parts of the software when used in combination with function ParS (parameter store).

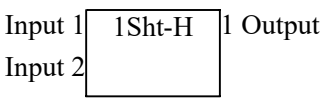
---

## 44 *ParS* Parameter Store


**Pars:** Parameter Store - is used to store a numerical data in one part of the program, so you could use it elsewhere in other location of the program by using Parameter function in the software.

---

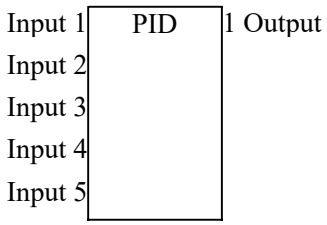
## 45 *1Sht-H* One Shot & Hold



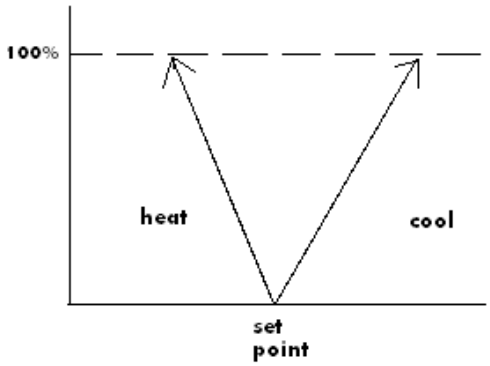
**1Sht-H:** The function creates a pulse on its output (one shot) based on time in seconds as specified in input 2..
When input 1 change from 0 to 1 the output will change immediately to 1 and will remain 1 according to time (in seconds) as defined in input 2.
The output of the function will remain 1 untill the end of the pulse time, even if Input 1 changes to 0 in the middle of the pulse .

---

# 46 *PID* PID logic control

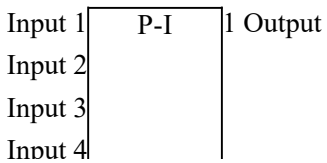



Old function for PI control - recommended to use PIDT instead
**Input 1:** Deviation from the desirable controlled value (Set Point)
**Input 2:** P - Proportional Band
**Input 3:** I - Reset time (seconds)
**Input 4:** D - Preset time (seconds)
**Input 5:** I - Integration limits - when the Input is not connected, the Output will be limited to +-75.
**output:** Correction signal.




---

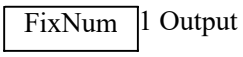
# 47 *P-I* P-I logic control



Old function for PI control - recommended to use PIDT instead
**Input 1:** Deviation from the desirable controlled value (Set Point)
**Input 2:** P - Proportional Band.
**Input 3:** I - Reset time (seconds).
**Input 4:** I - Integration limits - when the Input is not connected, the Output will be limited to +-75.
**Output:** Correction signal

---

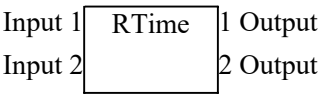
## 48 *FixNum* Fixed Number




This function enables to insert a fixed numerical value in any part to the software, this value is fixed and cannot be changed via communication.
**Output:** chosen value

---

## 49 *RTime* Run Time



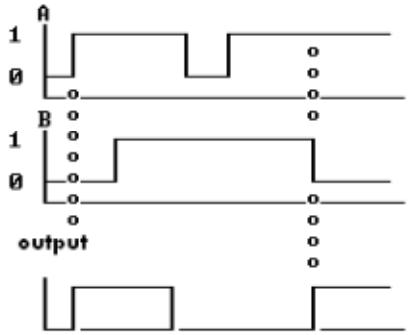

Run Time function is counting the elapsed time, the function will start counting at the moment it gets "1" in Input 1. Input 2 allows to reset the counter.
**Input 1:** enable
**Input 2:** reset
**Output 1:** elapsed time in seconds
**Output 2:** elapsed time in hours

---

## 50 *SRFF* SR. Flip Flop


Standard SR Flip Flop
Output will change to "1" when the First Input is "1" and the second Input is "0".
Output changes to "0" when the first Input is "0" and the second Input is "1".
In other cases the output will stay the same as previous.

| Input 1 | Input 2 | output |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 0 | 1 |



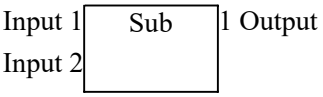

---

## 51 *EndM* End Modul


End Module: Allows to stop and diactivate part of the program.
**EndM:** When the function input gets "1" , it brakes the scanning of the software at the column in which the function is located, untill the function receives 0 again.

---

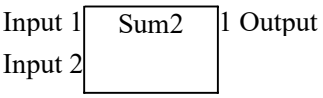
## 52 *Sub* Subscribe




Mathematic Subtraction
**Sub:**The output of the function presents to the mathematical subtraction of the two Input values.
**Input 1:** A
**Input 2:** B
**Output:** = A-B

---

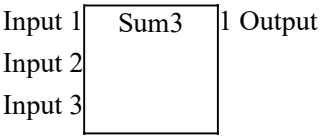
## 53 *Sum2* Summing 2 numbers




**Sum2:** The output equals to the sum of the two inputs.
**Input 1:** value A
**Input 1:** value B
**Output:** = A+B

---

## 54 *Sum3* Summing 3 numbers

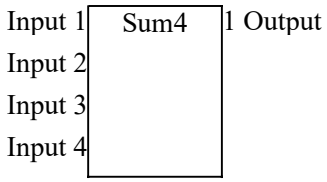



**Sum 3:** The output equals to the sum of the three inputs.

**Output** = **Input 1** + ... **Input 3**

---

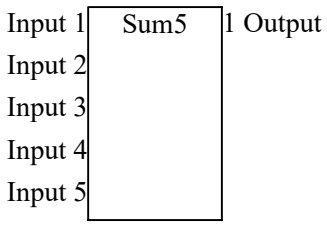
## 55 *Sum4* Summing 4 numbers




**Sum 4:** The output equals to the sum of the four inputs.
**Output** = **Input 1** + **Input 2** +... **Input 4**

---

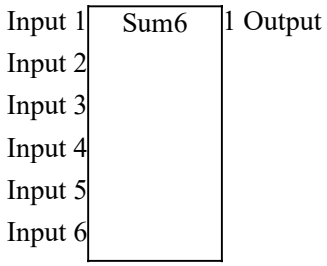
## 56 *Sum5* Summing 5 numbers




**Sum 5:** The output equals to the sum of the five inputs.
**Output** = **Input 1** + **Input 2** +... **Input 5**

---

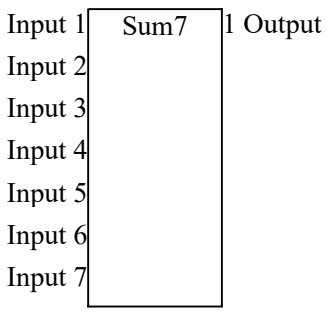
## 57 *Sum6* Summing 6 numbers




**Sum 6:** The output equals to the sum of the six inputs.
**Output** = **Input 1** + **Input 2** +... **Input 6**

---

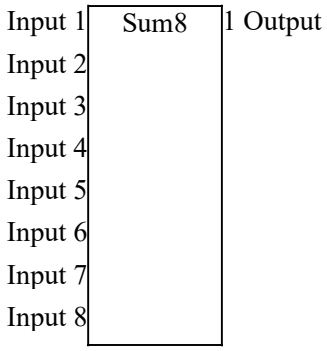
## 58 *Sum7* Summing 7 numbers



**Sum 7:** The output equals to the sum of the seven inputs.
**Output** = **Input 1** + **Input 2** +... **Input 7**

---

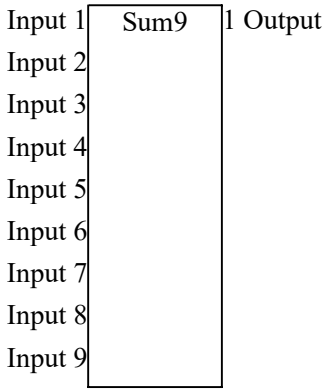
## 59 *Sum8* Summing 8 numbers




**Sum 8:** The output equals to the sum of the eight inputs.
**Output** = **Input 1** + **Input 2** +... **Input 8**

---

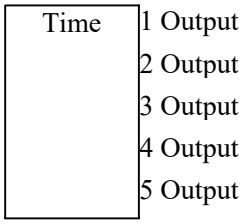
## 60 *Sum9* Summing 9 numbers




**Sum 9:** The output equals to the sum of the nine inputs.
**Output** = **Input 1** + **Input 2** +... **Input 9**

---

## 61 *Time* Time: Sec,Min,WDay,Day,Month

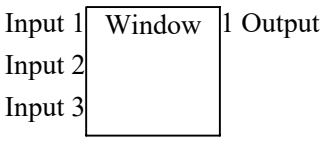



**Time:** The function shows the real time off the controller's clock, its week day and month day at the 5 outputs of the function..
**Output 1:** displays seconds 0-60
**Output 2:** displays minutes 0-1440
**Output 3:** displays day of the week 1-7

**Output 4:** displays day of the month 1-31
**Output 5:** displays months 1-12

---

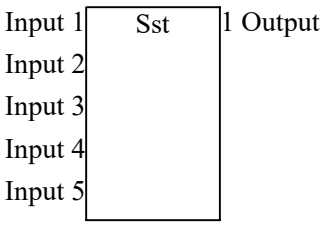
## 62 *Window* Window




**Window:** Output of the function changes to "1" when the value in Input 1 is between the values of two other Inputs.
In any other case the output is "0".

---

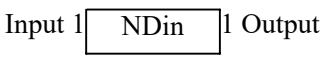
## 63 *Sst* Start Stop Time Program (1)




SST start stop time function can be linked to SCADA time schedule program
**Sst:** The output of the function will change it status to "1" if the first Input (enable) is "ON" and the current time is within the range of start and stop time as defined in other inputs of the function.
**Input 1:**Enable
**Input 2:**First start time
**Input 3:**First stop time
**Input 4:**Second start time
**Input 5:**Second stop time
**Important:**Time is measured in minutes from midnight, for example 13:10 p.m = 790.

---

## 64 *NDin* Not Digital Input

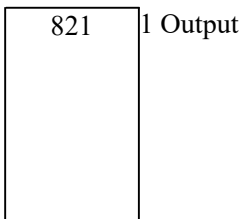


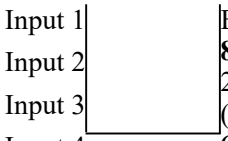

**NDin:** The output of the function, provides the opposite status of the relevant digital input of the controller.

| Input | Output |
|---|---|
| 0 | 1 |
| 1 | 0 |

---

## 65 *821* Collect 8 bits to byte

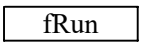





Eight to One binary pack
**821 :**The output of the function will change to a number between 0 and 255, that includes values of the Inputs according to the next formula (binaryc encoding).
Out = In[1] * 1 + In[2] * 2 + In[3] *4 + In[4] * 8 + In[5] * 16 + In[6] * 32 + In[7] * 63 + In[8] * 128

---

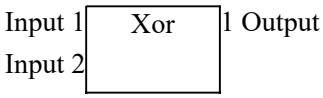
## 66 *fRun* Fast Run (NOP)




Fast Run - when used, the function dedicated to increase the repsonce time of Analog Inputs which are used as Digital Inputs.

---

## 67 *Xor* Logical XOR function




**Xor:** Standard XOR logic - if the two Inputs are different, the Output = "1"
In any other case the Output = "0"

| Input 1 | Input 2 | output |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

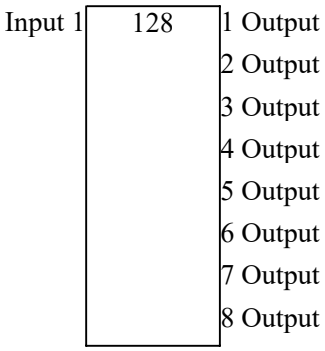
If Input values are real numbers, the function works with the values as if they are integers expressed as binary numbers.
For example:

| Input 1 | Input 2 | output |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 0 | 2 | 2 |
| 0 | 3 | 3 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| 1 | 2 | 3 |
| 1 | 3 | 2 |
| 2 | 0 | 2 |
| 2 | 1 | 3 |
| 2 | 2 | 0 |
| 2 | 3 | 1 |
| 3 | 0 | 3 |
| 3 | 1 | 2 |
| | | |

| | | |
|---|---|---|
| 3 | 2 | 1 |
| 3 | 3 | 0 |

---

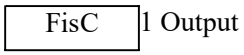
## 68 *128* Seperate Byte to 8 bits




One to Eight binary unpack
**128:** This function converts one Input Byte (0 - 255) to 8 Digital Outputs.

---

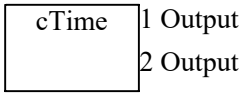
## 69 *FisC* First Cycle (NPF)




First Cycle is dedicated to do an operation on first cycle of the program run (for example right after power up of the controller)
**Fisc:** Output changes to "1" right after the first program running cycle in the controller (occures right after turning ON the controller and after sending the program), the output of the function changes back to 0, a cycle afterwards.

---

## 70 *cTime* Cycle Time




**cTime:** The function displays the current cycle time (scan time) of the program and the average cycle time.
Output 1 - current cycle time
Output 2 - average cycle time

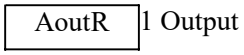
---

## 71 *SQRT* Square Root


**SQRT:** Output equals to mathematic square root of the Input.

---

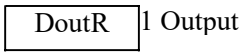
## 72 *AoutR* Read Analog Out



**AotR:** The function enables to see the command status of particular Analog output of the controller.

---

## 73 *DoutR* Read Digital Out




**DotR:**The function enables to see the command status of particular Digital Output of the controller.

---

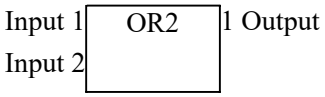
## 74 *NOT* Logical NOT function


Output = "1" , when the Input value = "0"
Output = "0" , when the Input value = "1"

| Input | Output |
|---|---|
| 0 | 1 |
| 1 | 0 |

---

## 75 *OR2* Logical OR 2 function

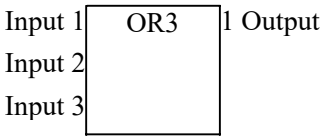



**OR2 :** The output equals "0" if all the Inputs are equal to "0" .
If at least one Inputs equal to "1" , the output will change to "1".

| Input 1 | Input 2 | Output |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

---

## 76 *OR3* Logical OR 3 function



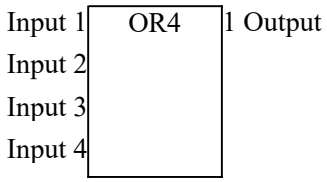

**OR3 :** The output equals "0" if all the Inputs are equal to "0" .
If at least one Input equal to "1" , the output will change to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |

| | | | |
|---|---|---|---|
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 |

---

# 77 *OR4* Logical OR 4 function

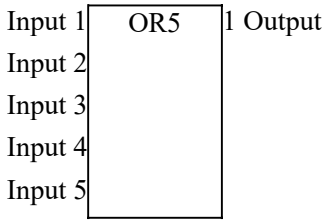



**OR4 :** The output equals "0" if all the Inputs are equal to "0" .
If at least one Input equal to "1" , the output will change to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 |

---

# 78 *OR5* Logical OR 5 function

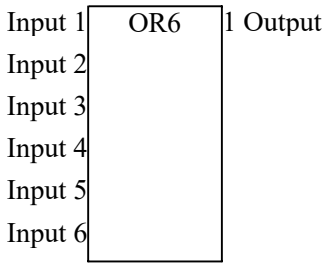



**OR5 :** The output equals "0" if all the Inputs are equal to "0" .
If at least one Input equal to "1" , the output will change to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 |

---

# 79 *OR6* Logical OR 6 function

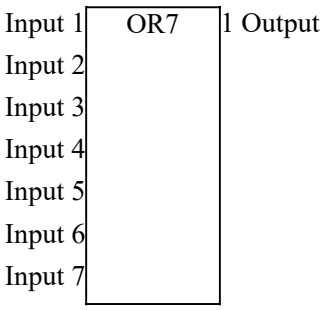


**OR6 :** The output equals "0" if all the Inputs are equal to "0" .
If at least one Input equal to "1" , the output will change to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 |

---

# 80 *OR7* Logical OR 7 function

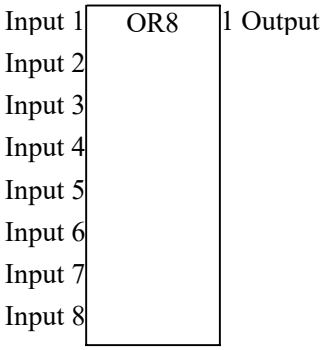



**OR7 :** The output equals "0" if all the Inputs are equal to "0" .
If at least one Input equal to "1" , the output will change to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 |

---

# 81 *OR8* Logical OR 8 function



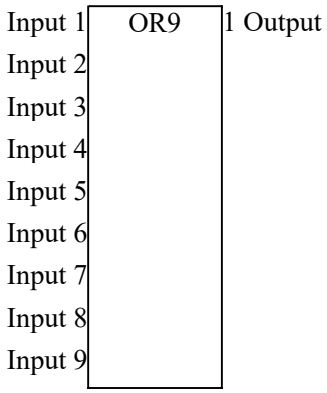

**OR8 :** The output equals "0" if all the Inputs are equal to "0" .
If at least one Input equal to "1" , the output will change to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| | | | |

| | | | |
|---|---|---|---|
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 |

---

## 82 *OR9* Logical OR 9 function

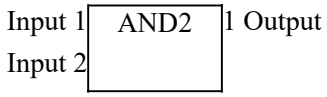



**OR9 :** The output equals "0" if all the Inputs are equal to "0" .
If at least one Input equal to "1" , the output will change to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 |

---

## 83 *AND2* Logical AND 2 function

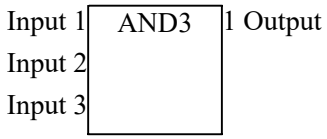



**AND2 :** The output equals "0" as long as at least one Input equals to "0" .
The output will change to "1", only if all of the Inputs are equal to "1".

| Input 1 | Input 2 | Output |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

---

## 84 *AND3* Logical AND 3 function



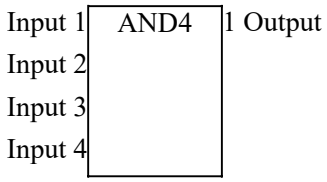

**AND3 :** The output equals "0" as long as at least one Input equals to "0" .

The output will change to "1", only if all of the Inputs are equal to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |

---

# 85 *AND4* Logical AND 4 function

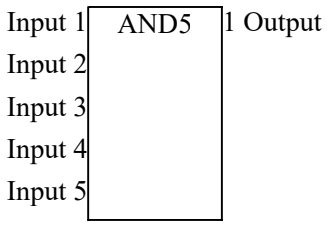



**AND4 :** The output equals "0" as long as at least one Input equals to "0" .
The output will change to "1", only if all of the Inputs are equal to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |

---

# 86 *AND5* Logical AND 5 function




**AND5 :** The output equals "0" as long as at least one Input equals to "0" .
The output will change to "1", only if all of the Inputs are equal to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |

---

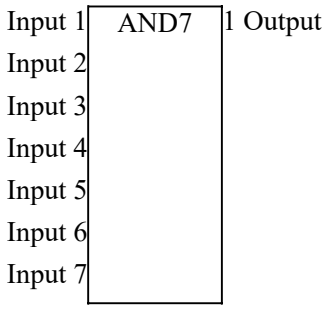
# 87 *AND6* Logical AND 6 function

**AND6 :** The output equals "0" as long as at least one Input equals to "0" .
The output will change to "1", only if all of the Inputs are equal to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |

---

# 88 *AND7* Logical AND 7 function

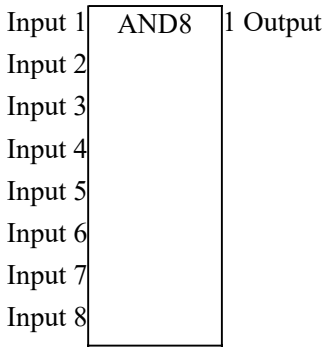



**AND7 :** The output equals "0" as long as at least one Input equals to "0" .
The output will change to "1", only if all of the Inputs are equal to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |

---

# 89 *AND8* Logical AND 8 function



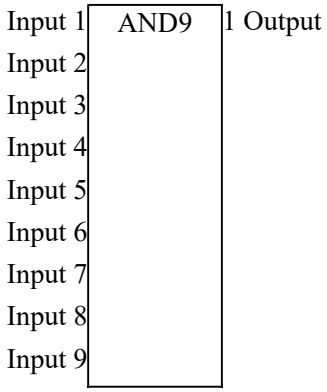

**AND8 :** The output equals "0" as long as at least one Input equals to "0" .
The output will change to "1", only if all of the Inputs are equal to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| | | | |

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |

---

# 90 *AND9* Logical AND 9 function

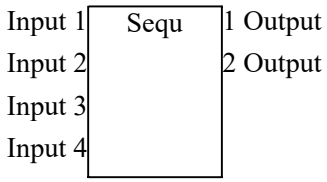



**AND9 :** The output equals "0" as long as at least one Input equals to "0" .
The output will change to "1", only if all of the Inputs are equal to "1".

| Input 1 | Input 2 | Input 3...9 | Output |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |

---

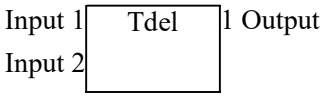
# 91 *Sequ* Sequance Controller




Sequance function allows to "move" from one value to another one based on predefined time interval (seconds).
Output 1 will change to 1 only when the function reached the second (target) value.
Output 2 has two modes: as long as input 1 (enable) of the function is "0" the Output 2 will always receive the minimum (start) value, once the function is enabled
Output 2 will receive real time value (moving between start value and end value).
**Input 1:** enable.
**Input 2:** start value.
**Input 3:** end value.
**Input 4:** time in seconds that will take for the start value (Input 2) will reach to end value (Input 3).

**ouput 1:** indecates whether the function reached its end value (changes to 1) also can be used as enable for the next sequ function.
**ouput 2:** minimum (start value) or real time value (depends on the enable input 1) also used as a start value for the next sequ function .

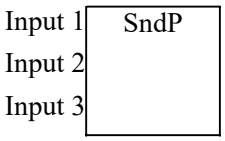
## 92 *Tdel* Time delay




**tDel :** When the first Input changes to "1" the Output will change to "1" as well after time delay in seconds as defined in Input 2 of the function.

---

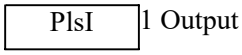
## 93 *SndP* Send Par




**SndP:** Send Parameter function is used to send parameter (data) to a parameter in another controller or in another file in the same controller.
The internal number of the function "num" (sndp - num) refers to the controller number (address) that will recieve the data.
**Input1:**The value to be sent.
**Input2:** Number of destination file in the target controller.
**Input3:** Number of destination parameter in the target controller that will receive the data.

---

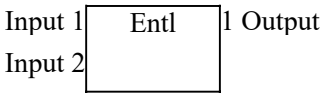
## 94 *PlsI* Pulse On Digital In




**PlsI:** The output of the function is dedicated to present fast (short) pulse input signals - works only with fast digital inputs of the controller.
**The internal number** is the number of the relevant Input of the controller on which the function applied on. This function is responding faster than regular Digital In function.

---

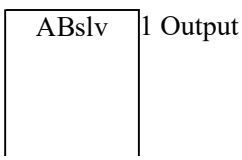
## 96 *Entl* Enthalphy - Energy




**Entl :** The Output of the function presents the calculation of the air Enthalpy, based on temperature and humidity.
**First Input** - Temperature
**Second Input** - Humidity.

---

## 97 *ABslv* AB Solver (Linear)

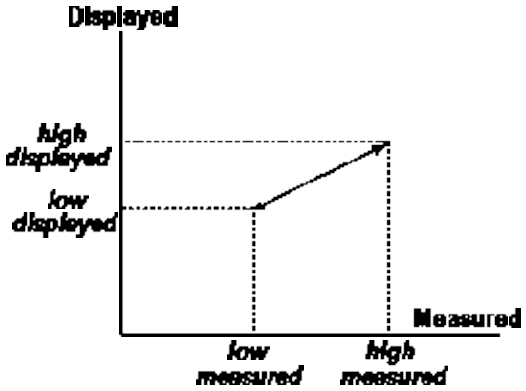


Input 1
Input 2
Input 3
Input 4
Input 5

**ABslv:** The function diplays the relative proportion (liniar) between the measured value and displayed value (cascade).

**Input 1:** Source value
**Input 2:** Low measured value
**Input 3:** High measured value
**Input 4:** Low displayed value
**Input 5:** High displayed value

**for example:** If Input1=5 , Input2=0, Input3=10, Input4=0 ,Input5=100 the output will be 50, if the first input will be 10 so the output will be 100.




---

## 98 *Pass* Pass Information

Input 1 | Pass | 1 Output

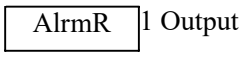
**Pass:** This function is used to pass information from one clolumn of the software to another one.

---

## 99 *COP* Month average

1 2 3 4 5 | COP | 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

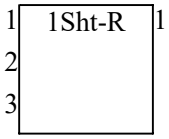
## 111 *AlrmR* Read Alarm




**AlrmR :** Alarm read function shows the status of relevant alarm in the controller.

---

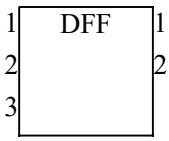
## 112 *1Sht-R* One Shot & Hold & Reset




**1Sht-R:** One shot and hold + reset: the function behaves as standard 1Sh&H function - creates a pulse on its output (one shot) based on time in seconds
as specified in input 2. When input 1 change from 0 to 1 the output will change immediately to 1 and will remain 1 according to time (in seconds) as defined in input 2.
The output of the function will remain 1 untill the end of the pulse time, even if Input 1 changes to 0 in the middle of the pulse .
However if during the pulse Input 3 will change from 0 to 1 the function output will change to 0.

---

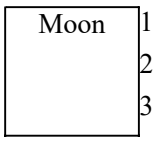
## 113 *DFF* Digital Flip Flop




**DFF :** Digital flip flop - when input 1 changes from "0" to "1" output1 will get the value from input 3.
Input 2:- reset, when the input changes to "1" the function will reset.
Output 2 will always display the opposite value from output 1.

---

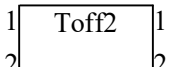
## 114 *Moon* Moon phase



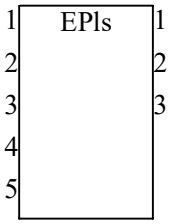

**Moon:** The function displays the current moon phase.
The value on output 1 starts from 1 and can reach up to 29, depends on the moon phase.
While there is a full moon, the numbers at the output will be approximately at the midle of the scale (between 14-13).
**Input 2-3 :** spear ouputs

---

## 115 *Toff2* Time Delay Off 2

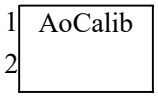


## 116 *EPls* Energy Pulse



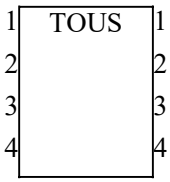

---

## 119 *AoCalib* Analog out Clibration input \r card:



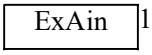

---

## 120 *TOUS* Tou Sigma (total)




---

## 121 *ExAin* IO-Card Analog input \r card:




**ExAin :** Provides the real time status of the relevant (internal number) external analog Input from analog extension card of the VeroPoint controller.

---

## 122 *ExAout* IO-Card Analog output \r card:


**ExAout :** External Analog Output control in the analog extension card of the VeroPoint controller.
The function controls over the relevant (internal number) analog output in the extension card.

---

## 123 *ExDin* IO-Card Digital input \r card:


**ExDin :** External Digital Input from extension card of the VeroPoint controller.
The function displays the real time status of the relevant (internal number) digital input from the extension card.

## 124 *ExNDin* IO-Card Not-Digital input \r card:


**ExNDin :** External Not Digital Input from extension card of the VeroPoint controller.
The function displays the real time status of the relevant (internal number) digital input with logic NOT (oposit status) from the extension card.

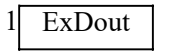
---

## 125 *ExRAout* IO-Card Read Analog output \r card:


**ExRAout:** External Analog Out Read from analog extension card of the VeroPoint controller.
The function displays the real time status of the relevant (internal number) analog output from the extension card.

---

## 126 *ExDout* IO-Card Digital output \r card:




**ExDout :** External Digital Output control in the extension card of the VeroPoint controller.
The function controls over the relevant (internal number) digital output in the extension card.

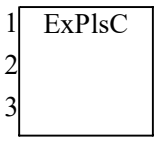
---

## 127 *ExRDout* IO-Card Read Digital output \r card:


**ExRDout:** External Read Digital Output control in the extension card of the VeroPoint controller.
The function shows the status of relevant (internal number) digital output command in the extension card.

---

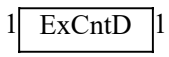
## 128 *ExPlsC* External Pulse out with cycle time \r Card:




**ExPlsC :** External Pulse Out function is applied on digital output of the extension card of the VeroPoint controller.
**Input 1:** Enable
**Input 2:** Pulse Cycle time (in seconds)
**Input 3:** ON time of the pulse in percents

---

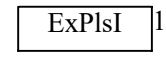
## 129 *ExCntD* External Coun D.In\r Card:



**Cntd:** External Pulse In counter function applied on fast digital inputs of extension card of the VeroPoint controller.
The function counts the pulses from relevant digital input (internal number of the function), the count result is on the output of the fucntion.
**Input -** Counter Reset
**output -** Amount of pulses.

---

## 130 *ExPlsI* External Pulse On Digital In \r Card:




**ExPlsI:**External Pulse In - monitoring pulse inputs of the extension card of VeroPoint controller (the function has better responce time than regular Din function and dedicated to detect short pulces).

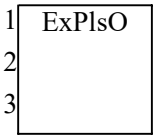
---

## 131 *ExHCD* External Hold Counter D-In\r Card:


**ExHCD:** External Hold of Counter D-In, used with extension cards of VeroPoint controller in combination with External Count Din function.
As long as the input of the function receives "1", the count din function **will ignore** pulses and will not count them.
NUM - Number of the relevant digital input that is used.

---

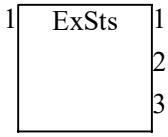
## 132 *ExPlsO* External Pulse On D.Out \r Card:




**ExPlsO:** External Pulse Out function, applied on Digital Outputs of VeroPoint extension cards.
The function creates pulses on relevant (internal number) digital ouput.
Input 1: Enable
Input 2: On time in seconds
Input 3: Off time in seconds

---

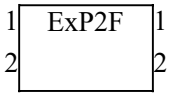
## 133 *ExSts* External IO status




**ExSts :**The function provides the status of the extension cards of VeroPoint controller.
**Input:** Number of the extension card to be checked.
**output1:** The status of the relevant card: "0"-no card, "1" - digital card , "2"- analog card .
**output2:** Displays the IO configuration.
**output3:** Displays the comminucation status with the card:"0" - no comminucation , "1"- good comminucation , "2"- comminucation error.

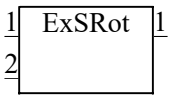
---

## 134 *ExP2F* Ex - Pulse to flow \r Cart1:



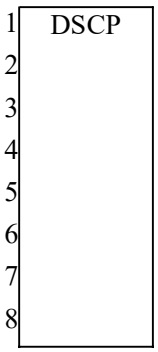

---

## 136 *ExSRot* Step Relay Output \r Card:



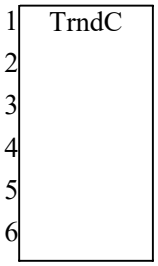

---

## 137 *DSCP* Define Screen Parameter



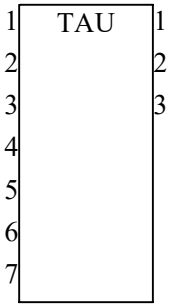

---

## 138 *TrndC* Internal Trend Control



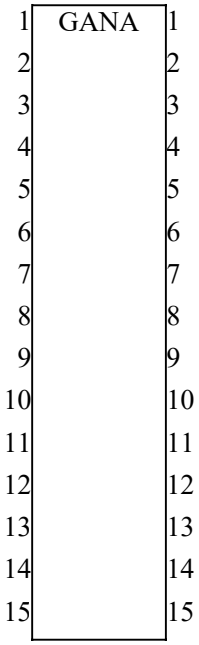

---

## 139 *TAU* Time of use split



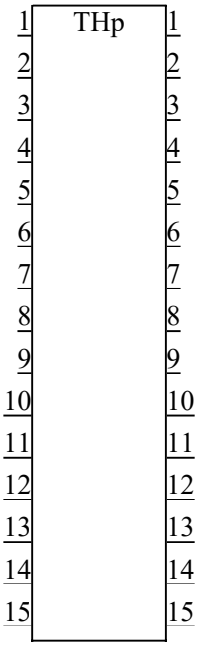

---

## 140 *GANA* GANA Panel



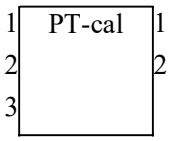

---

## 141 *THp* Temp & Humidity pannel



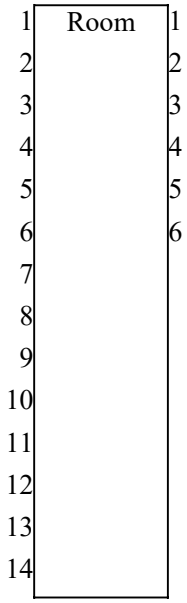

---

## 142 *PT-cal* pt-calibrate




---

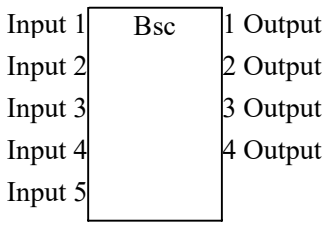
# 150 *Room* Shabat / Holiday status




**Room:** Used for Shabat mode (Jewesh holiday mode).
**Input1:** Enable.
**Input2:** Days of week in a binary number, if equals "1" it's saturday or holiday.
**Input3:** Start time of shabat (friday evening).
**Input4:** End time of shabat.
**Input5:** Start time for light .
**Input6:** End time for light.
**Input7:** Second start for light.
**Input8:** Second end time for light.
**Input9:** Start time for the Air conditioner.
**Input10:**End time for the Air conditioner.
**Input11:**Second Start time for the Air conditioner.
**Input12:**Second End time for the Air conditioner.
**Input13:**sper Input.
**Input14:**sper Input.
**output1:**Enable (Input 1) AND shabat mode.
**output2:**shabat mode.
**output3:**Friday mode.
**output4:**Light output.
**output5:**Air conditioner output.
**output6:**sper output.

---

# 151 *Bsc* Binary Step Controller

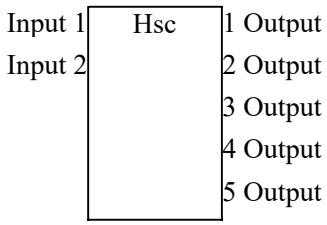



**Bsc:** Binaric outputs command with time delay ON and OFF.
For example when the flow switch input changes to "1" (enable), the heating elements (function outputs) will operate in a binary step sequence.
**Input1:** Demmand value 0 - 100 % .

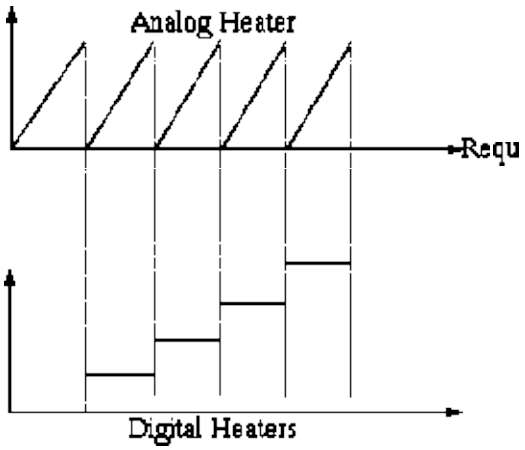
**Input2:** Number of heating elements.
**Input3:** Enable .
**Input4:** Delay On (seconds).
**Input5:** Delay Off (seconds).
The active outputs amount is defined in input 2.
**for example:**If input2 =3, so only three of the outputs will be used.

---

# 152 *Hsc* Heating Step Controller

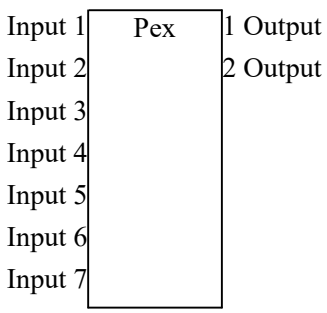



**Hsc heating step controller with proportional command and digital steps :** Once demand (input 1) starts to rise, the analog heating element command (output 1)
will start rising from 0% untill it reaches its maximum 100% level, if demand is still rising then first digital command output will be acitvated (output 2)
while the analog command will start rising again from 0%, if still demand exists then another digital command output (output 3) will be acitvated while the
analog command again will drop to 0% and start rising again and so on depends on amount of heating elements (input 2).
**Input1** - demand in %.
**Input2** - number of heating steps to use.
**Output1:** Proportional analog output command to be connected to an analog heating element.
**Output 2 - 5:** Digital outputs command




---

# 153 *Pex* Pump Exchange

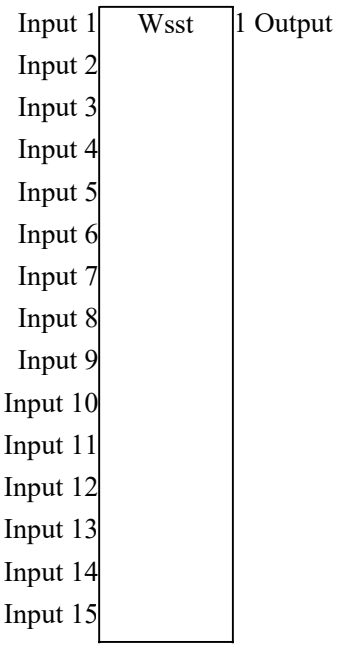


**Pex:** This function is used for an exchanging between two pumps.
**Input 1:** status of the first pump.
**Input 2:** status of the second pump.
**Input 3:** select what pump will work.
**Input 4:** day of mounth.
**Input 5:** day of week.
**Input 6:** time to exchange ( in hours).
**Input 7:**enable.

**output1:** pump 1.
**output2:** pump 2.
**output3: output4:**

---

## 154 *Wsst* Weekly Start/Stop




**Wsst :** Input 1 - Enable,
The even Inputs are the start time for every day of the week starting from Input 2 (Sunday),
The odd Inputs (starting with 3) are stop time for every day of the week.
Output of the function will change to "1" as long as first Input is "1" and the controller time is within the range of start and stop time of the relevant day.
**The time values are in minutes starting from midnight.**

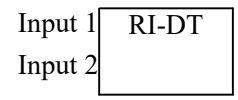
---

## 155 *RI-DN* Remote IO D.In


**RI-DN :** For compatibility only

---

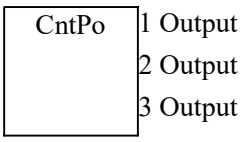
## 156 *RI-DT* Remote IO D.Out



**RI-DN :** for compatibility only

---

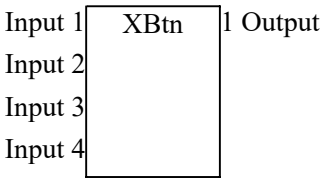
## 157 *CntPo* Count power on




**CntPo:**This function provides general information such as:
**Output 1:** displays the time since last power on of the controller.
**Output 2:**displays the number of times the controller was restarted.
**Output 3:** displays the number of times a program was sent to the controller.

---

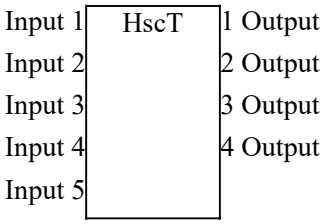
## 159 *XBtn* Special Button




**XBtn:** External button: dedicated for light acitvation from pushbuttons inside and outised of time schedule
**Input 1** - input from time schedule (from SST function), when the input is 1 the output gets 1 as well.
**Input 2** - pushbutton input, each pulse in this input changes the output status (during time schedule - input 1), also depends on time in input 4.
**Input 3**- number of parameter that can be used as a pushbutton for the function or to be affected by its output.
**Input 4** - time in minutes to stay activated after a pushbutton was pushed.

---

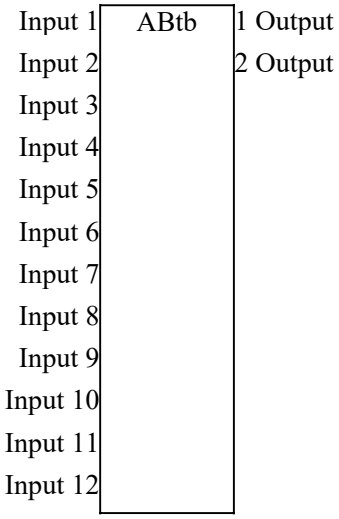
## 160 *HscT* Heat Step Con+Timer




**HscT:** HscT heating step controller with four digital steps with time delay on and off : Once demand (input 1) starts to rise, after time delay as specified
in input 3 the first output will switch ON, if demand is still rising then after timed delay next output command will be acitvated , if still demand exists then after timed delay
another digital command output will be acitvated and so on depends on specified amount of heating elements (input 4).
If the demand (input 1) will start decreasing the outputs of the function will be swtiched off one by one after delay off as specified in input 3. **Input 1** - desirable value.
**Input 2** - delay on (seconds).
**Input 3** - delay off (seconds).
**Input 4** - number of heating steps to use.
**Input 5** - enable.
**Output 1 - 4:** Digital outputs.

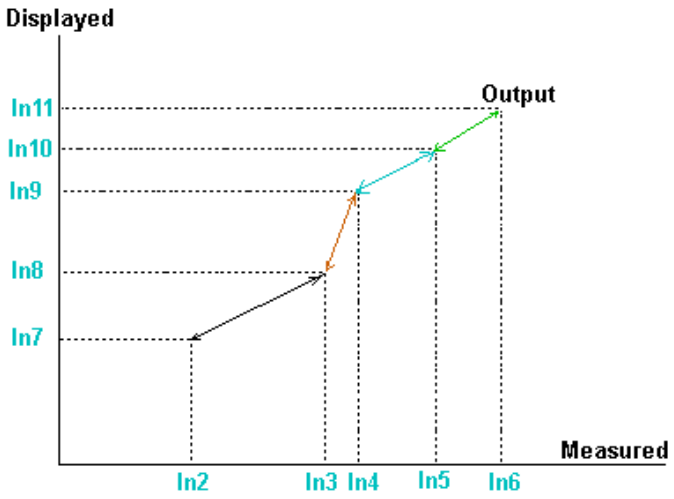
---

# 161 *ABtb* AB Table



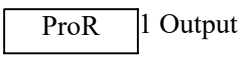

**ABtb:** The function diplays the relative proportion (liniar) between the measured value and displayed value (cascade).
An improved ABSOLVER with 4 ranges of values.
**Input 1:**Value to convert.
**Input 2:**Lowest value to measuring range 1.
**Input 3:**Highest value to measuring range 1 and Lowest value to measuring range 2.
**Input 4:**Highest value to measuring range 2 and Lowest value to measuring range 3.
**Input 5:**Highest value to measuring range 3 and Lowest value to measuring range 4.
**Input 6:**Highest value to measuring range 4.
**Input 7- 11:**Those Inputs are related to Inputs 2 up to 6 for Display values (for example input 7 is lowest displayed value for range 1).
**Input 12:**Defines how many active ranges there are.

**output1:**value for display.
**output2:** displays in what section the function is now.




---

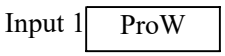
# 162 *ProR* Protocol Reg Read



**ProR:** The function is used to read data from the regsiters of the slave device as defined in the Modbus table and use it in the program . The internal number of the function is the row number in the Modbus table.

---

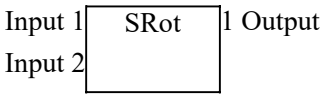
## 163 *ProW* Protocol Reg Write




**ProW :** The function is used to write data to the regsiters of the slave device as defined in the Modbus table . The internal number of the function is the row number in the Modbus table.

---

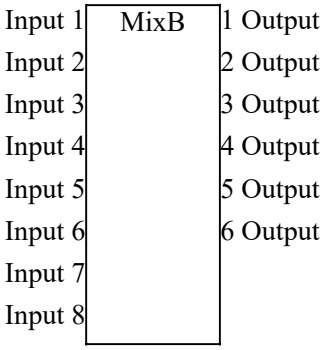
## 164 *SRot* Step Relay Output




**SRot:** The function activates a step relay commands on the output of the controller.
**Input 1:** Desirable state (ON \ OFF, for example from time schedule).
**Input 2:** Feedback .
**Output:** Pulse generation (for indication only - no need to connect to digital out, the function activates the digital output by itself) .
The function activates a pulse output command on relevant (internal number) digital output. It will generate pulses until the feedback (input 2) will reach
to the desirable state (input 1).

---

## 165 *MixB* Mix Box

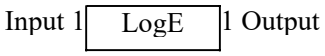



**MixB:**This function is used to control the temperature in VAV MixBox, based on minimum air flow and using cooling and heating commands .
**Input 1:**Temperature Input.
**Input 2:** Air flow input.
**Input 3:** Set point.
**Input 4:** Dead band 1.
**Input 5:** Dead band 2.
**Input 6:** P - proportional band.
**Input 7:** I- integration time (in seconds).
**Input 8:** Minimum air flow.
**Output:** Cool valve output.
**output2:** Heat valve output.
**output3:** Internal debuging.
**output4:** Internal debuging.
**output5:** Internal debuging.

**output6:** Internal debuging.

---

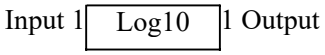
## 166 *LogE* Log Base E




**LogE:**Natural logarithm - changes the output to log base E ( ln ) of the Input.

---

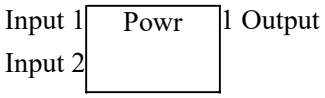
## 167 *Log10* Log Base 10




**Log10:**Changes the output to Log base 10 of the Input.

---

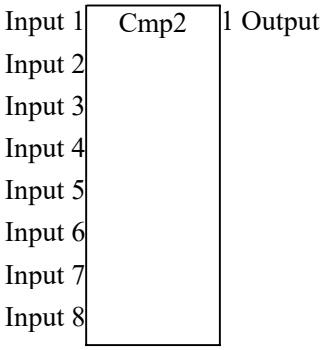
## 168 *Powr* Power (Exponent)




**Powr:** The output will receive the result of Mathematic power (exponent) of value in Input 1 with the value in Input 2.

---

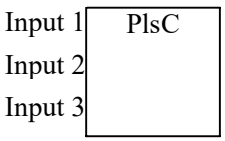
## 169 *Cmp2* Compare 2 Inputs




**Input Comparison -**If there are at least two equal Inputs, the Output of the function will change to "1", otherwise Output will remain "0".
NUM - defines how many Inputs to compare.

---

## 174 *PlsC* Pulse out with cycle time

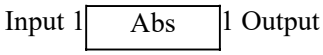



**PlsC:** The function produces pulses on the relevant digital output (internal number).

**Input 1:** Enable.
**Input 2:** Cycle time in seconds.
**Input 3:** Percentage of the ON time of the pulse.

---

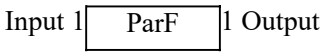
## 175 *Abs* Absolute value




**Abs:** Output changes to absolute value of the Input.

---

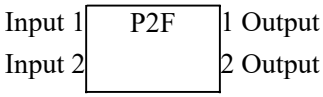
## 176 *ParF* Parameter With File




**ParF:** The function is pulling data from a parameter used in another file of the same UWP.
**Input:** - number of the file to pull the data from.
**Output:** - the received value.
Internal number - is the target parameter number to pull from.

---

## 177 *P2F* Pulse to flow




**P2F :** The function translates the pulses to value (for example each pulse equals to 10 litrs of water).
The function is applied on relevant digital input (internal number ).
**Input 1:** The value of each pulse.
**Input 2:** Reset.
**Output 1:** Total accumulated value.
**Output 2:** Total accumulated value in the last 30 seconds.

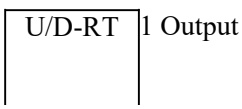
---

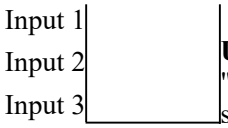
## 178 *SigI* Sigma input


**SigI:**The function is running an average sampling and storing the sampled data.
**Input 1:** Data to sample.
**Input 2:** Sampling frequency in seconds.
**Input 3:** Reset the stored data.
**Output:** Stored data.

---

## 179 *U/D-RT* Up and Down RTime



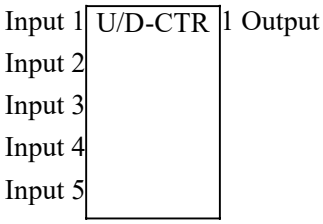



**U/D-RTup down read time:** Counting for how long was the "UP" or "Down" pushbutton pressed and the output receives the result in seconds.

**Input 1:** UP pushbutton status .
**Input 2:** Down pushbutton status .
**Input 3:** Reset .
**Input 4:** Minimum output value.
**Input 5:** Maximum output value.
**Output:** The calculated time in seconds (differnce between the 2 inputs).

---

# 180 *U/D-CTR* Up and Down Counter




**U/D-CTR :** Counting the pulses for the "UP" or "Down" inputs and the output receives the result (delta in between of the 2 inputs).
**Input 1:** Pulse up.
**Input 2:** Pulse down.
**Input 3:** Reset .
**Input 4:** Minimum output value.
**Input 5:** Maximum output value.
**Output:** The total number of pulses (Delta between the 2 inputs).

---

# 181 *SIN* Sinus (Rad)


**SIN:** Changes the output to mathematic sinus of the Input.
The output will be in radians.

---

# 182 *COS* Cosinus (Rad)


**COS:**Changes the output to mathematic cosinus of the Input.
The output will be in radians.

---

# 183 *TAN* Tangans (Rad)


**TAN:**Changes the output to mathematic tangens of the Input.
The output will be in radians.

---

## 184 *ASIN* Arc Sinus (Rad)


**ASIN:**Changes the output to mathematic ARCSIN (the opposite function of sin) of the Input.

---

## 185 *ACOS* Arc Cosinus (Rad)


**ACOS:**Changes the output to mathematic ARCOS (the opposite function of cos) of the Input.

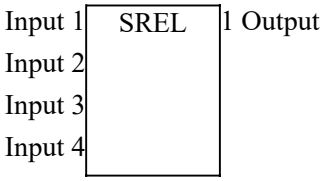
---

## 186 *ATAN* Arc Tangans (Rad)


**ATAN :**Changes the output to mathematic ARCTAN (the opposite function of tan) of the Input.

---

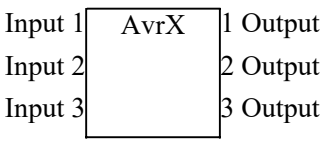
## 187 *SREL* Step Relay




**SREL :** Step relay
The function will apply a step relay commands (short pulses) on its output, which can be connected to a Digital Out. Once receiving "1"
in inputs 1 or 2 , the function will perform a short pulse on its output to turn ON, ones the input will change back to "0" the function will create another pulse to turn OFF.
If the funtion receives a value of 1 in input 3 (external button), the function will create a pulse on its output to turn ON and after a time specified in input 4 will
create another pulse on its output to turn OFF.
**Input 1:** First Start command (from time schedule).
**Input 2:** Second Start command (from time schedule) .
**Input 3:** External button.
**Input 4:** Delay time in minutes for activation after the external button was pushed.
**Output:** Gets value of "1' if input 1 or 2 is ON ,or if input 3 is ON after the time delay.

---

## 188 *AvrX* Average

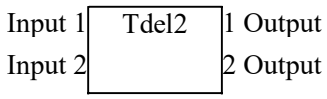



**AvrX :** The function is making an average of the input 1 every N seconds.

**Input 1:** The value..
**Input 2:** N - seconds..
**Input 3:** Reset.
**Output 1:** Every N seconds gets the average of the value.
**Output 2:** Every second gets the average for last N seconds.
**Output 3:** Shows how much N time elapsed.

---

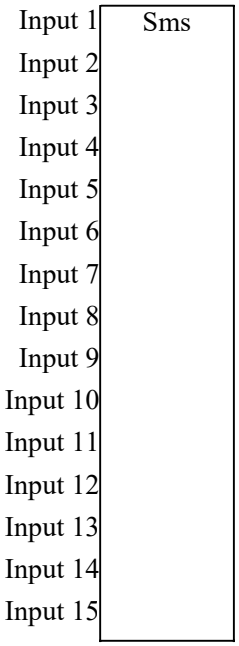
## 189 *Tdel2* Time Delay 2




**Tdel2:** When the first Input changes to "1", output 1 will change to "1" after time delay (seconds) as defined in Input 2.
Output 2 shows the elapsed time of the delay.

---

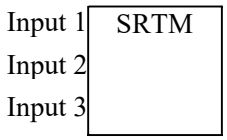
## 190 *Sms* Send Sms




**Sms:** Used for sending SMS messages from a controller (using GSM modem).
**Inputs 1- 12:** the number of alarms to be sent.
**Input 13:** zip code .
**Input 14 :** phone number to send to.
**Input 15 :** enable function.

---

## 191 *SRTM* Step Relay Time

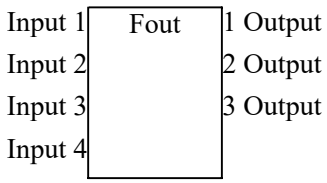



This function is dedicated to be used with the function **SRot (step relay control)** , it allows to define the next

settings:
**Input 1** - The size of the pulse in seconds.
**Input 2** - Time to wait for the feedback after the pulse , before trying another pulse .
**Input 3** - Number of retries to reach the desirable status.

---

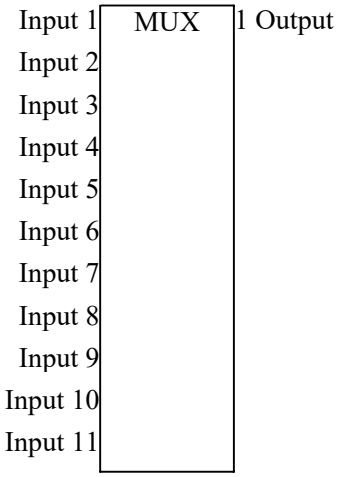
## 192 *Fout* Float out




**Fout :** This function is used to maintain a set point by commanding a Float control .
**Input 1:** The demand value 0-100 %.
**Input 2:** Time in seconds for the engine stroke from minimum to maximum position (from close to open)
**Input 3:** Manual reset.
**Input 4:** Time in minutes for daily reset.
**Output 1:** Closing command.
**Output 2:** Opening command.
**Output 3:** Location (calculated).

---

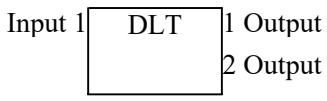
## 193 *MUX* Multiplexer 10-1




**MUX11:**The output will display value of the input number that was inserted to to the input no. 11.

---

## 194 *DLT* Delta




**DLT:** Delivers to the outputs the delta between readings of the input per one cycle of the program.
**Output 1:** Gets the positive and the negative delta value.
**Output 2 :** Gets only the positive delta value.

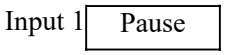
---

## 195 *HCD* Hold Counter D-In

**HCD:** Used with function CNTD , when the input equal "1", the count stops.
NUM - Number of the digital input on which applied.

---

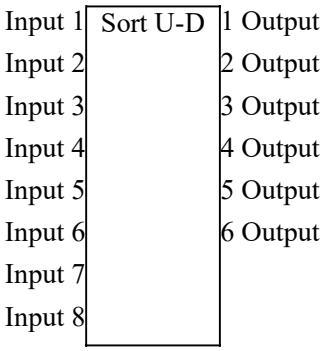
## 196 *Pause* Pause the program




**Pause:** Pause the program for the amount of seconds as specified on the input of the function.

---

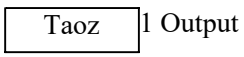
## 197 *Sort U-D* Sort Up-Down




**Sort U-D 6:** The function sorts the inputs from high to low values and from low to high values.
**Inputs 1-6:** The values to sort.
**Input 7:** If equal "0" the function will sort from high to low, if equal "1" the function will sort from low to high.
**Input 8:** The function will sort upon change in this input, from "1" to "0" or from "0" to "1".
**outputs1-6:** The result of the sorting process, the outputs will display the numbers of the inputs.

---

## 198 *Taoz* Electricity Rate




**for Israeli use only - shows the current active electrical rate status 0 = Low rate, 1 = Medium rate, 2=High rate**

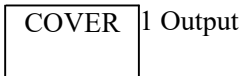
---

## 199 *Toff* Time Delay Off


**Toff :** Delay OFF - once the input 1 changes from "1" to "0", the output will change to 0 as well only after the delay value in seconds as specified in input 2.

---

## 201 *COVER* Change Over - Select

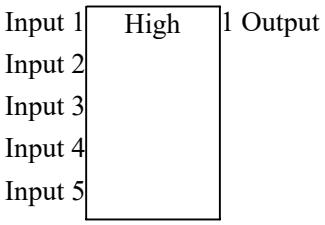



**COVER - change over:** Transfers the values from the inputs to the output, according to the status of the control input 3.
If the control input ( input 3) is set to "0" , then the output will receive the values from inputs "1" .
If the control input ( input 3) is set on "1", then the output will receive the value from inputs "2" .

| Input 3 | Output 1 |
|---|---|
| 0 | Input 1 |
| 1 | Input 2 |

---

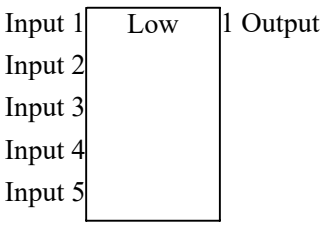
## 202 *High* MAX Select




**High:**This function enables you to choose a high function from the list.

---

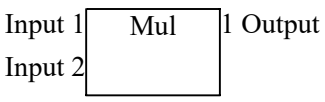
## 203 *Low* Minimum Select




**Low:**This function enables you to choose a low function from the list.

---

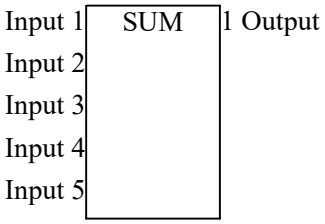
## 204 *Mul* Multiply Select




**Mul :**This function enables you to choose a mul function from the list.

---

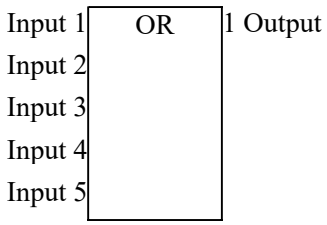
## 205 *SUM* Summery Select



**SUM :**This function enables you to choose a sum function from the list.

---

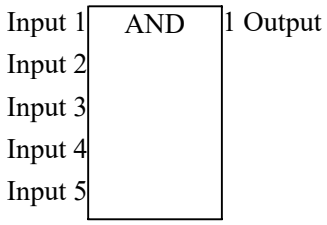
## 206 *OR* Or - Logic , Select




**OR :**This function enables you to choose an OR function from the list.

---

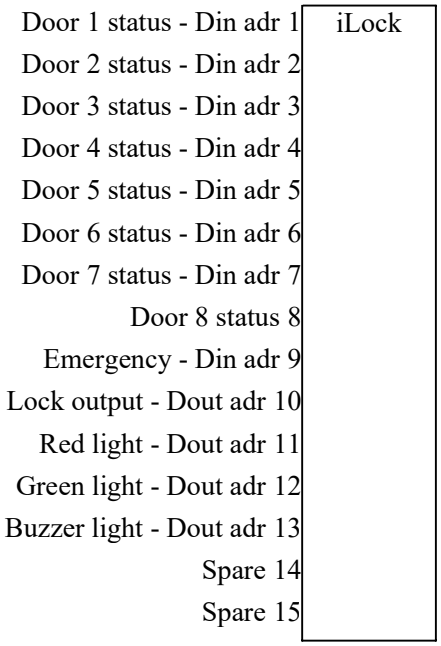
## 207 *AND* AND - Logic , Select




**AND :**This function enables you to choose an and function from the list.

---

## 208 *iLock* inter lock



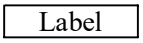

## Veropoint only!

Interlock - manage doors interlock.
wire up address (use fix numbers only - Don't use parameters!)
**adr - address of point calculated like this: card_number * 100 + point_number**

unwired leg - not in use address.
If any digital input (on lags 1-8) is opened, then door is locked. and red light is on.
If door is unlocked, then green light is on.
leg 9 - emergency switch (enable all doors) - all doors are enabled, and Buzzer is on.

---

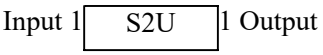
## 209 *Label* Label




**Label :** Allows to insert free text at any place in the software.
The internal number of the functions defines how lond will be the label, in order to insert text click with right mouse button on the left corner of the label and select properties.

---

## 210 *S2U* Signed to UnSigned Int




**S2U:** Turns Signed values to Unsigned.

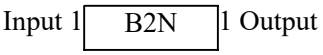
---

## 211 *U2S* UnSigned int to signed int


**U2S:**Turns Unsigned values to Signed.

---

## 212 *B2N* BCD to number




**B2N:** Turns BCD format value to decimal.

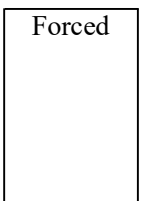
---

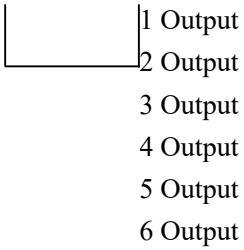
## 213 *PRWR* Protocol Reg Read Write


**PRWR:** The function is used to read and write data at the regsiters of the slave device as defined in the Modbus table and use it in the program . The internal number of the function is the row number in the Modbus table.

---

## 214 *Forced* Forced IO flags

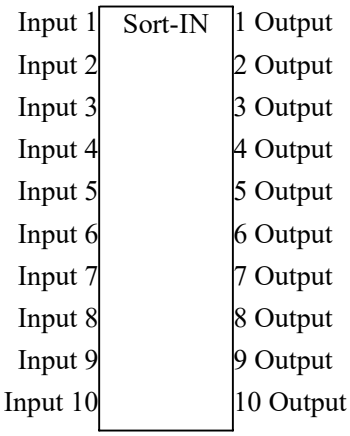





**Forced :** The functions checks whether there is forced IO in the controller.
**Output 1:**- Get value of 1 whether Analog In forced.
**Output 2 :**- Get value of 1 whether Analog Out forced.
**Output 3:**- Get value from 1-16 for the number of Analog In that is forced.
**Output 4:** - Get value from 17-32 for the number of Analog In that is forced.
**Output 5:**- Get value from 1-16 for the number of Analog Out that is forced.
**Output 6:** - Get value from 17-32 for the number of Analog Out that is forced.

---

## 215 *Sort-IN* Sort input low to high




**Sort-IN**:This function sorts the inputs from low to high values.
NUM- The amount inputs to sort.
Inputs1-10: The values to sort.
Outputs1-10: The result of the sorting process.

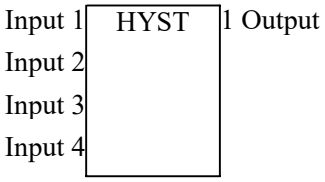
---

## 216 *RAND* Randomize number


**RAND:** At each rising of input no. 1 from 0 to 1, the output will receive a random number from 0 - N .
**Input 2:** Specify the high limit up to which random numbers can be received (N). For example if Input 2 = 10, the random numbers will be between 0 to 10.
If the function input remains with constant value such as 0 or 1 the function will be disabled.

---

## 217 *HYST* Hysteresis Time

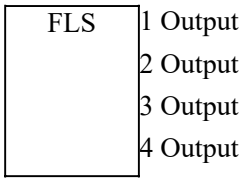



**HYST:**The function behaves as standard HYS function, with aditional input no. 4 that defines the delay time in

seconds before changing the output from 0 to 1.
If Input 1 is higher or equal to Input 2, then the Output will change to "1".
If Input 1 is lower than Input 2 minus Input 3, the Output changes to "0".

---

## 218 *FLS* Flood sensor




**FLS:** Flood sensor: The function is dedicated to read the value from the water flood sensor, the internal number of the function is the relevant analog input,
according to its values the function will provid the following data:
**Input:** An analog input of the sensor.
**Output 1:** will receive 1 when the input is in between 6-10 volt - "no flood status".
**Output 2:** will receive 1 when the input is in between 2.2-5.5 volt -" flood alarm".
**Output 3:** will receive 1 when the input ireceives any other voltage - "fault in the flood sensor device".
**Output 4:** The value .

---

## 219 *DoutRT* Read digital out time (SB)


**DoutRT :** Displays a digital output status, in case that it was triggered ON by internal time software of a Super Brain or VeroPoint controller.

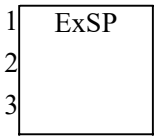
---

## 220 *SST-N* SB SST Status


**SST-N:** The status of the SuperBrain or VeroPoint time program 1 or 0, according to the operation hours.
The output of the function will change to 1 as long as the real time of the control is in the range of time schedule as defined in SuperBrain or VeroPoint build in time schedules.

---

## 221 *ExSP* External setpoint

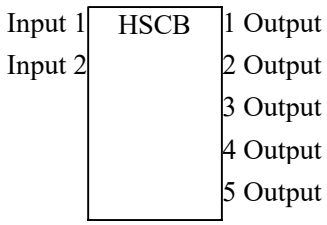



**ExSP :** External set point to read data from external potentiometer - used only in SuperBrain and VeroPoint controllers.
Transfers a value from Analog Input ((as specified in input 1 of the function) to target parameter (as specified in input 2 of the function).
Inputs:
**Input 1:** The number of the analog input used for the remote set point.
**Input 2:** The number of the parameter used to receive the value from the analog input.

**Input 3:** File number of the parameter that receives the data.
If the value in analog in is 9999 the parameter will be free to use and the function is deactivated.

---

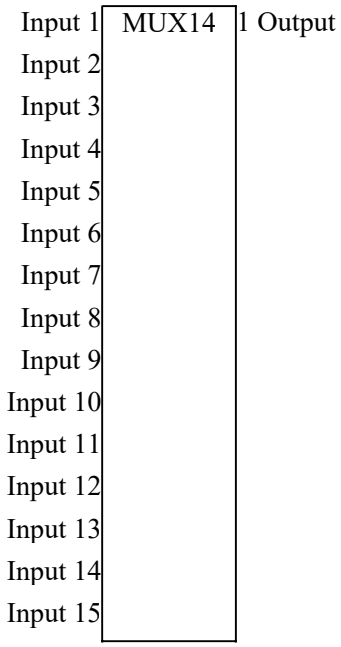
## 222 *HSCB* Heating Step Controller Binary




**HSCB:** HSCB heating step controller with proportional command and binaric digital steps : Once demand (input 1) starts to rise, the analog heating element command (output 1)
will start rising from 0% untill it reaches its maximum 100% level, if demand is still rising then a digital command output according to binaric sequence will be swithced ON
while the analog command will start rising again from 0%, if still demand exists then another digital command output according to binaric sequence will be swithced ON
analog command again will drop to 0% and start rising again and so on depends on amount of heating elements (input 2).
**Input1** - demand in %.
**Input2** - number of heating steps to use.
**Output1:** Proportional analog output command to be connected to an analog heating element.
**Output 2 - 5:** Digital outputs command

---

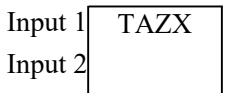
## 223 *MUX14* MultiPlexer 14-1

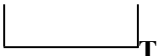



**MUX14:** The output will display value of the input number that was inserted to the input no. 15.

---

## 224 *TAZX* Tau



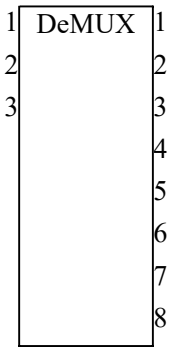


**TAZX :** Stores the value according to TOU (time of usage).

**Input 1:** Low.
**Input 2:** Medium.
**Input 3:** High.

---

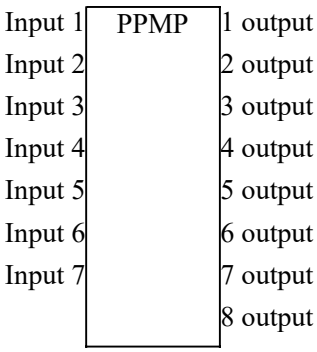
## 225 *DeMUX* DeMultiPlexer




**DeMUX :**
**Input 1 :** Inserted value.
**Input 2 :** The output that will get the inserted value from input 1.
**Input 3 :** "1"- freez mode, "0" - resets the outputs that not in use.

---

## 226 *PPMP* Presure Pumps

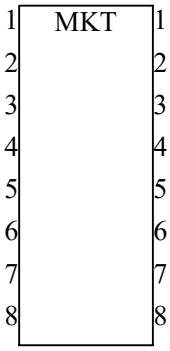



**PPMP :** Water pressure pumps control - analog output commands.
Speed control of water pressure pumps - according to the demand the speed of the first pump will rise, once the demand for additional pump appears
the new output will rise at the same time the first output speed control will decrease its value and will rise again so the speed of all of the pumps will rise together to the same value.
Inputs:
**Input 1:** Value 0-100 (from PID).
**Input 2:**The amount of pumps in the system.
**Input 3:**Minimum value for the VSD command output (variable speed demand).
**Input 4:**Minimum time for output (pump) operation.
**Input 5:**Hysteresis for decreasing one output.
**Input6:**Binary input of pumps alarm status (0-255)
**Input7:**Reserved.
Outputs:
**Input1-8:** Receives the command of 0-100 to activate the pressure pumps.

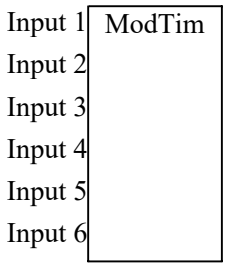
---

## 227 *ExFLS* External Flood sensor \r card:

## 228 *MKT* Storage min-max temp computed

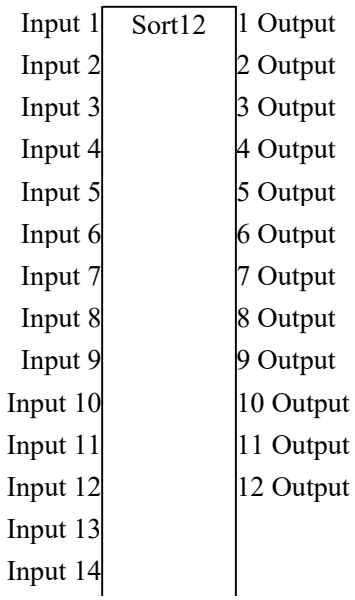



## 229 *ModTim* Modbus Timing




**ModTim :** Modbus timming allows to define the intervals and number of retries in modbus commands.
**Input 1:** Number of retries when reading data.
**Input 2:** The wait before time out (give up waiting) when reading data.
**Input 3:** Time interval when reading data.
**Input 4:** Number of retries when writing data.
**Input 5:** The wait before time out (give up waiting) when writing data.
**Input 6:** Time interval when writing data.
All the time intervals are in msec.

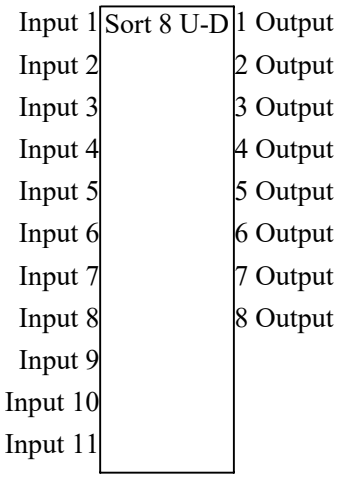
## 231 *Sort12* Sort Up-Down (12)



**Sort12:** Sorts the Inputs from high to low and from low to high.
**Inputs 1-12:** values to sort.
**Input 13:**"1"- sorts from low to high , "0"- sorts from high to low.
**Input 14:** The function will work only when changes from "0" to "1".

---

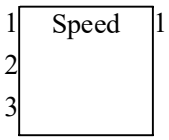
## 232 *Sort 8 U-D* Sort 8 Up-Down




**Sort 8 U-D:** Sorts the Inputs from high to low and from low to high.
**Inputs 1-8:**values to sort.
**Input 9:**"1"- sorts from low to high , "0"- sorts from high to low.
**Input 10:** The function will work only when changes from "0" to "1".
**Input 11:** The binary number of the unused inputs , when equal "1" this current input is disabled.

---

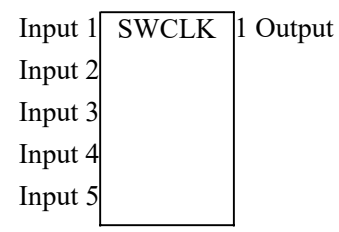
## 233 *Speed* Speed Up-Down Number




**Speed :** Dedicated to measure speed of process.
**Input 1:** Inserted value.
**Input 2:** Inserted average time.
**Input 3:** Reset.
**Output 1:** Inserted value rate change multiplied by avarge time (Input 2).

---

## 234 *SWCLK* Summer Winter Clock

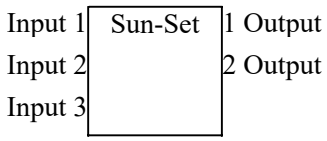



**SWCLK:** Summer Winter Clock change - changes the CPU's time at desirable date and time.

**Input 1:** The date to apply the time shifting.
**Input 2:** The month to apply the time shifting.
**Input 3:** The year to apply the time shifting (if nothing is connected will happen every year).
**Input 4:** The time (in min.) to apply the time shifting.
**Input 5:** The time shifting (in min.).
**Output :** Changes to "1" if the function worked.

---

## 235 *Sun-Set* Sunrise Sunset Time




**Inputs:**
Allows to caclulate the sun rise and sun set times according to location.
**Input 1:** Latitude (hours)
**Input 2:** Longitude (hours)
**Input 3:** Offset from Greenwich mean time (min)
**Outputs:**
**Output 1:** Sunrise time (min)
**Output 2:** Sunset (min)

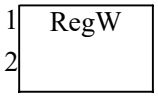
---

## 236 *RegR* Register Read (ITEM NUM)


**RegR:** Reads item numbers (Uniart parameters) - no need to specify file just enough to use the correct Item number from the communciation list of the controller.
(According to most right column in the PDF communication file)

---

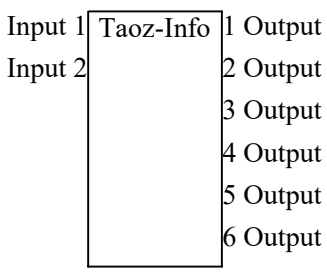
## 237 *RegW* Register Write (ITEM NUM)




**RegR:** Writes to item number (parameter) according to communication register table (according to most right column in the PDF communication file).
**Input 1:** Value to write in.
**Input 2:** Item number.

---

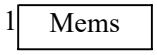
## 238 *Taoz-Info* Electricity Rate Information



**Taoz-Info :** Used to display data at shabat (saturday), it works with sun-set function.
**Input 1:** The time shabat start.
**Input 2:** Offset time.
**output 1:** TOU properties - "0"- low, "1"- medium. "2"- high.
**output 2:** The current season- "0"- summer, "1"- winter. "2"- autumn/spring.
**output 3:** "0"- Week day, "1"- friday, "2"- shabat( saturday). **output 4:** "1" - If the current day is an Israeli holiday . **output 5:** Shabat status.
**output 6:** not used.

---

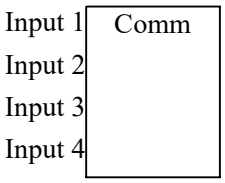
# 239 *Mems* memory param store (no volt)




**Mems:** Memory store function, this function stores data even if a power failure occured.

---

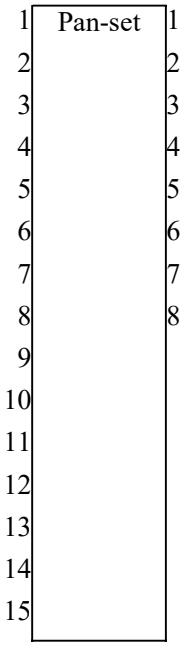
# 240 *Comm* Communication Setup




**Comm :** Allows configuring the communication definitions of the controllers comports.
**Input 1:** Baud Rate.
**Input 2:** Parity: 0=None, 1= Even, 2= Odd.
**Input 3:** Stop bit 2 or 1 (the data bit is default 8).
**Input 4:** Port: 1= Port A (PC) slave, 2= Port B (Panel) slave, 11 = Port A (PC) modbus master, 12 = Port B (Panel) modbus master.
**Only one of the ports can be defined as modbus master.**

---

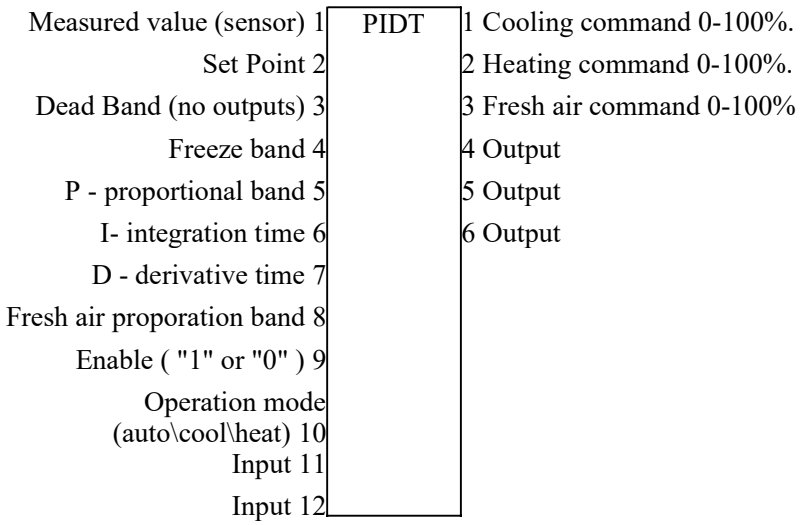
# 241 *Pan-set* Panel Setting



**Pan-set:** Allows defining and using Unisense room panel with SuperBrain and VeroPoint controllers.
The function internal number is the Panel address (in case of using more than 1 panel).
First 6 inputs of the function defines which parameter will be influenced by using pushbuttons of the Unisense panel.
**Input 1:** System status - System status inflected by push-buttons ON\OFF and presents the system status.
**Input 2:** S.P. (summer) - Inflected by UP\Down arrows.
**Input 3:** Speed status (1-2-3) - Presentable and changeable by speed push-button (depends on parameter defined in input 1).
**Input 4:** Summer = 0, winter = 1, auto = 2 - Summer\Winter selector (for auto mode press for several seconds on the button).
**Input 5:** S.P. (winter) - If par 116 = 1, and par 107 = 1, presentable and changeable by UP\Down arrows.
**Input 6:** Temperature value from Panels sensor.
**Inputs 7-13:** Allows defining panels limits and settings.
**Input 7:** Lowest Set Point limit - Limits the change of Set Point.
**Input 8:** Highest Set Point limit - Limits the change of Set Point
**Input 9:** Number of speed level (1-3) - Limits parameter defined in input 3.
**Input 10:** Internal sensor calibration - the value of calibration added to the value of the sensor which is installed in Unisense panel.
**Input 11:** Set Point type - 0 = regular, 1 = par defined in input 5 replacing par defined input 2 at winter mode.
**Input 12:** Set Point change levels - specifies the size of change being made by using the UP\Down arrows.
**Inputs 13-15:** Saved for future options.

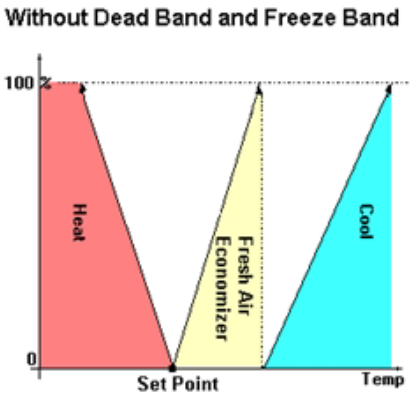
---

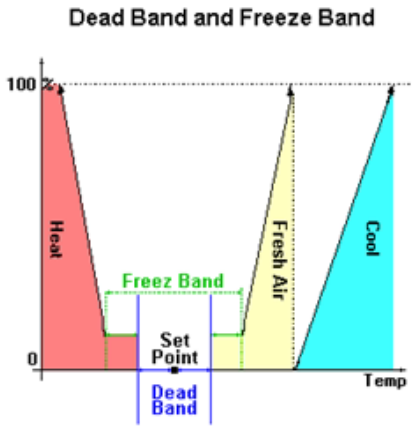
# 242 *PIDT* PID-Turbo




**PIDT:** In order to maintain a set point the function activates its proportional outputs by appling a PID control.
**Inputs:**
**Input 1:** Measured value (sensor).
**Input 2:** Set Point.
**Input 3:** Dead Band (no outputs).
**Input 4:** Freeze band (freezing the output status, if combined also with dead band then the freeze status will begin only after the dead band).
**Input 5:** P - proportional band.
**Input 6:** I- integration time (in seconds).
**Input 7:** D - derivative time (in seconds) .
**Input 8:** Fresh air proporation band (befor cooling used only with economizer, if no economizer exists then should be set to 0).
**Input9:** Enable ( "1" or "0" ).
**Input10:** Operation mode: 1 = auto mode (using cooling and heating) , 2 = cooling only mode, 3 = heating only mode.
**Input11:** reserved .
**Input12:** reserved .

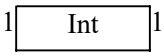
**Outputs:**
**Output1:** Cooling proportional command 0-100%.
**Output2:** Heating proportional command 0-100%.
**Output3:** Fresh air proportional command 0-100%.
**Output4:** debug (P).
**Output5:** debug (I).
**Output6:** debug (D).




---

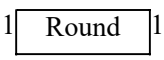
## 243 *Int* Integer




**Int :** Integer - the function "cuts" the number after the decimal separator and leaves only the integer part of the value on the output of the function.
For example if the input of the function receives 2.9 the output would be = 2

---

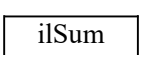
## 244 *Round* Round

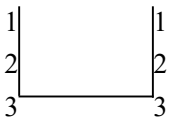



**Round:** The function rounds the value to the closest integer value.
For example if the input will receive a value of 2.4 the output of the function would be 2
if the input will receive a value of 2.7 the output of the function would be 2

---

## 246 *ilSum* summer-time, israel




1
2
3
1
2
3