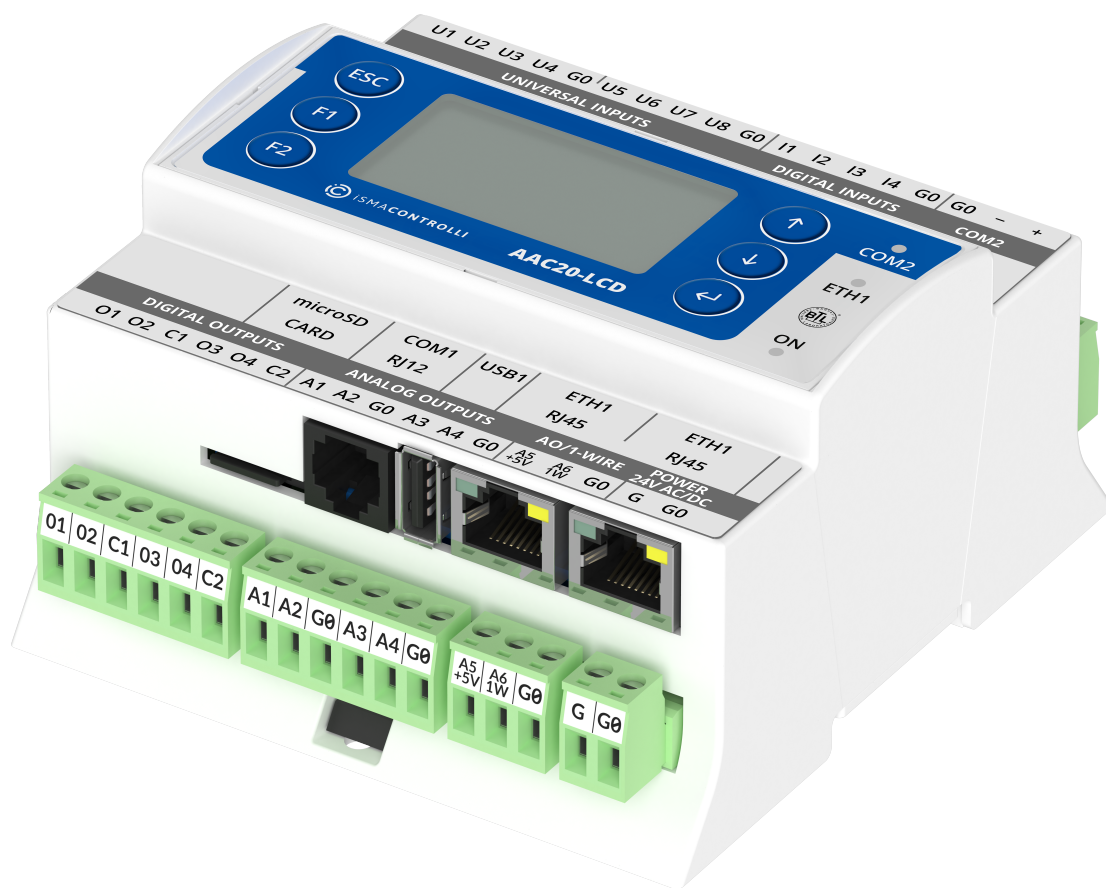


# iSMA-B-AAC20

User Manual

## Control Kit



Powered by  
**sedona**  
FRAMEWORK™

## Table of Contents

1	Introduction .....	6
1.1	Revision History.....	6
2	Components of the Control Kit.....	8
2.1	Conversion Components.....	8
2.1.1	BooleanToFloat .....	8
2.1.2	BooleanToPulse .....	9
2.1.3	FloatToBoolean .....	9
2.1.4	FloatToInteger .....	10
2.1.5	FloatToLong .....	11
2.1.6	FloatToString.....	12
2.1.7	IntegerToFloat .....	12
2.1.8	LongToFloat .....	13
2.2	Select Components.....	13
2.2.1	BooleanSelect.....	13
2.2.2	IntegerSelect.....	14
2.2.3	NumericSelect .....	15
2.3	Demux Components .....	16
2.3.1	BooleanDemux .....	16
2.3.2	IntegerDemux.....	17
2.3.3	NumericDemux.....	18
2.4	Energy Components .....	19
2.4.1	DegreeDays.....	19
2.4.2	OptimizedStartStop .....	21
2.4.3	NightPurge .....	25
2.4.4	Psychrometric.....	26
2.4.5	OutsideAirOptimization .....	27
2.5	HVAC Components .....	29
2.5.1	LeadLagCycles .....	29
2.5.2	LeadLagRuntime .....	31
2.5.3	LoopPoint .....	33
2.5.4	BackwardEulerLoopPoint .....	35
2.5.5	SequenceBinary .....	37
2.5.6	SequenceFailover .....	39
2.5.7	SequenceLinear .....	40
2.5.8	Thermostat.....	43

2.5.9	Tstat .....	44
2.6	Latch Components.....	45
2.6.1	BooleanLatch.....	45
2.6.2	IntegerLatch.....	46
2.6.3	NumericLatch .....	47
2.6.4	SRLatch .....	47
2.7	Logic Components .....	48
2.7.1	And.....	48
2.7.2	LogicExpr .....	50
2.7.3	Or.....	51
2.7.4	Nand .....	52
2.7.5	Nor .....	53
2.7.6	Not.....	54
2.7.7	Xor .....	55
2.7.8	Comparator.....	56
2.7.9	CompareExpr.....	57
2.7.10	Equal .....	58
2.7.11	GreaterThan.....	59
2.7.12	GreaterThanEqual .....	60
2.7.13	LessThan.....	60
2.7.14	LessThanEqual .....	61
2.7.15	NotEqual .....	62
2.8	Math Components .....	62
2.8.1	Add.....	62
2.8.2	MathExpr .....	63
2.8.3	Maximum.....	65
2.8.4	Minimum.....	65
2.8.5	MinMaxAvg.....	66
2.8.6	Multiply.....	67
2.8.7	Divide.....	68
2.8.8	Modulus .....	69
2.8.9	Power.....	69
2.8.10	Subtract.....	70
2.8.11	AbsValue .....	71
2.8.12	ArcCosine.....	71
2.8.13	ArcSine .....	72
2.8.14	ArcTangent.....	73

2.8.15	Cosine.....	73
2.8.16	Exponential.....	74
2.8.17	Factorial.....	75
2.8.18	LogBase10.....	75
2.8.19	LogNatural.....	76
2.8.20	MinMaxAverage.....	77
2.8.21	Negative.....	78
2.8.22	Reset.....	78
2.8.23	Round.....	80
2.8.24	Sine.....	80
2.8.25	SquareRoot.....	81
2.8.26	Tangent.....	82
2.8.27	Truncate.....	82
2.9	Switch Components.....	83
2.9.1	BooleanSwitch.....	83
2.9.2	IntegerSwitch.....	84
2.9.3	NumericSwitch.....	85
2.10	Timer Components.....	86
2.10.1	BooleanDelay.....	86
2.10.2	OneShot.....	88
2.10.3	NumericDelay.....	88
2.10.4	Timer.....	89
2.11	Utility Components.....	90
2.11.1	NumericBitXor.....	90
2.11.2	Counter.....	91
2.11.3	Multivibrator.....	93
2.11.4	NumericBitAnd.....	93
2.11.5	NumericBitOr.....	94
2.11.6	Ramp.....	95
2.11.7	Random.....	96
2.11.8	RateOfChange.....	96
2.11.9	ReheatSequence.....	97
2.11.10	SineWave.....	98
2.11.11	Frequency.....	99
2.11.12	Hysteresis.....	100
2.11.13	Limiter.....	101
2.11.14	Linearize.....	102

2.11.15	TempConversion.....	104
2.11.16	Toggle .....	105
2.11.17	UpDown.....	106

# 1 Introduction

This manual contains information about the iSMA Control kit in the AAC20 controller. The Control kit can be used in all AAC20 hardware versions with firmware 3.4 version or higher. The Control kit is installed by default in the AAC20 controller and cannot be uninstalled.

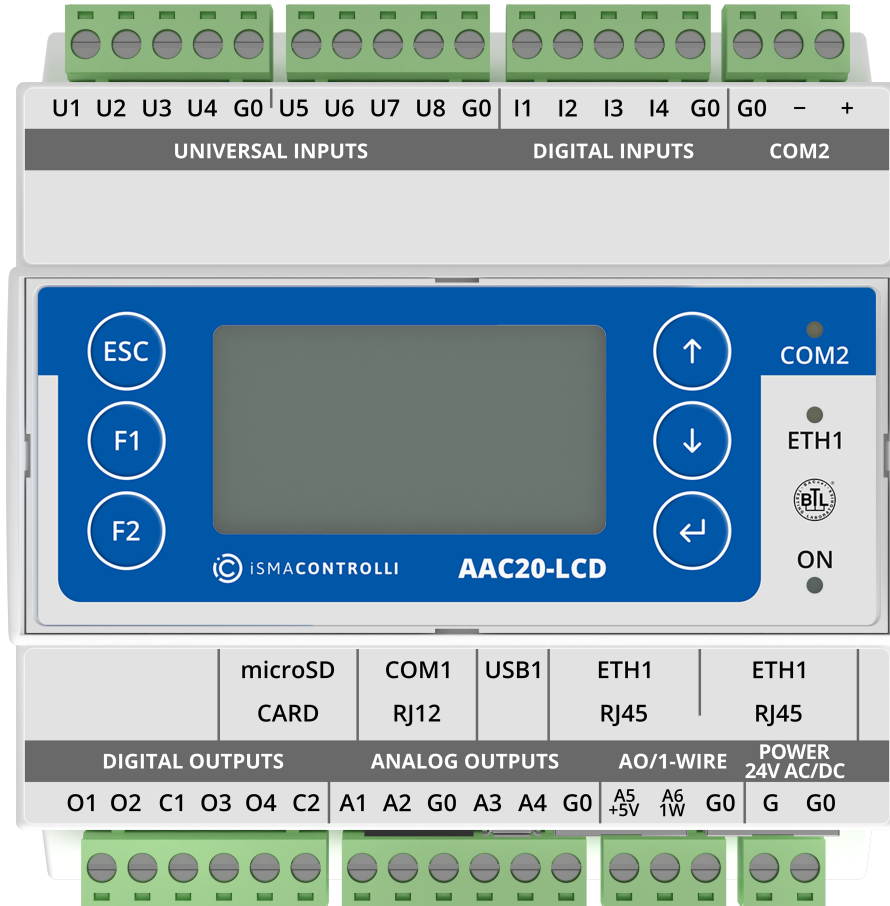


Figure 1. AAC20-LCD controller

## 1.1 Revision History

Rev.	Date	Description
1.3	19 Jun 2024	<ul style="list-style-type: none"> <li>Added/corrected descriptions of components:                             <ul style="list-style-type: none"> <li>RateOfChange;</li> <li>BackwardEulerLoopPoint.</li> </ul> </li> </ul>
1.2	21 Apr 2022	<ul style="list-style-type: none"> <li>Added/corrected descriptions of components:                             <ul style="list-style-type: none"> <li>LeadLagCycle;</li> <li>LeadLagRuntime;</li> <li>Tstat;</li> <li>Tharmostat;</li> <li>Minimum.</li> </ul> </li> <li>Rebranded</li> </ul>
1.1	21 Jan 2020	<ul style="list-style-type: none"> <li>Replaced environment of programming from Workplace to iSMA Tool</li> </ul>

Rev.	Date	Description
1.0	30 Jan 2018	• First edition

*Table 1. Revision history*

## 2 Components of the Control Kit

This section outlines components included in the Control kit grouped by their functionalities.

### 2.1 Conversion Components

This section outlines components converting signal types.

#### 2.1.1 BooleanToFloat

The BooleanToFloat component converts 16 Boolean signals to 1 float signal.

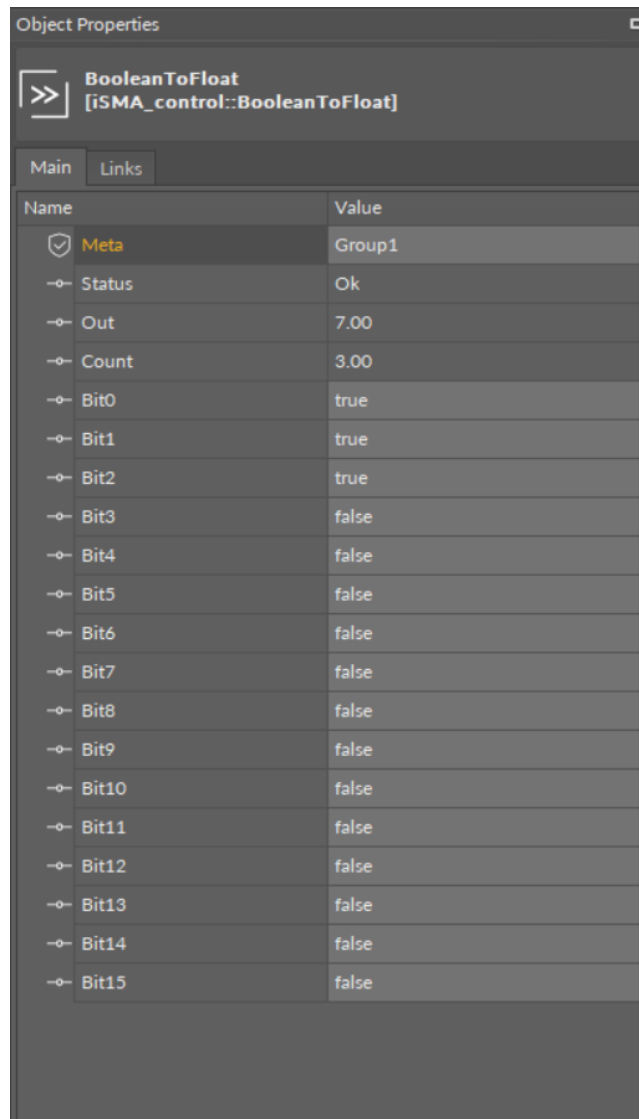


Figure 2. BooleanToFloat component

### Slots

The BooleanToFloat component has the following slots:

- **Status:** shows the component's status;
- **Out:** encoded value of inputs with bit 15 (MSB) and bit 0 (LSB);
- **Count:** sum of active inputs;
- **Bit0-Bit16:** any bit of true (1, +) or false (0, -) value.



## 2.1.2 BooleanToPulse

The BooleanToPulse component converts 1 Boolean signal to pulse. The BooleanToPulse component is a simple monostable oscillator object.

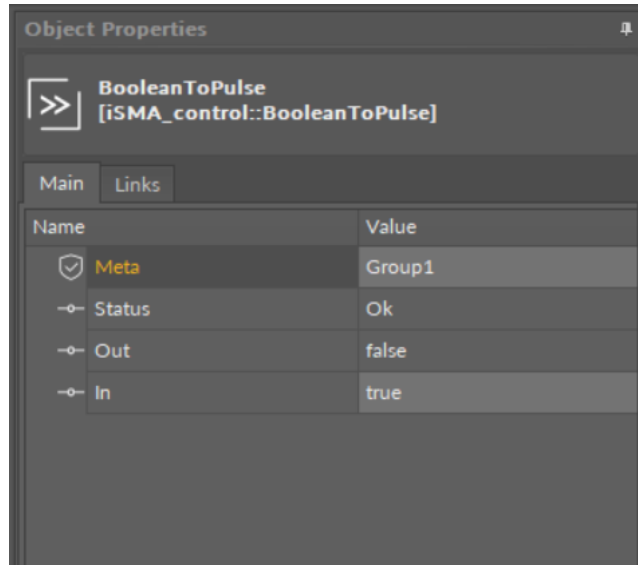


Figure 3. BooleanToPulse component

### Slots

The BooleanToPulse component has the following slots:

- **Status:** shows the component's status;
- **Out:** generates an impulse during the cycle where the rising edge occurs;
- **In:** the input value.

## 2.1.3 FloatToBoolean

The FloatToBoolean component converts a 16-bit float to a binary decoder object.

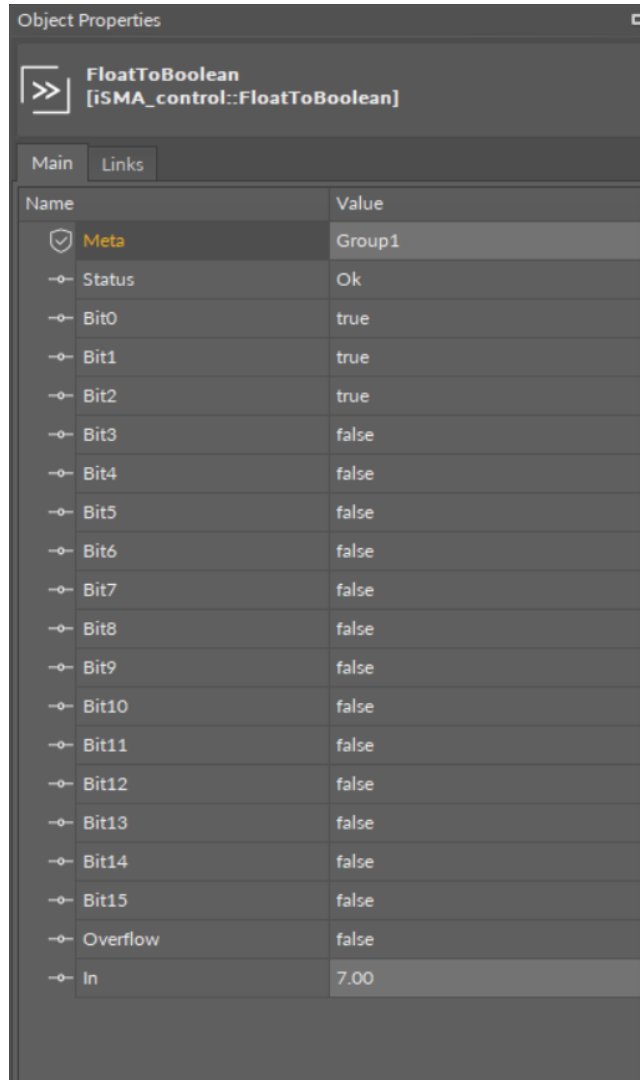


Figure 4. FloatToBoolean component

## Slots

The FloatToBoolean component has the following slots:

- **Status:** shows the component's status;
- **Bit0-Bit15:** the output slots showing decoded value of inputs with bit15(MSB) and bit0(LSB);
- **Overflow:** true if inNumeric > 65535
- **In:** the numeric input value.

### 2.1.4 FloatToInteger

The FloatToInteger is a float to 32-bit integer converter object.

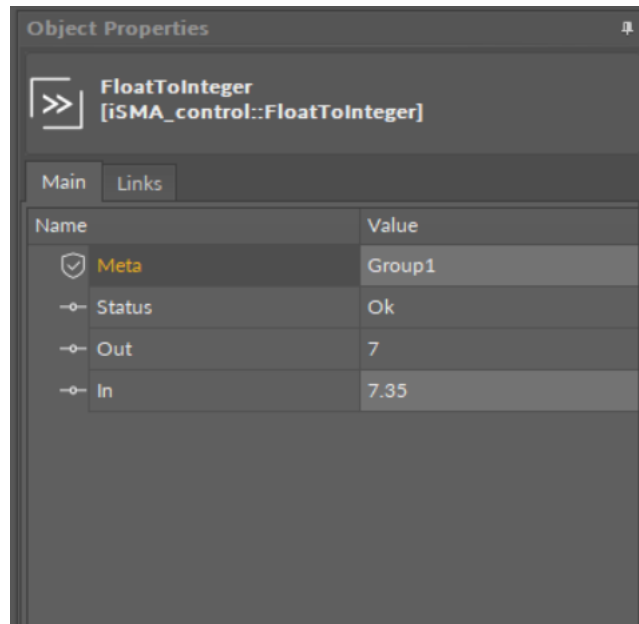


Figure 5. FloatToInteger component

## Slots

The FloatToInteger component has the following slots:

- **Status:** shows the component's status;
- **Out:** the input value converted to the 32-bit integer with fractional part truncated;
- **In:** the input value.

### 2.1.5 FloatToLong

The FloatToLong component is a float to 64-bit signed integer (long) converter object.

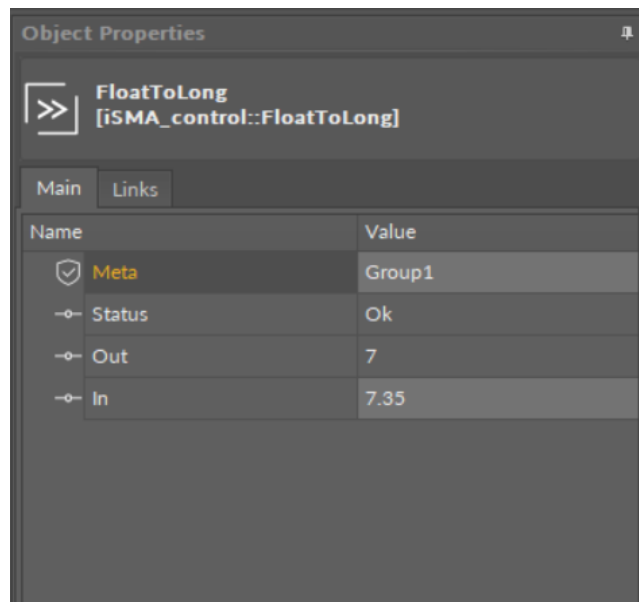


Figure 6. FloatToLong component

## Slots

The FloatToLong component has the following slot:

- **Status:** shows the component's status;

- **Out:** the input value converted to the 64-bit signed integer with fractional part;
- **In:** the input value.

### 2.1.6 FloatToString

The FloatToString component converts float to string objects.

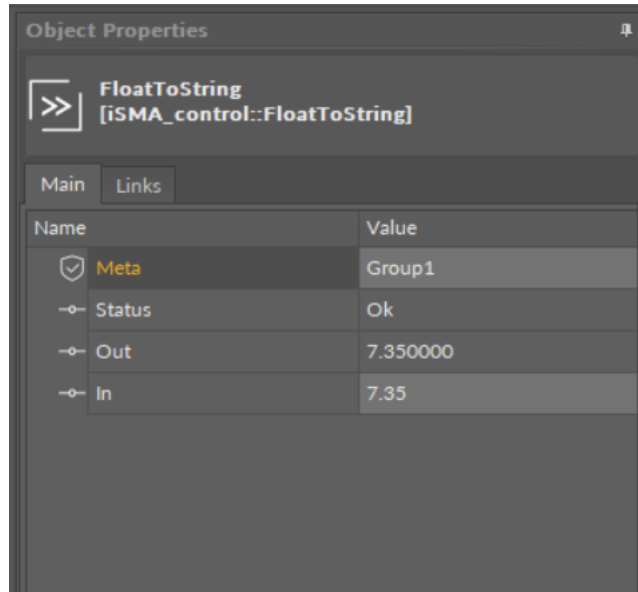


Figure 7. FloatToString component

### Slots

The FloatToString component has the following slot:

- **Status:** shows the component's status;
- **Out:** the input value converted to the Buff (64-bit) with fractional part truncated;
- **In:** the input value.

### 2.1.7 IntegerToFloat

The IntegerToFloat component converts 32 integer bits to a float object.

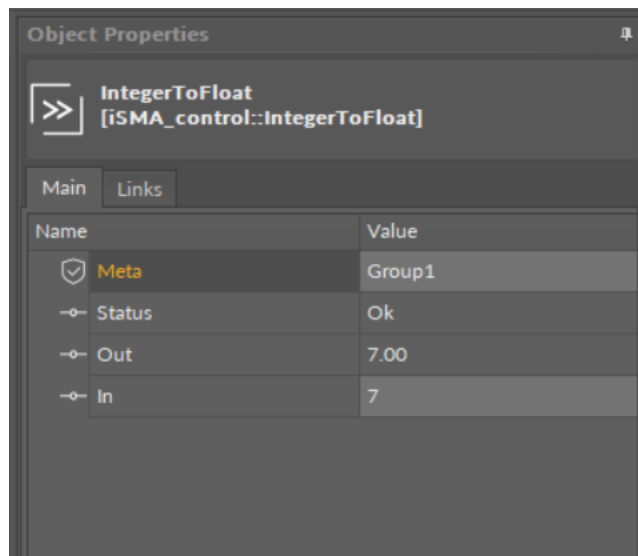


Figure 8. IntegerToFloat component

## Slots

The IntegerToFloat component has the following slots:

- **Status:** shows the component's status;
- **Out:** the input value converted to the float;
- **In:** the input value.

### 2.1.8 LongToFloat

The LongToFloat component converts a 64-bit signed, integer (long) to a float object.

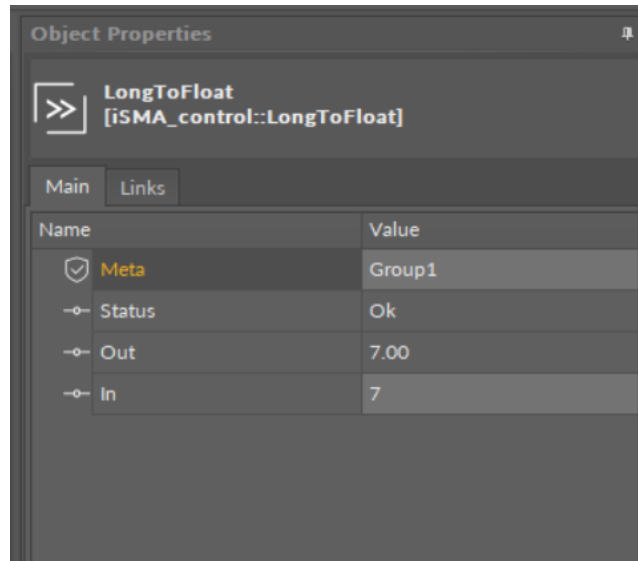


Figure 9. LongToFloat component

## Slots

The LongToFloat component has the following slot:

- **Status:** shows the component's status;
- **Out:** the input value converted to the float;
- **In:** the input value.

## 2.2 Select Components

This section outlines select components.

### 2.2.1 BooleanSelect

The BooleanSelect component allows one of the multiple Boolean inputs to be selected (passed to the output), based of the Select (integer) input. Inputs from A to J can be specified (1-10).

**Note:** All select components require an integer input in the Select slot to properly operate and select the input to be passed to the Out slot (depending on the type of component: Boolean, integer, or numeric).

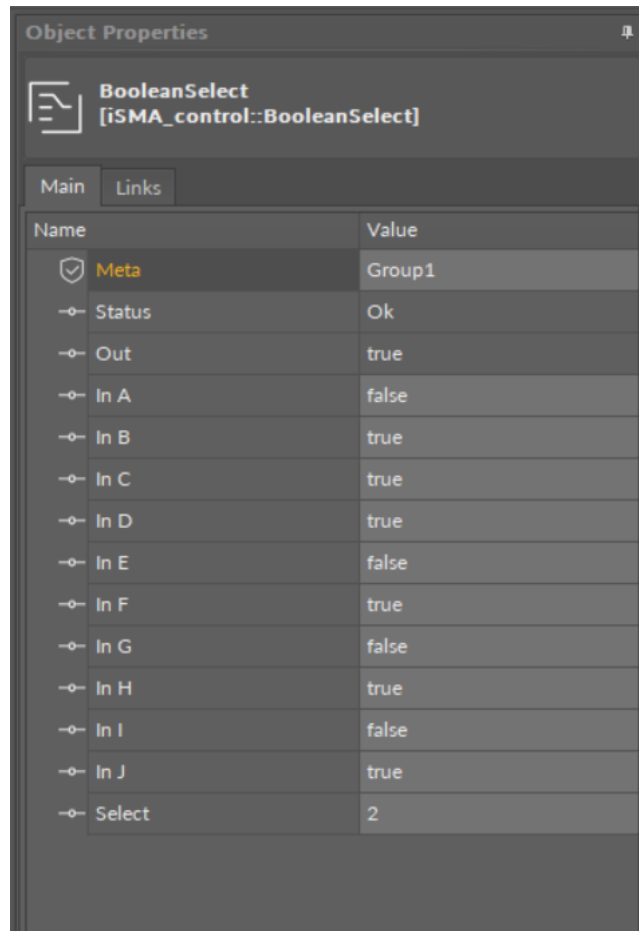


Figure 10. BooleanSelect component

## Slots

The BooleanSelect component has the following slots:

- **Status:** shows the component' status;
- **Out:** the value passed from the In slot indicated in the Select slot;
- **InA-Inj:** 10 input slots;
- **Select:** indicates the number of input, which passes the value to the Out slot;

### 2.2.2 IntegerSelect

The IntegerSelect component allows one of the multiple integer inputs to be selected (passed to the output), based of the Select (integer) input. Inputs from A to J can be specified (1-10).

**Note:** All select components require an integer input in the Select slot to properly operate and select the input to be passed to the Out slot (depending on the type of component: Boolean, integer, or numeric).

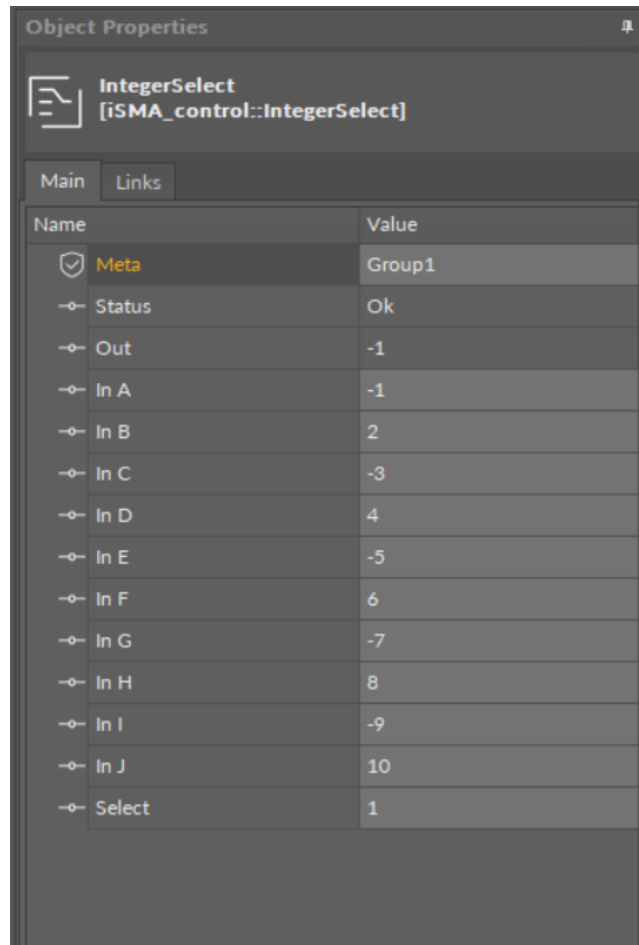


Figure 11. IntegerSelect component

## Slots

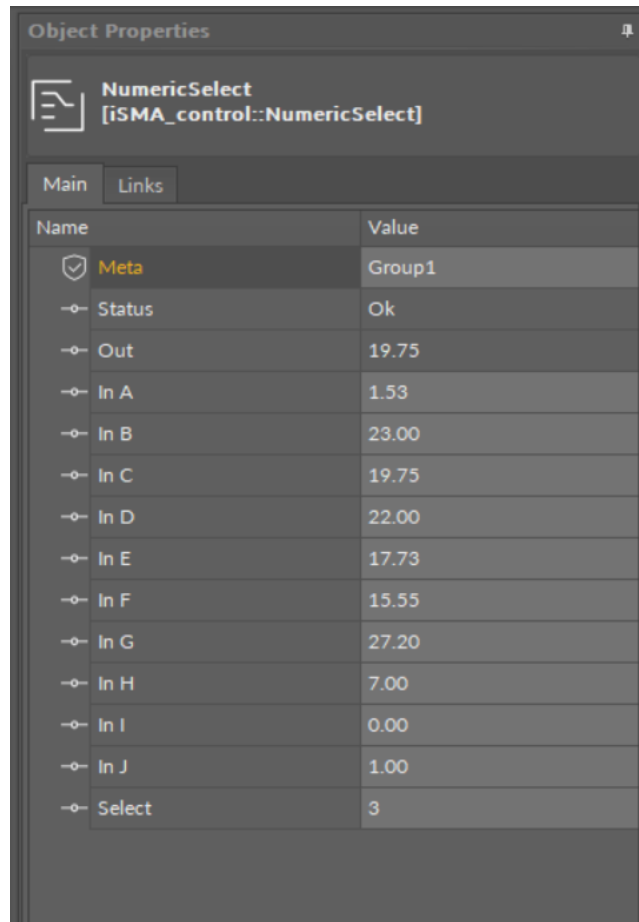
The IntegerSelect component has the following slots:

- **Status:** shows the component' status;
- **Out:** the value passed from the In slot indicated in the Select slot;
- **InA-InJ:** 10 input slots;
- **Select:** indicates the number of input, which passes the value to the Out slot;

### 2.2.3 NumericSelect

The NumericSelect component allows one of the multiple numeric inputs to be selected (passed to the output), based of the Select (integer) input. Inputs from A to J can be specified (1-10).

**Note:** All select components require an integer input in the Select slot to properly operate and select the input to be passed to the Out slot (depending on the type of component: Boolean, integer, or numeric).



Name	Value
Meta	Group1
Status	Ok
Out	19.75
In A	1.53
In B	23.00
In C	19.75
In D	22.00
In E	17.73
In F	15.55
In G	27.20
In H	7.00
In I	0.00
In J	1.00
Select	3

Figure 12. NumericSelect component

## Slots

The NumericSelect component has the following slots:

- **Status:** shows the component' status;
- **Out:** the value passed from the In slot indicated in the Select slot;
- **InA-InJ:** 10 input slots;
- **Select:** indicates the number of input, which passes the value to the Out slot;.

## 2.3 Demux Components

This section outlines components converting signal types.

### 2.3.1 BooleanDemux

The BooleanDemux component selects one of two outputs to receive the input (Boolean) value, depending on the value of the selected Boolean input. The value of the other output remains unchanged.



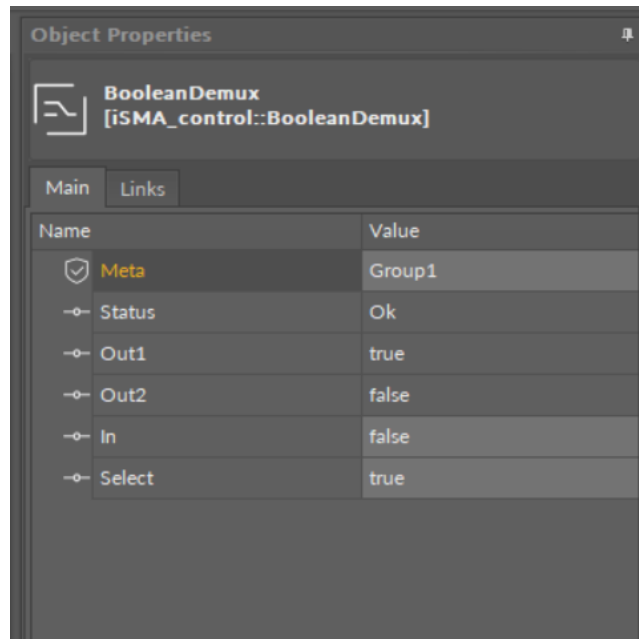


Figure 13. BooleanDemux component

## Slots

The BooleanDemux component has the following slots:

- **Status:** shows the component's status;
- **Out1-Out2:** two output slots showing the transferred input value depending on the Select slot value;
- **In:** the input value;
- **Select:** indicates the number of output, which receives the value from the In slot.

### Select Slot

If the Select slot is false, the input value is transferred to the Out1 slot, and the Out2 slot shows a previous value or null.

If the Select slot is true, the input value is transferred to the Out2 slot, and the Out1 slot shows a previous value or null.

## 2.3.2 IntegerDemux

The IntegerDemux component selects one of two outputs to receive the input (integer) value, depending on the value of the selected Boolean input. The value of the other output remains unchanged.

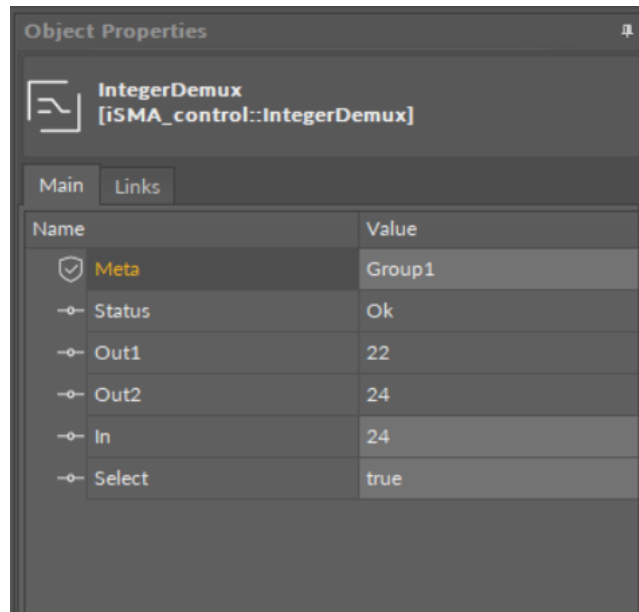


Figure 14. IntegerDemux

## Slots

The IntegerDemux component has the following slots:

- **Status:** shows the component's status;
- **Out1-Out2:** two output slots showing the transferred input value depending on the Select slot value;
- **In:** the input value;
- **Select:** indicates the number of output, which receives the value from the In slot.

### Select Slot

If the Select slot is false, the input value is transferred to the Out1 slot, and the Out2 slot shows a previous value or 0.

If the Select slot is true, the input value is transferred to the Out2 slot, and the Out1 slot shows a previous value or 0.

### 2.3.3 NumericDemux

The NumericDemux component selects one of two outputs to receive the input (numeric) value, depending on the value of the selected Boolean Input. The value of the other output remains unchanged.

Name	Value
Meta	Group1
Status	Ok
Out1	23.00
Out2	0.00
In	23.00
Select	false

Figure 15. NumericDemux

## Slots

The NumericDemux component has the following slots:

- **Status:** shows the component's status;
- **Out1-Out2:** two output slots showing the transferred input value depending on the Select slot value;
- **In:** the input value;
- **Select:** indicates the number of output, which receives the value from the In slot.

### Select Slot

If the Select slot is false, the input value is transferred to the Out1 slot, and the Out2 slot shows a previous value or 0.

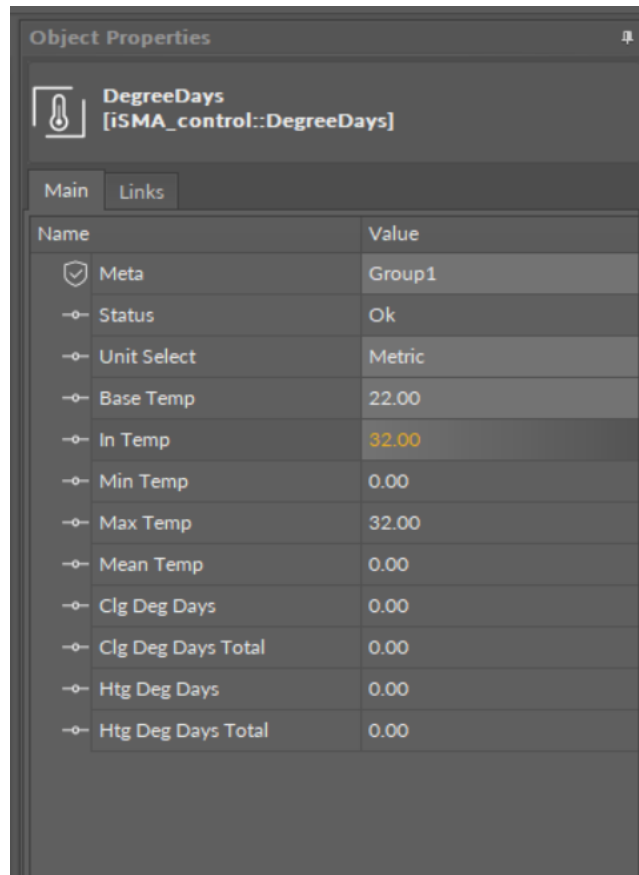
If the Select slot is true, the input value is transferred to the Out2 slot, and the Out1 slot shows a previous value or 0.

## 2.4 Energy Components

This section outlines components for energy saving management.

### 2.4.1 DegreeDays

The DegreeDays component provides degree-day calculations, based upon temperature received at the input Temperature slot and values of various other properties.



Name	Value
Meta	Group1
Status	Ok
Unit Select	Metric
Base Temp	22.00
In Temp	32.00
Min Temp	0.00
Max Temp	32.00
Mean Temp	0.00
Clg Deg Days	0.00
Clg Deg Days Total	0.00
Htg Deg Days	0.00
Htg Deg Days Total	0.00

Figure 16. DegreeDays component

## Slots

The DegreeDays component has the following slots:

- **Status:** shows the component's status;
- **Unit Select:** used to set the units of the Temp In, Min Temp, Max Temp, and Mean Temp properties;
- **Base Temperature:** specifies the base temperature used in the degree-day calculation;
- **Input Temperature:** the input for the outside air temperature used in the degree-day calculation;

**Note:** If this input is not valid then no calculations will be done.

- **Minimum Temperature:** the minimum temperature recorded for the current day; tested and set on each calculation;
- **Maximum Temperature:** the maximum temperature recorded for the current day; tested and set on each calculation;
- **Mean Temperature:** the mean temperature recorded for the previous day; calculated when the day changes.  $\text{Mean Temp} = (\text{Max Temp} + \text{Min Temp}) / 2.0$ ;
- **Cooling Degree-day:** this is the cooling degree-day calculated for the previous day; calculated when the day changes;
- **Totalized Cooling Degree-days:** this is the totalized cooling degree-days since the last Reset Totals action was invoked; calculated when the Cooling Degree-day changes;
- **Heating Degree-day:** the heating degree-day calculated for the previous day; calculated when the day changes;

- **Totalized Heating Degree-days:** the totalized heating degree-days since the last Reset Totals action was invoked; calculated when the Heating Degree-day changes.

## 2.4.2 OptimizedStartStop

The OptimizedStartStop component allows using Start Time Optimization and Stop Time Optimization for energy saving. This component uses a space temperature input and area characteristics to calculate an optimal amount of lead-time before a scheduled event. It can analyze area temperature changes and adjust the optimization parameters based on the actual temperature change rates after an optimized start or stop.

Name	Value
<input checked="" type="checkbox"/> Meta	Group1
<input type="checkbox"/> Status	Ok
<input type="checkbox"/> Heat Cool Mode	heatMode
<input type="checkbox"/> Parameter Reset Time	0
<input type="checkbox"/> Start Enable	true
<input type="checkbox"/> Stop Enable	false
<input type="checkbox"/> Schedule Status	false
<input type="checkbox"/> Next Event Time	0
<input type="checkbox"/> Next Event Value	false
<input type="checkbox"/> Outside Temp	15.00
<input type="checkbox"/> Space Temp	22.00
<input type="checkbox"/> Start Time Command	null
<input type="checkbox"/> Stop Time Command	null
<input type="checkbox"/> Upper Comfort Limit	77.00
<input type="checkbox"/> Lower Comfort Limit	68.00
<input type="checkbox"/> Dynamic Parameter Adjust	true
<input type="checkbox"/> Old Parameter Multiplier	2
<input type="checkbox"/> Earliest Start Time	0
<input type="checkbox"/> Earliest Stop Time	0
<input type="checkbox"/> Drifttime Per Degree Coolin...	0.00
<input type="checkbox"/> Drifttime Per Degree Heatin...	0.00
<input type="checkbox"/> Runtime Per Degree Coolin...	0.00
<input type="checkbox"/> Runtime Per Degree Heatin...	0.00
<input type="checkbox"/> Drifttime Per Degree Cooling	10.00
<input type="checkbox"/> Drifttime Per Degree Heating	10.00
<input type="checkbox"/> Runtime Per Degree Cooling	10.00
<input type="checkbox"/> Runtime Per Degree Heating	10.00
<input type="checkbox"/> Last Start Time	0
<input type="checkbox"/> Last Stop Time	0
<input type="checkbox"/> Outside Temp At Beginning	0.00
<input type="checkbox"/> Space Temp At Beginning	0.00
<input type="checkbox"/> Calculated Command Time	0
<input type="checkbox"/> Program Mode	0

Figure 17. OptimizedStartStop component

## Slots

The OptimizedStartStop component has the following slots:

- **Heat Cool Mode:** (Boolean) allows enabling either the Heat mode or the Cool mode. The selected option applies only to optimized stop calculations, which means that optimized stop calculations are performed only for the selected mode. Optimized start

calculations are performed for both heat and cool modes, regardless of this property value;

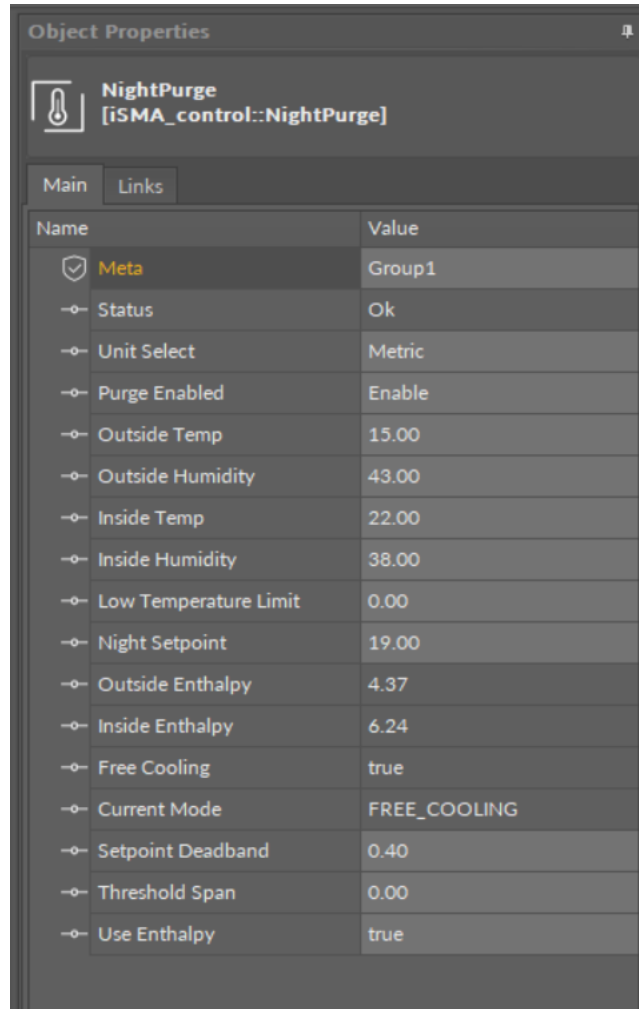
- **Parameter Reset Time:** displays the time when any of the four runtime or drift time properties change to the User Defined values. The OSS component copies the user-defined drift time and runtime property values to the corresponding actual drift time and runtime property values;
- **Start Enable:** allows to manually or automatically enable or disable the optimized start function;
- **Stop Enable:** this property allows to manually or automatically enable or disable the optimized stop function;
- **Schedule Status:** (Boolean) monitors and displays the status of the schedule that is linked to it;
- **Next Event Time:** this property is linked to the schedule for the time of the next scheduled event;
- **Next Event Value:** this property is linked to the schedule and reflects the value of the action for the next scheduled event;
- **Outside Temp:** this property is linked to outside temperature and displays the value for information only;
- **Space Temp:** this property is linked to a space temperature output and displays the temperature of the area affected by equipment associated with the OSS component;
- **Start Time Command:** (Boolean) the output that if linked to the control algorithm invokes an equipment start command. For example, it can be linked to a prioritized input of a Boolean writable point - or directly to the equipment start control;
- **Stop Time Command:** (Boolean) the output that if linked to the control invokes an equipment stop command. For example, it can be linked to a prioritized input of a Boolean writable - or directly to the equipment stop control;
- **Upper Comfort Limit:** this property value is the Cooling mode target temperature.
- **Lower Comfort Limit:** the Heating mode target temperature;
- **Dynamic Parameter Adjust:** controls whether or not calculation parameters are programmatically adjusted after an execution. After the OSS component completes the start or stop control, if this property value is set to true, the component evaluates the actual recovery rate (degrees/hour) and automatically adjusts the Runtime and Drifftime properties values so that they are influenced by actual drift time and run time;
- **Old Parameter Multiplier:** weighs the dynamic parameter adjustment calculation. The value that is specified in this field affects how much weight is assigned to the previous runtime property value when it is used in the dynamic parameter adjustment calculation. A larger value increases the weight given to the previous runtime and a smaller value decreases the weight;
- **Earliest Start Time:** allows to specify a time before which no optimized start command may be issued. If this value is set earlier than the Calculated Command Time, the Calculated Command Time is adjusted to equal to this time;
- **Earliest Stop Time:** allows to specify a time before which no stop command may be issued. If this value is set earlier than the Calculated Command Time, the Calculated Command Time is adjusted to equal to this time;
- **Drifftime Per Degree Cooling User Defined:** allows to set a default value for calculating the rate of drift in a cooling mode. A value saved in this field is copied to the Drifftime Per Degree Cooling field;

- **Drifftime Per Degree Heating User Defined:** allows to set a default value for calculating the rate of drift in a heating mode. A value saved in this field is copied to the Drifftime Per Degree Heating field;
- **Runtime Per Degree Cooling User Defined:** allows to set a default value for calculating the runtime value in a cooling mode. A value saved in this field is copied to the Runtime Per Degree Cooling field;
- **Runtime Per Degree Heating User Defined:** allows to set a default value for calculating the runtime value in a heating mode. A value saved in this field is copied to the Runtime Per Degree Heating field;
- **Drifftime Per Degree Cooling:** displays the actual value that is used for calculating an optimized stop time when the equipment is in a cooling mode. This value is adjusted automatically if the Dynamic Parameter Adjust value is set to true;
- **Drifftime Per Degree Heating:** displays the actual value that is used for calculating an optimized stop time when the equipment is in a heating mode. This value is adjusted automatically if the Dynamic Parameter Adjust value is set to true;
- **Runtime Per Degree Cooling:** displays the actual value that is used for calculating an optimized start time when the equipment is in a cooling mode. This value is adjusted automatically if the Dynamic Parameter Adjust value is set to true;
- **Runtime Per Degree Heating:** displays the actual value that is used for calculating an optimized start time when the equipment is in a heating mode. This value is adjusted automatically if the Dynamic Parameter Adjust value is set to true;
- **Last Start Time:** the record of the last Start Time that was used for calculating an optimized start time. Since only one optimized start per day is allowed, this value does not display Start Times (restarts) that are subsequent to the initial Start Time for a day;
- **Last Stop Time:** the record of the last Stop Time that was used for calculating an optimized stop time. Since multiple Optimized Stops are allowed in a day, this value changes to reflect the latest Optimized Stop time;
- **Outside Temp at Beginning:** the record of what the outside air temperature was at the time of the last start or stop command. This is the temperature that was used in calculations of dynamic parameter adjustment;
- **Space Temp at Beginning:** the record of what the space temperature was at the time of the last start or stop command. This is the temperature that was used in calculations of dynamic parameter adjustment;
- **Calculated Command Time:** shows the calculated time for the next command. This could be a start or a stop command;
- **Program Mode:** As a part of the logic that the OSS component uses, there are five program mode states. These states serve primarily in logic control; however, they may as well be informative to the system engineer. The Program Mode value displays the current heating or cooling state for the optimized start or stop. The following list describes the possible display values and meanings:
  - 0 ("No" Calculation): indicates that no calculation is being made;
  - 1 ("Start" Calculation): indicates that the optimized start calculation process is ongoing, but that an optimized start or stop is not yet in progress;
  - 2 ("Start" in Process): indicates that the optimized start has been initiated;
  - 3 ("Stop" Calculation): indicates that the optimized stop calculation process is ongoing, but that an optimized start or stop is not yet in progress;
  - 4 ("Stop" in Process): indicates that the optimized stop has been initiated.



## 2.4.3 NightPurge

The NightPurge component uses the two sets of temperature and humidity inputs to find the air supply with the least amount of heat if the Purge Enabled input is true. The Free Cooling output will be set to false if outside is greater than or equal to inside or set to true if outside equals to the Night Setpoint.



Name	Value
Meta	Group1
Status	Ok
Unit Select	Metric
Purge Enabled	Enable
Outside Temp	15.00
Outside Humidity	43.00
Inside Temp	22.00
Inside Humidity	38.00
Low Temperature Limit	0.00
Night Setpoint	19.00
Outside Enthalpy	4.37
Inside Enthalpy	6.24
Free Cooling	true
Current Mode	FREE_COOLING
Setpoint Deadband	0.40
Threshold Span	0.00
Use Enthalpy	true

Figure 18. NightPurge component

## Slots

The NightPurge component has the following slots:

- **Status:** shows the component's status;
- **Unit Select:** specifies the units of the temperature and humidity properties;
- **Purge Enabled:** Boolean data type, must be true to enable the night purge operation. Whenever false, the Free Cooling output is set to the opposite of the Free Cooling Command (or null, if the Use Null Output is set to true), and the Current Mode slot value is Disabled. The Purge Enabled is often linked to a "Not" object sourced from the Boolean Schedule output;
- **Outside Temperature:** input for the current outside air temperature; the input must be valid for this object to function;
- **Outside Humidity:** input for the current outside air humidity; the input must be valid for this object to function;

- **Inside Temperature:** input for the current inside air temperature; the input must be valid for this object to function;
- **Inside Humidity:** input for the current inside air humidity; the input must be valid for this object to function;
- **Low Temperature Limit:** this property is used to provide freeze protection;
- **Night Setpoint:** inside night temperature setpoint, at or below which the Free Cooling is not applied. Instead, the Current Mode is set to Satisfied;
- **Outside Enthalpy:** this is the calculated outside air enthalpy;
- **Inside Enthalpy:** this is the calculated inside air enthalpy;
- **Free Cooling:** Boolean output set to the value of the Free Cooling command when it is determined that free cooling should be used. Otherwise, the value is set to the opposite state or null (if the Used Null Output is set to true);
- **Current Mode:** indicates which of the following modes the component is currently in;
  - Available options: Disabled (Purge Enabled is false), Free Cooling, No Free Cooling (free cooling not available), Low temperature (Outside Temp Below Low Temperature Limit, free cooling disabled), Input error (temperature or humidity is invalid-down, fault, etc.), free cooling disabled, satisfied (inside temperature below the Night Setpoint, free cooling disabled);
- **Setpoint Deadband:** the Temperature Setpoint Deadband is applied when the inside temperature falls below the Night Setpoint before free cooling can be enabled; the default value is 1.0;
- **Threshold Span:** the difference between the inside enthalpy and the outside enthalpy must be greater than this value before free cooling will be enabled; the default value is 1.0;
- **Use Enthalpy:** setting this property to true will enable the use of enthalpy for determining if free cooling is available. Otherwise, it will just use outside and inside temperature to decide.

#### 2.4.4 Psychrometric

The Psychrometric component is used to support applications that need to calculate the properties of moist air using given temperature and humidity inputs.

Name	Value
Meta	Group1
Status	Ok
Unit Select	Metric
In Temp	22.0
In Humidity	38.0
Out Dew Point	7.0
Out Enthalpy	37.943
Out Sat Press	2.645
Out Vapor Press	1.005
Out Wet Bulb Temp	13.6

Figure 19. Psychrometric component

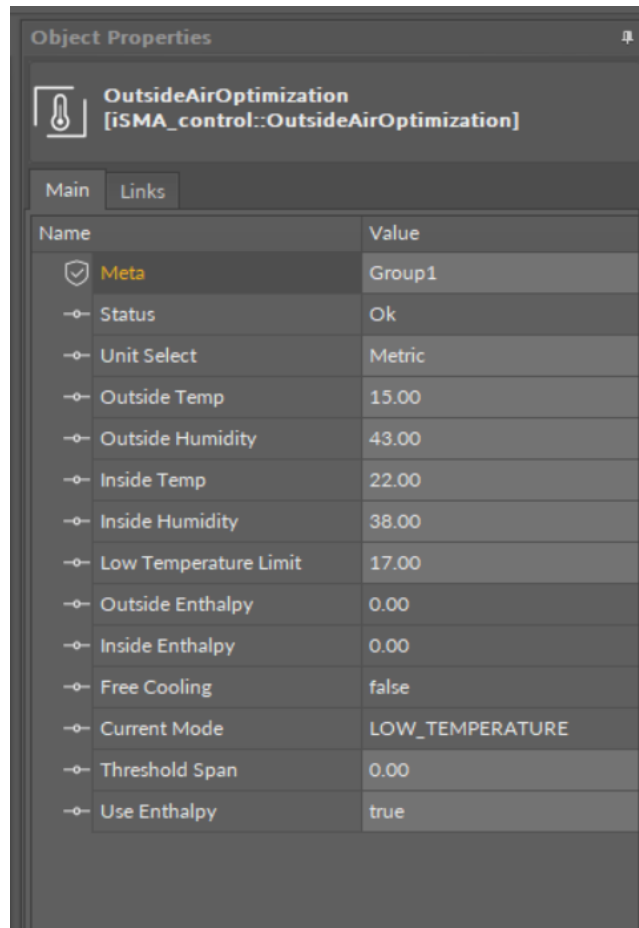
## Slots

The Psychrometric component has the following slots:

- **Status:** shows the component's status;
- **Unit Select:** used to set the units of the Temperature and Humidity properties;
- **Input Temperature:** input temperature;
- **Input Humidity:** input humidity;
- **Dew Point Temperature:** calculated dew point temperature. Requires valid Input Temperature and Input Humidity to calculate;
- **Enthalpy:** calculated enthalpy. Requires valid Input Temperature and Input Humidity to calculate;
- **Saturated Pressure:** calculated saturated pressure. Requires valid Input Temperature to calculate;
- **Vapor Pressure :** calculated vapor pressure. Requires valid Input Temperature and Input Humidity to calculate;
- **Wet Bulb Temperature:** calculated wet-bulb temperature. Requires valid Input Temperature and Input Humidity to calculate.

### 2.4.5 OutsideAirOptimization

The OutsideAirOptimization component is used to support applications that need to allow for enthalpy based free cooling. This object is typically used during occupancy periods. The Free Cooling output is set to false if outside is greater than or equal to inside and set to true if outside is lesser than or equal to the inside - (abs) Threshold Span. The temperature or enthalpy comparisons can be selected. There is also a check of the low temperature to protect against freezing.



Object Properties	
OutsideAirOptimization [iSMA_control::OutsideAirOptimization]	
Main Links	
Name	Value
✓ Meta	Group1
→ Status	Ok
→ Unit Select	Metric
→ Outside Temp	15.00
→ Outside Humidity	43.00
→ Inside Temp	22.00
→ Inside Humidity	38.00
→ Low Temperature Limit	17.00
→ Outside Enthalpy	0.00
→ Inside Enthalpy	0.00
→ Free Cooling	false
→ Current Mode	LOW_TEMPERATURE
→ Threshold Span	0.00
→ Use Enthalpy	true

Figure 20. OutsideAirOptimization component

## Slots

The OutsideAirOptimization component has the following slots:

- **Status:** shows the component's status;
- **Unit Select:** used to set the units of the Temperature and Humidity properties;
- **Outside Temperature:** input for the current outside air temperature. This input must be valid for this object to function;
- **Outside Humidity:** input for the current outside air humidity. This input must be valid for this object to function;
- **Inside Temperature:** input for the current inside air temperature. This input must be valid for this object to function;
- **Inside Humidity:** input for the current inside air humidity. This input must be valid for this object to function;
- **Low Temperature Limit:** used to provide freeze protection;
- **Outside Enthalpy:** the calculated outside air enthalpy;
- **Inside Enthalpy:** the calculated inside air enthalpy;
- **Free Cooling:** Boolean output value is set to the value of the Free Cooling command when it is determined that free cooling should be used. Otherwise, the value is set to null;
- **Current Mode:** indicates what mode this component is currently in;
  - Available options: Input out of range, Free Cooling, No Free Cooling, Low temperature, Input error;

- **Threshold Span:** the difference between the inside enthalpy and the outside enthalpy must be greater than this value before free cooling will be enabled;
- **Use Enthalpy:** setting this property to true will enable the use of enthalpy for determining if free cooling is available. Otherwise, it will just use outside and inside temperature to decide.

## 2.5 HVAC Components

This section outlines components for HVAC applications.

### 2.5.1 LeadLagCycles

The LeadLagCycles provides a lead-lag control of 2 to 16 loads based upon their accumulated COS (change of state) counts. This component balances the number of change of states cycles of each of the devices. Only one of the controlled devices will be active at a time based on cycle count.

Object Properties

**LeadLagCycles**  
[iSMA\_control::LeadLagCycles]

Main Links

Name	Value
<input checked="" type="checkbox"/> Meta	Group1
<input type="checkbox"/> Status	Ok
<input type="checkbox"/> Out A	false
<input type="checkbox"/> Out B	false
<input type="checkbox"/> Out C	true
<input type="checkbox"/> Out D	false
<input type="checkbox"/> Out E	false
<input type="checkbox"/> Out F	false
<input type="checkbox"/> Out G	false
<input type="checkbox"/> Out H	false
<input type="checkbox"/> Out I	false
<input type="checkbox"/> Out J	false
<input type="checkbox"/> Out K	false
<input type="checkbox"/> Out L	false
<input type="checkbox"/> Out M	false
<input type="checkbox"/> Out N	false
<input type="checkbox"/> Out O	false
<input type="checkbox"/> Out P	false
<input type="checkbox"/> Max Runtime	15
<input type="checkbox"/> In	true
<input type="checkbox"/> Feedback	false
<input type="checkbox"/> Rotate Timer Active	false
<input type="checkbox"/> Number Outputs	4
<input type="checkbox"/> Cycle Count A	10
<input type="checkbox"/> Cycle Count B	20
<input type="checkbox"/> Cycle Count C	0
<input type="checkbox"/> Cycle Count D	0
<input type="checkbox"/> Cycle Count E	0
<input type="checkbox"/> Cycle Count F	0
<input type="checkbox"/> Cycle Count G	0
<input type="checkbox"/> Cycle Count H	0
<input type="checkbox"/> Cycle Count I	0
<input type="checkbox"/> Cycle Count J	0
<input type="checkbox"/> Cycle Count K	0
<input type="checkbox"/> Cycle Count L	0
<input type="checkbox"/> Cycle Count M	0
<input type="checkbox"/> Cycle Count N	0
<input type="checkbox"/> Cycle Count O	0
<input type="checkbox"/> Cycle Count P	0

Figure 21. LeadLagCycles componet

## Slots

The LeadLagCycles component has the following slots:

- **In:** Boolean input that controls whether any control device should be on. If this input is true, one of the outputs will be active based on the cycle count of each controlled device;
- **Number Outputs:** specifies the number of devices (outputs) that are controlled;
- **Max Runtime:** specifies the maximum amount a given output will be true before switching to another output;
- **Feedback:** Boolean input, which provides positive feedback that a controlled device actually has started. Setting this value to true (and not linking) disables the alarm feature;
- **Rotate Timer Active:**
- **Out A-P:** Boolean outputs, each typically linked to the BooleanWritable control point with the DiscreteTotalizerExt. Outputs are typically used to control loads of some type, such as 2 or more pumps;
- **Cycle Count A-P:** integer inputs that are used for cycle count feedback for the corresponding Out A-P. These inputs will typically be linked to the ChangeOfStateCount property of the DiscreteTotalizerExt that is measuring the cycles of the corresponding Out A-P.

### 2.5.2 LeadLagRuntime

The LeadLagRuntime provides a lead-lag control of from 2 to 16 loads based upon their accumulated runtimes (elapsed active time). This component balances the active runtime of each of the devices. Only one of the controlled devices will be active at a time based on runtime.

Object Properties

**LeadLagRuntime**  
[iSMA\_control::LeadLagRuntime]

Main Links

Name	Value
<input checked="" type="checkbox"/> Meta	Group1
<input type="checkbox"/> Status	Ok
<input type="checkbox"/> Out A	true
<input type="checkbox"/> Out B	false
<input type="checkbox"/> Out C	false
<input type="checkbox"/> Out D	false
<input type="checkbox"/> Out E	false
<input type="checkbox"/> Out F	false
<input type="checkbox"/> Out G	false
<input type="checkbox"/> Out H	false
<input type="checkbox"/> Out I	false
<input type="checkbox"/> Out J	false
<input type="checkbox"/> Out K	false
<input type="checkbox"/> Out L	false
<input type="checkbox"/> Out M	false
<input type="checkbox"/> Out N	false
<input type="checkbox"/> Out O	false
<input type="checkbox"/> Out P	false
<input type="checkbox"/> Max Runtime	15
<input type="checkbox"/> In	true
<input type="checkbox"/> Feedback	false
<input type="checkbox"/> Rotate Timer Active	false
<input type="checkbox"/> Number Outputs	4
<input type="checkbox"/> Runtime A	123
<input type="checkbox"/> Runtime B	10
<input type="checkbox"/> Runtime C	0
<input type="checkbox"/> Runtime D	0
<input type="checkbox"/> Runtime E	0
<input type="checkbox"/> Runtime F	0
<input type="checkbox"/> Runtime G	0
<input type="checkbox"/> Runtime H	0
<input type="checkbox"/> Runtime I	0
<input type="checkbox"/> Runtime J	0
<input type="checkbox"/> Runtime K	0
<input type="checkbox"/> Runtime L	0
<input type="checkbox"/> Runtime M	0
<input type="checkbox"/> Runtime N	0
<input type="checkbox"/> Runtime O	0
<input type="checkbox"/> Runtime P	0

Figure 22. LeadLagRuntime component



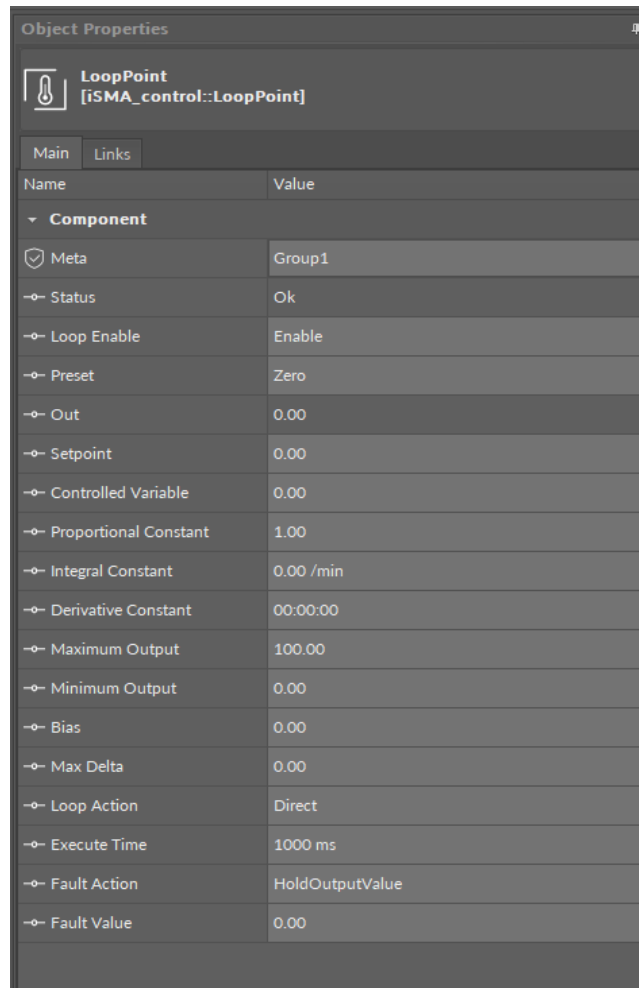
## Slots

The LeadLagRuntimecomponent has the following slots:

- **Status:** shows the component's status;
- **In:** Boolean input that controls whether any control device should be on. If this input is true, one of the outputs will be active based on runtime;
- **Number Outputs:** specifies the number of devices (outputs) that are controlled;
- **Max Runtime:** specifies the maximum amount a given output will be true before switching to another output;
- **Feedback:** Boolean input, which provides positive feedback that a controlled device actually has started. Setting this value to true (and not linking) disables the alarm feature;
- **Rotate Timer Active:**
- **Out A-P:** Boolean outputs, each typically linked to the BooleanWritable control point with the DiscreteTotalizerExt. Outputs are typically used to control loads of some type, such as 2 or more pumps;
- **Runtime A-P:** inputs that are used for runtime feedback for the corresponding Out A-P. These inputs will typically be linked to the ElapsedActiveTime property of the DiscreteTotalizerExt that is measuring the runtime of the corresponding Out A-P.

### 2.5.3 LoopPoint

The LoopPoint component implements a simple PID control loop. Loop objects provide closed-loop PID control (proportional, integral, derivative) at the controller level. Independent gain constants allow the loop to be configured as P-only, PI, or PID.



Name	Value
<b>Component</b>	
Meta	Group1
Status	Ok
Loop Enable	Enable
Preset	Zero
Out	0.00
Setpoint	0.00
Controlled Variable	0.00
Proportional Constant	1.00
Integral Constant	0.00 /min
Derivative Constant	00:00:00
Maximum Output	100.00
Minimum Output	0.00
Bias	0.00
Max Delta	0.00
Loop Action	Direct
Execute Time	1000 ms
Fault Action	HoldOutputValue
Fault Value	0.00

Figure 23. LoopPoint component

## Slots

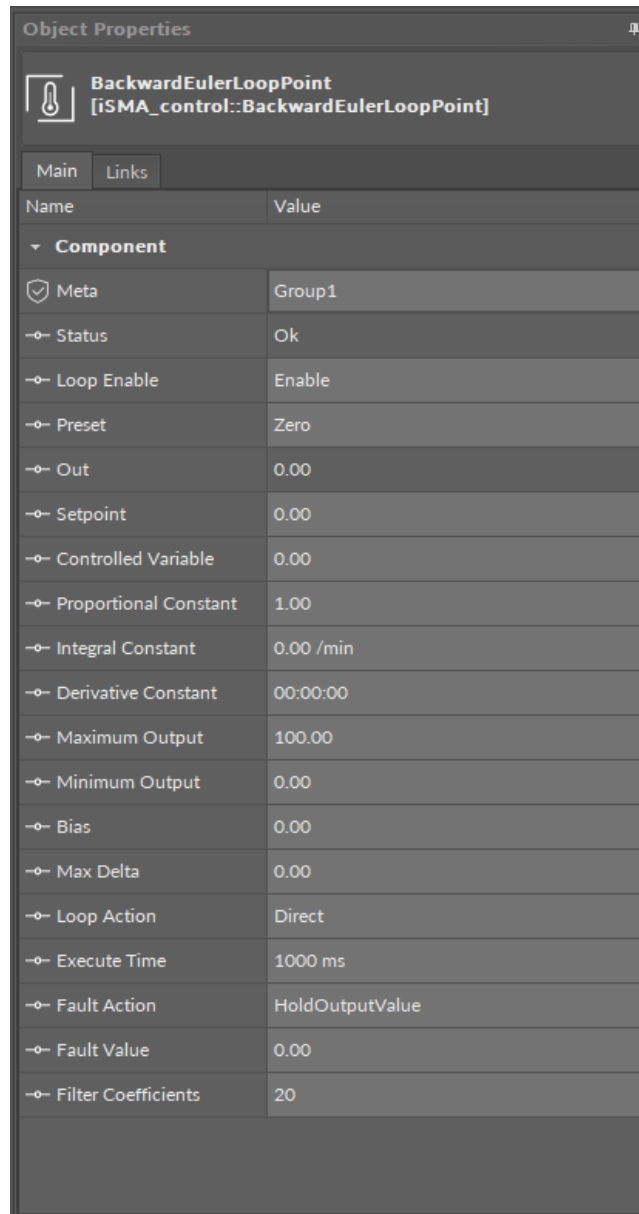
The LoopPoint component has the following slots:

- **Loop Enable:** setting this input to true will enable the PID loop algorithm to execute at the rate selected by the Execute Time property. Setting this input to false will force the PID loop output to a value dependent on the selection in the Preset property.
- **Preset:** if the component is switched from enabled to disabled, the Out value is determined according to the function selected in the Preset slot:
  - **Max:** sets the loop output value to the Maximum Output property value.
  - **Min:** sets the loop output value to the Minimum Output property value.
  - **Zero:** sets the loop output value to a zero (0.0) value.
- **Out:** the result of the loop algorithm.
- **Setpoint:** input for the Setpoint value (for example, space temperature Setpoint). This input must be valid for this object to function.
- **Controlled Variable:** input for the controlled parameter (for example, space temperature). This input must be valid for this object to function.
- **Proportional Constant:** defines the value of the proportional gain parameter used by the loop algorithm. Used to set the overall gain for the loop. A starting point for this value is found by output range/throttling range.
- **Integral Constant:** defines the integral gain parameter, in repeats per minute, used by the loop algorithm. Also, called reset rate. Acts on the magnitude of the Setpoint error. A typical starting point is 0.5.

- **Derivative Constant:** defines the derivative gain parameter, in seconds, used by the loop algorithm. Acts on the rate of change of the Setpoint error.
- **Maximum Output:** defines the maximum output value that the loop algorithm can produce.
- **Minimum Output:** defines the minimum output value that the loop algorithm can produce.
- **Bias:** defines the amount of the output bias added to the output to correct offset error, normally used only with proportional control.
- **Max Delta:** defines the max amount the Out value can change in one period set in the Execute Time slot. Setting to 0 disables this function.
- **Loop Action:** determines whether the control algorithm is direct or reverse acting. Loops setup for direct acting mode increases the loop output as the value of the controlled variable becomes greater than the Setpoint value. In a temperature loop, this is typically considered to be a cooling application. Loops setup for reverse acting mode increases the loop output as the value of the controlled variable becomes less than the Setpoint value. In a temperature loop, this is typically considered to be a heating application.
- **Execute Time:** controls the execution frequency for the PID algorithm, where the default value is 1 second.
- **Fault Action:** defines an action to be invoked once a component goes into the Fault status:
  - **HoldOutputValue:** upholds a last known Out value (from when the status was OK) for the time when the Fault status lasts;
  - **OutputValueOnFault:** the Out value is set to the FaultValue for the time when the Fault status lasts;
  - **MinimumOutputValue:** the Out value is set to the MinimumOutput slot value for the time when the Fault status lasts;
  - **MaximumOutputValue:** the Out value is set to the MaximumOutput slot value for the time when the Fault status lasts;
- **Fault Value:** a value set to the Out slot if the Fault Action is set to the OutputValueOnFault option; must be within limits of the Minimum Output and Maximum Output slots.

## 2.5.4 BackwardEulerLoopPoint

The BackwardEulerLoopPoint component implements a PID control loop based on a backward Euler method. The component uses a simple way to convert integral and derivative components into their discrete-time equivalents using the backward Euler's method.



Object Properties	
BackwardEulerLoopPoint [iSMA_control::BackwardEulerLoopPoint]	
Main Links	
Name	Value
Component	
Meta	Group1
Status	Ok
Loop Enable	Enable
Preset	Zero
Out	0.00
Setpoint	0.00
Controlled Variable	0.00
Proportional Constant	1.00
Integral Constant	0.00 /min
Derivative Constant	00:00:00
Maximum Output	100.00
Minimum Output	0.00
Bias	0.00
Max Delta	0.00
Loop Action	Direct
Execute Time	1000 ms
Fault Action	HoldOutputValue
Fault Value	0.00
Filter Coefficients	20

Figure 24. BackwardEulerLoopPoint component

## Slots

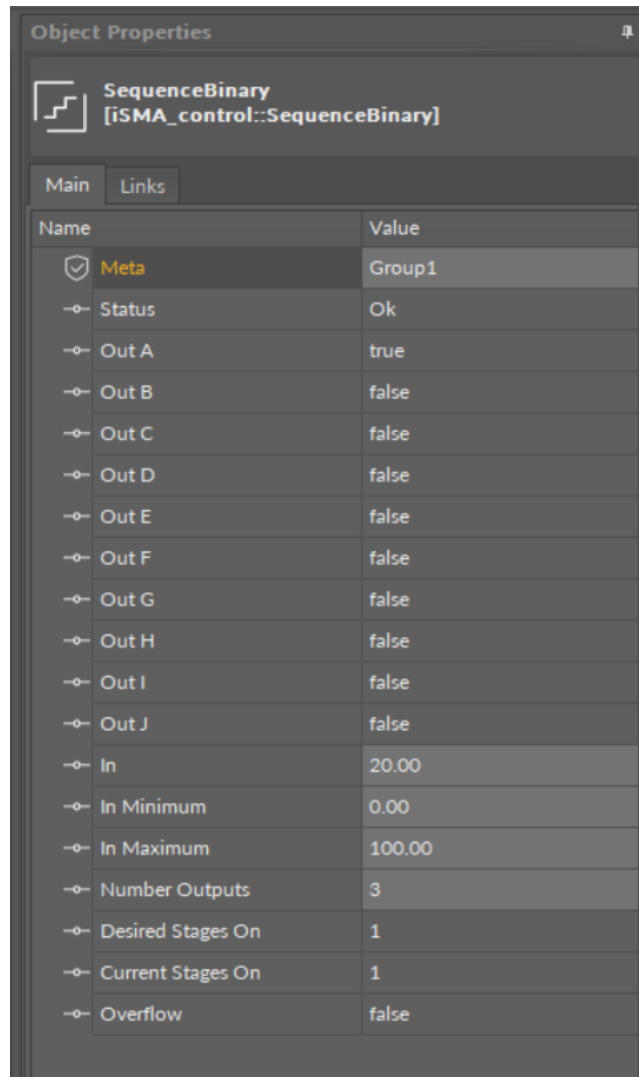
The BackwardEulerLoopPoint component has the following slots:

- **Loop Enable:** setting this input to true will enable the PID loop algorithm to execute at the rate selected by the Execute Time property. Setting this input to false will force the PID loop output to a value dependent on the selection in the Preset property.
- **Preset:** if the component is switched from enabled to disabled, the Out value is determined according to the function selected in the Preset slot:
  - **Max:** sets the loop output value to the Maximum Output property value.
  - **Min:** sets the loop output value to the Minimum Output property value.
  - **Zero:** sets the loop output value to a zero (0.0) value.
- **Out:** the result of the loop algorithm.
- **Setpoint:** input for the Setpoint value (for example, space temperature Setpoint). This input must be valid for this object to function.
- **Controlled Variable:** input for the controlled parameter (for example, space temperature). This input must be valid for this object to function.

- **Proportional Constant:** defines the value of the proportional gain parameter used by the loop algorithm. Used to set the overall gain for the loop. A starting point for this value is found by output range/throttling range.
- **Integral Constant:** defines the integral gain parameter, in repeats per minute, used by the loop algorithm. Also, called reset rate. Acts on the magnitude of the Setpoint error. A typical starting point is 0.5.
- **Derivative Constant:** defines the derivative gain parameter, in seconds, used by the loop algorithm. Acts on the rate of change of the Setpoint error.
- **Maximum Output:** defines the maximum output value that the loop algorithm can produce.
- **Minimum Output:** defines the minimum output value that the loop algorithm can produce.
- **Bias:** defines the amount of the output bias added to the output to correct offset error, normally used only with proportional control.
- **Max Delta:** defines the max amount the Out value can change in one period set in the Execute Time slot. Setting to 0 disables this function.
- **Loop Action:** determines whether the control algorithm is direct or reverse acting. Loops setup for direct acting mode increases the loop output as the value of the controlled variable becomes greater than the Setpoint value. In a temperature loop, this is typically considered to be a cooling application. Loops setup for reverse acting mode increases the loop output as the value of the controlled variable becomes less than the Setpoint value. In a temperature loop, this is typically considered to be a heating application.
- **Execute Time:** controls the execution frequency for the PID algorithm, where the default value is 1 second.
- **Fault Action:** defines an action to be invoked once a component goes into the Fault status:
  - **HoldOutputValue:** upholds a last known Out value (from when the status was OK) for the time when the Fault status lasts;
  - **OutputValueOnFault:** the Out value is set to the FaultValue for the time when the Fault status lasts;
  - **MinimumOutputValue:** the Out value is set to the MinimumOutput slot value for the time when the Fault status lasts;
  - **MaximumOutputValue:** the Out value is set to the MaximumOutput slot value for the time when the Fault status lasts;
- **Fault Value:** a value set to the Out slot if the Fault Action is set to the OutputValueOnFault option; must be within limits of the Minimum Output and Maximum Output slots;
- **Filter Coefficients:** allows to set filter coefficients.

### 2.5.5 SequenceBinary

The SequenceBinary component provides the sequenced weighed staging control of 2 to 10 loads based upon the numeric Input value (0-100). It can be used to support applications that need to sequence 2 to 10 loads or stages in a binary sequence. Binary sequencing provides an analog to binary converter function that selects the outputs whose total load rating relates directly to the control need. For each successive output, the output rating is twice the previous output.



Name	Value
Meta	Group1
Status	Ok
Out A	true
Out B	false
Out C	false
Out D	false
Out E	false
Out F	false
Out G	false
Out H	false
Out I	false
Out J	false
In	20.00
In Minimum	0.00
In Maximum	100.00
Number Outputs	3
Desired Stages On	1
Current Stages On	1
Overflow	false

Figure 25. SequenceBinary component

## Slots

The SequenceBinary component has the following slots:

- **Status:** shows the component's status;
- **In:** input property that is used to determine the number of stages that should currently be on;
- **In Minimum:** value of the input that produces all outputs off;
- **In Maximum:** value of the input that produces all outputs on;
- **Number Outputs:** this object can be configured to support 2 to 10 outputs or stages;
- **OutA - OutJ:** Boolean values that can be used to control 2 to 10 loads. The number of outputs used is defined by the Number Outputs property;
- **Desired Stages On:** read-only property that indicates the calculated number of stages that should be on based on the In property;
- **Current Stages On:** read-only property that indicates the current number of stages that are currently on. Normally the Current Stages On and the Desired Stages On will be the same. They will be different when going through a transition.

Control Signal (In) %	OutC (4kw load size)	OutB (2kw load size)	OutA (1kw load size)	Stage Hysteresis
100	On	On	On	14.3
85,7	On	On	Off	14.3
71,4	On	Off	On	14.3
57,1	On	Off	Off	14.3
42,9	Off	On	On	14.3
28,6	Off	On	Off	14.3
14,3	Off	Off	On	14.3
0	Off	Off	Off	14.3

Table 2. Table illustrates how, by controlling 3 loads, eight unique levels of control can be achieved

### 2.5.6 SequenceFailover

The SequenceFailover component is used to cascade and sequence/stage loads based on the Stage status, Cascade Msg In, and Fail status inputs.

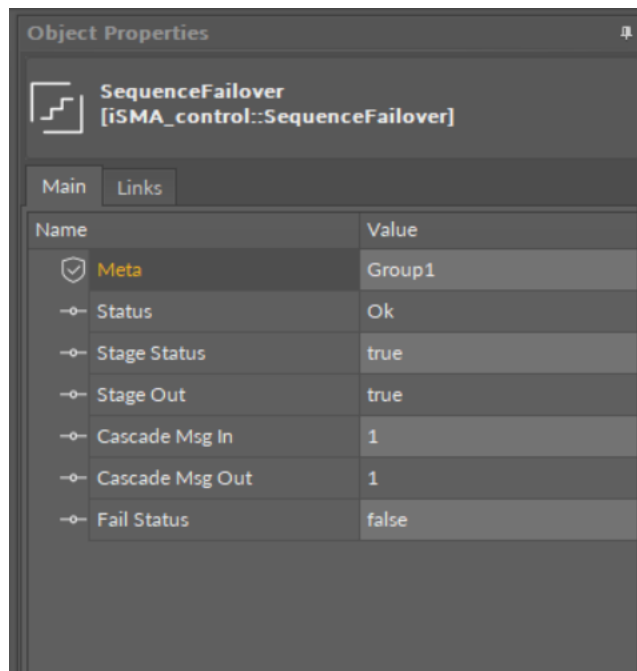


Figure 26. SequenceFailover component

### Slots

The SequenceFailover component has the following slots:

- **Stage Status:** this configuration property defines where the cascade calculation would be performed;

**Stage Status**

If true Msg Out = Msg In.

If false Msg Out = Msg In – 1.

- **Stage Out:** read-only property that indicates Stage Out status;
- **Cascade Msg In:** cascade input value slot;
- **Cascade Msg Out:** out value slot which indicates cascade value after component calculations;
- **Fail Status:** this configuration corrects the Cascade output value.

**Fail Status**

If true Msg Out = Msg In + 1; Stage Out = false.

If false Msg Out = Msg In. Stage Out = Stage Status.

**2.5.7 SequenceLinear**

The SequenceLinear component provides the sequenced rotating (staging) control of 2 to 10 loads based upon the numeric input value (0-100). A similar object is the SequenceBinary, which uses a weighed method (vs. rotating) for sequencing.

The SequenceLinear component can be used to support applications that need to sequence 2 to 10 loads or stages in a linear or rotating sequence. With linear sequencing, the first stage on will be the last stage off. With rotating sequencing the first stage on will be the first stage off. The In property, which is a numeric, is used to control the number of stages that should be on. The input range is defined by the InMinimum and InMaximum properties.

**Note:**

For the input value, the component has a built-in hysteresis of half the delta value, where delta equals (In maximum-In minimum)/number outputs.



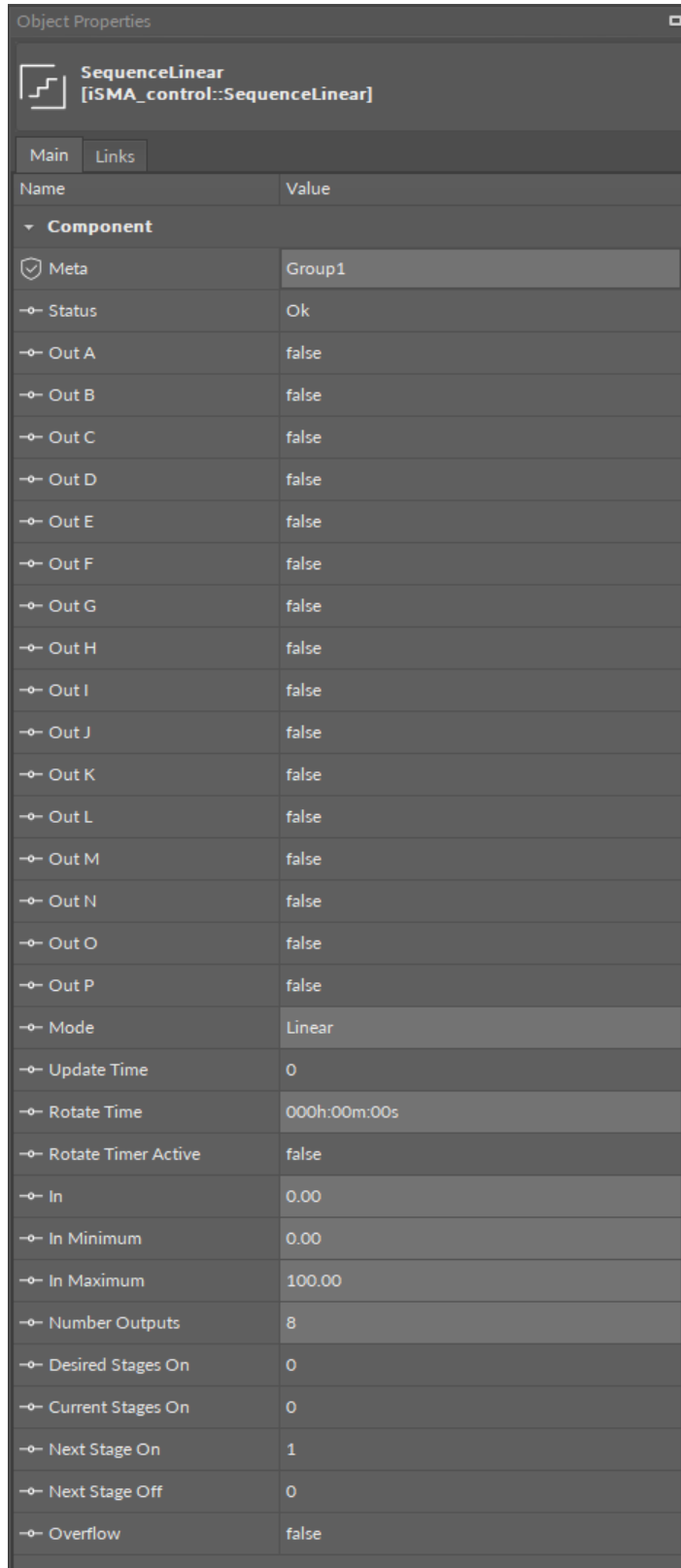


Figure 27. SequenceLinear component

## Slots

The SequenceLinear component has the following slots:

- **Status:** shows the component's status;

- **OutA - OutP:** these are Boolean values that can be used to control 2 to 16 loads. The number of outputs used is defined by the Number Outputs property;
- **Mode:** allows to select between linear and rotating mode of operation;
  - linear mode: stages turn on (true state) linearly, from Out A (stage 1) to Out X, and turn off linearly, from Out X to Out A (Out A is the first to turn on and the last to turn off);
  - rotating mode: stages turn on consecutively, the next one turns on only after the preceding one turns off (Out A turns to true and only after it turns false, Out B turns true, and so on);
- **Update Time:** indicates last update time represented in nanoseconds;
- **Rotate Time:** this configuration property specifies the amount of time that the outputs will remain in fixed configuration before they are shifted to the next configuration;
- **Rotate Timer Active:** read-only property that indicates that the rotate timer is active.
- **In:** input property that is used to determine the number of stages that should currently be on;
- **In Maximum:** value of the input that produces all outputs on;
- **In Minimum:** value of the input that produces all outputs off;
- **Number Outputs:** this object can be configured to support 2 to 16 outputs or stages;
- **Desired Stages On:** read-only property that indicates the calculated number of stages that should be on based on the In property;
- **Current Stages On:** read-only property that indicates the current number of stages that are currently on. Normally the Current Stages on and the Desired Stages On will be the same. They will be different when going through a transition;
- **Next Stage On:** read-only property that indicates the next stage that will be turned on, if needed. This is primarily used when the Mode is selected to be Rotating;
- **Next Stage Off:** read-only property that indicates the next stage that will be turned off, if needed. This is primarily used when the Mode is selected to be Rotating;
- **Overflow:** informs about an overflow state, which occurs if the In slot value is higher than the In Maximum value.

	Linear	Rotating
Range = InMaximum - InMinimum	100 = 100 - 0	100 = 100 - 0
Delta = range/ NumberOutputs	20 = 100 / 5	20 = 100 / 5
OnSetpointA = 1 * delta	20	20
OnSetpointB = 2 * delta	40	40
OnSetpointC = 3 * delta	60	60
OnSetpointD = 4 * delta	80	80
OnSetpointE = 5 * delta	100	100
OffSetpointA = 0 * delta , 4 * delta	0	80
OffSetpointB = 1 * delta, 3 * delta	20	60
OffSetpointC = 2 * delta, 2 * delta	40	40

	Linear	Rotating
OffsetpointD = 3 * delta, 1 * delta	60	20
OffsetpointE = 4 * delta, 0 * delta	80	0

Table 3. SequenceLinear On / Off calculation formulas

## 2.5.8 Thermostat

The Thermostat component executes a basic thermostatic control. The controlled variable is compared with the setpoint value, and the thermostat is switched on or off accordingly (true or false value in the Out slot). The Cut In Offset and Cut Out Offset slots provide a deadband for the controlled variable. The Thermostat component can be set to operate in the heating or cooling mode.

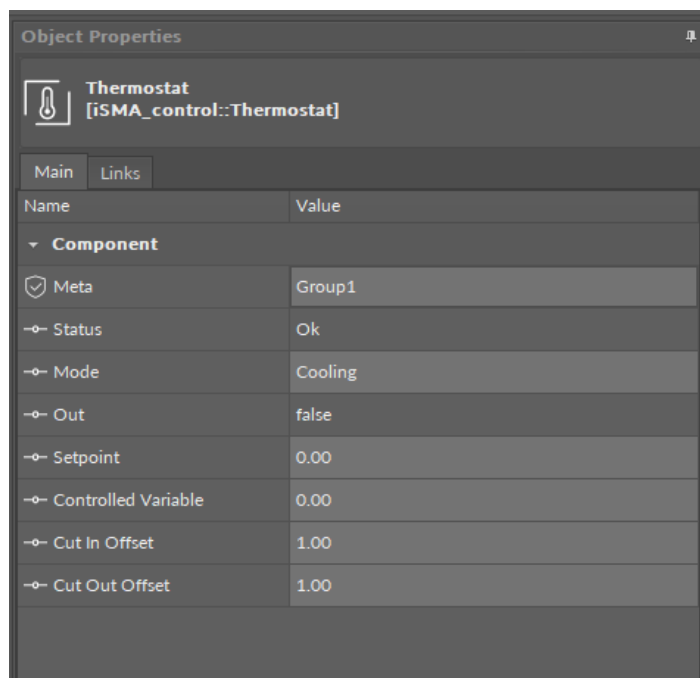


Figure 28. Thermostat component

## Slots

The Thermostat component has the following slots:

- **Mode:** allows to set the thermostat to operate in the heating or cooling mode.
- **Out:** the output value indicating whether the thermostat is switched on (true) or off (false).
- **Setpoint:** the temperature target value.
- **Controlled Variable:** the current input value (measured temperature).
- **Cut In Offset:** defines the differential value between the controlled variable and setpoint to determine the Thermostat output on state. A positive Cut In Offset value means the value is greater than the Setpoint, and a negative CutInOffset value means the value is lower than the Setpoint, during the comparison. For cooling control, use positive value, and negative value for heating control.
- **Cut Out Offset:** defines the differential value between the controlled variable and setpoint to determine the Thermostat output off state. A positive Cut Out Offset value

means greater than the Setpoint, and a negative CutOutOffset value means lower than the Setpoint, during the comparison. For cooling control, use a negative value, and positive value for heating control.

In the heating mode, the thermostat is switched on if the controlled variable is lower than the setpoint plus the Cut In Offset value, switches off when the controlled variable reaches the setpoint plus the Cut In Offset value, and remains switched off until the controlled variable drops below the setpoint minus the Cut Out Offset value.

In the cooling mode, the thermostat is switched on if the controlled value is higher than the setpoint plus the Cut In Offset value, switches off when the controlled variable reaches the setpoint plus the Cut In Offset value, and remains switched off until the controlled variable rises above the setpoint minus the Cut Out Offset value.

**Note:** The above rules apply providing the Cut In Offset and Cut Out Offset values are set as follows:

- Heating mode: the Cut In Offset value is negative, and the Cut Out Offset value is positive;
- Cooling mode: the Cut In Offset value is positive, and the Cut Out Offset value is negative.

## 2.5.9 Tstat

The Tstat component provides a basic thermostatic (on/off) control with the Boolean Out property and numeric inputs for controlled variable (Cv), Setpoint (Sp), and differential (Diff).

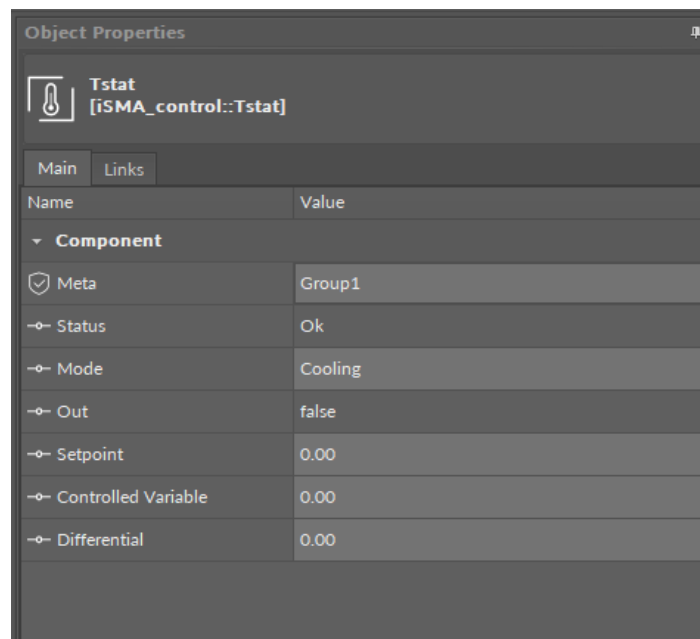


Figure 29. Tstat component

## Slots

The Tstat component has the following slot:

- **Status:** shows the component's status;
- **Mode:** defines the heating or cooling mode of operation;
- **Out:** the input value converted to the float;

- **Setpoint:** the setpoint value (temperature setpoint);
- **Controlled Variable:** the current input value (measured temperature);
- **Differential:** the deadband set for the current input value—if the Setpoint value is, for example, 25, the Differential value set to 5 means that for the Controlled Variable slot values from 22.5 to 27.5 no action is taken.

In the heating mode, the output is switched off if the controlled variable is greater than the setpoint.

In the heating mode, the output is switched on if the controlled variable is lesser than the  $\text{setpoint} - \text{differential} / 2$  (!).

The behavior is similar in the cooling mode.

If the idea is to split the differential value across the setpoint, then the differential value/2 should be applied to both edges:

In the cooling, mode the output is switched off if the controlledVariable is greater than the  $\text{setpoint} + \text{differential} / 2$ .

In the cooling mode, the output is switched on if the controlled variable is lesser than the  $\text{setpoint} - \text{differential} / 2$ .

## 2.6 Latch Components

This section outlines latching components.

### 2.6.1 BooleanLatch

The BooleanLatch component provides a latch for a Boolean input. Any latch that is invoked using the Clock property must include a method for setting the Clock property status back to false before the Clock is available for latching again.

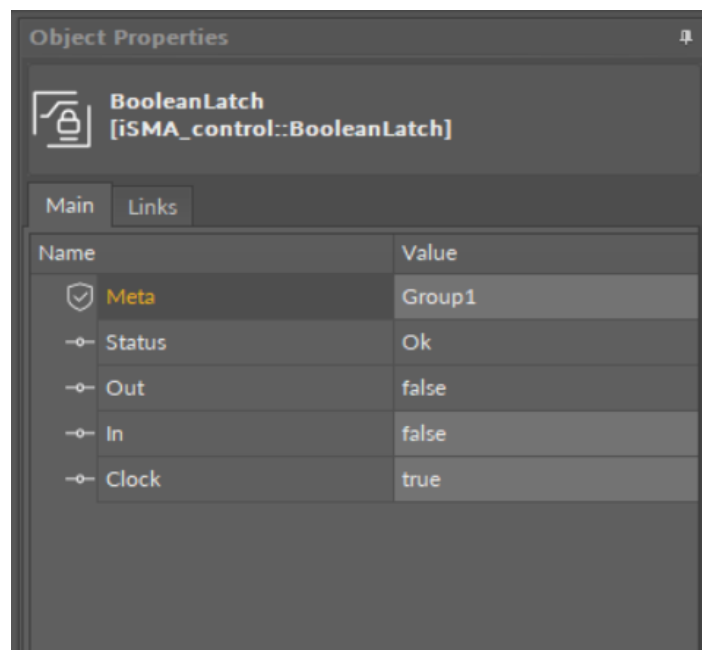


Figure 30. BooleanLatch

### Slots

The BooleanLatch component has the following slots:

- **Status:** shows the component' status;
- **Clock:** the Boolean property that has either a true or false state for all latch components. This property latches the input property to the output property on the rising edge of input. It means that a single input property is captured and sent to the output property at the instant that the Clock status changes from false to true state, and NOT when the property changes from true to false state;
- **Out:** provides the actual latched value that is captured from the input property at the latch time. Link to this property to display the value on a graphic or to process the value with another component;
- **In:** this is the standard component input property that may be linked into from a data source. For example, there may be a link into this property from a control point or the Schedule output.

## 2.6.2 IntegerLatch

The IntegerLatch component provides a latch for an integer input. Any latch that is invoked using the Clock property must include a method for setting the Clock property status back to false before the Clock is available for latching again.

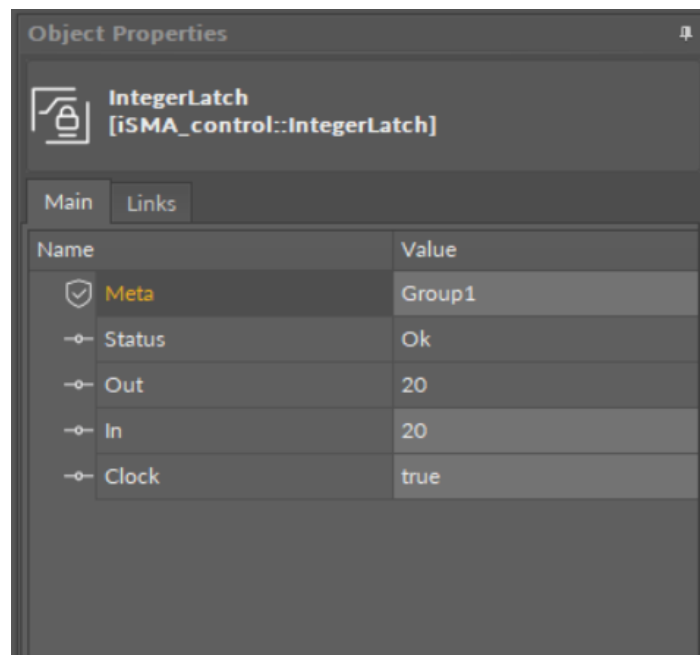


Figure 31. IntegerLatch component

## Slots

The IntegerLatch component has the following slots:

- **Status:** shows the component's status;
- **Clock:** the Boolean property that has either a true or false state for all latch components. This property latches the input property to the output property on the rising edge of input. This means that a single input property is captured and sent to the output property at the instant that the Clock status changes from false to true state, and NOT when the property changes from true to false state;
- **Out:** provides the actual latched value that is captured from the input property at the latch time. Link to this property to display the value on a graphic or to process the value with another component;

- **In:** the standard component input property that may be linked into from a data source. For example, there may be a link to this property from a control point or a Schedule output.

### 2.6.3 NumericLatch

The NumericLatch component provides a latch for a Boolean input. Any latch that is invoked using the Clock property must include a method for setting the Clock property status back to False before the Clock is available for latching again.

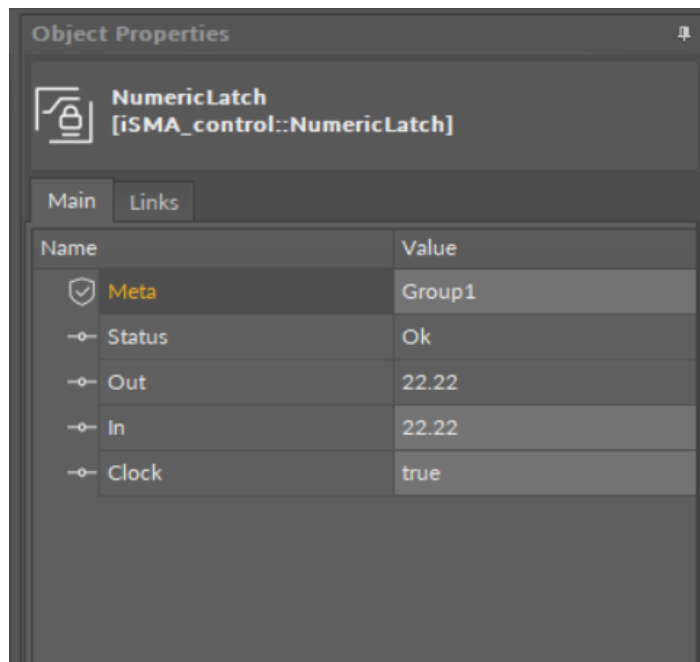


Figure 32. NumericLatch component

### Slots

The NumericLatch component has the following slots:

- **Status:** shows the component's status;
- **Clock:** the Boolean property that has either a true or false state for all latch components. This property latches the input property to the output property on the rising edge of input. This means that a single input property is captured and sent to the output property at the instant that the Clock status changes from false to true state, and NOT when the property changes from true to false state;
- **Out:** provides the actual latched value that is captured from the input property at the latch time. Link to this property to display the value on a graphic or to process the value with another component;
- **In:** this is the standard component input property that may be linked into from a data source. For example, there may be a link into this property from a control point or a Schedule output.

### 2.6.4 SRLatch

The Set/Reset Latch component—a single-bit edge-triggered data storage. The following logic applies to the false-to-true transition of S or R:

- If S goes true and R does not change, then Out is true and remains true.

- If R goes true and S does not change, then Out is false and remains false.
- If both S and R go true on the same scan, then Out is false and remains false.

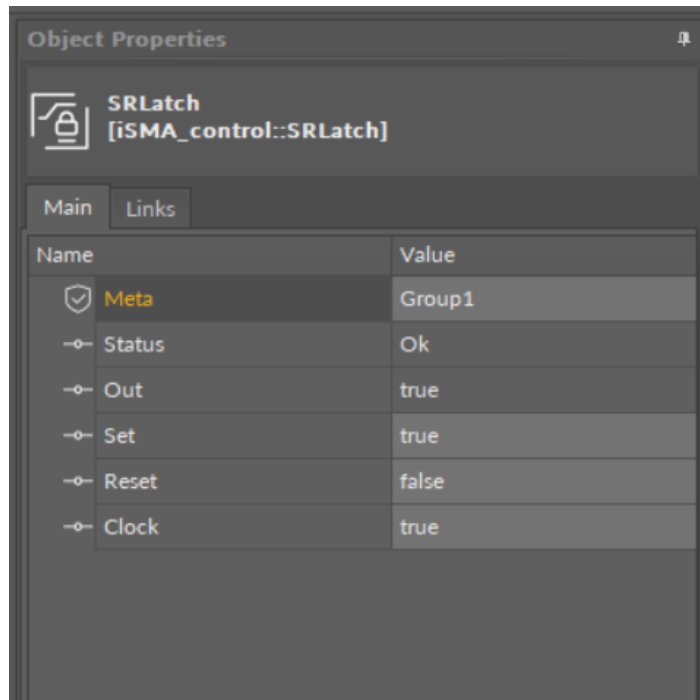


Figure 33. SRLatch component

## Slots

The SRLatch component has the following slots:

- **Status:** shows the component's status;
- **Out:** provides the actual latched value that is captured from the input property at the latch time. Link to this property to display the value on a graphic or to process the value with another component;
- **Set:** the Boolean property determining the Out value, depending on the Reset value;
- **Reset:** the Boolean property determining the Out value, depending on the Set value;
- **Clock:** the Boolean property that has either a true or false state for all latch components. This property latches the input property to the output property on the rising edge of input. This means that a single input property is captured and sent to the output property at the instant that the Clock status changes from false to true state, and NOT when the property changes from true to false state.

## 2.7 Logic Components

Logic components represent logical operations. It consists of components representing logical operations such as conjunction, alternative denial, joint denial, disjunction, or simple comparisons, and more.

### 2.7.1 And

The And component performs a logical AND on all inputs and writes the result to the out property. The first table below shows the AND object truth table if using two inputs. The second table below shows the AND object truth table if using all four inputs.



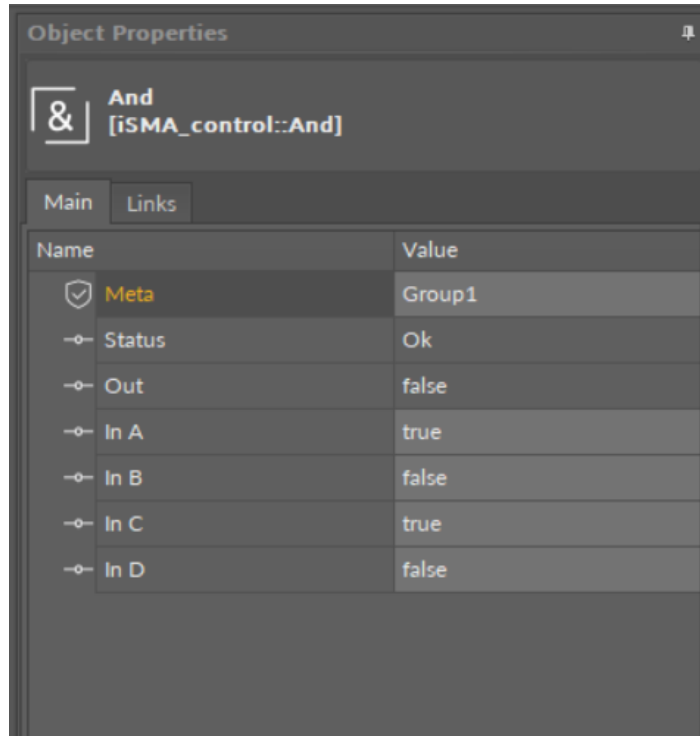


Figure 34. And component

### Slots

The And component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output slot with the outcome of the and operation;
- **InA-InD:** input slots.

In A	In B	Out
False	False	False
False	True	False
True	False	False
True	True	True

Table 4. And component truth table (2 inputs)

In A	In B	In C	In D	Out
False	False	False	False	False
False	False	False	True	False

In A	In B	In C	In D	Out
False	False	True	False	False
False	False	True	True	False
False	True	False	False	False
False	True	False	True	False
False	True	True	False	False
False	True	True	True	False
True	False	False	False	False
True	False	False	True	False
True	False	True	False	False
True	False	True	True	False
True	True	False	False	False
True	True	False	True	False
True	True	True	False	False
True	True	True	True	True

Table 5. And component truth table (4 inputs)

### 2.7.2 LogicExpr

LogicExpr is a binary logic object, where various logic operations are being performed on one/two Boolean inputs based on the operator.

- Out:= (InA & InB) if operator == 0 (And)
- Out:= (InA | InB) if operator == 1 (Or)
- Out:= (InA ^ InB) if operator == 2 (Xor)
- Out:= !InA if operator == 3 (Not)
- Out:= !(InA & InB) if operator == 4 (Nand)
- Out:= !(InA | InB) if operator == 5 (Nor)

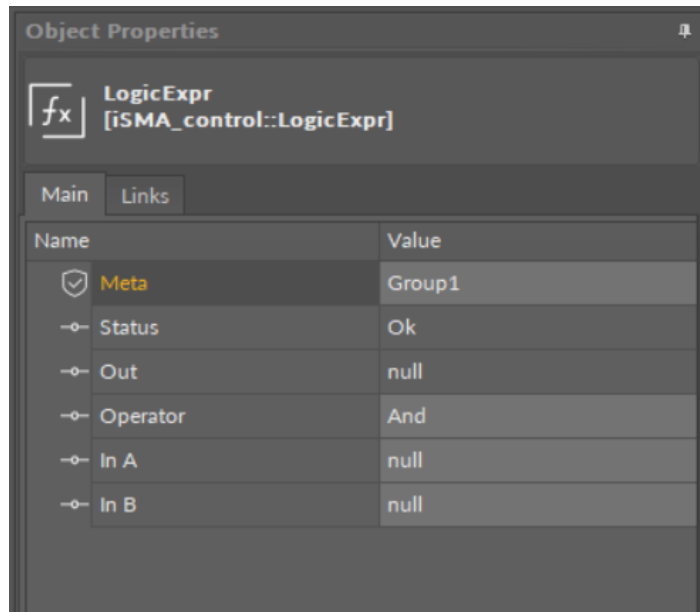


Figure 35. LogicExpr component

### 2.7.3 Or

The Or component performs a logical OR on all valid inputs and writes the Boolean result to the Out property. The first table below shows the OR object truth table if using two inputs. The second table below shows the OR object truth table if using all four inputs. NOR gate logic is accomplished by linking to a Not object.

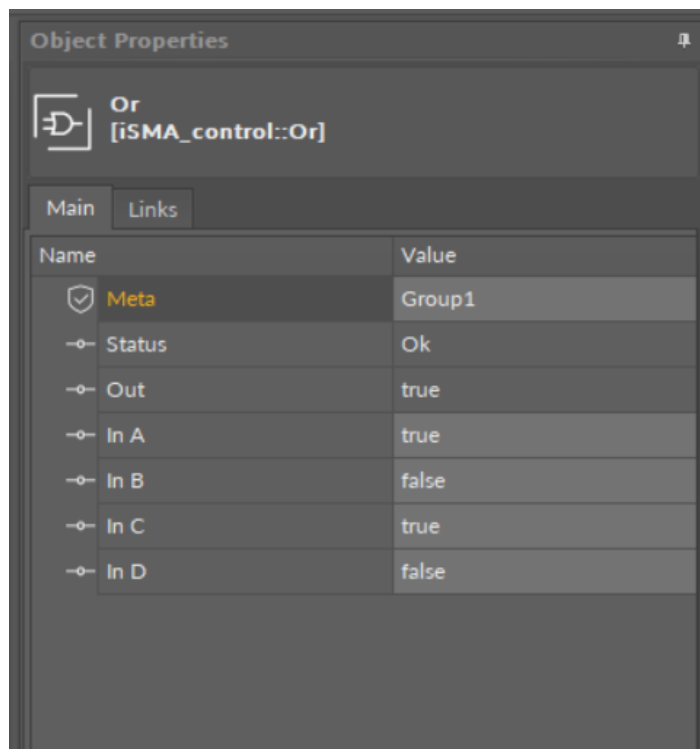


Figure 36. Or component

### Slots

The Or component has the following slots:

- **Status:** shows the component's status;

- **Out:** the output slot showing the outcome of the or operation;
- **InA-InD:** the input values.

In A	In B	Out
False	False	False
False	True	True
True	False	True
True	True	True

Table 6. Or component truth table (2 inputs)

In A	In B	In C	In D	Out
False	False	False	False	False
False	False	False	True	True
False	False	True	False	True
False	False	True	True	True
False	True	False	False	True
False	True	False	True	True
False	True	True	False	True
False	True	True	True	True
True	False	False	False	True
True	False	False	True	True
True	False	True	False	True
True	False	True	True	True
True	True	False	False	True
True	True	False	True	True
True	True	True	False	True
True	True	True	True	True

Table 7. Or component truth table (4 inputs)

### 2.7.4 Nand

The Nand component sets the Out value to false if all inputs are true.

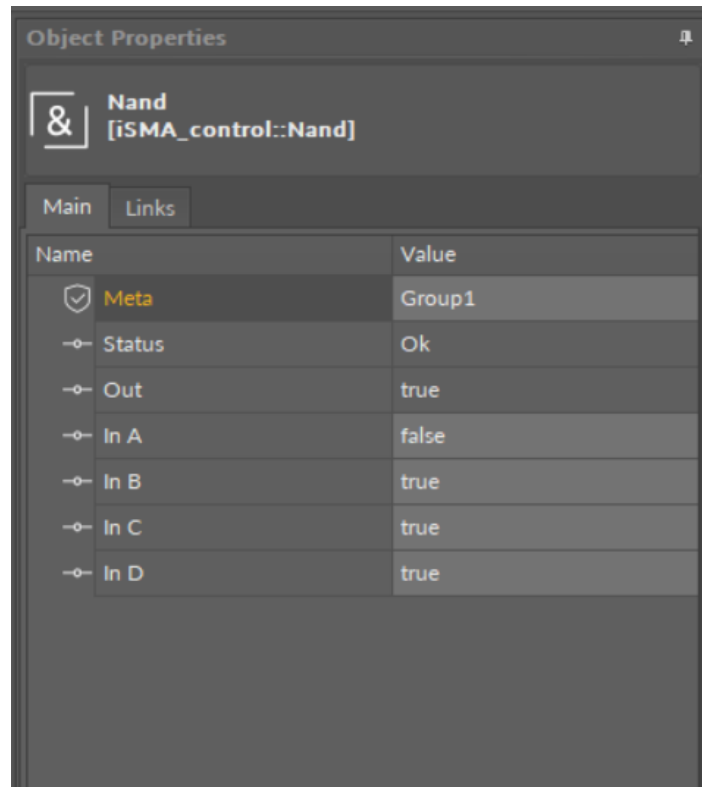


Figure 37. Nand component

## Slots

The Nand component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output slot showing the outcome of the nand operation (false if all inputs are true);
- **InA-InD:** the input values.

### 2.7.5 Nor

The Nor Component set the out value to true if all inputs are false.

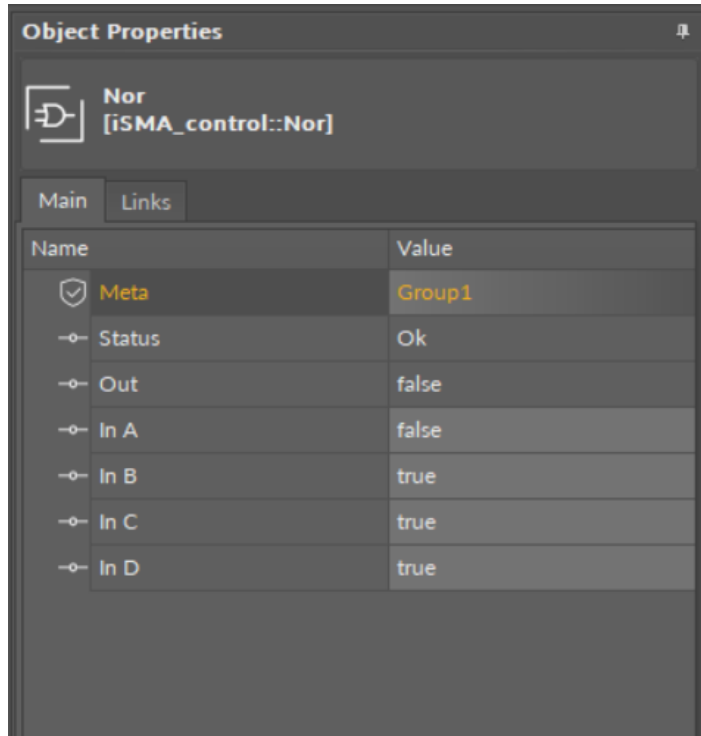


Figure 38. Nor component

## Slots

The Nor component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value with the outcome of the nor operation;
- **InA-In-D:** the input value.

### 2.7.6 Not

The Not component simply inverts the Boolean logic value currently at the (single) object input.

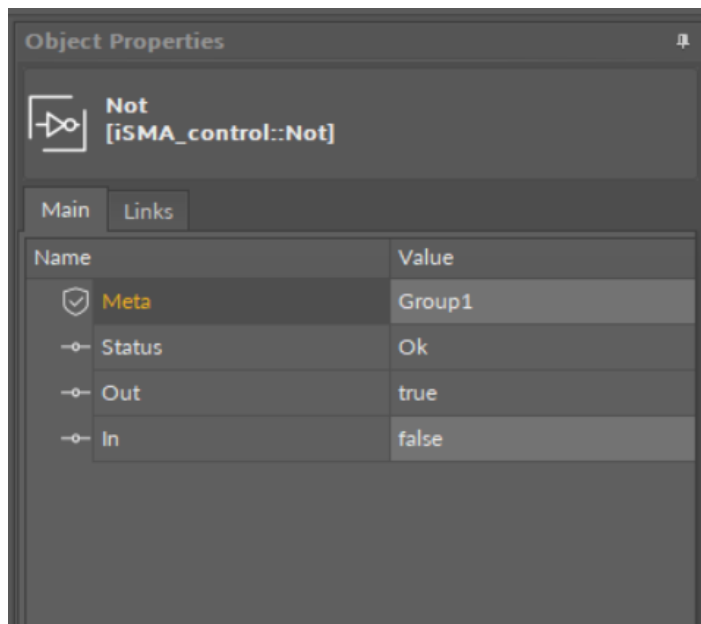


Figure 39. Not component

## Slots

The Not component has the following slots:

- **Status:** shows the component's status;
- **Out:** the inverted input value;
- **In:** the input value.

### 2.7.7 Xor

The XOR component performs a logical XOR on all valid inputs and writes the result to the Out property. The first table below shows the XOR component truth table if using all four inputs. The second table below shows the XOR component truth table if using two inputs (typical).

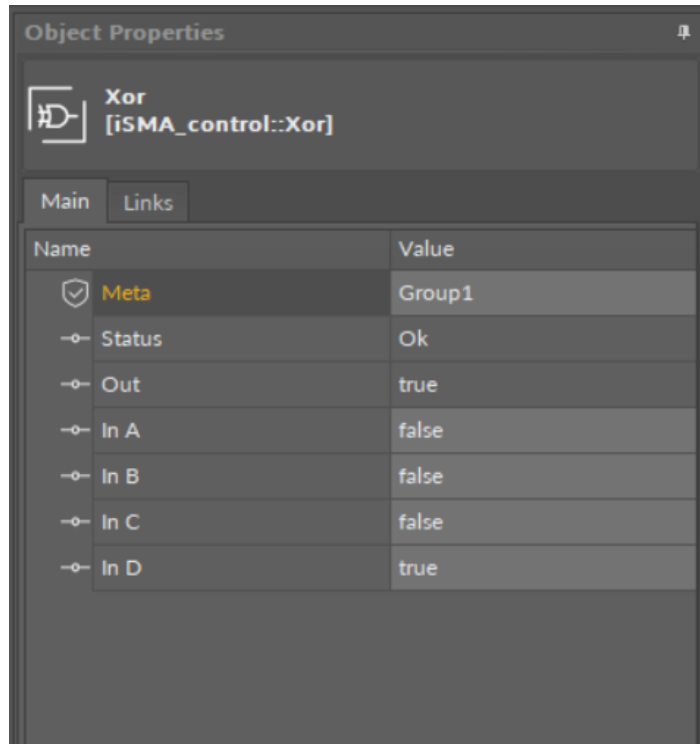


Figure 40. Xor component

## Slots

The Xor component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value with the outcome of the xor operation;
- **InA-InD:** the input value.

In A	In B	Out
False	False	False
False	True	True
True	False	True

In A	In B	Out
True	True	False

Table 8. Xor component truth table (2 inputs)

In A	In B	In C	In D	Out
False	False	False	False	False
False	False	False	True	True
False	False	True	False	True
False	False	True	True	False
False	True	False	False	True
False	True	False	True	False
False	True	True	False	False
False	True	True	True	True
True	False	False	False	True
True	False	False	True	False
True	False	True	False	False
True	False	True	True	True
True	True	False	False	False
True	True	False	True	True
True	True	True	False	True
True	True	True	True	False

Table 9. Xor component truth table (4 inputs)

### 2.7.8 Comparator

The Comparator component performs a numeric comparison of two numeric inputs and raises the respective flags.



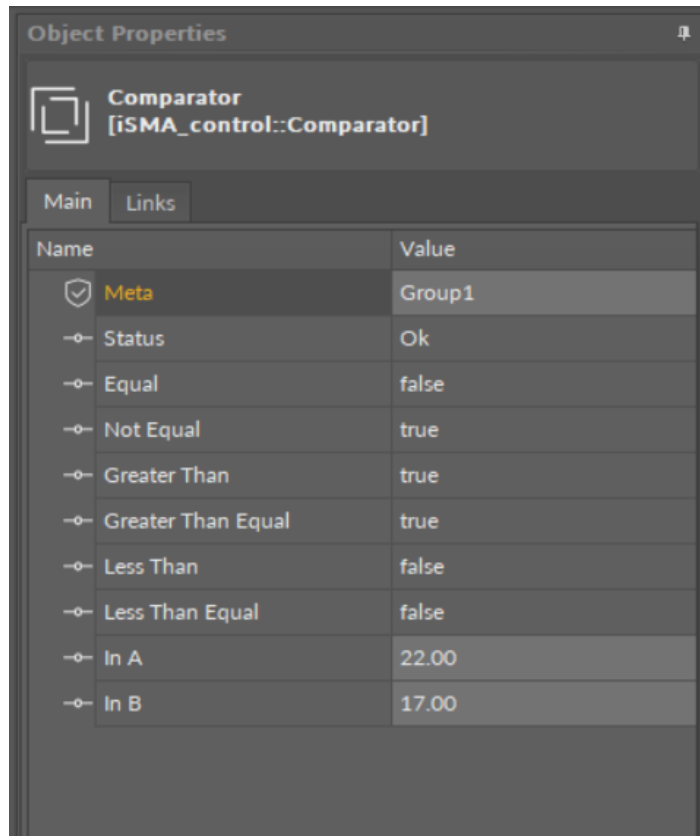


Figure 41. Comparator component

## Slots

The Comparator component has the following slots:

- **Status:** shows the component's status;
- **Equal:** true if the input values are equal;
- **NotEqual:** true if the input values are not equal;
- **GreaterThan:** true if the InA value is greater than the InB value;
- **GreaterThanEqual:** true if the InA value is greater than or equal to the InB value;
- **LessThan:** true if the InA value is less than the InB value;
- **LessThanEqual:** true if the InA value is less than or equal to the InB value;
- **InA-InB:** the input values

### 2.7.9 CompareExpr

The CompareExpr component is a comparator object, where various comparator operations are being performed on two float inputs, based on the operator.

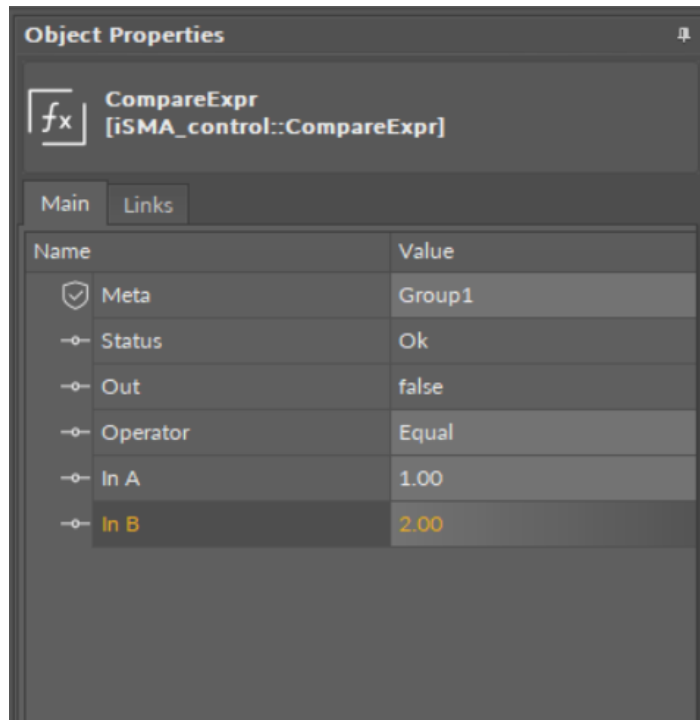


Figure 42. CompareExpr component

## Slots

The CompareExpr component has the following slots:

- **Status:** shows the component's status;
- **Out:** the input value converted to the float;
- **Operator:** the value determining the logical operation for the Out value;
- **InA-InB:** two input values.

Operator	Out Value
0 (Equal)	out:= (inA == inB)
1 (NotEqual)	out:= (inA != inB)
2 (GreaterThan)	out:= (inA > inB)
3 (GreaterThanOrEqual)	out:= (inA >= inB)
4 (LessThan)	out:= (inA < inB)
5 (LessThanOrEqual)	out:= (inA <= inB)

Table 10. CompareExpr component's output value depending on the operator

### 2.7.10 Equal

The Equal component performs the operation  $InA == InB$  for numeric values. Nan values are never equal.

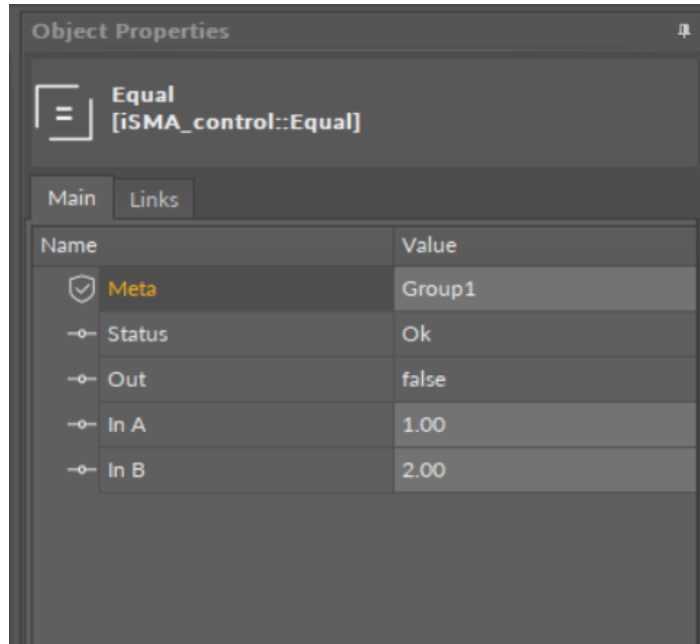


Figure 43. Equal component

## Slots

The Equal component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value indicating whether input values are equal;
- **InA-InB:** two input numeric values.

### 2.7.11 GreaterThan

The GreaterThan component performs the operation  $InA > InB$  with a Boolean result.

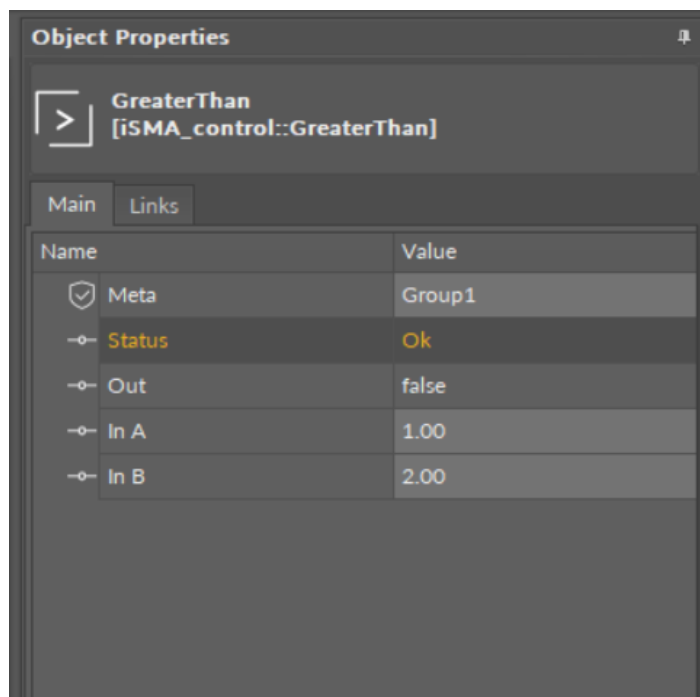


Figure 44. GreaterThan component

## Slots

The GreaterThan component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value indicating whether the InA value is greater than the InB value;
- **InA-InB:** two input values.

### 2.7.12 GreaterThanEqual

The GreaterThanEqual component performs the operation  $InA \geq InB$  with a Boolean result.

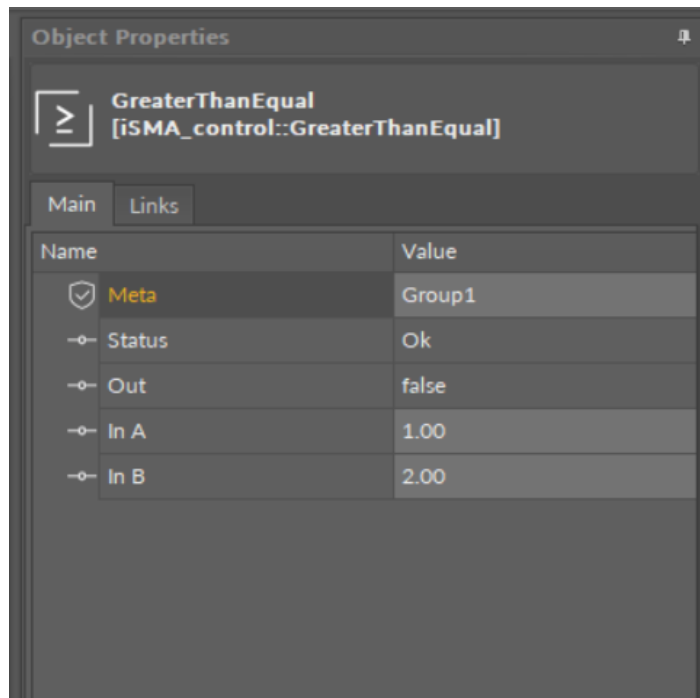


Figure 45. GreaterThanEqual component

## Slots

The GreaterThanEqual component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value indicating whether the InA value is greater or equal to the InB value;
- **InA-InB:** two input values.

### 2.7.13 LessThan

The LessThan component performs the operation  $InA < InB$  with a Boolean result.

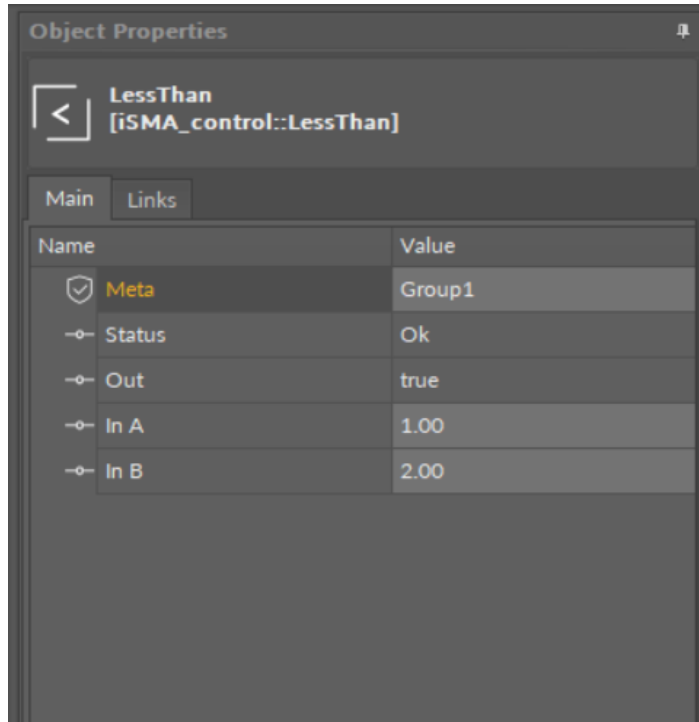


Figure 46. LessThan component

## Slots

The LessThan component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value indicating whether the InA value is less than the InB value;
- **InA-InB:** two input values.

### 2.7.14 LessThanEqual

The LessThanEqual component performs the operation  $InA \leq InB$  with a Boolean result.

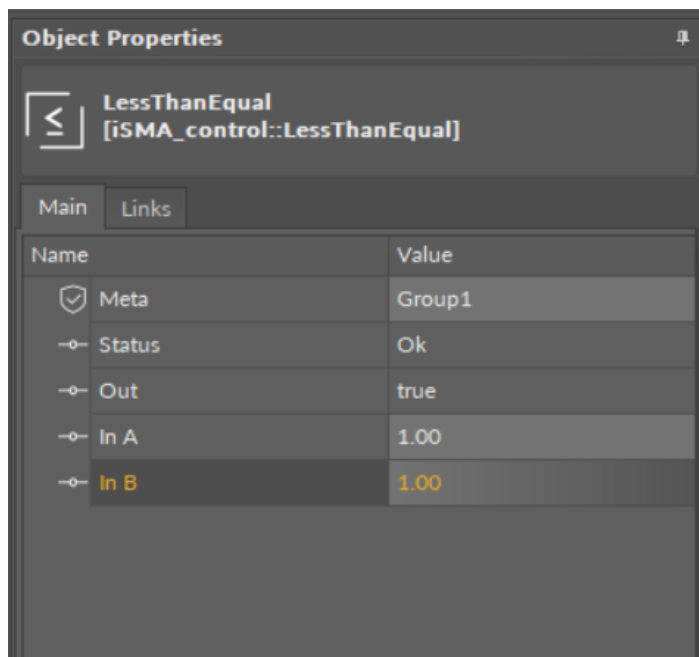


Figure 47. LessThanEqual component

## Slots

The LessThanEqual component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value indicating whether the InA value is less than or equal to the InB value;
- **InA-InB:** two input values.

### 2.7.15 NotEqual

The NotEqual component performs the operation  $InA \neq InB$  with a Boolean result.

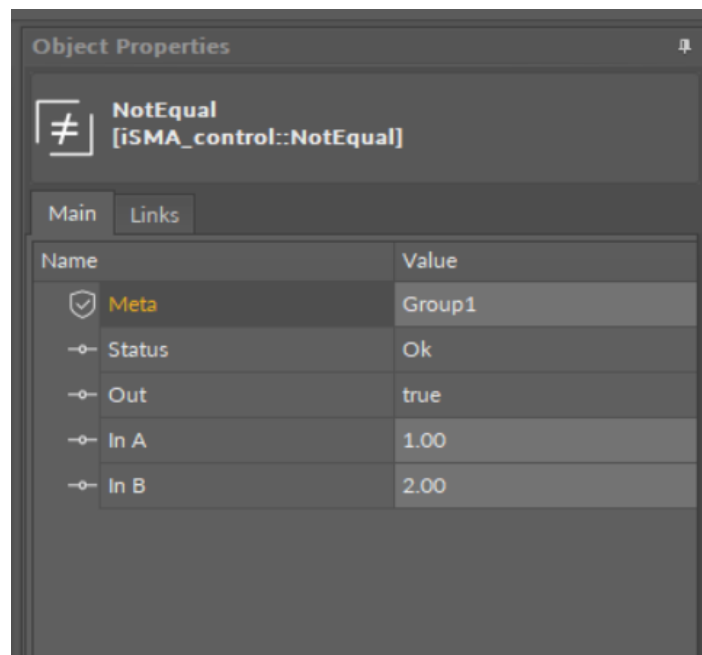


Figure 48. NotEqual component

## Slots

The NotEqual component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value indicating whether two input values are not equal;
- **InA-InB:** two input values.

## 2.8 Math Components

Math components represent mathematical operations.

### 2.8.1 Add

The Add component adds the input values (performs the operation  $out := (InA + InB + InC + InD)$ ).

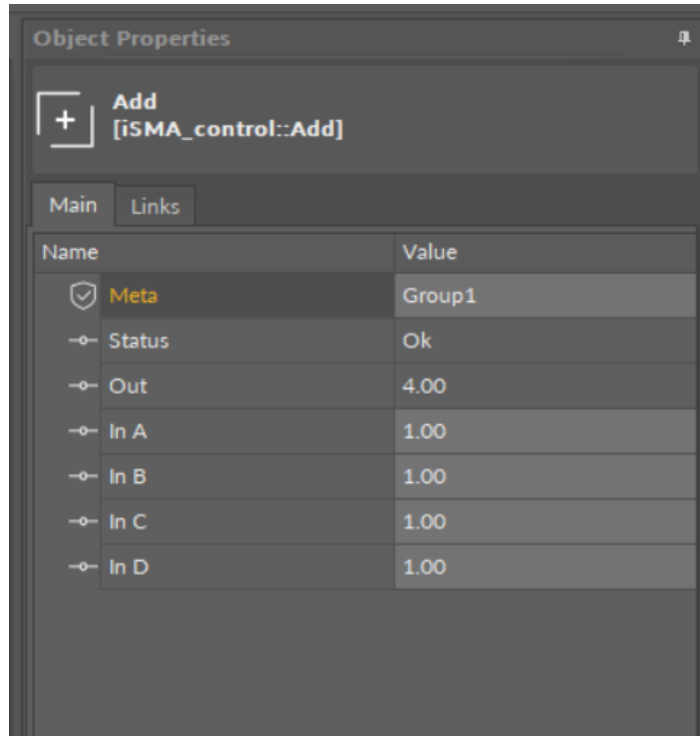


Figure 49. Add component

## Slots

The Add component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output slot with the sum of input values;
- **InA-InD:** the input values.

### 2.8.2 MathExpr

The MathExpr is a component in which various mathematical and trigonometric operations can be performed on one/two numeric inputs, based on the operator.

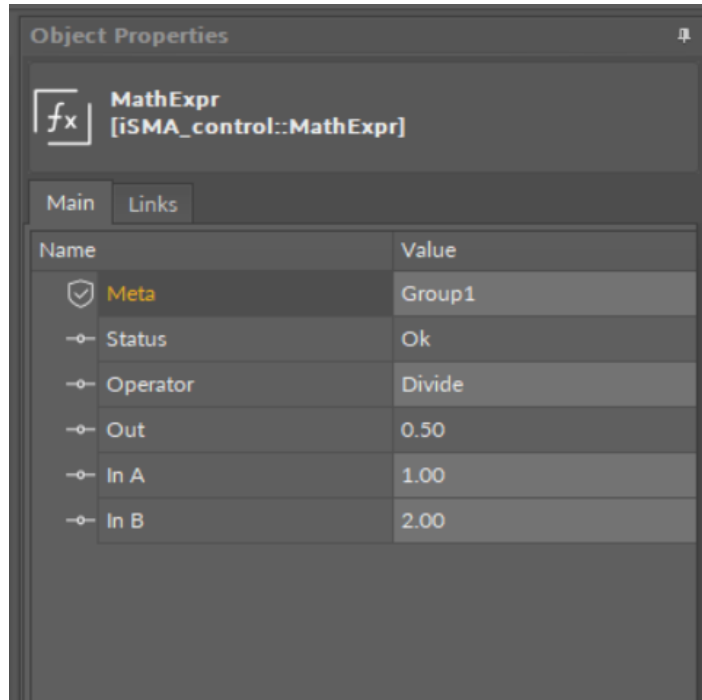


Figure 50. MathExpr component

Operator	Out Value
0 AbsValue	out:= fabs (in)
1 Add	out := inA + inB
2 ArcCosine	out := acos (in)
3 ArcSine	out := asin (in)
4 ArcTangent	out := atan (in)
5 Cosine	out := cos (in)
6 Divide	out := inA / inB
7 Exponential	out := e ^ in
8 Factorial	out := inA!
9 LogBase10	out := log10 (in)
10 LogNatural	out := ln (in)
11 Modulus	out := inA % inB
12 Multiply	out := inA * inB
13 Negative	out := -in
14 Power	out := inA ^ inB
15 Round	out := round (in)



Operator	Out Value
16 Sine	out := sin (in)
17 SquareRoot	out := sqrt (in)
18 Subtract	out := inA - inB
19 Tangent	out := tan (in)
20 Truncate	out := trunc (in)

Table 11. MathExpr operators

### 2.8.3 Maximum

The Maximum component determines the maximum value of valid inputs and writes that value to out.

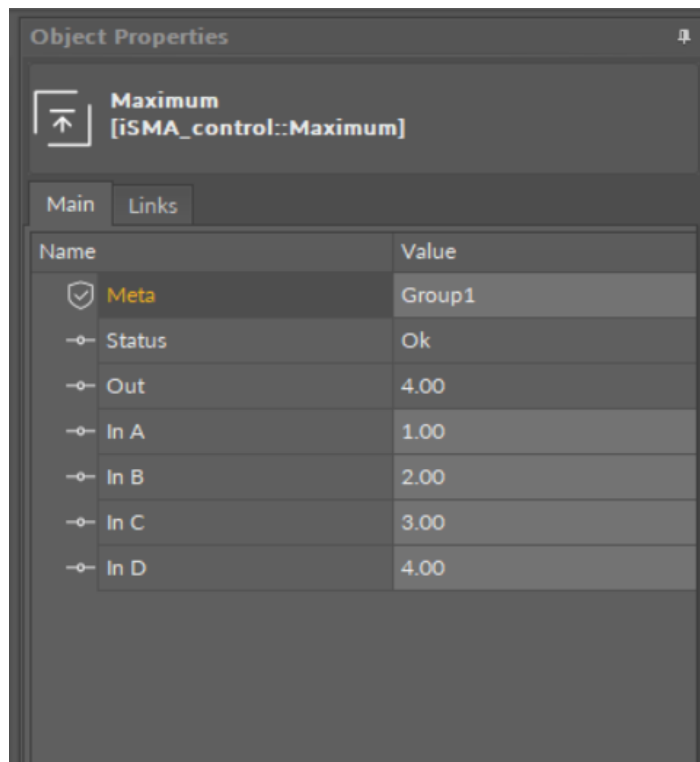


Figure 51. Maximum component

### Slots

The Maximum component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output slot indicating the maximum value of the input values;
- **InA-InD:** the input values.

### 2.8.4 Minimum

The Minimum component determined the minimum value of valid inputs and writes this value to the Out slot.

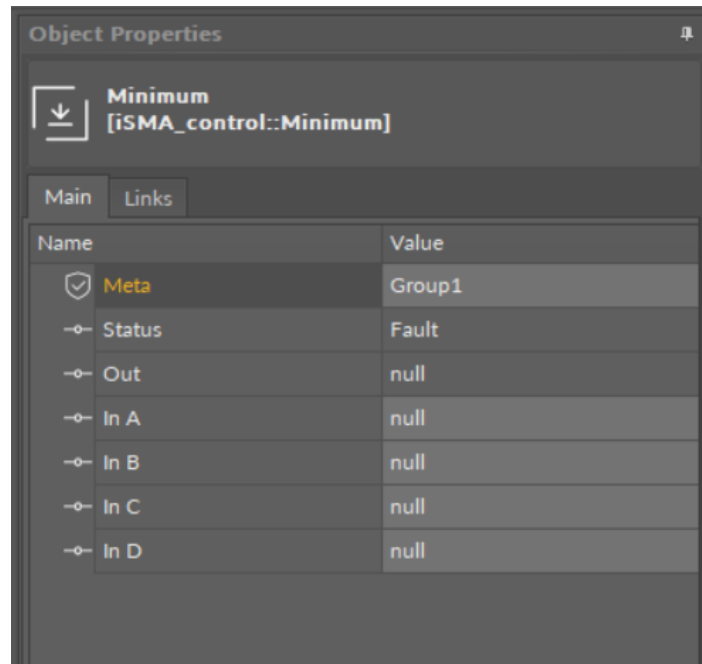


Figure 52. Minimum component

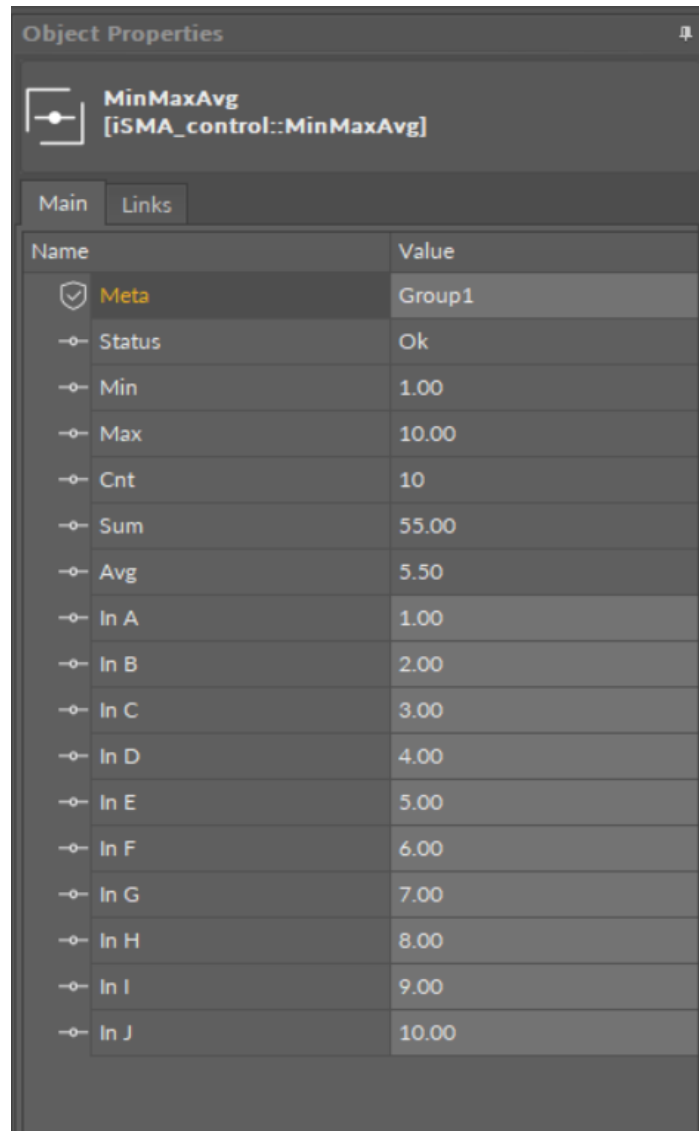
## Slots

The Minimum component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output slot indicating the minimum value of the input values;
- **InA-InD:** the input values.

### 2.8.5 MinMaxAvg

The MinMaxAvg component has 5 numeric output slots that provide the current minimum, maximum, count, sum, and average values of the linked numeric input.



Name	Value
Meta	Group1
Status	Ok
Min	1.00
Max	10.00
Cnt	10
Sum	55.00
Avg	5.50
In A	1.00
In B	2.00
In C	3.00
In D	4.00
In E	5.00
In F	6.00
In G	7.00
In H	8.00
In I	9.00
In J	10.00

Figure 53. MinMaxAvg component

## Slots

The MinMaxAvg component has the following slots:

- **Status:** shows the component's status;
- **Min:** the output indicating the minimum value of the input values;
- **Max:** the output indicating the maximum value of the input values;
- **Cnt:** the output indicating how many input values are active;
- **Sum:** the output indicating the sum of input values;
- **Avg:** the output indicating the average value of the input values;
- **InA-In-J:** the input values.

### 2.8.6 Multiply

The Multiply component multiplies the input values (performs the calculation  $Out := InA * InB * InC * InD$ ).

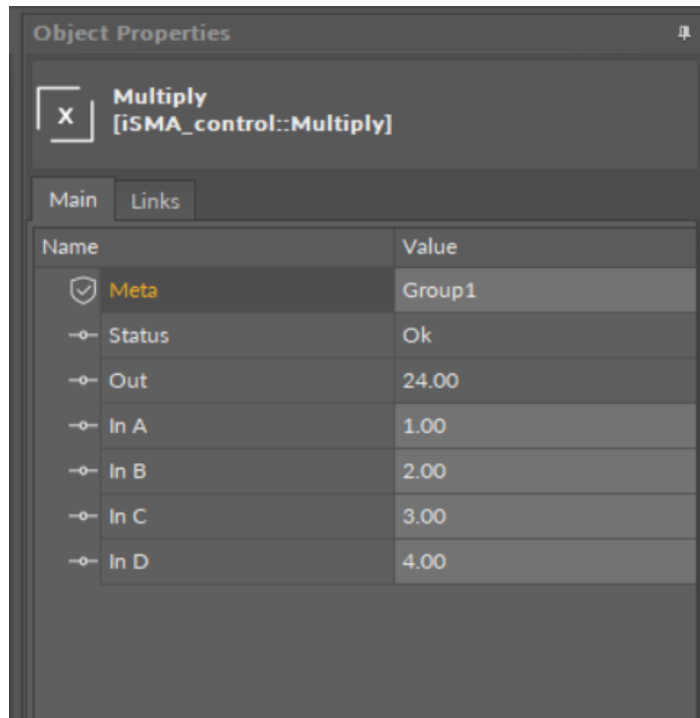


Figure 54. Multiply component

## Slots

The Multiply component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the outcome of multiplying the input values;
- **InA-InD:** the input values.

### 2.8.7 Divide

The Divide component divides the input values (performs the operation  $out := in A / in B$ ). If either input is numeric.null, the output will be numeric.null.

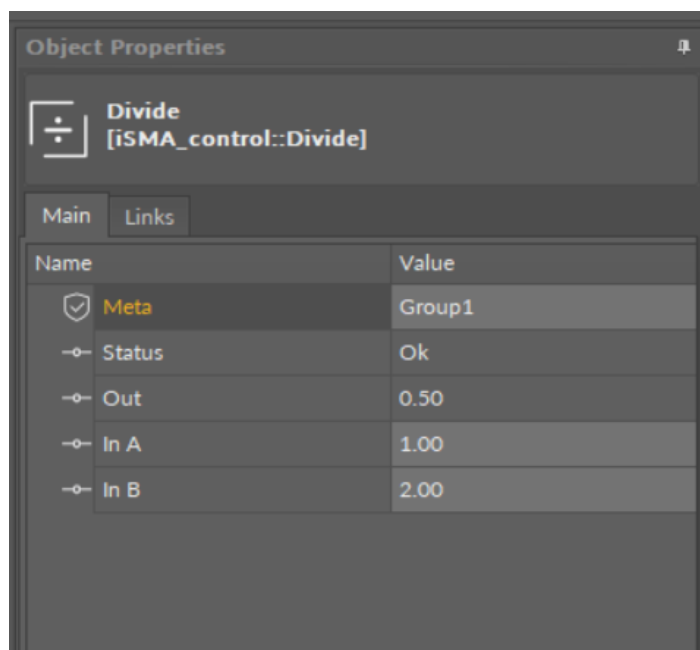


Figure 55. Divide component

## Slots

The Divide component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the outcome of dividing the input values;
- **InA-InB:** the input value.

### 2.8.8 Modulus

The Modulus component provides a modulus operation based on values at its two numeric inputs. The output is the remainder of dividing the InA value by the InB value. If the InB value is 0, the output is NaN (not a number).

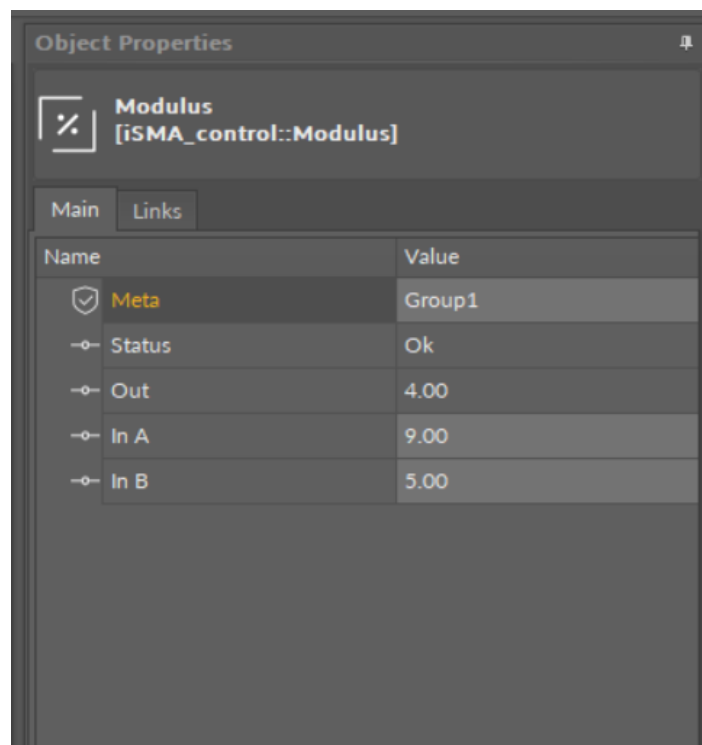


Figure 56. Modulus component

## Slots

The Modulus component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value indicating the remainder of the dividing of input values;
- **InA-InB:** the input values.

### 2.8.9 Power

The Power component raises the InA to the power of InB (performs the operation  $out := InA \wedge InB$ ).

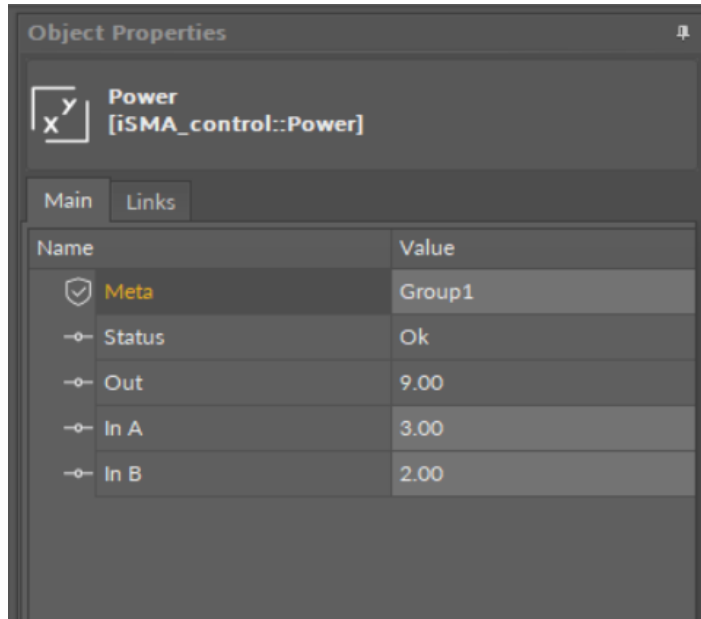


Figure 57. Power component

## Slots

The Power component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the outcome of raising the InA value to the power of InB;
- **InA-InB:** two input values.

### 2.8.10 Subtract

The Subtract component performs the operation  $out := (InA - InB)$ . If either input is numeric.NaN, the output will be numeric.NaN.

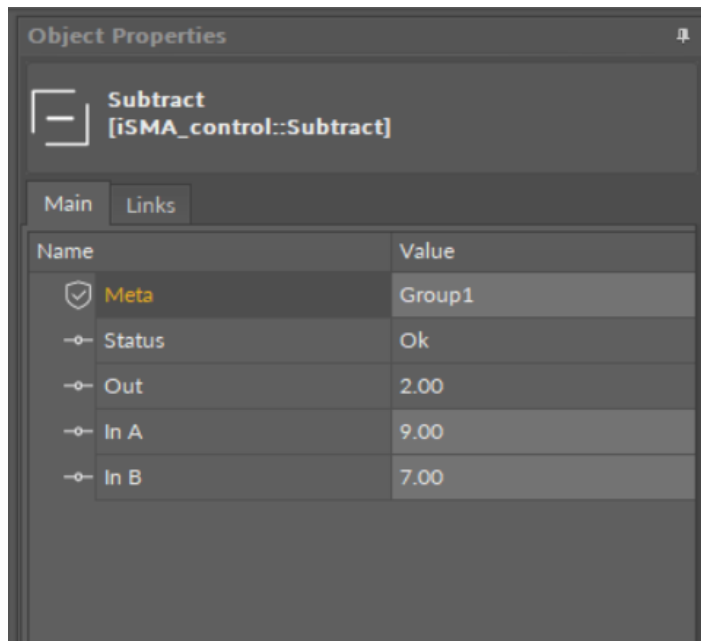


Figure 58. Subtract component

## Slots

The Subtract component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value with the outcome of the input values subtract;
- **InA-InB:** two input values.

### 2.8.11 AbsValue

The AbsValue component returns the absolute value of In (performs the operation  $out := abs(In)$ ).

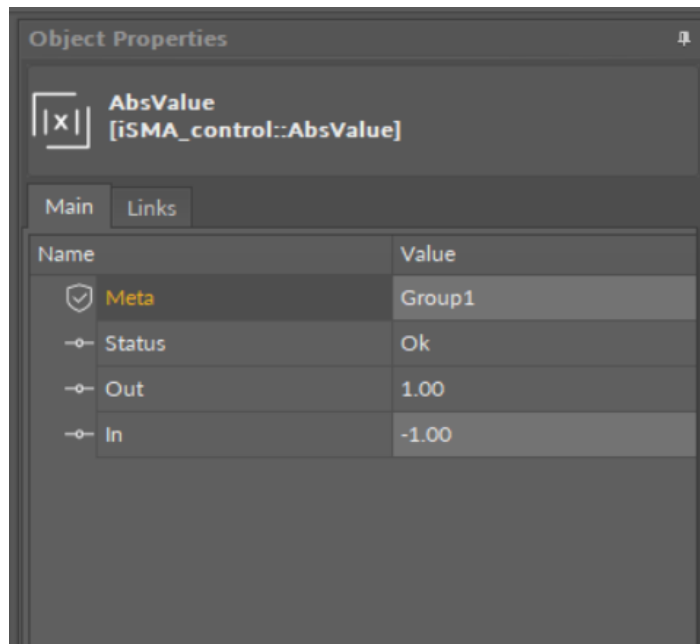


Figure 59. AbsValue component

## Slots

The AbsValue component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the absolute value of the input;
- **In:** the input value.

### 2.8.12 ArcCosine

The ArcCosine component returns the arccosine value of the input (performs the operation  $out := acos(inA)$ ).

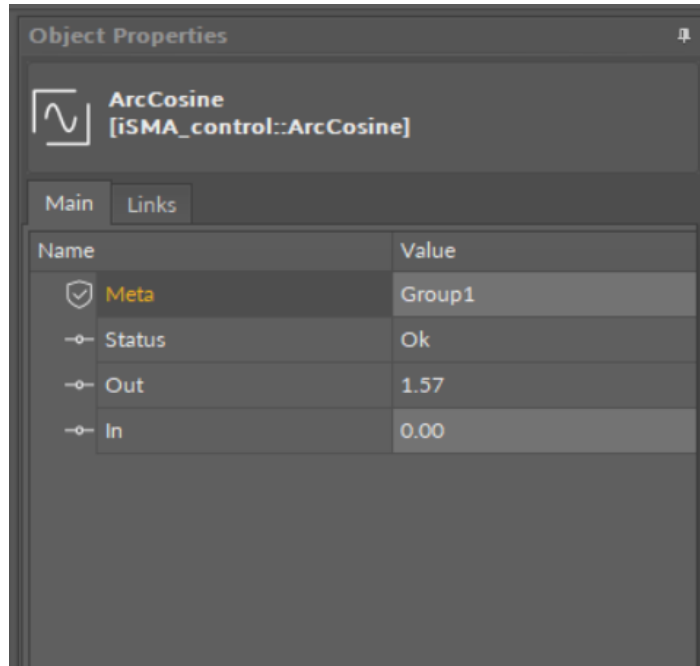


Figure 60. ArcCosine component

## Slots

The ArcCosine component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the arccosine value of the input;
- **InA:** the input value.

### 2.8.13 ArcSine

The ArcSine component returns the arcsine value of the input (performs the operation  $out := asin(inA)$ ).

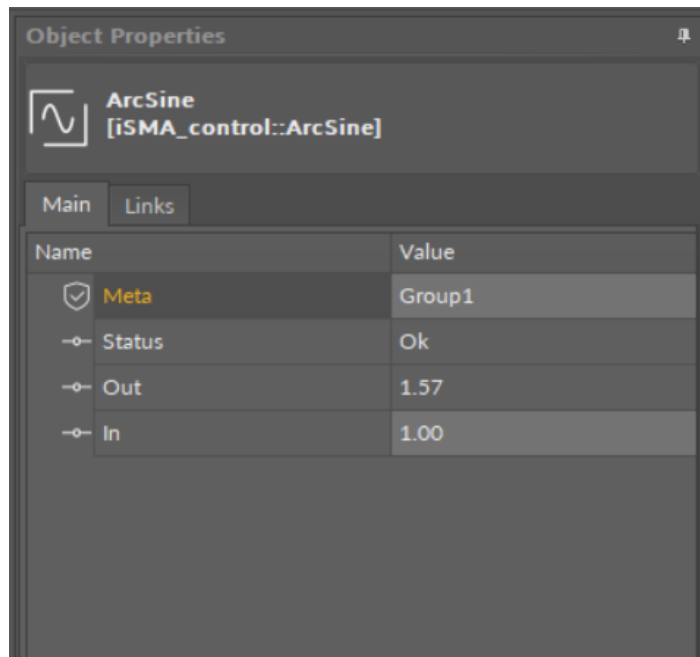


Figure 61. ArcSine component



## Slots

The ArcSine component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the arcsine value of the input;
- **In:** the input value.

### 2.8.14 ArcTangent

The ArcTangent component returns the arctangent value of the input (performs the operation  $out := \text{atan}(inA)$ ).

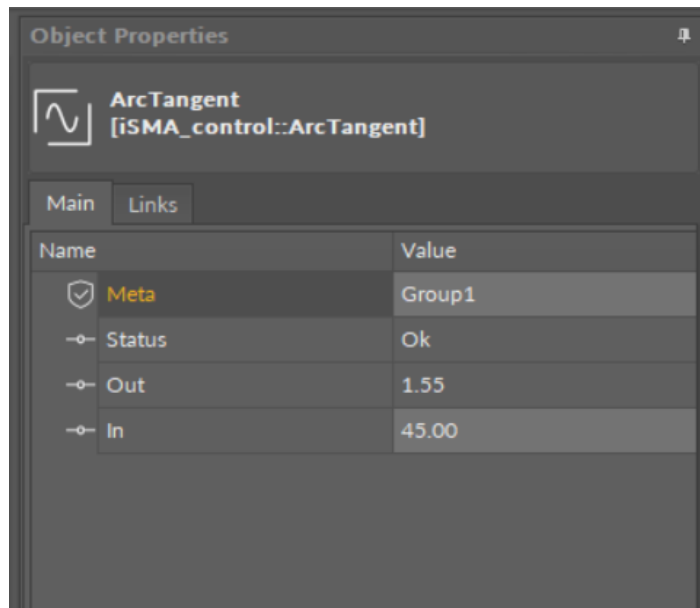


Figure 62. ArcTangent component

## Slots

The ArcTangent component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the arctangent value of the input;
- **In:** the input value.

### 2.8.15 Cosine

The Cosine component returns the cosine value of the input (performs the operation  $out := \cos(in A)$ ).

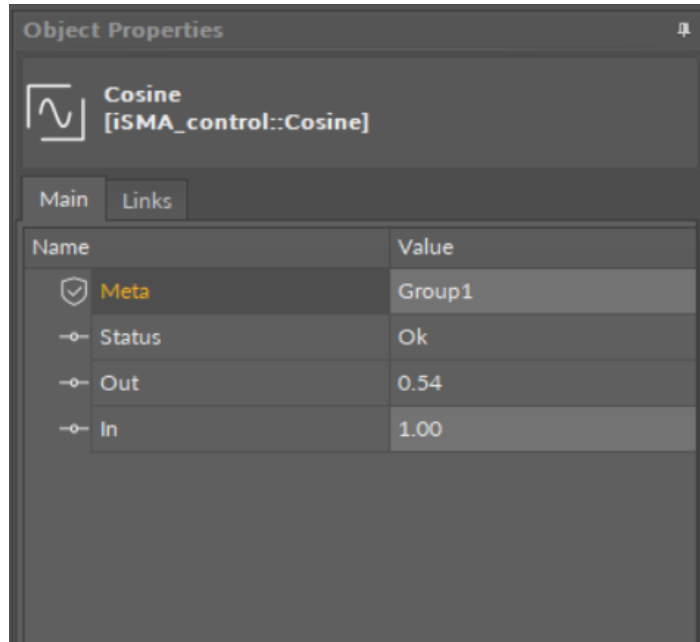


Figure 63. Cosine component

## Slots

The Cosine component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the cosine value of the input;
- **In:** the input value.

### 2.8.16 Exponential

The Exponential component raises the e number the power of InA (performs the operation  $out := e^{InA}$ ).

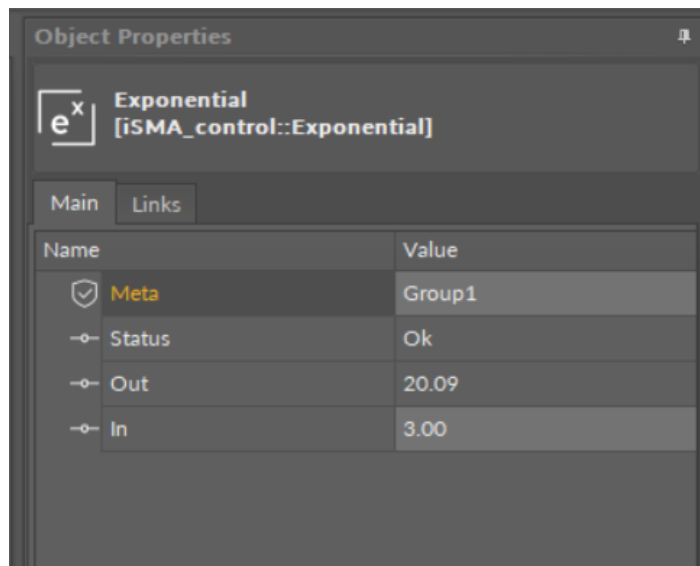


Figure 64. Exponential component

## Slots

The Exponential component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value with the outcome of the e number raised to the power of input;
- **In:** the input value.

### 2.8.17 Factorial

The Factorial component provides a factorial math output, based on the numeric input. Only the integer part of the input value is evaluated, for example, the values of 1.03 or 1.9999 are evaluated as 1.

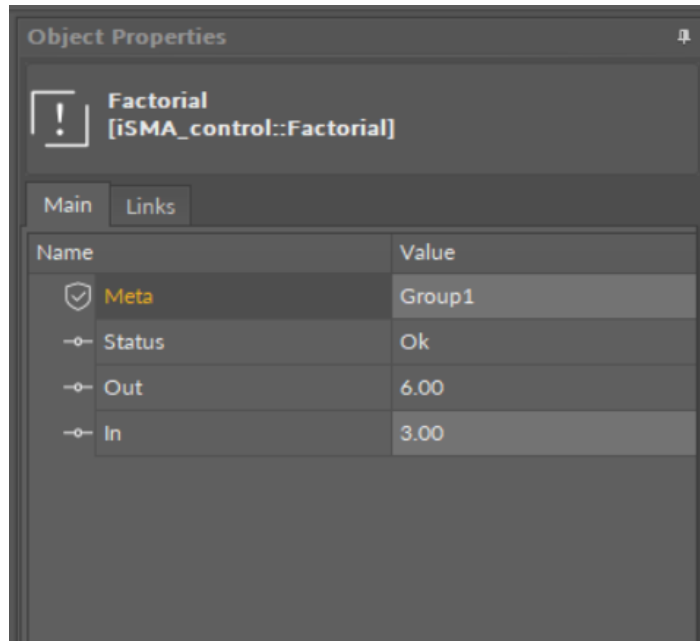


Figure 65. Factorial component

### Slots

The Factorial component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the factorial outcome of the input;
- **In:** the input value.

### 2.8.18 LogBase10

The LogBase10 component returns a common logarithm (a logarithm to the base 10) of the In value (performs the operation  $out := \log_{10}(inA)$ ).

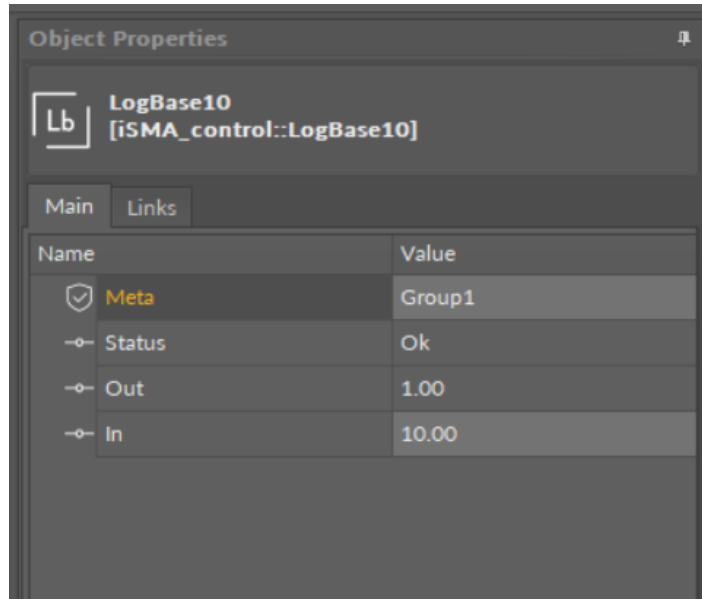


Figure 66. LogBase10 component

## Slots

The LogBase10 component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the common logarithm of the In value;
- **In:** the input value.

### 2.8.19 LogNatural

The LogNatural component returns a natural logarithm (a logarithm to the base of the mathematical constant e) of the In slot value (performs the operation  $out := \ln(inA)$  (log base e of inA)).

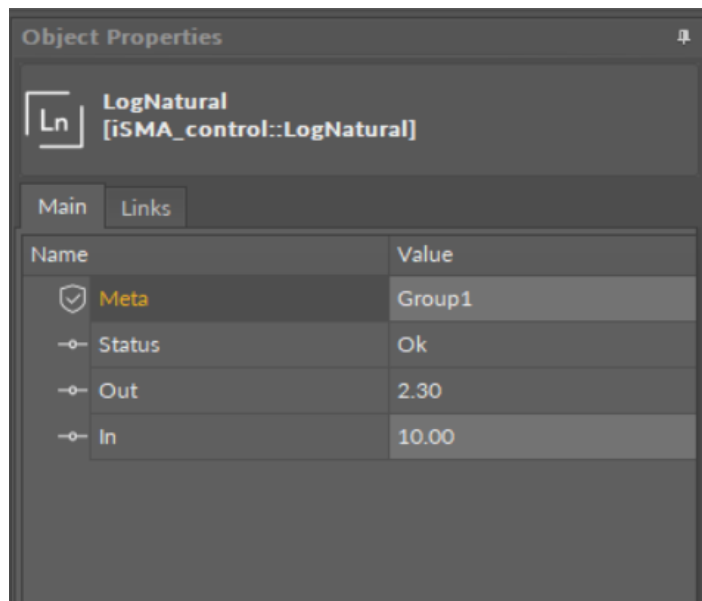


Figure 67. LogNatural component

## Slots

The LogNatural component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the natural logarithm of the input;
- **In:** the input value.

## 2.8.20 MinMaxAverage

The MinMaxAverage component has 5 numeric output slots that provide the current minimum, maximum, count, sum, and average values of a linked numeric input. In the Count slot, the component adds the number of value changes in the In slot. The Reset action sets all slots to 0.

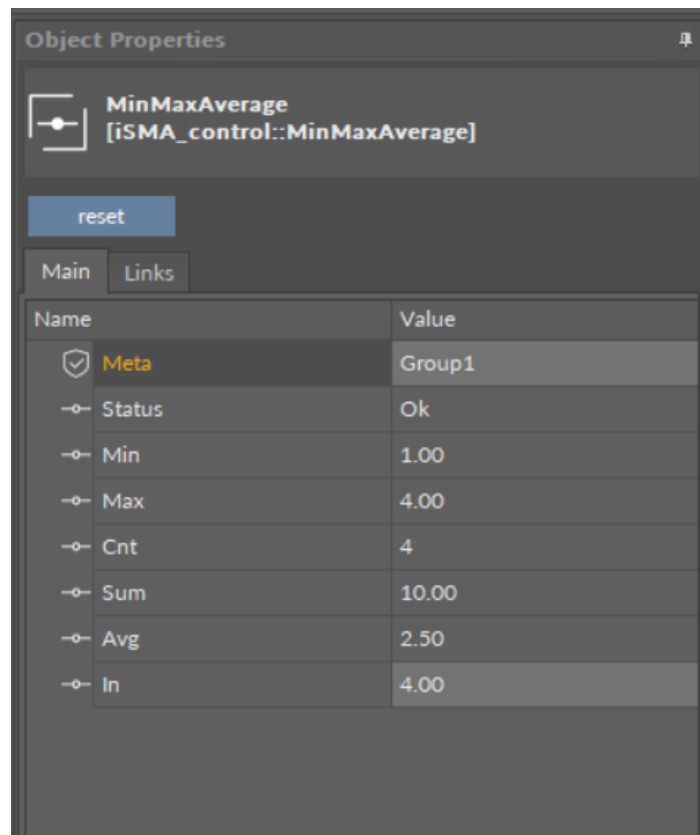


Figure 68. MinMaxAverage component

## Slots

The MinMaxAverage component has the following slots:

- **Status:** shows the component's status;
- **Min:** indicates the minimum value, which has been given to the In slot in the ongoing count;
- **Max:** indicates the maximum value, which has been given to the In slot in the ongoing count;
- **Cnt:** shows the number of value changes to the In slot; counting (together with other calculated values) can be reset and restarted with the Reset action;
- **Sum:** shows the sum of values given to the In slot in the ongoing count;
- **Avg:** shows the average of values given to the In slot in the ongoing count
- **In:** the input value.

## Action

The MinMaxAverage component has the following action:

- Reset: sets all slots back to 0.

### 2.8.21 Negative

The Negative component simply converts any numeric input to a negative output value.

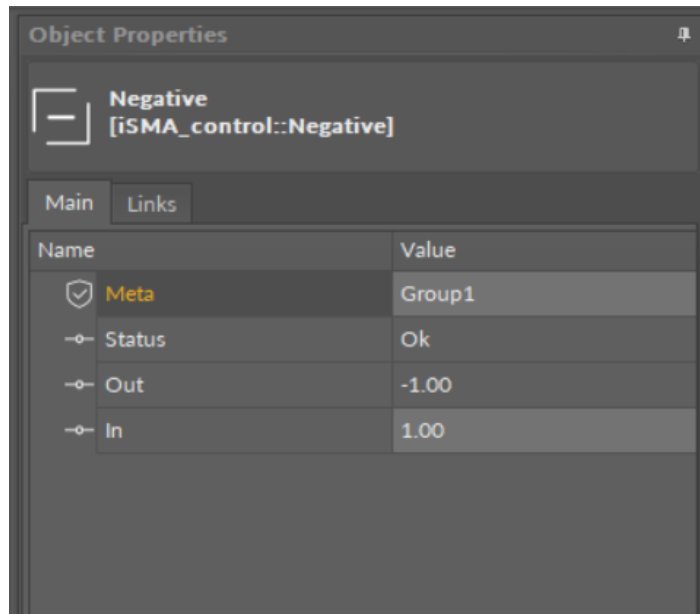


Figure 69. Negative component

## Slots

The Negative component has the following slots:

- **Status:** shows the component's status;
- **Out:** the negative value of the In slot value;
- **In:** the input value.

### 2.8.22 Reset

The Reset component performs a linear reset on the In value.

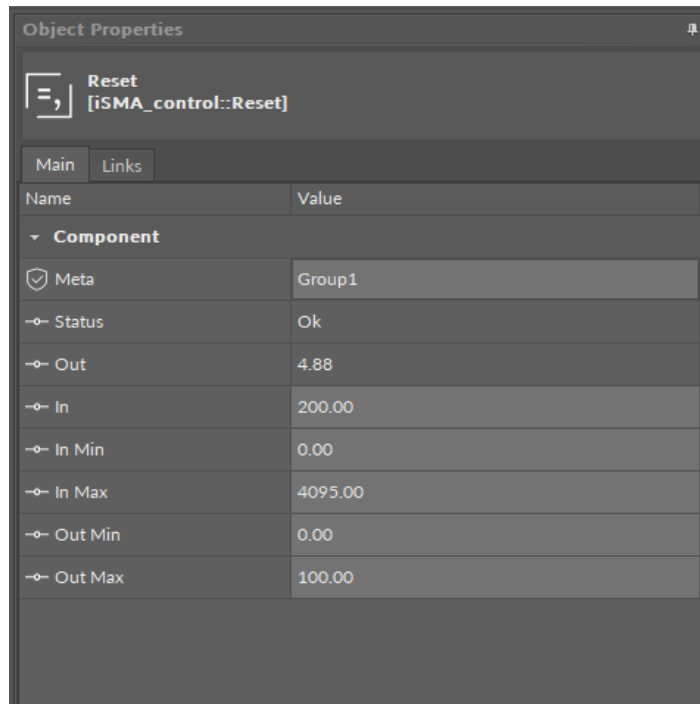


Figure 70. Reset component

## Slots

The Reset component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value;
- **In:** the input value;
- **In Min:** the input low limit, must be less than the input high limit;
- **In Max:** the input high limit, must be greater than the input low limit;
- **Out Min:** the output low limit, may (or may not) be greater than the output high limit;
- **Out Max:** the output high limit, may (or may not) be greater than the output low limit.

The Reset operation is defined by the In Min/Max and Out Min/Max slots. For example, the Reset component is used to establish a hot water control setpoint, based on the outside air temperature at the In slot. If the outside air temperature is 0°F, the hot water setpoint is 200°F. If the outside air temperature is 75°F, the hot water setpoint is 100°F. The Reset component is configured as follows:

In Min (input low limit) = 0.0

In Max (input high limit) = 75.0

Out Min (output low limit) = 200.0

Out Max (output high limit) = 100.0

Whenever the In value is beyond the input limits, the output is limited by the corresponding output limit (in this case, 200 at 0°F or below, 100 at 75°F, or above). If the input is at an intermediate value, the output scales linearly. For example, if the outside air temperature is at 38.2°F, the Reset output is 149.1°F.

## 2.8.23 Round

The Round component returns the nearest integer, rounding away from zero in halfway cases.

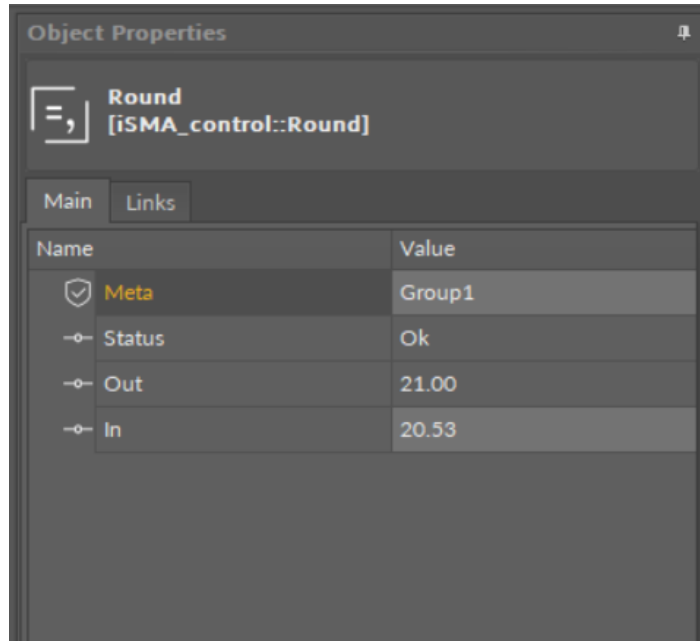


Figure 71. Round component

## Slots

The Round component has the following slots:

- **Status:** shows the component's status;
- **Out:** the rounded integer value of the In slot value;
- **In:** the input value.

## 2.8.24 Sine

The Sine component returns a sine value of the In slot (performs the operation  $out := \sin(InA)$ ).



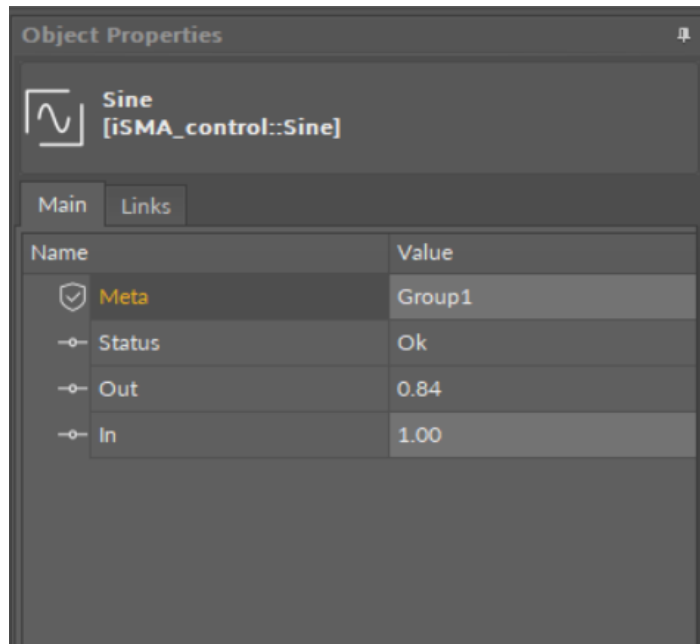


Figure 72. Sine component

## Slots

The Sine component has the following slots:

- **Status:** shows the component's status;
- **Out:** the sine value of the In slot;
- **In:** the input value.

### 2.8.25 SquareRoot

The SquareRoot component returns a square root of the In slot value (performs the operation  $out := \sqrt{InA}$  (square root of InA)).

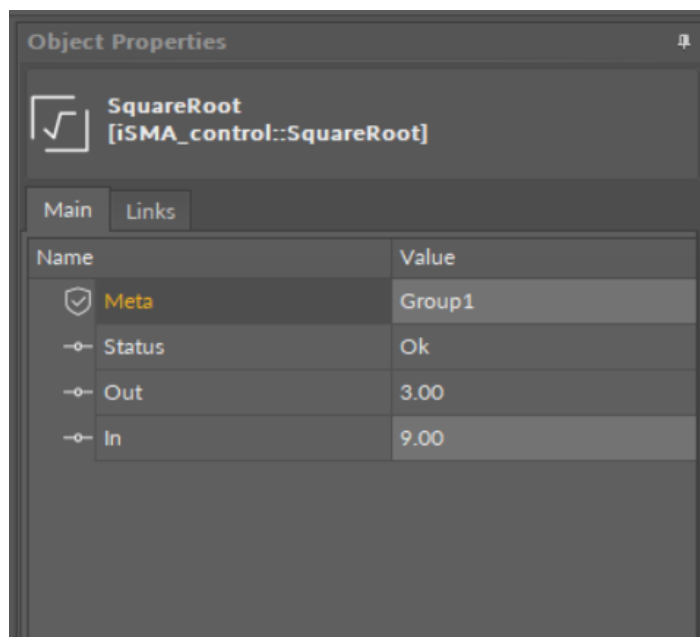


Figure 73. SquareRoot component

## Slots

The SquareRoot component has the following slots:

- **Status:** shows the component's status;
- **Out:** the square root value of the In slot;
- **In:** the input value.

### 2.8.26 Tangent

The Tangent component return a tangent value of the In slot (performs the operation  $\text{out} := \tan(\text{InA})$ ).

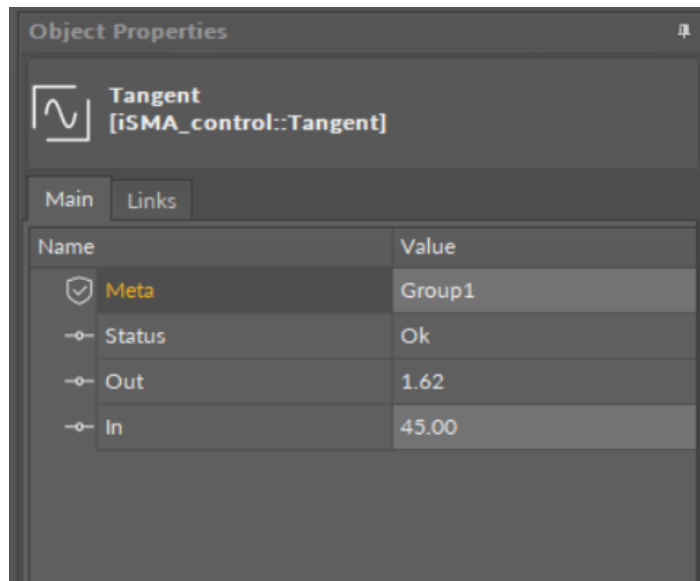


Figure 74. Tangent component

## Slots

The Tangent component has the following slots:

- **Status:** shows the component's status;
- **Out:** the tangent value of the In slot;
- **In:** the input value.

### 2.8.27 Truncate

The Truncate component performs the mathematical operation of returning the nearest integer, not greater in magnitude than the input float.

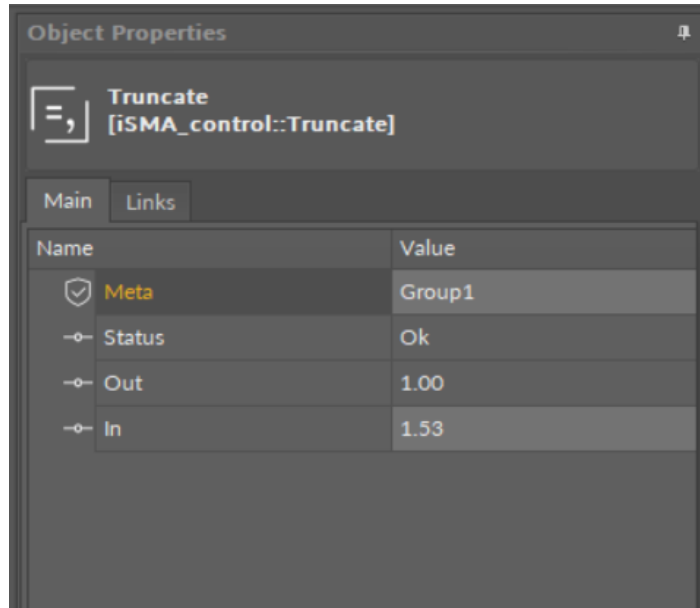


Figure 75. Truncate component

## Slots

The Truncate component has the following slots:

- **Status:** shows the component's status;
- **Out:** the truncated value of the In slot;
- **In:** the input value.

## 2.9 Switch Components

This section outlines switch components.

### 2.9.1 BooleanSwitch

The BooleanSwitch component selects between two Boolean inputs based on the Boolean In Switch slot.

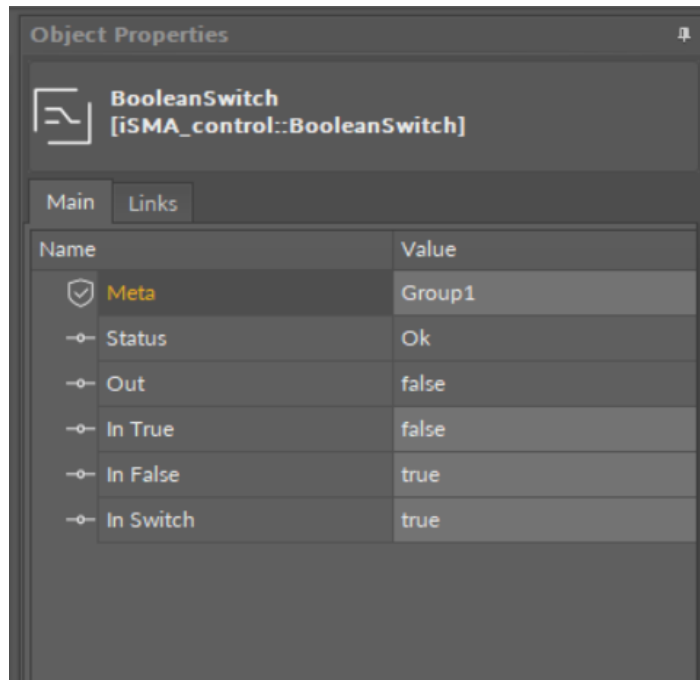


Figure 76. BooleanSwitch component

## Slots

The BooleanSwitch component has the following slots:

- **Status:** shows the component' status;
- **Out:** the value passed either from the In True or In False slot, indicated in the In Switch slot;
- **In True:** the value transferred to the Out slot if the In Switch slot is set to true;
- **In False:** the value transferred to the Out slot if the In Switch slot is set to false;
- **In Switch:** the value indicating, which slot transfers value to the output: true for the In True value, false for the In False value.

### 2.9.2 IntegerSwitch

The IntegerSwitch component selects between two integer inputs based on the Boolean In Switch slot.

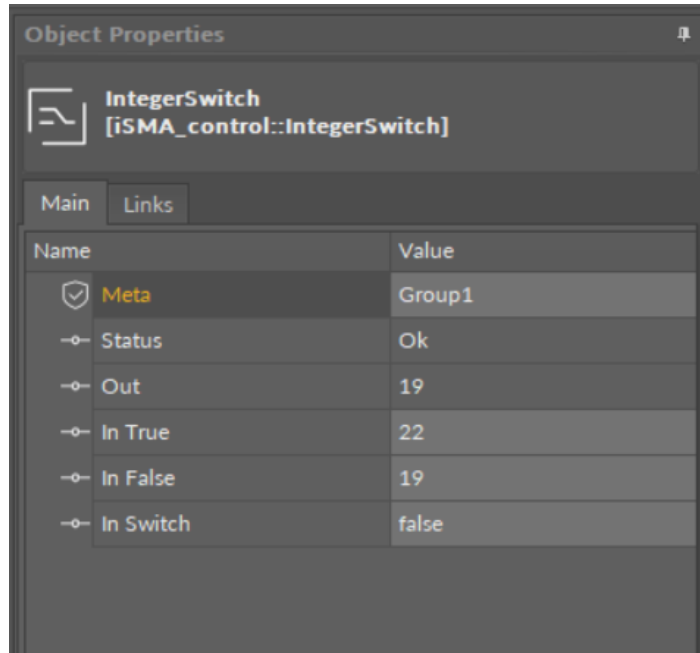


Figure 77. IntegerSwitch component

## Slots

The IntegerSwitch component has the following slots:

- **Status:** shows the component' status;
- **Out:** the value passed either from the In True or In False slot, indicated in the In Switch slot;
- **In True:** the value transferred to the Out slot if the In Switch slot is set to true;
- **In False:** the value transferred to the Out slot if the In Switch slot is set to false;
- **In Switch:** the value indicating, which slot transfers value to the output: true for the In True value, false for the In False value.

### 2.9.3 NumericSwitch

The NumericSwitch component selects between two numeric inputs based on the Boolean In Switch slot.

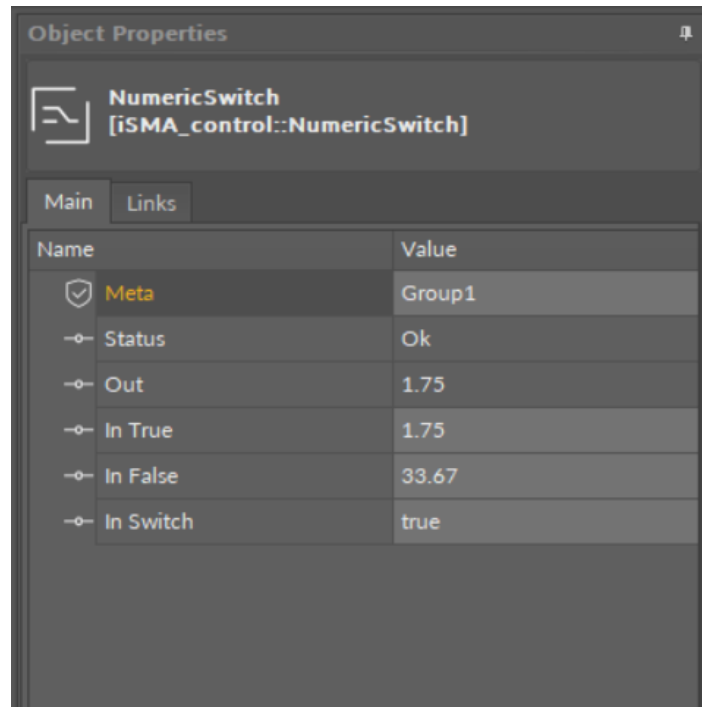


Figure 78. NumericSwitch component

## Slots

The NumericSwitch component has the following slots:

- **Status:** shows the component' status;
- **Out:** the value passed either from the In True or In False slot, indicated in the In Switch slot;
- **In True:** the value transferred to the Out slot if the In Switch slot is set to true;
- **In False:** the value transferred to the Out slot if the In Switch slot is set to false;
- **In Switch:** the value indicating, which slot transfers value to the output: true for the In True value, false for the In False value.

## 2.10 Timer Components

This section outlines timer components.

### 2.10.1 BooleanDelay

The BooleanDelay component provides the way to delay the change of the Boolean Out property value by configuring the associated Delay property. Delay properties are provided for on (true) and off (false) statuses and are labelled On Delay and Off Delay, respectively. The delay applies to any transition (status change from on to off or off to on) at the component's Boolean input. Both delay times are configurable in terms of hours, minutes, and seconds.

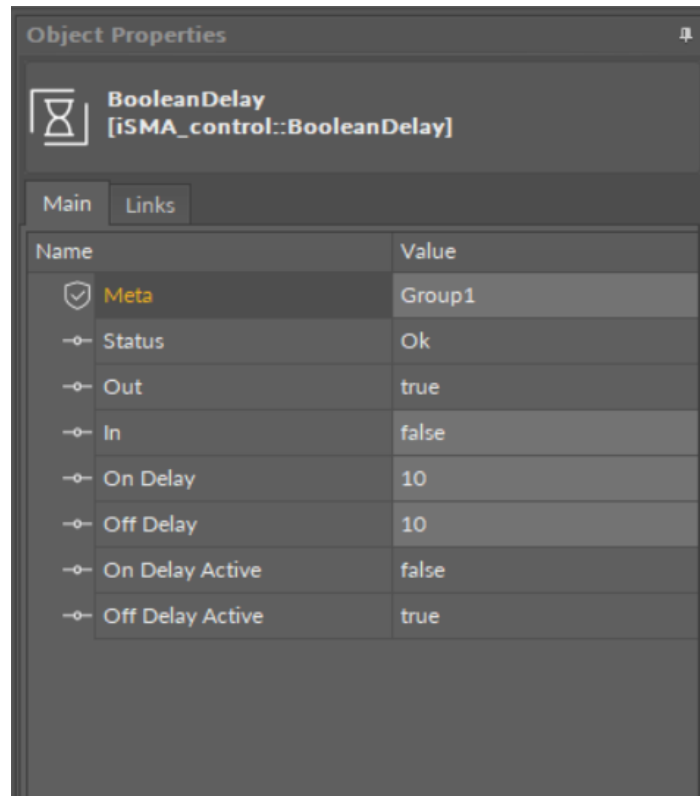


Figure 79. BooleanDelay component

## Slots

The BooleanDelay component has the following slots:

- **In:** typically, this property is set by linking a Boolean out value into it. The default state can be manually configured to a true or false value so that when no other value is linked into this property, the default value is used. This property value is passed to the Out (after any On Delay or Off Delay) whenever there is a change in this property;
- **On Delay:** allows to set the amount of time (in hours, minutes, and seconds) that will expire before sending a true (On) value to the Out property. Time begins to expire at the moment that a change in the In property occurs (a transition from false or null to true). If the On Delay value is 0, the change of the state proceeds without delay;
- **Off Delay:** allows to set the amount of time (in hours, minutes, and seconds) that will expire before sending a false (Off) value to the Out property. The time begins at the moment that a change in the In property occurs (a transition from true to false or false to true). If the Off Delay value is 0, the change of the state proceeds without delay;
- **On Delay Active:** shows whether or not the On Delay time is actively counting down to expiration. This (normally false) value changes to true anytime that a transition from false to true occurs at the In property and stays at true until any Off Delay time is expired. If the On Delay value is set to 0, then this value does not change to true;
- **Off Delay Active:** shows whether or not the Off Delay time is actively counting down to expiration. This (normally false) value changes to true anytime that a transition from true to false occurs at the In property and stays at true until any Off Delay time is expired. If the On Delay value is set to 0, then this value does not change to true;
- **Out:** the property has true and false options available. These values are set at the end of any On Delay or Off Delay to reflect the In property value.

## 2.10.2 OneShot

The OneShot component provides a single, temporary, Boolean output for a specified duration (as set in the Time property). The OneShot action occurs with a false-to-true value transition at the In property or with an invoked Fire action. If either of these conditions occurs, the Out property value is set to true, and the Out Not property value is set to false for a time that is equal to the value of the Time property. If the time expires, these values revert to the previous (default) values.

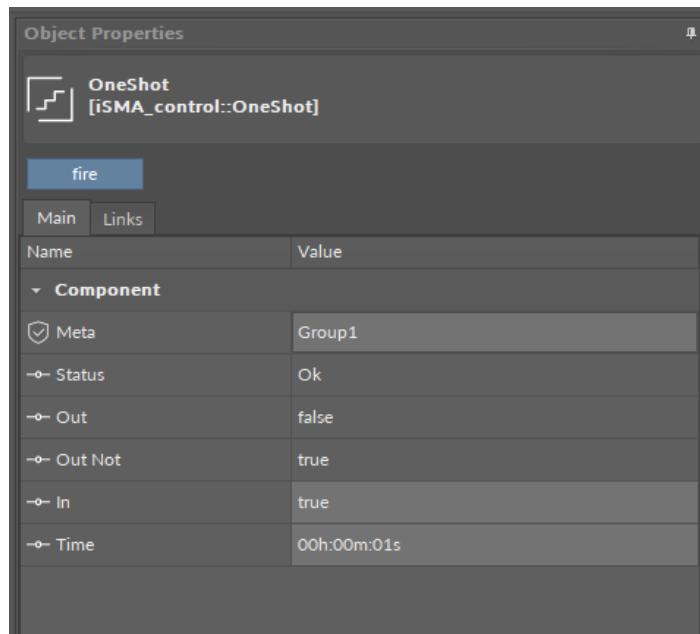


Figure 80. OneShot component

## Slots

The OneShot component has the following slots:

- **Status:** shows the component's status;
- **Out:** displays the current value that changes from false to true at the In property value. After the OneShot is triggered and the Time value period expires, this value returns to the default (false) value;
- **Out Not:** the property has true or false options available. The Out value changes from false to true at the In property value or a Fire action. After the OneShot is triggered and the Time value period expires, this value returns to the default (true) value.
- **In:** typically, this property is set by linking a Boolean Out value into it. The default state can be manually configured to a Boolean value so that, when no other value is linked into this property, the default value is used. This property value is passed to the component's Out property for the amount of time set in the Time property;
- **Time:** determines how long the Out and Out Not properties hold their one-shot values.

## 2.10.3 NumericDelay

The NumericDelay component provides a way to delay the change of a numeric Out property value by configuring an associated Delay property. The delay applies to any change at the component's numeric input. The delay time is configurable in terms of hours, minutes, and seconds.



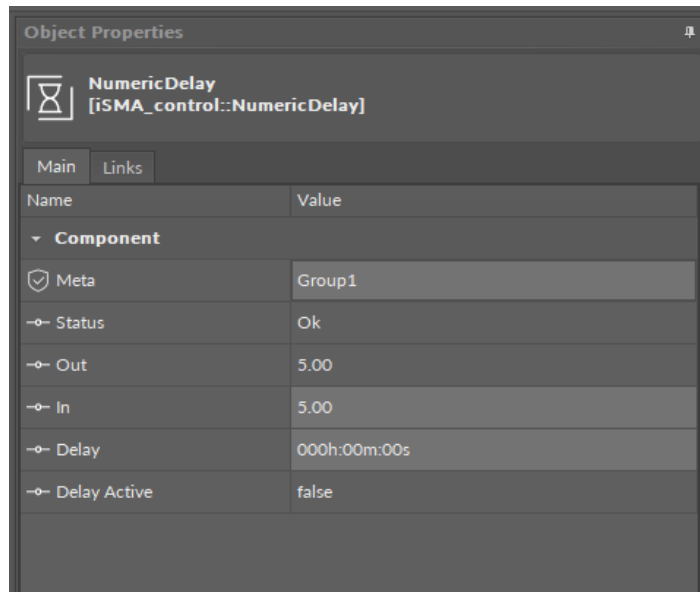


Figure 81. NumericDelay component

## Slots

The NumericDelay component has the following slots:

- **Status:** shows the component's status;
- **Out:** a numeric output. These values are set at the end of any delay to reflect the In property value;
- **In:** typically, this property is set by linking a numeric out value into it. The default state can be manually configured to a true or false value so that, when no other value is linked into this property, the default value is used. This property value is passed to the Out (after Delay) whenever there is a change in this property;
- **Delay:** allows to set the amount of time (in hours, minutes, and seconds) that will expire before sending the In value to the Out property. The time begins to expire at the moment that a change in the In property occurs;
- **Delay Active:** shows whether or not the Delay time is actively counting down to expiration. This (normally false) value changes to true anytime that a change in the In property occurs and stays at true until any Delay time is expired. If the Delay value is set to 0, then this value does not change to true.

### 2.10.4 Timer

The Timer component outputs a pulse for the configured amount of time; the In slot is used to fire the timer:

- if low, the Out slot is forced to false,
- if high, the Out slot equals 1 until the timer reaches the Time value in seconds.

Alternatively, the pulse can be fired from the Start Timer action if the In slot is not linked.

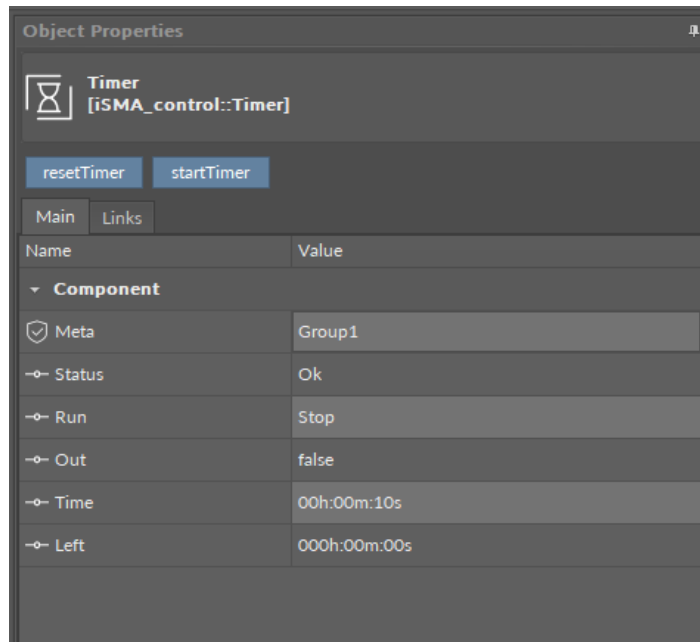


Figure 82. Timer component

## Slots

The Timer component has the following slots:

- **Status:** shows the component's status;
- **Run:** used to fire the timer on transition from false to true;
- **Out:** the timed pulse output;
- **Time:** sets the desired duration of the output pulse;
- **Left:** shows the remaining time before the output transition from true to false.

## 2.11 Utility Components

This section outlines utility components available in the iSMA-B-AAC20 Control kit.

### 2.11.1 NumericBitXor

The NumericBitXor component performs a logical XOR on the bit equivalent of the numeric In value against the bit equivalent of its numeric Mask slot value. It may be useful in cases where the Boolean information is mapped into integer values.

For example, some manufacturers multiplex binary data into a single numerical point by converting the bits from hexadecimal to decimal format. To obtain the status of the individual binary data, the number must be converted back from decimal to hex format. Each digit of the hex number represents a particular binary parameters state (0 = false, 1 = true). The NumericBitXor component converts a numeric input to a hex value and compares it against the Mask value. Each digit is analysed using exclusive OR (XOR) logic, setting the corresponding digit value to either 1 or 0.

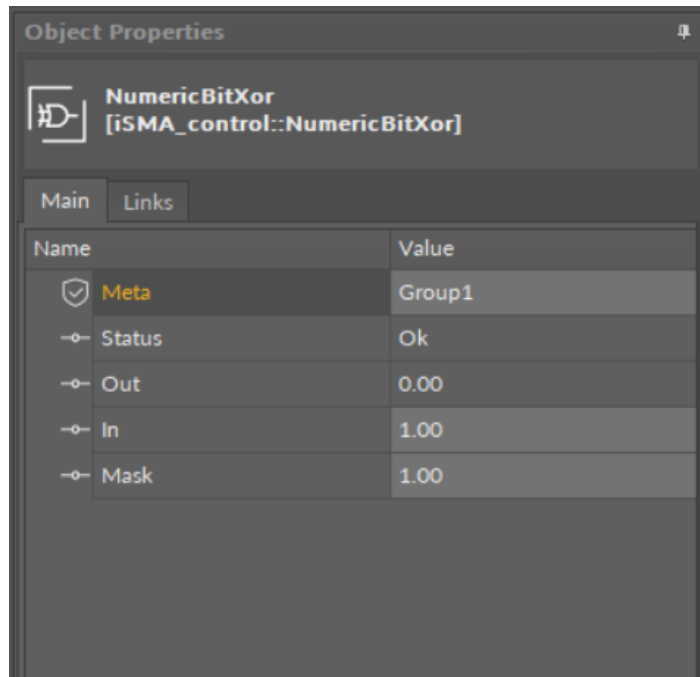


Figure 83. NumericBitXor component

## Slots

The NumericBitXor component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the outcome of the XOR operation;
- **In:** the numeric input value;
- **Mask:** the numeric input value.

### 2.11.2 Counter

The Counter component counts the Boolean inactive to active transitions. It supports counting up, counting down, presetting, and clearing.

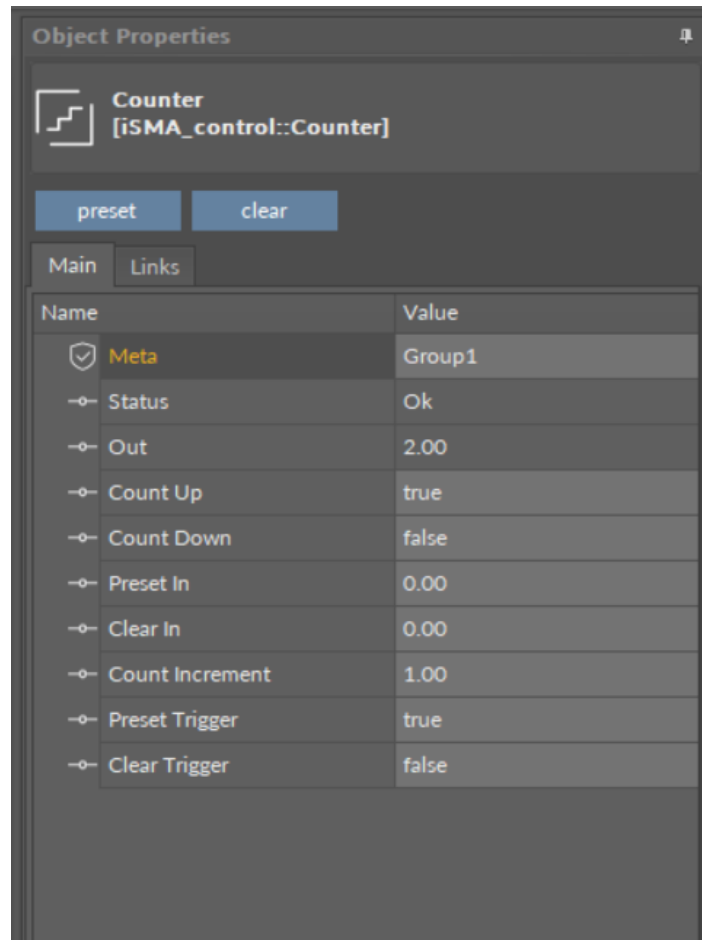


Figure 84. Counter component

## Slots

The Counter component has the following slots:

- **Count Up:** the Boolean input. With the inactive to active transition the value of the Out property increments by the Count Increment value;
- **Count Down:** the Boolean input. With the inactive to active transition the value of the Out property decrements by the Count Increment value;
- **Preset In:** the numeric input, which is set in the Out property if the Preset action is invoked;
- **Clear In:** the numeric input, which is set in the Out property if the Clear action is invoked;
- **Count Increment:** the value that the Out property changes for a single count up or count down in active transition;
- **Preset Trigger:** the Boolean input. If this input changes from inactive to active, it invokes the Preset action;
- **Clear Trigger:** the Boolean input. If this input changes from inactive to active, it clears the Preset value.

## Actions

The Counter component includes the following actions:

- **Preset:** sets the value of the Out property to the Preset In value;
- **Clear:** sets the value of the Out property to the Clear In value.

### 2.11.3 Multivibrator

The Multivibrator component provides an oscillating binary pulse output (Boolean) with a period configurable from 1 s and a duty cycle configurable from 0 to 100.

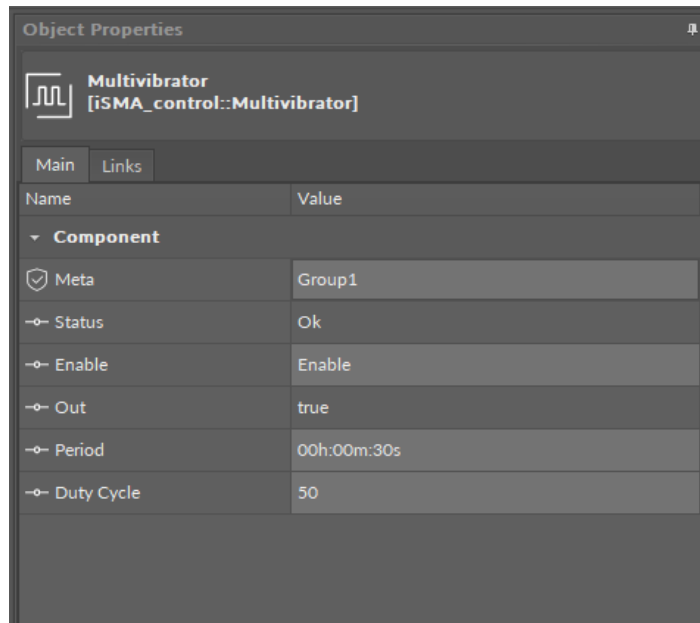


Figure 85. Multivibrator component

### Slots

The Multivibrator component has the following slots:

- **Status:** shows the component's status;
- **Enable:** enables or disables the component;
- **Out:** the binary pulse output;
- **Period:** sets the period time in seconds;
- **Duty Cycle:** sets the duty cycle of the component.

### 2.11.4 NumericBitAnd

The NumericBitAnd component performs a logical AND on the bit equivalent of the numeric In value against the bit equivalent of its numeric Mask slot value. It may be useful in cases where the Boolean information is mapped into integer values.

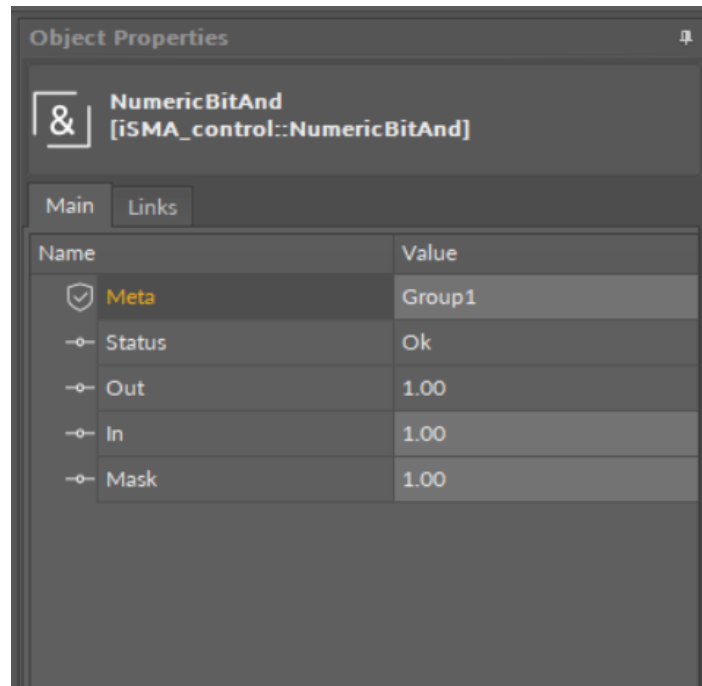


Figure 86. NumericBitAnd component

## Slots

The NumericBitAnd component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the outcome of the AND operation;
- **In:** the numeric input value;
- **Mask:** the numeric input value.

### 2.11.5 NumericBitOr

The NumericBitOr component performs a logical OR on the bit equivalent of the numeric In value against the bit equivalent of its numeric Mask slot value. It may be useful in cases where the Boolean information is mapped into integer values.

For example, some manufacturers multiplex binary data into a single numerical point by converting the bits from hexadecimal to decimal format. To obtain the status of the individual binary data, the number must be converted back from decimal to hex format. Each digit of the hex number represents a particular binary parameters state (0 = false, 1 = true). The NumericBitOr component converts a numeric input to a hex value and compares it against the Mask value. Any digits with a value of 1 in the Mask or the input will result in a corresponding value of 1 in the same digit of the output. Any value on the output slot greater than 1 indicates that at least one of the binary parameters is true.

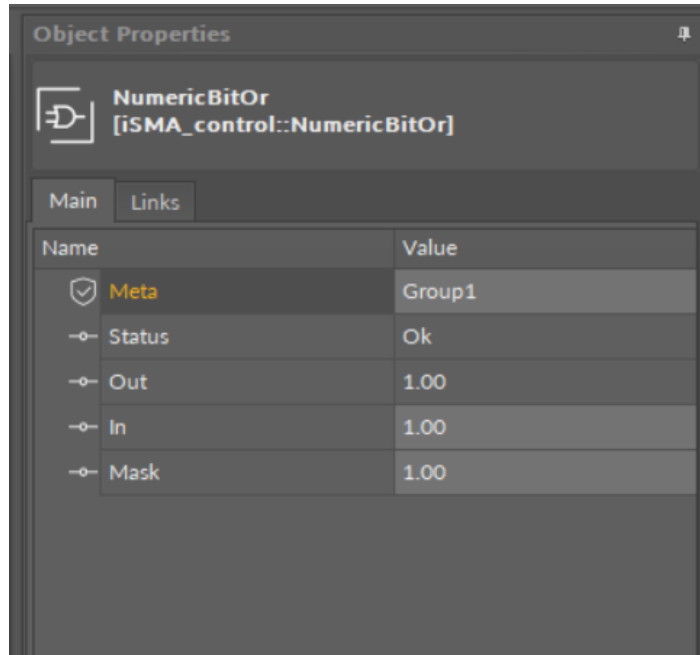


Figure 87. NumericBitOr component

## Slots

The NumericBitOr component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output with the outcome of the OR operation;
- **In:** the numeric input value;
- **Mask:** the numeric input value.

### 2.11.6 Ramp

The Ramp component provides a numeric Out with a linear ramping output. Slots define the Period, Amplitude, and Offset.

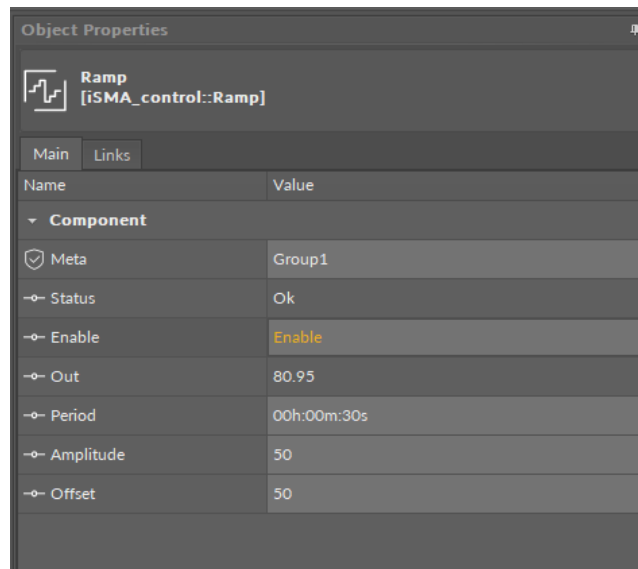


Figure 88. Ramp component

## Slots

The Ramp component has the following slots:

- **Status:** shows the component's status;
- **Enable:** enables or disables the component;
- **Out:** the output value calculated based on the set parameters;
- **Period:** allows to set the length of one cycle of the component's function;
- **Amplitude:** allows to set an amplitude for the output values function;
- **Offset:** allows to set an offset for the output values function.

### 2.11.7 Random

The Random component generates random numbers. The output is derived by multiplying a random number (that is greater than 0 but less than 1) times a variable Multiplier plus Offset value.

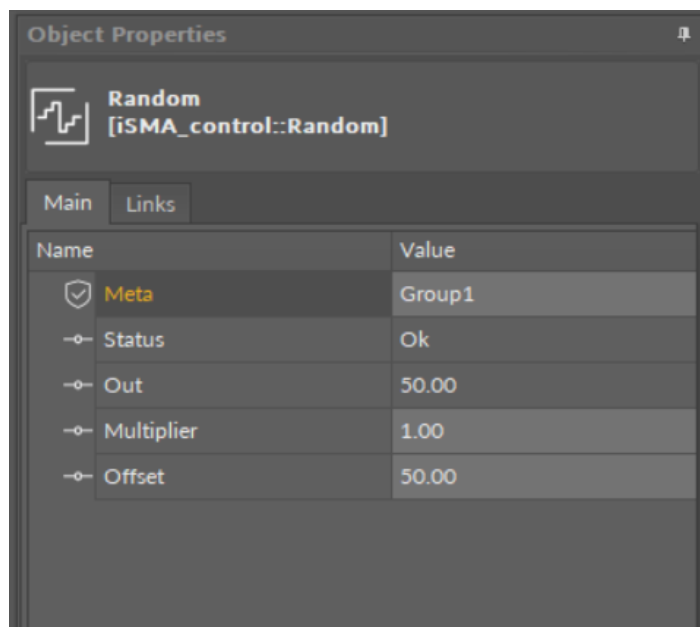


Figure 89. Random component

## Slots

The Random component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value calculated based on the Multiplier and Offset values;
- **Multiplier:** multiplies the random number (the random number is  $\geq 0.0$  but  $< 1.0$ ). The multiplier is set to 1.0 by default;
- **Offset:** the positive or negative distance from zero that the wave's amplitude is centered on. The default offset value is 50.

### 2.11.8 RateOfChange

The RateOfChange component limits the change of the output slot based on the derivative of the input. The output will change to the value of the input if the input is changed by more than the value of deviation. Smaller changes in the input will leave the



output unchanged. Range of values of component can be restricted by setting values in the Off Normal slots.

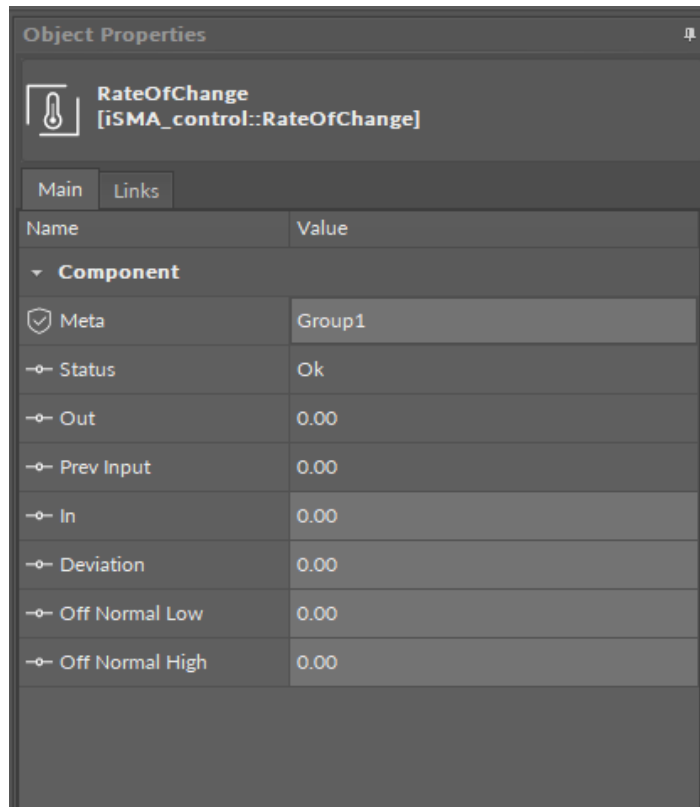


Figure 90. RateOfChange component

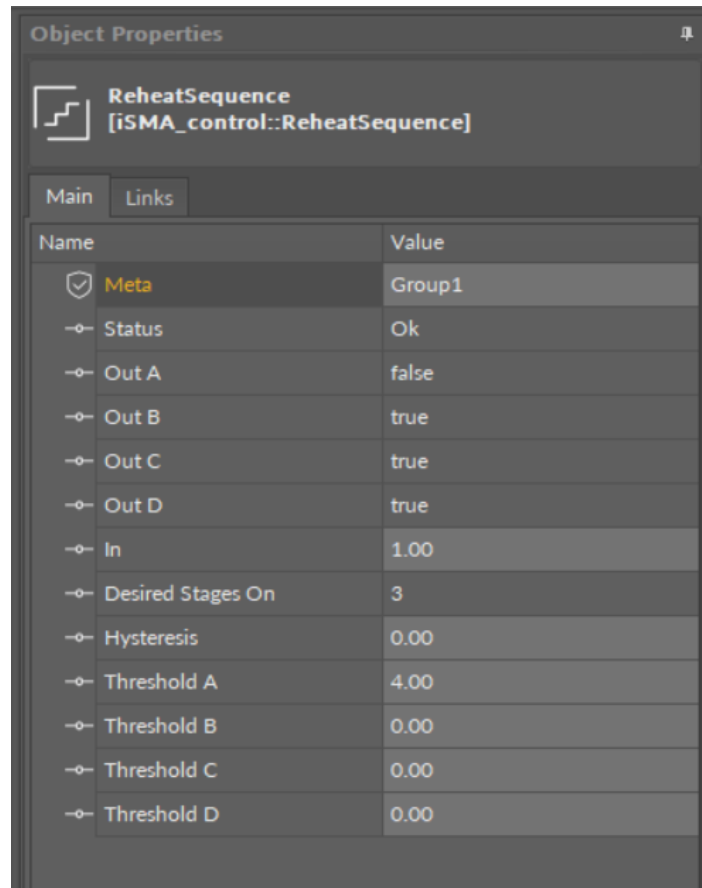
## Slots

The RateOfChange component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value counted according to set parameters;
- **Prev Input:** (read-only) output slot presenting the previous state of the In slot;
- **In:** the input value;
- **Deviation:** defines a minimal value that the input needs to change by to force a change in the output;
- **Off Normal Low:** low limit restriction value for the range of values for which the component works; to disable, set 0;
- **Off Normal High:** high limit restriction value for the range of values for which the component works; to disable, set 0.

### 2.11.9 ReheatSequence

The ReheatSequence component provides a linear sequence of up to 4 loads based on configurable thresholds. The component's algorithm sets an output to true if the In value is greater than a corresponding threshold, and returns the output to false if the In value is less than the threshold minus the hysteresis value.
















Name	Value
 <b>Meta</b>	Group1
 Status	Ok
 Out A	false
 Out B	true
 Out C	true
 Out D	true
 In	1.00
 Desired Stages On	3
 Hysteresis	0.00
 Threshold A	4.00
 Threshold B	0.00
 Threshold C	0.00
 Threshold D	0.00

Figure 91. ReheatSequence component

## Slots

The ReheatSequence component has the following slots:

- **OutA-OutD**: four output slots;
- **In**: controlled variable;
- **Desired Stages On**: number of currently turned on outputs;
- **Hysteresis**: s hysteresis value, which—if exceeded—allows to turn an output off;
- **ThresholdA-ThresholdD**: four threshold values enabling to turn a respective output on.

### 2.11.10 SineWave

The SineWave component generates a sine wave as a numeric out.

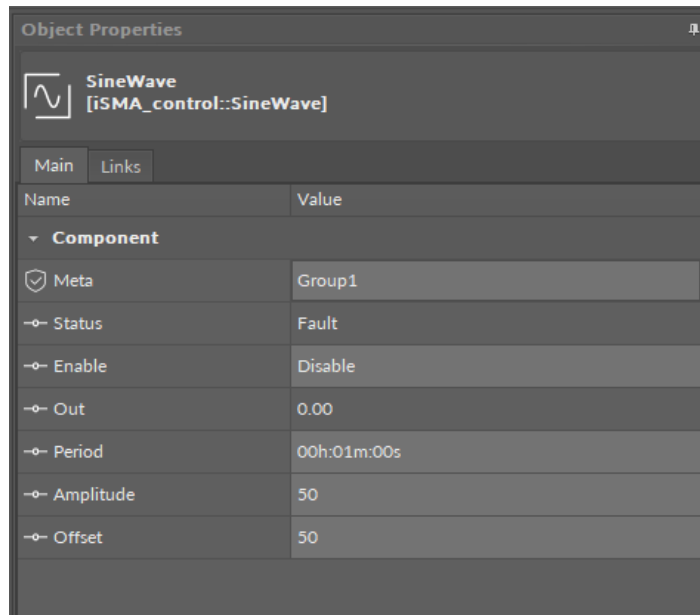


Figure 92. SineWave component

## Slots

The SineWave component has the following slots:

- **Status:** shows the component's status;
- **Enable:** enables or disables the component;
- **Out:** the sine wave value calculated based on set parameters;
- **Period:** allows to set the length of one cycle of the component's function;
- **Amplitude:** allows to set an amplitude for the output values function;
- **Offset:** allows to set an offset for the output values function.

### 2.11.11 Frequency

The Frequency component calculates a pulse input frequency.

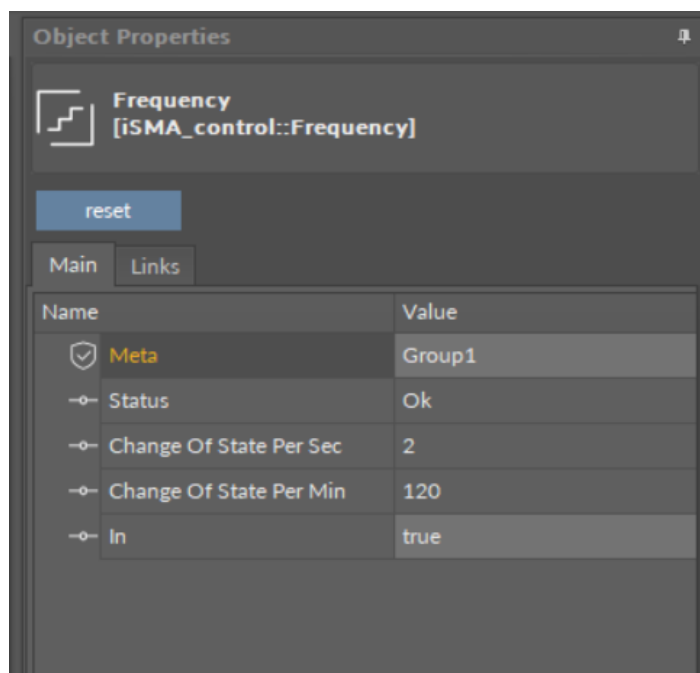


Figure 93. Frequency component

## Slots

The Frequency component has the following slots:

- **Status:** shows the component's status;
- **Change Of State Per Sec:** indicates the frequency of changes on the In slot per second;
- **Change Of State Per Min:** indicates the frequency of changes on the In slot per minute;
- **In:** the input slot.

## Action

The Frequency component has the following action:

- **Reset:** resets the slots values to 0.

### 2.11.12 Hysteresis

The Hysteresis component sets on/off trip points to an input variable.

There are two internal floats called Rising Edge and Falling Edge, which are configurable:

- If the Rising Edge > Falling Edge, the Out behaves normally, i.e.

Out = true if the In rises above the Rising Edge value,

Out = false if the In falls below the Falling Edge value.

- If the Rising Edge < Falling Edge, the Out behaves inverted, i.e.

Out = false if the In rises above the Falling Edge value,

Out = true if the In falls below the Rising Edge value,

- If the Rising Edge = Falling Edge, the object behaves as a simple comparator,

Out = true if the In is greater than the Rising Edge.

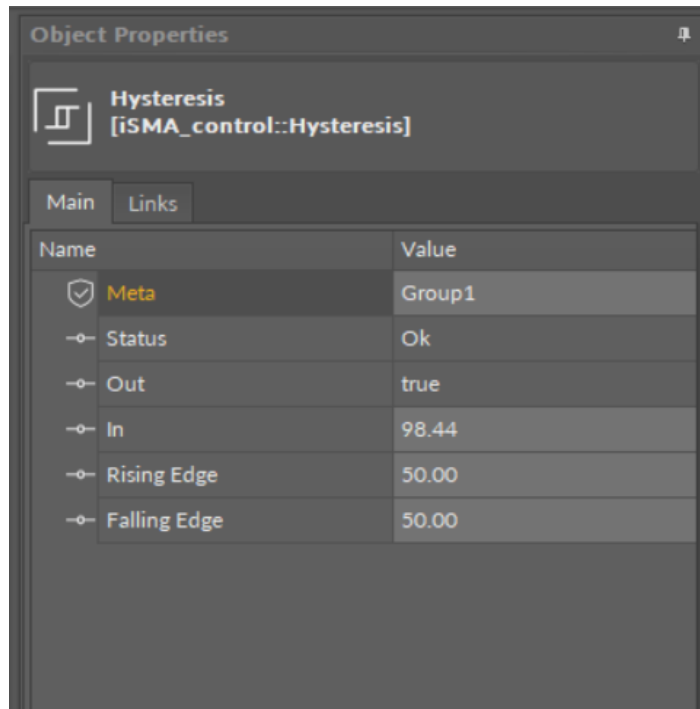


Figure 94. Hysteresis component

## Slots

The Hysteresis component has the following slots:

- **Status:** shows the component's status;
- **Out:** the true or false value changing depending on the In slot value in relation to the Rising/Falling Edge slots;
- **In:** the input value;
- **Rising Edge:** a trip point for the Out value if the In slot value is rising;
- **Falling Edge:** a trip point for the Out value if the In slot value is falling.

### 2.11.13 Limiter

The Limiter component allows to set the limits for the Out slot values. The In slot value is transferred directly to the Out slot only if it is within limit values. In case the In slot value exceeds limit values, so it is either lower than the Low Limit value, or higher than the High Limit value, the respective limit value is transferred to the Out slot.

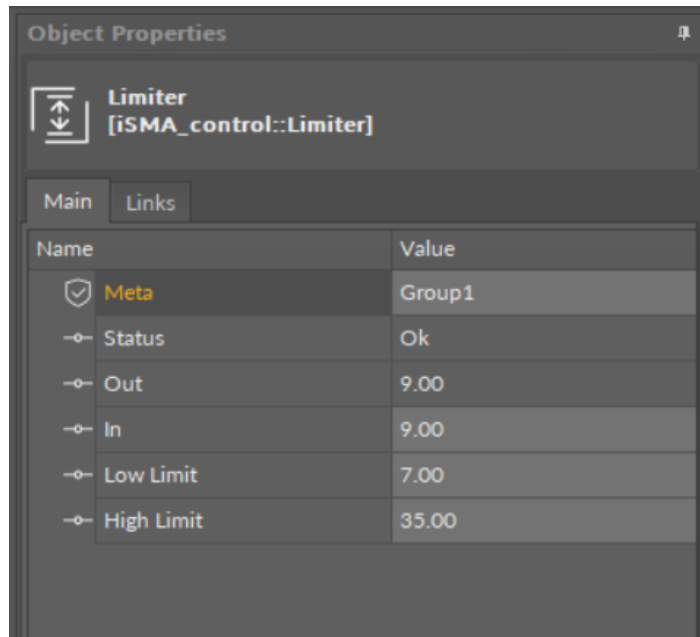


Figure 95. Limiter component

## Slots

The Limiter component has the following slots:

- **Status:** shows the component's status
- **Out:** the value transferred from the In slot if it is within limits, if otherwise—the Low Limit value is transferred in case the In slot value is lower than the Low Limit value, and the High Limit value is transferred in case the In slot value is higher than the High Limit value;
- **In:** the input value;
- **Low Limit:** the low limit value;
- **High Limit:** the high limit value.

### 2.11.14 Linearize

The Linearize component performs a piecewise linearization of a float and allows to scale the Out slot value depending on the set input values interval. The In slot value is compared with  $x$  intervals, and the component estimates the Out slot value based upon the linear function.

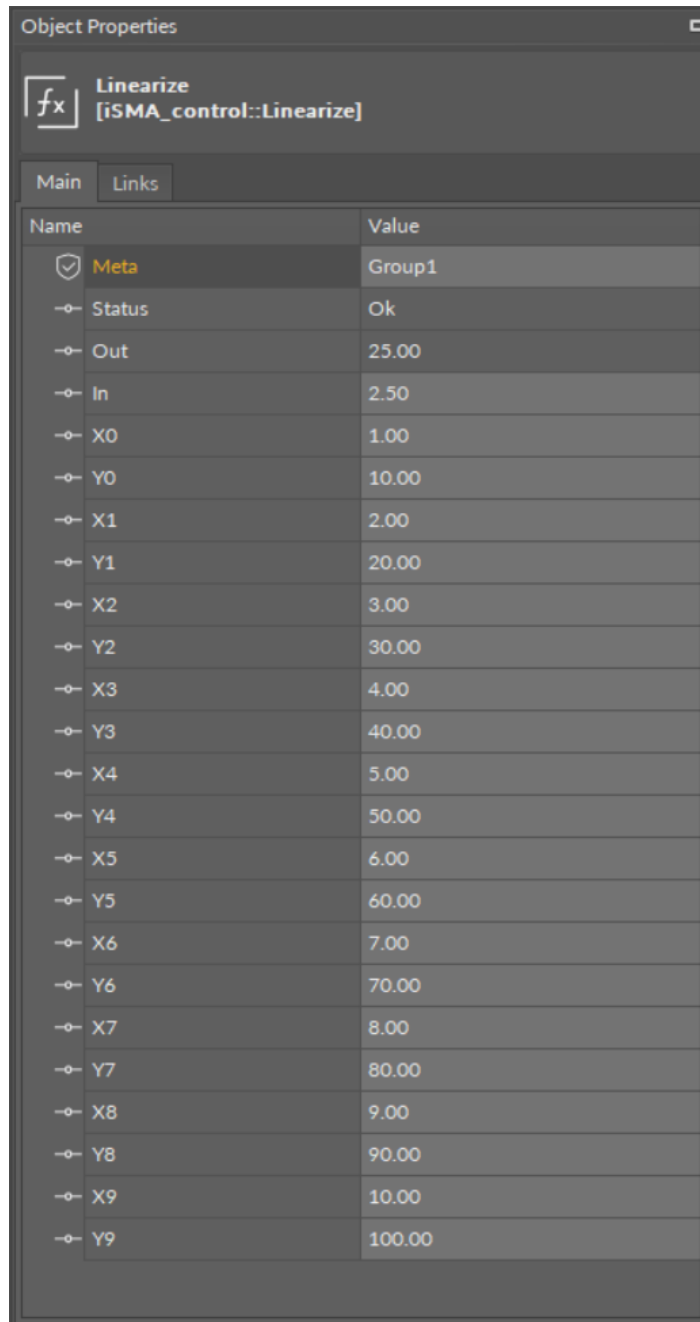


Figure 96. Linearize component

The X, Y pairs indicate points along the input curve. For the X value of the input, there is a corresponding Y value of the output. For input values between these points, the component estimates the output based upon the linear equation, therefore, it converts a table of values into a curve using linear interpolation between the values. Individual slope/intercept constants are computed between the X's and Y's using the formula  $y = mx + b$ , where  $m = \frac{y_m - y_n}{x_m - x_n}$ .

If the In value is not in the range of X0 to X9, then output is set to null.

**Note:** The slope may be positive or negative, and it is indicated by comparing X1 and X0:

- positive if  $x_1 > x_0$ ;
- negative if  $x_1 < x_0$ .

$Out := (m * in) + b$ , where m is the slope between the adjacent points and b is the Y intercept.

## Slots

The Linearize component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value estimated based on the linear function of the input value;
- **In:** the input value;
- **X0-X9:** values representing the intervals for the input value;
- **Y0-Y9:** values representing the intervals for the output value.

### 2.11.15 TempConversion

The TempConversion component converts the temperature from one unit to another (available units: Celsius, Fahrenheit, kelvin).

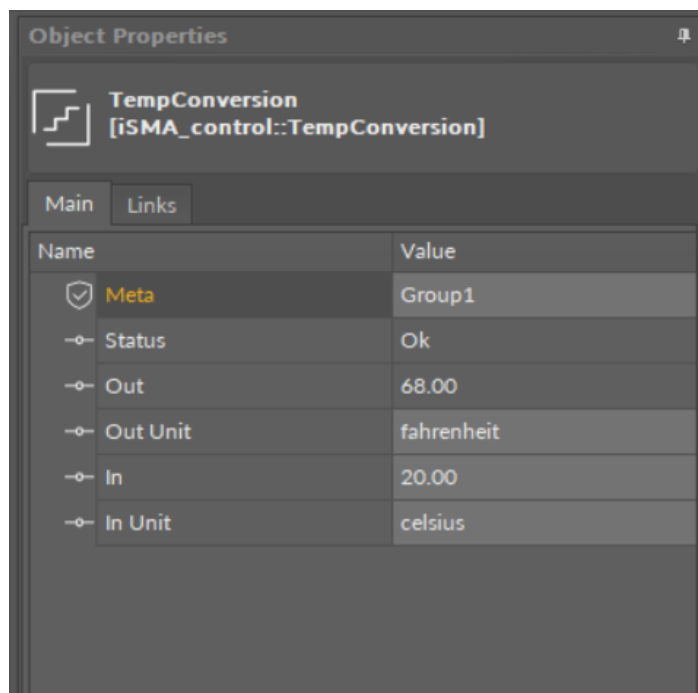


Figure 97. TempConversion component

## Slots

The TempConversion component has the following slots:

- **Status:** shows the component's status;
- **Out:** the temperature expressed in units defined in the Out Unit slot;
- **Out Unit:** defines the units that the input temperature is converted to;
- **InA-InB:** the input temperature;
- **In Unit:** defines the units of the input temperature.

Input Unit	Output Unit	Output Value
Celsius	Celsius	= In
Celsius	Fahrenheit	= (In - 32.0) * (5.0/9.0)



Input Unit	Output Unit	Output Value
Celsius	kelvin	$= In + 273.0$
Fahrenheit	Celsius	$= (In * 1.8) + 32.0$
Fahrenheit	Fahrenheit	$= In$
Fahrenheit	kelvin	$= (In * 1.8) + 32.0 + 273.0$
kelvin	Celsius	$= In - 273.0$
kelvin	Fahrenheit	$= ((In - 273.0) - 32.0) * (5.0/9.0)$
kelvin	kelvin	$= In$

Table 12. Temperature conversion table

## 2.11.16 Toggle

The Toggle component changes its Out slot value on the rising edge of its In slot. If the Out slot value is true, it changes to false if the In slot changes its value to true (rising edge), and it is held so until the next rising edge of the In slot (changing the In slot value to false takes no effect on the Out slot). If the Out slot value is false, it changes to true if the In slot changes to true (rising edge), and it is held so until the next rising edge of the In slot (changing the In slot value to false takes no effect on the Out slot).

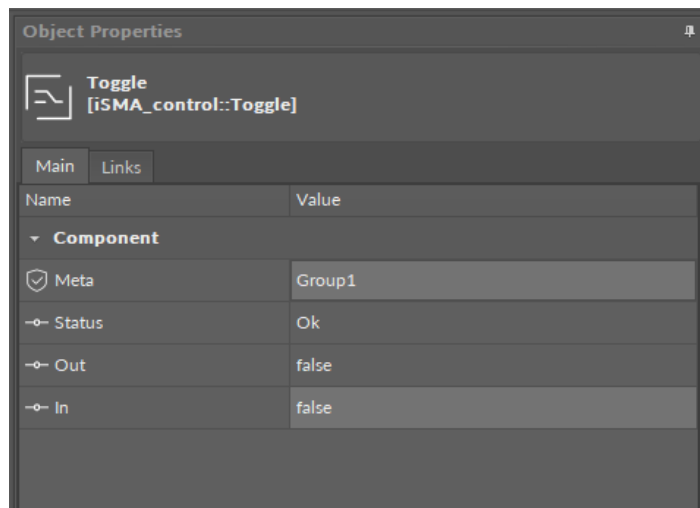


Figure 98. Toggle component

## Slots

The Toggle component has the following slots:

- **Status:** shows the component's status;
- **Out:** the Boolean value changed on the rising edge of the In slots;
- **In:** the input slot which changes the Out slot value on the rising edge—the first rising edge switches the Out slot to true, the next rising edge switches it to false, and the sequence recurs.

## 2.11.17 UpDown

The UpDown component counts basing on the Count Increment property. It supports counting up, counting down, presetting, and clearing.

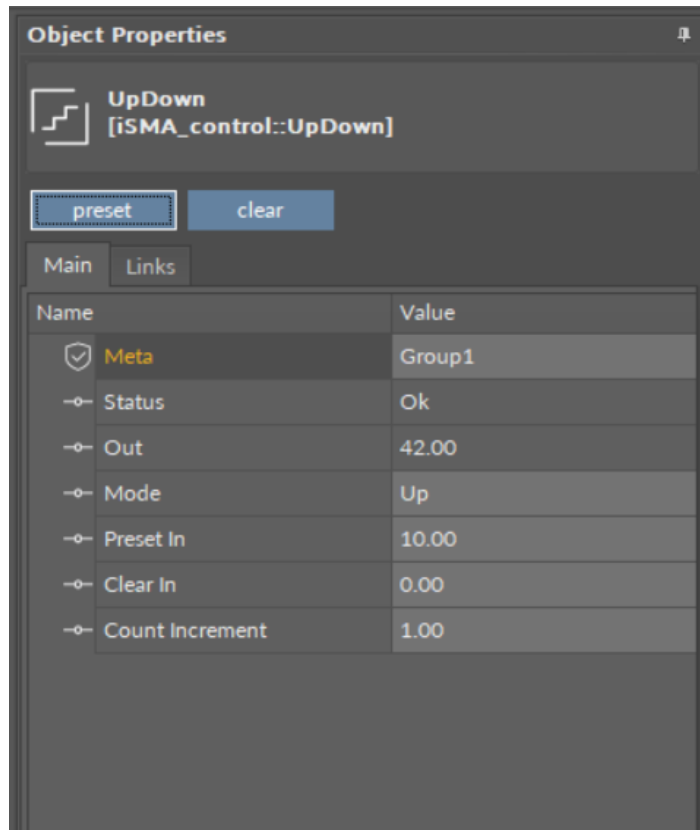


Figure 99. UpDown component

### Slots

The UpDown component has the following slots:

- **Status:** shows the component's status;
- **Out:** the output value counted according to set parameters;
- **Mode:** defines counting mode, upwards or downwards;
- **Preset In:** allows to define a preset value for the input (if the Preset action is invoked, counting starts from this value);
- **Clear In:** allows to define a value that the output value is reset to if the Clear action is invoked;
- **Count Increment:** defines the counting step.

### Actions

The UpDown component has the following actions:

- **Preset:** sets the Out value to the Preset In value;
- **Clear:** sets the Out value to the Clear In value.