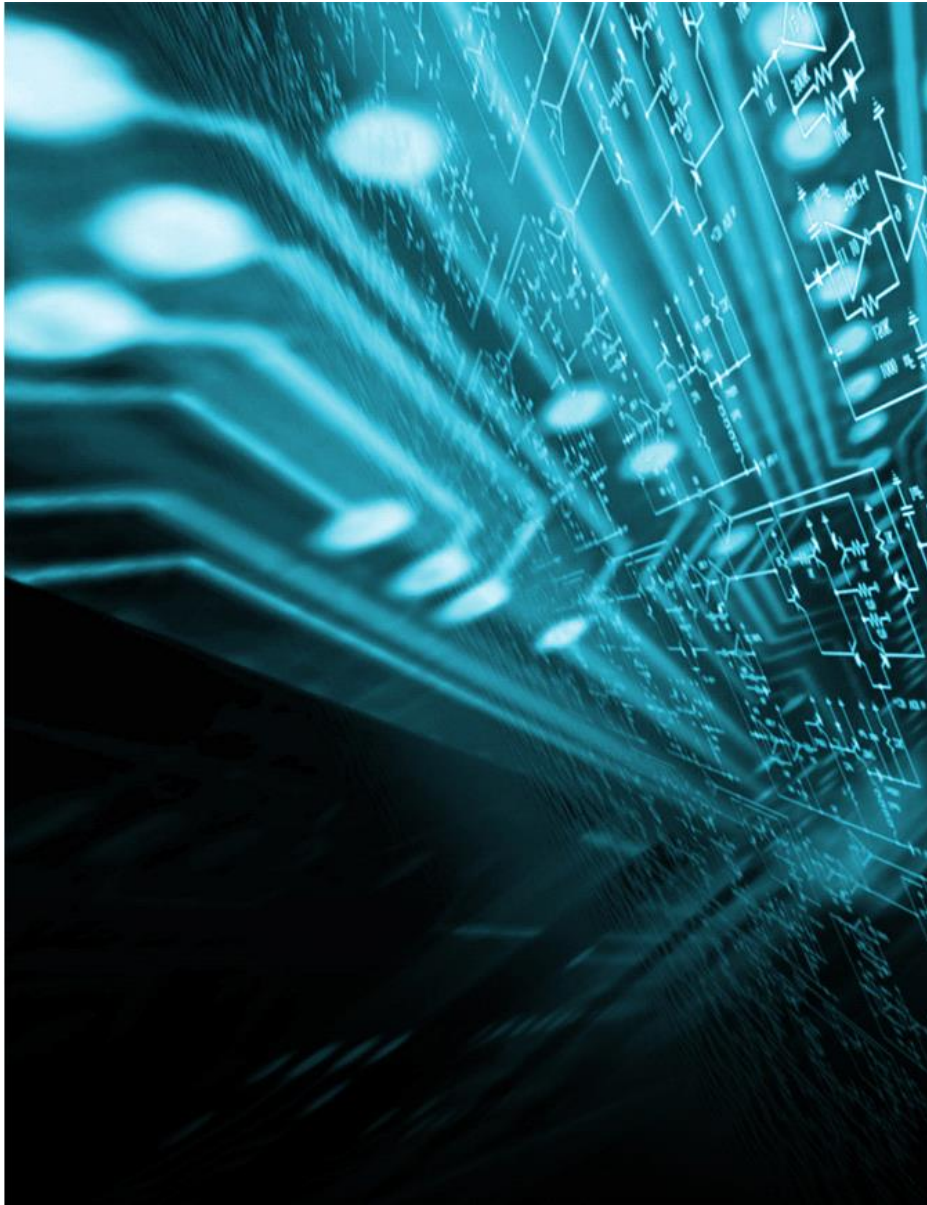


Product Manual



WBL

Logical Module

WEB-BASED

The access to the product interface does not need any installation. It is done by a web browser (IE, Chrome, Firefox, Safari). However, it does not require an internet connection.

VISUAL

Visual and intuitive, WBL makes the creation of complex automatism on your domotics system a lot easier.

KNX

Using the reading and writing functionalities on the KNX bus you can treat the data captured and easily communicate with the actuators.

GPIO (General Purpose Input/Output)

The reading and writing of the input/output ports allows interacting with the electronical circuits.

ALERTS

The triggering and sending of automatic SMS and/or Emails informs you immediately in case of urgencies.

Table of Contents

1	Installation	4
1.1	Local IP	Error! Bookmark not defined.
1.2	Activation	4
1.3	Authentication	Error! Bookmark not defined.
2	WBL Circuits	6
2.1	Ports	6
2.1.1	Standard Logical Ports.....	Error! Bookmark not defined.
2.1.2	Extended Logical Ports.....	Error! Bookmark not defined.
2.1.3	Triggers.....	Error! Bookmark not defined.
2.1.4	Actuators.....	11
2.1.5	Flow Controllers	12
2.1.6	Others Ports	15
2.2	Links	16
2.2.1	Memory Link	17
2.2.2	Passive Memory Link.....	17
2.2.3	Consumable Link	Error! Bookmark not defined.
2.2.4	Synchronization.....	17
2.3	Values.....	18
2.3.1	Numbers.....	18
2.3.2	Strings.....	Error! Bookmark not defined.
2.3.3	Tables	18
2.3.4	Inversion of the values.....	19
2.4	Addresses	19
2.4.1	KNX Group Addresses	19
2.4.2	KNX Physical Addresses.....	21
2.4.3	GPIO Addresses	21
2.4.4	Virtual Addresses	22
2.4.5	Label	Error! Bookmark not defined.
2.5	Advanced Functionalities.....	23
2.5.1	Advanced Mode	Error! Bookmark not defined.
2.5.2	Task Scheduling.....	Error! Bookmark not defined.

2.5.3	Advanced Expressions.....	Error! Bookmark not defined.
2.5.4	Programmable Port (JavaScript)	29
3	Interface.....	33
3.1	Selection Panel.....	33
3.2	Layout.....	34
3.3	Edition	35
3.4	Debug.....	36
3.5	Status	37

WBL – Logical Module

The logical module, by the elaboration of simplified logical circuits, is a powerful tool to exploit the potentialities of a domotics installation. In effect, a logical circuit allows to link and to loop several operations, which can be based on information received from sensors or manually set-up according to parameters of your choice. With the aid of our interface web, it is easy to construct interactive circuits, performing complex operations. The example below (Exhibit 1) illustrates how a task relatively complex can be performed easily without requiring prior knowledge in computer programming.

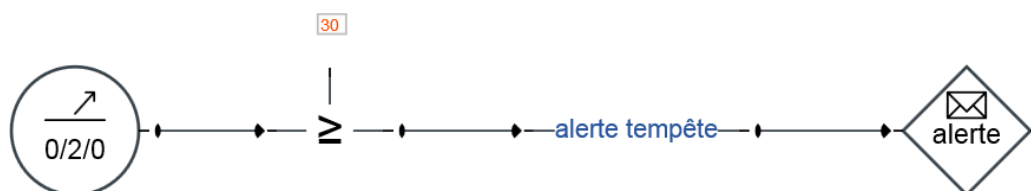


Exhibit 1 A SMS alert containing the text « storm alert » is automatically sent when the wind speed measured by the anemometer circulating on the KNX group address KNX 0/2/0 exceeds 30 m/s

Main functionalities offered by the logical module :

- **Edition/Import/Export of logical circuits** (automatisms)
- **GPIO** (General Purpose Input/Output) supported according to hardware (Raspberry)
 - Reading of the inbound bits (Input)
 - Writing and reading of the outbound bits (Output)
- **KNX**
 - Import of KNX addresses configured from **ETS**
 - Connection to KNX buses through **KNX IP or USB gateways**
 - **Multi-gateways** : can be connected and interact simultaneously with multiple KNX buses
 - Reading of communication telegrams passing through the KNX bus
 - Detection and automatic addition of addresses communicating on the KNX bus
 - Writing and dispatch of reading requests of reading on the group addresses (e.g. 1/2/3)
 - Resetting of a participant through its physical address (e.g. 1.1.1)

This manual is divided into three parts :

- Chapter 1: instructions for the installation of a logical module
- Chapter 2: complete explanation of the circuits and their operation
- Chapter 3: description of the web interface of the logical module

1 INSTALLATION

1.1 LOCAL IP

The installation of the device is performed as for a router. At the start the default IP address of the device is 192.168.1.99. Once the device has been connected to the network, you can access its interface using a web browser (IE/Firefox/Chrome/Safari/..) and entering the following address

<http://192.168.1.99>

1.2 ACTIVATION

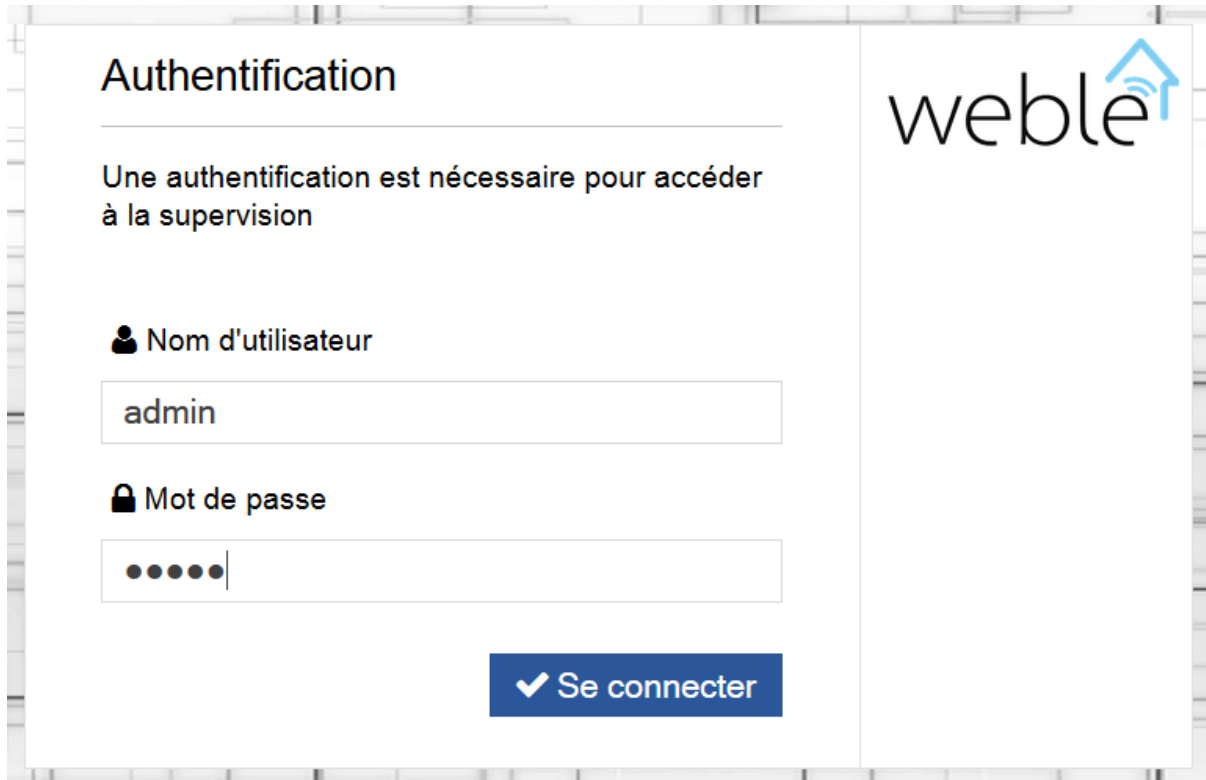
To use the module, you have to activate it (Exhibit 2). Its « online » activation can be made using a « client code » provided together with the product. It is also possible to activate the device offline using a « physical license ». However, the physical license does not give access to updates and patches.

The screenshot displays the web interface for activating a Weble device. On the left, a blue sidebar contains two icons: a network diagram labeled 'Enregistrement' and a router icon labeled 'Réseau'. The main content area is titled 'Activation en ligne' and includes the text: 'Afin de pouvoir bénéficier des mises à jours de votre appareil, vous devez l'enregistrer en ligne avec votre code client.' Below this, there is a section for 'Entrez votre code client' with a text input field labeled 'Entrez la licence' and a blue button labeled '✓ Valider la licence'. The next section is titled 'Activation hors ligne' and includes the text: 'Si vous ne disposez pas de connection internet, il vous est possible d'entrer la licence matérielle fournie avec votre appareil.' Below this, there is a section for 'Licence matérielle' with a text input field labeled 'Entrez la licence' and a blue button labeled '✓ Valider la licence'.

Exhibit 2 The product activation can be made online or offline. However, to get updates and patches, you have to use the online activation.

1.3 AUTHENTICATION

The access to the device is protected by a username and password. At the first log-in, the default administrator credentials are: username « admin » and password « admin » as well (Exhibit 3). The administrator will then be able to manage at his will the user accounts from the interface.



The screenshot shows a web browser window displaying the 'Authentication' page of the 'weble' interface. The page has a white background with a blue header bar containing the 'weble' logo. The main content area is white and contains the following elements: a title 'Authentication' in bold black text, a message 'Une authentification est nécessaire pour accéder à la supervision' in black text, a label 'Nom d'utilisateur' with a user icon, a text input field containing 'admin', a label 'Mot de passe' with a lock icon, a password input field with five dots, and a blue button with a white checkmark and the text 'Se connecter'.

Exhibit 3 Once completed the activation, insert the default administrator credentials: username "admin", password "admin"

2 WBL CIRCUITS

The WBL circuits are an adaptation of standard logical circuits. They take and extend their functionalities by using simple but effective mechanisms.

The circuits are composed of ports (Section 2.1) connected together by directional links (Section 2.2), carrying values (Section 2.3) and hence triggering the execution.

The interaction with the sensors and the actuators of the KNX bus or the digital inputs /outputs (GPIO) are done through an address system (Section 2.4).

The advanced applications are showed on the last section of this chapter (Section 2.5). Their utilization requires a more expert knowledge, but allows to treat any kind of situation.

2.1 PORTS

The circuits include around twenty different ports (Table 1), each having several configurable parameters. This flexibility of configuration allows to easily create complex automatisms.

The ports are divided into the following categories :

- **Logical ports :**
 - **Standard logical ports** (Section **Error! Reference source not found.**) : basic logical functions as *OR/AND*.
 - **Extended logical ports** (Section **Error! Reference source not found.**) : functions apt to manipulate logical values (**1/0**) and, more generally, numerical values (i. e. **1.233**)
- **Triggers** (Section **Error! Reference source not found.**) : the triggers wait the realization of an event to start the execution of actions. Moreover, they allow to perform scheduled tasks or to capture telegrams passing on the KNX bus.
- **Actuators** (Section 2.1.4) : the actuators allow performing concrete actions as writing on the KNX bus and sending SMS and/or emailing alerts.
- **Flow controllers** (Section 2.1.5) : the flow controllers act as referrals, helping to control and synchronize the execution flows on the circuit.
- **Other ports** (Section 2.1.6) : this last category includes annotations (visual and informative effects) and more advanced ports as the programmable port (Section 2.5.4).

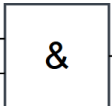
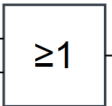
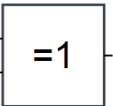
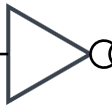

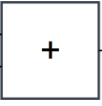
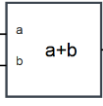
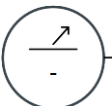
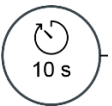

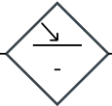
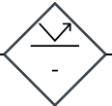
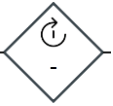

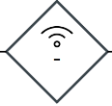



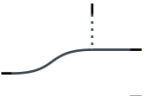
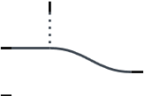
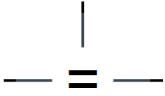





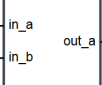
LOGICAL PORTS	 AND	 OR	 Exclusive OR	 Reverse
	 Constant	 Computation	 Expression	
ACTUATORS	 Address	 Timer	 Reception	
	 Writing	 Reading	 Restart	 Alert
TRIGGERS	 Release			
FLOW CONTROLLERS	 Junction	 Disjunction	 Switching	 Outputs Selection
	 Inputs Selections	 Filter	 10 s Delay	 Buffer
OTHER PORTS	Annotation example Annotation	 Group	 Merge	 Split
	 Programmable			

Table 1 Overview by category of all available ports.

2.1.1 Standard Logical Ports

The standard logical ports are the graphical representation of basic logical functions. Performing these logical operations, you can check whether certain conditions have been complied with before initiating an action.

Traditionally, in the Boolean algebra, both input and output data can only have two values : **1/0**. Table 2 presents the ports AND, OR, and exclusive OR as well as their truth tables, describing the output value according to the input values.

As you can see on Exhibit 4, it is possible to reverse the inputs and outputs of these ports as well as include additional inputs.

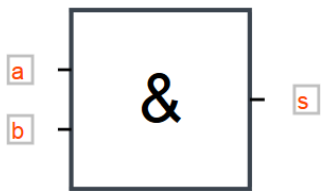
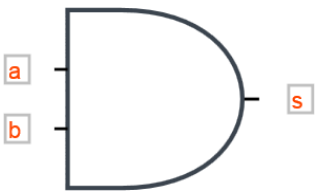
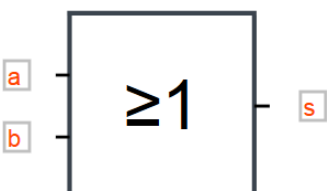

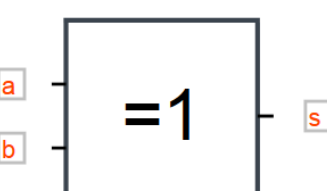

LOGICAL FUNCTION	EUROPEAN REPRESENTATION	AMERICAN REPRESENTATION	TRUTH TABLE															
AND ($n \rightarrow 1$) Send 1 as output only if all inputs equal 1			<table><tr><th>a</th><th>b</th><th>s</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	s	0	0	0	0	1	0	1	0	0	1	1	1
a	b	s																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR ($n \rightarrow 1$) Send 1 as output only if at least one of the inputs equals 1			<table><tr><th>a</th><th>b</th><th>s</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	s	0	0	0	0	1	1	1	0	1	1	1	1
a	b	s																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
Exclusive OR ($n \rightarrow 1$) Send 1 as output if only one of the inputs equal 1			<table><tr><th>a</th><th>b</th><th>s</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	s	0	0	0	0	1	1	1	0	1	1	1	0
a	b	s																
0	0	0																
0	1	1																
1	0	1																
1	1	0																

Table 2 Definition of the standard logical ports. The European and American formats are both supported by the interface.

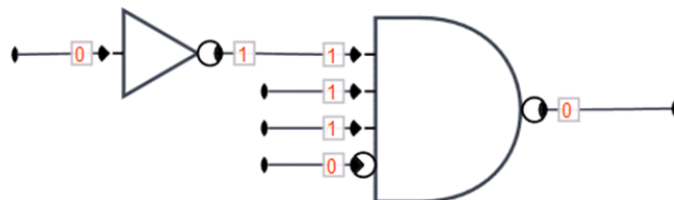


Exhibit 4 Shows how it is possible to invert the inputs and the outputs of the logical ports, either adding a reverser or acting directly on the inputs and outputs of the port. Note that here two additional inputs have been added to the AND port.

2.1.2 Extended Logical Ports

The logical circuits not only can treat Boolean values. Also, the links connecting the different ports can carry other kinds of values, such as numbers (integers or rationales) or text strings. The extended logical ports, described in Table 3, represent a solution when the standard logical ports cannot satisfy your needs.

DESCRIPTION OF THE LOGICAL PORT	REPRESENTATION
<p>INVERTER ($1 \rightarrow 1$)</p> <p>The inverter port allows inverting the value circulating on the link. 3 different kinds of inversion are possible. The binary reverse is 0 if the input is different from 0, and 1 if the input equals 0. The additional inversion is the equivalent of resending on the output $-x$, where x is the input value. Similarly, the multiplicative inversion equals to resend $1/x$</p>	
<p>CONSTANT ($1 \rightarrow 1$)</p> <p>The constant overwrites the value currently circulating on the link, by setting a constant, default value on the parameters of the port.</p> <p>Parameters</p> <ul style="list-style-type: none"> Constant value: any kind of value (binary, numerical, string ...). 	
<p>COMPUTATION ($n \rightarrow 1$)</p> <p>The computation port performs a sum (+), a multiplication (\times), an arithmetic mean (μ), or even finds the minimum value (min) or the maximum (max) within the inputs.</p> <p>The comparison operators $<$, $>$, \leq, \geq, $=$, \neq can also be used. In the case of a comparison, the output provides the result on a binary form : 1 (<i>true</i>) / 0 (<i>false</i>).</p> <p>Note : the sum (+) has a double use as it allows also to concatenate different strings (Exhibit 5).</p> <p>Parameters</p> <ul style="list-style-type: none"> Operation : sum (+) / multiplication (\times) / average (μ) / maximum (max) / minimum (min) / less than ($<$) / more than ($>$) / smaller or equal than (\leq) / more or equal than (\geq) / equal ($=$) / different (\neq) 	

DESCRIPTION OF THE LOGICAL PORT	REPRESENTATION
<p>EXPRESSION ($n \rightarrow 1$)</p> <p>The expression port is less constrained than the computation port, and supports the creation of mathematical equations. You can assign to each input an identifier and hence refer to them in the computation. However, the functionality of this port is not limited just to mathematical formulas, it can perform a variety of other operations (see Section 2.5.2).</p> <p>Parameters</p> <ul style="list-style-type: none"> Expression : here we put the formula $(a+b)/c$ 	

Table 3 Overview of extended logical ports.

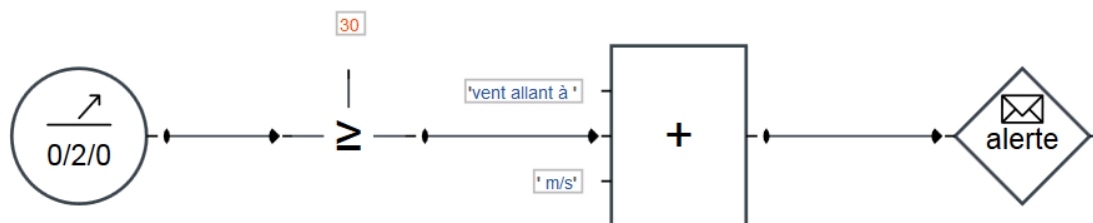


Exhibit 5 The operator « + » of the computation port allows to concatenate strings and hence to send by SMS/email the message “wind speed at 31 m/s” when the wind speed is measured at 31 m/s

2.1.3 Triggers

Through the use of triggers, you can react to external events (for instance, to telegrams circulating on the KNX module). When the event happens then the output of the trigger is executed. By convention, all triggers are represented as a circle. Table 4 describes the triggers.

DESCRIPTION OF THE TRIGGER	REPRESENTATION
<p>TIMER ($0 \rightarrow 1$)</p> <p>The timer starts a new execution sending a 1 on the link at regular intervals of x time.</p> <p>Parameters</p> <ul style="list-style-type: none"> Repeat : yes / no Timeframe : second / minute / hour / day Delay : number of units of time (e.g. 10 secondes) 	

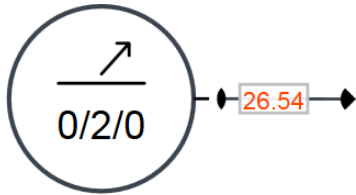
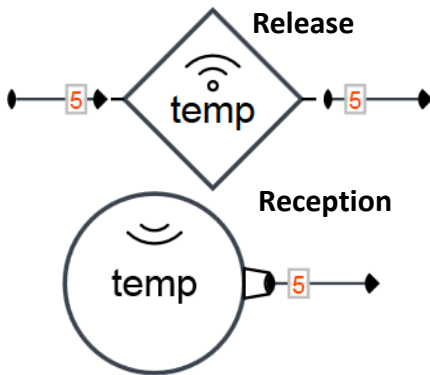
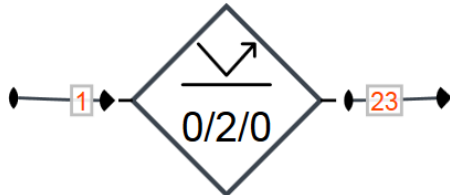
DESCRIPTION OF THE TRIGGER	REPRESENTATION
<p>ADDRESS (0 → 1)</p> <p>This port reads on the bus the telegrams regarding the address configured on the KNX group address. It sends to the output the last read values.</p> <p>Parameters :</p> <ul style="list-style-type: none"> • Send value at boot: yes/no • Play mode: each value / each value change • Read address: KNX group address 	
<p>RECEPTION (0 → 1)</p> <p>The split of the « release » and « reception » ports allows to create wireless links. The reception port captures the values transmitted on the channel given by the configured identifier. As soon as the value is received on the channel, this is carried to the output of the reception port.</p> <p>Parameters</p> <ul style="list-style-type: none"> • Propagation: Internal / External. Defines whether values transmitted by all circuits are captured (External) or, instead, only those transmitted from the same circuit (Internal) • Channel: text key identifying the reception channel (here "temp") 	

Table 4 Overview of the triggers.

2.1.4 Actuators

Concretely, the actuators are used to read or write on the KNX bus, make a reset of a KNX actuator/sensor, or even send SMS/email alerts. Normally, all actuators are represented as a rhomb. Table 5 describes the main actuators.

DESCRIPTION OF THE ACTUATORS	REPRESENTATION
<p>READING (1 → 1)</p> <p>As soon as the reading port receives a new input signal, it reads the value on the KNX group address and sends it to the output.</p> <p>Parameters</p> <ul style="list-style-type: none"> • Read mode : read request on the bus / last known value • Reading address : KNX group address 	

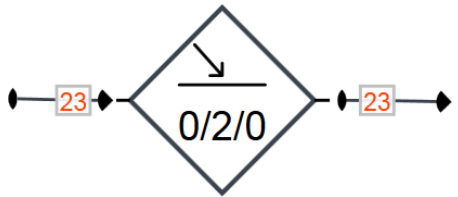
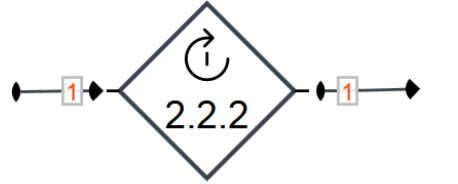

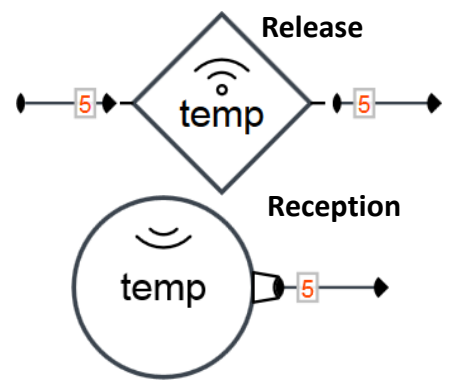


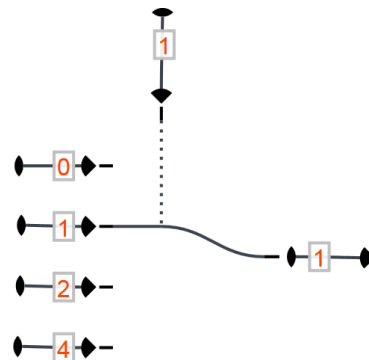
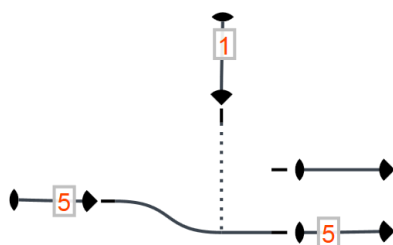
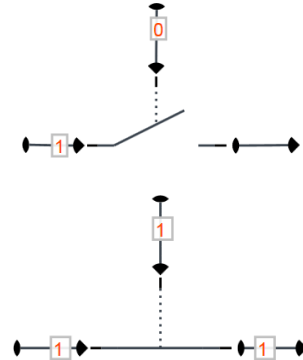
DESCRIPTION OF THE ACTUATORS	REPRESENTATION
<p>WRITING (1 → 1)</p> <p>The reading actuator takes the input value to be written on the configured group address. If the telegram has been correctly sent to the bus, the writing value is transcribed on the output.</p> <p>Parameters</p> <ul style="list-style-type: none"> • Writing address : KNX group address 	
<p>RESET (1 → 1)</p> <p>This actuator sends a <i>reset</i> KNX telegram to a participant (physical address). Useful for the troubleshooting of a sensor/actuator. If the sensor/actuator succeeds to start, the input value is replicated on the output.</p> <p>Parameters</p> <ul style="list-style-type: none"> • Participant address : address of the KNX participant 	
<p>ALERT (1 → 1)</p> <p>The alert allows to emails and/or SMS to alert the responsible person in case of urgency or to release externally information relating to the domotics installation. It takes on the input the text to be sent.</p> <p>Parameters</p> <ul style="list-style-type: none"> • Recipients : phone number / email address • CLIP / Object : title of the SMS / email object 	
<p>RELEASE (1 → 1)</p> <p>The split of the ports « release » and « reception » allows to create wireless links. When it receives an input value, the reception actuator retransmits it on the channel.</p> <p>Parameters</p> <ul style="list-style-type: none"> • Propagation: Internal / External. Defines whether values transmitted by all circuits are captured (External) or, instead, only those transmitted from the same circuit (Internal) • Event identifier : text key, here "temp" 	

Table 5 Description of the different actuators.

2.1.5 Flow Controllers

The ports controlling the execution flow are at the heart of the circuits. They play a fundamental role of referral and influence how the values are treated and spread over the circuit. Table 6 illustrates the flow

controllers and the Exhibits below (Exhibit 6, Exhibit 7, Exhibit 8) provide some practical examples of their utilisation.

DESCRIPTION OF THE FLOW CONTROLLERS	REPRESENTATION
JUNCTION ($n \rightarrow 1$) <p>The junction ties several inputs on the same output. The input values received are replied instantaneously on the output.</p>	
DISJUNCTION ($1 \rightarrow n$) <p>When an input value is received, this is replicated and sent simultaneously to the different outputs.</p>	
INPUTS SELECTION ($n \rightarrow 1$) <p>The upper input of the referral allows to select the operating input. The left inputs are indexed starting from 0. For instance, if the referral is configured as 1, then the second input is selected.</p> <p>Parameters</p> <ul style="list-style-type: none"> Synchronisation: yes/no. Defines if the whole of the inputs must have a certain value in order to allow the continuation of the execution (see Section 2.2.4). 	
OUTPUT SELECTION ($1 \rightarrow n$) <p>The referral input allows to select the chosen output. For instance, if the referral is set as 1, then the value of the second input (counting from the top towards the bottom, starting from 0) is redirected on the output.</p>	
SWITCH ($1 \rightarrow 1$) <p>If the input of the push button is 1, then the execution of the standard input (on the left) can be transmitted to the output. Instead, if the push button is set as 0, the execution is stopped.</p>	

DESCRIPTION OF THE FLOW CONTROLLERS	REPRESENTATION
<p>FILTER ($1 \rightarrow 1$)</p> <p>The upper input is compared to the standard input (left), and if the condition of the filter is not satisfied, then the execution will be stopped.</p> <p>Parameters</p> <ul style="list-style-type: none"> Comparator: smaller ($<$) / bigger ($>$) / smaller or equal (\leq) / bigger or equal (\geq) / equal ($=$) / different (\neq) 	
<p>DELAY ($1 \rightarrow 1$)</p> <p>The delay port delays of a defined timeframe the transmission of the execution. Different reactions are possible when the port is reactivated by a new value before the previous delay has expired.</p> <p>Parameters</p> <ul style="list-style-type: none"> Delay timeframe: millisecond/second/minute/hour/day Delay: number of timeframes Associated inputs: Ignore the new input/ Restart the delay/ Execute the delays/ Cumulate the delays/ Execute the delays simultaneously 	
<p>BUFFER ($1 \rightarrow 1$)</p> <p>The buffer port allows to access to values previously passed through the link. As soon as a new input comes, it records the new value and sends the previous value to the output, thus introducing a lag of the outputs compared to the inputs.</p> <p>Parameters</p> <ul style="list-style-type: none"> Number of memories : number of values stored between the input and the output. 	

Table 6 Description of the flow controllers.

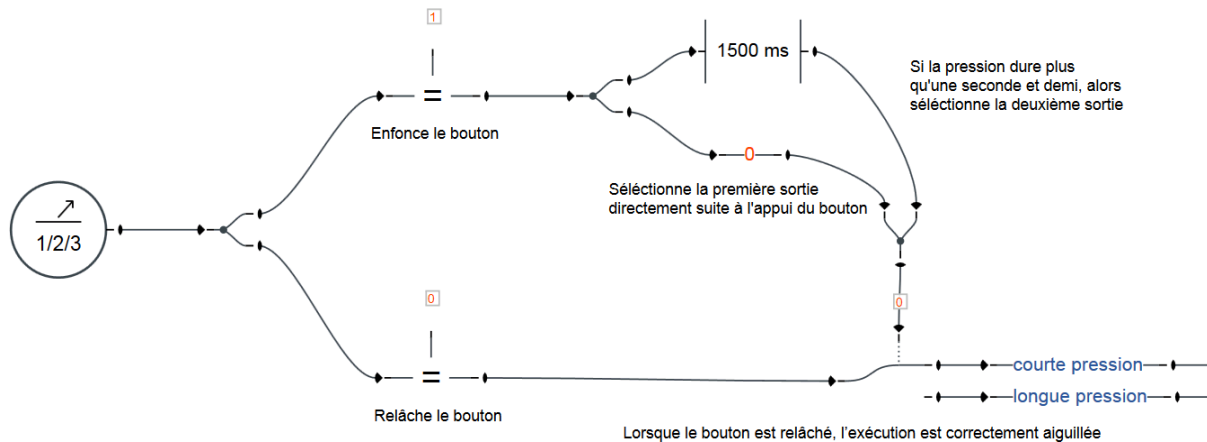


Exhibit 6 The circuit distinguishes a long support from a short one. In this exhibit, the push button communicates on the KNX group address 1/2/3. A 1 is sent when the button is pressed and a 0 when the button is released.

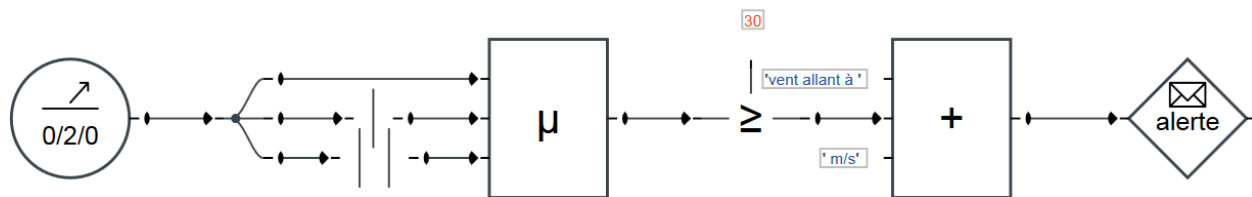


Exhibit 7 Sends the alert only if the mean of the last 3 latest wind speed checks is more than 30 m/s.

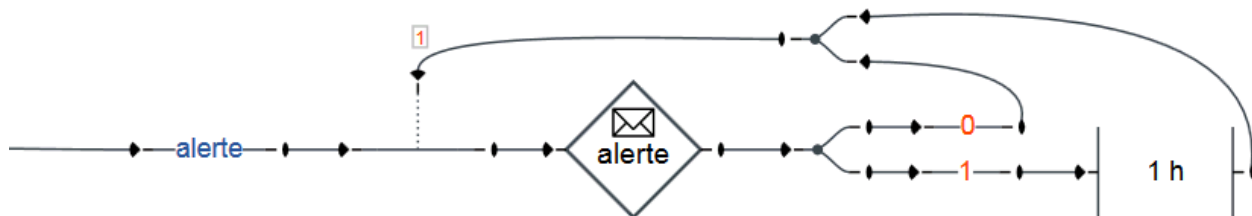


Exhibit 8 Limits the number of sent SMS/emails to one per hour.

2.1.6 Other ports

This category includes some annotation ports (with just a representation and informative aim) and more advanced ports as the programmable port (Section 2.5.4). Table 7 describes those ports.

DESCRIPTION OF THE PORT	REPRESENTATION
ANNOTATION Use it to add simple informative text to the logical circuits. Parameters <ul style="list-style-type: none"> • Colour, Font, Interline, Bold, Spacing, Gras, Underlined 	<div>Exemple d'annotation</div>

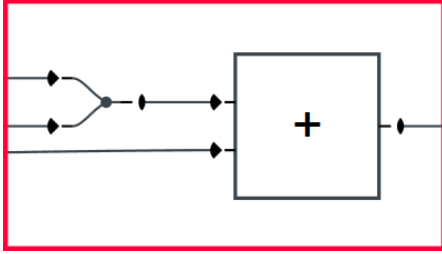
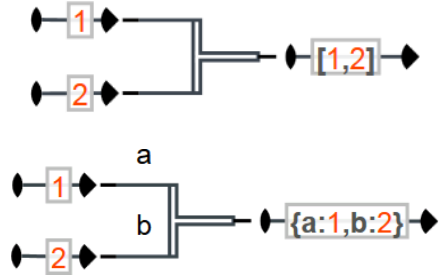
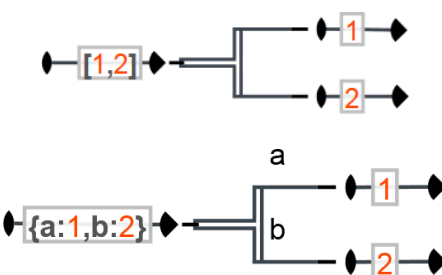
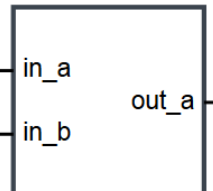
DESCRIPTION OF THE PORT	REPRESENTATION
<p>GROUP</p> <p>Use it to group visually different parts of the logical circuit.</p> <p>Parameters</p> <ul style="list-style-type: none"> • Colour, Border, Border Width 	
<p>MERGE ($1 \rightarrow 1$)</p> <p>Merges together the values received in input on a table (see Section 2.3.3).</p> <p>Parameters</p> <ul style="list-style-type: none"> • Association : yes/no. Defines if a table is created at the output of the port. If yes, then the inputs are identified by configurable keys. • Synchronisation : yes/no. Defines if all outputs have to be defined in order to continue the execution (see Section 2.2.4). 	
<p>SPLIT ($1 \rightarrow 1$)</p> <p>Splits the values of a table (see Section 2.3.3) and sends it to the different outputs.</p> <p>Parameters</p> <ul style="list-style-type: none"> • Associative : yes/no. Defines whether an associative table is expected. If yes, keys are introduced in order to access the elements of the associative table. 	
<p>PROGRAMMABLE ($n \rightarrow m$)</p> <p>The programmable port (Section 2.5.4) is an advanced port whose logical functionalities can be implemented by using the programming language. Its inputs/outputs and its parameters are totally configurable.</p>	

Table 7 Description of the ports pertaining to the category « other ports ».

2.2 LINKS

The links connect the different ports and transmit the execution through the circuit.

There are three kinds of links:

- Memory links (Section 2.2.1)
- Passive memory link (Section 2.2.2)
- Consumable link (Section **Error! Reference source not found.**)

These types of links are apt to the synchronisation of the execution flows (Section 2.2.4)

2.2.1 Memory Link



The memory link stores always its current value. Once executed, the old value is overwritten by the new one.

2.2.2 Passive Memory Link



The passive memory link is similar to the normal memory link, but it transmits the execution only if the new value received is **different** from the old one. For instance, switch of **0** → **1** (incremental flow) or of **1** → **0** (reducing flow).

2.2.3 Consumable link

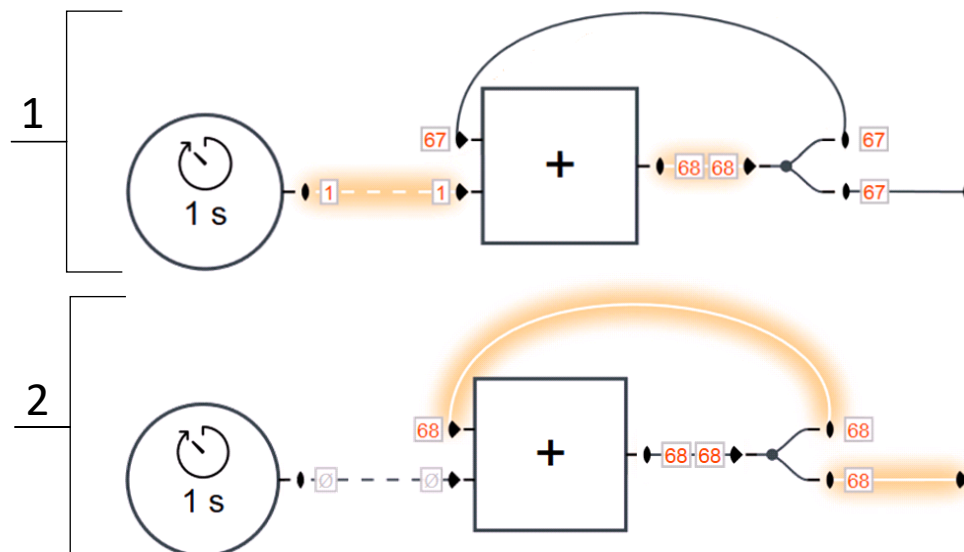


The links connecting the ports can be empty and miss any value. On this case, the absence of values is marked with the sign \emptyset . The consumable link, differently from the memory links, loses its value (becomes \emptyset) when the value is consumed by the targeted port.

2.2.4 Synchronization

Before the execution, the ports have to wait that all output values have been correctly defined. That is, if one of the inputs of the port is \emptyset , then the execution flow is stuck in pause.

The application of this rule associated to the different kinds of links represents a simple but effective synchronisation mechanism. For instance, this allows the creation of a counter as shown in Exhibit 9.



*Exhibit 9 This exhibit details the execution of a counter incremented by **1** at every second. Note that the consumable link loses its value once the addition has been performed. This forces the computation port to wait the following impulse before acting another increment of the counter (to avoid a never-ending loop). The memory link stores the current value of the counter (here **68**).*

2.3 VALUES

The circuits are not limited just to Boolean values (1/0). Three types of values can transit through the links :

- Numbers : 1.23 (Section 2.3.1)
- Strings : example of string (Section **Error! Reference source not found.**)
- Value tables : [0, 1, example of string] (Section 2.3.3)

As for the traditional logical circuits, these values can be reverted on the inputs and outputs of the ports (Section 2.3.4).

2.3.1 Numbers

The internal representation of the numbers is always in a double-precision binary floating-point format (64 bits) according to the international standard IEEE 754.

Regarding the syntax, many different notations are possible :

- Relative numbers : 0.01, 10.00, -10.505
- Integers (base 10) : 0, 1, 2, -2, 12456
- Hexadecimal integers (base 16) : 0xFF (= 255), 0x12 (= 18), 0xA (= 10), -0xA (= -10)
- Octodecimal integers (base 8) : 001 (= 1), 00100 (= 64), 0077 (= 63), -0077 (= -63)
- Scientific notation : 1e2 (= 100), -1.2e-2 (= -0.012), 0.0023e+4 (= 23)
- Special values : PI (= 3.141...), E (= 2.718...), ∞, Infinity, NaN (Not a Number)
- Computational equations : (sin(PI/4)+0.5)*2 (= 2.414...), sqrt(5)+log(10) (= 4.538...). Available functions : http://www.w3schools.com/js/js_math.asp

2.3.2 Strings

Using the strings you can form texts : Example of string. The strings can be concatenated one another or with numbers in order to form a new string (Exhibit 5, Exhibit 7). When a string can be mistaken for a number, its inclusion in between of quotation marks (" or ') avoids ambiguities : PI → "PI", 1.23 → '1.23', sqrt(5)+log(10) → 'sqrt(5)+log(10)'

Note : particular string formats are subject to a special treatment for certain ports :

- The writing and reading addresses (Section 2.4)
- The phone numbers and the email addresses (interpreted by the « alert » port)
- The date system for the planification of the tasks (Section **Error! Reference source not found.**)

2.3.3 Tables

Tables allow to transmit several values through the same link. Two kinds of tables are available :

- The **simple table**. The simple tables are delimited by square brackets [...]. The value are separated by commas and each value is referenced according to its position on the table. Example: [1, 2, test@example.ch, [3, 4, address]].
- The **associative tables**. The associative tables are delimited by curly brackets {...}. The values are separated by commas and referenced by keys. Example : {n: 1, m: 2, c: test@example.ch, table: [3, 4, address]}.

2.3.4 Inversion of the values

By extension of the standard logical circuits, you can revert the values received as inputs of the ports as well as those sent as outputs.

There are three kinds of inversion :

- Binary inversion : $[\neq 0] \rightarrow 0, 0 \rightarrow 1$
- Additional inversion : $x \rightarrow -x$
- Multiplicative inversion : $x \rightarrow 1/x$

Value	Binary inversion	Additional inversion	Multiplicative inversion
0	1	0	∞
1	0	-1	1
2	0	-2	0.5
String	0	NaN	NaN
[2,{a:0,b:string}]	[0, {a: 1, b: 0}]	[-2, {a: 0, b: NaN}]	[0.5, {a: ∞ , b: NaN}]

Table 8 Inversions (binary, additional and multiplicative) with different values.

2.4 ADDRESSES

The « writing » and « reading » actuator ports allows to read and write on the addresses. Using logical module you can treat different address types :

- KNX addresses (Sections 2.4.1 et 2.4.2) : reads, writes, and captures KNX telegrams on the bus.
- GPIO Addresses (Section 2.4.3) : utilises the digital inputs and outputs of the hardware.
- Virtual addresses (Section 2.4.4) : logs or stores values worked internally.

2.4.1 Addresses of the KNX Group

KNX, also named Konnex, is a fieldbus and a protocol of automatism for the building. The KNX protocol is a protocol with a distributed logic. Contrarily to other automatism protocols, it does not work in a master/slave mode, as each automation is independent form the others.

The logical module accesses the KNX bus through a USB gateway, as the two types of connection are supported (Exhibit 10).

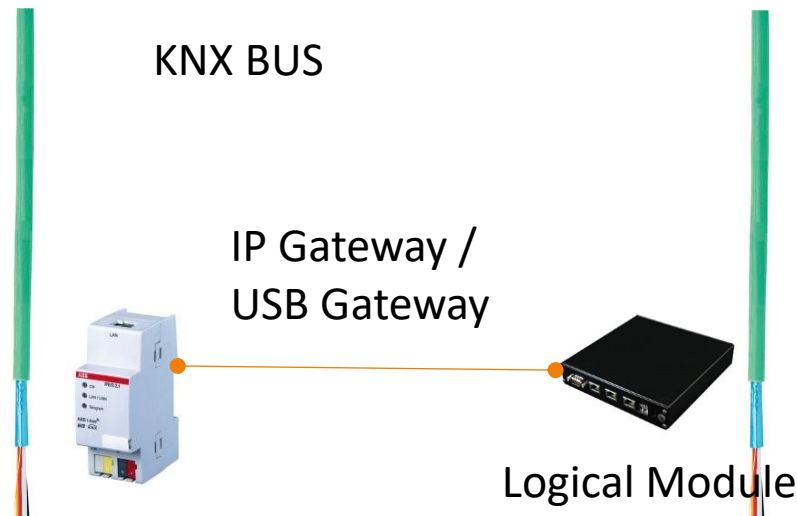


Exhibit 10 Connection of the logical module to the KNX bus through a KNX IP gateway or a KNX USB gateway

The communication between the devices is assured by telegrams sent on the group addresses. Those ones are used by the sensors and the actuators to transmit data or communicate actions. The group addresses are encoded on 2 bytes. The logical module uses the notation structured into three levels main/middle/sub : 5/3/8 bits respectively (Exhibit 11).

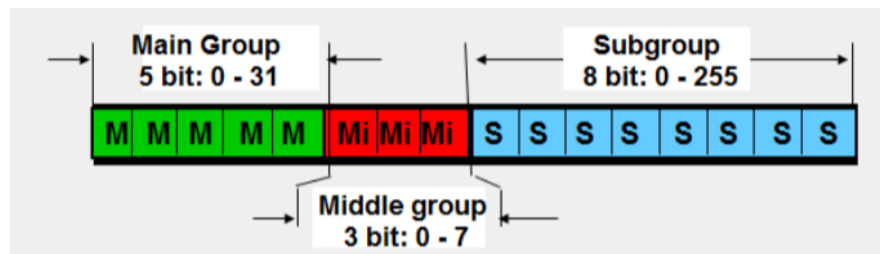


Exhibit 11 three-levels KNX encoding.

All communications transmitted on the bus amongst the different participants are captured and sent to the logical module which then can treat the information, record them, or reply to them by writing on the bus. The following actions are supported on the KNX group address.

- Reading of the telegrams flowing through the bus.
- Writing of telegrams on the bus.
- Sending a reading request on the bus and receiving the answer.

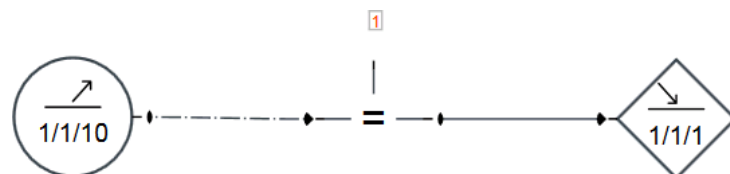


Exhibit 12 Turn on the lights (group address 1/1/1) when the value of the detector of presence (communicated on the group address 1/1/10) changes from 0 to 1 (incremental flow)

2.4.2 KNX physical addresses

The sensors and actuators joining the KNX bus always have a physical address. A physical address is of the type n.n.n (for instance 1.1.1) where the number represents a real topology. A physical address is assigned to each device during the installation. The reset of a participant can be done from a circuit using its physical address (Exhibit 13).

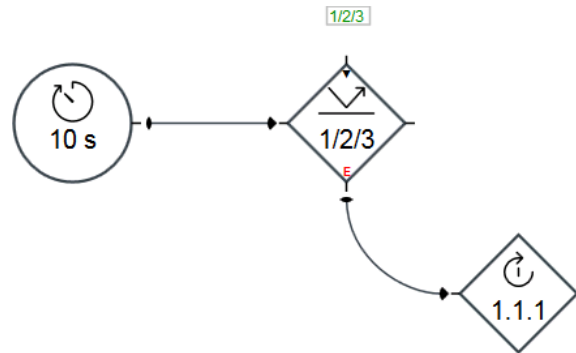


Exhibit 13 Reset of 1.1.1. Sends each 10 seconds a reading request regarding the group address 1/2/3 on the KNX bus. If no reply is received, then the error output (Section 0) is executed and the corresponding device (1.1.1) reset.

2.4.3 GPIO Addresses

If the hardware of the logical module is equipped with GPIO (General Purpose Input/Output) ports, as it is on the [raspberry](#), it can be used to communicate directly on the electrical circuit. These inputs and outputs are accessible from the logical module through the following addresses:

- **gpi** log of the digital inputs (8 bits)
- **gpo** log of the digital outputs (8 bits)

It is possible to read the status of the inputs (**gpi**) and of the outputs (**gpo**), you can not only write in the log but also read its status. The writing and reading of the inputs is done by using positive integers included between 0 et 2^n-1 , where n represents the number of the bits to be encoded (8 for a whole log). You can also write/read in only a portion of the log. For instance, to access separately to each of the bits of the **gpo** (numbered from 0 to 7), the following « sub-addresses » are available: **gpo(0)** **gpo(1)** **gpo(2)** ... **gpo(7)**. The Exhibit 14 shows the usage of the GPIO in a circuit and the Table 9 provides various examples of « sub-addresses » valid for the **gpi**, highlighting the relating bits.

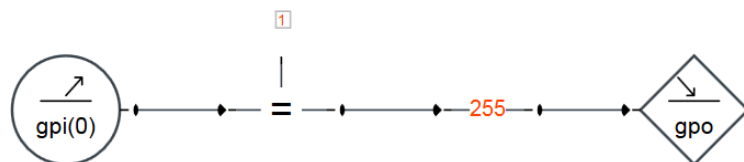


Exhibit 14 If the first bit of the **gpi** (digital inputs) changes from 0 → 1, then writes 1 on the bits of the **gpo** (digital outputs). Note that the address **gpo** is equivalent to the « sub-address » **gpo(0:7)**.

Address	gpi content	Binary value	Decimal value
gpi(1)	0 0 0 0 0 0 1 0	1	1
gpi(5)	1 1 0 1 1 1 1 1	0	0
gpi(0:7)	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0
gpi(0:7)	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	255
gpi(0:3)	1 1 1 1 0 0 1 0	0 0 1 0	2
gpi(3:0)	1 1 1 1 0 0 1 0	0 1 0 0	4
gpi(0,1,6,7)	1 1 0 0 0 0 0 1	1 1 0 1	13
gpi(2:0,5,7)	1 0 0 0 0 1 0 0	1 0 0 1	17

Table 9 Allowed « Sub-addresses » derived of the **gpi** (digital inputs) with an example of the corresponding value.

2.4.4 Virtual Addresses

Conversely to the other addresses, the virtual addresses are not transmitted to the external of the device through a field bus. They are useful internally to store values, transmit values between different logical circuits, or even save values in a registry (log file). The virtual addresses do not have a particular format: each one is identified by a string (see below label).

2.4.5 Label

No matter the nature of an address (KNX, GPIO, ...), you can assign a label to it, in order to access to the address directly from the circuits. Add an extra indirection level to the addressing by using labels presents several advantages :

- **Clarity:** allows to explicit the meaning of an address on the logical circuit. For instance if the KNX group address 1/1/1 is configured to turn on or off the lights of a room, it is more clear to refer to it by using a label as **light_room_1**.
- **Transferability:** allows to transfer the logical circuits from one location to another, even if the KNX addresses used are different. For instance, if the wind speed measured by an anemometer is transmitted on 0/2/0 to the location **A** and on 0/2/3 to the location **B**, then for portability reasons it is more practical, on the two cases, to access to it by using the common label « **wind** » (see Exhibit 15).
- **Multi-gateway :** allows to solve address conflicts when the logical module makes different KNX installations interact. For instance, if the module has access to two different KNX gateways (one in the gateway **A** and the other in the gateway **B**) and in each of the two buses the wind speed is measured and communicated on the group address 0/2/0, then it is possible to distinguish the addresses by assigning to each a different label (see Exhibit 16).

Location A		
Label	Address	Description
wind	0/2/0	Wind speed (m/s)

Location B		
Label	Address	Description
wind	0/2/3	Wind speed (m/s)

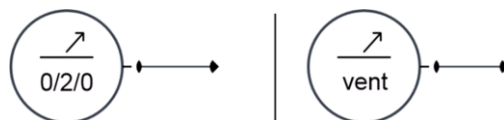


Exhibit 15 Portability, the left port refers to the KNX group of the wind (0/2/0). This port works only on the A location. However, using a label as on the right port (wind), it can be used on the two locations without having to modify the circuit.

Gateway	Label	Address	Description
KNX A	A_wind	0/2/0	Wind speed (m/s)
KNX B	B_wind	0/2/0	Wind speed (m/s)

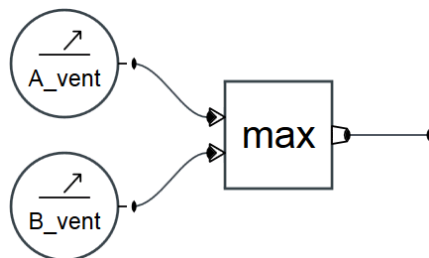


Exhibit 16 Multi-gateway. Retrieves thanks to the label A_wind and B_wind wind speeds transmitted with the same KNX address KNX (0/2/0) on two distinct gateways (gateways KNX A and KNX B). Then selects the maximum value.

2.5 ADVANCED FUNCTIONALITIES

The command of the advanced features requires more skills but allows to operate in all kinds of situations. The following topics will be covered :

- The advanced mode (Section **Error! Reference source not found.**), which adds to certain ports inputs/outputs and so provides a more precise and dynamic control of the inputs/outputs.
- The tasks scheduling (Section **Error! Reference source not found.**), which explains how to schedule unique or recurrent tasks.
- The advanced expressions (Section 2.5.2). This section shows more powerful ways of using the « expression » port.
- The programmable port (Section 2.5.4), with which you can create your own tailored ports in JavaScript.

2.5.1 Advanced Mode

The advanced mode, when it is activated on a port, adds inputs/output to the port. On this way, you can for instance dynamically control the port : enable it / disable it / pause it / / change its execution parameters. The following ports support the advanced mode :

- The triggers (see Exhibit 17)
- Les actuators (see Exhibit 18, Exhibit 19)
- The delay port (see Exhibit 20)

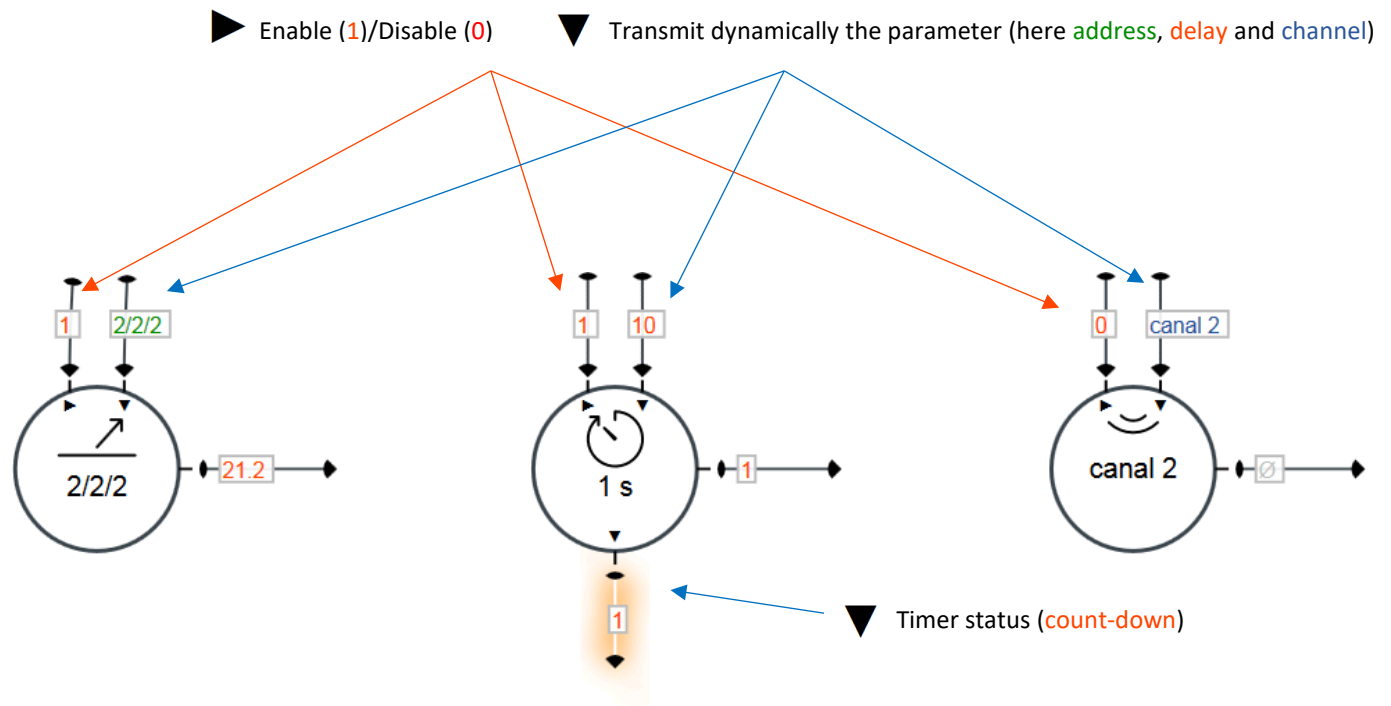


Exhibit 17 Triggers in advanced mode.

▼ Transmit a parameter dynamically (here **address**, **recipient** and **channel**) **E** Error output if the action fails

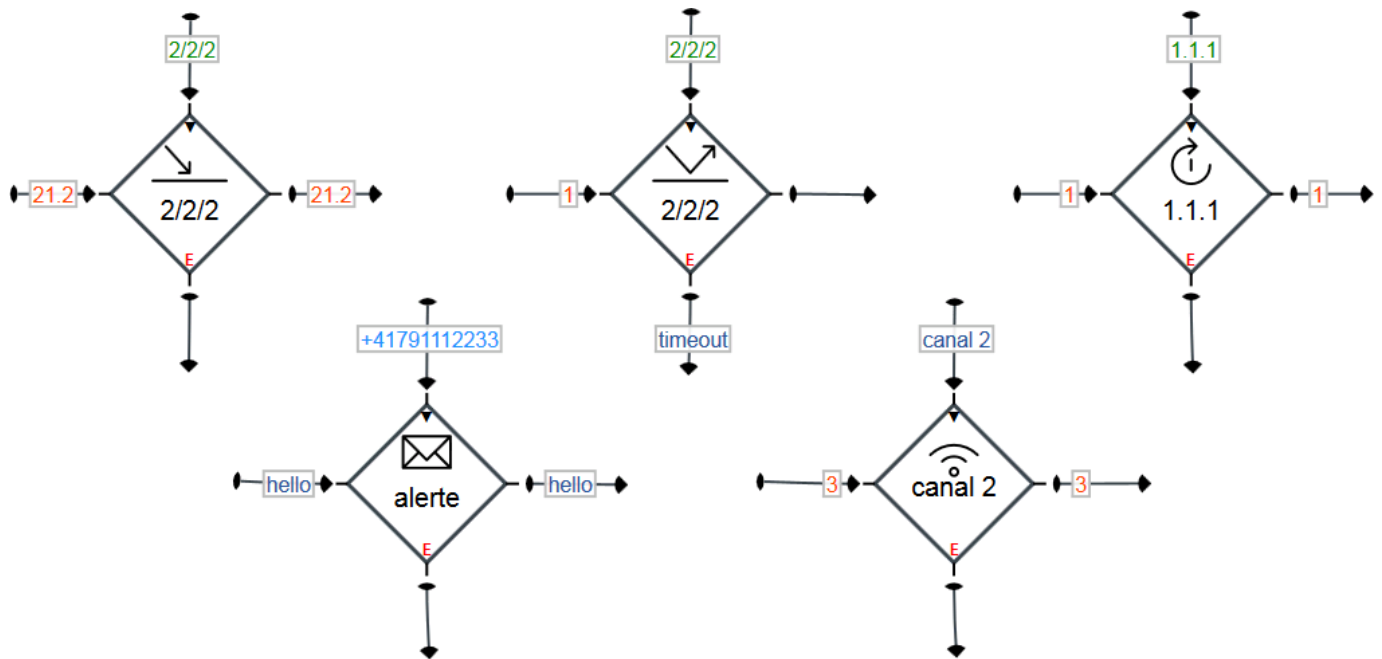


Exhibit 18 Actuators in advanced mode.

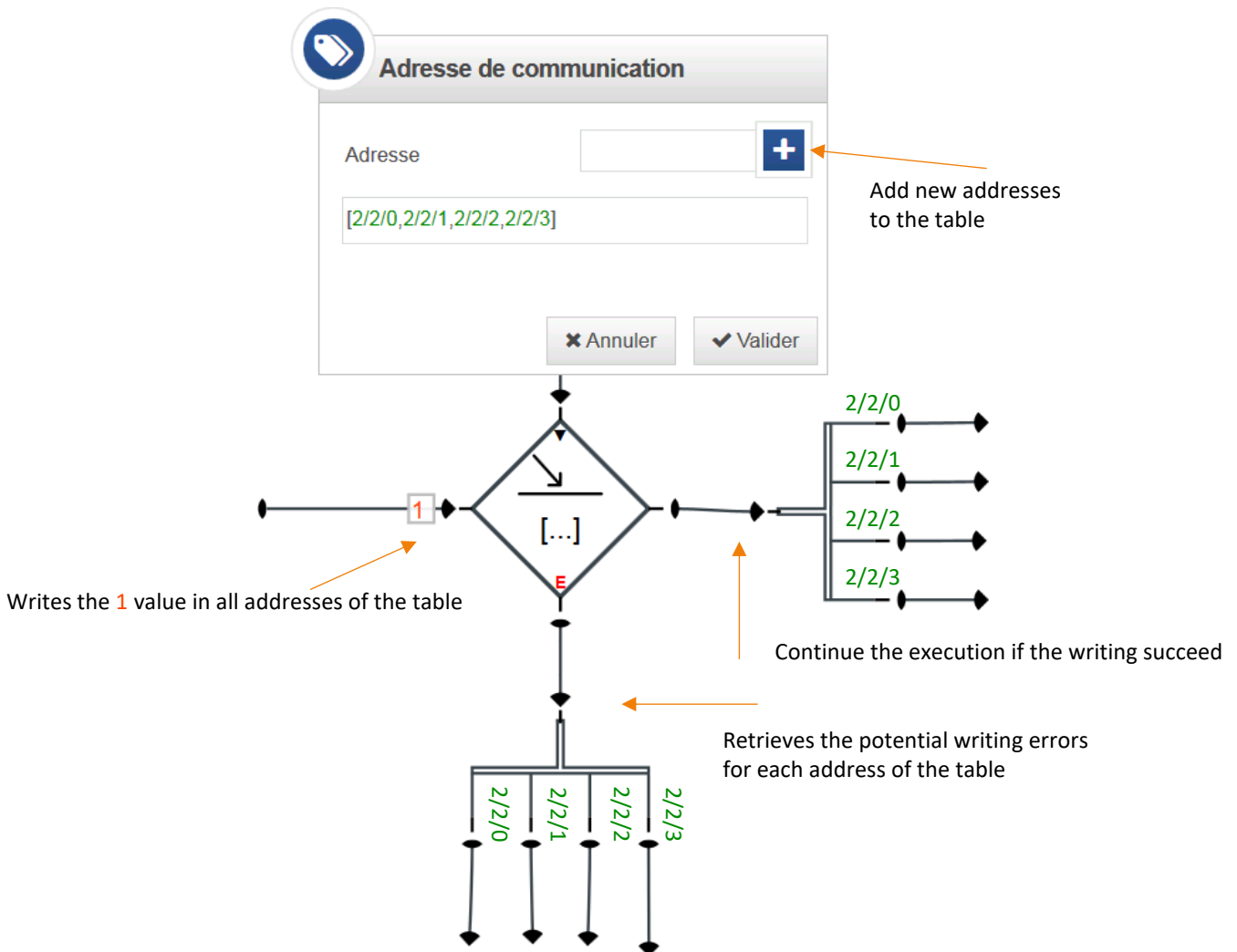


Exhibit 19 Multiple writings. Writes a value (here 1) on a number of multiple addresses using a table of addresses (here [2/2/0,2/2/1,2/2/2,2/2/3]). By exploiting the « split » and the advanced mode, you can retrieve the result of the writing (success or error) for each address of the table.

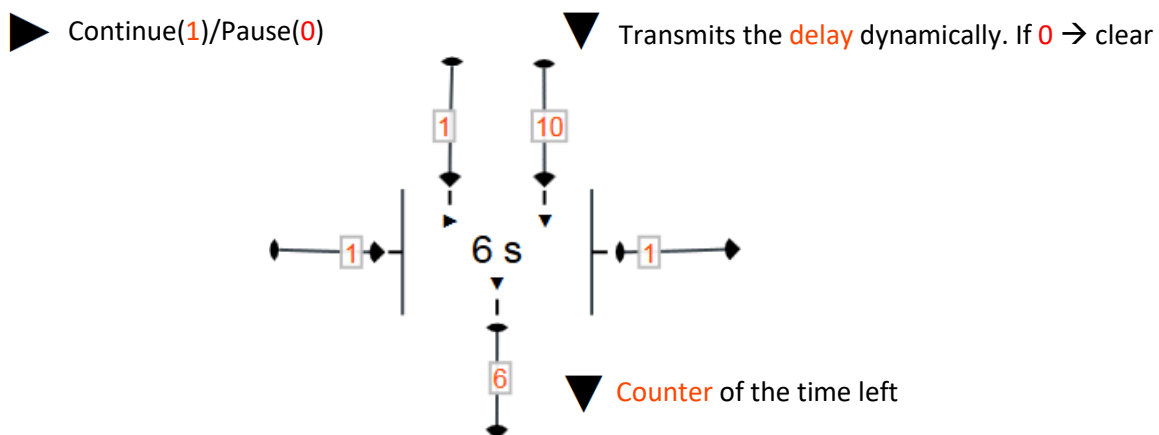


Exhibit 20 Delay in the advanced mode. Allows to pause and modify the delay dynamically.

2.5.2 Task Scheduling

The task scheduling can be done by using the « timer » port. The timer port includes different kinds of values (see Exhibit 21) :

- The **number** (positive integers), allows to repeat an action each **x** milliseconds/seconds/minutes/hours/days.
- The **dates** (Table 10), allowing the scheduling of a task at a defined date.
- The **cron** format (Table 11, Exhibit 22) which is the UNIX system of scheduling recurring tasks.

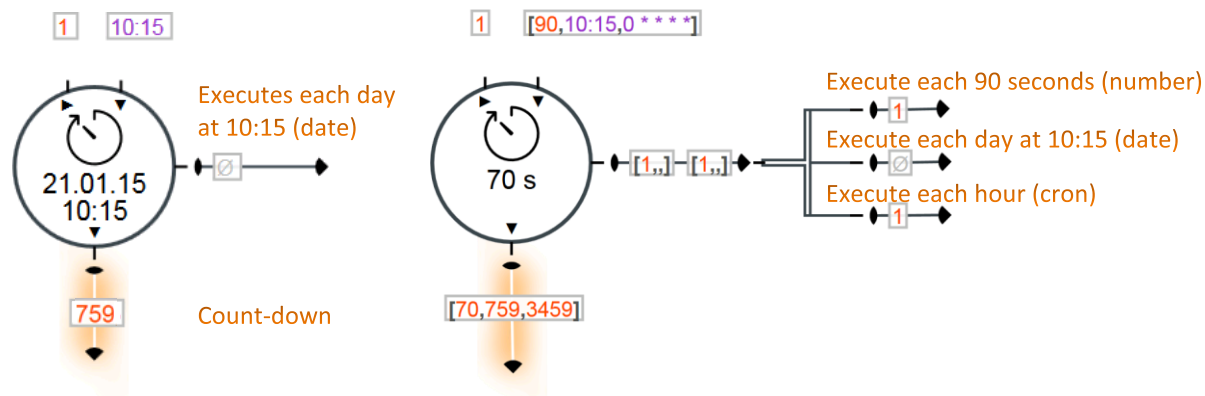


Exhibit 21 Schedule the execution of several tasks. The first example uses a simple value (10:15). In the other example a table of values ([90,10:15,0 * * * *]) is used.

Description	Long date format	Short date format
Christmas's Eve 2016 at midday	12:00:00 25.12.2016	12:00 25.12.2016
New Year's Eve in 2016	00:00:00 01.01.2016	1.1.2016
Every new year	00:00:00 01.01.* * * *	1.1.*
Every first day of the month	00:00:00 01.*.* * * * *	1.*.*
Every day at 13:00	13:00:00 *.*.*.* * * * *	13:00
Every hour	*.*.:00:00 *.*.*.* * * * *	*.:0
Every minute	*.*.*.:00 *.*.*.* * * * *	*.*
Every 10 minutes	*.*.*0:00 *.*.*.* * * * *	*.*0
Every second	*.*.*.*.* *.*.*.* * * * *	*.*.*

Table 10 Task scheduling by using the standard date format « hh:mm:ss jj.mm.aaaa ». The recurring tasks are scheduled with the joker character « * ».

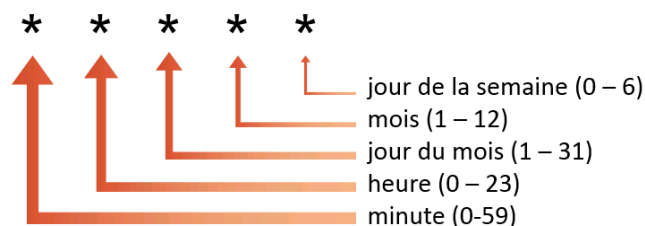


Exhibit 22 Cron format. The days of the week are numbered from Sunday (0) to Saturday (6).

Description	Cron
Execute once a year at midnight on January, 1 st	0 0 1 1 *
Executes once a month at midnight on the first day of the month	0 0 1 * *
Executes once a week at midnight, on Sundays	0 0 * * 0
Executes once a day at midnight	0 0 * * *
Executes once every hour	0 * * * *
Executes every minute	* * * * *
Executes every 5 minutes	*/5 * * * *

Table 11 Examples of cron.

2.5.3 Advanced expressions

The expression port is not limited only to simple formulas as $(a+b)/2$. Instead, an expression actually is a line of code interpreted by the JavaScript engine. This port makes possible a number of operations on the values :

- Logical operations and comparisons. For instance, $(a > b \ \&\& \ a > c) \ || \ a == 0$ gives **1** if a is bigger than b and c , or if a equals 0 . Otherwise gives **0**.
- Ternary operator: $a == b \ ? \ 10 : -10$. Gives **10** if a equals b . Otherwise gives **-10**
- Call [native JavaScript mathematical functions](#) : $\text{acos}(x)$ $\text{asin}(x)$ $\text{atan}(x)$ $\text{atan2}(y,x)$ $\text{ceil}(x)$ $\text{cos}(x)$ $\text{exp}(x)$ $\text{floor}(x)$ $\text{log}(x)$ $\text{max}(x,y,z,...,n)$ $\text{min}(x,y,z,...,n)$ $\text{pow}(x,y)$ $\text{random}()$ $\text{round}(x)$ $\text{sin}(x)$ $\text{sqrt}(x)$ et $\text{tan}(x)$
- Call [string functions](#) : $\text{charAt}()$ $\text{concat}()$ $\text{indexOf}()$ $\text{lastIndexOf}()$ $\text{match}()$ $\text{replace}()$ $\text{search}()$ $\text{slice}()$ $\text{split}()$ $\text{substr}()$ $\text{substring}()$ $\text{toLowerCase}()$ $\text{toUpperCase}()$ $\text{trim}()$...
- Use [regular expressions](#) : $\text{/raclette}s+\text{magistrale/}.\text{test}(\text{"Je mange une raclette magistrale"})$ gives **1**.
- Call functions to work on [tables](#) : $\text{concat}()$ $\text{every}()$ $\text{filter}()$ $\text{join}()$ $\text{map}()$ $\text{reduce}()$ $\text{reduceRight}()$ $\text{reverse}()$ $\text{slice}()$ $\text{some}()$ $\text{sort}()$... Exhibit 23 shows how to work on a table of values using expressions.
- Etc..

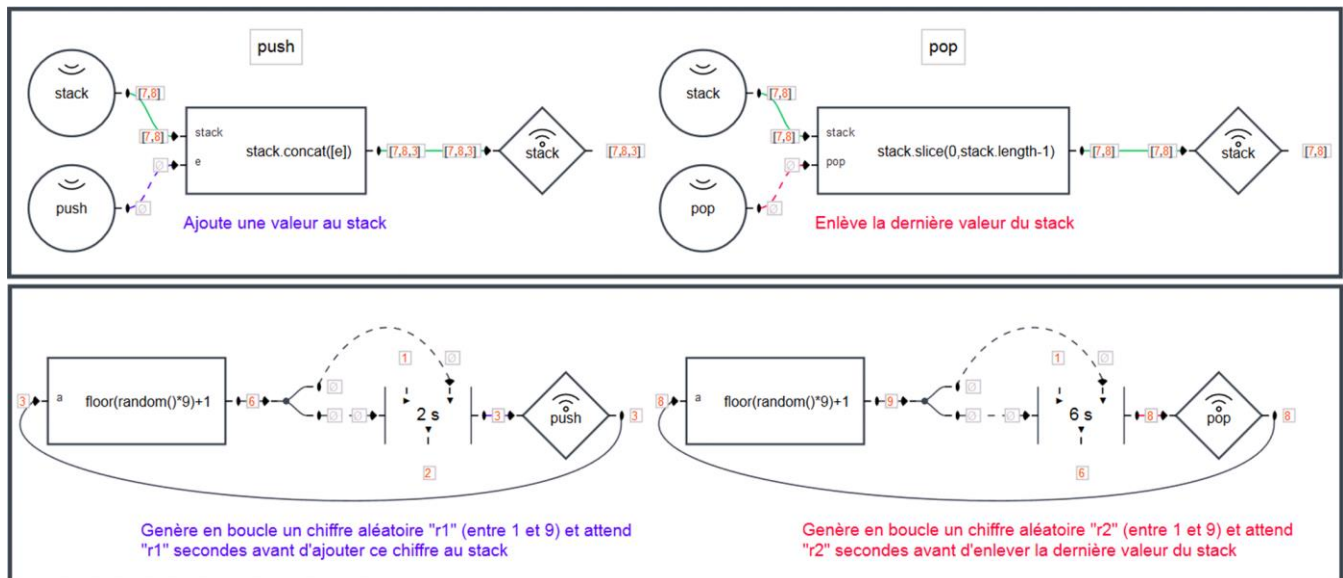


Exhibit 23 The upper part illustrates an interesting way of implementing a « stack » : a pile of values by using a « push » function, adding a new value and a « pop » function, removing the last value. In the lower part, values are randomly piled on the stack and then removed from the stack. Note that the « expressions » ports uses native JavaScript functions `concat()`, `slice()`, `floor()`, and `random()` to build the stack and generate random figures.

2.5.4 Programmable port (JavaScript)

Create its own customized ports using the programming language offers more flexibility. In addition, the ports you created can be reused in different projects through the import/export tools of the interface.

- Unlimited number of inputs and outputs.
- Unlimited number of configurable parameters.
- Access to [JavaScript](#) native functionalities (Object, Array, Buffer, Math, string, ...).
- Access to the following [node.js](#) modules : [assert](#), [buffer](#), [crypto](#), [dgram](#), [dns](#), [domain](#), [events](#), [http](#), [https](#), [later](#), [net](#), [os](#), [path](#), [stream](#), [string_decoder](#), [suncalc](#), [tty](#), [url](#), [util](#), [zlib](#).
- [Ace](#) code editor, integrated with the automatic detection of errors and auto-completion.
- Console showing information useful for the debugging.

Exhibit 24 exhibits the basic implementation of a counter and Table 12 explains how to access to the value of an input and execute an output from the code of the programmable port.

Exhibit 25 represents a more complex but very useful example in domotics : the implementation of a port computing the time of the sunrise and sunset starting from information about the latitude and the longitude. This port is practical to execute actions at particular moments of the day (raise the curtains in the morning, drop the curtains in the evening, switch on the lights, close a gate, etc...). The implementation (Table 13) uses the [suncalc](#) module to perform solar computations. Note: the computations do not take into account the relief which might retard the sunrise of a few minutes. If it's the case, then a delay can be added using a « delay » port (Exhibit 26) or directly working on the code.

Context

1 //code exécuté au démarrage du circuit
2 console.log("hello"); //écris dans la console
3 var count = 0; //déclare et initialise le compteur à 0

inputs.a

1 //code exécuté à la réception d'une nouvelle valeur sur l'input a
2 count = count + 1; //incrémente de 1 le compteur à chaque nouvelle entrée.
3 console.log("compteur", count); //log la nouvelle valeur du compteur
4 c(count); // envoie la valeur du compteur sur la sortie c

1

a

compteur

c

5

compteur

13:39:54 - hello
13:39:57 a compteur 1
13:39:59 a compteur 2
13:40:01 a compteur 3
13:40:03 a compteur 4
13:40:05 a compteur 5

☒ autoscroll ☐ clean

Exhibit 24 Implementation of a counter. Open the code editor by double clicking on the port from the interface. On the « Context » tab you can find some code valued at the start of the circuit. The other tabs contain code called when a new value is received on the corresponding input (here « a »). Note : the function console.log() allows to keep track of the execution of a log window linked to the port.

Access to the « a » input value	Execute 1 on the « c » output
Gate.inputs.a.value	Gate.outputs.c.exec(1)
inputs.a.value	outputs.c.exec(1)
a	c(1)

Table 12 : Three alternative ways of accessing the value of an input and execute an output from the code of a programmable port.

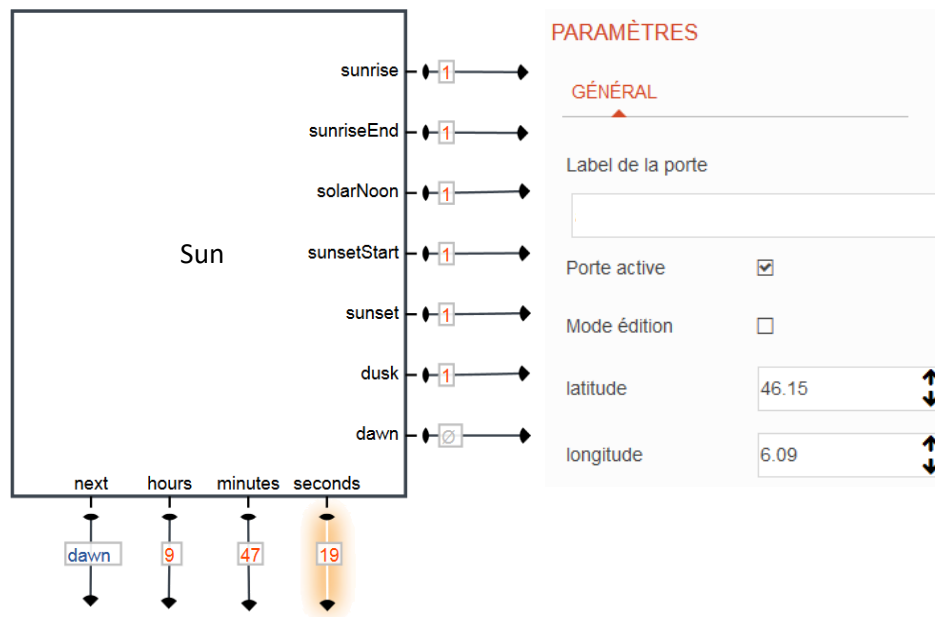


Exhibit 25 « Sun » port. Some executions (right-side outputs) are launched at the sunrise (sunrise), when the sun reaches the zenith (solarNoon), at the sunset (sunset), etc... The down-side outputs represent the next event and the time left to its realization (here the sunrise is expected in 9 hours, 47 minutes and 10 seconds). In order to perform the computations, two parameters have been added to this port: the latitude and the longitude (here 46°12'N, 06°09'E, corresponding to the position of Geneva).

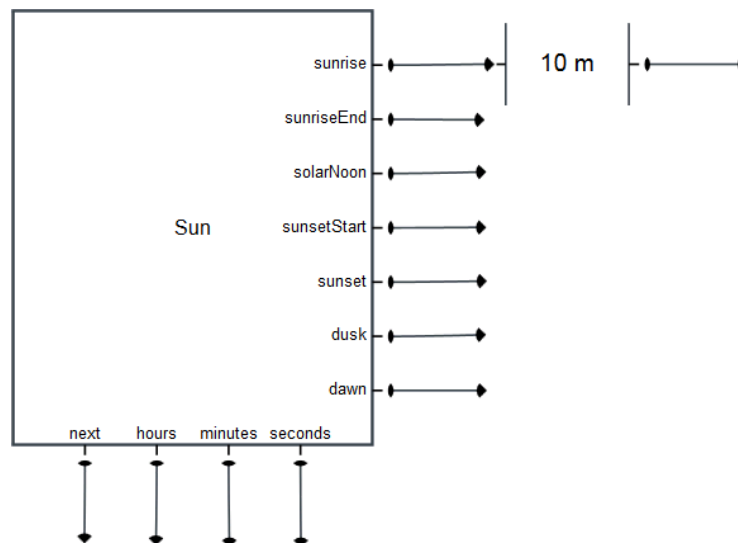


Exhibit 26 « Sun » port. Here we add a delay of 10 minutes to the sunrise to take consideration of the relief.


```

var suncalc = require('suncalc'); //https://github.com/mourner/suncalc
var nextTimer = null;

//schedules execution of next event (sunrise/sunset/...)
function scheduleNextEvent(startTs) {
    var now = Date.now();
    //compute events timestamps based on latitude and longitude parameters
    var res = suncalc.getTimes(startTs ? startTs : now, latitude, longitude);
    var keys = outputs.filter(function(o) {return o.position ==
'right'}).map(function(o) {return o.name});
    var tss = keys.map(function(k) {return res[k].getTime()});
    var tsMin = tss.map(function(ts) {return ts < now ? Number.MAX_VALUE :
ts}).reduce(function(a,b) {return Math.min(a,b)});

    if(tsMin !== Number.MAX_VALUE) { //timestamp of the next event found
        var indexMin = tss.indexOf(tsMin);
        var output = outputs[keys[indexMin]];

        nextTimer = setTimeout(function() { //executed when the event happens
            output.exec(1); //send an impulsion to the correct output.
            scheduleNextEvent();
        }, tsMin-now);

        next(keys[indexMin]); //indicates the next event(sunrise/sunset/...)
    }
    else { //all the events (sunrise/sunset/..) are past for today
        scheduleNextEvent(now+1000*60*60*24); //find tomorrow next event
    }
}

//countdown until next event (sunrise/sunset/...)
function countdown() {
    var tsl = Math.round(nextTimer.next()/1000); // total seconds left
    var s = tsl % 60; // seconds left
    var m = Math.floor(tsl/60) % 60; //minutes left
    var h = Math.floor(tsl/3600); // hours left
    seconds(s); //outputs the seconds left
    if(this.m !== m) { //outputs the minutes left (when it changes)
        this.m = m;
        minutes(m);
    }
    if(this.h !== h) { //outputs the hours left (when it changes)
        this.h = h;
        hours(h);
    }
}

scheduleNextEvent(); //start at initialization

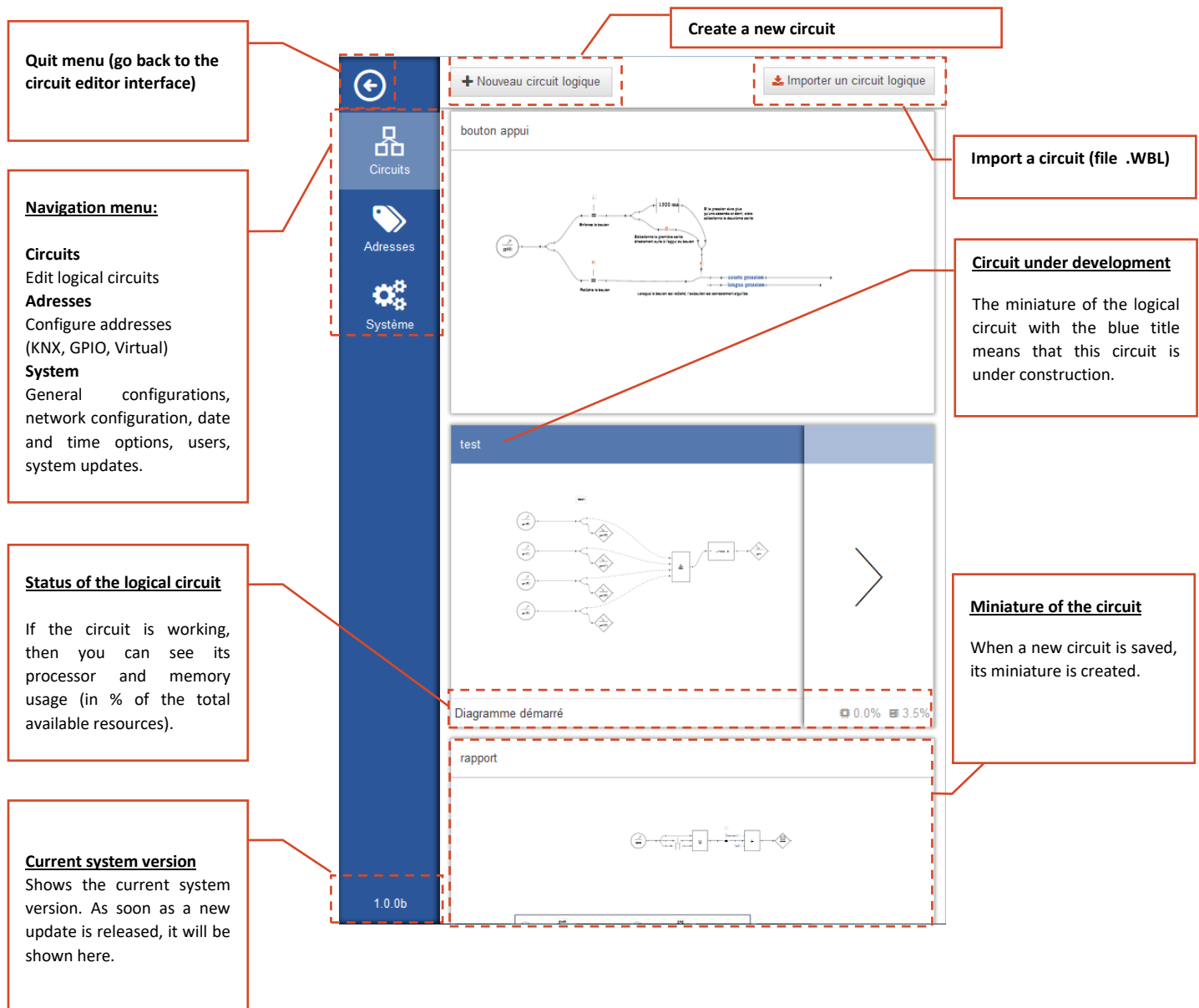
countdown(); //update countdown at initialization
setInterval(countdown,1000); // update countdown every seconds

```

Table 13 Set up of the « Sun » port (computation of the sunrise and sunset times) by using the nodejs suncalc module. This code is executed once at the boot of the circuit.

3 INTERFACE

3.1 SELECTION PANEL



Quit menu (go back to the circuit editor interface)

Create a new circuit

Import a circuit (file .WBL)

Navigation menu:

- Circuits**
Edit logical circuits
- Adresses**
Configure addresses (KNX, GPIO, Virtual)
- Système**
General configurations, network configuration, date and time options, users, system updates.

Circuit under development

The miniature of the logical circuit with the blue title means that this circuit is under construction.

Status of the logical circuit

If the circuit is working, then you can see its processor and memory usage (in % of the total available resources).

Miniature of the circuit

When a new circuit is saved, its miniature is created.

Current system version

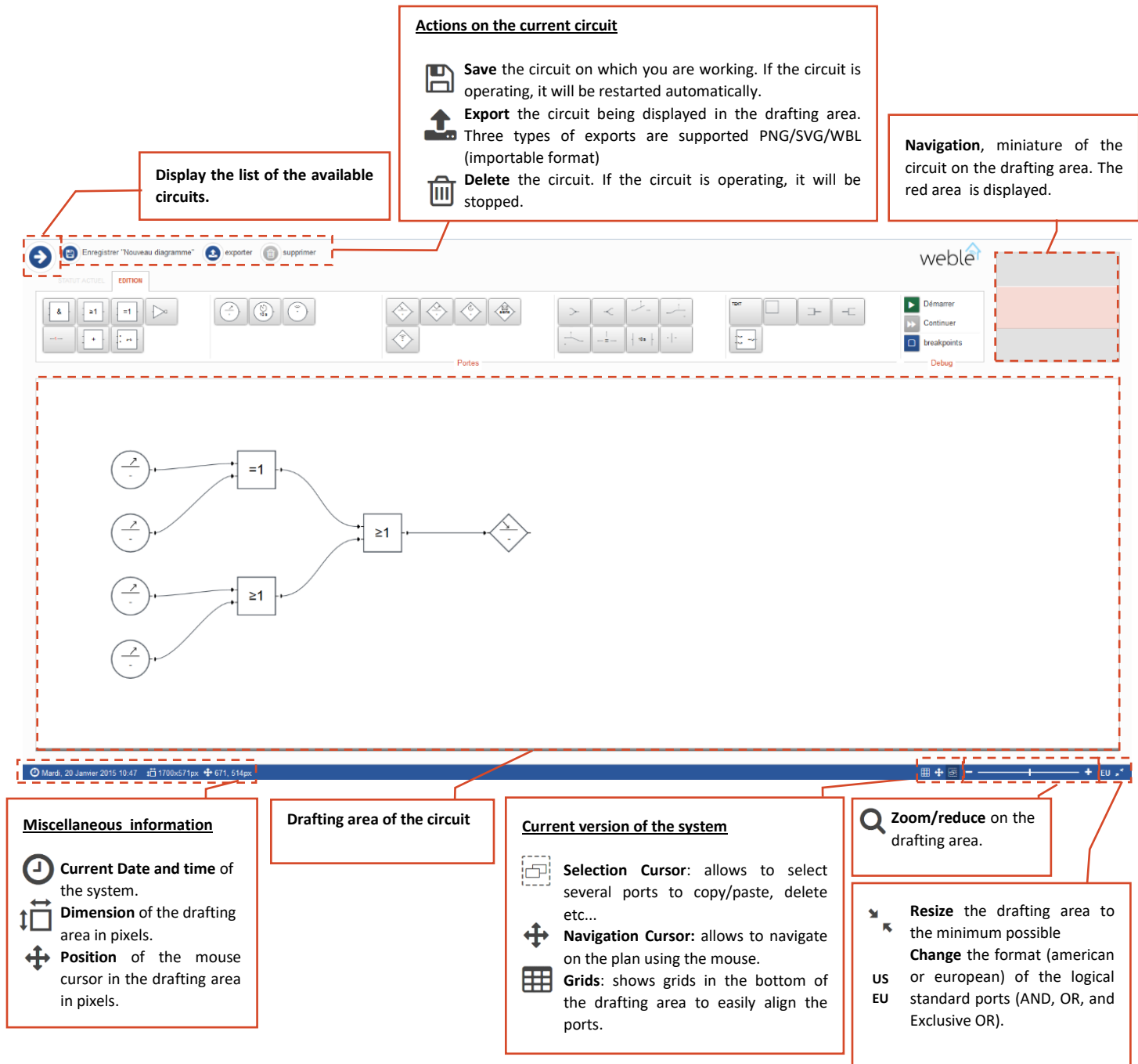
Shows the current system version. As soon as a new update is released, it will be shown here.

Diagramme démarré 0.0% 3.5%

rapport

1.0.0b

3.2 LAYOUT



Display the list of the available circuits.

Actions on the current circuit

- Save** the circuit on which you are working. If the circuit is operating, it will be restarted automatically.
- Export** the circuit being displayed in the drafting area. Three types of exports are supported PNG/SVG/WBL (importable format)
- Delete** the circuit. If the circuit is operating, it will be stopped.

Navigation, miniature of the circuit on the drafting area. The red area is displayed.

Miscellaneous information

- Current Date and time** of the system.
- Dimension** of the drafting area in pixels.
- Position** of the mouse cursor in the drafting area in pixels.

Drafting area of the circuit

Current version of the system

- Selection Cursor**: allows to select several ports to copy/paste, delete etc...
- Navigation Cursor**: allows to navigate on the plan using the mouse.
- Grids**: shows grids in the bottom of the drafting area to easily align the ports.

Zoom/reduce on the drafting area.

Resize the drafting area to the minimum possible

Change the format (american or european) of the logical standard ports (AND, OR, and Exclusive OR).

US
EU

3.3 EDITION

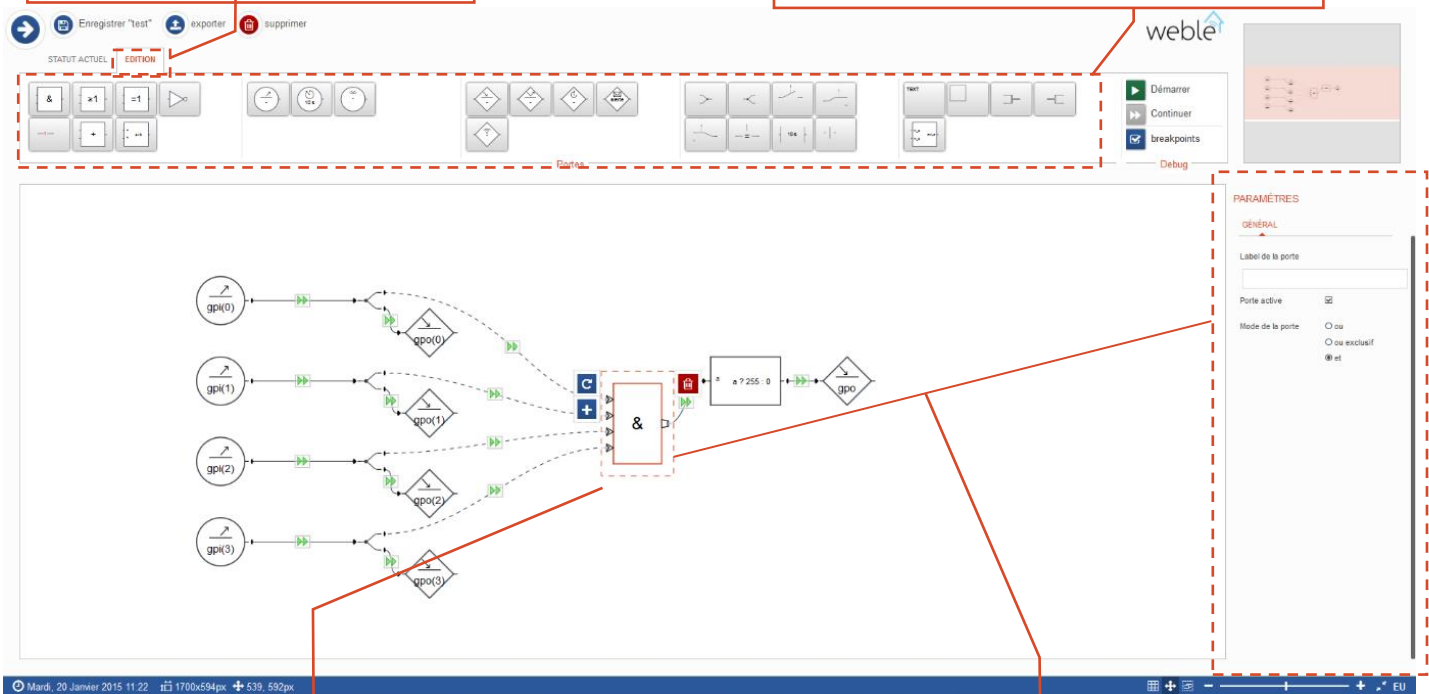
Edit

From the Edit view you can change the circuit, add ports, configurations etc...

This view includes a debugger you can use to test the circuit.

Ports List

The available ports are presented by category: logical ports, triggers, actuators, flow controls and other. To add a port, click between them and place the new port in the drafting area, then reclick on the desired position to place it.



Ports Selection

Click on a port to select it. Then the port becomes red and is centered. Some selection actions are common to all ports, others are specific to the selected port.

Rotate the port. You can rotate all kinds of port. They can be rotated (each time of 90°) to make the circuit more understandable.

Add inputs/outputs. The majority of the ports support the add/delete of inputs or outputs. By convention, the outputs are positioned at the right of a port and the inputs at its left. If this button is positioned at the right of a port, then an output will be added when you click on it.

Delete the port. All ports can be deleted with this button. Once selected, you can also delete the port using the command « del/Delete » of your keyboard.

Ports Configuration

Once you selected a port, its parameters will be displayed on the right menu. All ports present two options:

The Label displays a text on top of the port. This text has only an informative purpose.

Active/Disable the port to make inoperative in the circuit.

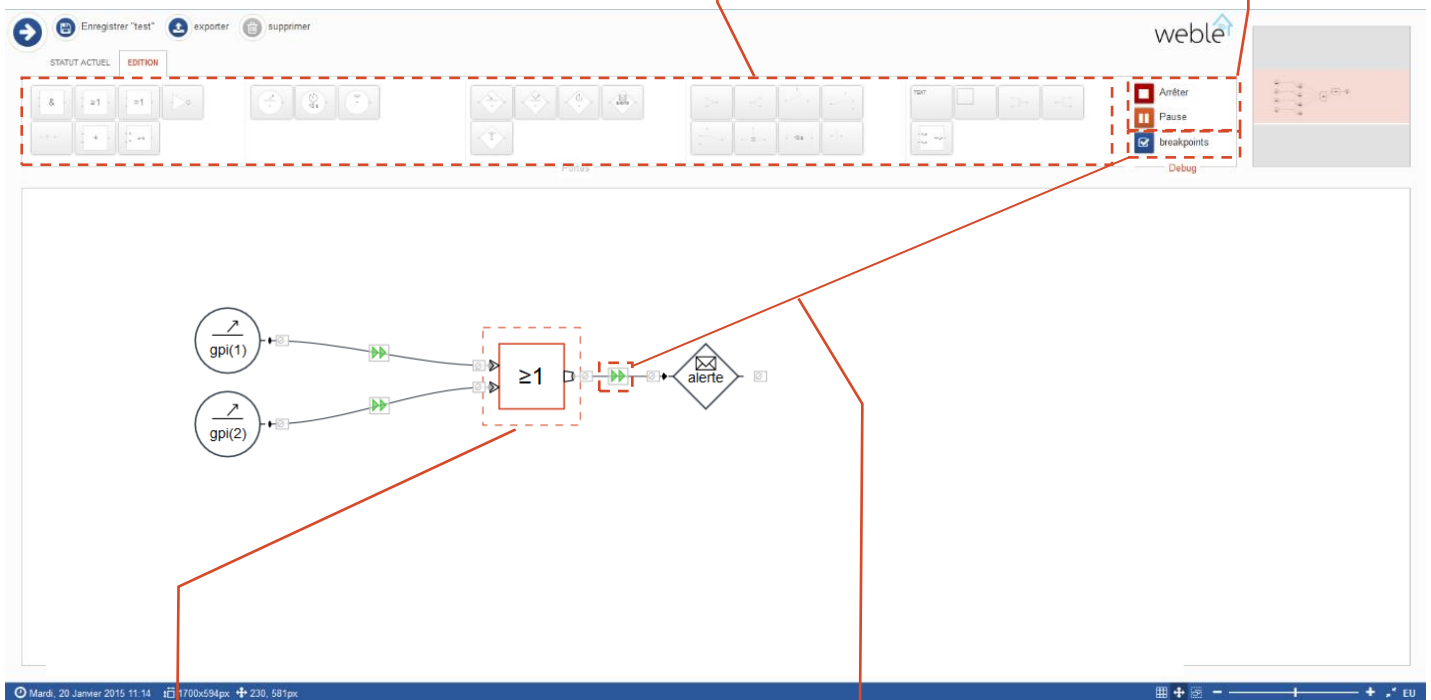
3.4 DEBUG

Blocked add of new ports

Once you started the debugger, you cannot add new ports.. Clicking on the menu of the port categories will automatically stop the debugger.

Debug

- ▶ **Start/stop** the debugger. Allows to test the diagram before putting it into production. Using the debugger you can pause the circuit, so helping to view and locate the errors more easily.
- ▶▶ **Play/pause** : once the debugger started, you can pause the circuit at any time by clicking on this button.



Relocatable but not configurable ports

When the debugger is active, you can move the ports in the drafting area but not edit their configurations.

Breakpoints

- ☒ **Display/hide breakpoints** : displays or hides the « breakpoints » over the links, thus pausing the circuit during the execution of the link concerned.
- ☐

3.5 STATUS

Status

The « status » view allows to manage and display the status of the circuit.

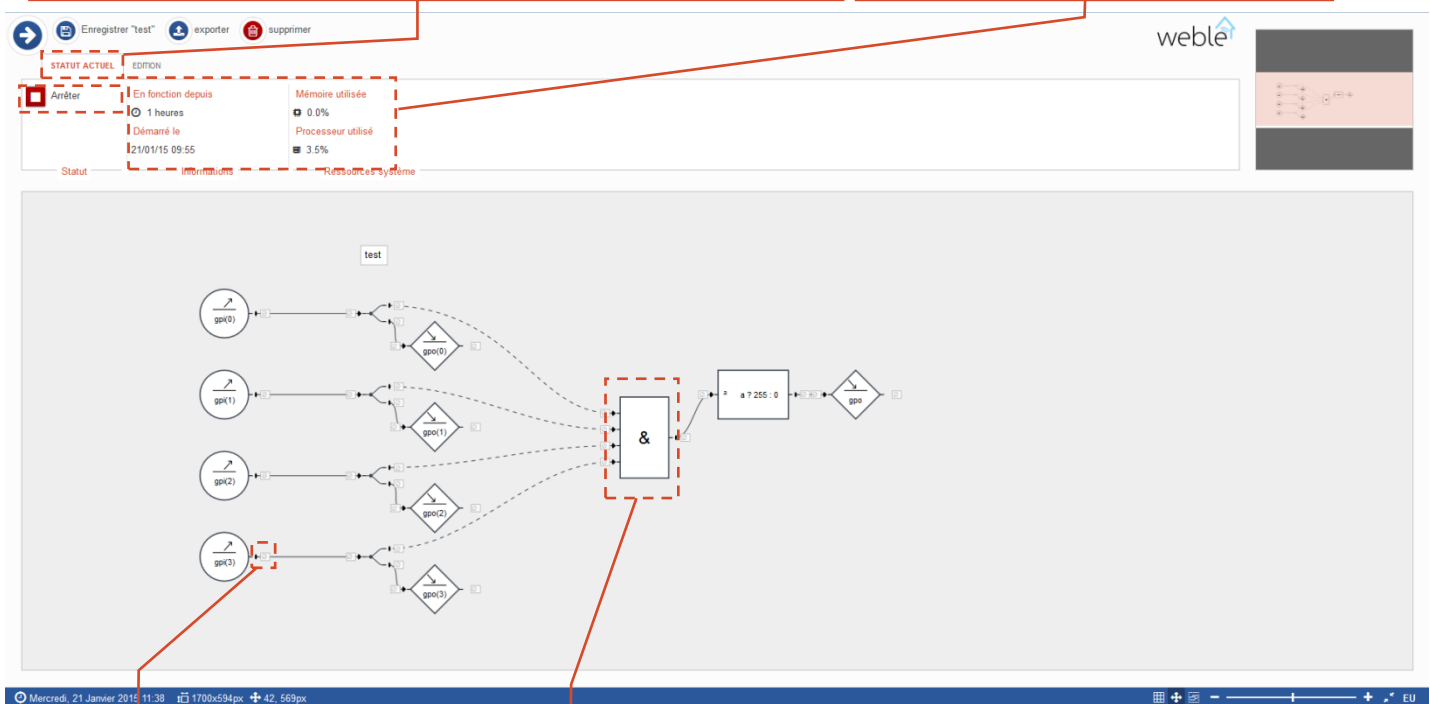
From the menu you can start (release into production) the circuit and stop it if needed.

Once started, the execution status of the circuit is showed in real-time in the diagram. On this way, you can check at any time that the circuit is working properly.

Note : if some modifications/corrections are saved while the circuit is working, it will automatically restarted to update the circuit with the latest modifications.

Circuit working information

When the circuit is working, the information regarding its operation times and the machine resources consumed are displayed on this area.



Value change

While the circuit is working, it's possible to enter a value and force its execution on a selected input or output. On this way, you can manually execute just certain parts of the circuit.

Blocked port configuration

From this view you cannot move ports or edit their configurations. Only the navigation through the drafting area is allowed.