Appendix 11
Z-Wave

This appendix describes support for Z-Wave interfaces and devices. Included are these sections:

- What is Z-Wave?
- Building a Z-Wave network
- Configuring the Leviton Z-Wave Interface
- Z-Wave Devices
- Z-Wave Visual Program Element
- Z-Wave Triggers

Before you begin with Z-Wave!

Z-Wave is a bit like Linux – lots of capability with little consistency between providers or documentation. Z-Wave can do a lot if you can just find the right piece of the puzzle and know where to put it.

Z-Wave is very different from other HCA supported protocols in that HCA provides only a method to send and receive commands from the interface. It does not attempt to encode or decode those commands. Because of this, HCA technical support can offer little beyond what is in this User Guide Appendix and the Z-Wave technical notes available on the support web site. Beyond this you are on your own.

What is Z-Wave?

Z-Wave is a proprietary wireless mesh networking technology originally developed by the Danish company Zensys, now part of Sigma Designs. Because Sigma Designs considers documentation about Z-Wave technology proprietary, and makes such documentation only available under license, it can be a bit challenging to figure out how it works and how to control Z-Wave devices.

Unlike HCA support for UPB, Insteon, X10 and Wireless devices, there is very little built-in to HCA for Z-Wave. There is support for very simple one-way *on-off-dim* and *on-off* devices but that's all. All other device types you may want to add to your design – thermostats, two-way devices, door locks, etc – all have to have their protocol determined and then that protocol implemented in HCA programs you create. This is much different than other supported protocols!

Fortunately, there is a good deal of Z-Wave information on the Internet. A recommendation is to search for protocol and command information on your devices. On the HCA web site is a technical note that discusses using the HCA Z-Wave support for Z-Wave door locks.

Building a Z-Wave network

While there are several Z-Wave interfaces available, HCA supports only the *Leviton Vizia RF+ serial to Z-Wave* interface. This interface is model number VRC0P.

Just as a UPB network is initially configured using a tool external to HCA (UPStart), a Z-Wave network is configured with a Windows program. This software, available for download from Leviton is called the *Vizia RF+ Installer Tool*. To use this program you need also a LEVVRUSB-1US USB stick for configuration.

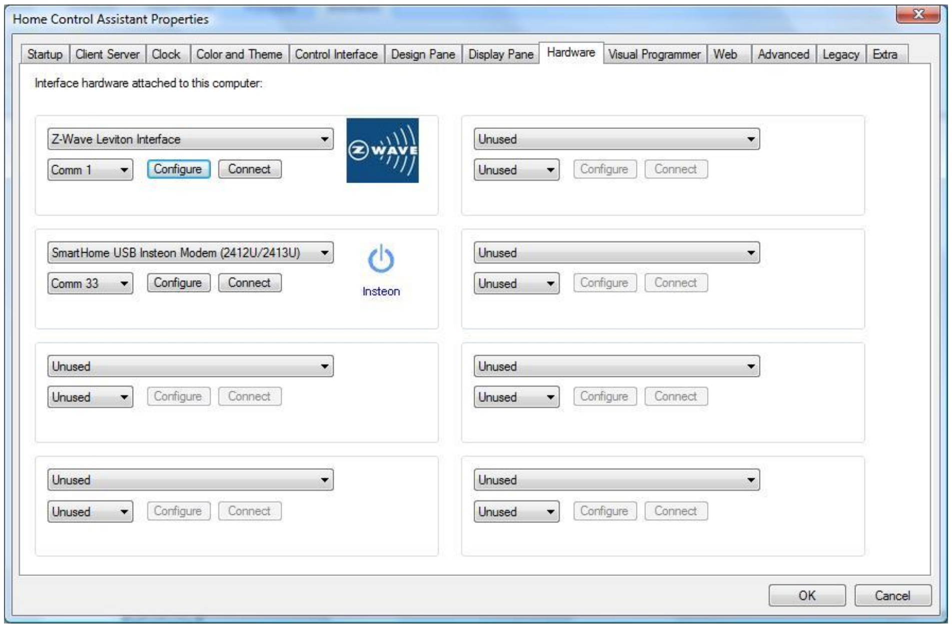
This USB stick will be the "primary" controller in your network. The Leviton VRC0P will function as a "secondary" controller, in Z-Wave parlance, even though it will be the way HCA controls your Z-Wave devices.

Instructions for creating a Z-Wave network using the Leviton USB stick and installation software can be found in the Help file included with the *Vizia RF+ Installer Tool* and on the Leviton support web site.

Tip: When you do the final network configuration with the Leviton tool, all Z-Wave devices need to be in their final location. Moving Z-Wave devices affects the network topology, and can adversely affect network routing if the network is not re-optimized.

Configuring the Leviton Z-Wave interface

The Leviton Z-Wave interface is identified to HCA like all interfaces from the hardware setup tab of *HCA Options*. The only piece of information needed is the serial port it uses.



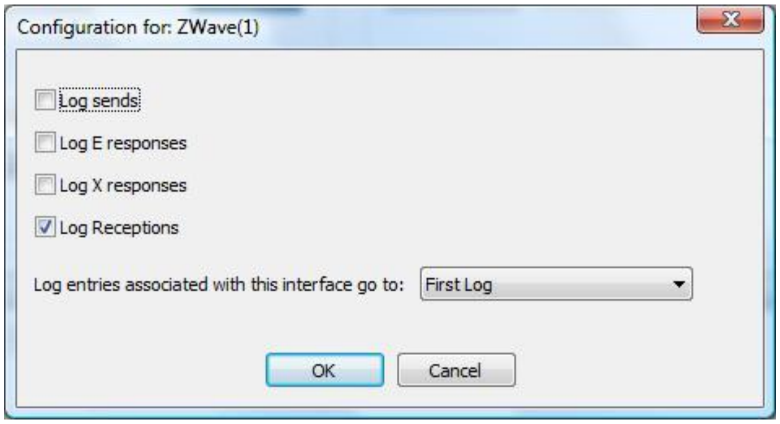
Interface hardware attached to this computer:

Z-Wave Leviton Interface — Comm 1

SmartHome USB Insteon Modem (2412U/2413U) — Comm 33

Unused — Unused

The *configure* button sets the logging options:


Configuration for: ZWave(1)

- [ ] Log sends
- [ ] Log E responses
- [ ] Log X responses
- [x] Log Receptions

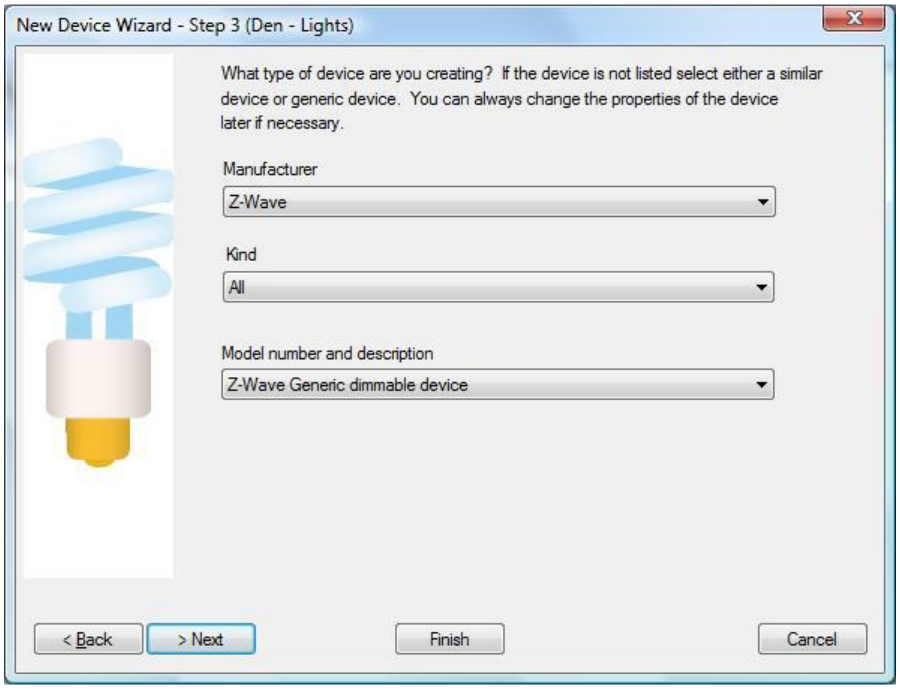
Log entries associated with this interface go to: First Log

As described in the Leviton application note, each message sent to the interface usually generates an “E” response with an error code, followed by a “X” response that shows the transmission status. You can choose to log these or not.

Tip: The Leviton application note that describes the Leviton Z-Wave protocol is available on the HCA support web site in the technical notes section.

Z-Wave devices

HCA directly supports only simple one-way *on-off-dim* and *on-off* devices. To use these devices select as their type *Z-Wave Generic dimmable device* or *Z-Wave generic non-dimmable device*.


New Device Wizard - Step 3 (Den - Lights)

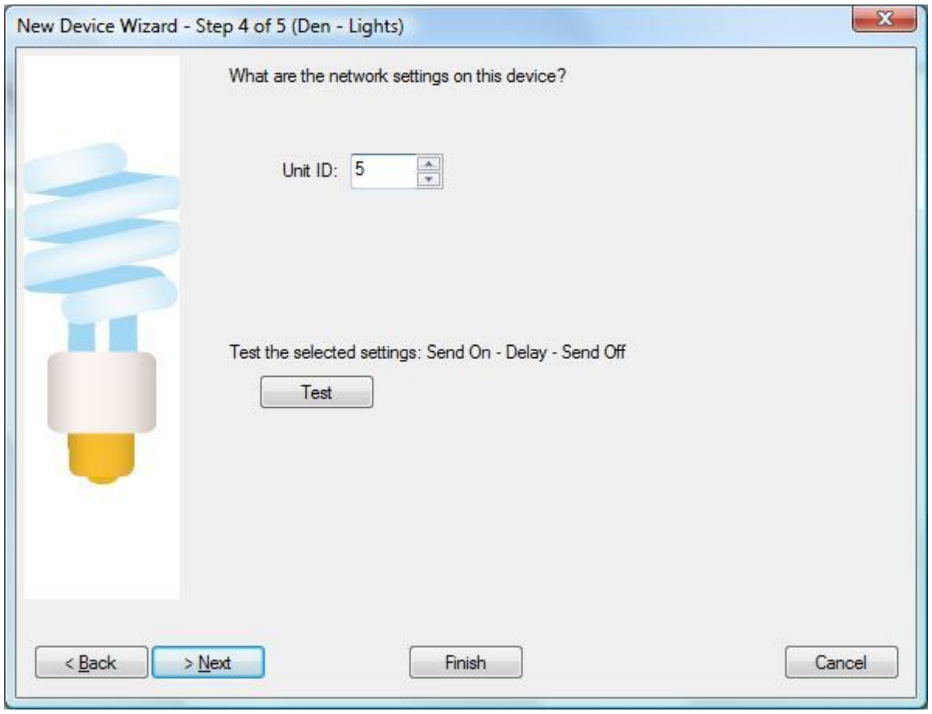
What type of device are you creating? If the device is not listed select either a similar device or generic device. You can always change the properties of the device later if necessary.

Manufacturer: Z-Wave

Kind: All

Model number and description: Z-Wave Generic dimmable device

In the next step of the wizard the unit id of the Z-Wave device is set:



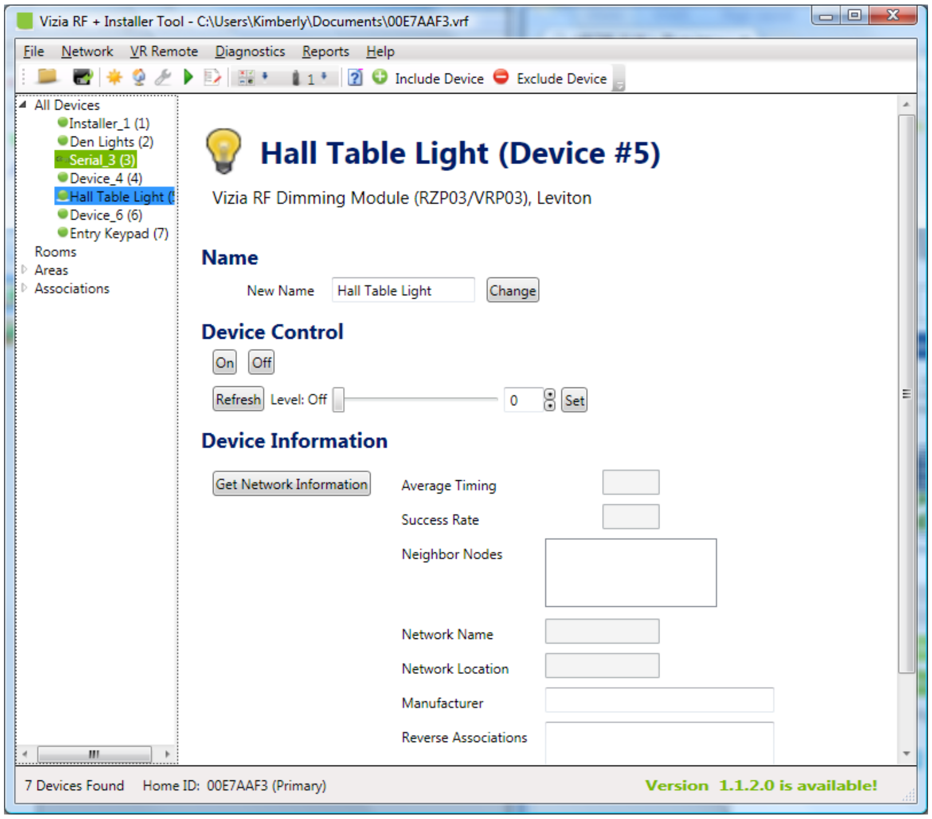
New Device Wizard - Step 4 of 5 (Den - Lights)

What are the network settings on this device?

Unit ID: 5

Test the selected settings: Send On - Delay - Send Off

The unit id you enter into HCA comes from the Leviton setup tool:



Vizia RF + Installer Tool - C:\Users\Kimberly\Documents\00E7AAF3.vrf

All Devices
- Installer_1 (1)
- Den Lights (2)
- Serial_3 (3)
- Device_4 (4)
- Hall Table Light (
- Device_6 (6)
- Entry Keypad (7)

Rooms
Areas
Associations

Hall Table Light (Device #5)

Vizia RF Dimming Module (RZP03/VRP03), Leviton

Name

New Name: Hall Table Light

Device Control

Level: Off 0

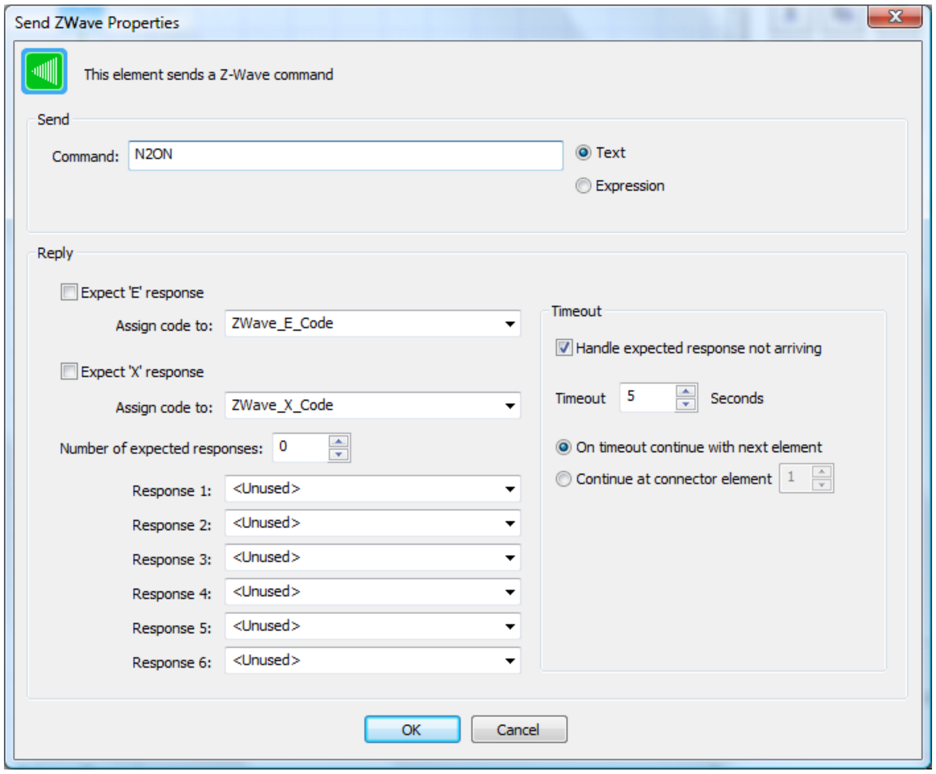
Device Information

Average Timing
Success Rate
Neighbor Nodes
Network Name
Network Location
Manufacturer
Reverse Associations

7 Devices Found   Home ID: 00E7AAF3 (Primary)   Version 1.1.2.0 is available!

Z-Wave Visual Program Element

The real power of Z-Wave is not in the simple devices described in the above section but rather in the devices that you can implement. The key piece is the Z-Wave Visual Programmer element.


Send ZWave Properties

This element sends a Z-Wave command

Send

Command: N2ON [✓ Text] Expression

Reply

☐ Expect 'E' response

Assign code to: ZWave_E_Code

☐ Expect 'X' response

Assign code to: ZWave_X_Code

Number of expected responses: 0

Response 1:	<Unused>
Response 2:	<Unused>
Response 3:	<Unused>
Response 4:	<Unused>
Response 5:	<Unused>
Response 6:	<Unused>

Timeout

☑ Handle expected response not arriving

Timeout 5 Seconds

[✓ On timeout continue with next element] Continue at connector element 1

Enter the text to send into the *Send Text* box. This can be the actual text or the result of an expression evaluation if the *Expression* option is used.

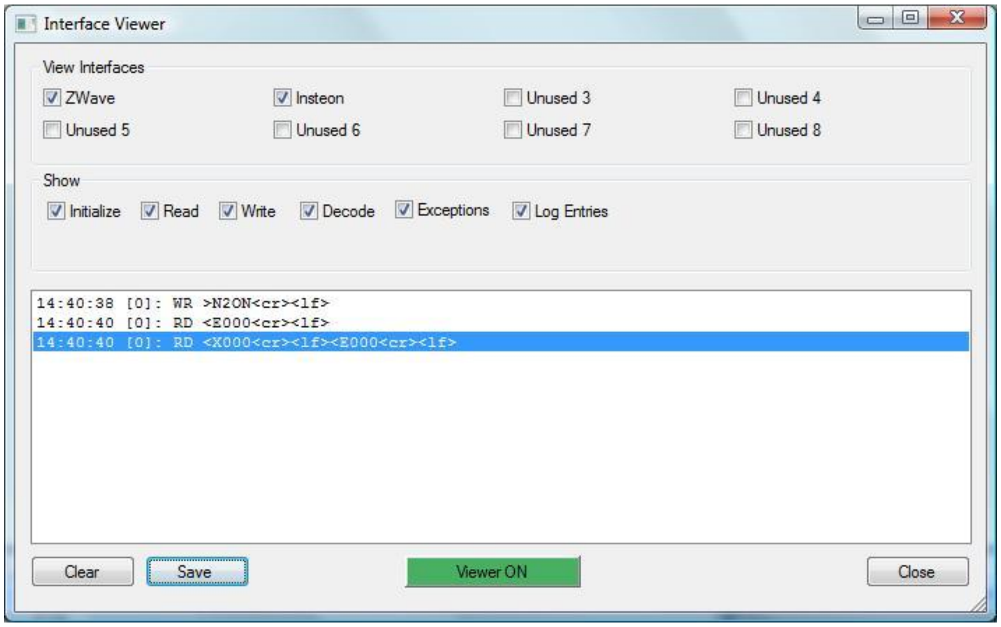
Depending upon the command you can expect “E” and “X” responses and if so tick those boxes and those messages are specifically decoded and the values assigned to the selected flags. If the command generates additional responses, enter the number of expected responses and then, for each response, select the flag that reception is assigned to. You can select an existing flag or type in the name of a new flag that is created.

Interface Viewer

What does a very simple Z-Wave command look like? You can use the *Interface Viewer*.

Open the *Interface Viewer* from the *Interfaces* ribbon category. This viewer shows the messages sent to and received from one or more interfaces. This gives you a window into the communications at the lowest level.

In the image below, an ON command was sent to unit number 2. The interface replied back with an “E” response with code 0 (success) and an “X” response with code 0 (success).



[✓] ZWave [✓] Insteon [ ] Unused 3 [ ] Unused 4
[ ] Unused 5 [ ] Unused 6 [ ] Unused 7 [ ] Unused 8

[✓] Initialize [✓] Read [✓] Write [✓] Decode [✓] Exceptions [✓] Log Entries

```
14:40:38 [0]: WR >N2ON<cr><lf>
14:40:40 [0]: RD <E000<cr><lf>
14:40:40 [0]: RD <X000<cr><lf><E000<cr><lf>
```

The names of the eight interfaces that can be configured are the first set of checkboxes. Tick the ones that you want to see data from. Interfaces not configured are listed as *Unused*.

The *Show* box determines what sort of information you want to view. Tick or clear the boxes for the type of data you want to see.

You can turn the viewer on or off with the button at the bottom of the dialog. When green the viewer is enabled and when red it isn't.

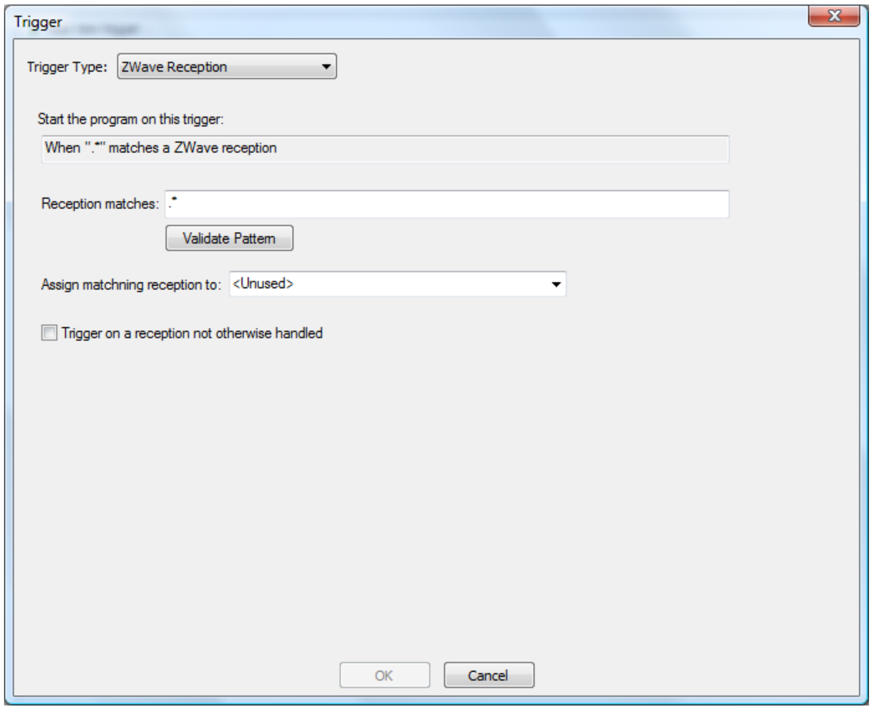
In the dialog list, messages received from the interface are prefixed with RD, messages sent to the interface are prefixed with WR, and if the message is decoded – messages from some interfaces are decoded but Z-Wave messages aren't – they are prefixed with DC. Enclosed in []'s is the number of the interface the line is for. The first interface is numbered 0 and the last is numbered 7.

This interface is at a very low level so you may see, because of the nature of network and serial reads, a single reception broken across multiple RD lines.

You can clear the list with the *Clear* button and save the list into a text file with the *Save* button.

Triggers

Messages received from the Z-Wave interface can be used as a trigger for a program. To create a trigger of this type, select as the trigger type *Zwave Reception.*



Trigger Type: ZWave Reception

Start the program on this trigger:
When ".*" matches a ZWave reception

Reception matches: .*

Assign matchning reception to: <Unused>

☐ Trigger on a reception not otherwise handled

There are several parts to a *ZWave Reception* trigger and these are covered in the next sections.

Pattern

Unlike, for example, Insteon triggers where the reception from an Insteon interface is decoded into a source device and command, and Insteon triggers specify that source and command, the ZWave Reception trigger is based only on the text of the reception itself.

A ZWave Reception trigger contains a pattern that is matched against the received text. If the pattern matches the received text then the program is triggered. If the reception doesn’t match the pattern then the program is not triggered.

The pattern is specified as a *Regular Expression*. A regular expression is a sequence of characters that forms a search pattern. The full description of a regular expression is beyond the scope of this User Guide but here are the fundamentals.

A regular expression is composed of a series of characters and metacharacters. Characters that are not metacharacters match the corresponding changes in the text. Meta characters describe one or more possible characters to match in the text. The metacharacters are:

Character	Use
. (dot)	Matches any single character. For example, a.c matches "abc", or "axc" or "a6c" etc.
[ ]	A bracket expression. Matches a single character that is contained within the brackets. For example, [abc] matches "a", "b", or "c". [a-z] specifies a range which matches any

	lowercase letter from "a" to "z". These forms can be mixed: [abcx-z] matches "a", "b", "c", "x", "y", or "z", as does [a-cx-z].
[^]	Matches a single character that is not contained within the brackets. For example, [^abc] matches any character other than "a", "b", or "c". [^a-z] matches any single character that is not a lowercase letter from "a" to "z".
^	Matches the starting position within the string
$	Matches the ending position of the string
*	Matches the preceding element zero or more times. For example, ab*c matches "ac", "abc", "abbbc", etc. [xyz]* matches "", "x", "y", "z", "zx", "zyx", "xyzzy", and so on.
{m, n}	Matches the preceding element at least m and not more than n times. For example, a{3,5} matches only "aaa", "aaaa", and "aaaaa". This is not found in a few older instances of regular expressions
\	The following character is a character and not a metacharacter

Here are some examples:

- `.*` matches any text.
- `.at` matches any three-character string ending with "at", including "hat", "cat", and "bat".
- `[hc]at` matches "hat" and "cat".
- `[^b]at` matches all strings matched by .at except "bat".
- `[^hc]at` matches all strings matched by .at other than "hat" and "cat".
- `^[hc]at` matches "hat" and "cat", but only at the beginning of the string or line.
- `[hc]at$` matches "hat" and "cat", but only at the end of the string or line.
- `\[.\]` matches any single character surrounded by "[" and "]" since the brackets are escaped, for example: "[a]" and "[b]".

Since it can be complex to compose a regular expression, in the dialog is a *Validate* button that checks the syntax of the regular expression. You must use the *Validate* button to check the pattern before creating the trigger.

However, just because the syntax is correct that doesn’t mean that the pattern matches strings as you expect. Only testing shows that.

Tip: Use your favorite search engine or Wikipedia to find the full syntax of regular expressions.

Flag assignment

ZWave Reception triggers are also different than other triggers in an additional aspect. The text of the reception can be assigned to a flag for subsequent decoding in the triggered program.

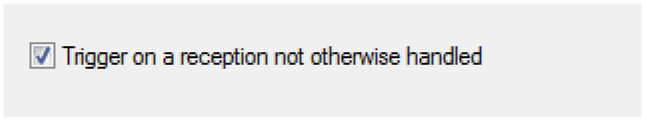
Select the name of an existing flag or type in the name of a new flag. If the program is triggered – when the pattern matches the reception – the received text is assigned to the flag before the program starts. This lets you do further decoding of the reception in the program that is started.

This is an optional feature of the trigger. If you don’t want to use this, set the flag as *<Unused>*

Trigger of last resort

You can also create a trigger that starts a program if no trigger on any program matches the Z-Wave reception. Of course, only one program can be designated with this trigger.

To enable this option tick the *Trigger on a reception not otherwise handled* checkbox.



[✓] Trigger on a reception not otherwise handled