ZUMUNGO

WiFi Relays

Software for ESP32 and ESP8266

Smart – Programmable – Controllable

Control by HTTP, TCP and MQTT

Reports by HTTP, TCP and MQTT

USER GUIDE

rev. 3.1.2 October, 2019

WARNING !!!

This is not a consumer software !

Basic skills and understanding of electrical circuits are required to properly install the software and operate the device!

Improper handling can result in property damage, bodily injury or death !

Table of Contents

1. Specifications

WiFi /controller module	ESP8266 or ESP32
AdHoc WiFi network SSID	Zumungo
Factory set password for the AdHoc network	Zumungo123 (do not forget to change it to your own password for protection. Go to “WiFi settings” at the bottom of the main web page.)
IP address within the AdHoc WiFi network	192.168.4.1
Modes of operation	5 modes – normal, I/O driven, schedule1, schedule2, schedule3
Sub modes for IO	5 sub modes Normal IO – Relay status follows the status of IO OVR – overlay, IO control runs on top (Overlays) of Normal mode MONO N – triggered by IO transition for certain time MONO R – recurring MONO mode MONO K – First pulse triggers and Second pulse terminates the action
Schedules	3
Units within single schedule	6
Timing for schedules	Absolute time or relative to SS (sunset) or SR (sunrise)
Source of time	Online NTP servers (high accuracy) or on board (manual setting, less accurate) if not connected to the Internet.
Time zones	6 – Eastern, Central, Mountain, Pacific, Alaska, Hawaii
MQTT control	All major functions and reports can be managed by MQTT Factory settings for MQTT test.mosquitto.org port 1883 Home/Zumungo/MultiRelay/Input Home/Zumungo/MultiRelay/Output
TCP control	All major functions and reports can be managed by TCP Factory settings for TCP client IP address – blank setup your own at the web page go to “Board Settings” server IP address – blank setup your own at the web page go to “Board Settings” server port 7000 client port 8000
HTTP control	Most major functions and reports can be managed by HTTP calls

	Due to the nature of HTTP, it can't process asynchronous messages/reports such as Input change message for example. Default port for HTPP is 80 (can be changed).

2. How to install the software in your device

2.1 Copy the free "Zumungo Installer". Add your own SSID and WiFi password in the code as instructed. Flash it and run it on your board.
2.2 Find your board's 6 characters ID. Go to "settings"-->"WiFi" of your smart phone and look for WiFi network Zumungo XXXXXX, where XXXXXX is your 6 characters ID
2.3 Go to zumungo.com and purchase the licenses that you need. Free trial is available. No credit card required for a free trial.
2.4 Congratulations !!! Your board is operational now.
2.5 Find your board's local IP address by one of the following methods

a) looking in your router reports
b) sending "CallAll" message to UDP port 60560. Many free TCP/UDP clients are available for that. The board will report it's name, software version and IP address.
c) connect to the AdHoc Access Point your device creates (see p. 3 below)

3. Connect to AdHoc WiFi Network

ZUMUNGO creates its own WiFi network with SSID

ZUMUNGO XXXXXX where XXXXXX is unique board ID.

Use any WiFi capable device such as laptop, tablet, smart phone etc and find the ZUMUNGO network there and connect using the default password

Zumungo123

Once connected to ZUMUNGO network go to this default IP address (not subject to change)

192.168.4.1

Use the default password for the relay board

WifiRelay123456

to open the web interface and setup the board.

Please do not forget to change the default password to your own password for your own protection. Go to “WiFi settings” at the bottom of the main web page to do that.

You can use the board as it is at this moment, however you have to always have a device (computer, laptop, smart phone) connected to ZUMUNGO network. Also any MQTT, TCP or HTTP commands have to come from devices connected to the same ZUMUNGO network. This could be inconvenient. See next chapter for how to connect ZUMUNGO powered device to your home WiFi.

4. Connect to home WiFi network.

Place ZUMUNGO powered device in an area where you have good coverage of your home WiFi network. This applies also to the final permanent destination of your ZUMUNGO powered device. If you are to control irrigation for example from your garage, first make sure you have decent WiFi coverage there.

IMPORTANT !

Never place your ZUMUNGO powered device in a metal enclosure if you want to control it over WiFi.

While connected to the adhoc network ZUMUNGO (see previous chapter) go to the main web page and take a note of the 6 character Board ID that looks similar to this with different digits and letters.

Board ID: 6AB41A

After you have taken the Board ID on a piece of paper than go to the bottom of the page and click the button “WiFi settings” at the bottom.

Select the name of your home WiFi network from the drop-down box, enter the password for your home WiFi network and click “connect”.

After few seconds your ZUMUNGO powered device will connect to your home WiFi network.

How do you find it is another question. To do so you need to go to your router's DHCP tables and look for the MAC address you just took note of. You will find the IP address next to it that looks like this

192.168.1.150 for example your digits will be different

Go to any browser on your home WiFi network and enter the IP address of the board and you are in business. While at the router you may consider turning the dynamic IP address assigned by the router into a static one so that you always find your board at the same IP address.

Alternatively you can issue "CallAll" command by UDP to port 60560. Multiple free software TCP/UDP clients are available for desktops, tablets and smart phones. One such software is PacketSender for desktops.

5. Setup name and password for the board

Default name is Zumungo
Default password is: WifiRelay123456

Go to home web page and log-in with the default password.
Scroll down to the bottom and click on “Board Setup” button.

At the top of the “Board Setup” page choose new name for your board and click “save”
Also at the top enter the existing password (default Wifirelay123456 but enter your previously selected password if there is one).
Pick a new password and enter it twice. Also pick a password hint to remind you of your new password.
Click “Save password” and wait 10-20 sec as the board reboots after this change.

Useful hints.

1. Board name and password will be a part of each command you send to the relay board.
2. Pick short ones in order to avoid a lot of writing thereafter.

6. Setting up HTML and TCP ports

Default port is 80 for the HTML commands to the board and for the web pages.
Default port is 8000 for the TCP server residing on the relay board where you will be sending commands to if using TCP.
Default port is 7000 for the external TCP server (if any) that your relay board will report to.
The IP address of the external TCP server (if any) is provisional. You must update it if you intent to use external TCP server to receive all reports back from the relay board.
Setting up an external TCP server may be an overkill as under normal circumstances the responses from the relay board are immediate after the commands and can be received over the established TCP connection for issuing a command. However this connection will not be alive if the relay board issues an asynchronous report not related to a command such as “Watchdog activated” for example. In this regard you will need a separate server to listen for such asynchronous responses from the relay board.

Please enter the appropriate port numbers and IP address and click “Save” button. Please note that there is separate “save” button for the TCP server setup.

Useful hints.

1. You will lose your web pages if you switch the HTTP port to anything different than 80.
To get them back you need to use web address with suffix :xx where xx is your new port number. If you have chosen port 82 for example and your relay is at this address for example 192.168.1.150 than your main web page will be at 192.168.1.150:82 (just an example. Use your own settings)

2. Most likely port 80 is already taken by something on your home WiFi network. Use another port say 82 for your relay board. Than go to your router and forward port 82 to the IP address of the relay board. Now you can access your relay board from anywhere in the world by myhomenetwork.com:82 where “myhomenetwork.com” is the global IP address of your home network (see your internet provider or Google “My IP address” from your home network).

7. Setting up MQTT connection

Default broker is set at test.mosquitto.org
Default port is 1883
Default topic for command input to the relay board is Home/Zumungo/MultiRelay/Input
Default topic for responses from the relay board is Home/Zumungo/MultiRelay/Output

All of these are subject to change, however you need to understand the MQTT protocol and to mate the settings with the settings on the site that will control the relay board and also to make sure that there is a working MQTT broker at the IP address you provide to replace test.mosquitto.org

Useful hints.

1. At the time of this writing test.mosquitto.org is well maintained and working reliably all the time. It is open to the public and if you are going to use it than it may be smart to change the topics from the default topics so that you do not mess with other relay owners.

2. Be careful using public MQTT brokers as the relay commands are not encrypted and your board name and password may get exposed, allowing others to control your relays.

3. Running your own Mosquitto MQTT broker at your home network is best and easy to do. Mosquitto is free software and requires practically no maintenance.

8. Setting up location, date and time

Your relay board allows running on schedules among many other things.
Schedules may include absolute time as well as timing related to SS (sunset) and SR (sunrise) time.
That's why it is important for the board to be set properly with location, time and date.

It all happens automatically if you select one of the 5 time zones from the drop down box at the bottom of the “Board Setup” page.

Your relay board must be connected to your home WiFi network that is connected to the Internet.

If there is no Internet connection than your only option is to setup date and time manually.

9. GPIO configuration

Your relay board can be field configured in multiple different configurations to serve you in a best possible way.

Zumungo supports up to 9 GPIOs that can be configured as shown in the table below

On the other side you may want your relays to be hooked to and controlled by external sensors. Real life examples are rain sensor for irrigation applications, temperature sensor for HVAC applications, high temp sensor for sauna heater, motion detector for security applications, RFID or NFC sensor for access control etc.

In this latter case you will configure your relay board to use external sensors. Up to 4 inputs can be configured for this purpose.

The table below shows various possible configurations

Native real relays on board	4	4	4	4	4
Virtual relays (on board or external "dumb" relay board)	4	3	2	1	0
Sensor inputs	0	1	2	3	4
1wire input (optional)	1	1	1	1	1
Total relays	8	7	6	5	4

In order to configure your GPIOs (General Purpose Input Output) go to the bottom of the home web page and click on the button “GPIO Setup”.

You must know the diagram of your board. The minimum you need to know is what GPIO controls what Relay. It is also useful to know when Relay is on (GPIO=1 or GPIO=0) Depending on the drivers either one can be true although it is more likely GPIO=1 to turn relay on. This is called positive logic and will come in play when setting the relays. Zumungo can handle negative logic as well.

ATTENTION !

You can easily blow and damage your WiFi relay board if you do not observe important rules and restrictions.

1. Never change the GPIO settings while your WiFi relay board is connected to external inputs or outputs. For example if a particular input is connected to the ground by an external sensor and you re-program it to become an output than you are shortening an output and excessive current will blow your relay board.
2. The outputs have limited voltage (3.3V and current 20 ma capabilities). If you connect external “dumb” relay board please make sure that it has isolated inputs (opto coupler) and is sensitive enough not to draw more than 20 ma current. There are many boards available that meet those requirements.
3. The Inputs are very sensitive and you can't connect external devices that will burn them. The

input voltage should be within the limits from 0V to 3.3V max. As far as the inputs are digital (logical 0 and logical 1 only) The best is to use sensors that provide relay type of output (closed or open circuit). The inputs are internally pulled up to 3.3V so open circuit will be “high” or logical 1 and closed circuit will be “low” or logical 0. Avoid using sensors that are analog in nature directly. Such sensors must have their own processing board and generate “open” or “close” contact on the output.

10. Modes of operation

10.1 Normal Mode.

In this mode each relay responds to commands ON and OFF.
Commands are available to turn the Relay ON for certain time and turn it OFF automatically thereafter. The syntax of the commands will be discussed later.

10.2 Normal Mode with Override (OVR).

In this mode each relay responds to commands ON and OFF, however those commands can be overridden by an external input.

For example: if your relay controls a sauna you can issue a command to turn sauna ON for 2 hours and have a high temp sensor connected to the respective input. In OVR mode the high temp sensor will take over the control turning the sauna ON and OFF depending on the temperature until the 2 hours expire at which time the relay will be OFF.

Please note that the Inputs (if configured as inputs see GPIO configuration) are firmly assigned to the 4 on board relays as follows

IO#1 → Relay #1
IO#2 → Relay #2
IO#3 → Relay #3
IO#4 → Relay #4

IMPORTANT !

Inputs affect the Relay status and operation by it's potential (or logical 0 or 1) level not by transition from 0 → 1 or 1 → 0, which is the case in another mode called MONO.

You will have the option to choose positive (POS) or negative (NEG) active state of the IO in OVR mode.

In the previous sauna example if your temp sensor provides close contacts when the temperature is high, than you will select OVR with NEG and vice versa.

10.3 IO Mode

In this mode the relay is controlled by the corresponding input if such input is assigned (see GPIO setup). POS and NEG options are available for additional flexibility. When POS logical 1 at the input will turn the relay on and logical 0 will turn it off. NEG is the other way around.

In IO mode the selection of OVR is irrelevant. IO Mode with no OVR = IO Mode with OVR.

10.4 MONO modes N/R /K.

In this mode the relay is controlled by the corresponding input if such input is assigned (see GPIO setup). The word MONO comes from the term “monostable” that describes circuits with one stable state. The other possible state is considered unstable or temporary. This is exactly how the relay performs in MONO mode. Normally the Relay is OFF and this is it's stable status. When a transition happen at the properly assigned IO associated with the relay than the relay turns ON for pre-set time from 1 to 9999 seconds. The Relay returns to its normal state after the expiration of the pre-set time interval.

IMPORTANT ! It is the transition (trigger) 0 → 1 or 1 → 0 that matters in mono mode. Whereas in OVR mode it is the value 0 or 1 that matters.

10.4.1 MONO N (Normal)

In this mode the trigger will activate the relay for preset time of up to 9999 sec. The relay will return back to its original (stable) state after the expiration of the preset time.
The board will ignore any and all additional triggers while the relay is in active state . See the example below.

10.4.2 MONO R (Recurring)

In this mode the trigger will activate the relay for preset time of up to 9999 sec. The relay will return back to its original (stable) state after the expiration of the preset time.
Any new trigger that occurs during active state will re-trigger and add the pre-set time to the total time. Multiple triggers can increase the active time significantly. See the Example below.

10.4.3 MONO K (Kill)

In this mode the trigger will activate the relay for preset time of up to 9999 sec. The relay will return back to its original (stable) state after the expiration of the preset time or if and when forced by another trigger.
Any new trigger that occurs during active state will return the relay to its original stable state. See the Example below.

Examples:

a) **MONO mode N**. Turn public restroom hand dryer for 1 minute – connect a push button to IO and run the relay in MONO mode N. Let the relay power up the dryer. Short push of the button will start the dryer for 1 minute. You can not trust the public how often they will push the button. That's why you run in N mode. A new cycle will start only after the previous one ends and the dryer stops.

b) **MONO mode K.** Control hot iron in a hotel room. Hot iron can cause fire if forgotten. That's why you control the hot iron by push button connected to IO port and our relay will power the hot iron for 30 minutes only. If guests need more they push the button again after 30 min. expire. If however they want to leave the room earlier then they don't to wait for the 30

min. cycle to finish. Just push the button again and K (kill) the power to the hot iron.

c) **MONO mode R.** Control ventilation fan in the kitchen. Connect IO to a push button. Relay will power the exhaust fan for 10 min. after push of the button. You just fried fish that smells too much and you realize that 10 minutes will not be enough. No need to come back every 10 minutes. Just push the button 3 times and you'll get 3 x 10 min =30 min. in R (recurring) mode.

10.5 OVR-kill mode

In “OVR-kill” mode the relay will be turned off (killed) first time the respective IO becomes active and (unlike normal OVR mode) will not be turned on again by IO. This mode can be used as a fuse. If for example your relay is heating a sauna. You can hook high temp sensor (fire alarm sensor) to IO and stop this relay from heating the sauna if fire alarm triggers.

10.6 Schedule Mode

The Relay is turned on and off by previously set schedule. Up to three different schedules are supported with up to 6 events per schedule. ON and OFF times can be in absolute time or relative to SunRise (SR) or SunSet (SS).

Examples of schedules

	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF
Sch1	SR-15	SR+5	10:00:00 AM	10:15:00 AM	12:00:00 AM	12:45:00 PM	02:30:00 PM	02:45:00 PM	04:00:00 PM	04:15:00 PM	SS	SS+20
Sch2	04:00:00 AM	04:30:00 AM	09:00:00 AM	11:15:00 AM	12:00:00 AM	12:30:00 PM	02:30:00 PM	02:45:00 PM	04:00:00 PM	04:15:00 PM	SS-10	SS
Sch3	02:00:00 AM	02:15:00 AM	10:00:00 AM	10:15:00 AM	12:00:00 AM	12:45:00 PM	02:30:00 PM	02:45:00 PM	04:00:00 PM	04:15:00 PM	10:00:00 PM	10:30:00 PM

10.7 Schedule mode with OVR

Same as scheduled mode with the exception that a properly assigned Input can overrule the schedule when active and have no effect when non active. Active can be POS or NEG for additional flexibility.

Example: You setup your relays to irrigate your yard per schedule. You may use external rain sensor and attach it to the relay board. If you use Schedule with OVR mode you will be able to stop the irrigation when it rains.

10.8 One wire temperature sensors mode.

Up to (32) 1 wire temperature sensors (based on DS18B20 chip) can be hooked to a single GPIO input of the relay board. Sensors are setup automatically with 12 bit resolution. Each sensor will report temperature but can also act as a trigger completely emulating the hardware IO triggers (see above) in all modes – Normal with OVR, IO mode, MONO and OVR-kill. If 1wire sensor is assigned to a relay than such sensor will drive the relay and nothing else.

The GPIO input for 1 wire is setup in the GPIO setup webpage.

One wire sensors also need GDN and +3.3V connection.

The setup and control of 1wire is by POST JSON (see sample JSON file below) to the this web address

URI: 192.168.1.249/1wire
Header: Authorization: pass

where 249 is example only. Put the real address there.
Where pass is example only. Use the same password as for the Zumungo web pages (see p.5 above)

Sample JSON (for 5 sensors)


```
{
"board_name": "Z",
"IP": "192.168.1.111",
"ReportIP": "192.168.1.248",
"scan": "25000",
"sensors": [
{
"name": "test1",
"hex": "2898d5751b1301c7",
"TempF": "77.45",
"TempC": "25.25",
"TriggerF": "83.00",
"hystF": "1.00",
"val": "194",
"triggered": "1",
"triggered_level": "1",
"live": "1",
"relay#": "1",
"error": "0"
},
{
"name": "test2",
"hex": "28cac65c1f1301c8",
"TempF": "77.22",
"TempC": "25.12",
"TriggerF": "81.00",
"hystF": "0.50",
"val": "192",
"triggered": "1",
"triggered_level": "1",
"live": "1",
"relay#": "2",
"error": "0"
```

```
},
{
"name": "test3",
"hex": "28aa38d31713028b",
"TempF": "77.00",
"TempC": "25.00",
"TriggerF": "82.00",
"hystF": "0.75",
"val": "190",
"triggered": "1",
"triggered_level": "1",
"live": "1",
"relay#": "3",
"error": "0"
},
{
"name": "test4",
"hex": "28aab1d417130235",
"TempF": "77.45",
"TempC": "25.25",
"TriggerF": "84.00",
"hystF": "3.00",
"val": "194",
"triggered": "1",
"triggered_level": "1",
"live": "1",
"relay#": "4",
"error": "0"
},
{
"name": "test5",
"hex": "2893265b1f13012c",
"TempF": "77.34",
"TempC": "25.19",
"TriggerF": "84.00",
"hystF": "5.00",
"val": "193",
"triggered": "1",
"triggered_level": "1",
"live": "1",
"relay#": "3",
"error": "0"
}
]
}
```


Where

board name – must be the name of the WiFi relay board. Error will be returned if it is not.

IP address – must be the IP address of the relay board

Report IP is the IP address of a TCP server where 1wire will report by TCP in the same JSON format

scan – is time in ms of how often the software will take temperature measurements and report back

name – the name of the temperature sensors (multiple sensors can be hooked to the same 1wire interface, including other 1wire devices that are not temp sensors or temp sensors of another brand). Default names starting with name1, name2, etc are assigned automatically until a real name is assigned by the user. Names are for convenience only. Real action is based on sensor's hex address (below)

hex – hexadecimal value of the 1wire address. Each sensor is identified uniquely by this address. It is used in the control JSON and it rules if the name is mistaken.

TempF – temperature in F

TempC – temperature in C

TriggerF – the trigger temperature in F. Turn on relay when temperature drops below TriggerF. Turn off relay when temperature reaches TriggerF+hystF. Ignore (disable) triggers if TriggerF is set to 0 by control. Relay # may still show for sensor with disabled trigger, but it will not be active and also error flag will be raised to indicate this fact.

hystF – hysteresis in F. Used in calculations see above

val – raw value of the temperature or any other value if sensor is not temperature as provided by the "1wire" device.

Triggered - "0" if not triggered "1" if triggered see TriggerF above.

Triggered level – Defines the polarity of the trigger. If 0 than the respective GPIO will be 0 when triggered and vice versa.

live – if "0" device is dead not reporting if "1" device is live reporting

delete – if "0" no action is taken if "1" than delete this device from the list

error – reserved for reporting errors. No error if "0"

relay – r (integer 0 to 4) is the relay number triggered by the sensor emulating IO (1 to 4). Multiple sensors can target the same relay. Resulting action will be LOGICAL AND of the individual triggers. If relay=0 than this 1wire sensor has no affect on the operations.

scan – how often to read the sensors in ms

The list between the […] is as long as many "1wire" devices are hooked up.

Devices that are deleted do not show. However if the same device is brought live again than it shows up with default name "name5" for example.

Control JSON can be of any length. For example often it is practical to start by sending this JSON only


```
{
        "board_name": "MRBoard",
```

```
        "IP": "192.168.1.249",
        "ReportIP": "192.168.1.248"
}
```


1wire will respond with full JSON, where you will find the number of active sensors and their addresses. Copy this JSON response to design your future control JSON.

ONLY the following fields in JSON are subject to control.

ReportIP, scan, name, TriggerF, Triggered Level, HystF, delete and relay.

1wire also responds to HTTP GET command from a regular browser.

192.169.1.249/1wire

where .249 is an example only. Use board's real IP address here. Unlike JSON POST control that is secured by Authorization: "string" (see above) , the GET is not secured. Anyone who knows the IP address of the board can get the information in JSON format however he can NOT control or change anything by HTTP GET.

11. Commands

Commands can be issued over HTML, TCP and MQTT. Command format is the same for all media excluding HTTP where commas (,) are replaced by (&).
TCP and MQTT commands are not case sensitive.
HTTP commands are case sensitive.

Examples:

MQTT command OpenRelay?Name=MR1, n=3, t=5, pp=mypass
TCP command OpenRelay?Name=MR1, n=3, t=5, pp=mypass
HTTP command http://192.168.1.123/OpenRelay?Name=MR1& n=3& t=5& pp=mypass

OpenRelay – activates the relay

OpenRelay?Name=xxx, n=x, t=x, pp=xxxxx

where

Name – the name of the relay board
n – number of the relay in multi relay boards from 1 to 10
t – time (in seconds) the relay will open. 0 – for indefinite time
pp = password for the relay board (default is WiFiRelay123456)

Response (to a command to open Relay#1 for 10 sec.):

2019-06-30T01:29:32, Warning=11, ZumungoWiFiRelays, Relay #1 open was triggered by command

2019-06-30T01:29:32, Confirmation=10, ZumungoWiFiRelays, Relay #1 Opened!

2019-06-30T01:29:42, Confirmation=20, ZumungoWiFiRelays, Relay #1 Closed!

Response (if Relay was opened by IO)

2019-06-30T01:48:22, Warning=12, ZumungoWiFiRelays, Relay #1 open was triggered by I/O

2019-06-30T01:48:22, Confirmation=10, ZumungoWiFiRelays, Relay #1 Opened!

CloseRelay – deactivates the relay

CloseRelay?Name=xxx, n=x, pp=xxxxx

where

Name – the name of the relay board
n – number of the relay in multi relay boards from 1 to 10
pp = password for the relay board (default is WiFiRelay123456)

Response:

2019-06-30T01:36:12, Warning=21, ZumungoWiFiRelays, Relay #1 close was triggered by command

2019-06-30T01:36:12, Confirmation=20, ZumungoWiFiRelays, Relay #1 Closed!

Setup timer – sets up one unit of a particular timer (3 timers are available with 6 units each)

Timers? Name=xxx, n=x, sch=x, u=x, st=hh:mm:ssAM/PM, en= hh:mm:ssAM/PM, d=1111111, m=111111111111, pp=mypass

where

Name – the name of the relay board
n – Relay number within the board
sch – schedule number from 1 to 3
u – unit numbers from 1 to 6
st – start time in absolute time (12 hour format) or relative to SunSet (SS) or SunRise (SR)

Examples

11:35:00AM
04:30:00PM (please note the leading 0 can not be omitted)
SR+15 – fifteen minutes after Sunrise
SS-30 – 30 min before Sunset

en – stop time same format as start time

d – days of the week this unit to run only 1 and 0 allowed in 7 positions representing 7 days of the week starting with Sunday. 1 -will run 0 – will not run.

Examples:

1111111 will run all days of the week
0111110 will run every business day will not run Sat and Sun.

m – months of the year this unit to run only 1 and 0 allowed in 12 positions representing 12 months of the year starting with January. 1 -will run 0 – will not run.

Examples:

111111111111 will run all months
011111111100 will run all months except for Nov., Dec. and Jan.

pp – password

Response (example):

2019-06-30T01:38:42, Confirmation=40, ZumungoWiFiRelays, Relay #1, Schedule1, Unit1, Setting: Start Time=03:00:00 PM, Stop Time=03:05:00 PM, Day:1111111, Month:111111111111

Setup mode of operation

Mode?Name=xxx,Choice=x, n=x, P1=x, P2=x, P3=xxxx, P4=x, pp=mypass

where

Name – the name of the board
n – one digit relay number within the board
Choice is 1 digit number as follows

1 – Normal mode
2 – IO control mode
3 – Run Schedule 1.
4 – Run Schedule 2
5 – Run Schedule 3

P1 – one digit specifying the polarity 0 positive and 1 negative
P2 – one digit specifying IO effect on Relay's performance if an IO is associated with a Relay
    0 – None
    1 – OVR (Override mode see description of OVR mode above)
    2 – MONO (MONO mode see description of MONO mode above)

P3 – up to 4 digits time in seconds for MONO mode only (omit if not selecting MONO mode)
P4 – one digit submode of MONO mode (see description of MONO mode above)
    0 – Normal (N)
    1 – Recurring (R)
    2 – Kill (K)

P3 and P4 are related to MONO mode only and can be omitted for any other mode. They are ignored if specified for any other mode than MONO.

pp – password

Response to set normal mode:

2019-06-30T01:41:13, Confirmation=31, ZumungoWiFiRelays, Relay #1,Change,Mode=1,pos=0,ovr=0

Response to set OVR mode:

2019-06-30T01:44:33, Confirmation=31, ZumungoWiFiRelays, Relay #1,Change,Mode=1,pos=0,ovr=1

Response (error) to set OVR mode if IO not set

2019-06-30T01:43:12, Error=39, ZumungoWiFiRelays, Relay #1 OVR mode is not set: relay doesn't have associated I/O assigned

Response to set IO control mode:

2019-06-30T01:51:06, Confirmation=31, ZumungoWiFiRelays, Relay #1,Change,Mode=2,pos=0,ovr=0

Response to set IO MONO mode :

2019-06-30T01:53:12, Confirmation=31, ZumungoWiFiRelays, Relay #1,Change,Mode=2,pos=0,mono,p3=20,p4=N

Get MAC address of the board

MAC?Name=xxx, pp=mypass

where

Name – the name of the board and
pp – password

Response:

2019-06-30T01:56:45, Confirmation=60, ZumungoWiFiRelays, MAC address is DC4F220BFABB

Get IO status

iostatus?Name=xxx, n=x, pp=mypass

where

Name – the name of the board
n- single digit the number of the IO port
pp – password

Response:

2019-06-30T01:58:12, Confirmation=72, ZumungoWiFiRelays, I/O #1, status=1

Get Relay Status

Relaystatus?Name=xxx, n=x, pp=mypass

where

Name – the name of the board
n- single digit the number of the Relay
pp – password

Response:

2019-06-30T01:59:47, Confirmation=71, ZumungoWiFiRelays, Relay #1, status=0

Get Relay board Status

Status?Name=xxx, pp=mypass

where

Name – the name of the board
pp – password

Response:

2019-06-30T02:01:22, Confirmation=75, ZumungoWiFiRelays, Status=02901901901920401901 9019

Explanation of Status=xyzxyzxyz.... in the Response

repeat xyz n times no spaces
xyz repeats n times one for each relay and i/o
n- number of relays/io assigned to the board

x - relay status 1 on, 0 off, or if it is an IO than 2 on 3 off
y - mode of operation of this relay from 1 to 5
z- related to IO 0 pos no ovr, 1 neg no ovr, 2 pos ovr, 3 neg ovr, 9 this is not IO

Setup ports – sets up communication ports for the relay board

Port?Name=xxx, http=xxxx, tcp=xxxx, pp=password

where

Name – the name of the board
http – http port number (default 80)
tcp – tcp port number (default 8000)
pp – password

Important ! Add :xx to the web address of the Relay web pages if you change the http port to any other than default port 80 where xx is the new port number.
Example:

before
http://192.168.1.180
after you change the http port number to 82
http://192.168.1.180:82

Response:

2019-06-30T02:10:56, Confirmation=80, ZumungoWiFiRelays, New Port Settings
http=80, tcp=7000, callAll=60560

CallAll

This command doesn't have attributes. It is a multi-cast command over UDP. The relay board is UDP server listening to port 60560 (default, can be configured to any other value at "Board setup page). Use any UDP client (such as SocketTest for example) to send out the command. To broadcast use address 255.255.255.255

Each relay board that is alive on the network will respond like this over UDP

Name, FW=2.23.140518, IP=192.168.1.172, TCP=8000, HTTP=80, 019019019019200200200300, Location

where FW is the firmware version, IP is the IP address of the board and the 24 numbers string is the status of the board (See status command).

12. One wire temperature sensors

12.1. General information

ZUMUNGO is capable of managing multiple "1 wire" devices. Most practical and most popular of them all is the "1 wire" temperature sensor DS18B20 that can be used to measure temperature as well as for triggering relays.

https://www.maximintegrated.com/en/products/sensors/DS18B20.html

Recommended to use +3.3V to power to the 1 wire devices. Recommended 4.7K resistor between the live 1 wire and +3.3V power.

ZUMUNGO will report the temperature of each connected 1 wire temperature sensor by JSON file over TCP connection. No TCP server no problem. You can read the same information at a web page

http://192.168.1.249/1wire (example only. Replace the IP address ...249 with yours)

ZUMUNGO will trigger relays ON and OFF depending on the temperature reaching some preset thresholds and taking in consideration preset hysteresis.

12.2 Setup

Unlike all other ZUMUNGO functionality the temperature sensors are driven by JSON files for both control and reports.

First, go to the web site of the board and setup the GPIO that will connect to the 1 wire sensors. Only one GPIO can be set for that but you may have as many sensors as you wish as they all share same 1 wire bus. Pls. Make sure that the GPIO selected is not used for anything else in the settings.

Second, connect all of your 1 wire DS18B20 temperature sensors to the selected GPIO. DS18B20 has three pins – ground, power and data. Connect the ground to the ground of your ESP board. Connect the power to +3.3V power of your board and connect the data pin to the selected GPIO. Place 4.7K resistor between the data pin and 3.3V. Repeat as many times as many DS18B20 you want installed (one resistor only for all).

Third, setup ZUMUNGO software by

a) HTTP POST JSON file to the IP address of the ESP board as in the example below.

Note: Requires Header: Authorization: "pass" , where pass is the password for the board configurable at the board's web pages.

Example:

```
{
        "board_name": "MRBoard",
        "IP": "192.168.1.249",
        "ReportIP": "192.168.1.248",
        "scan": "15000"
}
```

where

IP – is the IP address of the ESP board
ReportIP* – the IP address where ZUMUNGO will report the temperature and other information by TCP protocol. You should have TCP server listening on this IP address and port if you wish to receive such information. However it is not mandatory. Same information is available by HTTP GET to the IP address of the board

http://192.168.1.249/1wire

Note. Default TCP port is 8091 (can be changed at the web page)

Scan – is the scan interval in ms. ZUMUNGO will read the temperature every "scan" ms and will report to the Report IP address by sending JSON file (see below).

Example of HTTP POST address and authorization:

192.168.1.249/1wire (where .249 is example only replace with real)

Header: Authorization: pass (where pass is the board's web pages password that can be changed at ZUMUNGO web pages)

12.3 Reports

ZUMUNGO will automatically find and report all connected 1 wire temperature sensors by sending the following JSON file to the Reporting IP or by HTTP (see above).

Example JSON for 5 sensors.


```
{
"board_name": "Z",
"IP": "192.168.1.111",
"ReportIP": "192.168.1.248",
"scan": "25000",
"sensors": [
{
"name": "test1",
"hex": "2898d5751b1301c7",
"TempF": "77.45",
"TempC": "25.25",
"TriggerF": "83.00",
"hystF": "1.00",
"val": "194",
"triggered": "1",
"triggered_level": "1",
"live": "1",
"relay#": "1",
"error": "0"
},
{
"name": "test2",
"hex": "28cac65c1f1301c8",
"TempF": "77.22",
"TempC": "25.12",
"TriggerF": "81.00",
"hystF": "0.50",
"val": "192",
"triggered": "1",
"triggered_level": "1",
"live": "1",
"relay#": "2",
"error": "0"
},
{
"name": "test3",
"hex": "28aa38d31713028b",
"TempF": "77.00",
"TempC": "25.00",
"TriggerF": "82.00",
"hystF": "0.75",
"val": "190",
"triggered": "1",
"triggered_level": "1",
"live": "1",
```

```
"relay#": "3",
"error": "0"
},
{
"name": "test4",
"hex": "28aab1d417130235",
"TempF": "77.45",
"TempC": "25.25",
"TriggerF": "84.00",
"hystF": "3.00",
"val": "194",
"triggered": "1",
"triggered_level": "1",
"live": "1",
"relay#": "4",
"error": "0"
},
{
"name": "test5",
"hex": "2893265b1f13012c",
"TempF": "77.34",
"TempC": "25.19",
"TriggerF": "84.00",
"hystF": "5.00",
"val": "193",
"triggered": "1",
"triggered_level": "1",
"live": "1",
"relay#": "3",
"error": "0"
}
]
}
```


where the data for each sensor is as follows

name – set by user (see Control section below)
hex – hexadecimal unique ID of the sensor
Temp – temperature in C and F respectively
TriggerF – the trigger temperature in F
hystF – hysteresis in F
val – hexadecimal value of the sensor (could be used for other non temperature 1 wire devices)
triggered: 0 if not triggered and 1 if triggered
live: 1 if sensor is live and operational 0 if sensor is temporarily not available
relay # - 1 to 8 the relay that the sensor controls.
Error : error # 0 if no errors

12.4 Control

Control is by HTTP POST JSON to IP address setup (see above) and Authorization Header (see above)

Subject to control are the following parameters (in **bold** in the Example below). There is no need all of

them to be supplied.

Example:


```
{
        "board_name": "Sauna",
        "IP": "192.168.1.249",
        "ReportIP": "192.168.1.248",
        "scan": "15000",
        "sensors": [{
                "name": "SaunaTemperature",
                "hex": "2898d5751b1301c7",
                "TempF": "71.71",
                "TempC": "22.06",
                "TriggerF": "150.00",
                "hystF": "10.00",
                "val": "161",
                "triggered": "0",
                "live": "1",
                "relay#": "4",
                "error": "0"
        }]
}
```


Note: "hex" is not subject to change and it is a MANDATORY field as it is used to identify the sensor. To get the **"hex"** values of all sensors you need to run a report first (see above). The sensors is identified by it's **"hex"** value. The **"name"** can be changed at any time. It is for convenience only and is not mandatory.

12.5 Special consideration about 1 wire sensors

12.5.1 1wire creates “virtual IO” that rides on top of a physical IO ONLY. 1wire can NOT create IO of its own. If 1wire is active (live) it rules and the physical IO is irrelevant. Once 1 wire is non active (disconnected or deleted) than the physical IO takes over. **WARNING !** This can be damaging if IO drives the relay on when 1wire fails for any reason. Make sure IO is in inactive state (ground the respective GPIO).

12.5.2 As far as 1wire emulates IO, you can use 1wire trigger in ALL IO modes including OVR, MONO, Kill etc.

12.6 How to de-activate and re-activate 1wire

You can de-activate 1wire action upon a particular Relay # by 3 different ways

a) change Relay# to 0.

Example: Was "relay#"="2" change to "relay#"="0". This will relinquish the control of relay #2 by 1wire and assign the control to associated IO.

b) set "Trigger F" = 0 - same effect as above

c) physically disconnect the 1wire sensor – same effect as above effective not immediately but by the next scan (scan interval is configurable see above).

Just undo any of the above actions in order to re-activate the 1wire control of a relay. Please note that the board will "remember" all the settings and will activate them automatically upon re-activation. Please be aware of this if you are changing the relay during reactivation as settings for the previous relay may not be appropriate for the new one.

12.7. How to delete 1wire sensors

In order to delete a sensor insert line "delete": "1" on top of the JSON entry for the sensor like this

```
{
        "delete": "1",
        "name": "name1",
"hex": "28ff0978b5160322"

}
```