Hello YouTubers, here is the guy with the Swiss accent. I wanted a nice automation system for my lab using the things I built in my last videos. So, I use my Raspberry Zero with Node-Red and Mosquitto as the “brains”, some Sonoffs as switches for my LED lamps, and some Wemos Minis as PIR sensors to make sure, that the lights are on and off at the right time. And I want to use one of the new “frameworks” to program the Sonoffs.

And of course, I also want to have some “modern” stuff. This is, why I wanted to include Amazon’s Alexa into the whole play. I like this concept where my orders are followed without arguing… Very rare, these days. This alone should be worth the 60 dollars for the Amazon echo box.

And, not to be forgotten, the whole thing should be IOTappstory compatible…

Is this a good idea? Do these frameworks really reduce the programming effort? How can we make Node-Red behave how we want it?

In this and the next video, we will learn a lot about these components and at the end, you will have some additional knowledge to make an educated guess for your own projects. And we will even write some lines of java script to augment existing functions in Node-Red.

So, let’s start.

First an overview of the final system: There are two LED lamps, one for the main work space and one for the bench. To detect where I work, I have two PIR sensors. My newly created Pi Zero is in the middle and runs Mosquitto MQTT broker, Node-Red, SQlite and webadmin. As an additional system, I recently added an Amazon Echo Dot.

What is the scenario? If I sit in front of my computer, I only want the main lamps on. If I work on the bench, I want all lamps on. This should be switched by the PIR sensors, but I also want the ability to override the sensors from my computer, or with Alexa.

If I ask e.g. Alexa to keep the lights on or off, I want an immediate reaction. All devices are, of course, connected via MQTT and the Raspberry Zero should be able to log whatever messages I want to log into a SQlite database. And I want to monitor the Pi Zero using Webadmin.

Today, we will build, program, and test the devices and in the next video, we will connect them to the Mosquitto broker and implement the logic in Node-Red.

Frequent viewers remember, that I made some videos on how to program the Sonoff devices. Each program was a special build for a particular purpose. In the meantime, some frameworks like Tasmota or Espurna submerged. They provide an infrastructure to run Home Automation devices and it should be easier and faster to program these devices than writing individual programs for each one. So, my first decision was to test one of these frameworks. Because I like the style of tinkerman, I will use his Espurna framework. This framework can read switches or sensors, can transfer the readings either to the built-in LEDs or relays, or transfer the readings via MQTT to other devices. Also built-in is an interface to Alexa and some other cool stuff we will not cover here. You just have to select, which device you want to use. The rest is done by the compiler. Espurna also has a web-interface to change some values during operation. If your device is not foreseen by tinkerman, you can define your own. As an example for a new device, I will use Espurna to build our PIR sensor.

A big advantage of such frameworks is, that you can change the device from for example a normal Sonoff Switch to a Sonoff dual. Then, the only thing would be to select another device, compile the code, and flash it to the dual switch. Cool.

So, let’s start with the programming of the Sonoff switch. Here you have two possibilities, as usual, a complicated and an easy one. Either you chose the complicated way and you follow the link in the comment and download the Espurna project from Github. Tinkerman uses many libraries and you have to install all of them, which is some work. But fortunately, I did all this stuff for you and uploaded a ready-made file to IOTappstory.com. So, the easy way is

Just four easy steps:

1. Flash and run ESP\_Loader to get the MAC address of your device
2. Prepare your device and your project on IOTappstory.com
3. Enter and save credentials of your Wi-Fi on your ESP and EXIT
4. Reboot and automatically download sketch from IOTappstory.com

You find a detailed description in IOTappstory.com. I include a link in the comments.

Because I do this frequently, I built a small device to do that. Those of you which do not know, how to do it, please watch video #93. Please make also sure, that you use the version 1M-128k for the Sonoff. And please run your Serial Monitor and note the MAC address of your device. Now, you can log into your account on IOTappstory.com, go to control panel and devices, and create a new device. Give it a name, and select the Sonoff switch as a basis. The only thing you have to add is the MAC address from before. Save the device and create a project. I name it “lab light”. In the project, you select the app you need, which is the “Sonoff Standard Espurna”, and select your device you created before. I added here a second device for the bench. All devices in this project will always get the same firmware delivered over the air.

Save the project and make sure, it is “online”. Done. As soon as you switch your Sonoff off and on again, it will get the firmware delivered over the air. And if I update the sketch, you will always get the newest version. You can follow the update process in the serial monitor, if you want.

Now, you should see a new Wi-Fi network. Because this is not your home Wi-Fi network, most people use their mobile phones to do that.

A new screen should show up and you can enter the Wi-Fi credentials of your home network. If no screen appears, connect to 192.168.4.1. After you entered your credentials, press save, and wait for at leat 10 seconds. Then, you press exit and your ESP reboots. Now, it should connect to your Wi-Fi and download the sketch from IOTappstory.com.

and log into your device using admin and admin. The first step is to change your password and then, you can give your device a name. In the future, the MQTT topics automatically will use this name to distinguish between different switches. Here, you find also the IP address for future reference.

Add you home Wi-Fi network or networks and enable “Alexa”, if you want to use it later on.

In the MQTT tab, you have to provide the IP address of your MQTT broker. For our tests, I use cloudMQTT. Because CloudMQTT uses credentials, you have to enter them here. For my private broker, I leave these two fields empty.

If we go now to the Websocket UI of cloudMQTT, we should see the first messages of our Sonoff switch. This is really cool. Not one line of coding…

If you search now for the topic with the name “relay”, and copy-paste it into this “send” topic field, add a “1” as a message and press “send”, you should hear the relay on your Sonoff click. If not, send a “0” and you will hear it. If your serial monitor is still connected, you should also see the messages coming.

So, communication in both ways work and we can continue with the next step.

I started with a Wemos mini board, connected a PIR sensor to pin D5, a red LED to pin D6 and a green one to D7. This was the whole soldering. And then, I printed a small 3D case and ready was the PIR sensor. Before we close the case, we also have to connect the Wemos board with a USB cable, and flash the ESP-loader, this time, the 4M-1M version. Then, we go back to iotappstory.com and create a new device. Exactly as with the Sonoff switch before. Next, we create a second project, called “LAB PIR sensors” and select the device created before. The app this time is “PIR Sensor Espurna”. Save the data, and reset the Wemos. After a minute or so, your Wemos mini is a fully functional PIR sensor, and you again can connect to it using your smartphone. Enter also a new password, the Wi-Fi credentials, and the CloudMQTT credentials. Now, you should also find the first messages of your PIR sensor on CloudMQTT.

But this is not all. As said before, I want to use Alexa. So, we have to add a few more steps. I assume, you already have your Echo registered at Amazon. After logging in, we go to the “Smart Home” tab and press the button “discover devices”. After a while, you will find the board names in the list. They are “WeMo Switches”. This has nothing to do with our “Wemos” boards. Wemo is a commercial brand which developed an interface to Alexa. And a hacker wrote a code to emulate these Wemo products with our devices. This is how the brave new world works… BTW, this library is called “Fauxmo” and is already implemented in Espurna.

If we ask now Alexa to switch our board on, we see the message on CloudMQTT. So, we just created a PIR sensor, which acts also as a Gateway to Alexa. No additional hassle or cost. Big thanks to Xose alias tinkerman!

One remark to Alexa: Do not call your board “switch”.

And I wanted to name the second lamp “bench”. But she did not like my swiss accent and did not understand. So, I decided for “table” which seems perfect for her. Also with the Swiss accent!

Summarized,

* we started to build a fully functional home automation system consisting of two PIR sensors and two Sonoff switches
* We used the Espurna framework to save us a lot of time. This framework is configurable for many Sensors and switches and saved us a lot of time
* To further reduce the effort, I ported Espurna to IOTappstory.com and created two apps on IOTappstory.com which can be used by everybody
* To start with a new device, we had to download the respective ESP-loader and flash it to the device. This is a one-time effort
* After that, we created our devices on IOTappstory.com and linked the device with the app into a project. If we activate the project, our device always gets the newest version of the firmware, because, during boot, it calls “home” and downloads any new version if necessary
* Because Espurna emulates a Wemo switch, we were able to connect our PIR sensor to Amazon’s Alexa. Now, we can also control our lamps with our voice

In the next episode, we will dig into node- red and learn some cool stuff there.

The only remaining thing is to ask Alexa: Alexa, did you like this episode?

I hope, you have an opinion. And you do not forget to press the “like” button accordingly.

Bye.

Alexa, please sing a song!