Grüezi YouTubers. Here is the guy with the Swiss accent. With a new episode around sensors and microcontrollers.

Lately, IKEA jumped on the IOT bandwagon and released a new line. It is called TRÅDFRI smart lightning, and is a direct competitor to Philips hue. Tradfri is Danish and means “wire-free” This line consists of controllers, lamps, and a gateway. As always, IKEA has an attractive price point for its devices. The cheapest set is around 20 dollars. Mine has variable color temperature and costed around 30 dollars.

Recently I got these two devices from my local IKEA store it is a Tradfri LED lamp and a controller. So let's hack it!

including a battery

Dangerous but nicely looking: Actually, you see this as a European E27 I think it's called plug here we can screw it in somewhere. So, let's test if it works and how it works. This is magnetic here and can be taken off. So, we open it up to mount the battery. And here we have the manual even for people who cannot read. We have to switch the lamp on and press this button for more than 10 seconds and then it will be paired I assume. So, you guys still have to be able to read a little bit at least that you can read the five centimeters or two inches. Distance. So, let's try.

One two three four five six seven eight nine ten. So that's for sure more than ten seconds. Personally, I think it was already paired because I bought this as a kit together and it is a nice color, not a cold white, more a warm white. Ahh, it works! I can switch it on and switch it off. So, we have three functions: On/off; the color temperature and the intensity. Now of

course this is nice but I want to use this with my MQTT stuff. So we have to hack this one here.

So always if you start to hack you take out the battery, because otherwise you might be able to destroy somehow your device electrically, and this is not what we want.

So, let's continue on the bench. One thing is interesting here: This is marked from the supplier not from me. It is some handmade stuff here. This is probably not yet a full production run, it's maybe still done with some manual work.

Here we see, we have one module which is soldered to a second module. And we also see here that we have some test points. Fortunately, I got some help. This is a German magazine and they do some stuff with IOT and so on and they also hacked this one here So I have some information about the different pins. I do not need to hack them myself. But it's in German. I will leave a link in the description.

Let's check if it still works because otherwise it's useless.

It still works!

Fortunately, I did not destroy it completely when I had to take it apart.

I connect now all wires according the “MAKE Magazine” with the different colors. I was really glad that I had wires with all these colors. So, we have now a clear separation:

Here we have ground, the test points here all connected and three pins of this module - so let's do some measurements. Now everything is connected to the test points and to some other points here and now we try to find out which one is which. I connect it to the orange and here I have the oscilloscope, and if I press the on/off button you see signal on the orange wire is on/off. And it has negative logic. If I press it, it shows zero, if I release it shows three volts.

So, let's check the next one: Yellow is the left temperature. We hacked the whole thing. We have now three six eight wires and a we Wemos board which has a 3.3 volts output. So, we can power also this one and I will do something like that, I will print probably a round housing that I still can use it manually and I will put in a Wemos board here. Like that. So, I have now to connect all these wires to this Wemos board. And this is done with my third hand which I built once in another video. Let's do it:

I come back when it's done.

Now everything is connected to this Wemos and I have already a programming USB cable and now I do not need a battery because I have this red wire connected to 3.3 volts. So, I get the 3.3 volts from the Wemos chip. Let's now start with programming. Before I start to program, I have a table here with all the different pins and commands and colors and functions

The sketch is based on the pubsubclient library and on the work of Markus Ulsaß from Hamburg. I used his work, but changed quite a lot in the area of pin handling. I also added a function which reads the physical buttons of the remote control and sends it via MQTT. This can be used to control other devices than the IKEA lamp.

Here we have our definitions. The on/off function listens to command 0, and an orange wire connects it to pin D7 on the Wemos.

The «darker» listens to command 2, and a brown wire is connected to pin D5

In order to simplify the pin handling later on, I use an array. This array defines which pin belongs to which command. So, pin D2 belongs to command 0. This is, why D2 is at the first place in the array. All other pins follow in the right order.

If we go now to setup(), we find the MQTT specific functions. The only thing I had to add is the definition of the pins as Input pins. Here you see, that I do not need to define each pin separately, I just need a for loop.

The loop() section consists of three parts: In function «readbutton», I check, if a physical button is pressed on the remote control. If so, I publish a message with the topic IKEA/BUTTON and the command which fits this button. For non experienced C++ programmers, I focus on the The variable msg for a moment. It is defined as an array of 50 characters. This means, that we can stuff up to 50 characters in this variable. But C++ does not know, how long this string really is. Only if we place a «zero» character at the end of our string, C++ can handle it. Our message is just one character long. So, I add a zero character at the second place in the array. Now, everything should work.

Here, I make sure, that I do not send the same command twice.

This part is called every 10 seconds and publishes a «alive» message to the IKEA/STATUS topic. It can be used to check, if something happened to our remote control.

I could have used a delay () to wait the 10 seconds. But then, the whole sketch would not work, because we frequently have to check, if a button was pressed or if a message arrived. This is, why we used this statement to write a «non-blocking» code.

The rest of the loop() is MQTT specific.

But where do we get our messages with the commands? There is nothing in the loop!

This library uses the concept of «callbacks». Here, we defined a so called «callback function”. This function is always called, if a message arrives. Let’s have a look at what this function does: It prints the message in Serial monitor and decodes the command. I use human readable ASCII characters as MQTT payload. So, we have to convert these commands to a number from 0 to 5, because our commands are byte variables.

This is easily done. If we subtract ascii ‘0’ from our human readable command, it is converted into a byte variable. Simple.

To test our sketch, I use my normal Mosquitto MQTT broker running on a Raspberry Pi zero. We built it in video #126.

I also use the Node Red flows which we built in video #127 and #128.

First, I check, if I get the commands and if the commands are executed by the remote control. And really, the lamp goes on and off if I send the command 0. And all other commands also work.

Now, I want to check if I can read the buttons. Again, success.

So, summarized:

We were able to hack the newest IKEA IOT devices and connect them to a MQTT broker using a small Wemos module.

We were also able to connect it to our home automation system without buying an expensive gateway.

In this process, some of us learned some small programming tricks with arrays.

And we learned, why delay() often has to be replaced by non-blocking code

But was this hack a good idea? Yes and no. If we hack available hardware, we often run into the same problem: These devices do not give any machine readable feedback about the current status. In manual mode, this is not necessary, because you can have a look at them and you see, if they are on or off. But if you want to automate them, your computer has to know the status of the device. Like with the brightness, you can just issue too many «darker» commands till you are very sure that the lamp is on the lowest level. Now, you can start to count and you are pretty sure about what happens. This is completely different with the on and off function. Here, your computer never knows, if the lamp is really on or off. And this is not what I call a reliable system. It might work for a long time, but you are never sure. So, this hack is just a first attempt and good, if you are still around to correct the status if needed. In a next hack, we have to dig one level deeper and hack the Zigbee protocol. There, I hope, that we will find absolute commands to switch the lamp really on or off. So, stay tuned.

I hope, this video was useful or at least interesting for you. If true, then like. Bye