The Sonoff CH4 Pro2 is an internet enabled 4 channel smart switch capable of switching 10A loads at 250VAC. It has an RF433MHz receiver built in, which means that each of the 4 channels can be switched by a remote rf switch as well as over wifi. Sonoff supply a range of rf switches, and importantly for me, battery operated outside PIR sensors. All of the Sonoff range of devices are relatively inexpensive, particularly if purchased direct from China.

Sonoff also have an app that can be loaded onto your smart phone to allow for remote on/off switching of the device, as well as some simple rules based switching also.

My requirement is for a security lighting system for my house which switches various lighting zones depending on a number of strategically located PIR devices. I may want certain PIRs to trigger more than one zone, and I also want a number of PIRs to trigger one zone. I also want to trigger zones using a Sonoff key fob, and also remotely using my iPhone. It is very important for me to only switch the lighting during hours of darkness and poor light.

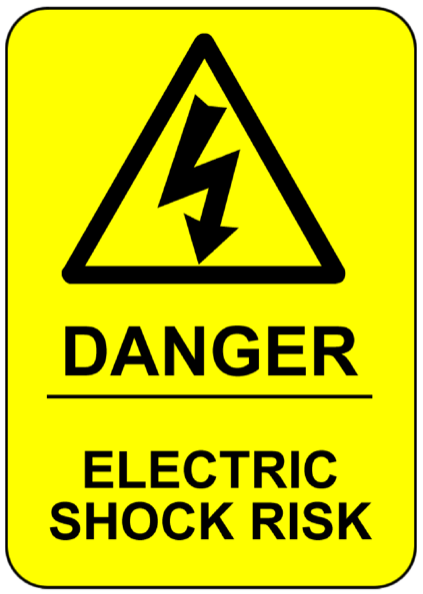
The limitation of the Sonoff unit at the moment is that each zone is only capable of learning one rf code per channel. Whilst it is possible to expand the system using the rf bridge unit that they also offer, the rules based setup is complicated (particularly to switch the lights only after sunset or before sunrise) and this relies on the internet connection to be up and running, and a willingness to submit your data to a server based in China. In researching this, I came across many internet posts from people trying and failing to achieve similar results.

My requirement is for an intelligent switch that meets the requirement above, which means that the algorithms must reside within the switch itself.

Fortunately the Sonoff CH4 has an onboard 32 bit processor (the STM32f030C6T6) with 32k code space – more than powerful enough to handle the task in hand. Sonoff helpfully publish the schematic for the switch, and this made it relatively simple to work create the required software.

The circuit board in the Sonoff CH4 contains 4 user switches which are accessible without removing the cover. These are used to activate one of the four channels via a single press. I have also used these buttons to enter the learn mode by double clicking, and this allows for up to four rf codes to be stored. A long press on the button (5s or more) will erase all learned codes from the zone.

Before going any further it must be said that **never open the case of the sonoff when mains electricity is fed to the unit**. All of the modifications can be done with the unit connected to a suitable DC power supply. I used a 7.5v 1.6A power supply connected to the port labelled Input 1 (DC 5-24v).



If you remove the cover there are 2 banks of four switches labelled K5 and K6 as well as a single slider switch and a single push button.

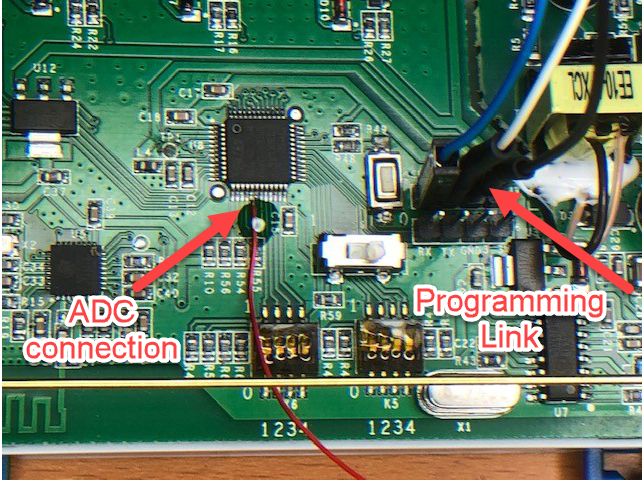
I have used K6 to indicate the PIR on delay (from 5 seconds to 15 mins) and K5 is used to set each of the four channels in either Toggle mode or Delay mode. In Toggle mode each time the zone is activated, the zone will toggle between the on and off states. This is not particularly useful for PIR activation, but will allow the zone to be used with a standard momentary switch, or an rf wall switch. In the Delay mode, each time the zone is activated, the output will come on for the duration set for the zone.

The other requirement that I specified was that the switch must only activate in hours of darkness (or near darkness). That means that we must have provision for a light sensor. Here things get a little tricky. There are two options, the first is to purloin the unused serial port and re-designate it as an I2C port. There is a header (without pins installed) which has 3.3v, Gnd and Tx and Rx pins and it is possible to solder pins to the motherboard to gain access to these. The Tx and Rx pins are connected to the PA9 and PA10 pins on the stm32 processor and fortunately they can be designated as I2C SCL and SDA pins respectively. This would allow a whole host of sensors to be attached to the board.

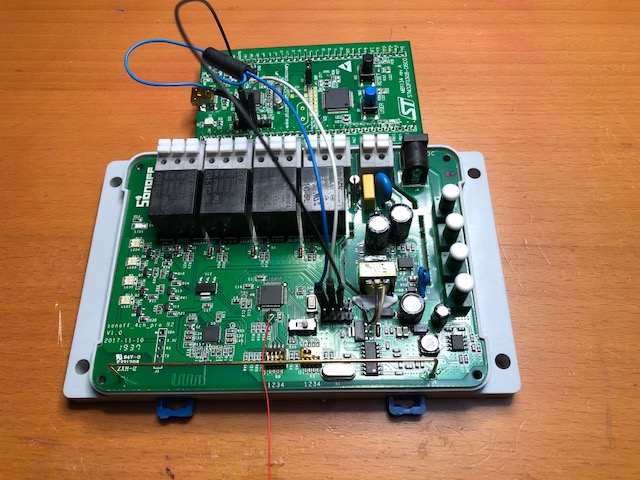
The other option is the cheap and cheerful one, and involves using a spare pin on the processor as an Analogue input (ADC). The problem here is that there are no spare pins available. What there are, are two unused pins (PA7 and PB0) which luckily are adjacent to one-another but have no copper tracks attached. With very careful soldering, it is possible to solder a very fine wire to both pins (placing the wire in the gap). Lightly tin the wire before attempting to solder it, and don’t use any additional solder as you risk ending up with a solder mess. Then hold the wire to the pcb with a spot of glue so that it remains affixed and can’t move. Run the wires (3.3v,Gnd and ADC) to a socket in the case so that the sensor can be plugged in.

In software both these pins are designated as ADC input so that there is no issue with tying them together but we use only PA7 input as the source of the light sensor. We only need to add a cheap Light dependant resistor (LDR) in a simple voltage divider circuit in order to get a measurable voltage which is light dependant – plenty good enough for this application. I solder this wire to one pin on a four pin 2.5mm jack socket along with Gnd and 3.3v. On the socket, I attached a 100k resistor between Gnd and ADC input, and the LDR connects between 3.3v and ADC input.

For programming, there is a header (again no pins attached) on the board which has 3.3v, Gnd and the DIO and CLK pins needed for serial programming on the stm32. In order to connect to the board using the ST-Link utility, I found that I also needed the NRST pin also and this is available on a small pcb pad labelled ‘TP1’ and again with a bit of fine soldering it is possible to connect a wire temporarily (or hold a pin against the pad).



You don’t need an expensive piece of kit for programming, STM provide a development board for a very similar processor (the STM32F0308-DISCO) which costs around $9. As well as the STM32 processor, it also has the ST-LINK serial programmer and debugger which can also be used to program external boards. Simply remove the jumpers and attach the two headers from the two boards with link cables.



I suggest you download two pieces of software from STM. The first is the ST-Link utility <https://www.st.com/en/development-tools/stsw-link004.html> this is a very useful utility which allows you to download and backup the existing firmware (always a good start) and to program the device with new firmware. The second piece of software is the STM32Cube MX software <https://www.st.com/en/development-tools/stm32cubemx.html> which is a graphical tool which allows you to configure the processor with the various input and outputs needed, and which then generates the skeleton software and drivers to allow you to start programming very easily and quickly.

You will then need an IDE and compiler and I chose to download this from Keil <https://www2.keil.com/mdk5/uvision/> . This integrates with the STMCube Mx software, allowing you to create a new project in Keil and then jump directly in order to use the graphical interface to create the skeleton application.

In order to update the firmware:

Copy the source files from github <https://github.com/nickdeacon/PIR-Lighting-controller> into a suitable directory.

Run STCubeMx and load the file Lighting.ioc. Go to Project Manager and make sure that ‘Copy only the necessary files’ is checked.

Generate the files and exit the application.

Run Keil uVision and open the project file contained in the MDK-ARM directory from your directory that you created for the project.

In the project tab, expand the Project item, and right click on Application/User folder. Click ‘Add existing item to group’ and add button.c, flash.c and RFdecode.c to the project.

Click on ‘Options for target’ and choose the Target tab. Alter the IROM1: setting so that the length is 0x6FFF. This ensures that there is no conflict between the code space and the area reserved for storing the RF codes (at 0x8007000)

Choose the ‘Linker’ tab and make sure that your ST-Link programmer/debugger is correctly identified and selected.

You should now be ready to build the project and download it to the Sonoff CH4 Pro unit.

Getting started with the new firmware

For each zone, select whether it should operate in timed/toggle mode using dip switches K5 open for timed, closed for toggle.

Select the desired delay time for all timed zones using the onboard K6 dip switches as follows:

SW4 SW3 SW2 SW1

5 seconds (test mode) close close close close

1 minutes close close close open

2 minutes close close open close

3 minutes close close open open

4 minutes close open close close

5 minutes close open close open

6 minutes close open open close

7 minutes close open open open

8 minute open close close close

9 minutes open close close open

10 minutes open close open close

11 minutes open close open open

12 minutes open open close close

13 minutes open open close open

14 minutes open open open close

15 minutes open open open open

To set the threshold light level below which the PIR sensors will operate the lights, you first need to install the LDR sensor as detailed above. Press the on-board push switch S5 when the light sensor is exposed to the correct light level. The light sensor is read by the unit every 10s, so allow time for the sensor to stabilise before pushing the button.

Power up the device using the connection to 250VAC or the DC connection. If you choose to connect to the Sonoff eWeLink app, then you may need to register the unit with the website, which is normally done by holding one of the buttons on the front for 7s. Once the firmware is updated, you now need to press the pushbutton S5 for 7s.

When you turn on each zone from the app, then that will override the on-board settings and turn on that zone – all RF input will not have effect. Turning the zone off at the app will revert to normal PIR or toggle operation.

To pair an rf device such as a switch or PIR sensor, double click on the relevant user button and set the rf device to pair. When the code has been received the green LED for that zone will illuminate to show that the code has been received and stored. If it fails to pair, then no led will show. This may be due to an incompatible RF code format or that the zone already has four stored codes.

I have not yet tested the ir receive routine fully with a range of RF devices, although it works with both the Sonoff PIR and Sonoff key fob which are the only devices I intend to use.

