Home Alone Monitor

Project Description

Peter K. Boxler

First Version January 2020

New Version April 2024

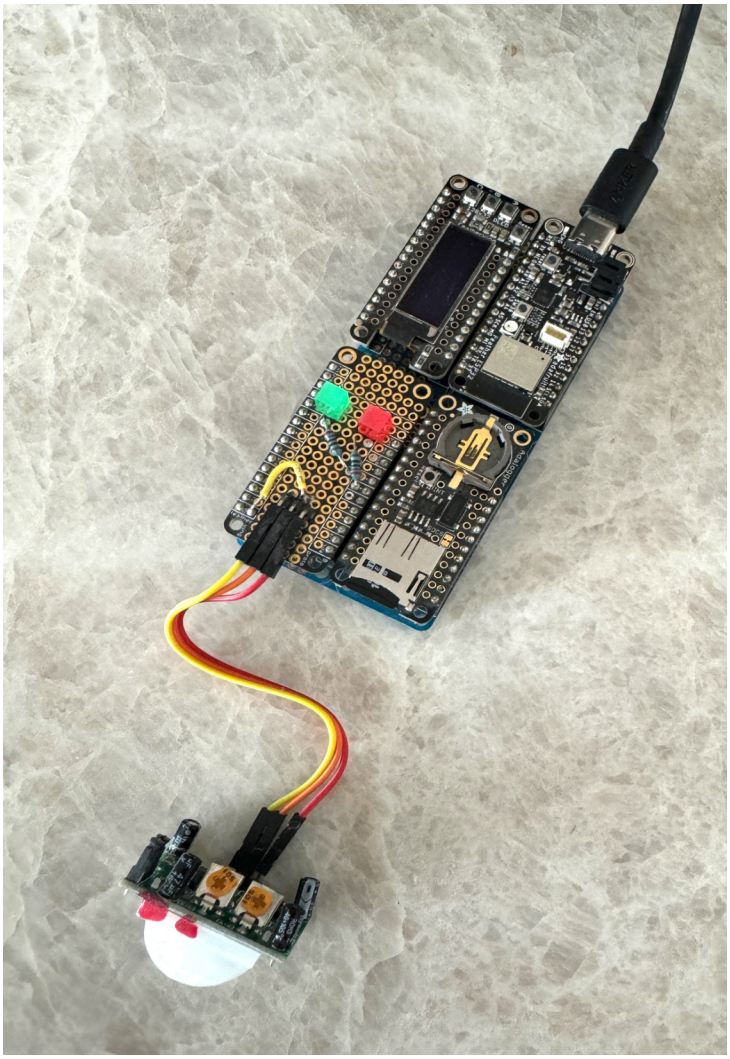
Table of Content

1. Abstract

This documentation presents a small electronic gadget called **Home Alone Monitor** (and not Kevin Home Alone). Some folks concerned with the well being of elderly people living alone could use it to keep track of the movements of their loved ones. This gadget detects, counts and reports movements in its surroundings. For privacy reasons no camera is used, a simple PIR sensor (Pyroelectric Infrared Sensor) detects movement. Movements (or the absence of it) are reported to the cloud (ThingSpeak) . Concerned party (or parties) can inspect the data (the movements or absence hereof) on a website from anywhere in the world.

2. Introduction

Recently a friend told a story of an elderly woman having been found dead by her daughter. Luckily, the daughter came by just one day after the old lady passed away. Everybody has heard of similar stories. That got me thinking and I knew it would be rather easy to build a gadget helping to prevent such stories. However, a couple of days later I discovered that (here in Switzerland at least) one can buy electronic devices that detect movement and they are connected to the landline. So I dropped the plan to build such a device.
A couple of weeks later, however, I came across Ralph Bacons Maker Videos on YouTube and bingo, he describes just such a devices in one of his videos.
Since most of the parts needed to build such a gadget were lying around in my lab I decided to give it a try. Here is what came out of it. It is just a prototype as of now.



Home Alone Monitor Prototype

3. What does it do

Let's assume we have an elderly uncle called uncle Dagobert (the target). He lives alone in an appartement in a city a couple of hours away. He still can live on his own, is even able to walk to the shop or to the city park and he can handle a phone. His neighbours keep an eye on him and they promised to call us if they have not seen him in days.
So this is where this device comes in handy: it detects movements (in the appartement) and counts them. The device is placed in Dagoberts living room. Movement counts are reported to the cloud in defined intervalls (every hour or so) and can be inspected on a smartphone anywhere in the world. The provider ThingSpeak is used for that. At least once a day (in the evening) a push-message is sent to a smartphone reporting the movement count during the day. So even if we forget to check the ThingSpeak website we will have a message on our phone. All of these thresholds and intervalls are configurable - the json configfile resides on a micro-sd card. This scheme gives us some reassurance that all is well with uncle Dagobert (or not, see discussion at the end of this document).

4. What you need

4.1 Hardware

I love the Adadfruit Feather series of boards and therefore decided to build with what I already had. Ralph Bacon device uses an ESP8266 but I was keen on using the more modern ESP32.
So I assembled these Feather boards for my prototype:

- Feather Huzzah ESP32 V2, Product ID: 5900
- Featherwing OLED 128x32, Product ID: 4091
- Adalogger Featherwing, Product Id: 2922
- Featherwing Proto, Product ID: 2884
- Adafruit Quad Side-by-side, Product ID: 4245
- PIR Sensor, Product ID: 189
- Two LED, resistors
- Micro-SD card (any size will do)
- Power Supply 5 Volt 1 A

Note: RTC on the Adalogger is not used.

4.2 Services

- You need a free ThingSpeak account and you need to create a channel and a field within that channel. Detected movements of uncle Dagobert are reported to this channel/field.
- You also need a free Pushover account for sending messages to your smartphone. On the smartphone (Android or Apple) you will need to install the Pushover app.

5. Software

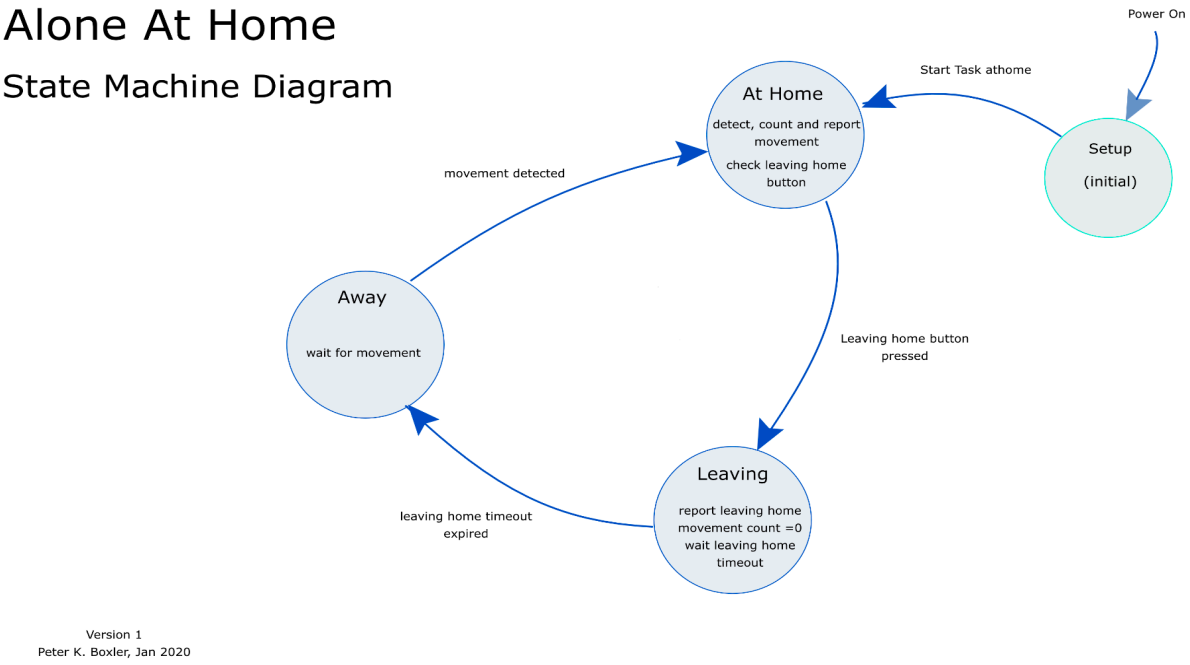
The nature of the problem calls for an implementation of a Finite State Machine. Uncle Dagobert, whose movements are to be detected/reported can either be at Home, Leaving the home or be away.
I choose to adopt Ralph Bacons idea that uncle Dagobert should press a button before leaving the apartement.
Frankly, however, I suspect that in most cases uncle Dagobert simply forgets to do this. So there is a great deal of interpretation when one looks at the movement data reported to the cloud.

Note: at the outset I also had a state NIGHT during which no reporting was done. After a while, I felt that this is not really needed.

I also wanted to make use of the multitasking capabilities of the ESP32 chip.
I used the Arduino IDE to develop the code and for readabiltiy and beauty I arranged the code into a couple of modules. Ralphs code is hard to read - all in one long piece.

6. Finite State Machine

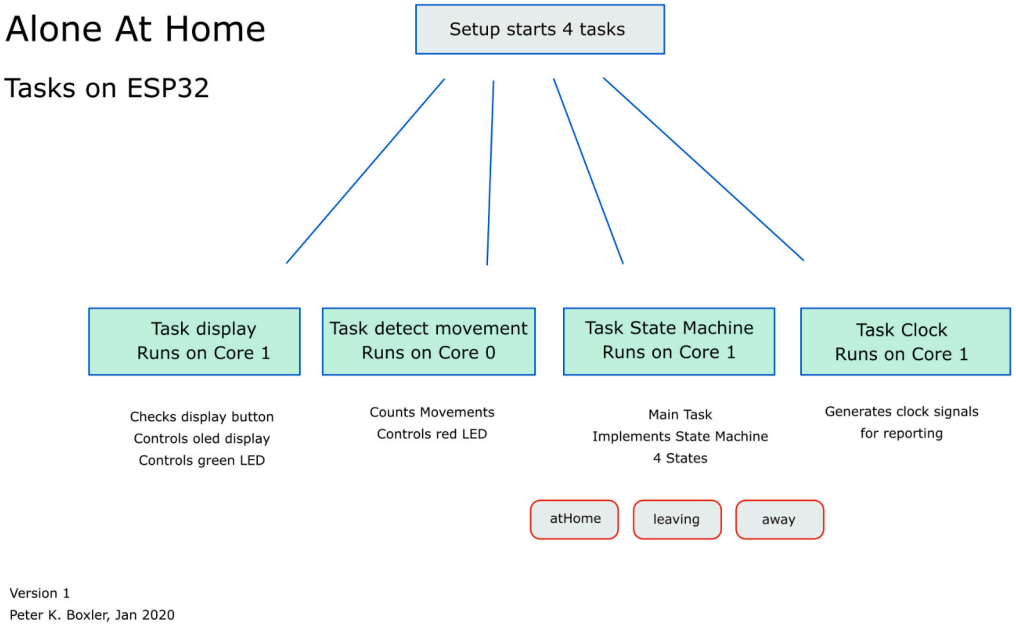
This the state diagram



State Diagramm of the application

7. Multitasking ESP32

This chip has two cores, runs with 240 Mhz clock an there is even a Real Time Operating System (RTS) on board. My code makes use of this capabilities and the setup function (arduino speak) starts 4 concurrently running tasks.



ESP32 Tasks

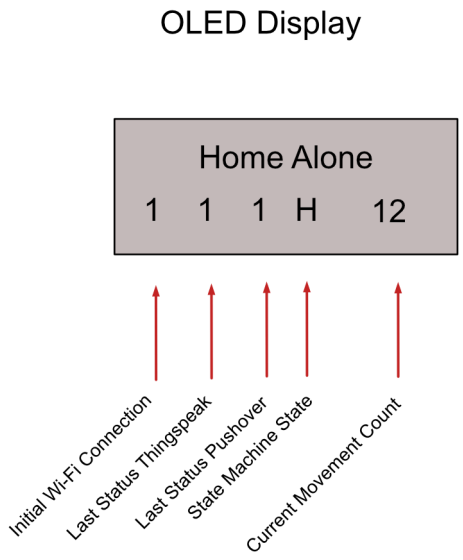
8. Oled Display

8.1 What does it show

A small oled display shows just a few simple internal states of this gadget - the display is kept simple and a large font is used for easy reading.
These values are shown (1 meaning all is well):

- Status of the initial wifi connection (during setup),
- Status of the last reporting to Thingspeak.com
- Status of the last message to Pushover.net
- Current state state of the state machine
- Current movement count

This is the display:



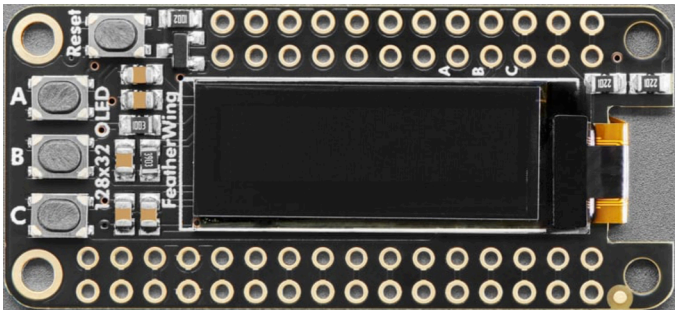
OLED display Home Alone

8.2 Oled Library

The regular Adafruit Library for OLED 1306 displays is used.

8.3 Buttons

There are 3 user buttons (A, B and C) on the FeatherWing OLED 128x32.



Adafruit OLED Featherwing

All three buttons are assigned a function:

- Button A: Pressing this button switches the display on for 30 seconds.
- Button B: Pressing this buttons results in sending a test-message to Pushover (system test)
- Button C: Pressing this button signals leaving the premises.

9. Config File

Configuration Information is kept in a Json file on the Micro-DS card. Contents of this file is read into a corresponding data-structure in module func_config.ino.

```
//---- Config related definitions and variables ----

struct Config {  /// config struct
  int ThingSpeakChannelNo;          // Channel number ThingSpeak
  int ThingSpeakFieldNo;            // Field number ThingSpeak
  char ThingSpeakWriteAPIKey[20];   // API Key ThingSpeak
  char Title[20];                   // Title für ThingSpeak Anzeige
  char PersonName[20];              // Name der überwachten Person
  int MinutesBetweenUploads;        // Minutes between reporting to ThingSpeak
  char wlanssid_1[20];              // SSID Wlan
  char wlanpw_1[25];                // PW Wlan
  char NTPPool[20];                 // Pool Info NTP Server
  char Timezone_Info[60];           // Time Zone Info
  char Email_1[20];                 // Email Entwickler
  char Email_2[20];                 // currently not used
  int TimeOutLeavingSec;            // Timeout after Press Leaving button
  int MaxActivityCount;             // Max number reported Movements
  int ScreenTimeOutSeconds;         // How many Seconds display is on
  char PushoverUserkey[32];         // User Key Pushover
  char PushoverToken[32];           // User Token Pushover
  char PushoverDevices[30];         // Devices Pushover
  int HoursbetweenNoMovementRep;    // How many hours between no movements
  int EveningReportingHour;         // Time for eveneing reporting
  int EssentialDebug ;
};
```

Timezone Info see here:

https://remotemonitoringsystems.ca/time-zone-abbreviations.php

10. Reporting of Movements

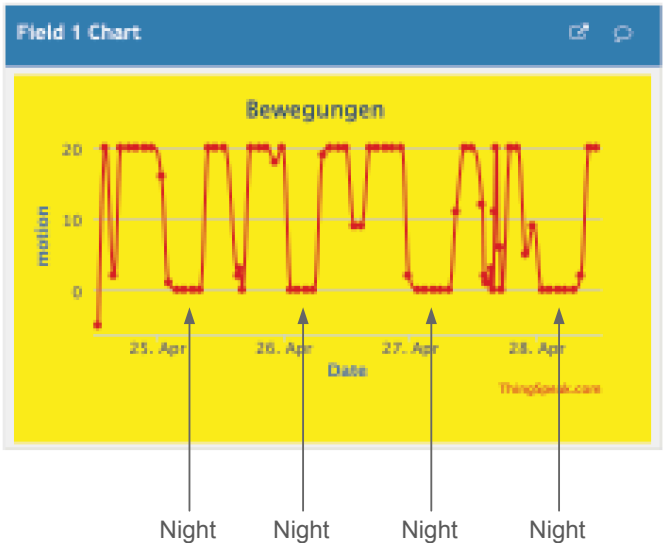
10.1 ThingSpeak Channel

Movement reporting is done using the service ThingSpeak. The reporting intervall is configurable, I figured 90 minutes to be about right (translating to about 18 reportings within a 24 hour period).
Note: the number of reported movements is capped at 20 for one reporting period irrespective of how many are detected.

The test-channel is public and can be found here.

Uncle Dagobert Home Alone

The following graph shows a regular multiday report on Uncle Dagobert who seems to be alive and well. We can see that Dagobert is a sound sleeper - no movements are reported during the nights



Movements as shown on the ThingSpeak Website

10.2 Pushover Messages

Instead of the sending of emails I opted to used **Pushover.net** to send messages to mobile devices. There is regular **evening reporting** and an **alert reporting** if no movements are detected within a configurable length of time. I use an Apple iPhone and the Pushover app - it pings nicely whenever a message is received.
Evening reporting is configurable. It reports the number of movements since last time (the reporting was done the previous day). Evening reporting hour is configurable (zero meaning no reporting).
If there are no movements during a configurable length of time (I use 24 hours) are detected, an alert message is sent: **'Attention: No movements within the last 24 hours'.**

The first version of this project used a Pushover Arduino lib that I modified for my own use. The second Version, however, does not use a library for pushover, the code simply uses the conventions given on the Pushover Website. A certificate für secure connection is used (HTTPS Post Message).

Pushover API

11. Code

The code is available on GitHub.
Home Alone on GitHub.

12. What the code demonstrates

The Home Alone code demonstrates several useful things/concepts:

- The use of multiple tasks on the ESP32
- Use of Semaphores to protect access to global variables

- Use of a modified version of Andreas Spiess Debug Prepocessor directives (to keep production code small)
- Logging to an Micro-SD card
- Use of an Adafruit 1306 OLED display.
- Use of a Json config file on same Micro-SD card
- Use of ThingSpeak service
- Use of Pushover service
- Use of ThingSpeak App on Apple iPhone (also on Android)

ThingSpeak Website

Pushover Website

Andreas Spiess YouTube

13. Notes ThingSpeak

Pushover is a free service with a REST API and a MQTTP API. You have to create an account and also create a channel first. This project uses the REST API, documentation see ThingSpeak Website.
This works well with the exeption that the channel display has no sliding time window - so after about 150 reportings things get squezzed together and that affects readabiliy. You need to clear the channel to improve the readability. Do this either on the website (channel settings) or via the API.
Limits (from the website): Users of the free option will be limited to sending no more than 3 million messages each year to the ThingSpeak service. Users of the free license will also be limited to 4 channels.

14. Notes Pushover

ThingSpeak as used with this project is a free service.
Limits (from website): Each application registered to send Pushover notifications may send 10,000 messages per month for free, where one message is defined as successfully sending a message through our API to one user.

15. Discussion, Conclusion

15.1 About the code itself

The code is not as clean as I hoped it would turn out. But since it works ok and I have other projects in the pipeline this will have to do for now. Also, I now think that the whole thing would work well without the different states. The finite statemachine was nice to work out but the gadget itself would work well without it. As I mentioned before, it is unlikely that the leaving button will be used often.

15.2 About the idea

When uncle Dagobert shuffles around in his appartement and his WLAN is up we have no problem: we can see that he is alive. This, however, is only true if Dagobert does not own a pet.

But what do we really know if no movements are reported to the cloud - this is the tricky part.

- The WLAN might be dead in his appartement for a multitude of reasons unconnected with Dagobert himself.
- ThingSpeak as a company has folded or their computers have a serious problem.
- He might have died in his sleep last night
- While shopping, he might have met this lovely old lady and they ran off together
- Dagobert might have tripped over the power cord and the device is dead
- The ESP32 might have died but Dagobert is still alive and well

We conclude that this device is nice and can help us to some degreee but....
All in all, we are happy we have it in place in uncle Dagobert's appartement.

Peter, April 2024.

end of document