

# Alarm Project

## Part I Alarm Controller

### **Project objective**

To create a home security monitor/alarm using elements of a pre-existing but redundant alarm system. The Alarm Controller should also be capable of triggering a separate security camera wirelessly. The camera is to be part II of this project.

### **Background**

With the demise of old PSTN landline phone system, some older home security systems that relied on that PSTN network to access remote monitoring became redundant. There are several ways to address that situation, but for fun it seemed to me that it was a good candidate for some kind of IoT connected micro controller. Another reason you might go down this path is if you have an existing security system that had a free internet based monitoring, but that has now changed to a subscription model.

### **System design requirements:**

- Be powered entirely from the exiting alarm system power supply.
- Use a mobile phone for all control and monitoring from any where.
- Connect to an internet based IoT service for monitoring, control and communication with other IoT devices.
- Monitor up to four sensors and raise an alarm if a breach of security is detected.
- In the event of an alarm, respond in any or all of the following actions:
  - Sound an alarm
  - Flash a strobe light.
  - Send an alert email.
  - Send a push notification to a mobile phone.
- Be compliant with relevant regulations regarding limitations to duration of siren on time etc.
- Behave in a predictable and compliant way in the event that the internet connection is lost, or a dependent internet service fails.
- Have the ability to connect wirelessly with other IoT devices to add functionality.
- Be capable of receiving firmware updates via Wi-Fi, known as over the air (OTA).

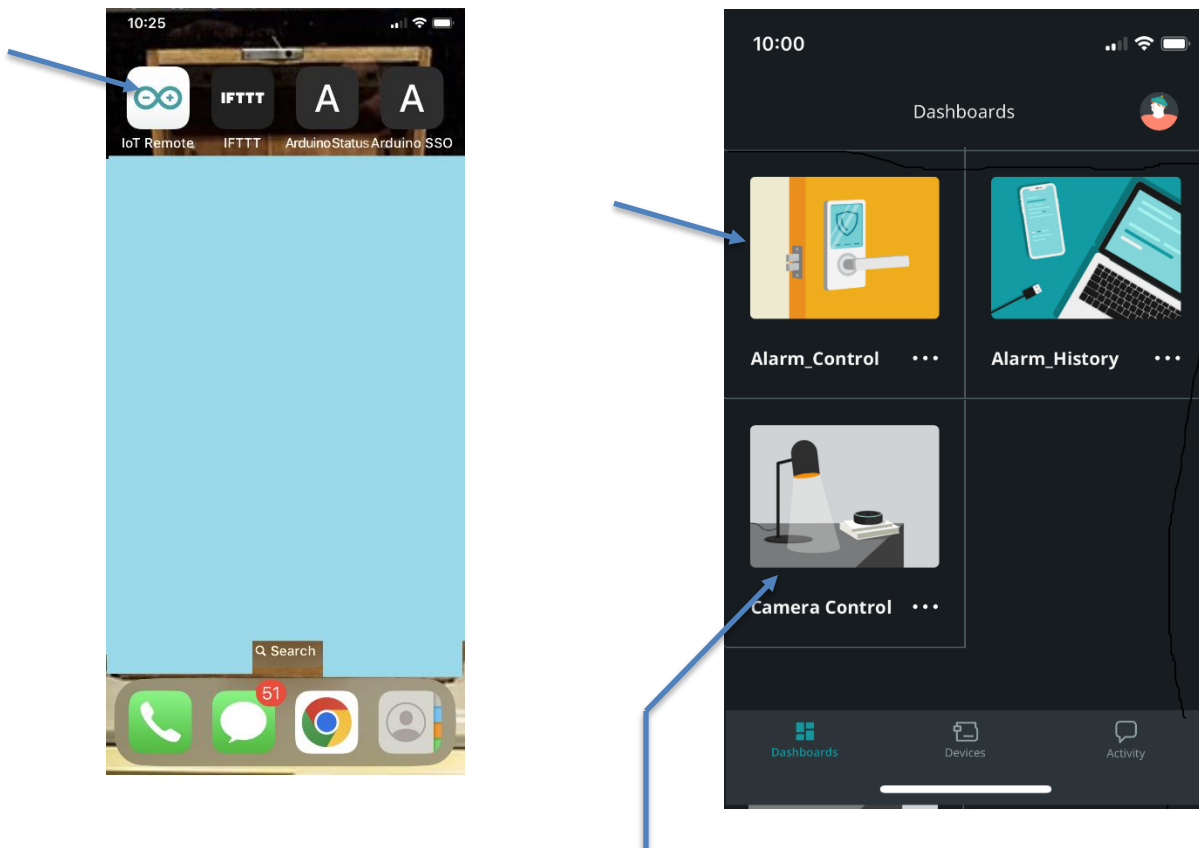
## Finished project operation .

The alarm system is very simple to operate, it can be controlled and monitored with a mobile phone from anywhere as long as there is coverage and internet. There are only two user inputs and seven status indicators. No codes are required to access the app or the alarm panel.

## Alarm system operation

Start by opening the Alarm control panel on your mobile phone.

Open the IoT Remote App , then select the Alarm\_Control dashboard.



A separate but related project. I have built up a an ESP32-CAM video camera triggered wirelessly by this alarm system using a shared ArduinoCloud Variable. It uploads AVI's or stills to google drive. The part II is complete. Code and documentation available.

# Alarm Panel controls

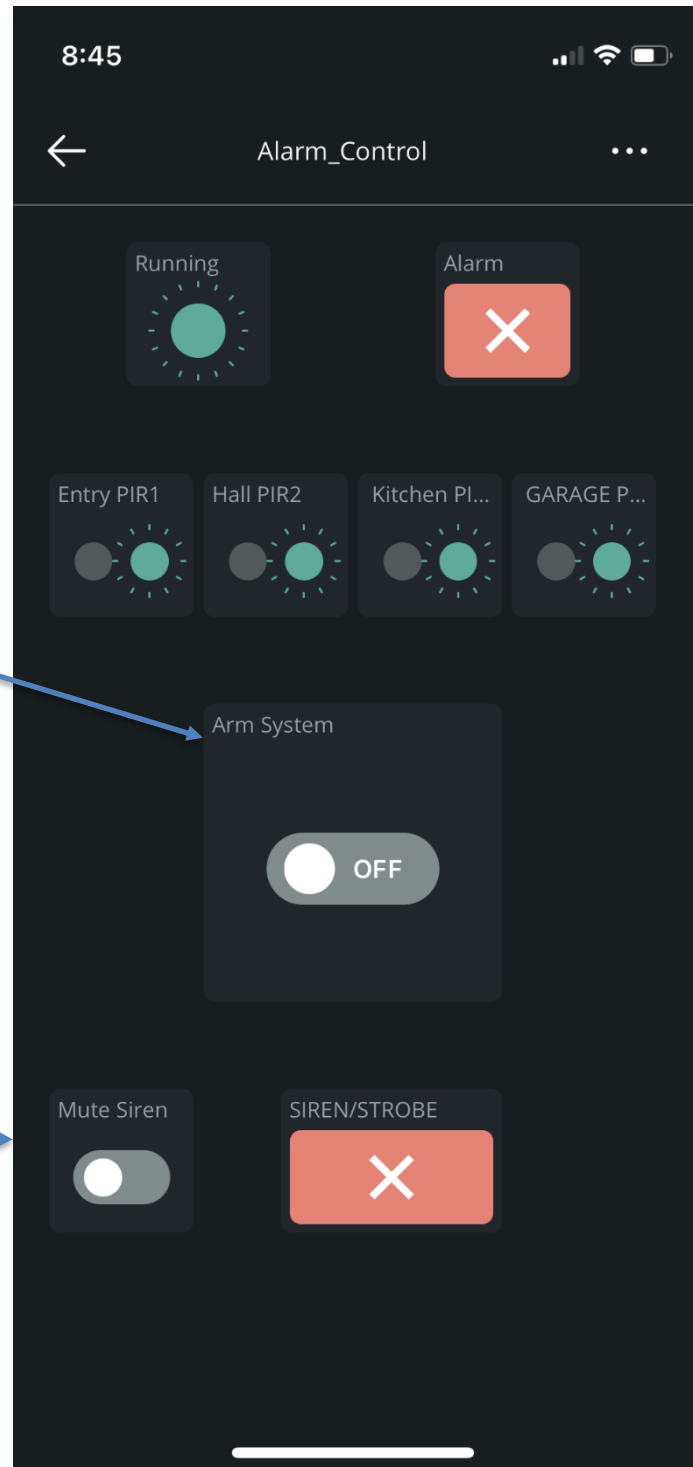
## User control switches

### Arm\_System.

When on, puts the system into alarm monitoring mode, if an alarm is detected the Sirens will sound, the Strobe light will activate. Emails and mobile phone notifications will periodically be sent, alerting the owner. When off the system is idle.

### Mute\_Siren

Stops both sirens from sounding immediately. Other than that the system will still record Alarms send alerts etc if armed.



# Status indicators

## Running

Blinks green once per second, if its not blinking there is a problem. The alarm panel could be powered off or faulty, the local WiFi or internet or other critical service could be down where the alarm system is located.

## Alarm

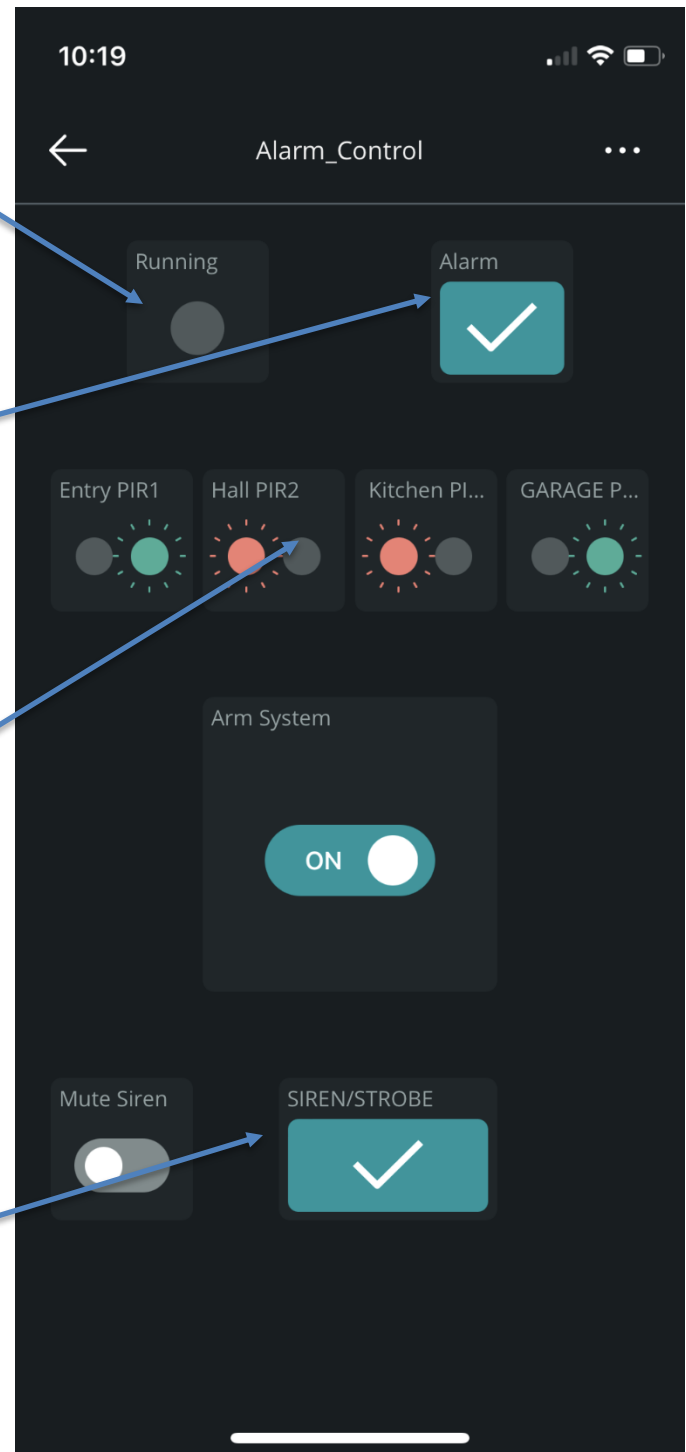
Shows as a red X when not in an Alarm state, shows a green tick when an alarm has been detected, alarm condition. Alarms will hold for 10 minutes assuming there are not further alarm triggers the Alarm state will be cleared the Alarm indicator will go back to a red X, an email will be sent indicating the alarm has cleared. If the Arm\_System switch is turned of during an alarm state, the system will immediately. return to the idle state (Siren's and Strobe off, Alarm indicator to red X)

## PIR's 1-4

When no movement is detected all four indicators will show as green, when movement is detected one or more of the indicators will turn red, if the system is armed this in turn will trigger an alarm state.

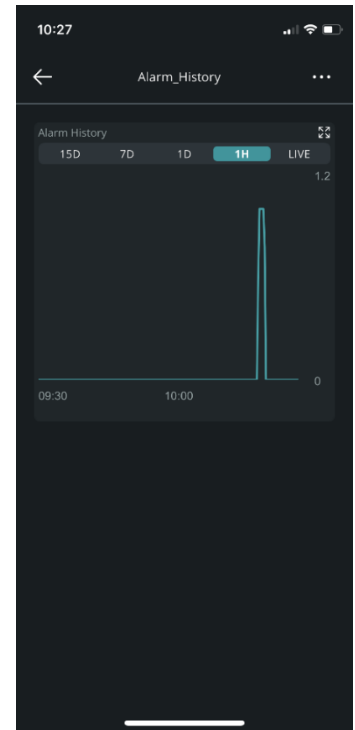
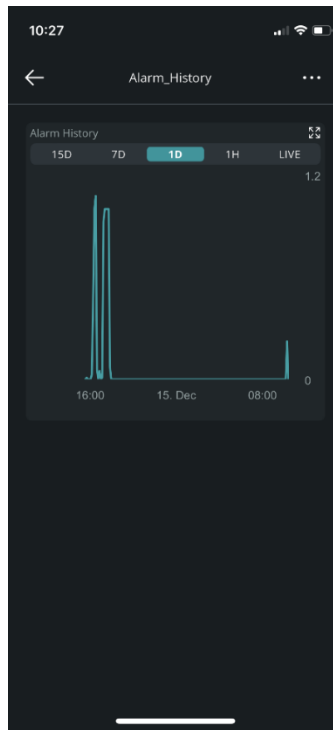
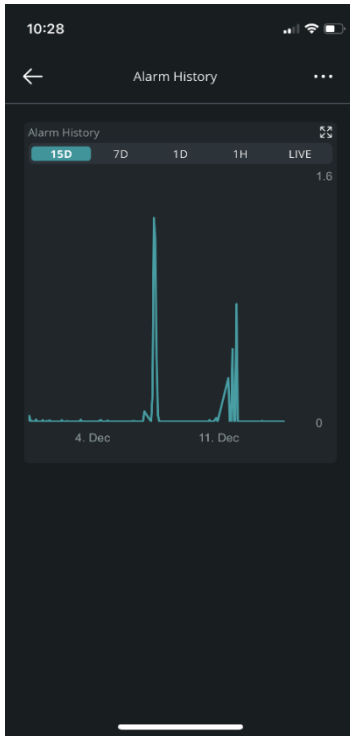
## Siren/Strobe

When showing a red X the Strobe and Siren's are off, if it's a green the Siren's the strobe are logically on, however if Mute\_Siren is set to on, only the strobe will be active.



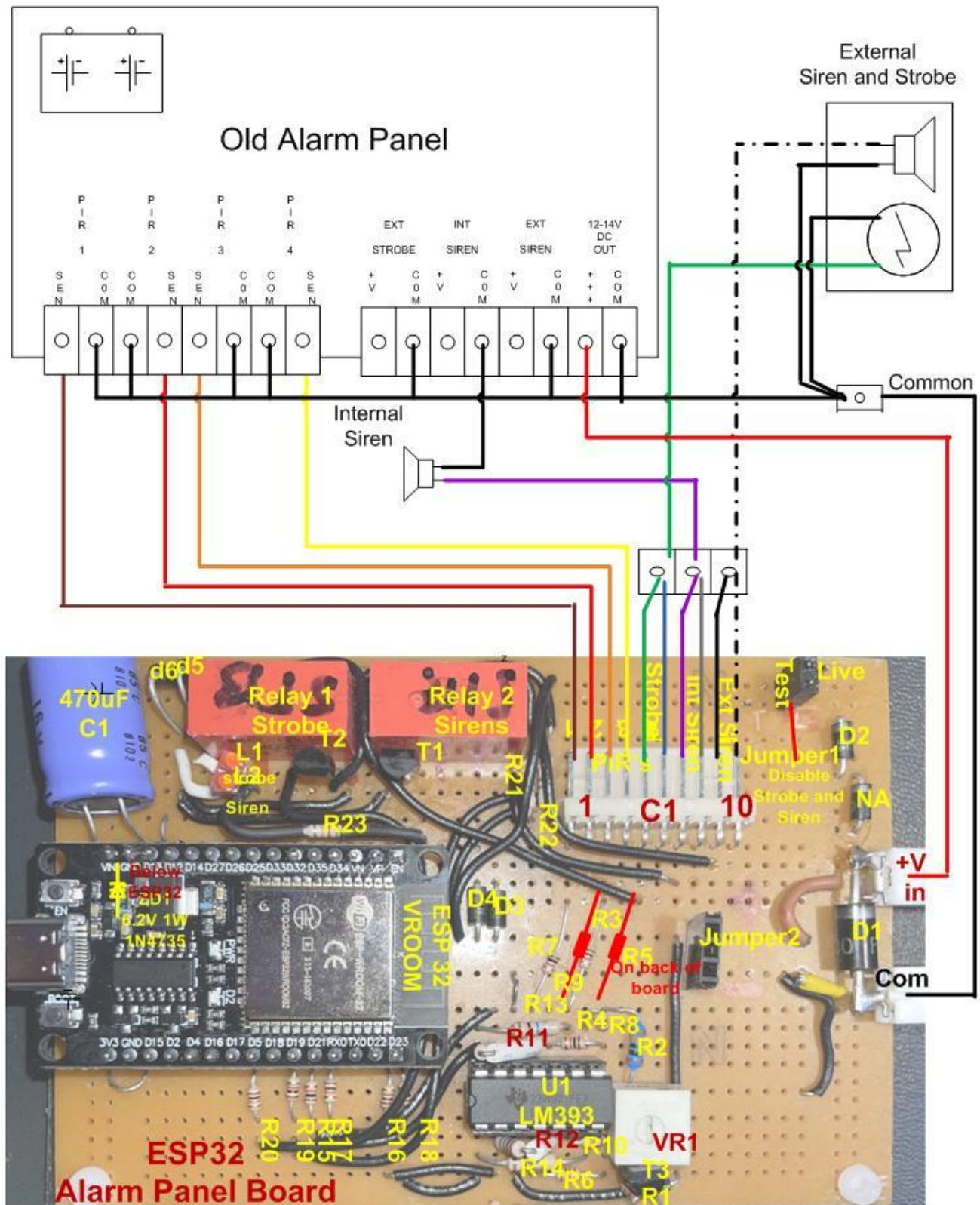
# Alarm History

Alarm history is kept for 15 days and can be viewed over five different periods, Live, 1 Hour, 1, 7 or 15 Days. An alarm count is kept while the system is armed it the count resets to zero each time the alarm system is re-armed. To acces alarm history open the phone app and choose The Alarm\_History dashboard.



# Connecting up the system

## Wiring diagram



### Notes:

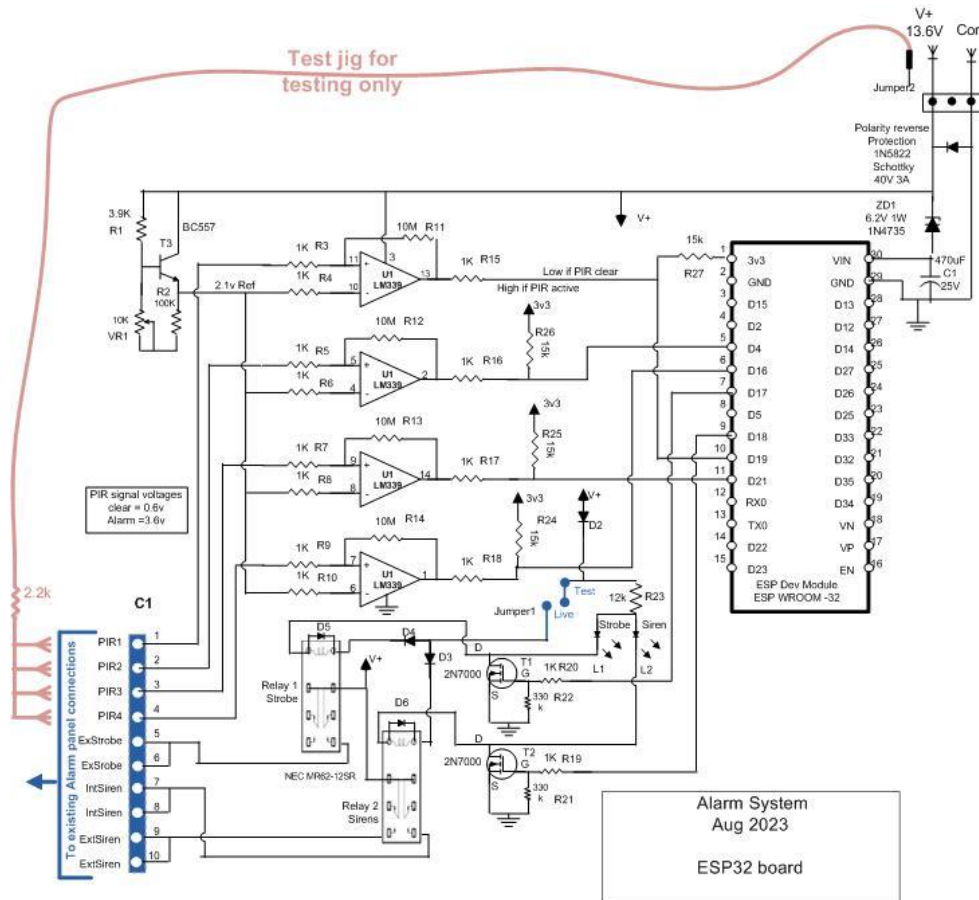
The PIR wiring can remain connected to the old panel terminals. The Alarm board PIR's inputs have a voltage threshold of approx 2.1v, inputs above that voltage are seen as high, below that low. The PIRs can be selected in firmware as logically Alarm high or Alarm low. The positive side of the siren's and the strobe need to be isolated from the old panel.



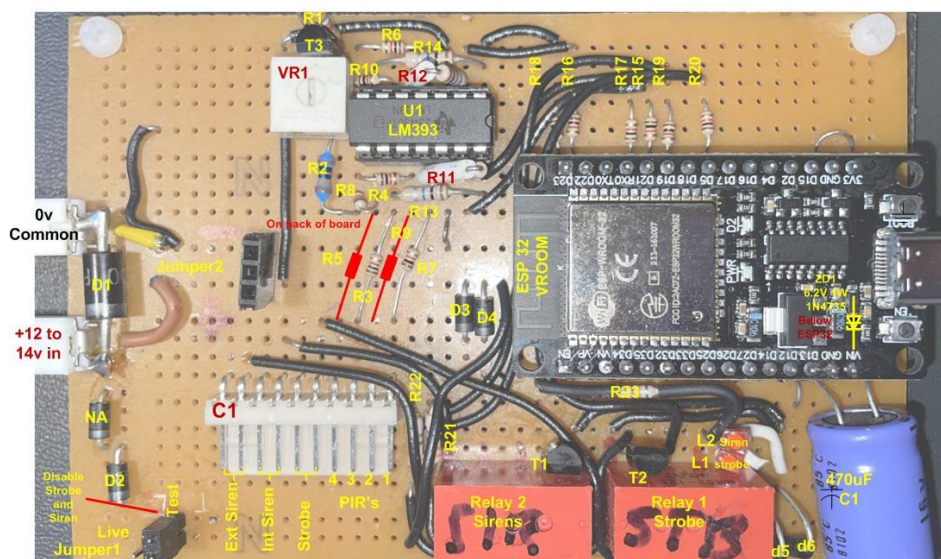
## Circuit description

The ESP-32 uses 6 pins to control all the alarm sensors etc. Four input pins are connected to four PIR's. via an LM339 quad comparator IC which does the buffering and level shifting. The other two pins are outputs that connect to 2N7000 FETs that then drive two DPDT relays, one of which drives the internal and external sirens, the other the strobe light. Two on board LEDs indicate when siren's and or strobe is on. Jumper1 can be moved between the live and test positions to disable the sirens and strobe for test purposes. Jumper2 just exposes +V and Gnd for use with the test jig to simulate alarm/clear conditions during testing.

When choosing the matrix board size for the project you may want to make it a size and shape so that it will fit inside your existing alarm panel box.

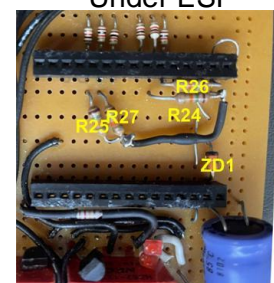


High res versions  
of the cct and  
layout are in  
the package



Board  
Dimensions  
120mm x 78mm

Under ESP



## Software Description

### The project is dependent on two services

#### **ArduinolotCloud (Provides dasboards and IoT communications)**

You will need at least an Arduino base level subscription to implement this project, it currently costs about \$20US per year. (that's the only ongoing cost for this project)

#### **IFTTT (emails and mobile phone push notifications)**

You will need a free subscription which gets you 30 emails/notifications a day, plenty. For sending alert emails and mobile phone notifications.

The source code is commented and its pretty straight forward, so will just give an overview here. A flow chart on the next page give the words below some more context. I am a very basic coder, so I am sure there is a lot of room for improvement by someone more skilled than I.

## Overview

**The system has two operating states armed and idle.**

### **Features common idle or armed state.**

- The dashboard is updated once every second if displays.
- A 'Running' Indicator that blinks every second. If this is not blinking the main unit is powered off or there is a hardware or dependent service fault.
- The status of the 4 PIR's, green normal red activated.
- 'Alarm' status a green tick indicates an alarm condition is active, a Red Cross means no alarm.
- 'Siren/Strobe' will show a green tick if activated if not a Red Cross
- The large 'Arm System' slider switch, when on Arms the system when off returns system to idle state.
- The small 'Mute Siren' slider switch when on inhibits siren sounding but otherwise the system is unaffected.



## **Armed state operation.**

In the event a PIR's enter an 'alarm condition':

- An alarm sequence will start and hold for the alarm hold time.
- The sirens sound (if not muted)
- The Strobe light will flash.
- An alert email and or a mobile phone push notification are sent.

When the alarm hold time expires and the 'alarm condition' has not been cleared:

- The sirens will keep sounding (if not muted) until the 'alarm condition' is cleared or the maximum siren time is expired, whichever is less,

If the 'alarm condition' is cleared:

- The sirens and strobe will turn off.
- The 'alarm condition' will be reset.
- An email is sent.

If the 'Arm System' switched off and an 'alarm condition' is active.

- The sirens. and strobes will turn off.
- An email will be sent.
- The 'alarm condition' will be reset.
- the system enter the 'idle state'

'alarm condition' not active

- The system will enter the 'idle state'

## **Idle State operation.**

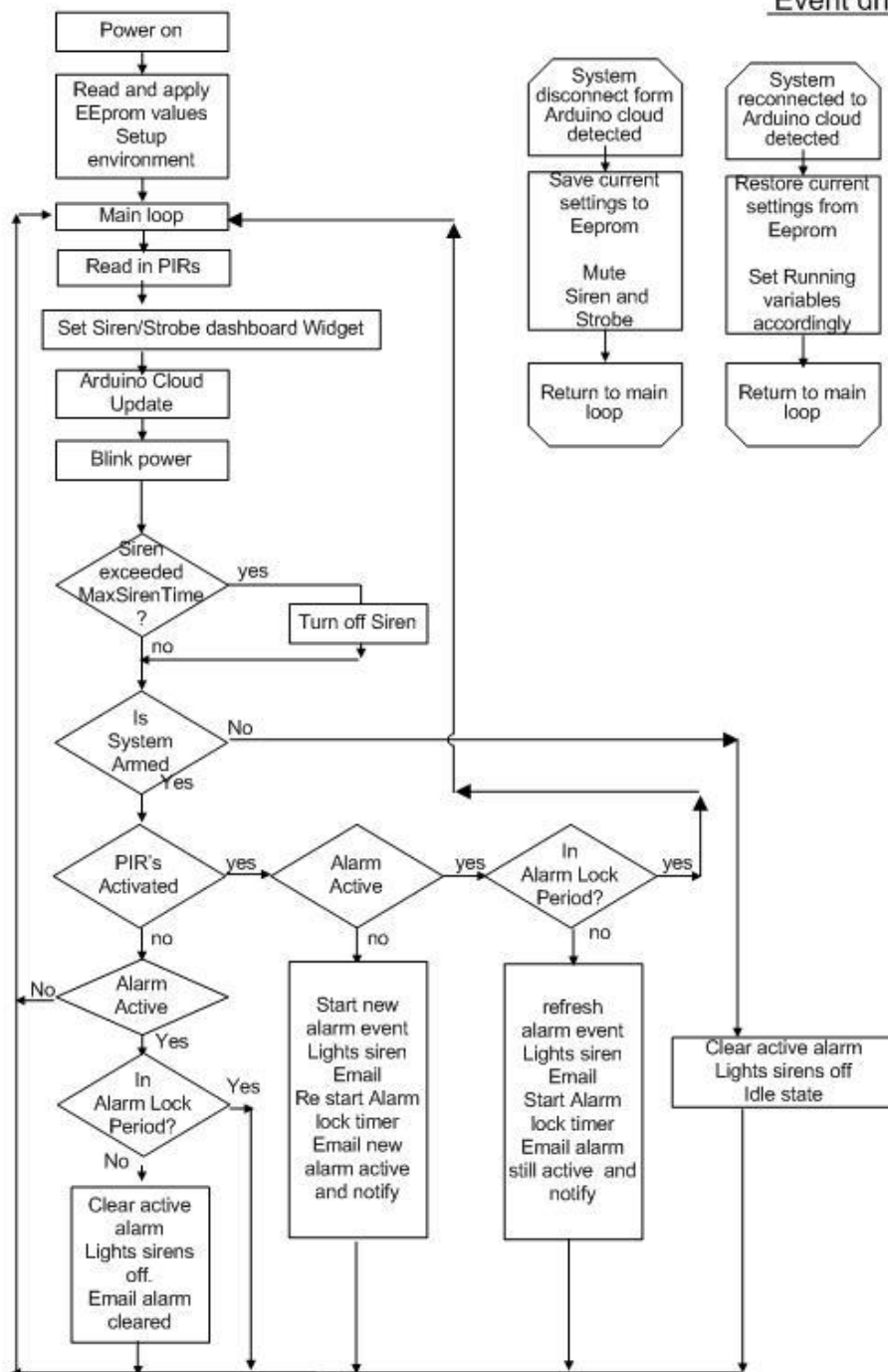
The 'Arm System' slider is turned on:

- The system enters the 'Armed State'

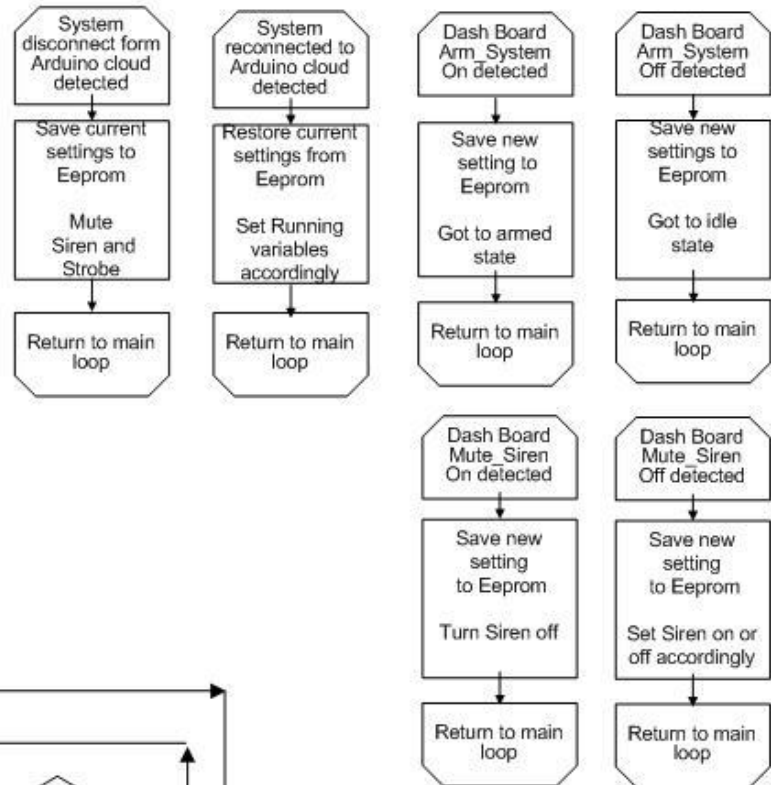
The 'Mute Siren' slider is turned off:

- Siren muting is activated until turned off.

## Software flow diagram



## Asynchronous Event driven function



## Operation

There are effectively three inputs to the alarm system

1. Dash board System\_Aarm switch, that switches alarm from the idle state to the armed state
2. Dash board Mute\_Siren when on disables the siren. Logical state of the siren is not effected.
3. The sensor input (PIRs) these are logically just one input, If the system is in the armed state and any Sensor comes on an alarm will be triggered.

There are several built in software timers that determine system behaviour the durations are controlled by variables in the software at compile time.

1. When an alarm is triggered at timer starts that holds the system in the 'Alarmed State' until the timer expires or the system is disarmed. Variable AE\_Hold sets the duration on the hold period. (default 10 minutes)
2. The duration the siren can sound is limited, for each alarm even, for regulatory and common sense reasons The MaxSireTime sets the duration.(default 180000ms) 3 minutes
3. In addition to timer 2 above there is a lock out period after the siren has stopped sounding until it can sound again. Variable Siren\_Lockout sets the duration. (default 900000ms) 15 minutes.

# Construction and programming

## Prerequisites

- An existing electronic alarm system that includes sensors (PIR's), siren, strobe light(optional), a battery backed up dc power supply.
- An internet connected Wi-Fi service accessible from the location of the new system.(probably the location of the existing system)
- A Windows 10 or later computer administrative user access.
- An Android or IOS mobile phone, with Arduino IOT remote app and IFTTT app.

## Hardware

The size and shape requirement of the hardware will likely be different depending on the space available to fit it into an existing alarm panel box. My hardware is just a single board wired point to point on matrix prototype board. There are no hardware build details, just circuit diagram and layouts that I used, I expect if anyone decided to build this, they will do so in a form that suits them.

## Parts List

Item	Description	QTY	Jaycar#	\$Item	\$Tot
1.	ESP32 Development Board TYPE-C USB CH340C WiFi+Bluetooth Ultra-Low Power Dual Core ESP32-DevKitC-32 ESP-WROOM-32 Expansion Board	1	AliExpress	\$7.00	\$7.00
2.	LM339 Quad comparator IC	1	ZL3339	\$2.25	\$2.25
3.	Relay DPDT 12V 2Amp NEC MR62-12SR (Jaycar Equiv listed)	2	SY4062	\$6.00	\$12.00
4.	Resistor 1K .5W	14		\$0.15	\$2.10
5.	Resistor 10M .5W	4		\$0.15	\$0.60
6.	Resistor 330K .5W	2		\$0.15	\$0.30
7.	Resistor 100K .5W	1		\$0.15	\$0.15
8.	Resistor 3.9K .5W	1		\$0.15	\$0.15
9.	Resistor 100K .5W	1		\$0.15	\$0.15
10.	Resistor 2.2K .5W	1		\$0.15	\$0.15
11.	Trimpot 10K 25t	1	RT4650	\$2.95	\$2.95
12.	Capacitor Electro 470uF 25v	1	RE6195	\$0.75	\$0.75
13.	Diode 1N4001	5	ZR1004	\$0.25	\$1.25
14.	Diode 1N5822 Schottky 40V 3A	1	ZR1048	\$1.45	\$0.15
15.	Zenner 6.2V 1W 1N4735	1	ZR1405	\$0.95	\$0.95
16.	10 way header socket	1	HM3410	\$1.75	\$1.75
17.	10 way header plug	1	HM3430	\$1.45	\$1.45
18.	spade connector male	2			\$1.00
19.	spade connector female	2			\$1.00
20.	Transistor FET 2N7000	2	ZT2400	\$1.25	\$2.50
21.	Transistor BC557	1	ZT2164	\$1.95	\$1.95
22.	ProtoType BOARD	1	HP9552	\$8.95	\$8.95
23.	IC Socket 30pin Make from 40 pin)	1	PI6508	\$1.25	\$1.25
24.	LED red 2mm	2	ZD0040	\$0.75	\$1.50
25.	Three position screw type terminal block (siren/strobe hook up)	1		\$1.00	\$1.00

Where known I have included Jaycar stock numbers

Total ~\$46.5

Other: Sundry wire solder heat shrink etc

Tools: soldering iron, digital volt meter, wire cutters, screw drivers etc

# Software setup

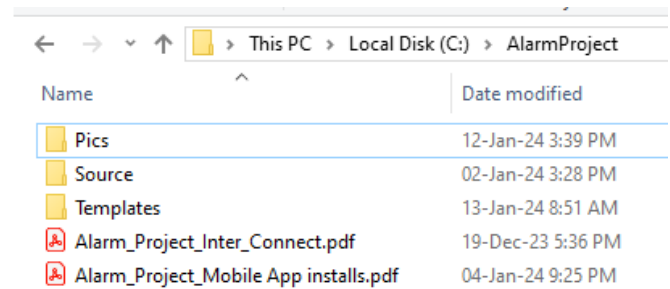
## Summary steps

- Copy the project package to your computer.
- Setup Arduino Cloud account (if required)
- Manually add the esp32 dev module to your cloud environment
- Download and setup Arduino cloud CLI (if required)
- Use CLI templates to create the ALARM\_THING and two dashboards.
- Link device ALARM\_DEV to thing ALARM\_THING
- Set up an ifttt account (if required) and create two applets.
- Configure software user specific items like Wi-Fi etc.
- Configure Mobile Phone apps

## Detail

### #### Copy the project package to your computer. ####

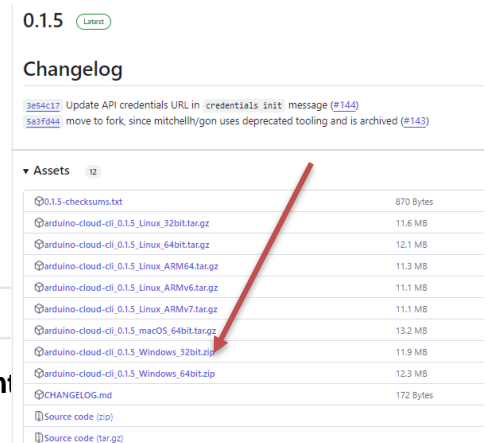
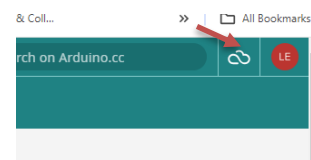
1. Go to <https://github.com/lewlev/AlarmProject/blob/main/Pack1.zip> and download Pack1.Zip this included all software source code files, templates for the IOT Thing and Dashboards, CLI executable higher res circuit and layouts some more detailed set up guides for IFTTT etc.
2. Create a folder **C:\AlarmProject** (Elevated permission may be required) Extract [Pack1.zip](#) to it.  
You should see Source and Templates folders along with other files and folders ( it is important to use the exact folder name and Location as the rest of this guide uses links to that Folder)  
You may want to add C:\AlarmProject to Quick Access



Name	Date modified
Pics	12-Jan-24 3:39 PM
Source	02-Jan-24 3:28 PM
Templates	13-Jan-24 8:51 AM
Alarm_Project_Inter_Connect.pdf	19-Dec-23 5:36 PM
Alarm_Project_Mobile App installs.pdf	04-Jan-24 9:25 PM

### #### Setup Arduino Cloud account (if required) ####

3. Setup Arduino cloud account if you don't have one already  
Ref <https://support.arduino.cc/hc/en-us/articles/360016416280-Sign-up-for-an-Arduino-Cloud-plan>  
Choose at least 'Entry' plan currently \$US1.99 per month
4. Ctrl Click [Arduino cloud](#) and Login if you are not already.  
Click the cloud button (top right) to go to the home page
5. Skip this if you extracted the Pack1.zip file the CLI is included.  
Click on this link <https://github.com/arduino/arduino-cloud-cli/releases> to download the zip file and extract directly into [C:\AlarmProject\templates](#). (This assumes windows 10 or 11 OS)



0.1.5 Latest

### Changelog

- 3e54c17 Update API credentials URL in 'credentials' init message (#144)
- 5a3f644 move to fork, since mitchellh/gon uses deprecated tooling and is archived (#143)

### Assets 12

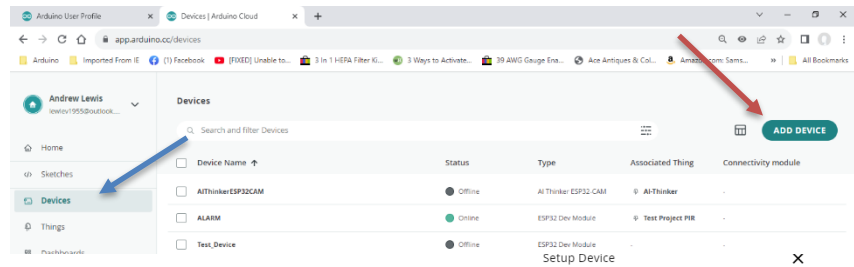
0.1.5-checksums.txt	870 Bytes
arduino-cloud-cli_0.1.5_Linux_32bit.tar.gz	11.6 MB
arduino-cloud-cli_0.1.5_Linux_64bit.tar.gz	12.1 MB
arduino-cloud-cli_0.1.5_Linux_ARM64.tar.gz	11.3 MB
arduino-cloud-cli_0.1.5_Linux_ARMv6.tar.gz	11.1 MB
arduino-cloud-cli_0.1.5_Linux_ARMv7.tar.gz	11.1 MB
arduino-cloud-cli_0.1.5_macOS_64bit.tar.gz	13.2 MB
arduino-cloud-cli_0.1.5_Windows_32bit.zip	11.9 MB
arduino-cloud-cli_0.1.5_Windows_64bit.zip	12.3 MB
CHANGELOG.md	172 Bytes
Source code (zip)	
Source code (tar.gz)	

### #### Manually add the esp32 dev module to your cloud environment

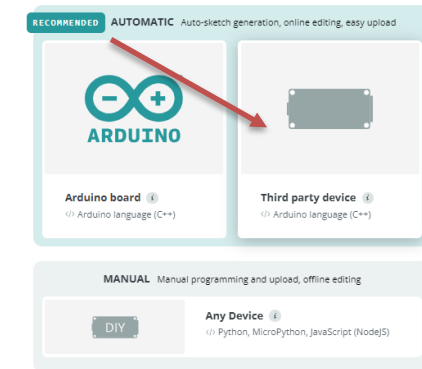
6. Add the ESP32 as a device into Arduino CCloud

- Select Devices in the left side pane.

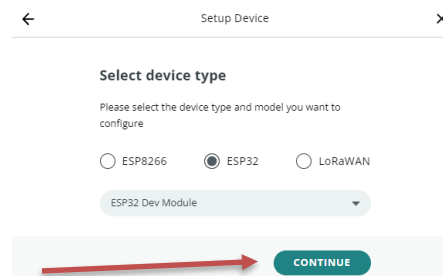
- Select ADD DEVICE



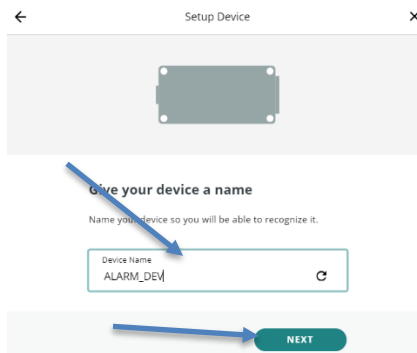
- Choose third party device.



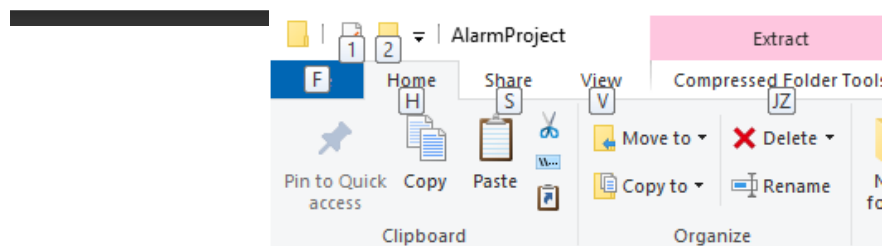
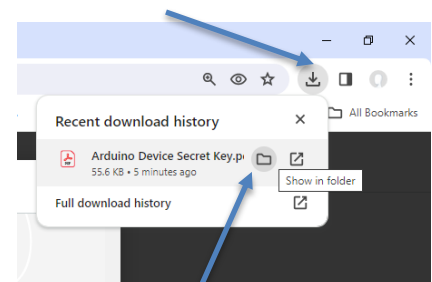
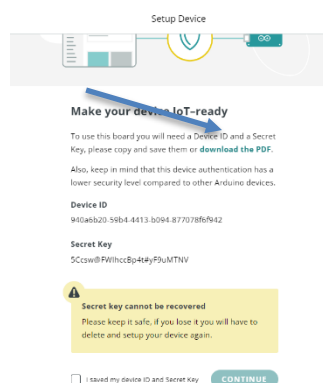
- Select device type and model
- **ESP32      ESP32 DEV MODULE**
- Click on Continue.



- Type in **ALARM\_DEV** as the Device Name
- Click Next

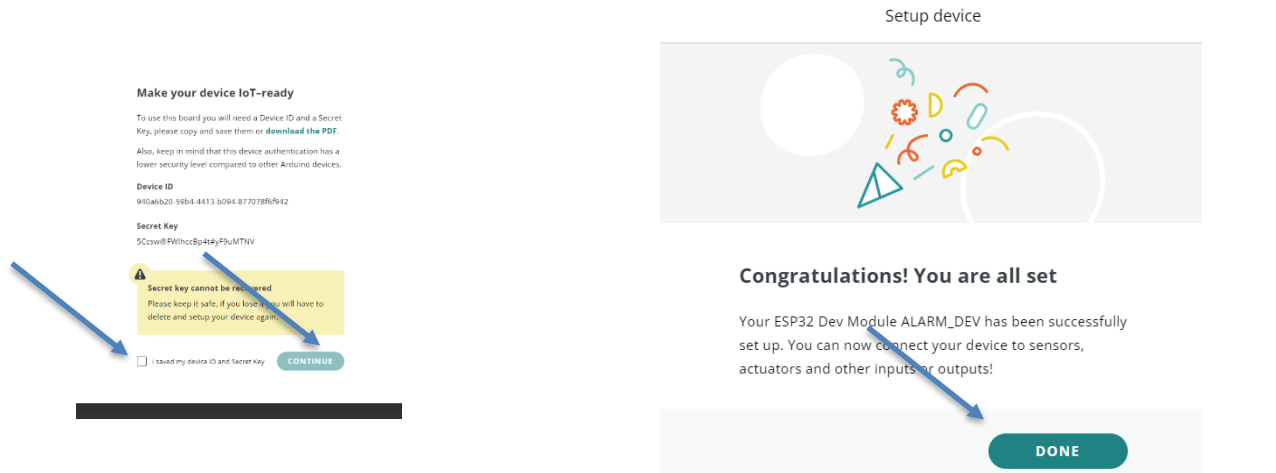


- A dialogue will come up with the device ID and Secret Key
- Click 'Download the PDF'
- Click the downloads button in the top right of the browser
- Then hover over the file and click on the folder button



- Right click on the pdf file and rename it
  - ALARM\_DEV\_KEYS.pdf
  - Copy and paste to the project folder
- Save it in the project folder <C:\AlarmProject>

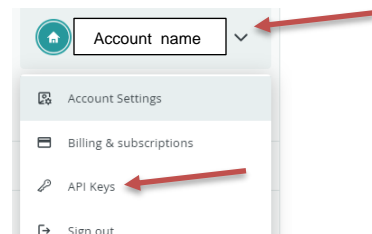
- Back in the browser
- Click on the 'I saved my Device ID and Secret' check box then Continue and done.



#### #### Setup Arduino cloud CLI (if required) ####

- The task involves using the ArduinoCloud CLI if you have not used this before it needs to be authenticated with your Arduino Cloud Account if you have already done that skip to step 11.

- Go to your Aduino Cloud home page
- In the top left corner, you will find your account name



- Click on the down arrow beside your name select API Keys



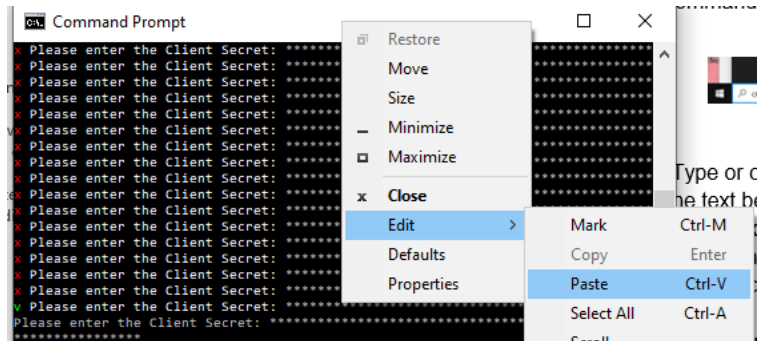
- Click on the Create API Key button
- A new key will be created.
- You will be prompted to save a pdf file with the, new keys details.
- Save it to <C:\AlarmProject> say call it CloudCLI\_API\_Keys.pdf

- Open a command window by typing 'cmd' into the windows search bar.





- Type or copy and past the text below shown in **"bold text"** at theCommand prompt.  
(don't include the "")  
Note to paste right click on the white Bar at top cmd window the edit Paste Ctrl V does not seem to work.



- "cd\AlarmProject\Templates"**

Enter the command below, when prompted enter the **ClientID** and **Client Secret** Obtained in step 7.

- "arduino-cloud-cli credentials init"**

To obtain your API credentials visit <https://create.arduino.cc/iot/integrations>

Please enter the Client ID **^&%YUYUIYT\*&^\*&^\*(^\*&^**

Please enter the Client Secret:

**\*\*\*\*\***

v Please enter the Organization ID - if any - Leave empty otherwise:

Credentials file successfully initialized at:

D:\Users\User\AppData\Local\Arduino15\arduino-cloud-credentials.yaml

- If successful you should a response like the one above.

## 9. Open ArduinoCloud CLI on your computer

(If you just added a new set of keys skip to step 11)

(computer must be running Windows 10 at a minimum)

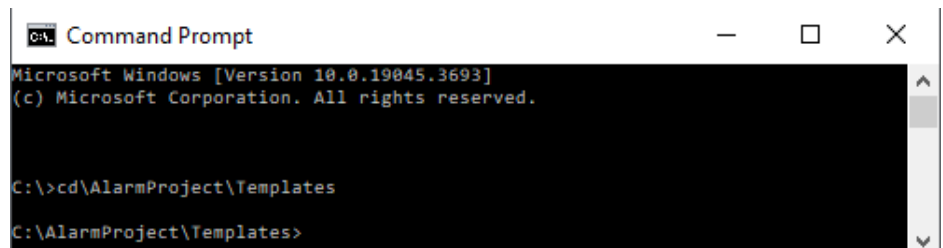
the CLI executable will be in the C:\AlarmProject \Templates folder.

Ref Cli guide docs <https://docs.arduino.cc/arduino-cloud/arduino-cloud-cli/getting-started>

- If you don't have one open, open a command window by typing 'cmd' into the windows search bar



- Type or copy and past the text below shown in **"bold text"** to the Command prompt.  
(don't include the "")  
followed by the Enter key



**"cd\AlarmProject\Templates"**

C:\AlarmProject\Templates>

#### Use CLI templates to create the ALARM\_THING and two dashboards ####

10. Create the **ALARM\_THING** thing from template.

- Type or copy and paste the text below shown in "**bold text**" to the Command prompt. (don't include the "") followed by the Enter key
- **"arduino-cloud-cli thing create --name ALARM\_THING --template ALARM\_THING\_Temp.yaml name: ALARM\_THING"**

name: ALARM\_THING

id: c5713bfc-00ad-4a9b-b75a-6564d207606b

device id:

variables: AlarmHist, AlarmAct, ARMED, BlinkPower, MUTESIREN, PIR1, PIR2, PIR3, PIR4, SIRENsTROBEWidget

If you get a response like the one above the new ALARM\_THING was created OK. The ID (Yellow highlight) will of course be different and unique to the new thing.

Take note of the thing id above it is needed to create the dashboards via the CLI in next steps. (tip, to copy txt from the command prompt, right click the top white command prompt bar, select edit select mark, use the mouse to select txt, it will highlight in white, hit the return key. The text can now be pasted anywhere with ctrl P)

11. Create the **Alarm\_Control** dashboard from template

- Please copy just the ID in yellow and paste it into the command window. Don't hit return or any other key except the left arrow to move the cursor back to the beginning of command prompt, like this  
C:\AlarmProject\Templates > c5713bfc-00ad-4a9b-b75a-6564d207606b
- Now type or copy and paste the section of the command below in bold (don't copy the quotes) followed by the Enter key
- **"arduino-cloud-cli dashboard create --name Alarm\_Control --template Alarm\_Control\_Temp.yaml --override ALARM\_THING=" c5713bfc-00ad-4a9b-b75a-6564d207606b**

name: Alarm\_Control

id: d5355171-c4fe-4818-841f-42df411738cc

updated at: 2023-12-13 00:25:20.7042 +0000 UTC

widgets: Hall PIR2, SIREN/STROBE, Alarm, Mute Siren, Entry PIR1, GARAGE PIR4, Kitchen PIR3, Running, Arm System

## 12. Create the **Alarm\_History** dashboard from template

Repeat step 11 again for the Alarm\_History Dashboard

**“arduino-cloud-cli dashboard create --name Alarm\_History --template Alarm\_History\_Temp.yaml --override ALARM\_THING=”****c5713bfc-00ad-4a9b-b75a-6564d207606b****”**

name: Alarm\_History

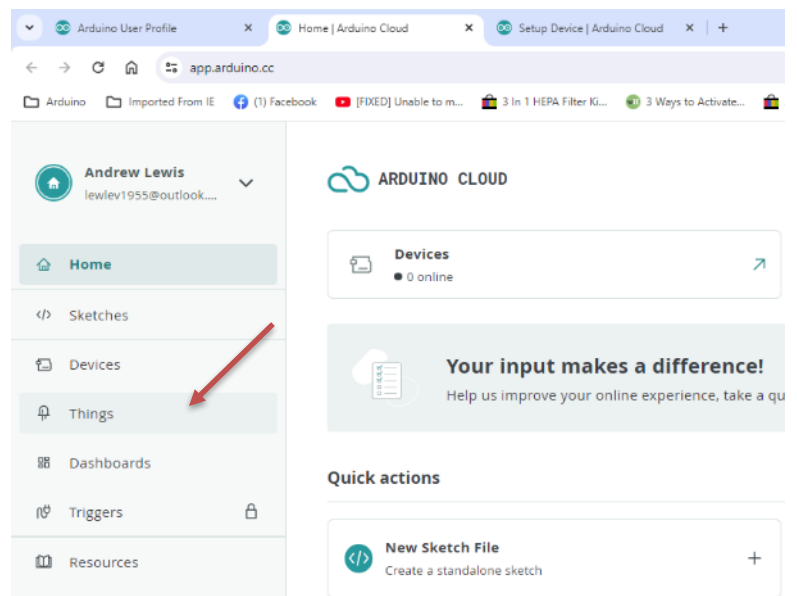
id: f3bab9f9-3b6d-42b2-9540-1eabdb7a4e54

updated\_at: 2023-12-13 03:06:18.990525 +0000 UTC

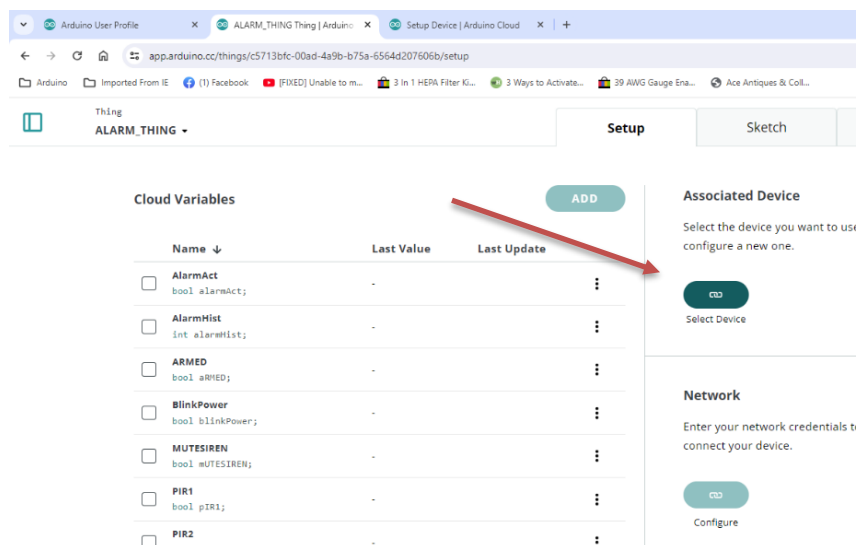
widgets: Alarm History

## 13. Associate ALARM\_DEV and ALARM thing

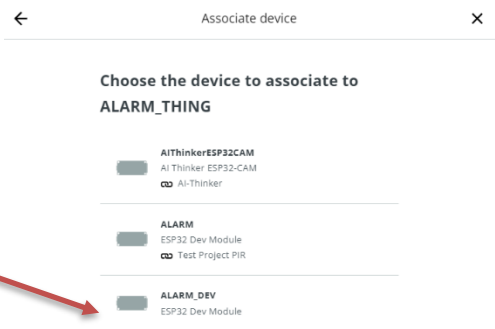
From the Arduino Cloud home page choose Things



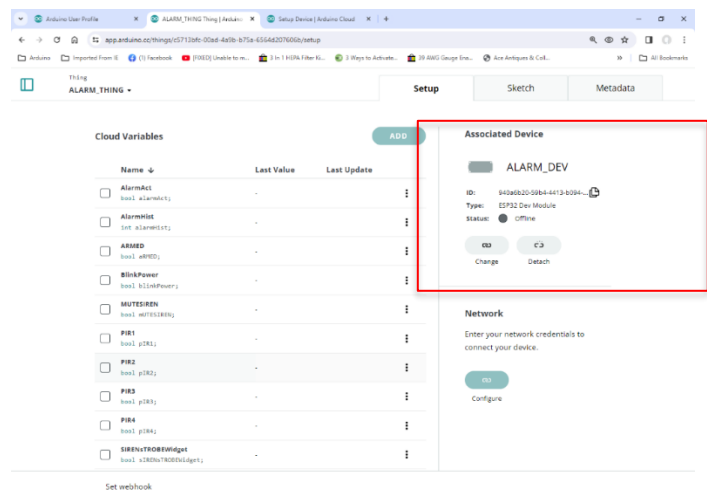
Then choose ALARM\_THING  
Then click on 'Select Device'



Then choose ALARM\_DEVICE



Done



That completes all the Arduino Cloud items needed

#### Set up an ifttt account (if required) and create two applets. #####

14. Set up ifttt service to generate email and mobile phone push notifications.

Open a browser and got <https://ifttt.com/>  
Set up a free account using the email address that Alarm notification emails are to go to  
Create two applets as detailed below  
Retrieve connection key  
A detailed set of screen shots to guide you through the steps required is in the project folder  
<C:\AlarmProject\IftttSetup.pdf> if needed.

Setting	Applet 1	Applet 2
Name	Alarm_Event	Alarm_Nofication
If This Service	Webhooks	Webhooks
Trigger	Web requ with json load	Web requ with json load
Then that Service	Email	Notifications
Action	Send an Email	Send notification from the IFTTT app.
Target	Your ifttt account email	mobile phone # entered during create.

15. Test the applets by pasting the URLs below (with your connection key) into a browser  
An email and a mobile phone notification should be generated.

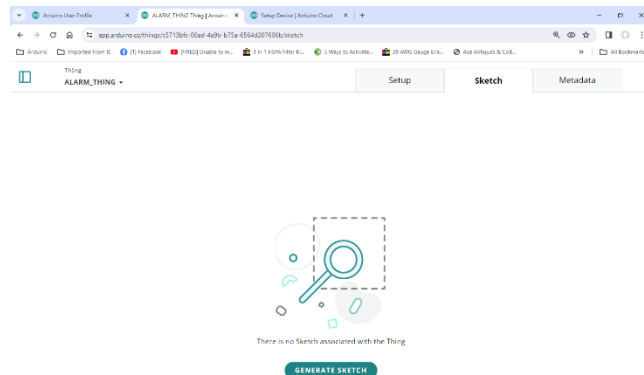
[https://maker.ifttt.com/trigger/Alarm\\_Event/with/key/your\\_key](https://maker.ifttt.com/trigger/Alarm_Event/with/key/your_key)

[https://maker.ifttt.com/trigger/Alarm\\_Notification/with/key/](https://maker.ifttt.com/trigger/Alarm_Notification/with/key/)

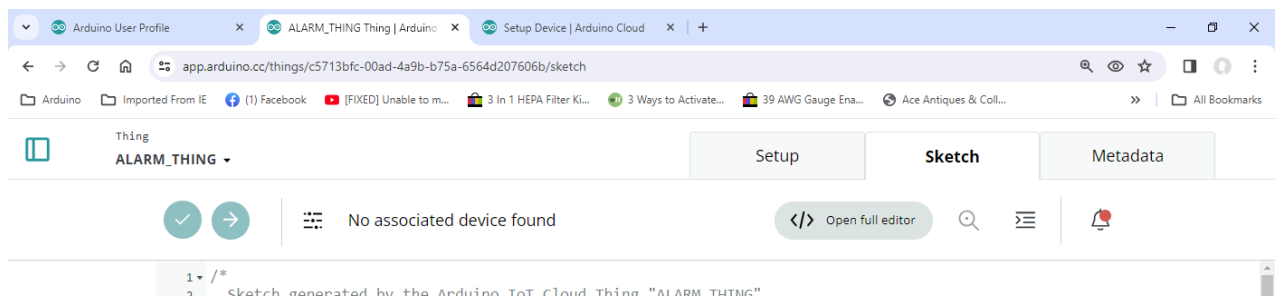
#### #### Configure software user specific items like Wi-Fi etc. ####

16. Load the source files into your cloud environment, there is a little bit of file manipulation to do Here because of the project and file name conventions work.

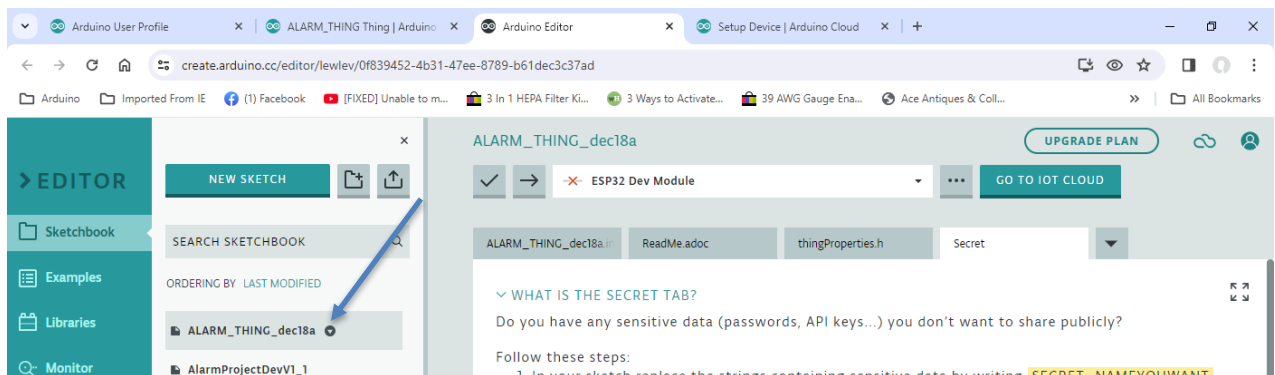
- Open [Arduino Cloud](#) home page login if required,
- Open Things
- Open thing ALARM\_THING then click on the sketch tab
- You should see the dialogue below click on “Generate Sketch”



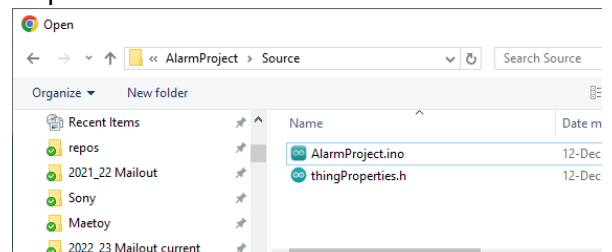
- The sketch will open in the basic editor



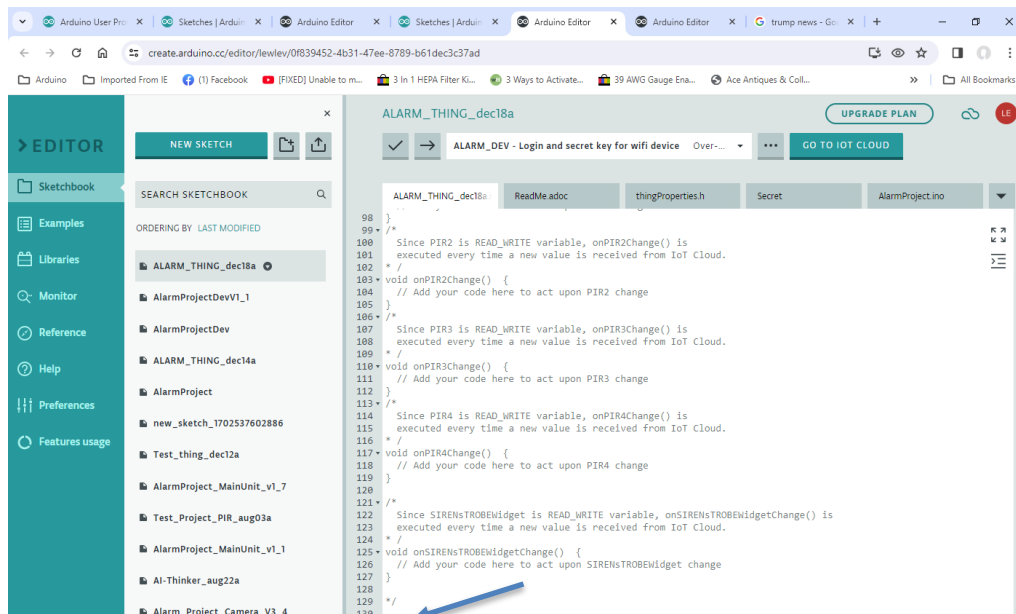
- Click on Open full editor  
The web editor will open with 4 tabs  
ALARM\_THINGxxxx.ino, ReadMe, thingProperties.h and Secret  
We will effectively replace ALARM\_THINGxxxx.ino, with the AlarmProject code.



- The Web editor will have ALARM\_THINGxxxx.ino at the top of the sketch list in the left pane.
- Click on the down arrow and choose
- ‘Import File into Sketch’



- Browse to the project folder and
- open the Source folder C:\AlarmProject\Source
- Select AlarmProject.ino and click open
- You should now have five file tabs with AlarmProject.ino added
- Open **ALARM\_THINGxxxx.ino** in the web editor
- If you want to keep auto generated comments etc follow the next steps. If not just **Select All** and **delete**.
  - right click and choose **'Find and Replace'**
  - Type **\*/** into the find box and **\*/** into the replace box click on the **ALL** button
  - Move the cursor past the last of the text in the file, then type in **\*/** to close the last comment. This will effectively comment out the auto generated section of code that will be replaced by the incoming .ino file



- Open the AlarmThing.ino tab in the web editor, right click and choose **'Select All'** right click again and choose **'Copy'**
- Open the **ALARM\_THINGxxxx.ino** tab again and place the cursor below any existing text. Right click and choose **'Paste'**
- Now you should have the code from AlarmProject.ino in ALARM\_THINGxxxx.ino.
- Select the **AlarmThing.ino** tab then click on the little down arrow tab and select **delete AlarmThing.ino**.

17. We now need to match the template sketch and Thing security keys etc to your environment.

click on the cloud button (top right) click on Sketches.

### Secrets.h

Open the Secrets tab and enter your Wi-Fi SSID and password and the **Secret Key** from the **ALARM\_DEV**, it will be in the pdf file **ALARM\_DEV\_KEYS.pdf** file saved earlier the project folder.

Note this will populate WiFi credential and Secret ID in ALARM\_THING as well.



## AlarmProject.ino

Search for "// ##### Enter ifttt key here #####"

On the next line enter or paste between the "" your ifttt key from the account setup earlier

- `const char *key = "Your ifttt key";`

## #### Configure Mobile Phone apps. ####

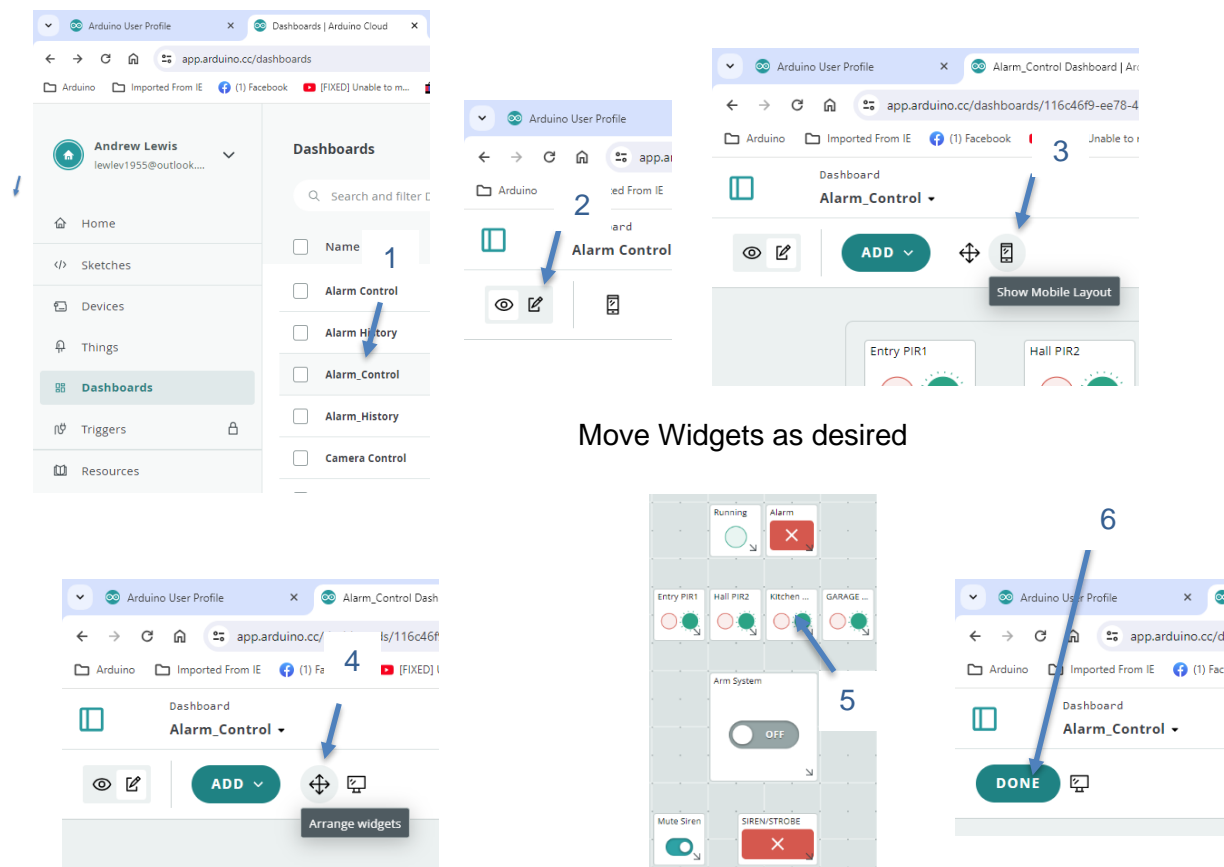
### 18. Goto the app store

- Install the Arduino IOT remote app use the Arduino Cloud Account credentials.
- Install the IFTTT app use your IFTTT account credentials.

A brief guide is included in the pack <C:\AlarmProject\Alarm Project Mobile App installs.pdf> if required.

### 19. Fix dashboard widget layout

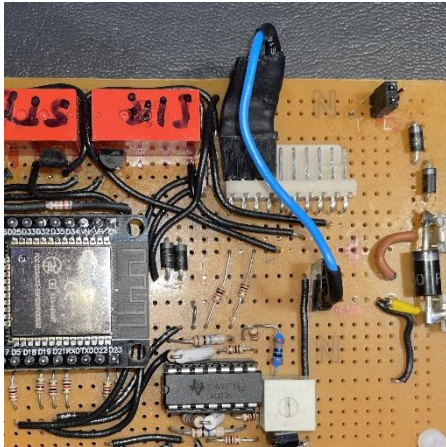
Note for reasons unknown to me, the Dashboards created through the CLI, don't automatically layout the widgets correctly in the Mobile view. So it is necessary to adjust Mobile layout. Open ArduinoCloud follow the steps below to edit the layout



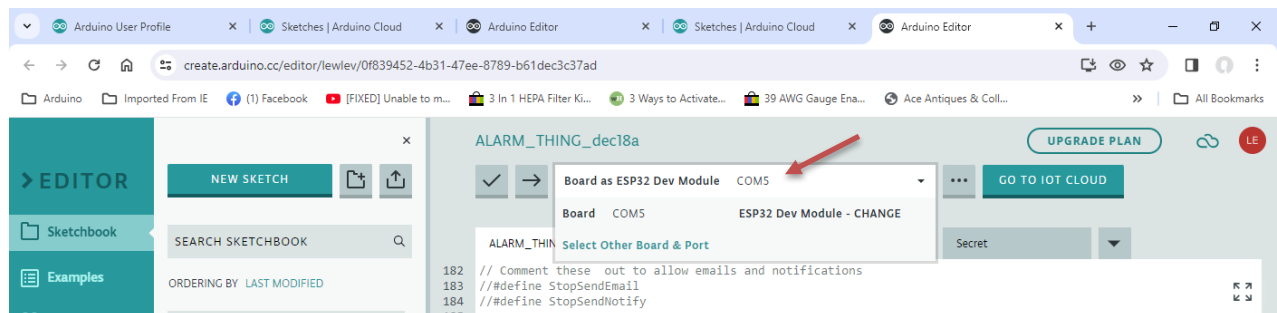
20. Thats it.

## 21. Bench Testing

- You may like to make up a test jig see the circuit diagram for details just a resistor a 4 pin piece of header socket and some wire with a pin to connect to +v or ground to simulate alarm on/off PIR conditions. The blue wire in the picture is part the jig.

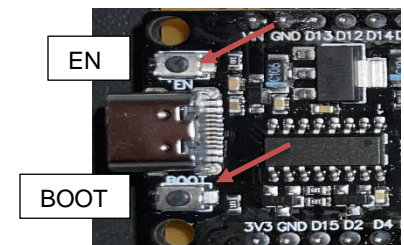


- Connect your board to a 12v power supply and the ESP32 to your PC via USB conditions.
- Open the Arduino Cloud Web Editor then the ALARM\_THINGxxxx.ino file
- Uncomment the Debug definition in the ino file so we get some definition serial monitoring
- Check /set the device connection to Board COMx ESP32 Dev Module



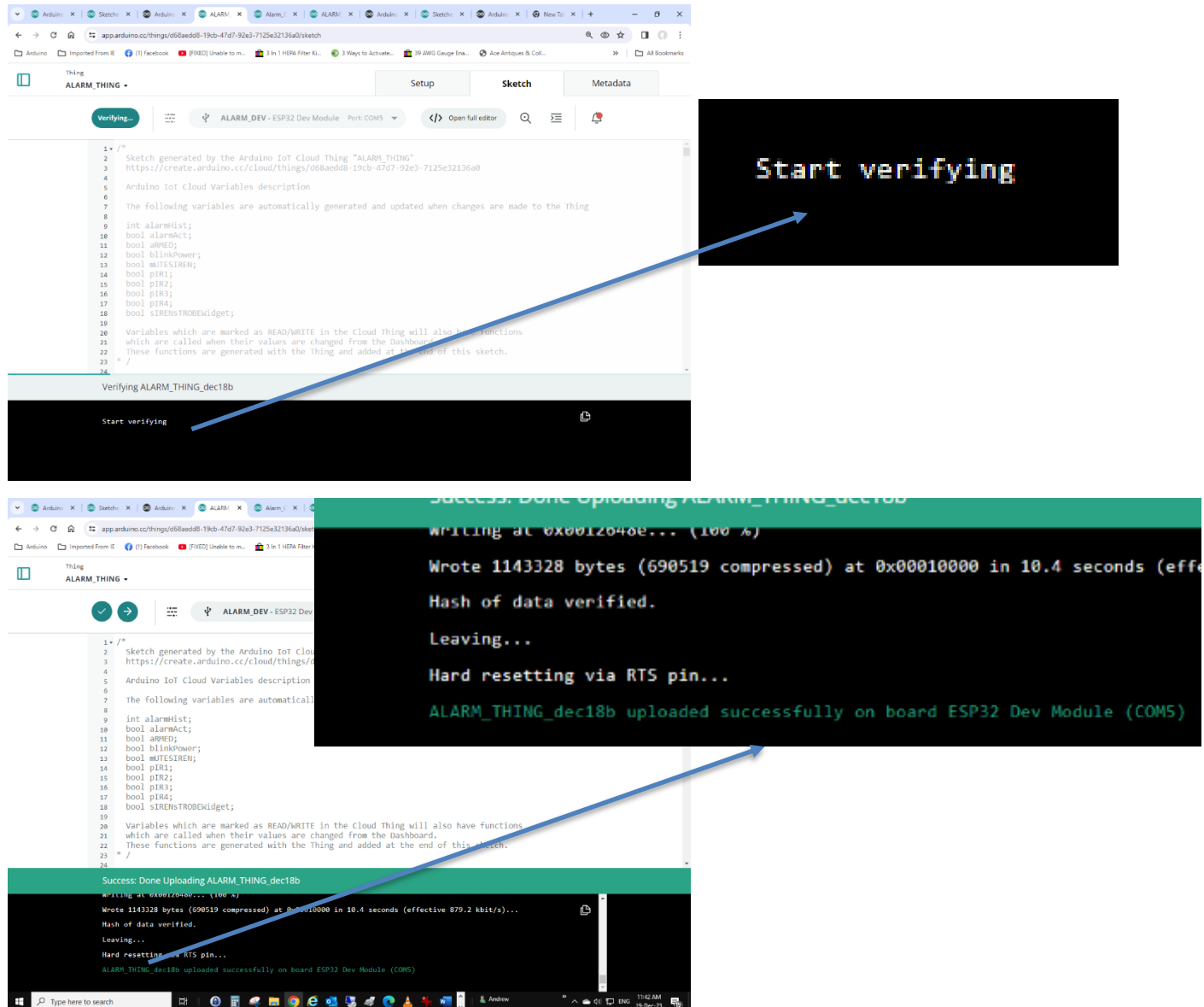
- Open the serial monitor (@115200 baud).

- Put the ESP32 into upload mode Pres and hold BOOT button briefly press EN button, release BOOT button, you should see 'waiting for download' in the monitor.



- Compile and upload.
- you should see the ESP go through setup and connect to WiFi and then Arduino Cloud
- If not there is some trouble shooting to do
- Open the Arduino Remote App you should have at least the Alarm\_Control and Alarm\_History Dashboard to pick for., if not see previous dot point.
- Open Alarm\_Control you should be able get all the PIR's to swap between red to green by connecting the test jig to +v and ground. If not you know the drill.
- Arm the system using the Arm\_System Slider switch ensure the Mute\_Siren Slider is off. Simulate an alarm with the jig you should see both the LED's on the board light up the alarm and Siren/Strobe indicators should change from red to green ticks. Slide the Arm\_System switch off. The two LED's should go off the indicators should go back to red X's. If not sorry!
- Time for a light refreshment or other indulgence of your choice.

22. Tested OTA upload works fine but it must be done from the thing page rather than the full editor.



23. Alarm board PI sensor reference voltage adjustment refer to:

[C:\AlarmProject\AlarmProject\\_Setting\\_PIR\\_Voltage\\_ref.pdf](C:\AlarmProject\AlarmProject_Setting_PIR_Voltage_ref.pdf)

# Connecting Up

24. Connecting up the to the existing panel is going to be different depending on what the old panel is.

Refer to the wiring diagram for guidance on connection,

In my case I wanted the ESP board to fit into the old alarm box so I made sure the matrix board the project is built on would be able to squeeze in. As mentioned, I just left the PIR's wired into the old system terminal block and connected the ESP board sensor connections into the active side of the PIR's. Siren's and Strobe are connected directly to the ESP board, which switches the positive side. The common side can remain connected to the old panel.

