Alarm Project

Part I Alarm Controller

About this project.

There is not any hardware build instructions with this project, as the finished hardware is just a single board wired point to point on matrix proto type board. 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. So I expect for anyone that did decide to build this, its their call to produce the hardware, from the circuit, in whatever form suits them.

Project objective

To create a home security monitor/alarm using elements of a pre-existing but redundant alarm system.

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 connected monitoring system that is now changed to a subscription model.

System 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).

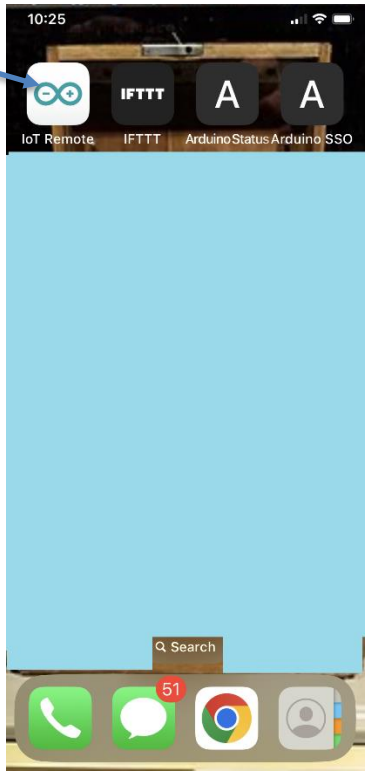
Finished project operation .

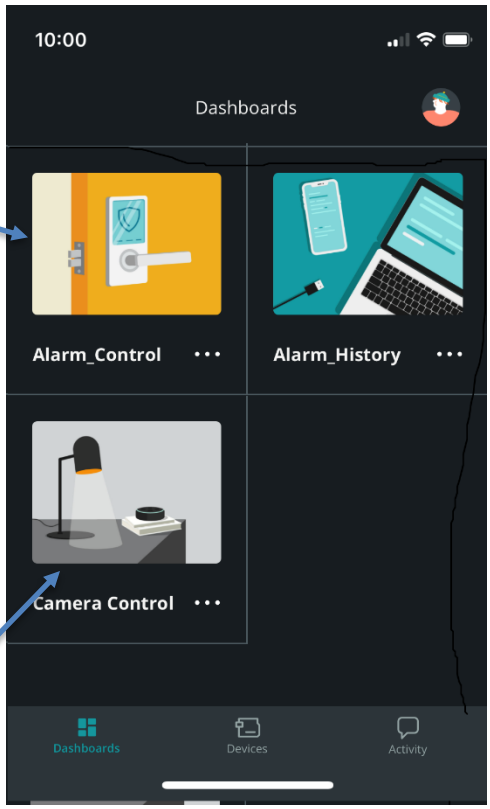
The alarm system is very simple to operate, it can be controlled and monitored with a mobile phone from anywhere as long as there is coverage and internet. There are only two user inputs and seven status indicators. No codes are required to access the app or the alarm panel.

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.





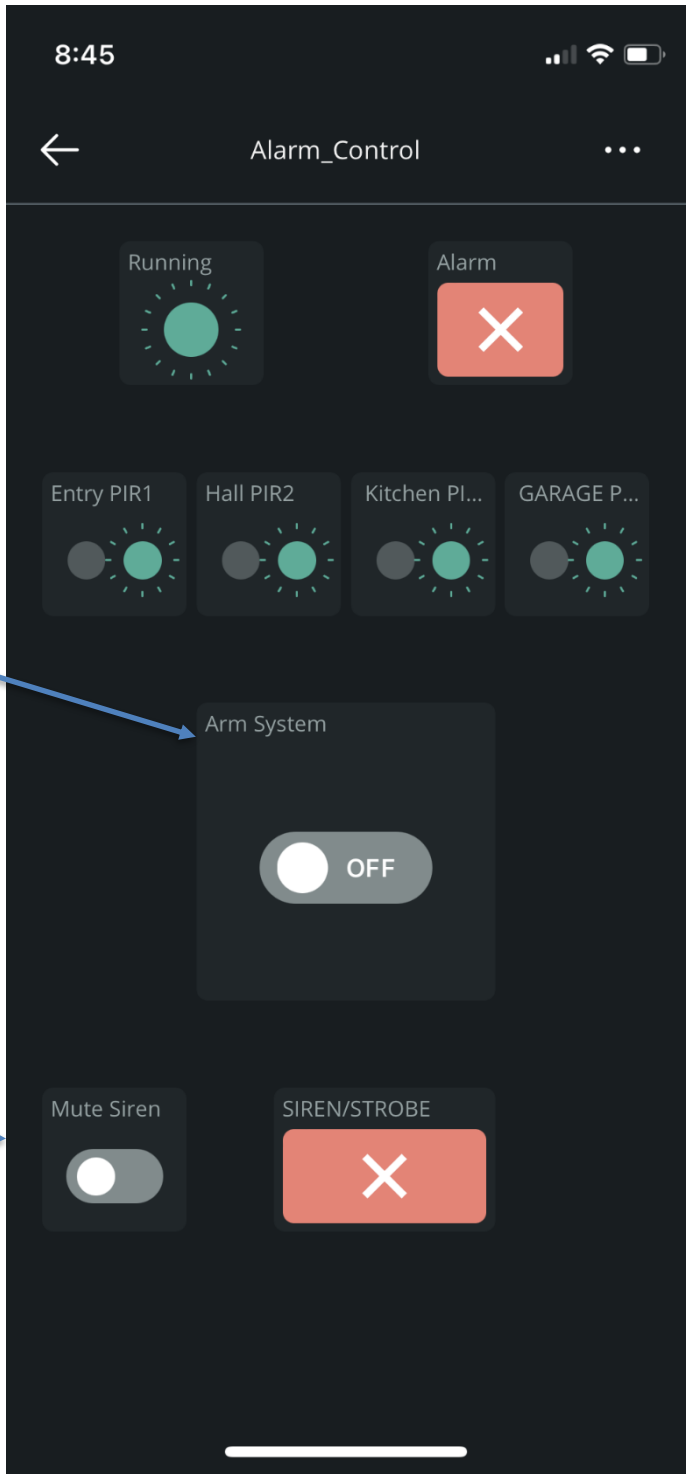
A separate but related project. I have built up a an ESP32-CAM video camera triggered wirelessly by this alarm system using a shared ArduinoCloud Variable. It uploads AVI's or stills to google drive. The project part II is complete, but I need to find time to write up.
From a programming perspective it is more challenging and more interesting than this project.

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

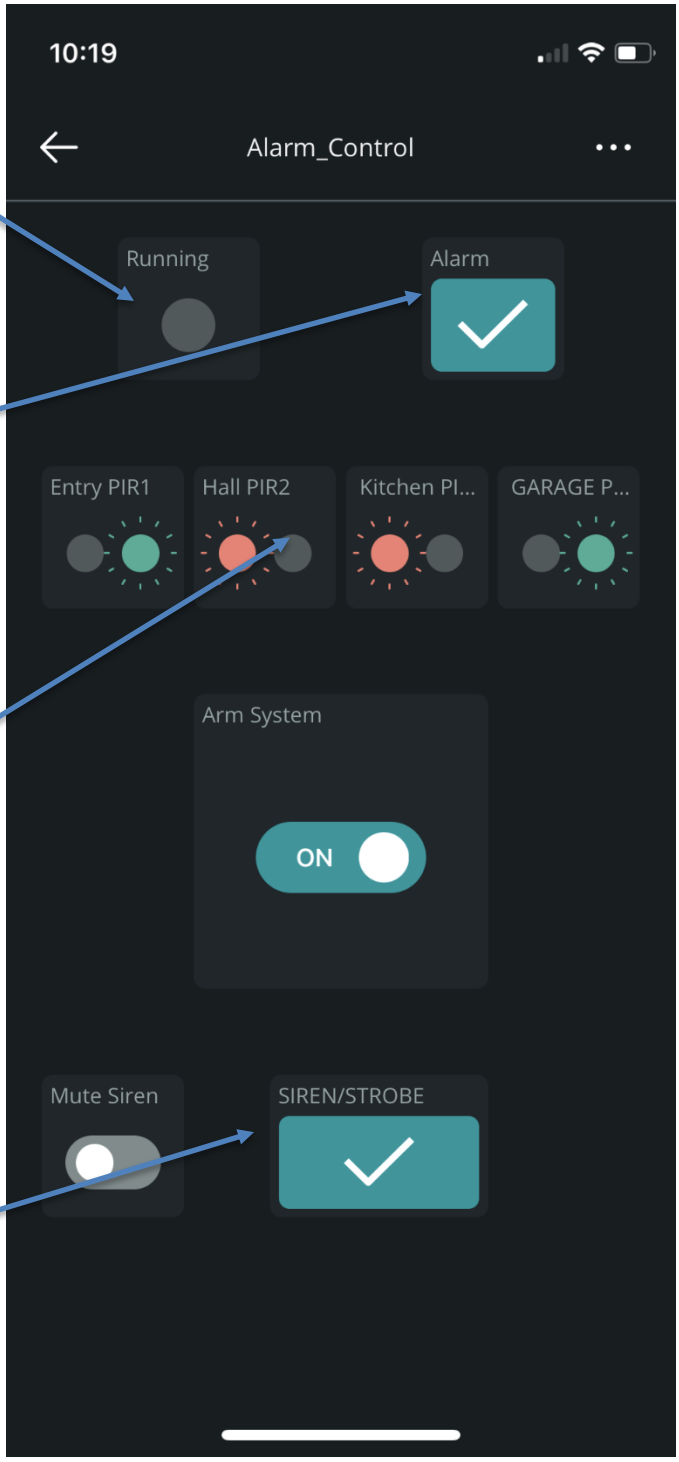
Running
Blinks green once per second, if its not blinking there is a problem. The alarm panel could be powered off or faulty, the local WiFi or internet or other critical service could be down where the alarm system is located.

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.

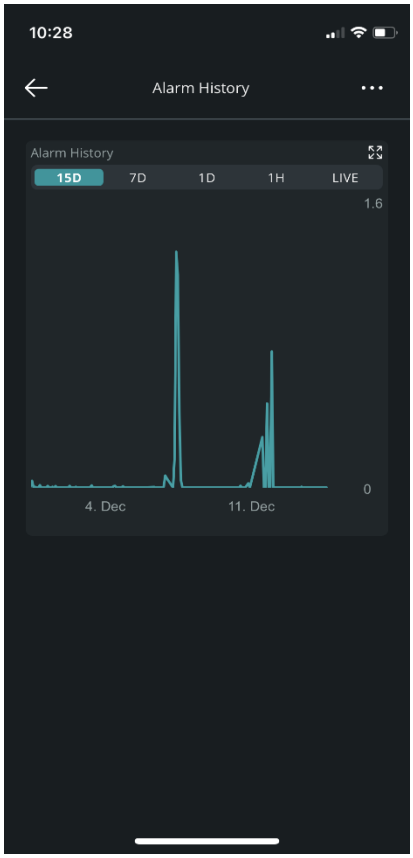
.

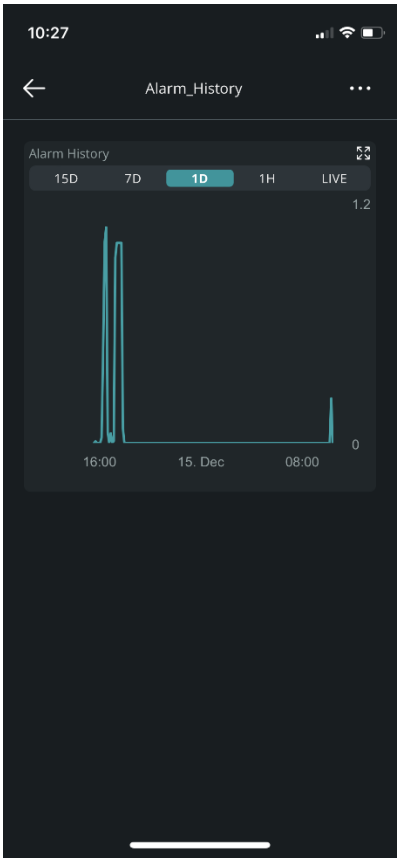
Siren/Strobe
When showing a red X the Strobe and Siren's are off, if it's a green the Siren's the strobe are logically on, however if Mute_Siren is set to on, only the strobe will be active.

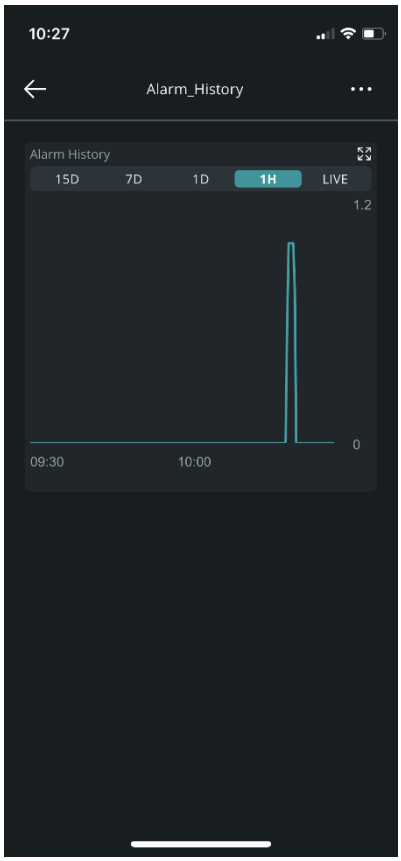


Alarm History

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 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.

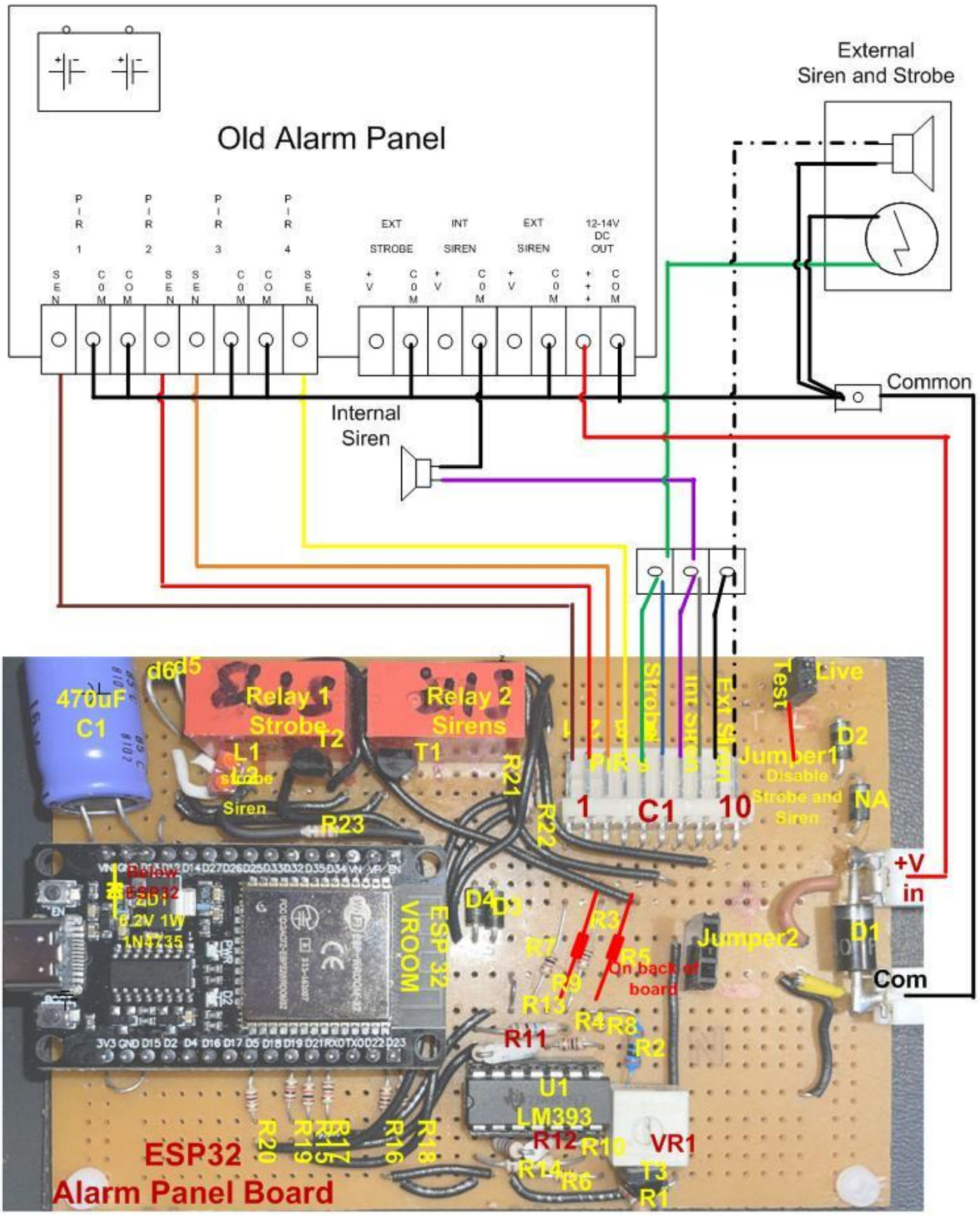






Connecting up the system

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.

Construction and programming

The file Pack.Zip is included with this project includes all software source code files, templates for IOT Thing and Dashboards, Cloud CLI executable, higher res circuit and layouts some more detailed set up guides for IFTTT etc. First thing to do if you are interested in building this is to create a folder in the root of the C drive C:\AlarmProject. (you may need to be granted elevated privileges to do this) Click Pack.zip to download the file then extract it to into that folder.

Prerequisites and hardware requirements

- 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.

Parts List

Item	Description		QTY	Jaycar Stock# AliExpress	@	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			$7.00	$7.00
2.	LM339 Quad comparator IC	1	ZL3339		$2.25	$2.25
3.	Relay DPDT 12V 2Amp NEC MR62-12SR	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		$075	$1.50
25.	Three position screw type terminal block (siren/strobe hook up)	1			$1.00	$1.00
				Total		~$46.5

The two NEC relays are hard to get so I have listed Jaycar equivalent.
Where known I have included Jaycar stock numbers

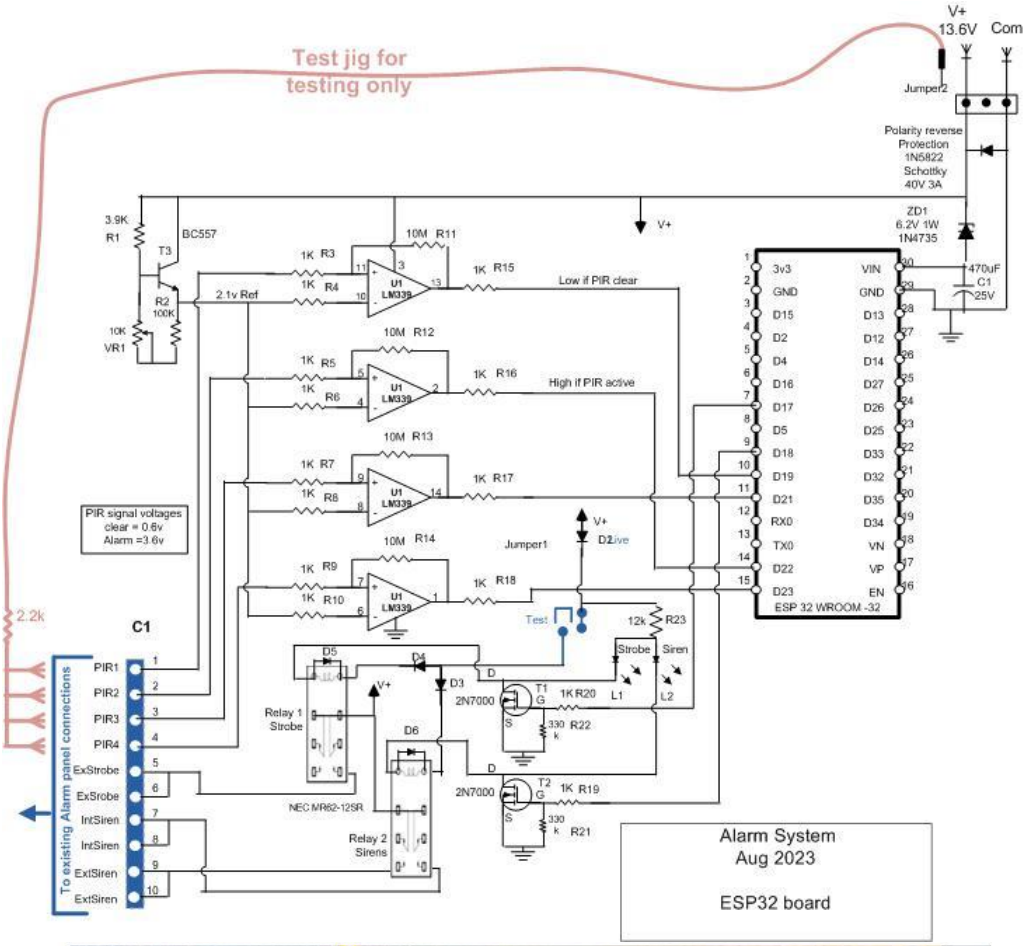
Other:
Sundry wire solder heat shrink etc
Tools: soldering iron, digital volt meter, wire cutters, screw drivers etc

Circuit description

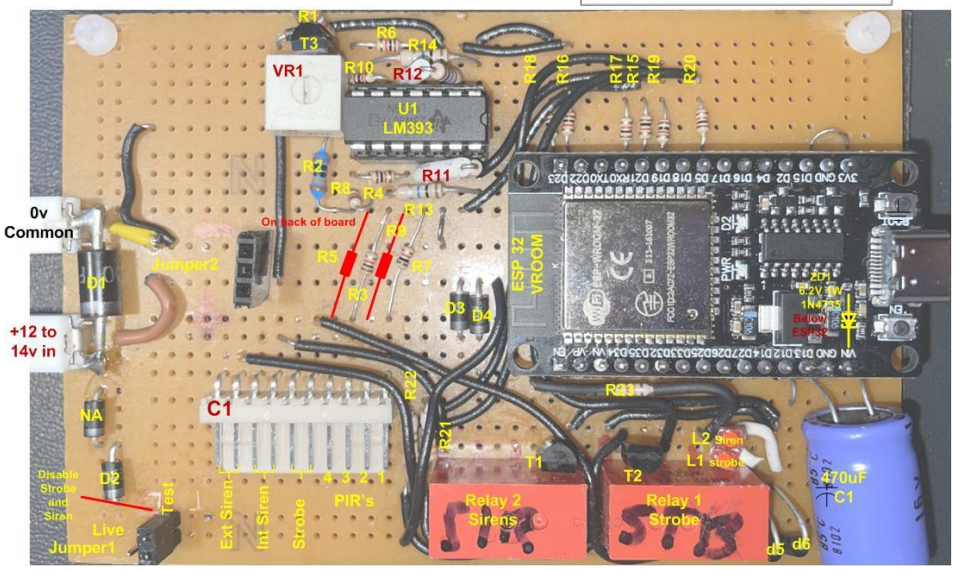
There is not much to it, the design evolved over time and I would certainly do it a little different if starting from scratch. That said, this is how it works.

The ESP-32 uses 6 pins to control all the alarm sensors etc. Four input pins are connected to four PIR's. via am 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 15mm x 87mm

Software description

The source code is commented and its pretty straight forward, so will just give an overview here. I am a very basic coder, so I am sure there is a lot of room for improvement by someone more skilled than I. Please refer to the flow chart on the next page for an over view of software operation.

I have chosen to use ArduinoIotCloud as the IoT service and IFTTT to send emails and mobile phone push notifications. (Source files included in C:\aLARMpROJECT

Source code configuration options that set default behavior and timings in AlarmProject.ini

Compile time options

```
// Comment these  out to allow emails and notifications respectively.
#define StopSendEmail
#define StopSendNotify

//Comment this out to aset time out to live system mode rather  than TEST
#define ALARMTEST

// comment out the line below to turn off most serial printing
#define DEBUG

// comment out this line for alarm active low PIRS else Active High
#define AlarmHigh
```

Timming control variables

AE_Hold	default 10 minutes minimium alarm state hold time This also determines how often alarm status emails are sent out
MaxSirenTime	default 180000ms 3 minutes Maximium time the Siren can continuously sound
SirenLockout	default 900000ms 15 minimum duration of lockout after siren has timed out(MaxSirenT expired).

The project is dependant on two services

ArduinoIotCloud

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

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

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.

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

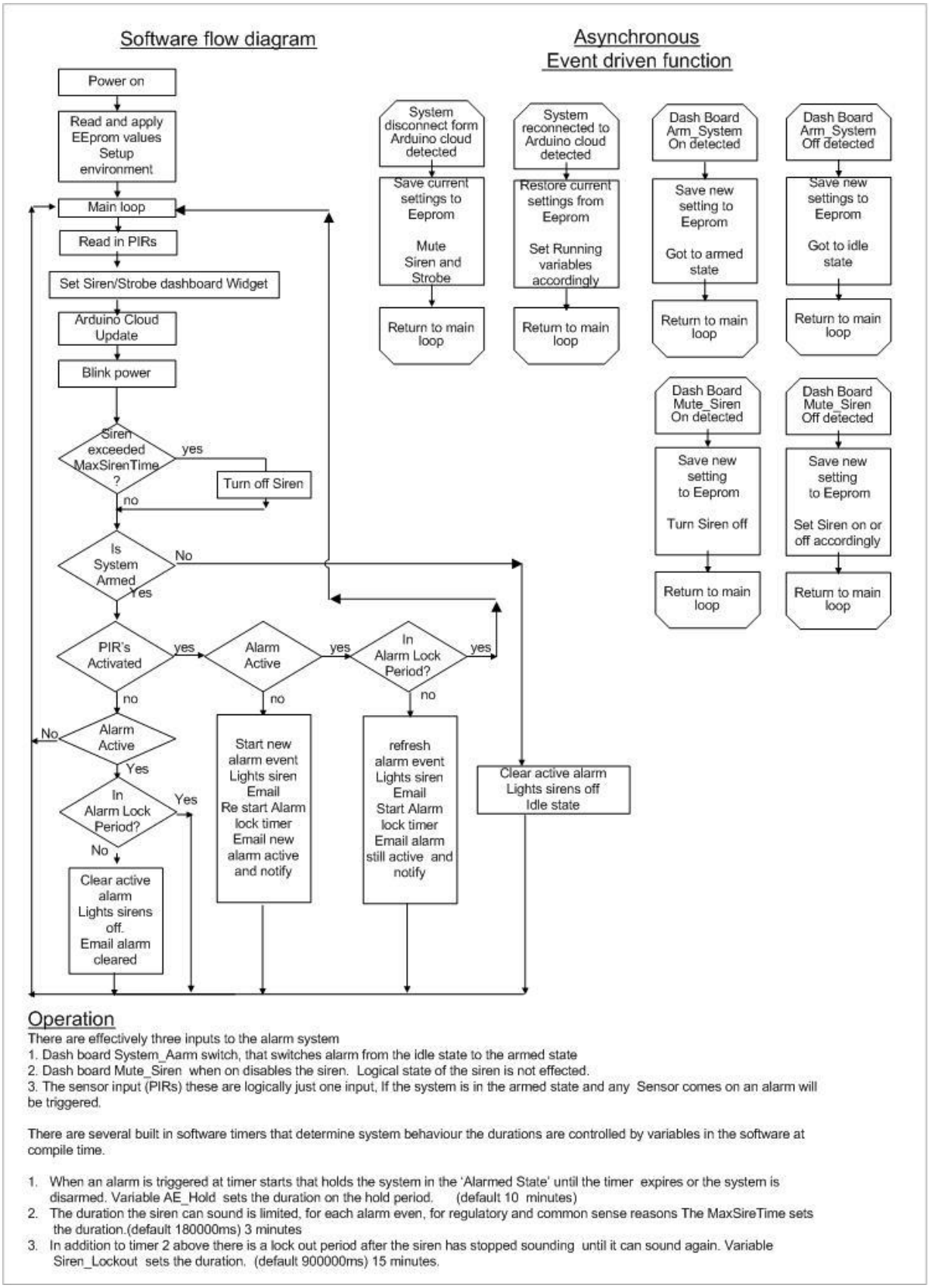
- The system will enter the ‘idle state’

Idle State

- 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.



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.

Software setup

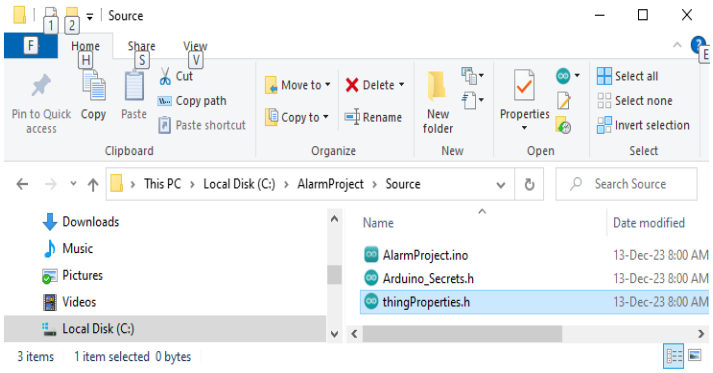
Summary steps

- Copy the project package to your computer.
- Setup Arduino Cloud account (if required)
- Manually add the esp32 dev module to your cloud environment
- 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. Create a folder C:\AlarmProject that you have full Read write change access to.
   (Elevated permission may be required)
2. Extract AlarmProject.zip to C:\AlarmProject.
3. You should see Source Template and CLI folders Along with other files and folders



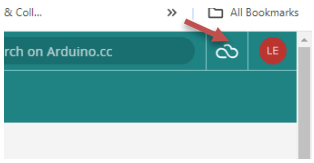
(it is important to use the exact folder name and Location as the rest of this guide uses links to that Folder)

You may want to add C:\AlarmProject to Qick Access

Setup Arduino Cloud account (if required)

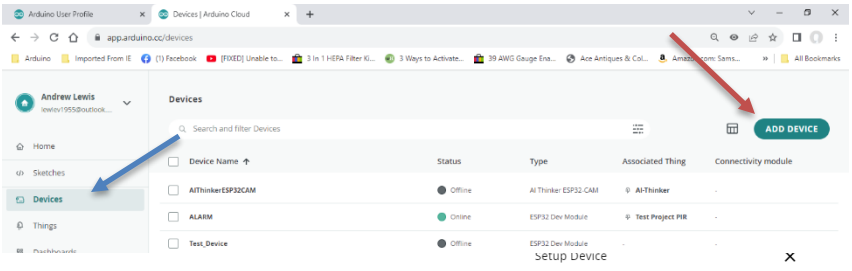
4. Setup Arduino cloud account if you don't have one already
   Ref https://support.arduino.cc/hc/en-us/articles/360016416280-Sign-up-for-an-Arduino-Cloud-plan
   Choose at least 'Entry' plan currently $US1.99 per month

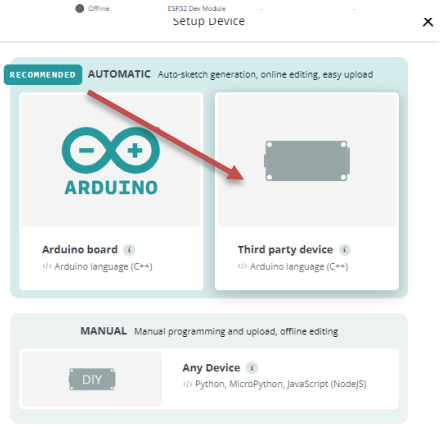
5. Ctrl Click Arduino cloud and Login if you are not already.
   Click the cloud button (top right) to go to the home page



Manually add the esp32 dev module to your cloud environment

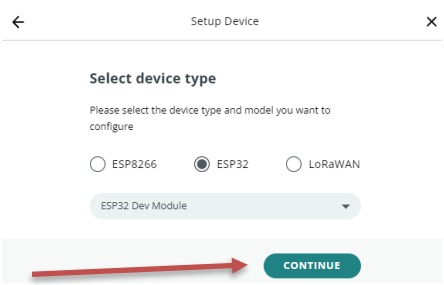
6. Add the ESP32 as a device into Arduino CLoud
    - Select Devices in the left side pane.
    - Select ADD DEVICE





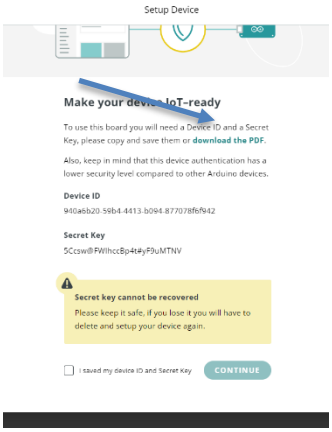
    - Choose third party device.

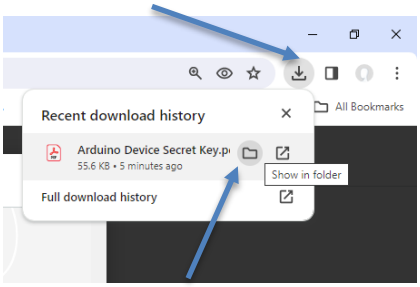
    - Select device type and model
    - ESP32 ESP32 DEV MODULE
    - Click on Continue.



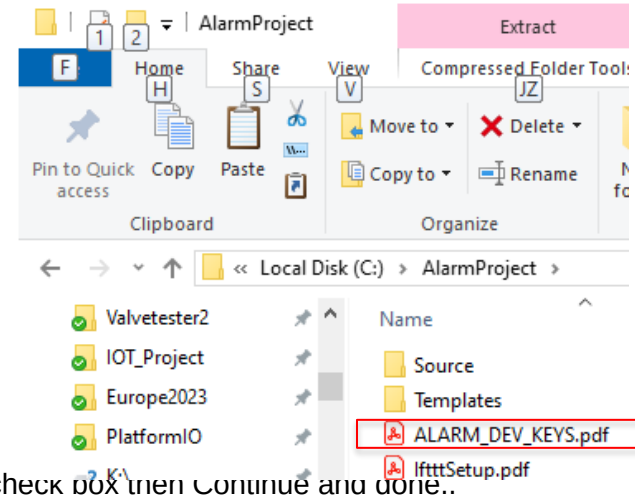
    - Type in ALARM_DEV as the Device Name
    - Click Next

    - 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

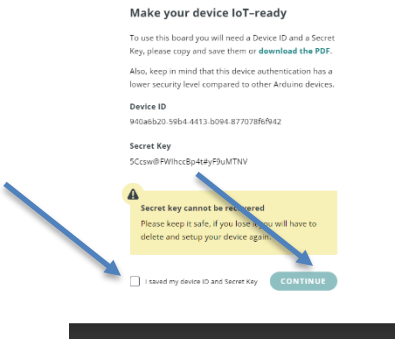


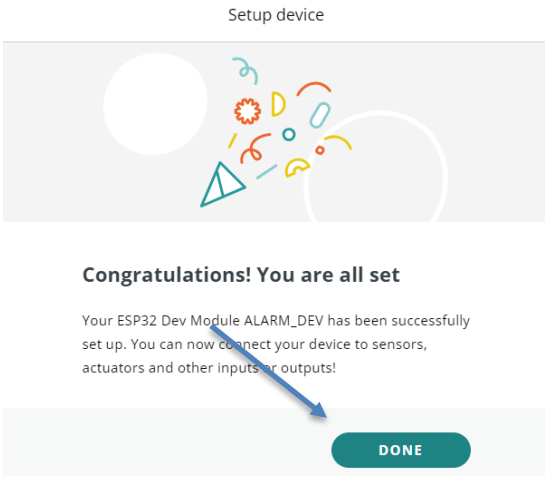


- Right click on the pdf file and rename it
- ALARM_DEV_KEYS.pdf
- Copy and paste to the project folder
  Save it in the project folder [C:\AlarmProject](C:\AlarmProject)



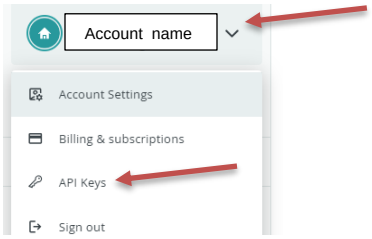
- Back in the browser
- Click on the 'I saved my Device ID and Secret' check box then Continue and done..



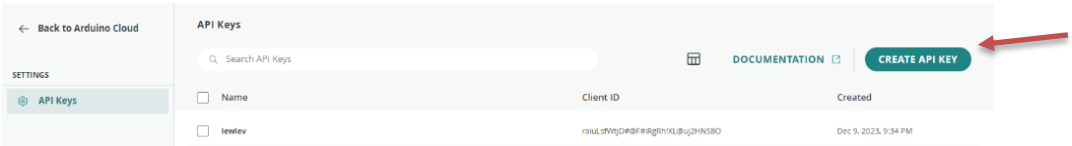


Setup Arduino cloud CLI (if required)

7. 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 Adruino Cloud home page
   - In the top left corner, you will find your account name

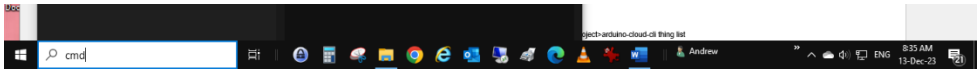


   - Click on the down arrow beside your name select API Keys

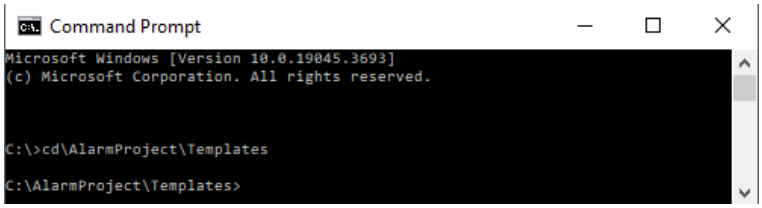


   - Click on the Create API Key button
   - A new key will be created.
   - You will be prompted to save a pdf file with the,new keys details.
   - Save it to *C:\AlarmProject\Templates say call it CloudCLI_API_Keys.pdf*

8. 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 the Command prompt. (don’t include the “”)



- **“cd\AlarmProject\Templates”**

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

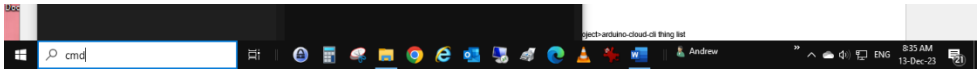
- **“arduino-cloud-cli credentials init”**

  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

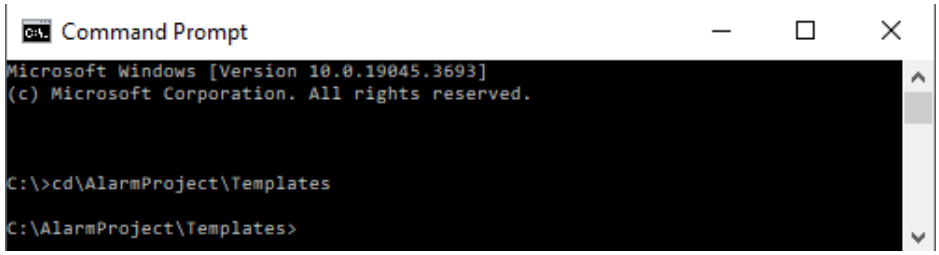
- If successful you should a response like the one above.

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 dasboards

10. Create the **ALARM_THING** thing from template.
    - 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

    - **"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 pasted any where with cntrl P)

11. Create the **Alarm_Control** dashboard from template

    - Please copy just the ID in yellow and past 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

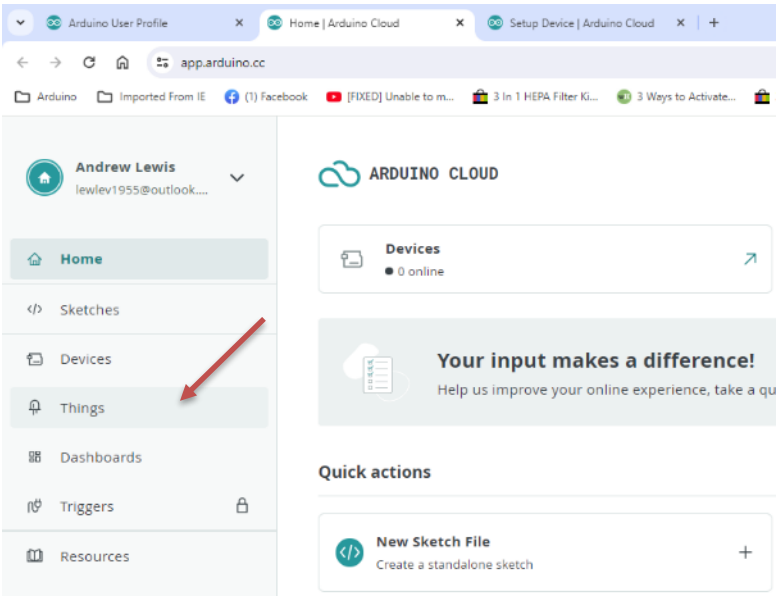
Repeat step 11 again for the Alarm_History Dashboard

“arduino-cloud-cli dashboard create --name Alarm_History --template Alarm_History_Temp.yaml --override ALARM_THING=” *c5713bfc-00ad-4a9b-b75a-6564d207606b*

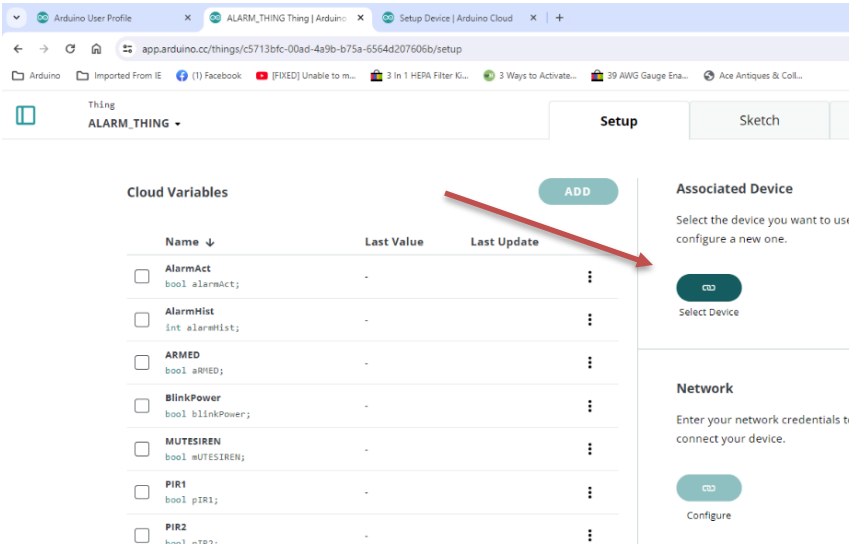
name: Alarm_History
id: f3bab9f9-3b6d-42b2-9540-1eabdb7a4e54
updated_at: 2023-12-13 03:06:18.990525 +0000 UTC
widgets: Alarm History

13. Associate ALARM_DEV and ALARM thing

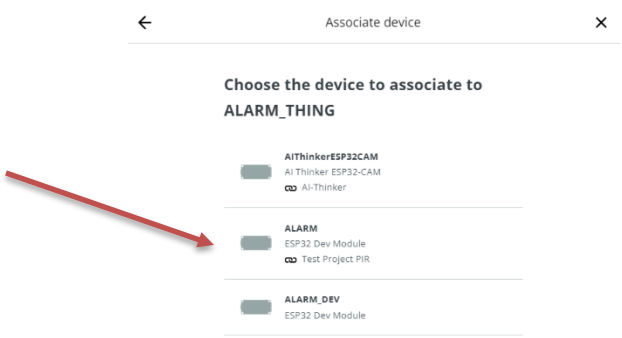
From the Arduino Cloud home page choose Things



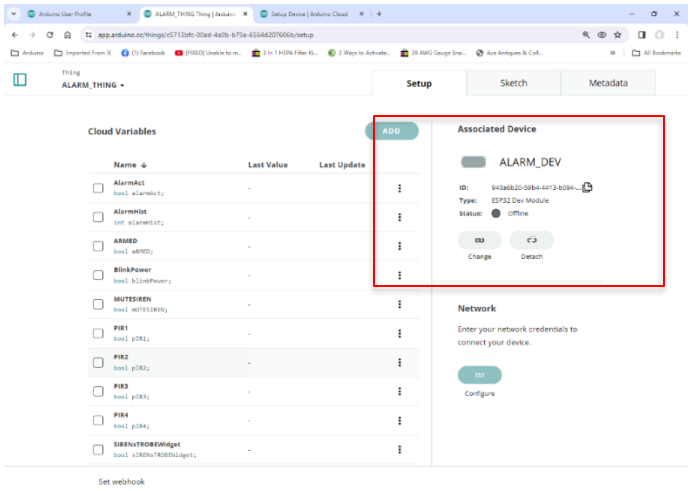
Then choose ALARM_THING
Then click on ‘Select Device’



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\IfttSetup.pdf if needed.

Setting	Applet 1	Applet 2
Name	Alarm_Event	Alarm_Nofification
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.

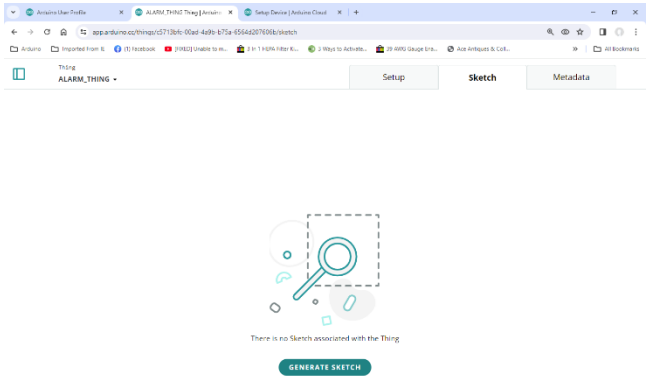
15. Test the applets by pasting the URLs below (with your connection key) into a browser
An email and a mobile phone notification should be generated.

```
https://maker.ifttt.com/trigger/Alarm_Event/with/key/your_key
https://maker.ifttt.com/trigger/Alarm_Notification/with/key/
```

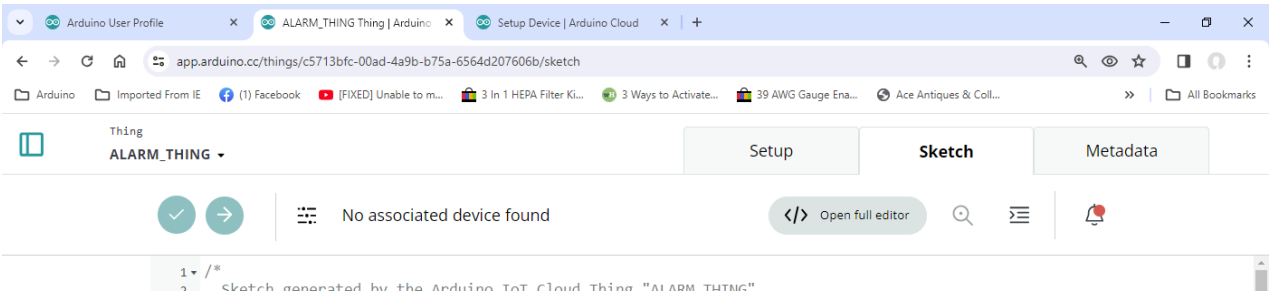
Configure software user specific items like Wi-Fi etc.

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.

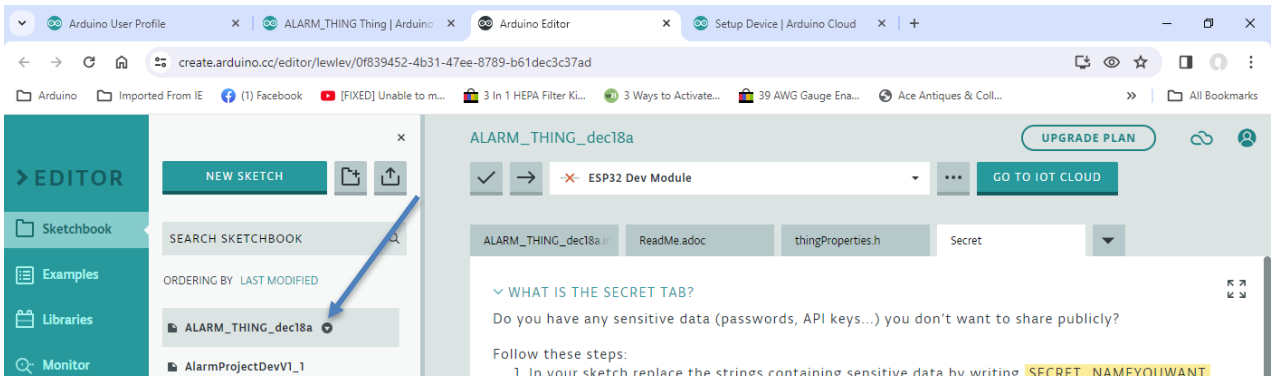
- Open [Arduino Cloud] home page login if required,
- Open Things
- Open thing ALARM_THING then click on the sketch tab
- You should see the dialogue below click on “Generate Sketch”



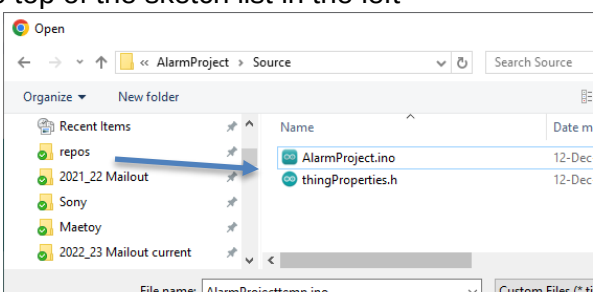
- The sketch will open in the basic editor



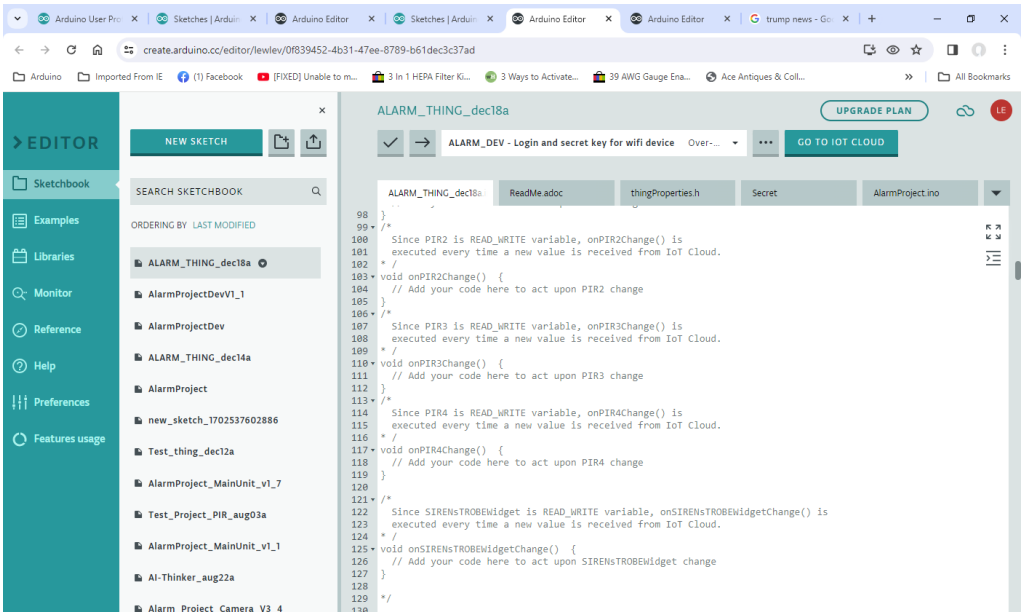
- Click on Open full editor
  The web editor will open with 4 tabs
  ALARM_THINGxxxx.ino, ReadMe,thingProperties.h and Secret
  We will effectively replace ALARM_THINGxxxx.ino, with the AlarmProject code.



- 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 file tabs with AlarmProject.ino added
- Open ALARM_THINGxxxx.ino tab in the web editor, 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 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 AlarmProject.ino tab in the web editor, right click and choose ‘Select All’ right click again and choose ‘Copy’
- Open the ALARM_THINGxxxx.ino tab and place the cursor at the very bottom of the file after the last line of text in the file. Right click and choose ‘Paste’
  Note this will populate WiFi credential and Secret ID in ALARM_THING as well.
- Now you should have the code from AlarmProject.ino in ALARM_THINGxxxx.ino.
- Select the AlarmProject.ino tab then click on the little down arrow tab and select delete AlarmProject.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.

    AlarmProject.ino
    Approx Line 138 paste between the quotes the ifttt key from the account setup earlier
    - const char *key = "Your ifttt key";

Configure Mobile Phone apps.

18. Goto the app store

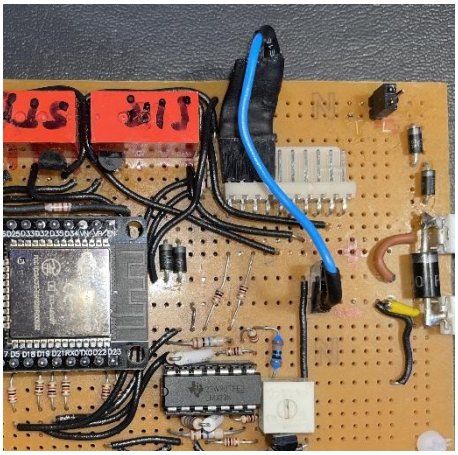
- Install the Arduino IOT remote app use the Arduino Cloud Account credentials.
- Install the IFTTT app use your IFTTT account credentials.

A brief guide is included in the pack C:\AlarmProject\Alarm_Project_Mobile App installs.pdf "Mobile App Installs.pdf" if required.

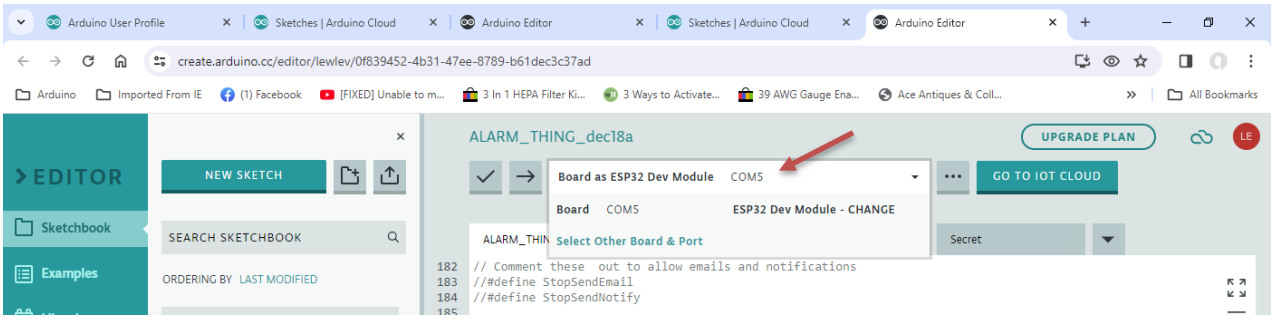
19. Thats it.

20. 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.

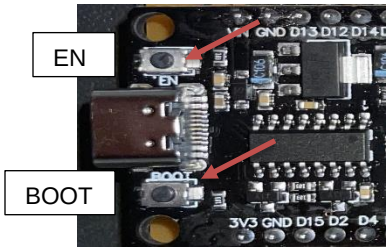


- Connect your board to a 12v power supply and the ESP32 to your PC via USB conditions.
- Open the Arduino Cloud Web Editor then the ALARM_THINGxxxx.ino file
- Uncomment the Debug definition in the ino file so we get some definition serial monitoring
- Check /set the device connection to Board COMx ESP32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



- Open the serial monitor (@115200 baud.

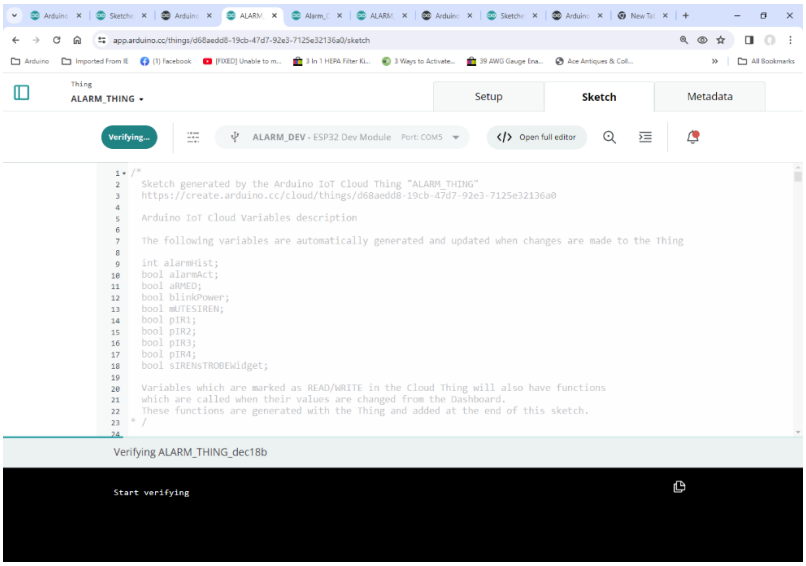
- Put the ESP32 into upload mode Pres and hold BOOT button briefly press EN button, release BOOT button, you should see 'waiting for download' in the monitor.

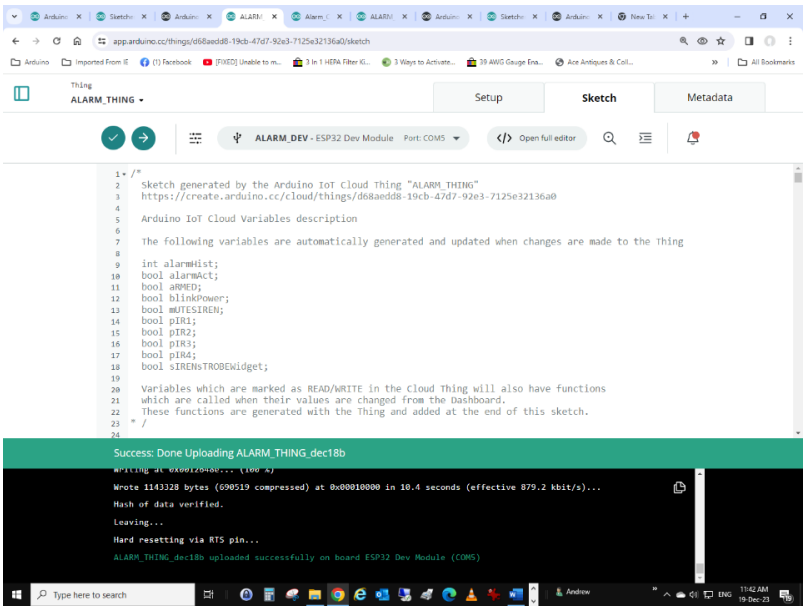


- Compile and upload.

- 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.

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





22. Alarm board PI sensor reference voltage adjustment refer to: C:\AlarmProject\AlarmProject_Setting_PIR_Voltage_ref.pdf

Connecting Up

23. 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 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,