

MODULE DE SURVEILLANCE AUTOMATIQUE

JEEDOM WATCH SYSTEM (JWS)



Table des matières

Historique des versions	3
V0.1 – 06 Septembre 2024.....	3
V1.0 – 04 Octobre 2024	3
V1.1 – 11 Octobre 2024	3
V1.11 – 02 Novembre 2024	3
V1.2 – 09 Novembre 2024	3
V1.3 – 20 Janvier 2025	3
Présentation générale	4
Matériel	4
Principe de fonctionnement.....	5
Premier lancement.....	5
Détection d’une erreur.....	7
Affichage sur l’écran OLED	7
Affichage des LEDs	8
Codage des erreurs	9
Autosurveillance de la connexion au réseau WiFi.....	9
Arrêt de la surveillance.....	9
Installation et mises à jour	10
Installation initiale.....	10
Flashage d’un ESP32 avec web.esphome.io	11
Mises à jour par OTA.....	12
Procédure avec le serveur web.....	12
Procédure avec l’IDE Arduino	13
Affichage pendant les mises à jour	13
RAZ du module	14
Configuration de Jeedom	15
Causes probables des erreurs résultants lors des tests	17
Liens pour l’achat du matériel nécessaire.....	18
Réalisation pratique	19
Réalisation des prises USB	19
Montage de la LED RGB	21
Montage de l’ESP avec son support et de la platine relais dans le boîtier	22
Finitions.....	24
Schéma électronique.....	26
Table des figures	27

Historique des versions

V0.1 – 06 Septembre 2024

Version initiale

Cette version est minimaliste et ne gère aucun dispositif de retour d'état (écran OLED ou LED). Tous les paramètres Wifi et MQTT (serveur, mots de passe, @IP, ports) sont codés directement dans le fichier source.

V1.0 – 04 Octobre 2024

La version 1.0 est la première version diffusée et mise en service, et va prendre en charge :

- Un écran OLED de type SH1106 ou SSD1306 (optionnel),
- Un indicateur d'état par diode LED RGB (ou 3 LEDs séparées rouges, verte et bleue),
- La mémorisation des paramètres Wifi et MQTT en mémoire EEPROM,
- La gestion de la connexion au réseau Wifi avec les paramètres renseignés,
- Les mises à jour par procédure OTA, soit par serveur web dédié, soit par l'IDE Arduino.

Ainsi que la correction de quelques bugs résiduels mineurs.

V1.1 – 11 Octobre 2024

Les évolutions et corrections apportées par la version 1.1 sont les suivantes :

- Modification de la gestion des LED pour passer en mode PWM, permettant en particulier un allumage et une extinction progressive, et le réglage de la luminosité (via le code source).
- La gestion du multi-tâche pour l'initialisation du module en exploitant les deux cœurs de l'ESP32,
- L'ajout d'une barre de progression et du libellé de l'étape d'initialisation effectuée,
- La correction des bugs suivants :
 - Le comportement lorsque les identifiants Wifi entrés dans le mode AP sont erronés,
 - La suppression du délai d'attente de 30 secondes juste après l'initialisation.

V1.11 – 02 Novembre 2024

Augmentation du temps de cycle des tests de 30 secondes à 1 minute, permettant au système hôte de procéder à certaines mises à jour pouvant être plus longues que prévues, en particulier pour MQTT.

Un arrêt/marche ne sera déclenché que 5 minutes après l'apparition d'une erreur, au lieu de 2 minutes 30 secondes.

V1.2 – 09 Novembre 2024

Cette version apporte les évolutions suivantes :

- Après l'initialisation du module, il y a une vérification permanente de la connectivité avec le réseau WiFi défini dans les paramètres. En cas de détection de déconnexion du réseau, on tente de se reconnecter et si la tentative n'aboutit pas, on effectue un RESET du JWS.
- Définition du nom d'hôte 'JWS_ESP32' permettant l'identification du module sur le réseau WiFi.

V1.3 – 20 Janvier 2025

Ajout d'une fonction de mode maintenance/opérationnel permettant de suspendre la surveillance de la box jeedom.

Amélioration des interfaces web.

Présentation générale

Le module de surveillance JWS s'articule autour d'un microcontrôleur ESP32 et permet, en toute autonomie, de détecter des défaillances éventuelles d'un système domotique (Jeedom, Home Assistant, ou autre) hébergé sur une box autonome (type Atlas, Luna, RPi, mini-PC, etc...).

La RAZ étant effectuée par un redémarrage électrique (arrêt/marche), **ce module ne convient pas** pour des systèmes domotiques virtuels hébergés sur des NAS ou autres serveurs assurant d'autres fonctions.

Les défaillances sont détectées au niveau applicatif (Jeedom/HA via protocole MQTT) et réseau (IP), et le module va redémarrer électriquement jusqu'à deux fois la box domotique pour tenter de récupérer un fonctionnement normal.

Il se connecte d'une part sur une alimentation 5v par un câble USB-C mâle (15W max), et d'autre part sur la box domotique également avec un câble USB-C mâle. Il s'auto-alimente directement via la prise USB-C qu'il commande, et ne nécessite donc aucune alimentation externe.

Matériel



Figure 1 - Matériels utilisés

L'ensemble du matériel s'articule autour :

- D'un ESP32 4Mo/240Mhz/Wifi standard 30 broches, avec son support,
- D'un module relais avec contacts NO (Normally Open) et NC (Normally Closed),
- D'un écran OLED de type SH1106 ou SSD1306 offrant une résolution de 128 (L) x 64 (H) pixels avec une interface I2c (4 broches : GND, VCC, SCL, SDA),
- D'une LED RGB (ou 3 LED rouge, verte, et bleue) avec ses 3 résistances de limitation de 180 à 270 ohms,
- De deux prises USB-C type châssis disposant au minimum de 6 contacts (VBUS, GND, D+, D-, CC1, CC2),
- D'un câble USB-C to USB-C mâle (prise USB-C OUT vers la box),
- D'un boîtier (dimensions intérieures minimales : L120 x l75 x h27 mm),
- D'une feuille acrylique translucide fumée de 75x35x1mm (cache pour l'écran OLED),
- De la quincaillerie standard (vis, rondelles, et écrous de 1.6, 2.5 et 3mm), du câblage (calibre 24 AWG), de la gaine thermo-rétractable.

Principe de fonctionnement

Au démarrage, l'ESP32 va vérifier la présence d'un écran OLED et afficher le cas échéant un logo, puis lancer la procédure d'initialisation :

- Autotest des LED (allumage et extinction progressive et successif pendant une seconde des LED rouge, verte et bleue),
- Initialisation de l'EEPROM,
- RAZ éventuelle de l'EEPROM,
- Lecture des paramètres écrites dans l'EEPROM,
- Connexion au réseau Wifi (à l'exception du premier lancement, voir le chapitre '[Premier lancement](#)').
- Initialisation des routines de mise à jour par OTA,
- Initialisation des échanges MQTT,

Ces phases sont matérialisées sur l'écran OLED par une barre de progression et l'affichage du libellé des étapes terminées.

En phase de surveillance (dans le mode dit 'opérationnel'), il va ensuite effectuer toutes les 60 secondes l'ensemble des tests suivants :

- Ping réseau vers la box domotique,

Ce ping permet de s'assurer que la box domotique est toujours connectée au réseau, et qu'elle n'est pas figée ou plantée.

- Maintien de la connexion MQTT,

Une connexion MQTT doit être activée et gérée par un serveur MQTT (broker Mosquitto,...), qui se signale via la messagerie associée. L'absence de ce signalement peut résulter d'un arrêt logiciel du serveur. Généralement, les box domotiques s'appuient sur ce protocole pour gérer les modules présents sur le réseau Zigbee ou les modules connectés en Wifi. Il n'est donc pas nécessaire dans la plupart des cas de prévoir l'installation et l'activation spécifique d'un plugin de gestion de ce protocole.

- Emission et attente de la réponse à un message transmit en MQTT.

Outre la vérification de la présence d'un broker, le module JWS va émettre un message MQTT spécifique (ping) et attendre en retour la réponse de la box domotique (pong). S'il n'y a pas de réponse, il est possible que le démon du plugin se soit arrêté.

Dans tous les cas, toute(s) détection(s) d'erreur(s) seront signalées par un affichage d'un écran récapitulatif sur l'écran OLED et l'allumage des LEDs correspondantes.

En mode 'maintenance', la surveillance de la box domotique est suspendue sans limite de durée. Ce mode permet par exemple de procéder à des mises à jour longues stoppant les services MQTT et/ou IP, et pouvant excéder les limites au-delà de lesquelles le module JWS procéderait à un arrêt / marche correctif de la box.

Voir le chapitre '[Arrêt de la surveillance](#)'.

Premier lancement

Au premier démarrage (après téléversement ou une RAZ de l'EEPROM), l'ESP32 doit mémoriser les paramètres de connexion au réseau Wifi utilisé par la box domotique (SSID et mot de passe), l'adresse IP du serveur MQTT et de la box domotique à surveiller (il s'agira de la même adresse si le serveur MQTT a été

installé en mode local), le nom d'utilisateur, le mot de passe et le port MQTT utilisé (par défaut, le port est 1883).

L'ESP va donc initialiser un serveur web et une liaison Wifi en mode point à point (AP), visible avec le SSID `JWS ESP32-AP` à l'adresse 192.168.4.1, sur laquelle il faudra se connecter avec n'importe quel terminal équipé d'une connexion Wifi et d'un navigateur Internet, comme un smartphone par exemple.

Dans ce mode, l'écran OLED laisse affiché le logo de démarrage, et la LED rouge s'allume (fixe). Il suffira ensuite de renseigner les différents champs affichés, avant de valider les paramètres de configuration avec le bouton 'Connecter'.

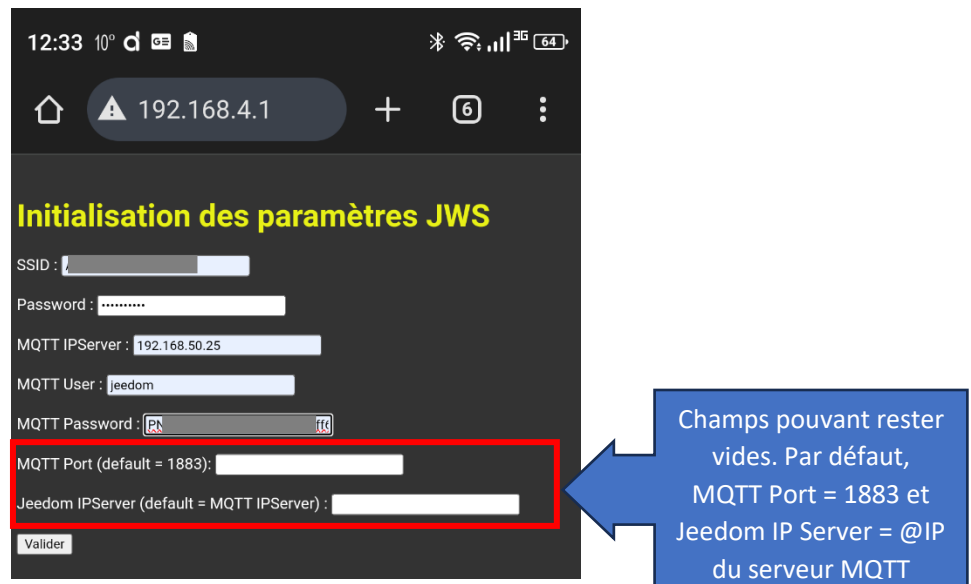


Figure 2 - Copie d'écran sur un smartphone du serveur web en mode AP

Une fois les paramètres validés, l'ESP quitte le mode AP et va se réinitialiser pour tenter de se connecter sur le réseau Wifi ainsi défini et initialiser les liaisons sur le réseau MQTT.

L'écran OLED va afficher le logo de démarrage et la barre de progression de l'initialisation,



Figure 3 - Logo de démarrage

Puis après quelques secondes, la page principale.



Figure 4 - Page principale

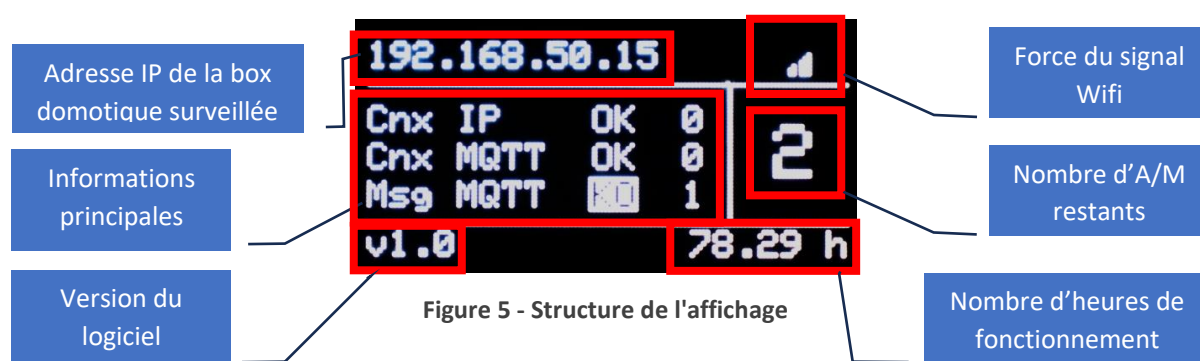
A partir de ce moment, le module va surveiller les connexions MQTT et Wifi avec la box domotique, et l'écran OLED va s'éteindre.

Détection d'une erreur

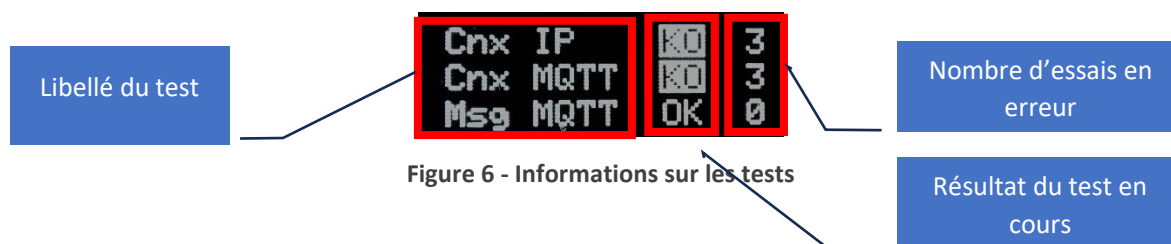
En cas de détection d'erreur sur l'un des tests, la (ou les) LEDs correspondantes vont s'allumer et l'écran OLED va être activé, et ce tant que l'erreur ne sera pas résolue.

Affichage sur l'écran OLED

L'affichage est structuré en différentes zones distinctes :










- **Adresse IP du module** : indique l'adresse IP attribué au module JWS par le routeur du réseau Wifi,
- **Force du signal Wifi** : indique la puissance reçue du signal Wifi,
- **Version du logiciel** : indique la version du logiciel,
- **Nombre d'heure de fonctionnement** : indique le nombre d'heure de fonctionnement depuis la mise en marche ou le dernier RESET effectué, en heure + 1/10 d'heure,
- **Informations principales** : indique le résultat des tests effectués (test, résultat, nombre d'occurrences en erreur),



- **Nombre d'arrêt/marche restants** : indique le nombre total d'arrêt/marche pouvant être effectués, décroissant de 2 à 0. A 0, il n'y aura plus d'arrêt provoqué par la suite.

Affichage des LEDs

LED RGB	Fixe	Clignotante	Pulsée
Rouge 	Erreur ping réseau IP Au démarrage, l'ESP est en mode AP (attente des paramètres)	Mise à jour OTA en cours	3x = fin de mise à jour OTA avec erreur
Vert 	Erreur connexion MQTT	-	3x = fin de mise à jour OTA sans erreur
Bleu 	Erreur échange de messages MQTT	-	1x toutes les 60 secondes = JWS en service En permanence = mode maintenance
Cyan 	Erreurs échange de messages MQTT et connexion MQTT	-	-
Magenta 	Erreurs ping réseau IP et échange de messages MQTT	-	-
Jaune 	Erreurs ping réseau IP et connexion MQTT	-	-
Blanc 	Erreurs ping réseau IP, connexion MQTT et échange de messages MQTT	-	-

Après 5 tests sans succès, le module va provoquer un arrêt /marche de la box domotique en coupant l'alimentation de la prise USB pendant 0,5 secondes pour la redémarrer.

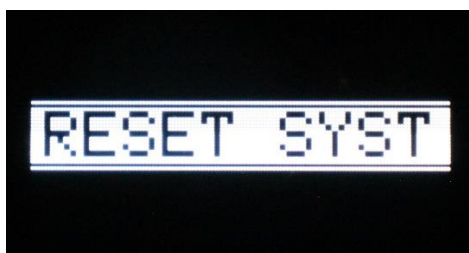


Figure 7 - RESET de la box

Après deux tentatives successives de redémarrage infructueux, le module JWS ne provoque plus d'arrêt et se limite à signaler les occurrences des tests en erreur, et ce tant que les erreurs détectées ne sont pas résolues.

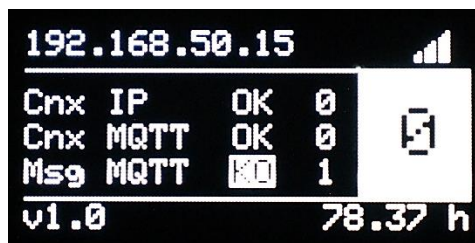


Figure 8 - Plus aucun redémarrage n'est possible (compteur à 0)

Codage des erreurs

Le codage interne des erreurs est le suivant :

Erreur Messages MQTT	Erreur Connexion MQTT	Erreur Connexion IP	N° Avarie
-	-	-	0
-	-	X	1
-	X	-	2
-	X	X	3
X	-	-	4
X	-	X	5
X	X	-	6
X	X	X	7

Autosurveillance de la connexion au réseau WiFi

Après l'initialisation, le module JWS va effectuer une surveillance permanente de la connexion avec le réseau Wifi.

En effet, en cas de perte de la connexion réseau, il faut tout d'abord s'assurer que le module JWS n'est pas incriminé dans la perte de cette connexion, avant de lancer les réactions prévues dans le cas où la box domotique est hors service ou injoignable.

Le cas échéant, le module JWS va procéder à une tentative de reconnexion avec un time-out de 10 secondes, et si cette tentative n'aboutit pas, le module JWS est redémarré.

Arrêt de la surveillance

Le module JWS peut être disposé dans deux modes différents :

- Mode opérationnel (mode normal)

Dans ce mode, le module JWS va assurer une surveillance permanente de la box Jeedom à laquelle il est associé.

- Mode Maintenance

Lorsque ce mode est sélectionné, le module JWS va arrêter toute surveillance sans limite de durée de la box Jeedom, et ne tiendra plus compte de l'absence éventuelle de réponses aux messages MQTT ou aux pings IP, ou de la perte de la liaison MQTT.

Ce mode permet d'isoler et de sécuriser la box Jeedom pour lui permettre d'effectuer des opérations de maintenance longues (mises à jour, ...), et dont la durée prévisible serait supérieure à la période maximale au-delà de laquelle le module JWS réagirait par un arrêt/marche (en principe après 5 minutes).

L'activation de ce mode se traduit par un clignotement continu en mode pulsé de la LED bleue, et l'affichage permanent sur l'écran OLED du message « JEEDOM MTN ».



Figure 9 - Mode maintenance

Le changement de mode s'effectue à partir de la page web 'Paramètres JWS' disponible sur l'@IP attribuée au module JWS :



Figure 10 - Sélection du mode à partir du serveur web

Les boutons Maintenance et Opérationnel permettent la bascule entre ces deux modes, le mode activé étant rappelé sur la ligne Mode : Maintenance ou Mode : Opérationnel.

Installation et mises à jour

Installation initiale

L'installation initiale du firmware (fichier binaire en .bin) sur l'ESP32 peut être effectuée avec le site web spécialisé pour les ESP ici : <https://web.esphome.io>. Mais il n'est accessible uniquement qu'avec les navigateurs Google Chrome ou Microsoft Edge à ce jour (l'utilisation du composant WebSerial est requis).

A défaut, il est possible d'installer et d'utiliser un outil de téléversement spécifique des fichiers .bin comme ESPHome Flasher (disponible au téléchargement sur Github ici : <https://github.com/esphome/esphome-flasher>).

Flashage d'un ESP32 avec web.esphome.io

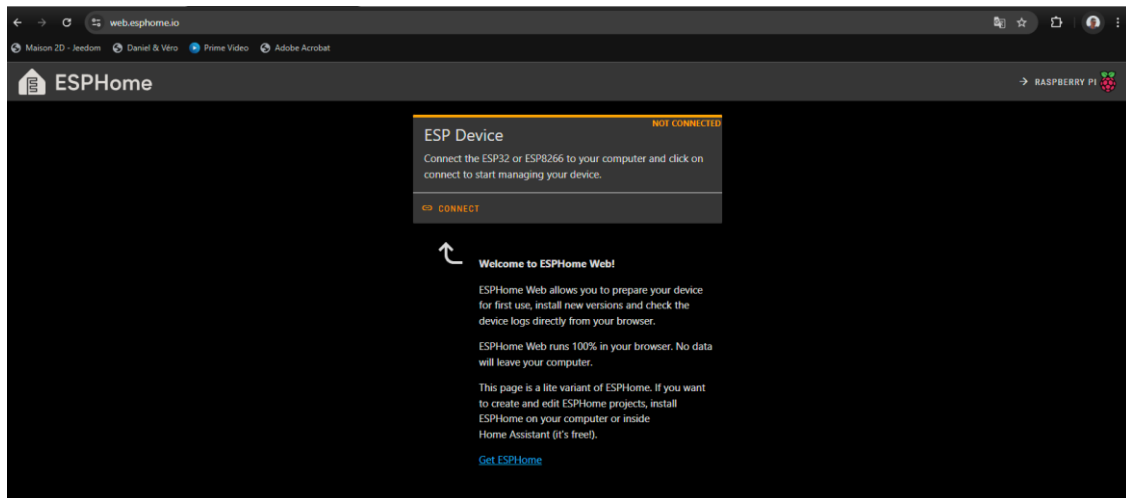


Figure 11 - Page d'accueil du site web.esphome.io

Connecter l'ESP32 en USB sur l'ordinateur, puis se rendre sur le site <https://web.esphome.io> (avec Chrome ou Edge). Cliquer sur **CONNECT**. Sélectionner le port USB correspondant, puis sur **Connexion**.

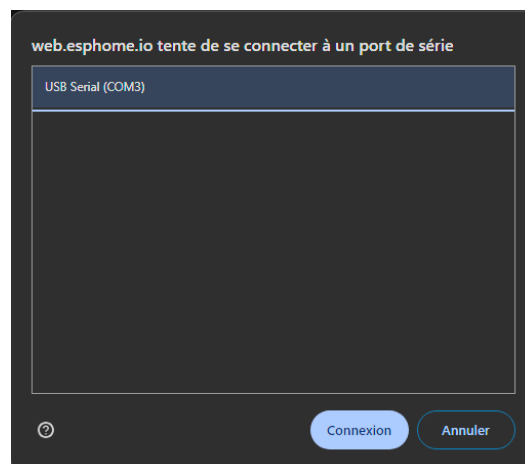


Figure 12 - Connexion de l'ESP32

Cliquer sur **INSTALL**,

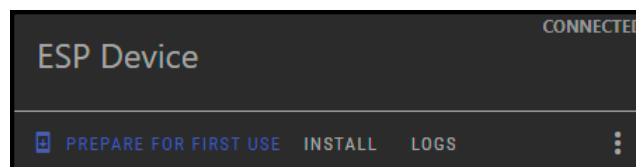


Figure 13 - ESP connecté et prêt

Puis sélectionner le fichier .bin sur l'ordinateur (**Choisir un fichier**) et cliquer sur **INSTALL**.

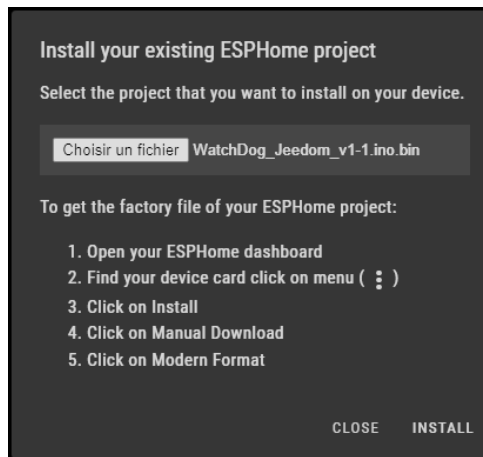


Figure 14 - Fichier .bin sélectionné et prêt pour le téléversement

Le téléversement sur l'ESP32 débute.

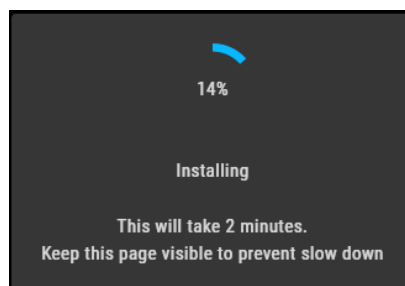


Figure 15 - Téléversement en cours

Et s'il n'y a pas eu d'erreurs, celui-ci se termine avec ce message :

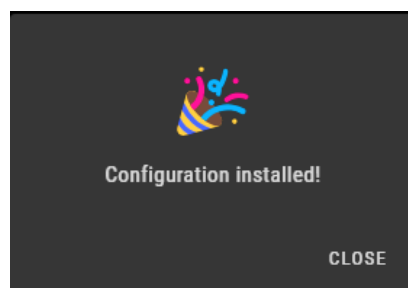


Figure 16 - Fin avec succès du téléversement

Mises à jour par OTA

Après un premier téléversement initial par la prise série (USB), le module peut-être mis à jour par la méthode OTA ('Other The Air', via le réseau Wifi) soit avec un serveur web dédié, accessible en fonctionnement courant en permanence sur l'adresse IP dédiée au module JWS, soit avec l'IDE Arduino par recompilation du code source et envoi via le réseau Wifi.

Procédure avec le serveur web

Se connecter sur la page web à l'@IP du module JWS :

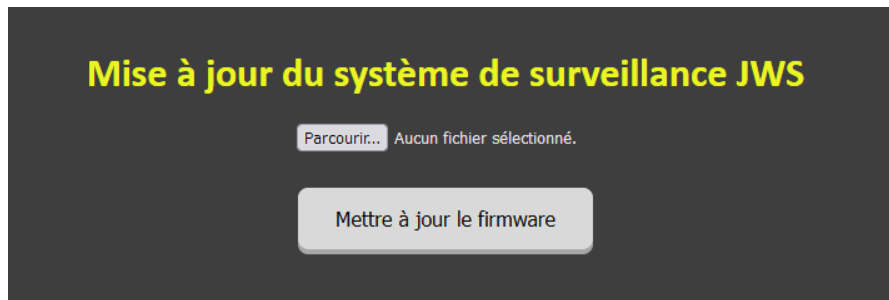


Figure 17 - Serveur web pour le téléversement du firmware

Cliquer sur **Parcourir...** et sélectionner le fichier binaire du firmware .bin

Puis cliquer sur **Mettre à jour le firmware** pour commencer la procédure de mise à jour.

Après un téléversement effectué avec succès, ce message doit s'afficher dans le navigateur.

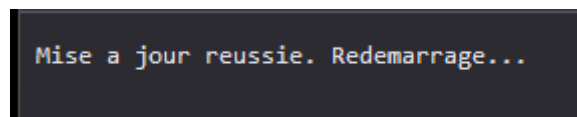


Figure 18 - Mise à jour effectuée avec succès

Procédure avec l'IDE Arduino

Lancer l'IDE Arduino, et sélectionner le module ESP32 dev et le port Wifi désigné par le nom **JWS_ESP32**.

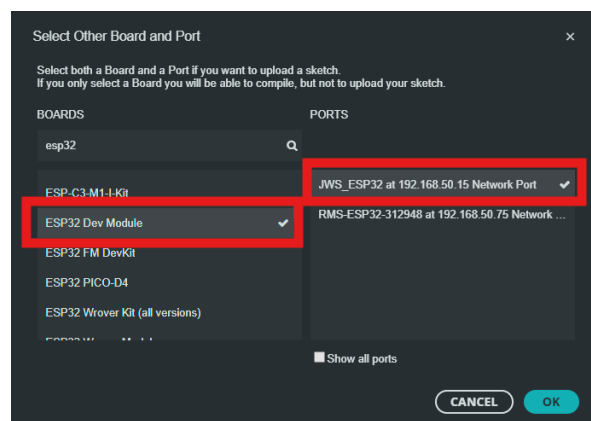


Figure 19 - Mise à jour OTA avec l'IDE Arduino

Lancer une compilation, le téléversement s'effectuera automatiquement vers l'ESP32 à l'issue à la condition qu'il n'y ait pas eu d'erreur pendant le processus.

Au premier téléversement, l'IDE Arduino demandera un mot de passe. Il n'y a aucun mot de passe défini, il suffit d'entrer n'importe quelle chaîne de caractère de n'importe quelle longueur.

Affichage pendant les mises à jour

Quel que soit la méthode utilisée, pendant le téléversement l'écran OLED affiche un message spécifique et la LED rouge va clignoter rapidement au rythme de la réception des données indiquant que la procédure est en cours.



Figure 20 - Mise à jour en cours

En fin de mise à jour, la LED verte va s'illuminer et s'éteindre progressivement trois fois consécutivement pour indiquer qu'il n'y a pas eu d'erreur pendant le téléversement.

Dans le cas contraire, c'est la LED rouge qui va s'illuminer et s'éteindre progressivement trois fois pour signaler une erreur qui s'est produite pendant le téléversement, avec l'affichage d'un message d'erreur spécifique sur l'écran OLED.

Le module redémarrera à l'issue.

Les paramètres enregistrés dans la mémoire EEPROM ne sont pas affectés par les mises à jour.

RAZ du module

Le module peut-être remis dans sa configuration initiale en effectuant une RAZ de l'ensemble des paramètres enregistrés dans la mémoire EEPROM interne.

Pour ce faire, il faut connecter au démarrage du module la broche 13 à la masse (elles sont situées côte à côte).

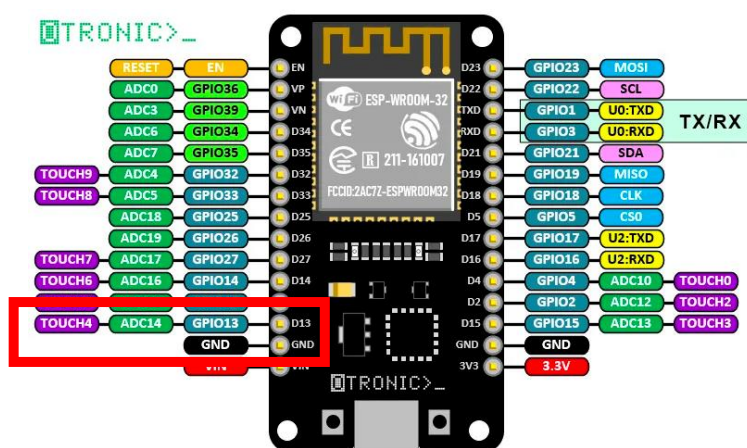


Figure 21 - Les broches GPIO13 et GND sur l'ESP32

Après l'effacement, l'ESP va repartir automatiquement sur l'établissement d'un réseau en mode AP pour la définition des nouveaux paramètres de connexion.

Note

Dans la mesure où cette fonction n'est normalement pas fréquemment utilisée, il n'est pas prévu de bouton poussoir permettant d'assurer cette fonction.

Configuration de Jeedom

Pour assurer le dialogue sur le réseau MQTT avec le module JWS, un broker MQTT (Mosquito par exemple) doit être installé, configuré et démarré pour activer le réseau et renvoyer une réponse aux messages ping reçus périodiquement.

Il faut créer un équipement **Auto-surveillance**, avec le plugin qui gère le protocole MQTT (MQTT Manager par exemple), en définissant pour le topic racine la valeur **ping**.

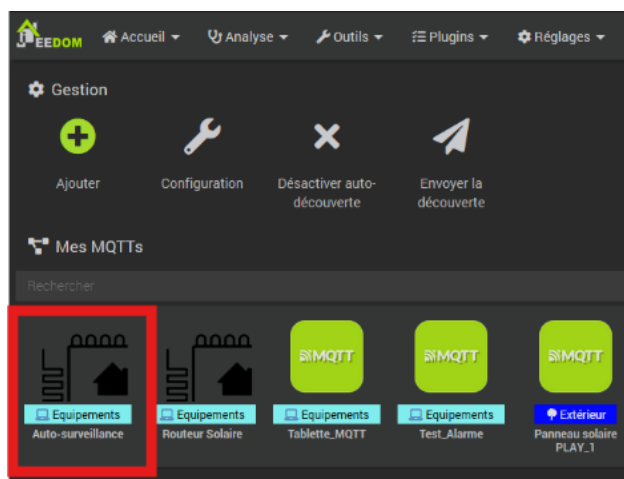


Figure 22 - Création de l'équipement Auto-surveillance

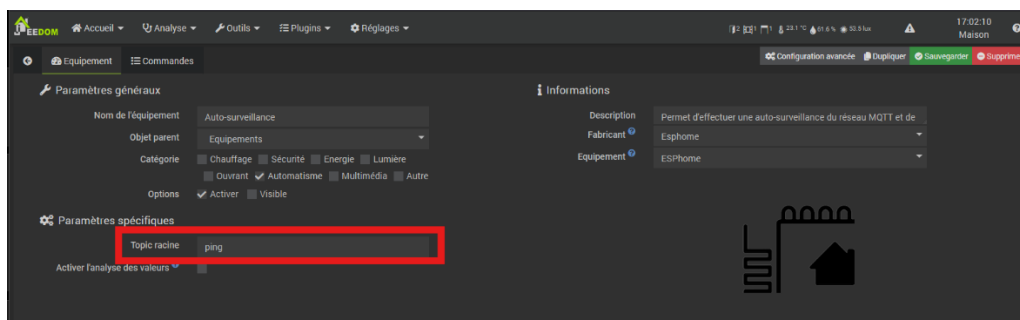


Figure 23 - Définition du topic racine

Dans l'onglet Commandes, créer deux commandes : **Ex_Pong** (type action, topic Rx, valeur = pong) et **Rx_Ping** (type info, topic Ex).

Ces commandes seront configurées comme suit :

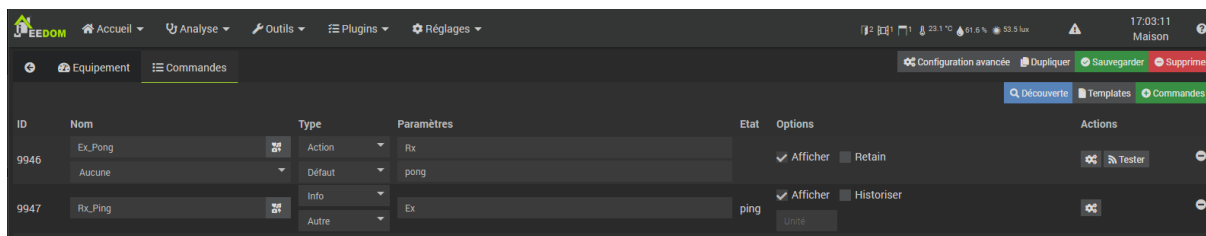


Figure 24 - Définition des commandes

Dans la configuration avancée pour l'information `Rx_Ping`, configurer une action sur la valeur reçue `ping` pour émettre le message `pong`.

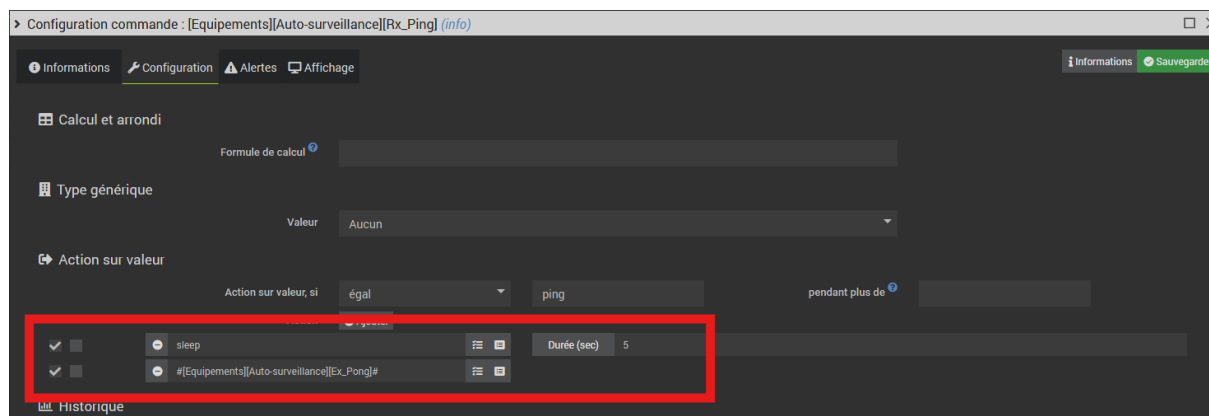


Figure 25 - Action à exécuter sur réception d'un ping

Astuce

Le délai d'attente `sleep` permet de ne renvoyer la réponse `pong` vers le module JWS qu'après l'expiration de ce délai. Celui-ci va donc détecter et signaler le test défaillant par la diode LED bleue et l'affichage de l'écran OLED pendant cet intervalle, pour les éteindre dès que la réponse sera reçue, provoquant un flash plus ou moins furtif. Ce flash permet ainsi de confirmer toutes les 60 secondes l'activité du module JWS. Un délai fixé à 5 secondes provoque un flash de la LED bleue pendant à peu près 2 secondes.

Causes probables des erreurs résultants lors des tests

Erreurs relevées lors du cycle de tests	Causes probables
Pas de ping	La box domotique est déconnectée du réseau domestique. La box domotique n'est pas démarrée.
Pas de connexion MQTT	Le serveur MQTT (Mosquito ou autre) est stoppé. La box domotique n'est pas démarrée.
Pas de réponse aux messages MQTT	Le démon du plugin gérant les connexions MQTT (MQTT Manager, ZigbeeLinker,...) est stoppé, L'équipement 'Auto-Surveillance' n'est pas activé, L'émission du message 'pong' n'est pas activé. La box domotique n'est pas démarrée.

Liens pour l'achat du matériel nécessaire

Matériel	Nb	Lien	Prix
ESP32 standard CH340C USB-C avec son support 30 broches	1	AliExpress	7,49€
Module relais 5v 1 canal	1	AliExpress	1,39€
Ecran OLED de 128x64 pixels, bus I2c	1	AliExpress	2,54€
LED RGB 5mm anode commune	1	AliExpress	1,99€ (x50)
Résistance 220 ohms 1/4W	3	AliExpress	0,94€ (x220)
Prises USB-C châssis 6 contacts CR-19	2	AliExpress	2.11€ (x5)
Câble USB-C coudé type C2C 0,25m	1	AliExpress	2.05€
Boitier (dimensions ext/int : 125(120) x80(75) x32(27) mm)	1	AliExpress	2,55€
Quincaillerie (vis + écrous Phillips acier noir M1,6 / 2,5 / 3)	14	AliExpress	15,23€ (x1500)
Gaine thermo rétractable 1,5mm	-	AliExpress	1,89€
Câblage 22-24 AWG	-	AliExpress	1,27€ (x5m)
Feuille acrylique translucide 10x10x1mm coloris 'Light black'	1	Ali Express	3,91€ (x2)
Total			17.58€ à 43,36€

Note

Le matériel présenté sur les lignes avec est soit **optionnel**, soit pouvant être **récupéré** sur stock



Figure 26 - Le module JWS

Réalisation des prises USB

Les prises USB-C châssis utilisées doivent comporter 6 broches au minimum :

- VBUS (+5v),
- GND (masse),
- CC1, CC2 (canal de configuration),
- D+, D- (données).

Les signaux CC1 et CC2 (Configuration Canal 1&2) permettent d'indiquer à l'alimentation la présence d'une charge sur la prise USB-C. Ces deux broches sont en principe reliées à la masse par une résistance de 5,1Kohms, conformément aux spécifications de la norme USB. Sans ces signaux, la prise ne délivrera aucun courant.

Les signaux D+ et D-, utilisés normalement dans le transfert de données, vont être utilisés ici par la box domotique pour la négociation avec l'alimentation de la délivrance d'un courant jusqu'à 3A max. Sans ces broches, l'alimentation ne fournira par défaut qu'une tension de 5v sous 1A maximum, ce qui est notamment insuffisant pour permettre l'alimentation d'une box domotique.



Figure 27 - Brochage des prises USB-C châssis

Les signaux CC1, CC2, D+, D- ne sont pas utilisés par le module JWS, et doivent être connectées en vis-à-vis. Le module JWS ne sera donc pas alimenté s'il n'y a pas de box domotique branchée sur la prise USB-C OUT.

En conséquence, les prises USB nécessitent la soudure de 6 fils, dont les deux fils d'alimentation (GND et VBUS) doivent être conformes au minimum à la norme 24 AWG ou plus (soit au minimum un diamètre de 0,5 mm pour une section de 0.205 mm²), pour pouvoir véhiculer le courant de 3A maximum sous 5V nécessaire à la box domotique.

L'alimentation en propre du module JWS est assurée par la même alimentation que la box domotique, via les broches VBUS et GND. Seule la tension VBUS (+5v) sera coupée par le relais lors des redémarrages de la box domotique.

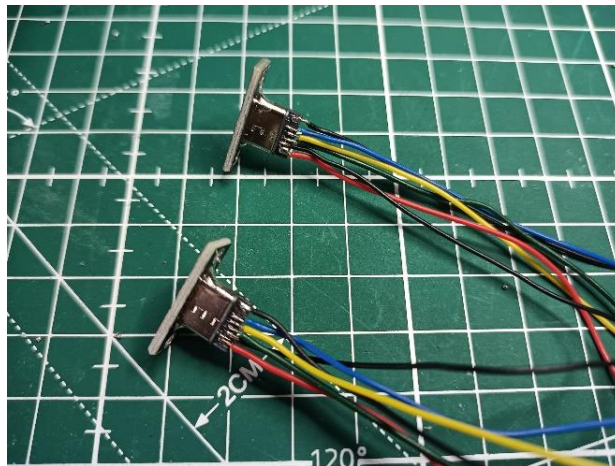
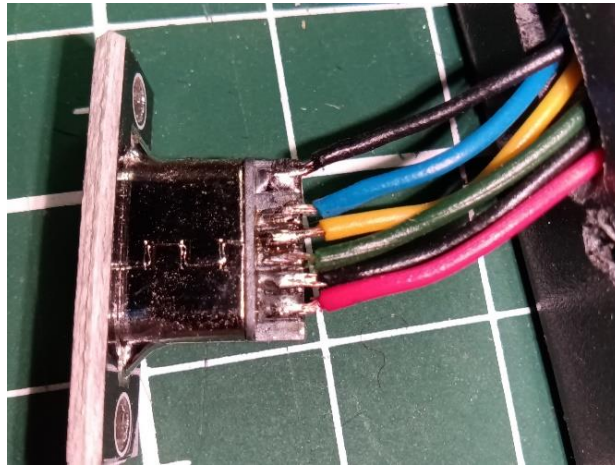


Figure 28 - Détails du câblage des prises USB-C

Montage final sur le boîtier, avec une fixation sur la demi-coque inférieure du boîtier afin de faciliter les opérations d'ouverture.



Figure 29 - Installation à leur emplacements finaux des prises USB-C

Montage de la LED RGB

La LED RGB sera soudée avec ses trois résistances de limitation en série sur les broches R, G et B. Elle est du type Anode Commune (AC), et donc pilotée par un signal bas sur ses broches RGB. Elle peut être soit de type claire (pour une haute luminosité directive), soit opaque (avec une lumière plus diffuse).

Le type de LED utilisée (AC ou CC) peut-être néanmoins être paramétré dans le code source en commentant la ligne ad-hoc :

```
#define Type_LED_AC    1           // Définition du type de LED Anode Commune (AC)
// #define Type_LED_CC  1           // Définition du type de LED Cathode Commune (CC)
```

Note

Les fichiers binaires disponibles sur Github sont compilés par défaut pour une LED de type Anode Commune.

Les résistances de limitation peuvent avoir une valeur comprise entre 180 et 270 ohms (la LED étant alimentée en 3,3v), 220 ohms étant la valeur recommandée. L'utilisation de gaine thermo-rétractable sur chaque broche permettra un montage sans risque de courts-circuits intempestifs.

Tout comme les prises USB, pour faciliter l'ouverture le montage de la LED s'effectuera sur la partie inférieure du boîtier.

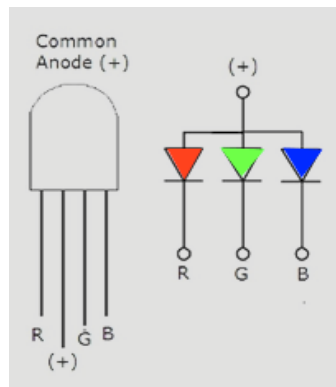


Figure 30 - brochage d'une LED RGB à Anode Commune

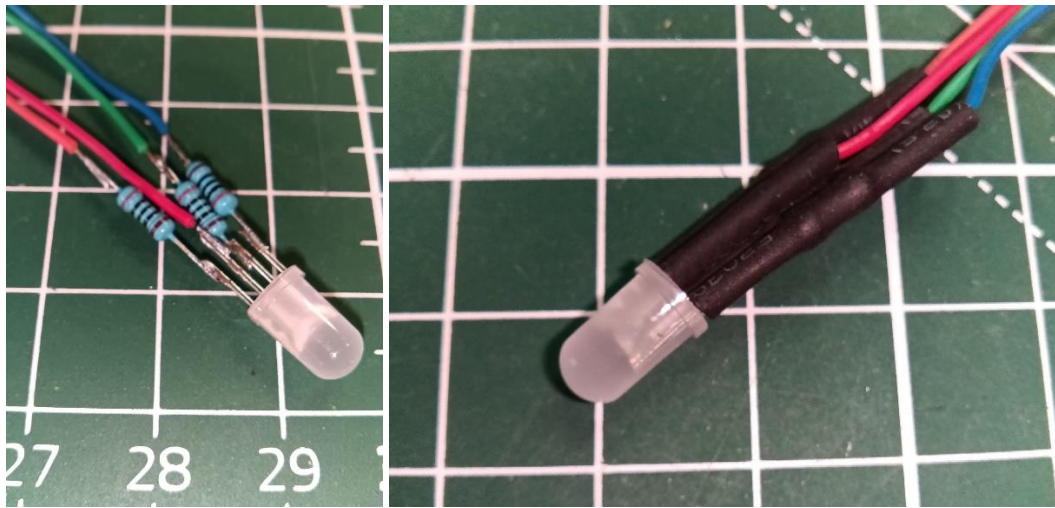


Figure 31 - Détails du montage des résistances et des gaines thermo-rétractables

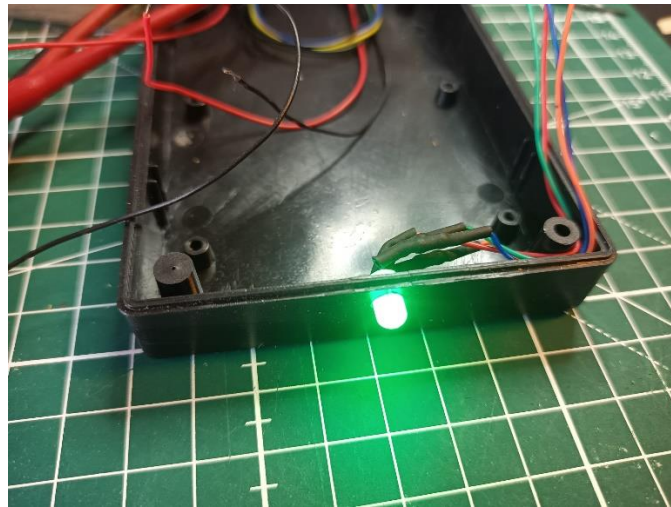


Figure 32 - Montage de la LED RGB dans le boîtier

Montage de l'ESP avec son support et de la platine relais dans le boîtier

Pour le montage, l'ESP avec son support et la platine relais seront fixés à l'aide des ergots prévus pour les fixations au fond de ce modèle de boîtier. A ce stade, on peut procéder au câblage entre les éléments.

L'écran OLED quant à lui, sera fixé avec 4 boulons (vis de 2,5 mm) sur la demi-coque supérieure du boîtier après l'ouverture d'une fenêtre de taille suffisante.

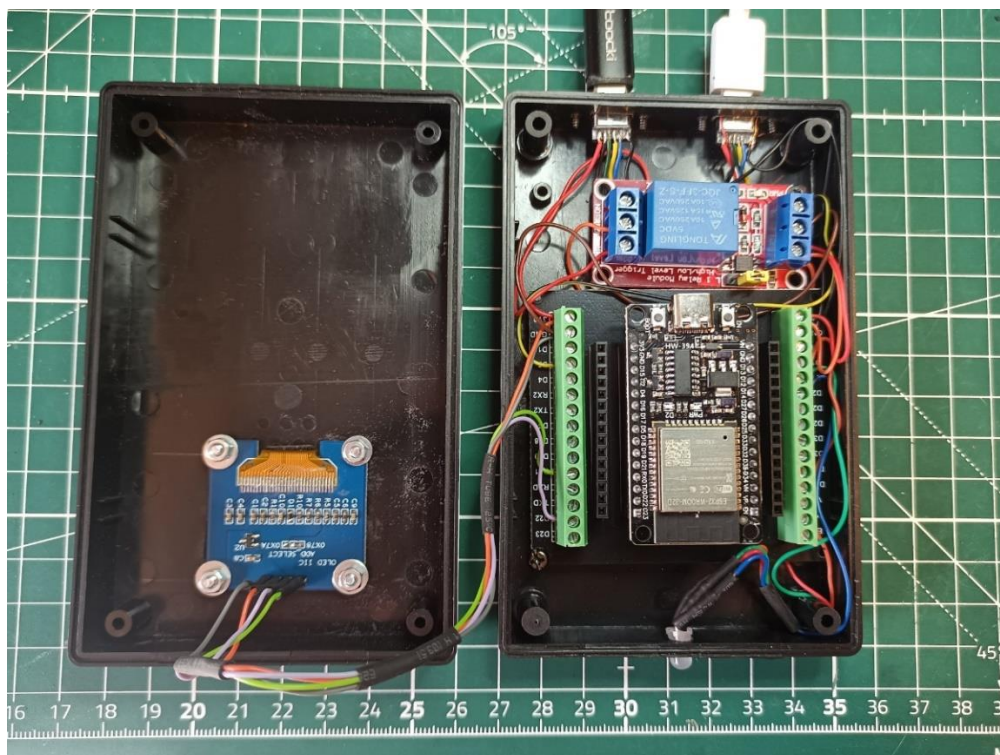


Figure 33 - Montage de l'ESP, de la platine relais et de l'écran OLED

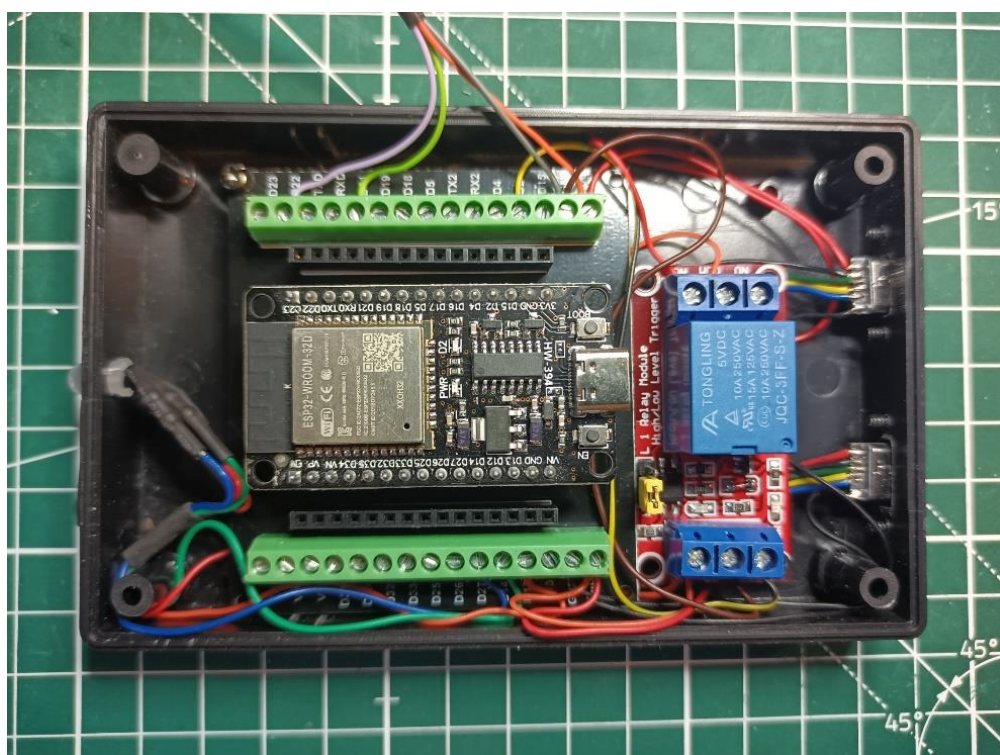


Figure 34 - Détails du montage de l'ESP, de la platine relais, la LED RGB, les prises USB-C et le câblage



Figure 35 - Découpe d'une fenêtre pour l'écran OLED sur la partie supérieure du boîtier

Finitions

La finition du boîtier pourra s'effectuer en fixant une plaque de 1mm d'épaisseur en acrylique translucide de couleur noire fumée, fixée par des vis noires et écrous de 1,6mm. Cette plaque occultera efficacement l'écran OLED avec ses fixations (des vis plates de couleur noire de 2,5 mm) ainsi que l'ouverture pratiquée sur le boîtier, tout en conservant une lisibilité optimale.

Ce sera également l'occasion de placer une étiquette d'identification, et de repérer les prises USB IN et OUT (décalcomanies). Un léger vernis sera appliqué pour fixer et protéger les indications, et obtenir une finition brillante.



Schéma électronique

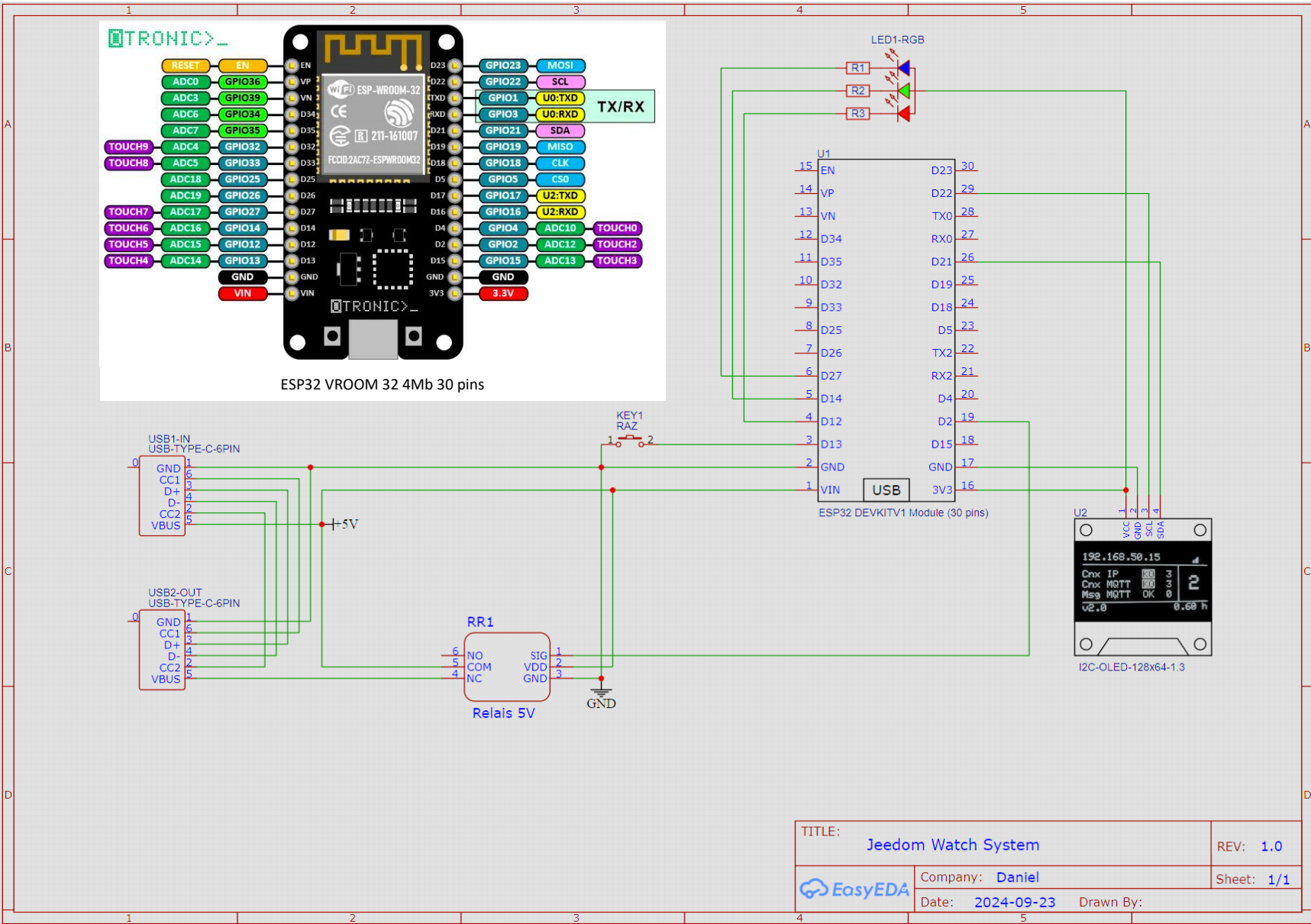


Table des figures

Figure 1 - Matériels utilisés	4
Figure 2 - Copie d'écran sur un smartphone du serveur web en mode AP	6
Figure 3 - Logo de démarrage	6
Figure 4 - Page principale	6
Figure 5 - Structure de l'affichage	7
Figure 6 - Informations sur les tests	7
Figure 7 - RESET de la box	8
Figure 8 - Plus aucun redémarrage n'est possible (compteur à 0)	9
Figure 9 - Mode maintenance	10
Figure 10 - Sélection du mode à partir du serveur web	10
Figure 11 - Page d'accueil du site web.esphome.io	11
Figure 12 - Connexion de l'ESP32	11
Figure 13 - ESP connecté et prêt	11
Figure 14 - Fichier .bin sélectionné et prêt pour le téléversement	12
Figure 15 - Téléversement en cours	12
Figure 16 - Fin avec succès du téléversement	12
Figure 17 - Serveur web pour le téléversement du firmware	13
Figure 18 - Mise à jour effectuée avec succès	13
Figure 19 - Mise à jour OTA avec l'IDE Arduino	13
Figure 20 - Mise à jour en cours	14
Figure 21 - Les broches GPIO13 et GND sur l'ESP32	14
Figure 22 - Création de l'équipement Auto-surveillance	15
Figure 23 - Définition du topic racine	15
Figure 24 - Définition des commandes	16
Figure 25 - Action à exécuter sur réception d'un ping	16
Figure 26 - Le module JWS	18
Figure 27 - Brochage des prises USB-C châssis	19
Figure 28 - Détails du câblage des prises USB-C	20
Figure 29 - Installation à leur emplacements finaux des prises USB-C	20
Figure 30 - brochage d'une LED RGB à Anode Commune	21
Figure 31 - Détails du montage des résistances et des gaines thermo-rétractables	22
Figure 32 - Montage de la LED RGB dans le boîtier	22
Figure 33 - Montage de l'ESP, de la platine relais et de l'écran OLED	23
Figure 34 - Détails du montage de l'ESP, de la platine relais, la LED RGB, les prises USB-C et le câblage	23
Figure 35 - Découpe d'une fenêtre pour l'écran OLED sur la partie supérieure du boîtier	24
Figure 36 - Le boîtier finalisé	25

Projet conçu et réalisé par J. Daniel ©

11 octobre 2024

Les sources, fichiers binaires et documentation sont disponibles en libre téléchargement ou consultation sur GitHub :

<https://github.com/jnqdan/JWS>