Buzzer Passiu

Buzzer, problemes amb els tons (notes), s’han eliminat les notes que donen errors per sortida de rang. S’ha creat un diccionari amb notes preestablertes amb les seues freqüències i un diccionari on es recullen les melodies preestablertes amb les notes de la melodia

Comunicació entre Raspberry Pi 3 i Raspberry Pi pico

Es va iniciar la comunicació amb SPi per la “senzillesa” en la codificació i la velocitat de transmissió. Després de dos dies de feina intensiva no es va aconseguir que funcionés bidireccionalment de cap manera, ni individualment, és a dir, només una de les dues plaques envia a l’altra i l’altra ho llegeix.

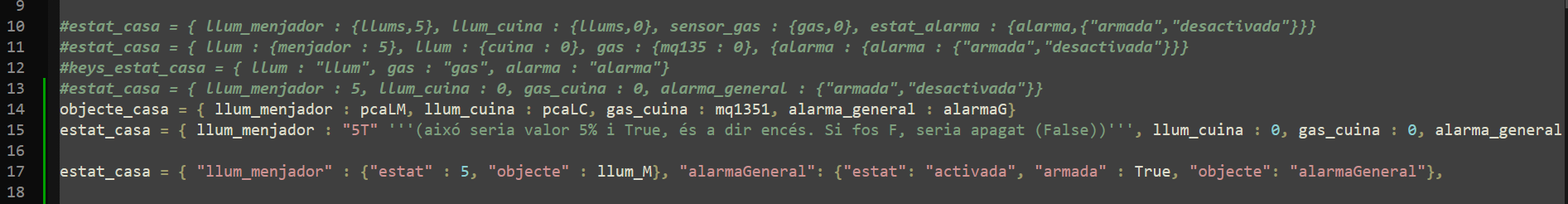
Es decideix canviar a port sèrie TX/RX (UART) perquè, en teoria, és més senzill de configurar/programar. Després d’altres dos dies intensius, s’aconsegueix la bidireccionalitat en diferit, és a dir, que una placa enviï i l’altar rebi. Però no és estable i dóna molts errors.

Llavors, es decideix recuperar la idea inicial d’usar WIFI i en unes hores s’aconsegueix una comunicació estable.

Tot i així, dóna errors i es dedica un dia sencer a estabilitzar la comunicació. S’arriba a una possible conclusió que el que passa quan es queda, aparentment, bloquejada la WIFI que és possible que trigui massa en restablir-se la comunicació i sembli que no està funcionant quan s’està restablint automàticament. En tot cas, però, durant la investigació de resolució d’errors, es troba una manera de reiniciar el mòdul hardware mitjançant codi i s’implementa per a assolir una reconnexió molt més ràpida d’uns pocs segons en comptes de temps al voltant del minut.

Una vegada estabilitzada, es procedeix a la realització de la classe. Com es vol usar una única classe per ambdues plaques, es genera un mòdul detector de placa amb l’ajut de la IA i es procedeix a usar-lo per a realitzar la classe única. Tot i poder ser més eficients fent una classe on ho fes i poder-ho cridar d’es de qualsevol classe, s’ha fet, en una primera instància, internament a la classe.

Es realitzarà amb diccionaris la comunicació, enviats en text pla com una string i descodificats i transformats amb regeix en diccionaris al destí. Finalment, després d’analitzar diverses maneres de fer els diccionaris, es decideix usar diferents diccionaris, un estàtic on es registrarà la referència a l’objecte de cada element que serà fix un cop es s’executi el codi i específic per a cada placa. I un altre diccionari que serà el dinàmic on registrarà l’estat de cada element, els quals seran actualitzats segons les circumstàncies del dia a dia de la casa i, serà la part que s’anirà comunicant entre plaques.



Exemple de diverses versions descartades

Baglock:

* Acabar part sensors IR
* Gestionar error per manca d’alimentació del mòdul PCA
* Gestionar error per manca d’alimentació del mòdul X
* Fer redundant el sistema per si hi ha falles i/o un sistema analògic/mecànic per a funcionament manual
* Sectoritzar i separar les funcions en ESPs32VROOM diferents
* CORREGIR (o revisar la totalitat de l alògica) els mètodes o funcions per a que no realitzin accions en la fase de captació de valors de sensors i que aquesta cció es faci en la dels actuadors en funció del valor del diccionari q s’ha actualitzat amb la fase dels sensors
* AFEGIR entrada de SSID i PASSWORD a les classes WIFI, q ara mateix estan fixes i no com a paràmetre
* Els sensors magnètics REED manca ser llegits pel chip multiplexor
* Els sensors d’InfraRoig IR manca ser llegits pel chip multiplexor
* Implementar un Display a la raspberry i/o al ESP32
* Implementar una web per al maneig dels estats remotament (per l’sesp32 (o rasp)) (veure el projecte de l’oscar
* Usar fils d’execució, per exemple en l’enviament un, en el revçbut un altre i el programa
* Un mapa, tkinter e.g., del estat de les coses
* Investigar codis preestabler dels protocols domòtics: MQTT o ZIGBEE protocolos domoticos