

Informe de la Pràctica 2: Interrupcions

1. Introducció

L'objectiu principal d'aquesta pràctica és comprendre el funcionament de les interrupcions en un microcontrolador ESP32 a través de dues parts diferenciades:

- **Interrupció per GPIO:** en aquesta part, es configura un sistema en què un botó connectat a un pin GPIO genera una interrupció en ser premut, alterant l'estat del sistema de manera immediata.
- **Interrupció per temporitzador:** en aquesta part, es configura un temporitzador que genera interrupcions periòdiques, permetent executar tasques de manera regular sense la necessitat de bucles actius.

L'ús d'interrupcions permet millorar l'eficiència del sistema evitant la necessitat de comprovar constantment l'estat dels pins (polling) i optimitzant el rendiment del microcontrolador. Aquesta tècnica és fonamental en aplicacions en temps real on es requereix una resposta immediata als esdeveniments.

2. Fonaments Teòrics

Una interrupció és un senyal que interromp l'activitat normal del microprocessador i salta a executar una funció especial anomenada Interrupt Service Routine (ISR). Quan la ISR finalitza, el processador torna a l'estat anterior i continua amb l'execució del codi.

Les interrupcions poden ser generades per:

- Un esdeveniment de maquinari (ex. canvi d'estat d'un pin GPIO).
- Un temporitzador programat (Timer).
- Una crida per programari (no suportada per Arduino ESP32).

El microcontrolador ESP32 permet definir interrupcions per a qualsevol dels seus pins GPIO utilitzant la funció `attachInterrupt(GPIOPin, ISR, Mode)`, on:

- **GPIOPin:** defineix el pin que generarà la interrupció.
- **ISR:** funció que s'executarà quan es produeixi la interrupció.
- **Mode:** defineix el moment en què es dispara la interrupció (**LOW**, **HIGH**, **CHANGE**, **FALLING** o **RISING**).

També es poden configurar interrupcions per temporitzadors mitjançant `timerAttachInterrupt()`, associant-les a una funció ISR.

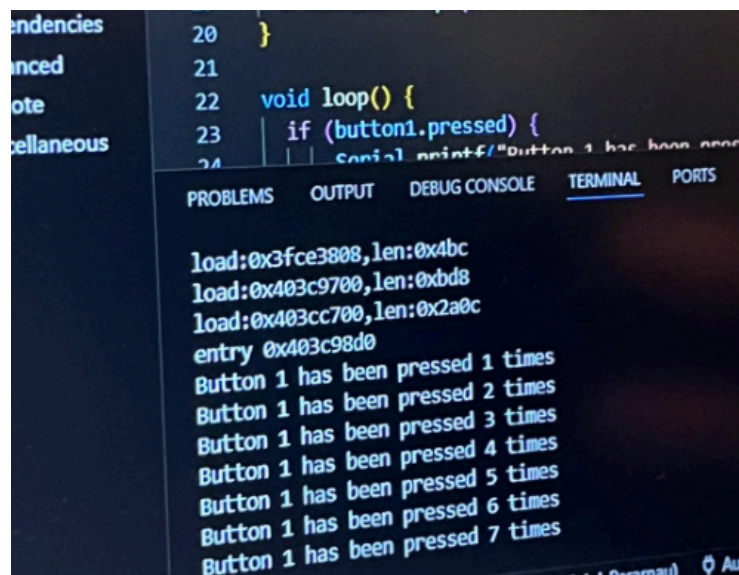
3. Desenvolupament de la Pràctica

3.1 Interrupció per GPIO

En aquesta part de la pràctica, s'ha implementat un sistema en el qual un botó connectat a un pin GPIO genera una interrupció quan és premut. L'ISR associat incrementa un comptador de pulsacions i estableix una variable booleana per indicar que el botó ha estat premut.

Resultats:

Cada vegada que el botó es prem, es genera una interrupció i es mostra el nombre total de vegades que ha estat premut a través del monitor sèrie.



```
20 }
21
22 void loop() {
23     if (button1.pressed) {
24         Serial.printf("Button 1 has been pressed %d times\n", count);
25     }
26 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

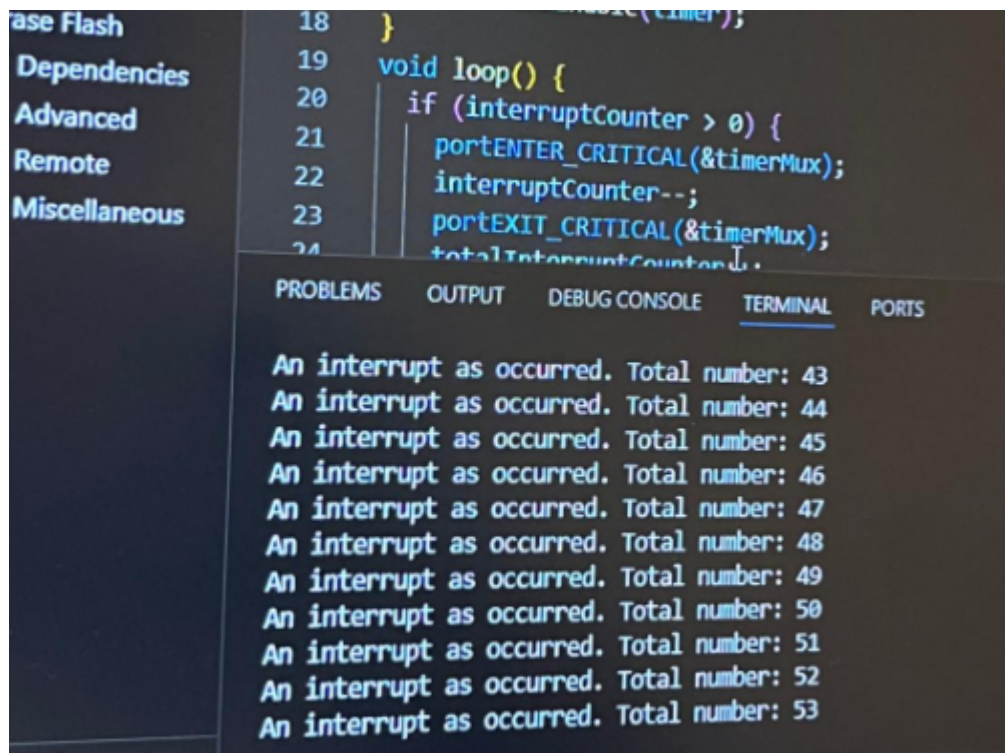
```
load:0x3fce3808,len:0x4bc
load:0x403c9700,len:0xbd8
load:0x403cc700,len:0x2a0c
entry 0x403c98d0
Button 1 has been pressed 1 times
Button 1 has been pressed 2 times
Button 1 has been pressed 3 times
Button 1 has been pressed 4 times
Button 1 has been pressed 5 times
Button 1 has been pressed 6 times
Button 1 has been pressed 7 times
```

3.2 Interrupció per Temporitzador

En aquesta part de la pràctica, s'ha implementat un temporitzador que genera una interrupció periòdica per portar el recompte d'interrupcions generades.

Resultats:

Cada segon, el temporitzador genera una interrupció que incrementa un comptador i mostra per monitor sèrie el nombre total d'interrupcions generades.



```
18 }
19 void loop() {
20   if (interruptCounter > 0) {
21     portENTER_CRITICAL(&timerMux);
22     interruptCounter--;
23     portEXIT_CRITICAL(&timerMux);
24     totalInterruptCounter++;
25   }
26 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

An interrupt as occurred. Total number: 43
An interrupt as occurred. Total number: 44
An interrupt as occurred. Total number: 45
An interrupt as occurred. Total number: 46
An interrupt as occurred. Total number: 47
An interrupt as occurred. Total number: 48
An interrupt as occurred. Total number: 49
An interrupt as occurred. Total number: 50
An interrupt as occurred. Total number: 51
An interrupt as occurred. Total number: 52
An interrupt as occurred. Total number: 53

4. Anàlisi de Resultats

L'ús d'interrupcions permet millorar l'eficiència en la gestió d'esdeveniments en un microcontrolador ESP32:

- En la interrupció per GPIO, el microcontrolador pot realitzar altres tasques mentre espera que es premi el botó, evitant l'ús de `delay()` i millorant la resposta del sistema.
- En la interrupció per temporitzador, es genera un senyal periòdic sense necessitat d'utilitzar un bucle actiu, fent el codi més eficient i precís.

Aquestes tècniques són fonamentals per al desenvolupament de sistemes de temps real, on és necessari reaccionar a esdeveniments de manera immediata.

5. Conclusions

Aquesta pràctica ha permès comprendre la importància i el funcionament de les interrupcions en sistemes embeguts. Les interrupcions per GPIO permeten reaccionar de manera eficient als canvis d'estat d'un pin sense necessitat de comprovar-lo contínuament. D'altra banda, els temporitzadors proporcionen una forma precisa de generar interrupcions periòdiques sense consumir recursos del processador innecessàriament.

Aquest enfocament és especialment útil en aplicacions en temps real, com sistemes de control, automoció o comunicacions, on la rapidesa de resposta és crítica.