

Informe de la Pràctica 4: Sistemes Operatius en Temps Real

1. Introducció

L'objectiu principal d'aquesta pràctica és comprendre el funcionament d'un sistema operatiu en temps real (RTOS) utilitzant l'ESP32 i el sistema FreeRTOS. En aquest context, s'explora com executar diverses tasques simultàniament aprofitant els dos nuclis del microcontrolador, cosa que és essencial en aplicacions IoT que requereixen gestió eficient dels recursos.

A través de diversos exercicis pràctics, s'analitza com es poden crear, executar i sincronitzar tasques, així com assignar prioritats i utilitzar semàfors per coordinar accions concurrents.

2. Fonaments Teòrics

2.1 Multitasca i FreeRTOS

FreeRTOS és un sistema operatiu en temps real per a microcontroladors que permet executar múltiples tasques simultàniament mitjançant un planificador. L'ESP32, amb doble nucli, és especialment adequat per aprofitar aquest entorn multitasca.

Conceptes clau:

- **Tasca:** Funció que s'executa independentment dins del sistema.
- **Prioritat:** Determina quina tasca té preferència quan diverses competeixen per la CPU.
- **vTaskDelay():** Fa una pausa en l'execució de la tasca, permetent que altres s'executin.
- **xTaskCreate():** Funció per crear una nova tasca.
- **xTaskCreatePinnedToCore():** Permet assignar una tasca a un nucli específic.
- **vTaskDelete():** Finalitza una tasca, alliberant recursos.
- **xPortGetCoreID():** Retorna l'identificador del nucli que executa una tasca.
- **Semàfors:** Utilitzats per sincronitzar tasques entre si.

2.2 Creació de Tasques

Es defineixen funcions que es convertiran en tasques independents. Per exemple, es crea una tasca que fa parpellejar un LED usant un bucle infinit amb `vTaskDelay()`. Aquesta estructura permet alternar l'execució entre tasques sense bloquejar el sistema.

2.3 Tasques Úniques

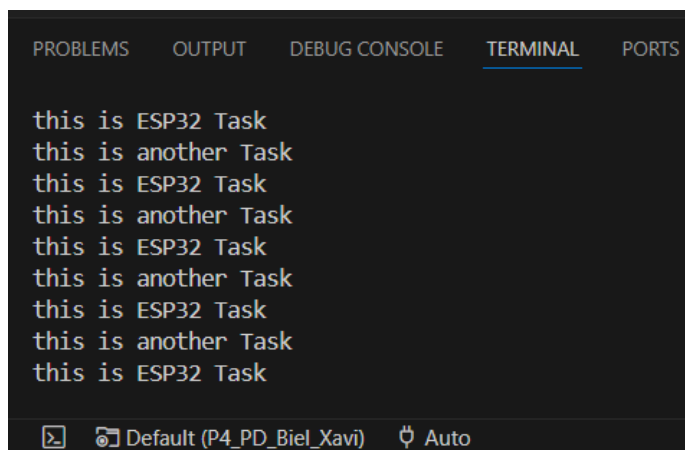
També es poden crear tasques que només s'executen una vegada. Aquestes es tanquen manualment amb `vTaskDelete(NULL)` per evitar que quedin actives innecessàriament.

3. Exercicis Pràctics

3.1 Exercici Pràctic 1

En aquest exercici es crea una nova tasca que s'executa en paral·lel amb la tasca principal d'Arduino (la funció `loop()`). Aquesta nova tasca mostra missatges per monitor sèrie de manera periòdica, alhora que el `loop()` també continua imprimint els seus propis missatges.

Sortida per monitor sèrie:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

this is ESP32 Task
this is another Task
this is ESP32 Task
this is another Task
this is ESP32 Task
this is another Task
this is ESP32 Task
this is another Task
this is ESP32 Task

[Icon] Default (P4_PD_Biel_Xavi) [Icon] Auto
```

Explicació:

Aquest programa crea dues tasques:

- La funció `loop()` representa la tasca principal del sistema.

- La tasca **anotherTask**, creada amb **xTaskCreate**, imprimeix un missatge diferent.

Ambdues tasques s'executen concurrentment i mostren missatges alternatius al monitor sèrie, aprofitant el planificador de FreeRTOS.

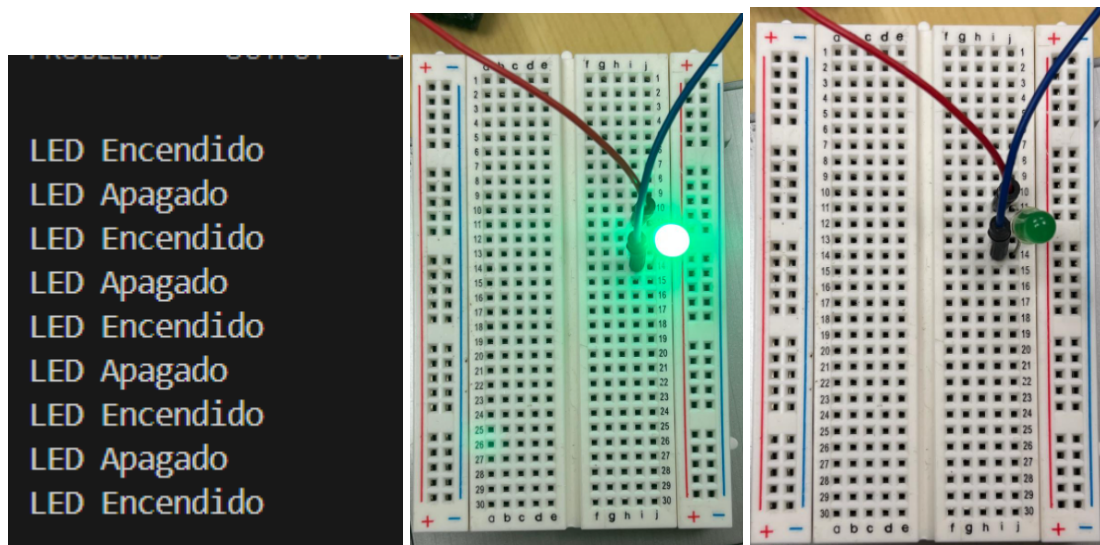
3.2 Exercici Pràctic 2

Aquest exercici ha consistit a controlar un LED mitjançant dues tasques i tres tasques, implementades en dues versions diferents:

Versió 1: Sense sincronització (sense semàfor)

En aquesta versió, es creen dues tasques:

- Una tasca **encén** el LED.
- L'altra tasca **l'apaga**.



Funcionament:

- Les dues tasques escriuen directament al pin del LED sense cap mecanisme de coordinació.
- Això provoca que el LED es comporti de forma poc estable i amb parpelleigs irregulars, ja que les dues tasques poden accedir al LED en qualsevol moment, produint **condicions de competència**.

Versió 2: Amb sincronització mitjançant semàfor

S'ha millorat el sistema afegint un **semàfor** per sincronitzar les dues tasques:

- La tasca que encén el LED només ho fa quan pot obtenir el semàfor.
- Un cop encès, allibera el semàfor perquè l'altra tasca pugui apagar el LED i repetir el cicle.

```
Tarea 2: LED Apagado  
Tarea 3: Esperando...  
Tarea 1: LED Encendido  
Tarea 2: LED Apagado  
Tarea 3: Esperando...  
Tarea 1: LED Encendido  
Tarea 2: LED Apagado  
Tarea 3: Esperando...  
Tarea 1: LED Encendido
```

Funcionament:

- El semàfor garanteix que només una tasca accedeixi al LED al mateix temps.
- El comportament del LED es torna **estable i controlat**, amb una alternança clara entre encès i apagat.

Aquesta versió mostra la importància dels **mecanismes de sincronització** per evitar comportaments inesperats i garantir un funcionament correcte en sistemes multitasca.

4. Anàlisi de Resultats

S'ha pogut verificar com FreeRTOS permet executar múltiples tasques en paral·lel a l'ESP32, millorant la resposta del sistema i aprofitant els recursos de forma eficient. A més, la utilització de prioritats i semàfors permet dissenyar sistemes robustos i deterministes.

5. Conclusions

Aquesta pràctica ha permès familiaritzar-se amb el concepte de sistemes operatius en temps real i la seva aplicació amb FreeRTOS sobre l'ESP32. La capacitat de definir, prioritzar i sincronitzar tasques és essencial per a aplicacions complexes dins del món de l'IoT. La comprensió i ús correcte de FreeRTOS obre les portes a desenvolupar sistemes fiables, responsius i escalables.