**P 10: Temporizator, afișoare, serial**

* Utilizează un temporizator pentru a declanșa actualizarea celor două afișaje cu șapte segmente la fiecare 500 ms.
* Permite unui proces să gestioneze logica afișajului, folosind un grup de evenimente pentru a indica modificările stărilor de intrare (din butoane sau întrerupătoare).
* Un alt proces (pentru comunicarea serială) trebuie să trimită valoarea curentă pentru a fi afișată ori de câte ori aceasta se modifică.

**1. Inițializarea perifericelor (init\_perif)**

* Configurează GPIO-urile (General Purpose Input/Output) pentru:
  + **Afisajele cu 7 segmente**:
    - GPIO Bcd și Bcd\_st sunt utilizate pentru a controla cele două afișaje.
    - Sunt setate ca ieșiri (SetDataDirection la 0x0000).
  + **Comutatoarele (switch-uri)**:
    - GPIO SwitchGpio este configurat pentru a citi stările comutatoarelor.
    - Este setat ca intrare (SetDataDirection la 0xFFFFFFFF).
* Inițializează un **Event Group** pentru sincronizarea task-urilor.

**2. Task-uri și responsabilități**

**a. prvProc1 (Citirea stării switch-urilor)**

* Monitorizează constant starea comutatoarelor folosind XGpio\_DiscreteRead.
* Dacă starea unui comutator se schimbă, setează un **bit în Event Group** (EVENT\_SWITCH\_CHANGED) pentru a semnala evenimentul altor task-uri.
* Se execută periodic la fiecare 50 ms folosind vTaskDelay.

**b. prvTxTask (Generarea valorilor aleatorii și trimiterea în coadă)**

* La fiecare 500 ms:
  + Generează un număr aleatoriu între 0 și 99.
  + Scrie acest număr în coadă (xQueueSend) pentru a fi procesat de alte task-uri.
* Este utilizată pentru a simula o sursă de date.

**c. prvRxTask (Primirea și afișarea valorilor)**

* Așteaptă semnalizarea unui eveniment (EVENT\_VALUE\_UPDATED) pentru a citi date din coadă.
* Actualizează variabila dspUnsignedInt care controlează afişajele.
* Actualizeaza afisajele cu cifrele calculate.

**3. Temporizatorul (xTimer și vTimerCallback)**

* Este configurat să expire la fiecare 500 ms.
* Când expiră:
  + Setează un **bit în Event Group** (EVENT\_VALUE\_UPDATED) pentru a actualiza afișajele.
* Temporizatorul este utilizat pentru a coordona operațiunile periodice fără a bloca alte task-uri.

**4. Afișarea valorilor pe 7 segmente**

* Valorile afișate sunt codificate cu ajutorul unui tabel (segmentEncoding).
* Funcția **prvRxTask**:
  + Împarte valoarea de afișat (dspUnsignedInt) în cifrele unității (digit1) și zecilor (digit2).
  + Scrie aceste valori pe afișaje folosind XGpio\_DiscreteWrite.

**5. Sincronizarea și comunicarea între task-uri**

* **Coada (xQueue)**:
  + Permite transferul de date între prvTxTask (producător) și prvRxTask (consumator).
* **Event Group (xEventGroup)**:
  + Sincronizează task-urile pe baza evenimentelor:
    - EVENT\_SWITCH\_CHANGED: Schimbarea stării comutatorului.
    - EVENT\_VALUE\_UPDATED: Actualizarea afișajelor.

**6. Fluxul principal (main)**

* Inițializează perifericele.
* Creează task-urile (prvProc1, prvTxTask, prvRxTask).
* Creează coada și temporizatorul.
* Pornește planificatorul FreeRTOS (vTaskStartScheduler), care începe execuția task-urilor.

**7. Cum funcționează aplicația?**

1. **Inițializare**:
   * GPIO-urile sunt configurate pentru afișaje și comutatoare.
   * Temporizatorul este pornit pentru actualizarea afișajelor la fiecare 500 ms.
2. **Execuție continuă**:
   * prvProc1 monitorizează comutatoarele și semnalizează schimbările.
   * prvTxTask generează valori noi la fiecare 500 ms și le trimite în coadă.
   * prvRxTask citește valorile din coadă și le afișează, actualizând afișajele cu valorile schimbate.
   * Temporizatorul declanșează actualizări periodice pentru afișaje.
3. **Afișare și comunicare serială**:
   * Valorile actualizate sunt afișate pe cele două afișaje cu 7 segmente.
   * Mesajele sunt trimise prin UART pentru depanare.

**Funcționalitate completă:**

Acest cod implementează:

1. Actualizarea periodică a afișajelor cu 7 segmente.
2. Gestionarea evenimentelor (schimbări în starea comutatoarelor sau valori noi de afișat).
3. Comunicare serială pentru vizualizarea datelor în timp real.