TP programmation temps-réel, IF4

Séance 3, TP1, travail en salle PC

Sommaire.

1 Introduction

L'objectif de cette séance est de se familiariser avec les mécanismes multi-tâches temps réel offerts par *FreeRTOS* sur un microcontrôleur *ESP32*.

1.1 Configuration de l'environnement de développement

Référez-vous au document de préparation de séance disponible sur moodle

1.2 Compte rendu

Un compte rendu manuscrit est exigé, à déposer 1 semaine jour pour jour après la séance. Il vous sera utile pour le TP2, puis le jour de l'examen car des questions porteront sur ce que vous avez fait en TP. Les enseignants évalueront également la préparation de votre travail et votre avancement pendant la séance de TP.

2 Séance

Pour démarrer l'environnement de développement Arduino **v2** (les programmes ne fonctionneront pas avec la v1) :

- soit ouvrir une console (CTRL+ALT+T) puis taper **/opt/arduino-ide_2.3.2_Linux_64bit/arduino-ide**
- soit télécharger le fichier Arduino2.desktop parmi les « fichiers pour la séance de TP1 », l'enregistrer sur le bureau et double-cliquer dessus. S'il râle que le fichier n'est pas exécutable, cliquer sur « **Lancer quand même** ».

2.1 Tâches en concurrence

2.1.1 Premier essai

Programme à exécuter : **IF4_TP1_Q21a.ino**

1. Décrire succinctement ce qui est réalisé par le programme et le résultat obtenu à l'oscilloscope. Expliquer pourquoi on obtient ce résultat.

2. Ouvrir le moniteur série, que constatez-vous, pourquoi ?

2.1.2 2 tâches en concurrence, variante

Programme à exécuter : **IF4_TP1_Q21b.ino**

1. Quelle est la différence avec le programme précédent ?

2. Quel est l'impact sur le fonctionnement, visible via l'oscilloscope et le moniteur série ?

3. Comment éviter ce problème ?

4. Modifiez les priorités des 2 tâches pour les mettre à la même valeur.
   Que constatez-vous ?

2.2 Tâches périodiques

2.2.1 Première approche

Programme à exécuter : **IF4_TP1_Q22a.ino**

1. Quelles sont les différences avec le programme précédent ?

2. Quelles périodes obtient-on pour les tâches 1 et 2 ? Expliquez pourquoi.

3. Que se passe-t'il si on force les 2 tâches à tourner sur le même cœur ?

4. La période voulue est-elle respectée ? Est-ce une bonne approche pour obtenir des tâches périodiques ?

2.2.2 Deuxième approche

Programme à exécuter : **IF4_TP1_Q22b.ino**

1. Quelles sont les différences avec le programme précédent ?

2. Quelles périodes obtient-on pour les tâches 1 et 2 ? Expliquez pourquoi.

3. Que se passe-t'il si on force les 2 tâches à tourner sur le même cœur ?

4. Monter progressivement `nb_iterations` afin d'allonger le temps de travail des deux tâches.
   Que se passe-t'il ?

5. Quel critère permet d'anticiper ce problème ?

2.3 Synchronisation entre tâches

2.3.1 Principes

- **Sémaphore binaire** utilisé pour synchroniser deux tâches :

  une tâche attend que l'autre lui donne « le feu vert » pour démarrer

- **Mutex (exclusion mutuelle)** : sémaphore particulier utilisé spécifiquement pour protéger l'accès à une ressource partagée

  Il **permet l'héritage des priorités** : quand plusieurs tâches demandent à prendre un Mutex, la priorité du détenteur du Mutex est fixée momentanément à la valeur de la plus haute priorité parmi les tâches qui attendent sa libération. Cette technique a pour effet de prévenir les phénomènes d'inversion de priorité vus en cours à la première séance, même si cela ne garantit pas une sécurité infaillible face à ces phénomènes.

2.3.2 Expérimentation 1

Programme à exécuter : **IF4_TP1_Q23a.ino**

Expliquer ce que fait le programme et comment il le réalise.

2.3.3 Expérimentation 2

Programme à exécuter : **IF4_TP1_Q23b.ino**

1. Expliquer ce que fait le programme (utiliser l'oscilloscope et la console série).

2. Qu'est-ce qui ne fonctionne pas correctement ?

3. Enlevez les commentaires lignes 57-58, 65-66, 92-93, 100-101. Expliquez ce qu'il advient ?

2.3.4 Expérimentation 3

Programme à exécuter : **IF4_TP_Q23c.ino**

1. Expliquer ce que fait le programme

2. À la ligne 168, remplacer xSemaphore = xSemaphoreCreateBinary(); par xSemaphore = xSemaphoreCreateMutex();. Expliquer la différence ?

   Analyser la différence

2.4 Les interruptions

Programme à exécuter : **IF4_TP1_Q24.ino**

Expliquer ce que fait le programme (utiliser l'oscilloscope et la console série).

2.5 Communication entre tâches

Programme à exécuter : **IF4_TP1_Q25.ino**

1. Expliquer ce que fait le programme (utiliser l'oscilloscope et la console série).
2. Prenez le programme exemple n°7 de la page web ESP32+FreeRTOS[1].
   Faîtes le fonctionner, modifiez les paramètres des tâches et commentez.

2.6 Synchronisation

Proposez un programme qui organise un rendez-vous entre 3 tâches :

a. 1 première tâche exécute un travail ;
b. 1 deuxième tâche exécute en parallèle un autre travail ;
c. 1 troisième tâche doit attendre que les 2 premières aient fini pour démarrer.

indice : http://www.openrtos.net/uxSemaphoreGetCount.html et/ou xSemaphoreCreateCounting pour créer un sémaphore qui peut compter (non obligatoire)

1 cf. http://tvaira.free.fr/esp32/esp32-freertos.html

Commencez-le TP2 si vous avez terminé le TP1