ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

2A GPSE FISA – Année 2024-2025

SYSTÈMES TEMPS-RÉEL (TP)

Nom	Prénom	Table

Examen pratique

Durée : 1h30

Ressources autorisées :
- Tous documents
- Internet,
  - sauf IA générative (Chat GPT et équivalents)

Consignes

IMPORTANT : il y a plusieurs tâches et plusieurs fonctionnalités à implémenter. Ne cherchez pas à tout faire d'un coup ! Commencez par une fonctionnalité dans une tâche, puis la même fonctionnalité entre deux tâches, et ainsi de suite.

Appelez l'enseignant pour valider CHAQUE étape.

Ne restez pas bloqué trop longtemps : appelez l'enseignant pour avoir des indices ou corriger une erreur persistante.

1. Projet

10 min

Créer un projet STM32 pour répondant aux caractéristiques suivantes :

- nom : « eval_rtos_VOTRENOM »
- emplacement : un endroit judicieux
- configuration du projet : USART2 désactivé
- configuration de l'OS :
  - FreeRTOS, CMSIS_v1
  - Dans l'onglet *Config Parameters*
    - *Memory Allocation* à « *Dynamic* »
    - *USE_TASK_NOTIFICATIONS* à « *Enabled* »
    - tous les autres champs à « Disabled »
  - Dans l'onglet *Include Parameters*
    - tous les champs à « Disabled »

Assurez-vous de la **bonne compilation du projet**.

Appelez l'enseignant pour validation de la compilation (pas d'erreur, pas de warning).

2. Drivers UART

10 min

Ajoutez à votre workspace les deux fichiers de drivers de l'UART fournis sur Moodle.

Dans le fichier d'en-tête, dé-commentez la ligne permettant de sélectionner votre carte.

Initialisez le périphérique UART au démarrage du MCU puis envoyez le message suivant une fois les périphériques prêts :

```
"\r\n*** APPLICATION STARTING NOW ***\r\n"
```

Assurez-vous de la **bonne compilation du projet**.

Appelez l'enseignant pour validation sur cible (terminal série avec le texte affiché).

3. Application

Description

Vous allez développer une application de « minuteur ». Voici le principe du point de vue de l'utilisateur, adapté à la NUCLEO utilisée en TP.

Via un terminal série (TeraTerm, PuTTY, …) l'utilisateur appuie sur la touche 'U', 'D', ou 'C' pour incrémenter le minuteur d'une Unité, d'une Dizaine ou d'une Centaine, respectivement. De manière analogue, l'appui sur la touche 'u', 'd' ou 'c' permet cette fois-ci de diminuer la valeur du minuteur d'une unité, d'une dizaine ou d'une centaine.

L'appui sur le bouton poussoir bleu de la carte permet d'alterner entre la mise en marche et la mise en pause du minuteur. En cas de mise en marche, le minuteur décompte d'une unité à chaque seconde. En cas de mise en pause, le minuteur conserve sa dernière valeur.

Quand le minuteur atteint la valeur '0', il s'arrête de décompter.

Contraintes techniques

Nous utiliserons la carte STMicrolectronics STM32L073RZ (ou STM32L476RG), programmée grâce à l'IDE STM32CubeIDE et le configurateur graphique STM32CubeMX.

Le port série de la NUCLEO sera le ST-Link. La communication en UART entre le MCU STM32 cible et le ST-Link se fera grâce aux fichiers `ensi_uart.c` et `ensi_uart.h` fournis, imposant un débit à 9600 bps.

Vous utiliserez FreeRTOS et les outils étudiés en cours et travaux pratiques.

Tous vos développement devront se faire dans le fichier `Core/Src/freertos.c`.

Contraintes FreeRTOS

Les deux contraintes suivantes sont à prendre en compte dans la configuration de FreeRTOS :

- Vous utiliserez FreeRTOS en mode **préemptif**.
- Vous affecterez la **taille totale du heap** de FreeRTOS à la valeur **8000** (en octets).

4. Exercice initial

45-60 min

L'architecture logicielle, correspondant au cahier des charges initial, est donnée sur le diagramme en dernière page du sujet. Vous devez implémenter le programme équivalent dans le STM32.

Note 1 :

Tout élément nommé sur ce diagramme doit avoir le même nom dans le programme.

Note 2 :

La lecture du bouton poussoir inclura un mécanisme logiciel d'anti-rebond, présenté ci-dessous. Ces instructions sont donc à intégrer au code de la tâche responsable de la lecture du bouton poussoir. Vous trouverez ces lignes sur la page Moodle du cours, dans l'onglet dédié à l'examen, pour faire un copié-collé rapide.

```
	while(1) {
		// Bloqué ici tant que le bouton est relaché
		while(HAL_GPIO_ReadPin(B1_GPIO_Port, B1_Pin) == GPIO_PIN_SET);

		// Active ou désactive le décompte, selon son état actuel
		decompte_actif = 1 - decompte_actif;

		// Bloqué ici tant que le bouton est appuyé
		while(HAL_GPIO_ReadPin(B1_GPIO_Port, B1_Pin) == GPIO_PIN_RESET);
	}
```

Note 3 :

Il est important de procéder étape par étape, en implémentant les fonctionnalités une par une, d'abord au sein d'une seule tâche puis entre plusieurs tâches.

De même, il est inutile de rester bloqué trop longtemps. N'hésitez pas à demander des indices à l'enseignant.

Note 4 :

L'enseignant ne peut attribuer les points que si la démonstration est effectuée en sa présence.

Vous pouvez effectuer des démonstrations partielles pour engranger des points petit à petit.

5. Exercices supplémentaires

Ces exercices sont indépendants, et à réaliser dans l'ordre qui vous convient. L'estimation du temps nécessaire est donnée à titre indicatif.

Mécanismes de détection de débordement mémoire

5-10 min

Toute application doit disposer de mécanismes de détection d'erreurs. Mettez en œuvre les mécanismes de détection de débordement de la stack des tâches et du tas de FreeRTOS.

Rien de plus, rien de moins de ce qu'on a fait en TP.

Validez auprès de l'enseignant.

Protection de l'accès à la variable partagée

15 min

La variable globale `minuteur` est une ressource partagée. L'accès simultané à cette variable provoque un comportement indésirable.

Mettez en œuvre un mécanisme de protection d'accès à cette variable, de sorte à ce qu'une seule tâche à la fois puisse y accéder.

Activation du décompte par queue de message

15 min

La variable `decompte_actif` est une variable globale (premier défaut) et elle est activement scrutée par la tâche `decompteur` (deuxième défaut).

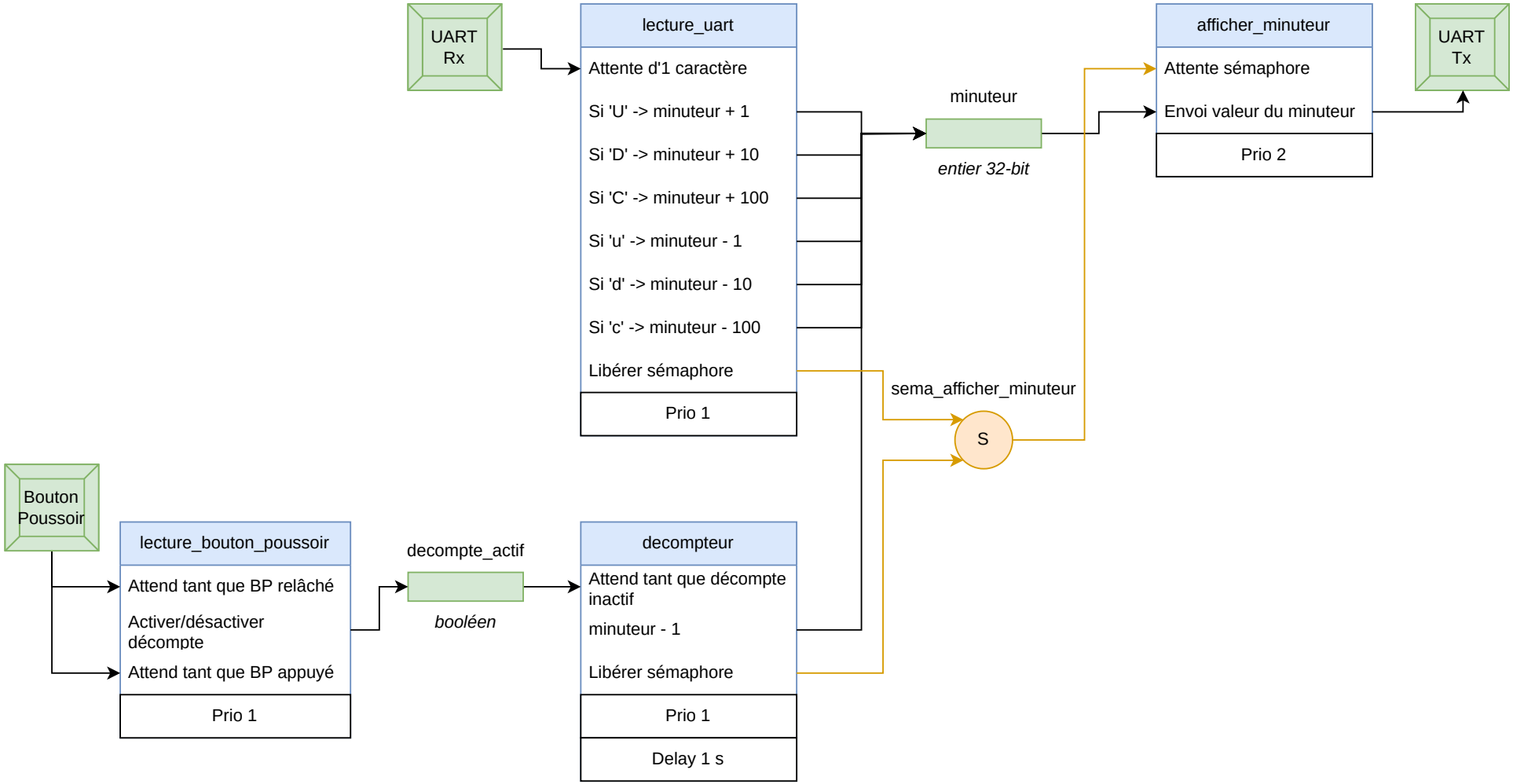
Supprimez cette variable globale et transmettez l'information `decompte_actif` par queue de message. La tâche `decompteur` doit être bloquée si `decompte_actif` n'a pas évolué, tout en restant périodique en cas de décompte activé.

Ajout de fonctionnalités

15 min

Quand le minuteur atteint la valeur '0', la tache `decompteur` libère un nouveau sémaphore, qui débloque une nouvelle tâche `minuteur_a_zero`. Cette tâche a pour seul but d'allumer la LD2 de la carte NUCLEO.

La LD2 est remise à l'état *off* quand l'utilisateur tape une lettre dans le terminal série.


UART
Rx
lecture_uart
Attente d'1 caractère
Si 'U' -> minuteur + 1
Si 'D' -> minuteur + 10
Si 'C' -> minuteur + 100
Si 'u' -> minuteur - 1
Si 'd' -> minuteur - 10
Si 'c' -> minuteur - 100
Libérer sémaphore
Prio 1
minuteur
entier 32-bit
afficher_minuteur
Attente sémaphore
Envoi valeur du minuteur
Prio 2
UART
Tx
sema_afficher_minuteur
S
Bouton
Poussoir
lecture_bouton_poussoir
Attend tant que BP relâché
Activer/désactiver décompte
Attend tant que BP appuyé
Prio 1
decompte_actif
booléen
decompteur
Attend tant que décompte inactif
minuteur - 1
Libérer sémaphore
Prio 1
Delay 1 s