swissuniversities

Hes·so Master

Cours T-EmbReal – Projet final

Spécifications

Vous devez réaliser une extension du programme BikeComputer selon la description suivante :

- Le comportement tel que défini dans la dernière étape du codelab BikeSystem (partie 3, étape 3) doit être réalisé.
- Le mécanisme du bootloader et de l'updateClient tel que décrit dans le codelab Mbed OS bootloader doit être entièrement fonctionnel selon la description du codelab.
- Au démarrage du BikeSystem, une analyse mémoire runtime du programme est effectuée (mémoire Heap et stack de chaque thread) selon ce qui a été effectué dans le codelab « Memory Profiling and Optimization ».
- Sur la base du code source distribué sur https://gitlab.forge.hefr.ch/serge.ayer/bikecomputer/-/tree/master/WithSensor/Sensor (code provenant de https://os.mbed.com/teams/ST/code/HelloWorld_ST_Sensors avec quelques modifications), vous devez ajouter une classe SensorHub qui :
    - Intègre les capteurs HTS221 et LPS22HB.
    - Permet d'obtenir des mesures de température, d'humidité et de pression atmosphérique à intervalle régulier dans une tâche réalisée par un thread indépendant.
    - Permet d'échanger les données récoltées avec le propriétaire d'une instance de la classe à l'aide d'une référence de Mail.
- La classe BikeSystem doit posséder une instance de SensorHub et obtenir des données de la façon suivante :
    - Dans la boucle du ProcessingThread, lorsque des données du SensorHub sont disponibles, celles-ci sont obtenues.
    - Une moyenne des mesures obtenues est recalculée à chaque nouvelle donnée disponible.
    - Les moyennes calculées sont transmises à l'instance de LCDDisplay en étendant le mécanisme existant (et en étendant donc la structure ProcessedData).
- La classe LCDDisplay est modifiée afin d'afficher les moyennes des mesures de capteurs.
- La classe ResetDevice doit être renommée en ButtonDevice et modifiée de sorte à pouvoir spécifier un callback pour le « rise » et le « fall » du bouton. L'utilisation de cette classe renommée doit être modifiée dans la classe BikeSystem :
    - Lorsque le bouton est pressé pendant une durée inférieure à 1 seconde, une mesure des données de capteurs doit être effectuée. Ce mécanisme est réalisé en indiquant au thread effectuant les mesures de capteurs de ne pas attendre le prochain intervalle de temps et d'effectuer une mesure immédiatement.
    - Lorsque le bouton est pressé pendant une durée supérieure à 1 seconde, une analyse mémoire différentielle du programme est effectuée selon les descriptions du codelab correspondant.
- (Optionnel) Le capteur de distance VLS53L0X est intégré de sorte à ce qu'une mesure des données de capteurs soit effectuée lorsque la distance mesurée par le capteur est inférieure à 10 cm. La mesure de distance à l'aide du capteur VLS53L0X doit être effectuée dans la

classe SensorHub, à l'aide d'un mécanisme laissé au choix des étudiants. La contrainte est que la mesure des données de capteurs doit être effectuée au plus tard 1 seconde après le placement d'un objet devant le capteur de distance et ne doit être effectuée qu'une seule fois avant que l'objet soit retirée.

Contraintes

Les contraintes suivantes doivent être respectées :

1. Le projet est réalisé par groupe de 2 étudiant(e)s. Les groupes doivent être annoncés avant le 20 décembre 2020, par e-mail à Serge.Ayer@hefr.ch (envoyé par 1 étudiant, avec copie au deuxième étudiant).
2. Le code source doit être délivré sur github.com (projet privé et partagé avec Serge.Ayer@hefr.ch et Daniel.Gachet@hefr.ch). Une invitation doit parvenir aux deux adresses mentionnées. Si cette solution n'est pas réalisable pour un groupe d'étudiants, celui-ci doit proposer une alternative qui sera validée par les enseignants.
3. Le code source rendu doit inclure tout le code source du programme BikeComputer et du programme bootloader, ainsi que les fichiers de configuration des applications (sans oublier les fichiers .mbedignore). La librairie mbed-os doit être incluse sous la forme du fichier mbed-os.lib. Afin de faciliter la compilation des deux applications, le code du UpdateClient doit être copié dans les deux applications.
4. La référence au fichier binaire du bootloader doit être correcte dans le fichier de configuration de l'application BikeComputer – ce qui signifie qu'après compilation de l'application bootloader, il suffit de compiler l'application BikeComputer sans copier ou déplacer le fichier .bin dw l'application bootloader.
5. Le nom des projets sur github.com doit être ProjetTEmbReal_XX_YY, où XX et YY sont les initiales des deux étudiants participant au projet. A la racine du projet, vous devez créer deux répertoires BikeComputer et bootloader, dans lesquels vous déposez les deux applications.
6. Le délai pour le dépôt du code est le 20 janvier 2021 à 23h59.
7. Il vous est recommandé de tester que les projets tels que déposés sur github peuvent être compilés avec succès avec un simple clone des projets.
8. Le code doit compiler avec la version 6.3 de la librairie mbed-os. Le code doit compiler sans warning (à l'exclusion des warnings sur le code de la librairie mbed-os).
9. Le comportement de base du BikeSystem (point 1 des spécifications) peut être réalisé à partir du code source distribué sur https://gitlab.forge.hefr.ch/serge.ayer/bikecomputer ou sur la base de votre propre code source.

Modalités d'évaluation

1. Evaluation du code fourni selon les spécifications et contraintes (50%)
2. Evaluation de la partie optionnelle (bonus : max 0.5 pt)
3. Présentation (par groupe, max 5 min, 2 transparents, architecture du code de l'application, présentation des validations effectuées, 10%)
4. Examen oral (individuel, 10 mins, 2 questions portant sur le projet réalisé et les concepts relatifs, 40%)