

Exemple de sujet d'examen TP

Notes

Ce sujet n'a jamais été proposé en examen TP : il est long (trop long pour tenir en 1h30) et les questions sont moins guidées que sur les sujets habituels. La dernière question par exemple est trop complexe pour être demandée en examen.

En revanche il est représentatif de ce qui peut être donné pour l'examen TP, tant sur les périphériques que sur le type d'application demandée.

Entraînez-vous, seul ou à plusieurs, et n'hésitez pas à me contacter pour avoir des conseils ou indices.

Les fichiers fournis sont stockés dans une archive complète comprenant :

- tous les corrigés des périphériques et applications étudiés en TP
 - Vous pouvez si vous préférez ré-utiliser vos fichiers de TP
- un fichier `apps/minuteur/main.c` vierge, à compléter
- les corrections de chaque question du `main.c` sont dans le répertoire `apps/minuteur/solutions`.

Vous avez droit pour l'examen de TP à tous les documents papiers et numériques utilisés en séances de TP.

Q1 (facile)

Créez un projet dans le répertoire `eval_tp/apps/minuteur/project /`, portant le nom `eval_VOTRENOM`.

Ajoutez au projet le fichier `eval_tp/apps/minuteur/src/main.c`.

Compilez, il ne doit avoir ni erreur ni warning.

Q2 (facile)

Configurez le périphérique UART1, en imposant un baudrate de 115200 bauds.

Au démarrage de l'application (et uniquement au démarrage), envoyez le message `"\r\n\r\n### DEMARRAGE DE L'APPLICATION MINUTEUR"` via le périphérique UART1.

Compilez, il ne doit avoir ni erreur ni warning.

Validez en affichant le message sur un terminal série (pensez également à configurer le moniteur série).

Q3 (moyen)

Au démarrage de l'application, initialisez et éteignez toutes les LEDs.

Configurez le timer0 pour déclencher une interruption de priorité basse, chaque seconde.

À chaque seconde, la LED2 doit changer d'état.

Note : ne pas oublier d'autoriser les interruptions au niveau du micro-contrôleur (registre `INTCON`).

Indice : vous pouvez déjà confirmer le fonctionnement sans forcément modifier la période du timer0. Une fois que le principe de fonctionnement est validé, là vous pourrez chercher à régler la période à 1 seconde.

Q4 (facile)

Modifiez l'application (contenu du `while(1)`) de sorte à ce que chaque appui sur le bouton poussoir SW1 prépare le déclenchement d'un minuteur (décompte) :

- On lit la valeur du SW1
- s'il est appuyé, alors :
 - on allume la LED3
 - on envoie le message `"\r\nTop minuteur"` par l'UART1

Q5 (on corse les choses)

L'appui sur le SW1 lance aussi le décompte :

- le minuteur est démarré (`minuteur_actif` à `TRUE`);
- le minuteur est affecté à sa valeur initiale

À chaque seconde, si le minuteur est actif, alors sa valeur est décrémentée.

Au bout de 5 secondes, la LED3 s'éteint et le message `"\r\nTemps ecoule"` est envoyé à l'UART1.

Indice 1 : la variable `decompteur` doit être globale pour pouvoir être décomptée entre chaque appel à la fonction d'interruption.

Indice 2 : une autre variable globale pour être utilisée pour stocker l'état du décompteur (on ou off).

Q6 (facile avec l'indice)

À chaque seconde écoulée, la valeur du décompteur est envoyée par UART.

Indice : la valeur `valeur_courante_minuteur + '0'` permet de transformer la valeur du compteur en caractère imprimable sur la console.

Q7 (presque facile)

On modifie l'application de sorte qu'un appui sur le bouton SW2 met le décompteur en pause :

- la LED2 clignote toujours (le timer0 n'est pas désactivé)
- le décomptage ne se fait plus, donc aucun message supplémentaire (valeur du décompteur) n'arrive sur le terminal série

Un nouvel appui sur le bouton SW2 remet le minuteur en marche.

Indice : une variable globale est maintenant nécessaire pour stocker l'état du décompte (on ou off).

Q8 (compliqué)

L'utilisateur peut maintenant saisir la valeur initiale du décompte avec le terminal série. On commence par un décompte dont la valeur est limitée entre 0 et 9 secondes.

Modifiez l'application principale de sorte que :

- une réception sur l'UART1 doit déclencher une interruption de haute priorité
- on vérifie si on a reçu un caractère
- si oui, alors on teste si le caractère reçu est compris entre '0' et '9'
- si oui, alors on stocke la valeur lue dans une variable qui contient la valeur initiale du décompte

Indice 1 : la valeur initiale du décompte est une variable globale.

Indice 2 : pour convertir la valeur reçue en valeur entière, il faudra stocker `caract_recu - '0'`.

Q9 (mode expert)

L'utilisateur peut maintenant saisir la valeur initiale du décompte avec le terminal série. Cette fois la valeur peut s'étendre à plus de 9 secondes (soit plusieurs caractères à lire).

Modifiez l'application principale de sorte que

- on vérifie si on a reçu une chaîne de caractères
- si oui, alors on teste si tous les caractères reçus sont compris entre '0' et '9'
- si oui, il faut convertir la valeur représentée sous forme de chaîne de caractères, en valeur entière.