ENSMM
2e année

Commande des systèmes par microcontrôleur

Examen final – durée 2h
Documents manuscrits autorisés

Exercice : 5 points

Question 1. Quelle est l'utilité des interruptions ? Donner une réponse argumentée.

Question 2. Développez un programme en C pour le PIC 16F877 qui permet de compter à l'aide d'une **interruption** le nombre d'appuis sur un bouton poussoir. Au démarrage du programme, ce nombre vaut zéro et il doit s'incrémenter de 1 à chaque appui.

L'électronique externe associée au bouton est conçue telle que :

- un appui sur le bouton poussoir engendre un niveau logique **bas** sur la pin RB0/INT du microcontrôleur,
- un relâchement du bouton poussoir engendre un niveau logique **haut** sur la pin RB0/INT.

Les bits à configurer et à tester pour mettre en œuvre cette interruption se situent dans les registres INTCON et OPTION_REG.

Problème : 15 points (les parties I et II peuvent être traitées de manière indépendante)

Noël approche... Vous avez été embauché par une société qui conçoit des jouets pour développer la logique de commande d'un nouveau jouet pour enfants de 2 ans et plus. Il s'agit d'une « coccinelle magique » (cf. figure 1) capable de fonctionner selon deux modes différents : un mode manuel et un mode automatique. Dans le mode manuel, l'enfant choisit lui-même le mouvement de la coccinelle en appuyant sur des boutons de commande. Dans le mode automatique, la coccinelle se déplace toute seule pendant 10 secondes en direction de la source la plus lumineuse.

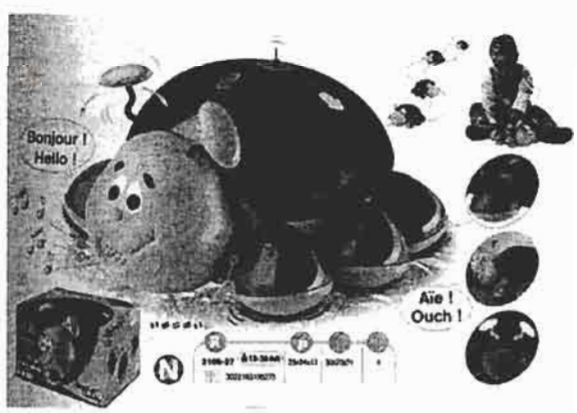


Fig. 1 – Coccinelle magique

Fort de vos connaissances sur la commande des systèmes par microcontrôleur, vous décidez d'utiliser un PIC 16F877 pour répondre au cahier des charges du jouet.

Partie I : mode manuel (7 points)

Dans le mode manuel, l'enfant a la possibilité d'appuyer sur 3 boutons (bouton_A, bouton_G, bouton_D) pour faire aller la coccinelle vers l'avant, la faire tourner à gauche ou la faire tourner à droite (cf. figure 2). Quand un bouton est appuyé, il s'allume grâce à une LED incorporée.

Les boutons bouton_A, bouton_G, bouton_D sont reliés aux bits 0,1,2 du port C.

Les trois LED des boutons lumineux sont reliées aux bits 4,5,6 du port C. Si un des ces bits est à 1, la LED correspondante est allumée (éteinte sinon).

L'électronique qui gère les moteurs de la coccinelle est commandée via les bits 0,1,2 du port B. La coccinelle avance si le bit 0 est à 1, elle tourne à gauche si le bit 1 est à 1, elle tourne à droite si le bit 2 est à 1.

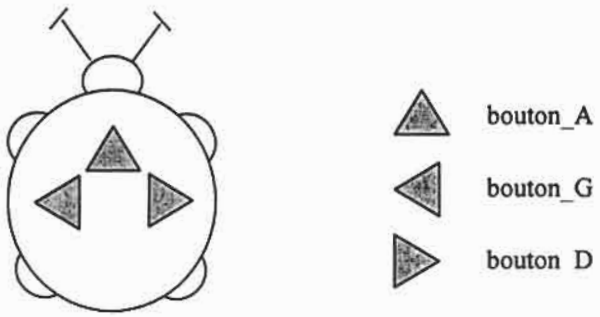
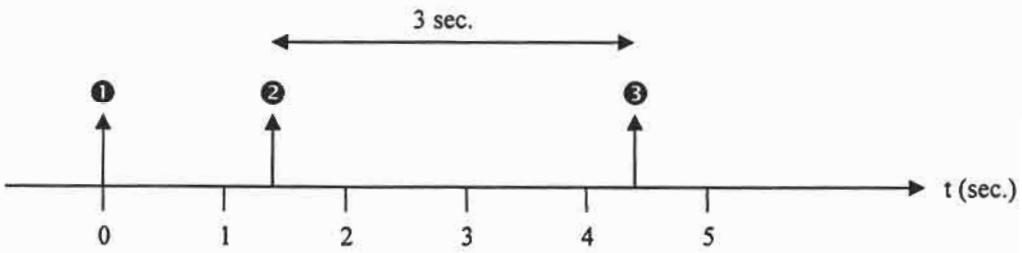


Fig. 2 – Boutons lumineux de commande de la coccinelle

Tout bouton appuyé déclenche un déplacement de la coccinelle pendant trois secondes. Le bouton s'éteint à l'issu du déplacement. Si un bouton est appuyé durant le déplacement, son action remplace l'ancienne pendant à nouveau trois secondes (cf. l'exemple de chronogramme de la figure 3).



❶ Appui sur bouton_A : la coccinelle avance
❷ Appui sur bouton_D avant la fin des trois secondes : la coccinelle tourne à droite
❸ Arrêt de la coccinelle trois secondes après le dernier appui

Fig. 3 – Chronogramme d' « addition » de mouvements

Question 1. Spécifier les entrées et les sorties numériques de la coccinelle. Comment faut-il configurer TRISB et TRISC (justifier) ?

Question 2. Programmer le mode manuel en utilisant le langage C et en mettant en œuvre une interruption par le timer 0.

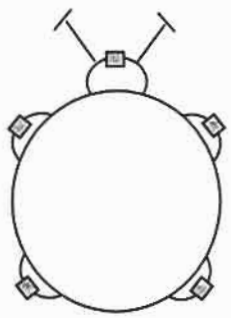
La **scrutation** de l'état des boutons aura lieu dans la procédure **main()** et l'interruption par le **timer 0** trois secondes après le dernier appui de bouton servira à arrêter le mouvement de la coccinelle et à éteindre les boutons.

Le quartz qui pilote l'horloge interne du PIC 16F877 a une fréquence Fosc égale à 4 MHz.

Les bits initialisés ou testés dans les registres doivent **impérativement** faire l'objet d'une explication.

Question 3. Pour éviter d'endommager la coccinelle, 5 capteurs de contact sont disposés sur le pourtour de la coccinelle (cf. figure 4). Ces 5 capteurs sont reliés aux bits 3,4,5,6,7 du port B. Ces bits passent à 1 s'il y a contact et valent 0 sinon.

Modifier la procédure **main()** (configuration du Port B, etc.) pour qu'en cas de contact, toutes les commandes en cours soient stoppées et que les LED des boutons soient éteintes.



Capteurs de contact numériques « tout ou rien » reliés au port B.

Fig. 4 – Capteurs de contact disposés sur la coccinelle

Partie II : mode automatique (8 points)

Le mode automatique est déclenché par un nouveau bouton « bouton_Aut » relié au bit 3 du port C. (cf. figure 5). La LED correspondant à ce bouton est reliée au bit 7 du port C.

Lorsque ce bouton est pressé, il s'allume et la coccinelle se dirige automatiquement pendant 10 secondes vers la source la plus lumineuse située devant elle. Si l'enfant ré-appuie sur bouton_Aut avant que 10 secondes ne se soient écoulées, le mode automatique redémarre pour 10 secondes.

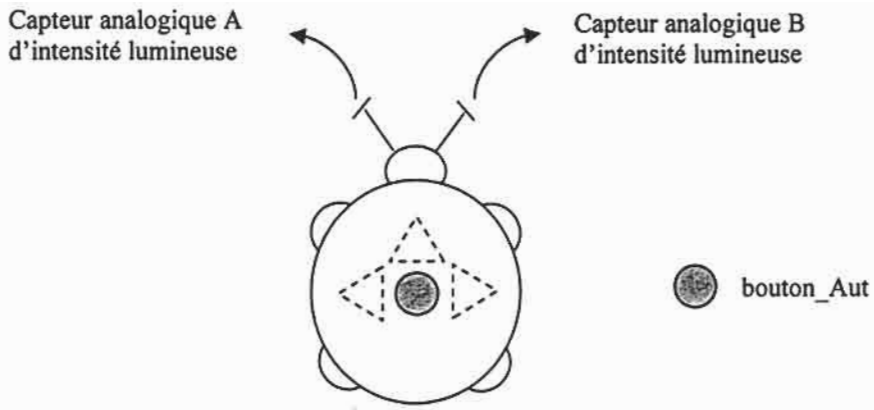


Fig. 5 – Bouton lumineux de déclenchement du mode automatique
et capteurs analogiques d'intensité lumineuse

A l'issue du déplacement, le bouton s'éteint. Par ailleurs, le mode automatique s'interrompt si :

- les 10 secondes sont écoulées,
- au moins un des cinq capteurs de contact est activé,
- l'enfant appuie sur un des trois boutons bouton_A, bouton_G, bouton_D, ce qui déclenche les actions correspondantes du mode manuel (donc pendant 3 secondes).

Nota : si vous traitez la partie II sans avoir fait la partie I, le mode automatique s'interrompt uniquement si les 10 secondes sont écoulées.

Deux capteurs analogiques A et B d'intensité lumineuse sont utilisés pour implémenter le comportement automatique (cf. figure 5). Ces capteurs fournissent des tensions analogiques U_A et U_B comprises entre V_{DD} et V_{SS}. Ils sont reliés aux entrées RA0/AN0 et RA1/AN1 du démultiplexeur du convertisseur analogique-numérique (CAN).

Toutes les 0.25 secondes, les tensions U_A et U_B sont acquises par le CAN et la différence $U_A - U_B$ est calculée. En fonction de la valeur de cette différence, une commande est exécutée, comme l'illustre la figure 6. Si U_A est très supérieure à U_B, la coccinelle tourne à gauche. Si U_A est environ égale à U_B, elle va tout droit et si U_A est très inférieure à U_B elle tourne à droite.

Question 1. Détailler les registres à utiliser ainsi que la séquence d'instructions qui permet l'acquisition sous la forme d'un octet des tensions U_A et U_B (sans utiliser d'interruption).

Question 2. Quel type faut il utiliser pour représenter la différence $U_A - U_B$?

Question 3. Modifier le programme de la partie I pour ajouter le mode automatique. La scrutation de l'état du bouton_Aut sera ajoutée dans la procédure **main()**. Les tensions U_A et U_B seront lues toutes les 0.25 secondes (environ !) via une interruption générée par le **timer 1**. La durée de 10 secondes du mode automatique doit également être implémentée à l'aide du timer 1.

Le quartz qui pilote l'horloge interne du PIC 16F877 a une fréquence F_{OSC} égale à 4 MHz.

Les bits initialisés ou testés dans les registres doivent **impérativement** faire l'objet d'une explication.

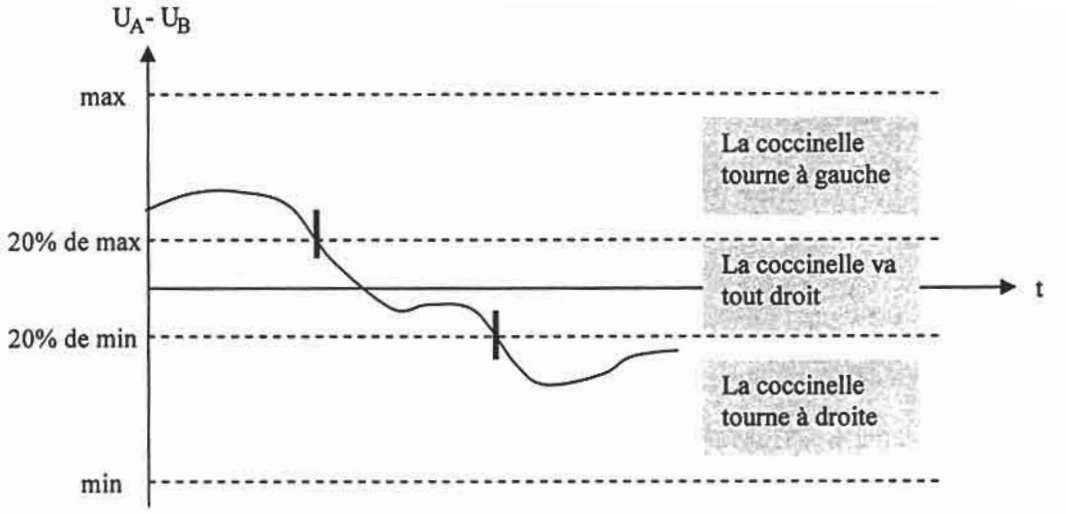


Fig. 6 – Comportement de la coccinelle en fonction des tensions U_A et U_B

Partie III : mode programmation

Ceci est une dissertation libre si vous avez tout fini qui vous permettra d'obtenir un bonus de points

Afin de donner plus d'intérêt au jouet (notamment pour les enfants plus âgés), le responsable du bureau d'étude souhaite que vous ajoutiez un mode programmation. L'enfant aura alors en plus la possibilité de faire mémoriser à la coccinelle un maximum de 10 commandes ayant chacune une durée de 3 secondes avant de les exécuter en mode automatique. Vous avez carte blanche pour implémenter cette nouvelle fonctionnalité avec le PIC 16F877. Comment procéderiez vous ?