
Projet
Exultant
Lusibus

Projet EXULTANT LUSIBUS

Vidéo de présentation :
https://drive.google.com/file/d/1YE70hnQ3wgruwF1HO6QocrSbMleH4HwV/view?usp=sharing

Principe

Les deux robots commencent face à face. En appuyant sur SW1, chaque robot avance jusqu'à toucher les bumpers de l'autre robot, déclenchant ainsi le début de la danse sur la musique de Charlie Puth, Betty Boop (Electro-Swing Remix). La danse dure environ 1mn55 et est composée de divers mouvements (notamment des mouvements avec des vitesses différentes sur chacune des roues, permettant ainsi de varier les figures possibles).

Description du projet

Il y a trois phases distinctes dans ce projet :

La **phase de démarrage** qui s'initialise dès que les robots sont allumés. La LED 1 s'il s'agit du ROBOT 1 et la LED 2 s'allume s'il s'agit du ROBOT 2.

La **phase de synchronisation** débute dès l'appui sur le SWITCH 1. Les robots se mettent à avancer, face à face. Lorsque leurs bumpers se touchent, la phase suivante s'enclenche.

La **phase de danse** dure 1mn55 sur la musique Betty Boop (Electro-Swing Remix) de Charlie Puth. La musique étant composée de plusieurs voix/leads distincts, le premier robot va suivre la voix principale tandis que le deuxième s'appuie sur les voix secondaires. Lorsqu'un moteur est en fonctionnement 'avant', la led du côté de ce moteur s'allume; elle s'éteint dans les autres cas.

Détails techniques

Configuration des GPIO utilisés

Dans ce programme, les switches, les bumpers, les moteurs et les leds sont utilisés. Pour établir une connexion entre le processeur du robot et ces périphériques, il est nécessaire de passer par les ports GPIO correspondants.

Pour faire cela, nous avons besoin des ports H (moteurs), F lignes 4 et 5 (leds), D ligne 6 (switch 1) et E lignes 0 et 1 (bumpers). Pour terminer l'initialisation des ports, on connecte la clock aux ports utilisés.

```
ldr r6, = SYSCTL_PERIPH_GPIO
mov r0, #0x00000038 ; On active la clock sur GPIO D, F et E

str r0, [r6]
```

Ensuite, après trois cycles d'horloge, il faut configurer indépendamment les différents périphériques. Nous allons prendre l'exemple des bumpers, les autres configurations étant assez similaires. Ainsi, on utilise le masque BROCHE0_1 égal à 0x03 (lignes 0 + 1 du port E). On pourra ensuite lire dans r11 l'état des bumpers (changement de valeur si l'un des deux bumpers est activé).

```
ldr r7, = GPIO_PORTE_BASE+GPIO_I_PUR    ;; Pul_up
ldr r0, = BROCHE0_1
str r0, [r7]

ldr r7, = GPIO_PORTE_BASE+GPIO_O_DEN    ;; Enable Digital Function
ldr r0, = BROCHE0_1
str r0, [r7]

ldr r11, = GPIO_PORTE_BASE + (BROCHE0_1<<2)     ; both bumpers
```

Utilisation de plusieurs vitesses pour les moteurs

Nous avons eu besoin d'utiliser plusieurs vitesses de rotation pour les moteurs afin de diversifier quelque peu notre chorégraphie. Pour cela, nous avons créé trois MOTEUR_INIT différents.

- MOTEUR_INIT_RAPIDE : même vitesse pour les deux roues
- MOTEUR_INIT_GAUCHE : la vitesse de la roue gauche est nettement plus grande que celle de la roue droite.
- MOTEUR_INIT_DROITE : la vitesse de la roue droite est nettement plus grande que celle de la roue gauche.

Ensuite, pour changer la vitesse pendant le déroulement du __main, il suffit d'appeler un de ces trois fonctions et les changements seront immédiats.

Différents timers afin de rester synchronisé avec la musique

On ne peut pas parler musique sans penser tempo. Il a donc bien sûr été nécessaire de créer une fonction WAIT_NOIRE qui corresponde au tempo de la musique : 118 BPM. Pour cela, nous avons dû tâtonner jusqu'à trouver l'hexadécimal qui fonctionnait (ce ne fut pas de tout repos).

```
WAIT_NOIRE  ldr r1, =0x293C00 ; Duree correspondant à 1 noire à 118BPM
wait4   subs r1, #1
        bne wait4
        BX  LR
```

Ensuite, nous avons également implémenté les timers correspondants aux différents rythmes musicaux présents dans la musique (blanche, croche, double-croche, triolet, triple-croche). Cela nous a permis de faciliter grandement la mise en place de la chorégraphie.

Duplication de code

Afin d'éviter toute duplication de code (le programme étant déjà très long), nous avons eu recours à de nombreuses boucle FOR permettant de répéter aisément des portions de codes.

```
        ldr r12,=0x03 ; Boucle FOR évitant la duplication de code
move1
        BL MOTEUR_GAUCHE_AVANT
        BL MOTEUR_DROIT_AVANT
        BL WAIT_NOIRE

        BL MOTEUR_GAUCHE_ARRIERE
        BL MOTEUR_DROIT_ARRIERE
        BL WAIT_NOIRE

        subs r12,#1
        bne move1
```

Démarrage de la danse et synchronisation avec la musique

Certes, l'utilisation des bumpers permet de synchroniser les deux robots entre eux. Néanmoins, la synchronisation avec la musique est une toute autre affaire. Nous avons songé à ajouter un capteur son sur les robots, mais, le temps étant assez limité, nous avons dû nous tourner vers une solution différente.

Nous avons ainsi créé un fichier .wav spécial dans lequel la musique ne démarre pas immédiatement. Elle est en effet précédée d'un métronome à 118 BPM. Pour démarrer la séquence de danse convenablement, il faut alors procéder ainsi :

- Allumer les deux robots
- Les placer face à face à une distance d'une feuille A4 l'un de l'autre.
- Démarrer la lecture du fichier WAV.
- Lorsque le cinquième coup retentit, appuyer sur le switch 1 des deux robots.