Polytechnique Montréal

Département de génie informatique et génie logiciel

Cours INF1995:
Projet initial en génie informatique et travail en équipe

Travail pratique 8

Makefile et production de librairie statique

Par l'équipe

No 8292

Noms:

Cédric Tessier
Martin Pouliot
Pascal-Alexandre Morel
Sean Costello

Date:
12 mars 2018

Partie 1 : Description de la librairie

La librairie que nous avons créée se veut versatile. Certains bouts de code produits dans les laboratoires précédents ont été réutilisés, mais plusieurs ont été créés pour l'occasion. L'une des fonctionnalités qui ont été développées pour l'occasion a été le son. Nous avons choisi d'utiliser les namespace pour séparer chaque élément de notre librairie. Les namespace étaient un choix évident, car nous avions plusieurs fonctions init pour l'initialisation et test pour tester les fonctionnalités de la librairie. Les deux fichiers memoire_24.h et can.h qui ont été fournis au courant des travaux pratiques ont été rajoutés dans la bibliothèque. Nous avons également créé le fichier ohboy.h qui s'occupe d'inclure tous les autres .h nécessaires. Sur les prochaines pages de ce rapport, les autres .h seront expliqués.

timer.h

Regroupe les fonctions permettant de contrôler le timer 1.

init	Description	Activer le timer 1 et les interrupts tout en ajustant les registres
	Paramètre(s)	ms : Intervalle de temps entre chaque incrémentation du compteur
	Return	void
on	Description	Activer les interrupts et le timer
	Paramètre(s)	Aucun
	Return	void
off	Description	Désactiver les interrupts et le timer
	Paramètre(s)	Aucun
	Return	void
test	Description	Permettre de tester toutes les fonctions de timer.h
	Paramètre(s)	ms : paramètre pour la fonction init
	Return	void

light.h

Regroupe les différentes fonctions permettant le contrôle de la DEL sur le PORTD. La DEL doit être branchée sur les broches 3 et 4 du PORTD (câble rouge sur la broche 4 et le +).

init	Description	Mettre les broches 3 et 4 du PORTD en mode sortie pour le bon fonctionnement de la DEL
	Paramètre(s)	Aucun
	Return	void
green	Description	Donner la valeur “1” à la broche 3 et la valeur “0” à la broche 4 du PORTD ce qui a pour effet de donner la couleur verte à la DEL si elle est branchée correctement
	Paramètre(s)	Aucun
	Return	void
red	Description	Donner la valeur “1” à la broche 4 et la valeur “0” à la broche 3 du PORTD ce qui a pour effet de donner la couleur rouge à la DEL si elle est branchée correctement
	Paramètre(s)	Aucun
	Return	void
amber	Description	Faire appel aux fonctions red et green en alternance durant un certain nombre de ms
	Paramètre(s)	ms : Temps que la DEL est ambre
	Return	void
test	Description	Permettre de tester toutes les fonctions de light.h
	Paramètre(s)	Aucun
	Return	void

pwm.h

Regroupe les différentes fonctions en lien avec le pwm qui aura pour fonction de contrôler les moteurs. Les câbles jaunes doivent être branchés sur les broches 5 et 6 pour chaque moteur. De plus, le cable jaune doit être vers le haut sur le pont en H. Les autres connecteurs doivent être sur les broches 7 et 8.

init	Description	Ajuster les registres pour le pwm
	Paramètre(s)	Aucun
	Return	void
setA	Description	Ajuster le pwm avec OCR1A
	Paramètre(s)	signal: valeur de OCR1A (entre 0 et 255) wheelDirection: direction des moteurs (1 ou 0)
	Return	void
setB	Description	Ajuster le pwm avec OCR1B
	Paramètres	signal: valeur de OCR1B (entre 0 et 255) wheelDirection: direction des moteurs (1 ou 0)
	Return	void
test	Description	Permettre de tester toutes les fonctions de pwm.h
	Paramètre(s)	Aucun
	Return	void

sound.h

Regroupe plusieurs fonctions pouvant jouer différentes mélodies selon contexte (succès/échec).

init	Description	Mettre la broche 3 (PD2) du PORTD en sortie.
	Paramètre(s)	Aucun
	Return	void
succes	Description	Jouer une mélodie de succès
	Paramètre(s)	Aucun
	Return	void
fail	Description	Jouer une mélodie de défaite
	Paramètre(s)	Aucun
	Return	void
beep	Description	Jouer la note D4 pour une certaine durée.
	Paramètre(s)	ms : durée de la tonalité
	Return	void
crazyFrog	Description	Jouer une chanson (Crazy Frog)
	Paramètre(s)	Aucun
	Return	void
test	Description	Permettre de tester les fonctions de sound.h
	Paramètre(s)	Aucun
	Return	void

uart.h

Permet d'afficher à l'écran différents messages à l'aide de SerieViaUSB.

init	Description	Ajuster les registres pour le uart
	Paramètre(s)	Aucun
	Return	void
print	Description	Permettre d'afficher différents messages selon l'argument entré.
	Paramètre(s)	Plusieurs fonctions ayant le même nom, mais prenant différents arguments, comme des uint8_t, uint16_t, char*, etc.
	Return	void
println	Description	Permettre d'afficher un retour à la ligne ('\n').
	Paramètre(s)	Aucun
	Return	void
test	Description	Permettre de tester l'affichage des fonctions print.
	Paramètre(s)	Aucun
	Return	void

Partie 2 : Décrire les modifications apportées au Makefile de départ

Plusieurs modifications ont été apportées au Makefile de départ. Pour commencer, nous avons décidé d'utiliser un Makefile commun à la racine du projet pour s'occuper de gérer les variables communes. Les variables communes au Makefile de départ et au Makefile de la librairie sont les suivantes.

- CC (le compilateur)
- MCU (le choix du microcontrôleur)
- OPTLEVEL (Optimisation du compilateur)
- CFLAGS (Paramètres à passer au compilateur)

Les déclarations de ces variables ont été retirées du Makefile de départ pour qu'elles soient prises dans le Makefile commun à la racine du projet.

Ensuite nous avons inclus le dossier ../lib et inclus la librairie ohboy dans le Makefile de départ qui se trouve dans le dossier test. Ce dossier contient notre code bidon qui permet de tester toutes les fonctions de notre librairie.

Une fois ces modifications effectuées, nous nous sommes concentrés sur le deuxième Makefile, soit celui qui fait la compilation de la librairie. Pour ce code, nous sommes partis de zéro, car le Makefile fourni avait beaucoup de choses dont nous n'avions pas besoin. Nous avons fait le graphe de dépendance de notre librairie et écrit le Makefile en conséquence. Nous avons utilisé les variables pour le compilateur et pour les flags. Dans la règle libohboy.a, nous avons utilisé la commande avr-ar pour combiner tous les objets binaires en une bibliothèque. Une règle clean a été rajoutée pour effacer les artéfacts.