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 911

Noms:

Hamdi Mansour
Marc-Antoine Jacob
Grégoire Geffroy
Jean-Philippe Anctil

Date:
12 mars 2018

Partie 1 : Description de la librairie

La librairie est constitué à partir d'un seul fichier ,h qui contient les fonctions et les énumérations suivantes:

bool isPressed(uint8_t reboundTime = 10)

Cette fonction retourne true si le bouton poussoir est pressé tout en appliquant le mécanisme d'antirebond.

enum Color

Cette énumération définie les couleurs possible que peut prendre la DEL en associant efficacement la borne positive de la Del au bit le plus significatif. Cette énumération sera utilisée par changeLEDColor.

void changeLEDColor(Color c, volatile uint8_t& port = PORTA, uint8_t offset = 0)

Cette fonction change la couleur de la LED en utilisant les valeurs numérique de l'enumération Color sur le port A par défaut. Dans le cas où c vaut AMBER on alterne entre RED et GREEN avec un décalage de temps offset pour pallier l'asymétrie potentielle causée par des délais précédents. Notez bien qu'il faut dans ce cas appeler cette fonction en boucle.

void PWM(const float freq, const float rate, volatile uint8_t& out, const uint8_t low = 0, const uint8_t high = 1)

Cette fonction génère un PWM logiciel sur la sortie out (un port quelconque) à la fréquence spécifié f. Pour rendre cette fonctione la plus flexible possible, on peut spécifier une valeur high et low autre que les classiques 1 et 0 . Par exemple, on pourrait utiliser RED pour low et GREEN pour high.

void remap(uint8_t& value, uint8_t min, uint8_t max, uint8_t newMin, uint8_t newMax)

Petite fonction utilitaire pour rééchelonner une valeur d'un invertervalle donné vers un autre intervalle spécifié.

Nous avons également inclus les fichiers memoire_24.cpp et can.cpp à la librairie.

Bien sûr, la librairie va évoluer en fonction de nos besoins. Nous n'avons pas instauré une architecture orientée objet, car nous n'avons pas encore tous les concepts ni la bonne structure en tête.

Partie 2 : Décrire les modifications apportées au Makefile de départ

Décrire les quelques modifications apportées au Makefile de la librairie pour démontrer votre compréhension de la formation des fichiers. Faire de même pour les modifications apportées au Makefile du code (bidon) de test qui utilise cette librairie.

Librairie : Nous avons modifié la variable TRG pour exporter un fichier .a au lieu d'un fichier .out. Nous avons changé l'implémentation de la cible pour produire une libraire avec la commande ar avec les options c (création de l'archive), r (insertion de tous les fichiers), s (indexation des fichiers).

Test : On ajoute le dossier librairie aux inclusions additionnelles

INC= -I../librairie

afin que l'éditeur de liens puisse trouver le fichier .h de la librairie charger les symboles nécessaires à la compilation.

Aussi, nous donnons le chemin relatif pour lier notre librairie dans la variable LIBS avec l'option -L

LIBS=-L../librairie -lMagic

L'éditeur de lien se chargera automatiquement de trouver la librairie correspondant à libMagic.a

Commun : Nous avons factorisé ce qui restait en commun des 2 makefiles pour en former un général que nous incluons dans les tous les autres

include ../Makefile.commun