Polytechnique Montréal

Département de génie informatique et génie logiciel

Cours INF1900:
Projet initial de système embarqué

Travail pratique 7

Makefile et production de librairie statique

Par l'équipe

No 7071

Noms:

Akinotcho Azhar
Mohand Allache
Batista-Moniz Dylan
Charland Jeremy

Date:
03 Mars 2019

Partie 1 : Description de la librairie

Notre librairie est constituée de plusieurs classes. Nous décrirons ces classes en nous appuyant sur leurs objectifs, leurs constructeurs et les méthodes implémentées dans les lignes qui suivent :

Class Memoire24 du fichier memoire_24.h

Objectif : il s'agit d'une classe qui a pour rôle de permettre à l'utilisateur de manipule la mémoire externe de notre robot grâce aux ports C0 et C1

Constructeur : Son constructeur Memoire24CXXX() ajuste la taille de la page appelle une méthode d'initialisation ; cette dernière initialise le port série et l'horloge de l'interface I2C.

Méthodes implémentées : les méthodes implémentées sont celles de lecture et d'écriture ; il existe deux types pour chacune de celles-ci :

- Lecture :
  -uint8_t lecture(const uint16_t adresse, uint8_t *donnee):
  La lecture qui se fait caractère par caractère et prends en paramètre l'adresse et un pointeur vers les données .

  -uint8_t lecture(const uint16_t adresse, uint8_t *donnee, const uint8_t longueur):
  Lecture qui se fait par bloc qui prend en paramètre l'adresse, un pointeur vers les données, et la longueur(127 et moins) de la chaine à lire.

- Écriture :
  -uint8_t ecriture(const uint16_t adresse, const uint8_t donnee):
  Elle prend en paramètre l'adresse, un pointeur vers les données et copie les données dans l'adresse de la mémoire externe

  -uint8_t ecriture(const uint16_t adresse, uint8_t *donnee, const uint8_t longueur) :
  Elle prend en paramètre l'adresse, un pointeur vers les données et la longueur de la chaine.

Class Can du fichier can.h

Objectif : Cette classe permet l'accès au convertisseur analogique/numérique du microcontrôleur ATMega16

Constructeur : Le constructeur initialise le convertisseur.

Méthode implémentée : il n'y a qu'une méthode, celle de lecture.

-uint16_t lecture(uint8_t pos): Cette fonction prend en paramètre pos qui est un entier sur 8 bits représentant la broche du port a connecté au capteur analogique .Elle retourne la valeur numérique correspondant à la valeur analogique sur le port A. Ici, il faut faire un décalage de deux bits vers la droite pour manipuler les 8 bits significatifs, car ses deux derniers bits ne sont pas importants.

Également, plusieurs fonctions composent notre librairie. Elles ont été implémentées individuellement dans des fichiers (.cpp). Il s'agira de les décrire.

Fonction PWM_delayVariable du fichier PWM_delayVariable.h

Cette fonction est de type void et est axée sur la fonction delay_us qui effectue un délai de 1 microseconde pendant une période souhaitée. Elle a pour rôle de modifier le PWM par la fréquence voulue, et le pourcentage désiré tout en contrôlant sa durée.

-void PWM(int a, int b, int c) : Cette fonction prend en paramètre trois entiers qui sont respectivement le produit du pourcentage et de la période, la période et le nombre de fois que notre boucle est effectuée pour atteindre le temps voulu.

Fonction Debounce du fichier Debonce.h

Cette fonction est de type booléen. Elle a pour rôle de vérifier si le bouton poussoir est appuyé. Dans le cas où il est appuyé, elle retourne un true, et dans le cas contraire, elle retourne un false.

-bool CheckIfPressed() : Cette fonction ne prend aucun paramètre et retourne une valeur booléenne qui indique si le bouton-poussoir est pesé ou non. Pour son fonctionnement, le Port D doit être en mode sortie.

Fonction Affichagememoire du fichier Affichagememoire.h

Comme son nom l'indique, cette fonction permet l'affichage du contenu de la mémoire flash du atmega324pa. L'affichage est possible par le programme serieViaUSB et le port RS232.

-void affichageMemoire() : Cette fonction ne prend aucun paramètre et ne retourne rien.

Fonction Minuterie du fichier Minuterie.h

Cette fonction permet d'initialiser et de contrôler un compteur à l'aide d'un des 3 compteurs matériels présents sur le microcontrôleur.

-void partirMinuterie(uint16_t duree): Cette fonction prend en paramètre un entier sur 16 bits représentant la durée. Rien n'est retourné par la fonction.

Fonction DEL du fichier DEL.h

Cette fonction permet d'allumer la DEL libre du robot selon les paramètres passés.

-void DEL(int couleurHexa, char port, int rangeePin): Cette fonction prend en paramètre une valeur hexadécimale (0x00 = Eteint, 0x01 = Rouge, 0x02 = Vert) qui détermine la

couleur de la DEL, un char (A, a, B, b, C, c, D, d) qui détermine le port mis en mode sortie et un int (0, 2, 4 , 6) qui détermine la rangé de pins en sortie. Rien n'est retourné par la fonction.

Fonction Sensorlumiere du fichier Sensorlumiere.h

Elle a pour but de changer la couleur de la DEL dépendamment de l'éclairage présent dans le milieu: en effet, quand l'éclairage est super élevé, elle prend la couleur rouge. Cependant lorsque cette dernière est basse, elle prend une couleur verte, et quand l'éclairage est moyen, elle prend la couleur ambrée.

-void SensorLumiere(): Cette fonction ne prend aucun paramètre et ne retourne rien.

Fonction initialisationIntExterne du fichier InterruptionExterne.h

Cette fonction initialise tous les éléments nécessaires pour permettre une interruption externe du robot. Entre autres, la mise en sortie du port B, l'ajustement du registre EIMSK à la pin INT0 et l'ajustement du registre EICRA à ISC00 pour permettre les interruptions par bouton-poussoir. Lors de cette initialisation, les interruptions sont bloquées par cli().

-void initialisationIntExterne(void) : Cette fonction ne prend rien en paramètre et ne retourne rien.

Partie 2 : Décrire les modifications apportées au Makefile de départ

- **Makefile de la librairie :**

PROJECTNAME= libstatique
Commande pour nommer la librairie.

PRJSRC= $(wildcard *.cpp)
Commande pour récupérer tous les fichiers .cpp dans le répertoire.

AR=ar
Commande qui permet l'ajout d'une librairie statique.

TRG=$(PROJECTNAME).a
Commande qui spécifie l'extension du fichier libstatique, donc libstatique.a est la cible par défaut.

```
$(TRG) : $(OBJDEPS)
        $(AR) -crs $(TRG) $(OBJDEPS)
```
Commande qui fait l'implémentation de la cible AR avec son flag -crs.

- **Makefile de l'exécutable :**

PROJECTNAME= MakeFileExecutable

Commande pour nommer la librairie.

PRJSRC= $(wildcard *.cpp)
Commande pour récupérer tous les fichiers .cpp dans le répertoire.

INC= -I .. /lib_dir
Commande pour spécifier le chemin des inclusions additionnelles.

LIBS= -L ../lib_dir/-llibstatique
Commande pour spécifier le chemin vers la librairie à lier.