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 6165

Noms:

Francis Lavigueur
Rostan Lord-Belleville
Bei Ning Pan
Virasone Manibod

Date:
12 mars 2018

Partie 1 : Description de la librairie

La librairie contient d'abord le fichier ambre.cpp. Ce fichier contient la fonction ambre() nous permettant d'afficher la couleur ambre. Donc, elle fait en sorte que la del alterne de couleur entre la rouge et vert pour tromper l'œil. Dans le fichier ambre.h, on retrouve les constantes pour afficher la couleur vert, rouge et pour éteindre la del. Il y a aussi l'inclusion des fichiers avr/io.h et util/delay.h pour faire des delays. On a défini la fréquence du CPU pour que les delays puissent se faire dans les bons temps.

Ensuite, il y a le fichier InterruptBouton.cpp. Ce fichier contient la fonction initialisation() nous permettant de faire des interruptions avec un bouton-poussoir. Cette fonction initialise tous les registres nécessaires pour permettre ce genre d'interruption. Le fichier InterruptBouton.h *include* les mêmes fichiers qu'ambre.h mais avec le ficher avr/interrupt.h en plus pour pouvoir faire des interruptions.

Les fichiers mémoire_24.cpp et mémoire_24.h, qui ont été fournis par le professeur, ont aussi été mis dans la librairie pour que l'on puisse écrire et lire dans la mémoire du ATmega324PA.

Pour continuer, on retrouve le fichier pwm.cpp qui contient la fonction qui permet de donner un PWM à une del ou aux moteurs. La fonction prend en paramètre un uint16_t qui est la valeur de comparaison pour le compteur. Aussi, la fonction ajuste tous les registres nécessaires pour permettre d'ajuster le PWM de façon matérielle.

Enfin, on peut retrouver le fichier UART.cpp dans la librairie. Ce fichier contient la fonction initialisationUART() contenant les instructions permettant

de communiquer avec le PC. Comme avec les fonctions précédentes, les modifications des registres du ATmega324PA sont faites dans cette fonction. Ensuite, on retrouve la fonction transmissionUART qui prend en paramètre la donnée qui sera initialiser au registre UDR0. Le fichier UART.h *include* avr/io.h, delay_basic.h, delay.h, avr/interrupt.h et memoire_24.h.

Partie 2 : Décrire les modifications apportées au Makefile de départ

Makefile de la librarie :

En premier lieu, le Makefile de la librarie a été modifier pour qu'il puisse inclure tous les fichiers sources .cpp. Ceci a été réalisé en attribuant à la variable PJCSRC tous les fichiers cpp. Pour ce faire, la fonction wildcard a été appelée. La ligne originale a donc été remplacée par PRJSRC = $(wildcard *.cpp).

La création des fichiers .o est réutilisée du Makefile original. Il n'y a donc eu aucune modification aux lignes suivantes :

```
# Production des fichiers object
# De C a objet
%.o: %.c
    $(CC) $(CFLAGS) -c $<
# De C++ a objet
%.o: %.cpp
    $(CC) $(CFLAGS) $(CXXFLAGS) -c $<
```

La dernière modification apportée au Makefile consiste à remplacer la section qui transforme les fichiers .o en .hex et les installe sur le ATmega324PA par un ligne qui assemble tous les .o en une archive. Cette section a donc été remplacée entièrement :

```
# Production des fichiers hex a partir des fichiers elf
%.hex: %.out
    $(OBJCOPY) -j .text -j .data \
        -O $(HEXFORMAT) $< $@

# Make install permet de compiler le projet puis
# d'ecrire le programme en memoire flash dans votre
# microcontroleur. Celui-ci doit etre branche par cable USB
install: $(HEXROMTRG)
    $(AVRDUDE) -c $(AVRDUDE_PROGRAMMERID)   \
    -p $(MCU) -P -e -U flash:w:$(HEXROMTRG)
```

Par ces lignes qui créent une archive library.a avec tous les fichiers .o qui sont stockés dans OBJDEPS :

```
#creation de l'archivre
library.a: $(OBJDEPS)
    avr-ar -rcs library.a $(OBJDEPS)
```

Makefile du code :

Les seules lignes qui ont été modifiées sont celles qui servent à inclure la librarie dans notre code. INC = a donc reçu le chemin menant à la librarie et LIBS la librarie elle-même.

```
# Inclusions additionnels (ex: -I/path/to/mydir)
INC= -I./library/

# Libraires a lier (ex: -lmylib)
LIBS= library/library.a
```

Le rapport total ne doit par dépasser 7 pages incluant la page couverture.

Barème: vous serez jugé sur:

- *La qualités et le choix de vos portions de code choisies (5 points sur 20)*
- *La qualités de vos modifications aux Makefiles (5 points sur 20)*
- *Le rapport (7 points sur 20)*
  - *Explications cohérentes par rapport au code retenu pour former la librairie (2 points)*
  - *Explications cohérentes par rapport aux Makefiles modifiés (2 points)*
  - *Explications claires avec un bon niveau de détails (2 points)*
  - *Bon français (1 point)*
- *Bonne soumission de l'ensemble du code (compilation sans erreurs, …) et du rapport selon le format demandé (3 points sur 20)*