ECE 375 LAB 2

C -> Assembly -> Machine Code -> TekBot

Lab Time: Wednesday 5-7pm

Rhea Mae V. Edwards

INTRODUCTION

The purpose of this first lab is to provide a deeper understanding of the seven general-purpose input-output (I/O) ports programmed onto the ATmega128 microcontroller, by implementing the wanted movement behavior onto the ATmega128 microcontroller written in the C language.

PROGRAM OVERVIEW

The BumpBot program provides the basic behavior that allows the TekBot to react to whisker input. The TekBot has two forward facing buttons, or whiskers, a left and a right whisker. By default the TekBot will be moving forward until one of the whiskers are triggered. If the left whisker is hit, then the TekBot will backup and then turn right for a bit, while a right whisker or when both whiskers are hit will backup and turn left. After the either whisker routine completes, the TekBot resumes its forward motion.

Within a while loop that runs forever, the HitRight, HitLeft, HitBoth routines provide the basic functionality for handling either a Right or Left whisker hit, respectively. Additionally a _delay_ms() function was provided allowing time for the TekBot to backup, turn, and moving forward between these commands.

INITIALIZATION ROUTINE

The initialization routine provides a one-time initialization of key registers that allow the BumpBot program to execute correctly. First Port B is configured to accept inputs and outputs and then initialized an initial value for Port B outputs; which initially disables both motors. Port D is then configured to accept inputs and outputs.

MAIN ROUTINE

The Main routine executes a while loop that runs forever and checks to see if a whisker was hit. This is accomplished by first reading 8-bits of data from PIND reading the left and right whisker bits. The TekBot is first set to move forward for about 500 milliseconds. Then the data collected by reading PIND is checked to see if the right whisker is hit and if so, then it follows the HitRight routine (explained below). The Main routine then else checks to see if the left whisker is hit or if both of the whiskers have been hit and if so, then it follows the HitLeft routine (explained below) all being contained in a loop that runs forever.

HITRIGHT ROUTINE

The HitRight routine first moves the TekBot backwards for 500 milliseconds by first sending the Move Backwards command to PORTB followed by the use of the _delay_ms() function. Then the Turn Left command is sent to PORTB to get the TekBot to turn left for 500 milliseconds by the use of the _delay_ms() function.

HITLEFT AND HITBOTH ROUTINES

The HitLeft and HitBoth routines are identical to the HitRight routine, except that a Turn Right command is sent to PORTB instead.

STUDY QUESTIONS

1. Some benefits of writing code in a language like C that can be “cross compiled” are that it can be widely used by being less complex and straight to the point, and also by writing code in such a language, that program can be used to program multiple platforms from one location.
   Some drawback of writing code this way are that you can never always be sure that two different platforms will read the cross compiled type of program the same way, and that the space to run the program may not be efficient enough across the multiple of platform being used to run the program.

2. The size of my Lab 1 output .hex files is 485 bytes.
   The size of my Lab 2 output .hex files is 913 bytes.
   Even though they both pretty much perform the same BumpBot behavior, there is a size difference between these two file where my Lab 2 is almost double the size of my Lab 1. The reason being is because a program written in C such as my Lab 2, versus a program written in just assembly code such as the sample given in my Lab 1, contains a greater amount of content within just the program itself, compared to an assembly program, even though both types of programs pretty much perform the same type of behavior.

DIFFICULTIES

The code can be seen and written in a simpler way than one may think, and that the visualization of the movement of a TekBot being displayed on the AVR board just takes some time getting used to.

CONCLUSION

In this lab we are given a program that causes the TekBot to “dance” represented with the AVR board. By reading and understanding that given code, we are required and given the opportunity to program the AVR board perform the same/similar behavior that was illustrated form the week’s before lab, while understanding the variety of responsibilities the registers are given. Overall the lab allowed us to understand the various ways to write programs for the AVR board that can represent the same work being done.

SOURCE CODE

```
SAMPLE CODE:

/*
This code will cause a TekBot connected to a mega128 board to 'dance' in a cool
pattern. No pins are used as input, and four Port B pins are used for output.

PORT MAP
Port B, Pin 4 -> Output -> Right Motor Enable
Port B, Pin 5 -> Output -> Right Motor Direction
Port B, Pin 7 -> Output -> Left Motor Enable
Port B, Pin 6 -> Output -> Left Motor Direction
*/
#define F_CPU 16000000
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>

int main(void)
{
      DDRB = 0b11110000;      // configure Port B pins for input/output
      PORTB = 0b11110000;     // set initial value for Port B outputs
                              // (initially, disable both motors)
      while (1) { // loop forever

            PORTB = 0b01100000;     // make TekBot move forward
            _delay_ms(500);         // wait for 500 ms
            PORTB = 0b00000000;     // move backward
            _delay_ms(500);         // wait for 500 ms
            PORTB = 0b00100000;     // turn left
            _delay_ms(1000);        // wait for 1 s
            PORTB = 0b01000000;     // turn right
            _delay_ms(2000);        // wait for 2 s
            PORTB = 0b00100000;     // turn left
            _delay_ms(1000);        // wait for 1 s
      }
}
```

LAB SOURCE CODE:

```
/*
This code will cause a TekBot connected to the AVR board to
move forward and when it touches an obstacle, it will reverse
and turn away from the obstacle and resume forward motion.

PORT MAP
Port B, Pin 4 -> Output -> Right Motor Enable
Port B, Pin 5 -> Output -> Right Motor Direction
Port B, Pin 7 -> Output -> Left Motor Enable
Port B, Pin 6 -> Output -> Left Motor Direction
Port D, Pin 1 -> Input -> Left Whisker
Port D, Pin 0 -> Input -> Right Whisker
*/
#define F_CPU 16000000
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>

int main(void)
{
       DDRB = 0b11110000;      // configure Port B pins for input/output
       PORTB = 0b11110000;     // set initial value for Port B outputs
       // (initially, disable both motors)

       DDRD = 0b00000000;      // configure Port D pins for input/output

       while (1) { // loop forever

              PORTB = 0b01100000;     // make TekBot move forward
              _delay_ms(500);         // wait for 500 ms

              if (PIND == 0b11111110) { //Right Whisker Hit
                     PORTB = 0b00000000;     // move backward
                     _delay_ms(500);         // wait for 500 ms
                     PORTB = 0b00100000;     // turn left
                     _delay_ms(500);        // wait for 500 ms
              }
              //Left Whisker and Both Whiskers Hit
              else if ((PIND == 0b11111101) | (PIND == 0b11111100)) {
                     PORTB = 0b00000000;     // move backward
                     _delay_ms(500);         // wait for 500 ms
                     PORTB = 0b01000000;     // turn right
                     _delay_ms(500);        // wait for 500 ms
              }
       }
}
```

CHALLENGE SOURCE CODE:

```
/*
```

```
This code will cause a TekBot connected to the AVR board to
move forward and when it touches an obstacle, it will reverse
and turn toward the obstacle and resume forward motion,
pushing/moving into the obstacle in front of it.

PORT MAP
Port B, Pin 4 -> Output -> Right Motor Enable
Port B, Pin 5 -> Output -> Right Motor Direction
Port B, Pin 7 -> Output -> Left Motor Enable
Port B, Pin 6 -> Output -> Left Motor Direction
Port D, Pin 1 -> Input -> Left Whisker
Port D, Pin 0 -> Input -> Right Whisker
*/
#define F_CPU 16000000
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>

int main(void)
{
        DDRB = 0b11110000;      // configure Port B pins for input/output
        PORTB = 0b11110000;     // set initial value for Port B outputs
        // (initially, disable both motors)

        DDRD = 0b00000000;      // configure Port D pins for input/output

        while (1) { // loop forever

                PORTB = 0b01100000;     // make TekBot move forward
                _delay_ms(500);         // wait for 500 ms

                if (PIND == 0b11111110) { //Right Whisker Hit
                        PORTB = 0b00000000;     // move backward
                        _delay_ms(500);         // wait for 500 ms
                        PORTB = 0b01000000;     // turn right
                        _delay_ms(500);        // wait for 500 ms
                }
                else if ((PIND == 0b11111101)) { //Left Whisker Hit
                        PORTB = 0b00000000;     // move backward
                        _delay_ms(500);         // wait for 500 ms
                        PORTB = 0b00100000;     // turn left
                        _delay_ms(500);        // wait for 500 ms
                }
                else if ((PIND == 0b11111100)) { //Both Whiskers Hit
                        PORTB = 0b00000000;     // move backward
                        _delay_ms(500);         // wait for 500 ms
                }
        }
}
```