ECE 375 LAB 2

C -> Assembler -> Machine Code

Lab Time: Friday 16:00 ~ 17:50

Hyunjae Kim

INTRODUCTION

The purpose of this lab is to code the program in C language which acts same as the assembly program provided from the previous lab, and understanding the concept of PORT and PIN. By applying PIN and PORT in C program, the "BumpBot" will be implemented in C language in Atmel studio, building the project, and then using the Universal Programmer to download the program onto ATmega 128 board.

PROGRAM OVERVIEW

The BumpBot program allows the Bumpbot to react differently depending on the input from whisker. The TekBot has two forward facing buttons(Whiskers), a left and a right whisker. If there is no input from both whisker, TekBot will keep moving forward.

If there is an input from right whisker, TekBot will go backward for a second, turn left for a second, and then keep moving forward until there are inputs from whiskers.

If there is an input from left whisker, TekBot will go backward for a second, turn right for a second, and then keep moving forward until there are inputs from whiskers.

If there are inputs from both whiskers, TekBot will act same as when the right whisker is hit.

INITIALIZATION ROUTINE

The Bumpbot program is initialized by setting the I/O ports.

First, Port B was be initialized as outputs of motors by setting DDR B and PORT B as outputs. Then Port D was initialized as inputs from whiskers by setting DDR D and PORT D as inputs.

Finally, after initializing I/O ports, the Bumpbot program executes a loop which starts moving forward command.

MAIN ROUTINE

The main routine of Bumpbot is keep moving forward, and then checks the inputs from PIN D. The input from right whisker is connected to bit0 of PIN D, and the input from left whisker is connected to bit1 of PIN D. Bumpbot goes to subroutine if the bit 0 or bit 1 of PIN D are zero. (The whisker is active low.)

After completing the subroutines, the Bumpbot returns to the main routine, and then keep checking the conditions of subroutines.

SUBROUTINES

1. HitRight Routine

   The HitRight routine first moves the Bumpbot backwards by sending the command to PORT B and then delays sending next command for a second. Then Then Bumpbot turns left by sending the command to PORT B and then delays sending next command for a second. Finally, the HitRight routine sends a command to PORT B in order to keep moving forward (returning to main routine).

2. HitLeft Routine

The HitLeft routine is identical to the HitRight routine, except sending Turn Right command to PORT B. Then the HitLeft routine sends Go Forward command to PORT B in order to return to the main routine.

3. HitBoth Routine

The HitBoth routine in identical to the HitRight routine. If the bit 0 and bit 1 of PIN D are low(zero), then the HitBoth routine executes the same process of commands of the HitRight routine.

STUDY QUESTIONS

1. This lab required you to compile two C programs (one given as a sample, and another that you wrote) into a binary representation that allows them to run directly on your mega128 board. Explain some of the benefits of writing code in a language like C that can be "cross compiled". Also, explain some of the drawbacks of writing this way.

The benefit of languages which can be cross compiled is that it is flexible to program a code, and does not require any setups before programming. However, these languages converted two times in the process of compiling. First, it is converted to assembly code, then converted to machine language. Passing the two processes can be a delay in executing the program.

2. The C program you just wrote does basically the same thing as the sample assembly program you looked at in Lab 1. What is the size (in bytes) of your Lab 1 & Lab 2 output .hex files? Can you explain why there is a size difference between these two files, even though they both perform the same BumpBot behavior?

(1) Lab1 output .hex file was 1Kbyte and Lab 2 output .hex file was 2Kbyte.

(2) Even though they both perform the same behavior, the C program has more size of the file because there are additional commands to convert the C program file to assembly file, and then convert the assembly file to machine language. This is the reason why these two files have size difference.

DIFFICULTIES

In the process of programming the TekBot in C program, the inputs of subroutines were swapped. The reason was that there was a misunderstanding of the property of whiskers – active high. By setting the inputs of subroutines as active low, the subroutine worked properly.

CONCLUSION

By programming the Bumpbot in C language, there were differences in C program and Assembly program even if they both perform the same action. Moreover, by comparing both programs, it is predicted what form of Assembly code if the C program is converted to Assembly code.

By controlling I/O, the concepts of PORT, PIN, and DDR were showing how the AVR board communicate with other external machines.

Through the lab activity, the same program coded in much higher language and communication between the board and external machines were observed and applicated.

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
*/

//Author: Hyunjae Kim
//Date: 01/14/2022

#define F_CPU 16000000
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>

int main(void)
{
        DDRB = 0b11110000; // set bit 7~4 as outputs.
        PORTB = 0b11110000; // turn on led connected to bit 7~4.

        DDRD = 0b00000000; // set bit 7~0 all inputs.
        PORTD = 0b00000011; // enable pull-up register of bit 1~0.
        while (1) // loop forever
        {
                PORTB = 0b01100000; //go forward unless left or right whisker being hit

                if(PIND == 0b11111110 ){ // when the right whisker is hit. When the pin is
pressed,  the input going to the bus is zero.
                        PORTB = 0b00000000; // go backward
                        _delay_ms(1000); // for a second
                        PORTB = 0b00100000; // go leftward
                        _delay_ms(1000); // for a second
                        PORTB = 0b01100000; // after turning left, keep go forward
                }
                if(PIND == 0b11111100 ){ // when  both whisker are hit. When both pins are
pressed,  the inputs going to the bus are both zero. Act same as when the right whisker is hit.
                        PORTB = 0b00000000; // go backward
                        _delay_ms(1000); // for a second
                        PORTB = 0b00100000; // go leftward
                        _delay_ms(1000); // for a second
                        PORTB = 0b01100000; // after turning left, keep go forward
                }

               if(PIND == 0b11111101){ // when the left whisker is hit.
               //When the pin is pressed, the input going to the bus is zero.
                        PORTB = 0b00000000;// go backward
                        _delay_ms(1000); // for a second
                        PORTB = 0b01000000; // go rightward
                        _delay_ms(1000); // for a second
                        PORTB = 0b01100000; // after turning right, keep go forward.
                }

        }
}
```

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 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
*/

// Author: Hyunjae Kim
// Date: 01/14/2022

#define F_CPU 16000000
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>

int main(void)
{
        DDRB = 0b11110000; // set bit 7~4 as outputs.
        PORTB = 0b11110000; // turn on led connected to bit 7~4.

        DDRD = 0b00000000; // set bit 7~0 all inputs.
        PORTD = 0b00000011; // enable pull-up register of bit 1~0.
        while (1) // loop forever
        {
                PORTB = 0b01100000; //go forward unless left or right risker being hit

                if(PIND == 0b11111100){ //When both whiskers are hit.
                      //When the pins are pressed, the inputs going to the bus are zero.
                         PORTB = 0b01100000; //go forward.
                         _delay_ms(500); // for 0.5seconds.
                         PORTB = 0b00000000; // go backward
                         _delay_ms(500); // for 0.5 seconds.
                         PORTB = 0b01100000; // after going backwards keep going forward.
                }
                if(PIND == 0b11111110){ // when the right risker is hit.
               //When the pin is pressed,  the input going to the bus is zero.
                         _delay_ms(100);
                         PORTB = 0b00000000; // go backward
                         _delay_ms(1000); // for a second
                         PORTB = 0b01000000; // go rightward
                         _delay_ms(1000); // for a second
                         PORTB = 0b01100000; // after turning left, keep go forward
                }
                if(PIND == 0b11111101){ // when the left risker is hit.
                      //When the pin is pressed, the input going to the bus is zero.
                         _delay_ms(100);
                         PORTB = 0b00000000;// go backward
                         _delay_ms(1000); // for a second
                         PORTB = 0b00100000; // go leftward
                         _delay_ms(1000); // for a second
                         PORTB = 0b01100000; // after turning right, keep go forward.
                }


        }
}
```