ECE 375 LAB 2

C->Assembly->Machine Code->TekBot

Lab Time: Tuesday 4-6

Alexander Uong

INTRODUCTION

The purpose of this lab is to achieve a greater understanding of how to program our Tekbot, and how to write actual code to configure the Tekbot to our needs. This lab involves learning about what specific "BumpBot" behavior means, and understanding the ports and pins provided on our Tekbot.

This lab's main objective is to write a C program that leads to the basic BumpBot routine seen in lab 1. The code provided to us in the first lab was written in assembly language, so the goal is to convert that program into a C program that provides the same expected behavior.

PROGRAM OVERVIEW

This program allows the Tekbot to behave in the same manner as in lab 1, except the code will be written in C, rather than assembly language. The Tekbot reacts to whisker input, using a left and a right whisker. In this case, the left and right whiskers are Port D pin 0 and Port D pin 1 respectively. The Tekbot will continue to move until it encounters an object (i.e. the left, right, or both whiskers have been pressed). Once a whisker has been pressed, the Tekbot will move backwards, turn away from the object, and move forward again.

In my program, I will initially configure the Port B pins on the Tekbot for input and output, as well as setting the initial values for the Port B outputs; the initial values are setting bit 6 and bit 5 to be turned on, meaning the left and right motor are forward. The inputs are the pins on the Tekbot and the LED's are the outputs.

The program code will be within an infinite while loop, meaning the program will continue to run on the Tekbot as long as the Tekbot is connected. Within the while loop are a set of if statements, checking if the user presses the left, right, or both whiskers. The command used here is: `if(PIND == 0b11111110)`. This is for the case if the right whisker is pressed. The 0 indicates that the certain pin at that location has been pressed (there are eight pins).

If the right whisker is pressed, I set bit 6 and bit 5 to off, indicating the left and right motor are backwards. I then use the command `_delay_ms(1000)`, to indicate to the program to wait 1 second until executing the next line of code. I then turn bit 5 on, meaning the left motor is backward and the right motor is forward. This is for the Tekbot to turn left. I then use the command `_delay_ms(1000)` again for the program to wait 1 second until executing the next line of code. Following this, I turn bit 6 on, meaning the left motor is now forward along with the right motor. The Tekbot is now moving forward once again.

This same logic is used for when the left whisker is pressed, except rather than turning bit 5 on after setting bit 6 and bit 5 to off, I set bit 6 to on. This instead means the Tekbot is turning right, with the left motor forward and the right motor backward. I then proceed to turn bit 5 on, allowing the Tekbot to move forward again.

If both whiskers are pressed, the Tekbot's behavior will remain exactly the same as if only the right whisker is pressed.

ADDITIONAL 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.

Benefits of writing code in a language that can be cross compiled is it gives you the ability to run your program or create an executable code for any system or platform you would like. Clearly the benefits are modularity and flexibility, being able to generate code that runs on other platforms and systems. Disadvantages to this are the program/executable code may not be as optimized on another system or platform. Cross compiling also may produce messy code that will be hard to maintain or update on another system.

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?

Given my case, the lab 1 .hex file written in assembly was 294 bytes, as the lab 2 .hex file written in C was 326 bytes. This could due to a number of reasons. First off, my C program simply may not be as efficient as it can be. This could lead to the C program being a larger size. Secondly, it may depend on the compiler itself. The compiler used may not create code that is as compact as it can be. I also believe the C program we were asked to write included a case for if both whiskers were hit at the same time. I believe that in the lab 1 .hex file, there was no case to handle this.

CONCLUSION

For this lab, we were required to write a functional C program that models the behavior of the BumpBot routine seen in lab 1. We are then asked to compile the project and download it into our Tekbot bases. In conclusion, this lab provided me a greater understanding of how to configure my Tekbot Atmega128 microcontroller, as well as understanding the Tekbot's behavior. This lab also provided me the opportunity to get a glimpse of how to write a functional C program for our Tekbot, as well as understanding how to convert an assembly language program into a C program, and vice versa.

SOURCE CODE

Provide a copy of the source code. Here you should use a mono-spaced font and can go down to 8-pt in order to make it fit. Sometimes the conversion from standard ASCII to a word document may mess up the formatting. Make sure to reformate the code so it looks nice and is readable.

```
;***********************************************************


/*
 * Alexander_Uong_Lab2_sourcecode.c
```

```
 *
 * Created: 1/18/2021 4:26:59 PM
 * Author : Alex Uong
 */

/*
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 = 0b01100000;	//set initial value for Port B outputs

	while (1) // loop forever
	{
		if(PIND == 0b11111100){		//if both left and right whisker are
pressed, do the following
			PORTB = 0b00000000;		//move backward
			_delay_ms(1000);		//wait 1 second
			PORTB = 0b00100000;		//turn left
			_delay_ms(1000);		//wait 1 second
			PORTB = 0b01100000;		// move forward
		}
		if (PIND == 0b11111101){	//if left whisker is pressed, do the following
			PORTB = 0b00000000;		//move backward
			_delay_ms(1000);		//wait 1 second
			PORTB = 0b01000000;		//turn right
			_delay_ms(1000);		//wait 1 second
			PORTB = 0b01100000;		// move forward
		}
		if(PIND == 0b11111110){		//if right whisker is pressed, do the
following
			PORTB = 0b00000000;		//move backward
			_delay_ms(1000);		//wait 1 second
			PORTB = 0b00100000;		//turn left
			 delay ms(1000);		//wait 1 second
			PORTB = 0b01100000;		//move forward
		}

	}
}
```

CHALLENGE CODE

```
/*
 * lab2challengecode.c
 *
 * Created: 1/18/2021 5:58:24 PM
 * Author : Alex Uong
 */
```

```
#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 = 0b01100000;	//set initial value for Port B outputs

	while (1) // loop forever
	{
		if(PIND == 0b11111100){		//if both left and right whisker are
pressed, do the following
			PORTB = 0b01100000;		//move forward
			_delay_ms(500);			//wait half a second
			PORTB = 0b00000000;		//moves backwards slightly
			_delay_ms(500);			//wait half a second
			PORTB = 0b01000000;		// turn slightly toward object (in this case
turn right)
			 delay ms(500);			//wait half a second
			PORTB = 0b01100000;		//move forward again
		}
		if (PIND == 0b11111101){	//if left whisker is pressed, do the following
			PORTB = 0b01100000;		//move forward
			 delay ms(500);			//wait half a second
			PORTB = 0b00000000;		//moves backwards slightly
			_delay_ms(500);			//wait half second
			PORTB = 0b00100000;		//turn slightly toward object (in this case
turn left)
			_delay_ms(500);			//wait half a second
			PORTB = 0b01100000;		// move forward
		}
		if(PIND == 0b11111110){		//if right whisker is pressed, do the
following
			PORTB = 0b01100000;		//move forward
			_delay_ms(500);			//wait half a second
			PORTB = 0b00000000;		//moves backwards slightly
			_delay_ms(500);			//wait half a second
			PORTB = 0b01000000;		// turn slightly toward object (in this case
turn right)
			_delay_ms(500);			//wait half a second
			PORTB = 0b01100000;		//move forward again
		}

	}
}
```