Lab 7: Implementation of a Robot

EE 234: Section 2



Michael Mentele

Partner: Jonathan Quissenberry

November 13, 2012

**Introduction**

In this lab we are going to complete our ROBO-MAL program from lab 5 and flush out the RCI commands. We will implement the four “figure skating” routines developed with our RCI commands. The routines are shown in the table below:

|  |  |
| --- | --- |
| Routine | Description |
| Square | Traced an approximate 2’ square using the forward and right command. |
| Circle | Drove a circle of approximately 8 inches using the forward command and setting the strength of the wheels diseparately. |
| Figure Eight | Used multiple forward commands and essentially performing the circle command twice but reversing relative wheel speeds for the second circle. |
| Triangle | We performed this by using forward and multiple right turns to get a turn of 120 degrees because the maximum angle one ‘Right’ command can do is 90 degrees. |

\*All programs end with a ‘Halt’ command

We used the timer and the output compare module and communicated to the DC motors on the board using the HB5 peripheral bridge. To do this easily we used the PWM mode on the output compare module to generate a signal that we can easily manipulate the duty cycle of.

The major change in the lab (given in the pre-lab) was that we altered the forward command to control both motors independently. This made our control more precise and enabled us to perform the routines necessary without adding additional commands.

**Software**

In this lab we essentially just flushed the code out from Lab 5. The main differences in this lab are that we will be using timers and output compare modules to control the DC motor. So from lab 5 we are adding a few setup portions to the code for timers and OC modules.

Please refer to the Lab 5 flow chart in the Appendix. In this section we will discuss the changes made to the lab. The software setup for lab seven is simply implementation of timer 2 and the output compare modules 2 and 3.

The first task is setting and enabling your timer to count. The period was set to 10010 to simplify operations. However it is worth remembering that we now need to change by decimal or at least be aware of the decimal values we add. You set timer 2 by using the T2CON register to disable the timer and set the PR (period register). Then you enable pin 15 to enable your timer. We decided to use a single timer so that the motors would be synchronized or at least running off the same timer. Multiple timers would have added needless complications.

After the timers we need to set up OC2 and OC3 (chosen for port/pin convenience). The output pins OCx of these modules are used in PWM mode to control the enable pin and therefore the duty cycle of the motor. Differing average voltages depend on the duty cycles of OC2 and OC3 respectively. You set the OC1RS register after initializing the OCXR register because the OCXRS register is loaded in after every match.

You must set up the H-bridge so that it connects to the OC2 and OC3 on the Cerebot board and then the two motors respectively. For OC2 and OC3 the four main bits that control the direction and enable pins are on the D port. Bits 1 and 2 control the directions of the motors via the H-bridge. Bits 6 and7 control the enable pins and they are controlled by the OC modules. We protected our H bridge by:

1. Disabling timer 2
2. Set the pin the enable goes out on to an input
3. Clear the pins on LATD (enable)

To execute the RCI commands you need to be able to change the direction of the motors and the duty cycle of the PWM signal by changing the duration of the enables. To change the PWM the OCxRS register is given a value specified by the operand from whatever instruction you are executing.

The forward function operand allows the speed of the left and right wheel to be specified. The backwards command acts the same but with opposite directions on the wheels. When turning left and right the pivot wheel turns backwards while the other goes forward so as to give you a sharp turn. For turning we have a set speed at 50% duty cycle and we have calculated the speed per degree we will need.

One thing to note is since the motors are rotated 180 degrees from each other you have to have their spin directions opposite to have them go in the same direction.

**Test Procedure and Results**

**Methodology**

Testing was performed from a “Top Down” viewpoint where functionality is tested first followed by a debugging phase before finally being tested by an independent user. We also made great use of the step through function in the iterative design project. This greatly reduced the time it took to debug the program before getting a testable prototype.

**Procedure**

1. Test Correct Operation
   1. Test functionality
      1. Do the RCI commands all function individually?
         1. Forward
         2. Backward
         3. Left
         4. Right
         5. Brake
         6. Halt
      2. Does it need a minimum duty cycle to function?
   2. Test Routines
      1. Do the operations execute correctly in sequence?
         1. Square
         2. Circle
         3. Triangle
         4. Figure Eight
2. Let someone else play it!

Using Murphy’s Law… and another perspective; this is an effective way to achieve unexpected results.

**Results**

We began by testing the RCI commands individually. While testing the FORWARD command we noticed that the right motor is very slightly weaker than the leftmost. So we compensated by changing the motor speed slightly by changing the OC3RS register to be slightly more (thus increasing the duty cycle).

The turning parts took some experimentation to find an appropriate speed to turn with. We decided to use a 100% duty cycle because there seems to be a threshold torque where the robot won’t turn and we wanted to make sure for small angles it actually turned. BRAKE and HALT were very similar; all we did in BRAKE was disable everything and thus stopping the motors. In HALT we jumped to an endless loop after we ‘BRAKED.’ Both worked as expected. We also found that you need a minimum duty cycle of about 16% to get the wheels to turn at all.

As for the routines it was difficult to fine tune them without feedback and surfaces made significant differences for this open-loop system. However they all functioned close to as expected after minor tweaking. It took a while to get the right delay times at our turn speeds to achieve 90 degree turns but once we did so it was simple to do the other turns.

**Conclusion**

For the final project I think it will be important to go back and improve some of the code. We will likely want to add interrupt to improve control and fluidity and to simplify future interrupt integration.

The changes to FORWARD allow for a great deal of control and fluidity for our robot. What we could do from here is implement more functionality. It would be fun to incorporate some kind of feedback either in the form of remote control or sensor control so that the robot can do certain functions fluidly and react to stimuli. Of course this would mean we would need to add interrupts…

Appendix A

**Pre-Lab:**

Additional Command

Modification to the forward command so that speed is specified separately for each wheel. So for 42XX the first X corresponds to the speed of the left wheel. The second X corresponds to the speed on the right wheel.

Square:

|  |  |  |
| --- | --- | --- |
| Memory Cell | Instruction or Data Value | Description |
| 00 | 4222 | FORWARD at medium speed |
| 01 | 3300 | HALT program, robot stops |
| 02 | 4190 | RIGHT 90 degrees |
| 03 | 4222 | FORWARD at medium speed |
| 04 | 3300 | HALT program, robot stops |
| 05 | 4190 | RIGHT 90 degrees |
| 06 | 4222 | FORWARD at medium speed |
| 07 | 3300 | HALT program, robot stops |
| 08 | 4190 | RIGHT 90 degrees |
| 09 | 4222 | FORWARD at medium speed |
| 10 | 3300 | HALT program, robot stops |
| 11 | 4190 | RIGHT 90 degrees |
| 12 | 3300 | HALT program, robot stops |

Triangle:

|  |  |  |
| --- | --- | --- |
| Memory Cell | Instruction or Data Value | Description |
| 00 | 4222 | FORWARD at medium speed |
| 01 | 3300 | HALT program, robot stops |
| 02 | 4190 | RIGHT 90 degrees |
| 03 | 4130 | RIGHT 30 degrees |
| 04 | 4222 | FORWARD at medium speed |
| 05 | 3300 | HALT program, robot stops |
| 06 | 4190 | RIGHT 90 degrees |
| 07 | 4130 | RIGHT 30 degrees |
| 08 | 4222 | FORWARD at medium speed |
| 09 | 3300 | HALT program, robot stops |
| 10 | 4190 | RIGHT 90 degrees |
| 11 | 4130 | RIGHT 30 degrees |
| 12 | 3300 | HALT program, robot stops |

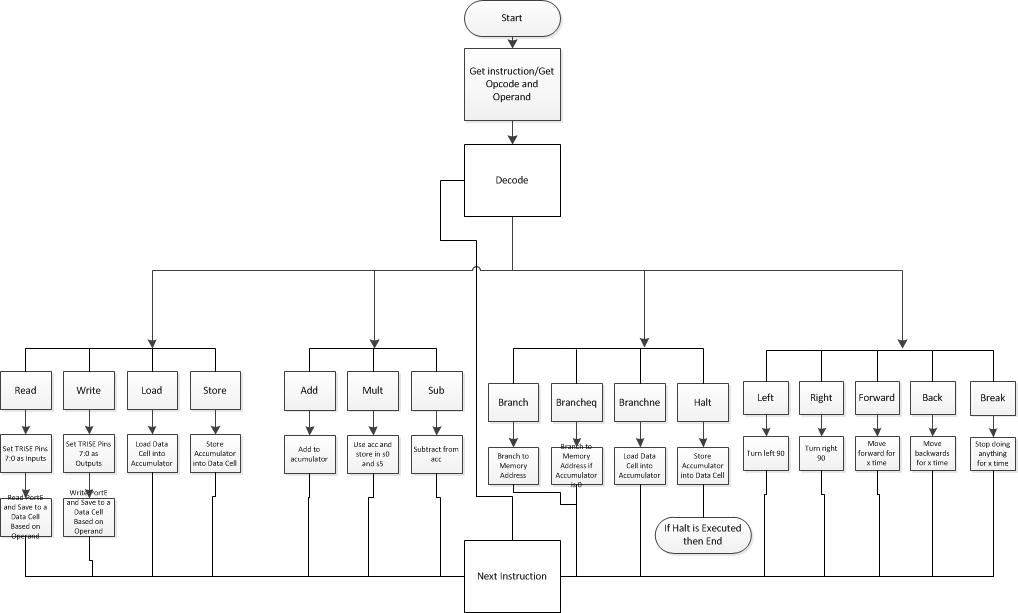
Figure Eight:

|  |  |  |
| --- | --- | --- |
| Memory Cell | Instruction or Data Value | Description |
| 00 | 4222 | FORWARD at medium speed |
| 01 | 4212 | FORWARD with a slow left wheel and a fast right wheel to turn left. |
| 02 | 4222 | FORWARD at medium speed |
| 03 | 4212 | FORWARD with a slow left wheel and a fast right wheel to turn left. |
| 04 | 3300 | HALT program, robot stops |

Circle:

|  |  |  |
| --- | --- | --- |
| Memory Cell | Instruction or Data Value | Description |
| 01 | 4212 | FORWARD with a slow left wheel and a fast right wheel to turn left. |
| 04 | 3300 | HALT program, robot stops |

Appendix B

****