DC Motor Control Project

The purpose of this lab is to synthesize the programming concepts learned in the first three 68HC11 labs along with some circuit design into a working controller for a DC Motor with user input from the LCD display. The PWM modulation technique learned in the third lab will be used to control the speed of the motor while pulse detection will also be implemented to detect the RPM of the motor. Lastly, the approximation types will be used to approximate the curve of the motor speed with respect to the duty cycle of the PWM.

Objectives

• Understand how to use the 68HC11 to detect pulses.
• Understand some of the constraints in designing a circuit to drive a DC Motor using the 68HC11.
• Apply Pulse Width Modulation (PWM) to the application of controlling the speed of a DC Motor.

Turn-In Requirements:

Student signature page plus answers, code, and T.A. signatures for parts 1-5.

References


Equipment for this lab:

• 68HC11 trainer kit, to include 68HC11 EVBU and prototyping strips
• IBM compatible PC to connect to the trainer kit via an RS-232 serial cable
• Agilent E3649A DC power supply
• Agilent 54621D oscilloscope
• Floppy disk provided by the student

<table>
<thead>
<tr>
<th>Part Description</th>
<th>Part Number</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Photo Emitter</td>
<td>160-1063-ND</td>
<td>1</td>
</tr>
<tr>
<td>Photo Detector</td>
<td>160-1065-ND</td>
<td>1</td>
</tr>
<tr>
<td>Mounted DC Motor with Slotted Disk</td>
<td>FE-260-18130</td>
<td>1</td>
</tr>
<tr>
<td>NPN General Transistor</td>
<td>2N222A</td>
<td>≤2</td>
</tr>
<tr>
<td>Rectifier Diode</td>
<td>1N4004</td>
<td>1</td>
</tr>
<tr>
<td>Resistors and Capacitors (Varying Sizes as Needed)</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
*In completing this lab, I have abstained from any form of academic dishonesty or deception (e.g. cheating, lying, stealing or plagiarism). This work is entirely of my own origin (beyond the code that was supplied to me by the instructor).

Name: __________________________            Signature: ___________________________

**Laboratory Exercise**

**Timing**

It is anticipated that this lab will take two weeks to complete.

**Notes**

- In this lab, all numbers such as addresses and data are given in hexadecimal format (“hex”) unless otherwise indicated. In completing the lab, record all information in hexadecimal unless directed differently.

- Make sure your development EVBU board is connected to power(green LED on board ON) and the serial port of the EVBU is connected to the serial port of your development PC containing the AXIDE software, configured to the correct port at the correct baud rate, etc.

- Be sure to bring your RTF file from the Timers lab. One additional piece of code is available on the EE-3306 Web site under RESOURCES. The file name is *neg_quad.RTF* and might be useful for implementing the approximation portion of the lab (see p.4 for details). The final code you submit should be named *DCMotor.RTF*.

- The motor’s speed **should not exceed 6000 RPM**. If the motor is shaking enough to move around the table, this is a good indication that the motor is moving too quickly.

- **IMPORTANT:** Do not touch the slotted disk while it is spinning. Doing so could result in bodily harm.

**Part 1—Fully Functional DC Motor Control**

In this lab you will modify the control interface you created in lab 2 and use the knowledge from lab 3 to create a PWM square wave to drive a DC motor. To accomplish this, you will need to add code to output the wave with a duty cycle proportional to the speed entered by the user (the equation used will be dictated by which approximation type is selected currently). You will also need to accept pulses generated by an emitter-detector pair, calculate the motor’s RPM, and display it on the screen. Below is a list similar to that of lab 2 with the specifications for the interface program. The TA will have a working example of one implementation.

When in “Display Mode”:
Increment Speed by 100………………………………… …….A  (same as Lab 1)
Decrement Speed by 100……………………………….............B  (same as Lab 1)
Double Speed…………………………………………………..C  (same as Lab 1)
Reset Speed and Approximation Type to Defaults……………*  (same as Lab 2)
Enter “Speed Manual Entry Mode”…………………………….#  (same as Lab 2)
Enter “Acceleration Type Entry Mode”………………………..D  (same as Lab 2)
**NEW:** Update the Display with the Current RPM…………….0

When in “Speed Manual Entry Mode”:
Enter Speed and Return to “Display Mode”……………….#  (same as Lab 2)
When in “Approximation Type Entry Mode”:
Use Linear Approximation……………………………………..0 (same as Lab 2)
Use Positive Quadratic Approximation………………………...1 (same as Lab 2)
Use Negative Quadratic Approximation………………………..2 (same as Lab 2)

In order to complete this lab, you will need two basic circuits. IMPORTANT: Your TA **must** verify your circuits before you supply power to them. The first circuit was discussed in the prelab and will use the switch circuit that you designed to control the speed of the motor. You will need to modify your program to control the duty cycle of the output wave from the 68HC11 to spin the motor according to (1) the speed entered and (2) the current approximation type:

- Linear: Duty Cycle = A*Speed + B
- Quadratic: Duty Cycle = A*Speed² + B*Speed + C

In these equations, A, B, and C are constants. Also, the positive quadratic approximation will be concave up while the negative quadratic approximation will be concave down. Your program will ONLY need to handle POSITIVE speeds and should be limited so that the maximum RPM of the motor does not exceed 6000. For convenience you may implement the speed so that it is initialized and resets to 100 (extra credit will be given for programs that properly implement zero speeds and/or negative speeds). The diode in Figure 1 is included because of the use of Pulse Width Modulation. During the period of time at which the 68HC11 output is logically low, the motor windings will want to continue to spin, creating a current flow through the switch when it is “open” which could damage the components of the switch. However, the addition of the diode allows current to continue to flow through an alternate path.

![Motor Control Circuit](image)

The second circuit that you will need will be a pulse detector circuit using the photo emitter-detector pair provided. The disk attached to the motor will spin between the emitter and detector, blocking the light except when the slot passes by them. The emitter of this pair is simply a light emitting diode (LED). When the light from the emitter strikes the detector, a voltage is created across the terminals of the transistor (the light basically takes the place of a base current). Your program must then use the output of the detector as input for the 68HC11 to read in the generated pulses. These pulses can then be used to calculate the RPM of the motor and display it on the LCD display. The general circuit is shown in Figure 2. Please enter the values of R1 and R2 used in your circuit below (you must design the circuit so that I_F = 20 mA).
The keypad input and LCD output program used in the labs 2 and 3 can be used to read display inputs. As previously noted, an additional piece of code is provided:

**neg_quad:** This file contains example code for the negative quadratic equation. The code assumes that accumulator A holds the approximation type and register X contains the speed. “High” refers to the time to be stored for the high portion of the square wave in a TOC register to cause the appropriate interrupt. The end result stored in high is:

$$\text{high} = \frac{\text{speed} - \left(\frac{\text{speed}}{100}\right)^2}{2}$$

Note: Since speed is incremented by 100 and multiplied by 100 in the manual entry mode, the result of (speed/100) is always an integer.

**Requirements of the program:**
Your program should implement all of the requirements from the second and third labs (Motor Control Interface and Timers, Etc.). In addition, you program should:

1. When in the “Display Mode”:
   - The LCD should display the current RPM of the motor (default should be 0)
   - The program should respond appropriately to the keys specified on p.2 (and display an error message for others)—IMPORTANT: note the **new** addition of the ‘0’ key.

2. For the approximation types, design the equation so that the duty cycle is 25% for a speed input of 5000 and 0% for 0 (note, your program doesn’t need to take 0 as an input—this is just for the equation you will use).

3. For the speed of the motor and RPM display:
   - IMPORTANT: The output duty cycle must be at least 1% to hold the switch low.
   - The speed should increase and decrease immediately after the increment or decrement keys are pressed.
   - The speed should immediately adjust upon entering of a new speed or approximation type.
   - The RPM should update immediately any time there is an update of the “Display Mode” (the displayed RPM should be within +/- 100 of the actual RPM).
4. The time display with the time that the program has been running should update any time the “Display Mode” is updated.

5. Demonstrate to the T. A. that you program is running. T. A. Initials:______________________

Suggestions

Your code will likely need the following:
- New case statement (e.g. check_0) in the main loop of the program for the “Display Mode” key press.
- A new subroutine to display the RPM (or a modification of the subroutine that displays the speed and approximation type).
- A new portion of code within the interrupt subroutines that counts the number of pulses accumulated over a known period of time.
- A subroutine that updates the duty cycle of the output wave according to the speed and approximation type.

The following steps for programming and debugging are suggested (but not necessary):
- Generate a PWM wave similar to Part 2 of the Timers, Etc. lab and verify that the 68HC11 is outputting it correctly with the oscilloscope.
- Create code that will modify the duty cycle of the PWM wave and again verify its functionality.
- Construct the motor control circuit from the prelab and use it to drive the motor with your output signal.
- Construct the pulse accumulator circuit from Figure 2 and verify that pulses are created using the oscilloscope.
- Create code that will count these pulses (over a known period of time) and store the value so it can be displayed when the “Display Mode” is updated (you might need a simple filter to clean up the input signal to the 68HC11 so that it only counts pulses from the light and not pulses from line noise).
- Verify the functionality of your RPM calculation by calculating the actual RPM from the oscilloscope.

Part 2—Documentation

When you are done (or as you code), do the Following to document your program (Note: comments, structure, and readability counts!!!)

1. Hand-in a complete Flowchart and for the entire program (control.RTF).

2. Hand in the listing of the complete, well-commented source-code that matches the flowchart. Remember, if the TA can not read your code with “reasonable” effort, then it is wrong.

Part 3—Questions

When you turn in the lab report, please answer the following questions. Provide flow charts and/or assembly language programs as necessary to explain your answers.

1. What did you find easiest about this lab? What was most challenging?

2. Take at least 5 points of data for varying speeds for each of approximation types and graph the user-entered speed versus the actual speed (from the oscilloscope). Which curve best approximates the speed curve of the motor? Name one way of more accurately controlling the speed of the motor so it reflects the speed entered by the user.

3. Take at least 10 points of data for the RPM calculated by your program and the RPM calculated from the oscilloscope. What could cause a difference between the actual and computed RPM? Draw a circuit that could aid in the accurate counting of pulses.
4. The circuit designed in the prelab was only one way of stepping up the current to drive the motor. Design an alternate circuit and name at least 2 advantages and 2 disadvantages of your new circuit as compared to the one designed in the prelab (hint, you can use online resources such as www.digikey.com to determine parts, prices, electrical characteristics, etc.)