ECE315 – Lab #4
Stepper Motor Control Using the L297 and L298 Integrated Circuits

Lab Dates, Report Dates and Demo Due Dates:

Please login to Moodle at eclass.srv.ualberta.ca to access all relevant lab dates. The lab schedule section contains the important dates for this semester.

Objectives:

- To gain experience interfacing a microcontroller to a stepper motor.
- To gain experience with the L297 and L298 integrated circuits (IC).

Equipment, Parts, and Provided Software:

We'll be using the keypad circuitry from lab #2.

<u>New Portion of Lab Kit for Lab #4</u>

- One Jameco 12 VDC, 480-mA, 2-phase bipolar stepper motor, 3.6 degrees per step in full-step mode and 1.8 degrees per step in half-step mode.
- L298 Dual Full-Bridge Driver
- L297 Stepper Motor Controller
- 3.3 nF Ceramic Capacitor
- Two 0.1 uF Ceramic Capacitor
- 100uF 50V Electrolytic Capacitor
- 22K Ohm 0.25 Watt Resistor
- 20-pin Cable from the buffer board attached to the Coldfire-based board.
- 4-pin header to attach to the motor
- Stepper Class Shell

Documentation:

Please login to the Moodle lab pages at eclass.srv.ualberta.ca to see a list of necessary documents for the ColdFire board used in ECE315. The documents are listed under the Lab Reference Materials heading.

The **MOD54415 PinIO Class Appnote** provides information on how to select the modes for each pin available on the J2 header.

The L297 Stepper Motor Contoller datasheet provides information on the motor controller and the control lines that control movement of the stepper motor via the L298.

The **L298 Dual Full-Bridge Driver** datasheet provides information on the device that drives the motor.

Introduction:

Microcontrollers are sometimes used to control rotating mechanisms, such as shafts and axles as well as attached pulleys, gears, and/or wheels. A stepper motor is a motor whose rotor position can be controlled by switching direct current (DC) voltages to two or more stator windings. Each winding can be energized in the forward direction, the reverse direction or not energized at all. Such a motor is convenient to control using a microcontroller since only digital input signals are required.

The stator windings in a stepper motor must be energized in the correct direction and the correct sequence to produce controlled rotation of the rotor. Each of the four steps in the full-step sequence advances the rotor of the Jameco motor by 3.6 degrees. In full-step mode, 100 steps would advance the motor one full rotation. If the motor were setup in half-step mode then each step would travel 1.8 degrees so 200 steps would be required to travel a full revolution. Please note that DC means don't care in the half-step sequence table.

Full-Step Sequence												
Steps	Winding Current A	Winding Current B	L 298 Outputs				L 298 Inputs					
			Out 1	Out 2	Out 3	Out 4	En A	En B	IN 1	In 2	In 3	In 4
			A	#A	B	#B			A	#A	B	#B
#1	Forward	Forward	Src	Sink	Src	Sink	1	1	1	0	1	0
#2	Forward	Reverse	Src	Sink	Sink	Src	1	1	1	0	0	1
#3	Reverse	Reverse	Sink	Src	Sink	Src	1	1	0	1	0	1
#4	Reverse	Forward	Sink	Src	Src	Sink	1	1	0	1	1	0
#1	Forward	Forward	Src	Sink	Src	Sink	1	1	1	0	1	0
#2	Forward	Reverse	Src	Sink	Sink	Src	1	1	1	0	0	1

Table 1: Full-Step Sequence for a Bipolar Stepper Motor

Half-Step Sequence												
Steps	Winding Current A	Winding Current B	L 298 Outputs				L 298 Inputs					
			Out 1	Out 2	Out 3	Out 4	En A	En B	In 1	In 2	In 3	In 4
			A	#A	B	#B			A	#A	B	#B
#1	Forward	Forward	Src	Sink	Src	Sink	1	1	1	0	1	0
#2	Forward	Off	Src	Sink	Off	Off	1	0	1	0	DC	DC
#3	Forward	Reverse	Src	Sink	Sink	Src	1	1	1	0	0	1
#4	Off	Reverse	Off	Off	Sink	Src	0	1	DC	DC	0	1
#5	Reverse	Reverse	Sink	Src	Sink	Src	1	1	0	1	0	1
#6	Reverse	Off	Sink	Src	Off	Off	1	0	0	1	DC	DC
#7	Reverse	Forward	Sink	Src	Src	Sink	1	1	0	1	1	0
#8	Off	Forward	Off	Off	Src	Sink	0	1	DC	DC	1	0
#1	Forward	Forward	Src	Sink	Src	Sink	1	1	1	0	1	0
#2	Forward	Off	Src	Sink	Off	Off	1	0	1	0	DC	DC

Table 2: Half-Step Sequence for a Bipolar Stepper Motor

The main difference between full-step mode and half-step mode is the different pin sequences for the enable lines. The enable signals are fixed in full-step mode but switch on and off in half-step mode.

In order to generate the sequence of values in the tables above, we will be using the L297 stepper motor controller. This simple device has four control signals that we will be using to generate the pin sequences in Table 1 and Table 2.

- CW/CCW (Pin 17 on the L297) controls the direction the shaft turns. Select clockwise or counter-clockwise.
- HALF/FULL (Pin 19 on the L297) controls the mode of the motor. Select half or full step mode.
- /CLOCK (Pin 18 on the L297) controls the stepping of the motor. The motor is moved on the rising edge of the input signal. Send one hundred pulses to the motor to move the motor one hundred steps.
- ENABLE (Pin 10 on the L297) controls the inputs to the L298 enables and coils. Set the pin low to disable the L298 and cut off current flow to the motor. Set the pin high to turn the motor on. Note that this does not move the shaft.

Please try not to leave the motor on for too long without moving it. This is known as stall mode. The L298 will get very hot if the motor is in stall mode.

The stepper motor that you will be controlling in this set of laboratory exercises will be operated in open loop mode. By this we mean that the microcontroller sends out digital signals to the stepper motor and assumes that the rotor will always rotate through precisely the correct number of steps without slippage. The microcontroller lacks the feedback means to verify the actual position of the rotor before and after each step command. Open loop operation is more risky in the sense that the microcontroller cannot verify that it does indeed have accurate control over the rotor's position. The rotor position may, in fact, deviate from the expected position if, for example, the limited torque of the motor is unable to rotate the loaded shaft fast enough to keep up with the stepping commands. A better alternative to open loop operation would be to operate in closed loop mode.

To provide closed loop operation, feedback would need to be provided through a shaft encoder sensor that would allow the microcontroller to independently verify the actual position of the rotor. But we will not be using feedback in this lab.

Prelab:

Download the schematic for this lab from the Moodle eClass site and wire up the circuit.

Take special care to ensure that you connect the proper voltage to the correct chip, as the 12-volt line can (and will) destroy every other chip on the board if connected incorrectly! Also, confusing the input and output lines of the L298 will cause 12V to be driven back into the microcontroller! The diodes are also critical to the health of the NetBurner board. The silver lines on the diodes correspond to the lines on the diodes in the schematics. Take care to make sure the orientation is correct.

Your kit does not include four passive components that are on the schematic. The following components are available in the lab in labeled bins: 22K resistors, 0.1uF ceramic capacitors, 100uF electrolytic capacitors, 3.3nF ceramic capacitors. Please help yourself.

Exercise 1:

Goal: To build a new project for lab 4.

Steps:

1. Connect the 20-pin cable from the ColdFire microprocessor to your breadboard.
2. Please remember that the odd-side pins are the only pins used on this header. The even-side pins are all no-connects.
3. Get the lab4.zip file from the eClass course website.
4. Create a new project called lab4 in your ECE 315 workspace. Follow the same procedure as you did in the tutorial.
5. Unzip and import all of the files into your project and delete the main.cpp file before building your project.
6. Run the project and check the heartbeat and context switching pins (J2[3]andJ2 [4]). Verify that the default webpage is being served.

Exercise 2:

Goal: Implement the Stepper driver class that controls the movement of the stepper motor.

Steps:

1. Complete the following methods in the Stepper class that controls the movement of the motor. The schematic for this lab contains the PinIO class pins that are used to control the motor.

    - Stepper::Init
    - Stepper::Step

- Stepper::SetState
- Stepper::GetState
- Stepper::SetMode
- Stepper::GetMode
- Stepper::SetDirection
- Stepper::GetDirection

2. The direction, state, and mode pins can be controlled simply by changing the levels of the associated PinIO class pins.
3. The clock pin requires edges to move the motor. Sending pulses to the motor will move the shaft. The rising edge of the signal will move the motor.
4. The while() loop in UserMain is setup to rotate the motor two rotations clockwise and two rotations counter-clockwise in each mode. Test that your motor moves as expected in both half-step and full-step modes.

Exercise 3:

Goal: In this exercise, the goal is to use key presses from your keypad to control the motor. Each key press will alternate between the following states: off, clockwise full-step, off, counter-clockwise half-step.

Steps:

1. Import your keypad class code from lab #2.
2. Your ISR should simply call the methods in the Stepper class to cycle through the 4 states listed above.
3. Modify the UserMain while loop to have very long motor movements. Consult the code package for the recommended values.

Report Requirements:

- Your typed report is to briefly describe your design solution for exercises one, two, and three.
- Be sure to provide neatly formatted and well-commented code for all exercises.
- Briefly describe your test cases for verifying that each of your solutions worked properly. Describe which case was tested, as well as the anticipated and actual results for each case. Place this data in a table. **Do not write out your test cases. Please tabulate them.** Non-tabulated test cases will not receive full credit on the reports.

Consult the report writing guidelines if you have any questions regarding the report format. The report writing guidelines are available on the Moodle page under Lab Reference Materials.

Marking Scheme:

Lab #4 is worth 20% of the final lab mark. Please view the lab 4 marking sheet to ensure that you have completed all of the requirements of the lab. The marking sheet also contains a limited test suite in the demo section. Please make use of it.

Report Submission:

Reports and code should be submitted via eClass by section. Please note that late reports and demos will not be accepted.