Project 2: Op Amp DC Motor Interface

Thomas Boynton

CET 346

5/18/2010

This project will supplement the project started earlier that used a transistor circuit to control a DC motor. This project will use an op amp, OPA 541AM to control our dc motor and incorporate a delay to control this motor.

The servo motor to be used with the computer interface control is a cylindrical, inch and three quarter diameter motor with a black and red twisted pair coming from the rear of the tube and what appears to be an encoder device on the rear of it. This encoder has a five wire connector on it. A bracket is then fabricated to hold the motor in place on the bench. The design that is chosen is just a simple aluminum L-channel with the appropriate mounting holes machined in it. This gives the user the option of whether to simply rest it on the bench for test or holes can be drilled in the bracket to fix it to the assembly it will be used in.

To begin to understand the problem at hand, we have to start by researching the motor that we have. It is an Ametek/Pittman part number GM8224S021. After a few hours of searching the Internet, it is determined that it is an 8224 Series 12V Brush DC Servo Motor with a spur gearbox on the front and a Pittman E30B Incremental Optical Encoder on the back of it. The pin-outs are obtained for the optical encoder and are listed below. The twisted black and red wires on the motor are for motor control.

E30B Optical Encoder Motor Wire Designations

E30B Connection Chart		
Pin	**Color**	**Function**
1	Black	Ground
2	Green	Index
3	Yellow	Channel A
4	Red	Vcc
5	Blue	Channel B

Table 1: Optical Encoder Pin Connections

Now that the pin/wire designations are known and the motor voltage is known, a circuit will be designed for it. Since the circuit coming from the computer is a low current signal circuit, the motor has to be controlled with something that has enough current to control the DC Servo motor. In this case, this will be our OPA 541 op amp.

Our OPA541 is a power op amp. It can have a power supply input from +- 40 volts and can deliver current up to 5 amps. It uses a single current limiting resistor to control the current in both directions. This T0-3 package has the outside isolated so a Mylar insulator is not necessary from the case to the

heat sink. The pin-outs are shown in figure 1 and the circuit that is internal to the op amp is shown in figure 2.

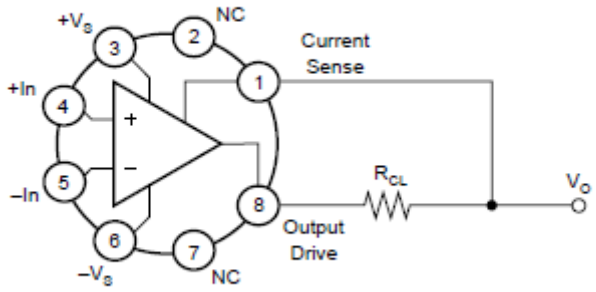

Figure 1: T0-3 package pin-outs

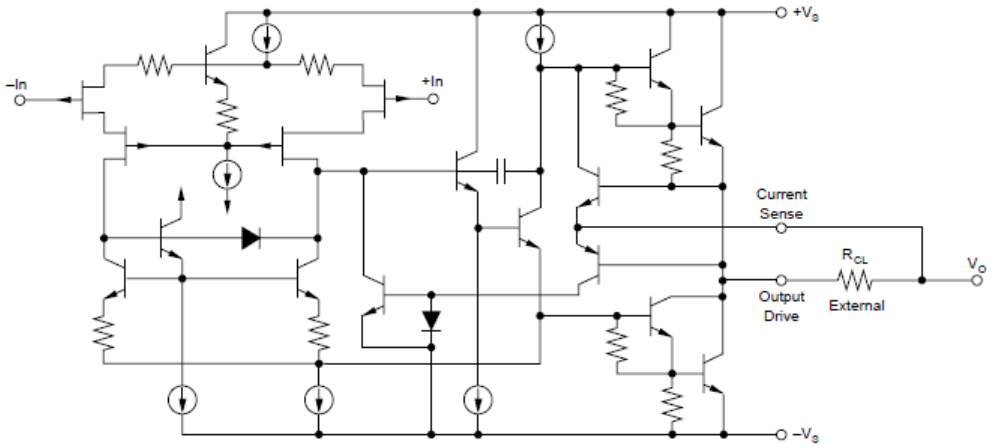

Figure 2: Internal circuitry to the OPA541 AM Operational Amplifier

As per the manufacturer's data sheet, if this device is to be used with any type of EMF generating load such as a motor that has an inductive spike it should be wired with some diodes for protection. Making a circuit in Mutlisim for this, it is shown in figure 3.

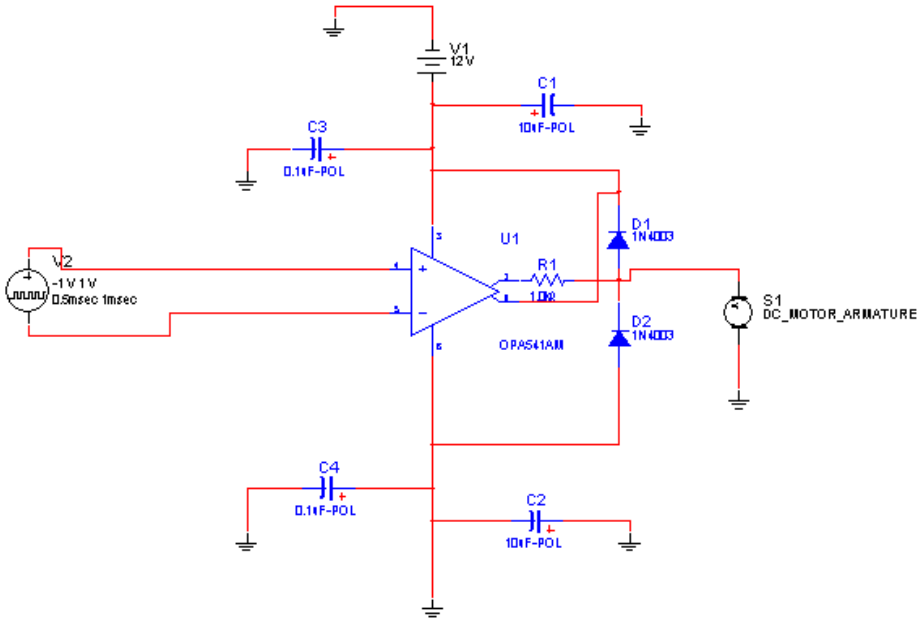

Figure 3: EMF Protection Circuit.

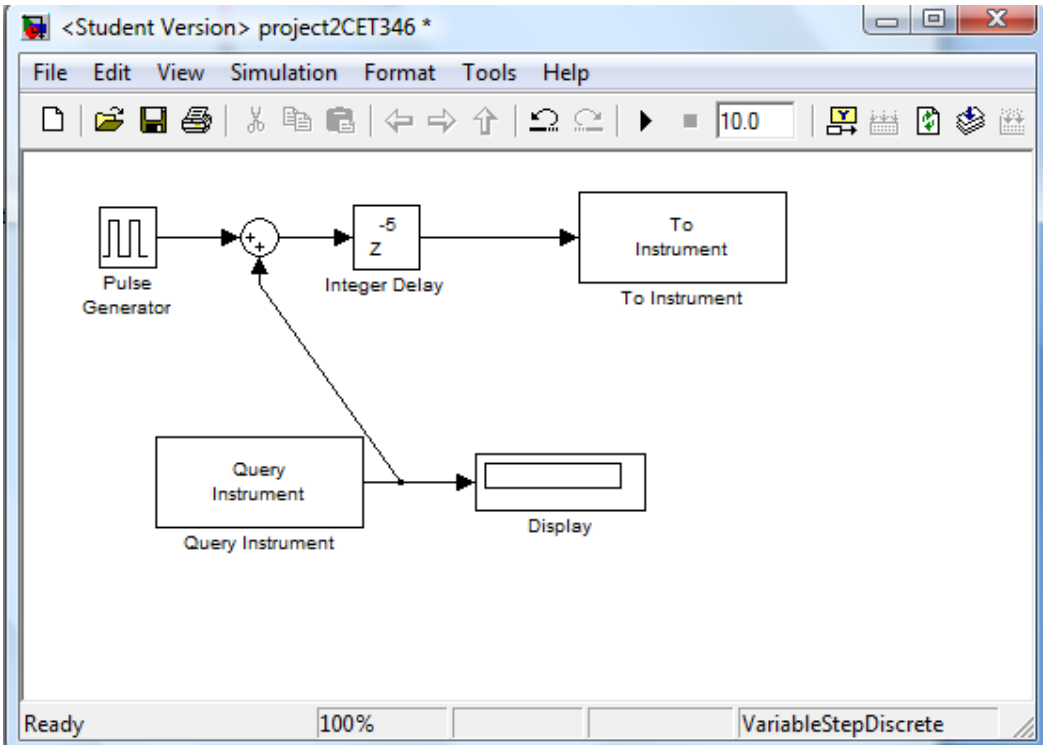

Figure 4: Simulink Motor Control Panel

This circuit now needs to be controlled in a suitable interface with a delay in it. I will try to use Matlab with the Simulink interface to try and accomplish this. To do this I open Simulink and start a new sheet. I can then use a “To Instrument” block to send my output of this model to the serial data port which will connect to my new circuit. To monitor this circuit and the encoder inside the DC motor, we can use the block ”Query Instrument” to process the signal coming in from the serial port to see what our motor is really spinning at. This block can then be attached to the display block to show our actual motor speed. In figure 3, above, the signal supply can be replaced by our serial cable attached to our computer which outputs the waveform. Our Simulink set-up is shown in figure 4.

To power the motor, we can use a pulse width modulated square wave to control the speed of the motor. We start with a pulse generator and then use the feedback coming back from the motor and the integer delay block to create our delay for the motor and to account for changes in motor speed. This particular setup has a delay setup of five cycles to change the motor speed and the sum block can be used to take the difference between the real speed and the commanded speed to adjust the motor speed as required.

We want to make sure we set our block priorities to make the “to instrument“ block the number one block and the query instrument, the number two block so the interface sends the signal to the motor and then reads the information from the motor afterwards. We can adjust our pulse width for the motor in the pulse generator block using amplitude and period to get our frequencies.