Lab 4 – Part 1
MC9S12 Subsystems: Pulse Width Modulation

In this sequence of three labs you will learn to use three of the MC9S12's hardware subsystems.

Introduction and Objectives

The speed of a motor can be adjusted by powering it with a pulse width modulated signal. Figure 1 shows how this can be done. The SN754410 motor controller works as a switch. When the signal on the enable (EN1) input of the L293D is high the switch is closed, current flows through the motor, and the motor speeds up. When the signal is low, the switch is open, no current flows, and the motor slows down. IN1 and IN2 determine the direction of the motor. When IN1 is high and IN2 is low, OUT1 will be high (VM) and OUT2 will ground. Changing the logic levels of IN1 and IN2 reverse the voltage levels on OUT1 and OUT2, and the motor changes direction. With a high enough frequency PWM signal the amount the motor speeds up and slows down in one period is negligible, and the motor turns at a constant speed. By adjusting the duty cycle the speed of the motor can be controlled. For the motors you will be using in this lab, use a PWM frequency of about 5 kHz.

To make a motor turn at a desired speed it is necessary to know how fast the motor is turning. The motors you will use in this lab have encoders which will generate 15 pulses in a single revolution of the motor. A light emitting diode (LED) sends light through a wheel with holes in it which is attached to the shaft of the motor, and an optical sensor detects light. When the light shines through a hole and hits the sensor, the output of the optical sensor is VDD (a digital 1); when the wheel blocks the light from reaching the sensor, the voltage goes to 0 V (a digital 0). Using this optical encoder you will be able to measure the speed of the motor. The motors we will use have two optical encoders, on opposite sides of the wheel. This allows you to determine both the speed and the direction of the motor, using a technique called quadrature detection. This will be discussed more in the final lab for the course. For this lab, use just one of the optical encoders. In this lab you will use the PWM subsystem of the MC9S12 to adjust the speed of a small motor, and use the input capture subsystem to measure the speed of the motor.

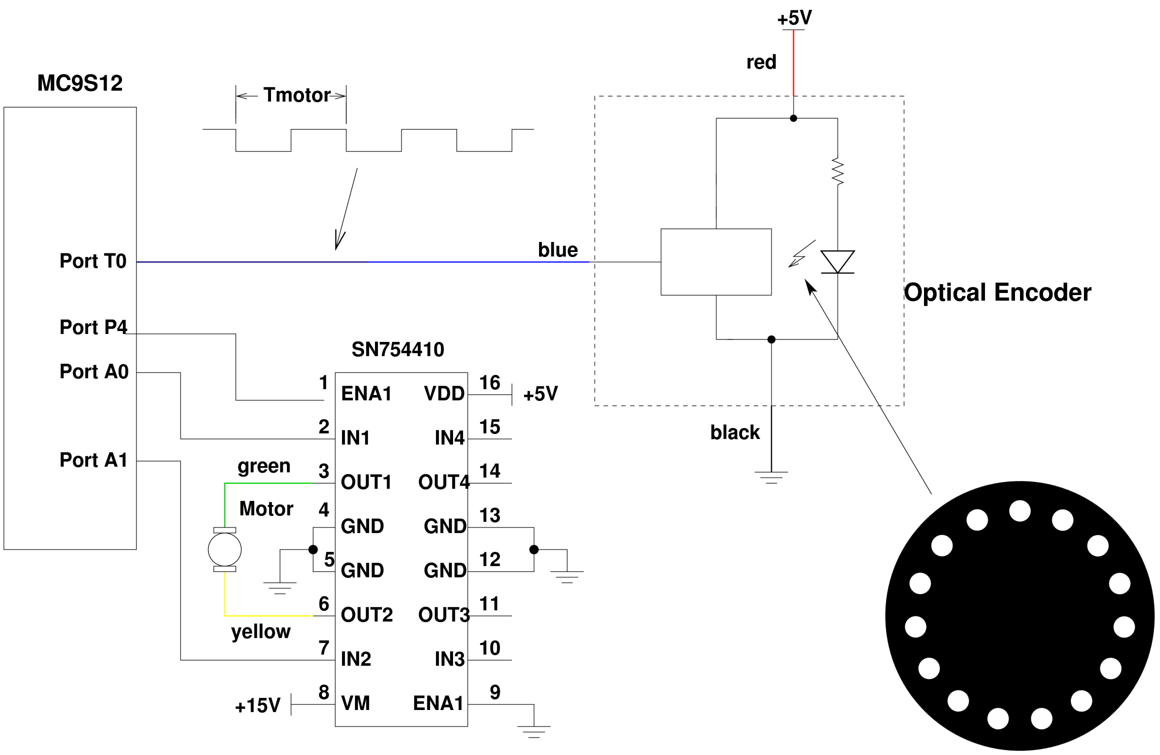

Figure 1: Using a PWM signal to adjust the speed of a motor.

1. Prelab

For the pre-lab do the calculations required for Parts 3 and 4. Write the program required for Parts 5.

2. The Lab

1. Connect the motor directly between 0 and 15 V to verify that the motor turns (Do not use the SN754410 for this part.) Connect the Red lead from the optical encoder to VDD, and the Black lead to ground. Have a TA or Instructor check your wiring before turning the power on. The motors and encoders are fairly expensive, and we do not want to burn up any of them. Use a logic probe to verify that the Blue lead from the optical encoder is pulsing, indicating that the encoder is working.

2. Program your MC9S12 to generate a PWM frequency of 5 kHz, with an active high PWM signal with a duty cycle of 50%. (Use Bit 4 of the PWM, since Bits 0 through 3 of Port P are used to enable the seven-segment LEDs.) Make Bit 0 of Port A high, and Bit 1

of PortA low. Connect the circuit shown in Figure 1. Use your logic probe to verify that the optical encoder is working.

3. Connect the output of the optical sensor to one of the input capture pins of the MC9S12. To your program from Part 2, add code to use the input capture function to measure the period of the signal from the optical encoder, and to display the values on the seven-segment LEDs. The motor speed at 100% duty cycle will be about 3000 RPM. Set the prescaler such that you can measure speeds as low as 50 RPM (to avoid overflow of the TCNT register in case the motor slows down below the 50 RPM lower limit).

(a) With a 3000 RPM motor speed, how much time will there be between successive rising edges from the optical sensor?
(b) With a 50 RPM motor speed, how much time will there be between successive rising edges from the optical sensor?
(c) Set the prescaler of the MC9S12 so the rollover time of the TCNT register is the smallest time which is larger than the above two numbers. What value did you use for the prescaler?

4. Modify your program to set the duty cycle based on the state of four of your DIP switches. The duty cycle of the signal should be determined by reading the four DIP switch values into the four least significant bits of Port H:

Port H	Duty Cycle	Port H	Duty Cycle
0000	06.25%	1000	56.25%
0001	12.50%	1001	62.50%
0010	18.75%	1010	68.75%
0011	25.00%	1011	75.00%
0100	31.26%	1100	81.25%
0101	37.50%	1101	87.50%
0110	43.75%	1110	93.75%
0111	50.00%	1111	100.0%

Then use Bit 4 of the DIP switches to indicate which to display – if Bit 4 is on, display the number of timer cycles between input capture edges. If Bit 4 is off, display the speed of the motor in RPM.

5. Run the motor with each of the duty cycles from Part 4. Have the MS9S12 display the number of timer pulses between rising edges on the seven-segment LEDs. Record these numbers, and convert them to RPM. Record the speed in RPM for each duty cycle. Plot the motor speed as a function of duty cycle. How linear is it?

6. Compare your results (speed of the motor vs. duty cycle) to the results of at least two other groups. Do the motors all behave the same, or are there significant differences?