Embedded Hardware Design

<u>Experiment 4/5 : Speed Control of a small DC Motor</u> 15 Sept. 2010

Small d-c motors like those used in cassette players and disc drives, use permanent magnet stators, and the speed of the motor depends only on the voltage applied to the rotor. The most convenient and energy-efficient way of controlling the speed of such motors is to apply a Pulse Width Modulated (PWM) voltage to the motor and control the duty cycle of the PWM waveform.
The ATMega16 Microcontroller provides several PWM outputs with programmable duty cycles. In this experiment, the motor will be driven by OC1B (PD4 pin) output (vP) through a transistor (TIP 31C), and the motor speed will be measured by means of a circular disc with 20 holes along its periphery mounted on the motor shaft, passing through the slot of an optical switch, as shown in the circuit diagram.
As the motor runs, the infra-red beam of the optical switch gets alternately passed and blocked, resulting in a pulse stream having 20 pulses per revolution of the motor. This pulse output (vS) will be applied as the interrupt input INT1 of the microcontroller (PD3 pin) and the speed will be measured by counting the number of interrupts coming in a pre-determined time interval, set by Timer/Counter2. This time interval can be checked by configuring one of the port pins (say PD0) as output and keeping it HIGH as long as Timer/Counter2 is not disabled.

I. Measurement of Speed

1) Set up the circuit shown in the diagram. Apply 5V d-c to the motor input and observe the pulse output vS on a CRO. Measure the time period of the waveform and hence calculate the speed of the motor in revolutions per minute (RPM). This gives the maximum speed the motor will be able to attain during this experiment, as the PWM output applied to the motor will be a square wave having maximum value 5V.
2) Write an interrupt routine to increment a variable (global & volatile) "count" every time the ISR is executed.
3) Write a main program to perform the following sequence of jobs:
   a) Program Port C as usual for LCD operation. Program Port D pins as follows: PD0 as output, PD3 as input and PD4 as output.
   b) Set Timer/Counter2 for normal mode of operation, with the clock pre-scaled by a factor 1024, so that “count” reaches the value 250 for the maximum speed in one window. and create a window of suitable width (monostable pulse) using Timer/Counter2,
   c) Enable external interrupt for falling edge by setting the values in GICR,MCUCR & SREG registers.
   d) Disable interrupts by using cli(). Disable Timer/Counter2 by setting the relevant bits in TCCR2 to no clk source.
   e) Print the value of “count” on the LCD.
   f) Go back to step b after a delay of 300 msec.
4) The value displayed on the LCD gives the count corresponding to the maximum speed of the motor, and one can hence find the value of count corresponding to any required speed.

II. **Speed Control**

1) Modify the main program by adding the following in step 3a:
   a) Generate a PWM of frequency 1 KHz and 5% duty cycle using Phase Correct PWM mode of Timer/counter 1.
   b) Take some value of the required speed (within the possible range) and calculate the corresponding value of “count”. Specify this to be the value of a variable “set_count”.
2) Insert the following steps after step 3e:
   a) Calculate “diff” = “count” – “set_count”. Feed this error back to correct the duty cycle by increasing/decreasing the value of OCR1B by a suitable step size (1 – 5) depending on whether “diff”< or ≥ tolerance (in the range 5 – 10).
   b) Display the value OCR1B and “set_count” on the LCD along with “count” value displayed so far.
3) Display the PWM output vP and the interrupt signal vS on the CRO, and study the effects of varying the values of “set_count”, step size and tolerance on the speed control action.