Processor Hardware/Software Interface
EECS 113

Assignment 3 - Motor Revolution Counter

University of California, Irvine

Assigned: April, 29, 2021

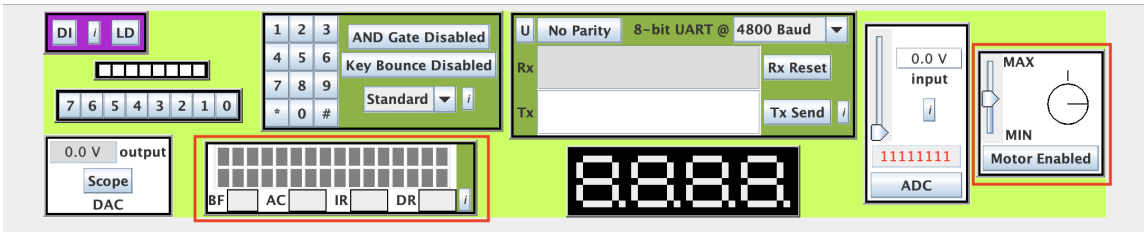
Due: May, 10, 2021 by 11:55PM

1 8051 Micro-controller Programming using Edsim51

In this assignment, you are going to write a program and simulate a motor revolution counter using a 8051 micro-controller, DC motor, and LCD in Edsim51.

Here in Figure 1, DC motor panel and LCD (8051 peripherals in Edsim51) are highlighted in red boxes. The motor can be disabled/enabled by clicking on the motor Enabled/Disable button.

Figure 1: Edsim51



1.1 The Motor

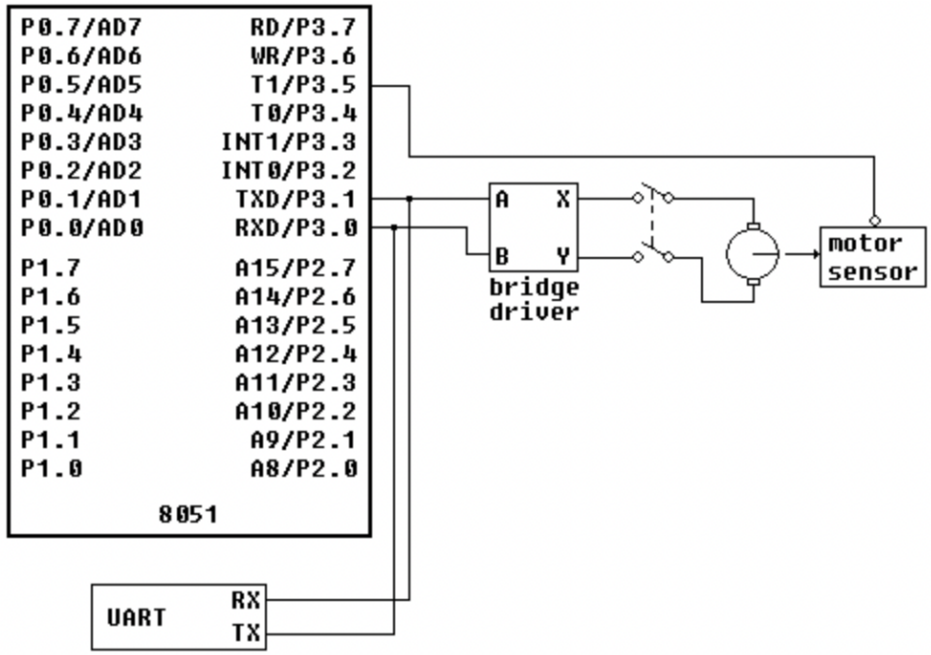
A DC motor is a rotatory instrument that works on DC power to convert electrical energy into mechanical energy. DC motors can vary in size and power from small motors in toys and appliances to large mechanisms that power vehicles, pull elevators and hoists, and drive steel rolling mills, etc.

Figure 2 shows how the DC motor may be interfaced with the 8051 in the Edsim51 simulator. P3.0 and P3.1 are applied to a dual bridge driver, the outputs of which are applied to a bi-directional DC motor. The truth table for the bridge and its effect on the motor is shown in Table 1.

Table 1: Truth table for the bridge driver

A	B	motor
0	0	stop
0	1	forward
1	0	reverse
1	1	stop

Figure 2: Schematic of interfacing Motor with 8051



Note: The motor control lines share the TXD and RXD lines for the 8051's internal serial port. As can be seen in the Figure 2, these lines are also connected to the external UART. Therefore, when exercising the motor, garbage messages may appear in the UART's receiver window.

The motor sensor, which is applied to P3.5, goes low once every revolution (in the simulator, whenever the motor shaft lines up with the sensor (positioning at 12 o'clock), the sensor changes from black to red and P3.5 goes to logic 0). P3.5 is the external clock source for timer 1. Therefore, code can be written that, using timer 1, counts the motor's revolutions. The speed of the motor can be varied manually from MIN to MAX (using the slider to the right of the motor - take a look at the hardware screenshots above). This will make the revolution counting programs more interesting.

1.2 Motor Revolution Counter

The motor should be setup to rotate in a clockwise (forward) direction and the number of revolutions per second should be displayed on LCD. It needs to be observed if the number of revolutions per second increases or decreases (LCD gets updated) as the motor slider (see Figure 1) is moved up (toward MAX) or down (toward MIN).

The motor sensor is connected to P3.5 (see Figure 2), which is the external clock source for timer 1. Therefore, timer 1 is put into event counting mode and counts how many revolutions have taken place.

To get the number of revolutions per second (RPS), one must inspect the revolution count in timer 1 every second, convert it to decimal, and display it (in decimal) on the LCD panel and then reset the count. This process should be repeated endlessly. Due to the limited bit precision (8 bit registers), we will approximate RPS by inspecting timer 1's count every 10ms (instead of doing so every second), and multiplying that count by 100 (100=1second/10ms). One way to do this is to have interrupts occur every 10ms. Another way is to have the microcontroller continually checking if 10ms have elapsed. In either case, when 10ms have elapsed, the microcontroller should initiate a "Hex to Decimal" conversion of the revolution count. When the conversion is complete, the LCD is updated with new value. At this time, the previous revolution count should be cleared and counting should be started again.

Note: The result of conversion (from Hex to Decimal) shows the number of motor revolutions in one

second, which can be approximated by multiplying the 10ms revolution count by 100.

1.3 LCD

LCD should be connected to port P1 of 8051, and Set Interface data length to 8 bits, 2 line, 5x7 character font, also set the entry mode to increment with no shift. The display should be updated once the slider has moved up or down. **Make sure to use P3.2 for LCD E and P3.3 for LCD RS in your program and Edsim51 pin configuration.**

1.4 Testing Your Program

Run your program with the motor at different speeds (use the slider to the right of the motor to increase the motor speed). The LCD should display the number of motor revolutions per second in 5 decimal digits. You can run your program with updating frequency from 1 to 100. Submit 3 screenshots of your program changing the motor slider in different speeds.

2 Assignment Deliverable

- Your code should be commented.
- Submit your assembly file along with 3 screenshots (mentioned in Section 1.4) through **Canvas** before deadline in a **.zip** file.