

For any required I2C functionality, use subroutine calls *i2c_start()*, *i2c_rstart()*, *i2c_stop*, *i2c_put(char byte)*, *char i2c_get(char ackbit)*. If you use *i2c_put*, you must pass in as an argument the byte that is to be written to the I2C bus. If you use *i2c_get*, you must pass in an as argument the bit value to be sent back as the acknowledge bit value.

ECE 3724 Quiz #10 Reese NAME: _____

Answer each of the following questions (you can use a calculator)

- a. (4) Assume you have an analog input voltage to the PIC A/D that represents a motor shaft speed that varies linearly from 0 to 5 V over the range from 0 to 2400 RPMs. What voltage represents a speed of 300 RPM? Assuming the PIC A/D is used for conversion with $V_{REF} = 5\text{ V}$ and only the upper 8-bits is retained, what is the value (give this in hex or decimal) for this speed of 300 RPM?

$$300/2400 * 5\text{ V} = 0.625\text{ V} \quad \text{RPM to Volts}$$
$$0.625\text{ V} / 5\text{ V} * 256 = 32 \quad \text{Volts to ADC code}$$

- b. (3) For the MAX 517 DAC, assume that both A1, A0 are tied low and write a sequence of I2C function calls that causes a voltage of 3.5 V to appear on the output.

$$3.5\text{ V} / 5\text{ V} * 256 = 179 \quad \text{Volts to DAC code}$$

```
i2c_start();
i2c_put(0x58); //DAC i2c address
i2c_put(0x00); //DAC conversion command
i2c_put(179); // DAC conversion data
i2c_stop();
```

- c. (3) Assume that the PIC A/D has already been configured and that V_{REF} is 5 V. Write a C function that does continual A/D conversions and does not exit until the input voltage falls below 1 V. Assume left justification and that you only need to consider the upper 8 bits. Before doing the conversions, select input channel AN2.

$$1.0\text{ V} / 5\text{ V} * 256 = 51 \quad \text{Volts to DAC code}$$

```
CHS2 = 0; CHS1 = 1; CHS0 = 0;
DelayUs(20); //wait for acquisition
do {
    GODONE = 1; //start conversion
    while(GODONE); // wait for conversion to end
}
while (ADRESH > 51); //loop while input voltage > 1V
```