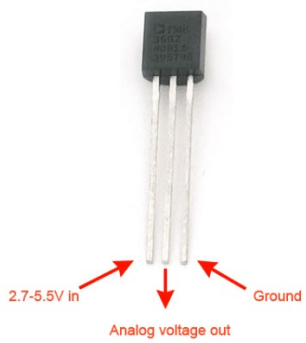
EE4144 Midterm 1 (Spring 2025) Name: _____

NYU Tandon School of Engineering ID: _____

1. Suppose we are using the Atmel32U4 microcontroller connected to a 1 MHz clock (unlike the one we have which uses a 8 MHz clock). Our goal is to provide an indication of temperature by using a blinking LED (using a PWM output). You can use any pins you choose and you may use the defined register macros like DDRD, PORTD etc. We are using the TMP36 temperature sensor shown below:



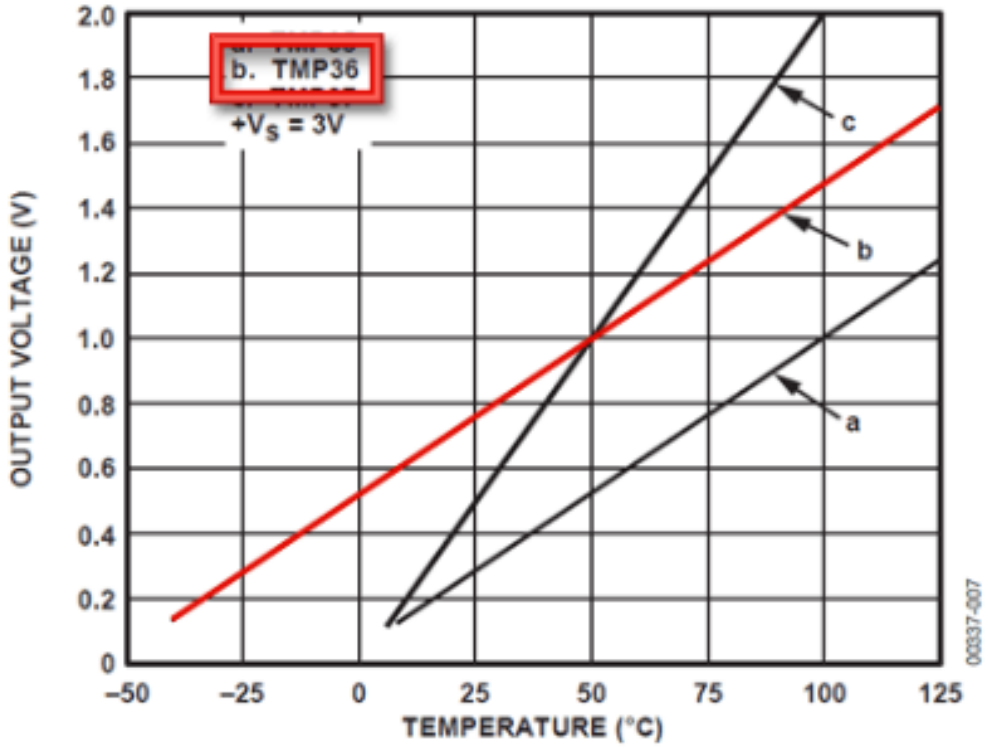


Figure 6. Output Voltage vs. Temperature

Temperature is calculated as follows (as per the data sheet):

Centigrade temperature = [(analog voltage in mV) - 500] / 10

We would like to achieve the following characteristics:

- If the temperature is 0°C or less, an LED blinks at a rate of 2 Hz. That is 0.25s "On", 0.25s "Off" (50% duty cycle)
- If the temperature is 50°C or greater, the LED appears to stay "On" permanently.
- The blinking rate changes proportionally to the temperature. For example 25°C would blink the LED "On" for 0.125s, "Off" for 0.125s.
- The duty cycle is fixed at 50% for all temperatures. Just the blinking rate is changing.
- You will not be able to actually build this since you don't have the sensor. The requirement is just to submit the diagrams and code necessary (see below)

a. Sketch a schematic of the hardware setup, showing the connections between the pins, TMP36 etc. Don't forget the LED needs a current limiting resistor in series. You can use a 330Ω resistor.
b. Write the C code to set up the analog input registers for the TMP36. (ADMUX, SDCSRA, ADCSRB, DIDR0)
c. Write the C code to set up the PWM(timer) configuration registers (TCCRxA, TCCRxB) to satisfy the timing requirements. NOTE: You can use fast or slow PWM.
d. Lastly, write the C code that updates the output compare registers (OCRxA, OCRxB) based on the ADC data (ADCH, ADCL) from the analog input, to satisfy the blinking rate requirement.

2. Often times we use PWM to simulate dimming of an LED. We know that LEDs can only be turned on or off, so adjusting the duty cycle of a PWM signal connected to an LED can look like a dimmed LED. Of course a 0% duty cycle PWM would make the LED look off, while a 100% duty cycle PWM signal would make the LED look bright (the brightness being mapped linearly to the PWM duty cycle percentage). Design a setup that uses the built-in temperature sensor on your board to dim an LED. For example, the warmer the temperature, the brighter the LED. You can use any hardware and any pins on your controller that you choose.
    a. Sketch the hardware schematic for your design. Include the specific pins on your microcontroller used for any connections.
    b. Write the setup routine in C that sets up the GPIO, ADC to read the sensor, and PWM/Timer to output the PWM signal.
    c. Write the loop code that reads the ADC and adjusts the PWM duty cycle appropriately to dim the LED.