REPUBLIC OF THE PHILIPPINES

SURIGAO DEL NORTE STATE UNIVERSITY

Narcisso Street, Surigao City 8400, Philippines

Caspe,Mark Angelo J.

BSECE-3A

Laboratory Activity no.8

Twisting:Potentiometers

**Introduction:**

Along with the digital pins the Arduino has it also has 6 pins which can be used for analog input.These inputs take a voltage (from 0 to 5 volts) and convert it to a digital number between 0 (0 volts) and 1024 (5 volts) (10 bitsof resolution). A very useful device that exploits these inputs is a potentiometer (also called a variable resistor). When it is connected with 5volts across its outer pins the middle pin will read some value between 0 and 5 volts dependent on the angle to which it is turned (ie. 2.5 volts in the middle). We can then use the returned values as a variable in our program.

**Objectives:**

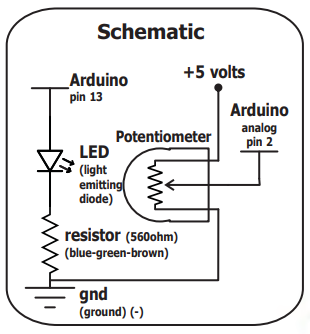
The objective of this activity is to understand how to interface analog sensors, like a potentiometer, with an Arduino board. By using the analog input pins on the Arduino, you can read voltage values from analog sensors and convert them into digital values that can be used in your program. In simulation with Proteus, you can test and validate your code without the need for physical hardware, allowing you to troubleshoot and refine your project before implementing it in the real world.

**Materials:**

1.) Atmega328P 2.) Crystal 3.) Potentiometer 10k ohm x1 4.) 22pf Capacitor

5.) Green LED x1 6.) 150 Ohm Resistor

**Schematic Diagram:**



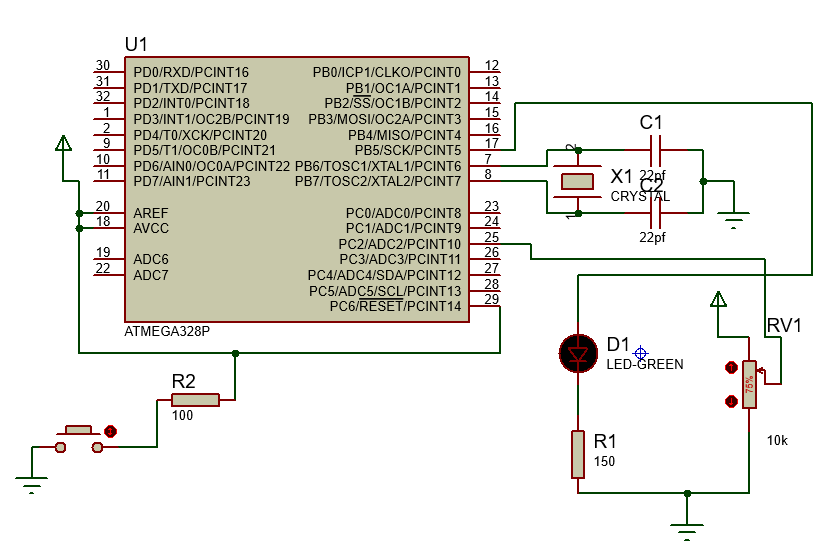
**Procedure:**

* To begin, open Proteus and create a new project. Then, find the components needed for your activity in the Proteus component library. This includes the Arduino board (such as Arduino Uno), a potentiometer, and any other components you plan to use.
* Next, refer to the schematic diagram or datasheets for guidance on connecting the components. Make sure to connect the potentiometer to one of the analog input pins on the Arduino.
* Open the Arduino IDE and create the code for your project. Once you've finished writing the code, save it.
* Now, go back to Proteus and add the ATmega328P microcontroller to your project. Then, click on the microcontroller and find the file you saved in step 5 (the .hex file) and choose it. This will load your Arduino code onto the simulated microcontroller.
* Finally, simulate or run the simulation in Proteus to see your project in action. You should be able to observe how the Arduino reads the analog input from the potentiometer and responds accordingly based on your code.

**Results and Discussion:**

During the simulation, adjusting the potentiometer will control the speed of the LED blinking: turning it one way speeds up the blinking, while turning it the other way slows it down. This illustrates how analog input from the potentiometer influences digital output, showcasing a practical application commonly seen in Arduino projects.

**Circuit:**



**Program:**

int sensorPin = A2;

int ledPin = 13;

int sensorValue = 0;

void setup() {

pinMode(ledPin, OUTPUT);

}

void loop() {

sensorValue = analogRead(sensorPin);

digitalWrite(ledPin, HIGH);

delay(sensorValue);

digitalWrite(ledPin, LOW);

delay(sensorValue);

}

**Findings:**

The simulation demonstrates the direct relationship between the position of the potentiometer and the speed of the LED blinking. By adjusting the potentiometer, users can effectively control the speed of the LED blink, showcasing the real-time interaction between analog input and digital output in Arduino projects.

**Recommendations:**

* Try out other sensors to see how they interact with the Arduino.
* Add more features like extra LEDs or a display for more functionality.
* Write clear comments in your code for easier understanding and troubleshooting.
* Share your project with others for feedback and ideas.
* Keep learning and experimenting to improve your Arduino skills.

**Conclusions**

The simulation demonstrates the ability to control the LED blinking speed by adjusting the potentiometer, showcasing the integration of analog input and digital output in Arduino projects. This simple interaction highlights the versatility and practicality of Arduino microcontrollers in various applications. By understanding how to manipulate analog signals, users can create responsive and dynamic projects that respond to real-world inputs. This project serves as a foundation for exploring more complex interactions and functionalities in future Arduino endeavors