Arduino Investigation Report

Joystick and RGB LED

This report covers the experimentation with the Joystick and the RGB LED. The objective is to learn how to use such components. As an extra and fun project I used the joystick to control the LED colors and mix them in an interesting way.

1. Hardware

Besides the well-known Arduino Duemilanove and a small breadboard, the following components were used in this project:

Joystick:

The joystick has 5 pins. They are:

- GND
- VCC
- X axis
- Y axis
- SW (Z axis button)

The pins for X and Y axes return a range of analog values from 0 to 1023, indicating the position of the joystick on the given axis. X returns 0 in the left most position and goes al the way to 1023 in the right most position. Y returns 0 in the bottom position and goes all the way to 1023 in the top position. The variation happens close to the center of the axes, and unfortunately reaches the maximum or minimum values not far from the center. The SW pin returns a digital value indicating one of two possible states for the Z axis. That is, it works like a button.

RGB LED:

The RGB LED has 4 pins. They are:

- GND
- Red color
- Green color
- Blue color

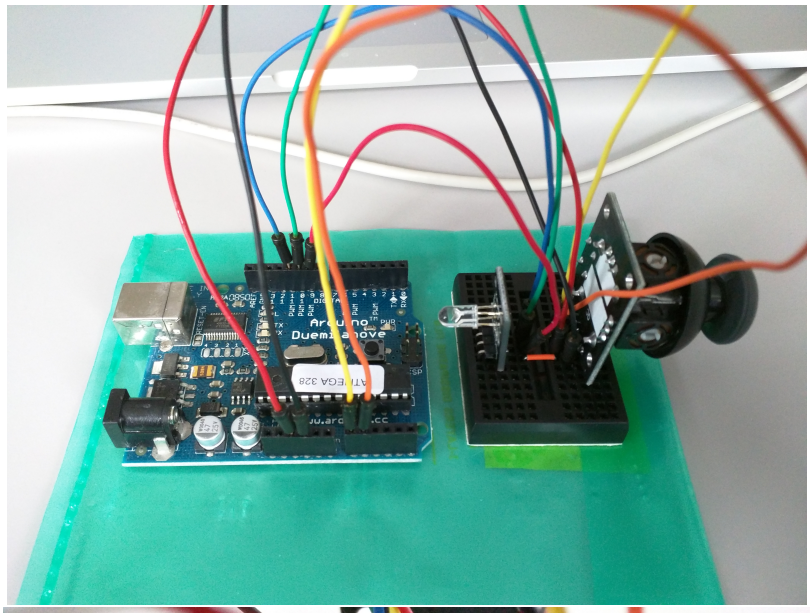
To make the LED glow in one of the three colors you need to send an analog signal to the respective pin. The accepted value ranges from 0 to 255. The colors may be mixed to produce different colors. You can also use different intensities for each color, sending values within the accepted range, and obtain several different result colors. Our unit of this RGB LED appears to have the red and blue colors swapped.

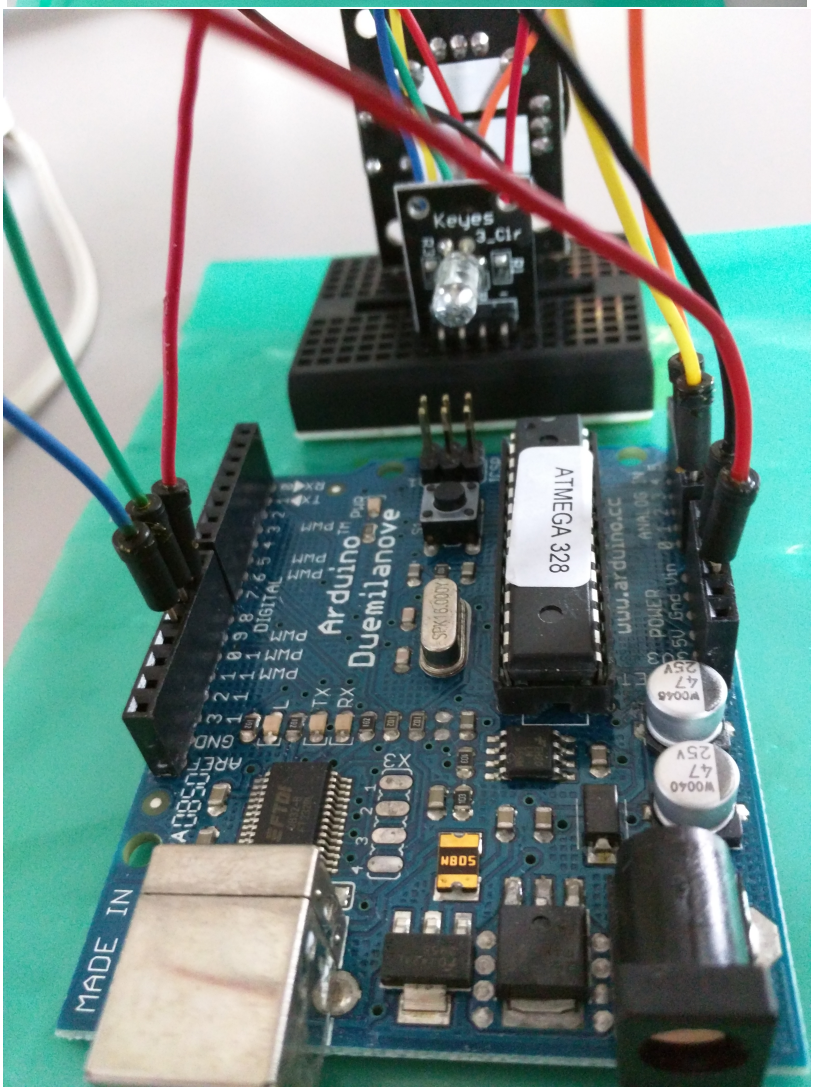
The circuit:

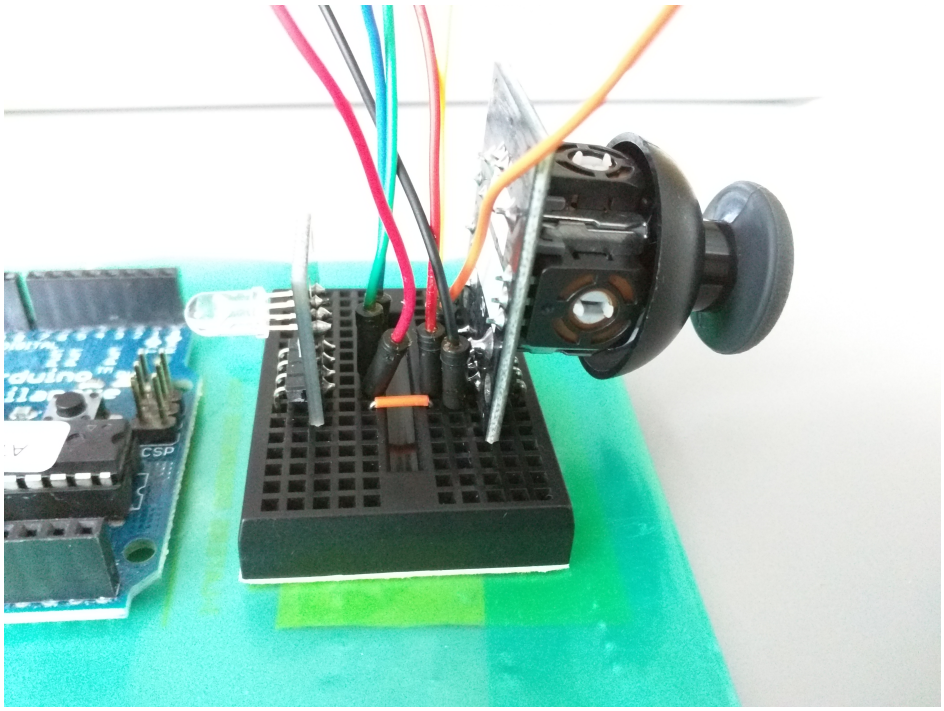
The joystick was connected to VCC (5V) and GND. The X and Y axes pins were connected to Arduino's analog pins A0 and A1, respectively. The SW button wasn't used in this experiment.

The RGB LED was connected to GND, and the colors were connected to Arduino's pins 9, 10 and 11.

The following pictures show how the circuit was assembled.




Keyes
3_Clr
ATMEGA 328
Arduino
Duemilanove
MADE IN



2. Software

To make the Joystick control the LED colors I assumed the following:

- The Blue color varies with the Y axis
- The Green color varies with the X axis
- The Red color varies as the opposite of the X axis

A simple conversion had to be done to adapt the range of input values to the range of output values. This conversion was just dividing the input value by 4.

Perhaps the most interesting thing about this project was the interface between the two components. Besides the conversion, I had to implement a simple debouncer to get more stable reads from the joystick.
As the first implementation, the LED light would glow and blink in strange patterns. It was not really stable and the color variation wasn't smooth. To correct that, I implemented a loop where I read the joystick input values several times (10, but you can change it. Just beware of overflowing the variables) and calculate the average. The calculated value is then used to control the respective color. This made the color variation smoother and more stable. The reason for this behavior is that the Joystick sensor may feel intermediate values while you're moving it, or even values that are not precise. Averaging several input values decreases the error impact.

3. Key Things Learned

- Make sure you are working with the right numbers. At first I didn't realize the joystick and the LED work with different ranges. I was using the same, and the result was way more unstable.
- When possible, try to compensate the lack of precision of your equipment with software. You can add some intelligence to the way the sensor work. The debouncer on this experiment is an example.

4. Problems / Questions / Future Work

The way I defined how the colors would vary cannot access all possible colors an RGB LED can produce. You cannot produce white, for example, because you would need the three colors with maximum values. Since the Red and Green colors vary opposite from each other, there is not way to have bot at maximum glow.
Let me know if you can think of a way to improve the range of colors we can reproduce with this experiment.

Version 1.1 – Changed the debouncer reading from 100 values to 10 values. It was overflowing the variables.
Tiago de Almeida, February 16, 2015

Version 1.0 (original)
Tiago de Almeida, February 7, 2015

Code

```
// Defining pins
#define BLUE 9
#define GREEN 10
#define RED 11
#define X A0
#define Y A1

// Defining debounce length
#define DEBOUNCE 10

// Variables
int Rint, Gint, Bint;
int Xvalue, Yvalue;
int times;

// Setting up serial output
void setup() {
  Serial.begin(9600);
  Serial.println("Starting...");
}

void loop() {

  // Initializing values
  Rint = 0;
  Bint = 0;
  Gint = 0;
  times = DEBOUNCE;

// Read input DEBOUNCE times
  while(times != 0){
    Rint += (1023 - analogRead(X));
    Gint += analogRead(X);
    Bint += analogRead(Y);
    times--;
  }

// Calculate average of input values
  Rint = Rint / DEBOUNCE;
  Gint = Gint / DEBOUNCE;
  Bint = Bint / DEBOUNCE;

// Convert value to range 0..255
  Rint = Rint / 4;
  Gint = Gint / 4;
  Bint = Bint / 4;
```

```
// Printing out calculated values
  Serial.print("R: ");
  Serial.print(Rint);
  Serial.print("G: ");
  Serial.print(Gint);
  Serial.print("B: ");
  Serial.println(Bint);

// Writing calculated values
  analogWrite(RED, Rint);
  analogWrite(GREEN, Gint);
  analogWrite(BLUE, Bint);
}
```