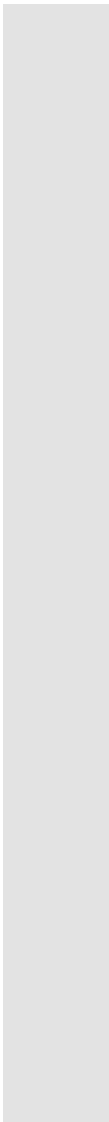




CS 5780 Mini Project **RGB LED Controller**



Kylee Fluckiger
JoCee Porter

Disclaimer

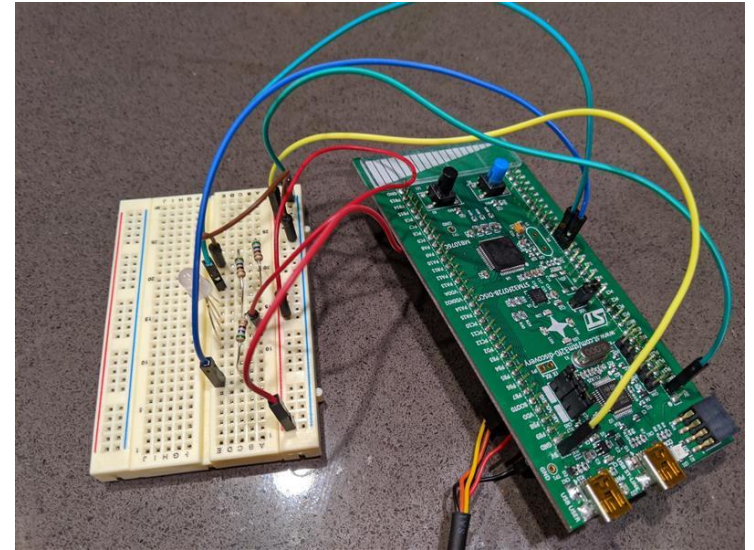
Our original LED Message Board project was aborted after we broke the LED matrix.

We discussed this with the professor and agreed upon this new project last-minute.

System Overview

The LED RGB Controller takes user input from the terminal to adjust the color of an RGB LED connected to the Discovery board.

As a bonus, we also wrote a Python script to enable direct serial communication with the system via code.



Laboratory Components

DAC

- We utilize both channels of the digital to analog converter on the Discovery board to control two color channels
- User input for RGB values are converted from an 8-bit digital signal to the analog voltage on each pin

UART

- Informative messages are sent to the user
- User sends digital RGB value back
- Additionally, a Python script allows for programmatic control of the serial port

Interrupts

- The UART receive interrupt is utilized to allow for efficient processing of user messages

"Milestones"

1. RGB LED circuit + DAC

- Designed the correct control circuit for the common anode RGB LED
- Set up and programmed the DAC for manual control of color values
 - Both DAC channels enabled for red and green control

2. UART communication

- Established bidirectional serial communication
- PuTTY used to receive informative messages and send color channel selection and 8-bit (0-254) digital color values
- Input parsing and error handling implemented in communication protocol

3. Python script for serial communication

- As a bonus, we utilized the PySerial library in a custom Python script
- Replaces PuTTY as the user terminal to communicate with the board via the serial port
- No COM port setup required by the user, simply execute
- Future extendibility now easy via programmatic control

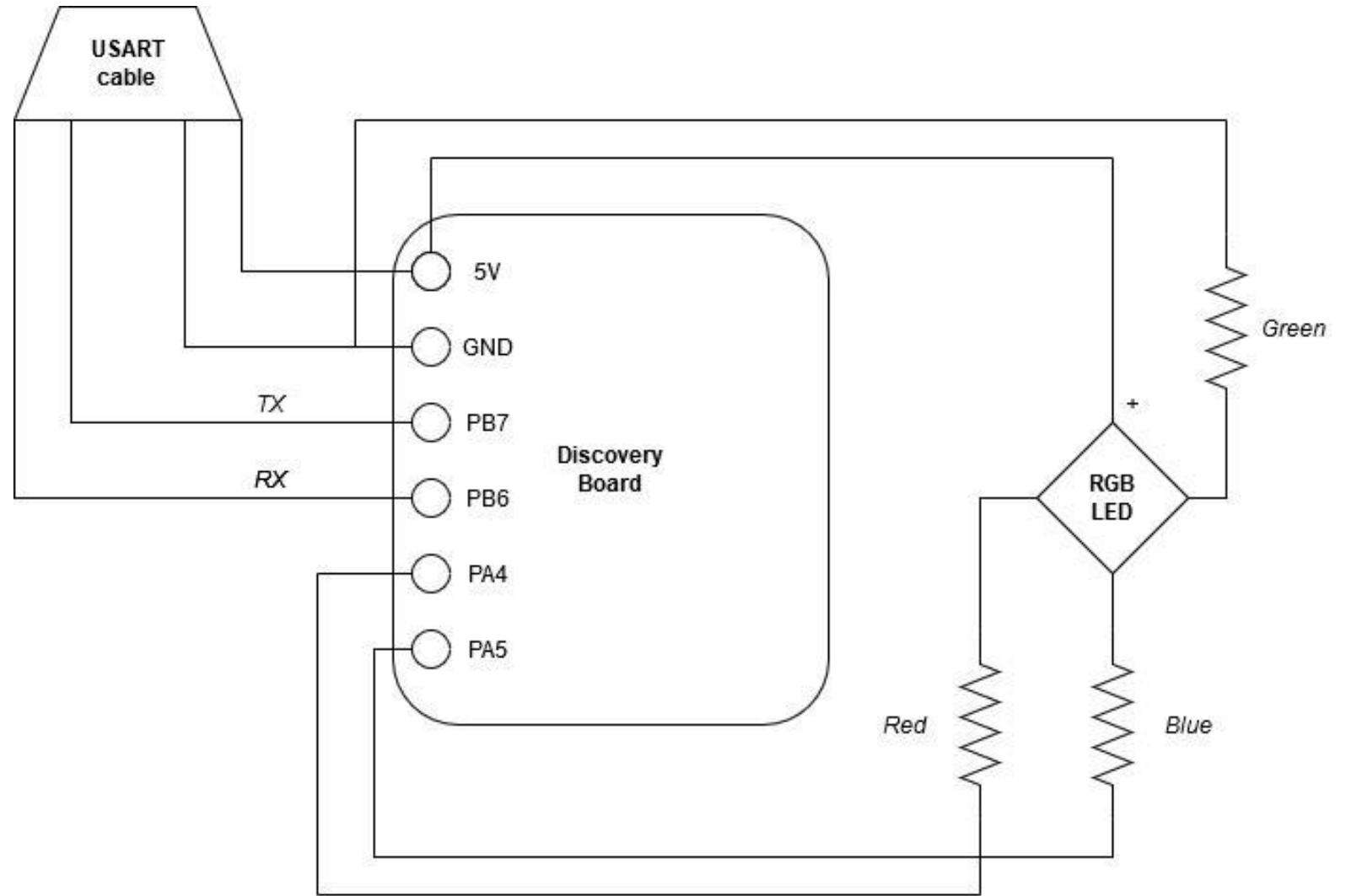
Hardware

Discovery board

RGB LED (common anode)

560 Ω resistors (x3)

UART cable + serial terminal



Software

C programming language

Software Interrupt Handling

Discovery Board Configuration

```
HAL_GPIO_Init(GPIOB, &initStr);
HAL_GPIO_Init(GPIOC, &initStr2);

GPIOB->AFR[0] &= ~(1 << 24);
GPIOB->AFR[0] &= ~(1 << 28);

RCC->APB2ENR |= RCC_APB2ENR_USART1EN;
USART1->BRR = HAL_RCC_GetHCLKFreq() / 115200;
USART1->CR1 |= USART_CR1_TE | USART_CR1_RE;

USART1->CR1 |= USART_CR1_RXNEIE;
NVIC_EnableIRQ(USART1_IRQn);
NVIC_SetPriority(USART1_IRQn, 3);

USART1->CR1 |= USART_CR1_UE;

/* ----- Now set up DAC! ----- */
RCC->APB1ENR |= RCC_APB1ENR_DACEN;

GPIOA->MODER |= GPIO_MODER_MODER4_0 | GPIO_MODER_MODER5_0;

DAC1->SWTRIGR |= DAC_SWTRIGR_SWTRIG2 | DAC_SWTRIGR_SWTRIG1;

DAC1->CR |= DAC_CR_EN1 | DAC_CR_EN2; //Finally, enable

while (1)
{
    if(!promptPrinted) {
        sendString("Color? (r or g)\r\n");
        promptPrinted = 1;
    }
}
```

Results

The Final result of the project was a Python terminal that accepted an input of a color letter, and a brightness.

For example, if "r 255" was entered, the LED would turn Red and be very bright. While "b 25" would change the LED to blue and not be that bright.

THANK
YOU

