Addressable RGB LED Node

Luke Longo and Kevin O'Hare
Rowan University

October 21, 2018

1 Design Overview

The overall objective of Milestone 1 is to build addressable RGB LED nodes on one of the four boards used this year. We decided to use the MSP430F5529 and its pins, 1.2, 1.3, and 1.4, for our LED outputs. These pins were connected with jumpers to a breadboard with one RGB LED. This LED was a common anode and had four pins being red, blue green, and voltage supply. On the MSP430F5529 our Pin 1.2 is red, Pin 1.3 is green, and Pin 1.4 is blue. Our code is designed to take an input of bytes from a terminal such as PuTTY or Realterm. The first byte is the packet size, the rest are the duty cycle's for the red, green and blue LEDs. The bytes received ont the RX line from the terminal will change the individual duty cycles of the red, green, and blue LED's, changing the overall color of the LED on the breadboard. Afterwards, all received bytes will be sent along to the TX line.

1.1 Design Features

These are the design features:

- UART Communication
- Able to send instructions to other RGB LED nodes

1.2 Featured Applications

Possible applications include:

- Color Coded Signals
- Lighting Strips
- Back-lights

1.3 Design Resources

The code for this assignment is located on github.com. The link to our repository is:
`https://github.com/RU09342-F18/milestone-1-art-thou-feeling-it-now-mr-krabs/blob/master/Milestone_StrangerThings/MilestoneV2/main.c`

1.4 Block Diagram

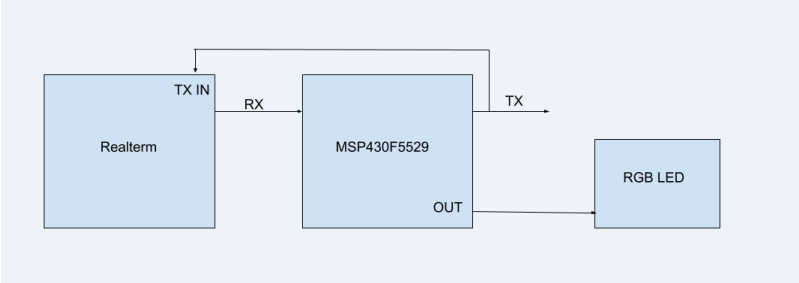

Figure 1: Simple Block Diagram

1.5 Board Image

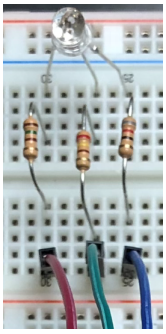

Figure 2: Photo of RGB LED Node

2 Key System Specifications

PARAMETER	SPECIFICATIONS	DETAILS
Micro Controller	MSP430F5529	Micro controller required for code
RGB LED	Common Anode	Type of RGB LED required
Anode Voltage	3.3V	Voltage for RGB LED Anode

3 System Description

This system is used to take byte commands from a terminal such as PuTTY or Realterm and use UART for communication from the terminal to the microcontroller and vice versa. UART only has an outgoing line to send data called TX or Transmitter, and also has an incoming line to receive data Rx. The byte data is received from the terminal and decoded in the order of:

- **Byte 1**: Packet Size
- **Byte 2**: Duty Cycle for Red LED
- **Byte 3**: Duty Cycle for Green LED
- **Byte 4**: Duty Cycle for Blue LED

After the fourth byte is used, the rest are sent along the TX line.

Pins 1.2, 1.3, and 1.4 are used to output the duty cycle values to the RBG LED on the breadboard. Pin 1.2 is used for red, Pin 1.3 is used for green, and Pin 1.4 is used for blue. The transmit line TX will be used to transmit the byte data we received to another board so we can then receive more data to perform the next action.

3.1 Detailed Block Diagram

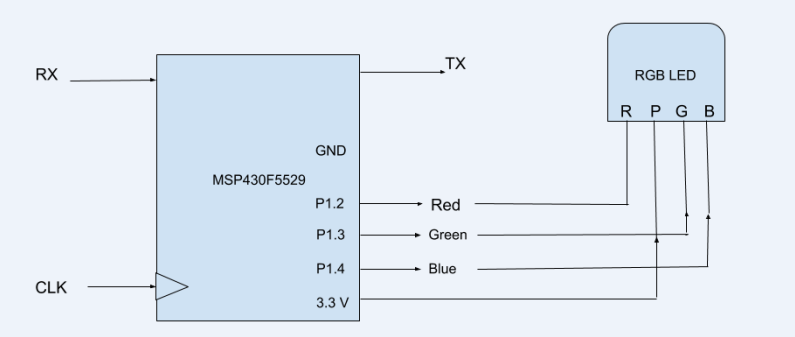

Figure 3: Detailed Block Diagram

3.2 Highlighted Devices

1. **MSP430F5529** - The MSP430F5529 (referred to as the F5529) is a micro controller manufactured by Texas Instruments. It is used for its timer module, GPIO pins, and its UART mode. The timer module Timer A0 is capable of setting or resetting GPIO pins based on values in its Capture/Compare registers (referred to as CCR). When the timer count reaches a value stored in a CCR, an action will be performed. UART mode allows the micro controller to receive and transmit data via RX and TX pins respectively. We also use the RX interrupt service routine to perform action upon receiving a byte on the RX line.

2. **Common Anode RGB LED** - The RGB Node for used for this code was constructed using a common anode RGB LED and three resistors. Each resistor was chosen so that 10 mA would pass through each color section of the RBG. The resistor values chosen were 150 Ω, 100 Ω, and 82 Ω for red, green, and blue respectively.

4 SYSTEM DESIGN THEORY

The operation of the RGB Node is heavily reliant on the F5529's timer module and UART capabilities. The timer module allows us to output to a pin when the timer reaches a value in the selected CCRx. It can also send the opposite output when the timer reaches CCR0. For example, if we wanted to have a pin set at when the timer reaches a value in CCRx, and reset when the timer reaches the value stored in CCR0, we can set the control register for CCrx to do just that. This allows us to change how often the LED turns off an on each second, giving us control of its brightness.

UART allows us to interract withthe RGB even further. By sending bytes to the F5529 over the RX line we can store values in the CCR's that control the LED pin's duty cycle. This means it is possible to directly set the brightness of each LED in the RGB LED.

4.1 Design Requirements

Our entire system is designed to make a single RGB LED perform various color combinations. In order for us to do this we needed to use CCR's to control the pulse width modulation of the red, blue, and green pins off the RGB. The MSP430G2553's timer module, Timer A0, has only three CCR's, and the MSP430F5529 for Timer 1 has five CCR's. If we chose the MSP430G2553 then we would have to use two different timers, but instead we made it easier on ourselves and used the MSP430F5529 which allowed us to use one single clock for our whole system. This allowed us to meet the design requirements of using four different CCR's, and keeping it simple with only having to implement one clock.

5 Getting Started/How to use the device

In order to get started on this project you will need to download the following software:

1. Code Composer
2. Realterm

Code Composer is a program that will allow you to directly write code to your board and test it all in one window. Realterm is a program that is used as a terminal that allows you to send bytes of information to your microprocessor.

For completing this project and showing off your results you will need the following equipment:

1. Breadboard
2. Jumpers
3. Resistors
4. RGB LED
5. Oscilloscope(Optional)

The RGB LED is placed on the breadboard with resistors before each pin for current protection. Using jumpers connect the 3.3 V pin, and LED output pins from your microcontroller to your breadboard in their respective places. For the common anode RGB LED, the order of the pins is red, power, green, and blue, with power being the longest pin.

6 Getting Started Software/Firmware

The software you will need for this device is Cod Composer Studio (CCS) and Realterm. CCS is used to create the program run by the micro processor. After a project is opened in CCS, it can be built for debugging by clicking the hammer icon on the toolbar. Then the program can be debugged by pressing the bug icon. This programs the micro controller with the program.

6.1 Communicating with the Device

Realterm is the preferable method of communicating with the F5529 as it is able to send bytes of data at a time. To setup up Realterm two things must be chosen: the desired baud rate, and the COM port linked with the processor. The baud rate for this device is 9600, but may be changed if necessary by tweaking the UART peripheral. The COM port can be found by connecting the microssor the you computer and checking your Device Manager (System Information for Mac users). Under Ports will be listed a COM port likely labled COM Port. This is the COM Port you will be using for Realterm.

Now that we know the baud rate and the COM Port, open Realterm and select the Port tab. Set the baud rate and COM Port. If you cannot find the COM Port in the drop-down list, you can double click the box to scan for ports. With these set, select the Send tab. From here you can send bytes to the F5529 by writing numbers in the form ”0xYY” where ”YY” is a 2-digit, hexadecimal number of 8 bits.

7 Test Setup

Testing this device can be done using the CCS Debug mode. Before entering the debug mode, breakpoints can be added to your code by double-clicking the area to the left of your code's scroll bar. When you run the code, the compiler will pause

at these breakpoints, allowing you to stop and checking any registers or values of interest. Variables can be viewed by going to "View" and selecting "Expressions". Then, select "*Add new expression*" and enter the name of the variable you wish to keep track of. This will allow you to view the variable's value while the program is paused in debug mode.

To test the TX line to confirm the board is properly transmitting and receiving, an oscilloscope can be used to to observe the serial signal on the TX or RX line.