RGB LED Node

Andrew Hollabaugh
Rowan University

November 1, 2019

1 Design Overview

This project is an RGB LED controller that can be used in a network of similar devices to create an individually-addressable RGB LED strip. An MSP430G2553 processor is used, on the MSP430G2ET development board. Three PWM pins on the processor are used for each channel of the RGB LED. UART serial communication is used for the nodes to receive commands.

1.1 Design Features

- Three output channels to output to three LEDs or one RGB LED
- Independently controllable output channels
- Communication to other nodes over UART

1.2 Featured Applications

- RGB LED strip
- Array of RGB lighting fixtures
- Sending visual information

1.3 Design Resources

Link to Github page

1.4 Block Diagram

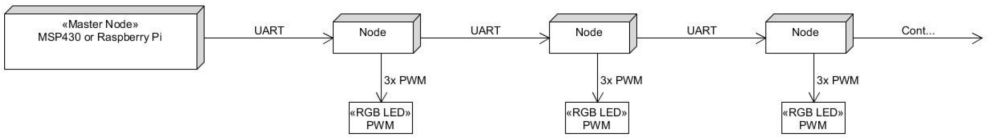


Figure 1: System Block Diagram

1.5 Board Image

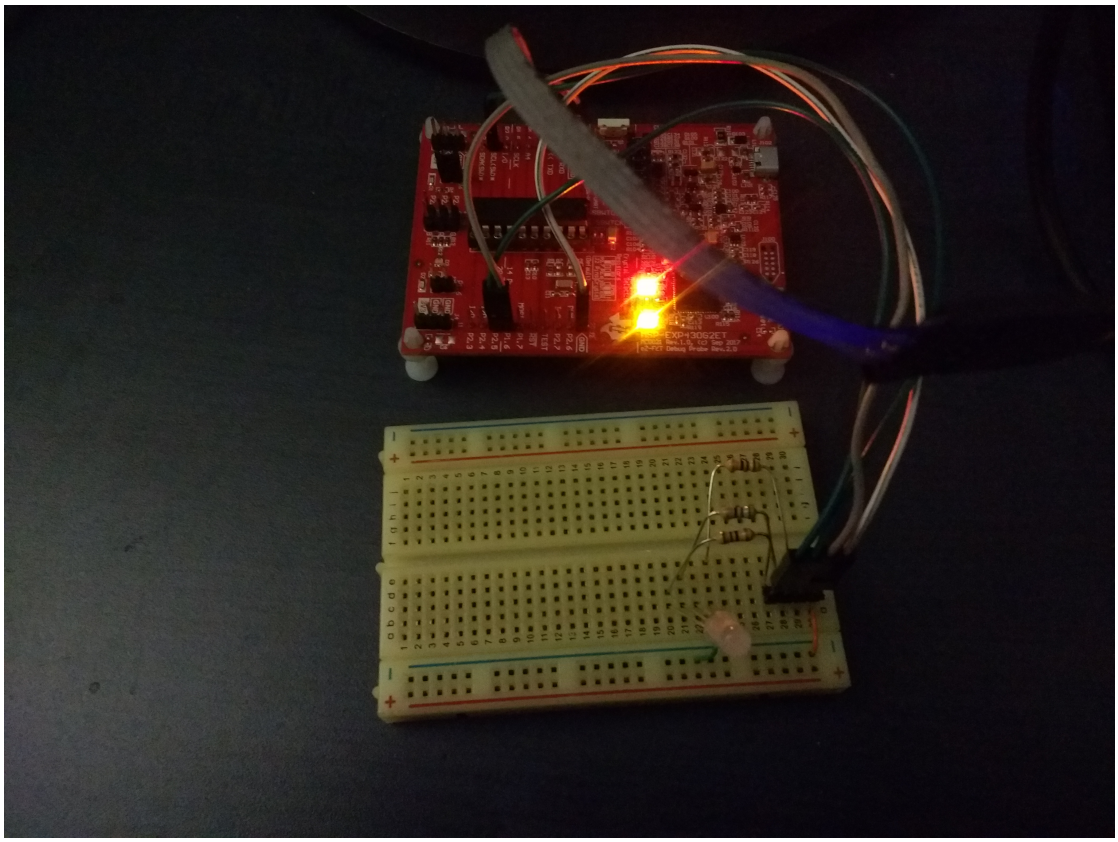


Figure 2: Picture of Breadboard and Dev Board

2 Key System Specifications

You can organize these as a table. This is meant to talk about the specifications which your system is capable of performing. I highly recommend looking up a LaTeXtable generator online.

PARAMETER	SPECIFICATIONS	DETAILS
Communication	UART (serial)	UART Serial communication is used with 8 data bits, 1 stop bit, and 9600 baud rate. A stream of bytes is received, some of the bytes are used to control the brightness of the leds, then it transmits the rest of the data to the next node.
LED Channels	3	The LED output channels go to each color of the RGB led (red, green, and blue). Other colors could also be used.

3 System Description

This device allows individual nodes in a network, to be configured independently. By using RGB LEDs, a network of these nodes create a string of LEDs whose colors can be independely configured.

3.1 Detailed Block Diagram

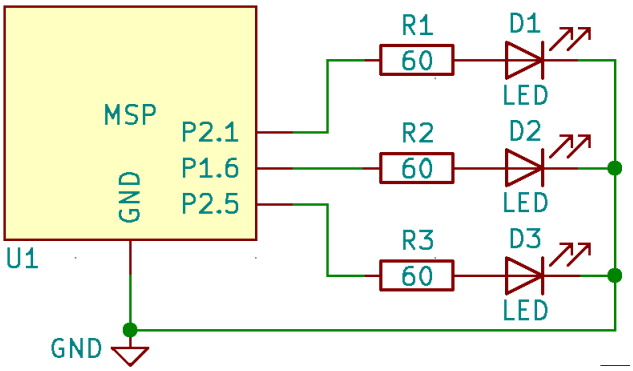


Figure 3: Detailed Block Diagram

3.2 Highlighted Devices

- MSP430G2553 processor (on MSP430G2ET development board): processes serial data and creates PWM signal for LEDs
- RGB LED: lights up in a combination of the three primary colors (red, green, and blue) corresponding to the PWM signals received from the MSP
- USB-to-serial cable: contains FTDI chip that allows the USB port of a computer to communicate with the MSP's UART interface

3.3 Device/IC 1

The MSP430G2553 is a 16-bit, RISC-based microcontroller. In this project, it operates at 3.3V and has a clock frequency of 1 MHz. This processor is a good choice due to its low power consumption and its UART and PWM output capabilities. This device receives a string of bytes over UART, and interprets the string as a set of commands. It picks out the values for the three LED brightnesses, then sends the rest of the data to the next node. One the brightness values are known, it outputs three PWM signals, one for each LED channel, to the LEDs. The duty cycle of the wave, which determines the channel's brightness, is determined by an 8-bit value in transmission it receives.

3.4 Device/IC 2

The RGB LED contains three single-color LEDs, one for each primary color (red, green and blue). Therefore, it can be viewed as three separate LEDs for simplicity. The RGB LED has four pins: one is ground, and the other three are positive voltages for each LED. The LEDs are connected to the MSP430 via digital PWM output pins and a series resistor. The series resistors were calculated to be 60 Ohms, using the forward voltages of the LEDs.

4 SYSTEM DESIGN THEORY

The program running on the MSP430 does two major things: interpret and send serial commands, and output PWM signals to the three LEDs.

4.1 Serial Communication

The serial communication protocol used is UART (Universal Asynchronous Receiver Transmitter). This protocol allows for easy tranmission of bytes. A standard protocol is used so the nodes can communicate effectively. This protocol can be any number of bytes long, and always starts with a byte containing the number of bytes in the entire transmission. The next three bytes are the brightness values for the first node's LEDs in the order of red, green, then blue. The next sequence of bytes is is the brightness values for next node's LEDs. Any number of multiples of three bytes can be added to this section to add commands for more nodes. The last byte of the transmission is always a newline (0x0d). In the way this protocol is interpreted by the program, the newline character is uncessesary. The program cannot use detecting newlines as a way to detect the end of the transmission, because 0x0d could be a brightness value. Instead, only the first byte (number of bytes in transmission) is used. The program keeps track of how many bytes have been processed to determine the end of the transmission accurately. The program uses an interrupt, which triggers when a UART character is received. When the interrupt is not being run, the system is doing nothing and is set to low-power mode 0 (LPM0).

4.2 PWM Signals

The MSP430 uses three PWM outputs to control the three LED channels. Timers are used to generate the PWM signals. The MSP430G2553 contains two TimerA peripherals, each of which has two PWM outputs. Two PWM outputs from TimerA1 are used, as well as one PWM output from TimerA2. The timers are set to UP mode, with SMCLK as the clock source. No prescaling is used for a high frequency output, resulting in a more constant brightness. The brightness values from the serial command are assigned to the capture/compare registers. The PWM signal is generated by comparing the value in the capture/compare register (the brightness value) with the current timer value. This comparison between a horizontal line and a triangle wave results in a square wave, the PWM signal. Pins P1.6, P2.1, and P2.5 are used for PWM output. Only certain pins can be used, since only certain pins are connected to the PWM-generating hardware.

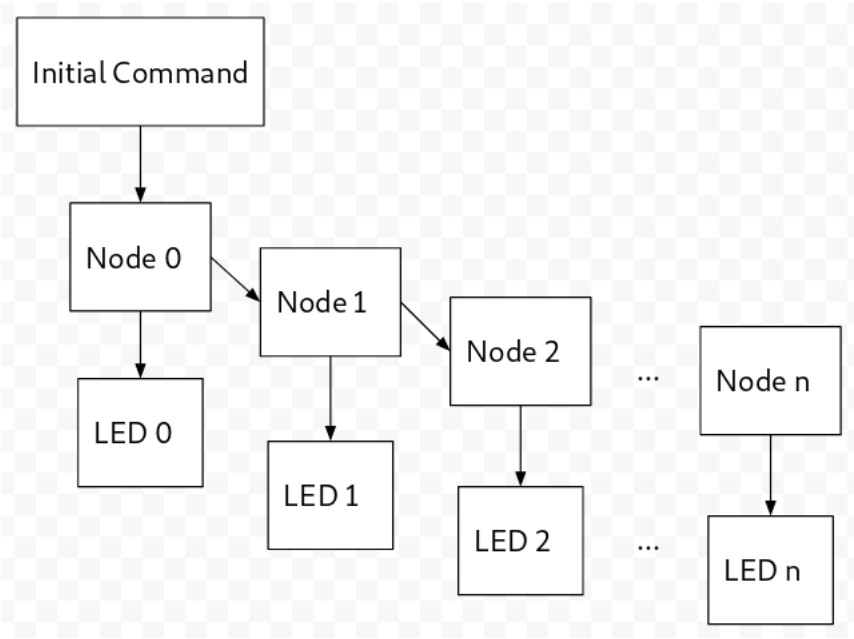
5 Getting Started/How to use the device

Connect the LED and resistors on a breadboard according to Figure 3.
Interfacing this device with a computer involves using a USB-to-serial cable. Connect the +5V, GND, RX, and TX on the UART side of the USB-to-serial cable to the corresponding pins on the MSP430. Connect the USB end to a computer.

6 Getting Started Software/Firmware

The code needs to be compiled using msp-gcc. MSP430Flasher is a good tool for flashing the chip.

6.1 Hierarchy Chart



6.2 Communicating with the Device

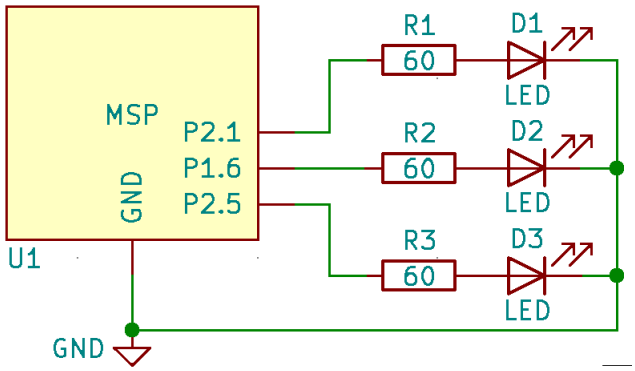
Use a serial terminal to communicate with the node. First it must be connected using a USB-to-serial cable. Ensure the serial terminal is set to 9600 baud rate, and 8 data bits and 1 stop bit. Data is sent in the following format: first byte is number of total bytes in transmission, next three bytes are the R, G, and B values for the first node, the next any number of bytes (in multiples of three) is the R, G, and B values for nodes further down the line, and the last byte is a newline (0x0d).

7 Test Setup

Ensure the program is flashed onto the MSP430. Use a breadboard to connect the RGB LED to the three resistors. Use jumper cables to connect to the three pins on the MSP430. Use a USB-to-serial cable to connect the MSP430's UART to a computer. Use a serial terminal to connect to the processor, such as CuteCom, minicom, RealTerm, putty, etc. Send an example command, such as 05ffffff0d. This should set all three LEDs to full brightness, making the RGB LED appear white in its full brightness. To connect more nodes, connect the TX pin of the first node to the RX pin of the second. Also ensure they are both powered. Any hex values after the first four will specify brightnesses for the second node.

8 Design Files

8.1 Schematics



8.2 Bill of Materials

- MSP430G2553
- RGB LED
- 60 ohm, 1/4 watt resistor (3)