RGB LED Control with the MSP430G2553

Scott Wood and David Sheppard
Rowan University

October 22, 2018

1 Design Overview

Milestone 1 involved the design of a user controlled RGB LED with the utilization of UART communication. The MSP430G2553 was programmed to act as both a transmitter and receiver of desired binary packets when connected with a UART cable and configured to a serial terminal. This allowed for the linkage of several microprocessor devices, and increased overall user RGB LED control on each development board. The RGB LED control was accomplished with Software PWM such the RGB LED would respond upon a received packet of valid bytes from 0x00-0xFF. Specific received byte(s) allowed for the modification of the duty cycle of each RGB color such that it they can be individually on at specific duty cycles, all on, or all off.

1.1 Design Features

These are the design features:

- 9600 BAUD rate
- Accepts all valid byte values (0x00 through 0xFF)
- Utilizes low power mode to ensure that power consumption is minimized
- Software PWM

1.2 Featured Applications

Possible applications for the technology:

- Warning lights
- Color displays

- Decorative lighting
- Communication

1.3 Design Resources

The code and README for the project can be found at the follwing link: `https://github.com/RU09342-F18/milestone-1-team-nice/tree/master/Milestone_StrangerThings`

1.4 Block Diagram

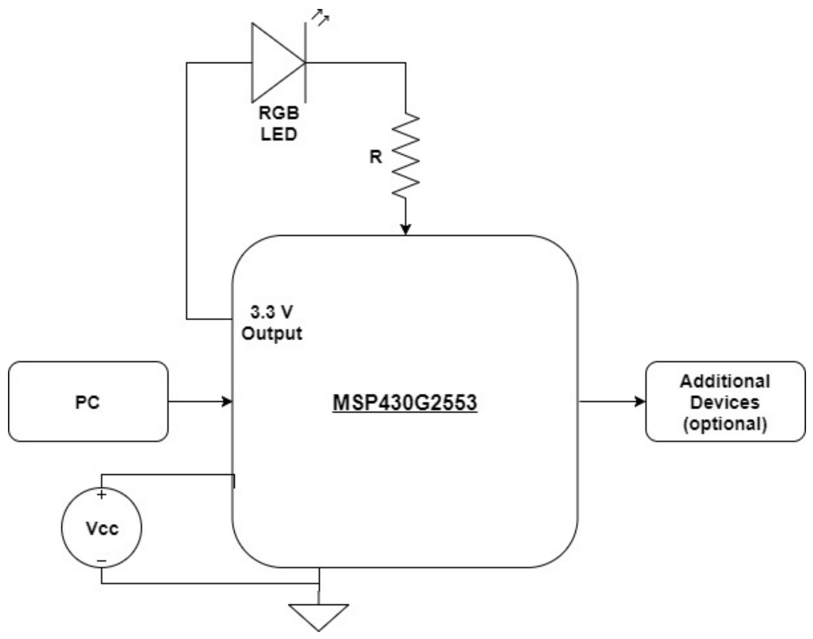

Figure 1: Basic Block Diagram of the System

1.5 Board Images

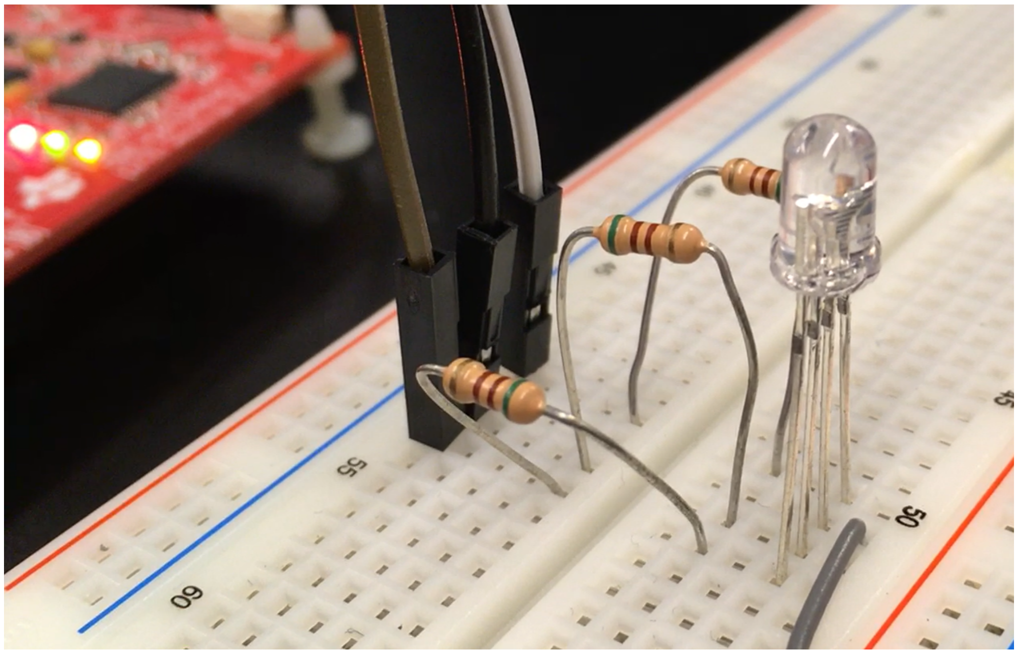

Figure 2: LED Wiring on the Breadboard

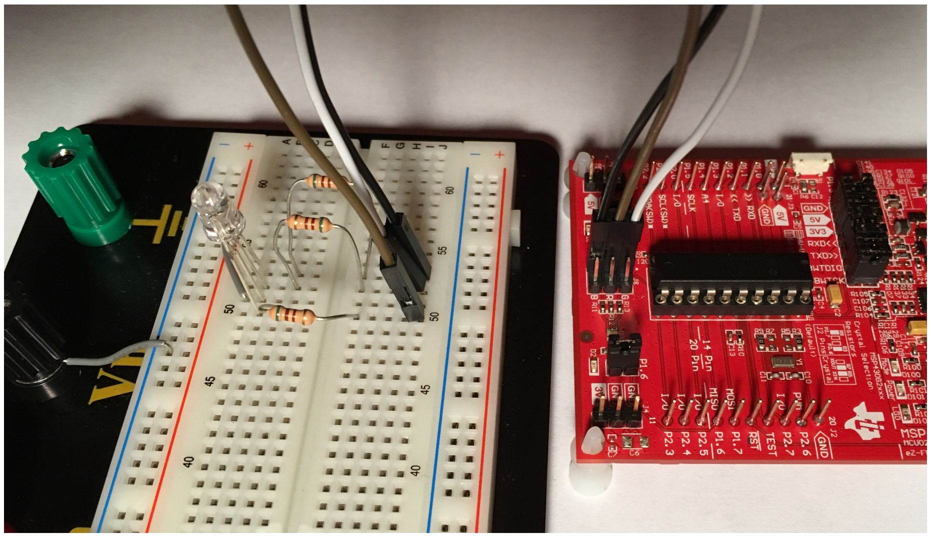

Figure 3: Complete Wiring on the Breadboard

2 Key System Specifications

PARAMETER	SPECIFICATIONS	NOTES
Maximum current per pin	6 mA	Cannot exceed 48 mA for entire circuit
Minimum supply voltage	1.8 V	
Maximum supply voltage	3.6 V	
Optimal RGB LED type	common anode	
Minimum operating temperature	-40° C	
Maximum operating temperature	85° C	
Low Power Mode	0	ACLK and SMCLK High
Operating clock frequency	8 MHz	Created from 1 MHz clock using frequency multiplier

3 System Description

Communication through the means of a user controlled RGB LED programmed by a micro-controller was a challenge that required immense research into the MSP430G2553 device, its respective capabilities to function as a receiver and transmitter, and the overall system requirements that needed to be satisfied to meet the goal, and be optimal when compared to other micro-controllers' performance.

This research into the performance and features of the MSP430G2553 led to the following overall system for the user controlled RGB LED program: External RGB LED connected to breadboard in Common Anode design such that P2.1, P2.3, P2.5 are connected to the cathode of each respective red, green, and blue node of the LED. The RGB LED is controlled by the user through the means of a UART cable in connection with the USB port of the MSP430G2553 such that UART commands are transmitted to the device by RealTerm and received by the device at Baud Rate of 9600, which is configured by the user within RealTerm.

This single system can then be extended to include multiple MSP430 launchpad devices such that one can be configured to act as the master node and transmit UART commands and communicate to other linked MSP430 devices such that the range of binary packet transmission is increased and each RGB LED node has a specific message. Doing this simply required that a connection from the Tx port of the master node device be connected to the Rx port of the receiving MSP430 device, this set up is continued for the desired transmission range.

3.1 Detailed Block Diagram

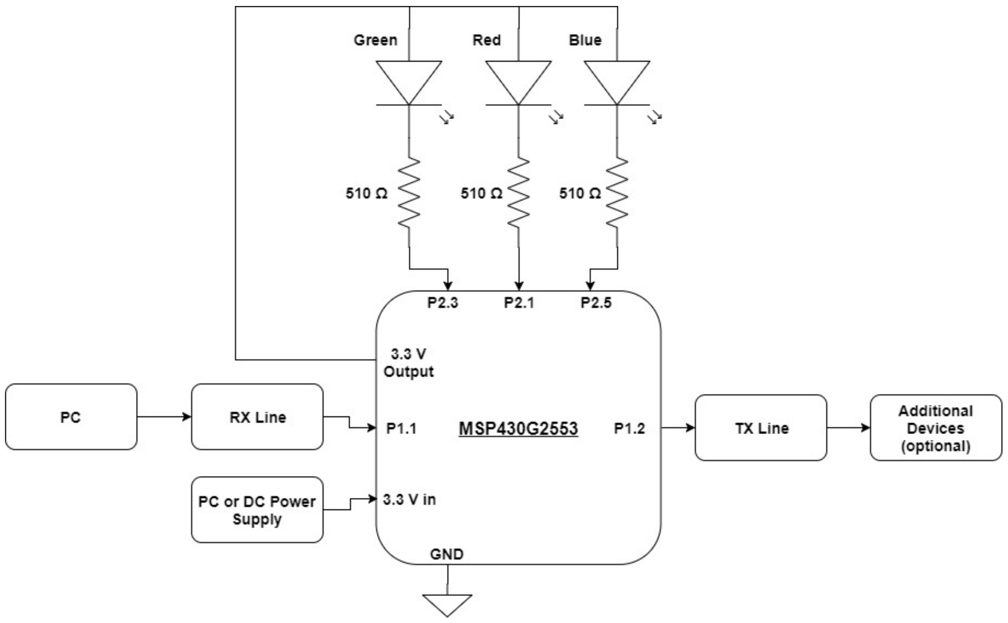

Figure 4: Detailed Block Diagram of the System

3.2 Highlighted Devices

Necessary components to construct the system are as follows:

- MSP430G2553
- RGB LED
- Breadboard
- 3x 510 Ω Resistors
- Breadboard jumper wires
- Male to female jumper wires

3.3 Device 1: MSP430G2553

Out of all the available micro-controllers to act as the master node of the overall system of UART communication, the MSP430G2553, which is a 20 pin device was chosen for its ease to be programmable in software so that it could perform the required goal of a user control RGB LED and communicate with other devices. Also, the MSP430G2553 has the unique functionality that it can be programmed on the

MSP-EXP430G2 launchpad to control the on board RGB LEDs or it can be separated from the launchpad altogether and control separate LEDs with the Tx and Rx lines connected with a wire leading to the launchpad. The MSP430G2553 is equipped with two 16 bit Timer A modules: Timer A0 and Timer A1 both equipped with three capture compare registers. Timer A1 was selected to perform the software PWM since it is sourced to the RGB LEDs on the MSP-EXP430G2 Launchpad: P2.1, P2.3, and P2.5.

The MSP430G2553 can also be placed into low power mode with ease and was done so in this specific program. In the specific case of this program, the MSP430G2553 was set to Low Power mode 0, which means that the CPU and MCLK are disabled, and that ACLK and SMCLK remain active, which was all that was necessary to perform the main functionality of this program. The sub system clock was utilized in the program and was initialized to clock at a frequency of 8MHz to reduce visible blinking of the LED.

3.4 Device 2: RGB LED

The default implementation for this program involves a common anode RGB LED which an be placed onto a breadboard or included as part of a PCB for specific applications. With a common anode RGB LED, the common pin shared by all of the devices is the pin that must be connected to the higher voltage. The other three pins, each corresponding to the red, green, and blue LEDs within the complete RGB LED, are then connected to pins 2.1, 2.3, and 2.5 of the MSP430G2553. As a result, the LEDs turn on when the pins of the microprocessor are at a low voltage (ground) and are off when the pins are at a high voltage (3.3 V).

The default implementation as seen in Fig. 2 and Fig. 3 shows each color pin of the RGB LED connected to a resistor between the LED pins and the microprocessor. This prevents high current flow wich could damage the circuit (including the processor). For this implementation, the resistors chosen were 510 Ω resistors. This helps to keep the current at around 6 mA per pin when the LED is on.

4 SYSTEM DESIGN THEORY

In general a 3.3V power supply is provided to the first main part of the system, which can be described as the 20 pin MSP430G2553 IC. Also, a proper UART cable is connected to the USB connection of the MSP-EXP430G2 launchpad, which enables UART functionality such that any user of a PC can control the RGB LED with UART commands, which is connected to the 3.3V output of the MSP430G2553 IC.

The first main part of the system being the MSP430G2553 IC, allowed for configuration of its TimerA1 module to allow for proper Software PWM and to control the overall dimness of the RGB on the MSP-EXP430G2 launchpad (P2.1, P2.3, P2.5) or on a separate RGB LED configured on a breadboard in a common anode configuration. This device also enabled UART functionality since it could be programmed to perform specific duty cycles for corresponding received bytes and transmit binary packets of its own along the Tx channel so it could communicate with other devices. As of right now, the main functionality of the system is that it can act as either a master node, or act as a link in a node of many micro-controllers, which means it can both transmit and receive in such a way that zero error is made on either end to allow for successful communication.

The secondary part of the overall system, which can be described as the user controlled RGB LED acts as the means of communication for the device, and allows purpose for the received and transmitted binary packets along the UART channel. Without the RGB LED, the overall communication goal of the system would not be successful since the binary data or UART commands would not represent anything.

4.1 Design Requirement 1: Pulse Width Modulation

The overall goal of enabling the MSP430G2553 to dim each LED for a corresponding byte required several design questions to be answered and corresponding decisions to be made regarding how the PWM would be performed with the limited capture and compare registers (CCRs) available on the MSP430G2553. In order to efficiently perform PWM with the device, it was decided that software PWM would be implemented with Timer A1.

Software PWM was chosen because hardware PWM would require the use of multiple timers and would increase the complexity of the program. This is due to the fact that when performing hardware PWM, the timer must be set to operate in a specific output mode, or OUTMOD. If operated in Up Mode, the set/reset OUTMOD can be used to perform PWM. In this case, the timer's CCR0 would be dedicated to holding the value at which the timer would overflow. As a result, only CCR1 and CCR2 would be able to correspond to a duty cycle, thus requiring the use of two timers in order to perform PWM for three signals. Additionally, no other OUTMOD would produce desirable results for hardware PWM if used in either continuous mode or up/down mode. These other methods would require even more timers or an eve greater complexity ot acheive the desired results.

As a result, software PWM was deemed to be the optimal method when performed with Timer A1 in continuous mode. Unlike Timer A0, Timer A1 has three capture and compare registers. Therefore, software PWM can be performed with Timer A1 in continuous mode such that each CCR value corresponds to the duty cycle of one of the LED colors. This allows for simplistic implementation of the duty cycles and requires minimal hardware usage.

4.2 Design Requirement 2: Clock Speed

The internal clock within the MSP430G2553 that was utilized as the main clock source was determined and selected to be the sub system clock (SMCLK), which was necessary since the MSP430G2553 was set and desired to be used in low power mode 0, which limited the clock source selection to be either: ACLK or SMCLK. Between the two, the SMCLK was selected because the peripherals of the MSP-EXP430G2 were to be utilized so that software PWM was available for each peripheral pin of the MSP430G2553, specifically P2.1, P2.3, and P2.5. Initially the clock source was set to 1MHz with zero divider, and an undesired blinking effect would occur at the initial start of the user RGB LED program, and would disappear when UART commands were received. To eliminate this issue and stay consistent with the utilization of Software PWM with only timer, the internal clock source was set to 8MHz, and the blinking effect was completely fixed since the internal clock source was then set to blink at such a rate it would be negligible to the human eye.

4.3 Design Requirement 3: Turning the LED Off

Once final issue that was encountered during testing was the inability to fully turn off the RGB LED when the duty cycle was set to 0. This is believed to be caused by the fact that a CCR interrupt would occur at virtually the same time as a timer overflow if the CCR was set to 0. As a result, this special condition had to be implemented in a different way. Rather than using the software PWM with a duty cycle of 0 when the one or more of the LEDs should be off, the LED pins were set high (since the RGB LEDs were of the common anode type which require negative logic) when the inputted duty cycle was 0.

This was done by declaring three variables, each intended to represent the state (zero or nonzero) of the duty cycle of each RGB value. The value of this variable is then check in the software PWM code. If the duty cycle should be 0%, then the software PWM never lets the output pin(s) go low (which would turn on the LED). This prevents the desired LED pin from ever turning on while the duty cycle of that pin is set to 0%.

5 Getting Started/How to use the device

5.1 Connecting the Device

The development board requires both a source of power and UART input in order to function as intended. The micro USB cable that comes with the development board can be used to supply power to the board and communicate with the device using UART. If the microprocessor is being used while mounted on the development board, then the connectors on the RX and TX lines above the microprocessor must be flipped such that they are connected horizontally and are perpendicular to all of the other connectors around them. When running the program to control an RGB LED, the board can be used as a standalone device since it has an RGB LED or it can be used with an external RGB LED. If the board is being used with an external LED, pins 2.1, 2.3, and 2.5 must be connected to the red, green, and blue pins on the RGB LED, respectively.

5.2 Setting Up the RGB LED

While the program was written with a common anode RGB LED in mind, it can be made to work with a common cathode RGB LED as well. In order for this to work, pins 2.1, 2.3, and 2.5 would still be connected to their corresponding pins on the LED, however the last pin on the RGB LED would have to be grounded. As a result, the UART input values would have to be given such that 0x00 corresponds to a 100% duty cycle and maximum brightness while 0xFF would correspond to a 0% duty cycle in which the LED is off. This also applies when using the program with the development board's built-in RGB LED, since that LED is also of the common cathode type.

6 Getting Started Software/Firmware

The program itself, which allows for the user control of an RGB LED with the MSP430G2553 is written in C within an Integrated Development Environment (IDE) called Code Composer Studio (CCS). One would have to have CCS installed and the correct MSP430G2553 Family Device library selected to run and build the program on the MSP430G2553 and see the initial start of the program, which would be the LEDS in an off state since the LEDS are set to be off initially. The user can then communicate with P2.1, P2.3, P2.5 and control the RGB LED display with use of a serial terminal program called RealTerm.
In order to successfully transmit data with the use of RealTerm, the proper UART settings had to be calibrated within RealTerm, which required a 9600 baud rate to be selected and set from the drop down menu labeled as Baud. This Baud Rate defined the rate at which the binary packets were transferred in the UART communication channel. Furthermore, the user has to ensure that the correct COM port is selected for proper communication to be achieved from their device to another or to their own device, in order to do this device manager must be pulled up and the USB PORT number, which is connected to the UART cable and visible in device manager must be entered and selected from the drop down menu labeled as Port.

6.1 Hierarchy Chart

6.2 Communicating with the Device

Information can be sent to and received from the MSP430G2553 using a UART cable and a terminal such as RealTerm. In order to transfer data from a PC to the device, the MSP430G2553 chip can be inserted into the MSP-EXP430G2ET development board and the development board's micro USB cable should be used to connect the development board to the PC. Upon making these connections, the user must then

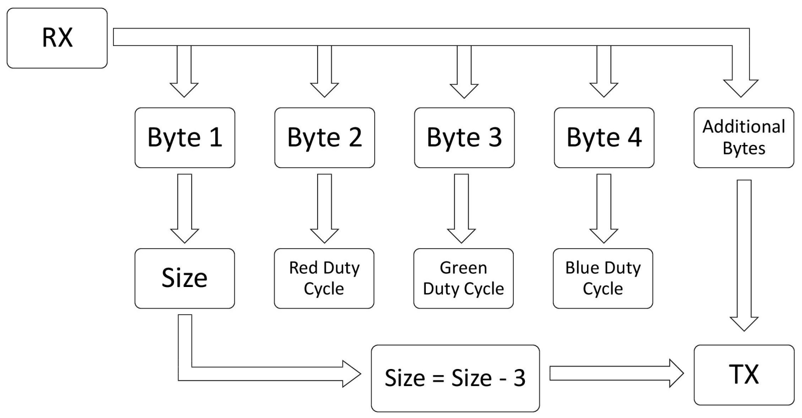

Figure 5: Dataflow and Hierarchy of UART Information

open the PC's device manager and find the COM Port number of the connected board. This is found under *Ports (COM & LPT)* in the device manager. The desired COM Port number is that which follows the term *UART* in the list.

Once the COM Port number is determined, RealTerm can be opened up and the BAUD rate (9600) and COM Port number specified under the *Port* tab. Under the *Send* tab, the desired bytes can be send as hexadecimal numbers using hexadecimal notation (e.g. the decimal number 255 could be sent as 0xFF).

When sending data, the first byte corresponds to the size of the string of bytes that will be sent after it. The bytes that follow the size byte correspond to RGB values, where the first byte following the size is the red LED's value, the second is the green LED's value, and the third is the blue LED's value. All following bytes are passed on out through the TX line of the processor which can be connected to the RX line of another device. The flow of information is represented in Fig. 5.

6.3 Device Specific Information

The MSP-EXP430G2ET development board contains its own RGB LED and can also be connected to an external LED. The program was designed for use with an external common anode RGB LED. The red, green, and blue pins of an external LED must be connected to pins 2.1, 2.3, and 2.5, respectively. Additionally, it must be ensured that the current through any input or output pin does not exceed 6 mA. Likewise, the total current through the device must not exceed 48 mA. Additionally, the supply voltage for the processor must remian between 1.8 and 3.6 V.

While the software was designed for use with an external common anode RGB LED, the program can be used with the on board LED by allowing the UART commands to represent the duty cycle of the time that the LED spends off (e.g. 0x00 would turn the LED on and 0xFF would turn it off). This also applies if using the program with an external common cathod RGB LED.

7 Test Setup

To verify the functionality of the user controlled RGB program a specific packet of bytes: 0x03 0xFF 0xFF 0xFF was sent from the serial terminal: RealTerm, which for proper communication required that a proper UART cable or MSP430G2553 USB be connected to the USB connection on the MSP-EXP430G2ET Launchpad to a PC for proper power and UART communication. This test sequence resulted in the RGB LEDs (P2.1, P2.3, and P2.5), to turn on at maximum brightness and a color of white was generated and 0x00 was sent back and appeared on the terminal display of RealTerm.

Furthermore, the user RGB LED control program was verified to function the same with a RGB LED separate from the on board RGB LED on the Launchpad. This RGB LED was designed in a common anode configuration, meaning the 3.3VDC was connected to the Anode, and P2.1, P2.3, P2.5 were connected in series with a 510 Ω resistor to the cathode. This allowed for better visualization and demonstration of the several distinct duty cycles that were within the required test sequence of each byte packet. For example, to truly test the user control RGB program the following test sequence was sent ”0x08 0x7E 0x00 0xFF 0x40 0xFF 0x00 0x0D.” Of course the first packet size byte had no effect on the duty cycle, but 0x7E set node red of the external RGB LED on at 50 % duty cycle, 0x00 set green off (0 % duty cylcle), 0xFF set blue at maximum brightness (100 % duty cycle), 0x40 set red at 25 % duty cycle, 0xFF set green to maximum brightness, and of course 0x00 set blue off, and 0x0D acts as the last byte to conclude the message.

To verify the overall goal of the project was completed and a functional user controlled RGB LED, which can communicate to other devices was achieved, a linkage of several MSP430 Launchpad devices was created such that the MSP430G2553 device mentioned in this report was the master node and transmitted successfully the several received UART commands. Furthermore, this same MSP430G2553 was set as a random node among a linkage of several microcontroller devices to verify it did not corrupt any received data, and would transmit successfully once again.

7.1 Test Data

The data was tested by sending the following sequences sent over UART:

- 0x03 0xFF 0x00 0x00
- 0x03 0x00 0xFF 0x00
- 0x03 0x00 0x00 0xFF

- 0x03 0xFF 0xFF 0xFF
- 0x03 0x7F 0x7F 0x7F
- 0x05 0x00 0xFF 0xFF 0x7D 0x4E

This produced a sequence of the following colors: red, green. blue, white, white at half brightness, and teal. When the final data set was sent, the processor returned the new size byte of 0x02 along with the final two values of 0x7D and 0x4E back through the TX line to the PC. These results were exactly as they should have been.

8 Design Files

8.1 Schematics

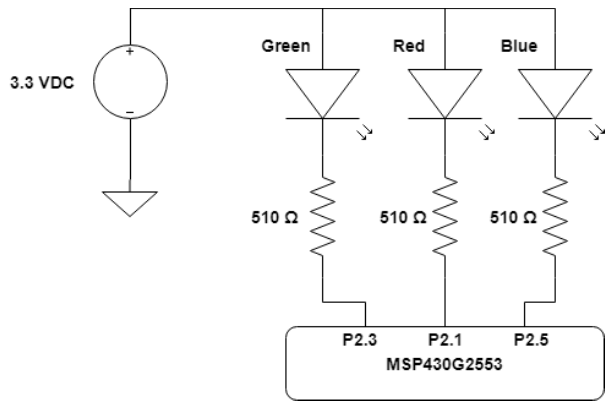

Figure 6: Schematic of Breadboard LED Setup

8.2 Bill of Materials

Device	Cost	Quantity
MSP-EXP430G2 LP	\$25.99	1
RGB LED	\$1.95	1
Breadboard Kit	\$3.99	1
510 Ω Resistors	\$1.47	3