Application Note Template

Jacob Matteo and Alex Jackson
Rowan University

October 31, 2019

1 Design Overview

This project set out to model a light wall from the show ”Stranger Things”. The board used for this was the MSP430G2553, and featured many elements showcased in previous labs. An RGB LED was positioned on a breadboard, that was then wired to the G2 micro-controller. A code was loaded onto the G2, which allows for the micro-controller to communicate with the RGB LED using UART. Sending HEX codes through the UART port changes the color of the RGB LED. The end goal is to connect every board in the class, allowing for each LED to receive data from one UART port, and send it to the next LED in the chain.

1.1 Design Features

These are the design features:

- UART communication through MSP430G2553 board
- HEX codes for certain colors of the RGB LED
- Pulse Width Modulation system to control the brightness of the LED, allowing for a variation of different colors to be showcased
- Able to send bytes to LED nodes that store information about the color of the LED
- Takes three HEX messages through UART for each color, and sends any additional messages along to the next node

1.2 Featured Applications

- Transmits data
- Receives data
- Can be used across multiple boards
- Used for programming LED displays

1.3 Design Resources

Alex's GitHub:
https://github.com/Intro-To-Embedded-Systems-RU09342/milestone-1-ajax7778.git

Jake's Github:
https://github.com/Intro-To-Embedded-Systems-RU09342/milestone-1-jmatteo.git

1.4 Block Diagram

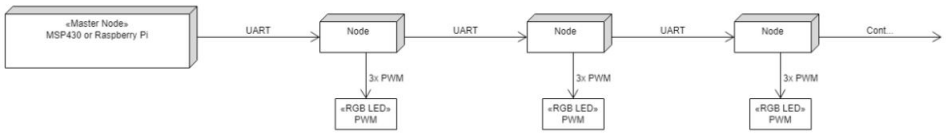

Figure 1: Simple Block Diagram

1.5 Board Image

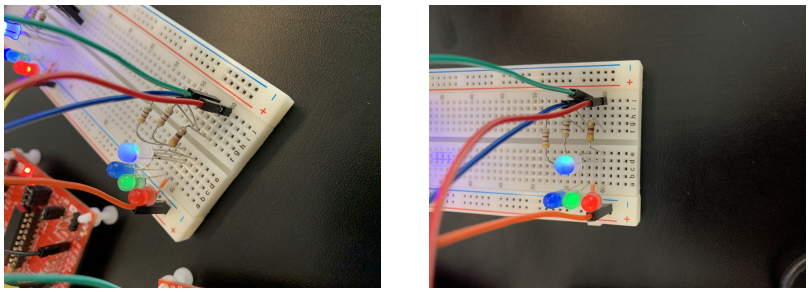

Figure 2: Breadboard Images

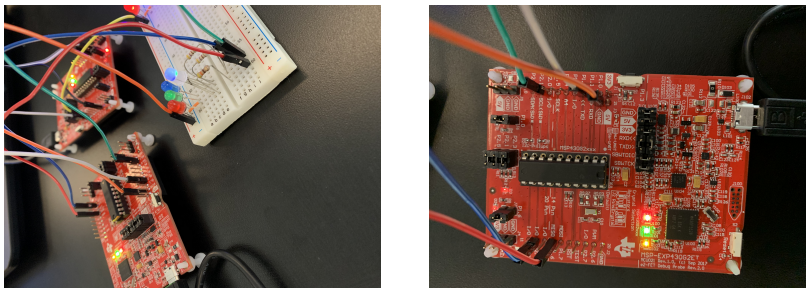

Figure 3: Connection to Micro-controller Images

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
TA0CCR0	P1.6	RED LED port of RGB timer
TA1CCR1	P2.1	GREEN LED port of RGB timer
TA1CCR2	P2.4	BLUE LED port of RGB timer
Forward Voltage	2V	RED LED
Forward Voltage	2.2V	GREEN LED
Forward Voltage	3.3V	BLUE LED

3 System Description

The problem trying to be solved is to create a color coded communication system based on an RGB LED and a microprocessor. The data is sent from the microprocessor to a breadboard where an RGB LED receives the data and displays the color. The data is able to be sent using the microprocessor's built in UART communication system. The data transmitted includes the pulse width for which each color of the LED should be set to, so that many different colors can be showcased. This processor and the code sent to it is designed to work with other processors, so a chain can be made that connects many of these systems together.

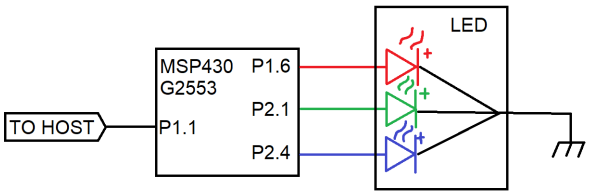

Figure 4: Detailed Block Diagram

3.1 Detailed Block Diagram

3.2 Highlighted Devices

- MSP430G2553: This is the microprocessor used for this project. The code is loaded onto this device and drives the RGB LED.
- Code Composer: The code was written in C using this software, and from there loaded onto the micro-controller
- Breadboard: RGB LED was placed on the breadboard along with the resistors, and the microprocessor was routed there
- RGB LED: Main output of the code

3.3 Device- MSP430G2553

The MSP430 family of microprocessors includes many different boards with hundreds of features. The MSP430G2 family is a part of the MSP430 series that includes multiple 16-bit timers, 24 accessible pins, ultra-low power, and UART communication capability. This specific board includes a 10 bit analog to digital converter. The reason this was the board chosen for this project is because the timers and clocks are easily programmable. The other option for a board was the MSP430FR6989, which had issues with a register that was not easily accessible.

3.4 Device- RGB LED

The RGB LED was used as the main source of communication for the light wall. It works by connecting it to a breadboard, and connecting a resistor to each of the RGB nodes in order to properly regulate the voltage.

4 SYSTEM DESIGN THEORY

The system design for this project contains two major components, the MSP430G2553 code, and the breadboard circuit. The breadboard supporting circuit is there in order

to take the inputs from the UART programmed in the G2 micro-controller, and display them as a unique LED color.

4.1 Pin Setup

The pins on the micro-controller are set up to correspond with the colors of the RGB LED. Pin P1.6 is connected to the Red LED, P2.1 to the Green LED, and P2.4 connected to the blue LED. The on board LED is also initialized to indicate when data has been received from UART. This pin is P1.0.

The other on board pins used are the transmit and receive pins, TXD (P1.2) and RXD (P1.1). The receive pin takes in data from the UART communication and processes it, lighting the LED according to the received HEX codes. The transmit pin connects multiple boards together, allowing for the UART to send data down a line of boards. The ground of the breadboard and the G2 are also utilized.

4.2 UART Communication

The G2 board allows for universal asynchronous receiver-transmitter, or UART, communication. The code sets the UART transmission speed to a baudrate of 9600. Using the software RealTerm, data bytes are sent to the G2 board through UART. The program then receives the data and counts the number of bytes. The program is set up so that the first byte sent indicates the total number of bytes, and the last byte in the string is 0x0D, to indicate the end. It takes the first three bytes after the byte that indicates the total number, and assigns them to the three LED colors. If there are more bytes in the string, the remaining data is repackaged and sent to the next board connected. The figure below shows an example byte string.

Byte Number	Contents	Example
Byte 0	Number of bytes (N) including this byte	0x50 (80 bytes)
Bytes 1-(N-2)	RGB colors for each node	0xFF (red) 0x00 (green) 0x88 (blue) ...
Byte N-1	End of Message Character	0x0D (carriage return)

Figure 5: Example Byte Configuration

In the above example, the first board connected should display a purple color. This is clear because the red LED is at full brightness, indicated by the HEX value 0xFF, and the blue LED is at over half brightness, indicated by the HEX value 0x88.

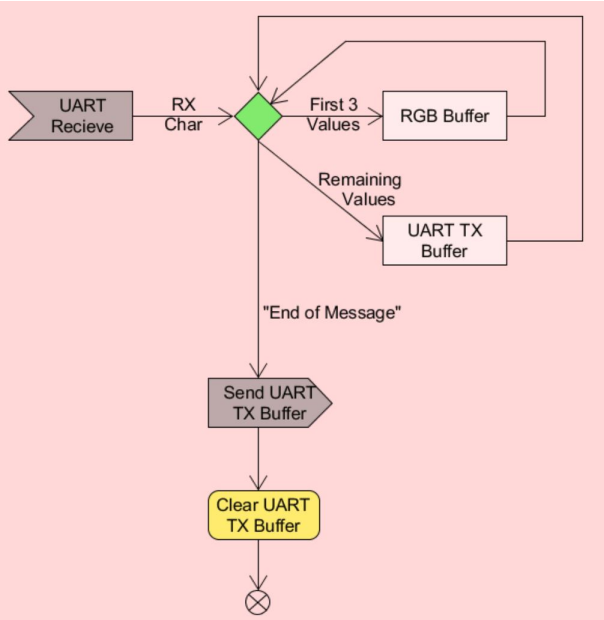

Figure 6: UART Communication Diagram

4.3 Pulse Width Modulation

The first requirement is the input of the PWM. PWM stands for Pulse Width Modulation, and allows for the LED's brightness to be controlled and display many different colors. The PWM works by modifying the duty cycle of the LED to a certain percentage. The lower the percentage, the dimmer the LED will be. With the RGB LED, the duty cycles of the red, blue, and green LEDs can be modified separately.

In the code, this is implemented using the timers capture and compare values. The LED pins are all connected to Timer A in the G2, and the capture and compare values control the duty cycle. The SMCLK in timer A is set to trigger in up mode, or the high edge, meaning it will count up to the capture and compare value repeatedly. An if statement within the code compares the capture and compare value to the value sent through the UART communication and modifies it accordingly.

5 Getting Started/How to use the device

To interface this device with a computer, the Txd line of the first or only board in the chain must be connected from a computer or other controller via a wire or the integrated USB to serial UART controller built into the G2GET board from Texas Instruments.

NOTE: If using the integrated USB to serial UART controller on the G2GET board, a USB type A to USB type micro B must be connected between the host computer and the G2GET board.

6 Getting Started Software/Firmware

In order to interface with this board, a serial terminal is needed on the host computer. This terminal will be used to send and receive data from user to the MSP430 board. RealTerm was used for testing.

6.1 Hierarchy Chart

Every board is connected to one another through the Rxd and Txd links, forming a long chain. The first board in the chain link is the board connected to the host terminal, typically through the USB interface with no connections to the Rxd pin. The next board, and all boards thereafter, is connected from the previous board's Txd pin to its own Rxd pin. As more boards are interconnected to one another, each addition is farther down the chain. The last board will have no connections to its Txd pin.

6.2 Communicating with the Device

The boards can be connected up to a user terminal through the USB port on the test board, allowing it to link up to a user's PC, or through the UART Rxd and Txd pins on the chip itself. These pins are P1.1 and P1.2 respectively. If using the Rxd and Txd pins, incoming signals are to be connected to Rxd pin 1.1 while outgoing signals get connected to the Txd pin 1.2.

7 Test Setup

The setup of the device starts with plugging the board into a computer in order to load the code. Once the code is debugged, it can be flashed onto the board so it will run when connected to other boards.

Once the code is set up and running, the transmit and receive pins can be connected to transmit and receive pins of other boards in the chain. The grounds should be connected, and the LED properly set up on the breadboard.

Using RealTerm, bytes can then be sent to the board and the LED. In RealTerm, the baudrate should be set to 9600, the display set to HEX[space], and the Half Duplex box checked. In the laptop's Device Manager, the COM port associated with UART can be viewed. This is the port that it should be set to in RealTerm. Pushing the

”Change” button in the Display and Port tabs saves these settings. The figures below show what RealTerm should look loke before bytes are sent.

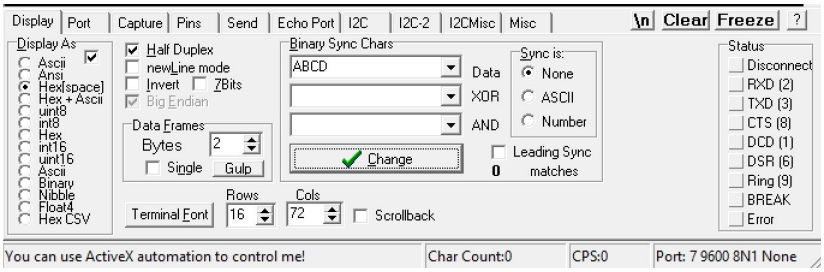

Figure 7: RealTerm Display Tab Setup

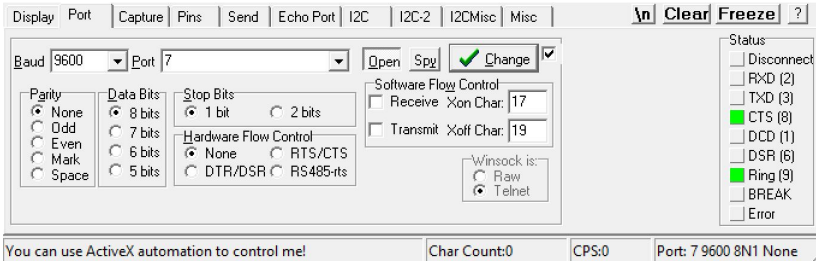

Figure 8: RealTerm Port Tab Setup

7.1 Test Data

Using Realterm to send data, the string ”0x05 0xFF 0xFF 0x00 0x0D” which will turn on the red and green LEDs inside the package, producing a yellow color. Then, the value ”0x05 0x00 0x00 0xFF 0x0D” was sent, turning the LED from yellow to blue. Since this worked in testing, two boards were connected with a wire from the first’s Txd to the second’s Rxd. Then, the string ”0x08 0x00 0xFF 0xFF 0xFF 0x00 0xFF 0x0D” was sent to turn the LED of board 1 to cyan and board 2 to purple. Then the string ”0x08 0x00 0x00 0x00 0x00 0x00 0x00 0x0D” was sent, turning the LEDs off.

8 Design Files

8.1 Schematics

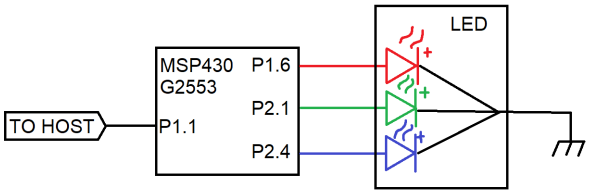

Figure 9: Schematic for a single board setup

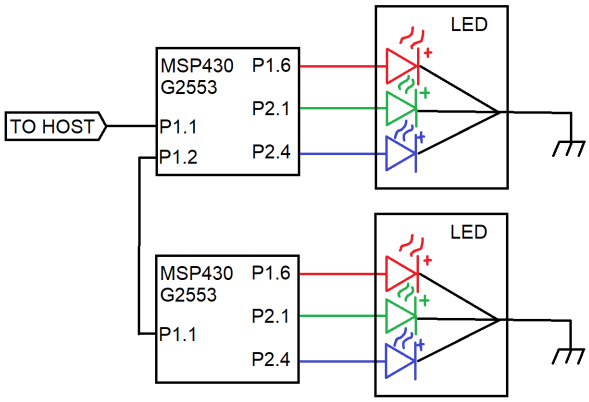

Figure 10: Schematic for a dual board setup

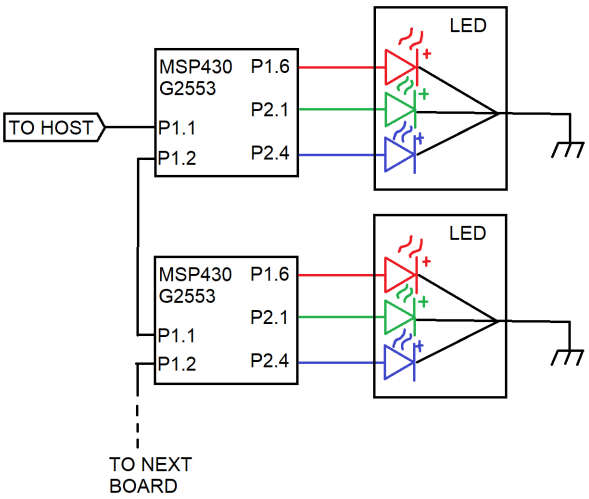

Figure 11: Schematic for a multiple board setup