UART controlled RGB LEDs for MSP430

Jan Garcia
Rowan University

October 22, 2018

1 Design Overview

This application note dives into the world of Emebedded Systems using a MSP430 microcontroller to interface with a serial terminal using UART protocol. The application uses serial communication and transmits signals to the MSP430 and performs hardware PWM to address a RGB LED. The end result leads to a connection of other emebedded processors link together to ultimatley control a node-based system of many LEDs.

1.1 Design Features

The MSP430F5529 microcontroller was chosen to be used for this application note. These are several of the design features demonstrated using the MSP430:

- Adressing Values to an RGB LED
- UART Communication
- Hardware PWM
- USCIA0 ISR

1.2 Featured Applications

- Replicate Stranger Things Ouija Board Lights
- Christmas Lights for Decoration
- LED Lights for Display

1.3 Design Resources

This is just a quick link to the code and design folders you generated during the product. Make sure to actually make the links live.

1.4 Block Diagrams

Fig 1 shows how each node will be configured to initialize timers and use peripherals when starting up and entering low power mode while waiting for a message to be received over UART.

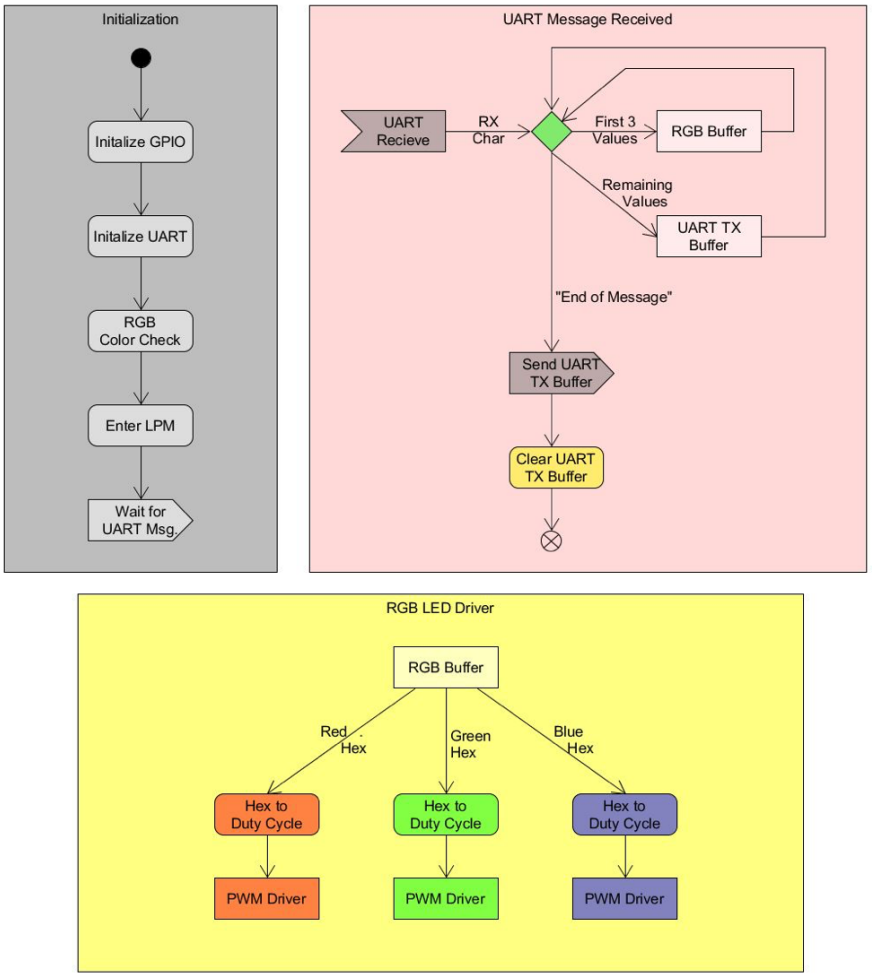

Figure 1: Outline for UART Configuration

Fig 2 shows how the configuration of the network of node will be connected with the purpose of addressing RGB LEDs.

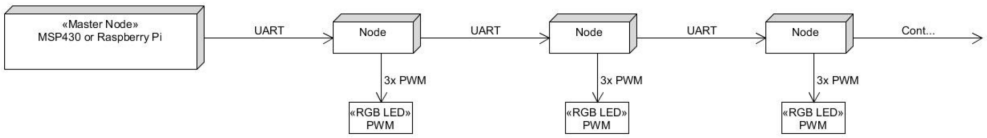

Figure 2: Network of Nodes

1.5 Board Image

Fig 3 shows the setup of the MSP430F5529 pins wired to a breadboard with the common anode RGB LED:

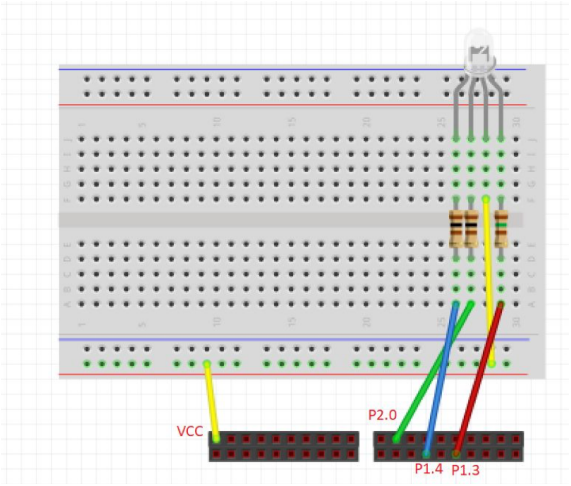

Figure 3: Breadboard Configuration

2 Key System Specifications

The table below shows some of the specifications the system is able to perform and is required for the application:

PARAMETER	SPECIFICATIONS	DETAILS
Microcontroller	MSP430F5529	Needed to be controlled by the timer peripheral .
RGB LED Common Anode	5 V	Voltage for RGB LED Anode

3 System Description

In a scene from Netflixs Stranger Things, Will Byers is stuck in a parallel dimension and can only communicate to his mother by causing Christmas Lights to light up. The mother then strings up a set of lights like a Ouija Board so they can communicate. The objective here is to show a way how this can scene can be recreated using a MSP430 and a RGB LED.

3.1 Detailed Block Diagram

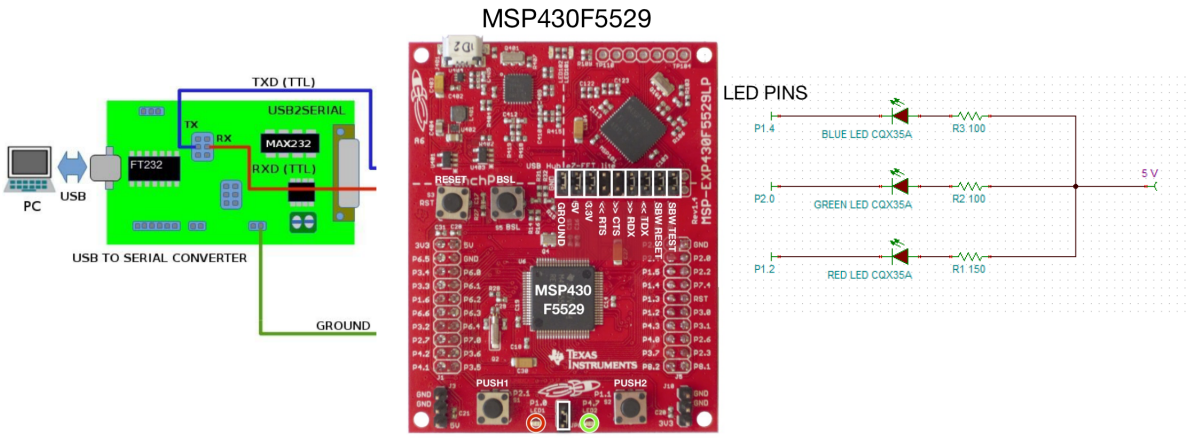

Figure 4: RGB LED with Resistor Values

3.2 Highlighted Devices

- MSP430F5529 - Microcontroller
- RGB LED - Common Anode

3.3 MSP430F5529

The MSP430F5529 processor was chosen to be used for this application note. The F5529 has enough hardware addressable pins from the timer modules for hardware

PWM applications. Additionally, the F5529 is one of the easier of the family of processors for PWM and UART applications. The MSP430 family of ultra-low power microcontrollers consists of several devices featuring peripheral sets targeted for a variety of applications. The architecture, combined with extensive low-power modes, is optimized to achieve extended battery life in portable measurement applications. The microcontroller features a powerful 16-bit RISC CPU, 16-bit registers, and constant generators that contribute to maximum code efficiency.

3.4 RGB LED

The RGB LED used in this application is a common anode. The anode of the LED is connected to VCC (+5V) and a current limiting resistor is connected to each individual LED. The forward current for all colors (red, green, blue) is 20mA.

The red LED current limiting resistor can be calculated by the following:
$R_{lim} = \frac{V_i - V_f}{I_f} = \frac{5V-2V}{20mA} = 150\Omega$
The green LED current limiting resistor can be calculated by the following:
$R_{lim} = \frac{V_i - V_f}{I_f} = \frac{5V-3.2V}{20mA} \approx 100\ \Omega$
The blue LED current limiting resistor can be calculated by the following:
$R_{lim} = \frac{V_i - V_f}{I_f} = \frac{5V-3.2V}{20mA} \approx 100\ \Omega$

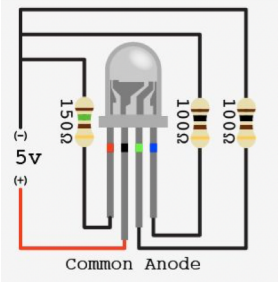

Figure 5: RGB LED with Resistor Values

4 SYSTEM DESIGN THEORY

4.1 Hardware PWM

The MSP430F5529 was configure to hardware PWM (Pulse Width Modulation) and had to be used to control the duty cycle of the LED utilizing the Timer A peripheral from the microcontroller. The PWM purpose was to vary the different color of combinations and control the brightness of the LED by determining the duty cycle percentage. All of the LEDs initially will begin at a 0 percent duty cycle or will be fully OFF, but essentially, the LED will be able to vary between 0 to 100 percent duty cycle. The PWM sets the duty cycle by pulsing the output of the microcontroller for a certain amount of time, hence the name pulse width modulation. Therefore, the microcontroller will

be activating the clock peripheral as a hardware element and will let the pulsing be controlled by the clock as hardware PWM.

The microcontroller's hardware PWM is done by initializing the timer modules to output to a timer output pin directly. In this application, the timer was set in UP mode and OUTPUT MODE 7: RESET/SET. When the timer is in UP mode, the timer will begin to increment until it reaches the values set by the capture compare register. This results in determining the period of the signal by the capture compare register. The capture compare register was set to 255 as this helped to map the values for the RGB to a duty cycle and create a 1kHZ signal. As the capture compare register periodically reaches its values, the output will perform the RESET/SET, then reset the timer and begin to increment again until the capture compare register value is reached. The PWM process can be view down below in Fig 6.

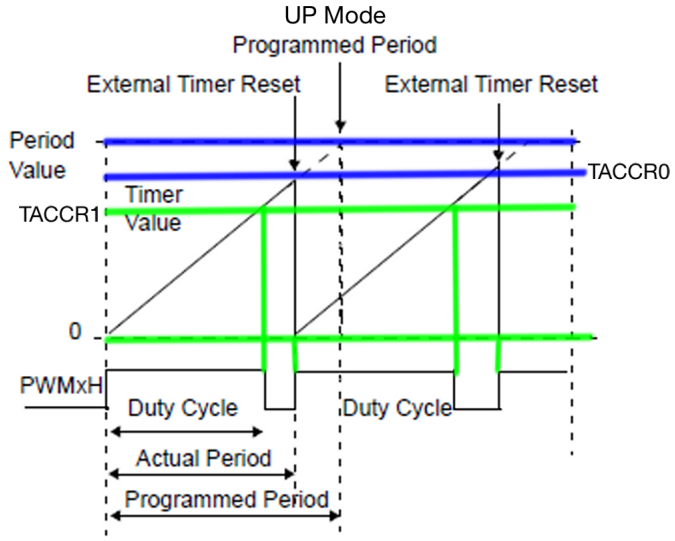

Figure 6: Hardware PWM Diagram

4.2 Communication Between Nodes

In this application, UART communication will be used to receive the data to each node. The UART transmission speed is set at 9600 baud rate. For testing, the master terminal will be sending each node multiple bytes of data and at maximum it can be expected to be a total of 79 bytes. Each node should be expected to take the data from the master node, perform the required action to the LEDs, then repackage the data to the next node. In other words, the first byte indicates how many bytes there are total (including itself). The next three bytes are values to control the Red Green

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	0x00 (carriage return)

Blue LEDs respectively. The bytes after should be repackaged to be sent to the next node. The following first table below explains what each node should receiving in the format of bytes.

5 Getting Started/How to use the device

5.1 Require Items

Before explaining how to configure the device, there are several items that are needed for the application to work.

- MSP430F5529
- Breadboard
- Resistor values: 1 x 150Ω , 2 x 100Ω
- RGB LED - Common Anode
- Jumpers
- USB to UART TTL Cable

5.2 How to Setup the Device for the Application

Fig 4 below shows the entire setup for the application of the UART communication between the MSP430 processor and the PC. The Receive, Transmit, and Ground lines of the processor are connected to the USB-to-Serial converter and the data is converted to be read by software in UART protocol. On the MSP430 board, the UCA0RXD can be connected to pin P3.4, the receiver (green) and the UCA0TXD can be connected to pin P3.4, the transmitter (white). The RGB LED is a common anode component and should be configured in the following manner. The red LED should have its own current limiting resistor of 150Ω, and the green and blue LED should have their own current limiting resistors of 100Ω. The common anode should be connected to VCC (+5V). On the board from the MSP430, the pin of the green LED is connected to pin P2.0, the pin of the blue LED is connected to pin P1.4, and the red LED pin is connected to the pin 1.3.

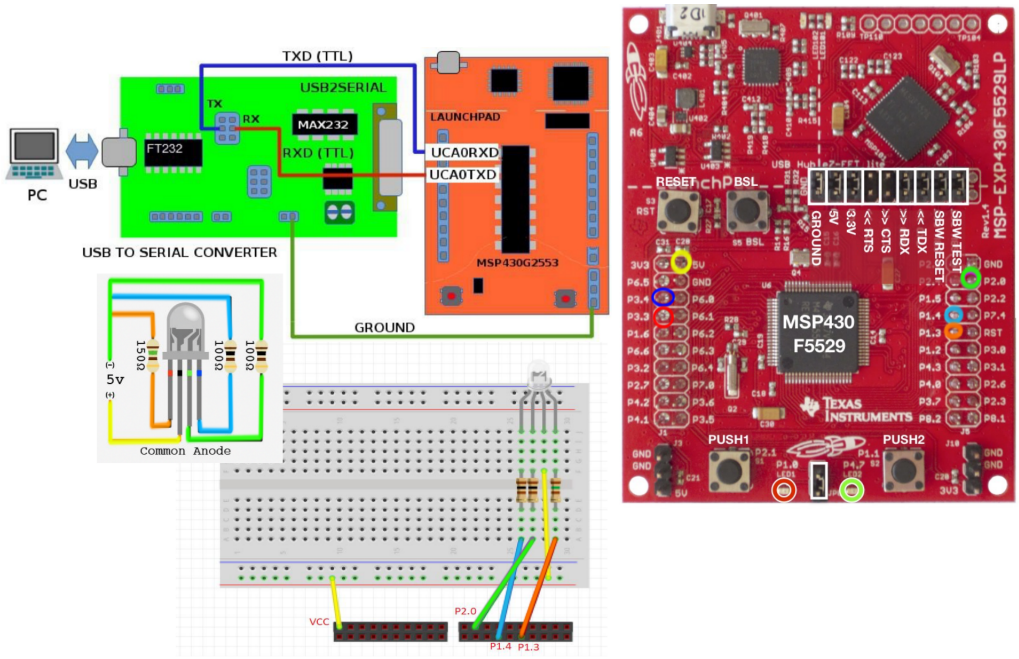

Figure 7: MSP430F5529 Connected with UART and RGB LED

6 Getting Started Software/Firmware

6.1 Require Software

Now that the device has been setup, there are two important softwares that are needed for the application to work.

- Code Composer (Latest Version)
- Realterm

6.2 How to Setup the Software for the Device

Open Code Composer and create a project with the following c. file that runs the program for the application. At this time, the MSP430F5529 needs to be connected to the computer for the program to run on the microcontroller. Debug the c. file and run the program to configure MSP430 with the application.

Now that the MSP430 is connected and has been configure with the code, the microcontroller can then be serial communicated using the software Realterm as a terminal.

Open Realterm and ensure the following configurations are set:

- Display: Display as Hex [space]
- Display: Half-Duplex On
- Port: Baud 9600
- Port: Prolific USB-to-Serial Converter

7 Test Setup

Now communicate with the microcontroller by inputting a message of hexadecimal values in the Realterm terminal. Bytes 1 - 3 correspond to the RGB values which are to be set for the node. For example, 0x03 0x00 0xFF 0x00 (0, 255, 0) will turn the LED fully bright green and 0x03 0x7E 0x00 0xFF (126, 0, 255) will turn the LED to purple. Because Half-Duplex is enabled, it can be seen what was transmitted in green and what was received from the MSP430F5529 in yellow.

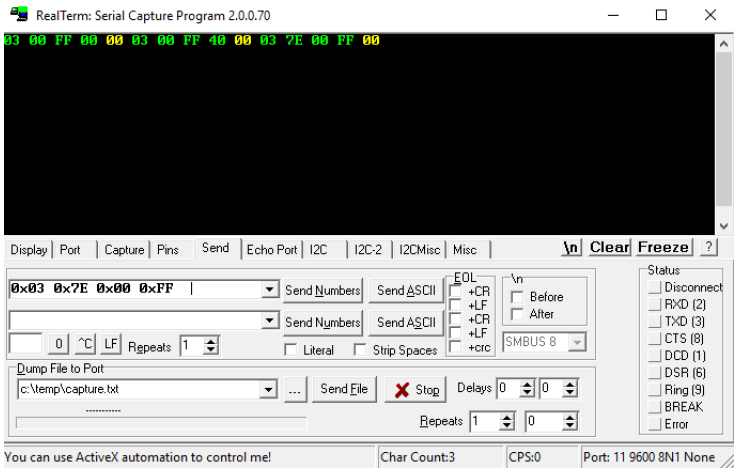

Figure 8: Realterm

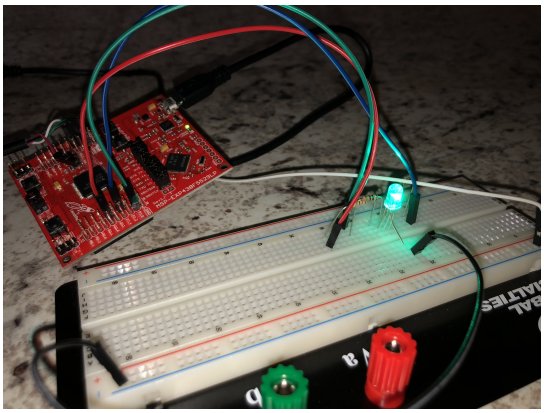

Figure 9: Green LED

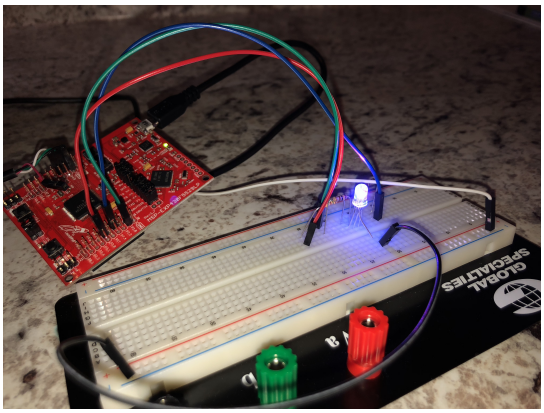

Figure 10: Purple LED

8 Design Files

8.1 Schematics

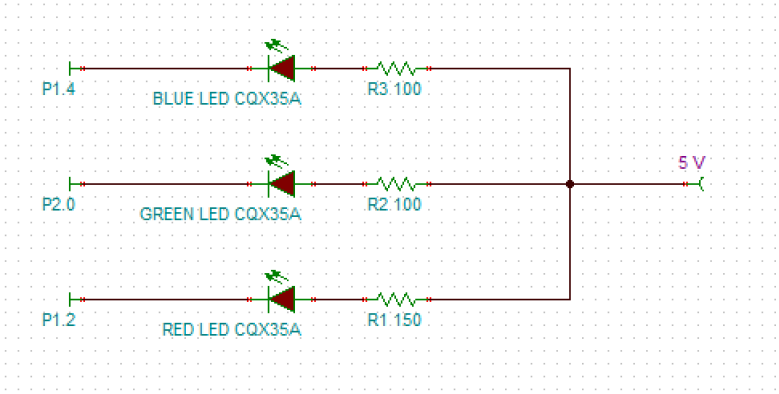

Figure 11: RGB LED with Resistor Values

8.2 Bill of Materials

Part Name	Quantity	Price USD
MSP430F5529	1	12.99
RGB LED	1	0.99
Breadboard	1	5.99
150 ohm resistor, 1/2 watt, 5 % tolerance	1	0.05
100 ohm resistor, 1/2 watt, 5 % tolerance	2	0.05
Total		20.12