Milestone: Stranger Things

Patrick Wilk
Rowan University

October 21, 2018

1 Design Overview

For the stranger things milestone a Ti-MSP430F5529 microcontroller board was used to control several features on a RGB LED. PWM and UART were used to vary brightness and color of the LED. The leads of the LED were connected to three ports of the controller. The MSP430F5529 received bytes through the UART at 9600 BAUD, took the first four and transmitted the excess bytes. The UART packet is sent to the first node in bytes, either sent straight from the serial source or from previous nodes.The first byte was the number of expected bytes also called the packet length. The microcontroller takes these bytes and transmits the packet length minus three. The other three bytes are what controlled the PWM or brightness of the RGB LED.The number is rated between 0 and 255 for a 0% to 100% intensity. Once, the instruction is completed it resets and expects the next byte to be packet length once again,

1.1 Design Features

These are the design features:

- MSP430F5529
- UART serial connection operating at 9600 BAUD
- PWM controlling RGB LED
- LED operating at 3.3V from pins

1.2 Featured Applications

- PWM REG LED
- State Machine/switch statement UART

- UART 9600 bits/second
- Common anode LED circuit

1.3 Design Resources

- Datasheet: http://www.ti.com/product/MSP430F5529
- Github link: https://github.com/RU09342-F18/milestone-1-team6

1.4 Block Diagram

Figure 1 shows the pin assignment from the board to the RGB LED. Betweent the pins and the LED three 100 ohm resistors were used. The common anode of the LED was connected to 3.3V through the board.

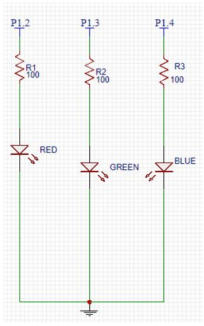

Figure 1: Pin and LED connection

1.5 Board Image

The simple connections from the pins to the RGB LED allowed the micro-processor to modulate the color and brightness of the LED using PWM.

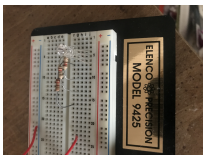

Figure 2: Pin and LED connection

PARAMETER	SPECIFICATIONS	DETAILS
SMCLCK Frequency	1 MHz	PWM timer speed
Baud Rate	9600	UART Communication speed
TA0CCR0	256	PWM period
TA0CCR1	P1.2	On time for red LED
TA0CCR2	P1.3	On time for green LED
TA0CCR3	P1.4	On time for blue LED

2 Key System Specifications

3 System Description

The system consists of three main components: the microprocessor, off board circuit and the RGB LED. The circuit was simple and only three resistors were used to regulate voltage. The common anode on the LED was connected directly to 3.3V. The microprocessor handled receiving and transmitting bytes to control the PWM of the LED's RGB pins. Finally, the LED was controlled and gave an output from the system.

3.1 Detailed Block Diagram

Reference Figure 1 for configuration of setup.

3.2 Highlighted Devices

- MSP430F5529- controls PWM signals for the RGB LED
- 100 ohm resistor- used in series with the diode
- RGB- common anode diode

3.3 Device/IC 1

MSP430F5529 board was used for Milestone 1. The microprocessor has 7 capture/compare registers for Timer A. Register 0 was set to be the max period, while 1-3 were used to change the PWM pulses for red, green, and blue.

3.4 Device/IC 2

The RGB LED was ran using 3.3 volts, straight from the micro-controller. The resistors acted as current limiting devices and then powers the individual LED colors. The longest pin on the LED is the common anode and directly connected to voltage.

4 SYSTEM DESIGN THEORY

4.1 Design Requirement 1

Part of the design requirements was to pulse the red, green, and blue pins of the RGB LED with the MSP430F5529. This was solved by using individual functions, Timer_Setup was the first function configured. Timer A0 was configured to use SMCLK and Up mode using the line TA0CTL = TASSEL_2 + MC_1 + ID_0 + TACLR. Three capture compare registers, CCR1, CCR2, and CCR3 were set to use OUTMOD3, which is Set/Reset mode. A value was set when the timer reached the value in the CCR register and reset when it reached the value in CCR0. This also included the duty cycle functionality with CCR0 being set to 256 or maximum duty cycle. CCR1, CCR2, and CCR3 were all set to 0 or OFF initially, with the registers being used for the duty cycles of the Red, Green, and Blue nodes respectively. CCR1 was set to P1.2, CCR2 to P1.3, and CCR3 to P1.4.

4.2 Design Requirement 2

The led setup was configured to the output pins used to drive each node on the LED. P1.2, P1.3, and P1.4 were all set to the output direction. P1.2 was connected to the Red node, P1.3 was connected to Green, and P1.4 was connected to Blue. The primary peripheral functionality set on each pin enabling Timer A on each pin. The circuit connecting the micro-controller and the LED was simple. Each pin was connected to a resistor of 100 ohms and drove the RGB LED depending on the input from UART.

5 Getting Started/How to use the device

The code needs to be uploaded to the micro-controller before use. Next the hardware needs to wired accordingly to Figure 1 with the common anode connected to 3.3V. Once everything is setup, one can finally write to the device and order the LED to output what is wanted. Realterm is a software used to control the LED. In Realterm the port and baud rate must be correctly chosen. Realterm allows to write bytes through the software to the MSP. The first byte is the length bit, and the remaining three determine the red, green and blue light.

In order to link multiple boards or daisy chain, the Rx and Tx pins must be used. The Rx pin reads the data and the Tx transmits it. The preceding board must be connected to Tx and Rx to the board after. This allows for data to be used by the first board, changed and then sent to the next board.

6 Getting Started Software/Firmware

Two variables and three functions were used in the code. The two variables used were 'byte', which was initialized to zero, and 'total', which kept track of the total bytes.

Three functions were used to setup the timer, LED, and UART. Timer A was used for the PWM, the output pins for the LED nodes, and UART RX and TX for receiving and sending bytes. The UART setup communication to the LED. The UART TX and RX pins were set using the line P4SEL —= BIT4 + BIT5. Other software configuration included resetting and initializing the state machine, selecting clock for UART, and setting the BAUD Rate to 9600. Setting the UCA1BR0 register to 104 and the UCA1BR1 register to 0 configured the BAUD rate. A switch statement with a series of cases was used. Case 0 set the total byte integer to the value received.Case 1, case 2, and case 3 set the duty cycles for the Red, Green, and Blue LEDs. The first three bytes in the initial package that was received demands the LED's function. Case three also removes three bytes that were used. This ensures that the package passed on will display the correct color and brightness.

To make sure that the boards work together when daisy chained, the correct port must be chosen on the computer and the correct BAUD rate so that the devices can communicate with each other. All microprocessors must share a common ground.

6.1 Hierarchy Chart

The flow chart shows how MSP boards can be linked together or daisy chained. When linking the boards, the connection enters the Rx pin and exits the Tx pin. This allows the board to read and use the data, then transmit the rest of the package. The computer initializes the first package sent.

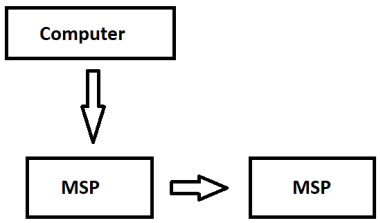

Figure 3: Flow chart

6.2 Communicating with the Device

The MSP430F5529 uses pins RXB and TXB to communicate. The jumper must be removed to use these pins, since they are usually jumped together so the USB can

communicate with the microprocessor. RealTerm was a program used to send bytes to the microprocessor. The BAUD rates must match (9600) and the correct port where the board is connected must be selected. To connect another board the RXB must be connected to the TXB pin of another board and the TXB pin to a RXB of the next board.

7 Test Setup

7.1 Test Data

In Figure 4 a table of the bytes shows the effects on the LED. The sequence of bytes from the computer set the first LED to red, second to green, and third to blue. Two colors can be initialized to create different colors such as magenta.

Byte Number	Content	Meaning
Byte 0	0x08	8 total bytes in the package
Byte 1	0x7E	Red (current node): 50% duty cycle
Byte 2	0x00	Green (current node): 0% duty cycle
Byte 3	0xFF	Blue (current node): 100% duty cycle
Byte 4	0x40	Red (next node): 25% duty cycle
Byte 5	0xFF	Green (next node): 100% duty cycle
Byte 6	0x00	Blue (next node): 0% duty cycle
Byte 7	0x0D	End of Message Check byte

Figure 4: Hex table