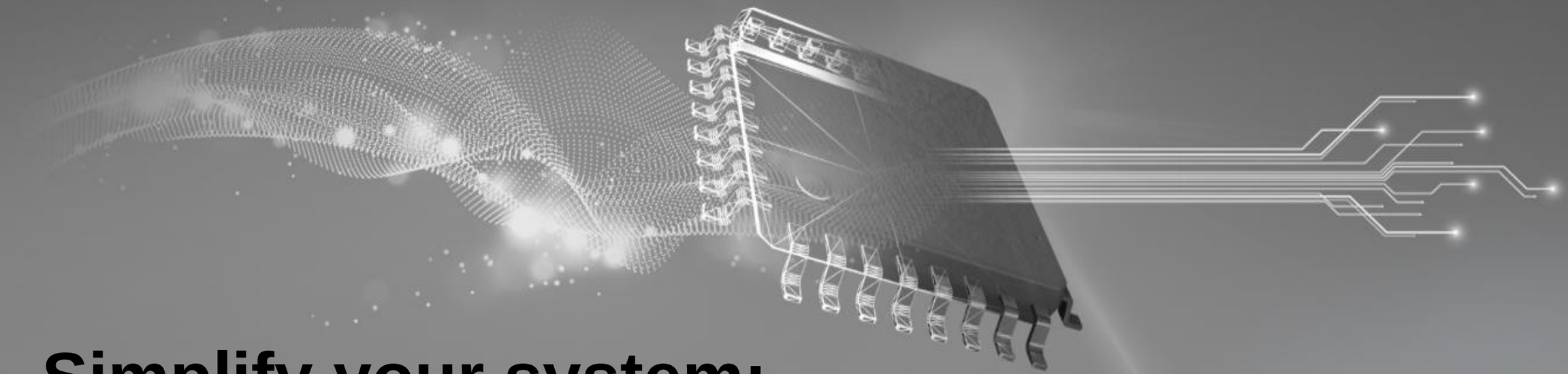
TI TECH DAYS



Simplify your system:
How to offload multiple functions to an MSP430™ MCU

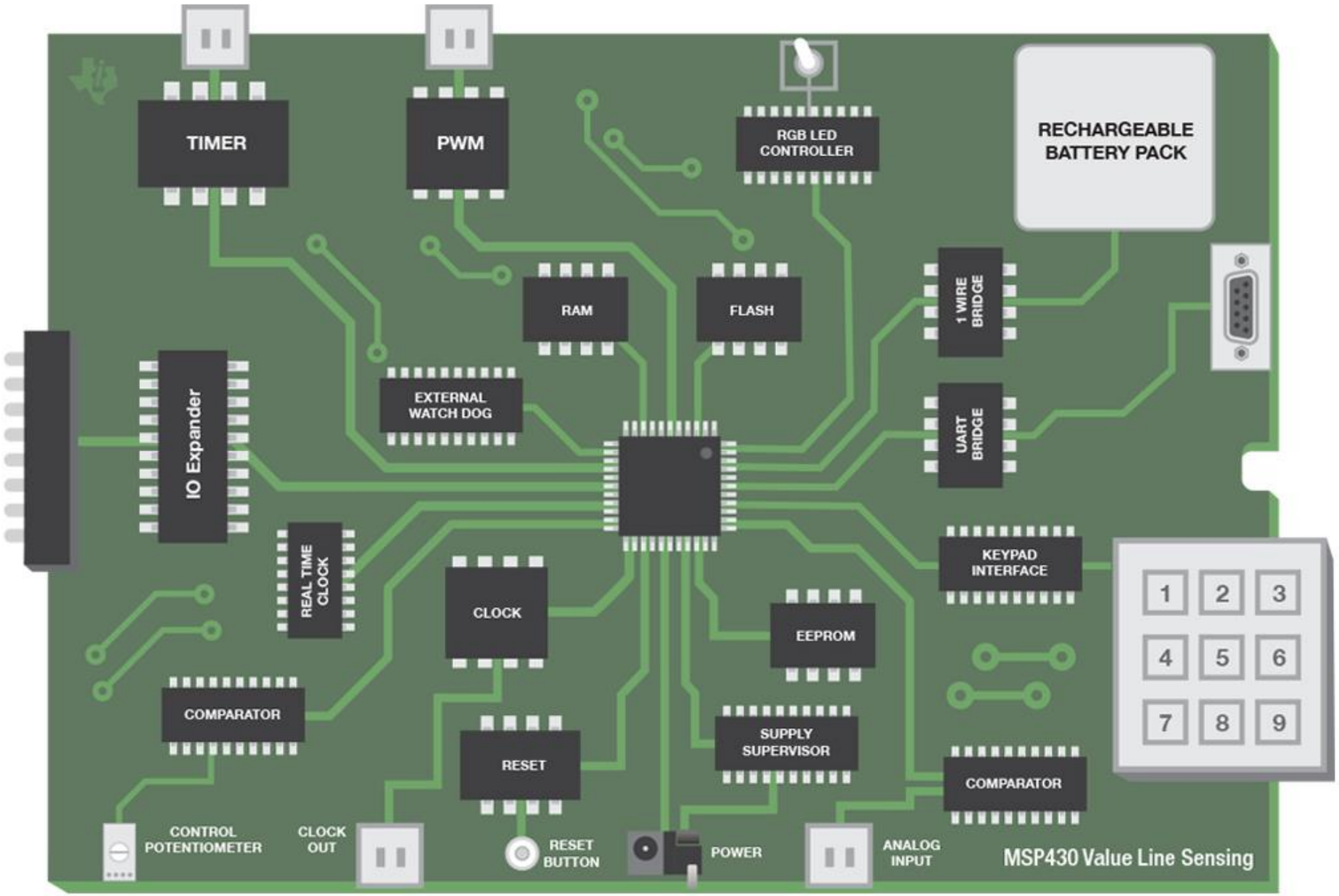
Presenter: JD Crutchfield – MSP430 Applications

Texas Instruments

Agenda

- A system level approach to microcontroller selection
  - Adding a secondary MCU to your design
  - Example design scenario
- MSP430 product offering
- Live demonstration of MSP430 software examples & TI Cloud Tools

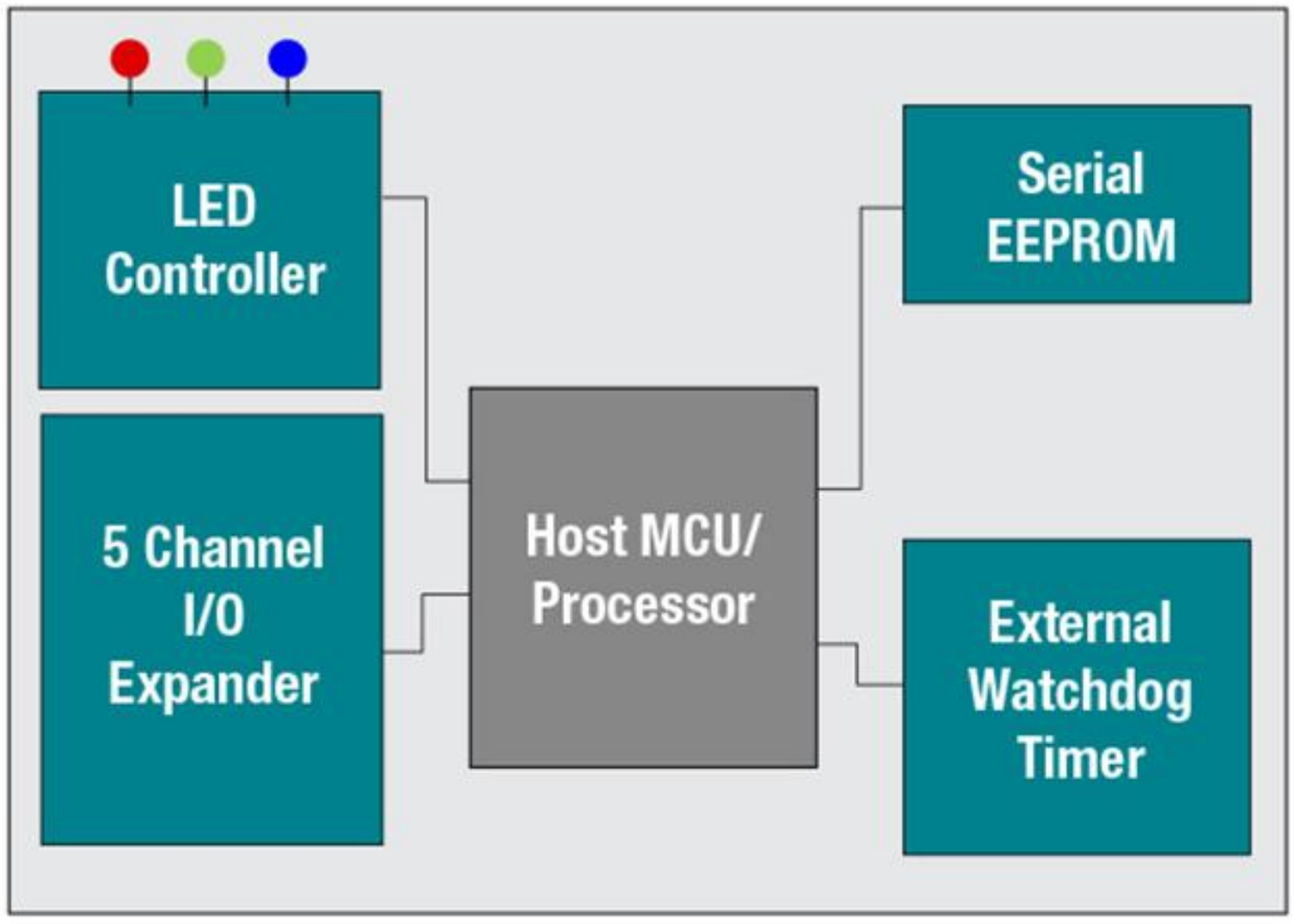
A system level approach to microcontroller selection



- Systems are getting more complicated
- MCUs are becoming lower cost, easier to use
- Customers are starting to integrate discrete components into software

Starting a new system design

A discrete component approach

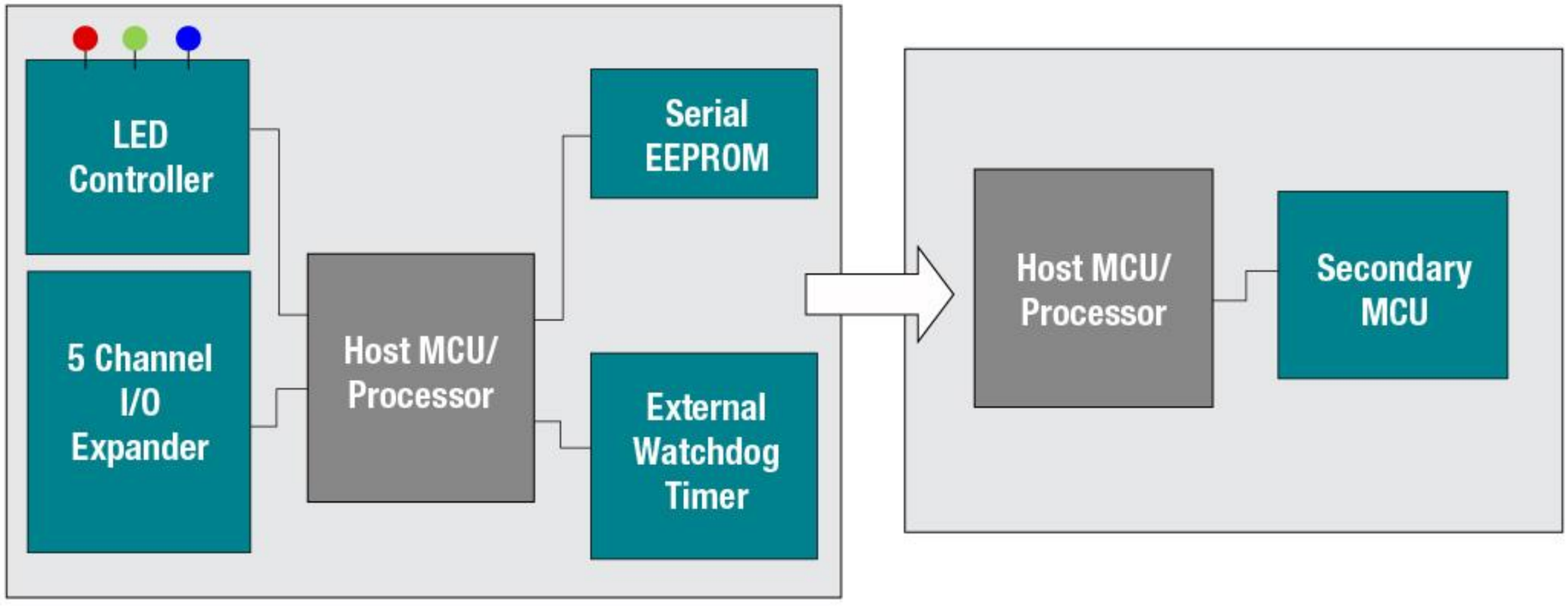


Total system bill-of-materials (BOM):

- LED Controller
- I/O Expander IC
- EEPROM IC
- External Watchdog Timer
- Microcontroller

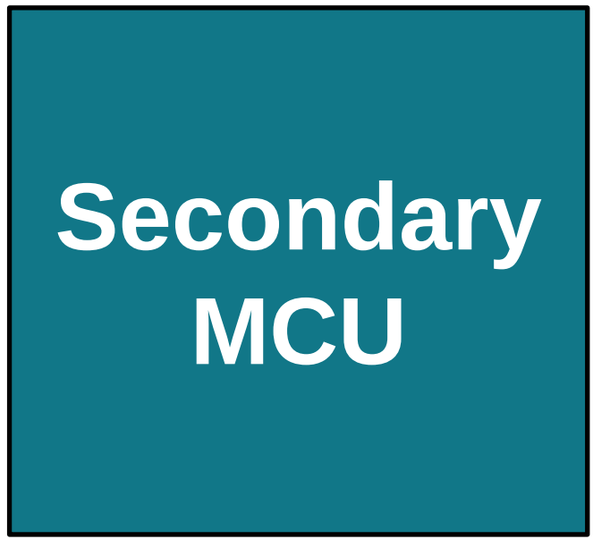
Starting a new system design

An integrated approach



Integrate your discrete system level functions into software on your microcontroller!

Implementing discrete component functionality in a secondary MCU



Memory Footprint

- LED Controller (x3 LEDs): **0.9kB**
- I/O Expander via SPI: **0.36kB**
- EEPROM Emulation: **0.33kB + 4kB of EEPROM**
- External Watchdog Timer: **0.37kB**

Total Memory Footprint: ~5.96kB

Implementing discrete component functionality in a secondary MCU

Secondary MCU

- ✓Save BOM costs
- ✓Save board space
- ✓Simplify design

Cost Breakdown (web pricing)

- LED Controller IC (3 channels): **~ $0.20**
- 5 channel I/O Expander IC: **~$0.25**
- Serial EEPROM (4kB) : **~$0.20**
- External Watchdog Timer: **~$0.31**

Total Cost for all 4 chips: ~ $0.96

Cost for an 8kB MSP430 device: $0.32

MSP430 MCUs for housekeeping functions

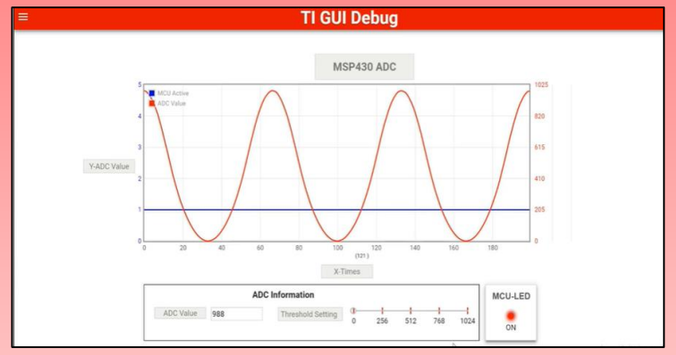
Low-cost MCU offering to help customers combine multiple functions/interface to other parts around the board.
Flexible offering based on memory needs.

Memory	Primary Feature	Part Number	1ku Price
2KB		MSP430FR2110	$0.24
4KB		MSP430FR2111	$0.30
8KB		MSP430FR2422	$0.32
8KB	CapTIvate	MSP430FR2512	$0.52
16KB		MSP430FR2433	$0.50
32KB		MSP430FR2155	$0.62
64KB		MSP430FR2476	$0.83

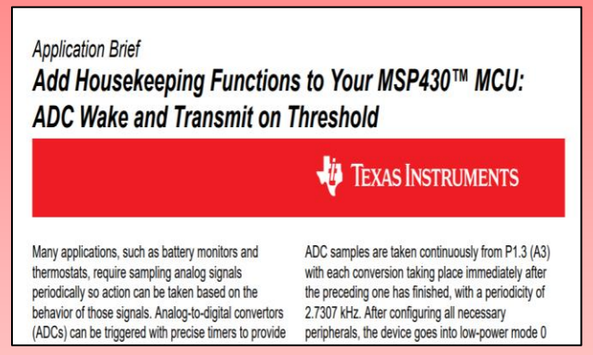
MSP Housekeeping MCUs training

Implement simple functions in your system quickly with a low-cost microcontroller!

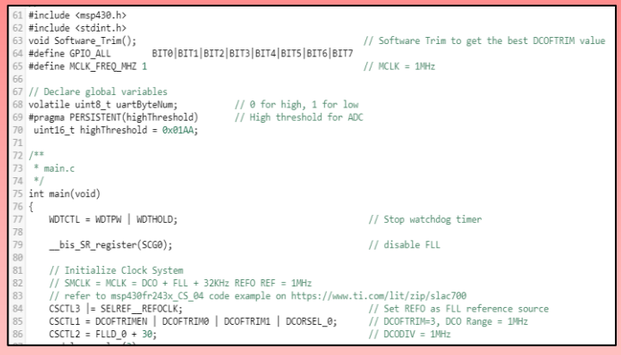
Training videos
(available in both English & Chinese)



Evaluation GUIs



Tech Notes



```
#include <msp430.h>
#include <stdint.h>
void Software_Trim();                                   // Software Trim to get the best DCOFTRIM value
#define GPIO_ALL        BIT0|BIT1|BIT2|BIT3|BIT4|BIT5|BIT6|BIT7
#define MCLK_FREQ_MHZ 1                                 // MCLK = 1MHz

// Declare global variables
volatile uint8_t uartByteNum;           // 0 for high, 1 for low
#pragma PERSISTENT(highThreshold)       // High threshold for ADC
 uint16_t highThreshold = 0x01AA;

/**
 * main.c
 */
int main(void)
{
    WDTCTL = WDTPW | WDTHOLD;                           // Stop watchdog timer

    __bis_SR_register(SCG0);                            // disable FLL

    // Initialize Clock System
    // SMCLK = MCLK = DCO + FLL + 32KHz REFO REF = 1MHz
    // refer to msp430fr243x_CS_04 code example on https://www.ti.com/lit/zip/slac700
    CSCTL3 |= SELREF__REFOCLK;                          // Set REFO as FLL reference source
    CSCTL1 = DCOFTRIMEN | DCOFTRIM0 | DCOFTRIM1 | DCORSEL_0;    // DCOFTRIM=3, DCO Range = 1MHz
    CSCTL2 = FLLD_0 + 30;                               // DCODIV = 1MHz
```

Software

Learn → Evaluate → Implement

All you need is an [MSP430FR2433 Launchpad]() to get started today!

MSP430 housekeeping example: UART-controlled RGB LED controller

Includes live demo

UART RGB LEDG color mixing application brief

Application Brief

Add Housekeeping Functions to Your MSP430™ MCU: UART RGB LED Color Mixing

One common MSP430 application is to mix colors in an RGB LED using Pulse-width Modulation (PWM) outputs from the Timer module. An RGB LED is an LED with red, green, and blue LEDs in one package sharing a common anode or cathode. RGB LEDs are commonly used in many diverse applications including stage lighting, indoor lighting, outdoor lighting, and home decorations. Each individual LED's current is controlled by using a current-limiting resistor and a PWM waveform with a variable duty cycle. By individually controlling the PWM duty cycle of the red, green, and blue LEDs inside the RGB LED, a wide array of color hues can be observed that range the scale of the visible light spectrum. Typically, RGB colors range in value from 0-255 (decimal). Protocols such as universal asynchronous receiver/transmitter (UART) can communicate to the MCU what color is to be outputted to the RGB LED by sending a 24-bit hex value that contains the values for all colors or individually sending the byte value to each color. In addition, interrupts can be used to update the PWM duty cycles via UART while keeping the MCU in a low-power state.

Note

This example can be used with any MSP430 LaunchPad Development Kit with the required MCU peripherals. For migrating pinouts and peripherals, see the device-specific data sheet.

Implementation

This implementation uses a UART to send 8-bit values (0-255 decimal) to the red, green, and blue LEDs. The firmware takes those values, converts it into a proportional PWM duty cycle waveform, and outputs the signal to their respective RGB pins of the LED. A graphical user interface (GUI) was developed to select RGB color hues individually or simultaneously using a slider, number input, or a color wheel palette.

Timer_A0 and Timer_A1 are configured to output varying duty cycle PWM waveforms on P1.2 (red), P1.4 (green), and P1.5 (blue). P1.2 is tied to TA0.2 by selecting the secondary I/O function for that pin, and TA0.2 corresponds to Timer_A0 and its Compare/Capture Registers 0 and 2. Therefore, the compare/capture registers for Timer_A0 are configured so that the duty cycle is adjustable by adjusting TA0CCR2, with the duty cycle equal to TA0CCR2/TA0CCR0. A similar process is used for configuring P1.4 (TA1.2) and P1.5 (TA1.1) using Timer_A1 and its Compare/Capture Registers 0, 1, and 2.

Figure 1 shows the block diagram for this implementation.

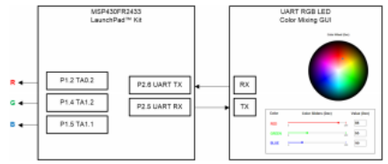


Figure 1. Implementation Overview

The MSP430FR2433 LaunchPad™ Development Kit is used with this example project, but it can be used for any MSP430 microcontroller with proper code migration. Backchannel UART interface on eZ-FET of the LaunchPad kit is used for UART communication with the GUI, however, the UART pins must connect to P2.5 and P2.6 rather than its intended configuration (P1.5 and P1.4) since those pins are being used respectively for TA1.1 and TA1.2. To do this, remove the RXD and TXD jumpers and use female-to-female jumpers to connect the top pins of RXD and TXD (on the eZ-FET Debug Probe side) to P2.5 and P2.6, respectively. These pins correspond with UCA1 (instance 1 of the eUSCI_A module) rather than UCA0 (used in other Housekeeping examples), so the firmware has been updated to configure the UART properly.

The COM channel number information can be found in the PC device management under the control panel. Figure 2 shows the MSP430FR2433 LaunchPad kit including eZ-FET, UART into P2.5 (RXD) and P2.6 (TXD), and RGB PWM outputs on P1.2 (TA0.2), P1.4 (TA1.2), and P1.5 (TA1.1). In this example, a common-cathode RGB LED with current-limiting series resistors was used, as shown on the right hand side. Use female-to-female jumpers to connect the series resistors and RGB as shown in the schematic and connect the common cathode to the GND pin of the Launchpad. It may be helpful to use a breadboard if necessary. Ensure resistor values are selected based on the current ratings and forward voltage drops of the individual LEDs.

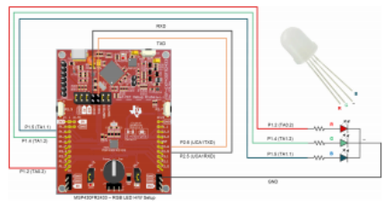


Figure 2. MSP430FR2433 LaunchPad and Connections

As shown in Figure 3, a GUI is used to set the RGB color values to output to the LED using individual color sliders or input boxes, or simultaneously using a clickable color wheel palette. To use the sliders, simply drag the slider to the desired number from 0-255 (decimal) of the color value. To use the input box, type in the desired number from 0-255 (decimal) of the color value. To use the color wheel, click or drag the marker to the desired color. This will output that color in real-time to the RGB LED.

The eUSCI_A1 peripheral was used in UART mode to enable commands to be received on P2.5/UCA1RXD and transmitted on P2.6/UCA1TXD. The eZ-FET inside of the LaunchPad was used for evaluation. A baud rate of 9600 must be selected with one stop bit and no parity. If the GUI is enabled in software, the PC sends a byte of data to the MCU via UART that contains the individual color value in hexadecimal. If the color wheel is used, the PC will send 3 bytes of data containing the RGB values in hex.

If not using the GUI, then the MCU will output the RGB-mixed color after every 3 bytes of data are transmitted from the PC. For instance, if the PC sends the data 0xFF, 0x00, and 0x00, then the RGB LED will output red because the first byte is the red value, the second byte is the green value, and the third byte is the blue value.

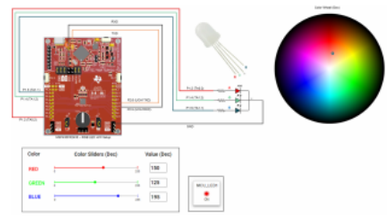


Figure 3. RGB LED Color Mixing GUI

Performance

The operation of the demo can be run as described in the implementation section regarding the use of UART to configure the PWM duty cycle of the red, green, and blue LEDs inside of the RGB LED. P1.2 (TA0.2), P1.4 (TA1.2), and P1.5 (TA1.1) are used to output controllable PWM waveforms to the RGB LED in order to create any hue of color desired by the user. Figure 3 shows the GUI interface with selectable color sliders, input boxes, and a color wheel as options for the user to determine what intensities of red, green, and blue are to be mixed and outputted. When the color sliders, input boxes or color wheel values are changed, the values will update automatically for the other and the RGB LED will output that color mix immediately.

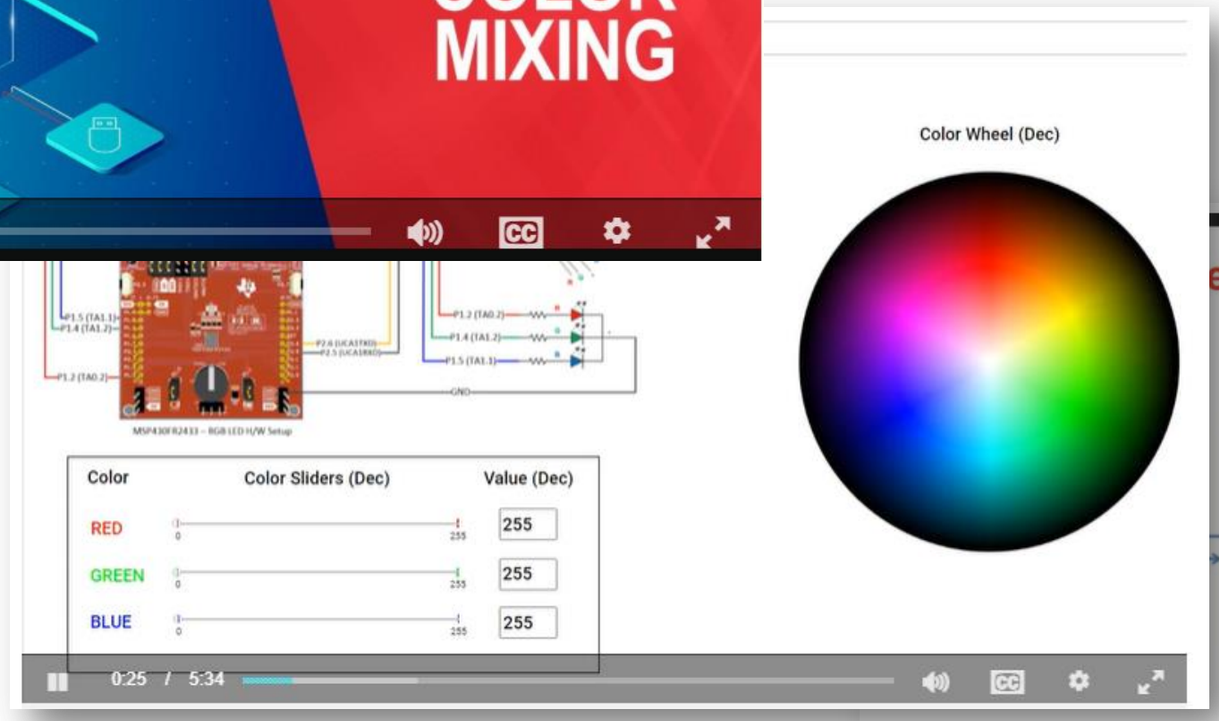
To Get Started

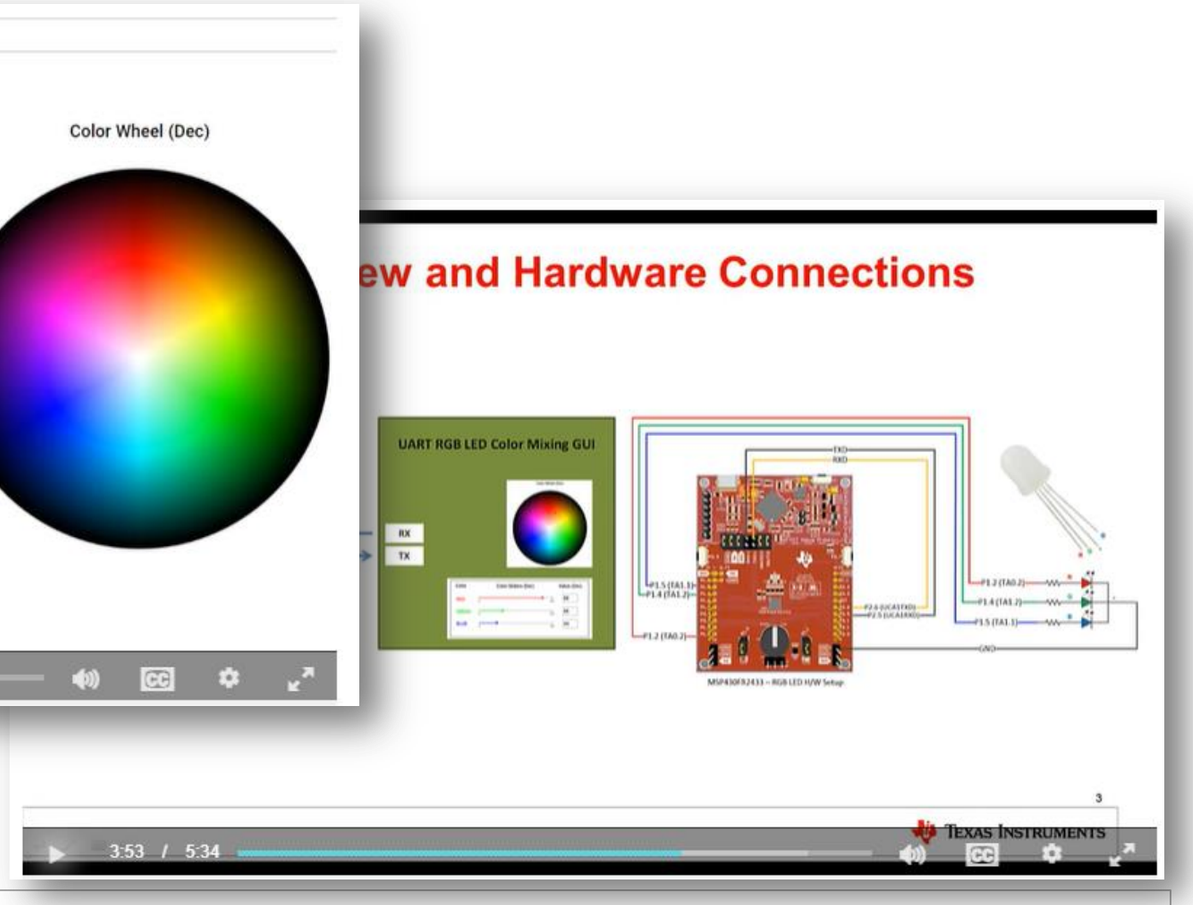
1. Watch the training video "UART RGB Color Mixing with a Housekeeping MCU" to learn how to use the GUI to mix colors on an RGB LED.
2. Order a MSP430FR2433 LaunchPad kit to evaluate the UART RGB Color Mixing example code.
3. Download and test this example with the UART RGB Color Mixing example GUI to mix any combination of red, green, and blue colors in a RGB LED.
4. Evaluate the UART RGB Color Mixing example code for the MSP430FR2433 LaunchPad kit.

Device Recommendations

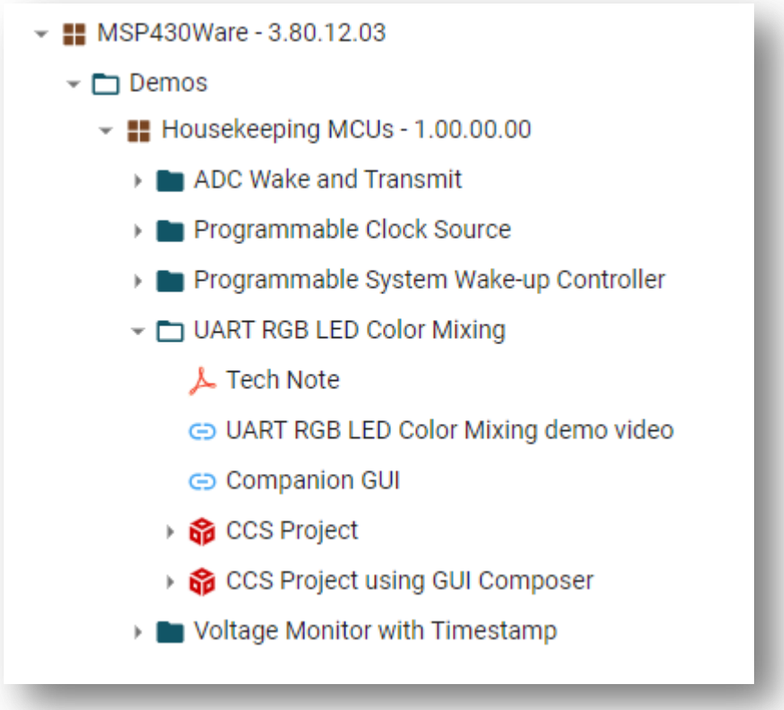
Part Number	Key Features
MSP430FR2433	16KB FRAM, 4KB SRAM, 10-bit ADC, UART/SPI/I2C, Timer
MSP430FR2422	8KB FRAM, 2KB SRAM, 10-bit ADC, UART/SPI/I2C, Timer

UART RGB LEDG color mixing demo video





UART RGB LEDG color mixing example code



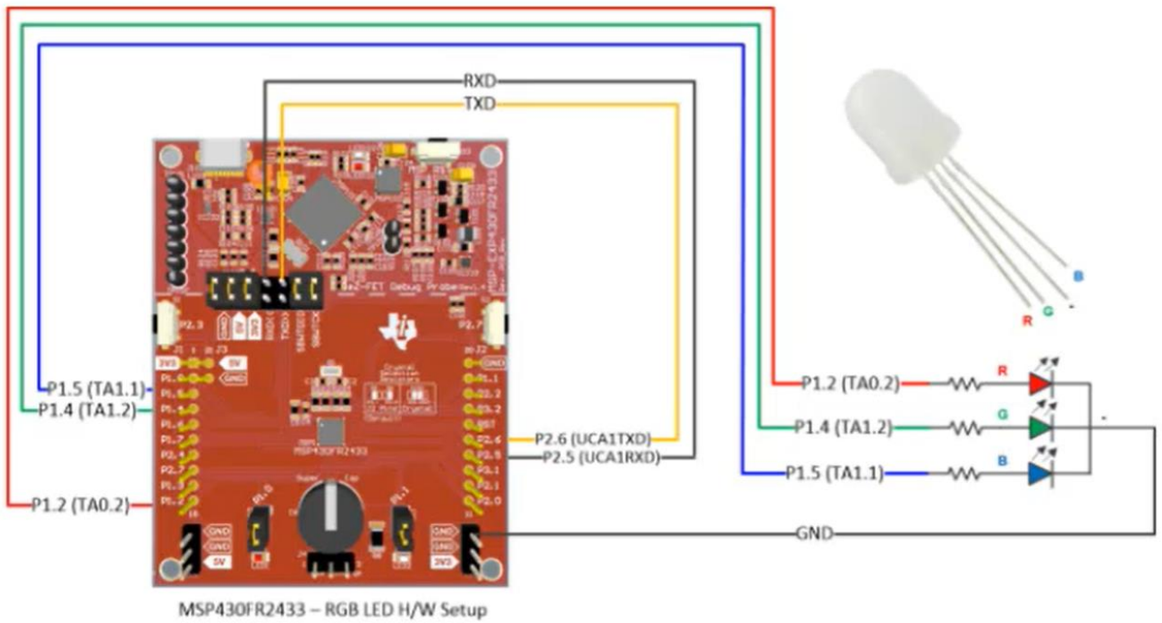
/ Software / MSP430Ware (3.80.12.03) / Demos / Housekeeping MCUs (1.00.00.00) / UART RGB LED Color Mixing / CCS Project / **main.c**

```
 * OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
 * WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
 * OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
 * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 * --/COPYRIGHT--*/
//******************************************************************************
//  MSP430FR2433 Demo - UART RGB LED Color Mixing
//
//  Description: This example sets up Timer A to output PWM waveforms with
//  variable duty cycles based on UART byte values. These PWM waveforms control
//  the color of an external RGB LED. The PWM values are updated after every 3rd
//  byte received via UART.
//  ACLK = default REFO ~32768Hz
//  SMCLK = MCLK = DCO + FLL + 32KHz REFO REF = 1MHz
//
//           MSP430FR2433
//         ---------------
//     /|\|               |
//      | |               |
//      --|RST            |
//        |          P2.5 |<--- UART RX
//        |               |
//        |          P2.6 |---> UART TX
//        |               |
//        |          P1.2 |---> TA0.2 (Red)
//        |          P1.5 |---> TA1.1 (Green)
//        |          P1.4 |---> TA1.2 (Blue)
//
//   Aaron Barrera
//   Texas Instruments Inc.
//   October 2020
//   Built with Code Composer Studio v9.2 and IAR 7.20
//******************************************************************************

#include <msp430.h>
#include <stdint.h>
#include <stdbool.h>

unsigned int valueIntoCCR(unsigned char colorVal);
#define GPIO_ALL        BIT0|BIT1|BIT2|BIT3|BIT4|BIT5|BIT6|BIT7
#define MCLK_FREQ_MHZ 1                                     // MCLK = 1MHz

// Declare global variables
volatile uint8_t uartByteNum;                               // 0 for high, 1 for low

 // These variables are initialized in callbacks.h, so must be global here to use for UART
 volatile uint8_t redVal;       //Value of red
 volatile uint8_t greenVal;     //Value of green
 volatile uint8_t blueVal;      //Value of blue

// main.c
int main(void)
{
    WDTCTL = WDTPW | WDTHOLD;                               // Stop watchdog timer

    __bis_SR_register(SCG0);                                // disable FLL

    // Initialize Clock System
    // SMCLK = MCLK = DCO + FLL + 32KHz REFO REF = 1MHz
    CSCTL3 |= SELREF__REFOCLK;                              // Set REFO as FLL reference source
    CSCTL1 = DCOFTRIMEN | DCOFTRIM0 | DCOFTRIM1 | DCORSEL_0;// DCOFTRIM=3, DCO Range = 1MHz
    CSCTL2 = FLLD_0 + 30;                                   // DCODIV = 1MHz
    __delay_cycles(3);
    __bic_SR_register(SCG0);                                // enable FLL

    CSCTL4 = SELMS__DCOCLKDIV | SELA__REFOCLK;              // set default REFO(~32768Hz) as ACLK source, ACLK = 32768Hz
                                                            // default DCODIV as MCLK and SMCLK source
    // Initialize globals
    uartByteNum = 0;
```

UART RGB LEDG color mixing GUI

Housekeeping RGB Color Mixing Demo

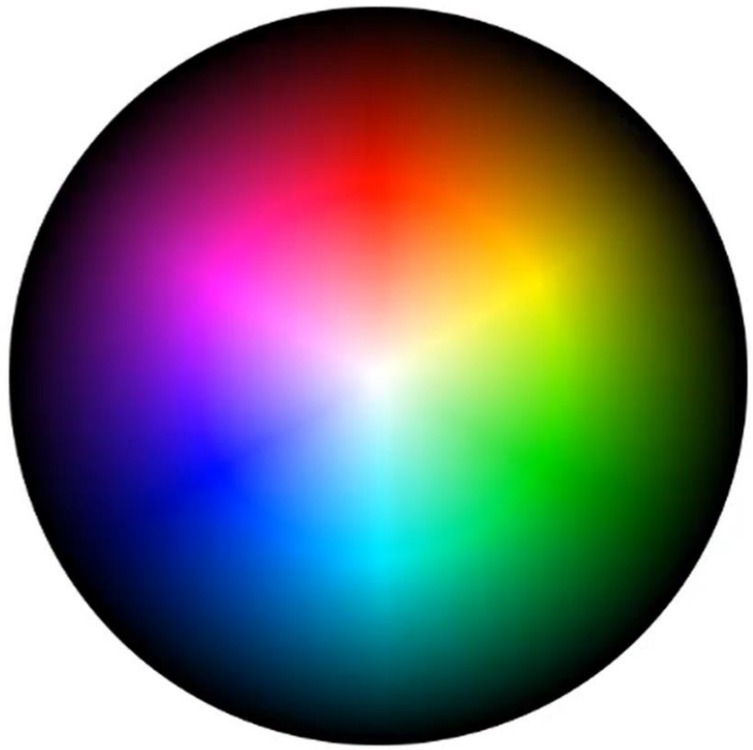
Instructions: Connect a common-cathode RGD LED in the configuration shown below. Use the color sliders, color wheel, or value box to output an RGB-mixed color to the LED. For best performance, only click on one location in the color wheel at a time when using the color wheel.



MSP430FR2433 – RGB LED H/W Setup

Color	Color Sliders (Dec)	Value (Dec)
RED	0 — 255	255
GREEN	0 — 255	255
BLUE	0 — 255	255

Color Wheel (Dec)



UART RGB LEDG color mixing resources

MSP430FR2433: https://www.ti.com/product/MSP430FR2433

MSP430FR2433 LaunchPad™ Development Kit: https://www.ti.com/tool/MSP-EXP430FR2433

RGB LED color mixing tech note: https://www.ti.com/lit/an/slaa979/slaa979.pdf

RGB LED color mixing source code: https://dev.ti.com/tirex/

GUI Composer: https://dev.ti.com/gallery/view/6284581/RgbColorMixing/

TEXAS INSTRUMENTS

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265