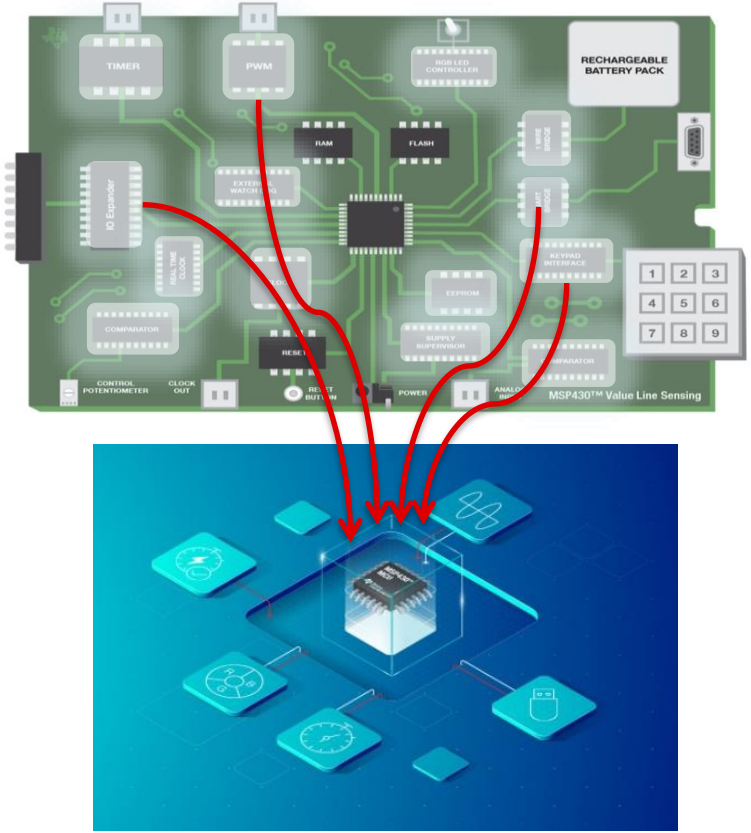
Analog and Digital Functions Made Easy with MSP Housekeeping MCUs

J.D. Crutchfield, MSP430 Applications

Agenda

- What is “housekeeping”?
  - Housekeeping functions
  - Example design scenario
  - Benefits
- MSP430 portfolio options
- Housekeeping function example walkthroughs
  - UART-controlled RGB LED controller
  - Programmable system wake-up controller

What is housekeeping?

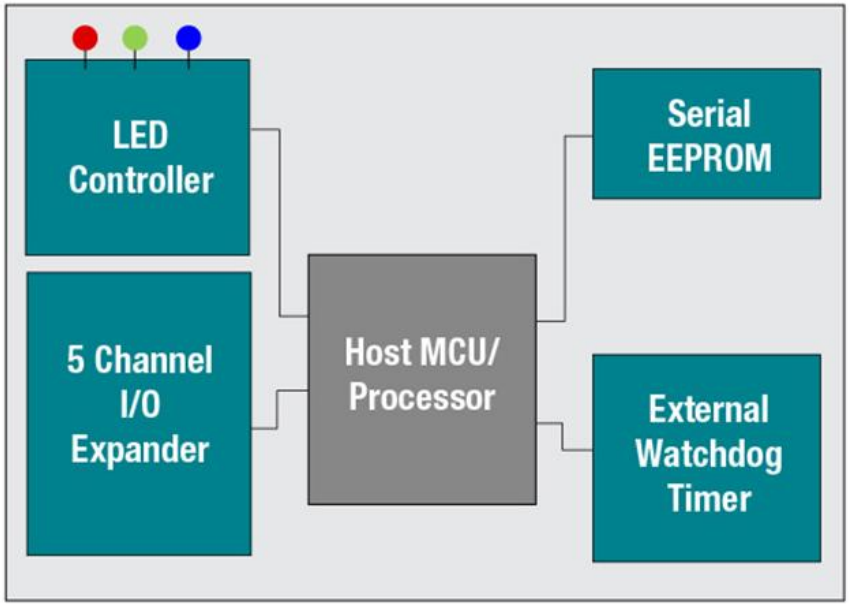


- Many designs have multiple, discrete analog and digital components

- Housekeeping MCUs replace discrete analog parts and manage a system's peripheral affairs

Starting a new system design

A discrete component approach

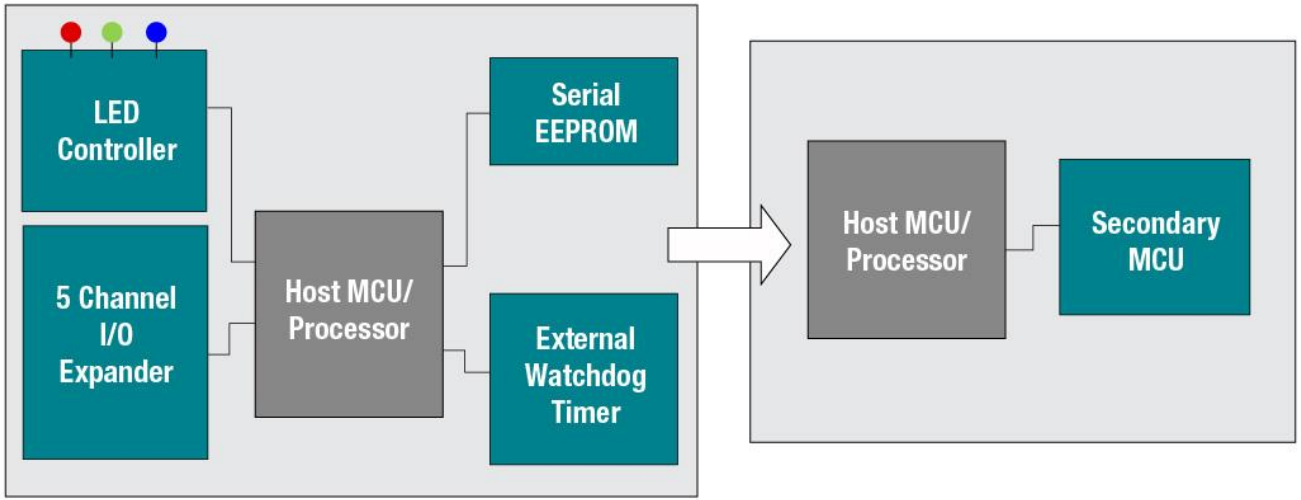


Total system bill-of-materials (BOM):

- LED controller
- I/O expander IC
- EEPROM IC
- External watchdog timer
- Microcontroller

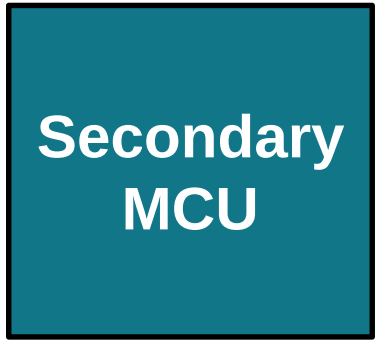
Starting a new system design

An integrated approach



Integrate your discrete system level functions into software on your microcontroller

Memory footprint



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

BOM cost

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**

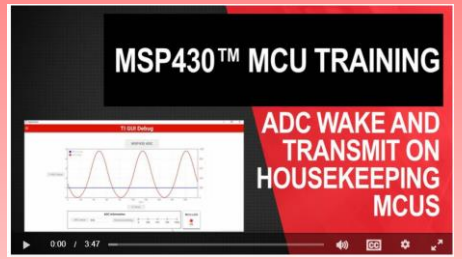
MSP430 MCUs for housekeeping functions

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

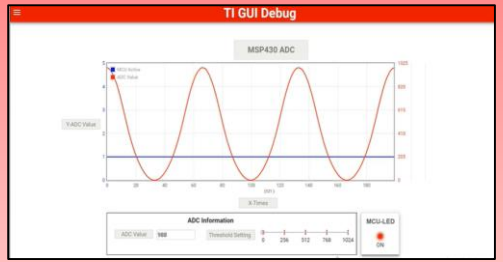
Memory	Primary Feature	Part Number	1ku Price
2KB		MSP430FR2110	$0.24
4KB		MSP430FR2111	$0.30
8KB		MSP430FR2422	$0.32
8KB	CapTIvate	MSP430FR2512	$0.52
16KB		MSP430FR2433	$0.50
32KB		MSP430FR2155	$0.62
64KB		MSP430FR2476	$0.83

Resources

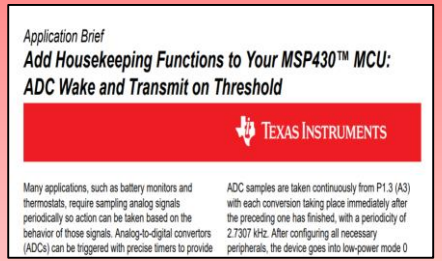
Implement simple functions in your system quickly with a low-cost microcontroller! All you need is an MSP430FR2433 LaunchPad™ development kit to get started. View our Housekeeping Resource Explorer and Housekeeping Home Page to find all of our Housekeeping resources, including videos, GUIs, app briefs and more



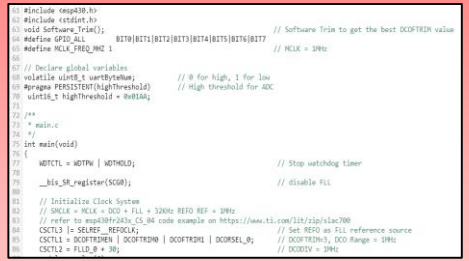
Training videos
(available in both English & Chinese)



Evaluation GUIs



Tech notes



Software

Learn → Evaluate → Implement

Resources

Replace combinations of simple functions with a housekeeping MCU


Training


E-book

Software

Timer Functions

- External RTC with backup memory
- 7-Segment LED Stopwatch
- External Programmable Watchdog Timer
- Programmable System Wake-up Controller
- Simple RTC-based System Wake-up Controller
- Voltage Monitor with a Time Stamp

Pulse Width Modulation Functions

- Analog Input to PWM Output
- Dual output 8-bit PWM DAC
- Servo Motor Control
- Stepper Motor Control
- UART Software-Controlled RGB LED Color Mixing

System Functions

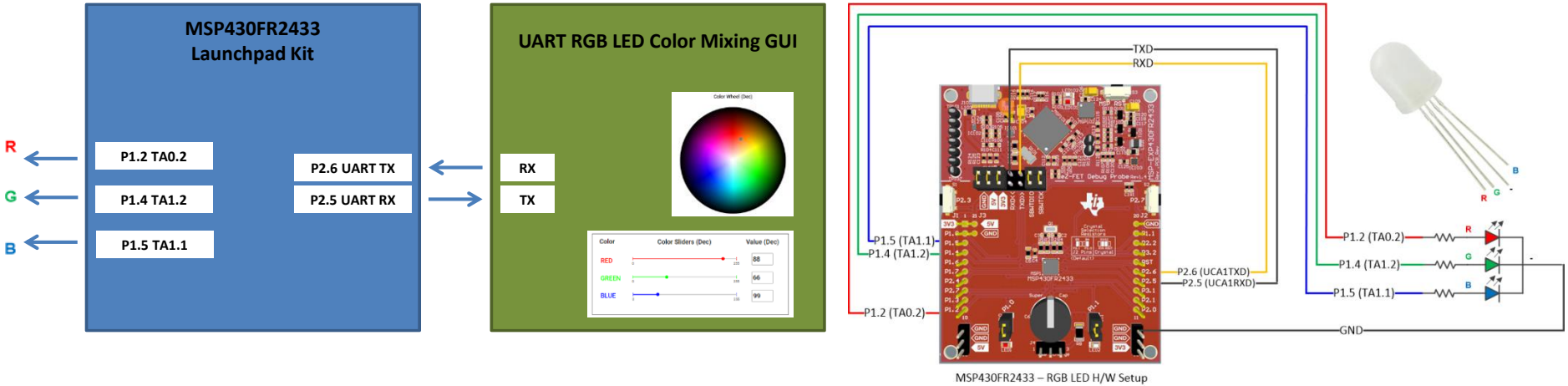
- ADC Wake and Transmit on Threshold
- EEPROM Emulation
- Low Power Hex Keypad
- Quadrature Encoder Position Counter
- Hysteresis Comparator with UART
- Multi-Function Reset Controller
- Single Slope Analog-to-Digital Conversion Technique
- Tamper Detection
- Programmable Clock Source
- Programmable Frequency-locked Loop

Communication Functions

- Single Wire Communication Host
- SPI IO Expander
- USB-to-UART Bridge
- SPI-to-I2C Bridge

MSP430 housekeeping example:
UART-controlled RGB LED controller

System overview and hardware connections



UART RGB LEDG color mixing application brief

Application Brief

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

Texas Instruments

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.

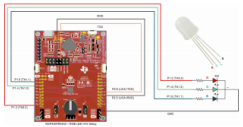


Figure 2. MSP430FR2433 LaunchPad and Connections

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.

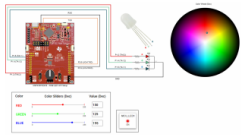


Figure 3. RGB LED Color Mixing GUI

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

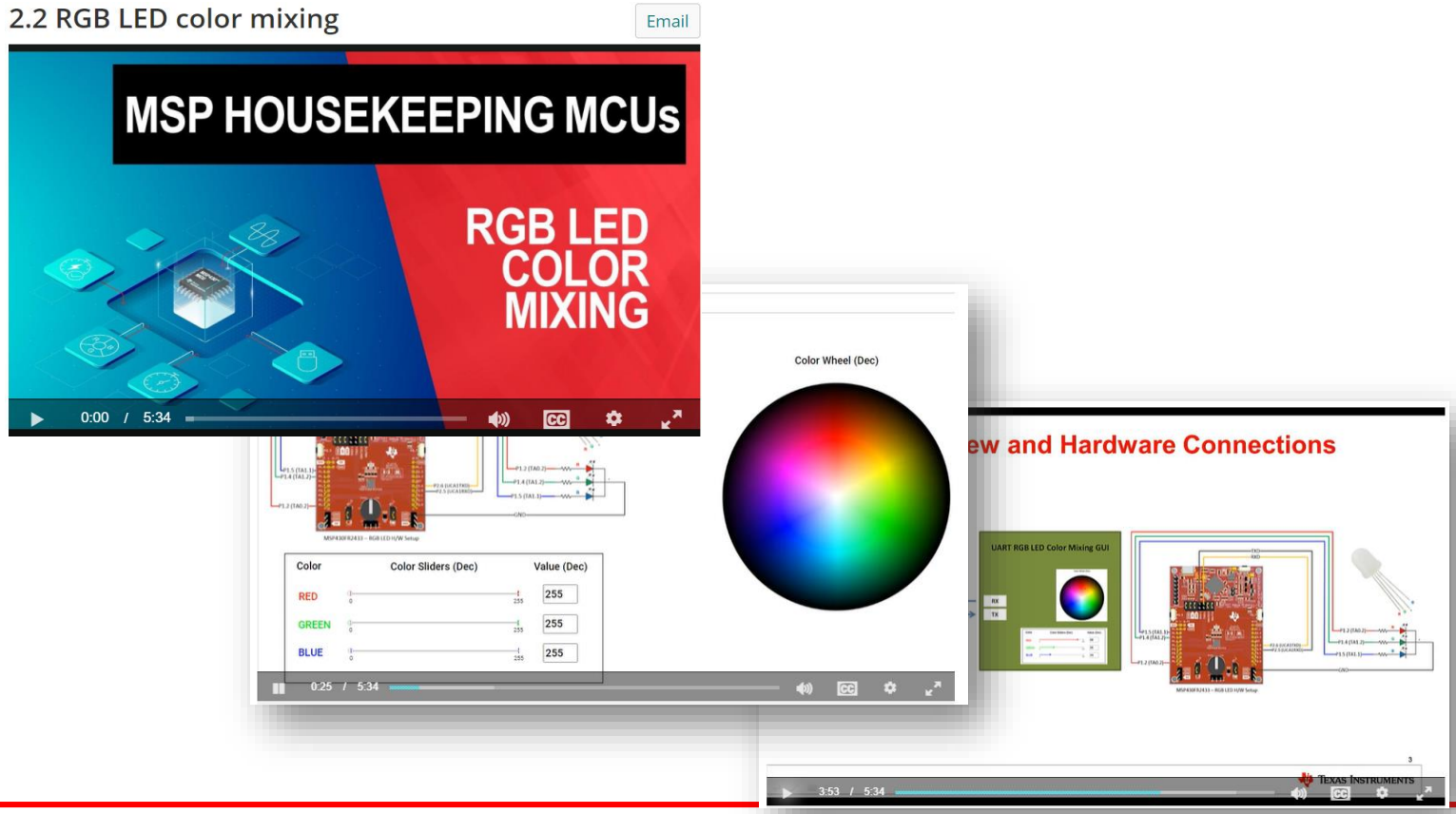
1. Watch the training video "UART RGB Color Mixing with a Housekeeping MCU" to learn how to use the GUI to mix colors on an RGB LED.
2. Order a MSP430FR2433 LaunchPad kit to evaluate the UART RGB Color Mixing example code.
3. Download and test this example with the UART RGB Color Mixing example GUI to 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

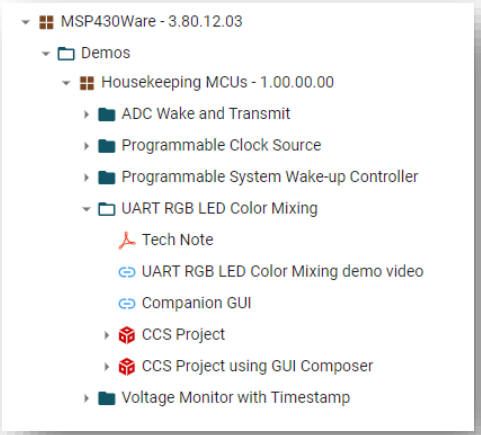
Part Number	Key Features
MSP430FR2433	16KB FRAM, 4KB SRAM, 10-bit ADC, UART/SPI/I2C, Timer
MSP430FR2422	8KB FRAM, 2KB SRAM, 10-bit ADC, UART/SPI/I2C, Timer

Texas Instruments

UART RGB LEDG color mixing demo video



UART RGB LEDG color mixing example code



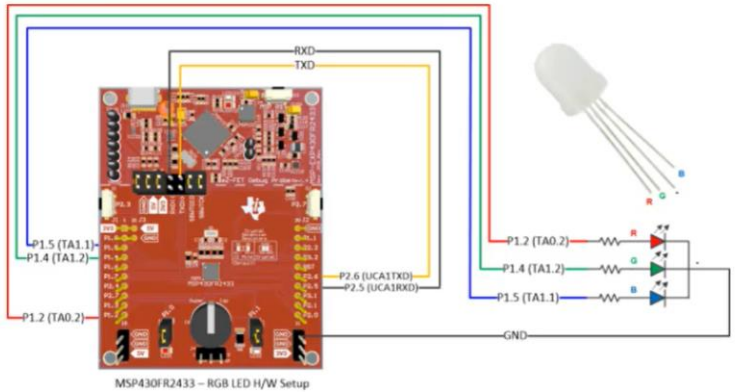
Software / MSP430Ware (3.80.12.03) / Demos / Housekeeping MCUs (1.00.00.00) / UART RGB LED Color Mixing / CCS Project / **main.c**

```
 * 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;
    redVal = 0;
```

UART RGB LEDG color mixing GUI

Housekeeping RGB Color Mixing Demo

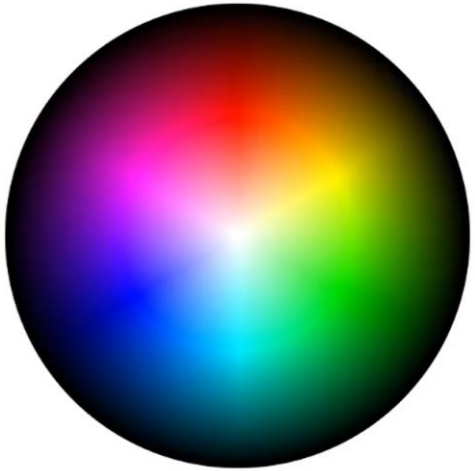
Instructions: Connect a common-cathode RGD LED in the configuration shown below. Use the color sliders, color wheel, or value box to output an RGB-mixed color to the LED. For best performance, only click on one location in the color wheel at a time when using the color wheel.



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

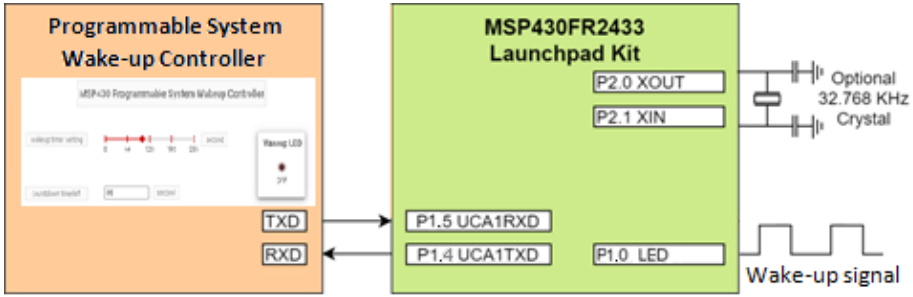
RGB LED color mixing tech note: https://www.ti.com/lit/an/slaa979/slaa979.pdf

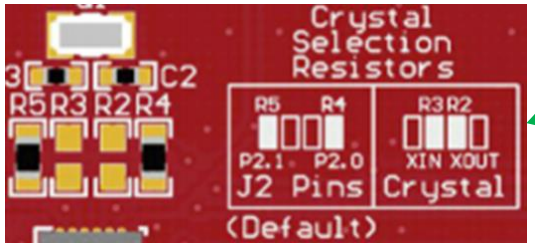
RGB LED color mixing source code: https://dev.ti.com/tirex/

GUI Composer: https://dev.ti.com/gallery/view/TIMSPGC/RgbColorMixing/ver/1.0.0/

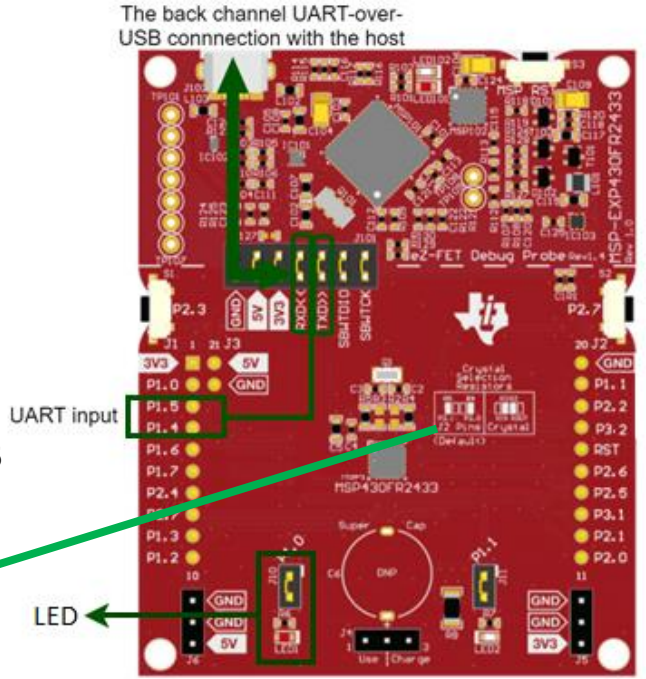
MSP430 housekeeping example:
Programmable system wake-up controller

System overview and hardware connections





MUST CHANGE R4/R5
TO R2/R3 FOR 32kHz
XTAL



Programmable system wake-up controller resources

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

MSP430FR2433 LaunchPad™ development kit: https://www.ti.com/tool/MSP-EXP430FR2433

Programmable system wake-up controller tech note: https://www.ti.com/lit/an/slaa790a/slaa790a.pdf

Programmable system wake-up controller source code: https://dev.ti.com/tirex/

GUI Composer:
https://dev.ti.com/gallery/view/TIMSPGC/Programmable_System_Wakeup_Controller_GUI/ver/1.0.0/

Links and more information

- Housekeeping training series (includes videos, example code, and application notes):
  - https://training.ti.com/msp-housekeeping-mcus

- Get the MSP430FR2433 Launchpad:
  - https://www.ti.com/tool/MSP-EXP430FR2433

- TI's Cloud Tools ecosystem:
  - https://dev.ti.com/tirex/explore

- Explore MSP430™ devices:
  - https://www.ti.com/msp

- Application support:
  - https://www.e2e.ti.com

Contact the presenter

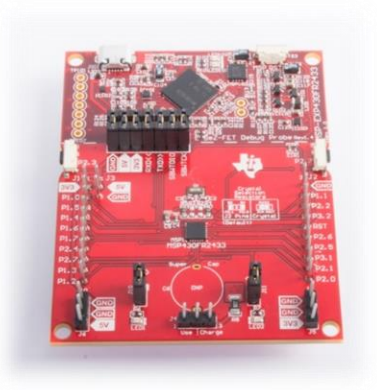
- JD Crutchfield
  - jd@ti.com

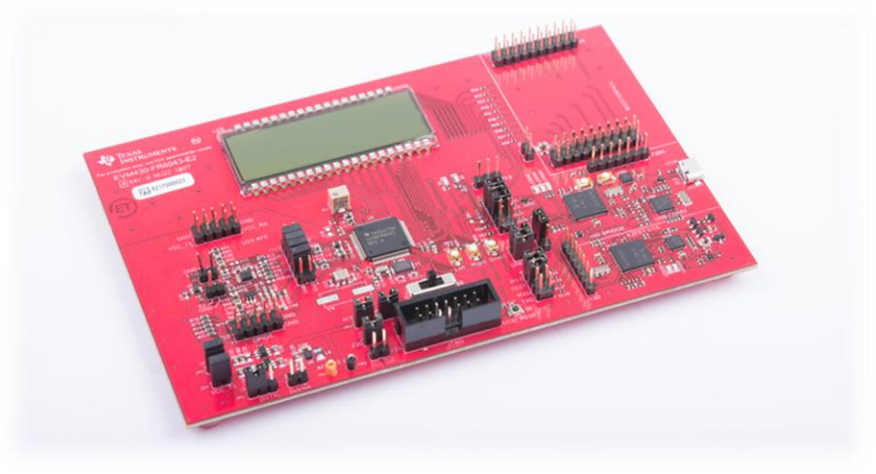
Backup

MSP430 housekeeping example:
External programmable watchdog timer

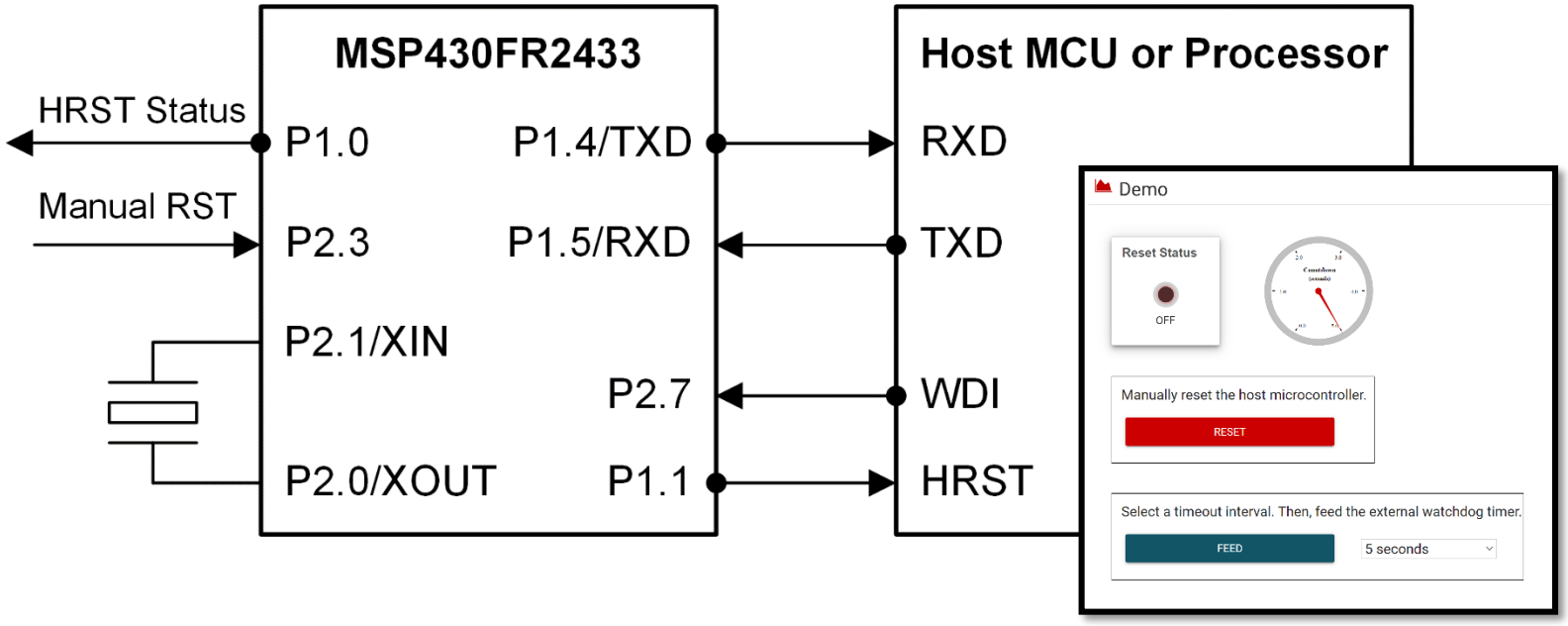
Hardware requirements

- MSP430FR2433 LaunchPad development kit
- Host microcontroller (MCU) or processor (optional)
- Jumper wires (optional)





System overview and hardware connections



External programmable watchdog timer resources

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

MSP430FR2433 LaunchPad™ development kit: https://www.ti.com/tool/MSP-EXP430FR2433

This example tech note: https://www.ti.com/lit/an/slaa987/slaa987.pdf

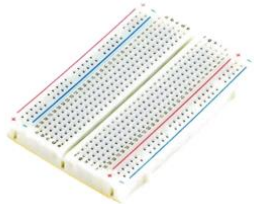
This example source code: https://dev.ti.com/tirex/

GUI Composer:
https://dev.ti.com/gallery/view/TIMSPGC/External_Programmable_Watchdog_Timer/ver/1.0.0/

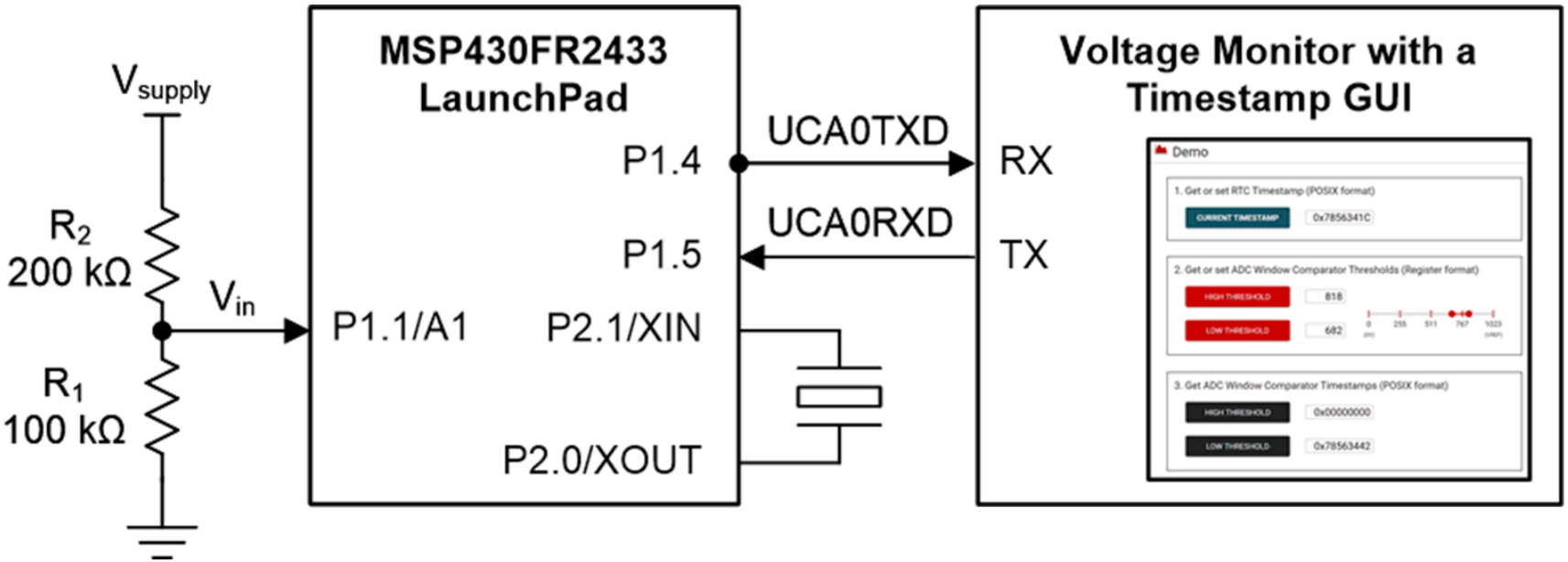
MSP430 housekeeping example:
Voltage monitor with a timestamp

Hardware requirements

- MSP430FR2433 LaunchPad Development Kit
- Resistors
- M-to-F jumper wires
- Breadboard (optional)



System overview and hardware connections



Voltage Monitor with a Timestamp Resources

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

MSP430FR2433 LaunchPad™ development kit: https://www.ti.com/tool/MSP-EXP430FR2433

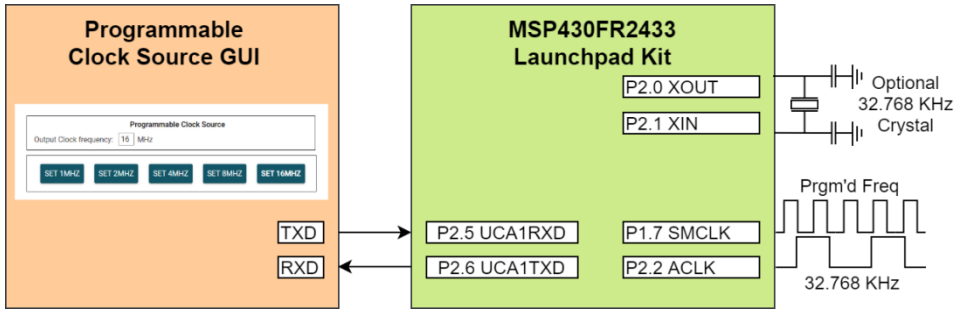
This example tech note: https://www.ti.com/lit/an/slaa980/slaa980.pdf

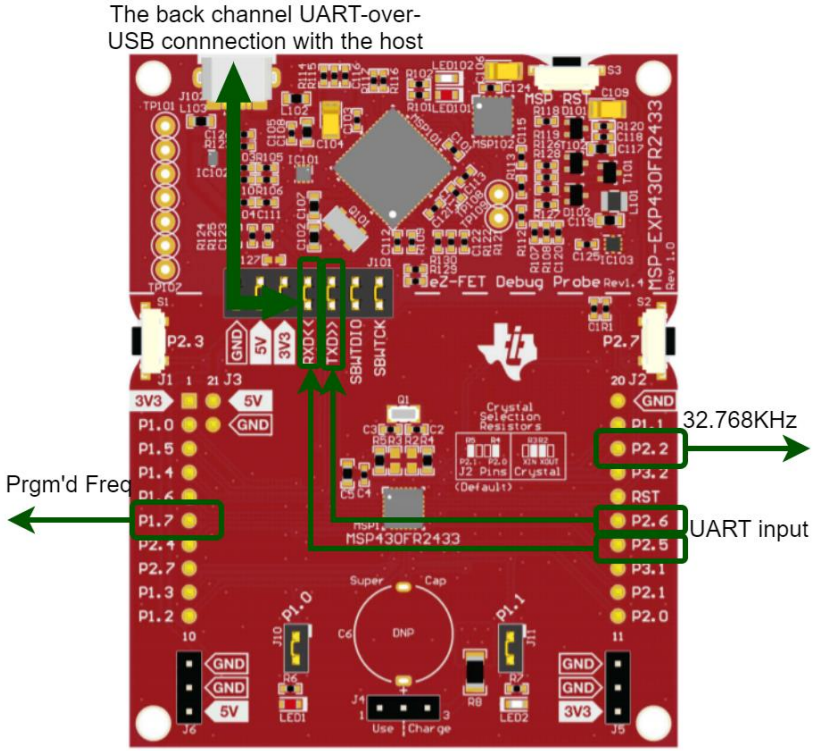
This example source code: https://dev.ti.com/tirex/

GUI Composer: https://dev.ti.com/gallery/info/TIMSPGC/Voltage_Monitor_with_Timestamp/

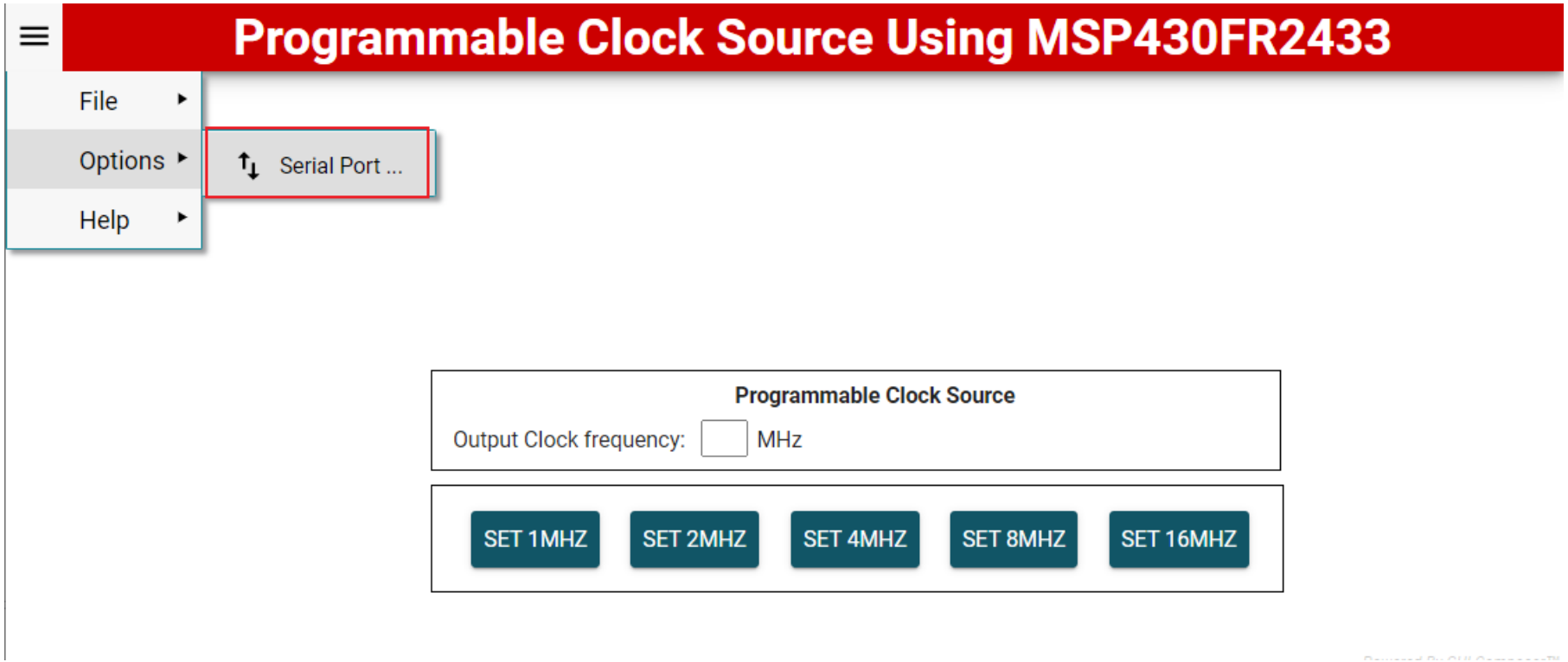
MSP430 housekeeping example:
Programmable clock source

System overview and hardware connections





GUI configuration



Programmable clock source resources

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

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

This example tech note: https://www.ti.com/lit/an/slaa981/slaa981.pdf

This example source code: https://dev.ti.com/tirex/

GUI Composer:
https://dev.ti.com/gallery/view/5617547/Programmable_clock_source_FR2433_GUI/ver/1.0.0/

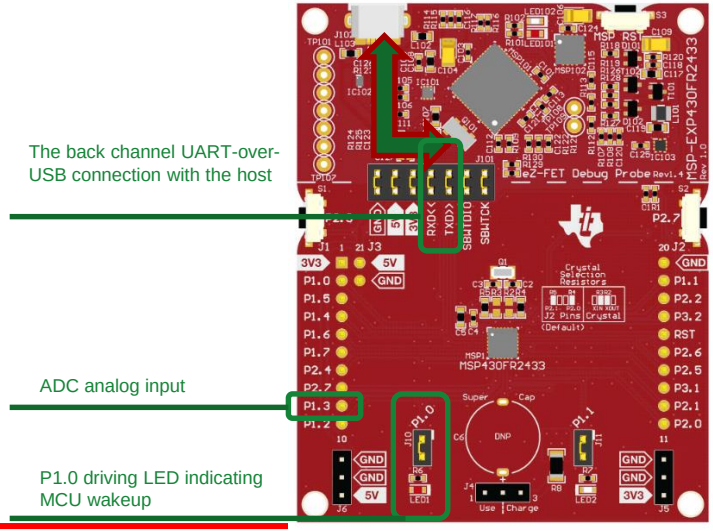
MSP430 housekeeping example:
ADC wake and transmit

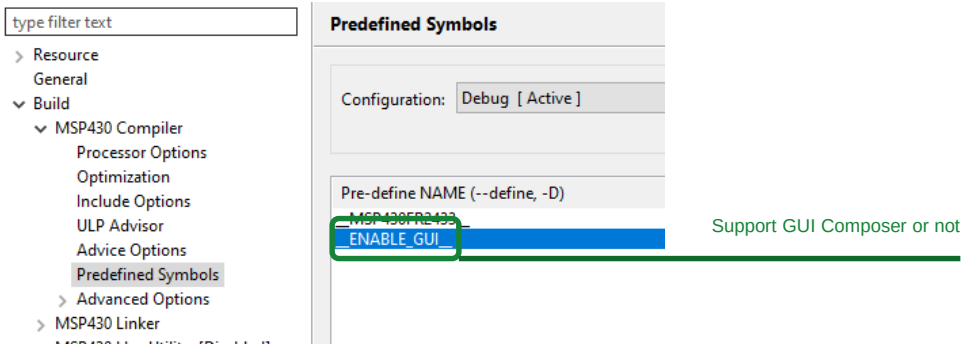
ADC wake and transmit example background

- MSP430 device is suitable on analog-to-digital converter application case such as battery monitors, temperature sensors and so on. In these use-case, low-power standby, ADC monitor and communication with host in needed.
  - Microcontroller is in low power mode and waiting for ADC conversion value higher than the pre-set threshold or waiting for UART communication updating threshold value
  - Once ADC conversion value higher than the threshold, microcontroller is woken up from low power mode, one port is set to indicate this event and start 1 second interval timer
  - After 1 second interval timer is started, ADC conversion code is sent to host through UART every 1 second
  - Once UART received the data, microcontroller is woken up from low power mode and store the data on FRAM to update the threshold

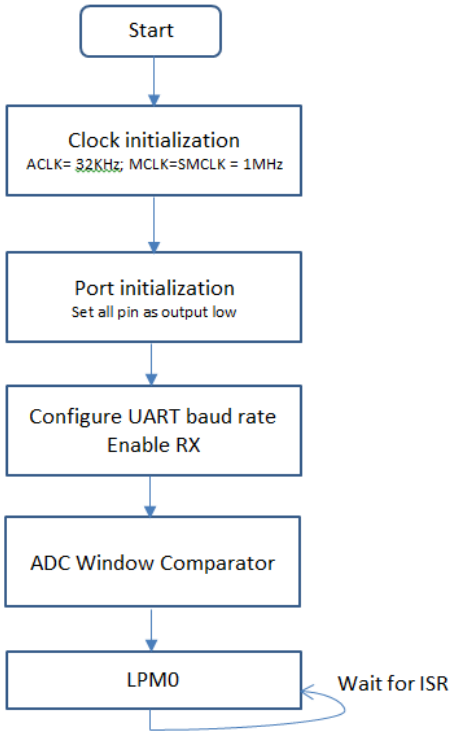
Development environment

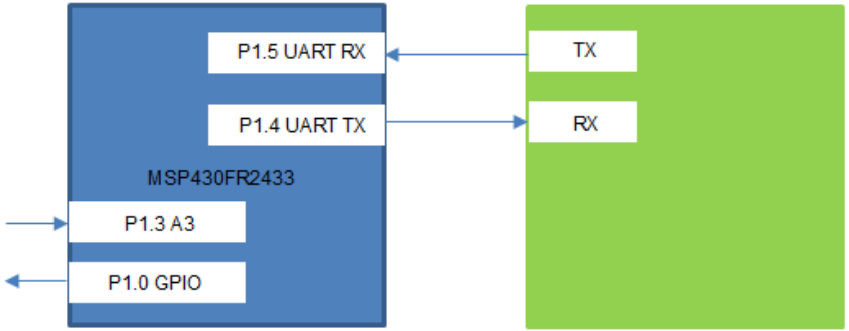
- ADC Wake and Transmit Example code runs on MSP430FR2433 Launch Pad
- GUI composer provided the PC graphic interface to show microcontroller wake up status, ADC conversion code and threshold setting.
- “ENABLE_GUI” pre-definition on CCS project is the flag to set the example code supporting GUI or not

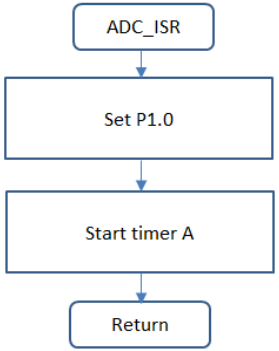


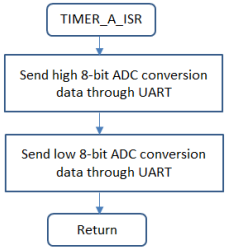


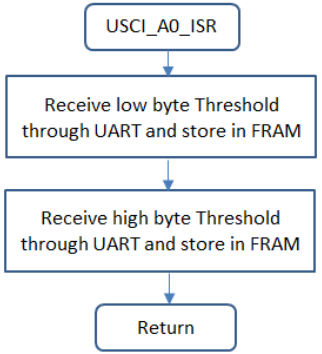
Example code flow chart





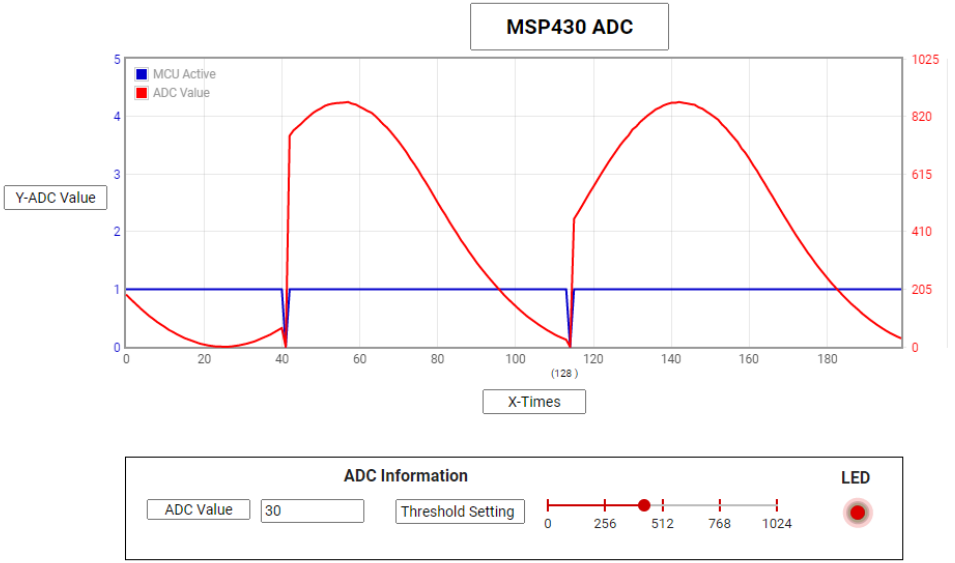






MSP430FR2433 ADC wake and transmit example demo on GUI

- Scalar graph widget real-time display LED status and ADC conversion code
- ADC conversion code is displayed on textbox as well
- Threshold value can be set on slider widget
- Run GUI Composer Application
- Input sine wave on analog input
- LED on when the analog voltage is higher than the pre-set threshold. MCU wakeup and continue send ADC code to GUI
- Set threshold using slider, Repeat the above process, you can see ADC waveform and LED on happens when the analog voltage is higher than the updated threshold



ADC wake and transmit resources

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

MSP430FR2433 LaunchPad™ development kit: https://www.ti.com/tool/MSP-EXP430FR2433

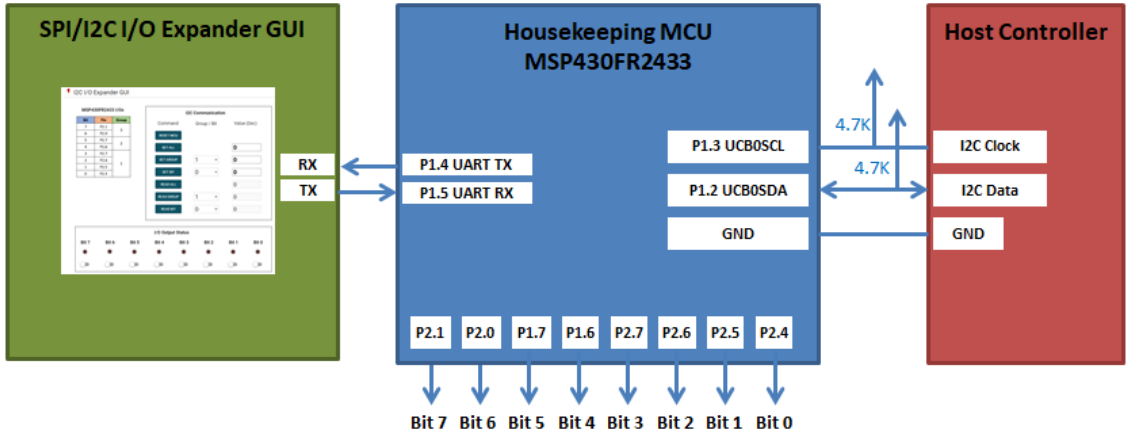
This example tech note: https://www.ti.com/lit/an/slaa963/slaa963.pdf

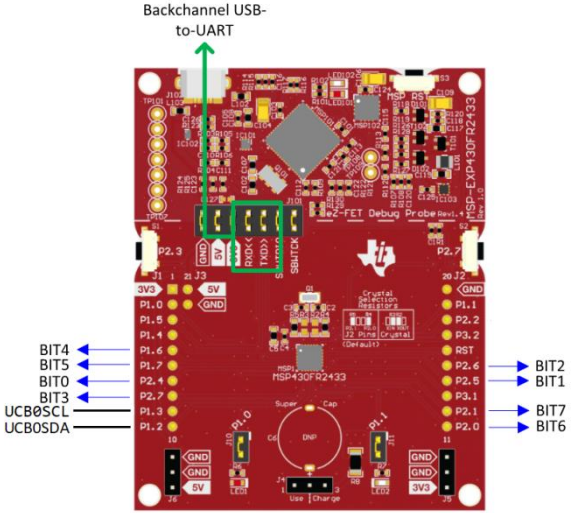
This example source code: https://dev.ti.com/tirex/

GUI Composer: https://dev.ti.com/gallery/view/TIMSPGC/ADC_LPM_FR2433_GUI/ver/1.0.0/

MSP430 housekeeping example:
I2C and SPI IO expander

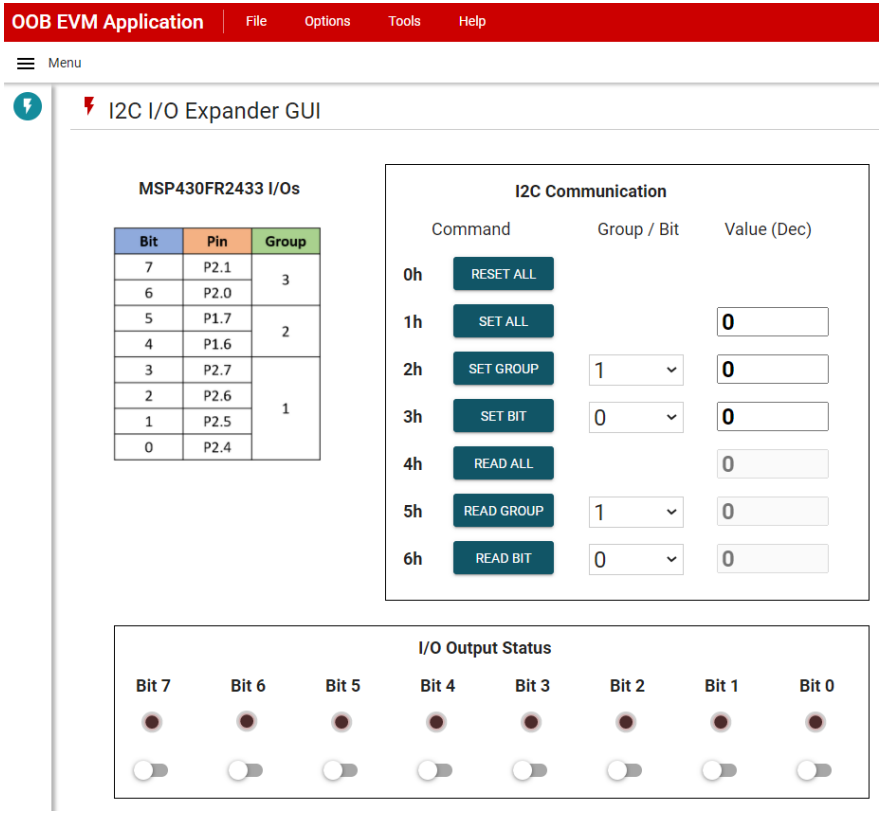
System overview and hardware connections



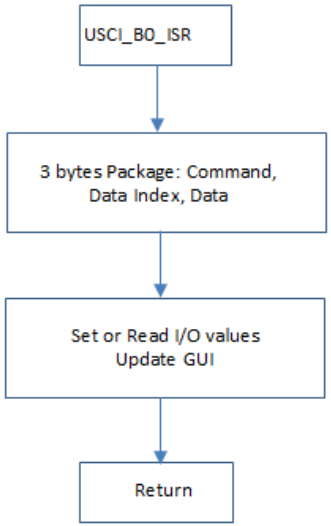


Optional

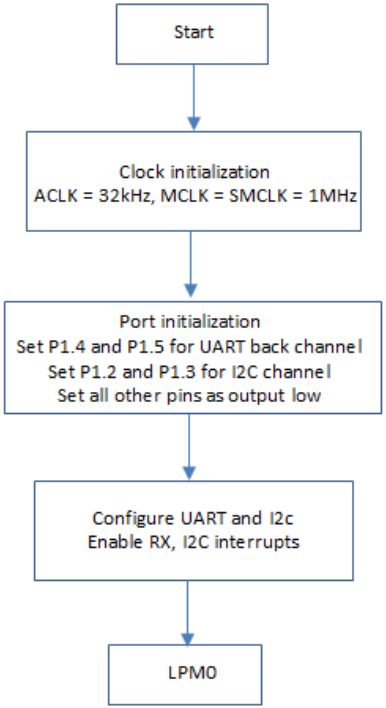
GUI



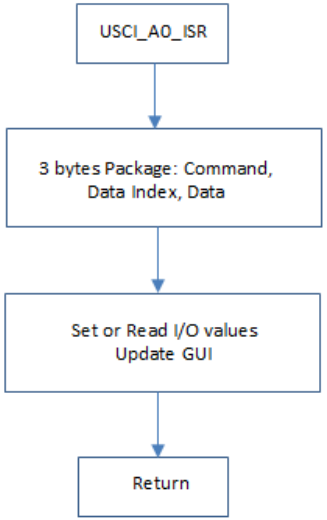
Example code flow



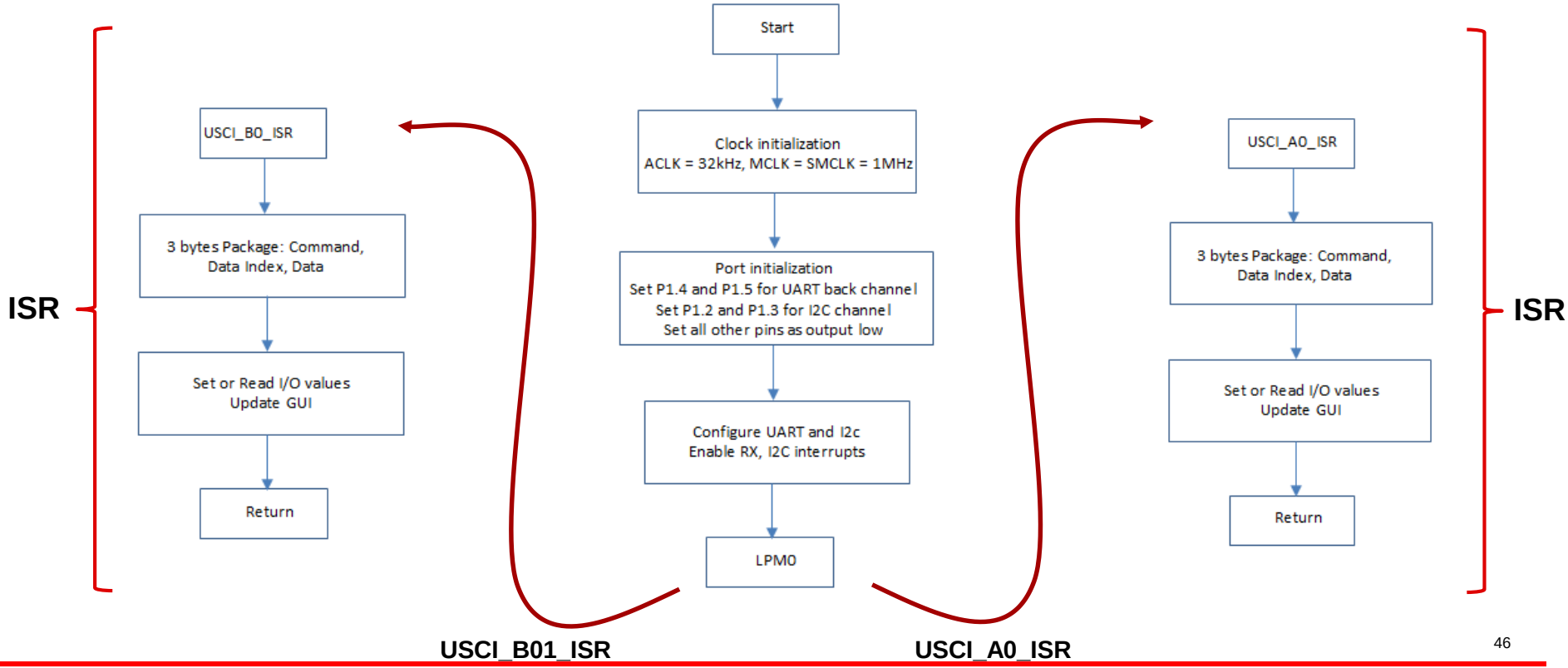
Init.



Low-power mode



Example code flow



I2C and SPI IO expander resources

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

MSP430FR2433 LaunchPad™ development kit: https://www.ti.com/tool/MSP-EXP430FR2433

This example tech note:

- https://www.ti.com/lit/an/slaa985/slaa985.pdf
- https://www.ti.com/lit/an/slaa997/slaa997.pdf

This example source code: https://dev.ti.com/tirex/

GUI Composer:

- https://dev.ti.com/gallery/view/TIMSPGC/I2C_IO_Expander/ver/1.0.0/
- https://dev.ti.com/gallery/view/TIMSPGC/SPI_IO_Expander/ver/1.0.0/