MSP432P401R SimpleLink™ Microcontroller LaunchPad™ Development Kit (MSP-EXP432P401R)

The SimpleLink™ MSP-EXP432P401R LaunchPad™ development kit is an easy-to-use evaluation module for the SimpleLink MSP432P401R microcontroller. It contains everything needed to start developing on the SimpleLink MSP432™ low-power + performance Arm® 32-bit Cortex®-M4F microcontroller (MCU), including onboard debug probe for programming, debugging, and energy measurements. The MSP432P401R device supports low-power applications requiring increased CPU speed, memory, analog, and 32-bit performance.

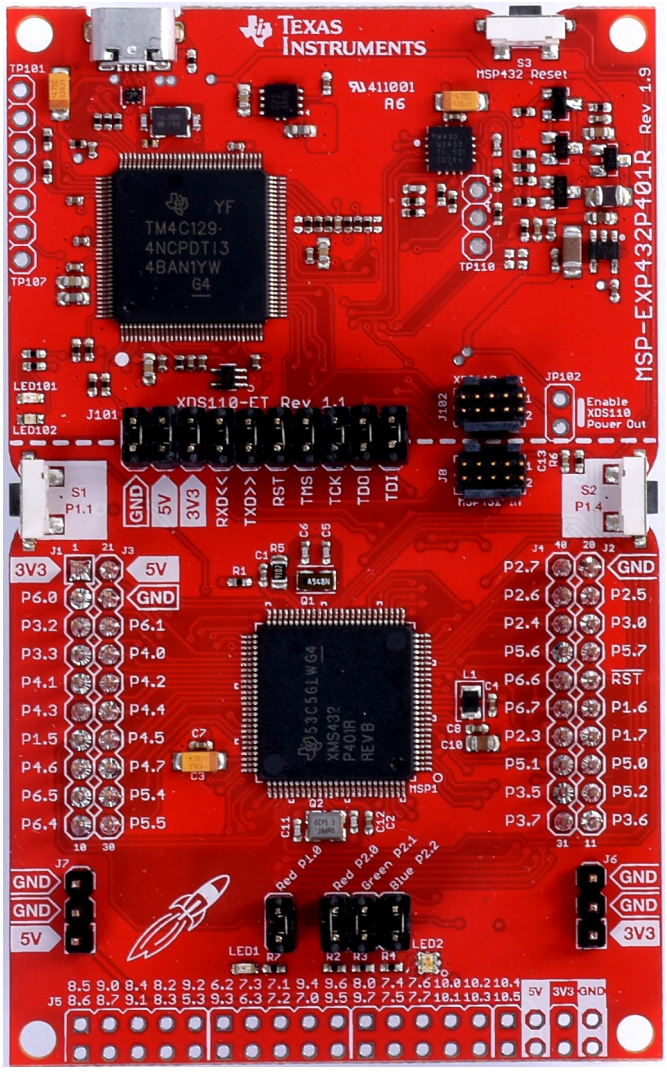

Figure 1. MSP-EXP432P401R LaunchPad™ Development Kit

Contents

1 Getting Started 3
2 Hardware 5
3 Software Examples 20
4 Resources 27
5 FAQ 31
6 Schematics 36

List of Figures

1 MSP-EXP432P401R LaunchPad™ Development Kit 1
2 MSP-EXP432P401R Overview 5
3 Block Diagram 5
4 MSP432P401RIPZ Pinout 6
5 XDS110-ET Debug Probe 7
6 XDS110-ET Isolation Block 8
7 Application Backchannel UART in Device Manager 9
8 EnergyTrace™ Technology Preferences 11
9 EnergyTrace™ Windows 12
10 MSP-EXP432P401R Power Block Diagram 13
11 LaunchPad™ Development Kit to BoosterPack™ Plug-in Module Connector Pinout 16
12 Differences Between Rev 1.0 (Black) and Rev 2.0 (Red) 18
13 BSL Update Utility in TI Resource Explorer 19
14 Out-of-Box GUI Running Locally 21
15 Out-of-Box GUI Running From TI Cloud Tools 22
16 Importing and Converting an Image With MSP Image Reformer 23
17 Hardware Setup and Connections 24
18 Determine COM Port Number Using Device Manager on Windows 25
19 Example Serial Terminal Configuration 25
20 Snapshot of Serial Terminal Connected to Running Fuel Gauge Demo 26
21 Using TI Resource Explorer to Browse MSP-EXP432P401R in SimpleLink SDK 30
22 SWD Mode Settings 31
23 Target Configurations 32
24 Launch Selected Configuration 32
25 Show All Cores 33
26 Connect Target 34
27 MSP432_Factory_Reset Script 34
28 Schematics (1 of 6) 36
29 Schematics (2 of 6) 37
30 Schematics (3 of 6) 38
31 Schematics (4 of 6) 39
32 Schematics (5 of 6) 40
33 Schematics (6 of 6) 41

List of Tables

1 Isolation Block Connections 7
2 Default Clock Operation 14
3 Hardware Change Log 17
4 Software Examples 20
5 IDE Minimum Requirements for MSP-EXP432P401R 20

6 Source File and Folders 24
7 Source Files and Folders 27
8 How MSP Device Documentation is Organized 30

Trademarks

1 Getting Started

1.1 Introduction

The SimpleLink MSP-EXP432P401R LaunchPad development kit is an easy-to-use evaluation module for the MSP432P401R microcontroller. It contains everything needed to start developing on the MSP432 Low-Power + Performance Arm 32-bit Cortex-M4F microcontroller (MCU), including onboard debug probe for programming, debugging, and energy measurements. The MSP432P401R microcontroller supports low-power applications that require increased CPU speed, memory, analog, and 32-bit performance.

Rapid prototyping is simplified by access to the 40-pin headers and a wide variety of BoosterPack™ plug-in modules that enable technologies such as wireless connectivity, graphical displays, environmental sensing, and many more. Free software development tools are also available such as TI's Eclipse-based Code Composer Studio™ IDE, IAR Embedded Workbench™ IDE, and Keil® µVision® IDE. Code Composer Studio IDE supports EnergyTrace™ technology when paired with the MSP432P401R LaunchPad development kit. More information about the LaunchPad development kit, the supported BoosterPack plug-in modules, and the available resources can be found at TI's LaunchPad development kit portal. To get started quickly and find available resources in the SimpleLink MSP432 software development kit (SDK), visit the TI Cloud Development Environment.

1.2 Key Features

- Low-power Arm Cortex-M4F MSP432P401R
- 40-pin LaunchPad development kit standard that leverages the BoosterPack plug-in module ecosystem
- XDS110-ET, an open-source onboard debug probe featuring EnergyTrace+ technology and application UART
- Two buttons and two LEDs for user interaction
- Backchannel UART through USB to PC

1.3 What's Included

1.3.1 Kit Contents

- One MSP-EXP432P401R LaunchPad development kit
- One Micro USB cable
- One Quick Start Guide

1.3.2 Software Examples (Section 3)

- Out-of-Box Software Example
- BOOSTXL-K350QVG-S1 Graphics Library Example
- 430BOOST-SHARP96 Graphics Library Example
- BOOSTXL-BATPAKMKII Fuel Gauge Example
- BOOSTXL-SENSORS Sensor GUI Example
- BOOSTXL-SENSORS Sensor GUI with TI-RTOS Example

1.4 First Steps: Out-of-Box Experience

An easy way to get familiar with the EVM is by using its preprogrammed out-of-box code. It demonstrates some key features of the LaunchPad development kit from a user level, showing how to use the pushbutton switches together with onboard LEDs and basic serial communication with a computer.

For a more detailed explanation of the out-of-box demo, see Section 3.

1.5 Next Steps: Looking Into the Provided Code

To get started, you need an integrated development environment (IDE) to explore and start editing the code examples. See Section 4 for more information on IDEs and where to download them.

The out-of-box source code and more code examples can be downloaded from the MSP-EXP432P401R tool folder. Find what code examples are available and more details about each example in Section 3. All code is licensed under BSD, and TI encourages reuse and modifications to fit specific needs.

2 Hardware

Figure 2 shows an overview of the EVM hardware.

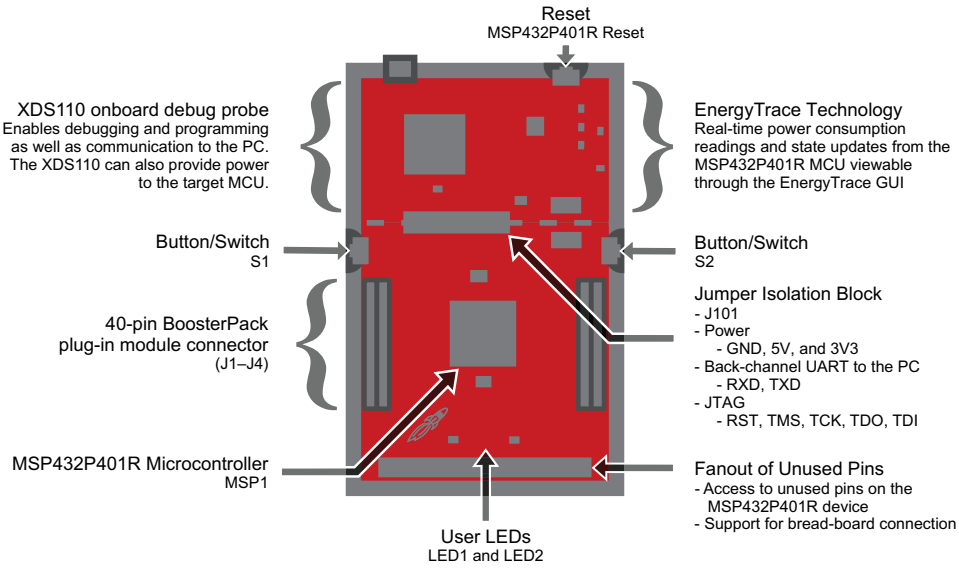

Figure 2. MSP-EXP432P401R Overview

2.1 Block Diagram

Figure 3 shows the block diagram.

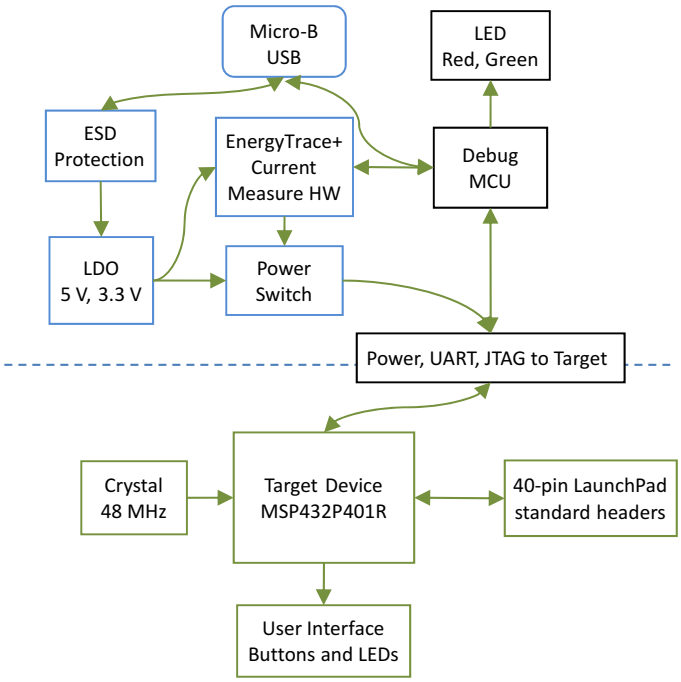

Figure 3. Block Diagram

2.2 SimpleLink MSP432P401R MCU

The MSP432P401R is the first MSP432 family device featuring low-power performance with an Arm Cortex-M4F core. Device features include:

- Low-power Arm Cortex-M4F MSP432P401R
- Up to 48-MHz system clock
- 256KB of flash memory, 64KB of SRAM, and 32KB of ROM with SimpleLink MSP432 SDK libraries
- Four 16-bit timers with capture, compare, or PWM, two 32-bit timers, and an RTC
- Up to eight serial communication channels (I^2C, SPI, UART, and IrDA)
- Analog: precision ADC, capacitive touch, comparator
- Digital: AES256, CRC, µDMA

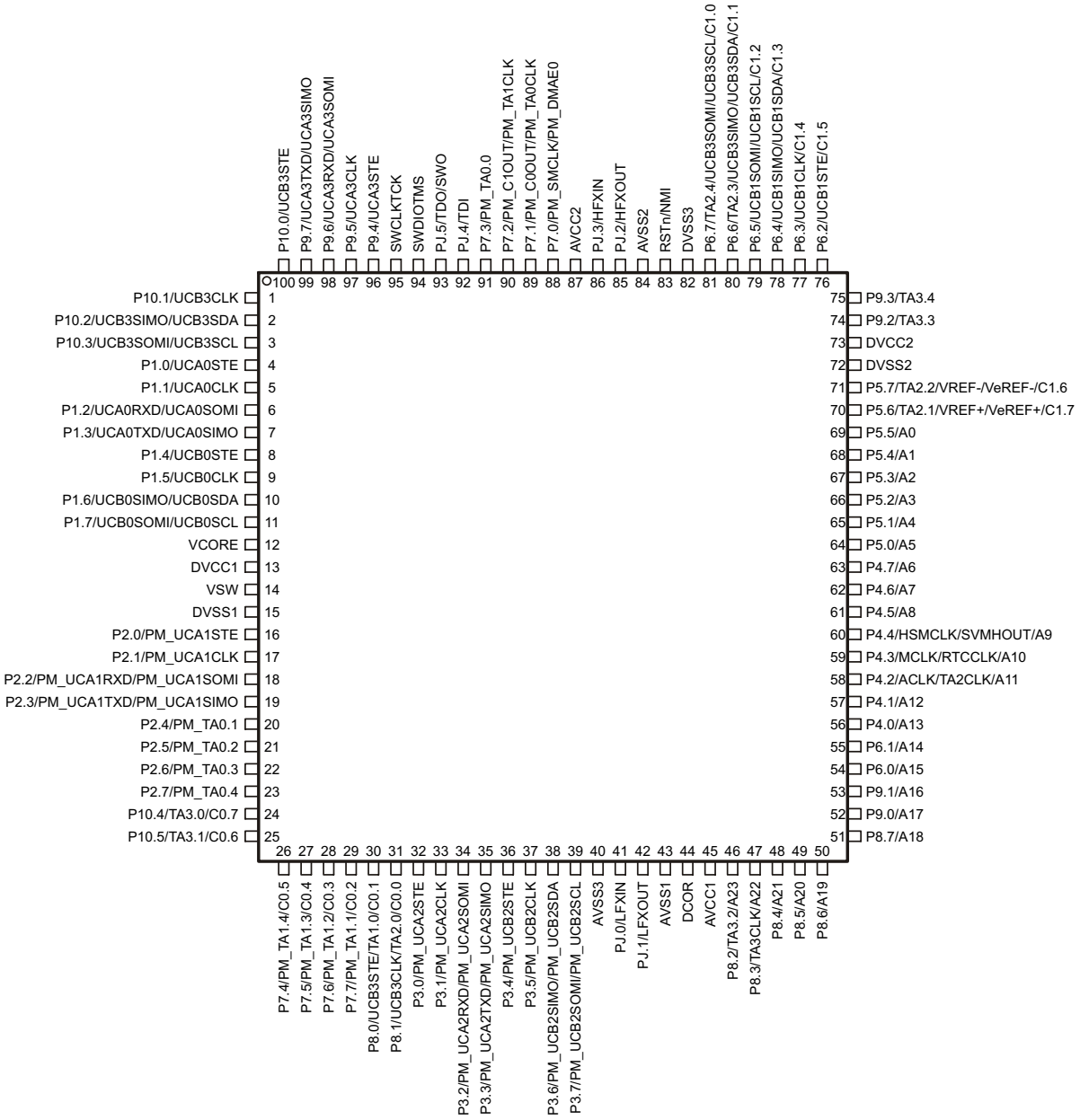

Figure 4. MSP432P401RIPZ Pinout

2.3 XDS110-ET Onboard Debug Probe

To keep development easy and cost effective, TI's LaunchPad development kits integrate an onboard debug probe, which eliminates the need for expensive programmers. The MSP-EXP432P401R has the XDS110-ET debug probe, which is a simple low-cost debug probe that supports nearly all TI Arm device derivatives.

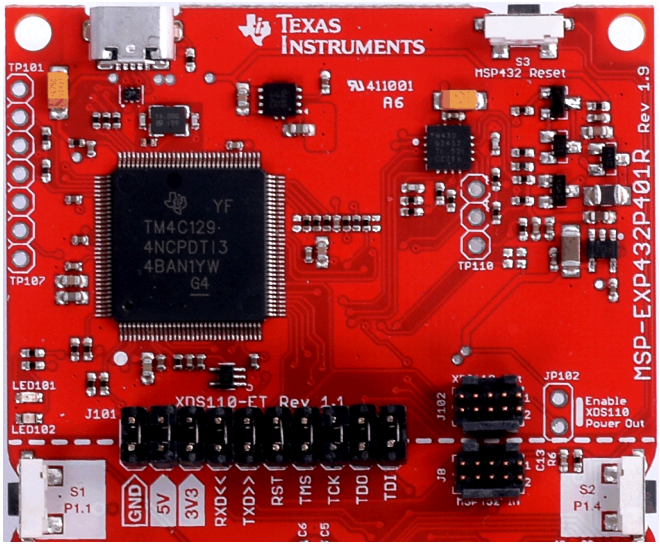

Figure 5. XDS110-ET Debug Probe

The XDS110-ET hardware can be found in the schematics in Section 6 and in the MSP-EXP432P401R Hardware Design Files.

2.3.1 XDS110-ET Isolation Block J101

The J101 isolation block is composed of J101 jumpers shown in Table 1. The J101 isolation block allows the user to connect or disconnect signals that cross from the XDS110-ET domain into the MSP432P401R target domain. This crossing is shown by the silkscreen dotted line across the LaunchPad development kit through J101. No other signals cross the domain, so the XDS110-ET can be completely decoupled from the MSP432P401R target side. This includes XDS110-ET power and GND signals, UART, and JTAG signals.

Table 1 lists the signals that are controlled at the isolation block.

Table 1. Isolation Block Connections

Signal	Description
GND	GND power connection between XDS110 and MSP432 target GND planes. The GND jumper is populated to connect the separate GND planes. This connection is required for proper operation with 3V3, 5V, UART, and JTAG.
5V	5-V power rail, VBUS from USB
3V3	3.3-V power rail, derived from VBUS by an LDO in the XDS110-ET domain
RXD <<	Backchannel UART: The target MCU receives data through this signal. The arrows indicate the direction of the signal.
TXD >>	Backchannel UART: The target MCU sends data through this signal. The arrows indicate the direction of the signal.
RST	MCU RST signal (active low)
TCK_SWCLK	Serial wire clock input (SWCLK) / JTAG clock input (TCK)
TMS_SWDIO	Serial wire data input/output (SWDIO) / JTAG test mode select (TMS)
TDO_SWO	Serial wire trace output (SWO) / JTAG trace output (TWO) (Also PJ.5)
TDI	JTAG test data input (Also PJ.4)

Reasons to open these connections:

- To remove any and all influence from the XDS110-ET debug probe for high accuracy target power measurements
- To control 3-V and 5-V power flow between the XDS110-ET and target domains
- To expose the target MCU pins for other use than onboard debugging and application UART communication
- To expose the UART interface of the XDS110-ET so that it can be used for devices other than the onboard MCU.

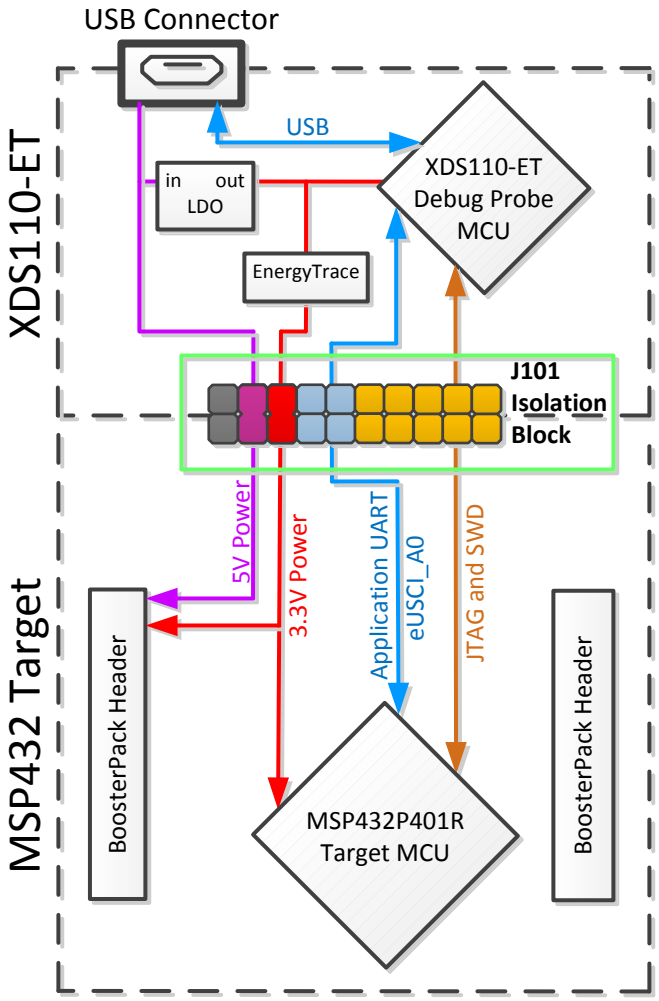

Figure 6. XDS110-ET Isolation Block

2.3.2 Application (or "Backchannel") UART

The XDS110-ET provides a "backchannel" UART-over-USB connection with the host, which can be very useful during debugging and for easy communication with a PC.

The backchannel UART allows communication with the USB host that is not part of the target application's main functionality. This is very useful during development, and also provides a communication channel to the PC host side. This can be used to create GUIs and other programs on the PC that communicate with the LaunchPad development kit.

Figure 6 shows the pathway of the backchannel UART. The backchannel UART eUSCI_A0 is independent of the UART on the 40-pin BoosterPack plug-in module connector eUSCI_A2.

On the host side, a virtual COM port for the application backchannel UART is generated when the LaunchPad development kit enumerates on the host. You can use any PC application that interfaces with COM ports, including terminal applications like Hyperterminal or Docklight, to open this port and communicate with the target application. You need to identify the COM port for the backchannel. On Windows PCs, Device Manager can assist.

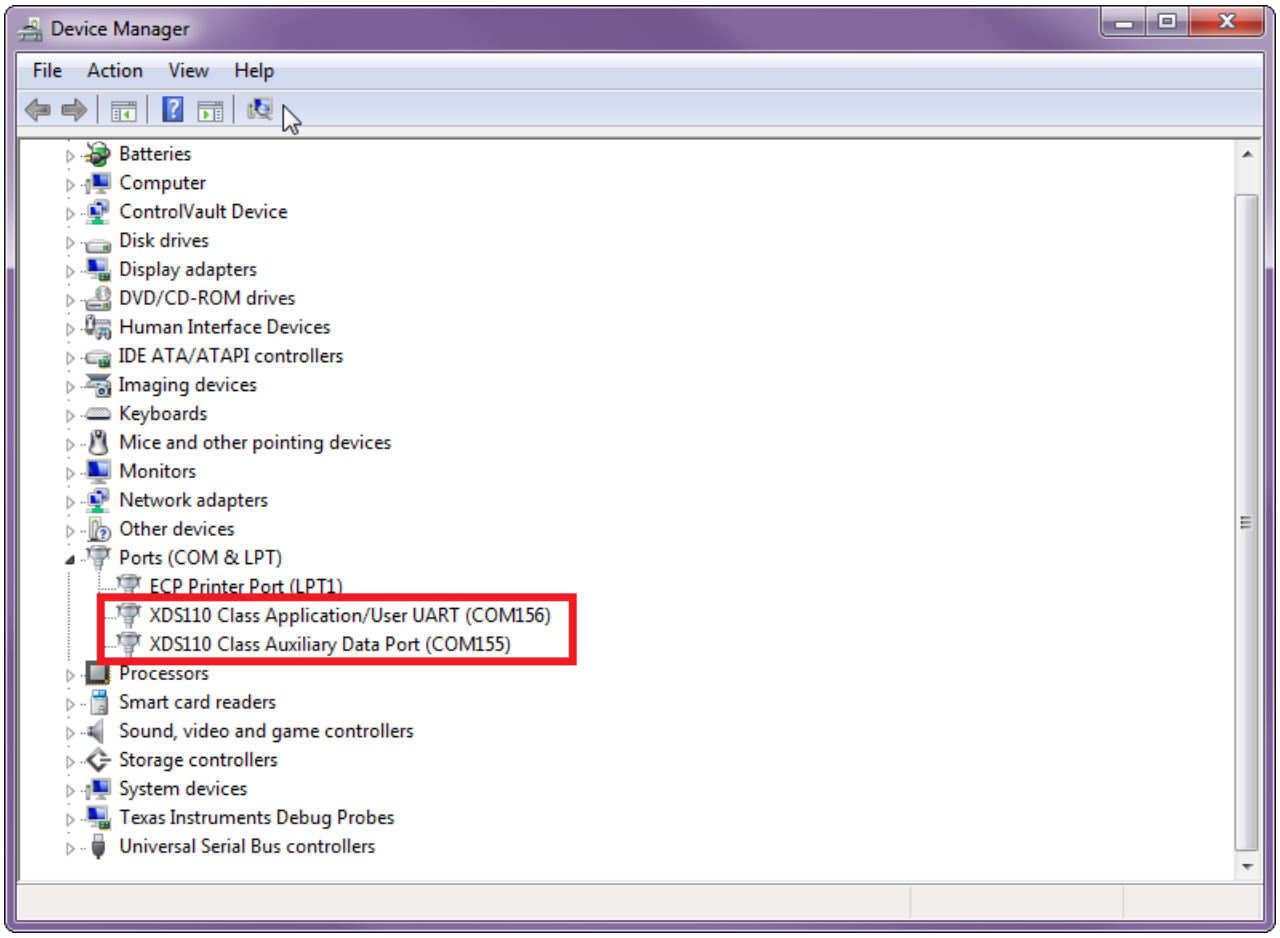

Figure 7. Application Backchannel UART in Device Manager

The backchannel UART is the XDS110 Class Application/User UART port. In this case, Figure 7 shows COM156, but this port can vary from one host PC to the next. After you identify the correct COM port, configure it in your host application according to its documentation. You can then open the port and begin communication to it from the host.

The XDS110-ET has a configurable baud rate; therefore, it is important that the PC application configures the baud rate to be the same as what is configured on the eUSCI_A0 backchannel UART.

2.3.3 Using an External Debug Probe Instead of the Onboard XDS110-ET

Many users have a specific external debug probe that they prefer to use, and may wish to bypass the XDS110-ET debug probe to program the MSP432 target MCU. This is enabled by jumpers on isolation block J101, and the connector J8. Using an external debug probe is simple, and full JTAG access is provided through J8.

1. Remove jumpers on the JTAG signals on the J101 isolation block, including RST, TMS, TCK, TDO, and TDI.
2. Plug any Arm debug probe into J8.
   a. J8 follows the Arm Cortex Debug Connector standard outlined in *Cortex-M Debug Connectors*.
3. Plug USB power into the LaunchPad development kit, or power it externally.
   a. Ensure that the jumpers across 3V3 and GND are connected if using USB power.
   b. External debug probes do not provide power, the VCC pin is a power sense pin.
   c. For more details on powering the LaunchPad development kit, see Section 2.4.

2.3.4 Using the XDS110-ET Debug Probe With a Different Target

The XDS110-ET debug probe on the LaunchPad development kit can interface to most Arm Cortex-M devices, not just the onboard target MSP432P410R device. This functionality is enabled by the J102 10-pin Cortex-M JTAG connector and a 10-pin cable, such as the FFSD-05-D-06.00-01-N (sold separately from the LaunchPad development kit).

Header J102 follows the Cortex-M Arm standard; however, pin 1 is not a voltage sense pin. The XDS110-ET outputs only 3.3-V JTAG signals. If another voltage level is needed, the user must provide level shifters to translate the JTAG signal voltages. Additionally, 3.3 V of output power can be sourced from the XDS110-ET when jumper JP102 is connected. This allows the XDS110-ET to power the external target at 3.3 V through pin 1. By default JP102 is not populated as it does not explicitly follow the standard.

1. Remove jumpers on the JTAG signals on the J101 isolation block, including RST, TMS, TCK, TDO, and TDI.
2. Plug the 10-pin cable into J102, and connect to an external target.
   a. J102 follows the Arm Cortex Debug Connector standard outlined in *Cortex-M Debug Connectors*.
3. Plug USB power into the LaunchPad development kit, or power it externally
   a. JTAG levels are 3.3 V ONLY.
   b. 3.3-V power can be sourced through J102 by shorting the JP102 jumper.

2.3.5 EnergyTrace+ Technology

EnergyTrace™ technology is an energy-based code analysis tool that measures and displays the application's energy profile and helps to optimize it for ultra-low power consumption.

MSP432 devices with built-in EnergyTrace+[CPU State] (or in short EnergyTrace+) technology allow real-time monitoring of internal device states while user program code executes.

EnergyTrace+ technology is supported on the LaunchPad development kit MSP432P401R device + XDS110-ET debug probe. EnergyTrace technology is available as part of TI's Code Composer Studio IDE. During application debug, additional windows are available for EnergyTrace technology.

To enable EnergyTrace technology, go to:

- Window > Preferences > Code Composer Studio > Advanced Tools > EnergyTrace™ Technology
- Check the Enable Auto-Launch on target connect box

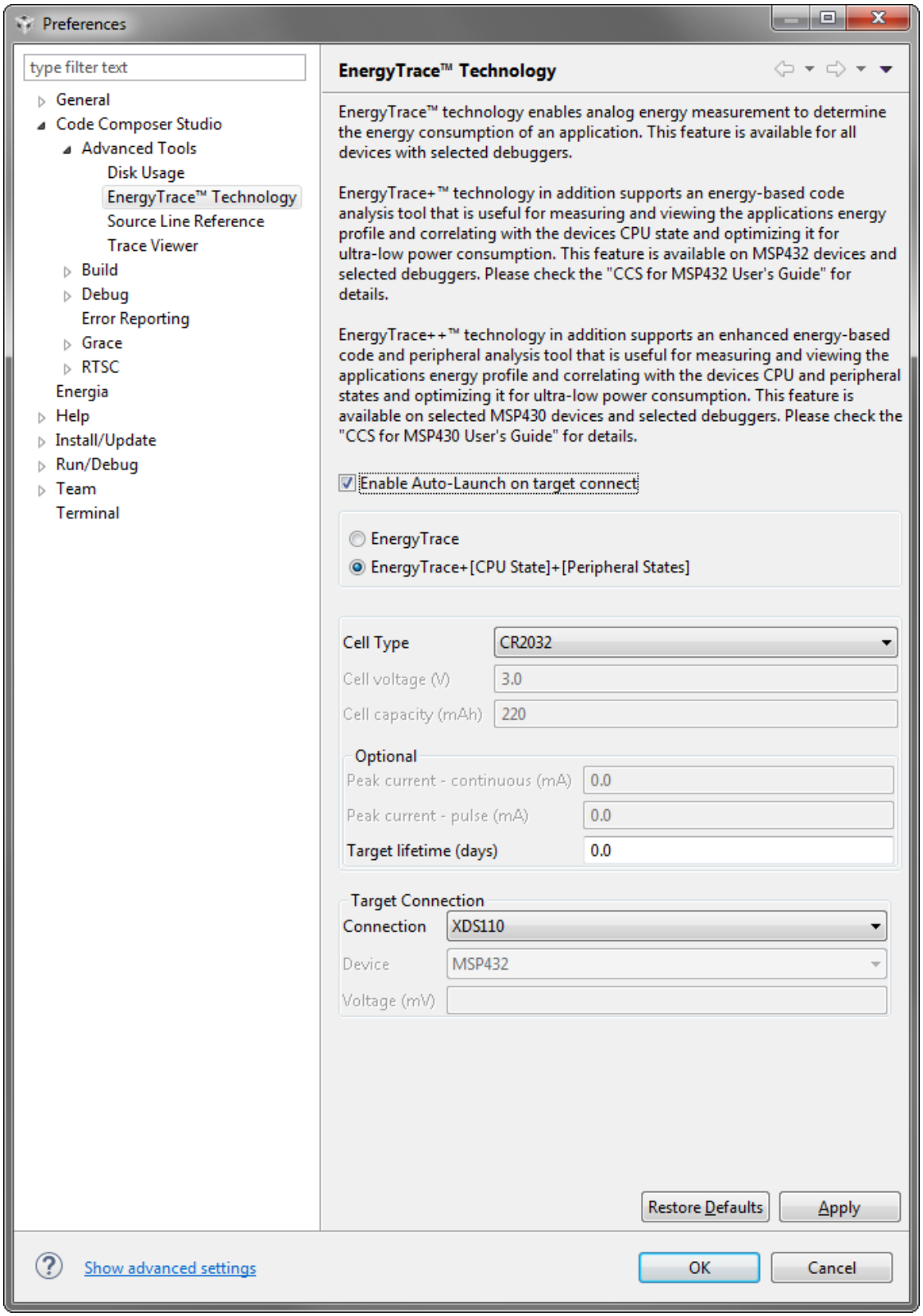

Figure 8. EnergyTrace™ Technology Preferences

Starting a debug session will now open EnergyTrace technology windows. These windows show energy, power, profile, and states to give the user a full view of the energy profile of their application.

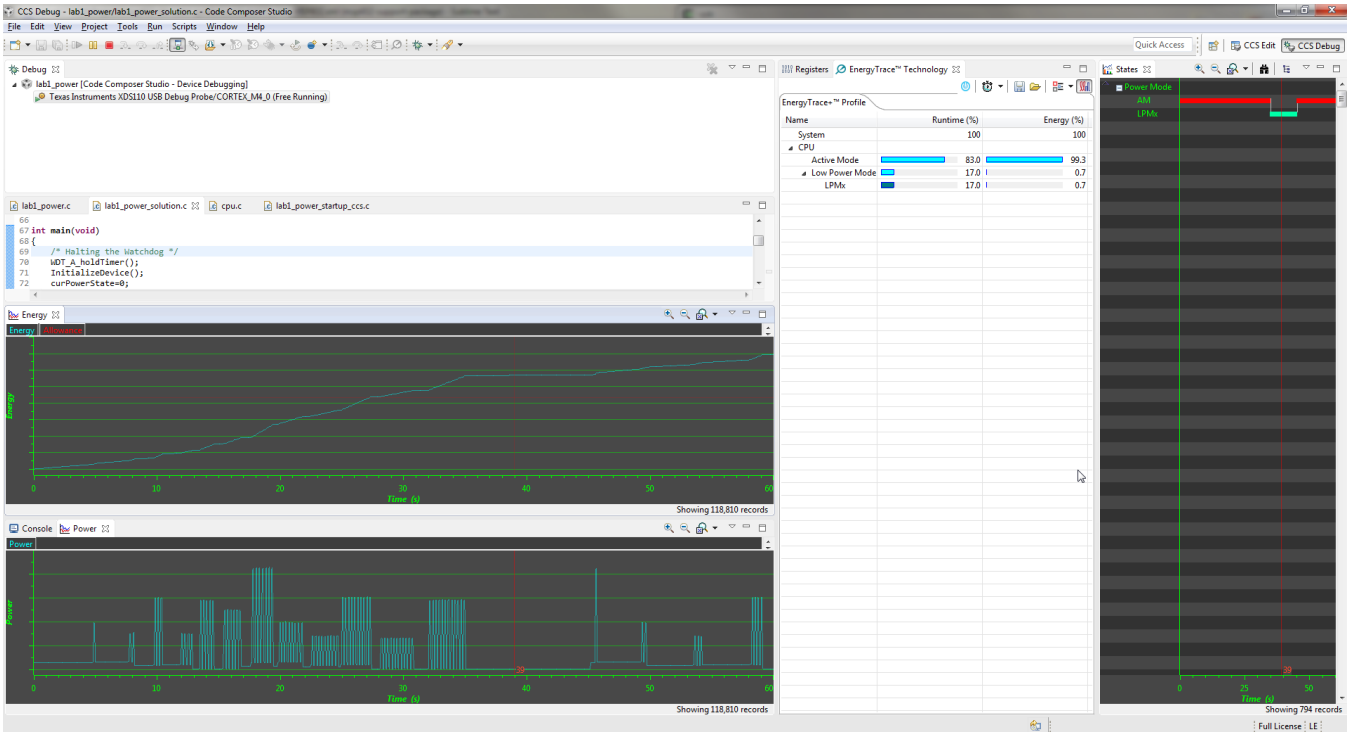

Figure 9. EnergyTrace™ Windows

This data allows the user to see exactly where and how energy is consumed in their application. Optimizations for energy can be quickly made for the lowest power application possible.

On the LaunchPad development kit, EnergyTrace technology measures the current that enters the target side of the LaunchPad development kit. This includes all BoosterPack plug-in modules plugged in, and anything else connected to the 3V3 power rail. For more information about powering the LaunchPad development kit, see Section 2.4.

For more information about EnergyTrace technology, see www.ti.com/tool/energytrace.

For more details and questions about setting up and using EnergyTrace technology with the MSP432P401R MCU, see the *Code Composer Studio™ IDE 7.1+ for SimpleLink™ MSP432™ Microcontrollers User's Guide*.

2.4 Power

The board was designed to accommodate various powering methods, including through the onboard XDS110-ET and from an external source or BoosterPack plug-in module.

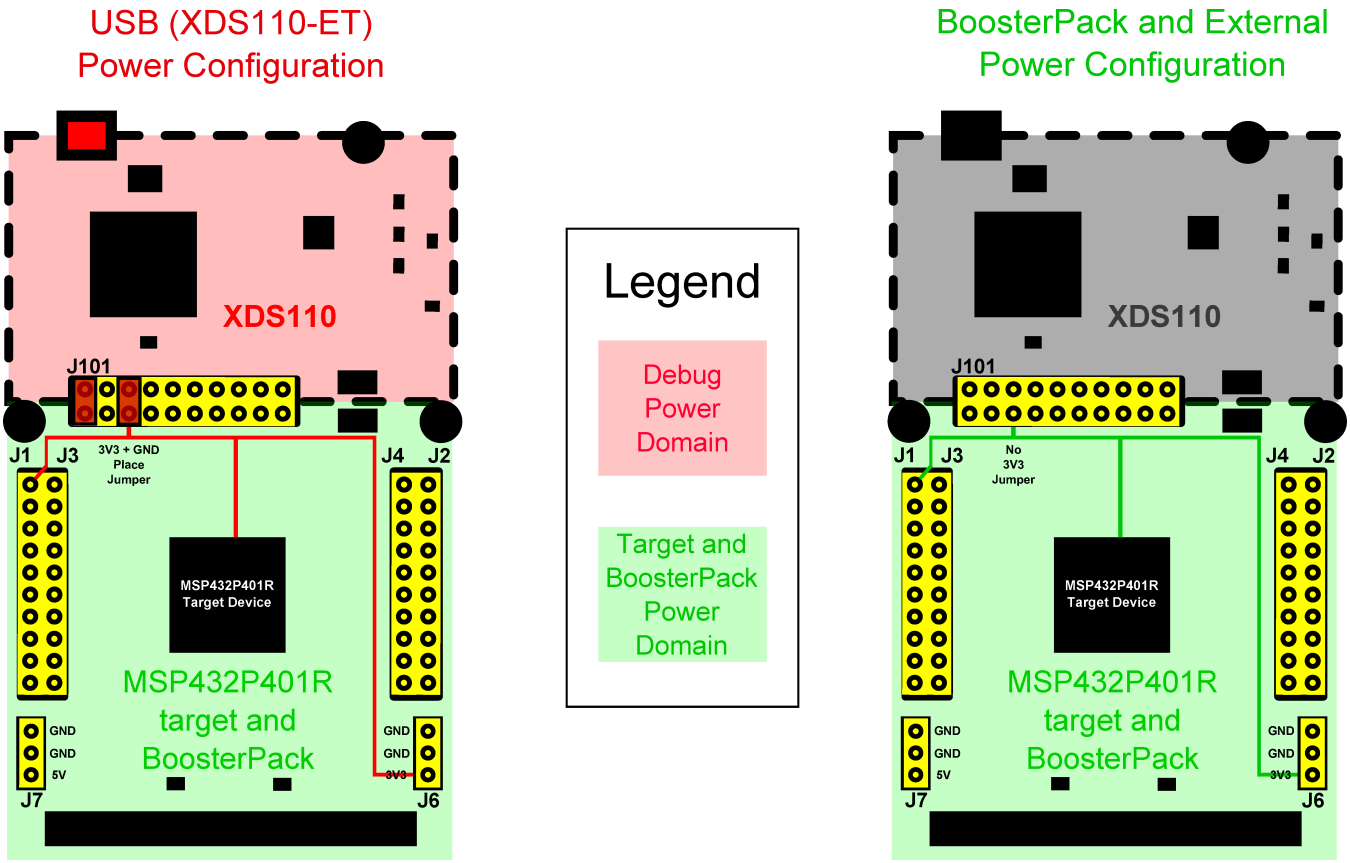

Figure 10. MSP-EXP432P401R Power Block Diagram

2.4.1 XDS110-ET USB Power

The most common power-supply scenario is from USB through the XDS110-ET debug probe. This provides 5-V power from the USB and also regulates this power rail to 3.3 V for XDS110-ET operation and 3.3 V to the target side of the LaunchPad development kit. Power from the XDS110-ET is controlled by the isolation block 3V3 jumper, ensure this jumper is connected for power to be provided to the target MCU side.

Under normal operation, the LDO on the XDS110-ET can supply up to 500 mA of current to the target side including any BoosterPack plug-in modules plugged in. However, when debugging and using the EnergyTrace technology tool, this current is limited to 75 mA total. Be aware of this current limitation when using EnergyTrace technology.

2.4.2 BoosterPack Plug-in Module and External Power Supply

Header J6 is present on the board to supply external power directly. It is important to comply with the device voltage operation specifications when supplying external power. The MSP432P401R has an operating range of 1.62 V to 3.7 V. More information can be found in *MSP432P401xx SimpleLink™ Mixed-Signal Microcontrollers*.

2.5 Measure Current Draw of MSP432 MCU

To measure the current draw of the MSP432P401R MCU, use the 3V3 jumper on the jumper isolation block. The current measured includes the target device and any current drawn through the BoosterPack plug-in module headers.

To measure ultra-low power, follow these steps:

1. Remove the 3V3 jumper in the isolation block, and attach an ammeter across this jumper.
2. Consider the effect that the backchannel UART and any circuitry attached to the MSP432P401R may have on current draw. Disconnect these at the isolation block if possible, or at least consider their current sinking and sourcing capability in the final measurement.
3. Make sure there are no floating input I/Os. These cause unnecessary extra current draw. Every I/O should either be driven out or, if it is an input, should be pulled or driven to a high or low level.
4. Begin target execution.
5. Measure the current. Keep in mind that if the current levels are fluctuating, it may be difficult to get a stable measurement. It is easier to measure quiescent states.

For a better look at the power consumed in the application, use EnergyTrace+ Technology. EnergyTrace+ Technology allows the user to see energy consumed as the application progresses. For more details about EnergyTrace+ Technology, see Section 2.3.5.

2.6 Clocking

The MSP-EXP432P401R provides external clocks in addition to the internal clocks in the device.

- Q1: 32-kHz crystal (LFXTCLK)
- Q2: 48-MHz crystal (HFXTCLK)

The 32-kHz crystal allows for lower LPM3 sleep currents and a higher-precision clock source than the default internal 32-kHz REFOCLK. Therefore, the presence of the crystal allows the full range of low-power modes to be used.

The 48-MHz crystal allows the device to run at its maximum operating speed for MCLK and HSMCLK.

The MSP432P401R device has several internal clocks that can be sourced from many clock sources. Most peripherals on the device can select which of the internal clocks to use to operate at the desired speed.

The internal clocks in the device default to the configuration listed in Table 2.

Table 2. Default Clock Operation

Clock	Default Clock Source	Default Clock Frequency	Description
MCLK	DCO	3 MHz	Master Clock Sources CPU and peripherals
HSMCLK	DCO	3 MHz	Subsystem Master Clock Sources peripherals
SMCLK	DCO	3 MHz	Low-speed subsystem master clock Sources peripherals
ACLK	LFXT (or REFO if no crystal present)	32.768 kHz	Auxiliary clock Sources peripherals
BCLK	LFXT (or REFO if no crystal present)	32.768 kHz	Low-speed backup domain clock Sources LPM peripherals

For more information about configuring internal clocks and using the external oscillators, see the *MSP432P4xx SimpleLink™ Microcontrollers Technical Reference Manual*.

2.7 BoosterPack Plug-in Module Pinout

The MSP-EXP432P401R LaunchPad development kit adheres to the 40-pin LaunchPad development kit pinout standard. A standard was created to aid compatibility between LaunchPad development kit and BoosterPack plug-in module tools across the TI ecosystem.

The 40-pin standard is compatible with the 20-pin standard that is used by other LaunchPad development kits like the MSP-EXP430FR4133. This allows some subset of functionality of 40-pin BoosterPack plug-in modules to be used with 20-pin LaunchPad development kits.

While most BoosterPack plug-in modules are compliant with the standard, some are not. The MSP-EXP432P401R LaunchPad development kit is compatible with all 20-pin and 40-pin BoosterPack plug-in modules that are compliant with the standard. If the reseller or owner of the BoosterPack plug-in module does not explicitly indicate compatibility with the MSP-EXP432P401R LaunchPad development kit, compare the schematic of the candidate BoosterPack plug-in module with the LaunchPad development kit to ensure compatibility. Keep in mind that sometimes conflicts can be resolved by changing the MSP432P401R device pin function configuration in software. More information about compatibility can also be found at www.ti.com/launchpad.

Figure 11 shows the 40-pin pinout of the MSP-EXP432P401R LaunchPad development kit.

Note that software configuration of the pin functions plays a role in compatibility. The MSP-EXP432P401R LaunchPad development kit side of the dashed line in Figure 11 shows all of the functions for which the MSP432P401R device's pins can be configured. This can also be seen in the MSP432P401R data sheet. The BoosterPack plug-in module side of the dashed line shows the standard. The MSP432P401R function whose color matches the BoosterPack plug-in module function shows the specific software-configurable function by which the MSP-EXP432P401R LaunchPad development kit adheres to the standard.

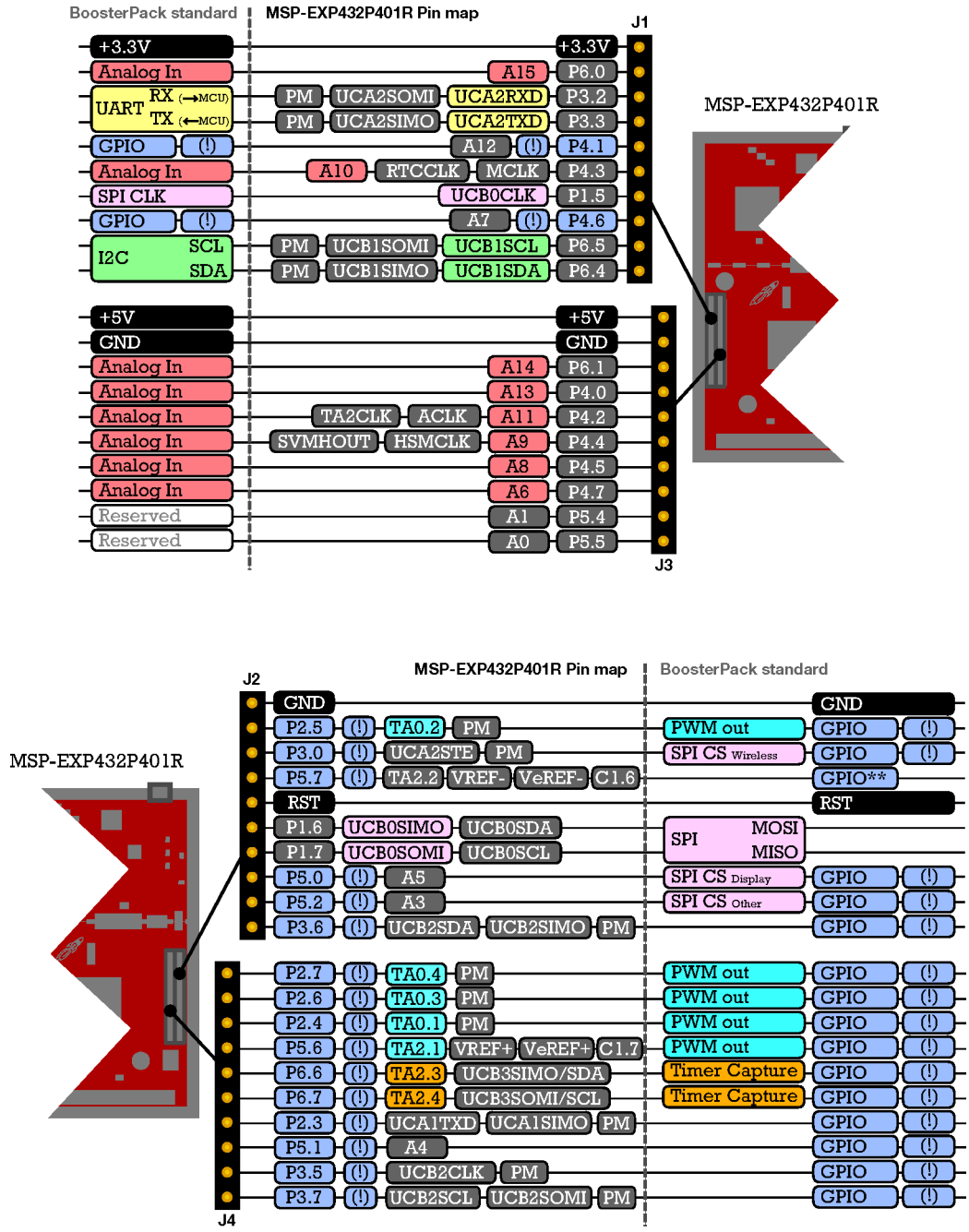

Figure 11. LaunchPad™ Development Kit to BoosterPack™ Plug-in Module Connector Pinout

2.8 Design Files

2.8.1 Hardware Design Files

Schematics can be found in Section 6. All design files including schematics, layout, bill of materials (BOM), Gerber files, and documentation are available in the MSP-EXP432P401R Hardware Design Files.

2.9 Hardware Change Log

Table 3 lists the hardware revisions.

Table 3. Hardware Change Log

PCB Revision	Date	Description	MSP432P401R Device Revision	MSP-EXP432P401R Hardware and Software Download Version (for Downloading Zip Packages Only)
Rev 1.0	March 2015	Preproduction release	XMS432P401R Rev B	2_00_00_03
Rev 2.0	June 2016	Production silicon release	XMS432P401R Rev C or MSP432P401R Rev C (check device markings to determine version)	3_00_00_03
Rev 2.1	January 2018	Production silicon release with updated silkscreen for CE compliance and latest LaunchPad standards	MSP432P401R Rev C	3_00_00_03

2.9.1 MSP-EXP432P401R Rev 1.0 (Black) LaunchPad Development Kit

As shown in Table 3, this was the initially released LaunchPad development kit, with XMS432P401R Rev B silicon. Connecting any debugger to this version of the MSP432 MCU will generate a warning telling the user to update their silicon. TI will continue to support this board for the foreseeable future. However, upgrading to the latest LaunchPad development kit gets you all the silicon upgrades and the final production version of Driver Library in ROM.

In addition to the hardware being different, the device revision also changes with the LaunchPad development kit revisions. Because of this, you must download a software package that matches your exact hardware. The software example files for older versions of the LaunchPad development kit are available from MSP-EXP432P401R Software Examples – navigate to previous release versions according to Table 3.

2.9.2 MSP-EXP432P401R Rev 2.0 (Red) LaunchPad Development Kit

As shown in Table 3, this is the updated LaunchPad development kit for the production silicon, with XMS432P401R Rev C or MSP432P401R Rev C silicon. In addition to the latest silicon, several updates were made to the LaunchPad development kit hardware to enhance the user experience (see Section 2.9.2.1).

2.9.2.1 MSP-EXP432P401R Updates

From the perspective of the board, all of the changes are aesthetic or make the kit easier to use (see Figure 12). For example, moving the user buttons to the side of the board makes them easier to reach when you have a BoosterPack plug-in module connected to the top of the LaunchPad development kit. The button placement has changed, but the physical button connections are the same. The most significant change is the addition of an extra 10-pin Arm JTAG connector. This connector lets you use the LaunchPad development kit as a stand-alone XDS110 debug probe.

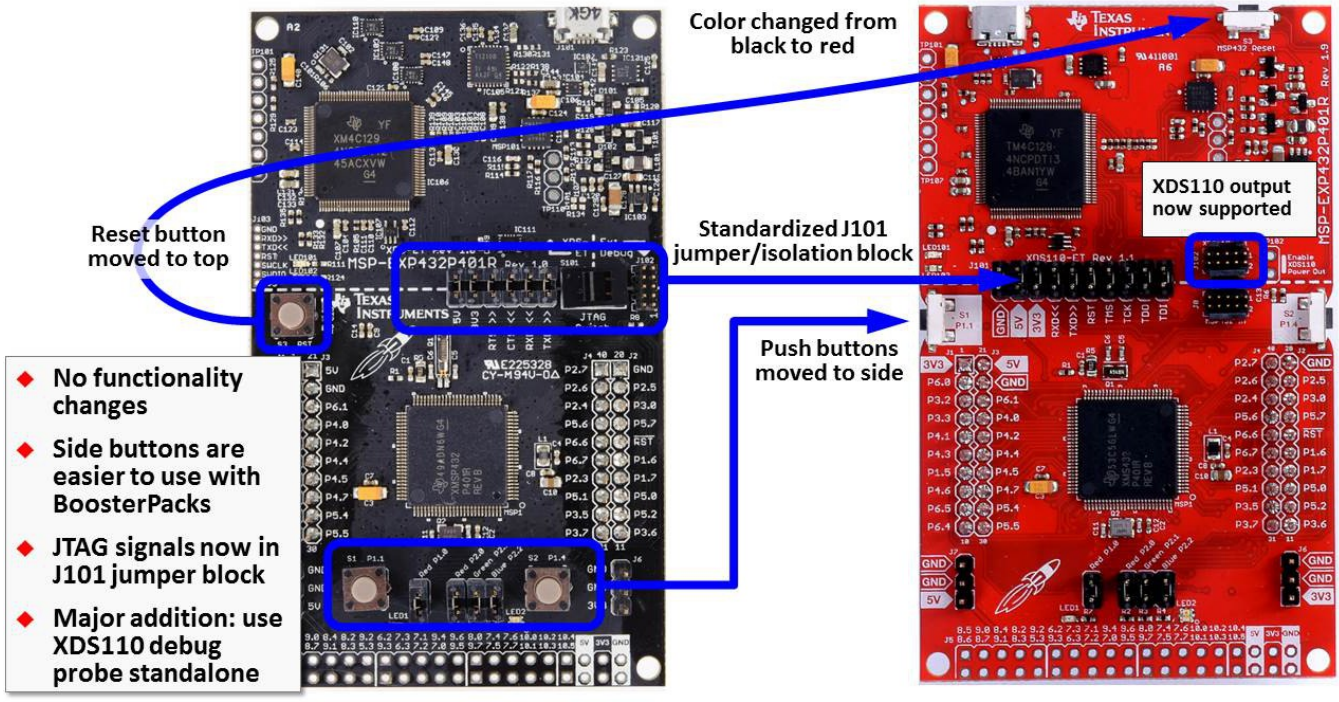

Figure 12. Differences Between Rev 1.0 (Black) and Rev 2.0 (Red)

2.9.2.2 MSP432P401R Device Revision Differences

The primary MSP432P401R silicon differences are the differences between Rev. B and Rev. C devices. For details of the differences, see *Moving From Evaluation to Production With SimpleLink™ MSP432P401x Microcontrollers*.

The first shipments of Rev 2.0 (Red) LaunchPad development kit have XMS432P401R Rev C. preproduction silicon before the final production version of MSP432P401R Rev C. silicon is released. Which device is on a particular LaunchPad development kit can be determined by looking at the markings on the MSP432P401R device. The XMS version have a marking of "XMS" instead of "MSP". For details on the differences between the preproduction and production silicon, see *XMS432P401x Rev. C Preproduction Advisories*.

To work around Advisory 1 in the document above, and as a general way to update to the latest device BSL, TI provides a utility to download the latest BSL. This utility is available inside of TI Resource Explorer (see Figure 13). Alternatively, the BSL can be updated by running Program_MSP432_BSL.bat in the source files for MSP432P401R BSL update.

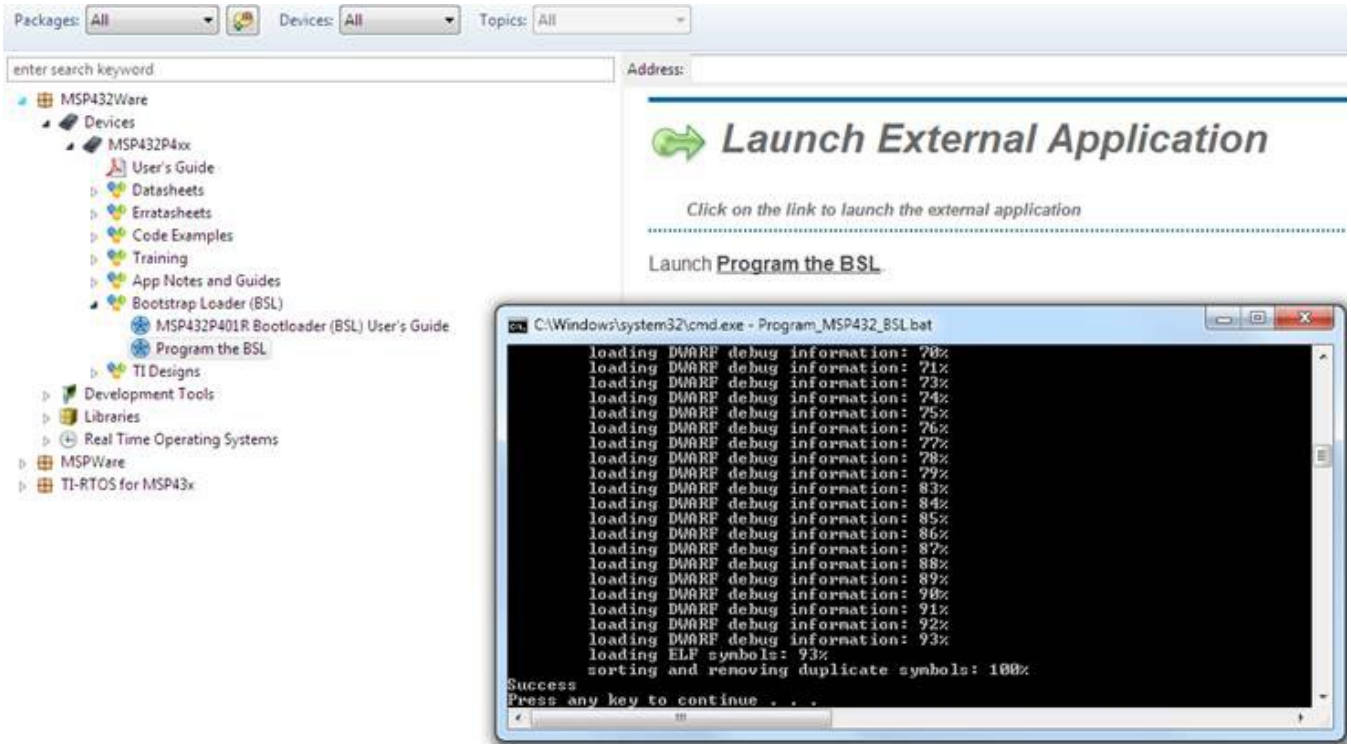

Figure 13. BSL Update Utility in TI Resource Explorer

3 Software Examples

Table 4 lists the software examples that are included with the MSP-EXP432P401R LaunchPad development kit. These examples can be downloaded with the SimpleLink MSP432 SDK.

Table 4. Software Examples

Demo Name	BoosterPack Required	Description	More Details
Out-of-Box Software Example	N/A	The out-of-box demo programmed on the LaunchPad development kit from the factory.	Section 3.1
BOOSTXL-K350QVG-S1 Graphics Library Example	BOOSTXL-K350SVG-S1	A simple example showing how to use MSP Graphics Library (grlib) to display graphics primitives and images and implement touchscreen functionality	Section 3.2
430BOOST-SHARP96 Graphics Library Example	430BOOST-SHARP96	A simple example showing how to use MSP Graphics Library (grlib) to display graphics primitives and images	Section 3.3
BOOSTXL-BATPAKMKII_FuelGauge_ MSP432P401R	BOOSTXL-BATPAKMKII	Demonstrates how to initialize bq27441-G1 fuel gauge configurations and how to control and read data registers	Section 3.4
BOOSTXL-SENSORS_SensorGUI_ MSP432P401R	BOOSTXL-SENSORS	Demonstrates how to sample data from the five onboard digital sensors and communicate that over UART in a JSON payload	Section 3.5
BOOSTXL-SENSORS_TI-RTOS_SensorGUI_ MSP432P401R	BOOSTXL-SENSORS	Demonstrates how to sample data from the five onboard digital sensors and communicate that over UART in a JSON payload using TI-RTOS	Section 3.6

To use any of the software examples with the LaunchPad development kit, you must have an integrated development environment (IDE) that supports the MSP432P401R device (see Table 5).

Table 5. IDE Minimum Requirements for MSP-EXP432P401R

Code Composer Studio™ IDE	IAR Embedded Workbench® IDE	Keil® µVision® MDK-Arm
v6.1 or later	v7.10 or later	v5 or later

For more details on how to get started quickly and where to download the latest TI, IAR, and Keil IDEs, see Section 4.

3.1 Out-of-Box Software Example

This section describes the functionality and structure of the out-of-box software that is preloaded on the EVM. The source code can be found in the SimpleLink MSP432 SDK.

The out-of-box software extends a basic blink LED example to allow users to control the blink rate and color of an RGB LED on the MSP432 LaunchPad development kit.

This software example is created to work with TI-RTOS or FreeRTOS, with a POSIX layer to make it compatible with both kernels. There are many advantages to using a Real Time Operating System over bare-metal code. An RTOS manages many aspects of the system, which allows a developer to focus on the application. Typically, an RTOS is used when the application needs to do more than a few simple actions.

This Out-of-Box example is intentionally designed to be much simpler than what would typically be expected of an RTOS project. The purpose of this design is to provide the user with a very simple and straight-forward example of how to use an RTOS kernel. Using an RTOS kernel in this example requires more overhead in memory than bare-metal code, but it allows the user more modularity and flexibility when adding peripherals or modules. This becomes particularly useful with more complex systems such as when using connectivity devices. For more information, go to the MSP432 SimpleLink Academy training on TI Resource Explorer.

3.1.1 Operation

Upon powering up the out-of-box demo, the RGB LED2 blinks red at 1 Hz. Switch S1 can be tapped repeatedly at a constant rate to set the blink frequency of LED2. Switch S2 cycles LED2 through four different color settings: Red, Green, Blue, and random RGB color. Each color setting retains its own blink frequency.

A PC GUI accompanies the out-of-box demo to allow user to set the color and blink rate of the RGB LED. Connect the LaunchPad development kit to a computer using the included USB cable. The out-of-box GUI can be opened from within CCS using the TI Resource Explorer: Software > SimpleLink MSP432 SDK > Development Tools > MSP-EXP432P401R > Demos > outOfBox_msp432p401r, and launch the Out of Box Experience GUI.

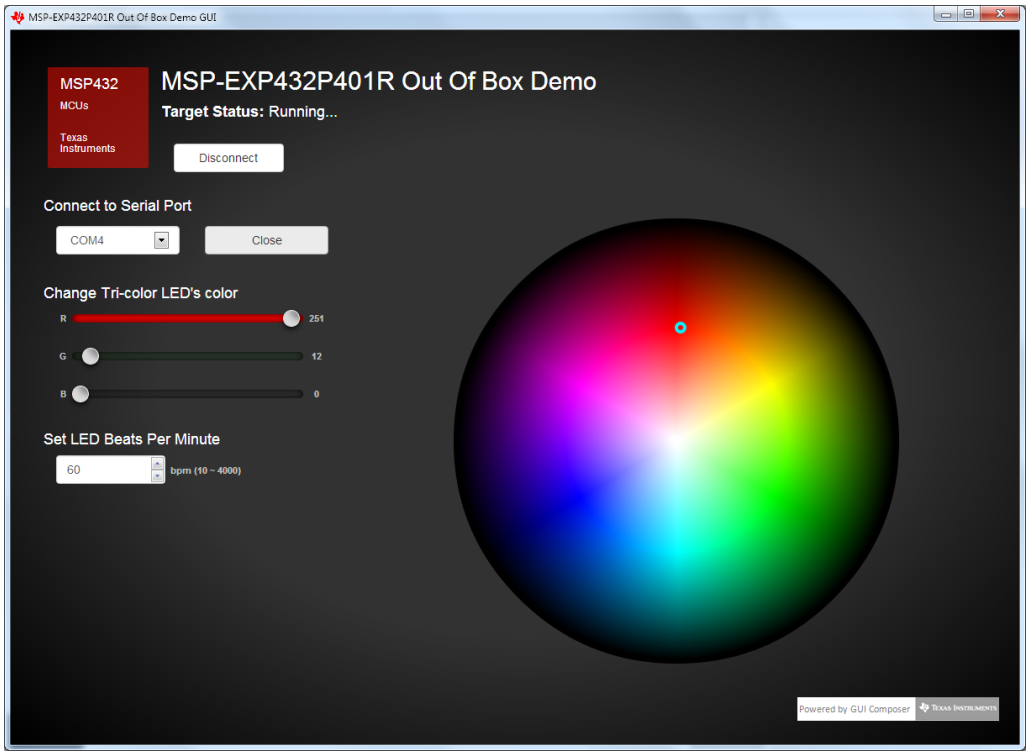

Figure 14. Out-of-Box GUI Running Locally

The GUI can also run directly from the TI Cloud Tools (see Section 4.1.2).

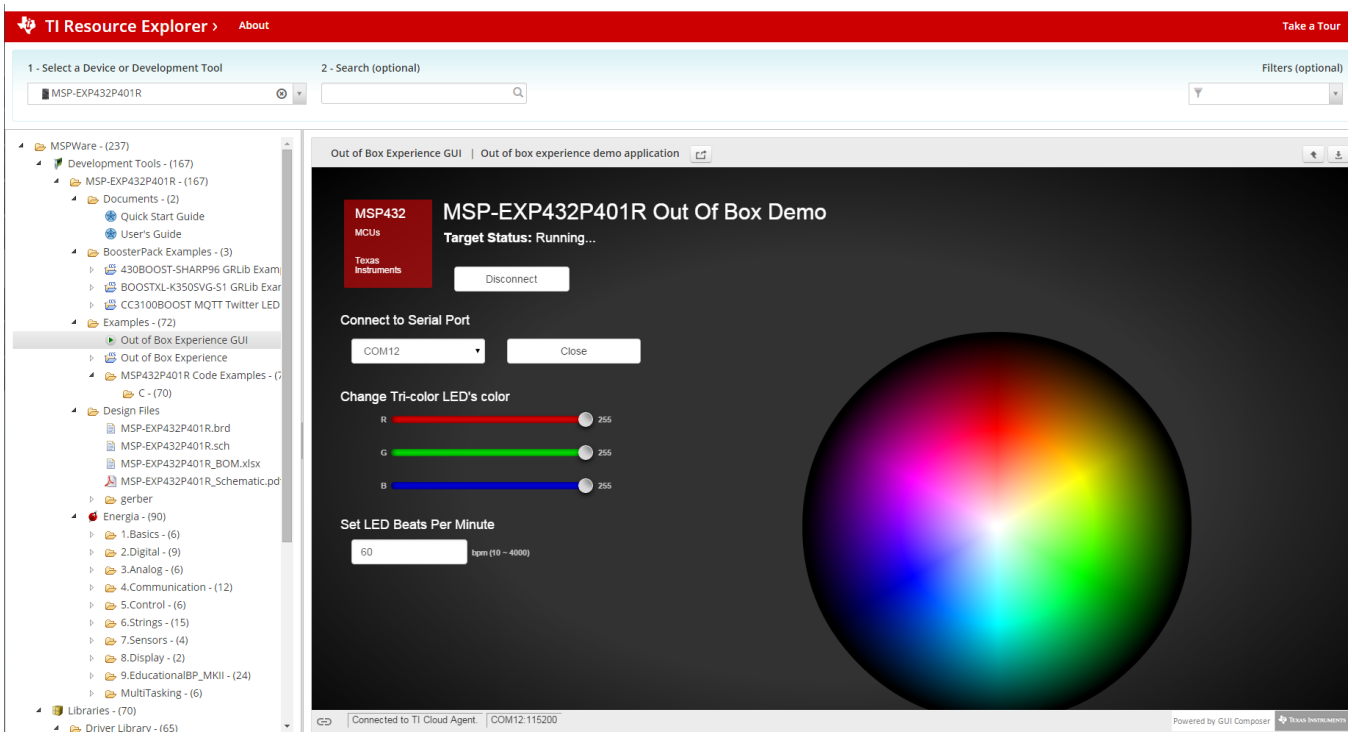

Figure 15. Out-of-Box GUI Running From TI Cloud Tools

Click on the Connect button to connect to the LaunchPad development kit then open the serial COM port. Once the connection has been established and the GUI indicates, "Target Status: Running…," you can use the color wheel or the Red, Green, and Blue color sliders to set the color of the LaunchPad development kit RGB LED. Changing the LED Beats Per Minute input box sets the RGB LED blink rate.

3.2 BOOSTXL-K350QVG-S1 Graphics Library Example

This software is available in the SimpleLink MSP432 SDK (see Section 4.3).

The demo shows how to use the *MSP Graphics Library*, or "grlib," in a project with the Kentec display. This demo shows the user how to enable the touch screen, create buttons, and use graphics primitives including colors and images.

The program begins by calibrating the touch screen. There is a routine that detects the four corner coordinates to determine if an eligible rectangle boundary is formed. If the calibration was incorrect, a message displays on the screen to indicate that the calibration failed. When successful, the calibration provides a reference for all button presses throughout the rest of the program.

The next step is to select the mode of the program: display primitives or images. Each mode simply cycles through without user interaction to demonstrate features of the display. In the graphics primitives mode, the following primitives are shown:

- Pixels
- Lines
- Circles
- Rectangles
- Text

The application is heavily commented to ensure it is very clear how to use the grlib APIs. The above primitives are shown as well as the underlying concepts of grlib including background and foreground colors, context, fonts, opacity, and more.

The images mode shows the drawing of a few different images both compressed and uncompressed. Image compression can have a big impact to drawing speeds for simple images. To draw images with the MSP Graphics Library, they must first be converted into the right file format. These files can be generated by the Image Reformer tool that comes packaged with grlib. Launch this tool from the grlib folder or directly from TI Resource Explorer.

- File Path: <grlib root>\utils\image-reformer\imagereformer.exe

The Image Reformer tool allows you to import images and output into grlib specific files to add to your grlib project. Image Reformer does not manipulate any images (such as color modifications, rotation, or cropping), any image manipulation must be done before importing into the Image Reformer tool. More information about MSP grlib and the Image Reformer tool can be found in *Design Considerations When Using the MSP430 Graphics Library*.

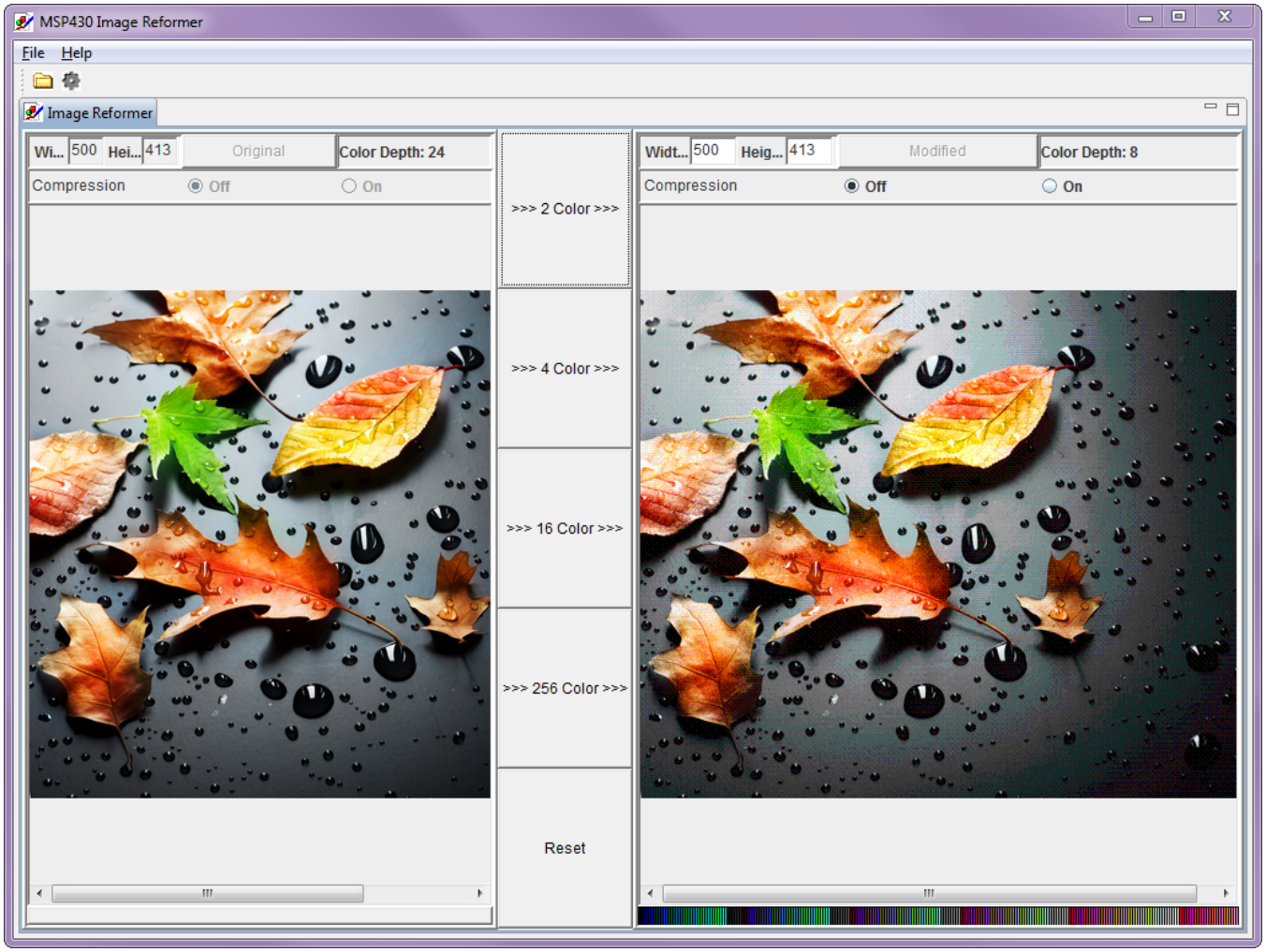

Figure 16. Importing and Converting an Image With MSP Image Reformer

3.3 430BOOST-SHARP96 Graphics Library Example

This software example is similar to the BOOSTXL-K350QVG-S1 Graphics library example. It shows how to use the *MSP Graphics Library*, or "grlib," in a project with the Sharp 96×96 display. The Sharp 96×96 display BoosterPack plug-in module does not support touch or color, it is a simple monochrome LCD. It is a great LCD for ultra-low power display applications and has a unique mirrored pixel display.

This demo cycles screens without user interaction to show simple graphics primitives.

- Pixels
- Lines
- Circles
- Rectangles
- Text
- Images

This demo introduces the functions to configure grlib such as initialization, color inversion, and using foreground and background colors properly.

3.4 BOOSTXL-BATPAKMKII_FuelGauge_MSP432P401R

This section describes the functionality and structure of the BOOSTXL-BATPAKMKII_FuelGauge_MSP432P401R demo that is included in the SimpleLink MSP432 SDK (see Section 4.3).

3.4.1 Source File Structure

The project is split into multiple files (see Table 6). This makes it easier to navigate and reuse parts of it for other projects.

Table 6. Source File and Folders

Name	Description
Library: driverlib	Device driver library (MSP432DRIVERLIB)
startup_msp432p401r.c	MSP432™ MCU family interrupt vector table for CGT
HAL_BQ27441.c	Driver for communicating with the bq27441-G1 fuel gauge
HAL_I2C.c	Board specific support driver for I^2C communication
HAL_UART.c	Board specific driver for UART communication through Application/User UART
main.c	The main function of the demo, global variables, and more

3.4.2 Running the Fuel Gauge Example

After the compiling and loading the BOOSTXL-BATPAKMKII_FuelGauge_MSP432P401R project or downloading the prebuilt firmware binary onto the MSP-EXP432P401R LaunchPad development kit, follow the steps below to run the demo firmware.

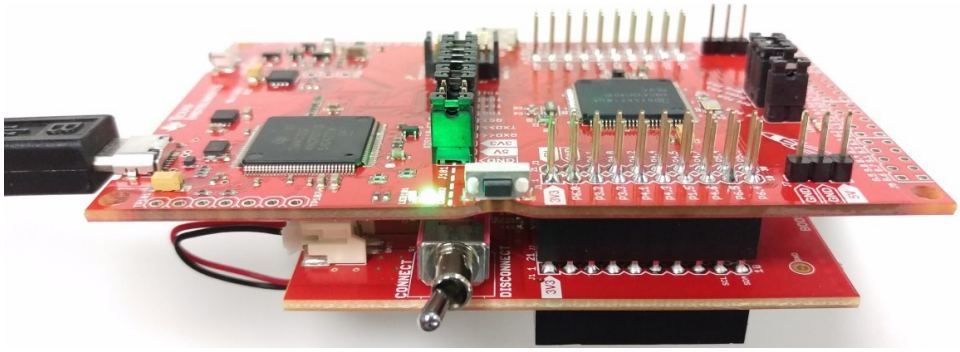

Figure 17. Hardware Setup and Connections

1. Attach the BOOSTXL-BATPAKMKII Battery BoosterPack plug-in module to the LaunchPad development kit.
2. Flip the switch to the "ON" position on the side of the BOOSTXL-BATPAKMKII Battery BoosterPack plug-in module.
3. Connect the MSP-EXP432P401R LaunchPad development kit to a computer via micro-USB cable.
4. Launch a serial terminal application and connect to the COM port for "XDS110 Class Application/User UART" at 115200 baud rate (see Figure 18 and Figure 19).

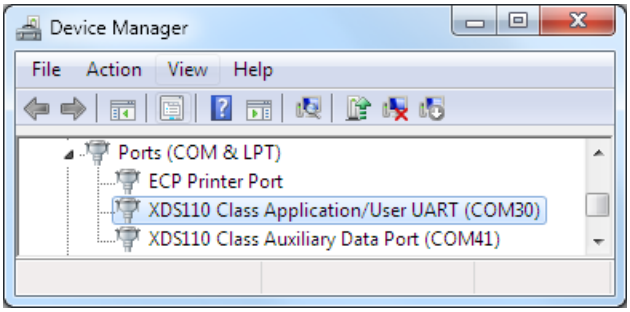

Figure 18. Determine COM Port Number Using Device Manager on Windows

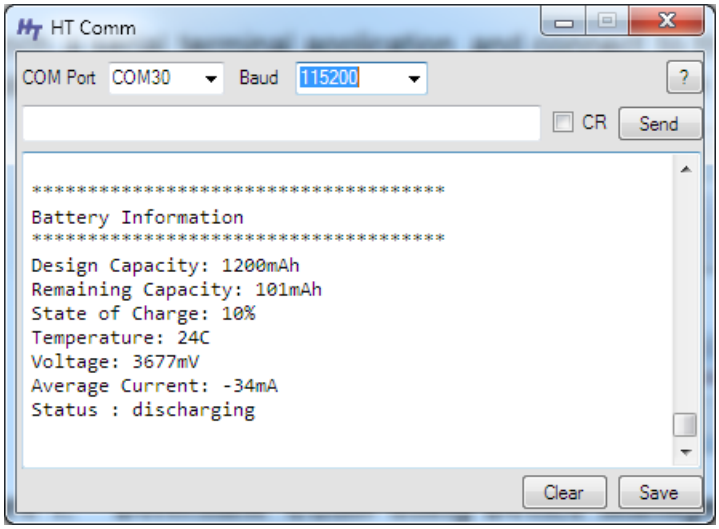

Figure 19. Example Serial Terminal Configuration

5. Press the reset button on the MSP-EXP432P401R LaunchPad development kit.
6. Observe serial data displaying Fuel Gauge configuration and Battery Information (see Figure 20).

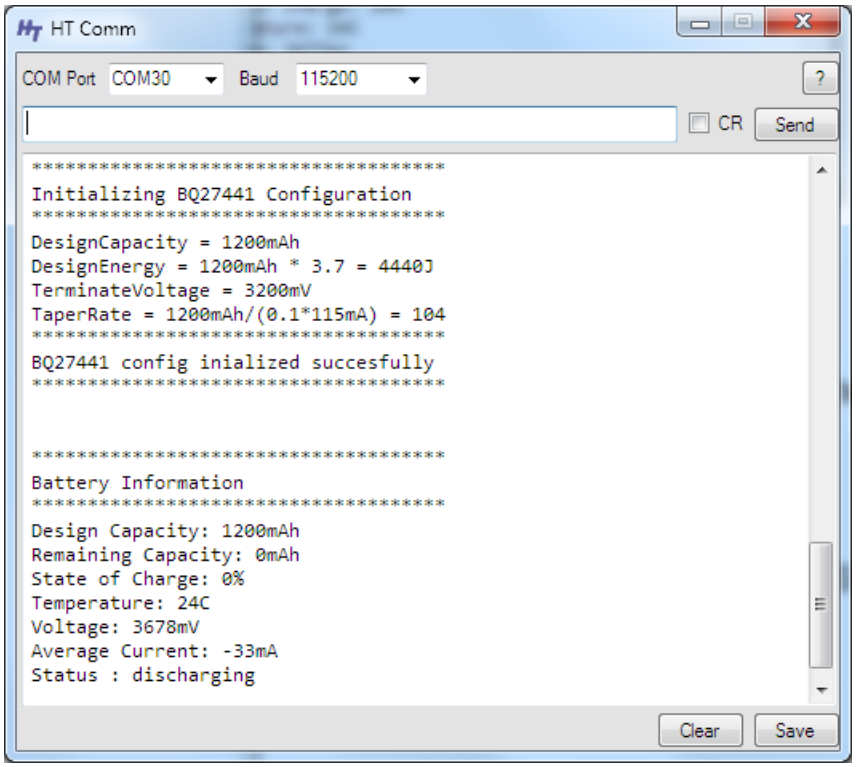

Figure 20. Snapshot of Serial Terminal Connected to Running Fuel Gauge Demo

3.4.3 Firmware Overview

See the *Quick Start Guide for bq27441-G1* and the *bq27441-G1 Technical Reference Manual* for detailed description of the Battery Fuel Gauge usage.

The demo program begins by initializing a number of configuration parameters in the bq27441-G1 to match the target battery. Four important parameters are Design Capacity, Design Energy, Terminate Voltage, and Taper Rate. Values are determined based on the target battery properties and bq27441-G1 documentation.

Next, the host MSP432P401R MCU clears the BIE (Battery Insert Enable) bit in the fuel gauge operation configuration register. When BIE is cleared, the battery detection relies on the host to issue a BAT_INSERT subcommand to indicate battery presence, bypassing the J6 BIN jumper on the BOOSTXL-BATPAKMKII BoosterPack plug-in module that the fuel gauge relies on for battery detection by default when BIE is set (J6 shorted = battery inserted; J6 open = battery removed). This is done to ensure that the demo application works whether or not J6 is connected.

In end user applications, a switch or the host MCU is more likely to control the BIN state of the fuel gauge depending on battery connection. However, this is not implemented on the BoosterPack plug-in module and a jumper is used to manually toggle between battery insertion and removal.

When the bq27441-G1 has been configured properly, the main loop repeatedly reads back DESIGN_CAPACITY, REMAINING_CAPACITY, STATE_OF_CHARGE, TEMPERATURE, VOLTAGE, and AVERAGE_CURRENT from the fuel gauge. Results are formatted and transmitted through Application/User UART.

3.5 BOOSTXL-SENSORS_SensorGUI_MSP432P401R

The Sensors BoosterPack kit (BOOSTXL-SENSORS) is an easy-to-use plug-in module for adding digital sensors to your LaunchPad development kit design. The SimpleLink MCU LaunchPad development kit allows developers to use this BoosterPack plug-in module to start developing sensor applications using the onboard gyroscope, accelerometer, magnetometer, pressure, ambient temperature, humidity, ambient light, and infrared temperature sensors. For information on the Out-of-Box experience and how to use the BOOSTXL-SENSORS BoosterPack plug-in module, see the *BOOSTXL-SENSORS BoosterPack™ Plug-in Module User's Guide*.

3.6 BOOSTXL-SENSORS_TI-RTOS_SensorGUI_MSP432P401R

This section describes the functionality structure of the BOOSTXL-SENSORS_TI_RTOS_SensorGUI_MSP432P401R demo that is included in the SimpleLink MSP432 SDK (see Section 4.3).

This example requires TI-RTOS MSP43x version 2_16_01_14 to be installed in CCS.

More information on the use of TI-RTOS can be found in the TI-RTOS user's guides, available in the TI-RTOS tool folder.

3.6.1 Source File Structure

Table 7 lists the source files and folders.

Table 7. Source Files and Folders

Name	Description
OS: TI-RTOS	Real-time operating system using TI-RTOS kernel
Library: driverlib	Device driver library (MSP432DRIVERLIB)
bme280.c	Driver for communicating with the environmental sensor
bme280_support.c	Support driver for communicating with the environmental sensor
bmi160.c	Driver for communicating with the IMU and magnetometer sensors
bmi160_support.c	Support driver for communicating with the IMU and magnetometer sensors
MSP_EXP432P401R.c	Driver for setting up board specific items (for example, I^2C and UART)
main.c	The demo's main function, tasks, semaphores, global variables, and more
opt3001.c	Driver for communicating with the ambient light sensor
tmp007.c	Driver for communicating with the infrared temperature sensor

3.6.2 Working With the GUI

Collaboration with the Sensor GUI is identical to , except for programming the device directly from the GUI. The .out file located within the GUI is specific to the BOOSTXL-SENSORS_SensorGUI_MSP432P401R example project. To download the program, you must use a separate IDE, such as CCS or IAR, and the BOOSTXL-SENSORS_TI_RTOS_SensorGUI_MSP432P401R source code in the SimpleLink MSP432 SDK.

4 Resources

4.1 Integrated Development Environments

Although the source files can be viewed with any text editor, more can be done with the projects if they're opened with a development environment like Code Composer Studio (CCS) IDE, Keil µVision, IAR Embedded Workbench, or Energia.

4.1.1 SimpleLink MSP432 SDK

The MSP432P401R device is part of the SimpleLink microcontroller (MCU) platform, which consists of Wi-Fi®, *Bluetooth*® low energy, Sub-1 GHz, and host MCUs. All share a common, easy-to-use development environment with a single core software development kit (SDK) and rich tool set. A one-time integration of the SimpleLink platform lets you add any combination of devices from the portfolio into your design. The ultimate goal of the SimpleLink platform is to achieve 100 percent code reuse when your design requirements change. For more information, visit www.ti.com/simplelink.

4.1.2 TI Cloud Development Tools

TI's Cloud-based software development tools provide instant access to SimpleLink SDK content and a web-based IDE.

4.1.2.1 TI Resource Explorer Cloud

TI Resource Explorer Cloud provides a web interface for browsing examples, libraries and documentation found in SimpleLink SDK without having to download files to your local drive.

Try the TI Resource Explorer Cloud now at dev.ti.com.

4.1.2.2 Code Composer Studio™ Cloud IDE

Code Composer Studio Cloud is a web-based IDE that allows code edit, compile and download to devices right from your web browser. It also integrates seamlessly with TI Resource Explorer Cloud with the ability to import projects directly on the cloud.

A full comparison between Code Composer Studio Cloud and Code Composer Studio Desktop is available here.

See Code Composer Studio Cloud now at dev.ti.com.

4.1.3 Code Composer Studio™ Desktop IDE

Code Composer Studio Desktop is a professional integrated development environment that supports TI's Microcontroller and Embedded Processors portfolio. Code Composer Studio comprises a suite of tools used to develop and debug embedded applications. It includes an optimizing C/C++ compiler, source code editor, project build environment, debugger, profiler, and many other features.

NOTE: The MSP432 LaunchPad development kit requires CCS Version 6.1.0 or later. See the *Code Composer Studio™ IDE 7.1+ for SimpleLink™ MSP432™ Microcontrollers User's Guide* for detailed instructions of using the IDE with MSP432. To use the SimpleLink MSP432 SDK, CCS Version 7.1.0 or later is required.

Learn more about CCS and download it at www.ti.com/tool/ccstudio.

4.1.4 Keil® µVision® IDE

The µVision IDE is an embedded project development environment included in Keil's Microcontroller Development Kit Version 5, that provides an source code editor, project manager, and make utility tool. µVision supports all the Keil tools including C/C++ Compiler, Macro Assembler, Linker, Library Manager, and Object-HEX Converter.

NOTE: The MSP432 LaunchPad development kit requires µVision IDE/MDK Version 5 or later. See the *Arm® Keil® MDK Version 5 for SimpleLink™ MSP432™ Microcontrollers User's Guide* for detailed instructions of using the IDE with MSP432.

Learn more about Keil µVision and download it at www.keil.com/arm/mdk.asp.

4.1.5 IAR Embedded Workbench® for Arm IDE

IAR Embedded Workbench for Arm IDE is another very powerful integrated development environment that allows you to develop and manage complete embedded application projects. It integrates the IAR C/C++ Compiler, IAR Assembler, IAR ILINK Linker, editor, project manager, command line build utility, and IAR C-SPY Debugger.

NOTE: The MSP432 LaunchPad development kit requires the IAR Embedded Workbench for Arm IDE Version 7.10 or later. See the *IAR Embedded Workbench for Arm 7.x for SimpleLink™ MSP432™ Microcontrollers User's Guide* for detailed instructions of using the IDE with MSP432. To use the SimpleLink MSP432 SDK, IAR Version 7.80.3 or later is required.

Learn more about IAR Embedded Workbench and download it at https://www.iar.com/iar-embedded-workbench/arm.

4.1.6 Energia

Energia is a simple open-source community-driven code editor that is based on the Wiring and Arduino framework. Energia provides unmatched ease of use through very-high-level APIs that can be used across hardware platforms. Energia is a light-weight IDE that does not have the full feature set of CCS, Keil, or IAR. However, Energia is great for anyone who wants to get started very quickly or who does not have significant coding experience.

Learn more about Energia and download it at energia.nu.

4.2 LaunchPad Development Kit Websites

More information about the LaunchPad development kit, supported BoosterPack plug-in modules, and available resources can be found at:

- MSP-EXP432P401R tool folder: Resources specific to this particular LaunchPad development kit
- TI's LaunchPad development kit portal: Information about all LaunchPad development kits from TI

4.3 SimpleLink SDK and TI Resource Explorer

TI Resource explorer is a tool integrated into CCS that allows you to browse through available design resources. TI Resource Explorer will help you quickly find what you need inside packages including SimpleLink SDK for the SimpleLink MCUs such as MSP432, CC3200, CC2640, and more. TI Resource Explorer is well organized to find everything that you need quickly, and you can import software projects into your workspace, find documentation, and browse libraries in your workspace in just a few clicks.

TI Resource Explorer Cloud is one of the TI Cloud Development tools and is tightly integrated with CCS Cloud. See Section 4.1.2 for more information.

The SimpleLink SDK is a collection of code examples, software libraries, data sheets, and other design resources for all SimpleLink MCU devices delivered in a convenient package – essentially everything developers need to become SimpleLink experts.

The SimpleLink MCU portfolio offers a single development environment that delivers flexible hardware, software, and tool options for customers developing wired and wireless applications. With an ultimate goal of 100 percent code reuse across host MCUs, Wi-Fi, Bluetooth low energy, Sub-1 GHz devices, and more, choose the MCU or connectivity standard that fits your design. A one-time investment with the SimpleLink software development kit (SDK) lets you reuse often, opening the door to create unlimited applications. For more information, visit ww.ti.com/simplelink.

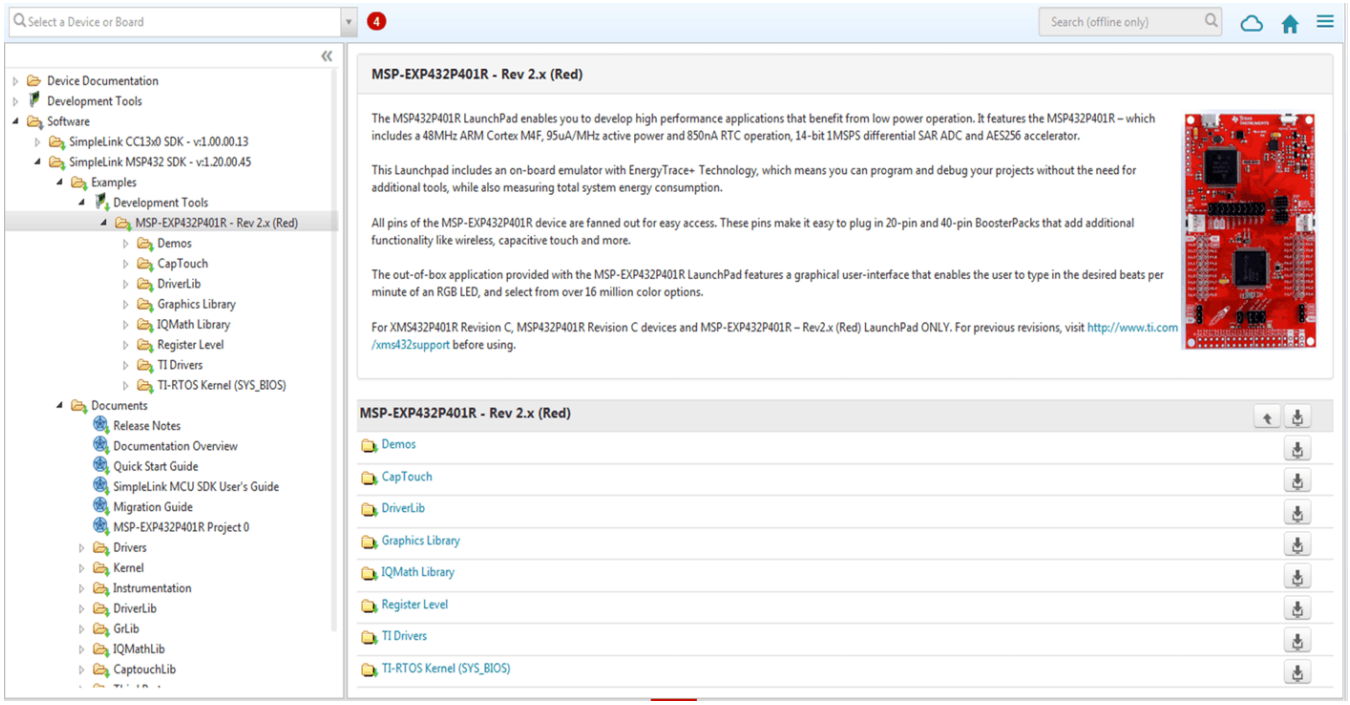

Figure 21. Using TI Resource Explorer to Browse MSP-EXP432P401R in SimpleLink SDK

Inside TI Resource Explorer, these examples and many more can be found and easily imported into CCS with one click.

4.4 MSP432P401R

4.4.1 Device Documentation

At some point, you will probably want more information about the MSP432P401R device. For every MSP device, the documentation is organized as shown in Table 8.

Table 8. How MSP Device Documentation is Organized

Document	For MSP432P401R	Description
Device family user's guide	*MSP432P4xx SimpleLink™ Microcontrollers Technical Reference Manual*	Architectural information about the device, including all modules and peripherals such as clocks, timers, ADC, and so on
Device-specific data sheet	*MSP432P401xx SimpleLink™ Mixed-Signal Microcontrollers*	Device-specific information and all parametric information for this device

4.4.2 MSP432P401R Code Examples

Inside of the SimpleLink MSP432 SDK, a set of very simple MSP432P4xx code examples can be found that demonstrate how to use the entire set of MSP432 peripheral: serial communication, precision ADC, Timer_A, Timer_B, and so on. These examples show both the direct register access and driver library methods.

Every MSP derivative has a set of these code examples. When starting a new project or adding a new peripheral, these examples serve as a great starting point (see Section 4.3).

4.4.3 MSP432 Application Notes and TI Designs

There are many application notes that can be found at www.ti.com/msp432, as well as TI Designs with practical design examples and topics.

4.5 Community Resources

4.5.1 TI E2E™ Community

Search the E2E forums at e2e.ti.com. If you cannot find your answer, post your question to the community.

4.5.2 Community at Large

Many online communities focus on the LaunchPad development kit; for example, www.43oh.com. You can find additional tools, resources, and support from these communities.

5 FAQ

Q: I can't program my LaunchPad development kit; the IDE can't connect to target. What's wrong?

A: Check the following:

- Are the JTAG jumpers on J101 populated (GND, RST, TMS, TCK, TDO, TDI)?
- Check for power to the target
  - Are the 3V3 and GND jumpers on J101 populated and USB cable plugged in?
  - If using an external debug probe, is USB power provided as shown above? Otherwise is external power provided to the target?
- Check the debug probe settings: change to Serial Wire Debug (SWD) without SWO.
  1. Under targetconfigs, double-click the *.ccxml file.
  2. Click the Advanced tab at the bottom.
  3. Click on Texas Instruments XDS110 USB Debug Probe.
  4. Under Connection Properties, change SWD Mode Settings to Use SWD Mode with SWO Trace Disabled.

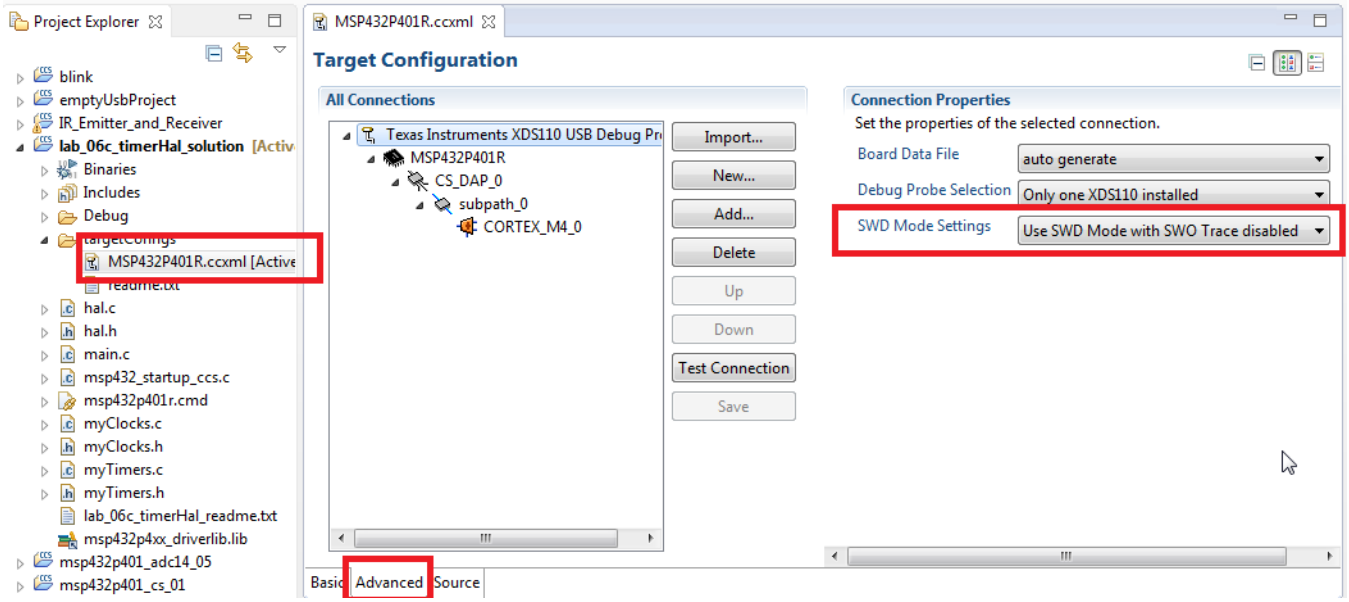

Figure 22. SWD Mode Settings

5. When the settings of Port J (PJSEL0 and PJSEL1 bits) are changed, full JTAG access is prevented on these pins. Changing to use SWD allows access through the dedicated debug pins only.

- If even this cannot connect, reset the device to factory settings:
  1. Click **View → Target Configurations**. CCS shows the target configuration.

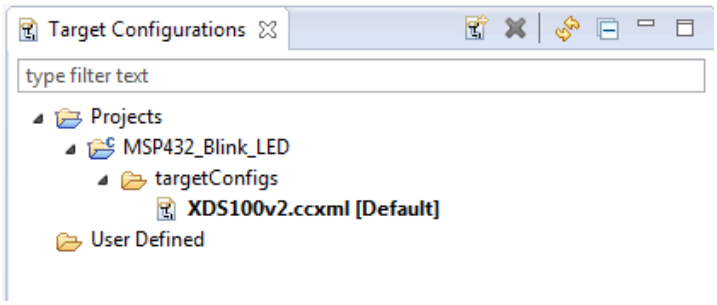

Figure 23. Target Configurations

If using the onboard debug probe, XDS110-ET is shown.

2. Right click **Launch Selected Configuration**.

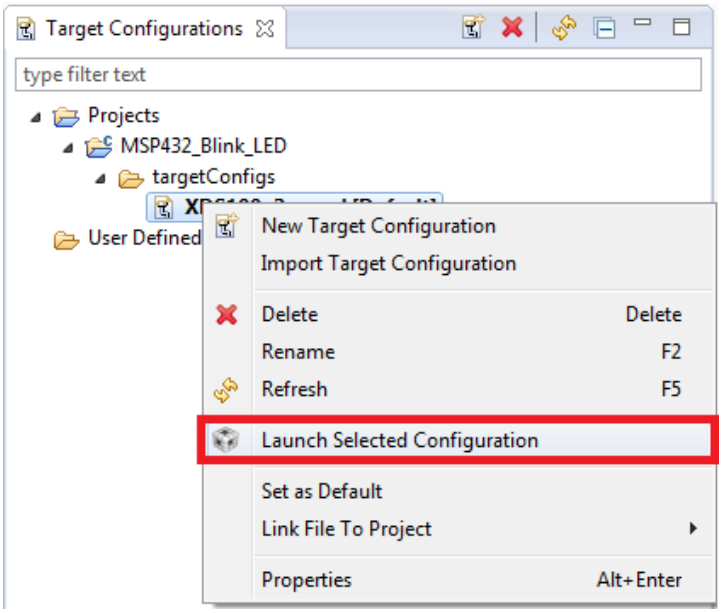

Figure 24. Launch Selected Configuration

3. The debug probe now connects to the device (which is still possible) but does not try to halt the CPU, write to registers, or even download code (which would not be possible). The Debug view that is spawned shows the CPU core but marks it as disconnected.
4. Right click **Show all cores**.

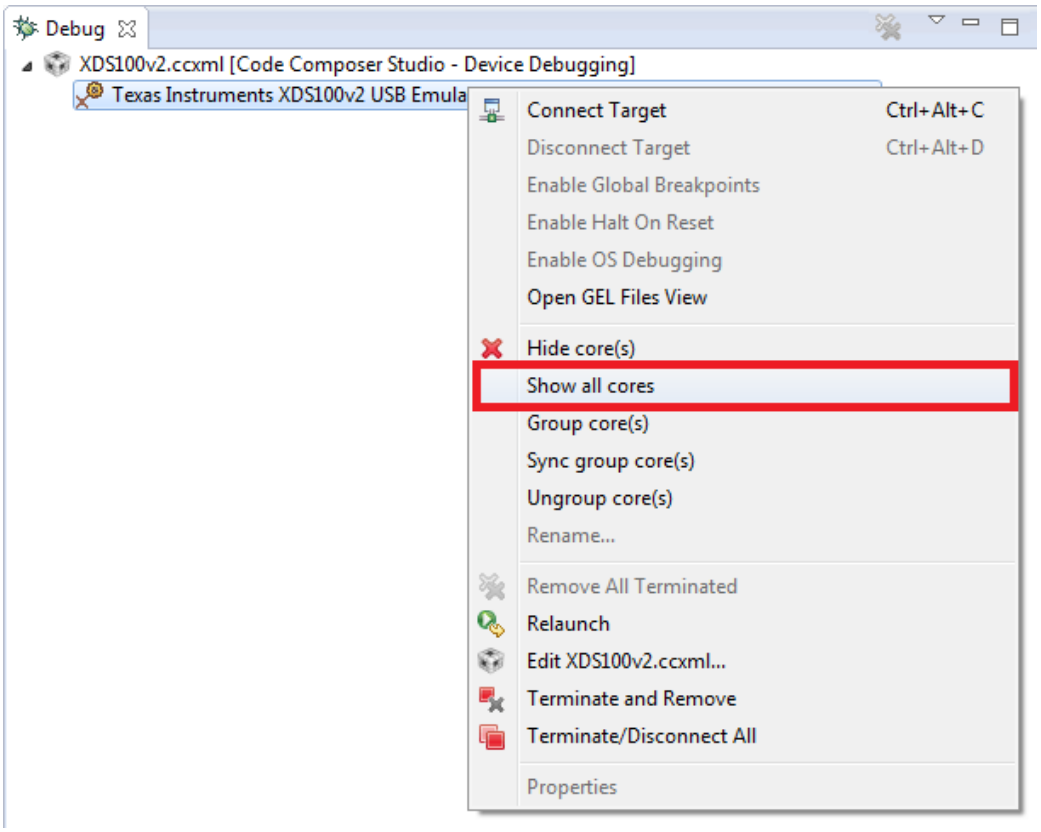

Figure 25. Show All Cores

The MSP432 Debug Access Port, or DAP, is shown under **Non Debuggable Devices**.

5. Right click **Connect Target**

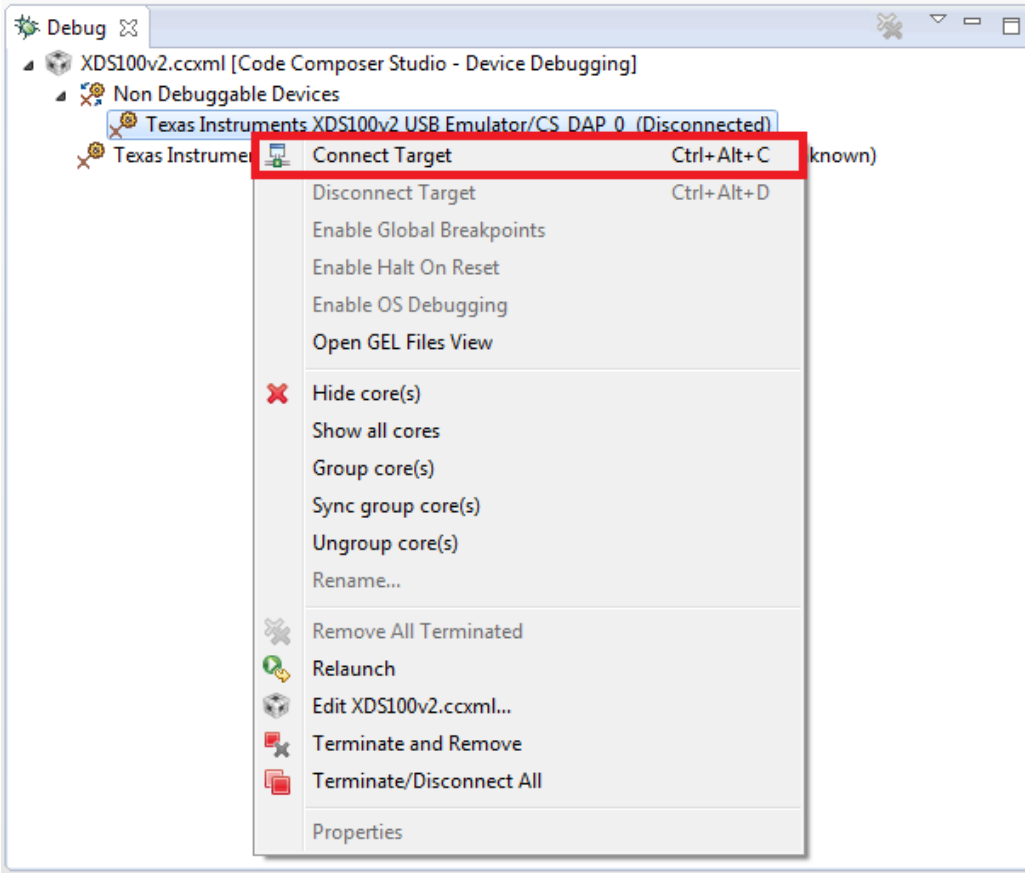

Figure 26. Connect Target

Now run a script to return the device back to factory settings:

Scripts > default > MSP432_Factory_Reset

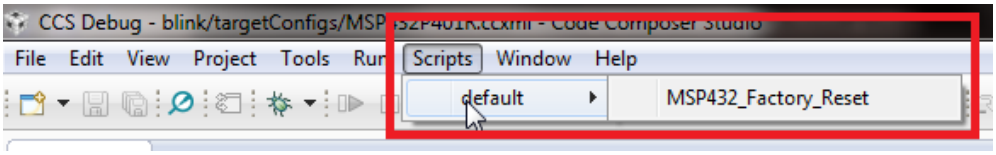

Figure 27. MSP432_Factory_Reset Script

- These instructions are generally the same for all IDEs, but the exact steps may vary slightly by IDE. See the following IDE user's guides for additional details:
  - Code Composer Studio™ IDE 7.1+ for SimpleLink™ MSP432™ Microcontrollers User's Guide
  - Arm® Keil® MDK Version 5 for SimpleLink™ MSP432™ Microcontrollers User's Guide
  - IAR Embedded Workbench for Arm 7.x for SimpleLink™ MSP432™ Microcontrollers User's Guide

Q: How do I use the LaunchPad development kit and my Segger J-Link to debug the target externally? It won't connect to the onboard connector.

A: The Segger J-Link does not come with an adapter for the 10-pin small-pitch Arm connector. The adapter cable is available from Segger.

Q: Problems plugging the MSP432 LaunchPad development kit into a USB3.0 port.

A: It has been observed that when the MSP432 LaunchPad development kit is connected to USB3.0 ports provided by a certain combination of USB3.0 host controller hardware and associated device drivers that the IDE is unable to establish a debug session with the LaunchPad development kit, resulting in an error message like "CS_DAP_0: Error connecting to the target: (Error -260 @ 0x0) An attempt to connect to the XDS110 failed." in the case of Code Composer Studio IDE. In this case the CCS-provided low-level command line utility ‘xdsdfu' will also not be able to establish a connection with the LaunchPad development kit.

Specifically, this issue was observed on PCs running Windows 7 that show the "Renesas Electronics USB 3.0 Host Controller" and the associated "Renesas Electronics USB 3.0 Root Hub" in the device manager. After updating the associated Windows USB drivers to more recent versions obtained from the hardware vendor the issue went away. There might be other USB3.0 hardware and device driver combinations that will lead to the same issue. If you think you might be affected, try to contact your PC vendor or try to locate and install more recent versions of the USB3.0 device drivers. Alternatively, connect the LaunchPad development kit to an USB2.0 port on your PC, if one is available.

Q: I can't get the backchannel UART to connect. What's wrong?

A: Check the following:

- Do the baud rate in the host's terminal application and the eUSCI settings match?
- Are the appropriate jumpers in place on the isolation jumper block?
- Probe on RXD and send data from the host. If you do not see data, it might be a problem on the host side.
- Probe on TXD while sending data from the MSP432. If you do not see data, it might be a configuration problem with the eUSCI module.
- Consider the use of the hardware flow control lines (especially for higher baud rates).

6 Schematics

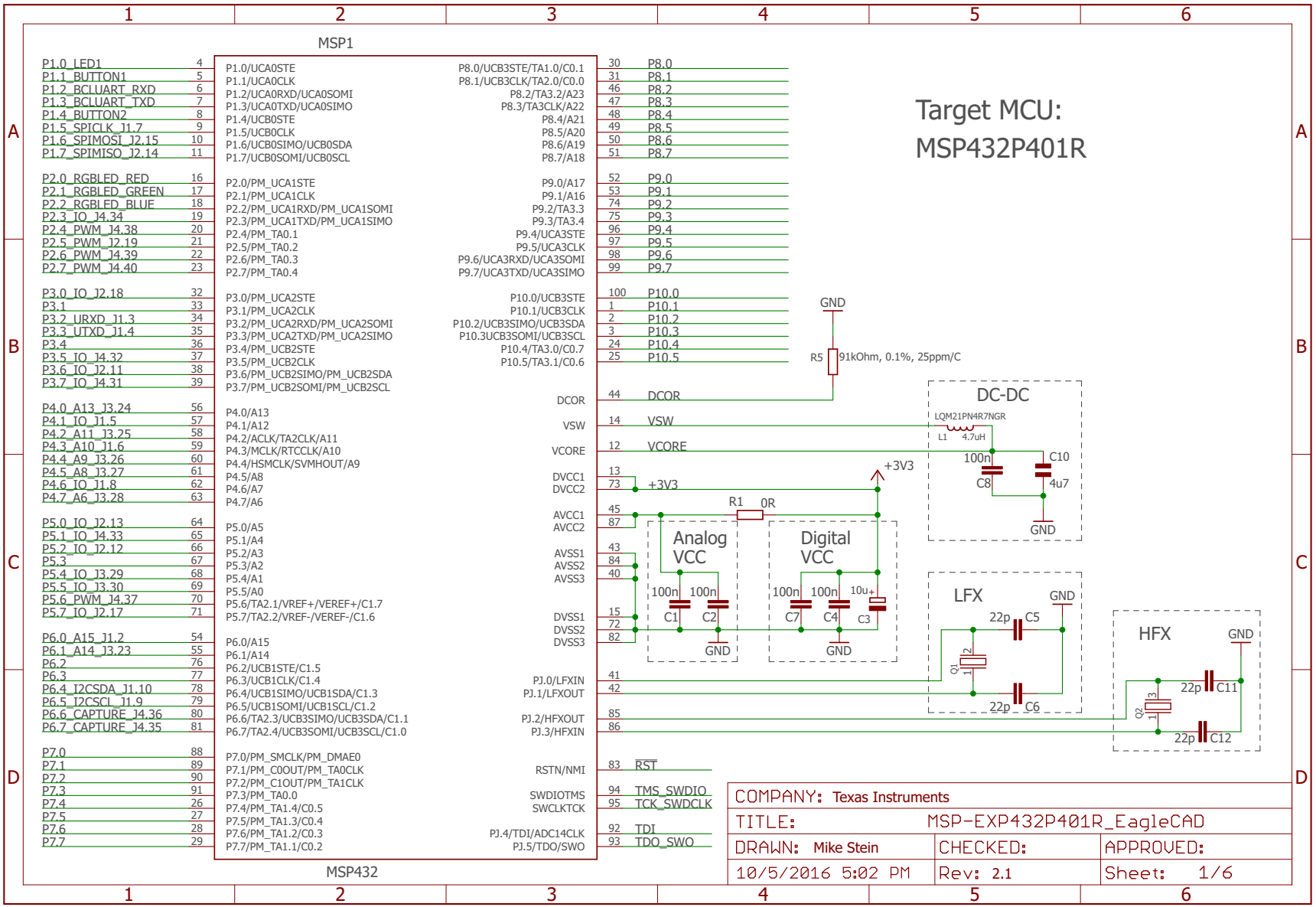

Figure 28. Schematics (1 of 6)

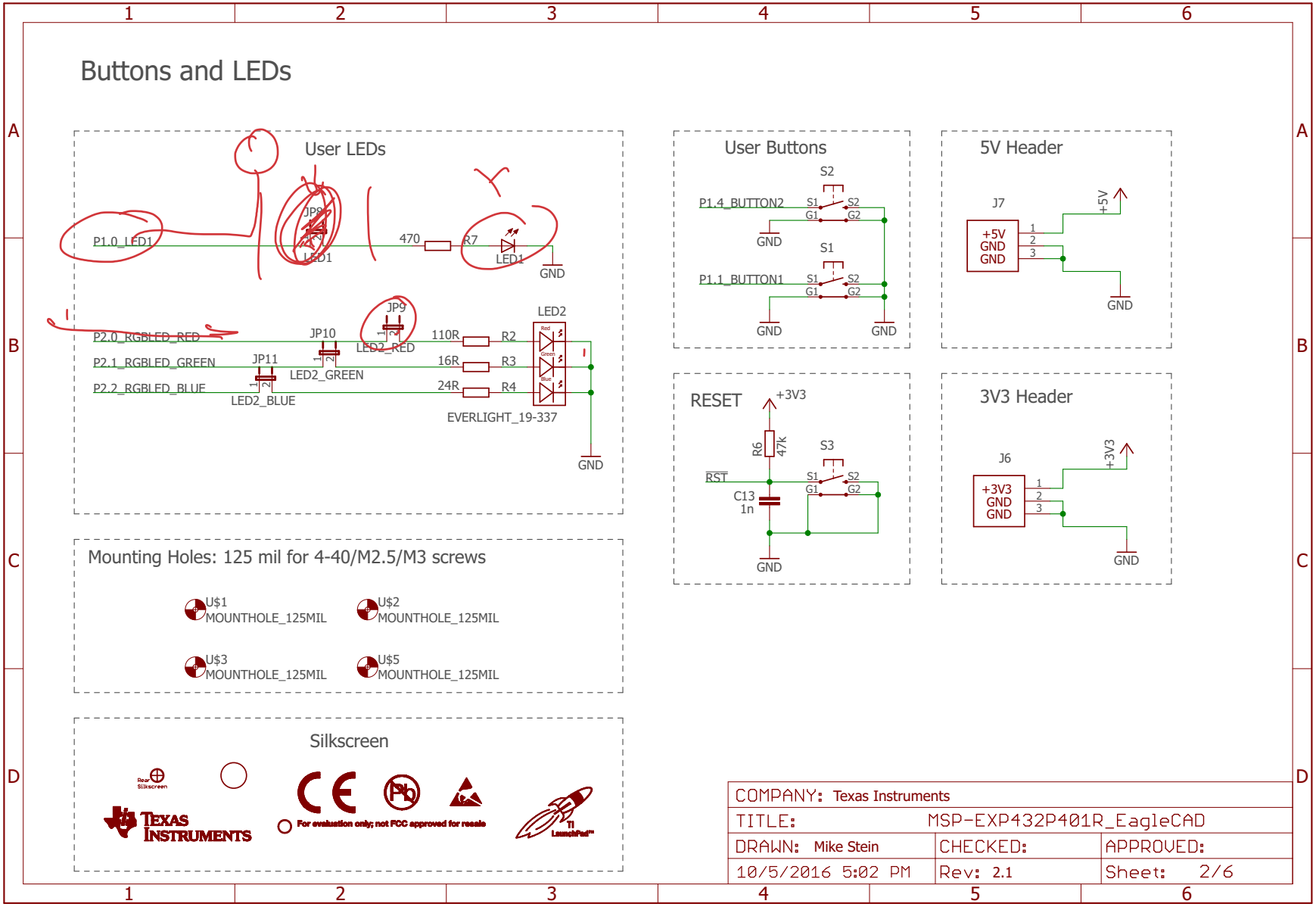

Figure 29. Schematics (2 of 6)

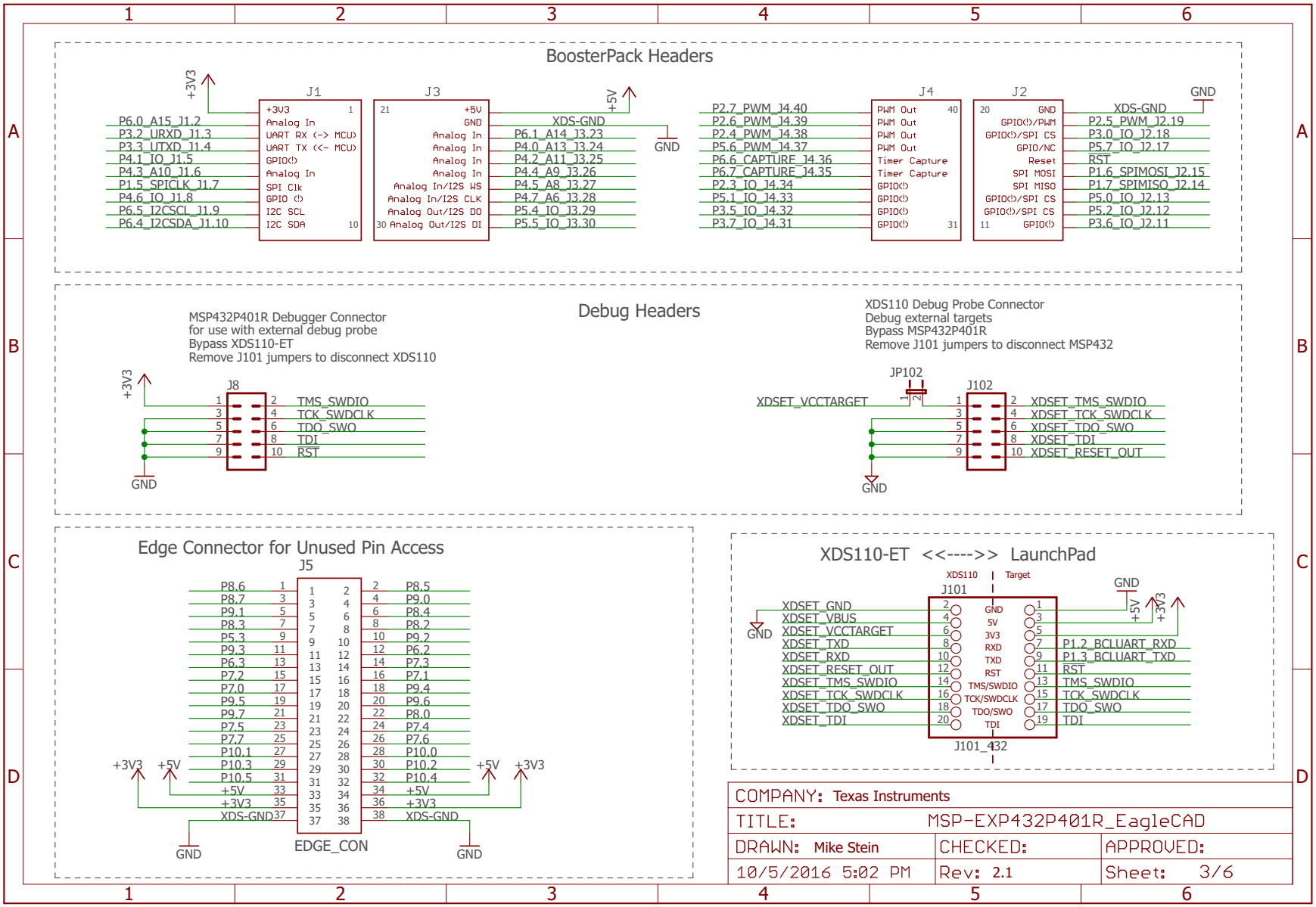

Figure 30. Schematics (3 of 6)

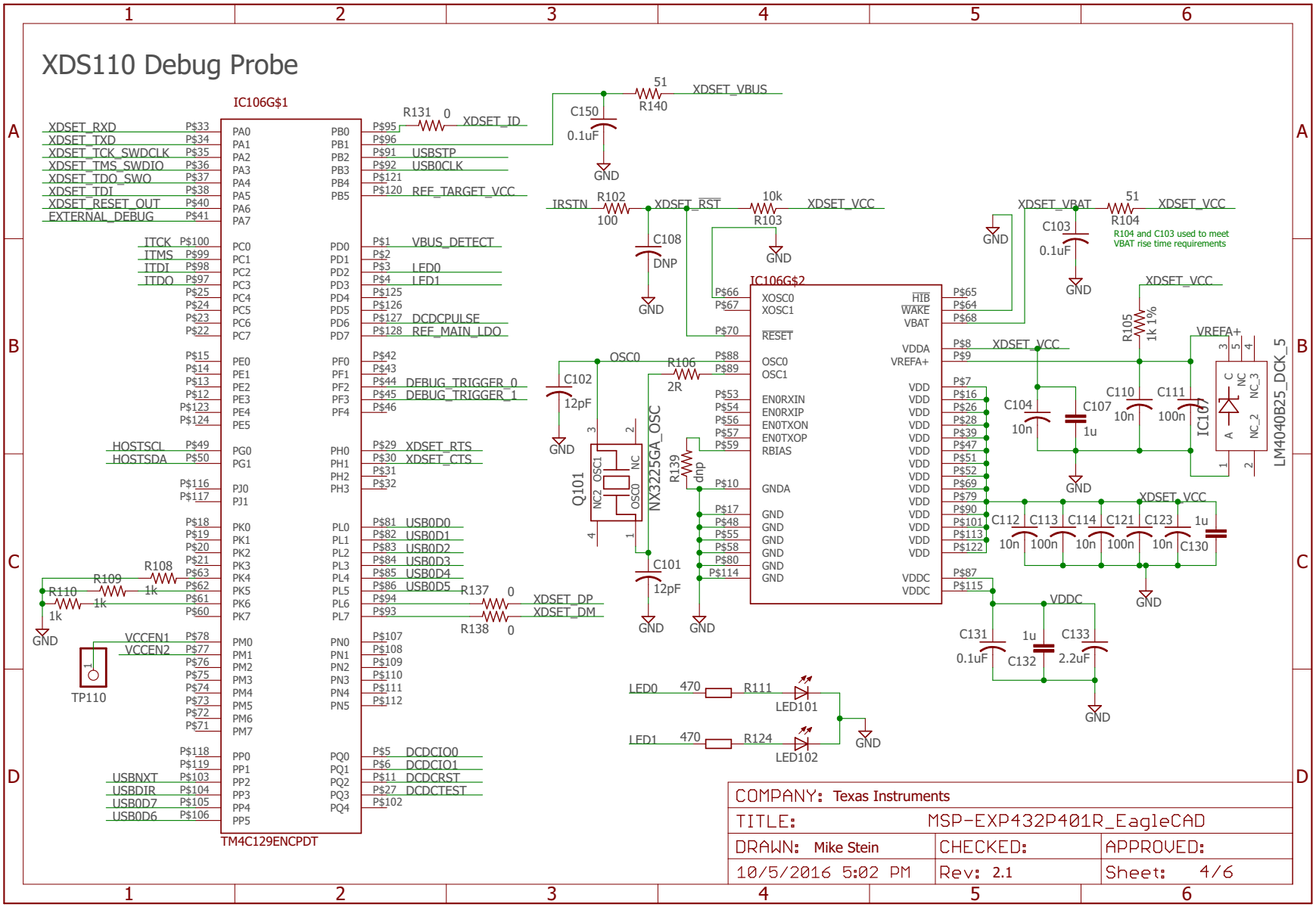

Figure 31. Schematics (4 of 6)

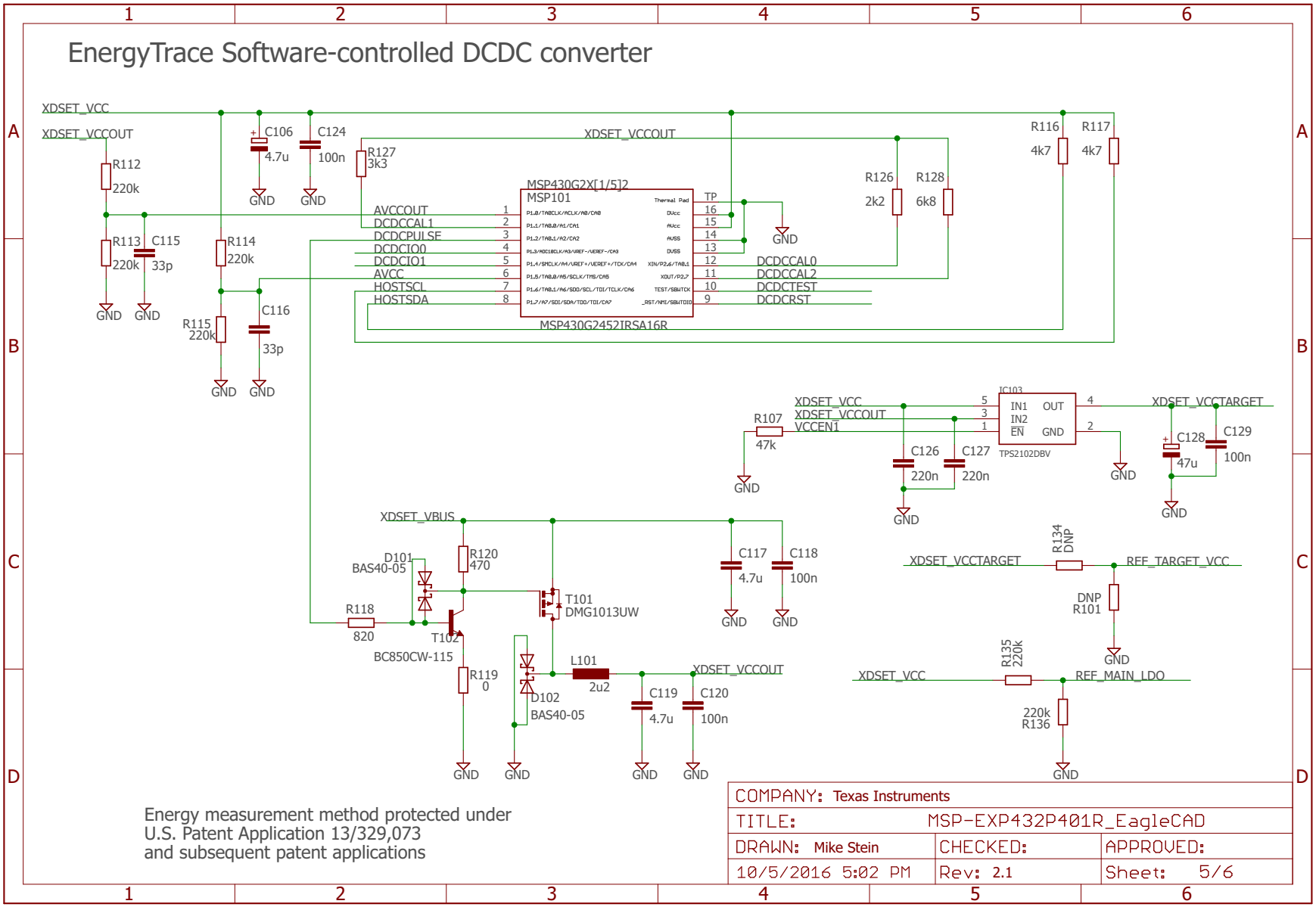

Figure 32. Schematics (5 of 6)

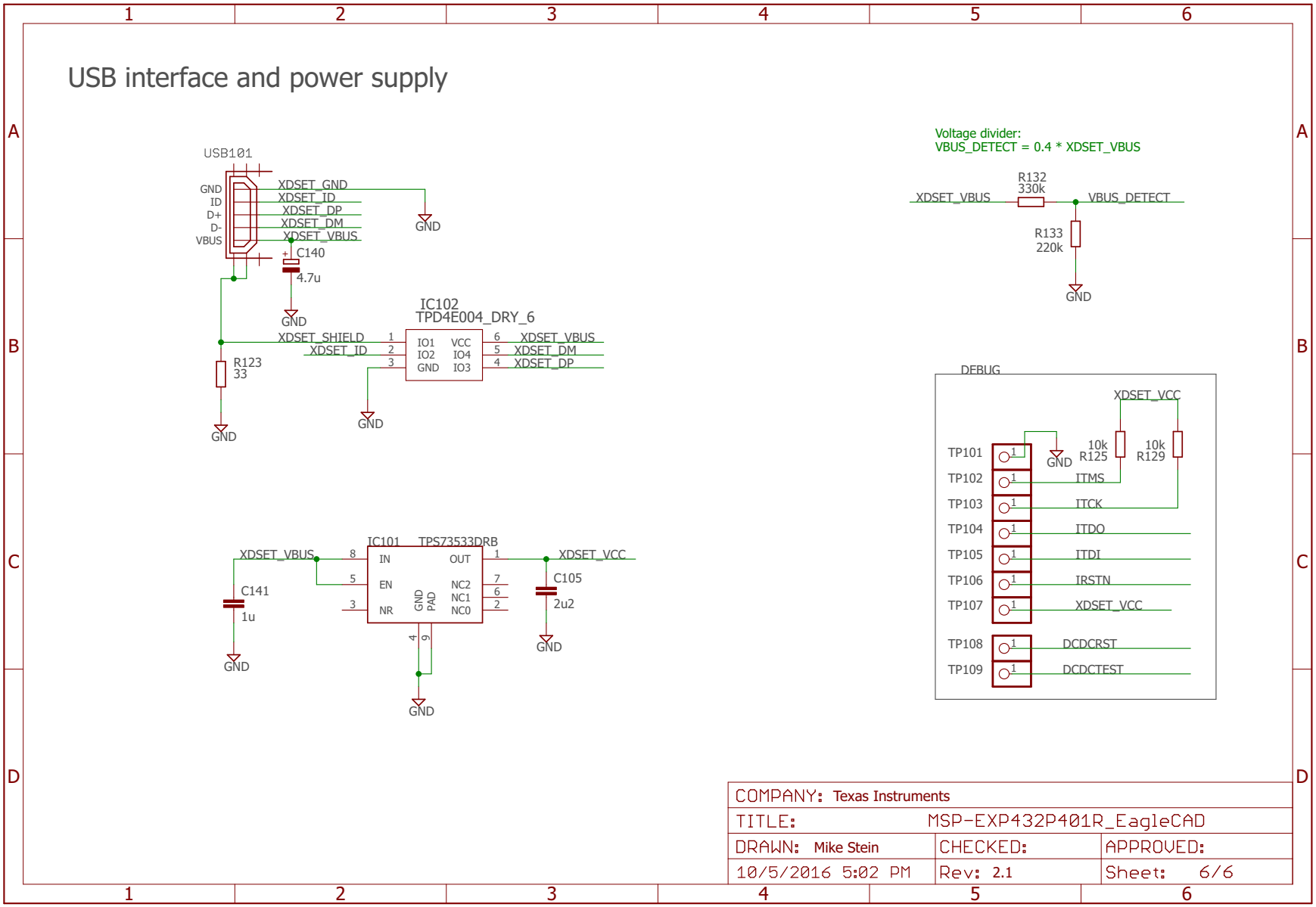

Figure 33. Schematics (6 of 6)

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from January 9, 2018 to March 8, 2018	Page
• Changed the link to the 10-pin Cortex-M JTAG cable in Section 2.3.4, *Using the XDS110-ET Emulator With a Different Target*	10
• Updated the answer to the question that begins "Q: How do I use the LaunchPad development kit and my Segger J-Link..." in Section 5, FAQ	34

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

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