User's Guide

SLAU595B–October 2014–Revised January 2017

MSP430FR4133 LaunchPad™ Development Kit (MSP-EXP430FR4133)

The MSP-EXP430FR4133 LaunchPad™ Development Kit is an easy-to-use evaluation module (EVM) for the MSP430FR4133 microcontroller (see Figure 1). It contains everything needed to start developing on the MSP430™ ultra-low-power (ULP) FRAM-based microcontroller (MCU) platform, including on-board emulation for programming, debugging, and energy measurements. The board features on-board buttons and LEDs for quick integration of a simple user interface and a liquid crystal display (LCD) that showcases the integrated driver with flexible software-configurable pins.

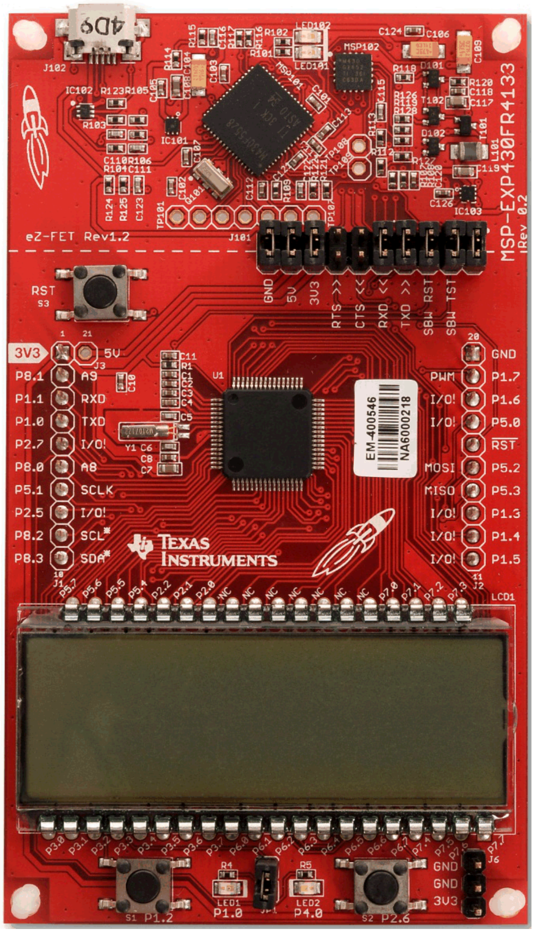


Figure 1. MSP-EXP430FR4133

Contents

List of Figures

List of Tables

Trademarks

LaunchPad, MSP430, BoosterPack, Code Composer Studio, EnergyTrace, E2E are trademarks of Texas Instruments.
IAR Embedded Workbench is a registered trademark of IAR Systems.

1 Getting Started

1.1 Introduction

The MSP-EXP430FR4133 LaunchPad™ Development Kit is an easy-to-use Evaluation Module (EVM) for the MSP430FR4133 microcontroller (see Figure 1). It contains everything needed to start developing on the MSP430™ ultra-low-power (ULP) FRAM-based microcontroller (MCU) platform, including on-board emulation for programming, debugging, and energy measurements. The board features on-board buttons and LEDs for quick integration of a simple user interface and a liquid crystal display (LCD) that showcases the integrated driver with flexible software-configurable pins. The MSP430FR4133 device features embedded FRAM (ferroelectric random access memory), a nonvolatile memory known for its ultra-low power, high endurance, and high-speed write access.

Rapid prototyping is simplified by the 20-pin BoosterPack™ Plug-in Module headers, which support a wide range of available BoosterPacks. You can quickly add features like wireless connectivity, graphical displays, environmental sensing, and much more. Design your own BoosterPack or choose among many already available from TI and third-party developers.

Free software development tools are also available, including TI's Eclipse-based Code Composer Studio™ (CCSTUDIO) and IAR Embedded Workbench® IAR-KICKSTART. Both of these integrated development environments (IDEs) support EnergyTrace™ technology when paired with the MSP430FR4133 LaunchPad. More information about the LaunchPad, the supported BoosterPacks, and available resources can be found at TI's LaunchPad portal.

1.2 Key Features

- MSP430 ultra-low-power FRAM technology based MSP430FR4133 16-bit MCU
- 20-pin LaunchPad standard that leverages the BoosterPack ecosystem
- eZ-FET, an open-source onboard debugger that features EnergyTrace technology
- On-board segmented LCD
- Two buttons and two LEDs for user interaction
- Backchannel UART through USB to PC

1.3 What's Included

1.3.1 Kit Contents

- 1 x MSP-EXP430FR4133 LaunchPad Development Kit
- 1 x micro-USB cable
- 1 x Quick Start Guide

1.3.2 Software Examples

- Out-of-Box Software

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 from a user level.

1.4.1 Connecting to the Computer

Connect the LaunchPad using the included USB cable to a computer. A green power LED should illuminate. For proper operation, drivers are needed. It is recommended to get drivers by installing an IDE such as TI's CCS or IAR EW430. Drivers are also available at www.ti.com/MSPdrivers.

1.4.2 Running the Out-of-Box Demo

When connected to the computer, the LaunchPad powers up and displays a greeting message on the LCD. Press and hold the S1 and S2 buttons simultaneously to select a new mode. A more detailed explanation of each mode can be found in Section 3.4.

1.4.2.1 Stopwatch Mode

This mode provides a simple stopwatch application. It supports split time, where the display freezes while the stopwatch continues running in the background.

Timer Stopped:

S1: Start time

S2: Reset time

Timer Running:

S1: Stop time

S2: Split time (lap time)

1.4.2.2 Temperature Mode

This mode provides a simple thermometer application. Using the on-chip temperature sensor, the temperature is displayed on the LCD.

S1: Pause current temperature

S2: Toggle temperature between °F and °C

1.5 Next Steps: Looking Into the Provided Code

After the EVM features have been explored, the fun can begin. It's time to open an integrated development environment and start editing the code examples. Refer to Section 3.3 for more information on IDEs and where to download them.

The out-of-box source code and more code examples are provided for download at http://www.ti.com/tool/msp-exp430fr4133. Code is licensed under BSD, and TI encourages reuse and modifications to fit specific needs.

Section 3 describes all of the functions in detail and describes the project structure to help familiarize you with the code.

With the onboard eZ-FET emulator debugging and downloading new code is simple. A USB connection between the EVM and a PC through the provided USB cable is all that is needed.

2 Hardware

Figure 2 shows an overview of the MSP-EXP430FR4133 hardware.

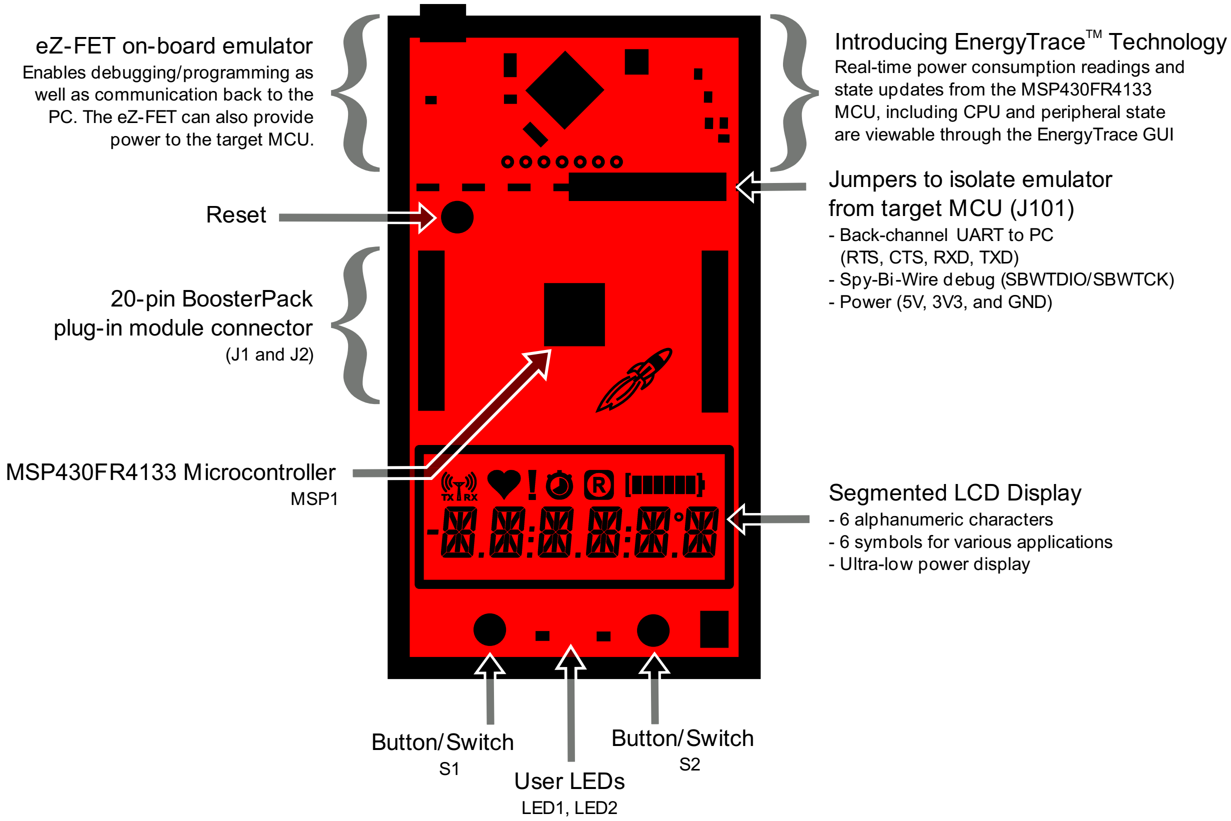

eZ-FET on-board emulator
Enables debugging/programming as well as communication back to the PC. The eZ-FET can also provide power to the target MCU.
Reset
20-pin BoosterPack plug-in module connector
(J1 and J2)
MSP430FR4133 Microcontroller
MSP1
Introducing EnergyTrace™ Technology
Real-time power consumption readings and state updates from the MSP430FR4133 MCU, including CPU and peripheral state are viewable through the EnergyTrace GUI
Jumpers to isolate emulator from target MCU (J101)
- Back-channel UART to PC (RTS, CTS, RXD, TXD)
- Spy-Bi-Wire debug (SBWTDIO/SBWTCK)
- Power (5V, 3V3, and GND)
Segmented LCD Display
- 6 alphanumeric characters
- 6 symbols for various applications
- Ultra-low power display
TX
RX
Button/Switch
S1
User LEDs
LED1, LED2
Button/Switch
S2

Figure 2. MSP-EXP430FR4133 Overview

2.1 Block Diagram

Figure 3 shows the block diagram.

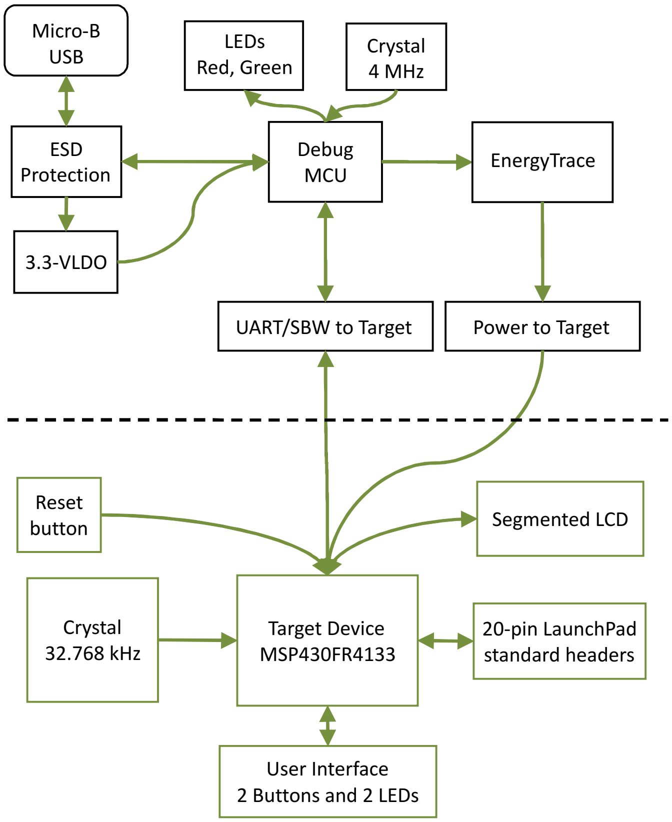


Figure 3. Block Diagram

2.2 Hardware Features

2.2.1 MSP430FR4133

The MSP430FR4133 is the next device in TI's new ULP FRAM technology platform. FRAM is a cutting edge memory technology, combining the best features of flash and RAM into one nonvolatile memory.

Device features include:

- 1.8-V to 3.6-V operation
- Up to 16-MHz system clock and 8-MHz FRAM access
- 16KB of nonvolatile FRAM
- Industry's lowest-power LCD controller
- IR modulation logic
- Two timer blocks and up to three serial interfaces (SPI, UART, or I^{2}C)
- Analog: 10-channel 10-bit differential ADC
- Digital: RTC, CRC

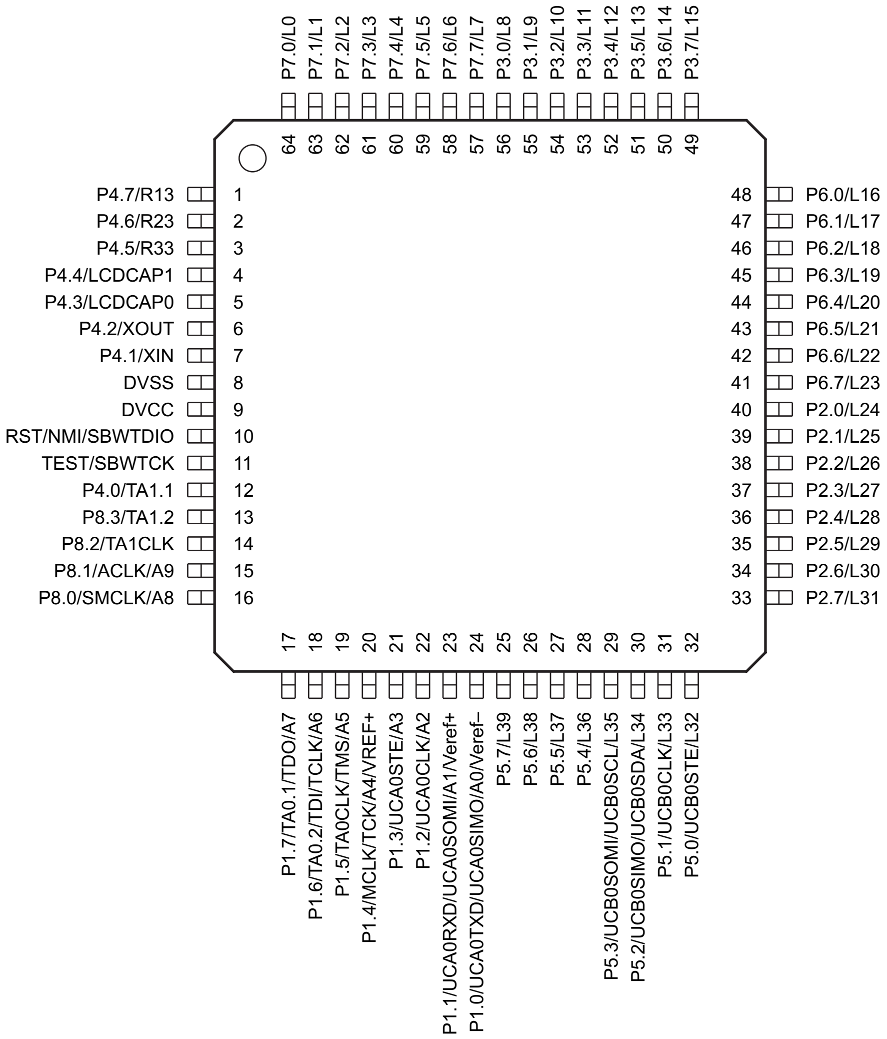


Figure 4. MSP430FR4133 Pinout

2.2.2 eZ-FET Onboard Emulator With EnergyTrace™ Technology

To keep development easy and cost effective, TI's LaunchPad Development Kits integrate an onboard emulator, which eliminates the need for expensive programmers. The MSP-EXP430FR4133 has the eZ-FET emulator (see Figure 5), which is a simple and low-cost debugger that supports all MSP430 device derivatives.

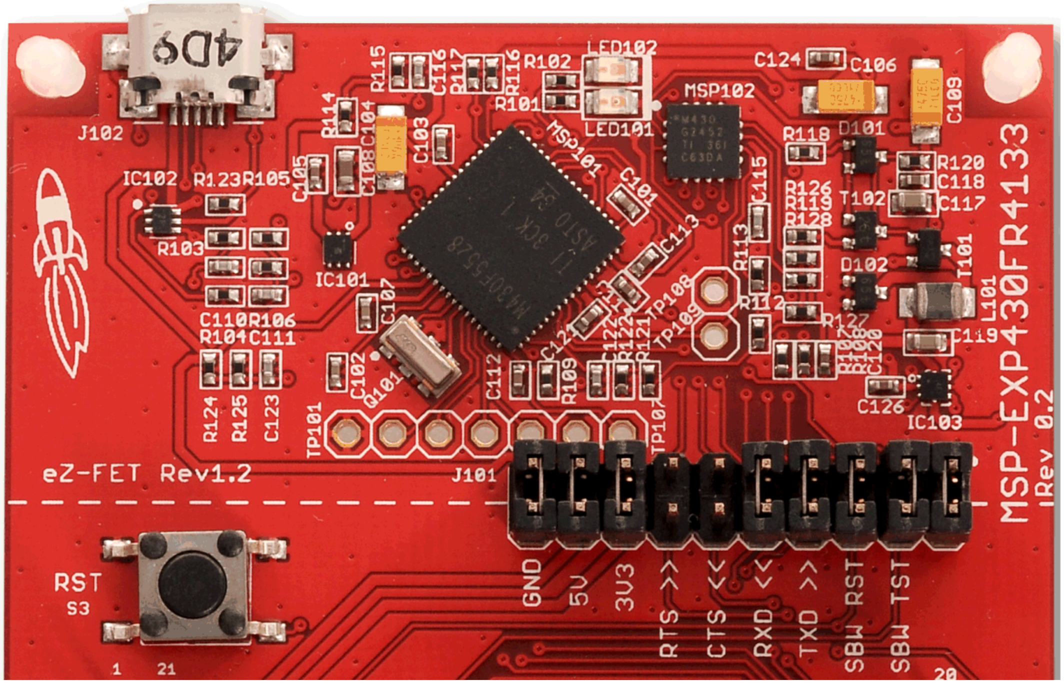


Figure 5. eZ-FET Emulator

The MSP-EXP430FR4133 LaunchPad features EnergyTrace technology but does not have support for EnergyTrace++™ technology. The EnergyTrace functionality varies across the MSP portfolio (see Table 1).

Table 1. EnergyTrace Technology

Feature	EnergyTrace™ Technology	EnergyTrace++™ Technology
Current Monitoring	✓	✓
CPU State		✓
Peripheral and System State		✓
Devices Supported	All MSP430 MCUs	MSP430FR59xx and MSP430FR69xx MCUs
Development Tool Required	MSP-FET or eZ-FET	MSP-FET or eZ-FET

The eZ-FET also 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 provided UART supports hardware flow control (RTS and CTS), although by default these signals are not connected to the target.

The dotted line through J101 shown in Figure 5 divides the eZ-FET emulator from the target area. The signals that cross this line can be disconnected by jumpers on J101, the isolation jumper block. More details on the isolation jumper block are in Section 2.2.3.

The eZ-FET hardware can be found in the schematics in Section 6 and in the accompanying MSP-EXP430FR4133 Hardware Design Files. The software and more information about the debugger can be found on the eZ-FET lite wiki.

2.2.3 Emulator Connection: Isolation Jumper Block

The isolation jumper block at jumper J101 allows the user to connect or disconnect signals that cross from the eZ-FET domain into the MSP430FR4133 target domain. This includes eZ-FET Spy-Bi-Wire signals, application UART signals, and 3.3-V and 5-V power (see Table 2 and Figure 6).

Reasons to open these connections:

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

Table 2. Isolation Block Connections

Jumper	Description
GND	Ground
5V	5-V VBUS from USB
3V3	3.3-V rail, derived from VBUS by an LDO in the eZ-FET domain
RTS >>	Backchannel UART: Ready-To-Send, for hardware flow control. The target can use this to indicate whether it is ready to receive data from the host PC. The arrows indicate the direction of the signal.
CTS <<	Backchannel UART: Clear-To-Send, for hardware flow control. The host PC (through the emulator) uses this to indicate whether it is ready to receive data. The arrows indicate the direction of the signal.
RXD <<	Backchannel UART: The target FR4133 receives data through this signal. The arrows indicate the direction of the signal.
TXD >>	Backchannel UART: The target FR4133 sends data through this signal. The arrows indicate the direction of the signal.
SBW RST	Spy-Bi-Wire emulation: SBWTDIO data signal. This pin also functions as the RST signal (active low).
SBW TST	Spy-Bi-Wire emulation: SBWTCK clock signal. This pin also functions as the TST signal.

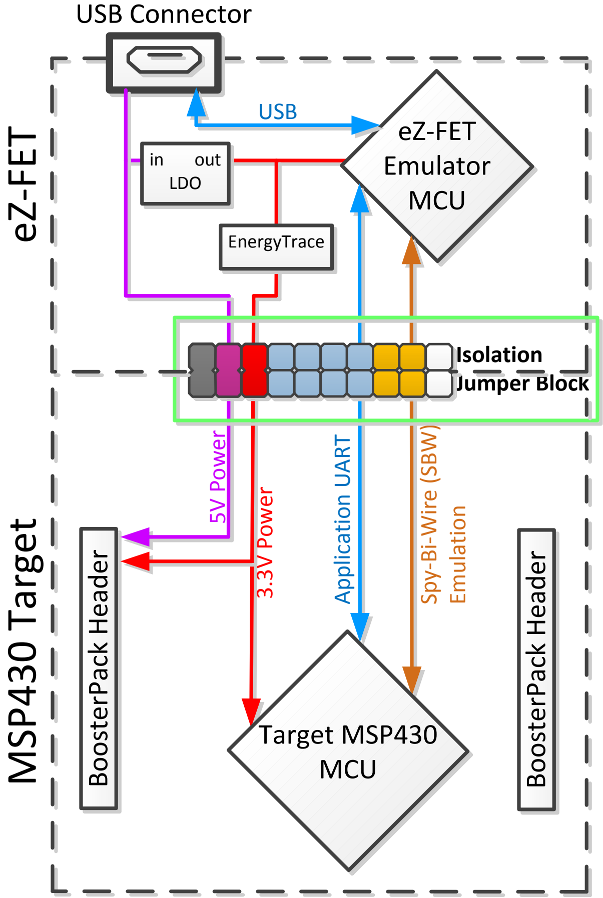


Figure 6. eZ-FET Isolation Jumper Block Diagram

2.2.4 Application (or "Backchannel") UART

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

The pathway of the backchannel UART is shown in Figure 6. The backchannel UART is the UART on eUSCI_A0. Because of USCI limitations, this UART channel is shared with the UART on the 20-pin BoosterPack connector (eUSCI_A0). This UART channel is also shared with the red LED on P1.0. This is a lot of sharing, but all 64 pins of the device are used and this was done out of necessity. The P1.0 pin serves more functions than any other.

On the host side, a virtual COM port for the application backchannel UART is generated when the LaunchPad 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 (see Figure 7).

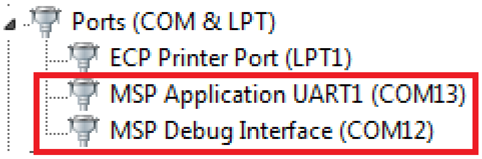

Ports (COM & LPT)
ECP Printer Port (LPT1)
MSP Application UART1 (COM13)
MSP Debug Interface (COM12)

Figure 7. Application Backchannel UART in Device Manager

The backchannel UART is the "MSP Application UART1" port. In this case, Figure 7 shows COM13, 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.

On the target MSP430FR4133 side, the backchannel is connected to the eUSCI_A0 module. The eZ-FET 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.

The eZ-FET also supports hardware flow control, if desired. Hardware flow control (CTS and RTS handshaking) allows the target MSP430FR4133 and the emulator to tell each other to wait before sending more data. At low baud rates and with simple target applications, flow control may not be necessary. Applications with higher baud rates and more interrupts to service have a higher likelihood that the will not be able to read the eUSCI_A0 RXBUF register in time, before the next byte arrives. If this happens, the eUSCI_A0 UCA0STATW register reports an overrun error.

2.2.5 Special Features

2.2.5.1 Liquid Crystal Display (LCD)

The MSP430FR4133 LaunchPad features an on-board LCD (see Figure 8). This LCD is driven by the internal LCD driver on the MSP430FR4133 device.

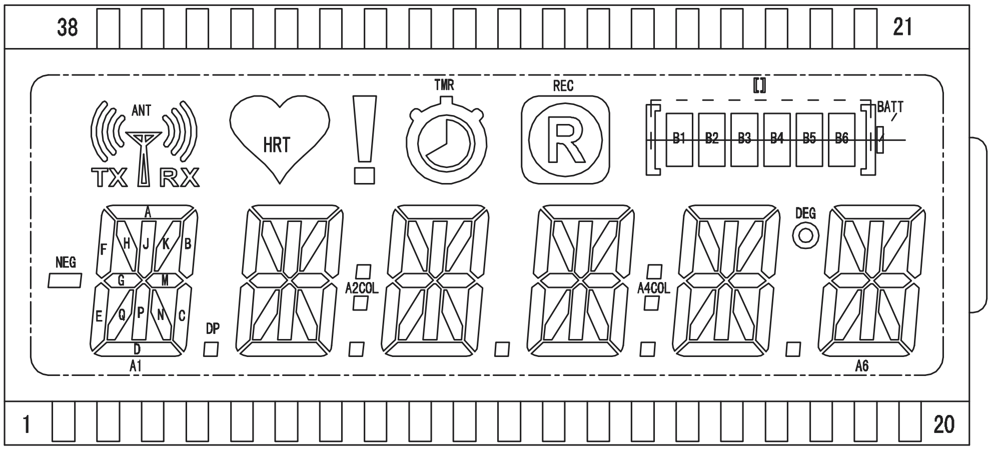


Figure 8. LCD Segment Layout

There are many available LCD segments, including six full alpha-numeric numbers and letters in addition to several symbols at the top for various modes or applications. Table 3 shows the mapping of these segments.

Table 3. LCD FH-1138P Segment Mapping

Pin	COM3	COM2	COM1	COM0
1	A1E	A1F	A1G	A1M
2	A1A	A1B	A1C	A1D
3	A1Q	NEG	A1N	A1DP
4	A1H	A1J	A1K	A1P
5	A2E	A2F	A2G	A2M
6	A2A	A2B	A2C	A2D
7	A2Q	A2COL	A2N	A2DP
8	A2H	A2J	A2K	A2P
9	A3R	A3F	A3G	A3M
10	A3A	A3B	A3C	A3D
11	A3Q	ANT	A3N	A3DP
12	A3H	A3J	A3K	A3P
13	A4R	A4F	A4G	A4M
14	A4A	A4B	A4C	A4D
15	A4Q	A4COL	A4N	A4DP
16	A4H	A4J	A4K	A4P
17	A5E	A5F	A5G	A5M
18	A5A	A5B	A5C	A5D
19	A5Q	DEG	A5N	A5DP
20	A5H	A5J	A5K	A5P
21	COM3	-	-	-
22	-	COM2	-	-
23	-	-	COM1	-
24	-	-	-	COM0
25	-	-	-	-
26	-	-	-	-
27	-	-	-	-
28	-	-	-	-
29	-	-	-	-
30	-	-	-	-
31	-	-	-	-
32	TMR	HRT	REC	!
33	B6	B4	B2	BATT
34	B5	B3	B1	[]
35	A6E	A6F	A6G	A6M
36	A6A	A6B	A6C	A6D
37	A6Q	TX	A6N	RX
38	A6H	A6J	A6K	A6P

The MSP430FR4133 device has flexible LCD pins allowing any pin to be a SEG or a COM. This simplifies layout for the user. The LCD connections are typically a tradeoff between easy layout and optimal memory settings for cleaner user software, among other considerations. The flexibility of the MSP430FR4133 allowed an optimal memory setting for easy software, along with a simple layout on the PCB (see Table 4).

Each LCDMEM register is eight bits, controlling up to eight segments. The FH-1138P is a 4-mux LCD, so four COM pins are needed from the MSP430FR4133. With four COM pins, each segment pin controls four bits or segments. This means that each LCDMEM register controls two segment pins, as shown in Table 4. Note that LCDMEM14 to LCDMEM17 and LCDMEM20 and higher are not used due to layout considerations.

Each alphanumeric character A1 to A6 is controlled by two adjacent LCDMEM registers for efficiency and ease of use in software. This allows for a single 16-bit memory access to control the whole character, as opposed to split memory regions requiring separate memory accesses.

Table 4. LCD to MSP430 Connections

LCDMEM	Port Pin	FR4133 Pin	LCD Pin	COM3	COM2	COM1	COM0	Port Pin	FR4133 Pin	LCD Pin	COM3	COM2	COM1	COM0
LCDM19	P5.7	L39	38	A6H	A6J	A6K	A6P	P5.6	L38	37	A6Q	TX	A6N	RX
LCDM18	P5.5	L37	36	A6A	A6B	A6C	A6D	P5.4	L36	35	A6E	A6F	A6G	A6M
LCDM17	P5.3	L35						P5.2	L34					
LCDM16	P5.1	L33						P5.0	L32					
LCDM15	P2.7	L31						P2.6	L30					
LCDM14	P2.5	L29						P2.4	L28					
LCDM13	P2.3	L27						P2.2	L26	34	B5	B3	B1	[]
LCDM12	P2.1	L25	33	B6	B4	B2	BATT	P2.0	L24	32	TMR	HRT	REC	!
LCDM11	P6.7	L23	16	A4H	A4J	A4K	A4P	P6.6	L22	15	A4Q	A4COL	A4N	A4DP
LCDM10	P6.5	L21	14	A4A	A4B	A4C	A4D	P6.4	L20	13	A4R	A4F	A4G	A4M
LCDM9	P6.3	L19	12	A3H	A3J	A3K	A3P	P6.2	L18	11	A3Q	ANT	A3N	A3DP
LCDM8	P6.1	L17	10	A3A	A3B	A3C	A3D	P6.0	L16	9	A3R	A3F	A3G	A3M
LCDM7	P3.7	L15	8	A2H	A2J	A2K	A2P	P3.6	L14	7	A2Q	A2COL	A2N	A2DP
LCDM6	P3.5	L13	6	A2A	A2B	A2C	A2D	P3.4	L12	5	A2E	A2F	A2G	A2M
LCDM5	P3.3	L11	4	A1H	A1J	A1K	A1P	P3.2	L10	3	A1Q	NEG	A1N	A1DP
LCDM4	P3.1	L9	2	A1A	A1B	A1C	A1D	P3.0	L8	1	A1E	A1F	A1G	A1M
LCDM3	P7.7	L7	20	A5H	A5J	A5K	A5P	P7.6	L6	19	A5Q	DEG	A5N	A5DP
LCDM2	P7.5	L5	18	A5A	A5B	A5C	A5D	P7.4	L4	17	A5E	A5F	A5G	A5M
LCDM1	P7.3	L3	21	COM3	-	-	-	P7.2	L2	22	-	COM2	-	-
LCDM0	P7.1	L1	23	-	-	COM1	-	P7.0	L0	24	-	-	-	COM0

2.3 Power

The board was designed to accommodate various powering methods, including through the on-board eZ-FET and from external or BoosterPack power (see Figure 9).

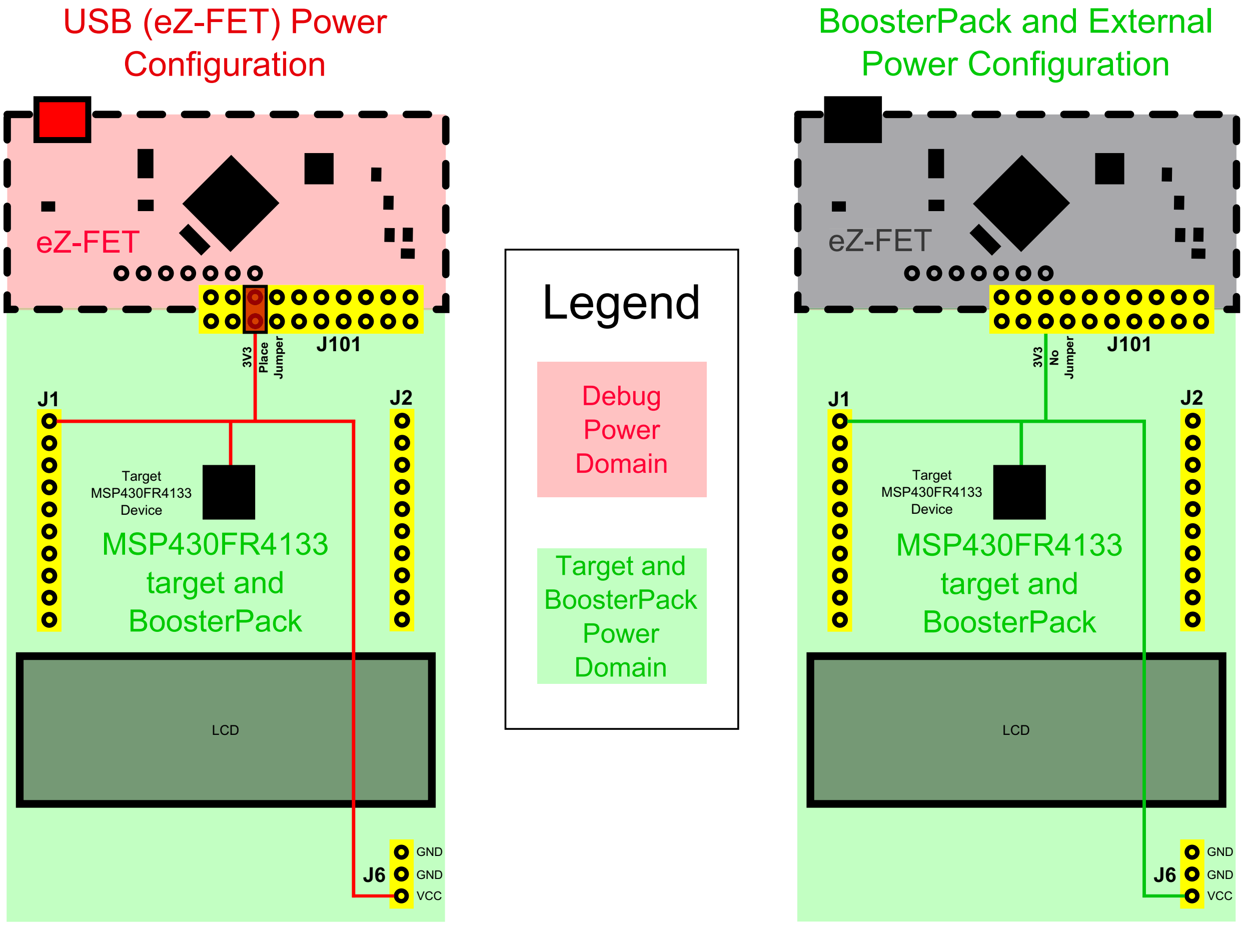

USB (eZ-FET) Power Configuration
BoosterPack and External Power Configuration
eZ-FET
J101
3V3
Place Jumper
No Jumper
J1
J2
Target MSP430FR4133 Device
MSP430FR4133 target and BoosterPack
LCD
J6
GND
GND
VCC
Legend
Debug Power Domain
Target and BoosterPack Power Domain

Figure 9. MSP-EXP430FR4133 Power Block Diagram

2.3.1 eZ-FET USB Power

The most common power-supply scenario is from USB through the eZ-FET debugger. This provides 5-V power from the USB and also regulates this power rail to 3.3 V for eZ-FET operation and 3.3 V to the target side of the LaunchPad. Power from the eZ-FET is controlled by jumper J101. For 3.3 V, make sure that a jumper is connected across the J101 3V3 terminal.

2.3.2 BoosterPack 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 MSP430FR4133 has an operating range of 1.8 V to 3.6 V. More information can be found in the MSP430FR4133 data sheet (SLAS865).

2.4 Measure MSP430 Current Draw

To measure the current draw of the MSP430FR4133 using a multi-meter, use the 3V3 jumper on the jumper isolation block. The current measured includes the target device and any current drawn through the BoosterPack 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 MSP430FR4133 may have on current draw. Consider disconnecting these at the isolation jumper block, or at least consider their current sinking and sourcing capability in the final measurement.
3. Make sure there are no floating input/output (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.

Alternatively, EnergyTrace technology can be used to measure the same current, and see energy profiles through integrated GUI in CCS and IAR. EnergyTrace allows you to compare various current profiles and better optimize your energy performance!

2.5 Clocking

The MSP-EXP430FR4133 provides an external clock in addition to the internal clocks in the device.

- Y1: a 32-kHz crystal

The 32-kHz crystal allows for lower LPM3 sleep currents than do the other low-frequency clock sources. Therefore, the presence of the crystal allows the full range of low-power modes to be used.

The internal clocks in the device default to the following configuration:

- MCLK: DCO, 2 MHz
- SMCLK: DCO, 2 MHz
- ACLK: REFO, 32.768 kHz

For more information about configuring internal clocks and using the external oscillators, see the *MSP430FR4xx and MSP430FR2xx Family User's Guide* (SLAU445).

2.6 Using the eZ-FET Emulator With a Different Target

The eZ-FET emulator on the LaunchPad can interface to most MSP430 derivative devices, not just the on-board MSP430FR4133 target device.

To do this, disconnect every jumper in the isolation jumper block. This is necessary, because the emulator cannot connect to more than one target at a time over the Spy-Bi-Wire (SBW) connection.

Next, make sure the target board has proper connections for SBW. Note that to be compatible with SBW, the capacitor on RST/SBWTDIO cannot be greater than 2.2 nF. The documentation for designing MSP430 JTAG interface circuitry is the *MSP430 Hardware Tools User's Guide* (SLAU278).

Finally, wire together these signals from the emulator side of the isolation jumper block to the target hardware:

- 5 V (if 5 V is needed)
- 3.3 V
- GND
- SBWTDIO
- SBWTCK
- TXD (if the UART backchannel is used)
- RXD (if the UART backchannel is used)
- CTS (if hardware flow control is used)
- RTS (if hardware flow control is used)

This wiring can be done either with jumper wires or by designing the board with a connector that plugs into the isolation jumper block.

2.7 BoosterPack Pinout

The LaunchPad adheres to the 20-pin LaunchPad pinout standard. A standard was created to aid compatibility between LaunchPad and BoosterPack tools across the TI ecosystem.

The 20-pin standard is compatible with the 40-pin standard that is used by other LaunchPads like the MSP-EXP430F5529LP. This allows some subset of functionality of 40-pin BoosterPacks to be used with 20-pin LaunchPads.

While most BoosterPacks are compliant with the standard, some are not. The MSP-EXP430FR4133 LaunchPad is compatible with all 20-pin BoosterPacks that are compliant with the standard. If the reseller or owner of the BoosterPack does not explicitly indicate compatibility with the MSP-EXP430FR4133 LaunchPad, compare the schematic of the candidate BoosterPack with the LaunchPad to ensure compatibility. Keep in mind that sometimes conflicts can be resolved by changing the MSP430FR4133 device pin function configuration in software. More information about compatibility can also be found at http://www.ti.com/launchpad.

Figure 10 shows the 20-pin pinout of the MSP430FR4133 LaunchPad.

Note that software's configuration of the pin functions plays a role in compatibility. The LaunchPad side of the dashed line in Figure 10 shows all of the functions for which the MSP430FR4133 device's pins can be configured. This can also be seen in the MSP430FR4133 data sheet. The BoosterPack side of the dashed line shows the standard. The LaunchPad function whose color matches the BoosterPack function shows the specific software-configurable function by which the MSP430FR4133 LaunchPad adheres to the standard.

The following pins are exposed at the BoosterPack connector.
Also shown are functions that map with the BoosterPack standard.
* Note that to comply with the I²C channels of the BoosterPack standard, a software-emulated I²C must be used.
** Some LaunchPads do not 100% comply with the standard, please check your LaunchPad to ensure compatability
(!) Denotes I/O pins that are interrupt-capable.

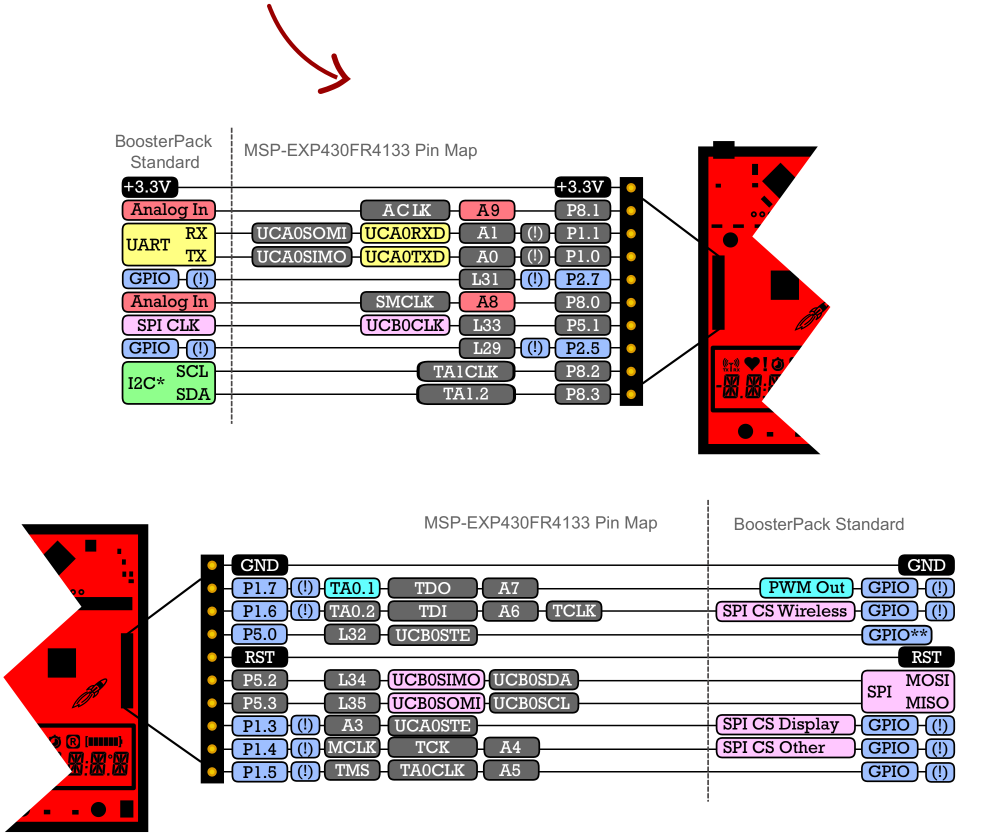

BoosterPack Standard
MSP-EXP430FR4133 Pin Map
+3.3V +3.3V
Analog In ACLK A9 P8.1
UART RX UCA0SOMI UCA0RXD A1 (!) P1.1
TX UCA0SIMO UCA0TXD A0 (!) P1.0
GPIO (!) L31 (!) P2.7
Analog In SMCLK A8 P8.0
SPI CLK UCB0CLK L33 P5.1
GPIO (!) L29 (!) P2.5
I2C* SCL TA1CLK P8.2
SDA TA1.2 P8.3
MSP-EXP430FR4133 Pin Map
BoosterPack Standard
GND GND
P1.7 (!) TA0.1 TDO A7 PWM Out GPIO (!)
P1.6 (!) TA0.2 TDI A6 TCLK SPI CS Wireless GPIO (!)
P5.0 L32 UCB0STE GPIO**
RST RST
P5.2 L34 UCB0SIMO UCB0SDA SPI MOSI
P5.3 L35 UCB0SOMI UCB0SCL MISO
P1.3 (!) A3 UCA0STE SPI CS Display GPIO (!)
P1.4 (!) MCLK TCK A4 SPI CS Other GPIO (!)
P1.5 (!) TMS TA0CLK A5 GPIO (!)

Figure 10. LaunchPad to BoosterPack Connector Pinout

2.8 Design Files

2.8.1 Hardware

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-EXP430FR4133 Hardware Design Files.

2.8.2 Software

All design files including TI-TXT object-code firmware images, software example projects, and documentation are available in the MSP-EXP430FR4133 Software Examples.

2.9 Hardware Change Log

Table 5. Hardware Change Log

PCB Revision	Description
Rev 1.0	Initial release
Rev 1.1	Added CE marking to silkscreen for compliance. No functional or layout change.

3 Software Examples

The software examples included with the MSP430FR4133 LaunchPad can be found in the MSP-EXP430FR4133 Software Examples.

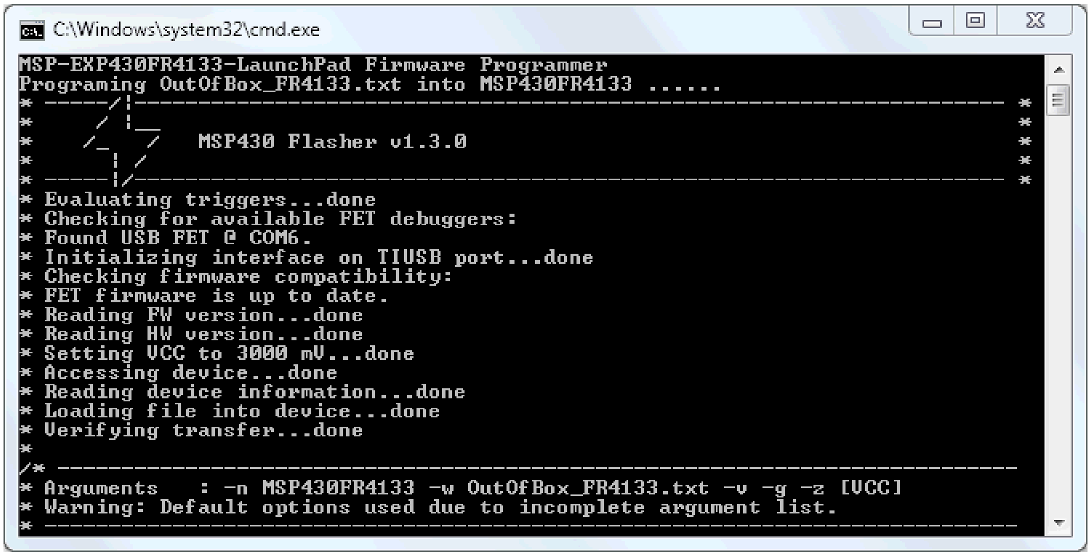
Table 6. Software Examples

Demo Name	BoosterPack Required	Description	Details
OutOfBox_FR4133	None	The out-of-box demo pre-programmed on the LaunchPad from the factory. Demonstrates features of MSP430FR4133 device	Section 3.4

3.1 Precompiled Binary

The /Binary/ folder inside the MSP-EXP430FR4133 Software Examples includes precompiled TI-TXT binary files for each of the projects in Table 6 that are ready to be flashed onto the LaunchPad. A copy of the MSP430Flasher tool is also shipped to interface with the eZ-FET Emulator.

To quickly program a demo onto the LaunchPad, simply navigate into the corresponding demo project's directory and double click the "Program <Example>.bat" file.



```
MSP-EXP430FR4133-LaunchPad Firmware Programmer
Programing OutOfBox_FR4133.txt into MSP430FR4133 ......
* -----/|-------------------------------------------------------------------- *
*     / |__                                                                   *
*    /_   /    MSP430 Flasher v1.3.0                                          *
*      | /                                                                    *
* -----|/-------------------------------------------------------------------- *
* Evaluating triggers...done
* Checking for available FET debuggers:
* Found USB FET @ COM6.
* Initializing interface on TIUSB port...done
* Checking firmware compatibility:
* FET firmware is up to date.
* Reading FW version...done
* Reading HW version...done
* Setting VCC to 3000 mV...done
* Accessing device...done
* Reading device information...done
* Loading file into device...done
* Verifying transfer...done
*
/* ---------------------------------------------------------------------------
* Arguments   : -n MSP430FR4133 -w OutOfBox_FR4133.txt -v -g -z [VCC]
* Warning: Default options used due to incomplete argument list.
* ----------------------------------------------------------------------------
```

Figure 11. Programming the LaunchPad With Program Batch Files

If desired, the "Program <Example>.bat" file can be modified to point to your own projects' binary file.

NOTE: After importing and compiling the software source code in an IDE such as CCS or IAR, the TI-TXT binary files located in the /Binary/ folder are not updated automatically. Copy the newly compiled binary from your IDE's /Workspace/Project/ directory and replace the "<Example>.txt" in /Binary/ for the batch file to program your own binary file.

3.2 MSP430Ware Library

The examples are built upon MSP430 libraries shown below that are available from TI. All libraries are available as part of MSP430Ware. Downloading CCS includes MSP430Ware along with TI Resource Explorer.

- Driver library (MSP430DRIVERLIB): a foundational MSP430 software library that is useful for interfacing with all MSP430 core functions and peripherals, especially clocks and power.
- Graphics library (MSP430-GRLIB): a library for interfacing MSP430 devices to dot-matrix LCD displays. Contains primitives for simple drawing as well as images and more.
- Capacitive Touch Library (CAPSENSELIBRARY): a library for capacitive touch sensing applications. This library supports the use of buttons, sliders, wheels and more. Highly configurable for each application.

When you begin your own development, you will need more information about these libraries than can be included in this user's guide. All of the information that you need is in MSP430Ware or the specific library documentation linked above.

3.3 Development Environment Requirements

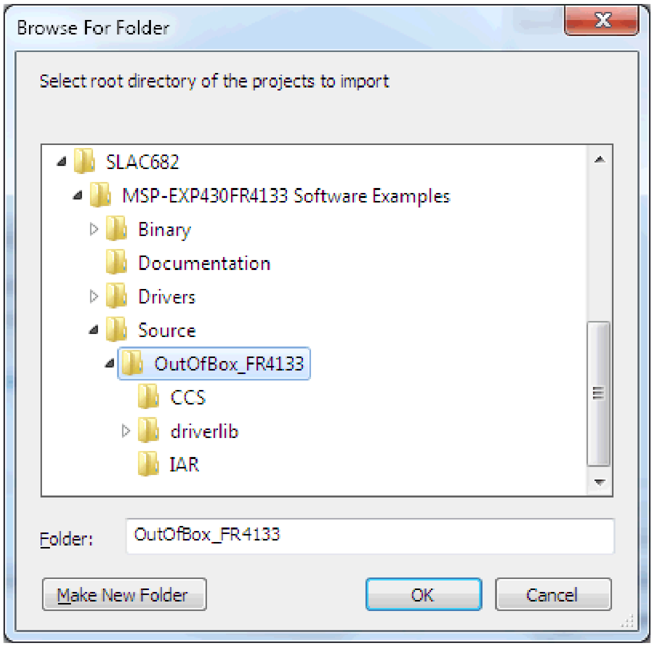
To use any of the software examples with the LaunchPad, you must have an IDE that supports the MSP430FR4133 device (see Table 7). For more details on where to download the latest IDE, see Section 4.3.

Table 7. IDE Minimum Requirements for MSP-EXP430FR4133

Code Composer Studio™ IDE	IAR Embedded Workbench™ IDE
CCS v6.0 or later	IAR EW430 v6.10 or later

3.3.1 CCS

CCS v6.0 or later is required. When CCS has been launched, and a workspace directory chosen, use Project>Import Existing CCS Eclipse Project. Direct it to the desired demo project directory that contains main.c (see Figure 12). Selecting the \CCS subdirectory also works. The CCS-specific files are located there.

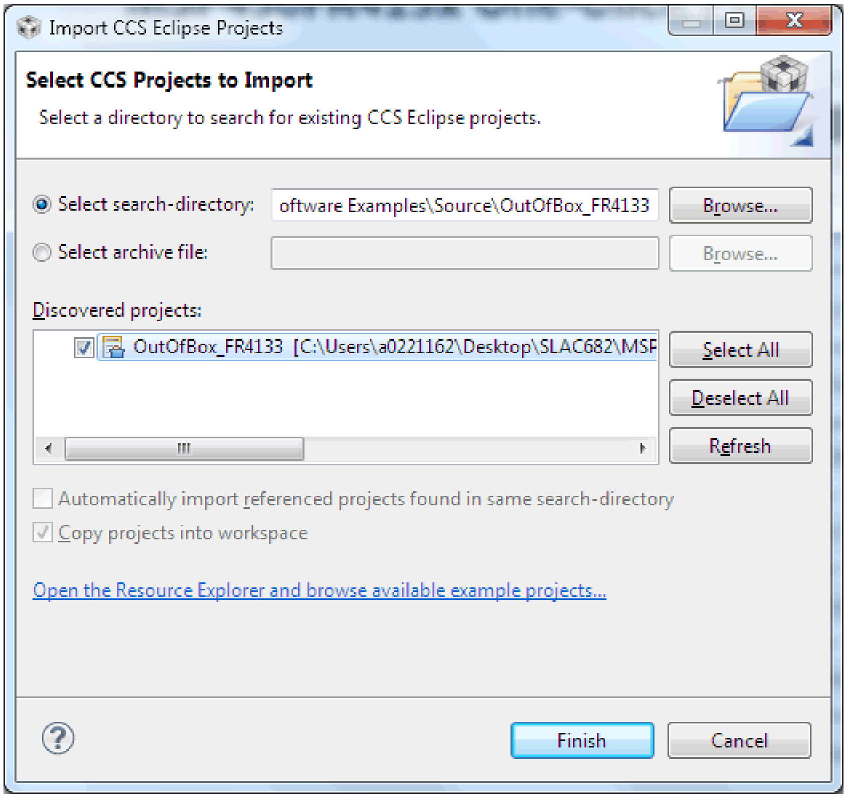


Select root directory of the projects to import

Folder: OutOfBox_FR4133

Figure 12. Directing the Project>Import Function to the Demo Project

When you click OK, CCS should recognize the project and allow you to import it. The indication that CCS has found the project is that it appears in "Discovered projects" and is checked (see Figure 13).



Import CCS Eclipse Projects

Select CCS Projects to Import

Select a directory to search for existing CCS Eclipse projects.

● Select search-directory: oftware Examples\Source\OutOfBox_FR4133 Browse...

○ Select archive file: Browse...

Discovered projects:

☑ OutOfBox_FR4133 [C:\Users\a0221162\Desktop\SLAC682\MSP

□ Automatically import referenced projects found in same search-directory

☑ Copy projects into workspace

Open the Resource Explorer and browse available example projects...

Figure 13. When CCS Has Found the Project

Sometimes CCS finds the project but does not show a checkmark; this might mean that your workspace already has a project by that name. You can resolve this by renaming or deleting that project. (Even if you do not see it in the CCS workspace, be sure to check the workspace directory on the file system.)

3.3.2 IAR

IAR 6.10 or higher is required. To open the demo in IAR, click File>Open>Workspace…, and browse to the *.eww workspace file inside the \IAR subdirectory of the desired demo. All workspace information is contained within this file.

The subdirectory also has an *.ewp project file. This file can be opened into an existing workspace by clicking Project>Add-Existing-Project….

Although the software examples have all of the code required to run them, IAR users may download and install MSP430Ware, which contains MSP430 libraries and the TI Resource Explorer. These are already included in a CCS installation (unless the user selected otherwise).

3.4 Out-of-Box Software Example

This section describes the functionality and structure of the out-of-box software that is preloaded on the EVM.

There are two modes in the out-of-box software, stopwatch mode and temperature sensor mode, which can be controlled with S1 and S2 push buttons on the LaunchPad. This demo shows how to utilize the LCD_E module (active in low power mode 3.5), combined with the RTC counter, ADC, and internal temperature sensor, to implement simple stopwatch and thermometer.

3.4.1 Source File Structure

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

Table 8. List of Source Files and Folders

Name	Description
main.c	The out-of-box demo main function, initializations, shared ISRs, and so on
hal_LCD.c	Hardware abstraction layer for LCD
StopWatchMode.c	Main function file for stopwatch mode
TempSensorMode.c	Main function file for live thermometer mode
Library: Driverlib	Device driver library (MSP430DRIVERLIB)

3.4.2 Power Up and Idle

Upon powering up the out-of-box demo, the LCD displays a scrolling welcome message. The MSP430FR4133 then enters a loop, in which the LCD cycles through all of its segments followed by a scrolling instruction message to "Hold S1 and S2 to switch modes".

3.4.3 Stopwatch Mode

While in the power up and idle state or in the temperature sensor mode, the stopwatch mode can be entered by holding down both S1 and S2 buttons shortly. The LCD displays scrolling text "STOPWATCH MODE" to indicate successful entry into this mode.

The MSP430FR4133 initializes the stopwatch calendar to HH:MM:SS:CC = 00:00:00:00, then goes to sleep in LPM3.5. Since the onboard LCD has 6 alphanumeric digits, the stopwatch format is initially MM:SS:CC but becomes HH:MM:SS when the timer reaches the first hour.

Press the S1 button to start the stopwatch timer (counts up). While the timer is running, the MSP430FR4133 sleeps and wakes between LPM3 (waiting for RTC interrupt) and active mode (incrementing calendar and updating LCD). Press the S1 button again to stop the stopwatch timer and return the MSP430FR4133 back to LPM3.5 to conserve power. When the stopwatch timer is stopped, press the S2 button resets the timer back to 00:00:00.

While the stopwatch timer is running, press the S2 button to pause the LCD at the current time but continue the timer running in the background, allowing for the "split timer" functionality. Press the S2 button to resume display of the running timer on the LCD.

3.4.4 Temperature Sensor Mode

While in the stopwatch mode, the temperature sensor mode can be entered by holding down both S1 and S2 buttons shortly. The LCD displays scrolling text "TEMPSENSOR MODE" to indicate successful entry into this mode.

Upon entering this mode, the MSP430FR4133 initializes the ADC input to its internal temperature sensor and starts sampling/conversion at four times per second. Each time an ADC conversion completes, the LCD shows the calculated temperature to the tenths decimal place.

The temperature unit can be toggled between Celsius and Fahrenheit by pressing the S2 button.

The temperature measurement can also be paused/resumed by pressing the S1 button. While the temperature measurement is running, the MSP430FR4133 sleeps and wakes between LPM3 (waiting for ADC sample/conversion to finish) and active mode (processing the results and updating LCD). When the temperature measurement is paused, the MSP430FR4133 enters LPM3.5 with the LCD remaining on displaying the last measured temperature.

4 Additional Resources

4.1 LaunchPad Websites

More information about the MSP430FR4133 LaunchPad, supported BoosterPacks, and available resources can be found at:

- Tool Folder: resources specific to this particular LaunchPad
- TI's LaunchPad portal: information about all LaunchPads from TI for all MCUs

4.2 Information on the MSP430FR4133

At some point, you will probably want more information about the MSP430FR4133 device. For every MSP430 device, the documentation is organized as shown in Table 9.

Table 9. How MSP430 Device Documentation is Organized

Document	For MSP430FR4133	Description
Device family user's guide	*MSP430FR4xx and MSP430FR2xx Family User's Guide* (SLAU445)	Architectural information about the device, including all modules and peripherals such as clocks, timers, ADC, and so on.
Device-specific data sheet	*MSP430FR413x Mixed-Signal Microcontrollers* data sheet (SLAS865)	Device-specific information and all parametric information for this device

4.3 Download CCS or IAR

Although the 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) or IAR Embedded Workbench.

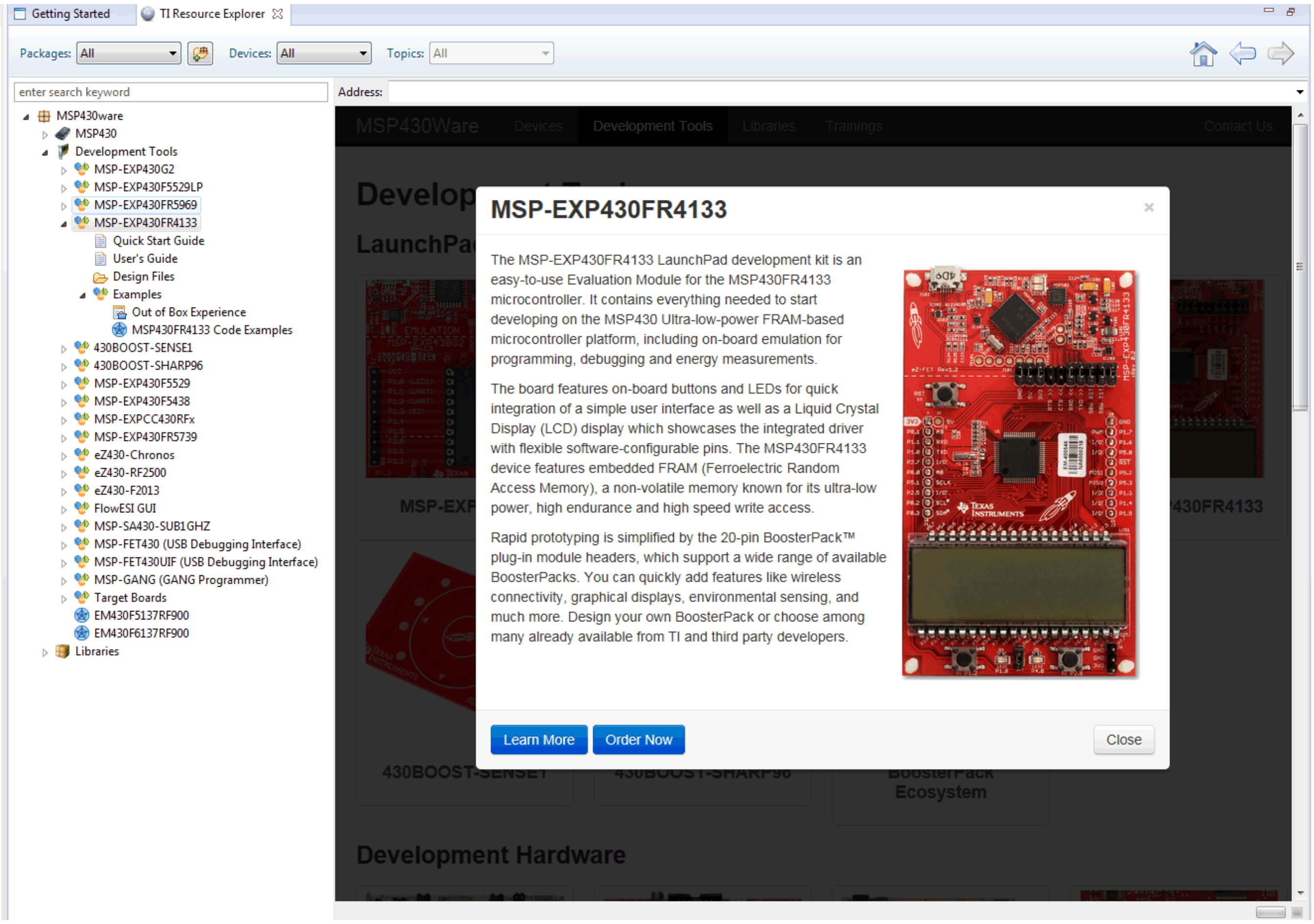
CCS and IAR are available in full, free code-size, or time limited versions. The full out-of-box demo can be built with the free versions of CCS and IAR (IAR KickStart), because it is under the code size limit. For full functionality, download the full version of either CCS or IAR.

Go to the MSP430 software tools page to download them and for instructions on installation.

4.4 MSP430Ware and TI Resource Explorer

MSP430Ware is a complete collection of libraries and tools. It includes all of the MSP430 libraries used in the software examples. By default, MSP430Ware is included in a CCS installation. IAR users must download it separately.

MSP430Ware is built into the TI Resource Explorer, for easily browsing tools. For example, all of the software examples are shown in Figure 14.



MSP-EXP430FR4133

The MSP-EXP430FR4133 LaunchPad development kit is an easy-to-use Evaluation Module for the MSP430FR4133 microcontroller. It contains everything needed to start developing on the MSP430 Ultra-low-power FRAM-based microcontroller platform, including on-board emulation for programming, debugging and energy measurements.

The board features on-board buttons and LEDs for quick integration of a simple user interface as well as a Liquid Crystal Display (LCD) display which showcases the integrated driver with flexible software-configurable pins. The MSP430FR4133 device features embedded FRAM (Ferroelectric Random Access Memory), a non-volatile memory known for its ultra-low power, high endurance and high speed write access.

Rapid prototyping is simplified by the 20-pin BoosterPack™ plug-in module headers, which support a wide range of available BoosterPacks. You can quickly add features like wireless connectivity, graphical displays, environmental sensing, and much more. Design your own BoosterPack or choose among many already available from TI and third party developers.

Figure 14. MSP-EXP430FR4133 Software Examples in TI Resource Explorer

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

4.5 MSP430FR4133 Code Examples

This is a set of very simple code examples (SLAC625) that demonstrate how to use the entire set of MSP430 peripherals: ADC12, Timer_A, eUSCI, and so on. These do not use driver library, rather they access the MSP430 registers directly.

Every MSP430 derivative has a set of these code examples. When writing code that uses a peripheral, they can often serve as a starting point.

Code examples can be accessed directly through MSP430Ware and the TI Resource Explorer without downloading the code examples mentioned above. TI Resource Explorer allows a one-click import of the device example code.

4.6 MSP430 Application Notes

There are many application notes at www.ti.com/msp430, with practical design examples and topics.

4.7 Community Resources

4.7.1 TI E2E™ Community

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

4.7.2 Community at Large

Many online communities focus on the LaunchPad – for example, http://www.43oh.com. You can find additional tools, resources, and support from these communities.

5 FAQ

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 USCI settings match?
- Are the appropriate jumpers in place, on the isolation jumper block?
- Probe on RXD and send data from the host. If you don't see data, it might be a problem on the host side.
- Probe on TXD while sending data from the MSP430. If you don't 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).

Q: So the onboard emulator is really open source? And I can build my own onboard emulator?

A: Yes! We encourage you to do so. The design files are on ti.com.

Q: The MSP430 G2 LaunchPad had a socket, allowing me change the target device. Why doesn't this LaunchPad use one?

A: This LaunchPad provides more functionality, and this means using a device with more pins. Sockets for devices with this many pins are too expensive for the LaunchPad's target price.

Q: With the female headers on the bottom, the board doesn't sit flat on the table, and I can't unsolder them. Why did TI do this?

A: For several reasons. A major feedback item on previous LaunchPads was the desire for female headers instead of male ones. But simply using female instead is problematic, because compatibility with existing BoosterPacks would be lost, and some people prefer male headers. So, adding female headers without removing male ones satisfies both preferences. It also allows more flexibility in stacking BoosterPacks and other LaunchPads.

6 Schematics

The following figures show the schematics for the MSP-EXP430FR4133.

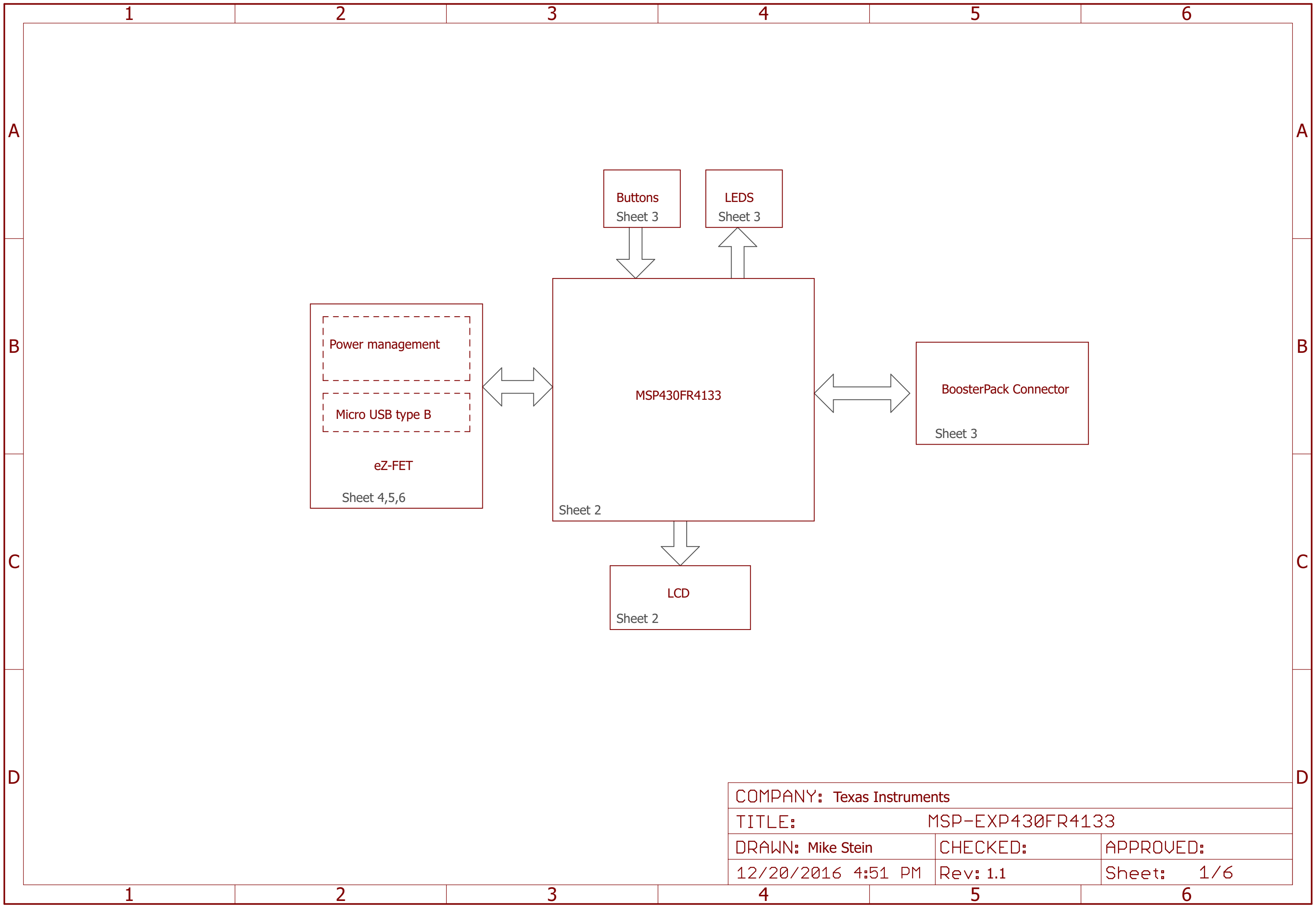


Figure 15. Schematics (1 of 6)

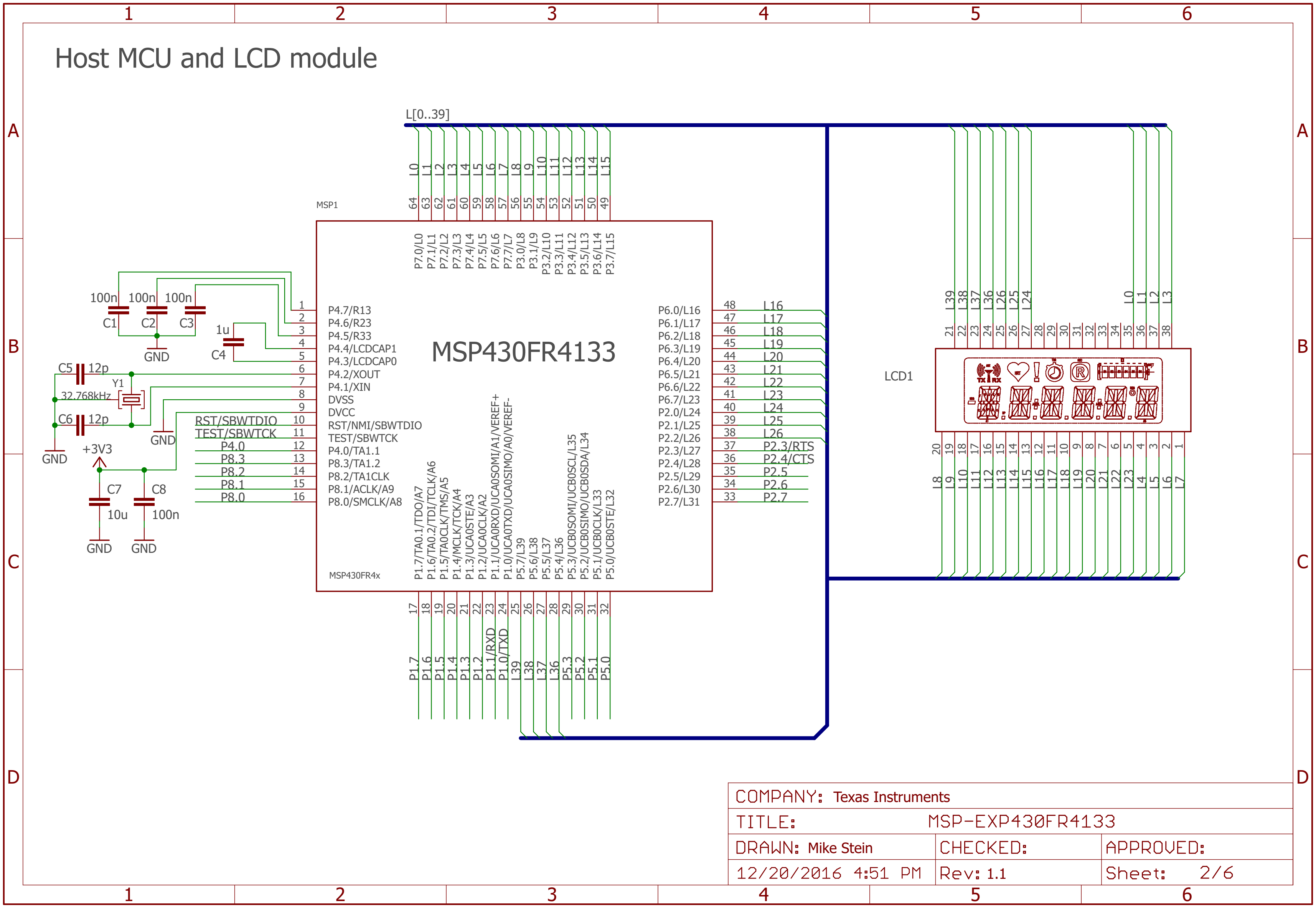


Figure 16. Schematics (2 of 6)

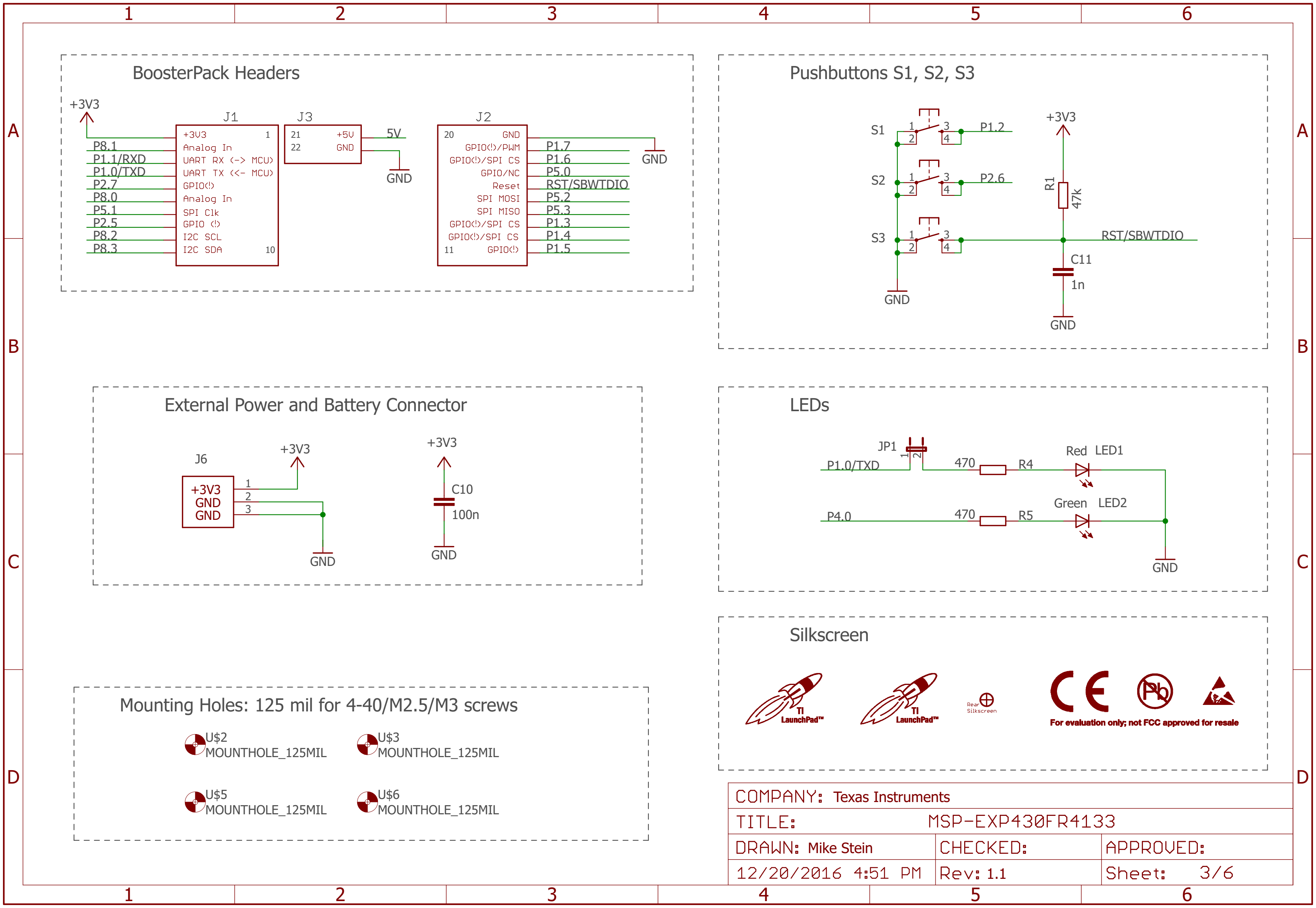


Figure 17. Schematics (3 of 6)

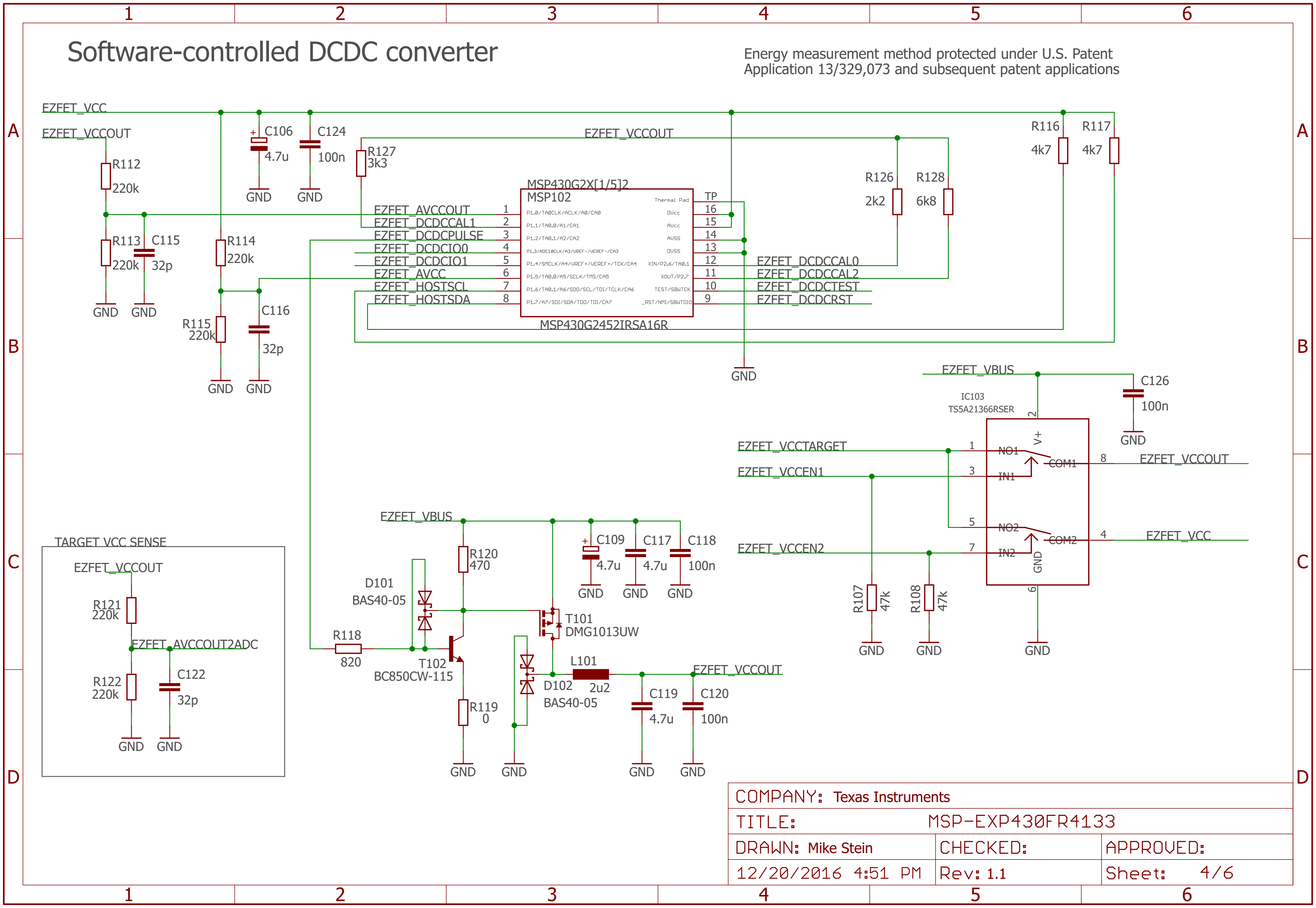
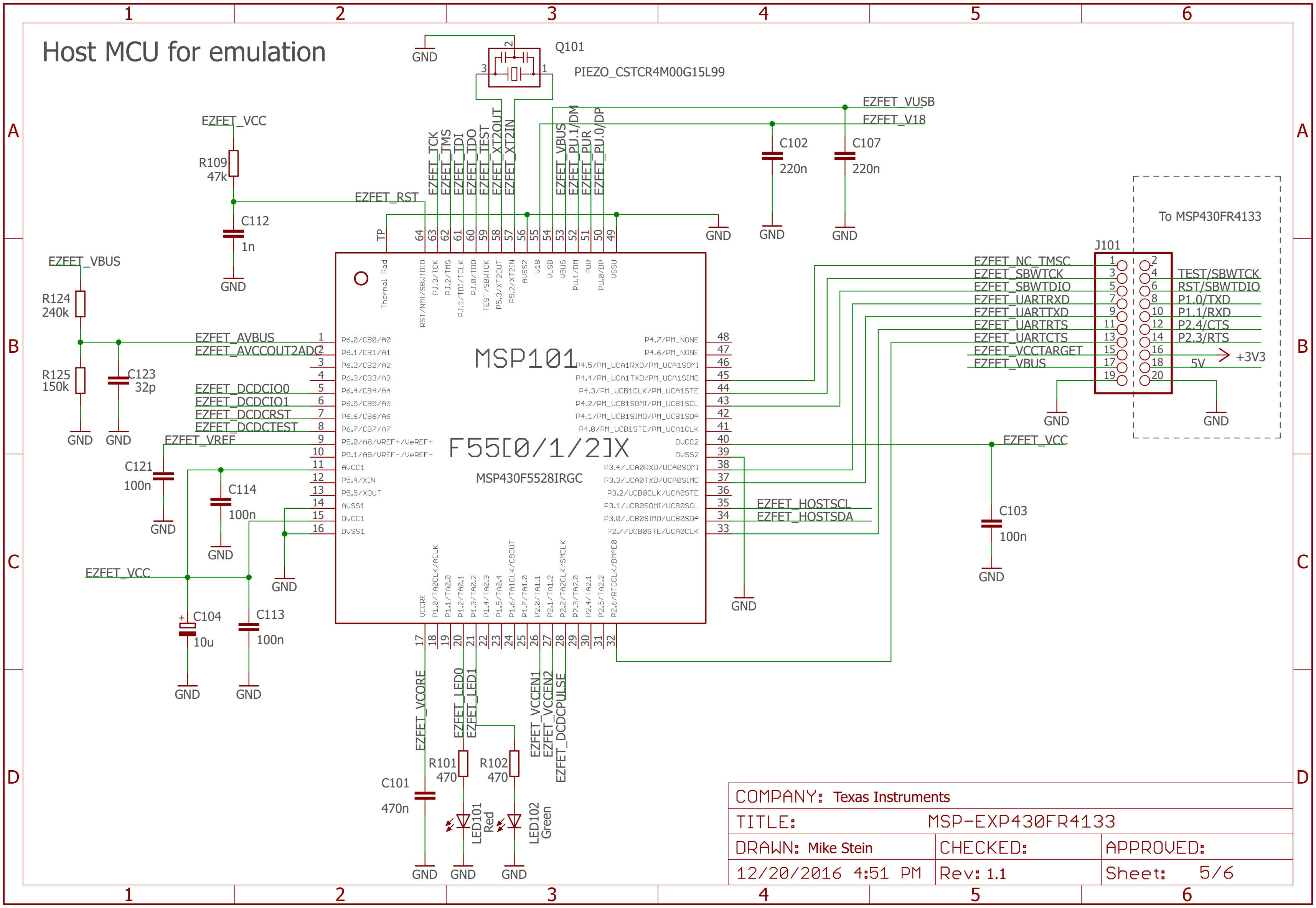


Figure 18. Schematics (4 of 6)



Copyright © 2017, Texas Instruments Incorporated

Figure 19. Schematics (5 of 6)

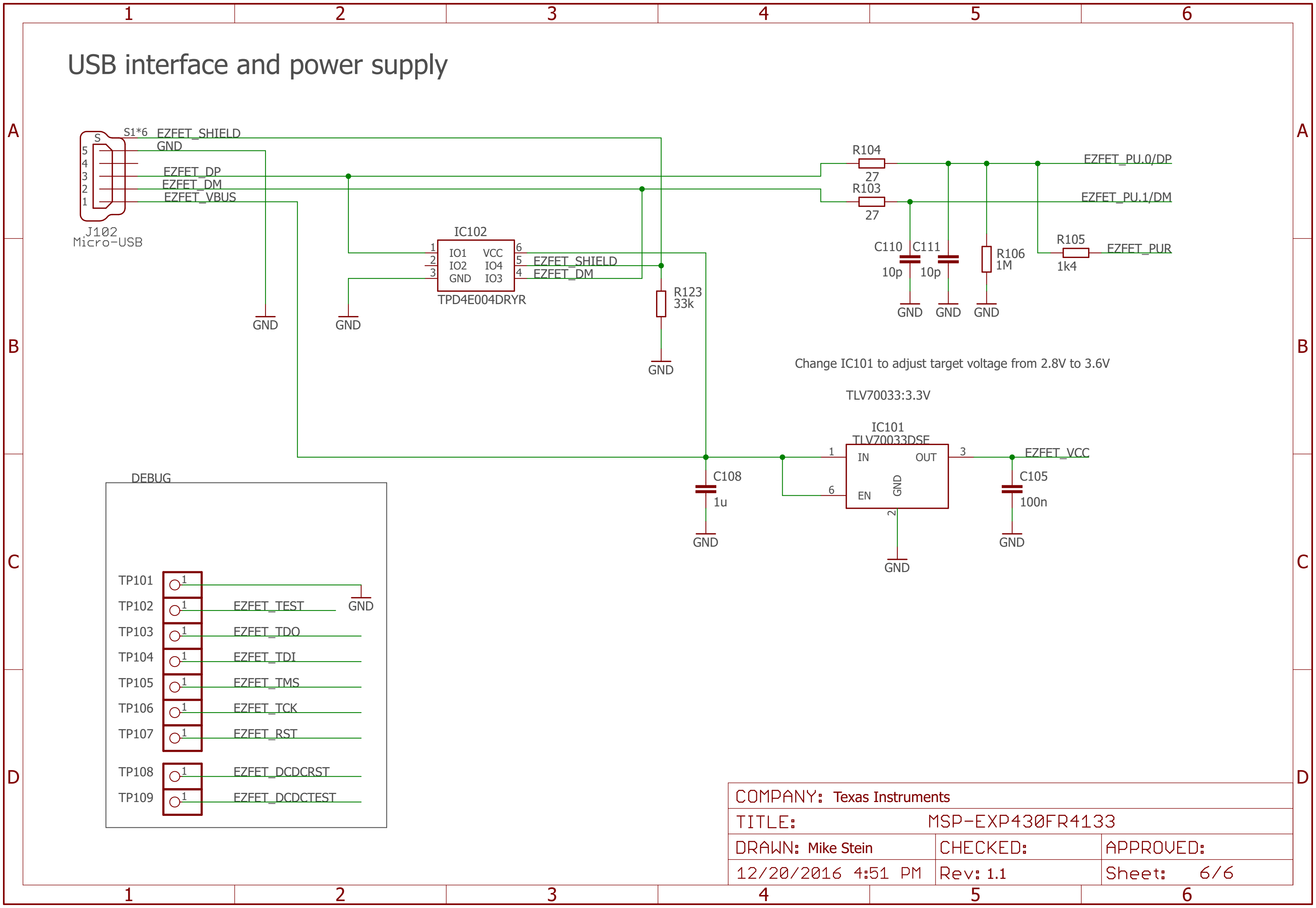


Figure 20. Schematics (6 of 6)

Revision History

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

Changes from July 21, 2015 to January 17, 2017	**Page**
• Added Rev 1.1 to Table 5, *Hardware Change Log*	17
• Updated all figures in Section 6, *Schematics*	25

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