MSP-EXP430FR5969 LaunchPad™ Development Kit

User's Guide

Literature Number: SLAU535A
February 2014–Revised June 2014

Contents

1 Getting Started 5
- 1.1 Introduction 5
- 1.2 Key Features 6
- 1.3 Kit Contents 6
- 1.4 First Steps – Out-of-Box Experience 6
- 1.5 Next Steps – Looking Into the Provided Code 7

2 Hardware 7
- 2.1 Block Diagram 8
- 2.2 MSP430FR5969 8
- 2.3 eZ-FET Onboard Emulator With EnergyTrace™ Technology 11
- 2.4 Power 18
- 2.5 BoosterPack Plug-in Module Headers 22
- 2.6 Design Files 23
- 2.7 Hardware Change log 24

3 Software Examples 24
- 3.1 MSP430 Software: Driver Library, Graphics Library, and Capacitive Touch Library 24
- 3.2 Development Environment Requirements 25
- 3.3 Out-of-Box Software Example 27
- 3.4 430BOOST-SHARP96 ULP FRAM Demo 30
- 3.5 430BOOST-SHARP96 Graphics Library Demo 34

4 Additional Resources 36
- 4.1 LaunchPad Websites 36
- 4.2 Information on the MSP430FR5969 36
- 4.3 Download CCS, IAR, or MSPGCC 36
- 4.4 MSP430Ware and TI Resource Explorer 37
- 4.5 MSP430FR5969 Code Examples 37
- 4.6 MSP430 Application Notes 37
- 4.7 The Community 38

5 FAQs 38

6 Schematics 39

Revision History 44

List of Figures

1 MSP-EXP430FR5969 5
2 EVM Overview 7
3 Block Diagram 8
4 MSP430FR5969 Pinout 9
5 eZ-FET Emulator 11
6 Application Backchannel UART in Device Manager 12
7 EnergyTrace Technology Settings 13
8 Debug Properties 14
9 Debug Session With EnergyTrace++ Windows 15
10 eZ-FET Isolation Jumper Block Diagram 16
11 MSP430FR5969 LaunchPad Power Domain Block Diagram 18
12 Debugger Power Configuration – USB eZ-FET and JTAG 19
13 External Power Configuration – External and BoosterPack 20
14 Super Cap Power Configuration – Charging and Running Standalone 22
15 FR5969 LaunchPad to BoosterPack Connector Pinout 23
16 Program <Example>.bat 25
17 Directing the Project→Import Function to the Demo Project 26
18 When CCS Has Found the Project 27
19 Live Temperature Mode 28
20 FRAM Log Mode 29
21 FRAM Unified Memory With Dynamic Partitioning 35
22 MSP-EXP430FR5969 Software Examples in TI Resource Explorer 37
23 Schematic 1 of 5 39
24 Schematic 2 of 5 40
25 Schematic 3 of 5 41
26 Schematic 4 of 5 42
27 Schematic 5 of 5 43

List of Tables

1 EnergyTrace++ Debug Windows 14
2 EnergyTrace™ Technology Control Bar Icons 15
3 Isolation Block Connections 16
4 Hardware Change Log 24
5 Software Examples 24
6 IDE Minimum Requirements for MSP430FR5969 25
7 Source Files and Folders 28
8 Source Files and Folders 30
9 FRAM Endurance Calculation for 1KB Block of FRAM 32
10 How MSP430 Device Documentation is Organized 36

MSP-EXP430FR5969 LaunchPad™ Development Kit User's Guide

1 Getting Started

1.1 Introduction

MSP430™ ultra-low-power (ULP) MCUs with embedded Ferroelectric Random Access Memory (FRAM) technology now join the MCU LaunchPad™ Development Kit ecosystem. The MSP-EXP430FR5969 (or the "FR5969 LaunchPad") is an easy-to-use evaluation module (EVM) for the MSP430FR5969 microcontroller. It contains everything needed to start developing on the MSP430 FRAM platform, including on-board emulation for programming, debugging, and energy measurements. The board features buttons and LEDs for quick integration of a simple user interface as well as a super capacitor (super cap) that enables standalone applications without an external power supply.

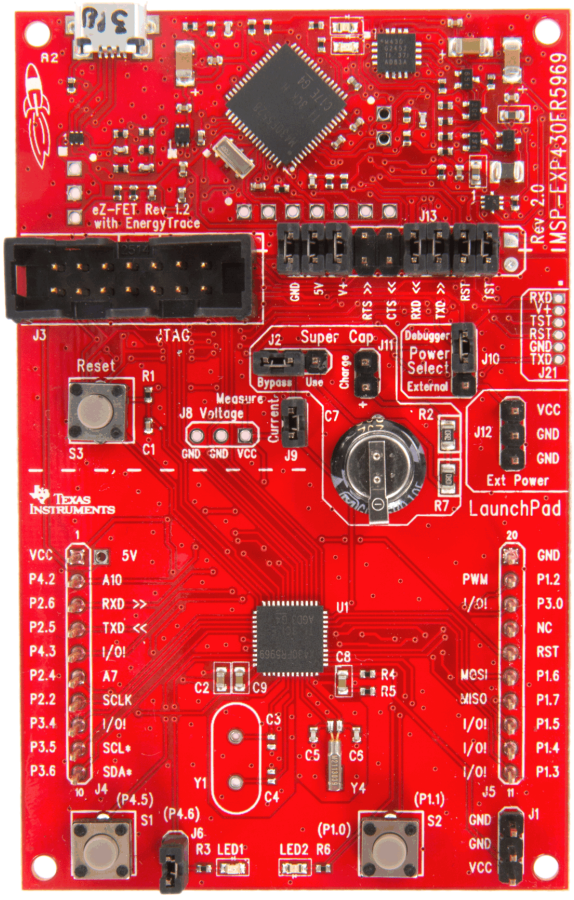

Figure 1. MSP-EXP430FR5969

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. You can either design your own BoosterPack or choose among many already available from TI and third-party developers.

The MSP430FR5969 device features 64KB of embedded FRAM, a nonvolatile memory known for its ultra-low power, high endurance, and high-speed write access. The device supports CPU speeds up to 16 MHz and has integrated peripherals for communication, ADC, timers, AES encryption, and more – plenty to get you started in your development.

Free software development tools are also available - TI's Eclipse-based Code Composer Studio™ IDE (CCS) and IAR Embedded Workbench™ IDE (IAR), and the community-driven Energia open-source code editor. More information about the LaunchPad including documentation and design files can be found on the tool page at www.ti.com/tool/msp-exp430fr5969.

1.2 Key Features

- MSP430 ultra-low-power FRAM technology based MSP430FR5969 16-bit MCU
- 20-pin LaunchPad standard that leverages the BoosterPack ecosystem
- 0.1-F super capacitor for standalone power
- Onboard eZ-FET emulation with EnergyTrace++™ Technology
- Two buttons and two LEDs for user interaction
- Backchannel UART through USB to PC

1.3 Kit Contents

- 1 x MSP-EXP430FR5969
- 1 x Micro USB cable
- 1 x Quick Start Guide

1.4 First Steps – Out-of-Box Experience

An easy way to get familiar with the EVM is by using its pre-programmed out-of-box demo code, which demonstrates some key features of the MSP-EXP430FR5969 LaunchPad.

The out-of-box demo showcases MSP430FR5969's ultra-low power FRAM by utilizing the device's internal temperature sensor while running only off of the on-board Super Capacitor.

First step is to connect the LaunchPad to the computer using the included Micro-USB cable.

The RED and GREEN LEDs near the bottom of the LaunchPad toggle a few times to indicate the pre-programmed out-of-box demo is running.

Using the out-of-box demo GUI, included in the MSP-EXP430FR5969 Software Examples download (SLAC645), the user can place the LaunchPad into two different modes:

- Live Temperature Mode

  This mode provides live temperature data streaming to the PC GUI. The user is able to influence the temperature of the device and see the changes on the GUI.
- FRAM Logging Mode

  This mode shows the FRAM data logging capabilities of the MSP430FR5969. After starting this mode, the LaunchPad will wake up every five seconds from sleep mode (indicated by LED blink) to log both temperature and input voltage values. After reconnecting to the GUI, these values can be uploaded and graphed in the GUI.

A more detailed explanation of each mode can be found in Section 3.

1.5 Next Steps – Looking Into the Provided Code

After the out-of-box demo, more features can be explored. The hardware features on the LaunchPad are shown in Section 2, and the provided code examples and how to use them are in Section 3. More details and documentation can be found at http://www.ti.com/tool/msp-exp430fr5969. Code is licensed under BSD and TI encourages reuse and modifications to fit your needs.

2 Hardware

Figure 2 shows an overview of the LaunchPad hardware.

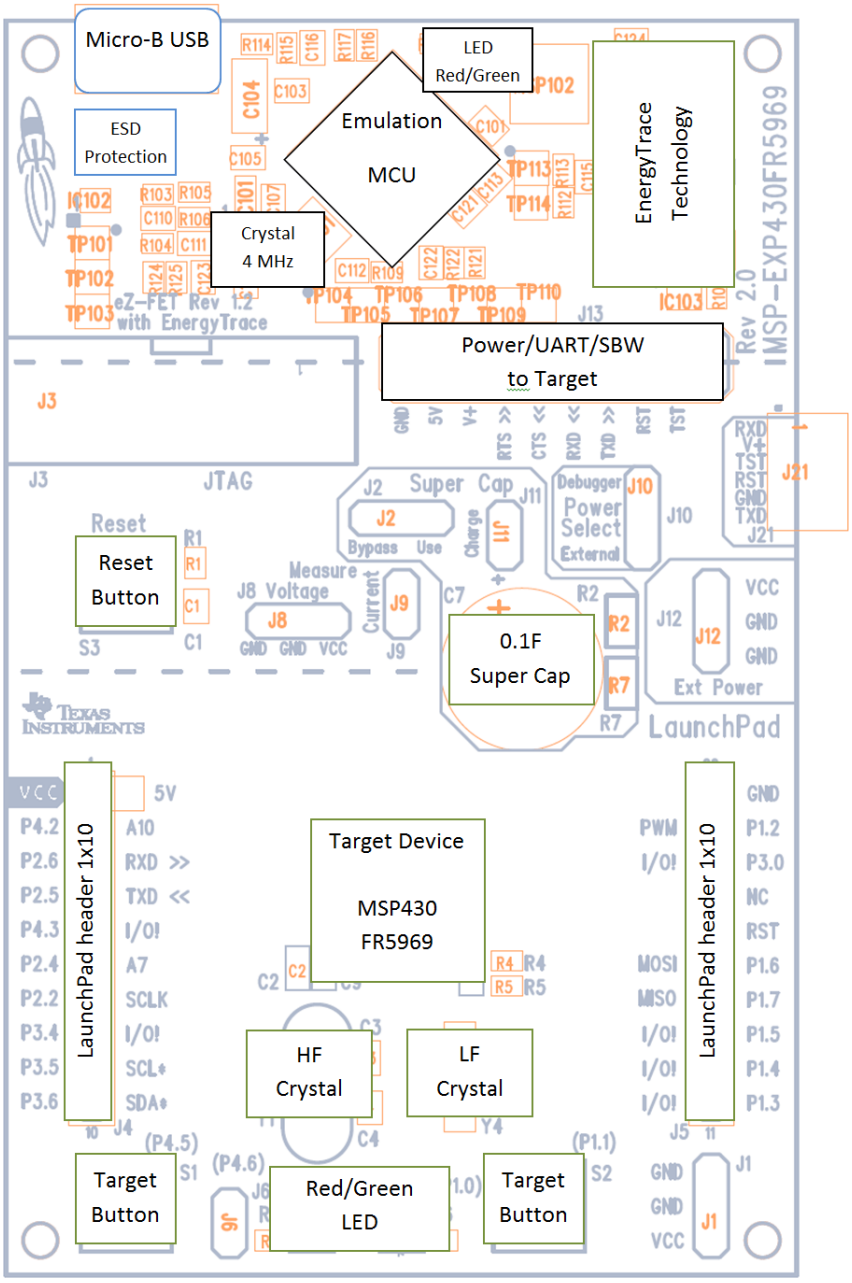

Figure 2. EVM Overview

2.1 Block Diagram

Figure 3 shows the block diagram.

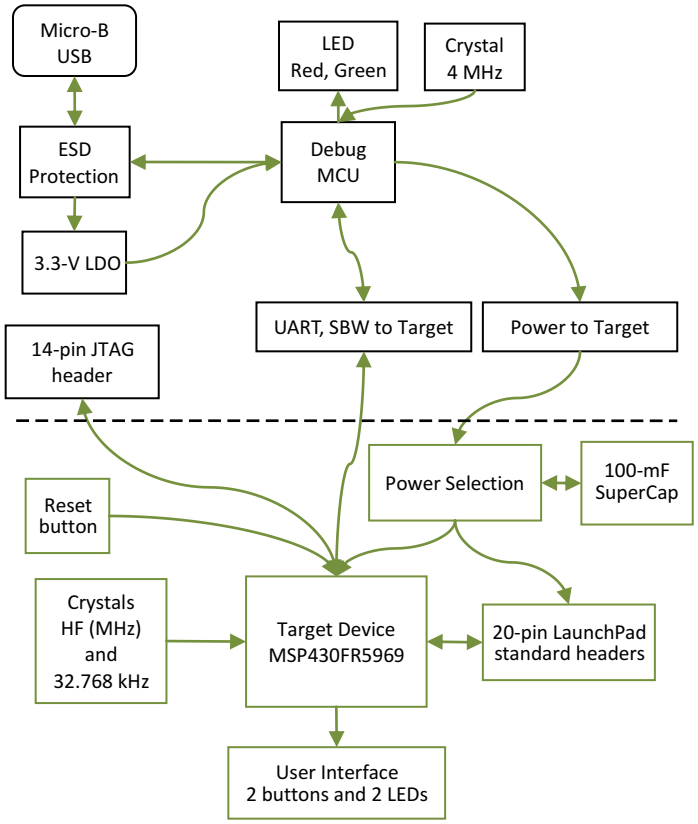

Figure 3. Block Diagram

2.2 MSP430FR5969

The MSP430FR5969 is the first 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. More information on FRAM can be found at www.ti.com/fram.

Device features include:

- 1.8-V to 3.6-V operation
- Up to 16-MHz system clock and 8-MHz FRAM access
- 64KB FRAM and 2KB SRAM
- Ultra-low-power operation
- Five timer blocks and up to three serial interfaces (SPI, UART, or I^2C)
- Analog: 16-channel 12-bit differential ADC and 16-channel comparator
- Digital: AES256, CRC, DMA, and hardware MPY32

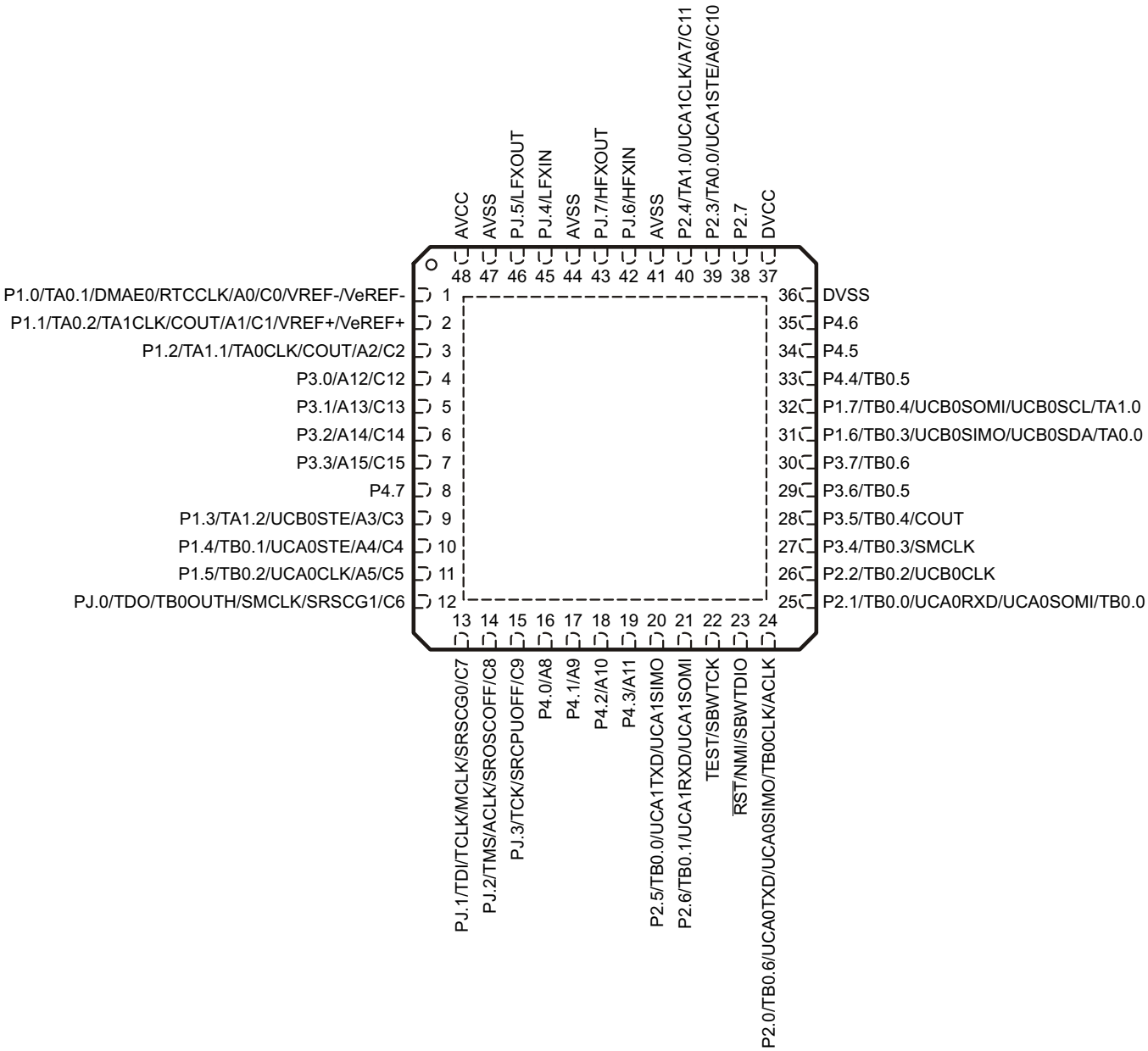

Figure 4. MSP430FR5969 Pinout

To compare the various MSP430 derivatives, download the MSP430 Product Brochure (SLAB034), which is also available from http://www.ti.com/msp430. The brochure has a table that lets you see, at a glance, how the families compare, and their pricing. This document is frequently updated, as new MSP430 derivatives become available.

2.2.1 Measure MSP430 Current Draw

A specific jumper J9 is placed on the LaunchPad to allow for measuring current draw of the MSP430FR5969 device. The current measured includes the FR5969, and any current drawn through the BoosterPack headers and jumper J1.

To measure ultra-low power, follow these steps:

1. Remove the J9 jumper; attach an ammeter across this jumper.
2. Consider the effect that the backchannel UART and any circuitry attached to the FR5969 may have on current draw. Maybe these should be disconnected at the isolation jumper block, or their current sinking and sourcing capability at least considered 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 an input, should be pulled or driven to a high or low level.
4. Begin target FR5969 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.)

2.2.2 Clocking

The FR5969 LaunchPad provides external clocks in addition to the internal clocks in the device.

- Y4: a 32-kHz crystal
- Y1: an unpopulated region that supports HF crystal or resonator (4 to 24 MHz)

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.

For more information about internal clocks and how to use the 32-kHz or HF crystal, see the MSP430FR59xx family user's guide.

2.3 eZ-FET Onboard Emulator With EnergyTrace™ Technology

To keep development easy and cost effective, TI's LaunchPad development tools integrate an onboard emulator, eliminating the need for expensive programmers.

The FR5969 LaunchPad has the new eZ-FET emulator (see Figure 5), a simple and low-cost debugger that supports almost all MSP430 device derivatives.

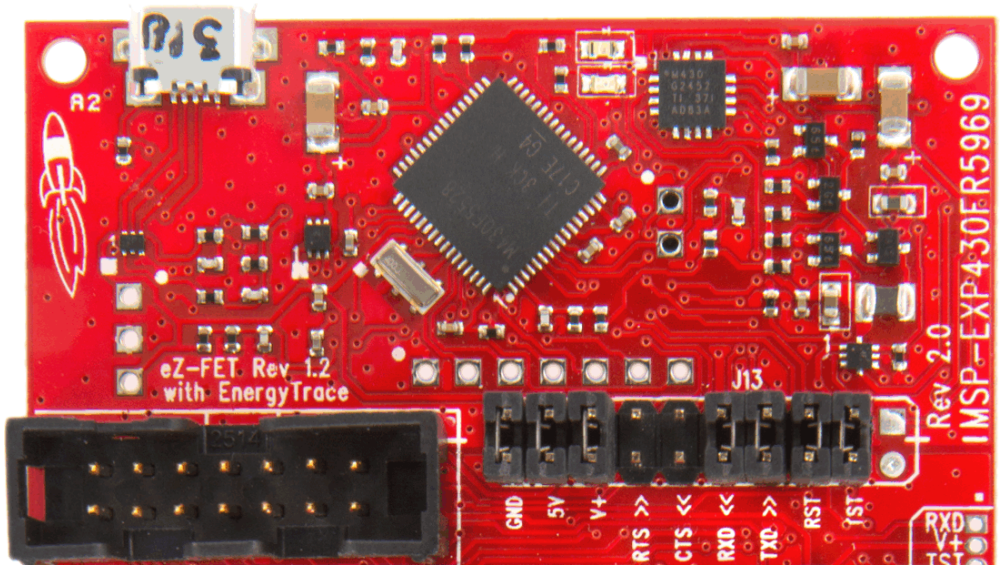

Figure 5. eZ-FET Emulator

The eZ-FET provides many features that make debugging an easy experience. Included is a "backchannel" UART-over-USB connection with the host, EnergyTrace Technology, and a distinct isolation from the target side microcontroller.

The eZ-FET hardware can be found in the schematics in Section 6 and in the accompanying hardware design files (SLAR091). The eZ-FET software and more information about the debugger can be found at the eZ-FET lite wiki.

2.3.1 Emulator/Debugger

The eZ-FET is an on-board emulation solution for MSP430 microcontrollers. It allows direct interfacing to a PC for easy programming, debugging, and evaluation. The eZ-FET uses Spy-Bi-Wire (SBW) two-wire protocol to interface with the MSP430 devices. These pins are the SBW RST and SBW TST pins located on the emulator isolation block. More details on the isolation block can be found in Section 2.3.4.

The eZ-FET on-board emulation is supported by the MSP430 DLL and can be used with IAR Embedded Workbench for MSP430 Integrated Development Environment (IDE) or Code Composer Studio (CCS) IDE to write, download, and debug applications.

The debugger is unobtrusive and allows the user to run an application at full speed with hardware breakpoints and single stepping available while consuming no extra hardware resources. The firmware for the eZ-FET is field updateable, so any updates can be received as needed.

More information on SBW can be found in the *MSP430 Hardware Tools User's Guide* (SLAU278), and information on the MSP430 DLL can be found at www.ti.com/mspds.

2.3.2 Application (or "Backchannel") UART

The backchannel UART allows communication with the USB host that isn't 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 FR5969 LaunchPad.

The pathway of the backchannel UART is shown in Figure 10. The backchannel UART (USCI_A0) is independent of the UART on the 20-pin BoosterPack connector (USCI_A1).

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 6).

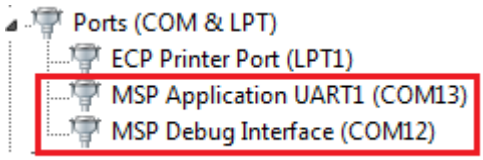

Figure 6. Application Backchannel UART in Device Manager

The backchannel UART is the "MSP Application UART1" port. In this case, Figure 6 shows COM13, but this varies 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 talking to it from the host.

On the target FR5969 side, the backchannel is connected to the USCI_A0 module.

The eZ-FET has a configurable baudrate, therefore, it is important that the PC application configures the baudrate to be the same as what's configured on the USCI_A0.

The eZ-FET also supports hardware flow control, if desired. Hardware flow control (CTS/RTS handshaking) allows the target FR5969 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 faster baud rates and more interrupts to service have a higher likelihood that they cannot read the USCI_A0's RXBUF register in time, before the next byte arrives. If this happens, the USCI_A0's UCA0STATW register will report an overrun error.

2.3.3 EnergyTrace™ Technology

EnergyTrace™ Technology is an energy-based code analysis tool set that is useful for measuring and viewing the application's energy profile and optimizing it for ultra-low power consumption. The MSP430FR5969 device supports EnergyTrace++ (EnergyTrace + [CPU State] + [Peripheral States]) for full access to the application energy profile as well as CPU and peripheral states.

By default, EnergyTrace technology is disabled in CCS. To enable EnergyTrace, click Window → Preferences → Code Composer Studio → Advanced Tools → EnergyTrace™ Technology. Select the "Enable" checkbox and the EnergyTrace++ setting for full functionality (see Figure 7).

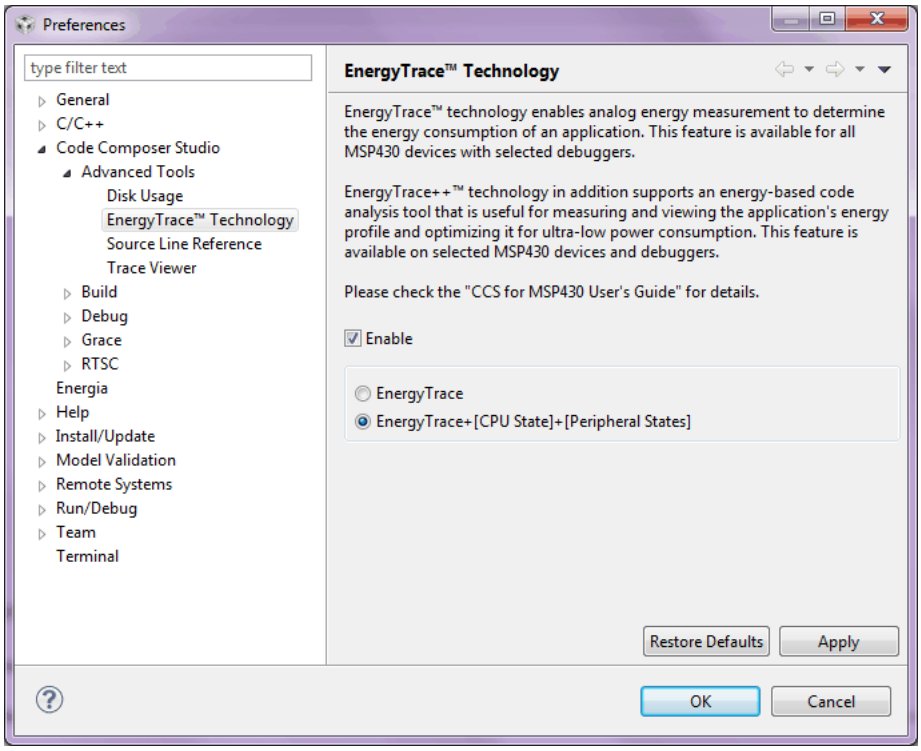

Figure 7. EnergyTrace Technology Settings

To fully enable the EnergyTrace++ setting, the ultra-low-power debug mode must be enabled. Right click on the active project in the project explorer and click Properties. In the Debug section, enable "Ultra Low Power debug/ Debug LPMx.5" option in the Low Power Mode Settings (see Figure 8). If this option is not enabled, the EnergyTrace++ mode cannot capture data from the device.

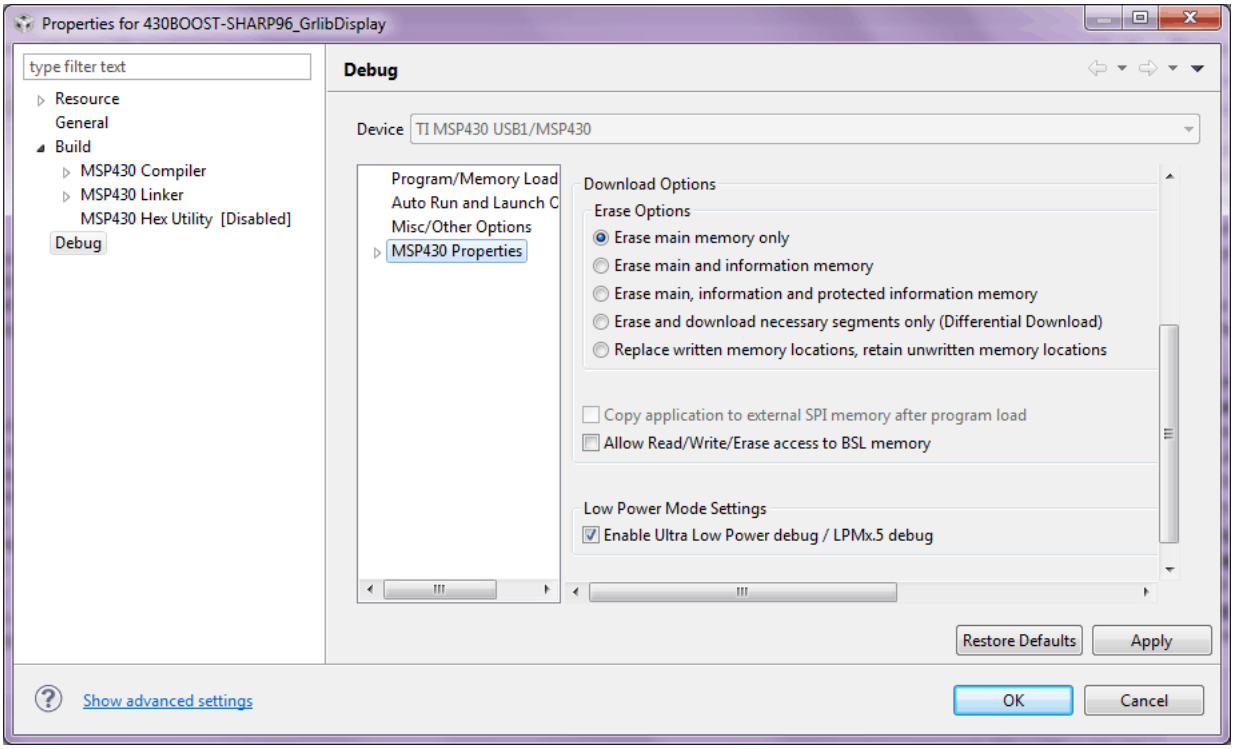

Figure 8. Debug Properties

After the correct settings are chosen, the EnergyTrace window automatically opens when debug is started. The EnergyTrace++ window has four separate tabs: Profile, States, Power, and Energy.

Table 1. EnergyTrace++ Debug Windows

EnergyTrace++ Debug Window	Description
Profile	Displays a compressed view of captured data and allows comparison with previous data
States	Real-time trace of the target microcontroller's internal states captured. Includes power modes and peripheral on/off states.
Power	Dynamic power consumption of the target over time. A previous trace profile for comparison is yellow in color.
Energy	Accumulated energy of the target over time. A previous trace profile for comparison is yellow in color.

These tabs allow you to see exactly what is happening in your application, and find power black holes. An example application with the EnergyTrace++ windows is shown in Figure 9.

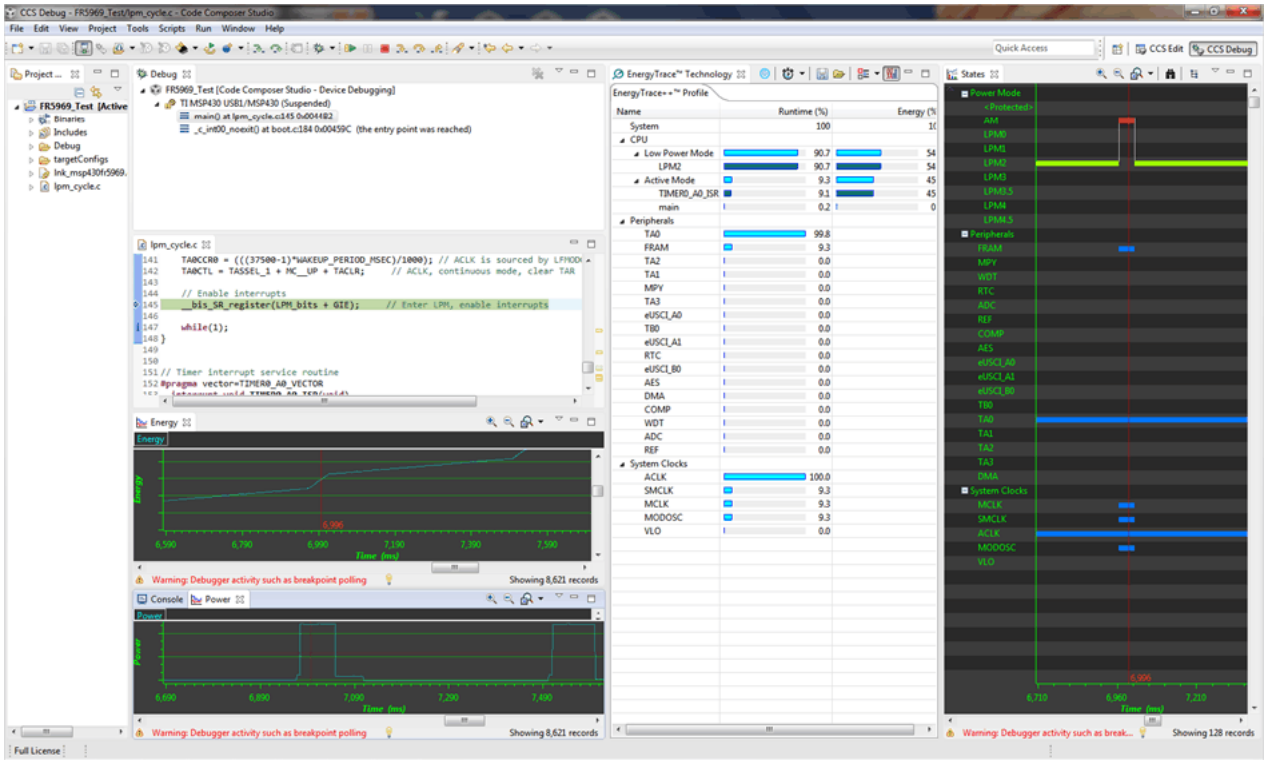

Figure 9. Debug Session With EnergyTrace++ Windows

After the EnergyTrace window is active, it can be controlled with the buttons in Table 2.

Table 2. EnergyTrace™ Technology Control Bar Icons

	Enable or disable EnergyTrace Technology. When disabled, icon turns gray.
	Set capture period: 5 sec, 10 sec, 30 sec, 1 min, or 5 min. Data collection stops after time has elapsed. However, the program continues to execute until the Pause button in the debug control window is clicked.
	Save profile to project directory. When saving an EnergyTrace++ profile, the default filename will start with "MSP430_D" followed by a timestamp. When saving an EnergyTrace profile, the default filename will start with "MSP430" followed by a timestamp.
	Load previously saved profile for comparison.
	Restore graphs or open Preferences window.
	Switch between **EnergyTrace++** mode and **EnergyTrace** mode

An example application using the MSP-EXP430FR5969 with EnergyTrace++ Technology is provided in the application note: *MSP430 Advanced Power Optimizations: ULP Advisor SW and EnergyTrace Technology* (SLAA603). For more details on EnergyTrace, refer to the *Code Composer Studio v6.0 for MSP430 User's Guide* (SLAU157).

2.3.4 Emulator Connection – Isolation Jumper Block

The isolation jumper block at Jumper J13 allows the user to connect/disconnect signals that cross from the eZ-FET domain into the FR5969 target domain. This includes eZ-FET Spy-Bi-Wire signals, application UART signals, and 3V3 and 5V power (see Table 3).

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 eZ-FET and target domains
- To expose the target MCU pins for other use than onboard debugging and application UART communication
- To expose programming and UART interface of the eZ-FET so it can be used for devices other than the onboard MCU.

Table 3. Isolation Block Connections

Jumper	Description
GND	Ground
V+	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 or not it is ready to receive data. The arrows indicate the direction of the signal.
RXD <<	Backchannel UART: the target FR5969 receives data through this signal. The arrows indicate the direction of the signal.
TXD >>	Backchannel UART: the target FR5969 sends data through this signal. The arrows indicate the direction of the signal.
RST	Spy-Bi-Wire emulation: SBWTDIO data signal. This pin also functions as the RST signal (active low)
TST	Spy-Bi-Wire emulation: SBWTCK clock signal. This pin also functions as the TST signal

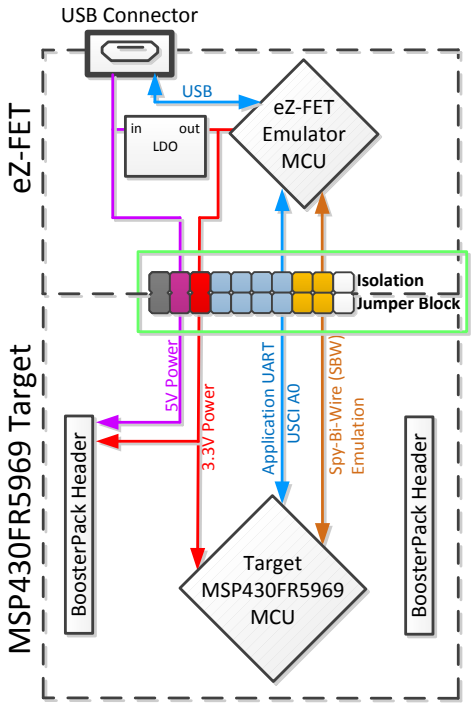

Figure 10. eZ-FET Isolation Jumper Block Diagram

2.3.5 14-Pin JTAG Connector

This EVM contains a footprint for TI's standard 14-pin MSP430 JTAG header. This connector can be used as needed. For debug purposes, this connector offers 4-wire JTAG compared to the 2-wire Spy-Bi-Wire from the eZ-FET. In certain use cases this can be advantageous. The MSP-FET or another MSP430 external debug tool such as MSP-FET430UIF or third-party tool can be used. This JTAG connector can be used to power the system directly or can be used with external power. The MSP-FET tool supports EnergyTrace™ technology and can perform the same measurements as the eZ-FET onboard emulator. See Section 2.4 for more details on the JTAG system power requirements.

2.3.6 Using the eZ-FET Emulator With a Different Target

The eZ-FET emulator on the FR5969 LaunchPad can interface to most MSP430 derivative devices, not just the on-board FR5969 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 Spy-Bi-Wire. 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's side of the isolation jumper block to the target hardware:

- 3.3 V (V+)
- GND
- SBWTDIO
- SBWTCK
- TXD (if the UART backchannel is to be used)
- RXD (if the UART backchannel is to be used)
- CTS (if hardware flow control is to be used)
- RTS (if hardware flow control is to be 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.

2.4 Power

The board is designed to support five different power scenarios. The board can be powered by the eZ-FET or JTAG debugger, external power, BoosterPack power, or standalone super cap power.

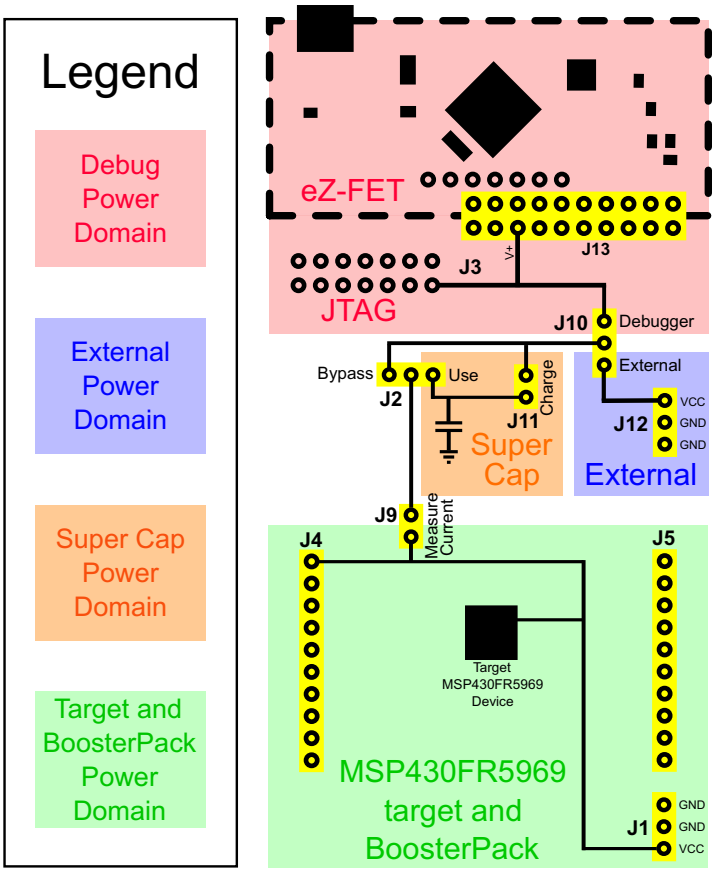

Figure 11. MSP430FR5969 LaunchPad Power Domain Block Diagram

2.4.1 eZ-FET USB Power

The most common scenario is power from USB through the eZ-FET debugger. This provides 5-V power from 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 J13. For 3.3 V, ensure that a jumper is connected across the J13 "V+" terminal. The eZ-FET is a debugger, so J10 must be set to debugger for power to reach the target MSP430FR5969 device.

For the power configuration diagram, see Figure 12.

2.4.2 14-Pin JTAG

When powering directly from the JTAG connector through the MSP-FET430UIF or other MSP430 debugger tool, ensure that jumper J10 is set to "Debugger." JTAG debugging can also be used with an external power source, when J10 is set to "External," and some external power source is connected through J12. In this case the JTAG debugger will sense the external power and debug the system without providing its own power.

For power configuration diagram, see Figure 12.

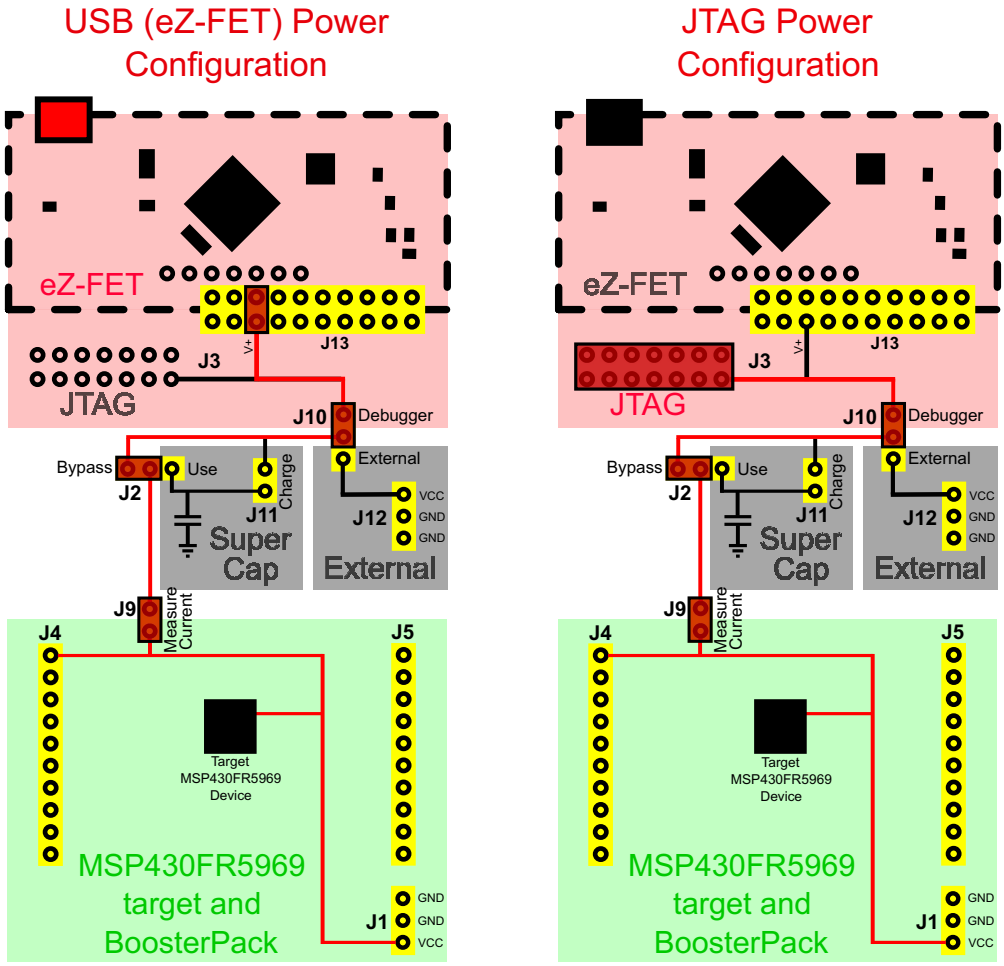

Figure 12. Debugger Power Configuration – USB eZ-FET and JTAG

2.4.3 External Power Supply

An extra header J12 is present on the board to supply external power. When supplying external power, jumper J10 must be set to "External." It is important to understand the device voltage operation specifications when supplying external power. The MSP430FR5969 has an operating range of 1.8 V to 3.6 V. More information can be found in the device data sheet.

For power configuration diagram, see Figure 13.

2.4.4 BoosterPack

In some use cases it might be required to power the board from a BoosterPack. When powered from a BoosterPack, the BoosterPack voltage should be across J4 Pin 1 (Vcc) and J5 Pin 20 (GND). This complies with the BoosterPack pinout shown in Section 2.5. These pins are connected directly to the FR5969 target device, and do not require any specific jumper configuration. Header J1 also provides power directly to the target device.

Because J1 and the BoosterPack headers are connected directly to the target device V_{cc}, there are two primary consequences:

- The super cap cannot charge through J11. Use of the super cap with this power scenario is not recommended.
- Current of the target device through J9 cannot be measured. It is best to remove J9 in this scenario to prevent back-powering of any additional circuitry such as the eZ-FET.

For power configuration diagram, see Figure 13.

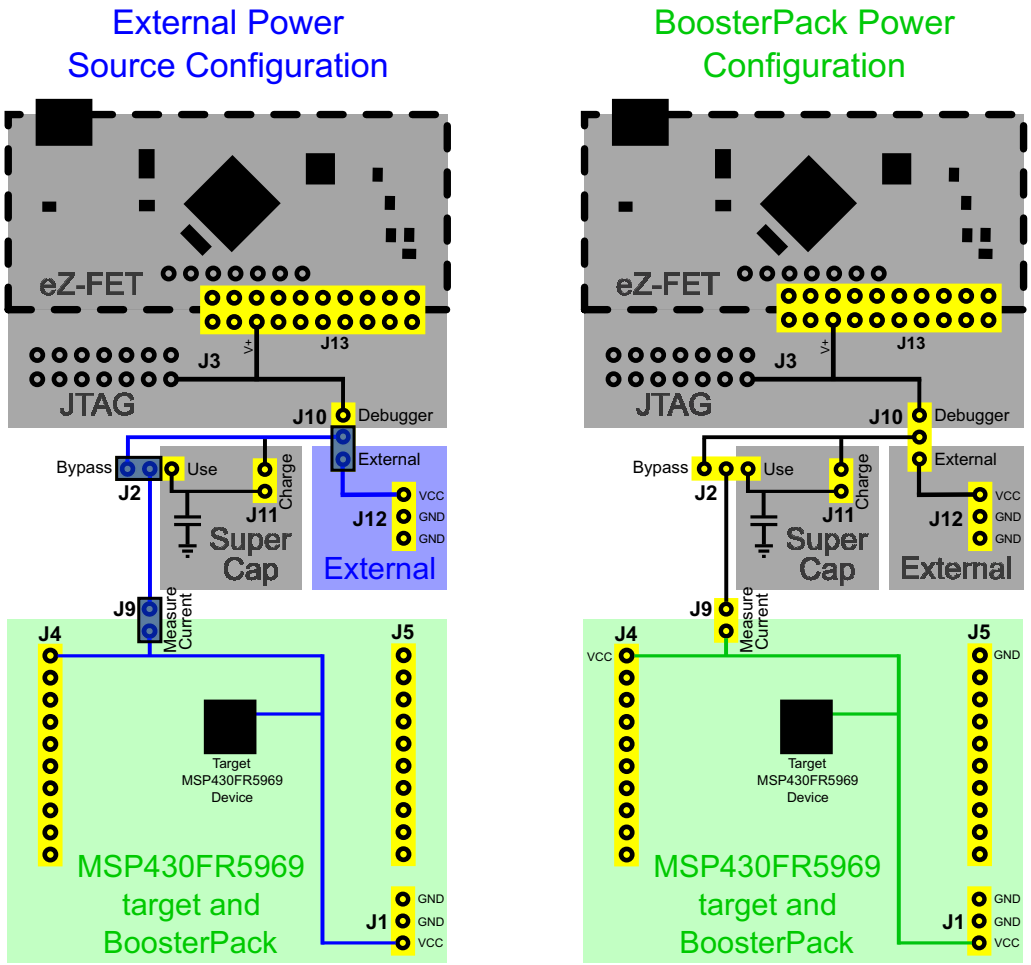

Figure 13. External Power Configuration – External and BoosterPack

2.4.5 Super Cap

A 100-mF (0.1-F) super cap is mounted onboard and allows powering the system without any external power. This highlights the ultra-low power of the MSP430FR5969 target device. See how long you can run your application on the super cap alone!

2.4.5.1 Charging the Super Cap

To charge the super cap, jumper J11 is used. By default there is no jumper across J11. Place a jumper across J11 to charge the super cap. If another jumper is not handy, the GND jumper on the isolation jumper block can be used- as this jumper doesn't actually disconnect the GND connection.

To charge the super cap, power must be coming from a debugger (eZ-FET or JTAG) or external power through J10. External power through J1 or a BoosterPack will not charge the super cap through J11.

Placing a jumper across J11 will charge the super cap when there is 3.3-V V_{cc} present, regardless of the state of the Bypass/Use J2 jumper, however if J2 is in the "Bypass" state, changing it over to the "Use" state will remove power from the target MSP430FR5969 and it will be reset.

2.4.5.2 Using the Super Cap

To use the super cap to power the LaunchPad, first change the J2 jumper to select "Use" and then set a jumper on J11 to charge the super cap. After waiting for it to charge, any external power can be removed from the system, and it will be powered completely by the super cap.

To remove any additional power drain from the super cap, remove any jumper to disconnect power to any external source. This can be J11, J10, or J13 depending on the power configuration. This prevents the super cap from back-powering the debug circuitry or any external power circuitry connected.

The most effective method for charging the capacitor is outlined in the following steps. These steps assume the LaunchPad is powered by USB cable through the eZ-FET debugger.

1. Set "Power Selector" jumper (J10) to "Debugger" position.
2. Set jumper J2 to "Use" super cap position.
3. Set jumper J11 to "Charge" super cap position.
4. Set "V+" jumper J13.
5. Connect board to PC with USB cable.
6. Allow two to three minutes for the super cap to charge (time may vary depending on initial charge of the super cap) to full V_{CC}.
7. Remove the "V+" jumper J13.

For power configuration diagram, see Figure 14.

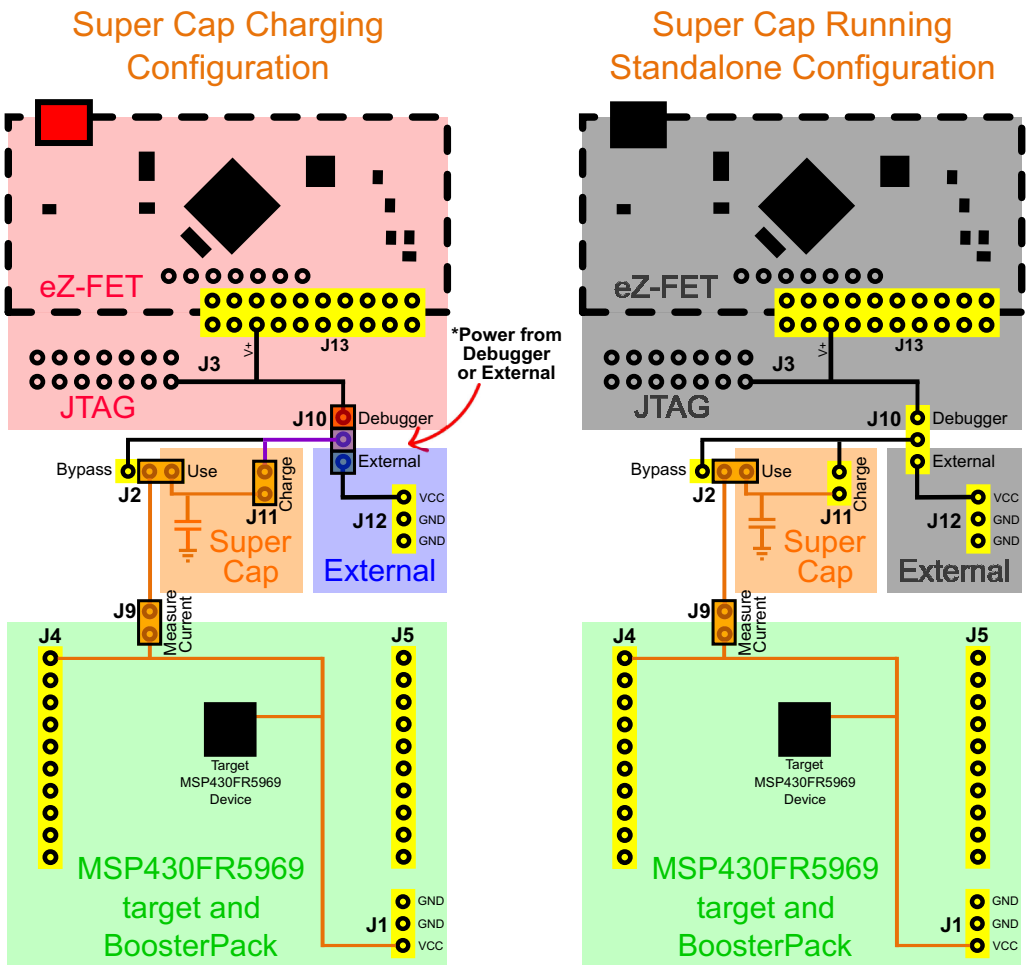

Figure 14. Super Cap Power Configuration – Charging and Running Standalone

2.4.5.3 Disabling the Super Cap

To disable the super cap, change J2 to "Bypass," and remove jumper J11 to prevent additional current for charging the super cap. With these two jumper selections, the super cap is completely disconnected from the system.

2.5 BoosterPack Plug-in Module Headers

The BoosterPack headers allow for a variety of applications to be created by plugging BoosterPacks onto the LaunchPad. BoosterPacks cover a wide range of possible applications including wireless connectivity, environmental sensing, and LCD displays. Leverage the growing BoosterPack ecosystem to rapidly prototype nearly any application with the LaunchPad rapid prototyping platform. Available BoosterPacks can be found at www.ti.com/boosterpacks.

The FR5969 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 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 FR5969 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 FR5969 LaunchPad, you should 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 FR5969 device pin function configuration in software. More information about compatibility can be found at http://www.ti.com/launchpad.

Figure 15 shows the 20-pin pinout of the FR5969 LaunchPad.

Note that software's configuration of the pin functions plays a role in compatibility. The FR5969 LaunchPad side of the dashed line shows all of the functions for which the FR5969 device's pins can be configured. This can also be seen in the MSP430FR5969 data sheet. The BoosterPack side of the dashed line shows the standard. The FR5969 function whose color matches the BoosterPack function shows the specific software-configurable function by which the FR5969 LaunchPad adheres to the standard.

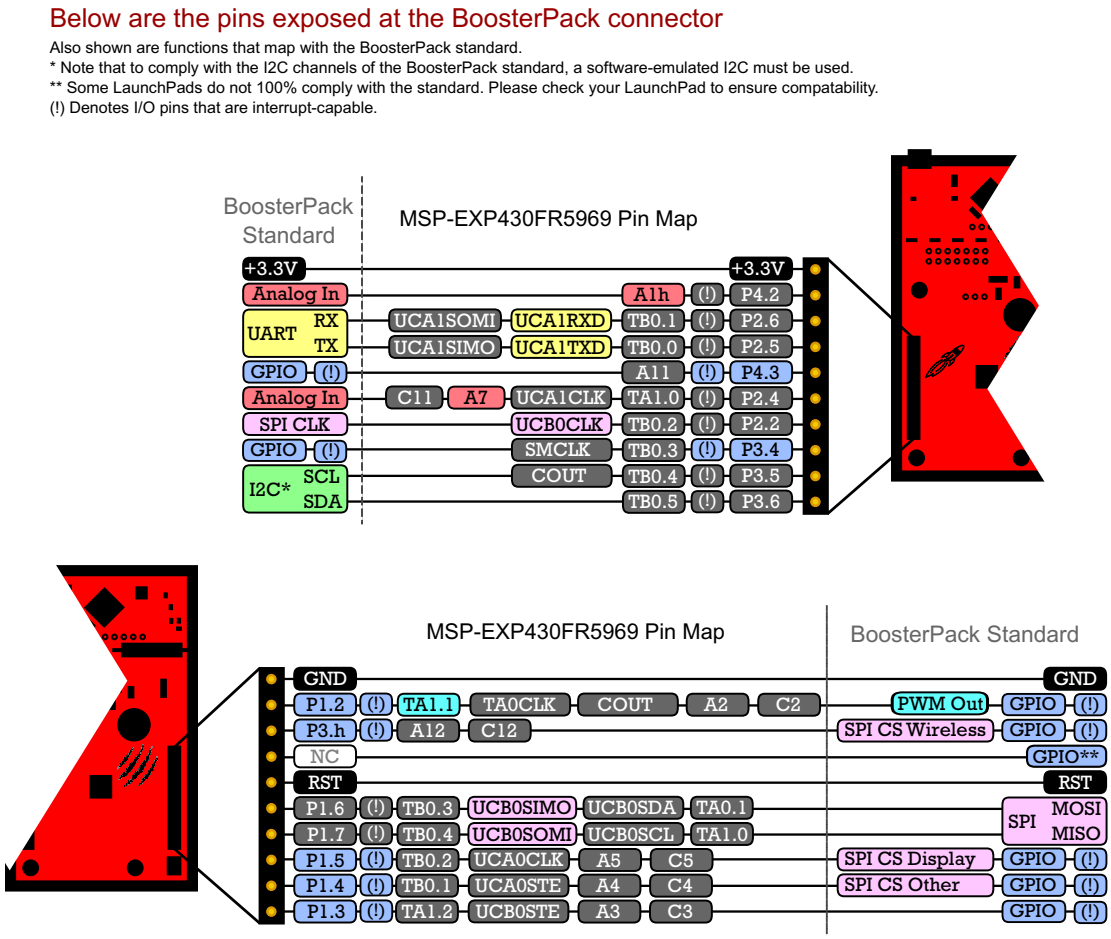

Figure 15. FR5969 LaunchPad to BoosterPack Connector Pinout

2.6 Design Files

A complete schematic is available in Section 6. All hardware design files including schematics, layout, bill of materials (BOM), and Gerber files are available in a zip folder (SLAR091) from ti.com. The software examples are made available in a different zip folder (SLAC645). For more information on the software examples, see Section 3.

The MSP-EXP430FR5969 LaunchPad was designed in Mentor Graphics PADS schematic and layout. A free viewer is available to see both the schematic and layout files on the Mentor Graphics website at http://www.mentor.com/pcb/downloads/pads-pcb-viewer. A full time-limited version of PADS is available online for free. This version has complete functionality until the 30 day license expires. This version can be downloaded directly from http://www.mentor.com/pcb/product-eval/pads-download-evaluation.

2.7 Hardware Change log

Table 4 shows the hardware revision history.

Table 4. Hardware Change Log

PCB Revision	Description
Rev 1.6	Initial Release
Rev 2.0	Added EnergyTrace functionality, extended PCB dimensions, added mounting holes, updated isolation block order, added 5V BP pin, updated silkscreen

3 Software Examples

There are three software examples included with the FR5969 LaunchPad, which can be found in the zip source folder (SLAC645). Table 5 describes these examples.

Table 5. Software Examples

Demo Name	BoosterPack Required	Description	More Details
OutOfBox_FR5969		The out-of-box demo pre-programmed on the LaunchPad from the factory. Its function was described in Section 1.4. Demonstrates features of MSP430FR5969 ULP FRAM device.	Section 3.3
430BOOST-SHARP96_ULP_FRAM	430BOOST-SHARP96	Demonstrates features of MSP430FR5969 ULP FRAM device with various application modes.	Section 3.4
430BOOST-SHARP96_GrlibDisplay	430BOOST-SHARP96	A very simple example showing how to use MSP430 Graphics Library (grlib) to display graphics primitives and images.	Section 3.5

3.1 MSP430 Software: Driver Library, Graphics Library, and Capacitive Touch Library

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

- Driver library (driverlib): a foundational MSP430 software library, useful for interfacing with all MSP430 core functions and peripherals, especially clocks and power.
- Graphics library (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: 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 the information you need is in MSP430Ware or specific library documentation linked above.

3.2 Development Environment Requirements

To use any of the below software examples with the MSP430FR5969 LaunchPad, you must have an integrated development environment (IDE) that supports the MSP430FR5969 device.

Table 6. IDE Minimum Requirements for MSP430FR5969

Code Composer Studio™ IDE	IAR Embedded Workbench™ IDE
CCS v5.5 or later	IAR EW430 v5.60 or later

For more details on where to download the latest IDE, see Section 4.3.

3.2.1 Pre-Compiled Binary

The /Binary/ folder inside the software zip package includes pre-compiled TI-TXT binary files for each of the above projects 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.

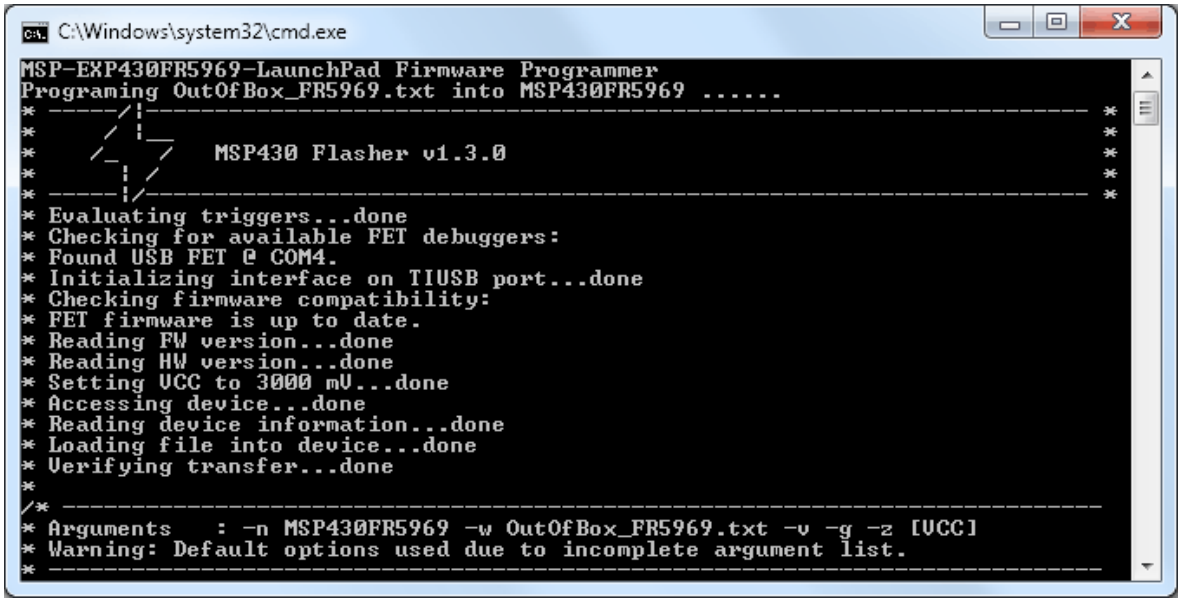

Figure 16. Program <Example>.bat

If desired, the "Program <Example>.bat" file can be modified to point to your own project's 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 will not be updated automatically. You must copy your 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.

3.2.2 CCS

CCS v5.5 or higher 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's project directory containing main.c. This is one of the OutOfBox_FR5969, 430BOOST-SHARP96_ULP_FRAM, or 430BOOST-SHARP96_GrlibDisplay projects (see Figure 17).

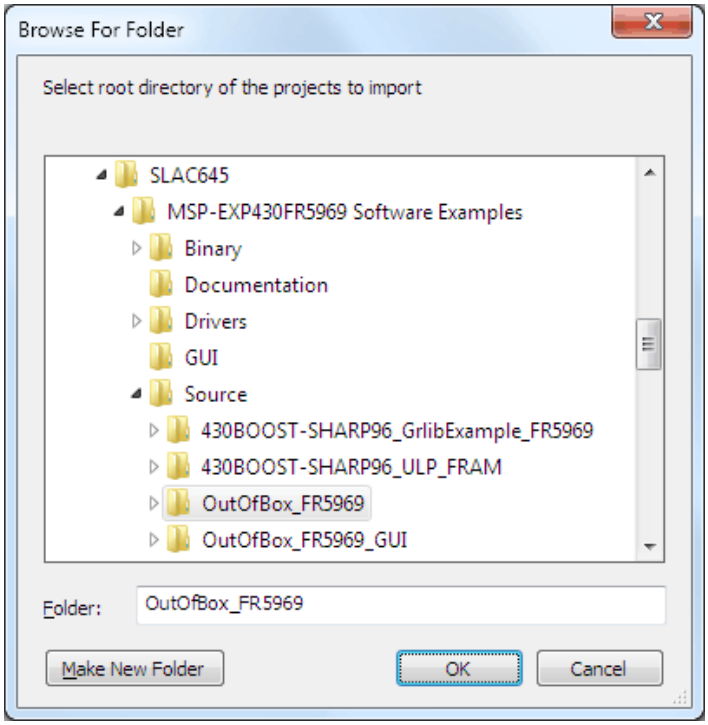

Figure 17. Directing the Project→Import Function to the Demo Project

Selecting the \CCS or \CCS_Code_Size_Limited folder also works. The CCS-specific files are located there.

When you click "OK", CCS should recognize the project and allow you to import it. The indication that CCS has found it is that the project appears in the box shown in Figure 18, and it has a checkmark to the left of it.

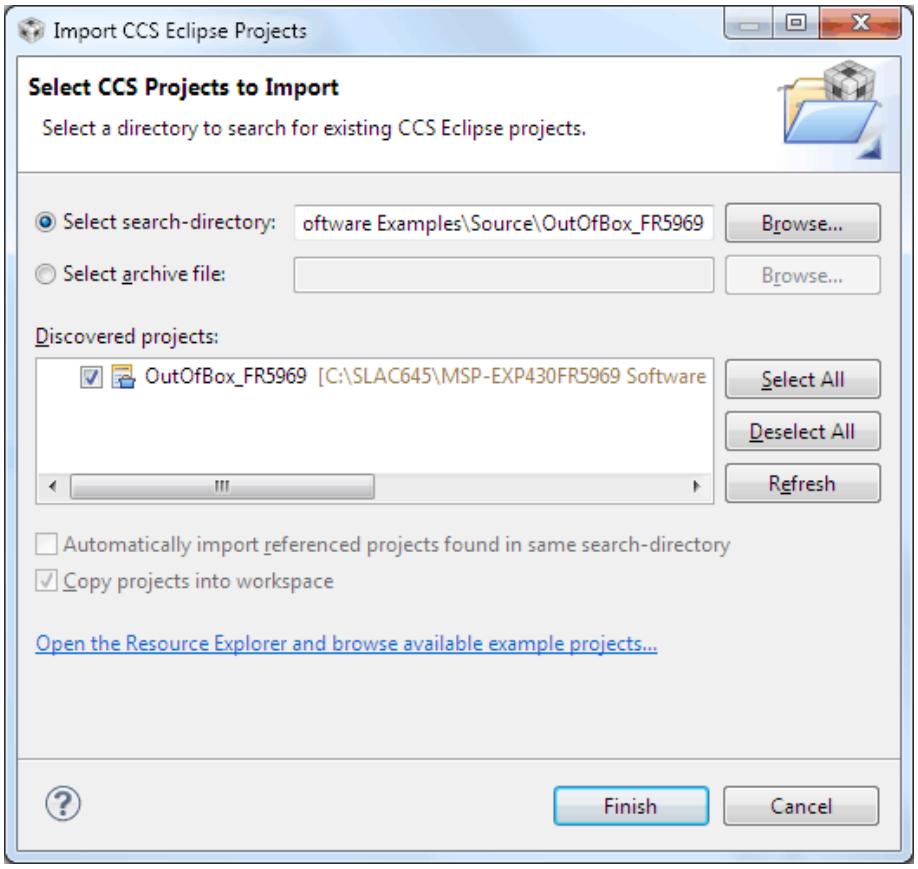

Figure 18. When CCS Has Found the Project

Sometimes CCS finds it but does not have a checkmark; this might mean that a project by that name is already in the workspace. Rename or delete that project to resolve this conflict. (Even if you don't see it in the CCS workspace, be sure to check the workspace's directory on the file system.)

3.2.3 IAR

IAR Embedded Workbench™ IDE v5.60 or higher is required. To open the demo in IAR, simply choose File→Open→Workspace…, and direct it 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, using Project→Add-Existing-Project….

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

3.3 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 full out-of-box demo cannot be built with the free version of CCS or IAR (IAR Kickstart) due to the code size limit. To bypass this limitation, a code-size-limited CCS version is provided, that has most functionality integrated into a library. The code that is built into the library is able to be viewed by the user, but it cannot be edited. For full functionality download the full version of either CCS or IAR.

There are two modes in the out-of-box software, which can only be interacted with using the provided GUI in the MSP-EXP430FR5969 Software Examples.

3.3.1 Source File Structure

The project is organized in multiple files. This makes it easier to navigate and reuse parts of it for other projects. Table 7 describes each file in the project.

Table 7. Source Files and Folders

Name	Description
main.c	The out-of-box demo main function, initializations, shared ISR's, etc
LiveTempMode.c	Main function file for live temperature streaming mode
FRAMLogMode.c	Main function file for FRAM data logging mode
Library: Driverlib	Device driver library (MSP430DRIVERLIB)

3.3.2 Power Up and Idle

When the out-of-box demo powers up, the red and green LEDs toggle several times. The MSP430FR5969 then enters low-power mode 3 to wait for UART commands from the PC GUI.

The GUI that is included in the MSP-EXP430FR5969 Software Examples (SLAC645) download is required to connect to the serial port that the LaunchPad's UART communication uses. Follow the "Setup" instructions in the GUI to establish the connection. After connection has been established, the GUI pings the LaunchPad every few seconds to make sure that it is still present. If no response is received from the LaunchPad, the GUI automatically closes the serial port connection.

3.3.3 Live Temperature Mode

To enter the live temperature mode, click the "Start" button below "Live Temp Mode" in the GUI's Application Controls panel (see Figure 19).

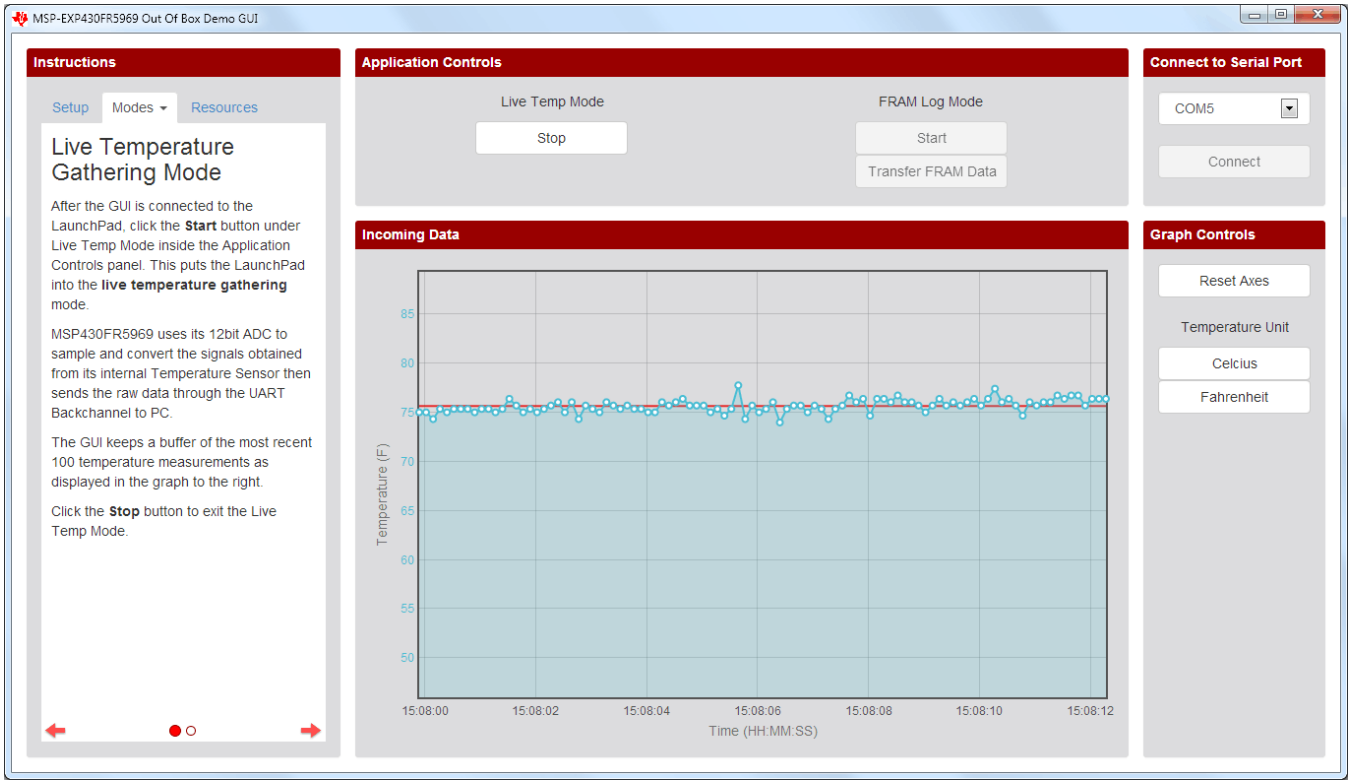

Figure 19. Live Temperature Mode

The MSP430FR5969 first sends two calibration constants for the temperature sensor to the PC GUI. It then sets up its 12-bit ADC for sampling and converting the signals from its internal temperature sensor. A hardware timer is also configured to trigger the ADC conversion every 0.125 seconds before the device enters low-power mode 3 to conserve power. As soon as the ADC sample and conversion is complete, the raw ADC data is sent the through the UART backchannel to the PC GUI.

As the raw ADC data is received by the PC GUI, Celsius and Fahrenheit units are calculated first. The PC GUI keeps a buffer of the most recent 100 temperature measurements, which are graphed against the PC's current time on the Incoming Data panel.

A red horizontal line is drawn across the data plot to indicate the moving average of the incoming data.

To exit this mode, click the "Stop" button under "Live Temp Mode". You must exit this mode before starting the FRAM Log Mode.

3.3.4 FRAM Log Mode

To enter the FRAM Log Mode, click the "Start" button under "FRAM Log Mode" in the GUI's Application Controls panel. The PC GUI immediately sends the current system timestamp to be stored in the LaunchPad. This timestamp is later used to extrapolate the X-axis time values when the FRAM logged data are transferred to the GUI (see Figure 20).

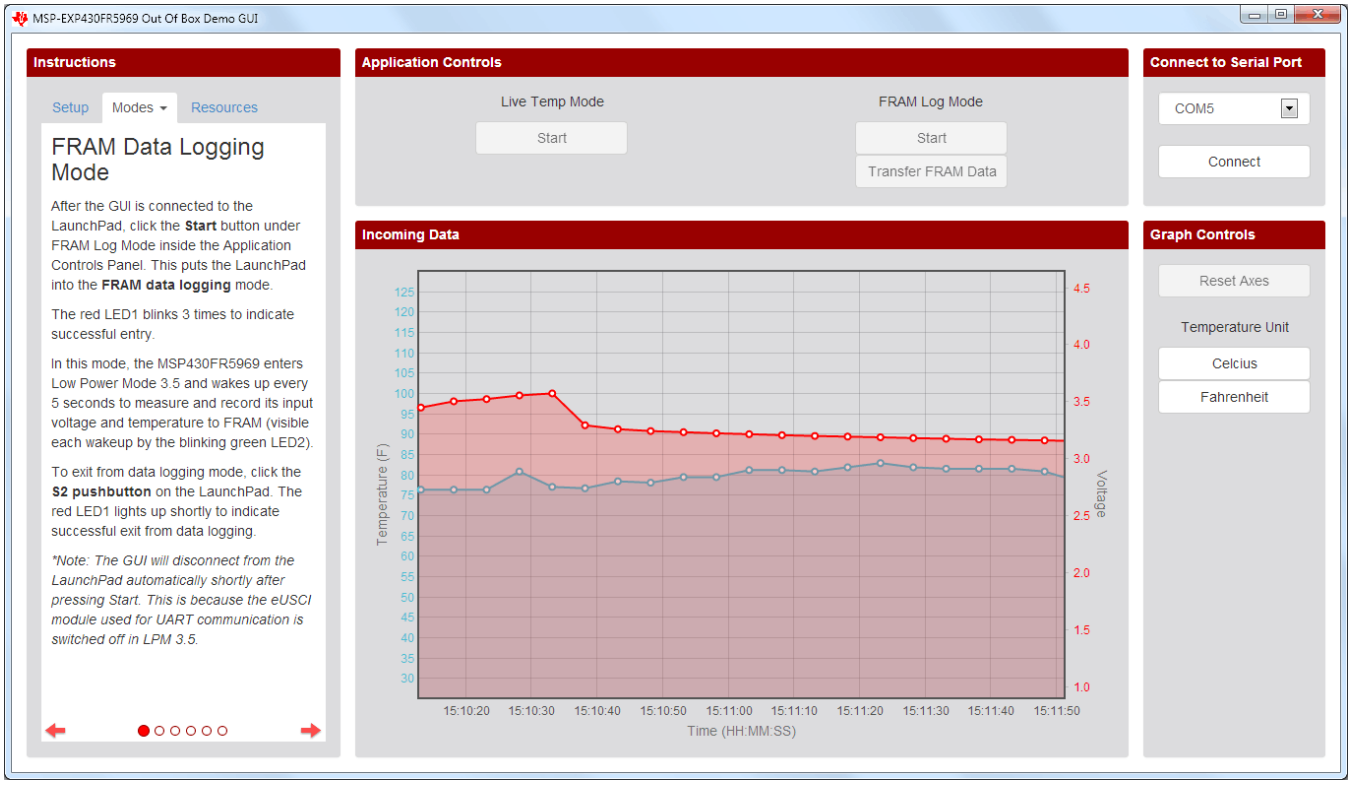

Figure 20. FRAM Log Mode

When the MSP430FR5969 receives the UART command from the GUI, it starts the entry sequence by initializing the Real-Time Clock to trigger an interrupt every 5 seconds. The red LED blinks three times to indicate successful entry into FRAM Log Mode.

Unlike in the Live Temperature Mode, the MSP430FR5969 enters low-power mode 3.5 to further decrease power consumption and wakes up every 5 seconds to perform data logging. Because the UART communication module does not retain power in LPM3.5, the GUI automatically disconnects from the LaunchPad after entry into FRAM Log Mode.

Each time the device wakes up, the green LED lights up to indicate its state to the user. The 12-bit ADC is set up to sample and convert the signals from its internal temperature sensor and battery monitor (Super Cap voltage).

A section of the device's FRAM is allocated to store the raw ADC output data (address 0x9000 to 0xEFFF). This allows the demo to store up to 6144 temperature and voltage data points (5 seconds/sample is approximately 8.5 hours of data).

The FRAM Log Mode also provides the option to log temperature data while powered either through the USB cable or only by the on-board Super Cap. The PC GUI contains step-by-step instructions in its side panel for configuring the jumpers on the LaunchPad to power the device with the Super Cap. See Section 2.4.5 for more detail on the Super Cap.

To exit the FRAM Log Mode, press the S2 (right) push button on the LaunchPad. The red LED turns on briefly to indicate successful exit. The LaunchPad returns to the Power up and Idle state, and you can reconnect the LaunchPad with the GUI to transfer the logged data from FRAM to the PC. Click the "Transfer FRAM Data" button in the GUI to begin transfer. A progress bar shows progress until the transfer completes, and the temperature and voltage data are plotted in the Incoming Data panel.

3.4 430BOOST-SHARP96 ULP FRAM Demo

This section describes the functionality and structure of the 430BOOST-SHARP96 ULP FRAM demo that is included in the software examples zip download.

NOTE: The 430BOOST-SHARP96 ULP FRAM demo relies on the 430BOOST-SHARP96 BoosterPack and has a very limited use without it.

The full demo source code cannot be built with the free version of CCS or IAR (IAR KickStart) because of the code size limit. To bypass this limitation, a code-size-limited CCS version is provided that has most functionality integrated into a library. The code that is built into the library can be viewed by the user, but it cannot be edited. For full functionality, download the full version of either CCS or IAR.

There are five applications in the demo software. All of them are in one project and the different applications can be cycled through in the user interface.

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. Source Files and Folders

Name	Description
Main.c	The user experience demo main function, shared ISRs, and other functions
ActivePowerMeasure.c	Main function file for Active Mode Power app
ClockApp.c	Main function file for Clock app
FR59xx_EXP.c	File for handling system init, main menu, and button operations
FRAMSpeedApp.c	Main function file for FRAM Speed app
Game.c	Main function file for SliderBall video game app
SYS.c	Functions to enter and exit LPM3.5
myTimer.c	Contains all timer-based functions and interrupts
ULPMeter.c	Main function file for Battery Free Stopwatch app
Library: CTS	Capacitive Touch Software Library (CAPSENSELIBRARY)
Library: Driverlib	Device driver library (MSP430DRIVERLIB)
Library: grlib	Graphics library for the SHARP LCD (MSP430-GRLIB)
Folder: Preloaded images	Images for the LCD screen

3.4.2 Navigation and Main Menu

Upon powering up the demo, the title screens appear on the LCD, and are followed by the main selection menu. The main menu displays all the applications available in the demo. The application options in the menu are highlighted by using the left capacitive touch slider.

NOTE: Only the left capacitive touch slider is activated for navigation.

After an application is selected, the right button (S2) is used to enter the application. To change the application or exit, use the left button (S1) and then navigate the main menu to switch to a different application.

3.4.3 Clock Application

NOTE: This application relies on the operation of the 32.768-kHz clock crystal that is pre-populated on the LaunchPad.

To enter this application, highlight the "Clock" option on the main menu and then push the right button (S2).

Immediately upon entering the Clock app, the user is expected to set the date and time details before the clock starts running. This needs to be done every time the application is entered, because the clock values are not maintained when running any of the other applications. To set the time and data parameters use the following steps:

1. When the app starts, the parameter being modified begins to blink.
2. The left capacitive touch slider can be used to increment or decrement the blinking parameter by placing a finger on the top or bottom portion, respectively, of the slider.
3. The value of the blinking parameter can be locked by placing a finger in the middle of the left capacitive touch slider.
4. The parameter that is being modified can be changed with the right button (S2).
5. Repeat steps 2 to 4 until all parameters have been set, after which the clock resets the seconds and begins to track the time from the set time and date.

When the clock begins to run, note that an option to turn on or off the seconds display is provided using the left button (S1). This is useful when attempting to measure power. The device spends most of the time in standby (LPM3), waking up every one second to update the RTC values. However if the display is updated every second, the average power is much higher than just the LPM3 power due to time and energy required to modify the LCD through SPI. If the SecON option is turned off, the device continues to provide a one second wakeup to update the RTC values but the display is updated only once a minute to save power. In this configuration the device power will be similar to power in LPM3 (refer device data sheet for exact values).

When attempting to measure power using the Current jumper J9, ensure that the meter is in place before the board is powered up. Removing this jumper while the application is running results in a power cycle of the device (because the connection to V_{CC} is broken), and the clock parameters must be re-entered.

3.4.4 FRAM Speed Application

To enter the FRAM Speed app, the "FRAM Speed" option on the main menu must be highlighted and the right button (S2) then pushed. In this application, the FRAM write speed (in kilobytes per second), the total data written to FRAM (in kilobytes), and the FRAM endurance (in percentage) is tracked and displayed on the LCD. No user interaction is required.

The application uses Direct Memory Access (DMA) to transfer data to a 1KB block of FRAM. The starting address of this block is defined and can be modified within the FRAMSpeedApp.h file. Note that changing this location can cause an overlap with other application code. This is not advised, because code may be overwritten while running the application. Hence special care needs to be taken to evaluate the size of the code to ensure that it is not over-written while measuring the FRAM write speed.

It should be noted that this application is optimized for speed rather than power. The speed of this application is approximately 7500KB (7.5MB) per second. On a flash device, the achievable speed would be approximately 13KB per second.

Larger blocks of data can be written, which results in faster write speeds but also higher power consumption. For more information on how to optimize FRAM write speeds, refer to the application report *Maximizing FRAM Write Speed on the MSP430FR57xx* (SLAA498).

In this application, the system main clock is configured to use the DCO at 8 MHz. The application configures the DMA transfer of data and continuously executes it while remaining in LPM0. Each time the DMA writes the 1KB block, a count variable is incremented and the next DMA transfer is triggered. A timer is set up to interrupt the FRAM writing every 0.25 second to calculate the speed, total the kilobytes of data written, update the endurance, and then output these parameters on the LCD.

Note that the FRAM endurance percentage is retained after a power cycle. To exit the application and stop the FRAM writes, the left button (S1) can be pushed allowing the user to return to the main menu.

3.4.4.1 Understanding the Numbers Behind the FRAM Speed Application

The LCD is updated every 250 ms with an updated percentage change in the FRAM endurance. To calculate the endurance, some approximations were made in order to provide a meaningful output on the LCD.

Every 250 ms, 1.8MB of FRAM are programmed with a pattern. Hence the speed of FRAM writes is calculated as 7.564 MB/s. The FRAM is written to in blocks of 1K bytes; it is this 1KB block that is subject to the lifetime FRAM endurance specification.

FRAM endurance of block = **E** = 10^{15} write cycles. This is a minimum specification for FRAM endurance found in the device data sheet.

Table 9. FRAM Endurance Calculation for 1KB Block of FRAM

Variable		Derived From	Value
E	FRAM endurance	Data sheet	10^{15} writes
W	Write speed	Application	7.564 MB/second
B	FRAM block size	Application	1KB (1024 bytes)
N	Number of writes to a unique byte/sec	N = W / B	7386 writes/second
$\mathbf{T_{LCD}}$	Time between LCD updates	Application	250 ms
$\mathbf{T_{LIFE}}$	Time until endurance specification is met	$T_{LIFE} = E / N$	1.35×10^{11} seconds (more than 4000 years)
L	Lifetime percentage reduction per LCD update	$L = (T_{LCD} / T_{LIFE}) \times 100$	1.85×10^{-10}%

The calculated value of **L** is rounded up to 2×10^{-10}%, or 0.0000000002%. This is the amount the FRAM endurance is decremented on the LCD every 250 ms.

Note that the FRAM endurance percentage is retained during on a power cycle. This parameter is preserved by storing it in FRAM and preventing the variable from being overwritten on a power cycle. Refer to the NO INIT and LOCATION pragmas in the CCS compiler documentation for more details. This parameter will be reset when the device is reprogrammed, and the address overwritten.

3.4.5 Battery Free Application

To enter the Battery Free Stopwatch application, select the "Battery Free" option on the main menu and then press the right button (S2).

This mode is intended to be used when running from the super cap only. See Section 2.4.5 for more information on how to power the LaunchPad from the super cap.

When the application is entered, a submenu appears showing two possible actions to be taken. The first action is to "Run App," which starts the Battery Free Stopwatch application and log . The other option is to "Transmit Data", which transmits all logged data from previous runs through the MSP430 UART to a PC.

The "Run App" selection has four modes:

1. Title Mode (Warning Page)
2. Deep Sleep -LPM3.5 Mode
3. Display Mode
4. Low Battery Indicator Mode

When in Title mode or Display mode, press the right button (S2) to put the device into LPM3.5 (Deep Sleep Mode). Also in these two modes, press the left button (S1) at any time to exit the app and return to the main menu.

When in Deep Sleep mode, the device remains in LPM3.5, and only the RTC is active. The left button (S1) is deactivated while in this mode, and the right button (S2) can be used to wake the device from LPM3.5 and send the device into Display mode. Also, in this mode the RTC wakes up the device periodically to allow the ADC to sample the supply voltage before returning to LPM3.5. These ADC samples of the supply voltage are logged into FRAM and can be transmitted back to a PC in the "Transmit Data" mode.

The Display mode shows the stopwatch display and also the charge consumed while in LPM3.5. The stopwatch is started when the device enters LPM3.5 and stopped on exit. Hence the total time spent in LPM3.5 is displayed in HH:MM:SS format. The charge indicator is a reflection of the most recent ADC sample of supply voltage. If the device is left inactive at the "Display Mode" screen for more than ten seconds, the app times out, and control reverts to the main menu.

Low Battery mode is entered conditionally following an ADC measurement of the supply voltage. When V_{CC} < 2.2 V, the device displays a "Low Battery" warning screen. The screen recommends that the device be plugged into the PC through USB for charging. In this mode, the left button (S1) is deactivated, and the right button (S2) is used to check if USB has been plugged in or not. If the device has not been plugged into USB and the right button (S2) is pushed, the device remains in Low Battery mode. If the device has been plugged into USB and the right button (S2) is pushed, the device enters Deep Sleep mode again.

When running this application, the ADC measurements are logged in FRAM while the device is running from the super cap indicating that the ADC sampling and FRAM write have a very low-power footprint. These logged values can then be sent to the PC and the data processed to analyze the reduction of charge over time. The transfer of data can be done in the UART transmit mode.

The basic operation of the UART transmit mode is outlined below.

1. The eUSCI-UART and DMA modules are set up to transfer the data from FRAM.
2. "Sending Data – Please Wait" screen is displayed while the operation is in progress.
3. On completion "Data Send Complete" screen is displayed.
4. The data can be viewed using any hyperterminal application on the PC.

3.4.6 Active Power Application

The active power of the MSP430FR5969 device is directly dependent on the code and data cache hit ratio and the clock speed of the CPU. The Active Power application shows the impact of both these factors on overall system power.

To measure the power consumption of the MSP430FR5969 for the different frequencies and cache hit ratios, the following steps should be followed:

1. Remove the "Measure Current" jumper from the LaunchPad
2. Use an ammeter set to the "mA" range and connect the leads of the ammeter to the nodes of the "Measure Current" jumper
3. Navigate the main menu to the "Active Mode" app
4. Choose a frequency and cache hit ratio from the subsequent menus
5. Press the right button (S2) to enter the cache hit code
6. Tune the ammeter range to obtain the most accurate current measurement values
7. Prior to exiting the cache hit code, ensure that ammeter is in "mA" range, then press right button (S2) to exit cache hit code

3.4.7 SliderBall Game

This application was designed to show the functioning ability of the two Capacitive Touch sliders in conjunction with the LCD from the 430BOOST-SHARP96 BoosterPack.

To enter the application, the SliderBall game option on the main menu must be highlighted and the right button (S2) then pushed. The SliderBall game requires the player to use a sliding paddle to keep the ball in play. The goal of the game is to keep the ball alive and on the screen by having it hit off of the two paddles at each end of the screen. Users start off with five lives to accumulate as many points as possible. For each time that the ball is blocked by the paddle, points are awarded. The higher the difficulty, the more points are awarded for each hit. Each time the ball reaches the end of the screen and the paddle has not hit the ball, the user loses a life. After the life is lost, the ball automatically starts again for another round. This repeats until all lives are exhausted, and the game is over. If the high score has been achieved, a congratulations screen will be displayed to notify the user. At this point the final score, as well as the board high score will be displayed and the user may then choose a new level of difficulty to play once again.

To navigate the user levels menu and choose an option, the left capacitive touch slider and right button are used similar to all previous menus. The user may choose between the following: easy, normal, hard, and Insane. After selecting a difficulty, the game will begin to start, with the ball moving to the right-hand side first. Both capacitive touch sliders are used to control their respective paddles along the side of the screen. When the user misses the ball, it will be held in place for a few cycles before starting to move again to give the user a chance to regroup following losing a life. To create "easier" versions of the game, sleep cycles are added to slow down the game play.

The high score for each user level is stored in FRAM and is retained on subsequent power cycles. This value is erased only when the device is re-programmed.

3.4.8 Special Notes: Inverting the Display Color Scheme

A feature that has been built in to the demo code is the ability to invert the display colors. This can be a useful feature for times when the original display color settings are difficult to read.

To invert the colors edit the file 'sharp96x96.h' within the 'grlib' directory. In the 'User Configuration for the LCD Driver' section under 'Invert Display Option' use either one of the # defines 'NORMAL_DISPLAY' or 'INVERT_DISPLAY' as needed.

When INVERT_DISPLAY is defined it allows the demo to display with a black background and white foreground once the demo code is re-downloaded onto the MSP-EXP430FR5969 board.

3.5 430BOOST-SHARP96 Graphics Library Demo

NOTE: This graphics library demo is dependent on the 430BOOST-SHARP96 BoosterPack that comes with the MSP-BNDL-FR5969LCD bundle.

The grlib demo shows how to use the MSP430 Graphics Library http://www.ti.com/tool/msp430-grlib or "grlib," in a project with the Sharp® display. This demo cycles screens without user interaction to show simple graphics primitives.

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

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

FRAM memory devices like the MSP430FR5969 are touted for ultra-low power, but in some applications the FRAM memory can provide additional benefits such as dynamic memory allocation. In applications with dot matrix LCD displays, it is often advantageous to keep a RAM buffer of the contents currently on the display. For a smaller display such as the Sharp display on the 430BOOST-SHARP96 BoosterPack, this doesn't require much RAM to keep the display contents.

$$RAM\ bytes\ required = \frac{96\ pixels/row}{8\ pixels/byte} \times 96\ rows = 1152\ bytes \quad (1)$$

But in displays with more pixels or color displays, these RAM buffers can quickly become very large. If the Sharp display was a color display with 16 bits or color per pixel, (common in color displays) this buffer would be significantly larger.

$$RAM\ bytes\ required = \frac{96\ pixels/row}{0.5\ pixels/byte} \times 96\ rows = 18432\ bytes \quad (2)$$

When selecting a microcontroller for an application with a display like this would require a very large memory device for a typical RAM/Flash microcontroller. Typical RAM memory cutoffs would likely require a 32KB RAM device with around 128KB or 256KB of Flash. This may be significantly more memory than the application requires.

FRAM's unified memory block can be dynamically partitioned into data or code memory, providing unmatched flexibility. Applications like this can be easily supported with a 32KB or 64KB FRAM device.

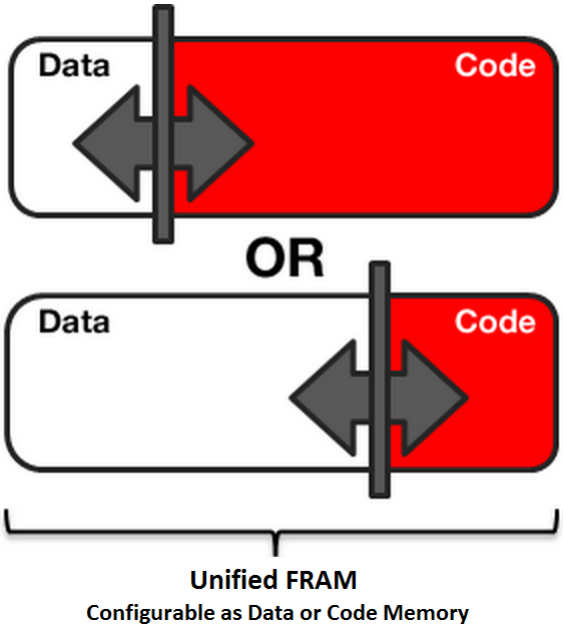

Figure 21. FRAM Unified Memory With Dynamic Partitioning

4 Additional Resources

4.1 LaunchPad Websites

More information about the FR5969 LaunchPad, supported BoosterPacks, and available resources can be found at:

- FR5969 LaunchPad tool page: resources specific to this particular LaunchPad
- TI's LaunchPad portal: information about all LaunchPads from TI for all MCUs

4.2 Information on the MSP430FR5969

At some point, you will probably want more information about the FR5969 device. For every MSP430 device, the documentation is organized as shown in Table 10.

Table 10. How MSP430 Device Documentation is Organized

Document	For FR5969	Description
Device family user's guide	*MSP430FR58xx, MSP430FR59xx, MSP430FR68xx, and MSP430FR69xx Family User's Guide* (SLAU367)	Architectural information about the device, including clocks, timers, ADC, and other peripherals.
Device-specific data sheet	*MSP430FR59xx, MSP430FR58xx Mixed Signal Microcontroller* data sheet (SLAS704)	Device-specific information and all parametric information for this device

4.3 Download CCS, IAR, or MSPGCC

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), IAR, or Energia.

CCS and IAR are each available in a full version, or a free, code-size-limited version. The full source code for some of the included example projects cannot be built with the free version of CCS or IAR (IAR KickStart) due to the code size limit. To bypass this limitation, a code-size-limited CCS version is provided, that has most functionality integrated into a library. The code that is built into the library is able to be viewed by the user, but it cannot be edited. For full functionality download the full version of either CCS or IAR.

See 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 a driver library (driverlib) and the graphics library (grlib) used in the software demo. By default, MSP430Ware is included in a CCS installation. IAR users must download it separately.

MSP430Ware includes the TI Resource Explorer, for easily browsing tools. For example, all the software examples are shown in the tree below.

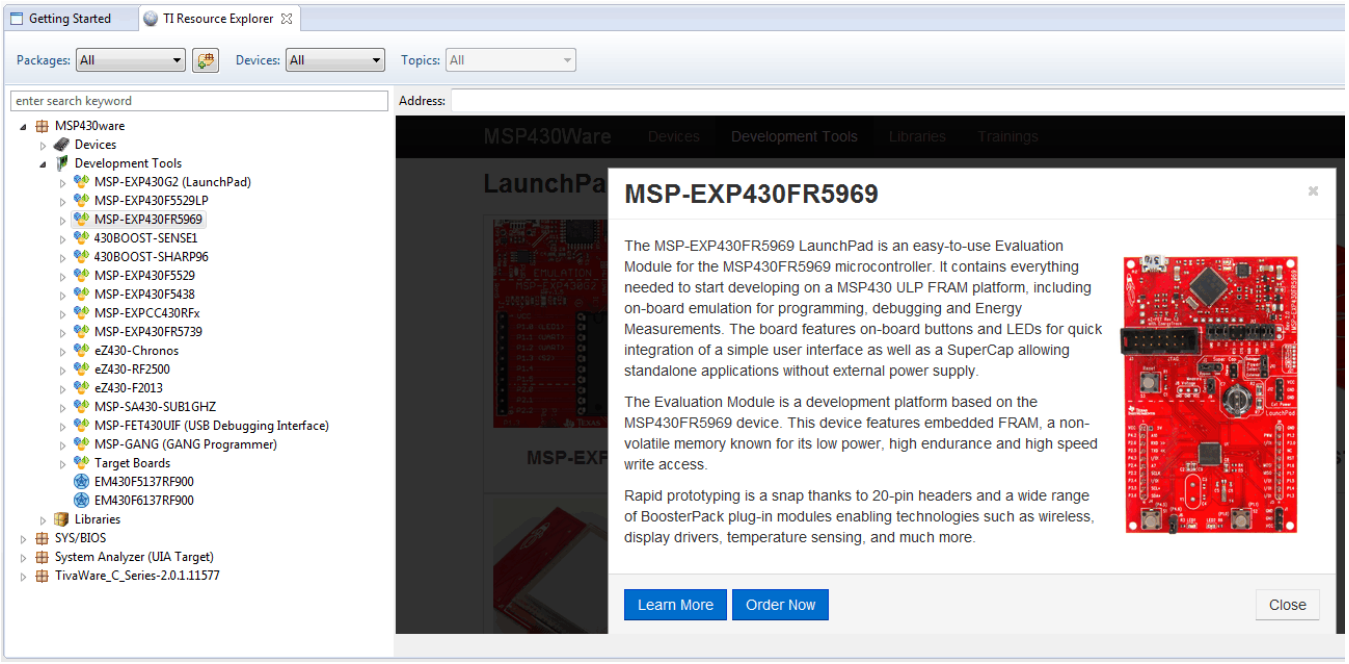

Figure 22. MSP-EXP430FR5969 Software Examples in TI Resource Explorer

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

4.5 MSP430FR5969 Code Examples

This is a set of very simple code examples that demonstrate how to use the MSP430's entire set of peripherals: ADC12, Timer_A, Timer_B, and so on. These do not use driverlib, 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.

There are also code examples available that use driver library. These code examples are part of the driverlib download included with MSP430Ware. To access these code examples, navigate into the driverlib folder or use the TI Resource Explorer to import into CCS.

4.6 MSP430 Application Notes

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

4.7 The Community

4.7.1 TI E2E Community

Search the forums at e2e.ti.com. If you cannot find an 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 FAQs

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_A0 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 on the USCI_A0 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.

6 Schematics

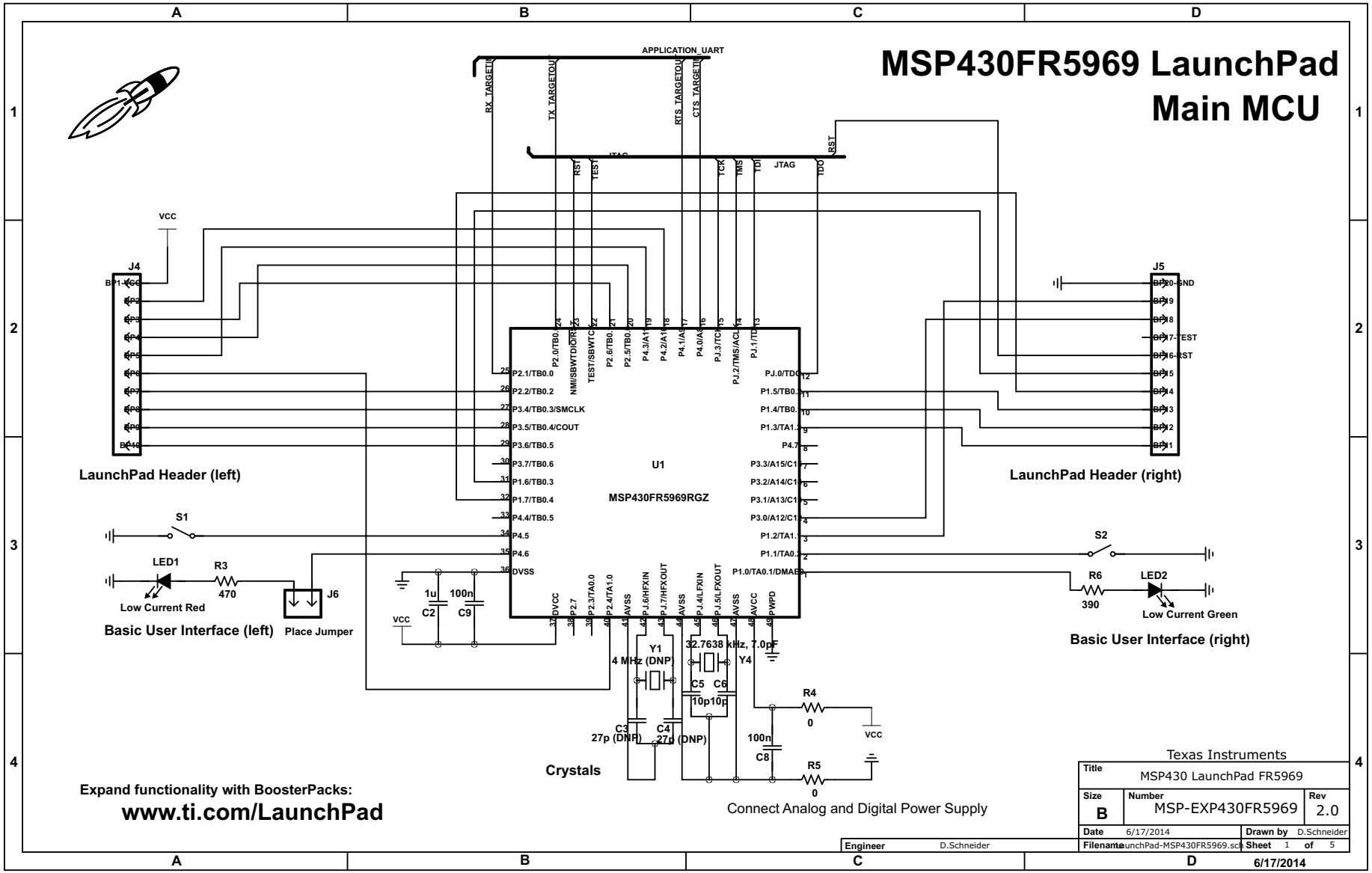

Figure 23. Schematic 1 of 5

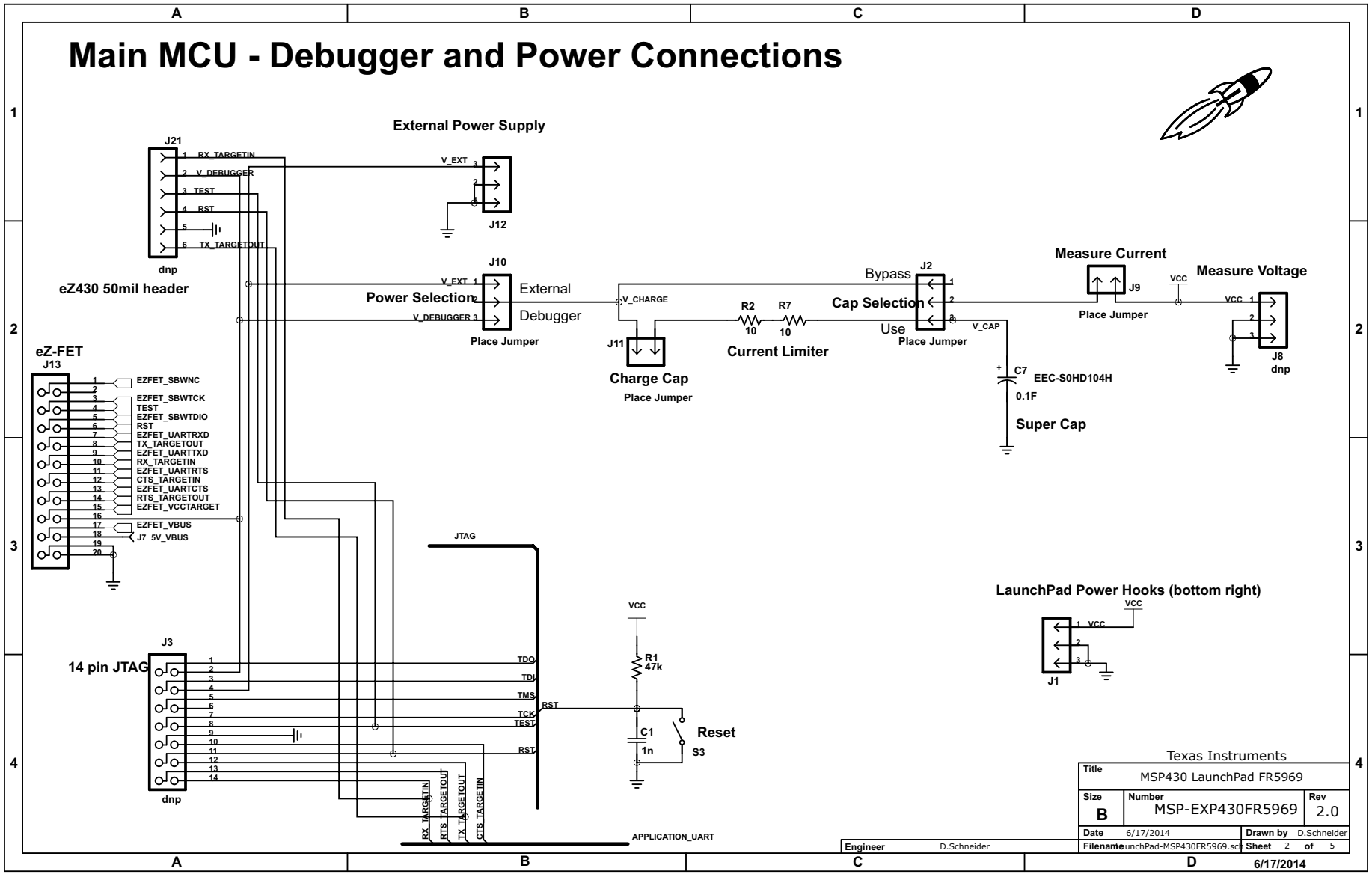

Figure 24. Schematic 2 of 5

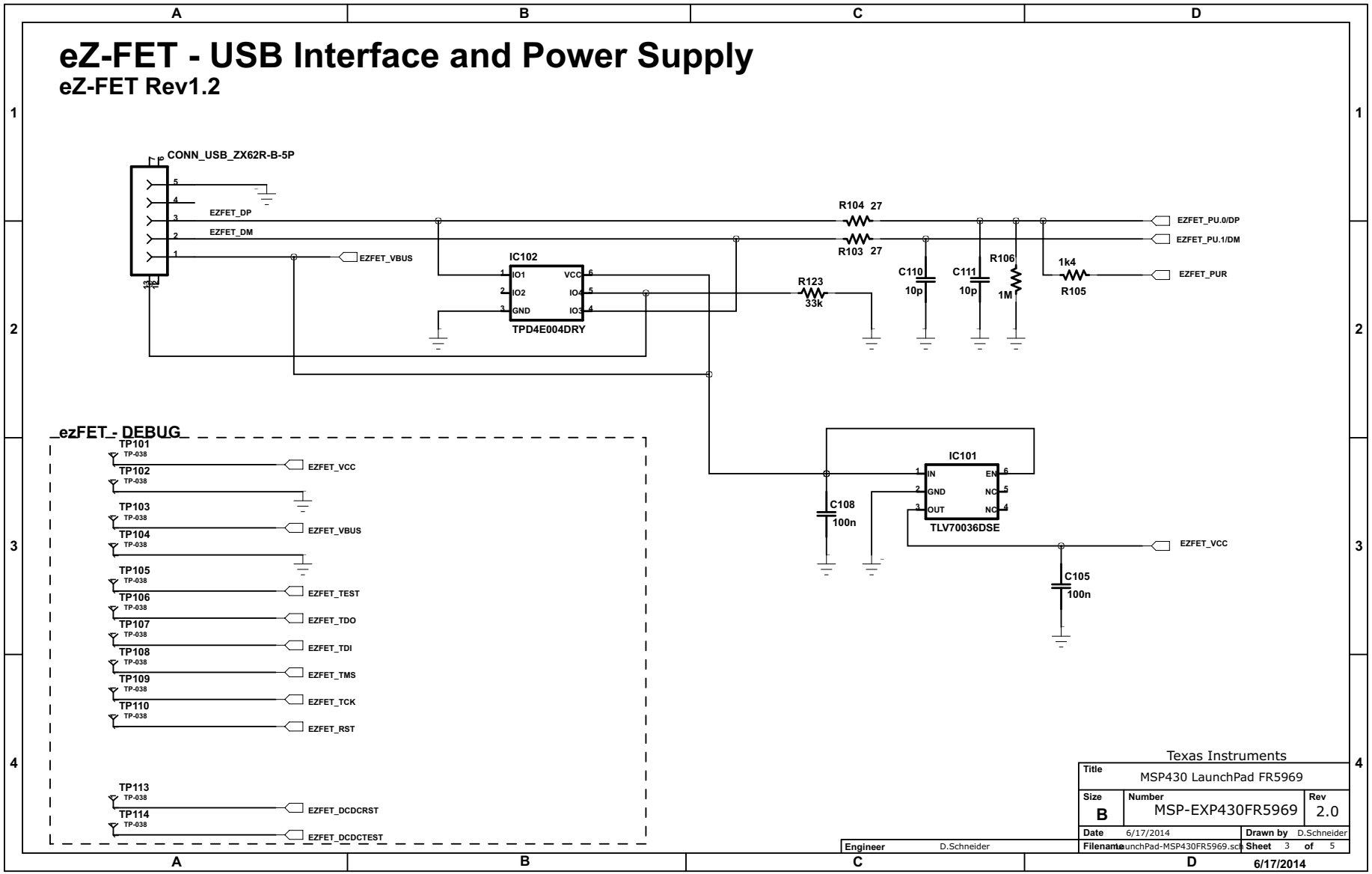

Figure 25. Schematic 3 of 5

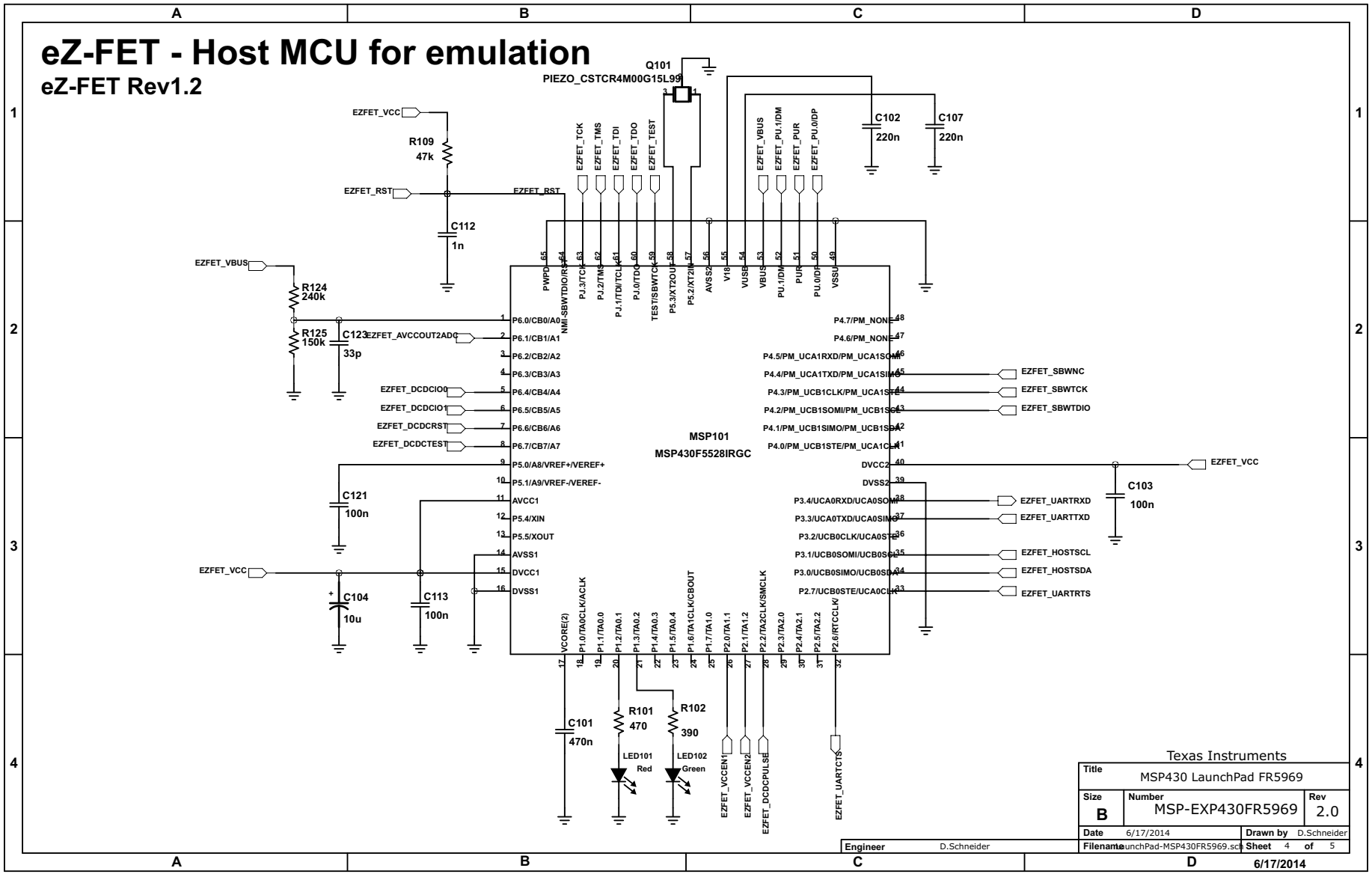

Figure 26. Schematic 4 of 5

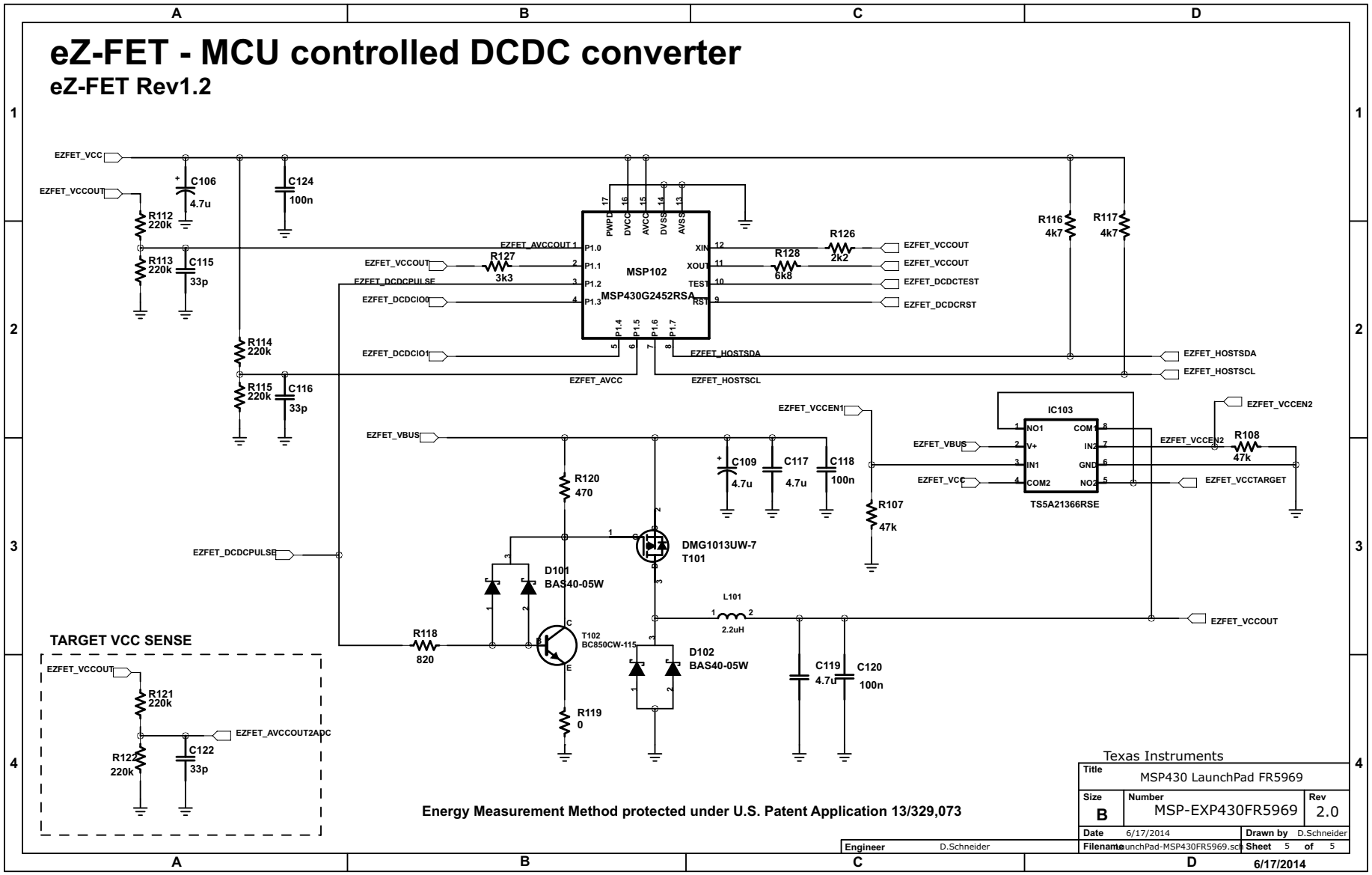

Figure 27. Schematic 5 of 5

Revision History

Changes from Original (February 2014) to A Revision **Page**

- Changed title of document 5
- Changed contents of Section 1.1 to improve description and add links 5
- Changed Figure 1 5
- Changed Figure 2 7
- Changed Figure 5 11
- Changed link to hardware design files (SLAR091) 11
- Added Section 2.3.1 11
- Added Section 2.3.3 13
- Changed description in Section 2.3.5 17
- Changed Figure 11 18
- Changed Figure 12 19
- Changed Figure 13 20
- Changed Figure 14 22
- Added first sentence in Section 2.5 22
- Changed Figure 15 23
- Changed contents of Section 2.6 to show latest links and description 23
- Added Rev 2.0 to Table 4 24
- Changed text and table in Section 3 24
- Added Section 3.2.1 25
- Changed contents of Section 3.2.2 26
- Deleted paragraph that started "Finally, click "Finish"..." 27
- Added Section 3.3 27
- Changed name of demo and its description in Section 3.4 30
- Changed Figure 21 35
- Changed all schematics in Section 6 39

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

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information 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.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

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