TI Designs

Code Generation and Optimization With FlowESI GUI and EnergyTrace™

TI Designs

TI Designs provide the foundation that you need including methodology, testing and design files to quickly evaluate and customize the system. TI Designs help *you* accelerate your time to market.

Design Resources

TIDM-FLOWESI-ETRACE	Design Folder
FlowESI GUI	Software Folder
MSP430 Flash Emulation Tool	Software Folder
EnergyTrace Technology	Software Folder
MSP430FR6989	Product Folder
EVM430-FR6989	Tool Folder

TI E2E™ Community

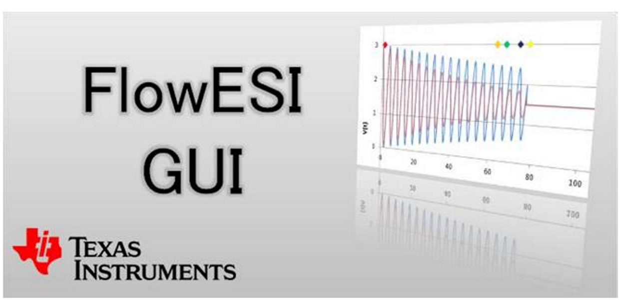
ASK Our E2E Experts
WEBENCH® Calculator Tools

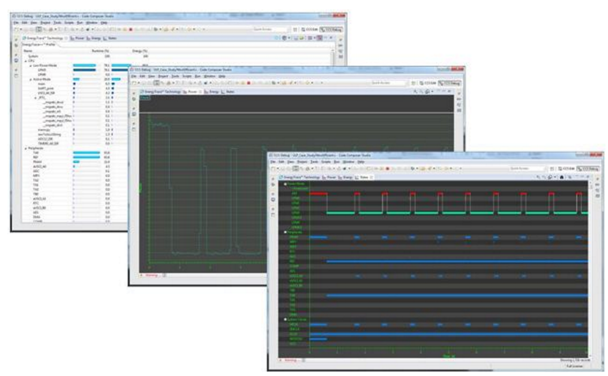
Design Features

- Showcase of FlowESI GUI
- Showcase of Energy Trace Technology
- Ultra-Low-Power Rotation Detection Using LC Sensors and ESI

Featured Applications

- Flow Meter
- Gas Meter
- Heat Meter
- Other Applications for Rotation Detection





An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

1 System Description

When designing battery-powered applications, ultra-low power consumption is the key factor in extending the lifetime of a system. Long-running designs must not waste the energy they are provided. Despite choosing appropriate low-power hardware components, firmware also takes an important role to reduce power consumption. Careful firmware design, such as minimizing MCU active time, maximizing MCU sleep time, and controlling external circuit to reduce leakage current, can reduce power consumption significantly. This task may not be easy, especially when the firmware design is complicated where multiple device modules are involved.

This TI design highlights the usage of FlowESI GUI and the EnergyTrace technology to help developers to design and optimize ultra-low-power applications on the EVM430-FR6989.

1.1 EVM430-FR6989

The EVM430-FR6989 (water meter reference design) kit is an easy-to-use evaluation module for the MSP430FR698x family of microcontrollers. The kit consists of three boards: the main board, the sensor board, and the motor board.

The main board of the EVM is built on MCU MSP430FR6989 with different user interfaces such as LCD, buttons and LEDs. The built-in eZ-FET enables direct programing to the MCU without extra FET tools. The eZ-FET also supports EnergyTrace technology for monitoring power consumption of the system. The MSP430FR6989 also supports EnergyTrace++™ to monitor the usage of different modules inside the MCU. A dedicated hardware provides high-speed communication between the target board and FlowESI GUI installed on the PC.

Designed for flow meter applications, the sensor board is a daughter board consisting of two LC sensors. The sensors are connected to the ESI module of the MSP430FR6989.

The motor board drives the rotor disc to simulate water or gas flow. The buttons control the rotating direction of the disc while the variable resistor controls the rotating speed.

Visit http://www.ti.com/tool/EVM430-FR6989 for a detailed description.

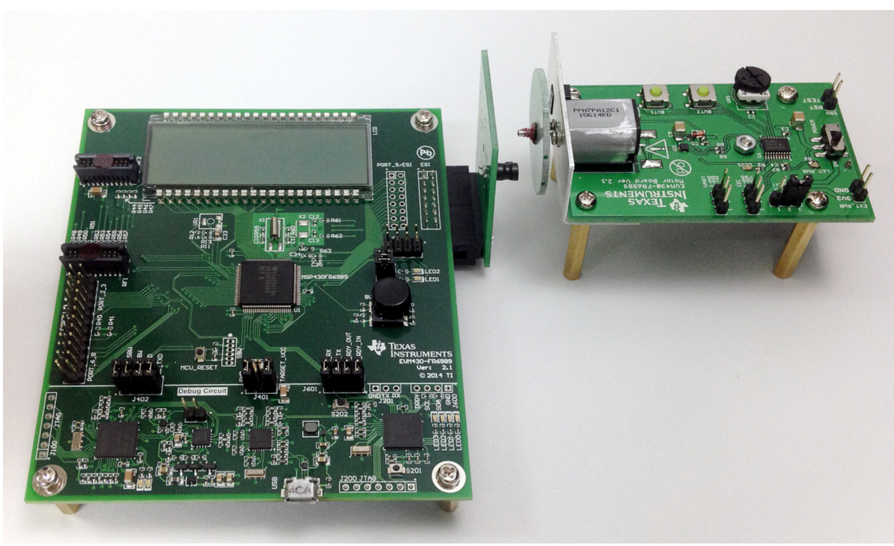


Figure 1. EVM430-FR6989

1.2 FlowESI GUI

The FlowESI GUI is a PC software tool that allows the user to develop configuration code for the ESI module without digging through the user's guide. The ESI is configured by simple clicks on this GUI without typing source code. The code generator creates the source code or fully functional Code Composer Studio™ (CCS) or IAR projects. The generated project consists of communication function; if enabled, that function enables communication between the FlowESI GUI and the target board for monitoring ESI status in real time.

Visit http://www.ti.com/tool/FLOWESI-GUI for a detailed description.

1.3 EnergyTrace Technology

EnergyTrace technology for MSP430 microcontrollers is an energy-based code analysis tool that measures and displays the application's energy profile and helps to optimize it for ultra-low-power consumption. A special debugger circuitry calculates the amount of energy being transferred to the target devices. Unlike other measuring devices like multi-meters, even the shortest device activity that consumes energy contributes to the overall recorded energy.

EnergyTrace++ technology, also known as EnergyTrace+[CPU States]+[Peripheral States], brings the capabilities of EnergyTrace to the next level. When debugging with devices that contain the built-in EnergyTrace++ support, the technology yields information about energy consumption as well as the internal state of the microcontroller. The debugger records the ON/OFF status of the peripherals and all system clocks (regardless of the clock source) as well as the low-power mode (LPM) currently in use. This tool provides a means of directly verifying whether an application is demonstrating the expected behavior at the correct points in the code, such as ensuring that a peripheral is turned off after a certain activity.

Visit http://www.ti.com/tool/energytrace for a detailed description.

2 Test Setup

The EVM430-FR6989 evaluation board is used for the testing. Install CCS (version 6 or higher) and FlowESI GUI on the computer. Basic knowledge of CCS is expected. Find the CCS user's guide and a detailed description at http://www.ti.com/tool/ccstudio.

3 Test Procedure

The test is divided into two sections. Section 3.1 describes the procedure to generate CCS project for the EVM430-FR6989 and to monitor the ESI status using the FlowESI GUI. Section 3.2 describes the procedure to modify the code and optimize the power consumption with the help of EnergyTrace technology.

To setup the hardware:

1. Connect the main board and the PC with an USB cable.
2. Connect the jumpers on the main board listed below.

Table 1. Jumper Configuration for EVM430-FR6989

NAME	PIN	FUNCTION
J402	1-2 3-4	TEST_SBW RST_SBW
J401	1-2 3-4	GND TARGET_VCC
J601	1-2 3-4 5-6 7-8	COMM_RX COMM_TX COMM_RDY_OUT COMM_RDY_IN

3. Plug the sensor board to the main board.
4. Insert batteries to the motor board. Short pin 1-2 of the PWR_SEL with a jumper.
5. Switch on the motor board.
6. Rotate the variable resistor VR1 at the middle of its adjustable range.

3.1 Code Generation Using FlowESI

3.1.1 FlowESI GUI

1. Open FlowESI GUI.
2. In the main screen, click the MSP icon and click the workspace to place an MCU. Click the LC icon and then the workspace to place an LC sensor. Place two LC sensors to the workspace.

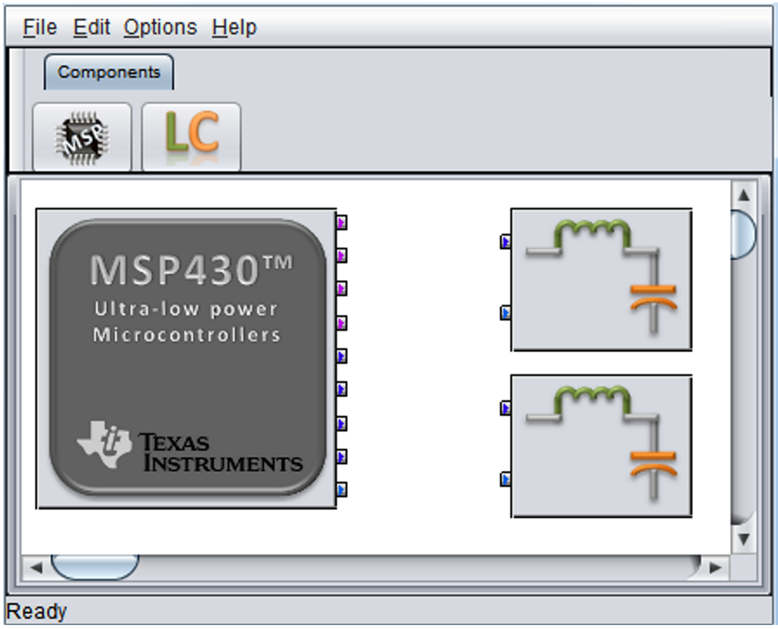


Figure 2. Place an MCU and Two LC Sensors

3. Click *Options* on the menu bar and select "Auto-Connect sensors". The two LC sensors are connected to ESICH0 and ESICH1 of the MCU.

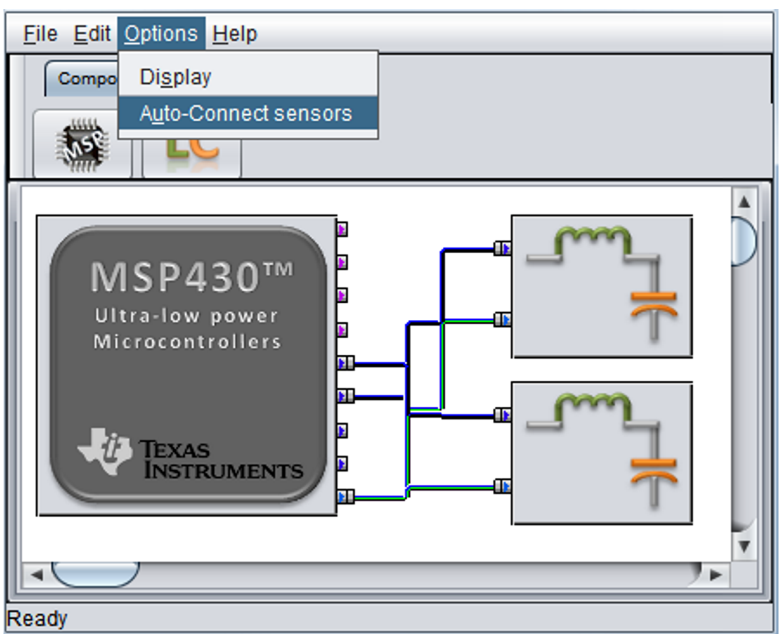


Figure 3. Connect the MCU and LC Sensors

4. Double click the MSP430 in the workspace to open the *MSP properties* window.
5. In the *Output Selection* tab, choose "EVM430-FR6989" under "Hardware Selection". Under "Output Options", check "Display rotations on LCD" and "Enable target communication via HID/UART communication bridge".

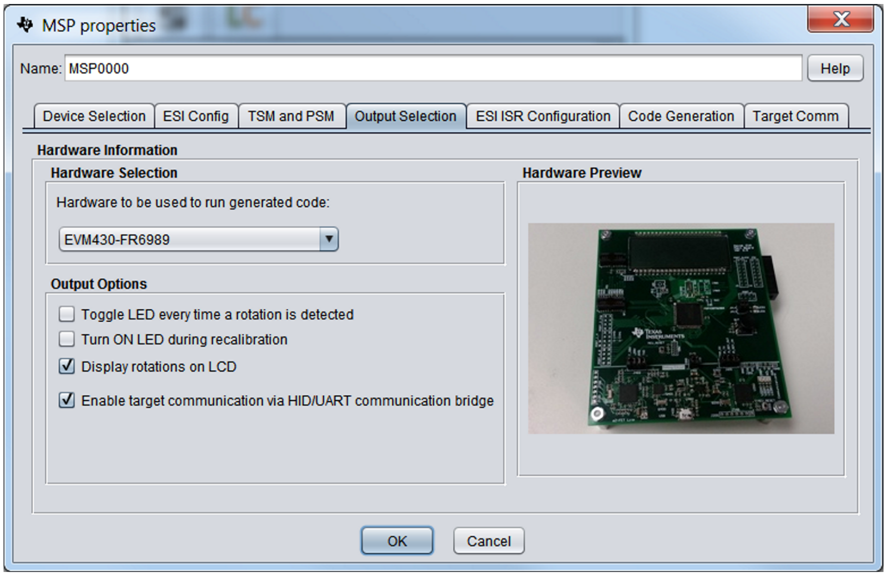


Figure 4. Output Selection

6. In the *Code Generation* tab, select "Full CCS Project" in the "Generation Type". Click *Generate* to generate the CCS project. Click *Yes* to confirm when a message pops up. Remember the directory where the project is being created.

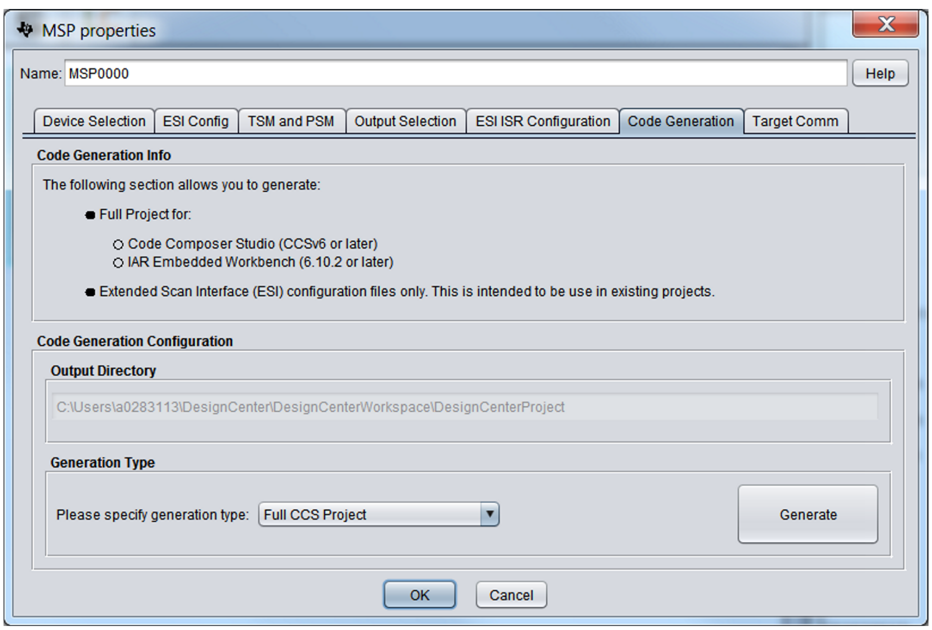


Figure 5. Code Generation

7. Do not close FlowESI GUI yet.

3.1.2 CCS

1. Open CCS.
2. Make sure the EnergyTrace function is enabled. (*Menu→Window→Preferences*)

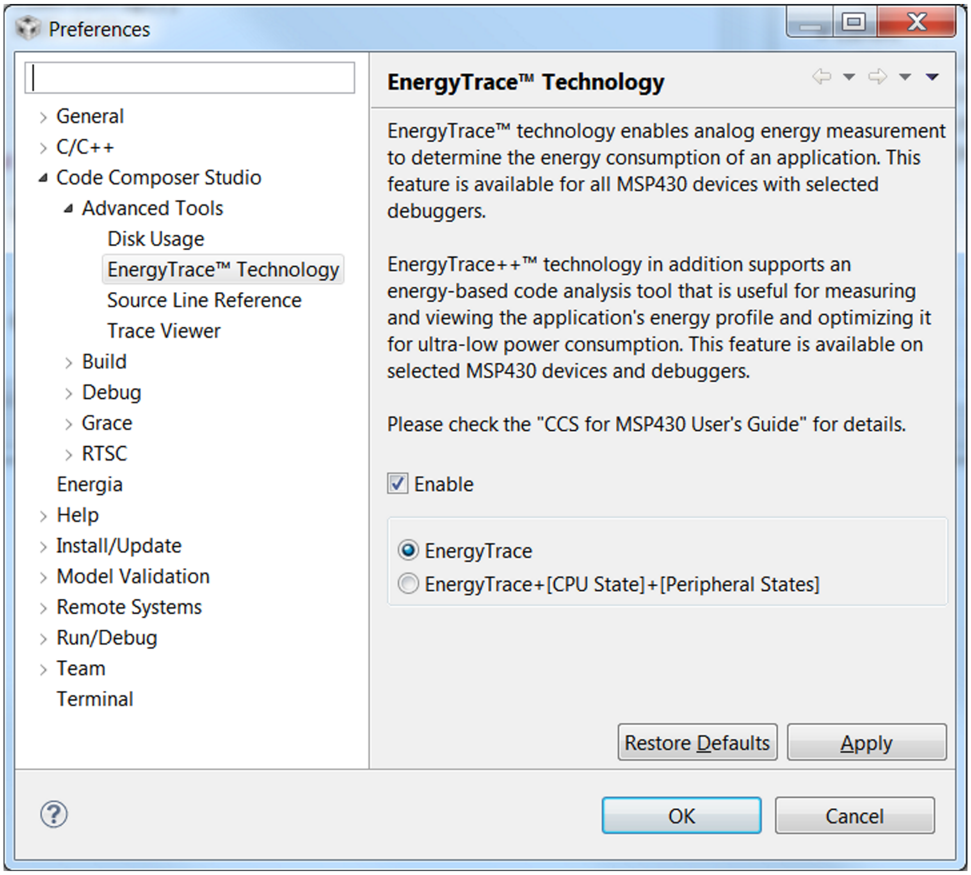


Figure 6. Enable EnergyTrace Technology

3. Import the CCS project.
   (a) Go to *Menu→File→Import*.
   (b) Select *Code Composer Studio→CCS Project→Next*.

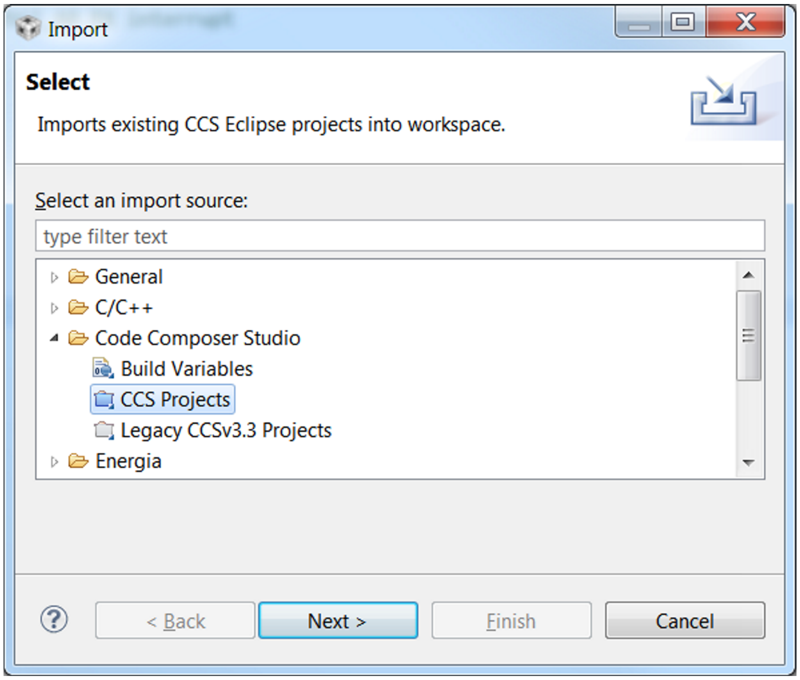


Figure 7. Import CCS Project

4. Browse for the directory where the CCS project is generated by the FlowESI GUI.

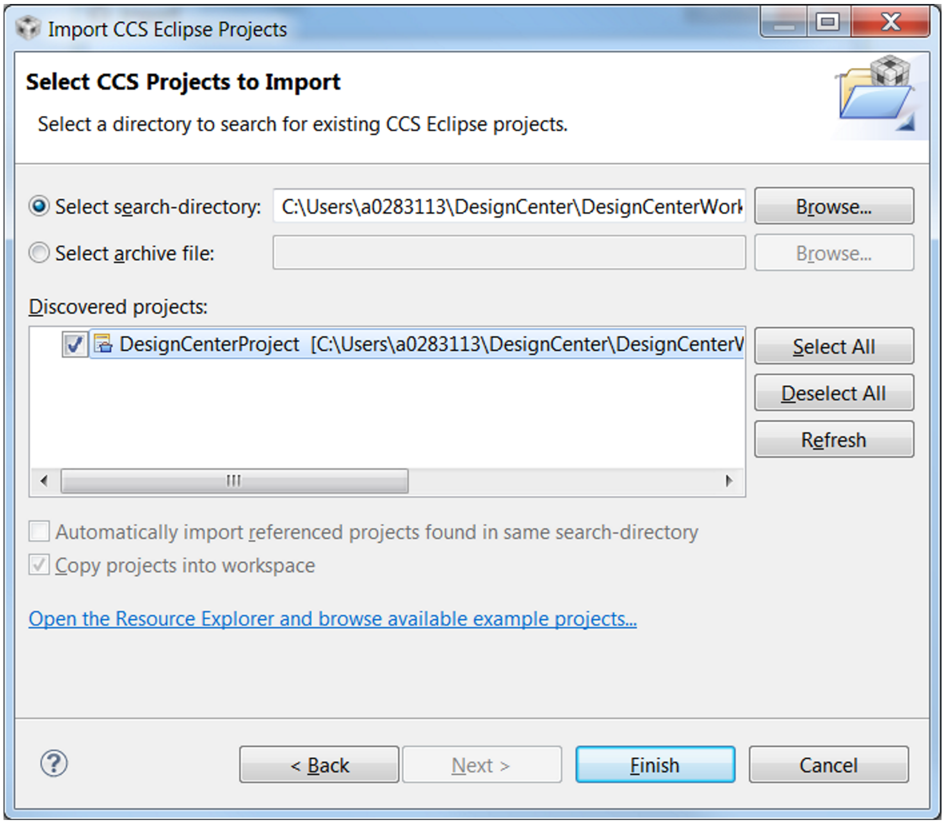


Figure 8. Select "DesignCenterProject"

5. Select and check "DesignCenterProject", then click *Finish*.
6. Start the debug session (*Menu→Run→Debug*, or press F11). Wait for some time to program the MCU of the test board.
7. Run the code (*Menu→Run→Resume*, or press F8).

3.1.3 Go Back to FlowESI GUI

1. In the *Target Comm* tab, click *Connect* in "HID/UART Bridge".

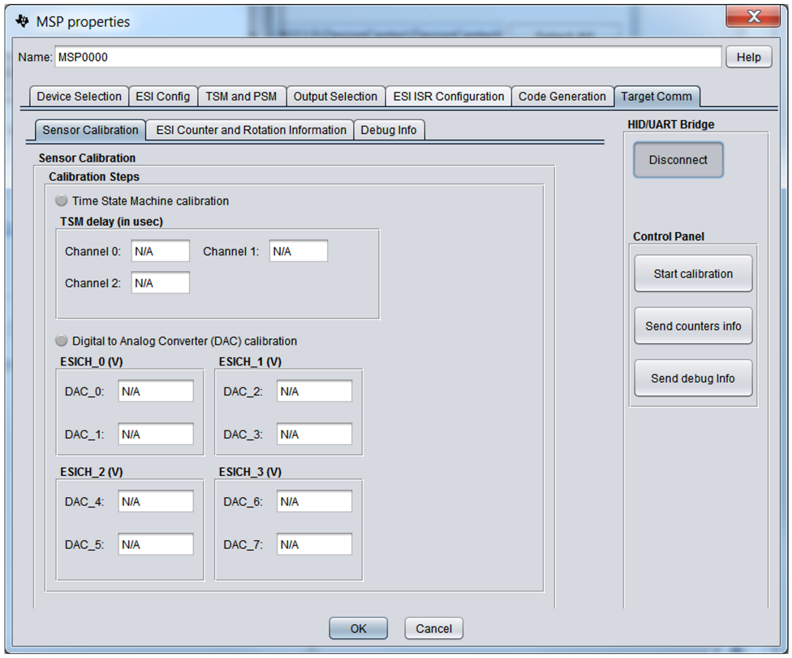


Figure 9. Sensor Calibration

2. Calibrate the LC sensors.
   (a) Click *Start calibration*.
   (b) Place the rotor disc of the motor board 5 mm away from the LC sensor. Click *Start* in the message box.

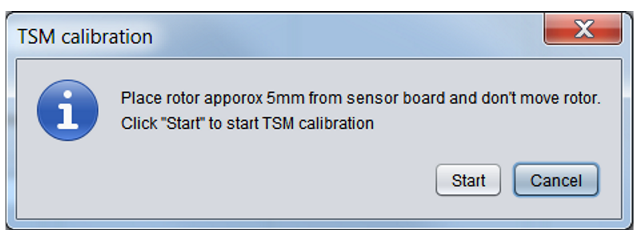


Figure 10. Place Rotor 5 mm From the Sensors

   (c) When the LCD screen of the EVM board shows "8888", a message box pops up from the FlowESI GUI. Start rotating the rotor disc by pressing BUT 2 of the motor board. Do not change the distance between the LC sensors and the rotor disc of the motor board. Click *OK* on the message box to continue.

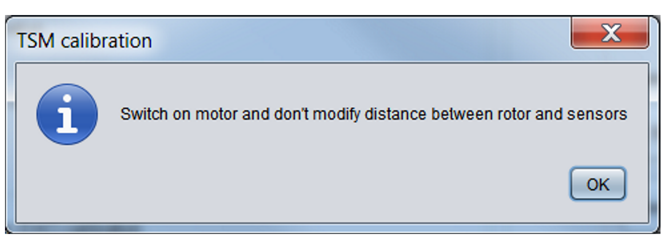


Figure 11. Switch on Motor

   (d) When calibration is done, the calibrated value is shown on the GUI.

3. Monitor the ESI status.
   (a) In the *ESI Counter and Rotation Information* tab, click *Send counters info* in "Control Panel".
   (b) The EVM board reports the counter value to the PC GUI. Change the rotating speed or direction of the motor board to see the effect.

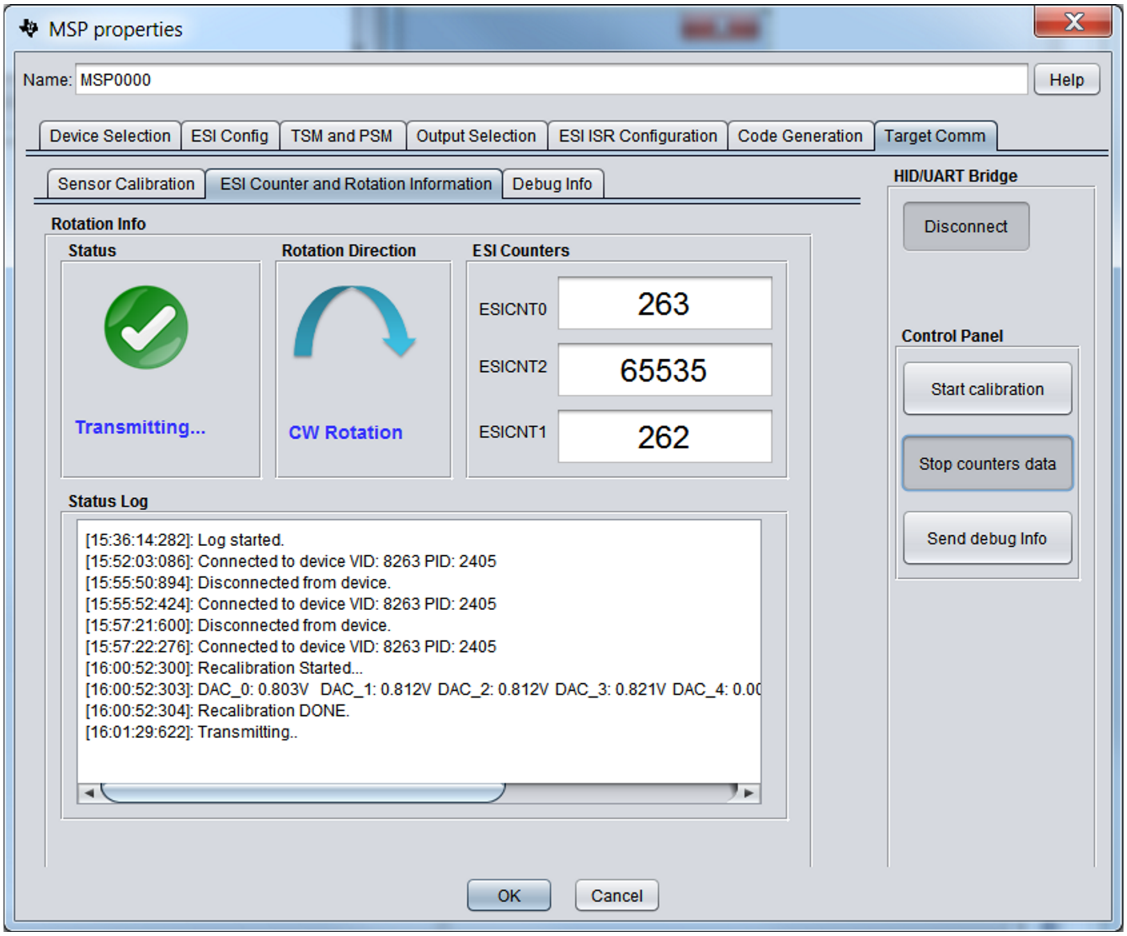


Figure 12. ESI Counter and Rotation Information

4. Prepare for next section.
   (a) Press *Stop counters data*.
   (b) Press *Disconnect*.
   (c) Close the FlowESI GUI.

3.2 Optimizing Power Consumption With EnergyTrace

3.2.1 Regenerate Code Without FlowESI GUI Support

1. In CCS, rename the current project "DesignCenterProject" to other name such as "DesignCenterProject_old".
2. Repeat Steps 4 through 6 in Section 3.1.1 to regenerate the code project.
   In Step 5, uncheck "Enable target communication via HID/UART communication bridge" to disable the FlowESI GUI function.
3. Repeat Steps 3 through 7 in Section 3.1.2 to re-import and run the new generated CCS project.
4. Now, manually calibrate the LC sensors.
   (a) Place the rotor disc of the motor board 5 mm away from the LC sensor.
   (b) When the LCD screen of the EVM board shows "8888", start rotating the rotor disc by pressing BUT 2 of the motor board. Do not change the distance between the LC sensors and the rotor disc of the motor board.
   (c) When calibration is done, the counter value shows on the LCD indicating the number of revolution has been detected.

3.2.2 View the Power Consumption of the Current Setup

1. Pause the debug session (*Menu→Run→Suspend*, or press Alt+F8).
2. Set the EnergyTrace measure time to five seconds.

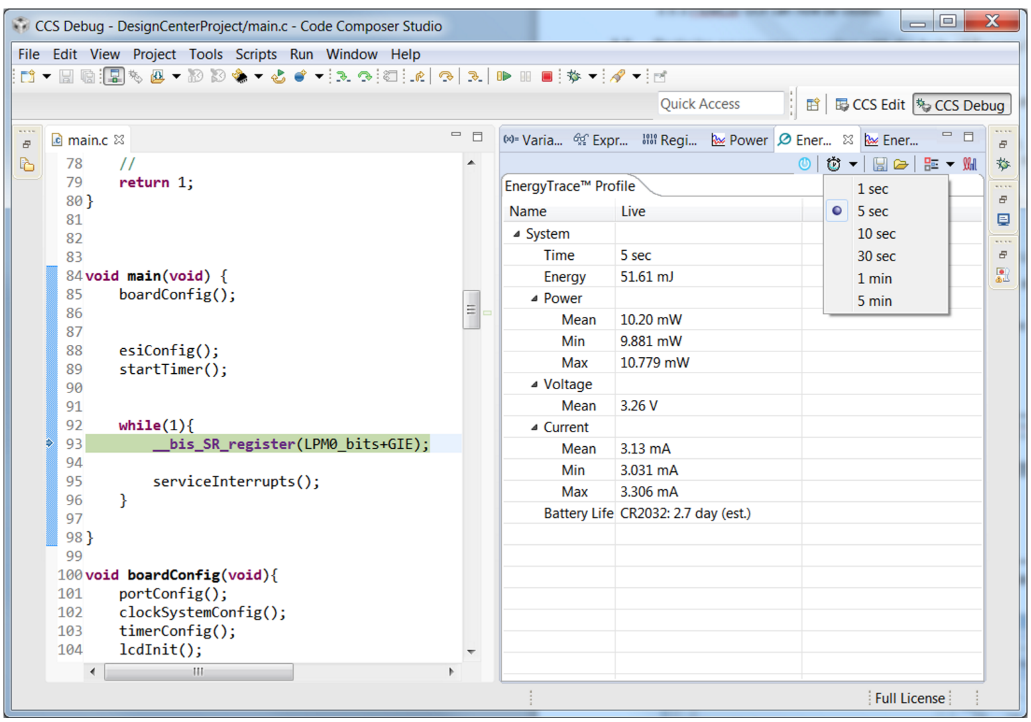


Figure 13. Set EnergyTrace Measure Time

3. Run in "Free Run" mode (*Menu→Run→Free Run*).
4. Wait for several seconds. The result will show under the *EnergyTrace™ Profile* tab.
5. The power consumption of the current setup is over 3 mA, which is quite high.

3.2.3 View the Usage Inside the MCU

1. Pause the debug session (*Menu→Run→Suspend*, or press Alt+F8).
2. Switch to EnergyTrace++ mode by clicking the button shown in Figure 14.

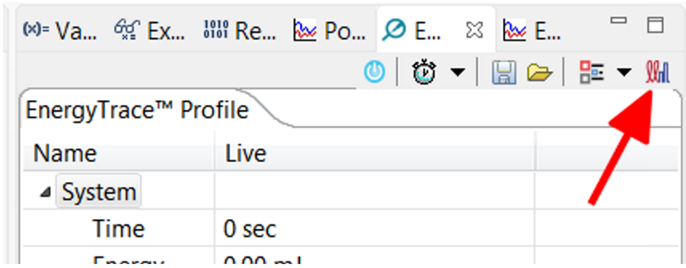


Figure 14. Switch to EnergyTrace++

3. Resume the debug session (*Menu→Run→Resume*, or press F8).
4. Wait for several seconds. The result shows under the *States* tab.

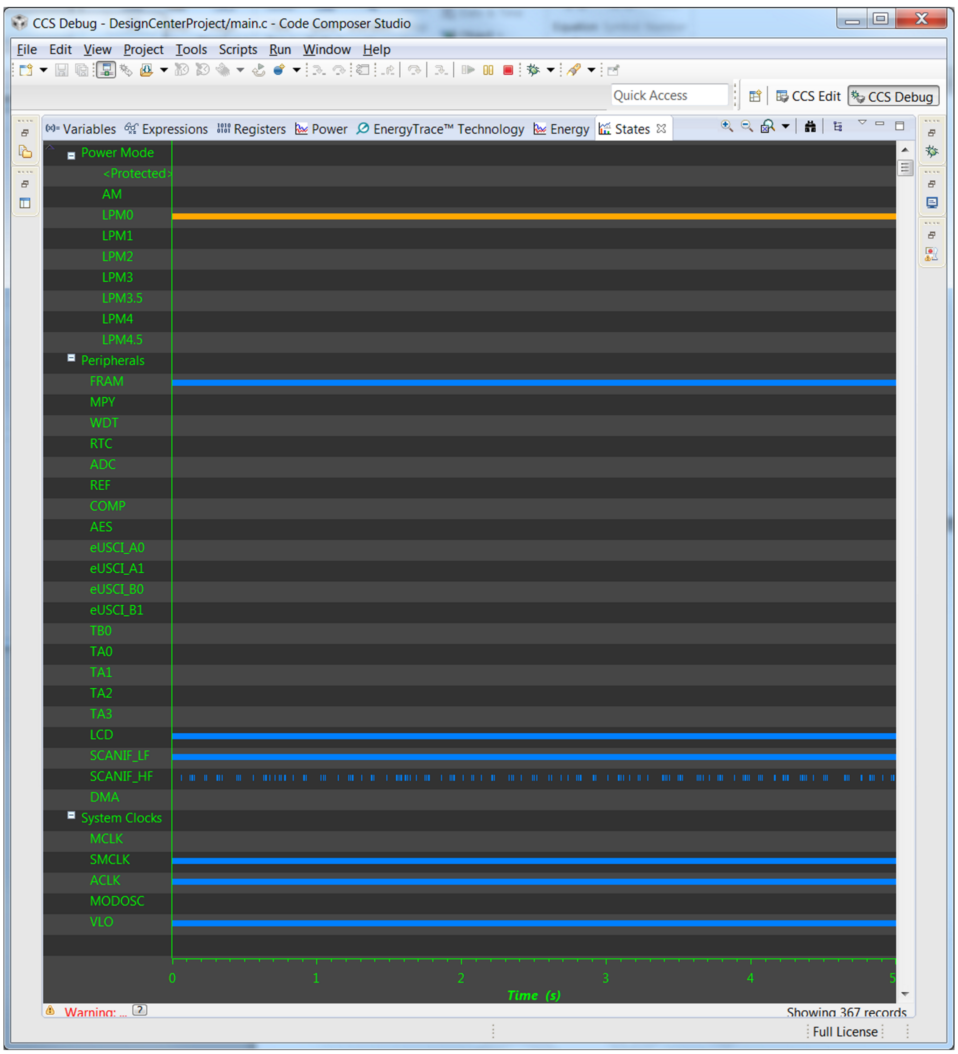


Figure 15. Module Status Before Code Optimization

5. The result shows that the system always in LPM0 mode. Several peripherals are also turned on. In Section 3.2.4, the code will be modified to optimize the power consumption.

3.2.4 Optimize the Code

1. Open "main.c".
2. Modify the "void main(void)" function as follows:

```
void main(void)
{
    boardConfig();

    // There are pull-up resistors connected.
    // Set output high to prevent leakage current
    GPIO_setOutputHighOnPin(GPIO_PORT_P1,
                            GPIO_PIN2 | GPIO_PIN6 | GPIO_PIN7);

    // Set LFXT at lowest drive current
    CS_LFXTStart(CS_LFXT_DRIVE0);

    // LC sensor calibration is done here
    esiConfig();

    // Optional: Disable LCD for further lower power consumption
    // LCDCCTL0 &= ~LCDON;

     startTimer();

     while(1)
     {
         // Use LPM3 instead of LPM0
         // __bis_SR_register(LPM0_bits+GIE);
         __bis_SR_register(LPM3_bits+GIE);

         serviceInterrupts();
    }
}
```

Figure 16. Code

3. Run the code.
4. Repeat Step 4 in Section 3.2.1 to calibrate the LC sensors.
5. Pause and resume debug mode to trigger EnergyTrace.

6. The new result shows that fewer modules inside the MCU are used after modifying the code.

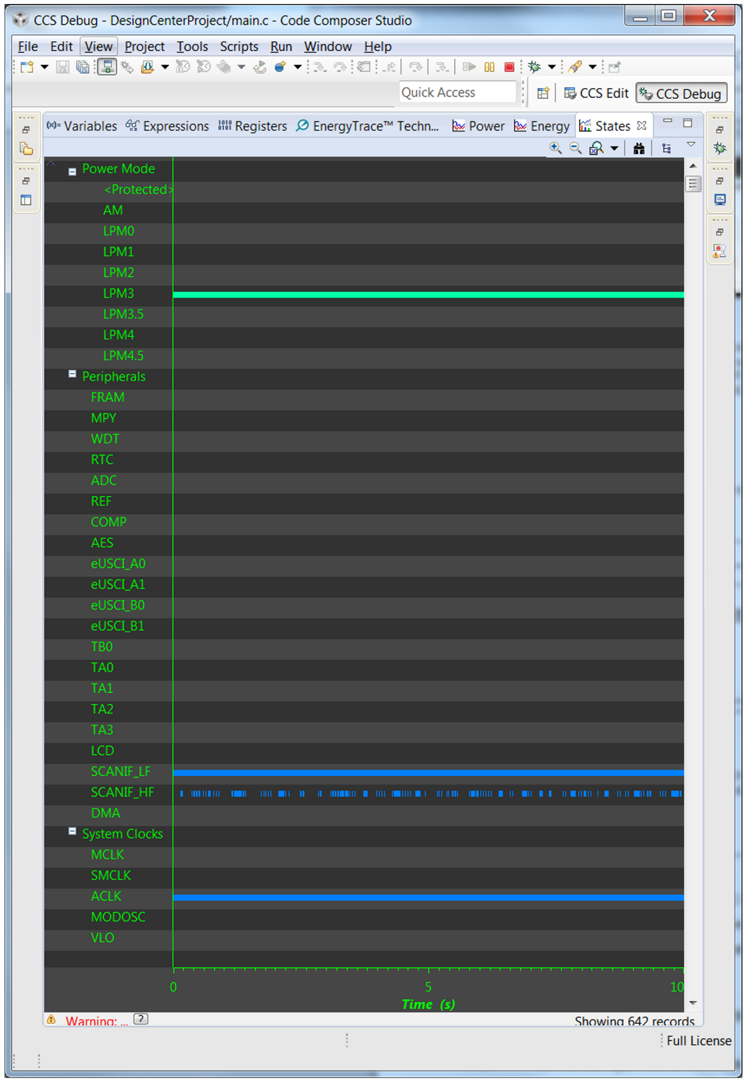


Figure 17. Module Status After Code Optimization

7. Switch back to EnergyTrace mode by pressing the button previously shown in Figure 14. Click *Menu→Run→Free Run* to record the system power consumption.

8. Now the overall power consumption is greatly reduced from 3 mA down to around 10 µA.

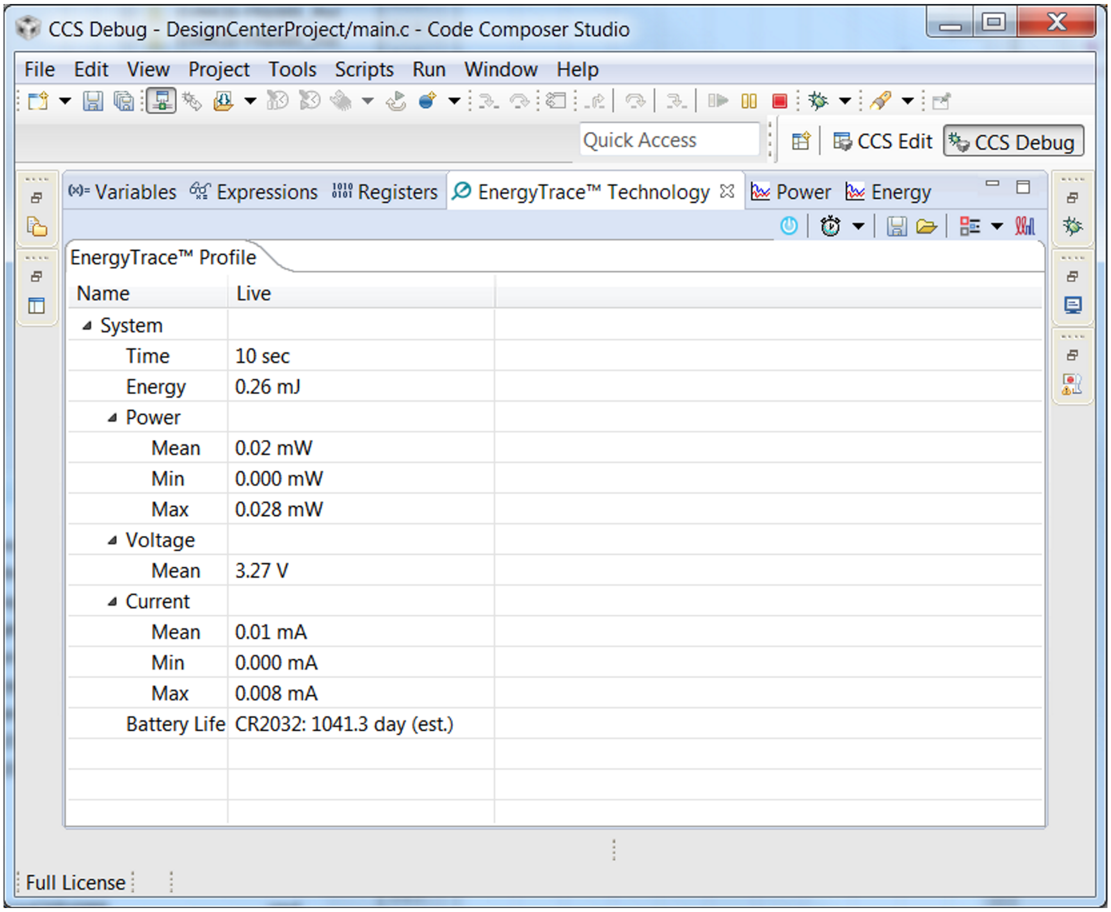


Figure 18. Power Consumption After Code Optimization

3.3 Note

To measure the actual current consumption of the board using EnergyTrace, use Free Run mode. Otherwise, the board draws more current for JTAG activities.

To monitor the state of different modules using EnergyTrace++, do not use Free Run mode. Press *Resume* instead.

4 Result

Table 2 shows the difference in power consumption and MCU state before and after code optimization. By just setting high to the pins where the pull-up resistors are connected, the current is greatly reduced by 3 mA. By turning off several modules and running in LPM3 mode instead of LPM0 mode, the system current is further reduced by about 90 µA.

Table 2. Comparison in Power Consumption Before and After Code Optimization

PARAMETER	BEFORE CODE OPTIMIZATION	AFTER CODE OPTIMIZATION
System current	3.1 mA	10 µA
Current consumed by pull-up resistors	(By calculation) I^2C: 2.97 mA (2×2.2-kΩ resistors @ 3.27 V) Nav switch: 32.7 µA (100 kΩ @ 3.27 V)	None
MCU state	Sleep mode: LPM0 Active module: • SCANIF_LF • SCANIF_HF • ACLK • SMCLK • LCD • FRAM • VLO	Sleep mode: LPM3 Active module: • SCANIF_LF • SCANIF_HF • ACLK

5 Design Files

5.1 Schematics

To download the most recent schematics, see the design files at TIDM-FLOWESI-ETRACE.

5.2 Bill of Materials

To download the most recent bill of materials (BOM), see the design files at TIDM-FLOWESI-ETRACE.

5.3 Layer Plots

To download the most recent layer plots, see the design files at TIDM-FLOWESI-ETRACE.

5.4 Gerber Files

To download the most recent Gerber files, see the design files at TIDM-FLOWESI-ETRACE.

5.5 Software Files

Software files are generated by the user. See Section 3.1.

6 About the Author

ZACK MAK is a system application engineer at Texas Instruments where he is responsible for developing reference design solutions for the industrial segment. Zack earned his bachelor of electronic and communication engineering from the City University of Hong Kong.

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