User's Guide

SLAU574K–March 2015–Revised June 2018

IAR Embedded Workbench® for Arm® 8.10.1+ for SimpleLink™ MSP432™ Microcontrollers

This manual describes the use of IAR Embedded Workbench® for Arm® (EWARM) with the SimpleLink™ MSP432™ low-power microcontrollers.

This guide is for EWARM releases 8.10.1 and later. Some descriptions in this guide, like debugging, are also valid for EWARM versions before 8.10.1.

Contents

1 Installing Embedded Workbench for Arm 4
2 Creating a SimpleLink Project From the MSP432 SDK 4
3 Creating a Project From MSP432 CMSIS Device Family Pack (DFP) 7
4 Debugging the Application 8
5 Using Serial Wire Output (SWO) Hardware Trace Analyzer 24
6 Using ETM Trace (MSP432E4 Devices Only) 26
7 Erasing the Bootloader (BSL) (MSP432P4xx Devices Only) 27
8 EnergyTrace™ Technology 29
9 Frequently Asked Questions 35
10 Additional IAR EWARM Information 37
11 References 37

List of Figures

1 TI Resource Explorer 5
2 SDK Quick Start Guide 6
3 Select Devices 7
4 Select Example 7
5 Change Device 8
6 Select Project Options 9
7 Selecting the TI XDS Debug Probe 10
8 Debug Download Options 11
9 Dialog Box Asking to Perform a Factory Reset 12
10 Selecting MSP-FET as Debug Probe 12
11 Securing Device With MSP-FET 13
12 MSP-FET Release JTAG on Go 14
13 Select the Segger J-Link Debug Probe 15
14 Enabling Usage of Flash Loaders 16
15 Using J-Link Commander to Enable Power Output to Target System 16
16 Launch J-Link Control Panel 17
17 J-Link Control Panel 18
18 J-Link Script Detecting That the Device Has Been Secured 18
19 Device Has Been Unlocked and Erased 18
20 Selecting the IAR I-jet Debug Probe 19
21 Enabling Use of Flash Loaders 20
22 Debug Probe Setup 21

23 Dialog Box Asking to Perform a Factory Reset 21
24 Adding Driver Library to a Project 22
25 Adding ROM Symbol to Debugger 23
26 Stepping Through ROM DriverLib Source Code 24
27 Trace Settings Configured From Running Debug Session Using Debugger Menu 25
28 SWO Trace Window Settings 25
29 SWO Configuration 25
30 ETM Trace With I-Jet Trace 26
31 Edit Flash Loader Settings 27
32 Edit Info Memory Settings 27
33 Add --bsl_erase Option 28
34 Pulse Density and Current Flow 29
35 Debug Session With EnergyTrace+ Windows 30
36 TI MSP-FET Emulator Pulldown Menu With EnergyTrace+ Related Functions 31
37 Enabling the State Log Window 31
38 State Log Window With EnergyTrace+ Data 32
39 State Log Summary With EnergyTrace+ Data 32
40 Power Log Setup Window 33
41 Power Log Window With EnergyTrace+ Data 33
42 Timeline With Power Log and State Graphs Disabled 34
43 Timeline With EnergyTrace+ Data 34
44 Choosing SWD in Debugger Settings 36

Trademarks

SimpleLink, MSP432, E2E, LaunchPad, EnergyTrace are trademarks of Texas Instruments.
CoreSight is a trademark of Arm Limited.
Arm is a registered trademark of Arm Limited.
IAR Embedded Workbench is a registered trademark of IAR Systems.
All other trademarks are the property of their respective owners.

Preface: Read This First

How to Use This User's Guide

This manual describes only EWARM features specific to the MSP432 low-power microcontrollers. It does not fully describe the MSP432 microcontrollers or the complete development software and hardware systems. For details on these items, see the appropriate TI documents listed in Important MSP432 Documents on the Web.

Important MSP432 Documents on the Web

The primary sources of information about MSP432 MCUs are the device-specific data sheets and user's guides. The SimpleLink MSP432 website contains the most recent version of these documents.

Documents that describe the IAR Embedded Workbench for Arm can be found at www.iar.com. The TI E2E™ Community support forums can provide additional help.

Information about the TI XDS100 and XDS200 debug probes is not included in this document and can be found at www.ti.com/tool/xds100 and www.ti.com/tool/xds200.

Documentation for third-party tools, such as the SEGGER J-Link debug probe, can be found on the respective third-party website.

If You Need Assistance

Support for the MSP432 devices and the hardware development tools is provided by the TI Product Information Center (PIC). Contact information for the PIC can be found on the TI website at www.ti.com/support. The TI E2E Community support forums for the MSP432 MCUs provide open interaction with peer engineers, TI engineers, and other experts. Additional device-specific information can be found on the MSP432 website.

1 Installing Embedded Workbench for Arm

IAR Embedded Workbench for Arm (EWARM) IDE is available from the IAR website or the TI IAR kickstart page. MSP432 low-power microcontrollers require EWARM 7.40.2 or higher.

The TI MSP432 CMSIS device family packs require EWARM 8.10.1 or higher.

The SimpleLink MSP432 SDK requires EWARM 7.80.3 or higher. See the SimpleLink MSP432 SDK documentation for finding the right SDK for your IDE version.

CMSIS device packs (DFP) are an alternative way of adding device support to EWARM and support is available in EWARM 8.10.1 and higher. CMSIS DFPs are available for all SimpleLink MSP432 MCUs through the integrated pack manager.

2 Creating a SimpleLink Project From the MSP432 SDK

The SimpleLink MSP432 software development kit (SDK) contains software examples, projects, documentation, application notes, and training for all MSP432 devices. This includes example projects for IAR EWARM that work with MSP432 MCUs. For more information, visit the following TI SimpleLink SDK pages.

NOTE: The SimpleLink MSP432 SDK supports the MSP432P4xx devices, and the SimpleLink MSP432E4 SDK supports the MSP432E4 devices

The SDK for your device and the examples can be either downloaded in form of an installer directly from the above links or, in the newer IAR EWARM 8.11.3+ versions, within the integrated TI resource explorer inside the IDE.

For accessing the online TI resource explorer open the IAR Information Center for Arm and click **Example projects**. In the section "Links to silicon partner example applications", click **Texas Instruments SimpleLink MCUs** and then **TI SimpleLink™ MSP432™ Software Development Kit (SDK)**. This accesses the online TI resource explorer (see Figure 1).

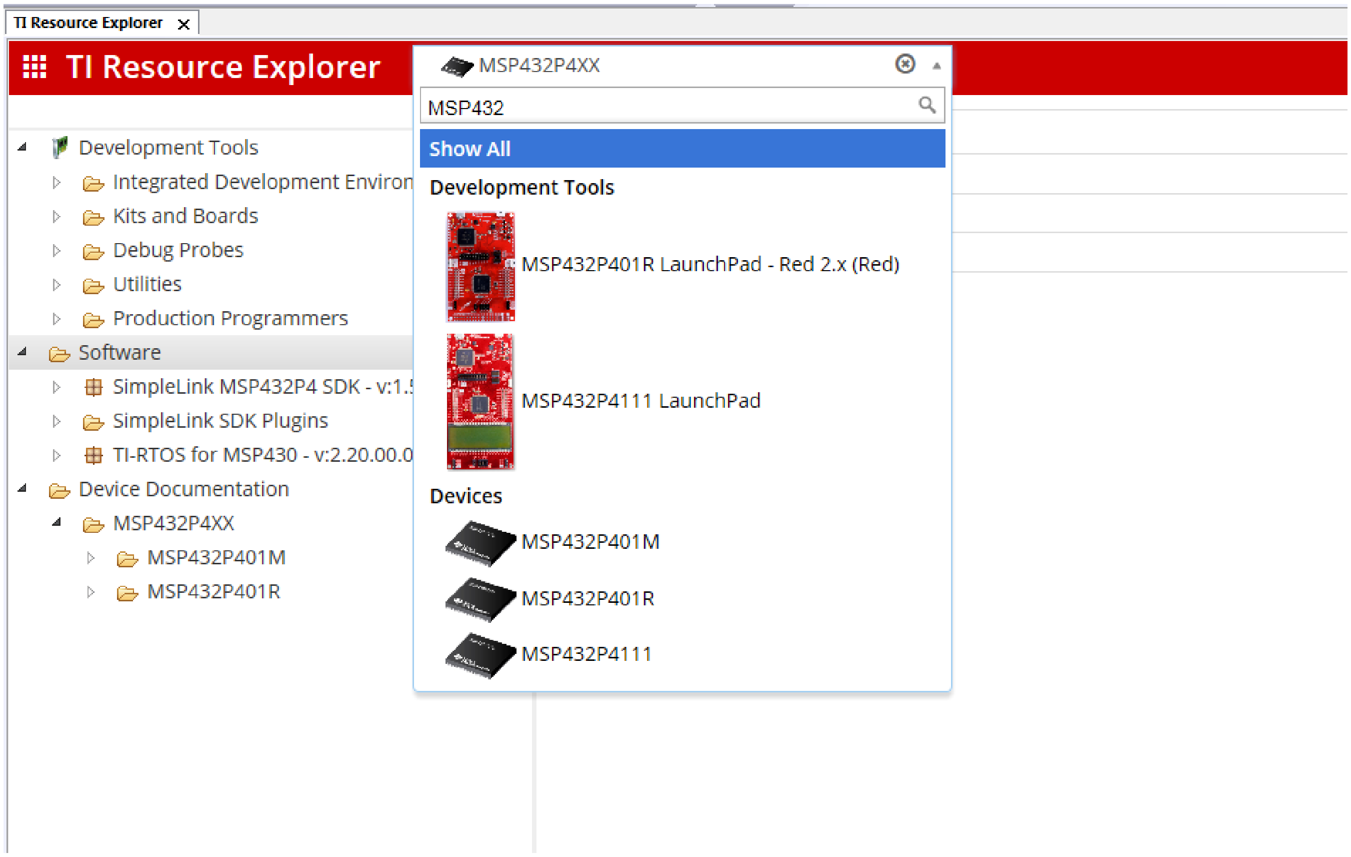
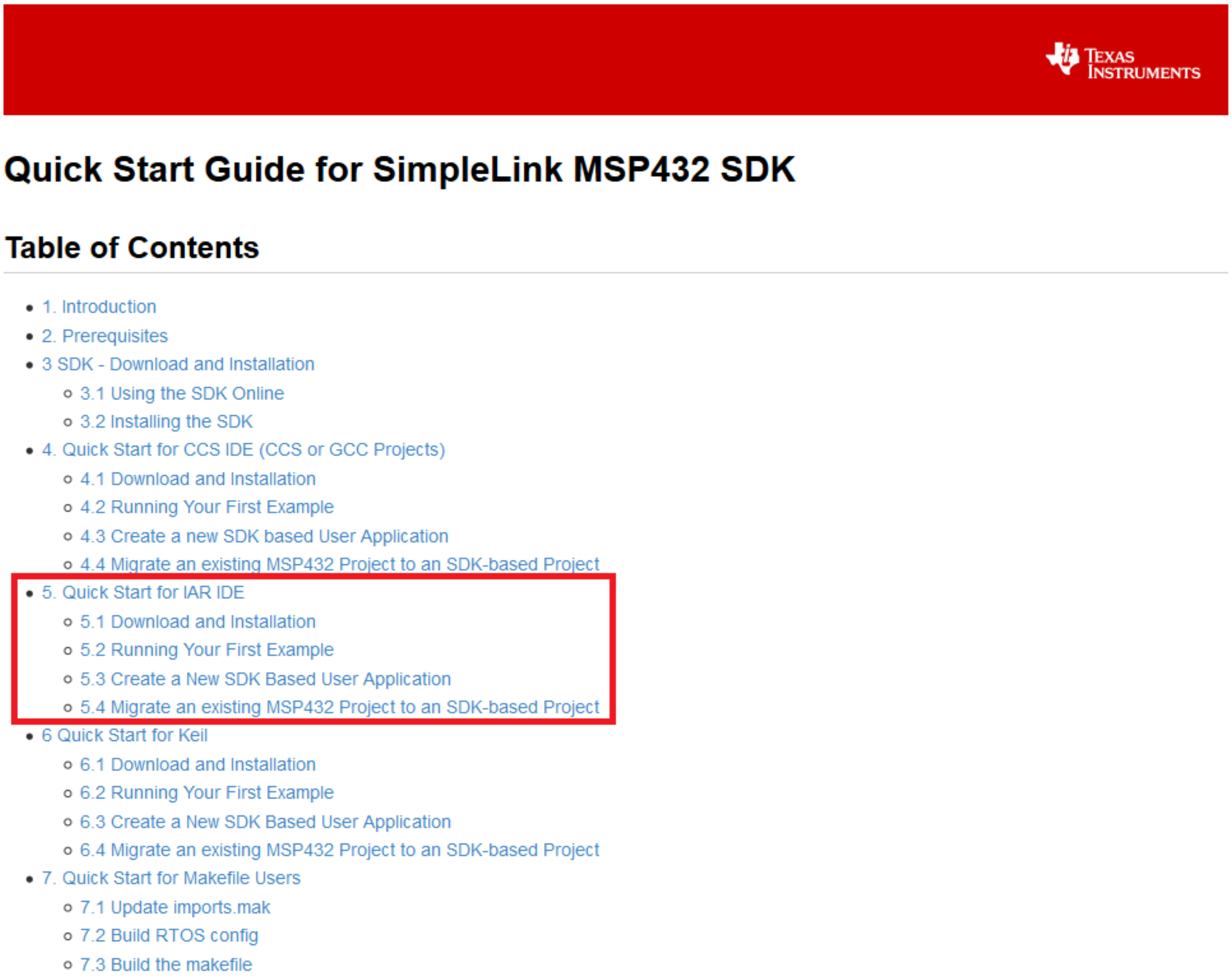


Figure 1. TI Resource Explorer

In the Resource explorer, use the search dialog to limit the examples to the correct ones for your device. Any example you select starts the download of a zipped SDK for your device. Unpack this zipped file to C:\TI.

NOTE: The TI example package in the "Example projects that can be downloaded" does not contain any examples for SimpleLink MCUs.

Regardless of how you obtained the SDK, all documentation for it can be found inside the SDK installation path in the docs folder. Open Documention_Overview.html from that folder and then navigate to the Quick Start Guide for your IDE (see Figure 2). Follow the instructions in this guide for required configuration before importing SDK examples.



Quick Start Guide for SimpleLink MSP432 SDK

Table of Contents

- 1. Introduction
- 2. Prerequisites
- 3 SDK - Download and Installation
  - 3.1 Using the SDK Online
  - 3.2 Installing the SDK
- 4. Quick Start for CCS IDE (CCS or GCC Projects)
  - 4.1 Download and Installation
  - 4.2 Running Your First Example
  - 4.3 Create a new SDK based User Application
  - 4.4 Migrate an existing MSP432 Project to an SDK-based Project
- 5. Quick Start for IAR IDE
  - 5.1 Download and Installation
  - 5.2 Running Your First Example
  - 5.3 Create a New SDK Based User Application
  - 5.4 Migrate an existing MSP432 Project to an SDK-based Project
- 6 Quick Start for Keil
  - 6.1 Download and Installation
  - 6.2 Running Your First Example
  - 6.3 Create a New SDK Based User Application
  - 6.4 Migrate an existing MSP432 Project to an SDK-based Project
- 7. Quick Start for Makefile Users
  - 7.1 Update imports.mak
  - 7.2 Build RTOS config
  - 7.3 Build the makefile

Figure 2. SDK Quick Start Guide

In particular, make sure to execute the following steps:

- Configuring custom argument variables
- RTOS configuration, if you intend to use an RTOS

A list of examples for IAR can be found under **<SDK installation path>/ tools/iar/Examples.html**. Follow the instructions in the quick start guide on how to import these examples to the EWARM IDE.

3 Creating a Project From MSP432 CMSIS Device Family Pack (DFP)

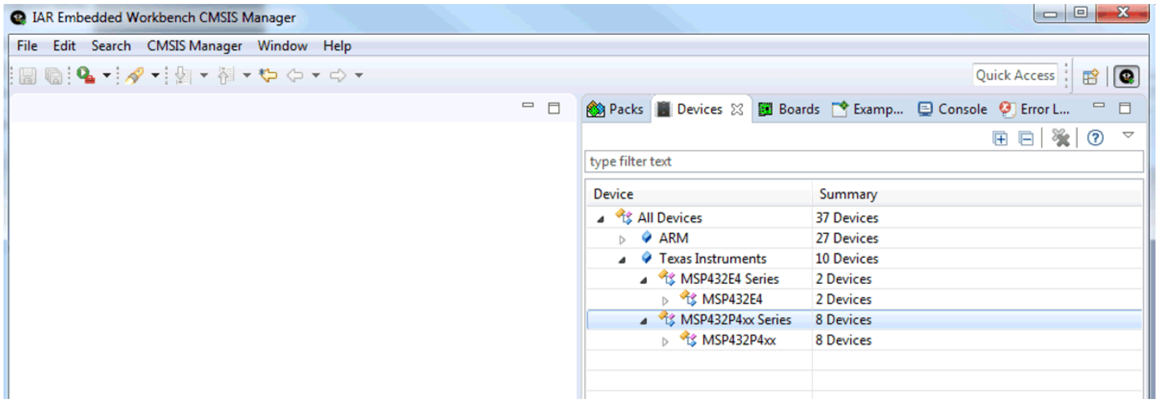
IAR EWARM 8.10 and higher supports CMSIS packs (see the CMSIS Pack Documentation for a description of CMSIS packs). The latest Texas Instruments MSP432P4xx and MSP432E4 Device Family Packs are supported by IAR EWARM IDE 8.10 or higher. The packs can be downloaded with the pack manager in the IDE or directly from the MSP432 CMSIS device family pack page. A device family pack adds device support to the IDE and features some basic examples.

NOTE: Examples from the SimpleLInk MSP432 SDK do not build on CMSIS device family packs in IAR EWARM. These rely on the native device support through the IDE.

NOTE: The following description is for the CMSIS Manager in IAR EWARM 8.30. The procedure might be different in other versions of IAR EWARM.

To create an example from the device family pack:

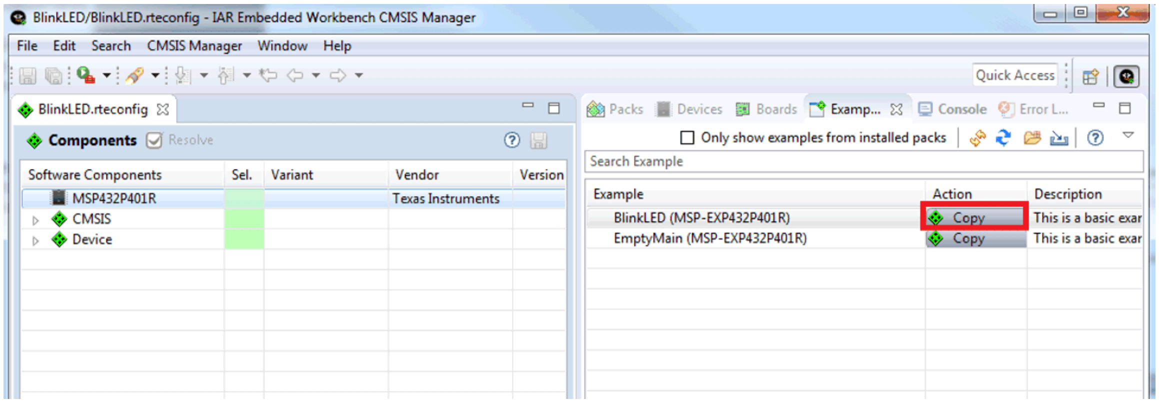
1. In a new workspace, click to start the CMSIS manager, go to the Devices tab, select the desired device family, and go to the examples tab (see Figure 3).


IAR Embedded Workbench CMSIS Manager
File Edit Search CMSIS Manager Window Help
Quick Access
Packs Devices Boards Examp... Console Error L...
type filter text

Device	Summary
All Devices	37 Devices
ARM	27 Devices
Texas Instruments	10 Devices
MSP432E4 Series	2 Devices
MSP432E4	2 Devices
MSP432P4xx Series	8 Devices
MSP432P4xx	8 Devices

Figure 3. Select Devices

2. In the examples tab, select the desired example and click **Copy** to add the example to your workspace (see Figure 4).


BlinkLED/BlinkLED.rteconfig - IAR Embedded Workbench CMSIS Manager
File Edit Search CMSIS Manager Window Help
Quick Access
BlinkLED.rteconfig
Components
Resolve
Packs Devices Boards Examp... Console Error L...
Search Example

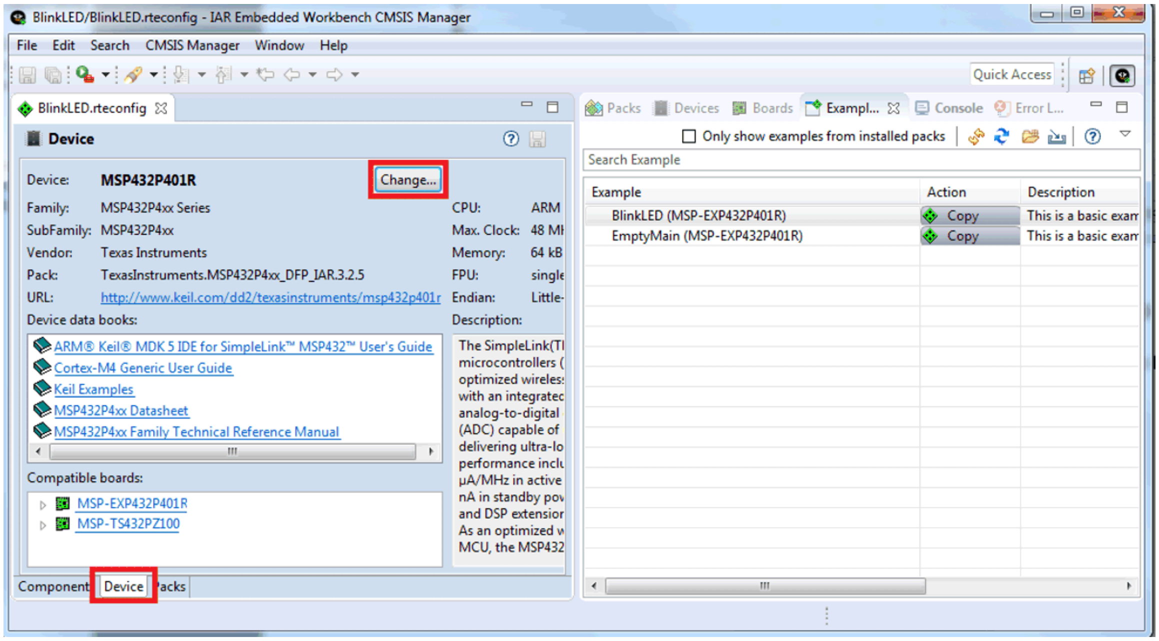
☐ Only show examples from installed packs

Software Components	Sel.	Variant	Vendor	Version
MSP432P401R			Texas Instruments	
CMSIS				
Device				

Example	Action	Description
BlinkLED (MSP-EXP432P401R)	Copy	This is a basic exar
EmptyMain (MSP-EXP432P401R)	Copy	This is a basic exar

Figure 4. Select Example

3. To change the device, if needed, click the device tab in the lower left and then click the **Change** button (see Figure 5).


BlinkLED/BlinkLED.rteconfig - IAR Embedded Workbench CMSIS Manager
File Edit Search CMSIS Manager Window Help
Quick Access
BlinkLED.rteconfig
Device
Change...
Packs Devices Boards Exampl... Console Error L...
Component Device Packs

Device: MSP432P401R
Family: MSP432P4xx Series
SubFamily: MSP432P4xx
Vendor: Texas Instruments
Pack: TexasInstruments.MSP432P4xx_DFP_IAR.3.2.5
URL: http://www.keil.com/dd2/texasinstruments/msp432p401r
CPU: ARM
Max. Clock: 48 MH
Memory: 64 kB
FPU: single
Endian: Little-

Device data books:
- ARM® Keil® MDK 5 IDE for SimpleLink™ MSP432™ User's Guide
- Cortex-M4 Generic User Guide
- Keil Examples
- MSP432P4xx Datasheet
- MSP432P4xx Family Technical Reference Manual

Description:
The SimpleLink(TI microcontrollers (optimized wireless with an integrated analog-to-digital (ADC) capable of delivering ultra-lo performance inclu µA/MHz in active nA in standby pov and DSP extension As an optimized w MCU, the MSP432

Compatible boards:
- MSP-EXP432P401R
- MSP-TS432PZ100

☐ Only show examples from installed packs

Search Example

Example	Action	Description
BlinkLED (MSP-EXP432P401R)	Copy	This is a basic exam
EmptyMain (MSP-EXP432P401R)	Copy	This is a basic exam

Figure 5. Change Device

For projects based on CMSIS packs, most setting must be done in the CMSIS manager. See the IAR EWARM help for more information.

4. Save the settings and return to your workspace. The example project is ready to be compiled.

4 Debugging the Application

The following debug probes can be used with MSP432 MCUs and EWARM.

- Texas Instruments XDS100v2, XDS100v3, XDS200, XDS110 (including the XDS110 stand-alone probe)
- TI MSP-FET (MSP432P4xx Devices Only)
- IAR I-jet
- Segger J-Link

To use a debug probe that is not listed here, check with the vendor of the debug probe or with IAR if you experience problems.

Be aware that for some debug probes power has to be supplied externally to the device. Check the probes user guides for details. For the TI XDS110 stand-alone probe, see the XDS110 Debug Probe User's Guide.

4.1 Using TI XDS100, XDS110, and XDS200 Debug Probes

TI offers a range of debug probes for Arm-based devices, including the XDS100v2, XDS100v3, XDS110, and XDS200 debug probes.

NOTE: TI XDS110 debug probes are enabled for use through the CMSIS-DAP protocol. However, the TI XDS native drivers currently give higher performance.

To use them with EWARM, installation of the XDS emulation package is required. A copy of the emulation package is located in the EWARM installation under **\arm\drivers\ti-xds**. See the Readme.txt document that is also located in this folder. TI recommends installing the emulation package in **c:\ti\xds\ewarm_version**. Do not use XDS emulation packages from other EWARM versions, as they might not be compatible and can result in errors when debugging.

When the emulation package has been installed, XDS debug probes can be selected in the Project Options menu. Right click the active project, then select **Options** (see Figure 6).

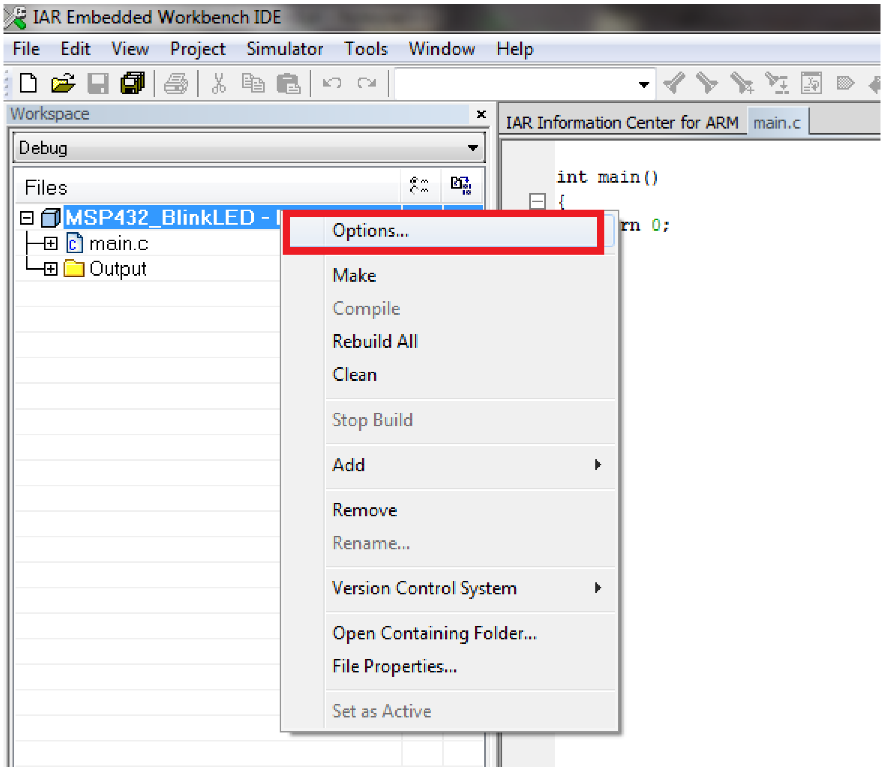
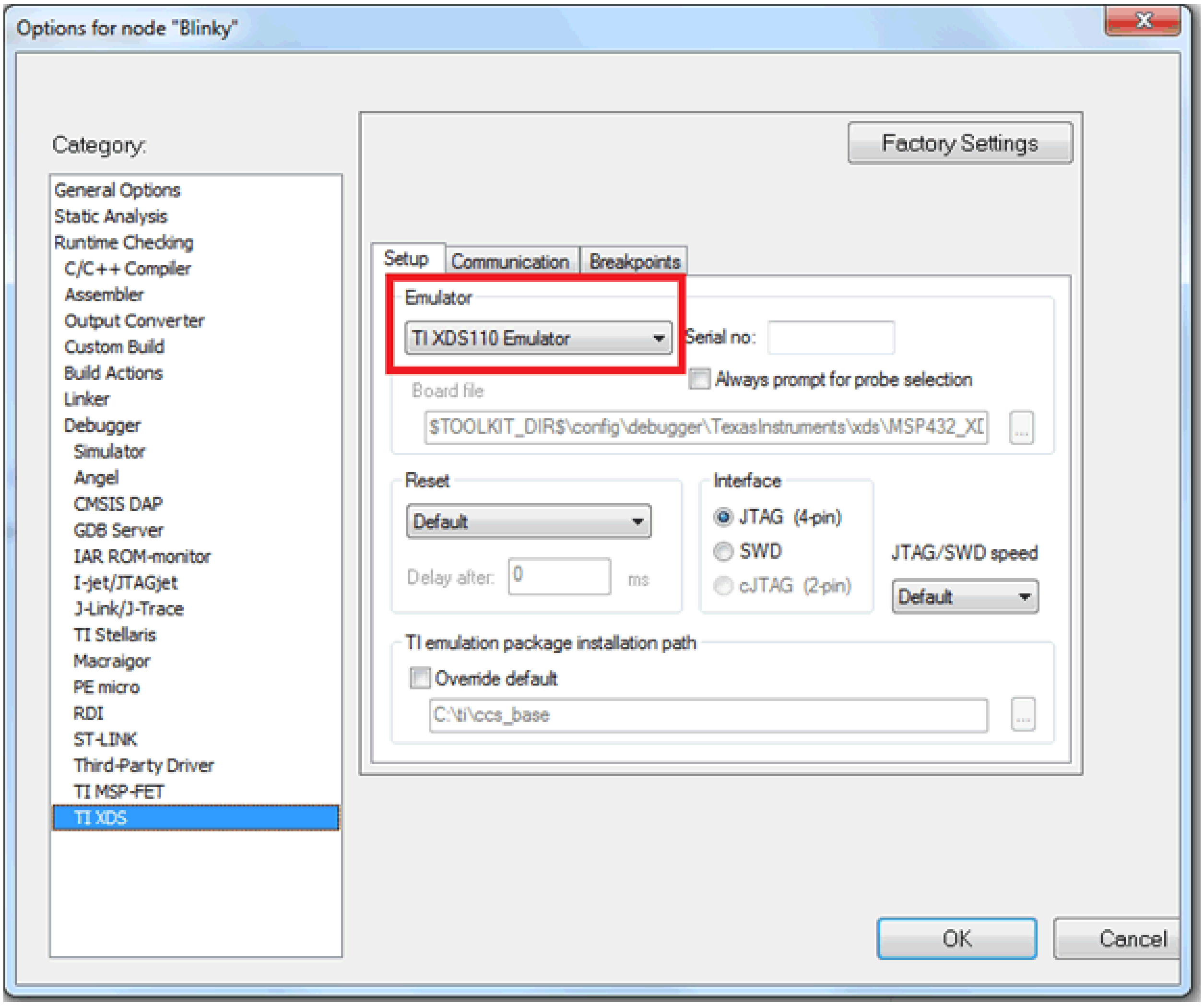


Figure 6. Select Project Options

Click **Debugger** in the left pane and select **TI XDS**. Then click XDS debuggers in the left pane and select the correct XDS debug probe from the list. (see Figure 7).



Options for node "Blinky"

Emulator: TI XDS110 Emulator

Serial no:

□ Always prompt for probe selection

Board file: $TOOLKIT_DIR$\config\debugger\TexasInstruments\xds\MSP432_XI

Reset: Default

Delay after: 0 ms

Interface: ● JTAG (4-pin) ○ SWD ○ cJTAG (2-pin)

JTAG/SWD speed: Default

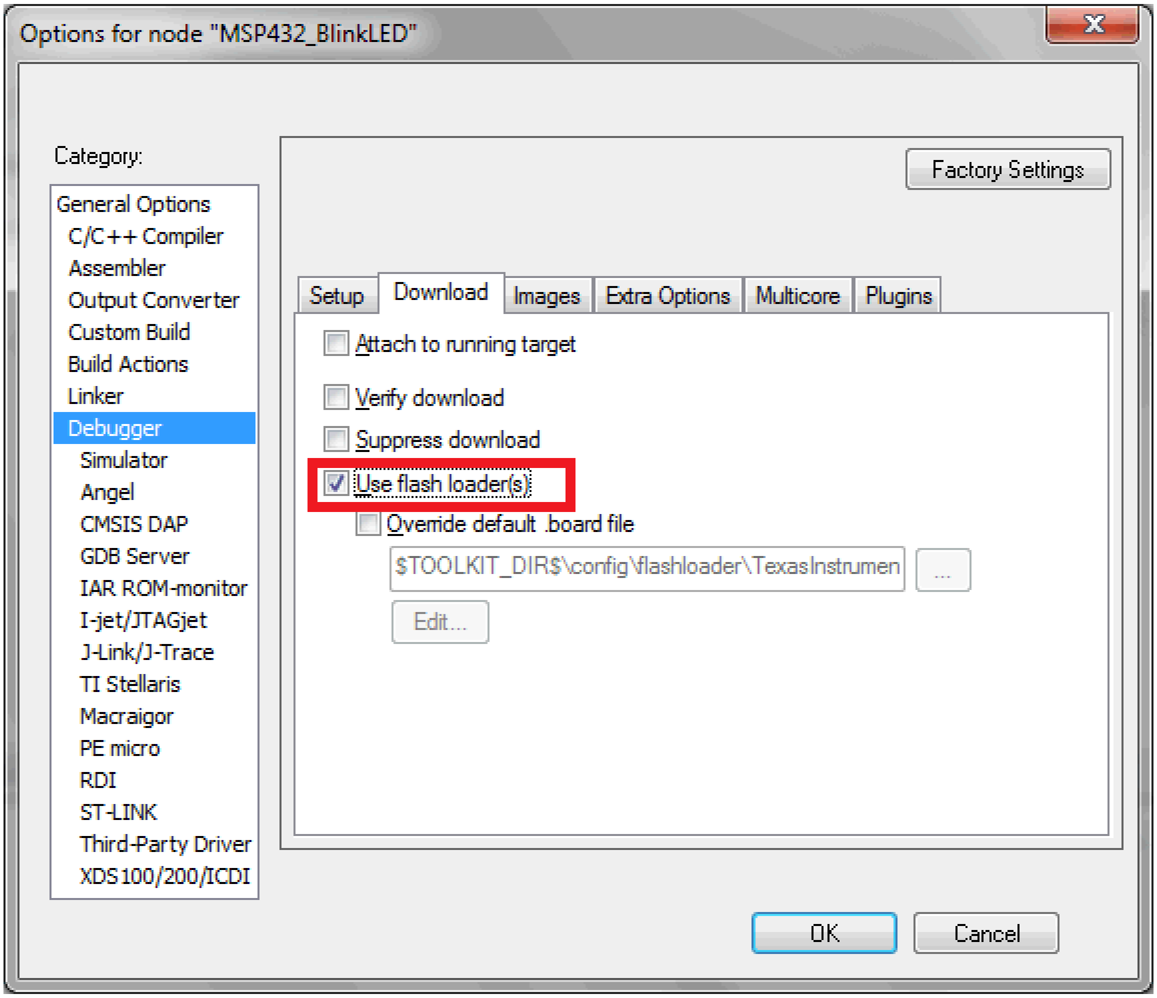
TI emulation package installation path

□ Override default

C:\ti\ccs_base

Figure 7. Selecting the TI XDS Debug Probe

Make sure that the **Use flash loader(s)** option is selected in the Download tab (see Figure 8).



□ Attach to running target

□ Verify download

□ Suppress download

☑ Use flash loader(s)

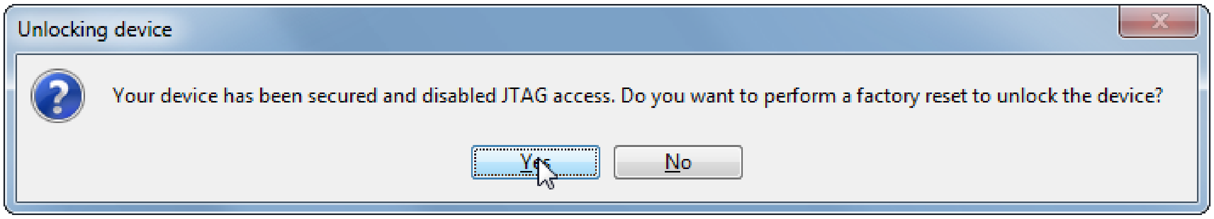
□ Override default .board file

$TOOLKIT_DIR$\config\flashloader\TexasInstrumen

Figure 8. Debug Download Options

4.1.1 Working With Device Security (MSP432P4xx Devices Only)

If you have disabled JTAG access on the device or are working on an application where you need to unlock a secure IP zone, IAR Embedded Workbench automatically runs a check on the device before downloading code. If IAR finds that the device has been secured, a dialog box opens as shown in Figure 9.



Unlocking device

Your device has been secured and disabled JTAG access. Do you want to perform a factory reset to unlock the device?

Figure 9. Dialog Box Asking to Perform a Factory Reset

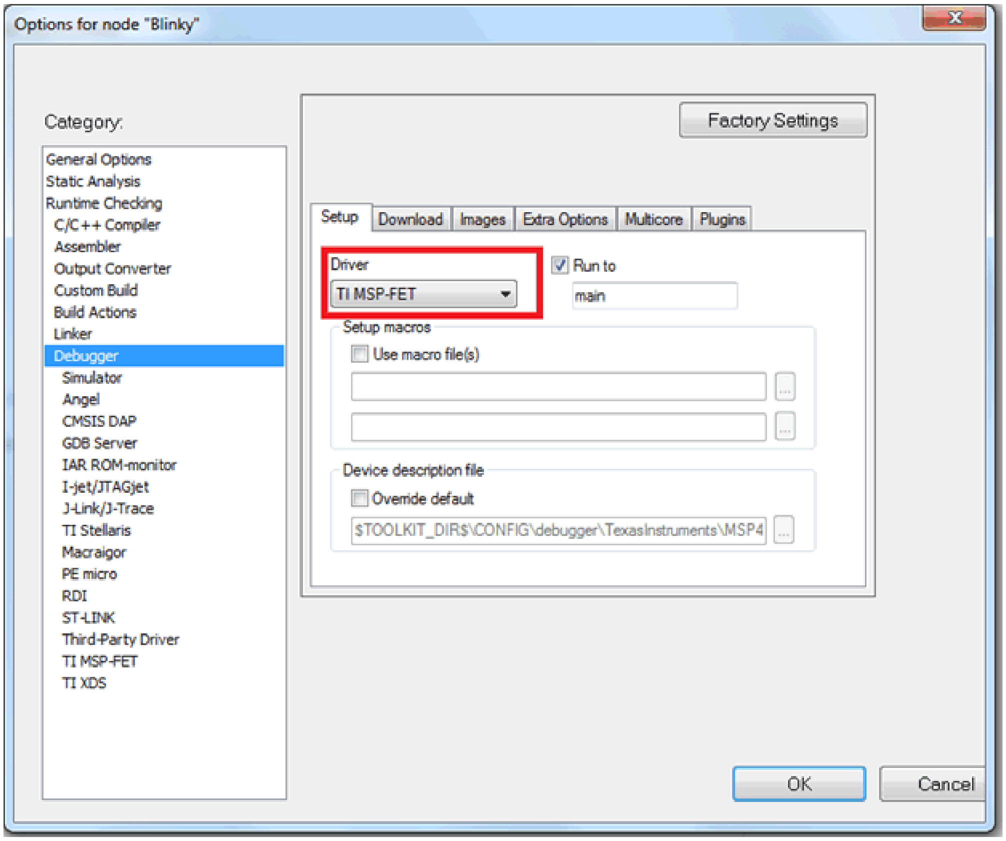
Click **Yes** to perform a factory reset and unlock the device so that code can be downloaded. Click **No** to end the debug session and leave the device locked. After the code has been downloaded, the debug session starts.

4.2 Using TI MSP-FET (MSP432P4xx Devices Only)

To use the MSP-FET for debugging MSP432x devices, IAR 7.60 or higher is required. No additional driver installation is required.

TI MSP-FET debug probes can be selected in the **Project Options** menu. Right click the active project, then select **Options** (see Figure 6).

Click **Debugger** in the left pane, and select**TI MSP-FET** (see Figure 10).



Options for node "Blinky"

Category: Debugger

Driver: TI MSP-FET

☑ Run to: main

Setup macros

□ Use macro file(s)

Device description file

□ Override default

$TOOLKIT_DIR$\CONFIG\debugger\TexasInstruments\MSP4

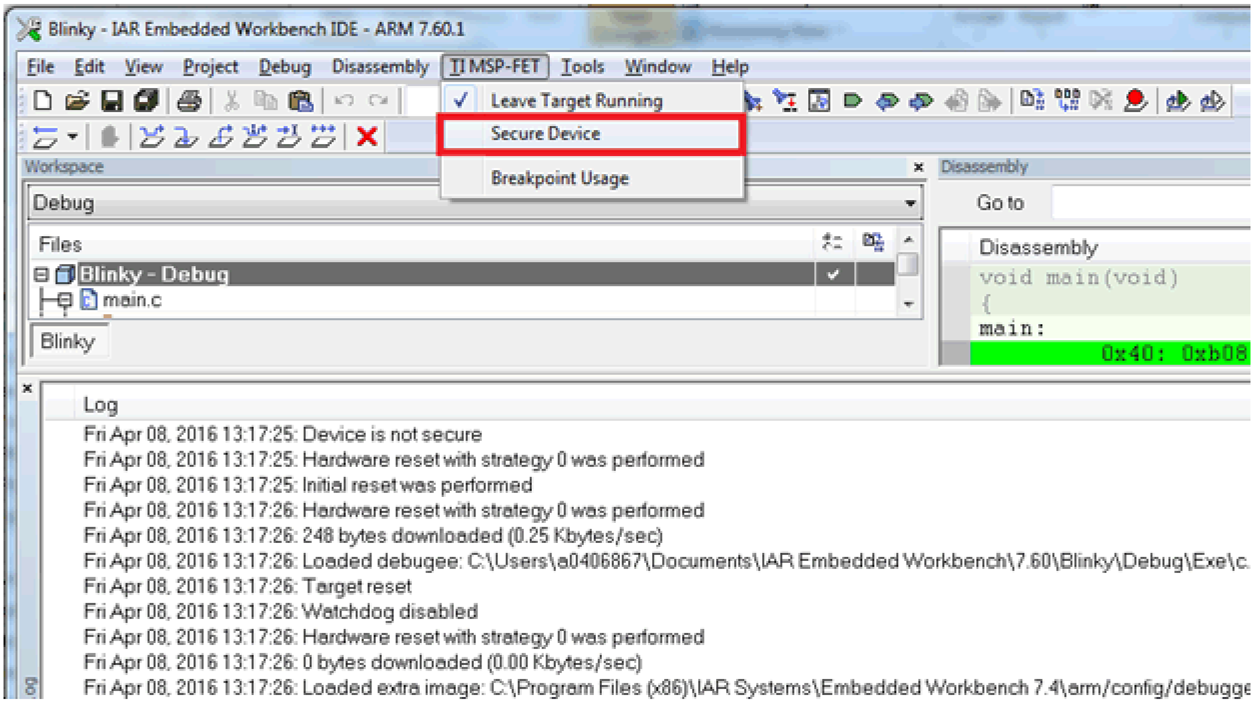
Figure 10. Selecting MSP-FET as Debug Probe

4.2.1 Working With Device Security (MSP432P4xx Devices Only)

Similar to the process when using TI XDS debug probes, you can unlock a protected device. If you have disabled JTAG access on the device or are working on an application where you need to unlock a secure IP zone, IAR Embedded Workbench automatically runs a check on the device before downloading code. If IAR finds that the device has been secured, a dialog box opens as shown in Figure 9.

Click **Yes** to perform a factory reset and unlock the device so that code can be downloaded. Click **No** to end the debug session and leave the device locked. After the code has been downloaded, the debug session starts.

The easy-to-use TI MSP-FET offers to secure the device by disabling JTAG access. Click **Secure Device** in the **TI MSP-FET** menu when a debugging session is running (see Figure 11).



```
void main(void)
{
main:
    0x40: 0xb08
```

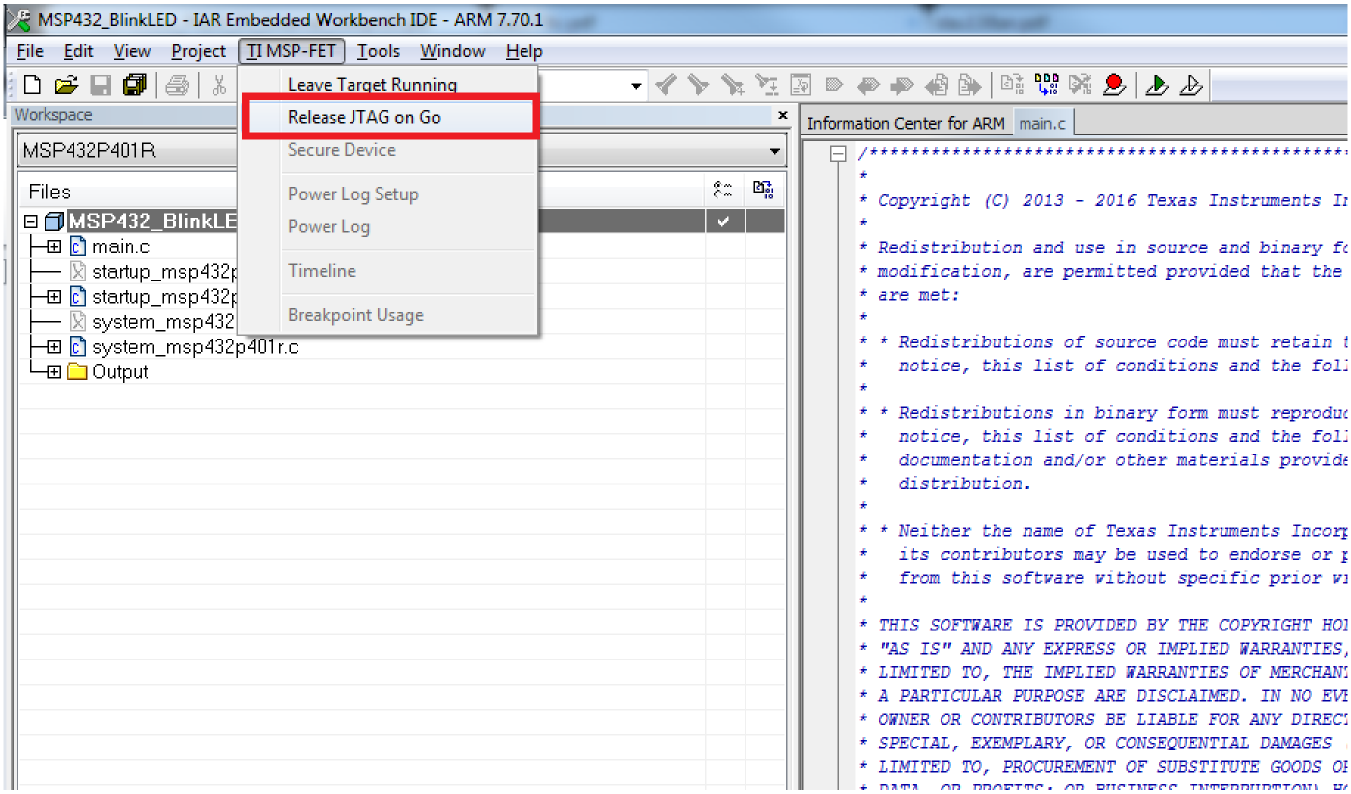
```
Fri Apr 08, 2016 13:17:25: Device is not secure
Fri Apr 08, 2016 13:17:25: Hardware reset with strategy 0 was performed
Fri Apr 08, 2016 13:17:25: Initial reset was performed
Fri Apr 08, 2016 13:17:26: Hardware reset with strategy 0 was performed
Fri Apr 08, 2016 13:17:26: 248 bytes downloaded (0.25 Kbytes/sec)
Fri Apr 08, 2016 13:17:26: Loaded debugee: C:\Users\a0406867\Documents\IAR Embedded Workbench\7.60\Blinky\Debug\Exe\c
Fri Apr 08, 2016 13:17:26: Target reset
Fri Apr 08, 2016 13:17:26: Watchdog disabled
Fri Apr 08, 2016 13:17:26: Hardware reset with strategy 0 was performed
Fri Apr 08, 2016 13:17:26: 0 bytes downloaded (0.00 Kbytes/sec)
Fri Apr 08, 2016 13:17:26: Loaded extra image: C:\Program Files (x86)\IAR Systems\Embedded Workbench 7.4\arm/config/debugge
```

Figure 11. Securing Device With MSP-FET

If only parts of the device are IP protected, MSP-FET displays a console message during connect: "IP protection is enabled on the device. Not all flash memory locations may be readable or writable". See the online documentation on www.ti.com/msp432 for details and tools for handling device security.

4.2.2 Release JTAG on Go Option

When debugging low-power modes the **Release JTAG on Go** option for the MSP-FET (see Figure 12) should be used to make sure that the debugger does not lose communication with the device. However, with this option selected, the debugger cannot identify breakpoints.



```
/*****************************************************
 *
 * Copyright (C) 2013 - 2016 Texas Instruments I
 *
 * Redistribution and use in source and binary f
 * modification, are permitted provided that the
 * are met:
 *
 *  * Redistributions of source code must retain t
 *    notice, this list of conditions and the fol
 *
 *  * Redistributions in binary form must reproduc
 *    notice, this list of conditions and the fol
 *    documentation and/or other materials provide
 *    distribution.
 *
 *  * Neither the name of Texas Instruments Incor
 *    its contributors may be used to endorse or p
 *    from this software without specific prior w
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HO
 * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHAN
 * A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EV
 * OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIREC
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS O
```

Figure 12. MSP-FET Release JTAG on Go

After enabling the **Release JTAG on Go** option, execute the **RUN** command.

4.2.3 Further Advice for MSP-FET in IAR EWARM

If the V_{CC} voltage is not high enough when trying to erase or write flash memory, the following message is displayed in the console:

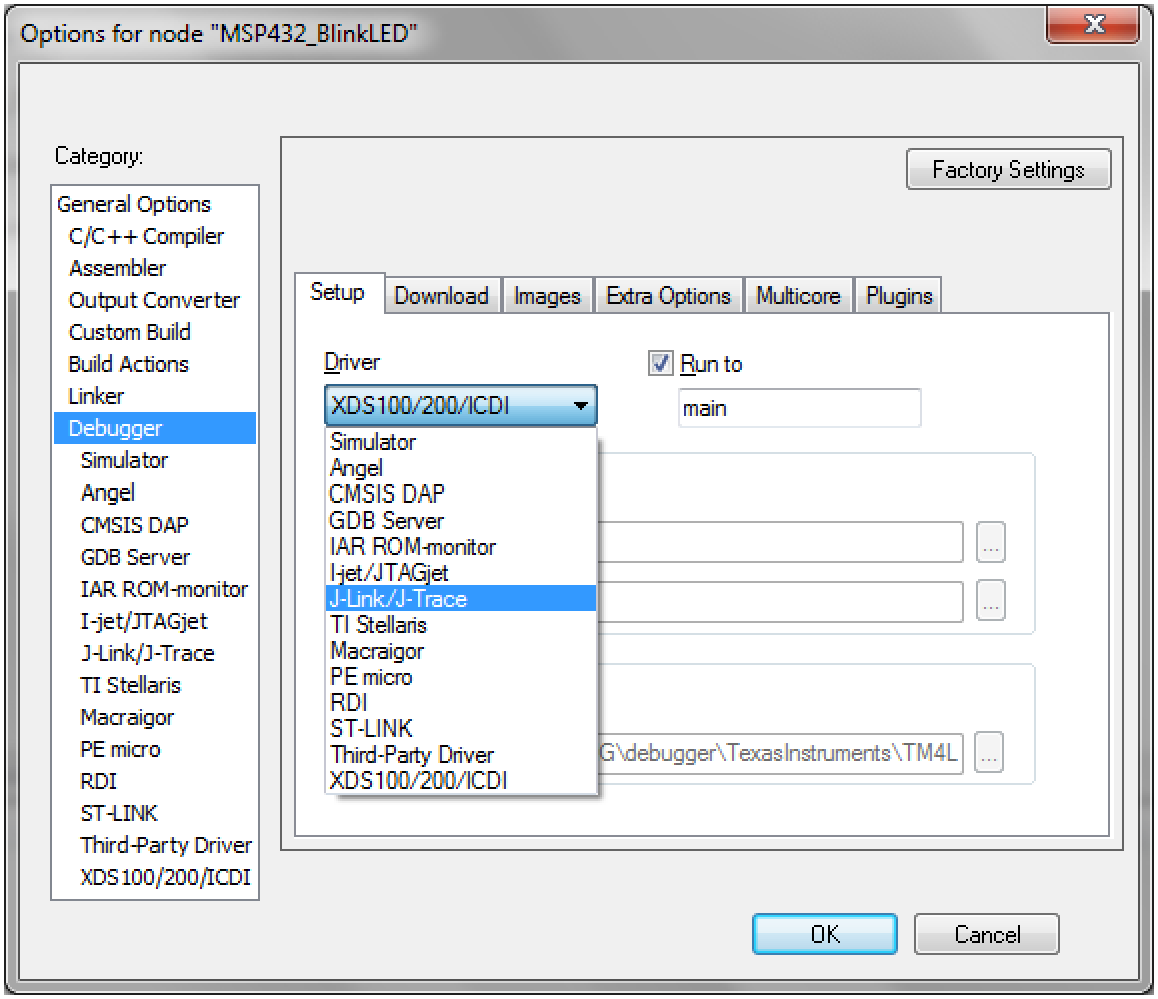
"Target device supply voltage is too low for Flash erase/programming".

Raise the supply voltage to correct this error.

Do not connect through a USB hub when performing a firmware update on the MSP-FET, the MSP-FET430UIF, or a LaunchPad™ development kit.

4.3 Using Segger J-Link Debug Probe

To use the Segger J-Link debug probe, right click the active project, then select **Options**. From the pulldown menu select **J-Link/J-Trace** (see Figure 13).



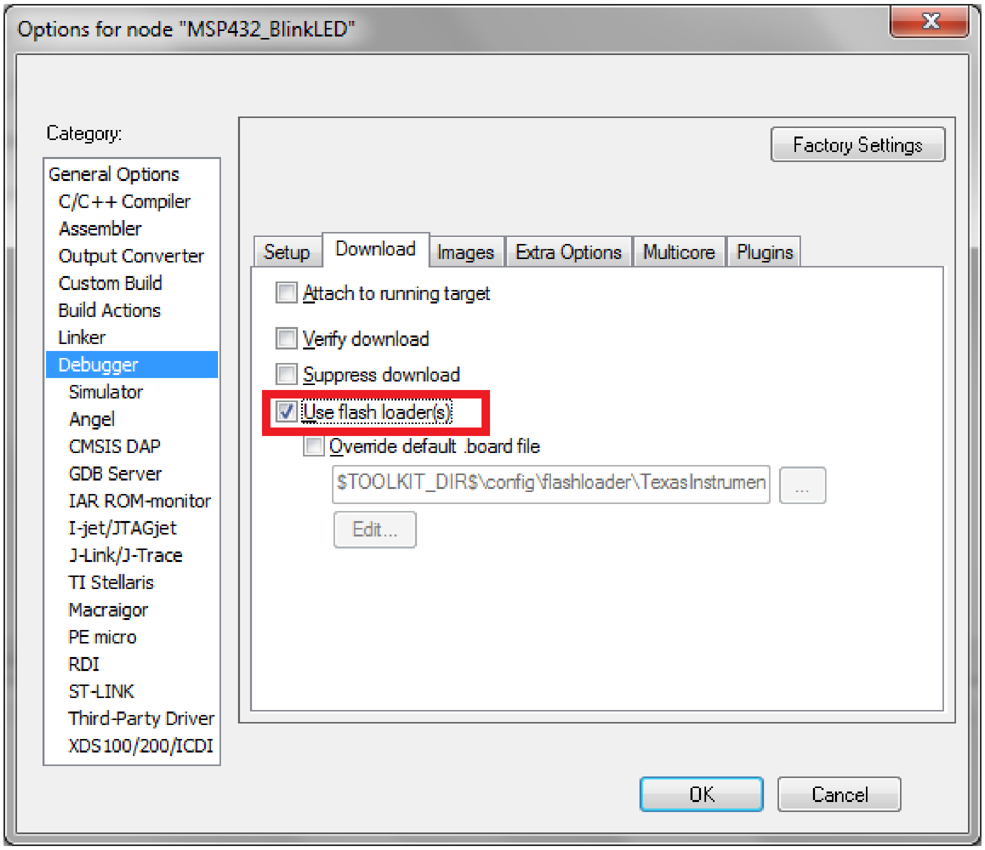
Options for node "MSP432_BlinkLED"

Driver: XDS100/200/ICDI

☑ Run to: main

Figure 13. Select the Segger J-Link Debug Probe

Next, switch to the Download pane and enable the **Use flash loader(s)** option (see Figure 14).



Options for node "MSP432_BlinkLED"

Category: Debugger

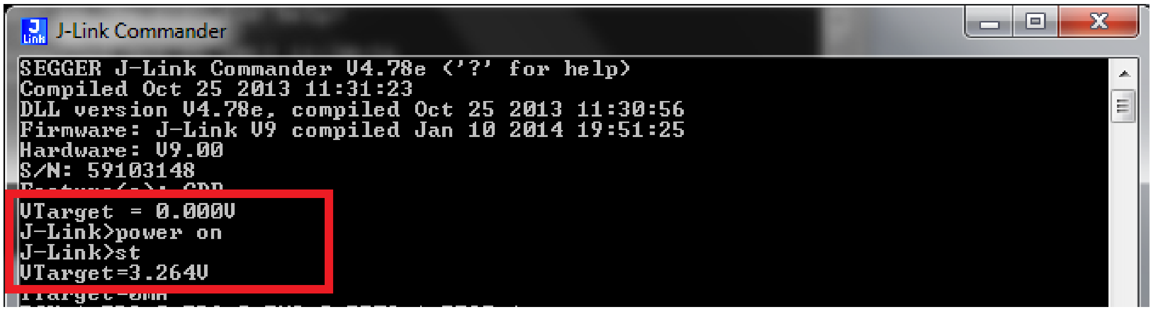
Download

- ☐ Attach to running target
- ☐ Verify download
- ☐ Suppress download
- ☑ Use flash loader(s)
  - ☐ Override default .board file
  - $TOOLKIT_DIR$\config\flashloader\TexasInstrumen

Figure 14. Enabling Usage of Flash Loaders

When using a target socket board for the MSP432 MCUs, you can benefit from the 5-V voltage output the J-Link provides on pin 19 of its Cortex-M debug connector. This option needs to be enabled through the J-Link Commander, a console application available from Segger. When enabled, the debug probe provide 5 V to the target system. See the *MSP432™ SimpleLink™ Microcontrollers Hardware Tools User's Guide* how to configure the target socket board to use the 5-V power supply to generate 3.3-V device voltage and to Segger's documentation how to enable the voltage output.

Figure 15 shows the effect of the **power on** command, when applied in the J-Link Commander. Before executing the command, the measured target voltage is 0 V, and right after applying target power, 3.3 V is available as target voltage.



J-Link Commander

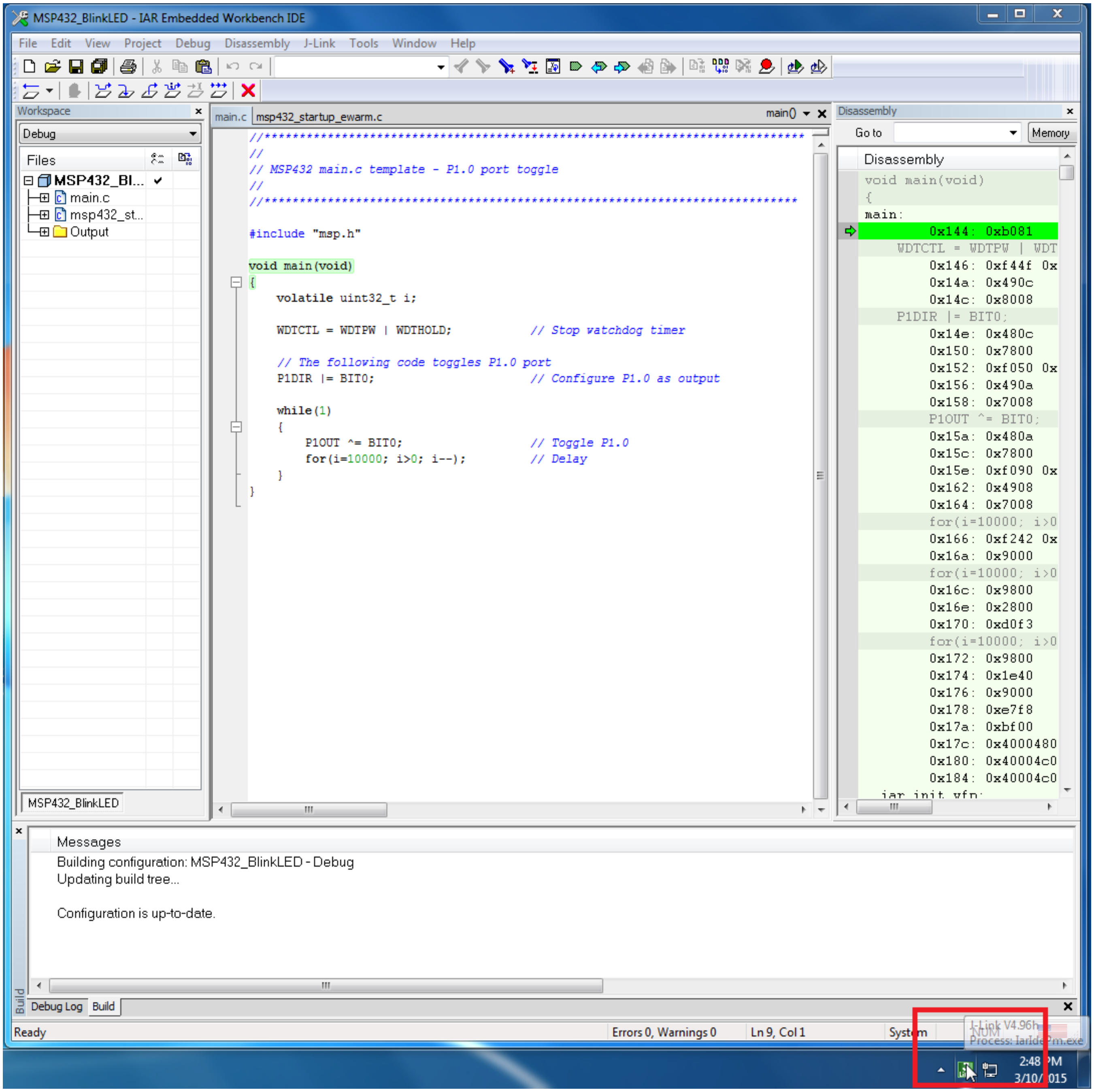
```
SEGGER J-Link Commander V4.78e ('?' for help)
Compiled Oct 25 2013 11:31:23
DLL version V4.78e, compiled Oct 25 2013 11:30:56
Firmware: J-Link V9 compiled Jan 10 2014 19:51:25
Hardware: V9.00
S/N: 59103148
Feature(s): GDB
VTarget = 0.000V
J-Link>power on
J-Link>st
VTarget=3.264V
ITarget=0mA
```

Figure 15. Using J-Link Commander to Enable Power Output to Target System

Now you can download the program and debug using the Segger J-Link debug probe with EWARM.

4.3.1 Working With Device Security (MSP432P4xx Devices Only)

If you have disabled JTAG access on the device or are working on an application where you need to unlock a secure IP zone, a J-Link Script needs to be added to the debug session to enable a factory reset. During a debug session, launch the J-Link control panel by clicking on the J-Link icon in the status bar (see Figure 16).



```
//*****************************************************************************
//
// MSP432 main.c template - P1.0 port toggle
//
//****************************************************************************

#include "msp.h"

void main(void)
{
    volatile uint32_t i;

    WDTCTL = WDTPW | WDTHOLD;           // Stop watchdog timer

    // The following code toggles P1.0 port
    P1DIR |= BIT0;                      // Configure P1.0 as output

    while(1)
    {
        P1OUT ^= BIT0;                  // Toggle P1.0
        for(i=10000; i>0; i--);         // Delay
    }
}
```

```
Disassembly
void main(void)
{
main:
        0x144: 0xb081
    WDTCTL = WDTPW | WDT
        0x146: 0xf44f 0x
        0x14a: 0x490c
        0x14c: 0x8008
    P1DIR |= BIT0;
        0x14e: 0x480c
        0x150: 0x7800
        0x152: 0xf050 0x
        0x156: 0x490a
        0x158: 0x7008
        P1OUT ^= BIT0;
        0x15a: 0x480a
        0x15c: 0x7800
        0x15e: 0xf090 0x
        0x162: 0x4908
        0x164: 0x7008
        for(i=10000; i>0
        0x166: 0xf242 0x
        0x16a: 0x9000
        for(i=10000; i>0
        0x16c: 0x9800
        0x16e: 0x2800
        0x170: 0xd0f3
        for(i=10000; i>0
        0x172: 0x9800
        0x174: 0x1e40
        0x176: 0x9000
        0x178: 0xe7f8
        0x17a: 0xbf00
        0x17c: 0x4000480
        0x180: 0x40004c0
        0x184: 0x40004c0
  iar_init_vfp:
```

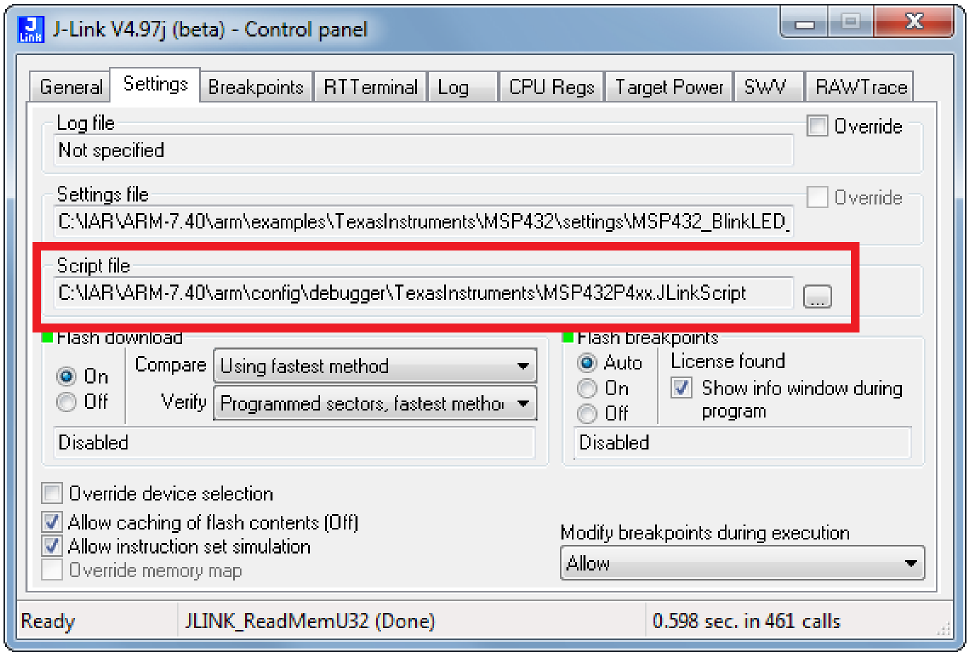
```
Messages
Building configuration: MSP432_BlinkLED - Debug
Updating build tree...

Configuration is up-to-date.
```

Ready | Errors 0, Warnings 0 | Ln 9, Col 1 | System | NUM

Figure 16. Launch J-Link Control Panel

After launching the J-Link control panel, add the MSP432 J-Link script provided in IAR. Figure 17 shows the location of the script file.



J-Link V4.97j (beta) - Control panel

Log file: Not specified □ Override

Settings file: C:\IAR\ARM-7.40\arm\examples\TexasInstruments\MSP432\settings\MSP432_BlinkLED_ □ Override

Script file: C:\IAR\ARM-7.40\arm\config\debugger\TexasInstruments\MSP432P4xx.JLinkScript

Flash download: ● On ○ Off; Compare: Using fastest method; Verify: Programmed sectors, fastest metho; Disabled

Flash breakpoints: ● Auto ○ On ○ Off; License found; ☑ Show info window during program; Disabled

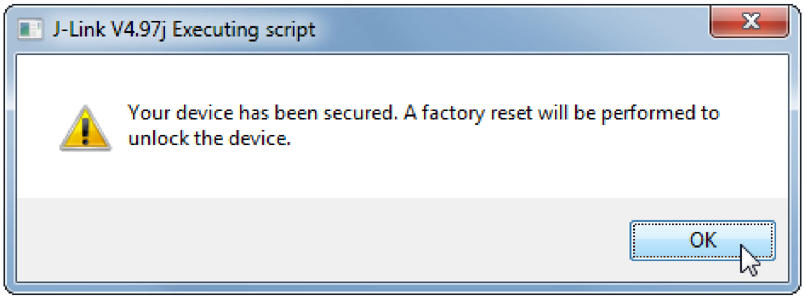
□ Override device selection
☑ Allow caching of flash contents (Off)
☑ Allow instruction set simulation
□ Override memory map

Modify breakpoints during execution: Allow

Ready | JLINK_ReadMemU32 (Done) | 0.598 sec. in 461 calls

Figure 17. J-Link Control Panel

The J-Link script now runs at the launch of very debug session and every time code is downloaded to the device. If the device has been secured when trying to download code, a dialog box reports that the device is secured and can be erased (see Figure 18).

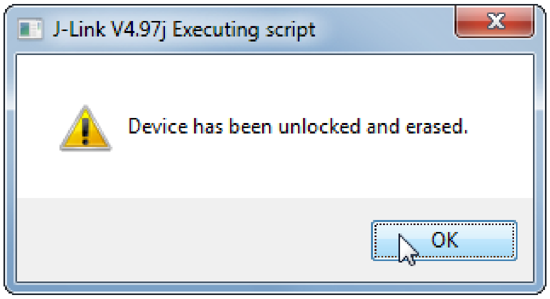


J-Link V4.97j Executing script

Your device has been secured. A factory reset will be performed to unlock the device.

Figure 18. J-Link Script Detecting That the Device Has Been Secured

Click **OK** on the dialog box to issue a factory reset, which erases any code present on the device and then starts to download the compiled code. When the factory reset is a complete, a confirmation dialog box reports that the process is complete (see Figure 19).



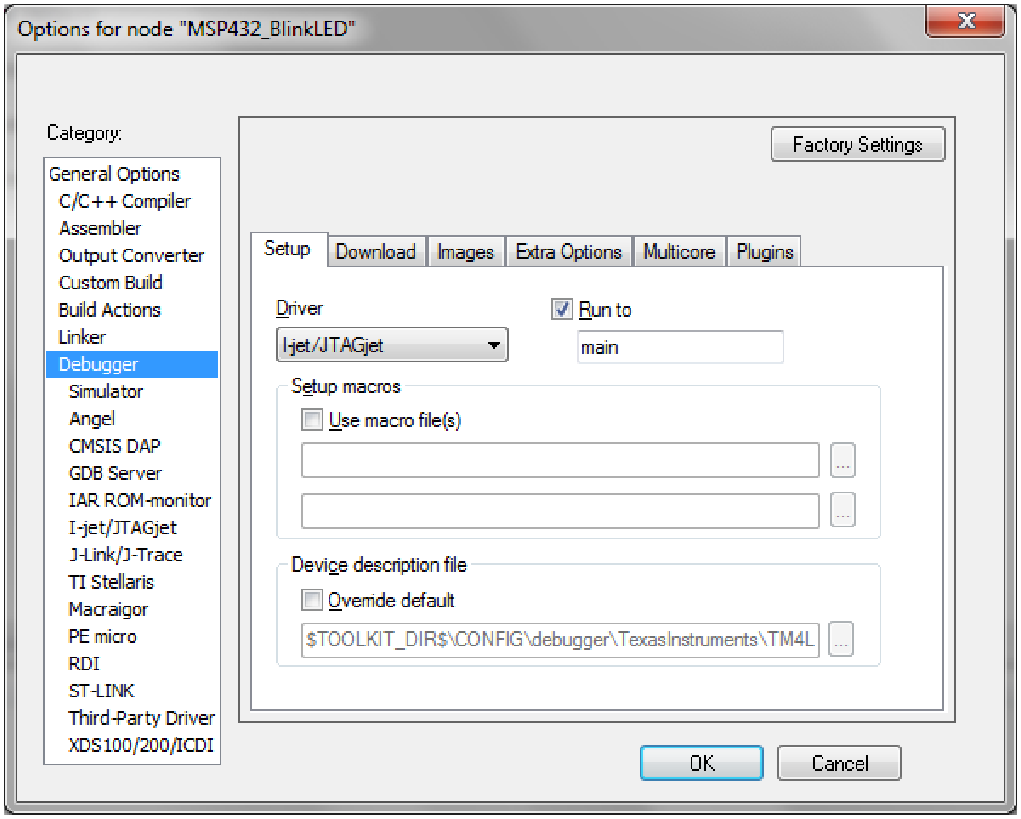
J-Link V4.97j Executing script

Device has been unlocked and erased.

Figure 19. Device Has Been Unlocked and Erased

4.4 Using IAR I-jet Debug Probe

To use the IAR I-jet debug probe, right click the active project, then click **Options**. From the pulldown **Driver** menu, select **I-jet / JTAGjet** (see Figure 20).



Options for node "MSP432_BlinkLED"

Driver: I-jet/JTAGjet

☑ Run to: main

Setup macros

☐ Use macro file(s)

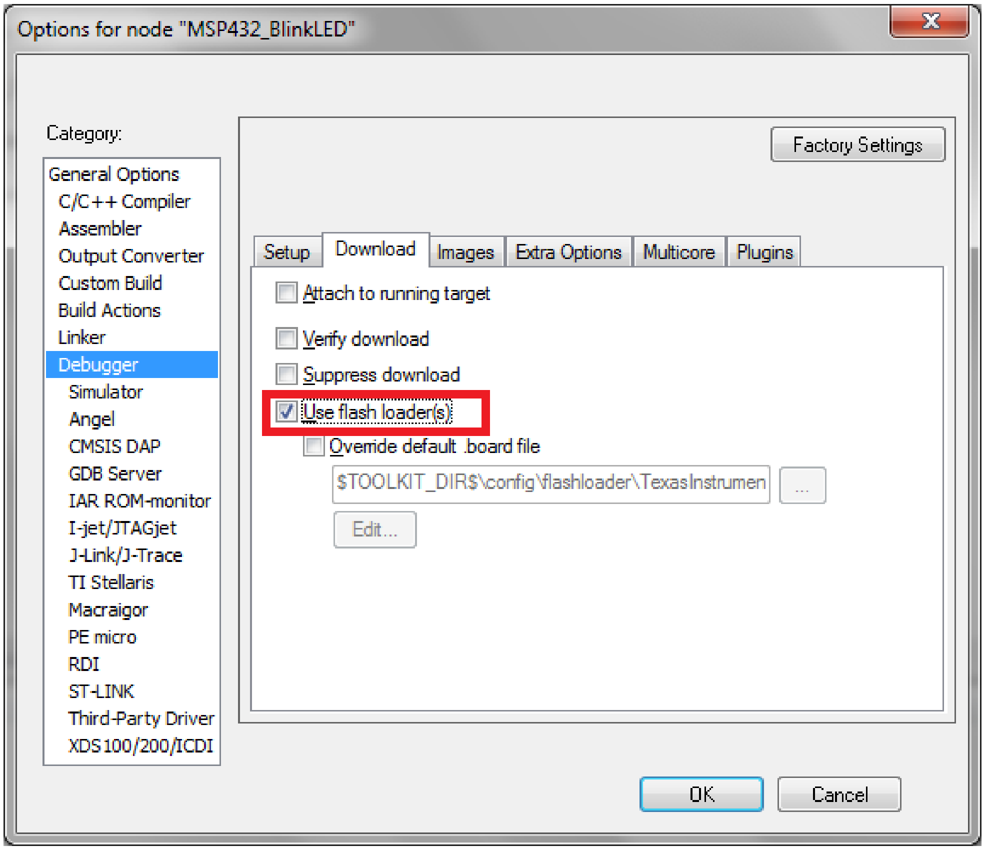
Device description file

☐ Override default

$TOOLKIT_DIR$\CONFIG\debugger\TexasInstruments\TM4L

Figure 20. Selecting the IAR I-jet Debug Probe

Next, switch to the Download pane and enable the **Use flash loader(s)** option (see Figure 21).



□ Attach to running target

□ Verify download

□ Suppress download

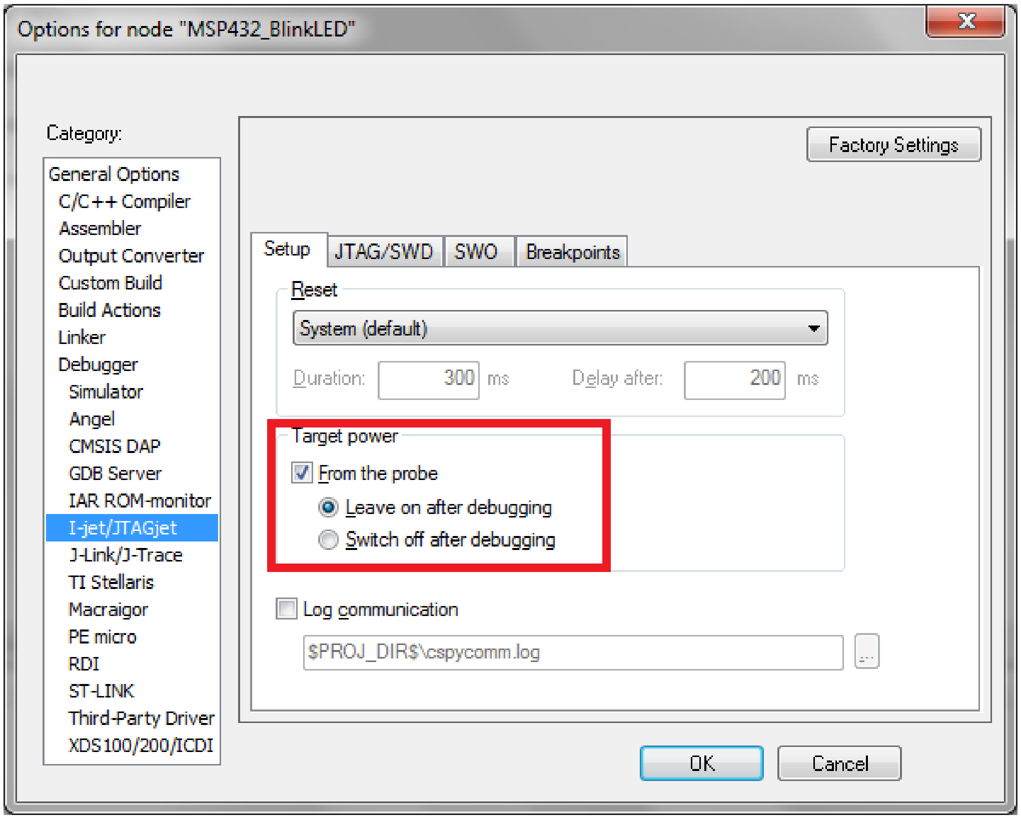
☑ Use flash loader(s)

□ Override default .board file

$TOOLKIT_DIR$\config\flashloader\TexasInstrumen

Figure 21. Enabling Use of Flash Loaders

When using a target socket board for the MSP432 MCUs, you can benefit from the 5-V voltage output the I-jet provides on pin 19 of its Cortex-M debug connector. Go to the I-jet/JTAGjet specific menu and enable the **Target Power** option (see Figure 22). See the *MSP432™ SimpleLink™ Microcontrollers Hardware Tools User's Guide* how to configure the target socket board to use the 5-V power supply to generate 3.3-V device voltage.



Options for node "MSP432_BlinkLED"

Category:

- General Options
  - C/C++ Compiler
  - Assembler
  - Output Converter
  - Custom Build
  - Build Actions
  - Linker
  - Debugger
    - Simulator
    - Angel
    - CMSIS DAP
    - GDB Server
    - IAR ROM-monitor
    - I-jet/JTAGjet
    - J-Link/J-Trace
    - TI Stellaris
    - Macraigor
    - PE micro
    - RDI
    - ST-LINK
    - Third-Party Driver
    - XDS100/200/ICDI

Reset: System (default)

Duration: 300 ms Delay after: 200 ms

Target power

☑ From the probe

● Leave on after debugging

○ Switch off after debugging

□ Log communication

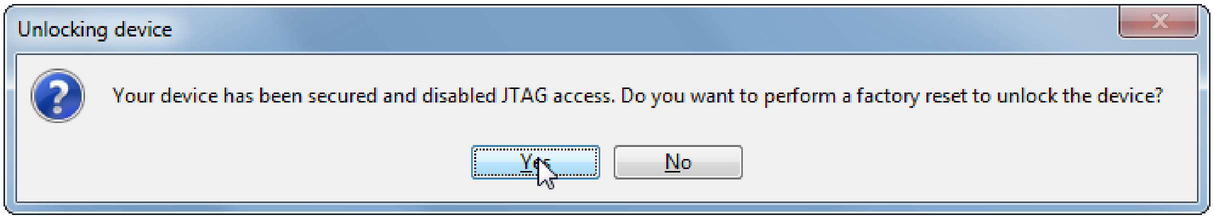
$PROJ_DIR$\cspycomm.log

Figure 22. Debug Probe Setup

Now you can download the program and debug using the IAR I-jet debug probe with EWARM.

4.4.1 Working With Device Security (MSP432P4xx Devices Only)

If you have disabled JTAG access on the device or are working on an application where you need to unlock a secure IP zone, IAR Embedded Workbench automatically runs a check on the device before downloading code if you are using an IAR I-Jet. If IAR finds that the device has been secured, a dialog box opens as shown in Figure 23.



Unlocking device

Your device has been secured and disabled JTAG access. Do you want to perform a factory reset to unlock the device?

Figure 23. Dialog Box Asking to Perform a Factory Reset

Click **Yes** to perform a factory reset and unlock the device so that code can be downloaded. Click **No** to end the debug session and leave the device locked. After the code has been downloaded, the debug session starts.

4.5 Debugging Driver Lib in ROM

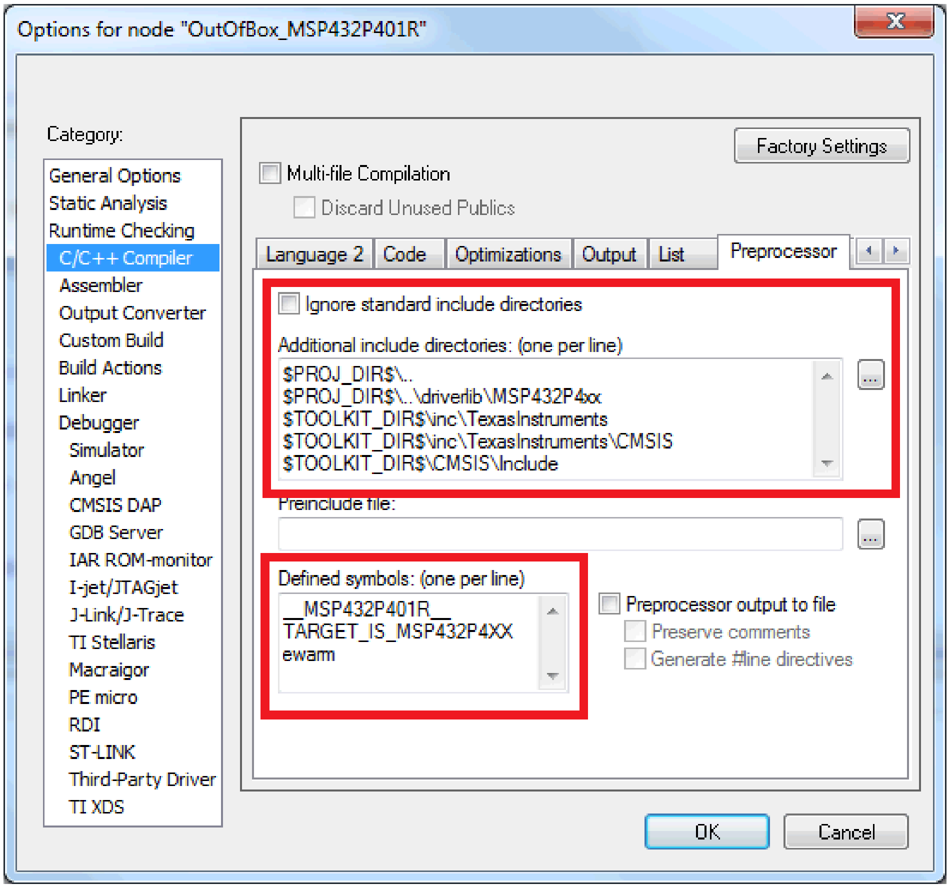
The MSP432P4xx family includes a complete peripheral driver library (DriverLib) that is fully integrated into the ROM memory. Developers can leverage the ROM DriverLib for multiple benefits including access to highly robust and tested APIs, single-cycle ROM execution speed at lower power consumption, and freeing up memory space for additional application code. Developers can gain access to ROM APIs by adding DriverLib header file to projects and linking to a prebuilt library.

For more information on MSP432P4xx Driver Library and what is provided in ROM DriverLib, see the MSP432P4xx Driver Library, which is available in the SimpleLink MSP432 SDK.

4.5.1 Enable Use of Software in ROM in MSP432 Project (MSP432P4xx Devices Only)

If the path to the driver library headers and sources is not yet included in the project, add it. Click **Project Options** (Alt+F7), then select **C/C++ Compiler**, and select the **Preproccessor** tab. Add the include path to the MSP432P4xx driver library source folder in the **Additional include directories** field (see Figure 24). For example, add C:\ti\MSP432_DriverLib_2_20_00_08\driverlib\MSP432P4xx, assuming that the driver library was extracted to C:\ti\MSP432_DriverLib_2_20_00_08.

Then in the **Defined Symbols** field, add TARGET_IS_MSP432P4XX to enable the Software in ROM use. (see Figure 24).



Options for node "OutOfBox_MSP432P401R"

□ Multi-file Compilation
□ Discard Unused Publics

□ Ignore standard include directories

Additional include directories: (one per line)

```
$PROJ_DIR$\..
$PROJ_DIR$\..\driverlib\MSP432P4xx
$TOOLKIT_DIR$\inc\TexasInstruments
$TOOLKIT_DIR$\inc\TexasInstruments\CMSIS
$TOOLKIT_DIR$\CMSIS\Include
```

Preinclude file:

Defined symbols: (one per line)

```
__MSP432P401R__
TARGET_IS_MSP432P4XX
ewarm
```

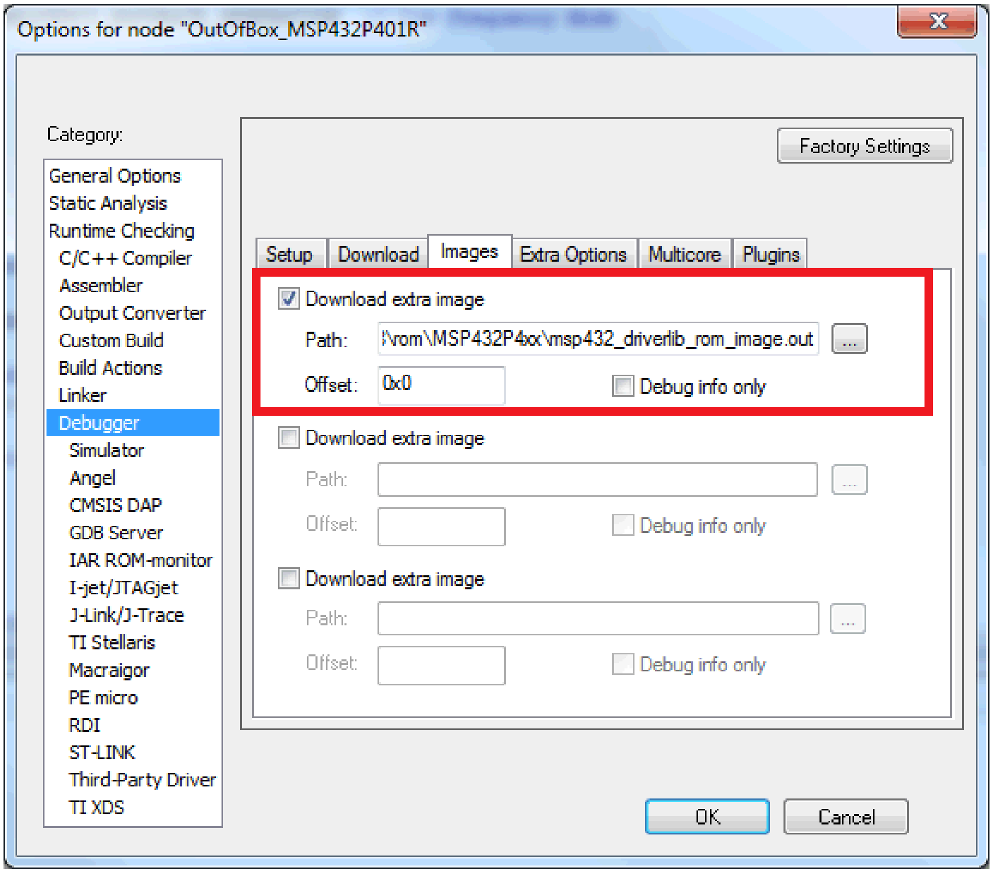
□ Preprocessor output to file
□ Preserve comments
□ Generate #line directives

Figure 24. Adding Driver Library to a Project

4.5.2 Load ROM Symbol Into Debugger

To enable debug of the software in ROM, the corresponding symbol should be loaded. Click **Project Options** (ALT+F7) and select **Debugger**. Then select the **Images** tab and proceed as follows (see Figure 25):

1. Check the box **Download extra image**
2. In the **Path** field, click the browse button and locate the ROM debugging symbols (*.out). The MSP432P4xx Driver Library provides this file msp432_driverlib_rom_image.out available in C:\ti\MSP432_DriverLib_2_20_00_08\rom\MSP432P4xx\msp432_driverlib_rom_image.out. It is important to keep the driverlib.c file on the same path as the symbol file in the same directory.
3. In the **Offset** field, type 0x0.
4. Click **OK** and start debugging.



Options for node "OutOfBox_MSP432P401R"

☑ Download extra image

Path: \rom\MSP432P4xx\msp432_driverlib_rom_image.out

Offset: 0x0 ☐ Debug info only

☐ Download extra image

Path:

Offset: ☐ Debug info only

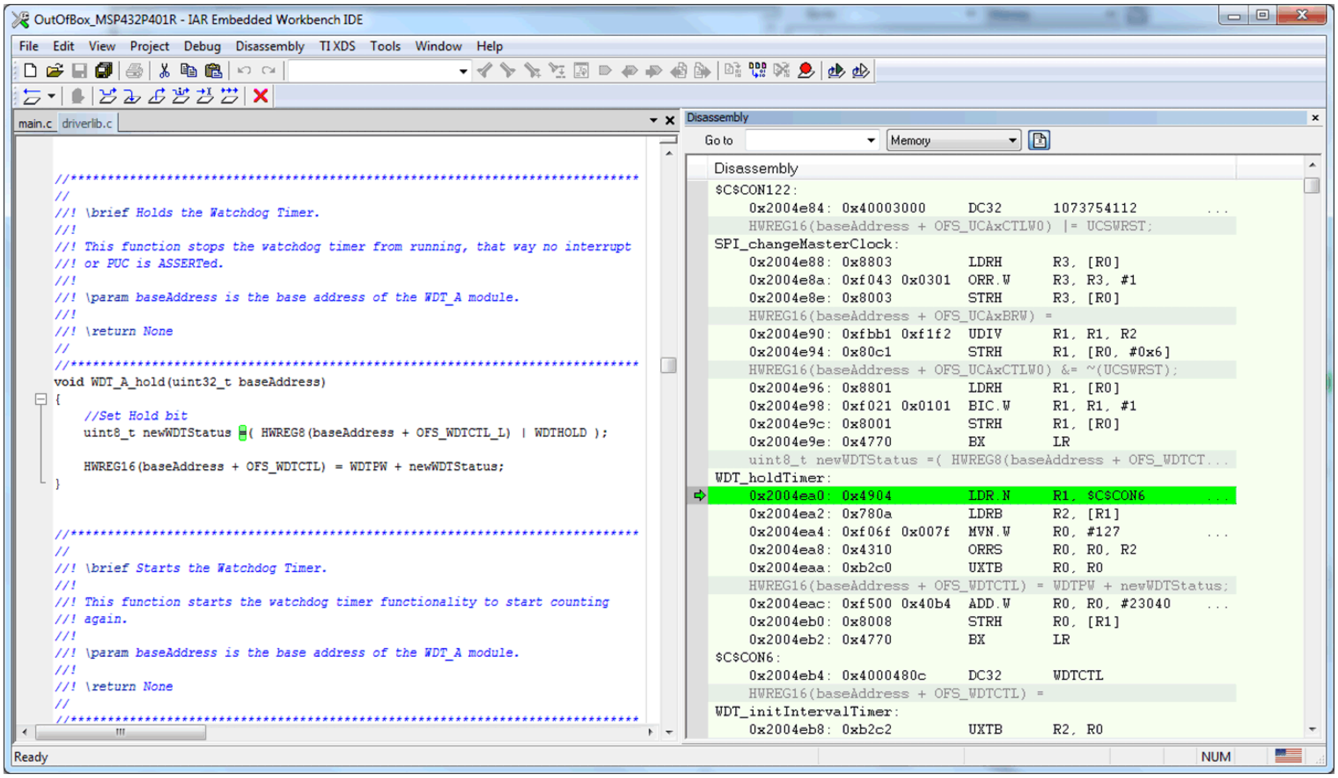
☐ Download extra image

Path:

Offset: ☐ Debug info only

Figure 25. Adding ROM Symbol to Debugger

When the next debug session is started, you can step through the ROM API. The IDE automatically opens the corresponding source file (see Figure 26). MSPWare also provides the software in ROM symbol available in C:\ti\MSPWare_2_20_00_19\driverlib\rom\MSP432P4xx.


OutOfBox_MSP432P401R - IAR Embedded Workbench IDE
File Edit View Project Debug Disassembly TI XDS Tools Window Help
main.c
driverlib.c
Disassembly
Go to
Memory
Ready
NUM

```
//*****************************************************************************
//
//! \brief Holds the Watchdog Timer.
//!
//! This function stops the watchdog timer from running, that way no interrupt
//! or PUC is ASSERTed.
//!
//! \param baseAddress is the base address of the WDT_A module.
//!
//! \return None
//
//*****************************************************************************
void WDT_A_hold(uint32_t baseAddress)
{
    //Set Hold bit
    uint8_t newWDTStatus =( HWREG8(baseAddress + OFS_WDTCTL_L) | WDTHOLD );

    HWREG16(baseAddress + OFS_WDTCTL) = WDTPW + newWDTStatus;
}


//*****************************************************************************
//
//! \brief Starts the Watchdog Timer.
//!
//! This function starts the watchdog timer functionality to start counting
//! again.
//!
//! \param baseAddress is the base address of the WDT_A module.
//!
//! \return None
//
//*****************************************************************************
```

```
Disassembly
$C$CON122:
    0x2004e84: 0x40003000     DC32      1073754112        ...
    HWREG16(baseAddress + OFS_UCAxCTLW0) |= UCSWRST;
SPI_changeMasterClock:
    0x2004e88: 0x8803         LDRH      R3, [R0]
    0x2004e8a: 0xf043 0x0301  ORR.W     R3, R3, #1
    0x2004e8e: 0x8003         STRH      R3, [R0]
    HWREG16(baseAddress + OFS_UCAxBRW) =
    0x2004e90: 0xfbb1 0xf1f2  UDIV      R1, R1, R2
    0x2004e94: 0x80c1         STRH      R1, [R0, #0x6]
    HWREG16(baseAddress + OFS_UCAxCTLW0) &= ~(UCSWRST);
    0x2004e96: 0x8801         LDRH      R1, [R0]
    0x2004e98: 0xf021 0x0101  BIC.W     R1, R1, #1
    0x2004e9c: 0x8001         STRH      R1, [R0]
    0x2004e9e: 0x4770         BX        LR
    uint8_t newWDTStatus =( HWREG8(baseAddress + OFS_WDTCT...
WDT_holdTimer:
⇨   0x2004ea0: 0x4904         LDR.N     R1, $C$CON6       ...
    0x2004ea2: 0x780a         LDRB      R2, [R1]
    0x2004ea4: 0xf06f 0x007f  MVN.W     R0, #127          ...
    0x2004ea8: 0x4310         ORRS      R0, R0, R2
    0x2004eaa: 0xb2c0         UXTB      R0, R0
    HWREG16(baseAddress + OFS_WDTCTL) = WDTPW + newWDTStatus;
    0x2004eac: 0xf500 0x40b4  ADD.W     R0, R0, #23040    ...
    0x2004eb0: 0x8008         STRH      R0, [R1]
    0x2004eb2: 0x4770         BX        LR
$C$CON6:
    0x2004eb4: 0x4000480c     DC32      WDTCTL
    HWREG16(baseAddress + OFS_WDTCTL) =
WDT_initIntervalTimer:
    0x2004eb8: 0xb2c2         UXTB      R2, R0
```

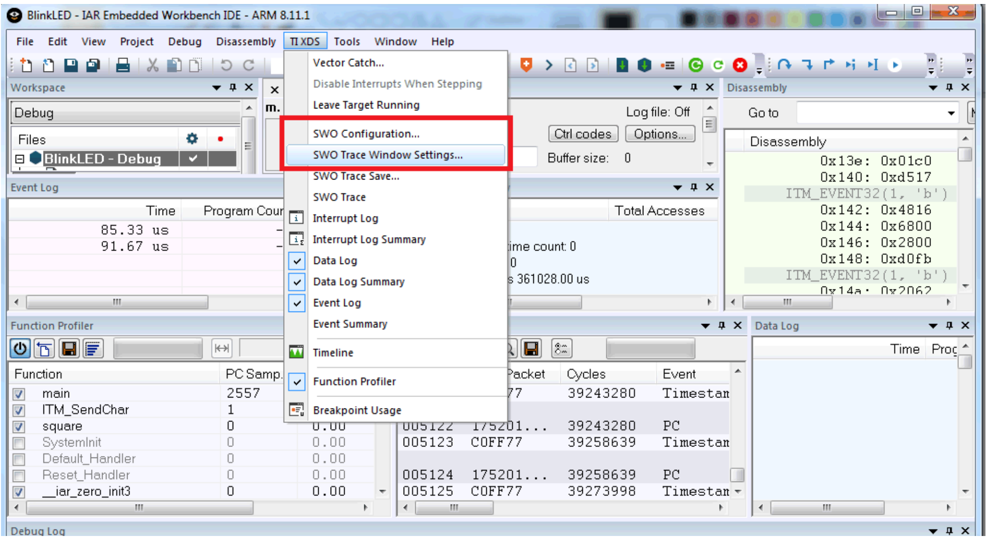
Figure 26. Stepping Through ROM DriverLib Source Code

5 Using Serial Wire Output (SWO) Hardware Trace Analyzer

The SWO Trace tools for MSP432 MCUs are implemented using the features of the Arm CoreSight™ components, especially the Instrumentation Trace Macrocell (ITM) and Data Watchpoint and Trace Unit (DWT) (ETM is not present in MSP432P4xx MCUs).

To use SWO trace, an SWO-enabled debug probe must be used. In IAR EWARM, this is currently possible with IAR I-JET, Segger J-Link, TI XDS200, and TI XDS110 debug probes. Change the interface of the debug probe to **SWD** and start a debug session. Trace itself is configured from a running debug session.

Adjust the trace settings to the project settings and frequencies, and start with a lightweight setting. For example, to enable SWO trace for a bare metal blinky example, use SWO trace settings as shown in the following figures. Figure 27 shows the menu to open the configuration dialogs.



Event Log

Time	Program Cour
85.33 us	–
91.67 us	–

ime count: 0
0
s 361028.00 us

Function Profiler

Function	PC Samp.	
☑ main	2557	
☑ ITM_SendChar	1	
☑ square	0	0.00
☐ SystemInit	0	0.00
☐ Default_Handler	0	0.00
☐ Reset_Handler	0	0.00
☑ __iar_zero_init3	0	0.00

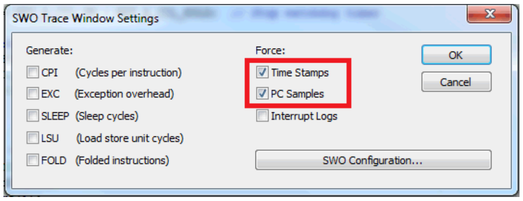
	Packet	Cycles	Event
	77	39243280	Timestam
005122	175201...	39243280	PC
005123	C0FF77	39258639	Timestam
005124	175201...	39258639	PC
005125	C0FF77	39273998	Timestam

Disassembly

```
        0x13e: 0x01c0
        0x140: 0xd517
    ITM_EVENT32(1, 'b')
        0x142: 0x4816
        0x144: 0x6800
        0x146: 0x2800
        0x148: 0xd0fb
    ITM_EVENT32(1, 'b')
        0x14a: 0x2062
```

Figure 27. Trace Settings Configured From Running Debug Session Using Debugger Menu

Force **Time Stamps** and **PC Samples** in the SWO Trace Window Settings (see Figure 28).



Generate:

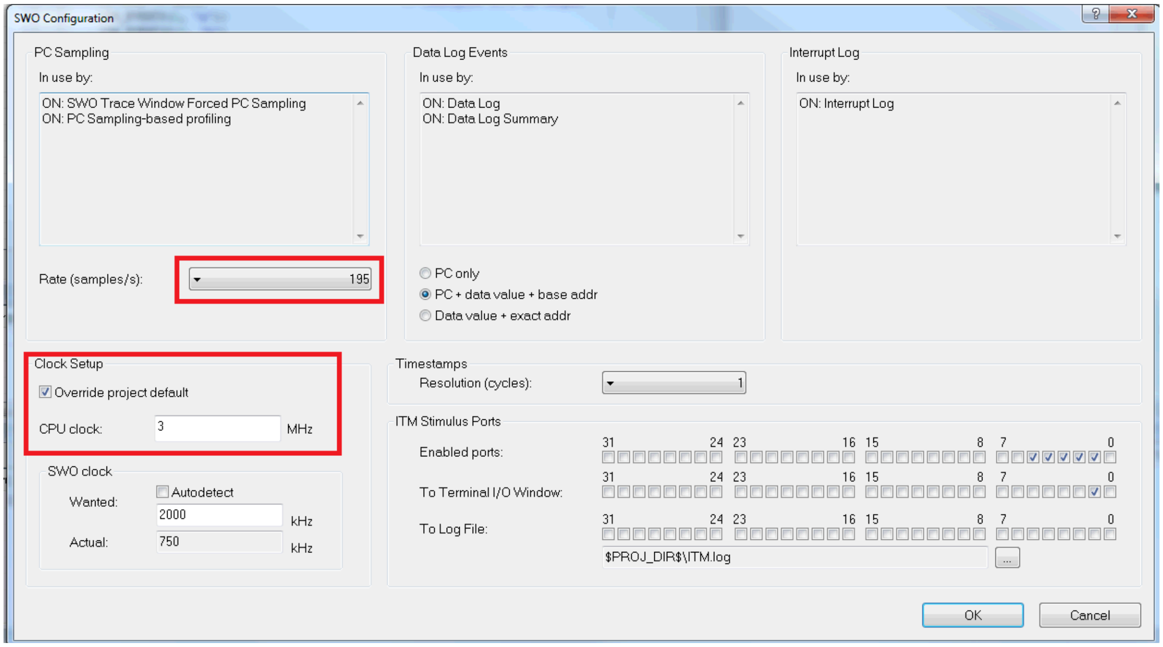
☐ CPI (Cycles per instruction)
☐ EXC (Exception overhead)
☐ SLEEP (Sleep cycles)
☐ LSU (Load store unit cycles)
☐ FOLD (Folded instructions)

Force:

☑ Time Stamps
☑ PC Samples
☐ Interrupt Logs

Figure 28. SWO Trace Window Settings

Override **Clock Setup** and set the sample rate to a low value in SWO Configuration (see Figure 29).



PC Sampling

In use by:

ON: SWO Trace Window Forced PC Sampling
ON: PC Sampling-based profiling

Rate (samples/s): 195

Data Log Events

In use by:

ON: Data Log
ON: Data Log Summary

○ PC only
● PC + data value + base addr
○ Data value + exact addr

Interrupt Log

In use by:

ON: Interrupt Log

Clock Setup

☑ Override project default

CPU clock: 3 MHz

SWO clock

Wanted: ☐ Autodetect 2000 kHz

Actual: 750 kHz

Timestamps

Resolution (cycles): 1

ITM Stimulus Ports

	31 – 24	23 – 16	15 – 8	7 – 0
Enabled ports:	☐☐☐☐☐☐☐☐	☐☐☐☐☐☐☐☐	☐☐☐☐☐☐☐☐	☐☐☑☑☑☑☑☐
To Terminal I/O Window:	☐☐☐☐☐☐☐☐	☐☐☐☐☐☐☐☐	☐☐☐☐☐☐☐☐	☐☐☐☐☐☐☑☐
To Log File:	☐☐☐☐☐☐☐☐	☐☐☐☐☐☐☐☐	☐☐☐☐☐☐☐☐	☐☐☐☐☐☐☐☐

$PROJ_DIR$\ITM.log

Figure 29. SWO Configuration

After that, open the trace windows of interest (for example, function profiling and interrupt profiling) and make sure that all the traces are enabled in the corresponding windows. Optionally, instruct the code to output custom events through the Terminal I/O window by adding the following lines:

```
#include "arm_itm.h" ITM_EVENT8(1, 'a');
```

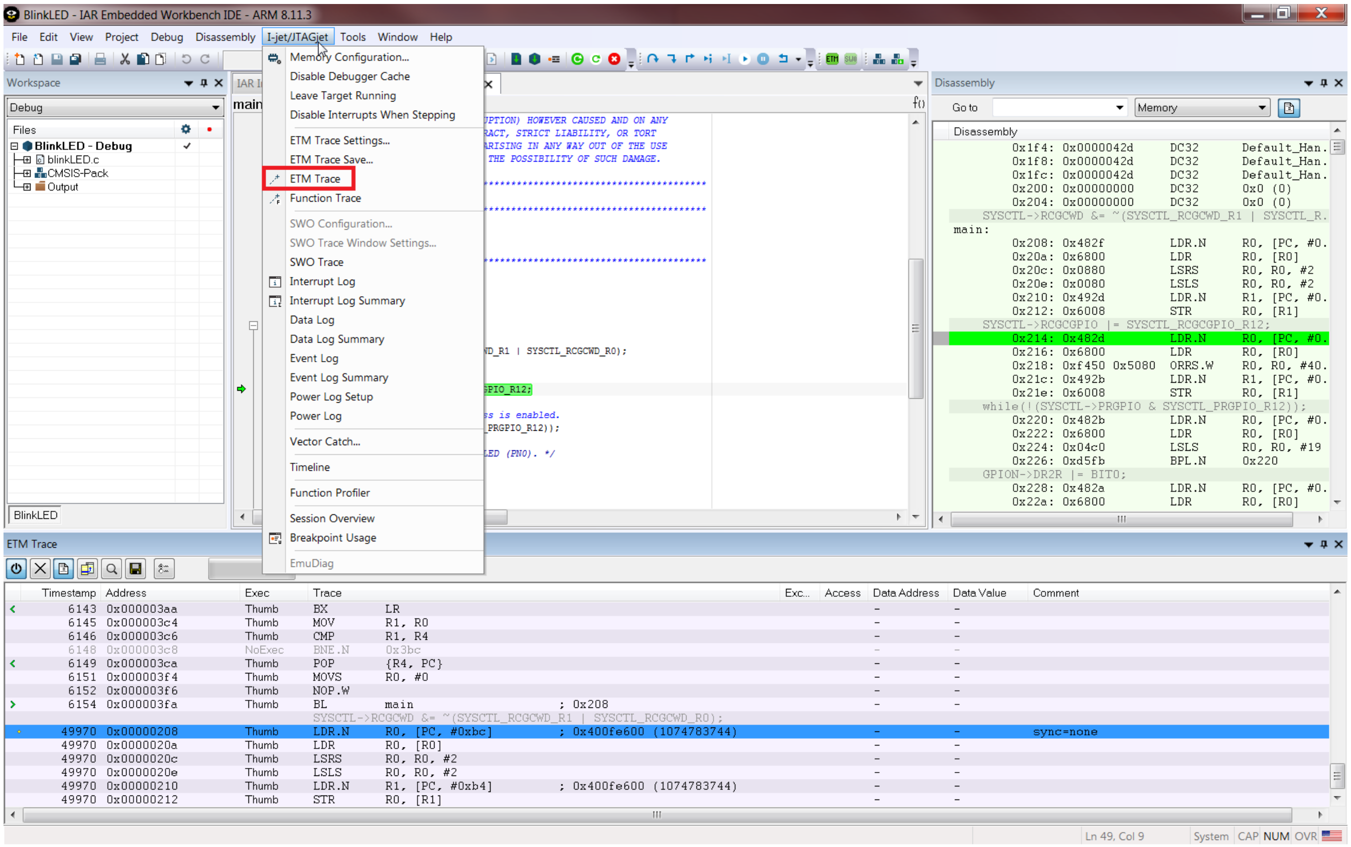
The prerequisite for this is to have the corresponding ITM stimulus ports enabled in the SWO configuration (see Figure 29).

For more information on SWO trace in IAR EWARM, see the user guides on www.iar.com.

6 Using ETM Trace (MSP432E4 Devices Only)

The Embedded Trace Macrocell (ETM) on some TI devices in combination with a ETM trace enabled probe like IAR I-Jet Trace and Segger J-Trace entitles you to perform non-intrusive tracing of every single instruction in your MCU.

For activating ETM just start a debug session and select **ETM trace** from the debugger menu (see Figure 30). Trace data width is preconfigured to 4 bit and ETM trace should work out of the box.



```
Disassembly
        0x1f4: 0x0000042d         DC32      Default_Han.
        0x1f8: 0x0000042d         DC32      Default_Han.
        0x1fc: 0x0000042d         DC32      Default_Han.
        0x200: 0x00000000         DC32      0x0 (0)
        0x204: 0x00000000         DC32      0x0 (0)
    SYSCTL->RCGCWD &= ~(SYSCTL_RCGCWD_R1 | SYSCTL_R.
main:
        0x208: 0x482f             LDR.N     R0, [PC, #0.
        0x20a: 0x6800             LDR       R0, [R0]
        0x20c: 0x0880             LSRS      R0, R0, #2
        0x20e: 0x0080             LSLS      R0, R0, #2
        0x210: 0x492d             LDR.N     R1, [PC, #0.
        0x212: 0x6008             STR       R0, [R1]
    SYSCTL->RCGCGPIO |= SYSCTL_RCGCGPIO_R12;
        0x214: 0x482d             LDR.N     R0, [PC, #0.
        0x216: 0x6800             LDR       R0, [R0]
        0x218: 0xf450 0x5080      ORRS.W    R0, R0, #40.
        0x21c: 0x492b             LDR.N     R1, [PC, #0.
        0x21e: 0x6008             STR       R0, [R1]
    while(!(SYSCTL->PRGPIO & SYSCTL_PRGPIO_R12));
        0x220: 0x482b             LDR.N     R0, [PC, #0.
        0x222: 0x6800             LDR       R0, [R0]
        0x224: 0x04c0             LSLS      R0, R0, #19
        0x226: 0xd5fb             BPL.N     0x220
    GPION->DR2R |= BIT0;
        0x228: 0x482a             LDR.N     R0, [PC, #0.
        0x22a: 0x6800             LDR       R0, [R0]
```

```
 Timestamp Address      Exec    Trace                                              Exc... Access Data Address Data Value Comment
<     6143 0x000003aa   Thumb   BX      LR                                                        -            -
      6145 0x000003c4   Thumb   MOV     R1, R0                                                    -            -
      6146 0x000003c6   Thumb   CMP     R1, R4                                                    -            -
      6148 0x000003c8   NoExec  BNE.N   0x3bc                                                     -            -
<     6149 0x000003ca   Thumb   POP     {R4, PC}                                                  -            -
      6151 0x000003f4   Thumb   MOVS    R0, #0                                                    -            -
      6152 0x000003f6   Thumb   NOP.W                                                             -            -
>     6154 0x000003fa   Thumb   BL      main                  ; 0x208                             -            -
                                SYSCTL->RCGCWD &= ~(SYSCTL_RCGCWD_R1 | SYSCTL_RCGCWD_R0);
     49970 0x00000208   Thumb   LDR.N   R0, [PC, #0xbc]       ; 0x400fe600 (1074783744)           -            -          sync=none
     49970 0x0000020a   Thumb   LDR     R0, [R0]                                                  -            -
     49970 0x0000020c   Thumb   LSRS    R0, R0, #2                                                -            -
     49970 0x0000020e   Thumb   LSLS    R0, R0, #2                                                -            -
     49970 0x00000210   Thumb   LDR.N   R1, [PC, #0xb4]       ; 0x400fe600 (1074783744)           -            -
     49970 0x00000212   Thumb   STR     R0, [R1]                                                  -            -
```

Figure 30. ETM Trace With I-Jet Trace

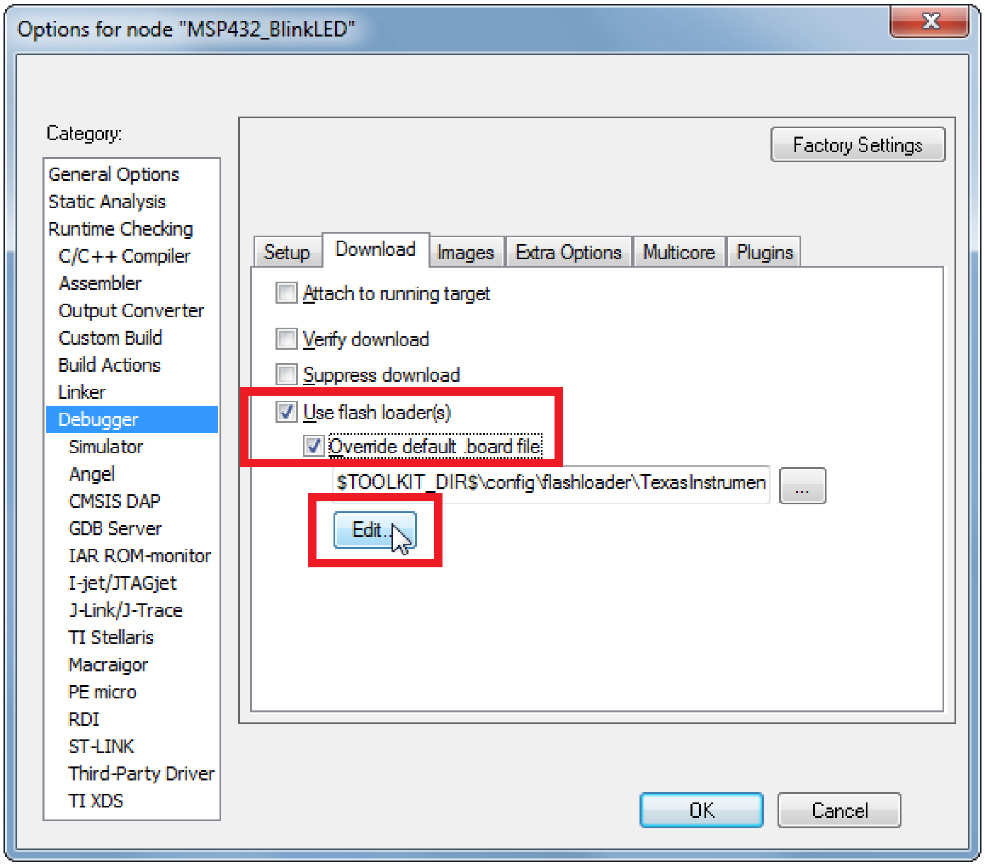
NOTE: ETM trace configuration settings for MSP432E devices has been added recently only. In order to have the best user experience make sure you have the latest IAR EWARM version or the latest MSP432E CMSIS DFP installed.

For more details on ETM trace refer to your debug probe documentation.

7 Erasing the Bootloader (BSL) (MSP432P4xx Devices Only)

The BSL is a program built into an MSP432 microcontroller designed to communicate with the device, primarily for the purpose of reading and writing to memory. The BSL can be erased and rewritten but, by default, the IAR Embedded Workbench tools protect it from accidental deletion.

To erase the BSL right click the active project, then select **Options.** Click the **Debugger** category on the left side and the **Download** tab. In the **Download** tab make sure **Use flash loaders(s)** and **Override default .board file** are checked. Then click the **Edit** button (see Figure 31).



Options for node "MSP432_BlinkLED"

Category:
- General Options
- Static Analysis
- Runtime Checking
  - C/C++ Compiler
  - Assembler
  - Output Converter
  - Custom Build
  - Build Actions
  - Linker
  - Debugger
    - Simulator
    - Angel
    - CMSIS DAP
    - GDB Server
    - IAR ROM-monitor
    - I-jet/JTAGjet
    - J-Link/J-Trace
    - TI Stellaris
    - Macraigor
    - PE micro
    - RDI
    - ST-LINK
    - Third-Party Driver
    - TI XDS

Download tab:

□ Attach to running target

□ Verify download

□ Suppress download

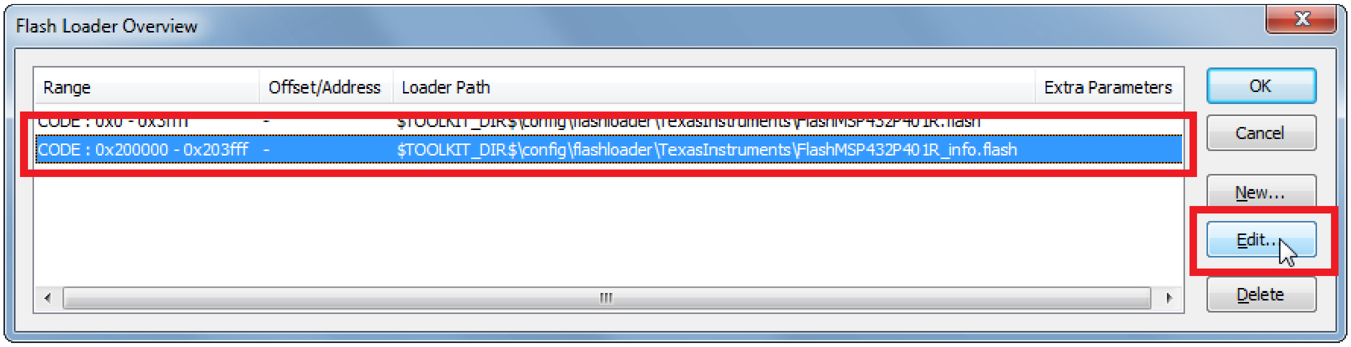
☑ Use flash loader(s)

☑ Override default .board file

$TOOLKIT_DIR$\config\flashloader\TexasInstrumen

Figure 31. Edit Flash Loader Settings

After you click the edit button, a dialog window pops up that allows to change the flash loader settings. Select the section of memory where the BSL resides (0x200000 to 0x203FFF), and then click **Edit** (Figure 32).

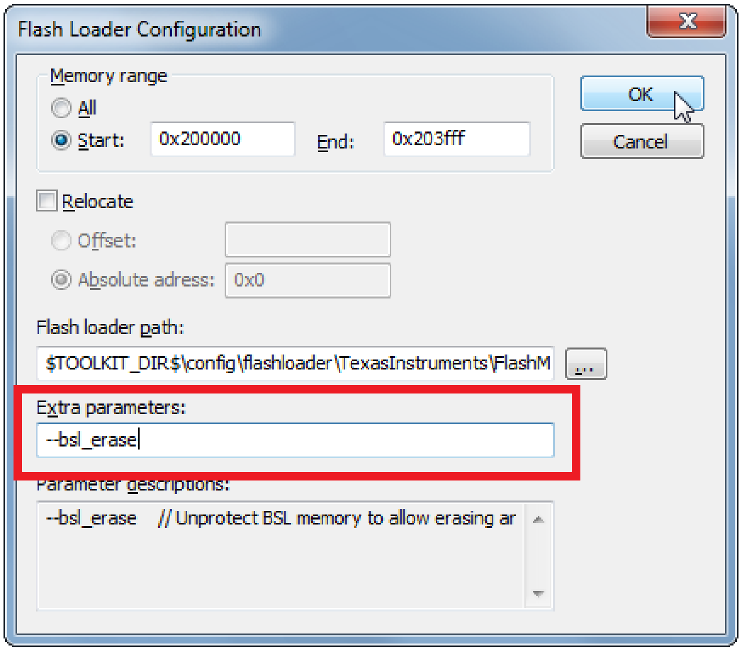


Flash Loader Overview

Range	Offset/Address	Loader Path	Extra Parameters
CODE : 0x0 - 0x3ffff	-	$TOOLKIT_DIR$\config\flashloader\TexasInstruments\FlashMSP432P401R.flash	
CODE : 0x200000 - 0x203fff	-	$TOOLKIT_DIR$\config\flashloader\TexasInstruments\FlashMSP432P401R_info.flash	

Figure 32. Edit Info Memory Settings

You can now specify extra parameters to pass to the RAM loader. There is also a list of parameters in the **Parameter descriptions** info group. To erase the BSL enter "--bsl_erase" in the **Extra parameters** field as shown in Figure 33.



Memory range
○ All
● Start: 0x200000 End: 0x203fff

□ Relocate
○ Offset:
● Absolute adress: 0x0

Flash loader path: $TOOLKIT_DIR$\config\flashloader\TexasInstruments\FlashM

Extra parameters: --bsl_erase

Parameter descriptions:
--bsl_erase // Unprotect BSL memory to allow erasing ar

Figure 33. Add --bsl_erase Option

The BSL memory is now unprotected during code download, and the user can erase and overwrite the current BSL. For more information regarding the BSL, see the *MSP432™ SimpleLink™ Microcontrollers Bootloader (BSL) User's Guide* .

8 EnergyTrace™ Technology

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

MSP432 MCUs with built-in **EnergyTrace+[CPU State]** (or in short **EnergyTrace+**) technology allow real-time monitoring of many internal device states while user program code executes. EnergyTrace+ technology is supported on selected MSP432 devices and debuggers (for example, MSP432P4xx devices).

EnergyTrace mode (without the "+") is the base of **EnergyTrace Technology** and enables analog energy measurement to determine the energy consumption of an application but does not correlate it to internal device information. The EnergyTrace mode is available for all MSP432 devices with selected debuggers, including IAR EWARM.

Devices that support EnergyTrace technology also benefit from the XDS110 EnergyTrace™ High Dynamic Range (ETHDR) debug probe add-on.

8.1 Energy Measurement

Debuggers with EnergyTrace technology support include a new and unique way of continuously measuring the energy supplied to a target microcontroller that differs considerably from the well-known method of amplifying and sampling the voltage drop over a shunt resistor at discrete times. A software controlled DC-DC converter is used to generate the target power supply. The time density of the DC-DC converter charge pulses equals the energy consumption of the target microcontroller. A built-in on-the-fly calibration circuit defines the energy equivalent of a single DC-DC charge pulse.

Figure 34 shows the energy measurement principle. Periods with a small number of charge pulses per time unit indicate low energy consumption and thus low current flow. Periods with a high number of charge pulses per time unit indicate high energy consumption and also a high current consumption. Each charge pulse leads to a rise of the output voltage VOUT, which results in an unavoidable voltage ripple common to all DC-DC converters.

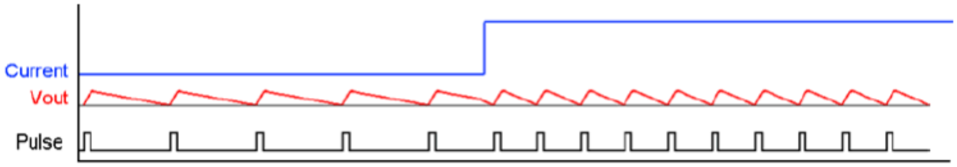


Figure 34. Pulse Density and Current Flow

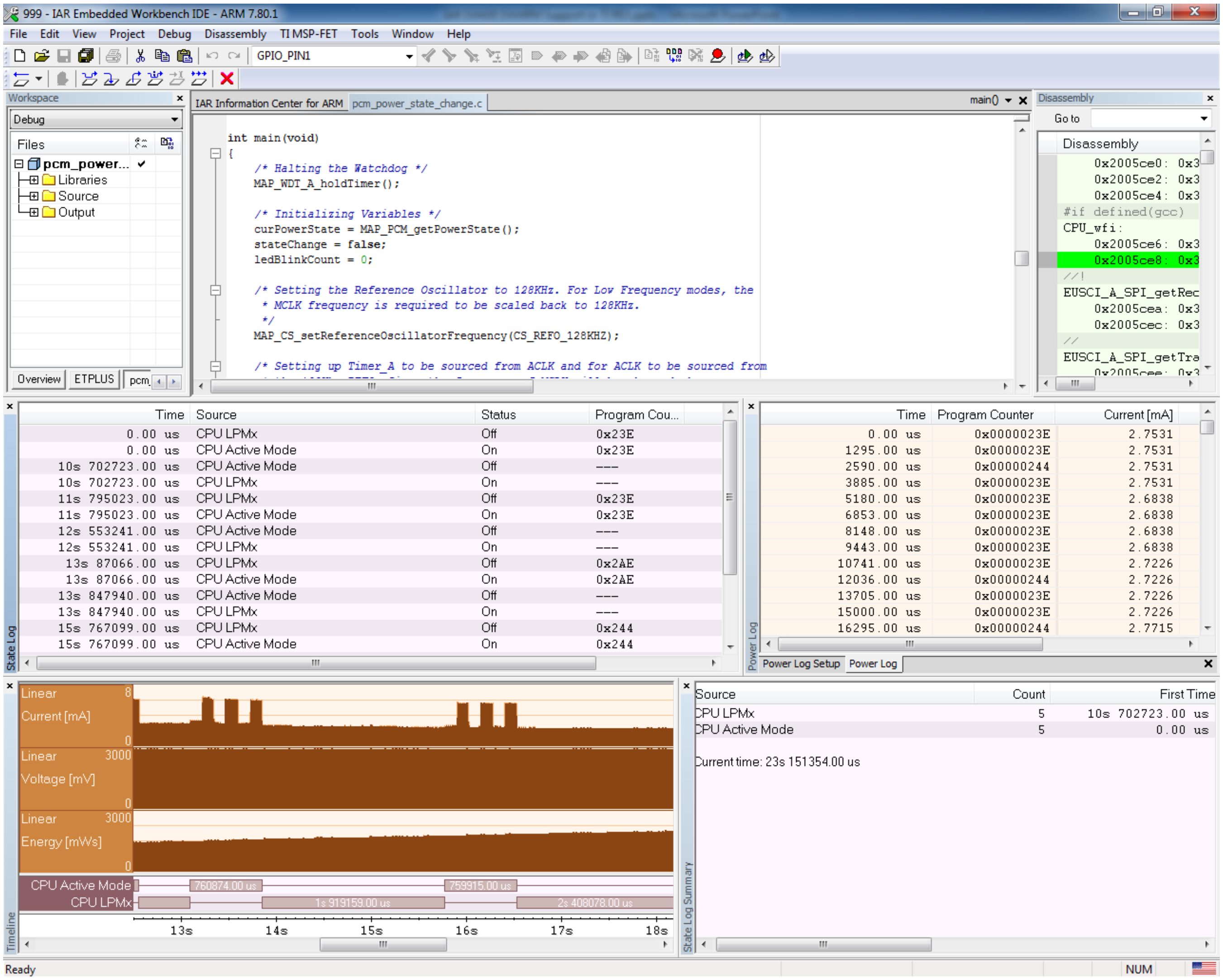
The benefit of sampling continuously is evident: even the shortest device activity that consumes energy contributes to the overall recorded energy. No shunt-based measurement system can achieve this.

8.2 IAR Embedded Workbench for Arm Integration

EnergyTrace Technology is available as part of IAR Embedded Workbench for Arm microcontrollers version 7.80 or higher exclusively for MSP432 MCUs. During debugging of an application, additional windows are available if the debug probe and the target device support EnergyTrace Technology.

The EnergyTrace+ Technology is available only when using the TI MSP-FET debug probe (MSP432P4xx Devices Only).

Devices that support EnergyTrace+ Technology allow sampling of internal device states while an application is executing (see Figure 35).



```
int main(void)
{
    /* Halting the Watchdog */
    MAP_WDT_A_holdTimer();

    /* Initializing Variables */
    curPowerState = MAP_PCM_getPowerState();
    stateChange = false;
    ledBlinkCount = 0;

    /* Setting the Reference Oscillator to 128KHz. For Low Frequency modes, the
     * MCLK frequency is required to be scaled back to 128KHz.
     */
    MAP_CS_setReferenceOscillatorFrequency(CS_REFO_128KHZ);

    /* Setting up Timer_A to be sourced from ACLK and for ACLK to be sourced from
```

```
Disassembly
      0x2005ce0:  0x3
      0x2005ce2:  0x3
      0x2005ce4:  0x3
#if defined(gcc)
CPU_wfi:
      0x2005ce6:  0x3
      0x2005ce8:  0x3
//!
EUSCI_A_SPI_getRec
      0x2005cea:  0x3
      0x2005cec:  0x3
//
EUSCI_A_SPI_getTra
      0x2005cee:  0x3
```

Time	Source	Status	Program Cou...
0.00 us	CPU LPMx	Off	0x23E
0.00 us	CPU Active Mode	On	0x23E
10s 702723.00 us	CPU Active Mode	Off	---
10s 702723.00 us	CPU LPMx	On	---
11s 795023.00 us	CPU LPMx	Off	0x23E
11s 795023.00 us	CPU Active Mode	On	0x23E
12s 553241.00 us	CPU Active Mode	Off	---
12s 553241.00 us	CPU LPMx	On	---
13s 87066.00 us	CPU LPMx	Off	0x2AE
13s 87066.00 us	CPU Active Mode	On	0x2AE
13s 847940.00 us	CPU Active Mode	Off	---
13s 847940.00 us	CPU LPMx	On	---
15s 767099.00 us	CPU LPMx	Off	0x244
15s 767099.00 us	CPU Active Mode	On	0x244

Time	Program Counter	Current [mA]
0.00 us	0x0000023E	2.7531
1295.00 us	0x0000023E	2.7531
2590.00 us	0x00000244	2.7531
3885.00 us	0x0000023E	2.7531
5180.00 us	0x0000023E	2.6838
6853.00 us	0x0000023E	2.6838
8148.00 us	0x0000023E	2.6838
9443.00 us	0x0000023E	2.6838
10741.00 us	0x0000023E	2.7226
12036.00 us	0x00000244	2.7226
13705.00 us	0x0000023E	2.7226
15000.00 us	0x0000023E	2.7226
16295.00 us	0x00000244	2.7715

Source	Count	First Time
CPU LPMx	5	10s 702723.00 us
CPU Active Mode	5	0.00 us

Current time: 23s 151354.00 us

Figure 35. Debug Session With EnergyTrace+ Windows

Click the TI MSP-FET Emulator menu for EnergyTrace+ related entries (see Figure 36).

- State Log
- Power Log
- Timeline

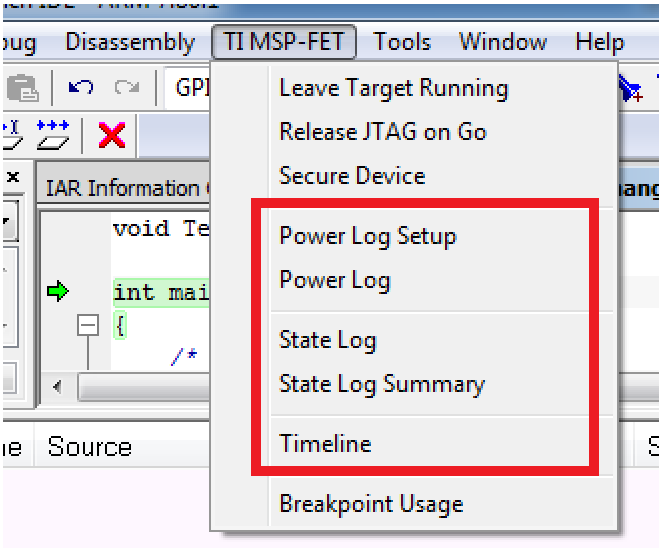


Figure 36. TI MSP-FET Emulator Pulldown Menu With EnergyTrace+ Related Functions

To enable the functions that are related to EnergyTrace technology, right click in the respective window and select **Enable** (see Figure 37).

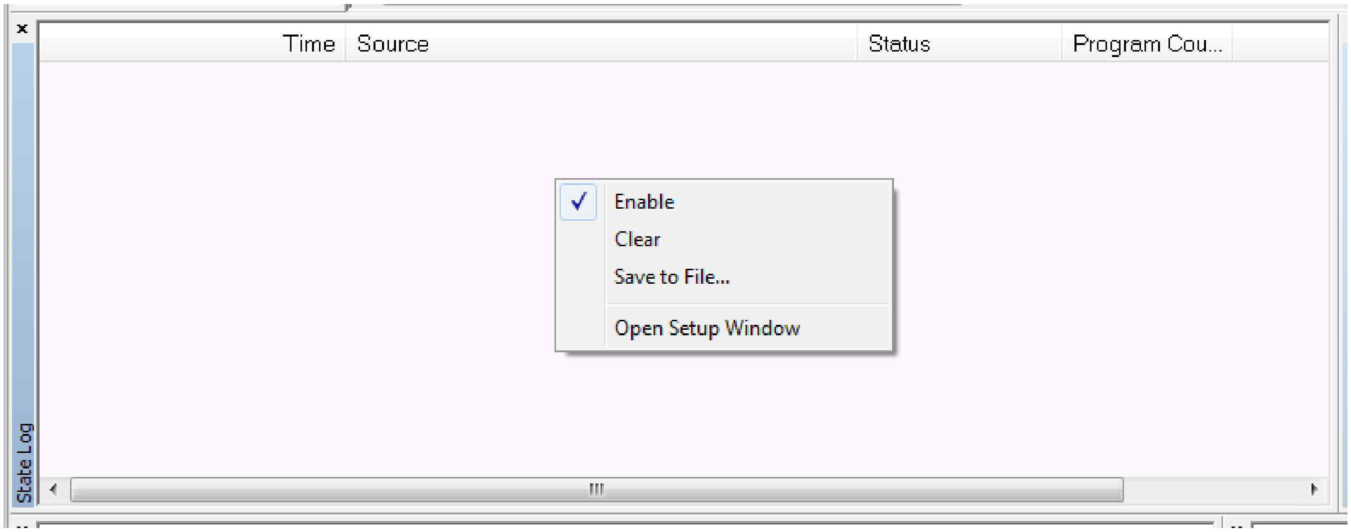
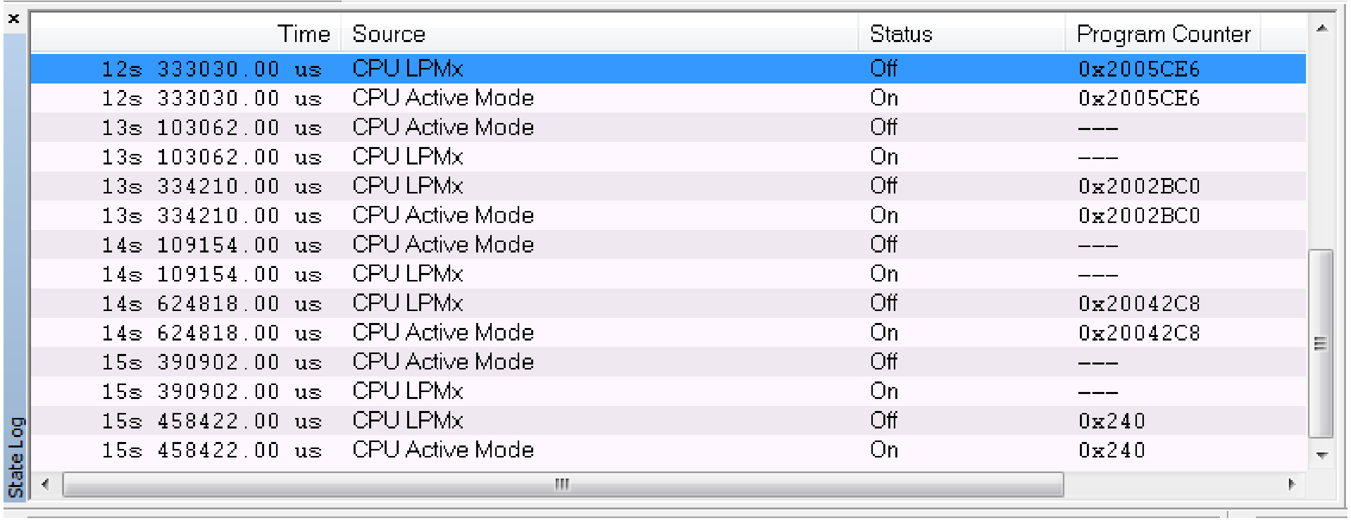


Figure 37. Enabling the State Log Window

8.2.1 State Log

CPU activity is grouped under States. When a program executes, digital data is collected from the target device and displayed in list format (see Figure 38). The State Log shows at what point in time the CPU has been activated or deactivated and gives a reference to the program counter location where this happened.

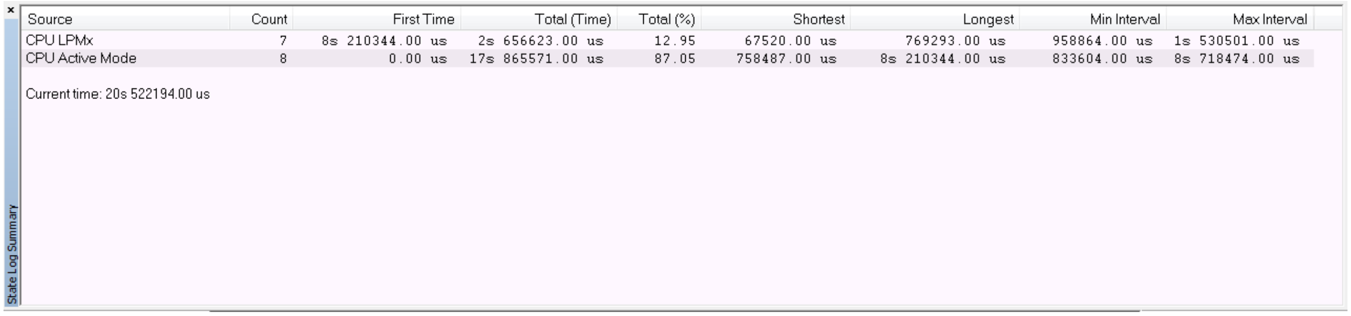


Time	Source	Status	Program Counter
12s 333030.00 us	CPU LPMx	Off	0x2005CE6
12s 333030.00 us	CPU Active Mode	On	0x2005CE6
13s 103062.00 us	CPU Active Mode	Off	---
13s 103062.00 us	CPU LPMx	On	---
13s 334210.00 us	CPU LPMx	Off	0x2002BC0
13s 334210.00 us	CPU Active Mode	On	0x2002BC0
14s 109154.00 us	CPU Active Mode	Off	---
14s 109154.00 us	CPU LPMx	On	---
14s 624818.00 us	CPU LPMx	Off	0x20042C8
14s 624818.00 us	CPU Active Mode	On	0x20042C8
15s 390902.00 us	CPU Active Mode	Off	---
15s 390902.00 us	CPU LPMx	On	---
15s 458422.00 us	CPU LPMx	Off	0x240
15s 458422.00 us	CPU Active Mode	On	0x240

Figure 38. State Log Window With EnergyTrace+ Data

8.2.2 State Log Summary

The State Log Summary window shows a condensed view of the CPU activity of a profiled program (see Figure 39). Click the column headers to sort the data.



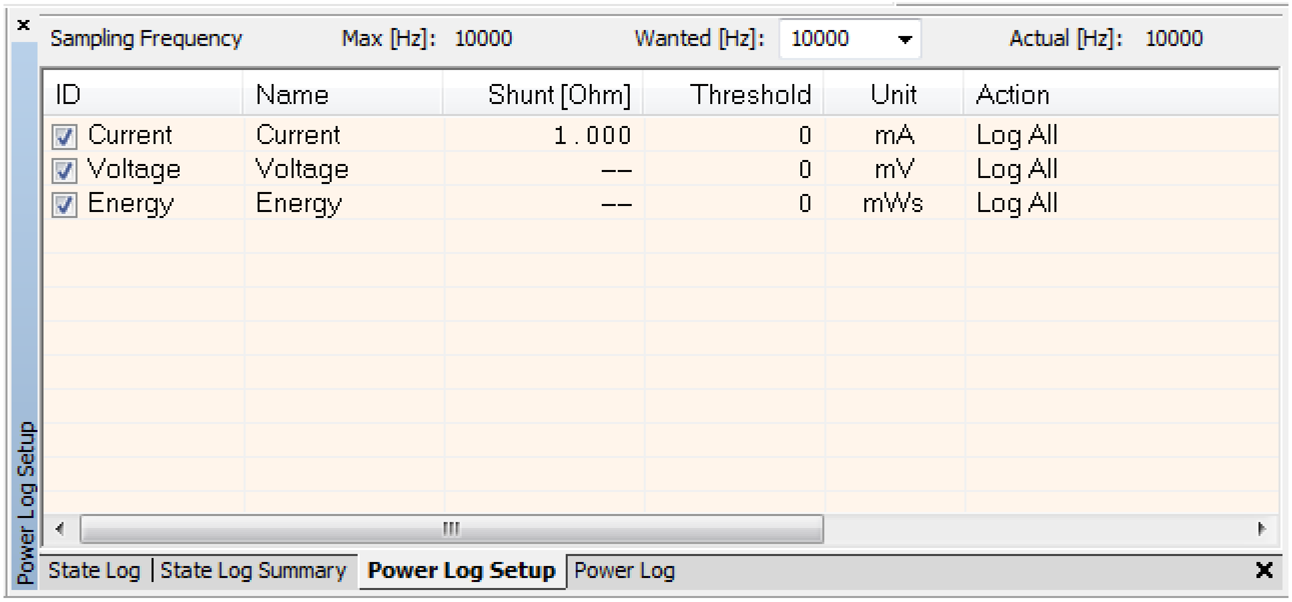
Source	Count	First Time	Total (Time)	Total (%)	Shortest	Longest	Min Interval	Max Interval
CPU LPMx	7	8s 210344.00 us	2s 656623.00 us	12.95	67520.00 us	769293.00 us	958864.00 us	1s 530501.00 us
CPU Active Mode	8	0.00 us	17s 865571.00 us	87.05	758487.00 us	8s 210344.00 us	833604.00 us	8s 718474.00 us

Current time: 20s 522194.00 us

Figure 39. State Log Summary With EnergyTrace+ Data

8.2.3 Power Log Setup

The Power Log Setup can be used to control the analog measurement (see Figure 40). Check each parameter to enable data collection.



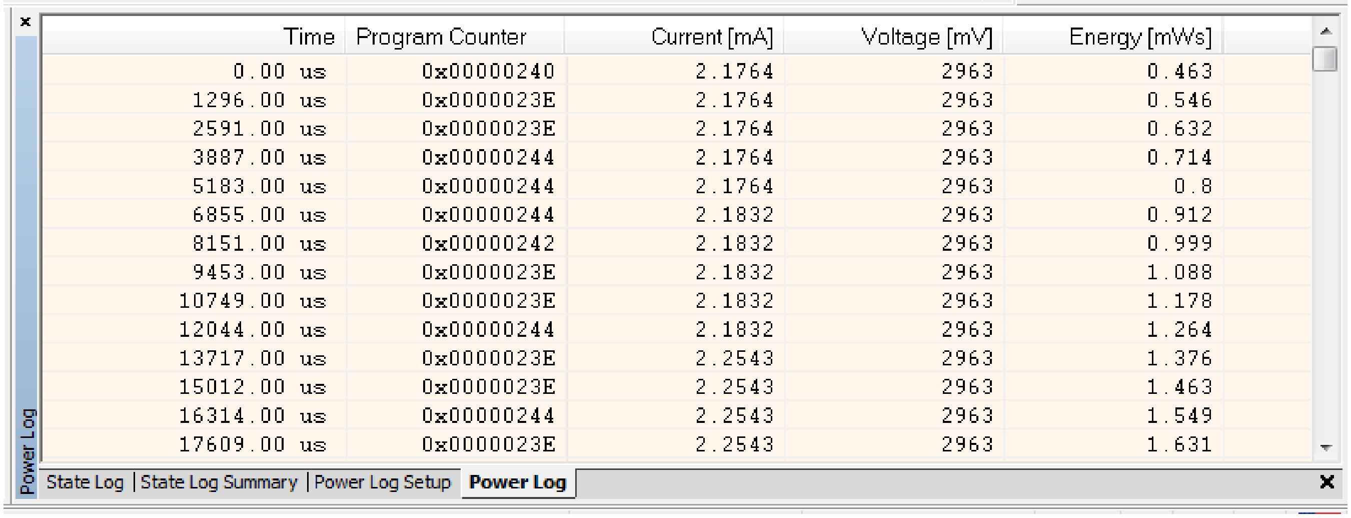
Sampling Frequency Max [Hz]: 10000 Wanted [Hz]: 10000 Actual [Hz]: 10000

ID	Name	Shunt [Ohm]	Threshold	Unit	Action
☑ Current	Current	1.000	0	mA	Log All
☑ Voltage	Voltage	--	0	mV	Log All
☑ Energy	Energy	--	0	mWs	Log All

Figure 40. Power Log Setup Window

8.2.4 Power Log Summary

Similar to the State Log window, the Power Log window shows the current, voltage, and energy profile over time, with reference to the program counter that was sampled at the given time stamp (see Figure 41).

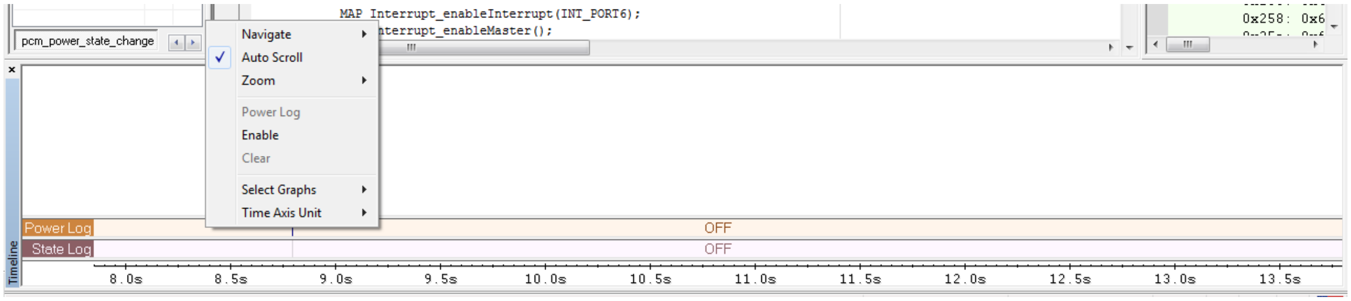


Time	Program Counter	Current [mA]	Voltage [mV]	Energy [mWs]
0.00 us	0x00000240	2.1764	2963	0.463
1296.00 us	0x0000023E	2.1764	2963	0.546
2591.00 us	0x0000023E	2.1764	2963	0.632
3887.00 us	0x00000244	2.1764	2963	0.714
5183.00 us	0x00000244	2.1764	2963	0.8
6855.00 us	0x00000244	2.1832	2963	0.912
8151.00 us	0x00000242	2.1832	2963	0.999
9453.00 us	0x0000023E	2.1832	2963	1.088
10749.00 us	0x0000023E	2.1832	2963	1.178
12044.00 us	0x00000244	2.1832	2963	1.264
13717.00 us	0x0000023E	2.2543	2963	1.376
15012.00 us	0x0000023E	2.2543	2963	1.463
16314.00 us	0x00000244	2.2543	2963	1.549
17609.00 us	0x0000023E	2.2543	2963	1.631

Figure 41. Power Log Window With EnergyTrace+ Data

8.2.5 Timeline

When invoking the Timeline for the first time, both Power Log and State graphs are disabled. Right click each section to enable it, and use the mouse wheel to zoom in and out (see Figure 42 and Figure 43).



```
MAP_Interrupt_enableInterrupt(INT_PORT6);
    nterrupt_enableMaster();
```

Figure 42. Timeline With Power Log and State Graphs Disabled

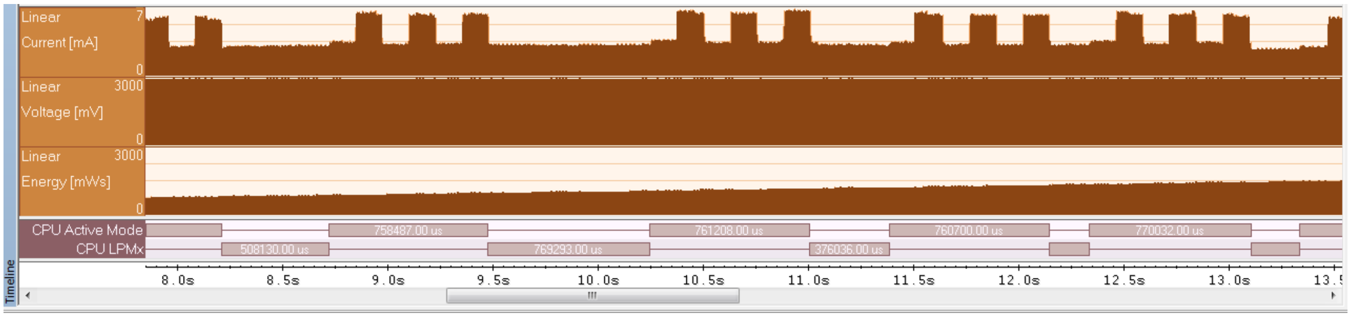


Figure 43. Timeline With EnergyTrace+ Data

8.3 *Measuring Low-Power Currents*

During the capture of the internal states or even when simply executing until breakpoint halt, the target microcontroller is constantly accessed by the JTAG or SWD debug logic. These debug accesses consume energy that is included in the numbers shown in the Power Log window and graph. To measure the absolute power consumption of the application, TI recommends using the EnergyTrace mode in combination with the Release JTAG on Go option. This combination makes sure that the debug logic of the target microcontroller is not accessed while measuring energy consumption.

See Section 4.2.2 for more details.

9 Frequently Asked Questions

Q: My project does not build or reports errors about missing files. What could be wrong?

A: Make sure you have selected the correct device and that your project options are correct. Open the “C/C++ Compiler options”. Verify in the preprocessor tab that the following additional include directories are added:

$TOOLKIT_DIR$\inc\TexasInstruments

$TOOLKIT_DIR$\CMSIS\Include

Also in Preprocessor tab, verify that a symbol is defined for the device and ewarm, for example:

__MSP432P4111__

ewarm

The main.c file should include msp.h:

#include “msp.h”

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

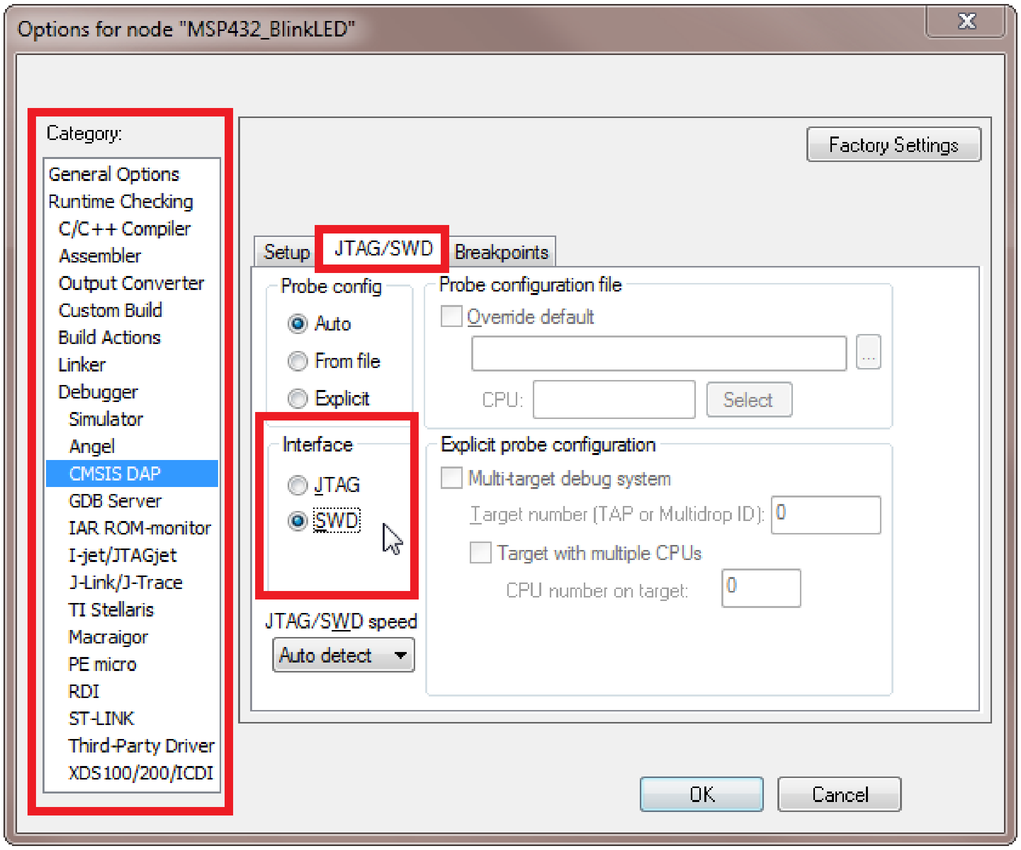
A: Check the following:

- Is the JTAG switch (S101) in the correct orientation?

  Switch to left for XDS110-ET onboard debugger

  Switch to the right for external debugger connection
- Check the debugger settings and change to Serial Wire Debug (SWD) without SWO. When the settings of Port J (PJSEL0 and PJSEL1 bits) are changed, full JTAG access is prevented on these pins. Changing to use SWD allows access through the dedicated debug pins only.

  Figure 44 shows how to configure the debugger to use SWD instead of JTAG by opening the debugger settings window.



Options for node "MSP432_BlinkLED"

Category:

General Options
Runtime Checking
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
CMSIS DAP
GDB Server
IAR ROM-monitor
I-jet/JTAGjet
J-Link/J-Trace
TI Stellaris
Macraigor
PE micro
RDI
ST-LINK
Third-Party Driver
XDS100/200/ICDI

Probe config
● Auto
○ From file
○ Explicit

Probe configuration file
□ Override default
CPU: Select

Interface
○ JTAG
● SWD

Explicit probe configuration
□ Multi-target debug system
Target number (TAP or Multidrop ID): 0
□ Target with multiple CPUs
CPU number on target: 0

JTAG/SWD speed
Auto detect

Figure 44. Choosing SWD in Debugger Settings

- If even this cannot connect, reset the device to factory settings. Review the *Device Security*section of the *Code Composer Studio™ IDE 7.1+ for SimpleLink™ MSP432™ Microcontrollers User's Guide* for information on how to perform a factory reset on the device.

Q: Why doesn't the backchannel UART on the MSP432 LaunchPad development kit work with my serial terminal program at speeds faster than 56000 baud?

A: Certain serial terminal programs such as HTerm or the CCS built-in terminal might not work with the MSP432 LaunchPad development kit at specific baud rates, resulting in the software not being able to open the virtual COM port or in the baud rate being configured incorrectly. An issue with the LaunchPad emulator firmware has been identified and will be fixed in the next release. Until the update is available, use Tera Term, ClearConnex, or HyperTerminal instead, or reduce the baud rate to speeds of 38400 baud or lower.

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

A: When the MSP432 LaunchPad development kit is connected to USB3.0 ports provided by a certain combination of USB3.0 host controller hardware and device drivers, the IDE cannot establish a debug session with the LaunchPad development kit, resulting in an error message similar to "CS_DAP_0: Error connecting to the target: (Error -260 @ 0x0) An attempt to connect to the XDS110 failed" in the case of Code Composer Studio. In this case, the CCS-provided low-level command line utility *xdsdfu* also cannot establish a connection with the LaunchPad development kit.

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

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

A: Check the following:

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

Q: My MSP432E device has been locked. What can I do?

A: MSP432P4xx and MSP432E devices behave differently when locked, and the unlock process also differs. See the corresponding sections in the device-specific technical reference manual or data sheet.

For MSP432E devices, The XDS debug probe supports a command line option to remove the JTAG lock. In contrast to MSP432P4xx devices, this is not assisted by the IDE.

```
C:\ti\ccs_base\common\uscif>dbgjtag.exe -f @<XDS debug probe> -Y unlock, mode=msp432e4
```

The <XDS debug probe> option in this command must be changed to specify the XDS probe in use, for example, *xds110*.

10 Additional IAR EWARM Information

For more information about IAR Embedded Workbench, visit the following links:

- IAR Support
- IAR User's Guides for IAR Embedded Workbench for Arm
- IAR Embedded Workbench
- IAR Embedded Workbench Product News

11 References

1. *MSP432™ SimpleLink™ Microcontrollers Bootloader (BSL) User's Guide*
2. SimpleLink MSP432 SDK
3. Debuggers for MSP432 Microcontrollers
4. Migration Guide for SimpleLink MSP432 SDK
5. CMSIS Pack Documentation
6. MSP432P4xx CMSIS Device Family Pack
7. XDS110 EnergyTrace™ High Dynamic Range (ETHDR) debug probe add-on
8. Uniflash Standalone Flash Tool with additional features for TI MCUs and TI debug probes

Revision History

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

Changes from February 15, 2018 to June 25, 2018 **Page**

- Added the note that begins "The following description is for the CMSIS Manager in IAR EWARM 8.30" in Section 3, *Creating a Project From MSP432 CMSIS Device Family Pack (DFP)* 7
- Changed the steps to create an example in Section 3, *Creating a Project From MSP432 CMSIS Device Family Pack (DFP)*.......... 7
- Added the note that begins "TI XDS110 debug probes are enabled for use..." in Section 4.1, *Using TI XDS100, XDS110, and XDS200 Debug Probes* 9

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

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

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

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

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

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

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

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

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

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