One Technology Way · P.O. Box 9106 · Norwood, MA 02062-9106 · Tel: 781.329.4700 · Fax: 781.461.3113 · www.analog.com

BeMicro FPGA Project for AD8403 with Nios driver

Supported Devices

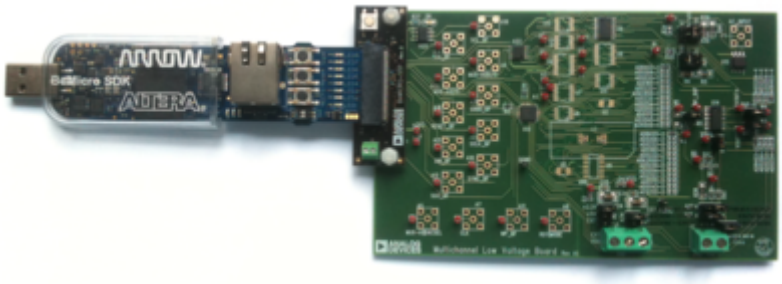
- AD8403

Evaluation Boards

- EVAL-AD8403SDZ

Overview

This lab presents the steps to setup an environment for using the **EVAL-AD8403SDZ** evaluation board together with the **BeMicro SDK** USB stick, the Nios II Embedded Development Suite (EDS) and the Micrium uC-Probe run-time monitoring tool. Below is presented a picture of the EVAL-AD8403SDZ Evaluation Board with the BeMicro SDK Platform.



For component evaluation and performance purposes, as opposed to quick prototyping, the user is directed to use the part evaluation setup. This consists of:

- 1. A controller board like the SDP-B (EVAL-SDP-CS1Z)
- 2. The component SDP compatible product evaluation board
- 3. Corresponding PC software (shipped with the product evaluation board)

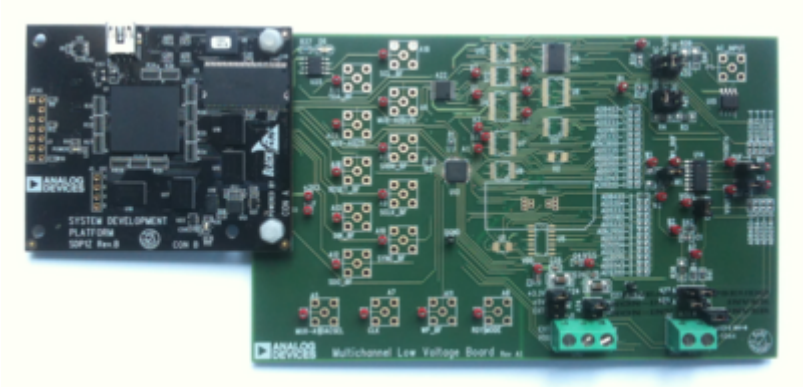
The SDP-B controller board is part of Analog Devices System Demonstration Platform (SDP). It

provides a high speed USB 2.0 connection from the PC to the component evaluation board. The PC runs the evaluation software. Each evaluation board, which is an SDP compatible daughter board, includes the necessary installation file required for performance testing.

Note: it is expected that the analog performance on the two platforms may differ.

28 Sep 2012 08:00 · Adrian Costina

Below is presented a picture of **SDP-B** Controller Board with the **EVAL-AD8403SDZ** Evaluation Board.



The **EVAL-AD8403SDZ** evaluation board is designed to help customers quickly prototype new **AD8403** circuits and reduce design time. The EVAL-AD8403SDZ incorporates several test circuits to evaluate the **AD8403** performance.

The AD8403 is a quad-channel, 256-position, digitally controlled variable resistor device. This device performs the same electronic adjustment function as a mechanical potentiometer or variable resistor. The AD8403 contains four independent variable resistors. Each variable resistor offers a completely programmable value of resistance between the A terminal and the wiper or the B terminal and the wiper. The fixed A-to-B terminal resistance of 1 kΩ, 10 kΩ, 50 kΩ, or 100 kΩ has a ±1% channel-to-channel matching tolerance with a nominal temperature coefficient of 500 ppm/°C

More information

- AD8403 Product Info - pricing, samples, datasheet
- EVAL-AD8403SDZ evaluation board user guide
- BeMicro SDK
- Nios II Embedded Development Suite (EDS)
- Micrium uC-Probe

Getting Started

The first objective is to ensure that you have all of the items needed and to install the software tools

so that you are ready to create and run the evaluation project.

Hardware Items

Below is presented the list of required hardware items:

- Arrow Electronics BeMicro SDK FPGA-based MCU Evaluation Board
- BeMicro SDK/SDP Interposer adapter board
- **EVAL-AD8403SDZ** evaluation board
- Intel Pentium III or compatible Windows PC, running at 866MHz or faster, with a minimum of 512MB of system memory

Software Tools

Below is presented the list of required software tools:

- Quartus II Web Edition design software v11.0
- Nios II EDS v11.0
- uC-Probe run-time monitoring tool, version 2.5

The **Quartus II** design software and the **Nios II EDS** is available via the Altera Complete Design Suite DVD or by downloading from the web.

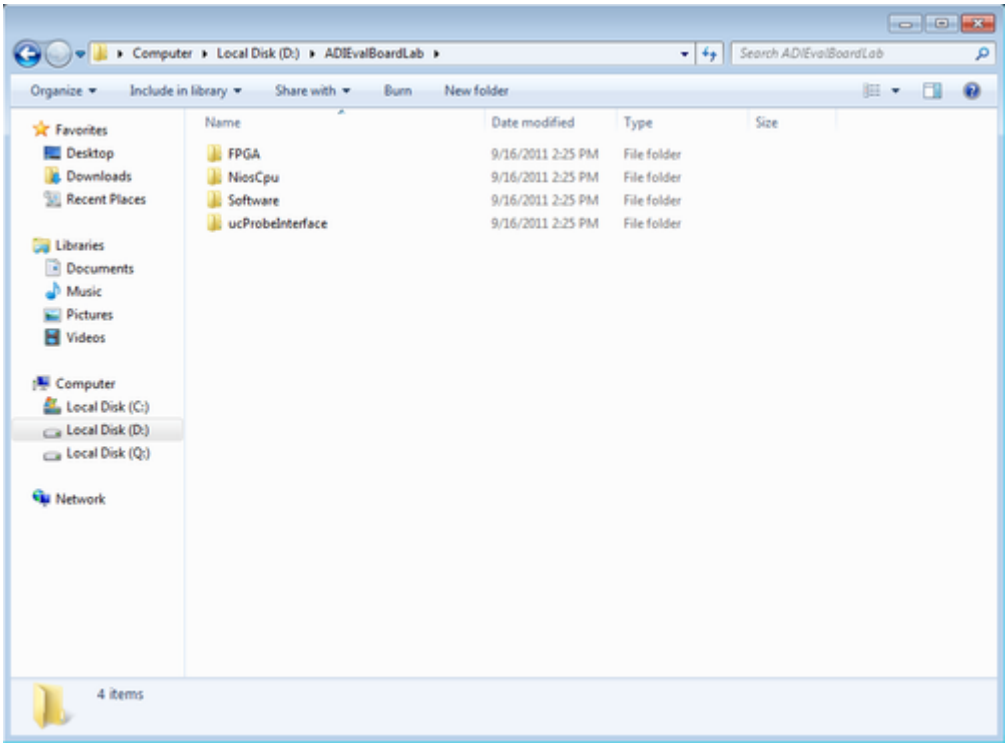
The **Micrium uC/Probe Trial** version 2.5 is available via download from the web at http://micrium.com/tools/ucprobe/trial/. After installation add to the "Path" system variable the entry "*%QUARTUS_ROOTDIR%\bin*" on the third position in the list.

Downloads

- Lab Design Files

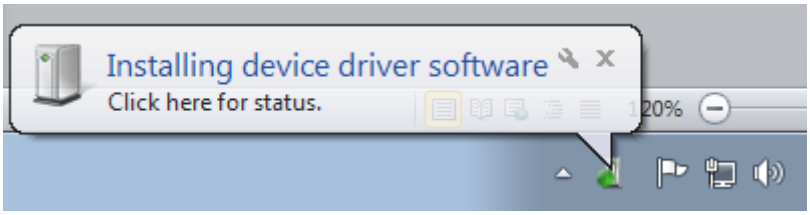
Extract the Lab Files

Create a folder called "***ADIEvalBoardLab***" on your PC and extract the ***ad8403_evalboardlab.zip*** archive to this folder. Make sure that there are ***NO SPACES*** in the directory path. After extracting the archive the following folders should be present in the ***ADIEvalBoardLab*** folder: ***FPGA***, ***Software***, ***ucProbeInterface***, ***NiosCpu***.

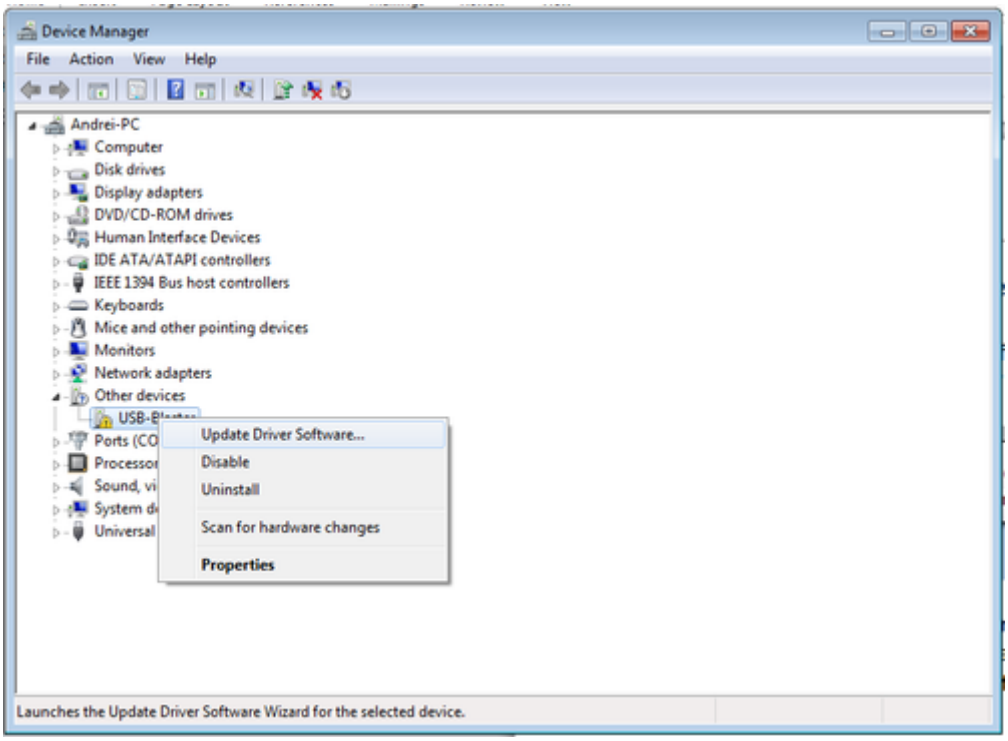


Install the USB-Blaster Device Driver

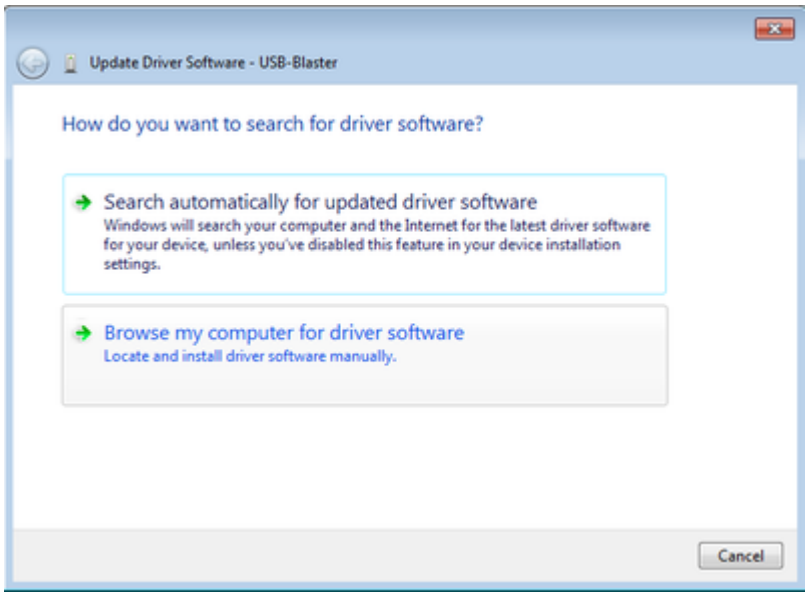
After the **Quartus II** and **Nios II** software packages are installed, you can plug the BeMicro SDK board into your USB port. Your Windows PC will find the new hardware and try to install the driver.

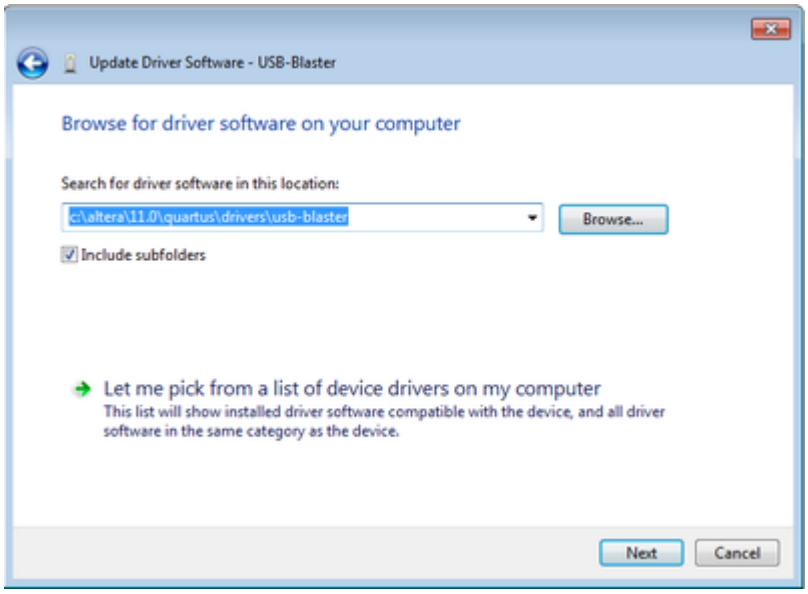


Since Windows cannot locate the driver for the device the automatic installation will fail and the driver has to be installed manually. In the *Device Manager* right click on the **USB-Blaster** device and select ***Update Driver Software***.

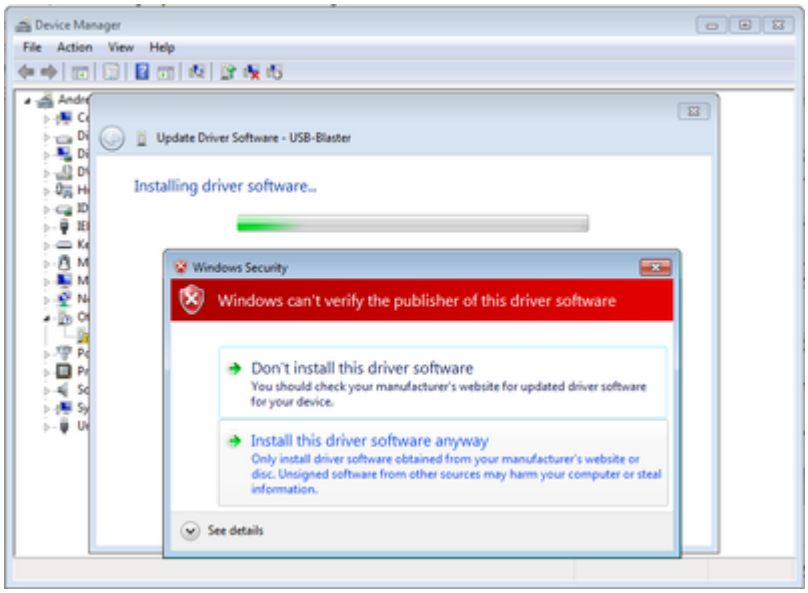


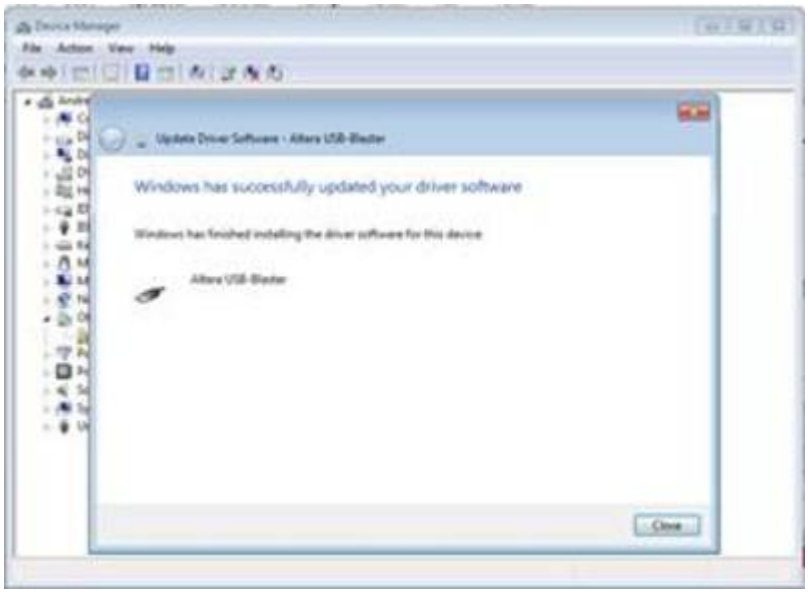
In the next dialog box select the option ***Browse my computer for driver software***. A new dialog will open where it is possible to point to the driver's location. Set the location to **altera\<*version number*>\quartus\drivers\usb-blaster** and press ***Next***.





If Windows presents you with a message that the drivers have not passed Windows Logo testing, please click "***Install this driver software anyway***". Upon installation completion a message will be displayed to inform that the installation is finished.





15 Sep 2011 14:23 · Andrei Cozma

Quick Evaluation

The next sections of this lab present all the steps needed to create a fully functional project that can be used for evaluating the operation of the ADI platform. It is possible to skip these steps and load into the FPGA an image that contains a fully functional system that can be used together with the uC-Probe interface for the ADI platform evalution. The first step of the quick evaluation process is to program the FPGA with the image provided in the lab files. Before the image can be loaded the **Quartus II Web Edition** tool or the Quartus II Programmer must be installed on your computer. To load the FPGA image run the ***program_fpga.bat*** batch file located in the ***ADIEvalBoardLab/FPGA*** folder. After the image was loaded the system must be reset. Now the FPGA contains a fully functional system and it is possible to skip directly to the **DEMONSTRATION PROJECT USER INTERFACE** section of this lab.

15 Sep 2011 14:43 · Andrei Cozma

FPGA Design

The lab is delivered together with a set of design files that are used to evaluate the ADI part. The FPGA image that must be loaded into the BeMicroSDK FPGA is included in the design files. This section presents the components included in the FPGA image and also the procedure to load the image into the FPGA.

FPGA Components

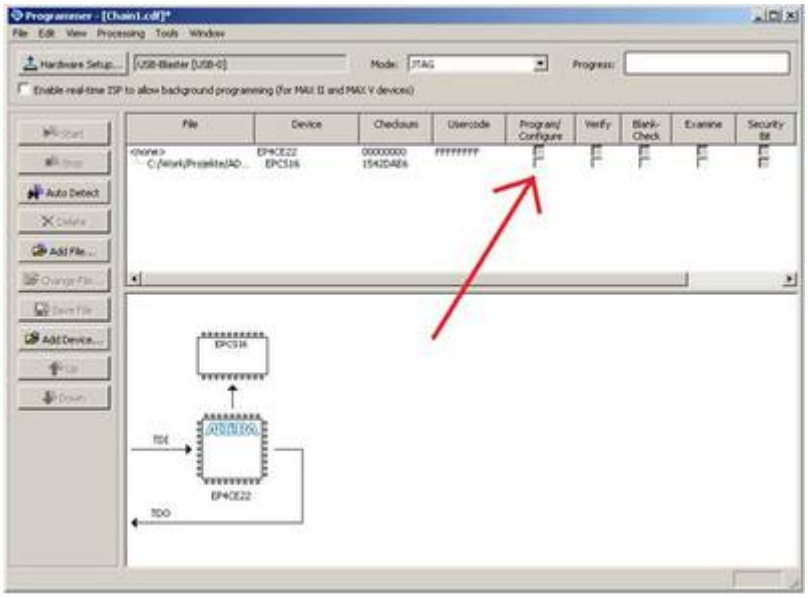
The following components are implemented in the FPGA design:

Name	Address	IRQ
CPU	800	-
Main PLL	80	-
JTAG UART	90	0
uC-Probe UART	A0	1
EPCS FLASH CONTROLLER	1800	2
OnChip RAM	10000	-
LED GPIO	100	-
SPI_0_P0	2000	4
SPI_1_P0	2040	6
GPIO	2080	-
CTRL GPIO	20A0	-
SPI_0_P1	0	5
SPI_1_P1	20	7
SYS ID	40	-
TIMER	60	3
I2C_0	C0	8
I2C_1	E0	9

Load the FPGA Image

To load the FPGA image the following steps must be performed:

- Plug in the **BeMicroSDK** Stick into a USB port
- Start **Altera Quartus Web edition** and start the programmer by selecting the menu option ***Tools→Programmer***
- Select ***Add File*** and select the file ***ADIEvalBoardLab/FPGA/SDP1_bemicro2.jic***
- Check the ***Program/Configure*** box and press ***Start***



After finishing, the image is permanently loaded to the configuration Flash and the system will start with a blinking LED after reset or power up.

15 Sep 2011 14:47 · Andrei Cozma

NIOS II Software Design

This section presents the steps for developing a software application that will run on the **BeMicroSDK** system and will be used for controlling and monitoring the operation of the ADI evaluation board.

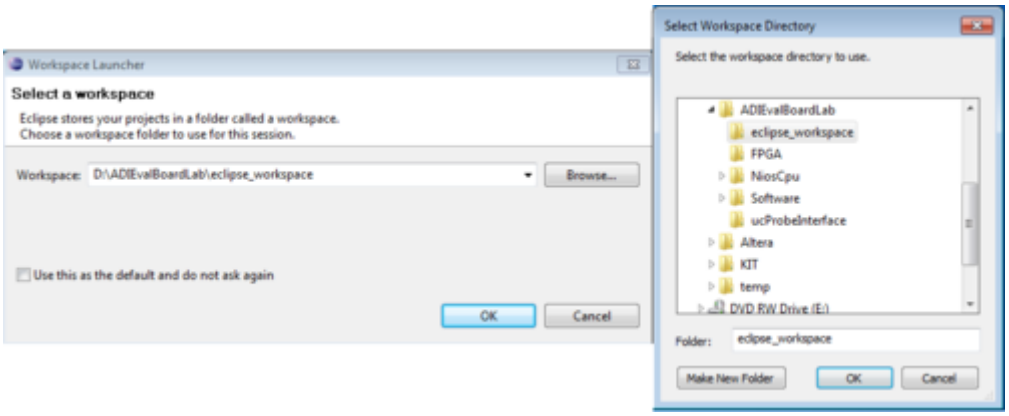
Create a new project using the NIOS II Software Build Tools for Eclipse

Launch the **Nios II SBT** from the ***Start → All Programs → Altera → Nios II EDS 11.0 → Nios II 11.0 Software Build Tools for Eclipse (SBT)***.

NOTE: Windows 7 users will need to right-click and select ***Run as administrator***. Another method is to right-click and select ***Properties*** and click on the ***Compatibility*** tab and select the ***Run This Program As An Administrator*** checkbox, which will make this a permanent change.

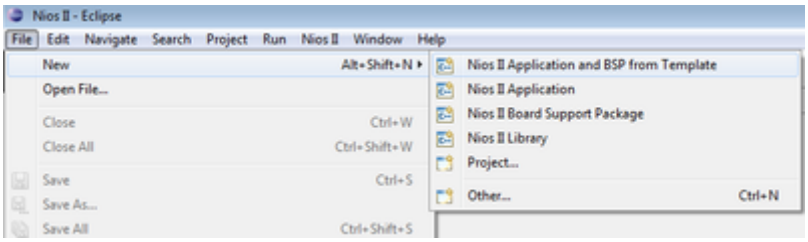
1. Initialize Eclipse workspace

- When Eclipse first launches, a dialog box appears asking what directory it should use for its workspace. It is useful to have a separate Eclipse workspace associated with each hardware project that is created in SOPC Builder. Browse to the ***ADIEvalBoardLab*** directory and click ***Make New Folder*** to create a folder for the software project. Name the new folder "***eclipse_workspace***". After selecting the workspace directory, click ***OK*** and Eclipse will launch and the workbench will appear in the ***Nios II*** perspective.

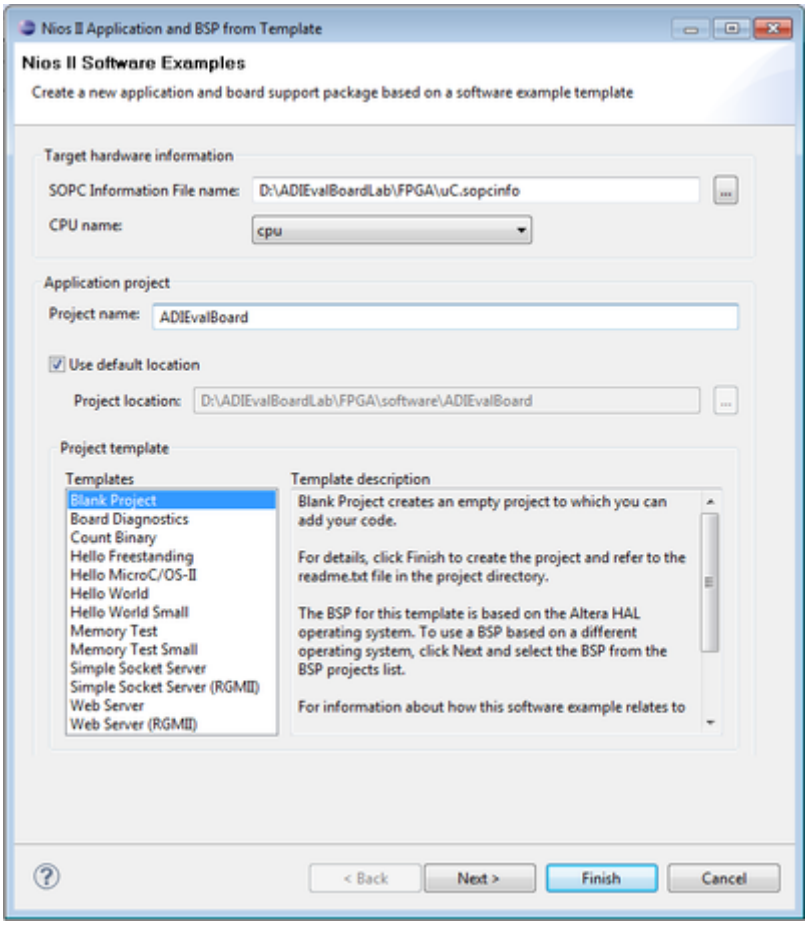


2. Create a new software project in the SBT

- Select ***File → New → Nios II Application and BSP from Template***.

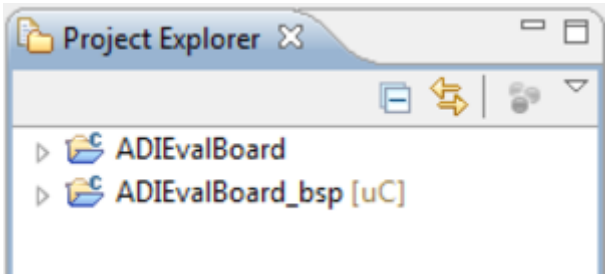


- Click the ***Browse*** button in the ***SOPC Information File Name*** dialog box.
- Select the ***uC.sopcinfo*** file located in the ***ADIEvalBoardLab/FPGA*** directory.
- Set the name of the Application project to "***ADIEvalBoard***".
- Select the ***Blank Project*** template under ***Project template***.
- Click the ***Finish*** button.



The tool will create two new software project directories. Each Nios II application has 2 project directories in the Eclipse workspace.

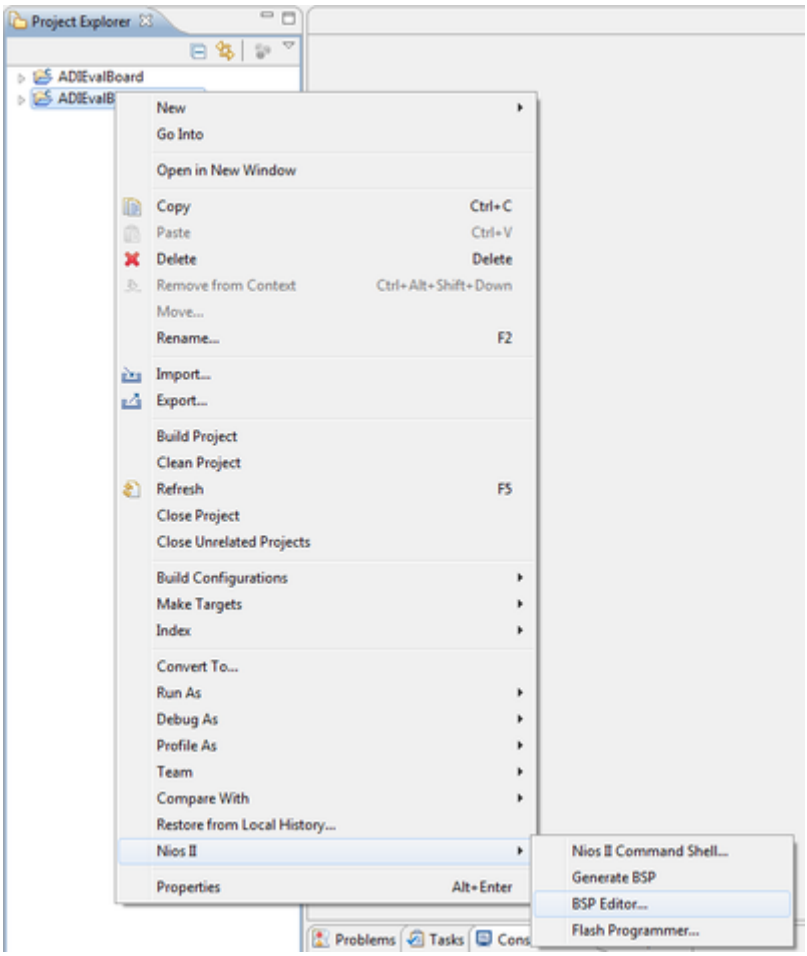
- The application software project itself - this where the application lives.
- The second is the ***Board Support Package (BSP)*** project associated with the main application software project. This project will build the system library drivers for the specific SOPC system. This project inherits the name from the main software project and appends "***_bsp***" to that.



Since you chose the blank project template, there are no source files in the application project directory at this time. The BSP contains a directory of software drivers as well as a system.h header file, system initialization source code and other software infrastructure.

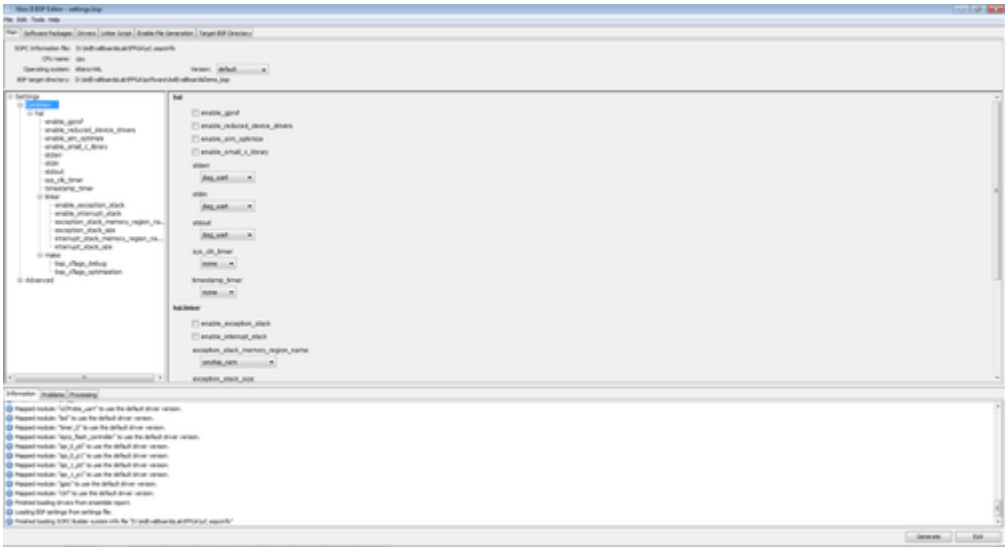
Configure the Board Support Package

- Configure the board support package to specify the properties of this software system by using the ***BSP Editor*** tool. These properties include what interface should be used for *stdio* and *stderr* messages, the memory in which stack and heap should be allocated and whether an operating system or network stack should be included with this BSP.
- Right click on the ***ADIEvalBoard_bsp*** project and select ***Nios II → BSP Editor…*** from the right-click menu.



The software project provided in this lab does not make use of an operating system. All *stdout*, *stdin* and *stderr* messages will be directed to the *jtag_uart*.

- Select the ***Common*** settings view. In the ***Common*** settings view, change the following settings:
  - Select the ***jtag_uart*** for *stdin*, *stdout* and *stderr* messages. Note that you have more than one choice.
  - Select ***none*** for the *sys_clk_timer* and *timestamp_timer*.

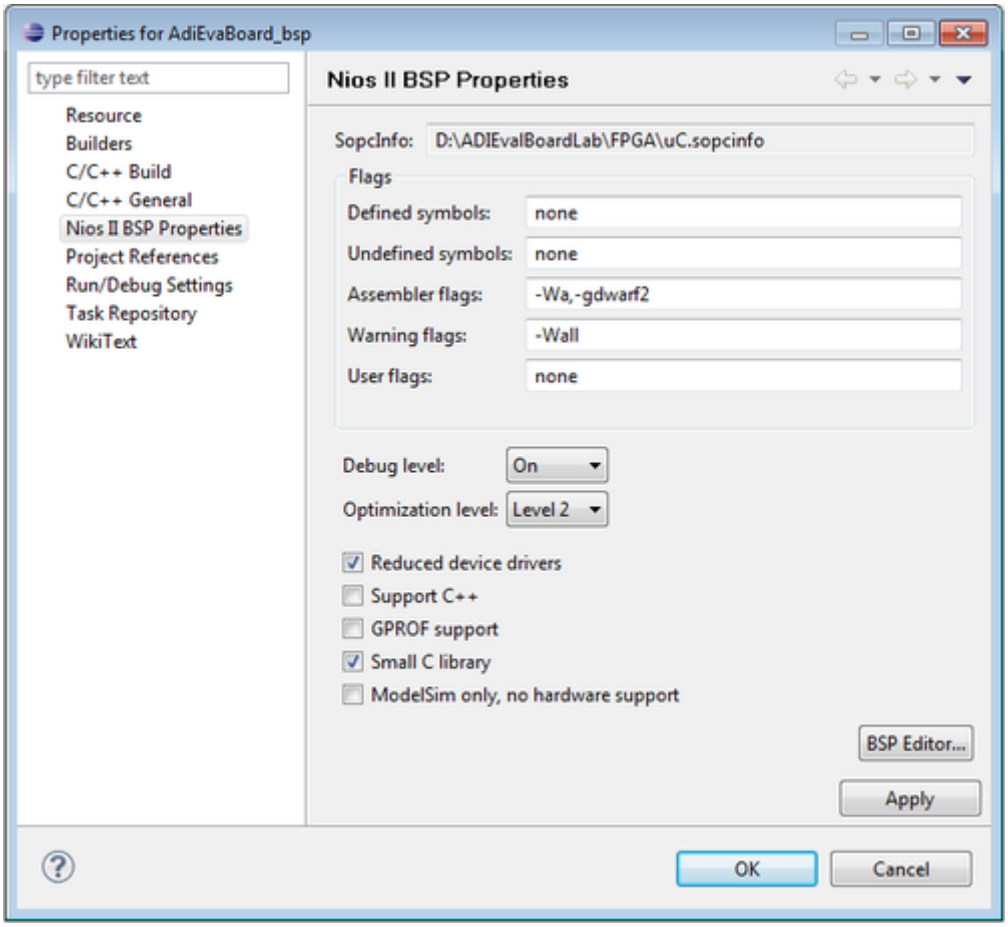


- Select ***File → Save*** to save the board support package configuration to the *settings.bsp* file.
- Click the ***Generate*** button to update the BSP.
- When the generate has completed, select ***File → Exit*** to close the BSP Editor.

Configure BSP Project Build Properties

In addition to the board support package settings configured using the ***BSP Editor***, there are other compilation settings managed by the Eclipse environment such as compiler flags and optimization level.

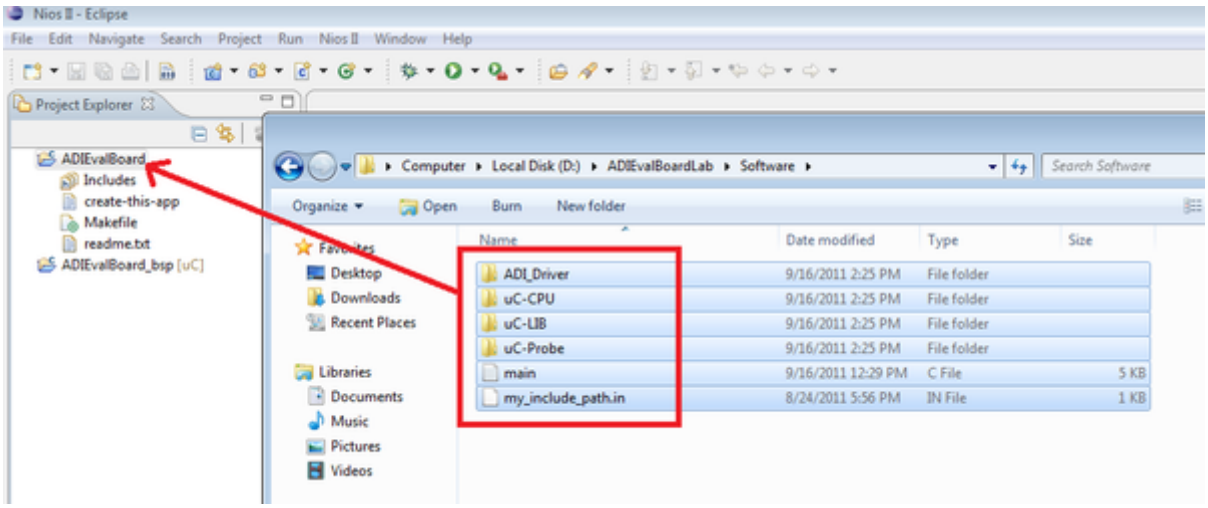
- Right click on the ***ADIEvalBoard_bsp*** software project and select ***Properties*** from the right-click menu.
- On the left-hand menu, select ***Nios II BSP Properties***.
- During compilation, the code may have various levels of optimization which is a tradeoff between code size and performance. Change the ***Optimization level*** setting to ***Level 2***
- Since our software does not make use of C++, uncheck ***Support C++***.
- Check the ***Reduced device drivers*** option
- Check the ***Small C library*** option
- Press ***Apply*** and ***OK*** to regenerate the BSP and close the ***Properties*** window.



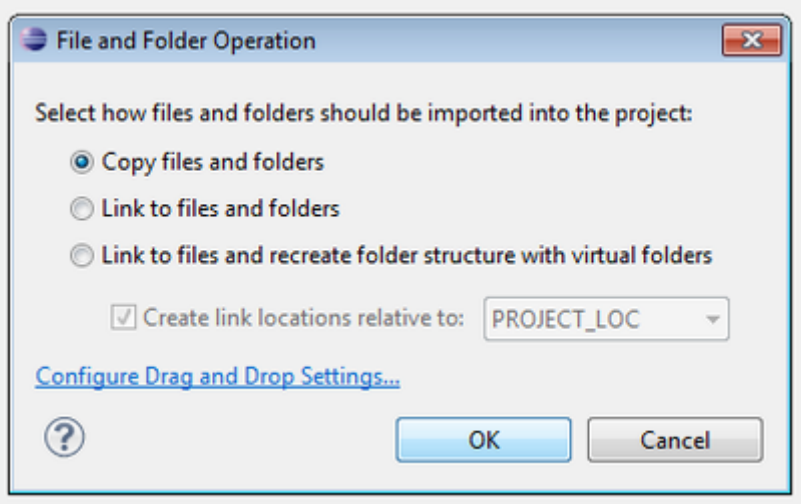
Add source code to the project

In Windows Explorer locate the project directory which contains a directory called ***Software***. In Windows Explorer select all the files and directories from the ***Software*** folder and drag and drop them into the Eclipse software project ***ADIEvalBoard***.

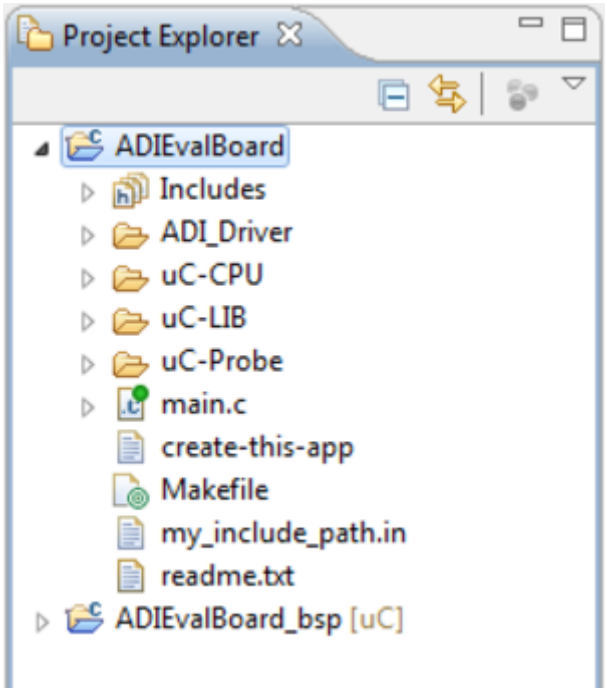
- Select all the files and folders and drag them over the ***ADIEvalBoard*** project in the SBT window and drop the files onto the project folder.



- A dialog box will appear to select the desired operation. Select the option ***Copy files and folders*** and press ***OK***.



- This should cause the source files to be physically copied into the file system location of the software project directory and register these source files within the Eclipse workspace so that they appear in the Project Explorer file listing.

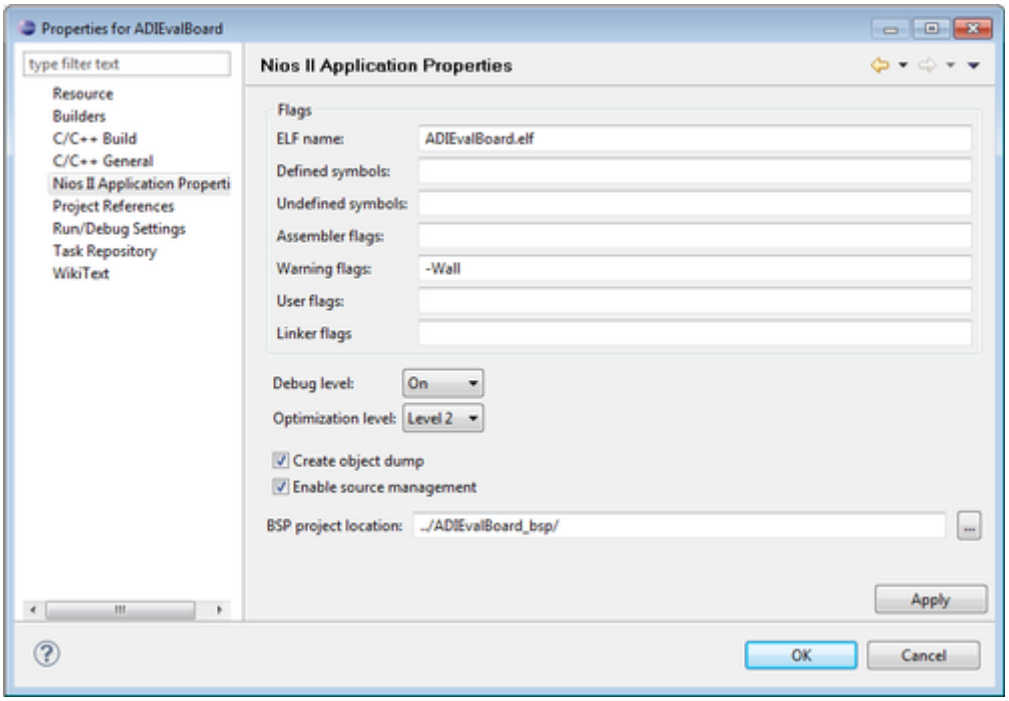


Configure Application Project Build Properties

Just as you configured the optimization level for the BSP project, you should set the optimization level for the application software project ***ADIEvalBoard*** as well.

- Right click on the ***ADIEvalBoard*** software project and select ***Properties*** from the right-click menu.
- On the left-hand menu, select the ***Nios II Application Properties*** tab

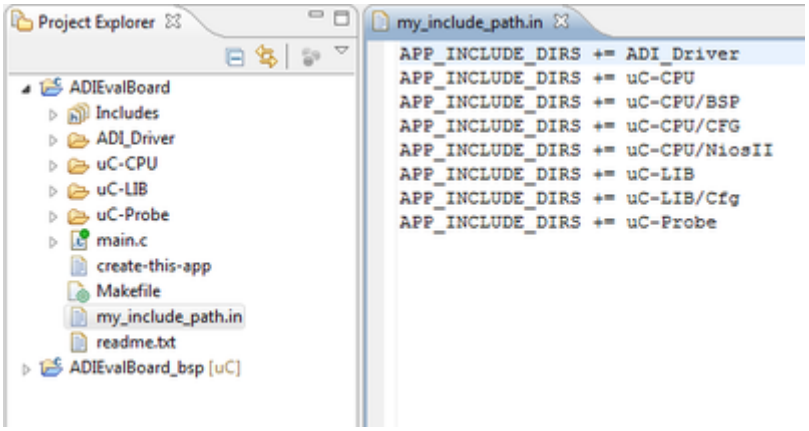
- Change the ***Optimization level*** setting to ***Level 2***.
- Press ***Apply*** and ***OK*** to save the changes.



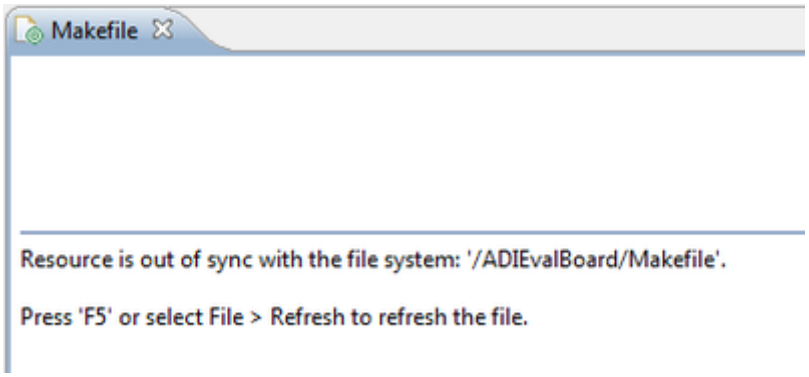
Define Application Include Directories

Application code can be conveniently organized in a directory structure. This section shows how to define these paths in the makefile.

- In the Eclipse environment double click on ***my_include_paths.in*** to open the file.
- Click the ***Ctrl*** and ***A*** keys to select all the text. Click the ***Ctrl*** and ***C*** keys to copy all the text.



- Double click on ***Makefile*** to open the file.
- If you see the message shown here about resources being out of sync, right click on the ***Makefile*** and select ***Refresh***.



- Select the line **APP_INCLUDE_DIRS :=**

```
# List of application specific include directories, library directories and library names
APP_INCLUDE_DIRS :=
APP_LIBRARY_DIRS :=
APP_LIBRARY_NAMES :=
```

- Click the ***Ctrl*** and ***V*** keys to replace the selected line with the include paths.

```
# List of application specific include directories, library directories and library names
APP_INCLUDE_DIRS += ADI_Driver
APP_INCLUDE_DIRS += uC-CPU
APP_INCLUDE_DIRS += uC-CPU/BSP
APP_INCLUDE_DIRS += uC-CPU/CFG
APP_INCLUDE_DIRS += uC-CPU/NiosII
APP_INCLUDE_DIRS += uC-LIB
APP_INCLUDE_DIRS += uC-LIB/Cfg
APP_INCLUDE_DIRS += uC-Probe
APP_LIBRARY_DIRS :=
APP_LIBRARY_NAMES :=
```

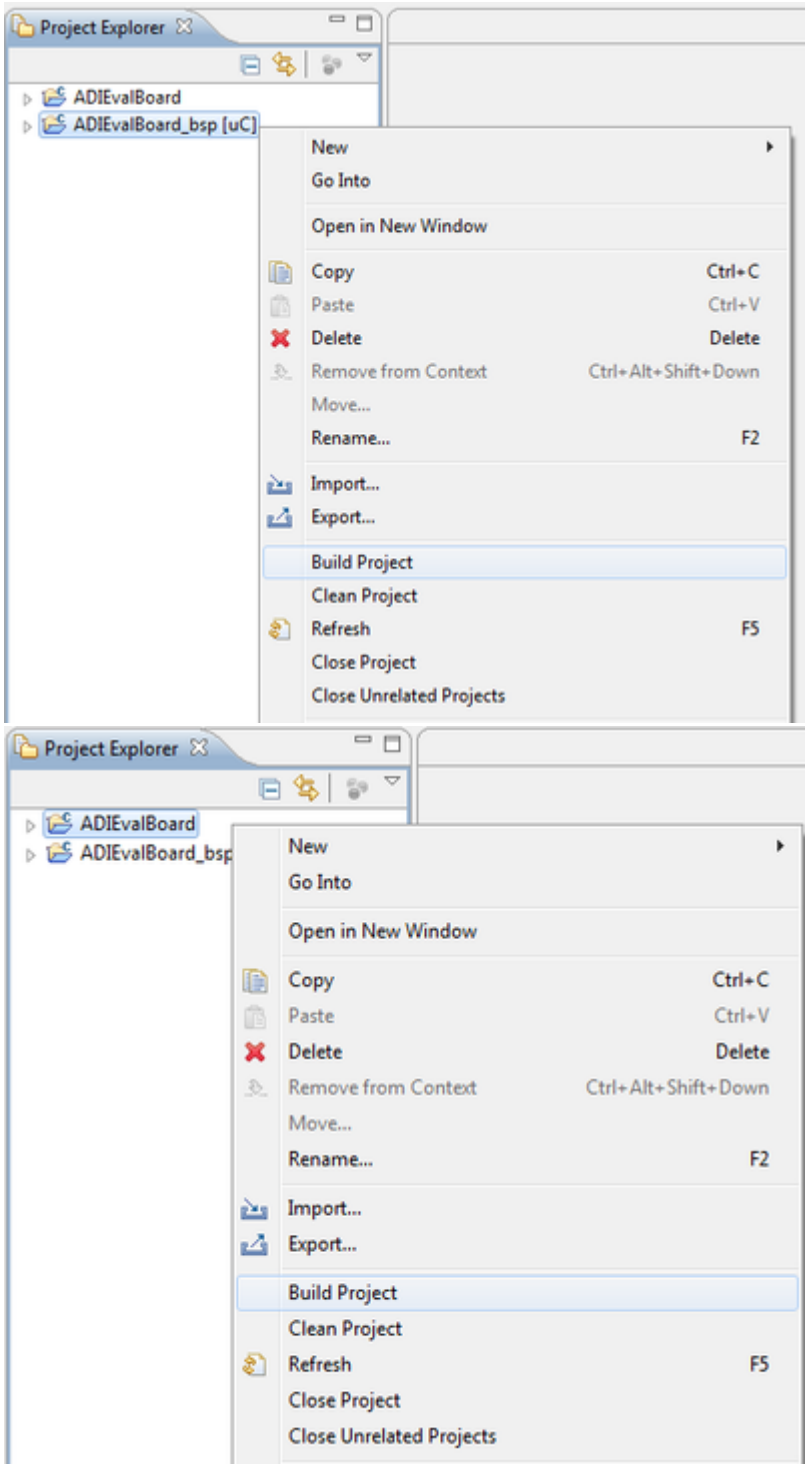
- Click the ***Ctrl*** and ***S*** keys to save the ***Makefile***.

Compile, Download and Run the Software Project

1. Build the Application and BSP Projects

- Right click the ***ADIEvalBoard_bsp*** software project and choose ***Build Project*** to build the board support package.
- When that build completes, right click the ***ADIEvalBoard*** application software project and choose ***Build Project*** to build the Nios II application.

These 2 steps will compile and build the associated board support package, then the actual application software project itself. The result of the compilation process will be an *Executable and Linked Format (.elf)* file for the application, the ***ADIEvalBoard.elf*** file.

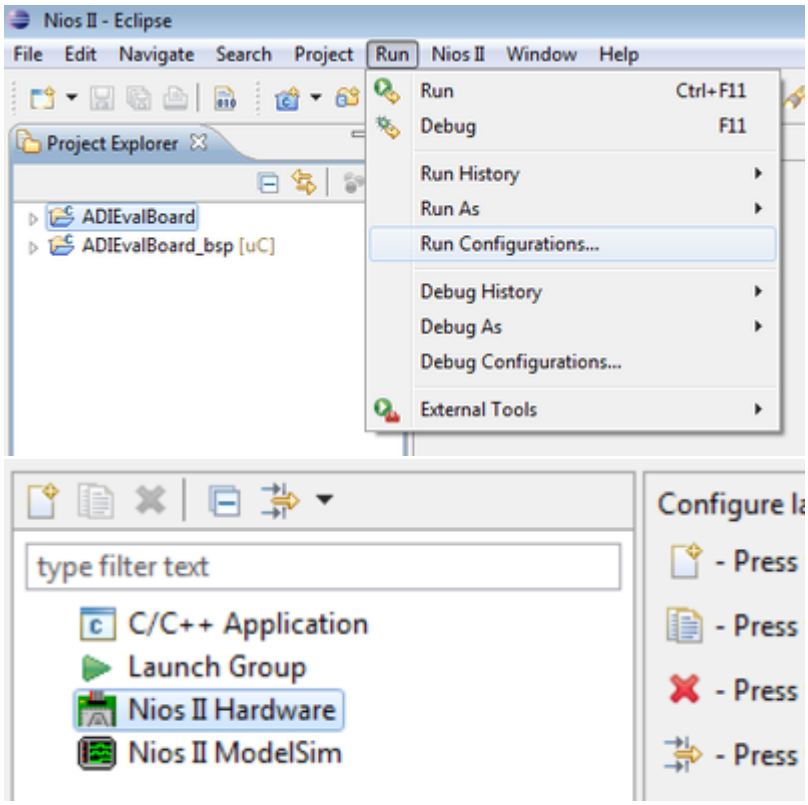


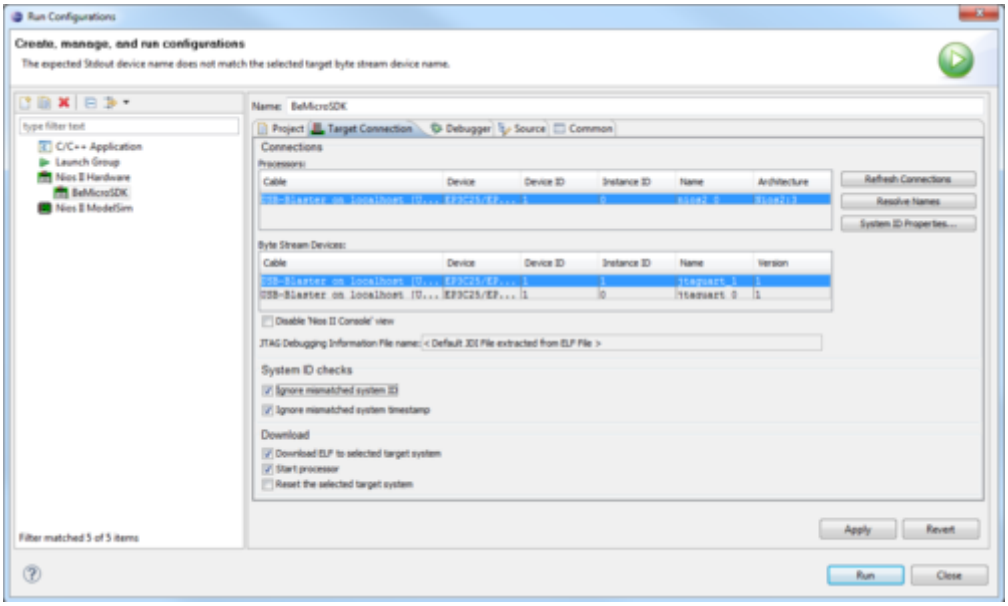
2. Verify the Board Connection

The **BeMicroSDK** hardware is designed with a *System ID* peripheral. This peripheral is assigned a unique value based on when the hardware design was last modified in the SOPC Builder tool. SOPC Builder also places this information in the *.sopcinfo* hardware description file. The BSP is built based on the information in the *.sopcinfo* file.

- Select the ***ADIEvalBoard*** application software project.
- Select ***Run → Run Configurations…***

- Select the ***Nios II Hardware*** configuration type.
- Press the ***New*** button to create a new configuration.
- Change the configuration name to ***BeMicroSDK*** and click ***Apply***.
- On the ***Target Connection*** tab, press the ***Refresh Connections*** button. You may need to expand the window or scroll to the right to see this button.
- Select the ***jtag_uart*** as the ***Byte Stream Device*** for *stdio*.
- Check the ***Ignore mismatched system ID option***.
- Check the ***Ignore mismatched system timestamp option***.



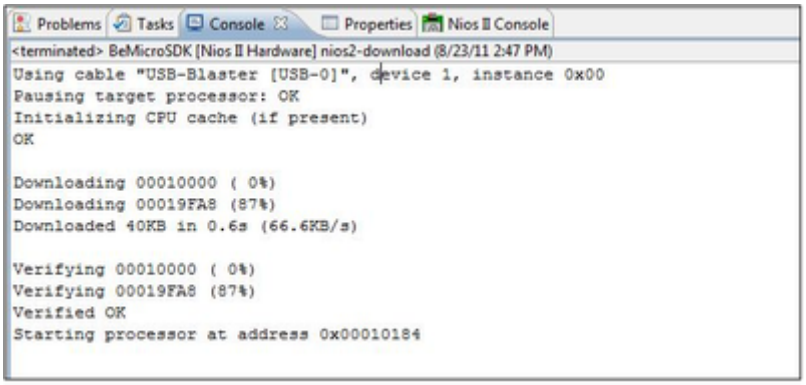


3. Run the Software Project on the Target

To run the software project on the Nios II processor:

- Press the ***Run*** button in the ***Run Configurations*** window.

This will re-build the software project to create an up–to-date executable and then download the code into memory on the **BeMicroSDK** hardware. The debugger resets the Nios II processor, and it executes the downloaded code. Note that the code is verified in memory before it is executed.



The code size and start address might be different than the ones displayed in the above screenshot.

12 Sep 2011 10:39 · Robin Getz

uC-Probe Interface

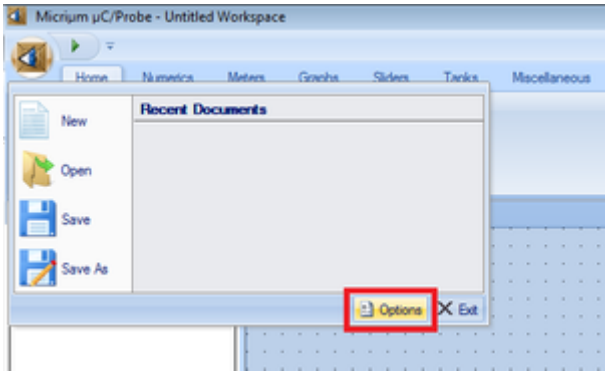
A notable challenge in embedded systems development is to overcome the lack of feedback that such systems typically provide. Many developers resort to blinking LEDs or instrumenting their code with *printf()* in order to determine whether or not their systems are running as expected. **Micrium** provides a unique tool named **µC-Probe** to assist these developers. With this tool, developers can effortlessly read and write the variables on a running embedded system. This section presents the steps required to install the **Micrium uC-Probe** software tool and to run the demonstration project for the ADI evaluation board. A description of the **uC-Probe** demonstration interface is provided.

Configure uC-Probe

Launch **uC-Probe** from the ***Start → All Programs → Micrium → uC-Probe***.

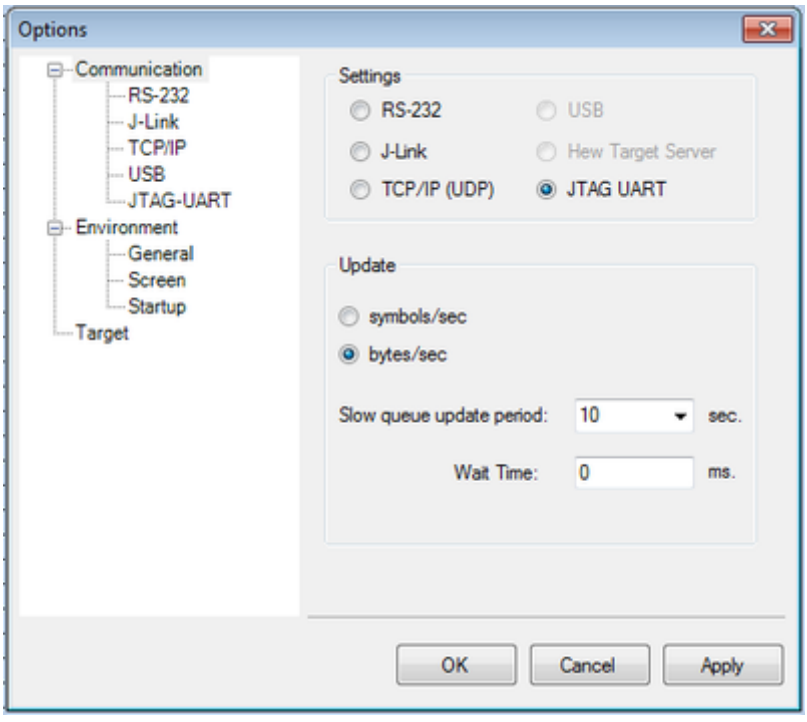
Select **uC-Probe** options.

- Click on the **uC-Probe** icon on the top left portion of the screen.
- Click on the ***Options*** button to open the dialog box.



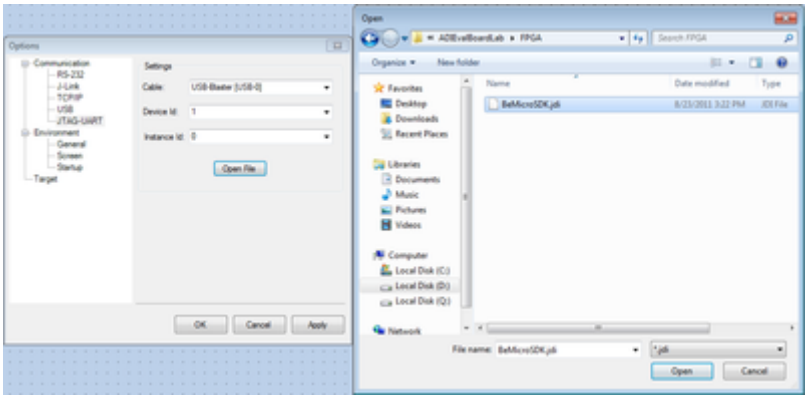
Set target board communication protocol as ***JTAG UART***

- Click on the ***Communication*** tab icon on the top left portion of the dialog box
- Select the ***JTAG UART*** option.

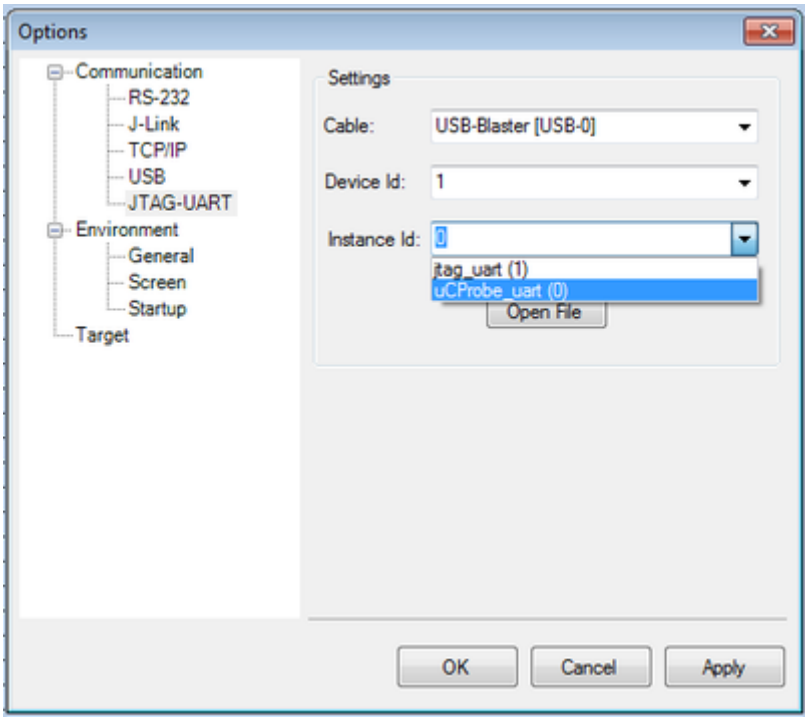


Setup ***JTAG UART*** communication settings

- Select the ***JTAG-UART*** option from the ***Communication*** tab.
- Press the ***Open File*** button to select the JTAG Debug Information file (***.jdi***)
- Navigate to the ***ADIEvalBoardLab/FPGA*** folder and select the BeMicroSDK.jdi file. Press Open.
- Type the value ***1*** in the the ***Device Id*** window.



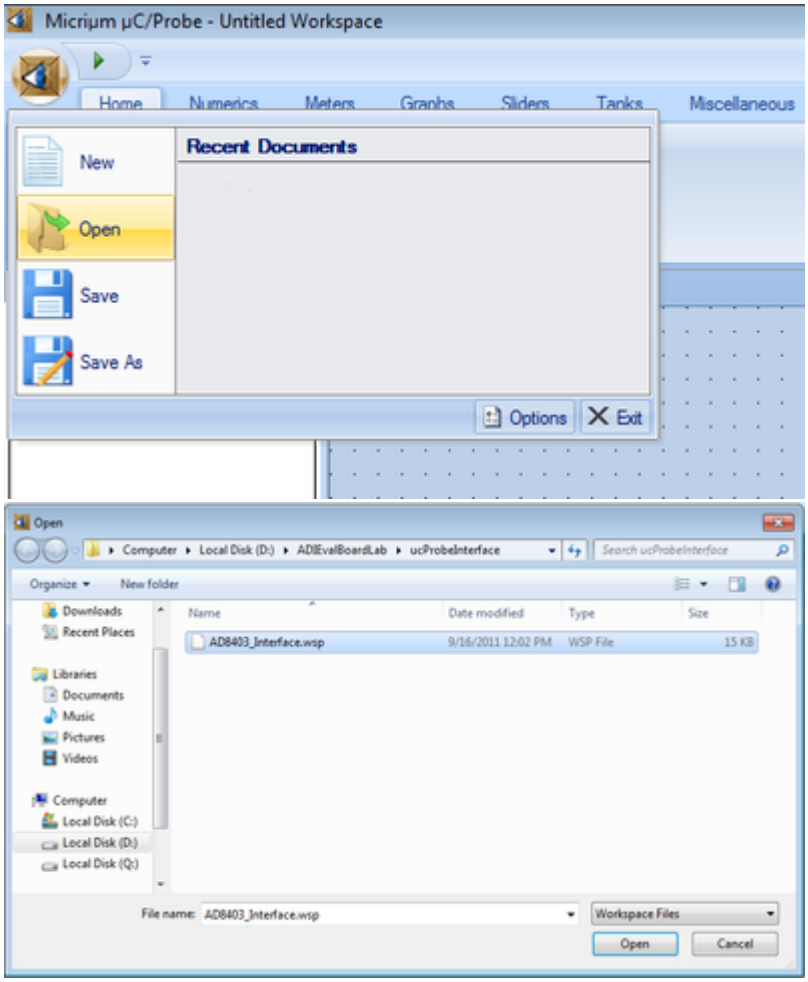
- Select ***uCProbe_uart(0)*** from the ***Instance Id*** pulldown menu.



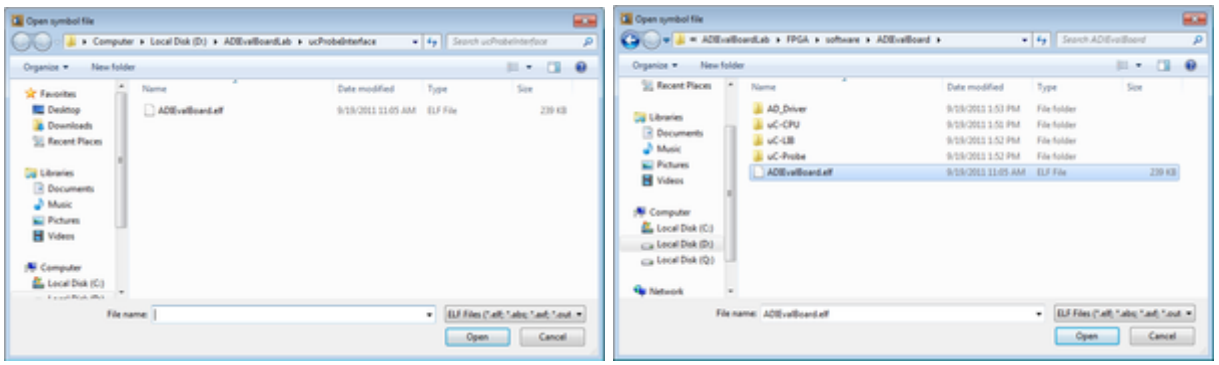
- Press ***Apply*** and ***OK*** to exit the options menu. The embedded target has two UARTs. **uC-Probe** will be communicating with the ***uCProbe_uart***.

Load and Run the Demonstration Project

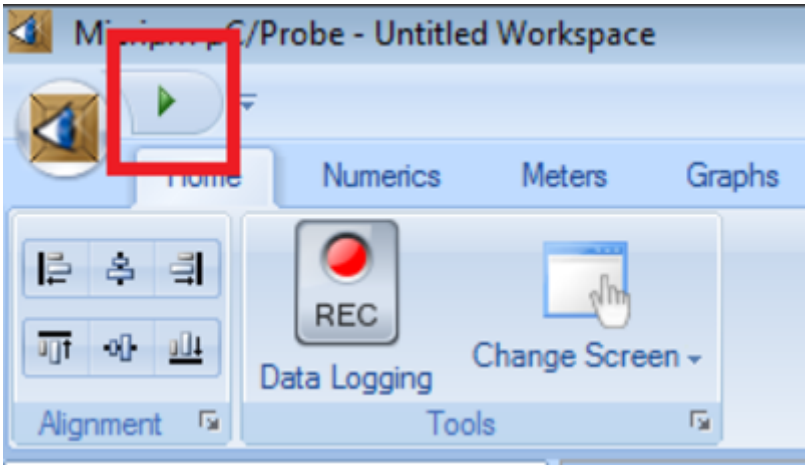
- Click the ***Open*** option from the **uC-Probe** menu and select the file ***ADIEvalBoardLab/ucProbeInterface/AD8403_Interface.wsp***.



- Before opening the interface **uC-Probe** will ask for a symbols file that must be associated with the interface. If the lab was done according to the steps provided in the **Quick Evaluation** section, select the file ***ADIEvalBoardLab/ucProbeInterface/ADIEvalBoard.elf*** to be loaded as a symbol file, otherwise select the file ***ADIEvalBoardLab/FPGA/software/ADIEvalBoard/ADIEvalBoard.elf*** to be loaded as a symbol file.

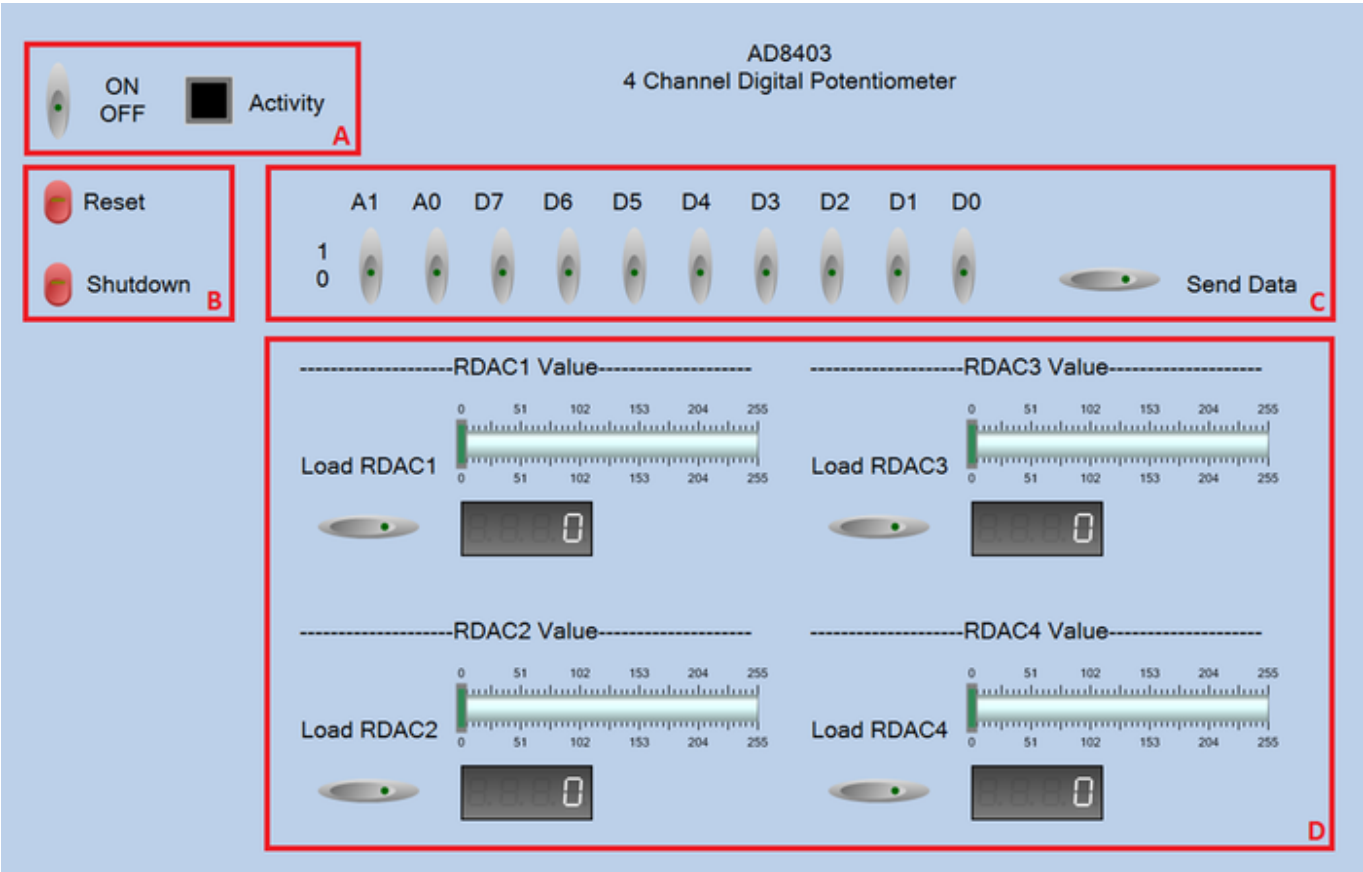


- Run the demonstration project by pressing the ***Play*** button.



Demonstration Project User Interface

The following figure presents the **uC-Probe** interface that can be used for monitoring and controlling the operation of the **EVAL-AD8403SDZ** evaluation board.



Section A is used to activate the board and monitor activity. The communication with the board is activated / deactivated by toggling the ***ON/OFF*** switch. The ***Activity*** LED turns green when the communication is active. If the ***ON/OFF*** switch is set to ***ON*** and the ***Activity*** LED is ***BLACK*** it means that there is a communication problem with the board.

Section B is used to shutdown and reset the device.

- **Shutdown**: When it is pressed, forces the resistors to an end-to-end open-circuit condition on the A terminal and shorts the wiper to the B terminal, achieving a microwatt power shutdown state. When it is released, the previous latch settings put the wiper in the same resistance setting prior to shutdown.
- **Reset**: When it is pressed, forces the wiper to midscale by loading 80H into the VR latch. When it is released, the VR latch remains loaded with 80H, but you can load the desired values.

Section C is used to send a 10-bits word to the device. This data-word can be sent by manually switching the buttons from 0 to 1 or from 1 to 0, as desired, and then toggling the ***Send Data*** switch.

Section D is used to update the RDAC registers by selecting a desirable value on the slider and toggling the ***Load RDACx*** switch.

www.analog.com