

Getting Started with the NI LabVIEW™ Embedded Module for ADI Blackfin Processors

Version 8.6

The NI LabVIEW Embedded Module for ADI Blackfin Processors is a comprehensive graphical development environment for embedded design. Jointly developed by Analog Devices and National Instruments, this module seamlessly integrates the LabVIEW development environment and Blackfin embedded processors.

This module builds on NI LabVIEW Embedded technology, which facilitates dataflow graphical programming for embedded systems and includes hundreds of analysis and signal processing functions, integrated I/O, and an interactive debugging interface. With the Embedded Module for Blackfin Processors, you can enable cache, optimize linking, and view live front panel updates using JTAG, serial, or TCP/IP, as well as use VisualDSP++ compiler options through LabVIEW. The Embedded Module for Blackfin Processors includes the LabVIEW C Code Generator, which generates C code from the LabVIEW block diagram.

Engineers and scientists can lower development costs, achieve faster development times, and still deliver a high performance embedded processing solution with the Embedded Module for Blackfin Processors.

Contents

System Requirements	3
Installing the Embedded Module for Blackfin Processors	3
Installing the EZ-KIT Lite or Emulator.....	4
What's New in Version 8.6	6
New User Interface Support	6
New Targets.....	6
New Project-Level Code Generation Options.....	6
New VI-Specific Code Generation Options	7
New Blackfin-Specific VIs.....	7
Fixed-Point Support.....	7
New Preset Build Specification Configurations.....	10
Improved Build Specification Conflict Notification	10
Connection Troubleshooting Wizard	10
Getting Started with Blackfin VIs, Applications, and Elemental I/O Tutorial	11
Creating the LabVIEW Project	11
Creating the Front Panel.....	12
Creating the Block Diagram.....	13
Configuring the Target and Debugging Options	15
Editing the Build Specification	17
Building and Running the Blackfin Application	18
Debugging and Profiling Blackfin Applications	19
Using Elemental I/O.....	19
Generating a User Interface on the Target LCD	22
Where to Go from Here	23
Where to Go for Support	24

System Requirements

The Embedded Module for Blackfin Processors has the following requirements:

- A computer with Windows Vista/XP/2000
- Analog Devices VisualDSP++ 5.0 with Update 3 (included)
- LabVIEW 8.6 with embedded support (included)

Refer to the *LabVIEW Release Notes*, available by selecting **Start» All Programs»National Instruments»LabVIEW»LabVIEW Manuals** and opening `LV_Release_Notes.pdf`, for information about LabVIEW system requirements.

Installing the Embedded Module for Blackfin Processors

Complete the following steps to install VisualDSP++ 5.0 and the Embedded Module for Blackfin Processors.

1. Log in as an administrator or as a user with administrator privileges.
2. Insert the NI LabVIEW Embedded Module for ADI Blackfin Processors Installation DVD and select to install ADI VisualDSP++ 5.0. If you already have installed VisualDSP++, you do not need to reinstall.



Tip If the installer does not automatically begin, double-click `VisualDSP++5.0.exe` on the DVD to begin installation of VisualDSP++ 5.0.

3. Follow the instructions on the screen for installing VisualDSP++ 5.0. When the VisualDSP++ 5.0 installation finishes, you can install the Embedded Module for Blackfin Processors.
4. Select to install the Embedded Module for Blackfin Processors. If the installer welcome screen is not visible, double-click `setup.exe` on the DVD to begin installation of the Embedded Module for Blackfin Processors.
5. Follow the instructions on the screen. The installation DVD installs both LabVIEW 8.6 with embedded support and the Embedded Module for Blackfin Processors. The installer also updates VisualDSP++ 5.0 with the necessary updates.

6. Follow the activation instructions that appear on the screen.
You also can use the NI License Manager, available by selecting **Start»All Programs»National Instruments»NI License Manager**, to activate National Instruments products. Refer to the *National Instruments License Manager Help*, available by selecting **Help»Contents** in the NI License Manager, for more information about activating NI products.
7. Restart the computer when the installer prompts you and log on as an administrator or as a user with administrator privileges.

Installing the EZ-KIT Lite or Emulator



Caution Be careful when removing the board from the package and handling the board to avoid the discharge of static electricity, which might damage some components.

The EZ-KIT Lite or USB-based ICE board is designed to run as a stand-alone unit. You do not have to open the computer case.



Note You must install VisualDSP++ before you can install the Blackfin target. Refer to the [Installing the Embedded Module for Blackfin Processors](#) section for information about installing VisualDSP++.

Complete the following steps to install the ADSP-BF537 EZ-KIT Lite or USB-based ICE.

1. Plug the power supply for the board into a surge-protected outlet.
2. Connect the USB assembly for the board to the USB port on the host computer using the provided USB cable.

Figure 1 shows the location of the A/C adaptor and USB port on the Blackfin target. Refer to the *ADSP-BF537 EZ-KIT Lite Evaluation System Manual* in the EZ-KIT box for more detailed information about the ADSP-BF537 EZ-KIT Lite hardware.

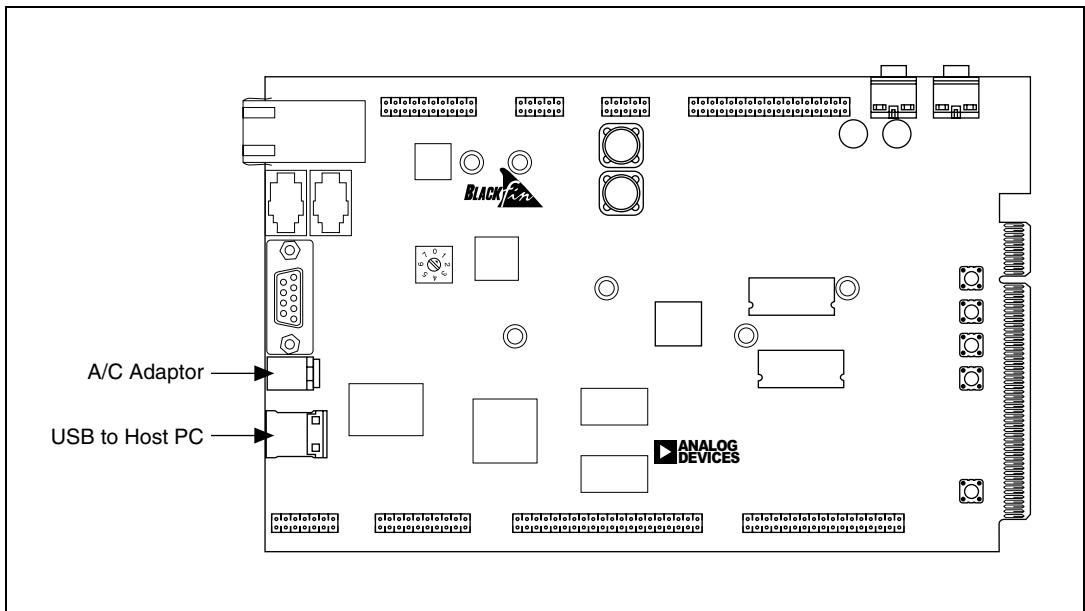


Figure 1. Locating the A/C Adaptor and USB Port on the ADSP-BF537 Target

On the board, the power LED illuminates, and you might see other visible activity, such as blinking LEDs. The connection activates the Windows Found New Hardware Wizard.

3. Follow the Found New Hardware Wizard instructions on the screen to install the software automatically. A Windows message notifies you when the new device is ready for use and the hardware installation is complete.
4. Verify that the USB monitor LED is lit. The LED is in close proximity to the USB connector. The lit LED signifies that the board is communicating properly with the host computer and is ready to run. Refer to the *ADSP-BF537 EZ-KIT Lite Evaluation System Manual* in the EZ-KIT box for more detailed information about the USB monitor LED.
5. For a USB-based ICE, attach the JTAG cable to the emulation target. Refer to the emulator documentation for more information about USB-based ICE.
6. For TCP/IP debugging, you must connect the EZ-KIT to a local network using an ethernet cable. By default, the Embedded Module for Blackfin Processors uses TCP/IP debugging. Refer to the [Editing the Build Specification](#) section for information about changing the method of debugging.

What's New in Version 8.6

The Embedded Module for Blackfin Processors includes the following new features:

New User Interface Support

In previous versions, the front panel of a Blackfin VI was only visible on the host computer. You now can use the front panel window to design a user interface for the LCD on the target if the target includes an LCD, such as the ADSP-BF527 or ADSP-BF548 EZ-KITs.

Designing user interfaces for embedded targets can be different than designing user interfaces for desktop computers running Windows because of differences in operating systems and the size of the LCD. Refer to the *LabVIEW Help* for information about creating user interfaces for Blackfin applications.

You enable user interface support in the **Build Specification** dialog box. Place a checkmark in the **Generate UI** checkbox on the **Advanced Options** page to generate the user interface on the target. If the target does not support a user interface, the **Generate UI** checkbox does not appear on the **Advanced Options** page.

New Targets

The Embedded Module for Blackfin Processors now includes target support for the following:

- ADSP-BF52x (six new targets)
- ADSP-BF561 (multi-core)

Each core of the ADSP-BF561 is a separate target in LabVIEW. Use the new Messaging VIs to pass data between the two cores.

New Project-Level Code Generation Options

You now can allocate and deallocate constants for arrays, clusters, strings, variants, and waveforms with build options on the **Application Information** page in the **Build Specification Properties** dialog box.

The allocation build options include allocating memory the first time you use constants on the block diagram, when the application is outside of the loop that contains the constants, when the VI that contains the constants is called, and when the built application begins running on the target.

The deallocation build options include deallocating memory when the constants are no longer used, when a VI containing the constants finishes executing, and when the built application finishes executing on the target.

New VI-Specific Code Generation Options

You can now optimize subVI calls and inline subVIs into callers. Optimizing subVI calls can eliminate overhead and increase code optimization in the following ways:

- **Optimize subVI calls**—Generates C code for subVI calls with as little default data initialization as possible. You cannot debug a VI with optimized subVI calls. Select **Optimize subVI calls** on the **Application Information** page in the **Build Specification Properties** dialog box.
- **Allow inlining**—Allows inlining of subVIs into callers. Inlining subVIs is most useful for small subVIs, subVIs called in a loop, or subVIs with only one call site. To inline a subVI, you must select **True** from the **Inline subVI** pull-down menu on the **Source File Settings** page in the **Build Specification** dialog box.

New Blackfin-Specific VIs

Version 8.6 contains the following palettes and VIs for the ADSP-BF52x peripherals and the ADSP-BF561 multi-core target:

- BF52x VIs for the MAX1233 touch screen and Varitronix T350MCQB01 LCD
- ADIS16003 VIs for the ADIS16003 accelerometer
- ADIS16251 VIs for the ADIS16251 gyroscope
- BF561 VIs to communicate between the ADSP-BF561 cores

Examples for using the BF561 VIs are located in `labview\examples\lvemb\Blackfin\BF561`.

Fixed-Point Support

The fixed-point data type has limited support.



Note Overflow mode is supported, but overflow status is not supported.

Supported Numeric Functions

The following **Numeric** functions support the fixed-point data type:

- Absolute Value
- Add
- Decrement

- Increment
- Multiply
- Negate
- Round To Nearest
- Round Toward +Infinity
- Round Toward -Infinity
- Scale By Power Of 2 Function
- Sign
- Subtract
- Square

Comparison Functions

The following **Comparison** functions support the fixed-point data type:

- Equal?
- Equal To 0?
- Greater Or Equal?
- Greater Or Equal To 0?
- Greater?
- Greater Than 0?
- Less Or Equal?
- Less Or Equal To 0?
- Less?
- Less Than 0?
- Not Equal?
- Not Equal To 0?

Conversion Functions

The following **Conversion** functions support the fixed-point data type:

- Boolean Array To Number
- Number To Boolean Array
- To Byte Integer
- To Double Precision Float
- To Extended Precision Float
- To Fixed-Point
- To Long Integer

- To Quad Integer
- To Single Precision Float
- To Unsigned Byte Integer
- To Unsigned Long Integer
- To Unsigned Quad Integer
- To Unsigned Word Integer
- To Word Integer

Data Manipulation Functions

The following **Data Manipulation** functions support the fixed-point data type:

- Flatten To String
- Logical Shift
- Rotate Left With Carry
- Rotate Right With Carry
- Type Cast
- Unflatten From String

String/Number Conversion Functions

The following **String/Number Conversion** functions support the fixed-point data type:

- Decimal String To Number
- Fract/Exp String To Number
- Hexadecimal String To Number
- Number To Decimal String
- Number To Engineering String
- Number To Exponential String
- Number To Fractional String
- Number To Hexadecimal String
- Number To Octal String
- Octal String To Number

New Preset Build Specification Configurations

The **Build Specification** dialog box now includes preset build configurations, such as optimizing for application speed or size. When you use one of the preset configurations, LabVIEW sets the appropriate build options automatically. You can override any of the options a preset configuration sets.

To use a preset build configuration, select a configuration from the **Run-Time Options** pull-down menu on the **Application Information** page of the **Build Specification** dialog box.

Improved Build Specification Conflict Notification

The **Build Specifications** dialog box includes a **Build settings conflicts** area at the bottom of each page. If you select build settings that are incompatible, such as parallel execution and expression folding, you receive notification in the **Build settings conflicts** listbox.

Connection Troubleshooting Wizard

Version 8.6 includes a new troubleshooting wizard that can help troubleshoot connection problems with the target. If you receive an **Unable to connect to target** error, click the **Troubleshooting** button in the error dialog box to launch the wizard.

Getting Started with Blackfin VIs, Applications, and Elemental I/O Tutorial

Use this tutorial to learn how to build, run, and debug an application for a Blackfin target. This example refers to the ADSP-BF537 target, but you also can use other Blackfin targets.

Creating the LabVIEW Project

Use LabVIEW projects to group together LabVIEW files and non-LabVIEW files, create build specifications for building a VI into a Blackfin application, and run the application on a Blackfin target. You must use a project to build Blackfin VIs into Blackfin applications.

LabVIEW project files have a `.lvproj` file extension. Project files contain target-specific build options and other information necessary for the LabVIEW C Code Generator to generate C code from the VIs.

Complete the following steps to create a project with an ADSP-BF537 target and a blank VI.

1. Launch LabVIEW.
2. Select **Blackfin Project** from the **Targets** pull-down menu in the **Getting Started** window.
3. Click the **Go** button to display the **Create New Blackfin Project** wizard.
4. Select **New Blackfin project, blank VI** in the **Project type** pull-down menu.
5. Click the **Next** button to display the **Select Blackfin target type** page.
6. Select **Analog Devices ADSP-BF537** from the **Target type** pull-down menu.
7. Click the **Next** button to display the **System preview** page.
8. Place a checkmark in the **Create a build specification** checkbox. You need to create a build specification to build a Blackfin application.
9. Click the **Finish** button.
10. Click the **Save** button when LabVIEW prompts you to save the project.
11. Click the **Yes** button when LabVIEW prompts you to save the new files in the project.
12. Save the project as `Tutorial.lvproj` when LabVIEW prompts you.
13. Save the Blackfin VI as `Tutorial.vi`.

14. Expand the Analog Devices ADSP-BF537 target in the **Project Explorer** window. LabVIEW automatically adds **Dependencies** under the target. SubVIs appear under **Dependencies** when you add a VI that contains subVIs to a project.
15. Expand the **Build Specifications** section under the Blackfin target in the **Project Explorer** window. The wizard labels the build specification **VDK Application**.
16. Rename the build specification to `Debug Build`. Complete the following steps to rename the build specification.
 - a. Right-click **VDK Application** and select **Rename** from the shortcut menu.
 - b. Enter `Debug Build` and press the <Enter> key.



Tip Most users create a debug and a release build specification for a project. For example, if you create a build specification with debug options, you can change the name to `Debug Build`. If you create a build specification with release options, you can change the name to `Release Build`. Refer to the [Editing the Build Specification](#) section for more information about using build specifications.

Creating the Front Panel

The front panel is the user interface for an embedded application or a debugging interface for Blackfin applications you create with LabVIEW. In this tutorial you create a VI with an LED indicator that lights on the front panel of the host computer if the input exceeds a threshold value you define.

Complete the following steps to create the front panel.

1. Add the following controls to the front panel window of `Tutorial.vi`:
 - Two numeric controls, located on the **Numeric** palette.
 - One numeric indicator, located on the **Numeric** palette.
 - One round LED, located on the **Boolean** palette.



Tip If you cannot find the object you want, click the **Search** button on the **Controls** palette toolbar. Type the name of the object for which you want to search. LabVIEW searches as you type and displays any matches.

2. Rename the controls by double-clicking the labels and entering new names.
 - Rename one of the numeric controls to **input**.
 - Rename the other numeric control to **threshold**.

- Rename the numeric indicator to **output**.
- Rename the round LED to **threshold exceeded?**.

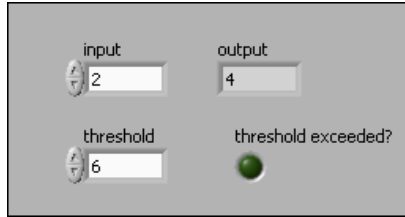


Figure 2. Changing the Labels



Tip Double-click to select a single word in a label. Triple-click to select the entire label.

Creating the Block Diagram

The block diagram is the source code for a VI and contains a pictorial description or representation of an application. Wires carry data between the objects, or nodes, on the block diagram. The controls and indicators you added in the [Creating the Front Panel](#) section appear as terminals on the block diagram.

Complete the following steps to build a block diagram that multiplies an input value by 2 and then lights an LED if the product is greater than the threshold value you specify.

1. Switch to the block diagram by clicking the block diagram if it is visible or selecting **Window»Show Block Diagram**.



Tip You also can switch to the block diagram by pressing the <Ctrl-E> keys.

2. Select **Help»Show Context Help** to display the **Context Help** window. The **Context Help** window displays basic information about LabVIEW objects when you move the cursor over each object.



Tip You also can press the <Ctrl-H> keys to open and close the **Context Help** window.

3. Place a While Loop, located on the **Structures** palette, around the controls and indicator on the block diagram. While Loops repeat the inner subdiagram until the conditional terminal receives a particular Boolean value.



4. Right-click the conditional terminal, shown at left, in the lower right corner of the While Loop and select **Create Constant** from the shortcut menu. The default Boolean constant in the While Loop is FALSE.

5. Place a **Multiply** function, located on the **Numeric** palette, on the block diagram inside the While Loop.
6. Wire the **input** control to the **x** input of the Multiply function.
7. Right-click the **y** input of the Multiply function and select **Create» Constant** from the shortcut menu.
8. Enter 2 to multiply the value of the **input** control by two.
9. Place a **Greater?** function, located on the **Comparison** palette, on the block diagram.
10. Wire the **x*y** output of the Multiply function to the **x** input of the Greater? function.
11. Wire the **threshold** control to the **y** input of the Greater? function.
12. Wire the **x > y?** output of the Greater? function to the **threshold exceeded** indicator.
13. Wire the **output** indicator to the wire connecting the Multiply function and the Greater? function.
14. Place a **Wait Until Next ms Multiple** function, located on the **Time, Dialog & Error** palette, inside the While Loop.
15. Right-click the **millisecond multiple** input and select **Create» Constant** from the shortcut menu.
16. Enter 100 to wait 100 milliseconds between loop iterations.

The block diagram should look similar to Figure 3.

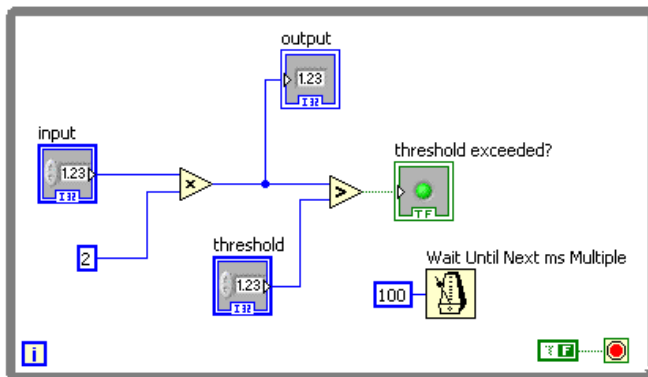


Figure 3. Creating the Block Diagram

17. Save the VI.

Configuring the Target and Debugging Options

In LabVIEW you must specify how the Blackfin target connects to the host computer using the **VisualDSP++ Target Configuration** dialog box. The EZ-KIT Lite is connected to the host computer through the USB port, which also is known as a debug agent.



Note You only have to configure the target once unless you change how you connect the target to the host computer.

Complete the following steps to configure the target options.

1. Right-click **Analog Devices ADSP-BF537** in the **Project Explorer** window and select **Configure Target** from the shortcut menu to display the **VisualDSP++ Target Configuration** dialog box, shown in Figure 4.

The **VisualDSP++ version** pull-down menu and **VisualDSP++ location** text box display the version of VisualDSP++ on the host computer.

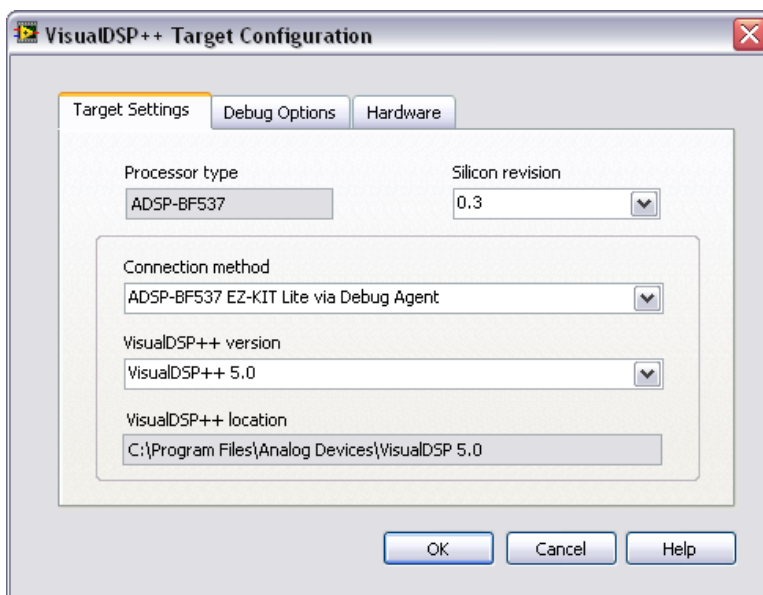


Figure 4. Configuring the Target Settings

2. Select **ADSP-BF537 EZ-KIT Lite via Debug Agent** from the **Connection method** pull-down menu.
3. Select the silicon revision in the **Silicon revision** pull-down menu that matches the silicon revision on the hardware. You can find the silicon revision printed on the Blackfin processor on the target.

- Click the **Debug Options** tab to configure the debug options you can use while debugging a Blackfin application on the Blackfin target.



Tip Click the **Help** button to open the *LabVIEW Help* and read a description of each debug setting.

- Change the **Front Panel / Probe Update Period (ms)** to 100 by moving the slider or typing 100 in the numeric control under the slider as shown in Figure 5. This setting configures how often the front panel updates with data from the Blackfin application.

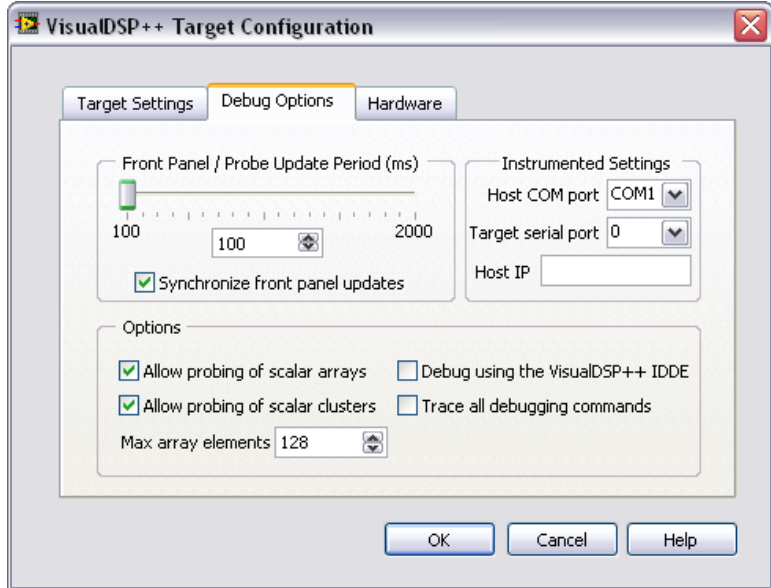


Figure 5. Configuring the Debugging Options

- Click the **OK** button to close the **VisualDSP++ Target Configuration** dialog box.
- If necessary, click the **OK** button in the dialog box that appears reminding you to rebuild the project.

Editing the Build Specification

Use build specifications to specify how the LabVIEW C Code Generator generates C code and how to build the Blackfin VI into an application.

You can have multiple build specifications for the same target. For example, you might want one build specification that generates debugging information and another build specification that does not generate this extra information.

Complete the following steps to edit the build specification you created using the **Create New Blackfin Project** wizard.

1. Right-click **Debug Build** in the **Project Explorer** window under the Blackfin target **Build Specifications** section and select **Properties** from the shortcut menu to display the **Build Specification Properties** dialog box.
2. (Optional) LabVIEW prompts you to configure the target if you previously have not configured the target. Click the **Yes** button and refer to the [Configuring the Target and Debugging Options](#) section for information about configuring the target.
3. Verify that the **Enable debugging** checkbox has a checkmark. The method in parentheses indicates the debugging method. You change the debugging method on the **Advanced Debugging Options** page.
4. Verify **No optimizations** is selected in the **Run-Time Options** pull-down menu.
5. Select **Advanced Debugging Options** in the **Category** list.
6. Select a debugging mode in the **Debugging Mode** section. By default, **TCP port** is the debugging mode. Make sure to connect the Blackfin target to the network with an Ethernet cable if you select this option. Otherwise, the application appears to run but does not.

If you want to debug using the USB cable, select **JTAG/EZ-KIT USB** in the **Debugging Mode** section.

7. Select **Communications Options** in the **Category** list.
8. If you selected **JTAG/EZ-KIT USB** in step 6, remove the checkmark from the **Enable lwIP TCP/IP support** checkbox. If you do not remove this checkmark and run the application, the target looks for a link to the network and does not continue unless it finds a DHCP address on the network.
9. Select **Source Files** from the **Category** list and verify that `Tutorial.vi` is in the **Top-level VI** text box. If `Tutorial.vi` is not in the **Top-level VI** text box, click the blue right arrow button, shown at left, to move the VI from the source files list to the **Top-level VI** text box.



When the Blackfin project contains other files, such as `.c` and `.lib` files, you can add these files to the list of files to build into the application on the **Source Files** page.

10. Click the **OK** button to close the dialog box.
11. Select **File>Save Project** in the **Project Explorer** window to save the project. Build specifications are saved with the project.

Building and Running the Blackfin Application

After you develop the Blackfin VI on the host computer, you build the VI into an application you can run on a Blackfin target. When you build a Blackfin application, the LabVIEW C Code Generator generates C code from the LabVIEW block diagram using the settings you configure.

Complete the following steps to build and run a Blackfin application.

1. Right-click **Debug Build** in the **Project Explorer** window under the Blackfin target **Build Specifications** section and select **Build** from the shortcut menu to build the Blackfin VI into an application. LabVIEW displays the status of the building and linking process.
2. Right-click **Debug Build** again and select **Debug** from the shortcut menu to download the application to the target and run the application with front panel updates on the host computer. The application automatically runs on the target when you select **Debug** from the shortcut menu.
3. In the **Processor Status** window, verify that the application is running on the target. The **Processor Status** window displays the download, connection, and execution progress of the application on the target. In addition, you can click the **Output** and **Errors** tabs to display current information about the running application.
4. Enter a value in the **threshold** numeric control of the Tutorial VI on the host computer.
5. Enter different values in the **input** numeric control. In Figure 6, the **output** value on the left does not exceed the **threshold** value. If you change the **input** value so that the **output** value is greater than the **threshold** value, the **threshold exceeded?** LED lights.

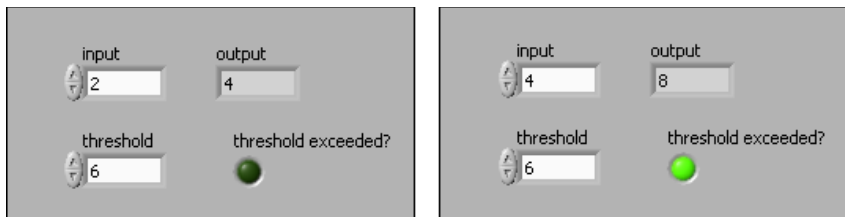


Figure 6. LED Lights when Output Exceeds Threshold



Tip LabVIEW uses default values for controls and indicators when building a Blackfin VI into a Blackfin application. To change the initial values, enter the new values in the front panel controls and then select **Edit»Make Current Values Default** to change the initial values. You must rebuild the application after you change the initial values of the controls.



6. Click the **Abort Execution** button, shown at left, to stop the Blackfin application.

Debugging and Profiling Blackfin Applications

Debugging is a real-time connection between LabVIEW and the Blackfin target. You enable debugging by placing a checkmark in the **Enable debugging** checkbox on the **Application Information** page of the **Build Specification Properties** dialog box.

The block diagram acts as a conduit between the application running on the target and the VI running on the host computer. Therefore, you can probe signals, set breakpoints, and step through code as you do in any other VI.

With debugging enabled, you also can enable profiling to display the execution times for VIs and functions in a Blackfin application. You enable profiling by placing a checkmark in the **Enable profile information** checkbox on the **Advanced Debugging Options** page of the **Build Specification Properties** dialog box. When the application terminates, the application sends the execution times to the host computer. You can use the execution times to help optimize a Blackfin application.

Refer to the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**, for more information about debugging Blackfin applications and profiling Blackfin VIs.

Using Elemental I/O

Elemental I/O resources are fixed elements of Blackfin targets that you use to transfer data among the different parts of the target. Each Elemental I/O resource has a specific type, such as digital, analog, or PWM. For example, you can use digital Elemental I/O resources to manipulate the LEDs on the target. Refer to the *LabVIEW Help* for more information about using Elemental I/O with Blackfin targets.



Note The Embedded Module for Blackfin Processors implements Elemental I/O only on BF53x targets.

The following sections describe how to use Elemental I/O to light an LED on the Blackfin target when the threshold is exceeded.

Selecting the Elemental I/O Target

You first must select the Elemental I/O set for the Blackfin target before you can add Elemental I/O items to the project. Complete the following steps to select the Elemental I/O set.

1. Right-click **Analog Devices ADSP-BF537** in the **Project Explorer** window and select **Select Elemental I/O** from the shortcut menu to display the **Select Elemental I/O** dialog box.

Use this dialog box to specify the target I/O. After you select the I/O, you can create new Elemental I/O items in the **Project Explorer** window and use Elemental I/O in a Blackfin VI.

2. Select **BF537** in the **Select Elemental I/O Device** list to select the **EZ-KIT I/O** set and click the **OK** button. After you select the I/O and click the **OK** button, the **Select Elemental I/O** dialog box becomes unavailable to the project. You must create a new project if you need to select a different I/O set for the Blackfin target.

Adding Elemental I/O Items to the Project

You must add Elemental I/O items to the project before you can use Elemental I/O in a Blackfin VI. Complete the following steps to add Elemental I/O items to the project.

1. Right-click **Analog Devices ADSP-BF537** in the **Project Explorer** window and select **New»Elemental I/O** from the shortcut menu to display the **New Elemental I/O** dialog box.
2. Expand **Digital Output** in the **Available Resources** tree.
3. Hold down the <Ctrl> key and click LED1 and LED2 to select both resources.
4. Click the **Add** button to add LED1 and LED2 to the **New Elemental I/O** list.
5. Click the **OK** button to add the Elemental I/O items to the LabVIEW project.

Many pins on the Blackfin target can have multiple configurations. For example, on the Blackfin BF537 EZ-KIT, LED1 and the pulse width modulation (PWM) timer TMR3 both use pin PF6. Therefore, you cannot use both LED1 and TMR3 in the same application.

After you add Elemental I/O items to the project, LabVIEW filters the available resources in the **New Elemental I/O** dialog box to remove resources with pin conflicts. In this example, if you right-click **Analog Devices ADSP-BF537** and select **New»Elemental I/O** from the shortcut menu, you notice that TMR3 is not available in the **Available Resources** list because you already added LED1 to the project.

Using Elemental I/O on the Block Diagram

You can use Elemental I/O on the block diagram after you add Elemental I/O items to the project. Complete the following steps to use Elemental I/O on the block diagram of the Blackfin VI to light the LEDs on the target.

1. Drag LED1 from the **Project Explorer** window to the block diagram above the **threshold exceeded?** indicator.
2. Expand the Elemental I/O Node by dragging the bottom handle until you see LED1 and LED2.
3. Wire the **x > y?** output of the Greater? function to the LED1 and LED2 items in the Elemental I/O Node.

Refer to the *Using Elemental I/O Nodes* topic in the *LabVIEW Help* for more information about using Elemental I/O Nodes.

4. Right-click the wire that connects the **x > y?** output to LED2 and select **Insert»Boolean Palette»Not** from the shortcut menu to place a Not function on the wire. Using the Not function specifies that LED1 and LED2 alternate status such that when LED1 is off, LED2 is on.

Building and Running the Application with Elemental I/O

Complete the following steps to run the Blackfin application with Elemental I/O.

1. Click the **Run** button. When you click the **Run** button, LabVIEW prompts you if you need to build the embedded application.
2. Click the **Save** button when LabVIEW prompts you to save the VI.
3. Enter different values in the **input** numeric control until the **threshold exceeded?** indicator lights on the front panel. When the **threshold exceeded?** indicator lights, LED1 on the Blackfin target also lights and LED2 turns off.
4. Click the **Abort Execution** button to stop the Blackfin application.

Generating a User Interface on the Target LCD

Some Blackfin targets, such as the ADSP-BF548, have a screen capable of handling a user interface. You can create a front panel user interface (UI) on the host computer and then download the UI to the target.



Note You cannot generate a user interface on the ADSP-BF537 target.

Complete the following steps to generate the UI on the Blackfin target.

1. Right-click **Debug Build** under the Blackfin target **Build Specifications** section and select **Properties** from the shortcut menu to display the **Build Specification Properties** dialog box.
2. Select **Advanced Debugging Options** in the **Category** list.
3. Place a checkmark in the **Generate UI** checkbox.
4. (Optional) Change the orientation of the user interface in the **Orientation** pull-down menu.
5. Scale the user interface by selecting a value other than 1 in the **Front panel scale factor** pull-down menu. Larger values make the front panel object smaller on the target user interface.
6. Click the **OK** button.
7. Right-click **Debug Build** in the **Project Explorer** window and select **Run** from the shortcut menu.
8. Click the **OK** button when prompted to rebuild the project.
LabVIEW rebuilds the application and downloads the application to the target. The user interface on the target displays the controls and indicators you created for the front panel window.
9. Use the increment and decrement controls on the user interface to change the values of the input and threshold components. The LED on the user interface lights up when the output value exceeds the threshold value.

Where to Go from Here

National Instruments provides many resources to help you succeed with your NI products. Use the following related documentation as you continue exploring LabVIEW and the Embedded Module for Blackfin Processors.

- *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help** in LabVIEW, provides information about LabVIEW programming, step-by-step instructions for using LabVIEW, and reference information about LabVIEW VIs, functions, palettes, menus, and tools. Refer to the **Embedded Module for Blackfin Processors** book on the **Contents** tab of the *LabVIEW Help* for information specific to the Embedded Module for Blackfin Processors and Blackfin applications.
- Context help provides brief descriptions of VIs and functions with a link to the complete reference for a VI or function. Select **Help»Show Context Help** to open the **Context Help** window.
- Examples are available in the `labview\examples\lvemb\Blackfin` directory and can help you get started creating Blackfin VIs.
- The readme file, available by selecting **Start»All Programs»National Instruments»LabVIEW»Readme** and opening `readme_BLACKFIN.html`, contains known issues and last-minute information.
- The *Getting Started with ADSP-BF537 EZ-KIT Lite* manual, available in the EZ-KIT box, familiarizes you with the hardware capabilities of the EZ-KIT.
- The *ADSP-BF537 EZ-KIT Lite Evaluation System Manual*, available in the EZ-KIT box, describes the operation and configuration of the board components and provides a schematic for reference.
- *Getting Started with LabVIEW* manual, available by selecting **Start»All Programs»National Instruments»LabVIEW»LabVIEW Manuals** and opening `LV_Getting_Started.pdf`, provides information about the LabVIEW graphical programming environment and the basic LabVIEW features you use to build data acquisition and instrument control applications.

Where to Go for Support

The National Instruments Web site is your complete resource for technical support. At ni.com/support you have access to everything from troubleshooting and application development self-help resources to email and phone assistance from NI Application Engineers.

National Instruments corporate headquarters is located at 11500 North Mopac Expressway, Austin, Texas, 78759-3504.

National Instruments also has offices located around the world to help address your support needs. For telephone support in the United States, create your service request at ni.com/support and follow the calling instructions or dial 512 795 8248. For telephone support outside the United States, contact your local branch office:

Australia 1800 300 800, Austria 43 662 457990-0,
Belgium 32 (0) 2 757 0020, Brazil 55 11 3262 3599,
Canada 800 433 3488, China 86 21 5050 9800,
Czech Republic 420 224 235 774, Denmark 45 45 76 26 00,
Finland 358 (0) 9 725 72511, France 01 57 66 24 24,
Germany 49 89 7413130, India 91 80 41190000, Israel 972 3 6393737,
Italy 39 02 41309277, Japan 0120-527196, Korea 82 02 3451 3400,
Lebanon 961 (0) 1 33 28 28, Malaysia 1800 887710,
Mexico 01 800 010 0793, Netherlands 31 (0) 348 433 466,
New Zealand 0800 553 322, Norway 47 (0) 66 90 76 60,
Poland 48 22 3390150, Portugal 351 210 311 210, Russia 7 495 783 6851,
Singapore 1800 226 5886, Slovenia 386 3 425 42 00,
South Africa 27 0 11 805 8197, Spain 34 91 640 0085,
Sweden 46 (0) 8 587 895 00, Switzerland 41 56 2005151,
Taiwan 886 02 2377 2222, Thailand 662 278 6777,
Turkey 90 212 279 3031, United Kingdom 44 (0) 1635 523545

National Instruments, NI, ni.com, and LabVIEW are trademarks of National Instruments Corporation. Refer to the *Terms of Use* section on ni.com/legal for more information about National Instruments trademarks. Other product and company names mentioned herein are trademarks or trade names of their respective companies. For patents covering National Instruments products/technology, refer to the appropriate location: **Help»Patents** in your software, the `patents.txt` file on your media, or the *National Instruments Patent Notice* at ni.com/patents.