Getting Started with the LabVIEW™ Embedded Module for ARM Microcontrollers 1.0
For the Keil MCB2300

The LabVIEW Embedded Module for ARM Microcontrollers is a comprehensive graphical development environment for embedded design. Jointly developed by Keil—An ARM Company and National Instruments, this module seamlessly integrates the LabVIEW graphical development environment and ARM microcontrollers. You can lower development costs and achieve faster development times by using the Embedded Module for ARM Microcontrollers to program ARM targets.

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 ARM Microcontrollers, you can optimize linking and view live front panel updates using JTAG, serial, or TCP/IP. The Embedded Module for ARM Microcontrollers includes the LabVIEW C Code Generator, which generates C code from the LabVIEW block diagram.

This manual includes system requirements, installation instructions, and a step-by-step tutorial that shows you how to build, run, and debug an ARM application.

Contents

Features ................................................................................................... 2
System Requirements.............................................................................. 2
Installing the Embedded Module for ARM Microcontrollers .......... 3
Installing the MCB2300 Evaluation Board......................................... 4
Tutorial for the Embedded Module for ARM Microcontrollers .......... 6
Creating the LabVIEW Project.......................................................... 6
Creating the Front Panel ................................................................. 8
Creating the Block Diagram ............................................................... 8
Editing the MCB2300 Build Specification........................................ 10
Features

The Embedded Module for ARM Microcontrollers includes the following features:

- **RealView Microcontroller Development Kit**—Includes the Keil µVision3 integrated development environment (IDE) and debugger, as well as the ARM RealView Compilation Tools.
- **Elemental I/O support**—Takes advantage of the Elemental I/O framework to access analog input, analog output, digital I/O, and PWM abilities on the ARM target.
- **VI-level C code generation options**—Allow you to set code generations options for individual VIs.
- **Expression folding**—Generates better-performing and more efficient code by collapsing groups of nodes into single expressions that C compilers easily recognize.

Refer to the *LabVIEW Help*, available by selecting Help»Search the LabVIEW Help in LabVIEW, for more information about using the Embedded Module for ARM Microcontrollers.

System Requirements

The Embedded Module for ARM Microcontrollers has the following requirements:

- A desktop computer with Windows Vista/XP/2000
- RealView Microcontroller Development Kit including Keil µVision3
- LabVIEW 8.5.1 with embedded support
- Keil ULINK2 USB-JTAG adaptor

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 development system requirements.
Installing the Embedded Module for ARM Microcontrollers

The Embedded Module for ARM Microcontrollers installer includes LabVIEW 8.5.1 with embedded support. If you have LabVIEW 8.5.x already installed, you can install LabVIEW with embedded support without first uninstalling LabVIEW 8.5.x. However, you must install the RealView Microcontroller Development Kit before you install the Embedded Module for ARM Microcontrollers.

Complete the following steps to install the RealView Microcontroller Development Kit and the Embedded Module for ARM Microcontrollers.

1. Log in as an administrator or as a user with administrator privileges.
2. Insert the LabVIEW Embedded Module for ARM Microcontrollers installation DVD and select to install the RealView Microcontroller Development Kit.

   **Tip** If the installer does not automatically begin, double-click MDK_LV.exe on the DVD to begin installation of the RealView Microcontroller Development Kit.

3. Follow the instructions on the screen for installing the RealView Microcontroller Development Kit.
4. Activate the Keil µVision License ID Code (LIC). Complete the following steps to activate the LIC.
   a. Launch Keil µVision by selecting Start> All Programs> Keil uVision3.
   b. Select File> License Management to display the License Management dialog box.
   c. Click the Help button to open the ARM Development Tools help file.
   d. Follow the instructions for obtaining a single-user license. You need an internet connection and a product serial number (PSN) to activate the license. The PSN is an alphanumeric value located on the Certificate of Ownership or license card included with purchased products.
   e. After you add the LIC to the License Management dialog box, click the Close button to close the dialog box.
   f. Exit Keil µVision before installing the Embedded Module for ARM Microcontrollers.

Refer to the Keil Web site at www.keil.com/license for more information about activating Keil µVision.
5. If the installer welcome screen is still visible, select to install the Embedded Module for ARM Microcontrollers. If the installer welcome screen is not visible, double-click setup.exe on the DVD to begin installation of the Embedded Module for ARM Microcontrollers.

6. Follow the instructions on the screen for installing the Embedded Module for ARM Microcontrollers. The installation DVD installs both LabVIEW with embedded support and the Embedded Module for ARM Microcontrollers.

   (Luminary Micro EK-LM3S8962) On the Features page of the installer, select to install the Luminary Micro Driver for EK-LM3S8962 to install the drivers for the Luminary Micro evaluation board. A Software Installation alert might appear during the driver installation. Click the Continue Anyway button to continue with the installation.

7. 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.

8. Restart the computer when the installer prompts you and log in as an administrator or as a user with administrator privileges.

Installing the MCB2300 Evaluation Board

You need the following items to use the MCB2300 evaluation board with JTAG emulation.

- MCB2300 board
- An IBM-compatible PC with two unused USB ports: one to supply power to the MCB2300 board and the other to perform ULINK2 USB-JTAG downloading and debugging
- ULINK2 USB-JTAG adaptor
- Two USB serial cables, each no longer than 10 feet

Refer to the hardware documentation for required accessories such as cables and adaptors.

⚠ 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.
Complete the following steps to install the MCB2300 board. You do not have to open the computer case to install the board.

1. Verify that you have Keil µVision3 installed. µVision is a part of the RealView Microcontroller Development Kit.

You can look for the Keil\uv3 directory on the hard disk or select Start»All Programs and locate the shortcut to Keil µVision3. Do not launch µVision3 from the shortcut if you are going to use LabVIEW.

Refer to the Installing the Embedded Module for ARM Microcontrollers section for information about installing the RealView Microcontroller Development Kit.

2. Connect the ULINK2 USB-JTAG adaptor to a USB port on the host computer.

If this is the first time connecting the ULINK2 USB-JTAG adaptor to the computer, the connection activates the Windows Found New Hardware icon in Windows. A Windows message notifies you when the new device is ready for use and the hardware installation is complete.

3. Connect the ULINK2 USB-JTAG adaptor to the JTAG connector on the MCB2300 board.

Figure 1 shows the location of the JTAG connector and other parts on the MCB2300.

![Figure 1. Locating Parts for the MCB2300 Installation](image)

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>USB Connector</td>
<td>3</td>
<td>JTAG Connector</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Power LED</td>
<td>4</td>
<td>Potentiometer (Analog Input AD0)</td>
<td></td>
</tr>
</tbody>
</table>
4. Connect the USB connector on the MCB2300 board to a USB port on the host computer. This USB connection provides power to the MCB2300 board. On the board, the power LED illuminates.

**Note** The MCB2300 board remembers the last program that ran because you must program the flash memory on the board to run an application. Therefore, the MCB2300 board begins running the last application as soon as the board receives power. You must download a new application to change the start-up behavior of the board.

5. Verify that jumpers J9 and J10 are off if you plan to use the COM0 port. To use the COM0 port with applications that LabVIEW creates, remove the jumpers on J9 and J10. Refer to the jumper settings configuration topic in the *MCB2300 User’s Guide*, available by navigating to Keil\ARM\Hlp and opening mcb2300.chm, for information about configuring jumpers for other programming utilities, such as Flash Magic.

---

**Tutorial for the Embedded Module for ARM Microcontrollers**

Use this tutorial to learn how to build, run, and debug an application for the ARM target.

**Creating the LabVIEW Project**

Use LabVIEW projects to group together LabVIEW files and non-LabVIEW files, create build specifications for building ARM VIs into ARM applications, and run the applications on ARM targets. You must use a project to build ARM VIs into ARM 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 MCB2300 target and a blank VI.

1. Launch LabVIEW.
2. Select **ARM Project** from the **Targets** pull-down menu in the **Getting Started** window.
3. Click the **Go** button to display the **Create New ARM Project** wizard.
4. Select **New ARM project, blank VI** in the **Project type** pull-down menu.
5. Click the **Next** button to display the **Select ARM target type** page.
6. Select MCB2300 from the Target type pull-down menu.

7. Click the Next button to display the System preview page.

8. Verify the Create a build specification checkbox contains a checkmark.

9. (Optional) Place a checkmark in the Run on simulator checkbox if you want to use the µVision simulator instead of the MCB2300 evaluation board to run the application.

10. Click the Finish button.

11. Click the Save button when LabVIEW prompts you to save the project.

12. Click the Yes button when LabVIEW prompts you to save the new files in the project.

13. Save the project as Tutorial.lvproj when LabVIEW prompts you.

14. Save the ARM VI as Tutorial.vi.

15. (Optional) Save the simulated I/O VI as Simulated IO if you chose to use the µVision simulator in step 9.

   The project now appears in the Project Explorer window.

16. Expand the MCB2300 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.

17. Expand the Build Specifications section under the MCB2300 target in the Project Explorer window. The wizard labels the build specification Application.

18. Rename the build specification to Debug Build. Complete the following steps to rename the build specification.

   a. Right-click 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 MCB2300 Build Specification section for more information about using build specifications.
Creating the Front Panel

The front panel is the user interface for a VI. You can use the front panel as a debugging interface for ARM applications you create with LabVIEW. In this tutorial you create a VI with an LED indicator that lights on the front panel if the input exceeds a threshold value you define.

Complete the following steps to create the front panel for this tutorial.

1. Add the following controls to the front panel window of the 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 in the search results text box.

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?.

   **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](image)

17. Save the VI.

**Editing the MCB2300 Build Specification**

Use build specifications to specify how the LabVIEW C Code Generator generates C code and how to build the ARM 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 create a build specification for this tutorial.

1. Right-click the Debug Build build specification in the Project Explorer window and select Properties from the shortcut menu to display the Build Specification Properties dialog box.

2. Select the Application Information category, if necessary.

3. Verify that the Enable debugging checkbox contains a checkmark.

4. Select Run on target using ULINK2 in the Debug Options section to run the application on the target, or select Run on host computer using simulator to run the application on the host computer.

5. Select the Source Files category and verify that Tutorial.vi is in the Top-level VI text box. Click the blue right arrow button, shown at left, to move a VI from the source files list to the Top-level VI text box.
When the ARM 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.

6. Click the OK button to close the dialog box.

7. Select File»Save All in the Project Explorer window to save the build specification with the project.

Building and Running the ARM Application

After you develop the ARM VI on the host computer, you build the ARM VI into an application you can run on an ARM target. When you build an ARM 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 an ARM application.

1. Right-click Debug Build in the Project Explorer window and select Build from the shortcut menu to build the ARM VI into an application. LabVIEW displays the status of the building and linking process.

   Note You must activate the Keil µVision License ID Code (LIC) before you can build an ARM application with LabVIEW. If the LIC is not activated, you receive an error when you try to build the application. Refer to the Activating the Keil µVision License ID Code Readme, available by selecting Start»All Programs»National Instruments»LabVIEW»Readme and opening readme_ARM_uVision_Licensing.html, for information about activating the LIC.

2. Right-click Debug Build again and select Debug from the shortcut menu to download the application to the ARM target and run the application with front panel updates. The application automatically runs on the ARM target when you select Debug from the shortcut menu.

   Note Click the OK button if a dialog box appears notifying you about an updated µVision template.

3. Enter a value in the threshold numeric control of the Tutorial VI on the host computer.

4. Enter different values in the input numeric control. In Figure 4, 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 4. LED Lights when Output Exceeds Threshold

Tip  LabVIEW uses default values for controls and indicators when building an ARM VI into an ARM 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 ARM application after you change the initial values of the controls.

5. Click the Abort Execution button, shown at left, to stop the ARM application.

Debugging with Breakpoints and Probes

Complete the following steps to debug the ARM tutorial application with breakpoints and probes.

1. Switch to the block diagram if it is not visible.
2. Right-click the Multiply function and select Set Breakpoint from the shortcut menu. The breakpoint is highlighted with a red border around the function. This breakpoint specifies to pause execution just before the function executes. If you are using JTAG for debugging, LabVIEW might prompt you to halt the processor.
3. Right-click Debug Build in the Project Explorer window and select Debug from the shortcut menu. LabVIEW prompts you to save changes to the VI. LabVIEW also prompts you if you need to rebuild or redownload the ARM application to the ARM target.

The ARM tutorial application begins running on the ARM target. When the application reaches the breakpoint during execution, the ARM target halts all operation, the application pauses, and the Pause button, shown at left, appears red and changes to a Continue button.

4. Add probes to see the values on the wires coming into the Multiply function.
   a. Click the wire coming into the x input.
   b. Click the wire coming into the y input.

A floating Probe window appears after you create each probe. LabVIEW numbers the Probe windows automatically and displays the same number in a glyph on the wire you click.
5. Enter a different value in the **input** numeric control.

6. Click the **Continue** button, shown at left, and enter different values in the **input** numeric control to see the value in the first **Probe** window change as the ARM application executes additional iterations of the **While Loop**.

7. Click the **Step Over** button, shown at left, to execute the Multiply function and pause at the **Greater?** function, which blinks when it is ready to execute.

8. Continue clicking the **Step Over** button to step through the rest of the block diagram.

9. Click the **Abort Execution** button to stop the application.

10. Right-click the Multiply function and select **Clear Breakpoint** from the shortcut menu to remove the breakpoint.

**Using Elemental I/O**

Elemental I/O resources are fixed elements of ARM 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 ARM target. Refer to the *LabVIEW Help* for more information about using Elemental I/O with ARM targets.

The following sections describe how to use Elemental I/O to light an LED on the ARM target when the threshold is exceeded.
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 an ARM VI. Complete the following steps to add Elemental I/O items to the project.

1. Right-click MCB2300 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 ARM target can have multiple configurations. For example, on the MCB2300 board, LED1 and PWM2 both use the same pin. Therefore, you cannot use both LED1 and PWM2 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 MCB2300 and select New » Elemental I/O from the shortcut menu, notice that PWM2 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 ARM 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.


   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

Note Before you can see the LEDs light up on the target, verify that LabVIEW is downloading to the target and not to the simulator. In the Build Specification Properties dialog box, select Run on target using ULINK2 in the Debug Options section of the Application Information page to specify the target and not the simulator.

Complete the following steps to run the ARM 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. Click the Yes button when LabVIEW prompts you to rebuild the embedded application.
4. 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 ARM target also lights and LED2 turns off.
5. Click the Abort Execution button to stop the ARM application.

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 ARM Microcontrollers.


- 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 display the Context Help window.

- Examples, available in the labview\examples\lvemb\ARM directory, can help you get started creating applications.
The readme file, available by selecting Start ▶ All Programs ▶ National Instruments ▶ LabVIEW ▶ Readme and opening readme_ARM.html, contains known issues and last-minute information.

*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.